International Journal of Computer Science and Network (IJCSN)
Volume 1, Issue 6, December 2012 www.ijcsn.org ISSN 2277-5420

Policies Based Intrusion Response System for DBMS

[1]Fatima Nayeem, [2]M.Vijayakamal

[1] Dept of CSE, JNTU H, Sridevi Womens Engimeering College
Hyderabad, Andhra Pradesh, India.

[2] Dept of CSE, JNTU H, Sridevi Womens Engimeering College
Hyderabad, Andhra Pradesh, India

Abstract

Intrusion detection systems play an important role in detecting online intrusions and provide necessary alerts. Intrusion detection can also be done for relational databases. Intrusion response system for a relational database is essential to protect it from external and internal attacks. We propose a new intrusion response system for relational databases based on the database response policies. We have developed an interactive language that helps database administrators to determine the responses to be provided by the response system based on the malicious requests encountered by relational database. We also maintain a policy database that maintains policies with respect to response system. For searching the suitable policies algorithms are designed and implemented. Matching the right policies and policy administration are the two problems that are addressed in this paper to ensure faster action and prevent any malicious changes to be made to policy objects. Cryptography is also used in the process of protecting the relational database from attacks. The experimental results reveal that the proposed response system is effective and useful.

Keywords– *Intrusion detection, intrusion response system, policies, relational database*

1. Introduction

Relational databases are built on Relational Model proposed by Dr. E. F. Codd. The relational model has become a consistent and widely used DBMS in the world. The databases in this model are efficient in storing and retrieval of data besides providing authentication through credentials. However, there might be many other attacks apart from stealing credentials and intruding database. Adversaries may always try to intrude into the relational database for monetary or other gains [1]. The relational databases are subjected to malicious attacks as they hold the valuable business data which is sensitive in nature. Monitoring such database continuously is a task which is inevitable keeping the importance of database in mind. This is a strategy that is in top five database strategies as identified by Gartner research which are meant for getting rid of data leaks in organizations [2]. There are regulations from governments like US with respect to managing data securely. The data management like HIAPP, GLBA, and PCI etc. is mentioned in the regulations as examples. The attacks made by adversaries are changing and they have more sophisticated. It does mean that there is the competition between the data protectors and security breakers like hackers. Hackers are using more and more sophisticated techniques to break security systems of IT. They also attack relational databases. For this reason organizations have to focus on the database security with much higher sophistication. Relational database systems are also coming with built in security mechanisms using credentials, access control list and so on. However, these are not sufficient when attacks are made by internal adversaries [3]. For this reason organizations have to reevaluate the security mechanisms to protect data. It does mean that organizations have to take some additional security measure instead of relying on database built in security mechanisms. Some of the database related attacks are performed by hackers are named as data infiltration, and SQL injection which are malicious to database but not for the underlying network and operating systems.

The ID mechanism approach in this paper has two important aspects. They are actually altered for database management systems. They are known as Anomaly response system and anomaly detection. The former is achieved using database access profiles or users and roles. Different levels of data can be recorded using profiles as explored in [4]. This paper focuses on the second aspect that is taking actions once detection of anomaly is completed. The proposed approach follows a proactive approach in showing alerts and blocking the anomalous request. The response actions are fine – grained and they are not aggressive or conservative. Such actions may suspend malicious request [5]. When a request is suspected, it is kept on hold until further authentication steps are carried out to verify the validity of the request. It is also possible to mark a request tainted indicating that the request is potentially suspicious. As there is need for different responses based on the malicious requests, the key is to address the response measure problem.

When a response system is sought which takes actions automatically when malicious requests are encountered, it is not an easy task. The key idea to solve this is to monitor the context in which such request is made. To address the

problem a suitable response policy is required that can cater to the needs of all situations. Policy administration and policy matching are the two issues addressed in terms of response policies. Response policy can be built as regular database object which takes care of particular response. However, it represents various challengers instead of simply storing data as other database objects. The user which DBA role only can create policy objects in the database. In this administration model the basic problem is identified and named as conflict of interest. Insider threat is the main issue in this case which throws challenges and demands such accurate response system which is made up of response policies. In order to overcome this problem, an administration model is proposed. Separation of duties is given to various users. Among all users corresponding privileges are maintained so as to make it more secure. Every policy operation is authorized by DBA. The main contributions of this paper include provision for providing intrusion response policies; an administration model for monitoring those policies; algorithms for interacting with policy database effectively; implementing the schemes using DBMS.

2. Related Work

Databases have been subjected to malicious attacks in the past. Intrusion response systems were developed to prevent that problem. Intrusion detection systems detect intrusions and provide responses. They depend on the notion of response policies which were first explored in [5] which also provides details about arbitrary predicates and other issues. It only used predicates with good quality. Policy administration plays important roles in this paper. Fine grained response actions are explored in [6] which provides design and implementation of ACL for the same. The ACL is for authorization mechanism. There are some internal threats from DBAs also. It is the worst case that has to be handled. In order to overcome this principle of least privilege is followed. This means that DBAs are given privileges only to the required extent that is least required privileges. This is achieved by creating roles and response policy objects. It is actually done through protected schema for administration of databases with respect to vault policies [7]. Vault in databases is a mechanism which is practically implemented by Oracle database which helps in reducing the risk of insider attacks. DVSYS protected schema is used to store vault database objects. The schema protects itself against improper utilization of privileges given to users including administrators. The privileges include DROP ANY, SELECT ANY TABLE and so on. More details are such privileges are found in [7].

Anomaly response system and Vault of Oracle database are presented in this paper in policy driven fashion. Thus the proposed system follows something which is similar to Oracle database vault. The response system thus resembles the vault system of Oracle. The advantages of this approach are described here. They are fundamental changes are required to access control mechanisms of DBMS; the principle of least privilege may not be suitable for many organizations. Some work is found on the threshold signature schemes in [8]. This paper provides technique of threshold signatures for managing DBMS objects. The policy matching problem addressed in this paper is similar to [9]. There are many algorithms for this purpose such as [9], [10], [11], [12], and [13]. The algorithm for event matching concept is as described in [20]. This is meant for preprocessing the concept of subscription trees as discussed in [10]. The leaves in the tree represent actual subscriptions. The algorithm walks thru the tree in order to find out matching subscriptions. However, the order of processing is not known. Arbitrary predicates are needed for policy matching problem. Many such algorithms are described in [11]. Cache hit ratio improvement is their main focus. However, our focus is not that as we store policies and their content is cached in DBMS.

The base policy of ours with respect to matching algorithm is similar to that of [12]. In this paper that is extended where elimination of predicates that are no longer required to be evaluated. In [13] an internal binary tree is used by the algorithm proposed for matching predicates. It is achieved based on equality and inequality predicates while our problem needs to support arbitrary predicates. In [14] event matching using BDD (Binary Decision Diagrams) is proposed which also considers arbitrary predicates with disjunctions support in the subscription language. However, in our work we need not to support disjunctions and for this reason BDD based scheme is not used. The problem of continuous query processing is also related to event matching which is explored in [15]. In this case the problem is effectively addressed using matching multiple streaming tuples that are related to various relations and stored queries or views. This is also somewhat different from policy matching problem that we addressed in this paper.

3. Proposed Intrusion Response System

The proposed intrusion response system for relational databases is described in the subsequent sub sections. It focuses on the policy language, policy administration, and policy matching and attack prevention. The intrusion response system proposed here is influenced by [16]. In fact the policy language, policy matching and policy administration used in this paper resemble [16].

3.1 Policy Language

Considering the detection of an anomaly as a system event, this policy language is proposed. The attributes considered for anomaly detection are SQL command, role and user. These anomaly attributes and also the environment or context in which anomalous request is made are considered and an Even driven language is developed. The language is influenced by [17]. For instance a rule in the event language is as follows.

ON {Event} IF {Condition} THEN {Action}
The structural attributes considered are database, schema, object type, SQL command and object attributes while the attributes used in the CONTEXTUAL aspect are user, role, client application, source IP, date and time etc. Some of the predicates used in the language include
Role != DBA
Objs Not In {dbo.*}
Source IP IN 192.168.0.1/12

Response actions are categorized into very low severity actions, low severity actions, very aggressive actions and aggressive actions. All these actions either suspend or taint the malicious request. The former keeps the request on hold until all security details are verified while the latter is simply marked as potential suspicious request. The low severity actions are NOP, LOG, and ALERT. The medium severity actions are TAINT, and SUSPEND while the high severity actions are ABORT, DISCONNECT, REVOKE, and DENY. The response policy framework has the keywords like ON, IF, THEN, CONFIRM, ON SUCCESS, ON FAILURE etc.

3.2 Policy Administration

Policies are stored in database and they are administered. The administration is done by DBAs. It is fine as far as external attacks are concerned. However, there is a concern with internal attacks that are done by DBAs who are supposed to protect databases from malicious attacks. This paper does not assume the DBMS to have secret key for verifying integrity of policies. The basic idea of our approach is that it does not trust a single DBA who has access to secret key. Instead, the secret key is distributed among DBAs. A threshold cryptographic scheme is used to overcome the security problems in sharing of secret key.

3.3 Policy Matching

The algorithms used for policy matching are taken from [16]. The policy matching is of two types. They are known as base policy matching and ordered policy matching. The base policy matching algorithm is called when an anomaly detection event is fired by response engine. The predicates defined on every attribute are evaluated. In the process the algorithm visits all policy nodes to compare with the required policy. A policy is matched when number of predicates in the policy condition and the predicate match count are equal in number. The ordered policy matching on the other hand does not go through predicates in any fixed order. It uses a heuristic to decide the next predicated to be visited. It uses descending order of policy count. It does this to know the correct predicates and to process them in correct order. In this algorithm, sorting predicates in the descending order is the preprocessing required.

4. Experimental Evaluation

The algorithms [16] are implemented and tested the proposed scheme which is meant for giving best response when an anomalous request is encountered. The aim of the experiments is to verify the overhead incurred by both algorithms namely basic policy matching algorithm and ordered policy matching algorithm. Three sets of experiments are conducted. The first two sets compare the overhead of the policy matching algorithm while the third set of experiments is meant for reporting signature verification overhead. The results are shown below.

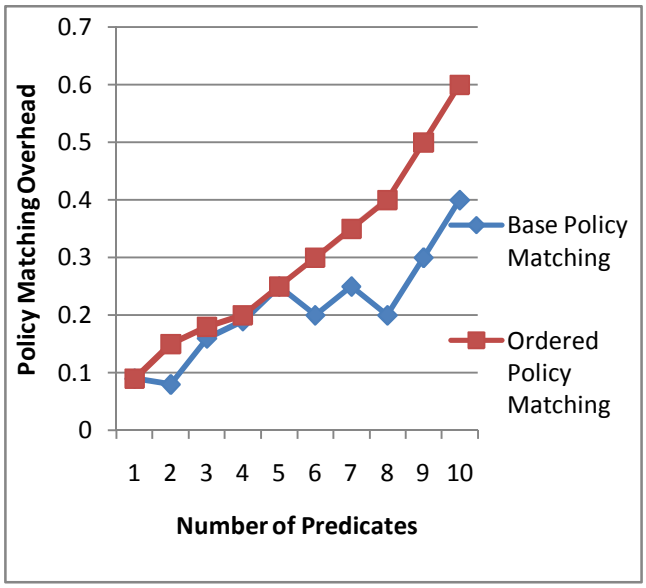

Fig. 1 – Experiment 1, Policy matching overhead vs. number of predicates

As can be seen in fig. 1, number of predicates is taken in horizontal axis while vertical axis represents policy matching overhead. The results reveal that the base policy matching shows better performance when compared with ordered policy matching in terms of overhead.

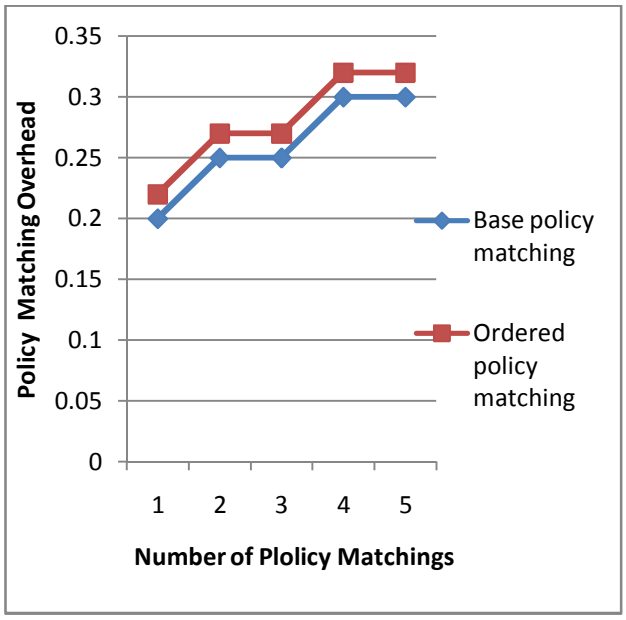

Fig. 2 - Experiment 2, Policy matching overhead vs. number of policy matchings

As can be seen in fig. 2, number of policy matchings is taken in horizontal axis while vertical axis represents policy matching overhead. The results reveal that the base policy matching shows better performance when compared with ordered policy matching in terms of overhead.

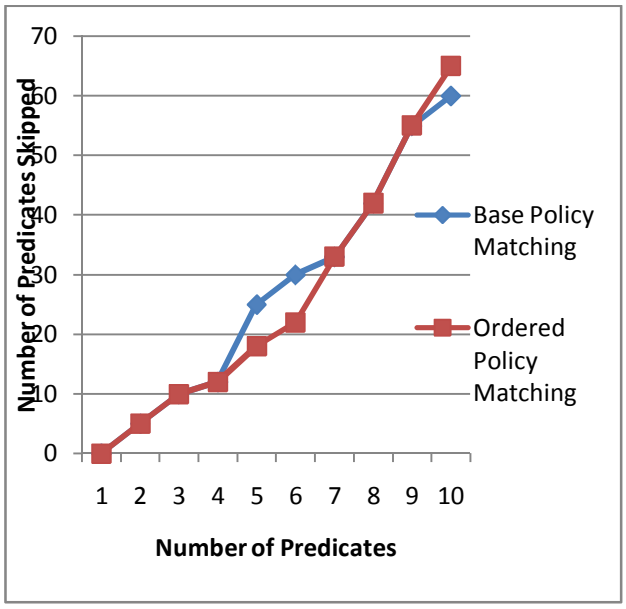

Fig. 3 – Experiment 1, Number of Predicates Skipped vs. Number of Predicates

As can be seen in fig. 3, number of predicates is taken in horizontal axis while vertical axis represents number of predicates skipped. The results reveal that the base policy matching shows better performance when compared with ordered number of predicates skipped.

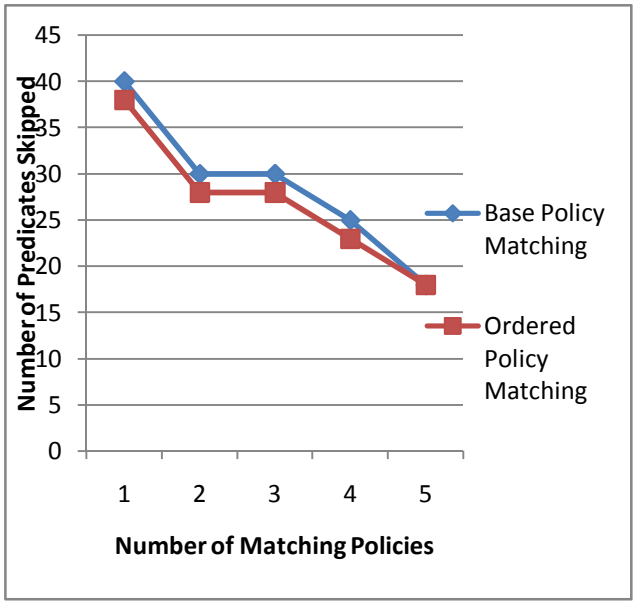

Fig. 4 – Experiment 2, Number of Predicates Skipped vs. Number of Matching Policies

As can be seen in fig. 4, number of matching policies is taken in horizontal axis while vertical axis represents number of predicates skipped. The results reveal that the base policy matching shows better performance when compared with ordered number of predicates skipped.

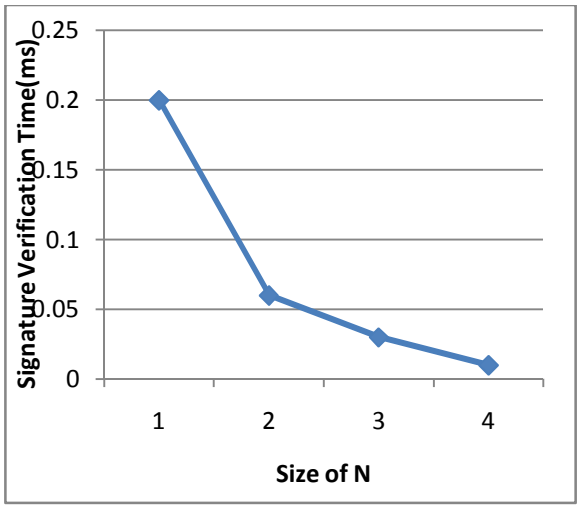

Fig. 5 – Size in bits vs. Signature Verification Overhead for given policy

As can be seen in fig. 5, Size in bits is represented by horizontal axis while the vertical axis represents signature verification time in milliseconds. The results revealed that the signature verification time is more when time is less. As size is increased, the signature verification time is decreased.

5. Conclusion and Future Work

In this paper, we have proposed a response system for DBMS which works when an intrusion of malicious request is encountered. The system is responsible to provide a suitable response when a malicious request is received. The proposed system is based on the notion of response policies. We have developed many policies that cater to various contexts in which anomalous request are made to DBMS. Policy matching and policy administration are the two issues addressed in this work besides providing required algorithms for the same. The proposed system also takes care of internal attacks from DBAs who have privileges to do important activities. Even for administrators also role based access restrictions are provided. The experimental results revealed that the proposed response system is effective and can provide accurate responses based on the response policies maintained in the policy database.

References

[1] R.B. Natan, Implementing Database Security and Auditing. Digital Press, 2005.

[2] M. Nicolett and J. Wheatman, "Dam Technology Provides Monitoring and Analytics with Less Overhead. Gartner Research (Nov. 2007)," http://www.gartner.com, 2010.

[3] D. Brackney, T. Goan, A. Ott, and L. Martin, "The Cyber Enemy within ... Countering the Threat from Malicious Insiders," Proc. Ann. Computer Security Applications Conf. (ACSAC). pp. 346-347, 2004.

[4] A. Kamra, E. Terzi, and E. Bertino, "Detecting Anomalous Access Patterns in Relational Databases," J. Very Large DataBases (VLDB), vol. 17, no. 5, pp. 1063-1077, 2008.

[5] A. Kamra, E. Bertino, and R.V. Nehme, "Responding to Anomalous Database Requests," Secure Data Management, pp. 50- 66, Springer, 2008.

[6] A. Kamra and E. Bertino, "Design and Implementation of SAACS: A State-Aware Access Control System," Proc. Ann. Computer Security Applications Conf. (ACSAC), 2009.

[7] "Oracle Database Vault Administrator's Guide 11g Release 1 (11.1)," http://download.oracle.com/docs/cd/B28359_01/ server.111/b31222/toc.htm, Jan. 2009.

[8] V. Shoup, "Practical Threshold Signatures," Proc. Int'l Conf. Theory and Application of Cryptographic Techniques (EUROCRYPT), pp. 207- 220, 2000.

[9] F. Fabret, F. Llirbat, J.A. Pereira, I. Rocquencourt, and D. Shasha, "Efficient Matching for Content-Based Publish/Subscribe Systems," technical report, INRIA, 2000.

[10] M.K. Aguilera, R.E. Strom, D.C. Sturman, M. Astley, and T.D. Chandra, "Matching Events in a Content-Based Subscription System," Proc. Symp. Principles of Distributed Computing (PODC), pp. 53-61, 1999.

[11] A.V.Saurkar, Prof. A.R. Itkikar ," Use of Inheritance Feature in Relational Database Development", ijcsn vol 1, issue 3, 2012

[12] T.W. Yan and H. Garcı´a-Molina, "Index Structures for Selective Dissemination of Information under the Boolean Model," ACM Trans. Database Systems, vol. 19, no. 2, pp. 332-364, 1994.

[13] E.N. Hanson, M. Chaabouni, C.-H. Kim, and Y.-W. Wang, "A Predicate Matching Algorithm for Database Rule Systems," Proc. ACM SIGMOD, vol. 19, no. 2, pp. 271-280, 1990.

[14] A. Campailla, S. Chaki, E. Clarke, S. Jha, and H. Veith, "Efficient Filtering in Publish-Subscribe Systems Using Binary Decision Diagrams," Proc. Int'l Conf. Software Eng. (ICSE), pp. 443-452, 2001.

[15] H.-S. Lim, J.-G. Lee, M.-J. Lee, K.-Y. Whang, and I.-Y. Song, "Continuous Query Processing in Data Streams Using Duality of Data and Queries," Proc. ACM SIGMOD, pp. 313-324, 2006.

[16] Ashish Kamra and Elisa Bertino, Fellow, IEEE, "Design and Implementation of an Intrusion
Response System for Relational Databases". IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 23, NO. 6, JUNE 2011.

[17] J. Widom and S. Ceri, Active Database Systems: Triggers and Rules for Advanced Database Processing. Morgan Kaufmann, 1995.

[18] J.A. Pereira, F. Fabret, F. Llirbat, and D. Shasha, "Efficient Matching for Web-Based Publish/Subscribe Systems," Proc. Int'l Conf. Cooperative Information Systems (CooplS), pp. 162-173, 2000.