A privacy protection approach for XML-based archives management in a cloud environment

Zongda Wu
Department of Computer Science and Engineering, Shaoxing University, Shaoxing, China and School of Information Management, Nanjing University, Nanjing, China

Jian Xie
Shaoxing University, Shaoxing, China

Xinze Lian
Oujiang College, Wenzhou University, Wenzhou, China, and

Jun Pan
Zhejiang University of Science and Technology, Hangzhou, China

Received 26 May 2019
Revised 22 July 2019
Accepted 30 August 2019

Abstract

Purpose – The security of archival privacy data in the cloud has become the main obstacle to the application of cloud computing in archives management. To this end, aiming at XML archives, this paper aims to present a privacy protection approach that can ensure the security of privacy data in the untrusted cloud, without compromising the system availability.

Design/methodology/approach – The basic idea of the approach is as follows. First, the privacy data before being submitted to the cloud should be strictly encrypted on a trusted client to ensure the security. Then, to query the encrypted data efficiently, the approach constructs some key feature data for the encrypted data, so that each XML query defined on the privacy data can be executed correctly in the cloud.

Findings – Finally, both theoretical analysis and experimental evaluation demonstrate the overall performance of the approach in terms of security, efficiency and accuracy.

Originality/value – This paper presents a valuable study attempting to protect privacy for the management of XML archives in a cloud environment, so it has a positive significance to promote the application of cloud computing in a digital archive system.

Keywords Archives management, Digital archives, Privacy protection, Cloud computing

Paper type Research paper

1. Introduction

Cloud computing enables people to obtain their required resources – such as network, server, storage, application and service – from a shared pool of configurable computing resources anytime, anywhere, and on demand (Peng *et al.*, 2016). It can minimize the overhead of resource management, and thus reduce the institutional investment in business operation and document management (Yoo and Kim, 2018). In recent years, government sectors in developed countries, such as the USA, the UK, and Australia, have launched the *Cloud First* strategy (Chen, 2017),

This study is supported by the National Social Science Foundation of China (No. 19BTQ056).

resulting in an increasing proportion of formation and management of archives in cloud environments. Now, the management of archives based on cloud computing has become a major trend (Gao and Huang, 2018). However, cloud computing also results in many negative effects (Gao and Huang, 2018; Wu *et al.*, 2018a), with the most important being that the archive data needs to be stored and managed by the cloud server, taking the archives out of the control of the owners, resulting in a serious threat to the security of privacy of the data in the archives. In general, the privacy threat includes two parts:

(1) an external threat, that is, attacks from hackers to cloud service providers (Attasena *et al.*, 2017); and

(2) an internal threat, that is, threats from the cloud service provider's staff (Wu *et al.*, 2018a).

In short, for an archives management system, the cloud server is untrusted, thus the security of the archives in the cloud has become an important obstacle to the further application of cloud computing technologies in the management of digital archives. Therefore, for the promotion of archives management based on cloud computing, how to ensure the security of archives data stored in the untrusted cloud server has become an important problem which needs to be solved urgently.

XML is a famous extensible markup language, which has become an important data exchange standard (Calvanese *et al.*, 2018). At present, in archives management systems, a large number of digital archives are organized and managed using XML (Post *et al.*, 2014). Thus, in this paper, a privacy protection approach is proposed for XML-based archives management in a cloud environment. The basic idea of the proposed approach is that the privacy data has to be strictly encrypted on a trusted client, before they are submitted to the cloud-side for storage, so that the untrusted cloud cannot infer the privacy data from them, to ensure the security of privacy data. Then, to improve the efficiency of each query defined over the privacy data, the query problem on the encrypted privacy data is studied. The basic idea is to append feature data for the encrypted data, so that most of the processing for each query defined on the privacy data can be performed by the cloud server-side, without the need to decrypt the data, thus greatly improving the query efficiency. Specifically, the contributions of this paper are threefold:

- Under the existing framework of a cloud-based archive management system, a system model with privacy protection is redesigned, where the privacy protection approach runs on the trusted client as a layer of middleware between the cloud and the archives management interface, resulting in a good integration with the existing archives system.
- A feature construction approach for XML privacy data is designed. The constructed feature data not only has good security (the attackers cannot infer the privacy data from the feature data), but also has good performance (it can support range queries and containing queries).
- A transformation approach for XML queries is designed, which can transform each query on the privacy data to a new query on the feature data, so that the new query can be executed correctly on the cloud server, thereby improving the query efficiency.

2. Problem statement

2.1 Literature review

Aiming at the privacy problem of cloud computing, the scholars from social sciences tried to solve the problem from the perspective of laws and regulations (Xiang and Cheng, 2018).

Major developed countries formulated related standards and specifications; for example, the *Document Management Guidelines in Cloud Computing Environment* issued by the US government and the *Risk Management Suggestions for Document Preservation in Cloud Computing* from the Australian document and archives committee. However, an endless stream of privacy leakage incidents showed that laws and regulations cannot fundamentally solve the privacy problem in the cloud environment, so the solution cannot depart from the support of technical means (Wu *et al.*, 2015, 2018b). To ensure data security, archives systems have used a variety of technical methods (Lu *et al.*, 2013), mainly including: identity authentication, access control, and data encryption. Below, the technical characteristics for the three kinds of methods are first introduced, and then their application limitations are analyzed in the privacy protection of cloud-based archives management.

User identity authentication refers to a process of user identity confirmation, which aims to eliminate illegal access to system resources by illegal users (Yuan *et al.*, 2014). User identity authentication can be divided into single-factor authentication, such as user name and password authentication, smartcard authentication, and dynamic password authentication (Wang *et al.*, 2017), and two-factor authentication that combines two single-factor methods together (Chandrasekhar and Singhal, 2017). *Access control* refers to restricting user access to unauthorized system resources, according to user identity (Chandrasekhar *et al.*, 2017), specifically including discretionary access control (Servos and Osborn, 2017) and mandatory access control (Qi *et al.*, 2017). Identity authentication and access control have been widely used in operating systems, database systems, and archives management systems (Power *et al.*, 2018). However, for the two kinds of technical methods, their implementations cannot depart from the support of server-sides (they are developed based on an assumption that the server-side is trusted). They are only targeted for external attackers of an archives system, and thus cannot prevent internal staff on the server-side or hackers breaking down server-side from accessing the privacy data in digital archives (Singh and Chatterjee, 2017). As mentioned previously, the cloud server-side is untrusted, thus the privacy problem in cloud archive management cannot be solved by traditional means of access control and identity authentication.

Data encryption refers to strictly encrypting privacy data in a trusted client, so as to make the data encrypted even if being revealed makes it difficult to understand (Ding *et al.*, 2017). It is an important means to solve the privacy problem in a cloud environment (Wei *et al.*, 2019). However, in a digital archive management system, there are a large number of query operations on archives privacy data (e.g. obtaining user profile information according to user names). Once an encryption algorithm is adopted to encrypt the privacy data stored in the cloud, the encrypted data would lose many inherent characteristics of the privacy data, such as orderliness and comparability (Mei *et al.*, 2018), making all kinds of query operations, such as text containing queries and range queries, defined on the privacy data no longer able to be executed correctly on the encrypted data. Thus, since a traditional encryption method would compromise the system availability, it cannot be used to solve privacy issues in cloud archive management.

In addition, there are still a number of other methods (Lu, 2019; Zhao, 2018), which can be used for the privacy protection of an archive management system. In general, all the existing methods mentioned above cannot meet the requirements of data security and system availability simultaneously. In summary, in a cloud archive management system, how to ensure the security of archival privacy data within the untrusted cloud without compromising the system availability remains for further in-depth study.

2.2 System framework

In general, a cloud-based archives management system consists of a cloud server-side (for the storage and management of digital archives) and a group of user interfaces on client-sides. The overall framework of the privacy protection approach used in this paper for cloud archives management is shown in Figure 1, whose processing flow can be described as follows.

- For any privacy data issued by a user via a client interface (Step 1), the feature construction component running on the client would carry out encryption and feature construction to obtain the encrypted data and the feature data (Step 2), and submit them (instead of the privacy data) to the cloud server-side, to improve the security of privacy data in the cloud.
- For any XML query defined on the privacy data issued by a user (Step 3), since it cannot be executed on the encrypted data, the query transformation component would transform it to a new query on the feature data (Step 4), and then submit the new query to the cloud, to filter out most of the non-target data, thus ensuring the query efficiency.
- For the intermediate query result returned from the cloud (Step 5), the decryption filtering component would decrypt the ciphertext data, and perform the original query (the query from Step 3) on the decrypted data, to further eliminate the non-target data, and return the accurate result to the user, thereby ensuring the query accuracy.

It can be seen that the framework is somewhat similar to that of a traditional data encryption method, whose basic idea is to encrypt privacy data in a trusted client to improve the security of privacy data in the untrusted cloud. However, in the proposed framework, the feature data are newly introduced, which aims to ensure the availability of an archives system in terms of accuracy and efficiency. Therefore, the feature construction plays a very important role in the framework, and an ideal approach to feature construction should meet the following three requirements:

(1) *Privacy*. Since the feature data (Step 2) is completely visible to the untrusted cloud server, it has to ensure its own privacy; that is, it should be difficult to infer the privacy data according to the analysis on the feature data. In addition, since the encrypted data (Step 2) is from a traditional encryption algorithm (e.g., AES), the security of encrypted data in the cloud can be fully guaranteed, so attention need no longer be paid to the encrypted data.

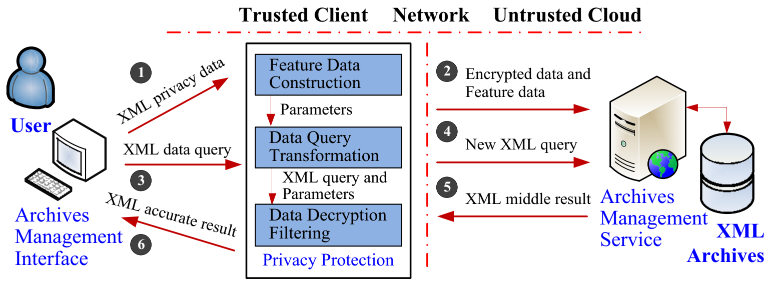

Figure 1.
The system framework with privacy protection for cloud archives management

(2) *Accuracy*. That is, the feature data should be able to help users to obtain accurate query results (Steps 3-6). It is a key requirement to ensure system availability. Specifically, with the help of the feature data, any query (including containing queries and range queries) defined on privacy data should be able to be transformed to a new query (Step 5) on the feature data, such that the new query can be executed correctly by the cloud server-side, and the returned result can contain the accurate result, so as to ensure the query accuracy.

(3) *Efficiency*. That is, the privacy protection should not compromise the efficiency of each query (Steps 3-6). It is another key requirement to ensure system availability. Specifically, with the help of the feature data, each new query (Step 5) generated by the transformation component should be able to filter out the non-target data as much as possible in the cloud, so as to make the intermediate result (Step 5) returned to the client as close as possible to the accurate result, and in turn improve the query efficiency.

3. Method

First, a brief discussion of how the encrypted data and feature data generated for the privacy data are stored in the XML archives on the cloud. To simplify discussion, it is assumed that there is a kind of element (whose name is *element*, and path is "/path/element") in the archives, whose contents are used to store some kind of privacy data (whose type is character string). To store the corresponding encrypted data and feature data, the archive schema must be modified to insert three sub-elements (named *ciphtext*, *conttext*, and *rangtext*, respectively) for the *element*, which are used to store the encrypted data ("//element/ciphtext"), containing feature data ("//element/conttext"), and range feature data ("//element/rangtext"), respectively. Figure 2 illustrates an XML schema of employee archives after modification (where address and phone are private, so there are two kinds of privacy data in the archives). In the rest of this paper, all the examples will be built on top of the schema.

From Figures 2 and 1, it is concluded that for any employee record issued by a client user, the feature construction component would strictly encrypt the privacy data, and store them into "//address/ciphtext" and "//phone/ciphtext". Meanwhile, the component would construct the containing feature data (used to support containing queries) and range feature data (used to support range queries) for the privacy data, and then store them into "//conttext" and "//rangtext". Here, the data encryption is completed by using a traditional encryption algorithm (not described in this paper). Below, first discussed is how to construct

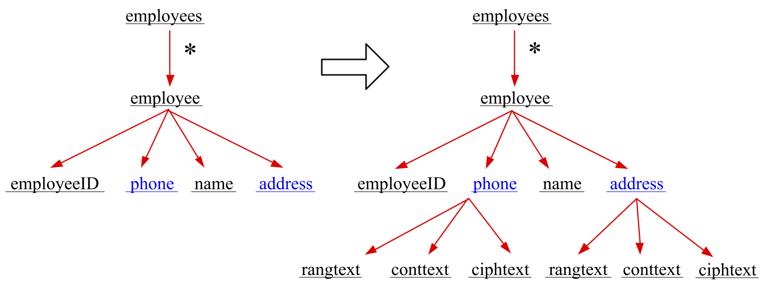

Figure 2.
An example of the modification of the XML schema of employees' archives

the containing feature data and range feature data for the privacy data, and then how to transform each query defined on the privacy data to a new query defined on the feature data.

3.1 Feature construction

First, this section introduces the construction of containing feature data using Definitions 3.1 and 3.2. The introduction of containing feature data aims to implement familiar XML containing queries (e.g. querying the employees whose addresses contain "Zhejiang Province"). A containing query is developed on a containing operation between two character strings.

Definition 3.1 (Character-pair Mapping): Let $\mathbb{A}$ denote the value domain of all the characters. Then, for any two characters a_1 and a_2 defined on $\mathbb{A}$, any character-pair mapping function can be expressed as $\mathbf{P}(a_1, a_2)$: $\mathbb{A} \times \mathbb{A} \rightarrow \mathbb{A}$ i.e. $\mathbf{P}(a_1, a_2) \in \mathbb{A}$.

In Definition 3.1, only an open function is described. According to the Unicode regulation, a character consists of two bytes (whose encoding value is between 0 and $2^{16} - 1$), so $|\mathbb{A}| = 2^{16}$. Thus, a specific character-pair mapping function can be constructed using the following strategy.

Step 1. Divide $\mathbb{A}$ into 2^8 subdomains $\mathbb{A}_1, \mathbb{A}_2, \ldots \mathbb{A}_{256}$, such that: (1) any subdomain is not empty (i.e. $\mathbb{A}_i \neq \oslash$), (2) any two subdomains do not overlap (i.e. $\mathbb{A}_i \cap \mathbb{A}_k = \oslash$), and (3) the union of all the subdomains is equal to $\mathbb{A}$ (i.e. $\mathbb{A}_1 \cup \mathbb{A}_2 \cup \ldots \cup \mathbb{A}_{256} = \mathbb{A}$).

Step 2. Assign integer identifiers for $\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_{256}$, denoted by $\varpi(\mathbb{A}_1)$, $\varpi(\mathbb{A}_2)$, ..., $\varpi(\mathbb{A}_{256})$, such that: (1) $0 \le \varpi(\mathbb{A}_i) \le 2^8 - 1$ and (2) $\varpi(\mathbb{A}_i) \neq \varpi(\mathbb{A}_k)$ $(i \neq k)$.

Step 3. Now, any character a defined on $\mathbb{A}$ can be mapped to a unique integer $\varpi(\mathbb{A}_k)$, where $\mathbb{A}_k$ is the subdomain containing a (i.e. $a \in \mathbb{A}_k$, denoted by $\mathbf{P}(a) = \varpi(\mathbb{A}_k)$ $(a \in \mathbb{A}_k)$).

Finally, a character-pair mapping function that meets Definition 3.1 can be constructed as follows: $\mathbf{P}(a_1, a_2) = \mathbf{P}(a_1) \cdot 2^8 + (a_2)$.

Here, $\mathbf{P}(a_1)$ and $\mathbf{P}(a_2)$ are both between 0 and $2^8 - 1$ (each takes up one byte), and the joining of them corresponds to a new character $\mathbf{P}(a_1, a_2)$ as the mapping result of a_1 and a_2. It can be seen that the function generated by the mapping strategy is not unique (since there are multiple settings in Steps 1 and 2), thus the construction of a character-pair mapping is also open.

Definition 3.2 (Containing Feature Mapping): Let $\mathbf{P}$: $\mathbb{A} \times \mathbb{A} \rightarrow \mathbb{A}$ represent a specific character-pair mapping function. For any string $a_1a_2 \ldots a_n$ defined on the string domain $\mathbb{A}^*$, a containing feature mapping function can be expressed as $\mathbf{P}(a_1a_2 \ldots a_n)$: $\mathbb{A}^* \rightarrow \mathbb{A}^*$, whose value is also a string on $\mathbb{A}^*$, denoted by $\mathbf{P}(a_1a_2 \ldots a_n) = a'_1a'_2 \ldots a'_{n'}$, which should meet that: (1) $1 \le n' \le n - 1$; (2) $a'_1 < a'_2 < \ldots < a'_{n'}$; (3) for any $a_i \in a_1a_2 \ldots a_n$, there should exist an unique $a'_k \in a'_1a'_2 \ldots a'_{n'}$, such that $\mathbf{P}(a_i, a_{i+1}) = a'_k$; and (4) for any $a'_k \in a'_1a'_2 \ldots a'_{n'}$, there should exist $a_i \in a_1a_2 \ldots a_n$ (it may not be unique), such that $\mathbf{P}(a_i, a_{i+1}) = a'_k$.

Similarly, in Definition 3.2, only an open function is also described. Thus, a specific function can be constructed that meets the constraints of Definition 3.2 using the following strategy:

Step 1. For any character string $a_1a_2 \ldots a_n$, based on a given specific character-pair mapping function $\mathbf{P}$: $\mathbb{A} \times \mathbb{A} \rightarrow \mathbb{A}$, map all the character pairs contained in the string to a series of characters, thereby obtaining a character set, denoted by $\mathbb{A}' = \{\mathbf{P}(a_k, a_{k+1}) \mid a_k \in a_1a_2 \ldots a_{n-1}\}$.

Step 2. After removing the duplicate characters in $\mathbb{A}'$, sort all the characters according to their encoding values in an ascending order, obtaining a new character string $\mathbf{P}(a_1a_2 \ldots a_n)$ (i.e. the containing feature data of $a_1a_2 \ldots a_n$).

It can be seen that unlike the previous character-pair mapping strategy, as long as the mapping function $\mathbf{P}$: $\mathbb{A} \times \mathbb{A} \rightarrow \mathbb{A}$ is specifically presented, the feature mapping function $\mathbf{P}$: $\mathbb{A}^* \rightarrow \mathbb{A}^*$ obtained according to the above mapping strategy is uniquely determined.

Example 3.1: Assume (1) Step 1 uses an equal-width continuity method to divide $\mathbb{A}$, where equal-width refers to $|\mathbb{A}_i| = 2^8$, and the continuity means that the characters in the same subdomain are continuous (the character encoding values in $\mathbb{A}_1$ are between 0 and $2^8 - 1$; those in $\mathbb{A}_2$ are between 2^8 and $2^9 - 1$; and so on); and (2) in Step 2, the identifier of each $\mathbb{A}_k$ is set to $\varpi(\mathbb{A}_k) = k - 1$. Now, a containing mapping function is uniquely determined. Figure 3 illustrates the containing feature construction for the Chinese address "浙江温州中路3号".

Next, the construction of range feature data is introduced. The introduction of range feature data aims to implement familiar XML range queries (e.g. querying the information of employees whose phone number is within the range of 13512340000 to 13512349999). It can be seen that a range query is developed on a comparison operation between two character strings.

Definition 3.3 (Range Feature Mapping): Let $\mathbb{A}^*$ denote the domain of the character strings. Then, for any character string $a_1a_2 \ldots a_n$ defined on $\mathbb{A}^*$, a range feature mapping function can be expressed as $\mathbf{R}(a_1a_2 \ldots a_n)\mathbb{A}^* \rightarrow \mathbb{A}^*$, whose value is also a character string defined on $\mathbb{A}^*$. It should meet the requirement that: (1) $1 \leq |\mathbf{R}(a_1a_2 \ldots a_n)| \leq n$; and (2) for any string $a'_1a'_2 \ldots a'_{n'}$ defined on $\mathbb{A}^*$, if $a_1a_2 \ldots a_n \leq a'_1a'_2 \ldots a'_{n'}$, then we have $\mathbf{R}(a_1a_2 \ldots a_n) \leq \mathbf{R}\left(a'_1a'_2 \ldots a'_{n'}\right)$.

In Definition 3.3, only an open feature function is described. Below is presented a strategy to construct a specific range mapping function that meets the constraints of Definition 3.3:

Step 1. Divide $\mathbb{A}^*$ into N subdomains $\mathbb{A}_1^*, \mathbb{A}_2^*, \ldots, \mathbb{A}_N^*$, such that any subdomain is not empty (i.e. $\mathbb{A}_i^* \neq \oslash$), any two subdomains do not overlap (i.e. $\mathbb{A}_i^* \cap \mathbb{A}_k^* = \oslash$), the union of all the subdomains is equal to the domain itself (i.e. $\mathbb{A}_1^* \cup \mathbb{A}_2^* \cup \ldots \cup \mathbb{A}_N^* = \mathbb{A}^*$) and each element is well-ordered (i.e. for $\forall \alpha_1^* \in \mathbb{A}_i^* \forall \alpha_2^* \in \mathbb{A}_k^*$, if $i < k$, then $\varpi(\mathbb{A}_i^*) < \varpi(\mathbb{A}_k^*)$).

Step 2. Assign identifiers for $\mathbb{A}_1^*, \mathbb{A}_2^*, \ldots, \mathbb{A}_N^*$, denoted by $\varpi(\mathbb{A}_1^*), \varpi(\mathbb{A}_2^*), \ldots, \varpi(\mathbb{A}_N^*)$, such that (1) each identifier is a character string (i.e. $\varpi(\mathbb{A}_k^*) \in \mathbb{A}^*$); and (2) all the identifiers are well ordered (i.e. if $i < k$, we have that $\varpi(\mathbb{A}_i^*) < \varpi(\mathbb{A}_k^*)$).

Step 3. Now, any character string α^* defined on $\mathbb{A}^*$ can be mapped to an unique identifier $\varpi(\mathbb{A}_k^*)$, where $\mathbb{A}_k^*$ is the subdomain containing α^*. It is denoted by $\mathbf{R}(\alpha^*) = \varpi(\mathbb{A}_k) \quad (\alpha^* \in \mathbb{A}_k^*)$.

Example 3.2: Assume (1) the privacy data is from a phone number that consists of 11 digital characters; (2) in Step 1, $\mathbb{A}^*$ is divided to 102 subdomains, whose details are shown as

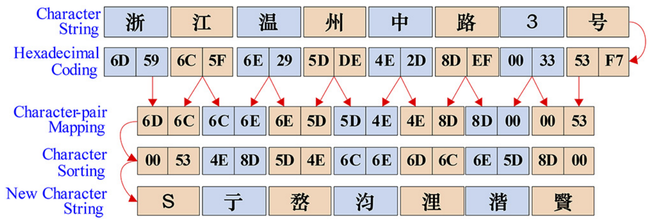

Figure 3.
An example of the construction for containing feature data

Figure 4 and 3 in Step 2, $\varpi(\mathbb{A}_k^*)$ of $\mathbb{A}_k^*$ is set to the character corresponding to the integer number k (i.e. $\varpi(\mathbb{A}_2^*) = 2$; $\varpi(\mathbb{A}_3^*) = 3$; and so on). Now, a range feature mapping function **R**: $\mathbb{A}^* \rightarrow \mathbb{A}^*$ is uniquely determined. For example, given a phone number "13587878787", its range feature data constructed from **R** is "per cent" – that is, the character corresponding to the decimal coding value 37.

Based on the feature construction approaches given in Examples 3.1 and 3.2, and the XML schema described in Figures 2 and 5 (where the star flags denote the omitted element content) illustrates the storage of an employee record in the cloud. Based on containing feature data and range feature data constructed in this section, in the next section, the mapping transformation for each kind of XML query operations will be introduced.

3.2 Query transformation

As shown in Figure 1, for any XML query defined on privacy data, since it cannot be executed correctly by the cloud server-side, it has to be first transformed to a new query defined on the feature data. Since the XPath language is the most popular query tool for XML (Bo, 2012), this paper uses XPath as the query tool for XML data. In XPath, a query condition is of various forms, but the most essential query conditions can be divided into equivalent queries, containing queries and range queries. Below, the mappings for the three kinds of query conditions are introduced, based on which an XML query condition defined on privacy data can be transformed to a new XML query condition on the related feature data.

Mapping 3.1 (Containing Mapping): A containing query condition is developed based on the function "**contains**", represented by "**contains** (/path/element, $a_1a_2...a_n$)", where "$a_1a_2...a_n$" denotes a character string constant that should be contained in the target data. Then, a containing query condition can be mapped as follows:

contains (/path/element, $a_1a_2...a_n$) ⇒ **contains** (/path/element/conttext, **P**(a_1, a_2)) **and** **contains** (/path/element/conttext, **P**(a_2, a_3)) **and** . . . **and**

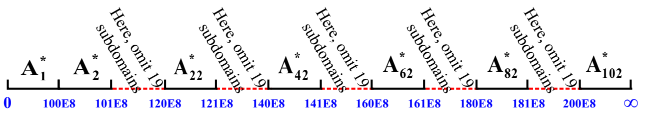

Figure 4.
An example of the partition of the phone number domain

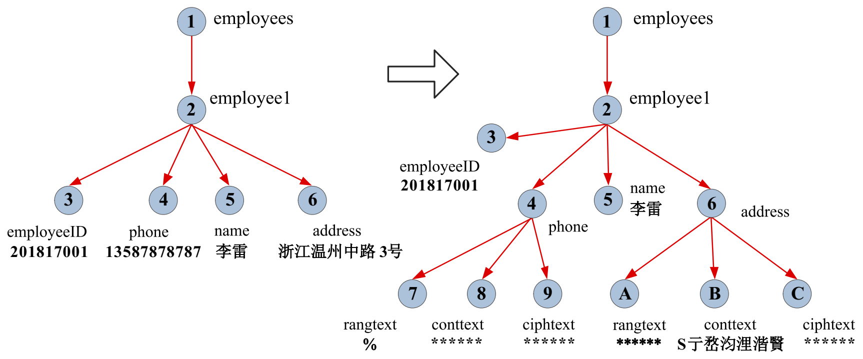

Figure 5.
An example of the XML storage for employees' archives

contains (/path/element/conttext, **P**(a_{n-1}, a_n))

Mapping 3.2 (*Range Mapping*): A range query condition is developed on the comparison of character strings according to their encoding values, denoted by "/path/element $\geq a_1a_2...a_n$". A range query condition can be mapped as follows:

/path/element $\geq a_1a_2...a_n \Rightarrow$/path/element/rangtext $\geq$ **R**($a_1a_2...a_n$)
/path/element $\leq a_1a_2...a_n \Rightarrow$/path/element/rangtext $\leq$ **R**($a_1a_2...a_n$)

Mapping 3.3 (Equivalent Mapping): An equivalent query condition can be viewed as a special containing query, or a special range query, denoted by "/path/element = $a_1a_2...a_n$". Thus, an equivalent mapping can be defined based on the range mapping and containing mapping:

/path/element = $a_1a_2...a_n \Rightarrow$/path/element/rangtext = **R**($a_1a_2...a_n$) **and**
contains (/path/element/conttext, **P**(a_1, a_2)) **and**
contains (/path/element/conttext, **P**(a_2, a_3)) **and** . . . **and**
contains (/path/element/conttext, **P**(a_{n-1}, a_n))

Example 3.3: Based on the feature construction approaches presented in Examples 3.1 and 3.2, three XML query instances (all defined on the XML schema in Figure 2) are used to briefly illustrate the transformations for a containing query, a range query, and an equivalent query.

Query 1. Query the employeeIDs of the employees whose addresses contain "浙江". Then, the original query and the transformed query are as follows (where **P**(浙江) = 浬):

/employees/employee [**contains** (/address, 浙江)]/employeeID⇒
/employees/employee [**contains** (/address/conttext, 浬)]/employeeID.

Query 2. Query the employeeIDs of the employees whose phone numbers are greater than "13587878787" (where **R**(13587878787) = per cent):

/employees/employee [/phone > 13587878787]/employeeID⇒
/employees/employee [/phone/rangtext > %]/employeeID.

Query 3. Query the employeeIDs of the employees whose house addresses are equal to "浙江温州" (where **P**(浙江温州) = 沟浬湝?, and **R**(浙江温州) = f (i.e. the character corresponding to the encoding value 102):

/employees/employee [**contains** (/address, 浙江温州)]/employeeID⇒
/employees/employee [/address/rangtext = f **and contains** (/address/conttext, 沟) **and**
contains (/address/conttext, 浬) **and contains** (/address/conttext, 湝)]/employeeID.

Next, in Section 4.1, will be demonstrated the accuracy of an XML query transformed from the mappings mentioned above. Based on them, it can be concluded that for any element in the XML archives, if its content meets a new query (from Mappings 3.1, 3.2, or 3.3), then the content after decryption certainly meets the original query before transformation. It can be further concluded that the data returned from the cloud by performing a new query related to the feature elements certainly contains the target data – that is, the approach can ensure the query accuracy.

4. Result

4.1 Accuracy analysis

In this section, the accuracy of feature data is analyzed. Below, first introduced are Observations 4.1, 4.2, and 4.3 to demonstrate that each new query operation (which is transformed from Mappings 3.1 to 3.3 and defined on the feature data) can ensure the query accuracy, and then the accuracy of the feature data are further analyzed and demonstrated.

Observation 4.1: Let $\mathbf{W}$ denote a containing query, and $\mathbf{W}^*$ denote the corresponding query after transformation (according to Mapping 3.1). For any privacy data $a_1a_2 \ldots a_n$ (the content of "/path/element"), if it meets the condition of $\mathbf{W}$, then the containing feature data $\mathbf{P}$ $(a_1a_2 \ldots a_n)$ (the content of "/path/element/conttext") certainly meets the condition of $\mathbf{W}^*$.

Explanation: Assume that $\mathbf{P}(a_1a_2 \ldots a_n) = a_1^{'}a_2^{'} \ldots a_{n'}^{'}$. Let $b_1b_2 \ldots b_m$ denote a string constant related to $\mathbf{W}$. Since $a_1a_2 \ldots a_n$ meets $\mathbf{W}$ (i.e. it contains $b_1b_2 \ldots b_m$), there is $a_{t+1}a_{t+2} \ldots a_{t+m} \in a_1a_2 \ldots a_n$ meeting that $a_{t+1}a_{t+2} \ldots a_{t+m} = b_1b_2 \ldots b_m$. From the constraints of Definition 3.2, we have that $\forall a_{t+i}a_{t+i+1} \in a_1a_2 \ldots a_n \rightarrow \mathbf{P}(a_{t+i}, a_{t+i+1}) \in a_1^{'}a_2^{'} \ldots a_{n'}^{'}$. As a result, we have that $\mathbf{P}(b_1, b_2) \in a_1^{'}a_2^{'} \ldots a_{n'}^{'}$, $\mathbf{P}(b_2, b_3) \in a_1^{'}a_2^{'} \ldots a_{n'}^{'}, \ldots,$ $\mathbf{P}(b_{m-1}, b_m) \in a_1^{'}a_2^{'} \ldots a_{n'}^{'}$; that is, the feature data $a_1^{'}a_2^{'} \ldots a_{n'}^{'}$ meets the condition of $\mathbf{W}^*$.

Observation 4.2: Let $\mathbf{W}$ denote a range query, and $\mathbf{W}^*$ denote the corresponding query after transformation (according to Mapping 3.2). For any privacy data $a_1a_2 \ldots a_n$ (i.e. the content of "/path/element"), if it meets the condition of $\mathbf{W}$, then the range feature data $\mathbf{R}(a_1a_2 \ldots a_n)$ (i.e. the content of "/path/element/rangtext") certainly meets the condition of $\mathbf{W}^*$.

Explanation: Let $b_1b_2 \ldots b_m$ denote a string constant associated with $\mathbf{W}$. Since the privacy data $a_1a_2 \ldots a_n$ meets $\mathbf{W}$ (i.e. $a_1a_2 \ldots a_n \geq b_1b_2 \ldots b_m$ (or $a_1a_2 \ldots a_n \geq b_1b_2 \ldots b_m$), based on the constraints of Definition 3.3, we conclude that $\mathbf{R}(a_1a_2 \ldots a_n) \geq \mathbf{R}(b_1b_2 \ldots b_m)$; that is, the feature data $\mathbf{R}(a_1a_2 \ldots a_n)$ meets the condition of $\mathbf{W}^*$.

Observation 4.3: Let $\mathbf{W}$ denote an equivalent query and $\mathbf{W}^*$ denote the query after transformation (according to Mapping 3.3). For any privacy data $a_1a_2 \ldots a_n$ (from the content of "/path/element"), if it meets $\mathbf{W}$, then the containing feature data $\mathbf{P}(a_1a_2 \ldots a_n)$ (from the content of "/path/element/conttext") and the range feature data $\mathbf{R}(a_1a_2 \ldots a_n)$ (from the content of "/path/element/rangtext") certainly meet the condition of $\mathbf{W}^*$.

Explanation: An equivalent query can be viewed as a special containing query or range query. Thus, based on Observations 4.1 and 4.2, Observation 4.3 can be easily proved.

Since equivalent queries, containing queries, and range queries are three kinds of the most essential query conditions in XPath, other kinds of queries can be described directly or indirectly based on them. Thus, can be further concluded that all kinds of XML query operations defined on privacy data can be transformed to new queries on the feature data, and the results returned from the cloud by executing the new queries certainly would contain the accurate results corresponding to the original queries, that is, the approach can well ensure the query accuracy.

4.2 Security analysis

In this section is a discussion of the security of feature data, that is, whether the attacker can obtain the privacy data from the containing feature data or range feature data obtained by the proposed approach. Below, the feature data security is first defined, and then the security metrics for the range feature data and the containing feature data are formulated.

Definition 4.1 (Feature Security): Let a^* denote any character string defined on $\mathbb{A}^*$, and b^* denote its feature data. Then, the feature security can be measured by the possibility of an attacker to guess the privacy data a^* from the feature data b^*, denoted by $Pr(a^* \mid b^*)$.

Definition 4.2 (Range Feature Security Metric): Assume that an attacker has mastered the feature function $\mathbf{R}$. Also, assume that $\mathbb{A}^*$ is divided into N subdomains (Step 1 of the range mapping strategy), and the attacker has known that the length of privacy data is equal to n.

Then, for any range feature data b^*, the possibility that an attacker can infer the corresponding privacy data a^* can be measured as follows:

$$Pr(a^*|\ b^*) = the\ \ domain\ \ size\ \ of\ b^*/the\ \ domain\ \ size\ \ of\ \ a^* = N \cdot |\mathbb{A}|^{-n}$$

Definition 4.3 (Containing Feature Security Metric): Assume that an attacker has mastered the containing feature mapping function **P**. Also, assume that the attacker has known that the length of privacy data is equal to n. Then, for any containing feature data b^*, the possibility that the attacker can infer the privacy data a^* can be measured as (where $m = |b^*|$):

$$Pr(a^*|\ b^*) = the\ \ domain\ \ size\ \ of\ b^*/the\ \ domain\ \ size\ \ of\ \ a^* \leq |\mathbb{A}|^{m-n}$$

From Definitions 4.2 and 4.3, we know that even if the attacker has completely mastered the feature mapping functions, for any feature data, the possibility of obtaining the corresponding plaintext data is still very small. In other words, the feature construction approach proposed in this paper has good security.

4.3 Efficiency evaluation

In the section, the efficiency of feature data is experimentally evaluated; that is, whether each query transformed from Mappings 3.1 to 3.3 can filter out most of the non-target data in the cloud. The experiment is based on the XML schema in Figure 2, and the archives consist of one million randomly generated employee records, where address (whose length is not greater than 30 Chinese characters) and phone number (whose length is fixed to 11 numeric characters) elements are viewed as privacy data.

Definition 4.4 (Efficiency Metric): Let **W** denote a query condition and $\mathbf{W}^*$ denote the new query (after transformation). Let N denote the number of employee records, N_1' the number of employee records meeting **W**, and N_2' the number of employee records meeting $\mathbf{W}^*$. Then, the efficiency of feature data can be measured by the filtering effect of $\mathbf{W}^*$ on the non-target data; that is, $Fr(\mathbf{W}^*,\ \ \mathbf{W}) = \frac{N-N_2'}{N-N_1'}$.

The efficiency evaluation includes two groups of experiments: containing queries for employee addresses; and range queries for employee phone numbers. The first group of experiments aims to evaluate the efficiency of containing queries. The experimental results are shown in Figure 6(a), where the X coordinate denotes the length of the string

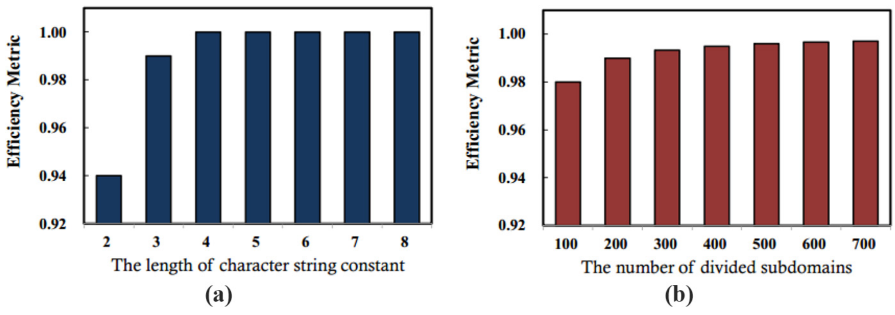

Figure 6.
The experimental results for feature data query efficiency

Notes: (a) Containing query evaluation; (b) range query evaluation constants related to containing queries, and the Y ordinate denotes the efficiency metric values. From Figure 6(a), it can be seen that with the growing of the length of the string constants, the efficiency metric values grow slightly; that is, the filtering effects of the new queries on the non-target data will be improved slightly. This is because the longer the string constant length, the larger the information entropy related to the containing query conditions (the stricter the constraint conditions), leading to the smaller the scales of the target data meeting the query conditions and, in turn, the greater the query efficiency metric values. Also, it can be seen that regardless of the change of the length of the string constants, the efficiency metric values are basically stable (approximate to 1 in most cases; and greater than 0.94 even in the worst case), so most of the non-target data can be filtered out in the cloud, thus improving the containing query efficiency. The second group of experiments aims to evaluate the efficiency of range queries. The experimental results are shown in Figure 6(b), where the X coordinate denotes the number of divided subdomains of the phone number domain (N in Step 1), and the Y coordinate denotes the efficiency metric values. From Figure 6(b), it can be seen that as the N value grows, the filtering effects of the new range queries on the non-target data become better. This is because the increase of the N values makes it become smaller for the amount of plaintext data corresponding to the same feature data, thereby improving the query efficiency. In addition, it can be seen that the range feature data enables most of the non-target data to be filtered out in the cloud (all greater than 0.98), which greatly improves the range query efficiency. From the two groups of experiments, we know that, whether it contains queries or range queries, the cloud by executing the new queries on the feature data can filter out most of the non-target data; that is, the feature data has good query efficiency.

5. Conclusion

Aimed at an XML archive management system in cloud environments, this paper proposes a privacy protection approach. Its basic idea is that any privacy data has to be encrypted in the trusted client before being submitted to the cloud, to improve the security of privacy data in the untrusted cloud. Then, to solve the query problem on the encrypted data in XML archives, this paper proposes to construct feature data for the encrypted data, such that any containing query or range query defined on privacy data can be executed correctly by the cloud, enabling most of the non-target data to be filtered out in the cloud, thus improving the query efficiency. Overall, the proposed approach has the following advantages. On the one hand, compared to policy tools and access control, the approach can prevent not only external attackers but also internal staff on the cloud server-side from accessing archival privacy data, resulting in better security performance. On the other hand, compared to data encryption, the approach can improve the query performance in terms of accuracy and efficiency without compromising data security, leading to better system availability.

This paper is a valuable study attempting to protect privacy in the management of XML archives in a cloud environment, thus of positive significance to promote the further application of cloud computing in the management of XML archives. However, this paper is not the end of the researchers' work. For future work, further studies will be to address some issues that are not solved in this work; for example, how to improve this approach to support more formats of digital archives (not just XML) and the practical implementation of this approach in a real archive system.

References

Attasena, V., Darmont, J. and Harbi, N. (2017), "Secret sharing for cloud data security: a survey", *VLDB Journal*, Vol. 2, pp. 1-25.

Bo, N. (2012), "XML filtering with XPath expressions containing parent and ancestor axes", *Information Sciences*, Vol. 210, pp. 41-54.

Calvanese, D., Giacomo, G.D. and Lenzerini, M. (2018), "Representing and reasoning on XML documents: a description logic approach", *Journal of Logic and Computation*, Vol. 9 No. 3, pp. 295-318.

Chandrasekhar, S. and Singhal, M. (2017), "Efficient and scalable query authentication for cloud-based storage systems with multiple data sources", *IEEE Transactions on Services Computing*, Vol. 10 No. 4, pp. 520-533.

Chandrasekhar, S., Ibrahim, A. and Singhal, M. (2017), "A novel access control protocol using proxy signatures for cloud-based health information exchange", *Computers and Security*, Vol. 67, pp. 73-88.

Chen, Y. (2017), "The problems of records and archives management in cloud environment", *Archives Science Study*, Vol. 2, pp. 35-40.

Ding, W., Yan, Z. and Deng, R.H. (2017), "Encrypted data processing with homomorphic re-encryption", *Information Sciences*, Vol. 409, pp. 35-55.

Gao, C. and Huang, X. (2018), "Research on the security evaluation system of digital archives in the cloud computing environment", *Archives Science Study*, Vol. 1, pp. 77-83.

Lu, C., Wu, Z., Liu, M., Chen, W. and Guo, J. (2013), "A patient privacy protection scheme for medical information system", *Journal of Medical Systems*, Vol. 37 No. 6, p. 9982.

Lu, K. (2019), "Research on privacy protection of university library readers under intelligent service environment", *Library Theory and Practice*, Vol. 3, pp. 18-24.

Mei, Z., Zhu, H., Cui, Z., Wu, Z., Peng, G., Wu, B. and Zhang, C. (2018), "Executing multi-dimensional range query efficiently and flexibly over outsourced ciphertexts in the cloud", *Information Sciences*, Vol. 432, pp. 79-96.

Peng, G., Wang, H., Dong, J. and Zhang, H. (2016), "Knowledge-based resource allocation for collaborative simulation development in a multi-tenant cloud computing environment", *IEEE Transactions on Services Computing*, Vol. 11 No. 2, pp. 306-317.

Power, D., Slaymaker, M. and Simpson, A. (2018), "On formalizing and normalizing role-based access control systems", *The Computer Journal*, Vol. 52 No. 3, pp. 305-325.

Post, G., Kingston, J.H., Ahmadi, S., Daskalaki, S., Gogos, C., Kyngas, J., Nurmi, C., Musliu, N., Pillay, N., Santos, H. and Schaerf, A. (2014), "XHSTT: an XML archive for high school timetabling problems in different countries", *Annals of Operations Research*, Vol. 218 No. 1, pp. 295-301.

Qi, L., Sandhu, R., Zhang, X. and Xu, M. (2017), "Mandatory content access control for privacy protection in information centric networks", *IEEE Transactions on Dependable and Secure Computing*, Vol. 14 No. 5, pp. 494-506.

Servos, D. and Osborn, S.L. (2017), "Current research and open problems in attribute-based access control", *ACM Computing Surveys*, Vol. 49 No. 4, pp. 1-45.

Singh, A. and Chatterjee, K. (2017), "Cloud security issues and challenges: a survey", *Journal of Network and Computer Applications*, Vol. 79, pp. 88-115.

Wang, C., Wang, D., Xu, G. and Guo, Y. (2017), "A lightweight password-based authentication protocol using smart card", *International Journal of Communication Systems*, Vol. 30 No. 16, p. e3336.

Wei, J., Hu, X., Liu, W. and Zhang, Q. (2019), "Forward and backward secure fuzzy encryption for data sharing in cloud computing", *Soft Computing*, Vol. 23 No. 2, pp. 497-506.

Wu, Z., Xu, G., Lu, C., Chen, E., Jiang, F. and Li, G. (2018a), "An effective approach for the protection of privacy text data in the CloudDB", *World Wide Web*, Vol. 21 No. 4, pp. 915-938.

Wu, Z., Zheng, C., Xiejian, J., Zhou, Z., Xu, G. and Chen, E. (2018b), "An approach for the protection of users' book browsing preference privacy in a digital library", *The Electronic Library*, Vol. 36 No. 6, pp. 1154-1166.

Wu, Z., Shi, J., Lu, C., Chen, E., Xu, G., Li, G., Xie, S. and Philip, S.Y. (2015), "Constructing plausible innocuous pseudo queries to protect user query intention", *Information Sciences*, Vol. 325, pp. 215-226.

Xiang, L. and Cheng, M. (2018), "The evolving path of chinese and foreign information security systems", *Journal of Library Science in China*, Vol. 44 No. 2, pp. 113-131.

Yoo, S.K. and Kim, B.Y. (2018), "A decision-making model for adopting a cloud computing system", *Sustainability*, Vol. 10 No. 8, pp. 2952-2962.

Yuan, H., Liu, Y.M., Pan, G.Z., Zhang, G., Zhou, J. and Zhang, Z.J. (2014), "Quantum identity authentication based on ping-pong technique without entanglements", *Quantum Information Processing*, Vol. 13 No. 11, pp. 2535-2549.

Zhao, T. (2018), "Protection of user's privacy in personalized information services of digital libraries", *Library Theory and Practice*, Vol. 2, pp. 101-113.

Corresponding authors
Zongda Wu can be contacted at: zongda1983@163.com and Jun Pan can be contacted at: panjun78@qq.com