

CryptoToolkit™ — Technical Document - Presentation

 **ENGLISH**

CryptoToolkit™ — Post-Quantum Cybersecurity Software

I am pleased to announce the upcoming release of our post-quantum, anti-HNDL and anti-Q-Day cybersecurity software: **CryptoToolkit™**.

CryptoToolkit™, developed by ARANEA SOFT & AI SYSTEMS (araneasoft.fr), brings together today's most effective cryptographic defense weapons. It directly targets the biggest vulnerabilities exploited by hackers and autonomous AI systems: network interconnection, weak passwords, and long-term espionage. Below are its stated technical features — on paper, this software is "theoretically" a perfect answer to next-generation threats.

1. The Ultimate Shield Against HNDL: Post-Quantum Cryptography and Crypto-Agility

- **The HNDL threat** (Harvest Now, Decrypt Later): Today, hackers and governments intercept and store vast amounts of corporate data, even encrypted. They are simply waiting for quantum computers (a few years away) to break current algorithms (RSA, ECC) and read everything at once.
- **The software's countermeasure:** By integrating NIST's post-quantum primitives (ML-KEM, ML-DSA), CryptoToolkit renders data unreadable, even to a future quantum computer.
- **Crypto-agility:** If a flaw is found tomorrow in a post-quantum algorithm, the software can instantly swap the mathematical building block, with no need for the company to rewrite its system. NIST already has HQC standing by (included in CryptoToolkit™) in case Kyber-KEM is ever compromised.

2. Hybrid Encryption and Argon2id: The End of Impersonation and AI-Driven Cracking

- Today's AI cracks simple passwords within minutes.
- Argon2id (the world's most robust password-hashing algorithm) forces an attacker's computer or AI to consume enormous memory and time per passphrase attempt, making brute-force attacks mathematically and financially impossible.
- Hybrid encryption combines symmetric-encryption speed (for large files) with asymmetric-encryption security (for key exchange), so no identity spoofing can ever unlock the files.

3. Zero-Knowledge and Air-Gapped: The End of Data Leaks

- **Zero-Knowledge (total confidentiality):** ARANEA SOFT never holds your keys or passphrases. Even if the publisher is hacked, your data stays safe, since only you hold the mathematical key.
- **Air-Gapped (physical isolation):** The decisive argument. An air-gapped system runs on machines fully disconnected from the internet (no Wi-Fi, no cable, no Bluetooth). A remote AI or hacker simply cannot attack a machine that isn't on the network.

Two specific features change the game entirely:

Out-of-Band Transfer via Giant QR Code (Nayuki Optical Matrix)

- **The principle:** Instead of sending a key or critical file over a cable, USB drive, or the internet, the source computer displays a large, high-density QR code. The fully isolated destination computer simply "reads" it via a camera.
- **Why it defeats HNDL:** Since data travels as a direct optical stream (screen to camera) and never touches a cable or Wi-Fi signal, it's physically impossible for an AI or government to intercept and store it for later decryption.

RAM Reset and "Zeroization" in Tactical Environments

- **The classic problem:** Even powered off, encryption keys stay readable in RAM for seconds or minutes. Attackers can use freezing gas to "freeze" RAM and extract keys (Cold Boot attack).
- **CryptoToolkit's countermeasure:** A Rust engine ("absolute memory safety") paired with a brute-force shield (Argon2id "Bunker"). On alert, zeroization instantly overwrites all RAM used by the software with zeros — keys and passphrases vanish instantly, with no physical trace left to analyze.

A Highly Attractive "Net-Free" and "Total Sovereignty" Positioning

The product's real strength is offering complete quantum immunity (via NIST-recommended Kyber and Dilithium) with zero external dependency. In a European context bound by the NIS 2 directive, a lifetime-license, 100%-local tool (no cloud, no internet required) compliant with export rules has enormous economic potential for banks, armed forces, and advanced industry — targeting sectors under the strictest regulation:

- **Operators of Vital Importance:** energy (nuclear plants), water, transport — already running air-gapped networks.
- **Banking and finance:** protecting transactions against the future quantum threat.
- **Healthcare and research labs:** preventing long-term theft of patents or patient data.

- **Defense and space:** where state secrecy demands absolute zero-knowledge.

Conclusion

On paper, if CryptoToolkit delivers on its integration promises (especially ease of use for a system that's both post-quantum and isolated), it's a genuinely disruptive cyberdefense tool. Its real-world release should be watched closely to ensure the code is audited by official bodies (e.g. ANSSI in France) to validate this excellent technical profile.

FAQ

1) Rust, explained for a non-technical decision-maker: Rust is a modern programming language built to fix the weaknesses of older languages like C++, which remains powerful but notoriously hard to fully secure. Most security flaws worldwide trace back to one root cause: poor memory management — a risk C++ never structurally prevents, even with experienced teams. Rust blocks this class of error automatically, before shipping — like a built-in quality check that won't let a defective product leave the factory. For a business: fewer post-release incidents, fewer costly emergency patches, less reputational risk, faster time-to-market. And Rust loses nothing on speed — it matches C++. That's why Microsoft, Google, and Amazon are migrating sensitive systems to it, and why government agencies now recommend it for critical software — a strong trust signal for a cryptography product.

Ten-line case for using Rust to deploy NIST libraries like liboqs in a tool such as CryptoToolkit™:

1. Memory safety by design — no garbage collector, no buffer overflows, use-after-free, or data races: historically the source of the most critical CVEs in C++ + crypto implementations.
2. Safe encapsulation of liboqs — liboqs stays in C, but Rust's oqs bindings give a safe FFI layer isolating residual memory risk while keeping native speed.
3. Native performance — LLVM compilation and zero-cost abstractions put Rust on par with C/C++, decisive for high-volume ML-KEM/ML-DSA operations.
4. Real-time determinism — no GC runtime means predictable behavior, vital for air-gapped and embedded constraints.
5. The RustCrypto ecosystem — audited, pure-Rust primitives (AES, SHA-3) cut reliance on unvetted third-party C code.
6. Safe concurrency — ownership/borrowing prevents data races in multi-threaded processing of large file volumes.
7. Cross-platform portability — cargo and LLVM targets simplify Windows/Linux/macOS builds without rewrites, matching a 200-language interface.
8. Side-channel resistance — Rust eases constant-time code, reducing timing-attack exposure on Falcon or SLH-DSA signatures.
9. Institutional alignment — agencies like CISA recommending memory-safe languages for critical infrastructure is a differentiator versus C/C++ competitors.

10. Certification ease — a more auditable, lower-surface Rust codebase simplifies paths to Common Criteria or FIPS certification.

2) "Kyber and Falcon mean much larger keys/signatures than RSA or ECC.

How does CryptoToolkit™ handle bandwidth and memory impact, especially offline/air-gapped?" "That's exactly why CryptoToolkit™ was built offline-first from day one. Removing the backend and continuous network flows eliminates the bandwidth bottleneck entirely — processing happens purely in local volatile memory. For memory allocation, our implementation optimizes the Merkle tree (notably for SPHINCS+) and uses compact data structures so the RAM footprint stays invisible to the user, even on hardened military hardware."

3) (CPU performance / energy consumption): "Post-quantum algorithms — Falcon signatures especially — demand heavy floating-point computation that can saturate non-optimized CPUs. How do you scale to lightweight terminals or industrial PCs?" "That's our strength: the multi-algorithm approach. CryptoToolkit™ doesn't force a technological 'big bang' — we offer hybridization. For low-resource terminals, we can pair classic AES-256 with NTRU (lattice-based, lighter than Kyber for encryption), or use optimized Kyber variants without heavy floating-point math. The user or admin sets the balance between maximum post-quantum security and CPU performance for the target hardware."

4) The 5 pillars of our value proposition:

1. Rust: only Rust's architecture allows implementing NIST-standardized libraries like "liboqs" with PQC algorithms (Kyber-KEM, NTRU, HQC, Falcon, Dilithium, SPHINCS+).
2. Zero backend / zero cloud: no telemetry, no server — network leak risk is zero.
3. Degraded environment: 100% functional on air-gapped machines, in the field, or during a systemic crisis.
4. Auditable code: strict local trust boundary (volatile memory only), sized for certifications (CSPN / Restricted Distribution).
5. Zero infrastructure cost: no servers to maintain for the publisher — immediate license profitability.

5) (Migration and interoperability): "If clients use CryptoToolkit™ fully isolated (no server, no backend), how do you distribute post-quantum public keys between users? How do you avoid offline key-management hell?" "CryptoToolkit™ is an encryption tool and a standalone testbed, not a centralized network manager. It imposes no transport layer. Public keys and encrypted files (bearing Falcon or Dilithium signatures) export in standardized formats (secure files, containers). In Defense or critical-industry contexts, they travel via the organization's existing trusted channels (secure USB drives, tactical data links, hermetic local intranets). We secure the data at the source — we don't manage the cable."

6) (Certification and trust): "NIST standardized Kyber and Falcon, but ANSSI (France) recommends a cautious hybrid approach and doesn't yet fully clear 100% autonomous post-quantum solutions. Where do you stand on certification?" "We

follow ANSSI's doctrine closely. That's why CryptoToolkit™ offers active coexistence: we don't replace classical crypto with quantum crypto, we layer them (e.g. AES-256 + Kyber). If a new algorithm shows an unforeseen weakness, the classical layer (RSA-4096 / AES) keeps protecting the data. CryptoToolkit™'s architecture (zero backend, auditable code, strict compartmentalization) was designed to maximally ease the path toward CSPN certification or Restricted Distribution clearance."

7) What is the key-derivation module protecting passphrases and institutional secrets? Argon2id is currently the most robust password-derivation algorithm for critical environments. Winner of the Password Hashing Competition (PHC), it combines a memory-hard design, resistance to massive GPU attacks, and side-channel protection. CryptoToolkit™ uses Argon2id to secure its 50-passphrase vault, with 4-criteria, 128-character passwords — ensuring even a state-level attacker cannot derive a stored secret.

Why Argon2id is indispensable for financial infrastructure:

- *Memory-hard architecture*: heavy memory load per attempt; unlike PBKDF2, bcrypt, or scrypt, it can't be efficiently parallelized — immediate GPU saturation, no massive attacks, exponential cost per try.
- *GPU-attack resistance*: GPUs run thousands of threads in parallel; Argon2id blocks this via per-thread memory cost, sequential dependency, and non-parallelizable computation — forcing attackers into prohibitively slow, resource-heavy attempts.
- *Protection against state-level actors*: financial infrastructure faces GPU farms, HPC clusters, and massive cloud resources — Argon2id makes all of that useless for deriving a secret.

8) Why Argon2id matters in a post-quantum context: Though not a PQC algorithm itself, Argon2id is essential to protect passphrases, passwords, internal keys, signing secrets, and sensitive archives. Quantum computers accelerate attacks on RSA, ECC, and Diffie-Hellman — but not Argon2id, since it's memory-hard, non-parallelizable, and not optimizable by Grover's algorithm.

9) The PANIC BUTTON / RAM RESET / zeroization module for physical intrusion or machine seizure: Anti-intrusion security (zeroization): a Panic Button triggers instant RAM reset and zeroing of data flows on unauthorized physical access. This exclusive CryptoToolkit™ feature wipes RAM and fills it with zeros in seconds, confirmed via a button available in every language.

On intrusion or tactical seizure:

Step 1 — Abort Streams: The Rust thread handling file read/write buffers (chunking) is killed instantly (abort / CancellationToken), halting all processing to prevent leaked file fragments.

Step 2 — Zeroize RAM: The critical step. Dropping a Rust variable removes its pointer, but sensitive data (a BIP-39 passphrase, RSA private keys) stays

physically in RAM until the OS reallocates it — an attacker with physical access could extract it. Solution: the official zeroize crate overwrites key-holding memory with binary zeros (0x00) before it's freed.

Step 3 — UI Lockdown: Once RAM is wiped of all cryptographic traces — the status gauge snaps back to Red  (zero); all open modal windows (RSA, PQC, QR) are force-closed and cleared; the interface locks and redirects to the passphrase-generator home screen, resetting the session.

Exact code behavior on Panic Button click (Schneider fist icon):

[Panic Button Click (Tauri Front-end)]



▼ (Rust Invoke command)

[1. Immediate Stream/Buffer interruption]



[2. Overwrite volatile memory with zeros (Zeroize RAM)]



[3. Global State purge & UI lockdown]