Protocol Deep Dive

- MeshCore Routing Architecture
- MeshCore Packet Format and Encryption
- MeshCore Network Topology Best Practices

MeshCore Routing Architecture

MeshCore Routing Architecture

MeshCore uses a **demand-driven path-based routing protocol**. Unlike Meshtastic's flooding approach, MeshCore establishes explicit routes before sending data.

path discovery packet Mechanism

Route discovery works in four steps:

1. When Node A wants to reach Node D for the first time, it broadcasts an **path discovery packet** (Route Request) packet.
2. Each intermediate node rebroadcasts the path discovery packet, **appending its identity** - building a path record as the packet propagates.
3. When the path discovery packet reaches Node D (or any node that already knows a route to D), it sends a **path acknowledgment** back along the reverse path.
4. Node A receives the path acknowledgment and now has a complete route: `A → B → C → D`.

Route Caching

Discovered routes are cached in each node's routing table. Subsequent messages to the same destination use the cached route without re-discovery, reducing overhead on established links.

Route Maintenance

If a packet fails to reach the next hop, the forwarding node sends a **Route Error (RERR)** message back toward the source. The source node then initiates a new route discovery cycle, ensuring the network self-heals after topology changes.

Advantages Over Flooding

- Only packets on the established route traverse the mesh - significantly less airtime consumption in large networks.
- Scales better: a 100-node MeshCore network consumes far less channel capacity than a 100-node Meshtastic network.
- Repeater nodes can handle more traffic since they are not blindly rebroadcasting everything.

Disadvantages

- Route discovery adds latency before first contact with a new destination.
- Route tables require memory on each node.
- Topology changes can invalidate cached routes, requiring re-discovery.

Repeater vs. Client Roles

MeshCore explicitly distinguishes between two node types:

- **Repeater nodes (infrastructure):** participate fully in route forwarding and carry the routing load of the network.
- **Client nodes (endpoints):** user devices that generate and receive messages but do not forward traffic for others.

This separation makes the protocol more efficient - only dedicated infrastructure carries the routing load, and adding more client devices does not degrade backbone performance.

MeshCore Packet Format and Encryption

This page covers MeshCore's packet encryption as verified from `docs/packet_format.md` and `src/Utils.cpp` in the official MeshCore repository.

Encryption at the Packet Level

Each MeshCore message packet is protected by AES-128 encryption followed by a 2-byte HMAC-SHA256 MAC:

```
[Cleartext header] [AES-128 ECB encrypted payload] [2-byte HMAC-SHA256 MAC]
```

Route Types

Packets carry one of four route types (from packet_format.md):

- `ROUTE_TYPE_FLOOD` - broadcast to all repeaters; used for initial contact and group messages
- `ROUTE_TYPE_DIRECT` - embeds a specific repeater path; only listed repeaters forward the packet
- `ROUTE_TYPE_TRANSPORT_FLOOD` - flood with transport/region code prefix
- `ROUTE_TYPE_TRANSPORT_DIRECT` - direct-routed with transport/region code

Path Learning (How Direct Routing Works)

MeshCore uses a flood-then-direct-route mechanism (not AODV path discovery/acknowledgment):

1. First message to a new destination is flood-routed
2. The destination node returns a `PAYLOAD_TYPE_PATH` packet containing the full repeater path it received the message through
3. The sender stores this path and uses `ROUTE_TYPE_DIRECT` for subsequent messages, embedding the learned path
4. Only the specific repeaters in the path forward the packet - all others ignore it

This mechanism reduces channel load significantly compared to pure flooding once paths are established.

Source: docs/packet_format.md and src/Utils.cpp in the official MeshCore repository. Verified 2026-05-03.

MeshCore Network Topology Best Practices

MeshCore Network Topology Best Practices

Backbone vs. Client Layer

A well-designed MeshCore network is organized into two distinct layers:

- **Backbone layer:** dedicated repeaters placed on elevated sites with clear line-of-sight between them. These form the routing backbone that carries traffic across the network. They are the infrastructure - always on, high antenna, fixed location.
- **Client layer:** user devices (phones, handhelds, base stations) that connect to the nearest backbone node. Clients are endpoints, not relays - they do not forward traffic for other nodes.

This two-layer separation means backbone traffic is predictable and high-performance. Adding more client nodes does not degrade backbone performance, because clients contribute no forwarding load.

Repeater Placement Guidelines

- Aim for **3 - 5 repeaters per coverage zone**, each with line-of-sight to at least 2 others in the backbone.
- **Avoid single points of failure** - if one repeater goes offline, the network should remain functional via alternate paths.
- Ensure **overlapping coverage** between adjacent repeaters so that clients are never more than 1 hop from the backbone.
- High sites (hilltops, building rooftops, water towers) dramatically extend backbone range - prioritize elevation over raw transmit power.

Hop Budget

MeshCore supports up to 64 hops. In practice, plan for **no message traversing more than 6 - 8 backbone hops**. Beyond this:

- Per-hop latency accumulates noticeably.
- Each additional hop adds another potential failure point.
- Route re-discovery after a link failure takes longer with more hops in the chain.

For wide-area networks that would otherwise require long hop chains, use **room servers as message hubs** rather than relying on extended peer-to-peer relay chains.

Advertisement Tuning

- **Flood advertisements** (visible network-wide) should be infrequent - **every 12 hours** is appropriate for stable infrastructure nodes. Frequent floods waste airtime and provide no benefit when the topology is static.
- **Zero-hop advertisements** (local only, for client discovery) can be more frequent - every few minutes is reasonable.
- Review your advertisement intervals if you observe unexplained airtime congestion on the channel.

Mesh Segmentation for Large Networks

In a very large network (50+ repeaters), avoid trying to relay everything peer-to-peer across the entire mesh. Instead:

- Use **room servers as message hubs** for cross-region delivery. Room servers provide message storage and delivery confirmation.
- Segment the mesh into regional clusters, each with its own backbone, connected via room servers at the regional boundaries.
- This reduces the hop count needed for cross-region delivery and localizes the impact of any regional topology change.

Monitoring Topology Health

The MeshCore app includes a **network map feature** that shows which repeaters a node can see and the routes between them. Use this to:

- Verify backbone connections are healthy after deployment.
- Identify repeaters that have lost contact with their neighbors (indicates a failure or coverage gap).
- Confirm that new repeaters have been discovered and integrated into the routing fabric.
- Check hop counts for key routes and identify bottleneck nodes carrying disproportionate traffic.