

HOTRFC · IETF · 5 MIN

# SAFE

Sealed, Algorithm-Flexible Envelope

Encrypted container for large files,  
multi-recipient keying, multi-factor credentials,  
random-access authenticated encryption

thesafe.dev

ENCRYPT DECRYPT KEYCHAIN A

INPUT

Hello, SAFE world!

RECIPIENTS

LOCK A

P-256 argon2id

LOCK B

ML-KEM-768 Passkey argon2id

+ Add Recipient

SETTINGS

AEAD AEGIS-256 Hash TurboSHAKE256

KEM ML-KEM-768 Encoding Armored

**ENCRYPT**

1 / 8

thesafe.dev draft-sullivan-safe-00

# Filling the gaps

What no existing format does natively

CAPABILITY	SAFE	OPENPGP	JWE	CMS
Large-file framing	✓	P	P	P
Streaming decrypt	✓	✓	P	✓
Random-access reads & writes (raAE)	✓	—	—	—
Multi-factor recipient unlock	✓	—	P	P
Algorithm agility	✓	✓	✓	✓

• Native support • Achievable / partial • No support

# One abstraction: LOCK

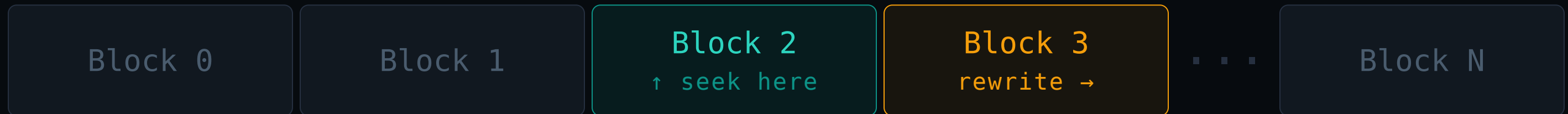
Each LOCK wraps the CEK independently. Any one is sufficient.



- Any one LOCK is sufficient to decrypt; each wraps the CEK independently
- Key privacy: omit identifiers to force trial decryption
- STEP types: HPKE Export , PASS , WEBAUTHN-PRF , PRIVACY PASS ; extensible via IANA registry
- PQ hybrid: add `HPKE·ML-KEM-768` alongside a classical DHKEM HPKE step: falls out for free, no format changes

# raAE: random-access authenticated encryption

Per-block AEAD extending STREAM: read or rewrite any block without decrypting or re-keying the rest



## RANDOM ACCESS

Read **any block** without decrypting the rest

Block-granular decryption. Unique IV per block. Snapshot accumulator ensures integrity across seeks.

## WRITE IN PLACE

Rewrite **one block** without re-keying the file

Selective re-encryption of modified blocks only. Streaming decryption still supported. Formally analyzed.

raAE from [eprint.iacr.org/2025/2275](https://eprint.iacr.org/2025/2275)

# Data section: three wire formats

Same ciphertext, different framing: choose by context

## ARMORED

### Text-safe encoding

```
-----BEGIN SAFE DATA-----
0QfC0etpCqiyFAGsDXqsIh5j0xeZpmw0uGo+/0Cv
D31GKD94PKg0+kgLfLx+BQ2JK8MqQz9ch31mwh2M
WAB3Nw00EgZX73Uw0gr8ZL0XnHwDV2MPUpm187Si
03Zzc+byLKqx/mkbp3yvvnJAzAL4HJvH3zRzwmBHE
x3ZMV5MHHLQwbXbmmDr9zxsM+xWU5CT7CGi+p2JL
FxrL/xPZyoPd24KEb17nqjqMCHpIkBFIbY7kf9dL
hwJbEmK0uysbXmQ=
-----END SAFE DATA-----
```

PEM fences + Base64 body. Embeds in email, config files, CLI pipelines, and agents. No binary-safe transport required.

## BINARY / LINEAR

### Streaming

salt	32B
commitment	32B
accumulator	32B
nonce <sub>0</sub>	12B
ciphertext <sub>0</sub>	64 KB
tag <sub>0</sub>	16B
nonce <sub>1</sub>	12B
ciphertext <sub>1</sub>	64 KB
tag <sub>1</sub>	16B
...	...

Nonce (12B) + ciphertext + tag (16B) per block, sequential. Unique nonce per block. Configurable block size (64 KB default).

## BINARY / ALIGNED

### Minimal editing profile

commitment	32B
block count N	u32
nonce <sub>0</sub> + tag <sub>0</sub>	28B
nonce <sub>1</sub> + tag <sub>1</sub>	28B
... all N metadata entries ...	
-- padding to next 16 KB boundary --	
ciphertext <sub>0</sub>	16 KB
ciphertext <sub>1</sub>	16 KB
...	...

All nonces and tags front-loaded; header padded to a 16 KB boundary. Seek directly to any block at offset  $i \times 16$  KB. Minimal editing profile: rewrite 16 KB without touching anything else.

Authenticity

Snapshot integrity

Key commitment

Multi-recipient isolation

Algorithm agility

Forward secrecy (ephemeral KEM)

Streaming + random-access decrypt

# Practical system

TWO PROFILES

## FIPS

PBKDF2 P-256 AES-256-GCM HKDF-SHA2

Random nonce per block satisfies FIPS nonce-uniqueness requirements.

## Modern high-performance

Argon2id ML-KEM-768 TurboSHAKE  
AEGIS-256

PQ hybrid via the LOCK step model. All algorithms from current IETF standards.

CROSS-PROTOCOL IMPLEMENTATIONS

- Go
- Rust
- Python
- TypeScript

Portable across text environments, CLI tools, and agents. Able to do line rate.

Read the draft. Try [thesafe.dev](https://thesafe.dev). Bring feedback to the author(s). [draft-sullivan-safe-00](https://github.com/sullivan-safe/draft-sullivan-safe-00)

## WHAT WAS BUILT

- **Flexible encrypted container**

Multi-recipient, multi-factor, algorithm-agile. LOCK step model composes PQ, WebAuthn, and Privacy Pass for free.

- **raAE: formally analyzed (ongoing)**

Generalizes STREAM to support random access and in-place writes. Key commitment. Snapshot integrity. Ristenpart et al.

- **Running code, two profiles**

Go, Rust, Python, TypeScript. FIPS and modern configurations. OS-page and disk-sector aligned. Line-rate capable.

# Working on something relevant?

Reach out.

DRAFT

[datatracker.ietf.org/doc/draft-sullivan-safe/](https://datatracker.ietf.org/doc/draft-sullivan-safe/)

TRY  
IT

[thesafe.dev](https://thesafe.dev)

DISCUSS

[draft-sullivan-safe@ietf.org](mailto:draft-sullivan-safe@ietf.org)

# References

PROJECT SITE

<https://thesafe.dev/>

CURRENT DRAFT: DRAFT-SULLIVAN-SAFE-00 · MARCH 2026

[datatracker.ietf.org/doc/draft-sullivan-safe/](https://datatracker.ietf.org/doc/draft-sullivan-safe/)

SECURITY ANALYSIS (RAAE)

<https://eprint.iacr.org/2025/2275>