TECHNICAL WHITEPAPER · v1.0 · MAI 2026

FAX

Post-Quantum sicherer Peer-to-Peer-Messenger ohne Server, ohne Account, ohne Tracking.

KEM **ML-KEM-1024** SIGS **ML-DSA-65** CIPHER **AES-GCM-256** STANDARD **NIST FIPS 203/204** TRANSPORT **WebRTC P2P**

INHALT

01 · Zusammenfassung

FAX ist ein browser-basierter, Peer-to-Peer-Messenger, der ohne zentralen Server arbeitet, keine Accounts benötigt und alle Nachrichten Ende-zu-Ende mit einer Hybrid-Konstruktion aus klassischer und Post-Quantum-Kryptografie schützt.

Die App ist auf **NIST FIPS 203/204**-Standards aufgebaut (ML-KEM, ML-DSA) und kombiniert diese mit etablierter Elliptischer-Kurven-Kryptografie (ECDH P-384 und P-256). Die Forward-Secrecy wird über einen Double-Ratchet erreicht, der durch einen Triple-Ratchet-Mechanismus erweitert wurde: alle zehn Nachrichten wird ein neuer Post-Quantum-Schlüssel rotiert, sodass auch bei zukünftigem Schlüsselbruch durch Quantencomputer keine Rückwirkung auf vergangene Nachrichten möglich ist.

Im Unterschied zu klassischen Messengern wie Signal, WhatsApp oder Telegram betreibt FAX **keinen zentralen Nachrichten-Server**. Die Kommunikation läuft direkt zwischen den Geräten der Teilnehmer via WebRTC. Lediglich für den initialen NAT-Traversal-Handshake werden öffentliche STUN-Server und ein eigener TURN-Relay genutzt — beide sehen ausschließlich verschlüsselten Datenverkehr.

WESENTLICHE SICHERHEITSMERKMALE

- **Post-Quantum-Hybrid:** ML-KEM-1024 + ECDH P-384 + ECDH P-256
- **Forward Secrecy & Post-Compromise Security:** Triple Ratchet
- **Traffic-Analyse-Schutz:** Tiered Constant-Length Padding
- **Hardware-gebundene Identitäten:** WebAuthn / Touch ID / Face ID
- **Audit-Trail:** Lokale Merkle-Hash-Chain aller Nachrichten
- **Notfall-Mechanismen:** Duress-PIN und kryptografischer Wipe

02 · Motivation & Threat Model

2.1 Motivation

Die meisten verbreiteten Messenger basieren auf einer zentralen Server-Architektur. Selbst bei Ende-zu-Ende-Verschlüsselung hat der Server Zugang zu **Metadaten**: wer mit wem kommuniziert, wann, wie oft, von welcher IP-Adresse. Diese Metadaten sind in vielen Threat-Modellen sensibler als die Inhalte selbst — sie zeigen soziale Strukturen, Bewegungsmuster und Beziehungen.

Zudem rückt mit dem Fortschritt bei Quantencomputern eine reale Bedrohung in den Fokus: das Konzept *Harvest now, decrypt later*. Angreifer könnten heute verschlüsselte Kommunikation aufzeichnen und in der Zukunft mit Quantencomputern entschlüsseln, sobald die Hardware verfügbar ist. Konventionelle ECC-basierte Verschlüsselung (Curve25519, P-256, P-384) ist gegen den Shor-Algorithmus angreifbar.

2.2 Threat Model

BEDROHUNG	SCHUTZ DURCH FAX
Passive Netzwerk-Überwachung	**Ende-zu-Ende verschlüsselt, keine Metadaten**
Aktiver MITM-Angriff	**Safety-Number-Verifikation, Sicherheits-Codes**
Server-Kompromittierung	**Kein zentraler Server existiert**
Harvest-now-decrypt-later (Quantum)	**ML-KEM-1024 hybrid mit ECDH**
Forward Secrecy	**Double Ratchet, neue Keys pro Message**
Post-Compromise Security	**Triple Ratchet, PQ-Rotation alle 10 Msg.**
Geräte-Verlust	**PIN, Biometrie, Duress-Code, Wipe**
Replay-Angriffe	**1024-Nonce-Sliding-Window**
Traffic-Analyse (Längen)	**Tiered Constant-Length Padding**
Endpoint-Kompromittierung (Malware)	**Außerhalb des Threat-Models**
Browser-Zero-Days	**Außerhalb des Threat-Models**

BEDROHUNG	SCHUTZ DURCH FAX
Physische Hardware-Manipulation	**Außerhalb des Threat-Models**

WICHTIGER HINWEIS

Kein kryptografisches System kann gegen kompromittierte Endgeräte schützen. Wenn das Smartphone oder der Computer eines Nutzers selbst mit Malware infiziert ist oder ungewöhnlicher physischer Zugriff besteht, ist auch das stärkste Verschlüsselungsschema wirkungslos. FAX schützt die Kommunikation zwischen sauberen Endgeräten.

03 · Architektur-Überblick

FAX besteht aus drei minimalen Komponenten: dem Client (eine Single-File-HTML-App), einem TURN-Relay (für NAT-Traversal) und einem Zahlungs-Worker (Cloudflare). Keine dieser Komponenten verarbeitet entschlüsselten Nachrichteninhalt.

3.1 Komponenten

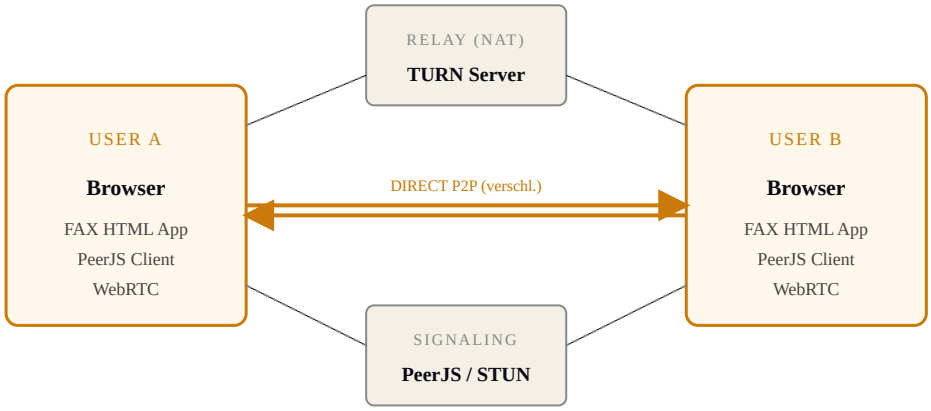

ABB. 3.1 — ARCHITEKTUR-ÜBERBLICK

3.2 Datenfluss

Nach dem initialen Handshake (Signalisierung via PeerJS-Server, NAT-Traversal via STUN/TURN) gehen alle Nachrichten **direkt zwischen den Browsern**. Es gibt keinen FAX-Server, der Nachrichten speichert, weiterleitet oder analysiert.

SCHRITT-FÜR-SCHRITT

1. **Identität:** User A erzeugt eine lokale Identität (ECDH-Keypair + ML-KEM-Keypair + ML-DSA-Keypair) im Browser.
2. **Discovery:** User A teilt seine ID mit User B über einen externen Kanal (QR-Code, Link, Sprachnachricht).
3. **Signaling:** Beide Browser verbinden sich mit dem öffentlichen PeerJS-Signal-Server, um SDP/ICE-Daten auszutauschen.
4. **NAT-Traversal:** Falls direkte Verbindung nicht möglich, leitet der TURN-Server verschlüsselten Traffic weiter (Server sieht nur Ciphertext).

5. **Hybrid-KEX:** Schlüsselaustausch über ECDH P-384, ECDH P-256 und ML-KEM-1024 parallel; die drei Shared Secrets werden via HKDF-SHA-512 zu einem Master-Key kombiniert.
6. **Ratchet-Init:** Aus dem Master-Key werden die initialen Chain-Keys für sendende und empfangende Richtung abgeleitet.
7. **Messaging:** Jede Nachricht wird mit AES-GCM-256 verschlüsselt; der Chain-Key wird per HKDF advanced.
8. **Triple Ratchet:** Alle zehn Messages wird ein frisches ML-KEM-Encapsulation durchgeführt und in den Chain-Key gemischt.

04 · Kryptografische Primitiven

Alle verwendeten Primitive sind etablierte NIST-Standards oder NIST-publizierte Post-Quantum-Algorithmen. Keine selbst entwickelten Algorithmen.

KATEGORIE	PRIMITIVE	STANDARD	SECURITY LEVEL
Symmetric Cipher	`AES-GCM-256`	NIST SP 800-38D	256-Bit
Klassischer KEX	`ECDH P-384`	NIST SP 800-186	192-Bit-äquiv.
Klassischer KEX (2nd)	`ECDH P-256`	NIST SP 800-186	128-Bit-äquiv.
Post-Quantum KEM	`ML-KEM-1024`	NIST FIPS 203	NIST Cat-5 (~AES-256)
Post-Quantum Sigs	`ML-DSA-65`	NIST FIPS 204	NIST Cat-3 (~AES-192)
Key Derivation	`HKDF-SHA-512`	RFC 5869	512-Bit
Hashing	`SHA-256, SHA-512`	NIST FIPS 180-4	256/512-Bit
Password Stretching	`PBKDF2-SHA-512`	NIST SP 800-132	100 000 iter.

4.1 Auswahlbegründung

AES-GCM-256

NSA-Suite-B-zertifiziert für Top-Secret-Daten. Autenticated Encryption (AE) verhindert Manipulationen am Ciphertext. 256-Bit ist der höchste verfügbare AES-Schlüssel und gilt selbst gegen Grover-Suchangriffe von Quantencomputern als ausreichend (effektiv 128-Bit-Quanten-Sicherheit).

ML-KEM-1024

NIST FIPS 203 (August 2024). Module-Lattice-basiertes Key Encapsulation Mechanism. Die **1024-Variante** entspricht NIST Sicherheits-Kategorie 5 — also stärker als AES-256 gegen klassische und Quantenangriffe. Public Keys sind etwa 1568 Bytes, Ciphertexts 1568 Bytes.

HYBRID-KONSTRUKTION

FAX nutzt **nicht ausschließlich** ML-KEM, sondern kombiniert es mit ECDH P-384 und P-256. Falls in der Zukunft ein bisher unbekannter Angriff gegen ML-KEM gefunden wird, bleibt die ECC-Schicht intakt — und umgekehrt schützt ML-KEM gegen den Bruch der ECC durch Quantencomputer. Der gemeinsame Master-Key ist nur sicher, wenn *alle drei* KEX-Komponenten sicher sind.

> HYBRID-GARANTIE
>
> Die Sicherheit der Hybrid-Konstruktion ist mindestens so stark wie die stärkste Einzelkomponente. Ein Angreifer muss *alle drei* Mechanismen brechen, um Klartext zu erhalten.

05 · Handshake & Hybrid Key Exchange

Der initiale Handshake etabliert ein gemeinsames Geheimnis zwischen zwei Geräten, ohne dass irgendeine Drittpartei dieses Geheimnis ableiten kann — selbst nicht der Signaling-Server.

5.1 Ablauf

```
// Initiator A → Responder B

// Schritt 1: A generiert Ephemeral-Keys
ecdh_a_p384 = ECDH.keygen(P-384)
ecdh_a_p256 = ECDH.keygen(P-256)
kem_a       = ML-KEM-1024.keygen()

// Schritt 2: A sendet öffentliche Schlüssel an B
A → B: { ecdh_a_p384.pub, ecdh_a_p256.pub, kem_a.pub, signature_dsa }

// Schritt 3: B führt parallelen KEX durch
ecdh_b_p384 = ECDH.keygen(P-384)
ecdh_b_p256 = ECDH.keygen(P-256)
ss_p384 = ECDH(ecdh_b_p384.priv, ecdh_a_p384.pub)
ss_p256 = ECDH(ecdh_b_p256.priv, ecdh_a_p256.pub)
(ct, ss_kem) = ML-KEM-1024.encaps(kem_a.pub)

// Schritt 4: B sendet seinen Anteil + KEM-Ciphertext
B → A: { ecdh_b_p384.pub, ecdh_b_p256.pub, ct, signature_dsa }

// Schritt 5: A entschlüsselt KEM-Ciphertext
ss_kem = ML-KEM-1024.decaps(ct, kem_a.priv)
ss_p384 = ECDH(ecdh_a_p384.priv, ecdh_b_p384.pub)
ss_p256 = ECDH(ecdh_a_p256.priv, ecdh_b_p256.pub)

// Schritt 6: Master-Key Ableitung
master_key = HKDF-SHA-512(
  ikm  = ss_p384 || ss_p256 || ss_kem,
  salt = SHA-512(transcript),
  info = "FAX-KEX-v17-MASTER"
)

// Schritt 7: Initiale Chain-Keys
send_ck_A = HKDF(master_key, "FAX-SEND-A→B")
recv_ck_A = HKDF(master_key, "FAX-SEND-B→A")
file_key  = HKDF(master_key, "FAX-FILE")
```

5.2 Sicherheitseigenschaften

- **Authentifizierung:** Jede Public-Key-Nachricht wird mit dem persistenten ML-DSA-65-Schlüssel signiert, der bei der Identitäts-Erzeugung erstellt wurde.
- **Forward Secrecy:** Alle KEX-Schlüssel sind *ephemeral* — sie werden nach dem Handshake gelöscht und niemals persistiert.
- **Replay-Schutz:** Transcript-Hash im HKDF-Salt verhindert die Wiederverwendung eines früheren Handshakes.
- **Quantum-Hybrid:** Selbst bei Bruch von ECDH (durch Quantencomputer) bleibt `ss_kem` unangreifbar; selbst bei einem unbekannten Angriff auf ML-KEM bleibt ECDH sicher.

5.3 Safety Number

Nach dem Handshake wird eine **Safety Number** berechnet, die beide Nutzer manuell vergleichen können (etwa per Telefonat oder QR-Code-Scan). Diese verhindert einen Man-in-the-Middle-Angriff während des Signalings:

```
safety_number = SHA-256(
  ecdh_a_p384.pub || kem_a.pub ||
  ecdh_b_p384.pub || kem_b.pub
)[0:24]  // erste 24 Bytes als Hex-Tupel
```

06 · Double Ratchet & Triple Ratchet

FAX nutzt den bewährten Signal-Double-Ratchet zur Erzielung von Forward Secrecy auf Nachrichten-Ebene und erweitert ihn um einen Triple-Ratchet, der periodisch frisches Post-Quantum-Schlüsselmaterial einbringt.

6.1 Symmetric (Sending) Ratchet

Pro Nachricht wird der aktuelle `chain_key` wie folgt avanciert:

```
message_key = HKDF(chain_key, "FAX-MK")
next_chain  = HKDF(chain_key, "FAX-CK-NEXT")
chain_key   = next_chain  // alter chain_key wird verworfen

ciphertext  = AES-GCM-256(message_key, iv, plaintext)
```

6.2 Triple Ratchet (Post-Quantum Rotation)

Alle zehn Nachrichten in eine Richtung wird zusätzlich eine neue ML-KEM-Encapsulation gegen den persistierenden ML-KEM-Public-Key des Gegenübers ausgeführt:

```
// Sender (alle 10 Messages)
(pq_ct, pq_ss) = ML-KEM-1024.encaps(peer_pq_pub)
chain_key      = HKDF(chain_key, pq_ss, "FAX-TRIPLE-v17")

// Nachricht enthält zusätzlich pq_ct
packet = { iv, ciphertext, pq_ct }

// Empfänger
pq_ss     = ML-KEM-1024.decaps(pq_ct, my_pq_priv)
chain_key = HKDF(chain_key, pq_ss, "FAX-TRIPLE-v17")
```

> **WARUM ALLE 10 NACHRICHTEN?**
>
> ML-KEM-Ciphertexte sind etwa 1,5 KB groß. Eine Rotation bei jeder Nachricht würde 1,5 KB Overhead pro Nachricht bedeuten. Die Rotation alle 10 Nachrichten bietet einen pragmatischen Kompromiss zwischen Sicherheit (selbst kompromittierte Chain-Keys werden innerhalb von 10 Messages "geheilt") und Bandbreite (~150 Byte Overhead im Mittel).

6.3 Garantien

- **Forward Secrecy:** Kompromittierung eines `chain_key` ermöglicht keine Entschlüsselung früherer Nachrichten — die alten Chain-Keys wurden verworfen.
- **Post-Compromise Security:** Nach einer Kompromittierung wird der Chain durch den nächsten Triple-Ratchet wieder undurchsichtig für den Angreifer.
- **PQ-Forward-Secrecy:** Selbst wenn ein zukünftiger Quantencomputer die ML-KEM-Public-Keys bricht, sind die zwischenzeitlich verworfenen Chain-Keys nicht rekonstruierbar.

07 · Transport & Netzwerk

FAX nutzt WebRTC für die direkte Peer-to-Peer-Kommunikation. Die WebRTC-eigene DTLS-SRTP-Schicht bildet einen zusätzlichen Sicherheits-Layer unter der App-Schicht-Verschlüsselung.

7.1 WebRTC-Stack

LAYER	SCHUTZ
FAX-App-Layer	Hybrid PQ + Triple Ratchet + AES-GCM-256
DataChannel (SCTP)	WebRTC-DTLS-Wrapping
WebRTC-Transport	DTLS-SRTP (TLS 1.2/1.3)
IP / UDP	—

7.2 NAT-Traversal

Zwischen Peers existieren in der Praxis oft NAT-Geräte (Router, Firewalls), die direkte UDP-Verbindungen blockieren. FAX adressiert das in drei Eskalationsstufen:

1. **Direkter UDP-Verbindung** via ICE-Hole-Punching (95% der Fälle).
2. **STUN-assistierte Verbindung** mit öffentlichen STUN-Servern (Google, Cloudflare).
3. **TURN-Relay** über den FAX-eigenen Server `turn.faxmessenger.com`. Dieser leitet nur DTLS-verschlüsselten Traffic weiter.

7.3 TURN-Server-Eigenschaften

Der TURN-Server läuft auf **coturn** mit time-limited HMAC-SHA1-Credentials (24 Stunden Gültigkeit). Er sieht ausschließlich verschlüsselten Datenverkehr — Inhalte, Identitäten und Metadaten der Konversation sind für ihn nicht zugänglich.

- UDP 3478/3479 — Standard
- TCP 3478 — Fallback bei UDP-Blockade
- TLS 5349/5350 — Verschleierung als HTTPS-Traffic

7.4 IP-Shield

Standardmäßig sehen die Peers ihre gegenseitigen IP-Adressen via ICE. Im **IP-Shield-Modus** wird die Verbindung erzwungenermaßen über TURN geleitet (auch wenn direktes P2P möglich wäre); damit sieht der jeweils andere Peer nur die TURN-Server-IP, nicht die echte IP.

08 · Identitäts- & Geräte-Sicherheit

8.1 Identitäts-Erzeugung

Eine FAX-Identität besteht aus drei Keypaaren plus einer öffentlich teilbaren ID (Hash der Public Keys):

```
ID = {
  ecdh_p384: { pub, priv },
  ecdh_p256: { pub, priv },
  ml_kem_1024: { pub, priv },
  ml_dsa_65: { pub, priv }
}

fax_id = SHA-256(
  ecdh_p384.pub || ml_kem.pub || ml_dsa.pub
)[0:16]  // 16 Bytes = 32 Hex-Zeichen
```

8.2 Speicherung

Die Identität wird im `localStorage` des Browsers persistiert. Optional kann der Nutzer drei zusätzliche Schutzschichten aktivieren:

SCHUTZ	MECHANISMUS	STÄRKE
PIN	PBKDF2-SHA-512, 100 000 Iter.	Brute-Force-resistent
Biometrie	WebAuthn (Touch ID, Face ID, Windows Hello)	Hardware-Secure-Element
Hardware-Bound	Identity wird mit WebAuthn-derived AES-Key gewrappt	Geräte-gebunden

8.3 Duress-Modus

Der Nutzer kann eine zweite, "Duress-PIN" einrichten. Wird diese eingegeben (statt der echten PIN), startet FAX in einem leeren Zustand: keine Kontakte, keine Nachrichten, keine Identität sichtbar. Für einen Beobachter sieht es aus, als wäre die App noch nie benutzt worden.

> ANWENDUNGSSZENARIO
>
> Bei Zwang an Grenzkontrollen oder ähnlichen Situationen, wo der Nutzer gezwungen wird, sein Telefon zu entsperren, kann er die Duress-PIN eingeben. Die App zeigt eine frische Oberfläche; die echten Daten

> bleiben unsichtbar und unzugänglich.

8.4 Notfall-Wipe

Ein Tipp auf das App-Logo mit definierter Sequenz löst einen **vollständigen Wipe** aus: Identitäts-Schlüssel werden überschrieben, Nachrichten-Speicher werden zurückgesetzt, IndexedDB wird gelöscht. Es gibt keinen Wiederherstellungs-Mechanismus.

09 · Traffic-Analyse-Schutz

Selbst bei perfekter Verschlüsselung lassen sich aus den *Größen* der übertragenen Pakete oft Rückschlüsse auf den Inhalt ziehen. FAX verschleiert dies durch konstantes Padding.

9.1 Tiered Constant-Length Padding

Alle Nachrichten werden auf eine der folgenden festen Bucket-Größen aufgefüllt:

```
PAD_BUCKETS = [256, 512, 1024, 2048, 4096, 8192, 16384]

function pad(plaintext):
  bucket = smallest_bucket >= (plaintext.length + 2)
  out = new bytes(bucket)
  out[0..2] = uint16_be(plaintext.length)
  out[2..2+plaintext.length] = plaintext
  out[2+plaintext.length..bucket] = csprng_bytes()  // kein NUL-Padding
  return out
```

9.2 Schutzwirkung

Ein Angreifer, der nur die **Größen** beobachten kann, sieht ausschließlich diese diskreten Stufen — niemals die echte Inhalts-Länge. Eine kurze Antwort ("OK") und eine lange Erklärung (256-Zeichen-Text) sind identisch lang (256 Byte). Erst bei längeren Inhalten unterscheiden sich die Buckets — und auch dann nur grob.

BANDBREITEN-OVERHEAD

Im Mittel etwa 30% Overhead. Bei sehr kurzen Nachrichten relativ hoch (256 Byte vs. ~20 Byte Klartext = ~92% Overhead), bei größeren Inhalten gering.

10 · Audit-Trail & Merkle-Chain

FAX führt eine **lokale Hash-Chain** aller Nachrichten. Jeder Eintrag enthält den Hash des vorherigen Eintrags:

```
block_n = SHA-256(
  block_{n-1}.hash ||
  timestamp ||
  event_type ||
  payload
)
```

Eine nachträgliche Manipulation einer Nachricht würde alle nachfolgenden Block-Hashes invalidieren — was beim nächsten Audit-Check sofort auffällt. Die Chain ist **nur lokal** sichtbar; sie wird nicht mit Peers synchronisiert oder zu einem Server geschickt. Sie dient ausschließlich der Selbst-Verifikation.

10.1 Verifikation

Der Nutzer kann jederzeit über das Audit-Menü die komplette Chain seit Erzeugung neu hashen und mit dem aktuellen Stand vergleichen. Bei einer Abweichung wird der entsprechende Block angezeigt.

11 · Bekannte Einschränkungen

Diese Einschränkungen sind bewusst transparent dokumentiert. Sie betreffen entweder grundlegende Limits browserbasierter Apps oder bewusste Designentscheidungen.

BROWSER-BASIERTE LIMITATIONEN

- **JavaScript-Speicher-Management:** Keys liegen kurzzeitig im JS-Speicher; Browser-Garbage-Collection ist nicht deterministisch. `_wipeBuffer()`-Funktion überschreibt Buffer mit Zufallsdaten, kann aber nicht garantieren, dass keine Kopien im Speicher verbleiben.
- **Browser-Zero-Days:** Ein kompromittierter Browser kompromittiert auch FAX. Empfehlung: Browser stets aktuell halten.
- **Side-Channels:** Timing- und Cache-Side-Channels im Browser sind schwer zu verhindern. Die verwendeten Bibliotheken (noble-post-quantum) implementieren Konstant-Zeit-Operationen wo möglich.

P2P-LIMITATIONEN

- **Beide Peers müssen gleichzeitig online sein** — es gibt keinen Server, der Nachrichten zwischenspeichert.
- **Offline-Queue** ist auf das Sender-Gerät beschränkt: Nachrichten an offline-Peers werden lokal gequeued und automatisch nachgesendet, sobald der Peer wieder online kommt. Funktioniert nur solange der Sender online bleibt.
- **Push-Notifications** außerhalb des geöffneten Browsers sind nicht möglich (siehe Web-Push für mögliche zukünftige Erweiterung).

GRUPPEN-LIMITATIONEN

- Mesh-Topologie: jeder Peer mit jedem direkt verbunden.
- Skaliert quadratisch (n·(n−1) Verbindungen). Praktische Obergrenze: **4 Teilnehmer**.

12 · Vergleich zu anderen Messengern

EIGENSCHAFT	FAX	SIGNAL	WHATSAPP	TELEGRAM
End-to-End-Verschlüsselung	✓	✓	✓	nur "Secret Chats"
Post-Quantum (FIPS 203)	**ML-KEM-1024**	**PQXDH (Kyber)**	—	—
Forward Secrecy	**Triple Ratchet**	**Double Ratchet**	**Double Ratchet**	teilw.
Zentraler Server	**Nein (P2P)**	**Ja**	**Ja**	**Ja**
Account-Pflicht	**Nein**	**Telefonnr.**	**Telefonnr.**	**Telefonnr.**
Traffic-Padding	**Tiered**	**teilw.**	—	—
Duress-Modus	✓	—	—	—
Hardware-bound Identity	**WebAuthn**	—	—	—
Metadaten beim Anbieter	**keine**	minimal	**umfangreich**	**umfangreich**
Installation nötig	**Nein (Browser)**	**Ja**	**Ja**	**Ja**

13 · Glossar

BEGRIFF	ERKLÄRUNG
AES-GCM-256	Symmetrischer Algorithmus zur authentifizierten Verschlüsselung. NSA-zertifiziert für Top-Secret-Daten.
ECDH	Elliptic Curve Diffie-Hellman. Schlüsselaustausch über elliptische Kurven.
P-384, P-256	Standardisierte NIST-Kurven für ECDH/ECDSA.
ML-KEM	Module-Lattice-based Key Encapsulation Mechanism. NIST FIPS 203 (Post-Quantum).
ML-DSA	Module-Lattice-based Digital Signature Algorithm. NIST FIPS 204.
HKDF	HMAC-based Key Derivation Function (RFC 5869). Ableitung mehrerer Keys aus einem Master.
PBKDF2	Password-Based Key Derivation Function 2 — macht Passwort-Hashing langsam.
WebRTC	Web Real-Time Communication. Browser-natives P2P-Protokoll.
DTLS-SRTP	Datagram Transport Layer Security + Secure Real-time Transport Protocol. WebRTC's Verschlüsselungs-Schicht.
NAT-Traversal	Techniken, um trotz Network Address Translation direkte Verbindungen zu erlauben.
STUN/TURN	Hilfsserver für NAT-Traversal. TURN leitet bei Bedarf verschlüsselten Traffic weiter.
Double Ratchet	Verfahren, das pro Nachricht einen neuen Schlüssel ableitet — Forward Secrecy.
Triple Ratchet	Double Ratchet erweitert um periodische Post-Quantum-Key-Rotation.
WebAuthn	Web Authentication API — nutzt Hardware-Secure-Elements (Touch ID etc.).
Merkle Chain	Hash-Chain, bei der jeder Block den Hash des vorherigen enthält. Manipulation ist detektierbar.

BEGRIFF	ERKLÄRUNG
Forward Secrecy	Eigenschaft: kompromittierter aktueller Key gibt vergangene Nachrichten nicht preis.
Post-Compromise Security	Eigenschaft: nach Kompromittierung wird Verbindung wieder sicher (durch Ratchet).

FAX MESSENGER · POST-QUANTUM SECURE P2P

faxmessenger.app

© 2026 · Alle Rechte vorbehalten