Data Leakage Detection Using Cloud Computing

Prof. Sushilkumar N. Holambe, Dr.Ulhas B.Shinde, Archana U. Bhosale

Abstract—In the virtual and widely distributed network, the process of handover sensitive data from the distributor to the trusted third parties always occurs regularly in this modern world. It needs to safeguard the security and durability of service based on the demand of usersA data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data are leaked and found in an unauthorized place (e.g., on the web or somebody's laptop). The distributor must assess the likelihood that the leaked data came from one or more agents, as opposed to having been independently gathered by other means. We propose data allocation strategies (across the agents) that improve the probability of identifying leakages. These methods do not rely on alterations of the released data (e.g., watermarks). In some cases, we can also inject "realistic but fake" data records to further improve our chances of detecting leakage and identifying the guilty party. The idea of modifying the data itself to detect the leakage is not a new approach. Generally, the sensitive data are leaked by the agents, and the specific agent is responsible for the leaked data should always be detected at an early stage. Thus, the detection of data from the distributor to agents is mandatory. This project presents a data leakage detection system using various allocation strategies and which assess the likelihood that the leaked data came from one or more agents For secure transactions, allowing only authorized users to access sensitive data through access control policies shall prevent data leakage by sharing information only with trusted parties and also the data should be detected from leaking by means of adding fake record`s in the data set and which improves probability of identifying leakages in the system. Then, finally it is decided to implement this mechanism on cloud server.

Index Terms— cloud environment data leakage, data security, fake records.

1 INTRODUCTION

In this paper, we develop a model for finding the guilty agents. We also present algorithms for distributing objects to agents, in a way that improves our chances of identifying a leaker. Finally, we also consider the option of adding —fake objects to the distributed set. Such objects do not correspond to real entities but appear realistic to the agents. In a sense, the fake objects act as a type of watermark for the entire set, without modifying any individual members. If it turns out that an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty. We also consider optimization in which leaked data is compared with original data and accordingly the third party who leaked the data is guessed. We will also be using approximation technique to encounter guilty agents. We proposed one model that can handle all the requests from customers and there is no limit on number of customers. The model gives the data allocation strategies to improve the probability of identifying leakages. Also there is application where there is a distributor, distributing and managing the files that contain sensitive information to users when they send request. The log is maintained for every request, which is later used to find overlapping with the leaked file set and the subjective risk and for Assessment of guilt probability.

Data leakage happens every day when confidential business information such as customer or patient data, source code or design specifications, price lists, intellectual property and trade secrets, and forecasts and budgets in spreadsheets are leaked out. When these are leaked out it leaves the company unprotected and goes outside the jurisdiction of the corporation. This uncontrolled data leakage puts business in a vulnerable position. Once this data is no longer within the domain, then the company is at serious risk.

When cybercriminals —cash out ‖ or sell this data for profit it costs our organization money, damages the competitive advantage, brand, and reputation and destroys customer trust. To address this problem, we develop a model for assessing the —guilt ‖ of agents. The distributor will —intelligently‖ give data to agents in order to improve the chances of detecting a guilty agent like adding the fake objects to distributed sets.

At this point the distributor can assess the likelihood that the leaked data came from one or more agents, as opposed to having been independently gathered by other means. If the distributor sees enough evidence that an agent leaked data then they may stop doing business with him, or may initiate legal proceedings. Mainly it has one constraints and one objective. The Distributor's constraint satisfies the agent, by providing number of object they request that satisfy their conditions.

2 LITERATURE SURVEY

The guilt detection approach we present is related to the data provenance problem [3]: tracing the lineage of S objects implies essentially the detection of the guilty agents. and assume some prior knowledge on the way a data view is created out of data sources. objects and sets is more general .As far as the data allocation strategies are concerned; our work is mostly relevant to watermarking that is used as a means of establishing original ownership of distributed objects. [3] Finally, there are also lots of other works on mechanisms that allow only authorized users to access sensitive data through access control policies [9], [2]. Such approaches prevent in some sense data leakage by sharing information only with trusted parties. However, these policies are restrictive and may make it impossible to satisfy agent's requests. Maintaining the Integrity of the Specifications

3 NEED FOR DATA ALLOCATION STRATEGIES

Using the data allocation strategies, the distributor intelligently give data to agents in order to improve the chances of detecting guilty agent. Fake objects are added to identify the guilty party. If it turns out an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty and when the distributor sees enough evidence that an agent leaked data then they may stop doing business with him, or may initiate legal proceedings. In this section we describe allocation strategies that solve exactly or approximately the scalar versions of approximation equation. We resort to approximate solutions in cases where it is inefficient to solve accurately the optimization problem.

3.1 Explicit Data Request

In case of explicit data request with fake not allowed, the distributor is not allowed to add fake objects to the distributed data. So Data allocation is fully defined by the agent's data request. In case of explicit data request with fake allowed, the distributor cannot remove or alter the requests R from the agent. However distributor can add the fake object. In algorithm for data allocation for explicit request, the input to this is a set of request ,……, from n agents and different conditions for requests. The e-optimal algorithm finds the agents that are eligible to receiving fake objects. Then create one fake object in iteration and allocate it to the agent selected. The e-optimal algorithm minimizes every term of the objective summation by adding maximum number of fake objects to every set yielding optimal solution.

Step 1: Calculate total fake records as sum of fake Records allowed.
Step 2: While total fake objects > 0
Step3: Select agent that will yield the greatest improvement in the sum objective
i.e. i = argma x((1/ | Ri |)-(1/(| Ri+1 |))) ΣRi∩Rj
Step 4: Create fake record
Step 5: Add this fake record to the agent and also to fake record set.

3.2 Sample Data Request

With sample data requests, each agent Ui may receive any T subset out of different object allocations. In every allocation, the distributor can permute T objects and keep the same chances of guilty agent detection. The reason is that the guilt probability depends only on which agents have received the leaked objects and not on the identity of the leaked objects. The distributor gives the data to agents such that he can easily detect the guilty agent in case of leakage of data. To improve the chances of detecting guilty agent, he injects fake objects into the distributed dataset. These fake objects are created in such a manner that, agent cannot distinguish it from original objects. One can maintain the separate dataset of fake objects or can create it on demand. In this paper we have used the dataset of fake tuples. For example, distributor sends the tuples to agents A1 and A2 as R1= {t1, t2} and R2= {t1}.
If the leaked dataset is L= {t1}, then agent A2 appears more guilty than A1. So to minimize the overlap, we insert the fake objects in to one of the agent's dataset. Practically server (Distributor) has given sensitive data to agent. In that distributor can send data with fake information. And that fake information does not affect to Original Data. Fake formation cannot identify by client. it also finds the data leakage from which agent (client)

- *Prof. Sushilkumar N.Holambe.He has completed master degree in computer science & Engg. C.O.E, Osmanabd. Dr. B.A.M. University. & persuing P.H.D.in Dr. B.A.M.U. snholambe@yahoo.com*
- *Dr.Ullhas B. Shinde, Dean, Faculty of Engg. & Technology. Dr. B.A.M.U. Aurangabad.*
- *Archna U. Bhosale , Completed B.E.C.S.E.,C.O.E. Osmanabad, persuing M.E.C.S.E.* in C.O.E. Osmanabad. *archana21arun@gmail.com*

4 METHODOLOGY

4.1 Problem Definition

The distributor owns the sensitive data set T= {t1, t2… tn}. The agent Ai request the data objects from distributor. The objects in T could be of any type and size, e.g. they could be tuples in a relation, or relations in a database. The distributor gives the subset of data to each agent., After giving objects to agents, the distributor discovers that a set L of T has leaked. This means some third party has been caught in possession of L. The agent Ai receives a subset Ri of objects T determined either by implicit request or an explicit request. Implicit Request Ri = Implicit (T, mi) : Any subset of mi records from T can be given to agent Ai
Explicit Request Ri = Explicit (T, Condi) : Agent Ai receives all T objects that satisfy Condition.

4.2 Data Allocation Module

The distributor may be able to add fake objects to the distributed data in order to improve his effectiveness in detecting guilty agents. However, fake objects may impact the correctness of what agents do, so they may not always be allowable. Our use of fake objects is inspired by the use of —trade re cords in mailing lists. In this case, company A sells to company B a mailing list to be used once (e.g., to send advertisements). Company A adds trace records that contain addresses owned by company A. Thus, each time company B uses the purchased mailing list, A receives copies of the mailing. These records are a type of fake objects that help identify improper use of data. The distributor creates and adds fake objects to the data that he distributes to agents. Depending upon the addition of fake tuples into the agent's request, data allocation problem is divided into four cases as:
i. Explicit request with fake tuples (EF)
ii. Explicit request without fake tuples (E~F)
iii. Implicit request with fake tuples (IF)
iv. Implicit request without fake tuples (I~F).
Implicit Request Ri = Implicit (T, mi) : Any subset of mi records from T can be given to agent Ai

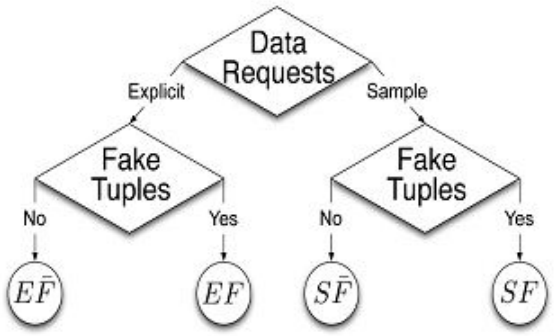


Fig. 4. Leakage problem instances.

4.3 Optimization Module

The distributor's data allocation to agents has one constraint and one objective. The distributor's constraint is to satisfy agents' requests, by providing them with the number of objects they request or with all available objects that satisfy their conditions. His objective is to be able to detect an agent who leaks any portion of his data. The objective is to maximize the chances of detecting a guilty agent that leaks all his data objects. The Pr { Gj|S =Ri } or simply Pr {Gj |Ri } is the probability that agent is guilty if the distributor discovers a leaked table S that contains all objects .

Let the distributor have data request from n agents. The distributor wants to give tables R1 ,R2……..Rn to agents A1 ,A2…… An respectively,

. so that Distribution satisfies agent's request; and

Maximizes the guilt probability differences

Δ (i, j) for all i, j= 1, 2, ……n and i≠j.

maximize(overR1….,Rn) (…,.Δ(i,j),…) i≠j……..(A)

minimize(over R1,….,Rn) (..,|Ri∩Rj|÷|Ri|,…) i≠j

4.4 Guitl Model Assessment

Let L denote the leaked data set that may be leaked intentionally or guessed by the target user. Since agent having some of the leaked data of L, may be susceptible for leaking the data. But he may argue that he is innocent and that the L data were obtained by target through some other means. Our goal is to assess the likelihood that the leaked data came from the agents as opposed to other resources. E.g. if one of the object of L was given to only agent A1, we may suspect A1 more. So probability that agent A1 is guilty for leaking data set L is denoted as Pr{Gi| L} .

Algorithm1:

Allocation of Data Explicitly:

Input: -

i. T= {t1, t2, t3, .tn}-Distributor's Dataset

ii. R- Request of the agent

iii. Cond- Condition given by the agent

iv. m= number of tuples given to an agent m<n, selected randomly

Output: - D- Data sent to agent

1. D=Φ, T′=Φ
2. For i=1 to n do
3. If(t .fields==cond) then
4. T′=T′U{ t i}
5. For i=0 to i<m do
6. D=DU{ti}
7. T′=T′-{ ti}
8. If T′=Φ then
9. Goto step 2
10. Allocate dataset D to particular agent
11. Repeat the steps for every agent

To improve the chances of finding guilty agent we can also add the fake tuples to their data sets.

Algorithm2:

Addition of fake tuples:

Input:

i. D- Dataset of agent

ii. F- Set of fake tuples

iii. Cond- Condition given by agent

iv. b- number of fake objects to be sent.

Output:- D- Dataset with fake tuples

1. While b>0 do
2. f= select Fake Object at random from set F
3. D= DU {f}
4. F= F-{f}
5. b=b-1

Similarly, we can distribute the dataset for implicit request of agent. For implicit request the subset of distributor's dataset is selected randomly. Thus with the implicit data request we get different subsets. Hence there are different data allocations. An object allocation that satisfies requests and ignores the distributor's objective to give each agent unique subset of T of size m. The s-max algorithm allocates to an agent the data record that yields the minimum increase of the maximum relative overlap among any pair of agents. The s-max algorithm is as follows:

1. Initialize Min_Overlap, the minimum out of the minimum relative overlaps that the allocations of different objects to Ai

2. for k do Initialize max_rel_ov←0, the maximum relative overlap between Ri the allocation of tk to Ai

3. for all j=1,……,n:j=I and tk∈Rj do calculate absolute overlap as abs_ov← calculate relative overlap as rel_ov←abs_ov/min(mi, mj)

4. Find maximum relative overlap as Max_rel_ov←MAX(max_rel_ov, rel_ov) If max_rel_ov≤ min_ov then Min_ov←max_rel_ovret_k←k Return ret_k

The algorithm presented implements a variety of data distribution strategies that can improve the distributor's

chances of identifying a leaker. It is shown that distributing objects judiciously can make a significant difference in identifying guilty agents, especially in cases where there is large overlap in the data that agents must receive.

5 BESICS OF CLOUD COMPUTING

Key to the definition of cloud computing is the ―cloud itself. For our purposes,
The cloud is a large group of interconnected computers. These computers can be personal computers or network servers; they can be public or private. For example, Google hosts a cloud that consists of both smallish PCs and larger servers. Google's cloud is a private on(that is, Google owns it) that is publicly accessible (by Google's users).
This cloud of computers extends beyond a single company or enterprise. The applications and data served by the cloud are available to broad group of users, cross-enterprise and cross-platform. Access is via the Internet. Any authorized user can access these docs and apps from any computer over any Internet connection. And, to the user, the technology and infrastructure behind the cloud is invisible. It isn't apparent (and, in most cases doesn't matter)whether cloud services are based on HTTP, HTML, XML, Java script, or other specific technologies.
From Google's perspective, there are six key properties of cloud computing:

- ***Cloud Computing is user-centric***. Once you as a user are connected to the cloud, whatever is stored there -- documents, messages, images, applications, whatever – becomes yours. In addition, not only is the data yours, but you can also share it with others. In effect, any device that accesses your data in the cloud also becomes yours.
- ***Cloud computing is task-centric***. Instead of focusing on the application and what it can do, the focus is on what you need done and how the application can do it for you., Traditional applications—word processing, spreadsheets, email, and so on – are becoming less important than the documents they create.
- ***Cloud computing is powerful***. Connecting hundreds or thousands of computers together in a cloud creates a wealth of computing power impossible with a single desktop PC.
- ***Cloud computing is accessible.*** Because data is stored in the cloud, users can instantly retrieve more information from multiple repositories. You're not limited to a single source of data, as you are with a desktop PC.
- ***Cloud computing is intelligent***. With all the various data stored on the computers in the cloud, data mining and analysis are necessary to access that information in an intelligent manner.
- ***Cloud computing is programmable***. Many of the tasks necessary with cloud computing must be automated. For example, to protect the integrity of the data, information stored on a single computer in the cloud must be replicated on other computers in the cloud. If that one computer goes offline, the cloud's programming automatically redistributes that computer's data to a new computers in the cloud.

Computing in the cloud may provide additional infrastructure and flexibility.

5.1 Databases in Cloud Computing Environment

In the past, a large database had to be housed onsite, typically on a large server. That limited database access to users either located in the same physical location or connected to the company's internal database and excluded, in most instances, traveling workers and users in remote offices.
Today, thanks to cloud computing technology, the underlying data of a database can be stored in the cloud, on collections of web server instead of housed in a single physical location.
This enables users both inside and outside the company to access the same data, day or night, which increases the usefulness of the data. It's a way to make data universal

5.2 Lineage Tracing General Data wearhouse Tranformations [9]

.
Yingwei Cui and Jennifer Widom focus on transformation or modification of data happening automatically due to mining of data or while storing the data in the warehouse.
In a warehousing environment, the *data lineage* problem is that of tracing warehouse data items back to the original source items from which they were derived. It formally defines the lineage tracing problem in the presence of general data warehouse transformations, and they present algorithms for lineage tracing in this environment. The tracing procedures takes advantage of known structure or properties of transformations when present, but also work in the absence of such information. Their results can be used as the basis for a lineage tracing tool in a general warehousing setting, and also can guide the design of data warehouses that enable efficient lineage tracing.
The major drawback is that it should not focus on the latest tools which will solve this kind of problem automatically and there is no clear explanation is given at its security part of this technique.

5.3 Databases in the cloud:a Work in Progress[10]

Edward P. Holden, Jai W. Kang, Dianne P. Bills, MukhtarIlyassov focus on trial of using cloud computing in the delivery of the Database Architecture and Implementation in the cloud.
It describes a curricular initiative in cloud computing intended to keep our information technology curriculum at the forefront of technology. Currently, IT degrees offer extensive database concentrations at both the undergraduate and graduate levels. Supporting this curriculum requires extensive lab facilities where students can experiment with different aspects of database architecture, implementation, and administration. A *disruptive technology* is defined as a new, and often an initially less capable technological solution, that displaces an existing technology because it is lower in cost. Cloud computing fits this definition in that it is poised to replace the traditional

model of purchased-software on locally maintained hardware platforms.

From this perspective in academic, cloud computing is utilizing scalable virtual computing resources, provided by vendors as a service over the Internet, to support the requirements of a specific set of computing curricula without the need for local infrastructure investment.

Cloud computing is the use of *virtual computing technology* that is scalable to a given application's specific requirements, without local investment in extensive infrastructure, because the computing resources are provided by various vendors as a service over the Internet.

6 Experimental Result

In our scenarios we have taken a set of 500 objects and requests from every agent are accepted. There is no limit on number of agents, as we are considering here their trust values. The flow of our system is given as below:

1. Agent's Request: Either Explicit or Implicit.
2. Leaked dataset given as an input to the system.
3. The list of all agents having common tuples as that of leaked tuples is found and the corresponding guilt probabilities are calculated.
4. It shows that as the overlap with the leaked dataset minimizes the chances of finding guilty agent increases.

The basic approaches for leakage identification system in various areas and there by proposing a multi-angle approach in handling the situational issues were all studied in detailed.

When the occurrence of handover sensitive data takes place it should always watermarks each object so that it could able to trace its origins with absolute certainty, however certain data cannot admit watermarks then it is possible to assess the likelihood that an agent is responsible for a leak, based on the overlap of the data with the leaked data and also based on the probability that objects can be guessed by any other methodologies.

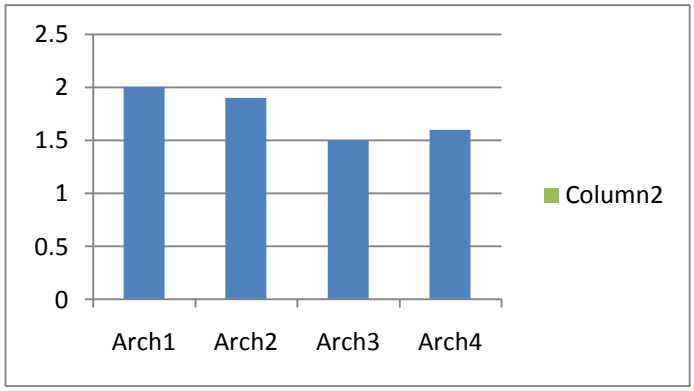
Sample request

Case 1) M>[t],M=∑i=1…n

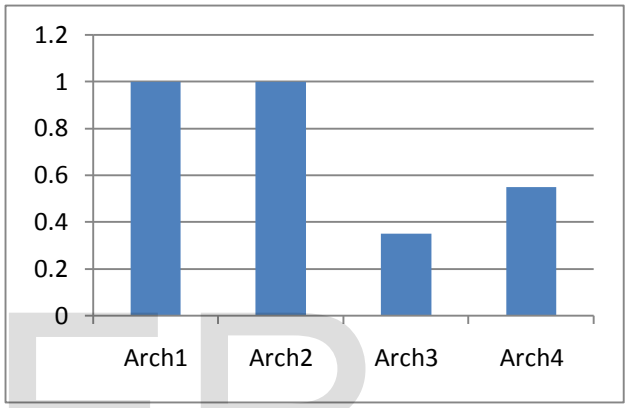
Agents	Files requested	Files given
Arch1	5	5
Arch2	5	-
Arch3	10	10
Arch4	10	-

Here M =30 i.e M>[t]

Graph probability (p)=0.3



Overlap graph probability at p=0.3

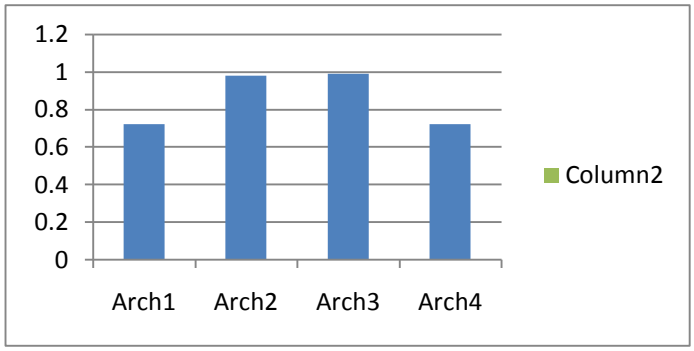


Overlap graph at p=0.3

Case II M<[t] , where M=∑i=1…n

Agents	Files requested	Files given
Arch1	8	8
Arch2	7	-
Arch3	8	5
Arch4	6	-

Random Graph at p= 0.3



7 CONCLUSION

Data leakage is a silent type of threat. Your employee as an insider can intentionally or accidentally leak sensitive information. This sensitive information can be electronically distributed via e-mail, Web sites, FTP, instant messaging, spread sheets, databases, and any other electronic means available – all without your knowledge. To assess the risk of distributing data two things are important, where first one is data allocation strategy that helps to distribute the tuples among customers with minimum overlap and second one is calculating guilt probability which is based on overlapping of his data set with the leaked data set.

7.1 Acknowledgments

We sincerely thank Prof. Sushilkumar N. Holambe ,my project guide,
Dr. Anilkumar N. Holambe, our P.G. co-ordinator, &
Head of the Department, COE, Osmanabadand .
Dr.S.M.Jagade,Principal,COE,Osmanabadand,for their constant encouragement and motivation to write this paper.

REFERENCES

[1] Papadimitriou P, Garcia-Molina H. A Model For Data Leakage Detection// IEEE Transaction On Knowledge And Data EngineeringJan.2011.

[2] International Journal of Computer Trends and Technology- volume3Issue1-2012 ISSN:2231-2803
http://www.internationaljournalssrg.org Data Allocation Strategies for Detecting
Data LeakageSrikanthYadav, Dr. Y. Eswararao, V. ShanmukhaRao, R. Vasantha

[3] International Journal of Computer Applications in Engineering Sciences [ISSN: 2231-4946]197 | P a g e Development of Data leakage Detection Using Data Allocation Strategies Rudragouda G PatilDept of CSE,The Oxford College of Engg, Bangalore.

[4] P. Buneman, S. Khanna and W.C. Tan. Why and where: Acharacterization of data provenance. ICDT 2001, 8th International Conference, London, UK, January4-6, 2001,Proceedings, volume 1973 of Lecture Notes in Computer Science, Springer, 2001

[5] S. Jajodia, P. Samarati, M.L. Sapino, and V.S. Subrahmanian, —Flexible Support for Multiple Access Control Policies,‖ *ACM Trans. Database* Systems, vol. 26, no. 2, pp. 214-260, 2001.

[6] P. Bonatti, S.D.C. di Vimercati, and P. Samarati, —An Algebra for Composing Access Control Policies,‖ ACM Trans. Information scientificcommons and System Security, vol. 5, no. 1, pp. 1-35, 2002.

[7] YIN Fan, WANG Yu, WANG Lina, Yu Rongwei A Trustworthiness-Based Distribution Model for Data Leakage Detection: Wuhan University Journal Of Natural Sciences.

[8] RakeshAgrawal, Jerry Kiernan. Watermarking Relational Databases// IBM Almaden Research Center.

[9] L. Sweeney, —Achieving K-Anonymity Privacy Protection Using Generalization ‖ And Suppression, http://en.scientificcommons.org/43196131,2002

.[10] Edward P. Holden, Jai W. Kang, Geoffrey R. Anderson, Dianne P. Bills, Databases in the Cloud: A Work in Progress,2012.