ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

Data Leakage Detection

Sandip A. Kale[1], Prof. S.V.Kulkarni[2]

Department Of CSE, MIT College of Engg, Aurangabad, Dr.B.A.M.University, Aurangabad (M.S), India[1,2]

ABSTRACT: **This paper contains the results of implementation of Data Leakage Detection Model. Currently watermarking technology is being used for the data protection. But this technology doesn't provide the complete security against date leakage. This paper includes the difference between the watermarking & data leakage detection model's technology. This paper leads for the new technique of research for secured data transmission & detection, if it gets leaked.**

Keywords: watermarking, robust watermarking, spectrum, guilty agent, implicit request, explicit request.

I. INTRODUCTION

Data leakage is the big challenge in front of the industries & different institutes. Though there are number of systems designed for the data security by using different encryption algorithms, there is a big issue of the integrity of the users of those systems. It is very hard for any system administrator to trace out the data leaker among the system users. It creates a lot many ethical issues in the working environment of the office.

The data leakage detection industry is very heterogeneous as it evolved out of ripe product lines of leading IT security vendors. A broad arsenal of enabling technologies such as firewalls, encryption, access control, identity management, machine learning content/context-based detectors and others have already been incorporated to offer protection against various facets of the data leakage threat. The competitive benefits of developing a "one-stop-shop", silver bullet data leakage detection suite is mainly in facilitating effective orchestration of the aforementioned enabling technologies to provide the highest degree of protection by ensuring an optimal fit of specific data leakage detection technologies with the "threat landscape" they operate in. This landscape is characterized by types of leakage channels, data states, users, and IT platforms.

II. 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. 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. We call the owner of the data the distributor and the supposedly trusted third parties the agents.

II. PROPOSED SYSTEM

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 guilty.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

III. TECHNIQUES OF WATERMARKING SYSTEM

A. *Embedding and Extraction*

In this technique the insignificant portion of the fractional part of the pixel intensity value of the cover image is encoded to provide watermark. A watermark in the insignificant part has helped to maintain the fidelity of the cover image. As seen from the results, imperceptibility is well preserved. Large capacity of watermarking is an added advantage of this scheme. Thus, large capacity watermark may be successfully embedded and extracted using this scheme, which can be extremely useful for companies engaged in developing watermarking applications and digital information security products. Embedding and extraction algorithms are used in this technique.

B. *Secure Spread Spectrum Watermarking*

We describe a digital watermarking method for use in audio, image, video and multimedia data. We argue that a watermark must be placed in perceptually significant components of a signal if it is to be robust to common signal distortions and malicious attack. However, it is well known that modification of these components can lead to perceptual degradation of the signal. To avoid this, we propose to insert a watermark into the spectral components of the data using techniques analogous to spread spectrum communications, hiding a narrow band signal in a wideband channel that is the data. The watermark is difficult for an attacker to remove, even when several individuals conspire together with independently watermarked copies of the data. It is also robust to common signal and geometric distortions such as digital-to-analog and analog-to-digital conversion, resampling, quantization, dithering, compression, rotation, translation, cropping and scaling. The same digital watermarking algorithm can be applied to all three media under consideration with only minor modifications, making it especially appropriate for multimedia products. Retrieval of the watermark unambiguously identifies the owner, and the watermark can be constructed to make counterfeiting almost impossible. We present experimental results to support these claims.

C. *DCT-Based Watermarking*

The image is first divided into 8 × 8 pixel blocks. After DCT transform and quantization, the midfrequency range DCT coefficients are selected based on a Gaussian network classifier. The mid-frequency range DCT coefficients are then used for embedding. Those coefficients are modified using a linear DCT constraints. It is claimed that the algorithm is resistant to JPEG compression.

D. *Spread Spectrum*

Cox et al. [1997] [2] used the spread spectrum to embed the watermark in the frequency components of the host image. First the Fourier Transform is applied to the host image is inserted to obtain a modified values V_ using the following equation: $V\_ = V + \alpha \times W$.
The scaling parameter α is used to determine the embedding strength of the watermark. Different spectral components exhibit different tolerance to modification. To verify the presence of the watermark, the cross correlation value between the extracted watermark W_ and the original watermark W is computed as follows:
sim =W_ × WT (W_ × W_T)(W × WT)
Here, we call the cross correlation the similarity (sim). Experimental results showed that this method resists JPEG compression with a quality factor down to 5%, scaling, dithering, cropping and collusion attacks.

E. *Wavelet Based Watermarking*

The multi resolution data fusion is used for embedding where the image and the watermark are both transformed into the discrete wavelet domain. The watermark is embedded into each wavelet decomposition level of the host image. During detection, the watermark is an average of the estimates from each resolution level of wavelet decomposition. This algorithm is robust against JPEG compression, additive noise and filtering operations.

F. *Robust Watermarking Technique*

Contrary to the LSB approach, the key to making a watermark robust is that it should be embedded in the perceptually significant components of the image. A good watermark is one which takes into account the behavior of human visual system. For the spread spectrum based watermarking algorithm, a scaling factor can be used to control the amount of energy a watermark has. The watermark energy should be strong enough to withstand possible attacks and distortions. Meanwhile large watermark energy will affect the visual quality of the watermarked image. A perceptual model is needed to

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

adjust the value of the scaling factor based on the visual property of the host image to achieve the optimal trade-off between robustness and invisibility.

G. *Invisible Watermarking*

This technique presents a novel invisible robust watermarking scheme for embedding and extracting a digital watermark in an image. The novelty lies in determining a perceptually important sub image in the host image. Invisible insertion of the watermark is performed in the most significant region of the host image such that tampering of that portion with an intention to remove or destroy will degrade the esthetic quality and value of the image. One feature of the algorithm is that this sub image is used as a region of interest for the watermarking process and eliminates the chance of watermark removal. Another feature of the algorithm is the creation of a compound watermark using the input user watermark (logo) and attributes of the host image. This facilitates the homogeneous fusion of a watermark with the cover image, preserves the quality of the host image, and allows robust insertion-extraction. Watermark creation consists of two distinct phases. During the first phase, a statistical image is synthesized from a perceptually important sub image of the image. A compound watermark is created by embedding a watermark (logo) into the statistical synthetic image by using a visible watermarking technique. This compound watermark is invisibly embedded into the important block of the host image. The authentication process involves extraction of the perceptive logo as well statistical testing for two-layer evidence. Results of the experimentation using standard benchmarks demonstrates the robustness and efficacy of the proposed watermarking approach. Ownership proof could be established under various hostile attacks.

H. *Watermarking of Digital Audio and Image using Mat lab Technique*

Watermarking, a Watermark is encrypted using RSA Algorithm and is embedded on the audio file using LSB technique. LSB technique is an old technique which is not very robust against attacks. Here, in audio watermarking we have embedded the encrypted watermark on the audio file, due to which removal of the watermark becomes least probable. This would give the technique a very high robustness. In the retrieval, the embedded watermark is retrieved and then decrypted. This method combines the robustness of Transform domain and simplicity of spatial domain methods. For image Watermarking, DWT technique is used. DWT technique is used in Image watermarking. Here, the watermark is embedded in the image as a pseudo-noise sequence. This gives a remarkable security to the image file as only if the exact watermark is known can the embedded watermark be removed from the watermarked image.

I. *Watermarking While Preserving the Critical Path*

The first intellectual property protection technique using watermarking that guarantees preservation of timing constraints by judiciously selecting parts of the design specification on which watermarking constraints can be imposed. The technique is applied during the mapping of logical elements to instances of realization elements in a physical library. The generic technique is applied to two steps in the design process: combinational logic mapping in logic synthesis and template matching in behavioural synthesis. The technique is fully transparent to the synthesis process, and can be used in conjunction with arbitrary synthesis tools. Several optimization problems associated with the application of the technique have been solved. The effectiveness of the technique is demonstrated on a number of designs at both logic synthesis and behavioural synthesis.

J. *Buyer-seller watermarking protocols*

This technique integrates watermarking techniques with cryptography, for copyright protection, piracy tracing, and privacy protection. In this paper, we propose an efficient buyer seller watermarking protocol based on homomorphism public-key cryptosystem and composite signal representation in the encrypted domain. A recently proposed composite signal representation allows us to reduce both the computational overhead and the large communication bandwidth which are due to the use of homomorphism public-key encryption schemes.
Both complexity analysis and simulation results confirm the efficiency of the proposed solution, suggesting that this technique can be successfully used in practical applications.

K. *Watermarking using Cellular Automata Transform*

Another watermarking technique is using cellular automata transform. An original image is CA-

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

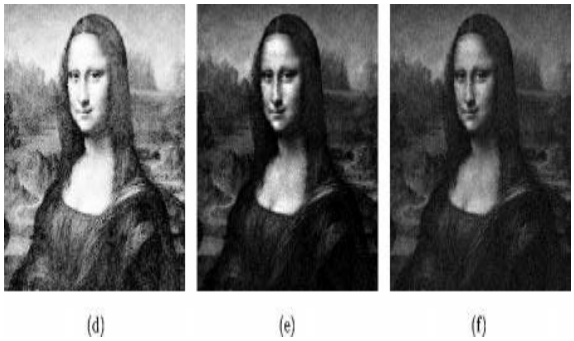
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

transformed and watermark is embedded in to coefficients of CA-transformed pattern. This watermarking model has flexibility in data hiding. it is possible to embed watermark in many CAT plans with different rule number parameters and CA bases class of CAT and all kind of image models such as shape, letter, photo can be used as watermark data. Using CAT with various rule number parameters, it is possible to get many channels for embedding.

V. RESULTS OF WATERMARK SYSTEM

A. *Results of Fragile Watermarking*

The degree of fragility was verified using the gray-scale "MonaLisa" image, size 256 256, as shown in Fig. 4(a). The length of a watermark depends on both the host image and the wavelet-based visual model. Here, its length was dynamically determined to 6593. Using cocktail watermarking [19],13 186 wavelet coefficients were modulated. The PSNR of the watermarked image shown in Fig. 4(b) was 39.7 dB. Next, the watermarked MonaLisa image was maliciously modified at the position of her face by means of texturing, as shown in Fig. 4(c). We wanted to see whether our fragile watermarks were sensitive to texture changes. Figs. 4(d)–(f) show when , the tampering detection results at different scales. Figs. 4(g)–(i) show another set of results when . It is found that in Fig. 4 that the altered regions were almost located. It is worth noticing that for different values, the difference between and only slightly reduced even when has been changed from one to ten. This implies that our multipurpose watermarking scheme is indeed fragile enough because the change of would not affect fragility significantly. As for color images, the beach image with size 512 512 (shown in Fig. 5) was also used to demonstrate the fragility of our approach. The watermarks were embedded in the illumination channel and the PSNR was 41.2 dB [Fig. 5(b)].



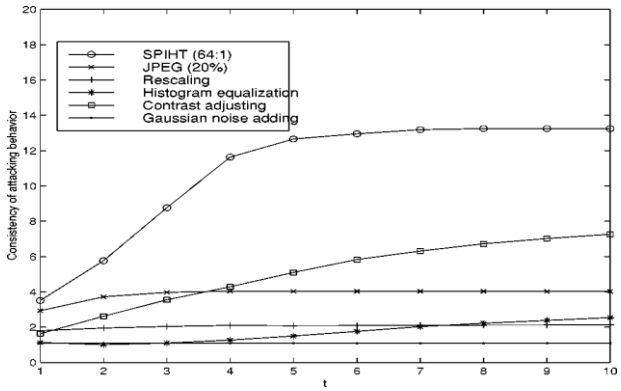


Fig. 1. Fragile watermarks facing incidental tampering: (a) *SPIHT* with compression ratio 64 : 1; (b) JPEG with quality factor 20% (compression ratio 20 : 1); (c) rescaled; (d) histogram equalized; (e) contrast adjusted; (f) Gaussian noise added; and (g) the BR values obtained at different t(1 _ t _ 10) with respect to six distinct incidental manipulations.

B. *Results of Robust Watermarking*

In this section, we shall discuss the experimental results with regard to robust watermarking. The same watermarked "MonaLisa" image [Fig. 4(b)], used for fragile test in the previous section, had also been used for robustness test in [20] under several attacks. Here, we used a different image for robust watermarking to demonstrate that our scheme adapts to different images.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

The "sailboat" image with size 256 256 was used to evaluate the robustness of our scheme. After watermarking, 23 different attacks including blurring, median filtering, rescaling, histogram equalization, jitter attack, changing the brightness/ contrast, the negative film effect, segmentation, Gaussian noise adding, mosaicing, sharpening, texturizing, shading, the ripple effect, net dotting, uniform noise adding, the twirl effect, *SPIHT* compression, *JPEG* compression, StirMark2 [27], dithering, pixel spreading, and cropping were selected to test the robustness of our watermarking scheme. Fig. 8 shows the robust watermark detection results. The lowest detector response as shown in Fig. 8 was 0.32 (the ninth attack), which corresponds to the Gaussian noise attack. We used the worst result to verify the uniqueness requirement, i.e., to show the false positive probability.

Fig. 9 shows the detector responses with respect to 10 000 random marks (including the hidden one, i.e., the 500th mark). It is obvious that the response with respect to the hidden one is a recognizable spike.

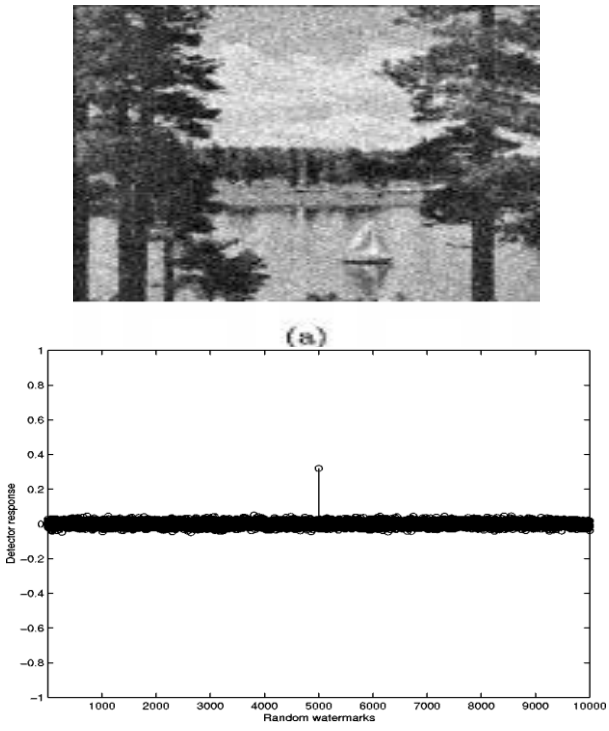


Fig. 2. Uniqueness verification of robust watermarking under a Gaussian noise adding attack (a) attacked image after the Gaussian noise was added and (b) the detector responses of the extracted watermark with respect to 10 000 random marks (including the hidden the hidden one, the 5000th mark).

VI. MODULES OF DATA LEAKAGE DETECTION SYSTEM

A. *Data Allocation Module*

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 Module*

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

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

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 100 individuals. 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 pt's though they become cumbersome 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.

IV. RESULTS OF DATA LEAKAGE DETECTION MODEL

A. ***Agent Guilt Model***

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 100 individuals. 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 pt's though they become cumbersome 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. 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 2 S 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:…..(Eqn 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 U1 or 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)=2

agent 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 Rig 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)

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

Assuming that all agents that belong to Vt can leak t to S with equal probability and using Assumption 2 we obtain:

$$Pr\{U_i \text{ leaked } t \text{ to } S\} = \begin{cases} \frac{1-p}{|V_t|}, & \text{if } U_i \in V_t \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

……..(Eqn 2)

Given that agent Ui is guilty if he leaks at least one value to S, with Assumption 1 and Equation 4 we can compute the Probability PrfGijSg that agent Ui is guilty:

$$Pr\{G_i|S\} = 1 - \prod_{t \in S \cap R_i} \left(1 - \frac{1-p}{|V_t|}\right) \quad (5)$$

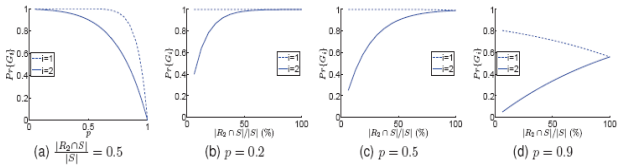
……..(Eqn 3)

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 PrfG1jSg and 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 isguilty decreases significantly: all of U2's eight objects were also given to U1, so it gets harder to blame U2 for the leaks.



Graph 1 of Impact of Guilt Probability p

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 all 16 values, so the agent's probability of guilt goes to 0. 5.2 Impact of Overlap between Ri and S 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, p has a low value of 0.2, and U1 continues to have all 16S objects. Note that in our previous scenario, U2 has 50 percent 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 that U2 is guilty with high confidence. This result matches our intuition: an agent that owns even a small number of incriminating objects is clearly suspicious. Figs. 1c and 1d show the same scenario, 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.

B.2 *Experimental Results*

We implemented the presented allocation algorithms in Python and we conducted experiments with simulated data leakage problems to evaluate their performance. In Section 8.1, we present the metrics we use for the algorithm evaluation, and in Sections 8.2 and 8.3, we present the evaluation for sample requests and explicit data requests, respectively. 8.1 Metrics In Section 7, we presented algorithms to optimize the problem of (8) that is an approximation to the original optimization problem of (7). In this section, we evaluate the presented algorithms with respect to the original problem. In this way, we measure not only the algorithm performance, but also we implicitly evaluate how effective the approximation is. The objectives in (7) are the _ difference functions. Note that there are nðn _ 1Þ objectives, since for each agent Ui, there are n _ 1 differences _ði; jÞ for j ¼ 1; . . . ; n and j 6¼ i.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

We evaluate a given allocation with the following objective scalarizations as metrics:

$$\overline{\Delta} := \frac{\sum_{i,j=1,\dots,n \atop i \neq j} \Delta(i,j)}{n(n-1)},$$
$$\min \Delta := \min_{i,j=1,\dots,n \atop i \neq j} \Delta(i,j).$$

(Eqn 4)

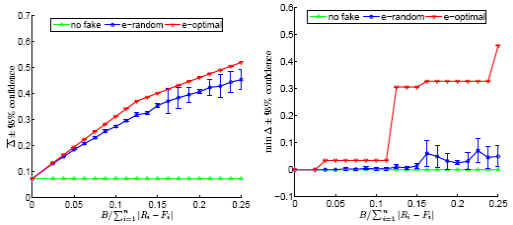
Metric _ is the average of _ði; jÞ values for a given allocation and it shows how successful the guilt detection is, on average, for this allocation. For example, if _ ¼ 0:4, then, on average, the probability PrfGijRig for the actual guilty agent will be 0.4 higher than the probabilities of non guilty agents. Note that this scalar version of the original problem objective is analogous to the sum-objective scalarization of the problem of (8). Hence, we expect that an algorithm that is designed to minimize the sum-objective will maximize _.

Metric min _ is the minimum _ði; jÞ value and it corresponds to the case where agent Ui has leaked his data and both Ui and another agent Uj have very similar guilt probabilities. If min _ is small, then we will be unable to identify Ui as the leaker, versus Uj. If min _ is large, say, 0.4, then no matter which agent leaks his data, the probability that he is guilty will be 0.4 higher than any other non guilty agent. This metric is analogous to the max-objective scalarization of the approximate optimization problem. The values for these metrics that are considered acceptable will of course depend on the application. In particular, they depend on what might be considered high confidence that an agent is guilty. For instance, say that PrfGijRig ¼ 0:9 is enough to arouse our suspicion that agent Ui leaked data. Furthermore, say that the difference between PrfGijRig and any other PrfGjjRig is at least 0.3. In other words, the guilty agent is ð0:9 _ 0:6Þ=0:6 _ 100% ¼ 50% more likely to be guilty compared to the other agents.

In this case, we may be willing to take action against Ui (e.g., stop doing business with him, or prosecute him). In the rest of this section, we will use value 0.3 as an example of what might be desired in _ values. To calculate the guilt probabilities and _ differences, we use throughout this section p ¼ 0:5. Although not reported here, we experimented with other p values and observed that the relative performance of our algorithms and our main conclusions 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.

B.3 *Explicit Requests*

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, we wanted to evaluate our e-optimal algorithm relative to a random allocation. We focus on scenarios with a few objects that are shared among multiple agents. These are the most interesting scenarios, since object sharing makes it difficult to distinguish a guilty from non guilty agents. Scenarios with more objects to distribute or scenarios with objects shared among fewer agents are obviously easier to handle. As far as scenarios with many objects to distribute and many overlapping agent requests are concerned, they are similar to the scenarios we study, since we can map them to the distribution of many small subsets.



Graph.2 Evaluation of Explicit Data Requests (1)

In our scenarios, we have a set of jTj ¼ 10 objects for which there are requests by n ¼ 10 different agents. We assume that each agent requests eight particular objects out
of these 10. Hence, each object is shared, on average, among Pn i¼1 jRij jTj ¼ 8 agents.

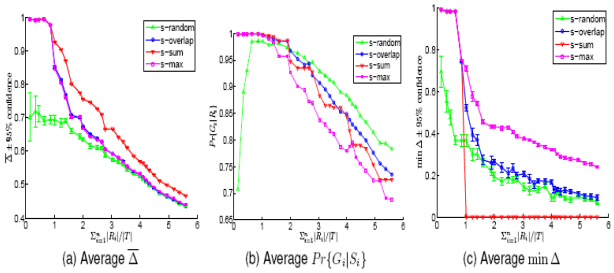
Such scenarios yield very similar agent guilt probabilities and it is important to add fake objects. We generated a random scenario that yielded _ ¼ 0:073 and min _ ¼ 0:35 and we applied the algorithms e-random and e-optimal to distribute fake objects to the agents (see

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

[14] for other randomly generated scenarios with the same parameters). We varied the number B of distributed fake objects from 2 to 20, and for each value of B, we ran both algorithms to allocate the fake objects to agents. We ran optimal once for each value of B, since it is a deterministic algorithm. Algorithm e-random is randomized and we ran it 10 times for each value of B. The results we present are the average over the 10 runs. Fig. 3a shows how fake object allocation can affect _. There are three curves in the plot. The solid curve is constant and shows the _ value for an allocation without fake objects (totally defined by agents' requests). The other two curves look at algorithms e-optimal and e-random. The y-axis shows _ and the x-axis shows the ratio of the number of distributed fake objects to the total number of objects that the agents explicitly request. We observe that distributing fake objects can significantly improve, on average, the chances of detecting a guilty agent. Even the random allocation of approximately 10 to 15 percent fake objects yields _ > 0:3. The use of e-optimal improves _ further, since the e-optimal curve is consistently over the 95 percent confidence intervals of e-random. The performance difference between the two algorithms would be greater if the agents did not request the same number of objects, since this symmetry allows nonsmart fake object allocations to be more effective than in asymmetric scenarios. However, we do not study more this issue here, since the advantages of e-optimal become obvious when we look at our second metric. Fig. 3b shows the value of min _, as a function of the fraction of fake objects. The plot shows that random allocation will yield an insignificant improvement in our chances of detecting a guilty agent in the worst-case scenario. This was expected, since e-random does not take into consideration which agents "must" receive a fake object to differentiate their requests from other agents. On the contrary, algorithm e-optimal can yield min _ > 0:3 with the allocation of approximately 10 percent fake objects. This improvement is very important taking into account that without fake objects, values min _ and _ are close to 0. This means that by allocating 10 percent of fake objects, the distributor can detect a guilty agent even in the worst-case leakage scenario, while without fake objects, he will be unsuccessful not only in the worst case but also in average case. Incidentally, the two jumps in the e-optimal curve are due to the symmetry of our scenario.

Algorithm e-optimal allocates almost one fake object per agent before allocating a second fake object to one of them. The presented experiments confirmed that fake objects can have a significant impact on our chances of detecting a guilty agent. Note also that the algorithm evaluation was on the original objective. Hence, the superior performance of optimal (which is optimal for the approximate objective) indicates that our approximation is effective. 8.3 Sample Requests 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 Pn i¼1 mi 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.



Graph No.3 of Evaluation of Sample Data Request Algorithm (2)

Consequently, the parameter that primarily defines the difficulty of a problem with sample data requests is the ratio Pn i¼1 mi jTj : We call this ratio the load. Note also that the absolute values of m1; . . .; mn and jTj play a less important role than the relative values=jTj. Say, for example, that T ¼ 99 and algorithm X yields a good allocation for the agents' requests m1 ¼ 66 and m2 ¼ m3 ¼ 33. Note that for any jTj and m1=jTj ¼ 2=3, m2=jTj ¼ m3=jTj ¼ 1=3, the problem is essentially similar and algorithm X would still yield a good allocation. In our experimental scenarios, set T has 50 objects and we vary the load. There are two ways to vary this number: 1) assume that the number of agents is fixed and vary their sample sizes mi, and 2) vary the number of agents who request data. The latter choice captures how a real problem may evolve. The distributor may act to attract more or fewer agents for his data, but

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

he does not have control upon agents' requests. Moreover, increasing the number of agents allows us also to increase arbitrarily the value of the load, while varying agents' requests poses an upper bound njTj.

Our first scenario includes two agents with requests m1 and m2 that we chose uniformly at random from the interval 6; . . . ; 15. For this scenario, we ran each of the algorithms s-random (baseline), s-overlap, s-sum, and s-max 10 different times, since they all include randomized steps. For each run of every algorithm, we calculated _ and min_ and the average over the 10 runs. The second scenario adds agent U3 with m3 _ U½6; 15_ to the two agents of the first scenario. We repeated the 10 runs for each algorithm to allocate objects to three agents of the second scenario and calculated the two metrics values for each run.

We continued adding agents and creating new scenarios to reach the number of 30 different scenarios. The last one had 31 agents. Note that we create a new scenario by adding an agent with a random request mi _ U½6; 15_ instead of assuming mi ¼ 10 for the new agent. We did that to avoid studying scenarios with equal agent sample request sizes, where certain algorithms have particular properties, e.g., s-overlap optimizes the sum-objective if requests are all the same size, but this does not hold in the general case. In Fig. 4a, we plot the values _ that we found in our scenarios. There are four curves, one for each algorithm. The x-coordinate of a curve point shows the ratio of the total number of requested objects to the number of T objects for the scenario. The y-coordinate shows the average value of _over all 10 runs. Thus, the error bar around each point shows the 95 percent confidence interval of _ values in the 10 different runs. Note that algorithms s-overlap, s-sum, and s-max yield _ values that are close to 1 if agents request in total fewer objects than jTj. This was expected since in such scenarios, all three algorithms yield disjoint set allocations, which is the optimal solution. In all scenarios, algorithm s-sum outperforms the other ones. Algorithms s-overlap and s-max yield similar _ values that are between s-sum and s-random.

Algorithm s-sum now has the worst performance among all the algorithms. It allocates all highly shared objects to agents who request a large sample, and consequently, these agents receive the same object sets. Two agents Ui and Uj who receive the same set have _ði; jÞ ¼ _ðj; iÞ ¼ 0. So, if either of Ui and Uj leaks his data, we cannot distinguish which of them is guilty. Random allocation has also poor performance, since as the number of agents increases; the probability that at least two agents receive many common objects becomes higher. Algorithm s-overlap limits the random allocation selection among the allocations who achieve the minimum absolute overlap summation. This fact improves, on average, the min_ values, since the smaller absolute overlap reduces object sharing, and consequently, the chances that any two agents receive sets with many common objects. Algorithm s-max, which greedily allocates objects to optimize max-objective, outperforms all other algorithms and is the only that yields min_ > 0:3 for high values of Pn i¼1 mi. Observe that the algorithm that targets at sum objective minimization proved to be the best for the _
Maximization and the algorithm that targets at max objective minimization was the best for min_ maximization.

VIII. CONCLUSION

From this study we conclude that the data leakage detection system model is very useful as compare to the existing watermarking model. We can provide security to our data during its distribution or transmission and even we can detect if that gets leaked. Thus, using this model security as well as tracking system is developed. Watermarking can just provide security using various algorithms through encryption, whereas this model provides security plus detection technique. This model is very helpful in various industries, where data is distribute through any public or private channel and shred with third party. Now, industry & various offices can rely on this security & detection model.

ACKNOWLEDGEMENT

For all the efforts behind the paper work, I first & foremost would like to express my sincere appreciation to the staff of Dept. of Computer Sci.& Engg., for their extended help & suggestions at every stage of this paper. It is with a great sense of gratitude that I acknowledge the support, time to time suggestions and highly indebted to my guide **Prof. S.V.Kulkarni (my project guide),** and **Dr.R.B.Naik (HOD).** Finally, I pay sincere thanks to all those who indirectly and directly helped me towards the successful completion of the paper.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 1, Issue 9, November 2012

REFERENCES

[1] Sandip A.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 Computer Engineering (IOSRJCE)ISSN : 2278-0661 Volume 1, Issue 6 (July-Aug 2012), PP 32-35 www.iosrjournals.org*

[2] 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)

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

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

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

[6] A. Shabtai, a. Gershman, M. Kopeetsky, y. Elovici Deutsche Telekom Laboratories at Ben-Gurion University, Israel. Technical Report TR-BGU-2409-2010 24 Sept. 2010 1 A Survey of Data Leakage Detection and Prevention Solutions P.P (1-5, 24-25)

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

[8]Web-based Data Leakage Prevention Sachiko Yoshihama1, Takuya Mishina1, and Tsutomu Matsumoto2 1 IBM Research - Tokyo, Yamato, Kanagawa, Japan fsachikoy

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

[10] Data Leakage Prevention: A news letter for IT Professionals Issue 5 P.P (1-3)

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

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

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

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

Biography

Sandip A. Kale- Passed Bachelor of Engg in Computer Science & Engg, Pursuing Master of Engg in software Engg at MIT College of Engg, MS, India. Has worked as Head of Computer Science & Engg Dept at MSS's College of Engg & Technology, Jalna, MS, India. Currently working as Lecturer at Govt. Polytechnic, Aurangabad, MS, India. Total teaching Experience-4.3 years.

Prof.Swati V. Kulkarni: Has completed Master of Engg in Computer Science & Engg. Currently working as Associate Professor at MIT College of Engg, Aurangabad, MS, India. Total teaching experience-8.6 years.