April 2013

A Combined Two Step Approach for Detecting Input Validation Attacks Against Web Applications

M. Kishore

KITS, Warangal, kishorem@gmail.com

B. Srinivas

KITS, Warangal, srinu1032@gmail.com

Jayadev Gyani

JITS, Warangal, Jayadevgyani@yahoo.com

Follow this and additional works at: <https://www.interscience.in/ijcsi>

Part of the [Computer Engineering Commons](#), [Information Security Commons](#), and the [Systems and Communications Commons](#)

Recommended Citation

Kishore, M.; Srinivas, B.; and Gyani, Jayadev (2013) "A Combined Two Step Approach for Detecting Input Validation Attacks Against Web Applications," *International Journal of Computer Science and Informatics*: Vol. 2 : Iss. 4 , Article 4.

DOI: 10.47893/IJCSI.2013.1099

Available at: <https://www.interscience.in/ijcsi/vol2/iss4/4>

This Article is brought to you for free and open access by the Interscience Journals at Interscience Research Network. It has been accepted for inclusion in International Journal of Computer Science and Informatics by an authorized editor of Interscience Research Network. For more information, please contact sritampatnaik@gmail.com.

A Combined Two Step Approach for Detecting Input Validation Attacks Against Web Applications

M. Kishore¹, B. Srinivas² & Jayadev Gyani³

^{1&2}KITS, Warangal, ³JITS, Warangal

E-mail : kishorem@gmail.com¹, srinu1032@gmail.com², Jayadevgyani@yahoo.com³

Abstract - Internet becomes a part of our daily life. Almost every service by the internet will be provided with the help of web applications with these we can say that without the web application we cannot do anything over the internet but at the same time web applications are mostly targeted by the hackers. In this paper we present an efficient intrusion detection system approach for detecting input validation attacks against web application. Web application attacks gives chance to the attacker to get unrestricted access to data base and web servers. In this approach we have proposed a two step approach to detect input validation attacks against web applications. By using our approach we can prevent input validation attacks against web applications very efficiently. In the first step we have used a honey pot system to find any malicious data is present or not. In the second step we have used efficient intrusion mechanism to identify the attacks that was not identified in the first step. In this we used an efficient Hirschberg algorithm this is a divide and conquer approach to find attacks against web applications. This system analyzes malicious code and immediately generates an alert to protect web applications from the attacker. By using this technique we can reduce the analysis time and false positive rate.

Keywords - Intrusion detection, web application, Hirschberg Algorithm.

I. INTRODUCTION

An input validations vulnerabilities have been described as one of the most serious threats for web applications [1] Web application input validation attacks may allow an attacker to gain complete access to their underlying systems and data bases and personal sensitive information. These web attacker attacks may damage the systems and may steal the personal sensitive data which is stored in the databases. And these attacks may be used to take control of and corrupt the systems that hosts the web applications. Web applications are vulnerable to input validation attacks are wide spread a study garner group on over 350 internet websites has that most of the them could be vulnerable to input validation attacks. In fact, Input validation have successfully targeted many web sites.

The gaping security loophole in web applications is being exploited by hackers worldwide. According to a survey by the Gartner Group, almost three-fourths of all Internet assaults are targeted at web applications. Web applications are popular attack targets due to the lack of coordination and lack of security awareness on part of the developers. Therefore, the number of security incidents with web applications is rapidly increasing. IVAs can occur when information from web requests is not validated before being used by a web application. An attacker can use these flaws to attack backend components through a web application [2]. Web

applications use network ports to open to public access and various functions are linked with one another, which makes web vulnerable to hacking.

Firewall, common security installation, is not an effective method for IVAs. Because web service is open to public access, it is functioning as a security hole detouring firewall and it cannot differentiate normal requests (web traffic) from attacked ones [3]. When it comes to IDS, it can be signature based or anomaly based in signature based it detects previously known attacks against web applications in anomaly unknown and new attacks can be detected. Our approach is signature based known and previous attacks can be detected and immediately reported to the administrator to take necessary actions against hacker using an efficient algorithm called Hirschberg algorithm.

Previous IDS techniques are less efficient tend to generate many false alarms and time taking approaches. In our demonstration, we present a two step detection method for web attacks in first step we are using honey pot technique and in second step Hirschberg algorithm technique from bioinformatics in order to identify sequential features and improve performance of existing security system. Our research aims to develop *Web Application Intrusion Detection System* (WAIDS) which can detect IVAs by a new intrusion detection method by comparing user submitted request[9]. It uses web request data to analyze the normal request patterns and

HTTP queries that the application user will submit. The resultant system can be more effective because it decreases the false-positive rate and improves performance of existing security system.

II. MODES OF ATTACK

A. SQL Injection

The hacker transmits SQL query commands to the database residing on the server via the Web application. This is done in two ways: SQL commands are entered in form fields on the webpage, or SQL queries are inserted into required input parameters. Thus, the hacker is able to run SQL queries and commands on the server.

B. Cross-Site Scripting

The hacker inserts malicious data into a dynamic webpage. Websites that include only static web pages have control over user interaction because a static webpage is a “read-only” page that does not permit user interaction. Therefore, hacker can only view the page without being able to cause any damage.

However, a dynamic webpage is open to user interaction, so a hacker can insert hazardous content to the dynamic pages the website or web application being not able to differentiate this content from innocuous content. The key to the CSS vulnerability is that a hacker can cause the actual web server to send a webpage with malicious content to the unsuspecting user.

C. Directory Traversal Attacks

This attack is also called the ../ (dot dot slash) attack. With this attack, the Web application is manipulated to allow access to files or other resources on the server that are not normally accessible. The attack works by changing the parameter that an application would use to access a certain file. For instance, suppose the value of the parameter includes the path of a particular file. Placing ../ at the beginning of the parameter value forces the application to access the file in the parent directory. By placing a series of ../ and then giving a different file name at the end, a particular file in the root directory can be retrieved.

D. Parameter Manipulation

This involves manipulating data transmitted between the browser and Web application. Parameter manipulation can be carried out in the following Ways.

Cookie manipulation :

Cookies maintain a certain state in HTTP by storing user preferences and information related to session maintenance. All cookies can be changed at the client

end and then sent to the server with URL requests. Thus, a hacker can easily manipulate the data residing within a cookie.

HTTP header manipulation: HTTP headers consist of control information that is sent from the Web client to the Web server during HTTP requests, and sent from Web servers to Web clients during HTTP responses. Since the HTTP request headers originate from the client, a hacker can easily modify them.

HTML form field manipulation :

Form fields contain values of all the check boxes checked, radio buttons selected, text fields filled or any other action by a user on a particular webpage. This data is then sent to the server. Moreover, there can be hidden fields not visible to the user on the page that are sent to the server. A potential hacker can manipulate the form fields to send any value. One example of this manipulation is to simply right-click the mouse on the webpage to view the source code, alter it, save the changes and then reload the page in the browser.

E. Authentication Attack

The hacker searches for valid authentication to access and enter the server through a web application. To protect from this kind of attack, a database of user names and passwords is maintained in order to maximize authentication and thereby obtain access to restricted domains.

F. Directory Enumeration

Analyzing the website’s entire directory structure, the hacker seeks out hidden directories. These hidden directories could contain administrative data that the hacker may find valuable when launching attacks.

III. COUNTER MEASURES

The analysis infrastructure that we are enhancing in this paper is targeted at the validation of input value in web application. One of the main purposes of IVAs attempt to submit data which the application does not expect to receive. Normally, an application will perform some type of sanity check on user input. This check tries to ensure that the data is useful. More important checks are necessary to detect the data from crashing the server.

A. Countermeasures

Server side counter measures techniques are best defenses against input validation attacks like character encoding, data typing authentication and error handling. Even server side measures are not completely provide security from attacks.

IV. OUR APPROACH

Our approach web application intrusion detection system against input validations attacks based on signature based approach, which has been used to address security problems related to input validation.

In this approach we have used two steps to find attacks. This technique uses two levels to find malicious code. In the first level or initial level the collected input is passed to the supervising module in this module we are we have used a honey pot mechanism to detect malicious data. If the malicious data or code has identified in this level then immediately alert will be generated for the administrator to take necessary actions to prevent attacks. The collected data is sent to the next module if no malicious code was found in this level.

This technique involves 4 modules and these can be in 4 functions to prevent input validation attacks.

A. Data Collection Module

This module receives the data sent by client. User request the web server through GET and POST method. Each argument of request forms (name, address, key word) and the parameters will be collected in this module.

B. Supervising Module

Supervising module gets an input from previous module from the web application and sends it to the honey pot system to check whether the request is to be forwarded for processing or to be rejected or generate an alert to the administrator take necessary actions to prevent attacks. If malicious activities are not identified then that request is to be sent to the next module that is calculation filter and report module for further checking.

C. Extraction Module

Collected data is transformed into alphabetic characters for data filtering according to keyword replacement matrix.

Detection, which is used to find corresponding normal profile. Selected a pair of sequences are aligned and then finally measure for identity.

Hirschberg Algorithm :

Hirschberg algorithm is divide conquer version of the Needleman-Wunch algorithm. This algorithm used in computational biology of global alignments of DNA and protein sequences. This is algorithm for finding optimal sequence. If x and y are strings, where $|x|=n$ and $|y|=m$, the Needleman-Wunch algorithm finds an

optimal alignment in $O(nm)$ time , using $O(nm)$ space. Hirschberg's algorithm is a clever modification of Needleman-Wunch algorithm which takes $O(nm)$ time, but needs only $O(\min\{m,n\})$ space

Fig.1: Combined approach for preventing input validation attacks

D. Calculation, Filter and Report Module

This module gets the input from extraction module and it uses Hirschberg algorithm for string comparison this uses divide and conquer methodology to prevent input validation attack. Any malicious activity is detected by this module immediately reported to administrator to take appropriate actions against attacker to provide security to the data.

Transformed request sequence is calculated with normal web request profile to get OSD (*Optimal Sequence* With this we can say that our approach is very effective and efficient one.

$$F(i,j) = \text{Max} \{ F(i-1,i-1) + s(x_i + y_i), F(i,j-1) + d, F(i-1,j) + d \}$$

d: Gap Penalty

TABLE I

Hirschberg Algorithm using an example string

F(i,j)	A	C	T	G	A
A	F(i-1,j-1)	F(i,j-1) + d			
C	F(i-1,j)+d				
T					

There are three paths in scoring matrix for reaching a particular position i,j :

For pair wise sequence, Needleman-Wunch algorithm and Smith-Waterman are mostly used. Both use dynamic programming based on identical mathematical background in order to locate most appropriate sequence alignment.

But in order to reduce space and time complexity it uses divide and conquer approach. This algorithm takes time complexity as $O(mn)$ in worst case and space complexity is $O(\min(nm))$ for two sequences.

- (1) A diagonal move from position $i-1, j-1$ to position i, j with no gap penalties
- (2) A move from any position column j to i, j with gap penalty
- (3) A move from any position in row i to i, j with a gap penalty.

But this system could not consider gap penalty. It directly match between the two sequences. For two sequences $a = a_1a_2a_3...a_n$ and $b = b_1b_2b_3...b_n$ where $s_{ij} = s(a_1, a_2... a_i, b_1, b_2 b_j)$.

Where s_{ij} is the score at position i in sequence a and position j in sequence b , $s(a_i b_j)$ is the score of alignment the characters at positions i and j , d is the penalty for a gap of length x in sequence a and d is the penalty for a gap of length y in sequence b .

S_{ij} is type of running best score as the algorithm moves through position in matrix. Eventually, all the matrix positions are filled. If a global alignment that involves all of the sequence is required, the matrix score in the

last row and last column is used in the alignment score. Use of dynamic programming method requires a scoring system for the comparison of the pairs, and a scheme for insertion and deletion penalties.

This method is based genetic distance between sequence pairs. The genetic distance between two sequence is the fraction of aligned position in which sequence has been changed. In our present system, it maintains the tables as it is like gene comparison that it contains keywords which are present in horizontal vertical line and it will compare the incoming tokens with predefined values using this algorithm for identity.

A. SQL Injection Attacks

We have taken SQL injection attack to demonstrate the web application intrusion detection against input validation attack.

In this SQL statement which has been sent to calculation and filter module to check whether this is a malicious one or normal request.

SQL statement:

```
SELECT id FROM logins WHERE username = 'Username'
```

```
AND password = '$password'
```

TABLE II

Using Hirschberg Algorithm to find similarities

(I,J)	SELECT	*	From	Dept	Where	Usr	=	"	"	And	passwd	=	"	"	;
SELECT	1														
*		1													
From			1												
Dept				1											
Where					1										
Usr						1									
=							1								
"								1							
"									1						
And										1					
Passwd											1				
=												1			
"													1		
"														1	
;															1

In similar way we can detect other remaining input validation attacks.

As per Hirschberg algorithm, it divides the token and it checks the each token with predefined tokens using the divide and conquer methodology.

```
Select * from dept where usr="kits" and  
passwd="anytext" or '1'='1'.
```

This calculation and filter module detects SQL injection taken place anytext this token to prevent SQL injection attacks.

The algorithm uses divide and conquer method as described here, X strings are stored in horizontal line where a Y stores in vertical line. Being a divide and conquer method, it divide one string problem in two sub problems. It compares one string with predefined data and it compares another sub problem with another set to match comparison and it combines the sub problem solutions to main problem solutions. So it makes time complexity as $O(nm)$ and it needs space complexity is $O(\min(m,n))$.

B. Cross Scripting Attacks

Cross Site Scripting is generally believed to be one of the most common application layer hacking techniques. Today, dynamic websites rely heavily on complex web applications to deliver different output or content to a wide variety of users. Dynamic websites suffer from serious cross site scripting attacks on their data.

A web page contains both text and HTML markup that is generated by the server and interpreted by the client browser. Web sites that generate dynamic pages do not have complete control over how their outputs are interpreted by the client. The heart of the issue is that if mistrusted content can be introduced into a dynamic page, neither the web site nor the client has enough information to recognize that something has happened and take protective actions." Cross Site Scripting allows an attacker to embed malicious JavaScript, VBScript, ActiveX, HTML, or Flash into a vulnerable dynamic page to fool the user, executing the script on his machine in order to gather data. The use of XSS might compromise private information, manipulate or steal cookies, create requests that can be mistaken for those of a valid user, or execute malicious code on the end-user systems. The data is usually formatted as a hyperlink containing malicious content and which is distributed over any possible means on the internet.

As a simple example, imagine a search engine site which is open to an XSS attack. The query screen of the search engine is a simple single field form with a submit button. Whereas the results page, displays both the matched results and the text you are looking for. Search Results for "XSS Vulnerability" To be able to bookmark pages, search engines generally leave the entered variables in the URL address. In this case the URL would look like: <http://test.searchengine.com/search.php?q=XSS%20Vulnerability>

Next we try to send the following query to the search engine:

```
<script type="text/javascript"> alert ('This is an XSS  
Vulnerability') </script>
```

By submitting the query to search.php, it is encoded and the resulting URL would be something like:

```
http://test.searchengine.com/search.php?q=%3Cscript%20%3E  
Ealert%28%91This%20is%20an%20XSS%20Vulnerability%92%2
```

```
%3C%2Fscript%3E
```

Upon loading the results page, the test search engine would probably display no results for the search but it will display a JavaScript alert which was injected into the page by using the XSS Vulnerability.

C. Directory Traversal Attacks

The goal of a Directory Traversal attack is to execute commands that will access files that are intended to be restricted. If an application developer fails to provide code to validate browser input, hackers can experiment with different input strings and access directories above the root.

When Directory Traversal vulnerabilities exist, an attacker is able to access directories above the root directory. Once these directories are accessed, commands can be executed on the Web server and secure data can be copied or modified. The security of the Web server is severely compromised.

For example, consider a Web application that processes an input field on a web page. The application takes character string from the input field and forms a URL that displays a dynamic page. The URL might look like this <http://www.mycalendar.com/mydata.asp?item=tuesday.html>

A hacker will notice that the parameter on the right side of '=' is a web page file and guess that 'mydata.asp' is a script that can retrieve files. This URL is then copied by the hacker and edited. The hacker then inputs this new URL into the browser's URL field. <http://www.mycalendar.com/mydata.asp?item=../../WINNT/win.ini>

In the above example, the characters '../' are used to access directories above the root directory. The hacker is experimenting to see if the WINNT directory is four levels up. If this is not the case, the hacker will try adding or subtracting '../' characters.

If vulnerability exists in the Web application and the input is not filtered, the hacker will eventually find the correct sequence to access the WINNT directory and then execute malicious commands

D. Parameter Manipulation

The Web Parameter manipulation (tampering) attack is based on the manipulation of parameters exchanged between client and server in order to modify application data, such as user credentials and permissions, price and quantity of products, etc. Usually, this information is stored in cookies, hidden form fields, or URL Query Strings, and is used to increase application functionality and control.

For example, consider a user who can select form field values (combo box, check box, etc.) on an application page. When these values are submitted by the user, they could be acquired and arbitrarily manipulated by an attacker. When a web application uses hidden fields to store status information, a malicious user can tamper with the values stored on his browser and change the referred information. For example, an e-commerce shopping site uses hidden fields to refer to its items, as follows

```
<input type="hidden" id="1008" name="cost" value="70.00">
```

In this example, an attacker can modify the “value” information of a specific item, thus lowering its cost.

V. CONCLUSION

In this paper we presented a new intrusion detection system technique for detecting input validation attacks. Previously known intrusion detection systems are network intrusion detection systems and web application intrusion detection systems are not efficient with more false positive alarms and more time and space complexity.

In Our proposed approach we have combined two techniques to identify the intrusions. This is an efficient one compared to available and previous intrusion detection systems. In the first level we have used a honey pot system and in the second level we have used an efficient intrusion detection system can detect known attacks very efficiently. This can be done by using Hirshberg algorithm it uses, divide and conquer methodology in order to reduce space and time complexity. By using this technique we can reduce false positives alarms and improve the efficiency of intrusion detection system. It can detect input validation attacks and prevent input validation attacks efficiently against web applications. This application can be widely used to protect web applications from the input validation attacks.

REFERENCES

- [1] OWASPD – Open Web Application Security Project. “Top ten most critical web application vulnerabilities”, http://www.owasp.org/index.php/OWASP_Top_Ten_Project, 2007.
- [2] D. Aucsmith. “Creating and maintaining software that resists malicious attack”, Distinguished Lecture Series, Atlanta, GA. September 2004.
- [3] CERT Coordination Center. “Overview of attack trends”. Technical report CERT CC, 2002.
- [4] Curphey, M., Endler, D., Hau, W., Taylor, S., Smith, T., Russell, A., McKenna, G., Parke, R., McLaughlin, K., Tranter, N., Klien, A., Groves, D., By-Gad, I., Huseby, S., Eizner, M., McNamara, R. “A Guide to Building Secure Web Applications.” The Open Web Application Security Project, v.1.1.1, Sep 2002.
- [5] JaeChul Park, “Anomaly Detection and Analysis Method of Web Application using Sequence Alignment”, PhD.diss., Chonnam Natio Univ 2008
- [6] Fredrik Valeur, Giovanni Vigna, Christopher Kruegel, Engin Kirda. “An anomaly-driven Reverse Proxy for Web Applications”, In Proceedings of the SAC '06 Dijon, France, ACM Press, April. 2006.
- [7] Christopher Krügel, Thomas Toth, Engin Kirda. “Service specific anomaly detection for network intrusion detection”, In Proceedings of the ACM symposium on Applied computing, ACM Press, 2002.
- [8] Needleman, S. B., Wunsch, C. D. “A general method applicable to the search for similarities in the amino acid sequence of two proteins”, J. Mol. Biol. 48:443-453, 1970.
- [9] Jae Chul Park and BongNam Noh. “Detection of Parameter Manipulation using Global Sequence Alignment”. International Conference on Next Generation Web Services Practices, IEEE Computer Society, September 2006.
- [10] Scambray Joel, Shema Mike, Wong David. “Hacking Exposed Web Applications”, SAC'06, Osborne Media, 2002.
- [11] R.Ezumalai, G.Aghila “Combinational Approach for Preventing SQL Injection Attacks” 2009 IEEE International Advance Computing Conference, Patiala, India.
- [13] O.Maor and A.Shulman “SQL Injection Signature Evasion”, White paper Imperva, Apr 2004.
- [14] V.Benjamin Livshits and Monica S.Lam, “Finding Security Vulnerabilities in Java Applications with Static Analysis”.

- [15] M. Roesch. Snort –“ Lightweight Intrusion Detection for Networks. In Proceedigs of the USENIX LISA’99 Conference,”November 1999.
- [16] V.Benjamin Livshits and Monica S.Lam, “Finding Security Vulnerabilities in Java Applications with Static Analysis”
- [17] Z.Su and G.Wassermann “The Essence of Command Injection Attacks in Web Applications”. 33rd ACM SIGPLAN,SIGACT posium on Principles of Pogramming Languages, Charleston, South Carolina,USA,2006,pp.372-382.