Hindawi
Security and Communication Networks
Volume 2022, Article ID 7926057, 12 pages
https://doi.org/10.1155/2022/7926057

Research Article

Fair and Efficient Robust Secret Sharing Scheme against Rushing Adversaries

En Zhang,[1,2] **Renlin Yang**,[1] **Haiju Fan**,[1] **and Leiyong Qin**[1]

[1]*College of Computer and Information Engineering, Henan Normal University, Xinxiang 453007, China*
[2]*Engineering Lab of Intelligence Business and Internet of Things of Henan Province, Xinxiang 453007, China*

Correspondence should be addressed to En Zhang; zhangenzdrj@163.com

Received 28 December 2021; Revised 6 February 2022; Accepted 24 February 2022; Published 2 May 2022

Academic Editor: Wei Feng

Robust secret sharing (RSS) is an extension of secret sharing, which can reconstruct a secret correctly even if any t shares are incorrect. The existing scheme would not effectively achieve fairness. Moreover, even for an optimal scheme, RSS still has the problem that one party is verified by multiple parties, leading to expensive communication costs. In this work, we construct a blockchain-aided RSS scheme that can ensure decentralization and fairness. The central building block for our scheme to handle a rushing adversary is a bulletin board we implement on the InterPlanetary File System. Furthermore, we design a monetary penalty mechanism to impose real penalties on corrupt parties. Each participant either uploads his share correctly or loses his deposit. In addition, our scheme eliminates considerable communication between participants. Compared with the previous schemes, our scheme has a lower communication complexity, which is close to $\widetilde{O}(n)$. We conduct experiments to show the performance of our scheme. To our knowledge, this is the first implementation of a fair RSS scheme. For shares with a length of 128 bits, the time for each participant to execute the verification phase is 470 ms.

1. Introduction

Robust secret sharing (RSS) is an extension of secret sharing that was introduced by the seminal works of Shamir [1] and Blakley [2]. In a t-out-of-n RSS scheme, a dealer splits the secret s into shares and sends these shares to n participants. The secret s can be reconstructed successfully even if t shares are incorrect due to natural damage or being corrupted by an adversary. Robust secret sharing is widely used in practical scenarios, such as the following: (1) data storage: users store their data on multiple cloud servers. The privacy and integrity of these data cannot be damaged even if up to t clouds collude. (2) Secure communication: when users send data over a communication network, the security and integrity of the users' data are still guaranteed even if up to t communication channels are controlled by an adversary.

Recently, research on the share size has made great progress, and the known optimal share size is close to $\widetilde{O}(\kappa)$. However, existing RSS schemes either require a trusted third party or require multiple interactions in the reconstruction phase, which causes expensive communication costs. In addition, previous RSS schemes cannot impose practical penalties on the corrupt party and cannot effectively achieve fairness. In this study, we solve the problems of centralization and fairness, and we improve the communication complexity to $O(n)$.

1.1. Related Work. Secret sharing was initially introduced by Shamir [1] and Blakley [2]. Many works consider stronger concepts based on the secret sharing, such as hierarchical secret sharing and threshold secret sharing [3–13]. Rabin et al. [14] first proposed the concept of robust secret sharing in 1989. In a robust secret sharing scheme, the number of corrupt parties is less than half of all participants. Therefore, the maximum corruption in a RSS scheme is set to $n = 2t + 1$. The failure probability of the reconstruction is negligible when the relationship between n and t becomes $n/3 \le t \le n/2$ [15]. In 1997, Carlo et al. [16] analyzed the lower bound of the share size and the probability of deception in an RSS scheme.

Serge Fehr [15] proposed an RSS scheme and proved that the average bit length of a share is lower-bounded by the bit length of the secret. Compared with [14], Serge Fehr improved the message authentication code (MAC) algorithm to reduce the share size to $m + O(n + \kappa)$, and the MAC algorithm that they used can verify the correctness of shares; however, their scheme cannot fight a rushing adversary. To solve this problem, Cevallos [17] constructed a new RSS scheme that can maintain security against a rushing adversary. Furthermore, the share size of their scheme is close to that of [15]. However, each participant needs to verify the shares of all other participants, which causes a high communication cost during the reconstruction phase. In addition, Cevallos reduced the length of the tag in the message authentication code; therefore, the probability of corrupt parties colluding together to pass the MAC authentication is not negligible.

Later, Bishop et al. [18] designed an essential optimal RSS scheme, and the share size of this scheme is $m + O(\kappa(\log^4 n + (\log^3 n)\log \kappa))$. The scheme can successfully reconstruct the secret in the case of the maximum corruption. The basic idea behind their reconstruction phase is the minimum graph. The scheme used a graph algorithm to make a judgement on the participants, which can verify whether they are good or bad. The graph recognition algorithm can eventually collect the shares of all good participants. Allison Bishop innovatively eliminated the linear relationship between the share size and the number of participants. However, the scheme is not able to fight a rushing adversary that can choose incorrect authentication keys freely. Fehr and Yuan [19] proposed a new scheme that can fight a rushing adversary. In addition, the share size of this scheme is close to $m + O(\kappa n^{\epsilon})$. The participants are required to submit shares in multiple rounds, which can limit the relationship between correct and incorrect shares. Unlike [18], participants do not need an authentication key for authentication. Each participant can verify whether his neighbour is good or bad. The algorithm eventually outputs a set that contains the shares of all good participants. In this way, the Solomon algorithm has enough redundant codes to reconstruct a secret efficiently. However, the algorithm has a high probability of success only if it starts with an honest party. In 2020, Fehr and Yuan [20] designed a new scheme that has the optimal share size, can guarantee safety against rushing adversaries, and can also run in polynomial time.

Regarding the share size, Bishop et al. [21] introduced a nearly optimal scheme that optimizes the share size to $m(1 + O(1)) + O(n)$. In short, the scheme first fixes the original secret with the detection of algebraic manipulation (AMD) code [22]. The algorithm outputs a short list containing the correct keys, which allows the reconstruction phase of the scheme to efficiently identify error messages. Then, the scheme used the folded Reed–Solomon codes [23] to construct error-correcting codes. Compared with the existing Solomon code, the folded Reed–Solomon codes can construct error-correcting codes more efficiently. Unfortunately, the scheme cannot fight a rushing adversary. Manurangsi et al. [24] presented a new scheme that can fight a rushing adversary. The share size of the scheme is further optimized to $m + O(\lambda \log n(\log n + \log m))$. The reconstruction phase of the scheme is a two-step process. In the first round, each participant publishes his share. In the second round, the scheme uses a private MAC to verify whether each party has changed his share. If the party does not modify his share, the scheme labels him as good. Otherwise, the scheme labels him as bad. The reconstruction phase eventually outputs a set that contains all the good parties. Moreover, the scheme uses fixed-point recognition algorithms to make the second round more efficient. Finally, the participants can use this set to reconstruct the secret locally. However, the scheme still has the problem that one party is verified by multiple parties, which causes high communication costs.

We summarize the above schemes, as shown in Table 1. The above schemes either cannot provide security against rushing adversaries or have higher communication complexity. In addition, none of the above schemes can effectively achieve fairness.

1.2. Contribution. In this study, we construct a decentralized and fair RSS scheme based on smart contracts, and the share size of our scheme is close to $m + \widetilde{O}(\kappa)$. The flow of the scheme is shown in Figure 1, and the main contributions are as follows.

1.2.1. The Scheme Is Secure against Rushing Adversaries. We implemented a bulletin board on the InterPlanetary File System (IPFS) to broadcast all tags before the reconstruction phase. Because the tag cannot be changed, our scheme can easily verify whether the party has changed his share.

1.2.2. The Scheme Can Guarantee Fairness in Malicious Model. We implemented a monetary penalty mechanism, which can impose real penalties on corrupt parties. Each participant either uploads his share correctly or loses his deposit. This motivates all parties to be honest and ensures the fairness of our scheme.

1.2.3. The Communication Complexity of This Scheme Is Close to $O(n)$. We designed a smart contract with the functions of collecting, verifying, and broadcasting shares. Compared with the scheme of [19] (Eurocrypt 2019), we have a lower communication complexity.

1.2.4. This Is the First Implementation of a Fair RSS Scheme. We demonstrate the implementation of a fair and decentralized RSS scheme using the blockchain. For shares with a length of 128 bits, the time for each participant to run the verification phase is 470 ms.

1.3. Roadmap. The organization of the study is as follows: in Section 2, the preliminary of message authentication code, Reed–Solomon code, InterPlanetary File System, and Ethereum are introduced. Section 3 introduces our smart contract model, and Section 4 introduces our protocol. We

Table 1: Summary of results in robust secret sharing scheme.

Paper	Share size	Communication complexity	Remark
[1]	$m + O(n\kappa)$	$O(n^2)$	Corruption setting with $t < n/3$
[16]	—	—	Give the lower bounds for RSS
[17]	$m + O(n + \kappa)$	$O(n^2)$	Secure against rushing adversaries
[18]	$m + O(\kappa)$	$O(n\ d)$	For $n = 2t + 1$
[19]	$m + O(\kappa)$	$O(n\ d)$	Secure against rushing adversaries
[20]	$m + O(\kappa)$	$O(n)$ with a third party	Secure against rushing adversaries
[21]	$m + O(\kappa)$	$O(n^2)$	For $n = 2t + 1$
[24]	$m + O(\kappa \log n(\log n + \log m))$	$O(n^2)$	Secure against rushing adversaries

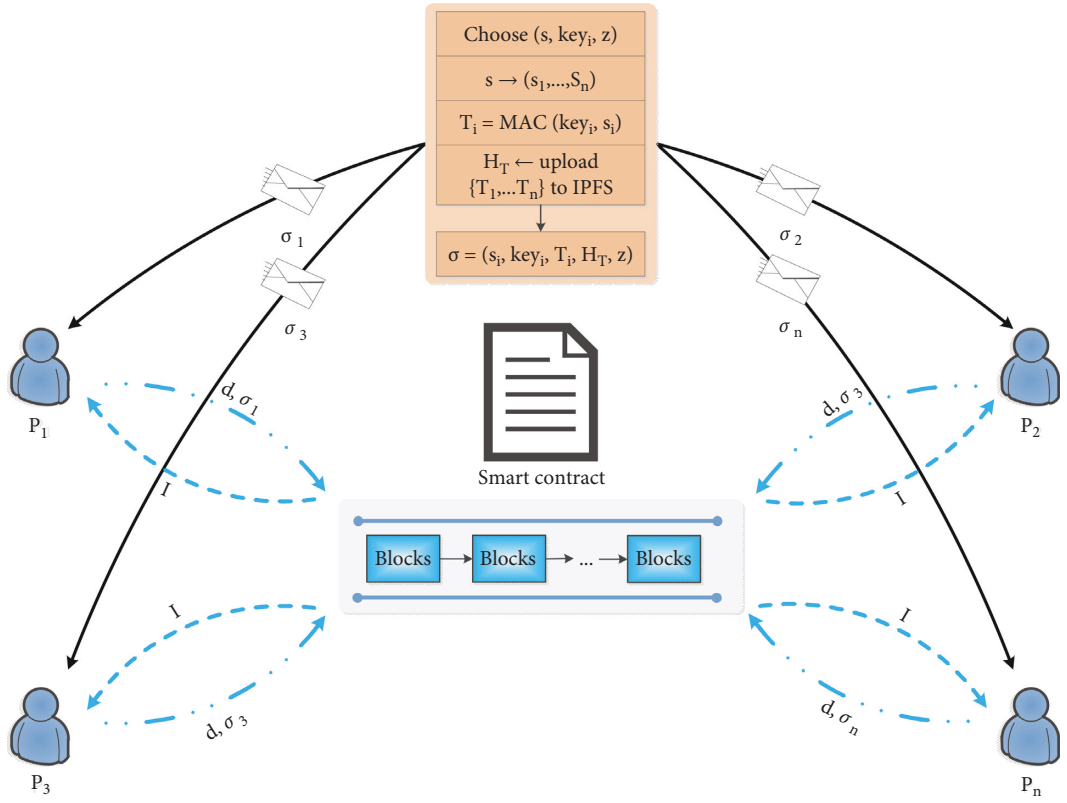


Figure 1: Robust secret sharing scheme based on smart contract.

analyze the security proof of the protocol in Section 5. We did the simulation experiment in Section 6. Finally, we present our conclusion in Section 7.

2. Preliminaries

2.1. Message Authentication Code. The message authentication code (MAC) can guarantee the integrity of the message and detect whether others have tampered with the message. Unconditionally secure MAC was invented by Carter and Wegman [25], and the definition is as follows.

Definition 1. A message authentication code for a finite message space $\mathcal{M}$ composed of a specific function: $\mathcal{T} \leftarrow \mathcal{M} \times \mathcal{K}$. This MAC is secure if and only if it follows: for any $m, m' \in \mathcal{M}$ and $m \neq m'$, any $t, t' \in \mathcal{T}$ and $t \neq t'$, and a negligible function poly(x), there is as follows:

$$\Pr_{k \leftarrow \mathcal{K}}[\mathrm{MAC}(m', k) = t' | \mathrm{MAC}(m, k) = t] \leq \mathrm{poly}(x). \quad (1)$$

The MAC authentication we used in this study is a key-dependent hash-based message authentication code (HMAC). Regarding the security of HMAC, researchers use various methods to attack it, such as collision attack and length extension attacks, but none of them have a good effect. Later, [26] gave a strict security proof about the HMAC and proved the close relationship between the security of HMAC and hash function. If the HMAC has security problems, then the bottom hash function must have security problems. Therefore, theoretically the security of HMAC is proved.

2.2. Reed–Solomon Code. The idea behind the Solomon code can be represented as polynomial, and it relies on an algebraic theory that any k points can construct a polynomial

of degree $k-1$. In the first place, the sender S has a message $m = \{m_1, \ldots, m_k\}$. There is a polynomial $f(x)$ of degree $k-1$ in a finite field, which is constructed with k data points. S picks k values $\{x_1, \ldots, x_k\}$ at random. The polynomial codes the data point according to the corresponding value of each point such as $f(x_i) = y_i$. S sends these values to the receiver R, but these values can be corrupted for some reasons during transmission. Therefore, we added some additional data points called redundant codes. The final number of these data points that S sends to R is actually $n(n > k)$. R starts decoding after receiving these data points. R can compute the polynomial $f(x)$ and reconstruct the raw data as long as R receives sufficient number of values correctly. There are many decoding methods for the Solomon codes, among which the Berlekamp–Welch algorithm [27] can efficiently reconstruct the polynomial $f(x)$ from $\{y_1, \ldots, y_k\}$. The error correction ability of this algorithm is related to the shortest Hamming distance $d = n - k + 1$. The algorithm can correct $(n-k)/2$ errors if the location of the error message is not known before the error correction. The algorithm can correct $(n-k)$ errors if the location of the error message is known before the error correction. Later, Gao [28] used fast Fourier transform (FFT) to greatly improve the decoding efficiency without reducing its error correction capability.

2.3. Blockchain. The blockchain is a distributed shared ledger and database, which was first proposed by Satoshi Nakamoto. The blockchain can store an ever-growing and unmodifiable list of records. These records are called blocks and are linked to previous blocks. Recently, the business models based on blockchain have developed rapidly. Researchers are exploring the blockchain business model in different fields, such as energy and physical products [29]. The development of the blockchain has made smart contracts possible. Blockchain-based smart contracts are computer programs executed by many nodes that can be tracked and immutable. Therefore, the design of smart contract must follow the principle of low cost. To enable smart contracts to perform more complex algorithms, off-chain smart contract frameworks are being developed [30].

2.4. InterPlanetary File System. The InterPlanetary File System (IPFS) is a distributed network transport protocol for storing and sharing files. A user uploads a file to an IPFS node and obtains a hash value H. Other users can use H to download the file directly from the IPFS node. In [31], Benet stated that the IPFS combines distributed hash tables, BitTorrent, Git, self-certified file systems, and the blockchain to save and transfer files. These systems bring significant characteristics to IPFS as follows:

(i) Permanent and decentralized storage of documents.

(ii) Users can use a specific hash value to find a stored file.

(iii) The document modification history can be traced.

These characteristics give the IPFS great potential in the field of cryptography. Compared with the blockchain, the IPFS has a large advantage in storing large files. In this study, we implement a bulletin board on the IPFS that contains the tags of all participants' shares, so that the corrupt party cannot cheat on the tags. Meanwhile, the share size is reduced to $\widetilde{O}(\kappa)$.

2.5. Security Model. The existing secure multiparty computing (MPC) protocol needs a security proof to ensure its security. In this study, ideal and real models are used as the security models of security proofs. The precise definition of the security model was proposed in [32], which argues that there is a trusted third party that can communicate securely with the participants and run the protocol honestly in an ideal world. The scheme is secure if participants' output and view in the real model are indistinguishable from those in the ideal model. In this section, we will provide the exact definition of this model as follows:

(i) Real World: in the real world, the honest party runs the scheme $\prod$ correctly. There is a malicious adversary $\mathcal{A}$ that can corrupt any participants. All message from the corrupt party is determined by the adversary $\mathcal{A}$. In addition, $\mathcal{A}$ can interact with $\prod$ at will to obtain some useful information. The input of the scheme generated by the participants is sent to $\prod$ directly, and the output of $\prod$ is returned to $\mathcal{A}$. Finally, $\mathcal{A}$ outputs $\text{Real}_{\prod,\mathcal{A}}(x_1, \ldots, x_n)$ as the output of the real world.

(ii) (ii)Ideal World: in an ideal world, there is a trusted third party that receives an input from the participant to safely compute function F. A simulator sim can extract the behaviour of an adversary and receive the input of the honest party in the real world. The input of the scheme generated by sim is sent to the function F directly, and the output of the function is returned to *sim*. Finally, *sim* outputs $\text{Ideal}_{F,\text{sim}}(x_1, \ldots, x_n)$ as the output of the ideal world.

3. The Smart Contract Model

Section 3 describes the smart contract model we designed, which includes each time point, the function of each module, and the transaction process of the smart contract, as shown in Figure 2.

A reconstruction scheme consists of two phases: the submission phase and the verification phase. Participants submit shares in two rounds during the submission phase, which limits the relationship between correct and incorrect shares. The verification phase includes tag verification and MAC verification. The tag verification can check whether T_i is correct, and the MAC verification can check whether s_i has been changed. We define some time points that are as follows:

t_{register}: all participants must pay deposit d and register their eligibility to participate in the smart contract before t_{register}. If the participant does not pay the deposit before t_{register}, the scheme will be terminated.

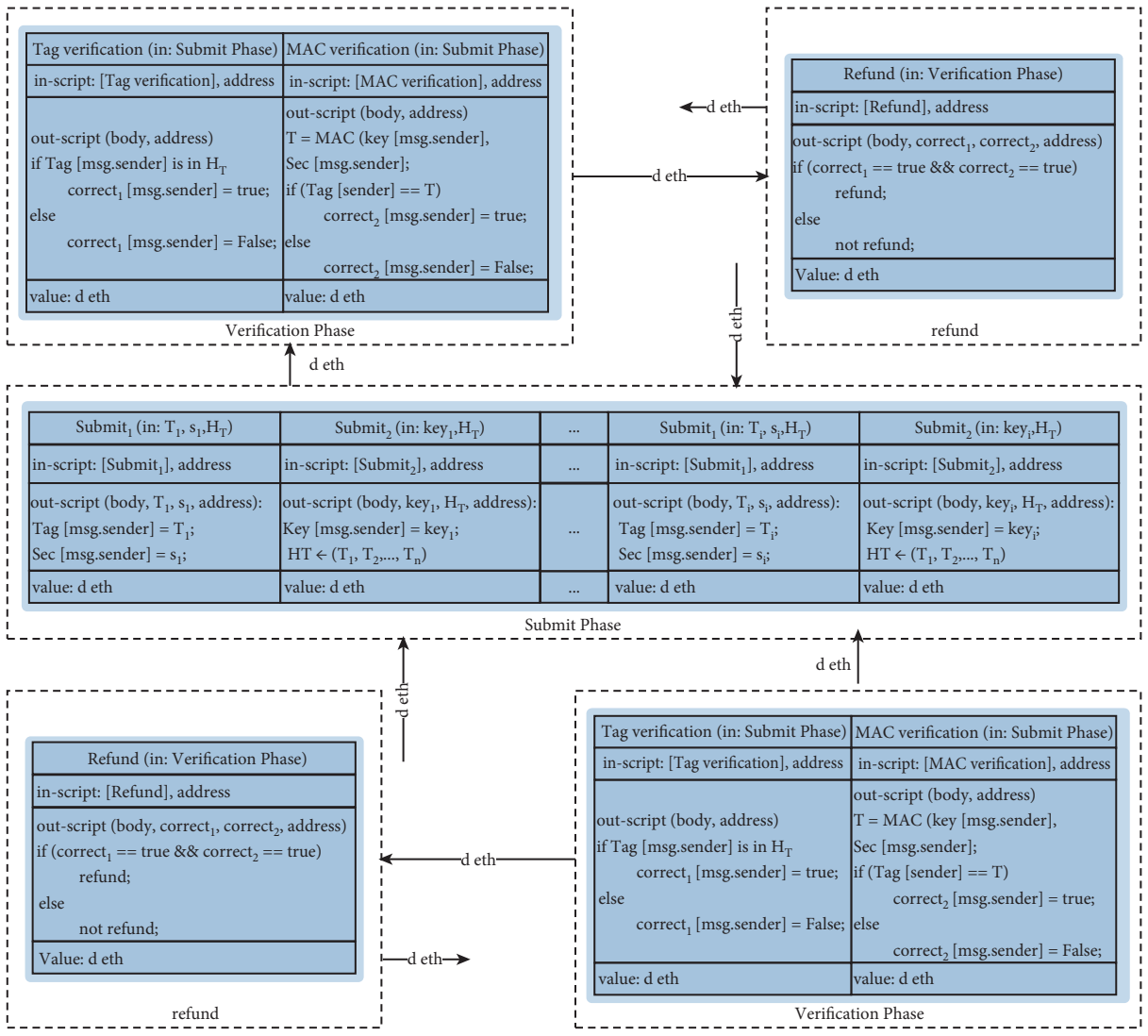


Figure 2: Smart contract model of RSS scheme.

t_{submit_1}: for each $i \in [n]$, P_i must upload (H_T, T_i, s_i) to the smart contract before t_{submit_1}. If P_i does not upload (H_T, T_i, s_i) within the specified time, his deposit will be deducted.

t_{submit_2}: for each $i \in [n]$, P_i must upload key_i to the smart contract before t_{submit_2}. If P_i does not upload key_i within the specified time, his deposit will be deducted.

The description of every phase is as follows.

3.1. Initialization

(1) For each $i \in [n]$, participant P_i has share $(H_T, z, s_i, T_i, \text{key}_i)$ and cannot access others' shares.

(2) The smart contract defines a deposit d. The value of d is greater than the secret s.

(3) P_i pays a deposit and registers to participate in the smart contract until he finds the transaction $\text{RSC} \cdot$ register in the blockchain.

3.2. Submission Phase

(1) $\text{RSC} \cdot \text{submit}_1$: The smart contract broadcasts the transaction address of $\text{RSC} \cdot \text{submit}_1$. P_i submits (s_i, T_i). The smart contract stores the share and address of P_i.

(2) $\text{RSC} \cdot \text{submit}_2$: The smart contract broadcasts the transaction address of $\text{RSC} \cdot \text{submit}_2$. P_i submits key_i. The smart contract stores the share and address of P_i.

(3) All participants wait for smart contract to broadcast the transaction addresses of $\text{RSC} \cdot \text{submit}_1$ and $\text{RSC} \cdot \text{submit}_2$.

(4) If the transaction addresses of $\mathrm{RSC}\cdot\mathrm{submit}_1$ and $\mathrm{RSC}\cdot\mathrm{submit}_1$ do not appear in the blockchain before t_{submit_1} and t_{submit_2}, respectively, the smart contract refunds the deposit.

3.3. Verification Phase

(1) RSC tag verification: the smart contract broadcasts the transaction address of $\mathrm{RSC}\cdot\mathrm{tagverification}$ and verifies whether T_i is in H_T.

(2) RSC MAC verification: the smart contract broadcasts the transaction address of $\mathrm{RSC}\cdot\mathrm{MACverification}$ and verifies whether $T_i = \mathrm{HMAC}(s_i, \mathrm{key}_i)$ is established.

3.4. Broadcast and Refund Phase. After the verification phase, the smart contract can get the set I, which contains all the honest parties and passive adversaries.

The smart contract broadcasts the transaction address of $\mathrm{RSC}\cdot\mathrm{refund}$. For each $i \in [I]$, P_i can use the transaction address to query transaction information. Meanwhile, the smart contract broadcasts the correct shares s_i and returns the deposit to the party P_i.

4. The Robust Secret Sharing Scheme

4.1. The Construction. Let t be an arbitrary positive integer, and $n = 2t + 1$. This scheme randomly selects n interpolation points $\{x_1, \ldots, x_n\}$ in a field $\mathbb{F}$ and $|\mathbb{F}| > n$. In addition, we use a HMAC algorithm with secret $s \in \mathbb{F}$. Assume that the adversary can obtain the profit worth d after breaking the RSS. The detailed steps of the RSS scheme are shown in Figure 3.

4.2. Analysis of the Construction. In the malicious model, there are always adversaries colluding to pass the MAC authentication of the protocol with the wrong shares, which makes the RSS cannot reconstruct the secret. In this section, we use the Chernoff bounds [33] to prove that this probability is small.

We define a set C, which contains all corrupt parties. There are two types of corrupt parties we define. One type is the active corrupt party, and they will provide the wrong shares, which are represented by set A and $|A| = a$. The other type is the passive corrupt party, and they will provide the correct shares, which are represented by set P and $|P| = p$. Obviously, $t = a + p$.

In the case of $p \geq t/2$, even if all active corrupt parties have passed MAC authentication, the Solomon decoding algorithm can still reconstruct the correct secret, so we do not consider this situation. We only pay attention to $\Pr[|A \cap I| > 0]$ in the case of $p < t/2$.

Let X be a random variable and $a \in R$. According to the Markov inequality [34], we have the following:

$$\forall l > 0, \quad \Pr(X \geq a) = \Pr\left(e^{lX} \geq e^{la}\right) \leq \frac{E\left(e^{lX}\right)}{e^{la}}, \tag{2}$$

where $M_X(l) = E(e^{lX})$. We compute the Taylor expansion of $E(e^{lX})$, and assuming it is convergent:

$$\begin{aligned} M_X(l)M_X(l) &= E\left(1 + lX + \frac{1}{2}l^2X^2 + \frac{1}{3}l^3X^3 + \cdots\right) \\ &= \sum_{i=0}^{\infty}\frac{1}{i!}l^iE\left(X^i\right). \end{aligned} \tag{3}$$

The message about distribution can be encoded by $E(X^i)$. We limit the moment generating function of each X_i to derive the Chernoff bounds.

Lemma 1. *Given independent and random variables* $\{X_1, X_2, \ldots, X_n\}$, *where* $X = \sum_{i=1}^{n} X_i$, *then:*

$$M_X(l) = \prod_{i=1}^{n} M_{Xi}(l). \tag{4}$$

Proof.

$$\begin{aligned} M_X(l) &= E\left(e^{lX}\right) \\ &= E\left(e^{l\sum_{i=1}^{n}X_i}\right) \\ &= E\left(\prod_{i=1}^{n}e^{lX_i}\right) \\ &= \prod_{i=1}^{n}M_{X_i}(l). \end{aligned} \tag{5}$$

Before deriving the Chernoff bounds, we use the Bernoulli trial to complete the following lemma. □

Lemma 2. *Given a random variable* Y, *and* $Pr(Y = 1) = p$, $Pr(Y = 0) = 1 - p$, *then*

$$\forall l \in R,\ M_Y(l) = E\left(e^{lY}\right) \leq e^{p\left(e^l - 1\right)}. \tag{6}$$

Proof

$$\begin{aligned} M_Y(l) &= E\left(e^{lY}\right) \\ &= p \times e^l + (1 - p) \times 1 \\ &= 1 + p\left(e^l - 1\right), \end{aligned} \tag{7}$$

where $y = p(e^l - 1)$, and then:

$$M_Y(l) \leq e^{p\left(e^l - 1\right)}. \tag{8}$$

From Lemmas 1 and 2, we start to derive the Chernoff bounds. □

Corollary 1. *Given independent and random variables* $\{X_1, X_2, \ldots, X_n\}$, *and* $X = \sum_{i=1}^{n} X_i$, *where* $Pr(X_i = 1) = p_i$, $Pr(X_i = 0) = 1 - p_i$, *and* $\mu = E(X) = \sum_{i=1}^{n} p_i$, *then:*

Distribute share phase

Local: input $s \in$F, dealer D chooses a random number z with the same length as s and computes $s' = s \oplus z$. D construct a polynomial $f(x)$ with maximum degree t and $f(0) = s'$. D starts to calculate the shares $S = \{s_1 = f(x_1), ..., s_n = f(x_n)\}$. For each $i \in$ [n], D randomly selects the $key_i \in K$ to compute T_i = HMAC (s_i, key_i). D gets the set $T = \{T_1, ..., T_n\}$ and uploads T to IPFS network to obtain the hash value H_T.

For each $i \in [n]$, D sends the shares $\sigma_i = \{H_T, z, s_i, T_i, key_i\}$ to Pi.

Reconstruct share phase

1. For each $i \in [n]$, P_i pays the deposit d and registers the eligibility to take part in the smart contract before $t_{register}$.
2. For each $i \in [n]$, P_i uploads (H_T, T_i, s_i) to the smart contract before $t_{submit1}$.
3. For each $i \in [n]$, P_i uploads key_i to the smart contract before $t_{submit2}$.
4. Smart contract initializes a set I which contains $\{P_1, ..., P_n\}$.
5. For each $i \in [n]$, the smart contract uses H_T to download the set T from IPFS, then verifies whether the T_i is in the set T. If yes, proceed to the next step. Otherwise, the smart contract deducts his deposit d, and deletes P_i in set I.
6. For each $i \in [n]$, smart contract executes the MAC verification algorithm to determine whether T_i = HMAC (s_i, key_i) is established. If yes, proceed to the next step. Otherwise, the smart contract deducts his deposit d, and deletes P_i in set I.
7. For each $i \in [I]$, the smart contract outputs s_i and returns P_i's deposit.

For each $i \in [I]$, The participants can reconstruct the polynomial $f(x)$ locally using the Solomon decoding algorithm such that $f(x_i) = s_i$. If polynomial exists, output $s' = f(0)$. Otherwise, output $\perp$. The participants locally compute the secret $s = s' \oplus z$.

Figure 3: RSS scheme based on smart contract.

$$\forall \delta > 0, \quad \Pr(X \geq (1+\delta)\mu) \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu} \leq e^{(\delta-\delta^2)\mu}. \tag{9}$$

Proof

$$\begin{aligned} M_X(l) &= \prod_{i=1}^{n} M_{X_i}(x) \leq \prod_{i=1}^{n} e^{p_i(e^l-1)} \\ &= e^{(e^l-1)\sum_{i=1}^{n} p_i} \\ &= e^{(e^l-1)\mu}. \end{aligned} \tag{10}$$

Let $a = (1+\delta)\mu$, $l = \ln(1+\delta)$. Applying (2), we can obtain the following:

$$\Pr(X \geq (1+\delta)\mu) \leq \frac{E\left(e^{lX}\right)}{e^{la}} = \frac{e^{l\mu}}{e^{l(1+\delta)\mu}}. \tag{11}$$

When $0 < l < e^l - 1$, we have the following:

$$\Pr(X \geq (1+\delta)\mu) \leq \frac{e^{(e^l-1)\mu}}{e^{l(1+\delta)\mu}} = \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu}. \tag{12}$$

Taking the logarithm of the $ln(e^{\delta}/(1+\delta)^{(1+\delta)})^{\mu}$, we can obtain the following:

$$\mu(\delta - (1+\delta)\ln(1+\delta)). \tag{13}$$

For any $x > 0, a < 1 + x$, there is $ln(1+x) \geq ax/1+x$, and then:

$$\mu(\delta - (1+\delta)\ln(1+\delta)) \leq \left(\delta - \delta^2\right)\mu. \tag{14}$$

□

Theorem 1. *For any positive integer t, any positive integer n = 2t+1, and any HMAC, the (n, t) robust secret sharing scheme has a small probability of failure with:*

$$\Pr(\text{failure}) < \frac{1}{e^{t\text{poly}(x)-1}}. \tag{15}$$

Proof. Consider t corrupt parties as t random variables $X_1, X_2, \ldots X_t$. If X_i passes HMAC authentication, $X_i = 1$. Otherwise, $X_i = 0$. $\Pr(X_i = 1) = \varepsilon, \Pr(X_i = 0) = 1 - \varepsilon$. We can obtain the following:

$$\begin{aligned} \mu &= E(X) \\ &= \sum_{i=1}^{t} \varepsilon = t\varepsilon. \end{aligned} \tag{16}$$

Applying Corollary 1, we have the following:

$$\Pr(X \geq 1) \leq \left(\frac{e^{1/\mu-1}}{(1/\mu)^{1/\mu}}\right) \leq e^{1-(1/t\varepsilon)\mu} = \frac{1}{e^{t\text{poly}(x)-1}}. \quad (17)$$

□

5. Security Proof

In this section, we present the secure analysis of the scheme.

Theorem 2. *This scheme achieves perfect privacy.*

Proof. There is a set $C \subset [n]$ that contains all the corrupt parties, and $|C| = t$. First, we assume that the secret $s \in \mathbb{F}$ is randomly distributed and the distributeshare phase is safely executed locally. Our purpose is to prove that the distribution of $\sigma_{i\in C}$ is not related to s. Here, $\sigma_i = \{H_T, z, s_i, T_i, \text{key}_i\}$.

In our RSS scheme, the secret s can only be reconstructed with more than t participants. For each $i \in [C]$, they cannot reconstruct the secret s even if all the corrupt parties join in the collusion. Therefore, s_i is independent of s. In addition, the values of z_i and key_i are selected randomly, so they are also not related to the secret s. The label T_i is generated by the MAC algorithm to verify the integrity of s_i, so T_i is independent of the shares s_i because of the nature of "privacy over randomness." H_T is a hash value generated by the IPFS, so it is obviously independent of s_i. In summary, the privacy of secret s is guaranteed. □

Theorem 3. *RSS scheme based on smart contract is secure when any t participants are corrupted.*

Proof. The rushing adversary can determine how to change the share of corrupt parties after receiving the share of honest parties. In this study, we implemented a bulletin board on the IPFS to store all tags before the reconstruction phase. The rushing adversary cannot change the tags due to the immutable nature of IPFS. Because the tag cannot be changed, the probability that the rushing adversary passes the share verification phase is negligible if he changes the share. Therefore, our scheme can fight against a rushing adversary.

Let Π denote the RSS scheme based on smart contract and F denote a function based on smart contract. There is a real-world scenario in which the participants can execute an RSS scheme. We built a simulator Sim that can extract a probabilistic polynomial time (PPT) adversary's operation in the real world such that $\text{REAL}_{\Pi,A} \approx \text{IDEAL}_{F,\text{Sim}}$. In addition, the adversary A can corrupt any t participants. First, we think that A cannot obtain any information about the shares from the channel between dealer D and other participants in the share distribution phase. The following section will prove the indistinguishability between the ideal world and the real world.

In the reconstruction phase, our scheme uses authenticated private channels. The simulator Sim can extract the view and input of A in the real world and can also simulate the input of honest participants in the real world. There are a set C that contains all the corrupt parties and a set H that contains all the honest parties. The capabilities of Sim are as follows:

(i) For each $i \in [H]$, Sim accesses the smart contract and initiates the transaction $\text{RSC} \cdot \text{submit}_1$. Sim sends the transaction address to P_i and receives $\{H_T, T_i, s_i, \text{key}_i\}$. Sim uploads these shares to the smart contract. Similarly, Sim sends transaction $\text{RSC} \cdot \text{submit}_2$ to P_i and receives key_i. Sim uploads key_i to the smart contract.

(ii) For each $i \in [C]$, Sim accesses the smart contract and initiates the transaction $\text{RSC} \cdot \text{submit}_1$. Sim invokes A and internally hands A the transaction address as if it was sent by P_i. Sim outputs whatever A outputs. If A does not upload $\{H_T, T_i, s_i\}$ to the *Sim* before t_{submit_1}, the smart contract can keep his deposit. Similarly, Sim invokes A and internally hands A the transaction $\text{RSC} \cdot \text{submit}_2$. Sim outputs whatever A outputs. If A does not upload key_i to the smart contract before t_{submit_2}, the smart contract can keep his deposit.

(iii) For each $i \in [n]$, Sim sends the transaction $\text{RSC} \cdot$ tagverification to the smart contract. If $T_i \notin H_T$, P_i loses his deposit d. Similarly, Sim sends the transaction $\text{RSC} \cdot \text{MACverification}$ to the smart contract. If $T_i \neq \text{MAC}(\text{key}_i, s_i)$, P_i loses his deposit d. If all the above transactions have been executed, then P_i honestly sends the right shares.

In an ideal world, the view of each corrupt party would be the same as that of each corrupt party in the real world due to the open and transparent nature of smart contracts. The privacy of shares is proved. The smart contract is deployed in the blockchain. A cannot tamper with the smart contract and obtain the deducted deposit unless he can construct a new blockchain. Finally, the smart contract outputs the correct share s_i for $i \in |I|$, and the honest parties are able to successfully reconstruct the original secret s using Solomon's algorithm. We can prove that

$$\text{REAL}_{\Pi,A} \approx \text{IDEAL}_{F,\text{Sim}}. \quad (18)$$

□

6. Performance Analysis

We compare our work with other RSS schemes and show the results in Table 2. Shamir [1] first proposed the concept of secret sharing. Cevallos et al. [17] improved the share size and optimized the share size to $O(n+k)$. However, a participant needs to verify all other participants and also needs to be verified by all other participants in the reconstruction phase, which causes a large communication cost. In 2015, Allison Bishop et al. [18] designed an essentially optimal RSS scheme, whose share size is close to the optimal $\widetilde{O}(\kappa)$. They innovatively eliminated the linear relationship between the share size and the number of participants, but their scheme cannot fight a rushing adversary. Fehr et al. [19] proposed a RSS scheme that can fight a rushing adversary. Unfortunately, their algorithm needs to start with an honest participant.

Table 2: Feature comparison of schemes.

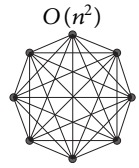
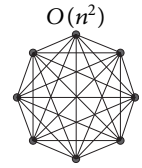
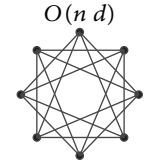
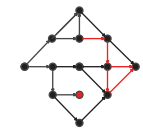
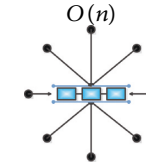
Paper	[1]	[17]	[18]	[19]	This work
Share size	$O(n\kappa)$	$O(n+\kappa)$	$\widetilde{O}(\kappa)$	$\widetilde{O}(\kappa)$	$\widetilde{O}(\kappa)$
Rushing adversary	No	Yes	No	Yes	Yes
Fairness	No	No	No	No	Yes
Communication complexity	$O(n^2)$	$O(n^2)$	$O(n\ d)$	$O(n\ d)$	$O(n)$
Communication topology					

Table 3: Table of gas cost.

Participant	$m = 5$	$m = 6$	$m = 7$	$m = 8$	$m = 9$	$m = 10$
submit_1	350 841	421 086	491 203	561 448	631 693	701 938
submit_2	245 845	295 014	344 183	393 352	442 649	491 946
Tag verification	257 005	308 406	359 807	411 208	462 609	514 010
MAC verification	623 926	759 801	886 234	1013 186	1139 419	1265 852
total	1477 617	1784 307	2081 427	2379 194	2676 370	2973 746

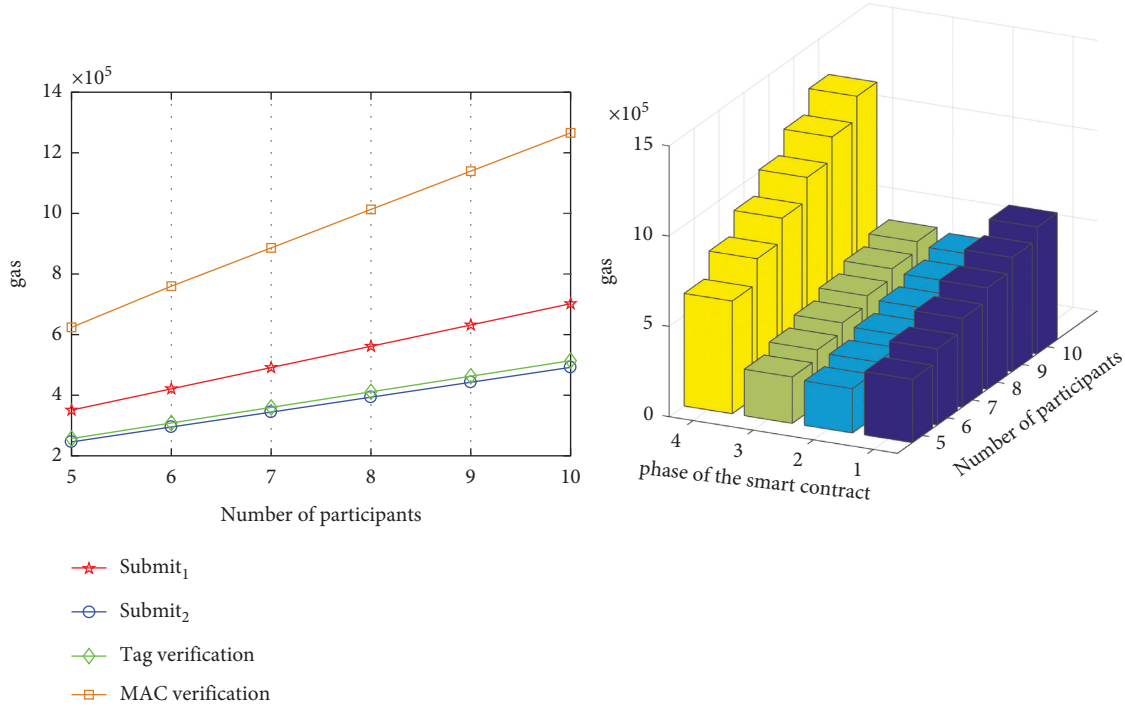


Figure 4: Gas cost.

We ran our scheme on a local laptop that has the 64-bit Windows system, an Intel (R) Core (TM) i5-9300H CPU (2.40 GHz), and 16 GB of RAM. The scheme uses Solidity to program the smart contract. Solidity is a language that was created to implement smart contract. We compiled the smart contract via Ethereum's official website and deployed it on the private chain that we created locally.

Every step of implementing the smart contract can be considered to be a transaction, which is open and transparent. Before the test, we set the security parameter κ to 128 bits. The parts we tested contain two rounds of the share submission phase, label verification phase, and MAC verification phase. The gas cost of each part is shown in Table 3. We use Figure 4 to vividly show the relationship between the gas cost and the number of participants. The figure shows that the gas cost is linearly related to the number of participants; the more participants there are, the more expensive it is to execute the smart contract.

The time cost of each part is shown in Table 4. It takes approximately 170 ms for label verification and

Table 4: Table of time cost (ms).

Participant	$m = 5$	$m = 6$	$m = 7$	$m = 8$	$m = 9$	$m = 10$
submit_1	1121	1328	1379	1495	1752	1916
submit_2	668	881	973	1142	1325	1517
Tag verification	930	1083	1215	1374	1536	1728
MAC verification	1633	1927	2285	2543	2787	3163
reconstruction	0.31	0.46	0.52	0.68	0.78	1.09

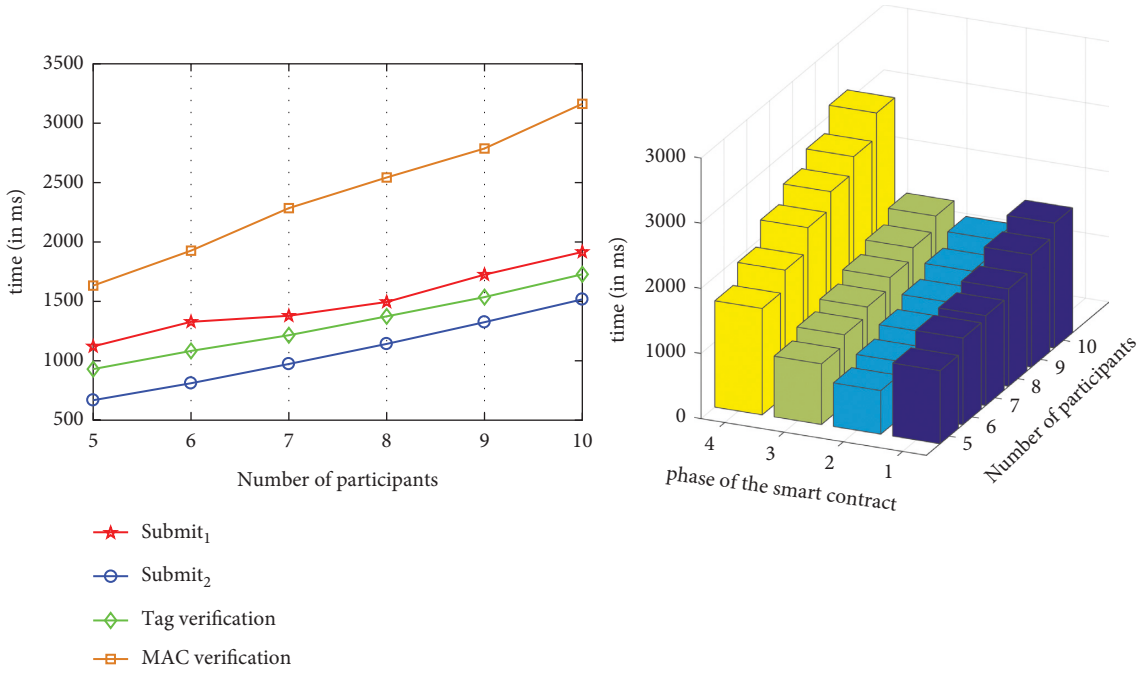


Figure 5: Time cost.

approximately 300 ms for MAC verification. Similarly, we also use Figure 5 to illustrate the relationship between the time cost and the number of participants. The figure shows that the time cost is linearly related to the number of participants; the more participants there are, the more time it takes to execute the smart contract.

7. Conclusion

We proposed a new RSS scheme based on smart contracts that ensure fairness and decentralization. Compared with previous schemes, the communication complexity of our scheme is reduced to $O(n)$. Out scheme is secure against a rushing adversary. The previous discussion shows that our scheme can finally reconstruct the correct secret by means of the Solomon decoding algorithm even if any corrupt parties (less than t) submit the wrong shares.

Unfortunately, the cost of executing smart contracts is expensive when the number of shares is large. The scheme is applicable to the situation where the number of shares is less than 2^{15}. Our work can be improved as follows. We can extend RSS to hierarchical robust secret sharing scheme, which allows secret shares to be distributed to different levels of parties [35]. What is more, the RSS scheme can be applied in the situation where the number of shares is large [36]. Also, we can apply RSS in other fields such as robust secret image sharing and threshold private set intersection [37, 38].

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (20200115 and 62002103), the Science and Technology Research Project of Henan Province (212102210388 and 172102210045), Henan Province Soft Science Research Plan Projects (212400410109), and Soft Science Program of the State Intellectual Property Office of China: Normative research on the application of intellectual property protection of blockchain technology from the perspective of "Digital Nation"(SS21-B-11).

References

[1] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[2] G. R. Blakley, "Safeguarding cryptographic keys," in *Proceedings of the AFIPS 1979 National Computer Conference*, pp. 313–317, NY, USA, June 1979.

[3] K. Kurosawa, S. Obana, and W. Ogata, "T-Cheater identifiable (k, n) threshold secret sharing schemes," in *Proceedings of the CRYPTO'95:15th Annual International Cryptology Conference*, pp. 410–423, Santa Barbara, CA, USA, August 1995.

[4] D. Aggarwal, I. Damgård, J. B. Nielsen et al., "Stronger leakage-resilient and non-malleable secret sharing schemes for general access structures," in *Proceedings of the CRYPTO 2019 39th Annual International Cryptology Conference*, pp. 510–539, Santa Barbara, CA, USA, August 2019,.

[5] I. Dinur, N. Keller, and O. Klein, "An optimal distributed discrete log protocol with applications to homomorphic secret sharing," in *Proceedings of the CRYPTO 2018:Part III 38th Annual International Cryptology Conference*, pp. 213–242, Santa Barbara, CA, USA, August 2018.

[6] V. Goyal and A. Kumar, "Non-malleable secret sharing for general access structures," in *Proceedings of the CRYPTO 2018, Part I 38th Annual International Cryptology Conference*, pp. 501–530, Santa Barbara, CA, USA, August 2018.

[7] O. Farràs, T. Hansen, T. Kaced, and C. Padró, "Optimal non-perfect uniform secret sharing schemes," in *Proceedings of the CRYPTO 2014 Part II 34rd International Cryptology Conference*, pp. 217–234, Santa Barbara, CA, USA, August 2014.

[8] J. B. Nielsen and M. Simkin, "Lower bounds for leakage-resilient secret sharing," in *Proceedings of the Advances in Cryptology - EUROCRYPT 2020 Part I 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 556–577, Zagreb, Croatia, May 2020.

[9] E. Boyle, L. Kohl, and P. Scholl, "Homomorphic secret sharing from lattices without FHE," in *Proceedings of the Advances in Cryptology - EUROCRYPT 2019 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 3–33, Darmstadt, Germany, May 2019.

[10] S. Qin, Z. Tan, F. Zhou, J. Xu, and Z. Zhang, "A verifiable steganography-based secret image sharing scheme in 5g networks," *Security and Communication Networks,*vol. 2021, Article ID 6629726, 14 pages, 2021.

[11] Y. Wang, J. Chen, Q. Gong, X. Yan, and Y. Sun, "Weighted polynomial-based secret image sharing scheme with lossless recovery," *Security and Communication Networks,*vol. 2021, Article ID 5597592, 11 pages, 2021.

[12] K. Meng, F. Miao, Y. Ning, W. Huang, Y. Xiong, and C. Chang, "A proactive secret sharing scheme based on Chinese remainder theorem," *Frontiers of Computer Science,*vol. 15, no. 2, Article ID 152801, 2021.

[13] S. Zhang, J. Wang, Y. Zhang, B. Pei, and C. Lyu, "An efficient proactive secret sharing scheme for cloud storage," in *Proceedings of the Applied Cryptography and Network Security Workshops*, pp. 346–357, Rome, Italy, October 2021.

[14] T. Rabin and M. Ben-Or, "Verifiable secret sharing and multiparty protocols with honest majority (extended abstract)," in *Proceedings of the 21st ACM STOC*, pp. 73–85, ACM Press, Seattle, WA, USA, May 1989.

[15] R. C. Serge, "Reducing the share size in robust secret sharing," 2011, http://www.algant.eu/documents/theses/cevallos.pdf.

[16] C. Blundo and A. D. Santis, "Lower bounds for robust secret sharing schemes," *Information Processing Letters*, vol. 63, no. 6, pp. 317–321, 1997.

[17] A. Cevallos, S. Fehr, R. Ostrovsky, and Y. Rabani, "Unconditionally-secure robust secret sharing with compact shares," in *Proceedings of the EUROCRYPT 2012 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 195–208, Cambridge, UK, April 2012.

[18] A. Bishop, V. Pastro, R. Rajaraman, and D. Wichs, "Essentially optimal robust secret sharing with maximal corruptions," in *Proceedings of the Advances in Cryptology - EUROCRYPT 2016 EUROCRYPT 2016, Part I 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 58–86, Vienna, Austria, May 2016.

[19] S. Fehr and C. Yuan, "Towards optimal robust secret sharing with security against a rushing adversary," in *Proceedings of the Advances in Cryptology - EUROCRYPT 2019, In: EUROCRYPT 2019, Part III;38th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 472–499, Darmstadt, Germany, May 2019.

[20] S. Fehr and C. Yuan, "Robust secret sharing with almost optimal share size and security against rushing adversaries," in *Proceedings of the Theory of Cryptography-18th International ConferenceTCC 2020, Part III*, pp. 470–498, Durham, NC, USA, November 2020.

[21] M. Cheraghchi, "Nearly optimal robust secret sharing," *Designs, Codes and Cryptography*, vol. 87, no. 8, pp. 1777–1796, 2019.

[22] R. Cramer, Y. Dodis, S. Fehr, C. Padró, and D. Wichs, "Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors," in *Proceedings of the EUROCRYPT 2008 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 471–488, Istanbul, Turkey, April 2008.

[23] V. Guruswami and A. Rudra, "Explicit codes achieving list decoding capacity: error-correction with optimal redundancy," *IEEE Transactions on Information Theory*, vol. 54, no. 1, pp. 135–150, 2008.

[24] P. Manurangsi, A. Srinivasan, and P. N. Vasudevan, "Nearly optimal robust secret sharing against rushing adversaries," in *Proceedings of the Advances in Cryptology - CRYPTO 2020-40th Annual International Cryptology Conference,Part III*, pp. 156–185, Santa Barbara, CA, USA, August 2020.

[25] M. N. Wegman and J. L. Carter, "New hash functions and their use in authentication and set equality," *Journal of Computer and System Sciences*, vol. 22, no. 3, pp. 265–279, 1981.

[26] M. Bellare, "New proofs for NMAC and HMAC: security without collision-resistance," in *Proceedings of the CRYPTO'06: Proceedings of the 26th annual international conference on Advances in Cryptology*, pp. 602–619, Santa Barbara, CA, USA, August 2006.

[27] L. Welch and R. Scholtz, "Continued fractions and Berlekamp's algorithm," *IEEE Transactions on Information Theory*, vol. 25, no. 1, pp. 19–27, 1979.

[28] S. Gao, "A new algorithm for decoding reed-solomon codes," *Communications, Information and Network Security*, Springer, Boston, MA, 712.

[29] Y. Xu, P. Ahokangas, S. Yrjölä, and T. Koivumäki, "The fifth archetype of electricity market: the blockchain marketplace," *Wireless Networks*, vol. 27, no. 6, pp. 4247–4263, 2021.

[30] D. C. Sánchez, "Raziel: private and verifiable smart contracts on blockchains," 2017, https://eprint.iacr.org/2017/878 Report 2017/878.
[31] J. Benet, "IPFS - content addressed, versioned, P2P file system," 2014, https://arxiv.org/abs/1407.3561.
[32] O. Goldreich, *Foundations of Cryptography: Basic Applications*, Cambridge University Press, Cambridge, UK, 2004.
[33] K. Chung, H. Lam, Z. Liu, and M. Mitzenmacher, "Chernoff-hoeffding bounds for Markov chains: generalized and simplified," in *Proceedings of the 29th International Symposium on Theoretical Aspects of Computer Science, STACS*, Paris, France, March 2012.
[34] V. Moulos, "A hoeffding inequality for finite state Markov chains and its applications to Markovian bandits," in *Proceedings of the IEEE International Symposium on Information Theory, ISIT*, pp. 2777–2782, Los Angeles,CA, USA, June 2020.
[35] E. Zhang, M. Li, S.-M. Yiu, J. Du, J.-Z. Zhu, and G.-G. Jin, "Fair hierarchical secret sharing scheme based on smart contract," *Information Sciences*, vol. 546, pp. 166–176, 2021.
[36] T. P. Thao, M. S. Rahman, M. Z. A. Bhuiyan, A. Kubota, S. Kiyomoto, and K. Omote, "Optimizing share size in efficient and robust secret sharing scheme for big data," *IEEE Trans. Big Data*, vol. 7, no. 4, pp. 703–716, 2021.
[37] X. Yan, L. Liu, L. Li, and Y. Lu, "Robust secret image sharing resistant to noise in shares," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 17, no. 1, p. 24, 2021.
[38] Y. Sun, Y. Lu, X. Yan, L. Liu, and L. Li, "Robust secret image sharing scheme against noise in shadow images," *IEEE Access*, vol. 9, Article ID 23300, 2021.