Secure data deduplication for Internet-of-things sensor networks based on threshold dynamic adjustment

Yuan Gao¹,², Hequn Xian¹,² and Aimin Yu³

Abstract
Large amount of data are being produce by Internet-of-things sensor networks and applications. Secure and efficient deduplication of Internet-of-things data in the cloud is vital to the prevalence of Internet-of-things applications. In order to ensure data security for deduplication, different data should be assigned with different privacy levels. We propose a deduplication scheme based on threshold dynamic adjustment to ensure the security of data uploading and related operations. The concept of the ideal threshold is introduced for the first time, which can be used to eliminate the drawbacks of the fixed threshold in traditional schemes. The item response theory is adopted to determine the sensitivity of different data and their privacy score, which ensures the applicability of data privacy score. It can solve the problem that some users care little about the privacy issue. We propose a privacy score query and response mechanism based on data encryption. On this basis, the dynamic adjustment method of the popularity threshold is designed for data uploading. Experiment results and analysis show that the proposed scheme based on threshold dynamic adjustment has decent scalability and practicability.

Keywords
Internet-of-things data deduplication, item response theory, threshold dynamic adjustment, ideal threshold

Date received: 26 November 2019; accepted: 7 February 2020

Handling Editor: Vishal Sharma

Introduction
With the rapid development of Internet-of-things (IoT) sensor networks and applications, an increasing amount of data are generated and stored in cloud services. Cloud storage not only evolves into a major storage scheme but also provides IoT applications with abundant if not limitless storage capability. Duplicate data are almost unavoidable with thousands of IoT sensors working all day long.¹ Data sharing among users or devices has become a common requirement. These challenges are new to cloud storage providers (CSPs). As the amount of uploaded data increases, so does the extent of data redundancy. Statistics show that up to 60% of the data stored in cloud storage are redundant data,² and a large amount of cloud storage resources are consumed, which greatly increases the cost of storage and maintenance of the CSP, especially for IoT-sensor network–based applications.³

¹College of Computer Science and Technology, Qingdao University, Qingdao, China
²State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China
³Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

Corresponding author:
Hequn Xian, College of Computer Science and Technology, Qingdao University, No. 308 Ningxia Road, Qingdao 266071, China.
Email: xianhq@qdu.edu.cn

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
In order to solve the above problems, the CSP generally adopts deduplication technology, which detects identical data objects in the upload stream based on data redundancy. Deduplication enabled systems store only a single copy of the data and create links for other users (or IoT devices) who upload the same data. Deduplication schemes can be classified into block-level and file-level deduplication depending on the size of the objects. Compared with traditional data compression technology, deduplication eliminates not only data redundancy in files but also redundancy between files in shared data sets. However, some users are unaware of data security, resulting in a large amount of private data information being shared without the user’s consent. In recent years, large-scale data leakage events triggered great concern about privacy issues. Therefore, how to protect user privacy while improving the efficiency of cloud deduplication for IoT applications has become a key issue. Harnik et al. first discussed the security problem of client-side deduplication. Since then, the subject has been extensively explored and it is still under investigation in methodological aspects and concrete applications as well. A deduplication scheme for encrypting upload data is proposed for the first time in Bolosky et al., known as convergent encryption. In this scheme, the hash value of the data is used as the encryption key. However, the direct relation between the key and the data reduces the security of the scheme. In Xu et al., the multi-client cross-deduplication scheme XuCDE was first applied for the encrypted ciphertext deduplication problem. The scheme protects the security of private data in the scenario, where external attackers coexist with honest but curious servers. However, in terms of applicability, this scheme has the disadvantages of low encryption efficiency, and it lacks real-time authentication mechanism. In view of the above shortcomings, MRN-CDE (MLE based and random number modified client-side deduplication of encrypted data in cloud storage) was proposed, which applies random number to ensure the instantaneous of the authentication credentials. In order to reduce the amount of computation in the encryption and decryption processes and to ensure data security, the scheme extracts the key from the original data using the KP algorithm in the message locked encryption (MLE) scheme. In addition, some CSP provides users with client-side encryption options, allowing users or IoT sensor networks to encrypt the data before uploading them. This method can effectively protect data privacy. However, even identical plaintext can be encrypted into different ciphertexts by different users, which makes it difficult for the CSP to perform deduplication. Therefore, although the above scheme improves the security of cloud storage, the storage efficiency is still unsatisfying.

For the efficiency of deduplication, Stanek et al. proposed a scheme based on popularity partition. Data of different popularity are encrypted with different encryption methods, which can effectively improve the efficiency of deduplication. The scheme assigns a fixed popularity threshold \( T \) to all data. When the number of copies of certain data in the cloud reaches \( T \), the data are considered to be popular; otherwise, it is regarded as nonpopular data. The cloud server only performs deduplication on popular data, which better protects data privacy while improving the efficiency. The PerfectDedup scheme was proposed by Puzio et al. They used perfect hash function to query the popularity of data with the assistance of a trusted third party. However, the introduction of a trusted third party increases the communication overhead of the CSP, and it causes new security risks. In response to the above problems, Liu et al. proposed a secure deduplication scheme that does not require any third-party server. This scheme uses a password-authenticated key exchange (PAKE) to achieve cross-user key delivery and then cross-user data deduplication. This solution eliminates the dependence on third-party servers and improves its practicability. However, for some popular data, users also need to encrypt them and perform PAKE protocol with other users, resulting in additional computational overhead. Existing popularity-based deduplication schemes do not perform deduplication operations until the total number of data copies reaches the threshold \( T \). But in real-world applications, the amount of privacy data that users upload to the cloud is massive, and as a result, nonpopular data also occupy a large amount of storage space in the cloud. To further reserve cloud storage space, deduplication schemes for nonpopular data were proposed, such as the elliptic curve-based encrypted data deduplication schemes in Zhang et al. and Singh et al. The scheme in Zhang et al. adopts an elliptic curve encryption algorithm, which is more secure and less computational intensive. Different encryption methods are used for popular data and nonpopular data. Client-side deduplication is used for popular data, which significantly reduces storage and bandwidth consumption.

In current practical cloud storage applications, the CSP simply sets a fixed threshold for all data uploaded, which leads to many problems. If the threshold is set too high, for data with low privacy, all copies need to be stored before the threshold is reached. If the threshold is set too low, data with higher privacy will be prematurely deduplicated, which may increase the security risk. Therefore, different thresholds should be set according to the privacy level of the data, and the user’s understanding of the privacy level should be considered at the same time. For example, a frequently used software installation package should have a low threshold, so that it can be quickly processed for deduplication, which minimizes the storage overhead without compromising user’s privacy. And when internal...
confidential files of a company are uploaded, according to the user’s understanding of the privacy level, the CSP could set a relatively large threshold for it, thereby effectively avoiding premature execution of deduplication and better protecting the user data. However, how to recognize the privacy level of each upload data and assign a reasonable threshold for the data according to its privacy level is still a difficult and open problem.

In summary, our work makes the following contribution:

- We propose a deduplication scheme suitable for IoT sensor networks based on threshold dynamic adjustment to ensure the security of upload data and related operations.
- We design the threshold dynamic adjustment mechanism, using the item response theory (IRT) to dynamically adjust the threshold. We use the query feedback mechanism to collect the privacy attitude of most users to determine a reasonable threshold for each data uploaded.
- The concept of ideal threshold is proposed for the first time, which eliminates the disadvantages of unified threshold in traditional schemes.

The remainder of this article is organized as follows. Section “System model and design goals” introduces the system model and design goals. Section “Preliminary knowledge” gives the preliminary knowledge of the scheme and its related formulas. Section “Deduplication scheme” elaborates on the design of our scheme from three parts: privacy score query, data uploading, and privacy score calculation. Section “Security analysis” details security analysis. Section “Performance evaluation and experiments” gives the experimental comparison and analysis. Section “Conclusion” summarizes the work and draws conclusions.

System model and design goals

System model

The system design is based on the IoT sensor network. The system model of this scheme only involves two types of entities: upload user and CSP. Upload user is an abstraction for IoT device, but sometimes it is an actual user in the IoT sensor network. When the system is established, the upload user can interact with the CSP. During the interaction, the upload user can play two roles: data uploader or data observer, and the CSP can only provide data storage and data sharing services for the upload user, without knowing the exact content of the data. The system model is shown in Figure 1.

This model introduces the concept of privacy scores (PR). The PR of the data M is an indicator of privacy risk, and the larger the PR, the higher the privacy of the data. In the IoT sensor network, privacy risk should be measured for data generated by various sensors or devices. In the stage of data uploading, the user sends an upload request to the CSP and calculates the query label of the data using the elliptic curve encryption algorithm. After receiving the upload request, the CSP performs data query and ciphertext comparison, and detects whether M is the first uploaded without leaking the data content. If the CSP finds that the data are stored in the cloud, it returns a suggested PR to the user. The user uploads the encrypted M and its PR rating to the CSP together. After each uploading operation, the CSP adjusts and updates the data’s PR to serve as the feedback information for the subsequent upload user. After consecutive upload and adjustment, each data will acquire a corresponding PR that tends to stabilize gradually and meets the anticipation of most upload users. The CSP calculates the popularity threshold T for M based on PR and performs deduplication operation according to the actual value of T. This not only reduces the consumption of storage space but also avoids the leakage of privacy data. In the data observation stage, only the user who has uploaded the data can make a query request to the CSP, and the CSP returns the encrypted data. Furthermore, although we assume that the CSP is honest but curious, it can perform offline analysis to infer additional information. Therefore, the CSP is not trusted from the fact that users do not expect their private information to be known by any third party.

Design goals

In order to better protect data privacy, the scheme should have the following characteristics:

- Confidentiality of uploaded data: uploaded data require a certain encryption operation.
- Queryability of privacy scores: when users upload data, they can query and get a reasonable privacy score from the CSP as a reference.
• Updateability of privacy scores and thresholds: the privacy scores and thresholds of data can be updated in real time according to the specific upload situation.

Preliminary knowledge

IRT and its characteristics

IRT\textsuperscript{23} is a famous psychological theory, which is often used to analyze the questionnaire results and test data. This theory can infer the probability that the tested user will correctly answer a given question by measuring the ability of the tested user and the difficulty of the specific test item. Moreover, it has been proved that IRT can be applied in cloud computing scenarios.\textsuperscript{24}

The Rasch model\textsuperscript{25} is one of the most common IRT models. It assumes that the probability function of correct response is only related to $\phi_i$, $\gamma_i$, and $\gamma_e$. The item $Q_i$ is represented by a pair of parameters $\omega_i = (\phi_i, \gamma_i)$. $\phi_i$ represents the capability level of the tested user, $\phi_i$ is the extent to which an item can be distinguished, and $\gamma_e$ represents the difficulty of the test item. Because of its few computational parameters and simple structure, the model has the advantage of requiring fewer samples. Invite the user $j$ to answer an item, and if the two-valued “correct” or “error” is used to indicate the answer to the item $Q_i$, then the probability that the item $Q_i$ is correctly answered by the user $j$ is

$$Re_{ij} = \frac{1}{1 + e^{-\phi_i (\gamma_i - \gamma_e)}}$$

Therefore, IRT has two notable features:

• Group stableness: the difficulty of the item is a natural attribute of the item, which is independent of the tested user’s response. In other words, the parameters of an individual project are not only applicable to users currently being tested but also to other types of users.\textsuperscript{26}

• User independence: one tested user will not affect the answer of another tested user to a question, and the answer only depends on the question itself.

General sensitivity calculation method

Generally, the more sensitive the data are, the less likely for the user to disclose it. As shown in equation (2), $|x_i|$ (i.e. the number of $\chi(i,j) = 1$) is used to indicate the number of tested users who wish to expose data item $i$. Then, the number of tested users who refused to disclose data item $i$ is proportional to $\gamma_i$ (the sensitivity of data item $i$). The more sensitive data item $i$ is, the higher the value of $\gamma_i$.\textsuperscript{22}

$$\gamma_i = \frac{N - |x_i|}{N} \quad (2)$$

where $N$ is the number of data items.

General visibility calculation method

In the case where the answer to the question is a binary value, we usually estimate the probability to calculate the visibility of the data.\textsuperscript{27} Assuming that the test project and the tested users are independent of each other, that is, in a test survey, the chance of the tested user answering each question is the same. We can calculate the value of $Re_{ij}$ by multiplying the occurrence probability of $\chi$ in the binary matrix row $\chi_i$ and the occurrence probability of $\gamma$ in the column $\gamma$. In other words, if $|\chi|$ represents the number of $Re(i,j) = 1$ data item set by the user $j$, then the probability $Re_{ij}$ will increase with the increment of user’s tendency to share information, and it will also increase with the decrement of the information item sensitivity. The visibility formula is as shown in equation (3)

$$Re_{ij} = \frac{|\chi_i|}{N} \times \frac{|\gamma_i|}{n}$$

where $N$ is the number of data items and $n$ is the number of users under test. The method of calculating the visibility above is to sample all possible response matrices according to the probability distribution statistically. In fact, the visibility is actually calculated by

$$V(i,j) = Re_{ij} \times 1 + (1 - Re_{ij}) \times 0 = Re_{ij}.$$

Data privacy score

The privacy score ($PR$) of the data is a numerical representation of the overall privacy of the data, which is calculated. The privacy score generated for data $i$ with user $j$ is represented as $PR(i,j)$, and its calculation formula is

$$PR(i,j) = \gamma_i \times V(i,j)$$

where operator $\otimes$ represents any monotone incremental combination of functions about sensitivity and visibility. The details of the calculation process are described in equations (2) and (3).

Bilinear mapping

Let $(G_0, +)$ and $(G_1, \cdot)$ be the addition cycle group of $p$-order and the multiplication cycle group of $p$-order, where $p$ is a large prime number and $\alpha$ is a unit element of group $G_1$. $Z_p$ is the residual class integral ring of module $p$, and $Z_p^*$ is the set of invertible elements of $Z_p$.
defines bilinear mapping \( e : (G_0, G_0) \rightarrow G_1 \), and satisfies three properties:\textsuperscript{28}

- **Bilinear:** \( \forall P, Q \in G_0 \) and \( a, b \in \mathbb{Z}_p^* \), all have \( e(Pa, Qb) = e(P, Q)^{ab} \).
- **Computability:** \( \forall P, Q \in G_0 \), \( e(P, Q) \) is computable.
- **Non-degenerate:** \( \exists P, Q \in G_0 \) making \( e(P, Q) \neq \alpha \).

### Deduplication scheme

#### Overview

When a user uploads encrypted data to the CSP, the CSP can check whether the data have already been stored using the query label generated by elliptic curve technique and return a suggested PR to the user according to data information in the database if it exists. Eventually, the user uploads the data and its privacy score to the CSP, and the CSP reaggregates the privacy score and updates the threshold for it. Based on the threshold and the current number of data holders, the CSP decides whether to store it or not. The scheme consists of three parts: (1) privacy score query, (2) data uploading, and (3) privacy score calculation and threshold update, as shown in Figure 2.

#### Privacy score query

When a user uploads data to the CSP, the user can query the current PR of the upload data. The context-based privacy score query method is an optional solution.\textsuperscript{24} Context refers to a summary and a general idea generated from a long list of words or content, and even further, it is a representative keyword combination sorted out from the entire upload data, the general form is (field1 = value1, field2 = value2, ...). For example, if an upload data contains “Bob shared the final exam score in the class group with a desktop computer,” the context in the above example would be (sharer = Bob, subject = final exam score, observer = classmate).

In Harkous et al.,\textsuperscript{24} a scheme of data privacy query using context is introduced in detail. Before querying the degree of privacy, the user forms a query set of multiple virtual contexts and sends them to the CSP, to hide the actual request context. The context-based privacy score query only needs to replace the data privacy with the corresponding privacy score based on the above scheme. When the user queries the CSP for the PR of the data by means of the context, he should avoid sending the real context directly to the CSP, to prevent the CSP from associating the data sharing operation with the context, which reduces the risk of privacy data leakage.

Context-based privacy score query is easy to implement, but we usually assume that the CSP is honest but curious, and it can obtain specific data information uploaded by users through offline analysis and other operations. Therefore, we adopt the privacy score query mechanism of encrypted data. Based on the existing research results of our team, this mechanism uses the elliptic curve-based file label query scheme\textsuperscript{20} to facilitate the privacy score query. In Zhang et al.,\textsuperscript{20} a popularity query protocol without online trusted third parties is proposed. By constructing a bilinear map query label \( s_j = e(Y_1, H(M_j))^{X_1} \) (\( Y_1 \) is the encryption public key, \( X_1 \) and \( X_2 \) are the auxiliary keys, and \( X_3 = X_1 + X_2 \)), the data popularity check can be quickly completed using...
label comparison without data privacy disclosure. By associating the privacy score with the query label, we can use the query label comparison method to achieve a quick query of the privacy score and avoid the lack of privacy protection that may be caused by the context-based privacy score query method.

**Popularity threshold**

In order to improve the efficiency of deduplication, the CSP assigns a popularity threshold $T$ to the upload data. When the total number of upload users of a certain data $M$ is greater than the $T$, we consider $M$ to be a popular data. We use the more efficient convergence encryption and perform deduplication operations on it; otherwise, we consider $M$ to be nonpopular, which has a high degree of privacy. In that case, $M$ needs to be protected by semantically secure symmetric encryption.

**Data uploading procedure**

When the $U_j$ upload data $M_i$, $U_j$ first uploads the query label $s_i$ to the CSP. The CSP divides the file upload operation into three cases according to the relationship between the total number of users currently uploading the file $\text{count}(U_M)$ and the dynamic threshold $T_i$: $\text{count}(U_M) < T_i$, $\text{count}(U_M) = T_i$, and $\text{count}(U_M) > T_i$, as shown in Figure 3.

When $\text{count}(U_M) < T_i$, the CSP checks if the same data already exists. If it is the first uploaded copy, $U_j$ encrypts the data and uploads it to the CSP. The CSP stores the privacy score $PR_{ij}$, the encrypted encryption key $K_{M_i} = E(K_M, M_i)$, and the ciphertext $c_{K_i}$, as shown in part (a) of the figure; if it is not the first uploaded copy, the CSP returns the symmetric encryption key $K_{M_i}$ and the current $PR_i$ to $U_j$. $U_j$ decrypts and obtains $K_{M_i}$ and encrypts $M_i$ with it. Then, $U_j$ uploads the ciphertext $c_{K_i} = E(K_{M_i}, M_i)$ and the privacy score $PR_{ij}$ to the CSP. The CSP updates the $PR_i$ and deletes the newly upload data and create access links for $U_j$, as shown in part (b) of the figure. When $\text{count}(U_M) = T_i$, the CSP informs the $U_j$ to upload the convergence encryption ciphertext $c_{X_i} = E(X_i, M_i)$ and stores it, where $X_i = H(M_i) + X_2$. When $\text{count}(U_M) > T_i$, the CSP tells $U_j$ to perform client-side deduplication and creates an access link for $U_j$.

**Privacy score calculation**

In this section, we design the privacy score calculation method based on IRT, in order to ensure that the data’s $PR$ meets the requirements of most upload users.

Because different upload data $M_i$ are independent of each other, the parameters $\omega_i = (\phi_i, \gamma_i)$ for $PR$ can be calculated independently, so the calculation of the $PR$ can be performed in parallel. The IRT-based $PR$ calculation still needs to use formula (1) to estimate the
probability $\text{Re}_g = \text{prob}\{\chi(i,j) = 1\}$. In the cloud storage environment, we treat upload user $U_j$ as the tested user and upload data $M_i$ as the information item. Thus, the user’s propensity to privacy corresponds to the ability of himself: for $U_j$, we use the privacy preference parameter $\phi_j$ to evaluate the degree to which $U_j$ cares about privacy. The higher the $\phi_j$ value, the more extroverted or open $U_j$ is. If we assume that all the problems are easy to understand and have the similar degree of discrimination, then $\phi_j$ is no longer a variable to consider, and it can be replaced by a constant or simply ignored. Finally, we use the difficulty parameters $\gamma_i$ of the problem to represent the sensitivity of data, $\gamma_i \geq 0$.

For each parameter $\omega_i = (\phi_i, \gamma_i)$ of the upload data, we can estimate it by the maximum likelihood function, as shown in the formula (5)

$$
\prod_{j=1}^{N} \text{Re}_g^{\chi(i,j)} (1 - \text{Re}_g)^{1-\chi(i,j)}
$$

In other words, we need to search for the data parameters $\omega_i = (\phi_i, \gamma_i)$ that maximize the result using the likelihood function. $N$ is the total number of users uploading the same data, and $\chi(i,j)$ is the evaluation of the privacy tendency of the data $M_i$ by $U_j$, $\chi(i,j) \in (0, 1)$. Specifically, when user $U_j$ uploads $M_i$, the evaluation parameter $\chi(i,j)$ of the privacy tendency is uploaded together to evaluate the evaluation of the privacy level of the upload data. When the sensitivity $\gamma_i$ is sought, the value of $\chi(i,j)$ is used as the privacy tendency parameter $\phi_i$. The value is finally obtained as the sensitivity $\omega_i = (\phi_i, \gamma_i)$ based on the privacy tendency parameter $\phi_i$, being a known amount.

Similarly, on the basis of the sensitivity $\gamma_i$ is a known amount, the NR_Attitude_Estimation algorithm is used to search for the likelihood function or its corresponding log likelihood function, and the maximum privacy tendency parameter $\phi_i$ is found.

Finally, we integrate it through formula (6) and calculate the $PR$ that meets the public’s wishes:

$$
PR(i) = \sum_{j=1}^{n} \gamma_i \otimes V(i,j)
$$

At the same time, current $PR$ of the data is represented by $pr_i = PR(i)/n$, where $pr_i \in [0, 1]$. Finally, the CSP updates the data threshold with formula (7) according to the change of privacy score after each upload

$$
T_i = \frac{\alpha}{(1 - pr_i)^2} - (\alpha - 1)
$$

As the number of upload users increases, the threshold changes and gradually approaches $\lim_{i \to \infty} |T_{i+1} - T_i|$ $\leq \sigma$ ($T^\circ$ is a negligible value). Here, we propose the concept of ideal threshold $T^\circ$. $T^\circ$ is a threshold value per data that meets the user’s wishes. As the number of users increases, the dynamic threshold $T_i$ will gradually approach $T^\circ$, that is, $\lim_{i \to \infty} |T_i - T^\circ| \leq \sigma$. In summary, the IRT-based $PR$ calculation has the following advantages:

- Because different upload data are independent of each other, the CSP can compute the $PR$ in parallel, so the scheme is efficient and of practical feasibility.
- The parameters used in the privacy score calculation based on IRT are estimated by likelihood function which satisfies the property requirement of group invariance. This makes the $PR$ corresponding to different uploaded data directly comparable.

### Security analysis

The scheme is designed to better protect the security of private data through threshold dynamic adjustability. Our proposed deduplication scheme makes it impossible for the CSP to be spoofed. The data could be obtained only by obtaining the query label of the data. Here, we mainly discuss the authenticity and differentiability of the query labels. The security theorem is as follows.

**Lemma 1.** For a safe hash function $H : \{0, 1\}^* \rightarrow G_1$, the probability of $\forall D_1, D_2 \in \{0, 1\}^*, D_1 \neq D_2, H(D_1) = H(D_2)$ is negligible. That is, $P[H(D_1) = H(D_2)|D_1 \neq D_2] \ll \varepsilon$.

**Theorem 1.** Authenticity of data query labels. Let the initial upload user $U_0$ upload data $M$, then the query label of $M$ is $s_M = e(Y, H(M))^X$. When user $U_j$ uploads data $M'$, the query label of $M'$ is $s_{M'} = e(Y, H(M'))^X$. If and only if $s_M = s_{M'}$, $M = M'$ is established. That is, if $s_M = s_{M'}$, the probability of $M \neq M'$ is negligible.

**Proof:** If $M \neq M'$, we discuss the attacking of data query labels from the following two aspects:

1. Assuming that adversary $A$ is a malicious user, then $A$ can get parameter $X$.
2. Assuming that adversary $A$ is CSP, parameters $H(M)$ and $X$ are unpredictable for $A$.

In both cases, adversary $A$ cannot construct a query label satisfying equation $s_M = s_{M'}$. According to Lemma 1.
\[ F \neq F' \iff H(F) 
eq H(F') \iff e(Y, H(F)) \\
\neq e(Y, H(F')) \iff e(Y, H(F))^X \\
\neq e(Y, H(F'))^X \iff s_F \neq s_{F'} \]

That is, \( P[M \neq M'|e(Y, H(M))^X = e(Y, H(M'))^X] \leq \varepsilon \). Therefore, the query label of the data cannot be forged. As \( s_M = s_{M'} \), it can be deduced that \( M = M' \).

**Theorem 2. Differentiability of query labels.** Let the initial upload user \( U_0 \) of data \( M \) upload the query label as \( s_M = e(Y, H(M))^X \). When user \( U_j \) uploads \( M' \), the upload query label as \( s_{M'} = e(Y, H(M'))^X \). If \( M \neq M' \), the probability of \( s_F = s_{F'} \) can be ignored. That is, \( P[s_M = s_{M'}|M \neq M'] \leq \varepsilon \).

**Proof.** Suppose there is \( M \neq M' \) such that \( s_M = s_{M'} \). According to the properties of bilinear mappings, an equation can be obtained

\[ s_M = s_{M'} \iff e(Y, H(M))^X = e(Y, H(M'))^X \\
\iff e(Y, H(M)) = e(Y, H(M')) \\
\iff H(M) = H(M'). \]

If the above formula is valid, \( H(M) = H(M') \) must be valid. According to Lemma 1, \( M = M' \) can be obtained, which is contradictory to the hypothesis. Therefore, the hypothesis is not valid. That is, \( P[s_M = s_{M'}|M \neq M'] \leq \varepsilon \). It can be seen that when and only when \( s_M = s_{M'} \), \( M = M' \) holds. That is to say, the query labels of data are distinguishable.

**Lemma 2.** Compute Diffie–Hellman (CDH) problem. Suppose \( (G_0, \cdot) \) is a multiplicative cyclic group of order \( P \), and the unit element is denoted as \( g \). For a given \( g, g^x, h \in G_0 \), it is difficult to calculate \( Q = h^x \in G_0 \), where \( a \in Z_n^* \).

**Theorem 3. Security of data labels.** CSP cannot attack the query label offline and get any plaintext information.

**Proof.** Let CSP execute an offline brute force attack on the query label \( s_M = e(Y, H(M))^X \) of data. CSP exhausts a large amount of data \( \{M_i\} \), trying to find data \( M' \) that satisfies \( s_M = s_{M'} \). CSP can calculate \( e(Y, H(M_j)) \), but because it is not an authorized user, it is impossible to obtain the security parameter \( X \) from the broadcasting center. Lemma 2 shows that it is still difficult to calculate \( e(Y, H(M_j))^X \) even if \( e(Y, H(M_j))^X \) and \( e(Y, H(M_j)) \) are known. Therefore, CSP cannot obtain explicit data information from query labels by brute force attack.

In addition, we also considered the situation of malicious scoring. We made simulations, and detailed results can be found in the next section.

**Performance evaluation and experiments**

The experiment uses PBC,\(^{30}\) GMP,\(^{31}\) PBC_bce,\(^{32}\) and OPENSSL\(^{33}\) function libraries, which are implemented by C + + language. It is deployed to a Tencent’s cloud storage server, which is equipped with 4 GB memory, 4-core CPU, 1 Mbps bandwidth, and 1 TB storage. In order to make it easier for users to understand and to operate, our scheme adopts a more user-friendly design in the implementation of PR. When the upload user scores some data for the PR, it is only necessary to choose a value between 1 and 100 as the score. The system automatically converts the data and updates the PR and \( T \). In accordance with the percentile scoring habit, users can understand the privacy of upload data more intuitively. Considering the difficulty of sample selection, we generate random numbers to simulate the \( PR \) of different users on a certain upload data.

**Data set**

In view of the problem that different data have different ideal thresholds, we carry out a comparative experiment on the overall \( PR \) and \( T \) with various of data. We use three sets of random numbers to simulate the \( PR \) of different data. Each data set consists of 100 random numbers. The first data set consists of random numbers from 80 to 100, which simulates the user’s \( PR \) for data with a high privacy level. The second data set consists of random numbers from 0 to 20, which simulates the user’s privacy score of certain data with a low degree of privacy. The third data set is composed of random numbers from 1 to 100, which simulates the random distribution of upload users’ privacy scores on given data.

In the performance comparison experiment, we choose 1000 files of 10 MB as the upload data, in which the ratio of data with lower privacy to data with higher privacy is about 3/2. Other schemes adopt the unified popularity threshold and set it to \( T = 7 \).

**Experimental analysis**

The data of the above three groups of experimental data sets are simulated by uploading and dynamic adjusting threshold, respectively, and the changes of the whole privacy score and \( T \) value of the data are compared and analyzed.

Figures 4–6 are derived from the data set with the interval of (80–100), where Figure 4 shows the change of the \( PR \) with the number of upload users. The curve
in Figure 4 is connected by 100 data points. The ordinate of each point is the result of the PR adjustment according to the feedback of the user. The horizontal coordinate is the number of users currently uploading data, and Figure 5 shows the relationship between the dynamic adjustment value of T and the PR. The curve in the figure shows how T of the data changes with the PR. The meaning of the ordinate of each point in the figure is the same as that of Figure 4, and the abscissa is the dynamic threshold of the data calculated according to the privacy fraction. The dynamic threshold T of the data is calculated as in equation (7). T is proportional to the PR of the data, where α value can be adjusted as needed. In this experiment, \(\alpha = 7\); Figure 6 is the result of overlapping of Figures 4 and 5 in the same coordinate system. The minimum abscissa value in all intersections is the T for actual deduplication. However, there is no intersection in Figure 6, which means there is no deduplication operation. Similarly, Figures 7–9 are derived from the data set with the interval (0–20). In this simulated scenario, the actual deduplication threshold is \(T = 10\). Figures 10–12 are derived from the data set with the interval (1–100), and the value of T at the actual deduplication is 31.

In Figure 4, all users are uploading data with low degree of privacy, but there are still small differences in the specific case in the value of numerical PR. We assume that each user will eventually choose a number from (80–100) as its PR. When the number of upload users is small, the PR chosen by the user has a greater impact on the overall PR. As the number of upload users increases, the impact of the PR chosen by a single user on the overall PR decreases. Finally, the PR is stable at about 90. Similarly, Figure 7 represents a situation where the overall PR is high, and it is stable at about 11. The curve in Figure 10 represents the PR of all users for some data which are not unified. From the curve in the figure, it can be seen that under the premise of inconsistent user opinions, when the number of upload users is small, the privacy score adjustment fluctuates greatly. As the number of upload users increases, the impact gets smaller, and the overall PR of the data will eventually be stabilized.

In Figures 4, 7, and 10, when the number of samples is large enough, T tends to stabilize as the number of
users increases. This value is only related to the nature of the data and the user’s concern for privacy. The results in Figure 10 further illustrate that the final $PR$ of data is determined by the user’s attitude. In the case of a large user population, the attitude of individual users (reflected by the score) has little impact on it. As can be seen from Figure 6, when the privacy level of data information is high, there is no intersection point between the two curves, that is, the data are not subjected to the deduplication operation, which protects the security of private data more effectively. According to Figure 9, when the data privacy level is very low, the threshold for actually deduplication is very small, which can effectively reserve cloud storage space. From Figures 10–12, it can be seen that when some upload user has a different attitude toward the privacy of certain data, the system will determine an appropriate threshold for the data according to the attitude of the majority. Therefore, on the whole, the scheme is feasible and applicable.

**Anti-abuse experiment**

The anti-abuse experiment mainly tests the anti-abuse capability of the scheme from the aspect of malicious scoring. We assume that a malicious user deliberately
sets the privacy score to 100 when the user privacy score is generally low (0–20) or sets the privacy score to 0 when the user privacy score is generally high (80–100). We tested the impact of malicious user scoring in three different settings, in which there are 2%, 3%, and 4% malicious users, respectively. The experimental results are shown in Figures 13 and 14. Figure 13 reflects the situation where some malicious user intentionally chooses a larger privacy score when uploading data with a lower privacy score. Figure 14 reflects the situation where some malicious user intentionally chooses a smaller privacy score when uploading data with a higher privacy score. To facilitate better observation, we assume that malicious users send their own malicious ratings at the beginning. As can be seen from the Figures 13 and 14, the higher the proportion of malicious users is, the greater the impact on the privacy score is. The greater the population of upload users is, the minor affect the malicious score causes. When the number of upload user exceeds 70, the four curves in the figure have little difference, which indicates that the scheme has anti-abuse capability.

Performance comparison

By uploading 1000 files of 10 MB, the total time consumption of our scheme is calculated and compared
with that of other schemes, namely the PerfectDedup scheme, the common popularity threshold-based deduplication scheme, and the Xu-CDE scheme. The experiment is repeated for 10 times and the average result is acquired as the final result, which is shown in Figure 15. In the data encryption phase, the time consumption of the four schemes is similar. In the query stage, our scheme has advantages over other schemes that recognize data popularity. Finally, compared with other schemes, our design does not cause additional time overhead while improving the security of the deduplication operation.

Conclusion

In this article, we address the issue of deduplication threshold in the cloud storage scenario and propose a secure deduplication scheme for IoT sensor networks based on threshold dynamic adjustment. The concept of the ideal threshold is proposed for the first time, and the IRT is applied. By uploading the user’s feedback on the data privacy level, the privacy score can be dynamically adjusted, thereby calculating and adjusting the threshold of the deduplication. This scheme can speed up the deduplication stage for data with lower privacy, while data with a higher extent of privacy can be better protected. Experiments show that our scheme not only improves the security of deduplication operation but also avoids additional time overhead. Compared with other schemes, our scheme is more practical for real-world applications.

How to improve the efficiency of data deduplication for IoT applications while ensuring data security will be studied in future works.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the National Natural Science Foundation of China (61702294), the Shandong Provincial Natural Science Foundation (ZR2019MF058), and the Open Project Program of The State Key Laboratory of Integrated Services Networks (ISN19-14).

ORCID iD

Hequn Xian https://orcid.org/0000-0002-7538-338X

References

1. Stergiou C, Psannis KE, Kim B-G, et al. Secure integration of IoT and cloud computing. Future Gener Comp Sy 2018; 78: 964–975.
2. Stanek J and Kencl L. Enhanced secure thresholded data deduplication scheme for cloud storage. IEEE T Depend Secure 2018; 15: 694–707.
3. Zheng X, Zhou Y, Ye Y, et al. A cloud data deduplication scheme based on certificateless proxy re-encryption. J Syst Archit 2020; 102: 101666.
4. Yuan J and Yu S. Secure and constant cost public cloud storage auditing with deduplication. In: Proceedings of the 2013 IEEE conference on communications and network security (CNS), National Harbor, MD, 14–16 October 2013, pp.145–153. New York: IEEE.
5. Wallace GR, Shilane PN, Huang ML, et al. Deduplication using sub-chunk fingerprints. Patent US10135462B1, USA, 2018.
6. Fu YJ, Xiao N and Liu F. Research and development on key techniques of data deduplication. J Comput Res Dev 2012; 49(1): 12–20. (in Chinese with English abstract)
7. Daniel E and Vasanthi NA. LDAP: a lightweight deduplication and auditing protocol for secure data storage in cloud environment. Cluster Comput 2019; 22(1): 1247–1258.
8. Cheung CHI, Oltean PA and R Kalach. Scalable chunk store for data deduplication. Patent 10394757, USA, 2019.
9. Li X, Lin J, Li J, et al. A video deduplication scheme with privacy preservation in IoT. In: International symposium on computational intelligence and intelligent systems, 2015, pp.409–417. Singapore: Springer.
10. Harnik D, Pinkas B and Shulman-Peleg A. Side channels in cloud services: deduplication in cloud storage. IEEE Secur Priv 2010; 8: 40–47.
11. Bolosky W, Douceur J and Theimer M. Encryption systems and methods for identifying and coalescing identical objects encrypted with different keys. US20040215962A1, USA, 2004.
12. Xu J, Chang EC and Zhou J. Weak leakage-resilient client-side deduplication of encrypted data in cloud storage.
In: Proceedings of the 8th ACM SIGSAC symposium on information, computer and communications security, Hangzhou, China, 7–10 May 2013, pp.195–206. New York: ACM.

13. Tang H, Cui Y, Guan C, et al. Enabling ciphertext deduplication for secure cloud storage and access control. In: Proceedings of the 11th ACM on Asia conference on computer and communications security, Xi’an, China, 30 May–3 June 2016, pp.59–70. New York: ACM.

14. Yang C, Ji Q, Xiong SC, et al. New method for file deduplication in cloud storage. J Commun 2017; 38: 25–33. (in Chinese with English abstract)

15. Bellare M, Keelveedhi S and Ristenpart T. Message-locked encryption and secure deduplication. In: Proceedings of the annual international conference on the theory and applications of cryptographic techniques, Athens, 26–30 May 2013. Berlin; Heidelberg: Springer.

16. Hou HY, Yu J and Hao R. Cloud storage auditing with deduplication supporting different security levels according to data popularity. J Netw Comput Appl 2019; 134: 26–39.

17. Stanek J, Sorniotti A, Androulaki E, et al. A secure data deduplication scheme for cloud storage. In: Christin N and Safavi-Naini R (eds) Financial cryptography and data security (FC 2014): lecture notes in computer science, vol 8437. Berlin; Heidelberg: Springer, 2014, pp.99–118.

18. Puzio P, Molva R, Önen M, et al. PerfectDedup: secure data deduplication. In: Data privacy management international workshop on quantitative aspects in security assurance, Heraklion, 26–27 September 2016. Berlin: Springer.

19. Liu J, Asokan N and Pinkas B. Secure deduplication of encrypted data without additional independent servers. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, Denver, CO, 12–16 October 2015, pp.874–885. New York: ACM.

20. Zhang SG, Xian HQ, Wang YZ, et al. Secure encrypted data deduplication method based on offline key distribution. J Softw 2018; 29: 1909–1921. (in Chinese with English abstract)

21. Singh P, Agarwal N and Raman B. Secure data deduplication using secret sharing schemes over cloud. Future Gener Comp Sy 2018; 88: 156–167.

22. Lin XH. Privacy protection in community-based networks. PhD Thesis, Shanghai Jiao Tong University, Shanghai, China, 2010. (in Chinese with English abstract)

23. DeMars C. Item response theory. New York; Oxford: Oxford University Press, 2010.

24. Harkous H, Rahman R and Aberer K. C3P: context-aware crowdsourced cloud privacy. In: Proceedings of the 14th privacy enhancing technologies symposium (PETS 2014), Amsterdam, 16–18 July 2014. Berlin: Springer.

25. Rasch G. Studies in mathematical psychology: I—probabilistic models for some intelligence and attainment tests. Copenhagen: Nielsen & Lydiche, 1960.

26. Wright BD. Solving measurement problems with the Rasch model. J Educ Meas 1977; 14: 97–116.

27. Wang DW, Liau CJ and Hsu T-S. Privacy protection in social network data disclosure based on granular computing. In: Proceedings of the 2006 IEEE international conference on fuzzy systems, Vancouver, BC, 16–21 July 2006, pp.997–1003. New York: IEEE.

28. Miller VS. The Weil pairing, and its efficient calculation. J Cryptol 2004; 17: 235–261.

29. Lord Frederic M. A theory of test scores, 1952, https://www.psychometricsociety.org/sites/default/files/pdf/ MN07.pdf

30. Lynn B. The pairing-based cryptography library, http://crypto.Stanford.edu/pbc/ (2015, accessed 12 October 2017).

31. Loukides M and Oram A. Programming with GNU software, vol. 86. Beijing, China: O’Reilly & Associates, 1997, pp.350–359.

32. Steiner M. The PBC_bce broadcast encryption library, 2006, https://crypto.stanford.edu/pbc/bce/ (2006, accessed 12 October 2016).

33. Hu XT, Qin ZP, Zhang H, et al. Research and improved implementation of AES algorithm in OpenSSL. Control Autom 2009; 25: 83–85.