Safari Limitations for Client-Side Cryptography (with libsodium)

1. Overview Modern client-side encryption requires stable WebAssembly (WASM), predictable memory behavior, strong entropy sources, and support for memory■hard algorithms such as Argon2id. Safari (both macOS and iOS) exhibits limitations that impact reliability, performance, and safety when using libsodium for file encryption in the browser.

2. WASM Threading and SharedArrayBuffer Restrictions Safari limits or disables SharedArrayBuffer unless strict cross■origin isolation headers are present. Libsodium's optimized implementations (especially Argon2id) require SharedArrayBuffer for safe and efficient parallel processing.

Sources: - MDN Web Docs: SharedArrayBuffer – crossOriginIsolated requirements. - WebKit Blog: New WebKit Features in Safari 15.2 – SharedArrayBuffer re■enabled with COOP/COEP. - WebAssembly.org Features Table – browser support differences.

3. Memory Usage and Stability Issues Safari has historically shown unpredictable behavior when WASM modules request large or growing memory regions. This affects file encryption where large ArrayBuffers and memory■hard KDFs are necessary.

Sources: - GitHub (emscripten-core): “Out of memory error in Safari with WASM shared memory.”

4. JavaScript and WASM Execution Throttling Safari aggressively throttles JavaScript and WASM execution during: - heavy CPU usage, - large memory consumption, - background tab execution.

This may interrupt encryption or corrupt output files during long-running operations.

5. WebCrypto Limitations and Legacy Algorithms WebCrypto is not designed for high-assurance cryptographic applications and supports legacy or less secure primitives. Safari’s implementation has inconsistent behavior for key generation and buffer handling, making it unsuitable as a fallback for libsodium.

Sources: - MDN Web Crypto API – warnings about misuse and limitations.

6. Entropy Source Constraints Safari may deliver reduced entropy in private mode or under restricted conditions. libsodium depends on a strong and consistent CSPRNG, and degradation affects key generation and nonce creation.

7. Blob and ArrayBuffer Inconsistencies Safari demonstrates low-level inconsistencies when reading large blobs into ArrayBuffers. File corruption risks arise when binary data does not map reliably into WASM memory.

8. Impact on Client-Side File Encryption Due to these limitations, Safari is not recommended for: - high■memory KDFs (Argon2id), - large file encryption, - streaming or chunked encryption processes, - long-running WASM cryptographic operations.

Many encryption-focused services explicitly warn users or disable advanced features in Safari.

9. Conclusion For client-side encryption tools using libsodium, Safari presents technical risks that do not affect Chromium■based or Firefox browsers. It is advisable to: - warn Safari users, - provide degraded fallback modes, - recommend Firefox/Chromium for full functionality.