Semantic Inference Model For Database Inference Detection Violation

Rupali Chopade Prof. B. N. Jagdale Prof. Amit Savyanavar

MAEER'S MIT College of Engineering, Department of Information Technology,

Pune University, Paud Road, Kothrud, Pune, Maharashtra, India.

Abstract — Database Security has become a major problem in modern day's applications. Generally for Information security everybody tries to protect sensitive data by using different security techniques. Despite these security measures, a malicious user can access non-sensitive information and can infer sensitive data from it.

We hereby investigate a semantic inference model based on non-deterministic approach to avoid the database inference problem. This model can work for single user as well as for multi user environment. Here Work is focused on employee information access. Probability of each employee goes on increasing on each query request. When a user poses a query, detection system will examine users past query log for last three days and calculates probability. If probability exceeds than the specified threshold, the query will be denied for that day. Also, to monitor activities, security officer can generate log.

Keywords-database; inference; probability; protection; security; query;

I. INTRODUCTION

In Today's era of e-commerce database and database security plays an important role. Database security begins with physical security for the computer systems that host the DBMS. No DBMS is safe from intrusion, corruption, or destruction by people who have physical access to the computers. Confidential information kept in databases becomes the purpose of criminals very often. From information of CSI/FBI Computer Crime and Security Survey in 2006, 32% organizations found out the unauthorized access to information, 10% organizations theft of confidential information. Total losses are related to the unauthorized access to information and theft of proprietary information. These results show all seriousness of the problem. Therefore we have developed Semantic Inference Model to avoid the data inference problem. An inference occurs when a user is able to infer some information without directly accessing it.
Following example explain how inference affects database security.

Name	Salary	City
Bob	45 K	Palo Alto
Ann	50 K	Palo Alto
Jackson	60 K	San Francisco

City	Salary
Palo Alto	45K
Palo Alto	50K
San Francisco	60K

In this database, the attribute City does not functionally determine attribute Salary, as both Ann and Bob live in Palo Alto but they earn different salaries. As a result, schema based inference detection systems do not report any inference threat in this database. However, if a user knows that Jackson is the only employee who lives in San Francisco, the user can infer the salary of Jackson by querying the database to find the salary of the employee who lives in San Francisco in the second table. This example illustrates that simply examining the database schema to detect inference is not sufficient, and taking the data in the database into consideration can lead to the detection of more inferences. We accessing them When any user fires a query, probability will be calculated and on each query request that probability goes on increasing. If probability is below threshold then user will have access to information but if probability exceeds specified threshold, then that user is not able to access information. This is the case for single user. In the same way for multi user environment, when different users tries to collaborate to increase probability of accessing information, then probability of the user will goes on increasing whose information other users are accessing. Here basically we have tried to implement inference controlling mechanism for employee module and for department manager. When employee is accessing other employee's information or department manager trying to access Research & Development data, their probability will be calculated.

II. LITERATURE REVIEW

Existing work on inference detection for database systems mainly employ functional dependencies in the database schema to detect inferences. It has been noticed that analyzing the data stored in the database may help to detect more inferences.

Raymond W. Yip and Karl N. Levitt discussed about Data Level Inference in [3] Database Systems. They have proposed five different inference rules. Each rule specifies about how data can be inferred, by firing different queries.

An inference violation detection system to protect sensitive information content has been proposed by Yu Chen and Wesley W. Chu. Initially they have developed model for single user. Based on data dependency, database schema, and semantic knowledge, a semantic inference model (SIM) is constructed. The model [2] represents the possible inference channels from any attribute to the pre assigned sensitive attributes. The SIM is then instantiated to a semantic inference graph for query-time inference violation detection. For a single user case, when a user poses a query, the detection system will examine his/her past query log and calculate the probability of inferring sensitive information. The query request will be denied if the inference probability exceeds the pre specified threshold. For multi user cases, the users may share their query answers to increase the inference probability. Therefore, later on they have developed a model for evaluating collaborative inference based on the query sequences of collaborators [1] and their task-sensitive collaboration levels.

Harry S. Delugach, Member,IEEE Computer Society, and Thomas H. Hinke, Member, IEEE Computer Society, have developed and constructed a system called [7] “Wizard” to analyze databases for their inference problems. Particularly, they have used database schema and human-supplied domain information to detect inference problems during database design time.

Farkas et al. proposed a mechanism that propagates update to the user history files to ensure that no query is rejected based on the outdated information to reduce the time in examining the entire history login computation inference.

Toland et al. proposed using a prior knowledge of data dependency to reduce the search space of a relation and thus reduce the processing time for inference. The previous work on data inference mainly focused on deterministic inference channels such as functional dependencies.

Alexander Brodsky, Csilla Farkas, and Sushil Jajodia investigate the problem of inference channels that occur when database constraints are combined with non sensitive data [5] to obtain sensitive information. They have presented an integrated security mechanism, called the Disclosure Monitor, which guarantees data confidentiality by extending the standard mandatory access control mechanism with a Disclosure Inference Engine. The Disclosure Inference Engine generates all the information that can be disclosed to a user based on the user's past and present queries and the database and metadata constraints. The Disclosure Inference Engine operates in two modes: data-dependent mode, when disclosure is established based on the actual data items, and data-independent mode, when only queries are utilized to generate the disclosed information. The disclosure inference algorithms for both modes are characterized by the properties of soundness (i.e., everything that is generated by the algorithm is disclosed) and completeness (i.e., everything that can be disclosed is produced by the algorithm). Mainly their concentration is on the development of sound and complete algorithms for both data-dependent and data-independent disclosures.

III. NFERENCE DETECTION FRAMEWORK

This implemented model consist of four different modules namely Administrator, Department Head, Manager and Employee. We have shown probabilistic approach for Department Head and Employee module. Admin is a security officer, who can generate a log to monitor various activities performed by employee. This model is helpful for any organization.

A. Working of System

A model is developed for evaluating inference based on the past query sequences. Semantic Inference Model (SIM) consists of data dependency, relational database schema, and domain-specific semantic knowledge. So, a Semantic Inference Model (SIM) representing them as probabilistic inference channels to access any data from the system. In any organization there are different types of data and some data is most important for organization. The organization works with different department and different products. In any organization every person not always knows all development work of the organization. If organization is developing some new product then everybody is not aware of the details at primary stage. It is within only that Research & Development department or with some management level person. It is generally told to employees when product is ready or at the final stage. So any organization requires more security for this type of data and don’t want to take any chance of leakage of information. For this purpose inference detection system can be used.

Also we are generating a Log, so that security officer can monitor various activities done by different employees.

B. System Architecture

As shown in figure 1, when any user fires a query, inference detection module will check probability of that user from past log as well as data probability. If probability is below 0.6 or data probability is below 0.8 then only that user will have access to data, otherwise query access will be denied.

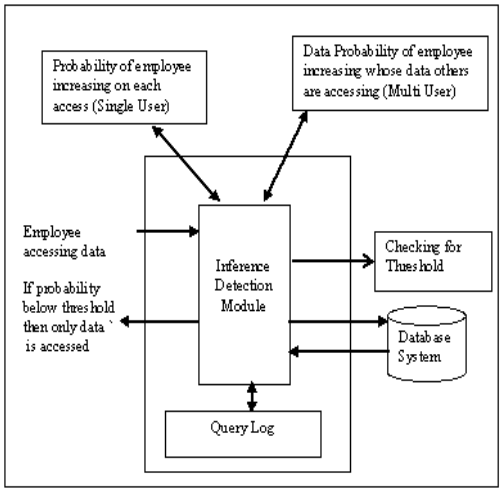

Figure 1: System Architecture

C. Setting Threshold and Probability Calculation

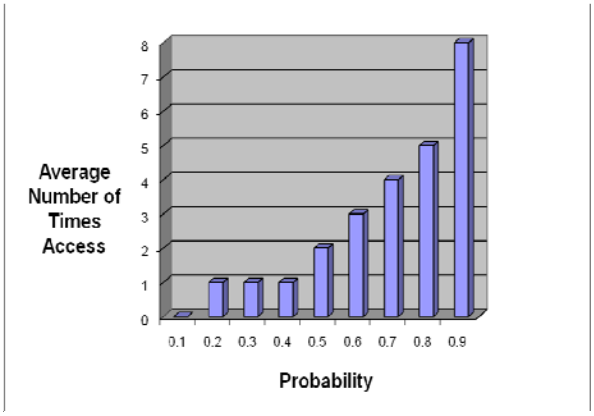

Figure 2: Threshold calculation for single user

Above graph from figure 2, shows the average number of access depending on different probability. If average access is considered threshold is calculated as 0.6.
Also we have considered few examples while deciding threshold.
Hence, Inference probability must be <0.6, otherwise access will be denied for particular user.
As we are calculating threshold for single user case, in the same way threshold should be calculated for multi user environment. Figure 3 gives idea about threshold calculation for multi user case.

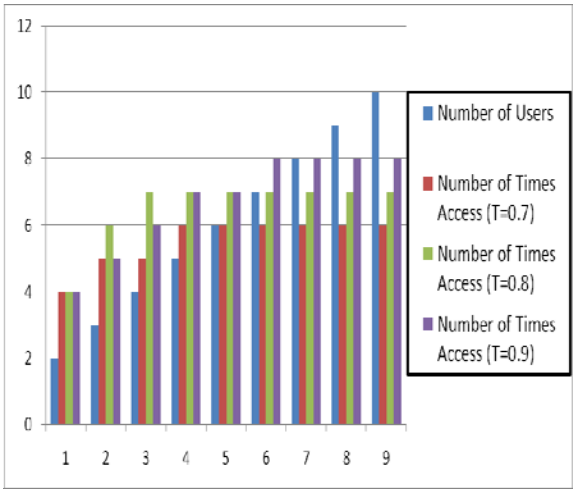

Figure 3: Threshold calculation for Multi user

As threshold is calculated as 0.6 for single user, it should be higher for multi user. Considering this fact, experimentation is done for 0.7, 0.8 and 0.9 for 2 users to 10 users. By checking consistency in number of access, threshold for multi user environment is calculated as 0.8.

Probability is calculated as conditional probability, given as Pi|j = Pr (B=bi|A=aj). It represents the occurrence of A and b and Co occurrences of A and B. Also it represents the dependency from B to A. Initially probability and data probability is set to 0.0. When user try to access data, probability is calculated as number of times user has accessed data of specific user within three days divided by total number of times user has accessed data within three days. This probability will be stored in log. Next time when same user fires query, probability will be checked from log. If it is below threshold user will have access to data, otherwise access will be denied.

Table 1: Probability calculation (single user)

Employee	**Accessing data from table**	**Probability**	**TotalCount**	**Count**
X		0.0	0	0
	A	0.1	0	0
	A	0.2000	1	1
	A	0.4000	2	2
	A	0.8000	3	3
X		0.0	0	0
	A	0.1	0	0
	A	0.2000	1	1
	B	0.3000	2	0
	B	0.6333	3	1

Probability = (count / TotalCount) + previous (Probability)
TotalCount = number of times employee accessing data
Count = Check for employee accessing which table.

Table 1, gives an idea about probability calculation. Two variables are maintained for it count and TotalCount. Initially these two variables are set to 0. First time probability is calculated as 0.1. Difference between above two examples is that, in first case employee is accessing data from same table and in second case employee is accessing data from two different tables. Depending on that count will be calculated differently.

As there is probability calculation for single user, in the same way probability is calculated for multi user. This is explained by following table.

Table 2: Probability calculation (Multi user)

Employee	Accessing another' s data	Data probability	Data count1	Data count2
	D	0.0	0	0
A	D	0.1	0	0
B	D	0.2000	0	1
C	D	0.3000	0	2
B	D	0.6333	1	3
A	D	0.8833	1	4

Data Probability = (datacount1 / datacount2) + Previous (Data Probability)
Datacount1 = Keeping record of employee accessing which record
Datacount2 = Total number of count incrementing depending on each access.
When employee A is accessing data of employee D, probability of A will be increased and data probability of employee D will be increased. Same goes on continuing, if probability or data probability which one is reaching to threshold earlier, access will be denied.
Probability calculation for multi user environment is explained by Table 2. Probability calculation is same like single user. Difference is, here data probability calculation is for employee whose data other employees are accessing.

IV. EXPERIMENT

A. Experiment

For single User Threshold is set to 0.6 and for multi user threshold is 0.8. We are keeping log with probability and data probability. When user access data of another user, then probability of user increases who is accessing data and increases data probability of user whose data is accessed.
Following is an example of a Single User Case:
Employee XYZ is accessing details of employee 24.

Query 1: select name from employee where empid=24;
[Prob=0.1] Access Given!

Query 2: select designation from employee where name='xyz';
[Prob=0.2] Access Given!

Query 3: select slipid from employeeslip where Empid=24;
[Prob=0.4] Access Given!

Query 4: select salary from employeeslip where slipid=2432;
[Prob=0.8] Access Denied!

As in the last query, probability is above threshold, access is denied.

Example of Multi User Case:
To increase individual probability users may work together and access data individually. Later on they can merge data which they have accessed individually.
Multi User: (Employee 6, 7, 8 accessing data of Employee 9).
If probability of employee 6, 7 or 9 reaches to 0.6 or above then access to them is denied and if dataprobability of employee 9 reaches to 0.8 or above then access to employee 9 is denied for all users, for that day.

Employee 6
Query 1: select name from employee where empid=9;
[Prob = 0.1] [Dataprob = 0.1]

Employee 7
Query 2: select designation from employee where empid=9;

[Prob = 0.1] [Dataprob = 0.2000]

Employee 8
Query 3: select contact from employee where empid=9;
[Prob = 0.1] [Dataprob = 0.3000]

Employee 6
Query 4: select address from employee where name='xyz';
[Prob = 0.2000] [Dataprob = 0.6333]

Employee 7
Query 5: select department from employee where name='xyz';
[Prob =0.2000] [Dataprob = 0.8833]

As we can see from above example, when employees are accessing details of another employee, probability of employees 6, 7 and 8 goes on increasing and data probability of user 9 goes on increasing. When user fires query 5, though probability is below threshold but data probability has crossed threshold, hence access to employee 9 is now denied.

B. Implementation

We run our experiment on employee access table and on research and development tables. Each table is having different number of attributes and different number of records. All attributes of different types. We are randomly firing user queries. Log generation is present on administrator side. Depending on date wise interest admin can generate log day wise. We have carried out these results under following software and hardware specifications.

Software Specification:
Operating system: Windows XP or Upper
Database Server: oracle 10g
Application server: Apache Tomcat 6.0
Browser: IE 5.0 and Upper or Mozilla, Google Chrome
IDE: Macromedia Dream weaver 8.
Programming Language: JSP

Hardware Specification:
Operating System: Microsoft windows 7 professional
Processor: Intel (R) Core (TM) i3 CPU
RAM: 3.00 GB
Speed: 2.39 GHz

C. Results

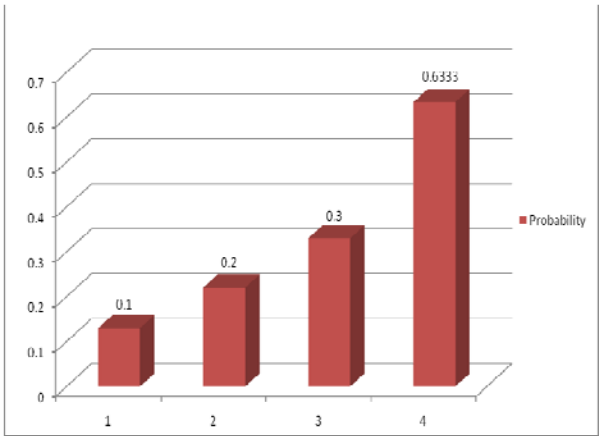

Figure 4: Probability Result (Single User)

As we can see from figure 4, whenever user try to access data content, probability and data probability is calculated. Threshold value is set as 0.6. From the example, when User 23 try to access data content from employee table, initially probability is 0, second time again when same user try to access data content probability will increase up to 0.2 and third time 0.4. But when user try to access data fourth time probability will be 0.633, which is above threshold and hence for that user Access will be denied.

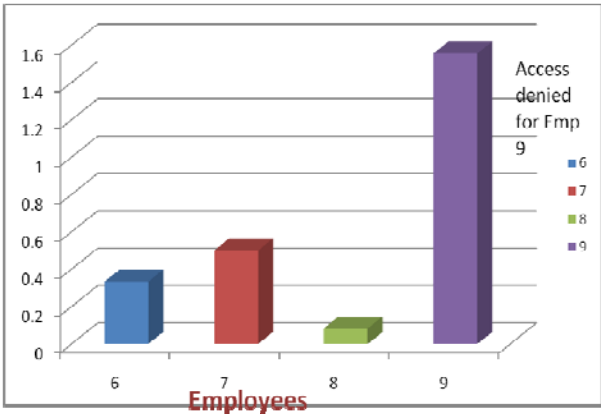

Figure 5: Probability Result (Multi User)

As we can see from figure 5 (Multi Users), whenever users (6, 7, and 8) try to access data of employee 9, probability of users 6, 7 and 8 will be calculated and data probability of employee 9 will be calculated. For multi user, Threshold value is set as 0.8. Example Users 6, 7 and 8 try to access data of employee 9 from employee table, initially probability is 0, second time again when user try to access data content probability will increase to some level and data probability of employee 9 will go on increasing. But when users try to access data continuously probability may be like 0.9333, which is above threshold and hence users Access for employee 9 will be denied.

V. CONCLUSION AND FUTURE RESEARCH

In this paper we have implemented a technique to protect sensitive information content. Malicious users can exploit the correlation among data to infer sensitive information from a series of seemingly innocuous data accesses. This developed inference detection system can be used for any organization with very small changes as per their database. Its ability to detect inference at the early stage rather than detecting after the attack is already committed. The developed Semantic Inference Model works for single user as well as for multi user environment. The developed system can be successfully deployed in any industry to deal with the threats that pose from internal users in an attempt to secure sensitive information. Further research and experiment in use of nested queries and use of multiple relations is needed.

REFERENCES

[1] Y. Chen and W. W. Chu, (2008), "Protection of Database Security via Collaborative Inference Detection", *IEEE Transactions on Knowledge And Data Engineering*, vol. 20, no. 8.
[2] Y. Chen and W.W. Chu, (2006), "Database Security Protection via Inference Detection , "*Proceeding of Third IEEE International Conference on Intelligence and Security Informatics (ISI '06).*
[3] Raymond W. Yip and Karl N. Levitt, (1998), "Data Level Inference in Database Systems", *IEEE Computer security Foundation workshops.*
[4] K. Aberer and Z. Despotovic, (2001), "Managing Trust in a Peer-2-Peer Information System," *Proceeding of 10th ACM international Conference on Information and Knowledge Management (CIKM '01).*
[5] Alexander Brodsky, Csilla Farkas, and Sushil Jajodia, (2000), "Secure Databases: Constraints, Inference Channels, and Monitoring Disclosures", *IEEE Transactions on knowledge and Data Engineering*, Vol. 12, No.6.
[6] Elisa and Ravi Sandhu, (2005), "Database Security- Concepts, Approaches, and challenges", *IEEE transaction on Dependable and secure computing*, Vol 2.
[7] H.S. Delugach and T.H. Hinke, (1996), "Wizard: A Database Inference Analysis and Detection System," IEEE Trans. Knowledge and Data Eng., vol. 8, no. 1, pp. 56-66.
[8] C. Farkas and S. Jajodia,(2002), " The Inference Problem: A Survey," SIGKDD Explorations, vol. 4, no. 2, pp. 6-11.
[9] [boo97] Booch Grady, James Rambaugh, Ivar Jacobon, (1997), "The unified modeling language", Pearson Education Asia.
[10] [PRE01] Pressman Roger, (2001), "Software Engineering", Mc-Graw Hill.
[11] Oracle Developer 2000 by Ivan Bayross, BPB publication 2001.
[12] Java Servlet Programming (Paperback) by Jason Hunter (Author).
[13] Professional JSP by Wrox publication.
[14] C.J. Date (1995),"An Introduction to Database Systems", Sixth Ed. Addison-wesley
[15] Elmasri, Navathe, "Fundamentals of Databases", 3rd Edition.