Multiclass Classification of XSS Web Page Attack using Machine Learning Techniques

S.Krishnaveni
Research scholar
PSGR Krishnammal college for women
Coimbatore

K.Sathiyakumari
Assistant Professor
PSGR Krishnammal college for women
Coimbatore

ABSTRACT
Web applications are most widely used technique for providing an access to online services. At the same time web applications are easiest way for vulnerable acts. When a security mechanism is failed then the user may download malicious code from a trusted web site. In this case, the malicious script is contracted to full access with all assets belonging to that legitimate web site. These types of attacks are called Cross-Site Scripting (XSS) attacks.

Cross Site Scripting (XSS) attacks are the most common type of attack against web application, which allows hackers to inject the malicious script code for stealing the user's confidential information. Recent studies show that malicious code detection has become the most frequent vulnerability. In web browsers, the malicious script codes are executed and used to transfer the sensitive data to the third party (or hackers) domain. Currently, most research areas are attempted to prevent XSS on both the client and server side. In this paper, we present a machine learning technique to classify the malicious web pages. This work focus some of the possible ways to detect the XSS script on client side based on the features extracted from the web document content and the URL to scan the web pages for check the malicious scripts.

General Terms
Naive bayes, Decision Tree, Multilayer perceptron, Security, Machine learning.

Keywords
Cross site scripting (XSS), SOL injection, Script injection attact, Cross site Request Forgery (CSRF), vulnerability, malicious code.

1. INTRODUCTION
The dynamic web pages are plays an important role for providing an advanced greater interactivity in web based services such as e-commerce, internet banking, on-line shopping, social networking, blog, forums and etc. According to Wassermann su[26], most of the web programming languages does not provide safe data transfer by default so it leads to most frequent attacks in web application like Cross Site Scripting (XSS), SOL injection, Cross Site Request Forgery (CSRF), Redirect and etc.

Recent years, the researchers found that XSS vulnerabilities are in the top lists of greatest vulnerability. There are many approaches and techniques are proposed for detecting the XSS vulnerable web pages. However, the machine learning techniques are the effective way to detect the web anomaly.

In this paper we present a possible ways to identifying the set of features for detecting the XSS web pages. We apply machine-learning techniques such as Naive Bayes, Decision tree and MLP to classify the web pages. Our dataset consists of 200 websites are correspond to the XSS attacks occurred from Sep 2009 to Jan 2013 obtained from XSSed dataset (http://www.xssed.com). We evaluate the classifiers based on two criteria like predictive accuracy and training time.

The remainder of this paper is organized as follows: Section II introductions and the concepts related to XSS web pages. Section III describes the proposed features and XSS scan for web pages. Section IV we present the experimental results and their analysis. Finally, Section V contains the conclusion and feature work.

2. CROSS SITE SCRIPTING
Malicious Code is a new category of threat, which cannot be blocked by anti-virus software alone. It is not like viruses because malicious code is an auto-executable application whereas the virus can cause damage only when the user executes the program.

New programming languages are used to enhance the web page and email services in this case the malicious code can take the form of Java Applets, ActiveX controls, plug-ins, pushed content, scripting languages and etc. The following are the most affected websites which are listed by category, the top-5 most infected websites are:

1. Blogs and Web communications
2. Hosting/Personal hosted sites
3. Business/Economy
4. Shopping
5. Education and Reference.

A basic example of XSS is when a malicious user injects a script in a legitimate shopping site URL that in turn redirects a user to a fake but identical page. Fig1 shows the XSS vulnerability statistics [1] in the year 2011 and 2012.

Grossman[7] defines Cross-Site Scripting (XSS) as an attack vector caused by malicious scripts on the client or server, where data from user input is not properly validated. This allows the theft of confidential information and user sessions, as well as it compromises the client's browser and the running system integrity. XSS vulnerabilities allow an attacker to inject malicious content into web pages served by trusted web servers. Since the malicious content runs with the same privileges as trusted content, the malicious content can steal a victim user's private data or take unauthorized actions on the user's behalf. To prevent XSS vulnerabilities in the web page [3] all the entrusted content must be sanitized. However, accurate sanitization is very challenging task. The server can

sanitize the content but in case if the browser interprets the content and the server is compromised the attackers take advantage of this discrepancy[8,9,10]. The XSS attacks are significantly categorized as follows:

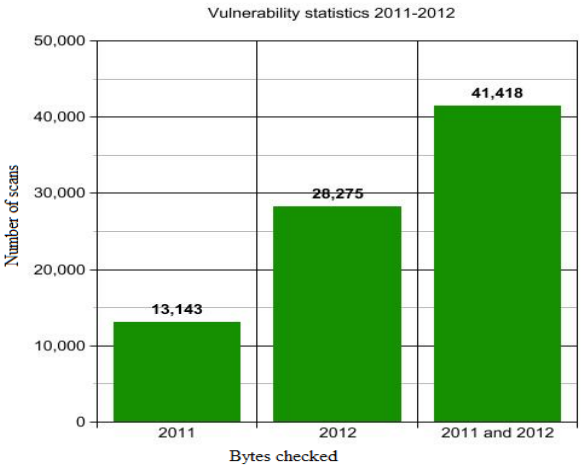

Fig1: XSS vulnerability statistics 2011-2012

a) Reflected: This is probably the most common type of cross-site scripting exploit. The client submits the data to the server, the server in turn process immediately generates the results and it send back to the browser. In this case the browser can interpret the results using HTML special character encoding when it fails the exploitation is successful. The result is displayed as static visible text.

b) Stored: It is a HTML injection attack here user input data containing inject code is sent to the server and is stored in the database and it is used or referenced while creating a webpage so whenever some user visits your webpage this form of cross-site scripting vulnerability is exploited. In this case the malicious script is stored in the server itself so it mostly listed as dangerous attacks.

c) Local: In this type, the malicious code is resides within the web page. When the user opens two or more web pages at the same time the web page with malicious code might alter the Document Object Model (DOM) content of the another page in the local system.

The following example demonstrates the Web application that lets travelers share tips about the places they have visited. The program contains four input fields like Action, Place, Tip and User that attackers can manipulate.

Example URLs that direct Web users to travelingForum. The bold lines describe the malicious scripts that cause XSS exploits. (c) Ordinary URL that activates travelerTip.jsp in "View" action. (d) URL that causes reflected XSS: it contains malicious HTML meta-script capable of making a refresh request to travelingForum's server for every 0.3 seconds, its potentially causing a denial of service. (e) URL that creates stored XSS scenario: it contains JavaScript capable of sending the client's cookie information to a hacker's web-site. (f) URL that causes DOM-based XSS: it contains script capable of injecting misleading information over an original message.

The program can be called via a URL such as the one shown in Figure 2(c). The statement at line 12 in Figure 2(a) is vulnerable to reflected XSS due to the replay of invalid input supplied by users (lines 8, 9, 12). An attacker could send a seemingly innocuous URL link like the one in Figure 2(d) to a victim via e-mail or a social networking site. The script which is highlighted in bold will execute on the victim's browser if the victim follows the link to traveling Forum. The statement at line 18 in Figure 2(a) is vulnerable to stored XSS; in this case the program stores user messages without proper sanitization (lines 8-10) and displays them to visitors (lines 14, 16-18). The URL in Figure 2(e) contains JavaScript capable of sending the client's cookie information to a hacker's website. The statement at line 4 in Figure 1a is vulnerable to DOM-based XSS, the program consists of a JavaScript file, travelerInfo.js, shown in Figure 2(b), that accesses "User" information from the URL (line 24) and displays it, without any sanitization, to users (line 25). Similar to the reflected XSS scenario, an attacker can use the using a crafted URL like that shown in Figure 2(f) to exploit this vulnerability.

3. FEATURE EXTRACTION

3.1 Feature Extraction

Feature extraction plays an important role for identify potential malicious features from different sources of information about web pages. Based on our analysis we identify the following sources of information:

1) *Script based features:* The script contents are used to delivering and hiding malicious codes by obfuscations. We list some the features based on script content which are differ from benign pages such as script size, string size, word size, argument size and etc.

2) *Core contents:* This type of content is targeted on specific vulnerabilities in web browsers, plug-ins and operating system. Mostly these codes are rarely found in direct form because it is encoded with scripts for hide from detection devices. The core content features are applet, objects, embed and etc.

3) *DOM objects:* document.URL, document.URLUnencoded, document.location, document.referrer, window.location and etc.

4) *Some other features:* node split, tag split, ESAPI.ensoder(), encodeForJavaScript(user_input) all JavaScript function calls like (javascript:alert('XSS');), event handlers and etc.

3.2 XSS Scan

XSS web page can be identified based on the features extracted from the given URL of the web site/ web page. First step, it will extract all the URLs (from href, img src, form action and etc). Second step, parse the HTML tags from the source of the web pages and list the tags, which are used in that. Third step, search the features like, <Script>, URL length, DOM objects, function calls and etc. Finally it displays the result. Using this we can trace the particular content on the web pages like a particular variables, functions, tags and etc. And also it displays the JavaScript functions in the web page. The main feature of this is to scan for evil calls and document calls of the given web page.

4. EXPERIMENTS AND RESULTS ANALYSIS

This section describes the data collections, classifiers and other parameters used to conduct the experiments, as well as the demonstrate results obtained using the tool. The open source data mining tool Weka (http://www.cs.waikato .ac.nz/ml/weka) was used to perform the experiments with

```
1<html>
 2 <title>Forum for Traveling Tips</title>
 3 <body>
4 <h1>Welcome <script language="javascript" src="travelerInfo.js">
```

```
</script>!</h1>
<%
5 String action = request.getParameter("Action");
6 String place = request.getParameter("Place");
7 if (place !=null && action.equals("Post")) {
8 String new_tip = request.getParameter("Tip");
9 if(new_tip.length < 100) {
10 stmt.executeUpdate("INSERT INTO forum VALUES (" +
place + ", "+ new_tip + ")" );
11 out.println("Your Post has been added under Place '"
+ HTMLencode(place)+"'");
}
else {
12 out.println("Your Message: '"+new_tip+ "' is too long!");
}
}
13 else if (place !=null && action.equals("View")) {
14 ResultSet rs = stmt.executeQuery("SELECT * FROM forum
WHERE place= "+place);
15 out.println("Tips about visiting p laces ...");
16 while(rs.next()) {
17 String tip = rs.getString("tip");
18 out.println("'"+tip+"'");
}
}
. . .
%>
19 </body></html>
<%
String HTMLencode(String value) { //server-side escaping method
20 value.replace("&", "&amp;");
21 value.replace("<", "&lt;");
22 value.replace(">", "&gt;");
23 return value;
}
%>
```

(a)

```
<SCRIPT>
24 var pos=document.URL.indexOf("User=")+5;
25
document.write(document.URL.substring(pos,document.URL.length)
);
</SCRIPT>
```

(b)

```
http://travelingForum/travelerTip.jsp?Action=
View&Place=Greece&User=Jesper
```

(c)

```
http://travelingForum/travelerTip.jsp?Action=
Post&Place=Greece&Tip=HiHiHiHiHiHiHiHiHiHiHiH
iHiHiHiHiHiHiHiHiH iHiHiHiHiHiHi<meta%20http
equiv="refresh"%20content="0;">&User=Jesper
```

(d)

```
http://travelingForum/travelerTip.jsp?Action=
Post&Place=Greece&Tip=<Script>document.location='htt
p://hackerSite/
stealCookie.jsp?cookie='+document.cookie;</Script>&User=H
acker
```

(e)

```
http://travelingForum/travelerTip.jsp?Action=
View&Place=Greece&Tip=HiHi&User=Jesper<Script>
document.
getElementByTagName('Tip')[child].innerHTML='<b>Our
Service is Bad, Please Go to Other Site!</b>'</Script>
```

(f)

Fig 2: Snippet from a server program for traveler tip. The above figure Includes (a) server side program (b) client side script (c) ordinary URL (d) reflected XSS (e) stored XSS (f) DOM-based XSS.

various classifiers and also it is used to shown the all classifier performance evaluation. It has a collection of machine learning algorithms and date preprocessing methods. The 10 fold cross validation is used to evaluate the robustness of the classifiers. The prediction accuracy is the performance measures for find the malicious web pages. Based on the correctly classified instances the prediction accuracy ratio is calculated. The prediction accuracy and training time are used to evaluate the performance of the trained models.

4.1 Machine Learning Methods

The experiments were carried out by three different machine learning methods like Naive Bayes (NB), Decision Tree (DT) and Multi-Layer Perceptron (MLP).

1) Naive Bayes: It is a statistical method based on Bayes rule. This method makes classification decisions by calculating the probabilities and costs related to each decision. The Bayesian classifiers adopt the concept of feature conditional independence by assuming that a feature value on a given class is independent of the values of other features. In order to classify a sample "x", the Bayes classifier calculates the a posteriori probability of each class given "x", and assigns to "x" the class with highest a posteriori probability. This classification method usually achieves high recognition rate keeping a low computational cost. Moreover, it is widely used in web applications and in anomaly detection.

2) Decision Tree: It is a common way to organize classification schemes. Every decision tree begins with the root node, and it is considered as a "parent" for other node. Each node in the tree evaluates an attribute to determine which path it should follow**.** Typically, the decision test is performed based on comparing a value against some constant. Classification using decision tree is performed by routing from the root node until it reaches a leaf node**.** These classification methods are a classic way to represent the data from a machine learning algorithm, which offers a fast and powerful way to express the structures in data.

3) Multi-Layer Perceptron : A multilayer perceptron (MLP) is a feed forward artificial neural network model that maps sets of input data onto a set of appropriate outputs. MLP utilizes a supervised learning technique called back propagation for training the network. Neural networks are a different paradigm for computing and it is an inspiration from neuroscience. Neural networks are particularly effective for predicting events when the networks have a large database. This network imitates the human brain. Artificial neurons or processing elements are highly simplified models of biological neurons. As in biological neurons, artificial neurons have a number of inputs, cell body and output that can be connected to a number of other artificial neurons. This network is densely interconnected together by learning rule which to adjust the strength of the connection between the units in response to externally supplied data.

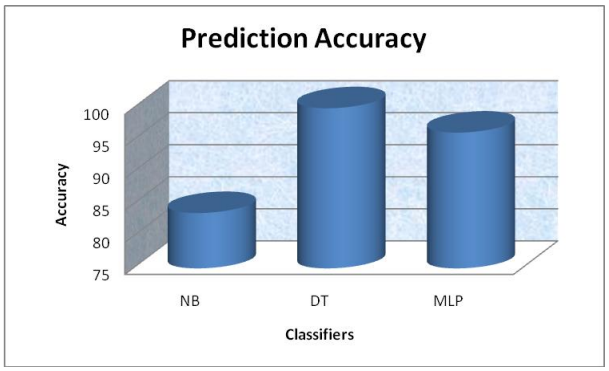

Fig3: Prediction Accuracy

4.2 Data Collection

We got 500 URLs from xssed database (http://www. xssed.com) for our experiments. These URLs are used for verification based on the result we identify the malicious URLs. Finally our dataset contains 200 malicious URLs correspond to the attacks of various types like Redirected attack, script Injection attack and XSS attack occurred from Sep 2009 to Jan 2013.

4.3 Performance Analysis

The so-called 10-fold cross-validation was used to evaluate the results. This technique aims to predict and to estimate how correct a model will be executed in practice. First, the original whole datasets are divided into 10 folds. Each time, one of the folds is used as test set, and the other 9 folds are put together to form the training set. This process is repeated 10 times. Thus, the performance rates are obtained as the mean across all 10 trials.

4.4 Classification using Weka

There are various classification algorithms are used for the XSS web page classification, such as multilayer perceptron, decision tree induction and naïve bayes. These are implemented and trained using WEKA. The Weka is an Open Source and collection of state-of-the-art machine learning algorithms and data preprocessing tools. The robustness of the

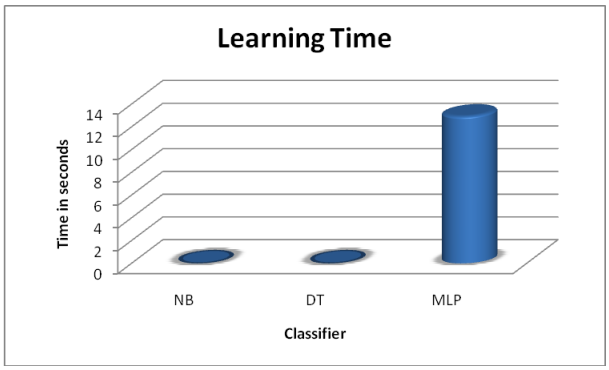

Fig4: Learning Time

classifiers are evaluated using 10 fold cross validation for all the algorithms. Predictive accuracy is used as a primary performance measure for predicting the XSS web pages. Based on the ration of correctly classified instances in the test dataset and the total number of test cases the prediction accuracy is measured. The prediction accuracy and the training time are two criteria used to evaluate the performances of the trained models and the prediction accuracy of the each model is compared. The 10-fold cross validation results of the three classifiers multilayer perceptron (MLP), decision tree induction (DT) and naive bayes (NB) are summarized in Table1 and Table2 and the performance of the models is illustrated in figures Fig 3 and Fig 4.

Table1: Performance comparison of classifiers

Criteria	Classifiers		
	NB	**DT**	**MLP**
Time taken to build model(sec)	0.01	0.01	12.93
Correctly classified instances	133	200	194
Incorrectly classified instances	27	0	6
Prediction accuracy	83.64%	100%	96.20%

Table2: Comparison of estimates

Criteria	Classifiers		
	NB	**DT**	**MLP**
Kappa statistic	0.5471	1	0.9294
Mean absolute error	0.1375	0.0604	0.0528
Root mean squared error	0.2883	0.064	0.1297
Relative absolute error	48.2159	22.293	13.4793
Root relative square error	77.9958	17.4911	29.635

The above table shows that comparison of NB, DT and MLP. The NB algorithm gives the low accuracy compare to other algorithms. Though Multilayer Perceptron consumes relatively more time for learning than J48, the outcome (accuracy) is appreciable and the interpretability of Decision tree is interesting. In general, both Decision Trees and Neural Networks has advantages and drawbacks, hence the current research is towards constructing a Hybrid algorithm which encompasses advantages of both the algorithms i.e., accuracy and performance of Neural Networks and the interpretability of Decision Trees.

5. CONCLUSION AND FUTURE WORK

Finding malicious web pages is a challenging task in internet security. This paper focus on classifying XSS attacks on web pages by extracting various features from the web document and URL using different data mining techniques.

In this, the experimental result shows that Multi-layer perceptron and Decision tree are giving the high accuracy rate when compare to Naïve bayes classifier. And also XSS Scan is used to demonstrate how t traces the XSS web pages using different features which are extracted by the web documents.

With regards to future work, while considering the XSS attacks, it is clear that the search for new features and attacks is a good opportunity for the researchers. Moreover, the experiments conducted were limited to three types of XSS attacks and techniques. Thus, there are other techniques are also can fit the problem and used to obtain the very good results.

6. ACKNOWLEDGMENTS

No words are enough to express my thanks towards my friends. They had helped me to explore this broad area in organized manner and provide me with all ideas on how to work towards research oriented work and reach success. Most importantly I would like to thanks my parents to show me a right direction that helps me to stay calm in oddest of the times.

7. REFERENCES

[1] C. Alme, "Web browsers: An emerging platform under attack," MCAfee, Tech. Rep., 2008.

[2] M. Arciemowicz. phpBB 2.0.18 XSS and Full Path Disclosure. http://archives.neohapsis.com/ archives/ fulldisclosure /2005-12/0829. html, December 2006.

[3] A. Barth, C. Jackson, and J. Mitchell, "Securing frame communication in browsers," Commun. ACM, vol. 52, no. 6, pp. 83–91, 2009.

[4] S. Bubrouski. Advisory: XSS in WebCal (v1.11-v3.04). http://archives.neohapsis.com/archives/ fulldisclosure/2005-12/0810.html, December 2005.

[5] S. Chen, J. Xu, N. Nakka, Z. Kalbarczyk, and R. Iyer. Defeating Memory Corruption Attacks via Pointer Taintedness Detection. In IEEE International Conference on Dependable Systems and Networks (DSN), 2004.

[6] F. Esponda, S. Forrest, and P. Helman, "A formal framework for positive and negative detection schemes," Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, vol. 34, no. 1, pp. 357–373, 2004.

[7] Grossman, J., Hansen R., Petkov, D.P., Rager, A. e Fogie, S. "Cross Site Scripting Attacks: XSS Exploits and Defense". Burlington, MA, EUA, Syngress Publishing Inc. 2007.

[8] B. Garrett, H. Travis, I. Micheal, P. Atul, and B. Kevin, "Social networks and context-aware spam," in Proceedings of the ACM 2008 conference on Computer supported cooperative work. San Diego, CA, USA: ACM, 2008.

[9] J. Goguen and J. Meseguer. Security Policies and Security Models. In *IEEE Symposium on Security and Privacy*, 1982.

[10] D. Gollmann, "Securing web applications," Information Security Technical Report, vol. 13, no. 1, pp. 1–9, 2008.

[11] Google, "Google trends." [Online]. Available: http://www.google.com/trends/hottrends [Accessed: 08/10/2012]

[12] M. Group. MyBB - Home. http://www.mybboard. com/, 2006.

[13] V. Haldar, D. Chandra, and M. Franz. Dynamic Taint Propagation for Java. In Twenty-First Annual Computer Security Applications Conference (ACSAC), 2005.

[14] . Hallaraker and G. Vigna. Detecting Malicious JavaScript Code in Mozilla. In *10th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS05)*, 2005.

[15] N. Jovanovic, C. Kruegel, and E. Kirda. Pixy: A Static Analysis Tool for DetectingWeb Application Vulnerabilities (Short Paper). In *IEEE Symposium on Security and Privacy*, 2006.

[16] E. Kirda, C. Kruegel, G. Vigna, and N. Jovanovic. Noxes: A Client-Side Solution for Mitigating Cross-Site Scripting Attacks. In The 21st ACM Symposium on Applied Computing (SAC 2006), 2006.

[17] C. Kruegel and G. Vigna. Anomaly Detection ofWeb-based Attacks. In 10th ACM Conference on Computer and Communication Security (CCS-03) Washington, DC, USA, October 27-31, pages 251 – 261, October 2003.

[18] Microsoft, "Microsoft security intelligence report," Microsoft, Tech. Rep., 2009.

[19] Mozilla Foundation. JavaScript Security: Same Origin. http://www.mozilla.org/projects/ security/components/same-origin.html, February 2006.

[20] Netscape. . Using data tainting for security. http://wp.netscape.com/eng/mozilla/3.0/handbook/javascript/advtopic.htm, 2006.

[21] N. Provos, D. McNamee, P. Mavrommatis, K. Wang, and A. Modadugu, "The ghost in the browser: Analysis of webbased malware," in Proceedings of the first USENIX workshop on hot topics in Botnets, 2007.

[22] Ryan Barnett Senior Security Researcher Trustwave's SpiderLabs rbarnett@trustwave.com. Revision 1(January 7, 2011) "XSS Street-Fight:The Only Rule Is There Are No Rules".

[23] C. Seifert, I. Welch, and P. Komisarczuk, "Identification of malicious web pages with static heuristics," in Telecommunication Networks and Applications Conference, 2008. ATNAC 2008. Australasian, 2008, pp. 91–96.

[24] Symantic, "Security threat report - trend for 2012," Symantic, Tech. Rep., April 2012.

[25] Yahoo, "Web search document for yahoo!" [Online]. Available: http://developer.yahoo.com/search/web/V1/ webSearch. html [Accessed: 08/10/2012].

[26] Wasserman, G e Su, Z. "Static Detection of Cross- Site Scripting Vulnerabilities". In: 30th International Conference on Software Engineering, 2008.