COMPUTATION AND SEARCH OVER ENCRYPTED DOCUMENTS

[1]Roshan Sahu, [2]Ameya Lotlikar, [3]Sanket Sarode, [4]Samarth Joshi, [5]Tabassum Maktum
[1,2,3,4] Computer Department, Mumbai University Terna Engineering College, Navi Mumbai, India
[5]Assistant Professor, Terna Engineering College, Navi Mumbai, India
[1]sroshan4020@gmail.com [2]ameyalotlikar1996@gmail.com [3]sarodesanket15@gmail.com
[4]samarth.au@gmail.com [5]tabsmaktum@gmail.com

***Abstract*— With the advent of cloud technology, the proliferation of cloud in various applications is enormous. Cloud can provide a variety of services. As major cloud service providers are public the data to be stored in cloud is at risk. A trivial solution would be keeping the files in encrypted form in the cloud, but whenever the file will be needed it will be downloaded and decrypted to get the data, but in a application which is user intensive this would lead to consumption of large bandwidth, hence it is not a feasible solution for providing a feasible computation and hence affecting the efficiency provided by the cloud. Therefore there is need for developing well planned solutions. In this work, we look into the problem of processing large amount of encrypted documents in XML formats. Here search or any computation operation can be performed on certain elements in the XML tree. The technique proposed makes use of index tables to allow fast addressing of keywords and location based queries. In order to perform computations on an untrusted server, symmetric encryption is used in along with homomorphic encryption to reduce computational and storage cost.**

***Index Terms*— Cloud, Homomorphic Encryption, Search, XML.**

I. Introduction

A. Cloud Computing

Cloud computing is a Internet-based computing that provides processing of shared data and resource to computers and other computing devices on demand [1]. This model enables omnipresent, on-demand access to a pool of shared, configurable computing resources (e.g., computer networks, storage, servers, applications and services), which can be rapidly provided and released with minimum management effort.

Cloud technologies promise great scalability and accessibility and provide various services like.

•Software as a Service (SaaS)

The capability provides the consumer to use the server's applications run on a cloud infrastructure. These applications can be accessed from various client devices.

•Infrastructure as a Service (IaaS)

The capability provide client to processing, storage, networks, and other fundamental computing resources where the client is able to install and run arbitrary software, which can be operating systems and applications.

•Platform as a Service (PaaS)

The capability provides the client to install on the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the server.

The common security issues of cloud computing can be divided into five main categories:

1) Data breaches

Cloud environments often face many of the same threats as usual corporate networks, but due to the large amount of data stored on cloud servers, service providers become an attractive target.

2) Shared Vulnerabilities

Cloud service providers share applications, infrastructure and platforms, and if a vulnerability is raised in any of these layers, it affects each and everyone in the hierarchy.

3) Cloud service abuses

Cloud services can be used to support criminal activities, such as using cloud computing resources or services to get an encryption key in attempt to make an attack.

4) Permanent data loss

Malicious hackers are known to delete cloud data permanently to harm businesses or organizations, and cloud data centres are as vulnerable to natural disasters as other facility.

5) Hacked interfaces and APIs

Interfaces and APIs are used to interact and manage cloud services, including those that offer cloud monitoring, management, orchestration, and provisioning. Weak interfaces and APIs expose business or organizations to security issues related to integrity, availability, confidentiality, and accountability.

B. XML Format

XML is a extensible markup language that specifies how documents encoded in simple and easy to use format [14]. Since its establishment, it has been widely used across the Internet and many variants, such as JSON (JavaScript Object Notation), have since been specified. A typical XML file includes tags, attributes and contents. A tag is a markup that begins with a ‘<’ and ends with ‘>’.Attributes are sometimes included in a tag to better describe the content. The following is an example of a XML file describing music in an album, where various tags such as title and artist are used and a number attribute describes ordering of the songs.

```
<ALBUM>
<SONG number="1">
<TITLE>Closer</TITLE>
<ARTIST>Chain smoker</ARTIST>
<PRICE>5.42</PRICE>
<YEAR>2016</YEAR>
</SONG>
...
</ALBUM>
```

C. Homomorphic Encryption

Homomorphic encryption is a type of encryption that allows carrying out computations on cipher text, hence generating an encrypted result, when decrypted; it matches with the result of operations performed on the initial plaintext [3].

This is sometimes a preferable feature in new communication system architectures. Homomorphic encryption would allow combining together different services without disclosing the data to each of those services. Designs of homomorphic encryption schemes are malleable. This enables use of these designs in cloud computing environment for ensuring the security aspect i.e. confidentiality of data processed. In addition to this homomorphic properties of different cryptosystems can be used to build many other secure systems. There are number of partially homomorphic and fully homomorphic crypto-systems. Although a crypto-system which is unintentionally malleable can have risk of attacks on this basis, if treated carefully homomorphism can also be used effectively to perform secure computations.

Example of additive Homomorphism: E (A) + E (B) = E (A + B)

Where E( ) is encryption function and A and B are plain text.

II. LITERATURE SURVEY

In [1] authors describes technique for encrypting and querying XML documents in a tree structure, While the scheme is reasonably efficient, it is limited to select type queries and the use of trees exposes the structure of the XML documents.

In [2] authors proposed a use of Elliptic curve cryptosystem to encrypt XML documents and provide Secrecy and integrity of document. Document is signed with digital signature. This system uses asymmetric encryption. Runtime complexity is affected.

In [3] authors investigated techniques to reduce the overhead in decrypting XML data. Both of these solutions considered only partially encrypted XML documents and are also limited to SELECT type queries. The problem of search over encrypted data is actively being investigated by many researchers.

In [4] authors proposed a solution involving the use of keyword searching using public key encryption (PEKS) which allows a set of keywords to be searchable without exposing the content of emails. Since it uses public key cryptography it is time consuming and has a huge overhead.

In [5] authors addressed the problem of searching over encrypted audit logs. While early works focused on single keyword searches, recent works have considered conjunctive keyword searches involving multiple keywords.

In [6] authors describe scheme to achieve high efficiencies in both computation and communication.

In [7] authors present a searchable encryption scheme that allows conjunctive keyword searches without specifying the position requiring only one search token with constant cipher text length.

In [8], [9] authors describe more advanced searching techniques such as phrase search.

In [10] authors investigate the fuzzy search over cloud data, then by using various filter technique, improve a efficiency of keyword search scheme to achieve fuzzy searching with low cost.

In [11] authors proposed a technique for secure approximate matching.

In [12] authors investigated the computation of scalar products.

In [13] authors describe data-mining techniques over encrypted data. But none proved to be efficient.

III. METHODOLOGY

A. System Architecture

The architecture model involves three parties: The data owner, the cloud server and the user. Figure 3.1 illustrates a standard protocol where the user initiates the request by sending the keywords (kwi) and the function, F(x), to the data owner.

The data owner then creates a trapdoor and sends it to the cloud.

A protocol to search over the requested keywords and function arguments are also sent to the cloud. If the user wants to perform search over encrypted data the algorithms described in Section III-B are used and if computations to be performed on the data, the homomorphic properties of Paillier Cryptosystem are used which is described in Section III-C and

Section III-D.
Finally, the results of the computations or search are returned to the user.

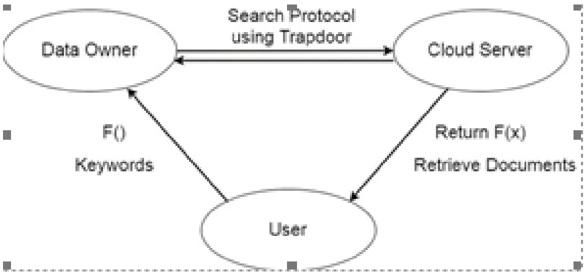


Figure 3.1: Architecture Model for Search and Computation over Encrypted data

To enable computations to be performed over encrypted data, homomorphic encryption is a natural choice. However, it is remarkably more expensive than symmetric encryption in both storage and computations. To benefit from its feature while minimizing its cost, we propose adapting the encryption scheme based on the content of the document. That is, only content which may be used for computations, assumed hereto be numeric data, are homomorphically encrypted, while remaining content are symmetrically encrypted.

B. Search over Encrypted Data

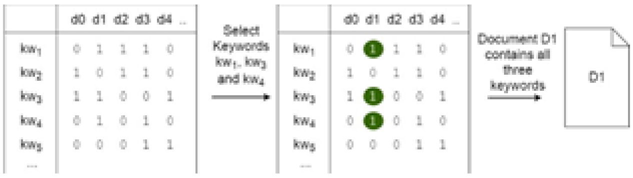
Basic Conjunctive Keyword Search Protocol:
We describe here a simple index based keyword search scheme. Given a document collection,

D = {D1, D2, . . . , Dn},

each document, Di, is parsed for a list of keywords, kwj. To generate an index entry,

I(kwj) = {da, db, . . . , dn}

mapping keywords to documents, we set each bit, di, to 1 if the keyword, kwj , is linked to the document, Di.
To perform a search, the user sends a set of keywords kw= {kw1, kw2, . . . , kwq} to the data owner. The data owner encrypts the search terms, EK(kw), and sends them to the cloud server.
The cloud server then locates and returns the encrypted index entries to the data owner. Finally, the data owner decrypts and finds the matching documents from the intersection of the index entries:
I(EK(kw1)) & I(EK(kw2))…….& I(EK(kwq) Where & denotes a bitwise AND operation.
Following diagram shows how the documents are searched from keywords.



	d0	d1	d2	d3	d4	..
kw_1	0	1	1	1	0	
kw_2	1	0	1	1	0	
kw_3	1	1	0	0	1	
kw_4	0	1	0	1	0	
kw_5	0	0	0	1	1	
...						

Select Keywords kw_1, kw_3 and kw_4

	d0	d1	d2	d3	d4	..
kw_1	0	1	1	1	0	
kw_2	1	0	1	1	0	
kw_3	1	1	0	0	1	
kw_4	0	1	0	1	0	
kw_5	0	0	0	1	1	
...						

Document D1 contains all three keywords
D1

Figure 3.2 : Conjunctive Keywords Selection

C. Computation over Encrypted Data:

Paillier Homomorphic algorithm:
This Scheme is additively homomorphic; which means that, given a public-key and the encryption of d1 and d2, one can compute the encryption of d1+d2. For example, an additively homomorphic scheme would have the following property:
E(A) + E(B) = E(A + B)
This feature allows for third parties to perform computations without exposing confidential information.
Paillier cryptosystem [14]: Paillier algorithm is one of the most popular probabilistic homomorphic encryption algorithms in the literature.

Encryption:
1. Let m be a message to be encrypted where m Є Zn
2. Select random r where r Є Z* n
3. Compute ciphertext as c= gm.rm mod n2

Decryption:
1. Ciphertext c Є Z* n2
2. Compute message: m= L(cλ mod n2)μ mod n

Homomorphic properties:
A notable feature of the Paillier cryptosystem is its homomorphic properties. As the homomorphic encryption function is additively homomorphic, the following identities can be described:
Homomorphic addition of plaintext:
The product of two cipher-texts will decrypt to sum of their corresponding plaintexts,

D(E(m1,r1)* E(m2,r2)mod n2) = m1+m2 mod n
The product of a ciphertext with a plaintext raising g will decrypt to the sum of the corresponding plaintexts:

D(E(m1,r1)*gm2 mod n2)=m1+m2 mod n

D. Combining Symmetric and Homomorphic Encryption

To enable computations over encrypted data, homomorphic encryption is a natural choice. However, it is notably more expensive than symmetric encryption in both storage and computations. To benefit from its feature while minimizing its cost, we propose adapting the encryption scheme based on the

content of the document. That is, only content which may be used for computations, assumed here to be numeric data, are homomorphically encrypted, while remaining content are symmetrically encrypted. Using the example in section I-B, we may have
EAES(</ARTIST>) EAES(<PRICE>)
EPaillier(5.42) EAES (</PRICE>)

The scheme can be described as follows:

a) Document Encryption: Each document, Di, is parsed and encrypted such that numeric data are encrypted using EPaillier(m), as described and the remaining content is symmetrically encrypted using AES EAES (m).

b) Document Indexing: Each document, Di, is parsed and indexed as described.

c) Keyword Search: To compute F(x) where tagm.= kw1, kw2 . . . kwq and x ∈tagx., we must first resolve the clause by searching for keyword, kwi using the keyword-to-document index and the keyword location index if phrases are queried. The location of tagx is then queried for the matched documents. Markup symbol locations are also queried to resolve potential conflicts between keywords and tags.

d) Function computation: Once the location of the function arguments is determined, the cloud server computes F(x) using EPaillier(x) where x ∈ tagx.
For example, if the average price is desired, then tagx= 'price_ and F(x) = EPaillier(x). Upon receiving F(x), it is decrypted to obtain the sum and the average is obtained by dividing the number of matched elements.

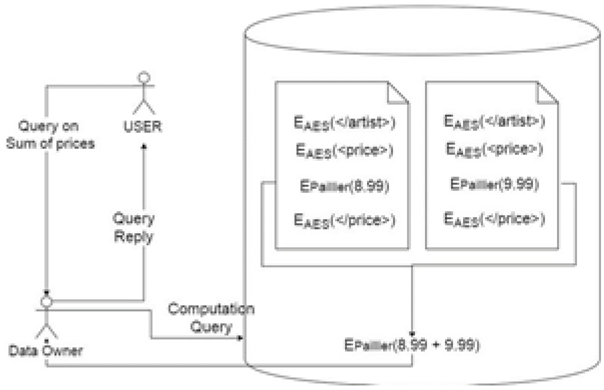


Figure 3.3: Combining Symmetric and Homomorphic Encryption

Note that the function, F (x), is not limited to summations. With minor modifications, other more complex functions can also be used such as scalar product [12] and comparison [13]. Our solution requires the exchange of two rounds of queries and the encrypted result of the function. In terms of computations, the queries consist of binary searches, hash computations and equality comparisons, each of which can be done efficiently. The efficiency of the computation of F (x) depends on the function and the encryption algorithm used. In terms of summation, Paillier's cryptosystem consists of a simple integer addition modulo n2. Since most XML documents consists of few numeric value relative to text, the computational cost of the document encryption is not believed to be significantly worse than pure symmetric encryption.
It is interesting to note that it is infeasible to deter- mine whether a ciphertext resulted from AES encryption or Paillier by examining the ciphertext alone. Therefore, a cloud server would yield no additional information from the encrypted documents. If further security is desired, it is possible to hide the location of the desired numeric values by requesting the computation of F (x) over other symmet-rically encrypted data, at the expense of computational and communication cost. Other security measures can also be included in the design of the keyword-to-document index and keyword location index to defend against statistical analysis at the expense of efficiency [9], [8].

CONCLUSION

We described a scheme for searching and computing over encrypted documents in XML-like formats with a low computational and communication cost. Due to the non-consistent placements in XML documents, a searching algorithm was required. In particular, a phrase search algorithm was required for many common search terms. Indexes provided an efficient way to access documents and keyword locations. To enable computations, homomorphic encryption was used. However, its computational and storage cost can be prohibitive depending on the application. To minimize its effect on the scheme, we restricted its use to numeric values. Despite its efficiency, the scheme is vulnerable to statistical analysis. In particular, a document set with commonly used tags in a known plaintext model could reveal meaningful information to a curious cloud server. Although the methods described in [9], [8] can provide some measures of security, they are expensive and would drastically reduce the practicality of the algorithms. As future work, we intend to investigate techniques to increase the level of security against statistical analysis while maintaining efficiency.

ACKNOWLEDGMENT

The authors would like to thank the college for giving them a platform to showcase their paper. They would also like to thank their assistant professor for supporting and guiding them in the creation and experimentation of this paper.

REFERENCES

[1] Ravi Chandra Jammalamadaka and Sharad Mehrotra, “Query-ing encrypted xml documents,” in International Database Engineering and Applications Symposium, 2006, pp. 129– 136.

[2] Kyung-Sang Sung, Hoon Ko, and Hae-Seok Oh, “Xml document encrypt implementation using elliptic curve cryp-tosystem,” in International Conference on Convergence Information Technology, 2007, pp. 2473–2478.

[3] Li Juan and Ming De-ting, “Research and application on the query processing for encrypted xml data,” in IEEE International Conference on Advanced Management Science, 2010, pp. 707– 711.

[4] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, “Public key encryption with keyword search,” in In proceed-ings of Eurocrypt, 2004, pp. 506–522.

[5] Brent Waters, Dirk Balfanz, Glenn Durfee, and D. K. Smet-ters, “Building an encrypted and searchable audit log,” in

[6] Network and Distributed System Security Symposium, 2004.

[7] Maozhen Ding, Fei Gao, Zhengping Jin, and Hua Zhang, “An efficient public key encryption with conjunctive keyword search scheme based on pairings,” in IEEE International Conference onNetwork Infrastructure and Digital Content, 2012, pp. 526– 530.

[8] F. Kerschbaum, “Secure conjunctive keyword searches for unstructured text,” in International Conference on Network and System Security, 2011, pp. 285–289.

[9] S. Zittrower and C. C. Zou, “Encrypted phrase searching in the cloud,” in IEEE Global Communications Conference, 2012, pp. 764–770.

[10] Yinqi Tang, Dawu Gu, Ning Ding, and Haining Lu, “Phrase search over encrypted data with symmetric encryption scheme,” in International Conference on Distributed Com-puting Systems Workshops, 2012, pp. 471–480.

[11] He Tuo and Ma Wenping, “An effective fuzzy keyword search scheme in cloud computing,” in International Conference on

[12] Intelligent Networking and Collaborative Systems, 2013, pp. 786–789.

[13] Wenliang Du and Mikhail J. Atallah, Protocols For Secure Remote Database Access With Approximate Matching, pp. 87– 111, 2001.

[14] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mieliki- nen, “On private scalar product computation for privacy- preserving data mining,” in International Conference in Information Security and Cryptology. 2004, pp. 104–120, Springer-Verlag.

[15] Ivan Damgard,˚ Martin Geisler, and Mikkel Kroigard, “Ho- momorphic encryption and secure comparison,” International Journal of Applied Cryptology, vol. 1, no. 1, pp. 22–31, 2008.

[16] “XML 1.0 Specification,” http://www.w3.org/TR/REC-xml/, Accessed: March 2015.

[17] H.T. Poon and A. Miri, “An efficient conjunctive keyword and phase search scheme for encrypted cloud storage sys-tems,” to appear in IEEE International Conference on Cloud Computing, 2015.

[18] Pascal Paillier, “Public-key cryptosystems based on compos-ite degree residuosity classes,” Lecture Notes in Computer Science, vol. 1592, pp. 223–238, 1999.