Research on Monitoring Technology of Industrial Cannabis Based on Blockchain and SM Series Cryptographic Algorithm

Zijian Ma[1], Zhiqiang Wang[1,2], Hang Wu[3], Xingyu Guo[2], and Xizhen Wang[3]
(Corresponding author: Zhiqiang Wang)
School of Computer and Information Technology, Beijing Jiaotong University[1]
Beijing 100044
Email: wangzq@besti.edu.cn
Beijing Electronic Science and Technology Institute[2]
School of Telecommunications Engineering, Xidian University[3]
(Received Dec. 16, 2020; Revised and Accepted June 6, 2021; First Online Nov. 9, 2021)

Abstract

In China, the production and marketing relationship of industrial cannabis is becoming more and more complex. The existing supervise means are mainly manual, with low efficiency and security. Therefore, this paper proposes a traceable supply chain monitoring solution based on the private chain and proof of work (PoW). The records of our scheme including the source, the destination, the amount, and the loss of industrial cannabis for each transaction. The directed graph is used to abstract the transaction process. And three types of illegal behavior can be identified by alarm strategies. SM3 Cryptographic Hash Algorithm is used for integrity protection, and SM9 Identity-Based Cryptographic Algorithm is used for authenticity and confidentiality protection. In the experiment, with the verification system designed, the PoW calculation is stable, and the optimized encryption algorithms work efficiently.

Keywords: Blockchain; Industrial Cannabis; SM3; SM9

1 Introduction

Cannabis represents one of the most important global growth opportunities in the next decade. In China, after Yunnan and Heilongjiang, Jilin will become the third province to liberalize the cultivation of industrial hemp. During the cannabidiol (CBD) extraction and production, it is possible to purify THC as the raw material of drugs. Therefore, the entire production process must be strictly controlled by the public security department, with all aspects of the network monitoring. With the increasingly complex relationship between the production and marketing of industrial hemp, problems of low efficiency and low safety are exposed. It is urgent for public security departments to trace industrial hemp and its related products of information, and effectively supervise the production, processing, circulation, and sales of industrial hemp.

This paper studies the monitoring technology of industrial hemp based on the blockchain, and adopts the Chinese cryptographic algorithm to realize the security and reliability of the monitoring technology, to improve the regulatory efficiency of industrial hemp and the security of the monitoring information.

2 Related Work

Due to its high economic value, China's industrial hemp cultivation scale is constantly expanding, and its products are also applied in various fields [4]. Fiber and seed are the main products of industrial hemp. The former is widely used as textile raw materials due to its breathable and bacteriostatic properties. The latter has edible and medicinal value. Hemp materials, such as flowers and leaves, are also of value after processing, from which CBD can be extracted and used in the production of nutrition products, skin-care products, and energy drinks.

2.1 THC and Related Purification Techniques

Generally, the mass fraction of THC in industrial hemp is less than 0.3 % and has no drug utilization value. Human ingests hemp containing more than $3.0 gkg^{-1}$ of THC (expressed as mass concentration), then it shows that a certain tendency of medicinal or abuse [16]. At the present stage, mature technologies such as chromatographic production [3] and supercritical fluid [14] can be used to efficiently extract and obtain high-purity THC from industrial hemp flowers, stems, and leaves. This means that it is possible to purify drugs in the whole process of in-

dustrial hemp production and processing, which indicates that risks exist in each part of the industrial hemp supply chain.

2.2 Industrial Hemp Supply Chain Analysis Method

Existing analysis methods for the cannabis supply chain include Predator-Prey based model [5], Classification Trees [19], Information Fusion Predictive model based on medical approaches [1], and Embedding role Classification based on Compositional Multiview [2]. Among them, the Predator-Prey based model analyzes illicit drug consumption by its users. Classification Trees quantifies and predicts the use of industrial hemp products. Information Fusion Predictive model presents a supply and demand analysis method based on online social network data. Classification of the role based on Compositional Multiview Embedding dynamically feedback relevant information of individual users, informed organizations, and retail accounts by integrating time and geographical location. However, with the increasingly complex relationship between its production and marketing, the above methods cannot meet the need for comprehensive and traceable monitoring.

2.3 Blockchain and Proof-of-Work

Although the specific implementation of blockchain technology is different, it still has many generalities in the whole system architecture. Taking Bitcoin and Ethereum as examples to analyze the general structure of blockchain, can be divided into five layers shown in Figure 1, which are the application layer, thecontrol layer, the consensus layer, the network layer, and the data layer. Among them, the control layer includes automatic scripts and sandbox environment written by the program, and the data layer includes block data structure and specific storage content.

Due to its characteristics of openness, cooperation and mutual trust, decentralization, and confidentiality, and reliability, blockchain technology has been applied to information sharing and product traceability, with most application scenarios in finance and medical treatment. Typically, blockchain can realize distributed Shared ledger and generate intelligent financial reports [7], realize fair payment contracts in cloud storage public audit [13], solve electronic medical record management problems in Internet of Things devices [12], and build a reverse supply chain of expired drugs [18]. Trust relationship and privacy protection are the key points in the practical application of blockchain, which need to be supported by reliable algorithms. Existing blockchain consensus algorithms include Proof of Work (PoW), Proof of Stake(PoS), Practical Byzantine Fault Tolerance (PBFT) and Raft algorithm, *etc.* [11]. Under the further study, Chinese researchers proposed a PBFT [20] based on a tree-topology network and a Raft [8] combined with a BLS signature. However, the blockchain hash algorithm is dominated by SHA-256 [15], and the asymmetric encryption algorithm is dominated by RSA, which is relatively limited and has some security risks, so it is not conducive to domestic generalize. Beijing Electronic Science and Technology Institute has proposed a provable security block chain privacy protection scheme based on the SM9 algorithm [17], but it only focuses on group signature and is limited to theoretical analysis, without combining specific application scenarios.

3 The Design of Blockchain

Combined with the Chinese cryptographic algorithm and the actual application requirements of industrial hemp monitoring, this paper focuses on the blockchain data layer. The storage model based on account stores data in the form of key-value pairs and constantly updates the account data by executing transactions, which is not convenient for the comprehensive and specific recording of industrial hemp flow information. In order to realize the reverse traceability of the supply chain, a private chain was constructed based on the transaction and the circulation of each batch of industrial hemp was recorded.

3.1 Proof of Work

This paper designs a private chain based on the Proof of Work (PoW) algorithm and controls the block generation. The node matches the target by calculating, and then generates the block.

The design of PoW is not complex, and the computational overhead is relatively stable. The reasons are as follows. First, private chain nodes are dominated by personal computers and do not have great computing power. Second, PoW sets the number of unrecorded transactions as a trigger condition. And the Stable computational overhead means a stable time from PoW to a new block generation. The number of transactions generated is limited so that the number of transactions recorded in each block is similar. Third, the purpose of the new block is generated for recording transactions. Nodes are not rewarded for finding blocks, and it is meaningless to skip the PoW attack strategy.

Starting from the work's initial value, the calulation process of each iteration is shown in Table 1. Current proof p' is increment by the previous proof p, and k is the increment. Before each increment, p and p' are concatenated as a string, and the result is substituted into the SM3 hash algorithm. If the number of 0 at the end of the hash is l, then hit the target, otherwise continue to calculate. In the verification system, the initial value of work 1, and k=3,l=1, the reasons are shown in Section 6.1.

3.2 Block Structure

The creation block and the general block are designed as the basic components of the block chain. This section de-

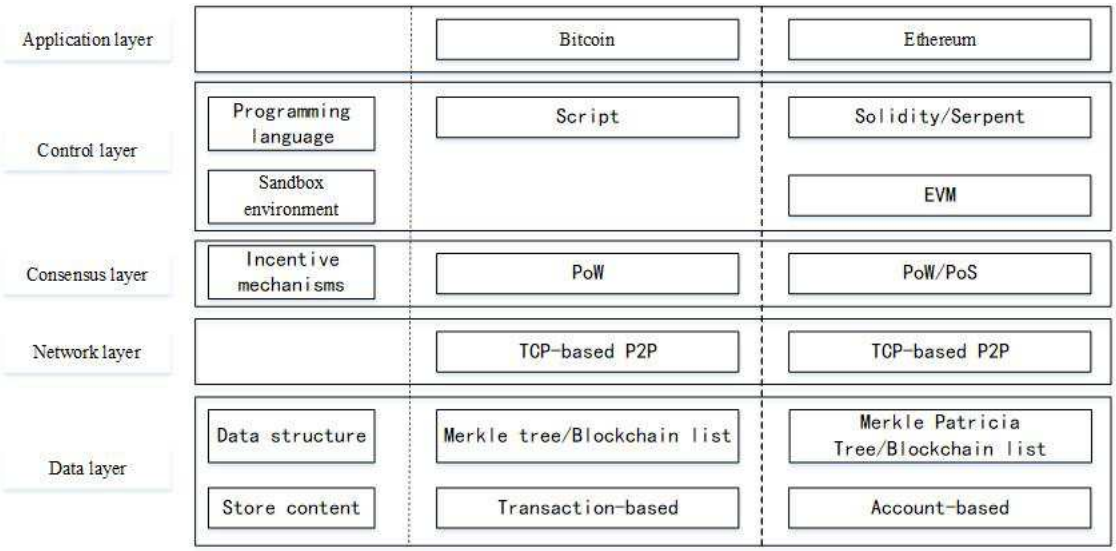

Figure 1: The proposed scheme

Table 1: Calculation process of PoW

Input: Pervious proof p Output: Current proof p'
Begin: $p' = p$ While $l! = 0$: $p' = p' + k$ End while End

scribes the two block structures respectively, as shown in Figure 2. In the header of the block, the previous block hash (PreBlockHash) is calculated by the SM3 hash algorithm, the Proof is calculated by the POW algorithm, and the block serial number (Index) and Time stamp (Time) are automatically generated.

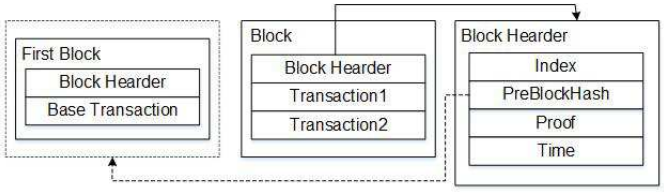

Figure 2: Industrial hemp block structure

The creation block is the first block in the private chain and can only be initiated by industrial hemp producers. And it only stores the corresponding industrial hemp production batches of Base Transaction. There is no preblock, so the hash of the previous block is the constant value 100, and the work is set to the initial value of 1.

Generally, a block has and only has one precursor, and each node is generated by work calculation. The record involves multiple transactions of this batch of industrial hemp. There are three reasons to eliminate incentives. First, the blockchain used is private and does not require or attract other computing power to participate in the PoW. Second, non-profit, but for recording transactions. Third, from a security perspective, it makes no sense to attack incentives such as Selfish Mining.

3.3 Transaction Structure

Transactions initiated by each node are recorded in the block in chronological order. In the context of industrial hemp application, the transaction can correspond to a certain process, as well as market behaviors such as transportation and purchase. In particular, the initial transaction refers specifically to the first transaction recorded in the creation block, the property of which is described in Section 2.4.

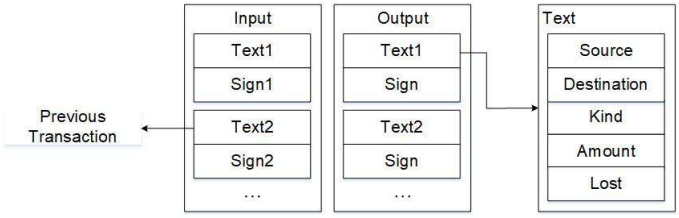

Figure 3: Transaction structure of industrial hemp

The designed transaction structure is shown in Figure 3.,where the target of the Sign is the Source of the sender. The Kind of industrial hemp product is transmitted in ciphertext, and only the Destination can be seen after decryption. Normal loss may be caused during processing and transportation, so in order to comprehensively and concretely reflect and quantify the flow of industrial hemp products, we record Amount and Lost by quality.

3.4 Monitoring and Alarm Policies

The transaction behavior in the block chain is abstracted by the directed generation graph D=(V,E,C). Among them,$\mathbf{V} = \{v_0, v_1, v_2...\}$ is the node set, v_0 is the super source point, v_1 is the originating node of the creation block. E is a directed edge set, and $e_{ab} \in E$ represents node v_a to send a transaction to the node v_b. C is the directed edge capacity set, c_{ab}=(i,j),$c_{ab} \in C$ represents the transaction in e_{ab}, the industrial hemp transaction is I and the loss is J.

The input degree of super source node v_0 is 0, and the output degree is 1, the corresponding entity is the same as v_1 . Industrial hemp producers initiate creation blocks that may send the initial trade directly to specific nodes $v_0 \to v_1 \to v_a$. It was also possible to send an initial transaction to myself and then initiate a transaction to multiple nodes$v_0 \to v_1 \to W$, $W \subseteq V$. Whatever in any case, $c_{01} = (i, j)$,j=0, i is always the sum of the quality of the industrial hemp batch. In the former case, v_a obtained all the industrial hemp in that batch. In the latter case, v_1 distributes the industrial hemp batch to multiple nodes. Blockchain records all transactions involving the batch of industrial hemp means $c_{01}[0] = \sum c_{ab}[1] + \sum c_{ba}[0] - \sum c_{ab}[0], a \neq 0, b \neq 0$, and the sum of the total loss and the amount of each node is equal to the initial trades.

When $a \neq 0$, a new transaction e_{ab}, is initiated, an exception is determined, and the transaction is terminated immediately. First, the receiver doesn't exist, that is $v_b \notin V$. Second, the node $v_a$lack holdings, that is $\sum c_{ka}[0] - \sum c_{ak}[0] - \sum c_{ak}[1] < i + j$,$c_{ab}$=(i,j). Third, the quality loss is over expected, that is $j \geq n * i$,c_{ab}=(i,j), n is the proportionality constant, which can be adjusted according to the actual situation. And n is equal to 1 in the verification system.

4 SM3 Hash Cryptographic Algorithm

SM3 hash algorithm was released as the national standard by State Cryptography Administration in 2016. Merkle-Damgard structure is adopted in the algorithm. And for input messages of variable length, 256 bit hash value is generated and output after filling and iterative compression [10]. This section describes the software optimization implementation of SM3 used in the blockchain.

4.1 Constants and Functions

Initial value IV consists of eight 32-bit series:

$$\begin{aligned} \text{IV} = \quad & 7380166\text{f} \;\; 4914\text{b}2\text{b}9 \;\; 172442\text{d}7 \;\; \text{da}8\text{a}0600 \\ & \text{a}96\text{f}30\text{bc} \;\; 163138\text{aa} \;\; \text{e}38\text{dee}4\text{d} \;\; \text{b}0\text{fb}0\text{e}4\text{e} \end{aligned}$$

Constant T_j is used in modular 232 addition operation to reduce the linear and differential genetic probability between input and output [6]:

$$T_j = \begin{cases} 79cc4519, & 0 \leq j \leq 15 \\ 7a879d8a, & 16 \leq j \leq 63 \end{cases}$$

Boolean function, $\wedge$ represents 32-bit **AND** operation,$\vee$ is 32-bit **OR** operation, $\oplus$is 32-bit **XOR** operation,$\neg$is 32-bit **NOT** operation:

$$FF_j(X,Y,Z) = \begin{cases} X \oplus Y \oplus Z, & 0 \leq j \leq 15 \\ (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z), & 16 \leq j \leq 63 \end{cases}$$

$$GG_j(X,Y,Z) = \begin{cases} X \oplus Y \oplus Z, & 0 \leq j \leq 15 \\ (X \wedge Y) \vee (\neg X \wedge Z), & 16 \leq j \leq 63 \end{cases}$$

Replacement function, $\oplus k$ represents k-bit ring left shift of 32-bit.

$$\begin{aligned} P_0(X) &= X \oplus (X <<< 9) \oplus (X <<< 17) \\ P_1(X) &= X \oplus (X <<< 15) \oplus (X <<< 23). \end{aligned}$$

Calculate 32 bit ring left shift K bit, $X \gg (32-k) || X \ll k$. Under the condition of multi-digit abandonment, then take the **XOR** operation between the result of moving X to the left k bit and X to the right 32-k bit.

4.2 Computational Flow and Software Optimization

We will fill the message m whose length L is not a multiple of 512. We add 1 bit "1", k bit "0", and a 64-bit string at the back of message m. Where K is the minimum non-negative integer satisfying l+k+1=448 **mod** 512, and the 64-bit string is the binary representation of length L. Take message 00000000 as an example, its length L =8, and the filling process is shown in Figure 4.

Figure 4: Example of SM3 message padding

Optimize the process with the help of bytes structure and mathematical reasoning.Encode the message m and get the bytes array, and convert it to a hexadecimal string. Take four bits as a group, and the filling rules are as follows. The first 1-bit is "1" which is combined with the subsequent 3-bit "0", then add them to hexadecimal string as an "8". The remaining k-3 bit "0" combinations are added as hexadecimal "0" and the number is (k-3) // 4. For the 64-bit strings, calculate the hexadecimal length l for h (leading 0 is reserved) firstly. Add hexadecimal "0" with the number of 16-L_h, where L_h is the hexadecimal length of H. Finally, add h. After coding, l % 4=0, and because of k=447-l **mod** 512, so $k > 3$ and $(k-3)\%4 \equiv 0$ when $0 < l < 512$. So we can optimize by bytes array, and the time complexity is reduced from O (k) to O (k/4).

Table 2: Iterative compression process

Input: message grouping $B^{(i)}$, initial value IV Output: compression results $V^{(n)}$

for $i = 0$ *to* $n-1$ do
$V^{(i+1)} = CF(V^{(i)}, B^{(i)})$
end for

Table 3: Message extension process

Input: Divide $B^{(i)}$ to get 16 words $W_0, W_1 ... W_{15}$ Output: $W_{16}, W_{17} ... W_{67}, W_0', W_1' ... W_{63}'$

for $j = 16$ *to* 67
$W_j = P_1(W_{j-16} \oplus W_{j-9} \oplus (W_{j-3} <<< 15)) \oplus (W_{j-1} <<< 7) \oplus W_{j-6}$
end for
for $j = 16$ *to* 63 do
$W_j' = W_j \oplus W_{j+4}$
end for

Iteratively compress the filled message m′. Then the message m′ is grouped with a length of 512 bits, denoted as B^i, where $0 \le i \le (l+k+65)/512 - 1$. The iterative process is shown in Table 2, where CF is the compression function, V^0 is the initial value IV, V^n is the compression result, and n=(l+k+65)/512. Message expansion in the unit of message grouping B^i, 132 message words are generated for the compression function, denoted as W_0, W_1 $\cdots$, W_{67}, W_0', W_1', $\cdots$, W_{63}', it is shown in Table 3.

In big-end storage mode, CF calculation process is shown in Table 4. Where *A,B,C,D,E,F,G,H* are directly calculated as integer data, and SS1, SS2, TT1, TT2 are intermediate variables.

5 SM9 Identification Cryptographic Algorithm

SM9 identity cipher algorithm is a bilinear pair-based identity cipher algorithm, published as the national standard by the State Cryptography Administration in 2015. It generates public and private key pairs based on the user's identity, and the identity information is associated with the password algorithm, eliminating the process of digital certificate, certificate library, and keystore management. This section introduces the specific functions of the algorithm and the implementation of software optimization.

5.1 Propaedeutics

The mathematical tools of SM9 are point group operation and bilinear pair operation of elliptic curve over finite field group. Let G_1 and G_2 be two additive cyclic groups of order N, G_T be a multiplicative cyclic group of order N, N is a prime number, and $N > 2^{191}$. P_1 and P_2 are generators of G_1 and G_2 respectively.

Point group operation is the basis, it includes addition and multiple points. The addition operation is the mod N addition of G_1 and G_2 domain and the symbol is expressed as m+n=(m+n) *mod* N, where $m, n \in G$. The multiple point operation is the repeated addition of the same point on an elliptic curve.

Bilinear pair operation e(G_1, G_2) is mapping $G_1 \times G_2 \to G_T$. Four kinds of intractable problems are used to ensure the safety of the algorithm.

Problem 1, bilinear inverse (BIDH), given ($a[P_1]$, $b[P_2]$), $a, b \in [1, N-1]$, to calculate $e(P_1, P_2)^{b/a}$ difficult.

Problem 2, judgmental bilinear inverse (DBIDH), given $(a, b, r) \in [1, N-1]$, it is difficult to distinguish between (P_1, P_2, $[a]P_1$, $[b]P_2$, $e(P_1, P_2)^{b/a}$).

Problem 3, τ-bilinear inverse (τ-Gap-BDHI), given (P_1, $[x]P_1, P_2, [x]P_2, [x^2]P_2, \cdots, [x^\tau]P_2), \tau, x \in [1, N-1]$ computing $e(P_1, P_2)^{1/x}$ difficult.

Problem 4, τ-Gap-bilinear inverse (τ-Gap-BDHI), given (P_1, $[x]P_1, P_2, [x]P_2, [x^2]P_2, \cdots, [x^\tau]P_2), \tau, x \in [1, N-1]$ computing $e(P_1, P_2)^{1/x}$.

5.2 The General Process

According to functions, four modules can be divided: parameter initialization, calculation initialization, key derivation, and password calculation. The calculation process is shown in Figure 5.

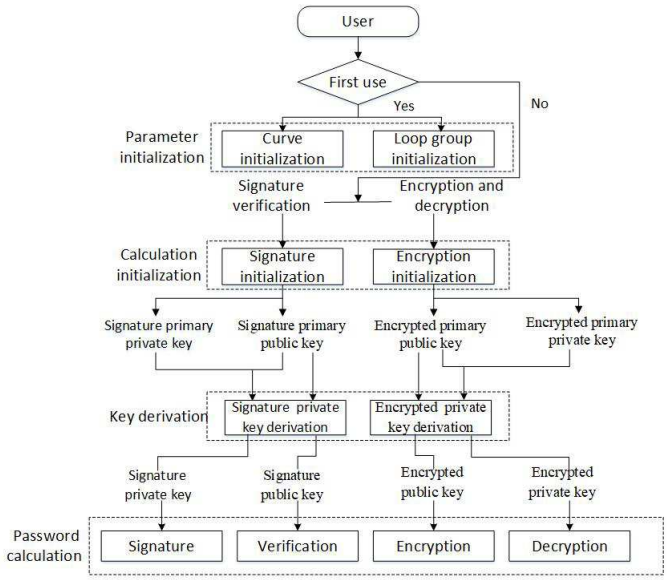

Figure 5: SM9 function module and calculation process

The parameter initialization module provides calculation parameters. The curve initialization completes the definition of finite field and elliptic curve. The loop group

Table 4: The compression function of the calculation process

Input: $W_0, W_1 ... W_{67}, W_0', W_1' ... W_{63}', V^{(i)}$ Output: $V^{(i+1)}$

$ABCDEFGH \leftarrow V^{(i)}$
for $j = 0$ *to* 63 do
$SS1 \leftarrow ((A <<< 12) + E + (T_j <<< (j \mod 32))) <<< 7$
$SS2 \leftarrow SS1 \oplus (A <<< 12)$
$TT1 \leftarrow FF_j(A,B,C) + D + SS2 + W_j'$
$TT2 \leftarrow GG_j(E,F,G) + H + SS1 + W_j$
$D \leftarrow C$
$C \leftarrow B <<< 9$
$B \leftarrow A$
$A \leftarrow TT1$
$H \leftarrow G$
$G \leftarrow F <<< 19$
$F \leftarrow E$
$E \leftarrow p_0(TT2)$
end for
$V^{(i+1)} = ABCDEFGH \oplus V^{(i)}$

initializes the calculated generators P_1 and P_2. For each user, generate tuples that are unique and generally not modified. So parameter initialization is only done for the first use by users, or when information security is compromised and the generation tuple must be regenerated.

The calculation initialization module realizes the generation of the primary key. In signature initialization, the signature's master private key s is generated randomly. The signature's primary public key S is s times P_2, published with a bilinear pair of P_2 and S. In the encryption initialization, the encryption main private key e is generated randomly. The encrypted primary public key E is e times P_1, published with a bilinear pair of E and P_2.

The key derivation module uses SM3 as a auxiliary hash function to derive the key based on the primary key and user identity. When the signature private key is derived, the user identity participates in the hash operation, s is added to the hash value, and after modulus inverse, the result is multiplied with P_1 to generate the signature private key. When the signature private key is derived, the user identity participates in the hash operation, and the sum of e and the hash value is performed. After modulus inverse, the result is multifold with P_2, and finally, the encrypted private key is generated.

The cipher calculation module provides four functions: signature, check, encryption and decryption. When signing, the generated signature private key is used. To verify the signature, SM3 is used to generate a hash value of the user id. The result is multipoint calculated with P_2, and then added with the signature's primary public key. The final result is used to verify the signature. To encrypt it, SM3 is used to generate a hash of the user id. The result is multiplied by P_1, then added to the encrypted primary public key and encrypted with the final result. When decrypted, the generated encrypted private key is used.

5.3 Software Optimization

Key derivation function *KDF (Z,L)* is called in key derivation module, cipher calculation module, and other relevant modules calculation. Its input Z is the data to be processed and L is the length of data to be obtained. In this paper, SM3 is used to assist KDF calculation. Consistent with the output format of SM3, KDF computations no longer use bit strings, but are grouped into hexadecimal strings. Z is entered directly as a hexadecimal string, and L indicates the desired hexadecimal string length. The 32-bit registers involved in the computation are represented as 8-bit hexadecimal in the implementation of the software. Compared with the computation bit by bit, the time complexity of the algorithm is reduced by 1/4.

Except for KDF, other data information processed by key derived module and cipher calculation module is expressed by hexadecimal. Hexadecimal strings have an advantage over strings in computation involving a decimal integers. As shown in Figure 6, if "4d " appears as a character, the conversion to a decimal integer must be encoded firstly, then each character is represented as a decimal integer and finally stored as a list. If "4d " is regarded as hexadecimal, it can be converted directly into hexadecimal, with less computational overhead. In particular, it is different from the hexadecimal representation of Large Numbers, we use bytes as a structure for grouping, which takes up less space. Take "test" as an example, as the information to be processed, its hexadecimal representation is e6b58be8af95, with "b" data storage, occupying 39 memory. It is stored with type 0x data, occupying 69 memory.

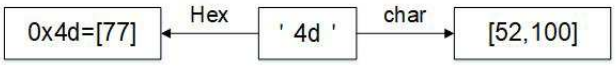

Figure 6: Results of type conversion under different strategies

6 Verification System Design

The verification system is based on B/S architecture and realizes application layer interaction through Web pages. Through this system, the block running information is checked, and the block running condition is detected.The system provides a verification platform for SM3 and SM9 related experiments. This section introduces its function module design, running process, and boundary treatment method.

6.1 Development Environment

Javascript, JS for short, is a widely used literal translation scripting language. In addition to UI (User Interface) design, the verification system uses this language for access control and boundary handling. With the help of this language, the validity of the input data on the web page can be verified, and the communication data encapsulation with server and communication result feedback can be realized in the verification system.

JSON (Javascript Object Notation) is a lightweight data exchange format. In Python, this format can be solved by flask library functions. It is simple and flexible. Blockchain uses this format to record block and transaction information. In the verification system, the communication data transmitted between the server and the browser is mainly in this format.

AJAX (Asynchronous JavaScript And XML) is for interactive web pages that update web content without reloading. In the verification system, the web page uses this technology to interact with the server, request data in the process of bill backtracking, holding query and block verification, and display the data on the page. Submit data when the new transaction is initiated and the block created, receive and display feedback information from the server.

6.2 Function Module and Running Process

The verification system is designed to meet both experimental and application requirements. It can be divided into three modules according to functions: data interaction module, transaction initiation module, and query detection module, as shown in Figure 7. The data interaction module authenticates the user. The interface includes a user identity submission interface and an identity file upload interface. The user id is submitted for the first time if the identity file has not been generated. The user id is used to generate an identity file, after the verification system receives it, SM9 is called for operation, and the calculated results such as a generator, primary public key, and primary private key are encapsulated and returned in the form of the file. For users with existing identity files, the verification system will verify the files, as detailed in Section 5.3. Allow the user to log in after confirming the identity file is correct.

Figure 7: Interface of verify system and function module

The transaction initiation module collects the specific information needed by the creation block and the new transaction initiation. On the interface, a text box prompts you to enter the "recipient" of industrial hemp trade, the "type" of industrial hemp traded, the "amount" of industrial hemp involved in the trade, and the normal "loss" of industrial hemp caused by the current node. The "quantity" and "loss" of industrial hemp were expressed in terms of quality (gram). Specifically, the industrial marijuana trade "sender" is fixed for the identity file read by the user id. The user clicks the "Start" button, and the input information is submitted as the initial transaction. The corresponding creation block is established, and the initial transaction is directly written into it. When the user clicks the "trade" button, the input information will be submitted as general trade, and the writing location will be determined according to the actual situation. Boundary handling and illegal input filtering are shown in Section 5.3.

The query and detection module provides three functions: "backtracking bill", "holding query" and "block check". Among them, "backtracking bill" returns all the current blockchain information, so that users can visually view the block generation and transaction records. Among the returned information, "signature" is the calculation result of SM9 signature algorithm, and the "sender" signature private key participates in the operation. "Type" is the result of SM9 encryption algorithm calculation, and "receiver" encrypted public key participates in the calculation. The "Possession query" returns the quality of the current user's industrial hemp possession, that is, the difference between the total input and output qualities. "Block check" returns the verified blockchain information, including workload proof, hash of the previous block, and user signature. Among them, if the signature is approved, the text "true" is returned. If the signature verification fails, the text "false" is returned. In particular, to confirm the correctness of encryption and decryption calculations, "category" is returned as SM9 decryption results, in contrast to the "back billing" return information.

In general, a new block is created using the verification system to record transactions, which can be done as shown in Figure 8. First, enter the user id in the data interaction module to get the identity file. The existing identity file is uploaded and read correctly in the data interaction module before entering the main interface. For certain batches of industrial hemp, the creation block cannot be initiated for general trade until it is produced. User A and user B log in respectively, and industrial marijuana consumption is 0 by default. User A, as an industrial hemp producer, initiated the initial transaction and generated the creation block. As the receiving party, user B's industrial marijuana consumption increases after receiving the transaction. At this point, User B can initiate a general transaction, but the transaction volume is limited and the sum of "quantity" and "loss" cannot exceed the total amount of industrial marijuana currently held. During the process, the trading volume limit that can be initiated can be confirmed through "holding query", and the current recorded trading information can be checked through "backtracking bill" to confirm the signature and encryption status. Finally, the results of PoW and cipher algorithm are verified by "block check".

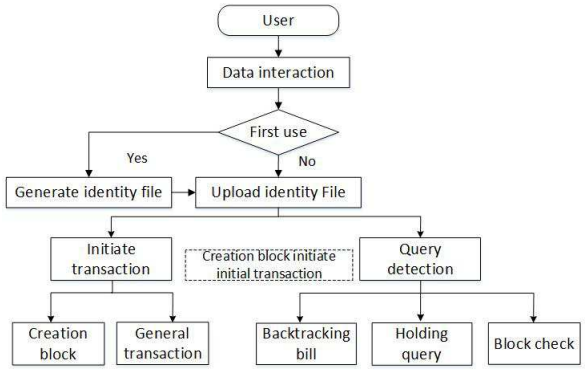

Figure 8: The flow of using

6.3 Access Control and Boundary Handling

This section describes in detail the access control and boundary handling policies adopted by function modules in the verification system. Data interaction module. When creating an identity file, the user's submitted identity cannot be empty. When a user submits an identity file, the content should correspond to what was encapsulated when the identity file was generated. If the identity file is missing or inconsistent, it will be judged illegal and users cannot access the transaction initiation module and query detection module.

Transaction initiation module. "Sender" is fixed and cannot be modified, especially the user ID read from the identity file. "Receiver" is the user id corresponding to the added node, and the server traverses the node information to detect it. If the "recipient" does not exist, the transaction is filtered and an error report is sent. Use regular expressions to determine the "amount" and "lost" input data types. Because the quality of industrial hemp corresponds, transactions are allowed to initiate if and only if "quantity" and "loss" inputs are real Numbers. In particular, the trigger condition of the "trade" and "Start" buttons is that all five input fields are not empty, that is, the transaction information filled in by the user must be complete.

Query detection module. In "block check", the detection contents related to blocks include workload proof and hash of the previous block. If the calculated results are inconsistent with the recorded content, the blockchain is not trusted. Transaction related detection content includes signature, encryption results. First, according to "sender" and "receiver", the signature and encryption principal are detected as registration nodes. If the subject does not correspond to the node information, the transaction will not be trusted. After that, the signature is verified.

7 Experimental Verification

7.1 PoW Test Analysis

Target hash: The number of 0's at the end is l.

Hit counts: PoW algorithm hits the target hash for the time of n.

Iterations: m represents the number of iterations of the PoW algorithm from the time n-1 to the time of n hit.

Intervals: k is the calculated increment used by the PoW algorithm.

This paper designs PoW algorithm based on SM3. Establish the blockchain private chain, cancel the incentive mechanism, require the new block mandatory generation. When a new transaction is initiated, if the record of the current number in transactions is p, it begins to calculate the work. In order to ensure that the number of transactions recorded in each block is similar and the block size is controlled at the same time, the PoW algorithm takes a stable hit as the main target and does not require excessive complexity.

This section measures and analyzes the hit of PoW algorithm under different conditions of k and l. In the experiment, $k \in \{1, 2, 3\}$, whose value is the prime factor of most natural numbers, can fully reflect the general influence of intervals on hit times. When the target is L, m_k is the number of iterations of the PoW algorithm. Record the measured value of m_k while l=1,$k \in \{1, 2, 3\}$, and part of the results are shown in Table 5.

As can be seen from Table 5, when l=1, the variation trend of m_k is similar for different k. That is, with the increase of hit times, the number of iterations increases and finally tends to be stable. Take l=2 and l=3 for multiple experiments, record data, and draw the curve of m

Table 5: The number of iterations under the hash of different targetse

N	m_1/times	m_2/times	m_3/times
1	11	11	88
2	33	33	106
3	56	87	106
4	59	107	106
5	73	127	106
6	75	129	106
7	78	165	106
8	79	187	106
9	82	211	106
10	123	243	106
11	125	273	106
12	137	297	106
13	144	325	106
14	157	333	106
15	166	347	106
16	190	381	106
17	196	441	106
18	204	491	106
19	227	493	106
20	240	597	106
21	259	711	106
22	272	743	106
23	311	743	106
24	313	743	106
25	333	743	106
26	347	743	106
27	381	743	106
28	406	743	106
29	432	743	106
30	435	743	106
31	435	743	106

Table 6: The number of hits required to achieve a constant time

N	l=1	l=2	l=3
k=1	30	71	6310
k=2	22	96	1759
k=3	2	112	379

changing with n, as shown in Figure 9. Where (a) is the changing trend of m with n when L =1; (b) is the changing trend of m with n when l=2; (c) is the changing trend of m with n when l=3.

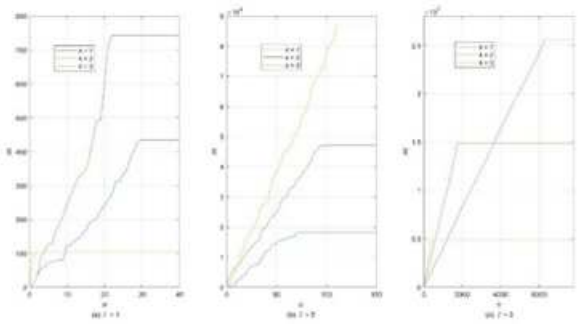

Figure 9: The change of iteration number under different target hashing

It can be seen from Figure 8 that the curve trend is increasing and finally reaches the level. In other words, for different k and l, the PoW algorithm hits a certain number of times, the target hash is uniformly distributed, and the number of iterations tends to be stable. This means that the blockchain using the PoW algorithm in this paper will have the following characteristics: as the chain length increases, the interval time required for the workload proof from the trigger of the calculation condition to the final generation of the new block gradually increases and finally reaches a constant. The different k and l only affect the hit number N needed to reach constant time, as shown in Table 6.

It can be seen from Table 6 that, when k is constant, N increases with the increase of l, and vice versa. This means that a blockchain using this PoW algorithm would have the following properties: the number of blocks that need to be generated before reaching a constant time can be controlled, and only by changing the target hash. The total number M_k of iterations needed to hit N times under different k values when l=3 is calculated by the following formula :

$$M_1 = \sum_{n=1}^{6310} m_1 = 81173004675$$

$$M_2 = \sum_{n=1}^{1759} m_2 = 12908482065$$

The minimum value is M_3, so the order of iteration required for N hits when l=4 is estimated:

$$M_3 = \sum_{n=1}^{379} m_3 = 909839005$$

It can be seen that when l=4, the number of iterations required to hit N times will be much larger than 9×10^8. Considering the time complexity of the SM3 algorithm, the iterative operation of the above order will consume huge computing resources, exceeding the computing capacity of personal computers. The block generation time

cannot be stable in a short time, which does not meet the practical application requirements.

To sum up, this paper designs a PoW algorithm based on SM3, and the blockchain using this algorithm will have the following two characteristics. First, when the blockchain reaches a certain length, the time taken to generate the block will tend to be stable. Second, the stable block generation time can and only can be achieved through target hash control. Based on the experiment in this section, the parameter values k=3,l=1 in the verification system were determined, so that the block generation time in the verification system could reach stability as soon as possible, and the calculation amount required for each hit was moderate, which could meet the requirements.

7.2 SM3 Test Analysis

Sample: Input data for any finite length string, password hash algorithm.

Sample length: l is the byte occupied by the sample.

Sample size: n is the total number of samples in the experimental group.

This paper designs the python software optimization implementation of the SM3 hash algorithm. In scheme A, the sample is processed as hexadecimal byte string directly, the cyclic left displacement is realized by the number theory method, and the message filling is optimized by mathematical reasoning. Scheme B is derived from gmssl-python and is used for comparison without computational optimization by mathematical reasoning. Scheme C is used for comparison, and the sample is transcoded to 0,1 string, and the operation is completed with data structures such as lists.

Due to the influence of hardware condition, external environment, and other factors, the code running time fluctuates in a small range. Therefore, the experimental group was designed, and the hash function was called for n times in the group to reduce the error. The contents of the 6 experimental groups are as follows:

Group one: n = 10, l = 100B, the samples are identical and consist of numbers, letters, and Chinese characters.

Group two: n = 15, l = 600B, the samples differ from one another, including five strings of numbers, five strings of letters, and five strings of Chinese characters.

Group three: n = 30, $l \in \{60B, 120B, 180B, \cdots, 600B\}$, the samples are different from each other, including 10 strings of numbers, 10 strings of letters, and 10 strings of Chinese characters. Different samples of the same kind appear alternately, and the length of the samples of the same kind increases by 60B.

Table 7: Time records for each scheme

	t_A/ms	t_B/ms	t_C/ms
Group One	9.973	10.935	69.814
Group Two	80.784	87.767	628.289
Group Three	108.357	123.633	715.047
Group Four	277.258	328.122	2045.533
Group Five	339.090	386.934	2581.123
Group Six	1042.207	1238.683	7613.630

Group four: n = 90, $l \in \{60B, 120B, 180B, \cdots, 600B\}$, the samples are different from each other, including 30 strings of Numbers, 30 strings of letters, and 30 strings of Chinese characters. Different samples of the same kind appear alternately, and the length of the samples of the same kind increases by 60B.

Group five: n = 60, $l \in \{60B, 120B, 180B, \cdots, 1200B\}$, it includes 20 different samples, each of which occurs 3 times in random order. The length of the sample increases by 60B and is composed of numbers, letters, and Chinese characters.

Group six: n = 1, l = 86079B, letters, Chinese characters, symbols, and English appear randomly in the sample.

The experimental encoding environment is UTF-8. t_A,t_B,t_C is the time using in each scheme and they were recorded respectively, as shown in Table 7. Scheme A takes the shortest time to complete 6 groups of experiments.

Compared with Scheme B, Scheme A has a stronger compressive resistance and is more advantageous in dealing with long samples. D_i is denoted as the difference between the time taken by Scheme B and Scheme A to process per byte in the experiment of group i. In group 3, n = 30, D_3=1.543 us was calculated. In group 4, n = 90, other conditions being equal, D_4=1.713 us was calculated, and it was found that scheme A took less time per byte when processing multiple samples continuously.

$$D_3 = \frac{t_{B_3} - t_{A_3}}{\sum_{i=0}^{n} l_i} = \frac{123.633 - 108.357}{9900} = 1.543\,us$$

$$D_4 = \frac{t_{B_4} - t_{A_4}}{\sum_{i=0}^{n} l_i} = \frac{386.934 - 339.090}{34140} = 1.401\,us$$

In group 6, l = 86079B was the maximum sample length used, D_6=2.283 us, and the difference of processing time per byte was also the maximum.

$$D_6 = \frac{t_{B_6} - t_{A_6}}{\sum_{i=0}^{n} l_i} = \frac{1238.683 - 1042.207}{86079} = 2.283\,us$$

Compared with plan C, Scheme A significantly reduces the time complexity. $\overline{E}_{AC}$ is the average ratio of time taken to complete each experiment in Scheme A and C. Scheme A deals with Bytes while Scheme C deals with

Table 8: The time taken of the derived key for each scheme

	$\bar{t}_A/ms$	$\bar{t}_B/ms$
d_1	158.50	227.69
d_2	190.36	262.39
d_3	204.84	294.60
d_4	177.11	340.47
d_5	190.48	260.33
d_6	176.18	294.58

bits. $\overline{E}_{AC} \equiv 0.138$ accords with the theoretical values and reflects the algorithm optimization.

$$\begin{aligned} \overline{E_{AC}} &= \frac{1}{6}\sum_{i=0}^{6}\frac{t_{A_i}}{t_{C_i}} \\ &= \frac{1}{6}(0.143+0.128+0.152+0.131+0.137) \\ &\approx 0.138. \end{aligned}$$

In conclusion, the SM3 hash algorithm python software implementation scheme designed in this paper reduces the computation time and achieves a better optimization effect. At the same time, from the perspective of the system, the SM3 software implementation scheme designed in this paper has a certain compressive ability, which can meet the practical application requirements of blockchain.

7.3 SM9 Test Analysis

This paper designs a python software optimization implementation of the SM9 identification cipher algorithm. This section will design experiments for key derivation, signature verification, encryption, and decryption to verify the optimization effect. Scheme A, based on bytes structure in python, is used to process data in byte bits, and integer type is used to complete bitwise ecotone or other Boolean operations. At the same time, the key export and hash function are optimized by mathematical reasoning. Scheme B is used for comparison. The operation process involves character data, and the calculation is not optimized by mathematical reasoning.

Take the signature private key as an example to measure the key derivation time of schemes A and B. In the experiment, the user id was randomly generated, with a total of 6 strings of characters within 20 in length, denoted as D= d_1,d_2,d_3,d_4,d_5,d_6 . Due to the influence of hardware condition, external environment, and other factors, the code running time fluctuates in a small range. Therefore, for $\forall d_i \in D$, adopt measures of average 6 times to reduce error. The time taken for each scheme to generate the signature private key was measured, and the average ta tb was calculated. The results are shown in Table 8: The signature and check algorithms were respectively executed to measure the operation times per minute of scheme A and B. The experimental encoding environment is UTF-8. The pending information is the same. For the signature algorithm, the 256 bit length SM3 hash calculation result is used, which directly participates in the signature operation to eliminate the interference of the hash algorithm to the experiment. For the signature check algorithm, the signature used is the above information. Execute continuously and record the completion times of each scheme per minute c_A, c_B, as shown in Table 9. The encryption

Table 9: Comparison of operation times of signature check

	c_A/tpm	c_B/tpm
signature	336	304
signature check	62	59

and decryption algorithms are respectively implemented to measure the encryption and decryption rates of scheme A and B. Randomly generate 10 strings of characters in bytes $N*10^3$,$N \in$ {1,2,3...9,10}. For $\forall N$, we use the program group key, and alternate with six groups of encryption and decryption operations, each time of encryption operation is recorded as e_i,and each time of decryption operation is recorded as d_i,$i \in [1,6]$. Computing the encryption number of bytes per second, and the encryption speed $v = 6*N*10^3/\sum_{i=1}^{6} e_i$. Computing the decryption number of bytes per second, and the decryption rate $w = 6*N*10^3/\sum_{i=1}^{6} e_i$, the results as shown in Table 10:

It can be seen from Figure 10 that the encryption and decryption rates of scheme A are higher than those of Scheme B. Both v_A,w_A remain stable and do not change significantly with the increase of plaintext length. Furthermore, calculating the average $\overline{v}_A = 614.26$,$\overline{w}_A = 370.22$ standard deviation $\sqrt{D(v_A)}$ =31.90,$\sqrt{D(p_A)}$ =23.23. It is difficult for an attacker to infer the length of a key by changing the rate of encryption and decryption.

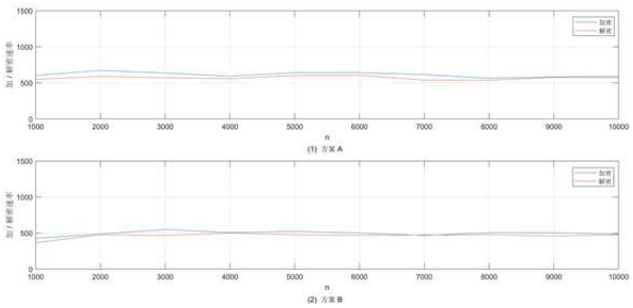

Figure 10: The encryption and decryption rates vary with the length of plaintext

Diffusion requires the relationship between Ming and ciphertext to be as complex as possible, and any small change in the plaintext will make the ciphertext greatly different [9]. Experiments are designed to verify that the security of the encryption algorithm itself is not damaged by the optimization of scheme A. Randomly generate a string of characters of length 100.3 groups of experiments

Table 10: Encryption and decryption rate of each scheme

N	$v_A/(B*s^{-1})$	$w_A/(B*s^{-1})$	$v_B/(B*s^{-1})$	$w_A/(B*s^{-1})$
1	602.05	548.72	426.08	368.30
2	673.63	590.69	486.54	475.37
3	636.12	571.77	547.85	467.43
4	592.52	558.54	507.40	500.41
5	641.66	601.28	523.50	476.30
6	642.47	605.85	500.34	467.72
7	615.41	538.48	466.77	473.19
8	564.05	537.23	505.58	476.03
9	583.13	576.28	504.51	458.77
10	591.59	573.38	482.50	475.33

were conducted to randomly generate characters and replace k in the plaintext,$k \in\{1,2,3\}$. For $\forall k$, six experiments, each plaintext character have the same number change, but the content and position of random. The mean value and standard deviation of character changes are calculated. The discrete distribution is shown in Table 11: In the experiment, the change of plaintext and

Table 11: The ciphertext character changes of scheme A

k	mean ± SD
1	93.50 ± 1.50
2	95.67 ± 1.49
3	94.50 ± 3.10

ciphertext will have great changes. The number of ciphertext character changes is above 90, accounting for more than 90% It can be seen that the plaintext diffusion is good, and the optimization of scheme A does not destroy the security of the algorithm.

To sum up, this section designs experiments to verify the software optimization effect of python for the SM9 identification cipher algorithm. In the key derivation, it takes less time to generate the signature private key with different user identities. In signature verification, experiments were conducted respectively, and the signature and verification algorithms were successively executed, with more operations per minute. In encryption and decryption, the plaintext length is different, multiple encryption and decryption operation, the rate is faster. At the same time, plaintext diffusion is good, security is not damaged.

Conclusion

At present, the supervision of industrial hemp has the problems of low efficiency and low safety. This paper analyzes the actual needs of industrial hemp monitoring and proposes a solution based on the technical background of blockchain and SM series algorithms. First, We use the private chain based on PoW, remove the incentive mechanism. Then we design a block and trading structure according to industrial hemp monitoring requirements. In the block record content, "hash of previous block" is generated by the SM3 hash algorithm to protect the integrity of the block. In the transaction record content, "signature" and "type" are generated by the SM9 identification algorithm respectively, which protect the authenticity and confidentiality of the transaction. In the design, the creation block and the initial transaction have a special nature and are initiated only by the producer. In the initial transaction, the "quantity" corresponds to the total quality of the batch of industrial hemp. Taking the initiation node of the Genesis block as the super source node and the transaction as the arc, a directed graph is constructed to monitor three types of anomalies which are the illegal recipient, insufficient holding, and excessive loss.

In the software implementation, SM3 and SM9 are optimized with the help of mathematical reasoning and Python3 bytes structure, which reduces the computational overhead.

The verification system was established, and the data interaction module, transaction initiation module, and query detection module are corresponding to various functions of the blockchain. It is supplemented by access control and boundary treatment strategies, which serve as an experimental verification platform with complete functions,a beautiful interface, and friendly interaction. In the experiment, in addition to testing various functions, PoW computing overhead and SM3 and SM9 optimization were also tested. PoW calculation overhead is stable and the number of transactions recorded in each block is similar. The optimization effect of SM3 and SM9 is obvious, the time and space overhead of hash, key generation, signature, check, encryption, decryption, and other calculation are reduced, and the security is not broken.

References

[1] T. M. Attard, C. Bainier, M. Reinaud, *et al.*, "Utilisation of supercritical fluids for the effective extraction of waxes and cannabidiol (CBD) from hemp wastes," *Industrial Crops and Products*, vol. 112, pp. 38-46, 2018.

[2] G. Baochang, S. Yufeng, Z. Xu, *et al.*, "Study on the content of cannabidiol in hemp leaves," *Heilongjaing Science*, vol. 9, no. 01, pp. 61-63, 2018.
[3] Z. Fake, *Method and Process for Producing High-Purity Tetrahy Drocannabinol by Chromatography*, CN201910619293.4, Sept. 13, 2019.
[4] K. Jiaqian, Z. Mingsen, M. Xiaokang, G. Jinhu, F. Xuping, K. Hongmei, "Adaptability analysis of different industrial hemp varieties in Shanxi," *Shanxi Agricultural Sciences*, vol. 47, no. 10, pp. 1803-1805, 2019.
[5] W. Jufeng, Y. Xiaoquan, "Optimization of enzyme-assistedsolvent extraction technologyofcannabidiol from hemp leaf," *China Brewing*, vol. 35, no. 04, pp. 79-82, 2016.
[6] H. Miyano, "Addend dependency of differential, linear probability of addition," *IEIcE Trans on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E81-A, no. 1, pp. 106-109, 1998.
[7] C. Ping, T. Siying, "Research on intelligent financial reporting based on blockchain technology," *Friends of Accounting*, no. 05, pp. 156-160, 2020.
[8] W. Rihong, Z. Lifeng, Z. Hang, *et al.*, "A byzantine fault tolerance raft algorithm combines with BLS signature," *Journal of Applied Sciences*, vol. 38, no. 01, pp. 93-104, 2020.
[9] C. E. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal*, vol. 28-4, pp. 656–715, 1949.
[10] Standardization Administration, *Information Security Techniques — SM3 Cryptographic Hash Algorithm*, Standardization Administration, GB/T 32905-2016, 2016.
[11] S. Taoyi, Z. Yunlei, "Comparison of blockchain consensus algorithm," *Computer Applications and Software*, vol. 35, no. 08, pp. 1-8, 2018.
[12] Md A. Uddin, A. Stranieri, I. Gondal, V. Balasubramanian, "Blockchain leveraged decentralized IoT eHealth framework," *Internet of Things*, vol. 9, 2020. DOI:10.1016/j.iot.2020.100159.
[13] H. Wang, H. Qin, M. Zhao, X. Wei, H. Shen, W. Susilo, "Blockchain-based fair payment smart contract for public cloud storage auditing," *Information Sciences*, vol. 519, pp. 348-362, 2020.
[14] G. Xianjin, X. Juncheng, *Method for Extracting Tetrahydrocannabinol from Medical Marijuana*, CN108654134A, Oct. 16, 2018.
[15] W. Xiaorui, Z. Xuechao, "Research and application of blockchain security technology," *Network Security Technology & Application*, no. 02, pp. 29-30, 2020.
[16] L. Yanxu, D. Peng, C. Xiaoguang, *et al.*, "An approach for the analysis of pharmacodynamic interactions and the simulation of combined response," *Chinese Pharmacological Bulletin*, no. 08, pp. 1112-1114, 2007.
[17] Y. Yatao, C. Juliang, Z. Xiaowei, *et al.*, "Privacy preserving scheme in block chain with provably secure based on SM9 algorithm," *Journal of Software*, vol. 30, no. 06, pp. 1692-1704, 2019.
[18] C. Yunchun, L. Haonan, "Study on the construction of reverse supply chain for expired drugs under block chain perspective," *China Pharmacy*, vol. 30, no. 24, pp. 3342-3349, 2019.
[19] G. Zhe, Z. Zhijun, L. Xiaojun, *et al.*, "Hot reflux extraction of cannabidiol from hemp leaves," *China Oils and Fats*, vol. 44, no. 03, pp. 107-111, 2019.
[20] B. Zhenshan, W. Kaixuan, Z. Wenbo, "A practical byzantine fault tolerance consensus algorithm based on tree topological network," *Journal of Applied Sciences*, vol. 38, no. 01, pp. 34-50, 2020.

Biography

The First Author biography. Zijian Ma received the B.S. degree in 2007 and Master degree in 2009 in Industrial engineering from Tsinghua University. He is pursuiting his PH.D degree of Computer and Information Technology in Beijing Jiaotong University now. His research interest includes artificial intelligence and data mining.

The Second Author biography. Zhiqiang Wang, born in China in 1985. He received the Ph.D. degree in information security from Xidian University.He is currently an Assistant Professor with the Department of Computer Science and Technology. His research interests include system security and network security.

The Third Author biography. Hang Wu, born in 1995, is currently a graduate student at Xidian University, majoring in cryptography. Her research direction is hardware Trojan detection technology based on convolutional neural network.

Other Author biography.
Xingyu Guo,he is a undergraduate student in Beijing Electronic Science and Technology Institute.His research direction is network security. Xizhen Wang, born in 1995, is currently a graduate student at Xidian University, majoring in cryptography. His research direction is the research of lattice-based signature schemes.