Open-Source Post-Quantum Cryptographic Trust Infrastructure: Architecture, Hardening, and Ethical Design of the AKR Platform

Adel Tammam
PTH Meridian — Precision Technology Heuristics
Calgary, Alberta, Canada
adel.tammam@pth-meridian.io

Abstract—The transition from classical to post-quantum cryptographic infrastructure represents one of the most significant security engineering challenges of the current decade. Widely deployed asymmetric cryptosystems, including RSA and elliptic curve cryptography, are vulnerable to polynomial-time attacks by sufficiently powerful quantum computers. This paper presents the design, implementation rationale, and ethical framework of the AKR platform — an open-source, multi-phase cryptographic trust system comprising AKR Naos and AKR KeyGen. AKR Naos implements six integrated phases: distributed secret management via Shamir's Secret Sharing, memory-hard key derivation using Argon2id, a two-tier certificate authority with post-quantum signature support, mutual authentication with authenticated encryption, a cryptographic audit engine with hash chaining, and a self-sovereign identity wallet with zero-knowledge proofs. AKR KeyGen implements the three 2024 NIST post-quantum standards: ML-KEM (FIPS 203), ML-DSA (FIPS 204), and SLH-DSA (FIPS 205). Seven hardening mechanisms are documented including HSM bridging, threshold signatures (3-of-5), key attestation, and ZKP v2 bounds binding. This paper documents the self-discovery and responsible disclosure of a zero-knowledge proof vulnerability (CVE-AKR-NAOS-2026-001) prior to any external deployment, and articulates an ethical framework for open-source cryptographic infrastructure grounded in transparency, independent auditability, and democratic access to security primitives.

Index Terms—post-quantum cryptography, zero-knowledge proofs, Shamir Secret Sharing, Argon2id, certificate authority, ML-KEM, ML-DSA, SLH-DSA, threshold signatures, responsible disclosure, open-source security, self-sovereign identity, decentralized identifiers.

I. Introduction

In 1994, Peter Shor demonstrated that a sufficiently powerful quantum computer could factor large integers and compute discrete logarithms in polynomial time [1], directly threatening the security foundations of RSA [2], elliptic curve cryptography, and the Diffie-Hellman key exchange family. While large-scale fault-tolerant quantum computers capable of executing Shor's algorithm at cryptographically relevant key sizes do not yet exist, the National Security Agency [3] and NIST [4][5][6] have concluded that migration to quantum-resistant standards must begin immediately. The rationale is twofold: cryptographic infrastructure has long operational lifespans, and adversaries may execute "harvest now, decrypt later" collection strategies against encrypted data whose confidentiality must be preserved beyond the quantum transition horizon.

In August 2024, NIST finalized three post-quantum cryptographic standards: ML-KEM (FIPS 203) [4] for key encapsulation, ML-DSA (FIPS 204) [5] for digital signatures, and SLH-DSA (FIPS 205) [6] for stateless hash-based signatures. U.S. federal agencies are mandated to complete the transition by 2030. The engineering challenge of implementing and integrating these standards into coherent, production-ready trust infrastructure — while maintaining interoperability with existing PKI, identity, and audit systems — is substantial and largely unaddressed by existing open-source implementations.

This paper documents the AKR platform: a six-phase open-source cryptographic trust system addressing this challenge. The primary contributions are: (1) a multi-phase system architecture integrating classical and post-quantum primitives into a coherent trust chain; (2) documentation of seven hardening mechanisms applied to the platform; (3) a case study in responsible disclosure for a self-discovered ZKP implementation vulnerability; and (4) an ethical framework for open-source cryptographic infrastructure development.

II. Background and Related Work

A. The Quantum Threat

The security of RSA [2] rests on the computational hardness of integer factorization. Shor's algorithm [1] solves this in O((log N)³) time on a quantum computer, rendering RSA and discrete-logarithm-based systems insecure against quantum adversaries. Grover's algorithm [7] provides a quadratic speedup for unstructured search, effectively halving the security level of symmetric primitives and hash functions — addressed by key length doubling rather than algorithm replacement. Bernstein and Lange [8] survey the post-quantum landscape comprehensively, assessing lattice, code-based, hash-based, and isogeny-based approaches by maturity and confidence level.

B. NIST Post-Quantum Standards

ML-KEM (FIPS 203) [4], derived from CRYSTALS-Kyber, is a lattice-based key encapsulation mechanism based on the Module Learning With Errors (MLWE) problem, providing IND-CCA2 security. ML-DSA (FIPS 204) [5], derived from CRYSTALS-Dilithium, provides EUF-CMA secure digital signatures based on the MSIS and MLWE problems. SLH-DSA (FIPS 205) [6], derived from SPHINCS+, is a stateless hash-based signature scheme with security reducing to hash function properties, providing algorithmic conservatism independent of lattice assumption confidence. Fig. 1 illustrates the key and signature size tradeoffs across all three standards and their security levels.

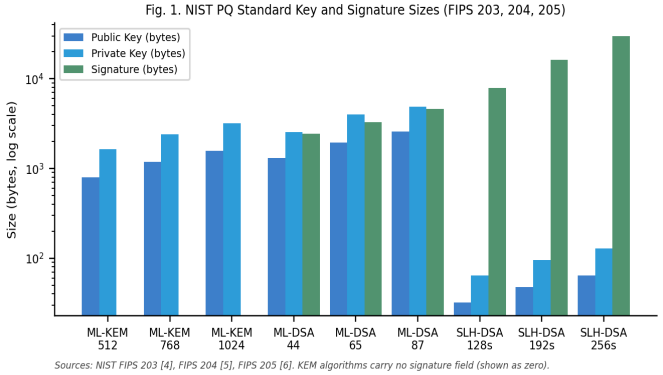

Fig. 1. NIST post-quantum standard key and signature sizes by algorithm and security level. Logarithmic scale. Sources: NIST FIPS 203 [4], FIPS 204 [5], FIPS 205 [6].

C. Related Primitives

Shamir's (k,n) threshold secret sharing [9], independently developed by Blakley [10], provides information-theoretically secure secret distribution since 1979. Memory-hard key derivation via Argon2id [11], winner of the Password Hashing Competition, resists GPU and ASIC-accelerated brute-force attacks. Zero-knowledge proofs, introduced by Goldreich, Micali, and Wigderson [12] and made non-interactive via the Fiat-Shamir transform [13], underpin the identity layer. The W3C Decentralized Identifier [14] and Verifiable Credential [15] specifications form the identity data model on which the AKR self-sovereign identity module is built.

III. System Architecture and Design Philosophy

The AKR platform comprises two components: AKR Naos, a six-phase cryptographic trust platform, and AKR KeyGen, a post-quantum key generation engine. Both are implemented in TypeScript on Node.js and published as open-source at github.com/PTHMeridian. Fig. 2 illustrates the overall architecture.

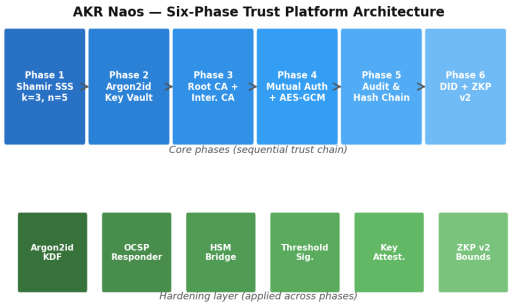

Fig. 2. AKR Naos architecture. Each phase extends the cryptographic guarantees of the preceding phase. Hardening mechanisms span multiple phases.

Fig. 2. AKR Naos six-phase trust platform architecture with hardening layer. Each phase builds on the cryptographic guarantees established by the preceding phase.

The system is designed around four principles. *Defense in depth*: each phase provides independent security guarantees such that single-layer compromise does not collapse the trust chain. *Algorithm agility*: multiple algorithms are supported at each layer, permitting migration as standards evolve. *Auditability*: all protocols and derivation parameters are open-source and independently inspectable per Kerckhoffs's principle. *Minimal trust assumptions*: the security model specifies exactly which guarantees degrade under each defined failure mode.

IV. AKR Naos: Six-Phase Trust Platform

A. Phase 1: Distributed Secret Management

Phase 1 implements Shamir's (k,n) Secret Sharing [9] with threshold k=3 and total shares n=5. The scheme constructs a random polynomial f(x) of degree k−1 over a prime field such that f(0) = S (the secret). n evaluations are distributed as shares; any k suffice for Lagrange interpolation. The scheme is information-theoretically secure: any fewer than k shares reveal zero information about S regardless of computational resources [9]. Fig. 3 illustrates the construction and threshold property.

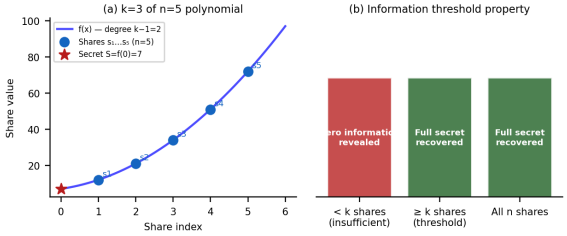

Fig. 3. Shamir (k,n) Secret Sharing [9]. (a) Secret at f(0) recovered by Lagrange interpolation from any k shares. (b) Fewer than k shares reveal no information.

Fig. 3. Shamir (k,n) Secret Sharing [9]. (a) Polynomial interpolation from k=3 of n=5 shares. (b) Information-theoretic threshold: fewer than k shares reveal nothing.

B. Phase 2: Memory-Hard Key Derivation

Phase 2 uses Argon2id [11], recommended by NIST SP 800-63B [16] and standardized for IETF protocols by RFC 9106 [17]. Two parameterization tiers are implemented: INTERACTIVE (64 MB memory, calibrated for interactive latency) and SENSITIVE (1 GB memory, for high-value key derivation). Both exceed RFC 9106 minimum recommendations. Argon2id's hybrid data-access pattern resists both time-memory tradeoff attacks (Argon2d property) and side-channel attacks (Argon2i property) [11].

C. Phase 3: Certificate Authority

Phase 3 implements a two-tier X.509 CA per RFC 5280 [18]: Root CA and Intermediate CA, with revocation via CRL (RFC 5280) [18] and OCSP (RFC 2560) [19]. The Intermediate CA signs end-entity certificates, isolating the Root CA from online operations. ML-DSA-65 (FIPS 204, security level 3) [5] is used for certificate signing, providing 128-bit post-quantum security with a 1,952-byte public key and 3,293-byte signature — a critical property for certificates whose validity period may extend into the post-quantum era.

D. Phase 4: Mutual Authentication and Secure Channel

Phase 4 implements mutual certificate authentication followed by an ML-KEM (FIPS 203) [4] key encapsulation exchange, yielding shared session key material. Session data is protected under AES-256-GCM per NIST SP 800-38D [20], with HMAC per FIPS 198-1 [21] providing per-message integrity. Tamper detection triggers immediate session termination and audit event generation.

E. Phase 5: Cryptographic Audit Engine

Phase 5 implements an append-only audit log using SHA-256 hash chaining [22]: each entry includes H(preceding entry), forming a tamper-evident chain. Retroactive modification of any entry invalidates all subsequent hashes. The HMIT (Hash-based Message Integrity and Tamper) Protocol governs alert lifecycle management, escalation thresholds, and cryptographically authenticated alert chains.

F. Phase 6: Self-Sovereign Identity and ZKP

Phase 6 implements a W3C DID v1.0 [14] and Verifiable Credentials v1.1 [15] compliant identity wallet. Zero-knowledge capability is provided via non-interactive Fiat-Shamir proofs [13] in three configurations: (1) ZKP v2 bounds-binding range proofs (Section VI.D), (2) Merkle membership proofs for set inclusion without element disclosure, and (3) selective disclosure presentations. Theoretical foundations follow Goldreich et al. [12] and Pedersen commitments [23].

V. AKR KeyGen: Post-Quantum Key Generation

AKR KeyGen exposes a REST API and developer dashboard implementing all three NIST post-quantum standards. ML-KEM [4] provides IND-CCA2 key encapsulation at three security levels (512/768/1024); ML-DSA [5] provides EUF-CMA signatures at levels 2/3/5 (ML-DSA-44/65/87). ML-DSA-65 is the primary AKR Naos CA parameter set. SLH-DSA [6] provides hash-reduction security for high-assurance contexts (root CA operations, firmware signing, long-lived document authentication) where signature size is not the primary constraint. AKR KeyGen has not yet undergone independent formal security audit and is not recommended for production deployment until audit is complete.

VI. Hardening Mechanisms

Seven hardening steps are applied to the AKR platform; six are complete. The outstanding step — independent cryptographic audit — is a prerequisite for production deployment.

A. HSM Bridge

Hardware Security Module integration provides FIPS 140-3 [24] compliant hardware-backed key storage. Private key material never leaves the HSM boundary in plaintext form. The bridge abstracts vendor APIs, supporting multi-vendor HSM deployment without modification to higher-layer platform code.

B. Threshold Signatures (3-of-5)

Following Desmedt and Frankel [25] and the construction of Gennaro et al. [26], threshold signature generation requires k=3 of n=5 parties to contribute partial signatures. This prevents unilateral signing by any single party, tolerates up to n–k=2 party failures, and distributes signing authority across geographically or organizationally independent key holders.

C. Key Attestation

Key attestation provides cryptographic evidence that key material was generated within a hardware-protected environment and has not been exported, linking the attested key to the device's hardware root of trust per NIST SP 800-57 [27]. This allows relying parties to verify key provenance without direct access to the generating device.

D. ZKP v2 — Bounds Binding

ZKP v2 introduces cryptographic commitment of range proof bounds in the Fiat-Shamir challenge [13]. The commitment is computed as: $C = H(min \blacksquare max \blacksquare nonce)$, where H is SHA-256 [22] and nonce is session-fresh. Including C in the challenge hash binds the proof to specific bounds, invalidating substitution attacks. The construction follows Pedersen [23]. This modification addresses CVE-AKR-NAOS-2026-001 (Section VII).

VII. Vulnerability Disclosure: CVE-AKR-NAOS-2026-001

During internal review of the Phase 6 ZKP module, a soundness vulnerability was identified: range proof bounds [min, max] were supplied as public parameters without cryptographic commitment in the Fiat-Shamir challenge. An adversary could substitute alternative bounds post-generation while maintaining proof validity, undermining the range proof's soundness guarantee. The vulnerability was self-discovered prior to any external deployment or code disclosure.

Disclosure followed coordinated responsible disclosure guidelines per CERT/CC [28] and the MITRE CVE program [29]: internal documentation of the vulnerability class, root cause, and impact scope; patch development and verification (bounds commitment via challenge binding); CVE identifier assignment; and public disclosure concurrent with the patched release. Fig. 4 illustrates the disclosure timeline.

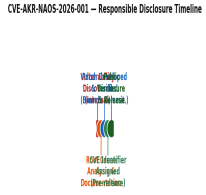

Fig. 4. Responsible disclosure timeline for CVE-AKR-NAOS-2026-001 (ZKP bounds binding). Vulnerability self-discovered, patched, and disclosed before any external deployment. Process follows CERT/CC coordinated disclosure guidelines [28].

Fig. 4. CVE-AKR-NAOS-2026-001 responsible disclosure timeline. Self-discovered, patched, and disclosed before any external deployment. Per CERT/CC guidelines [28].

This case illustrates a principle documented in the security engineering literature: the disclosed and patched vulnerability record is more trustworthy evidence of implementation quality than the absence of disclosed vulnerabilities in an unreviewed system [30][31]. The self-discovery and voluntary disclosure of CVE-AKR-NAOS-2026-001 — without external discovery or commercial pressure — demonstrates the practical value of rigorous internal review practices in open-source cryptographic projects.

VIII. Ethical Dimensions of Cryptographic Infrastructure

A. Cryptography as a Public Good

Strong cryptographic infrastructure is a prerequisite for privacy, secure commerce, democratic communication, and protection of personal and institutional data. When such infrastructure is controlled exclusively by proprietary systems, deployers must accept the security claims of the developer on trust. Open-source implementation enables independent verification — the condition for rational trust that Kerckhoffs identified as foundational in 1883 and that remains the operative standard of modern cryptographic practice.

B. Democratic Access to Security

Post-quantum cryptographic capability, if accessible only through expensive proprietary implementations, concentrates security advantage among well-resourced organizations. Open-source post-quantum implementations reduce this access differential, enabling small organizations, government agencies in resource-constrained environments, academic institutions, and individual developers to deploy the same cryptographic capability as large commercial actors. This is consistent with the public-good framing of the NIST standardization process [4][5][6].

C. Transparent Limitation Disclosure

AKR KeyGen carries an explicit limitation disclosure: independent formal security audit has not yet been completed and production deployment is not recommended until such audit is obtained. The authors consider this disclosure an unconditional ethical obligation. The practice of deploying cryptographic software without adequate review has historically resulted in widespread security failures [30][31]. Honest capability representation is a prerequisite for informed deployment decisions by adopters.

IX. Discussion

Several observations merit discussion beyond the technical implementation details.

The integration of post-quantum primitives with existing X.509 PKI infrastructure is technically feasible. The AKR approach — using ML-DSA-65 for CA signing within RFC 5280 [18] certificate structure — is consistent with emerging IETF guidance on hybrid PKI transition [32]. However, hybrid transition scenarios in which classical and post-quantum certificates coexist require careful handling of downgrade and algorithm negotiation attacks that are active areas of standardization.

The CVE-AKR-NAOS-2026-001 vulnerability demonstrates a well-documented class of cryptographic implementation error: correct mathematical foundations with incorrect parameter binding in the instantiation. This error class — present in high-profile deployed systems including TLS implementations [33] — underscores that correct theory does not guarantee correct instantiation, and that independent audit is necessary regardless of implementer competence.

Limitations. This paper describes a systems implementation rather than novel cryptographic results. Security rests on the security of the underlying cited primitives. No formal security proof of the composite system is provided here. Independent audit by a qualified cryptographic laboratory is required before production deployment. Performance benchmarking constitutes future work.

X. Conclusion

This paper has described the architecture, implementation rationale, hardening mechanisms, and ethical framework of the AKR post-quantum cryptographic trust platform. The system integrates established cryptographic primitives — Shamir Secret Sharing, Argon2id, X.509 PKI with CRL and OCSP, AES-256-GCM, SHA-256 hash chaining, Fiat-Shamir ZKP with Pedersen commitments, W3C DID and Verifiable Credentials — with the three 2024 NIST post-quantum standards into a coherent six-phase trust chain with seven applied hardening steps.

The self-discovery and responsible disclosure of CVE-AKR-NAOS-2026-001 demonstrates that rigorous internal review practices can identify cryptographic implementation vulnerabilities before external exposure, and that voluntary public disclosure of discovered vulnerabilities strengthens rather than undermines system confidence. The ethical framework presented argues that post-quantum cryptographic capability should be open-source, independently auditable, and broadly accessible — as a matter of both practical security and equitable access to infrastructure that protects privacy and enables secure communication for all. Independent security audit remains a prerequisite for production deployment and is actively sought from qualified cryptographic research groups.

Glossary of Terms

Argon2id
Memory-hard key derivation function combining data-dependent and data-independent access patterns. Winner of the Password Hashing Competition, 2015 [11]. Standardized by RFC 9106 [17].

Certificate Revocation List (CRL)
A signed list of revoked certificates published by a Certificate Authority per RFC 5280 [18], allowing relying parties to verify revocation status without real-time lookup.

CVE
Common Vulnerabilities and Exposures: a standardized identifier system maintained by MITRE [29] for publicly known cybersecurity vulnerabilities.

DID (Decentralized Identifier)
A W3C standard [14] for cryptographically verifiable identifiers that resolve to DID Documents without dependence on centralized identity registries.

Fiat-Shamir Transform
A method [13] for converting interactive identification protocols into non-interactive zero-knowledge proofs by replacing the verifier challenge with a hash function output.

HSM (Hardware Security Module)

FIPS 140-3 [24] certified hardware device providing key storage and cryptographic operations within a physically secure boundary. Private keys never leave in plaintext.

IND-CCA2

Indistinguishability under adaptive chosen-ciphertext attack: the standard security notion for key encapsulation mechanisms, including ML-KEM [4].

ML-KEM (FIPS 203)

Module-Lattice-Based Key-Encapsulation Mechanism [4]. 2024 NIST post-quantum KEM standard based on CRYSTALS-Kyber.

ML-DSA (FIPS 204)

Module-Lattice-Based Digital Signature Algorithm [5]. 2024 NIST post-quantum signature standard based on CRYSTALS-Dilithium.

OCSP

Online Certificate Status Protocol [19]. Provides real-time certificate revocation verification per RFC 2560.

Shamir Secret Sharing

A (k,n) threshold scheme [9] encoding a secret as the free coefficient of a random polynomial. Any k of n shares reconstruct the secret; fewer than k reveal zero information.

SLH-DSA (FIPS 205)

Stateless Hash-Based Digital Signature Standard [6]. 2024 NIST post-quantum signature standard based on SPHINCS+, with security reducing to hash function properties.

Zero-Knowledge Proof (ZKP)

A protocol [12] in which a prover demonstrates knowledge or set membership without revealing the secret or any information beyond the validity of the claim.

References

[1] P. W. Shor, “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,” *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997. doi: 10.1137/S0097539795293172

[2] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978. doi: 10.1145/359340.359342

[3] National Security Agency, “Announcing the Commercial National Security Algorithm Suite 2.0,” NSA Cybersecurity Advisory, Sep. 2022. [Online]. Available: https://media.defense.gov/2022/Sep/07/2003071834/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS_.PDF

[4] National Institute of Standards and Technology, “Module-Lattice-Based Key-Encapsulation Mechanism Standard,” FIPS PUB 203, Aug. 2024. doi: 10.6028/NIST.FIPS.203

[5] National Institute of Standards and Technology, “Module-Lattice-Based Digital Signature Standard,” FIPS PUB 204, Aug. 2024. doi: 10.6028/NIST.FIPS.204

[6] National Institute of Standards and Technology, “Stateless Hash-Based Digital Signature Standard,” FIPS PUB 205, Aug. 2024. doi: 10.6028/NIST.FIPS.205

[7] L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proc. 28th Annual ACM Symposium on Theory of Computing (STOC)*, New York, NY, 1996, pp. 212–219. doi: 10.1145/237814.237866

[8] D. J. Bernstein and T. Lange, “Post-quantum cryptography,” *Nature*, vol. 549, pp. 188–194, 2017. doi: 10.1038/nature23461

[9] A. Shamir, “How to share a secret,” *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979. doi: 10.1145/359168.359176

[10] G. R. Blakley, “Safeguarding cryptographic keys,” in *Proc. AFIPS National Computer Conference*, vol. 48, 1979, pp. 313–317.

[11] A. Biryukov, D. Dinu, and D. Khovratovich, “Argon2: New generation of memory-hard functions for password hashing and other applications,” in *Proc. IEEE European Symposium on Security and Privacy (EuroS&P;)*, Saarbrücken, Germany, 2016, pp. 292–302. doi: 10.1109/EuroSP.2016.31

[12] O. Goldreich, S. Micali, and A. Wigderson, “Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems,” *Journal of the ACM*, vol. 38, no. 3, pp. 690–728, 1991. doi: 10.1145/116825.116852

[13] A. Fiat and A. Shamir, “How to prove yourself: Practical solutions to identification and signature problems,” in *Advances in Cryptology — CRYPTO 1986*, LNCS vol. 263, Berlin: Springer, 1987, pp. 186–194. doi: 10.1007/3-540-47721-7_12

[14] M. Sporny, D. Longley, M. Sabadello, D. Reed, O. Steele, and C. Allen, “Decentralized Identifiers (DIDs) v1.0,” W3C Recommendation, Jul. 2022. [Online]. Available: https://www.w3.org/TR/did-core/

[15] M. Sporny, D. Longley, and D. Chadwick, “Verifiable Credentials Data Model v1.1,” W3C Recommendation, Mar. 2022. [Online]. Available: https://www.w3.org/TR/vc-data-model/

[16] P. Grassi et al., “Digital Identity Guidelines: Authentication and Lifecycle Management,” NIST SP 800-63B, Jun. 2017 (updated 2020). doi: 10.6028/NIST.SP.800-63b

[17] T. Irtimaa, O. Friel, and S. Turner, “The Argon2 Memory-Hard Function for Password Hashing and Proof-of-Work Applications,” RFC 9106, IETF, Sep. 2021. doi: 10.17487/RFC9106

[18] D. Cooper et al., “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,” RFC 5280, IETF, May 2008. doi: 10.17487/RFC5280

[19] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, “X.509 Internet Public Key Infrastructure Online Certificate Status Protocol — OCSP,” RFC 2560, IETF, Jun. 1999. doi: 10.17487/RFC2560

[20] M. Dworkin, “Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC,” NIST SP 800-38D, Nov. 2007. doi: 10.6028/NIST.SP.800-38D

[21] National Institute of Standards and Technology, “The Keyed-Hash Message Authentication Code (HMAC),” FIPS PUB 198-1, Jul. 2008. doi: 10.6028/NIST.FIPS.198-1

[22] National Institute of Standards and Technology, “Secure Hash Standard (SHS),” FIPS PUB 180-4, Aug. 2015. doi: 10.6028/NIST.FIPS.180-4

[23] T. P. Pedersen, “Non-interactive and information-theoretic secure verifiable secret sharing,” in *Advances in Cryptology — CRYPTO 1991*, LNCS vol. 576, Berlin: Springer, 1992, pp. 129–140. doi: 10.1007/3-540-46766-1_9

[24] National Institute of Standards and Technology, “Security Requirements for Cryptographic Modules,” FIPS PUB 140-3, Mar. 2019. doi: 10.6028/NIST.FIPS.140-3

[25] Y. Desmedt and Y. Frankel, “Threshold cryptosystems,” in *Advances in Cryptology — CRYPTO 1989*, LNCS vol. 435, Berlin: Springer, 1990, pp. 307–315. doi: 10.1007/0-387-34805-0_28

[26] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, “Robust threshold DSS signatures,” *Information and Computation*, vol. 164, no. 1, pp. 54–84, 2001. doi: 10.1006/inco.2000.2881

[27] E. Barker, “Recommendation for Key Management, Part 1: General,” NIST SP 800-57 Part 1 Rev. 5, May 2020. doi: 10.6028/NIST.SP.800-57pt1r5

[28] CERT Coordination Center, “Vulnerability Disclosure Policy,” Carnegie Mellon University Software Engineering Institute, Pittsburgh, PA. [Online]. Available: https://www.sei.cmu.edu/about/divisions/cert/

[29] MITRE Corporation, “Common Vulnerabilities and Exposures (CVE) Program,” 2024. [Online]. Available: https://cve.mitre.org

[30] M. Green, “The many flaws of Dual_EC_DRBG,” *A Few Thoughts on Cryptographic Engineering* (blog), Sep. 2013. [Online]. Available: https://blog.cryptographyengineering.com

[31] B. Schneier, *Secrets and Lies: Digital Security in a Networked World*. New York: Wiley, 2000.

[32] D. Stebila, S. Fluhrer, and S. Gueron, “Hybrid Key Exchange in TLS 1.3,” IETF Internet-Draft draft-ietf-tls-hybrid-design, 2023. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/

[33] N. Aviram et al., “DROWN: Breaking TLS using SSLv2,” in *Proc. 25th USENIX Security Symposium*, Austin, TX, 2016, pp. 689–706.