Deep into ethereum light client protocol

Gary Rong
garyrong@ethereum.org
rjl493456442

Agenda

- Introduction
- Merkle Trie & Merkle Proof
- Checkpoint Synchronization
- What can light client do
- Flow control & Capacity management model

1. Introduction

What's light client protocol

- The protocol used by "light client"
  - Low resource requirement
  - Data verification capability
- Download and check the validity of the block headers
- Don't check the validity of state transition
- Fetch and proof the other part of blockchain on-demand
- Regard the whole DHT as database, local database as cache

Category of Light Client Protocol

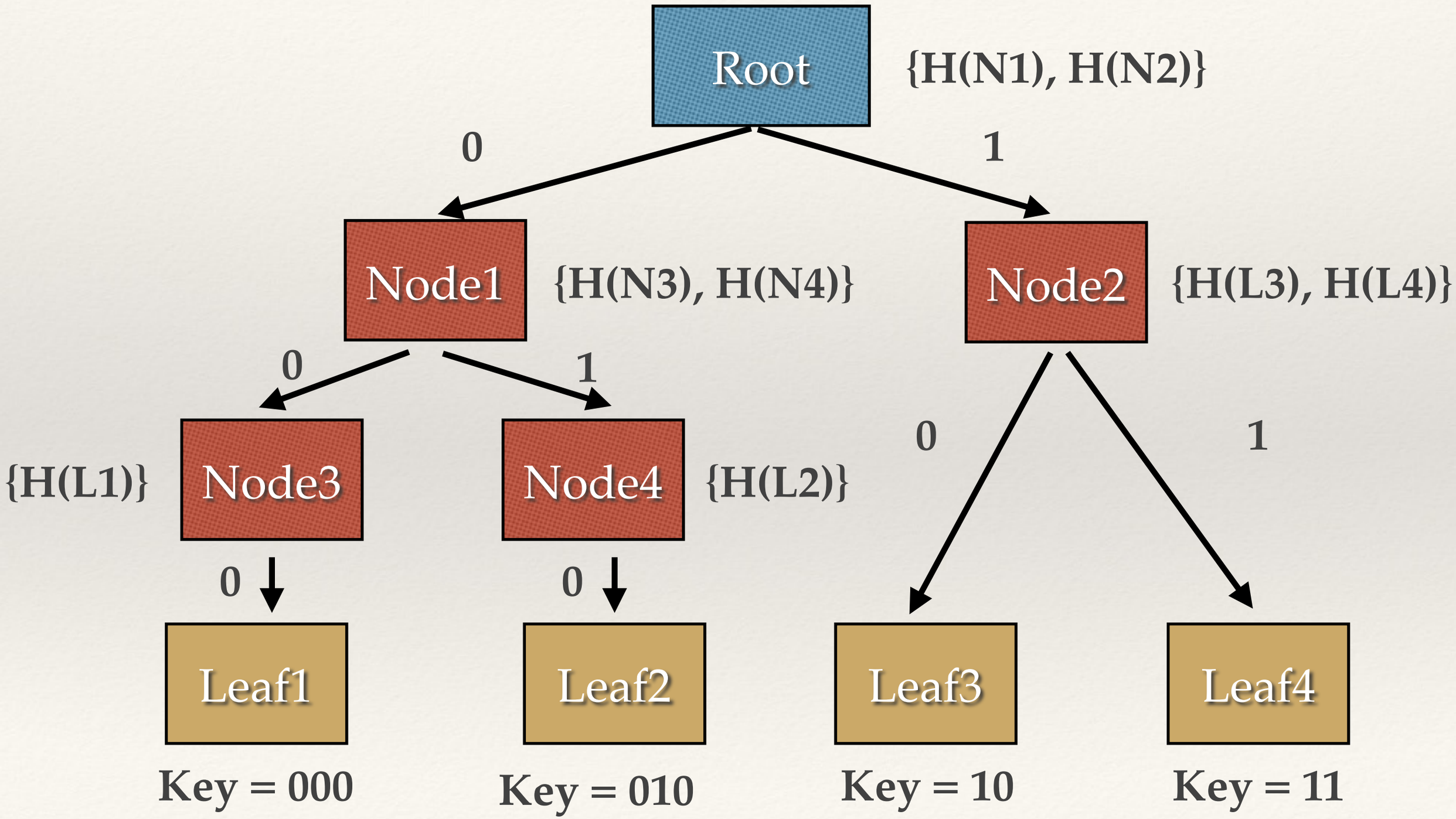
- Les(Light Ethereum SubProtocol)
- PIP(Parity Light Protocol)

Classification of nodes

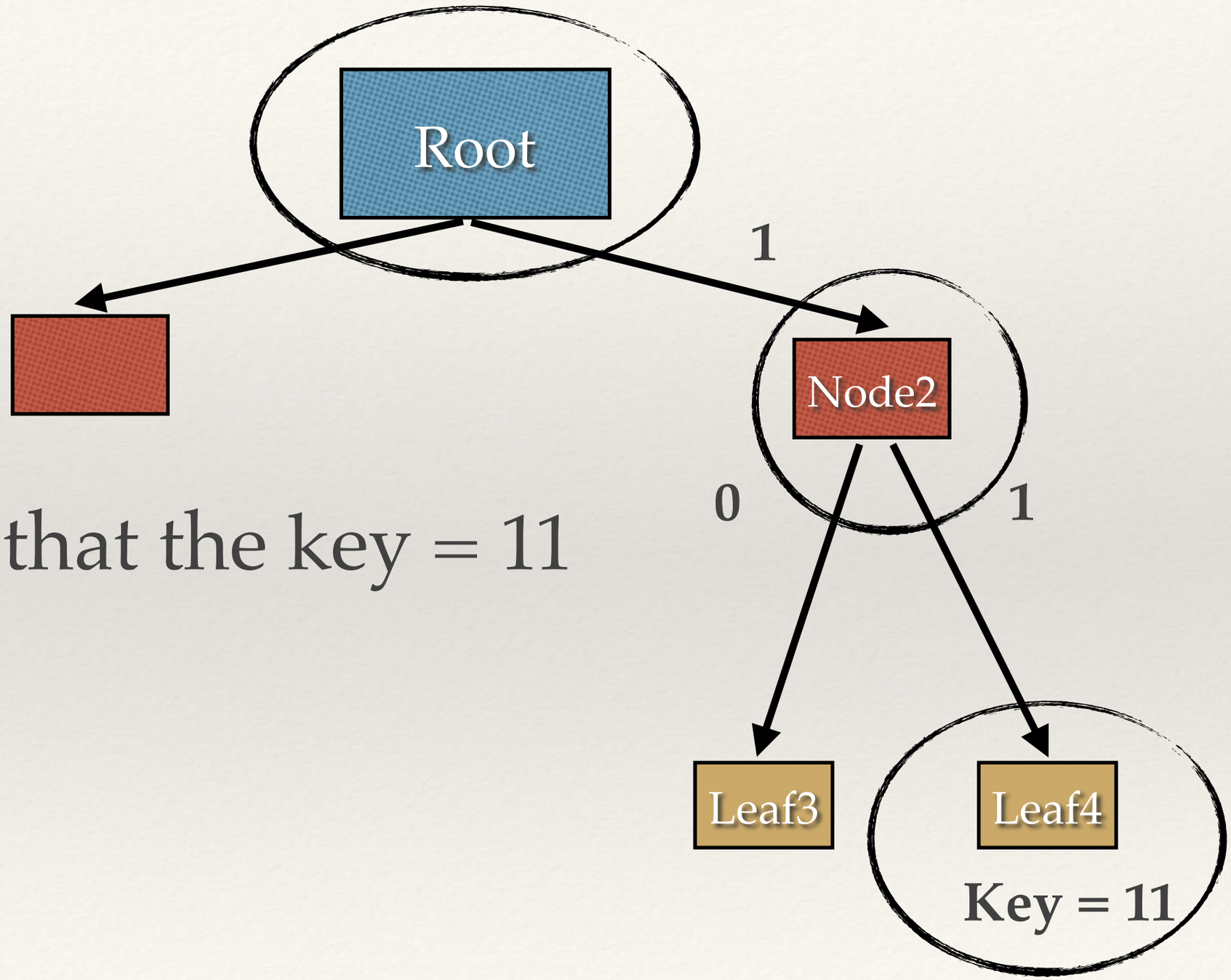
Type	Database size	Sync Time	Security Guarantee
Archive node	~2.3TB	~13days	High
Full node	~131GB	~4hours	Medium
Light client	~50MB	~1minute	Low

2. Merkle Trie & Merkle Proof

Merkle Trie


Root
{H(N1), H(N2)}
0
1
Node1
{H(N3), H(N4)}
Node2
{H(L3), H(L4)}
0
1
{H(L1)}
Node3
Node4
{H(L2)}
0
1
0
0
Leaf1
Leaf2
Leaf3
Leaf4
Key = 000
Key = 010
Key = 10
Key = 11

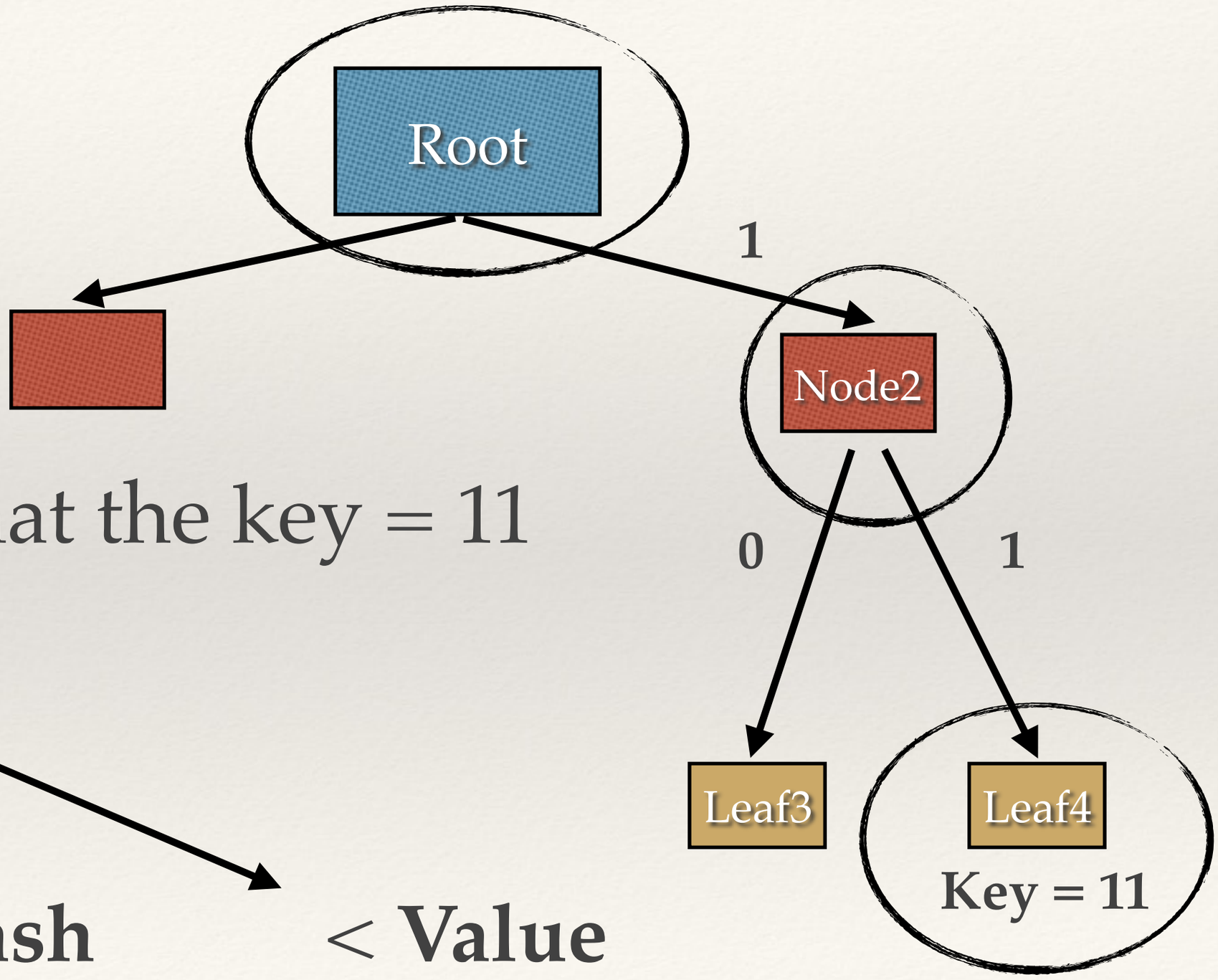
Merkle Proof (generation)



> Proof node that the key = 11

< [R, N2, L4]

Merkle Proof (verification)



> Verify node that the key = 11

> [R, N2, L4]

> **Local Root Hash** < **Value**

3. Checkpoint Synchronization

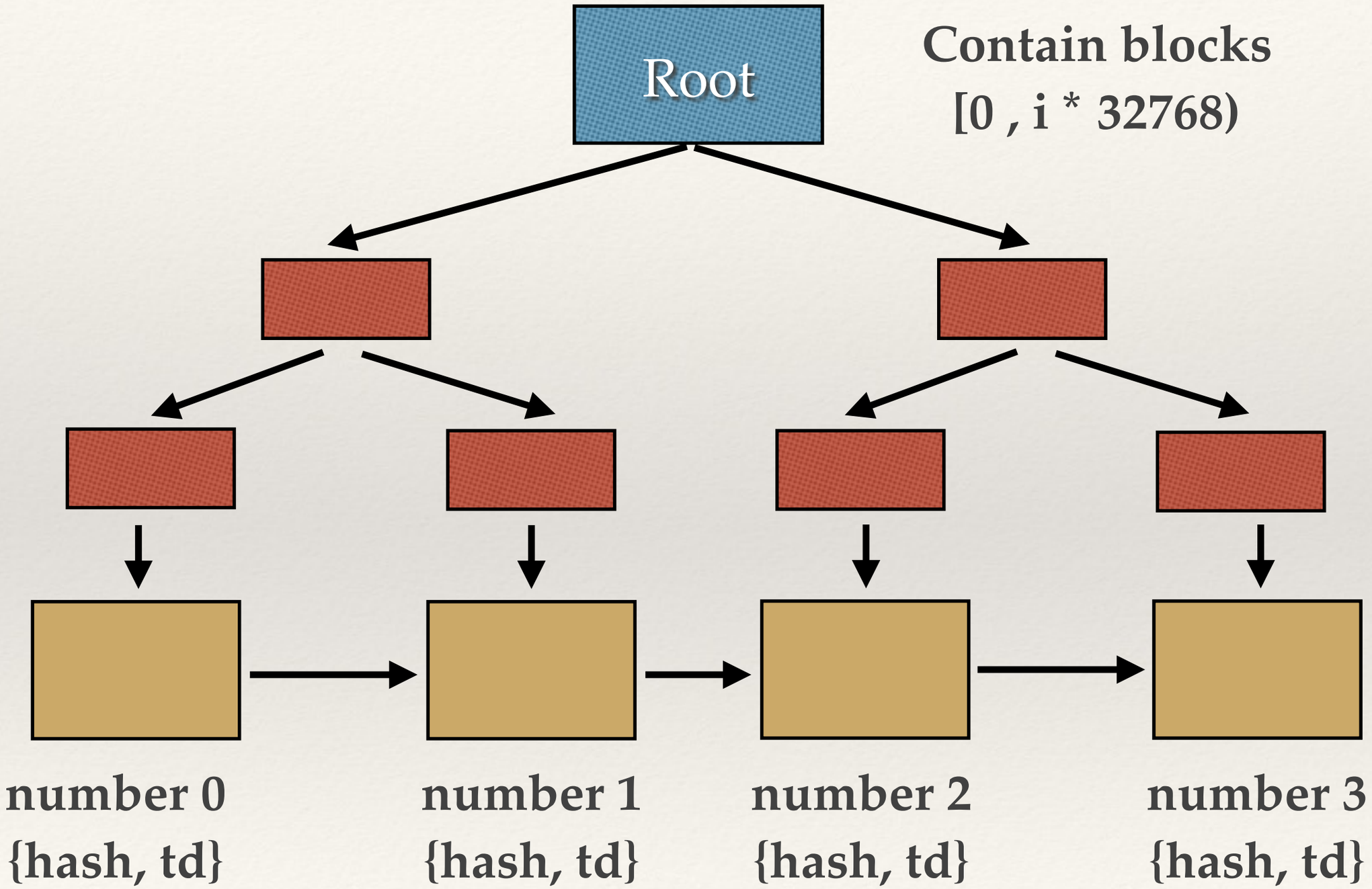
Checkpoint sync

- Start from a checkpoint
  - Trusted
    - Can verify the validity of historical chain
- Download block headers only
- One hundredth of the header is PoW verified(Network bandwidth >> Ethash bandwidth)

Checkpoint

- Generate a checkpoint every 32768 blocks(server or light client)
- Section index
- Cumulative canonical hash trie root
- Cumulative bloom trie root

Canonical Hash Trie

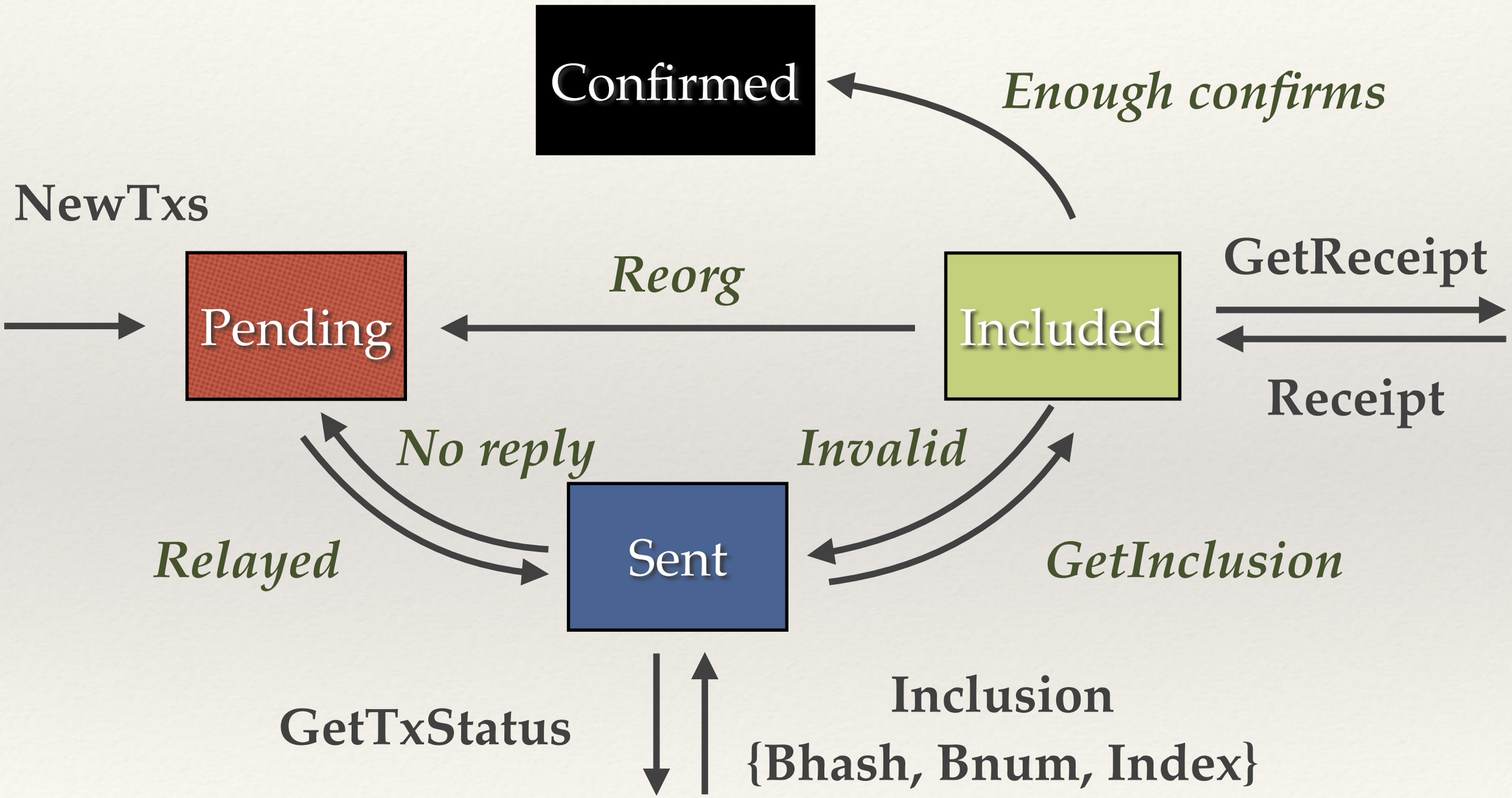


Checkpoint registration

Approach	Pros	Cons
Hardcode	Simple	1. Update according to release cycle 2. Old client can't use new checkpoint 3. Centralized
Checkpoint Oracle	1. Update according to admin registration 2. Old client can always use latest checkpoint	1. Complicated 2. Still centralized

4. What can light client do

Transaction relay



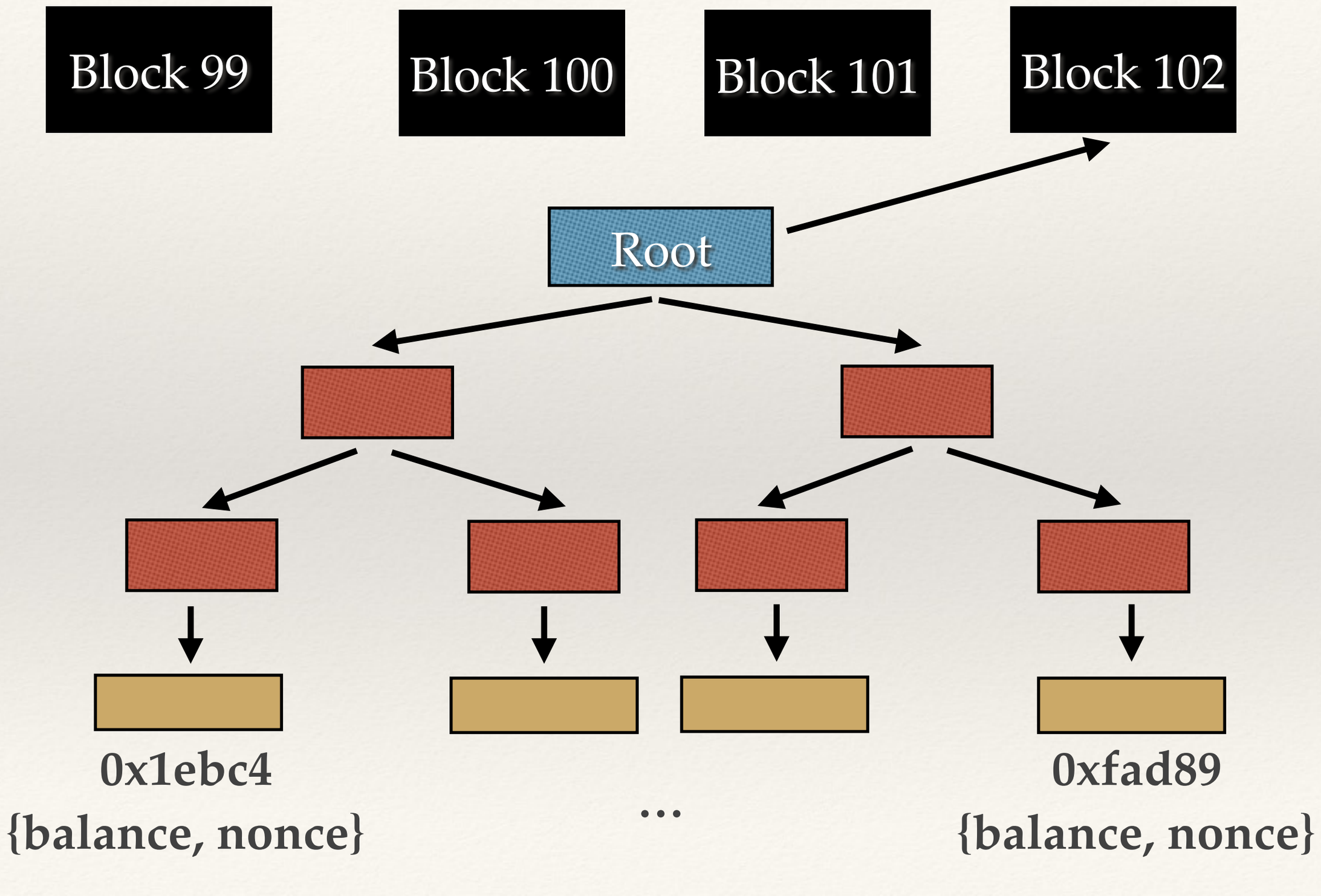
State query

Block 99

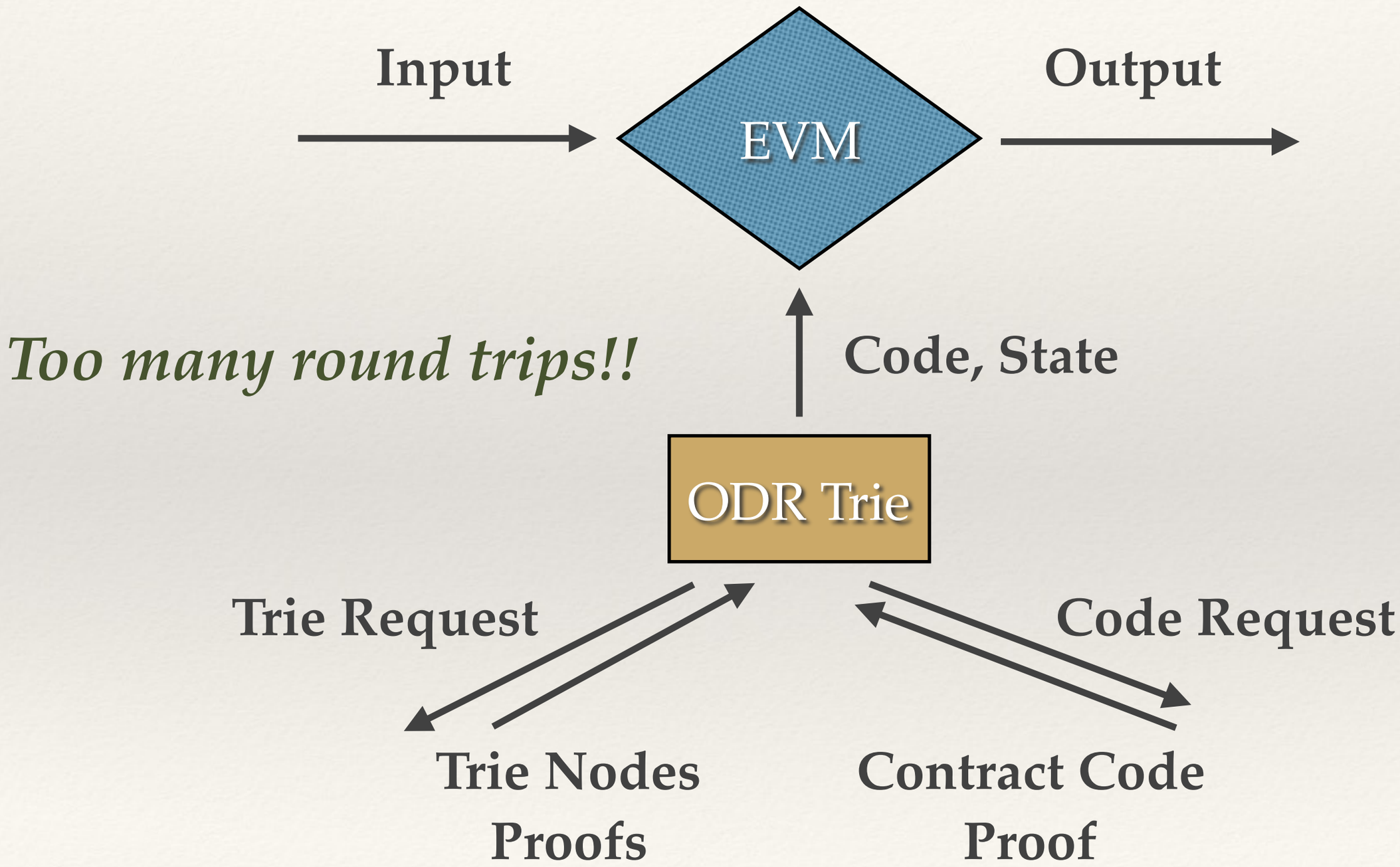
Block 100

Block 101

Block 102



Contract call



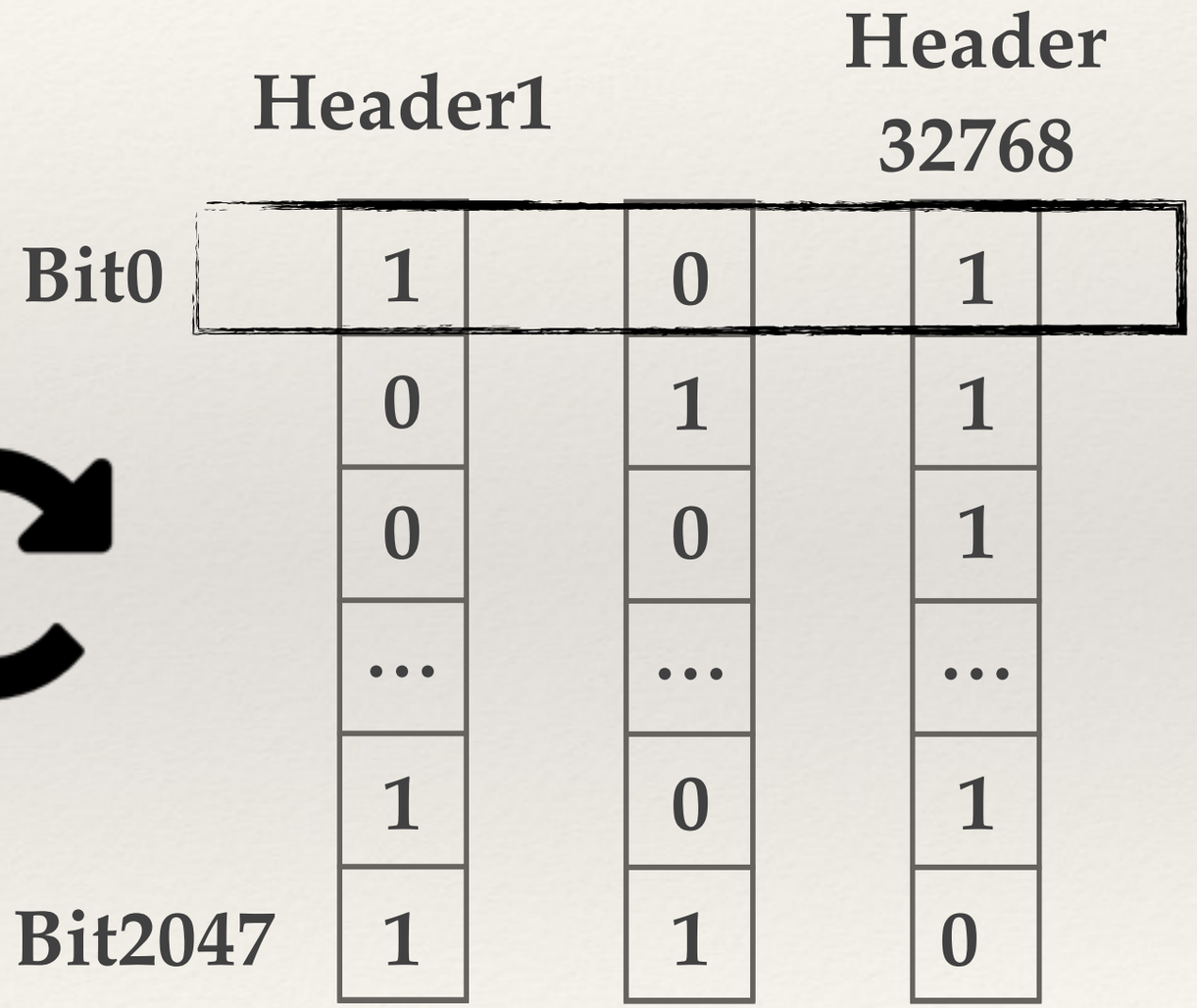
Event subscription

- Event logs are embedded in the receipts
- The bloom for logs is included in headers
- Match the header bloom first
- Download and verify the target receipts
- Filter logs

Event searching(history)

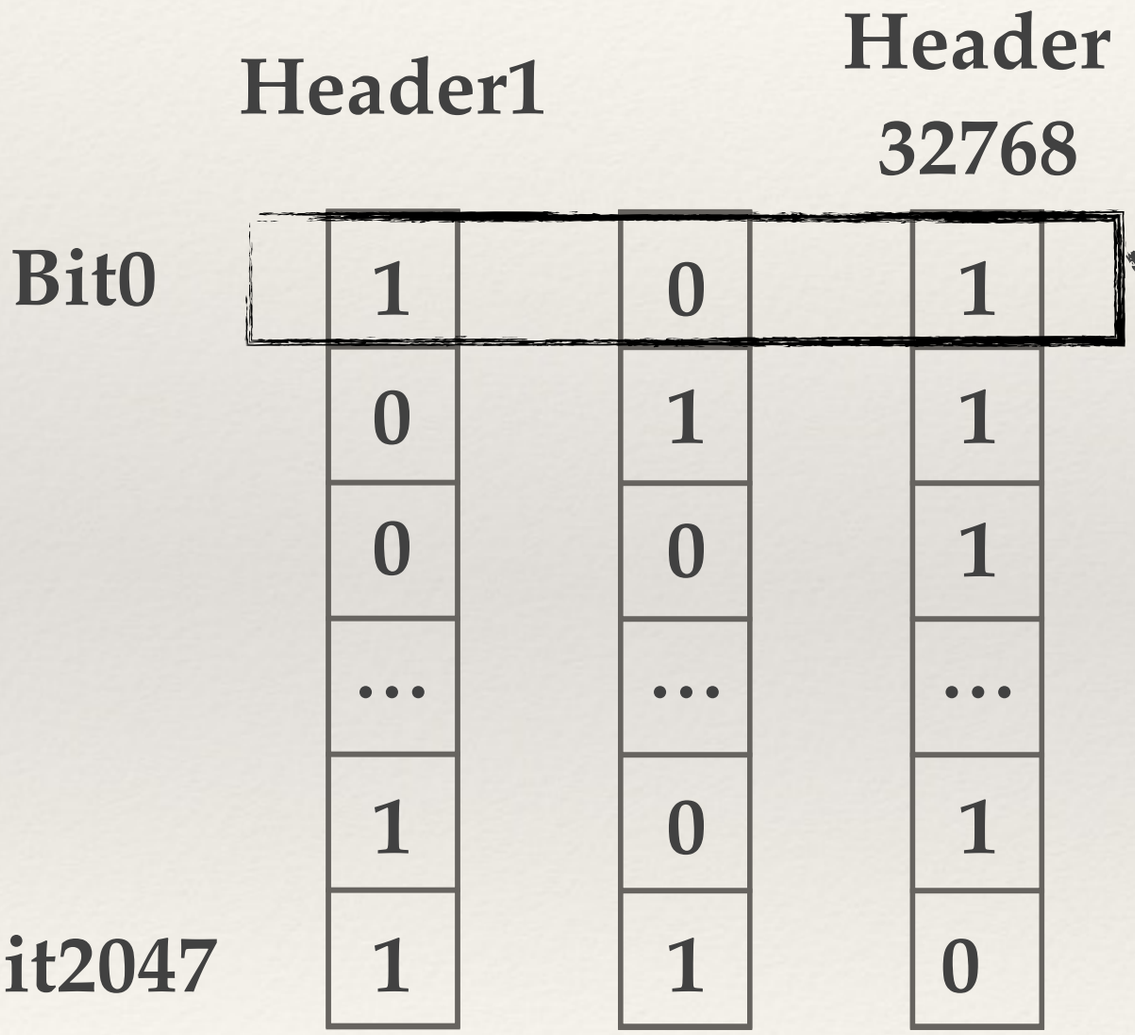
Bloom filter

Rotated Bloom filter

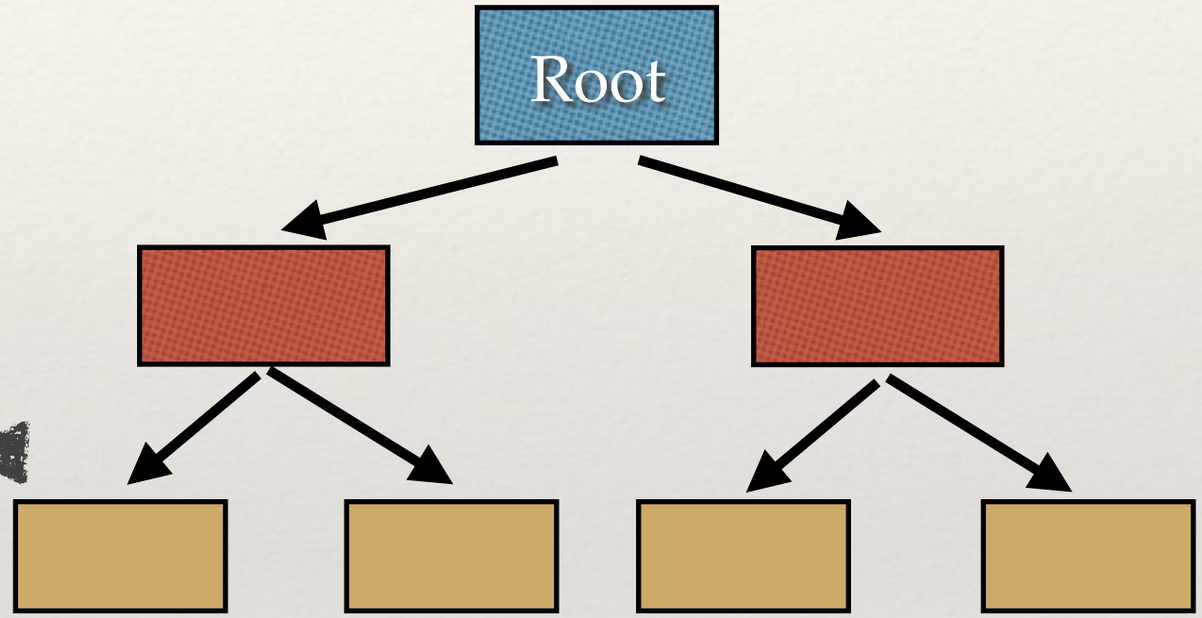


Event searching contd.

Bloom Section



Bloom Trie



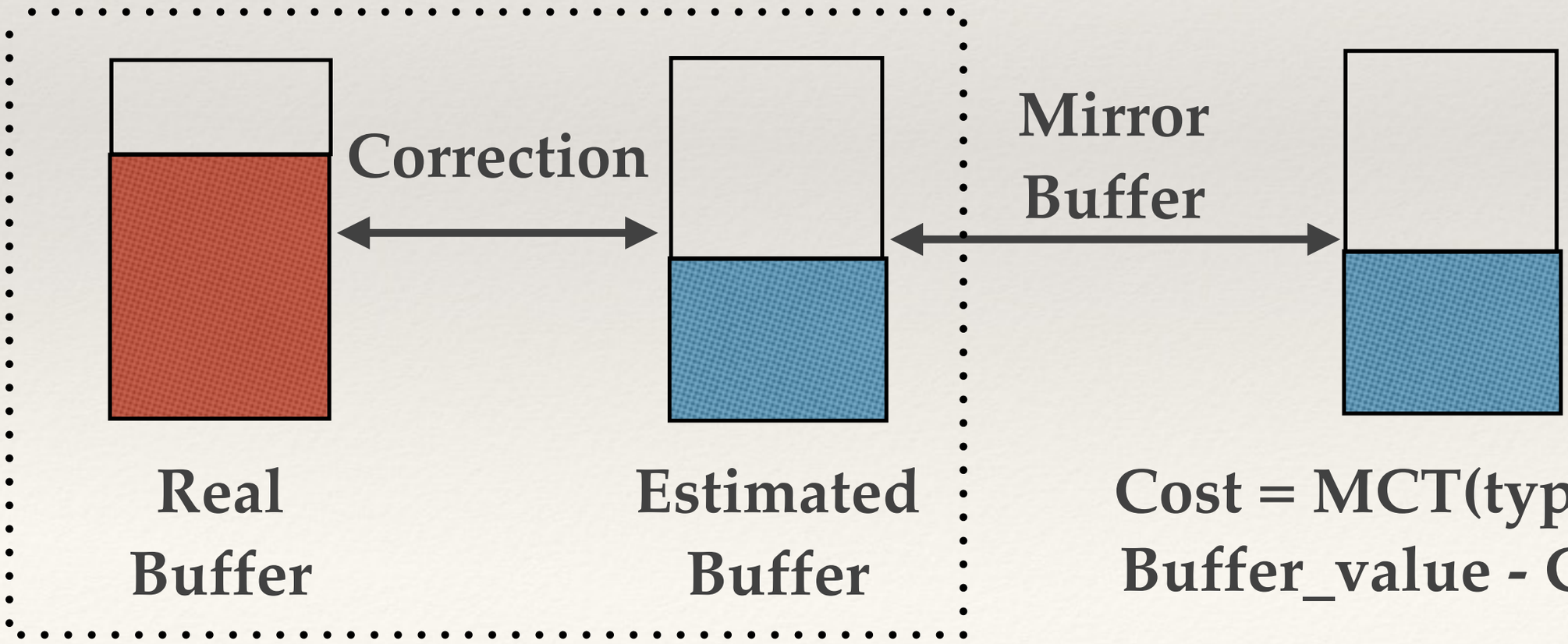
Key = BitIndex + SectionIndex

Value = Rotated bloom

5. Flow Control & Capacity management

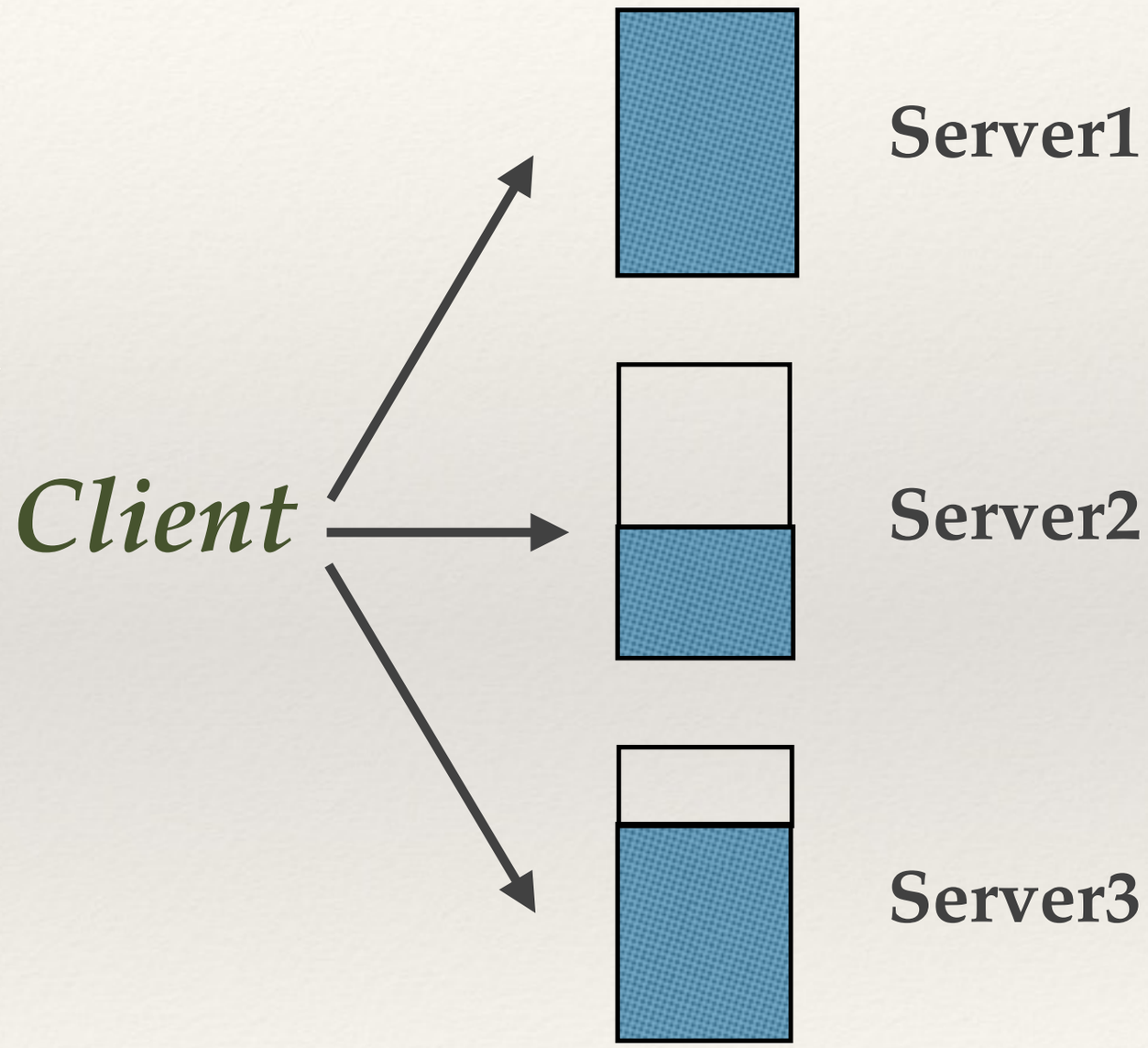
Flow control model

1. **BufferLimit**
2. **MinRecharge**
3. **MaxCostTable**



Flow control model contd.

- Local Load Balancer
- Higher buffer remaining, higher priority
- Avoid low process priority and high response latency



Capacity management

- —lightServ
  - Percentage of serving light client requests per second
  - Multi-threaded processing allows value over 100
- —lightbwout, —lightbwin
  - Network bandwidth limitation
- —lightpeers
  - Maximum number of light clients

Capacity management contd.

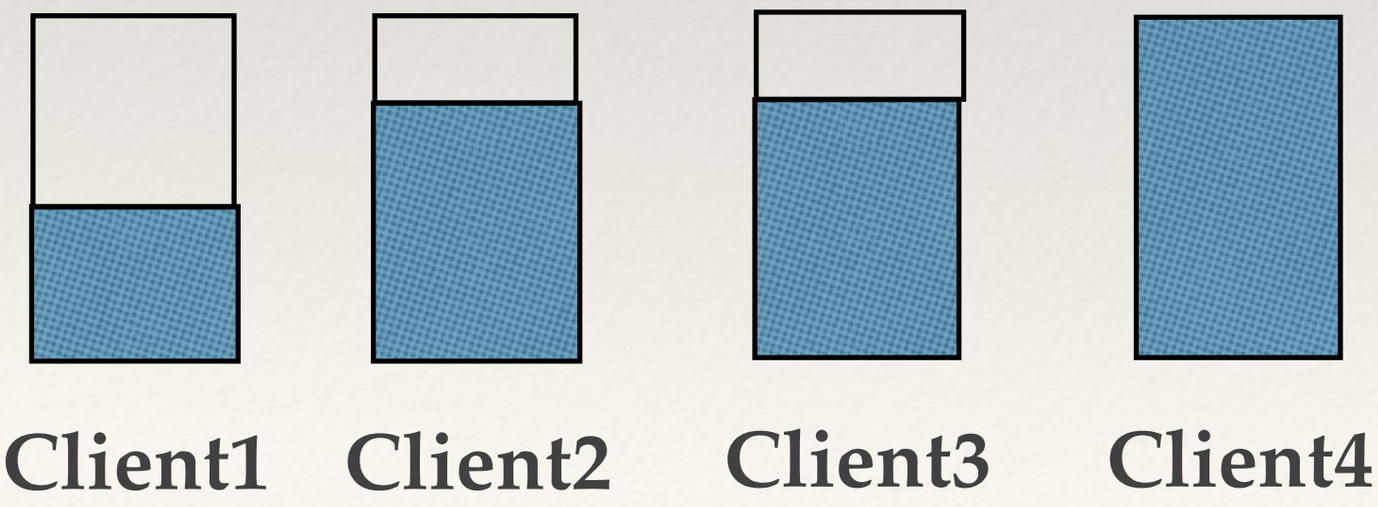
Cost calculation

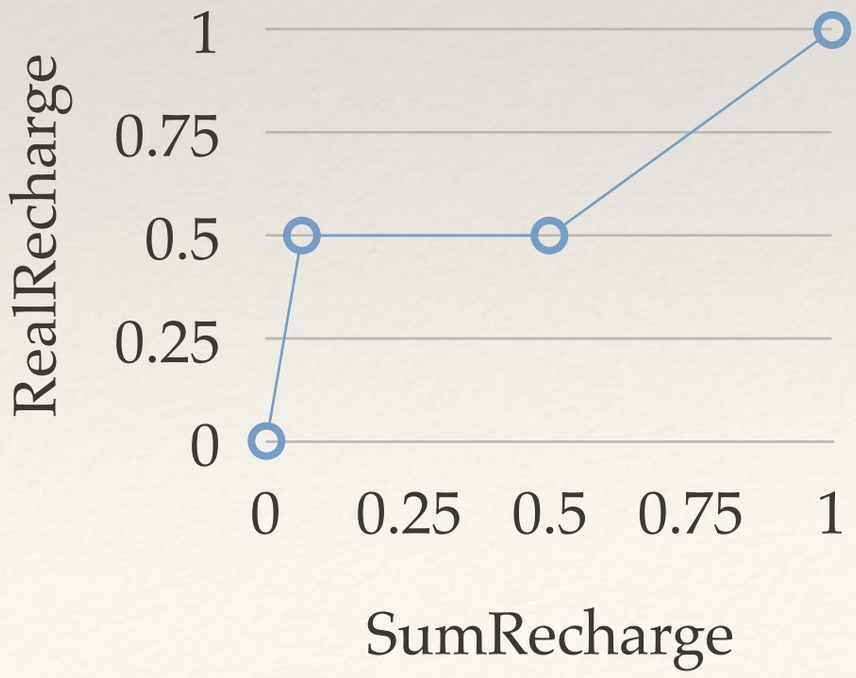
- TimeCost = ServingTime
- TrafficCost = PacketSize * TrafficFactor
- Cost = Max(TimeCost, InTrafficCost, OutTrafficCost)

Capacity management contd.

Buffer recharging

- TotalRecharge = LightServ * 1000,000,000 * Factor / 100
- Bonus = Curve(SumRecharge) / SumRecharge
- RealRecharge = MinRecharge * Bonus
- Enjoy recharge bonus if the server is idle





Capacity management contd.

Overbooking and Freezing

- The maximum capacity of system is determined by the maximum number of peers and the minimal capacity.
- The maximum capacity of system can be much higher than total recharge.
- Freeze and kick out some clients if the **recent serving time** plus the **queued estimated time** exceeds the limitation.

Capacity management contd.

Free client and Priority client

- Higher buffer limit and recharge speed
- Kick out free client first
- Economic incentive for full node

Thanks