Protection of Relational Database using Session Key

Julin Geevarghese, Celia Beryl Philips, M. N. Sushmitha, M. Roberts Masillamani

Abstract—Relational databases have lots of administrators to control every table. Administrator is authorized to control their own corresponding tables only. If the admin password is hacked, then data changes and updations can be proceeded by the hacker himself. There is no security factor. We propose 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 admin's mobile.

Keywords—database administrator, intrusion, session key.

I. INTRODUCTION

Prevention of data inside and outside the organisation has become a major challenge. Standard data base security mechanisms are not of much help in the case of theft from insiders[1],[2]. We have developed a model which give protection to the relational database using session key. We have created a database access profile of roles and uses. A user request that is not conformed to the profile is considered as anomalous. 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.[3][4]. 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 DBA's to possess all possible privileges in the DBMS. In our approach, a policy operation is invalid unless it has been authorized by at least k DBA's. When an administrator want to change any data in database, a session key is generated and every part of the key is sent as SMS to the corresponding Admin’s mobile.

II. 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.

We propose a condition action language for specifying response policies according to the events. The rule of the language is organized as follows.

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

Its semantics is as follows: if the event arises and the condition evaluates to true, the specified action is executed. An event is the anomaly, Condition is specified on the attribute of detected anomaly. An action is the response action executed by the engine.

2.1 Attributes And Conditions

2.1.1 Attributes

Anomalies can be accessed by using the attributes of anomaly. We can categorize attributes into two groups. The first category referred to as contextual category, includes all attributes describing the context of the anomalous request such as user, role, source, and time. The second category, referred to as structural category, includes all attributes conveying information about the structure of anomalous request such as SQL command, and access database objects.

2.1.2 Conditions

A response policy condition is a conjunction of predicates where each predicate is specified against a single normal attribute.

2.2 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, disconnecting the user or denying 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.
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.

2.3 Response Policy

The condition action policy is sufficient to manage simple response measures such as disconnecting users, dropping an anomalous request, or sending an alert. In some cases, we need to interact with users. If we want to execute a fine-grained response action, then we need to communicate with the user for the authentication. If the authentication fails, the user is disconnected. Otherwise, the request proceeds. As condition action is unable to proceed with the interaction action, we use confirmation action. The purpose of the confirmation action is to interact with the user. If the confirmation action is successful, the resolution action is executed, otherwise the failure action is executed.

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}

III. POLICY ADMINISTRATION

The threat scenario that we assume is that a DBA has all the privileges in the DBMS. We protect a response policy against malicious modifications by maintaining a digital signature on policy definition. 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 possessed such key, it could simply create a HMAC (Hashed Message Authentication Code) of each policy using its secret key, and later use the same key to verify the 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 DBA. 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. We thus base our project on a threshold cryptographic signature scheme. We send the session key to the database administrator through SMS.

Threshold Signature: A’ K’ out of ‘L’ threshold signature scheme is a protocol that allows any subset of ‘k’ users out of ‘L’ users to generate a valid signature, but that disallows the creation of a valid signature, but that disallows the creation of a valid signature if fewer than ‘K’ users participated in the protocol.[5][6].

IV. UNITS

Use either SI (MKS) or CGS as primary units. (SI units are strongly encouraged.) English units may be used as secondary units (in parentheses). This applies to papers in data storage. For example, write “15 Gb/cm² (100 Gb/in²).” An exception is when English units are used as identifiers in trade, such as “3½ in disk drive.” Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity in an equation.

The SI unit for magnetic field strength $H$ is A/m. However, if you wish to use units of T, either refer to magnetic flux density $B$ or magnetic field strength symbolized as $\mu_0 H$. Use the center dot to separate compound units, e.g., “A·m².”

![Policy State Transition Diagram](image)

Fig. 1 Policy State Transition Diagram

V. 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 IN-PROGRESS 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 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.

VI. CONCLUSION

We developed an interactive response policy that requires a second factor of authentication which will provide a second layer of defense when certain anomalous actions are executed against critical system resources such as anomalous access to system catalog tables. In this paper, we have described the response component of our intrusion detection system for a DBMS. We propose the appropriate response Action. And we propose a response policy language which helps the administrator to specify appropriate action for different action.

REFERENCES