Architecture of a Decentralized Data Provision System on Polkadot with Halo 2 ZK, Cross-Chain, and Governance Circuits

Ovcharov Vladimir (c) 2025
Institute of Cybernetics, Kyiv, Ukraine

July 1, 2025

Abstract

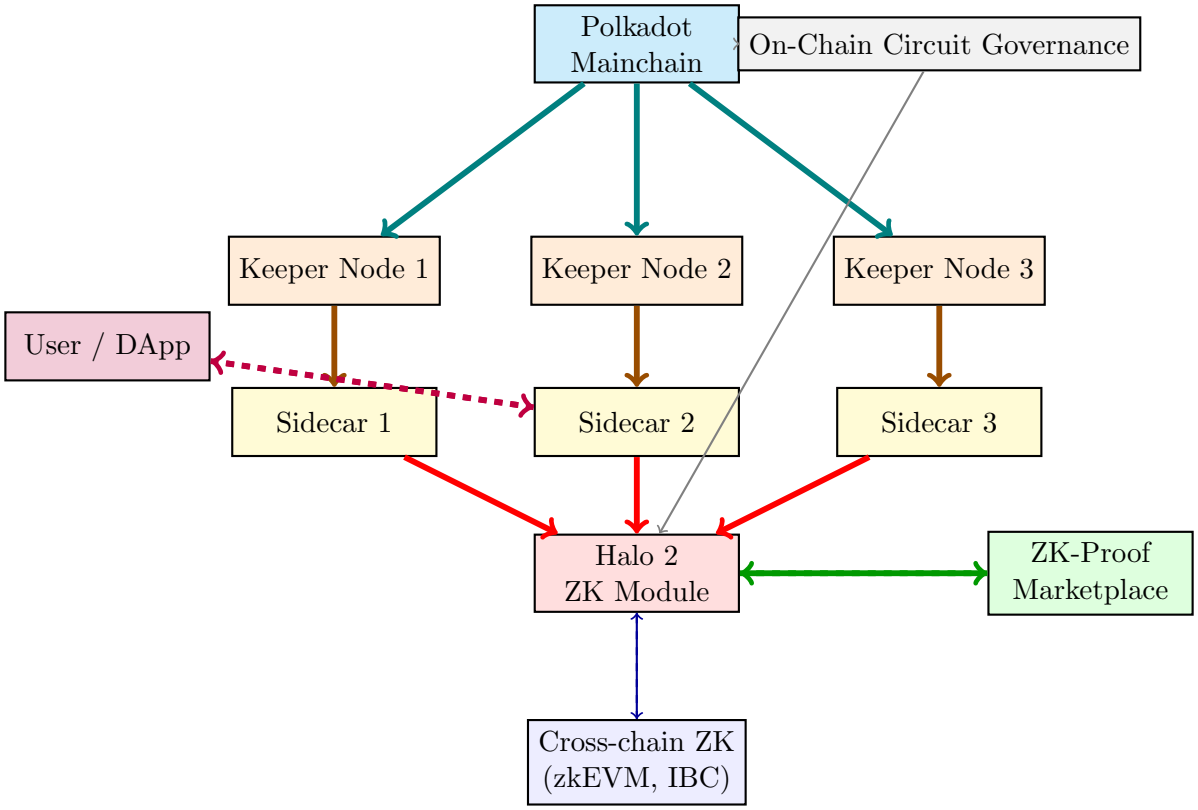
This document presents a next-generation architecture for decentralized, privacy-preserving data sharing and provisioning. The design leverages a Polkadot-based mainchain, modular sharded storage, fast Sidecar services, and an advanced zero-knowledge (ZK) layer powered by Halo 2. Key innovations include: recursive proofs for batching, on-chain governance for circuit approval, cross-chain ZK-proof interoperability (e.g., zkEVM), and a ZK-Proof Marketplace for scalable decentralized validation and monetization of proofs. This enables composable, auditable, and future-proof privacy infrastructure for data-driven applications.

1 System Overview

- **Polkadot/Substrate Fork**: Core mainchain for consensus, staking, reward logic, cross-chain messaging, and governance.
- **Keeper Nodes**: Sharded, vectorized, encrypted data storage with proof-of-uptime, sharding, and batching.
- **Sidecar Services**: High-speed search, caching, REST/gRPC API, and Halo 2 ZK proof handling.
- **Halo 2 ZK Module**: All privacy, authorization, and reward proofs built and validated using modular Halo 2 circuits.
- **Cross-Chain ZK Integration**: Native support for verifying/relaying ZK-proofs to and from zkEVM, other Substrate parachains, and Cosmos/IBC.
- **Circuit Governance**: On-chain DAO process for proposing, auditing, and approving new ZK circuits.
- **Recursive Proofs**: Batch aggregation of proofs for scalable, cheap group verification.
- **ZK-Proof Marketplace**: Decentralized market for proof validation, aggregation, and monetization.

- **Tokenomics**: Utility token for payment, rewards, staking, circuit voting, and marketplace fees.

2 Architecture Diagram



3 API Specification (Halo 2 Focus)

3.1 REST/gRPC API Endpoints (Sidecar/Halo 2)

Endpoint	Method	Description
/api/v1/search	POST	Vector similarity search, includes Halo 2 ZK proof, returns top-K results and access tokens.
/api/v1/access	POST	Shard access via validated Halo 2 proof.
/api/v1/report	POST	Keeper reports (data served, uptime, ZK activity).
/api/v1/challenge	POST	Storage or uptime challenge with Halo 2-based proof.
/api/v1/circuit_vote	POST	Propose/vote on new ZK circuits (governance).
/api/v1/market_submit	POST	Submit proof for validation in ZK-Proof Marketplace.
/api/v1/xchain_relay	POST	Submit/verify cross-chain ZK proof (to zkEVM, IBC, etc).

3.2 Sample ZK Proof Request (Halo 2)

```
POST /api/v1/search
{
  "query_vector": [0.25, 0.41, ...],
  "top_k": 8,
  "halo2_proof": {
    "circuit_id": "circuit123abc",
    "proof_bytes": "0xabcd...",
    "public_inputs": [...]
  }
}
```

4 Halo 2 ZK Module and Circuits

- **All ZK-proofs in the network are generated using Halo 2 circuits**. Each circuit defines allowed operations (access control, reward logic, aggregation).
- **Governance controls which circuits are accepted:** only DAO-approved circuits are considered valid for on-chain/marketplace validation.
- **Upgradeability:** new circuits can be introduced, deprecated, or replaced through on-chain voting.
- **Recursive proofs:** batch many actions into a single proof for group reward distribution or audit (reducing fees, speeding up consensus).

5 Cross-Chain ZK-Proofs and zkEVM Integration

- **Relaying**: Submit Halo 2 proofs to external chains (e.g., zkEVM, Cosmos, IBC) to authorize data access or prove actions cross-chain.
- **Verification**: Accept ZK-proofs from other chains, mapped to local access/reward events.
- **Composability**: Use cross-chain proofs in DeFi, on-chain data markets, oracles, and more.
- **Circuit Registry**: Mapping of circuit hashes/IDs between networks for permissioning.

6 ZK-Proof Marketplace: Why and How

- **Purpose:**
  - Offload expensive proof verification/aggregation from mainchain.
  - Allow independent validators (market actors) to earn by validating/aggregating ZK proofs.

  - Provide scalable, decentralized "proof-as-a-service" for DApps, DeFi, data market clients.

- **Workflow:**
  1. Proof submitter pays a small fee and submits a Halo 2 proof to the marketplace.
  2. Validators compete (or are selected) to verify/batch proofs.
  3. Once confirmed, results are relayed on-chain or cross-chain, and validators earn reward.
  4. DAO can add/remove validators, set fees, and audit marketplace rules.

- **Why needed:**
  - Removes bottleneck from mainchain, enabling massive scaling and fast batching.
  - Makes it possible to monetize specialized verification hardware/services.
  - Enables fair, transparent proof auditing and market-driven fees.

7 Security and Upgradability

- **Sybil protection**: All validators and marketplace actors must stake tokens; reputation tracked on-chain.
- **Circuit governance**: Circuits only used after DAO audit, reducing risk of backdoors/bugs.
- **Batch and recursive proofs**: Reduce fees, prevent spam, and speed up epoch close.
- **On-chain audit**: All proof and market actions are logged for compliance and review.
- **Upgradeable circuits**: Swappable via governance, allowing rapid reaction to cryptographic advances.

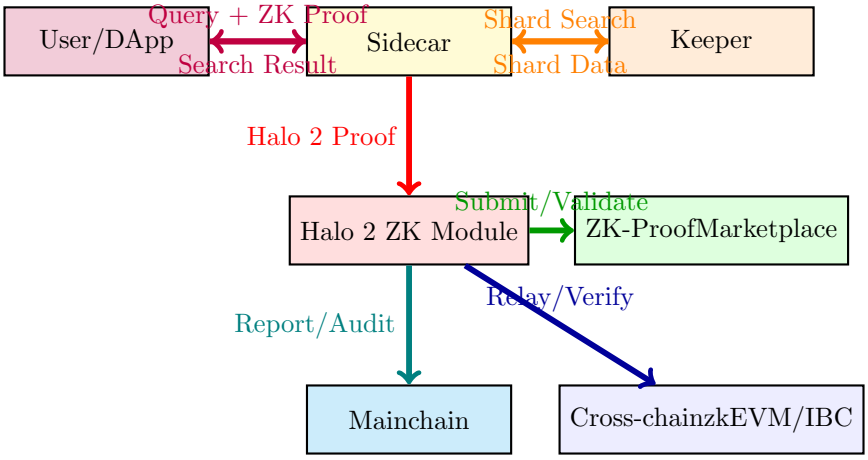
8 Step-by-Step Implementation Guide

1. **Deploy Mainchain**: Fork Substrate, integrate Halo 2, DAO governance, cross-chain, marketplace pallets.
2. **Build Keeper/Sidecar Nodes**: Sharded encrypted storage, vectorization, Halo 2 client, cross-chain relay.
3. **Set up ZK-Proof Marketplace**: Smart contracts, validator registry, batching engine.
4. **Circuit governance**: Launch DAO, implement circuit proposal/vote/audit flow.
5. **Connect clients and DApps**: ZK-enabled APIs, cross-chain logic, proof submission UI.

6. **Test recursive and cross-chain proofs**: Simulate batch actions, verify remote integration.

9 Diagrams

9.1 Sequence Diagram: Proof-Driven Access, Cross-Chain, and Marketplace



10 References

- Halo 2 Documentation: `https://zcash.github.io/halo2`
- ZK-Proof Marketplace Example: `https://github.com/privacy-scaling-explorations/halo2`
- zkEVM: `https://docs.polygon.technology/docs/zkEVM`
- Polkadot/Substrate: `https://substrate.dev/docs`
- Cosmos/IBC: `https://docs.cosmos.network`