Middle-East Journal of Scientific Research 20 (2): 185-191, 2014
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2014.20.02.11313

Identifying and Screening Multitier Web Applications for Intrusions Using Triple Guard

B. Nagalakshmi and S. Archana

Department of CSE,
Bharath University, Chennai - 600 073, India

Abstract: As the Internet makes it possible for web servers to publish information to millions of users; it also makes it possible for computer hackers, crackers, criminals, vandals and other "bad guys" to break into the very computers on which the web servers are running. Those risks don't exist in most other publishing environments, such as newspapers, magazines, or even "electronic" publishing systems involving teletext, voice-response and fax-back. Reputations can be damaged and money can be lost if web servers are subverted. Although the Web is easy to use, web servers and browsers are exceedingly complicated pieces of software, with many potential security flaws. A secure computing is considered to be an application or collection of applications that all trust a common security token for authentication, authorization or session management. In this paper, we present the Triple Guard, IDS that the network behaviors of the user sessions across both the front- end and the back-end database. The security levels are strengthened here by means of checking- the malicious user inputs, spoofed IP address and by creating the One Time Password.,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. Also the checking of the spoofed IP makes the hacker completely orn out. The one time password provides the security alert whenever the login is done. Thus, Triple Guard can build a causal mapping profile by taking both the web server and DB traffic into account.

Key words: Intrusion detection system · Virtualization · Triple Guard

INTRODUCTION

The World Wide Web is increasingly being used by corporations and governments to distribute important information and conduct business transactions [1-6]. Reputations can be damaged and money can be lost if web servers are subverted. Although the Web is easy to use, web servers and browsers are exceedingly complicated pieces of software, with many potential security flaws. Web services and applications have increased in both popularity and complexity over the past few years. 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 [7-15]. Intrusion Detection Systems (IDSs) currently examine network packets individually within both the web server and the database system. 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 [16-20]. A class of IDS that leverages machine learning can also detect unknown attacks by identifying abnormal network 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.

Literature Review

The Top Cyber Security Risks: Increasingly, organizations face security risks imposed upon them by attacker's intention achieving fame, glory, or profit.

Corresponding Author: B. Nagalakshmi, Department of CSE, Bharath University, Chennai - 600 073, India.

Attackers, familiar with common vulnerabilities inherent in many of today's websites, know how to exploit those vulnerabilities with attacks designed specifically to take advantage of them. Examples of such destructive activities have recently hit the news in stories about "hacktivist" groups such as LulzSec and Anonymous.

The HP 2011 mid-year edition of the biannual Top Cyber Security Risks Report features in-depth analysis and attack data from HP DV Labs, Application Security Center and Fortify security units as well as vulnerability disclosure data garnered from the OSVDB. Given the media attention paid to these recent attacks, as well as data HP obtained from its partners and customers, the bulk of this report is focused on Web applications, including the vulnerabilities that exist and the attacks that are exploiting those weaknesses.

This report is intended for IT, network and security administrators who are responsible for securing the public-facing communication with an organization's customers, partners and employees. The primary objective of this edition of the Top Cyber Security Risks Report is to clearly articulate the risks and weaknesses inherent in Web applications. We'll highlight the overall vulnerability landscape, including vulnerabilities in commercially available and custom-built applications that can lead to attacks, as well as how often these are being reported. The report will also highlight the rising number of attacks that are leveraging the vulnerabilities discussed throughout the paper [21].

Triple Guard IDS: Triple Guard, a system used to detect attacks in multitiered web services is proposed. 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. Triple Guard mechanism contains a container module at the primary level then to the web service application at the secondary level and finally Database is connected. This connection is used avoid the SQL Injection attacks. Initially the request from the user has to send via the web container to the server. The next level of authentication is by providing the user name, password and privilege and the request forwarding mechanism. Then only the user is allowed to access the web service.

We present a Prototype of DoubleGuard using a webserver with a back-end Database and used to detect attacks in multitiered web services with isolated user

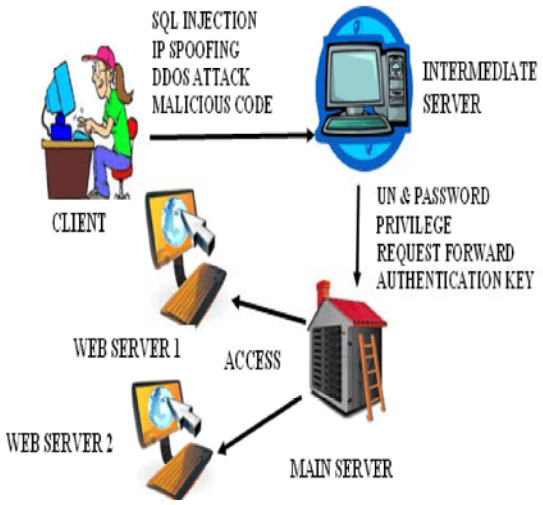

Fig. 1: DoubleGuard Architecture

session that include both HTTP and SQL network transactions. Then, use the virtualization technique to assign each user's web session. In this system used for two types testing website is static and dynamic. In the testing phase, traffic captured in each session is compared with the model. We were able to use the same session tracking mechanism as implemented by Apache server. To evaluate the detection result for our system with analysed attacks. We were able to identify all attacks and to clear the normal traffic.

Intrusion Detection Using Doubleguard

SQL Injection: SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution. Any procedure that constructs SQL statements should be reviewed for injection vulnerabilities because SQL Server will execute all syntactically valid queries that it receives. Even parameterized data can be manipulated by a skilled and determined attacker. The primary form of SQL injection consists of direct insertion of code into user-input variables that are concatenated with SQL commands and executed. A less direct attack injects malicious code into strings that are destined for storage in a table or as metadata. When the stored strings are subsequently concatenated into a dynamic SQL command, the malicious code is executed. A specific syntax is been used for the hacking of the sites. In this paper, the syntax or any other typical commands is not allowed and thus the attackers cant get into the profile, Only the admin can view and handle issues regarding password. A lso if in a chance the attacker tries to attack

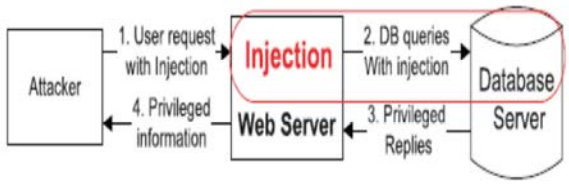

Fig. 2: Injection Attack

the db directly to know the password it cant be known because, the password will be stored in the encrypted form.

IP Spoofing: When authenticating a login page, it is necessary to check for the IP, so that no hackers can access the page using the duplicate IPs. This is much complex and also acts as an additional essential security source. This generally stores the original IP and cross-verify the entered IP and does authentication. This can be avoided by the knowledge preventing system and the unwanted users can be blocked in entering into the network.

DDOS Attack: DDOS is an attempt to make a machine or network resource unavailable to its intended users. Although the means to carry out, motives for and targets of a DDoS attack may vary, it generally consists of the efforts of one or more people to temporarily or indefinitely interrupt or suspend services of a host connected to the Internet.

One Time Password: If the user wants to update their information, an One time password will be generated by the server and send to the user's mobile number to identify the that request was made by the prominent user or not. It also ensures the security by informing the user, if the hacker is attempting to hack is user's account [22].

Container Model: All network traffic from both legitimate users and adversaries is received intermixed at the same web server. If an attacker compromises the web server, he/she can potentially affect all future sessions (i.e. session hijacking). As signing each session to a dedicated web server is not a realistic option, as it will deplete the web server resources. To achieve similar confinement while maintaining a low performance and resource overhead, we use lightweight virtualization. In our design, we make use of lightweight process containers referred to as containers as ephemeral, disposable servers for client sessions. It is possible to initialize thousands of containers on a single physical machine and these virtualized containers can be discarded, reverted, or quickly reinitialized to serve new sessions. A single physical web server runs many containers, each one an exact copy of the original web server.

Our approach dynamically generates new containers and recycles used ones. As a result, a single physical server can run continuously and serve all web requests. However, from a logical perspective, each session is assigned to a dedicated web server and isolated from other sessions. Since we initialize each virtualized container using a read-only clean template, we can guarantee that each session will be served with a clean web server instance at initialization, We choose to separate communications at the session level so that a single user always deals with the same web server. Sessions can represent different users to some extent and we expect the communication of a single user to go to the same dedicated web server, thereby allowing us to identify suspect behavior by both session and user. If we detect abnormal behavior in a session, we will treat all traffic within this session as tainted.

If an attacker compromises a vanilla web server, other sessions communications can also be hijacked. In our system, an attacker can only stay within the web server containers that he/s1he is connected to, with no knowledge of the existence of other session communications. We can thus ensure that legitimate sessions will not be compromised directly by an attacker.

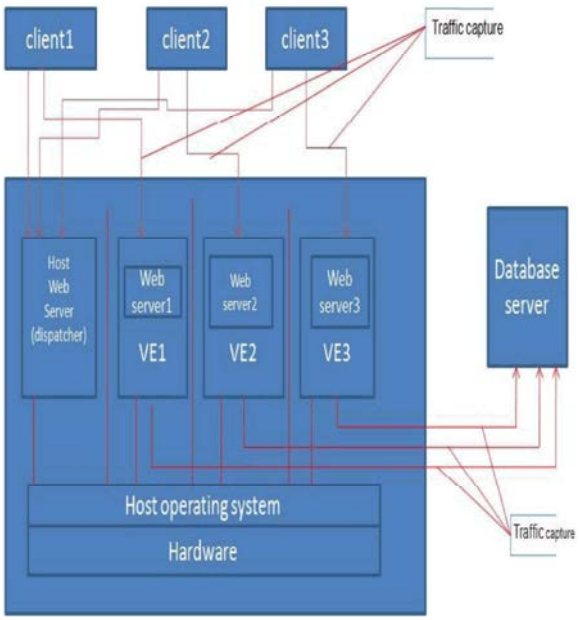

Fig. 3: Web server instances running in containers

Container Generation: The container will be generated for each and every session in the web server the container will provide session id for every session. The data and the information about the query processed are stored in the container.

Query Processing: The user query will be processed. The web server will check the query for authentication purpose after the query is authenticated the web server will process the query and retrieve the data from the database server and it is provided to the user by the web server.

Discarding Container: When the session is closed the container that has been generated to store the information about the query processing should be discarded and when new session starts then the container will be reinitiated.

Disconnecting Server: To finish the session the server is to be disconnected from the user. When the server gets disconnected it is considered to be session completion [23].

Multitier Web application Model

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. In static websites this type of mapping comprises the majority of cases since the same results should be returned for each time a user visits the same link.

The new networking technologies allow spontaneous connectivity among mobile devices, including hand held devices, computers in vehicles, computers embedded in the physical infrastructure and nano sensors. Mobile devices can suddenly become both sources and consumers of information. There is no longer a clean distinction between clients and servers, instead devices are now peers. Furthermore, there is no longer a guarantee of infrastructure support. Consequently, for obtaining data, devices cannot simply depend on a help of some fixed, centralized server. Instead, the devices must be able to cooperate with others in their proximity in order to pursue individual and collective tasks.

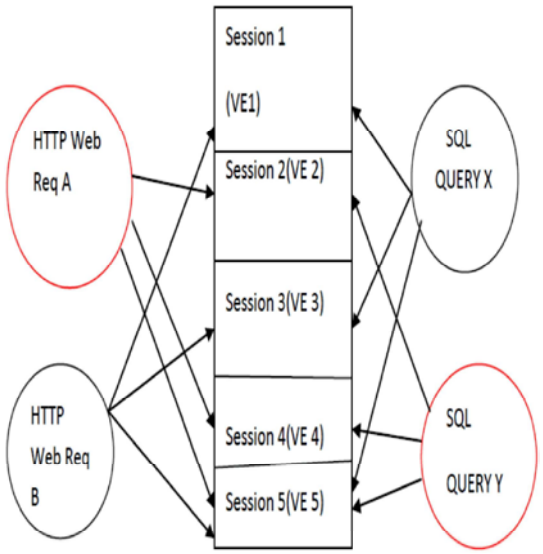

Fig. 4: Deterministic mapping using session ID of the container (VE).

Depending on the offered localization strategy, there are different possibilities to use this information. The cost evaluation of a query execution plan is guided by required resources of the plan. The main factors that are used in stationary systems are CPU usage and the number of hard disk access. In mobile systems, additional varying factors like energy consumption, available memory and CPU speed may be included. Thus, mobile database systems must be able to choose an execution site for the different phases of query processing depending on their current environment and should be able to revise that decision as flexible as possible.

Static Model: Static website can build an accurate model of the mapping relationships between web requests and database queries since the links are static and clicking on the same link always returns the same information. However, some websites (e.g. blogs, forums) allow regular users with non administrative privileges to update the contents of the server data. This creates tremendous challenges for IDS system training because the HTTP requests can contain variables in the past parameters. For example, instead of one to one mapping, one web request to the web server usually invokes a number of SQL queries that can vary depending on type of the request and the state of the system.

Some requests will only retrieve data from the web server instead of invoking database queries, meaning that no queries will be generated by these web requests.

In other cases, one request will invoke a number of database queries. All communications from the clients to the database are separated by a session. Assign each session with a unique session ID. Triple Guard normalizes the variable values in both HTTP requests and database queries, preserving the structures of the requests and queries. To achieve this Triple Guard substitutes the actual values of the variables with symbolic values.

Mapping Relations: In Triple Guard classify the four possible mapping patterns. Since the request is at the origin of the data flow treats each request as the mapping source. In other word, the mappings in the model are always in the form of one request to a query set rm TO Qn.

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 web server is made, a mapping relationship does not exist because only the web requests are observed. This type of mapping is called rm assign empty. During the testing phase, we keep these web requests together in the set EQS.

No Matched Request: In some cases, the web server may periodically submit queries to the database e server in order to conduct some scheduled tasks, such as cron jobs for archiving or backup. This is not driven by any web request, similar to the reverse case of the Empty Query Set mapping pattern. These queries cannot match up with any web requests and we keep these unmatched queries in a set NMR. During the testing phase, any query within set NMR is considered legitimate. The size of NMR depends on web server logic, but it is typically small [24].

Non Deterministic Mapping: The same web request may result in different SQL query sets based on input parameters or the status of the webpage at the time the web request is received. In fact, these different SQL query sets do not appear randomly and there exists a candidate pool of query sets. Each time that the same type of web request arrives, it always matches up with one (and only one) of the query sets in the pool. it is difficult to identify traffic that matches this pattern. This happens only within dynamic websites, such as blogs or forum sites. Decrypts the file.

CONCLUSION

The attacker uses SQL Injection Attack, in order to hit or access the database contents, Intrusion Detection System (IDS), filters abnormal SQL Queries and stores IP address. The counter measures are thus adding an alert by means of one time password and checking for IP Spoofing, also by generating a session key. It also provides verification code through mobile alert when the content was update or modify. After entering this code only updation will takes place? We presented an intrusion detection system that builds models of normal behavior for multitiered 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, DoubleGuard forms container-based IDS with multiple input streams to produce alerts. We have shown that 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. We achieved this by isolating the flow of information from each webserver session with a lightweight virtualization.

Furthermore, we quantified the detection accuracy of our approach when we attempted to model static and dynamic web requests with the back-end file system and database queries. For static websites, we built a well-correlated model, which our experiments proved to be effective at detecting different types of attacks. Moreover, we showed that this held true for dynamic requests where both retrieval of information and updates to the back-end database occur using the webserver front end. When we deployed our prototype on a system that employed Apache webserver, a blog application and a MySQL back end, DoubleGuard was able to identify a wide range of attacks with minimal false positives.

Future Enhancements: Triple Guard was able to identify a wide range of attacks with minimal false positives. As expected, the number of false positives depended on the size and coverage of the training sessions we used. Finally, for dynamic web applications, we reduced the false positives to 0.6 percent [25-30].

REFERENCES

1. Anley, C., 2002. Advanced Sql Injection in Sql Server Applications," technical report, Next Generation Security Software, Ltd.

2. Bai, K., H. Wang and P. Liu, 2005. Towards Database Firewalls, Proc. Ann. IFIP WG 11.3 Working Conf. Data and Applications Security (DBSec '05).
3. Barry, B.I.A. and H.A. Chan, 2009. Syntax and Semantics-Based Signature Database for Hybrid Intrusion Detection Systems, Security and Comm. Networks, 2(6): 457-475.
4. Bates, D., A. Barth and C. Jackson, 2010. Regular Expressions Considered Harmful in Client-Side XSS Filters, Proc. 19th Int'l Conf. World Wide Web.
5. Christodorescu, M. and S. Jha, 2003. Static Analysis of Executables to Detect Malicious Patterns, Proc. Conf. USENIX Security Symp.
6. Cova, M., D. Balzarotti, V. Felmetsger and G. Vigna, 2007. 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).
7. Debar, H., M. Dacier and A. Wespi, 1999. Towards a Taxonomy of Intrusion-Detection Systems, Computer Networks, 31(9): 805-822.
8. Felmetsger, V., L. Cavedon, C. Kruegel and G. Vigna, 2010. Toward Automated Detection of Logic Vulnerabilities in Web Applications, Proc. USENIX Security Symp.
9. Hu, Y. and B. Panda, 2004. A Data Mining Approach for Database Intrusion Detection, Proc. ACM Symp. Applied Computing (SAC), H. Haddad, A. Omicini, R.L. Wainwright and L.M. Liebrock, eds.
10. Huang, Y., A. Stavrou, A.K. Ghosh and S. Jajodia, 2008. Efficiently Tracking Application Interactions Using Lightweight Virtualization, Proc. First ACM Workshop Virtual Machine Security.
11. Kim, H.A. and B. Karp, 2004. Autograph: Toward Automated Distributed Worm Signature Detection, Proc. USENIX Security Symp.
12. Kruegel, C. and G. Vigna, 2003. Anomaly Detection of Web-Based Attacks, Proc. 10th ACM Conf. Computer and Comm. Security (CCS '03).
13. Lee, S.Y., W.L. Low and P.Y. Wong, 2002. Learning Fingerprints for a Database Intrusion Detection System, ESORICS: Proc. European Symp. Research in Computer Security.
14. Liang and Sekar, 2005. Fast and Automated Generation of Attack Signatures: A Basis for Building Self-Protecting Servers, SIGSAC: Proc. 12th ACM Conf. Computer and Comm. Security.
15. Newsome, J., B. Karp and D.X. Song, 2005. Polygraph: Automatically Generating Signatures for Polymorphic Worms, Proc. IEEE Symp. Security and Privacy.
16. Parno, B., J.M. McCune, D. Wendlandt, D.G. Andersen and A. Perrig, 2009. CLAMP: Practical Prevention of Large-Scale Data Leaks, Proc. IEEE Symp. Security and Privacy.
17. Pietraszek, T. and C.V. Berghe, 2005. Defending against Injection Attacks through Context-Sensitive String Evaluation, Proc. Int'l Symp. Recent Advances in Intrusion Detection (RAID '05).
18. Potter, S. and J. Nieh, 2010. Apiary: Easy-to-Use Desktop Application Fault Containment on Commodity Operating Systems, Proc. USENIX Ann. Technical Conf.
19. Robertson, W. F. Maggi, C. Kruegel and G. Vigna, 2010. Effective Anomaly Detection with Scarce Training Data, Proc. Network and Distributed System Security Symp. (NDSS).
20. Sekar, R. 2009. An Efficient Black-Box Technique for Defeating Web Application Attacks, Proc. Network and Distributed System Security Symp. (NDSS).
21. Kerana Hanirex, D. and K.P. Kaliyamurthie, 2013. Multi-classification approach for detecting thyroid attacks, International Journal of Pharma and Bio Sciences, 4(3): B1246-B1251.
22. Khanaa, V., K. Mohanta and T. Saravanan, 2013. Comparative study of uwb communications over fiber using direct and external modulations, Indian Journal of Science and Technology, 6(6): 4845- 4847.
23. Kumar Giri, R. and M. Saikia, 2013. Multipath routing for admission control and load balancing in wireless mesh networks, International Review on Computers and Software, 8(3): 779- 785.
24. Kumarave, A. and K. Rangarajan, Routing alogrithm over semi-regular tessellations, 2013 IEEE Conference on Information and Communication Technologies, ICT 2013.
25. Kumarave, A. and K. Rangarajan, 2013. Algorithm for automaton specification for exploring dynamic labyrinths, Indian Journal of Science and Technology, 6(6).
26. Shafaq Sherazi and Habib Ahmad, 2014. Volatility of Stock Market and Capital Flow Middle-East Journal of Scientific Research, 19(5): 688-692.
27. Kishwar Sultana, Najm ul Hassan Khan and Khadija Shahid, 2013. Efficient Solvent Free Synthesis and X Ray Crystal Structure of Some Cyclic Moieties Containing N-Aryl Imide and Amide,Middle-East Journal of Scientific Research, 18(4): 438-443.
28. Pattanayak, Monalisa. and P.L. Nayak, 2013. Green Synthesis of Gold Nanoparticles Using Elettaria cardamomum (ELAICHI) Aqueous Extract World Journal of Nano Science & Technology, 2(1): 01-05.

29. Chahataray, Rajashree. and P.L. Nayak, 2013. Synthesis and Characterization of Conducting Polymers Multi Walled Carbon Nanotube-Chitosan Composites Coupled with Poly (P-Aminophenol) World Journal of Nano Science & Technology, 2(1): 18-25.

30. Parida, Umesh Kumar, S.K. Biswal, P.L. Nayak and B.K. Bindhani, 2013. Gold Nano Particles for Biomedical Applications World Journal of Nano Science & Technology, 2(1): 47-57.