Meshtastic Repeater Network Patterns

- Designing for Reliability: N+1 Redundancy
- Planned Maintenance Procedures for Live Networks
- Channel Utilization Management

Designing for Reliability: N+1 Redundancy

Designing for Reliability: N+1 Redundancy

In plain terms: make sure that no single node failure can split your network into two halves that can no longer reach each other. A mesh network is only as reliable as its weakest single point of failure. In graph theory terms, a node whose removal splits a connected graph into two or more disconnected components is called a cut vertex. Real-world Meshtastic deployments often develop cut vertices without operators realizing it - especially as networks grow organically. N+1 redundancy means ensuring that for every critical backbone node, at least one backup path exists so that the loss of any single node does not partition the network.

Important caveat: a second node at the SAME site protects only against single-device failure (firmware crash, board death). It does NOT protect against the failures most likely in an emergency - site power loss, lightning, tower collapse, fire, flooding - because both nodes share that fate. For incident-grade redundancy, the backup MUST be at a separate site on independent power.

Identifying Critical Backbone Nodes

Start by drawing your network on paper or a whiteboard. Place each node as a dot and draw lines between nodes that can hear each other directly. Now ask: if I erase this dot, does the network split into two disconnected groups? Any node where the answer is yes is a cut vertex and a single point of failure.

For larger networks, the most reliable analysis is manual: run `meshtastic --traceroute` between nodes on opposite sides of a suspect backbone node - if the only path routes through that one node, it is critical. Third-party visualizers such as meshmap.net can help you eyeball topology, but they only show nodes that report to the public MQTT server (a gateway is required) and are not authoritative; do not rely on them as a complete picture of your mesh.

Providing Backup Paths

Once critical nodes are identified, the fix is straightforward in concept: ensure each one has a backup. Two approaches work well in practice:

- **Second node at the same site:** Place a second ROUTER node at the same location as the critical node. Both nodes cover the same area, so if one fails at the device level, the other continues forwarding. This costs hardware but is simple to maintain. Be aware of two limits: (1) two co-located relays both rebroadcasting add airtime and can raise channel utilization in that area - prefer one active ROUTER plus a standby, or stagger roles, and reconcile this against the channel-utilization guidance; and (2) as noted above, a same-site pair does NOT protect against site-level failures.
- **Alternate mesh path:** Add a node at a different location that bridges the same two network segments. This is more work to plan but is far more robust - it protects against site-level failures (power outage, lightning, physical damage) rather than just single-node failures. This is the approach required for incident-grade redundancy.

Testing Redundancy

Testing is non-negotiable. Make sure no single node failure splits your network: a backup path that exists on paper but has never been verified may fail in practice due to marginal signal, wrong node roles, or misconfigured hop limits. The test procedure is simple:

1. Identify two nodes on opposite sides of the critical backbone node you want to test.
2. Run a traceroute between them and record the path.
3. Briefly power the backbone node down to simulate failure. (Power it off - do NOT simply unplug its antenna while it is transmitting; transmitting into a disconnected antenna can damage the radio.)
4. Run the traceroute again. The path should route around the powered-down node.
5. If routing fails, the backup path is insufficient and must be improved before the critical node is trusted.

A passing reroute test proves the backup path *exists*, not that it has *capacity*. Under incident load the backup path carries the failed node's traffic on top of its own and may saturate. Verify the backup path also has channel-utilization headroom for surge traffic, not just connectivity.

Test redundancy at least quarterly, and ALWAYS as part of pre-incident readiness checks or before any planned event you expect to rely on the network. Annual testing is insufficient for networks used in emergencies - paths degrade silently between tests, and any time the physical environment changes significantly you should retest.

Network Mapping Tools

Third-party visualizers such as meshmap.net provide a visual overlay of node positions for nodes that report to the public MQTT server, which can help you spot obvious topological bottlenecks - but they are not authoritative and will not show nodes that lack an internet gateway. The ground

truth for redundancy verification is the `meshtastic --traceroute <destination_id>` CLI command, which reveals the actual hop-by-hop path packets take in real time.

Planned Maintenance Procedures for Live Networks

Planned Maintenance Procedures for Live Networks

Taking a backbone node offline for maintenance - whether for firmware updates, hardware replacement, or antenna adjustments - affects the users routing through it. With proper planning, that impact can be reduced to a brief interruption rather than a prolonged outage. This page describes a repeatable maintenance procedure for Meshtastic backbone nodes in active community networks.

Pre-Maintenance Checklist

Complete these steps before any planned maintenance window:

- **Notify the community:** Post advance notice (24-48 hours) in your community communication channels - Discord, Signal group, or whatever your community uses. Include the node name, scheduled time, and estimated duration. Users who depend on that backbone link can plan accordingly.
- **Verify backup paths:** Run traceroutes from nodes on either side of the target to confirm alternate routing exists. **If no backup path exists, do NOT take the node offline for non-emergency maintenance until a temporary relay is deployed and verified, or defer the maintenance.** For a sole-path backbone node, planned maintenance without a backup path is itself a planned outage - schedule it only when the network is not needed and notify all users that the network will be DOWN, not merely degraded. Always have a rollback plan: a failed firmware flash can leave a node crash-looping, so never update or remove the only repeater on a path without a verified way to restore it.
- **Schedule during low-traffic hours:** 2-5 AM local time is typically the quietest window for community mesh networks. Emergency networks may have different quiet windows - check your message logs to identify the lowest-traffic period.

- **Document current configuration:** If replacing hardware, record all node settings (node name, channel configuration, role, hop limit) so the replacement can be configured identically before going live.

During Maintenance

Where possible, power down gracefully rather than performing a hard power-off. A graceful shutdown lets the node stop transmitting cleanly and avoids cutting off a packet mid-transmission or corrupting the node's stored configuration during a flash or write. For solar-powered nodes, the practical approach is to disconnect the load output of the charge controller rather than physically unplugging the node in the dark.

Perform the maintenance task - firmware flash, hardware swap, antenna replacement - as quickly as practical. Every additional minute of downtime increases the chance of a user encountering a failed message delivery.

Post-Maintenance Verification

Before declaring the node returned to service, verify it from multiple directions:

1. Send a test message from a node that previously routed through the maintained node and confirm delivery.
2. Run traceroutes from multiple directions to confirm the node is routing normally.
3. If you run a monitoring system (for example MQTT/Grafana or a third-party map such as meshmap.net), confirm it shows the node online AND recently active. Note that seeing a node online is necessary but NOT sufficient - a node can appear up while failing to relay (for example, wrong role or rebroadcast mode). Always also complete step 1 (a real test message routed through the node) before declaring it back in service.
4. Update your network maintenance log with the date, work performed, and any configuration changes.

Emergency Rollback Procedure

If a replacement node does not work as expected - wrong firmware, hardware fault, or configuration error - act quickly. Restore the original node if it is still functional, or connect a known-good spare configured to match the original settings. If neither is possible, notify the community immediately that the outage is extended and provide an estimated restoration time. For any network used in emergencies, keep at least one spare node per critical site, PRE-CONFIGURED to match that site (role, channel/PSK, hop limit, fixed position). An unconfigured spare is not a substitute - configuring on-site under incident pressure wastes the very time the spare was meant to save.

After any unplanned extended outage, conduct a brief post-incident review: what failed, why, and what process change would prevent recurrence. Even a one-paragraph note in the maintenance log is valuable for future operators.

Channel Utilization Management

Channel Utilization Management

Channel utilization (CU) is one of the most important health metrics for a Meshtastic network, yet it is frequently misunderstood or ignored until problems become severe. Understanding what CU measures, what causes it to rise, and how to bring it back down is essential knowledge for any network operator running more than a handful of nodes.

What Is Channel Utilization?

Channel utilization is the percentage of time the radio channel is occupied by transmissions, measured over a rolling **1-minute window**. Meshtastic breaks the minute into six 10-second sub-windows and sums the milliseconds of airtime used in each. A CU of 10% therefore means the channel was busy for roughly 6 seconds out of the last minute (not 90 seconds out of 15 minutes). Meshtastic calculates CU locally on each node by monitoring the time its own radio is keyed up, plus time spent sensing channel activity from other nodes (via Channel Activity Detection). Note that CU counts the airtime of *all* LoRa traffic heard on the same modem settings, including non-Meshtastic LoRa. The reported CU figure is visible in the Meshtastic app under the node detail view and in the device telemetry metrics channel. (Do not confuse this 1-minute window with the EU 10% duty-cycle limit, which is measured over a 1-hour window.)

Healthy, Warning, and Critical Thresholds

These bands match the color coding used by the Meshtastic apps:

- **Under 25% (green):** Healthy. The channel has ample headroom for message traffic, routing overhead, and telemetry. Packet loss due to collisions is rare.
- **25-50% (orange):** Warning zone. You may begin seeing occasional packet loss, especially during bursts of activity. Investigate the causes and plan remediation before the situation worsens.
- **Over 50% (red):** Critical. At this level, the probability of two nodes transmitting simultaneously and causing a collision is high enough to cause significant packet loss on a sustained basis. User-visible symptoms include messages that appear to send but are not

received, slow acknowledgements, and missed position updates.

If you prefer a stricter operational target for infrastructure (for example, keeping a busy backbone channel below ~20-25% to preserve headroom for incident traffic), treat that as a conservative operator guideline, not the system threshold. CU is also a lagging rolling average: a channel can briefly saturate while the reported figure still looks healthy.

Common Causes of High Channel Utilization

Several factors drive CU up. Understanding which ones apply to your network guides the correct fix:

- **Too many ROUTER nodes in close proximity:** ROUTER (and REPEATER) nodes always rebroadcast packets they hear, and in a dense cluster those retransmissions pile on top of each other. As an illustration, several co-located ROUTERs can multiply retransmission traffic without providing a proportional coverage benefit. Managed flooding (listen-before-rebroadcast plus duplicate detection) partially mitigates this, so the multiplier is approximate rather than an exact law, but co-located always-rebroadcasting ROUTERs still inflate traffic and collision rates.
- **Hop limit too high for network geography:** A hop_limit of 5 in a network where the farthest node is only 2 hops from the origin means packets are retransmitted up to 5 times unnecessarily. Every extra retransmission is wasted airtime.
- **High message traffic:** Active communities that send many messages, combined with frequent position and telemetry updates from many nodes, can saturate even a well-configured channel.
- **Routing loops:** Meshtastic actively suppresses persistent routing loops via hop-limit decrement and duplicate detection, so true loops are uncommon. If a node appears twice in a traceroute path, treat it as an unusual symptom worth investigating (asymmetric links or misconfiguration) rather than a routine loop indicator.

Remediation Strategies

These remediations are proactive. They must be applied *before* an incident. Once a channel saturates during an event, Meshtastic has no automatic congestion recovery; the only mid-incident fix is to reduce traffic (fewer senders, shorter messages, suppress telemetry), which requires discipline from every operator rather than a single config change - and reconfiguring deployed nodes mid-incident is often impossible. Provision capacity headroom in advance.

- **Reduce hop_limit:** Set hop_limit to the minimum value that still delivers packets to all nodes in your network. For most community deployments, three to four hops is sufficient.
- **Convert excess ROUTER nodes to CLIENT_MUTE or CLIENT:** Audit your node roles. Nodes in the same area do not all need to be ROUTERs. Designate one or two strategically

placed nodes as ROUTER and set the rest to CLIENT or CLIENT_MUTE to suppress retransmissions from densely packed areas. (CLIENT still rebroadcasts under managed flooding; CLIENT_MUTE does not rebroadcast at all.)

- **Separate co-located networks by frequency slot:** Two networks on different logical channel numbers but the same modem preset still share the same RF frequency and will still collide. To actually separate them in RF you must change the LoRa frequency slot (or the modem preset / region settings), not just the logical channel index.
- **Consider MeshCore for large networks:** Meshtastic uses managed flooding for broadcasts - nodes listen first and suppress their rebroadcast if a neighbor already relayed the packet, while ROUTER/REPEATER nodes always rebroadcast (since v2.6, direct messages use next-hop routing rather than flooding). In networks with dozens of active nodes and high message traffic, broadcast flooding is a major driver of high CU. MeshCore uses path discovery with source/next-hop routing - it floods once to discover or recover a route, then sends future messages along the learned path - which can scale better in large, dense, high-traffic community networks. Consult the official MeshCore documentation for its exact routing model.