International Journal of Database Theory and Application
Vol.7, No.3 (2014), pp.31-40
http://dx.doi.org/10.14257/ijdta.2014.7.3.04

Efficient Query Integrity Protection for Multi-tenant Database

Li Lin[1,2], Li Qingzhong[1,2*], Kong Lanju[1,2] and Shi Yuliang [1,2]

[1]*School of Computer Science and Technology, Shandong University, Jinan, P.R. China*
[2]*Shandong Provincial Key Laboratory of Software Engineering*
ducklilin@126.com, lqz@sdu.edu.cn，klj@ sdu.edu.cn, syl@sdu.edu.cn

Abstract

In SaaS, since the service provider may be un-trusted, it is essential for tenants to enable query result correctness and completeness. However, existing data authentication methods can not fit well with the customized multi-tenants sharing storage mode. This paper put forward a multi-tenant data authentication model (TCDA). TCDA is a composite structure that constructs pivot authentication tree (α-tree) on the pivot table and combines it with signature set (β-set) built on sparse table to ensure that malicious insiders can't modify the data in pivot table and sparse table. The main contribution of TCDA is it can guarantee the tenant query result in one tree travels and return the verification object corresponding to the result on pivot table and sparse table. And in this paper, we propose an improved TCDA *model to minimize the processing overhead through appending aggregation signature of the node descendants to the internal* α-tree *node. We demonstrate effectiveness of our model compare with MHT and DSAC through the experiment.*

Keywords: *SaaS , Multi-tenant, Shared database, Query integrity*

1. Introduction

Software-as-a-Service, *i.e.*, SaaS[1]is a software delivery model in which software and associated data is centrally hosted on the cloud. By leasing the service and putting the data to the service providers, the tenants can be relieved of the burden of computation and storage and pay more attention to their business. However, the service provider may not be trusted, or may be compromised, it is essential to enable verification of the results by the tenants. In particular, tenants should be able to guarantee that the returned results are both correct and complete. Correctness implies that the result data records indeed the tenant's legitimate source data, and that they have not been tampered with in any way. Completeness requires that no qualifying records have been omitted.

Among existing query result authentication methods, Merkle hash tree based approaches [2-5] in which the MHT was embedded into the data index and the VO is created during query processing shown the advantage compared to the other approaches. However, there are some obstacles for the index authentication approaches such as MB tree [4] to apply suitably on tenant data authentication in SaaS for the following reasons:

First, because most SaaS service providers adopt the single instance multi-tenancy strategy to take full advantage of resources such as hardware and database, multiple tenants' data is stored in one physical table such as sparse table in which different data types are stored into a flex column based on tenants' customization [6]. While the MB tree needs to set up the index

* Corresponding author

ISSN: 2005-4270 IJDTA

on the query attribute that should be the same data type and MB tree lacks the ability to discern the tenant identifier. Although we can include the tenant identifier into the search key, the MB tree contains plenty of duplicate messages and can't support the isolation storage needs of tenant index.

Second, in a real-world scenario tenants may customize different integrity demand based on their needs, e.g., some tenant applications may favor a fast response over a verified one. While the query processing and VO computation imposes conflicting requirements on the MB tree. For instance, a high fanout is desirable in order to reduce the query evaluation cost, but that leads to a large VO [7]. And MB tree propagates every data update up to the root digest, so an update transaction must lock the entire index in exclusive mode and block all other updates and queries[8],which are not conform to the performance isolation requirements of the multi tenants.

Third, in order to guarantee performances of query operations in large multi-tenants database, adequate pivot table [9] for tenant data are set up to speed up the query process. Those data stored in pivot table should also be included in the integrity consideration of tenants' data integrity protection.

Based on the above reasons, it is improper to use the index authentication schemes directly in SaaS for their poor isolation performance between multi tenants. Besides MHT based approaches, signature aggregations [10] is another technique for query answer authentication. And signature aggregation offers an important advantage; since a record update affects only its own signature (and that of its immediate left/right neighbors in some schemes), it is easier to guarantee isolation between tenants and multiple updates can be executed simultaneously. But it needs additional means to guarantee the completeness of the query result such as using the continuity of the query attribute to set up signature chaining[11], which introduces a lot of supporting work for the result correctness and completeness checking.

The objective of our work is to devise a scalable query answer authentication mechanism for multi-tenant databases. In order to protect data integrity and ensure the performance isolation between tenants, we put forward a tenant composite data authentication model (TCDA). In order to meet the different integrity requirement of different tenants, TCDA is independent with the index structures built on pivot table. The main idea of TCDA is a composite structure that constructs pivot authentication tree (α-tree) on the pivot table and combines it with signature set β-set built on sparse table to ensure that malicious insiders can't modify the data in pivot table and sparse table. TCDA can guarantee the tenant query result in one tree travels, while return the VO (verification object) corresponding to the result on pivot table and sparse table. And in this paper, we propose an improved TCDA model to minimize the processing overhead through appending aggregation signature of the node descendants to the internal *α-tree* node. By the aggregate information at the intermediate nodes of the *α-tree* we can authenticate the query results without having to traverse the tree all the way to the leaves to get the VO. We demonstrate effectiveness of our model compare with applying the MHT and DSAC [11] directly on pivot table and sparse table through the experiment.

The rest of this paper is organized as follows. The next section covers related works. In Section 3 we present the secure system model. Section 4 introduces the TCDA model and Section 5 presents an improved TCDA model and Section 6 shows the experiment. Section 7 gives the conclusion of this paper.

2. Related Work

Integrity protection is research hot spot in outsourced database and cloud computing. Reference [12] proposes a partially materialized digest scheme in which split the

authentication structure from the data index and they extend their work to the spatial database, but it also did not apply for the multi-tenant circumstance. In [13] iBigTable provides data integrity assurance for BigTable and designed a set of security protocols to verify the integrity of data returned by BigTable. For secure data storage in cloud computing, [14] proposed an effective and flexible distributed schema by utilizing the homomorphic token with distributed verification of erasure-coded data. Reference [15] inserted certain fake tuples into the real data and verified query integrity by checking the fake tuple in the result. Reference [16] presented the dual encryption approach, where certain data are encrypted with different keys and query integrity could be checking by "cross examination". Reference [17] proposed PORs model, which enables an archive or back-up service to produce a concise proof that a user (verifier) can retrieve a target file. Reference [18] presented a formal security definition of query integrity in outsourced dynamic databases. All those works well in their scenario but could not apply to the multi-tenant storage ideally. Reference [19] focuses on the case that service providers are not always trustworthy and promote a meta-data driven data chunk based secure data storage model for SaaS to ensure the data integrity. But it also did not give an appropriate solution on how to guarantee both the pivot table and tenant data.

3. Secure System Model for SaaS

This section give the over view of system model, attack model and storage model in SaaS.

System Model. The system model includes three entities: tenant, trusted third party and service provider.

Tenant **T**: **T** is a client that customizes and consumes SaaS applications. In SaaS, tenants rely on the service provider for data maintenance and computation.

The trusted third party (**TTP**): The trusted third party is used to assist tenants for their secret key and integrity policy information management. TTP can prohibit unauthorized parties from getting tenant's privacy information.

Service provider (**SP**): Service provider is responsible for the operation of SaaS platform. SaaS platform is a mechanism that has significant computing resource and storage space to maintain the tenants' applications and data storage and provides public SaaS applications.

Attack Model. We assume that the SP are not necessarily trusted because the malicious insiders. Based on the research on trust platform [20], we assume that the SaaS platform can be trusted, and we explore an integrity protection module(IPM) in platform, IPM can assist tenants for their data integrity customization and verify the data integrity with the help of the trusted third party. Besides, we assume that all communications go through a secure channel between the SP, TTP and tenants.

Based on the above assumptions, we concentrate on the analysis of malicious behavior from the SP malicious insiders. For example, insiders may delete the record of a tenant in universal table, change the data item in pivot table or universal table or forge some non-existence record to tenants' data hosted by SP storage, which violates tenant data integrity.

Storage Model. In this paper, we mainly discuss the scenario that multiple tenants share a single application with logical view $R(A_1, A_2, ..., A_n)$,and tenant custom A_1 as the search attribute and register query on it (Here we mainly aim at the case that searching key data type is numeric type and does not have duplicates). The physical view in shared table corresponding to R containing records as $r(guid, T, value_1, value_2, ..., value_n)$, while $value_1$, $value_2$, ..., $value_n$ corresponding to $A_1, A_2, ..., A_n$. The values of A_1 are stored in pivot table as record t(indexID, value, guid).

In this paper we consider equality and range selections. Equality selections are treated as a special case of range selections, so we focus on the latter. Suppose tenant T request a query Q

such as (SELECT * FROM R WHERE ql< A1<qu), where q_l(q_u) is the lower(upper) bound of Q. The set of tuples that satisfy the query predicate is denoted by Set(Q), and the final answer returned is ANS(Q). The process of queries of tenant in SaaS can be defined as follows: When tenant pose queries Q to SP, the data engine takes charge of query transformations and submit those queries to the data node: first data engine register query QP on the pivot table to get the middle result set Set(QP) ; then data engine register query QS on the sparse table based on Set(QP) and get the result set ANS(Q). So the IPM gets the ANS(Q) along with the VO. VO enables the data engine to verify the correctness and completeness of ANS(Q). If the result set is legitimate, the data engine returns those data to the tenant though application , else the data engine reject the result set.

4. Tenant Composite Data Authentication Model

In this section we present the tenant composite data authentication model(*TCDA*) and verification method for tenant data. We analyze its performance and security.

4.1. The TCDA Approach

In this section, we proposed a solution for tenant data authentication in SaaS called tenant composite data authentication model (TCDA). Conceptually, TCDA is a composite tree. Letting *N* be the number of record in R.

Definition 1 Tenant Composite Data Authentication model (TCDA) TCDA consists two layers, *TCDA* = {*α-tree, β-set*}, where:

α-tree is authentication structure built for pivot table. *α-tree=<root, Lnode, Inode>, root* is the root node of *α-tree* . The root of the *α-tree* is built on top of the *N* leaf nodes and signed with the tenant's private key. *Lnode* presents leaf node. Every leaf node stores the hash value of the pivot table ordered by the searching attribute. $Lnode=<k,h,p_r,p_s>$, where *k* is the searching key of *α-tree* , *h* is the hash value corresponding to *k* in pivot table $h= H(t_j)=H(idxID|k|guid)$, p_r is a sibling pointer to the next leaf node and p_s is pointed to the signatures of the records in the sparse table corresponding to the searching attribute. *Inode* is internal nodes and $Inode=<k,h,p_l,p_r>$, where *k* is the searching key, *h* is the digest of its children hash concatenation $h = h_{j|k}=H(H(t_j)|H(t_k))$ and ,p_l,p_r indicates the pointers to the *Inode* children.

β-set is signature set corresponding to the records in the sparse table. *β-set= {<TenantID, GUID, SN>$_i$} (i=1…N)*, where the *TenantID* indicates the owner of the signature, *GUID* is the search key value and *SN* represents the digital signature, $SN_i = sig(h(r_i)) = sig(h(guid|R|a_{i1}|a_{i2}|\ldots|a_{in}))$ where | denotes string concatenation.

VO Construction and Authentication To prove authenticity and completeness, in addition to Set(QP) and ANS(Q), the two boundary leaves q_l-and q_{u+}, falling immediately to the left and to the right of q_l and q_u. DE proves if (i) ANS(Q) between q_l and q_u have not been tampered with, (ii) all records between q_l and q_u are returned. To prove (i) and (ii), we compute a VO using the TCDA.

Figure 1 illustrates an example of the TCDA in a scenario where the tenant T's logical view R contains N = 8 records and A_1 is the search attribute. The records corresponding to R in sparse table is r_i(i=1...8) and the records corresponding to A_1 is in pivot table t_j(j=1...8) here.

The TCDA's work flow is defined as follow. Consider the paths that lead to the left and to the right boundary leafs q_l-and q_{u+}. The VO for query Q contains (i) the signed α-tree root, (ii) all left sibling hashes to the path of q_l-, (iii) all right sibling hashes to the path of q_{u+} and (iv)

the aggregated signatures S of r_is that satisfy the Q in β-set. Upon receipt Set(QP) and ANS(Q), the IPM first combines Set(QP)with VO components (ii) and (iii), to reconstruct the missing part of the α-tree between the paths of q_l-and q_{u+}, and verifies with the owner's public key whether the root of α-tree (i.e., component (i) of the VO) matches the computed root hash. If they match, Set(QP) is deemed both complete and authentic; the collision-resistance ofthe hash function ensures that it is computationally infeasible for the server to tamper with the result and yetmanage to produce hashes that match the original ones. Then the DE verifies the S with ANS(Q) by aggregated signature scheme.

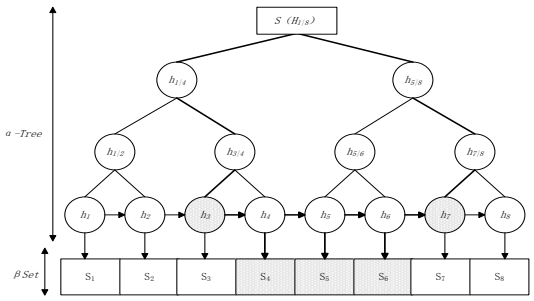

Figure 1. An Example of *TCDA*

Security analysis The correctness of Set(QP) is guaranteed by α-tree due to the security of collision-resistance hash functions and the public key digital signature for the hash value of the root node. Completeness can be assured by the sorted leaves and the boundary leaves that enclose the select range. Based on the β-set, the IPM can verify the correctness of ANS(Q), but it can't discover if the vicious insider delete the tenant record in sparse table for the records of ANS(Q) may be scattered in the share table based on the query attribute.

Lemma 1 If the Set(QP) is correct and complete, any deletion on the sparse table of ANS(Q) can be checked by compare Set(QP) with ANS(Q) on DUID, if {GUID| Set(QP)}={GUID| ANS(Q)},we can say that ANS(Q) is complete.

Proof As the query process of QS can be treated as equal-join query between pivot table and sparse table with join condition Pivottable.GUID= Sparsetable.GUID in their respective attribute. And the GUID attribute is the globally unique identifier for record level rapid positioning, it is a one-to-one correspondence between Pivot table and Sparse table. So, if {GUID| Set(QP)}={GUID| ANS(Q)},we can say that ANS(Q) is complete.

Based on the α-tree we can ensure the correctness and completeness of Set(QP). And according to Lemma 1 and aggregated signature, we can check the completeness and correctness of ANS(Q).

5. An Improved TCDA Model

The TCDA model in Section 4.1 needs to travel to the α-tree leaf node to get the all the corresponding signatures during the VO construction which may influence the performance of VO construction. In this section, we present a method to authenticate IPM queries efficiently using improved TCDA model. The basic idea is that the aggregate signatures at the intermediate nodes of the tree can be used to answer the aggregated signatures of ANS(Q) without having to traverse the tree all the way to the leaves to get signatures.

As shown in the Figure 2, we attach the internal node with the aggregation signature of its descendants marked as the black diamonds. By the aggregate information at the intermediate

nodes of the α-tree we can authenticate the query results without having to traverse the tree all the way down to the leaves to get the VO.

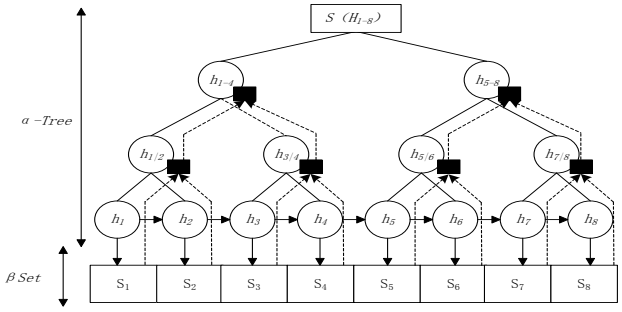

Figure 2. An Example of ITCDA

Here we first introduce the pre-existing concept of minimum covering set MCS [5]for the range query in the tree. The MCS is a set of nodes with disjoint subtrees whose leaves are the exact answer to the QP. Given a query Q with the lower(upper) bound $q_l(q_u)$. MCS(QP) can be computed by traversing the α-tree top-down and inserting all nodes contained in $[q_l, q_u]$ while its ancestors are not in MCS(QP). A node with its key value intersects with $[q_l, q_u]$ and its children key are contained in $[q_l, q_u]$, the signature corresponding to the node is put into the MCS(QP) , else downward to the next level. A node with its key value does not intersect with $[q_l, q_u]$ is ignored. Here, we discuss the algorithm for retrieving MCS(QP) in one pass of the tree. The VO construction algorithm is shown in Table 1.

Table 1. VO Construction Algorithm

```
Algorithm 1ITCDAVO(Query Q; α-tree T ;Signature S)
Begin
Compute [q_l, q_u] from Q
Rangeα (T.root, VO, [q_l, q_u])
Push information for verifying q_l-, q_u+ into VO
// in the α tree
Rangeα (Node N, range R)
Begin
VO.push(node start);
If N intersects with R
    If  N.leftchildren and N. rightchildren contained in R then
        VO.push(N.S)
     If  N is leaf node VO.push(N.S)
        Rangeα (N.leftchildren,  R1)
        Rangeα (N.rightchildren,  R2)
End
```

During the authentication, IPM first indicates the boundary entries, and with the returned VO, the client can reconstruct the hash value of the root node and verify it against the signature of α tree. If it fails, the client rejects the answer. Otherwise the IPM checks the aggregated signature from MCS(QP), to ensure the correctness and completeness of ANS(Q).

6. Experiment

We make a simulation experiment to demonstrate our analysis of the multi-tenant data authentication scheme approaches. The development environment is Eclipse-SDK-4.3.1-win 64 Bit, operating system is Windows XP Professional Service Pack 3, CPU is Inter Core

(TM)2 2,33GHz, and the memory is 2G. We utilize RSA signatures that are typically 128 bytes in size and SHA-1 with 20-byte outputs.

In our experiment, we set up a Tenants T_{01} with data set 10k records, while T_{01} specify A_1 as the searching key and stores $T_{01\text{-}A1}$ into pivot table. We compare our models TCDA and ITCDA with the case (1)that builds separate MHTs on sparse table and pivot table with searching key <TenantID,A_1> and case(2) builds separate DSAC on sparse table and pivot table with searching key <TenantID,A_1>. First we test the initial set up cost of MHT,TCDA, ITCDA and DSAC, as shown in Figure 3. We can see that MHT approach has the minimal set up cost on spares table and pivot table, while ITCDA has the maximum because he has to attach the aggregation signature into his internal node.

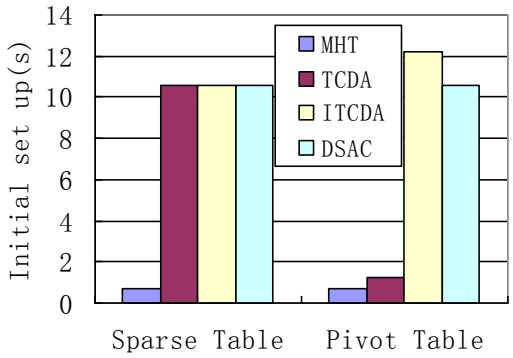

Figure 3. Initial Set Up Cost

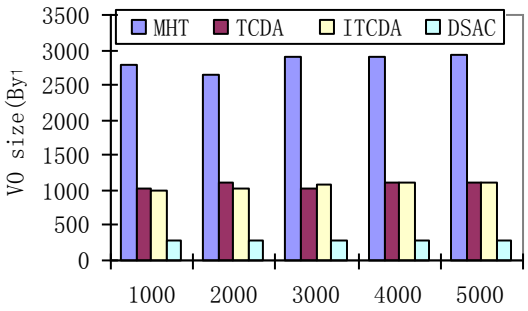

Figure 4. The Query Performances Influence

We can see that the TCDA and ITCDA have a high time consuming at the initial set up phase on sparse table, because they need to sign every record of the tenant in the sparse table. However, they show a better performance on VO construction, shown in Figure 4. From the experiment result MHT approach have to set up separate MHTs on sparse table and pivot table which leads to double travels of tree and relative larger VO size of the sub path nodes, while in TCDA and ITCDA, they only need to travel the tree once and combine with one aggregated signature to verify tenants data. From Figure 4, we also found that DSAC approach has the minimal VO size because he only needs to return two aggregated signature and the boundary records.

Figure 5 shows the VO construction time. Since the DSAC and TCDA needs to access every record signature, they have a high time consumption to construct the VO, while the ITCDA have a much lower time consumption compared with the others. Because ITCDA attaches aggregated signature into its internal node and does not need to access every record signature until the leaf nodes. Figure 6 gives the verification time consumption. From all the upper experiment, we can see TCDA and ITCDA show better efficiency on VO size, verification and VO construction time compared with MHT and DSAC.

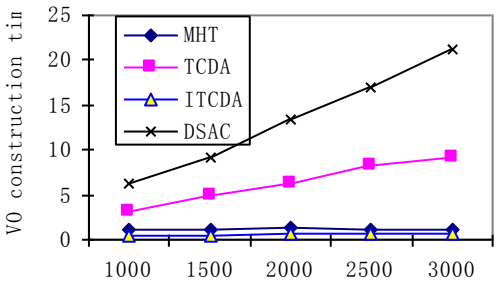

Figure 5. VO Construction Time

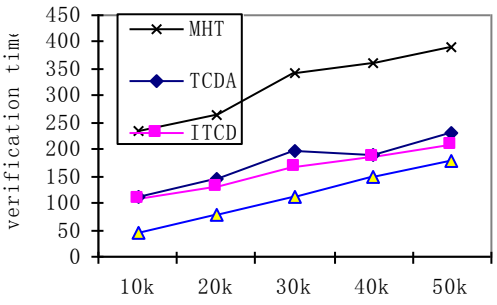

Figure 6. Verification Time

7. Conclusion

In this paper, we put forward a multi-tenant data authentication model TCDA. TCDA can accommodate the multi-tenant properties perfectly by taking tenant identification into account and establishing isolated authentication structures for each tenant based on their integrity demands. And we give a improve TCDA to get a better performance. Besides, there remains some problem of TCDA for the future work such as the tenant dynamic data operation and multiple attribution query authentication. And beyond those how to combine data integrity with data privacy in SaaS is a challenging problems which remains later to solve.

Acknowledgement

This work is supported by National Key Technologies R&D Program No.2012BAH54F01; National Natural Science Foundation of China under Grant No.61272241,No.61303085; Natural Science Foundation of Shandong Province of China under Grant No.ZR2013 FQ014; Science and Technology Development Plan Project of Shandong Province No. 2012GGX10134; Independent Innovation Foundation of Shandong University under Grant No.2012TS075, No.2012TS074; Shandong Province Independent Innovation Major Special Project No.2013CXC30201.

References

[1] wikipedia. Software as a service. http://en.wikipedia.org/wiki/Software_as_a_service.
[2] E. Mykletun, M. Narasimha and G. Tsudik, “Authentication and integrity in outsourced databases”, TOS, vol. 2, no. 2, **(2006)**, pp. 107-138.
[3] S. Papadopoulos, Y. Yang and D. Papadias, “Continuous authentication on relational streams”, VLDB J. (VLDB), vol. 19, no. 2, **(2010)**, pp. 161-180.
[4] P. T. Devanbu, M. Gertz, C. U. Martel and S. G. Stubblebine, “Authentic Third-party Data Publication”, DBSec, **(2000)**, pp. 101-112.
[5] F. Li, M. Hadjieleftheriou, G. Kollios and L. Reyzin, “Dynamic authenticated index structures for outsourced databases”, SIGMOD, **(2006)**, pp. 121-132.
[6] S. Aulbach, T. Grust, D. Jacobs, A. Kemper and J. Rittinger, “Multi-Tenant Databases for Software as a Service: Schema-Mapping Techniques”, SIGMOD, **(2008)**.
[7] C. U. Martel, G. Nuckolls, P. T. Devanbu, M. Gertz, A. Kwong and S. G. Stubblebine, “A general model for authenticated data structures”, Algorithmica, vol. 39, no. 1, **(2004)**, pp. 21-41.
[8] H. H. Pang, J. Zhang and K. Mouratidis, “Scalable Verification for Outsourced Dynamic Databases”, PVLDB, vol. 2, no. 1, **(2009)**, pp. 802-813.
[9] C. D Weissman and S. Bobrowski, “The Design of the Force.com Multitenant Internet Application Development Platform”, SIGMOD, **(2009)**.
[10] H. H. Pang and K.-L. Tan, “Authenticating Query Results in Edge Computing”, ICDE, **(2004)**, pp. 560-571.

[11] M. Narasimha and G. Tsudik, "Authentication of Outsourced Databases Using Signature Aggregation and Chaining", DASFAA, **(2006)**, pp. 420-436.
[12] K. Mouratidis, D. Sacharidis, and H. Pang, "Partially Materialized Digest Scheme: An Efficient Verification Method for Outsourced Databases", International Journal on Very Large Data Bases, vol. 18, no. 1, **(2009)**, pp. 363-381.
[13] W. Wei, T. Yu and R. Xue, "iBigTable: practical data integrity for bigtable in public cloud", CODASPY, **(2013)**, pp. 341-352.
[14] C. Wang, Q. Wang, K. Ren and W. Lou, "Ensuring data storage security in cloud computing", 17th IEEE International Workshop on Quality of Service (IWQoS 2009), **(2009)**, pp. 1-9.
[15] M. Xie, H. Wang, J. Yin and X. Meng, "Integrity Auditing of Outsourced Data", Proceedings of the 33rd International Conference on Very Large Data Bases (VLDB 2007), **(2007)**, pp. 782-793.
[16] H. Wang, J. Yin, C. Perng and P. Yu, "Dual encryption for query integrity assurance", Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM 2008), **(2008)**, pp. 863-872.
[17] A. Juels and B. Kaliski Jr, "PORs: proofs of retrievability for large files", Proceedings of the 2007 ACM Conference on Computer and Communications Security (CCS 2007), **(2007)**, pp. 584-597.
[18] H. Liu, P. Zhang and J. Liu, "Public Data Integrity Verification for Secure Cloud Storage", JNW, vol. 8, no. 2, **(2013)**, pp. 373-380.
[19] Y. Shi, K. Zhang and Q. Li, "Meta-data Driven Data Chunk Based Secure Data Storage for SaaS", JDCTA: International Journal of Digital Content Technology and its Applications, vol. 5, no. 1, **(2011)**, pp. 173-185.
[20] A. Brown and J. S. Chase, "Trusted platform-as-a-service: a foundation for trustworthy cloud-hosted applications", CCSW, pp. 15-20, **(2011)**.

Authors

Li Lin, Born in 1981, PHD candidate. She is learning in Shandong University and Shandong Provincial Key Laboratory of Software Engineering. Her main research interests include cloud security and privacy, cloud data management.

Li Qingzhong, Born in 1965, professor, Ph.D. supervisor. His research interests include cloud computing, large-scale network data management and web data integration.

Lanju Kong, Born in 1978, Ph.D. Her main research interests include computer software and theory, software and data engineering, XML query and access.

Shi Yuliang, Born in 1978, Ph.D. His main research interests include cloud computing, software and data engineering, cloud security and privacy.