Identity problem in the sharding blockchain
(Feat. Attacking Zilliqa)

Geunwoo Kim
Intro

- Who am I?
- Why you here?
- What I present?
Who am I?

- My name is Geunwoo Kim
- Born in Gyeong-ju
- Man
- Enjoy playing soccer but cannot play anymore
- Get military Exemption
- Winner(?)
- Food enjoyer
- ...
- POSTECH CSE 16
- PLUS 16 captain
- Student returning to school after taking time off
- Kodebox software engineer (working from home)
- The man who wants to get good grades...(help me)
Why you here?

For eating some snack?

Just chilling out?

Enjoy waste of time?

Fan of me? (Great!)
Why you here?

For your convenience,

1. Get a brief sketch of the blockchain and sharding concept
2. What kind of problem which derives this attack vector
3. Overall insight about the blockchain security research
What I present?

One of my research topics I did at NUS computing

1. How to measure Bitcoin topology
2. Bitcoin network layer vulnerability analysis
3. Sharding blockchain network layer vulnerability analysis
What I present?

I got a reward from the bug bounty program! Honey!
Index

1. Introduce Blockchain
2. Introduce sharding concept
3. The road to the sharding
4. Attacking recent model
5. The problem
- Blockchain
- Sharding
- The road to the sharding
Blockchain

1. A wants to send money to B
2. The transaction is represented online as a block
3. The block is broadcasted to every party in the network
4. The network approves the transaction
5. The block is added to the existing blockchain in a transparent and unalterable way
6. The transaction is complete
Why sharding?

Two metrics directly related to blockchain scalability

1. **transaction throughput** (the maximum rate at which the blockchain can process transactions)
2. **latency** (time to confirm that a transaction has been included in the blockchain)

Reparametrization of Bitcoin’s block size and inter-block interval can improve performance to a limited extent

Significant improvement in performance requires fundamental redesign of the blockchain paradigm.
The road to the sharding

Blockchain has two key functional components
1. Transaction validation (execution)
2. Extending the blockchain: only leader can append block
The road to the sharding

Initial model

Figure 3: The Bitcoin blockchain model
The road to the sharding

First approach: multiple blocks per leader
- Bitcoin NG (13th USENIX NSDI)
The road to the sharding

- Bitcoin NG decouples leader election from transaction serialization.
- Bitcoin-NG divides time into epochs, and a leader can unilaterally append multiple transactions to the blockchain for the duration of its epoch.
- There are two kinds of blocks in Bitcoin-NG: keyblocks and microblocks. Keyblocks contain a solution to the puzzle and are used for leader election. Keyblocks contain a public key that is used to sign subsequent microblocks generated by the leader.
The road to the sharding

- Probabilistic leader election only per key block
- probabilistic consistency => strong consistency per micro block
- Latency will be going down
- Problem
  - Malicious leader
The road to the sharding

- Second approach: collective leader
  Byzcoin(25th USENIX security)
The road to the sharding

- ByzCoin modifies how Bitcoin-NG generates keyblocks: a group of leaders, rather than a single leader, generates a keyblock followed by microblocks.
- The leader group is dynamically formed by a window of recent miners. When a new miner solves the puzzle, it becomes a member of the current leader group, which moves one step forward, ejecting the oldest miner.
This design addresses a limitation of Bitcoin-NG

1. A malicious leader can create microblock forks: in ByzCoin this would require a two-thirds majority of leader group members to be malicious.
2. Bitcoin-NG suffers from a race condition where an old leader who has not yet heard about the new leader may continue to incorrectly mine on top of older microblocks. In ByzCoin, leader group members ensure that a new leader builds on top of the most recent microblock.
The road to the sharding

- **Third approach: parallel blockchain extension**
  In this approach multiple leaders extend in parallel different parts of the blockchain.
  Finally, we got a linear proportionality between computing power and throughput, but not actually feasible.
The road to the sharding

- Fourth approach: sharding transaction
  Elastico (2016 ACM SIGSAC)
The road to the sharding

Elastico partitions nodes into groups called committees
1. Within a committee, nodes run a Byzantine consensus protocol to agree on a block of transactions.
2. The final committee collates sets of transactions received from committees into a final block, runs a Byzantine consensus protocol between its members to get agreement on extending the blockchain
Current designs for the sharding

- Elastico
- Chainspace
- Omniledger
- Rapidchain
Conclusion

1. Recent design of sharding is mostly based on ByzCoin where multiple leaders append multiple blocks.

2. A form of the final committee like DS(Directory Service) committee who actually extend a block is a part of mainstream of the sharding design.
● What is Zilliqa
● Identity generation in Zilliqa
● Attack vector

Problem
What is Zilliqa

Zilliqa is a representative sharding blockchain implementation in current industry which is mostly based on Elastico and Omniledger
Settings

- An identity is defined by a pair of pubkey and (IP:PORT) (but blacklisted by only IP)
  => each node need its own public IP
- Account-balance based model
Two blockchain

- There are two kinds of blockchain; DS blockchain, TX blockchain
- The basic idea behind this is to split the roles of bitcoin block into two separate blocks
  - 1. Election of a leader who decides the next block
  - 2. Verification of transactions

1. DS blockchain (= Identity blockchain) : chain of DS-Block
2. TX blockchain : chain of TX-Block

What are the DS-Block and TX-Block?
Epoch

- Every 100 TX-Block, one DS-Block is mined by previous DS committee
  - DS epoch: time interval between the DS-block (2 ~ 3 hours)
  - TX epoch: time interval between the final TX-block (1 ~ 2 mins)

- Every DS epoch, nodes membership who participate in the network changes
  - Nodes membership is constructed with the nodes who send valid PoW solution in the given time to previous DS committee
    - Identity is defined by PoW solution
    - PoW solution is made with IP and pubkey
    - What is the input of PoW?
    - How the every node in the network can get the argument for the PoW?

- Every TX epoch, transaction is finalized
PoW

The nodes start PoW at the first time entering the network

One node can submit the solution up to two (This is only restricted in one Ds member’s point of view)

Possible scenarios

1. Found solution that meets DS diff and shard diff
   a. Submit sol and wait for DS-Block

2. Found solution that meets only shard diff
   a. Submit sol and continue to do PoW till DS diff met or DS block received

Any node who is sync with the network can get PoW function
DS committee

- DS committee members are managed by fixed-sized double-ended queue
- Oldest member will be pushed out from tail of queue
- How many leaders can be changed?
  - Up to 10 members (Why?)
Protocol overview

1. DS committee are elected
2. Shard the network and assign nodes to their shard
3. Process transactions in each shard
At the start of a DS-epoch, each node that want to join the network send a PoW solution.
Nodes that has successfully produced a valid nonce for PoW which meets the DS diff will be a new member of the next DS committee (up to ten)
When the time is up, the leader of previous DS committee initiates a consensus protocol to agree on the set of valid PoW solutions, which are randomly distributed on each shard or go into DS committee. DS committee runs a consensus on the new DS-Block header and then builds a signature part.
Protocol overview (2)

DS-Block header is committed to the DS blockchain of each DS member
New DS node joins the DS committee and the oldest member of the DS committee is churned out
Nodes who succeed to join the network will get the shard information and new DS committee member information from the old DS committee.

Kicked ex-DS nodes will be included in the shard member of the next DS epoch without PoW.

Publish these information to every DS nodes and to shard nodes by tree-base broadcasting.
Protocol overview (3)

After all the shard nodes get the new network information, they randomly form a number of shard each of which selects a consensus leader randomly.
Protocol overview (3)

Each shard processes disjoint sets of transactions
Since Zilliqa is an account based model, transactions are categorized by sender’s account
After finishing consensus(PBFT) on a set of transactions, each shard submits a Micro block to the DS committee (to five members including the leader)
Protocol overview (3)

Once at least one Micro block from each shard is received, DS leader initiates a consensus (PBFT) for Final block and spread it to each shard.

After getting a Final block, a new shard leader initiates a TX-Block consensus again. This is a one TX epoch.
Protocol overview (3)

Every 99th TX epoch of each DS epoch is vacuous epoch in which there is no other transactions but only incentive transactions. After receiving this final block, each node in shards start new PoW for the next epoch.
A peer gets into a blacklist if:

+ Its gossiped-message size exceeds the limitation: 5MB
+ Issues occur (unreachable, down, timeout, connection refused) when writing messages via socket or socket connection failed.
+ Callback connection buffer exceeds 20MB
+ Its reputation is less than -500
A few attack vectors

1. Make innocent nodes be blacklisted
2. Delay Pow start (Effective DoS attack)
3. Halt a shard (Heavy cost)
4. Partitioning the network (Heavy cost)
Identity generation in Zilliqa

For sybil resistant identity, Zilliqa use the PoW to define one node in the consensus.

Every start of epoch, each node should submit PoW solution to the DS committee in given time window.

After timeout, DS committee makes consensus on a set of PoW solution for the next epoch.
Attack vector

1. An attacker as a member of DS committee can submit PoW solution which includes tuple of ip and invalid(unused) port of the victim
2. Submitted PoW will be included in the membership of the next epoch
3. Any node who tries to make connection with the invalid tuple of ip:port which is submitted by an attacker regards the node paired with ip as unresponsive which means crash node
4. Since a leader will be selected randomly, that node can be a leader
5. Not only because the node is crash itself, but also crashing leader should be punished as it ruins liveness
6. Punishing will be executed by ip not a tuple of ip and port
7. Honest victim node using different port but same ip will be punished as well
Reasons of this attack

1. Discrepancy of identity definition in blacklisting and consensus
2. Punishing on crashing nodes
3. Multiple nodes in the DS committee can collect PoW solution
More constructional reasons

The problem arises from ByzCoin where multiple leaders append multiple blocks

Asking one leader to issue multiple blocks means that other network members will expect the leader to issue the next block. Therefore if this leader does not issue a block and ruin the liveness than proper penalty should be imposed.
Why this problem is not just confined to implementation issues

- IP:PORT blacklisting
- The necessity of punishing crashing node
- Single node for collecting PoW solutions
- PoW solution contains IP:PORT
Ip:port blacklisting

If the blacklisting is executed by ip and port pair then an attacker can easily spawn another node using different port with the same ip
The necessity of punishing crashing node

Sharding protocol uses a leader-based PBFT in each committee including DS committee to make a local decision.

In the leader-based PBFT protocol, unresponsive leader, so called crashing node, is regarded as an malicious and subject for punishing which is not the case in Bitcoin.
Single node for collecting PoW solutions

Developers may set a single node in the DS committee to collect PoW solution in the given time. It will prevent malicious nodes from submitting PoW solution including different ip:port(e.g victim’s ip:port) from the source of the TCP connection by which the PoW solution is submitted

Still, the single node and other nodes with TCP spoofing can launch the attack

Many bad things can be happen in WAN setting especially when the single node is collecting all the PoW (e.g DOS attack, attacks in ISP level)
• Contributions
• PoC

Results
Contributions

- Inform that this problem is not just applied to Zilliqa but to all the most of the recent sharding paper
- New secure design for identity generation
- ???
Proof of Concept (PoC)

- https://hackmd.io/_DHiA3JXSeqvbS8K9mxpRw?view
References

- The Road to Scalable Blockchain Designs
  https://www.usenix.org/system/files/login/articles/login_winter17_06_bano.pdf (USENIX 2017 winter magazine)
- OmniLedger: A Secure, Scale-Out, Decentralized Ledger via Sharding
- A Secure Sharding Protocol For Open Blockchains
- Enhancing Bitcoin Security and Performance with Strong Consistency via Collective Signing(ByzCoin)
- Bitcoin-NG: A Scalable Blockchain Protocol
References

- **RapidChain: Scaling Blockchain via Full Sharding**
  62%2E4D4702B0C3E38B35%2E6D218144511F3437&__acm__=1562676456_c8ddfc6d2c9f590380fe71d9f7abe1
  6a (CCS '18 Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security)

- **Chainspace: A Sharded Smart Contracts Platform**