International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 10, October - 2013

The Container-Based and Session-Separated Webserver Architecture for Detecting Intrusions in Multitier Web Applications

K. Sravya Bindu [1], N. Srihari Rao [2]

[1] *M.Tech Student, CSE Dept, CMR Institute of Technology, Hyderabad, A.P., MIAENG*

[2] *Associate Prof., CSE Dept., CMR Institute of Technology, Hyderabad, A.P., MIAENG, MCSI*

Abstract

Due to the increase in application and data complexity, a singletiered or doubletiered design becomes unimportant. In order to handle this increase in application and data complexity multitiered design can be used. In a multitiered design, the webserver runs the application front-end logic and database or file server sits at the back-end. We need an architecture that monitors both web and the following database requests in order to identify an attack. DoubleGuard is an Intrusion Detection System (IDS) system which models the network behavior of user sessions across both the front-end webserver and the back-end database. Our project uses DoubleGuard architecture in order to detect intrusions in multitier web applications and shows that DoubleGuard can effectively detect a wide range of attacks with low false positives.

Keywords: *IDS, Webserver, Database Server, Attacker, Multitier Web Applications, World Wide Web (WWW).*

1. Introduction

Banking, travel ticket booking, social networking, messaging, information exchange are all being done via the World Wide Web. All these services typically employ a webserver front end that runs the application user interface logic as well as a back-end server that consists of a database or file server. Web services have always been the target of attackers due to their ubiquitous use for personal and or corporate data. Web application vulnerabilities are used to corrupt the back-end database system, for example, SQL injection attacks.

In general both the web and the database servers are vulnerable. Attacks are network borne and come from the web clients. Attackers 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. If the attacks can neither be detected nor prevented by the current webserver IDS, then the attackers may take over the webserver after the attack and that afterward they can obtain full control of the webserver to launch subsequent attacks. Attackers may strike the database server through the webserver or, more directly by submitting SQL queries. By submitting SQL queries, they may obtain and pollute sensitive data within the database.

In the classic three-tier model as shown in Fig. 1, at the database side, we are unable to tell which transaction corresponds to which client request. The communication between the webserver and the database server is not separated, and we can hardly understand the relationships among them.

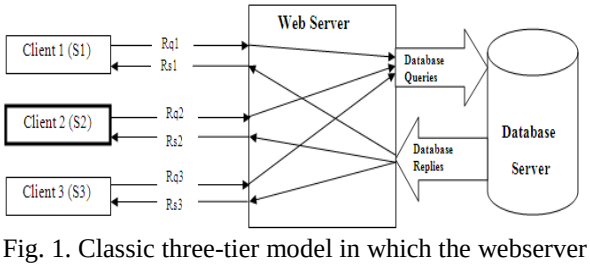

Fig. 1. Classic three-tier model in which the webserver acts as the front-end, with the file and database servers as the content storage back-end.

The web IDS and database IDS can detect abnormal network traffic sent to either of them individually. These IDSs cannot detect attacks when normal traffic is used to attack the webserver and database server. Within the current multithreaded webserver architecture detecting or profiling of causal mapping between webserver traffic and database server traffic is not feasible because traffic cannot be clearly attributed to user sessions.

The organization of the remainder of the paper is as follows. Section II describes the related work. Section III investigates the method employed for our

project. Section IV presents the results of our project work. Section V explains the conclusions and future work.

2. Related Work

The IDS in paper [2] detects known attacks by matching misused traffic patterns or signatures. Anomaly detection IDS in [3] and [4] requires defining and characterizing the correct and acceptable static form and dynamic behavior of the system, which afterwards can be used to detect abnormal changes or anomalous behaviors. We can build behavior models by performing a statistical analysis [5] on historical data or by using rule-based approaches to specify behavior patterns.

Intrusion detection alerts are transformed into brief intrusion reports in order to reduce the number of replicated alerts, false positives, and non relevant positives through a collection of components provided by intrusion alerts correlation [6]. The IDS in [7] uses temporal information to detect intrusions, which has the risk of mistakenly considering independent but concurrent events as correlated events. The IDS in [8] detects intrusions or vulnerabilities by statically analyzing the source code or executables. Information flow can be dynamically tracked [9] to understand taint propagations and detect intrusions. Lightweight virtualization [10] on desktop systems can be used to isolate different application instances.

LAMP is a web application stack that simplifies development and deployment. LAMP stack consists of Linux, Apache, MySQL, Perl/PHP technologies. LAMP stack provides a turnkey system that allows even inexperienced programmers to quickly and easily deploy a full-blown web service. CLAMP [11] is an architecture that adds data Confidentiality to the LAMP model. CLAMP isolates code at the webserver layer, and data at the database layer by users. This isolation guarantees that users' sensitive data can only be accessed by code running on behalf of different users. CLAMP architecture can prevent data leaks even in the presence of attacks. The DoubleGuard architecture [1] that we used for our project, utilized the container ID to separate session traffic as a way of extracting and identifying causal relationships between webserver requests and database query events.

3. Method

DoubleGuard uses the Container-Based and Session-Separated Webserver Architecture for Detecting Intrusions in Multitier Web Applications. DoubleGuard can detect these types of attacks by creating normality models of isolated user sessions that include both the web front-end and back-end network transactions. Lightweight process containers are lightweight, ephemeral and disposable servers for client sessions. These dedicated containers are isolated virtual computing environments to which each user's web session is assigned. Container ID is used to associate the web request with the subsequent database queries. Here, lightweight virtualization is used to isolate different application instances and are therefore useful for isolation and containment of attacks. DoubleGuard can build a causal mapping profile by taking both the webserver and database traffic into account.

As shown in Fig. 2, other sessions' knowledge is not revealed to attackers by restricting the attackers' stay within the webserver container, therefore legitimate sessions will not be compromised directly by an attacker.

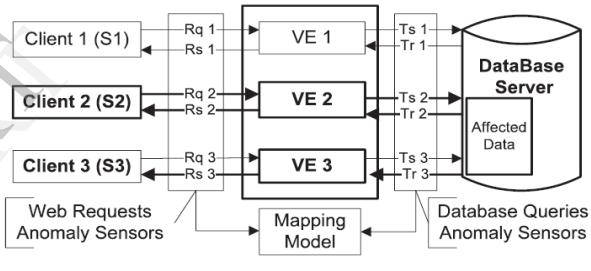

Fig. 2. Webserver instances running in containers.

A. Attack Scenarios

DoubleGuard system is effective at capturing the following types of attacks:

1. Privilege Escalation Attack: The website serves both regular users and administrators. For a regular user, the web request r_u will trigger the set of SQL queries Q_u; for an administrator, the request r_a will trigger the set of admin level queries Q_a. Now suppose that an attacker logs into the webserver as a normal user, upgrades his/her privileges, and triggers admin queries so as to obtain an administrator's data. This attack can never be detected by either the webserver IDS or the database IDS since both r_u and Q_a are legitimate requests and queries (shown in Fig. 3). Our approach, however, can detect this type of attack since the DB query Q_a does not match the request r_a, according to our mapping model.

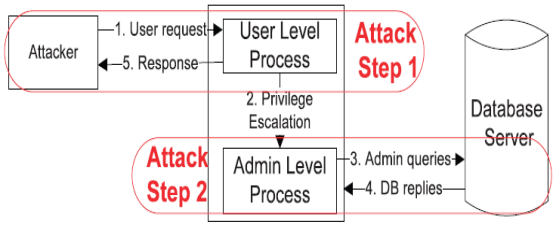

Fig. 3. Privilege escalation attack.

2. Hijack Future Session Attack: This class of attacks is mainly aimed at the webserver side. An attacker usually takes over the webserver 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 compromised webserver can harm all the Hijack future sessions by not generating any DB queries for normal-user requests(shown in Fig. 4). According to the mapping model, the web request should invoke some database queries (e.g., a Deterministic Mapping), then the abnormal situation can be detected. The isolation property of our container-based webserver architecture can also prevent this type of attack.

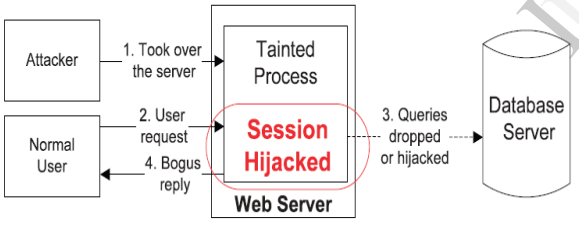

Fig. 4. Hijack future session attack.

3. Injection Attack: Attackers can use existing vulnerabilities in the webserver logic to inject the data or string content that contains the exploits and then use the webserver to relay these exploits to attack the back-end database (shown in Fig. 5). Since our approach provides a two-tier detection, even if the exploits are accepted by the webserver, the relayed contents to the DBserver would not be able to take on the expected structure for the given webserver request. Therefore these are detected as a deviation from the SQL query structure that would normally follow such a web request.

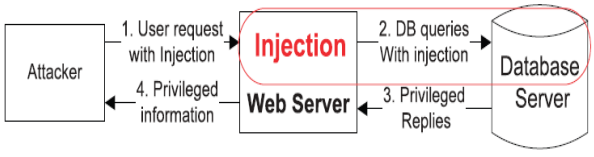

Fig. 5. Injection attack.

4. Direct DB Attack: It is possible for an attacker to bypass the webserver or firewalls and connect directly to the database. An attacker could also have already taken over the webserver and be submitting such queries from the webserver without sending web requests. Without matched web requests for such queries, a webserver 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 (shown in Fig. 6). However, this type of attack can be caught with our approach since we cannot match any web requests with these queries.

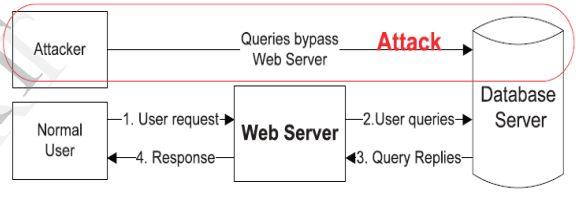

Fig. 6. DB Query without causing web requests.

B. DoubleGuard limitations:

Some of the operational and detection limitations of DoubleGuard:

1. Vulnerabilities due to Improper Input Processing: In DoubleGuard, all of the user input values are normalized so as to build a mapping model based on the structures of HTTP requests and DB queries. Once the malicious user inputs are normalized, DoubleGuard cannot detect attacks hidden in the values.

2. Possibility of Evading DoubleGuard: Within the same session that the attacker connects to, it is allowed for the attacker to launch "mimicry" attacks. 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.

3. Distributed DoS: DoubleGuard is not designed to mitigate DDoS attacks. These attacks can also occur in the server architecture without the back-end database.

4. Results

The following screens are shown to demonstrate as to how our project works using DoubleGuard as the detection architecture.

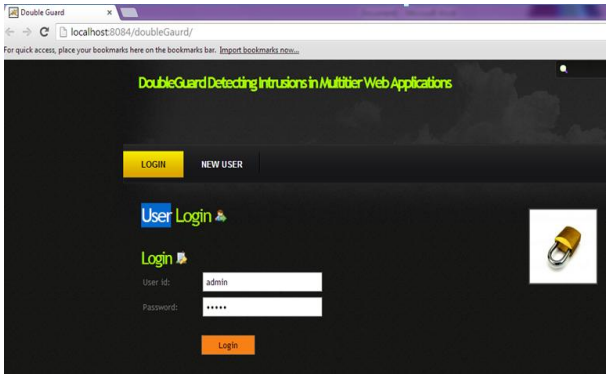

Fig. 7. Administrator login to the web application

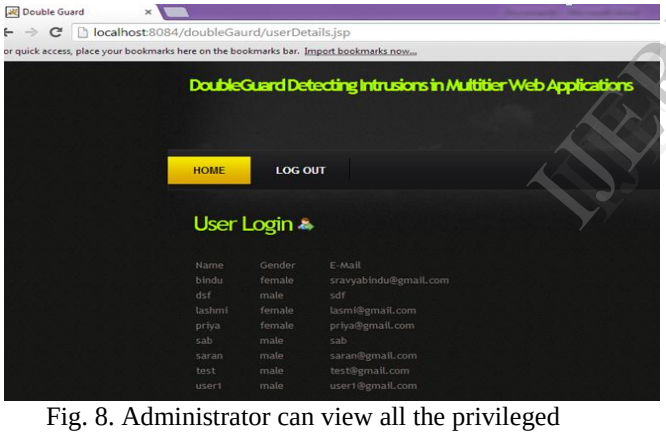

Fig. 8. Administrator can view all the privileged information

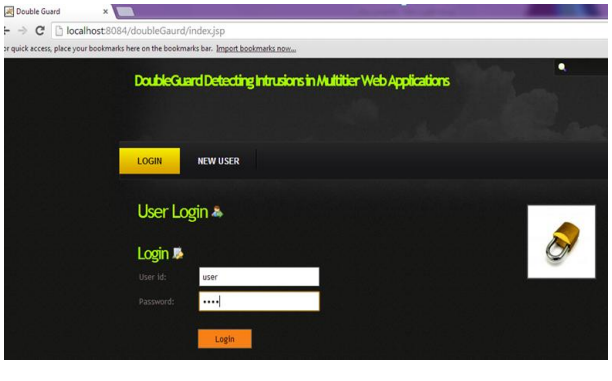

Fig. 9. Normal User login to the web application

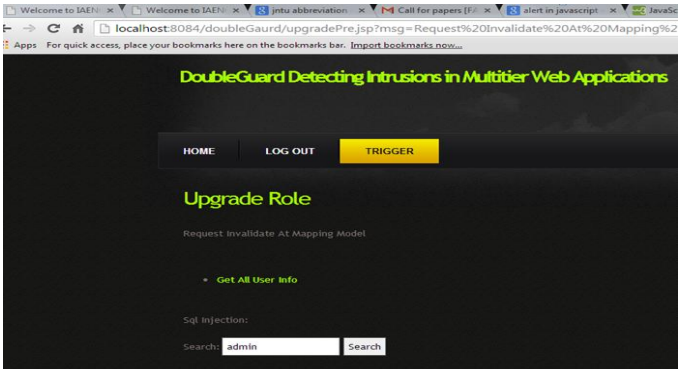

Fig. 10. Attacker fails at privilege escalation attack after login to the system as normal user

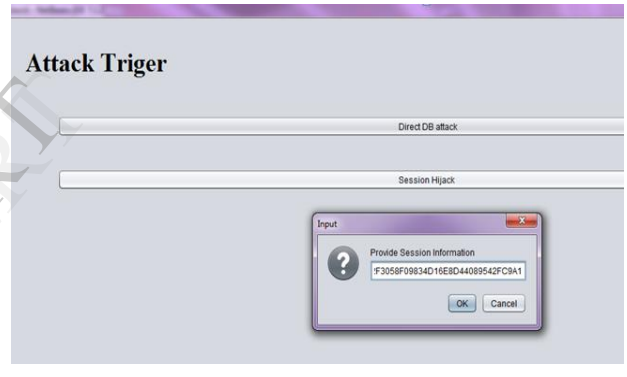

Fig. 11. Attacker gathers session information to hijack all future sessions

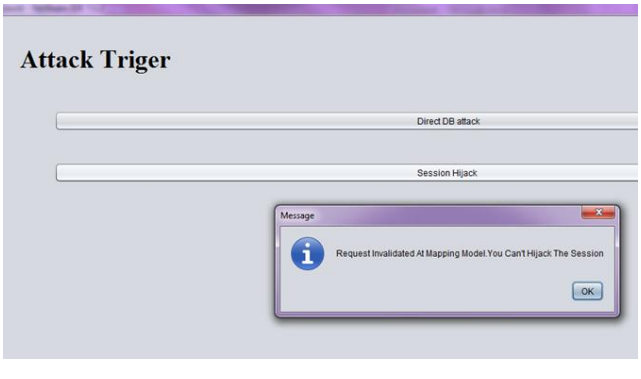

Fig. 12. Attacker fails to hijack all future sessions

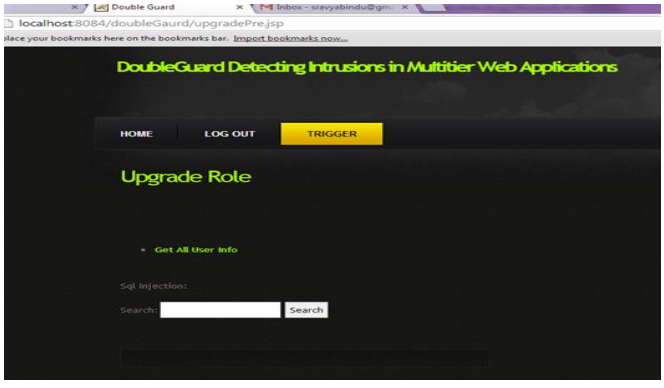

Fig. 13. Attacker fails at SQL injection attacks after login to the system as normal user

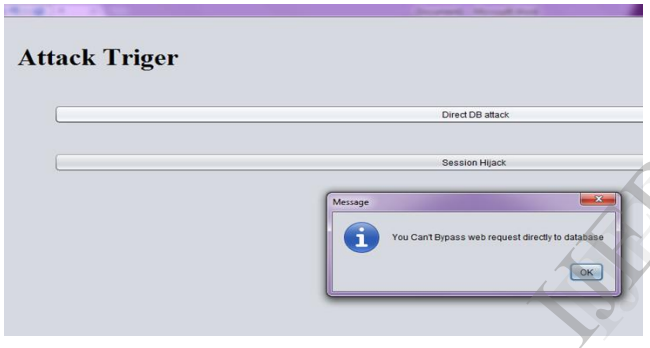

Fig. 14. Attacker fails at direct DB attack

5. Conclusions and Future Work

Existing approaches correlated or summarized alerts generated by independent IDSs, but DoubleGuard, a newer approach forms container-based IDS with multiple input streams to produce alerts. Such 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. Doubleguard achieves this by isolating the flow of information from each webserver session with a lightweight virtualization. For static websites, Doubleguard effectively detects different types of attacks by building a well-correlated model. For dynamic requests, Doubleguard effectively detects different types of attacks where both retrieval of information and updates to the back-end database occur using the webserver front end. For detecting intrusions in multitier web applications, our project used container-based and session-separated webserver architecture. DoubleGuard can identify a wide range of attacks with minimal false positives. The number of false positives depends on the size and coverage of the training sessions that are used.

As discussed above, DoubleGuard contains some operational and detection limitations. Our future work includes performing a thorough analysis on existing systems, and finding the pros and cons of each approach. Then we want to propose a new scheme which will overcome some of these limitations. The aim of our future research work is to propose a novel technique which will fully safeguard both the web server applications and database servers simultaneously from attackers. For example, adding DDoS protection to the prototype of DoubleGuard will save the system from DDoS attacks.

References

[1]. Meixing Le; Stavrou, A.; Kang, B.B., "DoubleGuard: Detecting Intrusions in Multitier Web Applications," *IEEE Transactions on Dependable and Secure Computing*, vol.9, no.4, pp.512-525, July-Aug 2012.

[2]. B.I.A. Barry and H.A. Chan, "Syntax, and Semantics-Based Signature Database for Hybrid Intrusion Detection Systems," *Security and Comm. Networks*, vol. 2, no. 6, pp. 457-475, 2009.

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

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

[5]. G. Vigna, W.K. Robertson, V. Kher, and R.A. Kemmerer, "A Stateful Intrusion Detection System for World Wide Web Servers," *Proc. Ann. Computer Security Applications Conf. (ACSAC '03),* 2003.

[6]. F. Valeur, G. Vigna, C. Kru¨ gel, and R.A. Kemmerer, "A Comprehensive Approach to Intrusion Detection Alert Correlation," *IEEE Trans. Dependable and Secure Computing*, vol. 1, no. 3, pp. 146-169, July-Sept. 2004.

[7]. A. Seleznyov and S. Puuronen, "Anomaly Intrusion Detection Systems: Handling Temporal Relations between Events," *Proc. Int'l Symp. Recent Advances in Intrusion Detection (RAID '99),* 1999.

[8]. D. Wagner and D. Dean, "Intrusion Detection via Static Analysis," *Proc. Symp. Security and Privacy (SSP '01),* May 2001.

[9]. R. Sekar, "An Efficient Black-Box Technique for Defeating Web Application Attacks," *Proc. Network and Distributed System Security Symp. (NDSS)*, 2009.
[10]. Y. Huang, A. Stavrou, A.K. Ghosh, and S. Jajodia, "Efficiently Tracking Application Interactions Using Lightweight Virtualization," *Proc. First ACM Workshop Virtual Machine Security*, 2008.
[11]. B. Parno, J.M. McCune, D. Wendlandt, D.G. Andersen, and A. Perrig, "CLAMP: Practical Prevention of Large-Scale Data Leaks," *Proc. IEEE Symp. Security and Privacy*, 2009.

Authors' Biography

K.Sravya Bindu had B.Tech from Kamala Institute Of Technology & Science, Huzurabad. She is an M.Tech Student in CSE Department of CMR Institute of Technology, Hyderabad. She is currently working for her M.Tech. research project work under the guidance of Mr.N.Srihari Rao. Her areas of interest include Network Security, Computer Networks, and Programming Languages.

N.Srihari Rao had his B.E. from C.B.I.T., Hyderabad, and he had M.E. from Karunya Deemed University, Coimbatore. He is currently working as Associate Professor in CSE Department of CMR Institute of Technology, Hyderabad. He is working for Ph.D. in CSE Discipline at JNTUA University, Anantapur. His areas of interest are Network Security, Data Mining, Image Processing, and ICT for various fields