Proof-of-Stake Sidechains

Peter Gaži, Aggelos Kiayias, Dionysis Zindros

IEEE Security & Privacy 2019
Imagine a **stake blockchain** where you want both the safety of Bitcoin and the features of Ethereum.

We start with one chain, the “Settlement layer”.

The SL is a safe, limited-feature blockchain.

We want to create a *network* of blockchains.
We introduce the “Computation Layer”, a different blockchain
CL will be a feature-rich smart contract chain
We need to move money between SL/CL

1. move 1 coin from SL to CL

2. move 1 coin around within CL enjoy smart contract functionality

3. move 1 coin from CL back to SL
We need to move money between SL/CL

- CL will begin with its own Genesis block when it’s ready
Two types of nodes

• Full nodes will come in two flavours:
  • “SL nodes”: Only monitor SL blockchain
  • “SCL nodes”: Monitor both SL and CL blockchains
Cross-chain transactions [out]

1. Money moves around in regular transactions in SL

2. A special transaction “destroys” money on the SL

3. A follow-up transaction “creates” new (corresponding) money on the CL
Cross-chain transactions [in]

1. Money moves around in regular transactions in CL
2. A special transaction “destroys” money on CL
3. A follow-up transaction “creates” new (corresponding) money on the SL
• SCL nodes can see outgoing transactions from SL
SCL nodes can see outgoing transactions from SL
The isolation problem

- SL nodes do not download CL blocks
- How can they learn about CL transactions?
- This is necessary so that SL can unlock the money in SL
Epoch synchronization

• We synchronize the epochs between SL / CL
The epoch committee

- **Basic idea:** Each epoch elects a small CL committee which represents the epoch.
- The committee is probabilistic and representative of the stake. It’s more probable you will be in the committee if you have large stake.
- **How to elect?**
  - Sample the last 2k slots of epoch
  - Those 2k slot leaders constitute the committee
- “Honest majority“ of stake translates to “honest majority” in the committee
- Committee is temporary -- changes once per epoch
Certificate-based cross-chain communication

- CL epoch committee *signs off* transactions destroying money in CL
- These signatures are submitted to the SL
- The signature is transmitted across chains once per epoch
How do the SL nodes verify incoming transactions?

- SL nodes know what the CL committee is for each epoch
- SL nodes know the CL committee at CL Genesis
- In addition to the transactions,
  - the old committee *signs off* the new committee at every epoch
- This passes control from the old committee to the new committee
The firewall property

- If the CL has a catastrophic failure, incoming money is limited to the outgoing amount
- The SL nodes keep count of how much money has left SL
- No more money can come back
- This ensures the *macroeconomic* properties of SL are maintained even if CL fails
References

- Aggelos Kiayias, Andrew Miller, Dionysis Zindros. *Non-Interactive Proofs of Proof-of-Work*, FC 2019
Thanks! Questions?

INPUT | OUTPUT

CARDANO

National and Kapodistrian University of Athens

Only some rights reserved
Fig. 1: Deployment options for PoS Sidechains.
Definition 8 (Pegging security). A system-of-ledgers protocol $\Pi$ for $\{L_i\}_{i \in [n]}$ is pegging-secure with liveness parameter $u \in \mathbb{N}$ with respect to:

- a set of assumptions $A_i$ for ledgers $\{L_i\}_{i \in [n]}$,
- a merge mapping $\text{merge}(\cdot)$,
- validity languages $\forall_A$ for each $A \in \bigcup_{i \in [n]} \text{Assets}(L_i)$,

if for all PPT adversaries, all slots $t$ and for $S_t \triangleq \{i : A_i[t] \text{ holds}\}$ we have that except with negligible probability in the security parameter:

**Ledger persistence:** For each $i \in S_t$, $L_i$ satisfies the persistence property.

**Ledger liveness:** For each $i \in S_t$, $L_i$ satisfies the liveness property parametrized by $u$.

**Firewall:** For all $A \in \bigcup_{i \in S_t} \text{Assets}(L_i)$,

$$\pi_A(\text{merge}(\{L_i^U[t] : i \in S_t\})) \in \pi_{S_t}(\forall_A).$$