**A Stable-aware Blockchain Consensus Protocol for Wireless Blockchain Systems**

**Abstract-** Blockchain can solve security and trust challenges in wireless networks. Most previous studies of blockchain consensus protocols in wireless network rely on efficient and stable transmissions and correct leaders. Nevertheless, nodes in wireless blockchain have limited physical resources, unreliable channels, and varying bandwidths influenced jamming attacks or environments. In this paper, we propose a novel Byzantine fault-tolerant consensus protocol SWIB (**S**tability-aware **Wi**reless **B**lockchain) for blockchain in wireless networks, which do not rely on reliable communications. SWIB selects a block proposer randomly to prevent adversary corrupting the block proposer, and uses a threshold signature scheme as block proposal voting mechanism to improve the performance of blockchain system. Because only one block will be confirmed in per round, SWIB protocol can avoid the occurrence of conflicting blocks and blockchain forks. Moreover, it can guarantee security while tolerating at most faulty nodes among consensus nodes. Extensive simulation results show that SWIB is resistant to jamming attack, double-spending attack, and Sybil attack.

# Introduction

With the rapid development of wireless communication and blockchain technologies, much work has been carried out to apply blockchains in wireless applications, such as mobile edge computing [1], intelligent 5G technology [2], the Internet of Vehicles [3], etc. Applications for wireless networks face with significant challenge of security and trust. Resources constrained mobile devices are vulnerable to various attacks. Meanwhile, the open communication of wireless networks is heavily impacted by environment. Both channel bandwidth and jamming attacks make the communication in wireless networks become unstable and unreliable. Blockchain has received great attention from both academia and industry. With salient properties of decentralization and persistence as well as traceability, blockchain provides a new way to solve security and trust problems. This means that blockchain technology can provide reliable and secure resource sharing services in wireless networks. In this case, secure, trust, and efficient services of data interaction, secure access control, data traceability, identity authentication in wireless field can be supported by integration of blockchain and wireless networks.

Currently, much work on wireless blockchain systems is to directly enable popular blockchain protocols in the global Internet to wireless network environment. Consensus protocols adopted by these blockchain protocols usually require massive resources consumption (e.g., Proof of Work [4]) and complicated design (e.g., Proof of Stake [5]), or rely on reliable communications (e.g., Practical Byzantine Fault Tolerant [6]). However, the limitation of wireless network makes these classical blockchain consensus protocols difficult to be deployed in wireless networks. This motivates research on design of blockchain protocols for wireless networks.

Recently, some scholars proposed blockchain consensus protocols for wireless networks. Considering the high dynamics of mobile ad-hoc network, Jiao et al. [7] designed a PoW-based stability-aware consensus protocol, whose leader election is based on node information and Proof of Work. This novel design can make sure wireless blockchain system work efficiently and steadily. But nodes running this consensus protocol require to consume massive resources, which is a heavy burden for wireless devices. In order to decrease computation power consumption, some research on wireless blockchain systems leverage the characteristics of wireless networks to design efficient wireless blockchain consensus protocols. Considering the low-power wireless devices and instability of wireless transmission, Xu et al. [9] proposed an efficient and fair Proof-of-Communication consensus protocol for wireless blockchain systems. Zou et al. [10] proposed a fast consensus protocol for permissioned wireless blockchain systems, which can achieve *k*-times consensus in unreliable and multi-access wireless environment. Besides, to overcome the interference of wireless broadcast communication, Xu et al. proposed a single-hop wireless blockchain consensus protocol under an adversarial SINR model based on a Proof-of-Channel consensus algorithm [11]. In [12], Xu et al. designed a fast fault-tolerant protocol for multi-hop wireless blockchain network wChain. To accelerate data aggregation, this protocol constructs communication spanner by maximized independent set. However, these consensus protocol cannot tolerate Byzantine failure, which is the common phenomena in wireless networks. Leveraging the transmit signal of wireless networks, Jiang et al. [8] proposed a Sybil-proof-based Byzantine fault-tolerant consensus protocol, which can achieve real-time consensus in wireless networks by selected group of nodes. This consensus protocol requires quadratic message reliable exchange to achieve consensus. Actually, all these consensus protocols work under assumption of reliable message transmission, and not consider the impact of message loss for consensus process in wireless networks.

In this paper, we propose a stability-aware wireless blockchain consensus protocol SWIB. This protocol is analogous to the Proof-of-Stake consensus protocol in the way that nodes can achieve consensus without consuming massive resources for mining. SWIB adopts a randomized election scheme and a secure threshold signature scheme to ensure that all nodes in the wireless blockchain system can reach consensus in a random and steady manner. It operates in a round by round fashion. Each consensus round contains block proposer election phase, block proposal generation phase, block verification phase, and block finalization phase. A single block proposer is randomly and non-interactively selected according to nodes' probability in block proposer election phase. This probability depends on nodes' stability, which is a function of nodes' active time and the number of recently generated blocks. . The unpredictability of leader election can reduce the corruption risk of adversary. In addition, we adopt the threshold BLS (Boneh-Lynn-Shacham) signature scheme to improve the efficiency of consensus reaching in blockchain system. Using such a scheme can greatly reduce the system communication overhead of consensus process. Decoupling block proposer from block verification and finalization can increase the stability of consensus termination. Consensus termination can be achieved by any node who has obtained sufficient votes, without relying on correctness of block proposer. Thus, our protocol can improve the stability and efficiency of consensus process, and also reduce the risk of faulty consensus caused by faulty nodes or unstable wireless channels. Our contributions in this paper are summarized as follows:

* We propose a new blockchain consensus protocol SWIB, which combines random election algorithm with threshold BLS signature scheme. It can ensure stable generation of blocks in wireless blockchain systems and reaching consensus in unreliable and unstable one-hop wireless networks.
* We propose a random block proposer election algorithm, which is suitable for wireless networks. Consequently, a high-quality nodes can be elected as block proposer in a randomized way, which can reduce the corruption risk of adversary and improve the chance of generating valid block.
* To improve the robustness of SWIB, we use threshold BLS signature scheme as voting mechanism to improve the efficiency and stability of consensus process. Even the block proposer fails after broadcasting a block, block finalization can still be completed through one round of partial signatures exchanges. Moreover, our protocol design satisfies strong consistency that can efficiently prevent blockchain forks.
* We analyze the consensus success probability and expected consensus latency of SWIB protocol, and discuss the consensus security and attack resistance of SWIB when adversary controls less than 50% of voting power. Finally, extensive simulation results validate the correctness of our theoretical analytical results.

The rest of this paper is organized as follows. Section 2 introduces related work on state-of-the-art blockchain protocols, wireless consensus algorithms. In Section 3, we introduce various models and assumptions in this study. In Section 4, we present the details of the stability-aware wireless blockchain consensus protocol. Security and performance analyses are conducted in Section 5. Extensive simulation results are presented in Section 6 for performance evaluation, and conclusion is given in Section 7.

# Related Work

In this section, we give a brief review of existing work for consensus protocols in blockchain. We will first introduce state-of-the-art blockchain consensus protocols and then consensus protocols for wireless networks.

## 2.1 Blockchain Consensus Protocols

In this subsection, we briefly introduce popular blockchain consensus protocols, and divide into two categories: resource-proof-based and communication-based. For a comprehensive overview in this aspect, please refer to [13].

Resource-proof-based consensus protocols require consensus nodes to compete for the block proposal right in each round using their physical resources (e.g., computational power, memory, etc.) or virtual resources (e.g., shares, reputation, wealth, etc.). The most classic proof-of-physical-resources consensus protocol is Proof-of-Work (PoW) [4], which is adopted in Bitcoin and Ethereum. In PoW consensus algorithm, nodes win the block proposal opportunity by solving a computational puzzle. However, this protocol cannot provide instant consensus finality of blockchain protocol [14]. Actually, even though adversary controls computing power is less than 50% of total power, multi-blocks confirmation can only guarantee probabilistic consistency of PoW-based consensus protocol. Due to the large time-varying of generate a block, the block-confirmation latency of PoW-based blockchain protocol is in general large, and transaction throughput is limited. In order to improve the performance of blockchain system, some other physical-resource-proof-based consensus protocols have been proposed. For example, consensus nodes running Proof-of-Space consensus protocol [15] compete for block proposal right through occupied memory or disk space, and achieve consensus within about 4 minutes. Proof-of-Burn consensus protocol [16] belongs to the category of physical-resources-proof-based consensus protocols. Consensus nodes obtain block proposal chance by burning another “coin” such as Bitcoin.

Physical-resource-proof-based consensus protocols consume huge physical resources of consensus nodes to pursue the block proposal chance, which leads to massive waste of physical resources. As an alternative, virtual-resource-proof-based consensus protocols can avoid such large resource overhead. Proof-of-Stake (PoS) [5] is a typical consensus protocol in this aspect. In PoS, consensus nodes are elected as block proposer according to their held stakes. The more stakes a node holds, the higher the probability it is elected. The first version of Casper [17] is a hybrid consensus of PoW and PoS, aiming to replace the PoW consensus algorithm with PoS consensus algorithm in Ethereum. In order to solve the “Nothing at stake” problem of PoS and improve the performance of blockchain system, some consensus protocol have been proposed. Such as Proof-of-Authority consensus protocol [], in which consensus nodes passing a preliminary authentication aware the right to generate new blocks. Proof-of-Reputation [18] is also a virtual-resource-proof-based consensus protocol, in which consensus nodes with enough reputation can obtain the right to generate a new block. Virtual-resource-proof-based consensus protocols do not consume physical resources, and are friendly to environment. Therefore, more and more blockchain systems pursue to use proof-of-virtual-resource consensus protocols.

In communication-based blockchain consensus protocols, all consensus nodes reach consensus on block proposal by exchanging messages and performing local computation. Most of these protocols can tolerate Byzantine failure and provide strong consistency. The most classical communication-based consensus protocol is practical Byzantine fault tolerant consensus protocol (PBFT). In PBFT, a block proposer is elected from all consensus nodes to propose a new block, which will then be responsible for driving communication of vote phase. Because consensus nodes are not required to mortgage assets or consume resource, adversary can be malicious in consensus process with small cost. PBFT consensus protocol eliminates the influence of malicious behavior to ensure blockchain system security by cooperation mechanism. Since the communications of nodes are driven by leader/primary, the consensus security of PBFT depends on the correctness of leader. Faulty leader can lead to the interruption of consensus process. The PBFT consensus protocol achieve consensus among consensus nodes with quadratic message complexity. Thus, PBFT consensus protocol usually has low scalability due to high communication overhead. To overcome the shortage of PBFT consensus protocol, some communication-based protocols are proposed to improve performance by reducing communication overhead. In NEO [20], partial nodes of delegated Byzantine fault tolerant consensus protocol are delegated to participant consensus process to reach agreement by voting on generated block. Reducing the number of consensus nodes can decrease the number of communication messages, and further increase the performance of system. Zyzzyva [21] utilizes a modified BFT consensus protocol SBFT. The consensus protocol employs a threshold signature scheme to reduce communication overhead of consensus process. Since consensus latency decreases, the performance of system will be improved.

Hybrid consensus protocols can alleviate the “impossible triangle” of blockchain systems. Single consensus protocol cannot satisfy security, decentralization and scalability of blockchain systems simultaneously. Physical-resource-proof-based consensus protocols are highly decentralized and secure, but its scalability is low. Virtual-resource-proof-based consensus protocols are highly decentralized and scalable, but exist serious security problems. Communication-based consensus protocols have high scalability and security, but decentralization is low. Therefore, hybrid consensus protocols have been proposed to balance the “impossible triangle” of blockchain system. The first version of Casper [17] is a hybrid consensus of PoW and PoS, aiming to replace the PoW consensus algorithm with PoS consensus algorithm in Ethereum. This consensus protocol improves the security of blockchain system by reducing the risk of chain fork. Fruitchain [] uses a hybrid consensus protocol of PoW and DPoS consensus algorithms. This protocol can achieve high performance and low fork probability. In order to prevent chain fork and achieve high performance, some hybrid consensus protocols combining resource-proof-based and BFT consensus algorithms have been proposed. ByzCoin [] improves transaction throughput by adopting a hybrid consensus protocol of PoW and BFT consensus algorithms. This protocol can achieve strong consistency of blockchain system. These consensus protocols are not suitable for wireless networks with limited resources devices due to high computation overhead. To decrease the waste of resources, Tendermint [22] uses a hybrid consensus protocol combining PoS and BFT consensus algorithms. This protocol adopts leader rotation mechanism to avoid adversary corruption, and achieve high security and performance of system by BFT consensus algorithm. Besides, Algorand consensus protocol [23] is a hybrid consensus protocol of PoS and Byzantine agreement algorithms. This protocol uses VRF-based leader and committee election algorithm to prevent adversary knowing the information of leader and committee in advance. Byzantine agreement protocol can make sure high performance and strong consistency of system. However, these consensus protocols adopting BFT consensus algorithms rely on reliable communication model. Thus, these protocols cannot be directly utilized in wireless scenarios with unreliable and unstable communication channels. Our protocol is a type of hybrid consensus protocol combining PoS and communication-based consensus algorithms. This protocol uses PoS consensus algorithm to ensure the fairness of consensus process, and utilize a special communication-based consensus algorithm to improve the performance and security of system.

## 2.2 Consensus Protocols for Wireless Networks

Since our study is closely related to consensus protocols and wireless networks, we briefly introduce the studies on consensus protocols for wireless networks in this subsection.

There has been extensive prior research on consensus protocols for wireless networks. Santoro and Widmayer [] study consensus in the presence of unreliable communication, and show that consensus cannot be achieved if as few as of the messages sent in a round can be lost.

Many consensus protocols have been proposed to achieve consensus in wireless networks. Leveraging the natural superposition property of wireless multiple-access channels, Zheng et al. [24] proposed fast average consensus in clustered wireless sensor networks to achieve consensus within short time. In order to efficiently achieve global consensus among nodes in clustered wireless networks, Goldenbaum et al. [25] presented an iterative gossip algorithm based on the superposition property of wireless channel. Newport and Robinson [26] proposed fault-tolerant distributed consensus algorithms to solve consensus problem of wireless systems through abstract MAC layer model. These consensus algorithms can guarantee consensus termination with high probability even facing any number of failures and no network information in advance. To achieve finite-time max-consensus in a multi-agent system, Molinari et al. [27] presented a switching consensus protocol according to the superposition property of fading wireless channel. Max-consensus can be achieved under this protocol within finite number of iterations. Besides, Moniz et al. [28] proposed an asynchronous Byzantine consensus protocol for resource-constrained wireless ad hoc networks. Even some messages are lost dynamically, the protocol can still achieve consensus efficiently.

In recent years, some studies combined wireless consensus algorithms with blockchain to design blockchain consensus protocols for wireless networks. According to the intristic property of wireless broadcast communication, Jiang et al. [8] proposed a Sybil-proof-based Byzantine fault-tolerant consensus protocol, which can realize real-time consensus in wireless networks. Some studies increase consensus efficiency by making full use of wireless broadcast operations. Xu et al. [9] proposed an efficient and fair Proof-of-Communication consensus protocol in wireless blockchain systems. Zou et al. [10] proposed a fast consensus protocol that can achieve *k*-times consensus in unreliable and multi-access wireless environment. Besides, to overcome the interference of wireless broadcast communication, Xu et al. proposed BLOWN that based on a Proof-of-Channel consensus algorithm under adversarial SINR model [11]. In addition, Xu et al. designed a fast fault-tolerant wireless blockchain protocol wChain [12], which can quickly aggregate data and reach consensus in multi-hop wireless communication networks.

这里应该总结一下本文工作与上述工作的主要差别…否则读了半天，毫无感觉.

# Models and Assumptions

SWIB adopting threshold signature technology allows nodes to agree on block in blockchain system under wireless networks with unreliable channels efficiently. We describe the basic preliminaries of SWIB, including network model, blockchain setting, threshold signature scheme and adversary assumptions.

## 3.1 Network Model

We consider a wireless broadcast network consisting of nodes, which are located within communication range of each other, and communicating with each other by transmitting messages. All nodes have same functions. Each node equipped with transceiver works in a half-duplex manner. This means that nodes can transmit or receive messages, but not both simultaneously. In practice, such a network can be formed by a group of unmanned aerial vehicles or intelligent vehicles.

Nodes running the SWIB can achieve consensus on proposals. For simplicity, we assume that the consensus processing is divided into synchronous rounds, each of which contains multiple slots. A slot is the time unit for nodes to transmit or receive a packet.

Nodes adopting digital signature technology can achieve node identity confirmation and the verification and integrity of communication messages. In digital signature, each node has its key pair, which used to message encryption and decryption, and generate signature. We assume that each node can get its private-public key pair and a main public key by independently running a secure distributed key generation protocol. Each node can obtain the public keys and identities of other nodes by exchanging messages. Thus, each node knows the identities and public keys of all other nodes.

## 3.2 Communication Model

We consider a wireless communication model with p-persistent carrier-sense multiple access (CSMA), where nodes continuously sense, and transmit message with probability when detecting channel idle. We assume that the wireless channels follow the Rayleigh fading model [32]. In detail, the wireless channel between nodes and experiences path-loss is modeled as

where is the path-loss at reference distance , and is the distance between nodes , is path-loss exponent Let be the channel gain from to , following the complex normal distribution with zero mean and variance (i.e., ). When a signal is transmitted from to with transmit power , the signal-to-noise (SNR) ratio at receiving node :

where is the additive white Gaussian noise.

Message losses can lead to consensus failure. In addition, message losses are mainly caused by channel collision and channel fading. Therefore, a successful transmission should satisfy two conditions: 1) if and only if there only one node transmitting in a time slot; and 2) the SNR ratio is equal to or bigger than the target one.

In channel contention process, nodes compete for the channel with same transmit probability . Only if one node transmits in a time slot can the node transmit message successfully. Let be a variable that denotes the number of nodes, transmission contention success probability can be expressed as

Even a node competes successfully, it may fail to transmit a message due to channel fading. The SNR between nodes varies with the time-variant communication environment. When SNR is less than a given target threshold, the communication between nodes is interrupted. Since is exponentially distributed, the SNR outrage probability between nodes and is expressed as follows

Both collision and channel fading can result in message losses. In order guarantee communication under unreliable networks, retransmission mechanism is necessary. When communication interruption occurs, retransmissions are carried out until the lost messages are successfully delivered.

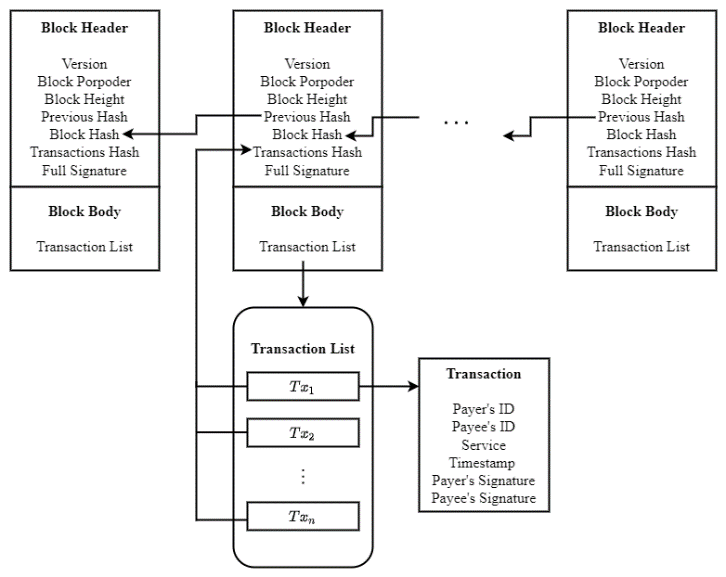
The absolute time is split into time slots of fixed time interval, which is calculated as follows:

where is the maximum size of packet sent by nodes during the consensus process in bits, is the target Signal-to-Noise ratio of a transmission , and denotes the bandwidth utilized for transmission. The communication between nodes is interrupted when SINR is below the given threshold . [这和上一句说的不一致, 前面说的是目标信噪比？这里变成了能够实现信道解码的最低门限] Since is exponentially distributed, the communication interruption probability between nodes and is expressed as follows,

Nodes communicate with other nodes within a single hop in the broadcast protocol, which means that any pair of nodes can be paired through a wireless channel. Thus, we assume that each transmit node that uses same transmission power must cover network area.

## 3.3 Blockchain Setting

We assume that each node maintains a local blockchain, which is composed of blocks chronologically linked to create a chain. Each block contains a block header and a block body. The block header records blockchain version, block proposer, block height, previous hash, block hash, block full signature, and transactions hash root, etc. The block body stores transaction recodes, each of which consists of payer's information, payee's information and other necessary contents. Figure 1 shows the data structure of a blockchain.



**Fig. 1 Blockchain**

## 3.4 Threshold Signature Scheme

We utilize a threshold Boneh-Lynn-Shacham (BLS) signature scheme [29] to improve the performance of consensus protocols. Consensus protocols can adopt the threshold BLS signature scheme as a voting mechanism to confirm the validity of proposals. A partial signature generated by a node for a proposal is the vote of the node for the proposal. Since several partial signatures can be aggregated into a special signature, nodes can verify multiple votes in an operation. In addition, the termination of a consensus process not relies on the leader of consensus system. Any consensus node can terminate a consensus process when it has constructed the special signature. Therefore, threshold BLS signature scheme can help consensus nodes quickly and steadily achieve consensus.

Threshold signature scheme [34] aims to allow multiple participants securely reconstruct a secret (i.e., a main private key) and perform computation (i.e., signature generation or decryption) even an adversary has corrupted some participants. In a threshold signature scheme, all participants use a share of a main private key as their private keys. To sign a message, multiple participants can execute an interactive signature generation algorithm, which uses their shares of the main private key and the message as inputs, and outputs a signature of the message. The signature can be verified by anyone using the unique main public key. The security for threshold signature scheme requires that no adversary that corrupts some participants can learn any information about the main private key or can forge a valid signature on a message. There are two important properties of threshold signature scheme: robustness and proactiveness. The former one requires that a valid signature can be generated even some malicious participants deviating from the scheme. The later one is also name periodic refreshment of shares of a main private key. The goal proactiveness is to protect a system from an adversary that obtains the information of a main private key.

Boneh-Lynn-Shacham (BLS) signature scheme [30] is closely related to bilinear maps and Gap Diffie-Hellman (GDH) groups. The signature is based on bilinear maps on elliptic curves. In GDH groups, Decision Diffie-Hellman problem is easy on certain elliptic-curves groups, but Computational Diffie-Hellman problem is hard. The BLS signature scheme based on GDH groups consists of key generation algorithm, signature generation algorithm and signature verification algorithm. Let be a GDH group pair of prime order , and be the generator of and , respectively. The signature scheme makes use of a secure hash function . In key generation, each participant has a key pair , where is private key that used to compute public key . Given a private key and a message signer can compute a digest and a signature Given a public key , a message and a signature , verifiers can compute and verify whether the Diffie-Hellman tuple is a valid.

Threshold BLS signature scheme [29] is the combination of threshold signature scheme and BLS signature scheme. The signature share generation of threshold BLS signature scheme concurs with the BLS, and the aggregation of group signature is congruent with threshold signature scheme. Threshold BLS signature scheme includes key generation algorithm, signature generation algorithm and verification algorithm. The key generation algorithm adopts a discrete log-based distributed key generation method [31] to distribute private-public key pair and a main public key to participants. Let be a GDH group pair of prime order , and be the generator of and , respectively. The signature scheme makes use of a secure hash function . The key generation algorithm outputs a private-public key pair for participant and a main public key , where is a share of a main private key and the public key is calculated as . The signature generation algorithm contains a partial signature generation method and a full signature recovery method. The former one is similar to the signature generation algorithm of BLS signature scheme. Given a message a signer running signature share generation method can compute a digest and a partial signature The full signature of the message can be recovered if participants aggregate a sufficient number of partial signature and execute the full signature recovery method. The full signature should be the Lagrange interpolation polynomial of these partial signatures, i.e., . The verification algorithm of threshold BLS signature scheme uses the main public key to verify the validation of the full signature. Given a main public key , a message and the full signature , verifiers can compute and verify whether the Diffie-Hellman tuple is a valid.

## 3.5 Adversary

* We consider a threat model that an arbitrary adversary controlling no more than of the total voting power can corrupt nodes to be arbitrarily malicious. The malicious behaviors of adversary are as followsAdversary can issue Sybil attack, in which adversary can generate a large number of node identities for a node. These malicious nodes controlled by the adversary will generate invalid blocks or not generate block when elected as block proposer, or transmit faulty messages in block voting phase to interrupt the consensus process of a valid block;
* Adversary can launch jamming attack to interfere with the message transmission in wireless networks. Without loss of generality, we assume that the capability of adversary is - bounded, i.e., in any interval of consecutive slots, adversary can jam no more than slots, where and 0 < ≤ 1.

Table 1 lists major notations used later.



# The Stable Wireless Blockchain Consensus Protocol

In this section, we propose the stable wireless blockchain consensus protocol, named SWIB (**S**table **Wi**reless **B**lockchain). We first give an overview architecture of the protocol, and then present a detailed protocol design of SWIB.

## 4.1 Architecture Overview

This subsection introduces the preliminary design of SWIB protocol. Next, we introduce how one round of block generation in SWIB works and how the consensus process transitions from round to round .



**Fig..** An overview of how SWIB works for a round .At the beginning of the round, all consensus nodes are assumed to maintain a same replica of the blockchain. (1) Block proposer election algorithm is executed to randomly determine the block proposer for the current round; (2) verify legality of the new block, and run signature generation algorithm to vote for valid block; (3) run signature aggregation algorithm and signature recovery algorithm to finalize the block when generating the full signature.

SWIB works in a round by round fashion. Fig. 2 shows how SWIB works in a round *r*. It starts with a secure randomness generation process for block proposer election. It is in general not secure to allow nodes to predict who will be the block proposer in the next round. Thus, a randomness source is needed to ensure that no node can know who will be the next block proposer in advance. In SWIB, consensus nodes can join the blockchain system by submitting Sybil-resistant-proof. Each node registers in the blockchain system by depositing certain amount of money, which will be stored in a virtual account. Only the depositor executing unpledged operation can take out the money. The protocol adopts a distributed randomness generation scheme, which ensures that all nodes can generate a same randomness per round independently. According to the round randomness, a block proposer will be elected at the beginning of each round through the block proposer election algorithm. Then, the elected block proposer will generate a block and broadcast it to other nodes. Each node will vote on the validity of the block by generating partial signature through a partial signature generation method. Once aggregating a threshold of partial signature shares, any consensus node can recover the full signature to finalize the block through a signature recovery method.

We aim to propose a protocol such that hundreds of nodes can achieve consensus in a wireless network with unreliable and unstable channels.

The consensus process contains four design components as given in the following:

* Block proposer election: At the beginning of each round, a random number is independently generated by nodes via a distributed randomness generation scheme. Each of the nodes checks whether it becomes the block proposer for the current round through verifiable block proposer election algorithm, which uses node's private key and the round random number as inputs.
* Block Proposal: The elected block proposer will pack transactions from its transaction pool to generate a new block, and disseminate it to other consensus nodes in the wireless network.
* Block validation: Upon receipt of the proposed block, nodes will verify validity of the block. Each node will generate a partial signature of block hash if the result of the verification is true. Then, they will broadcast their partial signature shares to other nodes.
* Block finalization: Full signature can be recovered if a node receives and then aggregates enough partial signature shares. Then, the full signature will be broadcasted to all nodes. The full signature can be seen as the proof of block finalization. The conditions of block finalization are: 1) collect enough partial signature shares and recover the full signature; 2) receive the valid full signature of block hash. Upon receipt or generation of a full signature, nodes will append the corresponding block into their local blockchain. After that, nodes will generate a new random value for the next round through distributed randomness generation scheme.



Algorithm 1 shows the process of a complete round in SWIB protocol. Before starting a new round, consensus nodes perform transaction broadcast. Note that the pseudo codes of broadcast operation in blockchain network presented in Algorithm 2, whose parameters are utilized to ensure jamming-resistant communications. The details of every stage will be given in the following subsections. We solve the challenges of block proposer election and block verification as well as finalization in consensus protocol to ensure the security and stability of wireless blockchain system. Even adversary occupies some nodes, SWIB can still work smoothly in a wireless blockchain system.

【你给我的所有算法代码、表，都不应该是个图片，而应该是可编辑的版本，你咋就能保证代码和表述没问题呢？…你现在这么整，代码我完全没法看，因为看了也改不了…欲速则不达】



## 4.2 The SWIB Protocol

Before participating consensus, each node first generates its key pairs and the main public key through a distributed key generation algorithm. Besides, nodes should also request the identities and public keys of other consensus nodes as well as necessary blockchain history information. SWIB protocol contains four design components: block proposer election, block generation, block verification, and block finalization. In the following, we shall present the detailed design of each of these components.

### 4.2.1 Block Proposer Election and Block Generation

[叙述技术性内容的时候，除了保证其可行性和层次逻辑严谨性，还需要考虑如何突出自己的位置，如何体现出哪些是自己干的、新意何在]

Here, we present a verifiable block proposer election algorithm, which combines distributed randomness generation scheme and threshold BLS signature scheme. Algorithm 1 shows the procedures for the block proposer election and block generation (see lines 10-14 therein).

The block proposer election algorithm adopts a random seed to guarantee the security of election process. The distributed randomness generation scheme can enable all consensus nodes to jointly generate a round-related randomness in an unbiased and unpredictable manner. The inputs for the selection in round *r* include the round number , block hash , and the full signature of the previous round . Using the full signature as randomness source can ensure the uniqueness and immediacy of the inputs. As shown in Fig. 3, the randomness is performed as the normalized hash value of the above input combination:

For simplicity, the very first randomness is set to be the normalized hash value of the genesis block of blockchain.



**Fig. 3. Randomness generation in each round.**

The output of the distributed randomness generation scheme is unpredictable and unique. The recovery process of full signature for each round is unpreventable, provided that majority of the consensus nodes are correct as we assume. The full signature can be generated by correct node(s) or received by other consensus nodes with high probability in finite time. Although the block hash and round number are known in advance, the full signature can only be revealed at the end of round. Thus, no one can predict the output of distributed randomness generation scheme in advance. The full signature cannot be tampered due to the security of threshold signature scheme, even a node first recovers full signature before others. Another benefit of the distributed randomness generation scheme is that nodes can enter the following round non-interactively. Upon receiving or generating a valid full signature, consensus nodes will append the corresponding block into their local blockchain. Then, each node will start a new consensus process by computing the following round randomness.



**Fig. 4. The block proposer election for round . [图中所有变量都应斜体显示，该讲究的地方要讲究， *i*代表啥？为啥用i-1, 且正好在N-3上面，BP前面给缩写了么？每个变量和缩写都要在第一次出现时定义]**

After execution of the distributed key generation scheme, each node obtains a private-public key pair , to sign and verify messages. The public key is known by all nodes in the system. All nodes will exchange their public keys before entering the consensus process. The public key list can be seen as the identities of nodes. To ensure a same view on the node list, the list is assumed to be sorted according to the hash values of public keys. As illustrated in Fig. 4, all participants have a same view of the public keys list.

The block proposer for each round is elected according to nodes' elected probabilities and the value of the round. All honest consensus nodes can obtain the same value by executing the distributed randomness scheme with the same inputs. In SWIB, the elected probability of a node is mainly determined by its stability, which is a relative concept. Let be the remaining active time of node in the system, then the sum of all consensus nodes' remaining active time will be . The active time ratio of node (denoted by ) is calculated as . The consensus ratio of node (denoted by ) is calculated as , where is the number of blocks generated by in the latest blocks on the blockchain. When the so far blockchain length is smaller than , the consensus ratio of every node is set as zero. Regarding the stability of (denoted by ), we have , where and are the weights of active time ratio and consensus ratio, respectively, and . According to nodes' stability values, we have the elected probability of each node : .

The random block proposer election algorithm ensures that the election process is randomized. To determine the block proposer for the current round, the election algorithm divides interval into consecutive intervals:. 【这里*v*代表啥？】 If , then the node whose public key ranks the *i*th [?????，总不该出现第0个吧] will be the block proposer of round . Each consensus node can independently check whether it becomes the block proposer of current round. Meanwhile, each node can also verify the legality of an elected block proposer according to the maintained node list. Besides, the block proposer election is built on top of secure and robust threshold BLS signature scheme, which can tolerate any malicious nodes among the nodes. 【这句话成立的前提是所有的消息传输都是及时可靠的】 The secure threshold of the scheme can be satisfied when majority of consensus nodes are correct. Therefore, a block proposer can be elected in each round when the full signature for its preceding round is recovered.

The node being elected as the block proposer will pack transactions to generate a new block. The header of the block can be represented by a tuple , where is the round number, is the hash value of previous block, is the current block hash, and is the identity of the block proposer, and is the root of transactions. Block body usually stores transaction metadata, which is a transaction list. The block proposer will broadcast the block and signature of the block hash to other nodes. In addition, the full signature can only be appended to this block when a node gathers enough partial signature shares and reconstructs the signature.

### 4.2.2 Block Verification and Finalization

[如何突出本文的创新内容/new design呢？别人问的是，你所的都对，但你们（作为作者）干了啥？叙述的都对，不代表你就有贡献，一个人把Windows说明书抄上一百遍，抄的再对，也没贡献…]

The block verification and finalization depend on secure threshold BLS signature scheme. The pseudo code of block verification and finalization is presented in Algorithm 1 (see lines 16-39 therein). SWIB uses three key algorithms in the threshold BLS signature scheme: a signature generation algorithm to generate partial signature; a signature recovery algorithm to reconstruct the full signature from a threshold of partial signature shares, as well as a signature verification algorithm to check the validity of both partial signature and full signature. Blocks and the signatures are sent to consensus nodes via wireless broadcast.

In the block verification phase, a node needs to check the validation of the proposed block through the following components:

* Block proposer: The result of block proposer election algorithm, which uses current round randomness as input, should be the same with the index of block proposer in the node list that it keeps.
* Previous hash : The previous hash has to equal the hash of the preceding block, which was confirmed during the previous round.
* Transactions: All transactions included in the proposed block should not conflict with previous confirmed transactions.

If all the above conditions are satisfied, then a node can generate partial signature of the block hash and further broadcast it to other nodes. As shown in Fig. 5, each node, which gathers a sufficient number of partial signature shares of the block hash, can recover the full signature. SWIB uses the full signature as the proof of block finalization. The reconstruction of valid full signature proves that a given threshold of nodes sign on the block, which means that a sufficient number of nodes vote for the block validity. Therefore, it is feasible to use the full signature as the signal of block finalization. The stability of consensus process is improved since full signature recovery can be done independently by any correct node. In this way, any node which aggerates enough partial signature shares can recover the full signature. Even if malicious nodes refuse to recover the full signature or to broadcast valid full signature to other nodes, block finalization can still be reached when there are enough honest nodes. This design enables block finalization to be steadily achieved in a wireless network with faulty nodes or unreliable communication channels. Moreover, since each correct node can only vote at most once in a round, only one block can be confirmed in a complete round. Therefore, SWIB satisfies strong consistency, which means that it can prevent blockchain forking. [这真能吗？因为这里没说消息丢失的事，只要一些节点没收到full signature而另外一些收到了，就可能分叉]



**Fig. 5. Block verification and finalization at a node . [图中的所有的j都应斜体]**

### 4.2.3 Incentive and Punishment Mechanism

We design an incentive mechanism to encourage consensus nodes to participate the consensus process. Block proposer might be reluctant in block generation due to considerable power consumption for block packing and broadcasting. Besides, both block verification and signature generation consume certain computational power at nodes. Rational nodes may be willing to wait for receiving full signature from others, rather than consuming their computational power to verify a block, generate partial signature, and recover full signature. Therefore, an incentive mechanism is needed to motivate nodes to participate the consensus process actively. The rewards in blockchain system are the submitted transaction fees in valid blocks. In our incentive mechanism, part of the transaction fees is rewarded to block proposer, and the rest will be averagely distributed among those nodes whose partial signature shares are used in recovering the full signature.

Part of the transaction fees is distributed to block proposers, which encourage them to generate valid blocks containing as much transactions as possible.

The rest fees will be shared among those nodes whose partial signature are used to recover the full signature that appended into the valid block in block finalization. Because the uniqueness of threshold BLS signature scheme, any different partial signature shares can recover the same full signature. Since the full signature for a block may be recovered by more than two distinct sets of partial signature shares due to message losses. In this case, the rest fees will be shared by the set of nodes who have the smallest average timestamps of partial signature shares used to recover the full signature. In other words, we reward those consensus nodes signed the block hash as quickly as possible. This incentive mechanism not only encourages nodes to verify block and generate partial signature, but also motivates nodes to broadcast partial signature and full signature. The earlier the signature broadcasting, the higher the chance being rewarded. [当存在部分消息丢失的情况下，有可能一部分节点认可某个recovered full signature，而另一部分节点认可另外一个，这样会产生分裂，这怎么解决？怎么分钱？ ]

In addition, we adopt a punishment mechanism to reduce the probability that rational nodes become malicious. The malicious behaviors of nodes contain the following: 1) when being elected as the block proposer for a round, a node generates an invalid block for that round or does not generate any block before timeout; 2) a node sends invalid signature or garbage messages in block verification and finalization phase. Irrational nodes may harm others without benefiting themselves. A punishment mechanism is necessary to suppress such malicious behaviors. In SWIB, the active time determines how long a node will work continuously in the blockchain system. Hence, reducing the active time of a node is a good measure to punish a node behaving maliciously.

Reducing the active time of malicious nodes can increase the security and performance of blockchain system. This is because reducing the active time of a node will decrease its stability value, which further reduce its probability being elected. In this way, there will be reduced chance to finalize empty blocks due to malicious node behaviors. The more valid blocks finalized, the higher the system performance will be. Moreover, irrational nodes will be quickly expelled from the system if they initiate malicious behaviors frequently. As a result, the security of blockchain system will be improved. Therefore, the punishment mechanism can reduce the continuous impact of malicious behaviors and encourage nodes to behave honest. This is because rational nodes prefer to follow the consensus protocol to win reward rather than become malicious. In this way, the system throughput will not be significantly decreased since honest block proposers still strive to generate valid blocks.

### 4.2.4 Synchronization Mechanism

In our protocol, we consider a more efficient and secure self-initiated mechanism. When a new node joins the blockchain system, it is necessary to get information of other consensus nodes and necessary blockchain history before participating the consensus process. In SWIB, a joining node will request necessary blocks of blockchain history from nodes with high stability. Nodes with higher stability are more likely to maintain the latest blockchain, and have lower probability of sending fault blocks.

When the total number of blocks of blockchain history is , a new joining node can request different blocks from different neighbors with high stability and small distance. Let be the communication interruption probability between and , then can successfully receive a block from with probability . [前面给出的那个式子，不是block interruption probability，最多是个message interruption probability] The probability that a node *v* can receive historical blocks successfully from node *u* is . This design choice can reduce the overhead of blockchain history retrieval. The load balancing strategy used in this process can also effectively prevent a single node failure due to large energy consumption for blockchain history transmission. Meanwhile, synchronization mechanism will reduce the risk that a node transmits error blockchain history to mislead new nodes. Once a node transmits error blockchain history information, the receiving node can immediately detect it through blocks received from other nodes. In this case, the new node can request blocks from other trustworthy nodes to ensure the receipt of correct blockchain history information.

Synchronization process can also happen when a consensus node receives a partial or full signature without receiving the corresponding block. In this case, the node will request the corresponding block from its neighbor with high stability and good link quality if the node itself is a honest node. [逻辑有点问题，因为不知道对方是否已经收到该block，这种request方式有些盲目；一个解决办法是，从多个收到partial/full signature的节点里，选择与自己链路质量最好的] If the previous hash of a received block does not equal the hash of the latest block on the node's local blockchain, the node will also request blockchain history from its neighbors. In this case, each received block will be appended to the node's local chain if it is absent. When the maintained chain is the latest, the node participates the consensus process via generating the round-related random number according to the full signature of latest block. This procedure ensures that blockchain system will not be stopped provided that the number of honest nodes meets the security threshold. Thus, SWIB can utilize the synchronization mechanism to guarantee the security of SWIB.

另外，如果一个节点收到了partial signature，但未收到full signature，也必须坚持request the full signature…否则这就是链分叉的源头…【这也还有问题，两种可能：1)有可能真的这轮就没有得到full signature，2)有，但自己没收到…所以我不知道对这个问题，有什么说法】

# Protocol Analysis

In this section, we will analyze the performance and security of SWIB. We analyze the performance in terms of consensus success probability and consensus latency. Besides, we analyze the security from consensus protocol security, random generation security, and attack resistance.

## Performance Analysis

In this subsection, we analyze consensus success probability and consensus latency, which is important metrics to measure the performance of a blockchain system.

### 5.1.1 Consensus Success Probability

In SWIB, block proposer is responsible for block generation. After generating a new block, the block proposer sends the new block to other consensus nodes for their validations. Consensus nodes then generate and broadcast partial signatures when they successfully verify the block. When a consensus node receives a sufficient number of partial signature shares, the block finalization is achieved by recovering a full signature. The full signature is seen as a synchronous block update message, which can be reported by any consensus node obtaining the full signature. Our protocol provides a synchronization mechanism to resolve the problem of possible blockchain inconsistency at different nodes. However, consensus process may fail if no consensus node receives the minimum required number of partial signature shares for recovering the full signature. This means no consensus node can generate/receive the full signature in the consensus round.

In our considered network model, we have the following consensus success probability of SWIB.

**Proposition 1**. For a given round, the consensus success probability of our protocol is lower bounded as

where the function is defined as, for and , and are the maximum and minimum SNR communication interruption probabilities in the wireless network, respectively.

**Proof**. In SWIB, aggregation failure occurs when a node fails to receive more than partial signature from other nodes in block verification. Let be the set of consensus nodes in a round, and be the set of nodes that fail to transmit partial signature to the receiver in that round. [the receiver到底是谁？] Denote the size of set as , i.e., . The aggregated failure probability of any node due to communication interruption is given as

where is the SNR communication interruption probability between and .

The security of SWIB does not rely on the correctness of block proposer. When block proposer is faulty, the rest honest nodes can vote for an empty block and reach consensus eventually. Thus, if all consensus nodes failed to aggregate enough partial signature shares, the consensus process in a round will be interrupted. The consensus success probability is calculated as

■

Obviously, the consensus success probability of SWIB is only related to network size and SNR-based communication interruption probability between nodes.

### 5.1.2 Consensus Latency Analysis

In SWIB, consensus latency is defined as the time interval from a block proposal to its finalization. We assume that the computing time is negligible as compared with the time for message delivery over wireless channels. Then, the consensus latency is computed as

with a consensus success probability greater than or equal to for some . This equation comes true because a single consensus round of SWIB consist of at most turns of independent message dissemination opportunities. The term is the number of time slots required for the delivery of a message. A block is composed of block header and a block body. The block header mainly stores metadata for identifying the block and takes a fixed size. Block body contains multiple transactions, which is assumed to take fixed size as well. Let be the number of bits that can be transmitted in a slot, and and respectively be block header size and transaction size. The maximum consensus latency of the SWIB without considering channel contention is expressed as

where is the number of slots required to transmit the maximum-sized block, and is the number of slots required to transmit a signature. We assume that the transmission probability of nodes is . In our considered network model, the consensus latency can be computed as follows:

**Proposition 2.** For a given consensus round, let represent the maximum number of time slots for transmitting a block, and be the number of time slots for transmitting a signature. Let denote the transmission probability of a node in a time slot. The required runtime of consensus process in SWIB is at least time slots, where is some target transmission probability.

**Proof**. [完整的信道接入行为是什么？是p坚持的信道接入么？即，别人正在传的时候，选择退避；当检测到信道空闲或别人传输结束时，以概率p发送，1-p继续等待下一个时隙，一直如此重复，直到传输出去当前消息] Based on the above definition of *p*, the probability that only one node broadcasts a message in a time slot (denoted by *ps*) is . Let be a random variable denoting the number of time slots required for a node to successfully transmit a message. Let be the minimum required number of time slots such that for reaching a target success transmit probability . Then, the required number of time slots is for target transmit probability .

For a given round, transmission of a (largest) block by the block proposer takes time slots, and transmission of a partial signature share by a consensus node takes time slots. In a block verification process, there are at most nodes transmitting partial signature shares. Moreover, one consensus node is required to transmits full signature for block finalization. Thus, without considering channel contention, the maximum time slots required for a consensus process is . Therefore, the worst-case time of a consensus process in wireless blockchain network is time slots, where is target transmission probability. ■

Next, we derive the expected consensus latency in SWIB when considering the consensus success probability and faulty nodes. Let and represent the number of time slots required to transmit a valid block and an empty block, respectively. Let be the consensus latency of a valid block in the round, and be the consensus latency of an empty block in the round, where is the secure threshold for BLS signature scheme to work properly. 【为什么这里, t就够了，如果考虑传输丢失的话, t只是最小的那个门限值吧】 and are maximum mean consensus latency of a valid block and that of an empty block, respectively. 【不太清楚有最大期望一致性延迟这一说法吗？】 Let denote the probability that elected block proposer is faulty in an *N*-node network having at most faulty nodes. According to the above consensus success probability, we can calculate the expected consensus latency as follows

where is the total number of consensus rounds. [这些加起来，是什么物理含义？]

Let and denote the number of required time slots for transmitting a signature in the round and that for transmitting a valid block, respectively. Let and be the number of time slots required for transmitting a maximum sized block and an empty block, respectively. The expected consensus latency is expressed as

where is target transmission success probability, is the transmission probability of each node, is the fraction of faulty nodes, and is the consensus success probability.

## 5.2 Security Analysis

In this subsection, we first analyze the consensus security of SWIB. Then, we analyze the random generation to prove the security of randomness generation scheme. Moreover, we analyze the attacks resistance of SWIB.

### 5.2.1 Consensus Security Analysis

We conduct a security analysis to show that SWIB protocol provides persistence and liveness to wireless blockchain system. With majority of consensus nodes being honest, our protocol satisfies:

* Persistence: If an honest node proclaims block is valid, then other honest nodes also agree on that the validity of block .
* Liveness: All honest nodes eventually commit a block, and generate the following round randomness.

SWIB can guarantee the security of consensus even if failures happen in blockchain system. Node failures can be roughly classified into crash failures and Byzantine failures. When a crash failure happens, the corresponding node will not send message(s) or response to any request. Besides, nodes may send some error messages to mislead other nodes when Byzantine failures occur. The security of our protocol relies on the secure threshold BLS signature scheme, instead of correctness of the block proposer. In SWIB, block proposer is only responsible for generating block. If the block proposer generates an invalid block or does not generate any block, all consensus nodes can securely proceed to the following round as well. There are two possible outputs at the end of a round: a finalized valid block or a finalized empty block. A valid block is confirmed if the following two condition are satisfied: 1) the block proposer honestly generates a valid block and 2) majority of correct consensus nodes vote for it. If either condition is not fulfilled, an empty block will be finalized eventually. However, block finalization requires a threshold of partial signature shares to recover the full signature. Therefore, the number of honest nodes should satisfy the requirement of threshold BLS signature scheme. Threshold signature scheme is secure, because its output is unforgeable and robust. The most important properties of the threshold signature scheme are:

* Uniqueness: Any set of more than partial signature shares can recover the full signature. That is, the recovery result of any enough partial signature shares is always the same. [问题是，这句话的前提是传输都是可靠的情况下，如果传输不可靠，即使节点发出了那么多的partial signature shares，仍然可能失败，这里对这事，黑也没提，白也没提…]
* Verifiability: The full signature can be verified by anyone using the unique main public key. The public key is generated and distributed by the distributed key generation scheme when a node requests to join the consensus process.

If there are a quorum of nodes fulfilling the requirement of threshold BLS signature scheme, our protocol can guarantee persistence:

[丢失情况下的安全性]

**Theorem 1. (Persistence)** If an honest node proclaims block is the block in the blockchain, then other honest nodes should report the same result.

**Proof.** In order to prove the persistence property, we need to show that for any two blockchains and of honest nodes and should have same block at the same position. To prove by contradiction, we assume that and are two different blocks at a same position of and as maintained by nodes and , respectively. There are two cases when the assumption holds:

Case 1. and are respectively appended to the position of blockchains and in the same round. The block finalization depends on a same unique full signature, which is recovered by at least valid partial signature shares. If two different blocks and are finalized in the same round, it indicates that the threshold signature scheme recovers two different full signatures in a round. This conflicts with the uniqueness property of threshold signature scheme, which contradicts our assumption.

Case 2. and are appended to the blockchains and respectively in two different rounds. Assuming that and are respectively appended to the position of blockchains and at rounds and . According to our protocol, block is finalized in the round if the number of nodes voting for the block is more than or equal to . Since two different blocks cannot be finalized in the same round, at least nodes agree that is the th block of their blockchains in round . Using contradiction, we assume that Since , node crashes before round and recovers in round so that block is appended into when node updates its blockchain. In this case, at least nodes have a same view on th block is at the end of round . Since there are also at least nodes agreeing on the th block in round is and the network size is , we have contradiction such that . Hence , which contradicts the assumption

In summary, all honest nodes are queried for a special position block should respond with the same result. In this case, if an honest node says block is valid, then other honest nodes also agree on its validity.

For liveness, we need correct nodes to have a same view to prevent SWIB from stalling. In other words, our protocol satisfies liveness of blockchain system with faulty nodes.

**Theorem 1. (Liveness)** Even if there are faulty nodes in blockchain system, the honest nodes can still terminate a block and obtain the following round randomness seed.

**Proof.** The best case is that no failures occur in blockchain system. Our protocol SWIB can always finalize a valid block by recovering full signature, which can produce the round randomness for the following round. We are required to prove that our protocol can guarantee liveness under the influence of adversary. The behaviors of malicious nodes controlled by adversary include: 1) intentionally not generating a valid block and 2) refusing to vote for the valid block. There are two cases for node failures:

Case 1. Block proposer is malicious. If a malicious block proposer does not generate any block, all honest nodes will timeout and commit on an empty block. If a malicious block proposer generates an invalid block, the block will fail to pass the block validation process. Thus, all honest nodes will refuse to vote for the invalid block and commit an empty block finally. In this case, each honest node will generate partial signature of an empty block hash. Once the full signature of the block hash is reconstructed from enough partial signature shares, the next round randomness will be generated successfully.

Case 2. malicious nodes refuse to vote for a block. In this case, the rest honest nodes would always generate partial signature shares of the block hash, and broadcast it to other nodes. Once collecting a threshold of partial signature shares, any honest node can recover the full signature and broadcast it to other nodes. Moreover, the rest honest nodes will also finalize the block and generate the following round randomness in the presence of node failures.

In summary, if the number of faulty nodes is at most , all honest nodes can always commit on a block and generate the following round randomness. [仍然是那个问题，这里说的没问题的前提条件都是“传输无丢失”，考虑无线信道传输丢失或adversary jamming的情况，大概就是另外一回事了…]

### 5.2.2 Random Generation Analysis

Our distributed randomness generation scheme is based on a robust threshold signature scheme. Existence of minority of malicious nodes cannot hinder the generation of a valid full signature if more than consensus nodes are honest. With the robust randomness, adversary cannot corrupt the process of block proposer election. Therefore, verifiable block proposer election adopting randomness can guarantee the block proposer is elected unpredictably and unbiasedly. However, it does not ensure that the elected block proposer works honestly. Suppose an adversary has less than 50% probability to be elected as block proposer per round, the upper bound of the probability that the adversary controls consecutive block proposer election is . [这一结论的表述和前面“less than 50%”的假设不一致，1/2k最多是上限] Let be the upper bound of the probability. Given , the adversary can at most control consecutive rounds to become block proposer. This indicates that the probability of adversary controlling more than consecutive rounds is less than , which is negligible. Therefore, our verifiable block proposer election algorithm can make sure that adversary cannot always control the election of block proposer.

The parameter used in the calculation of consensus ratio should be much greater than . If is too small, the consensus ratio of adversary will be high when it consecutively generates blocks. In this case, the elected probability of adversary might be very high, which means blockchain system may keep committing on empty blocks. Therefore, the system performance will be degraded. By choosing a large , the elected probability of adversary will not increase too much due to generation of consecutive blocks. In this case, honest nodes still have good chance to serve as block proposer. They will generate valid blocks, which can ensure the efficiency of transaction procession.

### 5.2.3 Attack Resistance Analysis

In this subsection, we analyze Sybil attacks resistance and Jamming attacks resistance of SWIB.

SWIB can efficiently prevent rational nodes from issuing Sybil attacks to affect the block proposer election process. Adversary usually launches Sybil attacks by generating pseudonyms. However, the elected probability will not increase due to adversary’s creating multiple pseudonyms in random block proposer election process. The elected probability of a node is proportional to its stability. We assume that the stability of an adversary is , and the total stability of the system is . If the adversary generates pseudonyms, whose stability values are , respectively. The sum of elected probability of the pseudonyms is . This indicates that generating multiple pseudonyms will not increase the elected probability. As a result, our protocol can resistant Sybil attacks in block proposer election.

Our protocol can ensure the security of block verification and finalization processes when adversary launches Sybil attacks. Since adversary can control nodes with pseudonyms to not send messages or send some error messages, the process of reaching consensus may be affected. Once a Sybil node becomes block proposer, it may generate an invalid block or does not generate any block. In this case, honest nodes will commit on an empty block and finalize a corresponding empty block eventually. Sybil nodes refuse to vote for a valid block if they are not the current block proposer. Since the number of Sybil pseudonyms is bounded by , the remaining honest nodes can terminate the block finalization and recover the full signature. Therefore, our protocol is resistant to Sybil attacks as long as adversary controls less than given threshold of voting power.

SWIB can normally operate when adversary issues jamming attacks. Adversary can influence consensus process by jamming the message deliveries of honest nodes. If honest nodes always cannot gather enough partial signature shares to recover full signature in time, the system liveness cannot be guaranteed. To resolve this problem, each node is required to maintain an estimate of adversary time window through sensing the wireless channel. According to the estimate, each node can dynamically adjust the timeout duration to ensure the collection of enough partial signature shares. As a result, SWIB can terminate consensus process even adversary issues jamming attacks. Such design can ensure the liveness of blockchain system under jamming attacks, further ensuring the security of wireless blockchain system.

Consensus interruption occurs when no node receives a sufficient partial signature in block finalization phase. In SWIB, utilizing an estimate of adversary time window ensures the finalization of block in wireless blockchain network under jamming attacks. [问题是无线链路传输丢失的事，怎么整，没有交代] We assume that the capability of adversary is - bounded, i.e., in any interval of consecutive slots, adversary can jam no more than slots, where and 0< ≤ 1.

**Proposition 3**. Let be the maximum number of time slots for each completed consensus round, and be the transmission probability of each node in a time slot. SWIB requires at least time slots to against any -bounded adversary, where is some target transmission probability.

**Proof**. Let . The minimum number of time slots for each consensus round is calculated as for target transmit probability . Assuming that in any interval of consecutive slots, adversary can jam no more than slots, where and 0< ≤ 1. For each slot, the jamming probability is . Thus, a message can be transmitted successfully after at least time slots. [这是说检测到信道被阻塞了的时候，会backoff，直到the channel is clear么？] A completed consensus process for each round requires time slots. Therefore, to ensure at least time slots with successful transmission probability , our protocol runs at least slots for each round under jamming attacks issued by any - bounded adversary. ■

Thus, to ensure the block finalization under jamming attacks, each node maintains an estimate of adversary time window to dynamically adjust round timeout. 【不该是每个节点自己独立估计，round timeout的时长，需要是全局一致的吧？】 In this case, SWIB can against -bounded adversary issuing jamming attacks.

# Simulation Results

In this section, we conduct simulation experiments to evaluate the performance of SWIB protocol. The impacts of various parameters are investigated, including block size, network size. We evaluate the performance of our protocol in terms of consensus latency and transaction throughput. The consensus latency is the average running time of a round. The transaction throughput is the average number of processed transactions per second (TPS). All the experiments were performed under Windows 10 operating system, and running on a machine with an Intel i7-10700F Core, 2.90GHz CPU, 8.00 GB RAM. All the reported results were the average of 10 tests.

We set a uniform transmit power for all nodes as mW, and the noise power as mW. The target transmission probability to be , and the target SNR threshold is assumed to be dB. We also fix the path loss exponent as , and bandwidth as MHz. In addition, all nodes are deployed into a square area with size , and the minimum distance between nodes is meter. From Friis equation, we assume and for the simulation we fix meter and meter, from the ISM (industrial scientific and medical) ratio bounds at 2.4GHz. Besides, the default transmission probability of each node in a time slot is set as  . [为啥*p*上要戴个帽子？；再有，你是否计算过，如果网络中节点总数是500的时候，*p* = 0.0125的话，每时隙的传输成功率么？会有多低？]

不管叫message传输中断率也好，block传输中断率也好，这个数据在上述设置之下，到底是多少？

## 6.1 Impact of Different Weights in Node Stability Calculation

[你是想说，，这个式子里，a和b这两个系数的影响么？]

[这说这么多，没看出有啥意义，一两句话就交代完了，干了啥就说啥就行了…说了半天，难以卒读] In SWIB, the verifiable block proposer election algorithm randomly determines a block proposer based on the stabilities of nodes. According to the calculation of node stability , nodes with large active time ratio and consensus ratio are believed to be more trustworthy. That is, nodes having invested much more deposits and packing more valid blocks are more willing to follow the SWIB protocol. Besides, nodes with lower stability values have lower cost to be malicious. This means that nodes have higher probability to generate valid blocks and maintain the latest blockchain history. However, only finalization of valid blocks can guarantee the efficiency of transaction processing in a blockchain system. Therefore, it is encouraging to select quality nodes as block proposers to generate more valid blocks. Both active time ratio and consensus ratio of nodes can affect the elected probability of nodes. In order to elect high-quality nodes, we study the probability of selecting quality nodes as block proposer in different cases.

We consider a blockchain system with network size , and set that nodes with lower active time and also lower number of latest generated blocks; nodes with lower active time and higher number of latest generated blocks; nodes with higher active time and lower number of latest generated blocks; nodes with higher active time and higher number of latest generated blocks. Fig. 6(a) shows how the nodes are distributed. To investigate the impact of active time ratio weight, we measure the percentage of electing high-quality nodes as block proposer in 100 consensus times.

(a) (b)

**Fig. 6. Percentage of high-quality under different coefficients.**

Fig. 6(b) shows that the percentage of high-quality nodes being elected as block proposers almost linearly increases with the active time ratio weight *a* increasing. Both active time ratio and consensus ratio affect the probability for high-quality nodes to be selected as block proposers. The maximum value of the percentage is less than 0.85 while the minimum value is larger than 0.7. When the active time ratio weight equals 0.7, the percentage high-quality nodes can approximately reach 0.8. [Fig. 6(b)里，哪有此事？] In this case, active time ratio has larger impact than consensus ratio in the stability calculation. Consensus nodes that have high remaining active time would like to be honest than those with low remaining active time due to the incentive and punishment mechanism in SWIB. To ensure a relative high probability for high-quality nodes to be elected as block proposers, we set the active time ratio weight *a* as 0.7 and consensus ratio weight *b* as 0.3 in the subsequent performance experiments.

## 6.2 Consensus Interruption Comparison

In this subsection, we evaluate how the consensus interruption probability is affected by the communication interruption probability. In this test, we compared the consensus interruption probability by SWIB with PBFT and assume all nodes are honest.

【别人可能更关心部分节点不诚实情况下的协议性能，或者说“也关心”】

  (a) Fault-free theoretical case (b) Fault-free practical case

**Fig. 7. Consensus interruption probability comparison. [这两个黄词，到底怎么个解释、什么差别，什么都不说，难道等着读者猜测个说法，安到你头上？]**

Fig. 7 compares the consensus interruption probability by different protocols with varying communication interruption probability. Fig. 7(a) shows the theoretical consensus interruption probability by the two protocols remains steady first, then increases to 1, which means no consensus can be reached in this case. The consensus in PBFT is interrupted with high probability when communication interruption probability is larger than 0.4. However, consensus process by SWIB can still work smoothly when the communication interruption probability is smaller than 0.5. This is because our protocol SWIB decouples leader selection (i.e., block proposer) from communication driven consensus process, our protocol is more stable than PBFT when nodes start to fail in receiving some messages. 【原因应该是收到partial signature shares而没有收到对应的block的时候，可以要求重传…而导致的, correct?】 The experiment results in Fig. 7(b) further validate the correctness of the theoretical results in Fig. 7(a).

## 6.3 Performance Comparison

In this subsection, we evaluate the impact of block size and network size on the protocol performance in terms of consensus latency and transaction throughput.

*1) Impact of block size*

In this experiment, we evaluate the impact of block size on the performance of SWIB. Each test lasts for 10 rounds. The block size varies from 0.5 MB to 5 MB, and the network size varies from 100 to 300 nodes.

1. Average latency (b) Average throughput

**Fig. 8. The performance of SWIB versus block size. [横轴“Block Sizes => Block size”; 左图纵轴“Sec. => s”； Figs. 9-10同样处理]**

[就图8里的试验，是怎么进行的？是从头到尾, 完整地把SWIB在你的仿真设置下按你说的设置跑的，然后画出相应的性能曲线？]

Fig. 8(a) shows that the consensus latency almost linearly increases with the block size increasing. When block size is smaller than 1 MB, the average consensus latency for all cases under study is shown to be below second. In Fig. 8(b), it is seen that the average throughput first increases rapidly, and finally tends to be flat, with the block size increasing. This indicates that only increasing block size cannot improve the average transaction throughput infinitely. When the block size equals 1 MB, SWIB can achieve large average transaction throughput with small average consensus delay. 【感觉不提每个节点以什么速率产生交易、以及交易是如何在网络中扩散的，缺乏这些，上面的结果，很难在读者心中产生共鸣】

*2) Impact of network size*

In this experiment, we evaluate the impact of network size on system performance. In here, each test lasts for 10 rounds.

(a) Average latency (b) Average throughput

**Fig. 9. The performance of SWIB versus network size. [左图纵轴改为Average Consensus Latency]**

Fig. 9 shows the average consensus latency and throughput performance by SWIB when the block size varies from 0.5 MB to 2 MB and the network size varies from 50 to 600 nodes. Fig. 9 (a) shows the average consensus latency versus network size. In Fig. 9 (a), it is seen that the consensus latency increases with network size while still acceptable even for 600-node networks and 1-MB blocks. Due to the broadcast nature of wireless channel, block transmission time does not vary with network size as we focus on studying one-hop wireless networks. However, block finalization requires enough partial signature shares, whose collection time grows proportionally with network size. This indicates that consensus nodes are required to verify more partial signature shares, and the recovery of full signature requires larger threshold. Meanwhile, the increase of network size will lead to more severe channel accessing competition, which eventually results in the increase of consensus latency. Each SWIB node is assumed to dynamically adjust its transmission probability by sensing the channel state in the estimated adversary time window. [这里，冒出adversary字样，似乎不太合适] This design choice can avoid sharp increase of consensus latency with the increase of network size. Since the block size is fixed, as consensus latency increases, the average transaction throughput drops. Fig. 9 (b) shows that average transaction throughput can reach approximate 2000 TPS for 600-node networks and 1-MB blocks. Thus, we can adjust the node transmission probability and block size to guarantee high transaction throughput with reasonable consensus latency for different network settings.

1. Average latency vs. network size (b) Average throughput vs. network size

**Fig. 10. Comparison of performance by different protocols.**

Fig. 10 compares the performance by SWIB and PBFT under fault-free scenario with varying network size and adversary power. In this experiment, each test lasts for 10 rounds, network size ranges from 50 to 600, and block size is fixed to 1 MB.

Fig. 10(a) shows that the average consensus latency by different protocols with different network size. In Fig. 10(a), it is clearly seen that SWIB has much lower consensus latency than PBFT. For example, our protocol SWIB can achieve consensus within 1 second in 600-node wireless networks while that for PBFT will be 2.7 second. This is because SWIB can securely reach block consensus in one round of voting while PBFT requires two-rounds communications to reach consensus on a block.

Fig. 10(b) compares the consensus latency by different protocols with varying adversary power in 500-node networks. In PBFT, view change happens when a malicious leader proposes an invalid block or does not propose any block. In this case, all honest nodes require two round communications to determine a new leader. In SWIB, honest nodes will discard invalid block and sign an empty block instead, then entering the next consensus round. In the experiment here, all malicious nodes are assumed not to send any messages in a consensus round. Under this assumption, in Fig. 10(b), it is seen that our protocol can work correctly when adversary power is smaller than 50% while PBFT will fail to reach consensus when adversary power is more than 33%. This indicates that our protocol has better fault tolerance performance than PBFT.

[层次逻辑清楚，是表述科技论文内容的基本要求，我希望你能从上面这一小节的修改，和你以前写的对比，真正地学习到如何到达这一目标！否则，如果哪天我不愿意再为乱写的内容修改的时候，事情就会搁置下去…]

## 6.4 Attack Resistance

In this subsection, we evaluate how Sybil percentage of Sybil nodes impacts the consensus performance in terms of transaction consensus latency and system throughput. We increase the percentage of Sybil nodes from 5% to 49%, and repeat the experiments in four different sized networks while fixing block size to 1 MB.



1. Average latency Vs. fraction of Sybil nodes (b) Average throughput Vs. fraction of Sybil nodes

**Fig. 11. The performance of SWIB when confronting Sybil attacks**

Fig. 11 shows the performance of SWIB with varying percentage of Sybil nodes in the network. In Fig. 11 (a), it is seen that the consensus latency by SWIB decreases with the percentage of Sybil nodes increasing. In SWIB, when a Sybil node is selected to serve as block proposer, empty block will be eventually generated. Compare with the valid block case, propagation of an empty block requires small number of time slots. Thus, the consensus latency will not significantly increase due to malicious block proposer in a round. Meanwhile, channel contentions between consensus nodes become light when the fraction of Sybil nodes increasing. This is because that Sybil nodes would not send messages during consensus process, or just send some error message with lower transmission probability. As result, the consensus latency declines when the percentage of Sybil nodes increases. With the increase of Sybil nodes percentage, the probability of generating empty blocks increases. In this case, the average number of confirmed transactions will decrease. In Fig. 11 (b), it can be seen that the average transaction throughput will decrease with the increase of percentage of Sybil nodes. Also, it is seen that SWIB can achieve more than 2000 TPS in blockchain network with 500 nodes and block size of 1 MB. As a result, our protocol can resist to Sybil attacks, and achieve acceptable performance when adversary issuing Sybil attacks. [中间的解释，没太看明白里面的逻辑]

Next, we evaluate the performance of SWIB in the presence of -bounded jammers. Here, we consider different jamming strategies in the attacks: one is random jamming such that the jammer randomly jams slots among slots, another is perpetual jamming such that the jammer jams consecutive slots in each period of slots. Each test lasts for 10 rounds. The value ranges from 0.1 to 1. The smaller is, the more time slots jammed. The adversary time window is chosen as 100 time slots.

(1) Average consensus latency (b) Average throughput

**Fig. 12. The performance of SWIB when confronting Jamming attacks.**

Fig. 12 (a) shows that the consensus latency decreases with the increase of under random jamming attacks while keeping steady under perpetual jamming attacks. This indicates that, for SWIB, random jamming attack is more powerful than perpetual jamming attack. When adversary issues perpetual jamming attacks, the transmission probability of nodes is reduced quickly by the maintained estimate of adversary time window . In this case, channel contentions in the network will be reduced, and nodes can transmit a message successfully within small time slots. However, when jammers issue random jam attack, transmission probability of nodes will not reduce significantly due to the estimate of adversary time window  not increase rapidly. The required time of partial signature shares aggregation will decline when increasing the value of . Thus, the consensus latency decreases with the increase of . Fig. 12 (b) shows that the throughput of SWIB does not fluctuate significantly under perpetual jamming attacks. As block size is fixed and consensus latency is reduced, the average throughput under random jamming attacks increases with the improvement of value . The throughput can exceed 1000 TPS in 500-node networks under -bounded adversary who launches random jamming attacks. Therefore, our protocol can securely operate in blockchain system when adversary issuing jamming attack, and achieve acceptable transaction throughput within reasonable consensus latency.

# Conclusion and Future Research

In this paper, we proposed a stable wireless blockchain consensus protocol SWIB, which can ensure stable and secure consensus process by adopting verifiable random function and threshold signature scheme in one-hop wireless networks. We use a new verifiable block proposer algorithm to prevent the formation of forks and choose a more reliable block proposer to generate a valid block. Besides, by combining the nature of wireless communication protocol with threshold BLS signature scheme, consensus nodes can verify and vote for block proposal in secure and efficient way. In addition, according to the characteristic of threshold BLS signature scheme, all honest consensus nodes can recover full signature to complete block finalization in distributed manner. Moreover, our protocol can still work efficiently in wireless blockchain system under Sybil attacks and jamming attacks. Analysis and results show the efficiency and security properties of the SWIB protocol.

In the future, we will investigate the multi-hop version of SWIB protocol in wireless ad hoc networks. We should study the energy consumption in consensus process, and design energy efficiency consensus protocol in wireless network. What's more, it is necessary to investigate the impacts of mobility in ad hoc wireless network. Also, we will explore whether our protocol can mitigate other attacks such as long-term attacks, nothing-at-stake attacks, eclipse attacks, etc.

# References:

[1] J. Xu, S. Wang, A. Zhou, and F. Yang, "Edgence: A blockchain-enabled edge-computing platform for intelligent IoT-based dApps," China Communications, vol. 17, no. 4, pp. 78-87, April 2020.

[2] T. Maksymyuk, J. Gazda, L. Han, and M. Jo, "Blockchain-Based Intelligent Network Management for 5G and Beyond," in Proceeding of 2019 3rd International Conference on Advanced Information and Communications Technologies (AICT), 2019, pp. 36-39.

[3] R. Jabbar, N. Fetais, M. Kharbeche, M. Krichen, K. Barkaoui, and M. Shinoy, "Blockchain for the Internet of Vehicles: How to Use Blockchain to Secure Vehicle-to-Everything (V2X) Communication and Payment?," IEEE Sensors Journal, vol. 21, no. 14, pp. 15807-15823, July15, 2021.

[4] S. Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System." https://bitcoin.org/bitcoin.pdf, 2008.

[5] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in Proceeding of 2017 Annual International Cryptology Conference, Springer, 2017, pp. 357–388.

[6] M. Castro, B. Liskov. "Practical Byzantine fault tolerance[C]." Proceedings of the 3rd Symposium on Operating Systems Design and Implementation (OSDI), 1999: 173-186.

[7] Z. Jiao, B. Zhang, L. Zhang, M. Liu, W. Gong, and C. Li. "A Blockchain-Based Computing Architecture for Mobile Ad Hoc Cloud," IEEE Network, vol. 34, no. 4, pp. 140-149, July/Aug. 2020.

[8] Z. Jiang, Z. Cao, B. Krishnamachari, S. Zhou, and Z. Niu, "SENATE: A Permissionless Byzantine Consensus Protocol in Wireless Networks for Real-Time Internet-of-Things Applications," IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6576-6588, July 2020.

[9] Q. Xu, Y. Zou, D. Yu, M. Xu, S. Shen, F. Li. "Consensus in Wireless Blockchain System," in WASA, 2020.

[10] Y. Zou, M. Xu, J. Yu, F. Zhao and X. Cheng, "A Fast Consensus for Permissioned Wireless Blockchains," in IEEE Internet of Things Journal, 2021.

[11] M. Xu, F. Zhao, Y. Zou, C. Liu, X. Cheng, and F. Dressler, "BLOWN: A Blockchain Protocol for Single-Hop Wireless Networks under Adversarial SINR," *IEEE Transactions on Mobile Computing*, early view, doi: 10.1109/TMC.2022.3162117.

[12] M. Xu, C. Liu, Y. Zou, F. Zhao, J. Yu and X. Cheng, "wChain: A Fast Fault-Tolerant Blockchain Protocol for Multihop Wireless Networks," in IEEE Transactions on Wireless Communications, vol. 20, no. 10, pp. 6915-6926, Oct. 2021, doi: 10.1109/TWC.2021.3078639.

[13] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," IEEE Commun.Surv. Tutorials, vol. 22, no. 2, pp. 1432–1465, 2020.

[14] M. Vukolic, "The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication," in Proc. Int. Workshop Open Problems Netw. Secur., 2015, pp. 112–125.

[15] S. Dziembowski, S. Faust, V. Kolmogorov, and K. Pietrzak, “Proofs of space,” in Annual Cryptology Conference. Springer, 2015, pp. 585–605.

[16] K. Karantias, A. Kiayias, and D. Zindros. "Proof-of-Burn," in Financial Cryptography and Data Security, 2020, pp. 523-540, doi: 10.1007/978-3-030-51280-4\_28.

[17] V. Buterin and V. Griffith, ‘‘Casper the friendly finality gadget,’’ 2017, arXiv:1710.09437. [Online]. Available: https://arxiv.org/abs/1710.09437

[18] F. Gai, B. Wang, W. Deng and W. Peng. "Proof of Reputation: A Reputation-Based Consensus Protocol for Peer-to-Peer Network." In Database Systems for Advanced Applications, 2018, pp. 666-681, doi: 10.1007/978-3-319-91458-9\_41.

[19] Hyperledger Fabric. [Online]. Available: https://cn.hyperledger. org/projects/fabric. 2019.

[20] Q. Wang, R. Li, S. Chen, and Y. Xiang. "Formal Security Analysis on dBFT Protocol of NEO."  [arXiv:2105.07459](https://arxiv.org/abs/2105.07459) [cs.CR], 2021

[21] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong, "Zyzzyva: Speculative byzantine fault tolerance," ACM Trans. Comput. Syst., vol. 27, no. 4, pp. 1–39, 2010.

[22] E. Buchman. "Tendermint: Byzantine fault tolerance in the age of blockchains." Ph.D. thesis, The University of Guelph, Guelph, Ontario, Canada, June 2016.

[23] Y. Gilad, R. Hemo, S. Micali, et al. "Algorand: Scaling Byzantine agreements for cryptocurrencies[C]." In Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28–31, 2017: 51–68.

[24] M. Zheng, M. Goldenbaum, S. Stańczak and H. Yu, "Fast average consensus in clustered wireless sensor networks by superposition gossiping," 2012 IEEE Wireless Communications and Networking Conference (WCNC), 2012, pp. 1982-1987, doi: 10.1109/WCNC.2012.6214113.

[25] M. Goldenbaum, H. Boche and S. Stańczak, "Nomographic gossiping for f-consensus", Proc. 10th Int. Symp. Model. Optimiz. Mobile Ad Hoc Wireless Netw., pp. 130-137, 2012.

[26] C. Newport and P. Robinson, "Fault-tolerant consensus with an abstract mac layer," arXiv preprint arXiv:1810.02848, 2018.

[27] F. Molinari, N. Agrawal, S. Stańczak and J. Raisch, "Max-Consensus Over Fading Wireless Channels," in IEEE Transactions on Control of Network Systems, vol. 8, no. 2, pp. 791-802, June 2021.

[28] H. Moniz, N. F. Neves and M. Correia, "Byzantine Fault-Tolerant Consensus in Wireless Ad Hoc Networks," in IEEE Transactions on Mobile Computing, vol. 12, no. 12, pp. 2441-2454, Dec. 2013, doi: 10.1109/TMC.2012.225.

[29] A. Boldyreva. "Threshold signatures, multi signatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme," in Proc. 6th Int. Workshop Theory Pract. Public Key Cryptogr., 2003, pp. 31–46.

[30] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing[C]". International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, 2001:514-532.

[31] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. "Secure distributed key generation for discrete-log based cryptosystems," in Proc.Int. Conf. Theory Appl. Cryptograph. Techn., vol. 1592, Aug. 2010, pp. 295–310.

[32] A. Goldsmith, Wireless Communications. Cambridge University Press, 1 ed., Aug. 2005.

[33] B. M. Nejad, S. A. Attia, and J. Raisch, “Max-consensus in a max-plus algebraic setting: The case of fixed communication topologies,” in Proc. 22nd Int. Symp. Inf., Commun. Autom. Technol., 2009, pp. 1–7.

[34] A. Boldyreva. Threshold signature, multisignature and blind signature schemes based on the gap-Diffie-Hellman-group signature scheme. In Y. Desmedt, editor, Proceedings of PKC 2003, volume 2567 of LNCS, pages 31–46. Springer-Verlag, Jan. 2003.