QuanTEEum: Quantum Cryptography via TEEs

Shoaib Ahmed

Independent Researcher
sufialhussaini@gmail.com

Abstract. One-shot signatures (OSS) have emerged as a versatile abstraction underpinning various blockchain-facing applications. Existing OSS constructions rely on quantum hardware that will not be widely available soon. Naïve emulations using a single trusted execution environment (TEE) inherit brittle unclonability requirements and concentrate integrity risk in one enclave, while certified deletion on a single device is notoriously hard.

We present QuanTEEum, an honest-minority distributed protocol that realizes certified deletion and OSS by running threshold signing entirely *inside* heterogeneous TEEs. In QuanTEEum, n enclaves run a threshold DKG, produce exactly one threshold signature on the designated message $m^\star$, and each emits a remote attestation (RA) binding an in-enclave deletion event; security is *additive* across heterogeneous TEEs, as independent deletion attestations strictly raise the minimum cost-of-attack, and TEE heterogeneity provides defense-in-depth against supply-chain risks. We formalize the *exact-one* acceptance condition for one-shot signing in a remote attestation model under an abstract replay predicate Fresh (enforcing per-session counter monotonicity), and we *discuss* pluggable attestation back ends—(a) conventional TEE RA, (b) cryptoeconomic attestations via escrowed collateral, and (c) future quantum instantiations—yielding a clean upgrade path. We sketch applications to OSS and Quantum money, and discuss liveness, freshness, and rollback subtleties.

1 Introduction

Motivation. One-shot signatures (OSS) guarantee that a secret signing key can produce at most one valid signature; thereafter any further signing with that key is infeasible. Certified deletion complements OSS by providing verifiable deletion of the relevant secrets after use. Together, these primitives enable applications such as quantum money [9], perfect-finality blockchains [3], smart contracts without a blockchain [7], ordered and budgeted signatures, and ceremony-free Groth16 CRS [4]. Current TEE-based attempts at realizing OSS typically use a single enclave, making certified deletion both hard and brittle. The central tension is that *unclonability* pushes toward keeping a single non-replicated secret, whereas *robustness and defense-in-depth* call for replication across heterogeneous TEEs. While supply chain hardening and reproducible builds are crucial, defense-in-depth via *heterogeneous* enclaves gives additive security. This motivates a design that achieves exact-once semantics while distributing trust across multiple enclaves.

Our approach. QuanTEEum lifts OSS to an *honest-minority* setting by distributing the signing key across independent (heterogeneous) enclaves via a threshold DKG and binding the resulting (group) public key with per-enclave DKG attestations. The system produces exactly one threshold signature on the designated message and then aggregates per-enclave certified-deletion attestations bound to the same session. Exact-once acceptance reduces to the infeasibility of recovering a deleted threshold share from *at least one* honest enclave, plus freshness/anti-replay for attestations and standard threshold unforgeability. We adopt the *relaxed OSS* notion [5]: one-shotness today via TEE-backed certified deletion, with a clean upgrade path to quantum implementations.

Contributions.

- A formalization of the *exact-one* acceptance condition for one-shot signing in a remote attestation model with explicit context binding, parameterized by an abstract freshness predicate Fresh.
- QuanTEEum: a DKG-first threshold-in-TEE protocol that yields *exactly one* threshold signature on a designated message and a set of per-enclave deletion attestations; security is additive across heterogeneous TEEs.
- A verifier-facing artifact and algorithm: the *attested one-shot certificate* $\mathsf{AOSC}_{\mathsf{sid}}$ and *VerifyAOSC*, enabling third-party validation under Fresh.
- Application sketches for (i) QuanTEEum One-shot signature, and (ii) QuanTEEum money.
- A discussion of liveness under abort, anti-replay/rollback resistance, policy-driven heterogeneity as defense-in-depth, and on-chain instantiation/incentives.

2 Background and Model

2.1 One-shot signatures in brief

A one-shot signature scheme [2] is a tuple $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ with syntax:

$$\mathsf{Gen}(\mathsf{crs}) \to (pk, sk), \qquad \mathsf{Sign}(sk, m) \to \sigma, \qquad \mathsf{Ver}(\mathsf{crs}, pk, m, \sigma) \to b \in \{0, 1\}.$$

Correctness: if $(pk, sk) \leftarrow \mathsf{Gen}(\mathsf{crs})$ and $\sigma \leftarrow \mathsf{Sign}(sk, m)$ then $\mathsf{Ver}(\mathsf{crs}, pk, m, \sigma) = 1$.

One-shot unforgeability: no probabilistic polynomial-time (PPT) adversary, given (crs, pk) and oracle access to $\mathsf{Sign}(sk, \cdot)$, can output two distinct valid pairs $(m_1, \sigma_1) \neq (m_2, \sigma_2)$ with $\mathsf{Ver}(\mathsf{crs}, pk, m_1, \sigma_1) = \mathsf{Ver}(\mathsf{crs}, pk, m_2, \sigma_2) = 1$ except with negligible probability.

2.2 Threshold signatures

We assume a Schnorr-style t-of-n threshold signature with a secure, coordinator-orchestrated DKG (e.g., FROST [6]), where n is the number of participating enclaves and t is the signing threshold. A coordinator C schedules rounds and

relays messages but learns no secret material; all DKG/signing traffic is carried over attested channels (RA–TLS), and secret shares sk_i never leave enclaves. Each party P_i holds an additive share sk_i; together they define the group public key pk, and interactive signing with fresh per-round nonces from any set of t parties produces signature σ on message m. Security assumes a correct DKG whose transcripts reveal nothing beyond pk, fewer than t shares exposed, and no nonce reuse or bias. We primarily instantiate $t{=}n$; the case $t{<}n$ is discussed in §6.

2.3 TEE trust and remote attestation

Each party P_i hosts an enclave E_i implementing $\mathsf{QuanTEEum}$. We assume:

- **Code identity**: attestation binds a code measurement and the *auxiliary reportdata* to outputs; verifier policy is committed via $\mathsf{policy\_hash}$ in sid.
- **Sealed state semantics**: used only pre-signing to persist the DKG result across a crash.
- **Freshness/anti-replay**: a monotonic *counter* is available (hardware counter or external append-only anchor) and is bound into attestations as $\mathsf{ctr} \in \{0, 1, 2\}$ corresponding to JOIN, DKG, and DEL.
- **Deletion API**: an enclave can make shares unrecoverable (*zeroization*) and prove the transition with an attestation carrying auxiliary data.

We *do not* assume perfect side-channel resistance or that all rollback vectors are impossible; instead we bind acceptance to the *freshness predicate* and to a one-shot latch.

Assumption (Replay protection Fresh). There exists a predicate $\mathsf{Fresh}(\mathsf{RA}, \mathsf{sid}) \in \{0, 1\}$ enforcing per-$(\mathsf{sid}, \mathsf{eid})$ *uniqueness* and *strict counter monotonicity* $0 \to 1 \to 2$. The verifier accepts at most one quote for each $(\mathsf{sid}, \mathsf{eid}, \mathsf{ctr})$, and any $\mathsf{ctr}{=}2$ (DEL) must be fresh relative to previously accepted $\mathsf{ctr} \in \{0, 1\}$. No trusted time is required [1]; realizations include enclave-backed monotonic storage or an external append-only anchor.

2.4 Adversary and network

The network is asynchronous with eventual delivery; the adversary may delay, drop, reorder, and duplicate messages. Channels are authenticated and confidential (RA–TLS). The adversary is rushing and may statically corrupt up to f enclaves and their hosts. We target $t{=}n$ for strongest one-shot semantics; extensions to $t < n$ are discussed in §6.

3 QuanTEEum Protocol

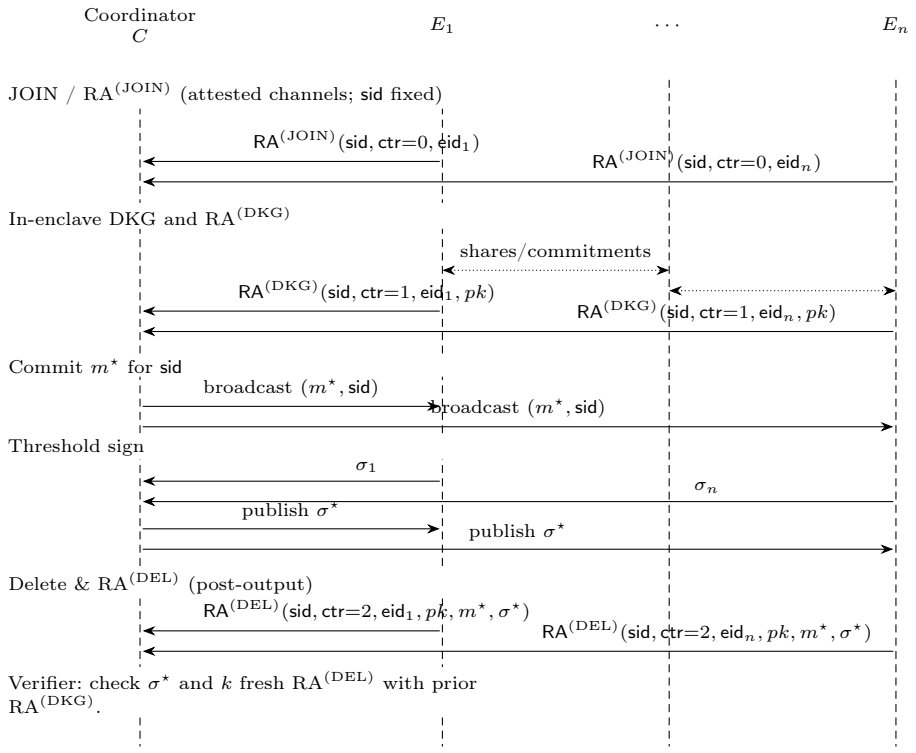

Fig. 1. QuanTEEum message flow: $\mathrm{RA}^{(\mathrm{JOIN})}$, $\mathrm{RA}^{(\mathrm{DKG})}$ (bind pk), Commit $m^\star$, Sign, $\mathrm{RA}^{(\mathrm{DEL})}$ (deletion).

Roles. Parties $\{P_1, \ldots, P_n\}$ run enclaves $\{E_1, \ldots, E_n\}$. A (rotating) coordinator, assumed to be honest and available for protocol liveness, C orchestrates rounds but learns no secrets. All protocol traffic (JOIN, DKG, Commit, Sign, Delete) is carried over *attested secure channels* (e.g., RA–TLS) whose channel binding includes an ephemeral enclave public key eid attested by the enclave.

Session binding. We use DKG-first. A session identifier sid is derived and embedded in all protocol messages and in the attestation *reportdata*

$$\mathsf{sid} \leftarrow H(\mathsf{suite} \,\|\, \mathsf{policy\_hash} \,\|\, \mathsf{nonce}),$$

Parameters.

– suite: cryptographic suite identifier (curve/group, hash, domain separators, transcript encoding, threshold-signing variant).

- policy: canonical verifier policy (e.g., n, t, minimum DEL attestations k, vendor/diversity constraints, RA roots, allowed measurements, freshness details).
- $\mathsf{policy\_hash} := H(\mathsf{policy})$ (committed by sid).
- nonce: 128-bit coordinator nonce ensuring sid uniqueness.
- eid: enclave's attested ephemeral DH/public key (per session).

3.1 Setup and DKG

Algorithm 1 performs policy-based roster selection, attested JOIN with peer vetting, runs the in-enclave threshold DKG coordinated by C, and binds the group key via $RA^{(\mathrm{DKG})}$.

Algorithm 1 QuanTEEum: *SetupAndDKG*

1: C selects a roster R of enclaves satisfying policy and sends $(\mathsf{suite}, \mathsf{policy}, \mathsf{nonce}, \mathsf{sid})$ to all $E_i \in R$.
2: **for** each enclave $E_i \in R$ **do**
3: Measure code meas; generate eid_i; establish RA–TLS channel bound to eid_i.
4: $\mathsf{RA}_i^{(\mathrm{JOIN})} \leftarrow \mathsf{RA.GenQuote}\big((\mathsf{sid}, \mathsf{ctr}{=}0, \mathsf{eid}_i)\big)$; send to C.
5: C distributes $\{\mathsf{RA}_j^{(\mathrm{JOIN})}\}_{j \in R}$ to all $E_i \in R$.
6: **for** each enclave $E_i \in R$ **in parallel do**
7: For each $\mathsf{RA}_j^{(\mathrm{JOIN})}$: run $\mathsf{RA.Verify}$; assert same sid and that the *set* of participants satisfies policy; abort on failure.
8: Participate (over RA–TLS) in the in-enclave *threshold DKG* coordinated by C to derive local share sk_i and common group pk; **never** export sk_i.
9: $\mathsf{RA}_i^{(\mathrm{DKG})} \leftarrow \mathsf{RA.GenQuote}\big((\mathsf{sid}, \mathsf{ctr}{=}1, \mathsf{eid}_i, pk)\big)$; send to C.
10: C verifies all $\mathsf{RA}_i^{(\mathrm{DKG})}$ and that they bind to the same (sid, pk); abort otherwise.
11: **return** pk (to the application/coordinator).

3.2 Single-signing phase

Algorithm 2 commits the designated message, latches it one-shot in each enclave, and performs threshold signing with C aggregating.

Algorithm 2 QuanTEEum: *SingleSign* on designated message $m^\star$

1: C broadcasts $(m^\star, \mathsf{sid})$ over the RA–TLS channel (using the already-established sid).
2: **for** each enclave E_i **in parallel do**
3: Check sid equality with local session; abort if mismatched or already latched.
4: $\mathsf{Latch}(\mathsf{sid}, m^\star)$ to bind the one-shot target (further signing under the same sid is refused).
5: Run partial signing on $m^\star$ to obtain σ_i; send σ_i to C over RA–TLS.
6: C aggregates $\{\sigma_i\}$ into $\sigma^\star$; **proceed to *DeleteAndPublish*** (Alg. 3).

3.3 Certified deletion and publication

Algorithm 3 has C request deletion attestations (including $\sigma^\star$) and enclaves attest deletion; C then publishes the attested one-shot certificate $\mathsf{AOSC_{sid}}$.

$$\mathsf{AOSC_{sid}} := \big(\mathsf{sid},\ \mathsf{suite},\ \mathsf{policy\_hash},\ \mathsf{nonce},\ pk,\ m^\star,\ \sigma^\star,\ \{(\mathsf{eid}_i,\ \mathsf{RA}_i^{(\mathrm{DKG})},\ \mathsf{RA}_i^{(\mathrm{DEL})})\}_{i\in S'}\big),$$

where S' is an index set with $|S'| \geq k$ satisfying the policy's diversity constraints.

Algorithm 3 QuanTEEum: *DeleteAndPublish*

C broadcasts over RA–TLS a deletion request containing $(\mathsf{sid}, pk, m^\star, \sigma^\star)$ to the session roster.
for each enclave E_i **in parallel do**
    Receive $(\mathsf{sid}, pk, m^\star, \sigma^\star)$ over RA–TLS; check sid matches local session (abort on mismatch).
    $\mathsf{Del}()$: zeroize local key material and nonces.
    $\mathsf{RA}_i^{(\mathrm{DEL})} \leftarrow \mathsf{RA.GenQuote}\big((\mathsf{sid}, \mathsf{ctr}{=}2, \mathsf{eid}_i, pk, m^\star, \sigma^\star)\big)$.
    Send $\mathsf{RA}_i^{(\mathrm{DEL})}$ to C over RA–TLS.
C initializes $S \leftarrow \emptyset$.
for each enclave E_i **do**
    C receives $\mathsf{RA}_i^{(\mathrm{DEL})}$; add $(\mathsf{eid}_i, \mathsf{RA}_i^{(\mathrm{DEL})})$ to S.
C chooses a policy-compliant subset $S' \subseteq S$ with $|S'| \geq k$.
C **publishes** $\mathsf{AOSC_{sid}} = (\mathsf{sid}, \mathsf{suite}, \mathsf{policy\_hash}, \mathsf{nonce}, pk, m^\star, \sigma^\star, \{(\mathsf{eid}_i, \mathsf{RA}_i^{(\mathrm{DKG})}, \mathsf{RA}_i^{(\mathrm{DEL})})\}_{i\in S'})$.

3.4 VerifyAOSC

Algorithm 4 validates the published certificate against policy binding, signature correctness, and per-enclave RA freshness.

Algorithm 4 QuanTEEum: *VerifyAOSC*($\mathsf{AOSC_{sid}}$)

Recompute $\mathsf{sid}' \leftarrow H(\mathsf{suite} \,\|\, \mathsf{policy\_hash} \,\|\, \mathsf{nonce})$ and assert $\mathsf{sid}' = \mathsf{sid}$
Assert $\mathsf{policy\_hash}$ matches the verifier's configured policy; check $|S'| \geq k$ and diversity constraints on $\{\mathsf{eid}_i\}_{i\in S'}$
Check $\mathsf{Verify}(pk, m^\star, \sigma^\star) = 1$
for each $i \in S'$ **do**
    (a) $\mathsf{RA.Verify}(\mathsf{RA}_i^{(\mathrm{DKG})};\ \mathsf{allow\_meas},\ \text{RA roots}) = 1$; parse aux and assert $(\mathsf{sid}, \mathsf{ctr}{=}1, \mathsf{eid}_i, pk)$; ensure $\mathsf{Fresh}(\mathsf{RA}_i^{(\mathrm{DKG})}, \mathsf{sid}) = 1$.
    (b) $\mathsf{RA.Verify}(\mathsf{RA}_i^{(\mathrm{DEL})};\ \mathsf{allow\_meas},\ \text{RA roots}) = 1$; parse aux and assert $(\mathsf{sid}, \mathsf{ctr}{=}2, \mathsf{eid}_i, pk, m^\star, \sigma^\star)$; ensure $\mathsf{Fresh}(\mathsf{RA}_i^{(\mathrm{DEL})}, \mathsf{sid}) = 1$.
    (c) Assert that both quotes share the same $(\mathsf{sid}, \mathsf{eid}_i, pk)$ and the counters respect $1 \to 2$.
Accept iff all checks pass

Note. The above acceptance rule enforces the exact-one semantics; see Theorem 1.

4 Security

We sketch the core arguments; full proofs follow standard game-hopping with an idealized Fresh predicate.

Safety: one-shot unforgeability.

Theorem 1 (One-shot from threshold-in-TEE). *Assume (i) the underlying threshold signature is unforgeable, (ii) at least one enclave is honest and transitions to post-delete state after producing $\sigma^\star$, (iii) Fresh prevents replay/rollback of pre-delete states, and (iv) vendor RA is unforgeable under the stated trust roots. Then producing two distinct accepting artifacts for the same pk in session sid—i.e., two AOSCs that pass VerifyAOSC—with $\sigma_1 \neq \sigma_2$ is infeasible for PPT adversaries.*

Proof (Proof sketch). Suppose two accepting artifacts exist. If either signature is produced without t valid shares, we break threshold unforgeability. Otherwise, at least t shares were live at each signing. Any accepting certificate contains at least k valid deletion attestations, and under the honest-minority assumption at least one of those enclaves is honest and has deleted its share bound to $(\mathsf{sid}, m^\star, \sigma^\star)$. The only routes to a second accepted signature are: (A) forge RA to claim deletion without deleting (contradicts RA unforgeability); (B) roll back an honest enclave past delete (contradicts Fresh); or (C) compromise enough enclaves (meeting policy constraints) to reassemble t fresh shares (violates honest-minority).

Liveness. With $t{=}n$ the protocol requires all enclaves to participate; a crashed or aborting enclave prevents progress. We adopt a bounded timeout: if signing fails, C abandons sid and re-instantiates (fresh sid and DKG) with fresh enclaves. In deployments desiring partial progress, QuanTEEum supports $t < n$ with two caveats: (i) *deletion* attestations must reach $k \geq t$; (ii) the exact-one property is now conditioned on at least one of the k attesting enclaves being honest.

Additive security via heterogeneity. If attestations are independent across vendors/geographies/operators, the minimum cost-of-attack is approximately the sum of the k cheapest per-enclave compromise costs (this approximation assumes independence and no bulk-attack economies; otherwise treat it as a lower bound). We recommend explicit *diversity constraints* in policy (e.g., "at least two vendors and three physical operators").

5 Applications

5.1 QuanTEEum One-shot signatures (OSS)

- **KeyGen ($\mathsf{Gen}(\mathsf{crs}) \to (pk, sk)$).** Instantiate a fresh sid; run *SetupAndDKG* (Alg. 1) to obtain pk. The secret key sk is an opaque handle to the distributed in-enclave state for sid (shares never leave TEEs).
- **Sign ($\mathsf{Sign}(sk, m) \to \sigma$).** Run *SingleSign* (Alg. 2) on m to obtain $\sigma^\star$, then *DeleteAndPublish* (Alg. 3) to get $\mathsf{AOSC}_{\mathsf{sid}}$. Output $\sigma := (\sigma^\star, \mathsf{AOSC}_{\mathsf{sid}})$.

– **Verify** ($\mathsf{Ver}(\mathsf{crs}, pk, m, \sigma) \to b$)**.** Parse $\sigma = (\sigma^\star, \mathsf{AOSC}_{\mathsf{sid}})$; accept ($b$=1) iff $\mathsf{Verify}(pk, m, \sigma^\star) = 1$ and *VerifyAOSC* (Alg. 4) accepts $\mathsf{AOSC}_{\mathsf{sid}}$.

Note (certificate canonicality). Our one-shot guarantee concerns signatures, not certificate canonicality. To prevent multiple valid artifacts for the same $\sigma^\star$ from enabling duplicate claims, deployments must enforce a canonicality rule at the application/contract layer.

5.2 QuanTEEum money

Following the OSS-based quantum money with *classical* communication paradigm [8], a coin is a chain of one-shot signatures over successive recipients' serials/keys: the mint signs Alice's serial; on transfer, Alice one-shot signs Bob's, and so on, so the previous signing capability self-destructs after each hop. QuanTEEum instantiates this with TEEs: each minting/transfer runs a fresh session to produce $(pk, \sigma^\star, \mathsf{AOSC}_{\mathsf{sid}})$ where $m^\star$ encodes the coin serial (and, if desired, the next recipient), and $\mathrm{RA}^{(\mathrm{DEL})}$ certifies in-enclave key erasure. The "one-shot" effect we rely on is precisely that the signing key can be used only once.

6 Discussion and Future Work

Why $t=n$ first? It maximizes the exact-one semantics and simplifies acceptance (*all* attest). Generalizing to $t < n$ is straightforward mechanically but shifts the trust statement to "among the $k \geq t$ attesters, at least one honest enclave deleted."

Cryptoeconomic attestations. Augment RA with bonded claims: each operator escrows collateral (and potentially zk-collateral) and signs a deletion statement under a registered key; slashing conditions penalize equivocation or reuse.

Upgrade to quantum OSS. Once practical, it should be possible to swap the attestation mechanism for a quantum-certified deletion proof with the same acceptance predicate. Because QuanTEEum already binds $(\mathsf{sid}, m^\star, \sigma^\star, pk)$ into attestations, the application layer remains unchanged.

On-chain instantiation and incentives. A blockchain can realize Fresh by recording JOIN/DEL quotes on-chain (keyed by $(\mathsf{sid}, \mathsf{eid}, \mathsf{ctr})$), letting verifiers enforce uniqueness and $0 \to 1 \to 2$ order from state, while fees/rewards and slashing incentivize honest participation.

7 Conclusion

QuanTEEum offers a deployable path to one-shot capabilities by composing threshold signatures with certified deletion inside heterogeneous TEEs. The exact-one property becomes an honest-minority statement with additive security across independent attestations. The same interface smoothly upgrades to future quantum OSS, enabling *QuanTEEum OSS* and *QuanTEEum money*, while preserving a clean upgrade path as TEE attestations and quantum back ends strengthen.

References

1. Alder, F., Scopelliti, G., Van Bulck, J., Mühlberg, J.T.: About time: On the challenges of temporal guarantees in untrusted environments. In: Proceedings of the 6th Workshop on System Software for Trusted Execution. pp. 27–33 (2023)
2. Amos, R., Georgiou, M., Kiayias, A., Zhandry, M.: One-shot signatures and applications to hybrid quantum/classical authentication. In: Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing. pp. 255–268 (2020)
3. Drake, J.: One-shot signatures. PROGCRYPTO (YouTube video), `https://www.youtube.com/watch?v=VmqkH3NPG_s`, talk
4. Groth, J.: On the size of pairing-based non-interactive arguments. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 305–326. Springer (2016)
5. Kiayias, A.: One-shot signatures: Applications and design directions (2024), invited talk
6. Komlo, C., Goldberg, I.: Frost: Flexible round-optimized schnorr threshold signatures. In: International Conference on Selected Areas in Cryptography. pp. 34–65. Springer (2020)
7. Sattath, O.: Quantum prudent contracts with applications to bitcoin. arXiv preprint arXiv:2204.12806 (2022)
8. Shmueli, O., Zhandry, M.: On one-shot signatures, quantum vs. classical binding, and obfuscating permutations. In: Annual International Cryptology Conference. pp. 350–383. Springer (2025)
9. Shor, P.W., Farhi, E., Gosset, D., Hassidim, A., Lutomirski, A.: Quantum Money, vol. 19. MIT. January (2012)