A Novel GLS Consensus Algorithm for Alliance Chain in Edge Computing Environment

Huijuan Wang[1, *], Jiang Yong[1], Qingwei Liu[2] and Alan Yang[3]

Abstract: Edge computing devices are widely deployed. An important issue that arises is in that these devices suffer from security attacks. To deal with it, we turn to the blockchain technologies. The note in the alliance chain need rules to limit write permissions. Alliance chain can provide security management functions, using these functions to meet the management between the members, certification, authorization, monitoring and auditing. This article mainly analyzes some requirements realization which applies to the alliance chain, and introduces a new consensus algorithm, generalized Legendre sequence (GLS) consensus algorithm, for alliance chain. GLS algorithms inherit the recognition and verification efficiency of binary sequence ciphers in computer communication and can solve a large number of nodes verification of key distribution issues. In the alliance chain, GLS consensus algorithm can complete node address hiding, automatic task sorting, task automatic grouping, task node scope confirmation, task address binding and stamp timestamp. Moreover, the GLS consensus algorithm increases the difficulty of network malicious attack.

Keywords: Alliance chain, consensus algorithm, GLS, data local sharing, arithmetic cross-correlation.

1 Introduction

With the development of information technology, the resource center based computing paradigm is no longer suitable for new network applications. So some paradigms proposed to offload some cloud computing tasks to the network edge. The representative ones are edge computing and fog computing [Lin, Zhou, You et al. (2019); Hui, Zhou, An et al. (2019); Lin, Zhou, An et al. (2018)]. They extend the computing, communicating, or other abilities of cloud computing to the network edge to achieve high bandwidth, low latency, mobility supporting, and location aware advantages.

Owing to the edge computing devices that are widely deployed suffer from security

[1] Information Security Department of the First Research Institute of the Ministry of Public Security of China, Beijing, 100084, China.

[2] College of NBC Defense, Beijing, 100084, China.

[3] Amphenol Assemble Tech, Houston, TX 77070, USA.

[*] Corresponding Author: Huijuan Wang. Email: whj409@163.com.

Received: 19 January 2020; Accepted: 30 April 2020.

CMC. doi:10.32604/cmc.2020.09803 www.techscience.com/journal/cmc

attacks [Su, Lin, Zhou et al. (2015); Hui, Zhou, Xu et al. (2020)]. To deal with this security problem, the blockchain is a suitable choice. The rise of digital currency represented by Bitcoin has caused widespread concern in blockchain technology [Biryukov and Pustogarov (2015); Groth and Kohlweiss (2015)]. Blockchain is the core technology for decentralized digital currency, but also applicable to other areas. The essence of blockchain is a decentralized secure data storage technology. It has solved the technical difficulties of data storage interaction as open, secure, trusted and distributed sharing. Blockchain is essentially a secure, trusted distributed database, or can be defined as a shared and unchangeable distributed accounting system. The blockchain combines several mature computer technologies, such as data encryption, time stamping and distributed consensus, to recognize the distributed and decentralized peer-to-peer transaction, coordination and collaboration between untrusted nodes. The technical feature of the blockchain is a secure distributed storage database. For a large number of well-established database systems in current business applications, such as Oracle and MySQL, the data application that the blockchain needs to solve is to ensure the spontaneity, security, anonymity, and traceability of data interaction under the condition that the write rights are peer at each node and mutual supervision are needed [Moore and Christin (2013); Wijaya, Liu, Steinfeld et al. (2016)].

The blockchain is divided into the public chain, the private chain and the coalition chain according to the conditions and business requirements of the nodes [Reid and Harrigan (2013)]. The participating nodes of the public chain are arbitrary nodes of the whole network. Any computer and computing server can participate voluntarily and can be regarded as a node of the public chain. The Bitcoin system is a public chain blockchain system based on the Proof of Work (POW) consensus protocol [Nakamoto (2008)]. Bitcoin's POW consensus protocol is considered a waste of resources and other public chain consensus protocols such as Proof of Stake (POS) and Delegated Proof of Stake (DPOS) are proposed [King and Nadal (2012); Duong, Lei and Zhou (2016)]. These protocols are capable of saving computing energy, while the cost of attack by the destroyer has become very small, and the security is far less than the POW [Gervais, Karame, Wüst et al. (2016)]. Furthermore, they are based on the size of the equity to determine the size of accounting opportunities of the blockchain nodes, which is lack of fairness. The private chain is an application of a small range of blockchain, and nodes on the private chain are only set according to private organization rules [Forte, Romano and Schmid (2016)]. At present, the application scenarios of the private chain are generally defined within the enterprise or the government [Chen, Feng, Zhang et al. (2019); Xia, Tan, Wang et al. (2019)]. Some database management and auditing tasks are solved. The security requirements of the private blockchain are relatively low, and more demands are spontaneous data reading and writing and interaction. The core value of the private chain is to provide the function that data is securely traceable and cannot be altered. Most of the enterprises and departments now use the alliance chain, which is written by the nodes participating in the alliance members. Compared with the private chain, the participants of the alliance chain are the interactive writing between different departments while that of the private chain are the interactive writing within the department. They are different in management and supervision. The rules for reading and writing permissions on the alliance chain are based on the agreement between the members of the alliance. The nodes participating in the alliance chain need to reach a rule agreement, and the data reading, writing and interaction of the blockchain are completed according to the rule agreement. Currently, the alliance chain that has been formed and put into use includes R3, which is participated by many banks, and hyperledger, which is supported by the Linux Foundation [Juan, Kiayias and Nikos (2015)].

The public chain, private chain, and alliance chain must be implemented according to the consensus algorithm in business implementation [Kiayias and Panagiotakos (2015)]. This paper mainly introduces a generalized Legendre sequence (GLS) [Wang, Wen and Zhang (2013)] consensus algorithm for the alliance chain. The GLS consensus algorithm can implement address hiding, automatic task sorting, task automatic grouping, task node range confirmation, task address binding and time stamping in the alliance chain. (GLS consensus algorithm can complete node address hiding, automatic task sorting, task automatic grouping, task node scope confirmation, task address binding, stamp timestamp). Section 1 introduces the blockchain and briefly introduces the concepts of the public chain, private chain, and alliance chain. Section 2 introduces the basics of designing and validating GLS algorithms and introduces the definition and nature of GLS sequences. GLS algorithms operate the 2-adic ring, which is a finite ring that can correspond to any bit string in a finite field. When designing with this theoretical basis, it can inherit the recognition and verification efficiency of binary sequence ciphers in computer communication and can solve a large number of nodes verification of key distribution issues. Section 3 mainly introduces a new type of GLS consensus algorithm of the alliance chain consensus agreement. The GLS consensus tests the validity and delay using the data block size of the blockchain and the latency of the consensus algorithm. For the specific attacks that blockchains are vulnerable to, such as Distributed Denial of Service (DDOS), link attacks, drop attacks, and false information write attacks, we analyze and verify the resistance of the GLS consensus algorithm in this paper. Section 4 mainly summarizes and proposes ideas for the areas that need to be improved and explores future research directions.

2 Preliminary

This paper mainly introduces the GLS consensus algorithm, whose main theoretical source is the cryptographic anti-attack property of the GLS sequence. The GLS sequence is a Legendre transformed sequence, the generalized Legendre sequence (GLS). The sequence is transformed based on the original sequence on the ring and has an Arithmetic Cross-correlation [Goresky and Klapper (1997)]. The GLS sequence inherits a high level of anti-attack capability and can generate a large number of bearer attack sequences. This extended nature provides a large number of authentication passwords for the consensus protocol. Since the GLS sequence is generated on the ring and can realize multi-dimensional operations, the GLS consensus algorithm can realize address hiding, task node range confirmation, task address binding, and time stamping between nodes in the alliance chain. Next, we introduce some basic knowledge of GLS consensus protocol design and verification. Since this chapter deals with a large number of Finite Ring knowledge, interested readers can refer to Wang et al. [Wang, Wen and Zhang (2013)].

The distribution password of the GLS consensus protocol is mainly derived from the

GLS sequence, and the GLS sequence is mainly generated from the Legendre transformation of the primitive sequence $Z/(p^e)$ over ring. The period of the N-th order primitive sequence is $p^{e-1}(p^n-1)$ ($\underline{a}=\{a(t)\}_{t\geq 0}$ has the least period $p^{e-1}(p^n-1)$). The GLS consensus algorithm mainly uses an important anti-attack property of the GLS sequence, the Arithmetic Cross-correlation. Due to the importance of this property in this paper, we describe it in detail here.

2.1 2-adic integer and arithmetic cross-correlation

Let binary sequence $\underline{s}=s(0),s(1),s(2),s(3),\cdots$ have least period T with pre-period $t_0>0$, so that $s(t+T)=s(t)$ with $t\geq t_0$. If $t\geq t_0$ we denote the sequence $\underline{s}$ as an eventually periodic sequence, if $t_0=0$ we denote the sequence $\underline{s}$ as a strictly periodic sequence.

A *2-adic* integer is a formal power series $\varpi=\sum_{t=0}^{\infty}s(t)\cdot 2^t$, with $s(t)\in\{0,1\}$. The set Z_2 of the *2-adic* integers forms a ring under the operations of addition and multiplication with carry. We denote the string $000\ldots$ as merely, and the string $100\ldots$ as 1. Besides, we define that $1+2+2^2+\cdots=-1$; that is, the infinite string $111\ldots$ is a base-2 expansion of a negative integer -1.

Specifically, addition of the Z_2 integers is given by

$$\sum_{t=0}^{\infty}s_1(t)\cdot 2^t+\sum_{t=0}^{\infty}s_2(t)\cdot 2^t=\sum_{t=0}^{\infty}s_3(t)\cdot 2^t \tag{1}$$

If there are carry integers $d_0,d_1,d_2,\cdots$, such that $d_0=0$, and for all $t\geq 0$, we have

$$s_1(t)+s_2(t)=s_3(t)+2d_{t+1}-d_t \tag{2}$$

Similarly, the multiplication of the Z_2 integers is given by

$$(\sum_{t=0}^{\infty}s_1(t)\cdot 2^t)\cdot(\sum_{t=0}^{\infty}s_2(t)\cdot 2^t)=\sum_{t=0}^{\infty}s_3(t)\cdot 2^t \tag{3}$$

If there are carry integers $d_0,d_1,d_2,\cdots$, such that $d_0=0$, and for all $d_0=0$, we have

$$s_1(t)\cdot s_2(0)+s_1(t-1)\cdot s_2(1)+\cdots+s_1(0)\cdot s_2(t)=s_3(t)+2d_{t+1}-d_t \tag{4}$$

Note that in Z_2, as

$$\begin{array}{r} 1000\cdots \\ \oplus \\ 1111\cdots \\ \hline 0000\cdots \end{array} \tag{5}$$

We define that $1+2+2^2+\cdots=-1$. Then the corresponding subtraction of 2-adic numbers is

$$\sum_{t=0}^{\infty}s_1(t)\cdot 2^t-\sum_{t=0}^{\infty}s_2(t)\cdot 2^t=\sum_{t=0}^{\infty}s_1(t)\cdot 2^t+\sum_{t=0}^{\infty}2^t\cdot\sum_{t=0}^{\infty}s_2(t)\cdot 2^t \tag{6}$$

It follows that Z_2 contains all the integers.

Let $q=1+q_1 2+q_2 2^2+\cdots+q_r 2^r$ be an odd integer, then the negative integer $-q$ is associated to the product

$$-q=(1+2+2^2+2^3+\cdots)(1+q_1 2+q_2 2^2+\cdots+q_r 2^r) \tag{7}$$

In Z_2, the formal power series $-q$ has a unique(multiplicative) inverse

$$(-q)^{-1}=1\cdot 2^0+b_1\cdot 2^1+b_2\cdot 2^2+b_3\cdot 2^3+\cdots \tag{8}$$

Thus the ring Z_2 contains every rational number h/q provided q is odd.

Proposition 1 [Klapper and Goresky (1997)]: There is a one-to-one correspondence between rational numbers $\varpi=h/q$ (where q is an odd number) and eventually periodic binary sequences $\underline{s}$, which associates to each rational number ϖ and the bit sequence $\underline{s}=s(0),s(1),s(2),\cdots$ of its $2-adic$ expansion. The sequence $\underline{s}$ is strictly periodic if and only if $\varpi\le 0$ and $|\varpi|<1$.

In this correspondence, we use the operations in Z_2 to introduce the arithmetic cross-correlation. Recall that the ordinary cross-correlation with shift τ of two strictly sequences $\underline{s_1}$ and $\underline{s_2}$ of period T can be defined either as the sum $\underline{s_2}^\tau=s_2(0+\tau),s_2(1+\tau),s_2(2+\tau),\cdots$ or as the number of zeros minus the number of ones in one period of the bitwise exclusive-or of $\underline{s_1}$ and the τ shift of $\underline{s_2}$, where the τ shift of $\underline{s_2}^\tau$ is denote as $\underline{s_2}^\tau=s_2(0+\tau),s_2(1+\tau),s_2(2+\tau),\cdots$. The arithmetic cross-correlation is the with-carry analog, and is given by the following definition.

Definition 1 [Goresky and Klapper (1997)]: Let $\underline{s_1}$ and $\underline{s_2}$ be two strictly binary periodic sequences with period T, and let $0\le\tau<T$ and $\underline{s_2}^\tau$ be the τ shift of $\underline{s_2}$. Denote ϖ_1 and ϖ_2^τ as the $2-adic$ integers corresponding to the sequences $\underline{s_1}$ and $\underline{s_2}^\tau$. Then, the corresponding sequence $\underline{s_3}$ of $\varpi_1-\varpi_2^\tau$ is strictly periodic or eventually periodic, and its period divides T. The shift arithmetic cross-correlation $C^a_{\underline{s_1},\underline{s_2}}(\tau)$ of $\underline{s_1}$ and $\underline{s_2}$ is the number of zeros minus the number of ones in one period of length T of $\underline{s_3}$.

As in the Definition 1, it is shown that the arithmetic cross-correlation of strictly periodic sequences $\underline{s_1}$ and $\underline{s_2}$ satisfy

$$C^a_{\underline{s_1},\underline{s_2}}=\sum_{t=0}^{T}(-1)^{s_3(t)} \tag{9}$$

where $\sum_{t=0}^{\infty}s_1(t)\cdot 2^t+\sum_{t=0}^{\infty}s_2(t)\cdot 2^t=\sum_{t=0}^{\infty}s_3(t)\cdot 2^t$.

If $\underline{s_1}$ and $\underline{s_2}^\tau$ are distinct for all $\tau\ge 0$, then $\underline{s_1}$ and $\underline{s_2}$ cyclically distinct. If $\underline{s_1}$ and $\underline{s_2}$ are cyclically distinct and satisfy $C^a_{\underline{s_1},\underline{s_2}}(\tau)=0$, then $\underline{s_1}$ and $\underline{s_2}$ are said to have optimal arithmetic cross-correlation. For instance, the sequences

$\underline{s_1} = 111101000010011101100010111101000010111101100010011101000010\cdots$ and $\underline{s_2} = 010011011001001101100100110110010011011001001101100100110110\cdots$ have optimal arithmetic cross-correlation as $\underline{s_1} - \underline{s_2}$ has the balanced property over a period in the 2-adic ring. We have defined $\underline{s_1} - \underline{s_2} = \varpi_1 - \varpi_2$ as the operation in Z_2.

2.2 GLS sequences

We introduce the nature of the GLS sequence, the detailed proof, please refer to Wang et al. [Wang, Wen and Zhang (2013)]. For each integer n, p is satisfied ($p > 7, 4/p - 1$) on the Galois ring $Z/(p^e)$. There is a maximum period sequence $\underline{a} = \{a(t)\}_{t \geq 0}$, $\alpha(t) \in GR(p^e, n)$, sequence $\underline{a} = \{a(t)\}_{t \geq 0}$ maximum period is $p^{e-1}(p^{n-1} - 1)$. Ring $Z/(p^e)$ Maximum sequence $\underline{a} = \{a(t)\}_{t \geq 0}$ composition of the sequence set is defined as $\mathrm{A}_{(p^e, n)}$. The GLS sequence $\underline{s} = \{s(t)\}_{t \geq 0}$ generated by $\underline{a} = \{a(t)\}_{t \geq 0}$ is defined by the following:

$$s(t) = \begin{cases} 1 & a(t) \in C_0 \cup D_0 \\ 0 & a(t) \in C_1 \cup D_1 \end{cases} \tag{10}$$

where $C_0 = \left\{ a_t \in Z/(p^e) \ \middle|\ a(t) \bmod p = 0 \quad \text{and} \quad t \bmod 4 = 0 \quad \text{or} \quad t \bmod 4 = 3 \right\}$;

$C_1 = \left\{ a_t \in Z/(p^e) \ \middle|\ a(t) \bmod p = 0 \quad \text{and} \quad t \bmod 4 = 0 \quad \text{or} \quad t \bmod 4 = 2 \right\}$;

$D_0 = \left\{ a_t \in Z/(p^e) \ \middle|\ a(t) \bmod p \neq 0 \quad \text{and} \quad a(t) \textit{ is quadratic residual} \right\}$;

$D_1 = \left\{ a_t \in Z/(p^e) \ \middle|\ a(t) \bmod p \neq 0 \quad \text{and} \quad a(t) \textit{ is Non-quadratic residual} \right\}$.

The quadratic residue of the element on the ring is for the element a, exist an element $r \in Z/p^e$, Satisfying $r^2 \bmod p^e = a$; Non-quadratic residual means that no element exists $r \in Z/p^e$, Satisfying $r^2 \bmod p^e = a$.

The GLS sequences generated by the largest periodic sequence of integers n in the ring $Z/(p^e)$ form a binary periodic sequence set. The largest periodic sequence of these binary periodic sequences make up the set $S(p^{e_1}, n)$, in which $S(p^{e_1}, n)$ is a GLS collection. GLS set $S(p^{e_1}, n)$ sequence $\underline{s} = \{s(t)\}_{t \geq 0}$ satisfy:

$\underline{s} = \{s(t)\}_{t \geq 0}$ is a binary periodic sequence, period is $2 \cdot p^{e-1}(p^n - 1)/(p - 1)$;

Any binary periodic sequence corresponds to a 2-adic correlation number [Klapper and Goresky (1997)], for any two binary periodic sequence $\underline{s_1}$, $\underline{s_2}$ corresponds to two 2-adic numbers τ_1, τ_2, τ_1-τ_2 get another correlation number τ_3, τ_3 corresponds to the binary periodic sequence in a cycle 0, 1 number difference recorded as $C_{\underline{s_1}\underline{s_2}}{}^{a}$. Any two sequences $\underline{s_1}$, $\underline{s_2}$ in GLS set $S(p^{e_1}, n_1)$, $S(p^{e_1}, n_2)$ ($e_1 \neq e_2, n_1 \neq n_2$) satisfy $C_{\underline{s_1}\underline{s_2}}{}^{a} = 0$.

3 GLS consensus algorithm

In the last section, we make a detailed analysis of the consensus algorithm applied to the affiliate chain. In this section, we introduce a new affiliate chain consensus algorithm-GLS consensus algorithm based on GLS. The GLS sequence set represents a class of Legendre sequences that satisfy some properties by themselves and can fulfill some of the consensus conditions in the coalition chain. The GLS consensus algorithm can realize the function of address concealment, task automatic sorting, task automatic grouping, task node scope confirmation, task address binding, time stamping and so on.

3.1 GLS consensus algorithm

In the application scenario of the federation chain, the nodes involved in the task are generally based on the trust. Each node can design a list of the tasks to be completed according to the actual situation, and dynamically allocate a large Prime P. GLS consensus algorithm has the following steps (Fig. 1).

Step 1: The requesting node applies for completing the task P, and the requesting node broadcasts to all the nodes in the network. Each node receives the task invitation, and if it agrees to participate in the task, the following steps are performed, and does not participate in the task to abandon the subsequent verification;

Step 2: Participate in the task node i, according to the local address to generate unique characters n_i , and add the consent to perform the task password e_i to generate character pairs（n_i, e$_i$）;

Step 3: Node i generates a GLS set $S(p^{e_1}, n_i)$ according to the task P and the generated character pair（n_i, e$_i$）, and randomly selects a binary sequence $\underline{s}_{\mathrm{i}}$ of the whole network to broadcast;

Step 4: Receive the sequence sent by another participating node;

Step 5: Verification $C_{\underline{s}_{\mathrm{i}}\underline{s}_{\mathrm{j}}}{}^{a}$, if the verification $C_{\underline{s}_{\mathrm{i}}\underline{s}_{\mathrm{j}}}{}^{a} \neq 0$ ends, if $C_{\underline{s}_{\mathrm{i}}\underline{s}_{\mathrm{j}}}{}^{a}=0$ continue to the next step;

Step 6: Through the periodicity of the sequence $\underline{s}_{\mathrm{j}}$, verify that the periodic rule of P and T_{j} are consistent with the GLS sequence, and discard the verification if it does not meet the condition. If the match is satisfied, the node i obtains the received sequence by the prime number P;

Step 7: Confirm（n_{j}, e$_{\mathrm{j}}$）whether it is legal, if the task cooperation.

The GLS consensus algorithm proves process is as follows:

Proof: The initiator of the execution task chooses a large prime P to form the task number. Broadcast alliance chain members and add a timestamp. Agree to join node i, calculate pairs（n_{j}, e$_{\mathrm{j}}$）according to consensus protocol algorithm. Where n represents the corresponding address code, e_i indicates that the user agrees to join the task password (including time stamp).

Node i generates GLS sequence sets $S(p^{e_1}, n_i)$, any sequence in GLS contains the current task information P, address information n_i, user password e_i.

Node i receives the sequences of tasks from other nodes with this time stamp. First of all, verification ${C_{\underline{s}_{\mathrm{i}}\underline{s}_{\mathrm{j}}}}^{a}$. When sequence $\underline{s}_{\mathrm{j}}$ belongs to the GLS sequence set $S(p^{e_1}, n_i)$, satisfy ${C_{\underline{s}_{\mathrm{i}}\underline{s}_{\mathrm{j}}}}^{a}=0$; if ${C_{\underline{s}_{\mathrm{i}}\underline{s}_{\mathrm{j}}}}^{a} \neq 0$, the received sequence is definitely not issued by the task node, to give up the next step verification. Due to the nature of the GLS sequence, it cannot guarantee that A does not belong to the GLS sequence set $S(p^{e_1}, n_i)$ must get ${C_{\underline{s}_{\mathrm{i}}\underline{s}_{\mathrm{j}}}}^{a} \neq 0$. So need to verify the next step: the period T_{j} of sequence $\underline{s}_{\mathrm{j}}$. After the received sequence satisfies Step 5, perform Step 6 verification. The received sequence $\underline{s}_{\mathrm{j}}$ results in a period T_{j}, if the sequence $\underline{s}_{\mathrm{j}}$ belongs to the GLS set $S(p^{e_1}, n_i)$, then the sequences of the period T_{j} satisfy

$$T_{\mathrm{j}}=2 \cdot p^{\mathrm{e\text{-}1}}(p^{\mathrm{n}}-1)\big/(p-1) \tag{11}$$

Cyclic division of prime P for T_{j}, get

$$T_{\mathrm{j}}=P^{\mathrm{e_j}\text{-}1}\bullet N_{\mathrm{j}} \tag{12}$$

for N_{j}, we have

$$\mathrm{n}_{\mathrm{j}}=\log_P^{(P-1)N_{\mathrm{j}}+1} \tag{13}$$

If the operation is established, we can get an integer, then the sequence $\underline{s}_{\mathrm{j}}$ can be identified as a task sequence sent by the node j, and get the node address and password（n_{j},e_j）.

Node i can receive all the time-stamped sequence, and filter out the task node, get the address and password of the participating node. Node i can automatically arrange the order of tasks, and get other nodes to participate in mission proof.

The GLS consensus algorithm can achieve the following consensus:

Hidden addresses: nodes involved in the task address can be hidden, the use of the algorithm is the address corresponding to the character n, the receiving node can only verify that n is legal, and cannot restore the real address.

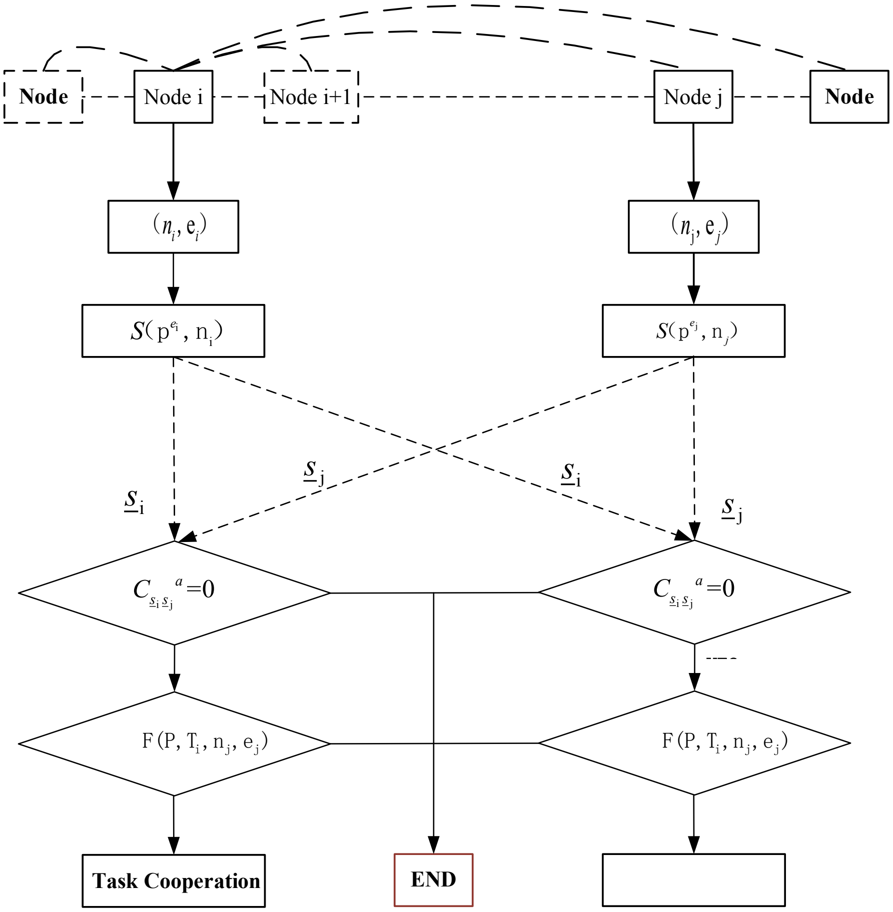

Figure 1: The flow chart of GLS consensus algorithm

Automatic sorting: If the task needs to be sorted to complete the task, GLS consensus algorithm can complete the random order. The address character *n* task P binding operation, according to the address of each node after the size of the characters are sorted.

Confirm the scope of the task: Node *i* can receive the entire network to send the sequence of each node, and through the verification get the number of participating nodes, and automatically generate node list.

The tasks are automatically grouped: The whole network can perform multiple tasks at the same time. The whole network nodes are automatically grouped by using GLS. When node *i* chooses to participate in task P, after receiving the sequence sent by other nodes, the algorithm verifies whether the sequence satisfies task P (Step 6), If satisfied, then grouped into a group, if not, then for other task nodes, not grouped in a group, to achieve automatic task grouping.

Task binding plus timestamp: The node can be a simple task P, signature generated signature *e*, *e* contains timestamps and nodes to agree to the task of tampering with the agreement, through the GLS algorithm to generate a sequence of network broadcasts, other nodes across the network receive sequence at the same time, also receive *e*, that is involved in the signature of the task node.

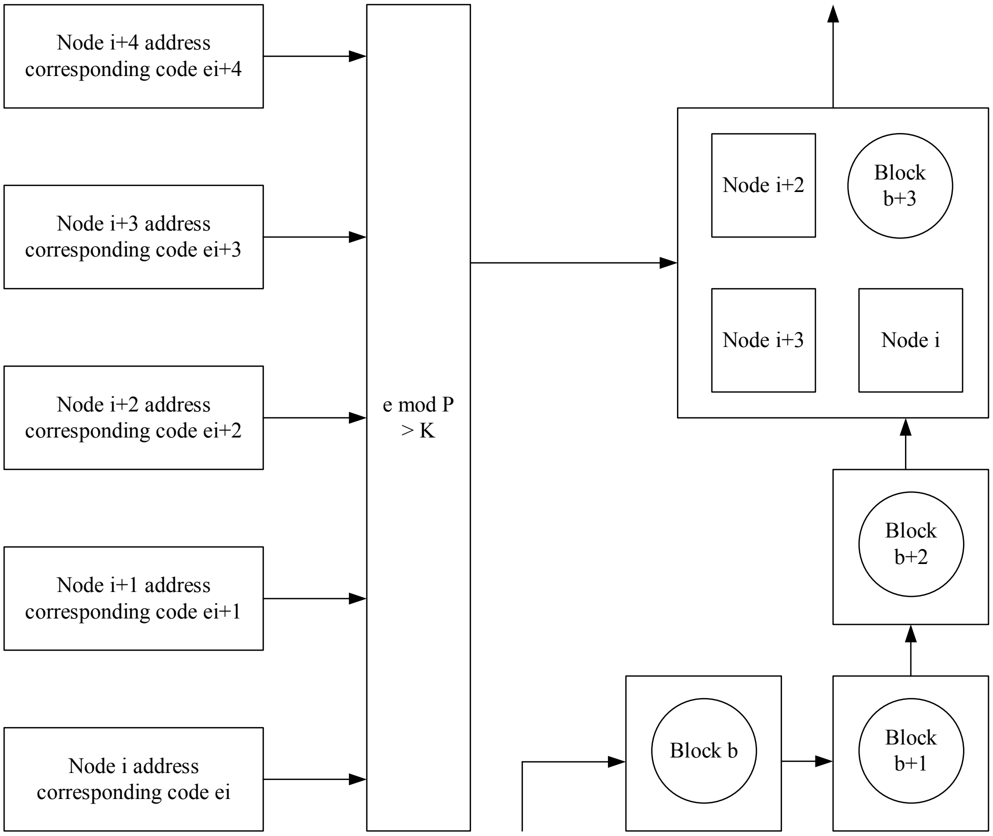
Node i+4 address corresponding code ei+4
Node i+3 address corresponding code ei+3
Node i+2 address corresponding code ei+2
Node i+1 address corresponding code ei+1
Node i address corresponding code ei
e mod P > K
Node i+2
Block b+3
Node i+3
Node i
Block b+2
Block b
Block b+1

Figure 2: Block generation

The Block generation process (Fig. 2) is as follows:

- The choice of write nodes: you can write a random number distribution algorithm, the participating nodes randomly distributed within the prime number P_N, node generated address corresponding code e_i, modulus P_N, the number of e_i is greater than a certain number of K, This node i is set as a write node (which may be multiple);
- The task is completed to form a task cycle, the task process and results written into the node selected in the previous step, similar to Ethereum in the fragment;
- Each write node to form a block, according to the task to complete the time to connect the previous block.

3.2 Functional verification of the GLS consensus algorithm

The process of the GLS Consensus Protocol validity is as follows.

The GLS consensus tests the validity and delay using the data block size of the blockchain and the latency of the consensus algorithm. Transaction Per Second (TPS) and consensus algorithm delay determines the block generation time and the number of verifications per second of the consensus algorithm. The simulation environment of the GLS consensus algorithm is written in Java language, and simulates one data generation process and nine consensus execution processes in a single machine environment. System operating environment: Intel Core m7-6Y75 1.51 GHz CPU, 8G memory and the Ethereum environment. During the simulation experiment, the data generation module continuously sends requests to the consensus module, which executes the GLS consensus algorithm. The Practical Byzantine Fault Tolerance (PBFT) and GLS consensus algorithm can provide a large number of data verification functions. Compared with the PBFT algorithm, the write rate of the GLS consensus algorithm can reach the general blockchain throughput level, as shown in Fig. 3.

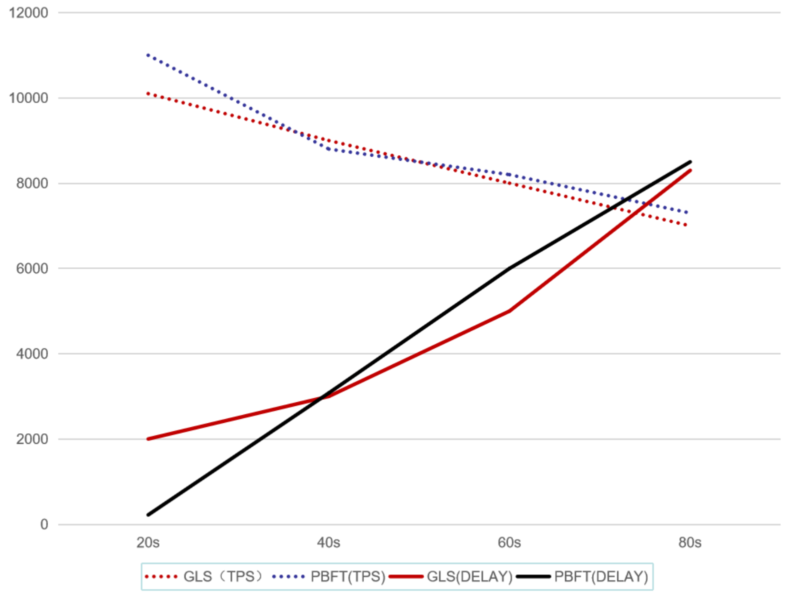

Figure 3: The relationship of TPS and DELAY generation time

Security analysis of the GLS consensus protocol is as follows.

For the specific attacks that blockchains are vulnerable to, such as Distributed Denial of Service (DDOS), link attacks, drop attacks, and false information write attacks, we analyze and verify the resistance of the GLS consensus algorithm.

DDOS: The GLS consensus algorithm can provide a large number of verification keys, and each of the n nodes can generate $p^{e\text{-}1}(p^{n}-1)/(p-1)$ verification keys with very little verification time on the ring, as shown in Fig. 3. DDOS attacks require a lot of computational power, and the GLS consensus algorithm increases the difficulty of DDOS attacks.

Link attack: Link attack means that the attacker writes the blockchain into the node and uses the same ID link to find the real identity corresponding to the anonymous node. For this attack, the GLS consensus algorithm can generate dynamic verification factors based on addresses, which is extremely resistant to link attacks.

Drop attack: A drop attack is an attack that attempts to get a special node and discard or modify other member information. The GLS consensus randomly sets the random number P of the node. After the random number is generated, the permissions of each node are the same, and the attack environment is not provided for the drop attack.

False information writing: Information writing between nodes can increase the node verification factor for encryption, bind information and nodes, and find problems in time to isolate problem nodes, which can improve resistance.

4 Conclusion

The widespread concern of blockchain technology in edge computing has led many departments and industries to consider using blockchain to solve some data security problems, but the development of blockchain is not yet mature. Blockchain now the main problem is the calculation of the data stored in the process of communication problems. Due to the design of blockchain security and the limitation of open source code writing of the original bitcoin system, the currently applicable blockchain can only store data hash. On the other hand, blockchain has some problems in practical application and deployment. Some code on the participating nodes also has some problems, mainly because the blockchain code is difficult to develop and skilled architects are needed in the deployment of blockchain Operate. This article mainly analyzes some consensus algorithms in the coalition chain, and proposes a new consensus algorithm for the coalition chain. As more and more applications of the blockchain, the blockchain consensus algorithm will be more perfect for the more applications of the blockchain.

Funding Statement: We gratefully acknowledge anonymous reviewers who read drafts and made many helpful suggestions. This work is supported by the National Key Research and Development Program No. 2018YFC0807002.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Biryukov, A.; Pustogarov, I. (2015): Bitcoin over tor isn't a good idea. *IEEE Symposium on Security and Privacy*, pp. 122-134.

Chen, W. J.; Feng, G.; Zhang, C.; Liu, P. Z.; Ren, W. et al. (2019): Development and application of big data platform for garlic industry chain. *Computers, Materials & Continua*, vol. 58, no. 1, pp. 229-248.

Duong, T.; Lei, F.; Zhou, H. S. (2016): 2-hop blockchain: combining proof-of-work and proof-of-stake securely. https://eprint.iacr.org/2016/716.

Forte, P.; Romano, D.; Schmid, G. (2016): Beyond bit-coin-part II: block-chain-based systems without mining. http://st13.reshaem.net/tasks/task_173057.pdf.

Gervais, A.; Karame, G. O.; Wüst, K.; Glykantzis, V.; Ritzdorf, H. et al. (2016): On the security and performance of proof of work blockchains. *ACM SIGSAC Conference*, pp. 3-16.

Goresky, M.; Klapper, A. (1997): Arithmetic cross correlations of feedback with carry shift register sequences. *Information Theory IEEE Transactions*, vol. 43, no. 4, pp. 1342-1345.

Groth, J.; Kohlweiss, M. (2015): One-out-of-many proofs: or how to leak a secret and spend a coin. *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, Berlin, Heidelberg, pp. 253-280.

Hui, H. W.; Zhou, C. C.; An, X. S.; Lin, F. H. (2019): A new resource allocation mechanism for security of mobile edge computing system. *IEEE Access*, vol. 7, pp. 116886-116899.

Hui, H. W.; Zhou, C. C.; Xu, S. G.; Lin, F. H. (2020): A novel secure data transmission scheme in industrial internet of things. *China Communications*, vol. 17, no. 1, pp. 73-88.

Juan, G.; Kiayias, A.; Nikos, L. (2015): The bitcoin backbone protocol: analysis and applications. *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, Berlin, Heidelberg, pp. 281-310.

Kiayias, A., Panagiotakos, G. (2015): Speed-security tradeoffs in blockchain protocols. http://eprint.iacr.org/2015/1019.

King, S.; Nadal, S. (2012): PPcoin: peer-to-peer crypto-currency with proof-of-stake. *Self-Published Paper*.

Klapper, A.; Goresky, M. (1997): Feedback shift registers, 2-adic span, and combiners with memory. *Journal of Cryptology*, vol. 10, no. 2, pp. 111-147.

Lin, F. H.; Zhou, Y. T.; An, X. S.; You, I.; Choo, K. K. R. (2018): Fair resource allocation in an intrusion-detection system for edge computing: ensuring the security of internet of things devices. *IEEE Consumer Electronics Magazine*, vol. 7, no. 6, pp. 45-50.

Lin, F. H.; Zhou, Y. T.; You, I.; Lin, J. Z.; An, X. S. et al. (2019): Content recommendation algorithm for intelligent navigator in fog computing based IoT environment. *IEEE Access*, vol. 7, pp. 53677-53686.

Moore, T.; Christin, N. (2013): Beware the middleman: empirical analysis of bitcoin exchange risk. *International Conference on Financial Cryptography and Data Security*, Springer, Berlin, Heidelberg, pp. 25-33.

Nakamoto, S. (2008): Bitcoin: a peer-to-peer electronic cash system in consulted. https://bitcoin.org/bitcoin.pdf.

Reid, F.; Harrigan, M. (2013): An analysis of anonymity in the bitcoin system. *Security and Privacy in Social Networks*, Springer, New York, pp. 197-223.

Su, J. T.; Lin, F. H.; Zhou, X. W.; Lü, X. (2015): Steiner tree based optimal resource caching scheme in fog computing. *China Communications*, vol. 12, no. 8, pp. 161-168.

Wang, H. J.; Wen, Q. Y.; Zhang, J. (2013): GLS: New class of generalized Legendre sequences with optimal arithmetic correlation. *RAIRO-Theoretical Information and Applications*, vol. 47, no. 4, pp. 371-388.

Wijaya, D. A.; Liu, J. K.; Steinfeld, R.; Sun, S.; Huang, X. Y. (2016): Anonymizing bitcoin transaction. *International Conference on Information Security Practice and Experience.* Springer International Publishing, pp. 271-283.

Xia, Z. Q.; Tan, J. J.; Wang, J.; Zhu, R. L.; Xiao, H. G. et al. (2019): Research on fair trading mechanism of surplus power based on blockchain. *Journal of Universal Computer Science*, vol. 25, no. 10, pp. 1240-1260.