# Case Study: Meshtastic in Production — North Georgia, June 2026

Loomwave — loomwave.org

**Status:** evidence document · **Data:** two live packet captures, analyzed 2026-06-13
**Companions:** meshtastic-routing-analysis.md (code-level), meshtastic-meshcore-differentiation.md (design response), north-georgia-validation.md (sim calibration against this network)

This is a data-driven account of what a real, healthy, well-run Meshtastic network does on the air: what it is genuinely good at, where the measured behavior reveals structural limits, what could be improved while staying Meshtastic, and which problems no amount of iteration can fix without becoming a different protocol. The companion code analysis predicted most of these behaviors from source; this document shows them happening at scale.

## 1. The datasets

| | Local broker | mtnme.sh regional broker |
|---|---|---|
| Window | 77.8 h continuous | first hours of access (growing) |
| Rows | 19,049 | 5,511 |
| Source nodes seen | 312 | 138 |
| MQTT gateways | 1 | **37** |
| Channel | MediumFast (SF9/BW250) | MediumFast dominant; LongFast, PKI minor |

Both captures log every MQTT-uplinked packet with per-reception SNR/RSSI, hop counters, port, and payload length. The regional broker is the more revealing instrument: with 37 gateways uplinking everything they hear, it shows each packet from every vantage at once.

## 2. What Meshtastic demonstrably does well

Credit first; the data is unambiguous about these.

- **It works, continuously, at metro scale.** 312 distinct nodes over three days on one channel, packets flowing every hour of the day (median 249/h, max 317/h at one gateway). Zero-infrastructure delivery across a metro area is a real, functioning capability.

- **Multi-hop flooding genuinely extends range.** 75% of received packets traveled ≥1 hop; the modal delivery is 2 hops; usable deliveries at 6–7 hops appear daily. For a protocol with no routing state at all, that is remarkable robustness.
- **It survives the SNR cliff.** 6% of local receptions arrive below −10 dB SNR — at or under the MediumFast demodulation floor — and the network shrugs; redundancy through flooding converts marginal links into statistical delivery.
- **The ecosystem is the moat.** 37 volunteer gateways uplinking 24/7, hardware from a dozen vendors, and a config culture (channel presets) that mostly keeps 300+ strangers interoperable. No protocol feature competes with this; it is why "stay on Meshtastic" must be taken seriously as the null hypothesis.

## 3. What the data shows it cannot do

### 3.1 The traffic is almost entirely self-description

Unique-packet traffic mix over 77.8 hours, local capture:

| Type | Share |
|---|---|
| nodeinfo | 42.9% |
| position | 26.7% |
| telemetry | 18.9% |
| traceroute | 8.0% |
| **text (the user payload)** | **1.3%** |

**~89% of all packets are the network describing itself**; under 2% is the messaging the network exists to deliver. This is not user misbehavior — it is the default-on cadence of NodeInfo/Position/Telemetry, multiplied by flooding. The regional capture agrees (text = 2.0%). Every scaling discussion must start here: the channel is not full of messages, it is full of presence.

### 3.2 Flooding's cost, photographed from 37 vantages

The regional broker shows what flooding means physically: **the median packet is uplinked by 16 gateways; p90 is 21 copies; 92% of the entire MQTT stream is redundant copies of something already received.** Each of those copies corresponds to RF airtime somewhere — the same bytes occupying the channel across the whole metro at once, because flooding has no concept of "delivered."

Airtime accounting (computed from payload sizes at MediumFast): - Source transmissions alone: **1.5% channel utilization** (local, 3.2 days) — modest. - Including the flood re-transmissions observable in hop counters: **≥4.7% local, ≥11.6% regional in the evening**

**hour** — and these are lower bounds, counting only relays some gateway heard. The validation work (`north-georgia-validation.md`) measured peaks near 36% on this same network. The flood multiplies a quiet network into a busy channel, and the multiplier grows with node count — the v1 sim collapse curve, observed in vivo.

### 3.3 Hop budget anarchy

`hop_start` settings observed simultaneously on one channel: 3 (30%), 5 (31%), 7 (19%), 4 (15%), plus stragglers at 1, 2, 6. Every operator picks their own flood radius; nobody owns the trade-off. The result: **11–14% of received packets arrive with their hop budget fully exhausted** — packets that died mid-network from the receiver's perspective, while high-hop settings (7) flood the entire metro with every NodeInfo from one suburban node. There is no mechanism — none — by which the network converges on a sane radius.

### 3.4 No notion of delivery

Routing-port packets (the ACK machinery) are 0.5% of traffic; traceroute — the manual "is anyone out there" probe — is **8%**, sixteen times the ACK volume. Users compensate for the absence of delivery feedback by manually probing the network, with 200-plus-byte flooded diagnostics. The protocol's answer to "did it arrive?" is "send another flood."

### 3.5 Two nodes are 35% of everything

The top two sources (both infrastructure-grade nodes) account for 32–38% of all packets in both captures. Flooding gives every node identical broadcast rights; position at elevation converts those rights into channel dominance. There is no admission control, no airtime fairness, no way for the network to push back.

## 4. What could be done staying on Meshtastic

Honest engineering: a motivated operator community could deploy all of the following without forking the protocol, and would get real headroom from it.

1. **Slash default beacon cadences.** NodeInfo/Position/Telemetry at half rate ≈ 40%+ of all airtime back, instantly, firmware-config only. The single highest-leverage change and it needs no code.
2. **Channel-wide hop-limit discipline** (community policy: hop_start 3 everywhere, 5 for sanctioned high sites). Would eliminate most exhausted-budget waste and metro-wide floods of local chatter. Costs only governance.
3. **Router-role hygiene**: the existing ROUTER/CLIENT roles, actually enforced by the community map, would stop the two-node dominance pattern from compounding (cf. `elevated-nodes.md` — elevation is capability, role is configuration).
4. **MQTT governance** (already in our design notes from the differentiation work): downlink injection rules to stop backhaul from re-entering as RF flood.

5. **ChannelUtil-driven self-throttling** exists in firmware today; defaults could honor it more aggressively (beacons skip when util > 15%).

Generously estimated, these recover **3–5× effective capacity**. They are worth doing — the people on this network are friends of this project, and we should say so publicly.

## 5. The boundary: what iteration cannot fix

The recommendations above tune the flood. They do not change what it is:

- **No scheduled access exists or can exist** without breaking every deployed node: the MAC is CSMA-and-hope, so the hidden-node loss measured in `hidden-node-analysis.md` (and reproduced in sim v1) is permanent. The Stone Mountain hub — ~92% of heard nodes mutually hidden — cannot be scheduled around within this MAC.
- **Flooding is the routing.** Hop-count tuning shrinks the storm radius but the delivered-traffic fraction still falls as N grows (v1 collapse knee); 16-gateway redundancy per packet is the design working as intended.
- **One modulation for everyone.** A channel-wide preset means the worst link sets the budget for the metro; adaptive modulation (sim v4: ×16–34 capacity) requires a coordinator role the protocol cannot express.
- **Self-description ≈ 89% is structural**: with no registry, no membership, and no schedule, presence must be broadcast perpetually. A protocol with cells and membership states (REGISTER once, scheduled DATA after) simply does not pay this tax.

That is the line. Below it, stay and tune; above it, a different protocol is required. The quantified version of "above it" is the Loomwave design decision record (`design-decision.md`), and every number in this case study is an input to it.

## 6. Methodology notes

Captures: `tools/capture_mqtt.py` (subscribe-only) on a local broker (3.2 d) and mtnme.sh (read access granted 2026-06-12; per-packet ServiceEnvelope metadata only). Airtime computed from payload+header sizes via the standard LoRa ToA formula at each channel's preset; flood re-TX estimated from per-packet max observed hops-traveled (lower bound — relays no gateway heard are invisible). Gateway-copy statistics use packet-id dedup across gateways. Encrypted packets (28% regional) contribute size and hop data but not port classification. The mtnme capture continues to accumulate; numbers here are its first hour and will be refreshed for the RFC paper.