International Journal of Engineering Research-Online
A Peer Reviewed International Journal
Articles available online http://www.ijoer.in

Vol.3., Issue.2, 2015

RESEARCH ARTICLE

ISSN: 2321-7758

AN EFFICIENT TECHNIQUE BY USING SESSION KEY TO PROTECT RELATIONAL DATABASE

GOPAL S. ADE[1], Prof.S.T.KHANDARE[2]
[1]ME 1stYEAR, Associate Professor
[2]CSE Department, CSE Department,
BNCOE, Pusad.

Article Received: 12/04/2015 | Article Revised on:17/04/2015 | Article Accepted on:20/04/2015

ABSTRACT
Relational database is one which we are using in our technical world daily. Relational databases have lots of administrators to control each and every table. Administrator is authorized to control their own corresponding tables only(i.e. the tables which is created by them). If the admin password is hacked, then data changes and updating can be done by the hacker himself. There is no protection factor. This paper proposes an interactive response policy which helps in protection of relational data base by sending the session key to the data base administers. The session key is sent as SMS to the corresponding administrators mobile.

Keywords— Relational Database, session key, database administrator , intrusion.

1. INTRODUCTION

Data prevention inside and outside the organisation is a major challenge. Previous data base security mechanisms are not of that much help in the case of theft from insiders. A session key which we want to use is a single-use symmetric key used for encrypting all messages in one communication sessions. There are two primary reasons to use session key, First, several cryptanalytic attacks become easier as more material encrypted with a specific key is available. By limiting the amount of data processed using a particular key, those attacks are made more difficult. Second, asymmetric encryption is too slow for many purposes, and all secret key algorithms require that the key is securely distributed. By using so called an algorithm which is asymmetric to encrypt the secret key for another, comprehensive, faster, symmetric algorithm, it's possible to improve overall performance considerably.

In this paper we proposes a model which give protection to the relational database using session key. In this paper we just created a database access profile of roles and uses. A user request that is not conformed to the profile is considered as strange. Once an anomaly is detected, we have to perform the following task such as 1) Sending an alert which allows anomalous request to go through.

2) An action which blocks the anomalous request.

3) Which may suspend or taint an anomalous request and a response policy is required by the data base security administrator to specify appropriate response action.

We applies the technique of session key to achieve separation of duty since we assume the Data Base Administrators to possess all possible privileges in the DBMS. When a database administrator want to alter any data in data base, a session key is generated and every part of the key is sent as SMS to the relevant Administrators mobile.

2. **INTRACTION WITHUSER: A CONDITION ACTION LANGUAGE**

An event constitute user role, SQL commands. A policy can be specified taking into account the anomaly attributes to guide the response engine in taking a suitable action. This paper put forward a condition action language for specifying response policies according to the events. The protocol of the language is organized as follows.

ON{Event}
IF{Condition}
THEN{Action}

This works as, if the event of ON arises and the condition of IF evaluates to true, the specified action in THEN is executed. An event is the anomaly; Condition is specified on the attribute of detected anomaly. An action is the revisiting action executed by the DBMS engine.

2.1**Attributes**

Anomalies can be accessed by using the attributes of anomaly. We can categorize attributes into two groups.

1) Exact contextual category, including all further attributes describing the context of the anomalous request such as user, role, source, and time.

2) Structural category, includes all attributes conveying information about the structure of anomalous request such as SQL command, and access database objects.

2.2 Conditions

A response policy condition has partial or full relation with of predicates where each predicate is specified against a single normal attribute.

2.3 Action in Response

When an anomalous request is detected, an action is executed by the response system to address the anomaly. The response action is to be executed is specified as part of response policy. There may occurs low severity action. Such action may log the anomaly details or send an alert and they do not prevent the intrusion. Second action consists of actions such as dropping the user, is allow access to the user or prohibiting the necessary privileges. Third action is neither too conservative as like first action nor too aggressive as like second action. Such action may suspend or taint an anomalous request. A suspended request is put on hold, until some specific actions are executed by the user, such as the execution of further authentication steps.

2.4 Response Policy

The condition action policy is sufficient to manage simple response measures such as disconnecting users, dropping an anomalous request, sending an alert. In some cases, we need to interact with users. If the authentication fails, the user is disconnected. Otherwise the request proceeds. As desired necessary condition action are failed to proceed with the interaction action, we use confirmation action. The purpose of the confirmation action is to interact with the user. If the resulting action is successful, the resolution action will be executed completely, otherwise the negative stage action is executed completely. The response policy can be symbolically represented as follows.

ON {event}
If {condition}
THEN {Initial Action}
CONFIRM {Confirmation Action}
ON SUCCESS {Resolution Action}
ON FAILURE {Failure action}

3. **POLICY ADMINISTRATION**

The threat scenario that we assume is that a Data Base Administrators has all the privileges in the DBMS. We protect a response policy against malicious modifications by maintaining a digital signature on policy definition. A valid digital signature gives a recipient reason to believe that the message was created by a known sender, such that the legitimate sender cannot deny having sent the message authentication and that the message was not altered in transit (integrity). The signature is then validated either periodically or upon policy usage to verify the integrity of the policy definition. We do not assume the DBMS to be in possession of a secret key for verifying the integrity of policies. If the DBMS had owned such key, it could simply create a HMAC (Hashed Message Authentication Code) of each policy using its secret key, and later use the same secret key to verify the legal integrity of the policy. However, management of such secret key is an issue since we cannot assume the key to be hidden from a malicious DB Administrators. The fundamental premise of our approach is that we do not trust a single DBA (with the secret key) to create or manage the response policies, but the threat is mitigated if the trust (the secret key) is distributed among multiple DBAs. This is also the fundamental problem in threshold cryptography, that is, the problem of secure sharing of a secret. Threshold cryptography in order to decrypt an encrypted message, several parties (more than some threshold number) must cooperate in the decryption protocol. The message is encrypted using a public key and the corresponding private key is shared among the participating parties. In the threshold setting, we would like to efficiently designed implement, via efficient protocols, the most secure cryptosystems and signature schemes. We would also like to make our own protocols secure in the highly tough possible model of faults. The following are some of the various considerations we make when modelling computer faults:

- The size of the threshold: What fraction of the servers can be corrupted by the adversary without any hazards to the service that these servers implement?
- Efficiency considerations: How much communication, storage, and computation do these fault-tolerant protocols require?
- Model of communication: How pure are the requirements we place on it? Do we want synchronous or partially synchronous communication, legally authenticated broadcast and secure links between servers?
- Type of adversary we are going to tolerate: How does the adversary choose which players to corrupt? Can a server securely erase its local data so that it cannot be retrieved by the adversary once the server is infiltrated?

We thus proposes base of our project on a threshold cryptographic signature scheme. We send the session key to the data base administrator through SMS.

4. OVERALL PROCESS

The overall process includes the signature share generation, the signature share combining, and the final signature verification operations. The steps in the life cycle of a policy object are policy creation, activation, suspension, alteration, and deletion. When the policy has been authorized by k _ 1 administrators, the policy state is changed to ACTIVATED. A policy in an ACTIVATED state is operational, that is, it is considered by the policy matching procedure in its search for matching policies. If a policy needs to be altered, dropped or made nonoperational, it must be moved to the SUSPENDED state. The transition from the ACTIVATED state to the SUSPENDED state must also be authorized by k _ 1 administrators, before which the policy is in the SUSPEND IN-PROGRESS state. Note that a policy in the SUSPEND INPROGRESS state is also considered to be operational. From the SUSPENDED state, a policy can be either moved back to the CREATED state or it can be moved to the DROPPED state. A single database administrator can move a policy to the CREATED state from the SUSPENDED state, while a policy drop operation must be authorized by k _ 1 administrators (before which the policy is in the DROP IN-PROGRESS state). We begin our detailed discussion of a policy object's lifecycle with the policy creation procedure.

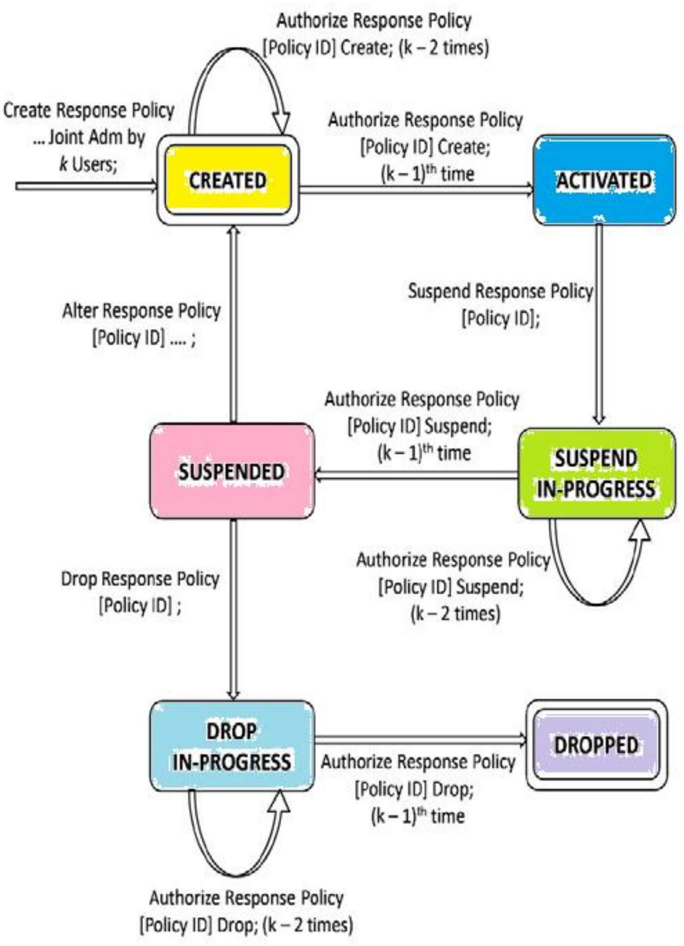

Fig. State diagram for EVENTS

5. **CONCLUSION**

In this paper we proposes an interactive user driven response policy that requires a second layer action of authentication which will provide a second layer of defence when certain anomalous actions are executed against critical system resources such as anomalous deauthenticated access to system critical catalog tables. In this paper, we have described the second side response component of our intrusion detection system for a DBMS, the appropriate secondary response Action and a response policy language which helps the administrator to specify appropriate action for different action.

6. **REFERENCES**

[1] R. Mogull, “Top Five Steps to Prevent Data Loss and Information
Leaks”, *Gartner Research* (July 2006),2010.
[2] A. Kamra, E. Bertino, and R.V. Nehme, “Responding to Anomalous
Database Requests,” *Secure Data Management*, pp. 50- 66, Springer,
2008.
[3] M. Nicolett and J. Wheatman, “Dam Technology Provides Monitoring
andAnalytics with Less Overhead”. *Gartner Research*(Nov. 2007),
http://www.gartner.com, 2010.
[4] A. Kamra and E. Bertino, “Design and Implementation of SAACS: A

State-Aware Access Control System,*"* *Proc. Ann. Computer Security
Applications Conf.* (ACSAC), 2009.

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

[6] R. Gennaro, T. Rabin, S. Jarecki, and H. Krawczyk, "Robust and
Efficient Sharing of RSA Functions," *J. Cryptology*, vol. 20, no. 3, pp.
393-400, 2007.

[7] A. Kamra and E. Bertino, —Design and Implementation of Intrusion Response system,

[8] 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.

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

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

[11] R. Mogull, —Top Five Steps to Prevent Data Loss Leaks. Gartner Research (July 2006), http://www.gartner.com, 2010.