# Day 8: Zero-Knowledge Proofs for Finance

## Interactive Proofs, Polynomial Commitments, and Privacy-Preserving DeFi

Prof. Dr. Jörg Osterrieder

PhD Seminar Series: Digital Finance Research

2026

**PhD Seminar Series: Digital Finance Research**

# Proving You Are Rich Without Showing Your Bank Statement

## Traditional Finance: Full Disclosure

- Alice applies for a DeFi loan
- Must reveal: bank statements, portfolio, tax returns, transaction history
- Lender sees **everything**
- Privacy cost of credit access

## ZK Finance: Selective Proof

- Alice generates a ZK proof
- Proves: "total assets $>$ \$100K"
- Reveals: **nothing** about composition, history, or identity
- Lender verifies in milliseconds

*"The most profound implication of zero-knowledge proofs is that verification need not require knowledge."* — Shafi Goldwasser

# The Seminar So Far

## Days 1–4: Quantitative Foundations

- Crypto derivatives pricing (Heston, jump-diffusion)
- DeFi mathematics (CFMM geometry, IL)
- Blockchain economics (MEV, mechanism design)
- ML microstructure (DeepLOB, RL)

## Days 5–7: Regulation & Innovation

- Systemic risk and network topology
- DeFi derivatives (perpetuals, funding rates)
- AI agents in autonomous finance

## Today: Day 8

- Zero-knowledge proof systems formalized
- Polynomial commitment schemes
- ZK-rollups and privacy-preserving DeFi

# Research Context: Why ZK Proofs Matter for Finance

## Three Fundamental Problems in Digital Finance

1. **Scalability:** Ethereum processes $\sim$15 TPS; global finance requires $>$100,000 TPS
2. **Privacy:** All DeFi transactions are public — institutional adoption requires confidentiality
3. **Compliance:** Regulators demand KYC/AML; users demand privacy

## ZK proofs address all three simultaneously

- **ZK-rollups:** 100$\times$ throughput via off-chain computation with on-chain verification
- **Private transactions:** Prove validity without revealing amounts or parties
- **ZK-KYC:** Prove compliance without revealing identity

# Today's Roadmap

1. Interactive Proof Systems Formalized
2. Polynomial Commitment Schemes
3. ZK-Rollup Validity Proofs
4. Privacy-Preserving DeFi
5. Research Frontiers and Open Problems

# Outline

1. Interactive Proof Systems Formalized
2. Polynomial Commitment Schemes
3. ZK-Rollup Validity Proofs
4. Privacy-Preserving DeFi
5. Research Frontiers and Open Problems

# Interactive Proof Systems: Formal Definition

## Definition 1 (Interactive Proof System [ ])

An **interactive proof system** for a language $L$ is a pair $(P, V)$ of interactive probabilistic Turing machines satisfying:

1. **Completeness.** $\forall\, x \in L$:
$$\mathbb{P}\big[\langle P, V\rangle(x) = 1\big] \geq 1 - \mathrm{negl}(|x|)$$
2. **Soundness.** $\forall\, x \notin L$, $\forall$ computationally unbounded $P^*$:
$$\mathbb{P}\big[\langle P^*, V\rangle(x) = 1\big] \leq \mathrm{negl}(|x|)$$

where $\mathrm{negl}(\cdot)$ is a negligible function: $\forall\, c > 0$, $\exists\, n_0$ s.t. $\nu(n) < n^{-c}$ for $n > n_0$.

$P$ is computationally unbounded ("all-powerful prover"); $V$ runs in probabilistic polynomial time. The class of languages with interactive proofs is **IP**.

# The Power of Interaction: IP = PSPACE

## Theorem 2 (Shamir 1992 [ ])

**IP** = **PSPACE**.

## Implications for Finance

- Any property computable in polynomial space can be interactively verified
- Includes: portfolio solvency, compliance with complex regulations, validity of arbitrary computations
- The verifier needs only polynomial time — exponentially less than recomputing the result

## Complexity Hierarchy

$$\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{IP} = \mathbf{PSPACE} \subseteq \mathbf{EXP}$$

Interactive proofs are *strictly* more powerful than NP witnesses (unless **NP** = **PSPACE**).

# The Zero-Knowledge Property: Simulation Paradigm

## Definition 3 (Zero-Knowledge [ ])

An interactive proof $(P, V)$ for $L$ is **zero-knowledge** if for every PPT verifier $V^*$, there exists a PPT **simulator** $S$ such that:

$$\left\{\text{View}_{V^*}\left[\langle P(w), V^*(z)\rangle(x)\right]\right\}_{x\in L} \overset{c}{\approx} \left\{S(x,z)\right\}_{x\in L}$$

where $w$ is the witness, $z$ is auxiliary input, and $\overset{c}{\approx}$ denotes computational indistinguishability.

## Key Insight

- The **view** includes: all messages from $P$, $V^*$'s random coins, and $V^*$'s input
- $S$ produces a *fake transcript* that is indistinguishable from a real one
- $S$ does **not** have the witness $w$ — it simulates without knowledge
- Anything $V^*$ "learns" from the proof, it could have computed alone

# Variants of Zero-Knowledge

## Three Flavors

1. **Perfect ZK:** $\mathrm{View}_{V^*} \equiv S(x, z)$ — identical distributions
2. **Statistical ZK:** $\mathrm{View}_{V^*} \overset{s}{\approx} S(x, z)$ — statistically close: $\Delta\big(\mathrm{View}, S(x, z)\big) \leq \mathsf{negl}(|x|)$
3. **Computational ZK:** $\mathrm{View}_{V^*} \overset{c}{\approx} S(x, z)$ — no PPT distinguisher

## Honest-Verifier vs. Malicious-Verifier

- **HVZK:** Simulator works only against honest $V$ (follows protocol)
- **Malicious-verifier ZK:** Simulator works against any PPT $V^*$
- **Theorem** (Goldreich–Krawczyk, 1996): Any HVZK proof can be compiled into a malicious-verifier ZK proof (for public-coin protocols)

Financial applications require malicious-verifier ZK — the counterparty is untrusted.

# Arguments of Knowledge: Extractability

## Definition 4 (Proof of Knowledge)

An interactive proof $(P, V)$ for relation $R$ is a **proof of knowledge** if there exists a PPT **extractor** $E$ such that for every prover $P^*$: if $P^*$ convinces $V$ on input $x$ with non-negligible probability, then $E^{P^*}(x)$ outputs a witness $w$ with $(x, w) \in R$.

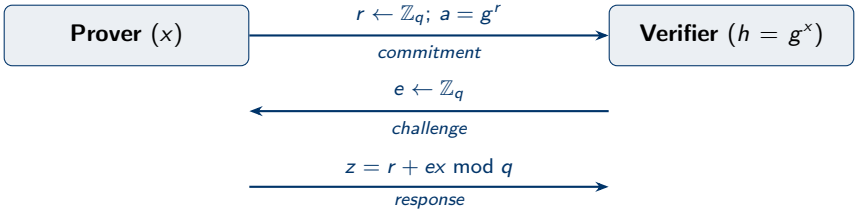
## Why This Matters for Finance

- **Soundness alone:** "The statement is true" (e.g., the transaction is valid)
- **Knowledge soundness:** "The prover *knows* a witness" (e.g., knows the private key)
- Prevents delegated proofs: cannot prove ownership without actually possessing the key
- SNARKs and STARKs are **arguments** of knowledge (soundness holds against computationally bounded provers only)

# Example: Schnorr Identification Protocol

## Setup

Group $\mathbb{G}$ of prime order $q$ with generator $g$. Prover knows $x \in \mathbb{Z}_q$ (secret key); public key $h = g^x$.




**Verification:** Accept iff $g^z = a \cdot h^e$. **Properties:** Complete, special-sound, HVZK.

# Security of Schnorr: Reduction to Discrete Logarithm

## Proposition 5 (Special Soundness)

*Given two accepting transcripts $(a, e_1, z_1)$ and $(a, e_2, z_2)$ with $e_1 \neq e_2$ (same commitment $a$), one can extract $x$:*

$$x = \frac{z_1 - z_2}{e_1 - e_2} \bmod q$$

## Proof.

From $g^{z_1} = a \cdot h^{e_1}$ and $g^{z_2} = a \cdot h^{e_2}$:

$$g^{z_1 - z_2} = h^{e_1 - e_2} = g^{x(e_1 - e_2)} \implies z_1 - z_2 \equiv x(e_1 - e_2) \pmod{q}$$

Since $q$ is prime and $e_1 \neq e_2$, we invert: $x = (z_1 - z_2)(e_1 - e_2)^{-1} \bmod q$. □

## HVZK Simulator

$S$ picks $z \leftarrow \mathbb{Z}_q$, $e \leftarrow \mathbb{Z}_q$, sets $a = g^z h^{-e}$. Transcript $(a, e, z)$ has identical distribution to real. No witness needed.

# Fiat–Shamir Transform: From Interactive to Non-Interactive

## Definition 6 (Fiat–Shamir Heuristic [ ])

Replace the verifier's random challenge $e$ with the output of a cryptographic hash function $\mathcal{H}$:

$$e = \mathcal{H}(x \| a)$$

The resulting protocol is **non-interactive**: the prover computes the entire proof $\pi = (a, z)$ without any verifier message.

## Security

- In the **Random Oracle Model**: provably sound if original protocol is special-sound [8]
- In the **standard model**: counterexamples exist (Goldwasser–Kalai, 2003)
- All practical SNARKs and STARKs use Fiat–Shamir — security is heuristic but holds in practice

This is the bridge from *interactive* proofs (theory) to *non-interactive* proofs (blockchain).

# Arithmetic Circuits: The Language of ZK Proofs

## Definition 7

An **arithmetic circuit** $C$ over field $\mathbb{F}_p$ is a DAG whose:

- Input nodes: public input $x$ and private witness $w$
- Internal gates: addition ($+$) and multiplication ($\times$) over $\mathbb{F}_p$
- Output: single field element; $C(x, w) = 0$ iff the statement holds

## Example: Proving Knowledge of a Preimage

Statement: "I know $w$ such that $\text{SHA256}(w) = h$" for public $h$.

- SHA256 decomposes into $\sim$25,000 arithmetic gates over $\mathbb{F}_p$
- Prover supplies $w$ as private input
- Circuit evaluates SHA256 and checks output $= h$
- Proof size: $\sim$200 bytes (SNARK) regardless of circuit size

**Circuit size** (number of gates) determines proving time and is the key efficiency metric.

# Outline

1 Interactive Proof Systems Formalized

2 Polynomial Commitment Schemes

3 ZK-Rollup Validity Proofs

4 Privacy-Preserving DeFi

5 Research Frontiers and Open Problems

# From Circuits to Polynomials: The Core Reduction

## Key Insight (Schwartz–Zippel Lemma)

Checking that a computation is correct reduces to checking that a **polynomial identity** holds. If $p(X)$ and $q(X)$ are distinct polynomials of degree $\leq d$ over $\mathbb{F}_q$, then:

$$\mathbb{P}_{r \leftarrow \mathbb{F}_q}\big[p(r) = q(r)\big] \leq \frac{d}{|\mathbb{F}_q|}$$

For $|\mathbb{F}_q| \approx 2^{256}$ and $d \approx 10^6$: probability $\approx 2^{-236}$.

## Proof System Architecture

1. **Arithmetization:** Encode computation as polynomial constraints
2. **Polynomial commitment:** Prover commits to polynomials; verifier queries at random points
3. **Evaluation proof:** Prover proves the committed polynomial evaluates correctly at the queried point

The choice of polynomial commitment scheme determines SNARK vs. STARK.

# KZG Polynomial Commitments [ ]

## Definition 8 (KZG Setup)

Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear pairing. **Trusted setup** generates: for secret $\tau \in \mathbb{F}_p$,

$$\mathrm{SRS} = \left(g, g^{\tau}, g^{\tau^2}, \ldots, g^{\tau^d},\ h, h^{\tau}\right) \quad \text{where } g \in \mathbb{G}_1,\ h \in \mathbb{G}_2$$

After generation, $\tau$ **must be destroyed** ("toxic waste").

## Definition 9 (KZG Commitment and Opening)

For polynomial $f(X) = \sum_{i=0}^{d} a_i X^i$:

- **Commit:** $C = g^{f(\tau)} = \prod_{i=0}^{d} (g^{\tau^i})^{a_i}$ (single $\mathbb{G}_1$ element, 48 bytes)
- **Open at $z$:** Compute quotient $q(X) = \frac{f(X)-f(z)}{X-z}$; proof $\pi = g^{q(\tau)}$
- **Verify:** Check $e(C/g^{f(z)},\ h) = e(\pi,\ h^{\tau}/h^{z})$

# KZG Security: Reduction to $q$-Strong Diffie–Hellman

## Definition 10 ($q$-SDH Assumption)

Given $(g, g^{\tau}, g^{\tau^2}, \ldots, g^{\tau^q})$, it is computationally hard to output $(c, g^{1/(\tau+c)})$ for any $c \in \mathbb{F}_p$.

## Theorem 11 (Binding under $q$-SDH [ ])

*If the q-SDH assumption holds, then no PPT adversary can open a KZG commitment to two different values at the same point, except with negligible probability.*

## Security Implications for Finance

- $q$-SDH is a **discrete-log-type** assumption on elliptic curves
- **Broken by quantum computers:** Shor's algorithm computes discrete logs in poly($n$) time
- All KZG-based SNARKs (Groth16, Plonk, Marlin) inherit this vulnerability
- Timeline concern: "store now, break later" attacks on committed data

# Trusted Setup: Powers-of-Tau Ceremony

## Protocol

1. Participant 1 picks random $\tau_1$, computes $(g^{\tau_1}, g^{\tau_1^2}, \ldots)$, **destroys** $\tau_1$
2. Participant 2 picks random $\tau_2$, raises each element to $\tau_2$: $(g^{\tau_1\tau_2}, g^{(\tau_1\tau_2)^2}, \ldots)$, destroys $\tau_2$

$\vdots$

N. Participant $N$ contributes $\tau_N$. Final SRS uses $\tau = \prod_i \tau_i$.

## Security Guarantee: 1-of-$N$ Trust

If **at least one** participant honestly destroys their $\tau_i$, the combined $\tau$ is unknown to all. An adversary must compromise **every** participant.

## Ethereum KZG Ceremony (2023)

- >140,000 contributors — largest trusted setup in history
- Contributions from diverse hardware, locations, entropy sources
- Used for EIP-4844 (proto-danksharding) blob commitments

# FRI: Fast Reed–Solomon Interactive Oracle Proofs [ ]

## Definition 12 (FRI Protocol)

**FRI** (Fast RS-IOPP) proves that a function $f : \mathcal{D} \to \mathbb{F}_p$ (given via a Merkle tree of evaluations) is "close" to a polynomial of degree $< d$.

## Key Steps

1. **Commit:** Prover evaluates $f$ over domain $\mathcal{D}$ ($|\mathcal{D}| \gg d$), commits via Merkle tree
2. **Fold:** Verifier sends random $\alpha \in \mathbb{F}_p$. Prover "folds": $f_1(X) = \frac{f(X)+f(-X)}{2} + \alpha \cdot \frac{f(X)-f(-X)}{2X}$
3. **Recurse:** $f_1$ has degree $< d/2$ over domain $\mathcal{D}/2$. Repeat $\log_2(d)$ times until constant.
4. **Query:** Verifier spot-checks consistency at random positions

**No trusted setup.** Security relies only on collision-resistant hash functions.

# FRI Security: Hash-Based, Post-Quantum

## Theorem 13 (FRI Soundness [ ])

*If $f$ is $\delta$-far from any polynomial of degree $< d$ (in relative Hamming distance over $\mathcal{D}$), then the FRI verifier rejects with probability $\geq 1 - O(d/|\mathcal{D}|)^{queries}$.*

## Security Assumptions: STARK vs. SNARK

| **Property** | **KZG (SNARK)** | **FRI (STARK)** |
|---|---|---|
| Assumption | $q$-SDH (elliptic curves) | Collision-resistant hashes |
| Trusted setup | Required (powers-of-tau) | None (transparent) |
| Post-quantum | No (Shor's algorithm) | Yes |
| Proof size | $\sim$300 bytes | $\sim$50–200 KB |
| Verification | $O(1)$ pairings | $O(\log^2 d)$ hashes |

Trade-off: SNARKs are compact but trust-dependent; STARKs are trustless but larger.

# Plonk: Universal SNARKs [ ]

## Plonk Innovation

- **Universal SRS:** One trusted setup works for *any* circuit up to size $N$ (no per-circuit ceremony)
- **Updateable:** New participants can strengthen the SRS post-hoc
- Proof system combines **permutation arguments** with **KZG commitments**

## Plonk Arithmetization: Gate Equation

For wire values $(a_i, b_i, c_i)$ at gate $i$, Plonk enforces:

$$q_L \cdot a_i + q_R \cdot b_i + q_O \cdot c_i + q_M \cdot a_i b_i + q_C = 0$$

where $q_L, q_R, q_O, q_M, q_C$ are **selector polynomials** (fixed by the circuit).

## Adoption

Used by zkSync Era, Polygon zkEVM, Scroll, Aztec — the dominant SNARK in production.

# Proof System Landscape: Comparative Analysis

| | Groth16 | Plonk | STARK | Halo2 | Nova |
|---|---|---|---|---|---|
| Commitment | KZG | KZG | FRI | IPA | IPA |
| Setup | Per-circuit | Universal | Transparent | Transparent | Transparent |
| Proof size | 192 B | ∼400 B | 50–200 KB | ∼1 KB | ∼10 KB |
| Verify time | 3 ms | 5 ms | 50 ms | 20 ms | 30 ms |
| Prover time | Fast | Medium | Slow | Medium | Fast (folding) |
| Post-quantum | No | No | Yes | No | No |
| On-chain gas | ∼200K | ∼300K | ∼2M | ∼500K | N/A |
| Used by | Zcash, Tornado | zkSync, Scroll | StarkNet, dYdX | Zcash Orchard | Research |

## Selection Criteria for Financial Applications

- **On-chain settlement** (gas-sensitive): Groth16 or Plonk
- **Regulatory compliance** (no trust assumption): STARK
- **Long-term privacy** (quantum resistance): STARK

# Outline

1 Interactive Proof Systems Formalized

2 Polynomial Commitment Schemes

3 ZK-Rollup Validity Proofs

4 Privacy-Preserving DeFi

5 Research Frontiers and Open Problems

# ZK-Rollup: Architecture and Security Model

## Definition 14

A **ZK-rollup** is an L2 scaling solution that:

1. Executes transactions off-chain (sequencer)
2. Generates a ZK validity proof $\pi$ that all state transitions are correct
3. Posts $(\pi, \Delta_{\text{state}}, \text{calldata})$ to L1
4. L1 verifier contract checks $\pi$ — if valid, new state root is accepted

## Security Properties

- **Validity:** Soundness of the proof system $\Rightarrow$ only valid state transitions are accepted
- **Data availability:** Calldata posted to L1 ensures anyone can reconstruct state ("rollup" not "validium")
- **Censorship resistance:** Forced inclusion via L1 contract
- **Liveness:** Depends on sequencer — centralized liveness risk

# Validity Proof: What Exactly Is Being Proved?

## The Statement

Given public inputs $(S_{\text{old}}, S_{\text{new}}, \text{txs})$, the proof $\pi$ attests:

$$\exists\, \sigma_1, \ldots, \sigma_n \text{ (valid signatures)} \;:\; \text{Apply}(S_{\text{old}}, \text{tx}_1, \sigma_1, \ldots, \text{tx}_n, \sigma_n) = S_{\text{new}}$$

## Circuit Components

1. **Signature verification:** ECDSA or EdDSA per transaction
2. **Merkle proof updates:** Verify and update account balances in state tree
3. **Balance checks:** Sender has sufficient funds (no negative balances)
4. **Nonce checks:** Prevent replay attacks
5. **State root computation:** Final Merkle root matches $S_{\text{new}}$

A batch of 10,000 transactions $\Rightarrow$ circuit with $\sim 10^9$ gates $\Rightarrow$ proving time $\sim$2–10 minutes on specialized hardware.

# ZK-Rollup Economics: Cost Analysis

## Cost Decomposition

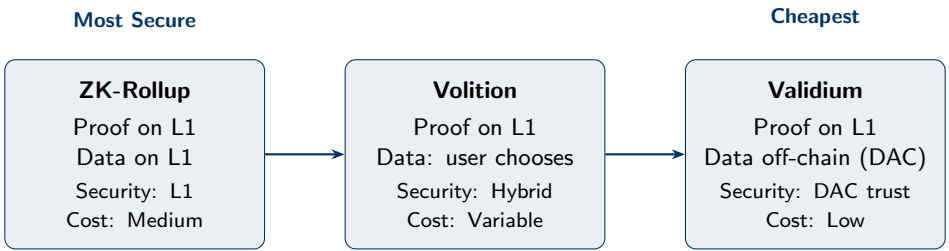
$$C_{\text{user}} = \underbrace{\frac{C_{\text{proof verify}}}{N}}_{\text{amortized verification}} + \underbrace{\frac{C_{\text{calldata}}}{N}}_{\text{amortized data}} + \underbrace{C_{\text{sequencer}}}_{\text{L2 execution}}$$

where $N$ is the batch size.

## Numerical Example (2025 data)

| **Component** | **L1 (Ethereum)** | **ZK-Rollup** |
|---|---:|---:|
| Simple transfer | \$5.00 | \$0.05 |
| Token swap | \$15.00 | \$0.15 |
| Throughput (TPS) | 15 | 2,000 |
| Finality | 12 min | 2 min (proof time) |
| Batch size | 1 tx | 10,000 tx |
| Proof verification | — | 200K gas (∼\$0.50 total) |

# Data Availability Spectrum




## Financial Application Mapping

- **Rollup:** High-value DeFi (lending, large swaps) — L1 security required
- **Volition:** Institutional trading — choose per-transaction
- **Validium:** High-frequency, low-value (gaming, micropayments)

# Recursive Proof Composition

## Definition 15 (Proof Recursion)

A proof $\pi_2$ verifies the correctness of a previous proof $\pi_1$, i.e.:

$$\pi_2 \text{ attests: } \text{“}\exists\, \pi_1 \text{ such that } V(\pi_1, x_1) = 1 \text{ and batch}_2 \text{ is valid”}$$

This allows **incremental verification**: a single proof covers all history.

## Applications in Rollup Design

- **Proof aggregation:** Combine proofs from multiple batches into one L1 submission — further amortizes gas cost
- **Cross-rollup bridging:** Rollup A verifies Rollup B's proof on-chain, enabling trust-minimized asset transfers
- **IVC (Incrementally Verifiable Computation):** Nova/SuperNova fold proofs in $O(1)$ time per step

Recursive proofs solve the “proof-of-proofs” problem: verify an entire chain in one check.

# ZK-EVM: Proving Ethereum Execution

## The Challenge

Proving arbitrary EVM execution in a ZK circuit:

- EVM has $\sim$140 opcodes, stack-based, 256-bit word size
- Naive circuit: $>10^{12}$ constraints for a single block
- Requires specialized "opcode circuits" and lookup arguments

## ZK-EVM Type Classification (Vitalik Buterin)

| Type | Equivalence | Compatibility | Speed |
|---|---|---|---|
| 1 | Ethereum-equivalent | Full | Hours |
| 2 | EVM-equivalent | Full | Minutes |
| 2.5 | Gas-adjusted EVM | Minor diffs | Minutes |
| 3 | Almost EVM | Most contracts | Seconds |
| 4 | High-level language | Source only | Seconds |

# ZK-Rollup Market Landscape (2025)

## Total Value Locked

| Rollup | TVL |
|---|---|
| zkSync Era | ∼$8B |
| StarkNet | ∼$5B |
| Linea | ∼$4B |
| Scroll | ∼$3B |
| Polygon zkEVM | ∼$2B |
| Others | ∼$6B |
| **Total** | **∼$28B** |

## Proof Systems Used

- **zkSync:** Boojum (custom Plonk)
- **StarkNet:** STARK (Cairo VM)
- **Linea:** Plonk + lattice lookups
- **Scroll:** Halo2 (KZG)
- **Polygon:** Plonky2 (FRI-based)

## Open Research Questions

- Prover decentralization
- MEV in ZK-rollups
- Cross-rollup composability

# Security Theorem for ZK-Rollup State Transitions

## Theorem 16 (Rollup Soundness)

*Let $\Pi = (\mathit{Setup}, \mathit{Prove}, \mathit{Verify})$ be a knowledge-sound SNARK with extraction error $\varepsilon$. Then for any PPT adversary $\mathcal{A}$:*

$$\mathbb{P}\left[\begin{array}{l} \mathit{Verify}(\mathrm{vk}, (S_{\mathit{old}}, S_{\mathit{new}}), \pi) = 1 \\ \wedge\ S_{\mathit{new}} \neq \mathit{Apply}(S_{\mathit{old}}, \mathrm{txs}) \end{array}\right] \leq \varepsilon(n)$$

*i.e., no invalid state transition is accepted except with negligible probability.*

## What This Does **Not** Guarantee

- **Liveness:** Sequencer may censor or halt (mitigated by forced inclusion)
- **MEV resistance:** Sequencer ordering is not constrained by the proof
- **Privacy:** Standard rollups are transparent — see Section IV for privacy

# Outline

1 Interactive Proof Systems Formalized

2 Polynomial Commitment Schemes

3 ZK-Rollup Validity Proofs

4 Privacy-Preserving DeFi

5 Research Frontiers and Open Problems

# Privacy in Financial Systems: Formal Definitions

## Definition 17 (Transaction Privacy [ ])

A payment scheme $\Pi$ provides **transaction privacy** if for any two transactions $\mathsf{tx}_0, \mathsf{tx}_1$ with identical public metadata, no PPT adversary can distinguish $\mathsf{tx}_0$ from $\mathsf{tx}_1$ given the ledger $\mathcal{L}$:

$$\left|\mathbb{P}[\mathcal{A}(\mathcal{L}, \mathsf{tx}_0) = 1] - \mathbb{P}[\mathcal{A}(\mathcal{L}, \mathsf{tx}_1) = 1]\right| \leq \mathsf{negl}(n)$$

## Privacy Dimensions

1. **Sender privacy:** Who initiated the transaction?
2. **Receiver privacy:** Who received the payment?
3. **Amount privacy:** How much was transferred?
4. **Function privacy:** What smart contract was invoked?

Current DeFi provides **none** of these. All four are public on Ethereum.

# Privacy via Commitments: The UTXO Model

## Pedersen Commitments

To hide a value $v$ with randomness $r$:

$$\mathrm{Com}(v, r) = g^v h^r \in \mathbb{G}$$

**Hiding:** Given $\mathrm{Com}(v, r)$, $v$ is information-theoretically hidden.
**Binding:** Cannot open to a different value (under DL assumption).

## Note Structure (Zcash/Aztec style)

A **note** $(v, r, \mathsf{pk}_{\mathsf{owner}})$ represents $v$ tokens owned by $\mathsf{pk}_{\mathsf{owner}}$:

- **Note commitment:** $\mathsf{cm} = \mathrm{Hash}(v \| r \| \mathsf{pk}_{\mathsf{owner}})$ stored in a Merkle tree
- **Nullifier:** $\mathsf{nf} = \mathrm{Hash}(\mathsf{sk}_{\mathsf{owner}} \| \mathsf{cm})$ published when note is spent (prevents double-spending)
- **Spending proof:** ZK proof that the prover knows $(v, r, \mathsf{sk})$ such that cm is in the tree and nf is fresh

# Private Transfer: The ZK Circuit

## Statement (Public)

Nullifiers $\mathsf{nf}_1, \mathsf{nf}_2$ (input notes spent), commitments $\mathsf{cm}_3, \mathsf{cm}_4$ (output notes created), Merkle root rt.

## Witness (Private)

Input notes $(v_1, r_1, \mathsf{sk}_1)$, $(v_2, r_2, \mathsf{sk}_2)$; output notes $(v_3, r_3, \mathsf{pk}_3)$, $(v_4, r_4, \mathsf{pk}_4)$; Merkle paths.

## Circuit Constraints

1. Merkle membership: $\mathsf{cm}_1, \mathsf{cm}_2 \in \mathsf{MerkleTree}(\mathsf{rt})$
2. Nullifier correctness: $\mathsf{nf}_i = \mathsf{Hash}(\mathsf{sk}_i \| \mathsf{cm}_i)$
3. Ownership: $\mathsf{pk}_i = g^{\mathsf{sk}_i}$ for input notes
4. Balance: $v_1 + v_2 = v_3 + v_4$ (conservation of value)
5. Range: $v_3, v_4 \in [0, 2^{64})$ (prevent overflow attacks)

The ledger sees only nullifiers and commitments — no amounts, no identities.

# Anonymity Set and Linkability

## Definition 18 (Anonymity Set)

The **anonymity set** for a transaction tx is the set of all notes that tx *could* have consumed. Larger set $\Rightarrow$ stronger privacy.

## Comparison of Privacy Protocols

| Protocol | Anonymity Set | Linkability Risk |
|---|---|---|
| Tornado Cash | Fixed denomination pool | Timing, amount fingerprint |
| Zcash (shielded) | All shielded notes ($\sim$500K) | Low (if fully shielded) |
| Aztec (v2) | All protocol notes | Low (encrypted notes) |
| Ethereum (base) | 1 (fully transparent) | Complete linkability |

## Practical De-anonymization Risks

- **Timing analysis:** Deposit and withdrawal patterns

# Privacy Pools: Compliance-Compatible Privacy [ ]

## Definition 19 (Association Set)

An **association set** $\mathcal{A}$ is a publicly defined set of deposit commitments considered "clean" (not linked to sanctioned activity). A withdrawal proof attests:

$$\exists\, \text{cm} \in \mathcal{A} \cap \text{MerkleTree(rt)} \text{ such that the prover owns cm}$$

without revealing *which* $\text{cm} \in \mathcal{A}$.

## Privacy–Compliance Trade-off

- **Full privacy** (Tornado Cash): anonymity set = all deposits. Includes sanctioned funds $\Rightarrow$ regulatory ban.
- **Privacy Pools:** anonymity set = $\mathcal{A}$ (clean deposits only). Excludes sanctioned funds $\Rightarrow$ compliant.
- Privacy degrades as $|\mathcal{A}|$ shrinks. Optimal $\mathcal{A}$ balances privacy and compliance.

# ZK-KYC: Formal Protocol

## Actors

- **Issuer $\mathcal{I}$:** Licensed KYC provider, signs credentials
- **User $\mathcal{U}$:** Holds credential, generates ZK proof
- **Verifier $\mathcal{V}$:** On-chain contract, checks proof

## Protocol Steps

1. **Credential issuance:** $\mathcal{I}$ signs attestation $\sigma = \mathsf{Sign}_{\mathsf{sk}_{\mathcal{I}}}(\mathsf{pk}_{\mathcal{U}}, \text{attributes})$
2. **Proof generation:** $\mathcal{U}$ computes $\pi$ proving:

$$\exists\, \sigma, \text{attr} \;:\; \mathsf{Verify}_{\mathsf{pk}_{\mathcal{I}}}(\sigma, (\mathsf{pk}_{\mathcal{U}}, \text{attr})) = 1 \;\wedge\; \text{attr} \models \text{policy}$$

3. **On-chain verification:** $\mathcal{V}$ checks $\pi$ against policy (no PII on-chain)

Proof size $\sim$300 bytes. Generation: $\sim$2 seconds on mobile. Verification: $\sim$200K gas.

# Credential Revocation: Merkle Non-Membership Proofs

## Problem

User passes KYC at time $t_0$. At time $t_1 > t_0$, user is sanctioned. How to revoke without linking the credential to an on-chain identity?

## Solution: Revocation Accumulator

1. Maintain a **revocation Merkle tree** $\mathcal{R}$ of revoked credential hashes
2. User proves **non-membership**: Hash(credential) $\notin \mathcal{R}$
3. Non-membership proof via sorted Merkle tree: show two adjacent leaves $\ell_i < \text{Hash} < \ell_{i+1}$ with valid Merkle paths

## Complexity

- Revocation tree with $m$ entries: proof size $O(\log m)$ hashes
- User must update proof when tree changes (or use accumulator with $O(1)$ updates)
- Privacy preserved: verifier sees only "not revoked," not the credential identity

# Aztec: Fully Private Smart Contracts

## Architecture

- ZK-rollup on Ethereum with **encrypted state**
- UTXO model with encrypted notes (not account model)
- Noir: domain-specific language for ZK circuits
- Every transaction is a ZK proof — even contract calls

## What the Blockchain Sees

| Data | Transparent DeFi | Aztec |
|---|---|---|
| Sender address | Visible | Hidden |
| Receiver address | Visible | Hidden |
| Amount | Visible | Hidden |
| Contract called | Visible | Hidden |
| Validity | Recomputed | ZK proof |

# The Privacy Premium: Economic Analysis

## Cost Comparison (per transaction, 2025 estimates)

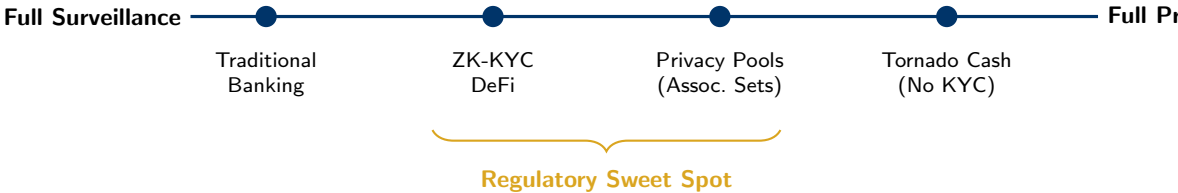
| Operation | L1 (Public) | ZK-Rollup (Public) | ZK-Rollup (Private) |
|---|---|---|---|
| Transfer | \$5.00 | \$0.05 | \$0.15 |
| Swap | \$15.00 | \$0.15 | \$0.50 |
| Lending | \$20.00 | \$0.20 | \$0.80 |
| Proving time | 0 | 2 min/batch | 5 min/batch |

## Privacy Premium Decomposition

$$C_{\text{private}} = C_{\text{public}} + \underbrace{\Delta C_{\text{circuit}}}_{\text{larger circuits}} + \underbrace{\Delta C_{\text{encrypt}}}_{\text{note encryption}} + \underbrace{\Delta C_{\text{nullifier}}}_{\text{double-spend checks}}$$

Privacy costs $\sim$3–4$\times$ more than transparent execution. As proving hardware improves (GPU/FPGA provers), this gap narrows.

# The Privacy–Compliance Spectrum




## Open Research Questions

- What is the optimal size of an association set $|\mathcal{A}|$ to satisfy both privacy ($k$-anonymity) and compliance (exclusion rate)?
- Can differential privacy quantify the information leakage from ZK-KYC proofs?
- Formal game-theoretic model of the regulator–user privacy negotiation

# Outline

1. Interactive Proof Systems Formalized
2. Polynomial Commitment Schemes
3. ZK-Rollup Validity Proofs
4. Privacy-Preserving DeFi
5. Research Frontiers and Open Problems

# Case Study: Tornado Cash and OFAC Sanctions

## Timeline

- 2019–2022: $7.6B in deposits
- Aug 2022: OFAC sanctions smart contract addresses
- 2024: Developer Alexey Pertsev sentenced to 64 months
- Legal question: Is publishing code a crime?

## What Privacy Pools Could Have Done

- Users prove funds are **not** from sanctioned addresses
- Association set excludes Lazarus Group deposits
- Privacy preserved for compliant users
- Protocol remains legally defensible

*"The technology existed to solve this. The timing did not align."*

# Formal Verification of ZK Circuits

## The Problem

ZK circuits are complex ($10^6$–$10^9$ constraints). A single bug can:

- Allow forged proofs (soundness break) $\Rightarrow$ steal funds
- Leak private information (ZK break) $\Rightarrow$ de-anonymize users

Historical bugs: Zcash "counterfeiting" vulnerability (2019), zkSync underconstrained circuits (2023).

## Formal Methods Approaches

1. **Constraint-level verification:** Prove circuit constraints $\Leftrightarrow$ intended relation (Ecne, Picus)
2. **Compiler correctness:** Verify that high-level code compiles to correct constraints (Circom $\rightarrow$ R1CS)
3. **Protocol-level proofs:** Machine-checked security reductions in Coq/Lean for SNARK constructions

# Post-Quantum Zero-Knowledge: Lattice-Based Constructions

## Motivation

- KZG-based SNARKs broken by Shor's algorithm (discrete log in quantum poly($n$))
- "Store now, break later": encrypted/committed data intercepted today, broken when quantum computers arrive
- Financial data has long-term sensitivity (decades)

## Post-Quantum Alternatives

| **Approach** | **Assumption** | **Status** |
|---|---|---|
| STARKs (FRI) | Collision-resistant hashes | Production (StarkNet) |
| Lattice-based | Ring-LWE, Module-SIS | Research |
| MPC-in-the-head | Symmetric crypto only | Limbo, Banquet |
| Isogeny-based | CSIDH | Broken (2023) |

STARKs are the only production-ready post-quantum proof system for finance.

# Open Research Problems

1. **Prover efficiency:** Can we achieve $O(n)$ prover time for general circuits? (Current: $O(n \log n)$ for STARKs, $O(n \log^2 n)$ for Plonk)
2. **Private smart contract composability:** How to compose private function calls across contracts without information leakage?
3. **Optimal association set design:** Game-theoretic model for Privacy Pools — what is the Nash equilibrium of the compliance–privacy game?
4. **MEV in private mempools:** Does transaction encryption eliminate MEV or merely shift it to the sequencer?
5. **Regulatory equilibrium:** Formal model of optimal regulation under information asymmetry where users have ZK capabilities

*"Zero-knowledge proofs are to financial privacy what public-key cryptography was to digital communication — a paradigm shift."*

# Day 8: Key Takeaways

## Core Results

1. **ZK proofs** enable verification without knowledge — formalized via the simulation paradigm ($\exists$ simulator $S$ s.t. View $\overset{c}{\approx} S(x)$)
2. **KZG commitments** (SNARKs): compact proofs, trusted setup, security from $q$-SDH (not post-quantum)
3. **FRI protocol** (STARKs): transparent, post-quantum, security from collision-resistant hashes
4. **ZK-rollups** achieve $100\times$ scaling with L1 security guarantees via validity proofs
5. **Private DeFi** (Aztec, Privacy Pools) resolves the privacy–compliance tension through selective disclosure and association sets

**Next:** Day 9 — Prediction Markets: Information Aggregation Theory

# References I


[1] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. "Scalable, Transparent, and Post-Quantum Secure Computational Integrity". In: *IACR Cryptology ePrint Archive 2018/046*. 2018.

[2] Eli Ben-Sasson et al. "Zerocash: Decentralized Anonymous Payments from Bitcoin". In: *IEEE Symposium on Security and Privacy*. 2014, pp. 459–474.

[3] Vitalik Buterin et al. *Blockchain Privacy and Regulatory Compliance: Towards a Practical Equilibrium*. 2023. arXiv: 2310.07806 [cs.CR].

[4] Amos Fiat and Adi Shamir. "How to Prove Yourself: Practical Solutions to Identification and Signature Problems". In: *Advances in Cryptology – CRYPTO '86*. Springer, 1987, pp. 186–194.

[5] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. "PLONK: Permutations over Lagrange-Bases for Oecumenical Noninteractive Arguments of Knowledge". In: *IACR Cryptology ePrint Archive 2019/953*. 2019.

[6] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. "The Knowledge Complexity of Interactive Proof Systems". In: *SIAM Journal on Computing* 18.1 (1989), pp. 186–208.

[7] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. "Constant-Size Commitments to Polynomials and Their Applications". In: *Advances in Cryptology – ASIACRYPT 2010*. Springer, 2010, pp. 177–194.

[8] David Pointcheval and Jacques Stern. "Security Proofs for Signature Schemes". In: *Advances in Cryptology – EUROCRYPT '96*. Springer, 1996, pp. 387–398.

[9] Adi Shamir. "IP = PSPACE". In: *Journal of the ACM* 39.4 (1992), pp. 869–877.