Topic 4.5: Zero-Knowledge Technology in Finance [ADVANCED]

Privacy and Scalability Through Mathematical Proofs

Joerg Osterrieder

Digital Finance

2025

Learning Objectives

By the end of this topic, you will be able to:

1. **Explain** what zero-knowledge proofs are and why they matter for finance
2. **Describe** the three essential properties: completeness (honest people succeed), soundness (liars get caught), and zero-knowledge (no secrets leaked)
3. **Compare** SNARKs (Succinct Non-interactive ARgument of Knowledge) and STARKs (Scalable Transparent ARgument of Knowledge) in terms of tradeoffs and use cases
4. **Identify** key applications of ZK technology in privacy, scaling, and identity
5. **Evaluate** the privacy-regulation tradeoff enabled by ZK proofs
6. **Assess** the implications of ZK technology for financial compliance

Core Question

How can you prove you know something without revealing what you know?

Prerequisites: Cryptographic Foundations

From Topic 3.1 and 3.2 – Key Concepts You'll Need:

Hash Functions

- One-way transformation
- Deterministic output
- Collision resistant
- Used in commitments

Think: Digital fingerprint – unique, can't be forged

Digital Signatures

- Private key signs
- Public key verifies
- Non-repudiation
- Authentication

Think: Handwritten signature, but mathematically unforgeable

Public Key Cryptography

- Asymmetric key pairs
- Mathematical trapdoors
- Encryption and signing
- Foundation for ZK proofs

Think: Mailbox – anyone can drop mail in (public key), only you can open it (private key)

Why This Matters:

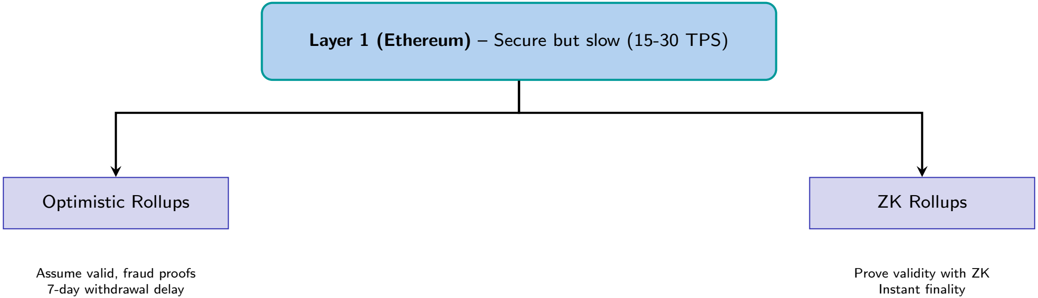
- ZK proofs build on these primitives
- Understanding foundations helps intuition
- Same cryptographic assumptions

If these concepts are unfamiliar, review Topics 3.1-3.2 before continuing

Prerequisites: Layer 2 and Rollups

What is Layer 2? Solutions built on top of Ethereum to make it faster and cheaper while inheriting its security.

From Topic 3.2 – Scaling Context:



Today's Focus: Understanding the "ZK" in ZK-Rollups

Key Insight

Zero-knowledge proofs enable trustless verification: instead of trusting someone's claim, you can mathematically verify it's true.

What is Zero-Knowledge?

"A zero-knowledge proof lets you prove you know something, without revealing what that something is."

Traditional Proof

- "I know the password"
- → Type the password
- Verifier sees the secret
- Password is now exposed

Zero-Knowledge Proof

- "I know the password"
- → Prove via cryptographic protocol
- Verifier is convinced
- Password stays secret

How? The prover demonstrates knowledge by responding to random challenges that only someone with the secret could answer correctly – but the answers themselves don't reveal the secret.

Formal Definition

A **zero-knowledge proof** is a method by which a *prover* can convince a *verifier* that a statement is true, without conveying any information apart from the fact that the statement is indeed true.

The Colorblind Friend Analogy (Part 1)

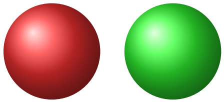
The Classic ZK Explanation:

The Setup:

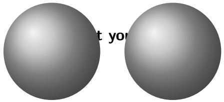
- You have two balls: one RED, one GREEN
- Your friend is colorblind – they look identical to him
- He claims: "These balls are the same color"
- You want to **prove** they're different
- But you don't want to reveal **which is which**

Why This is a ZK Problem:

- **Statement:** "I can distinguish these balls"
- **Secret:** Which color is which
- **Goal:** Convince friend without teaching him colors



Red (Ball A) Green (Ball B)



Gray (Ball A) Gray (Ball B)

What your friend sees

The Colorblind Friend Analogy (Part 2)

The Protocol:

Step 1: Friend holds one ball in each hand, shows you, then hides behind his back

Step 2: Behind his back, he *either* swaps them or keeps them the same

Step 3: He shows you again and asks: "Did I swap them?"

Step 4: You answer correctly (because you can see the colors!)

Repeat: After 20 rounds, chance of lucky guessing $= (\frac{1}{2})^{20} \approx 0.0001\%$

The Zero-Knowledge Property

Your friend becomes convinced the balls are different colors, but learns **nothing** about which ball is red and which is green. You could run this protocol a million times and he still couldn't learn the colors.

The Three Essential Properties

Every zero-knowledge proof must satisfy these three properties:

In Plain English

Completeness = honest people succeed. **Soundness** = liars get caught. **Zero-Knowledge** = no secrets leaked.

1. Completeness

- If the statement is **true**
- And prover is **honest**
- Verifier will be **convinced**

"True claims succeed"

2. Soundness

- If the statement is **false**
- No cheating prover can convince the verifier
- (Except with tiny probability)

"False claims fail"

3. Zero-Knowledge

- Verifier learns **nothing**
- Except that statement is true
- Could simulate proof alone

"No info leaked"

All three are required. Missing any one breaks the system:
No completeness = useless. No soundness = insecure. No zero-knowledge = not private.

The Three Properties Illustrated

Back to the Colorblind Friend:

Completeness

If balls really are different, you *will* detect every swap and answer correctly every time

Soundness

If balls were same color, you could only guess randomly – caught quickly

Zero-Knowledge

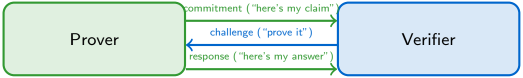
Friend learns balls differ, but not which is red or green

Mathematical Interpretation:

- **Completeness:** The chance that a true statement is accepted is 100%. Notation: $\Pr[\text{Verifier accepts}|\text{statement true}] = 1$
- **Soundness:** The chance that a false statement is accepted is negligible (essentially zero). Notation: $\Pr[\text{Verifier accepts}|\text{statement false}] \leq \epsilon$
- **Zero-Knowledge:** Verifier's view can be simulated without the secret

Interactive vs. Non-Interactive Proofs

Interactive Proofs



- Multiple rounds of communication
- Verifier sends random challenges
- Prover responds to each challenge
- Like the colorblind friend example

Problem: Requires both parties online

Non-Interactive Proofs

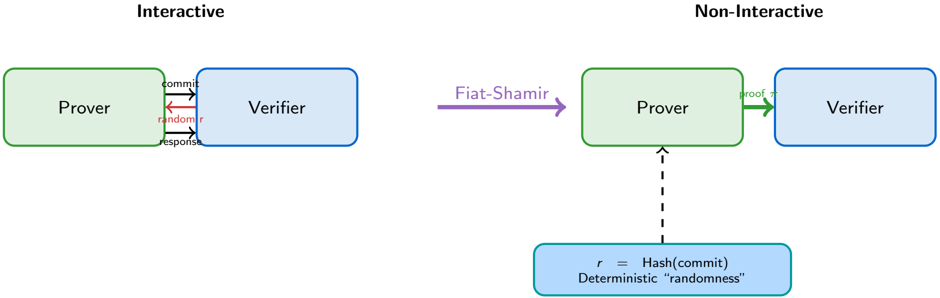
- Single message from prover
- No back-and-forth required
- Proof can be verified anytime
- Essential for blockchain use

Solution: Fiat-Shamir transform

Key Insight

The **Fiat-Shamir heuristic** converts interactive proofs to non-interactive by using a hash function to generate “random” challenges deterministically. This enables ZK proofs on blockchains.

How to Remove Interaction:



The Trick: Instead of waiting for verifier's random challenge, the prover:

1. Computes the commitment
2. Hashes the commitment to get a "random" challenge
3. Computes the response
4. Packages everything into a single proof

What does SNARK stand for?

Think of SNARK and STARK as two different recipes for making the same meal – both create zero-knowledge proofs, just with different ingredients and tradeoffs.

Succinct	**N**on-interactive	**AR**gument	of **K**nowledge
Tiny proof (200 bytes)	One message No interaction	Cryptographic security	Prover knows the secret

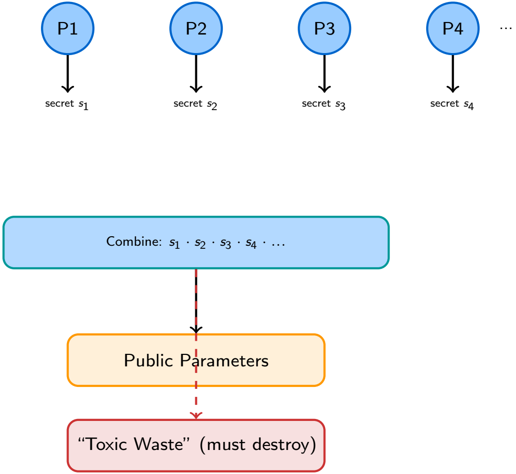
Key Properties:

- **Tiny proofs:** 200 bytes regardless of computation size
- **Fast verification:** Milliseconds
- **One-time setup:** Trusted setup ceremony

The Catch – Trusted Setup:

- Requires generating "toxic waste"
- If anyone keeps the waste, they can forge proofs
- Multi-party ceremonies distribute trust

What is a Trusted Setup?



Security Assumption: At least ONE participant must honestly destroy their secret. If all secrets combine and are destroyed, the system is secure.

What does STARK stand for?

STARK = the same meal as SNARK, but using publicly available ingredients (no secret recipe required).

Scalable	**T**ransparent	**AR**gument	of **K**nowledge
Proof size grows slowly	No trusted setup!	Cryptographic security	Prover knows the secret

Advantages:

- **No trusted setup** – Transparent
- **Quantum resistant** – Uses hash functions
- **Scalable proving** – Faster for large computations

Tradeoffs:

- Larger proofs – 50KB vs 200 bytes
- Newer technology – Less battle-tested
- Still fast verification

How do STARKs avoid trusted setup?

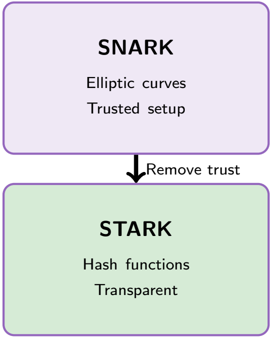
The Key Difference:

- SNARKs use **elliptic curves** – require special parameters
- STARKs use only **hash functions** – publicly known, no secrets needed

Why Hash Functions?

- SHA-256, Poseidon, etc. are standardized
- No hidden trapdoors
- Anyone can verify the math
- Quantum computers can't break them (easily)

The Tradeoff:
Transparency costs proof size. Hash-based cryptography produces larger proofs than elliptic curve cryptography.



SNARK vs. STARK: Head-to-Head Comparison

Property	SNARK	STARK
Proof Size	200 bytes (tiny)	50 KB (larger)
Verification Time	Fast (milliseconds)	Fast (milliseconds)
Prover Time	Moderate	Faster for large computations
Trusted Setup	Required (ceremony)	Not required (transparent)
Quantum Resistant	No	Yes (hash-based)
Maturity	Battle-tested since 2016	Newer (2018+)
Examples	Zcash, zkSync Era Polygon zkEVM	StarkNet, StarkEx dYdX, ImmutableX

Key Takeaway

SNARKs win on proof size (important for on-chain verification costs).
STARKs win on trust assumptions and future-proofing against quantum computers.

ZK in Privacy: Hiding Transaction Details

The Problem with Public Blockchains:

Bitcoin/Ethereum Today:

- All transactions are public
- Anyone can trace funds
- Link addresses to identities
- Complete financial surveillance

Would you want your bank statement public?

With ZK Privacy:

- Transactions are encrypted
- ZK proves validity without revealing details
- Amount, sender, receiver hidden
- Only authorized parties see data

Prove you paid without showing how much

Public: Alice sent 5 ETH to Bob $\xrightarrow{\text{ZK}}$ Private: Valid transaction occurred

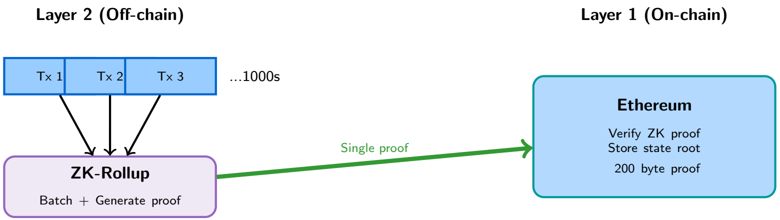
How Zcash Shielded Pools Work:

What the ZK Proof Shows (Without Revealing):

- The sender has sufficient balance (without showing the balance)
- No double-spending occurred (without showing transaction history)
- The amounts balance (inputs = outputs, without showing amounts)

Zcash launched in 2016 – first production ZK-SNARK system

How ZK Proofs Enable Scaling:

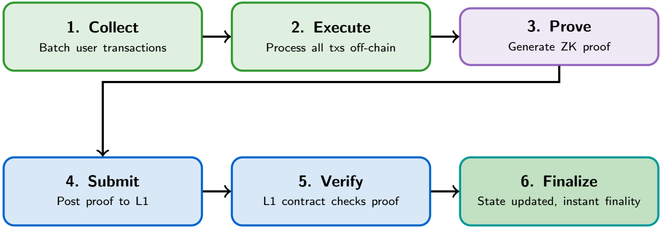


The Magic:

- 1000 transactions → 1 tiny proof (200 bytes)
- L1 verifies proof in milliseconds
- Cost shared across all transactions
- **Result:** 100x+ cheaper transactions with L1 security

ZK-Rollup Verification Flow

Step-by-step process:



Key Advantage over Optimistic Rollups:

- No 7-day withdrawal delay
- Instant finality on L1
- Mathematical certainty, not assumptions

Live ZK-Rollups:

- zkSync Era
- Polygon zkEVM
- StarkNet
- Linea, Scroll

ZK in Identity: Selective Disclosure

The Problem: Proving attributes without revealing everything

Traditional

ID Card Shows:
Name: Alice Smith
DOB: Jan 15, 1995
Address: 123 Main St
ID#: 123-45-6789
Photo, signature...

Bartender learns everything just to verify age

ZK →

With ZK Proof

ZK Proof Shows:

"I am over 21"

(nothing else)

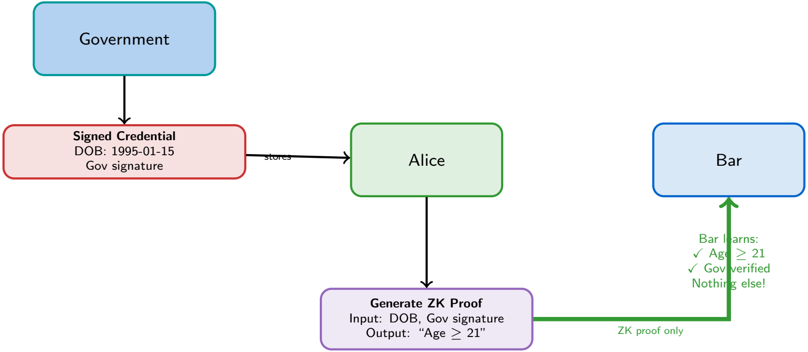
Bartender learns only what they need

Applications

- Age verification without revealing birthdate
- Income proof without showing bank statements
- Nationality verification without passport details
- Credential verification without identity exposure

ZK Age Verification: How It Works

Step-by-step example:



Projects: Polygon ID, Worldcoin, zkPass, Sismo

ZK in DeFi: Privacy-Preserving Protocols

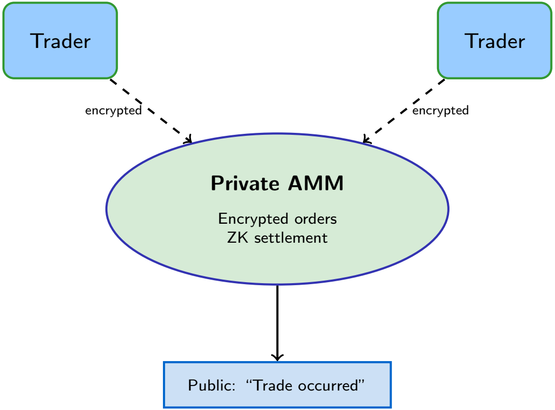
Current DeFi Problem:

Public DeFi Today:

- All trades visible on-chain
- MEV bots front-run transactions
- Whale movements tracked
- Trading strategies exposed

ZK-Enabled DeFi:

- Trade privately
- Prove solvency without showing balance
- Compliant privacy (selective disclosure)
- Protected trading strategies



Examples

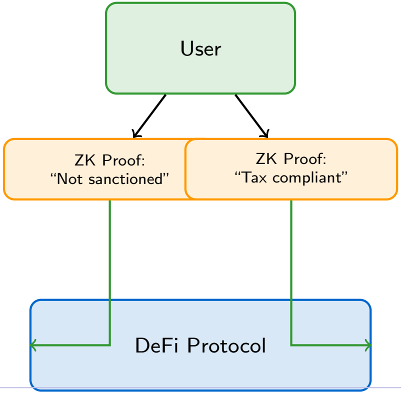
Penumbra (private DEX), Railgun (private DeFi), Aztec (private L2 for DeFi)

The Regulatory Challenge:

"How can we have financial privacy without enabling crime?"

ZK-Enabled Compliance:

- Prove funds are NOT from sanctioned addresses
- Prove tax compliance without revealing income
- Prove accredited investor status privately
- Selective disclosure to regulators



Key Insight:
ZK proofs can prove *absence* of problems (not on sanctions list) while maintaining privacy.

This approach is being explored for CBDC privacy as well (T4.4)

In the notebook, you will explore:

1. **Hash Commitments** – The building block of ZK
2. **Simple ZK Protocol** – Prove you know a number without revealing it
3. **Range Proofs** – Prove a value is in a range (like age ≥ 21)
4. **Merkle Proofs** – Prove membership without revealing the set

Learning Goal: Develop intuition for how ZK proofs work by implementing simple versions of the core concepts.

Note: Production ZK systems (SNARKs/STARKs) are much more complex, but the intuition from these exercises transfers directly.

Open NB16_zero_knowledge_intro.ipynb in Google Colab

NB16: What You'll Build

Exercise 1: Hash Commitment

```
# Commit to a secret
secret = "my_password"
commitment = hash(secret + random_nonce)
# Later: reveal secret + nonce
# Verifier: hash(secret + nonce) == commitment?
```

Exercise 2: Simple ZK Proof

```
# Prove you know x where hash(x) = H
# Without revealing x
# Interactive: challenge-response protocol
```

Exercise 3: Range Proof

```
# Prove: 18 <= age <= 100
# Without revealing exact age
# Uses bit decomposition + commitments
```

Key Observations:

1. Commitments hide data until reveal
2. Challenges ensure prover isn't cheating
3. Hash functions are the workhorse
4. Randomness is essential for ZK property

Time estimate: 45-60 minutes
Prerequisites: Python, basic understanding of hash functions

Libraries used:

- `hashlib` (hashing)
- `secrets` (randomness)

Discussion: Privacy vs. Regulation

The Fundamental Tension:

Case for Privacy:

- Financial privacy is a human right
- Surveillance chills legitimate activity
- Protects against discrimination
- Personal safety (wealth exposure)
- Competitive business interests

"If you have nothing to hide, you have nothing to fear" is a surveillance state argument

Case for Transparency:

- Prevents money laundering
- Tax compliance enforcement
- Sanctions effectiveness
- Counter-terrorism financing
- Consumer protection

"Complete privacy enables crime without consequence"

Discussion Questions

- Can ZK proofs thread the needle between these positions?
- Who decides what needs to be disclosed?
- Is "compliant privacy" an oxymoron or a solution?

Discussion: The Tornado Cash Case Study

What Happened:

- Tornado Cash: Ethereum privacy mixer
- Used ZK proofs to break transaction links
- August 2022: US Treasury sanctions
- Developer arrested in Netherlands
- Smart contract code itself sanctioned

The Debate:

- Is code speech? (First Amendment)
- Can open-source tools be sanctioned?
- Developer liability for user actions?
- Privacy tool vs. money laundering tool?

Key Statistics

- $7B+ total volume
- 30% allegedly illicit (Chainalysis)
- 70% legitimate privacy use
- Still operational (code is immutable)

Implications:

- Chilling effect on privacy development
- Precedent for sanctioning code
- Push toward compliant privacy solutions

Discussion: The Future of ZK Privacy

Where are we heading?

Possible Scenarios:

1. Compliant Privacy Wins

- ZK proofs with selective disclosure
- Regulators get access when needed
- Privacy by default, transparency by exception

2. Privacy Tech Restricted

- Heavy regulation of privacy tools
- KYC required for all crypto
- Privacy coins delisted

3. Parallel Systems

- Regulated "light" and unregulated "dark" finance
- Similar to offshore banking today
- Cat-and-mouse enforcement

4. ZK Becomes Standard

- All transactions private by default
- New compliance paradigms emerge
- Privacy seen as security feature

Key Question

Can we design systems where users have privacy, but bad actors can be identified? ZK proofs make this technically possible – the question is governance.

Executive Summary: Key Takeaways

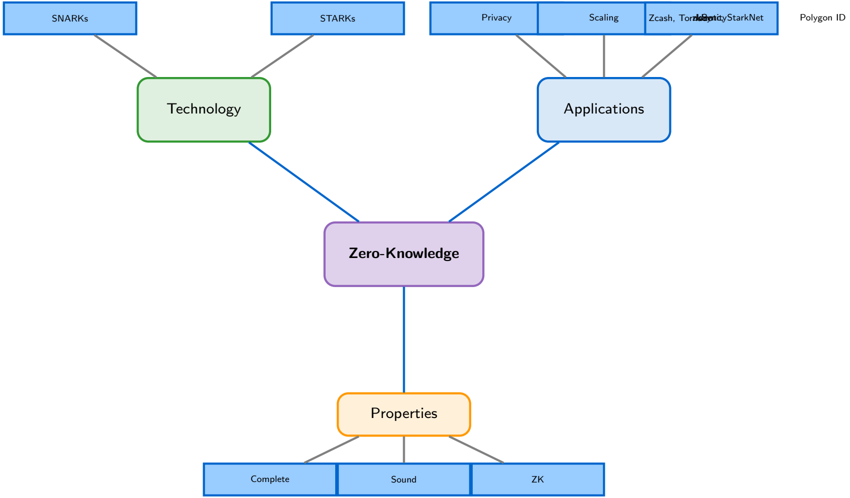
1. **Zero-knowledge proofs prove knowledge without revealing it**
   Enable privacy-preserving verification through cryptographic magic.

2. **Three essential properties: Completeness, Soundness, Zero-Knowledge**
   True claims succeed, false claims fail, no information leaks.

3. **SNARKs vs. STARKs trade off proof size for trust assumptions**
   SNARKs: tiny proofs, trusted setup. STARKs: larger proofs, transparent.

4. **Applications span privacy, scaling, and identity**
   Private transactions, ZK-rollups, and selective disclosure of credentials.

5. **ZK enables "compliant privacy" – a potential middle ground**
   Prove you're not a criminal without revealing your financial details.


SNARKs
STARKs
Privacy
Scaling
Zcash,
StarkNet
Polygon ID
Technology
Applications
Zero-Knowledge
Properties
Complete
Sound
ZK

Key Terms & Definitions (1/2)

Zero-Knowledge Proof A cryptographic method to prove a statement is true without revealing any information beyond the validity of the statement itself.

Prover The party who knows the secret and generates the proof.

Verifier The party who checks the proof without learning the secret.

Completeness Property ensuring that true statements can always be proven.

Soundness Property ensuring that false statements cannot be proven (except with negligible probability).

SNARK Succinct Non-interactive Argument of Knowledge – tiny proofs requiring trusted setup.

Key Terms & Definitions (2/2)

STARK Scalable Transparent Argument of Knowledge – larger proofs but no trusted setup and quantum-resistant.

Trusted Setup A ceremony to generate parameters for SNARKs; if compromised, allows forging proofs.

Fiat-Shamir Transform Technique to convert interactive proofs to non-interactive using hash functions.

ZK-Rollup Layer 2 scaling solution that uses ZK proofs to verify off-chain transactions on-chain.

Selective Disclosure Using ZK proofs to reveal only specific attributes (e.g., "over 21") without exposing underlying data.

Shielded Transaction A transaction where sender, receiver, and amount are hidden using ZK proofs.

Common Misconceptions

Myth	Reality
"ZK proofs hide everything"	ZK proofs hide **specific information** while proving **specific facts**. You choose what to reveal and what to hide.
"ZK is only for criminals"	ZK enables legitimate privacy (competitive trading, personal safety) and actually enables **compliant privacy** through selective disclosure.
"All ZK systems are the same"	SNARKs, STARKs, Bulletproofs, etc. have very different tradeoffs in proof size, speed, and trust assumptions.
"ZK proofs are slow"	Verification is extremely fast (milliseconds). Proof *generation* can be slow, but this happens off-chain.

Self-Assessment Questions (1/2)

Question 1: Which property of ZK proofs ensures that a cheating prover cannot convince the verifier of a false statement?

A. Completeness

B. Soundness

C. Zero-Knowledge

D. Transparency

Question 1: Which property of ZK proofs ensures that a cheating prover cannot convince the verifier of a false statement?

A. Completeness
B. Soundness
C. Zero-Knowledge
D. Transparency

Answer: B – Soundness
Explanation: Soundness is the property that ensures false statements cannot be proven. Completeness ensures true statements can be proven. Zero-knowledge ensures no extra information leaks. Transparency is a property of STARKs (no trusted setup), not a ZK proof property.

Self-Assessment Questions (2/2)

Question 2: What is the main advantage of STARKs over SNARKs?

A. Smaller proof sizes
B. No trusted setup required
C. Faster verification
D. Lower computational requirements

Question 2: What is the main advantage of STARKs over SNARKs?

A. Smaller proof sizes
B. No trusted setup required
C. Faster verification
D. Lower computational requirements

Answer: B – No trusted setup required
Explanation: STARKs are "transparent" – they don't require a trusted setup ceremony, eliminating the risk of compromised parameters. SNARKs actually have smaller proofs (A is wrong). Both have fast verification (C). STARKs often have higher computational requirements for proving (D is wrong).

Question 3: Name two applications of ZK proofs in finance.

Question 2: What is the main advantage of STARKs over SNARKs?

A. Smaller proof sizes
B. No trusted setup required
C. Faster verification
D. Lower computational requirements

Answer: B – No trusted setup required
Explanation: STARKs are "transparent" – they don't require a trusted setup ceremony, eliminating the risk of compromised parameters. SNARKs actually have smaller proofs (A is wrong). Both have fast verification (C). STARKs often have higher computational requirements for proving (D is wrong).

Question 3: Name two applications of ZK proofs in finance.
Possible Answers: Private transactions (Zcash), ZK-Rollups for scaling, identity verification, compliant privacy in DeFi, proof of solvency without revealing balances.

What's Next: Day 5 – Risk and Regulation

Connecting ZK to the broader picture:

Topics we'll cover:

- Crypto market risks and volatility
- Smart contract security
- Global regulatory landscape
- AML/KYC frameworks

ZK connections:

- How do regulators approach privacy tech?
- ZK audits and security considerations
- Compliant privacy implementations
- Future of privacy regulation

Key Question for Day 5

How do we build a regulatory framework that preserves innovation in privacy technology while preventing abuse? The Tornado Cash case will be revisited in the regulatory context.

Resources for Further Learning

Introductory:

- Vitalik Buterin: "An Incomplete Guide to Rollups" (vitalik.ca)
- ZK Podcast (zeroknowledge.fm)
- Electric Coin Co.: "What are zk-SNARKs?"

Technical Deep-Dives:

- zkSNARKs in a Nutshell (Eli Ben-Sasson)
- STARKs paper: "Scalable, transparent, and post-quantum secure computational integrity"
- Matter Labs: zkSync documentation

Interactive Learning:

- **ZK MOOC:** `zk-learning.org`
- **ZK Whiteboard Sessions:** YouTube (ZK Podcast)
- **Rareskills ZK Book:** `rareskills.io/zk-book`

NB16 provides hands-on practice with core ZK concepts

Questions?

Topic 4.5: Zero-Knowledge Technology in Finance [ADVANCED]

Privacy and Scalability Through Mathematical Proofs

Next: Day 5 – Risk and Regulation in Digital Finance

Additional: The ZK Project Landscape (2024)

Category	**Project**	**Technology**	**Status**
L2 Rollups	zkSync Era StarkNet Polygon zkEVM	SNARKs STARKs SNARKs	Mainnet Mainnet Mainnet
Privacy L1/L2	Zcash Aztec	SNARKs SNARKs	Mainnet (2016) Testnet
Identity	Polygon ID Worldcoin	SNARKs SNARKs	Mainnet Mainnet
DeFi	dYdX Penumbra	STARKs SNARKs	Mainnet Testnet

Market Observation: ZK technology has moved from research to production. Major projects are live with billions in TVL (Total Value Locked).

The ZK Era

2024 is called "The Year of ZK" – the technology is finally mature enough for mainstream blockchain adoption.