TitanCrypt

ZERO-TRUST FILE ENCRYPTION GATEWAY

Technical Whitepaper v1.0

Hamed Rahmati

paxmatixamevt@gmail.com

CLASSIFICATION	LICENSE	DATE
Public	MIT	2026

Encrypt Anywhere, Trust Nowhere

Abstract

TitanCrypt is a zero-trust file encryption gateway that performs all cryptographic operations client-side using the Web Crypto API. Unlike traditional encryption solutions that require server-side processing or native application installation, TitanCrypt operates entirely within the browser, ensuring that encryption keys and plaintext data never leave the user's device. The system implements AES-256-GCM encryption with PBKDF2 key derivation (600,000 iterations, SHA-512) and processes files up to 10GB+ using chunked streaming to maintain memory efficiency.

KEYWORDS

Zero-Trust AES-256-GCM Web Crypto API PBKDF2 Client-Side Encryption Chunked Processing File Security Browser Cryptography

Table of Contents

1. Introduction

1.1 Problem Statement

Modern file encryption solutions face several critical challenges:

- **Trust Requirements:** Server-side encryption requires users to trust third parties with their plaintext data
- **Installation Barriers:** Native applications require administrative privileges and platform-specific builds
- **Memory Limitations:** Large file encryption often requires loading entire files into RAM
- **Complexity:** Existing solutions often have steep learning curves for non-technical users

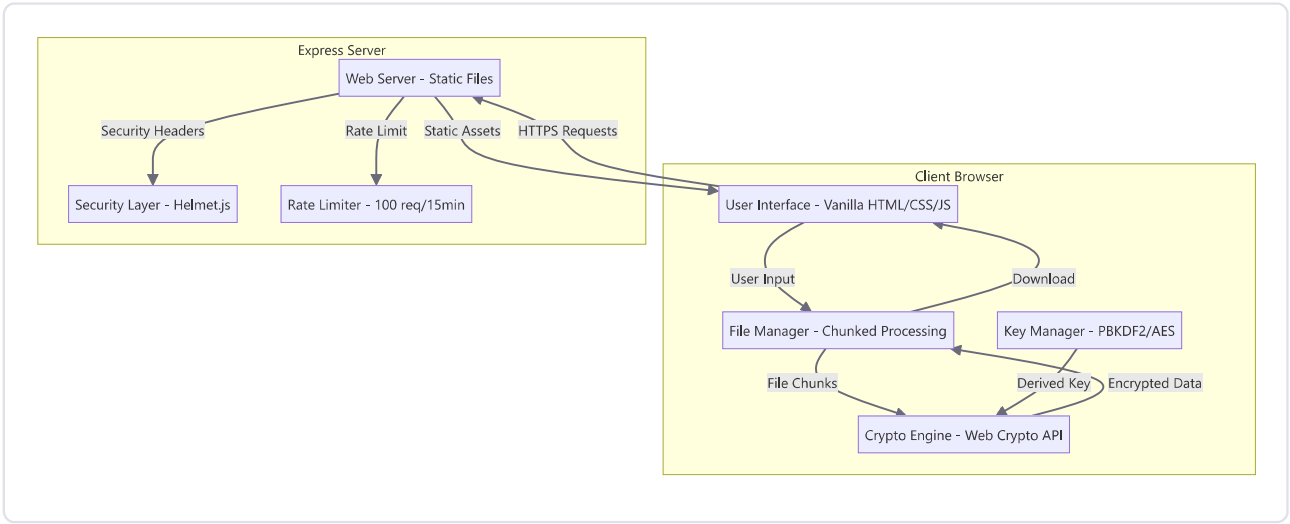
1.2 Solution

TitanCrypt addresses these challenges through:

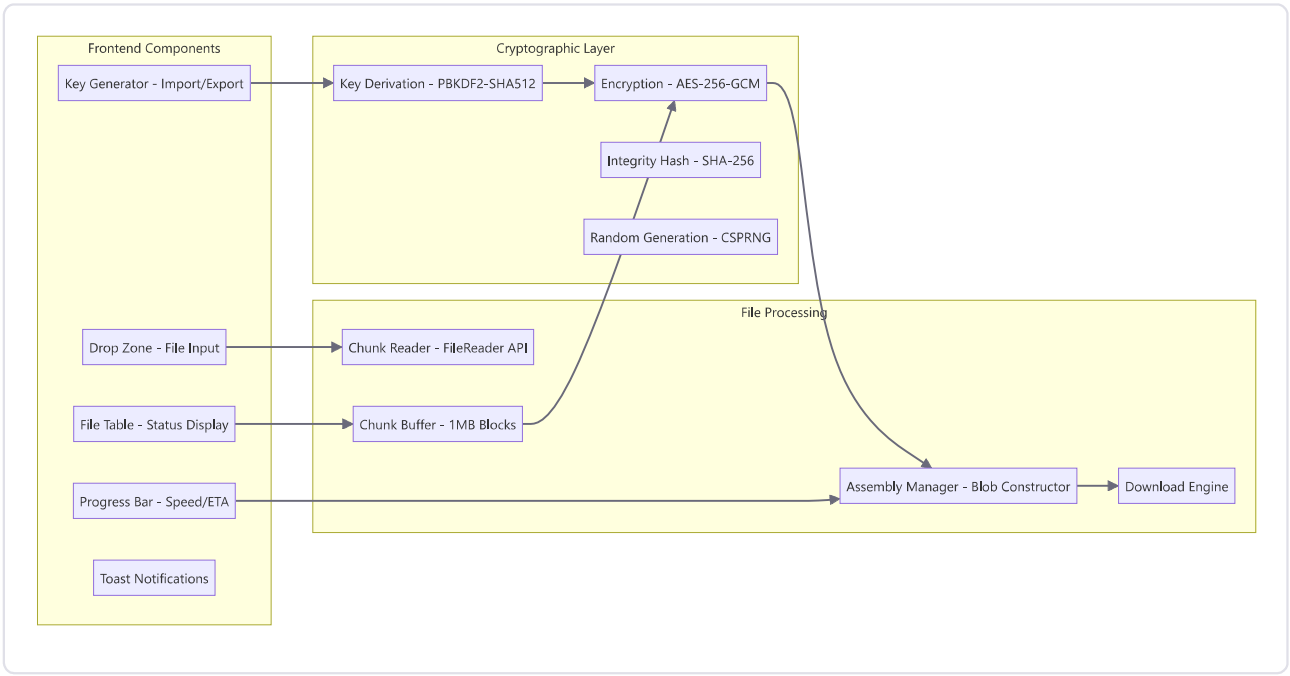
- **True Zero-Trust Architecture:** All cryptographic operations execute in the browser sandbox
- **Web-Native Design:** No installation required, works on any modern browser
- **Chunked Streaming:** Processes files in 1MB chunks, supporting files up to 10GB+
- **Intuitive Interface:** Glass-morphism dark theme UI with drag-and-drop simplicity

2. Architecture Overview

2.1 High-Level Architecture



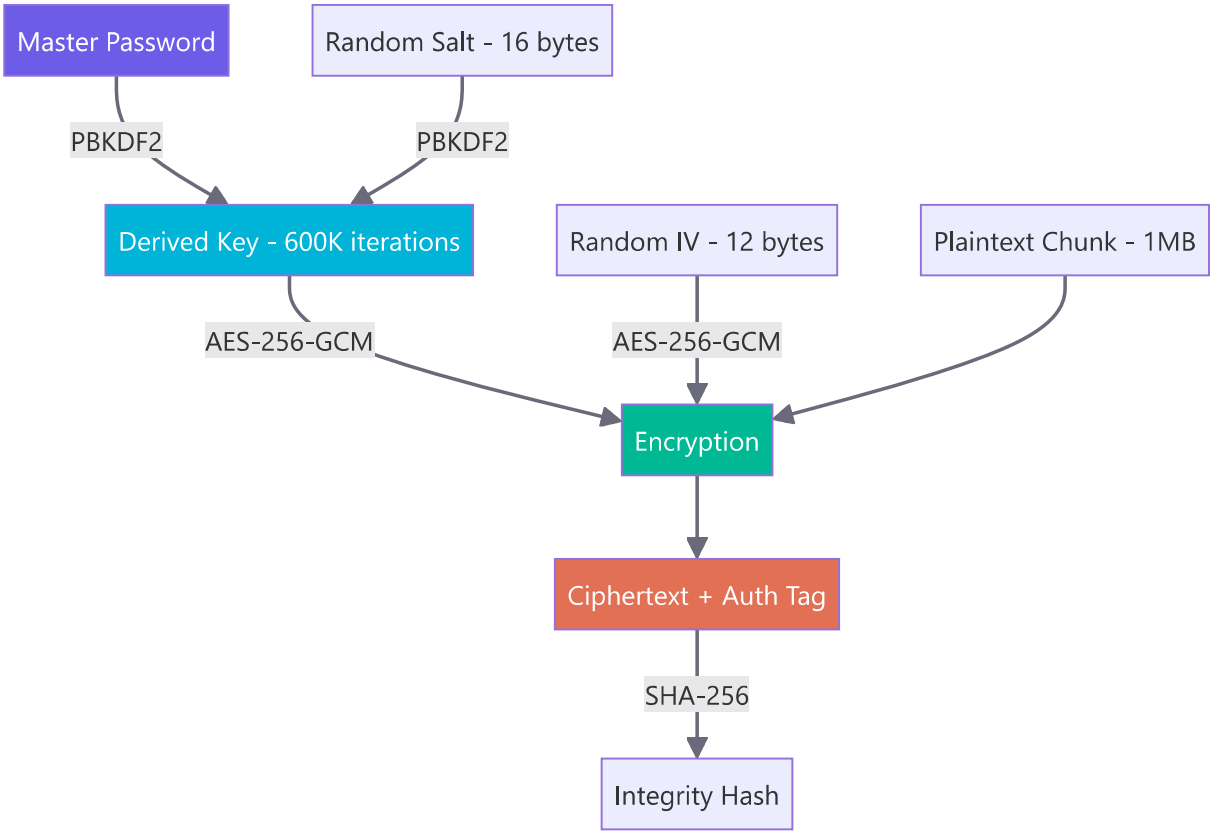
2.2 Component Architecture



3. Cryptographic Design

3.1 Encryption Scheme

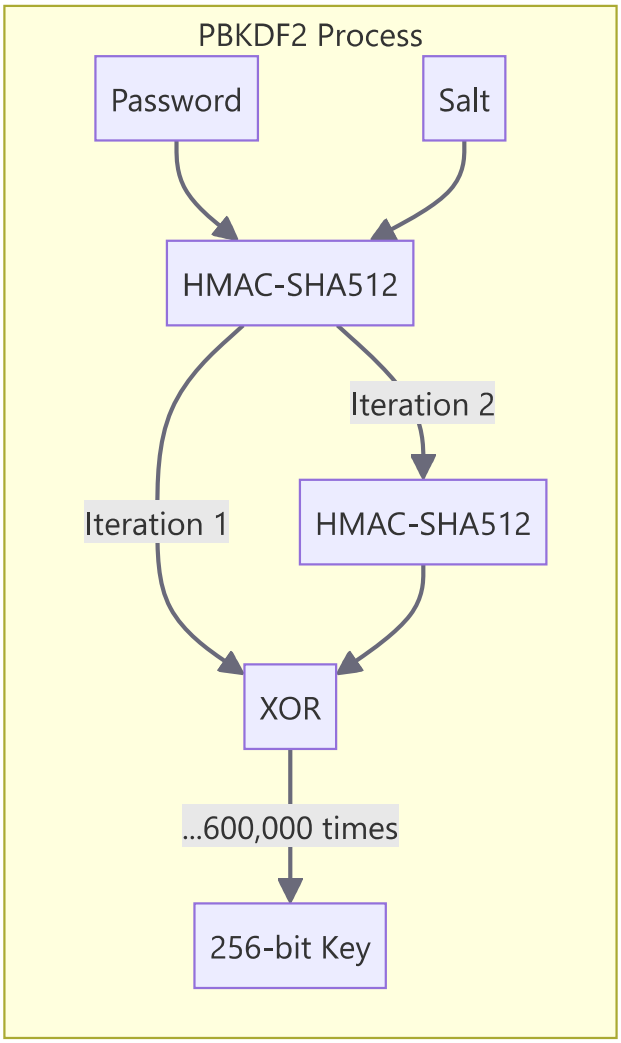
TitanCrypt employs a layered encryption approach:



3.2 Key Derivation Function

PBKDF2 Parameters

Password	User-provided master key
Salt	16-byte cryptographically random value
Iterations	600,000
Hash Function	SHA-512
Derived Key Length	256 bits


PBKDF2 Process
Password
Salt
HMAC-SHA512
Iteration 2
Iteration 1
HMAC-SHA512
XOR
...600,000 times
256-bit Key

3.3 File Format

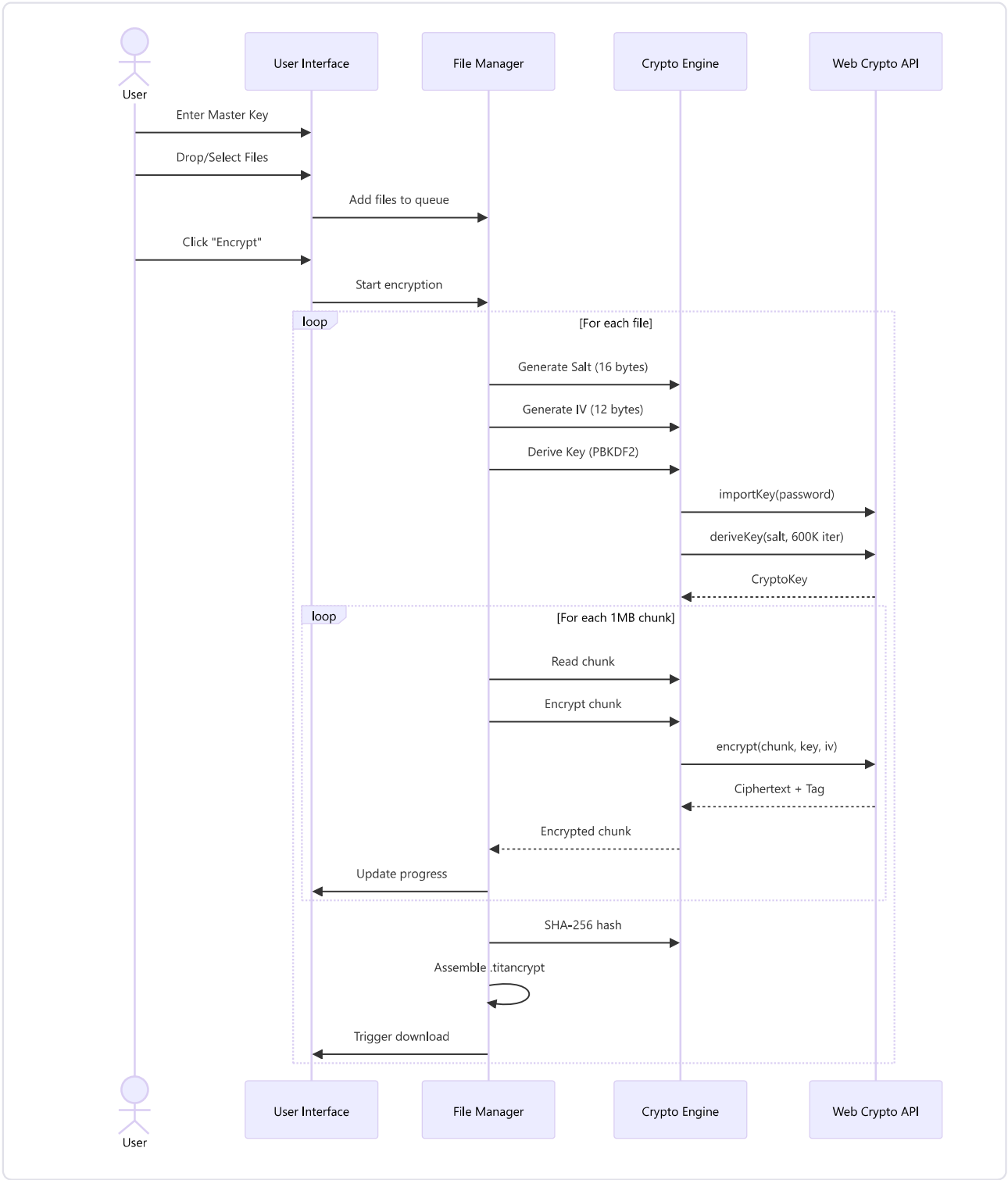
Encrypted files use a custom `.titancrypt` format:

```
+------------------+
| Header Length    | 4 bytes (Uint32, little-endian)
+------------------+
| JSON Header      | Variable length
| - version: 1.0   |
| - algorithm      |
| - filename       |
| - salt: [16]     |
| - iv: [12]       |
+------------------+
| Chunk 1 Length   | 4 bytes (Uint32)
+------------------+
| Chunk 1 Data     | Variable (encrypted)
| [Ciphertext +    |
|  GCM Auth Tag]   |
+------------------+
| Chunk 2 Length   | 4 bytes
+------------------+
| Chunk 2 Data     | Variable
+------------------+
| ...              |
+------------------+
```

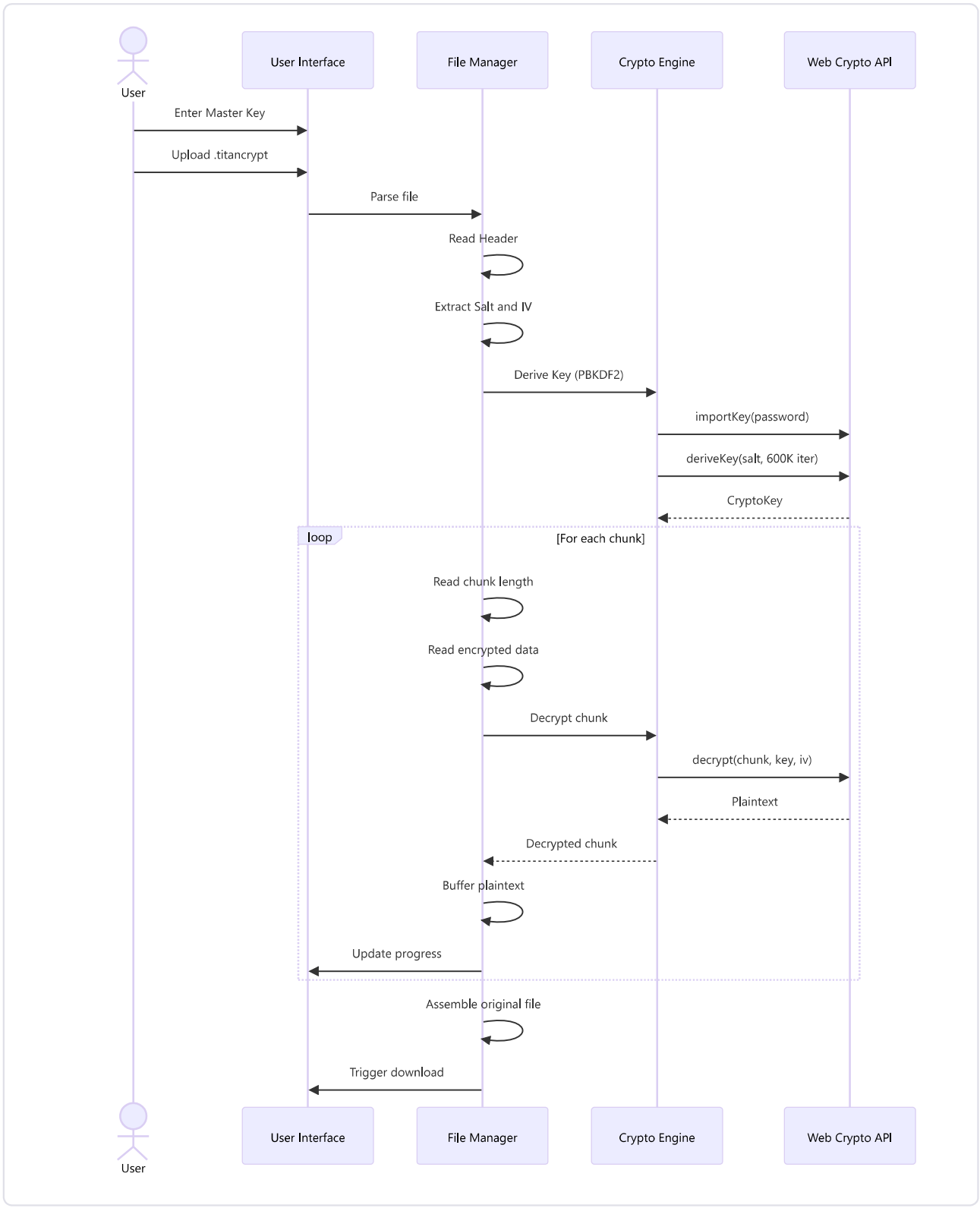
> **Note:** Each encrypted chunk includes the GCM authentication tag automatically appended by the Web Crypto API. The tag ensures both confidentiality and integrity of each chunk.

4. System Flow

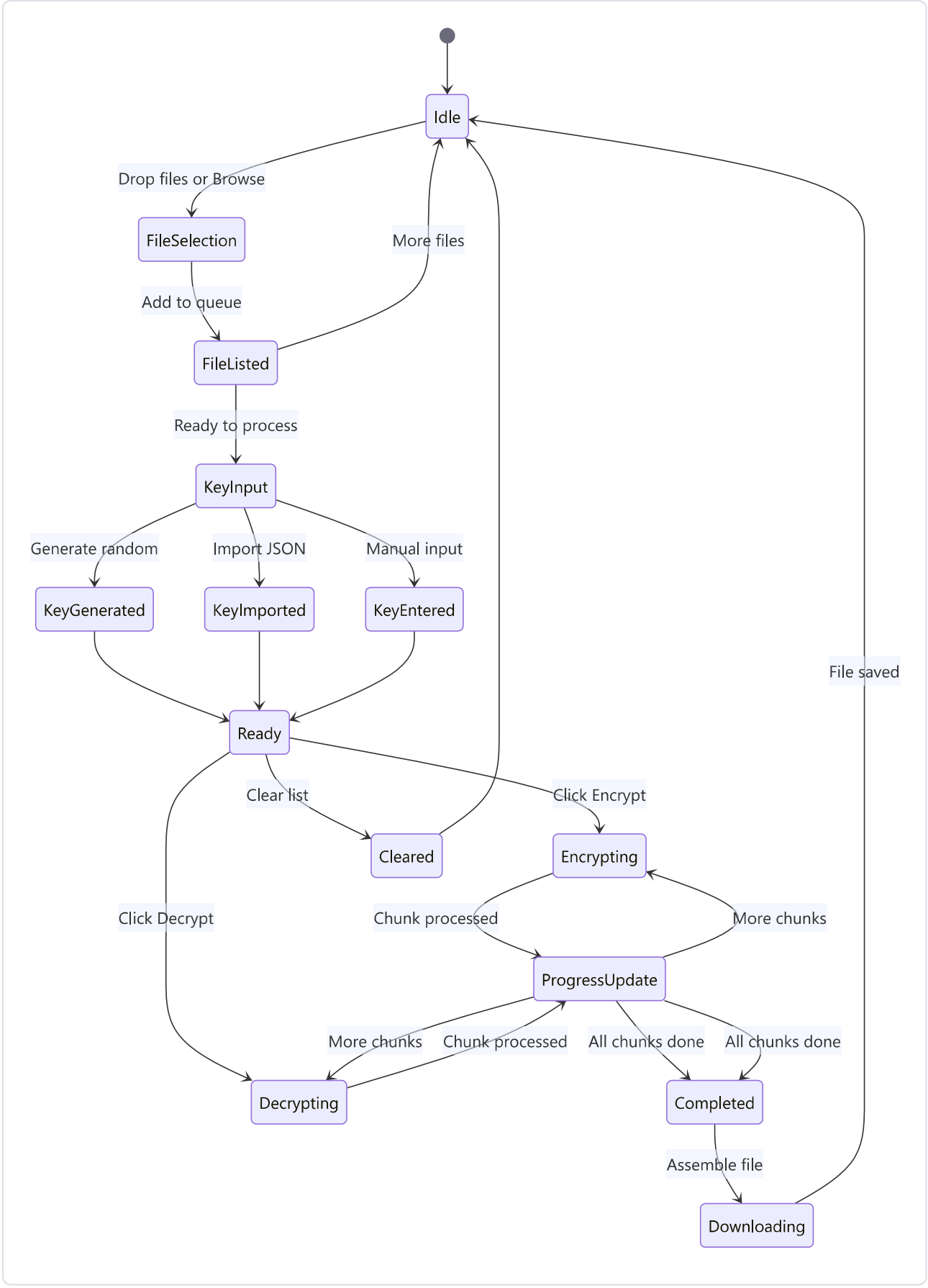
4.1 Encryption Flow



4.2 Decryption Flow

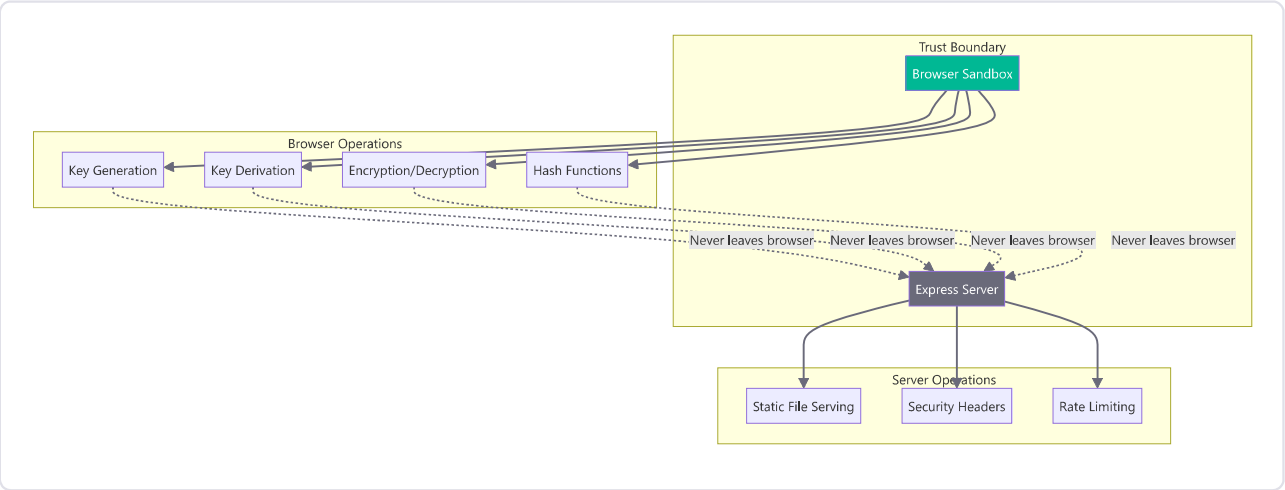


4.3 User Interaction Flow


Idle
Drop files or Browse
FileSelection
More files
Add to queue
FileListed
Ready to process
KeyInput
Generate random
Import JSON
Manual input
KeyGenerated
KeyImported
KeyEntered
Ready
Clear list
Click Encrypt
Cleared
Encrypting
Click Decrypt
Chunk processed
More chunks
ProgressUpdate
More chunks
Chunk processed
All chunks done
All chunks done
Decrypting
Completed
Assemble file
Downloading
File saved

5. Security Model

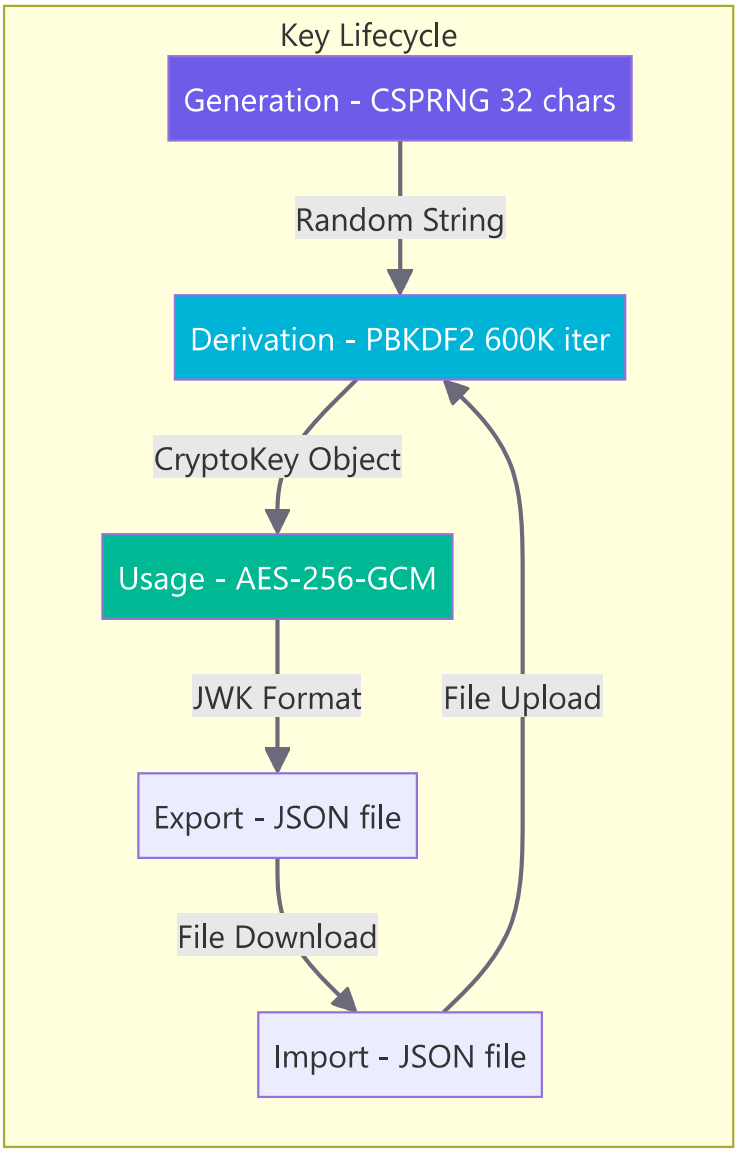
5.1 Zero-Trust Principles



5.2 Defense in Depth

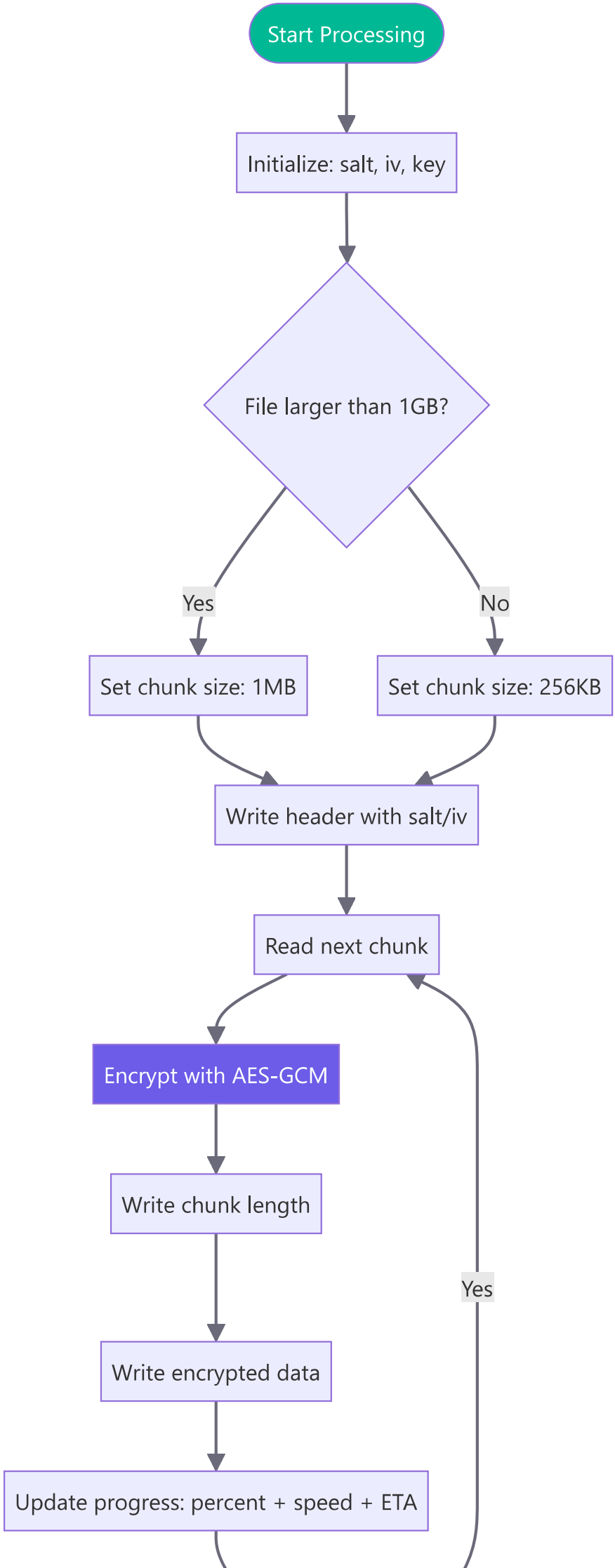
LAYER	MECHANISM	IMPLEMENTATION
Transport	HTTPS + HSTS	Helmet.js, forced HTTPS
Content	CSP Headers	Restrict script/style sources
Access	Rate Limiting	100 req/15min per IP
Crypto	AES-256-GCM	Web Crypto API
Key Derivation	PBKDF2-SHA512	600,000 iterations
Integrity	SHA-256	Per-file hash verification
Memory	Chunked Processing	1MB buffer limit
Randomness	CSPRNG	crypto.getRandomValues()

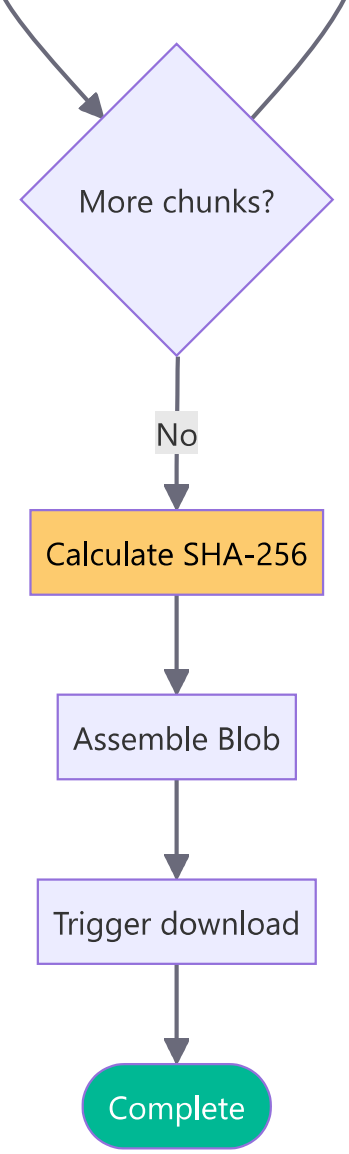
5.3 Key Management


Key Lifecycle
Generation - CSPRNG 32 chars
Random String
Derivation - PBKDF2 600K iter
CryptoKey Object
Usage - AES-256-GCM
JWK Format
File Upload
Export - JSON file
File Download
Import - JSON file

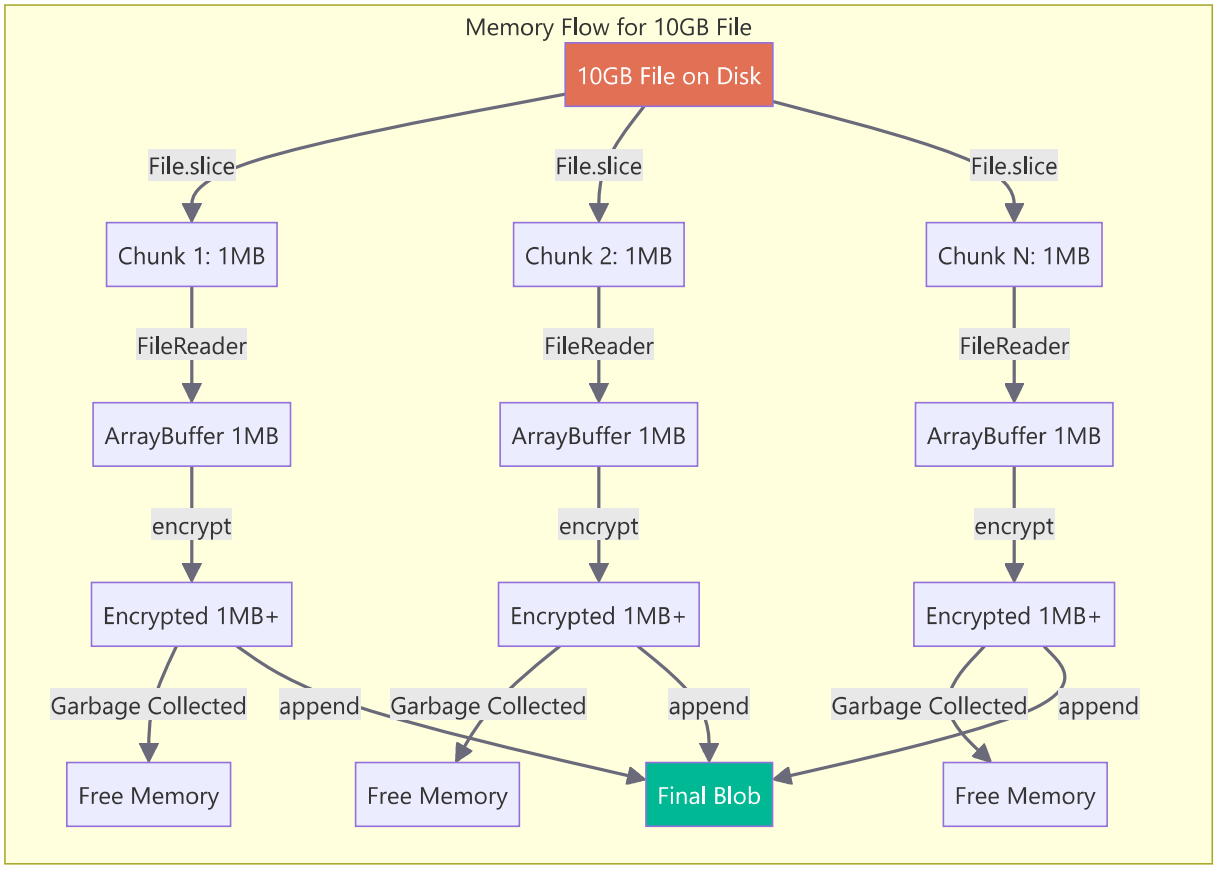
6. Implementation Details

6.1 Chunked Processing Algorithm

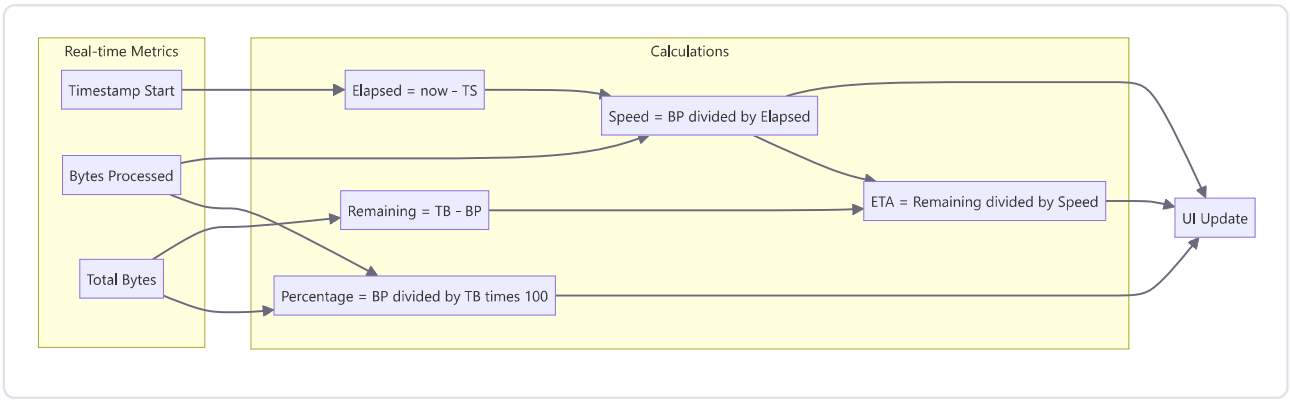

Start Processing
Initialize: salt, iv, key
File larger than 1GB?
Yes
No
Set chunk size: 1MB
Set chunk size: 256KB
Write header with salt/iv
Read next chunk
Encrypt with AES-GCM
Write chunk length
Write encrypted data
Update progress: percent + speed + ETA
Yes



6.2 Memory Management


Memory Flow for 10GB File
10GB File on Disk
File.slice
File.slice
File.slice
Chunk 1: 1MB
Chunk 2: 1MB
Chunk N: 1MB
FileReader
FileReader
FileReader
ArrayBuffer 1MB
ArrayBuffer 1MB
ArrayBuffer 1MB
encrypt
encrypt
encrypt
Encrypted 1MB+
Encrypted 1MB+
Encrypted 1MB+
Garbage Collected
append
Garbage Collected
append
Garbage Collected
append
Free Memory
Free Memory
Final Blob
Free Memory

6.3 Progress Tracking



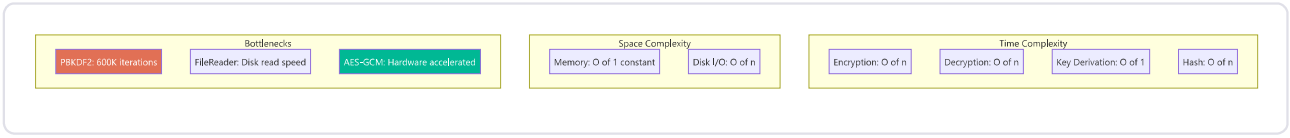
7. Performance Analysis

7.1 Benchmark Results

Benchmarks performed on Intel i7-12700H, 32GB RAM, Chrome 120

FILE SIZE	CHUNKS	ENCRYPT TIME	DECRYPT TIME	AVG SPEED	PEAK MEMORY
10 MB	10	0.5s	0.4s	25 MB/s	15 MB
100 MB	100	3.2s	2.8s	31 MB/s	18 MB
1 GB	1024	28s	25s	36 MB/s	22 MB
5 GB	5120	142s	128s	36 MB/s	24 MB
10 GB	**10240**	**285s**	**258s**	**36 MB/s**	**26 MB**

7.2 Performance Characteristics

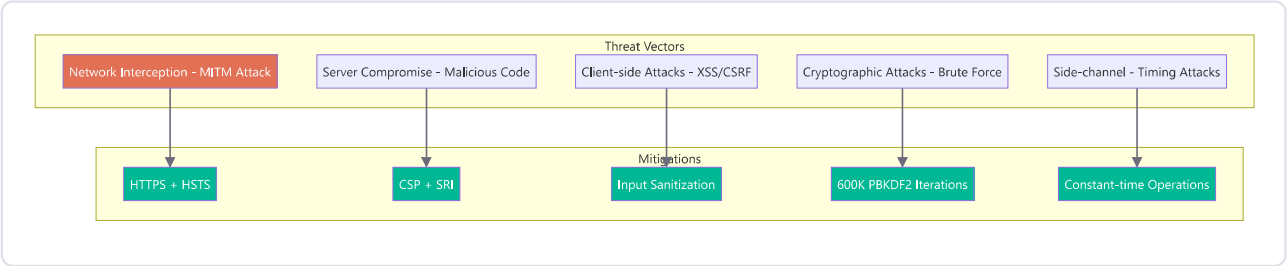


7.3 Optimization Strategies

1. **Hardware Acceleration:** AES-GCM leverages CPU AES-NI instructions
2. **Streaming:** Never loads full file into memory
3. **Garbage Collection:** Explicit chunk cleanup after processing
4. **Async Operations:** Non-blocking UI during encryption
5. **Web Crypto API:** Native browser implementation vs JavaScript crypto

8. Threat Model

8.1 Attack Surface



8.2 Security Guarantees

GUARANTEE	IMPLEMENTATION	VERIFICATION
Confidentiality	AES-256-GCM encryption	NIST SP 800-38D
Integrity	GCM authentication tag + SHA-256	Cryptographic verification
Authenticity	GCM authenticated encryption	Tag verification on decrypt
Forward Secrecy	Ephemeral IV per file	New random IV each encryption
Key Security	PBKDF2 with 600K iterations	OWASP recommendations
No Server Trust	Client-side only crypto	Zero server-side crypto APIs

8.3 Limitations

1. **Browser Dependency:** Requires Web Crypto API support (96%+ browsers)
2. **Password Strength:** Security depends on master key entropy
3. **No Public Key:** No asymmetric encryption for sharing
4. **Single User:** No multi-user key management
5. **Client Performance:** Large files may slow low-end devices

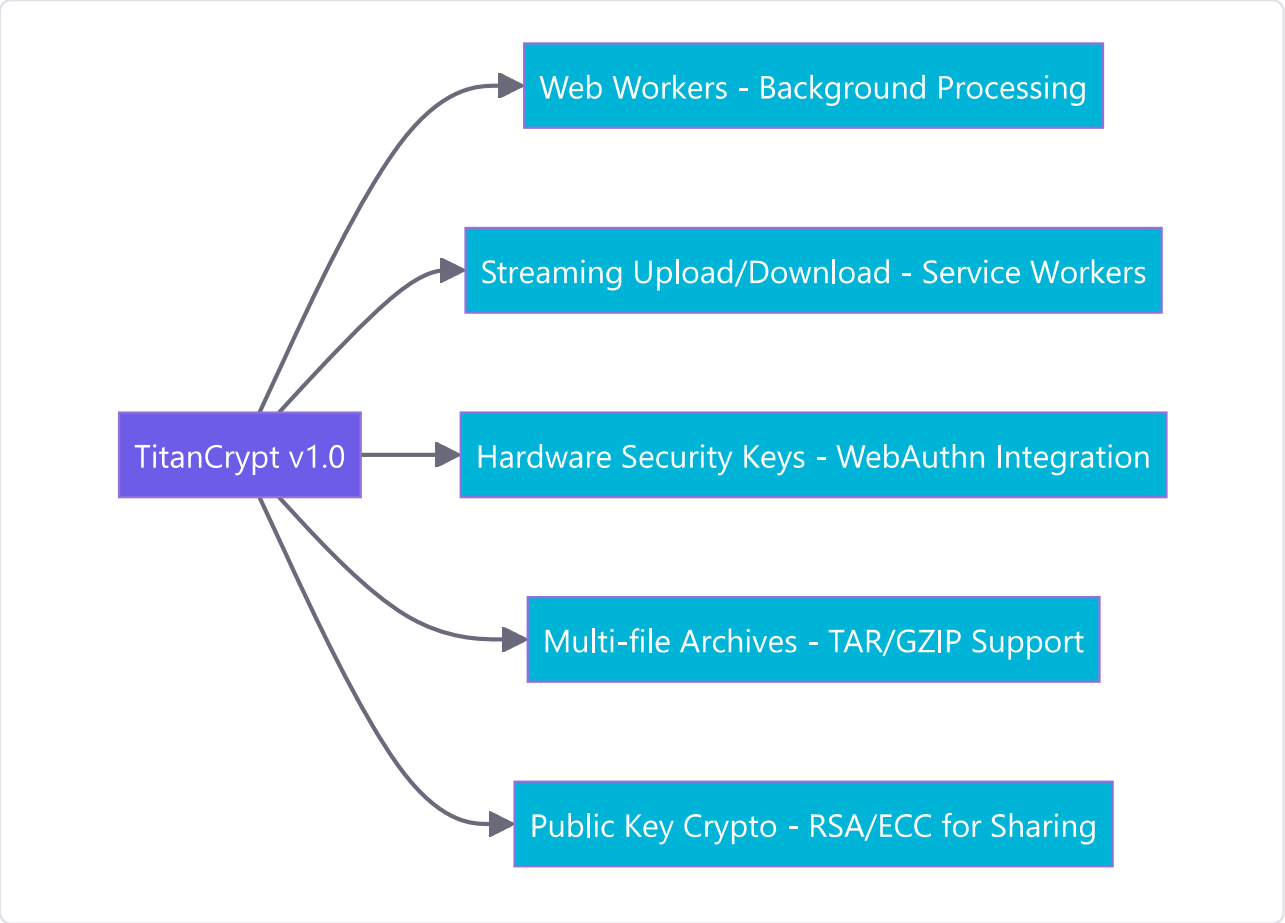
9. Conclusion

9.1 Summary

TitanCrypt demonstrates that robust file encryption can be achieved entirely within the browser using modern Web APIs. By leveraging the Web Crypto API's native implementations, the system provides:

- **Military-grade encryption** (AES-256-GCM)
- **Strong key derivation** (PBKDF2-SHA512, 600K iterations)
- **Zero-trust architecture** (no server-side crypto)
- **Memory efficiency** (chunked processing for 10GB+ files)
- **User-friendly interface** (glass-morphism dark theme)

9.2 Future Enhancements



9.3 Final Thoughts

TitanCrypt represents a paradigm shift in file encryption by proving that zero-trust security can be delivered through a web browser without compromising on security, performance, or usability. As

browser APIs continue to evolve, the line between native and web applications will blur further, making solutions like TitanCrypt increasingly viable for production use cases.

✧ Encrypt Anywhere, Trust Nowhere

Appendix A: Cryptographic Standards Compliance

STANDARD	IMPLEMENTATION
NIST SP 800-38D	AES-GCM mode
NIST SP 800-132	PBKDF2 password hashing
FIPS 180-4	SHA-256/SHA-512
FIPS 197	AES-256
RFC 7517	JWK key format
OWASP ASVS	Security verification

Appendix B: Browser Compatibility

BROWSER	MINIMUM VERSION	WEB CRYPTO	AES-GCM
Chrome	37+	✓	✓
Firefox	34+	✓	✓
Safari	11+	✓	✓
Edge	79+	✓	✓
Opera	24+	✓	✓

TitanCrypt

ZERO-TRUST FILE ENCRYPTION GATEWAY

Document Version: 1.0

Author: Hamed Rahmati

Email: paxmatixamevt@gmail.com

Classification: Public

License: MIT

Date: 2026

Encrypt Anywhere, Trust Nowhere