TLO: Topology-Lattice Obfuscation for Smart Contracts

Anonymous Submission

Abstract

We present TLO (Topology-Lattice Obfuscation), a practical on-chain compiler that produces publicly evaluable locked circuits whose internal gate-control representation is LWE-hidden and whose topology is engineered to defeat a defined suite of structural reverse-engineering attacks. Security derives from a two-layer defense: a topology layer using structural mixing that empirically defeats structural and statistical attacks, and an LWE layer that hides the gate-control representation via on-chain inner products.

The LWE layer provides representation hiding, not semantic security; unlocking security reduces to secret search in the input space (and/or breaking LWE). Attackers *can* simulate evaluation offline for any input x by computing $s(x) = H(x)$. Our security claims are heuristic and relative to our defined attack evaluation matrix; we achieve 6/6 resistance within that framework at ∼2.58M gas (8.6% of block limit). We give uniform-secret LWE security estimates (∼49-bit classical with n=64; see §5.3), providing post-quantum *representation* hiding (assuming LWE quantum-resistance). Target applications include predicates with eventually-expiring secrets (honeypots, sealed-bid auctions, lotteries).

1 Introduction

Smart contracts are fully transparent. Anyone can read bytecode, analyze logic, and exploit vulnerabilities. This conflicts with applications requiring hidden logic: cryptographic honeypots, MEV-resistant execution, sealed-bid auctions, and private liquidation thresholds.

Indistinguishability obfuscation (iO) provides strong security but requires impractical overhead ($10^6\times$). We take a different approach: *heuristic obfuscation* in the tradition of local mixing [1], providing representation hiding and attack-surface hardening through two complementary layers:

Two-Layer Security Model:

1. **Layer 1 (Topology):** Structural mixing empirically defeats structural/statistical attacks. *Security: heuristic.*
2. **Layer 2 (LWE):** On-chain inner products hide gate-control representation. *Security: computational (representation hiding under ∼49-bit uniform-secret LWE).*

Key property: Unlocking reduces to secret search, not to preventing offline evaluation.

Terminology note: We use "obfuscation" informally, in the tradition of code obfuscation and heuristic obfuscation (e.g., Canetti et al. [1]), to denote representation hiding and attack-surface hardening. We do *not* claim indistinguishability obfuscation, virtual-black-box security, or any standard strong obfuscation definition.

1.1 Contributions

1. **TLO Framework:** Two-layer heuristic obfuscation combining topology mixing with on-chain LWE inner products (∼2.58M gas for n=64).
2. **Structural Mixing:** Wire selection defeating structural/statistical attacks (heuristic, empirically validated).
3. **Wrong-Key-Garbage:** Attackers *can* simulate with arbitrary keys, but incorrect keys yield garbage. This does not improve brute-force search over the secret input space; it hides direct access to the gate-control representation.
4. **On-Chain LWE Representation Hiding:** Control functions hidden via LWE ciphertexts with full inner product computation.
5. **Post-Quantum Representation Hiding:** ∼49-bit classical / ∼45-bit quantum estimate via uniform-secret LWE (n=64); ∼81-bit for n=128; see §5.3.

1.2 Scope

We claim: Representation hiding based on LWE hardness; topology heuristics empirically validated;

6/6 attack resistance *within our defined matrix*; post-quantum representation hiding.

We do NOT claim: iO security; VBB security; semantic hiding beyond black-box behavior; prevention of offline evaluation.

2 Preliminaries

2.1 Learning With Errors

Definition 2.1 (LWE [6]). For dimension n, modulus q, and error distribution χ, the LWE problem is: given $(A, As+e \mod q)$ where $A \in \mathbb{Z}_q^{m\times n}$, $s \in \mathbb{Z}_q^n$, $e \leftarrow \chi^m$, distinguish from uniform (A, u).

LWE is believed quantum-resistant and forms the basis for post-quantum cryptography standards (ML-KEM) [2]. Our parameters (n=64, q=65521) are smaller than NIST profiles; see §5.3 for security estimates.

2.2 LWE Control Function Hiding

We hide each gate's control function via LWE ciphertexts. Each CF bit is encoded as (a, b) where $b = \langle a, s_{\text{enc}}\rangle + e + \mathsf{bit} \cdot q/2$. At encryption time, $s_{\text{enc}} = H(\mathsf{secret})$. At evaluation time, the contract derives $s(x) = H(x)$ from the candidate input—decryption succeeds iff $x = \mathsf{secret}$.

Important: Since $s(x) = H(x)$ is publicly computable, anyone can evaluate the locked predicate offline for arbitrary inputs. The LWE layer hides the *representation* (CF bits), not the *functionality*.

2.3 Reversible Circuits

Definition 2.2 (Reversible Gate). A gate $g = (a, c_1, c_2, c_f)$ operates on n wires: active wire a is XORed with $c_f(c_1, c_2)$ where $c_f : \{0,1\}^2 \to \{0,1\}$ is one of 16 control functions.

Gates are self-inverse: $g(g(s)) = s$. This enables commit-reveal protocols where the solver demonstrates knowledge without revealing the secret.

3 The Topology Layer

The topology layer is a reversible circuit mixing design that empirically defeats structural and statistical attacks through wire selection, without cryptographic primitives.

3.1 Wire Selection Algorithm

Structural mixing selects wires to defeat pattern detection:

1. **Non-pow2 distances:** Control wires at distances $d \notin \{1, 2, 4, 8, \ldots\}$ from active wire defeats butterfly/FFT pattern detection
2. **Uniform wire usage:** Prefer underused wires; defeats chi-squared statistical attacks
3. **Irregular layers:** Varying gates per layer (e.g., 30–70 for 256 wires) defeats regularity detection
4. **64+ wires:** Sufficient width defeats diagonal correlation (Pearson $r < 0.10$)

Listing 1: Wire Selection (pseudocode)

```
def select_control_wire(active, usage, target):
    # Choose non-pow2 distance
    d = random_choice(non_pow2_distances)
    candidate = (active + d) % num_wires

    # Prefer underused wires (70% prob)
    if random() < 0.7 and usage[candidate] < target:
        return candidate

    # Otherwise, find underused alternative
    return find_underused_wire(usage, target)
```

3.2 Topology Attack Resistance

Attack	Type	Defense	Mechanism
Compression	Structural	Topology	No duplicate gates
PatternMatch	Structural	Topology	Random CF cycling
Structural	Structural	Topology	Non-pow2 distances
Statistical	Statistical	Topology	Uniform wire usage
DiagCorrelation	Statistical	Topology	64+ wires

Table 1: Topology empirically defeats structural/statistical attacks in our evaluation.

Key insight: Unlike butterfly or derangement topologies that only rearrange gates, structural mixing has anti-attack properties *built into wire selection*.

4 LWE for Representation Hiding

4.1 The RainbowTable Problem

RainbowTable is a *semantic* attack—it matches truth-table behavior, not structure:

1. Extract subcircuit from obfuscated circuit
2. Evaluate subcircuit on sample inputs
3. Match behavior against pre-computed lookup table

Topology cannot defeat this. Any structural transformation preserves semantic behavior of reversible circuits.

4.2 What LWE Does and Does Not Provide

Critical clarification: Attackers can compute $s(x) = H(x)$ for any input x and thus can freely simulate the contract's 1-bit output offline. Therefore, our

LWE layer does *not* prevent offline evaluation of the locked predicate.

Instead, it hides the internal gate-control representation and aims to frustrate semantic *reverse-engineering* attacks such as RainbowTable-style structural matching.

> **On-Chain Inner Product:** Control functions encoded as LWE ciphertexts (a, b). At encryption: $s_{\text{enc}} = H(\mathsf{secret})$. At evaluation: $s(x) = H(x)$.
> **What this provides:** Representation hiding—CF bits are not directly readable from the static artifact without choosing an input x and computing $H(x)$.
> **What this does NOT provide:** Prevention of offline evaluation. Unlocking difficulty is governed by secret entropy.

Proposition 4.1 (LWE-Based Representation Hiding)**.** *Assuming the hardness of* LWE*, given only the obfuscated circuit, no* PPT *adversary can recover a non-negligible fraction of the gate control-function bits* c_f *with non-negligible advantage.*

Proof sketch: Each control-function bit is encoded as an LWE ciphertext (a, b) under a secret $s_{\text{enc}} = H(\mathsf{secret})$. Recovering c_f from these ciphertexts without knowledge of s_{enc} reduces to distinguishing LWE samples from uniform. □

5 Security Analysis

5.1 Two-Layer Security Model

TLO provides security through complementary layers with different bases:

1. **Topology layer (heuristic):** Empirically defeats structural/statistical attacks through wire selection. *Empirically validated, not proven.*

2. LWE **layer (computational):** Provides representation hiding via on-chain inner products. *Based on* LWE *hardness (*$\sim$*49-bit with* n=64*; see §5.3).*

3. **Unlocking security:** Reduces to secret search in the input space (and/or breaking LWE). Attackers can evaluate the locked predicate offline for arbitrary inputs.

Definition 5.1 (Attack-Matrix Resistance)**.** Given a fixed finite set of attack classes $\mathcal{A}$ and an evaluation metric, we say that an obfuscation scheme achieves $\mathcal{A}$-resistance if no PPT adversary implementing any attack in $\mathcal{A}$ succeeds with probability exceeding a specified threshold (here, empirically negligible on our benchmarks).

Theorem 5.2 (TLO Attack-Matrix Resistance)**.** *Under our* LWE*-based representation-hiding assumption, our empirical topology heuristics, and the wrong-key-gives-garbage property, TLO achieves* $\mathcal{A}$*-resistance for our 6-class attack matrix on the evaluated parameter sets.*

Proof (informal): Structural/statistical attacks in $\mathcal{A}$ are thwarted by the topology layer in our experiments. RainbowTable-style attacks in $\mathcal{A}$ that rely on explicit control-function recovery are blocked by LWE-based representation hiding. We do not claim resistance to all possible attacks outside $\mathcal{A}$. □

5.2 Attack Evaluation Matrix

Attack	**Defense**	**Basis**	**Status**
Compression	Topology	Structural	Blocked
PatternMatch	Topology	Structural	Blocked
Structural	Topology	Structural	Blocked
Statistical	Topology	Statistical	Blocked
DiagCorrelation	Topology	Statistical	Blocked
RainbowTable	LWE	Semantic	Blocked*

Table 2: TLO attack resistance within our defined matrix. *Blocks structural rainbow-table matching; does not prevent black-box evaluation.

5.3 Security Estimates

Uniform-Secret LWE**.** TLO derives the LWE secret as $s_{\mathsf{enc}} = H(\mathsf{secret})$, producing a *uniform* secret over $\mathbb{Z}_q^n$ rather than a small-coefficient secret. This variant is *harder* to attack: primal (uSVP) attacks fail when $\|s\| \approx \sqrt{n} \cdot q/2$. We validated this via BKZ attacks using fpylll—BKZ-50 on n=16 failed after 200+ iterations.

Using dual-attack analysis (which applies regardless of secret distribution), our parameters (n=64, q=65521, σ=$\sqrt{q}/4$, m=2560 samples) yield $\sim$49-bit security. This is suitable for *eventually-expiring secrets* with hour-to-day lifetimes. For n=128: $\sim$81-bit; for n=256: $\sim$132-bit (NIST-level).

Hash-Compare Baseline: A simple `H(secret) == H(input)` check costs $\sim$45K gas but provides *no* representation hiding—the predicate structure is visible on-chain.

Multi-Bit Output: The Key Distinction. Hash-compare returns a 1-bit output (true/false). TLO circuits compute an N-bit output that can encode hidden parameters, computed results, or payloads revealed only on correct input. Both implement *point functions*—predicates meaningful only at $x = \mathsf{secret}$—but TLO provides a hidden payload, not just confirmation.

Approach	**Output**	**What's Hidden**
Hash-compare	1 bit	Secret value only
TLO	N bits	Secret + hidden computation

Comparison: In both designs, an attacker can evaluate the predicate on arbitrary inputs (on-chain or off-chain), so the difficulty of finding a satisfying input before expiry primarily depends on the secret's entropy and application-level constraints, not on LWE. TLO's advantage over hash-compare is: (1) multi-bit hidden output, and (2) hiding the internal predicate representation (topology + control functions). The 57× gas premium buys multi-bit hidden computation, not stronger unlocking security.

5.4 Post-Quantum Security

TLO's LWE layer provides post-quantum *representation hiding* (assuming LWE quantum-resistance):

- **Topology layer:** No cryptographic assumptions
- **Lattice layer:** LWE is believed quantum-resistant

5.5 Assumptions

1. LWE **hardness:** Learning With Errors is computationally hard.
2. **Topology empirical security:** Wire selection defeats structural attacks in our evaluation (heuristic, not proven).
3. **Contract correctness:** Expiry logic is correctly implemented.

6 Implementation

6.1 Contract Architecture

TLO requires no external infrastructure—just a standard smart contract with timestamp-based expiry:

Listing 2: TLOHoneypot Contract

```
contract TLOHoneypot {
  uint256 public secretExpiry;
  bytes32 public commitHash;

  function check(bytes32 s) external view
      returns (bool) {
    require(block.timestamp < secretExpiry);
    return evaluate(s);
  }

  function commit(bytes32 h) external {
    commitHash = h;
  }

  function reveal(bytes32 s) external {
    require(keccak256(abi.encode(s,
        msg.sender)) == commitHash);
    require(evaluate(s));
    // Transfer reward
  }
}
```

Deployment: Set `secretExpiry` at deployment.

6.2 Gas Costs

Measured on 64-wire/640-gate circuits (Tenderly-confirmed):

LWE n	Security	Gas	Block %
16	∼22-bit	744K	2.5%
32	∼22-bit	1.27M	4.2%
64	**∼49-bit**	**2.58M**	**8.6%**
128	∼81-bit	4.9M	16.3%

Table 3: Gas costs by LWE dimension (uniform-secret security estimates for representation hiding).

7 Evaluation

7.1 Attack Resistance

We evaluated TLO against 14 attack implementations across 6 attack classes. All configurations achieve 6/6 resistance within our defined attack matrix:

LWE n	Score	Gas	Security
16	6/6	744K	∼22-bit
32	6/6	1.27M	∼22-bit
64	**6/6**	**2.58M**	**∼49-bit**
128	6/6	4.9M	∼81-bit

7.2 Comparison with Alternatives

Property	TLO	Hash	iO	TEE
Attack matrix	6/6	0/6	6/6	6/6
Repr. hiding	Yes	No	Yes	Yes
Unlocking	Search	Search	Search	Key
Gas (check)	2.58M	45K	$10^6\times$	1×
Infrastructure	None	None	None	HW
Post-quantum	Yes	Yes	Depends	No
Practical	Yes	Yes	No	Yes

8 Applications

8.1 Valid Applications

TLO is designed for predicates with *eventually-expiring* secrets where representation hiding provides value:

- **Cryptographic honeypots:** Reward condition is burned once triggered
- **Sealed-bid auctions:** Bids revealed at settlement
- **Lotteries/prediction markets:** Outcomes revealed after close
- **MEV protection:** Order flow is short-lived
- **Dark pools:** Trade conditions expire quickly

8.2 Invalid Applications

TLO is *not* intended for long-lived static secrets or applications requiring semantic security:

- Long-term decryption keys
- Permanent signing keys
- Static liquidation thresholds (see §9)

9 Limitations

Theoretical: Topology security is empirical (heuristic, not proven). We do not claim iO-level indistinguishability or semantic security.

Practical: TLO with n=64 LWE requires ∼2.58M gas (8.6% of block limit). Lower security configurations available for cost-sensitive applications.

What TLO does NOT provide:

- **Prevention of offline evaluation:** Attackers can evaluate the locked predicate for arbitrary inputs
- **Indistinguishability:** Two circuits have distinguishable obfuscations
- **Universal security:** Only resists our 6 attack classes
- **Forward secrecy:** Expired secrets may be analyzed retroactively
- **Security beyond hash-compare for unlocking:** Unlocking difficulty is governed by secret entropy, not LWE

10 Related Work

Indistinguishability Obfuscation: Theoretical iO [3, 5] provides strong security but requires impractical overhead.

Local Mixing: Canetti et al. [1] explore heuristic obfuscation via local perturbations of reversible circuits. Like their work, we provide candidate constructions with empirical (not proven) security.

Compute-and-Compare: Goyal-Koppula-Waters [4] and Wichs-Zirdelis [7] introduced C&C for evasive functions. We apply similar ideas to control function hiding.

Smart Contract Privacy: Previous work uses ZK-SNARKs (Tornado Cash) or TEEs (Secret Network). TLO provides a new point in the design space: on-chain representation hiding without trusted hardware.

11 Conclusion

TLO provides practical heuristic obfuscation for smart contracts through two-layer defense: a topology layer (heuristic) empirically defeats structural/statistical attacks, while on-chain LWE inner products provide representation hiding for control functions.

TLO achieves 6/6 resistance within our defined attack matrix at ∼2.58M gas (n=64, ∼49-bit uniform-secret LWE security for representation hiding). Post-quantum representation hiding (assuming LWE quantum-resistance). Deployment requires only a standard smart contract with timestamp expiry.

Key contributions: On-chain LWE inner products for representation hiding; honest characterization of security properties; discovery that uniform-secret LWE provides stronger security than small-secret variants.

Honest scope: Unlocking security reduces to secret search, not to preventing offline evaluation. TLO hides *how* the predicate is implemented, not *whether* it can be evaluated.

Code and attack suite: https://github.com/igor53627/tlo

References

[1] R. Canetti, C. Chamon, E. Mucciolo, and A. Ruckenstein. Towards general-purpose program obfuscation via local mixing. Cryptology ePrint Archive, Report 2024/006, 2024.

[2] National Institute of Standards and Technology. Module-Lattice-Based Key-Encapsulation Mechanism Standard (ML-KEM). NIST FIPS 203, 2024.

[3] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013.

[4] R. Goyal, V. Koppula, and B. Waters. Lockable obfuscation. In *FOCS*, 2017.

[5] A. Jain, H. Lin, and A. Sahai. Indistinguishability obfuscation from well-founded assumptions. In *STOC*, 2021.

[6] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, 2005.

[7] D. Wichs and G. Zirdelis. Obfuscating compute-and-compare programs under LWE. In *FOCS*, 2017.

[8] M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of Learning with Errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.