ISSN 2278-8808
An International Peer Reviewed
SCHOLARLY RESEARCH JOURNAL
FOR INTERDISCIPLINARY STUDIES

"DOUBLE GUARD: DETECTING INTRUSIONS IN MULTI-TIER WEB APPLICATIONS"

Shinde J. R, & Dabhade S. V.
&Mahalle P.N.
Department of Computer Engg,
Smt Kashibai Nawale College of Engg.
Vadgoan , India.

Received: 17 February 2013
Reviewed & Received: 25 February 2013
Accepted: 25 February 2013

Abstract

In today's world there is large amount use of computer especially for web application. Most of the people do their transaction through web application. So there are chances of personal data gets hacked then need to be provide more security for both web server and database server. For that double guard system is used. The double guard system is used to detect & prevent attacks using Intrusion detection system.

This system prevents attacks and prevents user account from intruder from hacking his/her account. By using IDS, system can provide security for both web server and database server using mapping of request and query. An IDS system that models the network behavior of user sessions across both the front-end web server and the back- end database. This system able to search for in a place (container) attacks that previous DoubleGuard would not be able to identify. System will try this by isolating the flow of information from each web server session. It quantify the detection accuracy when system attempt to model static and dynamic web requests with the back-end file system and database queries. For static websites, system built a well-correlated model, for effectively detecting different types of attacks. Moreover, system 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.

Keywords *– Session, Session Id, Query String, Ids*

INTRODUCTION

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 multi-tiered

design wherein the web server runs the application front-end logic and data is outsourced to a database or file server.

In this paper presents DoubleGuard, an IDS system that models the network behavior 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 an independent IDS would not be able to identify. Furthermore, we quantify the limitations of any multi-tier IDS in terms of training sessions and functionality coverage.

In this paper presents DoubleGuard, a system used to detect attacks in multi-tiered 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 lightweight virtualization technique to assign each user's web session to a dedicated container, an isolated virtual computing environment. We use the container ID to accurately associate the web request with the subsequent DB queries. Thus, DoubleGuard can build a causal mapping profile by taking both the web server and DB traffic into account.

The container-based web architecture not only fosters the profiling of causal mapping, but it also provides an isolation that prevents future session-hijacking attacks. to the compromised session; other user sessions remain unaffected by it.

Using our prototype, we show that, for websites that do not permit content modification from users, there is a direct causal relationship between the requests received by the front-end web server and those generated for the database back-end.

In addition to this static website case, there are web services that permit persistent back-end data modifications. These services, which call dynamic, allow HTTP requests to include parameters that are variable and depend on user input. Therefore, our ability to model the causal relationship between the front-end and back-end is not always deterministic and depends primarily upon the application logic.

I. BRIEF REVIEW OF PAPER

Before double guard was developed the system which is present prevents web server and database from Linearization attack only. Before doubleguard not much security provided to the web server and database. This system can not handle all attack. we need to use 2 different technology, one for web server and another for database to prevent from attacks.

Following types of attacks on Web server and database can handled in existing system.

Hijack Future Session Attack:

This class of attacks is mainly aimed at the web server side. An attacker usually takes over the web server and therefore hijacks all subsequent legitimate user sessions to launch attacks. For instance, by hijacking other user sessions, the attacker can eavesdrop, send spoofed replies, and/or drop user requests. A session hijacking attack can be further categorized as a Spoofing/Man-in-the-Middle attack, an Exfiltration Attack, a Denial-of-Service/Packet Drop attack, or a Replay attack.

As each user's web requests are isolated into a separate container, an attacker can never break into other users' sessions.

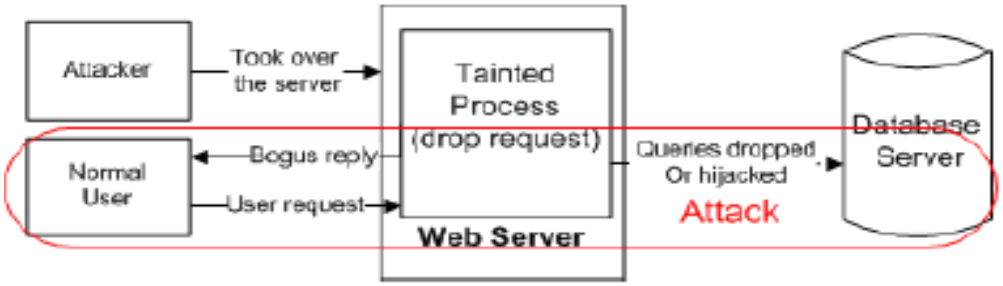

Figure 1.1 Hijack Future Session Attack

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 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. 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.

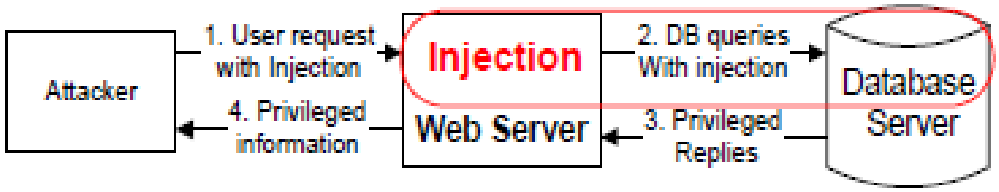

Figure 1.2 Injection Attack

Direct DB attack:

It is possible for an attacker to bypass the web server or firewalls and connect directly to the database. An attacker could also have already taken over the web server and be submitting such queries from the web server without sending web requests. Without matched web requests for such queries, a web server IDS could detect neither. Furthermore, if these DB queries were within the set of allowed queries, then the database IDS itself would not detect it either.

However, this type of attack can be caught with our approach since we cannot match any web requests with these queries.

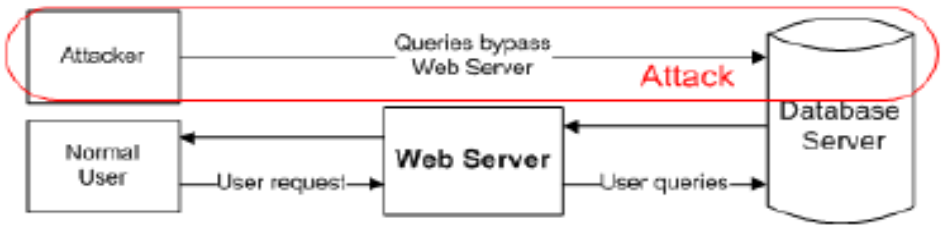

Figure 1.3 Direct DB attack

Following types of attacks on Web server and database can not be handled in existing system.

Input Validation Attack:

If hackers has disabled javascript validation then we can add more security by providing server side validation.

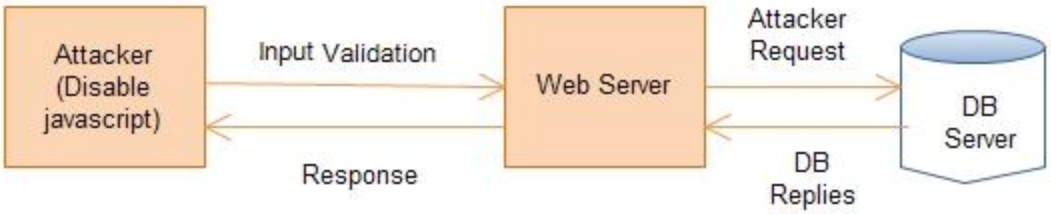

Figure 1.4 Input Validation Attack

Directory Browsing Attack:

Hackers can not directly get list of files on web servers. Directories on the web server or applications are typically locked down to prevent remote browsing when the directory contains executables, text files, documentation, or application-related install or configuration materials. In such cases either the entire directory is configured to block access, or access is granted on a per file basis, requiring a precise request to access objects in the directory. Directory listing can be prevented in server configuration files, but may also arise from vulnerability in a particular application.

Obtaining directory lists allows an attacker to map out the server's directory structure and identify potentially vulnerable files and sample applications. Often, an attacker will use the information gained from directory listings to plan additional attacks against the server. Obtaining directory lists is also useful because it provides a means for determining if other vulnerabilities are present or whether particular application attacks are successful (i.e., by testing whether or not it is possible to create files on the server via a security vulnerability in a particular script or service).

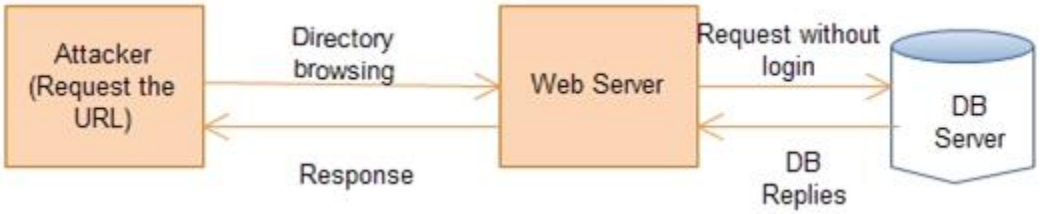

Figure 1.5 Directory Browsing Attack

Brute force attack

A password attack that does not attempt to decrypt any information, but continue to try different passwords. For example, a brute-force attack may have a dictionary of all words or a listing of commonly used passwords. To gain access to an account using a brute-force attack, a program tries all available words it has to gain access to the account. Another type of brute-force attack is a program that runs through all letters or letters and numbers until it gets a match.

Although a brute-force attack may be able to gain access to an account eventually, these attacks can take several hours, days, months, and even years to run. The amount of time it takes to complete these attacks is dependent on how complicated the password is and how well the attacker knows the target.

To help prevent brute-force attacks many systems will only allow a user to make a mistake in entering their username or password three or four times. If the user exceeds these attempts, the system will either lock them out of the system or prevent any future attempts for a set amount of time.

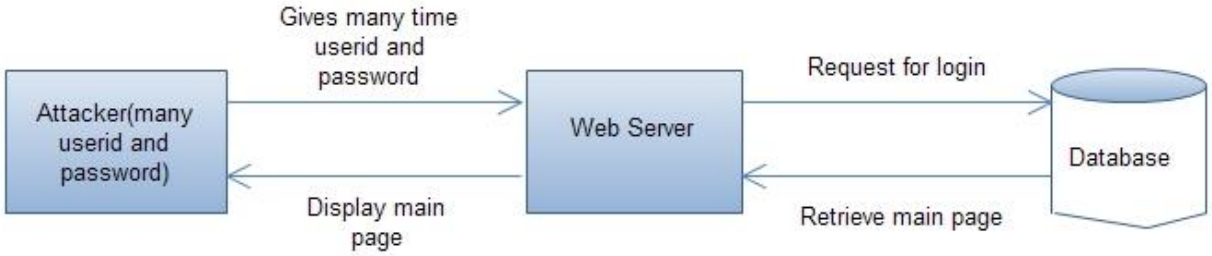

Figure 1.6: Brute force attack

Session replay

Session replay is a scheme an intruder uses to masquerade as an authorized user on an interactive Web site. By stealing the user's session ID, the intruder gains access and the ability to do anything the authorized user can do on the Web site.

Session IDs facilitate user tracking for a Web site and can provide automatic authentication for future visits to that site or associated sites. The session ID can be stored as a cookie, form field or URL. Once the intruder obtains session ID data (*see* session prediction), he can conduct either a session replay or session hijacking attack.

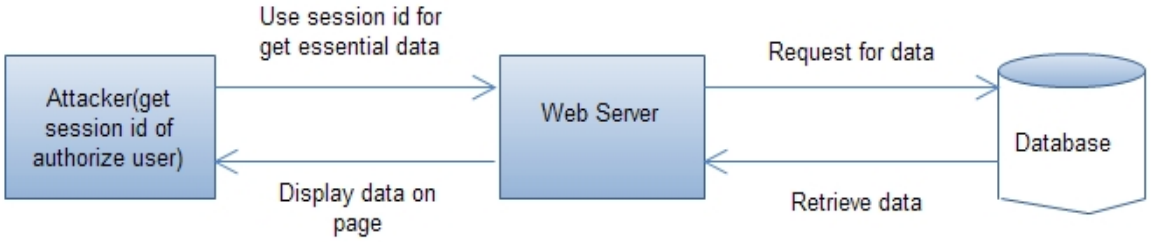

Figure 1.7: Session replay

II . **problem definition and scope**

This paper consist of intrusion detection system to prevent web application from intruders. Paper developed IDS for prevents both website static and dynamic from intruder. The attack which cannot be prevented by IDS that attacks also prevented by double guard. This paper find intruder by using container id that contain session id and IP address of that user.IDS do the mapping of request and query and by mapping it identify user is authorize user or intruder.

REFERENCES

R. Sekar. An Efficient Black-Box Technique For Defeating Web Application Attacks. In Ndss. The Internet Society, 2009.

A. Seleznyov And S. Puuronen. Anomaly Intrusion Detection Systems: Handling Temporal Relations Between Events. In Raid 1999.

Y.Shin, L.Williams, And T. Xie. Sqlunitgen: Test Case Generation For Sql Injection Detection. Technical Report, Department Of Computer Science, North Carolina State University, 2006.

A. Srivastava, S. Sural, And A. K. Majumdar. Database Intrusion Detection Using Weighted Sequence Mining. Jcp, 1(4), 2006.

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. In Proceedings Of The 2nd Acm Workshop On Security And Artificial Intelligence, 2009.

G. E. Suh, J. W. Lee, D. Zhang, And S. Devadas. Secure Program Execution Via Dynamic Information Flow Tracking. Acm Sigplan Notices, 39(11), Nov. 2004.

F. Valeur, G. Vigna, C. Kr¨Ugel, And R. A. Kemmerer. A Comprehensive Approach To Intrusion Detection Alert Correlation. Ieee Trans. Dependable Sec. Comput, 1(3), 2004.

T. Verwoerd And R. Hunt. Intrusion Detection Techniques And Approaches. Computer Communications, 25(15), 2002.

G. Vigna, W. K. Robertson, V. Kher, And R. A. Kemmerer. A Stateful Intrusion Detection System For World-Wide Web Servers. In Acsac 2003. Ieee Computer Society.

G. Vigna, F. Valeur, D. Balzarotti, W. K. Robertson, C. Kruegel, And E. Kirda. Reducing Errors In The Anomaly-Based Detection Of Web-Based Attacks Through The Combined Analysis Of Web Requests And Sql Queries. Journal Of Computer Security, 17(3):305–329, 2009.