Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 12, No. 2, (2021),
ISSN : **1906-9685**

PROTECTING PRIVACY: RANKED SEARCH WITH MULTIPLE KEYWORDS OVER ENCRYPTED CLOUD DATA

[#1]Ms.KOTHAPALLY HARINI PRIYA, *Assistant Professor*
[#2]Mrs.PADMA RAVALI, *Assistant Professor*
Department of Computer Science and Engineering,
SREE CHAITANYA INSTITUTE OF TECHNOLOGICAL SCIENCES, KARIMNAGAR, TS.

ABSTRACT - The spread of cloud computing has increased the possibility of unwanted access to private data. To protect the privacy of sensitive data, a data owner should encrypt it before sending it to a third party. Investigating secure encrypted cloud data retrieval services is critical. A approach that is searchable, adaptable, and capable of quickly and efficiently securing multiple keywords. A data distributor revealed sensitive information to a number of third-party agents. A portion of the data has been confiscated and transferred to an illegal location, such as a laptop or the internet. Instead of getting the information independently, the distributor must assess the possibility that the data breach was caused by one or more spies. These methods do not rely on alterations to previously published data, such as watermarks. In extremely rare cases, we can additionally upload "realistic but fake" data records to boost the chances of discovering the breach and its culprit.

Keywords: Multi-keyword search, ranked search, encrypted cloud data, security, Secure data.

1. INTRODUCTION TO CLOUD COMPUTING

Cloud computing is a younger, but more established, type of IT infrastructure for organizations that offers high-quality applications and services on demand. This is achieved by pooling adaptive computing resources. Individuals and enterprises who use the cloud can transfer their complicated local data system to the cloud to save the expense of developing and maintaining a private storage infrastructure. However, because the Cloud Service Provider (CSP) has complete control over the data being delivered, a number of difficulties may occur. Individuals motivated by avarice or with a desire to make money may make unlawful alterations to the outsourced data. The owner of the material should encrypt it before sending emails, photo albums, personal health records, financial records, and other sensitive information. This protects the privacy of the data and renders the traditional unencrypted keyword search approach obsolete. Obviously, you cannot download and decrypt all of the data on your own computer, which would be a simple but difficult option. Two aspects should be considered while determining effective privacy-protecting search services. First and foremost, ranking search is critical since it allows for the speedy identification of the most useful information.

Because so many documents are being uploaded to the cloud, it should be able to prioritize search results in order to meet the demand for quick data retrieval. Second, because searches with a single keyword typically provide inaccurate results, searches with numerous keywords are also important for enhancing search result accuracy.

A searchable, adjustable, and effective encryption mechanism. The Vector Space Model (VSM) is used to generate a document index that allows for multi-keyword searches and ranking of results. We use a balanced binary tree, a tree-based index structure, to speed up queries. The searching index tree is built using document index vectors.

Here's a rundown of what we did to help:

We study the issue of complying to high privacy rules while running ranked searches on encrypted cloud data with numerous terms.

The search time for this study's index tree is

about O (r log m), where r is the number of documents that include the search keywords and m is the total number of documents in the collection.

People who hold data are referred to as producers, whereas those who receive data are referred to as agents. The information delivered by agents to selected organizations. A group of workers receive information from the distributor. Any agent could give the subject information. The distributor must assess the likelihood of information leakage. To locate the particular leaker and assess the possibility of a data leak, the distributor develops a blame agent model.

BACKGROUND AND RELATED WORK

The system model is made up of three distinct components: the data owner, the user, and the cloud service. The data owner encrypts the document collection before transferring it to the cloud to secure sensitive information from unauthorized parties. In addition, the data owner will construct an encrypted, searchable index based on a set of distinguishing keywords that can be used to find relevant material. When someone performs a search, the system will create an encrypted search trapdoor depending on the terms they provide (if the data owner grants permission). After receiving the trapdoor, the cloud server will search the index and return ranked search results to the user. The search results have been thoroughly assessed by the cloud server, and the user can specify a parameter with their search query to acquire the most relevant results. We assume that individuals who utilize data have been given permission to do so by the owner of the data. This is due to the fact that the topic of this paper is not key exchange.

In some cases, maintaining the integrity of the data from the original source is critical. Watermarking was traditionally used to detect breaches. For example, each delivered copy is given a unique code. If that duplicate is later found in the hands of an unauthorized person, the leaker can be identified.

Watermarks can be quite useful in certain situations, but they necessitate changing the source file. Furthermore, watermarks may be deleted if the data recipient is malicious. This research looks on covert ways for determining when records or objects are released.

The distributor discovers some of the same things in an unapproved area after distributing a set of products to agents. The distributor can now estimate the chance that the stolen data was obtained from several agents rather than a single one. For example, the information could have been obtained through a website or the court's discovery procedure. In this article, we offer a mechanism for determining the "guilt" of agents. In addition, we equip agents with algorithms that assist us in assigning items in a way that maximizes the possibility of discovering a leaker. We also take into account the possibility of "fake" items existing in the distributed set. The agents feel that these things make sense, despite the fact that they are not founded on reality.

The phony things, in a sense, act as a stamp of approval for the entire group, without changing the individual components. If it is discovered that the vendor was provided at least one false item that was made public, the seller will be even more certain that the agent was responsible.

2. EXISTING SCHEME

Due to the large number of documents, the cloud server must perform result relevance ranking. This reduces the requirement to return a variety of outcomes. This type of ranked search enables data consumers to quickly find the most relevant information without having to go through each match in the content collection. Ranking search can help minimize network traffic under the "pay-as-you-go" cloud model by displaying just the most useful information. However, because privacy is important, these ranking techniques should not reveal any keyword information. This type of ranking system must also be capable of handling searches involving several keywords, as searches involving a single term frequently give an excessive amount of results. This improves the accuracy of search results as well as the user's searching experience.

3. PROPOSED SCHEME

Our goal is to identify the agent who disclosed confidential information about the distributor and show that this agent has done so previously. "Perturbation," which alters data to make it "less sensitive" before delivering it to bots, is a particularly useful phase. We provide distinct

approaches for detecting data breaches in a group of objects or documents.

A model for determining an agent's "guilt" is established in this section. In addition, we equip agents with algorithms that assist us in assigning items in a way that maximizes the possibility of discovering a leaker. We also take into account the possibility of "fake" items existing in the distributed set. The agents feel that these things make sense, despite the fact that they are not founded on reality. The phony things, in a sense, act as a stamp of approval for the entire group, without changing the individual components. If it is discovered that the vendor was provided at least one false item that was made public, the seller will be even more certain that the agent was responsible.

Problem Setup and Notation:

T=t1,...,tm denotes a collection of valuable data pieces owned by the distributor. Some of the things should not be distributed to others, hence the distributor desires to distribute them through a set of agents called U1, U2, and so on. T can hold objects of any size or type, including relational tuples and database relations. The collection of things received by an agent Ui is determined by whether the request is a sample or explicit request.

- Sample request
- Explicit request

Module Description

Fake objects: Because the seller manufactures counterfeit goods, the buyer is more likely to be fooled.

identifying software that leaks data. They can tamper with the data being sent, allowing the distributor to identify guilty agents more rapidly. The system used bogus objects due to "trace" entries in email listings. Fake items are actual objects that are identical reproductions of other real objects. The distributor creates counterfeit items before releasing data to agents. Each data set that the distributor provides to his agents will have a different quantity and arrangement of counterfeit merchandise. The amount of counterfeit items will change depending on the number of recordings, allowing the computer to easily identify the culprit.

Data allocation strategies: The problem with data distribution is determining how to "intelligently" send data to agents in a way that maximizes the likelihood of identifying a bad agent. The amount of data delivered is determined by the agent's request and the system's ability to add phony objects. The following requests can be accommodated by the agent:

Sample- A sample data request sends a requested sampling of data from the distributors.

Explicit- The agent receives data that meets a certain set of criteria in a clear data request.

Optimization Module: To supply data to agents, the distributor employs the optimization module, which has one purpose and one restriction. This means that the dealer must meet the agents' needs by offering either the precise quantity requested or all in-stock items that meet their specifications. He wishes to be able to determine the source of the information about him.

Data Distributor: A data distributor revealed sensitive information to a number of third-party agents. A portion of the data has been confiscated and transferred to an illegal location, such as a laptop or the internet. Instead of getting the information independently, the distributor must assess the possibility that the data breach was caused by one or more spies.

Algorithms:

Guilt Model Analysis:

The components of our model interact with one another. This section looks at two simple instances, Impact of Probability p and Impact of Overlap between Ri and S, to see if our knowledge of how the parameters interact is correct. Because the client has purchased everything from the distributor in each case, T equals S.

T is a set of data objects.

T= {t1, t2... tn};

Set of agents= {U1, U2... Un}

- T can hold objects of any size or type,

including relational tuples and database relations. A sample or explicit request determines the objects Ri acquired by an agent Ui. Ri is found within T:

- Ri = SAMPLE (T, mi): Any subset of the records mi in T can be delivered to Ui as a sample request.
- Agent Ui receives all T objects that satisfy the condition when the explicit request Ri = EXPLICIT (T, condi) is issued.
- Agent Ui is judged responsible if he or she offers the target one or more goods. We're talking about the occurrence where agent Ui is guilty as Gi and the set where agent Ui is guilty as Gi|S.

Algorithm for Find Guilt Agent:

- The information is delivered via an agent chosen by the distributor. When an agent asks for help, the distributor decides which agents will receive information.
- The wholesaler gives the agent misleading information that is falsified. The individual distributing the data has the ability to generate false information and send it with or without agent information. The distributor can generate more misleading information, increasing the likelihood of the responsible person being arrested.
- Determine the number of agents who have already received data. How many agents have previously received the data is determined by the sender.
- Look for any other agents. The remaining agents are chosen by the distributor to provide the information. The distributor can increase the number of possible assignments by entering false data.
- Determine the likelihood of the responsible agent. To calculate this probability, we must first determine the target's ability to "guess" numbers.

Evaluation of Explicit Data Request Algorithms

The initial goal of these experiments was to see if adding fake objects to the distributed data sets would make identifying the guilty agent much easier. The second goal was to compare the performance of our e-optimal method to a random distribution.

Evaluation of Sample Data Request Algorithms

When agents seek samples of data, they are not interested in a few items. As a result, their searches do not explicitly mention object sharing. When the number of objects requested exceeds the number of objects in set T, the distributor is "forced" to distribute some objects to multiple users. The more data objects agents seek, the more recipients an object normally has, and the more difficult it is to identify a malicious agent when objects are being transmitted between agents.

Performance Analysis

In this section, we install the safe search system on a Windows 7 PC powered by a 2.83GHz Intel(R) Core(TM)2 Quad CPU to see how well our overall advice works. The document set is built from genuine data using the Request for Comments Database (RFC), which contains roughly 6500 items.

"Record-by-record leak report" and "Probability of agent guilt" are the two most frequent ways for determining a system's effectiveness.

$$\text{Record wise leak report} = \frac{|T \cap S|}{|S|}$$

This method computes leak data record per record, disregarding agent overlaps. As a result, the seller is aware of the factors utilized to determine the likelihood that an individual is liable.

$$\text{Probability of agent guilty} = \frac{|R_i \cap S|}{|S|}$$

When identifying leaks record by record, this approach takes into account the number of agents that a record shares with. As a result, the seller is aware of the factors utilized to determine the likelihood that an individual is liable. Over time, the Agent Probabilities will improve.

4. CONCLUSION AND FUTURE WORK

There would be no reason to give secret information to staff who may mistakenly or willfully reveal it. In an ideal world, we could attach a label to every item to ensure its origin, even if it meant sharing sensitive information. However, because watermarks do not work on all data, we frequently have to work with people on whom we cannot totally rely. As a result, we may be unable to tell whether an escaping object came from an agent or another source. Nonetheless, we proved that by considering the overlap between his data and the revealed data, as well as the data of other agents and the fact that objects can be "guessed" in other ways, it is possible to

identify if an agent is responsible for a leak. By changing how data is dispersed, the strategies we demonstrated can raise the likelihood of catching a leaker. We have shown that precise object placement can have a considerable impact on detecting the guilty agents, especially when a large portion of the data that the agents require overlaps. In our next endeavor, we will look into agent guilt models that can manage leaky scenarios that were not addressed in this study. For example, which model should be used when algorithms may work together to detect false tuples? Another unresolved issue is how to make our allocation methods work with online agent requests (the schemes we've given thus far assume that there is a set group of agents whose demands are known). This type of paradigm has a historical explanation.

REFERENCES

1. Data Leakage Detection And E-Mail Filtering Mr.Zarif Shaukat Ansari 1, Ms. Anagha Mahadeo Jagtap 2, Ms. Shilpa Suresh Raut Student, Dept. Of Computer Engineering, Trinity College Of Engineering, Pune, Maharashtra, India.
2. An Efficient And Robust Model For Data Leakage Detection System Janga Ajay Kumar 1 And K. Rajani Devi 2 1Student, M.Tech (IT), Siddharth Nagar, Nalanda Institute Of Engineering And Technology, A.P, India.
3. Allocation Strategies For Detecting And Identifying The Leakage And Guilty Agents Lata Dudam[1], Dr.Prof.Mrs.S.S.Apte[2] Walchand Institute Of Technology,Solapur, India.
4. R. Agrawal And J. Kiernan. Watermarking Relational Databases. In VLDB '02: Proceedings Of The 28th International Conference On Very Large Data Bases, Pages 155–166. VLDB Endowment, 2002.
5. An Image Watermarking Method Based On Mean-Removed Vector Quantization For Multiple Purposes Zhe-Ming Lu, Zhen Sun, Department Of Automatic Test And Control, Harbin Institute Of Technology, Harbin 150001,China.
6. P. Bonatti, S. D. C. Di Vimercati, And P. Samarati. An Algebra For Composing Access Control Policies. ACM Trans. Inf. Syst. Secur.,5(1):1–35, 2002.
7. Provenance. In J. V. Den Bussche And V. Vianu, Editors, Database Theory - ICDT 2001, 8th International Conference, London, UK, January 4-6, 2001, Proceedings, Volume 1973 Of Lecture Notes In Computer Science, Pages 316–330. Springer, 2001.
8. P. Buneman And W.-C. Tan. Provenance In Databases. In SIGMOD '07: Proceedings Of The 2007 ACM SIGMOD International Conference On Management Of Data, Pages 1171– 1173, New York, NY, USA, 2007. ACM.
9. F. Guo, J. Wang, Z. Zhang, X. Ye, And D. Li. Information Security Applications, Pages 138–149. Springer, Berlin / Heidelberg, 2006. An Improved Algorithm To Watermark Numeric Relational Data.
10. S. Jajodia, P. Samarati, M. L. Sapino, And V. S. Subrahmanian. Flexible Support For Multiple Access Control Policies. ACM Trans. Database Syst., 26(2):214–260, 2001.