ISSN 2395-1621

Double Guard: Detecting intrusions in Multitier web applications with Security

[#1]Amit Patil ,[#2] Vishal Thorat ,[#3]Amit Mane

[1]amitpatil1810@gmail.com
[2]vishalthorat5233@gmail.com
[3]amitmane9975@gmail.com

[#123]Department of Computer Engg,
TSSM'S BSCOER, Pune, Maharashtra, India

ABSTRACT

Internet services and applications have become an inextricable part of daily life, enabling communication and the management of personal information from anywhere. To accommodate this increase in application and data complexity, web services have moved to a multitier design wherein the web server runs the application front-end logic and data are outsourced to a database or file server. In this paper, we present Double Guard, an IDS system that models the network behaviour of user sessions across both the front-end web server and the back-end database. By monitoring both web and subsequent database requests, we are able to ferret out attacks that independent IDS would not be able to identify. Furthermore, we quantify the limitations of any multitier IDS in terms of training sessions and functionality coverage. We implemented Double Guard using an Apache web server with MySQL and lightweight virtualization. We then collected and processed real-world traffic over a 15-day period of system deployment in both dynamic and static web applications. Finally, using Double Guard, we were able to expose a wide range of attacks with 100 percent accuracy while maintaining 0 percent false positives for static web services and 0.6 percent false positives for dynamic web services.

Keywords: **Anomaly detection, virtualization, multitier web application.**

ARTICLE INFO

Article History

Received :20th October 2015

Received in revised form :

21th October 2015

Accepted : 23rdOctober , 2015

Published online :

26th October 2015

I. INTRODUCTION

Web delivered services and applications have increased in both popularity and complexity over the past few years. Daily tasks, such as banking, travel, and social networking, are all done via the web. Such services typically employ a web server front end that runs the application user interface logic, as well as a back-end server that consists of a database or file server. Due to their ubiquitous use for personal and/or corporate data, web services have always been the target of attacks. These attacks have recently become more diverse, as attention has shifted from attacking the front end to exploiting vulnerabilities of the web applications in order to corrupt the back-end database system (e.g., SQL injection attacks).

A plethora of Intrusion Detection Systems (IDSs) currently examine network packets individually within both the web server and the database system. However, there is very little work being performed on multitier Anomaly Detection (AD) systems that generate models of network behavior for both web and database network interactions. In such multitier architectures, the back-end database server is often protected behind a firewall while the web servers are remotely accessible over the Internet. Unfortunately, though they are protected from direct remote attacks, the back-end systems are susceptible to attacks that use web requests as a means to exploit the back end.

To protect multitier web services, Intrusion detection systems have been widely used to detect known attacks by matching misused traffic patterns or signatures. A class of IDS that leverages machine learning can also detect unknown attacks by identifying abnormal net-work traffic that deviates from the so-called "normal" behavior previously profiled during the IDS training phase. Individually, the web IDS and the database IDS can detect abnormal network traffic sent to either of them. However, we found that these IDSs cannot detect cases wherein

normal traffic is used to attack the web server and the database server.

For example, if an attacker with non admin privileges can log in to a web server using normal-user access credentials, he/she can find a way to issue a privileged database query by exploiting vulnerabilities in the web server. Neither the web IDS nor the database IDS would detect this type of attack since the web IDS would merely see typical user login traffic and the database IDS would see only the normal traffic of a privileged user. This type of attack can be readily detected if the database IDS can identify that a privileged request from the web server is not associated with user-privileged access. Unfortunately, within the current multithreaded web server architecture, it is not feasible to detect or profile such causal mapping between web server traffic and DB server traffic since traffic cannot be clearly attributed to user sessions.

We present Double Guard, a system used to detect attacks in multitier web services. Our approach can create normality models of isolated user sessions that include both the web front-end (HTTP) and back-end (File or SQL) network transactions. To achieve this, we employ a light-weight virtualization technique to assign each user's web session to dedicated container, an isolated virtual computing environment. We use the container ID to accurately associate the web request with the subsequent DB queries. Thus, Double Guard can build a causal mapping profile by taking both the web server and DB traffic into account.

II. RELATED WORK

Double guard and its classification:-

Double Guard is a system used to detect attacks in multitier web services [1] [2].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 behaviour of the system, which can then be used to detect abnormal changes or anomalous behaviours. The boundary between acceptable and anomalous forms of stored code and data is precisely definable. Behaviour models are built by performing a statistical analysis on historical data or by using rule-based approaches to specify behaviour patterns. An anomaly detector then compares actual usage patterns against established models to identify abnormal events [1] [2] [3].

Methodology:-

This approach can create normality models of isolated user sessions that include both the web front-end (HTTP) and back-end (File or SQL) network transactions [1] [3]. It employs a light-weight virtualization technique [1] [3] to assign each users web session to a dedicated container, an isolated virtual computing environment. It uses the container ID to accurately associate the web request with the subsequent
DB queries. Double Guard forms container-based IDS with multiple input streams to produce alerts. The correlation 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 threats[1] [2] [3] [4].

Possible Attacks:-

Some of the important attacks are generally used by attackers for hacking i.e. SQL injection, Direct DB Attack ,Hijack future session attack, Privilege escalation[1] [2] [3] [5]and D-DOS attack [1].

Algorithm Used:-

In order to detect such attacks algorithms which are being used are Static model building algorithm [1] [2] [3] [4].

Limitations:-

Vulnerabilities Due to Improper Input Processing :-

Once the malicious user inputs are normalized, Double Guard cannot detect attacks hidden in the values [1].

Possibility Of Evading Double Guard :-

It is possible for an attacker to discover the mapping patterns by doing code analysis or reverse engineering, and issue expected web requests prior to performing malicious database queries [1].

Distributed DOS attacks:-

Previous Double Guard system was not designed to mitigate D-DOS attacks. These attacks can also occur in the server architecture without the back-end database. Denial-of-service attacks are common and fashionable these days. In denial-of service attack, attacker tries to prevent legitimate users from using a service or shutting down a service owing to some implementation vulnerability crashing the machine [1].

III.DOUBLE GUARD SYSTEM ARCHITECTURE

System architecture:-

This is the system architecture design. In this, first the client sends a request for price and other information related to a particular product then that request is analyzed in order to identify if the request is HTTP request or a query and this is done using static model building algorithm.

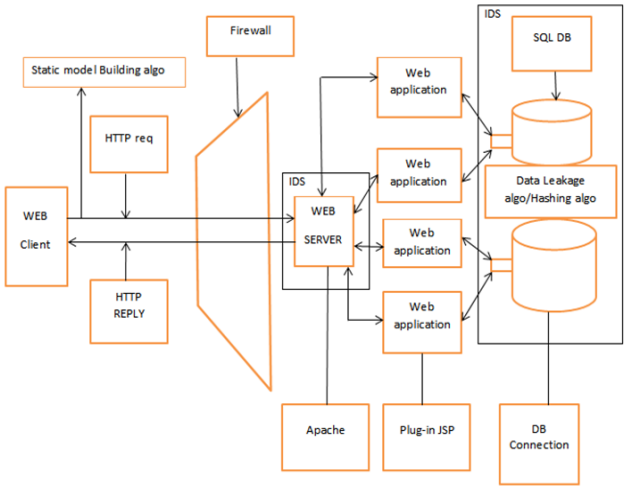

Fig 1: System Architecture diagram

After the request is categorized if the request is HTTP request then that request is passed through firewall and web server receives that request and that request is send as a query to database server and response is sent accordingly but the request is handled only after user authentication is satisfied .If the values in database is changed then data leakage occurs and that's when data leakage algorithm works and saves the
records of unauthorized user and sends it to admin. If a particular authorized user requests for hacked data then previous data is provided to that user and this is done using hashing algorithm.

Attacks scenario:-

1. **SQL-Injection Attacks:-**

Attacks such as SQL injection do not require compromising the web server. Attackers can use existing 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 [fig 2]. Since our approach provides a two-tier detection, even if the exploits are accepted by the web server, the relayed contents to the DB server would not be able to take on the expected structure for the given web server request.

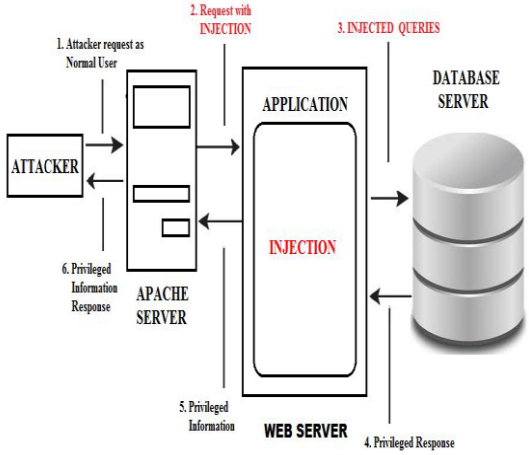

Fig 2. SQL- injection attack

Attacks such as SQL injection do not require compromising the web server. Attackers can use existing 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. Since our approach provides two-tier detection, even if the exploits are accepted by the web server, the relayed contents to the DB server would not be able to take on the expected structure for the given web server request. For instance, since the SQL injection attack changes the structure of the SQL queries even if the injected data were to go through the web server side, it would generate SQL queries in a different structure that could be detected as a deviation from the SQL query structure that would normally follow such a web request.

2. **D-Dos Attacks:-**

Double Guard is not designed to mitigate D-DoS attacks [fig 3]. These attacks can also occur in the server architecture without the backend database. In computing, a denial-of-service (DoS) attack is an attempt to make a machine or network resource unavailable to its intended users, such as to temporarily or indefinitely interrupt or suspend services of a host connected to the internet. A distributed denial-of-service (D-DoS) is where the attack source is more than one–and often thousands–of unique IP addresses. Criminal perpetrators of DoS attacks often target sites or services hosted on high-profile web server such as banks, credit card payment gateways; but motives of revenge, blackmail or activism can be behind other attacks. A distributed denial-of-service (D-DoS) attack occurs when multiple systems flood the bandwidth or resources of a targeted system, usually one or more web servers. Such an attack is often the result of multiple compromised systems flooding the targeted system with traffic.

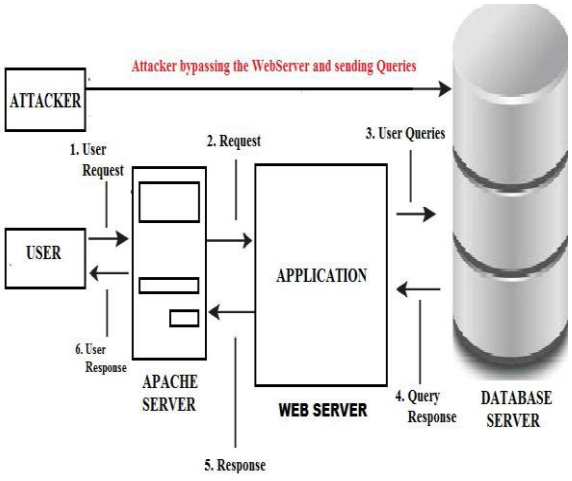

Fig 3. D-DoS attack

When a server is overloaded with connections, new connections can no longer be accepted. The major advantages to an attacker of using a distributed denial-of-service attack are that multiple machines can generate more attack traffic than one machine, multiple attack machines are harder to turn off than one attack machine,

and that the behavior of each attack machine can be stealthier, making it harder to track and shut down. These attacker advantages cause challenges for defense mechanisms. For example, merely purchasing more incoming bandwidth than the current volume of the attack might not help, because the attacker might be able to simply add more attack machines. This after all will end up completely crashing a website for periods of time. Malware can carry D-DoS attack mechanisms; one of the better-known examples of this was My Doom. Its DoS mechanism was triggered on a specific date and time. This type of D-DoS involved hard coding the target IP address prior to release of the malware and no further interaction was necessary to launch the attack.

IV.ALGORITHM

Static Model Building Algorithm (I):-

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.

Step 2: 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.

Data leakage algorithm (II):-

Input: Input data D = D1, D2, D3,.., Dn saves into the hash table.
Step 1: Arrange all input data into matrix format (save into log files).
Step 2: Consider m as a selected data act as a new selected data.
Step 3: m position gets changed after allocated time period.
Step 4: If Ms data get hacked.
Step 5: Data leakage is occurs.
Step 6: We have to check the leakage data and prevent
Step 7: Using Revert back function we have to get original data.
Step 8: When user calls that corrupted file, hash function gives to user a previous data.
Step 9: Return True.

MD5 Hashing algorithm (III):-

MD5 which stands for Message Digest algorithm 5 is a widely used cryptographic hash function The idea behind this algorithm is to take up a random data (text or binary) as an input and generate a fixed size hash value as the output The input data can be of any size or length, but the output hash value size is always fixed

Step 1: Start

Step 2: For each candidate set element.

Step 3: For PV (i) and CV (i) compare attributes and detect which fields are corrupted.

Step 4: get who and when of corruption event.

Step 5: Prepare a report.

Step 6: Stop

V.FUTURE SCOPE

The basic idea is provide two tier securities to for web applications. The aim is to secure the web server from the attacker client and to secure the data from the internal authorized persons in the data care centres. This security model can be further extended to provide security against other attacks.

VII.CONCLUSION

This paper States the design level approach taken by team for the project. In this document, a fair amount of elaboration has been done on the project scenario pointing out the most of the important detail. The goal for the final product has become
apparent as the scenario and the desired user interface is visually explained. Additionally, this report defines proposed system architecture and is discussed with attacks scenario. Further information on the technical design is given and progress is summarized. In this the system architecture is designed for detecting intrusions like SQL injection and D-Dos attacks.

REFERENCE

[1].Mixing Le, Angelos Stavros, Member, IEEE, and Brent ByungHoon Kang, Member, IEEE, IEEE Transactions on dependable and secure computing, Double Guard: Detecting Intrusions in Multitier Web Applications, VOL. 9,NO. 4, March, 2014.

[2]. Mr. Chaudhari Hitesh Kumar, Prof. Ajay V. Nadargi, Mr. Bodade Narendra, Mr. Shinde Sushil , Double Guard: Detecting Intrusions in Multi-tier Web applications, International Journal of Advanced Research in Computer and Communication Engineering Vol. 4, Issue 2, February 2015.

[3].K.Karthika, K.Sripriyadevi, To Detect Intrusions in Multitier Web Applications by using Double Guard Approach., International Journal of Scientific and Engineering Research Volume 4, Issue 1, January-2013.

[4]. ShapnaRani.E, G.Sathesh Kumar, Mythili.R, Karthick.R. Intrusion Detection System for Multitier Web Applications Using Double Guard, International Journal of Engineering And Computer Science ISSN: 2319-7242 Volume 2 Issue 7 (July 2013), Page No. 2162-2166.

[5].Niraj Gaikwad, Swapnil Kandage, Dhanashri Gholap, Double Guard: Detecting and Preventing Intrusions in Multitier Web Applications, Networks and Systems, 2(2), February March 2013, International Journal of Networks and system.

