이더리움 확장성 이슈 및 샤딩

한겨레 kr8534@gmail.com

Agenda

- Ethereum Scaling Challenges
- Ethereum Sharding
  - Block Proposer
  - Cross-shard Communication
  - Stateless Client
  - Fork Choice Rule of Shard Chain

Ethereum Scaling Challenges

Issues on Ethereum

CryptoKitties

Release

트랜잭션 정체현상 폭증

BAT

ICO

Gas Price: 580,000 GWei
=> $25000를 TX fee로 지불

Ethereum Scaling Challenges

Issues on Ethereum

실생활에 필요한 Dapp 지원 불가

거래가 지연될수록 **증가하는 거래비용**

탈중앙 지향 가상화폐의 **중앙화 현상** 심화

Ethereum Scaling Challenges

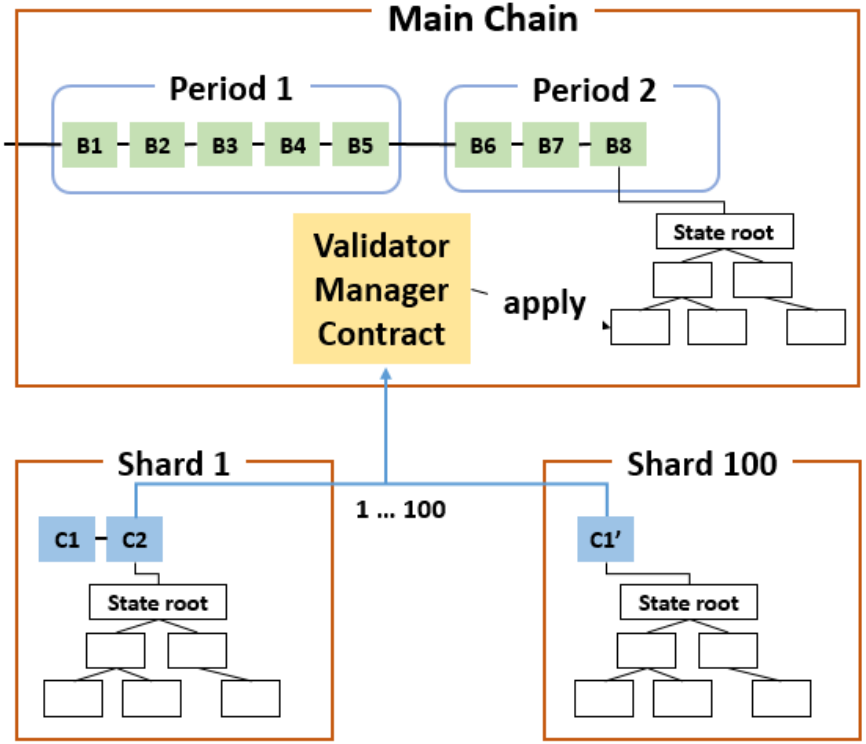
Current Attempts to Resolve

Name	Method	Description
Casper	**On**-chain	작업증명 방식에서 지분증명 방식으로 전환
Raiden Network	**Off**-chain, **Channel**-based	사용자간 처음과 마지막 거래만 기록하되, 중간 거래는 **Off**-chain의 채널에서 sign된 거래 교환으로 진행
Sharding	**On**-chain, **Chain**-based	검증자는 블록체인 state의 일부만 검증
Plasma	**Off**-chain, **Chain**-based	트리 구조의 다중 블록체인을 구축

Ethereum Sharding

Overview

출처: Ethereum Sharding: Overview and Finality

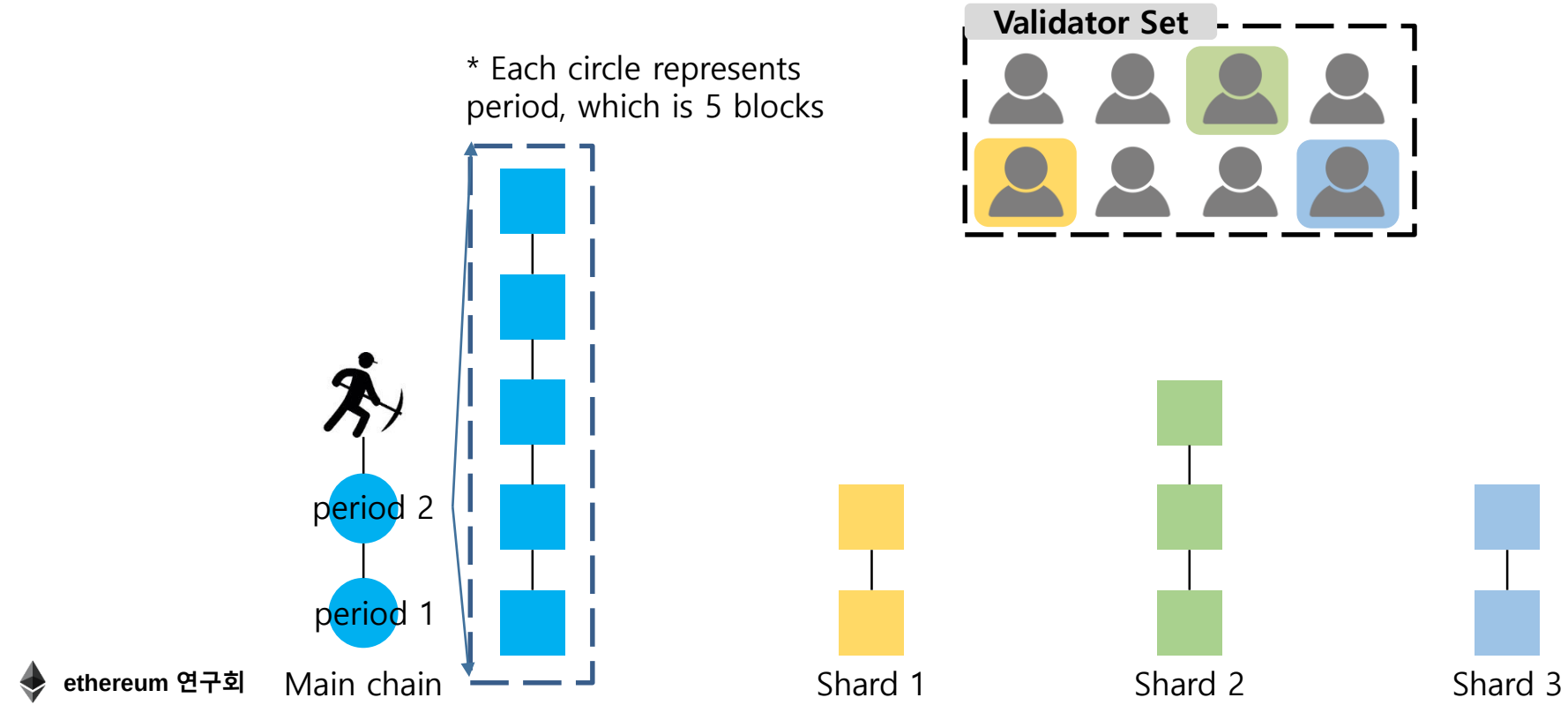


- No need to fork
- **Validator Manager Contract** 을 통해 mainnet에 바로 적용 가능
- Phase 1 will provide
  - A set of shard validators
  - 100 Ethereum shards
  - Each shard will have "stateless clients", "account abstraction"

Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup



Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup

1. **Validators use LOOKAHEAD to check which shards they will be validating**
2. Validators download latest shard state
3. Validators verify blocks until some depth and pick head to build on
4. Client submits transaction with access list and witness
5. Validator pull relevant transactions from the mempool
6. Validators submit collation header to the root chain

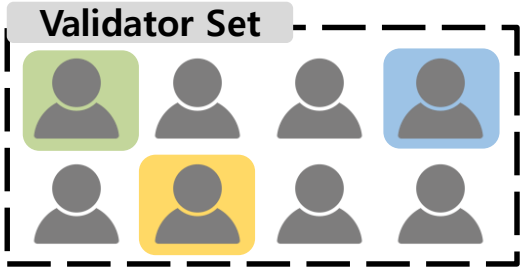
Ethereum Sharding

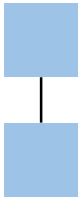
Block Proposal

출처: Asia-Pacific Ethereum Meetup

LOOKAHEAD_PERIODS = 4

* LOOKAHEAD는 Validator가 어느 시점에, 어느 샤드를 봐야 하는지 미리 알려주기 위함





Main chain

Shard 1

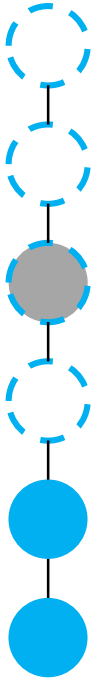
Shard 2

Shard 3

Ethereum Sharding

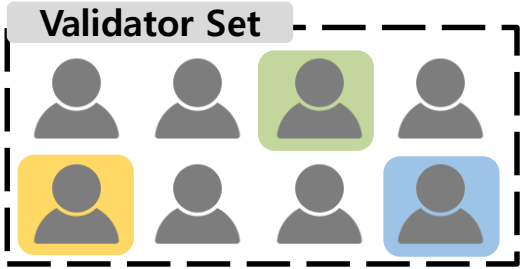
Block Proposal

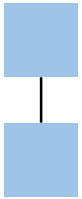
출처: Asia-Pacific Ethereum Meetup



LOOKAHEAD_PERIODS = 4

* LOOKAHEAD는 Validator가 어느 시점에, 어느 샤드를 봐야 하는지 미리 알려주기 위함





Main chain

Shard 1

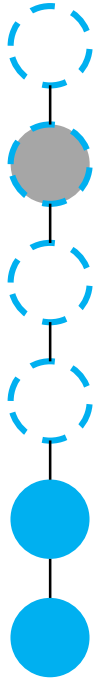
Shard 2

Shard 3

Ethereum Sharding

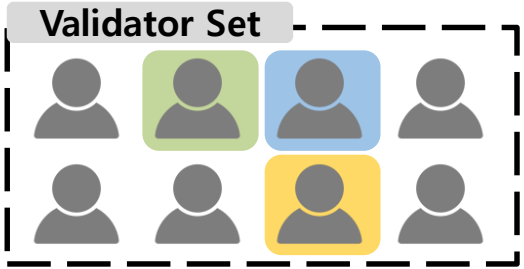
Block Proposal

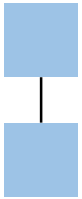
출처: Asia-Pacific Ethereum Meetup



LOOKAHEAD_PERIODS = 4

* LOOKAHEAD는 Validator가 어느 시점에, 어느 샤드를 봐야 하는지 미리 알려주기 위함





Main chain

Shard 1

Shard 2

Shard 3

Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup

1. Validators use LOOKAHEAD to check which shards they will be validating
2. **Validators download latest shard state**
3. Validators verify blocks until some depth and pick head to build on
4. Client submits transaction with access list and witness
5. Validator pull relevant transactions from the mempool
6. Validators submit collation header to the root chain

Ethereum Sharding

Block Proposal

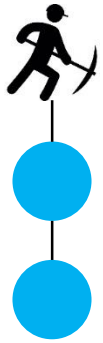
출처: Asia-Pacific Ethereum Meetup

1. Validators use LOOKAHEAD to check which shards they will be validating
2. Validators download latest shard state
3. **Validators verify blocks until some depth and pick head to build on**
4. Client submits transaction with access list and witness
5. Validator pull relevant transactions from the mempool
6. Validators submit collation header to the root chain

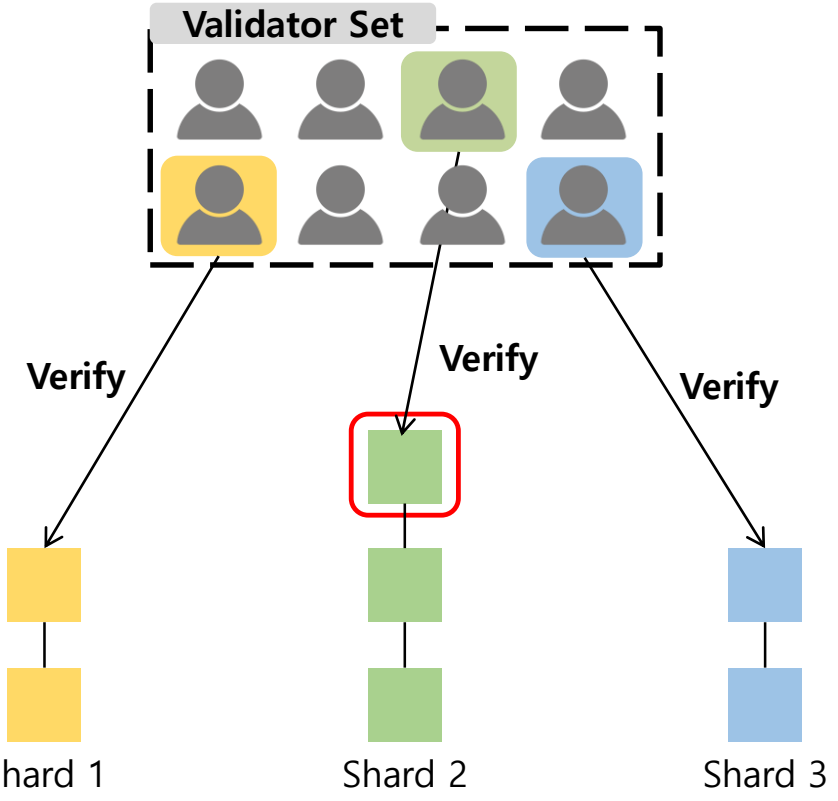
Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup



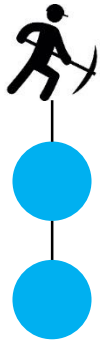
Main chain



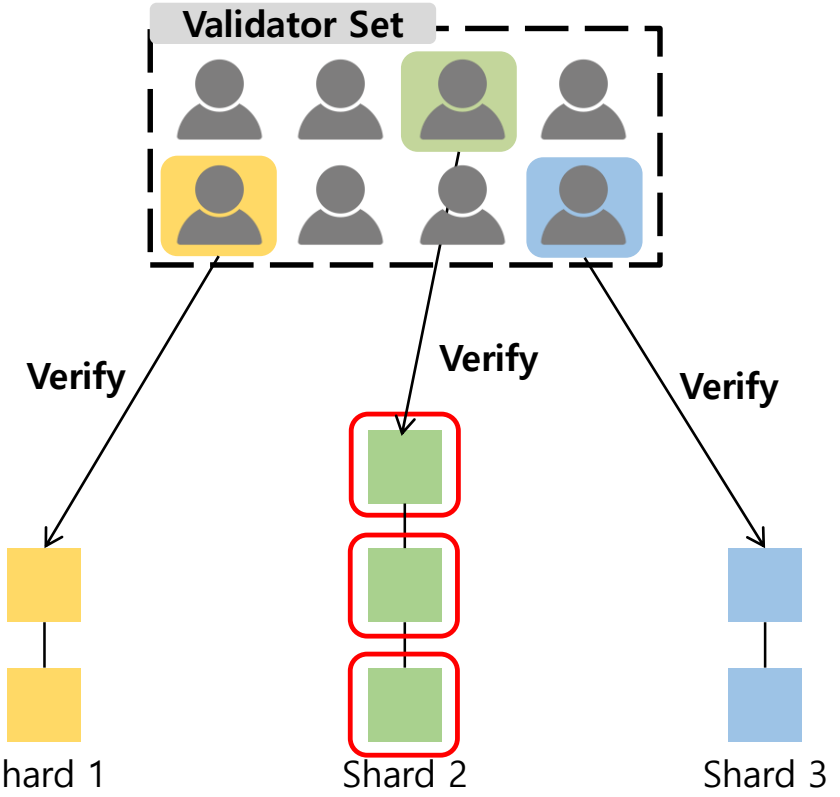
Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup



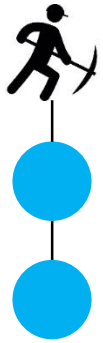
Main chain



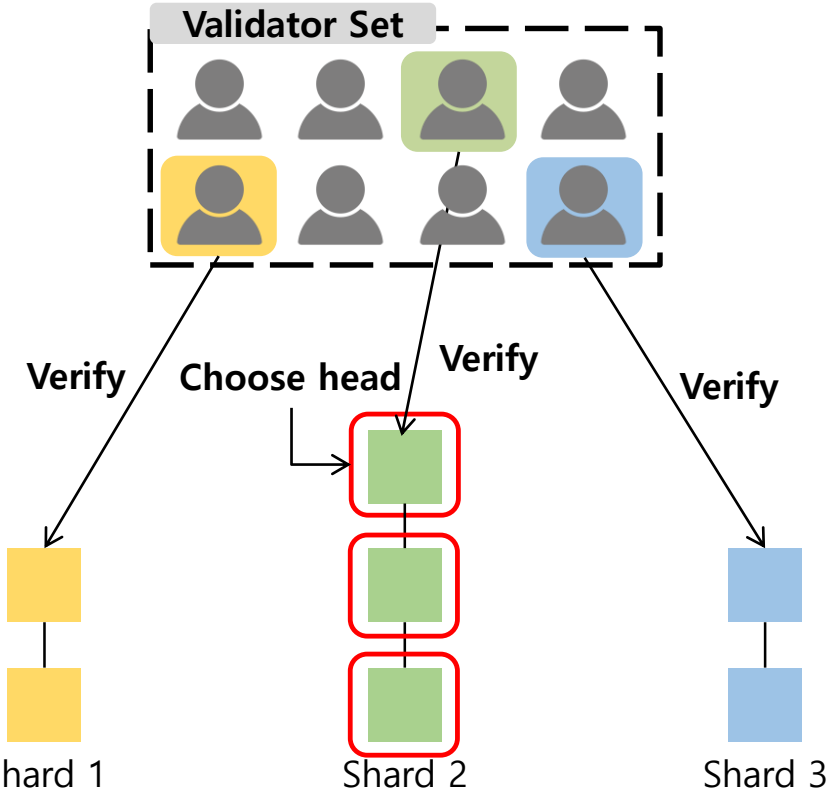
Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup



Main chain



Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup

1. Validators use LOOKAHEAD to check which shards they will be validating
2. Validators download latest shard state
3. Validators verify blocks until some depth and pick head to build on
4. **Client submits transaction with access list and witness => backup slides**
5. Validator pull relevant transactions from the mempool
6. Validators submit collation header to the root chain

Ethereum Sharding

Block Proposal

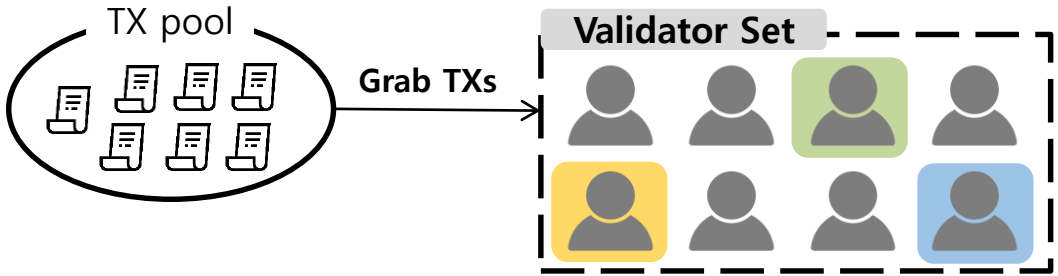
출처: Asia-Pacific Ethereum Meetup

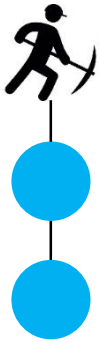
1. Validators use LOOKAHEAD to check which shards they will be validating
2. Validators download latest shard state
3. Validators verify blocks until some depth and pick head to build on
4. Client submits transaction with access list and witness
5. **Validator pull relevant transactions from the mempool**
6. Validators submit collation header to the root chain

Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup

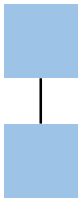




Main chain

Shard 1

Shard 2



Shard 3

Ethereum Sharding

Block Proposal

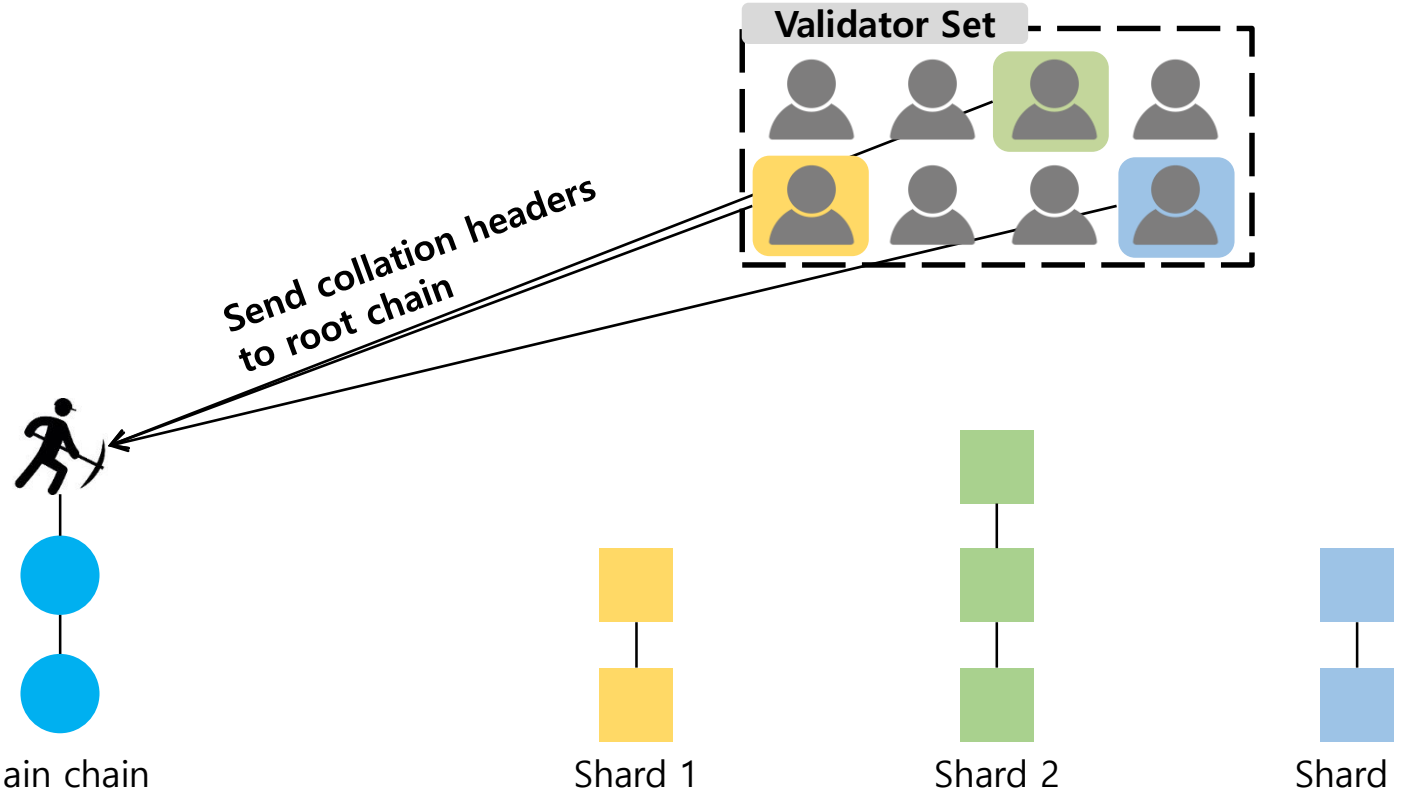
출처: Asia-Pacific Ethereum Meetup

1. Validators use LOOKAHEAD to check which shards they will be validating
2. Validators download latest shard state
3. Validators verify blocks until some depth and pick head to build on
4. Client submits transaction with access list and witness
5. Validator pull relevant transactions from the mempool
6. **Validators submit collation header to the root chain**

Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup



Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup

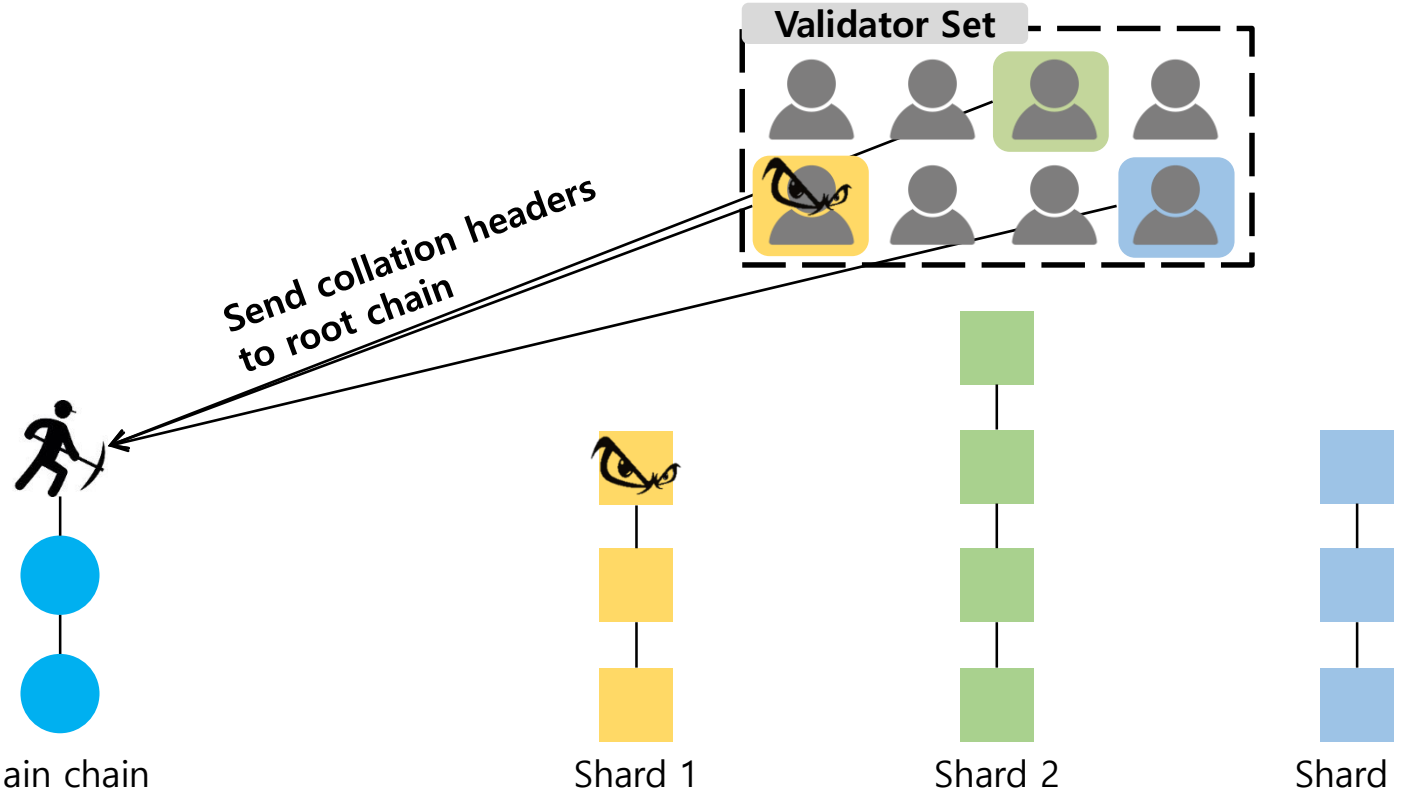
1. Validators use LOOKAHEAD to check which shards they will be validating
2. Validators download latest shard state
3. Validators verify blocks until some depth and pick head to build on
4. Client submits transaction with access list and witness
5. Validator pull relevant transactions from the mempool
6. Validators submit collation header to the root chain

How to handle the case that evil validator submits invalid block?

Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup



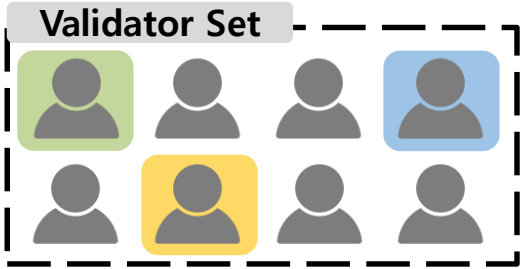
Ethereum Sharding

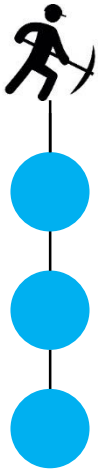
Block Proposal

출처: Asia-Pacific Ethereum Meetup

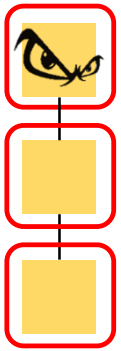
1) Validation 단계에서,
shard 1의 validator가
invalid collation을 notify

2) 새로운 분기 생성

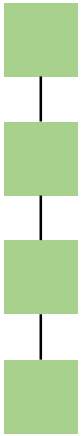




Main chain



Shard 1



Shard 2

Shard 3

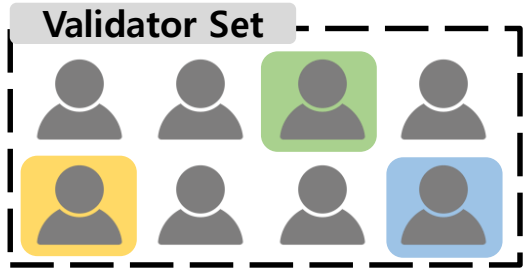
Ethereum Sharding

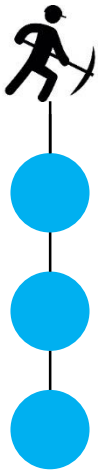
Block Proposal

출처: Asia-Pacific Ethereum Meetup

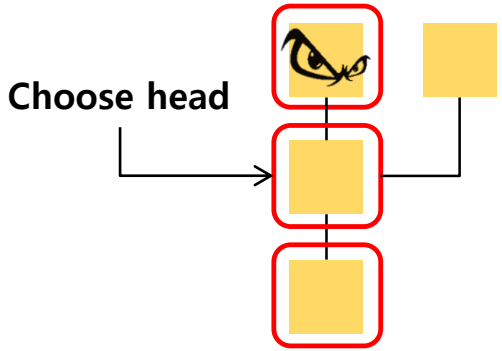
1) Validation 단계에서,
shard 1의 validator가
invalid collation을 notify

2) 새로운 분기 생성

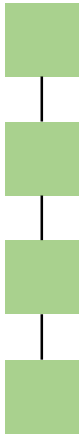




Main chain



Shard 1



Shard 2

Shard 3

Ethereum Sharding

Block Proposal

출처: Asia-Pacific Ethereum Meetup

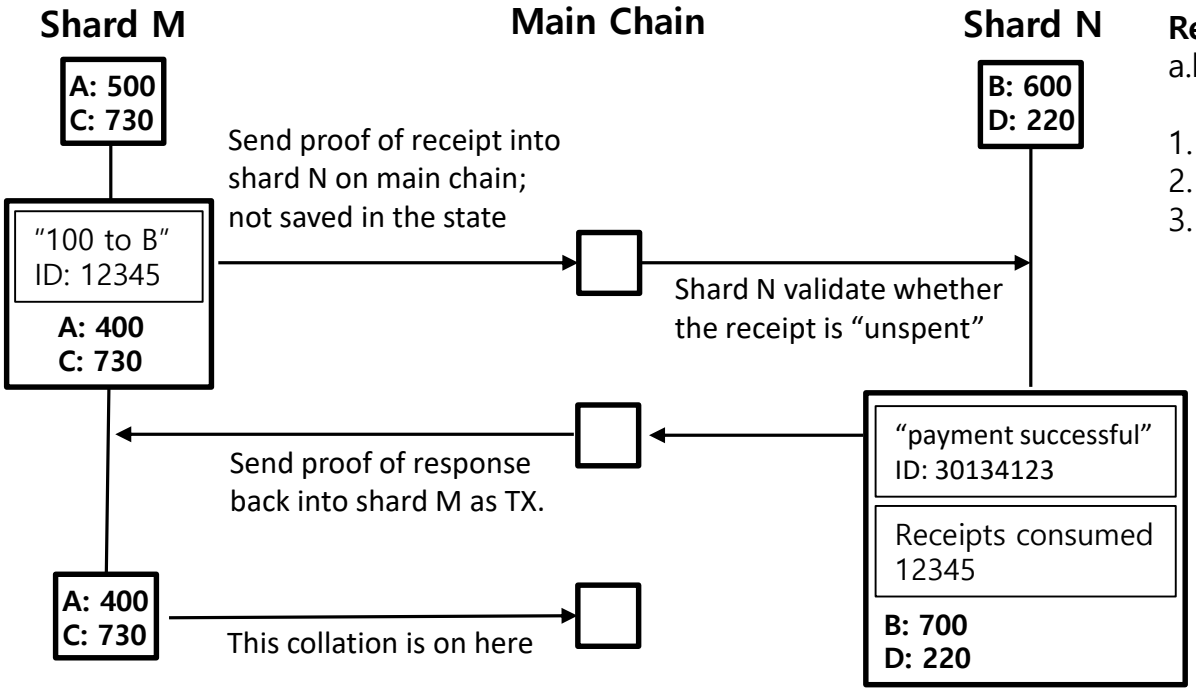
1. Validators use LOOKAHEAD to check which shards they will be validating
2. Validators download latest shard state
3. Validators verify blocks until some depth and pick head to build on
4. Client submits transaction with access list and witness
5. Validator pull relevant transactions from the mempool
6. Validators submit collation header to the root chain

How to handle the case that evil validator submits invalid block? → **How Casper works on this!**

Ethereum Sharding

Cross-Shard Communication

출처: Sharding FAQ



Receipt-based communication
a.k.a "debit" or "credit" base.

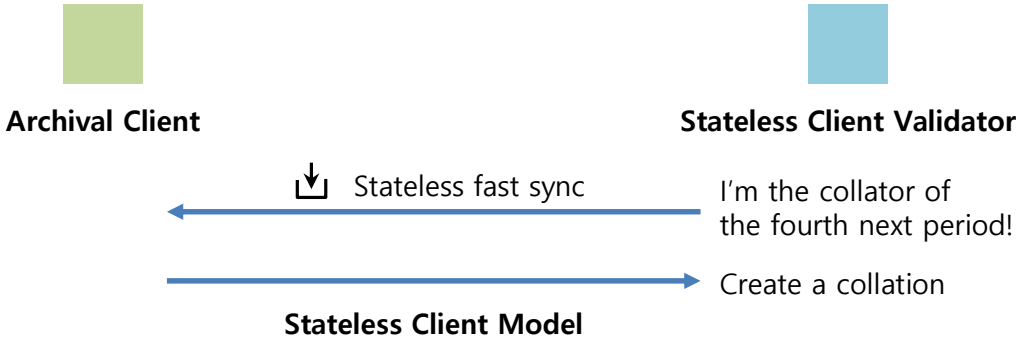
1. Send a TX on Shard M and create a receipt
2. Wait for the first TX to be included
3. Send a TX on shard N which includes the proof of the receipt. (validate the receipt & increase appropriate balance)

Ethereum Sharding

Stateless Client

출처: Ethereum Sharding: Overview and Finality

- **Basic Concept**
  - Stateless clients only store the state trie root
  - The archival clients store the full state trie and provide the merkle branches that the given collation needs
  - With these branches, the stateless clients are able to build partial state trie and verify
  - Validator only have to validate the recent collations to sync with the shard



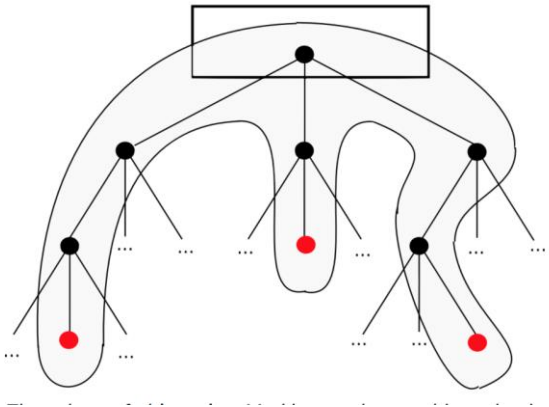
Ethereum Sharding

Stateless Client – Data Format

```
Transaction:
[
   [nonce, act, data, ...], # TX body
   [node1, node2, node3, ...]  # witness
]
```

```
Collation:
[
   [shard_id, ..., sig], # header
   [tx1, tx2, ...],  # TX list
   [node1, node2, ...]  # witness
]
```

- Transaction **must** specify an ***access list*** enumerating the parts of the state
- "**witness**", an RLP-encoded list of Merkle tree nodes that provides the portion of the state



"**Access list**" allows the collator to process the tx with only the state root

RLP(Access list)
=> witness

Ethereum Sharding

Stateless Client – State Transition Function

- **AS-IS:** STF(S, B) -> S' where S, S' are states, B is a block (or could be a transaction T)

- **TO-BE** (because, in a stateless client model, nodes do not store the state)
  S-> the state root of S
  B-> (B,W), where W is a "witness" – a set of merkle branches proving the values
  STF-> STF', which takes as input a state root and a block-plus-witness, and outputs the new state root

Ethereum Sharding

Stateless Client – Pros

- Miners and full nodes in general no longer need to store any state
- Blockchain economics can focus purely on pricing bandwidth and computation
- Disk I/O is no longer a problem for miners and full nodes; (Disk I/O has historically been the primary source of DoS attack)
- Security is increased by reshuffling clients between shards

Ethereum Sharding

Stateless Client – Who Store State?

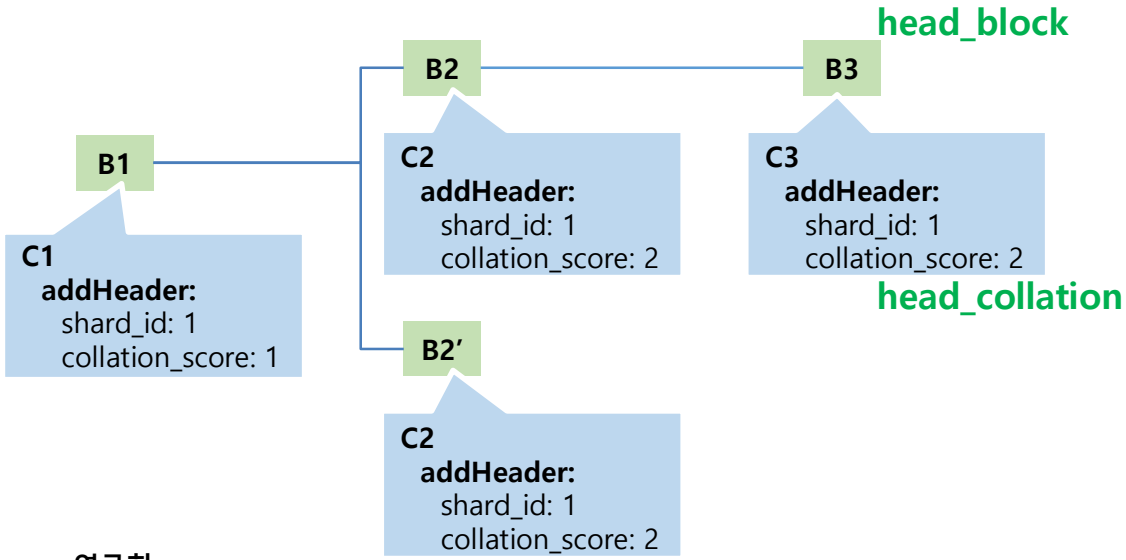
- Basically, “voluntary base”, “welfare storage”
- Any new state trie object by default stored by all full nodes for 3 months
- After 3 months, clients can forget randomly
  - After 12 months, still stored by 25% of nodes, 5% after 60 months
  - Clients can set up channels with paid archival nodes and make it last forever
- Dapp users randomly store some portion of storage keys

Ethereum Sharding

Fork Choice Rule of Shard Chain

출처: Ethereum Sharding: Overview and Finality

- The fork choice rule depends on the longest main chain.
- Not simply the head collation of "longest valid shard chain", but **"The longest valid shard chain within the longest valid main chain"**

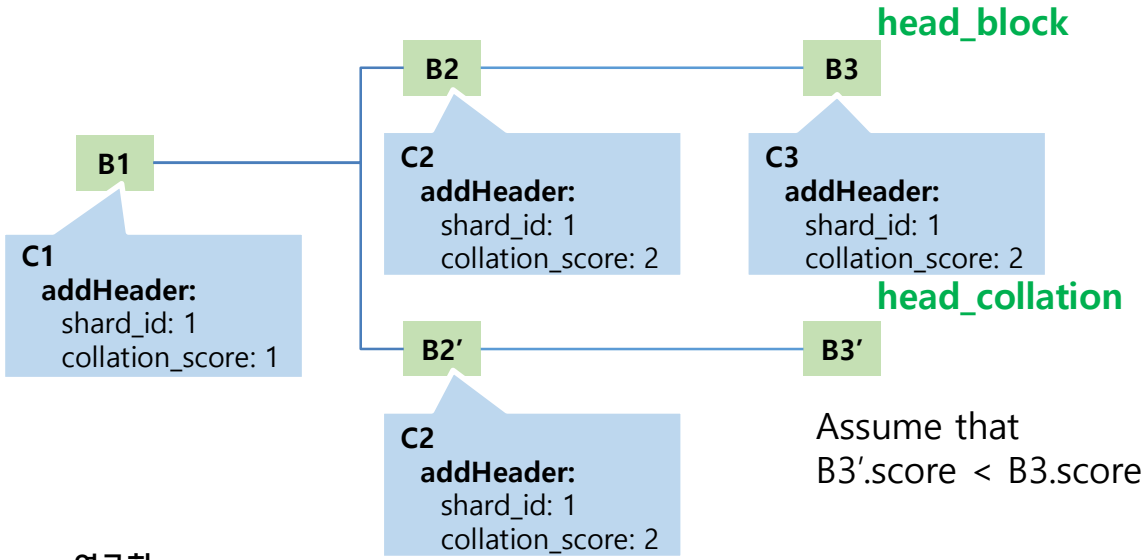


Ethereum Sharding

Fork Choice Rule of Shard Chain

출처: Ethereum Sharding: Overview and Finality

- The fork choice rule depends on the longest main chain.
- Not simply the head collation of "longest valid shard chain", but **"The longest valid shard chain within the longest valid main chain"**

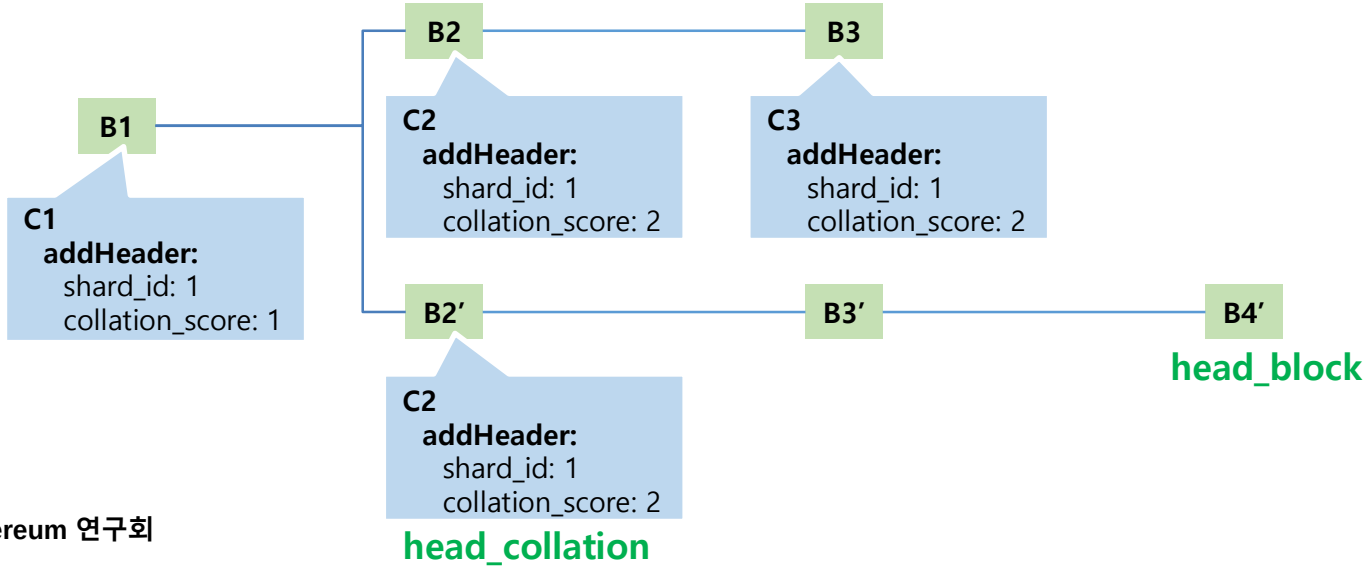


Ethereum Sharding

Fork Choice Rule of Shard Chain

출처: Ethereum Sharding: Overview and Finality

- The fork choice rule depends on the longest main chain.
- Not simply the head collation of "longest valid shard chain", but **"The longest valid shard chain within the longest valid main chain"**



Ethereum Sharding

Issues on Sharding

출처: Ethresear.ch

Many on-going topics on Ethereum Sharding, please visit ethresear.ch

- Merge blocks and synchronous cross-shard state execution
- Proposal/confirmation separation: a bug and a fix
- Delayed state execution, finality and cross-chain operations
- Delayed state execution in practice
- Fork-free sharding
- Cross Shard Locking Scheme
- State-minimised executions

References

- **Ethresear.ch**
- https://medium.com/@icebearhww/ethereum-sharding-and-finality-65248951f649
- https://github.com/ethereum/sharding/blob/develop/docs/doc.md
- https://github.com/ethereum/wiki/wiki/Sharding-FAQ
- https://github.com/ethereum/wiki/wiki/chain-fibers-redux
- https://github.com/ethereum/sharding/tree/develop/sharding
- https://medium.com/l4-media/making-sense-of-ethereums-layer-2-scaling-solutions-state-channels-plasma-and-truebit-22cb40dcc2f4
- https://www.youtube.com/channel/UC7tELjcjz84KlbQJf0t-euQ
- https://medium.com/cryptokitties/cryptokitties-birthing-fees-increases-in-order-to-accommodate-demand-acc314fcadf5
- https://etherscan.io/
- https://bravenewcoin.com/news/ethereum-difficulties-send-ether-tumbling/
- https://en.bitcoin.it/wiki/Scalability
- https://docs.google.com/spreadsheets/d/1oIaWzybcWLQ22eXTZlLrG7Lhn82EIrRvqi3S5v94l0k/edit#gid=0