# Data Privacy Preservation during Schema Evolution for Multi-tenancy Applications in Cloud Computing*

Kun Zhang, Qingzhong Li**, and Yuliang Shi

School of Computer Science and Technology,
Shandong University, Jinan China
jackie_119@mail.sdu.edu.cn, {lqz,liangyus}@sdu.edu.cn

**Abstract.** In cloud computing, multi-tenancy applications utilize shared resources to serve multi tenants through "single instance multi-tenancy". Applications and databases are both deployed at the platform of un-trusted service providers. Data privacy has become the biggest challenges in wider adoption of cloud computing. Specifically, data schema evolves due to on-demand customization in cloud computing. How to protect the data privacy during data schema evolution for multi-tenancy application is an interesting and important problem. We proposed the privacy requirements for data schema evolution based on the data combination privacy, and then represented the data privacy-preserving architecture. A data schema evolution graph is constructed, and then the privacy-preserving evolution path is searched based on the principle of privacy requirements consistency. Analysis and experiments demonstrate the corrective and effective of the data privacy preservation approach during data schema evolution in cloud computing.

**Keywords:** cloud computing, data privacy, schema evolution.

## 1 Introduction

In cloud computing, multi-tenancy applications utilize shared resources to serve multi tenants based on "single instance multi-tenancy" mechanism. Through multi-tenancy technology, tenants cloud store their data in a shared database in cloud computing and customize the application on demand. Service providers are in charge of management, maintenance and upgrade of applications and databases. Cost is reduced through the scale effect. Multi-tenancy applications have become an important and promising application type in cloud computing.

However, in cloud computing, application and data are both deployed at the platform of un-trusted service providers. Data privacy has become the biggest challenges in wider adoption of multi-tenancy application in cloud computing.

* The research is supported by the National Key Technologies R&D Program No. 2009BAH44B02; National Natural Science Foundation of China under Grant No.90818001; the Natural Science Foundation of Shandong Province of China under Grant No.2009ZRB019YT and No. ZR2010FQ026; Key Technology R&D Program of Shandong Province under Grant No. 2010GGX10105.

** Corresponding author.

Z. Gong et al. (Eds.): WISM 2011, Part I, LNCS 6987, pp. 376–383, 2011.

Especially, data schema may evolve due to on-demand customization of tenants. Data privacy leakage may happen during the process of data schema evolution.

There are already some approaches for privacy preservation, including encryption, obfuscation and so on. Encryption could protect data privacy, but the data processing efficiency is low. Data obfuscation technology retains some properties of data, and performs better than encryption in data processing efficiency. Information disassociation hides the association of sensitive data combination to protect the data privacy. In information disassociation approach, data is in plain text form. However, these privacy preservation approaches didn't consider the data evolution scenario for multi-tenancy application in cloud computing.

For data privacy preservation during data schema evolution for multi-tenancy applications in cloud computing, this paper proposed the solution based on the data combination privacy and privacy requirements consistency. We construct a privacy-preserving data schema evolution graph and translate privacy-preserving data schema evolution problem into the path searching problem in graph.

The rest of paper is organized as follows. Section 2 reviews the related works. Section 3 introduces the basics, i.e. data combination privacy preservation. Section 4 presents the privacy-preserving data schema evolution architecture, gives the privacy requirements for scheme evolution, and proposes the solutions to find the privacy preserving evolution path. Section 5 introduces the experiment. Section 6 makes a conclusion.

## 2 Related Works

In cloud computing, data privacy has become an inevitable challenge. There are already some data privacy preservation architectures and approaches.

In cloud computing, multi-tenancy applications has become an important service type with the development of software as a service. Salesforce.com proposed the multi-tenancy platform Force.com, and presented the meta-data driven multi-tenancy architecture [1].

For data privacy architecture in cloud computing, [2] proposed the "Privacy as a Service" platform. It utilized the cryptographic coprocessors to process the sensitive data and protected program. Reference [3] proposed that the cloud application is trustworthy, and proposed the privacy preservation architecture based on data obfuscation. It utilized token to obfuscate and de-obfuscate data. Reference [4] proposed the client-based privacy manager.

There are some privacy preservation approaches, including encryption, obfuscation and so on. Data encryption is an effective approach, but the data processing efficiency is low. Reference [5] proposed the privacy homomorphism based on ideal lattices. However, the efficiency is not applicable now for cloud computing.

Reference [6,7] proposed a privacy preservation approaches based on combination of encryption and information disassociation. It used the privacy constraint to represent the privacy requirements, and fragmented the sensitive data into different fragments to achieve data privacy. Reference [8] proposed data combination privacy and balancing to protect the data privacy for software as a service.

However, these approaches consider the data privacy preservation in storage. In cloud computing, data storage schema evolves on demand [9]. The data privacy preservation during data schema evolution for multi-tenancy applications in cloud computing should be paid more attention to it.

## 3 Data Combination Privacy Preservation

Based on the sensitive degree of different data combination, the concept of data combination privacy [8] is proposed. In cloud computing, privacy constraint is used to present privacy requirements of tenants. Based on the privacy constraints, tenants' data is fragmented into different data chunks, and the association between data shares of the same data record in different data chunks is hidden. The privacy is protected by the protection of the sensitive association of data shares. When data distribution makes a leakage of data privacy, balancing technology is utilized [8]. This section gives the basis of data privacy-preserving architecture for data schema evolution.

### 3.1 Data Combination Privacy

**Definition 1.** *Data Combination Privacy (DCP).* Data combination privacy is a set of data attributes which are not expected to exposed together. The concrete data value of the data combination privacy could determine a specific person. Specifically, data combination privacy is DCP{$A_1$,$A_2$…$A_m$}, where $A_i$($1 \leqslant i \leqslant m$) is $i^{th}$ data attribute.

In cloud computing, given an specific cloud data physical storage D, the privacy leakage probability of a data combination DC is $P_{DC}$=P(DC|D), which is a conditional probability. When the cloud data physical storage changed, the privacy leakage probability changed accordingly, such as data schema evolution.

Privacy constraint describes the privacy requirements of data combination privacy.

**Definition 2.** *Privacy Constraint (PC).* Privacy constraint is PC{AS(Attribute Set), PP(Privacy Policy)}. AS is the attribute set, and PP is the privacy policy of AS, including Non-Compatible and Compatible. Non-Compatible means that the data combination of AS could violate privacy. Compatible means that the data combination of AS would not violate privacy.

### 3.2 Privacy Preservation

In cloud computing, we use data chunk physical storage. The data chunks fragmented by privacy constraint could be stored into the data chunk storage, and the sensitive association between data chunks is protected. Cloud data chunk storage combines the data combination privacy preservation and cloud data storage mechanism.

After data chunking, un-trusted service providers could infer the data combination privacy from the data distribution in data chunks. So we define the concept of balancing for different scenarios to protect the data privacy, including $\alpha$ balancing, $\beta$ balancing and $\gamma$ balancing. We use fake tuples for data chunks balancing. When the data chunks didn't satisfying the balancing, the fake tuples generator algorithm is used to insert appropriate fake tuples [8].

In cloud computing, we introduced the trusted third party to store the sensitive association information. When data processed, tenant data logical view is reconstructed with the help of trusted third party.

In this scenario, when the tenant cloud data chunk physical storage has k data chunks, and meets α, β, γ balancing, then the data combination privacy of TDLV is not more than $\max(\prod(1/n_i),\beta^k,\gamma^k)$ [8], where $n_i$ is the number of data shares in data chunk i.

## 4 Privacy Preserving Cloud Data Evolution

On-demand customization is supported by cloud computing. Tenants customize the data objects, privacy constraints and other things, which could incur the cloud data chunk physical storage schema evolution. During schema evolution, data privacy leakage may happen and data combination privacy may be violated. This section proposed data schema evolution model and the privacy requirements during schema evolution, and presented the solution.

### 4.1 Cloud Data Chunk Evolution Model

In cloud computing, the tenant cloud chunk physical storage schema evolved caused by the privacy requirements. When tenant customize the privacy constraints, the tenant cloud data chunk physical storage should evolve complying with customized privacy constraints. In this paper, we just consider the privacy requirements change situation.

Firstly, we give the definition of primitive evolution operators.

**Definition 3.** *Creating Data Chunk*. Creating data chunk means creating a new data chunk DCPS(Data Chunk Physical Storage) with the new added data attribute sets. It used the tuple ID to generate the data share id for data chunks. CreatingDataChunk(ID, AS(Attribute Set))=DCPS(DataChunkID, DataShareIDO, AS(AttributeSet)). This evolution operation creates a new data chunk with input attribute set, and associates the data share ID obfuscated with the tuple ID.

**Definition 4.** *Deleting Data Chunk*. Deleting data chunk means deleting the specific data chunk physical storage by the data chunk ID.

**Definiton 5.** *Combining Data Chunks*. For two data chunks $DCPS_i$ and $DCPS_j$, combining data chunk means that combining these two data chunks into one $DCPS_{ij}$, and reconstructing association between data shares. CombiningDataChunks($DCPS_i$, $DCPS_j$)=$DCPS_{ij}$. $DCPS_{ij}$.AS=$DCPS_i$.AS ∪ $DCPS_j$.AS, $DCPS_{ij}$.DataShareIDO = Recompute($DCPS_i$.DataShareIDO, $DCPS_j$.DataShareIDO), DataChunkID = Recompute($DCPS_i$.DataChunkID, $DCPS_j$.DataChunkID).

**Definition 6.** *Splitting Data Chunk*. For a specific data chunk physical storage DCPS, splitting data chunk means that fragmenting DCPS into two data chunks based on the input attribute set, and reconfiguration the DataChunkID and DataShareIDO. SplitDataChunk(DCPS, AS)={$DCPS_i$,$DCPS_j$}. AS is the attribute set of one data chunk.

Based on the data chunk evolution primitive operator, we construct the data chunk schema evolution model. It is a undirected graph model. The vertex represents the specific data chunk physical storage. The edge represents the evolution between two vertexes and represents only one evolution operation. The privacy preserving data evolution problem is translated into the problem of finding a path from source schema to target one while satisfying privacy requirements.

**Definition 7.** *Data Chunk Schema Evolution Graph*. The data chunk schema evolution graph is SEG(V,E). V is the vertex set, and vertex represents the specific data chunk physical storage. The edge represents the evolution between two vertexes and represents only one evolution operation.

Then we should find an evolution path from the data chunk schema evolution graph complying with the following privacy requirements.

For simplicity, we just consider the Combining Data Chunk and Splitting Data Chunk.

### 4.2 Privacy-Preserving Requirements Consistency

We define the principle of privacy-preserving data schema evolution. During the data schema evolution process, privacy requirements consistency should be guaranteed, including privacy constraints consistency and balancing consistency.

#### 4.2.1 Privacy Constraints Consistency

In data chunk evolution path, we define the concept of privacy constraints consistency, which represents the principle of privacy-preserving evolution path.

**Definition 8.** *Privacy Constraints Consistency*. In a evolution graph, given a set of privacy constraints, $C_{pc\text{-}v}(V)$ represents the number of privacy constraints violation for vertex V, $C_{dc\text{-}v}(V)$ represents the number of data chunks violating privacy constraints for vertex V, and $C_v(V)= C_{pc\text{-}v}(V)+ C_{dc\text{-}v}(V)$ represents the number of privacy violation. For privacy constraints consistency in a privacy-preserving evolution path, for two vertexes $V_i$, $V_j(i<j)$, the $C_v(V_i) \geqslant C_v(V_j)$. For target vertex $V_m$, $C_v(V_m)=0$.

Then in the data chunk schema evolution graph, we should find a evolution path satisfying the privacy constraints consistency.

#### 4.2.2 Balancing Consistency

For balancing consistency, we make balancing for each data attribute before evolution. Then during the evolution process, the balancing is unchanged or consistency if data is not changed.

**Theorem 1.** If two data chunks satisfying balancing respectively, then the combination of these two data chunks also satisfying balancing.

**Proof.** We proof it by contradiction.

Suppose two data chunks A and B satisfying balancing, and the number of data shares in data chunk is n. Suppose CombiningDataChunk(A,B)=C didn't satisfying balancing.

For β balancing, there are a certain data combination dc satisfying $P(dc| C)=|dc|/n>\beta$. The percentage of dc projection in data chunk A is $P(dc_A| C)\geq P(dc| C) >\beta$.

$P(dc_A| A) \geq P(dc_A| C)$, so $P(dc_A| A) > \beta$, which is contradictive to the suppose. For $\alpha$, $\gamma$ balancing use the same proof process. Then the thermo is proofed.

If every data attribute satisfies the balancing condition, then in the evolution path, every data chunk physical storage satisfying the balancing consistency.

### 4.3 Shortest Privacy-Preserving Schema Evolution Path Search

Given the source schema and target schema, we proposed a solution to find the shortest evolution path based on graph theory. For data chunk evolution graph, we extend the shortest path algorithm to find evolution path satisfying privacy requirements.

```
Algorithm 1: Shortest Privacy-Preserving Schema Evolution Path Search
Input: Data Chunk Schema Evolution Graph: SEG(V,E)
        Source Schema Vertex: V0
        Target Schema Vertex: Vm
        Cv(V)
Output: The Evolution Path
Steps:
For(every data attribute in data chunks)
        Balancing  //balancing consistency guarantee
For every vertex v in SEG(V,E)
        d[v]=infinity //path length
        previous[v]=undefined //previous vertex in path
d[v0]=0
Set S=empty set
Set Q=SEG.V
While (Q is not an empty set)
        Vertices u=Extract_Min(Q)
        S=S union {u}
        For each edge (u,v) outgoing from u
           If d[v]>d[u]+w(u,v) and (0≤C[v]<C[u] or C[v]=C[u]=0)
              d[v]=d[u]+w(u,v)
              previous[v]=u
//print the evolution path from Vm to V0
Print vm
Vertex temp=vm;
while(previous[temp]!=v0)
        print previous[temp]
        temp=previous[temp]
print v0
```

**Fig. 1.** Shortest privacy-preserving schema evolution path search

The complexity of this algorithm is $O(n^2)$, where n is the number of V in SEG. Through this algorithm, the privacy requirements are satisfied during the evolution path. Firstly, the balancing consistency is guaranteed by the balancing checking. Then during the evolution path searching, the path is satisfying privacy constraints consistency. Then this algorithm is a privacy-preserving data chunk evolution algorithm.

## 5 Experiments

We make experiments to show the data processing efficiency when number of data chunks changes.

**Table 1.** Test types for data operations

| Test Type | Privacy Preservation | Data Chunks Number |
|---|---|---|
| A | Plain Text | 1 |
| B | Encryption | 1 |
| C-1 | Data Chunking | 2 |
| C-2 | Data Chunking | 4 |
| C-3 | Data Chunking | 6 |
| C-4 | Data Chunking | 8 |
| C-5 | Data Chunking | 10 |

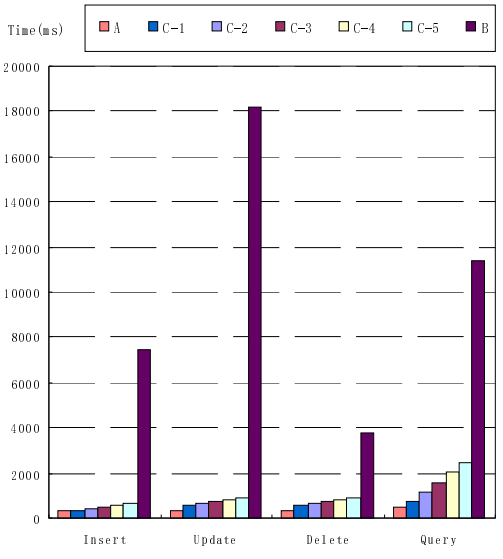


**Fig. 2.** Cost for different data operations

The database is MySQL 5.1.22, developing IDE is Eclipse-SDK-3.5.2-win32, the developing language is Java 5, the operating system is Windows XP Professional Service Pack 2, CPU is Inter Core 2 Duo, 2.33 GHz, and the memory is 2G.

We use the CUSTOMER in TPC-E as the experiment target for data chunking. CUSTOMER includes 24 attributes. We construct the test set based on CUSTOMER. Every test has 10000 tuples. Type A is in plain text form with data chunk 1. Type B is in encryption form with data chunk 1. Type C has different types with different data chunks numbers. The test types are as shown in Table 1.

From Fig.2, we can see that privacy preservation approach based on data chunks gets better performance than encryption. And with the number of data chunks number decrease, the performance gets better. Then we should restrict the number of data chunk, especially during data chunk evolution process.

## 6 Conclusions

The data privacy for multi-tenancy applications in cloud computing could be protected based on the data combination privacy. Due to the on-demand customization of cloud computing, data privacy leakage may happen during the data schema evolution process. This paper proposed privacy-preserving evolution architecture, guaranteeing the consistency of privacy constraints and balancing. The solution could find the shortest evolution path and could guarantee the privacy requirements during data schema evolution for multi-tenancy applications in cloud computing.

## References

1. Weissman, C., Bobrowski, S.: The design of the force.com multitenant internet application development platform. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2009), pp. 889–896. ACM, New York (2009)
2. Itani, W., Kayssi, A., Chehab, A.: Privacy as a Service: Privacy-Aware Data Storage and Processing in Cloud Computing Architecture. In: Proceedings of 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing (DASC 2009), pp. 711–716. ACM, New York (2009)
3. Gu, L., Cheung, S.: Constructing and Testing Privacy-Aware Services in Cloud Computing environment - Challenges and Opportunities. In: Proceedings of the 1st Asia-Pacific Symposium on Internetware (Internetware 2009), pp. 1–10. IEEE Computer Society, New York (2009)
4. Mowbray, M., Pearson, S.: A Client-Based Privacy Manager for Cloud Computing. In: Proceedings of the 4th International ICST Conference on Communication System Software and Middleware (COMSWARE 2009). ACM, New York (2009)
5. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of 41st Annual ACM Symposium on Theory of Computing (STOC 2009), pp. 169–178. ACM, New York (2009)
6. Ciriani, V., Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Fragmentation and Encryption to Enforce Privacy in Data Storage. In: Biskup, J., López, J. (eds.) ESORICS 2007. LNCS, vol. 4734, pp. 171–186. Springer, Heidelberg (2007)
7. Ciriani, V., Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: Combining fragmentation and encryption to protect privacy in data storage. ACM Transactions on Information and System Security (TISSEC) 13(3) (2010)
8. Zhang, K., Li, Q., Shi, Y.: Research on Data Combination Privacy Preservation Mechanism for SaaS. Chinese Journal of Computers 2010 33(11), 2043–2054 (2011)
9. Yan, J., Zhang, B.: Support Multi-version Applications in SaaS via Progressive Schema Evolution. In: Proceedings of the 25th International Conference on Data Engineering (ICDE 2009), pp. 1717–1724. IEEE Press, New York (2009)