ISSN 2395-1621

Security System For Multi-Tier Web Application

[#1]Snehal Gaikwad, [#2]Pratiksha Ingavale, [#3]Shruti Yewale

[1]snehal44gaikwad@gmail.com
[2]ingavale.pratiksha@gmail.com
[3]shruti.yewale@gmail.com

[#123]Department of Computer Engineering
RMDSSOE, PUNE.

ABSTRACT

Now-a-days usage of internet has increased for various purposes like online shopping, online transaction, internet banking, etc. Almost everything is done online these days. With this increased usage of internet, websites are prone to attacks. Presenting Double Guard, an Intrusion Detection System (IDS) that models the network behavior of user sessions across both the front-end webserver and the back-end database. By monitoring both web and subsequent database requests, it is possible to ferret out attacks that independent IDS would not be able to identify. Our contribution is to find leaked data which is done by hacker. Furthermore, it is also possible to quantify the limitations of any multitier IDS in terms of training sessions and functionality coverage.

Keywords:- **Anomaly detection, virtualization, multitier web application, data leakage detection.**

ARTICLE INFO

Article History

Received :18th October 2015

Received in revised form :

18th October 2015

Accepted : 22th October , 2015

Published online :

23th October 2015

I. INTRODUCTION

Web services are widely used by people. web services and applications have increased in popularity and complexity. As day to day our most of the task such as banking, social networking, and online shopping are done and directly depend on web. Due to the use of web services which is present everywhere for personal as well as corporate data they have been targeted for the attack. Attacker had diverged the front end attack by attacking the backend server which provides the useful and valuable data for the attackers.

Intrusion detection systems have been widely used to detect the attacks which are known by matching misused traffic patterns or signatures to protect the multi-tiered web services. The IDS class has a power of machine learning which can detect unknown attack by identifying the abnormal behaviour of the network traffic action from previous behaviour of IDS phase.

Double Guard is a system which is used to detect the attacks in multitier web services. In this system of Double Guard model of isolated user sessions which include both the web front-end as HTTP and back-end as File or SQL for network transaction. In Double Guard we are going to use lightweight virtualization technique for assigning each user's web session to a dedicated container which provides an isolating virtual environment. So, we will take each web request with its subsequent database queries which will be associate with the accurate container ID. Separate ID is assigned to each container.

By using the container based web architecture which provides isolation that will be helpful in detecting Future Session-Hijack attacks. In lightweight virtualization environment we can use different container each of which are separate from other container for running multiple instances of web server. As container are easily instantiated and destroyed for each user and which is lasting for only short time. If attacker would be able to attack the single user session, the other user sessions remain unaffected because the damage of the single user session is kept within the limit i.e. to that particular session only.

We are making direct causal relationship between the requests received by the front-end web server and those generated for the database back-end for the (website which do not have permissions for content modifications done from user) static website. According to the prior knowledge of web against established models to identify abnormal events. Our detection approach belongs to anomaly detection.

It is our main objective is to create software that is usable, intuitive, simple, and functions well consistently. Our chief concern is that our software be user friendly which means making menus effortlessly navigable and grouping UI components in a manner that makes them easy to find. Giving each employee the appropriate level of access to the

required components is also imperative for usability and security.

SQL injection

SQL injection is nothing but an injection attack wherein an attacker can execute malicious SQL statements that control a web application's database server. Since an SQL injection vulnerability could possibly affect any website or web application that makes use of an SQL-based database, the vulnerability is one of the oldest, most prevalent and most dangerous of web application vulnerabilities.By leveraging SQL injection vulnerability, given the right circumstances, an attacker can use it to bypass a web application's authentication and authorization mechanisms and retrieve the contents of an entire database. SQL injection can also be used to add, modify and delete records in a database, affecting data integrity.To such an extent, SQL injection can provide an attacker with unauthorized access to sensitive data including, customer data, personally identifiable information, trade secrets, intellectual property and other sensitive information.

DDOS attacks

A Distributed Denial of Service (DDOS) attack is an attempt to make an online service unavailable by overwhelming it with traffic from multiple sources. They target a wide variety of important resources, from banks to news websites, and present a major challenge to making sure people can publish and access important information. A malicious hacker uses a DDOS attack to make a computer resource (i.e. – website, application, e-mail, voicemail, network) stop responding to legitimate users. The malicious hacker does this by commanding a fleet of remotely-controlled computers to send a flood of network traffic to the target. The target becomes so busy dealing with the attacker's requests that it doesn't have time to respond to legitimate users' requests.

II. PRESENTED THEORY

A network Intrusion Detection System can be classified into two types: anomaly detection and misuse detection. Anomaly detection first requires the IDS to define and characterize the correct and acceptable static form and dynamic behavior of the system, which can then be used to detect abnormal changes or anomalous behaviors [7], [8]. The boundary between acceptable and anomalous forms of stored code and data is precisely definable. Behavior models are built by performing a statistical analysis on historical data [9], [10], [11] or by using rule-based approaches to specify behavior patterns [12]. An anomaly detector then compares actual usage patterns against established models to identify abnormal events. This detection approach belongs to anom-aly detection, and it depends on a training phase to build the correct model. As some legitimate updates may cause model drift, there are a number of approaches [13] that are trying to solve this problem. This detection may run into the same problem; in such a case, model should be retrained for each shift.

III. PROPOSED SYSTEM

We initially set up our threat model to include our assumptions and the types of attacks we are aiming to protect against. We assume that both the web and the database servers are vulnerable. Attacks are network borne and come from the web clients; they can launch application-layer attacks to compromise the webservers they are connecting to. The attackers can bypass the webserver to directly attack the database server.

We assume that the attacks can neither be detected nor prevented by the current webserver IDS, that attackers may take over the webserver after the attack, and that afterward they can obtain full control of the webserver to launch subsequent attacks. For example, the attackers could modify the application logic of the web applications, eavesdrop or hijack other users" web requests, or intercept and modify the database queries to steal sensitive data beyond their privileges.On the other hand, at the database end, we assume that the database server will not be completely taken over by the attackers. Attackers may strike the database server through the webserver or, more directly, by submitting SQL queries, they may obtain and pollute sensitive data within the database. These assumptions are reasonable since, in most cases, the database server is not exposed to the public and is therefore difficult for attackers to completely take over. We assume no prior knowledge of the source code or the application logic of web services deployed on the webserver. In addition, we are analyzing only network traffic that reaches the webserver and database. We assume that no attack would occur during the training phase and model building.

Data leakage

Data leakage is the big challenge in front of the industries & different institutes. Though there are number of systems designed for the data security by using different encryption algorithms, there is a big issue of the integrity of the users of those systems. It is very hard for any system administrator to trace out the data leaker among the system users. It creates a lot many ethical issues in the working environment of the office.

IV. SYSTEM ARCHITECTURE

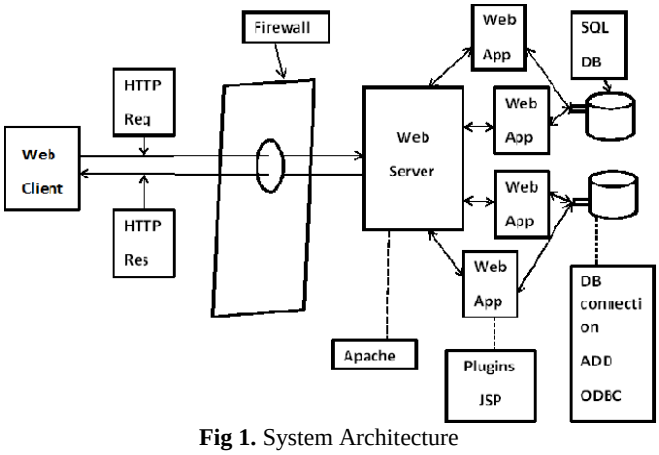

Fig 1. System Architecture

Our system architecture shows that, client send the request in with the help of HTTP request and filter attack with the help of firewall. Firewalls are programs that monitor traffic, which is the incoming and outgoing data communication that takes place when the user is online and act as a frontend guard. Database secured from SQL injection attack at a time of database connection service e.g. ODBC,ADO, etc and act as a backend guard.

V. SYSTEM IMPLEMENTATION

1. Deterministic Mapping

This is the most common and perfectly matched pattern. That is to say that web request rm appears in all traffic with the SQL queries set Qn. For any session in the testing phase with the request rm, the absence of a query set Qn matching the request indicates a possible intrusion. On the other hand, if Qn is present in the session traffic without the corresponding rm, this may also be the sign of an intrusion.

2. Empty Query Set

In special cases, the SQL query set may be the empty set. This implies that the web request neither causes nor generates any database queries. For example, when a web request for retrieving an image GIF file from the same webserver is made, a mapping relationship does not exist because only the web requests are observed. During the testing phase, we keep these web requests together in the set EQS.

3. No Matched Request

Unmatched queries in a set NMR are kept. During the testing phase, any query within set NMR is considered legitimate.

4. Algorithm

Static Model Building Algorithm.

Ensure: The Mapping Model for static website
Input: Set AQ for database query. Set AR for server request
Step 1: Identify the input type of HTTP request whether it is a query or a request.
Step2: for each different request do, if r is a request to static file.
Step 3: Store the input in hash table as per their type AQ for query and for request AR.
Step 4: The key for hash table entry will be set as the input itself.
Step 5: Forward AQ and AR to virtual server to validate.
Step 6: If attack identified then virtual system automatically terminate the HTTP request.
Step 7: Else HTTP request is forwarded to the original server.
Step 8: Display information.
Step 9: Exit.

VI. ADVANTAGES & APPLICATIONS

The system is advantageous in perfect secrecy without leakage of data. This method helps for secure data transfer with confidential data hiding. Its purpose is to ensure privacy by keeping the information hidden from anyone for whom it is not intended. The proposed system used in military imagery, medical imagery, banking sectors, government sector. This system is applicable for all applications where there is a requirement of higher level of security.

VII. CONCLUSION

In recent there is more trend toward use of web services due to their functionality and scalability. But the major problem is requirement of preserving the privacy. The proposed system fulfils this requirement of security.
An intrusion detection system builds models of normal behaviour for multi-tiered web applications from both front-end web (HTTP) requests and back-end database (SQL) queries. Unlike previous approaches that correlated or summarized alerts generated by independent IDSs, Double-Guard forms container-based IDS with multiple input streams to produce alerts. Hence, Double-Guard is able to identify a wide range of attacks with minimal false positives.

REFERENCE

[1] SANS, "The Top Cyber Security Risks," http://www.sans.org/ top-cyber-security-risks/, 2011.

[2] National Vulnerability Database, "Vulnerability Summary for CVE-2010-4332," http://web.nvd.nist.gov/view/vuln/detail? vulnId= CVE-2010-4332, 2011.

[3] National Vulnerability Database, "Vulnerability Summary for CVE-2010-4333," http://web.nvd.nist.gov/view/vuln/detail? vulnId=CVE-2010-4333, 2011.

[4] Autobench, http://www.xenoclast.org/autobench/, 2011.

[5] "Common Vulnerabilities and Exposures," http://www.cve. mitre. org/, 2011.

[6] "Five Common Web Application Vulnerabilities," http://www. symantec.com/connect/articles/five-common-web-application-vulnerabilities, 2011.

[7] H. Debar, M. Dacier, and A. Wespi, "Towards a Taxonomy of Intrusion-Detection Systems,"Computer Networks, vol. 31, no. 9, pp. 805-822, 1999.

[8]T. Verwoerd and R. Hunt, "Intrusion Detection Techniques and Approaches," Computer Comm., vol. 25, no. 15, pp. 1356-1365, 2002.

[9] C. Kruegel and G. Vigna, "Anomaly Detection of Web-Based Attacks," Proc. 10th ACM Conf. Computer and Comm. Security (CCS ¨03), Oct. 2003.

[10] G. Vigna, W.K. Robertson, V. Kher, and R.A. Kemmerer, "A Stateful Intrusion Detection System for World-Wide Web Ser-vers," Proc. Ann. Computer Security Applications Conf. (ACSAC ‟03), 2003.

[11] M. Cova, D. Balzarotti, V. Felmetsger, and G. Vigna, "Swaddler: An Approach for the Anomaly-Based Detection of State Violations in Web Applications," Proc. Int‟l Symp. Recent Advances in Intrusion Detection (RAID ‟07), 2007.

[12] M. Roesch, "Snort, Intrusion Detection System," http://www. snort.org, 2011.

[13] A. Stavrou, G. Cretu-Ciocarlie, M. Locasto, and S. Stolfo, "Keep Your Friends Close: The Necessity for Updating an Anomaly Sensor with Legitimate Environment Changes," Proc. Second ACM Workshop Security and Artificial Intelligence, 2009.

[14] Openvz, http://wiki.openvz.org, 2011.

