MeshCore Security and Encryption

MeshCore uses a layered cryptographic system verified from the project's source code. All claims on this page are sourced from `src/Utils.cpp`, `src/MeshCore.h`, and `src/Identity.h` in the official MeshCore repository.

Symmetric Encryption

- **Algorithm: AES-128** (ECB mode with zero-padding for the final block)
- Key size: 16 bytes (`CIPHER_KEY_SIZE = 16`)
- The shared AES key is derived via ECDH (see below)

Message Authentication

- **MAC: HMAC-SHA256** truncated to 2 bytes (`CIPHER_MAC_SIZE = 2`)
- Scheme: **Encrypt-then-MAC** - the ciphertext is MACed, not the plaintext
- Functions: `encryptThenMAC` / `MACThenDecrypt`

Key Exchange

- **ECDH via X25519** - Ed25519 identity keys are transposed to X25519 for Diffie-Hellman key exchange (`calcSharedSecret` in Identity.h)
- The resulting shared secret is used as the AES-128 key for the session

Identity and Signing

- **Identity keys: Ed25519**
- Public key size: 32 bytes (`PUB_KEY_SIZE = 32`)
- Private key size: 64 bytes (`PRV_KEY_SIZE = 64`)

- Signature size: 64 bytes (`SIGNATURE_SIZE = 64`)
- **Advertisements are signed** with Ed25519 to prevent node identity spoofing

What This Means in Practice

- Messages between two MeshCore nodes use a unique AES-128 key derived from their ECDH exchange - no shared secret needs to be pre-distributed
- The MAC is only **16 bits (2 bytes)**. It reliably catches accidental corruption, but a deliberate attacker who can transmit can forge a packet that passes the MAC check with modest effort (expected ~32,000 attempts). Do not rely on MeshCore's MAC to prevent message forgery by a capable adversary — it is an integrity check, not strong authentication.
- Advertisements are Ed25519-signed, so an attacker cannot forge an advert for a public key they do not control. Note, however, that nodes are addressed by a short public-key prefix, so prefix collisions can cause addressing ambiguity (see the troubleshooting reference). Impersonation/confusion at the routing/addressing layer is possible; identity cannot be assumed unforgeable in an absolute sense.
- Channel/group messages use a shared symmetric key derived from the channel configuration. The channel key is shared by all members, so any member can forge messages attributed to any sender — channel "authentication" is group-level only. The default public channel uses a publicly known key (`8b3387e9c5cdea6ac9e5edbaa115cd72`), so public-channel traffic provides no confidentiality against anyone who knows that key. Do not treat public-channel traffic as private or secure against observers.

Source: Official MeshCore repository, src/Utils.cpp, src/MeshCore.h, src/Identity.h. Verified 2026-05-03.

Revision #4
Created 2026-05-03 03:55:57 UTC by Mesh America Admin
Updated 2026-06-09 14:22:18 UTC by Mesh America Admin