A Trust-Centric Privacy-Preserving Blockchain for Dynamic Spectrum Management in IoT Networks

Jingwei Ye, Xin Kang, *Senior Member, IEEE*, Ying-Chang Liang, *Fellow, IEEE*, and Sumei Sun, *Fellow, IEEE*

Abstract

Blockchain is a promising technology for future dynamic spectrum access (DSA) management due to its decentralization, immutability, and traceability. However, many challenges need to be addressed to integrate the blockchain to DSA, such as the trustworthiness of participating nodes' spectrum sensing results, privacy protection of sensing nodes' identities, and affordable lightweight consensus algorithms for IoT devices. In this paper, we propose a trust-centric privacy-preserving blockchain for dynamic spectrum access in IoT networks. To be specific, we propose a trust evaluation mechanism to evaluate the trustworthiness of sensing nodes and design a Proof-of-Trust (PoT) consensus mechanism to build a scalable blockchain with high transaction-per-second (TPS). Moreover, a privacy protection scheme is proposed to protect sensors' real-time geolocation information when they upload sensing data to the blockchain. Two smart contracts are designed to make the whole procedure (spectrum sensing, spectrum auction, and spectrum allocation) run automatically. Simulation results demonstrate the expected computation cost of the PoT consensus algorithm for reliable nodes is low, and the cooperative sensing performance is improved with the help of trust evaluation mechanism. In addition, incentivization and security are also analyzed, which show that our system can not only encourage nodes' participation, but also resist many kinds of attacks which are frequently arise in trust management mechanism and blockchain based IoT systems.

Index Terms

Dynamic spectrum access, blockchain, trust model, consensus algorithm, cooperative spectrum sensing.

I. Introduction

Recent years have witnessed the exponential growth of mobile data traffic. The rapid development of new wireless applications such as autonomous vehicles, remote healthcare, augmented

and virtual reality will continue driving the demand for radio spectrum [1]. However, the current static spectrum management method has resulted in under-utilization of spectrum resources [2]. To this end, dynamic spectrum access (DSA) has been proposed to allow secondary users (SUs) sense and then use the idle spectrum bands. Since single SU's sensing capability is usually limited, cooperative sensing [3] is proposed to fuse the sensing results of multiple SUs to improve the sensing accuracy. However, the traditional usage of a centre to collect and fuse sensing results faces the risk of single point of failure and the increasing management and regulatory costs. Moreover, SUs usually need to upload their sensing data to the fusion center at the risk of leakage of their privacy, such as identity and location.

As an emerging technology, blockchain has attracted attention from both academia and industry. By using blockchain, a decentralized resource management system can be established without the requirement of trustworthy central management agencies [4]–[8]. Moreover, smart contracts which are implemented on a blockchain can be used to replace traditional central management agencies and facilitate the cooperation of users [9]. Recently, researchers have investigated how to apply the blockchain technology to DSA. [10] summarizes the types of blockchains applicable in different spectrum sharing scenarios, and the possible advantages and disadvantages of applying blockchain technology to DSA. With the help of cryptocurrency issued by the blockchain and smart contracts, flexible and automatic spectrum trading markets are made possible for spectrum sellers and buyers. [11]–[15].

However, the integration of blockchain and DSA still faces many challenges. Firstly, although the data recorded on a blockchain is prevented from being tampered, the quality or value of it cannot be guaranteed. Especially, in a public blockchain, a malicious user can easily join the blockchain and record their data in the blockchain. The data from such a malicious user or an unreliable user is valueless or even harmful to cooperation tasks based on the data, e.g., cooperative sensing. Therefore, there is compelling need to evaluate the quality of the data from each user. Secondly, though the communication among Ethereum external accounts can be protected by encryption and decryption with their public and private key pairs, a contract account is not equipped with a key pair hence such protection method is not applicable to smart contracts. As a result, sensing results uploaded to the smart contract cannot be encrypted, and the sensitive information in the sensing results (such as real-time geolocation) can thus be accessed by malicious users. Thirdly, traditional consensus algorithms like Proof of Work (PoW) used in public blockchain introduce much computation overhead and thus make the

transaction processing speed very low. As a result, it is inefficient to directly use existing consensus algorithms (such as PoW) for spectrum management, especially considering the limited computation capabilities of IoT devices.

In this paper, we consider the DSA for an IoT network where IoT devices opportunistically access the licensed spectrum bands through cooperative sensing [16]. Blockchain is used as a platform for dynamic spectrum management. Specifically, key component designs for such a blockchain-enabled DSA system, including the trust evaluation mechanism, privacy protection, consensus algorithm and smart contracts, are studied. The main contributions of this paper are summarized as follows.

- We propose a trust evaluation mechanism to evaluate the trustworthiness of sensing nodes which participate in the cooperative sensing. We show that the proposed trust evaluation mechanism is effective in incentivizing sensing nodes to be honest, and thus improving the spectrum sensing result.
- We propose a strong privacy protection mechanism by combining ring signature with the commitment scheme for providing sensing nodes' privacy such as their real-time geolocation. The ring signature is used to protect sensing nodes' identities when they upload sensing data. The commitment scheme is proposed to figure out the connection between sensing data packet and its originated sensing node after the fusion of sensing results, so that the trust value of each sensing node is updated and the incentive tokens are distributed.
- We propose a new consensus algorithm named as Proof-of-Trust (PoT) by connecting the mining difficulty with a miner's trust value. We show that the proposed PoT can greatly reduce the computation cost of honest nodes and enhance the scalability of current blockchain-based DSA system.
- We design the system protocol and two smart contracts to make the whole DSA procedure, including spectrum sensing, spectrum auction, and spectrum allocation, run automatically. In addition, we implement and verify a prototype of our proposed protocol and smart contracts using Solidity [17] on the Ethereum testnet.

The rest of the paper is organized as follows. In Section II, we introduce the basic concept of blockchain and smart contract. In Section III, we describe the works related to the application of blockchain to DSA. In Section IV, we introduce our system model. In Section V, we describe the design of our proposed protocol including the block structure, the trust evaluation mechanism,

the consensus algorithm and the privacy protection scheme. In section VI, the design of smart contracts and workflow are introduced. In Section VII, we discuss the performance of the trust evaluation mechanism, and analyze the security of the system. Finally, Section VIII concludes this paper.

II. Preliminaries

A. Blockchain

Blockchain is a growing chain of data blocks with a specific data structure constructed using cryptographic algorithms. It uses hash pointers to connect the data blocks to form an entangled chain. In this way, the integrity of the data can be protected. Blockchain uses a measurable and verifiable mechanism to reach a consensus among nodes in the network on the generation of new blocks. Such a mechanism is usually referred to as the *consensus algorithm*. On the premise that the data block entanglement is guaranteed, the structure and the consensus algorithm of the blockchain can be flexibly designed based on the demand of application scenarios.

Blockchain can be roughly classified into three types, depending on degree of openness, which are public blockchain, private blockchain, and consortium blockchain. Public blockchain is designed to enable and to record the transactions in a public network, which means that any node can freely join or leave the blockchain network without authorization. Bitcoin, the most famous digital cryptocurrency, is generated and managed by a public blockchain. Consortium blockchain is a permissioned blockchain, which is open to authorized members of external institutions in limited roles and functions. The authorization of the node to join are all determined by an authorized organization. The nodes in a private blockchain network trust each other. Thus, private chains can simplify operations in data authentication and the consensus algorithm to improve the efficiency.

B. Smart Contract

In the 1990s, Nick Szabo first proposed the concept of smart contract [18]. It was defined as computerized transaction protocol that execute terms of a contract. In the field of cryptocurrency, a smart contract is defined as an application or a program that runs on a blockchain. Normally, it contains a set of digital agreements with specific rules. These rules are predefined in the form of computer source codes, and all network nodes will copy and execute these computer source

codes independently. Smart contracts are highly customizable and can be flexibly designed to provide different services and solutions.

C. Ring Signature and the Commitment Scheme

Ring signature was first introduced in 2001 [19] which follow a ring-like structure of the signature algorithm. It is a type of digital signature that can be performed by any member of a group of users without the agreement of others. Therefore, the only "open" information is that a message signed with a ring signature is endorsed by someone in a particular group of people. Ring signatures are deliberately designed so that it is computationally infeasible to determine which of the group members' keys was used to produce the signature. Ring signatures are similar to group signatures but differ in two key ways: firstly, there is no group administrator to revoke the anonymity of an individual signature; secondly, any set of users can be used as a group by the signer without additional setup. In public blockchains, there is no leader or manager in the system, who can serve as the group manager required in the group signature scheme. Thus, ring signature scheme is more suitable for this scenario than group signature.

Commitment schemes are basic elements in many cryptographic protocols. Usually, the two-stage commitment scheme is used to enable a party to commit itself to a value while keeping the value secret first, and then rebuild the connections between the data packets and their corresponding owner [20].

III. Related work

Blockchain was first proposed as a secure decentralized ledger for recording spectrum transaction data and analyzing overall spectrum usage in dynamic spectrum management. In [21], the authors proposed to use the blockchain as a distributed database to securely store the transactions in dynamic spectrum sharing, and to use the digital currency issued by the blockchain as the currency for spectrum auctions. In [15], a smart contract running on Ethereum was proposed as a platform to provide spectrum sensing services. In [13], a blockchain-based spectrum trading and sharing scheme was proposed for Mobile Network Operators (MNOs) to lease their spectrum to secondary Unmanned Aerial Vehicle (UAV). In [22], spectrum sharing contracts deployed on a permissioned blockchain platform are constructed for multi-operators spectrum trading. In [14], the authors propose to use smart contracts to securely store and then autonomously implement

the spectrum leasing agreements, thus to avoid the interference from the disordered spectrum access.

However, these works did not make any improvement to the existing blockchain. Thus, the TPS of the blockchain is low, limiting the performance of the blockchain-based DSA platform. Thus, in [23], the authors proposed a consensus algorithm named as "blockchain-KM protocol" by modifying the PoW, to speed up the transaction processing for the license-free spectrum sharing. In [24], a novel consensus algorithm named as proof-of-strategy was proposed, where the best strategy to allocate spectrum bands in the whole network is the proof of authority to publish a new block. Such a consensus algorithm can not only regulate the transaction generation, but also optimize the spectrum allocation. However, proof-of-strategy couples the generation of transactions with spectrum allocation strategies which make consensus algorithm complex, and this reduces the efficiency of consensus mechanism and also the system throughput. In [25], by modifying the PoW, the authors proposed a consensus algorithm named as Proof-of-Device, which selects a device, represented by its unique identifier, to publish the new block in a way like lottery. It thus eliminates the heavy computation cost in PoW. However, this consensus algorithm relies on the unique ID of each device, and the ID is relatively constant. Thus, malicious users may predict the winning miner of each round, and launch a DoS attack on this device which may threat the security of the system.

There are also many other public blockchains that improve TPS through other types of consensus algorithms, such as PoS [26]–[28], DPoS [29]–[31], etc. They can improve the TPS of blockchain system, but they also have some shortcomings. For example, for PoS, all nodes can earn interest through long-term holding of tokens, which make fewer people willing to sell tokens and reduce the liquidity of currency. For the DPoS adopted in the EOS network, in order to improve performance, there are only 21 super nodes. Due to this reason, this blockchain is no longer really decentralized, but “partially decentralized” [32]. In this paper, we need a consensus algorithm which is more suitable for IoT network environment.

On the other hand, blockchain system is an open system, hence the reliability of data source needs to be guaranteed. In [33], the data uploaded by the sensors are compared with the data collected by the trustworthy validator to determine the credibility degree of the sensor, in this way, the system can evaluate the credibility of nodes for better performance. However, this method is a semi-centralized method which needs to deploy trustworthy validators.

Different from the above works, we not only delve into the design of block structure and

consensus algorithm, intending to overcome the scalability problem of the blockchain technology, but also design a decentralized trust evaluation mechanism and a privacy protection mechanism for the openness and transparency of blockchain system while protecting the data's confidentiality and the sensors' privacy. To the best of our knowledge, this is the first work of joint trust evaluation and privacy protection mechanism in cooperative sensing in a blockchain-enabled DSA system.

IV. System Model, challenges and design guidelines

In this section, we introduce the dynamic spectrum access scenario in future IoT network, and list the difficulties and challenges in applying blockchain technology to this scenario. Finally, we give the guidelines on how to design a blockchain-based DSA system.

A. Blockchain-enabled DSA System Model

In this paper, we consider the DSA in a two-tier cognitive radio network consisting of primary users (PUs) and secondary users (SUs). In this network, PUs are inactive users who may not use their spectrum all the time so SUs can detect these spectrum holes and use them. As illustrated in Fig. 1, SUs can be a variety of heterogeneous IoT devices, which form a mesh network. SUs access the spectrum bands of the PU in an opportunistic manner based on cooperative sensing, and a blockchain is used as the platform of the cooperative spectrum sensing decision and the spectrum allocation.

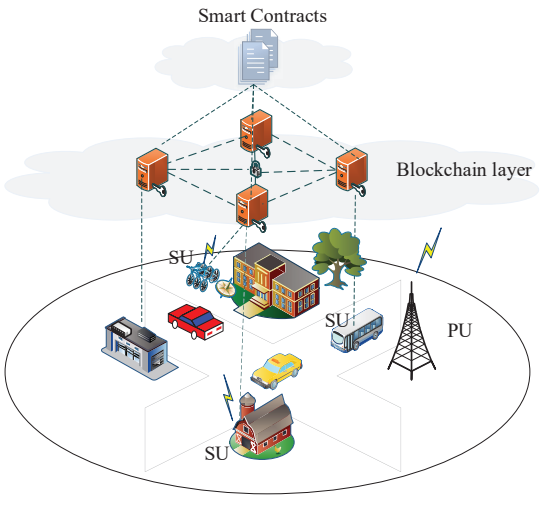


Fig. 1: System model

Basically, as shown in Fig. 2, there are four phases in our blockchain-enabled DSA system.

1) *Individual Sensing*: The sensors perform spectrum sensing to detect the state of the PU's spectrum band.
2) *Sensing Fusion*: Each of the sensor broadcasts its sensing data which mainly consists of the sensing result and the geolocation where the sensing is performed. These data can be raw data or quantized bits, which will be fed to a smart contract implemented upon a blockchain. Then, the smart contract provides the final sensing result according to predefined fusion rules.
3) *Spectrum Allocation*: When the final sensing result indicates that PU is inactive, it should be decided which spectrum requester can access the spectrum. Spectrum auction, as a common and fair way for spectrum allocation, is chosen to achieve that. In the spectrum auction, the digital cryptocurrency, which is issued by the blockchain through incentive mechanisms for spectrum sensing and mining, is commonly considered.
4) *Spectrum Access*: The SU who obtains the access right through spectrum auction accesses the idle spectrum for data transmission.

Compared with traditional cooperative spectrum sensing and spectrum allocation methods, blockchain-enabled DSA has the following benefits:

- *Decentralization*: Blockchain-based dynamic spectrum access does not require the deployment of trusted central nodes which avoids the single point of failure.
- *Transparency*: The blockchain ledger can record the whole process of DSA.
- *Automatic*: Using smart contract instead of traditional contracts, we can achieve automatic spectrum management and on-chain payment settlement.
- *Flexibility*: For spectrum trading on the blockchain platform, diversified spectrum trading rules can be dynamically enforced by adjusting parameters of smart contracts. Compared with the current fixed paper-based contracts signed between mobile users and operators, this method has higher flexibility.

B. Difficulties and Challenges

Traditional cooperative spectrum sensing is divided into two modes: centralized mode and distributed mode. Centralized cooperative spectrum sensing requires the deployment of a central fusion node to process the data collected from sensors which usually contain sensors' geolocation

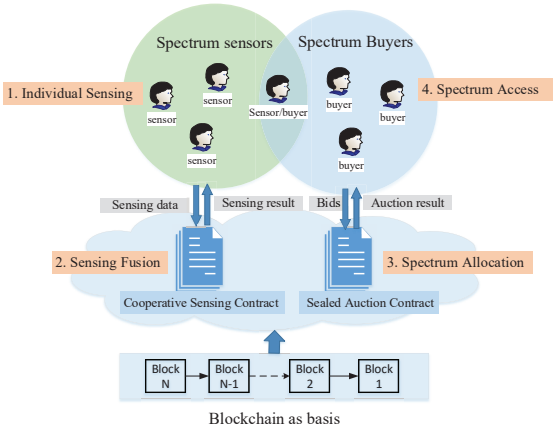


Fig. 2: Blockchain based DSA

information. Therefore, this mode not only requires redundant regulation fees (usually charged to maintain the operation of the central node), but also faces the risk of data leakage when the central node is attacked. While the distributed mode often suffers from the fake reporting issue caused by malicious nodes and the iterative-based malicious node detection algorithms [34] introduce extra computation cost.

Therefore, how to implement the cooperative sensing in a secure and effective way is a challenge. The decentralized nature of the blockchain system is promising to realize distributed cooperative sensing and encourage all sensors to share spectrum sensing results. However, there are still some challenges when applying blockchain into spectrum management. Firstly, IoT devices usually lack sufficient computing and storage resources to maintain and store the whole blockchain ledger. Secondly, a malicious node can easily join a public blockchain network and threaten the reliability of cooperative spectrum sensing. Moreover, the sensing-related data recorded on blockchain or collected by smart contracts is usually in an unencrypted format which may leak the private information of IoT devices contained in their sensing data packets.

C. Design Guidelines

Considering above challenges, our discussion on system design mainly includes the following three parts.

- *Trust Evaluation*: Specially, in a public blockchain network, anyone can join or exit the system at will, and one user can create multiple accounts in the system. Therefore, if there is no suitable node evaluation mechanism, malicious users can launch Sybil attacks easily.

This is because, malicious nodes can keep on applying for accounts and continue launching attacks. Therefore, the system should be able to evaluate the trust values for nodes, and identify the malicious nodes based on the their trust values [35].

- *Lightweight Consensus Algorithm*: Due to the limitation of computing power and storage resources of IoT devices, maintaining a normal public blockchain system is a heavy burden. To reduce the computing cost, it is necessary to invent a new consensus algorithm with low computational and storage complexity. On the other hand, as time goes by, the length of the ledger keeps on increasing. To reduce the storage cost, methods such as edge storage can alleviate storage pressure on the blockchain nodes but may introduce additional budget. Thus, a method that can directly reduce the storage cost of the blockchain is preferred.
- *Privacy Protection Mechanism*: Smart contracts does not have public-and-private key pair and thus can not communicate with external accounts in an encrypted manner. Therefore, when a smart contract is used as the data fusion center of cooperative spectrum sensing, a privacy protection mechanism is needed to ensure that the private information of the sensors will not be leaked during the interaction between sensors and the smart contract.

V. Blockchain Design, Trust Evaluation and Privacy Protection

In this section, we first introduce the block structure of our blockchain, then we propose the trust evaluation mechanism to evaluate the trustworthiness of each SU on spectrum sensing. After that, a high-efficiency consensus algorithm is proposed for our blockchain. Finally, a privacy protection mechanism is proposed for protecting participants' privacy.

A. Block Structure

The block structure should be tailor-designed for the blockchain-based DSA system. Some of the unnecessary components in the block can be removed to reduce the overheads on the block transmission and storage and other necessary information, such as the trust value of the block generator, needs to be recorded in the block header to prevent possible attacks in the DSA system.

- Block header: As shown in Fig. 3, The block header mainly contains (i) merkle root of transactions; (ii) the merkle root of account states; (iii) hash value of the header of the previous block, as the hash pointer of the chain structure; (iv) trust value of the miner who mines this block; (v) signature of the miner who mines this block; (vi) a *Nonce*, which is

TABLE I: List of notations

Notations	Descriptions
$N_{i,w}(n)$	Accumulated number of wrong sensing results of $sensor_i$ at round n
$N_{i,r}(n)$	Accumulated number of right sensing results of $sensor_i$ at round n
P	Length of window for calculating $N_{i,w}(n)$
$\mathcal{D}_n^i$	Mining difficulty of node i at block n
$\mathcal{H}(\cdot)$	Hash operation
R_{sleep}	Number of inactive rounds since last active round
RND_i	Random number picked by $sensor_i$
SR	Sensing result of unknown but legal sensors
SR_i	Sensing result of $sensor_i$
N_1	Number of required sensors in Cooperative Sensing Contract (CSC)
N_2	Number of required bidders in Spectrum Auction Contract (SAC)
TV_{thr}	Trust value threshold set in CSC
T_{ddl}	Sensing deadline in SAC
d_s	Deposit for sensing
d_a	Deposit for auction
T_{self-d}	Time for SAC to self-destruct
TS	Timestamp
loc	Location of unknown but legal sensors

the solution of Hash puzzle of the consensus algorithm; (vii) timestamp when the block is mined.

- Block body: The block body mainly stores transaction generated during spectrum allocation using merkle tree. Unlike the unspent transaction output (UTXO) [36] model used in Bitcoin, the system based on the account model usually contains an account tree to record the account states such as the balance and the trust value of each account. Merkle patricia trie (MPT) [37] which is a commonly adopted data structure, is used here to store the account states.

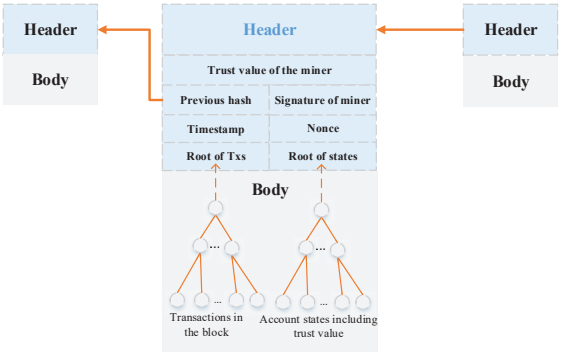


Fig. 3: Block structure

B. Trust Evaluation Mechanism

Although the blockchain can guarantee that data recorded in a block will not be tampered, it cannot guarantee that the data source is trustworthy. Especially, in a public blockchain, a malicious node can upload misleading data which may interfere the cooperative sensing based on such data. In this case, the performance of cooperative sensing will be degraded, and the honest and reliable sensors will be discouraged from participating in cooperative sensing. To this end, we design a trust evaluation mechanism to evaluate the credibility of every sensing node in the blockchain. Instead of using a central authority to record and maintain every account's trust value, we propose to include the trust value as an attribute of the blockchain account, which is recorded in the ledger and maintained by all the miners. In the following, we will discuss the design of trust value updating method.

Intuitively, the trust value of a sensing node should be adjusted based on its performance of the current and historical cooperative sensing. In detail, if it plays a positive role in the cooperative sensing, its trust value should be increased, and vice versa. Here, we define that the effect of a sensor is *positive* when its sensing result is consistent with the cooperative sensing result, and is *negative* when its sensing result is inconsistent with the cooperative sensing result, or when the node does not participate in cooperative sensing. Denote the number of times when the sensing result of the i-th node is consistent with the cooperative sensing result as $N_{i,r}$; the number of times when the sensing result of a node is inconsistent with the cooperative sensing result as $N_{i,w}$.

In a sensing-based system, trust is generally determined by two factors: the number of correct sensing and incorrect sensing. It can be expressed as

$$TV_i(n) = g\left(N_{i,r}(n), N_{i,w}(n)\right) \tag{1}$$

where $g(\cdot;\cdot)$ is a two-dimensional function, and should have the following properties:

- $\forall N_{i,r}(n), N_{i,w}(n) \in [0, +\infty)$, $g(\cdot;\cdot) \in [0,1]$;
- $g(\cdot;\cdot)$ is an increasing function of $N_{i,r}(n)$, and is a decrease function of $N_{i,w}(n)$;

Inspired by [38], we propose a new model to calculate trust value of $sensor_i$ denoted by $TV_i^{(1)}(n)$ in sensing round n, where n is also the number of sensing tasks published since this system was created.

$$TV_i^{(1)}(n) = e^{-\rho N_{i,w}(n)}\left(1 - e^{-\eta N_{i,r}(n)}\right), \tag{2}$$

where $\rho > 0$ and $\eta > 0$ are coefficient that determine how fast the trust value changes with respect to $N_{i,w}$ and $N_{i,r}$, respectively.

Moreover, we let $N_{i,w}(n)$ decay by $\frac{1}{P}$ every time when the node participates in sensing, and we only consider the latest P sensing rounds of each node. In this way, the effect of a wrong sensing result on the trust value will gradually be degraded over time. As a result, a node that unintentionally submitting an inconsistent result will be gradually forgotten. Mathematically, $N_{i,w}(n)$ can be defined as

$$N_{i,w}(n) = \sum_{m=n-P}^{n} r_i(m) \left(1 - \frac{n-m}{P}\right), \tag{3}$$

where $r_i(m) = 1$ if $sensor_i$ broadcasts the wrong sensing result in round m, and $r_i(m) = 0$ otherwise.

Besides, we denote the number of rounds when a sensing node is inactive as R_{sleep}, and we add a function $f(R_{sleep})$ which models the negative impact of being inactive on the trust value. It is designed to satisfy the following criteria.

- $\forall R_{sleep} \in [1, +\infty)$, $f(R_{sleep}) \in (0, 1]$,
- $\frac{\partial f_{dcv}}{\partial R_{sleep}} < 0$,

The first criterion is for normalization; the second criterion is to ensure that the negative effect increases as R_{sleep} increases. Moreover, as R_{sleep} increases, the downward trend of the function is first gentle, and then becomes severe. This is designed to punish the node that does not participate in the sensing process. The punishment is light when the peer does not participate only a few times, but becomes severe when the node always does not participate in the sensing. $f(R_{sleep})$ will reach a certain low value when R_{sleep} is big enough. For example, the trust value of a node will reduce to r_1 ($r_1 < 1$) of its original value for k_1 consecutive non-participation, and to r_2 ($r_2 < r_1 < 1$) for k_2 consecutive non-participation. Here, a piecewise function consisting of multiple linear functions is utilized as our $f(R_{sleep})$.

$$f(R_{sleep}) = \begin{cases} \frac{k_1 - R_{sleep}}{k_1} \cdot (1 - r_1) + r_1 & 0 \le R_{sleep} \le k_1 \\ \frac{R_{sleep} - k_2}{k_1 - k_2} \cdot (r_1 - r_2) + r_2 & k_1 \le R_{sleep} \le k_2 \\ r_2 & k_2 < R_{sleep} \end{cases} \tag{4}$$

These parameters can be set flexibly. If the actual deployment requires stricter penalties, then all these parameters should be set smaller.

By adding the attenuation function, the model in (2) can be modified as:

$$TV_i^{(2)}(n) = \begin{cases} TV_i^{(2)}(n-1) + \Delta TV_i \cdot f(R_{sleep}), & \text{if } \Delta TV_i > 0 \\ TV_i^{(1)}(n), & \text{if } \Delta TV_i < 0 \\ TV_i^{(2)}(n-1) \cdot f(R_{sleep}), & \text{if inactive} \end{cases} \tag{5}$$

where $\Delta TV_i = TV_i^{(1)}(n) - TV_i^{(2)}(n-1)$ is defined as the original increment of trust value, which can be either positive or negative. The design criterion of (5) is explained as follows. We consider three cases. First of all, when the sensor senses correctly, the original increment of trust value is positive, i.e., $\Delta TV_i > 0$. Such increment is first degraded by the negative effect of sleeps in the previous sensing activities, i.e., $f(R_{sleep})$, and then added to the trust value. Secondly, when the sensor gives the wrong sensing result, the original increment of trust value is negative, i.e., $\Delta TV_i < 0$. In this case, we choose not to consider the negative effect of the sleep in previous sensing rounds, in order to encourage the sensors which unintentionally derive the wrong sensing result in this time. Finally, for sensors who do not participate in sensing, their trust value will not remain unchanged but gradually decrease as their sleep time increases. This design is to ensure that sensors can only maintain its high trust value by actively participating in spectrum sensing.

C. Consensus Algorithm

The traditional consensus algorithm for public blockchain like PoW is computation-intensive, which may be inapplicable in IoT networks, where IoT nodes are usually with limited computational capabilities. One reason why the PoW is designed to be computation-insensitive is that it assumes there is nearly no trust among nodes in a blockchain network. Nevertheless, with the trust value mechanism proposed in this paper, we can evaluate the credibility of a node on spectrum sensing in the blockchain network. Therefore, based on the trust value, we can optimize our design of the blockchain consensus algorithm, forking solution and blockchain compression.

Proof of Trust (PoT): Intuitively, the nodes with higher trust value on sensing are more likely to be honest in publishing a new block. Therefore, we can reduce the difficulty of such nodes to mine a new block. In this way, the computation consumption for reliable nodes will be decreased,

and a block will thus be more quickly mined and published. Based on PoW and trust value, we propose a PoT consensus algorithm. Different from Proof-of-Stake (PoS), PoT assigns high-trust nodes lower mining difficulties, and trust can be gained only by providing correct sensing results and will decrease over time. Thus, high-trust nodes are not the same as high-stake nodes. To be specific, for successfully mining, the miner with higher trust value is required to find a hash value with fewer leading zeros and vice versa. Like PoW, PoT can work securely under the assumption that the majority of hashrate is controlled by honest nodes.

Mathematically, assuming that the difficulty of node i at block n is denoted as $\mathcal{D}_n^i$. D_n^i is determined mainly by β_n and a function $g(TV_i)$, where $g(TV_i)$ should have the following properties:

- $g(TV_i)$ is a decreasing function of TV_i;
- $\forall TV_i \in [0,1]$, $g(\cdot) \in$ [0,1];

In this paper, D_n^i is denoted as

$$\mathcal{D}_n^i = \beta_n \cdot \left(1 - sin\left(\frac{\pi}{2} \cdot TV_i\right)\right), \tag{6}$$

where β_n denotes the base difficulty of block n. $TV_i \in [0,1]$ is the trust value of node i. The initial difficulty is denoted as β_0 which can be determined by evaluating the computing power of actual IoT devices. How to determine β_0 will discuss this problem in the simulation section.

We denote the timestamp of block n as T_n, the ideal time interval of two consecutive blocks is T_0. Then, we have

$$q = \left\lfloor \frac{T_{n-1} - T_{n-2}}{T_0} \right\rfloor, \tag{7}$$

where q is defined as adaptive adjustment factor of base difficulty, and g is defined as the difficulty adjustment granularity, which can be written as

$$g = \left\lfloor \frac{\beta_{n-1}}{128} \right\rfloor, \tag{8}$$

The updating of β_n can be denoted as

$$\beta_n = \beta_{n-1} - q \cdot g, \tag{9}$$

Forking Solution: As the mining speed increases, there may be multiple blocks mined at nearly the same time. Because of the communication latency, the block that is first mined might not be the first to be received by all the nodes in the blockchain network. Instead, the block that is first received and recognized by different nodes might be different. In this case, the blockchain forking

occurs. It is harmful and thus needs to be solved. Due to the timeliness of spectrum, we expect that the spectrum sensing and allocation information can be recorded in time, which becomes a checkpoint in blockchain ledger at each sensing round. This means that our consensus algorithm has deterministic finality, i.e., PoT will not fork. To avoid forking, a unique legal block will be determined in time according to Algorithm 1 at each sensing round, and miners are required to validate every transaction of candidate blocks. In Algorithm 1, we propose a trust value based forking solution. It first compares the trust values in the head of blocks, and then validates the block with the highest trust value. If the trust values of multiple blocks are the same, the block whose timestamp is earlier is selected as the valid block. If it is still tied, it compares the hash value of the blocks, and then selects the block with the smallest hash value as the valid block. Since the probability of hash collision is negligible, Algorithm 1 can select only one valid block eventually.

Blockchain Compression: Since IoT devices are usually with limited storage space, each time when the blockchain grows by L blocks, compression will be performed. For the compression of blockchain, in the existing literature, the RSA accumulator as a data structure, which functions similarly to that of a Merkle tree, can be used to compress the blockchain [39]. Another approach to compress blockchain is to use the chameleon hash function to replace the traditional hash function of the blockchain [40] . Here, using the trust value, we propose a compression method called as *Trust-based Compression (TBC)*. The node with the highest current trust value will be authorized to compress the blockchain. To be specific, such a node will first extract the account tree in the last block as the body of the new block, calculate a hash value for the body, and finally combine the obtained nonce, miner's signature, and timestamp to form the block header. The obtained block is the new genesis block. Since each node stores the original blockchain, it is easy to check whether the account status has been tampered by the selected node during the compression process. After a node verifies the new genius block, it will clear the original blockchain.

D. Privacy Protection Mechanism

The location where a sensing node senses the spectrum bands is useful for the fusion center to cluster the sensing nodes and to improve the cooperative sensing accuracy [41]. Therefore, alongside the sensing result, the location is needed to be uploaded by a sensing node. However, it is difficult to protect the location information of a sensing node from being leaked when a smart

Algorithm 1 Block Selection

Input: the block to be compared: $Block_i$, $Block_j$;

Output: The winner block;

```
if TV_i < TV_j then
    Block_i wins.
else if TV_i > TV_j then
    Block_j wins.
else
    if Timestamp_i < Timestamp_j then
        Block_i wins.
    else if Timestamp_i > Timestamp_j then
        Block_j wins.
    else
        if H(Block_i) < H(Block_j) then
            Block_i wins.
        else
            Block_j wins.
        end if
    end if
end if
```

contract is used as the fusion center. This is because a smart contract account is not equipped with a key pair which can be used by sensors to encrypt their upload data packet.

The privacy protection issue in this case is to hide the source of a sensing packet, i.e., from which sensing node the sensing packet comes. However, the sensing node cannot be allowed to be totally anonymous when submitting sensing packet because this will make the cooperative sensing system vulnerable to malicious attacks. To this end, we propose the use of ring signature [19] to hide the source of a sensing packet in a group of valid sensing nodes. Also, the smart contract as the fusion center can identify the validity of each received sensing packet. Moreover, since sensing packets are unencrypted, fusing these sensing results can be carried out directly and automatically in the smart contract.

In the following, we illustrate the procedures of one sensor, denoted by $Sensor_s$, in generating the ring signature.

1. $Sensor_s$ selects $n-1$ legal sensors to form a group and collect their public keys. The format of a sensing data packet, denoted as msg, is given as follows

$$msg = \{\mathcal{H}(msgID), SR, time, location\}, \tag{10}$$

where $\mathcal{H}(msgID)$ is used in the update of the trust value and SR is the sensing result denoted by one bit.

2. $Sensor_s$ uses one-way hash function to compute $k = \mathcal{H}(msg)$, which is a symmetric key of the symmetric encryption function E_k.
3. $Sensor_s$ generates a random value for each of other members in the group where it belongs to. Specifically, the random number x_i is first generated for the i-th node in the group. It then calculates corresponding y_i=$g_i(x_i)$ using corresponding public key, where the function $g_i(\cdot)$ is the encryption function encrypted with the public key pk_i.
4. $Sensor_s$ finds the solution to the ring equation (11) and gets the undetermined parameter y_s, where v is a random value chosen by $Sensor_s$. Then, $Sensor_s$ calculates x_s using its own privacy key: x_s=$g_s^{-1}(y_s)$. To find solution, a private key of a sensor in this group is needed , anyone who is not in the group cannot generate a legal ring signature.

$$\begin{aligned} C_{k,v}(y_1, y_2, ..., y_n) = E_k(y_n \oplus E_k(y_{n-1} \oplus E_k(... \\ \oplus E_k(y_1 \oplus v)...))) = v, \end{aligned} \tag{11}$$

Finally, a valid ring signature, denoted as $Ring_{sig}$, can be generated by $Sensor_s$.

$$Ring_{sig} = (pk_1, pk_2, ..., pk_n, v, x_1, x_2, ..., x_n), \tag{12}$$

The verification process involves the following 3 steps.

1. Calculate all y_i using corresponding public key pk_i and x_i;
2. Calculate the symmetric key $k = \mathcal{H}(msg)$;
3. Verify that if $C_{k,v}(y_1, y_2, ..., y_n) = v$ holds.

If the verification succeeds, the smart contract recognizes that msg is sent by a valid sensing node from the group $\{Sensor_1, Sensor_2, ..., Sensor_n\}$. In this way, we can cut the connection between the data packet and its corresponding owner.

However, for the trust value update and payments assignment process, it is necessary to know the mapping of the sensing result and its corresponding sensor. To this end, we design a two stage commitment scheme as follows.

This scheme consists of two stages including *Commit Stage* and *Reveal Stage*. As shown in Fig. 4, at the commit stage, each sensor participating in the cooperative sensing needs to submit the msg whose privacy is protected by the ring signature, and the field "hash of $msgID$", i.e. $\mathcal{H}(msgID)$, is used as a commitment. $msgID$ is an identifier which is the input of hash function, and can be any message such as "I am User 3" or "I like apples". At the reveal stage, each sensor directly uploads the unhashed $msgID$ for miners to verify the consistency. This checking mechanism is effective since hash function is preimage resistance and second-preimage resistance [42], i.e., given a hash output $\mathcal{R}$=$\mathcal{H}(msgID)$, it is difficulty to find the input $msgID$ or another input $msgID'$ such that $\mathcal{H}(msgID) = \mathcal{R}$ or $\mathcal{H}(msgID') = \mathcal{R}$.

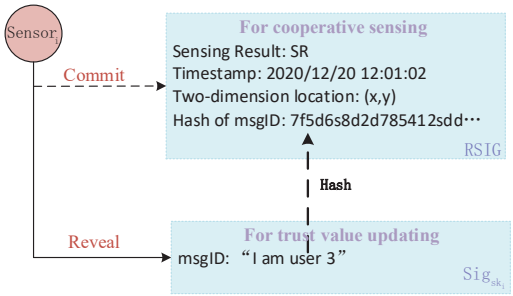


Fig. 4: Two-Stage commitment scheme.

VI. Smart Contracts and DSA Protocol Design

Smart contract are computer codes that run on the blockchain platform, which are automatically executed when the predefined conditions are satisfied. Moreover, smart contracts are not controlled by any third party. Therefore, we propose to use smart contracts to realize the automated operation of cooperative spectrum sensing and spectrum auction on blockchain. In this section, we will give the design of the corresponding smart contracts, and then propose the DSA protocol based on these smart contracts.

A. Smart Contract Design

We first discuss the design of our smart contracts for cooperative sensing and spectrum auction. In the following, we describe the parameters and functions in the two smart contracts,

respectively.

Cooperative Sensing Contract (CSC): The parameters in CSC that need to be predefined are T_{ddl}, d_s, N_1 and TV_{thr}, which are introduced as follows.

- T_{ddl}: The deadline for a sensor to send its sensing packet to CSC;
- d_s: The deposit that a sensor needs to pay before participating in sensing, which is used as a guarantee for acting honestly. The deposit can be withdrawn only if the sensor uploads the sensing result which is consistent to the cooperative sensing result.
- N_1: The maximum number of sensors needed in cooperative sensing. If there are more than N_1 sensors who apply to participate in cooperative sensing, those with N_1 highest trust value will be selected.
- TV_{thr}: the minimum trust value of a sensor is needed to participate in cooperative sensing.

The functions in CSC are introduced as follows.

- $SensorRegister(\cdot)$: The inputs to this function are address $Addr$, deposit dpt and trust value TV of the node who invokes this function. This function will decide whether this node can register successfully by checking its trust value.
- $CheckRegisterQuality(\cdot)$: This function is used to check node's quality of participating in sensing by checking the parameters passed into it, i.e., node's trust value TV_i and deposit $deposit_i$. For new users, participating in sensing is the only way to improve their trust value. This function offers users a way to convert their tokens to additional trust value, which enables the users whose trust value is under threshold to participate in sensing by paying more deposit.
- $Fusion(\cdot)$: The input to this function is $msgList$, which is a list consisting of all the sensing results from the registered sensors. This function outputs the final sensing result according to the predefined fusion rule.
- $UploadSensingData(\cdot)$: The inputs to this function are msg and $RSIG$, which are the sensing packets in (10) and the ring signature of the node, respectively. This function is invoked by legal sensors to upload their sensing result.

The pseudo-code of the CSC is summarized in Algorithm 2.

Algorithm 2 Cooperative Sensing Contract

```
1: function INIT(TV_thr, d_s, N_1)
2:     require(msg.sender==ContractOwner);
```

initialize TV_{thr}, d_s, N_1;
end function
function SENSORREGISTER($Addr, dpt, TV$)
    $sensor_{num} \leftarrow 0$;
    if CHECKREGISTERQUALITY(TV_i, $addr_i$) **then**
        $sensorMap \leftarrow sensor_i$;
        $sensor_{num} \leftarrow sensor_{num} + 1$;
        EMIT event("Registration success!");
    else
        EMIT event("Registration failed, please checkout the trust value and deposit.");
    end if
    if $sensor_{num} > N_1$ **then**
        Select top N_1 sensors according to trust value;
    end if
end function
function FUSION($msgList$)
    Decision fusion in majority rule;
    return Cooperative sensing result;
end function
function CHECKREGISTERQUALITY(TV_i,$deposit_i$)
    if $deposit_i > d_s$ **then**
        $TV_i^{'} \leftarrow convert(deposit_i - d_s)$
        if $(TV_i + TV_i^{'}) > TV_{thr}$ **then**
            if $totalNum < N_1$ **or** $TV_i > lowestTV$ **then**
                Register Successful;
            end if
        else
            Register Failed;
        end if
    end if
end function
function UPLOADSENSINGDATA(msg, $RSIG$)

35: **if** $RSIG$ is legal **then**
36: $msgList \leftarrow msg$;
37: EMIT event("Upload successfully!");
38: **else**
39: EMIT event("Upload failed, illegal sensor!");
40: **end if**
41: **end function**

Spectrum Auction Contract (SAC): The parameters in SAC are as follows.

- CSC_{id}: The identity of CSC, which is used to identify the connection between SAC and CSC since every SAC is associated with a particular CSC.
- N_2: The maximum number of bidders, which is used to prevent too many nodes from sending bidding message.
- T_{self-d}: The time that SAC conduct the self-destruct operation to release memory.
- d_a: The amount of tokens that bidders need to deposit before the final sensing result is released.

The functions in SAC are introduced as follows.

- $BidderRegister(\cdot)$: Bidders invoke this function to register in SAC.
- $Commit(\cdot)$: The input to this function is $bldBid$, which is the commitment of $bidder_i$. Note that since the account balance is transparent, nodes can identify others' bid by checking their balance [43]. Therefore, everyone makes several commits in order to prevent others from inferring your bid based on changes of their account balance.
- $Reveal(\cdot)$: The inputs of this function include Dps, $Bools$ and $RNDs$. Dps is the list of deposit (Dp) the bidders make; $Bools$ is the list of boolean values which indicate whether these bids are valid or not; $RNDs$ are random values to make the hash of $\{Dp, Bool, RND\}$ hard to guess. A $Commit$ is the hash of these three parameters, i.e., $Commit = \mathcal{H}\{Dp, Bool, RND\}$. $bidsList[msg.sender]$ is the list of the invoker's commits. By comparing the reveal messages and the commitments, this function can identify which bids are revealed correctly. Valid bids are added together as the bidder's total bid, invalid but correctly revealed bids are allowed to be withdrawn, bids that are not correctly revealed will not be returned.
- $Win(\cdot)$: The input to this function is $bidderList$, which is the list of all valid bids. Here we adopt the second-price sealed-bid auction for better economic profit [44]. Accordingly,

this function selects the bidder with the highest bid as the winner, who only needs to pay the second highest bid.

- $EndOfAuction(\cdot)$: SAC will execute self-destruct operation at the preset time T_{self-d}.

The pseudo-code of SAC is shown in Algorithm 3.

Algorithm 3 Sealed Spectrum Auction Contract

$Bid \leftarrow bldBid_i, msg.value$;
$bidsList \leftarrow mapping(address => Bid[])$;
$refund \leftarrow 0$

function COMMIT($bldBid$)
    $bidsList[msg.sender].push(bldBid)$;
end function

function REVEAL($Dps, Bools, RNDs$)
    $len \leftarrow bidsList[msg.sender].length$;
    while $len > 0$ **do**
        $Commit \leftarrow \mathcal{H}\{Dps[len], Bools[len], RNDs[len]\}$;
        if $Commit! = bidsList[msg.sender][len]$ **then**
            EMIT event("Illegal Reveal!");
            Continue;
        else if $Commit == bidsList[msg.sender][len]$ **and** $Bools[len] == true$ **then**
            $refund \leftarrow refund + Dps[len]$;
        else if $Commit == bidsList[msg.sender][len]$ **and** $Bools[len] == false$ **then**
            withdraw invalid but reveal correctly bid;
        end if
        $len \leftarrow len - 1$;
    end while
end function

function WIN($bidderList$)
    return Bidder with the highest bid;
end function

28:

29: **function** ENDOFAUCTION(T_{self-d})

30: Execute self-destruct operation;

31: **end function**

B. The DSA Protocol

In this part, we illustrate our proposed blockchain-based DSA protocol. To make it clearer, we elaborate this protocol in Fig. 5.

- Phase 1 (*Spectrum Sensing Request*): To ask for the channel state, SUs need to send a request message to Task Issuer (TI) through the control channel. TI is played by the node whose trust value is the highest at present. The role of TI is set to prevent multiple contracts from appearing in the network, which results in that users may participate in different contracts. TI will then creates and deploys the CSC and corresponding SAC in the blockchain.
- Phase 2 (*Smart Contracts and Nodes Register*): The CSC and SAC are instantiated when TI issues the transaction

$$CSC = \{CSC_{id} \mid T_{ddl} \mid N_1 \mid TV_{thr} \mid Fusion(\cdot) \mid d_s\} \tag{13}$$

with the signature $Sig_{TI}(STC)$, and the transaction

$$SAC = \{CSC_{id} \mid SAC_{id} \mid N_2 \mid T_{self-d} \mid Win(\cdot) \mid d_a\} \tag{14}$$

with the signature Sig_{TI}=(AC). To register to CSC, a sensor issues the transaction

$$Deposit_{CSC_{id}} = \{pk_i \mid TV_i \mid CSC_{id} \mid d_s\} \tag{15}$$

with the signature $Sig_{sk_i}(Deposit_{CSC_{id}})$ to make deposit. Similarly, a SU who intends to participate in SAC makes its deposit by issuing

$$Bty_{SAC_{id}} = \{pk_i \mid TV_i \mid SAC_{id} \mid d_a \mid \mathcal{H}(bid_i, RND_i)\} \tag{16}$$

with the signature $Sig_{sk_i}(Bty_{SAC_{id}})$. The $\mathcal{H}(bid_i, RND_i)$ denotes the bidding commitment this SU makes for the sealed auction.

- Phase 3 (*Players selection*): Players (including sensors and bidder) are selected based on $SensorRegister(\cdot)$ and $BidderRegister(\cdot)$, respectively. Finally, an event will be triggered to provide an appropriate notification to every selected nodes.

- Phase 4 (*Sensing phase*): Sensors upload the sensing data by issuing the transaction

$$msg_{tx} = \{CSC_{id} \mid \mathcal{H}(msgID) \mid TS \mid SR \mid loc\} \tag{17}$$

with the ring signature generated by (12) before T_{ddl}, and TS denotes timestamp, SR denotes the sensing result. In this process, we use the ring signature to protect the SU's identity. Meanwhile, these sensors need to make commitments mentioned in Section V-D by sending the transaction

$$Commit_i = \{CSC_{id} \mid \mathcal{H}(SR, RND_i, msgID) \mid pk_i\} \tag{18}$$

with $Sig_{sk_i}(Commit_i)$. After the sensing deadline, data fusion is done, and the final sensing result will be published in the blockchain network.

- Phase 5 (*Auction if necessary*): If the cooperative sensing result indicates that the corresponding spectrum bands are idle, the SAC will be invoked. Users who have been selected will reveal their bids for the spectrum by sending bids and random numbers. Then the winner will be published on the blockchain.
- Phase 6 (*Trust value updating*): In this phase, the trust value of each node is updated using equation (5) by miners in the network. It is worth mentioning that the trust value updating process does not compromise the decentralization of the system because the smart contract is decentralized.

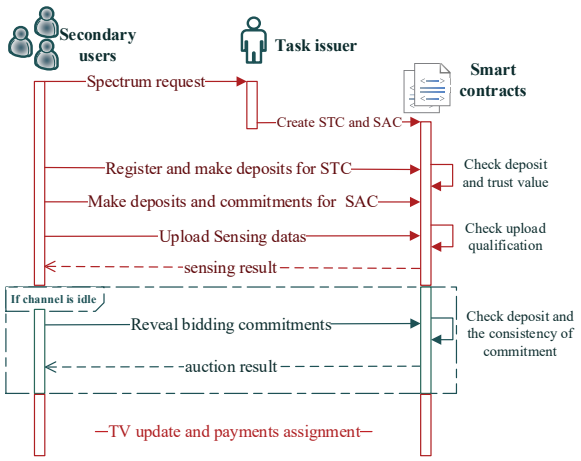


Fig. 5: The workflow of our proposed blockchain-based DSA.

VII. Implementation and Performance Analysis

In this section, we first introduce and analyze the workflow of our designed smart contracts, and then we analyze the performance of PoT consensus mechanism on different type of nodes. Next, we analyze the trust evaluation mechanism to improve the accuracy of cooperative spectrum sensing. Finally, we provide the analysis of the incentive mechanism and security issues of our system.

A. Implementation of Smart Contract

We implement the proposed smart contracts using Solidity [17] in the Ethereum virtual machine (EVM). The smart contracts are compiled and deployed in Remix IDE [45] which is used for testing. After that, nodes in the blockchain network can send transactions to deploy the smart contracts and invoke functions of them. In the cooperative sensing phase, five sensors are considered and the information of their accounts in the blockchain network are listed in table II. In the following, we describe the implementation procedures in detail.

1) *Smart Contract Preparation*: The smart contract is created and deployed by TI, i.e., the fifth account in table II. The relevant parameters and functions which need to be predefined are set as follows: N_1 is set as 3, the trust value threshold TV_{thr} is set as 900 where we magnify the trust values by 100 times and make they as integers, since fixed point numbers are not fully supported by Solidity yet. d_s is set as 100 wei,
2) *Smart Contract Deployment*: Fig.6 shows the details about the deployed smart contract. The field “from" is the account address of TI. The field “decoded input" shows all of our preset parameters.
3) *Sensor Registration*: All the five sensors send their registration request to the smart contract. Then, three sensors are selected by the function $SensorRegister(\cdot)$.
4) *Sensing Results Fusion*: The function $Fusion(\cdot)$, where the majority rule is selected, is used to fuse the sensing results from legal sensors.

If the spectrum is detected as idle, the auction phase initiates. In this phase, we consider two bidders: each bidder has one true bid and one false bid. The related information is listed in Table III. The first bidder has two bids: one is a true bid with 100 wei, the other one is a false bid with 200 wei. The second bidder also has two bids: one is true with 150 wei, the other is false with 300 wei.

TABLE II: Related information of sensor accounts

Account addresses	Trust value	Sensing Result
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4	0.91	0
0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2	0.92	1
0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db	0.87	1
0x78731D3Ca6b7E34aC0F824c42a7cC18A495cabaB	0.93	0
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4	0.94	1

Fig. 6: Initialization of Smart Contract

1) *Commit*: Bidders invoke $Commit(\cdot)$ function to make commitments. Fig. 7 shows the log of commitment made by bidder 2. There are address and bid information about this commit, thus others can know that bidder 2 make a bid with 150 wei, but they can not identify whether this is a real commit or not.
2) *Reveal*: After the commit phase, bidders invoke the function $Reveal(\cdot)$ to reveal its commitments. The field "decoded input" in Fig. 8 shows the whole information about this commit. The smart contract will automatically verify the information uploaded in the two phase and the final bidding result can be accessed by all nodes in the system.

TABLE III: Related Information of Bidders

Bidder address	Bid	isReal	Commit
0xAb84...5835cb2	100wei	yes	0x591a291ad67...1b62c96a2092
	200wei	no	0x4871a30b671...4d08cfaa8bc1
0x4B20...2C02db	150wei	yes	0x4e6a24e35c7be...23643d68efa
	300wei	no	0x0c98edecac...74d24d971ec88

Fig. 7: The commit made by bidder 2

B. Performance Analysis of The Proposed PoT Consensus Algorithm

In this part, we evaluate the performance of the proposed PoT consensus algorithm. We first discuss how to set the initial mining difficulty β_0 at the beginning when every node's trust value is 0. We simulate the mining process in our personal computer as a reference to explain how to select a suitable mining difficulty. We use a given string to represent the transactions of a block. The result is shown in Fig. 9. The running time increases exponentially when the number of leading zeros is larger than 20. Supposing that the *hashrate* of the IoT devices is similar to that of our personal computer, the interval between two blocks T_0 is about 1 second, then the mining difficulty should be less than 18 leading zeros, the β_0 is about $2^{18} = 262144$.

It's worth mentioning that even under the same mining difficulty and experimental environment, the mining time may be different. For example, given the mining difficulty where 32

Fig. 8: The reveal made by bidder 2.

leading zeros are needed in the target hash, it sometimes take tens of seconds, and sometimes take more than an hour. In order to eliminate the impact of luck, we convert the mining difficulty into the *expected mining cost* and evaluate the expected mining cost instead of running time. Theoretically, assuming that the output of each hash calculation is unpredictable, each time when we increase the leading zero by one, the success probability of each hash trial will be halved. The trust value is associated with the mining difficulty. Therefore, we first convert the trust value into the number of leading zeros, and then the number of leading zeros is converted into the expected mining cost proportionally. Simulation is conducted for 1000 time slots, and the *average expected mining cost* of every type of nodes is calculated in Fig.11. To be specific, we simulate a network with 20 nodes and 4 types of nodes are considered. *Rnode* means a node with high performance and always act honestly; *UAnode* means a node with high performance and act honestly, however it participates in sensing infrequently. *OOnode* is a node who conducts malicious behavior periodically. We assume that *OOnode* will act maliciously after every two normal sensing rounds. The *Lnode* is a node who randomly provides binary sensing result without sensing. There are 12 *Rnodes* with probability of detection $p_d = 0.90$ and probability of false alarm $p_f = 0.15$ [46]; 3 *OOnodes* whose $p_d = 0.90$ and $p_f = 0.15$; 3 *Lnodes* with $p_d = 0.5$ and $p_f = 0.5$; and 2 *UAnodes* with $p_d = 0.90$ and $p_f = 0.15$. Besides, it is assumed that they only

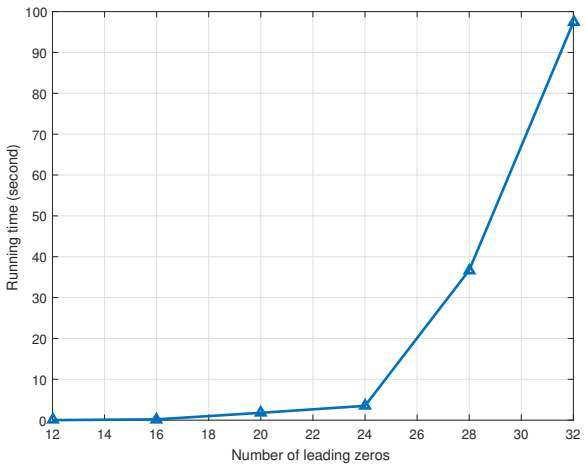


Fig. 9: Running time (second) with increasing number of leading zeros

have a 50% chance to sign up in CSS each sensing round.

Fig. 10 shows the curve of trust values of different types of nodes over time. It can be seen that the trust values of Rnodes are significantly higher than the trust values of other types of nodes. Therefore, the mining difficulty required by Rnodes will also be lower than that of other types of nodes. In Fig. 11, we can see that the mining cost of *Rnodes* is only one third of other nodes. Thus, they can be three times faster than PoW in the mining process, where the mining difficulty is the same for all types of nodes. This proves the effectiveness of our proposed consensus algorithm.

C. Performance of Cooperative Sensing

In this part, we evaluate the performance of cooperative spectrum sensing with the proposed trust evaluation mechanism. In traditional blockchain network, there is no trust among nodes hence anyone can record the sensing information into blockchain, including nodes with poor performance or malicious behavior. However, due to the introduction of the trust evaluation mechanism, the cooperative sensing contract can exclude bad nodes according to nodes' trust value, thus the system's sensing performance can be improved. For comparison, we consider 3 kinds of selection schemes to select the candidate nodes for the cooperative sensing. (i) random selection scheme; (ii) select according to register time; (iii) select according to sensor's trust value. We simulate a 20 nodes network with the same setup in VII-B. It can be seen from Fig.12 that when the number of needed sensors N_1 is small, p_d and p_f of cooperative sensing in the

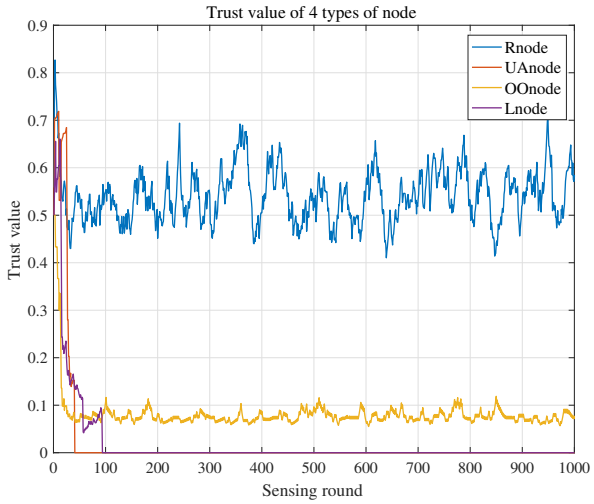


Fig. 10: Curve of trust value

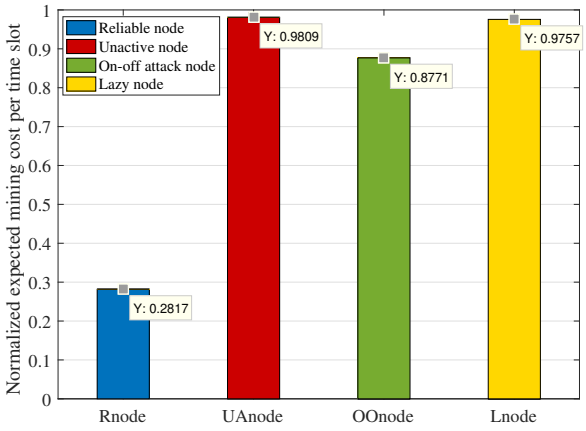
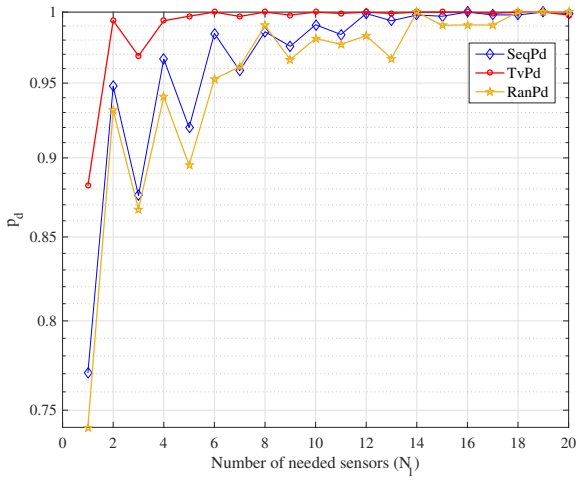
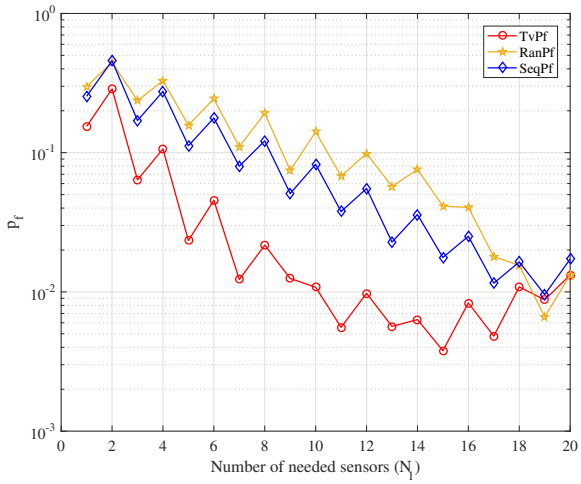


Fig. 11: Expected mining consumption of different nodes

first two schemes which do not take advantage of trust value perform worse than that of the last selection scheme. This indicates that our proposed trust value mechanism can effectively improve the cooperative sensing performance of the system. Moreover, in the last selection scheme, p_d is close to 1 when N_1 is about one fourth of all network nodes. The fewer nodes involved, the smaller the total monetary reward that the network needs to pay to sensors, and the smaller the economic burden for spectrum buyers. As N_1 grows, the performance of different schemes will gradually close, since most of the network nodes (including bad nodes) will participate in cooperative sensing.



(a) P_d of cooperative sensing



(b) P_f of cooperative sensing

Fig. 12: Performance of cooperative sensing under three selection schemes

D. Incentive Mechanism Analysis

Since both sensing and mining will consume SU's computing power, an incentive mechanism is needed to reward nodes for their work. The users who upload a sensing result that is consistent to the final cooperative sensing result will be rewarded with R_s tokens; The users who successfully mine will be rewarded with R_m tokens. The tokens can be used in auction to bid for spectrum resources. In implementation, the tokens rewarded for accurate spectrum sensing and successful mining, i.e., R_s and R_m, need to be designed based on the evaluation of computation consumption

of spectrum sensing and mining.

This incentive mechanism can not only encourage the nodes to participate in spectrum sensing, but also encourage them to behave honestly and accurately in spectrum sensing. This is because, on the one hand, an honest and accurate sensing node is more likely to derive a sensing result that is consistent to the final cooperative sensing result and obtain tokens rewarded for spectrum sensing. On the other hand, with the proposed consensus algorithm, the node with a higher trust value is easier to mine successfully so that they have a higher probability of obtaining tokens for mining. On the contrary, the dishonest nodes will be discouraged since they are less likely to obtain rewards since they need to spend more computing resources on mining.

E. Security Analysis

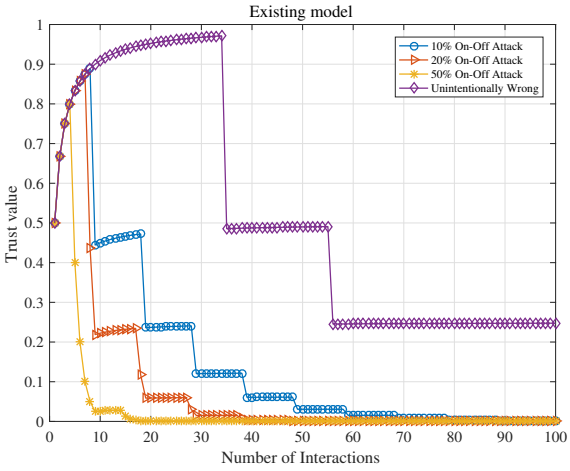
- *Punish rules*: Firstly, PoT is highly depends on trust evaluation mechanism. If a node behaves dishonestly, its trust will drop because of the negative factor $N_{i,w}(n)$ in Eq. (1). Secondly, the deposit mechanism is also a part of our punish rules. If a node behave maliciously in sensing or auction process, the corresponding deposit will be confiscated.
- *Distributed Denial of Service (DDoS) Attack*: The DDoS attack here means that malicious users try to make the sensing service unavailable to other users. Our system is resist to this attack since we adopt the deposit mechanism. Thus, under our scheme, the cost of launching large-scale DDoS attack is very high since the attackers need to obtain lots of tokens in order to launch the attack.
- *Sybil Attack*: One node can create many accounts in public blockchain. However, the initial trust of each account is zero, if new accounts want to compete with high-trust accounts in being selected into sensing group or mining, they must accumulate enough trust through providing correct sensing results. In this way, we increase the cost of Sybil attacks.
- *Collusion Attack*: Malicious nodes must undergo a process of trust accumulation to become a trusted user, during which they need to consume resources and under the risk of losing deposit. However, collusion in sensing cannot make them dominate spectrum directly, and collusion in mining also cannot launch forking attack. Hence, nodes do not have sufficient motivation to accumulate high trust and then launch such kind of attack.
- *Spoofing Attack*: Spoofing attack means someone tries to masquerade others to create forged transactions. Secure Elliptic Curve Digital Signature Algorithm (ECDSA) [47] used in our

blockchain can prevent this attack on the premise that attack does not have the user's private key.

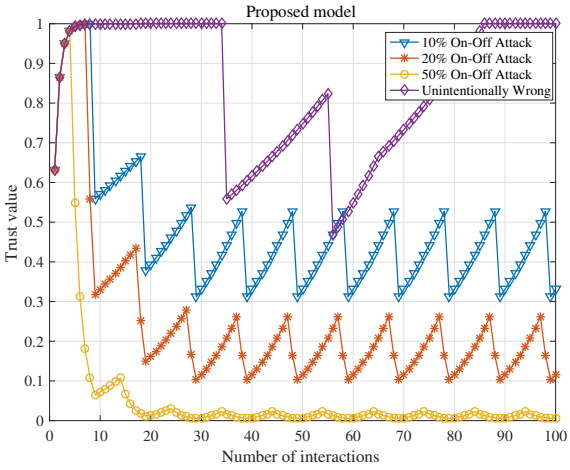
- *Free-riding Attack (lazy node)*: Free-riding attack means lazy users may directly copy others' sensing results at the phase of uploading sensing data. Firstly, there is no motivation for sensors to submit the sensing result before sensing deadline. Thus, when the lazy nodes get the sensing result, it is difficult for them to repack the sensing result and submit before the deadline. Secondly, even if a few sensors submit sensing results in advance, the connection between a user's identity and sensing data is cut by ring signature, thus the lazy users cannot determine the owner of the sensing data, and thus the credibility of data cannot be guaranteed. Thus, it is no better than submit a sensing result randomly.
- *On-off Attack*: On-off attack means that a node performs malicious behaviors periodically, and pretends to be a normal node who "unintentionally" sends sensing result incorrectly from time to time. If a trust management mechanism satisfy the condition that the dropping rate of trust value is larger than its increasing rate, it is considered to be resist to the on-off attack [38]. However, the model in [38] may cause the trust value drop a lot even the misbehavior is unintentional, since the increasing rate is much smaller than dropping rate. In our proposed model, as show in Fig.13, the unintentional mistakes made by sensors can be compensated by making right sensing decision. When the parameters τ and η satisfy $\rho > \frac{\eta}{1-e^{-\eta}} - \eta$, our proposed trust evaluation mechanism is considered to be resist to the on-off attack, the proof is given in Appendix A.

VIII. Conclusion

In this paper, we have proposed a blockchain-based dynamic spectrum sharing protocol. This protocol mainly consists of three parts: the first part is the trust evaluation mechanism, which is designed to evaluate the credibility of nodes; the second part is the PoT consensus algorithm, which makes the mining difficulty of malicious nodes greatly increased. The combination of a node's trust value and its mining difficulty can motivate it to behave honestly; the third part is the privacy protection mechanism, in which we combine the ring signature and the commitment scheme to solve the problem of privacy leakage in the process of cooperative spectrum sensing. Finally, a prototype of our proposed smart contracts has been implemented, and the performance of PoT consensus algorithm and the improvement in cooperative sensing has been analyzed.



(a) Existing model



(b) Proposed model

Fig. 13: Trust value of different types of nodes under two model

Security analysis of the system show that our framework can resist many types of attacks which are common in trust management mechanism and blockchain based IoT systems.

APPENDIX A
PROOF OF THE RESISTANCE OF ON-OFF ATTACK

Proposition 1. *The proposed trust model given in (2) is resistant to on–off attack when* $\rho > \frac{\eta}{1-e^{-\eta}} - \eta$.

Proof. Let $f(N_r, N_w) = e^{-\rho \cdot N_w} \cdot \left(1 - e^{-\eta \cdot N_r}\right)$ denotes the trust value update function in (2). Because when $N_r = 0$, $f(N_r, N_w)$ can only increase, this situation is not discussed here. At any other points (N_r, N_w) in this function, the decreasing rate should be larger than the increasing rate, that is:

$$\left\| \frac{\partial f}{\partial N_w} \right\| > \left\| \frac{\partial f}{\partial N_r} \right\|, \tag{19}$$

where

$$\left\| \frac{\partial f}{\partial N_w} \right\| = \rho \cdot e^{-\rho \cdot N_w} \cdot \left(1 - e^{-\eta \cdot N_r}\right) \tag{20}$$

and

$$\left\| \frac{\partial f}{\partial N_r} \right\| = \eta \cdot e^{-\rho \cdot N_w} \cdot \left(e^{-\eta \cdot N_r}\right), \tag{21}$$

then we have

$$\left\| \frac{\partial f}{\partial N_w} \right\| > \left\| \frac{\partial f}{\partial N_r} \right\| \Leftrightarrow \frac{\rho}{\eta} > \frac{e^{-\eta \cdot N_r}}{1 - e^{-\eta \cdot N_r}} \tag{22}$$

which can also be denoted as:

$$\left\| \frac{\partial f}{\partial N_w} \right\| > \left\| \frac{\partial f}{\partial N_r} \right\| \Leftrightarrow \frac{\rho}{\eta} > \frac{1}{1 - e^{-\eta \cdot N_r}} - 1 \tag{23}$$

When $N_r \geq 1$,

$$\frac{1}{1 - e^{-\eta \cdot N_r}} - 1 \leq \frac{1}{1 - e^{-\eta}} - 1, \tag{24}$$

Therefore, as long as it is satisfied $\frac{\rho}{\eta} > \frac{1}{1-e^{-\eta}} - 1$, the above inequality holds. □

REFERENCES

[1] G. Forecast, "Cisco visual networking index: global mobile data traffic forecast update, 2017–2022," *Update*, vol. 2017, p. 2022, 2019.

[2] P. Kolodzy and I. Avoidance, "Spectrum policy task force," *Federal Commun. Comm., Washington, DC, Rep. ET Docket*, vol. 40, no. 4, pp. 147–158, 2002.

[3] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan, "Cooperative spectrum sensing in cognitive radio networks: A survey," *Phy. Commun.*, vol. 4, no. 1, pp. 40–62, 2011.

[4] J. Wang, M. Li, Y. He, H. Li, K. Xiao, and C. Wang, "A blockchain based privacy-preserving incentive mechanism in crowdsensing applications," *IEEE Access*, vol. 6, pp. 17 545–17 556, 2018.

[5] L. Xiao, Y. Ding, D. Jiang, J. Huang, D. Wang, J. Li, and H. Vincent Poor, "A reinforcement learning and blockchain-based trust mechanism for edge networks," *IEEE Transactions on Communications*, vol. 68, no. 9, pp. 5460–5470, 2020.

[6] G. Liu, H. Dong, Z. Yan, X. Zhou, and S. Shimizu, "B4SDC: A blockchain system for security data collection in MANETs," *IEEE Transactions on Big Data*, pp. 1–1, 2020.

[7] W. Feng and Z. Yan, "MCS-Chain: Decentralized and trustworthy mobile crowdsourcing based on blockchain," *Future Generation Computer Systems*, vol. 95, pp. 649–666, 2019.

[8] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J. Liu, Y. Xiang, and R. H. Deng, "CrowdBC: A blockchain-based decentralized framework for crowdsourcing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 6, pp. 1251–1266, 2019.

[9] M. Yang, T. Zhu, K. Liang, W. Zhou, and R. H. Deng, "A blockchain-based location privacy-preserving crowdsensing system," *Future Generation Computer Systems*, vol. 94, pp. 408 – 418, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X18320909

[10] M. B. H. Weiss, K. Werbach, D. C. Sicker, and C. E. C. Bastidas, "On the application of blockchains to spectrum management," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 2, pp. 193–205, 2019.

[11] E. D. Pascale, J. McMenamy, I. Macaluso, and L. Doyle, "Smart contract slas for dense small-cell-as-a-service," *CoRR*, vol. abs/1703.04502, 2017. [Online]. Available: http://arxiv.org/abs/1703.04502

[12] G. Qiao, S. Leng, H. Chai, A. Asadi, and Y. Zhang, "Blockchain empowered resource trading in mobile edge computing and networks," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–6.

[13] J. Qiu, D. Grace, G. Ding, J. Yao, and Q. Wu, "Blockchain-based secure spectrum trading for unmanned-aerial-vehicle-assisted cellular networks: An operator's perspective," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 451–466, 2020.

[14] T. Ariyarathna, P. Harankahadeniya, S. Isthikar, N. Pathirana, H. M. N. D. Bandara, and A. Madanayake, "Dynamic spectrum access via smart contracts on blockchain," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–6.

[15] S. Bayhan, A. Zubow, and A. Wolisz, "Spass: Spectrum sensing as a service via smart contracts," in *2018 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2018, pp. 1–10.

[16] X. Kang, Y.-C. Liang, and J. Yang, "Riding on the primary: A new spectrum sharing paradigm for wireless-powered iot devices," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6.

[17] Solidity webpage, https://docs.soliditylang.org/en/v0.7.5/.

[18] N. Szabo, "Formalizing and securing relationships on public networks," *First Monday*, vol. 2, no. 9, Sep. 1997. [Online]. Available: https://firstmonday.org/ojs/index.php/fm/article/view/548

[19] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *Advances in Cryptology — ASIACRYPT 2001*, C. Boyd, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 552–565.

[20] O. Goldreich, *Foundations of Cryptography, Volume 2*. Cambridge university press Cambridge, 2004.

[21] K. Kotobi and S. G. Bilen, "Secure blockchains for dynamic spectrum access: A decentralized database in moving cognitive radio networks enhances security and user access," *IEEE Vehicular Technology Magazine*, vol. 13, no. 1, pp. 32–39, 2018.

[22] S. Zheng, T. Han, Y. Jiang, and X. Ge, "Smart contract-based spectrum sharing transactions for multi-operators wireless communication networks," *IEEE Access*, vol. 8, pp. 88 547–88 557, 2020.

[23] X. Fan and Y. Huo, "Blockchain based dynamic spectrum access of non-real-time data in cyber-physical-social systems," *IEEE Access*, vol. 8, pp. 64 486–64 498, 2020.

[24] H. Zhang, S. Leng, and H. Chai, "A blockchain enhanced dynamic spectrum sharing model based on proof-of-strategy," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[25] X. Ling, J. Wang, T. Bouchoucha, B. C. Levy, and Z. Ding, "Blockchain radio access network (b-ran): Towards decentralized secure radio access paradigm," *IEEE Access*, vol. 7, pp. 9714–9723, 2019.

[26] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, no. 1, 2012.

[27] "NXT Whitepaper," 2014. [Online]. Available: https://wiki.nxtcrypto.org/wiki/Whitepaper:Nxt#Proof_of_Stake

[28] I. Bashir, *Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained*. Packt Publishing Ltd, 2018.

[29] "BitShares Delegated Proof of Stake," 2014. [Online]. Available: https://github.com/BitShares/bitshares/wiki/Delegated-Proof-of-Stake

[30] I. Grigg, "Eos-an introduction," *White paper. https://whitepaperdatabase. com/eos-whitepaper*, 2017.

[31] G. Wood, "Polkadot: Vision for a heterogeneous multi-chain framework," *White Paper*, vol. 21, 2016.

[32] M. Salimitari and M. Chatterjee, "An overview of blockchain and consensus protocols for iot networks," *arXiv preprint arXiv:1809.05613*, pp. 1–12, 2018.

[33] M. A. A. Careem and A. Dutta, "Sensechain: Blockchain based reputation system for distributed spectrum enforcement," in *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2019, pp. 1–10.

[34] A. Vosoughi, J. R. Cavallaro, and A. Marshall, "Trust-aware consensus-inspired distributed cooperative spectrum sensing for cognitive radio ad hoc networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 2, no. 1, pp. 24–37, 2016.

[35] H. L. J. Ting, X. Kang, T. Li, H. Wang, and C.-K. Chu, "On the trust and trust modeling for the future fully-connected digital world: A comprehensive study," *IEEE Access*, vol. 9, pp. 106 743–106 783, 2021.

[36] Vitalik Buterin, Thoughts on UTXOs, 2016, http://timmurphy.org/2009/07/22/line-spacing-in-latex-documents/.

[37] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," pp. 1–32, 2014.

[38] X. Kang and Y. Wu, "A trust-based pollution attack prevention scheme in peer-to-peer streaming networks," *Computer Networks*, vol. 72, pp. 62 – 73, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128614002667

[39] D. Boneh, B. Bünz, and B. Fisch, "Batching techniques for accumulators with applications to iops and stateless blockchains," in *Annual International Cryptology Conference*. Springer, 2019, pp. 561–586.

[40] G. Ateniese, B. Magri, D. Venturi, and E. Andrade, "Redactable blockchain – or – rewriting history in bitcoin and friends," in *2017 IEEE European Symposium on Security and Privacy (EuroS P)*, 2017, pp. 111–126.

[41] S. Li, H. Zhu, Z. Gao, X. Guan, Kai Xing, and X. Shen, "Location privacy preservation in collaborative spectrum sensing," in *2012 Proceedings IEEE INFOCOM*, 2012, pp. 729–737.

[42] P. Rogaway and T. Shrimpton, "Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance," in *Fast Software Encryption*, B. Roy and W. Meier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 371–388.

[43] Solidity webpage, https://docs.soliditylang.org/en/v0.7.5/.

[44] Y.-C. Liang, *Blockchain for Dynamic Spectrum Management*. Singapore: Springer Singapore, 2020, pp. 121–146. [Online]. Available: https://doi.org/10.1007/978-981-15-0776-2_5

[45] Remix-Ethereum IDE, https://remix-ide.readthedocs.io/en/latest/.

[46] X. Kang, Y.-C. Liang, H. K. Garg, and L. Zhang, "Sensing-based spectrum sharing in cognitive radio networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 4649–4654, 2009.

[47] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ecdsa)," *Int. J. Inf. Sec.*, vol. 1, pp. 36–63, 08 2001.