Secret: Technical Whitepaper

Secure, decentralized messaging with built-in crypto and community-powered infrastructure.

Version: 1.0
Date: May 2025
Secret Foundation
https://secretapp.io

Table of Contents

---

1. Introduction

1.1 Architecture Overview

Secret is architected as a decentralized communication system composed of two core components: client-side applications and server-side infrastructure. Together, these components enable secure, private, and censorship-resistant messaging and community interactions across a globally distributed network.

Secret features a microledger-based architecture — where core entities in the network maintain a verifiable, cryptographically signed chain of events, forming the foundation of its decentralized trust model.

Client-Side Applications

Secret's client-side applications include native apps for iOS, Android, and Desktop platforms (Windows, macOS, Linux). These applications deliver the full user experience, enabling secure messaging, community participation, media sharing, and wallet functionality directly from users' devices.

Each client is a fully self-sovereign node responsible for:

- Generating and storing cryptographic keys securely on the client device
- Encrypting and decrypting all communications, ensuring zero knowledge to any third party
- Signing transactions and messages using private keys that are stored exclusively on the client device
- Establishing peer-to-peer or relayed communication channels, including over transport-agnostic layers
- Caching and persisting data locally for offline-first behavior
- Syncing state and message history using conflict-free replication once connectivity is re-established
- Enabling fully non-custodial wallet functionality, with private keys held exclusively by the user and never exposed to the network

Clients are designed to operate even under offline, low-connectivity, or adversarial network conditions — including censorship, surveillance, and firewalls — by leveraging decentralized routing, local data persistence, and alternative transport layers.

Server-Side Infrastructure

Secret's server-side infrastructure is a decentralized and modular backend system designed to support secure communication, data persistence, identity resolution, and network coordination — without relying on a single point of control.

Rather than operating on a traditional centralized server model, Secret distributes responsibilities across independently operated nodes that can be run by the Secret Foundation, trusted third parties, or even end users themselves. These nodes collectively provide the backbone for:

- Encrypted data storage and retrieval
- Message routing and relay across unreliable or censored networks
- Decentralized identity management and peer verification
- Support for public services such as discovery, notifications, and abuse reporting

Server nodes are designed to be permissionless and composable — any participant can contribute infrastructure by operating specific node types using client or standalone deployments.

In particular, Secret clients can optionally act as lightweight servers themselves, enabling users to participate directly in hosting or relaying encrypted data, thereby increasing the resilience, scalability, and censorship resistance of the network.

This architecture allows Secret to combine the privacy and autonomy of peer-to-peer systems with the availability and reliability of federated infrastructure, ensuring that users retain full control over their data — even when offline or under network constraints.

1.2 Networking Model

The networking layer of Secret is designed to support resilient, censorship-resistant communication in a decentralized environment. It enables peer-to-peer interactions, federated hosting, and multi-hop message routing — all while minimizing metadata exposure and ensuring data integrity.

At its core, Secret uses a transport-agnostic networking model, meaning it can operate across a variety of communication layers, including the public internet, local Wi-Fi, Wi-Fi Direct, Bluetooth, or mesh networks. This allows Secret to function reliably in both online and offline environments, as well as under adverse or adversarial conditions.

Core Components of the Networking Model

1. **Peer-to-Peer Messaging**
   Users can communicate directly between devices without relying on intermediaries. This peer-to-peer mode ensures:

   - Minimal latency and direct encryption handshake
   - No central dependency or surveillance vector
   - Local-only message transmission in private networks

2. **Federated Hosting**
   Users may choose trusted infrastructure providers to store encrypted data and assist in message delivery. These hosts:

   - Do not access message content (end-to-end encrypted)
   - Operate independently, enabling network diversity
   - Can be run by anyone, including clients

3. **Message Relaying**
   To protect metadata and improve message delivery in restricted environments, Secret supports encrypted multi-hop relaying:

   - Routes messages through multiple anonymous nodes
   - Obscures sender and recipient metadata
   - Enables communication even when peer discovery fails

4. **Transport Agnosticism**
   Secret is not bound to a specific network stack. Clients can communicate over:

- Internet (TCP/UDP/WebSocket)
- Local networks (Wi-Fi, Ethernet)
- Proximity transports (Bluetooth, Wi-Fi Direct)
- Private mesh configurations

This flexibility enhances offline-first capabilities, making Secret suitable for local, autonomous, and low-infrastructure deployments.

5. **Offline-First Design**
   Secret assumes disconnection as the default state. Clients can:

   - Queue and encrypt messages locally
   - Operate entirely on LAN or mesh without internet
   - Automatically sync messages and content when reconnected
   - Resolve state conflicts via CRDT-style strategies

Together, these networking capabilities form a robust and flexible foundation for the Secret platform — one that defends against censorship, supports low-resource deployments, and prioritizes user privacy by default.

1.3 Privacy and Security Model

Secret is built from the ground up to maximize user privacy, enforce strong data protection guarantees, and minimize metadata exposure — even in hostile or surveillance-prone environments. The platform combines end-to-end encryption, anonymous networking, and client-side key management to create a communication system that respects user sovereignty and resists compromise.

Unlike conventional messaging platforms that rely on centralized servers and expose user metadata, Secret ensures that only the intended recipients can access any message or content, and that network intermediaries — including infrastructure operators — gain no insight into user activity.

Core Privacy Principles

1. **End-to-End Encryption**
   All communications in Secret — including messages, media, and metadata where feasible — are encrypted on the sender's device and decrypted only on the recipient's device. This ensures:

   - No third party (including hosts or relays) can read or alter messages
   - All media and file attachments are encrypted with content-specific keys
   - Group communication uses shared or per-session keys, updated periodically

2. **Local Key Management**
   Secret follows a non-custodial model: all private keys are generated, stored, and used locally on the client device.

   - Private keys never leave the device
   - Clients manage key rotation, signing, and verification
   - Each user identity is cryptographically tied to a persistent local keypair

   This architecture eliminates server-side key exposure and reduces attack surfaces.

3. **Metadata Obfuscation**
   To protect users even when message contents are encrypted, Secret employs techniques to reduce metadata leakage:

   - Message relaying through anonymous hops to obscure sender and receiver IPs
   - No global usernames — user identifiers are locally resolved or cryptographically derived
   - Encrypted or ephemeral routing data, reducing traceability of who talked to whom, when

4. **Anonymous Participation**
   Secret allows users to join and use the platform without providing any personally identifiable information (PII) such as phone numbers, emails, or usernames. The onboarding process is fully anonymous and privacy-preserving by design.

   - **No PII Required**: Users can create accounts without linking to real-world identities.
   - **Cryptographic Identity Creation**: Each user is assigned a unique identity derived from locally generated or imported cryptographic key material (e.g., a mnemonic phrase).
   - **Ethereum Mnemonic Support**: Users can optionally use the same mnemonic phrase associated with their Ethereum wallet to generate their Secret identity, creating seamless interoperability with the Web3 ecosystem.
   - **Customizable Public Presence**: While anonymous by default, users may choose to attach a display name, avatar, or pseudonym to personalize their presence. This identity data remains under the user's control.

   This model ensures maximum privacy while still allowing users to build reputations, earn trust, or interact under chosen pseudonyms, all without revealing their real-world identity.

5. **Censorship Resistance**
   The combination of decentralized storage, multi-hop relaying, and transport-agnostic communication means users can:

- Continue sending messages even under network restrictions
- Fall back to local-based networking when the internet is blocked
- Access cached or mirrored content when original hosts are unavailable

Secret is designed to resist takedowns, internet shutdowns, and content-based filtering.

Security Guarantees

- **Confidentiality**: Only intended recipients can decrypt messages
- **Integrity**: All messages are signed and tamper-evident
- **Forward Secrecy**: Session keys are rotated periodically to limit the impact of key compromise
- **Authentication**: Message origin is cryptographically verifiable
- **Resistance to Traffic Analysis**: Through relays and metadata obfuscation, Secret limits third-party visibility into communication patterns

Together, these properties ensure that Secret offers not just encrypted chat — but a privacy-preserving communication system that's decentralized, anonymous, and built for the realities of modern surveillance and censorship.

1.4 Design Principles and Philosophy

Secret is built on the belief that private, secure, and decentralized communication is a fundamental right — not a privilege. Every aspect of the platform, from its cryptographic core to its user-facing design, is guided by principles that prioritize user sovereignty, resilience, and trust minimization.

Core Design Principles

1. **Privacy by Default**
   Privacy is not a setting — it's the default. All communications are end-to-end encrypted, and users are not required to share personal information to join or use the platform.

2. **Decentralization at Every Layer**
   Secret avoids centralized control wherever possible. Identity, content storage, message delivery, and network coordination are all designed to be distributed across user-operated or independent nodes. This decentralization enhances censorship resistance, availability, and community self-governance.

3. **User Sovereignty**
   Users own their identities, content, and data. All keys are generated and stored locally. Content can be hosted independently or with trusted third parties. The system is opt-in, self-custodial, and interoperable with external wallets and ecosystems.

4. **Offline-First Resilience**
   The platform is designed to operate in low-connectivity or offline environments by default. Peer-to-peer communication, local data caching, and deferred synchronization enable usage in restricted networks, remote areas, or during internet shutdowns.

5. **Interoperability with Web3**
   By supporting Ethereum-compatible mnemonics and integrating cryptographic identity with wallet functionality, Secret enables seamless interaction with decentralized applications, token economies, and identity systems — without compromising user privacy.

These principles serve as the foundation for all protocol, product, and ecosystem decisions within Secret. They ensure that as the platform evolves, it remains aligned with its mission: to empower users to communicate, connect, and organize freely — without surveillance or compromise.

2. Core Protocol Components

This section provides a deep dive into the core technical building blocks that power Secret's decentralized architecture. Each component works together to enable private communication, trustless coordination, and resilient content delivery — all without centralized control or surveillance.

2.1 Client-Side Node Responsibilities

Every Secret client acts as a fully independent node in the network, responsible for enforcing security, managing data, and participating in communication protocols. The client is not just a user interface — it's an active peer in the protocol.

Responsibilities include:

- **Key Management**
  - Generates and stores private keys locally
  - Signs messages and transactions
  - Manages key rotation and device linking
- **Encryption and Decryption**
  - Encrypts all messages and content before sending
  - Decrypts content locally upon receipt
  - Supports individual, group, and media-specific encryption schemes
- **Local Data Persistence**
  - Stores message history, media, and identity metadata locally
  - Operates in offline or low-connectivity environments
  - Caches unsent messages and syncs upon reconnection

- **Peer-to-Peer and Relay Communication**
    - Establishes direct or relayed encrypted channels
    - Selects routing paths based on availability, trust, or anonymity settings
    - Communicates over transport-agnostic channels
- **Optional Hosting and Relaying**
    - Can act as a Content Host or Relay Host directly from the client
    - Serves encrypted content for other users
    - Routes traffic anonymously for others to enhance network obfuscation

The client's local-first architecture ensures that the user's privacy is protected even when infrastructure is unavailable, untrusted, or adversarial.

2.2 Peer Discovery and Identity Resolution

In a decentralized network like Secret, users and hosts need a way to discover and verify each other's identities. Secret uses a combination of peer discovery, decentralized identity resolution, and authority coordination to establish secure and trustable connections between network participants.

Peer Discovery

Secret leverages a modular peer discovery mechanism, enabling clients and hosts to find each other across a variety of network types and conditions.

- **Multicast DNS (mDNS)**: For local, offline, or LAN-based discovery.
- **Distributed Hash Tables (DHT)**: For global discovery over the public internet.
- **Bootstrap Nodes**: Trusted entry points (e.g., Authority Hosts) that help new clients join the network.

These discovery mechanisms work across different transport layers, allowing Secret to operate over Wi-Fi, Bluetooth, mobile networks, and mesh topologies.

Identity Resolution

Each user and host in the Secret network is represented by a cryptographic identity, typically derived from a locally stored keypair or an imported Ethereum-compatible mnemonic. Identity resolution is handled through Authority Hosts, which form a decentralized registry of public keys and associated identifiers.

Key features include:

- **Decentralized Identity Registry**: Maintained across Authority Hosts to ensure consistency and verifiability.
- **Public Key Mapping**: Users and hosts are referenced by persistent identifiers tied to their public keys. No centralized username or account system is used.

- **Identity Verification**: All clients and hosts are verified through cryptographic key signatures to ensure authenticity.
- **Revocation and Updates**: Users can rotate keys, link new devices, or revoke compromised credentials while preserving identity continuity.

Onboarding and Trust Bootstrapping

When a user joins Secret, they generate or import a keypair. This identity is then registered within Authority Hosts for discoverability. Trust can be bootstrapped through:

- Invite codes or referrals from known users
- Staked identity verification via token bonding
- Out-of-band key exchange (e.g., QR codes, shared secrets)

This hybrid model supports both anonymous participation and reputation-building, without ever requiring centralized accounts or real-world identity.

2.3 Message Routing and Transport

Secret is designed to deliver private, censorship-resistant communication regardless of network conditions. To achieve this, it uses a flexible message routing layer that supports direct peer-to-peer delivery, multi-hop relaying, and transport-agnostic communication. All messages are end-to-end encrypted, and routing paths are designed to obscure sender and recipient metadata.

Direct Messaging (Peer-to-Peer)

When possible, Secret establishes a direct encrypted channel between the sender and recipient. This method offers:

- Low latency communication
- Minimal metadata exposure
- End-to-end encryption without intermediaries

Secret supports direct encrypted messaging between users when conditions and user preferences allow for peer-to-peer communication.

Relay-Based Routing

When direct messaging is not possible — due to firewalls, NAT restrictions, or offline recipients — Secret uses Relay Hosts to route messages across the network anonymously.

- **Multi-hop Routing**: Messages can be forwarded through several relay nodes, similar to onion routing, to obscure the original source.
- **Encrypted Transport**: Relays cannot decrypt or inspect message content; they only pass along encrypted payloads.

- **Fallback Resilience**: Relays ensure message delivery even when peers are temporarily unreachable or censored.

This mechanism enhances network reachability and privacy, especially in adversarial environments.

Transport-Agnostic Layer

Secret's communication layer is designed to operate over a wide range of physical and logical transports:

- Internet (TCP, UDP, WebSocket)
- Local Networks (Wi-Fi, LAN, Ethernet)
- Proximity Transports (Bluetooth, Wi-Fi Direct)
- Mesh and Offline Networks (PNET, ad hoc routing)

The transport layer is abstracted, allowing Secret to select the best available channel automatically based on the user's environment, preferences, and threat model.

Routing Privacy Enhancements

To prevent traffic analysis and metadata leakage, Secret's routing design includes:

- Ephemeral session keys and encrypted headers
- Unlinkable message packets with no shared identifiers between hops
- Optional route randomization to prevent traffic correlation

Combined with end-to-end encryption, these features ensure that message content, sender identity, and communication patterns remain private — even from infrastructure operators.

2.4 Offline Operation and Data Sync

One of Secret's core strengths is its ability to function independently of constant internet connectivity. The system is designed to be offline-first, ensuring that users can continue to access, create, and share content even in disconnected, remote, or censored environments. Synchronization occurs automatically when connectivity is available, using secure and conflict-resilient protocols.

Local-First Architecture

Secret treats the client device as the source of truth. All user data — including messages, profiles, media, and settings — is stored locally and doesn't leave the device unencrypted.

- Users can send and receive messages, participate in communities, and interact with content without an active internet connection
- Data is cached, queued, and encrypted locally until network connectivity is restored

- Clients can communicate over local transports such as Bluetooth, LAN, or Wi-Fi Direct

Deferred Synchronization

When the device reconnects to the network (either directly or via relays/hosts), the client performs:

- **State Reconciliation**: Using CRDT-style logic to resolve conflicts between local and remote state
- **End-to-End Encrypted Sync**: All data remains encrypted during transmission, even when syncing with content hosts or relays
- **Selective Syncing**: Clients sync only necessary data (e.g., specific communities or conversations), preserving bandwidth

Offline Networking Scenarios

Secret can operate in offline or “air-gapped” environments using:

- LAN-only community setups
- Bluetooth/Wi-Fi Direct for peer-to-peer exchange
- Private relay meshes within isolated networks

This makes Secret ideal for use in areas with unreliable infrastructure, disaster zones, or censorship-heavy regimes — where typical messaging platforms fail.

2.5 Blockchain-based Architecture

Secret implements a blockchain-based architecture in which identities, communities, and groups each maintain their own append-only, cryptographically verifiable ledger of signed events. Every event — such as a key rotation, role assignment, or membership change — is stored as a single block that references the previous event’s hash, forming a tamper-evident, ordered history for each entity.
These per-entity chains are verified locally by clients, not by trusting external infrastructure. This design ensures that identity and group state is always auditable, authentic, and consistent — even when served by untrusted hosts or relayed across the network.

Identity Ledgers

Each identity in Secret is a sequential chain of cryptographically signed events. Every identity ledger:

- Starts from a genesis event (initial key creation or mnemonic import)
- Contains ordered events (such as key rotations, device add / revocation, etc.)
- Is stored and replicated by Authority Hosts
- Is verified entirely on the client — no Authority Host is trusted blindly

- Supports slashing and reporting if hosts misbehave or provide tampered identity chains

Design Properties:

- Each identity maintains its own cryptographic event chain
- All events are hash-linked and digitally signed
- Identity timelines are validated entirely on the client
- No reliance on central authority or trusted servers
- Tamper-evident history of key and state changes
- Identity ledgers are primarily managed by Authority Hosts and replicated by any network participants
- All clients verify identity state before any interaction

Community and Group Ledgers

Communities and groups in Secret also operate as event chains. Each significant change (e.g. member add, role update, channel creation) is an event in the ledger.

- Each community / group maintains its own cryptographic event chain
- All events are hash-linked and digitally signed
- Community / group timelines are validated entirely on the client
- No reliance on central authority or trusted servers
- Tamper-evident history of key and state changes
- Community / group ledgers are primarily managed by Content Hosts and replicated by any network participants
- All clients verify community / group state before any interaction

How Secret's Architecture Aligns with Blockchain Principles

Secret implements a blockchain-based model tailored to decentralized communication — where each identity, community, and group maintains its own tamper-evident chain of signed events. While it doesn't rely on a single global ledger or traditional consensus mechanisms, it faithfully applies core blockchain principles such as **immutability, cryptographic integrity, decentralized replication, and local verification**.

3. Host Infrastructure

Secret's decentralized backend is powered by a diverse network of independently operated nodes called hosts. These hosts fulfill specialized roles such as content storage, identity resolution, traffic relaying, and centralized coordination services. Each host type is designed to operate with minimal trust assumptions, zero access to private user data, and opt-in participation by any user or organization.

Any user running a Secret client can also act as a host, contributing to the resilience, privacy, and decentralization of the network.

Overview of Host Types

Secret defines four primary host types:

1. **Foundation Host** – Handles essential public services that cannot be effectively decentralized, including push notification delivery, content and abuse reporting, and indexing of publicly searchable data like user profiles and communities. Foundation Hosts also operate the platform's dApp and visual assets marketplaces, which are integrated into the Secret client experience.
2. **Authority Host** – Manages decentralized identity infrastructure. This includes maintaining the identity registry, verifying identity claims via cryptographic signatures, and coordinating identity relationships between network participants.
3. **Content Host** – Stores and serves encrypted user data, including user account backups, private messages, media, and community content. Content Hosts can be run by any user directly from their client to support decentralized data persistence.
4. **Relay Host** – Anonymously forwards encrypted traffic to obscure sender/receiver metadata and may optionally serve cached content if the original content host is unavailable. Relay functionality can be enabled directly from the Secret client, allowing any user to contribute to message obfuscation and delivery.

Each host type performs a clearly defined set of responsibilities, operates independently, and never has access to decrypted user content.

3.1 Foundation Host

The Foundation Host is a centralized yet transparent infrastructure component operated by the Secret Foundation. It provides a set of critical services that are either impractical to decentralize or necessary to maintain a consistent user experience across the network. While it does not handle any private user data, the Foundation Host serves as a coordination layer between users, hosts, and communities.

Core Responsibilities

- **Push Notification Delivery**: Facilitates real-time notifications for message delivery, mentions, and system events across mobile and desktop clients.
- **Public Data Indexing**: Indexes communities and user profiles that have been explicitly marked as public. This enables users to discover new communities and profiles through search features in the client app.
- **Visual Assets Marketplace**: Hosts a marketplace where users can browse, purchase, and use various visual assets across the app — such as purchasable sticker packs, gifts, custom themes, and more. Visual assets are installed and used on a per-user basis.

- **dApp Marketplace**: Enables developers to publish decentralized applications that users can purchase for personal use or install into communities. These dApps extend client functionality to support different use cases.
- **Abuse Reporting and Feedback Handling**: Handles user-submitted reports related to content, communities, and hosts. Also collects user feedback, contributing to the health and governance of the ecosystem.

Trust Model

Foundation Hosts do not have access to any private user data and do not participate in message delivery or encryption. All services they provide — including search, notifications, and marketplaces — are optional and designed to enhance user experience without compromising privacy or decentralization.

3.2 Authority Host

The Authority Host is a decentralized infrastructure component responsible for managing identity within the Secret network. It acts as a trust anchor, maintaining a distributed identity registry, verifying identity claims, and enabling secure interactions between clients and hosts — all without relying on centralized accounts or personal information.

Core Responsibilities

- **Decentralized Identity Registry**: Maintains a distributed record of public keys and associated identity metadata used to authenticate users and hosts.
- **Identity Verification**: Validates cryptographic identity claims for both clients and infrastructure nodes, ensuring authenticity without revealing private information.
- **Identity Resolution**: Allows clients to securely resolve identifiers to initiate encrypted communication or validate peer trust.
- **Key Lifecycle Management**: Supports key registration, rotation, revocation, and linking of multiple devices to a single identity.
- **Host Coordination**: Synchronizes identity state and trust relationships with other Authority Hosts, enabling decentralized consensus over valid identities.

Trust Model

Authority Hosts never access private messages, user content, or encryption keys. Their role is strictly limited to identity verification and public key management.

While users can query any available Authority Host, they do not inherently trust the host's response. Instead, each client independently verifies identities using a signed identity timeline, where every identity change is authenticated by the user's own cryptographic keys. This model eliminates reliance on centralized or third-party trust.

To become an Authority Host, an operator must stake Secret's native token, reflecting the critical responsibility of the role. Misbehavior — such as going offline or serving falsified identity data — results in slashing of the staked tokens.

All Authority Hosts replicate identity data among each other to ensure availability, consistency, and fault tolerance — with trust enforced through cryptographic proofs and economic penalties, not assumptions.

3.3 Content Host

The Content Host is a decentralized infrastructure service responsible for storing and delivering encrypted user and groups/community data. It ensures data persistence, asynchronous message delivery, and availability of communities and groups.

Unlike centralized storage systems, Content Hosts never access or interpret the data they store — all content is end-to-end encrypted and controlled by the user.

Core Responsibilities

- **Encrypted Data Storage**: Stores user account data, private messages, media files, and community or group content — all encrypted and cryptographically signed. Hosts cannot read or alter any of the data they serve.
- **Message and Content Delivery**: Acts as a delivery layer for encrypted messages, invitations, and updates when recipients are offline or unavailable.
- **Data Availability and Retrieval**: Serves requested content to authorized users or client devices based on access control rules. Ensures that data is available even if the original sender is offline.

Trust Model

Content Hosts are untrusted by design — all data retrieved from them is verified on the client using digital signatures. There are two types of hosts:

- **Verified Content Hosts** — must stake Secret's native token and are publicly listed in the app. They are subject to uptime, correctness, and reliability requirements, and may be slashed for misbehavior.
- **Unverified Content Hosts** — can be operated by anyone but are not discoverable in the default UI. Users must manually input their details to use them.

Deployment Options

Content Hosts can be enabled directly from the Secret client, allowing any user to participate in hosting. Alternatively, users or organizations can deploy standalone Content Hosts on cloud providers like AWS, GCP, or Azure, offering flexible scalability and infrastructure control.

3.4 Relay Host

The Relay Host is a lightweight, decentralized service designed to anonymize message routing and ensure content availability even when original sources are offline. Relay Hosts enhance privacy and censorship resistance by acting as intermediary nodes for encrypted traffic across the Secret network.

Any user running a Secret client can enable Relay Host functionality or deploy a standalone relay on custom infrastructure.

Core Responsibilities

- **Traffic Relaying**: Forwards end-to-end encrypted messages between clients using multi-hop routing. This helps obscure the origin and destination of communication, making metadata harder to analyze.
- **Sender Anonymity**: Acts as a blind transport layer. Relay Hosts cannot see, decrypt, or associate messages with user identities — they only pass encrypted payloads along a defined path.
- **Content Caching**: Can cache previously requested public content (e.g., communities, group metadata) and serve it when the original Content Host is unavailable. This supports fault tolerance and limited offline access.

Deployment and Participation

- Relay Hosts can be enabled directly in the Secret client, allowing any user to contribute to message routing and network privacy.
- Alternatively, standalone Relay Hosts can be deployed on servers or cloud infrastructure.
- Relays require minimal resources compared to other host types, making them ideal for low-cost, distributed participation.

Trust Model

- All traffic through Relay Hosts is fully encrypted.
- Relays do not store or inspect any forwarded message content.
- They cannot link sender and receiver identities, as routing information is encrypted and context-free.

Relay Hosts are not involved in content verification or identity resolution — they are designed to amplify anonymity and availability, not act as trusted intermediaries.

4. Privacy and Security Architecture

This section explores how Secret achieves strong guarantees around confidentiality, anonymity, and data integrity, even in a decentralized and adversarial environment. Security is enforced not

through central control, but through cryptography, local verification, and intentional protocol design.

Secret's privacy model is built on the following foundational pillars:

4.1 End-to-End Encryption

All user communication — including messages, media, metadata (where possible), and content — is encrypted on the sender's device and only decrypted on the recipient's device.

- **Encryption Schemes**: Secret uses asymmetric and symmetric cryptography to protect both one-to-one and group conversations.
- **Media Protection**: All attachments are encrypted with symmetric keys and referenced securely within the message payload.
- **Forward Secrecy**: Session keys are rotated periodically to minimize the impact of key compromise.

No server, host, or relay can ever access unencrypted message content — encryption is enforced at the client level.

4.2 Local Key Management

Each user's identity and encryption keys are generated and stored locally on their device, ensuring full custody and control.

- Private keys are never transmitted or exposed to any external service.
- All cryptographic operations (encryption, decryption, signing) happen on the client.
- Users can optionally sign in using a mnemonic phrase compatible with Ethereum wallets, enabling consistent identity across Secret and other Web3 services.
- Multi-device support is handled by key linking and identity timelines.

This local-first approach removes the need for centralized key servers and eliminates most traditional attack surfaces.

4.3 Metadata Obfuscation

To protect user activity patterns, Secret incorporates techniques to minimize metadata exposure:

- **Relay-based Routing**: Messages can be sent through multiple Relay Hosts, hiding sender and recipient IPs from intermediaries.
- **No Global Usernames**: Identities are cryptographic and resolved locally, reducing traceability and scraping risk.

- **Encrypted Routing Headers**: Relays only see the next hop — they do not know who sent or will receive the message.

These strategies defend against traffic analysis, surveillance, and other forms of passive observation.

4.4 Zero Trust Infrastructure

All infrastructure in Secret — including Foundation, Authority, Content, and Relay Hosts — is considered untrusted by default.

- All retrieved data is verified using digital signatures.
- Authority Hosts may be slashed for misbehavior; Content Hosts never see decrypted data; Relays never access metadata.
- Even if a host is malicious or compromised, it cannot forge or alter user data without detection.

This model shifts trust away from infrastructure and toward cryptographic proof and local validation.

5. Token Economy

The Secret token powers a multi-layered economy that aligns incentives between users, hosts, relays, and the protocol itself. It enables decentralized monetization, participation rewards, service access, and governance — all within a privacy-first architecture where transactions remain secure and non-custodial.

Secret's token economy is structured across three pillars:

5.1 User as Consumer Economy

Users can earn and spend tokens directly within the Secret app, creating a self-contained economy for communication, content, and interaction.

Earning Tokens

- Create and sell visual assets: Design custom sticker packs, gifts, themes, and sell them in the marketplace.
- Create and sell dApps: Publish decentralized applications in the dApp marketplace for individual or community use.
- Receive tips: Accept token tips as appreciation for messages or contributions.
- Create paid communities or channels: Charge a one-time fee or recurring subscription.
- Complete engagement challenges: Earn tokens for achieving milestones (e.g., grow a community).

- Accept paid interactions: Restrict access to messages or calls unless a token fee is paid.

Spending Tokens

- Buy sticker packs: Purchase creative packs from other users.
- Buy and use dApps: Purchase decentralized applications for personal use or install them into communities to enhance functionality.
- Tip users or messages: Send small token rewards in social or communities.
- Join paid communities: Access exclusive spaces by paying an entry fee or subscribing.
- Pay for interactions: Unlock messaging or calls with users who require payment.
- Vote on features: Influence platform development through token-based governance.
- Place ads: Promote content or communities via token-backed ad placements.

Wallet Features

- Each user has a built-in crypto wallet to buy, sell, swap, and bridge tokens, enabling seamless Web3 interactions inside the app.

5.2 User as Content Host or Relay

Users can contribute infrastructure by hosting data or relaying messages — and earn tokens in return.

Content Hosting

- Host user data: Serve encrypted account, group, or community data.
- Customize limits: Define caps for file size, message count, retention, and more.
- Set pricing models: Offer free, one-time, or subscription-based hosting plans.
- Monetize with ads: Display ads in hosted communities and earn from engagement.

Verification & Staking

- Verified hosts must stake Secret tokens to appear in public listings.
- Hosts are ranked based on stake, uptime, performance, and community trust.
- Misbehavior (e.g., going offline or serving bad data) leads to slashing of staked tokens.

Content Relaying

- Run a relay node to forward encrypted messages or cached data anonymously.
- Earn tokens by contributing to network privacy and message availability.

5.3 Secret Foundation Economy

The Secret Foundation earns and redistributes tokens to support the protocol’s long-term health, while enabling optional monetization services.

Infrastructure Provider

- Free official content hosting, optionally monetized with:
    - Ads in hosted communities
    - Paid upgrades for file size, retention, and access control
- Wallet transaction fees: Small commission on swaps, bridges, and payments
- Verification listing fees: Processing fee for host verification requests
- Slashing rewards: Confiscated tokens from misbehaving hosts

Platform Operator

- Commissions on user transactions, including:
    - Tips, visual asset and dApp purchases, and paid interactions
- Revenue share from hosted community payments
- Ad network cuts: Earnings from ads placed via third-party content hosts
- Paid promotion slots: Featured placement in search, discovery, and marketplaces

Ecosystem Coordinator

- Governance token retention: Tokens used to vote on features stay in circulation via the foundation
- Challenge rewards: Distributed to users who complete community growth or crypto engagement tasks

6. Application Layer

The Secret app delivers a secure, modern communication experience while embedding Web3-native features like encrypted wallets, community monetization, and decentralized hosting. This section outlines the core functionality users interact with, how it integrates with the protocol, and how privacy and decentralization are preserved throughout the user experience.

6.1 Cross-Platform Client Experience

The Secret client is available for iOS, Android, Windows, macOS, and Linux, offering a consistent and polished user experience across devices.

- Responsive design with light and dark modes
- Multi-device sync via secure key linking
- Offline functionality: works over local/private networks without internet access
- Local caching and deferred synchronization
- Full feature parity between desktop and mobile apps

6.2 Messaging and Media

Secret supports secure, expressive communication in one-on-one and community/group conversations.

- End-to-end encrypted messaging
- Rich media support: images, video, voice, documents
- Message features: replies, reactions, threads, polls
- Encrypted audio and video calls
- Conversation folders and organization
- Stickers, emojis, and custom assets

Messages are stored locally or via encrypted Content Hosts and are always cryptographically signed and verified.

6.3 Communities and Channels

Secret's community model allows users to create public or private spaces, each with their own data hosting, access control, and moderation rules.

- Public communities
- Private communities
- Join policy: anyone, by request, or invite-only
- Per-community Content Host selection

6.4 Roles, Permissions, and Moderation

Each community can define a custom governance structure.

- Role-based access control (e.g., admin, moderator, member)
- Channel-specific permissions
- Ban and block systems for moderation
- Moderation logs and ban lists
- Anonymous community joining option
- Per-community profile customization

6.5 dApp Integration

Secret includes a powerful decentralized app (dApp) framework built into the client.

- Users can purchase and run dApps in their personal app interface
- Communities can install scoped dApps per community
- dApps extend functionality (e.g., games, bots, analytics, governance tools)

- dApps are distributed through the dApp marketplace

6.6 Personalization and Expression

Secret allows users to shape how they appear and express themselves across the app — both globally and within individual communities. Whether users prefer complete anonymity, playful customization, or consistent branding, Secret provides full control over identity, presence, and style.

- Per-Community Profiles
- User Avatars and Themes
- Story Publishing with Visibility Control
- Status and Presence Customization
- Stickers and Reactions

These personalization features are designed to give users control over how they engage with others — while respecting privacy, context, and choice.

6.7 Wallet and Payments

Each user account includes a built-in, non-custodial crypto wallet.

- Send and receive Secret tokens and other supported assets
- Buy, sell, swap, and bridge crypto within the app
- Tip users, purchase content, join paid communities
- Interact with dApps or gated messages that require payment
- All keys are stored locally and tied to user identity

6.8 Identity and Anonymity

Identity in Secret is flexible, privacy-focused, and cryptographically secure.

- No phone numbers, emails, or KYC required
- Users onboard via a locally generated keypair or Ethereum-compatible mnemonic
- Clients maintain an identity timeline for key changes and device linking
- Identities can be pseudonymous or tied to a public profile
- Users can join communities anonymously

7. Developer and Operator Ecosystem

Secret is not just a platform for users — it's an extensible, decentralized ecosystem that invites developers, host operators, and third-party service providers to contribute directly to its growth. This section outlines how developers can build on Secret, how operators can run infrastructure, and how both can participate in the token economy.

7.1 dApp Development and Marketplace Integration

Developers can create and distribute decentralized applications (dApps) that plug directly into the Secret client — for either personal use or community integration.

- Cross-platform runtime: dApps run seamlessly in mobile and desktop clients.
- Community-scope support: dApps can be installed into communities and accessed by all members.
- API access: dApps can interact with messaging, user state, community data, and token payments — but can only access personal user data that has been explicitly and deliberately shared by the user
- Marketplace distribution: dApps are published and monetized through the Foundation Host's marketplace.
- Revenue models: Developers can offer free, paid, subscription-based, or token-gated access to dApps.

7.2 Content and Relay Host Deployment

Any user or organization can operate Secret infrastructure to support the network and earn tokens.

- Client-hosted mode: Enable Content or Relay Host functionality directly in the app.
- Standalone mode: Deploy full Content or Relay Hosts on custom infrastructure (e.g., VPS, cloud).
- Configurable behavior: Choose data types to host (accounts, communities, groups) and set limits.
- Monetization support: Define pricing, retention policies, and ad settings.

Hosting Secret infrastructure earns tokens through usage-based payouts, subscriptions, or ad revenue.

7.3 Authority Host Participation

Authority Hosts participate in decentralized identity coordination and must stake tokens to join.

- Run your own Authority Host by staking the native token
- Replicate and sync identity data with other authority nodes
- Remain accountable through slashing mechanisms for downtime or misbehavior

Operators of Authority Hosts contribute to the integrity of the network's identity layer.

7.4 SDKs, APIs, and Tooling (Planned)

While Secret does not yet offer official SDKs or public APIs, a full developer toolkit is planned to support dApp creation, host integration, and client extensibility.

Planned components include:

- Client SDK for building cross-platform dApps and UI extensions
- Host APIs to interact with Content, Relay, and Authority hosts
- Wallet API for secure token operations
- Identity API to query and verify decentralized identity timelines
- Plugin architecture for extending client and community functionality

These tools will adhere to Secret's privacy and security model, with all critical operations (signing, encryption) handled locally by the user's device.

7.5 Community Governance and Funding

Secret supports decentralized development and community-driven growth:

- **Feature voting**: Users spend tokens to vote on roadmap priorities
- **Bounties and grants**: Foundation-funded opportunities for infrastructure, dApps, tools, and research
- **Host rewards**: Verified hosts may receive bonuses based on uptime, reliability, and usage
- **Open governance (future)**: Protocol-level decisions can be made through token-based proposals

This model enables an ecosystem where developers and operators build sustainable, privacy-first infrastructure without relying on centralized gatekeepers.

8. Competitive Analysis

Secret exists at the intersection of secure messaging, decentralized infrastructure, and Web3-native economies. While several platforms address subsets of these features, none combine privacy, modular hosting, offline-first architecture, and a token-driven economy in the same unified system.

8.1 Comparative Overview

Feature / Platform	Secret	Signal	Matrix / Element	Status	Discord	Telegram
End-to-End Encryption	✅ Default	✅	✅	✅	❌	✅(private chats only)
No PII Required	✅ Fully optional	❌(phone required)	✅	✅	❌	❌(phone required)
Decentralized Infrastructure	✅ Multi-host model	❌	✅Federation	✅P2P	❌	❌Centralized
Client-Side Hosting	✅ Yes	❌	❌	❌	❌	❌
Metadata Obfuscation	✅ Encrypted routing, relays	Limited	Limited	Some	❌	❌
Offline / Local Networking	✅ Mesh & LAN	❌	❌	❌	❌	❌
Multi-Hop Routing	✅ Optional Relays	❌	❌	❌	❌	❌
Native Wallet	✅ Built-in	❌	❌	✅	❌	✅
Token Economy	✅ Earn & spend	❌	❌	✅	❌	✅
dApp Marketplace	✅ Integrated	❌	❌	Limited	❌	❌
Sticker / Gift Economy	✅ Yes	❌	❌	❌	✅	✅
Open Governance	✅ Planned DAO	❌	✅	✅	❌	❌
Custom Content Hosting	✅ Communities, Groups and Users	❌	✅(server based only)	❌	❌	❌
Monetization Tools	✅ Built-in	❌	❌	✅	✅	✅

Note: Matrix refers to the protocol; Element is its most widely used client.

8.2 Key Differentiators of Secret

Secret delivers an unique blend of privacy, customization, and decentralization that stands apart from both traditional messaging apps and emerging Web3 messengers.

- **Hybrid Peer-to-Peer + Federated Hosting**
  - Users can opt into trusted content hosts or run their own directly from the client.
  - Communities and groups choose where to host their content — enabling data autonomy.
- **Client-Operated Infrastructure**
  - Every client can act as a Content Host or Relay Host, supporting message routing and encrypted storage.
  - Users contribute to network health and earn tokens for hosting or relaying.
- **Privacy Without Identity**
  - No signup requirements: users don't need emails, phone numbers, or PII.
  - Supports full anonymity, pseudonymous profiles, or Ethereum-based identity.
- **Modular Token Economy**
  - Earn tokens via content creation, infrastructure contribution, and engagement.
  - Spend tokens on sticker packs, dApps, community access, and ads.
  - Integrated wallet with swap, bridge, and payment support.
- **Offline-First & Local Networking**
  - Functions without internet via LAN, Bluetooth, or Wi-Fi Direct.
  - Ideal for remote, disconnected, or censored environments.
- **Built-in Marketplaces**
  - Users can buy/sell visual assets and dApps.
  - Communities can install dApps to extend community features.
- **Pluggable Governance & Monetization**
  - Creators can monetize channels and content directly.
  - Upcoming DAO support enables users to vote on features, fund ideas, and shape the protocol.

8.3 Why Secret Is Positioned Differently

Most competitors focus on one pillar:

- Signal: strong encryption, but centralized and phone-dependent.
- Matrix: federation-first, but complex to run and not optimized UX.
- Status: Web3-native, but limited features and slower adoption.
- Discord: feature-rich and community-oriented, but centralized, surveillance-prone, and non-private.
- Telegram: feature-rich, but centralized and ad-driven.

Secret combines all of the above — and more — into a cohesive, user-first platform.

It is not just a messaging app. It's a privacy-respecting social protocol, infrastructure marketplace, and creator economy — all embedded into one system.

9. Roadmap and Future Work

The Secret platform is designed to evolve continuously — with a roadmap driven by privacy innovation, community feedback, and decentralized governance. Each stage deepens the platform's privacy guarantees, user empowerment, and decentralized infrastructure.

Phase 1: Core Product Development

- Finalize foundational features for messaging, communities, content hosting, and wallet
- Launch private mobile beta (iOS and Android)
- Conduct third-party security audits
- Prepare for app store submissions and early PR campaigns

Phase 2: Mobile Launch & Growth Acceleration

- Official public release of the Secret mobile app
- Introduce major new features:
    - Audio/video calls
    - Expanded token and coin support in wallet
    - Initial dApp support
- Launch marketing and user acquisition campaigns targeting:
    - Influencers and creators
    - Communities on Discord, Telegram, Reddit
- Begin onboarding of early content hosts and relayers

Phase 3: Platform Expansion & Global Reach

- Develop and test desktop client (Windows, macOS, Linux)
- Launch private beta for desktop version
- Localize the app for regional growth (languages, UX customizations)
- Officially launch Secret desktop client to the public

Phase 4: Token & Economy Launch

- Introduce the native Secret Token
- Enable in-app token usage for:
    - Tips
    - Access to premium content
    - Paid messaging and community features

- Launch monetization options for creators and hosts:
    - Ad placements
    - Subscriptions and one-time payments
- Release initial ecosystem grants and infrastructure incentives

Phase 5: Decentralized Infrastructure & Governance

- Roll out full support for community-hosted and decentralized channels
- Expand the network of verified content hosts and relayers
- Introduce staking-based verification for infrastructure operators
- Launch DAO-style feature voting and treasury governance
- Open source major components for transparency and ecosystem growth

Phase 6: Long-Term Growth & Market Capture

- Launch community ambassador and referral programs
- Run experiments in paid acquisition and sponsored growth
- Establish long-term monetization models:
    - Creator and community monetization
    - Premium hosting tiers
    - Infrastructure fees
- Expand global reach via privacy-aligned partnerships and developer programs

10. Closing Vision

Secret is more than just a messaging app — it's a new foundation for private, decentralized digital communication. In a world where surveillance, data exploitation, and centralized control are the norm, Secret offers a radically different model: one where users own their identities, control their data, and build communities on their own terms.

Our mission is to redefine how people connect online by delivering a platform that combines:

- Strong encryption and metadata protection
- Decentralized infrastructure anyone can run
- Built-in crypto economy for creators and operators
- User-owned identities with no reliance on personal data
- Extensibility through modular dApps and marketplaces

By putting privacy, freedom, and sovereignty at the core of communication, Secret enables not only safer conversations — but stronger communities, more resilient networks, and a future where users are in control.

We invite developers, creators, hosts, and users around the world to join us in building the next generation of private, censorship-resistant communication.

This document outlines the current technical architecture and design principles of the Secret platform. As the system evolves, future revisions will expand on protocol details, cryptographic advancements, and governance mechanisms.