MeshCore-Specific FAQ

- What is the difference between a Repeater and Room Client in MeshCore?
- How do I connect to a MeshCore room server from the app?
- Can I run MeshCore and Meshtastic simultaneously on the same hardware?

What is the difference between a Repeater and Room Client in MeshCore?

What Is the Difference Between a Repeater and Room Client in MeshCore?

MeshCore ships several firmware variants. This page compares the **Repeater** firmware with the **Companion (room) client** firmware - the variant a user-carried node runs to connect to a room server. (Note: across the rest of this wiki this client is referred to as the **Companion** variant; "Room Client" here means the same Companion firmware used to interact with a room server. The room server itself - the node that actually stores and serves message history - runs separate Room Server firmware and is not covered in detail here.) Choosing the wrong one is a common source of confusion. This page explains both in detail.

REPEATER_FIRMWARE

A node flashed with REPEATER_FIRMWARE acts as pure RF infrastructure. Its job is to forward MeshCore packets toward their destination to extend the reach of the mesh. Unlike flood-based mesh systems, a MeshCore repeater forwards **selectively** (route/path-based) - it does not rebroadcast every packet it receives. Key characteristics:

- **No screen required:** Repeater firmware is designed to run on minimal hardware - a bare ESP32 or nRF52 board with a LoRa module. It has no UI and does not need a display.
- **Minimal power consumption:** Because it does not maintain Bluetooth, Wi-Fi, or app connections, a repeater node consumes significantly less power than a room client. This makes it ideal for solar-powered or battery-backed infrastructure deployments.
- **On-mesh identity:** A repeater has its own on-mesh identity (a keypair and a name) so it can be discovered and administered, but it does not originate or receive user messages; it

only forwards packets at the radio layer.
- **No room server interaction:** Repeater firmware does not connect to a room server. It has no concept of message storage or retrieval.

Best for: Hilltop relays, tower-mounted infrastructure nodes, solar repeaters in remote locations - any deployment where the goal is coverage extension rather than message origination.

ROOM_CLIENT_FIRMWARE

A node running the Companion ("room client") firmware has a full client identity and can connect to a MeshCore **room server** to retrieve stored messages. Note that the message storage and retrieval capability lives in the *room server* firmware; the room client is simply a Companion node that logs in to that server to fetch history it missed. Key characteristics:

- **Client identity:** The node generates a public/private key pair on first boot. This keypair is its identity on the mesh and with room servers. Messages can be addressed to it specifically.
- **Room server connectivity:** A room client connects to a room server primarily over RF through the mesh (or via BLE to a paired room server) to send messages and download stored messages it missed while offline. The room server, not the client, holds the stored message history.
- **Higher power use:** Maintaining a client identity and processing addressed messages consumes more CPU cycles and radio time than a pure repeater.
- **Does not relay:** A Companion/room client does *not* rebroadcast or relay mesh traffic the way a repeater does. Only Repeater firmware forwards packets for other nodes. If you need range extension, deploy a repeater node.

Best for: User-carried nodes, base station nodes that interact with a room server, nodes that need to send and receive addressed messages.

Can a node be both?

No. You must choose one firmware at flash time. A single physical node cannot simultaneously run the Repeater firmware and the Companion (room client) firmware. If you need both roles at one location - for example, a hilltop site that also needs a room-server-connected client - you need two separate physical nodes: a repeater for range extension and a separate client node.

Summary comparison

Feature	REPEATER_FIRMWARE	ROOM_CLIENT_FIRMWARE
Relays/forwards packets for others	Yes (selectively, route-based)	No

Feature	REPEATER_FIRMWARE	ROOM_CLIENT_FIRMWARE
Has on-mesh identity	Yes (infrastructure)	Yes (user)
Connects to room server	No	Yes
Sends/receives user messages	No	Yes
Power use	Low	Higher
Screen needed	No	Optional

How do I connect to a MeshCore room server from the app?

How Do I Connect to a MeshCore Room Server From the App?

A MeshCore room server stores messages for offline nodes and enables larger-group conversations that persist beyond the RF range of any single transmission. Importantly, a room server is reached **over the LoRa mesh, not over the internet**. There is no server IP address, hostname, or TCP port involved, and no firewall configuration is needed. To join one you need to be within RF range of the mesh (directly or via relays) and know the room server's password, which is set by the server operator.

Step-by-step connection

1. Open the **MeshCore app** on your phone and ensure your companion node is connected via Bluetooth (or USB serial).
2. Wait for your companion node to **discover the room server as a contact on the mesh** . Room servers advertise themselves over LoRa, so they appear in your contact list once your node hears them (directly or relayed through other nodes).
3. Select the **room server** from your contact list.
4. When prompted, enter the **room server password**. This is the shared secret set by the server operator. It must match exactly, including capitalization.
5. Once the password is accepted, you join the room and can send and receive messages. Messages are stored by the room server and delivered to members as they come into range.

Administration of the room server itself is done locally over Bluetooth or USB serial on the room server device, not over a network connection.

Troubleshooting connection failures

If you cannot join the room server, work through these checks:

Room server does not appear as a contact

- The room server node may not be powered on or may be out of RF range. Verify the room server has power and that your node can hear it on the mesh (directly or through relays).
- Move closer to the room server or to a node that relays it, then wait for your node to rediscover it on the mesh.
- Confirm your companion node is on the same region/frequency settings as the room server so they can hear each other on LoRa.

Password rejected or wrong password error

- The password you entered does not match the room server's configured password. Passwords are case-sensitive. Try re-entering it manually rather than copy-pasting to rule out invisible characters.
- The operator may have changed the password. Ask the server operator for the current password.

Messages are not arriving

- You may be out of RF range of the room server and any relaying nodes. Because delivery happens over the mesh, you only receive stored messages when your node is within reach of the server (directly or relayed).
- Confirm your companion node is still connected to your phone over Bluetooth or USB and is participating in the mesh.

Can I run MeshCore and Meshtastic simultaneously on the same hardware?

Can I Run MeshCore and Meshtastic Simultaneously on the Same Hardware?

The short answer is: **no**. You can only run one firmware at a time on a given LoRa node. However, there are practical workarounds if you need coverage of both protocols at one location.

Why only one at a time

Both MeshCore and Meshtastic are compiled firmware images flashed directly to the microcontroller (ESP32, nRF52840, RP2040, etc.) on your LoRa board. When you flash a firmware image, it replaces whatever was there before. There is no multi-boot capability, no virtual machine layer, and no way to timeshare the LoRa radio hardware between two independent firmware stacks. The LoRa transceiver (SX1276, SX1262, LR1121, etc.) is a single physical peripheral that can only be driven by one firmware at a time.

Additionally, even if you could somehow run both, they would need to use the same LoRa radio simultaneously - which is physically impossible without two separate radio modules. Each transmission requires exclusive use of the transceiver.

Switching between firmwares

You can re-flash a node from MeshCore to Meshtastic (or vice versa) at any time using the appropriate web flasher or CLI tool. The process takes a few minutes and requires a USB connection. However, you lose all configuration from the previous firmware when you do so,

making it impractical to switch frequently.

Running both protocols at one location

If your community or deployment site genuinely needs to participate in both a MeshCore mesh and a Meshtastic mesh, the practical solution is two separate physical nodes:

- **Node A:** Running MeshCore (REPEATER_FIRMWARE or ROOM_CLIENT_FIRMWARE as appropriate)
- **Node B:** Running Meshtastic firmware

Both nodes can be co-located at the same site and connected to the same power supply, but each should have its **own separate antenna**. Do **not** share one antenna between two transmitting nodes with a passive splitter: a splitter provides no port-to-port isolation, so each radio couples transmit power directly into the other's receiver front-end (risking damage to the LNA/PA), and it also adds roughly 3+ dB of insertion loss in each direction. Co-located 915 MHz transmitters instead need adequate physical antenna separation (ideally several feet of vertical or horizontal spacing) to limit desense and protect the receivers; sharing a single antenna between two transmitters requires a proper RF combiner or cavity duplexer, not a splitter. Many community infrastructure operators run exactly this dual-node, separate-antenna configuration at hilltop repeater sites to serve both ecosystems.

The future: software-defined gateways

Community developers have discussed building a software gateway that runs on a host computer (Raspberry Pi, etc.) and uses two LoRa radio modules - one for each protocol - to bridge messages between the two networks at the application layer. As of today, no such gateway exists in a production-ready state. Any such project would also need to handle the fundamental differences in addressing, encryption, and routing described in the cross-protocol FAQ page.

Recommendation

For most communities, the right answer is to **choose one protocol and standardize**. Mixed-protocol communities face ongoing friction in coordination, troubleshooting, and user experience. If your regional mesh has already standardized on one platform, matching that choice eliminates the need for dual-protocol coverage entirely.

If you are starting a new community from scratch and expect to attract users from both ecosystems, deploying two nodes at each infrastructure site is a legitimate and manageable approach - the hardware cost of an extra node (~$30 - 60 USD) is usually worth the dual-network coverage it provides (note: the two networks still cannot exchange messages with each other - each node simply lets you participate in its own network).