Proxy Re-encryption Scheme based on the Timed-release in Edge Computing

Yifeng Yin[1], Wanyi Zhou[2], Zhaobo Wang[3], Yong Gan[4], Yanhua Zhang[5]
School of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou, China[1, 2, 3, 5]
Zhengzhou Institute of Technology, Zhengzhou, China[4]

***Abstract*—With the growth of Industrial Internet, various types of data show explosive growth. Data is being moved from the cloud to the edge for computing more frequently and edge computing becomes an important factor affecting the deep application of Industrial Internet platforms. However, the security issue of data transmission sharing is not addressed. Therefore, this paper proposes a security scheme based on timed-release, multi-dimensional virtual permutation, and proxy re-encryption(PRE), to protect the confidentiality of the data during the transmitter; symmetric cryptography is employed to encrypt the transmission data. At the same time, a time server is used, making it impossible for data receivers to get the information about the data before the specified time arrives, it solves the application scenario of data transmission and sharing with timed-release requirements and the efficiency of a large number of data is solved, and the security of data is improved. Finally, the security of the scheme was proved theoretically. Compared to existing PRE schemes, this scheme adds timed-release controlled access, has resistance to ciphertext attacks and end-to-end security features, and uses fewer bilinear pair operations in the algorithm. The performance was tested experimentally and the results show that the scheme improves efficiency while ensuring security and has significant advantages in terms of data security and private data protection.**

***Keywords*—Timed-release; edge-computing; multi-dimensional virtual permutation; proxy re-encryption; symmetric encryption**

I. INTRODUCTION

Information technologies like big data, cloud computing, and artificial intelligence have developed so quickly in recent years, new ways of manufacturing and structuring are driving the intellectual conversion of the global industrial system [1]. Thus, with the benefits of both the industrial and internet revolutions combined, the industrial internet arose. As a result of big data and artificial intelligence, nowadays, if there is a lack of effective data, massive industrial data will be buried, and the Industrial Internet will lose its value of existence[2], [3]. At the same time, with the accelerating pace of the construction of Industrial Internet platforms, users' demand for security is more urgent. It has gradually evolved from the security of simple industrial equipment use to the security of data stored in the equipment and data transmitted in the network, from tangible security to intangible security[4], [5]. In contrast to the established cloud computing model, edge computing model makes up for the disadvantage of cloud computing away from terminal devices, it is a fundamental technology in the Industrial Internet and helps the Industrial Internet evolve[6]. Edge computing needs to provide the new capabilities of "device openness and data sharing" needed in industrial transformation and upgrading, and reliability-wise, edge computing can satisfy the industrial Internet's development requirements[7], [8]. Data may pass via numerous message-forwarding nodes in the edge computing environment before it reaches the collaborating nodes instead of being delivered directly from one device to another[9]. The data is conveyed by a third party, and if the communicated data is maliciously altered, it will have a significant impact on data security issues. Therefore, in such a scenario, it is crucial to guarantee the security of data transmission between users.

Proxy re-encryption technology is an encryption cryptosystem with ciphertext conversion function, which does not need data decryption in this process, so it can be used to achieve data transmission security[10]. PRE technology ensures the secure access and sharing of data without revealing any relevant information about the plaintext[11], [12]. However, in some application scenarios with timed-release requirements, this technology is not perfect, and the ciphertext generated by PRE needs to have the characteristics of timed-release[13]. Time-dependent encryption includes methods such as timed-release encryption, which can be used in time-sensitive scenarios[14]. Even the selected receiver is unable to decrypt the ciphertext prior to the semi-trusted time server opening the trapdoor linked to the publisher's chosen release time[15], which can be used as a cryptographic primitive to specify the future decryption time, and there are many scenarios in real life where the decryption time needs to be specified in advance[16], [17]. Therefore, PRE technology and timed-release encryption work better together to improve the confidentiality and flexibility of the scheme.

PRE was first proposed by Blace[18], it enables a partially trusted proxy to convert the publisher's ciphertext into the receiver's ciphertext, and increases the flexibility of data sharing. Since it was proposed, numerous PRE techniques have been put forth. Peng et al. [19]presented an identity-based conditional broadcast PRE system that enables users to communicate encrypted data with other users using the fine-grained approach, allows users to generate broadcast ciphertexts for numerous recipients, and share their encrypted data to multiple recipients in batches. Kaitai et al.[20] proposed a conditional proxy broadcast re-encryption method with timed-release, which allows the delegator to delegate the decryption right to multiple delegates in a fine-grained manner without losing security, and the delegator has the ability to schedule the delegation of the decryption right of the broadcast encryption to a certain group of receivers in addition to

delegating it. Massive privacy-sensitive data transmitted to edge nodes may attract attacks from network attackers, so data leakage and attacks are a great loss and danger for the Industrial Internet.

To address the aforementioned issues, this work proposes the timed-release based PRE scheme for edge computing. Since the efficiency of public key cryptography is not as high as the symmetric encryption algorithm, it is not suitable for the encryption of large data files. Therefore, a multi-user data-sharing scheme is designed by combining symmetric cryptography, time-release encryption and PRE techniques, and depends on a multi-dimensional virtual substitution mechanism. It effectively protects this publisher's privacy information, prevents malicious tampering and theft, meets the requirements of timeliness and multi-user sharing data, ensures the security of data transmission, reduces the time cost, and improves the operation efficiency.

The rest of the paper is organised as follows. In Section II, we describe the constructed scheme model. In Section III, we describe the design steps of the scheme in detail. In Section IV, we perform security analysis and performance evaluation of the scheme. Section V is discussion. Section VI concludes the paper.

II. Model Design

The traditional PRE scheme uses an asymmetric cryptosystem, which requires a large amount of computation, large consumption and waste of resources for large data files. And in some application scenarios, it is necessary to meet the requirements of time. Thus, a PRE scheme based on timed-release for edge computing is created to ensure the security of data information and address the issue of timeliness. According to the actual application of data information sharing, a multi-user data sharing model is established as shown in Fig. 1.

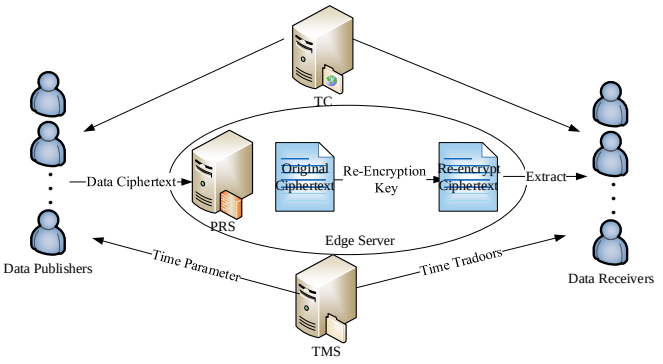

Fig. 1. Multi-user data sharing model.

Both data publishers and data receivers can access data in the network and interact with the proxy re-encryption server (PRS), trusted center (TC) and time server (TS) to create and access data. The third trusted time server generates time parameters according to the requirements, which is used to encrypt the plaintext. The generated time trapdoor algorithm can be used to confirm that the information has not been tampered with. The data publishers upload the data cipher text to the proxy server and then encrypt the original cipher text into the re-encrypted ciphertext by the re-encryption key, from which the data receivers can extract the re-encrypted cipher text without dumping and storing the data that has already been used.

III. Scheme Design

The symbols used in the scheme of this paper and their corresponding meanings are shown in Table I.

TABLE I. Symbols and Meanings in the Text

Symbol	Mention
sk_{ts}/pk_{ts}	Time server's private and public key
sk_i/pk_i	Data publisher $i's$ private/public key
sk_j/pk_j	The private/public key by data receiver j
T_G	Timed-release key
Sk_i'	Initial key array for data publisher i
$Sk_i[n]$	Key control array for data publisher i
$IVF(i)$	Virtual permutation function missing i-th argument
$MVPF$	The complete virtual permutation function
$MITE^{(m)}$	A virtual iteration function that iterates m times
C_T	Original ciphertext
C_S	Session key ciphertext
C_P	Re-encryption of ciphertext
$RK_{i\to j}$	Proxy re-encryption key for users i to j

Multi-user data sharing through timed-release proxy re-encryption algorithm can solve the time limitation problem in data sharing compared to traditional PRE schemes. This topic uses edge server as a tool to realize data sharing, the time server is introduced to create the release time parameters, and the multi-dimensional virtual replacement mechanism is combined in the symmetric key generation stage, which can improve confidentiality and efficiency. According to the multi-user data sharing model, the scheme can be realized as described below.

A. System Initialization

Choose a prime q, a finite cyclic group G and G_T of order q, a bilinear mapping $e: G \times G_T \to G_T$, where g is the generating element of G. Then define the hash functions $H_0: \{0,1\}^* \to G$, $H_1: \{0,1\} \to G$, $H_2: G_T \to \{0,1\}^L$, and randomly select an element γ from Z_q^* as the private key for the time server $sk_{ts} = \gamma$, then the public key for the time server $pk_{ts} = g^{\gamma}$ can be calculated.

B. Key Generation

Randomly select the elements x_i and x_j from Z_q^* as the private keys of data publishing user i and data receiving user j respectively, and make $sk_i = x_i$ and$sk_j = x_j$, then the public keys corresponding to data publishing user and data receiving user are $pk_i = g^{x_i}$ and $pk_j = g^{x_j}$ respectively.

C. Generate Time Trapdoor

Firstly, the time parameter $T \in \{0,1\}^*$ can be generated by the time server to display the publishing time. Then through the calculation of the private key sk_{ts} by the time server and the

time parameter T, this time trapdoor can be obtained, corresponding to the timing release key T_G, as shown in the formula.

$$T_G = H_0(T)^{\gamma} \quad (1)$$

D. Session Key Generation

This phase first requires constructing a multi-dimensional key space, then displacing the security subsystem, constructing a multi-dimensional virtual substitution function, and iterating over the security subsystem it displaces in a certain order to finally obtain a session key that conforms to the security rules. This process uses a lightweight cryptographic algorithm based on stream ciphers, and the specific steps are described as shown below.

1) Constructing a multi-dimensional key space: The private key $\mathrm{Sk_i}$ of data publishing user i and its private key array $\mathrm{Sk_i'}$ can be hashed and mapped to generate a key control array $\mathrm{Sk_i[n]}$ by the following formula. Then it is sent to other data publishing users, each data publishing user can get an incomplete virtual iterative function IVF(i) that lacks its own private key, where the user's private key array $\mathrm{Sk_i'} = [\mathrm{k_{i1}}, \mathrm{k_{i2}}, \ldots, \mathrm{k_{in}}]$, because it has no actual ability of control function, so it does not pass to other users.

$$Sk_i[n] = \{SK_i \oplus \sum_{a=1}^{n} H(Sk_i'[a])\} \quad (2)$$

Data publishing user i substitutes the key control array $Sk_i[n]$ into the incomplete virtual iterative function to obtain a multi-dimensional virtual iterative function $MVPF$ with complete key parameters, mapped into an n-dimensional spatial network with equal probability of security subsystems mapped into the n-dimensional space to establish an n-dimensional key space, where each small space maps a security subsystem as shown in the formula.

$$S = MVPF(Sk_1[n], Sk_2[n], \cdots, Sk_i[n], \cdots, Sk_n[n]) \quad (3)$$

2) Symmetric key generation: If each data publishing user randomly selects m key elements in the key control array, then the m secure subsystems can be noted as $\tau_1, \tau_2, \cdots, \tau_m$, respectively, and a secure symmetric key K can be obtained by iterative operations in a certain order according to the formula.

$$K = \tau_m(MITE^{(m-1)}) \quad (4)$$

E. Transformation Key Generation

The private key sk_i by data publishing user i, the public key pk_j by data receiving user j, through hash mapping, can get RK_1. Then the time server generates the publishing time parameter T and the public key pk_{ts} through the calculation of the following formula, RK_2 can be obtained, and finally the conversion key is obtained by the calculation of the formula, which can also be called this re-encryption key $RK_{i\to j}$, and put online via the proxy server.

$$RK_1 = sk_i \cdot (H_2(g^{\beta}, pk_j)^{\rho})$$
$$RK_2 = \varphi \oplus e(H_0(T), pk_{ts}) \quad (5)$$
$$RK_{i\to j} = (RK_1, RK_2)$$

F. Proxy Re-encryption

1) The original ciphertext generation: The data publishing user has the ability to safely encrypt the data plaintext M using the publicly available symmetric encryption algorithm based on the symmetric key generated in the previous session key phase, which can obtain the desired data ciphertext C_1. Then a random number $\beta \in Z_q^*$ and $\varphi \in G_T$ is chosen and the hash function $H_3: \{0,1\}^* \to Z_q^*$ is defined such that $\rho = H_3(M, \beta)$, and the ciphertext C_2 can be obtained after the calculation by the formula, and the ciphertext C_3 can be obtained through the time server's public key pk_{ts} and the published time parameter T. Finally, these obtained ciphertexts can be calculated by the following formula to get the required original ciphertext C_T and uploaded to the proxy server at the same time.

$$C_1 = SymEnc_K(M)$$
$$C_2 = (M||\beta) \oplus g^{H_2(g^{\beta}, \varphi)}$$
$$C_3 = \varphi \oplus e(H_0(T), pk_{ts})^{\rho} \quad (6)$$
$$C_T = H_1(C_1||C_2||C_3)$$

2) Session key ciphertext: This stage uses the encryption of the session key and then generates the key cipher, which can be used in the subsequent stage of using the proxy server for this PRE algorithm to generate this re-encrypted cipher, where the session key and the public key of the data publishing user are calculated by the following formula to obtain the ciphertext C_S of this session key and upload to the proxy server.

$$C_S = {pk_i}^{\beta} \cdot K \quad (7)$$

3) Data sharing: The PRE ciphertext C_P is obtainable by using that session key ciphertext C_S and the transformation key $RK_{i\to j}$, along with the time server's publication time parameter T. A proxy server that has some level of confidence performs this process, which does not disclose any message about the ciphertext, so the security of this scheme can be guaranteed.

$$C_P = C_S||e(H_1(T), RK_{i\to j})^{\beta} \quad (8)$$

The data recipient can get the plaintext M computing the decryption of that PRE ciphertext. Firstly, the PRE ciphertext C_P and the data receiver's private key sk_j are decrypted to obtain the session key K. Then the original ciphertext C_T is decrypted using the session key K to obtain the plaintext, as shown in the following formula.

$$K = Dec_{sk_j}(C_P)$$
$$M = Dec_K(C_T) \quad (9)$$

IV. Security Analysis and Performance Evaluation

A. Security Analysis

This scheme combines the multi-dimensional virtual permutation mechanism and timed-release encryption and uses the PRE algorithm to implement the transfer of data information, so as to solve the protection of data in the industrial Internet. The common attacks that can be dealt with and the specific analysis are as follows.

1) Confidentiality: In this program, data plaintext is converted into data ciphertext using symmetric encryption algorithm, which is stored in the proxy server, as well as using the public key to encrypt the symmetric key to generate key ciphertext. Because data information and session keys are transmitted in ciphertext throughout the communication process, a network attacker cannot decrypt its data information and restore the ciphertext to plaintext. To decrypt the ciphertext and acquire the plaintext data, only the data publisher and the data receiver are capable. If the proxy server tries to decrypt the data information, then it must get $RK_{i\to j}$ and pk_j, but the proxy server only has pk_j, as a result, it is unable to access information about the original data. This system thus achieves the security of private and sensitive data on the Industrial Internet by guaranteeing the secrecy of the data.

2) Replay attack protection: In this scheme, only within the set time validity period, the user who is receiving the data can get the plaintext data by decrypting the ciphertext that has been re-encrypted. The data-receiving user can get the corresponding reverse key only after getting the ciphertext that was re-encrypted and decrypting it, and because there is a certain time limit for the reverse key at this time, this reverse key can be saved only until this time arrives. However, when a network attacker performs a replay attack, the same reverse key can be obtained by decrypting the data message, but then it discards the message and does not deliver the data, and also cannot obtain the server's authentication, which can be used to ensure that the time trapdoor is always time-limited. Therefore, this scheme can resist the replay attack.

3) End to end: End-to-end security truly entails that the server won't be aware of the communication information shared by cooperating nodes, and the proxy server relies on the edge server, which $RK_{i\to j}$ has decided the conversion rule of decryption, so the server cannot convert the cipher text into other decryption combinations. Secondly, since the data information and session keys are always transmitted in cipher text during the communication, even the proxy server cannot decrypt them and get the data information. By decrypting the ciphertext that has been re-encrypted, only the users who publish data and those who receive it can access the plaintext data, so even if this server is attacked, this data information sent between users will not be disclosed. Therefore, the scheme satisfies end-to-end security.

4) Protection against collusion attack: In this scheme, the private key is secure against collusion attacks. The data receiver and the proxy cannot conspire to gain the private key of the data publisher, assuming that the proxy sends the re-encryption key to the data receiver for malicious collusion. Since $sk_i \cdot (H_2(g^{\beta}, pk_j)^{\rho})$ in β is random. Using the re-encryption key as a base, the data receiver cannot determine the data publisher's private key information. At the same time, the agent cannot conspire with the data publisher to get the recipient's private key, if the publisher of the data and the agent join together, because the re-encryption key $RK_{i\to j} = (RK_1, RK_2)$ does not have any private key information of the data recipient, and cannot get its private key through calculation, so the combination of the two cannot acquire the data receiver's private key. This proves that even if this proxy server and the data receiver or data publisher maliciously collude, they cannot get each other's private keys to decrypt the ciphertext without access rights, so this scheme resists the collusion attack.

5) Select ciphertext security: In arbitrary probabilistic polynomial time, the benefit of addressing the hypothetical issue of DBDH is negligible, so the scheme proposed in this study is to choose ciphertext attack security. Assuming the existence of attacker A and challenger C.

Firstly, A sends queries to C about system initialization, time trapdoor generation, key generation, session key generation, original ciphertext generation, session key ciphertext generation, transformation key generation and proxy re-encryption ciphertext generation, and then C returns the query results to A. C creates the corresponding public and private keys through this system initialization and key generation algorithm, as well as the time server's private key, and performs this re-encryption key algorithm on them to get the re-encryption key, then transmit it to A. A chooses two equal-length plaintexts M_0 and M_1 and sends them to C to challenge it. When A asks C for the original ciphertext generation algorithm, where the session key is kept secret. C randomly picks bits $d \in \{0,1\}$, computes the ciphertext $C^* = Encryption(T_G^*, pk_{ts}^*, K^*, M_d, T^*)$. A then returns a guess $d_1 \in \{0,1\}$ to C. At this point, if $d_1 = d$, the challenge is successful. Assuming that the probability superiority of A overcoming the challenge is ε and $\varepsilon = |\Pr[d_1 = d] - 1/2|$ is negligible. the probability that A guessing correctly in probabilistic polynomial time is negligible, then A fails the challenge, so it can be shown that the scheme of this paper is to choose ciphertext security.

B. Performance Evaluation

The scheme of this paper was analyzed in comparison with some relevant studies in recent years, and the following data were used to compare the time consumption with the literature, as shown in Table II.

Through the existing relevant references, the theoretical computational overheads of the schemes are compared with those of this paper, and the theoretical execution time comparison results of each scheme are shown in Table III. The time required for bilinear is much larger than that of other operations. Because the bilinear pairing used in reference [19] and [20] are more than those used in this scheme, the time used

in these two schemes will be more. According to the comparison results of the theoretical time overhead in Table II, we can get that the scheme in this chapter is smaller than reference [19] and [20] in terms of computational overhead, so our proposed scheme has higher efficiency and also higher security, which shows that this scheme has significant advantages in security and efficiency.

TABLE II. THE EXECUTION TIME OF EACH OPERATION

Parameter	Description	Value(ms)
T_p	Bilinear pairing operation time	4.211
T_m	Point multiplication operation time	0.015
T_e	Exponentiation operation time	3.886
T_h	Hash operation time	0.0001
$T_{E/D}$	Symmetric encryption/decryption operation time	0.0046

TABLE III. COMPARISON OF TIME COMPLEXITY OF EACH SCHEME

Reference	Data Processing	Proxy Re-encryption	Sum
Peng [19]	$(n+2)T_e + 2T_p + 5T_m + (n+1)T_h$	$3T_m + 2T_p + 3nT_h + (n+4)T_e$	$(2n+6)T_e + 4T_p + 8T_m + (4n+1)T_h$
Kaitai[20]	$2nT_e + T_p + (2n+5)T_h$	$(n+1)T_e + 8T_p$	$(3n+1)T_e + (2n+5)T_h + 9T_p$
Zheng[21]	$(n+3)T_p + 2nT_e$	$nT_m + nT_p$	$(2n+3)T_p + nT_m + 2nT_e$
Proposed	$(n+2)T_h + T_p$	$nT_e + T_p + 2T_{E/D} + (n+1)T_h$	$2T_{E/D} + 2T_p + nT_e + (2n+3)T_h$

In order to confirm the effectiveness of the plan put forth in this paper, an experimental environment is built for the experimental testing of the algorithm. Experimenting with Charm-Crypto 0.5, PBC 0.5.14 library in Python 3.9 under a Linux system with an Intel(R) Core(TM) i7-10700F CPU @ 2.90GHz and 16GB RAM. Simulation of the time spent comparing the literature with the solution proposed in this paper, and the simulation's outcomes are displayed in Fig. 2. After simulating the time required to implement each scheme. The results of the experiments reveal that the system suggested in this study has a much lower overall computing overhead than the research [19], [20] and [21] schemes.

For study [19], although the dot product operation is added compared to reference [20], it does not have much impact on the time overhead as it is negligible; it also reduces the bilinear pairing operation, which reduces the time overhead and improves the efficiency to some extent.

For study [20], although it uses more hashing operations compared to [21], it does not have as much time overhead as [21] because the execution time of the hashing operation is negligible. However, compared to the scheme in this paper, it still requires more time cost.

For research [21], it uses bilinear pairing operations, which require a lot of time to compute each bilinear pairing, and uses a lot of dot product operations and power operations. Although it uses less time consumption at the beginning, the time overhead increases as the number of users increases. Therefore, the most computationally expensive of these schemes.

Compared with the schemes proposed in the studies [19], [20] and [21], the time consumption of this scheme is the least. Although this paper uses one more symmetric encryption/decryption operation, it does not increase the time overhead because its execution time magnitude is much smaller than the other operations and negligible, and uses a small number of bilinear pairing operations and power operations, in a specific test of execution time overhead the lowest. This scheme uses lightweight operations, which reduces the overall time spent. Because this solution uses fewer bilinear operations, which are precisely the functions with the highest computational overhead, reducing the number of calls to bilinear operations in the algorithm has a clear advantage in terms of efficiency. The lightweight proxy re-encryption algorithm used in this scheme ensures secure data transfer and improves efficiency.

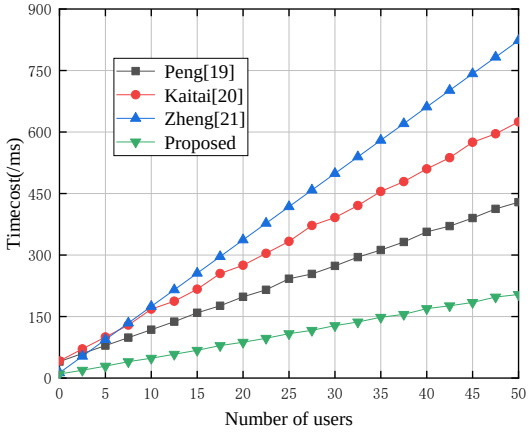

Fig. 2. Comparison of schemes' time cost.

V. DISCUSSION

The scheme in this paper introduces a third-party fully trusted time server to encrypt the plaintext by inserting a time attribute, which can only be decrypted by an authorised recipient within a set timeliness range to obtain the plaintext. Compared to existing PRE schemes, this scheme is more flexible, provides end-to-end data security and reduces the trust requirement for proxies. The security analysis shows that this scheme is resistant to ciphertext attacks and thus has higher security. The computational overhead comparison and simulation experiments show that this scheme can save a lot of transmission time and has better computational overhead performance and is more efficient.

VI. CONCLUSION

This paper analyzes the security needs for data transmission and sharing in an environment using edge computing. For the current PRE study is difficult to meet the efficiency and security for the transmission of many data files, and the requirements of time constraints, this paper proposes a PRE scheme based on timed-release, which offers efficient, flexible and secure data sharing services. The secure sharing of data information is completed by symmetric cryptographic

encryption and proxy re-encryption, and time servers are introduced to solve the problem of timeliness and protect privacy and security. On the premise of ensuring data security and confidentiality, reducing time consumption overhead and improving the efficiency of communication, this scheme is appropriate for data sharing from larger files, which can be completed in a short time, and has great advantages and practical value.

ACKNOWLEDGMENT

This paper is supported in part by the Natural Science Foundation of Henan Province of China under Grant No.202300410508, the Natural Science Foundation of Henan Province under Grant No.222300420371, the Key Research Project of Higher Education of Henan Province under Grant No. 22A520047, the National Natural Science Foundation of China under Grant U1804263, and key foundation of Science and Technology Development of Henan Province under Grant No.142102210081.

REFERENCES

[1] W. Qin, S. Chen, and M. Peng, "Recent advances in Industrial Internet: insights and challenges," Digit. Commun. Netw., vol. 6, no. 1, pp. 1–13, Feb. 2020.

[2] J.-Q. Li, F. R. Yu, G. Deng, C. Luo, Z. Ming, and Q. Yan, "Industrial Internet: A Survey on the Enabling Technologies, Applications, and Challenges," IEEE Commun. Surv. Tutor., vol. 19, no. 3, pp. 1504–1526, 2017.

[3] X. Zhang and X. Ming, "Implementation path and reference framework for Industrial Internet Platform (IIP) in product service system using industrial practice investigation method," Adv. Eng. Inform., vol. 51, p. 101481, Jan. 2022.

[4] L. Jun and L. Lan, "Research on Security Detection and Data Analysis for Industrial Internet," in 2019 Companion of the 19th Ieee International Conference on Software Quality, Reliability and Security (qrs-C 2019), Los Alamitos, 2019, pp. 466–470.

[5] L. Wang, Z. Ye, R. Zhang, J. Lin, F. Chen, and F. Tang, "The Growth Model of Industrial Internet Platform in Industrial 4.0," Wirel. Commun. Mob. Comput., vol. 2022, p. 5145641, Mar. 2022.

[6] M. Wei, X. Yang, J. Mao, and K. Kim, "Secure Framework and Security Mechanism for Edge Nodes in Industrial Internet," in Proceedings of the 13th International Conference on Ubiquitous Information Management and Communication (imcom) 2019, Cham, 2019, vol. 935, pp. 254–266.

[7] Y. Yin, Z. Wang, W. Zhou, Y. Gan, and Y. Zhang, "Group key agreement protocol for edge computing in industrial internet," Math. Biosci. Eng., vol. 19, no. 12, pp. 12730–12743, 2022.

[8] S. Zhu, K. Ota, and M. Dong, "Green AI for IIoT: Energy Efficient Intelligent Edge Computing for Industrial Internet of Things," Ieee Trans. Green Commun. Netw., vol. 6, no. 1, pp. 79–88, Mar. 2022.

[9] Q. Zhang, J. Cui, H. Zhong, and L. Liu, "Toward Data Transmission Security Based on Proxy Broadcast Re-encryption in Edge Collaboration," Acm Trans. Sens. Netw., vol. 18, no. 3, p. 48, Aug. 2022.

[10] H. Guo, Z. Zhang, J. Xu, N. An, and X. Lan, "Accountable Proxy Re-Encryption for Secure Data Sharing," Ieee Trans. Dependable Secure Comput., vol. 18, no. 1, pp. 145–159, Jan. 2021.

[11] J. Li, J. Peng, and Z. Qiao, "A Ring Learning with Errors-Based Ciphertext-Policy Attribute-Based Proxy Re-Encryption Scheme for Secure Big Data Sharing in Cloud Environment," Big Data.

[12] C. Ge, W. Susilo, J. Baek, Z. Liu, J. Xia, and L. Fang, "A Verifiable and Fair Attribute-based Proxy Re-encryption Scheme for Data Sharing in Clouds," IEEE Trans. Dependable Secure Comput., pp. 1–1, 2021.

[13] Q. Huang, Y. Yang, and J. Fu, "Secure Data Group Sharing and Dissemination with Attribute and Time Conditions in Public Cloud," Ieee Trans. Serv. Comput., vol. 14, no. 4, pp. 1013–1025, Aug. 2021.

[14] A. F. Loe, L. Medley, C. O'Connell, and E. A. Quaglia, "TIDE: A Novel Approach to Constructing Timed-Release Encryption," in Information Security and Privacy, Acisp 2022, Cham, 2022, vol. 13494, pp. 244–264.

[15] K. Yuan, Y. Wang, Y. Zeng, W. Ouyang, Z. Li, and C. Jia, "Provably Secure Security-Enhanced Timed-Release Encryption in the Random Oracle Model," Secur. Commun. Netw., vol. 2021, p. 5593363, May 2021.

[16] X. A. Wang, A. K. Sangaiah, N. Nedjah, C. Shan, and Z. Wang, "On the Security of a CCA-Secure Timed-Release Conditional Proxy Broadcast Re-encryption Scheme," in Advances on P2p, Parallel, Grid, Cloud and Internet Computing, 3pgcic-2018, Cham, 2019, vol. 24, pp. 192–198.

[17] K. Emura, A. Miyaji, and K. Omote, "A Timed-Release Proxy Re-Encryption Scheme," Ieice Trans. Fundam. Electron. Commun. Comput. Sci., vol. E94A, no. 8, pp. 1682–1695, Aug. 2011.

[18] W.-B. Kim, S.-H. Kim, D. Seo, and I.-Y. Lee, "Certificateless Group to Many Broadcast Proxy Reencryptions for Data Sharing towards Multiple Parties in IoTs," Wirel. Commun. Mob. Comput., vol. 2022, p. 1903197, Jun. 2022.

[19] P. Xu, T. Jiao, Q. Wu, W. Wang, and H. Jin, "Conditional Identity-Based Broadcast Proxy Re-Encryption and Its Application to Cloud Email," Ieee Trans. Comput., vol. 65, no. 1, pp. 66–79, Jan. 2016.

[20] K. Liang, Q. Huang, R. Schlegel, D. S. Wong, and C. Tang, "A Conditional Proxy Broadcast Re-Encryption Scheme Supporting Timed-Release," in Information Security Practice and Experience, Berlin, Heidelberg, 2013, pp. 132–146.

[21] T. Zheng, Y. Luo, T. Zhou, and Z. Cai, "Towards differential access control and privacy-preserving for secure media data sharing in the cloud," Computers & Security, vol. 113, p. 102553, Feb. 2022.