FINDING THE ATTACKS AND PROVIDING HIGH DEFINITIONAL SECURITY LEVEL FOR MULTI TIRE ARCHITECTURE

K. Mounika [1], Srinivasula Reddy D [2]

[1] M.Tech Scholar (CSE), [2] Assistant Professor

Nalanda Institute of Technology (NIT), Siddharth Nagar, Guntur, (India)

ABSTRACT

Network attacks are increased in number and severity over the past few years, intrusion detection system (IDS) is increasingly becoming a critical component to secure the network. Intrusion detection is the process of monitoring and analyzing the events occurring in a computer system in order to detect signs of security problems. Intrusion Detection Systems has the additional job of triggering alarms toward this security problem and some of it automated in the role of triggering or doing an action on behalf of the network administrator. The goal of intrusion detection system (IDS) is to provide another layer of defense against malicious (or unauthorized) uses of computer systems by sensing a misuse or a breach of a security policy and alerting operators to an ongoing attack. In this paper, we have illustrated difficulties to implement IDS in multitier architecture. Since it is difficult to implement multiple IDS, We have introduced a new protocol-Double Guard. Double Guard, is an IDS system that models the network behavior of user sessions across both the front-end web server and the back-end database. It monitors both the web and database request and identifies the attacks like SQL injection attack which independent IDS cannot do. Double Guard, is an IDS system that models the network behavior of user sessions across both the front-end web server and the back-end database. It monitors both web and database request and identifies the attacks like SQL injection attack which an independent IDS cannot do. The limitations of multitier IDS are its training sessions and functionality coverage problem. We have implemented Double Guard using an Apache web server with MySQL and lightweight virtualization. It uses the concept of causal mapping and assigns each client session to container. Each container is associated with an independent container ID and hence it enhances the security. The concept of container is a lightweight virtualization concept that provides a means of tracking the information flow from the web server to the database server for each session. For each client a virtual web server is created.

Keywords: Intrusion Detection System, Anomaly Detection, Web Server, Attacks, SQLIA, Classification of SQLIA.

I. INTRODUCTION

With the tremendous growth of internet and interconnections among network security, computer systems, is becoming a major challenge. Security is the process of detecting and preventing to your system or/and computer from unauthorized users. Detection helps you to determine whether or not someone attempted to break into your system and if they were successful what they may have done. Whereas prevention measures help you to stop or block unauthorized users, from accessing any part of your system. There are various software available for

security but they lack some degree of intelligence when it comes to identifying, recognizing and observing attack signatures that may be present in traffic or in case if there is a backdoor or hole in the infrastructure and that‟s where intrusion detection comes in. IDS categorized into mainly in three types on the basis of kind of activities, system, traffic or behavior they monitor which is host-based, network-based and application-based. Here we are focusing on network intrusion detection system. A network Intrusion Detection System can be classified into two types: anomaly detection and mistreatment detection. Irregularity detection first requires the IDS to define and characterize the correct and acceptable static form and dynamic behavior of the organization, which can next be used to detect nonstandard changes or anomalous behaviors. The boundary between acceptable and anomalous forms of stored code and data is correctly definable. Performance model are built by drama a statistical analysis on historical data or by using rule-based approaches to specify behavior patterns. An anomaly detector then compares actual usage patterns against established models to identify irregular events. Our discovery approach belongs to difference detection, and we depend on a training phase to build the correct model.

In this paper, we have showed the difficulties to implement IDS in multitier architecture. Since it is difficult to implement multiple IDS, We have introduced a new protocol-Double guard. A multi-tier architecture (often referred to as n-tier architecture) is a client–server architecture in which application processing, presentation, and data management functions are logically alienated. The most extensive use of multi-tier architecture is the three-tier architecture. Three-tier architectures typically comprise business tier, a presentation or data access tier, and a data tier Double Guard, is an IDS system that models the network behavior of user sessions across both the front-end web server and the back-end database. It monitors both the web and database request and identifies the attacks like SQL injection attack which independent IDS cannot do. The limitation of multitier IDS is its training sessions and functionality coverage problem. We have implemented Double Guard using an Apache web server with MySQL and lightweight virtualization. Each container is associated with an independent container ID and hence it enhances the security. The concept of container is a lightweight virtualization concept that provides a means of tracking the information flow from the web server to the database server for each session. For each client a virtual web server is created. This paper emphasizes on various aspects of SQL Inspection. Section II shows prevention techniques and operations in the previous work done in this field. Section III contains proposed solution using tokenization approach as well as conclusion part of this paper and future research directions to prevent SQLIA.

II. RELATED WORK

The attacker’s objective for using the injection technique is lies in gaining control over the application database. In a web based application environment, most of the web based banking websites, social web sites, applications, online shopping websites works on the principle of single entry point authentication which requires user identity and password. A user is identified by the system based on his identity.

This process of validation based on user name and password, is referred as authentication. Web architecture illustrated in Fig 1.showes general entry point authentication process. In general client send a HTTP request to the web server and web server in turn send it to the database layer. Database end contains relational tables so queries will be proceeding and result will be send to the web server. So entire process is database driven and each database contains many tables that are why SQLIA can be easily possible at this level. SQL Injection is a basic attack used for mainly two intentions: first to gain unauthorized access to a database and second to retrieve

information from database. Function based SQL Injection attacks are most important to notice because these attacks do not require knowledge of the application and can be easily automated.Oracle has generally aware well against SQL Injection attacks as there is are multiple SQL statements that support (SQL server and Postages SQL), a no. of executive statements (SQL servers) and no. of INTO OUTFILE functions (MYSQL). Also use of blind variables in Oracle environments for performance reasons provides strong protections against SQL Injection attack.

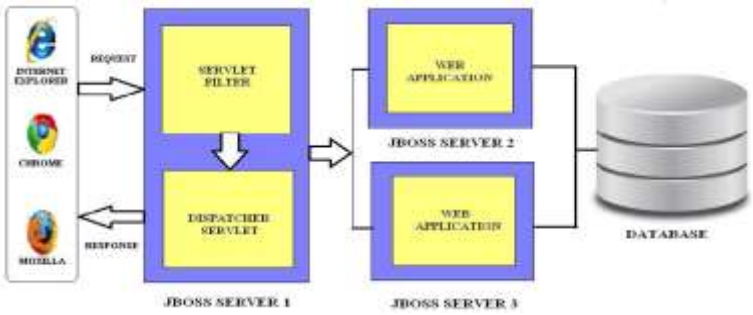

Fig: 1.Web Architecture

There are two types of SQLIA detection: Static approach: This approach is also known as pre-generating approach. Programmers follow some guidelines for SQLIA detection during web application development. An effective validity checking mechanism for the input variable data is also requires for the pre-generated method of detecting SQLIA. Dynamic Approach: This approach is also known as post generated approach. Post generated modus operandi are useful for psychoanalysis of active or runtime SQL queries, create with in user input data by a web application. Detection techniques under this post-generated category executes before posting a query to the database server. Classification of SQLIA: SQLIA can be classified into five categories:

- ✓ Bypass Authentication
- ✓ Unauthorized Knowledge of Database
- ✓ Unauthorized secluded Execution of Procedure
- ✓ Injected Additional Query
- ✓ Injected Union Query

2.1 Bypass Authentication

It is already discussed in Section I. Researchers have proved that query injection can‟t be applied without using space, single quotes or double dashes (--). In bypass authentication, intruder passes the query in such a way which is syntactically true and access the unauthorized data. For example:

SELECT SALARY_INFO from employee where username=‟ or 1=1 - - „and password=”

This SQL statement will be passed because 1=1is always true and - - which is used forcomments, when used before any statement, the statement is ignored. So the result of this query allows intruder to access into user with its privileges in the database.

2.2 Unauthorized Knowledge of Database

In this type of attack, interloper injects a query which causes a syntax, or logical error into the database. The result of mistaken query is shown in the form of error message generated by the database and in many database mistake messages, it includes some information concerning database and intruder can use those details. This type of SQLIA queries is as follows:

SELECT SLARY_INFO from employee where username = „rahul‟ and password =convert(select host from host);

This query is logically and syntactically wrong. The error message can show some information concerning database. Even some error messages display the table name also.

2.3 Unauthorized Remote Execution of Procedure

SQLIA of this type performs a task and executes the procedures for which they are unauthorized. The interloper can access the system and present remote execution of procedure by injecting reservation, for example:

SELECT SALARY_INFO from employee where username=‟‟‟‟; SHUTDOWN; and password =‟‟‟‟;

In this above query, only Shutdown operation is performed which shuts down the database.

2.4 Injected Additional Query

When an additional query is injected with main query and if main query generates Null value, even though the second query will take place and the additional query will harm the database file, for example: SELECT SALARY_INFO from employee where username=‟‟rahul‟‟ and password=‟‟‟‟; drop table user‟‟;

First query generates Null because the space is not present between username and password, but the system executes the second query and if the given table current in database, it will be dive.

2.5 Injected Union Query

In this type of attack, the impostor injects queries which surround set operators. In these queries, the main query generates Null value as a result but attached set operators data from database file for example:

SELECT SALARY_INFO from employee where username=‟‟‟‟ and password=‟‟‟‟ UNION SELECT SALARY_INFO from employee where emp_id=‟‟10125‟‟;

In above queries, the first part of query produce Null value but it allows the intruder to access the salary information of a user having id 10125. Major Elements of SQLIA:

It is shown in various research papers that SQLIA can‟‟t be performed without using space, single quotes and/or double dashes. These are the major elements of SQLIA. SQLIA is occurred when input from a user includes SQL keywords, consequently that the dynamically create SQL query changes the intended function of the SQL query in the application. When user input types a number, there is no need to use single quotes in the query. In this case SQL Injection is injected by using space. This query can be done on original query.

SELECT * from employee where emp_id=10125;

The injection queries can be of this form by means of space:

SELECT * from employee where emp_id=10125 or 1=1;

The injection query shown below is a query which uses single quotes:

SELECT*from employee where emp_name=‟‟rahul‟‟or1=1; In this case if an employee with name rahul is present in database, information is retrieved. But if the name is not present, even then the query is executed becausethestatement1=1 is always true. The injection query may contain double dashes:

SELECT * from employee where emp_name=‟‟rahul‟‟;and SALARY_INFO>25000;

SQLIA is a prominent topic and lots of research work has been done for the detection and prevention of SQLIA. In the author proposes the TransSQL model. In this model author proposes a model for SQLIA prevention. TransSQL is server side application so, it does not changes legacy of web application. This model uses the idea of database duplication and run time monitoring. The proposed model is fully automated and the result shows

the effectiveness of system. TransSQL propose to use two data bases, one is original relational database and another (LDAP) is copy of the first one, But data is arranged in hierarchical form. When a query is break by the user, the system ensures if the query surround the injection or not. The queries inserted in both original LDAP and database. If answer of together databases is same, it shows the input query is free from injection, but if results are dissimilar, it earnings, the query control injection. So the system shows the result as Null values.

The major shortcoming of this models that it is not applicable for injection queries which contain instances, alias, UNION ad UNIONALL [11]. In [9], tokenization method is propose, which is efficient but as well as query with injection. It is not possible for all queries that their origin al query is a ready stored. In [2], the author proposes rule-based detection technique, which is based on classification task. For a particular query, rule dictionary is generated and query is replaced with these rules. If another query is present, the rules are applied in new entry and using classification approach, identify that new query contains the SQL injection or not.[2] proposes, two levels of authentication: SQL authentication and XML authentication, and every query is passed though both systems for checking and preventing against SQLIA.

III. PROPOSED SYSTEM

In the title propose Double Guard which is used to detect attacks in multitier web services. In Double Guard, the new container-based web server architecture enables user to separate the different information flows by each conference. This provides a revenue of tracking the information flow from the web server to the database server for each session. This approach also does not require for user to analyze the source code or know the application logic. Double Guard container architecture is based on Open VZ and lightweight virtualization. Virtualization indicates that each client uses its own virtual web server i.e. each client is processed by a different web server. Thus, highly secure system is provided as each client process is taken as separate session.

3.1 Components of Proposed System

3.1.1 SQL Attack Module

In this module, we have analyzed the four attacks that generally takes place. These attacks are Hijack Future Session Attack, Privilege Escalation Attack, and Injection Attack Direct DB Attack. In Privilege Escalation Attack, the attacker login as a normal user and triggers admin queries so as to obtain an administrator's data. take control future session attack is class of attacks is mainly aimed at the web server side. An assailant usually takes in excess of the web server and therefore hijacks all subsequent legitimate user sessions to launch Attacks. SQL injections do not require cooperation the web server. Assailant can use obtainable vulnerabilities in the web server logic to inject the data or string content that contains the exploits and then use the web server to relay these exploits to attack the back-end database.

3.1.2 Prevention Module

After the server is activated, each client is initiated to use the service. Each client has its own web server i.e. multiple virtual web servers are created in a single system using same service. So each client access through a virtual web server, in this way we can create multiple instance of server. Hence client can access a service through the web server which indicates the basic concept of Double guard architecture. Once client is initiated, it tries to login to use the service. Here we depict the prevention provided against the attacker. The four attacks has been identified and shown how to overcome it. Here only authorized user can login and use the blog. If an

assailant login, his recognized and blocked. No further process can be done by them. Due to the use of multiple web server sometimes attacker get confused about the original server and instance of the server.

3.1.3 Blog Creation Module

In this module, we have showed both Static and dynamic website .Initially the clients logon to his blog. After identifying him as an authenticated user, he can visit the blog. The Home page is an dynamic site as it can be edited and changed. The client can add his profile name or do any changes to his blog. After you click preview, you can see the static site as all contents are static. Changes cannot be made in that site. In the blog you can type the content you want to post and do post. It is like all blog pages where user can post his blog. After all work has done the user can logout from the site which guarantee his security.

3.1.4 Traffic Capture Analysis Module

This module shows the traffic analysis captured between the client and web server and also between the server and database. It provides the overall information regarding the total packet sent, length of the packet and time of receiving of packets. It provides details about the destination IP, source IP and captured time of packet. It also give information about the Ethernet frames, the protocol used i.e. TCPIP etc and details about HTTP protocol. It also provide graphical display of various OSI layers like Network layer, Application layer etc. That is it provide visual scenario of the usage levels of each layer of OSI layer. Using this intruder can be detected as packet size and its all information is available. In this module, Intruder is detected and his activities have been noted. Generally, using the information about the capture time of each packet, last sent packet and its length can be identified. So analyzing this overall information an intruder can be detected. Usually an intruder login and does all his activities. This is stored in the database and can be used to detect the abnormal usage. Subsequent request can be noted and an intruder can be identified. Based on the usage, an Network layer is depicted. It shows the graphical usage pattern.

//Algorithm for Double Guard

Step 1: recognize the input type of HTTP apply for whether it is a query or a request.

Step 2: Store the input values in hash table as per their type AQ for query and for request AR.

Step 3: The key for hash table entrance determination be set as the input itself.

Step 4: Forward AQ and AR to practical server to validate.

Step 5: If attack identified then virtual system automatically terminate the HTTP request.

Step 6: Else HTTP request is onward to the original server.

Step 7: Display information.

Step 8: Exit.

IV. CONCLUSION

We have presented an intrusion detection system that builds models of normal behavior for multitier web applications from both front-end web (HTTP) requests and back-end database (SQL) queries. In the previous approach we have used independent IDS to provide alerts unlike that now we have used, Double Guard which

forms container-based IDS with multiple input streams to manufacture alerts. We have shown that such association of input streams provides a better characterization of the system for anomaly detection because the intrusion sensor has a more precise normality model that detects a wider range of intimidation. We realized this by dividing the run of information from each web server session with a lightweight virtualization. In addition, we quantified the discovery accuracy of our approach when we attempted to model static and dynamic web requests with the back-end file system and database queries. We have built a well-correlated model for static websites, which our experiments proved to be effective at detecting different types of attacks. It also showed that this held true for dynamic requests where both retrieval of information and updates to the back-end database occur using the web server front end. When we deployed our prototype on a system that employed Apache web server, a blog application, and a MySQL back end, Double Guard was able to identify a wide range of attacks with minimal false positives which depended on the size and coverage of the training sessions we used. In this project we use TDT4 method to provide effective summarization methods to extract the core parts of notice topics, as graphic demonstration methods to represent the relationships between the core divisions. Practical mutually, the two techniques, called theme anatomy, can recapitulate necessary in sequence about a topic in a structured manner. In future we can retrieve information from tscan by using our voice instead of typing text. The user can search through voice; the system will recognize this voice and provide essential information about a topic.

REFERENCES

[1] V. Shanmughaneethi, C. EmilinShyni and Dr. S.S wamynathan, "SBSQLI Securing Web Applications with Service Based SQL Injection Detection 2009 International Conference on Advancesin Computing, Control, and Telecommunication Technologies, 978-0-7695-3915-7/09, 2009 IEEE

[2] Yuji Kosuga, Kenji Kono, Miyuki Hanaoka, Hiyoshi Kohoku-ku, Yokohama ,Miho Hishiyama, Yu Takahama, Kaigan Minato-ku, "Sania: Syntactic and Semantic Analysis for Automated Testing against SQL Injection" 23rd Annual Computer Security Applications Conference, 2007, 1063-9527/07, 2007 IEEE

[3] Prof (Dr.) Sushila, MadanSupriyaMadan, "Shielding Against SQL Injection Attacks Using ADMIRE Model", 2009 First International Conference on Computational Intelligence, Communication Systems and Networks, 978-0-7695-3743-6/09 2009 IEEE

[4] A S Yeole, B BMeshram, "Analysis of Different Technique for Detection of SQL Injection", International Conference and Workshop on Emerging Trends in Technology (ICWET 2011) – TCET, Mumbai, India, ICWET‟11, Febr2011, Mumbai, Maharashtra, India. 2011 ACM.

[5] Ke Wei, M. Muthuprasanna, Suraj Kothari, "Preventing SQL Injection Attacks in Stored Procedures". Proceedings of the 2006 Australian Software Engineering Conference (ASWEC‟06).

AUTHOR PROFILE

K Mounika is currently pursuing M.Tech in the Department of CSE, from Nalanda Institute Technology (NIT), siddharth Nagar, Kantepudi(V), Sattenapalli (M), Guntur (D), Andhra Pradesh , Affiliated to JNTU-KAKINADA.

D Srinivasula Reddy working as Assistant Professor at Nalanda Institute of Technology (NIT), siddharth Nagar, Kantepudi(V), Sattenapalli (M), Guntur (D), Andhra Pradesh , Affiliated to JNTU-KAKINADA.