International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
ISSN: 0976-1353 Volume 9 Issue 1 –JULY 2014.

Mitigating Vampire Attack in Wireless Ad-Hoc Sensor Networks of Draining Life Using Resource Allocation

K.Divya Priya[#1] and V.Vijayalakshmi[*2]

[#]*PG Scholar, Bharathiyar Arts & Science College for Women*

[*] *Head of the Department (CSE), Bharathiyar Arts & Science College for Women*

Abstract- There are chances like a data distributor had given his sensitive data to a set of trusted agents. These agents can be called as third parties. There are chances that some of the data is leaked and found in an unauthorized place. This situation is called IDS. In existing case, the method called watermarking is using to identify the leakage. Or also uses the technique like injecting fake data that appears to be realistic in the data. I propose data allocation strategies that improve the probability of identifying leakages. In enhancement work I include the investigation of agent guilt models that capture leakage scenarios.

Keywords: IDS, Network-level security and protection, distributed networks, data privacy, allocation strategies.

I. 1.INTRODUCTION

A major threat to the reliability of Internet services is the growth in stealthy and coordinated attacks, such as scans, worms and distributed denial-of-service (DDoS) attacks. While intrusion detection systems (IDSs) provide the ability to detect a wide variety of attacks, traditional IDSs focus on monitoring a single subnetwork. This limits their ability to detect coordinated attacks in a scalable and accurate manner, since they lack the ability to correlate evidence from multiple subnetworks. An important challenge for intrusion detection research is how to efficiently correlate evidence from multiple subnetworks. Collaborative intrusion detection systems (CIDSs) aim to address this research challenge. A CIDS consists of a set of individual IDSs coming from different network administrative domains or organizations, which cooperate to detect coordinated attacks. Each IDS reports any alerts of suspicious behaviour that it has collected from its local monitored network, then the CIDS correlates these alerts to identify coordinated attacks that affect multiple subnetworks. A key component of a CIDS is the alert correlation algorithm, which clusters similar incidents observed by different IDSs, prioritises these incidents, and identifies false alerts generated by individual IDSs. The problem of alert correlation (also known as event correlation) is an active area of research. A key issue is how to improve the scalability of alert correlation while still maintaining the expressiveness of the patterns that can be found. Singledimensional correlation schemes have been widely studied due to their simplicity, but they lack the expressiveness to characterize many types of attack behaviors. For example, such schemes can correlate alerts pertaining to the same source addresses, but cannot discriminate between different types of behaviour. More sophisticated schemes use multi-dimensional correlation to identify patterns in events. For example, the AutoFocus system [1] uses a form of frequent itemset mining, which finds combinations of attribute values that appear in event records with a minimum frequency, known as their support threshold. A common problem with frequent itemset approaches is that many slight variations of a frequent pattern can also be frequent, e.g., if a particular source address and destination port combination are identified as a frequent pattern, then the source address on its own will also be reported as a frequent pattern. It is thus important to compress frequent patterns by filtering out redundant patterns of alerts that can be explained by more specific patterns.

In the course of doing business, sometimes sensitive data must be handed over to supposedly trusted third parties. For example, a hospital may give patient records to researchers who will devise new treatments. Similarly, a company may have partnerships with other companies that require sharing customer data. Another enterprise may outsource its data processing, so data must be given to various other companies. The owner of the data is called the distributor and the supposedly trusted third parties the agents. The goal is to detect when the distributor's sensitive data have been leaked by agents, and if possible to identify the agent that leaked the data. Here applications where the original sensitive data cannot be perturbed can be considered.

Another technique is Perturbation is where the data are modified and made "less sensitive" before being handed to agents. For example, one can add random noise to certain attributes (fake data), or one can replace exact values by ranges. However, in some cases, it is important not to alter the original distributor's data. For example, if an outsourcer is doing payroll, he must have the exact salary and customer bank account numbers. If medical researchers will be treating patients, they may need accurate data for the patients. 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. Here the Aim is to use unobtrusive techniques for detecting leakage of a set of objects or records. Specifically, I studied the following scenario: After giving a set of objects to agents, the distributor discovers some of those same objects in an unauthorized place. (For example, the data may be found on a website, or may be obtained through a legal discovery process.) At this point, the distributor can assess the likelihood that the leaked data came from one or more agents, as opposed to having been independently gathered by other means. Using an analogy with cookies stolen from a cookie jar, if i catch John with a single cookie, he can argue that a friend gave him the cookie. But if i catch John with five cookies, it will be much harder for him to argue that his hands were not in the cookie jar. That is, if the distributor sees "enough evidence" that an agent leaked data, he may stop doing business with him, or may initiate legal proceedings.

In this paper, I am developing a model for assessing the "guilt" of agents. Also present algorithms for distributing objects to agents, in a way that improves my chances of identifying a leaker. Finally, I also consider the option of adding "fake" objects to the distributed set. Such objects do not correspond to real entities but appear realistic to the agents. In a sense, the fake objects act as a type of watermark for the entire set, without modifying any individual members. If it turns out that an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty.

II. Background Work

Methodologies

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

1. Sample request
2. Explicit request

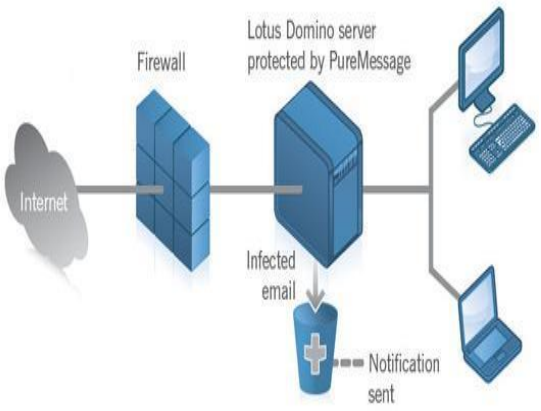

Fig:1. system model

Problem Setup and Notation:

Our model parameters interact and to check if the interactions match our intuition, in this section we study two simple scenarios as Impact of Probability p and Impact of Overlap between Ri and S. In each scenario we have a target that has obtained all the distributor's objects, i.e., T = S.

Data allocation problem

A distributor should allocate data intelligently such that agent's request should be satisfied at the same time distributor can be able to detect an agent who leaks the data.

Fake objects:

A distributor can assign fake objects with data in order to find out guilty agents. but it may not applicable in all cases because it affect the correctness of data.

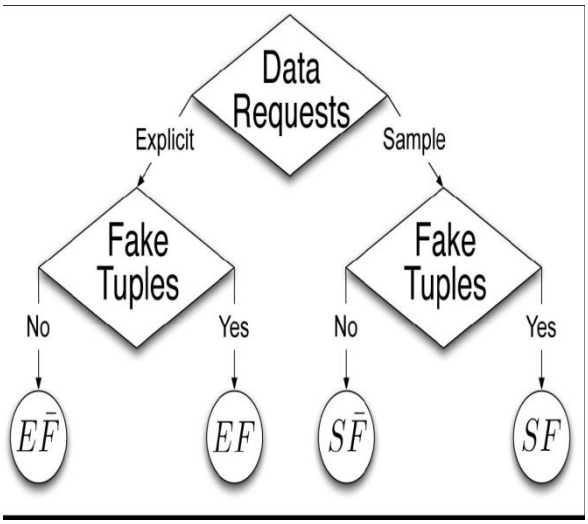

Fig :2. Leakage Problem Instances

OPTIMIZATION PROBLEM

Problem Definition:

If the distributor have data request from n agents, the distributor wants to give tables R1….Rn to agents U1….Un, respectively so that

He satisfies agent's request and

He maximizes the guilt probability differences.

ALLOCATION STRATEGIES:

There are two different allocation strategies that solve optimization problem. They are explicit allocation and sample allocation.

III. ALGORITHMS

1. Evaluation of Explicit Data Request Algorithms

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.

2. Evaluation of Sample Data Request Algorithms

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.

IV. RESULT AND ANALYSIS

The proposed two-stage alert correlation scheme equipped with the probabilistic threshold estimation achieves significant advantage in detection rate over a naive threshold selection scheme for stealthy attack scenarios. The 98% confidence interval scheme gains a high Detection Rate without significant increase in the number of messages exchanged. Our results demonstrate that by using this probabilistic confidence limit to estimate the local support threshold in our two-stage architecture, we are able to capture most of the variation between different sub networks during a stealthy scan.

Subscription Delay (SD), which represents the time required for the participants to subscribe to each suspicious IP address on the responsible destination node, i.e., SD = Message Routing Delay + Message Queueing Time. Information Correlation Time (ICT), which denotes the process time required for correlating the subscription message on the destination node. i.e., ICT = Message Queueing Time+ Data Processing Time. Load Balancing, the load on each node in terms of the number of subscription requests that are received.

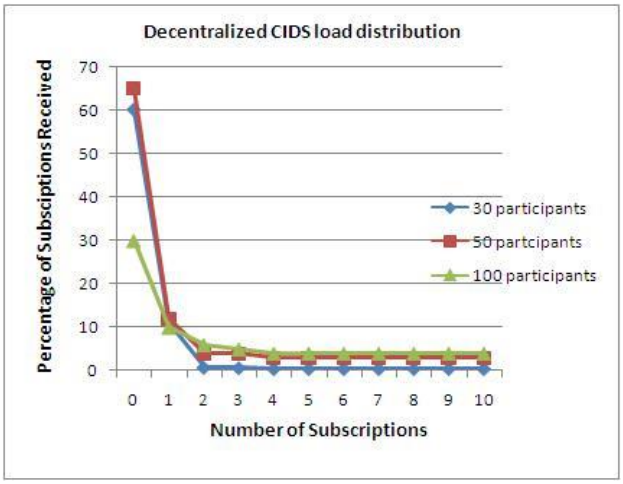

Fig:3. Cids percentage of load subscriptions

V. 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, hand over sensitive data, in a perfect world, distributor could watermark each object so that distributor could trace its origins with absolute certainty.

However, in many cases, Distributor must indeed work with agents that may not be 100 percent trusted, and 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, i 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. This model is relatively simple, but I believe that it captures the essential trade-offs. The algorithms I have presented implement a variety of data distribution strategies that can improve the distributor's chances of identifying a leaker. I 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.

VI. REFERENCES

[1] Agrawal R.and J. Kiernan,(2002), "Watermarking Relational Databases,"Proc. 28th Int'l Conf. Very Large Data Bases (VLDB '02), VLDB Endowment, pp. 155-166.

[2] Bonatti P,(2002) S.D.C. di Vimercati, and P. Samarati, "An Algebra for Composing Access Control Policies," ACM Trans. Information andSystem Security, vol. 5, no. 1, pp. 1-35.

[3] Buneman P and W.-C. Tan,(2007), "Provenance in Databases," Proc. ACM SIGMOD, pp. 1171-1173.

[4] Cui Y and J. Widom,(2003), "Lineage Tracing for General Data Warehouse Transformations," The VLDB J., vol. 12, pp. 41-58.

[5] Czerwinski Y, R. Fromm, and T. Hodes,(2007), "Digital Music Distribution and Audio Watermarking," .

[6] Guo F, J. Wang, Z. Zhang, X. Ye, and D. Li,(2006), "An Improved Algorithm to Watermark Numeric Relational Data," Information Security Applications, pp. 138-149, Springer.

[7] Hartung F and B. Girod,(1998), "Watermarking of Uncompressed and Compressed Video," Signal Processing, vol. 66, no. 3, pp. 283-301.

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

[9] Li Y, V. Swarup, and S. Jajodia,(2005),"Fingerprinting Relational Databases: Schemes and Specialties," IEEE Trans. Dependable and Secure Computing, vol. 2, no. 1, pp. 34-45.