zenrock

02/03/2026

# Zenrock Comprehensive Litepaper

*Decentralized Custody, Privacy and More*

**Zenrock is on a mission to expand and grow what is possible onchain with products and services users actually want. Our technology stack and deep cryptographic and cross-chain expertise allow us to venture into areas previously untapped.**

**$ROCK is the value capture mechanism for the Zenrock ecosystem, including all products developed by Zenrock Labs or by third parties leveraging Zenrock's infrastructure.**

## Overview

- **Zenrock:** A permissionless ecosystem supported by Zenrock Labs and the Zenrock Foundation.
- **zrChain:** A purpose-built Layer 1 for distributed multi-party computation (dMPC). It is flexible infrastructure powering everything in the Zenrock ecosystem
- **Decentralized Custody Tokens (DCTs):** Cross-chain assets secured by dMPC, eliminating centralized vaults and single points of failure. The first two DCTs, zenBTC and zenZEC, are live on Solana
- **Hush Protocol:** A Zcash inspired privacy layer on Solana. Shield assets, transfer privately, earn yield, and withdraw anywhere without exposing your wallets
- **Future Products:** dMPC unlocks use cases across DeFi, privacy, identity, key management, and beyond. Zenrock enables the development of products and services that solve real problems in DeFi and cloud
- **$ROCK benefits from ecosystem growth:** Every product built on zrChain, by Zenrock Labs or anyone else, flows revenue through the same tokenomics. More products, more fees, more value to $ROCK

[1] This Litepaper reflects the plans and expectations of its authors as of the date listed on the first page. Plans and circumstances can change, so we cannot assure that it will be accurate as to future events, developments, or plans, and it should not be relied on with respect to any such events, developments, or plans.

# zrChain

zrChain is a Delegated Proof-of-Stake chain built on the Cosmos SDK, purpose-built to run distributed multi-party computation with enshrined oracles that enable secure and complex cross-chain operations.

At the core of Zenrock's dMPC is a threshold signature scheme (TSS). To produce a valid signature, multiple parties exchange key fragments, but the private key is never assembled. No participant, including Zenrock, ever has enough information to reconstruct it. **The key doesn't exist in any single location because it was never created in full. Only mathematical fragments exist.**

This allows zrChain to control wallets on any third-party blockchain without any party ever knowing the underlying private key.

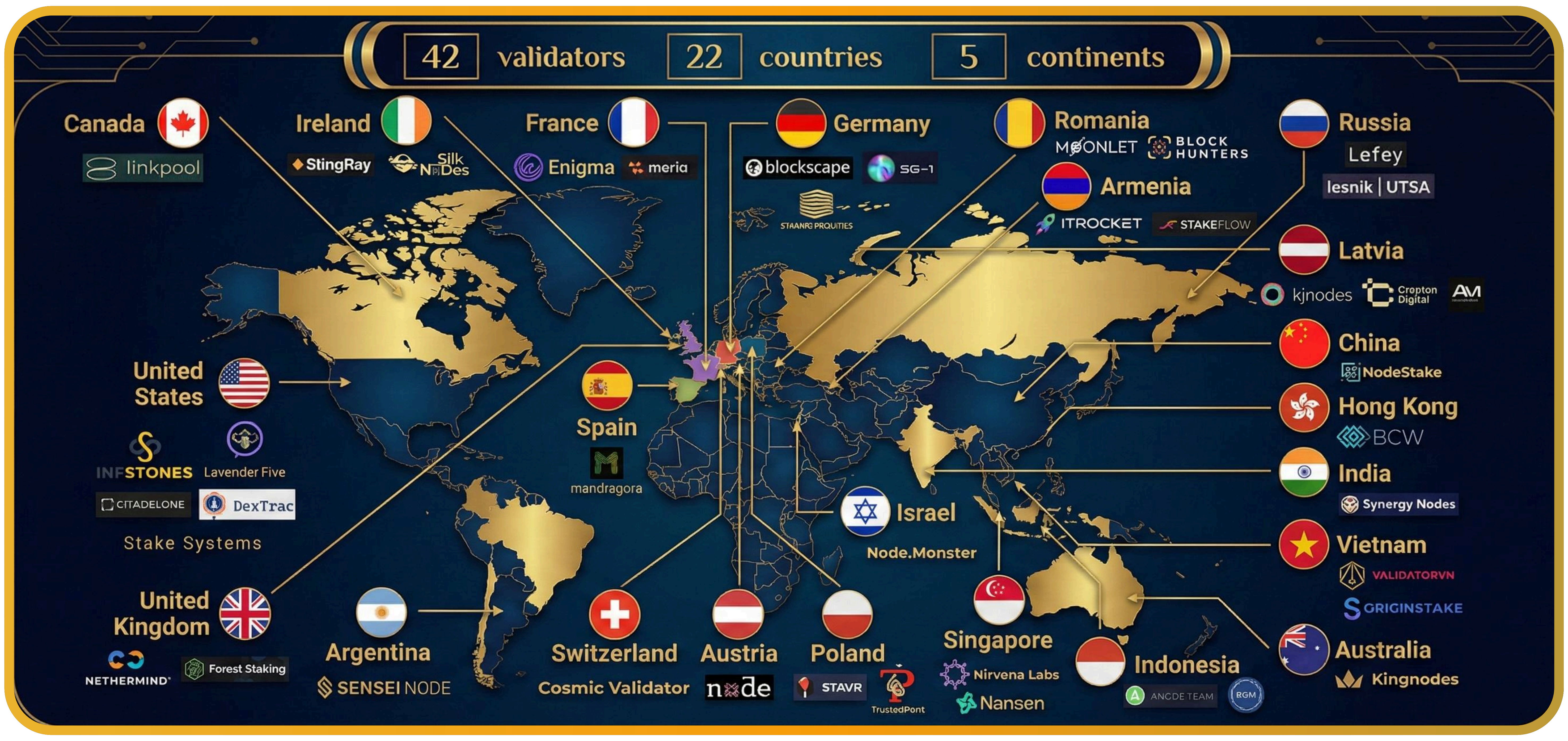
**dMPC signature instructions can only originate from zrChain, making the entire signing process decentralized.** Through Cosmos Vote Extensions (a mechanism allowing validators to include additional data in their consensus votes), each validator runs an enshrined oracle sidecar as part of their node.

This sidecar monitors events on external chains (for example, a deposit event on Bitcoin) and submits them directly into zrChain's core consensus process. Validators collectively observe the event, reach consensus, and only then does the chain instruct the dMPC to act.

**By enshrining the oracle mechanism directly into core consensus, zrChain ensures that as long as the chain itself is secure, every instruction passed to the dMPC is legitimate.**

zrChain launched with a genesis set of 8 dMPC operators and 60+ validators.

These infrastructure providers possess significant cumulative stake across 50+ chains with geographic distribution across 22+ countries. Over time, the dMPC operator set will expand to 16, then 32, each time increasing the cryptographic threshold and security guarantees.



# Distributed Multi-Party Computation with Secure Enclaves

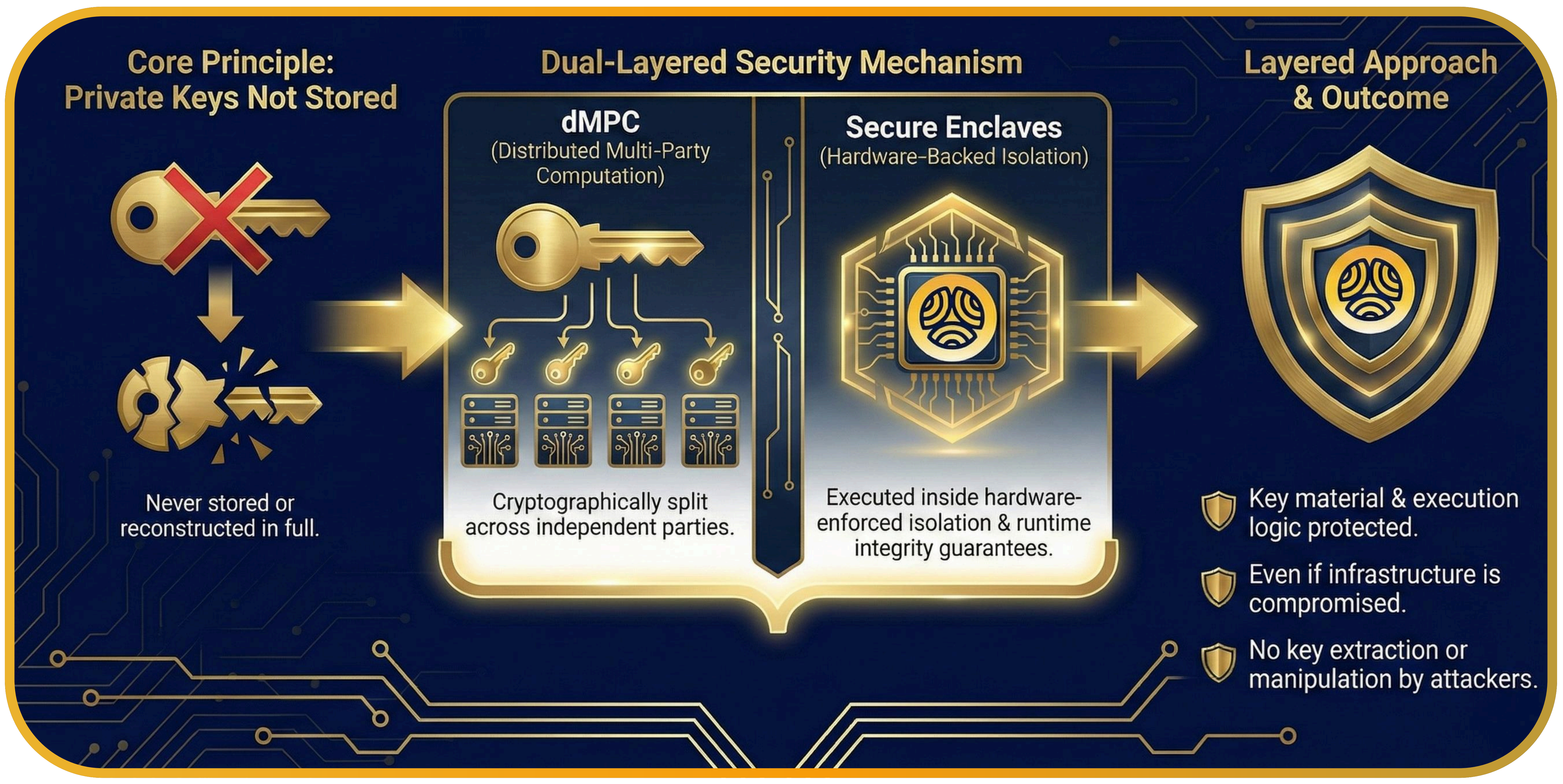
Our system combines distributed Multi-Party Computation (dMPC) with hardware-backed secure enclaves to provide strong cryptographic security, operational resilience, and verifiable trust — without relying on a single trusted machine or operator.

## Core Principle

Private keys are never stored or reconstructed in full. Instead, they are:

- Cryptographically split across multiple independent parties using MPC
- Executed inside secure enclaves, which provide hardware-enforced isolation and runtime integrity guarantees

This layered approach ensures that both key material and execution logic remain protected, even in the presence of compromised infrastructure. Even if an attacker gains control over parts of the hosting environment, they cannot extract private keys or manipulate signing behavior, because key material is cryptographically distributed and all signing operations are confined to attested, hardware-isolated environments.


Core Principle: Private Keys Not Stored
Never stored or reconstructed in full.
Dual-Layered Security Mechanism
dMPC
(Distributed Multi-Party Computation)
Cryptographically split across independent parties.
Secure Enclaves
(Hardware-Backed Isolation)
Executed inside hardware-enforced isolation & runtime integrity guarantees.
Layered Approach & Outcome
Key material & execution logic protected.
Even if infrastructure is compromised.
No key extraction or manipulation by attackers.

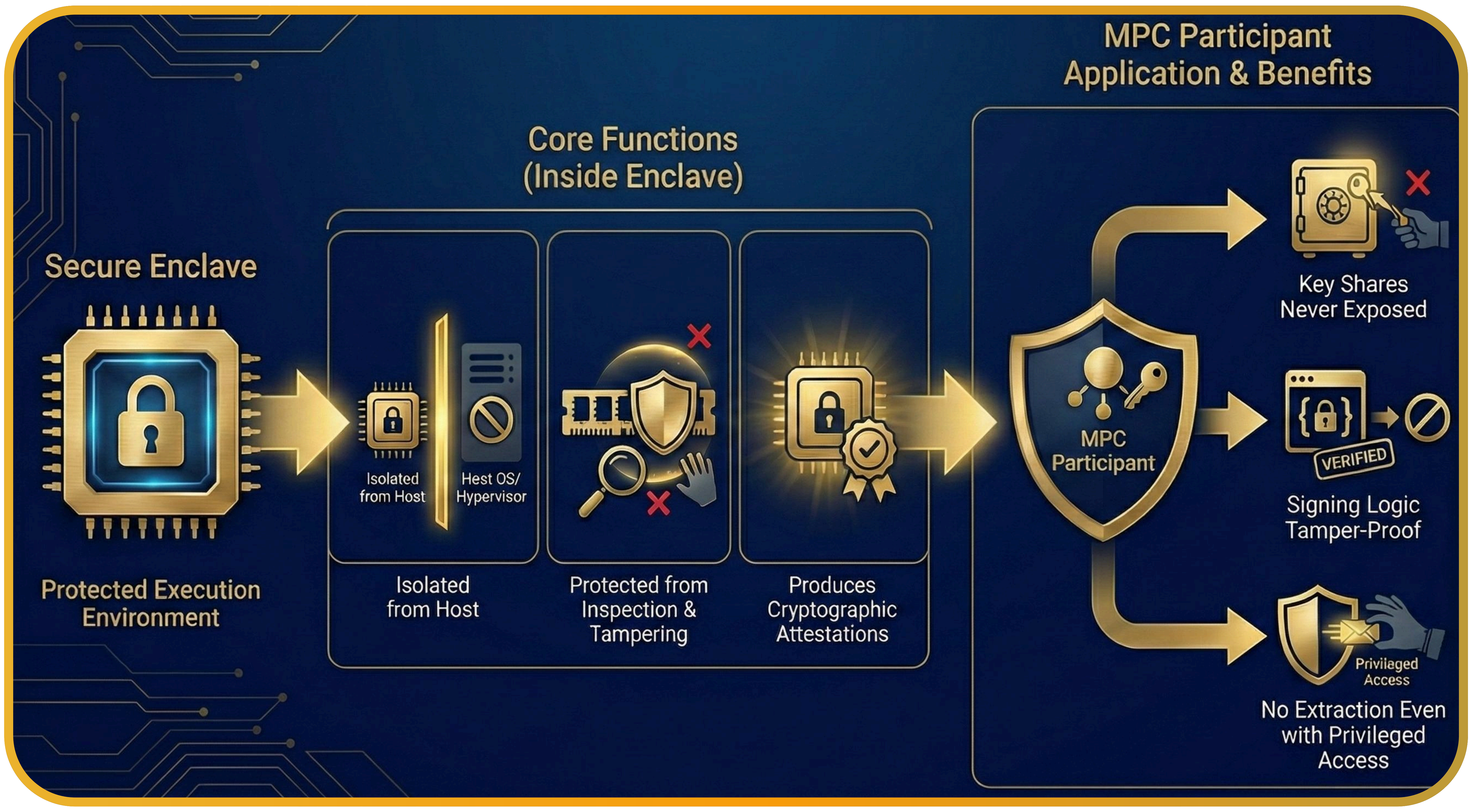
## Role of Secure Enclaves

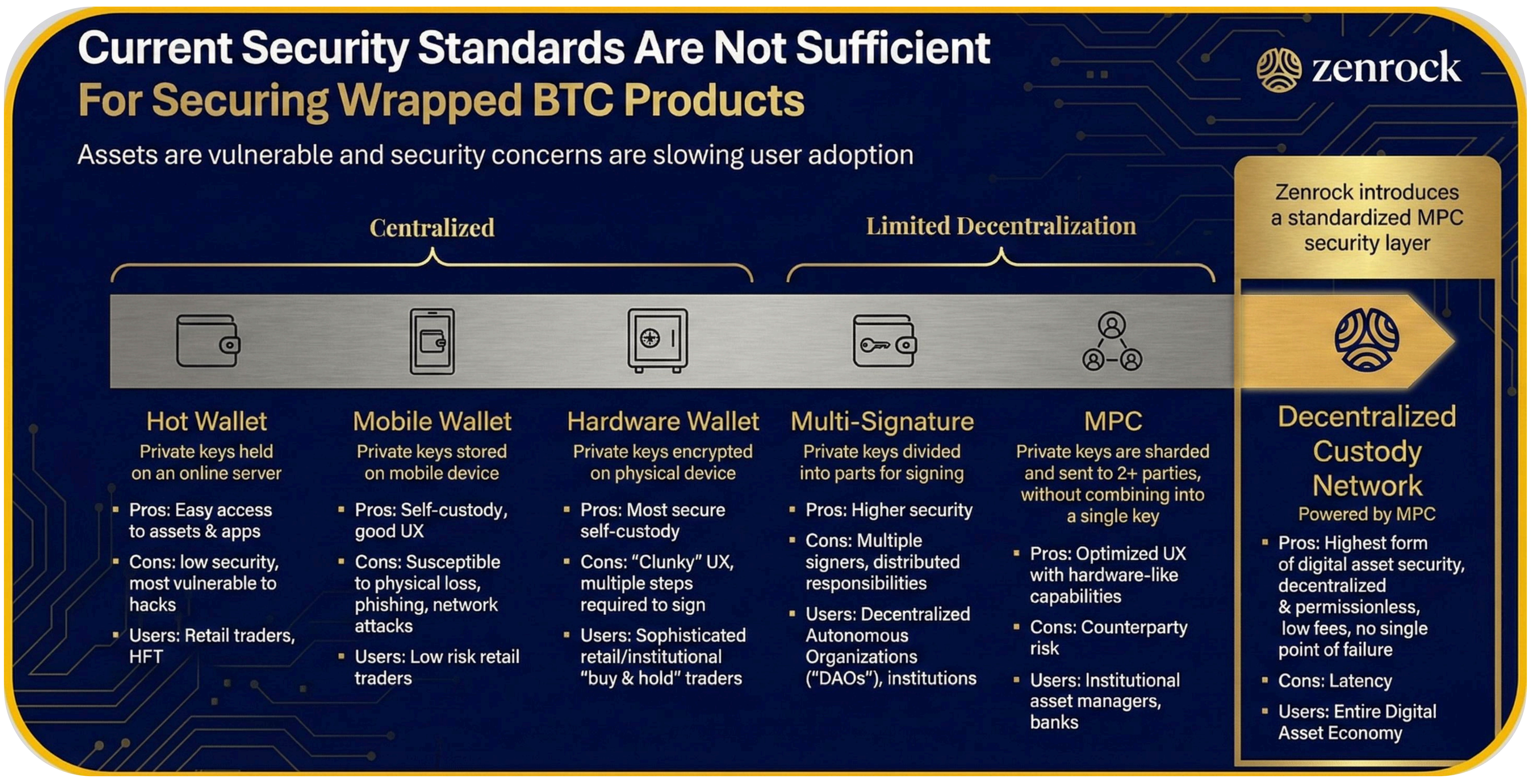
A secure enclave is a protected execution environment provided by modern hardware. Code running inside an enclave is:

- Isolated from the host operating system and hypervisor
- Protected from memory inspection or tampering
- Able to produce cryptographic attestations proving what code is running

In our architecture, each MPC participant runs inside its own enclave, ensuring that:

- Key shares are never exposed to the host machine
- Signing logic cannot be modified without detection
- Even privileged system access cannot extract sensitive material


Secure Enclave
Protected Execution Environment
Core Functions (Inside Enclave)
Isolated from Host
Hest OS/ Hypervisor
Isolated from Host
Protected from Inspection & Tampering
Produces Cryptographic Attestations
MPC Participant Application & Benefits
MPC Participant
Key Shares Never Exposed
VERIFIED
Signing Logic Tamper-Proof
Privileged Access
No Extraction Even with Privileged Access



**Current Security Standards Are Not Sufficient For Securing Wrapped BTC Products**

Assets are vulnerable and security concerns are slowing user adoption

zenrock

Centralized

Limited Decentralization

Zenrock introduces a standardized MPC security layer

**Hot Wallet**
Private keys held on an online server
- Pros: Easy access to assets & apps
- Cons: low security, most vulnerable to hacks
- Users: Retail traders, HFT

**Mobile Wallet**
Private keys stored on mobile device
- Pros: Self-custody, good UX
- Cons: Susceptible to physical loss, phishing, network attacks
- Users: Low risk retail traders

**Hardware Wallet**
Private keys encrypted on physical device
- Pros: Most secure self-custody
- Cons: "Clunky" UX, multiple steps required to sign
- Users: Sophisticated retail/institutional "buy & hold" traders

**Multi-Signature**
Private keys divided into parts for signing
- Pros: Higher security
- Cons: Multiple signers, distributed responsibilities
- Users: Decentralized Autonomous Organizations ("DAOs"), institutions

**MPC**
Private keys are sharded and sent to 2+ parties, without combining into a single key
- Pros: Optimized UX with hardware-like capabilities
- Cons: Counterparty risk
- Users: Institutional asset managers, banks

**Decentralized Custody Network**
Powered by MPC
- Pros: Highest form of digital asset security, decentralized & permissionless, low fees, no single point of failure
- Cons: Latency
- Users: Entire Digital Asset Economy

## dMPC Signing Flow with Enclaves

**Distributed Key Generation:** The private key is generated collaboratively by multiple enclave-resident participants using an MPC protocol. Each enclave receives a unique key share. The full private key never exists anywhere.

**Attested Execution:** Each enclave can produce a signed attestation proving the exact MPC code it is running and that the code is executing inside genuine enclave hardware. Only attested enclaves are permitted to participate.

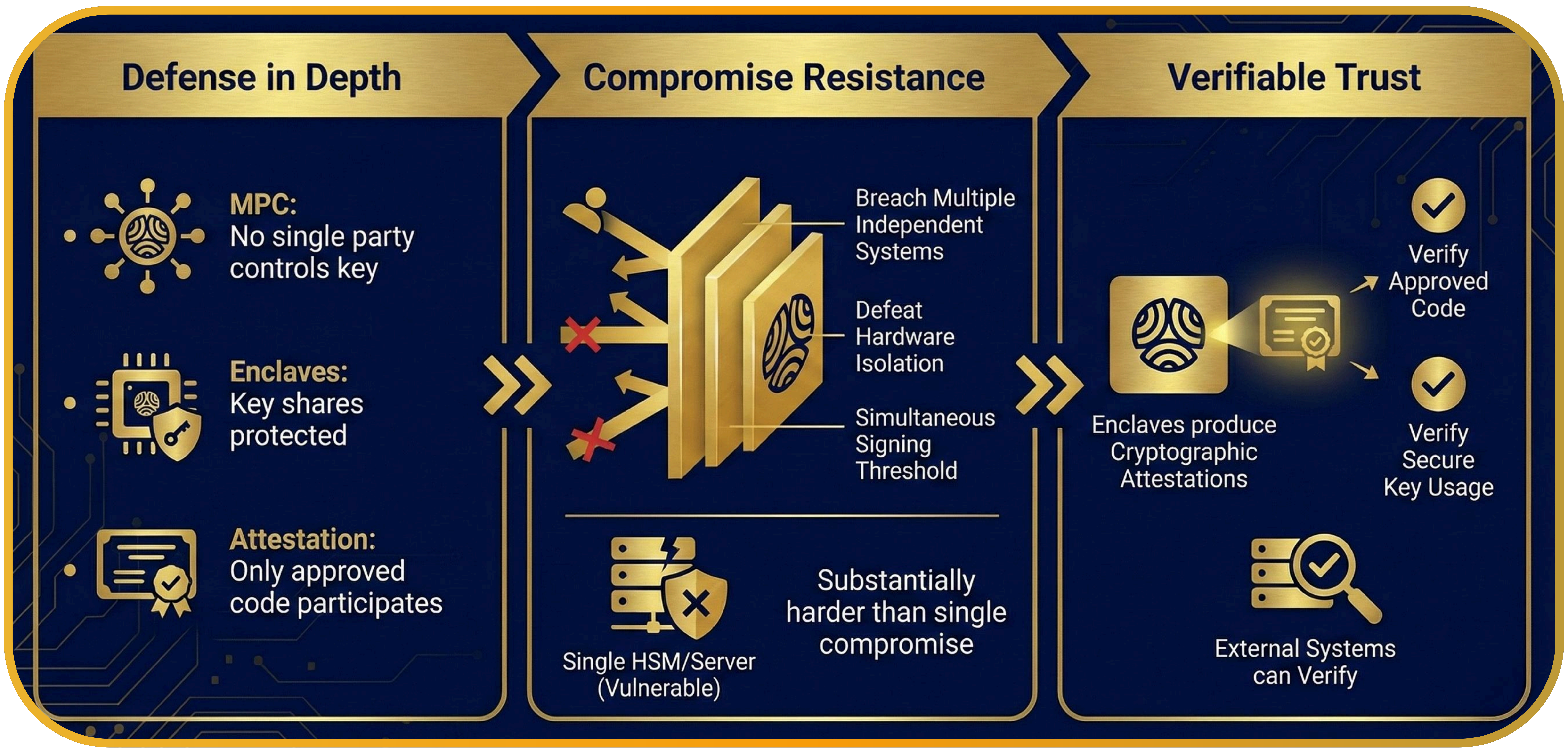
**Isolated Key Storage:** Key shares are encrypted inside their respective enclaves and cannot be accessed by the host system, administrators or cloud provider. They can only be accessed inside the enclave.

**Signing Request:** When a transaction needs to be signed, a request is sent to the required number of enclaves.

**Distributed Enclave Signing:** Each enclave computes a partial signature using its internal key share. These partial signatures reveal no information about the private key.

**Signature Assembly:** The partial signatures are combined into a standard blockchain signature, indistinguishable from one produced by a single private key.

**At no point is the private key reconstructed — inside or outside an enclave.**


Defense in Depth
MPC:
No single party controls key
Enclaves:
Key shares protected
Attestation:
Only approved code participates
Compromise Resistance
Breach Multiple Independent Systems
Defeat Hardware Isolation
Simultaneous Signing Threshold
Single HSM/Server (Vulnerable)
Substantially harder than single compromise
Verifiable Trust
Enclaves produce Cryptographic Attestations
Verify Approved Code
Verify Secure Key Usage
External Systems can Verify

## Security Features

**Defense in Depth:**

- MPC prevents any single party from controlling a key
- Enclaves prevent key shares from being extracted or tampered with
- Attestation ensures only approved code can participate

**Compromise Resistance:** An attacker would need to breach multiple independent systems, defeat hardware-level isolation, and do so simultaneously within a signing threshold. This is substantially harder than compromising a single HSM, server, or operator.

**Verifiable Trust:** Because enclaves produce cryptographic attestations, external systems can verify that signing operations originate from approved, unmodified code and that keys are being used only within expected security boundaries.

## Operational Flexibility

- Threshold policies (e.g. M-of-N signers)
- Node rotation and upgrades without changing public keys
- Geographic and administrative separation of trust
- Integration with onchain authorization and governance logic

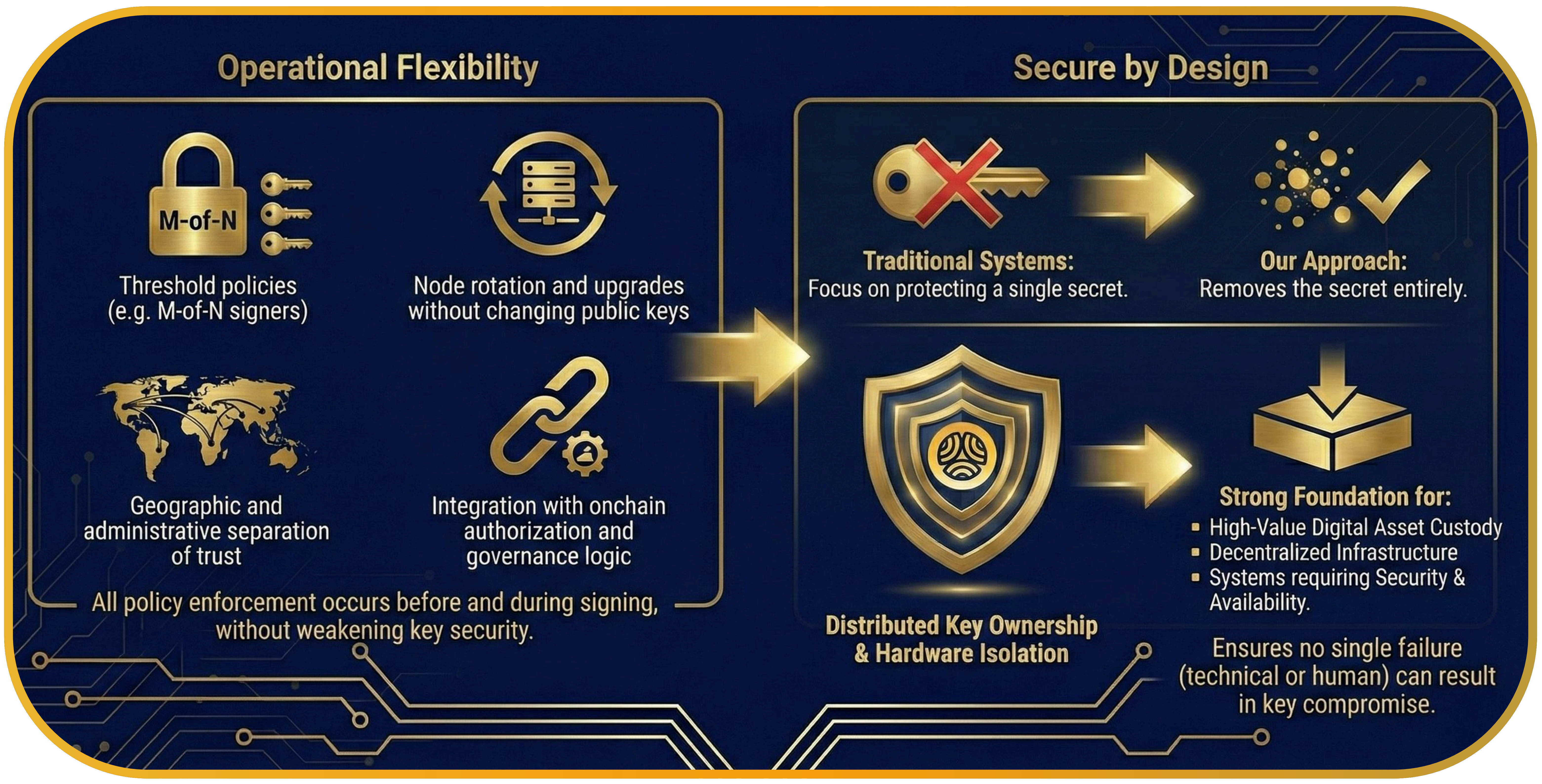
All policy enforcement occurs before and during signing, without weakening key security.

## Secure by Design

Traditional systems focus on protecting a single secret. **Our approach removes the secret entirely.**

**By distributing key ownership cryptographically and enforcing execution integrity through hardware isolation, the system ensures that no single failure (technical or human) can result in key compromise.**

This makes MPC with secure enclaves a strong foundation for high-value digital asset custody, decentralized infrastructure, and systems requiring both security and availability.


Operational Flexibility
M-of-N
Threshold policies (e.g. M-of-N signers)
Node rotation and upgrades without changing public keys
Geographic and administrative separation of trust
Integration with onchain authorization and governance logic
All policy enforcement occurs before and during signing, without weakening key security.
Secure by Design
Traditional Systems: Focus on protecting a single secret.
Our Approach: Removes the secret entirely.
Distributed Key Ownership & Hardware Isolation
Strong Foundation for:
High-Value Digital Asset Custody
Decentralized Infrastructure
Systems requiring Security & Availability.
Ensures no single failure (technical or human) can result in key compromise.

## zrChain: Technical Architecture

### Consensus & Validator Architecture

zrChain uses Delegated Proof-of-Stake with a hybrid validation model. The validator set includes both standard validators running consensus and a specialized subset of dMPC operators. This separation allows the network to scale the validator count for decentralization while maintaining a controlled set of cryptographic key holders for security-critical operations.

### dMPC Integration

The treasury module manages all interactions with the MPC infrastructure. Signing requests follow this flow:

1. A module (zenBTC, DCT, or Hush) creates a SignTransactionRequest specifying the key ID and unsigned transaction

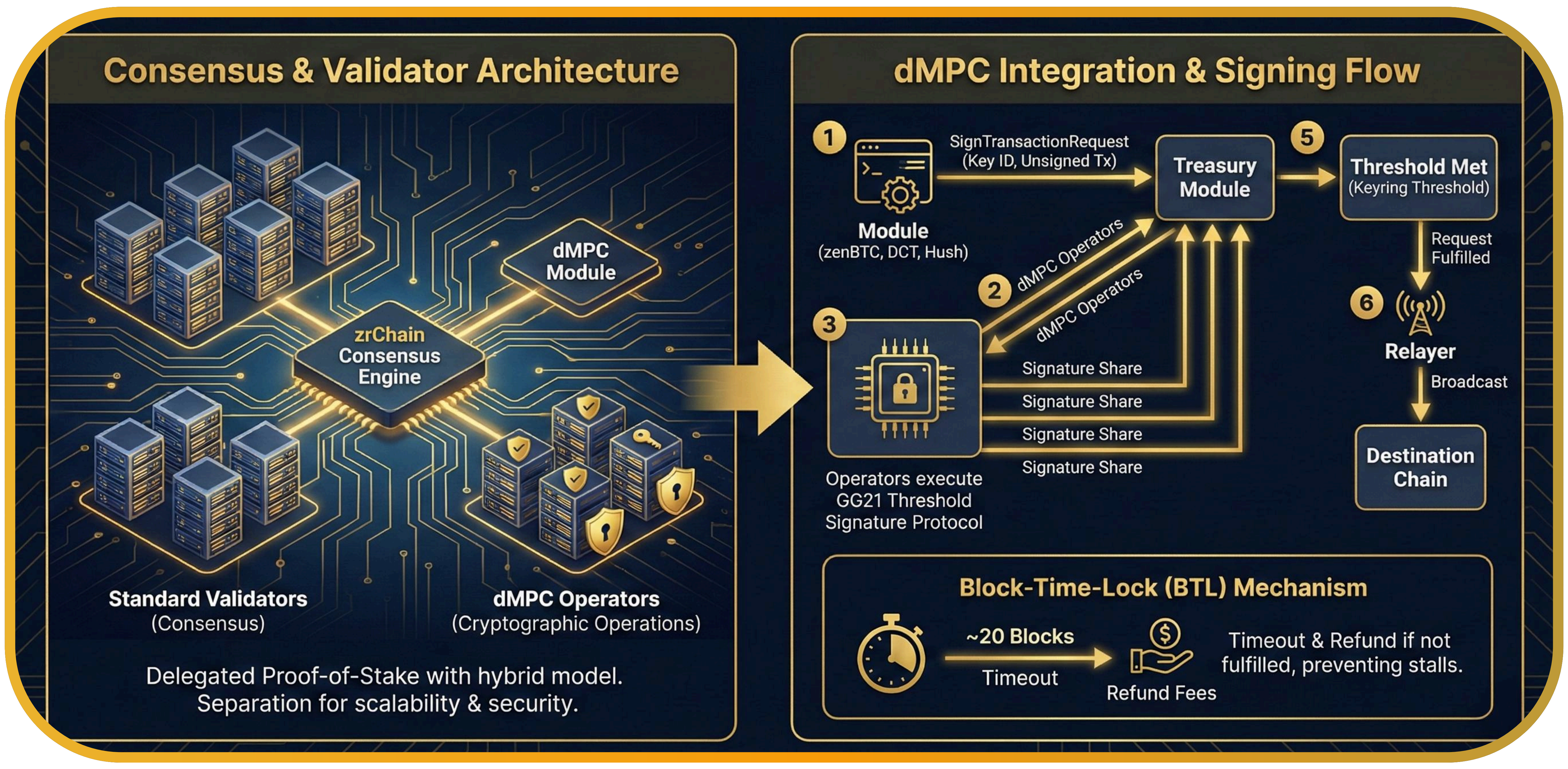
2. dMPC operators poll zrChain for pending requests

3. Operators execute the GG21 threshold signature protocol (detailed in the previous section)

4. Each operator submits their signature share to zrChain

5. Once sufficient signatures arrive (meeting the keyring threshold), the request is fulfilled

6. A relayer broadcasts the signed transaction to the destination chain


Consensus & Validator Architecture
dMPC Module
zrChain Consensus Engine
Standard Validators (Consensus)
dMPC Operators (Cryptographic Operations)
Delegated Proof-of-Stake with hybrid model. Separation for scalability & security.
dMPC Integration & Signing Flow
1
SignTransactionRequest (Key ID, Unsigned Tx)
Module (zenBTC, DCT, Hush)
Treasury Module
5
Threshold Met (Keyring Threshold)
Request Fulfilled
2
dMPC Operators
dMPC Operators
3
Signature Share
Signature Share
Signature Share
Signature Share
Operators execute GG21 Threshold Signature Protocol
6
Relayer
Broadcast
Destination Chain
Block-Time-Lock (BTL) Mechanism
~20 Blocks
Timeout
Refund Fees
Timeout & Refund if not fulfilled, preventing stalls.

## Enshrined Oracle via Vote Extensions

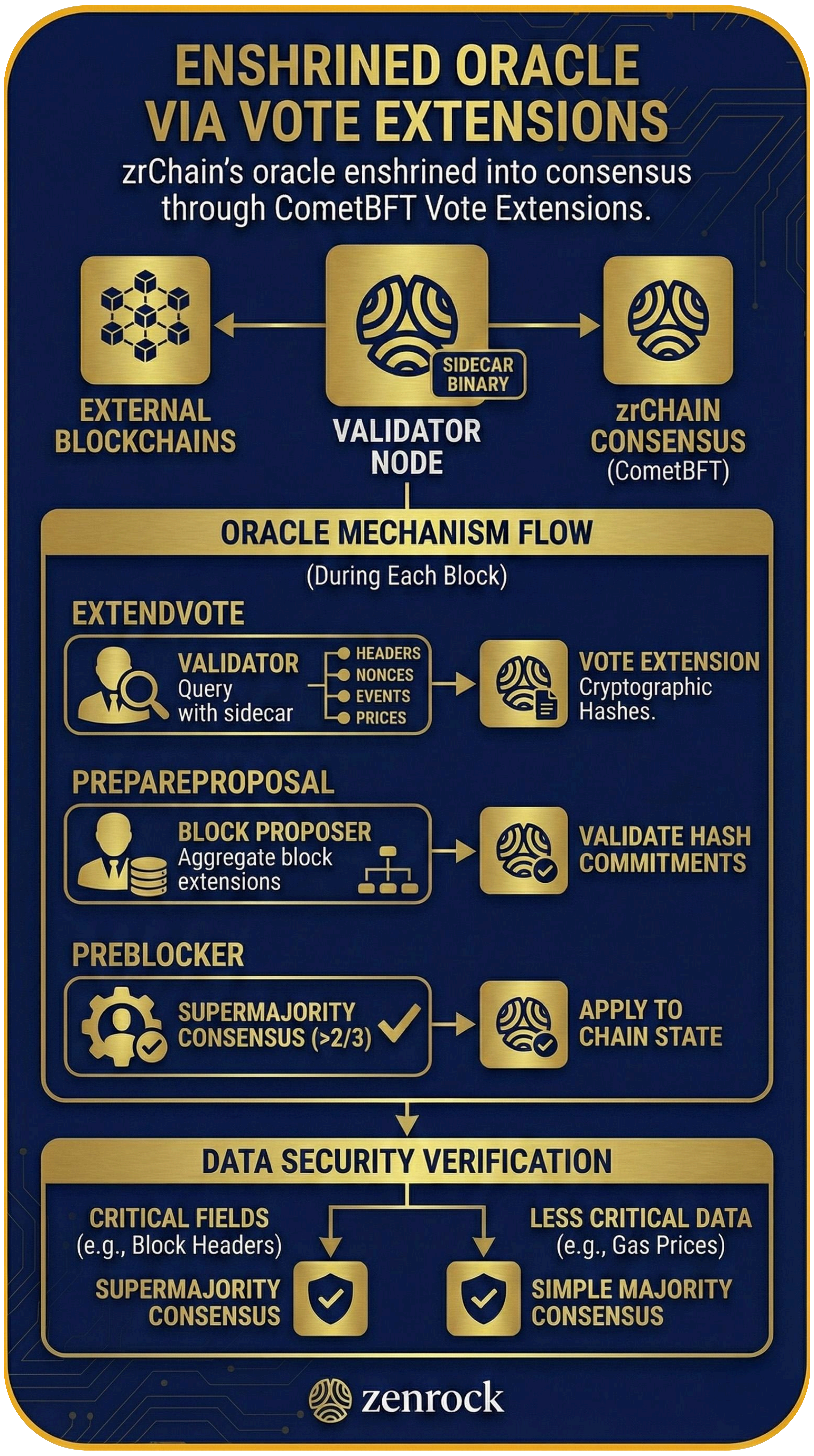
zrChain's oracle mechanism is enshrined directly into consensus through CometBFT Vote Extensions. Each validator runs a sidecar binary that monitors external blockchains and feeds data into the zrChain consensus process to be voted on.

During each block:

1. **ExtendVote:** Validators query their sidecar for the latest external chain state (block headers, account nonces, events, prices) and include cryptographic hashes in their vote extension

2. **PrepareProposal:** The block proposer aggregates vote extensions and validates that hash commitments match the actual data

3. **PreBlocker:** Only data achieving supermajority consensus (>2/3 voting power) is applied to chain state


ENSHRINED ORACLE VIA VOTE EXTENSIONS
zrChain's oracle enshrined into consensus through CometBFT Vote Extensions.
EXTERNAL BLOCKCHAINS
SIDECAR BINARY
VALIDATOR NODE
zrCHAIN CONSENSUS (CometBFT)
ORACLE MECHANISM FLOW
(During Each Block)
EXTENDVOTE
VALIDATOR Query with sidecar
HEADERS
NONCES
EVENTS
PRICES
VOTE EXTENSION Cryptographic Hashes.
PREPAREPROPOSAL
BLOCK PROPOSER Aggregate block extensions
VALIDATE HASH COMMITMENTS
PREBLOCKER
SUPERMAJORITY CONSENSUS (>2/3)
APPLY TO CHAIN STATE
DATA SECURITY VERIFICATION
CRITICAL FIELDS (e.g., Block Headers)
SUPERMAJORITY CONSENSUS
LESS CRITICAL DATA (e.g., Gas Prices)
SIMPLE MAJORITY CONSENSUS
zenrock

This design ensures that external chain data is verified by the majority of validators before being used. Critical fields like block headers require supermajority consensus, while less critical data like gas prices can proceed with simple majority.

## Cross-Chain Observation

The sidecar oracle monitors multiple blockchains simultaneously:

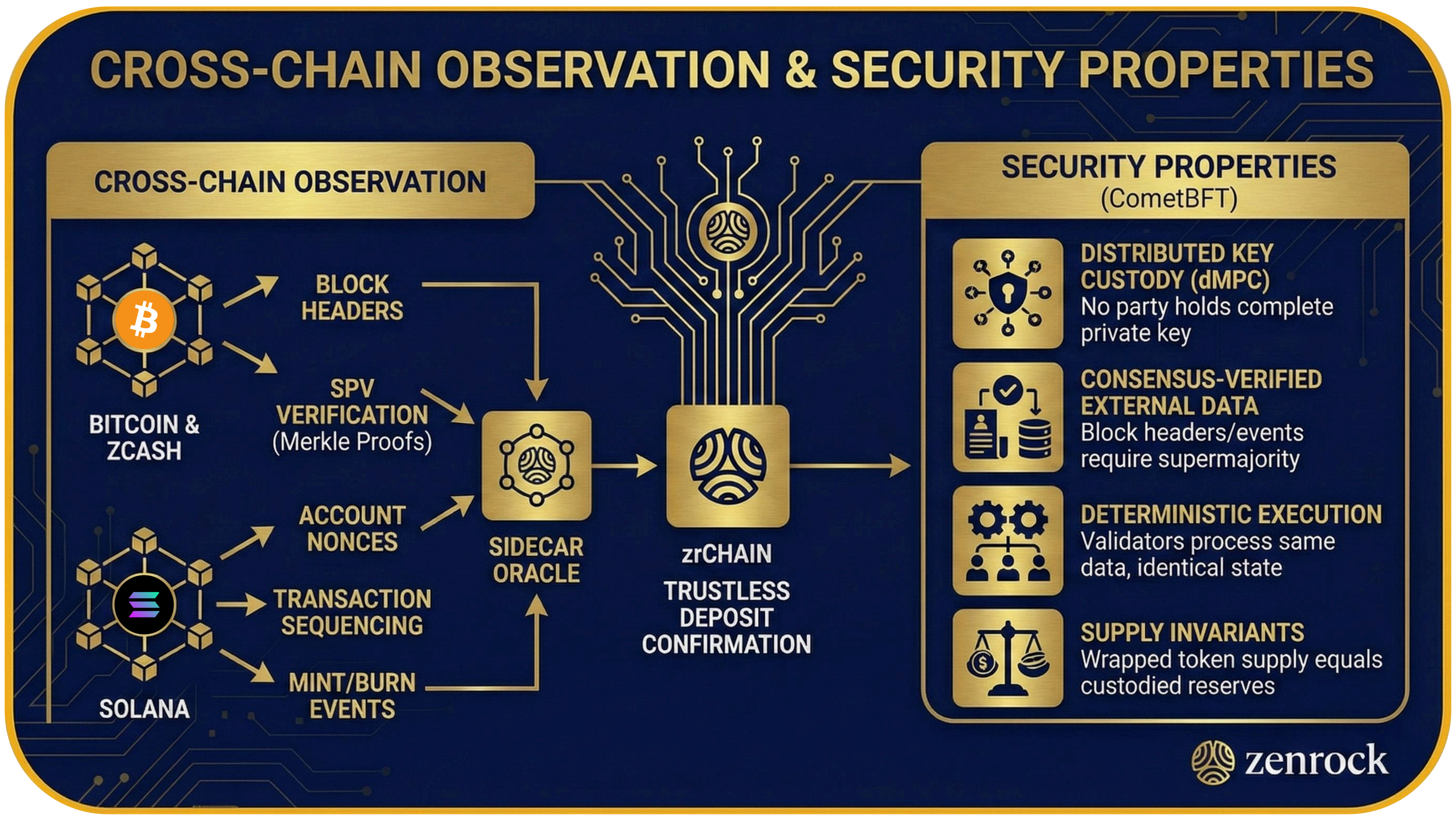
- **Bitcoin & Zcash:** Block headers for SPV-style verification (Simplified Payment Verification, a lightweight method for confirming transactions via Merkle proofs)
- **Solana:** Account nonces for transaction sequencing, mint/burn events for confirmation

Block headers are stored in dedicated collections and used to verify deposit transaction inclusion via Merkle proofs. This allows zrChain to trustlessly confirm external deposits without relying on centralized data providers.

## Security Properties

zrChain inherits CometBFT's Byzantine fault tolerance (hence the BFT in the name): the network remains secure as long as fewer than 1/3 of validators are malicious. Key security features include:

- **Distributed key custody:** dMPC ensures no party ever holds a complete private key
- **Consensus-verified external data:** Block headers and events require supermajority agreement
- **Deterministic execution:** All validators process the same data and reach identical state
- **Supply invariants:** Strict accounting ensures wrapped token supply equals custodied reserves

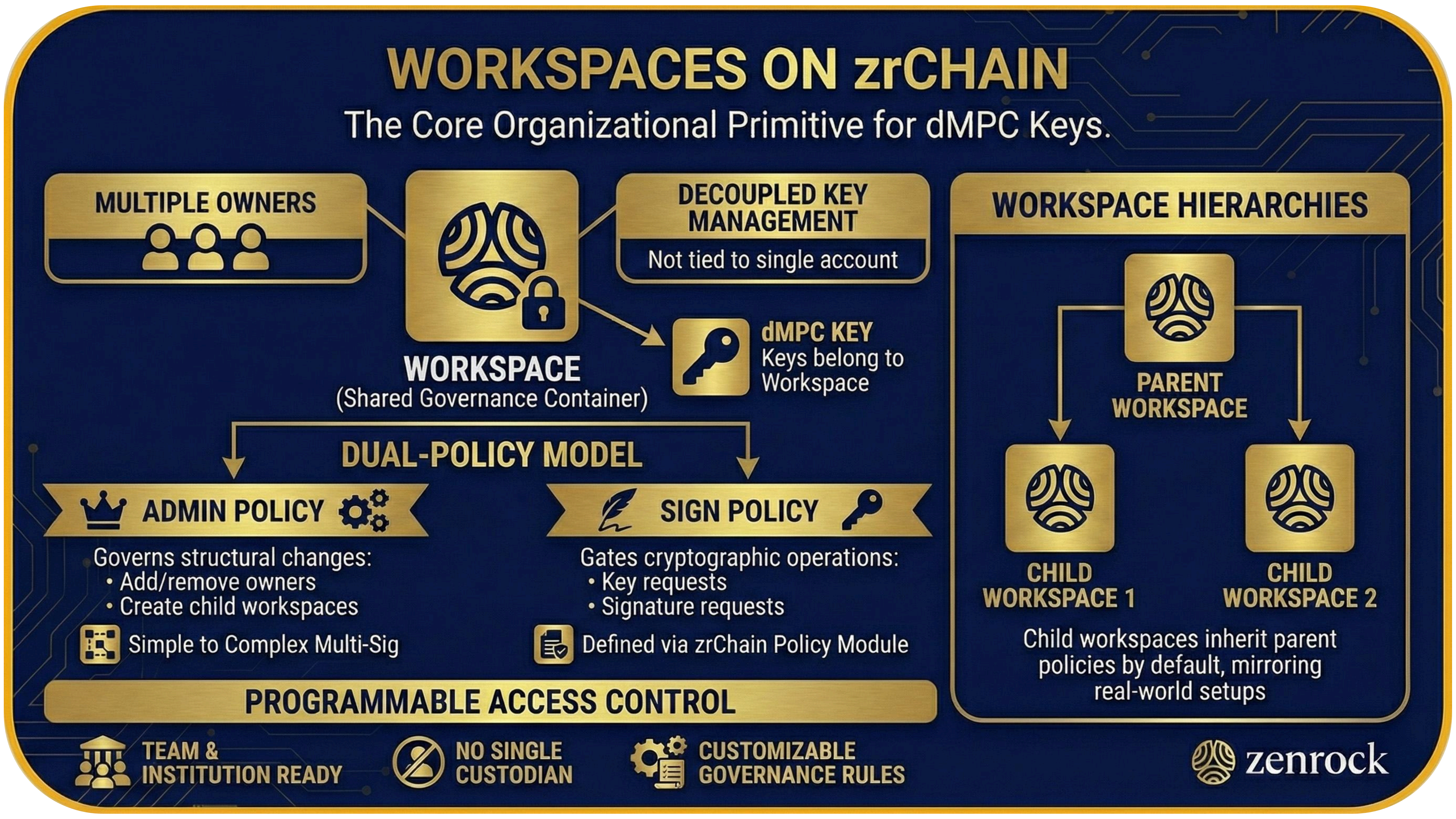

CROSS-CHAIN OBSERVATION & SECURITY PROPERTIES
CROSS-CHAIN OBSERVATION
BITCOIN & ZCASH
BLOCK HEADERS
SPV VERIFICATION (Merkle Proofs)
SOLANA
ACCOUNT NONCES
TRANSACTION SEQUENCING
MINT/BURN EVENTS
SIDECAR ORACLE
zrCHAIN
TRUSTLESS DEPOSIT CONFIRMATION
SECURITY PROPERTIES (CometBFT)
DISTRIBUTED KEY CUSTODY (dMPC)
No party holds complete private key
CONSENSUS-VERIFIED EXTERNAL DATA
Block headers/events require supermajority
DETERMINISTIC EXECUTION
Validators process same data, identical state
SUPPLY INVARIANTS
Wrapped token supply equals custodied reserves
zenrock

## Workspaces

Workspaces are the core organizational primitive on zrChain. Every dMPC key belongs to a workspace, not to an individual account. A workspace is a shared governance container, analogous to a team vault, where owners request keys, authorize signatures, and control access. This decouples key management from any single blockchain account, enabling teams, institutions, and individuals to manage cross-chain wallets under a shared governance structure.

Each workspace supports multiple owners and enforces a dual-policy model: an **admin policy** governs structural changes (adding or removing owners, creating child workspaces), while a separate **sign policy** gates all cryptographic operations (key requests, signature requests). Owners configure policies through zrChain's policy module, ranging from simple single-owner approval to complex multi-signature thresholds. Workspaces can also form hierarchies, where child workspaces inherit their parent's policies by default, enabling organizational structures that mirror real-world team setups.

**Workspaces give zrChain a programmable access control layer that makes dMPC practical for real teams and institutions, not just individual users. Any organization can create a workspace, define its own governance rules, and manage cross-chain wallets without ever trusting a single custodian.**


WORKSPACES ON zrCHAIN
The Core Organizational Primitive for dMPC Keys.
MULTIPLE OWNERS
DECOUPLED KEY MANAGEMENT
Not tied to single account
WORKSPACE
(Shared Governance Container)
dMPC KEY
Keys belong to Workspace
DUAL-POLICY MODEL
ADMIN POLICY
Governs structural changes:
• Add/remove owners
• Create child workspaces
Simple to Complex Multi-Sig
SIGN POLICY
Gates cryptographic operations:
• Key requests
• Signature requests
Defined via zrChain Policy Module
PROGRAMMABLE ACCESS CONTROL
TEAM & INSTITUTION READY
NO SINGLE CUSTODIAN
CUSTOMIZABLE GOVERNANCE RULES
WORKSPACE HIERARCHIES
PARENT WORKSPACE
CHILD WORKSPACE 1
CHILD WORKSPACE 2
Child workspaces inherit parent policies by default, mirroring real-world setups
zenrock

# Decentralized Custody Tokens (DCTs)

A DCT is any token issued through Zenrock's decentralized custody network.

Secured by dMPC and anchored by zrChain, DCTs set a new standard for cross-chain asset custody.

DCTs are more than wrappers. They're a security foundation.

DeFi primitives can be built on top: structured products, leveraged vaults, restaking strategies. Developers take creative risks.

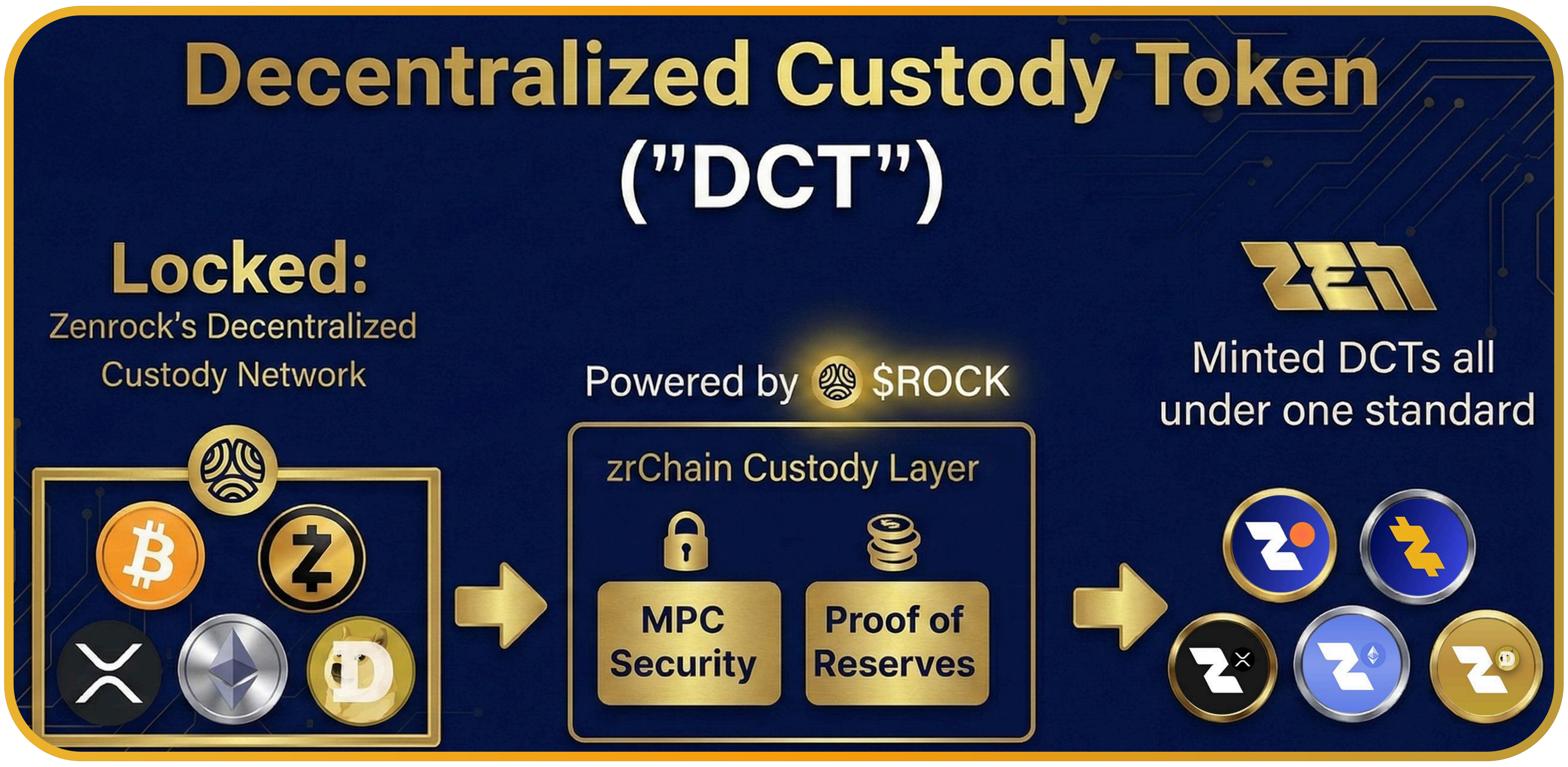
Users choose their exposure. **The underlying custody stays the same.**

**The underlying custody stays the same.**

## Decentralized Custody Tokens: Technical Architecture

The DCT framework handles the complete lifecycle: deposit detection, verification, minting, and redemption.

# Decentralized Custody Tokens (DCTs)


Decentralized Custody Token
("DCT")
Locked:
Zenrock's Decentralized Custody Network
Powered by $ROCK
zrChain Custody Layer
MPC Security
Proof of Reserves
Minted DCTs all under one standard

## Deposit & Verification Flow

Dedicated outpost services monitor external blockchains for deposits to dMPC-controlled addresses. When a deposit achieves sufficient confirmations (typically 6 for Bitcoin), the outpost generates a Merkle proof demonstrating the transaction's inclusion in the block and submits it to zrChain.

zrChain then performs multi-step verification:

1. **Block Header Validation:** The referenced block header is fetched from consensus-verified storage (see zrChain's enshrined oracle above)

2. **Merkle Proof Verification:** The transaction's inclusion is cryptographically verified against the stored block header

3. **Deduplication:** A SHA256 hash of the transaction output prevents double-processing

Upon successful verification, a PendingMintTransaction is created.

## Minting Flow

Once a deposit is verified, the minting process proceeds through multiple phases:

**Phase 1 - Consensus Verification:** The PreBlocker confirms validators have reached consensus on Solana nonce values and account states.

**Phase 2 - Transaction Construction:** The system prepares a Solana transaction specifying the exact mint amount and recipient.

**Phase 3 - MPC Signing:** The transaction is submitted to the treasury module, which collects threshold signatures from distributed key holders.

**Phase 4 - Broadcast & Confirmation:** The signed transaction is broadcast to Solana. The sidecar oracle monitors for mint events and reports confirmation back to zrChain.

## Redemption Flow

When users burn wrapped tokens on Solana to redeem the underlying asset:

1. **Burn Detection:** The sidecar oracle monitors Solana for burn events
2. **Maturity Period:** Burns enter an escrow period before being processed
3. **Transaction Construction:** The relevant outpost constructs an unsigned transaction
4. **MPC Signing:** The transaction is signed through the distributed MPC infrastructure
5. **Broadcast:** The outpost broadcasts the signed transaction to the external blockchain

## Supply Accounting

The DCT system maintains strict supply invariants:

- **Custodied:** Native assets held in custody addresses on external chains
- **Pending:** Deposits verified but not yet minted on Solana
- **Minted:** Wrapped tokens in circulation on Solana

The invariant Custodied = Pending + Minted is enforced at every state transition. This ensures wrapped token supply is always fully backed by custodied reserves.

## zenBTC

Mint: https://zenbtc.app.zenrocklabs.io/

CA: 9hX59xHHnaZXLU6quvm5uGY2iDiT3jczaReHy6A6TYKw

zenBTC is Zenrock's flagship DCT: **decentralized yield-bearing wrapped Bitcoin on Solana.**

The underlying BTC is never lent out, levered up, or used as risk capital. There are no basis trades, no delta-neutral strategies, no opaque vaults, no dependencies on other firms.

**Your BTC is in decentralized custody, designed for security.**

zenBTC: Decentralized, Institutional BTC on Solana
Live on Drift, Orca, & Kamino
Yield-Bearing
Native yield is generated automatically for all zenBTC owners, denominated in BTC
Solana-First
Natively minted on Solana with a consortium of DeFi partners
Decentralized
Secured by a distributed consortium of 8 node operators, leveraging the DCT standard

Most yield-bearing products rely on hidden strategies or unsustainable token emissions. zenBTC does neither.

**Yield is funded exclusively by real protocol fees:** 5% of all zrChain fees flow to zenBTC, plus 35% of zenBTC custody fees directly.

Yield breakdown: https://app.zenrocklabs.io/zenbtc/yield

Because zenBTC is permissionless to mint and redeem, yield is set by the market, not by marketing.

The yield rate is a function of two variables:

1. Total zenBTC supply
2. Total Zenrock protocol fees

As Zenrock earns more fees, yield rises. As yield rises, more users may choose to mint zenBTC to capture it, increasing supply and pushing yield back down. The cycle works in both directions: if yield falls too low, users redeem, supply decreases, and yield rises again.

**The result is a self-balancing mechanism that naturally discovers the market-clearing yield for low-risk BTC on Solana.**

No governance votes, no manual rate setting, no emissions schedules. Just supply and demand. And unlike most yield-bearing BTCs that pay in points or governance tokens, zenBTC yield is paid directly in sats on the Bitcoin blockchain, distributed daily.

## zenZEC

Mint: [https://app.zenrocklabs.io/zenzec/crucible](https://app.zenrocklabs.io/zenzec/crucible)

CA: JDt9rRGaieF6aN1cJkXFeUmsy7ZE4yY3CZb8tVMXVroS

zenZEC is the second DCT: **decentralized wrapped Zcash on Solana.**

Zcash pioneered zero-knowledge proofs in cryptocurrency, enabling shielded transactions where the sender, receiver, and amount are encrypted.

By bringing ZEC to Solana via the DCT standard, zenZEC inherits both Zcash's privacy heritage and Zenrock's decentralized custody guarantees.

## Hush Protocol

Use Hush: https://hushprotocol.xyz

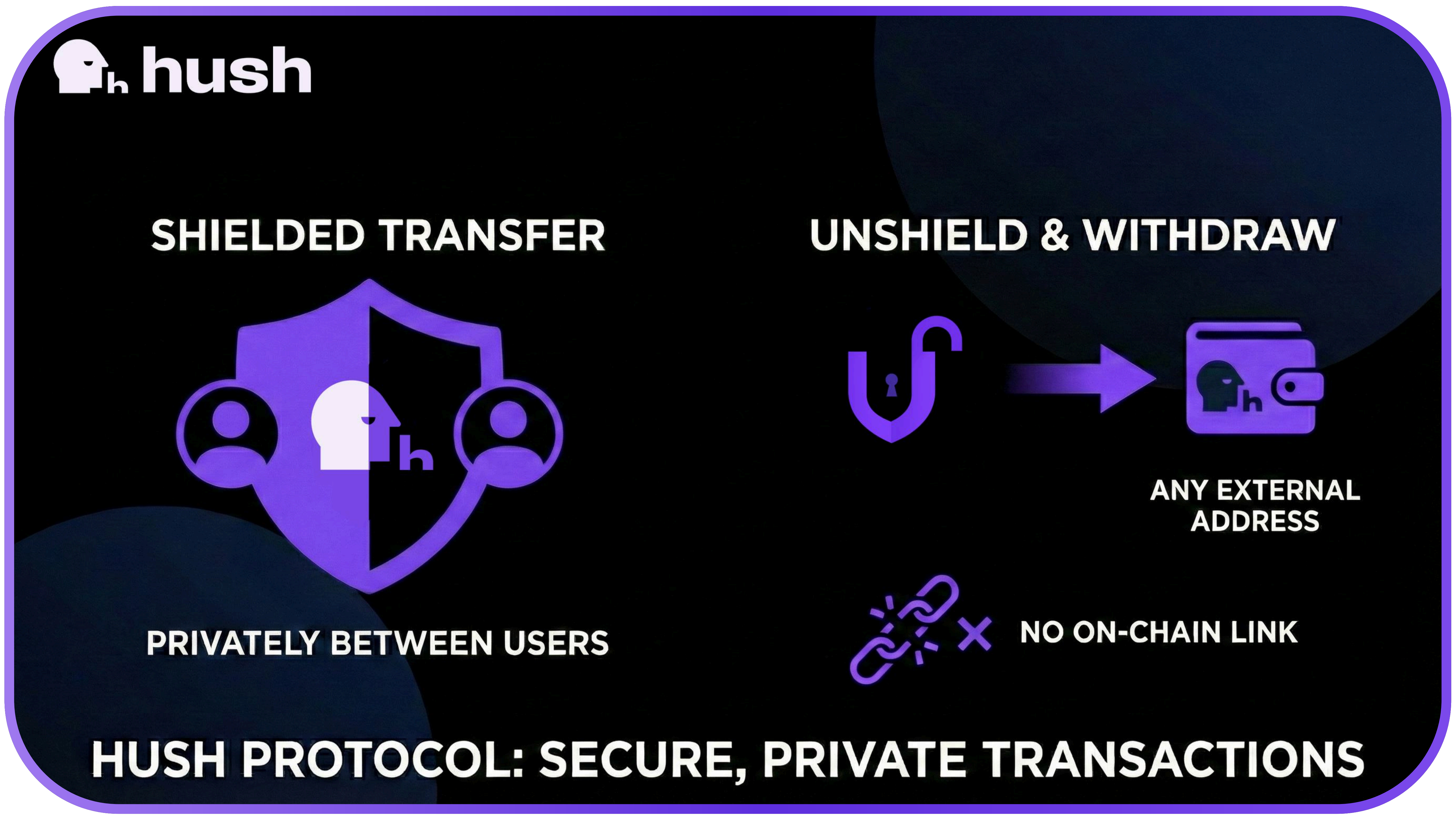
Hush is the privacy layer for Solana.

Solana is fast, cheap, and increasingly the default venue for DeFi. But it's a public ledger.

Every transaction, every balance, every decision is visible to anyone. For many users and use cases, that's a problem. And as Solana is used for more things (rent payment, salary, car payment, stock trading, and more), the need for privacy will only grow.

Hush enables anonymity via a shielded asset pool. Users deposit ("shield") jitoSOL into a pool where their assets become indistinguishable from everyone else's. They can then:

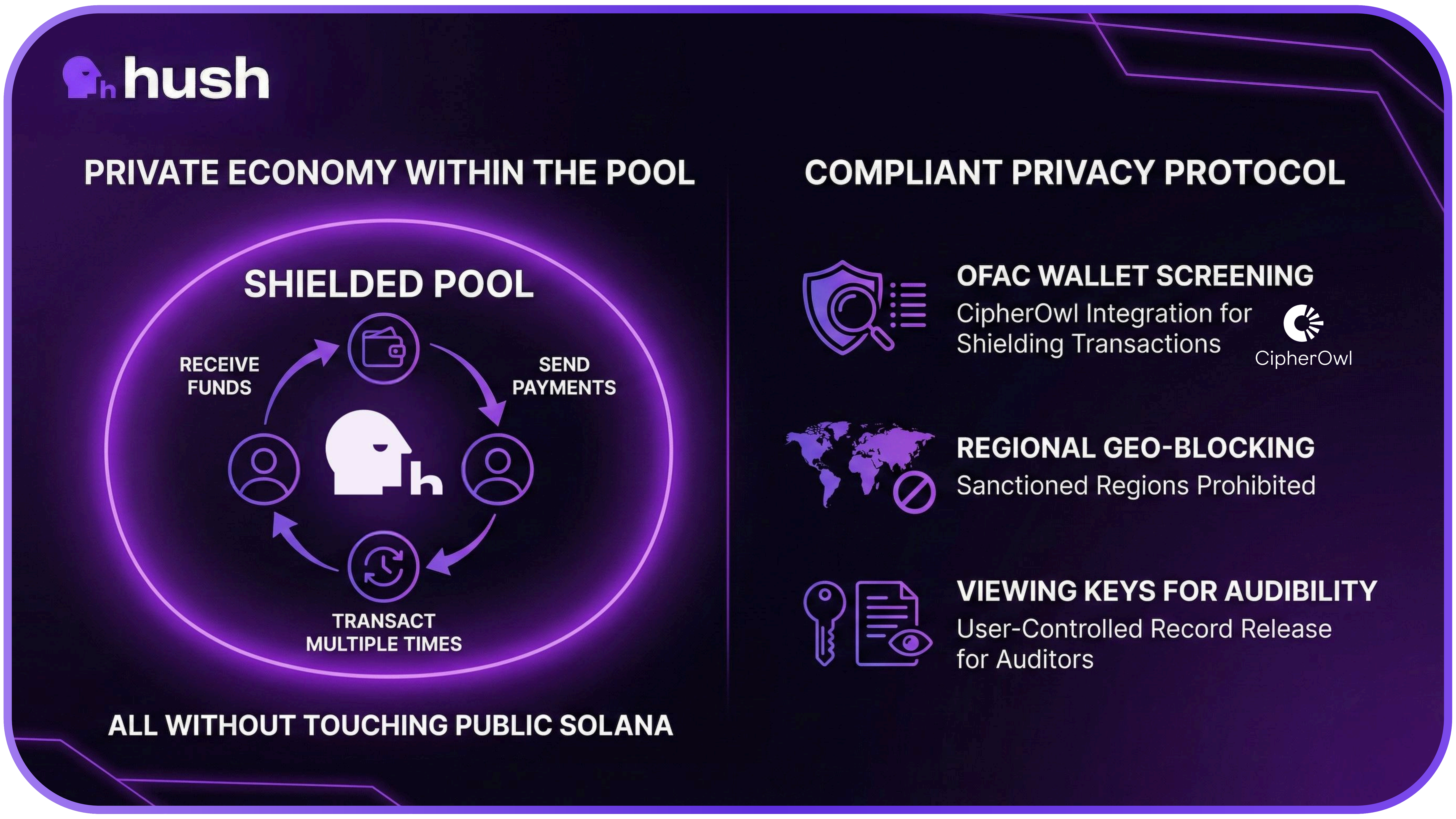
- **Transfer privately** to other Hush users without ever leaving the shielded pool
- **Unshield** to any address—with no onchain link to the original deposit



This creates an entirely private economy within the pool. Users can receive funds, send payments, and transact multiple times—all without touching public Solana until they choose to.

Hush is designed to incorporate core compliance features:

- **OFAC wallet screening:** All shielding transactions are screened via a CipherOwl integration
- **AML/ Theft screening:** All shielding transactions are screened via a CipherOwl integration
- **Terrorist financing / CSAM screening:** All shielding transactions are screened via a CipherOwl integration
- **Regional geo-blocking:** Sanctioned regions are prohibited from interacting with Hush
- **Viewing keys for audibility:** Users retain a record of their shielded activity that they can release to auditors at their own discretion


hush
PRIVATE ECONOMY WITHIN THE POOL
SHIELDED POOL
RECEIVE FUNDS
SEND PAYMENTS
TRANSACT MULTIPLE TIMES
ALL WITHOUT TOUCHING PUBLIC SOLANA
COMPLIANT PRIVACY PROTOCOL
OFAC WALLET SCREENING
CipherOwl Integration for Shielding Transactions
CipherOwl
REGIONAL GEO-BLOCKING
Sanctioned Regions Prohibited
VIEWING KEYS FOR AUDIBILITY
User-Controlled Record Release for Auditors

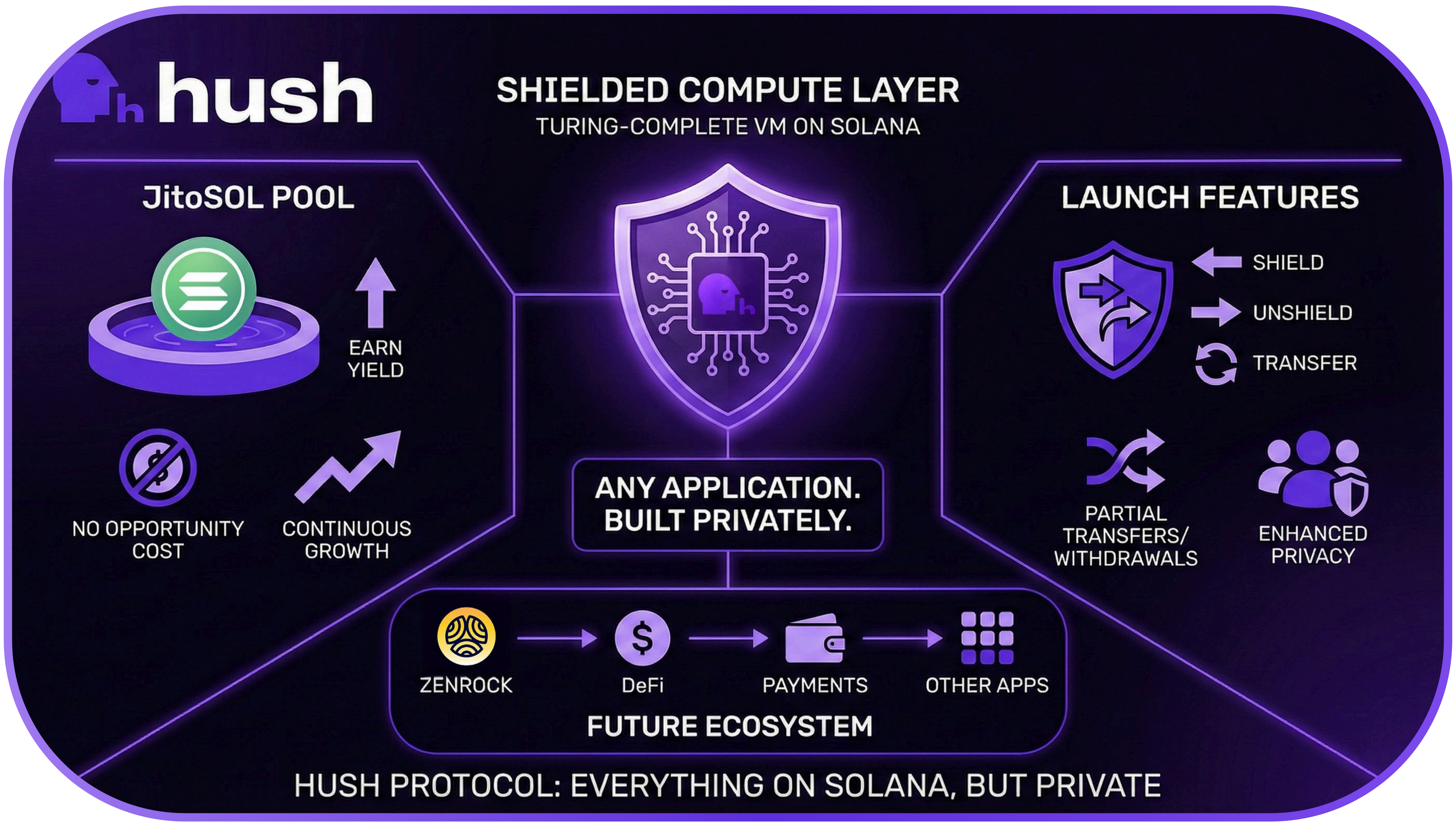
## The Vision

Hush is a shielded compute layer: a fully Turing-complete VM built on Solana. Any application that can be built on Solana can be built privately via Hush.

The pool is denominated in jitoSOL rather than native SOL so users do not have to sacrifice staking yield while remaining private. No opportunity cost means no reason to leave. This design allows the pool to grow continuously without tradeoffs, and as the pool grows, privacy guarantees strengthen for everyone.

At launch, shielding, unshielding, and transferring are enabled. Users can also perform partial transfers and partial withdrawals, which increases the privacy guarantees for the entire pool by allowing large deposits to strengthen anonymity for smaller ones.

Over time, Zenrock Labs and third parties will build on Hush, bringing DeFi, payments, and other applications into the shielded layer. The goal is privacy and usability without tradeoffs: everything you have on Solana, but private.



## Hush Protocol: Technical Architecture

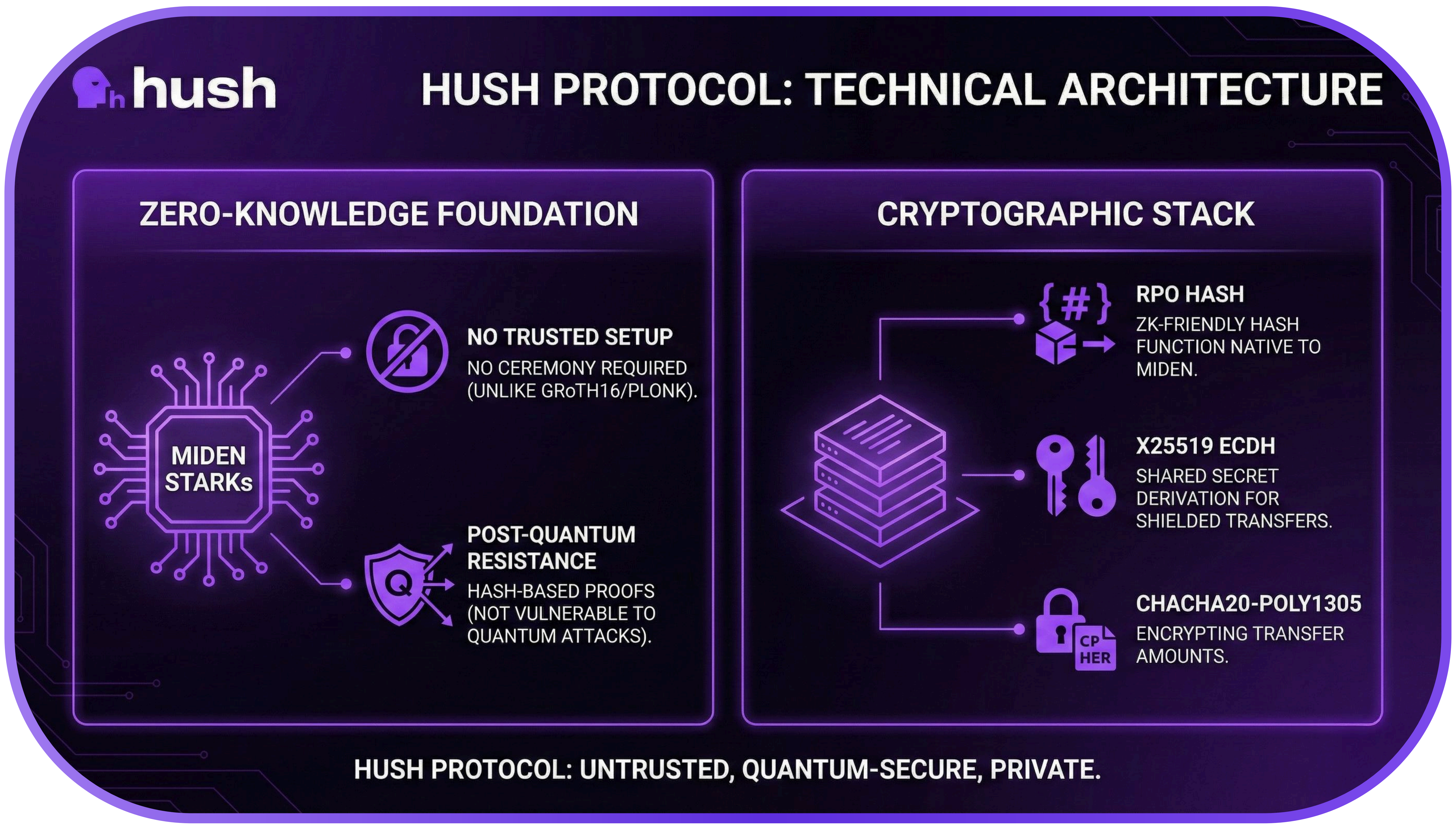
**Zero-Knowledge Foundation**
Hush uses **Miden zk-STARKs** as its proof system, chosen for two critical properties:

- **No Trusted Setup:** Unlike Groth16 or PLONK, Miden zk-STARKs require no ceremony or trusted parameters
- **Post-Quantum Resistance:** Hash-based proofs are not vulnerable to known quantum attacks like Shor's algorithm

The cryptographic stack includes:

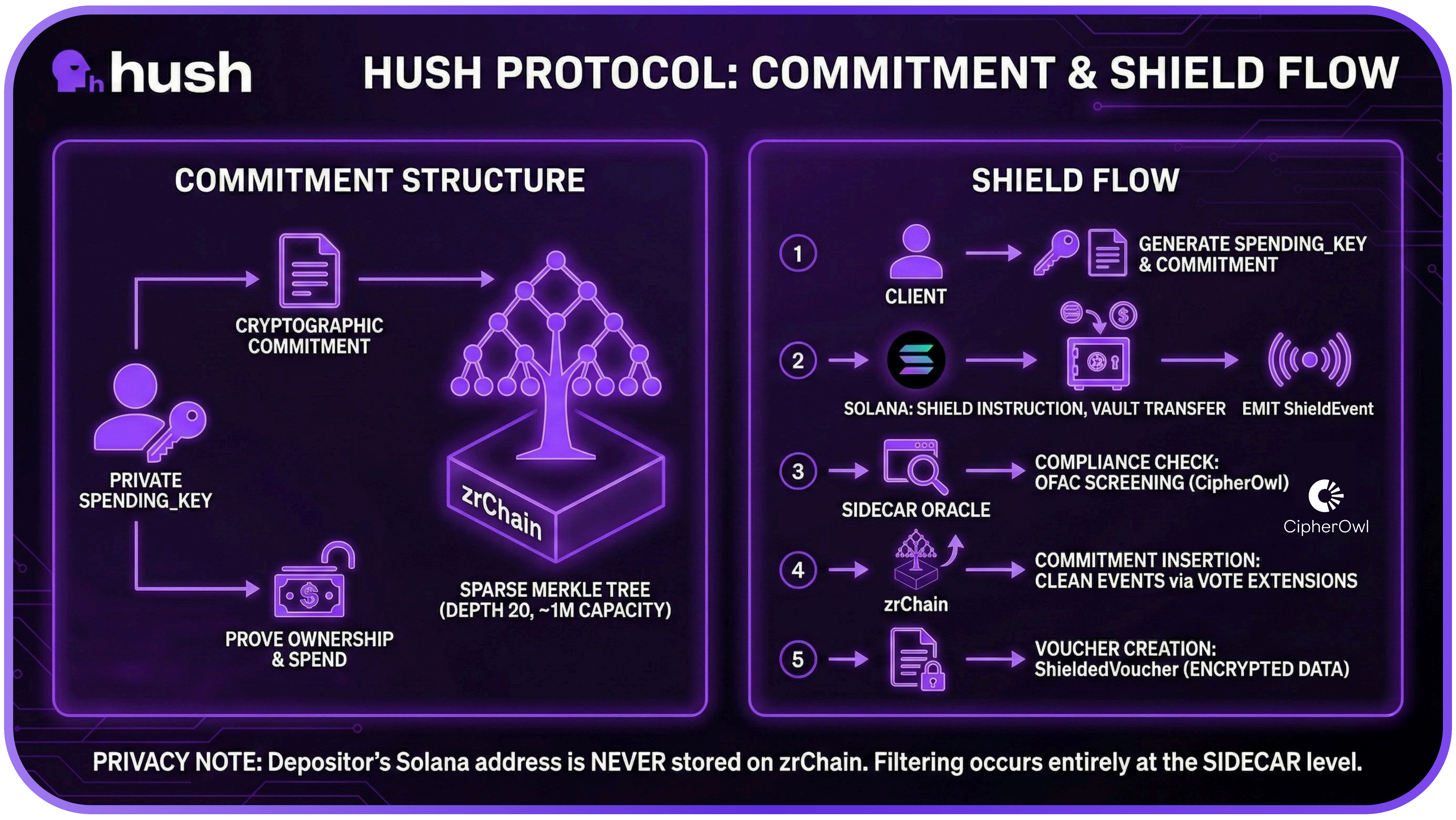
- **RPO Hash:** Rescue Prime Optimized, a ZK-friendly hash function native to Miden
- **X25519 ECDH:** For deriving shared secrets in shielded transfers
- **ChaCha20-Poly1305:** For encrypting transfer amounts

| Primitive | Algorithm | Purpose |
|---|---|---|
| Hash | RPO (Rescue Prime Optimized) | ZK-friendly hash native to Miden |
| Key Exchange | X25519 ECDH | Stealth addresses for shielded transfers |
| Encryption | ChaCha20-Poly1305 | Amount encryption and wallet recovery |


hush
HUSH PROTOCOL: TECHNICAL ARCHITECTURE
ZERO-KNOWLEDGE FOUNDATION
MIDEN STARKs
NO TRUSTED SETUP
NO CEREMONY REQUIRED (UNLIKE GRoTH16/PLONK).
POST-QUANTUM RESISTANCE
HASH-BASED PROOFS (NOT VULNERABLE TO QUANTUM ATTACKS).
CRYPTOGRAPHIC STACK
RPO HASH
ZK-FRIENDLY HASH FUNCTION NATIVE TO MIDEN.
X25519 ECDH
SHARED SECRET DERIVATION FOR SHIELDED TRANSFERS.
CHACHA20-POLY1305
ENCRYPTING TRANSFER AMOUNTS.
HUSH PROTOCOL: UNTRUSTED, QUANTUM-SECURE, PRIVATE.

## Commitment Structure

When a user shields tokens, they create a cryptographic commitment that is added to a sparse Merkle tree (depth 24, ~16.7 million capacity) stored on zrChain. The user retains their spending_key privately, which is required to later prove ownership and spend the shielded funds.


hush
HUSH PROTOCOL: COMMITMENT & SHIELD FLOW
COMMITMENT STRUCTURE
CRYPTOGRAPHIC COMMITMENT
PRIVATE SPENDING_KEY
PROVE OWNERSHIP & SPEND
zrChain
SPARSE MERKLE TREE (DEPTH 20, ~1M CAPACITY)
SHIELD FLOW
1 CLIENT
GENERATE SPENDING_KEY & COMMITMENT
2 SOLANA: SHIELD INSTRUCTION, VAULT TRANSFER
EMIT ShieldEvent
3 SIDECAR ORACLE
COMPLIANCE CHECK: OFAC SCREENING (CipherOwl)
CipherOwl
4 zrChain
COMMITMENT INSERTION: CLEAN EVENTS via VOTE EXTENSIONS
5 VOUCHER CREATION: ShieldedVoucher (ENCRYPTED DATA)
PRIVACY NOTE: Depositor's Solana address is NEVER stored on zrChain. Filtering occurs entirely at the SIDECAR level.

## Shield Flow

1. **Client:** User generates a spending_key from their connected Solana wallet and derives a commitment from their deposit amount

2. **Solana:** User calls the shield instruction, transferring tokens to a vault and emitting a ShieldEvent

3. **Compliance Check:** The sidecar oracle performs an OFAC screening via CipherOwl

4. **Commitment Insertion:** Clean shield events are reported to zrChain via vote extensions and inserted into the Merkle tree

5. **Voucher Creation:** A ShieldedVoucher is created with encrypted amount data

## Unshield Flow

1. **Client:** User generates a zk-STARK proof demonstrating knowledge of a valid commitment in the tree and correct nullifier derivation

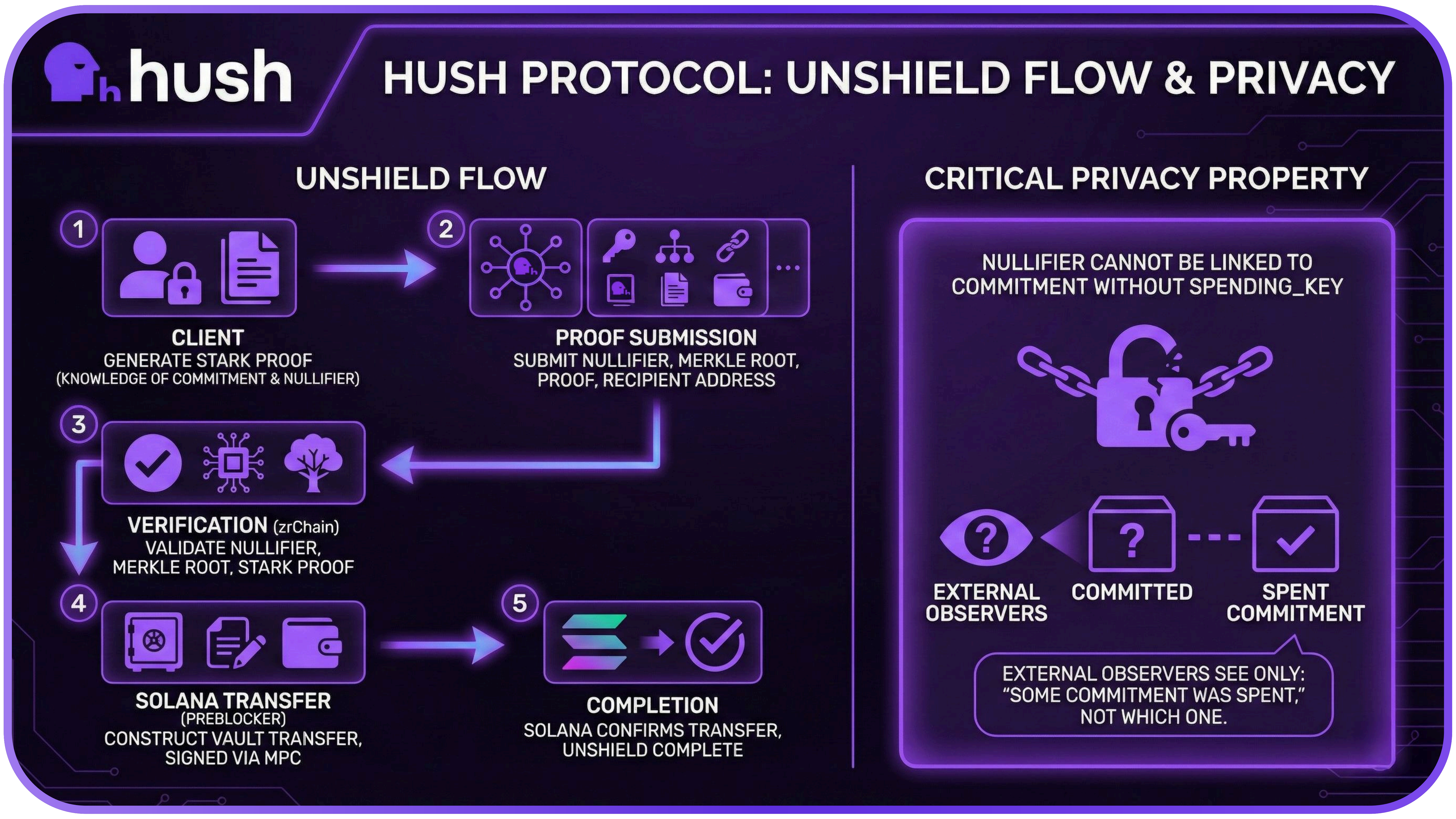
2. **Proof Submission:** User submits the nullifier, Merkle root, zk-STARK proof, and recipient address

3. **Verification:** zrChain validates the nullifier hasn't been spent, the Merkle root is valid, and the zk-STARK proof verifies

4. **Solana Transfer:** The PreBlocker constructs a transfer from the vault to the recipient, signed via MPC

5. **Completion:** Once Solana confirms the transfer, the unshield is marked complete

The critical privacy property: the nullifier cannot be linked to the original commitment without knowing the spending_key. External observers see only that some commitment was spent, not which one.


hush
HUSH PROTOCOL: UNSHIELD FLOW & PRIVACY
UNSHIELD FLOW
1
CLIENT
GENERATE STARK PROOF
(KNOWLEDGE OF COMMITMENT & NULLIFIER)
2
PROOF SUBMISSION
SUBMIT NULLIFIER, MERKLE ROOT, PROOF, RECIPIENT ADDRESS
3
VERIFICATION (zrChain)
VALIDATE NULLIFIER, MERKLE ROOT, STARK PROOF
4
SOLANA TRANSFER
(PREBLOCKER)
CONSTRUCT VAULT TRANSFER, SIGNED VIA MPC
5
COMPLETION
SOLANA CONFIRMS TRANSFER, UNSHIELD COMPLETE
CRITICAL PRIVACY PROPERTY
NULLIFIER CANNOT BE LINKED TO COMMITMENT WITHOUT SPENDING_KEY
EXTERNAL OBSERVERS
COMMITTED
SPENT COMMITMENT
EXTERNAL OBSERVERS SEE ONLY: "SOME COMMITMENT WAS SPENT," NOT WHICH ONE.

## Shielded Transfers

Users can transfer value within the privacy pool in a fully encrypted manner without ever exposing private info on Solana—a key feature that enables private payments between Hush users' shielded addresses:

1. Sender generates a zk-STARK proof showing they're spending a valid commitment and creating new commitments for recipient and change
2. Transfer amounts are encrypted using ECDH with the recipient's public viewing key
3. zrChain verifies the proof and adds new commitments to the Merkle tree
4. Recipient scans for incoming transfers using their incoming_viewing_key, decrypting amounts they receive

All amounts remain private—only the commitments and fee are visible onchain. This enables an entirely private economy within the shielded pool: users can receive funds, transact multiple times, and only unshield when they need to interact with public Solana.

## Nullifier System

Nullifiers prevent double-spending while preserving privacy. Each commitment can only produce one valid nullifier, derived from the spending key. When spent, the nullifier is permanently recorded.

Attempting to spend the same commitment again produces the same nullifier, which is rejected as already spent.

Crucially, knowing a nullifier reveals nothing about which commitment it corresponds to without the spending key. In addition, each transaction is padded with decoy nullifiers so that nobody is able to tell how many inputs each transaction has.

**This provides an even greater level of privacy than Zcash**, where input/output counts per transaction are public and could be used to assist in probabilistically deanonymising users.

### Key Hierarchy

Hush implements tiered keys for different permission levels:

| Key | Spend | View Spent | Decrypt Incoming |
|---|---|---|---|
| spending_key | Yes | Yes | Yes |
| full_viewing_key | No | Yes | Yes |
| incoming_viewing_key | No | No | Yes |

- **Spending Key:** Full control, kept private by the user
- **Full Viewing Key:** Share with auditors to reveal complete transaction history without spending ability
- **Incoming Viewing Key:** Share to receive shielded transfers

## Fee Model

| Operation | Fee |
|---|---:|
| Shield | Free |
| Transfer | 0.01 JitoSOL |
| Unshield | 0.50% |

For jitoSOL unshields, users can optionally request 0.01 SOL funding (+10 bps) to enable withdrawals to fresh wallets with zero SOL balance. Fees are collected into a protocol fee pool and distributed according to Zenrock's tokenomic architecture, flowing value to $ROCK holders via the Node Reward Pool architecture described below.

## Privacy Guarantees

**What's Hidden:**

- Link between deposit and withdrawal addresses
- Transfer amounts, sender and recipient addresses (encrypted with recipient's key)
- Which commitment is being spent (only nullifiers revealed)
- Amount of transaction inputs (decoy nullifiers are commingled with real ones)

**What's Visible:**

- Deposit/withdrawal amounts and addresses (visible on Solana)
- Total pool size (aggregate of all shielded funds)
- Nullifiers (but not their corresponding commitments)

---

With zrChain, DCTs, and Hush all generating fees, these revenue streams flow to a single token: $ROCK.

**Solana CA:** 5VsPJ2EG7jjo3k2LPzQVriENKKQkNUTzujEzuaj4Aisf

zenBTC, Hush, and every future product built on zrChain share one thing:
**$ROCK is the main value capture mechanism.**

This is achieved via our Node Reward Pool architecture (detailed in the Staking $ROCK section).

$ROCK is the native token of zrChain, serving as gas, governance, and economic foundation.

It exists natively on both Cosmos (ICS-20 implementation) and Solana (via zenTP native $ROCK bridge).

**$ROCK has a fixed supply of 1 billion tokens. With no inflation, ever.**

All rewards and incentives come from existing supply, including pre-allocated pools established at genesis.

There is no mechanism to mint new tokens.

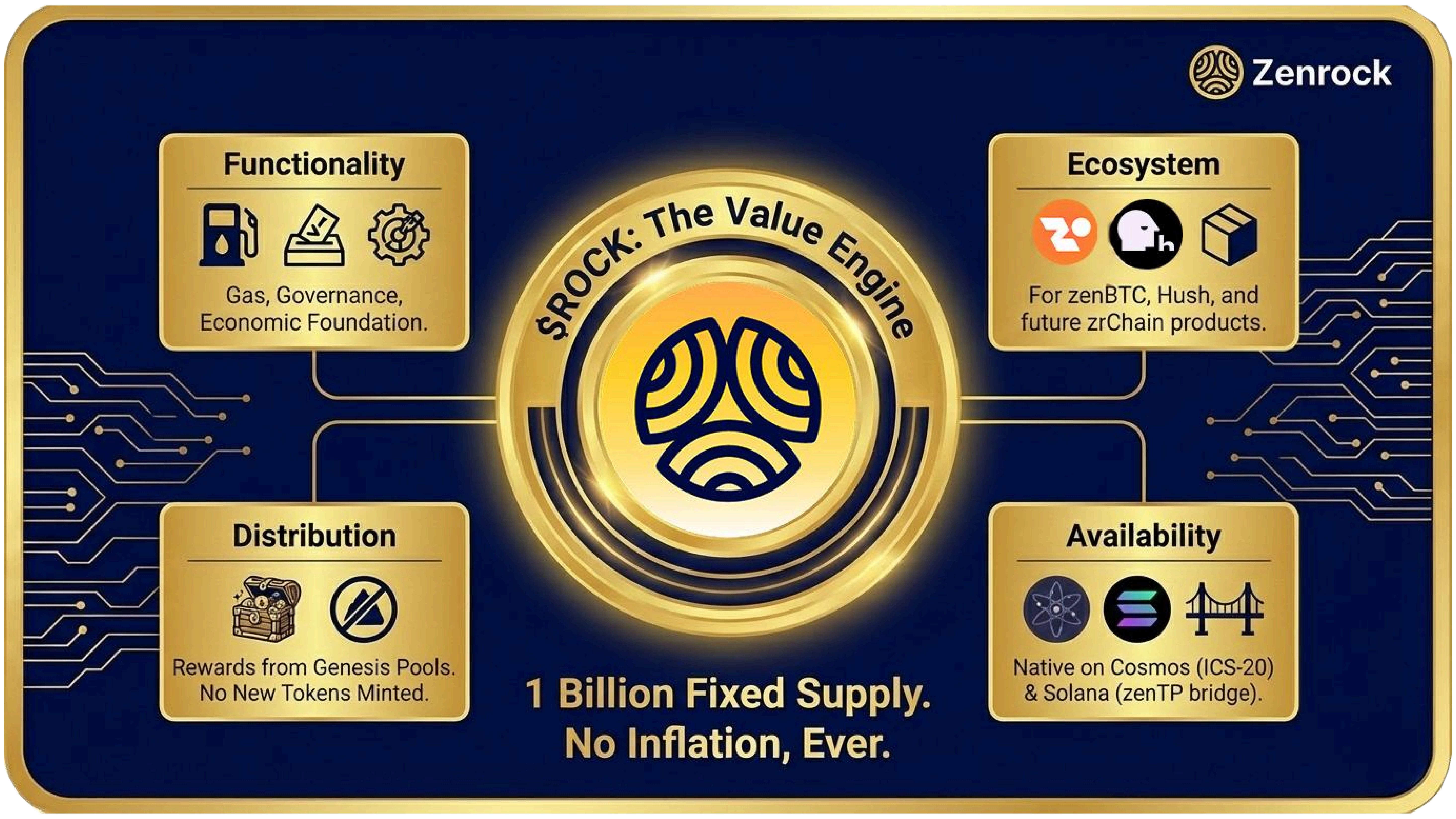
Unlike most tokens, **$ROCK had economic utility immediately at TGE**. All fees across the Zenrock ecosystem are paid in $ROCK (though often abstracted from the user).

**Every revenue line generated by Zenrock Labs, onchain and offchain, as well as all onchain revenue from all other builders flows through the token.**

**This creates a direct, protocol-driven bid on $ROCK.** As fees increase, protocol-driven demand for the token increases.

Those tokens are then distributed across the ecosystem according to Zenrock's tokenomic architecture (detailed in the Tokenomic Design section) and critically, are returned to the circulating market slowly over time due to the design of the Node Reward Pool.

As Zenrock Labs and others ship more products and more builders deploy on zrChain, protocol revenue grows. **More products, more fees, more protocol-driven demand for $ROCK.**


Zenrock
$ROCK: The Value Engine
Functionality
Gas, Governance, Economic Foundation.
Ecosystem
For zenBTC, Hush, and future zrChain products.
Distribution
Rewards from Genesis Pools. No New Tokens Minted.
Availability
Native on Cosmos (ICS-20) & Solana (zenTP bridge).
1 Billion Fixed Supply. No Inflation, Ever.

## Protocol Revenue

**All Zenrock products generate fees that flow through $ROCK. Here's how fees are structured across the ecosystem:**

**DCTs** (zenBTC, zenZEC, and future DCTs):

| Action | Fee |
|---|---|
| Mint | $5 flat |
| Redeem | 50bps |

**Hush:**

| Action | Fee |
|---|---|
| Shield | Free |
| Transfer | 0.01 JitoSOL |
| Unshield | 50bps |

All fees are converted to $ROCK and distributed according to Zenrock's tokenomic architecture (detailed in the next section). As more products launch and usage grows, protocol revenue scales accordingly.

# Zenrock's Tokenomic Design

**Every fee from every product built on Zenrock's infrastructure, whether developed by Zenrock Labs or an external team, is handled according to the same tokenomic framework. Fees earned are distributed as follows:**

- **35% to Zenrock Labs:** operations, development and $ROCK accumulation

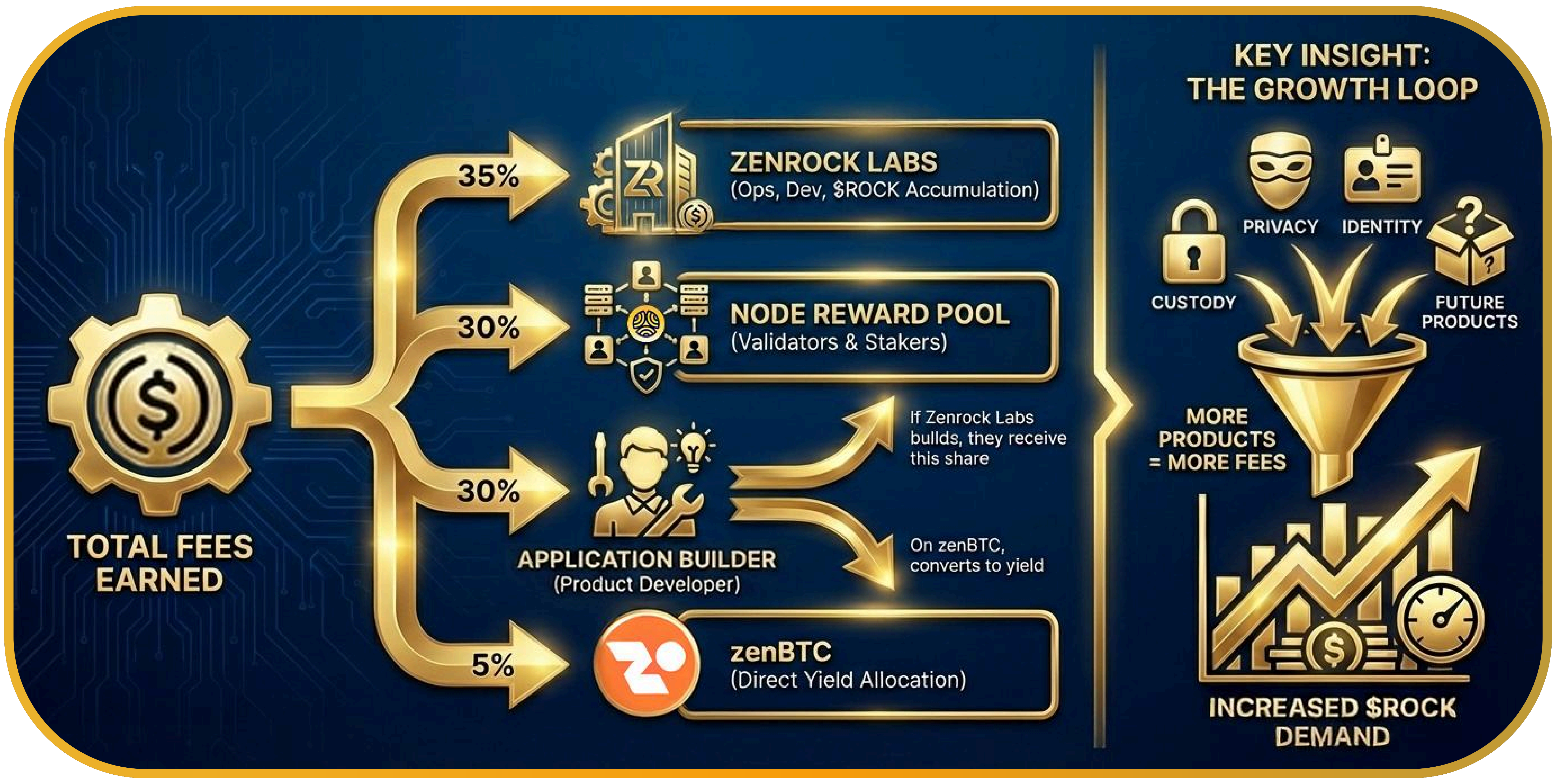
- **30% to Node Reward Pool (NRP):** validators and stakers (distributed over 36 months per the below schedule)

- **30% to Application Builder:** developer of the product
  - This can be taken by the builder or otherwise directed at their discretion
  - In instances where Zenrock Labs develops the product generating the fee, it is also counted as the application builder
  - On zenBTC (though developed by Zenrock Labs) the application builder share is converted to yield on zenBTC

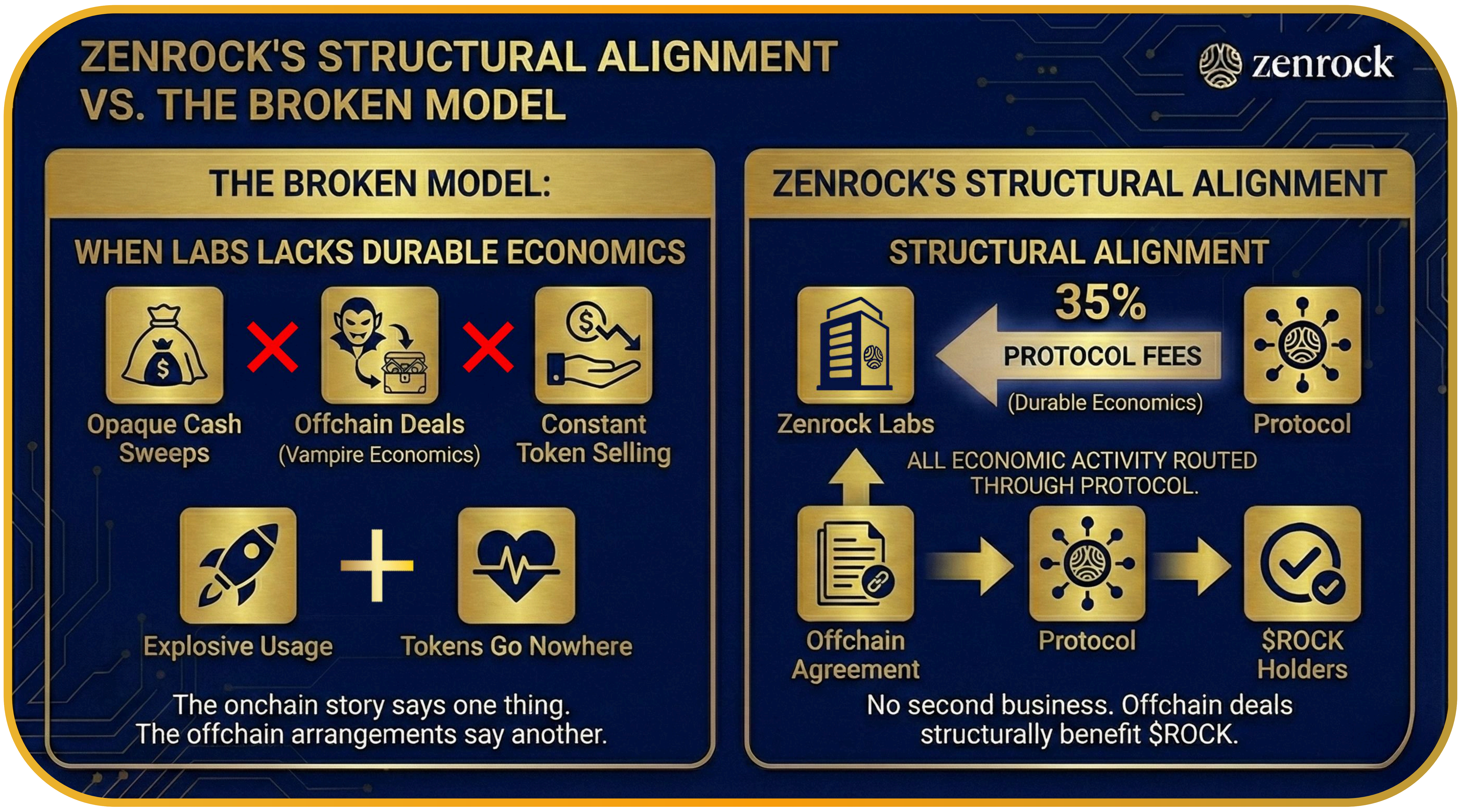
- **5% to zenBTC:** direct allocation to yield on zenBTC

**The key insight:** it doesn't matter *what* gets built on zrChain. Custody products, privacy products, identity products, things we haven't thought of yet. They all feed the same tokenomics.

**More products = more fees = more demand for $ROCK.**


TOTAL FEES EARNED
35%
ZENROCK LABS
(Ops, Dev, $ROCK Accumulation)
30%
NODE REWARD POOL
(Validators & Stakers)
30%
APPLICATION BUILDER
(Product Developer)
If Zenrock Labs builds, they receive this share
On zenBTC, converts to yield
5%
zenBTC
(Direct Yield Allocation)
KEY INSIGHT: THE GROWTH LOOP
PRIVACY
IDENTITY
CUSTODY
FUTURE PRODUCTS
MORE PRODUCTS = MORE FEES
INCREASED $ROCK DEMAND

# The Role of Zenrock Labs


ZENROCK'S STRUCTURAL ALIGNMENT VS. THE BROKEN MODEL
zenrock
THE BROKEN MODEL:
WHEN LABS LACKS DURABLE ECONOMICS
Opaque Cash Sweeps
Offchain Deals (Vampire Economics)
Constant Token Selling
Explosive Usage
Tokens Go Nowhere
The onchain story says one thing. The offchain arrangements say another.
ZENROCK'S STRUCTURAL ALIGNMENT
STRUCTURAL ALIGNMENT
35%
PROTOCOL FEES
(Durable Economics)
Zenrock Labs
Protocol
ALL ECONOMIC ACTIVITY ROUTED THROUGH PROTOCOL.
Offchain Agreement
Protocol
$ROCK Holders
No second business. Offchain deals structurally benefit $ROCK.

Crypto loves the slogan **"all value accrues to the token."** In practice, that is usually fiction.

Real protocols need a real operating company to build product, ship integrations, support partners, run audits, handle compliance, grow awareness through BD and events, help third-party builders deploy, and maintain core infrastructure. That's Zenrock Labs. And all of this costs money.

**Zenrock Labs is the most important developer contributing to the Zenrock ecosystem and thus has a key place in zrChain tokenomics with a fixed 35% share of protocol fees.**

**This fee share in turn will allow Zenrock Labs to direct all offchain revenue it earns to the tokenomics top of funnel, maximizing alignment while strengthening the transparency between Zenrock Labs and the protocol.**

Projects that underfund their Labs Co end up with stalled development, missed partnerships, and slow ecosystem growth. The token holder suffers.

**A well-resourced Labs Co means more products, faster growth, stronger partnerships, and a more valuable network.**

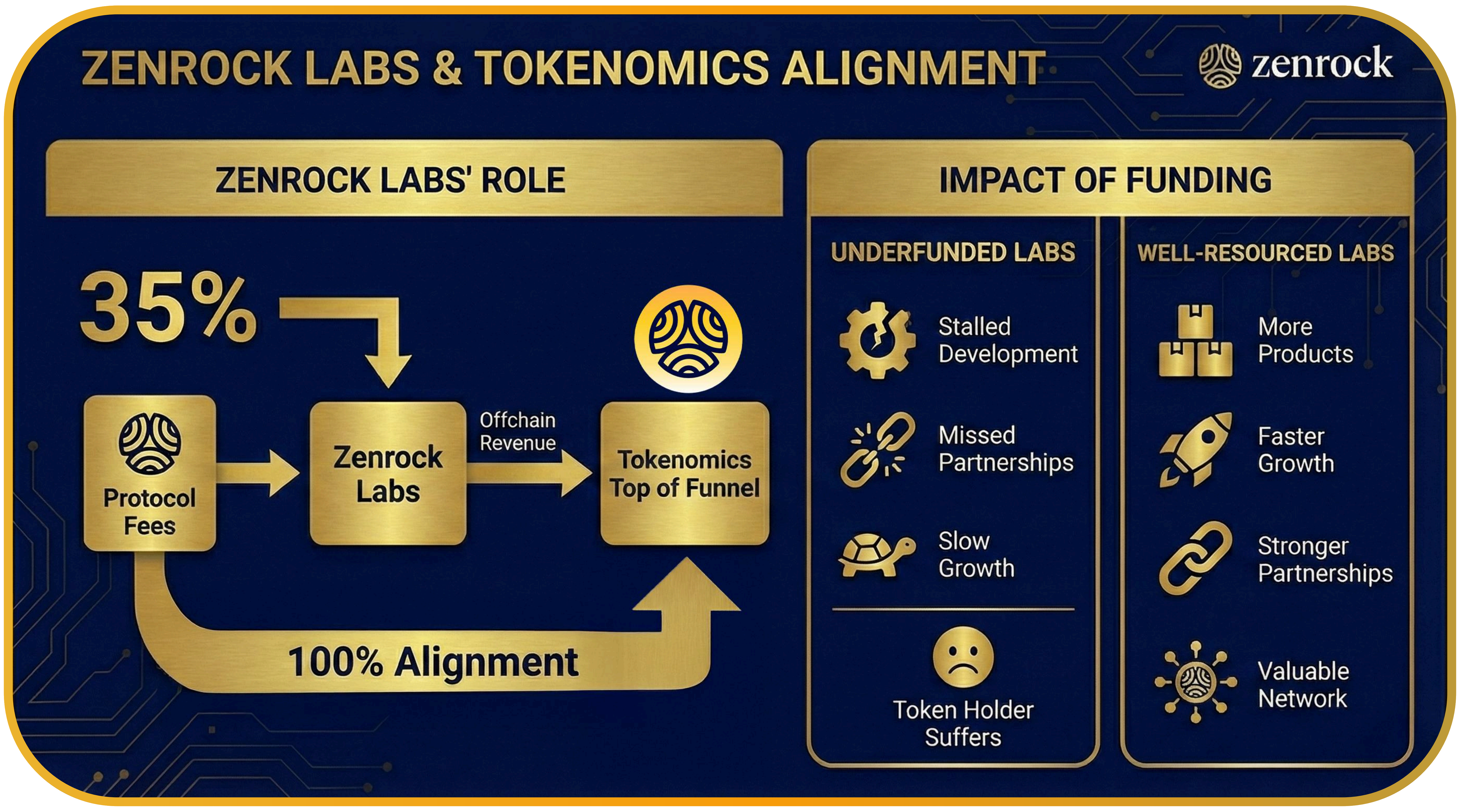
**But here is the problem: very often Labs Co has no direct way to participate in the protocol's success: they are completely cut out of the tokenomic model.**

So invariably they find other ways to make money. Offchain deals, backdoor arrangements, quiet revenue streams that never touch the token. **Labs Co says "all value accrues to the token" while quietly building a separate business on the side. This activity is pervasive.**

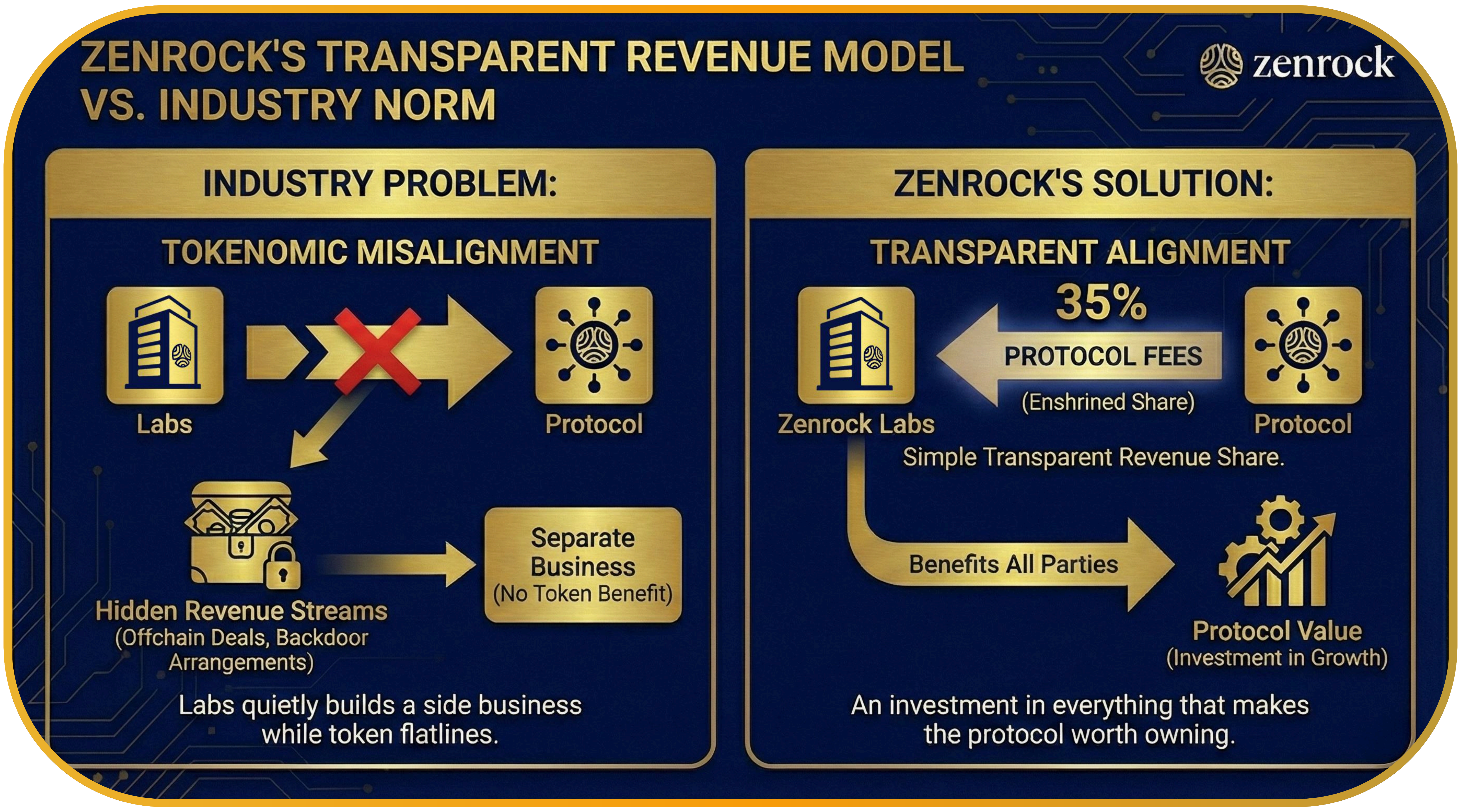
**Zenrock Labs does not operate that way.** We see transparency as a structural advantage. $ROCK tokenholders are not stakeholders in Zenrock Labs and we don't suggest they are, but we understand **that it is crucial for the protocol to align disparate economic incentives by design.**

**So instead of backroom deals that benefit a Labs Co while the token flatlines, we built alignment into the protocol itself with a simple transparent revenue share that aligns and benefits all parties.**

To that end, **Zenrock Labs receives an enshrined 35% of all protocol fees.** This isn't a hidden offchain drain on the protocol. **It's a transparent investment in everything that makes the protocol worth owning.**



## Offchain Revenues


ZENROCK'S TRANSPARENT REVENUE MODEL VS. INDUSTRY NORM
zenrock
INDUSTRY PROBLEM:
TOKENOMIC MISALIGNMENT
Labs
Protocol
Hidden Revenue Streams (Offchain Deals, Backdoor Arrangements)
Separate Business (No Token Benefit)
Labs quietly builds a side business while token flatlines.
ZENROCK'S SOLUTION:
TRANSPARENT ALIGNMENT
35%
PROTOCOL FEES
(Enshrined Share)
Zenrock Labs
Protocol
Simple Transparent Revenue Share.
Benefits All Parties
Protocol Value (Investment in Growth)
An investment in everything that makes the protocol worth owning.

When projects pretend their Labs doesn't need durable economics, one of three things happens:

- **Opaque cash sweeps:** quiet arrangements where money flows back to Labs through hidden mechanisms
- **Offchain deals that siphon value:** the classic "vampire economics" problem where usage and fees grow but the token does not benefit
- **Constant token selling:** creating perpetual sell pressure of the magnitude that has destroyed countless promising projects

This is why you see protocols with explosive usage paired with tokens that go nowhere. **The onchain story says one thing. The offchain arrangements say another.**

Zenrock is structured differently. Zenrock Labs is compensated 35% of Zenrock protocol fees, and in exchange, **Zenrock Labs agrees to route all economic activity through the protocol.**

**If Zenrock Labs signs an offchain agreement, all resulting revenue will be treated as protocol revenue and distributed through the same publicly visible tokenomic fee structure. Zenrock Labs will have no second business quietly capturing value outside the protocol. This way, even offchain deals are structurally designed to benefit $ROCK holders.**

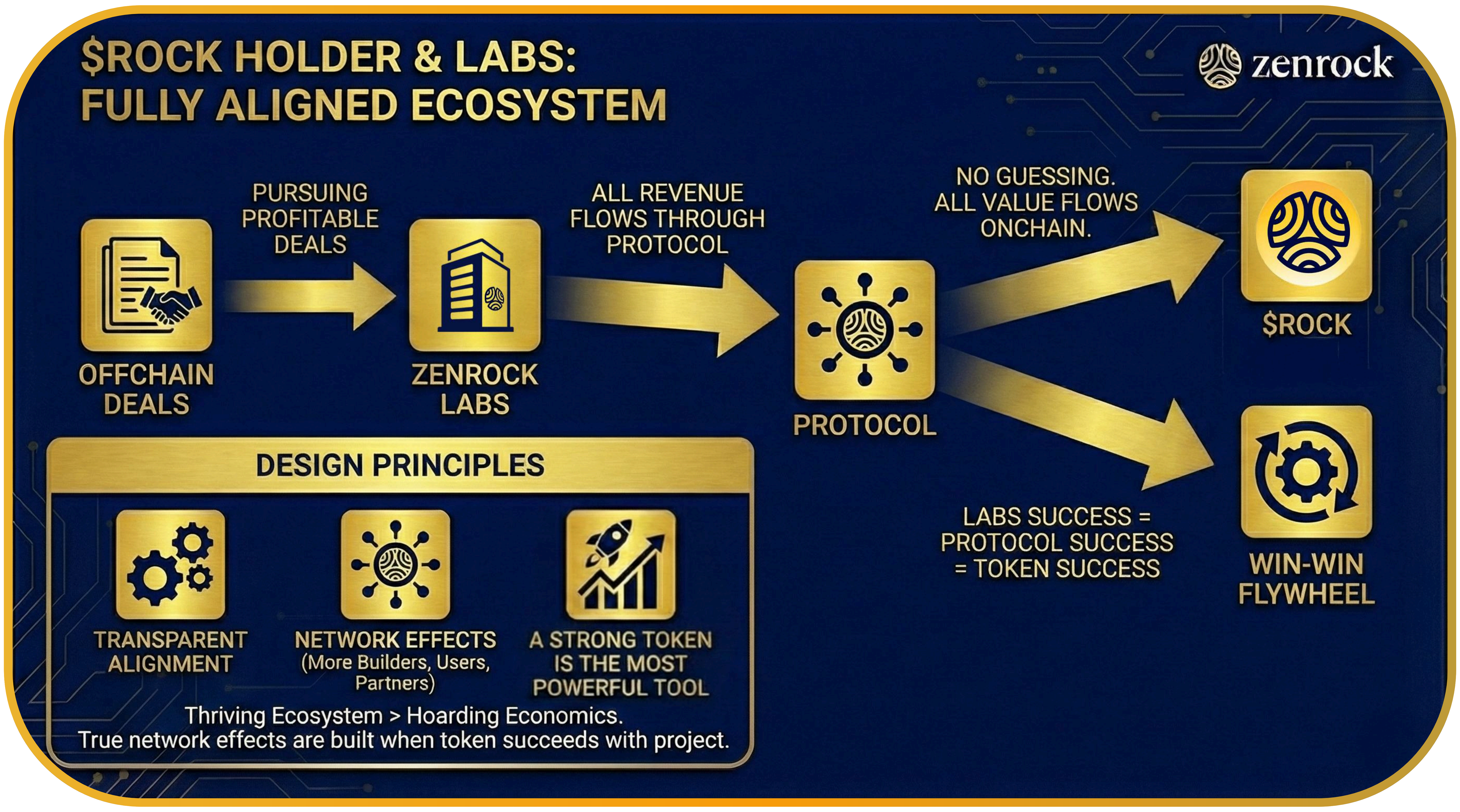
As a $ROCK holder, you do not have to guess whether some unseen side deal is draining your upside. **You want Labs to pursue as many profitable offchain deals as possible, because all of that revenue flows through the protocol as if it originated onchain.**

**To be clear:** this isn't charity. Zenrock Labs is a for-profit business and $ROCK holders are not stakeholders in that business. But we understand a few things that many projects seem to have forgotten:

***A thriving ecosystem benefits Labs far more than hoarding economics ever could.***

***A strong token is the most powerful tool in crypto: It attracts builders, users, partners, liquidity, attention. It creates a flywheel of decentralized support that no amount of centralized management or offchain deal-making can replicate.***

When the token succeeds alongside the project, that's where true network effects are built. That's the system we designed.


$ROCK HOLDER & LABS:
FULLY ALIGNED ECOSYSTEM
zenrock
OFFCHAIN DEALS
PURSUING PROFITABLE DEALS
ZENROCK LABS
ALL REVENUE FLOWS THROUGH PROTOCOL
PROTOCOL
NO GUESSING. ALL VALUE FLOWS ONCHAIN.
$ROCK
LABS SUCCESS = PROTOCOL SUCCESS = TOKEN SUCCESS
WIN-WIN FLYWHEEL
DESIGN PRINCIPLES
TRANSPARENT ALIGNMENT
NETWORK EFFECTS (More Builders, Users, Partners)
A STRONG TOKEN IS THE MOST POWERFUL TOOL
Thriving Ecosystem > Hoarding Economics.
True network effects are built when token succeeds with project.

## Staking $ROCK

Staking yield on $ROCK centers around the **Node Reward Pool (NRP):** a dedicated onchain module pre-funded with 8% of total supply at genesis. The NRP receives 30% of all fees earned on zrChain. All staking rewards flow from the NRP to validators and their delegators.

Staked $ROCK earns rewards from two sources:

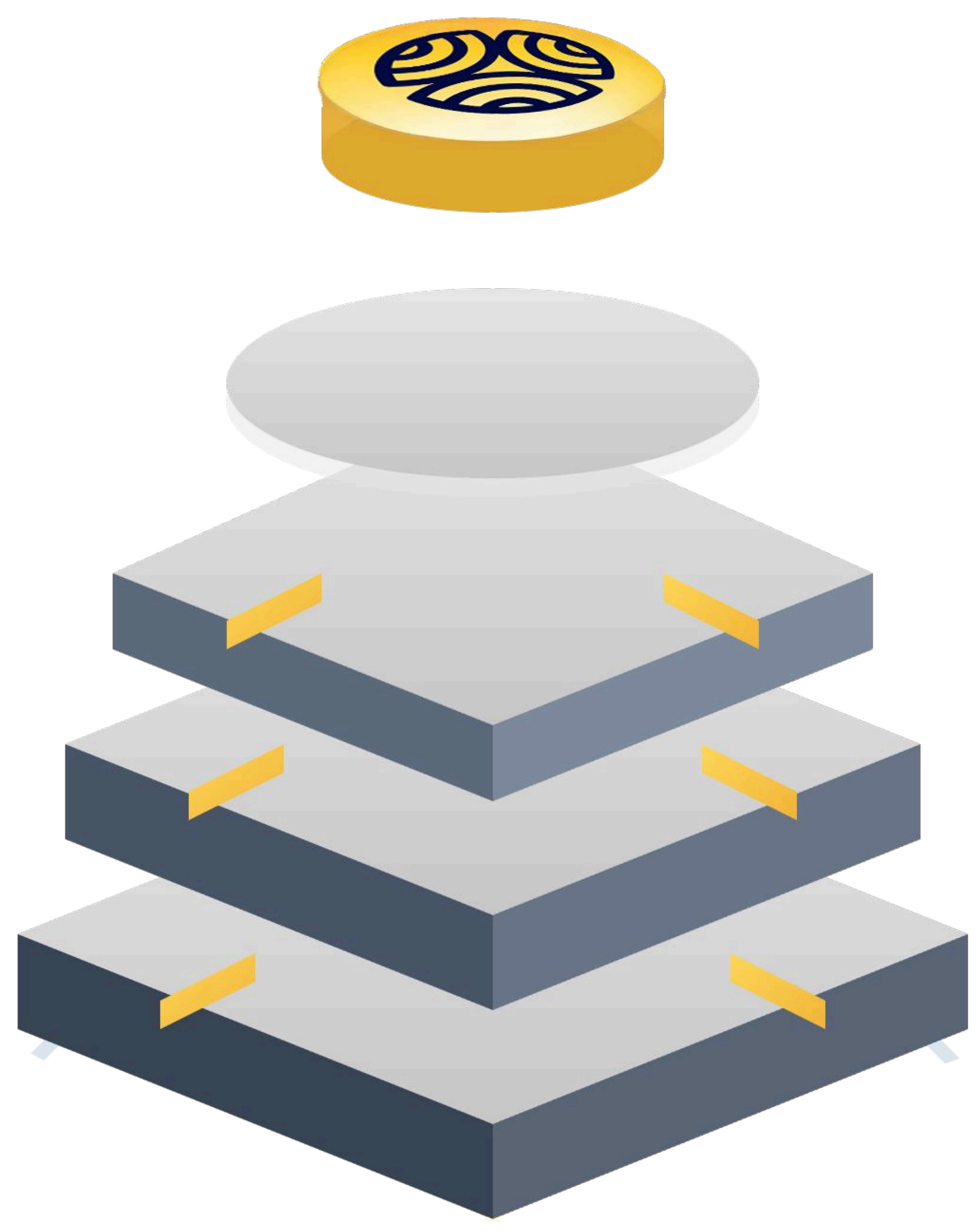
- Baseline Rewards
- Protocol Rewards

The rewards a staker receives at any given point is the sum of the two.

**Baseline Rewards** are fixed block emissions paid to stakers, funded by the NRP's genesis allocation. The per-block reward follows this schedule:

| Effective Date | Block Reward | Annual Emission |
|---|---|---|
| 2026-01-01 | 2.25 ROCK | ~12.2M ROCK |
| 2027-01-01 | 1.0 ROCK | ~5.4M ROCK |
| 2028-01-01 | 0.5 ROCK | ~2.7M ROCK |
| 2029-01-01 | 0.25 ROCK | ~1.35M ROCK |
| 2030-01-01 | 0.1 ROCK | ~0.54M ROCK |
| 2031-01-01 | 0 ROCK | 0 |

Total baseline emissions over the 5-year schedule: ~22.2M ROCK.



**Protocol Rewards** are based on fees earned by the protocol (including offchain revenue routed through the tokenomics). The 30% share of all fees flows to the NRP and is distributed over time according to the following schedule:

- 1/3 paid linearly over 30 days following the fee event
- 1/3 paid linearly over 5 months after that
- 1/3 paid linearly over 30 months after that

Implementation note: To avoid unbounded state growth, fees are aggregated into monthly epochs (30-day periods). Each epoch accumulates all fees received during that period, then distributes them according to the schedule above.

Distribution happens per-block for smooth rewards, but aggregation is monthly. This bounds onchain state to approximately 36 active epoch buckets regardless of transaction volume.

**Example:** Assume during month 1, 100 $ROCK is earned as fees. 30 $ROCK flows to the NRP per the tokenomics. When the epoch closes after 30 days:

- Over the next 30 days (month 2), 10 $ROCK is released at a constant rate (~0.33 ROCK/day)
- Over the following 5 months (months 3-7), 10 $ROCK is released (~0.067 ROCK/day)
- Over the following 30 months (months 8-37), the final 10 $ROCK is released (~0.011 ROCK/day)

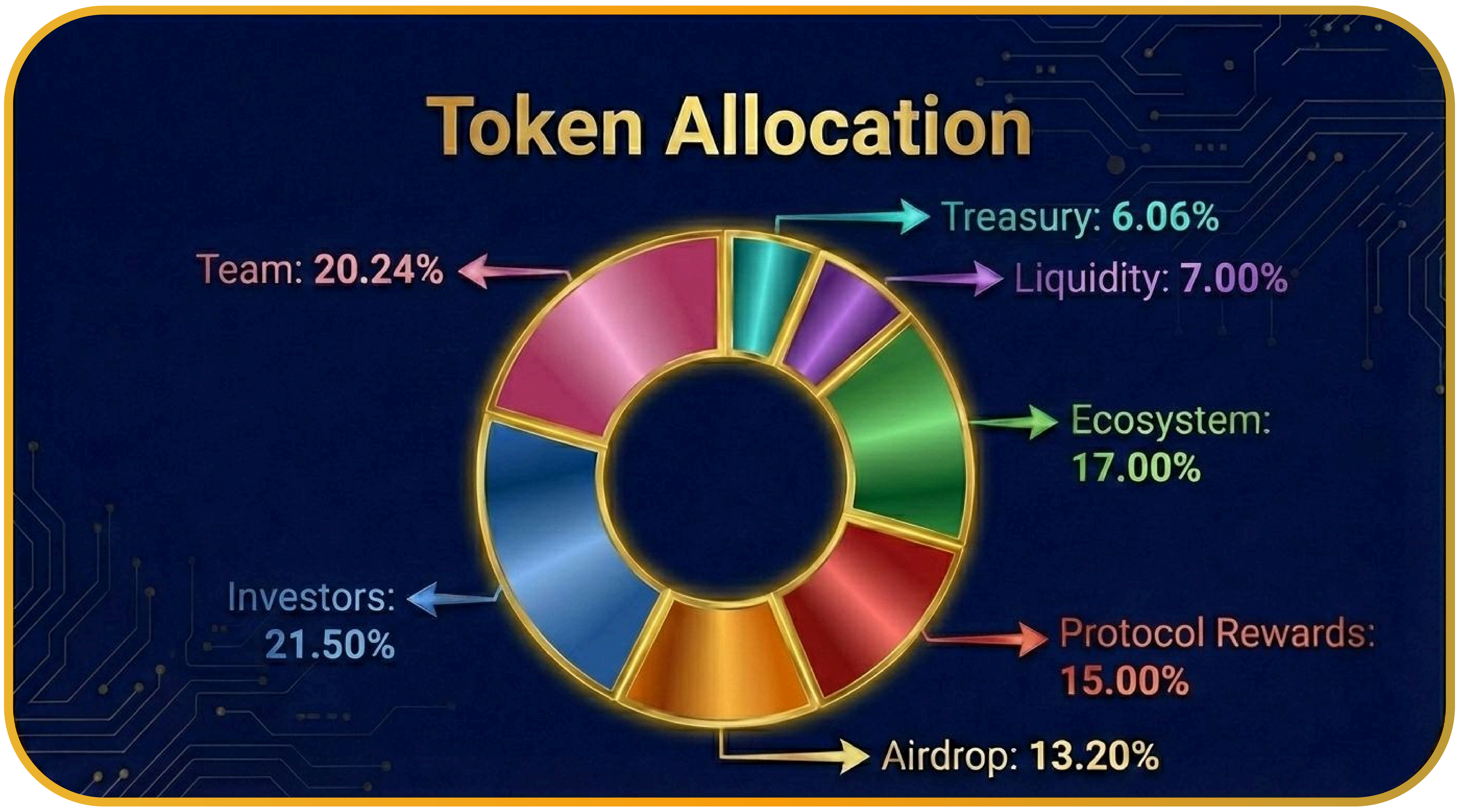
Total distribution period: 36 months from epoch close. This schedule smooths rewards and creates predictable returns, **but more importantly, it creates a structural token sink.**

**Because rewards are distributed over 36 months rather than immediately, there is always a growing backlog of $ROCK committed to future payouts.**

**As protocol activity increases, the pipeline of pending distributions grows. Tokens that would otherwise be liquid are locked in the distribution queue, temporarily removed from circulating supply.**

The more successful the protocol becomes, the larger this sink grows. It's not a temporary lockup or an artificial staking mechanism. It's a perpetual feature of how the protocol works, bringing stability and transparency to changes in circulating supply.

## Token Allocation



Fixed supply of 1 billion $ROCK:

- **Team:** 20.24%
- **Investors:** 21.50%
- **Airdrop:** 13.20%
- **Protocol Rewards:** 15.00%
- **Ecosystem:** 17.00%
- **Liquidity:** 7.00%
- **Treasury:** 6.06%

Insiders (team + investors) are subject to a 12-month cliff followed by 24-month linear vesting. All vesting schedules began 11/11/24.

# Build on zrChain

**Anyone can build on Zenrock and earn a share of what they generate, subject to Zenrock tokenomics.**

## Integration Paths

### Create a New DCT

For teams with existing assets seeking decentralized custody on Solana. Bring any token to Solana with institutional-grade security through dMPC custody. Requires asset specification, outpost infrastructure coordination, and protocol governance approval.

### Build on Existing DCTs

For DeFi protocols, yield strategies, and structured products. Use zenBTC, zenZEC, or future DCTs as primitives for lending, structured products, vaults, and other DeFi applications.

### Build on Hush Protocol

For privacy-focused applications and private payments. Leverage the shielded compute layer to build confidential DeFi: private lending, private trading, anonymous payments.

### Build Something New

The dMPC infrastructure is general-purpose. Identity systems, key management, cross-chain messaging: the infrastructure is live and extensible.

### Revenue Share

**Builders have the ability to receive, or direct the distribution of 30% of all fees their products generate, distributed in $ROCK according to protocol tokenomics. In instances where multiple teams contribute to a product, revenue can be configured at the protocol level.**

### Builder Support

**Zenrock Foundation and Zenrock Labs support builders with marketing, compliance support, go-to-market strategy, and technical integration.** We want you to succeed because when you succeed, the whole ecosystem benefits.

Reach out to begin building.

# zenTP

Use zenTP: [https://app.zenrocklabs.io/zentp](https://app.zenrocklabs.io/zentp)

Zenrock Transfer Protocol (zenTP) enables native $ROCK transfers between zrChain (Cosmos) and Solana. Unlike traditional bridges that wrap tokens, zenTP represents $ROCK as a native SPL token on Solana, providing the same user experience as any native Solana asset.

200M $ROCK were transferred to Solana to seed liquidity as fully bridged, circulating tokens represented as a native SPL token. As users move tokens across the bridge, supply fluctuates on each side, but total supply remains unchanged.

zenTP is the first transfer protocol connecting Cosmos directly to Solana with native token representation (rather than wrapped tokens), leveraging zrChain's dMPC infrastructure for security.

# What's Next

zrChain is live. zenBTC is minting. Hush is shielding.

Custody and privacy are just the beginning. As Zenrock Labs and other teams ship more products, all of it feeds the same tokenomics.

**Near-term focus areas include:**

- **Expanded DCT ecosystem:** Additional assets beyond BTC and ZEC
- **Hush applications:** DeFi primitives built on the shielded compute layer
- **Developer tooling:** Infrastructure to accelerate third-party building
- **Protocol upgrades:** Continued expansion of dMPC operator set and security guarantees

Expect continued shipping.

**More products. More fees. More value to $ROCK.**

**Get started:** https://www.zenrocklabs.io/

*02/03/2026*

[1] This Litepaper reflects the plans and expectations of its authors as of the date listed on the first page. Plans and circumstances can change, so we cannot assure that it will be accurate as to future events, developments, or plans, and it should not be relied on with respect to any such events, developments, or plans.