

A Privacy-Enhanced Retrieval Technology for the Cloud-assisted Internet of Things

Tian Wang, Quan Yang, Xuewei Shen, Thippa Reddy Gadekallu, *Senior Member, IEEE*,
Weizheng Wang*, *Student Member, IEEE*, Kapal Dev, *Member, IEEE*

Abstract—In the Cloud-assisted Internet of Things (IoT), most of the data is sent to the cloud for storage and processing. Data privacy and security are extreme concerns since retrieving data from the cloud will yield privacy disclosure risk due to the cloud’s openness. To this end, this paper proposes PERT, a privacy-enhanced retrieval Technology for Cloud-assisted IoT. This architecture is designed through an implicit index maintained by edge servers and a hierarchical retrieval model that preserves data privacy by hiding the information of data transmission between the cloud and the edge servers. For the hierarchical retrieval model, we designed a data partition strategy. The edge server stores partial data. In this way, data privacy is preserved since the attacker must get the data maintained by both cloud and edge servers. The detailed performance analysis and extensive experiments have displayed the effectiveness of the technology for data privacy. It is tested that the architecture can efficiently and securely retrieve the stored data while the computation cost is reduced through operation downsizing. Compared with the benchmark cloud encrypted storage model, the time cost of this method is significantly reduced when the number of users is relatively large.

Index Terms—cloud storage, edge computing, Cloud-assisted IoT, Intelligent Computing, privacy retrieval.

I. INTRODUCTION

CURRENTLY, with the continuous development of cloud computing technology, a large amount of data is stored and managed by cloud storage. Cloud storage is one of the core applications of cloud computing services. The huge storage capacity of cloud server attracts a large number of users. This method of outsourcing implies that data is beyond cloud users’ control [1]. Once data is stolen, leaked, or tampered with during retrieval, sensitive and private information will suffer from serious threats. Searchable Encryption (SE) is one of

the potential methods to preserve data privacy. However, most of the traditional SE schemes are well-known for processing static dataset and implementing selective security [2]. This kind of method cannot prevent malicious users from stealing shared information. Even worse, eavesdroppers may reveal, tamper with or infer information by cooperating with servers. The third party becomes not private and efficient enough [3]. All these problems signal that failure to preserve data privacy in the early stages of data analysis, storing, and retrieving data, will prevent the system from getting excellent Quality of Service(QoS) [4], [5].

To address the above issues, it is essential to design a natural framework to overcome these privacy threats to data retrieval and control. At present, there are few researches on layered cloud storage privacy preservation technology based on edge computing. This study envisages the process of collaboration between cloud computing and edge computing, addressing the privacy and resiliency of the retrieval architecture in near real-time [6], [7], [8]. “local-edge-cloud” three-layer retrieval architecture takes advantage of integrating edge computing, realizes to hide data relevance for strengthening data privacy. The SE algorithm [9] was also redesigned for flexibility and confidentiality. Meanwhile, the lightweight scheme based on data cognition with a hierarchical retrieval strategy can offer a new platform to expand application services and solve data privacy. Specifically, this algorithm is tailored to improve privacy preservation, aiming to design a Privacy-Enhanced Retrieval Technology (PERT) for Cloud-assisted IoT. The major contributions of the paper are summarized as follows:

- 1) A layered data partition strategy for “local-edge-cloud” is provided in the paper, which takes cognitive analytics as the basis of partition to achieve semantic security. Privacy violations can be prevented by dividing data into several parts. Accessing individual data blocks from any layer is difficult for recovering the original data; 2) An effective solution for cooperation among edge groups is designed by implicitly building a retrieval index between the local users and edge servers. This index aims to mitigate the disadvantage of revealed index content, providing private and quick data retrieval; 3) A mixed cryptogram model is formulated to prevent servers from all levels or eavesdroppers from getting or inferring data. Moreover, it achieves flexible data processing and systematic privacy preservation while keeping scalability.

The rest of the paper is arranged as follows. The related work is introduced in Section II. Section III briefly describes the basic idea of data encryption and protection. In section IV, it presents the details of PERT technology from three aspects.

T. Wang is with BNU-UIC Institute of Artificial Intelligence and Future Networks, Beijing Normal University (BNU Zhuhai), Zhuhai 519000, China, and also with the Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, Zhuhai 519000, China, and the College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China (e-mail: cs_tianwang@163.com).

Q. Yang is with the College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China (e-mail: cs_yangquan@163.com).

X. Shen is with the College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China; and the Department of Financial Science and Technology, the Industrial and Commercial Bank of China, Suzhou city branch, Suzhou 215000, China. (e-mail: shenxw_666@163.com).

Thippa Reddy Gadekallu is with the School of Information Technology and Engineering, Vellore Institute of Technology, Tamil Nadu, India (e-mail: thippareddy.g@vit.ac.in).

Weizheng Wang is with the Computer Science Department, City University of Hong Kong, Hong Kong (e-mail: weizheng.wang@ieec.org).

Kapal Dev is with Department of Institute of Intelligent Systems, University of Johannesburg, South Africa (e-mail: kapal.dev@ieec.org).

First, a semantic security principle is provided to divide data through qualitative and quantitative cognitive analytics. Then, the retrieval index between the location and edge server groups is implemented based on implicit chaining. Finally, a fine-grained encryption retrieval scheme is introduced to secure the data search process. Section V illustrates experiments and the results. Finally, the paper is concluded in Section VI.

II. RELATED WORK

In recent years, cloud-assisted IoT has received more and more attention [10], [11]. In particular, a central theme of many previous studies has been cloud storage security [12]. This article outlines cloud storage security from three aspects.

On the one hand, it focuses on the research of cloud storage of big data. Updating large data sets requires engineering and discretization. However, the traditional practice is to format new data to update model parameters. This practice is insufficient in time and resources and may introduce additional errors and then augment the workload of calibrating models. Hsu et al. [13] proposed an automated cloud storage system through hot-cold data classification, predicting categorized data. In [14], the timely collection of data is the crucial step to get a proper selection strategy. Only the accuracy and effectiveness of the key data can be guaranteed to choose the correct decision. Although the above researches are concerned with commitment validation and forking attacks, no security mechanism during data transmission and storage in the cloud has been found.

Therefore, data retrieval with the introduction of cryptography becomes very feasible. Wang et al. [15] proposed a secure data segmentation scheme based on homomorphic encryption. Considering that the available sensors are energy-intensive devices and have limited computing capacity, Wang et al. [16] proposed partial outsourcing encryption and decryption method based on group key policy attributes. Also, other encryption methods, such as mixed encryption, asymmetric encryption, and authenticated encryption, have been introduced. As the pervasive networks outstretch into intelligent computing environments, researchers designed a scheme that guarantees the security and availability of the remote access [17].

Other solutions are based on SE. It is a promising technology to preserve data privacy in intelligent systems, which shows better performance in cloud platforms. Theoretically speaking, SE schemes can be divided into two categories: public-key encryption and symmetric private key encryption. Xu et al. [15] presented a lightweight searchable public-key scheme. Based on a cloud computing model, the proposed strategy saved considerable time and reduced tremendous energy costs. However, these works face challenges such as certificate authorization and data management.

III. SYSTEM MODEL

In our solution, the data transfer and access are achieved collaboratively among edge and cloud servers. Our system model is shown in Fig. 1:

At the top of Fig. 1 is the “Three-Layer Storage Scheme”:

- 1) Data owner will run data partition; The data is divided

into two parts: the most private part of data, denoted as core-data, stored in local terminals; the other part is called non-core data; 2) The data owner sends non-core data to the edge server; 3) The edge server will divide the non-core data into the Public Data (PUB) and the Private Data (PRI). Edge servers generate ciphertext through our designed flexible SE algorithm for non-core data and sends encrypted PUB to cloud servers. In short, through the data partitioning technology, the data is divided into three parts: core-data, PUB, and PRI. At the bottom of Fig. 1 is the Smart Check strategy: 1) Data users send data retrieval requests to edge servers; 2) Cloud servers select all matching ciphertext and return query results to users hierarchically through edge servers; 3) Users decrypt the returned data and retrieve the expected results.

A. Users’ Data Partition and Storage

For semantic security, the scheme must divide data before uploading data to the cloud. In the method, the core-data will be stored locally by users, while the rest non-core data will be uploaded to the edge servers for further processing. The edge servers will partition data in the process of data encryption and uploading. Meanwhile, the non-core data will be divided into the PUB and the PRI through a hidden structure method. We assume that the cloud is untrustworthy and the edge server is trustworthy. So PUB will be stored in the cloud, and PRI will be held in the edge layer. It is worth noting that a large part of the original data is still stored in the cloud.

B. Users’ Data Retrieval from Local

For the “Intelligent Query Scheme” to search data, the retrieval process of local-to-edge and edge-to-cloud phases is constructed based on the storage structure. The scheme especially constructs the implicit links to accelerate the response process of the edge servers and make up for the lack of index information revealing. It becomes an implicit relationship between the local data and PRI stored in the edge servers.

C. Users’ Data Retrieval from Edge

After the local-to-edge retrieval, the edge servers perform the second stage through the hidden structure. The search trapdoor is generated to search for the rest of the data in the cloud. It is also a kind of authorization for edge-to-cloud retrieval. Finally, each layer returns retrieval results to the local users. The servers cannot judge specific keywords, let alone infer the original data from the retrieved data. To some extent, a lightweight algorithm also makes up for the efficiency loss caused by the hierarchical retrieval.

D. Users’ Data Retrieval from Local

For the “Intelligent Query Scheme” to search data, the retrieval process of local-to-edge and edge-to-cloud phases is constructed based on the storage structure. The scheme especially constructs the implicit links to accelerate the response process of the edge servers and make up for the lack of index information revealing. It becomes an implicit relationship between the local data and PRI stored in the edge servers.

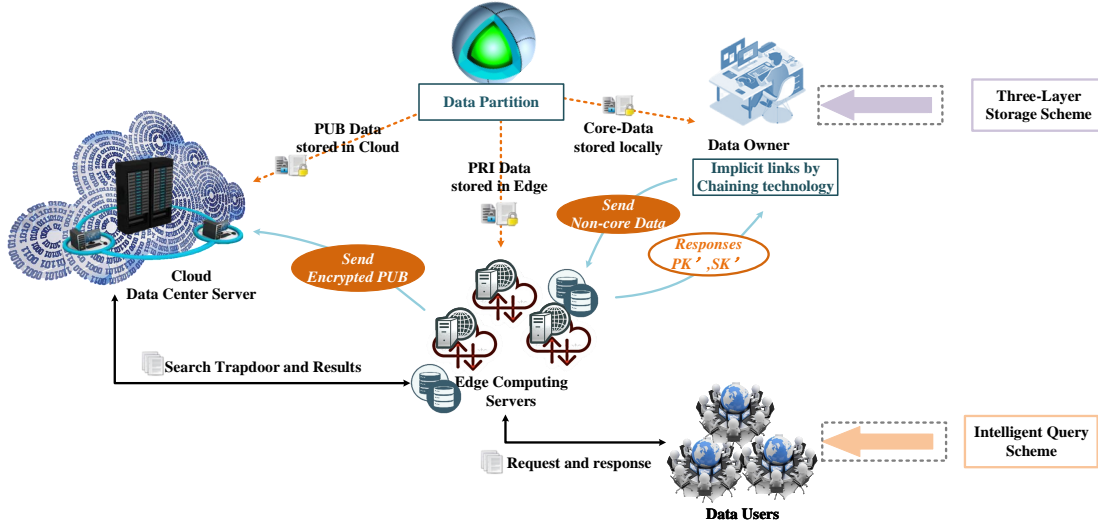


Fig. 1. The PERT Technology for Privacy Preservation.

E. Users' Data Retrieval from Edge

After the local-to-edge retrieval, the edge servers perform the second stage through the hidden structure. The search trapdoor is generated to search for the rest of the data in the cloud. It is also a kind of authorization for edge-to-cloud retrieval. Finally, each layer returns retrieval results to the local users. The servers cannot judge specific keywords, let alone infer the original data from the retrieved data. To some extent, a lightweight algorithm also makes up for the efficiency loss caused by the hierarchical retrieval.

IV. OUR PROPOSED PERT

The formed intelligent networking through cloud computing needs to address the main problem of data privacy [18]. PERT can effectively preserve privacy. We will elaborate on the details of PERT technology from three aspects.

A. Data Partition Strategy based on Cognitive Analytics for Semantic Security

1) *Core-Data Extraction*: According to the amount of information, the largest one can be regarded as the core-data, which can also be seen as the most essential part of data [19]. This part is key to the retrieval, but it tends to develop into hidden severe trouble. With the growing complexity in unknown parts, it becomes more difficult to speculate the original data through access to non-core data. Therefore, data cognition is introduced to narrow down the core-data area gradually, that can help in realizing the data extraction algorithm based on semantic protection.

The information entropy is generally used in the text recommendation algorithm. The basic theory of selecting metrics of spatial vector has been utilized to quantify data information. It refers to the probability of a particular message, which means the uncertainty degree of information during transmission. Data extraction is the process of computing the differences

between each dense point. The average value of the point sets can minimize the nearest point to the center of dense point sets. Later, the results will be judged and calculated according to the elements mentioned above.

2) *Data Partition*: The size of the extracted data is judged by the calculation result of information entropy. After being encoded by the Hash-Solomon code, the data will be divided into k parts and generates n redundant data. The stored data is mapped so that each block can correspond to the generation matrix and then hash the data matrix \bar{D} . Multiplying the encoding matrix with the matrix \bar{D} generates k blocks of data which constitutes matrix X . These blocks are divided into two categories: source data and redundant data. Accordingly, the decoding process of the data block is to inverse matrix operation by using k blocks which satisfy the recovery condition.

Following the coding rules, when selecting parameter w , the condition to be met is $k + n \leq 2^w$. In incomplete data, it is possible to prove the difficulty in restoring data by computing the size of different finite fields (denoted as $GF(2^w)$) and the attempt times of data recovery. In practical applications, the value of w is often large. In other words, it is difficult to recover unconventional data, which ensures data security.

B. Building Index by Implicit Chaining

When local users send requests to the edge servers for data retrieval, the servers respond to those requests by enumerating all the current sub-servers to locate the requested data [20]. To protect user data from being exposed by the retrieval process, to realize the correspondence between core data and non-core data, an effective implicit index needs to be considered within the retrieval scheme.

1) *Construction of Implicit Index*: We modify the chaining technique, linking one user's uploading data implicitly corresponding to edge sub-severs. The primary basis for index construction is the process of storing core-data locally and uploading non-core data. The preprocessing algorithm generates the file sequence number for the core-data as $ID(f)$.

Each local user runs the implicit link scheme to generate and store the index, which can be represented by $(\lambda_c, ID(f))$ ($\lambda_i = \rho_1(\kappa'_s, D \parallel i), 1 \leq i \leq c$). The ρ_1 is a secure pseudorandom function, and the D is the ID of the edge sub-server. The κ'_s is a private key as a response to the local user from the edge server, described in detail in the next section. As the local users upload data to the sub-server once, the counter c will become $c + 1$. Then, the index list will generate by the following operations:

$$\begin{aligned} \lambda_1 &\leftrightarrow \langle \lambda_0 \parallel 0^r \rangle \oplus \rho_2(\theta_1, \lambda_1) \leftrightarrow ID(f_1); \\ \lambda_2 &\leftrightarrow \langle \lambda_1 \parallel \theta_1 \rangle \oplus \rho_2(\theta_2, \lambda_2) \leftrightarrow ID(f_2); \\ &\dots \\ \lambda_c &\leftrightarrow \langle \lambda_{c-1} \parallel \theta_{c-1} \rangle \oplus \rho_2(\theta_c, \lambda_c) \leftrightarrow ID(f_c). \end{aligned} \quad (1)$$

where ρ_2 is a secure pseudorandom function, and the $\theta_i (1 \leq i \leq c)$ is a random private key derived from the counting result. The data structure is not specially required in the context, and the result will be calculated by bit operation. The correctness of the operation can be proved by Theorem 1.

Theorem 1 : Suppose that the secure pseudorandom functions ρ_1 and ρ_2 are both collision-free or have a small probability. The proposed algorithm is correct, except with a negligible chance.

Proof of Theorem 1 :

Here comes $\langle \lambda_{c-1} \parallel \theta_{c-1} \rangle \oplus \rho_2(\theta_c, \lambda_c)$.

When $\rho_2(U') = \rho_2((\theta_c, \lambda_c), \rho_1(\kappa'_s, D \parallel i))$ can be established, the algorithm will find a corresponding prefix that equals to $\rho_2(U')$, where the U' is assumed as the message to be calculated. Because both secure pseudorandom functions ρ_1 and ρ_2 are collision-free, there is little chance of exception. Hence, after the first matching, the algorithm can do the second one for (θ_c, λ_c) until the target index entry is found.

2) *Implicit Index Implementation*: An example will be given here to directly show the serviceability of the implicit index in Fig. 2. We take the user in the middle as an example. It can be seen that the user's data is stored in three edge sub-servers, where ID_1, ID_2, \dots, ID_5 are the storage sequences, and the sub-servers are denoted as D_1, D_2, D_3 in numerical order. The dashed lines represent the visual effect of the implicit index. It is assumed that the user has stored four copies of data ID_1, ID_2, ID_3, ID_4 and built the corresponding index in the initial state $D_1 \rightarrow ID_1, \dots, D_3 \rightarrow ID_4$. Then, as shown by the red line, the user stores the fifth data ID_5 now. The formation of the red line means that the index will be changed. It is generated by the previous round of results: After calculating the new $D_1 \rightarrow ID_5$ with $D_1 \rightarrow ID_1$, add the results into the list, and the previous round of records $D_1 \rightarrow ID_1$ is deleted.

When the system runs the first-level retrieval, the user just selects the final record of the related sub-server in the index (θ_c, λ_c) to obtain the actual link as follows:

$$\langle \lambda_{c-1} \parallel \theta_{c-1} \rangle \oplus \rho_2(\theta_c, \lambda_c) \oplus \rho_2(\theta_c, \lambda_c). \quad (2)$$

C. Mixed Cryptogram Mechanism for Hierarchical Retrieval

With the expansion of the application field, people have realized that cloud computing alone is not enough to secure data. On the one hand, the reliability of the cloud server cannot

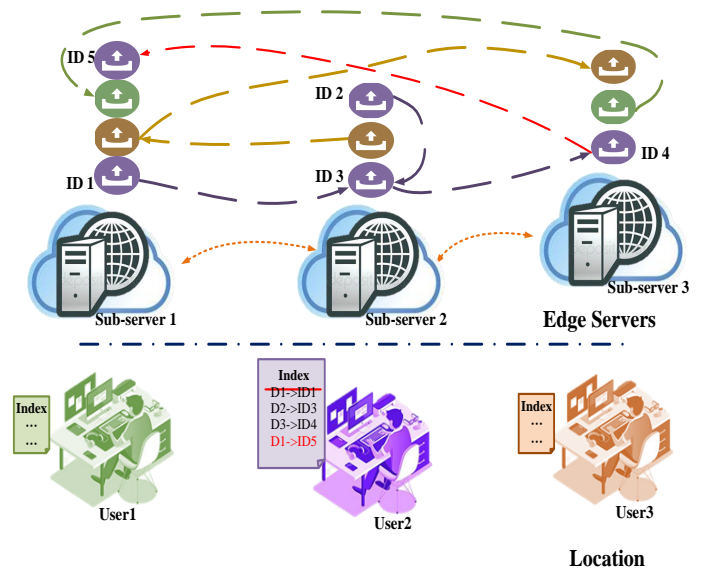


Fig. 2. The Implicit Index Schematic.

be fully ensured for users. It may cause data interference and related retrieval problems, and, as a result, incorrect or incomplete data will be returned. On the other hand, the eavesdroppers and even cloud servers themselves try to infer retrieval keywords and data which can lead to information leaking. The scheme is to protect retrieval privacy, ensure the security and correctness of data retrieval. Therefore, we consider applying lightweight and high-performance SE methods to the scheme.

1) *Encryption Model*: To retrieve the target data, the data owner will use a retrieval mechanism, and then, the target data will be returned from the edge and cloud. Finally, the user-end can decode and decrypt the search results [21]. In terms of security, the scheme will guarantee that both the inter-layer retrieval transparency and untrusted third party (e.g. the cloud server) cannot get any information about the actual data. The encryption-related details are described as follows. First, let \mathbb{Z}_1 and \mathbb{Z}'_1 be the algebraic groups, and computing $\hat{\Theta} : \mathbb{Z}_1 \times \mathbb{Z}'_1 \rightarrow \mathbb{Z}_2$ be pairing operations. If $\mathbb{Z}_1 = \mathbb{Z}'_1$, we can define $\hat{\Theta}$ as a symmetric group. The function for pairing operation $\hat{\Theta} : \mathbb{Z}_1 \times \mathbb{Z}'_1 \rightarrow \mathbb{Z}_2$ with the following properties. Let \mathbb{Z}_1 and \mathbb{Z}_2 with same order f :

- Effectiveness: there are two elements T and $W \in \mathbb{Z}_1$, constructing a polynomial-time algorithm for $\hat{\Theta}(T, W) \in \mathbb{Z}_2$.
- Bilinearity: there are $n, m \in \mathbb{Z}_f^*$, equation $\hat{\Theta}(nT, mW) = \hat{\Theta}(mT, nW) = \hat{\Theta}(T, W)^{nm}$.
- Calculability: in a polynomial time, $\hat{\Theta}(T, W)$ can be computed.

(1) *Parameter Preset*: This is the earliest operation in the scheme. It provides the initial parameters for other system operations according to private requirements. The parameters include two factors: the master public key and the private key. Here, the public key encryption and the private key encryption will be combined appropriately. The two will be computed on edge servers and send a responding message to the data owner.

First of all, we set $PG(1^\mu)$ as a generator and take a security parameter 1^μ as input. The keywords space is denoted as $\Phi = \{0, 1\}^*$. Hence, we can denote this step as $Sp(1^\mu, \Phi) \rightarrow (f, \mathbb{Z}_1, \mathbb{Z}_2, T, \hat{\theta})$, where is generated by running $PG(1^\mu)$. And, choosing $e \in Z_f^*$ randomly, lets $W = eT$ and outputs the master key $\kappa_s = e$. Let select two hash functions $h_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_1$ and $h_2 : \mathbb{Z}_2 \rightarrow \{0, 1\}^{256}$ and output the public key $\kappa_p = (f, \mathbb{Z}_1, \mathbb{Z}_2, T, \hat{\theta}, W, h_1, h_2, \Phi)$. This operation is run by the edge server; the edge will keep κ_s for data owner secretly; the master public key can be told to all users and cloud.

(2) Hidden Structure: For the sake of search performance, the ciphertext will be reorganized. A kind of structure that hides the related organization is designed to maintain semantic security by masking the relevance between the keywords and the data items. All ciphertext with the same keywords can form a chain through some corresponding hidden information. This design consists of two parts: the public one and the private one. Then, the information is kept as/keeps a record of a public head that connects each chain. When it runs in the edge servers, the first matching text will be found through the record until the final object is spotted.

Take this operation as $Hs(\kappa_p) \rightarrow (P_r, P_u)$, where initialize P_r is the private part of the hidden structure, and P_u is the public part by running calculation when κ_p is input. Randomly select $v \in Z_f^*$, computing $P_u = v \cdot T$, $P_r = (v)$. The list of P_r is $(v, \{(\varphi, U[v, \varphi]) \mid \varphi \in \Phi\})$, where $U[v, \varphi] \in \mathbb{Z}_2$. Algorithm $Hs(\kappa_p)$ will be calculated on the edge server. And its outputs P_r, P_u will be stored in edge and cloud, respectively. The private part stored in edge can be used as an index that connects the search between edge and cloud.

(3) Encrypted Ciphertext: This is still a combination of public and private key encryption. The operations to extract expected keywords are also included, which are not just for encrypting the plaintext. At the same time, it must output the pre-generated products about the hidden structure. For the initial phase, the keyword generation will be implemented through a series of operations. And, for the subsequent operations, it can provide updates for the hidden structures.

Algorithm $Ec(\kappa_p, \varphi, P_r)$ takes κ_p , one keyword $\varphi \in \Phi$, and the private part P_r as inputs. The different keywords of the same segment of encrypted content are computed and saved one by one. The procedure is as Alg. 1.

(4) Search Trapdoor: The expected keyword and the master private key are taken as the input to generate a keyword-related search trapdoor. This operation is mainly applied to the transmission of edge-to-cloud search instructions to ensure security.

Inputting a keyword φ , and the user's private key κ_s , the edge server executes $St(\varphi, \kappa_s)$. Through compute the trapdoor $H_{\varphi_i} = e \cdot h_1(\varphi_i)$, the edge will deliver output as an authorized instruction for performing follow-up query tasks.

Theorem 2 : Let the hash functions h_1 and h_2 are both modeled as random oracle $\mathbb{N}_{h_1}(\cdot)$ and $\mathbb{N}_{h_2}(\cdot)$ to create l_1 and l_2 . Suppose that there are $C \in \mathbb{C}$ hidden structures, and a probabilistic polynomial time (PPT) attacker β has an advantage of Adv_β to break our system, in which β models

Algorithm 1 The Encryption for Stored Cloud Part.

Input: The master public key κ_p ; a keyword φ_i ; and the public part P_r .

Output: An encrypted field corresponding to a keyword R_i ;

- 1: Choose $x = (\varphi_i, U[v, \varphi_i])$ from P_r for φ_i ;
- 2: **if** $x \neq \emptyset$ **then**
- 3: Randomly choose a number $r_i \in Z_f^*$;
- 4: Compute $u_i = U[v, \varphi_i] \cdot r_i$;
- 5: Set $R_i = (h_2(U[v, \varphi_i]), u_i)$;
- 6: Update $U[v, \varphi_i] = r_i$ in P_u ;
- 7: Then, add R_i to the ciphertext set \mathbb{R} as output.
- 8: **else**
- 9: Select $x = U[v, \varphi_i] \in \mathbb{Z}_2$ randomly, then add it into P_u ;
- 10: Compute
- 11: $R_i = (h_2(\hat{\theta}(v \cdot W, h_1(\varphi_i))), \hat{\theta}(v \cdot W, h_1(\varphi_i)) \cdot x)$.
- 12: Add R_i to the ciphertext set \mathbb{R} .
- 13: **end if**
- 14: After calculating R_i one by one, the output is the ciphertext set \mathbb{R} finally.

the problems of querying the P_r , querying the search trapdoor H_{φ_i} and querying the ciphertext by responding as $\varepsilon_1, \varepsilon_2$ and ε_3 , respectively. Then, we denote a PPT algorithm ρ that can solve these issues in $PG(1^\mu)$:

$$Adv_\rho(1^\mu) \geq \frac{256}{2e^4(\varepsilon_1 + \varepsilon_2)^4(l_2 + \varepsilon_3 + 1)} Adv_\beta, \quad (3)$$

where e is a logarithmic function.

Proof :

We will design an algorithm ρ to determine the three problems which may break our scheme.

• Suppose that algorithm ρ may abort in the trapdoor query, the P_r query. Any one of them will cause ρ to fail. Hence, we can obtain that $P(\bar{F}) = (1 - \partial)^{\varepsilon_1 + \varepsilon_2} \partial^4$. Let $\partial = \frac{4}{\varepsilon_1 + \varepsilon_2 + 4}$, the equation will be:

$$P(\bar{F}) \geq \frac{256}{e^4(\varepsilon_1 + \varepsilon_2)^4}. \quad (4)$$

• Suppose that algorithm ρ will do not abort in the attack β . If we denote Q_t as the query for keyword φ_t , and the attacker β will send a guess t' to algorithm ρ , we have that $Adv_\beta = P(Q_{t=t'}) - \frac{1}{2}$. Hence, we can draw some ratiocination.

$$\begin{aligned} P(Q_{t=t'}) - \frac{1}{2} &= P(Q_{t=t'} | Q) P(Q) + \\ &P(Q_{t=t'} | \bar{Q}) P(\bar{Q}) - \frac{1}{2} \\ &= P(Q_{t=t'} | Q) - \frac{1}{2}, \end{aligned} \quad (5)$$

$$P(Q) \geq Adv_\beta. \quad (6)$$

For the attacker β , it has the same chance to cause hash query \mathbb{N}_{h_2} to send two challenge pairs. And the algorithm ρ will generate the response for the problem, which denoted this records as $l_2 + \varepsilon_3 + 1$. There,

$$P(Q_i) \geq \frac{1}{2(l_2 + \varepsilon_3 + 1)} Adv_\beta. \quad (7)$$

- Finally, combining with Equ. 4, we can conclude that

$$Adv_\rho(1^\mu) \geq \frac{256}{2e^4(\varepsilon_1 + \varepsilon_2)^4(l_2 + \varepsilon_3 + 1)} Adv_\beta. \quad (8)$$

(5) Target Retrieving: It is used to seek ciphertext segments that match the keyword. The result of the retrieval is finally obtained from the cloud.

The algorithm **Sc** takes the master public key κ_p , the private part of hidden structure P_u , all ciphertext set \mathbb{R} and a search trapdoor H_{φ_i} as input, let $\mathbb{R}' = \emptyset$, and the detailed steps are as Alg. 2.

Algorithm 2 The Search Algorithm Program.

Input: The master public key κ_p ; the private part of hidden structure P_u ; the ciphertext set \mathbb{R} ($\mathbb{R}[i]$ is the i th element in \mathbb{R} , it can be parsed as $\mathbb{R}[i, 1] \in \{0, 1\}^{256}$ and $\mathbb{R}[i, 2] \in \mathbb{Z}_2$) and the search trapdoor H_{φ_i} .

Output: An encrypted search result as \mathbb{R}' ;

- 1: Compute $U' = \hat{\theta}(P_u, H_{\varphi_i})$;
 - 2: **repeat**
 - 3: Find a $\mathbb{R}[i]$ having $\mathbb{R}[i, 1] = h_2(U')$.
 - 4: Add $\mathbb{R}[i]$ into \mathbb{R}' ;
 - 5: Compute $U' = U'^{-1} \cdot \mathbb{R}[i, 2]$.
 - 6: **until** $\mathbb{R}[i, 1] \neq h_2(U')$.
 - 7: Output \mathbb{R}' , finally.
-

Theorem 3 : Suppose that the hash functions h_1 and h_2 are both collision-free or small probability. The proposed algorithm is correct, except with a negligible probability.

Proof :

$\mathbb{R}[j] = \left(\left(h_2 \left(\hat{\theta}(v \cdot W), h_1(\varphi_i) \right) \right), \hat{\theta}(v \cdot W, h_2(\varphi_i)) \cdot U[v, \varphi_i] \right)$. When $h_2(U') = h_2 \left(\hat{\theta}((v \cdot W), h_1(\varphi_1)) \right)$ can be established, the algorithm **Sc** will find a corresponding prefix that equals to $h_2(U')$. However, the ciphertext does not belong to Φ or P_u . Because both hash functions h_1 and h_2 are collision-free, there is little chance of exception. Hence, after the first matching, algorithm **Sc** can do the second one for φ_i , until all target ciphertext is found finally.

2) Applying Retrieval Scheme to the Three-Layer Structure:

As an example of the scheme, our retrieval algorithm is supposed to cooperate with the public-key encryption with keyword search (PEKS) [9] and symmetric-key encryption: DES. Therefore, the operation of the scheme will produce a different result from the traditional algorithm. Its content is summarized in the following phases.

- (1) Initialization: The edge servers will select 1^μ , run the parameter preset algorithm **Sp** to create (κ_p, κ_s) , and compute **Sp**_{PEKS} to generate (κ'_p, κ'_s) . The pair of public keys (κ_p, κ'_p) is stored in all devices, and, then, the edge server returns a pair of public and private keys (κ_s, κ'_s) to the data owner.
- (2) Data Processing: It can be denoted that a data owner uploads data B to the cloud. After the local-edge phase, the non-core data B' will be sent to the edge servers.

Then, the edge servers will run the hidden structure algorithm **HS** and extract keywords $\{\varphi_1, \varphi_2, \dots, \varphi_n\}$ from B' . In addition to computing the encrypted ciphertext algorithm **Ec** uses κ_p as an input to form ciphertext $\{R_1, R_2, \dots, R_n\}$, a symmetric key will be selected to run algorithms **Ec**_{PEKS} and **Ec**_{DES}. These processes will create ciphertext (R_{PEKS}, R_{DES}) , which will be inserted output as the uploading set $\{R_1, R_2, \dots, R_n, R_{PEKS}, R_{DES}\}$ to the cloud. Because of these operations, PERT only needs to pair once instead of linearizing with the ciphertext for a keyword M .

(3) Data Searching: If the edge node receives a request for a keyword φ_i , the server will first run the search trapdoor algorithm **St** to create a retrieval order for the next search step. The generated search token H_{φ_i} will be sent to the cloud. Then, the cloud will run the search algorithm **Sc** to realize ciphertext searching. Finally, the data user will receive all of the results and decrypt them to obtain the target data.

D. Security Analysis

In this section, we give a summary of the security or advantages of the system. Firstly, the core data is separated after the data is analyzed by using information entropy. Information entropy describes the degree of uncertainty of information. The greater the entropy of information, the less information the data contains, and vice versa. The introduction of the method of data cognition can effectively extract the core data. Assume that the attacker obtains the non-core data of the edge and cloud, but the uncertainty and complexity of the non-core data make it difficult for the attacker to infer valuable information. After the amount of data stored locally is determined by the size of the core data, the Hash-Solomon algorithm is run to generate k data blocks and n redundant data blocks. In the $k+n$ data blocks, if we have at least k data locks, we can recover the original data by combining it with the encoding matrix. But once the number of data blocks is less than k , it cannot be recovered. Therefore, it is very safe to put data blocks in a certain proportion into the side cloud server. It is impossible to recover the original data with any single server's data. We designed the shield query mechanism. According to the query content and related index items, the server cannot infer the original data and guess its keywords. The server can only return the corresponding encryption results to the user. Due to incorporating implicit index, it can flexibly adapt to dynamic data retrieval with edge servers. It reduces the traffic from local devices to the cloud since edge nodes reduce intermediate data while having the benefit of storage and retrieval cost savings.

V. PERFORMANCE ANALYSIS

We use Python 3.5 to implement the experiments. Taking the uploading and retrieving of a dataset as an example, we demonstrate the operation of the scheme. It is listed as several aspects for completing analysis: the cost time, the retrieval time, and the precision. At the same time, we compare our scheme with the effective “cloud storage” scheme [22] to illustrate the availability of our design. 5 to 60 edge devices are adopted as edge servers, and the data items in each test

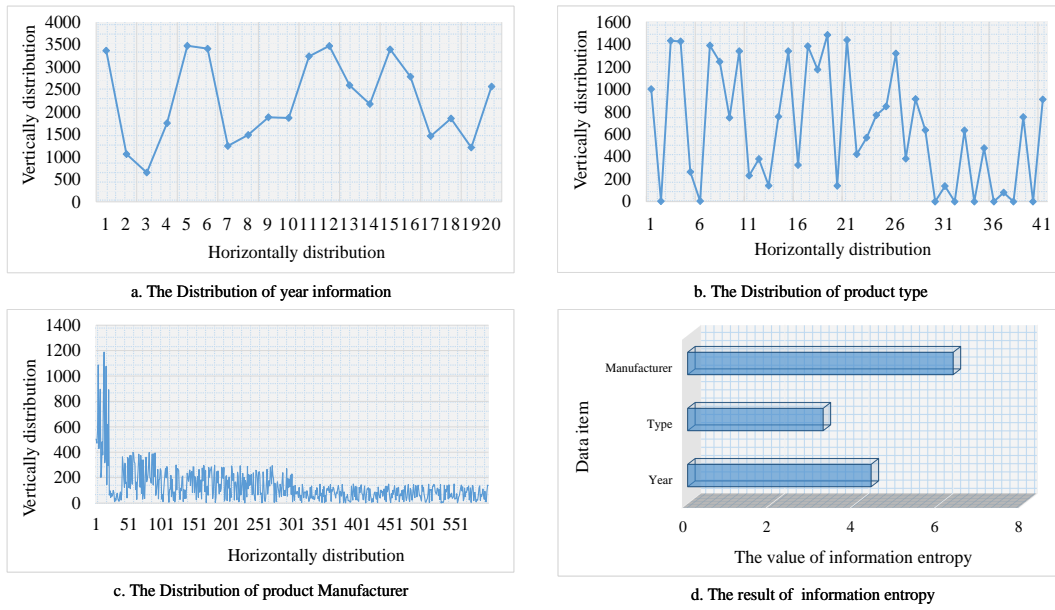


Fig. 3. The Performance of Data Partition.

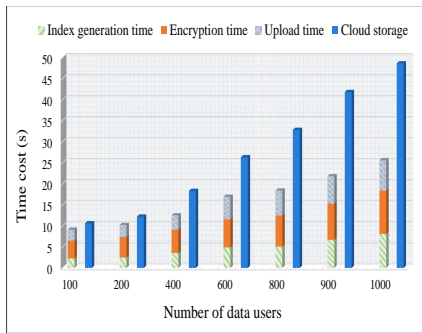


Fig. 4. The Time Cost of Data Storage.

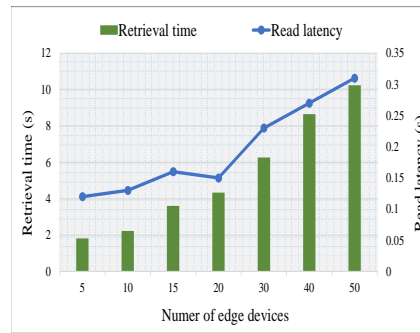


Fig. 5. The Time Cost of Data Retrieval.

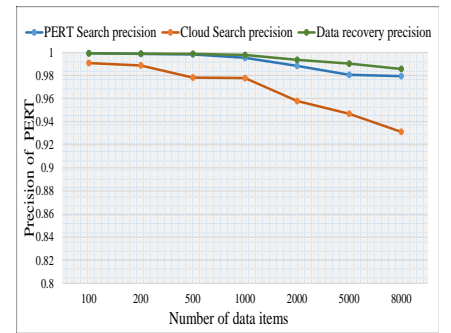


Fig. 6. The Precision of PERT on Search and Recovery.

dataset are set from 100 to 8000. Furthermore, the bandwidth of each edge server is from 10 Mb/s to 1 Gb/s, and all of the edge servers serve for 100 to 1000 data users.

A. Data Partition

The finance dataset accessible to the public is selected as the evaluation dataset since it contains several data items, which represents the real-world need for data privacy (<https://www.federalreserve.gov/>). Specifically, the scheme calculates the information distribution and obtains the corresponding information entropy, as shown in Fig. 3. We first calculate and describe the information distribution with the year, type, and manufacturer of the product as an example. Fig. 3 (a)-(c) display the differences of data distributed clearly. Finally, according to the amount of information, the information entropy to be compared is shown in Fig. 3 (d). Information entropy describes the degree of uncertainty of information. The greater the entropy of information, the less information the data contains, and vice versa. The introduction of the data cognition can effectively extract the core data. Therefore, type information can be considered core-data.

B. Data Storage

As shown in Fig. 4, the handling time for the storage phase in the scheme is divided into three parts: the index generation, the encryption, and the upload, which can explain the data-storage process in detail. As a baseline, the result is compared with the cloud storage scenario in [22], which is denoted as “cloud storage”. It can be seen that the total time of data storage increases with the number of data users in all schemes since more data content will lead to a longer processing time. Based on the observations, the scheme costs much less time while the “cloud storage” rises at a high rate from the point of 400. It must be noted that the operation of each part of PERT is more stable since PERT boasts better adaptability, especially in the situation of big data.

C. Data Retrieval

In Fig. 5, it can be observed that the average reading latency increases with the number of edge devices. However, it takes only a tiny part of the entire query time since more data queries lead to decreasing bandwidth and the resources in the edge are constrained. The reading latency is mainly caused by

the communication and cooperation of the servers at different levels. It can be seen from the diagram that the retrieval time is affected by the addition of edge devices because, with the increasing number of users' queries, each edge server will have more neighbors to connect. It can maintain linear growth, which is a more practical situation for big data.

D. Scheme Precision

Finally, we examine the precision and availability of the scheme, and the results are shown in Fig. 6. Compared with the traditional "cloud search", the scheme's data searching and recovery precision are always above 0.98, while the precision declines with the increase of data items. Their differences in stability can be indicated by the curve. Meanwhile, it is found that both the data retrieval and recovery precision decrease when the number of data items grows large. Obviously, due to the collaboration of edge servers, the precision and availability remain at a satisfactory level compared with a simple/traditional data search in the cloud-only environment.

E. Industrial Applications

For institutions and individuals, data is valuable and even subject to trade secrets [23]. Using PERT technology can effectively solve these problems and increase the trust mechanism between enterprises. For example, the PERT algorithm can be well applied to cloud medical records. We found that if users partition their medical records through the algorithm, store their core data in the medical insurance card, store non-core PRI on edge, and store PUB in the cloud. When people use electronic medical records in medical institutions, they only need to retrieve relevant information to complete the generation of electronic medical records [24]. The interoperability between various medical structures can be well achieved. This would be a breakthrough for the healthcare industry. PERT technology can be used in electronic medical records, intelligent prescriptions, community health systems, secret document storage, and so on.

VI. CONCLUSION AND FUTURE WORK

One of the necessary development visions for high-quality Cloud-assisted IoT is dependable privacy strategies [20]. In our design, the novel collaborative computing between the cloud and edge aims to prevent privacy disclosure during the retrieving by monitoring the information-environment. To optimize system performance and preserve the privacy of data in storing and retrieving, we propose PERT. The scheme effectively supports such extensive data distribution, processing, and analysis, and its precision can reach over 96%. It reduces the traffic time from local devices to the cloud; meanwhile, it can save the storage and retrieval costs. Compared with the benchmark cloud encrypted storage model, the time cost of this method is significantly reduced when the number of users is relatively large. Through extensive simulations, the results demonstrate the superior performance of our algorithm. For future studies, the potential advantages of collaborations for big data and edge computing will be further explored

to address more issues of related parameter privacy in the industry [25], such as securing parameters sharing among multi-models and improving the efficiency of retrieving the same content.

ACKNOWLEDGMENT

The above work was supported in part by grants from The Natural Science Foundation of Fujian Province of China (No. 2020J06023), The National Natural Science Foundation of China (NSFC) under Grant No. 61872154 and UIC Start-up Research Fund under Grant No. R72021202.

REFERENCES

- [1] S. Zhou, M. Ke, and P. Luo, "Multi-camera transfer gan for person re-identification," *Journal of Visual Communication and Image Representation*, vol. 59, no. 1, pp. 393–400, 2019.
- [2] T. Wang, P. Wang, S. Cai, X. Zheng, Y. Ma, W. Jia, and G. Wang, "Mobile edge-enabled trust evaluation for the internet of things," *Information Fusion*, vol. 75, pp. 90–100, 2021.
- [3] T. Wang, Y. Mei, X. Liu, J. Wang, H.-N. Dai, and Z. Wang, "Edge-based auditing method for data security in resource-constrained internet of things," *Journal of Systems Architecture*, vol. 114, p. 101971, 2021.
- [4] A. P. Plageras, K. E. Psannis, C. Stergiou, H. Wang, and B. Gupta, "Efficient iot-based sensor big data collection-processing and analysis in smart buildings," *Future Generation Computer Systems*, vol. 82, pp. 349–357, 2018.
- [5] C. L. Stergiou, A. P. Plageras, K. E. Psannis, and B. B. Gupta, "Secure machine learning scenario from big data in cloud computing via internet of things network," in *Handbook of computer networks and cyber security*. Springer, 2020, pp. 525–554.
- [6] C. L. Stergiou, K. E. Psannis, and B. B. Gupta, "Iot-based big data secure management in the fog over a 6g wireless network," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5164–5171, 2021.
- [7] C. Stergiou, K. E. Psannis, B. B. Gupta, and Y. Ishibashi, "Security, privacy & efficiency of sustainable cloud computing for big data & iot," *Sustainable Computing: Informatics and Systems*, vol. 19, pp. 174–184, 2018.
- [8] C. L. Stergiou, K. E. Psannis, and B. B. Gupta, "Infemo: Flexible big data management through a federated cloud system," *ACM Transactions on Internet Technology*, 2020.
- [9] T. Saito and T. Nakanishi, "Designated-senders public-key searchable encryption secure against keyword guessing attacks," vol. 1, pp. 496–502, Nov 2017.
- [10] A. K. Sangaiah, D. V. Medhane, T. Han, M. S. Hossain, and G. Muhammad, "Enforcing position-based confidentiality with machine learning paradigm through mobile edge computing in real-time industrial informatics," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4189–4196, 2019.
- [11] A. K. Sangaiah, D. V. Medhane, G.-B. Bian, A. Ghoneim, M. Alrashoud, and M. S. Hossain, "Energy-aware green adversary model for cyber-physical security in industrial system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3322–3329, 2020.
- [12] Y. Li, Y. Yu, R. Chen, X. Du, and M. Guizani, "Integritychain: Provable data possession for decentralized storage," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1205–1217, 2020.
- [13] Y. F. Hsu, R. Irie, S. Murata, and M. Matsuoka, "A novel automated cloud storage tiering system through hot-cold data classification," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, July 2018, pp. 492–499.
- [14] S. M. Ehsan and B. E. Zechman, "Complex adaptive systems framework to simulate the performance of hydrant flushing rules and broadcasts during a water distribution system contamination event," *Journal of Water Resources Planning and Management*, vol. 143, no. 4, p. 04017001, Jan 2017.
- [15] P. Xu, S. He, W. Wang, W. Susilo, and H. Jin, "Lightweight searchable public-key encryption for cloud-assisted wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3712–3723, Oct 2018.
- [16] Q. Wang, C. Yu, F. Li, H. Wang, and L. Cao, "A group key-policy attribute-based encryption with partial outsourcing decryption in wireless sensor networks," *Security and Communication Networks*, vol. 9, no. 17, pp. 4138–4150, Nov 2016.

- [17] C. Stergiou, K. E. Psannis, A. Plageras, and Y. Ishibashi, "Algorithms for efficient digital media transmission over iot and cloud networking," *Journal of Multimedia Information System*, vol. 5, no. 1, pp. 1–10, Feb 2018.
- [18] T. Wang, Y. Lu, J. Wang, H.-N. Dai, X. Zheng, and W. Jia, "Eihdp: Edge-intelligent hierarchical dynamic pricing based on cloud-edge-client collaboration for iot systems," *IEEE Transactions on Computers*, vol. 70, no. 1, pp. 1285–1298, 2021.
- [19] T. Wang, D. Zhao, S. Cai, W. Jia, and A. Liu, "Bidirectional prediction-based underwater data collection protocol for end-edge-cloud orchestrated system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 7, pp. 4791–4799, 2020.
- [20] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Transactions on Services Computing*, vol. 13, no. 2, pp. 289–300, 2020.
- [21] Y. Deng, X. Zheng, T. Zhang, C. Chen, G. Lou, and M. Kim, "An analysis of adversarial attacks and defenses on autonomous driving models," in *2020 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2020, pp. 1–10.
- [22] C. Hahn, H. Kwon, and J. Hur, "Toward trustworthy delegation: Verifiable outsourced decryption with tamper-resistance in public cloud storage," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*. IEEE Computer Society, jul 2018, pp. 920–923.
- [23] S. Lee, X. Zheng, J. Hua, H. Vikalo, and C. Julien, "Opportunistic federated learning: An exploration of egocentric collaboration for pervasive computing applications," in *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2021, pp. 1–8.
- [24] S. S. R. Krishnan, M. Manoj, T. R. Gadekallu, N. Kumar, P. K. R. Maddikunta, S. Bhattacharya, D. Y. Suh, and M. J. Piran, "A blockchain-based credibility scoring framework for electronic medical records," in *2020 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2020, pp. 1–6.
- [25] Y. Xu, Q. Zeng, G. Wang, C. Zhang, J. Ren, and Y. Zhang, "An efficient privacy-enhanced attribute-based access control mechanism," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 5, 2020.

Shen.jpg



Xuewei Shen received her B.S. degree in Henan Normal University of China in 2016. She is a master candidate in the National Huaqiao University of China. Her research interests include Edge Computing and Internet of Things.

Reddy Gadekallu.jpg



Thippa Reddy Gadekallu is currently working as Associate Professor in School of Information Technology and Engineering, VIT, Vellore, Tamil Nadu, India. He obtained his Bachelors in Computer Science and Engineering in the year 2003 from Nagarjuna University, India, Masters in Computer Science and Engineering from Anna University, Chennai, Tamil Nadu, India and completed his Ph.D in the domain of Machine Learning from Vellore Institute of Technology, Vellore, Tamil Nadu, India in the year 2017. He has more than 14 years of experience in teaching. He has published more than 80 international/national publications. Currently, his areas of research include Machine Learning, Internet of Things, Deep Neural Networks, Blockchain, Computer Vision.

Wang.jpg



Weizheng Wang received the B.S. degree in software engineering from Yangzhou University, Yangzhou, China, in 2019, the M.S. degrees in computer science and engineering from the University of Aizu, Aizu-Wakamatsu, Japan, in 2021. Now he is a Research Associate in University of Aizu and pursuing the Ph.D. degree at the Department of Computer Science, City University of Hong Kong, Hong Kong. His research interests include applied cryptography, blockchain technology and IoT system.

Wang.jpg



Tian Wang received his BSc and MSc degrees in Computer Science from the Central South University in 2004 and 2007, respectively. He received his PhD degree in City University of Hong Kong in Computer Science in 2011. Currently, he is a professor in the Institute of Artificial Intelligence and Future Networks, Beijing Normal University & UIC. His research interests include internet of things, edge computing and mobile computing. He has 27 patents and has published more than 200 papers in high-level journals and conferences. He has more than 8000 citations, according to Google Scholar. His H-index is 47. He has managed 6 national natural science projects (including 2 sub-projects) and 4 provincial-level projects.

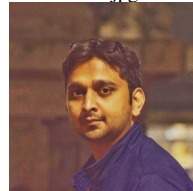
He has managed 6 national natural science projects (including 2 sub-projects) and 4 provincial-level projects.

Yang.jpg



Quan Yang received his BSc degree in Software Engineering from Jiangxi Agricultural University in 2020. Currently, he is a master candidate at the National Huaqiao University of China. His research interests include edge intelligence, mobile computing and edge computing.

Dev.jpg



Kapal Dev is Senior Researcher at MTU, Ireland working on a Industrial project where is Investigating the application of DLT and Smart Contracts in the manufacturing sector. He was a Post-doc at Trinity College Dublin (TCD). He is Associate Editor (AE) in Springer Wireless Networks, Elsevier Physical Communication, IET Quantum Communication, IET Networks, Topic Editor in MDPI Network. He is Guest Editor (GE) in Q1 journals; IEEE TII, IEEE TNSE, IEEE TGCN, IEEE Standard Communication Magazine, Elsevier COMCOM and COMNET. He contributed(ing) as Lead chair in one of ACM MobiCom 2021, Globecom 2021, CCNC 2021 workshops, TPC member of several top conferences. He contributed as PI for Erasmus + International Credit Mobility (ICM), Capacity Building for Higher Education, and H2020 Co-Fund projects. His research interests include Blockchain, Wireless Networks and Artificial Intelligence.