Data Combination Privacy Preservation Adjusting Mechanism for Software as a Service

Zhang Kun
Shandong Provincial Key Laboratory of Network Based Intelligent Computing
University of Jinan
Jinan, China
ise_zhangk@ujn.edu.cn

Ajith Abraham
[1] IT4Innovations, VSB-Technical University of Ostrava
Ostrava, Czech Republic
[2] Machine Intelligence Research Labs (MIR Labs), WA, USA
ajith.abraham@ieee.org

Shi Yuliang
School of Computer Science and Technology
Shandong University
Jinan, China
liangyus@sdu.edu.cn

Abstract—**In Software as a Service model, i.e. SaaS, tenants' sensitive data are stored and processed at the platform of un-trusted service providers. Data privacy has become the biggest challenge hindering wider adoption of software as a service. Data combination privacy has been proposed to protect privacy of data combination through sensitive association hidden. However, this approach doesn't consider the scenario where tenants' requirements changed and customization happened. When tenants customize data schema or privacy requirements, there is a possibility that underlying physical data chunk schema collides with the privacy requirements of tenants. This paper proposed the data combination privacy preservation adjusting mechanism for data privacy leakage caused by on demand customization of software as a service. Three principles of privacy preservation adjusting mechanism are proposed. Based on the adjusting mechanism, there would no more privacy leakage than before customization during the adjusting process to the customized schema. Analysis and experiments demonstrate the corrective and effective of the data privacy preservation adjusting mechanism for software as a service.**

Keywords-cloud computing; data privacy; multi-tenancy; data combination privacy; software as a service

I. INTRODUCTION

In Software as a Service model, i.e. SaaS, tenants' sensitive data are stored and processed at the platform of un-trusted service providers. Data privacy has become the biggest challenge in wider adoption of SaaS[5]. 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, which is called data combination privacy in our prior work[16], data is not encrypted. However, these privacy preservation approaches didn't consider the SaaS features. Especially, tenants could customize the SaaS applications on demand, such as data privacy requirements. When privacy requirement changed, existing data privacy preservation mechanism may conflict with it.

So based on the data combination privacy, this paper proposed the data combination privacy preservation adjusting mechanism for data privacy leakage caused by on demand customization in SaaS. Three principles of privacy preservation adjusting mechanism, including privacy constraints complying, leakage avoiding for potential data combination privacy and balancing keeping, are proposed to make the data privacy leakage no more than before.

Given the source schema before customization and target schema after customization, firstly, the adjusting mechanism checks the compatibility between source schema and privacy requirements; secondly, the adjusting solutions are proposed based on the adjusting principles and the target schema. In order to comply with the three principles of adjusting mechanism, we define the concept the privacy preserving adjusting graph for SaaS to represent the underlying physical data chunk schema change. In this adjusting graph, we find privacy-preserving adjusting paths from the source schema vertex to the target schema one, abiding by principles of privacy preservation adjusting, including privacy constraints complying, leakage avoiding for potential data combination privacy and balancing keeping.

The rest of paper is organized as follows. Section II reviews the related works. Section III introduces the basics, i.e. data combination privacy preservation and data chunk physical storage. Section IV presents the data schema adjusting mechanism based on data chunk storage schema for data combination privacy adjusting. Section V presents the data combination privacy adjusting architecture, and gives the privacy requirements for scheme adjusting. Section VI gives solutions for privacy preserving adjusting path search, and gives the path selection mechanism. Section VII introduces some experiments. Section VIII makes a conclusion.

II. RELATED WORKS

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

In SaaS, tenants didn't care about the underlying physical data storage schema and where their data is stored. And there are many research on the shared data schema[1][2][3] for SaaS. Salesforce.com proposed the multi-tenancy platform

978-1-4799-0652-9/13 $31.00 © 2013 IEEE
DOI 10.1109/SMC.2013.344

Force.com, and presented the meta-data driven multi-tenancy architecture[4].

For data privacy architecture in cloud computing, reference[5] proposed a trusted cloud computing platform. Based on this platform, cloud service providers could provide a black-box running environment and protect the confidentiality of virtual machines. Reference[6] proposed the "Privacy as a Service" platform. It utilized the cryptographic coprocessors to process the sensitive data and protected program. Reference [7] supposed 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 [8] 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 [9] proposed an keyword search on encrypted data in cloud computing, which could be reduce the client burden and protect the client query and data privacy. Reference[10] proposed the privacy homomorphism based on ideal lattices. However, the efficiency is not applicable now for cloud computing.

Reference [11][12][13][14][15] proposed a privacy preservation approaches based on 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 [15]consider the fragmentation algorithm in terms of workload. Reference [16] 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 [17]. The data privacy preservation during data schema adjusting for multi-tenancy applications in cloud computing should be paid more attention to it.

III. Data Combination Privacy

Based on the sensitive degree of different data combination, the concept of data combination privacy[16] is proposed. 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. This section gives the basis of data privacy-preserving architecture for data schema adjusting.

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 is not more than $\max(\prod(1/n_i),\beta^k,\gamma^k)$ [16], where n_i is the number of data shares in data chunk i.

IV. Data Schema Adjusting in Cloud

Due to customization, the cloud data chunk storage schema may adjust on-demand. We define this as data schema adjusting in cloud. This section gives the data schema adjusting architecture for tenants in clouds.

A. Adjusting Operation

For cloud data chunk storage schema, we firstly define the primitive operations, including creating data chunk, deleting data chunk, merging data chunks and splitting data chunk. For simplicity, we didn't consider adjusting at data field level and we didn't consider the impact of data replica in cloud.

Definition 1. *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 adjusting operation creates a new data chunk with input attribute set, and associates the data share ID obfuscated with the tuple ID.

Definition 2. *Deleting Data Chunk*. Deleting data chunk means deleting the specific data chunk physical storage by the data chunk ID.

Definition 3. *Merging 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. MergingDataChunks($DCPS_i$, $DCPS_j$)=$DCPS_{ij}$. $DCPS_{ij}$.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 4. *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. SplittingDataChunk(DCPS, AS)={$DCPS_i$,$DCPS_j$}. AS is the attribute set of one data chunk.

B. Schema Adjusting

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

Definition 5. *Data Chunk Schema Adjusting Graph*. The data chunk schema adjusting graph is SAG(V,E). V is the vertex set, and vertex represents the specific data chunk physical storage. The edge represents the adjusting between two vertexes and represents only one adjusting operation.

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

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

Definition 6. *Adjusting Path*. A adjusting path is a simple path from the source vertex to the target vertex. Simple path means that there is no repeated vertex or edges in the adjusting path.

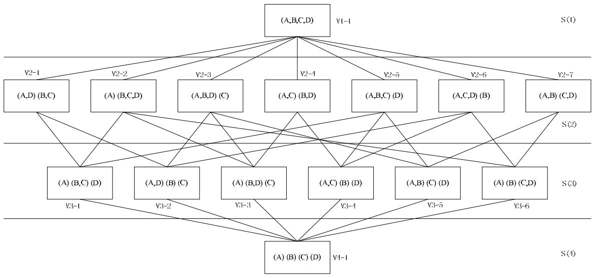


Figure 1. Cloud Data Chunk Schema Adjusting Graph

There are many adjusting path between source schema vertex and target one. Section IV will introduce the requirements of privacy-preserving adjusting path. Section V will give solutions.

V. PRIVACY-PRESERVING DATA SCHEMA ADJUSTING

This section gives the privacy-preserving data schema adjusting mechanism.

A. Privacy Leakage during Schema Adjusting

For tenants, there are two reasons giving rise to the adjusting of data chunk storage schema. (1) Tenants customize privacy constraints, which violate with existing data chunk storage schema and make data combination privacy leakage. (2) Tenants customize the data objects, including adding data fields or deleting data fields, which could make the data chunk storage scheme evolve. In short, the existing data chunk storage schema doesn't comply with the customized privacy constraints, the adjusting should be processed to comply with the privacy-preserving requirements.

As shown in Figure 1, there are many adjusting paths from the source schema vertex to the target one, and the degree of privacy preservation is different. Given the customized privacy constraints is ((A,B), Non-Compatible), ((C,D), Non-Compatible), ((A,D), Non-Compatible), the source schema is V2-1, the target schema is V2-4.

1) The number of violation of privacy constraints may increase during the adjusting path.

As we can see from the adjusting graph in Figure 1, the adjusting path EP: V2-1 → V1-1 → V2-4 belongs to this type. In vertex V1-1, more privacy constraints are violated.

2) The number of data combination leakage increase comparable with source schema and target one.

Consider the adjusting path EP, vertex V1-1 exposes more data combination than source schema and target schema.

3) The balancing is not kept during the adjusting path.

With the data chunks merging and splitting, the balancing condition of data chunks may not kept, which violates the data combination privacy requirements.

B. Principles of Privacy-Preserving Requirements during Schema Adjusting

Three principles of privacy-preserving data schema adjusting are proposed, including privacy constraints complying, leakage avoiding for potential data combination privacy and balancing keeping.

1) Privacy Constraints Complying

Definition 7. *Privacy Constraints Complying (PCC)*. In a schema adjusting graph, given a set of privacy constraints, Complying(V), C(V) in short, represents the number of privacy constraints violation for vertex V. For privacy constraints complying in a schema adjusting path, given two vertexes Vi、Vj(i<j, i.e. Vi is before Vj in the adjusting path), C(Vi)≥C(Vj). For target vertex Vm, C(Vm)=0.

2) Leakage Avoiding for Potential Data Combination Privacy

During the schema adjusting, the data-fields combination, which doesn't appear as subset of data chunk in source schema or target one, has the potential possibility of privacy leakage. So we propose the concept of leakage avoiding for potential data combination privacy, which assures there are no 'big' data combinations appear during the adjusting process. The data chunks, which appear in the adjusting path, should be the subset of the data chunk of source schema or target schema.

Definition 8. *Leakage Degree of Potential Data Combination Privacy*. In adjusting path, Let Potential(Vertex), P(V) in short, denotes the count of data chunks in Vertex V which is neither subset of data chunks in source schema nor subset of data chunks in target one. P(V) is the count of data chunks which potentially violate the privacy constraints in the future.

Definition 9. *Leakage Avoiding of Potential Data Combination Privacy(LAPDCP)*. Let DataChunks(V) denotes the data chunks set of vertex V in the adjusting graph, dc denotes the data chunk, V0 denotes the source schema, Vm denotes the target schema, then leakage avoiding of potential data combination privacy requires that for any vertex V_i in the adjusting path,

$$\forall dc \in DataChunk(V_i),\quad (\exists \mathrm{dc}_{\mathrm{V0}} \in \mathrm{DataChunks(V0)}\ , \mathrm{dc} \subseteq \mathrm{dc}_{\mathrm{V0}}\ )\ .$$
$$\text{or } (\exists \mathrm{dc}_{\mathrm{Vm}} \in \mathrm{DataChunks(Vm)}, \mathrm{dc} \subseteq \mathrm{dc}_{\mathrm{Vm}})$$

Leakage avoiding requires that the data chunks appears in the adjusting path should be either ssubset of data chunks of source schema or subset of data chunks of target schema, i.e. for any vertex V in the adjusting path, P(V)=0.

3) Balancing Keeping

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

C. Privacy-Preserving Schema Adjusting Mechanism

Based on the principles of privacy-preservation during adjusting process, privacy vector is proposed to measure the privacy preservation degree of adjusting vertex and is used to guide the privacy-preserving adjusting path search and selection.

Definition 10. *Privacy Vector*. Given the source schema, target schema, privacy constraints and balancing parameters, privacy vector is a three-dimensional vector for a certain vertex, i.e. privacy vector is PV{Privacy-Violation, Potential-Data-Combination-Privacy-Violation, Balancing-Keeping-Status}. Privacy- Violation is the sum of Privacy Constraints Violation and Data Chunks Violation, i.e. C(V). Potential-Data-Combination-Privacy-Violation is the number of the data chunks which is neither subset of data chunks of source schema nor the target one, i.e. P(V). Balancing-Keeping-Status is { (α,datachunks), (β,datachunks), (γ,datachunks)}, i.e. the balancing-keeping status. For simplicity, we just consider the Balancing-Keeping-Status has two entries, including balancing-keeping (BK) and balancing-violation (BV).

Given the privacy vector, we define the partial order between privacy vectors.

Definition 11. *Partial Order for Privacy Vector*. For two next vertexes V1, V2 in the adjusting path, if C(V1)≥C(V2) and P(V1)≥P(V2), BSK(V1)=BSK(V2) or (BSK(V1)=BK and BKS(V2)=BV), then PV(V1)≥PV(V2).

Based on the privacy vector, we give the definition of privacy-preserving adjusting path.

Definition 12. *Privacy-Preserving Adjusting Path*. Given the adjusting graph, source schema vertex, target schema vertex, and privacy vectors, a privacy-preserving adjusting path satisfies the partial order between vertexes in the adjusting path. For a privacy-preserving adjusting path V0->V1->…Vi->Vi+1->…Vj…->Vm, if i≤ j, then PV(Vi)≥PV(Vj).

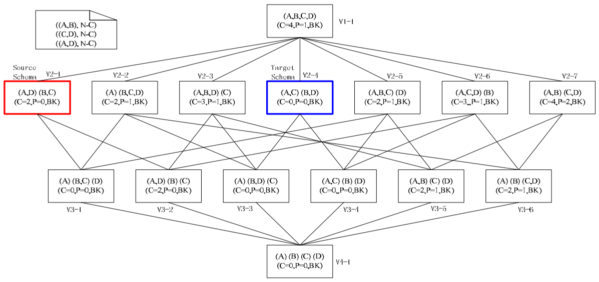


Figure 2. Privacy-Preserving Schema Adjusting Graph

Then along the privacy-preserving adjusting path, the privacy vector is getting smaller. A specific example is shown as below.

Property 1. In the layered adjusting graph, there exist two vertexes V_i and V_{i+1}, which is on the layer I and i+1 respectively. There are edges between V_i and V_{i+1}. Then $PV(V_i) \geq PV(V_{i+1})$.

As we can see from Table I and Figure 2, there are maybe more than one privacy-preserving adjusting path. How to select the most appropriate privacy-preserving adjusting path is an interesting problem. Section 5 will make research on it.

TABLE I. PRIVACY-PRESRVING ADJUSTING PATH

Path ID	Adjusting Path	PP
EP-1	V2-1(2,0,BK) → V3-1(0,0,BK) → V4-1(0,0,BK) → V3-3(0,0,BK) → V2-4 (0,0,BK)	Yes
EP-2	V2-1(2,0,BK) → V3-2(2,0,BK) → V4-1(0,0,BK) → V3-3(0,0,BK) → V2-4 (0,0,BK)	Yes
EP-3	V2-1(2,0,BK) → V1-1(4,1,BK) → V2-4(0,0,BK)	NO

VI. PRIVACY-PRESERVING ADJUSTING PATH SEARCH AND SELECTION

This section gives the solutions for privacy-preserving adjusting path search, evaluation and selection. We give the evaluation metrics of privacy-preserving adjusting path. And propose cross partition based solutions for privacy-preserving adjusting path selection.

A. Privacy-Preserving Adjusting Path Evaluation

This section gives the metric for the privacy-preserving adjusting path.

1) The length of privacy-preserving adjusting path. When the privacy-preserving adjusting path is shorter, the cost of adjusting and service down time is smaller.

2) The degree of privacy vector decay. When the degree of decay is huger, the period privacy violation is shorter.

Definition 13. *Privacy Vector Decay Ratio*. In a privacy-preserving adjusting path ppep, let L be the number of vertexes in ppep, the first well-defined adjusting vertex Vt is the t^{th} vertex in ppep, then the privacy vector decay ratio of ppep is

$$PVDR = {}^{L}\!/_{t} \cdot \qquad (1)$$

The huger PVDR, then more quickly the privacy vector decays, less periods the privacy violation, and the better the privacy preservation.

Definition 14. *Measure Vector for Privacy-Preserving Adjusting Path*. For a privacy-preserving adjusting path, the measure vector is Score(PPEP,L,PVDR). The partial order is:

1) If L1>L2, Score(PPEP1,L1,PVDR1) > Score(PPEP2,L2,PVDR2).

2) If L1=L2 and PVDR1 ≥ PVDR2, then Score(PPEP1,L1,PVDR1) ≥ Score(PPEP2,L2,PVDR2).

As shown in Figure 2. The privacy vector measure vector of two privacy-preserving adjusting paths is shown in Table 4.

TABLE II. MEASURE VECTOR FOR PRIVACY-PRESERVING ADJUSTING PATH

Path ID	**Privacy-Preserving Adjusting Path**	**Measure Vector**
EP-1	V2-1(2,0,BK) → V3-1(0,0,BK) → V4-1(0,0,BK) → V3-3(0,0,BK) → V2-4 (0,0,BK)	Score(EP-1,L=5,PVDR=5)
EP-2	V2-1(2,0,BK) → V3-2(2,0,BK) → V4-1(0,0,BK) → V3-3(0,0,BK) → V2-4(0,0,BK)	Score(EP-2,L=5,PVDR=2.5)

As we can see from Table II, the Score(EP-1)>Score(EP-2), then privacy-preserving adjusting path EP-1 is more appropriate than EP-2 in terms of measure vector.

B. Privacy-Preserving Adjusting Path Selection based on Cross Partition

In schema adjusting graph, the leakage avoiding needs the adjusting path along the refinement of source schema target or the target one. Cross partition is the point of junction of two refinements.

Definition 15. *Cross Partition*. Given a set DF (Data Fields) of data fields belonging to the same data object, partition A $\{A_1, A_2, \dots A_r\}$ and partition B$\{B_1, B_2, \dots, B_s\}$ are two partitions of set DF, then set containing the satisfying element, which is $A_i \cap B_j \neq \varnothing$, is called cross partition of two partitions.

Given the partition refinement, partition A and B could be refined into the cross partition. Along with the refinement, we could get an adjusting path from the partition A and B, i.e. the source schema vertex and the target one. It is observed that the adjusting path is privacy-preserving.

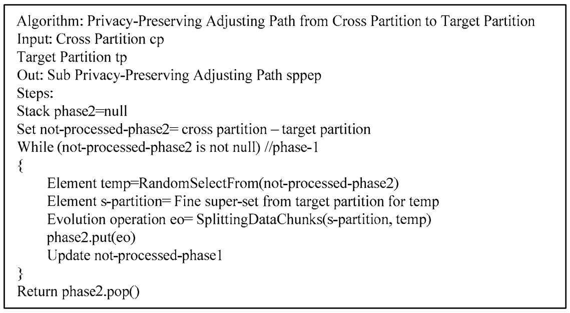
Definition 16. *Adjusting Path based on Cross Partition*. In schema adjusting graph, given the source schema partition and target schema one, cross partition is determined. The adjusting path along with the refinement process to the cross partition is called adjusting path based on cross partition. There are many adjusting paths based on cross partition given the source partition and target one.

We can prove that the adjusting path based on cross partition is the shortest in all privacy-preserving adjusting paths.

In phase 1 from source schema to cross partition schema, splitting data chunk operation is in charge, and combining data chunks operations in charge during phase 2 (from cross partition schema to target schema).

Just consider data privacy preservation and the adjusting length, the phase 2 can be done by combining any data chunks inverse the refinement from target schema to cross partition.

The adjusting algorithm is show as Figure 3.



```
Algorithm: Privacy-Preserving Adjusting Path from Cross Partition to Target Partition
Input: Cross Partition cp
Target Partition tp
Out: Sub Privacy-Preserving Adjusting Path sppep
Steps:
Stack phase2=null
Set not-processed-phase2= cross partition – target partition
While (not-processed-phase2 is not null) //phase-1
{
      Element temp=RandomSelectFrom(not-processed-phase2)
      Element s-partition= Fine super-set from target partition for temp
      Evolution operation eo= SplittingDataChunks(s-partition, temp)
      phase2.put(eo)
      Update not-processed-phase1
}
Return phase2.pop()
```

Figure 3. Privacy-Preserving Adjusting Path from Cross Partition to Target Partition

VII. EXPERIMENTS

We make experiments to show the effective of the privacy preservation during schema adjusting.

A. Experiments Configuration

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.

B. Privacy-Preserving Adjusting Path Searching Cost

TABLE III. TEST TYPES

Test Type	**Number of Data Fields**	**Privacy Constraints Ratio**
sc-a-1	100	20%
sc-a-2	100	40%
sc-a-3	100	60%
sc-a-4	100	80%
sc-a-5	100	100%
sc-a-6	100	120%
sc-a-7	100	140%

We make experiments to show the cost of different privacy-preserving adjusting path search solutions. We just compare three path selection algorithms for phase-1, because the phase-2 is the same process and has no contribute to the PVDR decay. Make experiments to get selection cost when fixed number of data fields.

C. PVDR test

We fixed the test environment. Number of data fields is 100, source partition is a partition with two data chunks, cross partition is the maximum partition. The privacy constraints are generalized randomly, and then make privacy constraints redundancy and conflict checking. Given different ratio of well-defined privacy constraints, three algorithms are tested for the PVDR decay.

From the source schema from layer 1 to the cross partition at layer 100, there are 100 adjusting points. For simplicity, the non-compatible privacy constraints in test include 2 data fields.

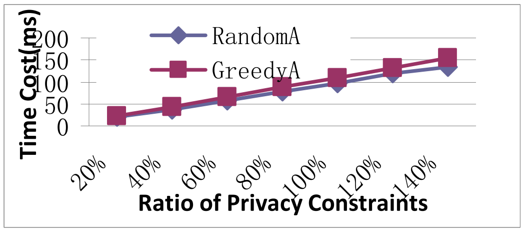


Figure 4. Time cost when ratio of privacy constraints changes

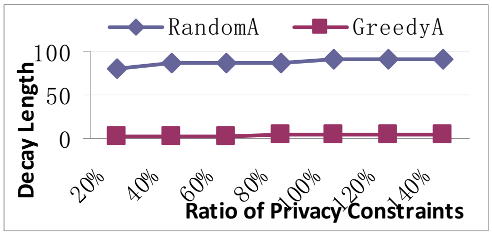


Figure 5. PVDR decay

VIII. Conclusions

The data privacy for SaaS could be protected based on the data combination privacy. Due to the on-demand customization of software as a service, data privacy leakage may happen when data privacy requirements collide with the underlying cloud data chunk storage schema. This paper proposed data combination privacy preservation adjusting mechanism for this problem. A privacy-preserving adjusting architecture is introduced, guaranteeing the consistency of privacy requirements. Solutions are proposed to search the privacy-preserving path, which could guarantee the privacy requirements during underlying data schema adjusting for software as a service.

Acknowledgment

This work is supported by Foundation for Doctors of University of Jinan under Grant No. XBS1316, the National Natural Science Foundation of China under Grant No.61272241, the Natural Science Foundation of Shandong Province of China under Grant No. ZR2010FQ026, Independent Innovation Foundation of Shandong University under Grant No. 2012TS074. This work is partially supported in the framework of the IT4 Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070 by operational programme ‘Research and Development for Innovations’ funded by the Structural Funds of the European Union and state budget of the Czech Republic, EU.

References

[1] Jacobs, D., Aulbach, S.: "Ruminations on multi-tenant databases". In: Kemper, A., Sch?ning, H., Rose, T., Jarke, M., Seidl, T., Quix, C., and Brochhaus, C. (eds.) BTW. pp. 514-521. GI 2007.

[2] Aulbach, S., Grust, T., Jacobs, D., Kemper, A., Rittinger, J.: "Multi-tenant databases for software as a service: schema-mapping techniques". Proceedings of the 2008 ACM SIGMOD international conference on Management of data. pp. 1195-1206. ACM, New York, NY, USA, 2008.

[3] Aulbach, S., Jacobs, D., Kemper, A., Seibold, M.: "A comparison of flexible schemas for software as a service". Proceedings of the 35th SIGMOD international conference on Management of data - SIGMOD ’09. pp. 881-888. ACM Press, New York, New York, USA, 2009.

[4] Weissman, C.D., Bobrowski, S.: "The design of the force.com multitenant internet application development platform". Proceedings of the 35th SIGMOD international conference on Management of data. pp. 889-896. ACM, New York, NY, USA , 2009.

[5] Santos, N., Gummadi, K.P., Rodrigues, R.: "Towards trusted cloud computing". Proceedings of the 2009 conference on Hot topics in cloud computing. USENIX Association, Berkeley, CA, USA, 2009.

[6] Itani, W., Kayssi, A., Chehab, A.: "Privacy as a service: privacy-aware data storage and processing in cloud computing architectures". 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing. 711-716, 2009.

[7] Gu, L., Cheung, S.-C.: "Constructing and testing privacy-aware services in a cloud computing environment: challenges and opportunities". Proceedings of the First Asia-Pacific Symposium on Internetware. p. 2:1--2:10. ACM, New York, NY, USA, 2009.

[8] Mowbray, M., Pearson, S.: "A client-based privacy manager for cloud computing". Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE - COMSWARE ’09. 1 , 2009.

[9] Liu, Q., Wang, G., Wu, J.: "An efficient privacy preserving keyword search scheme in cloud computing". Proceedings of the 2009 International Conference on Computational Science and Engineering - Volume 02. pp. 715-720. IEEE Computer Society, Washington, DC, USA, 2009.

[10] Gentry, C.: "Fully homomorphic encryption using ideal lattices". Proceedings of the 41st annual ACM symposium on Theory of computing. pp. 169-178. ACM, New York, NY, USA, 2009.

[11] Ciriani, V., di Vimercati, S.D.C., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: "Fragmentation and encryption to enforce privacy in data storage". In: Biskup, J. and Lopez, J. (eds.) ESORICS. pp. 171-186. Springer , 2007.

[12] Ciriani, V., Capitani Di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: "Enforcing confidentiality constraints on sensitive databases with lightweight trusted clients". Proceedings of the 23rd Annual IFIP WG 11.3 Working Conference on Data and Applications Security XXIII. pp. 225-239. Springer-Verlag, Berlin, Heidelberg, 2009.

[13] Ciriani, V., De Capitani Di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: "Keep a few: outsourcing data while maintaining confidentiality". Proceedings of the 14th European conference on Research in computer security. pp. 440-455. Springer-Verlag, Berlin, Heidelberg, 2009.

[14] Ciriani, V., Vimercati, S.D.C.D., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: "Combining fragmentation and encryption to protect privacy in data storage". ACM Trans. Inf. Syst. Secur. 13, 22:1--22:33 , 2010.

[15] Ciriani, V., Vimercati, S.D.C. di, Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P.: "Fragmentation design for efficient query execution over sensitive distributed databases". Proceedings of the 2009 29th IEEE International Conference on Distributed Computing Systems. pp. 32-39. IEEE Computer Society, Washington, DC, USA, 2009.

[16] ZHANG, K., LI, Q.-Z., SHI, Y.-L.: "Research on data combination privacy preservation mechanism for SaaS". Chinese Journal of Computers. 33, 2044-2054, 2010.

[17] Yan, J., Zhang, B.: "Support Multi-version applications in SaaS via progressive schema evolution. Proceedings of the 2009 IEEE International Conference on Data Engineering". pp. 1717-1724. IEEE Computer Society, Washington, DC, USA, 2009.