A Gas-Efficient Light Client for Cross-Chain Application

Ganesha Upadhyaya
ganesha@harmony.one
Who am i?

●  Research Engineer @ Harmony (https://harmony.one)

●  What do I do at Harmony?
  ○  Lead Harmony’s Interoperability Initiative (past 6 months)
    ■  Bridges & light clients
  ○  Building & Optimizing Harmony blockchain
    ■  Consensus, sharding, staking, on-/off-chain data,
  ○  Ecosystem
    ■  SDKs, oracles & other middlewares

●  MS & PhD in CS
  ○  MS: concurrent programming
  ○  PhD: scaling program analysis

●  Industries: Futurewei (Huawei), Amazon, SAP Labs, HP Labs

●  Interests: Blockchain, Programming Languages, Compilers, Software Engineering
What is Harmony?

Layer-1 blockchain w/

- 2 second block time & transaction finality
- Decentralized network of 1000 nodes
- Sharding (4 shards, 250 nodes each, uniform security)
- BFT consensus (optimized)
- Staking (Proof-of-stake)
In this talk: Harmony Light Client

**Context:** Cross-chain application, e.g., Horizon: A Harmony <> Ethereum Bridge

**Bridge scenarios**

- **Ethereum to Harmony**
  - User wants to lock their assets on Ethereum (e.g., DAI)
  - Mint equivalent amount of assets on Harmony (e.g., 1DAI)

- **Harmony to Ethereum**
  - User burns the bridged assets on Harmony
  - Redeem their locked assets on Ethereum

**Ethereum light client**

**Harmony light client**
Harmony Light Client

Requirements

● Trustless
  ○ User only need to trust the two blockchains
  ○ On-chain verification

● Gas efficient
  ○ Don’t ask the user to pay for more than 1 ERC20 transfer

● Fast
  ○ Don’t make the user wait hours or days
Ethereum Light Client

1. Sync chain data
2. Lock transaction $\tau_{lock}$
3. Proof of lock $\Pi_{lock}$
4. Mint transaction $T_{mint}$
5. Contract $S$

Sync chain data via SPV headers, lock transaction via $\tau_{lock}$, and mint transaction via $T_{mint}$.
Harmony Light Client

1. Sync chain data
2. Burn transaction $T_{burn}$
3. Proof of burn $\Pi_{burn}$
4. Unlock transaction $T_{unlock}$
5. Unlock $x$ tokens
6. Contract $S$

Full Node

Client

Harmony

Ethereum

Relay

Checkpoint headers

Checkpoint headers
Harmony Light Client

- A smart contract deployed on Ethereum
- Has the latest view of Harmony blockchain for Ethereum
- Validates user’s claims regarding Harmony transactions

- User burns bridged tokens on Harmony
- Constructs or obtains a proof-of-burn
- Proof-of-burn includes
  - Proof of tx inclusion in the block’s tx root
  - Proof of block inclusion in the Harmony chain
- Harmony light client must verify proof-of-burn before token unlock
  - Proof of tx inclusion is simple merkle proving (cheap)
  - Proof of block inclusion is BLS pairing-based verification (expensive)
Harmony Light Client

How gas-efficient?

- **Epoch syncing**
  - Syncing light client = Latest block signers
  - Block signers set remains same within every epoch, 1 epoch = 32,768 (2s blocks)

- **BLS signature aggregation**
  - BLS sig verification to accept a block header
  - Aggregate BLS public keys according to bitmap
  - Perform 1 BLS pairing with aggr. public keys and aggr. Signature
  - Still expensive?

- **Merkle Mountain Range (MMR) checkpointing**
  - All blocks within an epoch form a super merkle tree (MMR)
  - Proof of block inclusion = merkle proving over MMR
Harmony Light Client

1 epoch = 32768 blocks

Epoch block
Last block of the previous epoch
Contains committee for next epoch

Default checkpointing
Happens on epoch blocks
Harmony Light Client

- Epoch block
- Last block of the previous epoch
- Contains committee for next epoch

MMR_0 → H_0

MMR_1 → H_1

MMR_2 → H_2

... → H_n

User burn tx

Default checkpointing
Happens on epoch blocks

Wait for next checkpoint
or
Do self BLS verification
Thank you & Resources

- Horizon paper

- Horizon github
  - [https://github.com/harmony-one/horizon](https://github.com/harmony-one/horizon)

- Horizon medium article
  - [https://medium.com/harmony-one/horizon-a-gas-efficient-trustless-bridge-for-cross-chain-transactions-e791296ab43c](https://medium.com/harmony-one/horizon-a-gas-efficient-trustless-bridge-for-cross-chain-transactions-e791296ab43c)

- Layer-2 usecase enabled using Harmony light client
  - [https://medium.com/harmony-one/harmony-as-interoperable-layer-2-for-ethereum-47c10c9b68c8](https://medium.com/harmony-one/harmony-as-interoperable-layer-2-for-ethereum-47c10c9b68c8)