International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 5, No.5 (Sep. 2016)
http://dx.doi.org/10.12785/ijcds/050507

Enhance MOODLE Security Against XSS Vulnerabilities

Tawfiq S. Barhoom[1] and Rola J. Azaiza[2]

[1]*Information Technology, Isalmic University, Gaza, Palestine*
[2] *Information Technology, Isalmic University, Gaza, Palestine*

Received 11 Apr. 2016, Revised 7 Jun. 2016, Accepted 7 Jul. 2016, Published 1 Sep. 2016

Abstract: MOODLE (Modular Object-oriented Dynamic Learning Environment) is one of the most popular e-learning environment in the world, MOODLE is same as web application that vulnerable to illegal attacks so, the need for confidentiality, Integrity and availability in e-learning is extremely complex problem to meet the security requirements. One of the serious attacks to the MOODLE is cross site Scripting (XSS). XSS is a web application vulnerability that occurs whenever a web application takes data from user without proper encoding or validation and sends it to the browser. XSS allow attacker to executes scripts that can hijack victims session and deface web sites. MOODLE resources (file, page and student's assignment) are still vulnerable to XSS attacks. For this we need to secure the MOODLE against XSS attacks to keep both teachers and students accounts secure. A lot of researches have handled XSS attacks in CMS but most of these researches have a little attention on XSS attacks on MOODLE. So, we discussed some of PHP's functions that used to prevent XSS attacks. Additionally we conducted a comparative study between four published XSS filters to determine their weaknesses, then RT_XSS_Cln filter was developed to prevent XSS attacks and overcome the other filters weaknesses. RT_XSS_Cln filter is written using PHP language its evaluated by performing offline and online testing. Offline testing is done by nearly 80 files contain nearly 1000 malicious scripts, while online testing is done by plugging RT_XSS_Cln on the MOODLE from both sides teacher's side and students' side to protect both of them. RT_XSS_Cln filter catch all the tested malicious scripts also RT_XSS_Cln filter is faster than the other filters it has a little processing mean time than the others nearly 0.002s.

Keywords: *Filter, MOODLE, XSS, Malicious files, web applications, www, e-learning*

1. INTRODUCTION

E-learning is a method of learning using Internet usually e-learning is understood as online courses or online education. E-learning is systems that allow the activity monitoring participants, simulations, work on subgroups, audio and video interaction[1].
MOODLE become one of the most common environment for online learning, it has the ability to tracking the leaner's progress which is monitored by teachers[2]. MOODLE was developed by Martin Dougiamas in 2002 to help learners to interact with their teachers easily, it permits teachers to present and locate documents assignments, quizzes with students in an easy learning way, it's open source software and can be configured to run in various operating systems.

MOODLE is widely used among world's universities, colleges, schools and institutes by (Jan 2016) there are 64,962 registered sites all over the world nearly in 225 countries with 81,426,088 users[3].

MOODLE is exposed for a lot of attacks one of the serious attacks to internet is Cross Site Scripting (XSS). XSS is considered as the most direct harm to user privacy and spreading viruses. XSS is a web application vulnerability that caused by failure in checking up on user input before returning it to client web browsers. User's input may include malicious scripting code that may be sent to other clients and unexpectedly executed by their browsers thus causing a security exploit[4].

A. Types of Cross Site Scripting

1) Persistent XSS attacks "Stored Cross-site scripting"

Persistent XSS occurs when the attacker provides malicious data to the web application and stored permanently on a database or some other similar storage.

E-mail address: tbarhoom@iugaza.edu.ps, rola_azy@hotmail.com

The malicious data is later accessed and executed by the victims without being filtered or sanitized[5].

2) Non-persistent XSS attacks "Reflected XSS"

It is the common type of XSS attacks which the injected code is sent back to the visitor of the server such as in an error message, search result, that includes some or all of the input sent to the server as part of the request[5].

3) DOM-based attack

DOM-based attack is based on the Document Object Model (DOM) of the page. DOM-based attack happen if the JavaScript in the page accesses URL parameters and use URL's information to write HTML to the page [6].

B. MOODLE Vulnerabilities

MOODLE is same as web application that exposed to a lot of security attacks like SQL injection[21], Stack smashing attacks[22], Virus/Trojan injection, Cross Site Request Forgery[8], Password cracking[2] and Cross Site Scripting (XSS) which is a type of computer security vulnerability typically found in Web applications. XSS enables attackers to inject scripts into web pages viewed by other users this is allow the attacker to hijack user's sessions easily[8]. Unfortunately the injected JavaScript code is difficult to distinguish from the legitimate code at the client side [23].

To avoid XSS attacks:

1) Server-side

It can be done by sanitizing user inputs before it being stored on the web server.

2) Client-side

Clients may use secure browsers with XSS filter and keep it up to date against XSS scripts or clients may disable JavaScript in his browsers but this solution seems non adequate since most of web pages need JavaScript to display its contents.

Our contribution represented in a tool called RT_XSS_Cln filter that work on server side and has the ability to detect and prevent XSS attacks and overcome the selected filters weaknesses. RT_XSS_Cln has been plugged into the MOODLE server to enhance its security against XSS attacks.

This paper organized as follows. Section 2 discuss some of related works. Section 3 handle four published XSS filters. Section 4 explain our methodology. Section 5 proposed our filter and finally section 6 show the implementation setup and discussion.

2. RELATED WORKS

MOODLE is vulnerable to a lot of attacks one of the most crucial attack is cross site scripting [8]. we divided the related works into two parts:

A. security Issues in CMS (Content Management System) like Joomla, Wordpress and MOODLE.

B. Defenses techniques against cross site Scripting.

Part1: security Issues in CMS like Joomla, Wordpress and MOODLE

Hernández, J.C.G et al.[7] proposed an object oriented model of MOODLE using Unified Model Language (UML) which is represented into three models: analysis, design and components. Then they discussed some of security vulnerabilities and its solutions in MOODLE such as session hijacking, session fixation, prediction of username and password. Their solutions to the proposed vulnerabilities depend on modifying certain portions of code and adding new functions.

The represented research provided some of MOODLE's vulnerabilities with recommended solutions which may help MOODLE's users to protect MOODLE against the previous vulnerabilities but they didn't handle cross side scripting vulnerability in MOODLE and how to protect MOODLE against such attacks.

Costinela-Luminita, C.D. and C.I. Nicoleta-Magdalena[8] proposed some of MOODLE vulnerabilities such as cross site scripting, cross site request forgery, SQL injection, stack smashing attacks and session hijacking also they proposed some of considerations to avoid these vulnerabilities. The represented research considered as a defenses guidance to MOODLE's users for some of attacks but they didn't specify which MOODLE resources are vulnerable to XSS attacks.

Colton Floyd et al.[9] presented some of vulnerabilities on MOODLE (v. 1.9. v. 2.1) that can be exploited by students. These vulnerabilities like session hijacking, XSS which is appear on the external URL in the administrator accounts, session management flaws. Also they proposed a recommendation to overcome these vulnerabilities to protect both teachers and students. The represented research proposed a useful recommendations to overcome some vulnerabilities but in case of external URL vulnerability they didn't provide any defense technique or code patches to overcome this attacks. The vulnerable URL is already avoided in next versions of MOODLE but unfortunately MOODLE resources are still suffer from XSS attacks.

Patel, S.K et al.[10] conducted a comparative study of CMS security, they choose the most popular CMS Joomla, Drupal and Wordpress and perform two tested cases to discover their security:

Case1: By developing one common page in all the proposed CMS, hosting these pages and then applying different attacks such as SQL Injection, cross site scripting XSS, file inclusion function and remote file inclusion.

Case2: Using Acunteix reporter v.6.0 to find out the strength of security in different CMS.

Result1: they found that it's not easy to hack CMS's sites because of their community provide a basic security for CMS's pages.

Result2: they found that they can got the cookie information of some sensitive files which is not directly linked to the website which can able attackers to hack the website easily, also they found that Wordpress has the less number of sensitive files and directories that make it the stronger security ones.

This represented research is good but it still ambiguous due to case1's result because they didn't provide the implementation ways of attacks. They only said that CMS's pages can be hacked from CMS's plug-ins, but there are a lot of researches approved that most of CMS have a lot security issues.

Meike, M. et al.[11] proposed some of security vulnerabilities in open source web content management they choose Joomla and Drupal as a case study, they found that both Joomla and Drupal seems adequately prepared to prevent XSS attacks and SQL injection also they found that both Joomla and Drupal have secured login mechanism and session data this is because their communities were dedicated to fulfill security requirements like security patches, vulnerabilities reporting and tips but they found that both Joomla and Drupal contain password security weaknesses.

Arakelyan, A.[12] proposed some of security vulnerabilities problems in MOODLE these problems were classified into four groups: authentication, availability, confidentiality and integrity. Also he proposed solutions to the previous attacks by modifying certain portions of the code and adding new functions.

Kumar, S. and K. Dutta.[13] proposed some of security attacks on MOODLE such as session attacks, design attack and user logout, session not closed. Design attacks involve password prediction, username prediction and session hijacking. They suggested to use secure socket layer (SSL) to overcome session attack and design attacks. SSL establish an encrypted link between web server and browsers. Also they suggested to use CAPATCHA technique to avoid brute force in login page which generate random values that allow user to enter these random values during his login.

The latest two researchers suggested some of recommendations to avoid the previous attack but it didn't provide any details about the cross site scripting attacks on MOODLE.

Tawfiq Barhoom and Hijazi, M.I [14] proposed a guidance for matures to prevent XSS attacks in open CMS, they analyzed some of websites created on Joomla and Wordpress as CMS using some of scanning tools to extract the security issues especially XSS attacks. Due to the lack of details from scanning tools they injected manually different ten cases of malicious XSS codes in both Joomla and Wordpress pages to get more details of XSS attacks then they proposed defense way for each of attack case. The attacks and defense have been learned by matures to secure their websites.

Their work is useful and helpful especially for ones who try to secure their websites from XSS attacks. Her guidance is simple and easy to understand from matures but they only focused on Joomla and Wordpress as type of CMS.

Part 2: Defenses Techniques Against Cross Site Scripting

SHahriar, H. and M. Zulkernine.[15] developed an automatic framework that able to detect XSS attacks at server side by inserting boundaries e.g.: HTML comment (<!---!>), JavaScript comment (/*….*/) or token (- -t1- -) which uniquely identify legitimate script of dynamic contents then policies for JSP programs are generated according to the inserting boundaries. Their approach was success in detecting advanced XSS attacks where many of existing approaches have been failed without any modification of server or client side. The different between this research and the our proposed solution is that their work required a lot of policies checks in addition to they implemented their approach using JSP while our proposed model is written in PHP.

Shanmugam, J. and M. Ponnavaikko.[16] proposed solution in JSP/Servlet able to prevent XSS attacks, their solution consist of four components analyzer which check the input if it exceed the maximum number, if it; the input will be rejected also it check the input if it contain special characters, parser which break the input into multiple tokens to be passed to verifier, verifier check the input for its vulnerabilities by executing the rules using tag cluster, tag cluster is defined by author to determine whether the input provided is malicious or not.

Their approach is quite simple and understandable but the difference between their solution and our proposed model is that their solution it is implemented in JSP/Servlet while our model is written in PHP , also their solution require tag clusters which is defined by

author and need updating when new tag needed to be permitted while our model didn't.

Wurzinger, P. et al. [17] introduced SWAP (Secure Web Application Proxy) which is able to detect and prevent XSS attacks, SWAP operates on a reverse proxy installed in front of web server which relay all traffic between clients and web server and intercepts all HTML responses from server and subject them to analysis by JavaScript detection component. Their solution is utilize the reverse proxy for mitigation of XSS attacks also their solution didn't require any modification on client side but SWAP might not be suitable for high performance web service. Their solution is different from our proposed solution because they didn't handle MOODLE as target, while our model is focus on it and working to increase its security.

Di Lucca, G.A Et al.[4] proposed an approach to detect XSS vulnerabilities, their approach exploit both static and dynamic analysis of source code, static analysis determine whether the server web page is vulnerable to XSS while dynamic analysis is exploited to verify whether WA with vulnerable server is actually vulnerable.

Their work achieved good results in detecting XSS malicious code in many of open source web applications.

Shar, L.K. and H.B.K. Tan.[18] classified the XSS defenses techniques into four types: defensive coding practices, XSS testing, vulnerability detection and runtime attack prevention.

Mewara, B. et al.[19] proposed a comparative study between three browsers add-ones Internet Explorer11 (XSS filter), Google Chrome32 (XSS Auditor) and Mozilla Firefox 27(XSS-Me) against reflected XSS attacks by injected XSS malicious codes in POST parameters in Input, iframe, Hyperlink and different events they found that every browser's defenses add-ones has it's own limitation and cannot defend all cases but and Mozilla Firefox 27(XSS-Me) seems the better one in defending against XSS attacks, the difference between their research and our research is that their research is propose a comparative study between add-ones (XSS filters) of the browsers while our research perform a comparative study between four public XSS filters and propose an XSS filter.

Engin .K et al. [20] proposed client-side solution to mitigate cross side scripting attacks tool called Noxes which acts like proxy that allow user manually and automatically generated rules to block cross site scripting attacks. It detects XSS attacks from many perspectives e.g. Referrer Header, Request type "GET, POST", java Script code "pop-up window, frames, self-location" this is make it more stronger against XSS attack. But this tool is implement against stored and reflected XSS while DOM is not considered. The different between this research and the proposed solution is that our solution is a PHP filter that plugged on the MOODLE server.

Tawfiq Barhoom and Hamada, M.H.A [5] proposed XSSDetection tool able to detect XSS attacks in the client side. XSSDetection tool can be used in forums that takes user input as target to detect XSS attacks by inject malicious Java script code. XSSDetection is written in python language. The different between this research and the proposed solution is that our proposed model is able to detect XSS attacks in the server side and its written in PHP language.

3. XSS FILTERS

A lot of XSS filters have been published over the internet to able websites developers to protect their websites from XSS attacks. We selected four XSS filters and tested them offline by group of maliciously files. Unfortunately the selected filters have a lot of weaknesses. Table 1 summarized the selected filters weaknesses.

A. *XSS-Clean filter*[1]**:** is written in PHP by group of developers, it has the ability to detect a lot of XSS attacks, it tested against most exploits founded in http://ha.ckers.org/xss.html, XSS_Clean is coded using preg_replace() function.

B. *RemoveXSS filter*[2]: It's a PHP XSS filter, its considered as a good filter which able to detect most of XSS attacks but unfortunately RemoveXSS failed in testing some of XSS scripts.

C. *XSS-Master filter*[3]: It's a PHP filter which remove dangerous tags and protocols from HTML, it use preg_replace() and preg_match() functions in its coding. XSS-Master is so complicated due to its nested function with 300 lines of code.

D. *XSS_Protect filter*[4]**:** it's a PHP functions it use strip_tags() and htmlentities() functions to catch XSS vulnerabilities but the output is the same as input but fully escaped and encoded except of some limitations. XSS_Protect filter can be hacked using the allowed tags $data = strip_tags($data, $allowed_tags . "<b>").

[1]Published in https://gist.github.com/mbijon/1098477

[2]Published In https://gist.github.com/ozkanozcan/3378054

[3]Published in https:// github.com/ymakux/xss

[4]Published inhttp://www.jstiles.com/blog/

Table 1. The selected filter's weaknesses

Filters Weakness
1) Potential scripts
2) Allowed tags
3) Complexity
4) Processing time
5) Difficult to understand
6) Delete form feilds
7) Detect HTML5 entity char attacks
8) Malicious Strong attacks
a./'document.cookie'/ on the page body
b.Detect clicktt
c.Detect click
d.Detect "><h1/onmouseover='\u0061lert(XSS)'> %00
e.Detect</script><img/*%00/src="worksinchrome:prompt(1)"/%00*/onerror='eval(src)'>
f. DetectClick Here
g.><div/onmouseover='alert(1)'> style="x:">
h.Detect6
i. Detect 7

Moodle is still vulnerable to XSS attacks which may threat both teacher and students accounts. Our objective aim to enhance MOODLE security against XSS by developing XSS tool able to detect and prevent XSS attacks.

4. METHODOLOGY

The underlying attack and defense scenario will focus on teacher and student as MOODLE's users because both of them are potential victims to each other. Teacher may inject the course's assignment with malicious XSS script, and when the students viewed the uploaded assignment then the malicious scripts will activated on the student's side. In the same manner also teacher may become a victim to the student if the student inject his assignment with bad XSS script and uploaded it to the MOODLE, teacher going to assess the uploaded assignments then the malicious script will activated on the teacher side. We are going to test most of MOODLE resources against XSS vulnerabilities from both sides teacher side and student side. Then we are going to secure the vulnerable resources by developing XSS filter called RT_XSS_Cln. RT_XSS_Cln filter will be plugged on the weak MOODLE resources from both sides teacher and student to protect both of them from XSS attacks. The proposed defenses model has many stages as shown in figure 1 Login stage: teacher logged to the MOODLE by his user name and password. Filling Stage: teacher may fill the description field of Page, file or assignment with malicious script or upload malicious content like page or file to the MOODLE, Sanitizing Stage: This stage is divided into two parts: fields filtering part and content filtering part finally storing Stage: store the cleaned content into MOODLE database.

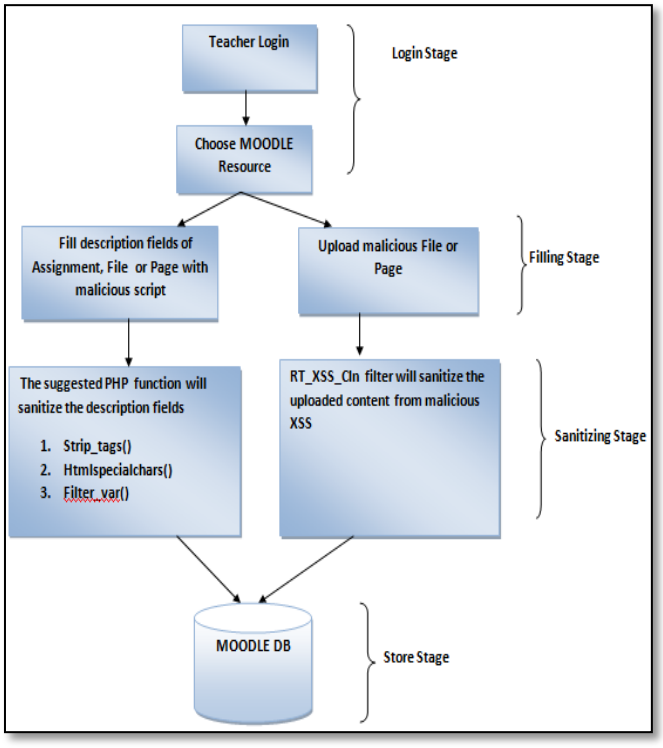

Figure 1. The proposed defenses model

In this section we discuss vulnerabilities of The MOODLE from both accounts teacher and student accounts. also we will discuss three PHP functions that able to prevent XSS attacks additionally we will develop RT_XSS_Cln filter that able to prevent XSS attacks.

A. MOODLE XSS Vulnerabilities

We will inject most of MOODLE resources with XSS scripts e.g. a Page, Assignment, File, Glossary, Chat room , External URL to determine whether these resources are vulnerable to XSS attacks or not.

B. PHP Functions

We will discuss three widely used PHP functions that able to sanitize fields from XSS attacks, these functions are strip_tags() , Htmlspecialchars() and Filter_Var().

These functions will be tested offline by group of XSS scripts then we will plug these functions into MOODLE resources to discover its effectiveness in catching XSS scripts.

C. Comparative study between the Selected XSS filters

We will conducted a comparative study between four published XSS filters to determine their weaknesses.

D. RT_XSS_Cln filter

We will develop RT_XSS_Cln filter that able to overcomes the selected filters weaknesses. We named it RT_XSS_Cln where R refer to my name Rola, T refer to Tawfiq the name of my supervisor, XSS type of studied attacks and Cln is refer to clean. RT_XSS_Cln will be able to detect and prevent XSS scripts on the collected scripts. RT_XSS_Cln will be written in PHP language to be plugged on the MOODLE easily, also RT_XSS_Cln will be tested offline by group of malicious scripts and online by plugging it into the MOODLE.

5. THE PROPOSED FILTER

In this section we implemented our methodology to achieve the desired objectives. We explored the weak resources in MOODLE (v 2.8.1) also we discussed some of PHP5 functions that used to prevent XSS scripts and deduced the better functions additionally we developed RT_XSS_Cln XSS filter that prevent XSS attacks and overcomes the other filters weaknesses.

A. Explore The XSS Vulnerabilities In The MOODLE From Teacher And Students Sides

From teacher's account most of MOODLE resources have been injected with XSS scripts, we discovered the following: some resources prevent the injected XSS like Glossary, Chat room , External URL while others are still vulnerable to XSS attacks such as:

1) *Page*:
- Page Description is vulnerable to XSS attack.
- Page Content is vulnerable to XSS

2) *File*
- File Description
- File Content

3) *Assignment*
- Assignment description

From student's account we discovered that the uploaded assignment is vulnerable to XSS attacks.

B. Discuss And Plug PHP functions Which Able To Prevent XSS Script.

We selected three PHP functions that able to sanitize string from malicious code e.g. strip_tags() , Htmlspecialchars() and Filter_Var(). These functions are tested offline by nearly 1000 XSS scripts figure 2 shows sample of scripts. Then each one of the selected functions is plugged on the MOODLE vulnerable resource Page, File and Assignment respectively to discover its effectiveness in catching XSS scripts.

```
HTML5 entity char <a
href="javas&Tab;cri&NewLine;pt:alert(document.cook
ie)">test</a>

Input[hidden] XSS <input type=hidden
style='x:expression(alert(/ @garethheyes /))'> target it .


>"imgsrc=x:xonerror='alert(/ @jackmasa ['"/(/

document.body.innerHTML=('<\000\0i\000mg
src=xx:xonerror=alert(1)>')

header('Refresh: 0;url=javascript:alert(1 ;('(

>script
language=vbs></script><imgsrc=xx:xonerror="::aler
t' @insertScript<"::'

<a href="data:text/html,<script>eval(name)</script>"
target="alert(' @garethheyes @0x6D6172696F
')">click</a<

<script/onload=alert(1)></script<

/>noscript><imgsrc=xx:xonerror=alert(1<-- (

>a
href="javascript&colon;alert&lpar;1&rpar;">clicktt</
a<

>a href="feed:javascript&colon;alert(1)">click</a>
Firefox

>link href="javascript:alert(1)" rel="next"> Opera,
pressing the spacebar execute! by @shafigullin

>embed
code="http://businessinfo.co.uk/labs/xss/xss.swf"
allowscriptaccess=always> works on webkit by
@garethheyes

>script /*%00*/>/*%00*/alert(14)/*%00*/</script
/*%00/*

>;62#&;34#&h1/onmouseover='\u0061lert(15)'>%00
```

Figure 2. Malcious XSS scripts

Preventing XSS attacks in MOODLE Page

1. Go to ***MOODLE/mod/page/lib.php*** directory.
2. Plug Htmlspecialchars(), strip_tags() or Filter_Var() in ***page_get_coursemodule_info*** function.

Preventing XSS attacks in MOODLE file

1. Go to ***mod/resource/locallib.php*** directory.
2. Plug Htmlspecialchars() , strip_tags() or Filter_Var() in ***resource_print_intro*** function.

Preventing XSS attacks in MOODLE Assignment

A. Go *to* ***mod/resource/locallib.php*** directory.
B. Plug Htmlspecialchars() , strip_tags() or Filter_Var() in ***resource_print_intro*** function.

It recommended not to use strip_tags() function due to its weakness. strip_tags() function support the allowed tags which can be exploited by attackers to attack users websites as shown if figure 3.

```
strip_tags(string,<b>);
<b
onmouseover="s=document.createElement('script');s.src='http://pastebin.com/raw.php?i=j1Vhq2aJ';document.getElementsByTagName('head')[0].appendChild(s)">hello</b>
```

Figure 3. Strip_tag() gap

Additionally strip_tags() break user input and remove the content that the user not expect e.g. (Happy Day *<:) or a puckered face.\n) will be Happy Day*. Htmlspecialchars() and FILTER_VAR are more preferable than strip_tags( ) because it cannot be hacked and keep the string as it with a minimum change.

C. *Comparative study between the selected XSS filters.*

Each filter of the selected XSS filters is tested offline by nearly 1000 scripts to measure its weaknesses and limitation. Unfortunately the selected filters have a lot of weaknesses as shown in table 1.

D. *Develop RT_XSS_Cln filter able to prevent XSS attacks.*

RT_XSS_Cln filter was developed to prevent XSS attacks and overcomes that other filters weaknesses. RT_XSS_Cln filter is PHP filter able to provide a high protection against XSS attacks comparing with the other filters additionally it has processing mean time better than the other filters, RT_XSS_Cln filter has five functions.

1) RT_XSS_Cln Main Function

This is the main function that call the other functions to complete the filtering process. RT_XSS_Cln function decode the content with html decoding function $content = html_entity_decode($content, ENT_COMPAT, 'UTF-8');The first argument is the string that will be decoded. The second argument tells the function how to treat quotes. Use ENT_COMPAT which will convert double quotes and leave single quotes, The third argument selects the character set to decode into.

2) Small_Case Function:

Change the letters case to small cases e.g. "SCRIPT", "script" or" ScRiPt" all become "script".

3) Replacement Function:

Which perform a series of replacement on the content to eliminate the malicious script

1. Replace the character entity name e.g. &lt, << with $1;.
2. Replace (&#) with $1; e.g. &#0060,&#060 that character code for "<" .

4) Replacement_Event Function

Replace the html events, because events can be potential for attacks. Event replacing done by replacing "on" so that all events are disables such as onload, onclick, ommouseover.

5) Replacement_MWords Function

Replace some of words that may hold malicious script e.g. JavaScript, script , Iframe, embed, base, cookie, bgsound, layer, data.

RT_XSS_Cln Model

First we check MOODLE users whether a teacher or a students in case of teacher: teacher choose the file or page to upload his tasks. Teacher can upload malicious script to MOODLE via file or page due to its vulnerabilities. Then the enrolled student will download teacher uploaded file or page which contain malicious scripts that will be activated in his side.

In the same manner the student will response to his teacher request and upload his assignment to the MOODLE. Student's assignment may contain XSS scripts that will affect teacher account when he assess the uploaded assignment shown in figure 4.

RT_XSS_Cln embedded in the course's page and in course's file to clean the uploaded contents of both, also RT_XSS_Cln embedded in the assignment so that any uploaded assignment from students is filtered and cleaned from XSS attacks.

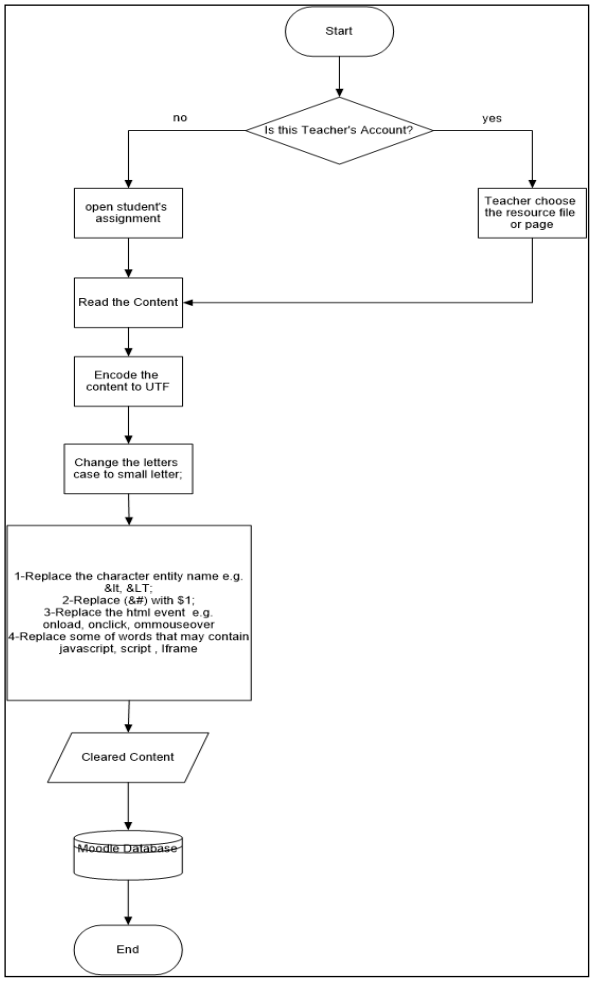

Figure 4. RT_XSS_Cln flowchart

6. DISCUSSION

In this section we evaluated RT_XSS_Cln filter by performing offline and online testing.

A. *Offline Evaluation*

Is done by a group of malicious files that contain nearly 1000 XSS scripts. These scripts are distributed over 80 malicious files. RT_XSS_Cln filter overcomes the other filters weaknesses and catch all the tested XSS scripts.

B. *Online Evaluation*

Is done by plugging RT_XSS_Cln filter in the vulnerable MOODLE resources from both accounts .

1) Teacher's account
2) Student's account

A. ***Teacher's account***

Plugging RT_XSS_Cln in the vulnerable MOODLE File

1. Check the file's type whether is it HTML or not.
2. Plug RT_XSS_Cln filter to the MOODLE resource file directory at ***mod/resource/locallib.php***, in ***resource_display_embed*** function.
3. RT_XSS_Cln will filter the file content before saving the file into MOODLE database.

Plugging RT_XSS_Cln in the vulnerable MOODLE page

It's necessary to filter the page content before being outputted to the students. filtering process of MOODLE page should be done in both processes adding and updating process to ensure that MOODLE page content is fully cleaned from XSS scripts.

Adding*:* To filter the page's content we should

1. Plug RT_XSS_Cln filter in page adding function. Adding page's code is found in ***mod/rpage/lib.php***.
2. Change the statement in the function ***page_add_instance, $data->content =$data->page['text'];***

```
$data->content=RT_XSS_Cln($data->page['text']);
```

Updating:

1. Change the statement in the function ***page_update_instance***, ***$data->content =$data->page['text'];***

```
$data->content=RT_XSS_Cln ($data->page['text']);//update for content of pages
```

B. ***Student's Account***

MOODLE has its own mechanism in storing its files on database, it encrypt both filename and directory so it difficult to be guessed e.g. if the uploaded file called Coll20-xss.htm, its name encrypted to become ***1caba34cc1a8ec640165559eb55cde6286037934*** where the first two digits is the name of the external folder (1c) and the second two digits is the insider folder where Coll20-xss.htm file is stored (ab). uploaded files are stored on the server not on the database but file information like name, directory are saved on database. Uploaded file directory is ***C:\wamp\MOODLEdata/filedir/t1/t2/filename*** where ***t1*** is the first two digits from hashed file's name and ***t2*** is the next two digits, e.g. the Coll20-xss.htm directory is ***C:\wamp\MOODLEdata/filedir/1c/ab/1caba34cc1a8ec640165559eb55cde6286037934***

1) Filtering Student's Assignments

To filter student's uploaded file we should perform the following:

1. Plug RT_XSS_Cln on the root directory of the MOODLE ***/mod/assign/submission/file/locallib.php***
2. Modify ***view_summary*** function to be able to read the file content and cleaned by RT_XSS_Cln filter**.**
3. Go to this directory ***www/MOODLE/lib/filestorage/stored_file.php***
4. Update the ***get_pathname_by_contenthash()*** function by declaring server variable that contain file directory of uploaded file.
5. Go to ***www/wamp/MOODLEdata/filedir/*** where the uploaded files are stored.
6. Open the ***1c*** folder, open ***ab*** folder you will find the uploaded file Coll20-xss.htm.

Plugging RT_XSS_Cln filter on the MOODLE will ensure that any uploaded html file from students are cleaned from XSS scripts thus we enhance the MOODLE security against XSS vulnerabilities and provide a good protection for both teacher and students.

7. CONCLUSION

MOODLE resources page, file and assignment are vulnerable to XSS attacks. Both teacher and student can be potential victims for each other's, We discussed some of PHP functions strip_tags() or filter_var(), Htmlspecialchars() that able to prevent XSS attacks. Additionally we conducted a comparative study between four published filter to determine their weaknesses in preventing XSS attacks. We developed RT_XSS_Cln filter that overcome the other filter's drawbacks. RT_XSS_Cln filter is written in PHP function, it has a complete ability to prevent XSS attacks unlike the other filters. RT_XSS_Cln is easy to understand and extensible so it easy to insert additional functionality. RT_XSS_Cln filter has a mean time equal to 0.0024s in processing group of tested files which is less than the other filter.

Offline and online evaluation are done on RT_XSS_Cln filter. Offline evaluation was done by group of malicious files, RT_XSS_Cln cover all XSS cases without any bugs mentioned. Online evaluation is done by plugging RT_XSS_Cln in the MOODLE in the vulnerable resources file, page and assignment. Online evaluation was performed from both accounts teacher's account and student's account to ensure that there is no attacks occurs.

Acknowledgment

My Great thanks to Allah the Most Merciful, the lord of the world for his help and guidance to finish my research, and the great thanks to our messenger Mohammad .

Firstly I would to express my sincere gratitude to may advisor Dr. Tawfiq S.M. Barhoom, Associated Professor of Information Technology in the Islamic University in Gaza for his continuous support, patience, motivation and immense knowledge. We would like to thank my family my mothers, brothers, sisters and my husband for their love and support during my study, they have always encouraged me towards excellence.

Big thanks for my husband family for their supporting and encouraging me for the better

REFERENCES

[1] Costinela-Luminiţa, C.D. and C.I. Nicoleta-Magdalena, "E-learning security vulnerabilities "Procedia-Social and Behavioral Sciences. 46: p. 2297-230.

[2] "The Top 8 Open Source Learning Management Systems", http://elearningindustry.com/top-open-source-learning-management-systems, [Accessed on: 16-02-2016].

[3] "MOODLE Statistics", https://MOODLE.net/stats/, [Accessed on:27-01-2016].

[4] Di Lucca, G.A., et al. "Identifying cross site scripting vulnerabilities in web applications", Telecommunications Energy Conference, INTELEC, 26th Annual International, 2004

[5] Tawfiq Barhoom and Hamada, M.H.A., "PALXSS: Client Side Secure Tool to Detect XSS Attacks", Saba Journal Of Information Technology and Networking", Vol 2, 2014.

[6] Kirda, E., et al., "Client-side cross-site scripting protection", computers & security, Vol 28, Issue 7, PP: 592–604,October 2009

[7] Hernandez, J.C.G. and M.A.L.n. Chávez, "MOODLE security vulnerabilities", Electrical engineering, computing science and automatic control, 5th international conference, IEEE, 2008.

[8] Costinela-Luminiţa, C.D. and C.I. Nicoleta-Magdalena, "E-learning security vulnerabilities "Procedia-Social and Behavioral Sciences. 46: p. 2297-2301.

[9] Floyd, Colton, Tyler Schultz, and Steven Fulton, " Security Vulnerabilities in the open source Moodle eLearning system.", Proceedings of the 16th Colloquium for Information Systems Security Education. 2012.

[10] Patel, S.K., V. Rathod, and J.B. Prajapati, "Comparative analysis of web security in open source content management system". Intelligent Systems and Signal Processing (ISSP), International Conference on: IEEE, 2013

[11] Meike, M., J. Sametinger, and A. Wiesauer, "security in Open source Web Content management systems", IEEE Computer Society, Vol 7,Issue 4, PP: 44-51, July/August 2009.

[12] Arakelyan, A., Vulnerable Security Problems in Learning Management System (LMS) MOODLE. Institute for Informatics and Automation Problems of NAS of RA.

[13] Kumar, S. and K. Dutta, "Investigation on security in LMS MOODLE", International Journal of Information Technology and Knowledge Management, Vol 4, Issue 1, PP: 233-238, 2011.

[14] Tawfiq Barhoom and Hijazi, M.I., "Exploring Guidance for prevent against XSS attacks in open CMS", Palestine Technical College Scientific journal: Gaza, Vol 2, 2016.

[15] Shahriar, H. and M. Zulkernine,"S2XS2: A Server Side Approach to Automatically Detect XSS Attacks". Dependable, Autonomic and Secure Computing (DASC), Ninth International Conference on IEEE, 2011.

[16] Shanmugam, J. and M. Ponnavaikko,"Behavior-based anomaly detection on the server side to reduce the effectiveness of Cross Site Scripting vulnerabilities", Semantics, Knowledge and Grid, Third International Conference on IEEE, 2007.

[17] Wurzinger, P., et al. SWAP," Mitigating XSS attacks using a reverse proxy", Proceedings of ICSE Workshop on Software Engineering for Secure Systems, IEEE Computer Society, 2009.

[18] Shar, L.K. and H.B.K. Tan, "Defending against cross-site scripting attacks", Computer, (3):PP: 55-62.

[19] Mewara, B., S. Bairwa, and J. Gajrani, "Browser's defenses against reflected cross-site scripting attacks", Signal Propagation and Computer Technology (ICSPCT), IEEE, 2014.

[20] Kirda, E., et al., "Client-side cross-site scripting protection", computers & security, Vol 28, Issue 7, PP: 592–604,October 2009.

[21] Halfond, W., J. Viegas, and A. Orso. "A classification of SQL-injection attacks and countermeasures", Proceedings of the IEEE International Symposium on Secure Software Engineering, IEEE, 2006.

[22] Cowan, C., et al. "Protecting systems from stack smashing attacks with StackGuard", in Linux Expo, 1999.

[23] Shahriar, Hossain, and Mohammad Zulkernine. "Injecting comments to detect JavaScript code injection attacks." Computer Software and Applications Conference Workshops (COMPSACW), 2011 IEEE 35th Annual. IEEE, 2011.

Tawfiq S. Barhoom Head of Computer Science and Software Development Departments Faculty of IT, Islamic University-Gaza, he got B.Sc. Computer Science from Omdurman Ahlia University-Sudan,(1991-1995) and Master degree, and he has – M.Sc. Computer science, Department of computer science and engineering from Shang hai Jiao Tong University (SJTU)– ShangHai – China, (1996- 1999) and his has – PhD in Applied computer Technologies, Department of computer science and engineering from ShangHai Jiao Tong University (SJTU) – Shanghai – China, (2001- 2004).

Rola J. Azaiza Previous Instructor at Palestine Technical College, Master student at Islamic university , Palestine she got a BS.c from Palestine Technical College, palestine