WRAPPED
ZCASH

WHITEPAPER

WRAPPED ZCASH

DECEMBER 2025

Wrapped Zcash (WZEC) - Whitepaper

Version 1.0 - 2025

Mark Scrine[2] Patrick Collins[3] Hans Friese[4]

[1] Cyfrin

[2] Mark Scrine is the CSO at Cyfrin. He previously led the Proof of Reserve and Real-World Asset initiatives at Chainlink Labs, where he contributed to the design and deployment of decentralized oracle networks securing billions in value. He co-authored this work in his capacity as a specialist in cross-chain system architecture and asset transparency mechanisms.

[3] Patrick Collins is the founder and CEO of Cyfrin, and a prominent educator in Web3 systems engineering. He co-authored this work in his separate capacity as a researcher and advisor focused on secure cross-chain infrastructure, decentralized finance interoperability, developer-first security practices, and smart-contract security

[4] Hans Friese is the creator of **Solodit**, one of the leading platforms for smart-contract security intelligence and audit discovery. He co-authored this work in his capacity as a researcher and engineer specializing in vulnerability analysis, decentralized system integrity, and the hardening of high-value blockchain infrastructure.

1. Abstract

Zcash is one of the most privacy-preserving assets in the digital economy, but its integration with decentralized finance has been limited due to the absence of a trust-minimized, transparent, and secure bridging mechanism. Wrapped Zcash (WZEC) introduces a fully collateralized ERC-20 representation of ZEC, allowing ZEC to interoperate across Ethereum, Layer-2 networks, other EVM-compatible systems and Solana.

WZEC leverages Chainlink Runtime Environment (CRE) and Chainlink Proof of Reserve (PoR) to provide cryptographically verifiable deposit proofs and real-time collateral transparency. Combined with secure mint/burn smart contracts and institutional custody, WZEC establishes a robust and extensible interoperability layer between Zcash and the broader decentralized economy.

Table of Contents

2. Introduction

Zcash is uniquely positioned as a privacy-focused store of value due to its shielded transaction capabilities and mature cryptographic foundations. Despite its strong properties, ZEC remains largely siloed from DeFi. WZEC removes this isolation by introducing a bridge system with:

- Deterministic minting based on verified ZEC deposits
- Deterministic burning for ZEC redemptions
- Cryptographically verifiable collateralization
- Modular upgradeability for future cross-chain support

The goal of WZEC is not simply to replicate wrapped Bitcoin models, but to create a next-generation wrapped asset standard that integrates decentralized oracle verification, transparency, and institution-grade risk controls.

3. System Overview

The Wrapped Zcash (WZEC) system is designed as a *hybrid cryptographic–oracle–custodial interoperability framework* that enables the secure, transparent, and deterministic representation of native Zcash (ZEC) on Ethereum and EVM-compatible execution environments. The system incorporates decentralized oracle reporting, institutional custody, and on-chain enforcement mechanisms to ensure that the supply of WZEC always corresponds exactly to the amount of ZEC held in reserve.

This section provides a detailed overview of the system's core modules and their interactions, forming the foundation for the more rigorous architectural, economic, and security analyses in subsequent sections.

3.1 Custodial Zcash Addresses

To maintain 1:1 collateralization, WZEC employs designated institutionally managed Zcash shielded or transparent addresses, depending on operational requirements. These addresses serve as the canonical entry point for all ZEC deposits that correspond to future WZEC minting events.

Key properties include:

- Deterministic State Observation:
  Each address is continuously monitored by independent Chainlink nodes that observe native Zcash chain state.
- Operational Redundancy:
  Multiple addresses may be deployed to support sharding of deposits, auditability, and multi-institution custody models.

- Cryptographic Isolation:
  Zcash’s Sapling shielded pool ensures privacy for depositors while maintaining verifiable commitment-level state integrity visible to oracle nodes.

3.2 Chainlink Runtime Environment (CRE)

The Chainlink Runtime Environment (CRE) operates as the secure cross-chain execution substrate that captures state transitions on the Zcash chain and transports them to the Ethereum network for deterministic mint execution.

CRE maintains several academically relevant security and correctness properties:

1. **Decentralized Observation:**
   A committee of oracle nodes independently observes Zcash balances and deposit events, reducing reliance on any single observer.
2. **Threshold Aggregation:**
   Observations are aggregated using a cryptographic signature scheme that ensures a signed report is valid only when a threshold of nodes agree on the observed state.
3. **Replay-resistance:**
   Reports include epoch identifiers or nonces that prevent re-submission of previously valid reports, even if captured by an adversary.
4. **Dedicated Forwarder Contract:**
   Only a pre-authorized Chainlink forwarder can submit CRE reports to the WZEC on-chain logic, drastically reducing risk of spoofed or malformed inputs.

This mechanism is essential for establishing the deterministic mapping:

Zcash Deposit → Verified CRE Report → Authorized Mint Event.

3.3 Chainlink Proof of Reserve (PoR)

The Proof-of-Reserve module serves as an independent collateral attestation mechanism operating alongside CRE. While CRE ensures that minting logic is correct, PoR ensures that collateral backing is correct.

PoR fulfills the following roles:

- **State Integrity Assurance:**
  An on-chain PoR feed continuously publishes the amount of ZEC held in custody. This allows external systems (e.g., risk engines, DeFi protocols) to verify solvency conditions without trusting the issuer.
- **On-Chain Circuit Breakers:**
  The WZEC mint contract references the PoR feed and automatically restricts minting if reserves fall below the outstanding WZEC supply.

- **Auditable Transparency for DeFi:**
  Because the PoR data is cryptographically pushed on-chain, any protocol integrating WZEC can programmatically enforce collateralization requirements.

PoR therefore augments CRE by acting as a *complementary real-time solvency guardian*.

3.4 WZEC Mint/Burn Contracts

The WZEC token system is composed of two primary UUPS-upgradeable smart contracts:

WrappedZcash.sol

An ERC-20 token implementation with additional mint/burn functionality and role-controlled access.

Academic properties:

- **Deterministic Supply Discipline:**
  Supply can only change through *mint* events (driven by CRE reports) or *burn* events (driven by user redemptions).
- **Non-Bypassable Access Control:**
  Only the Chainlink CRE consumer contract may call mintTo().
  No human operator or governance contract can mint tokens directly.
- **Upgradeability with Integrity:**
  UUPS ensures future-proofing while strictly isolating mint logic from governance compromise.

ChainlinkCreConsumer.sol

The execution entry point for CRE reports.

Features include:

- Signature Verification: Ensures that reports originate from a valid CRE oracle quorum.
- Report Freshness Enforcement: Nonce/epoch tracking prevents replay.
- Mint Trigger Logic: When validated, the contract calls the WZEC token’s mint function.

3.5 Off-Chain Redemption Processor

Redemption requires reversing the process:

WZEC burn event → Off-chain processing → ZEC release.

The redemption processor:

- Monitors Ethereum logs for BurnedForZec events.
- Validates the destination Zcash address (transparent or shielded).
- Releases ZEC to the user via institutional custody.
- Ensures operational redundancy with failover nodes.

This module provides the final leg of the interoperability cycle, ensuring that WZEC retains full convertibility into native ZEC.

3.6 Solana Integration Layer

Although WZEC initially targets Ethereum and EVM-compatible networks, the architecture is intentionally designed to support non-EVM high-performance blockchains, most notably Solana. Solana's Sealevel parallel runtime, fast block confirmation, and mature DeFi ecosystem make it a strong destination for ZEC liquidity.

The WZEC system can be extended to Solana using two models:

1. **Native Mint Model (Preferred)**

    - A Solana Program acts as a mint/burn authority
    - CRE reports are verified via a Solana-specific oracle consumer
    - WZEC-SPL tokens are minted directly on Solana
    - PoR ensures collateral backing remains chain-agnostic

2. **Canonical EVM → Solana Bridging Model**

    - WZEC is minted on Ethereum
    - A cross-chain bridge locks WZEC
    - Bridged WZEC-SPL is minted on Solana
    - Solana→Ethereum redemptions burn the SPL version

The native mint model offers higher trust minimization and cleaner mint/burn symmetry but requires a native CRE consumer program. This makes Solana a tier-1 expansion target in the WZEC multi-chain strategy.

4. System Architecture

The WZEC architecture is designed as a *modular, deterministic, oracle-secured interoperability pipeline* connecting the Zcash base layer with Ethereum and other EVM ecosystems. The architecture emphasizes separation of concerns, cryptographic verification boundaries, and deterministic state propagation from one chain to another.

At a high level, the system consists of:

1. A custody layer (Zcash chain)
2. An observation and verification layer (CRE + PoR oracles)
3. An execution layer (Ethereum smart contracts)
4. A redemption layer (off-chain processors and Zcash outbound transfers)

Together, these layers form a two-directional value-transfer framework with mathematically enforceable supply discipline and auditing guarantees.

4.1 Deposit → Mint Flow

The deposit-to-mint lifecycle provides the forward pathway for ZEC to enter the WZEC system. It is engineered to minimize ambiguity, eliminate discretionary control, and ensure that all minted WZEC is provably backed by real ZEC.

Step 1: User Deposit on the Zcash Chain

A user sends ZEC to one of the designated WZEC custodial addresses.
Depending on the selected privacy model, the deposit may be:

- A transparent ZEC transfer, allowing direct balance observation.
- A shielded ZEC transfer, where CRE nodes inspect commitment-level state changes and apply traceable logic to detect net deposits.

Step 2: Oracle Observation via CRE

Independent Chainlink nodes read Zcash chain state and detect:

- The deposit amount
- The deposit address
- The updated custodial wallet balance
- The corresponding epoch or block height

Multiple nodes independently derive the same state, ensuring that individual observer compromise cannot corrupt the reported result.

Step 3: Aggregation into a Cryptographically Signed Report

CRE uses threshold-signature or multi-signature aggregation to combine node observations into a single authoritative report.

Each report includes:

- Custody balance delta
- Cumulative custody balance
- Zcash block reference
- A CRE report ID (for replay protection)
- Committee signatures

This aggregation converts a multi-observer state into a *single verifiable truth*.

Step 4: Submission via the Chainlink Forwarder

Only a dedicated Chainlink forwarder contract can submit CRE reports to the on-chain consumer.

This provides:

- A single controlled ingress point
- Reduced attack surface
- Rejection of unsolicited or spoofed submissions

Step 5: On-Chain Validation and Mint Execution

Upon receiving a report:

1. Signature validation ensures threshold agreement.
2. Nonce/epoch validation ensures freshness.
3. Collateralization check consults the PoR feed.
4. Mint authorization is granted only if the report passes all checks.

The consumer contract then calls:

```
mintTo(recipient, amount)
```

Minting becomes a *mathematically bounded function* of the verified ZEC inflow.

Deterministic Minting Guarantee

Minting cannot occur without a valid CRE report, and CRE cannot issue valid reports without observing real Zcash deposits. Therefore:

ZEC-in → WZEC-out
No other pathway exists.

4.2 Burn → Redemption Flow

The burn-to-redemption lifecycle ensures that WZEC maintains full convertibility into native ZEC, preserving its role as a collateralized synthetic asset.

Step 1: User Burn Transaction on Ethereum

A user calls:

burnForZec(amount, zAddr)

This establishes:

- A provable destruction of WZEC
- An explicit recipient Zcash address for redemption
- An on-chain event log for off-chain processors

The transaction is irreversible after confirmation.

Step 2: Contract Validation

The redemption contract performs:

- Address-format verification (transparent or shielded format)
- Supply adjustment (burn reduces total supply)
- Event emission (BurnedForZec) including requester and destination

This event becomes the canonical trigger for off-chain ZEC release.

Step 3: Off-Chain Processor Action

Redemption processors:

- Subscribe to Ethereum logs
- Detect burn events
- Verify event parameters
- Execute custodial ZEC transfers to the specified Zcash address

The processor acts as an operational executor, not as a discretionary authority.

Step 4: Settlement Finality

Once the ZEC transfer is executed, the burn → redemption cycle reaches completion.
No minting path can convert this redeemed WZEC back into supply without a corresponding new ZEC deposit event.

Deterministic Redemption Guarantee

Because burns always reduce supply and always result in outbound ZEC:

WZEC-in → ZEC-out
No other pathway exists.

4.3 Cross-Chain Architecture for Solana

To support Solana, WZEC incorporates the following architectural components:

A. Solana Oracle Consumer Program (Future Component)

A Solana on-chain program mirroring the responsibilities of the Ethereum `ChainlinkCreConsumer.sol`, including:

- verification of CRE signatures
- nonce/replay protection
- stateful tracking of processed reports
- mint authority delegation for the WZEC-SPL token

This ensures that Solana-native mint events follow the same deterministic rules as EVM chains.

B. Solana Mint/Burn Rules

The WZEC-SPL token contract on Solana:

- mints only when a valid CRE report is processed
- burns only via user-initiated redemption instructions
- emits events/logs for off-chain redemption processors
- interacts with PoR to ensure solvency

C. High-Throughput Redemption Path

Solana's parallel runtime enables:

- batch burn processing
- high-frequency redemption events
- faster ZEC outbound flows

D. Optional EVM–Solana Routing Layer

In phase 4–5, a unified router enables:

WZEC (Ethereum) ↔ WZEC-SPL (Solana)
using either:

- Wormhole
- LayerZero
- Cross-Chain Message Passing (future Chainlink CCIP-Solana)

This expands WZEC from a single-chain asset into a cross-ecosystem collateral primitive.

Figure 1 - Global Architecture (EVM + Solana)

```
flowchart TB
  ZC[Zcash Network] --> CZ[Custodial Zcash Addresses]
  CZ --> CRE[Chainlink Runtime Environment (CRE)]
  CRE --> FWD[Chainlink Forwarder]

  FWD --> EVMCON[EVM CRE Consumer]
  FWD --> SOLCON[Solana CRE Consumer Program]

  EVMCON --> WZEC20[WZEC-ERC20 (Ethereum/L2s)]
  SOLCON --> WZECSPL[WZEC-SPL (Solana)]

  WZEC20 -->|burnForZec| OFF[Redemption Processor]
  WZECSPL -->|burnForZec| OFF
  OFF --> ZC
```

Figure 2 - Deposit → Mint Flow (EVM)

```
sequenceDiagram
  participant U as User (ZEC)
  participant C as Custodial Zcash Address
  participant N as CRE Node Committee
  participant A as CRE Aggregator
  participant F as Chainlink Forwarder
  participant X as EVM CRE Consumer
  participant T as WZEC-ERC20

  U->>C: Deposit ZEC
  N->>C: Observe deposits/balance
  N->>A: Submit observations
  A->>F: Threshold-signed report
  F->>X: Deliver report
  X->>T: mintTo(recipient, amount)
  T-->>U: WZEC minted
```

Figure 3 - Burn → Redemption Flow (EVM + Solana)

```
sequenceDiagram
  participant U as User
  participant T as WZEC Token (ERC20/SPL)
  participant O as Redemption Processor
  participant C as Custodial Zcash Address
  participant Z as Zcash Network

  U->>T: burnForZec(amount, zAddr)
  T-->>O: BurnedForZec event
  O->>C: Verify burn + prepare payout
  C->>Z: Send ZEC to zAddr
  Z-->>U: ZEC received
```

Figure 4 - Oracle Layer (CRE + PoR)

```
flowchart LR
  CZ[Custodial Zcash Addresses] --> CREN[CRE Nodes]
  CREN --> CREA[CRE Aggregator]
  CREA --> FWD[Forwarder]
  FWD --> CON[On-chain Consumer]

  CZ --> P◌ोरN[PoR Nodes]
  PopN --> PopF[PoR Feed (On-chain)]
  PopF --> CON

  CON -->|if PoR >= supply| MINT[Mint Enabled]
  CON -->|if PoR < supply| PAUSE[Mint Paused]
```

Figure 5 - Solana Native Mint Architecture

```
flowchart TB
  FWD[Chainlink Forwarder] --> SOLCON[Solana CRE Consumer Program]
  SOLCON --> SPLMINT[WZEC-SPL Mint Authority]
  SPLMINT --> USER[User receives WZEC-SPL]

  USER -->|burnForZec| SPLBURN[WZEC-SPL Burn]
  SPLBURN --> OFF[Redemption Processor]
  OFF --> ZC[Zcash payout]
```

Figure 6 - DeFi Integration

```
flowchart LR
  U[User holds WZEC] --> DEX[AMMs\nUniswap/Curve/Balancer]
  U --> LEND[Lending\nAave/Morpho/Silo]
  U --> PERP[Derivatives\nPerps/Options]
  U --> BRIDGE[Cross-chain Routers\nL2s/Solana]

  DEX --> U
  LEND --> U
  PERP --> U
  BRIDGE --> U
```

5. Oracle Layer Design

The oracle layer is the epistemic foundation of WZEC. It defines *how the system knows that ZEC exists*, *how it verifies custody*, and *how it safely propagates this knowledge to Ethereum*. The design builds upon the Chainlink Runtime Environment (CRE) and Chainlink Proof of Reserve (PoR), forming a dual-oracle structure that provides both *state transmission* and *state verification*.

The oracle layer establishes **two independent cryptographic assurances**:

1. **CRE (State Propagation Assurance)**
   Ensures that only real Zcash deposits trigger minting.
2. **PoR (Collateral Integrity Assurance)**
   Ensures that the entrusted custodian actually holds the ZEC backing all circulating WZEC.

Together, they guarantee that WZEC operates as a **verifiable, non-inflationary, and fully collateralized wrapped asset**.

5.1 Cross-Chain Environment Model

The Cross-Chain Runtime Environment (CRE) is designed to solve the fundamental problem of cross-chain communication: transmitting verified state from chain A to chain B without creating a new trusted intermediary.

For WZEC, CRE performs the following critical functions:

A. Distributed Observation

CRE nodes independently monitor:

- Zcash custodial wallet balances
- Zcash block height and finality progress

- Deposit transactions to monitored addresses
- Any anomalies or inconsistencies in address activity

This distributed approach ensures that no single node can unilaterally corrupt or manipulate observations.

B. Threshold-Signed Report Creation

Each epoch or deposit event generates a *deterministic data payload*, which is aggregated via:

- Threshold BLS signatures
- Multi-signature ECDSA (depending on configuration)
- Deterministic report hashes
- Nonce/epoch counters

This results in a **single canonical report** representing multi-node agreement.

C. On-Chain Report Verification

The Ethereum-side consumer contract performs:

- **Signature validation** against the CRE committee public key
- **Epoch verification** to ensure freshness
- **Replay protection** (nonces or report IDs)
- **Integrity checks** against expected custodial state

Invalid reports cannot execute state changes.

D. Authorized Forwarding

Only the Chainlink Forwarder contract can deliver CRE reports on-chain.
This architecture removes:

- Unauthenticated report submissions
- Front-running manipulation
- Malicious replay attempts

E. Deterministic State Translation

CRE does not “interpret” state; it merely **transmits** verified Zcash balance deltas.
This ensures:

- Minting = exactly observable net inflow of ZEC
- No discretionary reporting
- No subjective interpretation of Zcash activity

CRE therefore provides mathematically bounded cross-chain consistency.

5.2 Proof of Reserve Integration

Chainlink Proof of Reserve (PoR) introduces an additional cryptographic guarantee: **that the ZEC claimed to exist in custody actually exists**.

While CRE transmits *changes in state*, PoR verifies *absolute state*.

A. Independent Custody Verification

PoR oracles:

- Query custodial account balance
- Validate the balance independently from CRE
- Publish a live PoR feed on Ethereum
- Ensure that WZEC circulating supply never exceeds backing reserves

This is analogous to continuous automated auditing.

B. On-Chain Enforcement

WZEC contracts may enforce:

- **Mint ceilings**: Minting disabled if supply > PoR balance
- **Circuit breakers**: Minting paused if PoR feed deviates too far
- **Emergency halt**: Pauses triggered on oracle failure cases

These mechanisms reduce systemic risk and help protect WZEC holders even under extreme black-swan conditions.

C. Custodial Risk Mitigation

PoR transforms the custodian from a trust-based entity into a *verified entity*.
This mitigates:

- Insolvency risk
- Mismanagement risk
- Inaccessibility or operational failure

Without requiring users to trust the custodian directly.

D. Transparent Economic Boundary

PoR ensures the WZEC supply cannot exceed real ZEC deposits.
This creates:

- A transparent monetary boundary
- A permanently enforced 1:1 asset ratio
- A credibly neutral collateralization model

CRE + PoR = both motion and position of collateral.
Both are required for a fully verifiable wrapped asset.

6. Security Model

The WZEC security model integrates cryptographic guarantees, oracle decentralization, custodial verification, and upgradeability constraints to produce a robust, adversary-resistant cross-chain asset pipeline.

Security is evaluated along the following axes:

- Oracle Integrity
- Custodial Verifiability
- Smart Contract Correctness
- Systemic Interactions
- Upgrade Controls

6.1 Threat Model

A. Oracle Manipulation

Attacker attempts:

- To falsify deposit events
- To inject false balances
- To submit spoofed reports

Mitigations:

- CRE decentralized multi-node consensus
- Threshold signatures
- Forwarder-only submission
- Full on-chain verification

B. Unauthorized Minting

Attacker attempts to mint without deposits.

Mitigations:

- No mint function exposed to public
- Minting only executable by CRE consumer

- CRE consumer only executable from the forwarder
- PoR-backed ceilings prevent over-minting

C. Custodial Insolvency or Malfeasance

Mitigations:

- Continuous PoR auditing
- Multi-custodian future expansion
- Circuit breakers and failsafes

D. Report Replay or Reordering

Mitigations:

- Nonce/ID-based replay protection
- Stateful CRE consumer tracking
- Epoch-based freshness rules

E. Smart Contract Failures

Mitigations:

- UUPS upgradeability for patches
- External audits
- Formal verification potential

6.2 Core Security Properties

Property 1: Deterministic Minting

Minting can only occur when:

1. A valid ZEC deposit is observed
2. CRE reports it
3. PoR verifies full collateral backing

No discretionary minting path exists.

Property 2: Deterministic Burning

Once burned, WZEC **cannot be recreated** without a new ZEC deposit.
Supply reduction is irreversible.

Property 3: Guaranteed 1:1 Collateralization

PoR enforces that circulating WZEC ≤ total ZEC reserves.

Property 4: Non-Bypassable Mint Pathway

All mint requests flow through:

- CRE → Forwarder → Consumer → WZEC

There are no alternative pathways.

Property 5: Oracle-Verified Convertibility

WZEC maintains continuous, oracle-verified parity with ZEC.

7. Economic Model

WZEC is designed as a **fully collateralized, non-inflationary, cryptographically verified synthetic asset**. Its economics follow predictable, rule-based behavior.

A. Supply Discipline

Total supply is strictly bounded by ZEC reserves:

Supply(WZEC) = CustodialBalance(ZEC)

B. Predictable Monetary Policy

WZEC has:

- No algorithmic stability mechanism
- No synthetic leverage
- No endogenous collateral

This positions WZEC as a **low-volatility, low-risk wrapped asset similar to WBTC—except with transparency improvements via PoR**.

C. Fee Model

Governance may introduce:

- Mint fees
- Burn fees
- Transfer routing fees (for cross-chain expansions)

Fees fund:

- Oracle costs
- Custodial operations

- Upgrades and audits

D. DeFi Composability

WZEC can serve as:

- Collateral in lending markets (Aave, Morpho, Silo)
- Liquidity in AMMs (Uniswap, Curve, Balancer)
- A settlement asset for derivatives
- A margin asset

- A bridging asset across EVM chains

E. Demand Drivers

WZEC demand increases with:

- Growth of Zcash as a store of value
- Privacy-preserving DeFi
- Institutional usage of PoR-backed assets
- Layer-2 scaling adoption

8. Future Economic Extension: ZUSD Stablecoin

WZEC is primarily designed as a fully collateralized, cross-chain representation of ZEC, focusing on capital movement. The natural evolution of this primitive is the introduction of a stablecoin, ZUSD, which would be collateralized by WZEC. This stablecoin would function as a native, privacy-preserving, and oracle-verified medium of exchange across all WZEC-supported ecosystems.

A. ZUSD Concept and Rationale

ZUSD would be a synthetic US Dollar-pegged asset, minted against WZEC collateral. This design choice leverages WZEC’s core properties:

- **Privacy-Preserving Utility:** ZUSD enables users to engage in stable-value transactions that are natively settled on EVM and non-EVM chains, while the underlying collateral (ZEC) maintains its privacy-centric features.
- **Collateral Efficiency:** WZEC's PoR-verified 1:1 backing with ZEC makes it a cryptographically strong, non-inflationary asset for over-collateralization.
- **Deepened DeFi Integration:** A WZEC-collateralized stablecoin increases WZEC's utility, creating a native market for its risk-managed use as collateral and expanding the

total addressable market for ZEC liquidity.

B. Proposed Stability Mechanism

The ZUSD system would likely adopt a **Collateralized Debt Position (CDP)** model similar to existing decentralized stablecoins.

- **CDP Creation:** Users deposit WZEC into a ZUSD vault to mint ZUSD. The system would enforce an over-collateralization ratio (e.g., 150%) to absorb volatility.
- **Oracle Integration:** The ZUSD system would rely on Chainlink Price Feeds for the WZEC/USD price and potentially for PoR status.
- **Liquidation Mechanism:** Vaults falling below the minimum collateralization ratio would be programmatically liquidated, with WZEC sold to stabilize the ZUSD peg. This mechanism would be auditable and transparent, leveraging the existing oracle layer design.

C. Phased Deployment

ZUSD is envisioned as a later-stage component, fitting into the architecture post-multi-chain expansion:

- **Phase 1-4 Prerequisite:** Full multi-custodian and multi-chain expansion (Phase 4) must be complete to ensure WZEC has sufficient liquidity and security robustness to serve as the base collateral.
- **Phase 5 Extension:** The ZUSD smart contracts (Vaults, Liquidator, Stability Fee mechanisms) would be deployed as an extension of the existing WZEC system, ensuring architectural separation while utilizing the same security primitives (PoR, UUPS).

The introduction of ZUSD would solidify WZEC's role as the foundational, oracle-verified bridge for Zcash, expanding its function from a wrapped asset into a full-spectrum interoperability primitive.

9. Governance and Upgradeability

Governance controls:

- Contract upgrade path
- Oracle configuration
- Custodial sets
- Fee parameters
- Emergency response

8.1 UUPS Upgradeability

The Upgradeable Proxy Standard (UUPS) allows logic upgrades without changing:

- Token balances
- State variables
- Deployed address

This is essential for:

- Patching vulnerabilities
- Migrating oracle architectures
- Supporting new chains
- Enhancing mint/burn logic

All upgrades must pass:

- Governance signature
- Security review
- Compatibility validation

8.2 Governance Roles

A. Owner

Controls:

- Upgrades
- Oracle configuration
- Custodian registry

B. Minter Role

Held exclusively by the CRE Consumer.

C. Forwarder Role

Whitelisted address allowed to submit CRE reports.

D. Future Roles

Potential:

- Guardian (emergency pause authority)
- Multisig governance council
- DAO-based long-term control

10. Roadmap

A phased deployment ensures stability and security during ecosystem expansion.

Phase 1: Launch

- Ethereum Mainnet
- CRE integration
- PoR-backed custody
- Audits + bug bounties

Phase 2: DeFi Integration

- Liquidity pools on major AMMs
- Lending listings
- Institutional onboarding

Phase 3: Layer-2 Expansion

Deploy WZEC to:

- Solana
- Base
- Arbitrum
- Optimism
- Scroll
- zkSync

Phase 4: Multi-Custodian Architecture

Introduce diversified custody:

- Multiple Zcash custodians
- Multi-signature custody
- PoR-transparent multi-party holdings
- Define Solana custody model & Solana-native PoR extension

Phase 5: Cross-Chain Asset Routing

- Cross-chain mint/burn between EVM chains
- Integration with CCIP (future path)
- Research into IBC-style ZEC interoperability
- Solana Integration
- Deploy WZEC-SPL mint/burn program
- Integrate CRE consumer on Solana (or equivalent oracle system)
- Launch Solana-native liquidity pools (Jupiter, Orca)

- Establish Solana-to-Ethereum canonical routing
- Explore CCIP-Solana integration once supported
- This phase positions WZEC as a multi-runtime wrapped asset bridging privacy liquidity into both EVM and non-EVM environments.

11. Conclusion

WZEC introduces a new generation of wrapped asset architecture, improving upon first-generation models like WBTC by incorporating *decentralized oracle verification*, *automated collateral auditing*, and *upgradeable security primitives*. By transforming Zcash from a siloed privacy asset into a fully interoperable DeFi primitive, WZEC enables:

- Transparent, verifiable, and fully collateralized wrapped value
- Privacy-preserving capital flows
- Layer-2 liquidity expansion
- Institutional-grade trust minimization

WZEC is positioned to become the **definitive wrapped privacy asset** across the multi-chain economy.