

Is Meshtastic encrypted?

Can anyone read my messages?

Short Answer

Meshtastic messages are encrypted, but the level of protection depends on which channel you're using. The default channel (LongFast) uses a known public key and provides essentially no privacy. Custom channels with randomly generated keys provide strong message confidentiality.

The Default Channel: Not Private

The default Meshtastic channel uses a Pre-Shared Key (PSK) of `AQ==`, which is base64 for a single byte of value `0x01` - a publicly known and documented default key. Any Meshtastic user in radio range can read messages on the default channel, including the [Meshtastic app](#) developers and anyone who has read the public documentation.

The default channel is suitable for public community communication where privacy isn't a concern. Do not send anything private on the default channel.

Custom Channels: Strong Privacy

When you create a channel with a randomly generated PSK (using the app's random key generator), the app can produce a 256-bit AES key that cannot be recovered by brute force with current technology. Note that Meshtastic uses AES-128 or AES-256 in CTR mode depending on the key length (16 vs 32 bytes) - the default and many shared channels are AES-128, while the app's random-key generator can produce a 256-bit key. Messages on this channel are readable only by nodes that have the same PSK.

Encryption security: AES-256 is a strong, government-approved cipher and the cryptography itself is sound. However, security depends on key management (how you distribute the PSK to your community), and Meshtastic exposes metadata and relies on manual key distribution - so do not treat a custom channel as government-grade secure for highly sensitive traffic.

Direct Messages: Even Better (Firmware 2.5+)

Direct messages use X25519 (Curve25519) ECDH public-key (PKI) encryption. PKI direct messaging was introduced in firmware 2.3 and became the robust default in 2.5; run firmware 2.5 or later on all nodes for current PKI DM security. It provides:

- End-to-end encryption between just sender and recipient
- No shared secret to distribute - keys are derived automatically from each node's public/private key pair

Note: Meshtastic does **not** provide forward secrecy. Because the keys are long-lived, traffic captured today can be decrypted later if a key is compromised - the official documentation notes that Meshtastic is vulnerable to "harvest now, decrypt later" attacks. Do not assume that seizing a key in the future cannot expose past messages.

What an Eavesdropper Can See

Even with properly configured channel encryption, a radio observer can see:

- That LoRa transmissions are occurring on the frequency
- The approximate timing and frequency of transmissions
- Some packet header fields that are not encrypted (node IDs, a channel hash, hop count, and routing/metadata)

Message content, sender names, and channel names are inside the encrypted portion of the packet, so they are hidden if a custom PSK is in use. However, the unencrypted header still exposes node IDs, a channel-hash byte, and routing/metadata - so an eavesdropper can still learn who is transmitting and when.

Practical Recommendations

- For community chat where privacy isn't critical: default channel is fine
- For any sensitive coordination: create a custom channel with a random PSK

- For private one-on-one messages: use DMs with firmware 2.5+ on both ends
 - For highly sensitive communications: LoRa mesh is supplemental - use Signal or other end-to-end encrypted messaging for truly sensitive content
-

Revision #6

Created 2026-05-03 05:53:11 UTC by Mesh America Admin

Updated 2026-06-08 19:54:29 UTC by Mesh America Admin