DATA LEAKAGE DETECTION USING DATA ALLOCATION STRATEGIES

Jaymala Chavan, Priyanka Desai
Thakur College of Engg. & Tech, Mumbai, MH, India

ABSTRACT

In the proposed system, we study unobtrusive techniques for detecting leakage of a set of objects or records .We study the following problem: consider business process outsourcing (BPO). In many cases the service provider needs access to the company intellectual property and other confidential information to carry out their services. For example a human resources BPO vendor may need access to employee databases with sensitive information (social security numbers), a marketing service vendor to contact information for customers or a payment service provider may need to access the credit card numbers or bank account numbers. The main security problem in BPO is that the service provider may not be fully trusted or may not be securely administered. Due to digital nature, relational databases are easy to duplicate and in many cases a service provider may have financial incentives to redistribute commercially valuable data or may simply handle it properly. Hence, we need powerful techniques that can detect and deter such dishonest. So in this system, A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data may be leaked and may be 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. The paper proposes 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. In the proposed approach, a model shall be developed for assessing the "guilt" of agents. This paper proposed 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 the agent was guilty. In the Proposed framework the hackers can be traced with good amount of evidence. The objective of the system is to detect when the distributor's sensitive data has been leaked by the hackers (e.g. considering the date and time) and if possible to identify the agent who leaked the data.

KEYWORDS - *Allocation Strategies, Fake Records, Guilt Model*

I. INTRODUCTION

In the course of doing business, sometimes sensitive data must be handed over to supposedly trusted third parties. For example, a hospital may give patient records to researchers who will devise new treatments. Similarly, a company may have partnerships with other companies that require sharing customer data. Another enterprise may outsource its data processing, so data must be given to various other companies. We identify the owner of the data as the distributor and the supposedly trusted third parties as the agents. Our goal is to detect when the distributor's sensitive data has been leaked by agents, and if possible to identify the agent that leaked the data.
We consider applications where the original sensitive data cannot be perturbed. Perturbation is a very useful technique where the data is modified and made "less sensitive" before being handed to agents. For example, one can add random noise to certain attributes, or one can replace exact values by ranges

[8]. However, in some cases it is important not to alter the original distributor's data. In paper [1][2], there is an unobtrusive techniques for detecting leakage of a set of objects or records. Specifically we study the following scenario: After giving a set of objects to agents, the distributor discovers some of those same objects in an unauthorized place. (For example, the data may be found on a web site, or may be obtained through a legal discovery process.).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.

Traditionally, Leakage detection is handled by watermarking [5] [6] e.g. unique code embedded in each distributed copy. But this watermarking involve some modification of original data. Furthermore watermarks sometimes can be destroyed if data recipient is malicious. But in some cases it is important not to alter the original distributor's data.

So in this project, we study unobtrusive techniques for detecting leakage of a set of objects or records. We develop a model for assessing the "guilt" of agents. We also present allocation 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.

We start in Section IV by introducing our problem setup and the notation we use. In Sections V & VI, we present strategies for data allocation to agents. Then in Sections VII, we present a model for calculating "guilt" probabilities in cases of data leakage. In Section VIII, we present results of implementation. Then finally, section IX include conclusion and future work of this paper.

II. RELATED WORK

The data leakage prevention based on the trustworthiness [3] is used to assess the trustiness of the agent. Maintaining the log of all agent's requests is related to the data provenance problem [4] i.e. tracing the lineage of objects. The data allocation strategy [1] [2] used is more relevant to the watermarking [5], [6] that is used as a means of establishing original ownership of distributed objects.

There are also different mechanisms to allow only authorized users, to access the sensitive information [7] through access control policies, but these are restrictive and may make it impossible to satisfy agent's requests.

III. PROPOSED WORK

In this, we proposed a model for assessing the "guilt" of agents. An algorithms for distributing objects to agents is proposed, in such a way that improves our chances of identifying a leaker. Finally, considering the option of adding "fake" objects to the distributed set. Such objects do not correspond to real entities but appear realistic to the agents. So 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. The following fig.1 shows how protected database can be leaked from various sources.

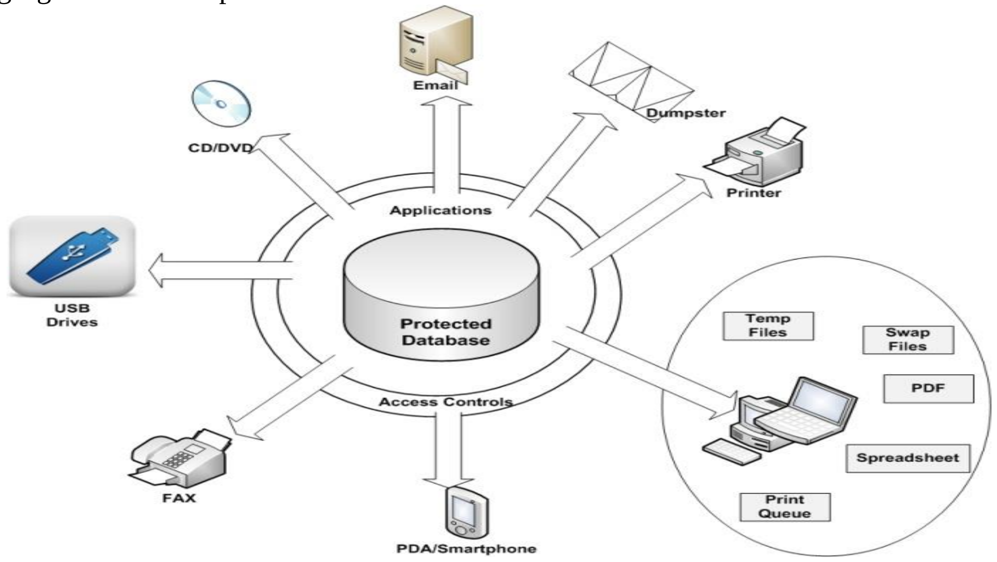


Fig. 1 An Example of Data Leakage [11]

IV. Problem Setup and Notation

A distributor owns a set T = {t1 . . . t_m} of valuable data objects. The distributor wants to share some of the objects with a set of agents U1; U2; . . . ; Un, but does not wish the objects be leaked to other third parties. The objects in T could be of any type and size. An agent Ui receives a subset of objects in T, i.e. Ri ⊆T, determined either by a sample request or an explicit request:

- Sample request Ri = **SAMPLE** (T, mi): Any subset of mi records from T can be given to Ui.
- Explicit request Ri=**EXPLICIT** (T, condi): Agent Ui receives all T objects that satisfy condi.
- Fig.2 shows system architecture with these two types of requests.

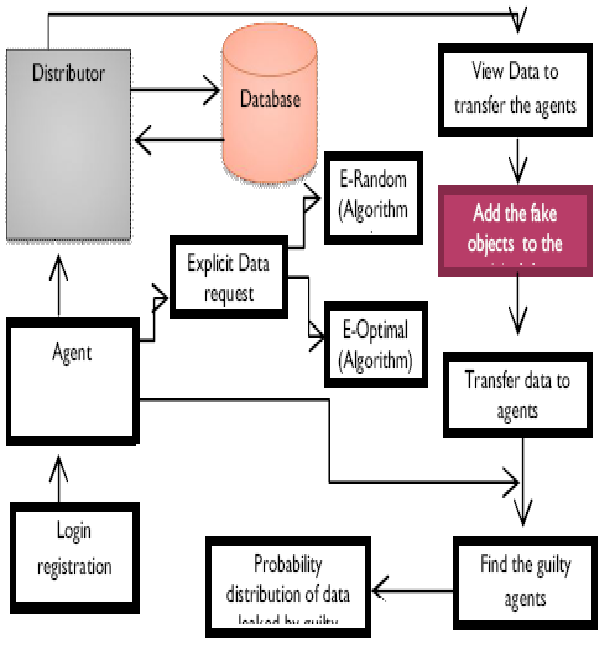


Fig.2 System Architecture [10]

V. Data Allocation Algorithm

In this [1][2], an algorithm is presented for the sample data allocation and explicit data allocation with the addition of fake tuples. Whenever any agent request for the data is received, it follows the following steps:
1. The request is sent by the agent to the distributor.
2. The request may be implicit or explicit.
3. If it is implicit, a subset mi of the data set T is given.
4. If request is implicit, it is checked with the log, if any previous request is same.
5. If request is same then system gives the data objects that are not given to previous agent.
6. If request is explicit, 10% tuples inserted in it are fake.
7. Leaked data set L, obtained by distributor is given as an input.
8. Calculate the guilt probability Gi of agent.

1 Algorithm for Sample Request

```
1: a ← 0|T|  a[k]:number of agents who have received object tk
2: R1 ← ∅, . . . ,Rn←∅
3: remaining ←
4: while remaining > 0 do
5: for all i = 1, . . . , n : |Ri| < mi do
6: k ← SELECTOBJECT(i,Ri)  May also use additional Parameters
7: Ri ← Ri∪ {tk}
8: a[k] ← a[k] + 1
```

```
9: remaining ← remaining – 1
```

2 Algorithm for Explicit Request

```
 1: R ← ∅  Agents that can receive fake objects
 2: for i = 1, . . . , n do
 3: if bi > 0 then
4: R ← R ∪ {i}
5: Fi ← ∅
6: while B > 0 do
7: i ← SELECTAGENT(R,R1, . . . , Rn)
8: f ← CREATEFAKEOBJECT(Ri, Fi, condi)
9: Ri ← Ri∪ {f}
10: Fi ← Fi ∪ {f}
11: bi ← bi − 1
12: if bi = 0 then
13: R ← R\{Ri}
14: B ← B – 1
```

VI. FAKE OBJECT

Fake objects must be created carefully so that agents cannot distinguish them from real objects. The distributor may want to limit the number of fake objects received by each agent, so as to not arouse suspicions and to not adversely impact the agents activities. Thus, we say that the distributor can send up to bi fake objects to agent Ui Creation. The creation of fake but real-looking objects is a non-trivial problem whose thorough investigation is beyond the scope of this paper. The creation of a fake object for agent Ui as a black-box function CREATEFAKEOBJECT(R_i; F_i; $cond_i$). R_i is the set of all objects, Fi is the subset of fake objects that Ui has received so far , the function returns a new fake object. This function needs condition to produce a valid object that satisfies Ui’s condition. Set Ri is needed as input so that the created fake object is not only valid but also indistinguishable from other real objects. The function CREATEFAKEOBJECT() has to be aware of the fake objects Fi added so far, again to ensure proper statistics. Although we do not deal with the implementation of CREATEFAKEOBJECT(), we note that there are two main design options. The function can either produce a fake object on demand every time it is called, or it can return an appropriate object from a pool of objects created in advance.

VII. GUILT MODEL ANALYSIS

The model parameters interact and check if the interaction matches the intuition. In this section we study two simple scenarios, Impact of Probability p and Impact of Overlap between Ri and S. In each scenario a target, that has obtained all the distributor’s objects, i.e., T = S.

GUILTY AGENTS

Suppose that after giving objects to agents, the distributor discovers that a set S (T)has leaked. This means that some third party, called the target, has been caught in possession of S. For example, this target may be displaying S on its website, or perhaps as part of a legal discovery process, the target turned over S to the distributor. Since the agents U1.. . . .Un has some of the data, it is reasonable to suspect them leaking the data. However, the agents can argue that they are innocent, and that the S data were obtained by the target through other means. For example, say that one of the objects in S represents a customer X. Perhaps X is also a customer of some other company, and that company provided the data to the target or perhaps X can be reconstructed from various publicly available sources on the web. Our goal is to estimate the likelihood that the leaked data came from the agents as opposed to other sources. Intuitively, the more data in S, the harder it is for the agents to argue they did not leak anything.
Similarly, the “rarer” the objects, the harder it is to argue that the target obtained them through other means. Not only do we want to estimate the likelihood the agents leaked data, but we would also like

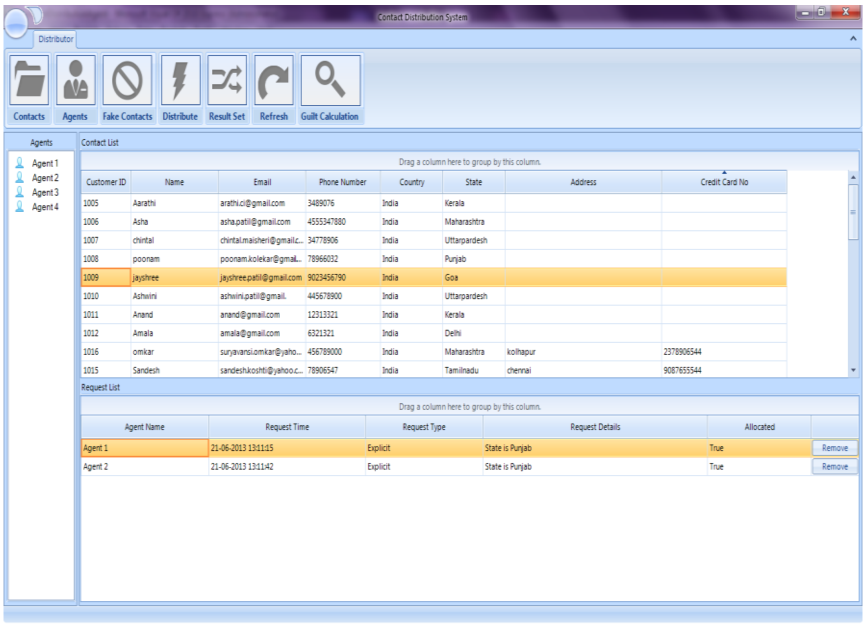
to find out if one of them, in particular, was more likely to be the leaker. For instance, if one of the S objects were only given to agent U1, while the other objects were given to all agents, we may suspect U1 more.
The model we present captures this intuition. We say an agent Ui is guilty and if it contributes one or more objects to the target. We denote the event that agent Ui is guilty by Gi and the event that agent Ui is guilty for a given leaked set S by G_i | S. Our next step is to estimate P_r{ G_i| S }, i.e., the probability that agent Ui is guilty given evidence S.

VIII. EXPECTED RESULTS

1. Distributor Module*:*

All the privileges of the system are only available with the distributor.

- Manage Users (Add, Edit, Delete, Lock/Unlock, Assign Permissions).
- Manage Articles (Add, Edit, Update, Delete).
- Manage Categories (Add, Edit, Update, Delete).
- Send Messages and upload Documents



Agents
Agent 1
Agent 2
Agent 3
Agent 4

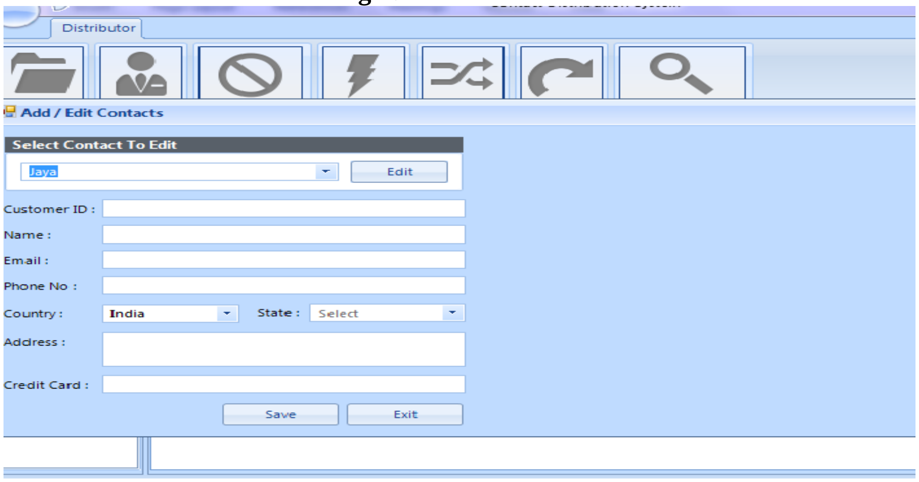
Contact List
Drag a column here to group by this column.

Customer ID	Name	Email	Phone Number	Country	State	Address	Credit Card No
1005	Aarathi	arathi.ci@gmail.com	3489076	India	Kerala		
1006	Asha	asha.patil@gmail.com	4555347880	India	Maharashtra		
1007	chintal	chintal.maisheri@gmail.c...	34778906	India	Uttarpardesh		
1008	poonam	poonam.kolekar@gmai...	78966032	India	Punjab		
1009	jayshree	jayshree.patil@gmail.com	9023456790	India	Goa		
1010	Ashwini	ashwini.patil@gmail.	445678900	India	Uttarpardesh		
1011	Anand	anand@gmail.com	12313321	India	Kerala		
1012	Amala	amala@gmail.com	6321321	India	Delhi		
1016	omkar	suryavansi.omkar@yaho...	456789000	India	Maharashtra	kolhapur	2378906544
1015	Sandesh	sandesh.koshti@yahoo.c...	78906547	India	Tamilnadu	chennai	9087655544

Request List
Drag a column here to group by this column.

Agent Name	Request Time	Request Type	Request Details	Allocated	
Agent 1	21-06-2013 13:11:15	Explicit	State is Punjab	True	Remove
Agent 2	21-06-2013 13:11:42	Explicit	State is Punjab	True	Remove

Fig 3: Distributor GUI



Select Contact To Edit
Jaya

Customer ID :	
Name :	
Email :	
Phone No :	
Country :	India
State :	Select
Address :	
Credit Card :	

Fig 4 : Distributor adding contact data into database

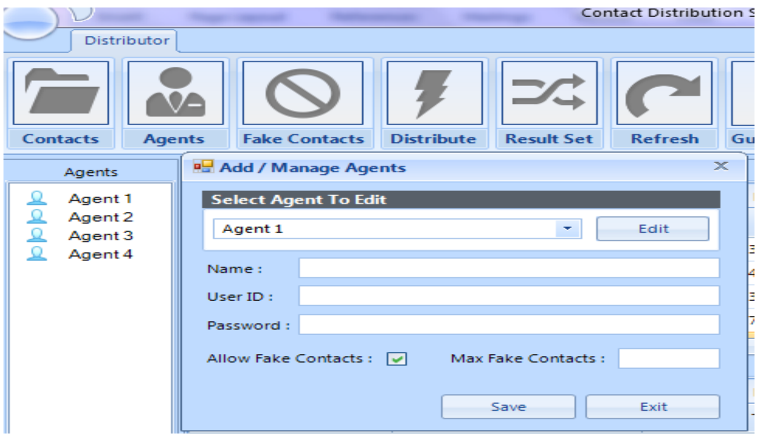
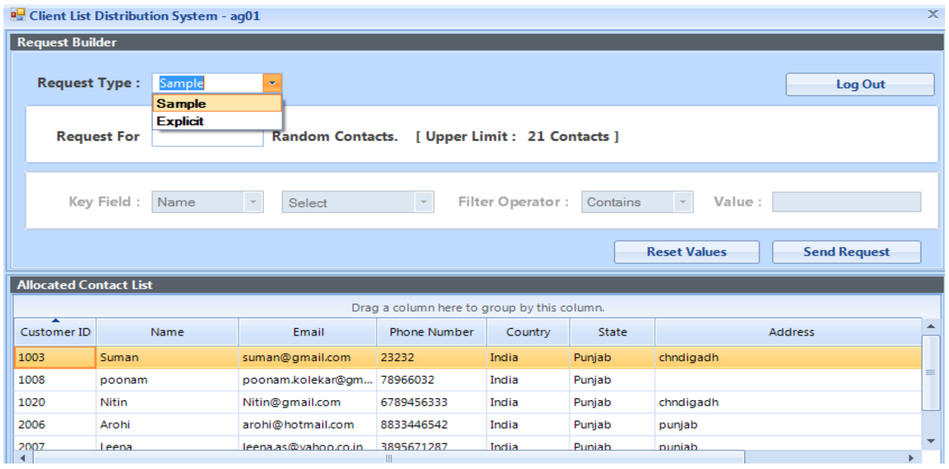


Fig 5: Distributor add/manage agents

2. Agent Module*:*

Some of the privileges are restricted to the agent by the distributor. Only few permissions are available with the agent. Agent has the following task/responsibilities. Agent has Read-only access to the content.

- Agent can Read the articles and download the Documents
- Read the messages which are send by the Distributor
- Discuss the content in the Discussion Board



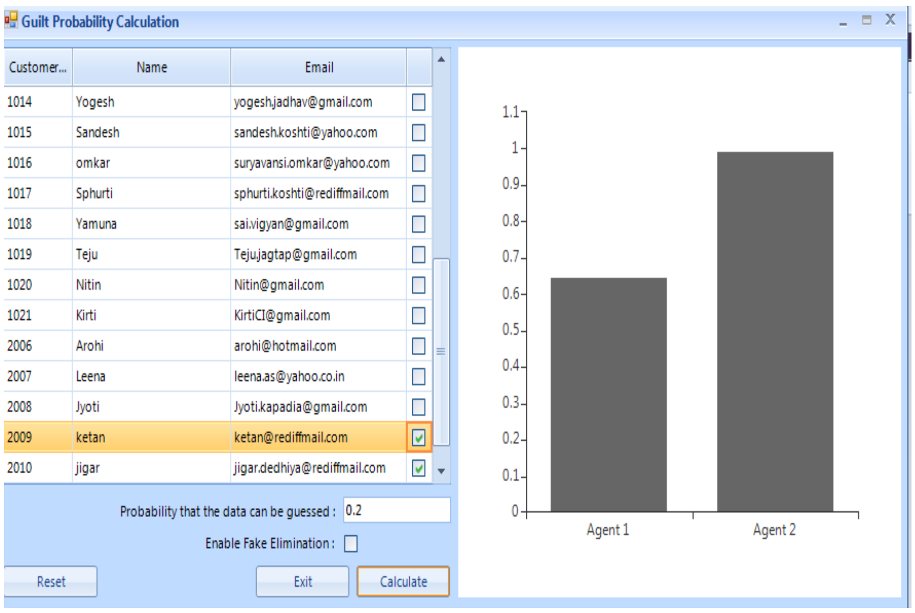
Customer ID	Name	Email	Phone Number	Country	State	Address
1003	Suman	suman@gmail.com	23232	India	Punjab	chndigadh
1008	poonam	poonam.kolekar@gm...	78966032	India	Punjab	
1020	Nitin	Nitin@gmail.com	6789456333	India	Punjab	chndigadh
2006	Arohi	arohi@hotmail.com	8833446542	India	Punjab	punjab
2007	Leena	leena.as@yahoo.co.in	3895671287	India	Punjab	punjab

Fig 6: Agent GUI

3. Data Leakage Detection*:*

The main scope of this module is to provide complete information about the data/content that is accessed by the users within the system.

- Forms Authentication technique will be used to provide security to the system in order to prevent the leakage of the data.
- Continuous observation will be made automatically and the information will be sent to the Distributor, so that he can identify whenever the data is leaked.
- Above all the important aspect providing proof against the Guilty Objects, Two techniques are used i.e. the Fake Object Generation technique and Data Allocation strategies



Customer...	Name	Email	
1014	Yogesh	yogesh.jadhav@gmail.com	☐
1015	Sandesh	sandesh.koshti@yahoo.com	☐
1016	omkar	suryavansi.omkar@yahoo.com	☐
1017	Sphurti	sphurti.koshti@rediffmail.com	☐
1018	Yamuna	sai.vigyan@gmail.com	☐
1019	Teju	Teju.jagtap@gmail.com	☐
1020	Nitin	Nitin@gmail.com	☐
1021	Kirti	KirtiCI@gmail.com	☐
2006	Arohi	arohi@hotmail.com	☐
2007	Leena	leena.as@yahoo.co.in	☐
2008	Jyoti	Jyoti.kapadia@gmail.com	☐
2009	ketan	ketan@rediffmail.com	✓
2010	jigar	jigar.dedhiya@rediffmail.com	✓

Probability that the data can be guessed : 0.2

Enable Fake Elimination : ☐

Fig 7: Calculation of guilt probability

IX. CONCLUSION

In a perfect world, there would be no need to hand over sensitive data to agents that may unknowingly or maliciously leak it. And even if we had to hand over sensitive data, in a perfect world, we could watermark each object so that we could trace its origins with absolute certainty.
However, in many cases, we must indeed work with agents that may not be 100 percent trusted, and we may not be certain if a leaked object came from an agent or from some other source, since certain data cannot admit watermarks. In spite of these difficulties, we have shown that it is possible to assess the likelihood that an agent is responsible for a leak, based on the overlap of his data with the leaked data and the data of other agents, and based on the probability that objects can be "guessed" by other means. Our model is relatively simple, but we believe that it captures the essential trade-offs. The algorithms we have presented implement a variety of data distribution strategies that can improve the distributor's chances of identifying a leaker. We have 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.

X. FUTURE WORK

Our future work is the extension of our allocation strategies so that they can handle agent requests in an online fashion (the presented strategies assume that there is a fixed set of agents with requests known in advance).Any application does not end with a single version. It can be improved to include new features. Our application is no different from this. The future enhancements that can be made to Data Leakage Detection are:

- Providing support for other file formats.
- Creation of a web based GUI for execution of the application.
- Improving the detection process based on user requirements.
- Provision of quality or accuracy variance parameter for the user to set.

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. Ms. Priyanka Desai, Dr. Sedamkar sir and Prof. Mr. Kiran Bhandari for invaluable guidance, encouragement and critics throughout the course of my project.

REFERENCES

[1] Papadimitriou and H. Garcia-Molina “Data leakage detection " *IEEE Transaction on knowledge and data engineering*, pages 51-63 volume 23,January 2011
[2] P. Papadimitriou and H. Garcia-Molina, “Data leakage detection”, *Technical report, Stanford University*, 2008.
[3] YIN Fan, WANG Yu, WANG Lina, Yu Rongwei. A Trustworthiness-Based Distribution Model for Data Leakage Detection: *Wuhan University Journal Of Natural Sciences.*
[4] P. Buneman, S. Khanna and W.C. Tan. Why and where: A characterization 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. Czerwinski, R. Fromm, and T. Hodes. Digital music distribution and audio watermarking.
[6] Rakesh Agrawal, Jerry Kiernan. Watermarking Relational Databases// IBM Almaden Research Center
[7] S. Jajodia, P. Samarati, M. L. Sapino, and V. S. Subrahmanian. Flexible support for multiple access control policies. *ACM Trans. Dataset Systems*, 26(2):214-260,2001.
[8] L. Sweeney. Achieving k- anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty*, Fuzzyness and Knowledge-based Systems-2002
[9] 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.
[10] Polisetty Sevani Kumari and Kavidi Venkata Mutyalu, ”Development of Data leakage Detection Using Data Allocation Strategies”, *International Journal of Computer Trends and Technology- volume3Issue4- 2012*
[11]ORLDatabase http://blogs.csoonline.com/1187/DataLeakage

AUTHORS

Jaymala Chavan, is a student of Thakur College of Engg. & Tech, Mumbai. She holds Bachelor of Engg. Degree in I.T. & currently pursuing M.E. She has a teaching experience of 3 years.

Priyanka Desai is working as Assistant Professor in Thakur College of Engg. & Tech, Mumbai. She holds B.E. & M.Tech. Degree in C.S.E. & currently pursuing Ph.D. She has a teaching experience of 9 years 4 months and an industry experience of 2 months.