Distributed Trust and Blockchains		Date:	*07-11-2019*
Instructor: *Sujit Prakash Gujar*		Scribes:	*Varun Gupta , Rajat Kumar Verma*

Lecture 16: Secure High-Rate Transaction Processing in Bitcoin and Proof of Stake Blockchains

1 Secure High-Rate Transaction Processing in Bitcoin(Yonatan Sompolinsky and Aviv Zohar)

1.1 Bitcoins

- Decentralised Cryptocurrency
- Bitcoin was created in 2009 by an unknown developer (or developers) under the pseudonym Satoshi Nakamoto.
- 1 Bitcoin = Rs. 6,37,764.03
- 2.9 million - 5 million users

1.2 Bitcoins - Obstacles

- Adoption
  - There are many retailers, particularly online, through which consumers can pay for transactions in bitcoin, but the digital currency still isn't anywhere close to being widely accepted. If big payment processing company, decides to allow retailers that use their hardware to start accepting bitcoin payments easily, it could be a game-changer for the mainstream adoption of bitcoin.
- Volatility
  - Bitcoin has been incredibly volatile since its inception.
- Regulation
- Scalability
  - As more and more bitcoin trades and purchase transactions are executed, the network will have a more difficult time keeping up, which could result in serious processing delays.

1.3 Bitcoins - Scalability

- To improve the throughput of Bitcoin via enlarging its block size or reducing block interval without changing Bitcoin's POW consensus algorithm.
- Disadvantages of this:-
  - Security issues
  - Decentralization issues - Adversaries with less than 50% power can alter the main chain.

1.4 Bitcoin Consensus

- This algorithm is simplified in that it assumes the ability to select a random node in a manner that is not vulnerable to Sybil attacks.
- Alogorithm:-
  - New transactions are broadcast to all nodes.
  - Each node collects new transactions into a block.
  - In each round, a random node gets to broadcast its block.
  - Other nodes accept the block only if all transactions in it are valid (unspent, valid signatures).
  - Nodes express their acceptance of the block by including its hash in the next block they create.

1.5 Ghost Protocol

Key Idea : a new policy for the selection of the main chain

Algorithm 1. *Greedy Heaviest-Observed Sub-Tree (GHOST)*
Input: Block tree T

1. *set* $B \leftarrow$ *Genesis Block*
2. *if* $Children_T(B) = \emptyset$ *then return(B) and exit*
3. *else update* $B \leftarrow \underset{C \in Children_T(B)}{\operatorname{argmax}} |subtree_T(C)|^8$
4. *goto line 2*

Figure 1: Image Credit : Paper

- **Ghost Protocol Representation**
- List the hashes of all childless blocks inside each block

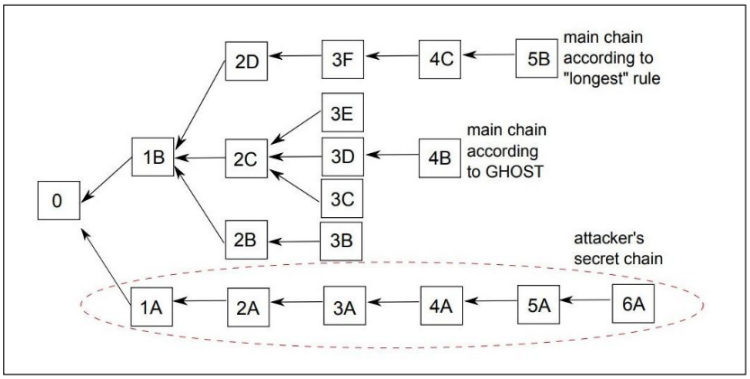

Figure 2: Image Credit : Paper

- **Ghost Protocol - Properties**
  - **The Covergence of History**
    - **Proposition :** Every block is eventually either fully abandoned or fully adopted.
    - **Proof :**
      1. Let t be the time at which B is neither adopted nor abandoned
      2. t event in which B' is created at (t + D, t + 2D); next block after t + 3D
      3. After t , B is either adopted or abandoned by all nodes
      4. During (t, t + D) each node is extending equal weighted subtrees
      5. Then B' is created which breaks all ties, propagated across in D time
  - **Resilience to 50% Attacks**
    - **Proposition :** The probability that B will be off the main chain, goes to zero as the time after its adoption goes to infinity.
    - **Proof :** Let time(B) be the time in which the B is created, B the time in which it is either adopted or abandoned with time(B)+T be the time at which B is off the main chain. Then the probability that by time time(B) + T, B was either already abandoned or already adopted, is P which goes to 1 as T goes to infinity. It follows from Markov's inequality and E[B] $<\infty$.
- **Ghost Implementation**
  - GHOST requires knowledge of off-chain blocks by all nodes.
    - Propagate only headers to all nodes
    - List the hashes of all childless blocks inside each block
  - For deployment, as the two rules only differ at higher throughput.
    - Adoption of GHOST can be gradual at low transaction rates
    - That is, nodes will be compatible with Longest-Chain here
  - Towards difficulty adjustment of the proof-of-work
    - The total rate of block creation (), be kept constant
    - Instead of rate of growth of of the longest chain,
    - Since the relation between and the difficulty is complex
  - Reward minted coins only to those whose blocks are on the main chain
  - Have checkpoints (similar to Bitcoin) so that attackers cannot create blocks off-chain.

2 Proof of Stake Blockchains

2.1 Basics of Blockchain

- A blockchain is a chain of blocks rendered immutable by hash links.
- A decentralized blockchain is maintained by a network of nodes.
- Due to delay and Byzantine behaviour, we require blockchain protocols to ensure consensus amongst nodes.

2.2 Blockchain Protocol

- To ensure consensus, we require that all nodes agree on a singular chain.
- Since several nodes would like to claim the rewards for publishing blocks, if we allow multiple nodes to publish blocks at any given point in time, the blockchain forks.
- Thus, we need a block publisher selection mechanism (BPSM), to ensure that only a subset of blocks are published at any given time (ideally only one).

- To deal with forks, if they arise we require a chain selection rule (CSR).
- Protocol:-
  - BPSM: Proof of Work
  - CSR: Longest chain rule
- Challenges:-
  - Extremely high power consumption
  - Low performance

2.3 Blockchain Performance

- Performance metrics:-
  - Transactions per second (tps)
  - Time to finality (ttf) - the time required to confirm a published transaction
- Typically in Bitcoin, tps = 4 and ttf= 60 minutes
  - Due to the poor tps, it is unable to scale to high volumes
  - High ttf deters usability and adoption

2.4 Attempts to address these challenges

- By changing CSR
  - Methods such as GHOST, which uses "Greedy Heaviest Observed SubTree", instead of longset chain
  - Such methods improve performance but do not address the high power consumption which is inhenrent in PoW protocols
- By changing BPSM
  - Methods such as Proof of Stake use the stake distribution to stochastically determine who should publish next
  - Such methids are highly power efficient, but the protcols are very challeging design owing to adversarial behaviour such as "nothing at stake" attacks

2.5 Tokenization

- Bitcoin uses hashes performed by the nodes as lottery tokens, to limit node participation
- PoS methods must also limit user participation. Failing this, an adversarial node can publish as many blocks as it likes to create any number of forks of any length, thus compromising the protocol ("nothing at stake" attack)
- Typically, PoS methods achieve this by segmenting the timeline, and restricting participation in each period of time

2.6 The Ouroboros Protocol

- The Ouroboros Protocol (OP), divides the timeline into epochs and then further into slots
- At the start of each epoch is the pseudo-genesis block which enlists the stake distribution and the seed randomness to used for this epoch.
- The stake distribution is required to enable the stake weighted probablity of publishing.
- The seed randomness is the result of a multiparty computation, which is used to determine node participation

2.7 Assumptions and compromises

- Honest majority
- Need a good upper bound on block propagation delay
- Reliance on network time protocol
- Synchronous w.r.t. to time slots
- Multiparty computation overhead for the seed randomness

3 References

- Secure High-Rate Transaction Processing in Bitcoin by YonatanSompolinsky and Aviv Zohar
- https://blockgeeks.com/guides/proof-of-work-vs-proof-of-stake/