MCES Feature Overview

Luke Miller

September 17, 2025

MCES Feature Overview

- **Cantor-Immune Stream Cipher (MCES):**
  - Adaptive 2D internal state; state size grows with Unicode password length.
  - Cantor-immune entropy placement: resists diagonal inversion and entropy resonance attacks by design.
  - Unicode-aware keying: accepts any language, emoji, and symbol—no entropy loss to encoding.
  - Full AEAD mode: all outputs authenticated via BLAKE3 keyed MAC.
  - Parallelizable: multi-threaded throughput up to 1.8 GB/s on commodity hardware.
  - Cross-platform: C and Rust implementations; supports ARM, x86, RISC-V, MIPS, PowerPC, and WebAssembly.
- **Security and Randomness:**
  - Passes all NIST SP800-22, Dieharder, PractRand, and BigCrush tests with no anomalies.
  - Strict Avalanche Criterion: single-bit flips in key or plaintext yield $\approx 46.8\%$ output bit changes.
  - Hamming weight and entropy remain balanced at all positions; output is statistically indistinguishable from random.
  - Resists neural cryptanalysis: only 8% success under avalanche-guided deep learning (versus 100% on AES, SM4, etc).
- **Password and Key Management:**
  - Native support for USB "sigilbook" tokens: passwords stored and retrieved securely via encrypted hardware device.
  - CLI and GUI utilities for encrypting, decrypting, verifying, and managing vaults.
  - Strong password policy: 30–100+ Unicode codepoints enforced.
  - Batch encryption/decryption modes with atomic, USB-only password export.
- **Performance and Edge/IOT Integration:**
  - Real-world performance: 151 MB/s on Raspberry Pi 3B+, 951 MB/s on Google Pixel 7.
  - Minimal memory overhead; instant key setup for all password lengths.

  - Live cam/mic encryption: supports real-time audio/video vaulting for security and privacy.

- **Usability and Developer Features:**
  - Modern desktop GUI (Rust, egui): intuitive file manager, batch tools, and USB key integration.
  - CLI dispatcher with subcommands for encryption, decryption, verification, and benchmarking.
  - Structured, human-readable file format: versioned headers, salt, timestamp, nonce, and HMAC tag per vault.
  - Modular, extensible: ready for new frontends and hardware integrations.
  - Auto decrypt and re-encrypt on double click of a vault file if sigilbook USB is inserted.