Chair of Network Architectures and Services
School of Computation, Information and Technology
Technical University of Munich

Idea 1

Hybrid Batched Threshold Encryption with TEEs (HbTPKE- TEE)

Ivan von Greiff

M.Sc Electrical and Information Engineering

1 Motivation and Inspiration

The problem of *mempool privacy* has gained urgency with the rise of DeFi, where adversaries exploit transaction visibility to frontrun, backrun, or censor users, leading to massive "Miner Extractable Value" (MEV) losses [1]. Existing threshold encryption approaches to protect the mempool (e.g., Shutter Network) either fall to malleability attacks or sacrifice *pending transaction privacy*, leaking the content of transactions that never make it on- chain. To address these flaws, *Mempool Privacy via Batched Threshold Encryption (bTPKE)* c.f. Link, introduced a new primitive, **batched-threshold encryption**, which achieves CCA-style non-malleability under ROM+AGM assumptions while achieving $\mathcal{O}(n)$ decryption communication independent of batch size. This construction represents the first "concretely efficient construction" for their new notion (FbTPKE, the instantiation of bTPKE) designed to solve the mempool privacy problem, but comes at a cost:

- A prohibitively heavy *one-time Setup* phase requiring either a trusted dealer or an expensive one-time distributed key generation (DKG) plus a power-of-τ common reference string (CRS) ceremony
- A decryption path dominated by simulation-extractable NIZKs and pairing checks (over 99% of runtime for realistic block sizes).

Importantly, while the authors dismiss *TEE-only* designs that entrust enclaves with long-term decryption keys, they do not explore the possibility of TEEs as *auxiliary accelerators* in non-critical paths. This motivates our proposed **hybrid extension** of their ideal functionality—which we term *HbTPKE-TEE*—that preserves FbTPKE's strong cryptographic guarantees while allowing optional, attested TEE oracles to offload exactly the two identified bottlenecks:

1. An *AttestedDealer* enclave that implements `Setup`, replacing the trusted dealer/MPC without touching long-term keys
2. An *AttestedIngress* enclave that certifies ciphertext well- formedness and non-malleability at the network edge, replacing validators' heavy SE-NIZK verification on the critical path.

This hybrid design maintains graceful fallback to pure cryptography when TEEs are distrusted, but enables practical deployments by sharply reducing both setup costs and validator overhead.

2 Problem Statement

The work of Choudhuri et al. established batched-threshold encryption (bTPKE) and gave the first concretely efficient construction (FbTPKE) that simultaneously achieves:

- Mempool privacy with CCA-style security (in ROM+AGM with straight-line extractability)
- Pending-transaction privacy
- $\mathcal{O}(n)$ decryption communication independent of batch size (while the combiner's work remains $\mathcal{O}(B \log B)$)

However, their construction inherits the two aforementioned major bottlenecks (c.f. Section 1) that hinder deployment in practice. We therefore pose the following problem: *How can one preserve the privacy and efficiency guarantees of FbTPKE for encrypted mempools, while*

i) achieving a practical and auditable *one-time Setup*, and

ii) removing per-transaction SE-NIZK verification from the validator's critical path,

without ever entrusting TEEs with decryption capabilities or long-term secrets?

As we discuss in Section 10, these bottlenecks become particularly acute in real-world blockchain settings (Ethereum, rollups, Cosmos/Tendermint), where block times and validator workloads impose strict performance constraints.

3 Proposed Contribution

We propose **HbTPKE-TEE**, a hybrid extension of the FbTPKE execution model that introduces two *optional* attested oracles. These oracles are designed to offload exactly the two bottlenecks identified in FbTPKE—the expensive `Setup` phase and the validator CPU cost of SE-NIZK verification—without entrusting TEEs with decryption capabilities or long-term secrets. If remote attestation fails or enclaves are distrusted, the system *gracefully falls back* to the baseline BTE++ protocol (pure cryptographic path with one-time setup), preserving all of the scheme's privacy and efficiency guarantees.

AttestedDealer (`Setup`).

The AttestedDealer enclave is a real-world instantiation of the trusted dealer assumed in FbTPKE. Its role is limited to a one-time setup phase, and it produces exactly the outputs expected by the cryptographic construction:

- publish $\mathtt{crs1} = (g, g^{\tau}, g^{\tau^2}, \dots, g^{\tau^B}, h, h^{\tau})$ and public commitments,
- privately deliver to each server i its secret shares $\{[L_j(\gamma)]_i\}$ (with PVSS commitments),
- emit a remote-attestation quote binding all outputs to the enclave's code hash (MRENCLAVE), and
- record $(\mathtt{eid}, \mathtt{crs1}, \mathtt{com}, \mathtt{att})$ to an append-only audit log.

Verifiability hooks allow the committee to cheaply check *consistency* of `crs1` via pairing equations, but to avoid a single point of trust the system either (a) co-generates τ across multiple enclaves or (b) falls back to the MPC ceremony described in the paper. If the enclave is unavailable or its attestation is revoked, the system transparently reverts to the MPC emulation.

Why this matters. The paper highlights that `Setup` is $\approx 100\times$ more expensive than prior work, but "acceptable" because decryption lies on the critical path. Our AttestedDealer makes setup *practically cheap* and auditable, lowering the barrier to real-world deployment while keeping decryption unchanged and FbTPKE's security intact. Long-term trapdoors are never retained by the enclave (they are split, shared, and zeroized).

AttestedIngress (Ciphertext Well-Formedness).

Edge ingress enclaves enforce the same non-malleability and well-formedness checks that the baseline cryptographic scheme requires via NIZK proofs. Instead of producing a heavy proof, they return a compact attestation signature

$$\sigma_{\mathsf{ing}} = \mathsf{Sign}_{\mathsf{TEE}}\big(H(\hat{x} \,\|\, S \,\|\, ct^{(1)} \,\|\, ct^{(2)} \,\|\, ct^{(3)} \,\|\, \mathtt{eid} \,\|\, \mathtt{ts} \,\|\, \mathtt{nonce})\big),$$

bound to all ciphertext components $(\hat{x}, S, ct^{(1)}, ct^{(2)}, ct^{(3)}, \mathtt{eid})$. Validators then accept ciphertexts under the following policy:

(1) $(ct, \sigma_{\mathsf{ing}}, \mathsf{RA})$ from any whitelisted attested ingress enclave, or

(2) $(ct, \pi_{\mathsf{SE\text{-}NIZK}})$ verified in the standard cryptographic path.

To prevent copy and replay attacks, enclaves (or a shared filter) maintain per-epoch Bloom filters of signed ciphertexts, refusing to re-sign variants of the same $(\hat{x}, H(S))$. This directly addresses the "copy-attack" the paper warns about.

Validator fast-path. In this model, validators verify only a lightweight TEE signature plus remote- attestation evidence per transaction, rather than running full SE-NIZK and pairing checks. The heavy cryptography is offloaded to horizontally scalable ingress boxes, removing the $> 99\%$ CPU hotspot identified in the paper (e.g., $\sim$ 3.2s per validator at 500 tx).

Security lens. Under the honest-TEE assumption, the system enforces the same well-formedness and non-malleability policy that SE-NIZKs provide. If TEEs are distrusted or attestation fails, nodes require SE-NIZKs, restoring the scheme's full cryptographic guarantees.

Together, the AttestedDealer and AttestedIngress provide a practical hybrid execution model: they preserve FbTPKE's strong privacy and $\mathcal{O}(n)$ communication guarantees, while making setup lightweight and shifting validator-critical verification off the hot path. HbTPKE-TEE thus offers a deployable balance between pure cryptography and trusted hardware, with *auditable TEEs as accelerators, not trust anchors.*

4 Model, Assumptions, and Goals

System Model. HbTPKE-TEE follows the execution model of FbTPKE with two optional attested oracles. The *AttestedDealer* is run once during the one-time setup (or upon committee refresh) by a whitelisted enclave that generates `crs1` and secret shares, then outputs attested transcripts. The *AttestedIngress* consists of enclaves deployed at the network edge (logically stateless, except for per-epoch replay filters); clients submit ciphertexts to these enclaves, which certify well-formedness and return signatures. Validators accept transactions carrying either enclave attestations or SE-NIZKs, ensuring liveness even when TEEs are unavailable or distrusted.

Adversary. We assume a network adversary that controls message scheduling and censorship; a Byzantine *minority* of committee members; malicious clients submitting malformed ciphertexts; and attackers attempting TEE compromise or remote- attestation (RA) forgery. Compromise of the RA root of trust (e.g., Intel/AMD root keys) is out of scope.

Assumptions. We inherit the BTE++ cryptographic assumptions (e.g., hardness underlying KZG and the SE-NIZKs in the ROM+AGM setting with straight-line extractability). We further assume sound remote attestation and that validators pin enclave measurements to an allowlist. TEEs never hold decryption keys or long-term secrets; any enclave state for setup is split/shared and then zeroized, so compromise cannot retroactively break ciphertext privacy.

Leakage Function $\mathcal{L}$. We model the information exposed by attested oracles via a leakage function $\mathcal{L}$, consistent with oracle-augmented cryptographic definitions:

- *AttestedDealer:* learns the randomness used to derive the CRS and secret shares (identical to a trusted/MPC dealer).
- *AttestedIngress:* learns ciphertext structure/metadata sufficient to check the well-formedness relation (the same information any SE-NIZK verifier sees); it does not learn plaintexts or decryption keys.
- No additional leakage about non-decrypted ciphertexts across batching windows (epochs/blocks) is introduced.

Security Objectives.

(1) **Privacy:** preserve pending-transaction privacy and *CCA- style* non-malleability in the same model as FbTPKE (ROM+AGM with straight-line extractability).

(2) **Dealer indistinguishability:** under sound RA and correct enclave code, outputs from *AttestedDealer* are indistinguishable from those of an honest (trusted/MPC) dealer.

(3) **Soundness-by-attestation:** accepting a σ_{ing} attestation is *policy-equivalent* to running the SE-NIZK verifier on the same relation, provided RA is sound and the ingress enclave enforces the specified checks.

(4) **Graceful degradation:** revocation or rejection of RA forces fallback to the pure-cryptographic path (SE-NIZKs + MPC dealer) with no loss of FbTPKE's security guarantees.

Non-goals. We do not attempt to reduce committee size n, protect against majority adversaries, or mitigate all TEE side-channels. Our goal is to demonstrate that TEEs can serve as auditable accelerators for setup and ingress verification without becoming trust anchors.

5 Potential Research Questions

RQ 1 **Setup Efficiency.** To what extent can AttestedDealer reduce the wall-clock time and operational complexity of `Setup` compared to MPC emulation, while maintaining indistinguishability from honest dealer outputs under sound RA?

RQ 2 **Validator Performance.** How much validator CPU and end-to-end block latency can be saved by replacing per-transaction SE-NIZK verification with AttestedIngress attestations, particularly at realistic block sizes $B \in \{128, 512\}$?

RQ 3 **Security Equivalence.** Under which precise assumptions does acceptance of a TEE attestation σ_{ing} provide the same non- malleability and CCA-style guarantees in the same model as FbTPKE (ROM+AGM with straight-line extractability)?

RQ 4 **Robustness and Fallback.** How reliably and with what latency overhead can HbTPKE-TEE revert to the pure-cryptographic path (MPC + SE-NIZKs) upon RA failure or revocation, while preserving liveness and privacy?

RQ 5 **Deployability.** Does the hybrid design lower the barrier to adoption compared to pure-cryptographic BTE++, and what are the trade-offs between operational trust in TEEs and cryptographic guarantees?

6 Approach

Our approach combines empirical benchmarking with prototype enclave implementations, structured into clear work packages that map directly to the research questions.

A. Baseline Reproduction

We first reproduce the results of Choudhuri et al. to establish a trustworthy baseline. This includes verifying ciphertext and share sizes (∼466 B ciphertexts for 32 B messages, ∼48 B partial decryptions), encryption and decryption timings (~ 8.5 ms encryption, ~ 3.2 s partial decryption for ∼500 transactions, ~ 3.0 s reconstruction at $B{=}512$), and confirming that $> 99\%$ of BatchDec cost arises from SE-NIZK verification and pairing checks. This step ensures our evaluation of HbTPKE-TEE is grounded against the published reference implementation.

B. Work Packages

WP1: AttestedDealer (RQ1). Implement the dealer logic inside a TEE. The enclave generates $\texttt{crs1}$, distributes per-party shares of Lagrange coefficients with PVSS commitments, and emits a remote attestation quote. An append-only audit log records $(\texttt{eid}, \texttt{crs1}, \texttt{com}, \texttt{att})$. We add public consistency checks (e.g., pairings) to allow validators to detect mis-generated $\texttt{crs1}$ out-of-enclave. *Deliverables:* empirical comparison of setup time and complexity between AttestedDealer and MPC emulation; evidence of indistinguishability of outputs under sound RA.

WP2: AttestedIngress (RQ2, RQ3). Build RA-TLS ingress enclaves that validate ciphertext well-formedness and non- malleability relations (as enforced in FbTPKE via SE-NIZKs). Instead of proofs, they output compact attestations $\sigma_{\text{ing}} = \text{Sign}_{\text{TEE}}(H(ct \,\|\, \texttt{eid} \,\|\, \texttt{ts} \,\|\, \texttt{nonce}))$, bound to all ciphertext components. Ingress enclaves maintain per-epoch Bloom filters to prevent re-signing of variants (anti-copy/replay). We begin with Ed25519/Schnorr signatures, with optional evaluation of threshold aggregation (FROST) for scalability. *Deliverables:* benchmark of validator CPU savings at $B \in \{128, 512\}$; formal argument that acceptance of σ_{ing} is *policy- equivalent* to SE-NIZK verification under sound RA.

WP3: Node Policy and Fallback (RQ4). Modify node acceptance policy so that validators accept either $(ct, \sigma_{\text{ing}})$ from whitelisted enclaves or $(ct, \pi_{\text{SE-NIZK}})$ verified cryptographically. If RA fails or is revoked, nodes immediately enforce SE-NIZK-only mode, preserving liveness and privacy guarantees. *Deliverables:* evaluation of fallback latency and robustness under simulated revocation events.

7 Evaluation Plan

To assess whether HbTPKE-TEE achieves its intended benefits without weakening FbTPKE's guarantees, we will evaluate along three dimensions: efficiency, security equivalence, and robustness.

Metrics.

- *Efficiency:* Setup wall-clock time and bytes; validator CPU during BatchDec; end-to-end block decryption latency; throughput (tx/s).
- *Overheads:* Attestation verification cost; network bytes and gossip-delay sensitivity (noting that additional KB beyond ~20 kB may impact propagation delay).
- *Robustness:* Recovery latency when falling back to SE-NIZK-only mode upon RA failure or revocation.

Experimental Environments.

- Microbenchmarks of enclave vs. cryptographic components.
- WAN emulation (e.g., 200 ms delay, 20 ms jitter, 10 Mbit/s).
- Committee sizes $n \in \{16, 64, 128\}$.
- Batch sizes $B \in \{128, 512\}$.

Baselines.

(1) Pure-cryptographic BTE++ (as in Choudhuri et al.).

(2) Strawman TEE-only design (keys inside enclave) for comparison only.

(3) HbTPKE-TEE with AttestedDealer and/or AttestedIngress enabled.

Hypotheses.

- **H1:** AttestedDealer reduces setup cost by a large constant factor relative to MPC emulation, while producing outputs indistinguishable from an honest dealer under sound RA.
- **H2:** AttestedIngress reduces validator CPU during BatchDec by $10\times$–$100\times$, yielding proportional reductions in block decryption latency at realistic batch sizes, with no loss of privacy or non-malleability guarantees.
- **H3:** The system degrades gracefully: revocation or failure of RA causes only short-lived performance loss, not liveness or security failure.

8 Risks and Mitigations

TEE Compromise. HbTPKE-TEE never places long-term decryption keys inside enclaves. Compromise therefore cannot expose past ciphertexts, but could bias CRS setup if undetected. We mitigate this by (i) requiring PVSS-style commitments and public checks on CRS structure, (ii) logging all enclave outputs to append-only audit trails, and (iii) reverting automatically to pure-cryptographic execution upon attestation failure.

Ingress Denial-of-Service. Ingress enclaves could become a bottleneck or DoS target. We mitigate this via rate limiting, committee-controlled allow-listing, and horizontal scaling of ingress TEEs. Fallback to SE-NIZK-only verification ensures liveness even under sustained attack.

Auditability and Accountability. All dealer outputs and ingress attestations are written to append-only logs, and committee members can cheaply validate `crs1` structure with public checks. This ensures that even if TEEs misbehave, their influence is detectable and attributable.

9 Expected Contributions

- **Architecture:** HbTPKE-TEE design with AttestedDealer and AttestedIngress oracles, including precise system model and formalized security goals. (addresses RQ1–RQ4)
- **Theory:** Proof sketches showing (i) indistinguishability of AttestedDealer outputs from MPC/trusted dealer, and (ii) *policy- equivalence* of ingress attestations to SE-NIZK verification under sound RA. (addresses RQ3)
- **Implementation:** Prototype enclaves, node integration, and microbenchmarks demonstrating concrete setup and decryption-time improvements at Ethereum-scale batch sizes. (addresses RQ1–RQ2)
- **Evaluation:** End-to-end performance results across WAN emulation, varying committee/batch sizes, and fallback stress tests. (addresses RQ2–RQ4)
- **Deployment Policy:** A practical mempool policy that enforces clean fallback to pure cryptography on RA failure or revocation. (addresses RQ4–RQ5)
- **Practical relevance:** Analysis of deployment contexts (Ethereum, rollups, Cosmos) showing how HbTPKE-TEE optimizations map to validator workloads and MEV mitigation needs (Section 10). (addresses RQ5)

10 Deployment Contexts and TEE-Based Optimizations

While HbTPKE-TEE is motivated by abstract efficiency and security goals, its true value lies in how it can be deployed in real blockchain systems. Below we outline several representative contexts and show how our design's optimizations reduce validator workload, improve performance, and strengthen security in practice.

10.1 Ethereum and Layer-1 Context

Ethereum's public mempool is the canonical setting for MEV attacks such as front-running and sandwiching. An encrypted mempool based on HbTPKE-TEE would mitigate these issues by concealing transaction contents until ordering is fixed. The key optimization is that TEEs can *offload heavy SE-NIZK verification*: rather than every validator performing hundreds of pairing or SNARK checks per block, a TEE verifies proofs once and outputs a lightweight attestation. Validators then need only check a signature, yielding substantial CPU savings and faster block propagation. This aligns naturally with Ethereum's proposer–builder separation, where block builders could operate enclaves to decrypt after sealing, while proposers retain ordering control. Importantly, the hybrid design maintains decentralization while avoiding a pure TEE-only trust model: decryption keys remain distributed, and enclaves only accelerate setup and proof-checking.

Environment:

- Large validator set (hundreds of thousands)
- Proposer-Builder Spearation: builders assemble block, proposers choose among them
- Mempool is open/public, big MEV problem

HbTPKE-TEE role:

- `AttestedIngress` (TEE at network edge): verifies ciphertext well-formedness once, then attaches lightweight signatures instead of heavy SE- NIZKs. Validators save massive CPU time.
- `AttestedDealer`: less critical because Ethereum already has large-scale ceremonies (trusted setup/MPC) and initial setup is amortized across many validators

Key Benefit:

- Scalability and fast block propagation (avoiding 2-3s of validator CPU just for proof checks)

Focus:

- Validator's workload in a very large, decentralized network

10.2 Layer-2 Rollups and Sequencers

Rollups such as Optimism, Arbitrum, or zkSync rely on centralized or small- committee sequencers that are natural points for MEV. In this context, a TEE- backed sequencer could:

1. Accept encrypted transactions
2. Attest to validity
3. Commit to block ordering before revealing plaintexts

This prevents even a centralized operator from exploiting transaction data. Performance is also improved: the enclave batches proof verifications and produces a single attestation, ensuring throughput is not bottlenecked by cryptographic checks. For zk-rollups, enclave attestations can be fed into zk-circuits as trusted inputs, avoiding the need to embed costly cryptographic relations inside the SNARK. The result is MEV-resistant ordering with negligible latency overhead.

Environment:

- Centralized or small-committee sequencers order transactions
- MEV risk is especially high, since a single sequencer can reorder everything
- Block times are short; high throughput is expected

HbTPKE-TEE role:

- `AttestedIngress`: runs at the sequencer front-end → transactions arrive encrypted, are attested by TEE (valid ciphertexts), but not decrypted until ordering is fixed
- `AttestedDealer`: lightweight one-time setup, but because sequencers are few/centralized, this is less of a bottleneck
- For zk-rollups, enclave attestations could even be inputs into zk- circuits (rather than all the heavy SE-NIZK verification logic), avoiding embedding expensive crypto checks inside the SNARK

Key Benefit:

- Prevents sequencers from abusing privileged visibility (they do not see plaintext until block order is sealed)

Focus:

- Making centralized sequencers provably unable to frontrun/censor for MEV
- In case of zk-rollups: efficiency of zk-proofs + MEV resistance

10.3 Cosmos/Tendermint Chains

BFT-style PoS chains with 50–150 validators face distinct challenges:

1. Distributed key generation (DKG) protocols are communication-heavy and fragile
2. Per-transaction proof verification is too slow for short block times

HbTPKE-TEE addresses both. A TEE-based dealer can perform DKG once (initially) and distribute shares with remote attestation, reducing a complex multi-round MPC to a one-shot operation. On the critical path, validator enclaves can pre-verify ciphertexts and output batch attestations, so other validators only check signatures instead of hundreds of SE-NIZKs. This can reduce verification cost by orders of magnitude (e.g., from $\sim$0.5s per 100 transactions to < 1ms). Cosmos DeFi applications such as Osmosis would benefit from encrypted mempools that resist front-running, with less operational burden than pure cryptographic schemes like Ferveo. Unlike TEE-only approaches (e.g., Secret Network), HbTPKE- TEE still distributes decryption trust, combining efficiency with decentralization.

Environment:

- Small-to-medium validator committees (50-150)
- Tendermint consensus requires multiple rounds of signed votes → communication cost is already high
- Block times are short (~ 6s), validators cannot spend seconds per block verifying hundreds of heavy SE-NIZKs
- DKG ceremonites for threshold cryptography are very communication- heavy

HbTPKE-TEE role:

- `AttestedIngress`: pre-verifies ciphertexts and outputs batch attestations → validators only check signatures, not proofs. Essential for keeping block latency low
- `AttestedDealer`: *very important here* → replaces complex multi-round MPC/DKG with a one-shot TEE-based dealer (attested setup). Huge simplification.

Key Benefit:

- Lightens both setup and critical path costs in small committees
- Makes encrypted mempools feasible in BFT PoS systems where validator resources and block times are tight

Focus:

- Operational simplicity for short block-time performance

10.4 Summary

- **Ethereum:** Validator CPU for SE-NIZK dominates, so `AttestedIngress` is crucial
- **Rollups:** Sequencer fairness dominates, so `AttestedIngress` is crucial
- **Cosmos/Tendermint:** Setup/DKG cost dominates, so `AttestedDealer` is crucial