By Applying Data Allocation Strategies and Guilt Model Analysis to Find Data Leakage Detection

KomalSurandase, PriyankaJadhavPallaviJadhav , AaratiPatil
Department of CSE, MIT College of Engg, University of Pune, PUNE, INDIA

Abstract -- In network many times user has to deal with sensitive data so it is needed to provide security while handling such type of data. In existing systems watermarking is used for data leakage detection. In watermarking there are many techniques such as unique code is embedded in data. But if this data is handled by any unauthorized person he can destroy watermarks. Sometimes watermarking leads to modify the original data. It may happen that any of agents gives this sensitive data to any unauthorized user without knowledge of owner. So 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 and what data has been leaked. To deal with such type of problems we have data allocation strategies and addition of fake records. These records are realistic but fake. Fake objects act similarly as that of watermarks but they do not modify the original data. Whenever distributor finds his data at unauthorized place or illegally the probability of guilty agents is calculated on the basis of calculation of fake objects which are added while distributing data.

Index Terms—Allocation strategies, data leakage, data privacy, fake records, leakage model

I.INTRODUCTION

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% 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 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 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. Perturbation is a technique in which the data is made “less sensitive” by modifying it and before being handed to agents. For e.g., add random noise to certain attributes, or replace exact values by ranges [15].

The owner of the data is called as distributor and the supposedly trusted third parties the agents. The goal is to detect [7] when the distributor’s sensitive data has been leaked by agents, and if possible to identify the agent that leaked [6] the data. In this paper, a model is developed for accessing the “guilt” of agents. Paper presents algorithms for distributing objects to agents, in a way that improves chances of identifying a leaker.

Traditionally, leakage detection is handled by watermarking, e.g., a unique code is embedded in each distributed copy. If that copy is later discovered in the hands of an unauthorized party, the leaker can be identified. Watermarks can be very useful in some cases, but again, involve some modification of the original data. Furthermore, watermarks can sometimes be destroyed if the data recipient is malicious[1]. *E.g.* 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 call the owner of the data the distributor and the supposedly trusted third parties the agents.

II. RELATED WORK

The guilt detection [7] approach presented in paper is related to the data provenance problem tracing the lineage of S objects implies essentially the detection of the guilty agents. Suggested solutions are domain specific, such as lineage tracing for data warehouses and assume some prior knowledge on the way a data view is created out of data sources. Leakage problem formulation [7] with objects and sets is more general and simplifies lineage tracing, since we do not consider any data transformation from Ri sets to S. As far as the data allocation strategies are concerned, main work is mostly relevant to watermarking that is used as a means of establishing original ownership of distributed objects. Watermarks were initially used in images, video and audio data whose digital representation includes considerable redundancy. Recently, and other works have also studied marks insertion to relational data. This approach and watermarking are similar in the sense of providing agents with some kind of receiver-identifying information. However, by its very nature, a watermark modifies the item being watermarked. If the object to be watermarked cannot be modified then a watermark cannot be inserted. In such cases methods that attach watermarks to the distributed data are not applicable. Finally, there are also lots of other works on mechanisms that allow only authorized users to access sensitive data through access control policies. Such approaches prevent in some sense data leakage by sharing information only with trusted parties

III.EXISTING SYSTEM

Traditionally, leakage detection is handled by watermarking, e.g., a unique code is embedded in each distributed copy. If that copy is later discovered in the hands of an unauthorized party, the leaker can be identified. Digital watermarking is the process of embedding information into a digital signal. The signal may be audio, pictures or video, for example. If the signal is copied, then the information is also carried in the copy.

In invisible watermarking, information is added as digital data to audio, picture or video, but it cannot be perceived as such. An important application of invisible watermarking is to copyright protection systems, which are intended to prevent or deter unauthorized copying of digital media. Steganography is an application of digital watermarking, where two parties communicate a secret message embedded in the digital signal. Annotation of digital photographs with descriptive information is another application of invisible watermarking. While some file formats for digital media can contain additional information called metadata, digital watermarking is distinct in that the data is carried in the signal itself. Problem with Water marking: Watermarks can be very useful in some cases, but again, involve some modification of the original data. Furthermore, watermarks can sometimes be destroyed if the data recipient is malicious. E.g. 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. Call the owner of the data the distributor and the supposedly trusted third parties the agents.

a. Overcoming Problem of Watermarking

Focus in this investigation is the identification of the problems that are arisen during executing program. This is another method called as unobtrusive. In this method, following functions are implemented.

By developing this software following objectives:

- Distributor “intelligently” give data to agents in order to improve the chances of detecting a guilty agent,
    1. Creating a fake object.
    2. Distributor creates and adds fake objects to the data that he distributes to agents.
    3. Detecting a guilty agent,

A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data is leaked and found in an unauthorized place (e.g., on the web or somebody’s laptop.

IV.PROBLEM DEFINITION AND DESCRIPTION

A .Data Leakage

In the course of doing business, sometimes sensitive data must be handed over to supposedly trusted third parties. 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. The distributor gives data to trusted third party over network.

A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data is 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.

B. Data Leakage Detection

Data Leakage Detection 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 distributor can also inject “realistic but fake” data records to further improve chances of detecting leakage and identifying the guilty party[1].

Distributor develops a model for assessing the “guilt” of agents. Project presents algorithms for distributing objects to agents, in a way that improves chances of identifying a leaker. Finally, Distributor also considers 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 acts as a type of watermark for the entire set, without modifying any individual members. 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.

C. Data Leakage Problem

Goal of this project 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. 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. However, in some cases it is important not to alter the original distributor's data. For example, if an outsourcer is doing our payroll, he must have the exact salary and customer bank account numbers.

1. 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.

- 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, also inject "realistic but fake" data records to further improve our chances of detecting leakage and identifying the guilty party[8].

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.

This involves the investigation of the existing system, which is time consuming with the user and is insufficient depth. This also includes the collection and study of detailed information and literature regarding the complete existing procedure.

The detailed initial study documented and the failing and problems are noted separately. The system is properly designed and proper outline of the proposed computerized system is prepared. The proposed design is brought against all the known facts and further proposal are made. Various resources including the software, hardware and manpower requirements are decided and are mentioned.

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. Perturbation is a very useful technique where the data is modified and made "less sensitive" before being handed to agents. We develop *unobtrusive* techniques for detecting leakage of a set of objects or records.

In this section we develop a model for assessing the "guilt" of 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 acts as a type of watermark for the entire set, without modifying any individual members. 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 guilt.

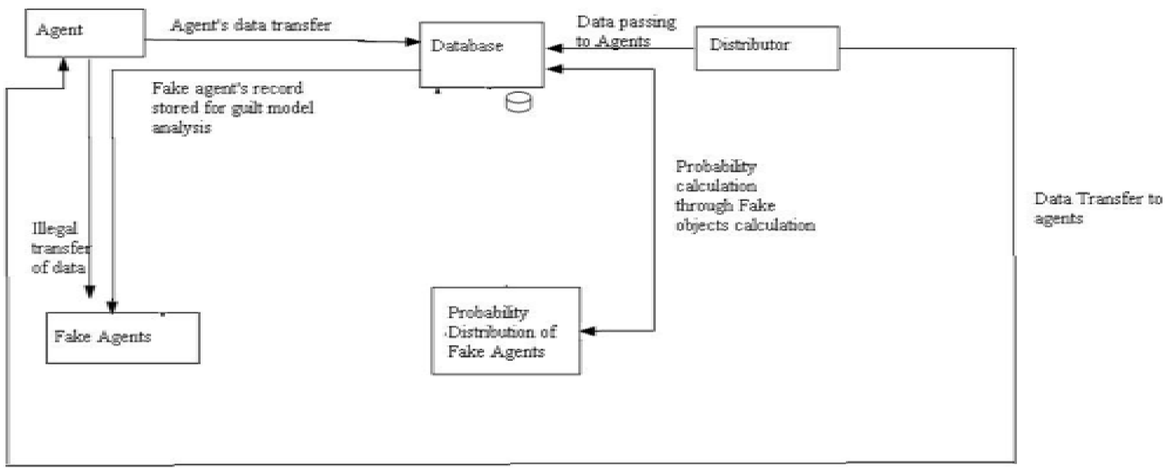
Agent
Agent's data transfer
Database
Data passing to Agents
Distributor
Fake agent's record stored for guilt model analysis
Probability calculation through Fake objects calculation
Data Transfer to agents
Illegal transfer of data
Fake Agents
Probability Distribution of Fake Agents

Fig.1 System Overview

V. Proposed Components

A. Data Allocation :

The main focus of our project is the data allocation problem as how can the distributor "intelligently" give data to agents in order to improve the chances of detecting a guilty agent, Admin can send the files to the authenticated user, users can edit their account details etc. Agent views the secret key details through mail. In order to increase the chances of detecting agents that leak data.

B. *Fake Object:*

The distributor creates and adds fake objects to the data that he distributes to agents. Fake objects are objects generated by the distributor in order to increase the chances of detecting agents that leak data. The distributor may be able to add fake objects to the distributed data in order to improve his effectiveness in detecting guilty agents. Our use of fake objects is inspired by the use of “trace” records in mailing lists. In case we give the wrong secret key to download the file the duplicate file is opened, and that fake details also send the mail. Ex: The fake object details will display.

C. *Optimization Module*

The Optimization Module is the distributor’s data allocation to agents has one constraint and one objective. The agent’s constraint is to satisfy distributor’s 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. User can able to lock and unlock the files for secure.

D. *Data Distributor Module*

A data distributor has given sensitive data to a set of supposedly trusted agents (third parties). Some of the data is 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 Admin can able to view the which file is leaking and fake user’s details also.

E. *Agent Guilt Module*

To compute this PrfGijSg, we need an estimate for the probability that values in S can be “guessed” by the target. For instance, say some of the objects in T are emails of individuals. We can conduct an experiment and ask a person with approximately the expertise and resources of the target to find the email of say 100individuals. If this person can find say 90 emails, then we can reasonably guess that the probability of finding one email is 0.9. On the other hand, if the objects in question are bank account numbers, the person may only discover say 20, leading to an estimate of 0.2. We call this estimate pt, the probability that object t can be guessed by the target. To simplify the formulas that we present in the rest of the paper, we assume that all T objects have the same pt, which we call p. Our equations can be easily generalized to diverse points though they become umber some to display. Next, we make two assumptions regarding the relationship among the various leakage events. The first assumption simply states that an agent’s decision to leak an object is not related to other objects.

Assumption 1: For all t; t0 2 S such that t 6= t0 the provenance of t is independent of the provenance of t0

To simplify our formulas, the following assumption states that joint events have a negligible probability[1]. As we argue in the example below, this assumption gives us more conservative estimates for the guilt of agents, which is consistent with our goals

Assumption 2: An object t 2S can only be obtained by the target in one of two ways:

A single agent Ui leaked t from his own Ri set; or
The target guessed (or obtained through other means) t without the help of any of the n agents, In other words, for all t 2 S, the event that the target guesses t and the events that agent Ui (i = 1; : : : ;n) leaks object t are disjoint.

Before we present the general formula for computing PrfGijSg, we provide a simple example. Assume that sets T, R’s and S are as follows:

T = ft1; t2; t3g; R1 = ft1; t2g; R2 = ft1; t3g; S = ft1; t2; t3g:…..(E question 1)

In this case, all three of the distributor’s objects have been leaked and appear in S. Let us first consider how the target may have obtained object t1, which was given to both agents. From Assumption 2, the target either guessed t1 or one of U1or U2 leaked it. We know that the probability of the former event is p, so assuming that the probability that each of the two agents leaked t1 is the same we have the following cases:

- the leaker guessed t1 with probability p;
- agent U1 leaked t1 to S with probability (1 p)=2agent
- U2 leaked t1 to S with probability (1 p)=2

Similarly, we find that agent U1 leaked t2 to S with probability 1 p since it is the only agent that has this data object.
Given these values, the probability that agent U1 is not guilty, namely that U1 did not leak either object is:

PrfG_1jSg = (1 (1 p)=2) _ (1 (1 p))…… (1)
Hence, the probability that U1 is guilty is:

PrfG1jSg = 1 Prf G_1jSg ………(2)

In the general case (with our assumptions), to find the probability that an agent Ui is guilty given a set S, first. we compute the probability that he leaks a single object t to S. To compute this we define the set of agents Vt =fUijt 2 that have t in their data sets. Then using Assumption 2 and known probability p, we have: Presume agent leaked t to Sg = 1 p…….. (3)Assuming that all agents that belong to Vt can leak t to S with equal probability and using Assumption 2 we obtain:

B.*Guilt Model Analysis:*
In order to see how our model parameters interact and to check if the interactions match our intuition, in this section, we study two simple scenarios. In each scenario, we have a target that has obtained all the distributor's objects, i.e., T ¼ S.

B.1 *Impact of Probability p:*
In our first scenario, T contains 16 objects: all of them are given to agent U1 and only eight are given to a second agent U2. We calculate the probabilities PrfG1jSgand PrfG2jSg for p in the range [0, 1] and we present the results in Fig. 1a. The dashed line shows PrfG1jSg and the solid line shows PrfG2jSg. As p approaches 0, it becomes more and more unlikely that the target guessed all 16 values. Each agent has enough of the leaked data that its individual guilt approaches 1. However, as p increases in value, the probability that U2 is guilty decreases significantly: all ofU2's eight objects were also given to U1, so it gets harder to blame U2 for the leaks.

On the other hand, U2's probability of guilt remains close to 1 as p increases, since U1 has eight objects not seen by the other agent. At the extreme, as p approaches 1, it is very possible that the target guessed all16 values, so the agent's probability of guilt goes to 0.

In this section, we again study two agents, one receiving all the T ¼ S data and the second one receiving a varying fraction of the data. Fig. 1b shows the probability of guilt for both agents, as a function of the fraction of the objects owned by U2, i.e., as a function of jR2 \ Sj=jSj. In this case, phase a low value of 0.2, and U1 continues to have all 16Sobjects. Note that in our previous scenario, U2 has 50percent of the S objects. We see that when objects are rare (p ¼ 0:2), it does not take many leaked objects before we can say thatU2 is guilty with high confidence. This result matches our intuition: an agent that owns even a small number of incriminating objects is clearly suspicious. Except for values of p equal to 0.5 and 0.9. We see clearly that the rate of increase of the guilt probability decreases as p increases. This observation again matches our intuition: As the objects become easier to guess, it takes more and more evidence of leakage (more leaked objects owned by U2) before we can have high confidence that U2 is guilty. In [14], we study an additional scenario that shows how the sharing of S objects by agents affects the probabilities that they are guilty. The scenario conclusion matches our intuition: with more agents holding the replicated leaked data, it is harder to lay the blame on any one agent.

VI. ALGORITHMS:

a. Sample Data Request Algorithm:
With sample data requests agents are not interested in particular objects. Hence, object sharing is not explicitly defined by their requests. The distributor is "forced" to allocate certain objects to multiple agents only if the number of requested objects exceeds the number of objects in set T. The more data objects the agents request in total, the more recipients on average an object has; and the more objects are shared among different agents, the more difficult it is to detect a guilty agent.

b. Explicit Data Algorithm:
In the first place, the goal of these experiments was to see whether fake objects in the distributed data sets yield significant improvement in our chances of detecting a guilty agent. In the second place, evaluate e-optimal algorithm relative to a random allocation.

Algorithm 1: Explicit Data Requests

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, R1, R2……,Rn 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 [5] every term of the objective summation by adding maximum number bi of fake objects to every set Ri yielding optimal solution.

Step 1: Calculate total fake records as sum of fake records allowed.
Step 2: While total fake objects > 0
Step 3: Select agent that will yield the greatest improvement in the sum objective i.e. i = arg max(-)∑j Ri ∩ Rj
Step 4: Create fake record
Step 5: Add this fake record to the agent and also to fake record set.
Step 6: Decrement fake record from total fake record set. Algorithm makes a greedy choice by selecting the agent that will yield the greatest improvement in the sum-objective.

Algorithm 2: Sample Data Requests

With sample data request each agent Ui may receive any T from a subset out of different ones. Hence there are different 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. Therefore, from the different allocations. An object allocation that satisfies requests and ignores the distributor's objective is to give each agent a 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:

Step 1: Initialize Min_overlap 1, the minimum out of the maximum relative overlaps that the allocations of different objects to Ui
Step 2: for k Є {k| tk Є Ri} do Initialize max_rel_ov 0, the maximum relative overlap between Ri and any set Rj that the allocation of tk to Ui
Step 3: for all j=1,….., n : j=I and tk Є Rj do Calculate absolute overlap as Abs_ov |RiRj|+1 Calculate relative overlap as Rel_ovabs_ov/min (mi , mj)
Step 4: Find maximum relative as max_rel_ovMAX(max_rel_ov, rel_ov) If max_rel_ovmin_overlap then min_overlapmax_rel_ovret_k k Return ret_k

It can be shown that algorithm s-max is optimal for the sum-objective and the max-objective in problems where, M |T| and n < |T|.

It is also optimal for the max- objective if |TI ≤ M ≤ 2 |T| or all agents request data of the same size. It is observed that the relative performance of algorithm and main conclusion do not change. If p approaches to 0, it becomes easier to find guilty agents and algorithm performance converges. On the other hand, if p approaches 1, the relative differences among algorithms grow since more evidence is needed to find an agent guilty. 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.

VI.DATA ALLOCATION PROBLEM

The distributor "intelligently" gives data to agents in order to improve the chances of detecting a guilty agent. There are four instances of this problem, depending on the type of data requests made by agents and whether "fake objects"[1] are to data list. Fake objects are objects generated by the distributor that are not in set T. The objects are designed to Leakage Problem Instances look like real objects, and are distributed to agents together with the T objects, in order to increase the chances of detecting agents that leak data. It represents four problem instances with the names EF, E , SF and S , where E stands for explicit requests, S for sample requests, F for the use of fake objects, and for the case where fake objects are not allowed.

The distributor may be able to add fake objects to the distributed data in order to improve his effectiveness in detecting guilty agents. Since, fake objects may impact the correctness of what agents do, so they may not always be allowable. Use of fake objects is inspired by the use of "trace" records in mailing lists.

The distributor creates and adds fake objects to the data that he distributes to agents. In many cases, the distributor may be limited in how many fake objects he can create. In EF problems, objective values are initialized by agents' data requests. Say, for example, that T = {t1, t2} and there are two agents with explicit data requests such that R1= {t1,t2 } and R2= {t1 }. The distributor cannot remove or alter the R1 or R data to decrease the overlap R1 \ R2. However, say the distributor can create one fake object (B = 1) and both agents can receive one fake object (B1 =B2 = 1). If the distributor is able to create more fake objects, he could further improve the objection.

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.

Our future work includes the investigation of agent guilt models that capture leakage scenarios that are not studied in this project. For example, what is the appropriate model for cases where agents can collude and identify fake tuples? A preliminary discussion of such a model is available in. Another open problem 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 UI 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

For all the efforts behind the paper work, we first& foremost would like to express our sincere appreciation to the staff of Dept. of CSE for their extended help & suggestions at every stage of this paper. Finally, we pay sincere thanks to all those who indirectly and directly helped us towards the successful completion of the paper.

REFERENCES

[1] Panagiotis Papadimitriou, Student Member, IEEE, and HectorGarcia-Molina, Member, Data Leakage Detection IEEE P.P (2-6)IEEEtransactions on knowledge and data engineering, vol. 23, no. 1,JANUARY 2011

[2] SandipA.Kale, Prof. Kulkarni S.V. *(Department Of Computer Sci.&Engg,MIT College of Engg, Dr.B.A.M.University, Aurangabad(M.S),India,* Data Leakage Detection: A Survey, *(IOSR Journal of ComputerEngineering (IOSRJCE)ISSN : 2278-0661 Volume 1, Issue 6 (July Aug2012), PP 32-35* www.iosrjournals.org

[3] IEEE Transactions On Knowledge And Data Engineering, Vol. 22,No. 3, March 2011 Data Leakage Detection Panagiotis Papadimitriou,Member, IEEE, Hector Garcia-Molina, Member, IEEE P.P (2,4-5)

[4] Faith M. Heikkila, Data Leakage: What You Need to Know, PivotGroup Information Security Consultant. P.P (1-3)

[5] Rudragouda G Patil*DeptOf CSE,* The Oxford College Of Engg,Bangalore.International Journal Of Computer Applications InEngineering Sciences [VOL I, ISSUE II, JUNE 2011] [ISSN: 2231-4946] P.P (1, 4) Development Of Data Leakage Detection Using DataAllocation Strategies

[6] 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.

[7] IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL., NO. 2010 Data Leakage Detection Panagiotis Papadimitriou, Member, IEEE, Hector Garcia-Molina, Member, IEEE

[8] Panagiotis Papadimitriou 1, Hector Garcia-Molina 2 StanfordUniversity 353 Serra Street, Stanford, CA 94305, USA P.P (1, 4-5)A Model for Data Leakage Detection

[9]Web-based Data Leakage Prevention Sachiko Yoshihama1, TakuyaMishina1, and Tsutomu Matsumoto2 1 IBM Research - Tokyo, Yamato,Kanagawa, Japan fsachikoy

[10] Joseph A. Rivela Senior Security Consultant P.P (4-6) DataLeakage: Affordable Data Leakage Risk Management

[11] Data Leakage Prevention: A news letter for IT Professionals Issue5 P.P (1-3)

[12] Archie Alimagno California Department of Insurance P.P (2-7),The Who, What, When & Why of Data LeakagePrevention/Protection

[13] An ISACA White Paper Data Leak Prevention P.P (3-7)

[14] Mr.V.Malsoru, NareshBollam/ REVIEW ON DATA LEAKAGEDETECTION , International Journal of Engineering Research andApplications (IJERA) ISSN: 2248-9622 www.ijera.com Vol. 1, Issue 3,pp.1088-1091 1088 | P a g e.

[15] L. Sweeney, "Achieving K-Anonymity Privacy Protection Using Generalization and Suppression," http://en.scientificcommons.org/43196131, 2002.

[16] Chun-Shien Lu, *Member, IEEE,* and Hong-Yuan Mark Liao, *Member, IEEE* MultipurposeWatermarking for Image Authentication and Protection