Multi-Agent System for Detecting and Blocking SQL Injection

Niraj Kulkarni
Department of Information Technology, Sinhgad Academy of Engineering, Pune, India.

D R Anekar
Department of Information Technology, Sinhgad Academy of Engineering, Pune, India.

Mayur Ghadge
Department of Information Technology, Sinhgad Academy of Engineering, Pune, India.

Rohit Garde
Department of Information Technology, Sinhgad Academy of Engineering, Pune, India

ABSTRACT

This study presents detection of SQL injection queries by a multi level architecture which uses multiple agents. The SQL injection attacks are one of the biggest security threats in databases. SQL Injection is one of the many web attack mechanisms used by hackers to steal data from organizations. The proposed architecture is based on a hierarchical and distributed strategy where the functionalities are structured on layers. SQL-injection attacks, one of the most dangerous attacks to online databases, are the focus of this research. The agents in each one of the layers are specialized in specific tasks, such as syntax check of queries, data classification, and visualization. The study uses multiple agents in a multi layer architecture, where each agent functions differently and assigns functions to other agent to detect and block SQL injection queries. This study describes two important agents under hybrid architecture: an agent which classifies SQL queries using a Case-Based Reasoning engine based on Legal/illegal/Suspicious. Later if query is still suspicious the query is passed to the human expert by control agents, from where query can be finally classified. The chance of the query reaching to the human expert agent in this system is very low. Thus this study is very effective and efficient to detect and block hazardous SQL injection query fired by an attacker. The system acts as a firewall between an application and database. The use of multi agents helps the cause effectively.

Keywords

MAS, CBR, Detect and Block SQL injection, multi agents, database security.

1. INTRODUCTION

In essence, SQL Injection attack takes place because the fields available for user input allow SQL statements to pass through and query the database directly. It typically involves malicious modifications of the user SQL input either by adding additional clauses or by changing the structure of an existing clause [1]. SQL injection enables attackers to access, modify, or delete critical information in a database without proper authorization [1]. Thus SQL injection remains at the top on list of security threats for databases. The solution proposed to prevent and block this type of attack seems insufficient because they lack the learning capabilities. Also, the majority of these solutions are based on centralized mechanisms, with little capacity to work in distributed and dynamic environments. Although researchers and practitioners have proposed various methods to address the SQL injection problem, current approaches either fail to address the full scope of the problem or have limitations that prevent their use and adoption [5].

With the help of previous research, this study presents Multi-Agent System for detecting and Blocking SQL injection, a solution based on a distributed architecture (multi agent system – MAS) which is capable of detecting and blocking SQL injection attacks [1]. The concept of multi-agent systems makes it possible to deal with SQL injection attacks. Every component in this system interacts and cooperates to achieve a global common goal: the detection and prevention of ongoing intrusions in a database [1]. This system presents a hierarchical organization structured by layers of agents, which distributes roles and tasks to detect and prevent SQL injection attacks. The agents at each level are assigned with specific tasks, due to their own abilities they execute at any physical location [1]. The agents are characterized by the unification of a CBR (Case-Based Reasoning) mechanism in a deliberative Belief - Desire – Intention (BDI) Agent [1]. The mechanism which is provided by the agents have a greater level of adaptation capability and also learning capabilities as CBR systems use past experiences to solve new problems. This technique is effective to block SQL injection attacks as this mechanism uses a strategy which is based on anomaly detection, which model's the normal/legal SQL queries. The main innovations of this study is the incorporation of a new classification strategy which is based on data mining in CBR agents, and of an agent with special capabilities for the visualization and subsequent analysis of data.. The use of CBR agents with advanced capabilities for analyzing and predicting SQL attacks is one of the main features of the architecture [1]. Furthermore, the human experts provides a very useful tool which analyzes those cases which are classified as suspicious by the CBR agent and which requires classification by expert.

2. PRESENT SYSTEMS

Following are the techniques used previously to block the SQL injection queries:

2.1 Blacklist malicious hosts

Once attacker fires SQL injection query from a particular host, that host is banned forever by the organization.

2.2 Pool Resources:

The Database attacks and threats should be updated regularly.

2.3 Minimize Access:

Minimize the access of data or that user or that host from which the attacker attacked the database.

2.4 Encrypt Data:

Never store the data in any plain text form. Rather encrypt any kind of data and then store it.

2.5 Normalize input:

Queries from current pool of threats should be checked.

3. NEED OF APPLICATION

Once SQL injection query is fired by an attacker, he can get access to any database and it can be extremely dangerous if important database is appended by the attacker. So it is very important to block these threats. All the previous attempts to reduce the SQL injection attack were insufficient to do so. But this application is vastly successful to detect and block these attacks with the help of agents and data mining.

As this application acts as a firewall between user application and database servers to stop this attack, it can be applied to any web application or any desktop application.

4. ARCHITECTURE

The present study proposes the use of a multi-agent architecture. It is based on a groundbreaking technique since there is no known architecture with these characteristics for detecting SQL injection attacks. It utilises all the old techniques used to block to block the SQL injection, along with the multiple agents. The distributed resolution of problems balances the workload, facilitates recovery from error conditions, and also avoids centralized traffic [1]. The working analysis, classification and decision making and among others are distributed throughout all three layers in the proposed architecture, as depicted in Fig. 1. The agents that make up the architecture are assigned specific roles to perform their tasks. Moreover, the distribution greatly simplifies the capacity to recover from errors or failures because if an agent fails, it is immediately replaced without affecting the other agents at the same level or in other levels. Additionally, the proposed architecture is based on a hierarchical model that reduces the complexity of tasks such as monitoring and capturing user requests, classifying user requests, evaluating the final solution, etc.

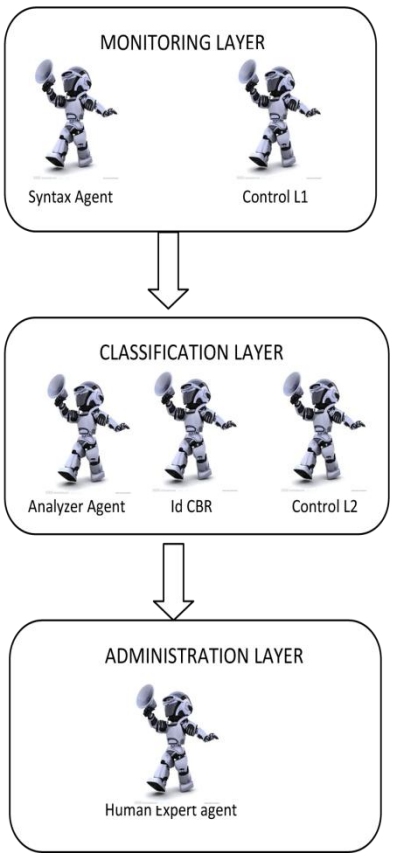

Fig. 1 Architecture

Distributing the functionality at each level, while maintaining each level independently, allows new changes to be easily adapted. Each level of the architecture houses a collection of agents with well-defined roles that allow their tasks and responsibilities to be clearly specified. The architecture has been divided into three levels so that the specific tasks are assigned according to the degree of complexity. Fig. 1 depicts the system architecture with each level and the respective agents. System is presented as an evolution of the SC-MAS architecture that has proposed a strategy to identify and block SQL injection attacks through a distributed approach based on the capabilities of CBR agents. CBR agents are a particular type of CBR-BDI agents. This agent as well as with CBR-BDI agent with their capabilities to visualize is used to assist the expert in decision making regarding queries that are classified as suspicious. To do so, a visualization mechanism is proposed which combines clustering techniques and neural models, based on unsupervised learning, to reduce dimensionality.

The different types of agents located at the different levels of the architecture can be described as:

4.1. Syntax:

This agent is situated in topmost layer in the architecture that is monitoring layer and is responsible for syntax checking of the SQL query fired.

4.2. Control-L1:

Its function is to communicate with the lower layers of the architecture. It is present in the monitoring layer, and all communication from this layer is administered by the agent. This agent receives data from the Syntax agent and assigns the

Analyzer agent the task of searching for patterns of attacks; and then it reports to the human expert in the administration layer the detection of any intrusion during the process of comparing attack signatures. Basically this agent controls and communicates with other levels.

4.3. Analyzer:

This type of agent is situated in the classification layer. Its work is to do matching patterns of known attacks; a database with previously built patterns allows this task.

4.4. CBR:

This type of agent is also situated in the classification layer and is a core component of the architecture as it carries out a classification of SQL strings through detection anomalies. It integrates a case based reasoning (CBR) mechanism. It generates a classification (legal, illegal or suspicious) to the query.

4.5. Control-L2:

This is the second type of agent for carrying out control and communication functions. Once the syntax checking is done in monitoring layer, this agent takes processed data from the monitoring layer and the assigns the work to CBR or Analyzer agent .All of the incoming and outgoing communication of the classification layer is administered by the Control-L2 agent. This agent is responsible for the evaluation and coordination of the overall architectural operation.

4.6. Human Expert:

This agent is located in the Administration layer; this agent facilitates the interaction between security personnel and the architecture. The human gets the query when above two layers does not process the query.

5. USEFULNESS OF APPLICATION

A SQL Injection attack is a form of attack that comes from user input that has not been checked to see that it is valid. The objective is to fool the database system into running malicious code that will reveal sensitive information or otherwise compromise the server.

SQL injection attacks hit Web applications 71 times per hour on average, but can peak at 1,300 unique attacks per hour or more. Consider this security advice to stop SQL attacks.

Security in software applications is an ever more important topic. This system is extremely useful for security as it helps to detect and block hazardous SQL injection query which is one of the biggest threat for modern database system

6. MAS SYSTEM AS A FIREWALL

The System is placed between the user application and the database system. When any user fires any kind of query, that query is first checked by MAS system. The System verifies the query and classifies it accordingly. As the system is placed before the database server, it does not allow any illegal query to access the database server. Fig.2 shows placement of the system in between server and application.

The system classification of the query is done before the query is submitted to the server, this helps server to decide whether the query should be processed or not. With this system in the middle of user application and server, whether query is legal/illegal/suspicious can de showed to user and the server both.

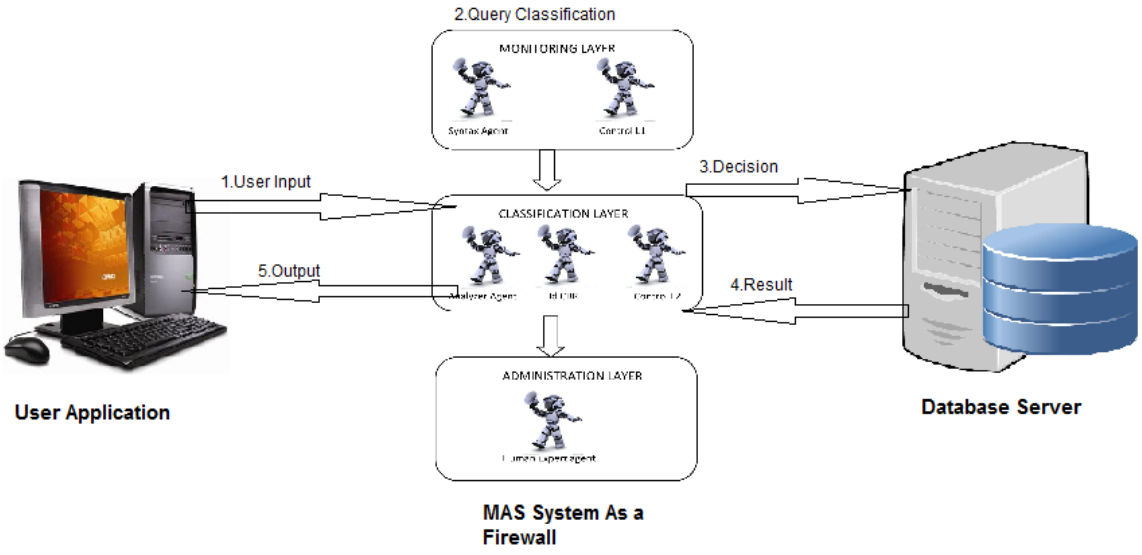

Fig 2 : MAS as a Firewall between user application and database server

7. APPLICATIONS

[1] This system is used to secure database of any web application or any desktop application.

[2] It also classifies the SQL query fired and informs the user whether it legitimate query or suspicious.

[3] As human expert is present in one of the layer, a suspicious query found can be blocked so that database is secure and that suspicious query can be added to the blacklist.

8. CONCLUSION & FUTURE SCOPE

With the help of multi agent system, this study helps to detect and block SQL injection query fired by the attacker. The proposed architecture in general and the CBR and human expert agents in particular, could easily be applied to the detection of application-layer SQL intrusions and provide security to the database.

This System acts as a firewall between the user application and the database of that application. When attacker fires the SQL injection query, that query is restricted from altering any kind of information from the database. The system is built in such a way that the query is characterised, and if a query is a threat, the system does not allow it to process through the firewall; and the approved query is passed and then the legitimate user gets the adequate information from the database. Thus the system presented in this paper is extremely effective and is capable of detecting and blocking the SQL injection attacks which are a big threat for security of databases.

The architecture projected in general and the agents in particular, could easily detect application-layer intrusions. Thus, further study will focus on the improvement in the system by covering any potential vulnerability. For this to happen, we need all the packets involved in the intrusions and gathered by the agents.

9. AKCNOWLEDGEMENT

We take this opportunity to thank all the people involved in making this project a success. We want to thank the authors whose study helped us in our study which are Cristian I. Pinzon, Juan F. De Paz , Alvaro Herrero , Emilio Corchado , Javier Bajo , Juan M. Corchado. We specially thank our guide for guiding us. Our Head of Department has also been very helpful and we appreciate the support he provided us. Last but not the least we would like to convey our gratitude to all the teaching and non-teaching staff members of Information Technology, our friends and families for their valuable suggestions and support.

10. REFERENCES

[1] Cristian I. Pinzon, Juan F. De Paz, Alvaro Herrero, Emilio Corchado, Javier Bajo, Juan M. Corchado idMAS-SQL: Intrusion Detection based on MAS to Detect and Block SQL injection through data mining.

[2] Cristian Pinzon, Álvaro Herrero, Juan F. De Paz, Emilio Corchado, and Javier Bajo: A CBR Intrusion Detector for SQL Injection Attacks.

[3] Cristian Pinzón, Juan F. De Paz, Álvaro Herrero2, Emilio Corchado1, Javier: A Distributed Hierarchical Multi-agent Architecture for Detecting Injections in SQL Queries.

[4] Indrani Balasundaram, Dr. E. Ramaraj: An Approach to Detect and Prevent SQL Injection Attacks in Database Using Web Service.

[5] William G.J. Halfond, Jeremy Viegas, and Alessandro Orso: A Classification of SQL Injection Attacks and Countermeasures.

[6] Varian Luong Intrusion Detection And Prevention System: SQL Injection Attacks.

[7] Christian Bockermann, Martin Apel, and Michael Meier: Learning SQL for Database Intrusion Detection Using Context-Sensitive Modelling.

[8] Sruthy Manmadhan and Manesh: A METHOD OF DETECTING SQL INJECTION ATTACK TO SECURE WEB APPLICATIONS.

[9] Shaimaa Ezzat Salama, Mohamed I. Marie, Laila M. El-Fangary & Yehia K. Helmy: Web Anomaly Misuse Intrusion Detection Framework for SQL Injection Detection.

[10] Lori Mac Vittie: SQL Injection Evasion Detection.