International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
International Conference on Humming Bird (01st March 2014)

RESEARCH ARTICLE OPEN ACCESS

Secure Data Retrieval over Graph Structured Data in Cloud

Ms.Jelba.J, PG scholar,
Department of Computer Science and Engineering,
University College of Engineering, Nagercoil,
India.
jelbaj@ymail.com

Dr.V.Kavitha, M.E., Ph.D.,
Dean,
University College of Engineering, Nagercoil,
India.

Abstract:
The main advantage of cloud computing is increasingly attracting individuals and organizations to outsource their data from local to remote cloud servers. For protecting data privacy, sensitive data has to be encrypted before outsourcing, it is a very challenging task. Here we define and solve the problem of privacy-preserving query over encrypted graph-structured data (PPGQ) and establish a set of privacy requirements for such a secure cloud data utilization in cloud computing. By utilizing the principle of filtering-and-verification, first pre-build a feature-based index to provide feature-related information about each encrypted data graph, and then choose the efficient inner product as the pruning tool to carry out the filtering procedure. To meet the challenge of supporting graph query without privacy breaches, propose a secure inner product computation technique, and then improve it to achieve various privacy requirements under the known-background threat model.

I. INTRODUCTION

In the increasingly prevalent cloud computing, datacenters play a fundamental role as the major cloud infrastructure providers [1], such as Amazon, Google, and Microsoft Azure. Datacenters provide the utility computing service to software service providers who further provide the application service to end users through Internet. The later service has long been called "Software as a Service (SaaS)", and the former service has recently been called "Infrastructure as a Service (IaaS)", where the software service provider is also referred to as cloud service provider. To take advantage of computing and storage resources provided by cloud infrastructure providers, data owners outsource more and more data to the datacenters [2] through cloud service providers, e.g., the online storage service provider, which are not fully trusted by data owners.

As a general data structure to describe the relation between entities, the graph has been increasingly used to model complicated structures and schema less data, such as the personal social network (the social graph), the relational database, XML documents and chemical compounds studied by research labs [3]–[8]. Images in the personal album can also be modeled as the attributed relational graph (ARG) [9]. For the protection of users' privacy, these sensitive data have to be encrypted before outsourcing to the cloud.

With the conventional graph data utilization method, we first take the query graph as an input, and then perform the graph containment query: given a query graph as Q and a collection of data graphs as $\mathcal{G}$ = ($G1,2, \ldots, Gm$), find all the supergraphs of Q in $\mathcal{G}$, denoted as $\mathcal{G}Q$. The straightforward solution is to check whether Q is subgraph isomorphic to every Gi in $\mathcal{G}$ or not. However, checking subgraph isomorphism is NP-complete, and therefore it is infeasible to employ such costly solution. To efficiently solve the graph containment query problem, there have been a lot of proposed techniques [3]–[8], most of which follow the principle of "filtering-and-verification". In the filtering phase, a pre-built feature-based index is utilized to prune as many data graphs from the dataset as possible and output the candidate supergraph set. Every feature in the index is a fragment of a data graph, e.g., the subgraph. In the verification phase, each candidate supergraph is verified by checking subgraph isomorphism. Since the candidate supergraph set is much smaller than the entire dataset, such approach involves less subgraph isomorphism checking, and therefore is significantly more efficient than the straightforward solution. However, when data graphs are stored in the encrypted form in the cloud, the encryption excludes the filtering method which is based on the plaintext index.

we define and solve the problem of privacy-preserving graph query in cloud computing (PPGQ). To reduce the times of checking subgraph isomorphism, we adopt the efficient principle of "filtering-andverification" to prune as many negative

data graphs as possible before verification. A feature-based index is firstly built to provide feature-related information about every encrypted data graph. Then, we choose the efficient inner product as the pruning tool to carry out the filtering procedure. To achieve this functionality in index construction, each data graph is associated with a binary vector as a subindex where each bit represents whether the corresponding feature is subgraph isomorphic to this data graph or not. The query graph is also described as a binary vector where each bit means whether the corresponding feature is contained in this query graph or not. The inner product of the query vector and the data vector could exactly measure the number of query features contained in the data graph, which is used to filter negative data graphs that do not contain the query graph. However, directly outsourcing the data vector or the query vector will violate the index privacy or the query privacy. To meet the challenge of supporting graph semantics without privacy breaches, we propose a secure inner product computation mechanism, which is adapted from a secure k-nearest neighbor (*kNN*) technique [15], and then show our improvements on it to achieve various privacy requirements under the known-background threat model. Our contributions are summarized as follows,

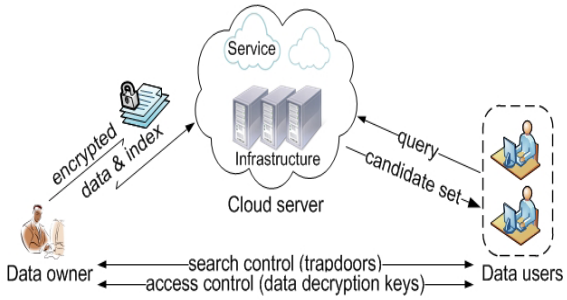
1) For the first time, we explore the problem of query over encrypted graph-structured data in cloud computing, and establish a set of strict privacy requirements for such a secure cloud data utilization system to become a reality.

2) Our proposed scheme follows the principle of "filteringand-verification" for efficiency consideration, and thorough analysis investigating privacy and efficiency guarantees of the proposed scheme is given.

3) The evaluation, which is performed with the widely-used AIDS antiviral screen dataset on the Amazon EC2 cloud infrastructure, further shows our proposed scheme introduces low computation and communication overhead.

II. THE SYSTEM MODEL

Considering a cloud data storage service, involving four different entities: the data owner, the data user, the storage service provider/cloud service provider, and the datacenter/cloud infrastructure provider. To take advantage of the utility computing services provided by the datacenter,



e.g., computing and storage resources, the storage service provider deploys its storage service on top of the utility computing in datacenter and delivers the service to end users (including data owners and data users) through Internet. In our system model, neither cloud service provider nor cloud infrastructure provider is fully trusted by data owners or data users, so they are treated as an integrated entity, named the cloud server, as shown in Fig. 1. The data owner has a graph-structured dataset $\mathcal{G}$ to be outsourced to the cloud server in the encrypted form $\tilde{\mathcal{G}}$. To enable the query capability over $\tilde{\mathcal{G}}$ for effective data utilization, the data owner will build an encrypted searchable index $\mathcal{I}$ from $\mathcal{G}$ before data outsourcing, and then both the index $\mathcal{I}$ and the encrypted graph dataset $\tilde{\mathcal{G}}$ are outsourced to the cloud server. For every query graph Q, an authorized user acquires a corresponding trapdoor TQ through the search control mechanism, e.g., broadcast encryption [10], and then sends it to the cloud server. Upon receiving TQ from data users, the cloud server is responsible to perform query over the encrypted index $\mathcal{I}$ and return the encrypted candidate supergraphs. Finally, data users decrypt the candidate supergraphs through the access control mechanism, and verify each candidate by checking subgraph isomorphism.

III. DESIGN GOALS

To enable the graph query for the effective utilization of outsourced cloud data under the aforementioned model, our design should simultaneously achieve security and performance guarantees.

Effectiveness: To design a graph query scheme that introduces few false positives in the candidate supergraph set.

Privacy: To prevent the cloud server from learning additional information over outsourced data and index in query interactions·

Efficiency: Above goals on effectiveness and privacy should be achieved with low communication and computation overhead.

IV. THE FRAMEWORK

Our proposed framework focuses on how the query works with the help of index which is

outsourced to the cloud server. We do not illustrate how the data itself is encrypted, outsourced or accessed, as this is a complementary and orthogonal issue and has been studied elsewhere [16]. The framework of PPGQ is illustrated as follows.

FSCon($\mathcal{G}$, σ):

Takes the graph dataset $\mathcal{G}$ and the minimum support σ as inputs, outputs a frequent feature set $\mathcal{F}$.

KeyGen(ξ):

Takes a secret ξ as input and outputs a symmetric key $\mathcal{K}$.

BuildIndex($\mathcal{G}$,$\mathcal{K}$):

Takes the graph dataset $\mathcal{G}$ and the symmetric key $\mathcal{K}$ as inputs, output a searchable index $\mathcal{I}$.

TDGen(Q,$\mathcal{K}$):

Takes the query graph Q and the symmetric key $\mathcal{K}$ as inputs, outputs a corresponding trapdoor TQ.

Query(TQ, $\mathcal{I}$):

Takes the trapdoor TQ and the searchable index $\mathcal{I}$ as inputs, returns $\mathcal{GF}Q$, i.e., the candidate supergraphs of query graph Q.

The first three algorithms, i.e., FSCon, BuildIndex, and BuildIndex, are run by the data owner as pre-processes. The query algorithm is run on the cloud server as a part of the cloud data storage service. According to various search control mechanisms, the trapdoor generation algorithm TDGen may be run by either the data owner or the data user. Besides, depending on some specific application scenarios, while search requests on confidential documents may be allowed for all users, the access to document contents may be forbidden for those low-priority data users.

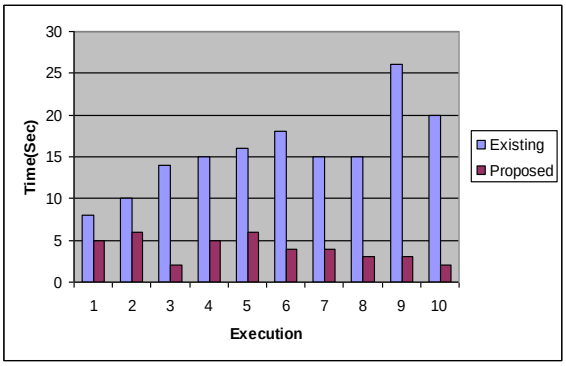
V. PRIVACY REQUIREMENTS

1) Index Privacy: With respect to the index privacy, if the cloud server deduces any association between frequent features and encrypted dataset from outsourced index, it may learn the major structure of a graph, or even the entire topology of a small graph. Therefore, searchable index should be constructed in such a way that prevents the cloud server from performing such kind of association attack.

2) Feature Privacy: Data users usually prefer to keep their query from being exposed to others like the cloud server, and the most important concern is to hide what they are querying, i.e., the features indicated by the corresponding trapdoor. Although trapdoor can be generated in a cryptographic way to protect the query features, the cloud server may do some statistical analysis over the search results to make an estimate. Especially, the feature support (i.e., the number of data graphs containing the feature), a kind of statistical information, is sufficient to identify the feature with high probability. When the cloud server knows some background information of the dataset, this feature-specific information can be utilized to reverse-engineer the feature. As presented in Fig. 2, the distribution of feature support in the AIDS antiviral screen dataset [22] provides enough information to identify most frequent features in the dataset. Such problem is similar with the keyword privacy issue in [23], where document frequency (the number of documents containing the keyword) is used as a statistical information to reverse-engineer the keyword.

VII PERFORMANCE ANALYSIS

A solution to Query Processing Over Graph Structured Out sourced Data describe secure data retrieval processing . The data owner has stored all the data in graph . He use two encryption methods for graph data and attributes, then he outsourced the data in the cloud. The graph which does not contain data is neglected by filtering procedure which is done by inner product computation tool. By this method the computation time and memory space is reduced. Then the authorized data consumer can retrieve the data in secure manner. In our proposed system, we use high Encryption methods and filtering procedure so the efficiency of this system is high.



Execution Overhead

VIII CONCLUSION

Here I define and solve the problem of query over encrypted graph-structured cloud data, and establish a variety of privacy requirements. For the efficiency consideration, we adopt the principle of "filtering-and verification" to prune as many negative data graphs as possible before verification, where a feature-based index is pre-built to provide feature-related information for every encrypted data graph. Then, we choose the inner product as the pruning tool to carry out the filtering procedure efficiently. To meet the challenge of supporting

graph semantics, we propose a secure inner product computation, and then improve it to achieve various privacy requirements under the known-background threat model. Thorough analysis investigating privacy and efficiency of our scheme is given, and the evaluation further shows our scheme introduces low overhead on computation and communication. As our future work, we will explore
privacy-preserving schemes under stronger threat models.

REFERENCES:

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," University of California, Berkeley, Tech. Rep. UCB-EECS-2009-28, Feb 2009.

[2] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *RLCPS, January 2010, LNCS. Springer, Heidelberg*.

[3] D. Shasha, J.-L. Wang, and R. Giugno, "Algorithmics and applications of tree and graph searching," in *Proc. of PODS*, 2002.

[4] X. Yan, P. S. Yu, and J. Han, "Graph indexing: a frequent structurebased approach," in *Proc. of SIGMOD*, 2004.

[5] P. Zhao, J. X. Yu, and P. S. Yu, "Graph indexing: tree + delta >= graph," in *Proc. of VLDB*, 2007.

[6] S. Zhang, M. Hu, and J. Yang, "Treepi: A novel graph indexing method," in *Proc. of ICDE*, 2007.

[7] J. Cheng, Y. Ke, W. Ng, and A. Lu, "Fg-index: towards verification-free query processing on graph databases," in *Proc. of SIGMOD*, 2007.

[8] H. Shang, Y. Zhang, X. Lin, and J. X. Yu, "Taming verification hardness: an efficient algorithm for testing subgraph isomorphism," in *Proc. Of VLDB*, 2008.

[9] S. Berreti, A. Bimbo, and E. Vicario, "Efficient matching and indexing of graph models in content-based retrieval," *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2001.

[10] R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," in *Proc. of ACM CCS*, 2006.

[11] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. of EUROCRYPT*, 2004.

[12] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in *Proc. of ICDCS*, 2010.

[13] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *Proc. of IEEE INFOCOM'10 Mini-Conference*, San Diego, CA, USA, March 2010.

[14] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," in *Proc. Of INFOCOM*, 2011.

[15] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases," in *Proc. of SIGMOD*, 2009.

[16] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Proc. Of INFOCOM*, 2010.

[17] S. Zerr, D. Olmedilla, W. Nejdl, and W. Siberski, "Zerber+r: Top-k retrieval from a confidential index," in *Proc. of EDBT*, 2009.

[18] M. R. Garey and D. S. Johnson, "Computers and intractability: A guide to the theory of np-completeness," Freeman, New York, NY, USA, 1990.

[19] X. Yan and J. Han, "gspan: Graph-based substructure pattern mining," in *Proc. of ICDM*, 2002.

[20] S. Nijssen and J. N. Kok, "A quickstart in frequent structure mining can make a difference," in *Proc. of SIGKDD*, 2004.

[21] J. Huan, W. Wang, and J. Prins, "Efficient mining of frequent subgraphs in the presence of isomorphism," in *Proc. of ICDM*, 2003.

[22] "Aids antiviral screen dataset," 1999, http://dtp.nci.nih.gov/docs/aids/ aids data.html.

[23] S. Zerr, E. Demidova, D. Olmedilla, W. Nejdl, M. Winslett, and S. Mitra, "Zerber: r-confidential indexing for distributed documents," in *Proc. Of EDBT*, 2008, pp. 287–298.

[24] R. Brinkman, "Searching in encrypted data," in *University of Twente,PhD thesis*, 2007.

[25] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Cryptography from anonymity," in *Proc. of FOCS*, 2006, pp. 239–248.

[26] H. Saigo, S. Nowozin, T. Kadowaki, T. Kudo, and K. Tsuda, "Gboost: A mathematical programming approach to graph classification and regression," in *Machine Learning*, 2008.

[27] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proc. of TCC*, 2007, pp. 535–554.

[28] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *Proc. of EUROCRYPT*, 2010.