ROUTER vs ROUTER_CLIENT vs REPEATER Role: Deep Dive

Meshtastic provides several device roles for infrastructure nodes that exist to extend network reach rather than serve end users. On current firmware the relevant roles are **ROUTER**, **ROUTER_LATE**, and **REPEATER** (with **CLIENT** being the right choice for the overwhelming majority of ordinary nodes, since CLIENT nodes already rebroadcast via managed flooding). Choosing the wrong role wastes resources, creates unnecessary air-time, or silently breaks the capabilities operators expect. This page dissects each role at the firmware level so you can make an informed decision for every node you deploy.

Important deprecation notice. The **ROUTER_CLIENT** role referenced in this page's title was **retired in firmware 2.3.15** and is no longer a selectable `device.role` value. Do not attempt to set it. If you need a node that both relays and is used interactively by its operator, use **CLIENT** (which already relays) or **ROUTER**. Separately, the **REPEATER** role has itself been **deprecated as of firmware ~2.7.x**; for new infrastructure prefer **ROUTER** or **ROUTER_LATE**. The valid current `device.role` values are CLIENT, CLIENT_MUTE, CLIENT_HIDDEN, ROUTER, ROUTER_LATE, REPEATER, TRACKER, SENSOR, TAK, TAK_TRACKER, and LOST_AND_FOUND. The historical ROUTER_CLIENT material below is retained for context only and clearly marked as legacy.

The Infrastructure Roles at a Glance

Capability	ROUTER	ROUTER_LATE	REPEATER (deprecated)
Rebroadcasts received packets (once, with higher priority than CLIENT roles)	Yes	Yes (rebroadcasts last, after other nodes)	Yes
Power-saving sleep behaviour	Forced on automatically (ESP32); cannot be disabled	Configurable via `power.is_power_saving`	Does not force sleep by default
Visible in the node list / sends NodeInfo	Yes	Yes	No (anonymous relay, sends no NodeInfo)

Capability	ROUTER	ROUTER_LATE	REPEATER (deprecated)
App connectivity (BLE / WiFi / Serial) by default	Off by default	On (client-style)	Off
Relative overhead	Standard	Standard	Minimal overhead (per docs)

All of ROUTER, ROUTER_LATE, and REPEATER rebroadcast each packet *once* with higher priority than ordinary CLIENT nodes. The retired ROUTER_CLIENT role has been omitted from this table; it is not settable on current firmware.

ROUTER Role

A **ROUTER** node is the canonical fixed-infrastructure role, designed for stationary, well-placed nodes. When you set a node to ROUTER it adjusts its behavior in the following ways:

- ROUTER automatically enables power-saving sleep (ESP32) via `power.is_power_saving`, and this **cannot be turned off**. The LoRa radio stays in standby to wake on incoming packets — it is not a node that ignores power management.
- It defaults BLE / WiFi / Serial connectivity **off**, so you don't normally connect to it to text directly.
- Received packets are rebroadcast once, with higher priority than CLIENT roles, subject to duplicate-detection and the hop counter.
- The device **is visible in the node list** and sends its own `NodeInfo` (and position, if configured) so other nodes know it exists and can display it on the map.
- A ROUTER relays standard packets, including traceroute responses, like any non-mute node. (NeighborInfo is a separate, optional module any role can enable — it is not a defining trait of the ROUTER role.)
- A ROUTER *can* still originate and receive messages at the protocol level. What's true is that its app connectivity (BLE/WiFi/Serial) is off by default, so in current firmware it does not present an interactive messaging UI and you typically won't have an app connected to send from it. This is a connectivity default, not a removed "Send" feature — the firmware does not omit the send code path. If interactive messaging at the site is needed, use CLIENT or ROUTER_LATE rather than ROUTER.

Setting the ROUTER role via CLI

```
meshtastic --set device.role ROUTER
```

Verifying the role was applied

```
meshtastic --get device.role
```

The `--get` verb is the standard counterpart to `--set` for reading any config field in the Python CLI.

ROUTER_LATE Role

ROUTER_LATE is the modern infrastructure role for a node that should rebroadcast *only after* other nodes have had a chance to, rather than with top priority. It is useful within a local cluster where you want a relay that fills gaps without dominating airtime. Unlike ROUTER, it does not force power-saving sleep on, and it keeps client-style app connectivity available. For most stationary backbone infrastructure prefer ROUTER; choose ROUTER_LATE for nodes that should defer to the rest of the mesh first.

ROUTER_CLIENT Role (retired — legacy reference only)

ROUTER_CLIENT was retired in firmware 2.3.15 and is not a settable role on current firmware. Historically it was described as a "superset of ROUTER" that retained ROUTER's relay behaviour while also letting the node originate messages as a client. This is preserved here only for operators encountering older documentation or very old firmware:

- It was described as allowing text messages to be composed and sent from the device. On current firmware, if you need a relay that an operator can also message from, use **CLIENT** (which relays via managed flooding).
- It was described as broadcasting position from the device's own GPS or configured coordinates. That position-broadcast behaviour now applies to CLIENT-family roles.
- It was promoted for an operator who wanted to use the node interactively during events or emergency activations without compromising the relay function. Today, use a **CLIENT** node for that. (Note: Meshtastic carries text, position, and telemetry — it does not carry voice — so "voice-channel communication" was never accurate terminology.)

Do not use the legacy command below on current firmware — it will fail because ROUTER_CLIENT is no longer an accepted `device.role` value. For a relay that an operator also uses interactively, set `device.role CLIENT` instead.

Legacy ROUTER_CLIENT CLI (no longer valid)

```
# Retired in firmware 2.3.15 — this command fails on current firmware.
# Use:  meshtastic --set device.role CLIENT
# (legacy:  meshtastic --set device.role ROUTER_CLIENT)
```

REPEATER Role (deprecated)

The **REPEATER** role is the most minimal relay option, designed for nodes that should retransmit packets with minimal overhead and produce no broadcasts of their own. **Note: REPEATER was deprecated as of firmware ~2.7.x; for new infrastructure consider ROUTER or ROUTER_LATE.** Its documented behaviour:

- Received packets are rebroadcast once, with higher priority than CLIENT roles — similar to ROUTER in the act of relaying.
- It is **not visible in the node list** and does *not* send `NodeInfo` — it is effectively anonymous on the mesh. It will not appear in the node list of clients that have not heard it directly.
- It does not originate its own broadcasts (telemetry, position, NodeInfo); per the docs it "only responds to other nodes' packets instead of originating messages."
- Per the docs it rebroadcasts with minimal overhead. (It does *not* force power-saving sleep, unlike ROUTER.)

The anonymity of the REPEATER role is a deliberate design choice: relay nodes that aren't individually addressed produce less management overhead on the mesh. Because a REPEATER is not in the node list, you generally cannot select or ping it by node ID in the app the way you can a ROUTER.

Setting the REPEATER role via CLI

```
meshtastic --set device.role REPEATER
```

Hop Count Handling Across Roles

All relaying roles participate in hop-count decrement identically. When a packet arrives with a remaining hop count of N, the forwarding node decrements it to N-1 before rebroadcasting. If N is already 0, the packet is consumed locally but not forwarded. ROUTER and REPEATER have rebroadcast *priority* (they may rebroadcast sooner, and defer less to other rebroadcasters) but they still decrement and honour `hop_limit` — there is no hop-counter bypass or "always forward regardless" mode.

The practical consequence is that placing a relay mid-path does use a hop. Default `hop_limit` is **3** and the maximum is **7**; "really, 3 is fine" per the docs. Ensure the number of relay hops between any two endpoints does not exceed your configured `hop_limit`. Raising `hop_limit` to "fix" dropped messages can worsen congestion (more airtime and collisions), so pair any increase with a check on channel utilisation.

Choosing the Right Role

Community fixed repeater on a hilltop or tower

Use **ROUTER**. The node should be visible to the community (appears in node list, sends NodeInfo, relays traceroutes) but should not generate user traffic of its own. Operators can still access the serial console locally for configuration. For a node that should defer to the rest of the mesh before rebroadcasting, consider **ROUTER_LATE**.

```
meshtastic --set device.role ROUTER
```

Ham operator's home station that also relays

Use **CLIENT**. CLIENT nodes already rebroadcast via managed flooding, so a home station set to CLIENT both relays and lets you participate in mesh conversations — sending alerts, coordinating with your community, or running net check-ins from the same hardware. (The old advice to use ROUTER_CLIENT here is obsolete; ROUTER_CLIENT was retired in 2.3.15.)

```
meshtastic --set device.role CLIENT
```

Minimal-overhead anonymous relay

Historically **REPEATER** was the choice for a small node tucked into a building to bridge two otherwise disconnected areas with no topology visibility. Because REPEATER is deprecated as of firmware ~2.7.x, prefer **ROUTER** (or **ROUTER_LATE**) for new deployments; only use REPEATER if you specifically need an anonymous relay and understand it is being phased out.

```
meshtastic --set device.role ROUTER   # preferred; REPEATER is deprecated
```

Power Consumption Comparison

Idle power profiles differ by role. A **REPEATER** keeps the radio active and does not force sleep, whereas a **ROUTER** uses forced power-saving sleep (ESP32) that cannot be disabled — so their idle draw is not the same. The differences otherwise arise from background processing:

- **REPEATER** has minimal overhead per the official docs, because it skips originating broadcasts and node-presence maintenance. (We do not cite a specific kilobyte heap figure, as no firmware source quantifies it.)
- **ROUTER** forces power-saving sleep on (ESP32); **ROUTER_LATE** leaves sleep configurable via `power.is_power_saving`. If a GPS is active, GPS module current (which varies widely by module — consult the module's datasheet) can dominate other draw.
- LoRa TX current is board- and TX-power-dependent. For an SX1262-class radio alone, TX is roughly ~118 mA at +22 dBm and RX is roughly ~5 mA (Semtech SX1262 datasheet); boards with an external PA draw more, and total board draw (including the MCU) is higher than the radio alone. Treat any single current figure as approximate and board-dependent.
- Power-saving sleep is broadly available via `power.is_power_saving` to most roles (all except TRACKER and SENSOR). It is **not** exclusive to CLIENT: ROUTER forces it on automatically, while REPEATER does not sleep by default.

Quick CLI power-related settings for infrastructure nodes

```
# Disable Bluetooth to save power (figure is approximate; verify against the Bluetooth config
docs)
meshtastic --set bluetooth.enabled false

# Reduce the screen on-time to save power. Note: the documented field is
```

```
display.screen_on_secs.
# A value of 0 is the 10-minute DEFAULT, not "off"; set a small nonzero value to dim sooner.
meshtastic --set display.screen_on_secs 10

# Set a lower TX power only if nodes are nearby (reduces TX draw and channel utilisation).
# tx_power 0 is the default and means "use the region-legal maximum." Never set a higher fixed
# value than your region/antenna combination legally allows (note the >6 dBi gain-reduction
rule).
meshtastic --set lora.tx_power 17
```

Revision #4
Created 2026-05-03 05:50:04 UTC by Mesh America Admin
Updated 2026-06-09 00:22:00 UTC by Mesh America Admin