SEARCH TECHNIQUE ON ENCRYPTED CLOUD DATA FOR TOP-K NEAREST KEYWORD USING GRAPH ENCRYPTION

A.SASI HIMABINDU, ASSISTANT PROFESSOR, BHIMAVARAM INSTITUTE OF ENGINEERING AND TECHNOLOGY-PENNADA,

D.HIMAJA, ASSISTANT PROFESSOR, BHIMAVARAM INSTITUTE OF ENGINEERING AND TECHNOLOGY-PENNADA.

ABSTRACT:

With the tremendous increase in security demands of data outsourcing applications in metropolitan smart cities, encrypting client's data has become a major issue for academia and industry. As the clouds and edges cannot be trusted, encryption should be done at the client side before outsourcing. Therefore the process of encrypting data in such a way that the encrypted and remotely stored data can still be queried has become a challenging issue. A lot of research has been made on keyword searches over encrypted textual data. The approach for encrypting graph-structured data capable of answering graph queries is still lacking.

This paper discusses about Graph encryption method for a graph query type called top-k-nearest keyword (Top-k nearest key word) searches. Several indexes are designed to store necessary information for answering queries while securing the private information about graph such as vertex identifiers, edges and keywords are encrypted and excluded.

INTRODUCTION:

Data owners are freeing themselves from maintaining IT infrastructure and data management related things by outsourcing data with the help of cloud computing and edge computing technologies. Security of clouds have become a major challenge in the present scenario. Privacy risks at the edge side are greater compared to that of cloud. In order to protect data privacy, data owners should encrypt data before outsourcing .The traditional data encryption technique leaves the data in non-query able state leading to severe impact on data usability. In spite of all the efforts made to enable keyword search on encrypted data, the process of querying an encrypted graph structured data is still a challenging problem.

The Top-k nearest key word search solves the problem to some extent because of its important applications in graph. Top-k nearest keyword search involves a graph G = (V, E) in which each vertex v < V is labeled with a range of keywords .On given input (k, v, w) Top-k nearest keyword search K vertices labeled with keyword w, and are nearest to V. For

example, in a social network consider three inputs given respectively,"k=7 most closely related persons to vertex V='rahul' with interest keyword w='cricket'". Top-k nearest keyword search returns the name of 7 persons who play cricket and those are close to rahul. Our matter of concern in this paper is to provide secure data outsourcing using encrypted graphs and answer Top-k nearest keyword search queries in a secure manner.

When performing Top-k nearest keyword queries on encrypted graph, there is a possibility of privacy information being leaked from both graph and query. Privacy can be maintained to some extent by hiding vertex identifiers such as email addresses, full names or phone numbers in real usage. For a query (k, v, w) the identifier of v and the contents of keyword w should be hidden. To attain these requirements, the encryption method for the graph should be specially defined. Applications of traditional cryptographic encryption tools such as AES to encrypt an entire graph may avoid information leakage but it makes the encrypted graph impossible to be queried. Our approach partially encrypts a graph by encrypting only vertex identifiers and keywords.

OUR DEVELOPMENTS:

This paper describes the following developments

- This paper presents Security model and Graph encryption scheme which supports Top-k nearest keyword queries.
- An effective and secure graph encryption Scheme is designed.
- We evaluate the performance of our Graph encryption Scheme on the real-world graphs also which produced a positive result.

RELATED WORK:

1. The first SSE (Searchable symmetric encryption) was proposed by song et al. They did not offer a good security model for SSE.

2. The above issue was addressed by Curtmola et al. He told that SSE scheme should provide security over adaptive chosen-keyword attacks (CKA).

3. Querying on structure data was first started by Chase and Kamara. For structured encryption they proposed notion of structured encryption scheme and extended CKA of SSE into chosen-query attacks (CQA).Chase and Kamara also proposed several structured encryption techniques such as matrix-structured data encryption scheme answering look-up queries, labeled data encryption scheme answering keyword search queries, graph encryption scheme answering neighbor queries, adjacency queries, focused sub graph queries etc…For higher level queries graph encryption scheme designing has become an challenging task.

4. To overcome the above drawback Meng, Kamara and Nissim has come up with a graph encryption scheme foe approximate shortest distance queries.

5. There are several other techniques available for graph encryption such as graph

structure anonymization method to hide neighborhood information.

The above mentioned works are feasible on their certain applications but, failed to address fully our data outsourcing problem.

SYSTEM MODEL:

Graph:

A graph G is a tuple G= (V,W) where V is a dictionary which stores all the information of its neighbors and W is a dictionary which stores all the information about keywords associated with vertex v.

Graph Encryption Scheme:

Our scheme comprises of five algorithms. They are as follows:

1. Key Generation Algorithm:

K← keyGen (λ), takes security parameter λ as input and produces a secret key K.

2. Encryption Algorithm:

I←Encrypt (K, G), takes secret key K and graph G as input and produces a secure index structure I as output.

3. Token Generation Algorithm:

T← TokenGen (K, k, v, w), takes secret key K, an integer k, a vertex v and a keyword w as input and generates a token T.

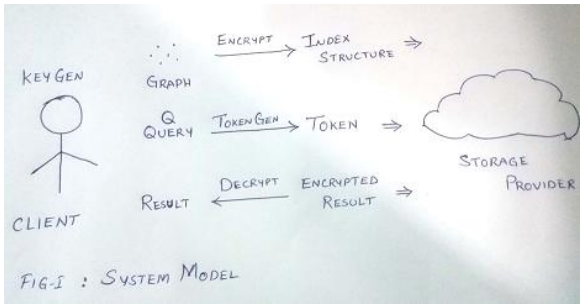
4. Answer Algorithm:

R← Answer (I, T), takes index I and token T as input and gives R as encrypted result.

5. Decryption Algorithm:

S← Decrypt (K, R), takes secret key K, encrypted result R as input and produces a set of k vertex identifiers S.

A graph encryption technique involves a client C and a database D.C has a graph G to be forwarded, and D will store encrypted form of G and answers C's top k nearest keyword queries. There are two stages in this system model. They are Setup Protocol and Query Protocol. In Setup protocol, C makes use of Graph encryption scheme to encrypt G, and outsources the encrypted form of G to D. In Query protocol C issues top k nearest keyword query tokens. For each query, D returns an encrypted list of k graph vertices. To decrypt that we have to provide a secret key such that we will get decrypted documents. The following figure illustrates this.



ALGORITHMS:

Consider a graph G = (V, E) containing V vertices each vertex 'v' contains the vertex identifiers which are to be kept secured and associated keyword 'w'. The aim of our algorithm is to encrypt the graph in such a way that even after encryption it should be able to answer queries such as Top-k nearest vertices to 'v' without displaying the vertex identifiers and the linkage between them. In

order to attain these total graph is encrypted in two phases with two different algorithms each.

Initially 3 Random Secret Keys are generated K1, K2, K3 by using KeyGen algorithm. Now in order to encrypt, two algorithms are used. The first round of algorithm uses DET and it takes K2 & V as input and provides enc_v which is an encrypted form of v which hides vertex identifiers. The second round of algorithm uses SKE which takes key_w and enc_v as input and provides enc_{encv} as output where key_w is generated by the Random function RF.

Reason for using two rounds of encryption:

The first round encryption makes use of DET rather than SKE because DET is used to hide the vertex identifier and we need 'D' to link the cipher text of same vertex without decrypting it. Thus we make use of DET to ensure that same plain text results in same cipher text. In order to hide linkages from 'D' before C's queries, we have added another round of encryption SKE

The first step of algorithm starts with Key Generation which can be illustrated by following algorithm.

KEY GENERATION:

This algorithm generates three Random Secret Keys K1, K2 and K3, whose length is decided by 'λ' which is a security parameter. The keys K1 and K2 are used in RF and DET where as K2 is used in OPE scheme.

Algorithm : KeyGen

Input : ' λ' a security parameter

Output: Secret Key K1, K2, K3

1. Generate Random Key K1.
   A string with 0's & 1's with length 'λ'.
2. Generate Random Key K2.
   A string with 0's & 1's with length 'λ'.
3. K3 is an integer key

The second step is generation of secured index structure I .I can be generated by using KeyWord IndexGen Algorithm.

Algorithm : KeyWordIndexGeneration

Input K, G: Secret Key, Graph

Output: I a Secure Index

1. Parse Graph G (V, W).
2. Parse Secret Key K (K1, K2 ,K3).
3. Initialize Dictionary I
4. For w ϵ W repeat steps 5 to 12.
5. $lab_w \leftarrow RF(K1,w||'lab')$
6. $key_w \leftarrow RF(K1,w||'key')$
7. Initialize list l_w
8. v ϵ W[w] repeat steps 9,10,11.
9. $enc_v \leftarrow DET.Enc(K2, v)$.
10. $enc_{encv} \leftarrow SKE.Enc(key_w, enc_v)$.
11. add enc_{encv} into l_w.
12. add (lab_w, enc_{encv}) into I.
13. Return I.
14. Stop.

By using the Encrypted KeyWordIndex we can identify 'k' nearest vertices by using 2-Hop labeling Technique. Information in the cloud database can be attained by posing different queries using Tokens.

Token Generation Algorithm:

A Token is generated with every query by using TokenGen algorithm. This outputs several Tokens enabling D to correctly locate the entries of indexes and decrypt needed information.

Algorithm : TokenGen

Input K, k , v ,w: Secret Key ,an integer ,a vertex identifier ,and a Keyword

Output: a query Token

1. Parse K as (K1, K2, K3)
2. $lab_w \leftarrow RF(K1,W||'lab')$
3. $key_w \leftarrow RF(K1,W||'key')$
4. $lab_v \leftarrow RF(K1,v||'lab')$
5. $key_v \leftarrow RF(K1,v||'key')$
6. $enc_v \leftarrow DET.Enc(K2,v)$
7. Return $T \leftarrow k, lab_w, key_w, lab_v, key_v$

Answer algorithm:

Our answer algorithm uses the hybrid search Strategy as follows: If the keyword frequency is less than (threshold keyword frequency) then the Forward search is used otherwise Backward search is used. This algorithm takes Index structure I and Token T, integer threshold t as input and produces an encrypted result.

Algorithm : Answer(Hybrid Search)

Input I, T, t: An index structure , a query token and an integer threshold

Output R: An encrypted Result.

1. Parse I as (D1, D2, D3, D4)
2. Parse T as $k, lab_w, key_w, lab_v, key_v, enc_v)$
3. $L_w \leftarrow D1[lab_w]$
4. If $|L_w| < t$ then
5. Perform Forward traversal
6. Else
7. Perform Backward traversal
8. If $|R| < k$ then

Decrypt Algorithms:

The decryption algorithm at the client side is simply decrypting the first round encryption for all vertices returned from D.

Algorithm :Decrypt

Input K, R: A Secret Key, an encrypted result

Output S: a plain text result

1. Parse K as (K1 , K2 ,K3).
2. Initialize a list S
3. For $enc_v \in R$ do
   - $V \leftarrow DET.dec(K2,enc_v)$
   - Add v into S
4. Return S

Conclusion:

In this paper, we present a graph encryption scheme for Top –K nearest vertices queries. The proposed graph encryption scheme only makes use of lightweight cryptographic primitives such as pseudorandom function and symmetric-key encryption, rather than slow homomorphic encryptions. Therefore, the proposed graph encryption scheme is friendly to a wide set of graph data based cloud computing and edge computing applications such as social networks, e-maps, criminal analyses, etc. Compare to graph anonymization approaches from database community, our scheme attains higher security level as the graph itself is encrypted and we do not make any assumptions on the types of attacks.

References:

1] I. Abraham, D. Delling, A. V. Goldberg, and R. F. Werneck. Hierarchical hub labelings for shortest paths. In Algorithms [–ESA, pages 24–35. Springer, 2012.
[2] R. Agarwal, P. Godfrey, and S. Har-Peled. Approximate distance queries and compact routing in sparse graphs. In IEEE INFOCOM,
[3] T. Akiba, Y. Iwata, and Y. Yoshida. Fast exact shortest-path distance queries on large networks by pruned landmark labeling. In ACM SIGMOD, pages 349–360, 2013.
[4] B. Bahmani and A. Goel. Partitioned multi-indexing: bringing order to social search. In WWW, pages 399–408, 2012.
[5] J. Blocki, A. Blum, A. Datta, and O. Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In ACM ITCS, pages 87–96, 2013.
[6] D. Cash, J. Jaeger, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Rosu, and M. Steiner. Dynamic searchable encryption in very large databases: Data structures and implementation. In NDSS, 2014.
[7] D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Ros¸u, and M. Steiner. Highly-scalable searchable symmetric encryption with support for boolean queries. In CRYPTO, pages 353–373. Springer,
8] V. Chang, Y.-H. Kuo, and M. Ramachandran. Cloud computing adoption framework: A security framework for business clouds. Future Generation Computer Systems, 57:24–41, 2016.
[9] M. Chase and S. Kamara. Structured encryption and controlled disclosure. In ASIACRYPT, pages 577–594. Springer, 2010.
[10] S. Chechik. Approximate distance oracles with constant query Time. In ACM STOC, pages 654–663, 2014.
[11] S. Chen and S. Zhou. Recursive mechanism: towards node differential privacy and unrestricted joins. In ACM SIGMOD,
[12] J. Cheng, A. W.-C. Fu, and J. Liu. K-isomorphism: privacy preserving Network publication against structural attacks. In ACM SIGMOD, pages 459–470, 2010.
[13] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: Improved definitions and efficient constructions. In ACM CCS, pages 79–88, 2006.