BCon

Secure Systems Group, Aalto University

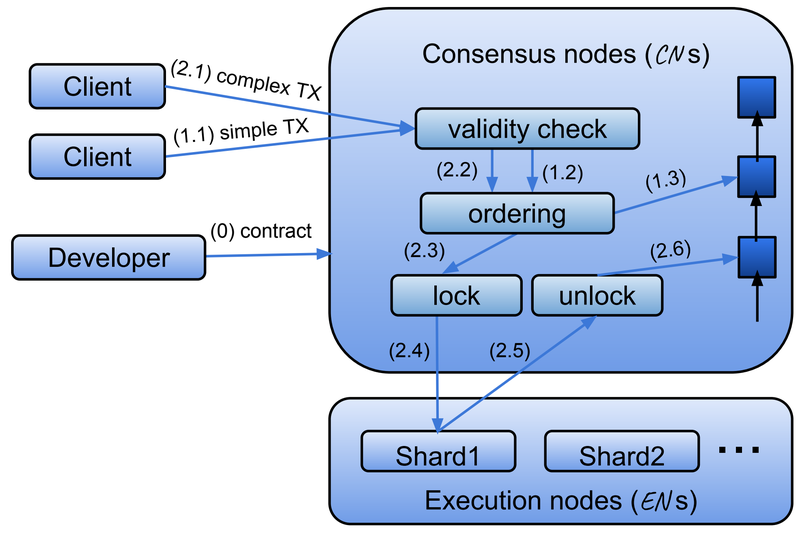
Jian Liu, Raymond Cheng, N. Asokan, Dawn Song

Robust and Efficient Sharding for Smart Contracts

- **Motivation:** Existing approaches to blockchain sharding require extensive coordination, and introduce livelocks for smart contracts.
- **Contribution:** a novel paradigm for sharding (or parallelizing) smart contract execution by separating execution from consensus; two ways of applying this paradigm to blockchains.

Saber: sharding by separating execution from consensus

- Consensus nodes (CNs):
  - Maintain and lock states
  - Check and order transactions
- Execution nodes (ENs):
  - Grouped into different "shards" (honest majority of each)
  - Execute the ordered transactions directly
- Advantages:
  - No intra-shard coordination or livelocks



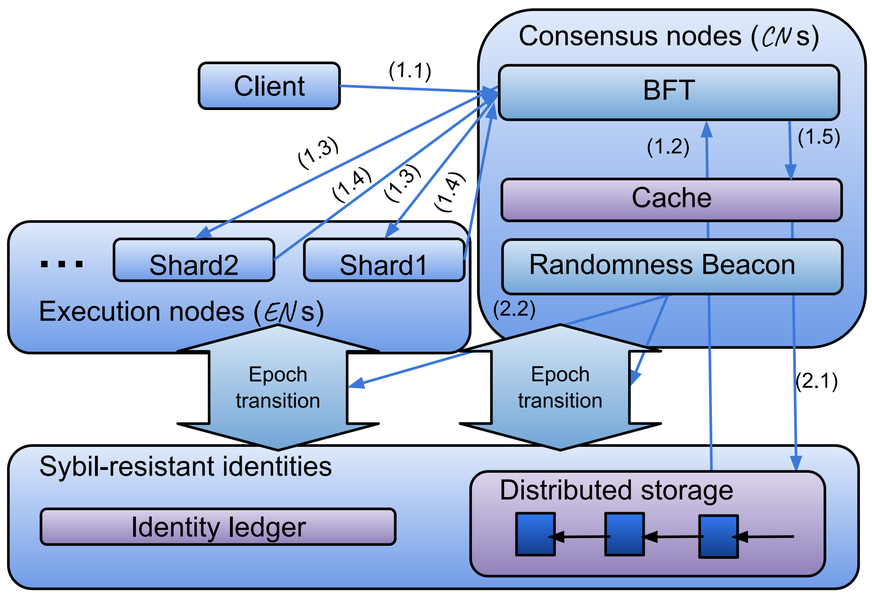
Execution sharding for Ethereum

- Separation only in logic
  - CNs: original Ethereum miners collectively
    - order transactions via PoW
    - designate shards for parallel execution
  - ENs: miners registering to the *ShardingManager* contract
    - nodes execute transactions off-chain
    - submit the results by making a new transaction
- Sharding management
  - Ethereum miners can *Register* by deposit some Ether
  - They are periodically assigned to different shards via Shuffle
    - *r* is an unbiased random number generated off-chain

```
1:  contract ShardingManager
2:      pk[] ENs
3:      pk[][] shards
4:      TX[][] tasks
5:      int sid                                  ▷ initialized as 0
6:      int m                                    ▷ number of shards
7:
8:      function Register()
9:          add caller's pk to ENs
10:     end function
11:
12:     function Shuffle(r)
13:         verify r
14:         empty shards
15:         for each EN in ENs
16:             i ← H(r, EN) mod m
17:             add EN to shards[i]
18:     end function
19:
20:     function Verify(i, M, σ̃)
21:         return Verify(shards[i], σ̃, M)
22:     end function
23: end contract
```

SaberLedger: public and permissionless blockchain

- Batch processing by grouping transactions into blocks
- Proof-of-stake (PoS) for Sybil resistant identities
- A new BFT protocol for the underlying consensus
- A randomness beacon for epoch transitions
- A distributed storage (e.g., IPFS) for state sharding



	Elastico	OmniLedger	Chainspace	Eris	SaberLedger
support blockchains	Yes	Yes	Yes	No	Yes
support cross-shard TXs	No	Yes	Yes	Yes	Yes
livelocks	-	Yes	Yes	No	No
intra-shard co-ordination	Yes	Yes	Yes	No	No
transparent to clients	Yes	No	Yes	No	Yes

Aalto University

Berkeley UNIVERSITY OF CALIFORNIA