Computing and Informatics, Vol. 45, 2026, 777–799, doi: 10.31577/cai_2026_4_777

POLICY-HIDING ATTRIBUTED-BASED ACCESS CONTROL SCHEME FOR IIOT DEVICE SECURE REMOTE OPERATION

Linghui Meng

Key Lab of Education Blockchain and Intelligent Technology
Ministry of Education, Guangxi Normal University
Guilin, China
&
Guangxi Key Lab of Multi-Source Information Mining and Security
Guangxi Normal University
Guilin, China
e-mail: linghuimeng00@seu.edu.cn

Feng Yu*

Key Lab of Education Blockchain and Intelligent Technology
Ministry of Education, Guangxi Normal University
Guilin, China
&
Guangxi Key Lab of Multi-Source Information Mining and Security
Guangxi Normal University
Guilin, China
&
State Key Laboratory of Nuclear Power Safety
Monitoring Technology and Equipment
Shenzhen, 518172, China
e-mail: yufeng@gxnu.edu.cn

Daicen Jiang

Southern Power Grid Supply Chain (Guangxi) Co., Ltd.
Nanning, Guangxi, China
e-mail: nfdwgylyngsxf@csg.cn

Jing Xi

Key Lab of Education Blockchain and Intelligent Technology
Ministry of Education, Guangxi Normal University
Guilin, China
&
Guangxi Key Lab of Multi-Source Information Mining and Security
Guangxi Normal University, Guilin, China
e-mail: HaiboYe@stu.gxnu.edu.cn

Shenglin Cao

Ningxia Normal University
Guyuan, China
e-mail: nxcsl@nxnu.edu.cn

Zhihua Zeng

State Key Laboratory of Nuclear Power Safety
Monitoring Technology and Equipment
Shenzhen, 518172, China
e-mail: zengzhihua@cgnpc.com.cn

Abstract. IIoT devices are the foundation of the Industrial Internet of Things (IIoT), including functions such as environmental perception, data processing, and remote operation. One core advantage of IIoT devices is their remote operation capability, allowing operators to control them via smart devices from distant locations. While remote operation greatly enhances flexibility, the resulting security issues must not be overlooked. Instructions can be tampered with during transmission. Existing devices can verify the sender's identity, but a legitimate sender may still issue non-compliant instructions, impacting safety and efficiency of industrial production. Most existing access control schemes for IIoT devices are unsuitable for secure remote operations. They focus on data exchange between devices, ignoring compliance of data content and sender. This paper proposes a policy-hiding attribute-based access control scheme for secure remote operation of IIoT devices. It provides fine-grained access control while ensuring safe operation by verifying sender compliance and instruction content through a compliance policy. Addition-

* Corresponding author

ally, policy hiding protects sender privacy. Furthermore, multi-receiver signcryption is employed to prevent unauthorized tampering with the instructions during transmission. The security analysis substantiates the security of our scheme, while the performance analysis demonstrates its applicability in IIoT scenarios.

Keywords: Access control, IIoT device, secure remote operation, policy-controlled signature, policy-hiding

Mathematics Subject Classification 2010: 94A60, 94A62, 68M11

1 INTRODUCTION

Industrial Internet of Things (IIoT) devices are the supporting information infrastructure of the modern intelligent manufacturing industry. In actual industrial production, operators and equipment often have a significant physical distance between them, necessitating the use of IIoT devices with remote operation capabilities. Operators, as senders, use smart devices like mobile phones and computers to remotely operate IIoT devices by sending instructions. There are some security problems in remote operation, such as sending non-compliant instructions and instruction tampering.

At present, some scholars have researched the security of equipment and related instruction data in remote operations [1, 2, 3, 4], inferring the legality of instruction data through the legitimacy of the sender. However, when a legitimate sender sends non-compliant instructions, judging solely by the sender's identity can lead to non-compliant remote operations. We can determine whether the instruction data is compliant through a compliance policy. But the compliance policy may lead to information leakage, necessitating its concealment. Ensuring the security of data transmission is crucial, as attackers may tamper with instructions after compliant verification, causing devices to exhibit incorrect behavior.

To solve the above problems, we propose a policy-hiding attribute-based access control scheme for IIoT device secure remote operations (PAAC-IIoT). The instructions and the sender are verified through a control table comprising multiple hidden compliance policies, ensuring the secure operation of industrial production. The main contributions of this scheme are as follows:

- Design a policy-hiding, policy-controlled signature for compliant verification of instruction data. It achieves fine-grained access control for the sender and its related operations. LSSS matrix and Garbled Bloom Filter (GBF) are employed to conceal the compliance policy and safeguard the sender's privacy.
- To prevent tampering and interception in transmission, a multi-verifier signcryption algorithm is used. Employing password and biometric authentication for secure user login.

- Security analysis shows that our scheme is unforgeable under the Random Oracle model. Experimental tests indicate that our PAAC-IIoT scheme has lower computational overhead in instruction transmission compared to other schemes, with the overhead of other algorithms being within a reasonable range.

2 RELATED WORK

Recent studies have proposed various advanced access control methods in the field of IIoT to address growing security challenges. Among them, Wang et al. [4] proposed an attribute-based access control (ABAC) method combining multi-level hash authentication and control process modeling of time and task logic to effectively prevent unauthorized access and illegal operations. Furthermore, Liu et al. [5] proposed a BP-AKAA scheme using blockchain and zero-knowledge proofs to address cross-domain authentication trust, protect device identity, and support privacy authentication, key agreement, and access control. Usman et al. [6] proposed using supervised machine learning with SCADA systems to enhance IIoT access control with neural networks and extreme learning machines. Zhang et al. [7] designed a decentralized cross-domain access control model using a master-slave chain and reputation mechanism to improve IIoT access control. Mandal et al. [8] proposed a certificateless-signcryption three-factor user access control scheme for IoT, enabling authorization, authentication, and revocation of users, and direct interaction with devices. These studies highlight advancements in IIoT access control, ensuring data and resource protection.

Many access control schemes proposed in the literature are either insecure against various known attacks or inefficient in communication processing. They often overlook rights management for different instructions across devices in industrial control system scenarios of the Industrial Internet. To ensure fine-grained management of instruction permissions and prevent malicious adversaries from hijacking IoT devices due to coarse-grained permission policies, thereby causing losses in industrial production. Additionally, blockchain technology is incorporated into the scheme to store instruction data.

3 PRELIMINARIES

3.1 Composite Order Biliner Groups

Boneh et al. [9] proposed a method to construct linear groups of compound order. In this scheme, three-prime order linear groups are selected to construct policy-controlled signatures within them. Let G and G_T denote cyclic multiplicative groups with the same order $N = pqr$. Within G_T, there exist three subgroups: G_p, G_q, and G_r, with generators g_p, g_q, and g_r respectively.Let $e : G \times G \to G_T$ presents three primes composite order bilinear groups mapping.The following four properties exist for this mapping:

- *Bilinearity*: For all $g_a, g_b \in G$, $c, d \in Z_p$, exist $e(g_a^c, g_b^d) = e(g_a, g_b)^{cd}$.
- *Nondegenerate*: There exist $g_a \in G$ such that the order of $e(g_a, g_a)$ in G_T is N.
- *Computability*: For all $g_a, g_b \in G$ there exist valid functions to compute $e(g_a, g_b)$.
- *Orthogonality*: For any $g_p \in G$ and $g_r \in G$, $e(g_p, g_r) = 1$ always holds.

3.2 Bloom Filter

The Bloom filter (BF) is a space-efficient probabilistic data structure introduced by Burton Howard Bloom in 1970 [10]. It is utilized for determining the presence of an element in a set. Bloom filters employ a fixed-length bit array and multiple hash functions. Upon adding an element to the set, the element is hashed to multiple positions in the array through various hash functions, setting these positions to 1. When querying whether an element exists in the set, the hash function also calculates the corresponding positions in the bit array. If all corresponding positions are 1, the element is deemed to exist in the set. If any position is 0, the element is determined not to be in the set.

3.3 Fuzzy Extractor

The Fuzzy Extractor (FE) plays a critical role in the security applications of biometrics. This technology primarily transforms the biometric data of the sender or receiver in the proposed scheme into an encrypted key form for secure storage and verification. It consists of two algorithms: a key generation algorithm and a key reconstruction algorithm.

Key generation algorithm. The fuzzy extractor includes an algorithm for generating the biometric key, typically denoted as $Gen(.)$, which takes as input the biometric data BIO_i of the sender or receiver, derived from a predefined metric space M. After processing, the algorithm produces a key string σ_i of length L bits, and an auxiliary parameter τ_i for key reconstruction. This process can be described by $Gen(BIO_i) \rightarrow (\sigma_i, \tau_i)$. Parameters σ_i and τ_i are securely stored in the memory cell of the sender or receiver device.

Key reconstruction algorithm. The reconstruction key algorithm $Rep(.)$ is responsible for recovering the original biological key σ_i from the noisy biometric data BIO_i^*. The algorithm requires the user to provide the noisy biological information BIO_i^*, together with the previously generated auxiliary parameters τ_i and a preset error tolerance threshold t. If the Hamming distance between the input biological information and the original information is not more than t, the algorithm will successfully output the original key τ_i, denoted as $Rep(BIO_i^*, \tau_i) \rightarrow \sigma_i$.

The fault-tolerant mechanism of the fuzzy extractor enables users to successfully complete the verification process even in the presence of bias in the biometric data. This mechanism is realized by defining a fault tolerance threshold t, ensuring that

key reconstruction is successful only when the deviation falls within an acceptable range.

3.4 CDH Hard Problem

In cryptography, the Computational Diffie-Hellman (CDH) problem is defined as follows: Let G be a finite cyclic group with prime order p, determined by the security parameter k. For $a, b \in Z_p$, there exists an algorithm A that takes as input a triple (g, g^a, g^b) with the goal of computing g^{ab}. If there exists an algorithm A such that, for a randomly chosen g, the algorithm correctly computes g^{ab} with probability at least ε, where ε is relative to the random choice of g, then the algorithm A is said to have an advantage of ε in solving the CDH problem in G, denoted as $\Pr[A(g, g^a, g^b) = g^{ab}] \geq \varepsilon$.

4 SCHEME FRAME

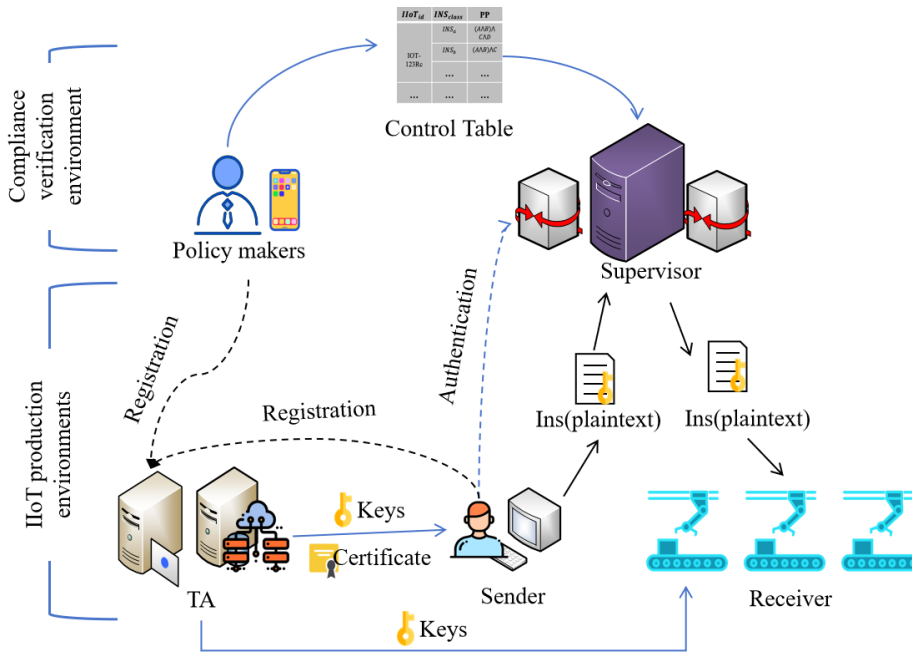


Figure 1. Framework of PAAC-IIoT scheme

Figure 1 shows the framework of the scheme, in which the participants are as follows:

Sender: Responsible for sending control instructions, generating relevant *proof*, and completing the manipulation of IIoT devices.

Supervisor: The Supervisor comprises edge devices responsible for verifying the compliance of instructions and senders, and transmitting the verified instructions to the IIoT devices.

Policy maker (PM): It is responsible for formulating the compliance policy in the control table to regulate the sender's operation permissions.

Trusted Authority (TA): TA is responsible for the registration of users and the distribution of keys or certificates to other participants.

Receiver: The receiver consists of IIoT devices with limited computing power and is responsible for executing instructions sent by the Supervisor.

5 THE PROPOSED PAAC-IIOT SCHEME

5.1 Overview

Symbol	Description
PK_{TA}	public key of TA
M_{ins}	operation instruction
SK_{TA}	private key of TA
D_i, C_i	assist validation code
ATT_i	sender's attributes set
m_{re}	ciphertext of operation instruction
Cre_U	sender's credential
PW_i	high entropy password
PK_S	public key of Supervisor
σ	sender's signature
SK_S	private key of Supervisor
TS_{max}	maximum time difference
ID_{se}	identity of the sender
TS_{now}	current timestamp
BIO_i	biometric information
ID_S	identity of the Supervisor
CT_i, DT_i, ET_i	localization code
ID_{re}	identity of the receiver

Table 1. Definitions of symbols used in scheme

In this section, we provide a comprehensive review of the PAAC-IIoT scheme. Table 1 presents the symbols used in the scheme and their meanings. The proposed PAAC-IIoT scheme consists of the seven phases: Setup, Key and Credential generation, User registration, Formulate compliance policies, Instruction sending,

Compliance verification and Execute instruction. The sequence of the seven phases is depicted in Figure 2.

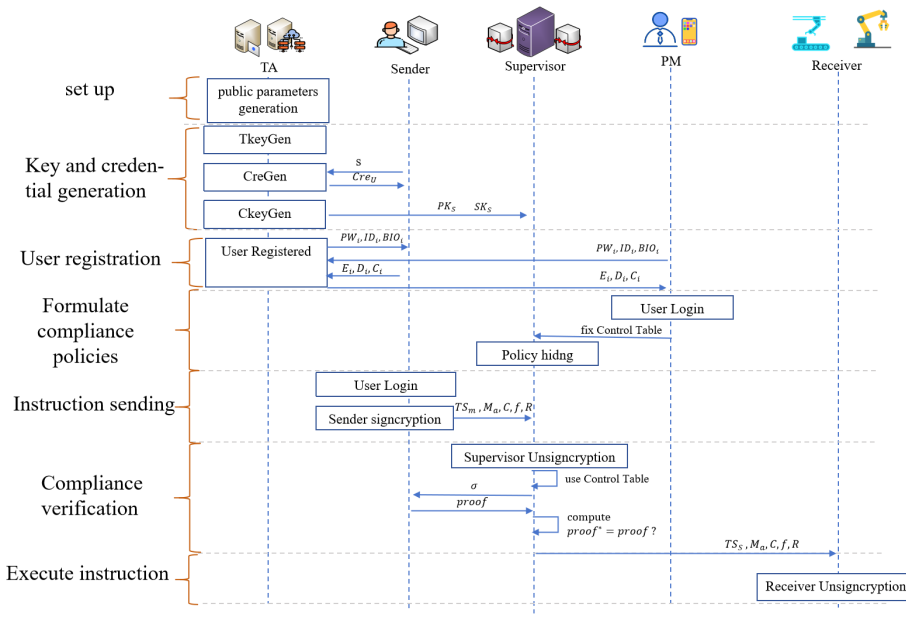


Figure 2. Work flow of PAAC-IIoT scheme

5.2 Setup

The objective of this phase is to establish public parameters, followed by Algorithm 1.

Algorithm 1 takes the security parameters λ and k as input to derive the system parameters $params_1$, $params_2$ and the master key msk. Here, $\hat{e} : G \times G \rightarrow G_T$ represents a map of three-prime compound order bilinear groups, while G_p, G_q and G_r denote three subgroups of G. Additionally, G_1 represents an additive cyclic group with a generator P, and G_2 represents a multiplicative cyclic group.

5.3 Key and Credential Generation

This phase is aim to generation keys and certificates. It comprises three algorithms: TkeyGen, CreGen, and CKeyGen, followed by these steps:

- First, TA executes Algorithm 2 with $params_1$ as input to obtain both the TA's private key SK_{TA} and public key PK_{TA}.

Algorithm 1 Public parameters generation

Input: λ, k

Output: $params_1$, $params_2$, msk

TA randomly selects $p \approx poly(1^{\lambda})$
Let $\Phi = (N = pqr, G, G_T, \hat{e})$
Randomly select $g_p \in G_p$, $g_q \in G_q$, $g_r \in G_r$
Select two hash functions: H_1:$\{0,1\}^* \to Z_N$ and $H_2 : \{0,1\}^* \to G_p$.
Randomly select $h_1, \ldots, h_n \in G_p$,
Randomly select $a, t_{1,1}, \ldots, t_{1,m_1}, \ldots, t_{m,m_n} \in Z_N$
Compute $ATT_1 = \{h_1 g_q^{t_{1,1}}, \ldots, h_1 g_q^{t_{1,m_1}}\}, \ldots, ATT_n = \{h_n g_q^{t_{n,1}}, \ldots, h_n g_q^{t_{n,m_n}}\}$
Let $params_1 = \{\Phi, \{ATT_i\}, g_p, g_p^a, H_1, H_2\}$
TA randomly selects $q \approx poly(1^k)$
Select $e : G_1 \times G_1 \to G_2$ and three hash functions: $H_a : \{0,1\}^* \to G_1$, $H_b : \{0,1\}^* \to \{0,1\}^l$ and $H_a : \{0,1\}^* \to Z_q^*$
Select $x \in Z_q^*$, $msk = x$
Compute $PK_{pub} = xP$
Let $params_2 = \{G_1, G_2, H_a, H_b, H_c, PK_{pub}\}$
return $params_1$, $params_2$, msk

Algorithm 2 TkeyGen

Input: $params_1$

Output: SK_{TA}, PK_{TA}

TA randomly selects $\alpha, r \in Z_N/0$
Let $SK_{TA} = (\alpha, r)$
Compute $U = g_p^{\alpha}$, $W = g_p^r$
Let $PK_{TA} = (U, W)$
return SK_{TA}, PK_{TA}

- Second, TA executes Algorithm 3 with $params_1$, the attribute set S of the sender, and PK_{TA} as input to obtain the sender's certificate Cre_U. Here, $U(i)$ represents the i^{th} attribute of the senders in S.

Algorithm 3 CreGen

Input: $params_1$, S, PK_{TA}

Output: Cre_U

TA randomly selects $t \in Z_N/0$
Compute $K = U g_p^{at}, L = g_p^t$
Compute $U = g_p^{\alpha}$, $W = g_p^r$
Randomly select $R_i \in G_r$
$AK_{U(i)} = h_{U(i)}^t R_i, U(i) \in S$
$Cre_U = \{K, L, \{AK_{U(i)}\}\}$
return Cre_U

- Third, TA runs Algorithm 4 and inputs $params_1$ and PK_{TA} to obtain the public key and the private key of Supervisor.

Algorithm 4 CkeyGen

Input: $params_1$, PK_{TA}
Output: SK_S, PK_S
1: TA randomly selects $a, b \in Z_N/0$
2: Let $SK_S = (a, b)$
3: Compute $X_1 = g_p^{ab}$, $X_2 = W^{ab}$
4: Let $PK_S = (X_1, X_2)$
5: **return** SK_S, PK_S

5.4 User Registration

The purpose of this phase is to facilitate user registration, and upon registration, to enable the user to authenticate locally upon login.

The sender or PM runs Algorithm 5, inputs high entropy password PW_i and identity ID_i, biometric information BIO_i, and obtains localization code: CT_i, DT_i, ET_i, $Gen(.)$, $h_1(.)$, $Rep(.)$, τ_i, t, and stores them in the users' smart device.

Algorithm 5 User registered

Input: PW_i, ID_i, BIO_i
Output: CT_i, DT_i, ET_i
1: User use Fingerprint sensor get BIO_i
2: Compute $\sigma_{log} = Gen(BIO_i, \tau_i)$
3: Randomly select $\acute{x} \in Z_p$
4: Compute $SPW_i = h_1(PW_i||\sigma_{log}||\acute{x})$ and send (ID_i, SPW_i) to TA
5: TA randomly select $\hat{q} \in Z_p$ and generate $key_{log} \in \{0,1\}^*$
6: Compute $Q_i = SPW_i \bigoplus key_{log} \bigoplus ID_i$
7: Send Q_i to Sender/Receiver
8: Sender/Receiver computes $\hat{Q}_i = Q_i \bigoplus SPW_i \bigoplus ID_i$, $CT_i = h_1(ID_i||\sigma_{log}) \bigoplus \acute{x}$
9: Compute $DT_i = h_1(ID_i||\sigma_{log}) \bigoplus \hat{Q}_i$
10: Compute $ET_i = h_1((ID_i||SPW_i||CT_i||Q_i)$
11: **return** CT_i, DT_i, ET_i

5.5 Formulate Compliance Policies

The purpose of this phase is to establish the compliance policy in the control table, enabling the computing agent to verify compliance of both the instruction and the sender.

After executing Algorithm 6 and successfully logging in, the IIoT device owner completes the Compliance Policy (CP) in the control table using Boolean-type representation and uploads it to Supervisor. To safeguard the CP, Supervisor carries out the following actions:

Using the CP as input, the shared generator matrix (M, P) is obtained using the LSSS algorithm [11], where matrix M represents the shared generator matrix of CP. The function p corresponds each row's attribute value in the shared matrix M to the PP. LSSS is derived to represent attribute values in a CP.

Algorithm 6 User login

Input: PW_i, ID_i, BIO_i
Output: *state*

User use Fingerprint sensor get BIO_i
Compute $\sigma_{log}^* = Rep(BIO_i, \tau_i)$
Compute $\acute{x}^* = h_1(ID_i||\sigma_{log}^*) \bigoplus C_i$
Compute $\hat{Q}_i{}^* = h_1(ID_i||\sigma_{log}^*) \bigoplus D_i$
Compute $SPW^* = h_1(PW_i||\sigma_{log}^*||\acute{x}^*)$
Compute $E_i^* = h_1(ID_i||SPW^*||C_i||\hat{Q}_i{}^*)$
if $E_i^* == E_i$ **then**
    $state = 1$
else
    $state = 0$
end if
return *state*

To conceal CP, the Supervisor executes Algorithm 7 and inputs the sharing matrix (M, P) to obtain the GBF. Here, M_i represents row i in the M matrix, and λ_i represents specific secret values corresponding to attribute values in the CP. The Supervisor chooses a suitable Garbled Bloom Filter $(m, n, k, H, l) - GBF$ based on the count of attribute values in the CP. m represents the size of the GBF, n denotes the count of elements inserted into the GBF (i.e., the count of attribute values in CP), k stands for the number of hash functions, and H denotes K independent collision-proof hash function groups $(H_j : Z_p \rightarrow [1, m])$, while l represents the length of the element bits inserted into the GBF. The Supervisor inserts the λ_i corresponding to attribute values into the GBF to contribute to the policy hiding operation. The whole process is shown in Figure 3.

5.6 Instruction Sending

After successfully executing Algorithm 7, the sender proceeds to Algorithm 8. It inputs the identity of the sender ID_{se}, the identity of the Supervisor ID_S, the identity of the receiver ID_{re}, $params_2$, and the operation instruction M_{ins} to obtain m_{re}=< TS_m,M_a,C,f,R >, which is then sent to the Supervisor.

Algorithm 7 Policy hiding

Input: (M, P)

Output: *GBF*

Receiver randomly selects $.v{=}(s, y_1, y_2, \ldots, y_l) \in Z_p$
Compute $C_s = g_p^s$
for $i \in [1, v.\text{length}]$ **do**
  $\lambda_i = v \times M_i$
  Convert λ_i to string λ_i^l
end for
Get $\{\lambda_i^l\}$ and choose a $GBF - (m, n, k, H, l)$
for $i \in [1, \{\lambda_i^l\}.\text{length}]$ **do**
  Randomly select $r_1, r_2, r_3, \ldots, r_{k-1} \in \{0, 1\}^l$
  Compute $r_k = r_1 \bigoplus r_2 \bigoplus r_3 \bigoplus \cdots \bigoplus \lambda_i^l$
  for j $\in [1, k]$ **do**
    $pos_j = H_j(p(i))$
    $GBF[pos_j] = r_i$
  end for
end for
return *GBF*

Algorithm 8 Sender signcryption

Input: ID_{se}, ID_{re}, ID_S, M_{ins}, $params_2$

Output: TS_m, M_a, C, f, R

TA computes $SK_{se} = xH_a(ID_{se})$ and sends it to the sender through the secure channel
Compute $PK_{se} = H_a(ID_{se})$, $PK_{S_1}{=}H_a(ID_S)$, $PK_{re} = H_a(ID_{re})$
Randomly select $m, n \in Z_q^*$
Compute $M_a = mP$, $C = H(M_a, n) \bigoplus M_{ins}$
Compute $V_1 = e(PK_{S_1}, PK_{pub})^m$, $V_2 = e(PK_{re}, PK_{pub})^m$
Compute $v_1 = H_c(M_a, V_1, ID_S)$, $v_2 = H_c(M_a, V_2, ID_{re})$
Compute $f(x) = (x - v_1)(x - v_2) + n$, $h = H_c(M_{ins}, M_a, n, ID_{re})$
Compute $R = hSK_{se} + mPK_{pub}$
return TS_m, M_a, C, f, R

5.7 Compliance Verification

The purpose of this phase is to complete the access control through the compliance verification of instructions and the sender. The process is as follows.

5.7.1 Supervisor Unsigncryption

The Supervisor receives the m_{re} sent by the sender. TA calculates $SK_S = xH_a(ID_S)$ and sends it back to the Supervisor. Subsequently, the Supervisor executes Algo-

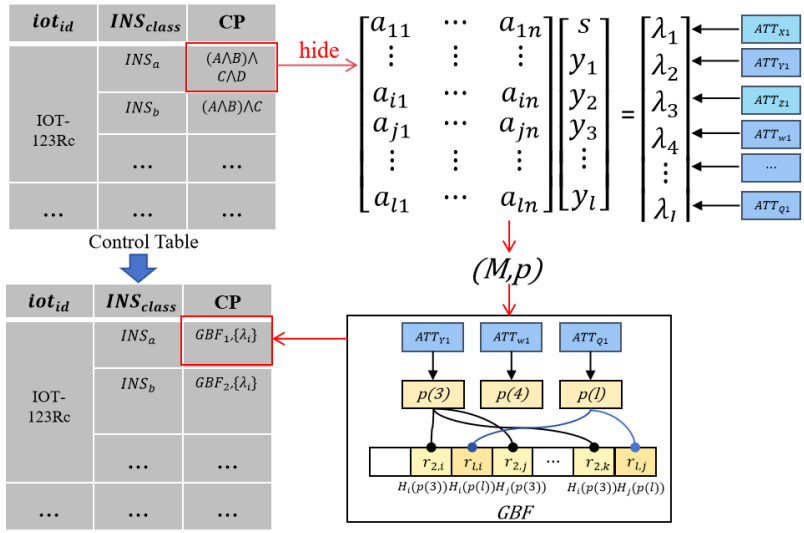


Figure 3. Process of policy hiding

rithm 9; if Supervisor gets M_{ins}, Supervisor will search it in control table and get its CP (GBF type).

Algorithm 9 Supervisor unsigncryption

Input: ID_S, ID_{re}, TS_m, M_a, C,f, R,$params_2$

Output: TS_S, M_a, C, f, R

1: **if** $|TS_m - TS_{now}| \leq TS_{max}$ **then**
2: Compute $V_1 = e(M_a, SK_S)$
3: Compute $v_1 = H_c(M_a, V_1, ID_S)$
4: Compute $n = f(v_1)$, $M_{ins} = C \bigoplus H(M_a, n)$
5: Compute $h = H_c(M_{ins}, M_a, n, ID_{re})$
6: Judge $e(R, P) = e(hPK_{se} + M_a, PK_{pub})$. If it is not true, report an error and exit
7: **end if**
8: **return** M_{ins}

5.7.2 Supervisor Processes the Instruction

The Supervisor employs a policy-controlled signature [12] on the received instruction M_{ins}. The Supervisor randomly chooses $r, f \in Z_p$, and computes in part signatures

$\sigma_1 = g_p^r$, $\sigma_2 = X_1^r$. Then calculate:

$$\triangle = \sigma_1 || \sigma_2 || PK_{TA} || PK_S || f || M_{ins},$$

$$R = f \bigoplus H_1(e(C_s, U) || H_1(\triangle)),$$

$$\varphi = \triangle || R || GBF.$$

Then, the Supervisor calculates: $\sigma_3 = H_2(\varphi)^h$, $h = ab$.

Output signature is: $\sigma = \{H_2(\triangle), \sigma_1, \sigma_2, GBF, R\}$. Finally, Supervisor transmits σ to the sender.

5.7.3 Sender Generates the Proof

The sender generates the assist-validation code for compliance verification using BFs. Only the sender's instruction complying with the CP can successfully pass the compliance verification.

When the GBF matches the BF, both the BF constructed by the sender and the GBF constructed by Supervisor share identical parameters. By leveraging the index positions on the BF, the hidden secret share $\{\lambda_i\}$ within the GBF can be deciphered. The detailed process is as follows:

1. The sender transforms its own attribute values to BFs and then inserts them into the BF to figure out $pos_i = H_i(p(j)), i \in [1, k]$ when the attribute value is $p(j)$. Then let $BF[pos_i] = 1$. The position pos_i of $BF[pos_i] = 1$ in the BF and the position of partial secret share of $\{\lambda_i\}$ in the GBF are the same.Namely, existence of $BF[pos_i] = 1 \rightarrow GBF[pos_i] = 1$.
2. Each attribute value that complies with the CP can be recovered by the BF with the matching $\{\lambda_i\}$, as follows:

$$\lambda_i = r_1 \bigoplus r_2 \bigoplus r_3 \bigoplus \cdots \bigoplus r_k.$$

3. The sender randomly chooses $r_i \in Z_N$, $\{Y_i \in G_{p_2}\}_{i \in \{1,\ldots,l\}}$, then computes assist-validation code:

$$C_i = g_p^{a\lambda_i} ATT_{i,j}^{-r_i} Y_i, \quad D_i = g_p^r Y_i.$$

The sender sets a separate BF for each attribute value, and computes the corresponding assist-validation code for each BF. The sender checks whether the equation $e(g_p, \sigma_2) = e(X_1, \sigma_1)$ holds or not. If not, the sender outputs "rejection". Otherwise, there exist constants $\{\omega_i \in Z_N\}_{i \in I}$ satisfying $\sum_{i \in I} \omega_i \lambda_i = s$ in time polynomial in the size of the share-generating matrix M. The sender uses $\{C_i, D_i\}$ to calculate

$$\theta = e(C_s, K) / \left(\prod_{i \in I} e(C_i, L) e(D_i, AK_{U(i)})^{\omega_i} \right).$$

After that, computes:

$$f_1 = R \bigoplus H_1(\theta || H_1(\triangle)),$$

$$\varphi_1 = \sigma_1 || \sigma_2 || PK_{TA} || PK_S || f_1 || M_{ins} || R || GBF.$$

Sender computes $proof = (e(H_2(\varphi_1), X_1), H_3(\varphi_1))$, and transmits it to Supervisor. The Supervisor computes $proof^* = (e(\sigma_3, g_p), H_3(\varphi_1))$, verifies $proof^* = proof$. Upon successful validation, sends the $m_{ca} = \langle TS_S, M_a, C, f, R \rangle$ to Receiver.

5.8 Execute Instruction

Upon receiving the message m_{ca} sent by the Supervisor at time, TA calculates $SK_{D_i} = xH_a(ID_{D_i})$ and sends it back to the receiver through the secure channel. The receiver receives the instruction M_{ins} after successfully executing Algorithm 10. The receiver subsequently executes M_{ins} to complete the operations it contains.

Algorithm 10 Receiver unsigncryption

Input: ID_{re}, TS_S, M_a, C, f, R, $params_2$
Output: M_{ins}

1: **if** $|TS_S - TS_{now}| \leq TS_{max}$ **then**
2: Compute $V_2 = e(M_a, SK_{re})$
3: Compute $v_2 = H_c(M_a, V_2, ID_{re})$
4: Compute $n = f(v_2)$, $M_{ins} = C \bigoplus H(M_a, n)$
5: Compute $h = H_c(M_{ins}, M_a, n, ID_{re})$
6: Judge $e(R, P) = e(hPK_{se} + M_a, PK_{pub})$. If it is not true, report an error and exit
7: **end if**
8: **return** M_{ins}

6 SECURITY ANALYSIS

6.1 Correctness

In this section, we analyze the correctness of the proposed scheme. We need to check that the policy-control signature in the scheme functions correctly under the condition that the attribute set S satisfies the requirements of the permission policy. The sender should be able to transform its attribute S set into the corresponding BF set and then use the BF to generate the auxiliary verification code. And then we use the assist-validation code to reconstruct $\{\omega_i\}_{i \in I}$ according to the attributes of the LSSS matrix $\sum_{i \in I} \lambda_i \omega_i = s$, which can recover s. The detailed derivation

process is as follows:

$$\begin{aligned}\theta &= \frac{e(C_s, K)}{\left(\prod_{i\in I} e(C_i, L)e(D_i, AK_{U_{(i)}})^{\omega_i}\right)} \\ &= \frac{e(g_p^s, g_p^{\alpha} g_p^{at})}{\left(\prod_{i\in I} e(C_i, L)e(D_i, AK_{U_{(i)}})^{\omega_i}\right)} \\ &= \frac{e(g_p, g_p)^{s\alpha} e(g_p, g_p)^{sat}}{\left(\prod_{i\in I} e(g_p^{a\lambda_i} ATT_{i,j}^{-r_i}, g_p^t)e(g_p^{r_i}, h_{U_{(i)}^t})^{\omega_i}\right)} \\ &= \frac{e(g_p, g_p)^{s\alpha} e(g_p, g_p)^{sat}}{e(g_p, g_p)^{at\sum_{i\in I}\lambda_i\omega_i}} \\ &= e(C_s, U).\end{aligned}$$

Upon calculating $e(C_s, U)$, the sender can generate a *proof* to pass the compliance verification

The correctness of instruction signcryption during transmission is as follows:

$$\begin{aligned}e(R, P) &= e(hxH_a(ID_{se}) + mxP, P) \\ &= e(x(hH_a(ID_{se}) + mP), P) \\ &= e(hH_a(ID_{se}) + mP, xP) \\ &= e(hPK_{se} + M_a, PK_{pub}).\end{aligned}$$

6.2 Unforgeability

Theorem 1. The scheme of compliant verification is said to be unforgeable under a selectable message attack if it satisfies the CDH problem assumption under a random oracle.

Proof. Assume there exists an attacker A who can execute a game in which unforgeability exists and win by a non-neglectable margin. That is, the CDH problem has been solved. The proof process consists of a sequence of games. Input (G, p, g, g^a, g^b) as an instance of the CDH problem, which satisfies $a, b \in Z_p, g \in G$. □

6.2.1 Setup

The challenger F executes the algorithms 1, 2, and 3. F randomly chooses $k \in Z_N/0$, and sets $X_1 = g_p^{ka}$, $X_2 = W^{ka}$ as the public key of signer. F generates them as public parameters to the attacker A.

6.2.2 Inquiry Stage

A performs q_{H_1} hash H_1 random oracle query, q_{H_2} hash H_2 random oracle query, q_{CO} certification oracle query and q_{SO} signing oracle query, respectively.

1. Hash random oracle query:

   - H_1: Challenger F maintains the list $List_{H_1}$ of oracle H_1. Assuming that A can make up to q_{H_1} times hash H_1 random oracle query at most. For a hash with parameter ϖ, F randomly selects $l_1 \in Z_N$ and computes $H_1(\varphi) = l_1$.
   - H_2: Challenger F maintains the list $List_{H_2}$ of oracle H_2. Assuming that A can make up to q_{H_2} times hash H_2 random oracle query at most. F receives the result of the query sent by A for message m_i, checks whether it exists in the $List_{H_2}$. Returns the result in the list if it exists; otherwise, randomly selects an integer $\gamma \in [1, q_{H_2}]$ and does the following:

   When $i \neq \gamma$, F randomly chooses $l_2 \in Z_N$, computes $H_2(m_2) = g_p^{l_2}$ and returns it to A. When $i = \gamma$, F computes $H_2(m_i) = g_p^b$ and returns it to A.
2. Certification oracle query: Run the *CreGen* to produce certificates $Cre_{U(i)}$ for the set of attribute values and return $Cre_{U(i)}$ to A.
3. Signing oracle query: A performs a signing oracle query, inputs CP^* and m^*, then runs the *sign* algorithm. By inputting CP^* and thus obtaining GBF^*, A randomly chooses $r, f \in Z_p$, and computes part of signature $\sigma_1 = g_p^r$, $\sigma_2 = X_1^r$, then calculates:

$$\triangle = \sigma_1 || \sigma_2 || PK_{TA} || PK_S || f || M_{ins} || GBF^*,$$

$$R = f \bigoplus H_1(e(C_s, U) || H_1(\triangle)),$$

$$\varphi = \triangle || R || GBF^*.$$

   F maintains the list $List_{sign}$ and then checks the equation: $H_2(\varphi) = g_p^b$.

   If the equations are not equal, bring l_2 in signing oracle query that makes $\sigma_3 = X_1^{l_2}$. Otherwise the output fails and the sign algorithm returns the signature σ.

6.2.3 Forging Stage

At the end of the inquiry stage, A generates a forged signature σ^* on the new message m^* and the compliance policy CP^* and if the signature is not queried in $List_{sign}$ and is valid, A is said to have succeeded in forging the signature and wins the game. Let $Succ_{EU-CMA} = \varepsilon$ be the probability that A wins the game, and A first makes a hash random oracle query, so we require that $q_s \leq q_{H_2}$, Only then $\sigma_3^* = H_2(m^*)^{ks} = (g^b)^{ks}$, A will have a probability of signing successfully on the message m^*, and wins the above game.

In conclusion, the probability that A wins the game and forges a signature $\sigma_3^* = H_2(m^*)^{ks} = (g^b)^{ks}$ on message m^* is less than $\varepsilon((1 - \frac{1}{q_{H_2}})^{q_{H_2}-1})^2$. That is, if and only if $ks = a$.F returns $\sigma_3^* = g_p^{ab}$ as the output of a CDH problem with non-negligible probability.

7 PERFORMANCE ANALYSIS

Our experimental environment consists of a Dell computer with a 12th-generation Intel i7-1700 processor (2.10 GHz) and 16 GB of memory, and a Raspberry Pi 4 Model B with a 1.5 GHz quad-core Cortex-A72 CPU and 4 GB of memory. The performance of the scheme is tested using the JPBC library with typeA1 and typeA curves.

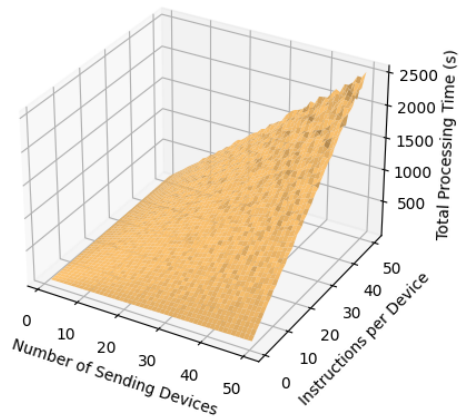


Figure 4. Time cost of processing instructions by resource-constrained device

Figure 4 and Figure 5 show the changes in time cost for compliance verification under varying instruction numbers. The experimental environment of Figure 4 is a Raspberry Pi 4 Model B, used to test the time cost on a resource-constrained device. Processing 2 500 instructions took 2 478 seconds. Figure 5 shows the time cost on a Dell computer with strong computing resources. Processing 2 500 instructions took 117 seconds. By using devices with strong computing resources, production efficiency is improved. However, resource-constrained devices still provide acceptable time costs, which helps to reduce overall production expenses.

Figure 6 shows the time cost of instruction transmission compared to related schemes. We test sender signcryption and Supervisor unsigncryption on a Dell computer. The receiver unsigncryption is tested on a Raspberry Pi 4 Model B to simulate an IIoT environment. In the PAAC-IIoT scheme, the running time of sender signcryption is 97 ms, GN unsigncryption is 419 ms, and receiver unsigncryption is

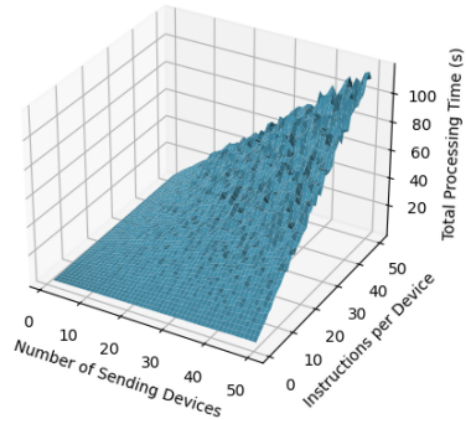


Figure 5. Time cost of processing instructions by strong computing devices

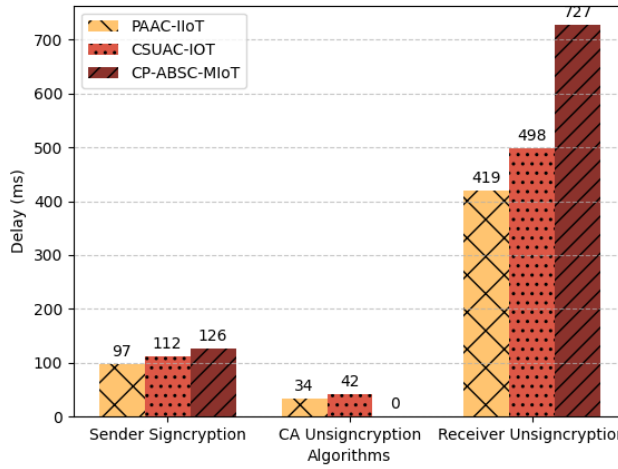


Figure 6. Comparison of time cost in instruction transmission

34 ms. The overhead of the proposed algorithms is lower than that of CSUAC-IOT [8] and CP-ABSC-MIIoT [13], and all are within a reasonable time range.

From Figure 7, we take the interval of 10 as the number of attributes in CP to test the computational overhead of generating *proof* when the sender has different numbers of attributes in CP. The test results show that the computational overhead of generating *proof* also increases with the increase of the number of attributes. When there are 100 attributes in the CP, the computation delay at the sender is approximately 2 400 ms, which is considered reasonable.

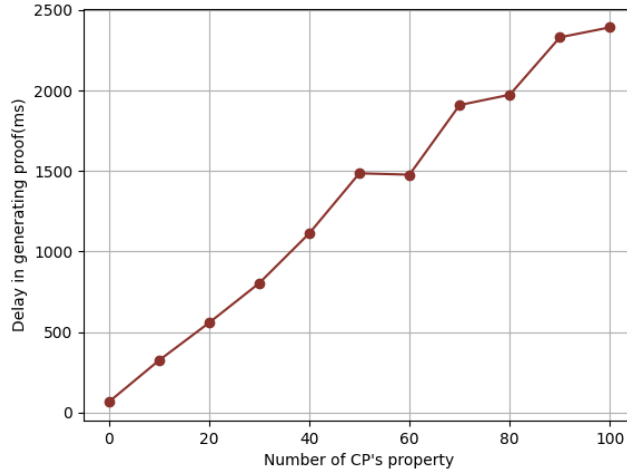


Figure 7. Time cost in proof generation

8 CONCLUSION

To mitigate security threats like non-compliant operations and malicious message tampering in IIoT device remote operation, this paper proposes an access control scheme named PAAC-IIoT. By hiding the compliance policy, performing compliance verification and using multi-verifier signcryption, PAAC-IIoT secures instruction transmission and protects sender privacy. Security analysis and experiments demonstrate that PAAC-IIoT effectively prevents forgery and tampering while maintaining an appropriate computational cost, thereby ensuring the security of IIoT remote operations. In future work, we aim to introduce blockchain to prevent tampering of instruction logs.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Nos. 62062016, U21A20474 and 62103114), Jiangsu Provincial Key Laboratory of Network and Information Security (No. BM2003201), and Guangxi Science and Technology project (Nos. GuikeAA22067070 and GuikeAD21220114). Finally, we thank the Guangxi "Bagui Scholar" Teams for Innovation and Research Project, Center for Applied Mathematics of Guangxi (Guangxi Normal University), and the Guangxi Talent Highland Project of Big Data Intelligence and Application.

REFERENCES

[1] Huo, R.—Zeng, S.—Wang, Z.—Shang, J.—Chen, W.—Huang, T.—Wang, S.—Yu, F. R.—Liu, Y.: A Comprehensive Survey on Blockchain in Industrial Internet of Things: Motivations, Research Progresses, and Future Challenges. IEEE Communications Surveys & Tutorials, Vol. 24, 2022, No. 1, pp. 88–122, doi: 10.1109/COMST.2022.3141490.

[2] Li, T.—Wang, H.—He, D.—Yu, J.: Permissioned Blockchain-Based Anonymous and Traceable Aggregate Signature Scheme for Industrial Internet of Things. IEEE Internet of Things Journal, Vol. 8, 2020, No. 10, pp. 8387–8398, doi: 10.1109/JIOT.2020.3045451.

[3] Chen, J.—Bao, Z.—Luo, M.—He, D.: A Security Remote Command Control Scheme for Industrial Internet of Things. Computer Engineering, Vol. 50, 2024, No. 3, pp. 28–35, doi: 10.19678/j.issn.1000-3428.0067299.

[4] Wang, J.—Zhu, M.—Li, M.—Sun, Y.—Tian, Z.: An Access Control Method Against Unauthorized and Noncompliant Behaviors of Real-Time Data in Industrial IoT. IEEE Internet of Things Journal, Vol. 11, 2023, No. 1, pp. 708–727, doi: 10.1109/JIOT.2023.3285992.

[5] Liu, S.—Chen, L.—Yu, H.—Gao, S.—Fang, H.: BP-AKAA: Blockchain-Enforced Privacy-Preserving Authentication and Key Agreement and Access Control for IIoT. Journal of Information Security and Applications, Vol. 73, 2023, Art. No. 103443, doi: 10.1016/j.jisa.2023.103443.

[6] Usman, M.—Sarfraz, M. S.—Habib, U.—Aftab, M. U.—Javed, S.: Automatic Hybrid Access Control in SCADA-Enabled IIoT Networks Using Machine Learning. Sensors, Vol. 23, 2023, No. 8, Art. No. 3931, doi: 10.3390/s23083931.

[7] Zhang, Z.—Wu, X.—Wei, S.: Cross-Domain Access Control Model in Industrial IoT Environment. Applied Sciences, Vol. 13, 2023, No. 8, Art. No. 5042, doi: 10.3390/app13085042.

[8] Mandal, S.—Bera, B.—Sutrala, A. K.—Das, A. K.—Choo, K. K. R.—Park, Y.: Certificateless-Signcryption-Based Three-Factor User Access Control Scheme for IoT Environment. IEEE Internet of Things Journal, Vol. 7, 2020, No. 4, pp. 3184–3197, doi: 10.1109/JIOT.2020.2966242.

[9] Boneh, D.—Goh, E. J.—Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. In: Kilian, J. (Ed.): Theory of Cryptography (TCC 2005). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 3378, 2005, pp. 325–341, doi: 10.1007/978-3-540-30576-7_18.

[10] Mitzenmacher, M.—Upfal, E.: Probability and Computing: Randomized Algorithms and Probabilistic Analysis. Cambridge University Press, 2005.

[11] Lewko, A.—Waters, B.: Decentralizing Attribute-Based Encryption. In: Paterson, K. G. (Ed.): Advances in Cryptology – EUROCRYPT 2011. Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6632, 2011, pp. 568–588, doi: 10.1007/978-3-642-20465-4_31.

[12] Zheng, X.—Zheng, F.—Liu, X.—Wang, D.—Wang, J.—Meng, B.: A Secure and Policy-Controlled Signature Scheme with Strong Expressiveness and Privacy-

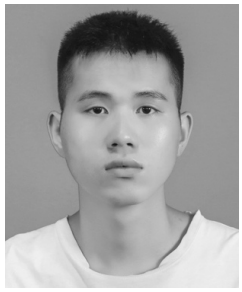
Preserving Policy. IEEE Access, Vol. 9, 2021, pp. 14945–14957, doi: 10.1109/ACCESS.2021.3052463.

[13] Patil, R. Y.: A Secure Privacy Preserving and Access Control Scheme for Medical Internet of Things (MIoT) Using Attribute-Based Signcryption. International Journal of Information Technology, Vol. 16, 2024, No. 1, pp. 181–191, doi: 10.1007/s41870-023-01569-0.

Linghui Meng is currently pursuing his Master's degree with the School of Computer Science and Engineering, Guangxi Normal University, Guilin, China. His research interests include data distributed system, data security, Internet of Things, and blockchain system.

Feng Yu received her Ph.D. degree from the Southeast University, China in 2015. She is currently Associate Professor at the Guangxi Normal University, China. Her main research fields are data distributed system, data security, blockchain, and cloud computing.

Daicen Jiang has his full-time Bachelor's degree and his Master's degree from the Guangxi University. He is an intermediate economist and tender-inviter, and he works in the China Southern Power Grid Supply Chain Group (Guangxi) Co., LTD. So far, he has published more than five articles, all as the first author. He has organized and implemented large-scale bidding and procurement projects for a long time, participated in frontier technology research and research for many times, and has rich experience in bidding and procurement management.



Jing Xi is currently pursuing his Master's degree with the School of Computer Science and Engineering, Guangxi Normal University, Guilin, China. His research interests include data distributed system, data security, Internet of Things, and blockchain system.

Shenglin Cao is Associate Professor at the Ningxia Normal University, he specializes in research on network information security and network management. So far, he has published multiple papers, including 6 as the first author. He has led three school level projects, taught one department level course, participated in one national project, and two provincial-level projects.

Zhihua Zeng works for the CGN Engineering Co., LTD., he is engaged in the construction, operation and maintenance of nuclear power design and production platform, information and network security. His research interests include information security system construction, automatic operation and maintenance of information systems, and the application of cloud technology and blockchain technology in enterprises.