IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 7, 2021, accepted January 25, 2021, date of publication February 1, 2021, date of current version February 8, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3056135

Bitmessage Plus: A Blockchain-Based Communication Protocol With High Practicality

LIUCHENG SHI, ZHAOZHONG GUO, AND MAOZHI XU
School of Mathematical Sciences, Peking University, Beijing 100871, China

Corresponding author: Maozhi Xu (mzxu@math.pku.edu.cn)

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant 62072011 and Grant 61672059.

ABSTRACT We are in the Internet era, when protecting the security of personal information is both vital and challenging. As the most commonly used methods of communication, centralized systems cannot meet the increasing need for information security. Blockchain, with its characteristics of openness, decentralization, and tamper resistance, is an innovative technology underlying Bitcoin. There is potential to use blockchain in developing decentralized and transparent communication systems. Bitmessage is a well-known decentralized messaging system that enables users to exchange messages and prevents accidental eavesdropping. Bitmessage achieves anonymity and privacy by relying on the blockchain flooding propagation mechanism and asymmetric encryption algorithm. Unfortunately, Bitmessage uses proof-of-work as the solution to prevent spam, which wastes computational power and makes it inefficient to be used in practice. To address this problem, we improve Bitmessage with a novel antispam mechanism based on proof-of-space, which requires the user to dedicate a certain amount of disk space to send a message. This improvement reduces the time and computing resource costs by eliminating computationally heavy hash operations. Moreover, we achieve a high level of anonymity by using the stealth address as the destination of the delivered message, which can only be identified by the intended receiver. Finally, we improve the protocol's reliability by taking the blockchain as an immutable database to store the delivered messages.

INDEX TERMS Decentralized communication system, blockchain, proof of space, stealth address, security analysis.

I. INTRODUCTION

Information technology pushes humanity to transfer from an industry society to an information society in which people communicate with each other more frequently. With the development of the Internet, protecting information and private communications has become more vital and harder. Currently, individual communications mostly rely on closed source and centralized systems, which requires trust that the providers (e.g., Google, Facebook, Microsoft, etc.) will protect users' information security. There is a risk that providers may provide users' information to the government for social network analysis [1], [2]. Even if the providers protect user privacy, the accidental leakage of information to third parties still occurs. Thus, a decentralized communication platform with anonymity and privacy has become an urgent need.

As an innovative technology, blockchain is essentially an open and distributed ledger of transactions that have been executed and shared by all participants [3]. Anyone is free to join the blockchain network with a digital address instead of a real-world identity. Moreover, rather than relying on a trusted third party, it achieves tamper resistance by using a combination of cryptography and consensus [4]. Finally, the flooding propagation mechanism makes it possible to deliver a message from the sender to the receiver without any direct communication. All these characteristics make the blockchain a compelling platform for developing distributed and secure communication protocols.

A. RELATED WORK

In recent years, a substantial number of blockchain-related communication protocols have been developed by academia and industry.

Bitmessage [5] is a trustless decentralized peer-to-peer messaging system with anonymity and privacy. Anonymity is achieved by broadcasting a message over the entire network, which is inspired by the blockchain's flooding propagation

The associate editor coordinating the review of this manuscript and approving it for publication was Haipeng Yao.

mechanism and makes it impossible to identify the communication parties. Privacy is instead achieved by encrypting the message with the receiver's public key; thus, only the intended receiver can decipher its content. Even so, Bitmessage has not been widely used in practice due to its inefficiency. To prevent spam, a proof-of-work must be completed in the form of a partial hash collision to send a message, which takes an average of four minutes and wastes energy. If the receiver is offline, the sender has to recompute the proof-of-work to rebroadcast the message. Moreover, as messages are sent to all nodes in the network, the receiver has to decipher each message to see whether it is bound for him.

Proof-of-work in Bitmessage reduces the risk of message pollution and distributed denial of service (DDoS) attack from spammers, but it also makes it hard to send legitimate messages. An improved antispam mechanism, which aims to design a new proof-of-work formula that can effectively protect the system from spam while allowing legitimate users to send messages at a reasonable cost, is proposed in [6]. Unfortunately, this solution cannot fundamentally solve the problem as computationally heavy hash operations are still required.

The Distributed Communication Channel (DCC) [7] is a new communication method that adapts the distributed storage system (DSS) concept to achieve reliability and security. It uses the blockchain as a tool to bypass the transformation matrix and exchange encryption keys between the sender and receiver. However, the communication relationship is easy to detect in the DCC. Cryptouch [8] is another blockchain-based communication application that introduces the interplanetary file system (IPFS) [9] to overcome the data storage limits of the blockchain. Currently, the system is designed to be used as a publicly accessible database; however, the communication functionality is still under development.

B. OUR CONTRIBUTIONS

We present ''Bitmessage Plus'', an improved version of Bitmessage. Our protocol achieves characteristics of low computing resource consumption, high efficiency, and high level of user anonymity, which make it a practical blockchain-based communication system The contributions are summarized as follows:

- We use the blockchain as a publicly accessible database to store the delivered messages. Users communicate with each other by sending and reading transactions in the blockchain. Due to the blockchain's tamper-resistance property, the delivered massages stored cannot be hacked or modified by anyone, which greatly improves the protocol's reliability.
- We propose a novel antispam mechanism based on proof-of-space, which is inspired by alternative efficient protocols designed to replace proof-of-work in blockchain consensus research. Specifically, to send a message, the user is required to dedicate a certain amount of disk space according to the sent message's size and lifetime. This improvement reduces the time and computing resource costs to send a legitimate message by eliminating computationally heavy hash operations while also preventing spammers from sending unlimited messages. We find that the new antispam mechanism improves our protocol's practicality
- We offer a high level of anonymity for the communication parties using cryptography tools. Since the blockchain is pseudoanonymous, heuristic analysis of transactions may reveal a user's real identity. To solve this problem, the transaction carrying the delivered massage is sent to a stealth address generated according to the receiver's public key. Only the intended receiver can identify this transaction using his private key, which protects the communication relationship and achieves anonymity.
- We provide an efficient method for a receiver to identify his messages from a flood of transactions. Specifically, a receiver no longer has to decrypt each received message and just checks whether the stealth address in each transaction belongs to him. This design further improves our protocol's efficiency.

C. PAPER ORGANIZATION

The remainder of the paper is organized as follows. In Sec. II, we present the preliminaries, including the cryptography tools and a brief introduction to proof-of-space. We provide a detailed description of the data structure and protocol procedure of our protocol in Sec. III. The security analysis is presented in Sec. IV. Finally, we make a conclusion in Sec. V.

II. PRELIMINARIES

A. CRYPTOGRAPHY TOOLS

1) STEALTH ADDRESS

A stealth address [10] is a technique widely used in blockchain systems to provide anonymity for the transaction receiver. Specifically, the sender generates a one-time address (OTA) efficiently based on the receiver's public key. The receiver can identify the one-time address using his private key. We abstract this technique as an ideal functionality $\mathcal{F}_{OTA}$, which is formally defined in Functionality 1.

Functionality 1 The Stealth Address Functionality

Functionality $\mathcal{F}_{OTA}$ works as follows:

- Upon receiving (`generate`, pk_i) from party P_j, generate a one-time address *ota* record (ota, pk_i), and send *ota* to P_j.
- Upon receiving (`verify`, ota, sk_i) from party P_i if some (ota, pk) is recorded and $pk = pk_i$, then send *True* to P_i; otherwise, send *False* to P_i.

In our protocol, a stealth address is used to protect the relationship of the message sender and receiver. To simplify our protocol description, we use $ota = GenOTA(pk_i)$ to denote sending (`generate`, pk_i) to $\mathcal{F}_{OTA}$ and receiving *ota* from $\mathcal{F}_{OTA}$. Similarly, $tag = VerOTA(ota, sk_i)$ denotes sending

$(\texttt{verify}, ota, sk_i)$ to $\mathcal{F}_{OTA}$ and receiving tag from $\mathcal{F}_{OTA}$ Concrete instantiations of $\mathcal{F}_{OTA}$ can be found in [11].

2) RANDOM BEACON

The concept of a public random beacon, which aims at continuous provision of randomness at regular intervals, was first proposed and formalized in [12]. We abstract it as an ideal functionality $\mathcal{F}_{RB}$, which is formally defined in Functionality 2.

Functionality 2 The Random Beacon Functionality

Functionality $\mathcal{F}_{RB}$ works as follows:

- Upon receiving $(\texttt{get}, sid)$ from party P_j, generate a random number r record (sid, r), and send r to P_j If some $(sid, *)$ is already stored, then ignore the message.

In our protocol, a random beacon is used in the block proposer selection process to prevent manipulation. For simplicity, we use $r = GetRB(sid)$ to denote sending $(\texttt{get}, sid)$ to $\mathcal{F}_{RB}$ and receiving r from $\mathcal{F}_{RB}$. Concrete instantiations of $\mathcal{F}_{RB}$ can be found in [13]–[15].

B. PROOF-OF-SPACE

Proof-of-space [16] is an alternative protocol for proof-of-work and consists of two phases, i.e., the initialization phase and execution phase, between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$

In the initialization phase, $\mathcal{P}$ picks a graph $G = (V, E)$ from a family of "hard-to-pebble" directed acyclic graphs [16], [17] depending on the amount of dedicated space, saves the nodes' labels and the corresponding Merkle tree, and then sends the Merkle root to $\mathcal{V}$ as the commitment. Each node's label l_i is computed as $l_i = hash(\mu, i, l_{p_1}, l_{p_2}, \ldots, l_{p_t})$, where $p_1, \ldots, p_t$ are the parents of node i and μ is the identity of $\mathcal{P}$. In the execution phase, $\mathcal{V}$ sends a challenge to $\mathcal{P}$ who returns a short answer after reading a small fraction of his storage. Based on the commitment, $\mathcal{V}$ verifies the answer of the challenge and outputs true or false.

To tackle the interactivity of challenge presentation, we use the Fiat-Shamir paradigm in [18] for challenge generation through a public random number in a noninteractive way. We abstract this technique as an ideal functionality $\mathcal{F}_{PoS}$, which is formally defined in Functionality 3.

In our protocol, proof-of-space is used to prevent spammers from sending messages limitedly. For simplicity, we use $cmt = Commit(space, \mu)$, $RC = GenRC(s, r)$, $pf = GenPf(space, RC, \mu)$, and $tag = VerPf(cmt, RC, \mu, pf)$ to denote the operations in the proof-of-space functionality. Concrete instantiations of $\mathcal{F}_{PoS}$ can be found in [16]–[18].

III. PROTOCOL DESCRIPTION

In this section, we first describe the data structure of the transaction, block, and chain. Then, we present the technical details of our protocol.

Functionality 3 The Proof-of-Space Functionality

Functionality $\mathcal{F}_{PoS}$ works as follows:

- Upon receiving $(\texttt{commit}, space, \mu)$ from $\mathcal{P}$ with identity μ, if some $(space, cmt, \mu)$ is recorded, send cmt to $\mathcal{P}$; otherwise, generate a commitment cmt for the storage $space$ record $(space, cmt, \mu)$, and send cmt to $\mathcal{P}$.
- Upon receiving $(\texttt{genrc}, s, r)$ from $\mathcal{P}$ or $\mathcal{V}$ if some (s, r, RC) is recorded, return RC; otherwise, generate a random challenge RC for a space of size s through random number r, record (s, r, RC), and return RC.
- Upon receiving $(\texttt{genproof}, space, RC, \mu)$ from $\mathcal{P}$, if some $(space, RC, \mu, pf)$ is recorded, send pf to $\mathcal{P}$; otherwise, generate a proof-of-space pf for space against challenge RC, record $(space, RC, \mu, pf)$, and send pf to $\mathcal{P}$.
- Upon receiving $(\texttt{verproof}, cmt, RC, \mu, pf)$ from $\mathcal{V}$, if some $(space, cmt, \mu)$ and $(space, RC, \mu, pf)$ are recorded, send *True* to $\mathcal{V}$; otherwise, send *False* to $\mathcal{V}$.

A. DATA STRUCTURE

1) TRANSACTION

There are three kinds of transactions, i.e., registration transactions, communication transactions, and Coinbase [19] transactions. The registration transaction is constructed to register an address, which will be used as the origin of the communication transaction. The communication transaction is used to deliver a message. The Coinbase transaction is the unique transaction in each block used to encourage users to participate in a consensus by rewarding a constant size of space. Each type of transaction is signed by the user generating it, then broadcasted in the blockchain network, and finally recorded in some block. The structures of different kinds of transactions are shown as follows:

- The registration transaction tx_reg contains the following:
  - Tag, "`register`".
  - Address, the registrant's public key.
  - Nonce, a parameter selected by the registrant.
  - Signature, the registrant's signature of the transaction.
- The communication transaction tx_com contains the following:
  - Tag, "`communicate`".
  - Origin, the sender's address.
  - Destination, the receiver's stealth address.
  - Msghash, the hash value of the delivered message.
  - Msgsize, the size of the delivered message.
  - Lifetime, the interval of the delivered message‘s existence in the blockchain.
  - Preproof, the proof of the sender's dedicated disk space.
  - Cmt, the commitment of dedicated disk space.
  - Signature, the sender's signature of the transaction.

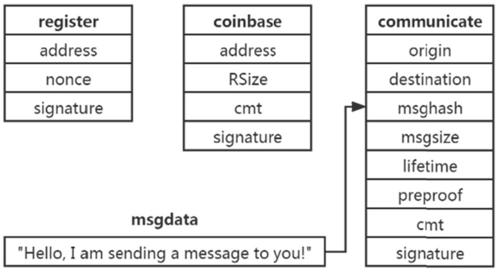

FIGURE 1. Transaction structures.

- The Coinbase transaction *tx_cb* contains the following:
  - Tag, "`coinbase`".
  - Address, the block proposer's public key.
  - RSize, the constant space size rewarded to the block proposer.
  - Cmt, the commitment of dedicated disk space.
  - Signature the block proposer's signature of the transaction.

2) BLOCK

A block consists of three parts, i.e., a block header, a block body, and a block pocket. Their structures are shown as follows:

- The block header contains the following:
  - Block number, the current block index.
  - Parent hash the previous block's hash value.
  - Timestamp, the creation time of the block.
  - Merkle root, the root of the Merkle tree of the transactions.
  - Rndnum, a random number generated by Random Beacon.
  - Counter, a number that constantly increases with time.
  - Address, the block proposer's public key.
  - Signature, the block proposer's signature of the block header.
- The block body contains the following:
  - Block number, the current block index.
  - A list of transactions.
- The block pocket contains the following:
  - Block number, the current block index.
  - A list of delivered messages.

The three parts are combined using the same block number. The transactions recorded in the block body are hashed in a Merkle tree [20]–[22] with only the root included in the block header. Each of the delivered messages stored in the block pocket is related to a communication transaction in the block body through msghash.

3) CHAIN

The chain is a sequence of blocks serving as a public ledger of all transactions, where the latter block header links to the former one through the parent hash.

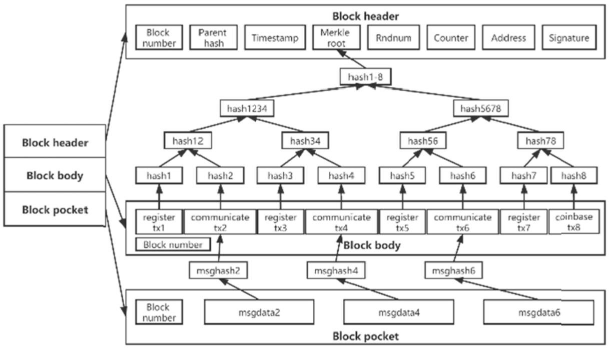

FIGURE 2. Block structure.

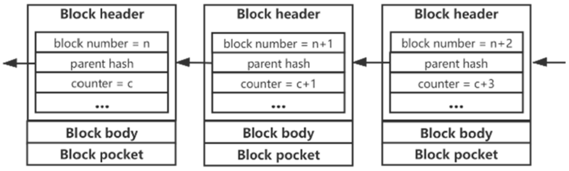

FIGURE 3. Chain structure.

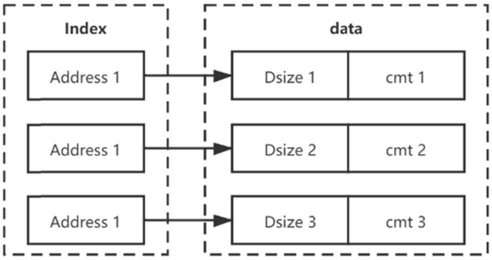

FIGURE 4. Worldstate structure.

As the chain grows, the block number increases constantly block by block. The counter works like a global clock by increasing with time in regular intervals. However, due to the uncertainty of block production, the counter value recorded in each block increases nonuniformly.

4) WORLDSTATE

The worldstate is a local database generated according to the transactions stored in the blockchain. Specifically, it consists of a list of records, whose structure is shown below.

- The record contains the following:
  - Address, the user's public key.
  - Dsize, the amount of dedicated disk space.
  - Cmt, the commitment of dedicated disk space.

Each record is indexed by the user's public key to record the amount of dedicated space and the corresponding commitment.

B. PROTOCOL SCHEME

In the protocol, there are logically two roles, i.e., clients who send messages through communication transactions and block proposers who participate in the consensus process to

TABLE 1. Used variables and functions.

Notation	Definition
M	The delivered message
$Dspace$	Local disk space dedicated to communication
$Hash()$	Secure hash function with output uniformly distributed in $\mathbb{S}$
$Enc()$	Encryption algorithm
$Dec()$	Decryption algorithm
$Sign()$	Digital signature generation algorithm
$Verify()$	Digital signature verification algorithm
$GenOTA()$	Stealth address generation algorithm
$VerOTA()$	Stealth address verification algorithm
$GenRC()$	Random challenge generation in $\mathcal{F}_{PoS}$
$Commit()$	Commitment generation in $\mathcal{F}_{PoS}$
$GenPf()$	Proof generation in $\mathcal{F}_{PoS}$
$VerPf()$	Proof verification in $\mathcal{F}_{PoS}$
$GetRB()$	Get a random number from Random Beacon
$Broadcast()$	Send a transaction to the blockchain network
$Download()$	Download transactions from the blockchain
$Size()$	Compute the size of msgdata
$ReadSize()$	Read the dedicated space size from worldstate
$ReadCmt()$	Read the space commitment from worldstate
$GetCounter()$	Get the counter value according to the current time
$Update()$	Reconstruct the dedicated space
res	Size of the dedicated space to be released

Algorithm 1 Registration

The user executes the following steps in the registration process:

1. Randomly generate a key pair (pk, sk) locally.
2. Complete a proof-of-work by scanning for a *nonce* satisfying
$$hash\,(pk, nonce) < target.$$
3. Construct the registration transaction tx_reg
$$sig = Sign(register||pk||nonce, sk),$$
$$tx\_reg = (register, pk, nonce, sig).$$
4. Broadcast tx_reg
$$Broadcast(tx\_reg).$$

tx_reg will be verified as follows:

1. Verify the proof-of-work by checking
$$hash\,(pk, nonce) < target.$$
2. Verify the signature
$$Verify\,(register\,||pk||\,nonce, pk, sig)\,.$$

If the above verification succeeds, the transaction will be included in a block. Thus, the address pk is successfully registered.

provide liveness and persistence for the whole blockchain system. We present the protocol scheme in three parts, i.e., the registration process, the communication process, and the consensus mechanism. We first list a few notions in Table 1.

1) REGISTRATION PROCESS

To register a valid address, a proof-of-work must be completed in the form of a partial hash collision, which is shown in Algorithm 1.

We remark that *target* is adjusted at regular intervals to ensure that it always takes constant time to register an address. Such proof-of-work is necessary to protect the blockchain system from Sybil attacks. Moreover, we can implement *hash*() with a memory-hard hash function [23] such as Equihash [24] to reduce the advantage of using dedicated hardware, such as ASIC [25], to speed up hash operations for registration.

2) COMMUNICATION PROCESS

Assume that Alice with key pair (pk_a, sk_a) sends a message M to Bob with key pair (pk_b, sk_b) in the block numbered Bn. The procedure is shown in Algorithm 2.

We remark that the stealth address is used as the destination of tx_com to protect the communication relationship of Alice and Bob, which achieves communication privacy. Moreover, a certain amount of space needs to be dedicated by Alice as the communication costs to prevent spam. The amount of dedicated space is proportional to the message size and lifetime.

Bob identifies tx_com from all the transactions recorded in the blockchain by verifying the stealth address in each transaction, downloads the corresponding *msgdata*, and decrypts to obtain M using his private key. The procedure is shown in Algorithm 3.

3) CONSENSUS MECHANISM

The consensus mechanism ensures the blockchain's persistence and liveness. The mechanism consists of two core components, i.e., leader selection and chain selection. The former determines the proposer of each block and the latter determines the unique valid chain.

Regarding leader selection, we use a key verifiable random function (VRF) technique [26] to randomly select block proposers in a private and noninteractive way. Specifically, each consensus participant can independently determine if he is chosen to be the block proposer by computing a function of his private key and public information from the blockchain, which can be verified by his public key. Moreover, the uniform distribution of the VRF's output ensures that the probability of a candidate being selected is proportional to its fraction of the total dedicated disk space, which means that the candidate can obtain advantages with more dedicated disk space for communication. The procedure is shown in Algorithm 4.

Algorithm 2 Message Sending

```
Alice executes the following steps to send $M$ to Bob:
1. Generate a stealth address for Bob
       $ota = GenOTA(pk_b).$
2. Encrypt $M$ with Bob's public key
       $msgdata = Enc(M, pk_b).$
3. Set the lifetime
       $l = N.$
4. Compute msghash and msgsize
       $h = hash\,(msgdata),$
       $s = Size(msgdata).$
5. Generate the preproof
       $r = GetRB(Bn),$
       $v = ReadSize(worldstate, pk_a),$
       $RC = GenRC(v, r),$
       $pf = GenPf\,(DSpace, RC, pk_a).$
6. Update DSpace and generate a new commitment
       $t = v - res + msgspace,$
       $Update(DSpace, t),$
       $c = Commit(DSpace, pk_a).$
7. Construct communication transaction $tx\_com$
   $sig = Sign(communicate||pk_a||ota||h||s||l||pf\,||c, sk_a),$
   $tx\_com = (communicate, pk, ota, h, s, l, pf\,, c, sig).$
8. Broadcast $tx\_com$ and $msgdata$
       $Broadcast(tx\_com, msgdata).$
$tx\_com$ and $msgdata$ will be verified as follows:
1. Check whether the following equation holds
       $h = hash\,(msgdata)\,.$
2. Verify the signature
       $Verify(communicate||pk||ota||h||s||l||pf\,||c, pk_a, sig).$
3. Verify $pf$ with $RC$
       $cmt = ReadCmt(worldstate,\ pk_a),$
       $VerPf\ (cmt, RC, pk_a, pf)\,.$
If the above verification succeeds, the transaction will be
included in a block. Thus, $M$ is successfully sent from Alice.
```

Algorithm 3 Message Receiving

```
Bob executes the following steps to get $M$ from Alice:
1. Download the communication transactions from the blockchain
       $TxSet = Download(chain).$
2. Verify the stealth address $ota$ of each transaction in $TxSet$
       $VerOTA\ (ota, sk_b).$
3. If the verification succeeds, download the corresponding
   $msgdata$ from the block pocket, and decrypt to get $M$
       $M = Dec(msgdata, sk_b).$
```

Algorithm 4 Leader Selection

```
Assume that a candidate has key pair $(pk, sk)$, the amount
of dedicated disk space for communication is $S_u$ and the total
size in the network is $S_t$. $Bn$ denotes the block number, and $Ph$
denotes the parent hash. The candidate executes the following
steps to check whether he is selected as the proposer for the
block with block number $Bn$:
1. Get random number and counter
       $r = GetRB(Bn),$
       $c = GetCounter().$
2. Compute
       $\lambda = hash\,(Bn||Ph||r||c||pk).$
3. Check whether $\lambda$ satisfies
       $\lambda < (S_u/S_t) \cdot |\mathbb{S}|.$
If the inequality holds, then the candidate is selected as the
valid block proposer.
```

We remark that any registered user can be a candidate for leader selection. Since the output of *hash*() is uniformly and randomly distributed in $\mathbb{S}$, the probability of being selected as a block proposer equals S_u/S_t. For simplicity, we abstract the leader selection algorithm as function *Leader*(), which takes *pk*,*Ph*, and *Bn* as its inputs and outputs true if and only if the candidate with public key *pk* is selected as the valid proposer for the block with block number *Bn*.

After being selected, the block proposer generates a valid block containing a Coinbase transaction and broadcasts it to the blockchain network. The block will be received by other users to extend their chains after verification.

Regarding chain selection, we apply the longest rule to identify the authoritative chain from several chains. Moreover, we also introduce the weight concept to handle the situation where there are several longest chains. Specifically, the weight of block B is computed as $w\,(B) = 2^{-\lambda}$, where λ is the hash value in Algorithm 4. The chain's weight is defined as the sum of the composed blocks' weights. The chain selection procedure is shown in Algorithm 5.

The worldstate is constructed according to the transaction sequence in the authoritative chain. Specifically, it is modified as follows:

- For a registration transaction, a new record indexed by the registered address will be inserted
- For a communication transaction, its msgsize will be added to the sender's dedicated disk space, and the

Algorithm 5 Chain Selection

Users execute the following steps to identify the authoritative chain:

1. Collect all valid chains received via a broadcast into a set $\mathbb{C}$.
2. Construct a subset of $\mathbb{C}$ with chains of the maximum length

$$\hat{\mathbb{C}} = \{C_1, C_2, \ldots, C_n\} \subseteq \mathbb{C}.$$

3. Compute the weight of each chain in $\hat{\mathbb{C}}$

$$C_i = B_{i,1}|B_{i,2}|\cdots|B_{i,l},$$
$$w\left(C_i\right) = \sum\nolimits_{j=0}^{l} w(B_{i,j}).$$

4. Select the chain with maximum weight from $\hat{\mathbb{C}}$

$$C_v = C_{\underset{i\in[1,n]}{\operatorname{argmax}} w(C_i)}.$$

msgsize of the expired communication transactions will be deducted. Moreover, the sender's space commitment will be replaced by cmt.

- For a Coinbase transaction, its RSize will be deducted from the sender's dedicated disk space, and the sender's space commitment will be replaced by cmt.

A slimming mechanism is implemented to save storage space by deleting the msgdata of expired communication transactions from the corresponding block pockets.

We present the consensus mechanism in Algorithm 6.

Algorithm 6 Consensus Mechanism

The consensus mechanism proceeds as follows:

Initialization A set of preregistered users will be contained in the first block B_0, called the genesis block, and B_0 is set as the local blockchain.

Chain Extension Each participant with key pair (pk, sk) performs the following steps:

1. Collect and verify the transactions received via a broadcast and save the valid transactions in the transaction pool.
2. Collect all valid chains received via a broadcast and select the authoritative chain $C_v = B_1|B_2|\cdots|B_l$ as the local chain.
3. Construct the worldstate according to C_v.
4. If $Leader(pk, Ph, Bn) = True$, where Ph is the hash value of B_l and Bn is B_l's block number increased by 1, the block proposer generates a new block B, constructs the new chain $C = C_v|B$, slims C, broadcasts C, and sets Cit as the new local chain.

IV. SECURITY ANALYSIS

In this section, we analyze the protocol's security from two aspects, i.e., blockchain security and spam resistance. The former ensures the persistence and liveness of the blockchain, and the latter addresses the main concern in a communication application.

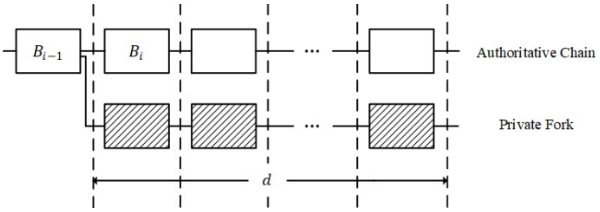

FIGURE 5. Scenario of breaking persistence, where the chain composed of rectangles with solid lines represents the authoritative chain, and the rectangles with shadows inside compose the private fork generated by the adversary. d denotes the depth of B_i to the current block number.

A. BLOCKCHAIN SECURITY

1) PERSISTENCE

Persistence means that once a block is "deep" enough in the chain, it is impossible to revert, which indicates the system's stability. We will show that the probability of a block being reverted decreases exponentially as the chain grows.

First, we present the adversary model as follows:

- The adversary does not have the capability to reverse cryptographic functions, including hash functions and digital signatures.
- The adversary cannot provide different blocks to different users.
- The adversary occupies a certain percentage of the total dedicated disk space, which is denoted by p.
- The adversary can broadcast a valid block or none when selected as a block proposer.

We also assume that honest users control the majority of the dedicated space, which indicates that $p < 0.5$. Now, we prove that the probability of the adversary being selected as a block proposer is no more than p.

Assuming the adversary's dedicated space is spread among n addresses, the corresponding percentages are denoted as p_1, p_2, ..., and p_n. Obviously, we have

$$p = p_1 + p_2 + \cdots + p_n.$$

Then, the probability of being selected is computed as

$$\begin{aligned} P &= 1 - (1-p_1)(1-p_2)\cdots(1-p_n) \\ &= \sum\nolimits_{1\le i\le n} p_i - \sum\nolimits_{t=2}^{n} (-1)^t \gamma_t \\ &\le p - \sum\nolimits_{t=1}^{(n-1)/2} (\gamma_{2t} - \gamma_{2t+1}) \\ &\le p, \end{aligned}$$

where $\gamma_k = \sum_{1\le i_1<\cdots<i_k\le n} p_{i_1}\cdots p_{i_k}$, $\gamma_{2t} - \gamma_{2t+1} > 0$.

We remark that $P = p$ when the adversary's dedicated space is in a single address.

Second, we describe the scenario where the adversary breaks persistence by generating a private fork and releasing it to revert the authoritative chain. Specifically, the adversary attempts to revert block B_i by maintaining a longer private fork linked to B_{i-1}, as shown in Figure 5.

Third, we calculate the probability of the adversary succeeding in breaking persistence. B_i can be reverted in the following two cases:

- The private fork is longer than the authoritative chain.
- The private fork has the same length as the authoritative chain but more weight.

Both cases require the adversary to be selected as proposers for at least d adjacent blocks since B_{i-1}. Therefore, the probability of B_i being reverted is

$$P_r < p^d + p^{d+1} < 1/2^{d-1}.$$

Obviously, it decreases exponentially as the depth of B_i increases. We conclude that a block that is deep enough is almost impossible to revert.

2) LIVENESS

Liveness means that a valid transaction can always be recorded in the blockchain, which requires that there is at least one block proposer for a block of fixed numbers.

In our protocol, the VRF, which works like a random oracle, is used in leader selection to determine block proposers. As each user executes the VRF independently, there is a risk that no valid proposer is selected with fixed inputs. To tackle this problem, we introduce the counter as a part of the VRF's input, which increases at regular intervals. Thus, the user can test whether he is selected for some block of a fixed block number at regular intervals, which ensures that there are always block proposers at any height of the chain. We conclude that the blockchain achieves liveness.

B. SPAM RESISTANCE

Spam is a common communication system attack that introduces unsolicited messages in the network, which is a main consideration in our protocol design. Specifically, we have to prevent the spammer from flooding the network with registration or communication transactions.

- Regarding registration transactions, the spammer has to perform a huge number of hash operations to complete proof-of-work.
- Regarding communication transactions, the spammer has to dedicate a large amount of disk space according to the size and lifetime of the delivered messages.

Thus, it costs a tremendous amount of computing or storage resources to send limited transactions in the network, which protects our protocol from spam.

V. CONCLUSION

In this paper, we focus on the design of a communication protocol based on blockchain that meets the urgent need for a decentralized communication platform with anonymity and privacy to prevent accidental information leakage and a single point of failure. After intensive research into related protocols, we present Bitmessage Plus, a practical blockchain-based communication protocol with a novel anti-spam mechanism. Compared with previous approaches, our protocol achieves high reliability, practicality, anonymity and efficiency. We have not implemented Bitmessage Plus in practice, which we leave as future work.

REFERENCES

[1] A. Harris. (2006). *Spy Agency Sought U.S. Call Records Before 9/11, Lawyers Say*. [Online]. Available: http://www.bloomberg.com/apps/news?pid=newsarchive&sid=abIV0cO64zJE

[2] J. Bamford. (2012). *The NSA is Building the Country's Biggest Spy Center (Watch What You Say)*. [Online]. Available: http://www.wired.com/threatlevel/2012/03/ff_nsadatacenter/all/1

[3] H. Tewari and E. O. Nuallain, "Netcoin: A traceable P2P electronic cash system," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, New York, NY, USA, Jun. 2015, pp. 472–478.

[4] G. Zyskind and O. Nathan, "Decentralizing privacy: Using blockchain to protect personal data," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, San Jose, CA, USA, May 2015, pp. 180–184.

[5] J. Warren. (2012). *Bitmessage: A Peer-to-Peer Message Authentication and Delivery System*. [Online]. Available: https://bitmessage.org/bitmessage.pdf

[6] A. Schaub and D. Rossi, "Design and analysis of an improved bitmessage anti-spam mechanism," in *Proc. IEEE Int. Conf. Peer Peer Comput. (P2P)*, Boston, MA, USA, Sep. 2015, pp. 1–5.

[7] J. Karamacoski, N. Paunkoska, N. Marina, and M. Punceva, "Blockchain for reliable and secure distributed communication channel," in *Proc. IEEE Int. Conf. Ind, Artif. Intell., Commun. Technol. (IAICT)*, Bali, Indonesia, Jul. 2019, pp. 91–97.

[8] R. A. Saritekin, E. Karabacak, Z. Durgay, and E. Karaarslan, "Blockchain based secure communication application proposal: Cryptouch," in *Proc. 6th Int. Symp. Digit. Forensic Secur. (ISDFS)*, Antalya, Turkey, Mar. 2018, pp. 1–5.

[9] J. Benet, "IPFS–content addressed, versioned, P2P file system," 2014, *arXiv:1407.3561*. [Online]. Available: http://arxiv.org/abs/1407.3561

[10] ByteCoin. *Untraceable Transactions Which Can Contain a Secure Message Are Inevitable*. BitcoinForum. Accessed: Oct. 15, 2020. [Online]. Available: https://bitcointalk.org/index.p hp?topic=5965.0

[11] N. T. Courtois and R. Mercer, "Stealth address and key management techniques in blockchain systems," in *Proc. 3rd Int. Conf. Inf. Syst. Secur. Privacy*, Porto, Portugal, 2017, pp. 559–566.

[12] M. O. Rabin, "Transaction protection by beacons," *J. Comput. Syst. Sci.*, vol. 27, no. 2, pp. 256–267, Oct. 1983, doi: 10.1016/0022-0000(83)90042-9.

[13] I. Cascudo and B. David, "SCRAPE: Scalable randomness attested by public entities," in *Proc. 15th Int. Conf. Appl. Cryptograph. Netw. Secur. (ACNS)*, Kanazawa, Japan, Jul. 2017, pp. 537–556.

[14] E. Syta, P. Jovanovic, E. K. Kogias, N. Gailly, L. Gasser, I. Khoffi, M. J. Fischer, and B. Ford, "Scalable bias-resistant distributed randomness," in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Jose, CA, USA, May 2017, pp. 444–460.

[15] Z. Guo, L. Shi, and M. Xu, "SecRand: A secure distributed randomness generation protocol with high practicality and scalability," *IEEE Access*, vol. 8, pp. 203917–203929, 2020, doi: 10.1109/ACCESS.2020.3036698.

[16] S. Dziembowski, S. Faust, V. Kolmogorov, and K. Pietrzak, "Proofs of space," in *Proc. 35th Annu. Int. Cryptol. Conf.*, Santa Barbara, CA, USA, Aug. 2015, pp. 585–605.

[17] L. Ren and S. Devadas, "Proof of space from stacked expanders," in *Proc. 14th Annu. Int. Conf. Theory Cryptogr. (TCC)*, Beijing, China, Oct. 2016, pp. 262–285.

[18] S. Park, A. Kwon, G. Fuchsbauer, and P. Gaži, "SpaceMint: A cryptocurrency based on proofs of space," in *Proc. 22nd Annu. Int. Conf. Fin. Cryptograph. Data Securi. (FC)*, Nieuwpoort, Curaçao, Feb. 2018, pp. 480–499.

[19] Bitcoin Wiki. *Coinbase*. Accessed: Dec. 27, 2020. [Online]. Available: https://en.bitcoin.it/wiki/Coinbase

[20] W. Feller, *An Introduction to Probability and its Applications*, 2nd ed., vol. 1. New York, NY, USA: Wiley, 1957.

[21] S. Haber and W. S. Stornetta, "Secure names for bit-strings," in *Proc. 4th ACM Conf. Comput. Commun. Secur. (CCS)*, Zurich, Switzerland, Apr.1997, pp. 28–35.

[22] C. Dwork, A. Goldberg, and M. Naor, "On memory-bound functions for fighting spam," in *Proc. 23rd Annu. Int. Cryptol. Conf.*, Santa Barbara, CA, USA, Aug. 2003, pp. 426–444.

[23] A. Biryukov and D. Khovratovich, ‘‘Equihash: Asymmetric proof-of-work based on the generalized birthday problem,’’ presented at the Netw. Distrib. Syst. Secur. Symp., San Diego, CA, USA, Feb. 2016, doi: 10.14722/ndss.2016.23108.

[24] T. Hanke, ‘‘AsicBoost—A speedup for bitcoin mining,’’ 2016, *arXiv:1604.00575*. [Online]. Available: http://arxiv.org/abs/1604.00575

[25] S. Micali, M. Rabin, and S. Vadhan, ‘‘Verifiable random functions,’’ in *Proc. 40th Annu. Symp. Found. Comput. Sci. (FOCS)*, New York, NY, USA, Oct. 1999, pp. 120–130.

LIUCHENG SHI received the B.S. degree in information and computing sciences from Peking University, China, in 2012, where he is currently pursuing the Ph.D. degree in applied mathematics. His current research interests include blockchain technology and public key cryptography and applied cryptography.

ZHAOZHONG GUO received the B.S. degree in information and computing sciences from Peking University, China, in 2012, where he is currently pursuing the Ph.D. degree in applied mathematics. His current research interests include blockchain technology and public key cryptography and multiparty computation.

MAOZHI XU received the B.S. degree from Huaibei Normal University, China, in 1983, the M.S. degree from Wuhan University, China, in 1987, and the Ph.D. degree from Peking University, China, in 1994, all in mathematics. He is currently a Professor with Peking University.