# *Enhanced doubleguard: detecting persistent cross site scripting attack in web application*

M.NANDHINI
Post Graduate Student - CSE
Dr.Mahalingam College of Engineering and Technology, Pollachi.
Email: nandhini.muruganandham@gmail.com

MR.A.GOWRISHANKAR M.TECH.,
Assistant Professor (SG) - CSE
Dr.Mahalingam College of Engineering and Technology, Pollachi.
Email:ags@drmcet.ac.in

***Abstract*— DoubleGuard an anomaly based intrusion detection system (IDS) detect web based attacks in normal traffic by monitoring web request and database query. As the query input values are normalized, DoubleGuard fails to detect cross site scripting attack. In this paper, we propose a detection system for the detection of cross site scripting attack using filtering mechanism and shadow pages. Filtering mechanism classifies the input as white-list and black-list. White- List filtering specifies the patterns which are allowed in the scripting. Black-List filtering specifies the patterns which are not allowed in the scripting. Shadow pages contain the scripts that do not contain malicious input. Based on the user's input, generated real page is compared with the shadow page to detect attack. Hence it is expected that DoubleGuard can detect 95% of cross site scripting attack in web application.**

***Index Terms*— Intrusion Detection System, White-List filtering, Black-List filtering, Shadow page**.

## I. INTRODUCTION

Internet services and applications have become an important part of daily life, which enables communication and the management of information. With the growth of internet services, security of networking systems has become very important [1]. To provide efficient service to the users, web services have moved to three tier design with client, web server as front end and database server as backend. Many attacks have recently been increased inorder to corrupt the back-end database by exploiting the vulnerabilities in front end.

Intrusion Detection System (IDS) will detect attacks by identifying the network traffic [2]. The traffic that deviates from the normal behavior is considered as attack. It examines the network packets individually within both the web server and the database server and then alerts the system administrator for malicious activity. Both web IDS and database IDS will detect attack in abnormal traffic.

A network Intrusion Detection System can be classified into two types: anomaly detection and misuse detection [2]. Anomaly based IDS define and characterize the correct and acceptable behavior of the system [3]. It detects the anomalous behaviors by comparing the actual patterns against established models. Misuse detection uses the pattern of known attack or weak spots of a system to identify intrusions.

DoubleGuard, a system used to detect attacks in multitier web services. It creates normality models of isolated user sessions that include both the web front-end and back-end. The user session is assigned to the dedicated container using a lightweight virtualization technique. Container ID is used to associate the web request with the subsequent database queries. Then DoubleGuard builds a causal mapping profile by taking both the web server and database traffic into account.

This is used for both static and dynamic website. Static website does not permit the user to modify the content of the web page. In dynamic website, it includes parameters which depend on user input.

## II. RELATED WORK

The detection method of IDS is classified into two types

- Behavior based intrusion detection system
- Knowledge based intrusion detection system

**Behavior based intrusion detection system** [4] is used to detect attack by comparing the actual user's behavior with the expected behavior of the user. The behavior of the user is observed by a number of variables sampled over a time. If the actual behavior deviates from the observed behavior, intrusion is reported by generating an alarm.

**Knowledge based intrusion detection system** [4] applies the knowledge accumulated about a specific attack and the vulnerabilities in the system. The knowledge about the various attacks should be updated regularly.

Kruegel and Vigna (2003) have proposed an anomaly detection system [5] that detects web-based attacks using

Attribute Length, Attribute Character Distribution, Structural Inference, Attribute Presence or Absence. The parameters of the queries are compared with the profiles of the specific the program, which are built already in the training phase. The anomaly detection system takes web server log as input and produces an anomaly score for each web request. The anomaly score is calculated based on the probability value and the weight associated with the model. Attacks are detected based on the calculated anomaly score.

Parno and Andersen (2009) have proposed CLAMP [6], an architecture that adds data Confidentiality to the LAMP model by preventing data leaks in the presence of attacks. It guarantees that the sensitive data can only be accessed by code running on behalf of different users, since the codes are isolated at web server and database server. But in contrast, DoubleGuard focuses on modeling the mapping patterns between HTTP requests and database queries to detect malicious user sessions. DoubleGuard uses process isolation whereas CLAMP requires platform virtualization.

Suh and Lee (2009) have proposed dynamic information flow tracking mechanism [7] to improve security. It tracks the information flow to understand taint propagations and detect intrusions. In DoubleGuard, container-based web server architecture information flow in each session is maintained separately. This provides a means of tracking the information flow from the web server to the database server for each session.

## III. EXISTING WORK

Web applications are three tier architecture, with client as first tier, web server and database server forms the consecutive tiers. Fig 1 illustrates the three tier architecture. The frontend web server receives HTTP requests from client and then issues SQL queries to the database server to retrieve or update the data from the database. IDS detect attack by analyzing the network traffic but cannot detect attack if normal traffic is used to launch attack. So DoubleGuard an anomaly based IDS models the network behavior of user sessions across both web server and database, which is used to detect attack in normal traffic [8].

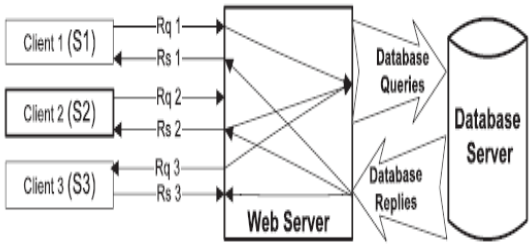


**Fig 1: Three tier architecture**

The request issued by client will be given to the web server, which in turn converts the web request into query. The query will be given to database server. Container based and session separated architecture provides an isolated information flow that is separated in each container session. By monitoring both web request and its database queries, doubleguard is able to detect attacks that independent IDS would not be able to identify.

DoubleGuard creates a normality model of isolated user sessions that include both HTTP request and SQL query. Web request is associated with the subsequent database queries by using the container ID. DoubleGuard detects attack by building a mapping profile by taking both the server into account.

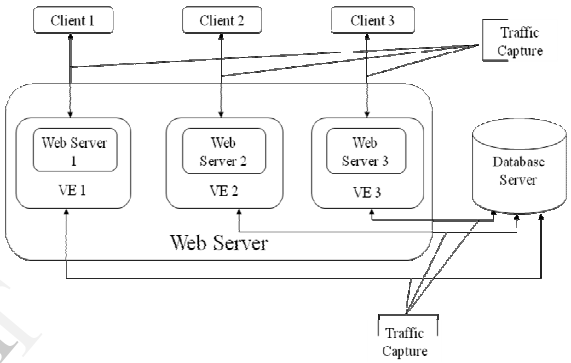


**Fig 2: Architecture of DoubleGuard**

## A. NORMALITY MODEL

Request from different client is intermixed at the web server. So the database server is unable to map the query with the corresponding request. So container-based and session separated web server architecture is employed, which not only enhances the security performances but also provides the isolated information flows that are separated in each container session. Fig 2 depicts the container based architecture.

It allows identification of the database query associated with the request. This mapping model is used to detect abnormal behaviors on a session or client level [8].

## B. NETWORK BEHAVIOR MODEL

Mapping patterns are built by analyzing the behaviors of different attacks such as Privilege Escalation Attack, Hijack Future Session Attack, Injection Attack, Direct DB, etc. Many websites serve only static content, in which the content remains static. In dynamic website, the users can update the content of the web page. For a static website, an accurate model is built by analyzing the relationship between web requests and database queries.

In dynamic website a queries will vary depending on the values of the parameters. The request and queries corresponding to that request are maintained in the database [8]. Dynamic website contains many basic operations such as reading an article, writing an article, posting a comment, visit next page etc... A Common Query Set (CQS) is maintained in database which contains queries for all basic operations of the web application.


**Fig 3: Mapping Pattern**

The mapping pattern is classified into three types.

1. Deterministic Mapping
2. Nondeterministic Mapping
3. Empty Query Set (EQS)

*Deterministic Mapping*

This is most commonly used pattern for static website, where the same web request will generate same query always. The web request $r_m$ in all traffic appears with SQL query set $Q_n$. The mapping pattern is $r_m \rightarrow Q_n$ ($Q_n \neq \emptyset$).

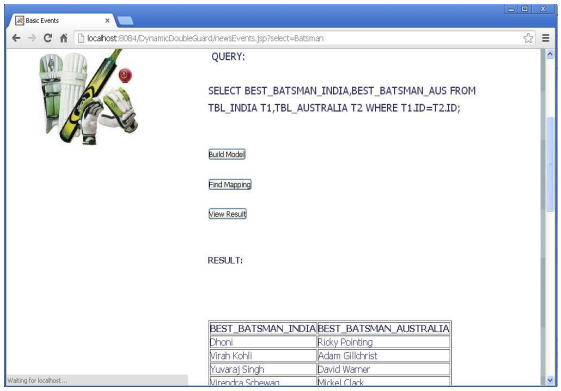


**Fig 4: Displaying result**

*Nondeterministic Mapping*

This mapping pattern is used mainly for dynamic web sites. The same web request may result in different SQL query sets based on input parameters. The mapping pattern is $r_m \rightarrow Q_i$ ($Q_i$ € { $Q_n$ , $Q_p$, $Q_q$…}).

*Empty Query Set*

The SQL query set may be empty for the given request. This implies that the web request does not generate any database queries. The mapping pattern is $r_m \rightarrow \emptyset$. Retrieval of images can be done from web server, so this type of requests will not generate queries to the database server.

Based on the mapping pattern, the request is compared to the query by using the table stored in database. In fig 3, the mapping pattern identified is nondeterministic mapping, since the query varies based on the parameter passed in the web request.

C. MAPPING PROFILE

For all basic operation, traffic captured in each session is compared with the model. Common Query Set (CQS) is formed by combining the query set of all basic operation of the web application. The given web request should match at least one of the request in the model or in Empty Query Set (EQS). If any unmatched web request remains, this indicates that the session has violated the mapping model. Similarly the structure of query is also compared with the common query set maintained in the database. If the structure of the query varies, an intrusion is reported, since attacks such as sql injection will change the query structure.




**Fig 5: Possibility of attack**

Based on the mapping pattern found in network behavior model, the request and query is mapped using the profile. In fig 3, the identified pattern is nondeterministic mapping. So the generated query should map with any of the basic operation of the web application or with the Common Query Set (CQS).

If query does not map with the model, then there is a possibility of an attack. Fig 5 depicts the possibility of an attack since the generated query does not match with the model.

If the identified pattern is Empty Query Set (EQS), then no query will be generated. In such case, the request should match with the profile related to EQS. If the request is found in EQS, the user is valid and the result is displayed.

## IV. PROPOSED SYSTEM

Cross-site scripting (XSS) is a security vulnerability that is found in web applications that allows the injection of malicious code into server by an attacker [9]. Malicious code will be injected into the client system and when the malicious code is executed by the client, it will perform some unwanted actions such as stealing the cookie of the user, redirecting the user to the attacker's site etc,

There are two types of XSS attack namely Persistent and Non persistent cross site scripting attack.

In **Persistent cross site scripting attack**, initially, malicious data will be stored in the server such as database server or file system. When the client sends request to the server, the malicious code will be injected along with the response. Malicious code will be executed in client's browser and steal client's cookie without the user's knowledge. Also referred as stored XSS [9].

In ***Non persistent cross site scripting attack***, web user provides data to the server to instantly generate a resulting page back to the user, a resulting page can be intercepted by an invalid user without html encoding. The malicious client- side code can then be injected into the dynamic page. The attacker can apply a little social engineering to persuade a user to follow a malicious URL that will inject code into the resulting page [9]. It is also referred as reflected XSS vulnerability. The proposed system deals with the detection of persistent cross site scripting attack.

### A. DETECTION MECHANISM

The cross site scripting attack can be detected in the server side using scripts before delivering the response to the client. Two layers of security are provided by using filtering mechanism and shadow pages.

*Filtering mechanism*

Filtering done by using two approaches namely **Black-List filtering** and **White-List filtering**. Black-List filtering specifies the patterns which are not allowed in the scripting. White-List filtering specifies the patterns which are allowed in the scripting [10]. The classification mechanism is used to identify whether the response produced by the database server is white list or black list. The java script may either be static or dynamic scripts based on the web application.

If the response is identified as black list, then the access is denied, and reported as intrusion, otherwise the response is identified as white list, and then the response is allowed for further processing.

*Shadow pages*

For each web application shadow pages are created based on the user input. In order to construct the shadow page, explicitly benign user inputs are used, those that do not contain any Meta characters of the scripting language. When the user gives input, database server will process it and the pages which is produced is known as real pages [10]. Then the real page is compared with the shadow page for the detection of XSS attack.

If the real page does not map with the shadow page, then the access is denied and reported as intruder. If it matches with shadow page, then the response will be given to the user.

DoubleGuard can detect various types of attack such as privilege escalation attack, hijack session attack, direct database attack, SQL injection attack, persistent cross site scripting attack etc.

In the proposed system feature selection using genetic algorithm is used to handle the high-dimensional data efficiently. Preprocessing is performed on the high-dimensional data set in order to handle the missing values. Generally all the feature in a dataset will not be supportive. Hence Feature selection using genetic algorithm is employed to identify the best set of features

## V. CONCLUSION

The normal behavior of multitier web application is built by intrusion detection system using web (HTTP) requests and back-end database (SQL) queries. DoubleGuard, a container-based IDS produces alert by using the multiple input streams, when an intrusion is detected. Persistent cross site scripting attack is detected before providing the data to the user, since the attack is carried out by injecting the malicious script along with the data which is to be delivered to the user. Thus persistent XSS attack can be detected with minimum false positive, by using filtering mechanism and shadow pages.

## VI. ACKNOWLEDGEMENT

I would like to express my gratitude to Mr.A.Gowrishankar, M.Tech., Assistant Professor (SG),

Department of CSE, Dr.Mahalingam College of Engineering and Technology for his useful comments, remarks and engagement through the learning process of this project. Furthermore I would like to thank Ms.G.Anupriya M.E., Assistant Professor (SG), Department of CSE, Dr.Mahalingam College of Engineering and Technology for introducing me to the topic as well as for the support. Also, I would like to thank my family members, who have supported me throughout entire process, both by keeping me harmonious and helping me. I will be grateful forever for their love.

REFERENCES

[1] Stephenb (2012), "Web application", http://en.wikipedia.org/wiki/Web application.

[2] Bgibbs (2012), "Intrusion detection system", http://en.wikipedia.org/wiki/Intrusion detection system.

[3] Yobot (2012),"Anomaly detection", http://en.wikipedia.org/wiki/Anomaly detection.

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

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

[6]. B. Parno, 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.

[7]. G.E. Suh, J.W. Lee, D. Zhang, and S. Devadas (2004), "Secure Program Execution via Dynamic Information Flow Tracking," ACM SIGPLAN Notices, vol. 39, no. 11, pp. 85-96.

[8] Meixing Le, Angelos Stavrou, and Brent ByungHoon Kang (2012), " DoubleGuard: Detecting Intrusions in Multitier Web Applications", IEEE Transactions on Dependable And Secure Computing, VOL. 9, NO. 4 pp.512-525.

[9] MrChupon (2012), "Cross site scripting attack", http://en.wikipedia.org/wiki/Crosssite scripting.

[10] Prithvi Bisht and Venkatakrishnan V N (2008),"XSS-GUARD: Precise Dynamic Prevention of Cross-Site Scripting Attacks",5th International Conference.