Sealio: A Peer-to-Peer Encrypted Messaging System on Solana

Roberto Salvatore Piras Murgia
info@sealio.io
www.sealio.io

Abstract

A purely peer-to-peer messaging system should allow two parties to exchange private messages without relying on a server operator who can read, retain, or censor them. We propose a system in which each message is end-to-end encrypted on the sender's device using HPKE and published as an opaque ciphertext memo in an ordinary Solana transaction. The public ledger serves as a globally replicated, timestamped mailbox; it stores only ciphertext and the sender and recipient addresses. Private keys never leave the device and are protected by the Secure Enclave and a local PIN. No message relay, account database, or phone number is required.

1. Introduction

Commercial messengers depend on a central operator for delivery, identity, and storage. Even when content is encrypted, the operator controls availability, holds metadata, and can be compelled to cease service. What is needed is a delivery medium that no single party controls, combined with encryption that no party other than the recipient can undo. A public blockchain provides the first property; modern hybrid public-key encryption provides the second.

2. Identities

A participant is identified by an Ed25519 Solana address. A keypair is either created on the device or imported as a private key. The key is stored in the iOS Keychain, wrapped by a key held in the Secure Enclave, and unlocked by a user PIN. An X25519 encryption key is derived for HPKE so that the signing address doubles as the messaging address.

3. Messages

To send plaintext m to recipient R, the sender computes c = HPKE.Seal(pk_R, info, aad, m) with DHKEM(X25519), HKDF-SHA256 and AES-GCM or ChaCha20-Poly1305. The encapsulated key and ciphertext are serialized, prefixed with a version tag, and written to the Memo program in a transaction that references R:

```
QPS1|M|<base64(enc || ciphertext)>
```

```
 Sender                  Solana ledger                 Recipient
 ------                  -------------                 ---------
 m --HPKE.Seal(pk_R)-->  tx{memo: QPS1|M|c, to: R} -->  scan R's txs
                         (ciphertext, public)          HPKE.Open(sk_R) -> m
```

The recipient retrieves transactions touching its address, filters memos carrying the QPS1|M tag, and decrypts locally. Messages that fail authentication are discarded.

4. Network

Delivery reuses Solana consensus. Once a transaction is confirmed, the message is ordered, timestamped, and replicated by every validator. Clients may query any RPC node; no Sealio server sits in the path. Each message costs one transaction fee, which also acts as a natural limit on spam.

5. Privacy

Content is confidential to the holder of sk_R. The ledger, however, is public: sender address, recipient address, time, and ciphertext length are visible to everyone and permanent. As with Bitcoin, privacy at the identity level can be maintained by keeping addresses unlinked to real-world identities and by using a fresh address per relationship.

6. Security Considerations

Compromise of a device key exposes all past and future messages to that address, since the permanent record removes the protection usually given by message expiry. Users should treat the PIN and device as the root of trust. Quantum-resistant KEMs may be adopted under a new version tag without changing the transport.

7. Funding

Sealio carries no token and no subscription. Development may be supported by optional donations to the treasury address 8LnT5zYXdWj3HFw1iFDZKSJa5gfq6LX8P3SV8Us3meYM.

8. Conclusion

We have described a messaging system that requires no trusted operator. Encryption is performed end to end with HPKE, keys remain on the device, and an existing public ledger provides ordering, persistence, and delivery. The network is robust in its unstructured simplicity: anyone running a client can send to any address, and only the recipient can read.