ALZA: An Efficient Hybrid Decentralized Payment System

Draft Version

Nate Zou, Eric Li, Henry Zhang
by Alza Lab

Abstract—The efficiency of decentralized book systems like Bitcoin and Ethereum has always been a challenge. It is usually measured by three major factors: scalability, throughput and latency. Scalability refers to how the system capacity is increased by adding more physical resources. Throughput measures the volume of transactions for a given period of time, where most current solutions attempt to improve such as NEO, EOS, etc. Latency measures the processing time of any single transaction. In current blockchain based systems, the block generation rate is the main latency bottleneck. Off-chain processes such as state channels are the most recent work that can integrate partial inbound transactions, reducing latency. Unfortunately, the state channel introduces more issues at the same time, such as cross-channel synchronization, which makes the state channel unavailable for full adoption of current blockchain solutions.

In order to solve the efficiency problem, we proposed an end-to-end solution called ALZA, which links the dedicated high-throughput blockchain with coordinated self-organizing payment fields. This mechanism allows arbitrary set of users to create payment fields that process extremely low latency transactions within each field. Therefore, users can make transactions almost immediately. Since all transactions are conducted within fields, transaction costs will be reduced by several orders of magnitude. In addition, ALZA distributes main ledger to each client through an innovative replication mechanism. Therefore, the system will be significantly more robust to blockchain system failures. In theory, ALZA can complete millions of transactions in one second, which naturally supports high-frequency trading.

Index Terms—Payment Field, Blockchain, High Throughput, Low Latency, Low Cost, High Availability, Decentralization.

1 Introduction

The distributed ledger technology, better known as blockchain, provides a way to record and transfer data in a transparent, safe, audible and fault-tolerant way. It could reduce the traditional reliance on a centralized ledger managed by a trusted entity for holding and transferring funds or other assets. It has ability to potentially disrupt many industries such as banking, payments, supply chain, insurance, etc. in the upcoming decade.

In mainstream blockchain products, Bitcoin is the first cryptocurrency that shows peer-to-peer ledger technology. At the same time, it is also the first cryptocurrency to adapt to real-world conditions, including bringing the value into the cryptocurrency through a third-party centralized exchange. It lays the foundation for modern cryptocurrency, as it formalizes the most important components of the blockchain, including verification of transactions through digital signatures, proof of work consensus, and reward models for network maintainers. [10]

Ethereum, on the other hand, although it shares several key components with Bitcoin, including proof of work consensus mechanisms, it has also innovated from UXTO transactional storage as a stateful transactional storage approach. In addition, Ethereum adopts a smart contract mechanism in the blockchain world, thus introducing more practical applications. [15]

However, with the passage of time, cryptocurrencies have become more widely adopted in the world. A problem arises when the transaction volume changes from one order of magnitude to another. Since Bitcoin takes 10 minutes [9] block confirmation time, the bitcoin transaction become impractical to real world use cases. Besides, the entire ledger needs to be replicated at each mining node in the blockchain, therefore restricting the bitcoin network to process only 7 transactions per second, whereas visa payment can reach 56,000 transactions per second [14]. Obviously, there is a gap between usability and the way Bitcoin works today. Industries have brought up this question several times: What does it take to scale the blockchain? [5]

The limitations can be categorized as follows: 1) Speed of transaction processing. The block generation and consensus mechanism bottlenecks the transaction speed, it is an infrastructure level tradeoff. 2) Cost of processing. In the blockchain, participants help to maintain and update shared ledgers and share maintenance costs among participants instead of having one entity directly bear the costs and then charging one entity. In this sharing of responsibilities, participants can see the direct costs of increasing system operations. 3) Lack of governance. As many approaches trade security for speed, the risk of the shared ledger increases. Governance structures can improve the safety of a

• *Nate Zou, Eric Li and Henry Zhang are with the ALZA Laboratory, Mountain View, CA, 94043 (e-mail: contact@alza.io)*

blockchain structure (for example, by the involvement of a broad spectrum of stakeholders) or weaken it (for example, by limiting the decision making participants). 4) Privacy. In some blockchain system, all nodes have access to a copy of the ledger and may, if allowed, see all transactional history. However, in applying blockchain technology in the practice, participants may not want or be permitted to provide full visibility of the data. In such cases, access to information may be restricted.

Various solutions have been proposed to improve the scalability of the blockchain. For instance, Litecoin reduces the block generation time from 10 minutes to 2.5 minutes, while Dogecoin reduces it from 10 minutes to 1 minute [4], but they all tradeoff security for usability. As Ghassan and Elli suggests [8], under the UXTO proof of work mining system, decreasing of block confirmation time results in the increased probability of a double spend attack.

On other spaces, we have seen solutions to establish side chains on the main chain. Altcoins adjusts new block mechanism segwit and introducing external payment network like lightning network. Ethereum's new plasma implementation builds a multi-layered tree structure to allow for the removal of blocks and fault tolerance. In the event of a failure or attack, there are multiple sub-blockchains that a node can fall over to [11]. OMG blockchain innovates at the wallet level by introducing third party exchange and inter-wallet proof of stake payment protocol to create more processing parties for off-chain transactions. Over the time the exchanges will retire to encourage the use of wallet level exchange of value to retain decentralization [12]. These methods inspired us and led us to propose a groundbreaking system - the ALZA blockchain mechanism.

The ALZA system aims to provide practical solutions to the about limitations by establishing a self-sustained and coherent infrastructure. It smoothly hybrids on-chain and off-chain solutions, which further addresses the above limitations. In summary, we make the following contributions.

- We designed ALZA ledger block – a dedicated chain block which optimized for dPOS consensus. This fundamental building block helps the entire system reach sequential performance to handle high transaction volume.
- We designed an off-chain transaction mechanism named Self-organizing Payment Field. It provides ultra low latency and high fault tolerant transaction functionality. In addition, it summarizes local high frequency payments into batches, which reduces transaction cost to several order of magnitude.
- We designed a privacy controlling module named Privacy Sentinel, which provide full data ownership with privacy and security.
- We designed a data validation and recovery system called Redundant Array of Satellite Infrastructure (RaFi). It provides ultimate availability as well as high trustworthiness to the overall system.

The rest of the paper is organized as follows: In the following Section 2 we present the overall system architecture. In Section 3, we show the detailed designs and algorithms in each module. We finally conclude the paper in Section 4.

2 System Overview

Conceptually, this decentralized system can be described as two conceptual groups. A group is responsible for computationally heavy calculations, such as ledger calculations. We refer to each node in the group as a SuperNode. The other group performs the most lightweight peer-to-peer calculations, such as consensus. We call the node in this group SatelliteNode.

2.1 Super Node

Super node have four conceptual modules. 1) Blockchain Module. 2) Payment Field Controlling Module. 3) System Access Module. 4) High Availability Module.

Blockchain Module is the core part of the story, it is responsible for decentralized ledger generation and maintenance. We have subcomponents listed as follows:

- **Data Block Module** is the module that receives blockchain from other SuperNodes and stored in the file system. The module also takes care of validating blockchain from other SuperNode and maintaining the validity of MemPool. To maintain the validity of MemPool, the SuperNode should validate each coming transactions and consume them properly.
- **Network Module** maintains a smooth connectivity for multiple other parties to connect to, including other SuperNodes and satellite nodes.
- **Scheduling Module** manages the timing of order of the staking SuperNode. Revive a SuperNode itself when the turn is its own. Hibernate the SuperNode itself and pass on the proper data when the turn is off.
- **Consensus Module** conducts the blockchain consensus mechanism. Enforcing rules like the longest blockchain fork is selected and all the transaction signature is checked.
- **Monitoring and Logging Module** manages the health of the blockchain participants. Leaving proper trace of network visits to ensure the robustness of the Internet environment.

System Access Module shares the same components across the system, it controls all access to the system. Detailed subcomponents are listed as follows:

- **Registration Module** controls the accounts creation, modification and deletion.
- **Identification Module** enables the right individuals to access the right resources at the right times and for the right reasons.
- **Privilege Module** manages and audits account and data access by privileged users. A privileged user is someone who has administrative access to critical systems.
- **Membership Module** manages the interface for third-party docking, such as other chains, wallets, dispute services and etc.

Payment Field Controlling Module manages all connections between Satellite Node and Super Node. Subcomponents are as follows:

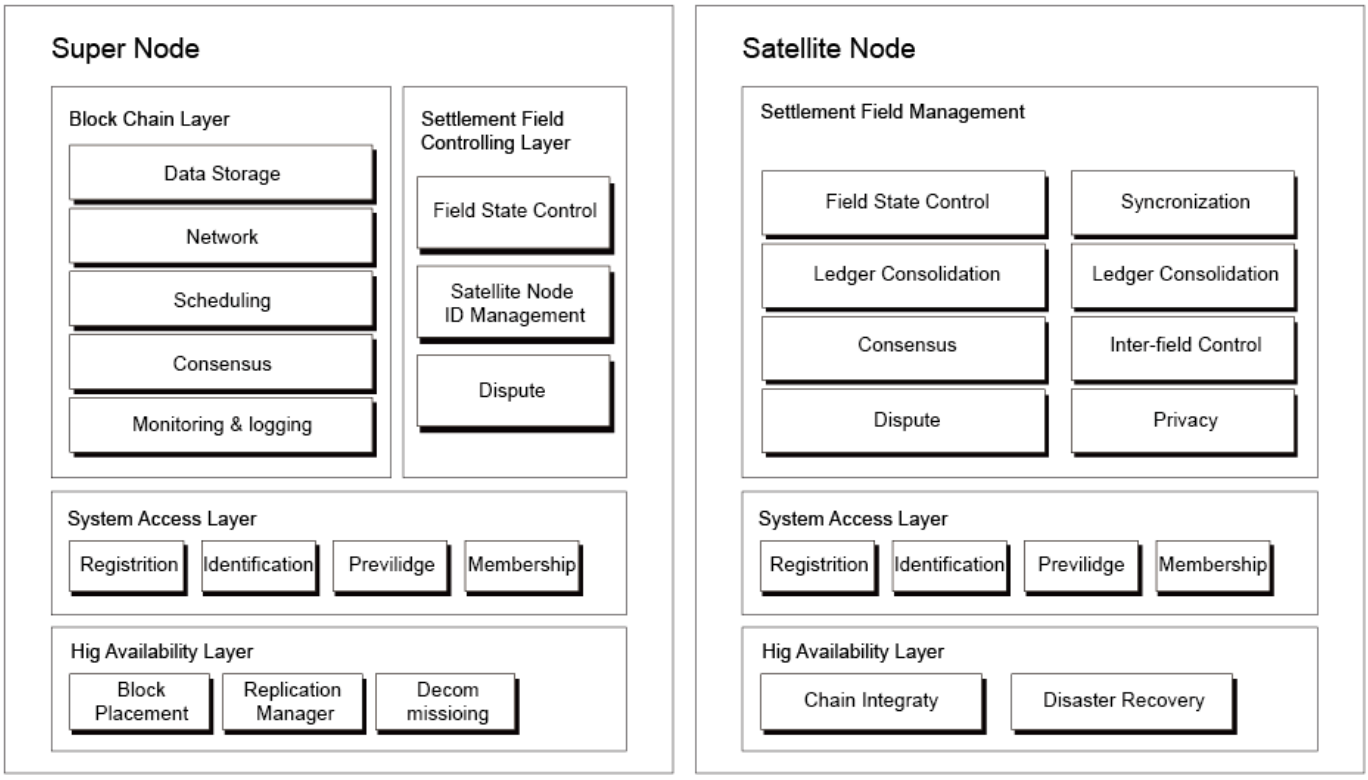

Fig. 1: Block generation procedure

- **Field Control Module** manages all self-organized fields by Satellite Nodes. It is responsible for synchronizes temporary ledgers with main ledger.
- **Reward Module** formulates different level of rewards for field leaders and participants.
- **Arbitration Module** manages and collaborates the evidences collected from Satellite Node for dispute. It is also responsible for docking third-party services.

High Availability Module refers to a system that is persistent and likely to operates continuously for a long time without failure. It is fulfilled by our specially designed system: Redundant Array of Satellite Infrastructure (RaFi). RaFi's subcomponents are listed as follows:

- **Block Placement Module** dictates the strategy of how and where to place replica data blocks in RaFi.
- **Replication Manager Module** enables data replications. It synchronizes the data on the destination data service with the data set on the source service, based on a specified replication schedule.
- **Decommissioning Module** manages the adding/removing for data nodes in RaFi dynamically.

2.2 Satellite Node

Satellite node have three major components. 1) Payment field management Module 2) System Access Module 3) High Availability Module. Subcomponents are listed as follows:

Payment Field Management Module is in pair with the Payment Field Controlling Module in SuperNode. However the differences are obvious, it manages in-field and cross-field activities locally. It contains several subcomponents as follows:

- **Field State Control Module** sends Off-Chain payment field initiate/close request to SuperNode, provide participant list and real-time update participant list to SuperNode.
- **Ledger Consolidation Module** consolidates current ledger locally based on transaction information received from other nodes.
- **Dispute Module** allows node to upload all relevant proof of the dispute case.
- **Role Management Module** manages state of these two roles, field leader and participant dynamically.
- **Privacy Module** hashes information like transaction details or dispute proof, before send them to ALZA blockchain.

System Access Module is sharing the same logic with SuperNode. It will support third-party access in the future.

High Availability Module is also in pair with SuperNode. It distributes the failure risk of SuperNode to numerous Satellites. The detailed submodules are:

- **Chain Integrity Module** checks the hash of the main ledger periodically. It serves as governance node to monitor SuperNode failures and is able to downgrade a unstable SuperNode in the node election.
- **Data Recovery Module** communicates with SuperNode High Availability Module to decides how data is replicated.

For better illustration, we list the notations in Table 1.

TABLE 1: Notations.

Symbol	Defination
$\mathcal{N}_{sup}$	The super node of ALZA. The quantity of super node is $\#\mathcal{N}_{sup}$.
$\mathcal{N}_{sat}$	The satellite node of ALZA. The quantity of super node is $\#\mathcal{N}_{sat}$.
$\mathcal{F}$	The self-organized payment field. The quantity of payment field is $\#\mathcal{F}$.
$\mho_{\mathcal{F}}$	Payment Hyper-Field.
$L_{\mathcal{F}}$	The leader of payment field. The quantity of payment field leader is $\#L_{\mathcal{F}}$.
$P_{\mathcal{F}}$	The participant of payment field. The quantity of payment field participent is $\#P_{\mathcal{F}}$.
$\mathcal{M}(x)$	Function for calculating Merkle tree of x.
$\Re(x)$	Function for calculating Reward of x.

3 Design Details

In this section, we describe how each component works and how do they hybrid together to orchestrate a payment procedure.

3.1 High Throughput

In ALZA , we employ the delegated proof of stake consensus mechanism. This contrasts with Bitcoin's proof of work system to provide a high throughput and low latency transaction processing capability.

In this delegated proof of stake (DPOS) system, we categorized nodes into SuperNodes and satellite nodes. The SuperNodes take care of the blockchain to ensure security and to prevent malicious parties from breaching the consensus

3.1.1 ALZA Block

As a hybrid system, ALZA mix on-chain and off-chain process. Hard splicing will certainly damages overall system performance. Therefore, in designing the ALZA Block, we adapt the Block structure to meet the need that the payment field establishing transactions and the payment field destructing transactions will be frequent to publish the the blockchain. The payment field establishing transaction will need to access data in the master node, though might not in the blockchain. The payment field destructure transaction will need to modify data in the master node, as well as saving the auditable transaction proof in the blockchain itself. All those requirements are the components that make the ALZA blockchain unique and effective. How does the payment field make ALZA effective will be elaborated in the later sessions.

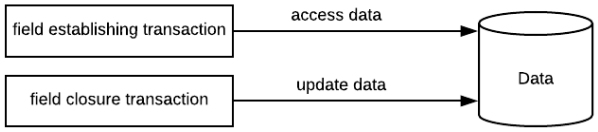

Fig. 2: ALZA block functional design.

Any blockchain system will have a consensus system to verify the transactions to achieve a network agreement. However the payment field in ALZA is not a direct part of blockchain and has no consensus mechanism in the field. Multiple transactions can happen in the field need to be verified somehow after disappearance of a payment field. ALZA introduce an additional merkle tree to store the payments in the field for a future verification. From the bottom up, the merkle tree can be calculated with equation 1:

$$\mathcal{M}(root) = \mathcal{H}(\mathcal{H}(left) + \mathcal{H}(right)), \tag{1}$$

where the $\mathcal{H}$ means any hash function, plus sign here represents concatenation. In the ALZA block, a transaction can come with one or more messages. A message is a special construct in an ALZA block. A message can be simply a transfer like transactions in the Bitcoin system. It can also be a complicated transfer from any number of senders to any number of receivers, which is not permitted in the Bitcoin system. Furthermore, a message can be a customized function call to a routine in the smart contract. Overly the message data field enable the native compatibility with smart contracts. So some essential contract code can be run directly on the bottom level of a chain instead of through a virtual machine?s execution rerouting.

A transaction also has multiple fields that are unique to the ALZA blockchain. They are code, scope, recipient, authorization and data field.

Code is also a field that specified the built-in function names a message can call into. Common example includes the simple transfer function, field establishing function, muture field closure function, solo field closure function used by when parties of a payment channel are missing and so on.

Scope field is to resolve data racing. Since a master node might have to read more than one piece of data in the field database when a field is established, and might have to write more than one piece of data when closure. A message will use a scope label to notify the master node with the accessing data ownership, which establishes an efficient way for the master node to concurrently access multiple pieces of data. This help accommodate the scenario where large amount of fields to establish or close at the same time.

Recipient field specifies more than one recipient to receive a message. It improves the flexibility to use limited amount of space to execute more syntax in the smart contract.

Authorization is a field for safety assurance. a piece of data in the master node can specify user rights to access. This implements a safe environment where transactions can include an authorization scheme making the ALZA blockchain more capable of using data with complicated rights management.

Data field is an encrypted field to include multiple parameters or other environment variables in the transaction. This field is designed to be the only place that is encrypted in the transaction, thus largely easing the burden a master node has to decode a large amount of data

Beyond the transactions, there will be a trailing space in the ALZA block where the field merkle tree is stored.

Overall the constructs above ensures the integration of smart contract into the ALZA blockchain. Also they allows ALZA to take various type of operations according to the design of the payment field system while allowing large quantity of payment fields to actively function and adapt to types of edge case, being fault tolerant.

3.1.2 Block Generation

As any blockchain projects, our consensus system produces blocks and saves blocks sequentially in the blockchain. In particular, SuperNodes are ones to produce blocks. There are 23 SuperNodes in ALZA . They take turns in producing blocks in a deterministic order.

Since ALZA employs the instrument of payment fields, ALZA's block structure contains reserved bits for accommodating the field establishment and field closure operations.

Also since ALZA allows transactions in the payment fields to exist anonymously, the reserved bits in the block take the anonymous transactions in the form of a payment field merkle tree.

The way to utilize the payment field merkle tree is that, the payment field merkle tree exists in the blockchain as part of the day to day blockchain. In case of a dispute, both parties from a dispute will present evidence to the SuperNode. The merkle tree will firstly locate the disputed transactions fast by traversing through the merkle tree. Secondly, the merkle hashes in the tree can be tested against the provided evident, such resolving to an unique consolidated result.

Here is an overview of how a SuperNode works,

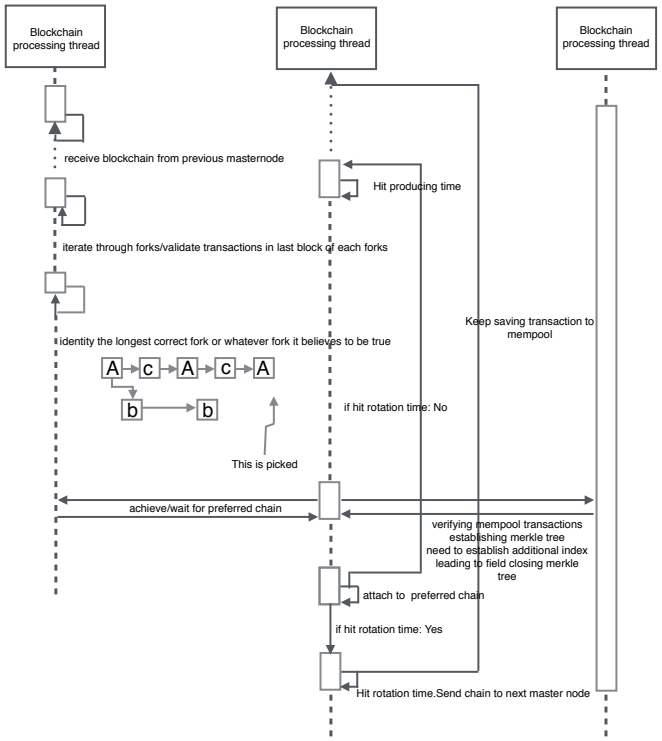

Fig. 3: Block generation procedure

The block producers are identified by ALZA's addresses. Also the order is predetermined in the voting process.

Not only the order for block producers to produce block is fixed, but also time slot for a block producers to produce block is scheduled. This means, if a block producer just generated a block, the next opportunity to produce a block is when every other nodes all spent their time slot. Even when the transaction amount is low, we do not expedite the time a node has to wait before finish its turn

Blocks are produced every 3 seconds. If the block is not produced at the scheduled time, the the block for that time slot is skipped. When one or more blocks are skipped, the gap is in multiple of 3 seconds

6 blocks are produced in each round in between 23 producers.

If a producer does not produce any block within the last 12 hours they are removed from the rotation until they notify the blockchain about their recovery. This is also an effort not trying to kick a block out of the rotation easily

Here is an example illustrating block production.

Algorithm 1 Block Generation Algorithm

Input: N lines of series $\{v^{(0)}, ..., v^{(t)}\}$. The number of mini batch is $\frac{N}{s}$.

Output: A trained model which is parameterized by $\mathbb{M}$: $\{W_{uh}, W_{uu}, W_{vu}, W, b_v, b_u\}$

Initialize u_0 and $\mathbb{M}$

First Pass

for each mini batch of size s **do**
  Initialize energy distance $\mathcal{C} = 0$
  for i from 0 to t **do**
    Get $u^{(i+1)}$
    Get $\widetilde{v}^{(i)}$ and $\widetilde{u}^{(i)}$
    $\mathcal{C} = \mathcal{C} + \mathcal{F}(v^{(i)}) - \mathcal{F}(\widetilde{v}^{(i)}) + \mathcal{F}(u^{(i)}) - \mathcal{F}(\widetilde{u}^{(i)})$
  end for
  $W_{vu} = W_{vu} - \lambda \frac{\partial \mathcal{C}}{\partial W_{vu} \cdot s}$
  $W_{uu} = W_{uu} - \lambda \frac{\partial \mathcal{C}}{\partial W_{uu} \cdot s}$
  $b_u = b_u - \lambda \frac{\partial \mathcal{C}}{\partial b_u \cdot s}$
end for
for i from 0 to t **do**
  Update $u^{(i+1)}$ given $\{v^{(i)}, W_{vu}, W_{uu}, b_u\}$
end for

Second Pass

for each mini batch of size s **do**
  Initialize energy distance $\mathcal{C} = 0$
  for i from 0 to t **do**
    Update $b_h^{(i)}$
    Calculate $\widetilde{v}^{(i)}$ replacing b_h with $b_h^{(i)}$
    $\mathcal{C} = \mathcal{C} + \mathcal{F}(v^{(i)}) - \mathcal{F}(\widetilde{v}^{(i)})$
  end for
  $W = W - \lambda \frac{\partial \mathcal{C}}{\partial W \cdot s}$
  $W_{uh} = W_{uh} - \lambda \frac{\partial \mathcal{C}}{\partial W_{uh} \cdot s}$
  $b_v = b_v - \lambda \frac{\partial \mathcal{C}}{\partial b_v \cdot s}$
end for

It is almost instant for a block producer to produce a block. There will be no need for a block producer to hash the data to a certain difficulty, but that block producer will only need to simply hash the data. This is because the block producers are trusted by the network. Nodes in the network votes for the block producers to validate transactions and

produce blocks over certain period of time, after which re-election will take place

3.1.3 Consensus and MemPool

ALZA software includes ALZA wallet, ALZA staking software and ALZA third party explorer. The main chain can use any decentralized consensus algorithm meeting the performance requirement. With the technology evolvement, many blockchain solutions can be our candidate. One recent algorithm meeting the is named Delegated Proof of Stake (DPOS).

This consensus system allows peers to stake to vote for trusted masternodes, forming a continuous approval mechanism. Any peer may register as a voting candidate and demonstrate their capability in terms of machine storage volume, computing power, band width and security measure to gain votes from the network general participants.

This consensus is not only fast and efficient, but also secure against various types of attack, including the famous nothing at stake attack against the normal proof of stake blockchain.

For a traditional proof of stake blockchain, an unintentional fork can be expected. Thus, nothing at stake will need caution from the development team. This means, in case of forking, the validation node can sign the both forks and double spend the fund. In case of ALZA , it is mandatory to force a longest fork to be selected. The final result will be to rule out an unexpected fork

Let us consider scenarios when forks can happen. If a node loses connections to other nodes in the network, this node will persist to produce block on its own. Over a period of time, this node will consider itself being the only productive block producer, thus only to generate only one block per other nodes with connectivity generating 20 blocks. Once the connectivity is resumed, the main chain is a lot more longer than the chain generated without connectivity

When malicious parties are involved in the block production, depending on how many parties were involved, the malicious parties are capable of generating a blockchain while multiple block producers are producing blocks. As long as the amount of block producers do not become the the majority of the SuperNodes, the righteous nodes are always capable of producing a chain with a longer length.

For the following code we can take a look at the fork processing ability,

Time is a key component of security. In most cases, it is not possible to know if the private key is compromised until it is used. Time-based security is even more important when people's applications need to store keys for everyday use on computers connected to the Internet. ALZA allows application developers to indicate that certain operations must wait for a period of time before they can be included in a block. In the meantime, they can be cancelled.

Users can receive notifications via email or SMS while broadcasting one of the actions. If they do not authorize, they can use the account recovery process to restore their account and take action.

The delay required depends on the sensitivity of the operation. There may not be any delay in paying for coffee, which is irreversible within a few seconds, while buying a house may require a 72-hour liquidation period. It may take up to 30 days to transfer the entire account to a new control. The exact delay is chosen by the application developer and user.

Algorithm 2 Fork Processing Algorithm

Input: A bigram dictionary extracted from dataset S; A BE pre-trained by S; Number of rounds N.
Output: Average distribution D between the word distribution of S and the the word distribution estimated by BE.

D = 0
repeat
  Initialize F = dict();
  Randomly pick a word w_i and extract all bigrams which contain w_i into vector B;
  for i in length(B) **do**
    $F_{B_i.w_{-i}}$ = B_i.frequency (w_{-i} indicate the other word in the bigram)
  end for
  loop
    feed one side of BE with w_i and get the reconstructed word $\tilde{w}_{-i}$ from the other side
    for j in length(B) **do**
      if B_j.contains $\tilde{w}_{-i}$ **then**
        F_j += 1
      end if
    end for
  end loop
  $D = F - B$.frequency
until N
$D = D$ / N

MemPool is not part of the blockchain, but it can meet customer needs. The node sends two types of transaction requests to the supernode. One is point-to-point transactions. The other is the ingredient transaction after the payment field is closed. So the transaction has been verified but it has not yet been confirmed. In the case of bulk transactions, internal on-site transactions are not uploaded, but the resulting Merkle tree is uploaded.

This code snippet shows what is the work on the MemPool server.

Algorithm 3 MemPool Algorithm

Input: A query sentence L; Φ_1, Φ_2, Φ_3 trained from dataset S; Bigram, trigram dictionary extracted from S
Output: Encoding E of L

n = 3, E = list()
run sliding window of size n and pick the n-gram N with the highest frequency
encode N using Φ_n and insert the encoding into E with the same relative order as N appears in L
L_1, L_2 = (L exclude N)
run step 2 on L_1, L_2 to get more segments, and run step 3 on these segments
stop when length($segments_i$) ¡ n or the picked n-gram's frequency less than a threshold σ
n -= 1 and repeat step 2-6

3.1.4 Permission management and Subjective Scheduling

Due to the consensus mechanism that ALZA employs, the SuperNodes take control over blockchain action and allowed action time. Not only SuperNodes are capable of checking signatures of each transaction, but also allow other essential activities to happen on the blockchain level such as arbitration for transactions that happen inside a payment field. Also ALZA allows the staking delegate scheme. The responsibility of a SuperNode and the opportunity to gain rewards are not beared by the SuperNode itself, but also the network participants who delegate their stake onto the SuperNode. This is a scheme that facilitate multi-user control over how the blockchain would work. When this scheme is applied, it should reduce the risk of being hacked

3.1.5 Rewards

ALZA allows a blockchain account to delegate its stake to a SuperNode. Through the staking operation the blockchain account can receive the portion of rewards in the SuperNode staking operation.

The SuperNode is the direct receiver who receives the staking rewards, although he is not entitled to the entirety of it. As introduced in the previous sectors, a SuperNode spends resources including time and computing powers in validating transactions, maintaining the blockchain integrity and serving as a transaction database. The node thus gets rewards for the operations. If the amount of staking stays the same, the rewards from one effective rotating time slot equals that from another. The rewards from a particular time slot are in proportion to the staking amount of the specific SuperNode.

In general, the staking rewards can be calculated through the following formula,

$$\mathcal{RW} = \frac{6 * 150 * 23 * SONO}{SONET}, \tag{2}$$

where RW: SuperNode staking rewards per each rotation period 6 : the amount of blocks to produce per each rotation period 150 : a reward constant 23: amount of SuperNodes SONO: Stake on this node SONET: Stake on the entire network In RW, the portion of reward that a staking account can share is calculated through the following formula,

$$\mathcal{IW} = \frac{RW * SOAN}{SONO}, \tag{3}$$

where IW: rewards for an individual account RW: SuperNode staking rewards per each rotation period SOAN: stake that belong to an account SONO: stake on the node

When a SuperNode generates a block, he will also produce a new transaction called a coinbase transaction. This transaction is from an nonexistent account 0x0, and paid to the SuperNode's account itself, also a group of staking nodes who are entitled. If during the block producing periods(which are many in a rotation period) there are closed payment fields, the SuperNode will gather the merkle the payment field leader produced and calculate the rewards according to the merkle tree's node count. This reward will also be paid from the account 0x0 and to the entitled state leader's account.

Note that the SuperNode also collect transaction fee as a form of rewards. When a SuperNode signs a block, there will be an additional coinbase transaction summing up the transaction fee from each transaction and also paid to the SuperNode's staking account.

Another form of rewards is for those satellite nodes who reserves data. The satellite nodes are responsible for proving its data reserving history to the SuperNode, thus the SuperNode can reward the data preserving capability accordingly.

3.1.6 Tokens, Accounts, Wallet

In ALZA, a wallet contains an ALZA account. An ALZA account can exist without a wallet. a wallet serves multiple purposes. Not only the wallet is a convenient software that display an account?s general information, but it also serves as a distributed storage of a portion of the main blockchain. To be exact, the entire blockchain in ALZA is a distributed hash table (or DHT) with added persistent using LevelDB and an interface to the blockchain. Data are sufficiently randomized across nodes to ensure high availability [16]. In case of disaster, the blockchain can be restored from all individual wallet node while maintain high availability.

An account is the address to a stakeholder. A stakeholder without a wallet is possible to delegate its stake to the SuperNode and participate in all sorts of voting [2].

Token is the value unit to use in the ALZA blockchain. The dividend to distribute in validation activity needs to be proportional to every participant?s stake in the master node. In this example, the stake is measured in the unit of ALZA tokens.

3.2 Low Latency and Low Fee

One of the most interesting developments in the blockchain space lately is state channels. They operate on the basic principle that in most cases, only the people affected by a transaction need to know about it. In essence, the transacting parties instantiate some state on a blockchain, e.g. an Ethereum contract or a Bitcoin multisig. They then simply send signed updates to this state between each other. The key point is that instead of them having to wait until the transaction has been validated and potentially finalized by the blockchain's consensus mechanism, state channel allows for transactions to be conducted as fast as information can be transmitted and processed by the parties. ALZA off-chain payment field is a step further on this state channel technology.

In the ALZA system, every stakeholders can delegate their stake to the main node. The delegation shows the level of trust one stakeholder put on the validator. In another word, the transaction fee would be minimal while everybody in the system can choose to profit from validation. As in the ALZA system not every single shareholder will participate in the validation, it can be certain that the validation cost will be much lower than a Proof of Stake system where every nodes participates in the validation. Assuming the validation cost exactly equals the transaction fee, a network is completely centralized and can only afford one validator. Assuming the transaction fee is 100x the cost of validation, the network can support 100 validators [3].

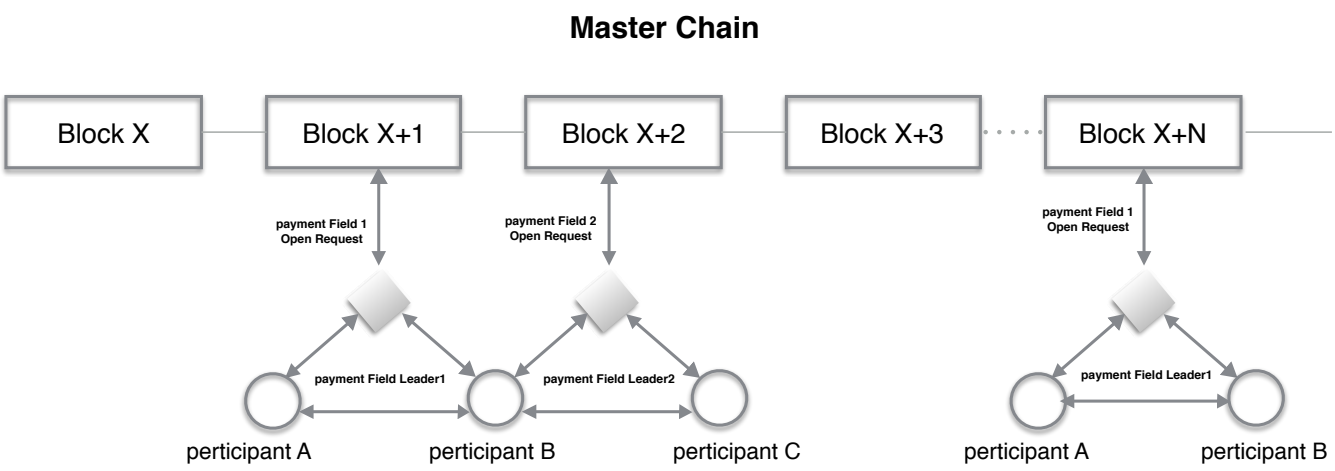

Fig. 4: Cross Field Algorithm.

3.2.1 Off Chain Payment Field

Instead of having to open payment channels between every pair of individuals that wish to make on-chain payments to each other, a linked payment field which utilizes a network of off-chain state channels from the sender to the receiver can be used. All satellite nodes can privately maintain and update a smaller ledger, such that their individual transactions are not required to be written to the blockchain.

At the same time, payment field guarantees that the participants can only spend their rightful amounts and that the ending balance can be written to the ALZA blockchain at any time.

Because payment field avoid transacting on the ALZA blockchain, they can in practice significantly improve the transaction throughput and reduce transaction latency. The transaction rate is effectively only limited by the network bandwidth between the participating peers. Another advantage of payment field is that they do not require the direct service of the blockchain miners, and therefore can perform transactions with lower transaction fees and consequently allow to economically perform micro-payments.

Payment Field Leader Before a payment field can commence, a leader needs to be claimed to act as a point of synchronization with ALZA blockchain for the protocol. Any node can claim to be the payment field leader when it is the node that will receive highest amount of deposits within the upcoming payment field. And upon receiving enough information about participants balances, calculate a set of the participants' current balances will be performed.

Payment Field Participant In response to the field initialization request, the participants reply with a participation confirmation which allows payment field leader to construct a list of who will be participant in the current payment field. This list is later used to enable the safe execution of the on-chain settlement. In a given payment field, participante can be as many as payment field leader allowed.

Off Chain Account Wallet All nodes will have off-chain account wallet to store their account balance and sync with their on-chain account on demand, which allows to be accessed by any other nodes. This will allow payment field leaders from different fields to sync the latest available balance of any given participant of their field.

Communication Network For the purpose of the off-chain payment field scheme we assume an underlying communication network, where all the participants can communicate directly off-chain (e.g. via a TCP connection).

3.2.2 Payment Field Protocol

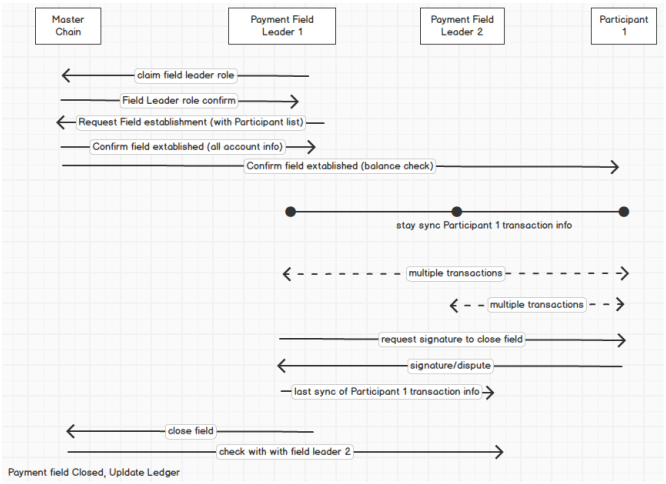

Fig. 5: Payment Field Algorithm.

Payment Field Establishment To initiate a payment field, the potential payment field leader will have to send a leader role claim to ALZA blockchain. Upon the ALZA blockchain confirmation, the payment field leader will need to send field establishment request with a list of field participants to ALZA blockchain and payment field leader is responsible for update the list real-time to ALZA blockchain. After receiving the payment field initiation request, ALZA blockchain will return all participants current balances. This will conclude the Payment field Open process and all participants of the payment field can start transactions within the network.

Field Consensus and Closure The transaction set, along with a list of participating members, can be sent in the form of a commitment to all nodes for verification and signing. The commitment is composed of the merkle-tree

Algorithm 4 Open-Close Field Algorithm

Input: A query sentence L; Φ_1, Φ_2, Φ_3 trained from dataset S; Bigram, trigram dictionary extracted from S
Output: Encoding E of L
1: n = 3, E = list()
2: run sliding window of size n and pick the n-gram N with the highest frequency
3: encode N using Φ_n and insert the encoding into E with the same relative order as N appears in L
4: L_1, L_2 = (L exclude N)
5: run step 2 on L_1, L_2 to get more segments, and run step 3 on these segments
6: stop when length($segments_i$) ¡ n or the picked n-gram's frequency less than a threshold σ
7: n -= 1 and repeat step 2-6

root of all transactions, and a hash of all the participants' public addresses (identities), both hashed together. (we will elaborate more in following chapter). When participants receive this commitment from the leader, they verify the proper construction of the hashes, and that the transactions are correctly recorded. Each participant then responds to the leader with its own signature on the commitment. Once all signatures are obtained by the leader, they can then consider the payment field ready to be closed, because the complete consensus on the transaction set can safely be considered as a binding state update for ALZA blockchain.

Participate Multiple Fields In the case when node X participates already involved in field A and request to join another field B.

- The leader of field B will be notified by ALZA blockchain that satellite node X already participant in another open payment field A.
- Field A leader address will be provided by ALZA.
- By communicate with field A leader directly, field B leader will be able to acquire all the transactions info that node X is currently involved.
- The two payment field leaders will then consolidate, sync and broadcast their ledgers to relevant participants.

3.2.3 Enforceability

For safety and efficiency purposes, we designed our off-chain payment field protocol to use the ALZA blockchain as a globale ledger recourse. A valid field that results from the full execution of this protocol must be enforceable, and thus updated to ALZA blockchain. Likewise, an invalid field, should not be enforceable. Field closure performed by the payment field leader are result in a ledger consolidation with ALZA blockchain. Prior to finalizing the settlement, the latest agreed upon balances of each channel peer must be broadcast on-chain in order to confirm that the amount to be withdrawn and deposited is correctly requested.

3.2.4 Privacy and Arbitration

ALZA excels at privacy in that the payment field does not disclose all the transactions the the SuperNode. In most of cases, the general public does not see the transactions that happen inside a payment field, although the general public does see the transactions that happen outside a payment field.

In a payment field, two or more satellite nodes participate. These satellite node has hardware equipment to maintain a temporary version of the participant's states. When a field state is live, the balance of each participant will be updated to multiple participants local copy. While a payment field closes, the transaction details are not submitted to the SuperNode. Rather, transactions hashes will be collected into a merkle tree, then to be uploaded to the SuperNode. Since a hash cannot deduce the real information, the information on the public blockchain cannot reveal an in-field transaction's details. In this sense, ALZA achieved the privacy goal.

However it is not that the in-field transactions will be hidden in every single scenario, but rather that transactions will be revealed at a business-consumer dispute. After the payment field round closure, if any participant request to dispute, it can issue an on-chain subsidized availability challenge for it. All proof can be submitted to ALZA blockchain, the proof will be Hashed and stored on the block ready to be reviewed anytime by a third party judge or ALZA blockchain.

The proof will be comprised of the complete payment field data, which includes the set of participants, their signatures and the merkle root of the transaction set. If this challenge is not answered in some predefined amount of time, the transaction round is considered annulled, and participants can safely assume that the latest state prior to the field is valid. The security lies in the assurance that in case of a dispute of payment or a need to withdraw deposits, the latest state of the ledger that the parties have agreed upon can be submitted to the blockchain and each party can claim its rightful balance.

Note that the security is not maintained by a third party verifier. As the payment field is closed, the system will send a closure message to the main chain, and sign the closure message with multiple participants' signature [1]. This signing operation is analogous to P2PKH in a bitcoin script, but utilizing multiple signatures. A subtlety here will be if another participant disappears when uploading the payment evidence, different closure measure would be required [7]. =When payment occurs in the payment field, privacy is also respected. Given an example we have a balance before the payment field exists,

```
Alice: 0.5 Bob: 0.5
```

When transacting within payment field, the balance state would change a few times like,

```
Alice: 0.5 Bob: 0.5 => Alice: 0.4
    Bob: 0.6 => Alice: 0.3 Bob: 0.3
```

However when the field closes, the interstates are not visible to the public, but rather the final state and transactions are uploaded to the blockchain. Most importantly, when each ALZA block is generated, the sender's address is encrypted with its private key, making it completely anonymous to the public.

3.3 High Availability

To prevent cluster level failures. We have a distributed self-replicating storage system that regularly pushes the main ledger to each satellite node to fully recover the latest snapshot of the ledger from the satellite. Inspired by the idea from Google File System [6] which is later opensourced as the well known Hadoop file system (HDFS) [13], we designed a lighter version which is more nailed to the edge computation scenario. We named it Redundant Array of Satellite File Infrastructure (RaFi). RaFi makes the HDFS idea available on mobile devices.

3.3.1 Roles

Edge Server The RaFi namespace is a hierarchy of files and directories. Files and directories are represented on Edge Server, which record attributes such as permissions, modification and access times, namespaces, and disk space quotas. The contents of the file are split into chunks, and each chunk of the file is replicated independently among multiple Satellite nodes (usually three, but the user can choose to file by file). The Edge Server maintains a mapping of the namespace tree and file blocks to the Satellite node (the physical location of the file data). The RaFi client (usually the Supernode) that wants to read the file first contacts the Edge Server to get the location of the data blocks that make up the file, and then reads the block contents from the Satellite node that is closest to the client. When writing data, the client requests the Edge Server to nominate a set of three Satellite nodes to carry a block copy. The client then writes data to the Satellite node in a pipelined manner. The current design has a Edge Server for each cluster. Each cluster can host multiple of Satellite nodes and RaFi clients because each Satellite node can execute multiple application tasks simultaneously.

RaFi saves the entire namespace in RAM. The list of data and blocks belonging to each file includes the metadata of a name system called an image. Persistent records of images stored on the local host's native file system are called checkpoints. The super node also stores the modification log of the image called log in the local host's native file system. For increased durability, redundant checkpoints and log copies can be created on other servers. During reboot, the super node restores the namespace by reading the namespace and replaying the logs. The location of the block copy may change over time, not as part of a persistent checkpoint.

Satellite nodes Each copy of the block on the Satellite node is represented by two files in the local host's native file system. The first file contains the data itself, and the second file is the metadata of the block, including the checksum of the block data and the block generation flags. The size of the data file is equal to the actual length of the block, and it can be rounded off with the nominal block size in a traditional file system without additional space. Therefore, if a block is half full, it only requires half of the full block on the local drive.

During the startup process, each satellite node is connected to the Edge Server and performs a handshake. The purpose of the handshake is to verify the Satellite node's namespace ID and software version. If any of them does not match the node of the Edge Server, the satellite node will automatically shut down. The namespace ID is assigned to the file system instance when it is formatted. The namespace ID is permanently stored on all nodes of the cluster. Nodes with different namespace IDs will not be able to join the cluster, thus maintaining the integrity of the file system.

Consistency of software versions is important because incompatible versions can result in data corruption or loss, and on large clusters of thousands of machines it is easy to ignore nodes that were not properly shut down before the software upgrade or were unavailable during the upgrade.

A Satellite node that is newly initialized and without any namespace ID is permitted to join the cluster and receive the cluster's namespace ID. After the handshake the Satellite node registers with the Edge Server. Satellite nodes persistently store their unique storage IDs. The storage ID is an internal identifier of the Satellite node, which makes it recognizable even if it is restarted with a different IP address or port. The storage ID is assigned to the Satellite node when it registers with the Supernode for the first time and never changes after that.

The satellite node identifies the block replica it owns to the super node by sending a block report. The block report contains the block ID, the generation tag, and the length of each block copy of the server host. The first blocking report is sent immediately after the satellite node registers. Subsequent block reports are sent every hour and provide the super node with the latest view of the location of the block copy on the cluster.

RaFi Client The user application uses the RaFi client to access the file system, which is a code library that exports the RaFi file system interface. In the initial phase of ALZA, the RaFi client is mostly the Supernode, for storing the main ledger.

Like most traditional file systems [13], RaFi supports reading, writing and deleting files, as well as creating and deleting directories. Users refer to files and directories through the path in the namespace. User applications typically do not need to know the file system metadata and store on a different server, or the block has multiple copies.

When an application reads a file, the RaFi client first asks the Edge Server for a list of Satellite nodes that are hosting a copy of the file block. It then directly contacts the Satellite node and requests the transmission of the required blocks. When the client writes, it first requires the super node to select the Satellite node to carry a copy of the first block of the file. The client organizes the pipeline from the node to the node and sends the data. When the first block is filled, the client requests to select a new Satellite node to host a copy of the next block. The new pipe is organized and the client sends more bytes of the file. The choice of each satellite node may vary.

3.3.2 File Read and Write

The application adds data to RaFi by creating a new file and writing data to RaFi. After the file is closed, the written bytes cannot be changed or deleted unless new data can be added by reopening the file and adding new data. RaFi implements a single-writer, multi-reader model.

A RaFi client that opens a file for writing is granted a lease for the file; no other client can write the file. The write client periodically updates the lease by sending a heartbeat

to the supernode. When the file is closed, the lease will be revoked.

RaFi files consist of blocks (storage block, not the block on blockchain). When a new block is needed, the super node assigns a block with a unique block ID and determines a list of Satellite nodes to carry a copy of the block. The satellite nodes form a pipeline whose order minimizes the total network distance from the client to the last satellite node. The bytes are pushed to the pipeline as a series of packets. The application writes bytes of the first buffer on the client. After the packet buffer is full (usually 64 KB), the data is pushed to the pipe. The next packet may be pushed to the pipeline before receiving an acknowledgment of the previous packet. The number of outstanding packets is limited to the client's outstanding packet window size.

In a Edge Server cluster, the failure of a Satellite node (the most common storage failure) is a daily event. Copies stored on the Satellite node may be damaged due to memory, disk, or network failure. RaFi generates and stores checksums for each block of data in the RaFi file. The RaFi client verifies the checksum while reading to help detect any damage caused by the client, satellite node or network. When a client creates a RaFi file, it calculates the checksum sequence for each block and sends it along with the data to the Satellite node. The Satellite node stores the checksum in a separate metadata file from the block's data file. When RaFi reads a file, the data and checksums of each data block are sent to the client. The client calculates the checksum of the received data and verifies that the newly calculated checksum matches the checksum it receives. Otherwise, the client notifies the supernode of a corrupted copy and then obtains a different copy of the block from another Satellite node.

When the client opens the file to read, it gets the list of blocks and the location of each block copy from the supernode. The location of each block is sorted by distance from the reader. When reading the contents of a block, the client first tries the closest copy. If the read attempt fails, the client will try the next copy in turn. If the target Satellite node is unavailable, the node no longer hosts a copy of the block, or the copy is found corrupted during the checksum test, the read may fail.

RaFi allows customers to read writable files. When reading a write file, the super node still does not know the length of the last block being written. In this case, the customer asks for the latest length of one of the copies before beginning to read its contents.

3.3.3 Block Placement

RaFi estimates the network bandwidth between two nodes by distance. Assume that the distance from the node to its parent is 1. The distance between two nodes can be calculated by summing them up to their nearest common ancestor. The shorter the distance between two nodes means that more bandwidth can be used to transmit data.

RaFi allows the administrator to configure a script that returns the node rack ID for a given node address. The supernode is the central location to resolve the gantry position of each satellite node. When a Satellite node registers with a super node, the super node runs a configuration script to determine which rack the node belongs to. If no such script is configured, the supernode assumes that all nodes belong to the default single rack.

Placement of replicas is critical to RaFi data reliability and read/write performance. A good copy placement strategy should improve data reliability, availability, and network bandwidth utilization. RaFi now provides a configurable block placement strategy interface so that users and researchers can experiment and test any strategy that best suits their application.

The default RaFi block placement strategy provides a trade-off between minimizing write costs and maximizing data reliability, availability, and aggregated read bandwidth. When creating a new block, RAFI local writer's location, the second and third copies to the first replica of the two nodes, the rest are placed at arbitrary nodes. The choice of placing the second and third copies on different Satellite node can better distribute block copies for individual files.

The default RaFi copy placement strategy can be summarized as follows: 1. No Satellite node contains multiple copies of any block. 2. If there are enough Edge Server on the cluster, no Edge Server cluster contains more than two copies of the same block.

3.3.4 Decommissioing

The RaFi administrator specifies which nodes can join the cluster by listing the host addresses of the nodes that are allowed to register and the host addresses of the nodes that are not allowed to register. Administrators can instruct the system to reevaluate these include and exclude lists. Excluded existing members of the cluster are marked as decommissioned. Once the Satellite node is marked as deactivated, it will not be selected as a target for replica placement, but it will continue to provide read requests. The supernode begins to schedule the replication of its blocks to other satellite nodes. Once the supernode detects that all blocks on the decommissioned satellite node have been duplicated, the node enters the decommissioning state. It can then be safely removed from the cluster without jeopardizing any data availability.

4 Conclusions

In this paper, we proposed an efficient Layer 1 & Layer 2 hybrid decentralized payment system named ALZA. It creatively marriages the benefits of blockchain's security with the efficiency of the proposed coordinated off-chain solution named Self-Organizing Payment Field. In theory, this infrastructure can stably complete millions of transaction per seconds, and minimizes the transaction cost at the same time. In addition, our proposed reliability component Redundant Array of Satellite File Infrastructure (RaFi) distributes the main ledger to every participant, giving governance power to everyone. RaFi is also able to fully recover the ledger when disaster failures happens on all SuperNode. Overall, ALZA will provide a seamless payment experience on electronic cash system.

Acknowledgment

We would like to thank Joyce Yang, Chris Liu, Howard Li and Allen Zhao for helpful comments and discussions.

DISCLAIMERS

Draft for open concept instruction. Algorithms are partial and figures are subject to change.

REFERENCES

[1] L. Apeltsin. A cryptocubic protocol for hacker-proof off-chain bitcoin transactions. *arXiv preprint arXiv:1408.2824*, 2014.

[2] ARK. Ark whitepaper. 2017.

[3] Bitshares. Delegated proof of stake. 2017.

[4] T. W. Club. Bitcoin vs litecoin vs dogecoin – cryptocurrency compared. 2015.

[5] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, et al. On scaling decentralized blockchains. In *International Conference on Financial Cryptography and Data Security*, pages 106–125. Springer, 2016.

[6] S. Ghemawat, H. Gobioff, and S.-T. Leung. *The Google file system*, volume 37. ACM, 2003.

[7] Z. Hess. Amoveo white paper. 2017.

[8] G. Karame, E. Androulaki, and S. Capkun. Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin. *IACR Cryptology ePrint Archive*, 2012(248), 2012.

[9] R. Khalil and A. Gervais. Revive: Rebalancing off-blockchain payment networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 439–453. ACM, 2017.

[10] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[11] J. Poon and V. Buterin. Plasma: Scalable autonomous smart contracts. *White paper*, 2017.

[12] J. Poon and O. Team. Decentralized exchange and payments platform. 2017.

[13] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on*, pages 1–10. Ieee, 2010.

[14] Visa. Visa inc. at a glance. 2015.

[15] G. Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151:1–32, 2014.

[16] G. Zyskind, O. Nathan, et al. Decentralizing privacy: Using blockchain to protect personal data. In *Security and Privacy Workshops (SPW), 2015 IEEE*, pages 180–184. IEEE, 2015.