ATSAM Protocol — Public Security Overview

A Layered Approach to Private Discovery, Mesh Routing, and Content Protection

Raven Protocol Research

May 2026

Contents

Executive Summary

ATSAM is the security protocol that powers Raven, a peer-to-peer messenger designed to work both online and offline. ATSAM is not a single cryptographic algorithm but a **stack of layered protections**, each with a precise job and a precise security claim.

The protocol addresses a problem that mainstream messengers do not solve well: how can two people on nearby phones find each other and exchange messages **without an internet connection**, without revealing **who is talking to whom** to nearby strangers, and without trusting any central server to broker the conversation?

ATSAM's answer is a deliberately layered design. Each layer protects a different aspect of the communication and uses a different mathematical tool. No single claim covers the entire protocol — instead, the security of each layer is stated, proved, and bounded independently. This separation is not just academic hygiene; it is what allows the system to fail gracefully when one component is weakened (for example, when quantum computing eventually defeats classical elliptic-curve cryptography).

This document is a public overview intended for investors, technical partners, and informed users. It describes what ATSAM does, what it does **not** do, and how each layer's guarantees compose into the end-user experience. It deliberately omits implementation details, exact byte layouts, key labels, and other artifacts that would aid attackers but not understanding.

A separate internal specification documents the precise wire formats, security bounds with full proofs, and reference implementation requirements.

1. The Problem ATSAM Solves

1.1 Messaging Without the Internet

Modern instant messaging — WhatsApp, Signal, iMessage, Telegram — assumes constant internet connectivity. The provider's servers route every message, mediate every introduction, and (in most cases) hold the metadata about who is talking to whom. This works extremely well when networks are healthy and the user trusts the provider. It works poorly when:

- The internet is unavailable: natural disasters, infrastructure failures, hostile network conditions, remote areas, or governmental shutdowns.
- The provider is compelled to disclose metadata, even when message content is end-to-end encrypted.
- The user wishes to remain operationally invisible to surveillance that monitors which devices communicate with which servers.

A peer-to-peer mesh messenger — one in which nearby phones relay messages for one another via Bluetooth Low Energy or Wi-Fi Direct — can address all three problems, in principle. But it introduces a new one: **how does device A find device B in the radio environment without broadcasting their identities to everyone within range?**

1.2 The Discovery Trap

Naive solutions fail in characteristic ways.

If A broadcasts "I am Alice" on Bluetooth, anyone within radio range learns Alice's location at that moment. Aggregated over time, this is a tracking nightmare.

If A broadcasts nothing and waits to be addressed, no one can reach her without already knowing her radio identifier — but then that identifier becomes the very tracking surface we tried to avoid.

If A and her friends share a group identifier, then anyone who has ever paired with A (or who has compromised a friend's device) can use that identifier to detect A from outside the trust circle.

If A rotates her identifier on a schedule, the rotation itself becomes a fingerprint, and the rotation key remains shared with everyone in the social graph.

What is needed is a discovery primitive in which A's broadcast is **recognizable to her friends, indistinguishable from random noise to strangers, and unlinkable to any prior or future broadcast** — even by the friends themselves. No widely deployed messenger has this. ATSAM provides it as one of its five layers.

1.3 The Content Trap

Encrypting message content is, in 2026, a solved problem. Signal's Double Ratchet provides forward secrecy and post-compromise security at computational hardness levels that no practical adversary can defeat. AES-GCM, ChaCha20-Poly1305, and similar AEAD constructions are well understood and widely deployed.

But computational security has a structural limit: any computational scheme can in principle be defeated by sufficient computation, including future quantum computation. For the most sensitive subset of messages — say, an encrypted text that must remain confidential for fifty years — even strong AEAD is not enough.

ATSAM addresses this by offering an **opt-in one-time-pad mode** for selected high-sensitivity messages. One-time-pad encryption provides *information-theoretic* secrecy: no amount of computation, classical or quantum, can recover the plaintext from the ciphertext alone. The trade-off is operational: the system must distribute, store, and consume one-time-pad material under strict discipline. ATSAM's Vault Mode handles this discipline, and it is offered alongside (not instead of) standard computational encryption.

2. Design Principles

ATSAM is shaped by five principles that we apply consistently across the stack.

2.1 Layer Separation

No single cryptographic primitive can solve the entire problem. We refuse to collapse discovery, authentication, routing, and content encryption into one algorithm with one composite security claim. Instead, each layer has a narrow job, a narrow security guarantee, and a narrow proof.

This makes the system easier to analyze, easier to audit, and easier to repair: if one layer's primitive is weakened by future cryptanalysis, only that layer needs to be replaced.

2.2 Honest Claims

Cryptographic marketing tends toward overclaiming: "unbreakable," "absolutely anonymous," "perfect secrecy." ATSAM avoids this language. Each layer makes a specific claim with specific conditions, and the public-facing user interface mirrors those claims precisely. When a property is *not* provided — for example, ATSAM does not by itself hide message timing or radio traffic volume — we say so plainly.

2.3 Post-Quantum Hybrid by Default

The pairing layer uses both classical elliptic-curve cryptography (X25519) and a post-quantum key-encapsulation mechanism (ML-KEM, NIST FIPS 203). An attacker would need to defeat **both** to break the resulting shared secret. As long as one component remains secure, the pairing remains secure. This is hybrid security in the strict sense: combined strength, not weakest-link strength.

2.4 Composition Beats Novelty

ATSAM does not invent new cryptographic primitives. It composes well-studied building blocks — X25519, ML-KEM, HKDF-SHA256, HMAC-SHA256, the one-time pad, Wegman-Carter universal hashing — in a specific arrangement suited to peer-to-peer mesh messaging. The novelty is in the composition and the surrounding operational discipline, not in any individual block.

2.5 Operational Security Matters

The cleanest mathematical proof is defeated by a stolen phone, a reused nonce, a backup that restores stale state, or a crash that leaves the system in an inconsistent place. ATSAM treats operational concerns — pad storage discipline, ratchet desynchronization recovery, crash-safe state machines — as part of the security boundary, not as implementation afterthoughts.

3. The ATSAM Stack

ATSAM consists of five layers. Each handles one concern. Each composes with the others through well-defined interfaces.

3.1 Layer 1 — Pairing (Establishing a Shared Root Secret)

What it does. When two users first establish a trust relationship — for example, when they meet in person and verify each other's identity — the pairing layer creates a shared root secret that only their two devices know.

How it works (at a high level). The two devices perform a classical Diffie-Hellman exchange over the X25519 elliptic curve, and in parallel exchange a post-quantum key encapsulation using ML-KEM-768. The two resulting shared values are combined through HKDF into a single root key.

Why it matters. This is the bedrock from which every other key in the stack is derived. If the pairing fails or is compromised, nothing downstream is safe. If the pairing succeeds, the two devices have a shared 256-bit secret that no observer — passive or active, classical or quantum — can recover, provided at least one of the two underlying mechanisms (X25519 or ML-KEM) remains hard.

Honest scope. Pairing is computational security, not information-theoretic. If both X25519 *and* ML-KEM are simultaneously broken by some future breakthrough, the root secret is at risk. The hybrid construction makes this dramatically less likely than relying on either alone.

3.2 Layer 2 — Private Discovery (Knowing a Friend Is Nearby)

What it does. Each device broadcasts short radio beacons every few minutes. Friends of the device recognize the beacons. Strangers see only random-looking bytes. Beacons cannot be linked across time, even by friends.

How it works (at a high level). From the pairing root, each pair of friends derives a discovery key. Every few minutes, the broadcasting device computes a small "slot" value from this key combined with the current epoch and a fresh random nonce. The slot is placed in a fixed-size beacon along with random padding, then shuffled. A friend who has the same discovery key can recompute the same slot value and find it in the beacon; a stranger cannot.

Why it matters. This is the key privacy improvement over conventional Bluetooth discovery. The device's identity, social graph density, and trust relationships are not visible to anyone except its actual friends. The beacon does not contain a name, a phone number, or a public key. To a passive observer, every beacon looks like a 244-byte random string.

Honest scope. Discovery is *candidate* discovery only. A beacon match means "a paired peer claims to be nearby." It does *not* prove the claimed peer is actually nearby and live — for that, see Layer 3. The discovery layer can be tricked by a recording-and-replay attacker; this is handled at the next layer.

3.3 Layer 3 — Live Confirmation (Verifying the Peer Is Really There)

What it does. When a device thinks it has detected a friend's beacon, it issues a fresh challenge and verifies that the friend can answer it in real time. Only after the challenge is answered does the application treat the peer as present.

How it works (at a high level). From the pairing root, a separate live-confirmation key is derived. The challenger sends a random value tied to the beacon it just observed; the responder must produce a message authentication code over that value plus the beacon's nonce. A replayed old beacon cannot help an attacker, because the challenge is fresh.

Why it matters. Without this layer, an attacker who records a friend's beacon at one location and replays it at another could trick the user interface into showing "your friend is nearby" when the friend is in fact thousands of miles away. The live-confirmation layer makes this attack fail.

Honest scope. Live confirmation establishes that the responder is *currently in radio range and holds the pairing key.* It does not, by itself, defend against relay attacks where an adversary forwards the challenge and response over a fast network. A separate distance-bounding mechanism (planned for a future version) would address this.

3.4 Layer 4 — Encrypted Mesh Routing (Hiding Who Talks to Whom)

What it does. When messages travel through a mesh of relays — phones forwarding messages on behalf of others — the relays should not know the sender or recipient identity of any message they carry.

How it works (at a high level). From the pairing root, each pair derives a routing key. Each message carries a per-message recipient tag derived from this key plus a fresh per-message nonce. Each recipient device, upon receiving a message envelope, tries to match the tag against all of its own routing keys; if any matches, the device decrypts the message. Relays see only the tag — never a username, public key, or stable identifier.

Why it matters. This is the metadata-privacy property for offline traffic. A node in the mesh that forwards messages cannot link successive envelopes to the same recipient, cannot identify who the original sender was, and cannot map the social graph from observed traffic alone.

Honest scope. Mesh routing protects against per-message linkability via stable identifiers. It does *not* hide the existence of traffic, the size of messages, or the radio timing pattern. A global passive observer who watches all radio traffic can still perform timing analysis. Defending against that requires cover traffic and traffic shaping, which is a separate concern and a separate engineering investment.

3.5 Layer 5 — Vault Mode (One-Time-Pad Content Encryption)

What it does. For high-sensitivity text messages, the user can opt into Vault Mode: the message is encrypted with a one-time pad rather than a computational cipher. This provides *information-theoretic secrecy* — meaning that no amount of computation, classical or quantum, can recover the plaintext from the ciphertext.

How it works (at a high level). Pairs of devices that have opted into Vault Mode establish a small pad bundle through the regular pairing channel. Each direction (A→B and B→A) gets its own independent pad region. When the user marks a message as "vault," the next bytes of pad are XORed with the message; an information-theoretic message authentication code is computed using a separate region of pad-derived key material. The consumed pad bytes are then wiped from the device.

Why it matters. Computational encryption assumes the adversary cannot perform certain mathematical operations within reasonable time. The one-time pad makes no such assumption: a perfectly

random, secret, one-time key gives mathematical certainty of secrecy, conditioned only on the key staying secret and not being reused.

Honest scope. Vault Mode is information-theoretic only at the message layer, and only when the pad conditions are met: the pad must be uniformly random, the pad slice must be at least as long as the message, the slice must be used exactly once, and consumed bytes must be securely wiped. The pad establishment, storage, and crash-safety are *computational* and engineering-dependent. Vault Mode also reveals message size (in coarse buckets) and is not intended to hide social-graph metadata; the discovery and routing layers handle that separately. It is intended for the *most sensitive subset* of a user's messages, not as the default for everything.

4. Threat Model

Different adversaries can do different things. The honest way to discuss security is to enumerate them.

4.1 Adversaries We Defend Against

Nearby stranger sniffing radio. Sees Bluetooth or Wi-Fi traffic in the local area but has no relationship with the user. Sees only random-looking beacons; gets no name, phone number, public key, or social-graph evidence from the protocol itself. (Radio traffic volume and timing are not hidden.)

Replay attacker. Records old beacons and tries to spoof presence. Defeated by the live-confirmation challenge layer.

Mesh relay carrying messages. Forwards envelopes between users but is not part of either user's trust circle. Sees per-message routing tags but cannot link them across messages and cannot identify sender or recipient.

Centralized server or cloud provider. Could in principle store metadata, log access patterns, or be compelled to disclose. Because ATSAM operates fully peer-to-peer for offline traffic, no central server exists to compel. For online traffic, server-side metadata is minimized by the routing layer.

Future quantum attacker. Defeats classical elliptic-curve cryptography (X25519) via Shor's algorithm. The hybrid pairing means the seed is protected by ML-KEM in addition to X25519. Vault Mode provides full information-theoretic content secrecy that is unaffected by quantum advances.

Backup rollback attacker. Restores an old device backup in an attempt to reuse a pad slice or a ratchet state. Defeated by atomic cursor persistence and monotonic state guards (operational defenses).

4.2 Adversaries We Do Not Fully Defend Against

Endpoint compromise. A jailbroken phone, sophisticated malware with root access, or a device that has been physically captured. ATSAM cannot prevent the attacker from reading whatever the legitimate user can read on that device. Endpoint security is the user's responsibility (and the operating system's). ATSAM does its best to minimize what is stored at rest, and to encrypt what must be stored.

Global passive observer of all radio. An adversary with the ability to monitor every Bluetooth and Wi-Fi packet in a wide area can still see traffic patterns: who emits when, who emits how much, who is nearby whom. Defending against this requires cover traffic, padding, and traffic shaping. This is a planned future improvement; the current ATSAM stack does not claim to defeat global traffic-shape analysis.

Active mafia-fraud relay attack. An adversary with two high-speed devices, one near A and one near B, can in principle forward the live-confirmation challenge faster than physical proximity would allow. Defending against this requires distance bounding based on radio signal characteristics, which is a planned future layer.

Social-engineering pairing. ATSAM provides a strong cryptographic pairing primitive, but it cannot prevent the user from pairing with the wrong person (e.g., an impostor who looks like a

friend). Identity verification at pairing time — typically through in-person QR code exchange or shared secret comparison — remains the user's responsibility.

4.3 Risk Stratification

We classify risks into three tiers:

- **Cryptographically defeated.** Properties for which we have a formal security proof under standard assumptions. Pairing IND-secrecy, discovery indistinguishability, live-confirmation replay resistance, routing-tag unlinkability, and Vault Mode content confidentiality fall here.
- **Operationally defended.** Properties that depend on correct implementation discipline. Pad storage integrity, cursor non-reuse across crashes, backup rollback prevention. These have engineering tests but not cryptographic theorems.
- **Acknowledged residual risk.** Properties we do not yet protect against. Global radio traffic analysis, full distance-bounding for relay attacks, side-channel attacks on cryptographic implementations.

The product user interface reflects this stratification: claims are made only where defended, and limitations are stated where they apply.

5. Honest Comparison to Existing Systems

The closest related work falls into four categories.

5.1 Signal / WhatsApp / iMessage

Mainstream end-to-end encrypted messengers provide excellent computational confidentiality and forward secrecy via the Double Ratchet protocol. They do not address offline mesh discovery or routing, and they require centralized servers for delivery and metadata mediation. ATSAM is complementary, not competing: a phone running Raven could also run Signal. Vault Mode offers information-theoretic content protection that no widely deployed messenger offers.

5.2 Briar, Berty, Bridgefy

Existing peer-to-peer mesh messengers provide offline operation over Bluetooth and similar transports. Each makes specific trade-offs around discovery and routing privacy. ATSAM's distinguishing contribution is the layered design with explicit, separately-proved security claims for each layer, the post-quantum hybrid pairing, and the optional information-theoretic Vault Mode. We believe this is the first peer-to-peer mesh messaging protocol to combine all of these properties with publicly documented security proofs.

5.3 Apple Find My Network

Apple's Find My uses a related but centrally-mediated bilateral beacon scheme. AirTags emit rotating beacons; crowd-sourced iPhones report observations to Apple's servers, which forward them to the owner under server-side encryption. The construction is elegant and Apple makes no claims to read the metadata. ATSAM's approach removes the central authority entirely: no server, no crowd-sourcing, no proprietary infrastructure.

5.4 Tor, Mixnets

Anonymity networks like Tor and mixnets hide who-talks-to-whom at the internet layer through onion routing. They do not address offline operation or local-radio discovery. ATSAM addresses a strictly different problem and could in principle be used alongside an anonymity overlay.

5.5 Summary of Property Coverage

Property	Signal	Briar	Find My	ATSAM
End-to-end encrypted	Yes	Yes	n/a	Yes
Works offline (mesh)	No	Yes	Partial	Yes
Private peer discovery	n/a	Partial	Yes (centralized)	Yes (P2P)
Post-quantum hybrid	Planned	No	No	Yes
Information-theoretic vault	No	No	No	Yes (optional)
Public formal proofs per layer	Partial	Partial	Proprietary	Yes
Honest scope statement	Yes	Yes	n/a	Yes

We acknowledge other research-grade proposals exist for many of these properties. ATSAM's contribution is bringing them together into an engineered protocol stack suitable for a consumer

product.

6. What ATSAM Does Not Do

A protocol that overclaims is harder to trust than one that is honest about its limits. We list what ATSAM **does not** claim:

- ATSAM does not make Raven unbreakable. No protocol does.
- ATSAM does not hide that two people are talking, only what they say and who exactly is on each end.
- ATSAM does not by itself hide radio traffic timing or volume from a global passive observer.
- ATSAM does not protect against a compromised endpoint, including jailbroken phones, malware with root access, or physical device capture.
- ATSAM does not provide social-graph anonymity in the strong sense (e.g., k-anonymity against linkage attacks). It provides per-broadcast unlinkability against passive observers.
- ATSAM does not replace good operational practices: secure device unlock, software update discipline, physical custody of phones.
- Vault Mode does not extend to media files, voice, or large attachments. It is for small, structured text messages.
- ATSAM's protections degrade if the user's phone is compromised, if the pad bundle is stolen, or if the cryptographic implementations have side-channel flaws.

This list is intentionally explicit. Marketing departments tend to obscure such limits; we choose to state them upfront.

7. Roadmap and Status

ATSAM is under active development. The current state of each layer:

- **Pairing layer (Layer 1).** Mathematical specification complete with formal security proofs. Reference implementation in progress on iOS via Apple CryptoKit and on Android via Google Tink. ML-KEM integration uses NIST FIPS 203 standardized parameters.
- **Discovery layer (Layer 2).** Specification complete with formal security proofs and concrete parameter analysis. Wire format settled. Implementation in progress for iOS; Android and embedded device ports planned.
- **Live confirmation layer (Layer 3).** Specification complete. Implementation pending discovery layer completion.
- **Routing layer (Layer 4).** Specification complete. Implementation pending discovery and confirmation layers.
- **Vault Mode (Layer 5).** Specification complete. Pad transport protocol design complete. Implementation includes pad storage, cursor state machine, and atomic crash-safety logic — substantial engineering work in progress.

We expect a closed beta of layers 1–3 in the next quarter, with layers 4–5 following over the subsequent two quarters. Each layer will be released only after internal review and ideally external security audit.

A separate roadmap covers planned future work:

- **Distance bounding (planned Layer 6).** Defense against active mafia-fraud relay attacks via radio signal characteristic analysis.
- **Cover traffic and traffic shaping (planned Layer 7).** Defense against global passive traffic analysis.
- **Compromise-aware re-pairing (operational extension).** Tools for users to recover gracefully from suspected device compromise.

8. Conclusion

ATSAM is the result of an effort to take seriously two questions simultaneously:

1. *What does it actually take to build a private peer-to-peer messenger that works offline?*
2. *How honest can we be about what such a system can and cannot do?*

The answer to the first question turned out to require five layers, each with a narrow job and a separate cryptographic primitive: post-quantum hybrid pairing, private bilateral discovery, fresh-challenge live confirmation, unlinkable mesh routing, and optional one-time-pad content protection. No existing protocol assembles all five with publicly documented proofs per layer.

The answer to the second question is the honest-claims discipline that runs throughout this document. Each layer states what it protects against, what it does not protect against, and what assumptions it relies on. The product user interface inherits this discipline directly: when ATSAM cannot deliver a property, we do not pretend that it can.

We believe this combination — serious cryptographic foundations plus rigorous honesty about scope — is what privacy-respecting communication infrastructure should look like in 2026 and beyond.

Raven is the application; ATSAM is the protocol that makes Raven trustworthy.

This document is a public overview. A detailed internal specification, including wire formats, security bounds with full proofs, and reference implementation code, is maintained separately under the Raven engineering organization.

For questions or research collaboration, see the project website.