

Meshtastic Flooding Mesh Protocol Explained

Understanding how Meshtastic actually routes messages helps you make better configuration decisions and troubleshoot problems faster.

Flooding: The Core Mechanism

Meshtastic uses a **flooding broadcast** approach to message delivery. When you send a message:

1. Your node broadcasts the packet over LoRa
2. Every node that receives it checks whether it has seen this packet before (via a packet ID hash)
3. If not seen before: the node rebroadcasts the packet (with hop count decremented by 1)
4. If seen before, or if hop count = 0: the node discards the packet

This is called "flooding" because the packet spreads outward like a flood, covering every node in range up to the hop limit. Every node participates in delivery - there's no designated router.

Hop Count and Hop Limit

Every Meshtastic packet carries a hop count. When a packet is first sent, it starts at the configured `hop_limit` (default: 3). Each time a node rebroadcasts the packet, it decrements the hop count. When hop count reaches zero, nodes still receive but don't rebroadcast.

This means a `hop_limit` of 3 allows a message to be relayed by up to 3 intermediate nodes, reaching a destination up to 3 "hops" away from the source (source → relay 1 → relay 2 → relay 3/destination).

```
# Check current hop limit:
meshtastic --get lora.hop_limit

# Change hop limit (3 is default, 7 is maximum):
meshtastic --set lora.hop_limit 5
```

Duplicate Suppression

Without duplicate suppression, flooding would create infinite loops - every node would keep rebroadcasting forever. Meshtastic prevents this by:

- Maintaining a "recently seen" packet ID cache (stored in RAM)
- Each packet is identified by the sender NodeID plus a 32-bit packet ID; a node treats a (sender, packet-ID) pair it has recently seen as a duplicate
- If a node has seen a packet ID recently, it discards the duplicate without rebroadcasting

Addressing: Broadcast vs Direct

Meshtastic packets can be addressed two ways:

- **Broadcast (channel messages)** - Addressed to all nodes (0xFFFFFFFF). Every node in the network receives and can read the message.
- **Direct (DM messages)** - Addressed to a specific node ID. On firmware v2.5+, DMs use per-recipient public-key cryptography (PKC), so only the destination node can read the content. On older firmware a DM is encrypted only with the channel key, so anyone holding that channel's PSK can read it - it is not truly private one-to-one. Either way, intermediate nodes still rebroadcast the packet (they just may not be able to read it).

Why Flooding Works Well for Small Networks

For networks up to ~50-100 nodes, flooding is remarkably effective:

- No routing tables to maintain or synchronize
- Extremely simple - even new, unconfigured nodes automatically participate in delivery
- Redundant paths - if one relay node is down, another may bridge the gap
- Works well in dynamic environments where nodes join and leave frequently

Flooding's Limitations in Large Networks

As networks grow, flooding creates a "broadcast storm" problem:

- One broadcast can trigger a rebroadcast from many nodes - up to roughly one per node within hop range that does not hear an earlier rebroadcast first. Managed flooding (listen-before-rebroadcast, SNR-based contention) suppresses many of these, but in dense meshes the cumulative retransmissions can still saturate the channel.
- Channel utilization increases with each new node; at high utilization, packets collide and delivery rates drop
- A 200-node network with active users can saturate the LoRa channel, causing significant packet loss

This is why MeshCore's path-based routing scales better for large networks - but for most community networks under 100 active nodes, Meshtastic flooding works well.

Revision #3

Created 2026-05-03 06:38:47 UTC by Mesh America Admin

Updated 2026-06-09 01:57:31 UTC by Mesh America Admin