Hard Problems in Computing — S08E11: Zero-Knowledge Proofs

While last week we resolved the battle of Garbage Collection vs. Ownership, today we tackle a paradox: proving you know a secret without ever revealing the secret itself.

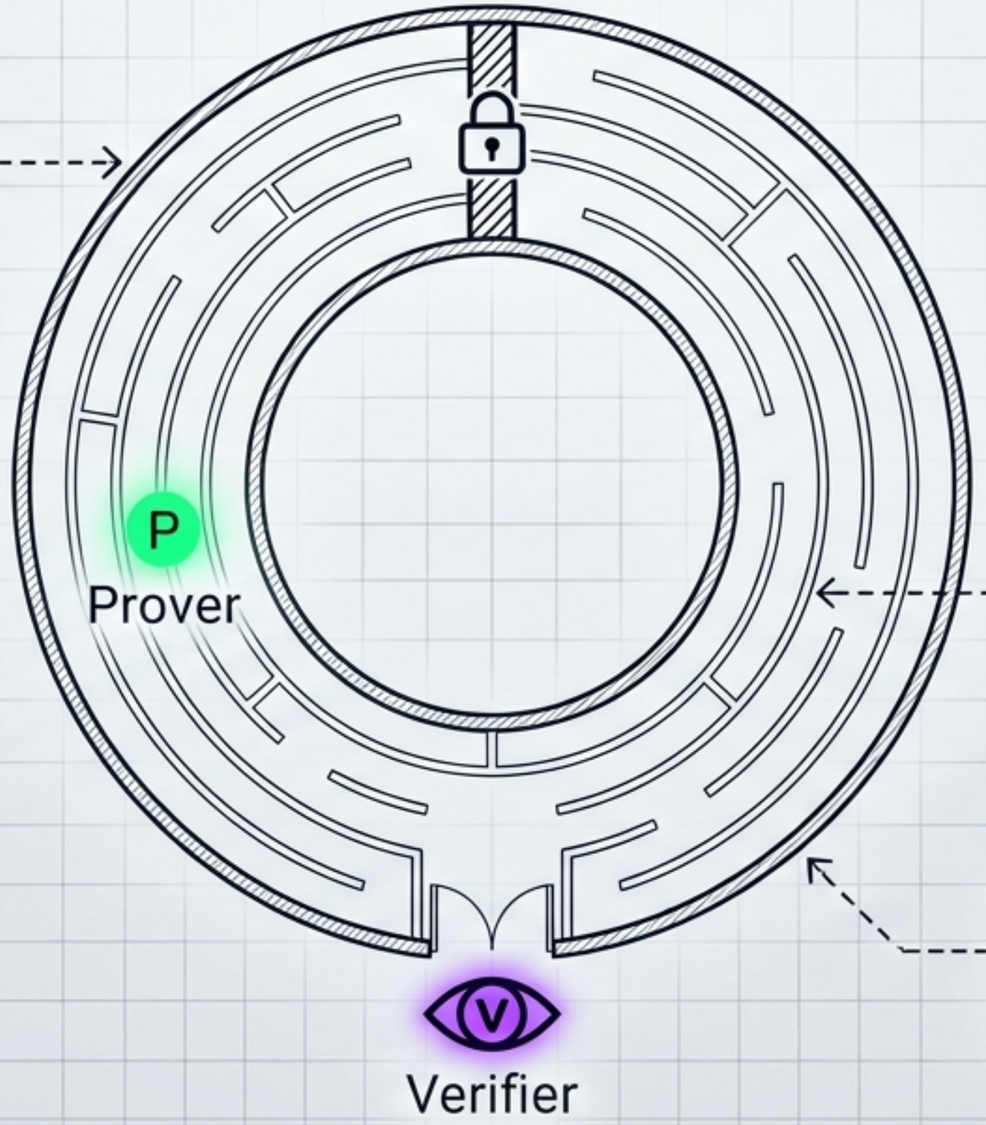


Gemini Notebook

The Origins of Zero-Knowledge

1985: Goldwasser, Micali, and Rackoff formally propose **interactive** proof systems

The Cave Metaphor: The Verifier never learns the magic word—they only observe the Prover consistently emerging from the correct path.



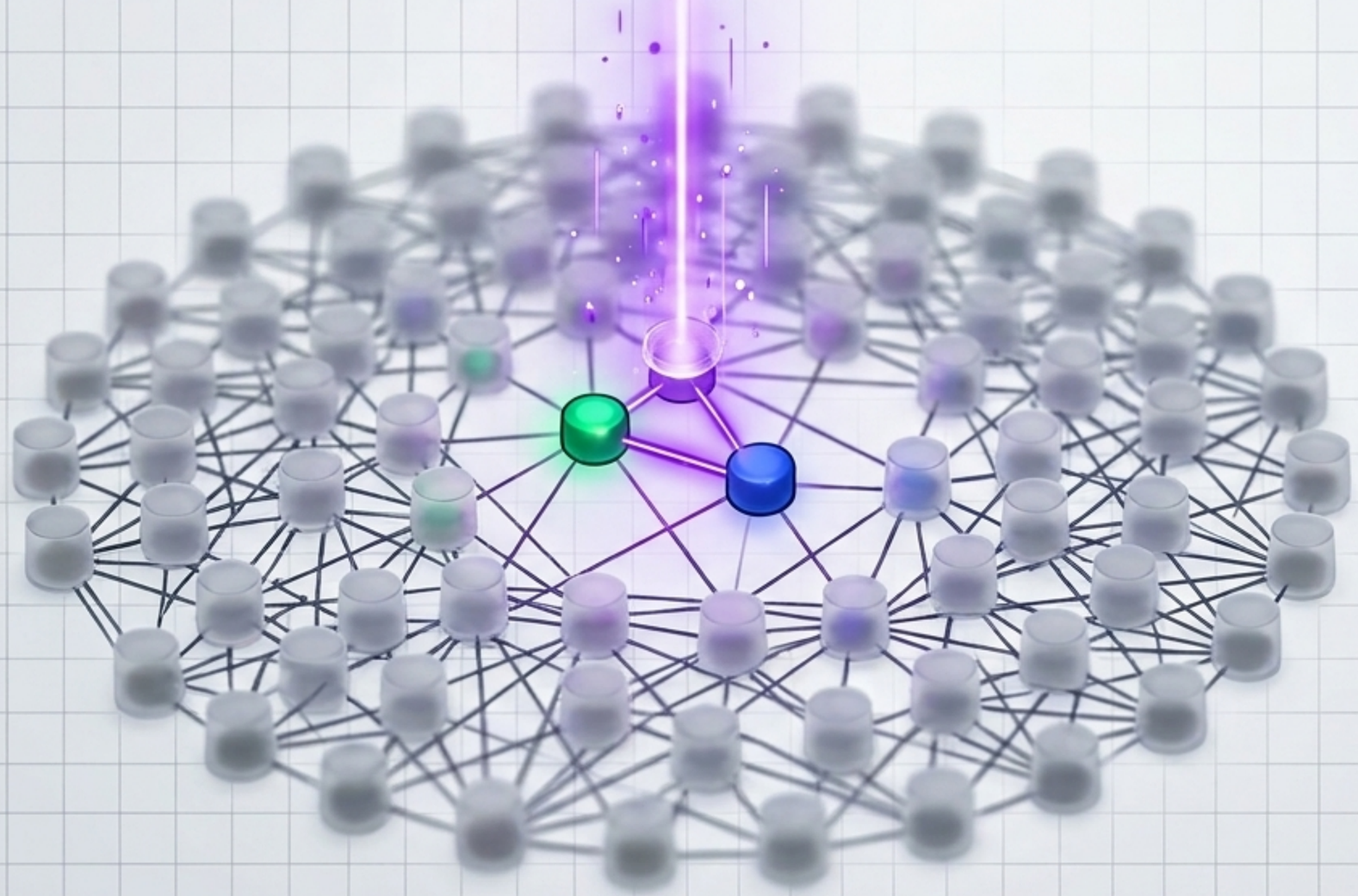
Interactive Rounds: The Verifier shouts a path (Left or Right).

Statistical Truth: With each round, the chance of a cheating Prover guessing the correct path halves.

Gemini Notebook

Matthew Green's 3-Colourable Graph

Cryptographic commitments for zero-knowledge proofs



The 3-Colourable Graph
Since graph 3-coloring is NP-complete, proving you have a valid coloring proves you can solve any problem in NP.

The Commitment Protocol:
The Prover commits to a randomized color scheme. The Verifier randomly challenges one edge.

The Math:
After E^2 rounds of interaction, the Verifier attains cryptographic certainty of the full map, without ever seeing enough connected nodes to reconstruct it.

Acknowledge this slide design inspirated from Protocol Pr
Gemini Notebook

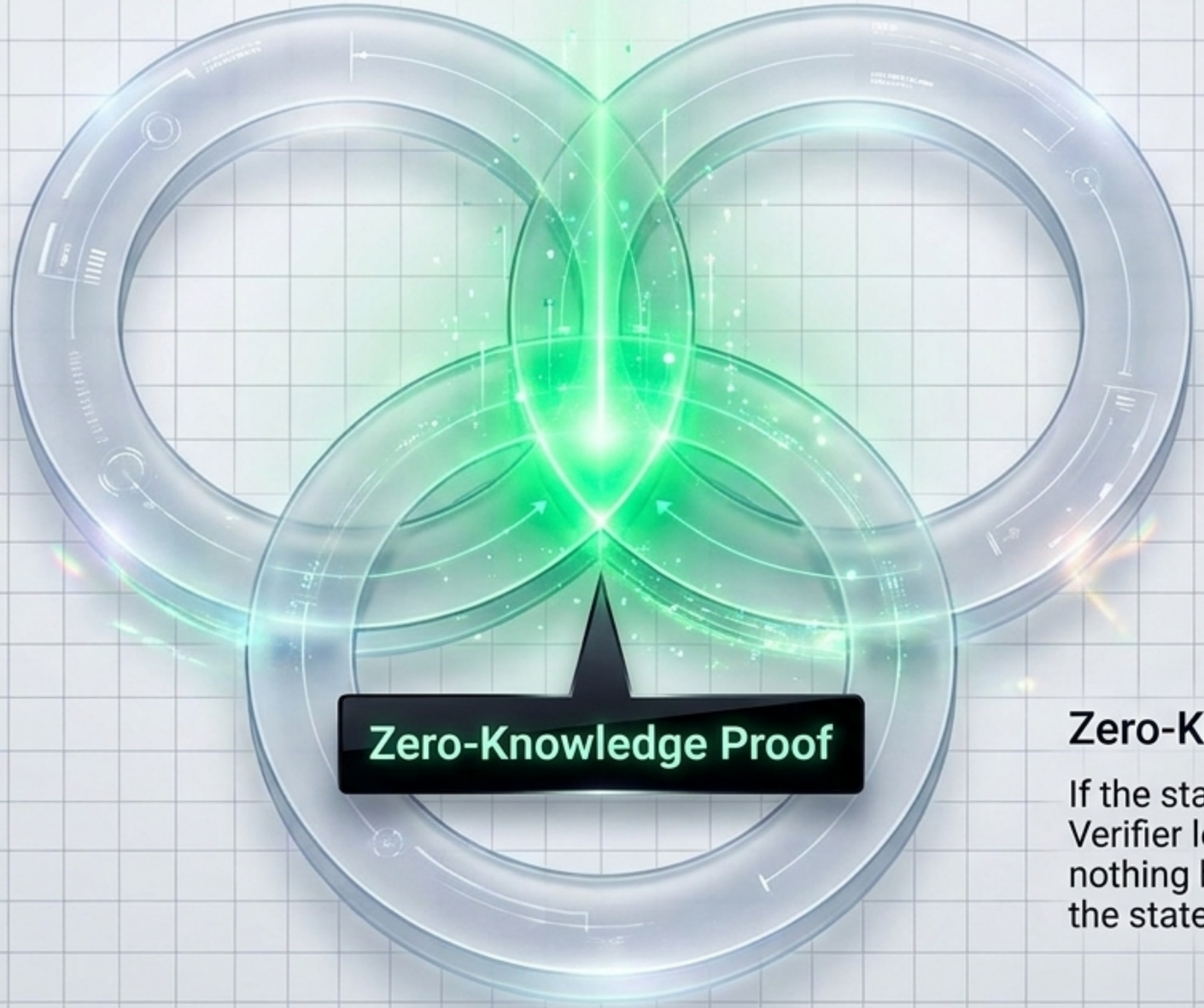
The Triad of ZKP

Completeness

If the statement is true, an honest Prover will always convince an honest Verifier.

Soundness

If the statement is false, no cheating Prover can convince the Verifier.



Zero-Knowledge

If the statement is true, the Verifier learns absolutely nothing beyond the fact that the statement is true.

Gemini Notebook

The Simulator Thought Experiment

Forward Time
(Soundness)



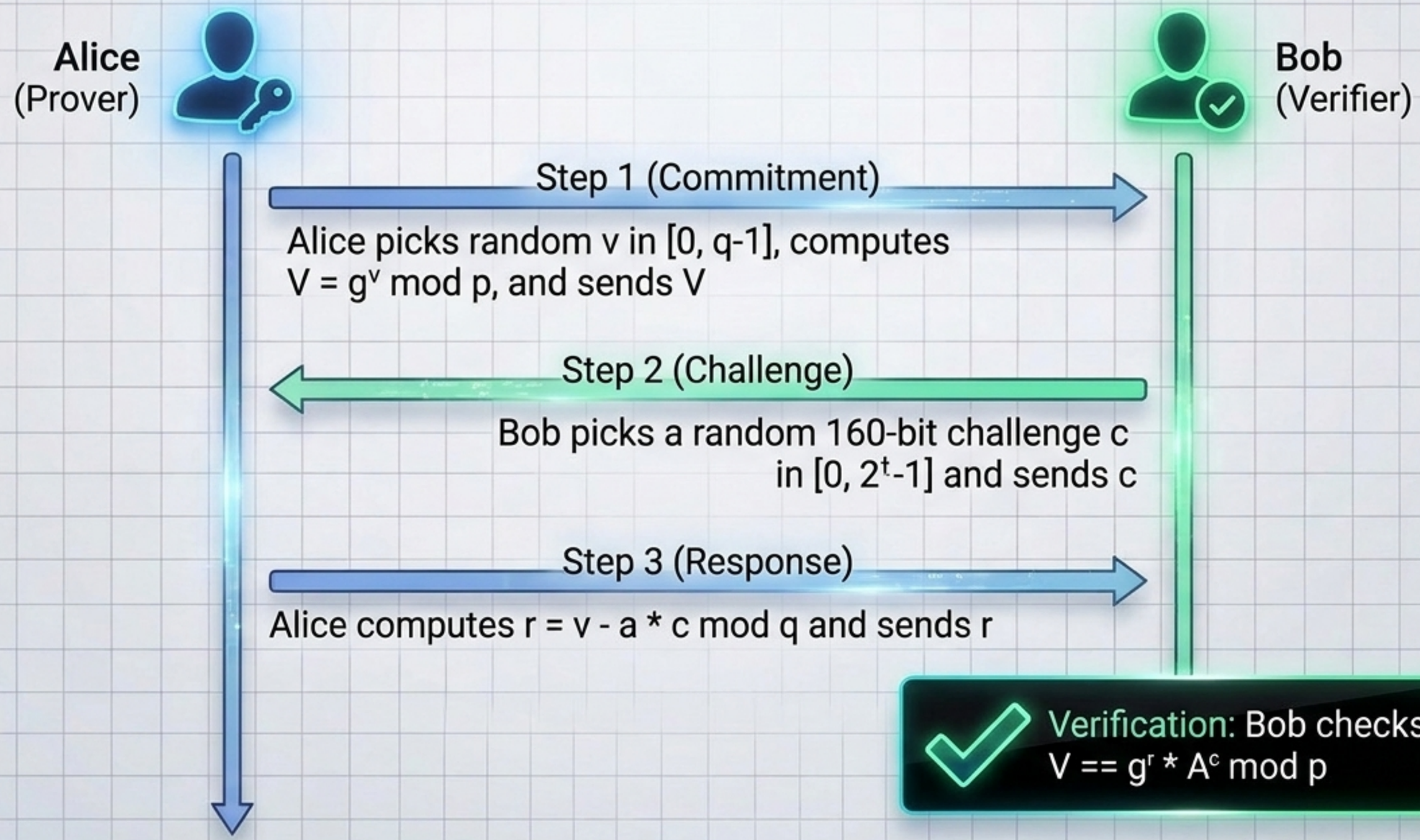
Rewound Time
(Zero-Knowledge)



The Thought Experiment: If a Simulator with no secret knowledge can use a rewind function to generate a transcript mathematically indistinguishable from a real proof...
...then the transcript itself leaks zero knowledge.
If a forgery is indistinguishable from reality, the reality contains no extractable secrets.

Gemini Notebook

Schnorr Identification (RFC 8235)



Gemini Notebook

The Fiat-Shamir Heuristic



Bob
(Verifier)

The Transformation

The Prover computes the challenge themselves by hashing the transcript: $c = H(g \| V \| A)$.

Cryptographic Hash Function (H)

The Result: The interaction collapses into a single, succinct, non-interactive proof payload that can be broadcasted and verified by anyone asynchronously.

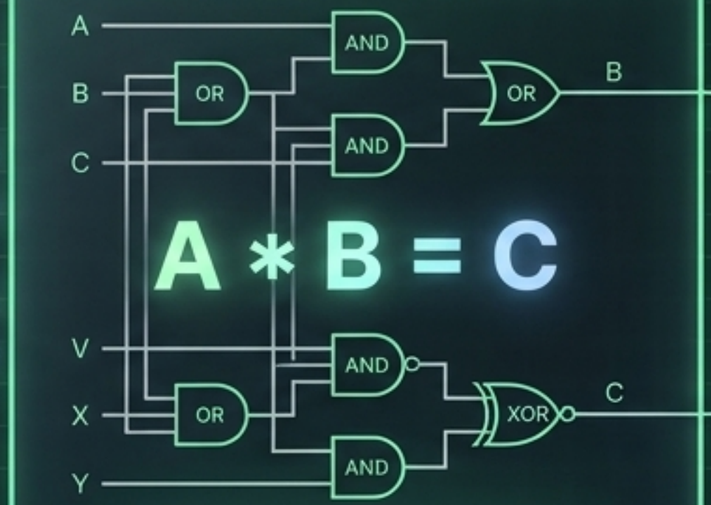
Gemini Notebook

Arithmetization: Flattening the Code

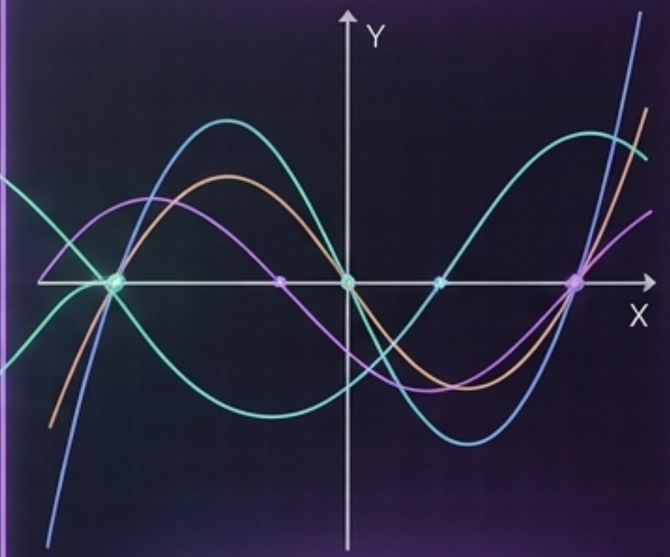
1. Standard Code

```
fn main() {
    let x = 5;
    let y = x * 2;
    println("{{}", y);
}
```

2. Rank-1 Constraint Systems (R1CS)



3. Polynomial Constraints (Plonkish / AIR)



Polynomials are single mathematical objects containing unbounded information. The equation $P(x) = Z(x) * H(x)$ compresses millions of computational steps into a single verifiable curve.

Gemini Notebook

Trusted Setups & Toxic Waste

The Powers of Tau

A Multi-Party Computation (MPC) ceremony to generate setup parameters.



Toxic Waste

The intermediate mathematical trapdoors. If even ONE participant destroys their entropy, the toxic waste is safely destroyed.

The Risk

If all participants collude and retain the toxic waste, they can forge infinite fake proofs (cryptographic counterfeiting).

Gemini Notebook

Proof-System Comparison Matrix

Proof System	Proof Size	Setup & Security	Math Base
zk-SNARKs (Groth16/PlonK)	Succinct (~200 Bytes)	Requires Trusted Setup. Vulnerable to Shor's.	Elliptic Curve Pairings
zk-STARKs (FRI-based)	Large (10s to 100s of KB)	Transparent (No Setup). Post-Quantum Secure.	Hash Functions / Polynomials
Halo 2	Moderate	Transparent. Vulnerable to Shor's.	Inner-product polynomial commitments

Gemini Notebook

Google Wallet & EU eIDAS (longfellow-zk)



The Problem

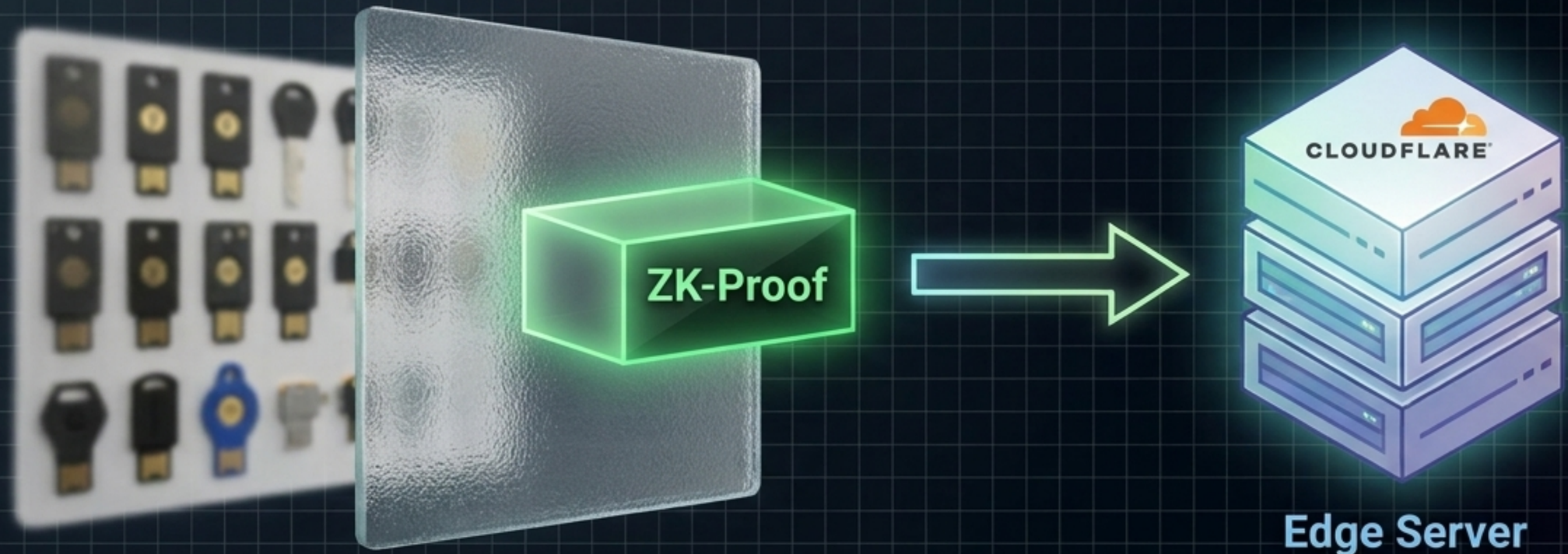
Handing over a digital ID leaks exact birthdates and creates linkable tracking vectors across services.

The ZK Solution

The client device evaluates the government-signed ID locally, generates a proof for the statement Age >= 18, and sends ONLY the proof. Complete compliance, zero linkability.

Gemini Notebook

Cloudflare Private FIDO Attestation



The Conflict

Enterprises must verify hardware keys are from approved vendors, but checking the exact batch/serial number allows cross-site user fingerprinting.

The ZK Solution

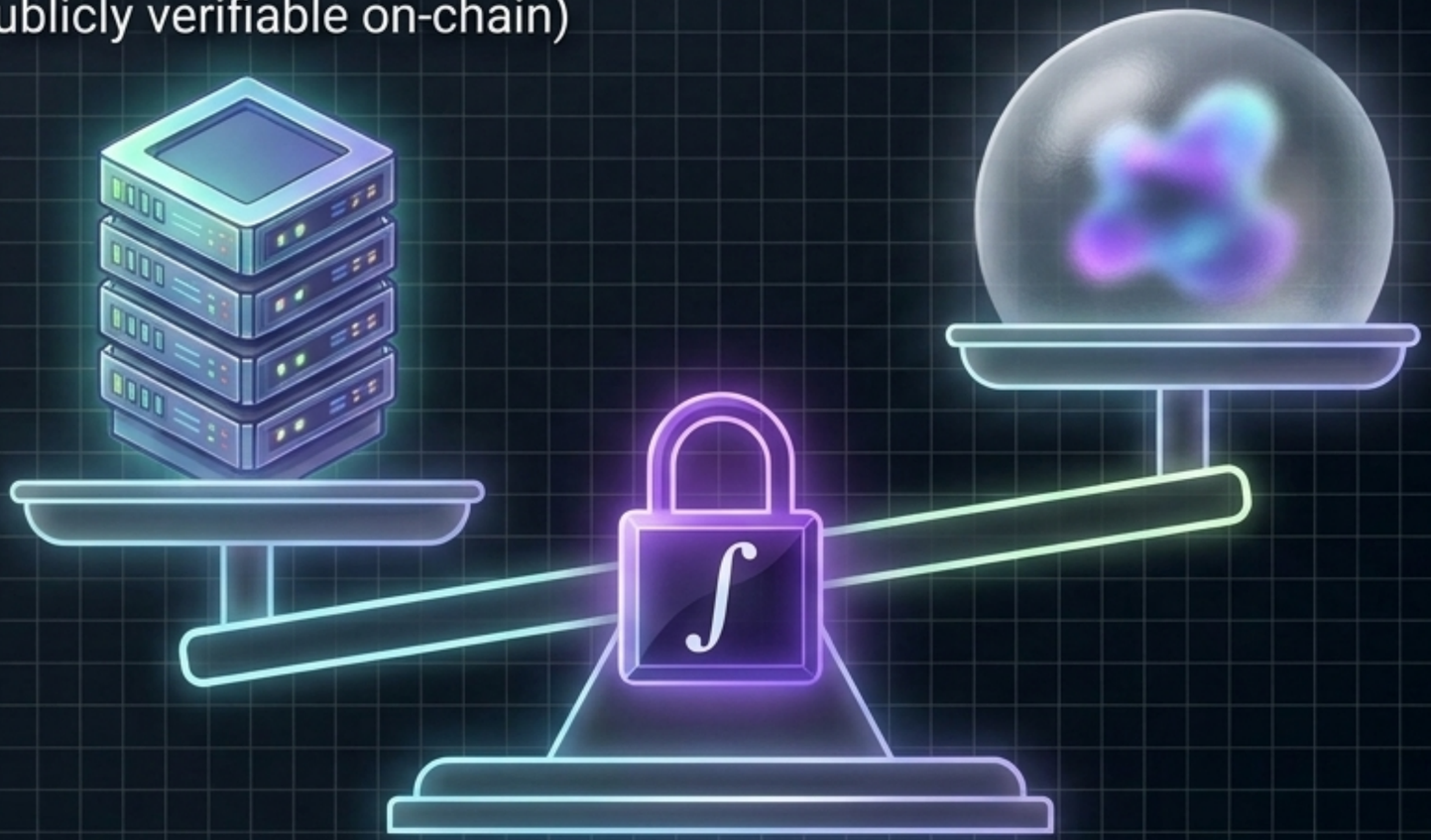
The browser generates a ZKP proving the key's certificate exists in a Merkle tree of approved vendor batches, blinding the specific hardware fingerprint.

Gemini Notebook

Solvency & Privacy: The Buterin Attestations

Cryptographic Assets
(Publicly verifiable on-chain)

User Liabilities
(Hidden)



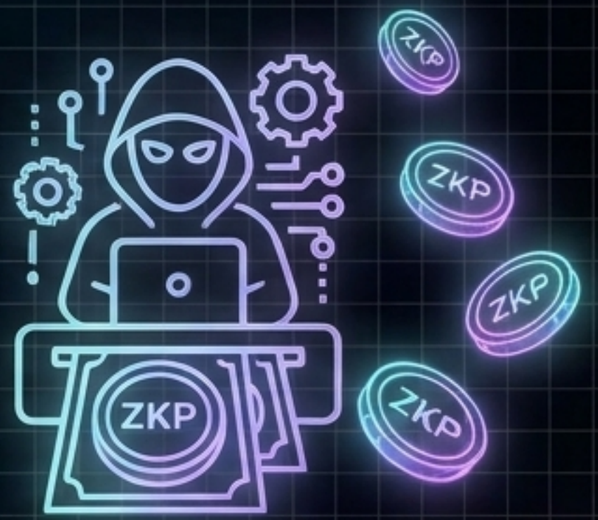
- **Solvency without Naive Transparency**
  - **The Dilemma:** Crypto exchanges must prove solvency to prevent bank runs, but publishing all user balances publicly violates financial privacy.
  - **The ZK Solution:** The exchange commits to a Merkle tree of all user balances. A ZK-SNARK proves that the sum of the tree (Liabilities) is strictly less than the known wallet addresses (Assets).

Gemini Notebook

Anatomy of Frozen Heart (Trail of Bits)

~~c = H(g || V || A)~~

c = H(g || [MISSING])



Malicious Prover

The Flaw:
In Bulletproofs/PlonK, implementations accidentally omitted the public inputs (commitments) from the Fiat-Shamir hash transcript.

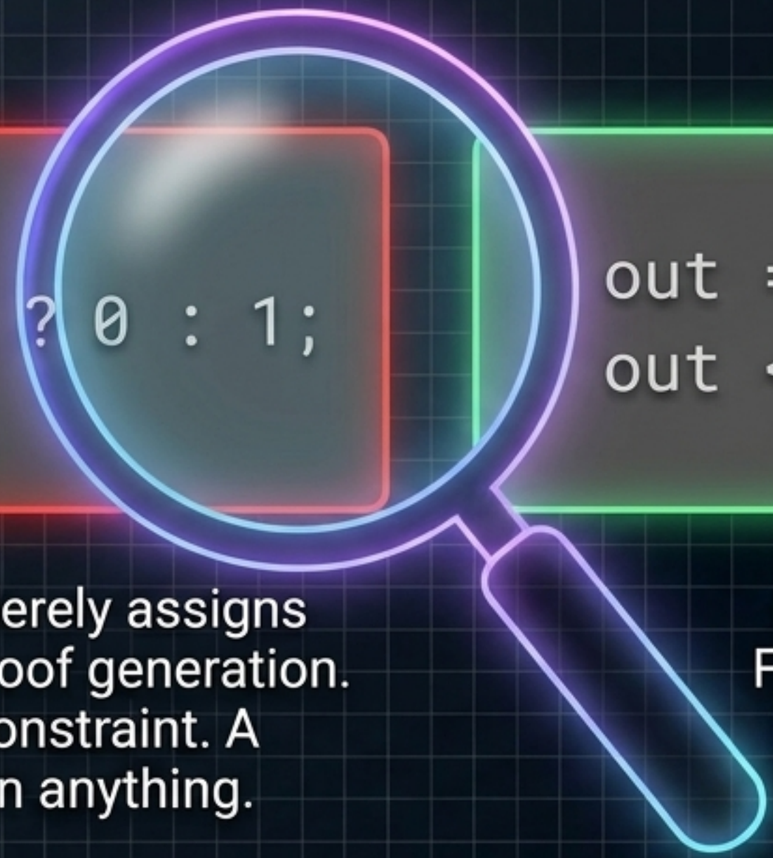
The Exploit: Because the challenge **c** is no longer bound to the prover's commitment, a malicious prover has the mathematical freedom to work backward, forging zero-knowledge proofs out of thin air.

Gemini Notebook

Under-Constrained Circuits (0xPARC Tracker #1)

```
temp <-- in != 0 ? 0 : 1;
```

```
out === temp;
out <== -in*temp + 1;
```



Witness Assignment (<--): Merely assigns a value to a variable during proof generation. It is NOT a mathematical constraint. A malicious prover can assign anything.

Constraint Enforcement (===): Forces the proof system to verify the mathematical relationship.

The Result: Confusing assignment for constraint allows attackers to submit successful proofs for entirely invalid logic.

Gemini Notebook

The Missing Bit-Range Check

The Vulnerability:
Smart contracts operate on 256-bit integers. SNARKs operate on ~254-bit prime fields.

The Exploit:
If a circuit lacks explicit bit-length constraints, an attacker can input a value larger than the scalar field, causing it to wrap around modulo P. This creates a duplicate valid proof for a spent nullifier (Double-Spend Attack).

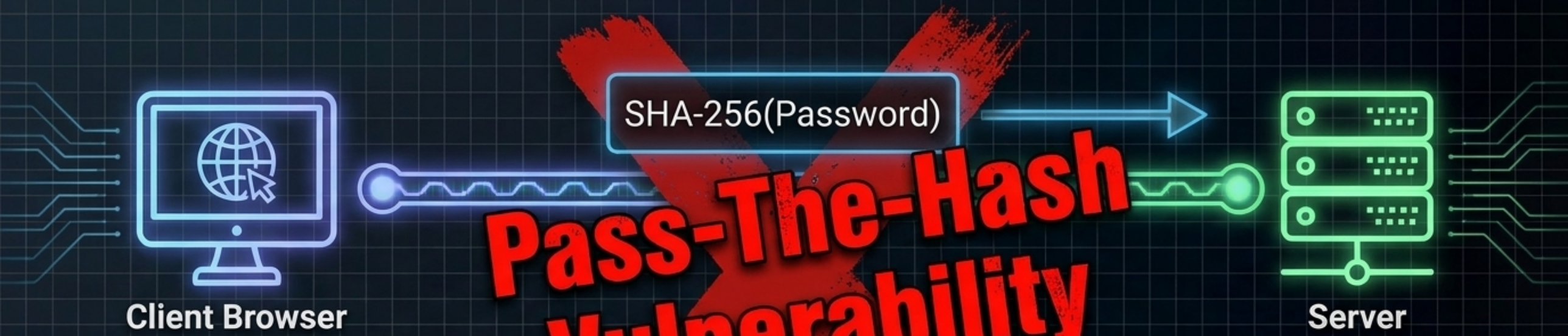


SNARK_SCALAR_FIELD
(~254 bits)

Mitigation:
Formal verification (Circomspect/ECNE) and strict ACIR typing in Noir.

Gemini Notebook

Disaster 1: The Marketing ZK Trap



Pass-The-Hash Vulnerability

The Myth:

If we hash the password on the client, the server never sees the plaintext! It's Zero-Knowledge Auth!

The Reality:

The hash simply becomes the new plaintext password. Anyone intercepting the hash can replay it to authenticate. This violates fundamental OWASP standards and provides exactly zero cryptographic proof of knowledge.

Gemini Notebook

Disaster 2: Homegrown Schnorr Math

```
v = Math.random() * q;
V = power(g, v) % p;
```



The Trap:

Prompting an LLM to build Schnorr identification in JS.

The Catastrophe:

Interactive protocols require cryptographically secure randomness. If the RNG is biased by even a single bit, or if a nonce is reused across two challenges, an attacker can instantly extract the private key via basic algebra: a = (r - r') / (c' - c) mod q.

Gemini Notebook

Disaster 3: Unverified Circuits

Vibe-Coded



Developer

.circom

```
.cirom =
%oed0677929431
apevnotation.a
poeminmas to
repirets to
recoiwmend:
25665697867839
excare:.handéia
rooepk.icon1
.circomt = 85
```



Smart Contract

Engineering

.circom

```
.ciron =
%ead0778920421
aoevnotation.a
noeiminmea ts
repizets to
recoumwend:
25965667867839
eacare.bendéia
recept.icons
.circomt = 86
```

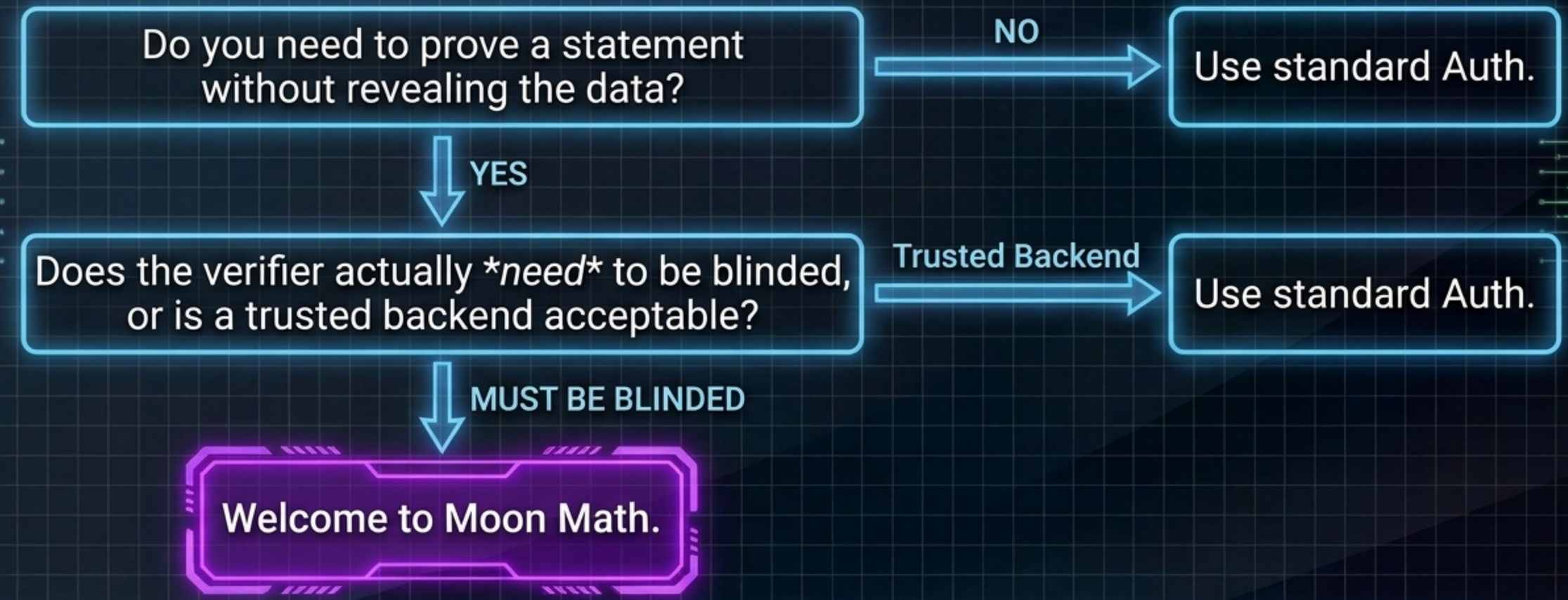


Circomspect Picus ECNE Manual Audit

Zero-Knowledge circuits are not traditional software; they are hardware logic designs mapped to polynomials. Deploying an under-constrained circuit is equivalent to deploying a bank vault with a hologram for a door. It looks secure until a malicious prover walks right through it.

Gemini Notebook

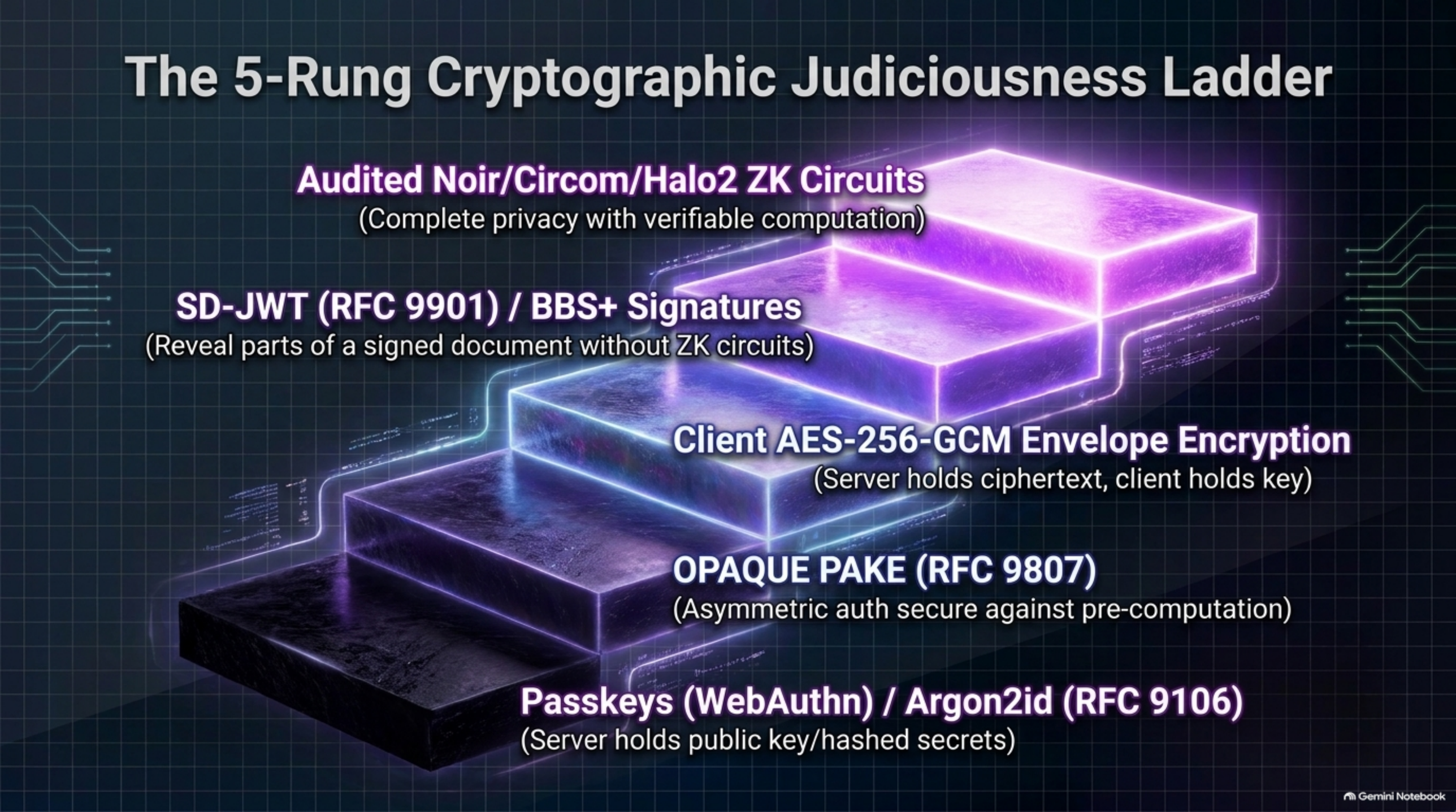
YAGNI & Trust Boundaries



ZKPs are computationally expensive, brutally difficult to audit, and unforgiving of errors. Never use a **ZK-SNARK** when an **asymmetric keypair** and a **trusted** server boundary will suffice.

Gemini Notebook

The 5-Rung Cryptographic Judiciousness Ladder



Gemini Notebook

Next Week — S08E12: The Halting Problem & Undecidability

```
ERROR: Execution cannot be determined.
```

Series Finale Preview

We've mastered memory, ownership, and cryptographic truth.
But some problems refuse to be solved.
The absolute limit of computation as we know it.

Gemini Notebook