SCP: A Computationally Scalable Byzantine Consensus Protocol for Blockchains

Loi Luu, Viswesh Narayanan, Kunal Baweja, Chaodong Zheng, Seth Gilbert, Prateek Saxena
National University of Singapore
Bitcoin doesn’t scale

• **Hard coded parameters**
  – 1 block per 10 minutes
  – 1 MB block size
  – 7 TXs per second

• **Today**
  – 1-2 TXs per second
  – VISA: 10,000 TXs per second
Our solution: SCP

• Scale up throughput **several orders of magnitude**
  – Without degrading any security guarantee

• Several blocks in each epoch
  – No. of blocks \( \approx \) network computation capacity

• Require minimum amount of network bandwidth
  – Broadcast only one block header
Byzantine consensus problem

• Problem
  – N nodes, f are malicious
  – Propose and agree on one value

• Byzantine consensus for blockchains
  – Set of valid TXs per epoch
Classical byzantine consensus protocol

- Intensive research
  - Can tolerate \( f < n/2 \)
- Assumption of known identity set
- Bandwidth limited
  - \( O(n^2) \) messages (e.g. PBFT)
  - Work for a small network (e.g. \( n < 1000 \))
Nakamoto consensus protocol

- Work for network of any size
  - Select leader by proof of work
- Linear message complexity
- Does not scale well in practice
  - One block per epoch
  - Bandwidth = \(O(\text{block size})\)
  - Reparameterization is not a long term solution
Reparameterization: reducing epoch time

• Setup
  • Using Amazon EC2
  • Run over 5 regions

• Results
  • TX rate increases until some threshold
  • Drops at 12 second epoch time

![Graph showing throughput vs epoch time]
Problem

- Secure & scalable consensus protocol
  - Compete with VISA?
SCP overview

• **Adjust throughput based on network mining power**
  – Split the network into several committees
  – Committees propose blocks in *parallel*
  – No. of committees \( \approx F(\text{network mining capacity}) \)

• **Data needed for reaching consensus is minimal**
  – Consensus data \( \neq \) transactional data
  – Verify block without block data
  – Selectively download block data
SCP protocol

1. Consensus Blk
2. Blk Header
3. Data Blk
4. Data Blk
5. Consensus Blk

1010101101
Step 1: Identity establishment

- **Solve PoW**
  - SHA2(EpochRandomness || IP || pubkey || nonce) < D
### Step 2: Assigning committees

- Randomly & uniformly distribute identities to committees
  - Based on the last k bits of PoW

<table>
<thead>
<tr>
<th>ID</th>
<th>PoW</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>00001...00</td>
</tr>
<tr>
<td>2</td>
<td>00000...01</td>
</tr>
<tr>
<td>3</td>
<td>00000...10</td>
</tr>
<tr>
<td>4</td>
<td>00001...11</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Diagram:
- PoW: 00001...00
- PoW: 00000...10
- PoW: ...
- PoW: 00000...11
- PoW: 00001...01
- PoW: ...
- PoW: 00001...11
- PoW: 00000...10
- PoW: ...
- PoW: 00001...00
- PoW: 00000...01
Size of a committee C

• **Decide the probability of majority honest**
  – $P(\text{error})$ reduces exponentially with $C$
    • $f = N/3$, $C = 400$, $p(\text{error}) \approx 10^{-12}$
    • $f = N/3$, $C = 100$, $p(\text{error}) \approx 0.0004$

• **Why majority honest within a committee?**
  – Run practical authenticated BFT
  – Allow others to verify committee's block without block's data
    • At least 1 member is honest in any $(C/2 + 1)$ members
Step 3: Propose a block within a committee

- Run a classical Byzantine consensus protocol
  - Members agree & sign on one valid data block
  - No. of messages $\approx O(C^2)$
- TX sets included in data blocks are disjoint
  - Include TXs with a specific prefix

<table>
<thead>
<tr>
<th>Block</th>
<th>TX’s IDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Block 1</td>
<td>00...</td>
</tr>
<tr>
<td>Data Block 2</td>
<td>01...</td>
</tr>
<tr>
<td>Data Block 3</td>
<td>10...</td>
</tr>
<tr>
<td>Data Block 4</td>
<td>11...</td>
</tr>
</tbody>
</table>
Step 4: Final committee unions all results

Propose a consensus block

Header of Data Block 1
- 00001...001
- 00000...101
- ...

Header of Data Block 2
- 00000...10
- 00001...10
- ...

Header of Data Block 3
- 00000...11
- 00001...11
- ...

Step 5: Generate an epoch randomness

• **Goal**
  – Generate a fresh randomness
  – Adversary cannot control or predict

• **Common approach: Use consensus block hash**
  – Problem: adversary can predict the consensus block early

• **Our approach: Users can have different randomness**

Commit $R_i$ in SHA2($R_i$) when join the committee

Agree on the consensus block

Broadcast the block header and $R_i$

Use any $c/2 \cdot R_i$ as the epoch randomness
Implement a SCP-based cryptocurrency

• **Challenges**
  – How to form committees efficiently
    • Too many new identities in each epoch
    • Epoch time may be long to prevent conflict
  – Double spending transactions
    • Without previous block data?
Forming committees efficiently

• **Approach:** Reuse identities from previous epoch
  – Elect one new member and remove the oldest one
  – Number of new identities $\approx$ number of committees
Avoid double spending

• **Approach:**
  – Split double spending check into both miners and users (recipients)

Checked by committee members

Double spending

Within a block

Across blocks

Checked by recipients*

*: Proof-of-publication
Checking double spending across blocks

- **Merkle tree of TX inputs**
  - An input is spent in a block
    - Proof of size $\log(N)$
  - An input is *not* spent in a block
    - Proof of size $2\log(N)$

Prove that 5 is not included?
Checking double spending across blocks (2)

• **Sender** proves that the TX’s input is not spent elsewhere
  – The proof of size $L \times \log(N)$
  – Can be optimized

• **Recipient** checks by using only consensus block headers
  – Actively support SPV clients without a trusted third party
  – Support 1-confirmation TXs
Conclusion

• SCP scales almost linearly with network mining capacity
  – More mining power, higher transaction rate
  – Reduced network bandwidth
  – Secure

• Applicable to several applications
  – Cryptocurrency, decentralized database, etc
Q&A

Loi Luu

loiluu@comp.nus.edu.sg

www.comp.nus.edu.sg/~loiluu
Future work

• Incentive structure
  – Incentivize committee members and other parties

• Prevent DoS attack by sending invalid TXs
  – Users can send arbitrary TXs to the blockchain now

• Rollback solution
  – $P(\text{error}) \neq 0$
Related work

• **Bitcoin-NG & Ghost**
  - ✓ Allow more blocks
  - x Does not separate consensus plane and data plane

• **Lighting network**
  - ✓ Allows more micro transactions
  - x Does not solve scalability problem

• **Sidechains**
  - ✓ Good for experimenting new blockchains
  - x Does not make Bitcoin scalable
Adjusts number of committees frequently

- Similar to how Bitcoin adjusts the block difficulty
  - $T$: the expected epoch time
  - $T'$: the averaged epoch time of the most 1000 recent blocks
  - $S$: Current number of committees
  - $S'$: adjusted number of committees

$$S'\log(S') = S\log(S) \frac{T}{T'}$$
<table>
<thead>
<tr>
<th>Consensus Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous Block Hash</td>
</tr>
<tr>
<td>Committee signatures</td>
</tr>
</tbody>
</table>

*Data block commitments*

<table>
<thead>
<tr>
<th>No.</th>
<th>Data Block’s hash</th>
<th>Merkle root of TXs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0x123abc...</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>0x123456...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Block 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous Consensus Blk</td>
</tr>
<tr>
<td>Block hash</td>
</tr>
<tr>
<td>Committee signatures</td>
</tr>
</tbody>
</table>

*Included TXs*

<table>
<thead>
<tr>
<th>Data Block 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous Consensus Blk</td>
</tr>
<tr>
<td>Block hash</td>
</tr>
<tr>
<td>Committee signatures</td>
</tr>
</tbody>
</table>

*Included TXs*
SCP properties

• Number of data blocks \( \approx \) network mining power
  – Frequent adjustment of no. of blocks

• Data broadcast to the network is minimal
  – Broadcast data is independent of block size

• Secure against adaptive adversary w.h.p.
  – Can reparameterize \( c \) to secure against stronger adversary