

# Telemetry & Monitoring

- [Built-in Telemetry Types](#)
- [Monitoring Channel Utilization](#)

# Built-in Telemetry Types

Meshtastic nodes broadcast telemetry data alongside messages. Understanding what each telemetry type reports helps you configure nodes correctly and interpret network monitoring data. Note that all telemetry below (battery, temperature, position, and the rest) is broadcast on the channel and, on any channel with an MQTT uplink, is republished to the broker - disable telemetry types you do not need on sensitive nodes.

## Device Metrics

Broadcast by all nodes automatically. Includes:

- **Battery voltage** - raw voltage of the battery pack
- **Battery percentage** - estimated state of charge
- **Air utilization TX** (`airUtilTx`) - rolling percentage of time this node's own radio spent transmitting, averaged over a moving window. This is a smoothed running metric, not an instantaneous reading, and should not be equated directly with a regulatory duty-cycle limit.
- **Channel utilization** - percentage of time the channel was occupied by any transmission heard by this node (includes all other nodes' traffic)

## Environment Telemetry (requires I2C sensor)

Available when a supported sensor board is connected via I2C:

- **Temperature** (°C or °F)
- **Humidity** (relative humidity %)
- **Barometric pressure** (hPa)
- **Gas resistance** - BME68x family (BME680 / BME688); a proxy for air quality / VOC concentration

Supported sensors include BME280, BME68x (BME680/BME688), SHT31, and others. Enable in app: **Radio Config → Telemetry → Environment Telemetry**.

# Position

- **GPS coordinates** (latitude, longitude)
- **Altitude** (metres MSL)
- **Speed** (km/h)
- **Heading** (degrees true)

Position is broadcast on the primary channel. Disable or reduce frequency for privacy or power savings: **Radio Config** → **Position** → **Position Broadcast Interval**.

# Power Metrics

For nodes with external power monitoring hardware:

- **Voltage** on external rails
- **Current** draw

Useful for solar-powered nodes where you want to monitor panel output and battery charge current over the mesh.

# Configuring Telemetry Intervals via CLI

Set the device telemetry broadcast interval (in seconds). The firmware default is 1800 seconds (30 minutes):

```
meshtastic --set telemetry.device_update_interval 1800
```

The default is 1800 s (30 min). Reduce toward 900 (15 min) for more frequent updates - but note that shorter intervals increase channel utilization and airtime on a busy mesh - or increase to 3600 (1 hr) / 7200 (2 hr) or more to reduce channel load on congested networks.

Set environment telemetry interval:

```
meshtastic --set telemetry.environment_update_interval 1800
```

# Enable/Disable via App

In the [Meshtastic app](#): **Radio Config** → **Telemetry**. Each telemetry type (device, environment, power) can be independently enabled or disabled and its interval set.

# Monitoring Channel Utilization

Channel utilization is the single most important metric for diagnosing a congested Meshtastic network. High channel utilization causes missed messages, failed relays, and poor network performance.

## What Channel Utilization Measures

Channel utilization (reported as a percentage in device metrics telemetry) measures the fraction of time that the radio channel is occupied by any LoRa transmission audible to a given node, averaged over a rolling 1-minute window (the firmware sums on-air milliseconds across six 10-second sub-windows). This includes:

- The node's own transmissions
- Relay transmissions from neighboring nodes
- Position and telemetry broadcasts from all nearby nodes

## The 25% Warning Threshold

A commonly used community rule of thumb is to treat **25% channel utilization** as a warning threshold — it also matches the firmware's own behavior, which starts delaying transmissions above roughly 25% utilization (the app's green/orange band boundary). Above this level:

- Packet collision probability increases significantly
- Message delivery reliability decreases
- Effective network throughput drops despite higher raw utilization

## Diagnosing High Channel Utilization

If channel utilization is above 25%, work through these checks:

1. **Count Router/Repeater nodes** - Too many infrastructure-role nodes in one area creates excessive relay traffic. Audit whether all Router/Repeater nodes are genuinely needed.
2. **Check telemetry intervals** - Frequent device metrics, position, or environment telemetry from many nodes adds up quickly. Increase intervals across the network.
3. **Identify high-traffic nodes** - Look at air utilization (the TX percentage) in device metrics. A node with very high air utilization is generating a disproportionate share of traffic.
4. **Switch dense-area personal nodes to Client Mute** - Each non-Mute client node attempts relay, multiplying traffic.

# Switching to a Faster Modem Preset

LoRa modem presets trade range for throughput. A faster preset carries the same data in less airtime, directly reducing channel utilization:

```
meshtastic --set lora.modem_preset LONG_FAST # default; good range, moderate speed
meshtastic --set lora.modem_preset MEDIUM_FAST # shorter range, faster; less airtime
meshtastic --set lora.modem_preset SHORT_FAST # short range, maximum speed
```

Switching from LONG\_FAST (SF11, ~1.07 kbps) to MEDIUM\_FAST (SF9, ~3.52 kbps) increases the data rate roughly 3×, cutting per-packet airtime to roughly a third for the payload portion (less once fixed preamble/header overhead is included), substantially reducing channel utilization. The trade-off is reduced range. **Important:** changing the modem preset must be coordinated across the entire network — a single node on a different preset goes deaf to the rest of the mesh. Use this when you have good node density and congestion is the problem.

# Viewing Channel Utilization

In the [Meshtastic app](#): **Node Details** → **Device Metrics** shows the reported channel utilization from each node's perspective. Check several nodes across the network to understand the overall picture - a node at the center of a dense cluster will see higher utilization than one on the fringe.

Via CLI:

```
meshtastic --info
```

The output includes current channel utilization and air utilization for the connected node (under `deviceMetrics.channelUtilization`). Note that channel utilization is a live telemetry metric read via `--info`, not a config key — `meshtastic --get channel_utilization` will not return it.