**DATA LEAKAGE DETECTION**

**ABSTRACT:** This report contains the details of the model to detect leakage of data. Currently watermarking technique is used for detecting the leaks but this technique is not completely secure.  Here we discuss new research on technique of detecting data leaks.

Consider a situation where a person has given data to a group of third person but later finds the same data in the hands of an unknown person. So, the owner must know who has leaked the data. So, we discuss Data leakage detection strategies (Across agents) that will help the owner to know the agent who has leaked the data. The proposed method will not depend on watermarking techniques.

**INTRODUCTION:**

In the 21st century, as technology improves daily and people rely increasingly on the internet to exchange data, both confidential and non-confidential, the data may be in a vulnerable state. This makes the confidentiality of the transferred data against leakage absolutely necessary. For instance, if person A provides confidential information to person B, there is a chance that person B could pose a threat to the business and end up leaking the information. To reduce data leakage, a number of fresh concepts including watermarking have been put forth.

Under this study, a model is created for locating the accountable parties. Algorithms are also suggested to distribute objects to agents in a way that improves the possibility of identifying the leaker. Additionally, the possibility of including bogus objects in the distribution is considered. These items don't correlate with actual things, yet toward the agents they seem genuine. The distributor can be more confident that an agent is guilty if it is discovered that the agent was allocated one or more artificial objects that were leaked. In addition, optimization is investigated in which the original data and the leaked information are compared, allowing speculation about the identity of the party that leaked the information.

Here we are employing an approximation approach to identify potential malicious agents. Our model can take on all requests from customers, regardless of their number. It provides data allocation strategies to raise the chances of detecting leakages. Furthermore, there is an application that allows a distributor to distribute and manage files containing sensitive data when customers send a request.

In order to do various surveys, we have seen that companies/firms hand over the precious data of the customers to some 3rd party. Let us take an example of a medical institute which has given the records of its patients to some 3rd party which claims that they need to do some research to invent new treatments. Similarly, we have seen many organisations which share customer data with 3rd parties. Let us call the owner as distributor and 3rd parties as agents. Our aim is to detect which agent has leaked the sensitive data of the distributor.

We will consider the scenario where the data cannot be bothered. We can use an analogy for our report. Consider we found ABC with a single toffee then ABC will argue that he got the toffee because someone gave him but if we found ABC with a bunch of toffees then it will be much difficult for ABC to argue. He can take legal action if they found the proof of leaked data**,**

In this project we propose a model to identify the guilty agents. To increase the probability of finding the leaker we also present the idea for give out objects to agents. Finally, we will also consider adding fakeobjects to the given data. These objects are not real objects but appear to be realistic. If it is found that 1 or more objects were provided to the agent then the Owner/distributor can be more assured about the agent’s guilt.

**LITERATURE SURVEY:**Instead of using primary keys, R. Sion, M. Atallah, and S. Prabhakar developed a technique to encoding watermarks within numeric large databases early 2003 like a method for ensuring integrity. To identify the exact location of such indicator, this technique sorted and partitioned that numeric group while also encoding small pieces of such imprint. The above method has developed to safeguard the information kept in database systems.

Utilizing generalizations & reduction approaches, K-anonymity privacy protection was introduced in 2002 to secure sensitive information from data providers. Whereas suppression means never revealing a value, generalization implies modifying or updating a significance by one that is lesser particular yet contextually compatible. Essentially guarantees that its facts are K-anonymous, meaning that each record has (k-1) or perhaps more identical data. These contents of such disclosed entries overlap with those of the sections inside the data sources indistinguishably.

Database systems might well be protected by watermarking, according to R. Agrawal and J. Kiernan's 2002 argument. Researchers defined essential prerequisites for something like an efficient watermarking system for this type of information while acknowledging its special characteristics of these kind of systems that provide new obstacles when watermarking.

One problem of information history inside a warehousing context entail identifying each data element to the primary source through which it was obtained. Throughout the case underlying general data warehouse modifications, Y. Cui and J. Wisdom outline that ancestry mapping issue and present a method for dealing with it.

**Limitation:** There are 2 major drawbacks of watermarking:

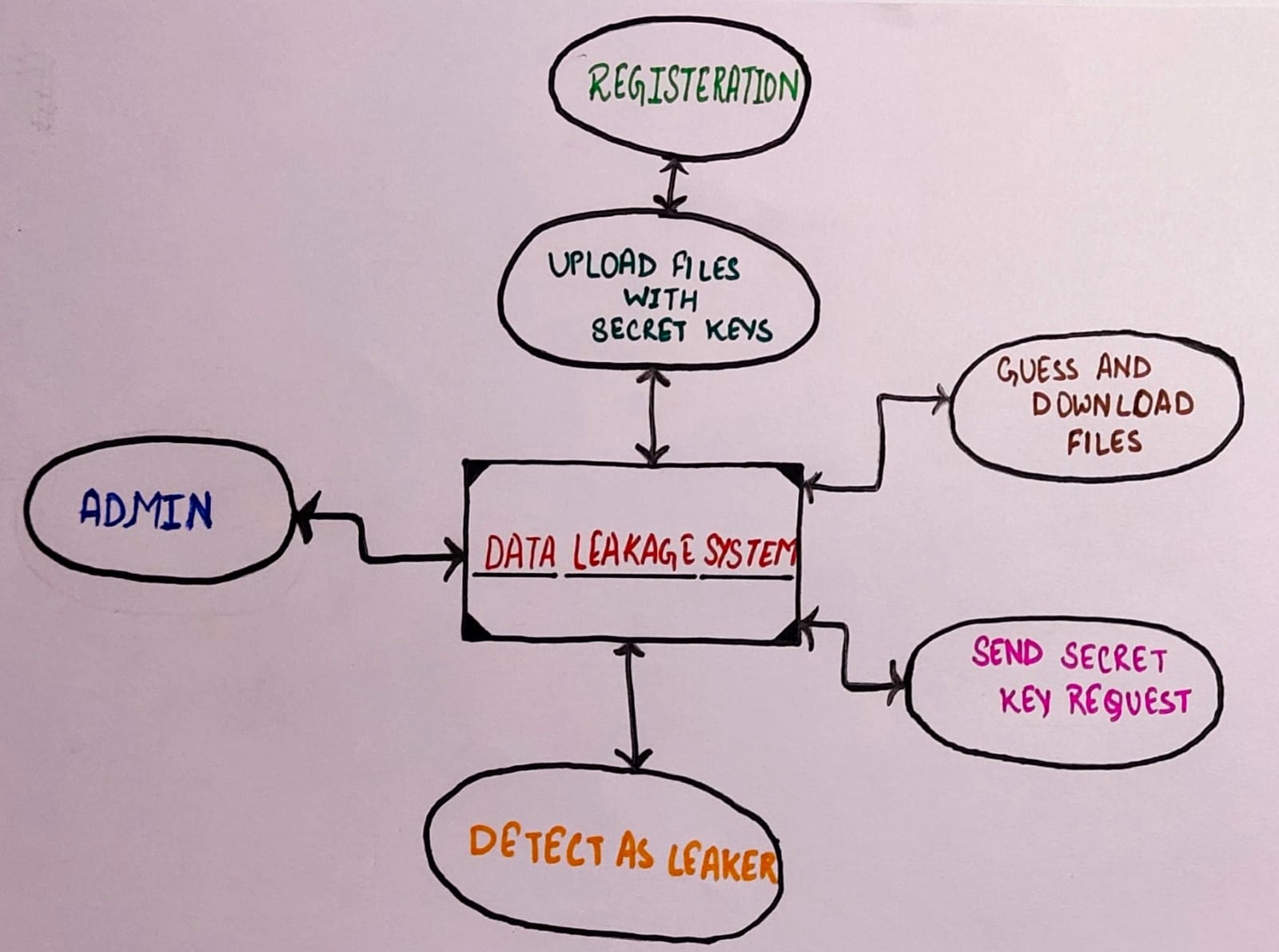
1. First is that it modifies the data. Modification/alteration of data is known as perturbation. In some cases, it is required not to alter the original data. For e.g. If agent doing any payroll then he should have Exact salary. Salary cannot be modified.
2. Watermarks can be destroyed by unethical means.

**Proposed System:**

We provide data allocation techniques that will improve the chances that leaks. If necessary, we will also employ fictitious but realistic data records to increase our chances of locating the leaker.

In the proposed model, to detect leakage of the set of data we use unobtrusive techniques. We'll also see the possibility of including "fake" objects in the distributed set. Although these things are not actual things, the agents see them as such. Alternately, we could claim that these fictitious items serve as a sort of watermark for the entire set without altering any data.

In more simpler terms the Proposed method is that whenever some user shares files with others there will be a secret key. If someone wants to download/view that file they will need the secret key which will be requested from the sender who has shared that file. When the Sender has shared the secret key, he can download it. But if someone downloaded the file without even requesting the secret key i.e. he used a guess method or he came to know about it from somewhere else then that will be highlighted as a leaker because he downloaded it without using the secret key.



The website has the feature of sending and receiving files in a secured manner which helps in preventing and detecting the leaks. It works in the following manner:

1. In the website there are different interfaces for admin and other users.
2. Interface of Admin provides the option to send files to other users.
3. When the user has received the file(s), he has to request for the key from the sender to download it.
4. Once the admin gets a request for the key, he can either share or decline the key.
5. The receiver has to use that secret key to download the file.
6. But if the user somehow got hold of the key through unethical means i.e without requesting the key then the user will get updated in the leaker list in the interface of admin.
7. The admin interface will have all details like date and time and admin can opt to deactivate the account of the leaker. This is how leakage of data is detected.

Modules used in system:

* 1. **Fake object module**: In hope to increase detection probability, the Owner may add “fake” objects to set of objects. We use “fake” entities in the way trace records are used in mailing lists. Consider Firm XYZ sells mailing list to Firm ABC to be used once. Firm XYZ adds trace records which contains addresses owned by Firm XYZ. So, every time Firm ABC uses the purchased mailing list, the firm XYZ will receive copies of the mails. These records can be thought of as a kind of fake objects that helps in identifying misuse of data.

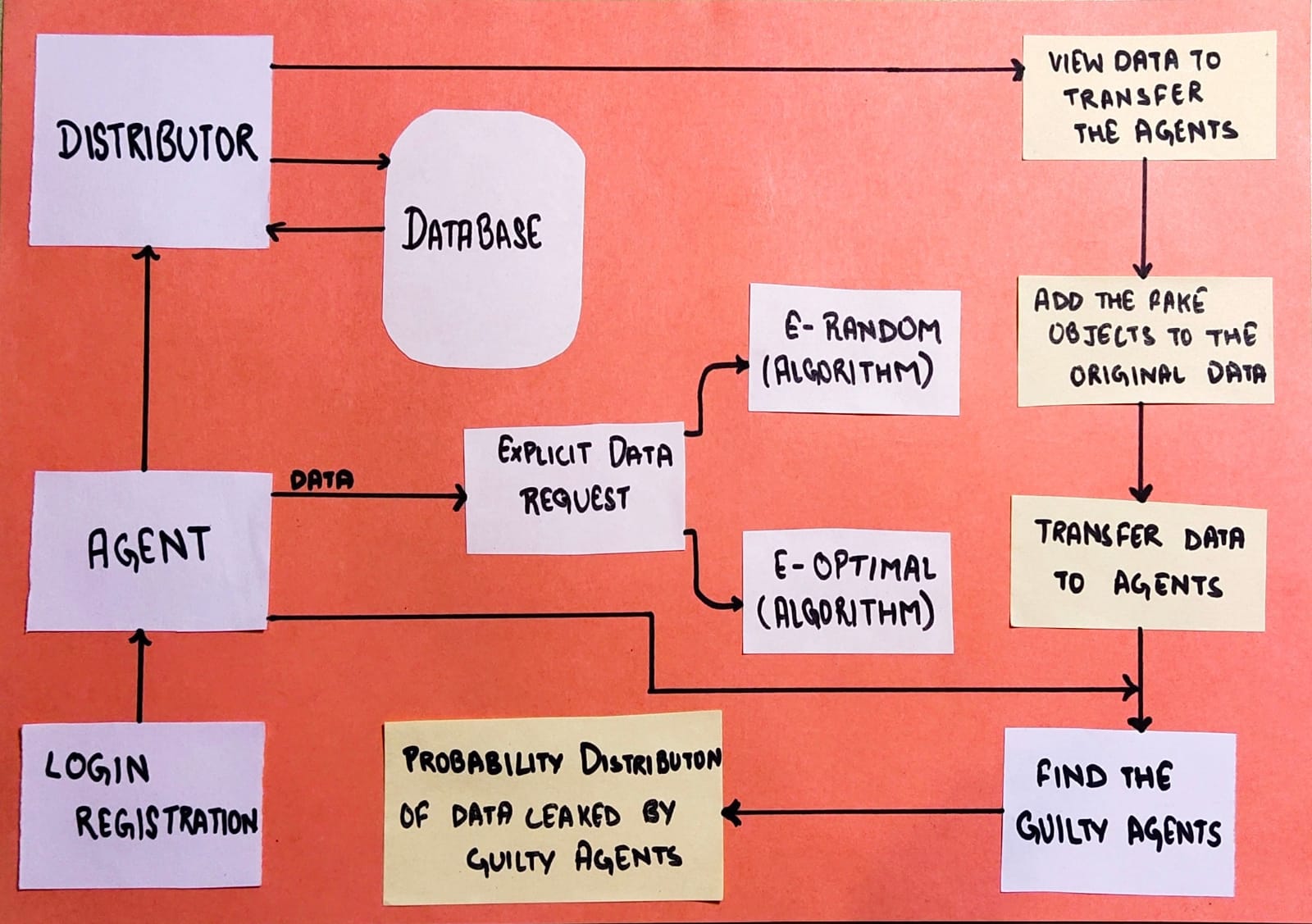
So, the distributor makes and add “fake” objects to the data. In case we provide wrong secret key for downloading file, the duplicate file is opened and fake details also sent on mail.

* 1. **Data allotment module**: The primary Focus is to how cleverly the distributor can give data to his agents so that probability of leakage detection increases?

The owner can forward the files to authorized user, users can also edit the details of the account. Agent sees the secret key details through e-mail.

* 1. **Module for Optimisation:** Owners’ data allocation has one objective and one constraint. The constraint is to satisfy the owner/distributor with the no. of objects they request with all the objects available. The objective is to identify the agent who leaked the data. For security purposes, the owner/Distributor should be able to lock/unlock the files.
  2. **Data Distributor module:** The leaked data has been found in the access of unauthorized agents while the owner/distributor has given data to trusted agents. Admin must know who has leaked the data. He should be able to view which files are leaked and the leaking agent’s details.
  3. **Agent Guilt module:** For this we need to estimate probability that carry values in S which can be predicted by our target/aim. For e.g., consider some of objects in Y are emails of persons. We can ask a person with resources of the target to find mails of 100 people. If the person can find 90 mails, then we can say that probability of finding one mail is0.9. If the objects are account nos. the person may be able to find only 20 leading the probability to 0.2.

We call this point the probability that object Y can be guessed by target. To simplify the formulae that are presented in next section, We consider all Y objects have same point, which we will call p. We will make 2 assumptions regarding the relationship among different events of leak.



**Result Analysis:**

**Agent guilt model:**

**Assumption1**: For all t, t’ ∈ S such that t 6= t 0 the provenance of t is independent of the provenance of t ‘.

Term provenance in statement cite the source of t value that appears in leaked set. Source can be any agent which have t in the set or the target itself .

The following assumption says that joint events have very less prob.

**Assumption2**: An object t ∈ S can only be obtained by the target in one of two ways:

1. A single agent Ui leaks t from its own Ri set
2. The target guessed without seeking any help of the n agents. In other words, for all t ∈ 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 P r{Gi |S}, lets consider a simple eg. Assume that sets T, R’s and S are as follows:

T = {t1, t2, t3}, R1 = {t1, t2}, R2 = {t1, t3}, S = {t1, t2, t3}.

In this case, all 3 distributor’s objects were leaked and appear in S. Lets first think about how the target would have got object t 1, which was given to both of the agents. From our assumption2, the target either guessed T 1 or one of U1 or U2 leaked it. We know probability of event is P, If the probability that each of the two agents leaked t1 is the same, then the following scenarios exist:

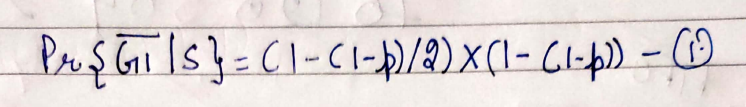
1.) Target predicted with probability p at t1

2.) Agent U1 exposed t1 to S using the probability (1-p)/2

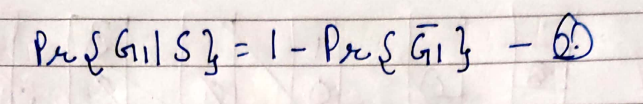
3.) Agent U2 compromised S by leaking t1. (1-p)/2

Because he is the sole agent with access to this data object, we may also discover that agent U 1 leaked t 2 to "S" with the prob(1-p).

We can compute the prob. that agent U 1 is not guilty provided that U 1 did not leak one of object:

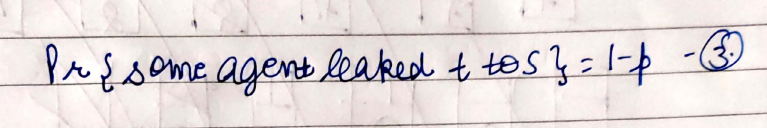


Hence, prob. that U1 is a guilty:

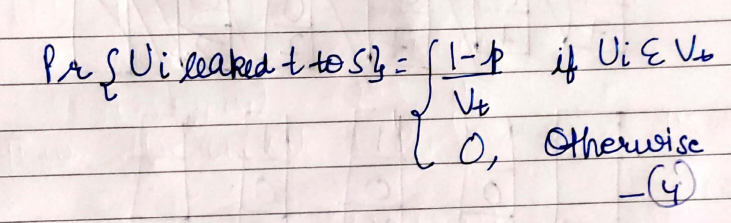


Our analysis would be more complex if Assumption2 did not hold, because we need to consider joint events eg. Target guesses T 1 and at the same time 1 or 2 agents leak value.

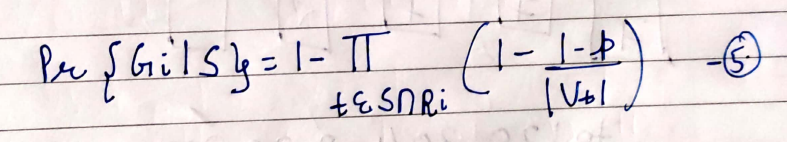
In general case, to find probability that agent Ui is guilty given a set S, we compute the probability that he leaks single object t to S. For this we define set of agents Vt = {Ui |t ∈ Ri} that have t in their datasets. Then by using Assumption2 and known prob. P, we can have



Considering all agents that belong to Vi can leak t to S with equal probability and by using Assumption2, we can obtain:



Provided that agent Ui is guilty if he leaks minimum 1 value to S, with Assumption1 and Equation 4 we can obtain probability P r{Gi |S} that agent Ui is guilty:



Guilt model Analysis:

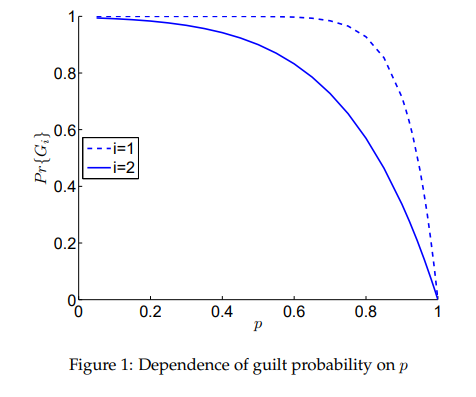
**Guilt Model Analysis:**

In our model we have to check and match  all the parameters with which our scenario parameters interact and to check if our scenario or match or not ,  we are  studying three simple scenarios. In each scenario we have a target that has obtained all the distributor’s objects, i.e. T = S

**Impact of Probability p:**

We calculate the probabilities P rG1 and P rG2 for p in the range [0,1] for the first scenario, which has 16 objects total and 8 objects each delivered to agent U1 and U2. Figure 1 displays the findings.

As p tends to  0 and  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 4 values, the probability that U2 is guilty decreases significantly: all of U2’s 8 objects were also given to U1, so it gets harder to blame U2 for the leaks. On the other hand, because U1 has 8 things that the other agent cannot see, U1's likelihood of guilt stays relatively constant as p rises. At the excrement, as p approaches 1, it is very possible that the target guessed all 16 values, so the agent’s probability of guilt goes to 0.



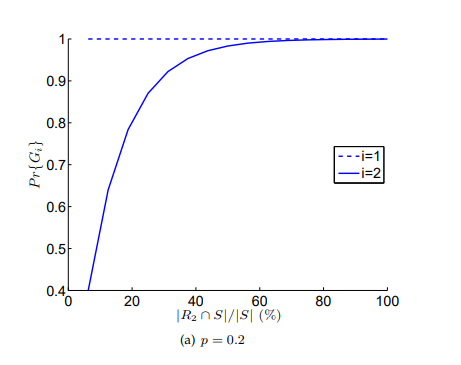
**Dependence on Overlap of Ri with S:**

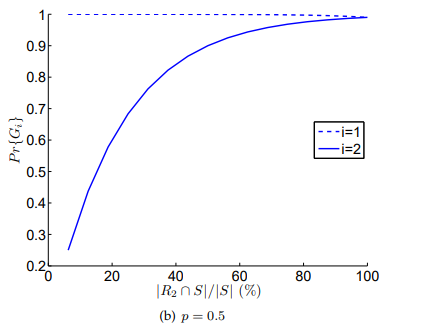
Inthis section we have two agents one is receiving a varying fraction of the data and 2nd one is T=S data . as a function of |R2 ∩ S|/|S|. Under the given circumstances, p has a low value of 0.2 and U1 still has all 16 S objects. Note that in our previous scenario, U 2 has 50S objects. when objects are rare (p = 0.2), it does not take many leaked objects before we can say U2 is guilty with high trust . This result matches our scenario an agent that  posses a  small number of implicate objects is clearly doubtful. in the same scenario, excluding for values of p equal to 0.5 and 0.9. We can see clearly that the rate of increase of the guilt probability decreases as p increases. This observation confirms our Scenario once again. It takes more and more evidence of leaking as the objects become more predictable before we can say with certainty that U2 is at fault.

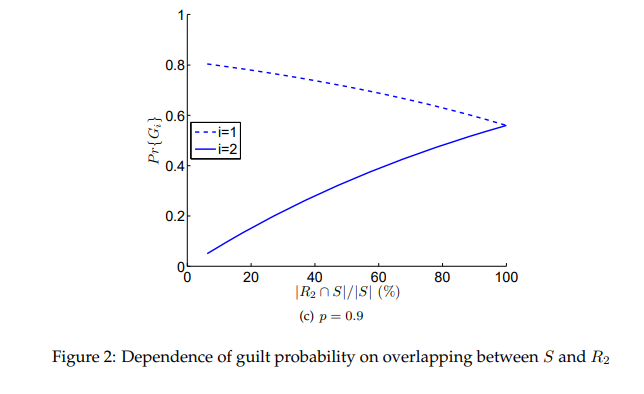
**Dependence on the Number of Agents with Ri Data:**

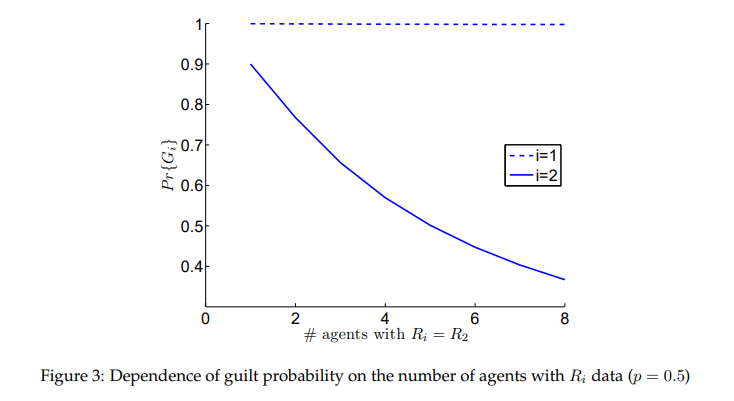
In this section, we will see how probabilities are impacted by the sharing of S objects by the agents that are guilty. Let us start with one agent U 1 with 16 T = S objects, & a 2nd agent U2 with 8 objects & P=0.5. Then we will increase the No. of agents to have same 8 objects.

For each scenario, we determine the probability P rG1 and P rG2, and the results are shown in Fig. 3. The number of agents with 8 objects is shown on the x-axis. For instance, the beginning point with U 1 and U2 is the leftmost value of x = 1. The rightmost point, where x = 8, depicts U2 with 8 agents and the same 8 objects as U1, each with 16 items. We note that the increase in the number of agents has little to no impact on P rG1 (dotted line). This insensitivity is caused by the fact that U1 is the owner of 8 of the leaked items that are unique. On the other hand, as more agents are added, the probability of guilt for the remaining agents (solid line) gradually decreases.









**Optimization Problem:**

We use alternatively the following objective to maximise the possibility of detecting a guilty agent that leaks all his data objects. We now initiate  some notation to formally the issue objective. Remember that P r{Gj |S = Ri} or simply P r{Gj |Ri} is the probability that agent Uj is guilty if the distributor locates a leaked tabletS that contains all Ri objects. We define the difference functions ∆(i, j) as:

                                   ∆(i, j) = P r{Gi |Ri} − P r{Gj |Ri} i, j = 1, . . . , n

So that dissimilarity  ∆ have non-negative values  given that set Ri has all the leaked objects, agent Ui is at least as likely to be guilty as any other agent. Difference ∆(i, j) is +ve for any agent U j , whose set R j does not have all data of “S”. It is zero, if Ri ⊆ Rj . In this case the distributor will observe the pair agents Ui and Uj identically guilty as they have both collected all the leaked objects. The larger a ∆ (i, j) value  is to identify U i as the leaking agent

**Conclusion:**

 So, we obtain that the representation of the data leakage detections system is very beneficial as opposed to the water marking model. All around the pass on data from one system to another our surety is maintained throughout the convey of data in many cases we have to do work with agents so it may not be guaranteed so we overcome the difficulties to the possibility that behind the leak of data is agent. It is generally built on overlap of leak data and other data which is came from by agent and build on a prospect that can be estimate by other means and our representation is very uncomplicated but we have trust it takes the crucial exchange.

REFERENCES:

1. Kale, S. A., & Kulkarni, S. V. (2012). Data leakage detection. *International Journal of Advanced Research in Computer and Communication Engineering*, *1*(9), 668-678.
2. Bollam, N., & Malsoru, M. V. (2011). Review on Data Leakage Detection. *International Journal of Engineering Research and Applications (IJERA)*, *1*(3), 1088-1091.
3. Vaidya, A., Lahange, P., More, K., Kachroo, S., & Pandey, N. (2012). Data leakage detection. *International Journal of Advances in Engineering & Technology*, *3*(1), 315.
4. Patil, R. G. (2011). Development of Data leakage Detection Using Data Allocation Strategies. *International Journal of Computer Applications in Engineering Sciences*, *1*.
5. Kavali, U., Abhang, T., & Narawade, M. V. (2012). Data Allocation Strategies in Data Leakage Detection. *International Journal of Engineering Research and Application*, *2*(2).
6. Singh, M. A. M. T. A., Tripathi, P. R. I. T. I., & Singh, R. E. N. U. K. A. (2013). Detection of data leakage. *International Journal of Computer & Communication Technology*, *4*(3), 22-24.