

Identifying XSS Attack using Combined Machine Learning Classifier

Aishwarya M¹ and H S Jayanna²

¹⁻²Department of ISE, Siddaganga Institute of Technology Tumakuru, India
Email: aishwarya.m898@gmail.com, jayannahs@sit.ac.in

Abstract—In recent years the number of cyber-attacks is being increased rapidly. There are many unique cyber-attacks emerging day by day that require more attention. One such attack is cross-site scripting (XSS). This attack can be caused by inserting the malicious code on a web page to interact with the attacker's web server. The attacker can get much useful information about the victim and it is necessary to identify and prevent this type of attack. There are many methods available to detect XSS attacks like using n-grams, machine learning methods, intrusion detection systems (IDS), etc. In this paper, we used Support vector machine (SVM), K-nearest neighbour (KNN), discriminant classifier and proposed a combined classifier that best identifies the attack.

Index Terms— Cross-site scripting (XSS), Classifiers (SVM, KNN, discriminant), feature selection, proposed combined classifier.

I. INTRODUCTION

The different sectors like medicine, management, business, research etc., make use of the internet every day and due to the covid-19 pandemic, internet usage is increasing day by day. Therefore the amount of cyber-attacks is also being increased rapidly. According to the cyber security threatscape, when compared to quarter 4 of 2020 the cyber-attacks were increased about 17% in the first quarter of 2021 and the number of targeted attacks was also increased by 1.2% Ref. [1]. The main motive of these attacks was to collect the data and the data refers to personal data, login credentials, session details etc. There are unique types of cyber-attacks emerging day by day and one such attack is XSS. According to the open-source foundation for application security (OWASP), the XSS attack is one of the top 10 vulnerabilities Ref. [2]. In web vulnerabilities the XSS vulnerability is the common one Ref. [3]. The usage of Web applications plays an important role in exchanging data over the internet because it contains much useful information about users like personal and sensitive information, session information, cookies details, etc. This type of data attracts the attacker to get the information from the web applications by unauthorized access by making use of malicious scripts, malware and this type of attack is known as XSS attack.

An XSS attack is a type of insertion attack where malicious code is inserted into trusted websites Ref. [4]. It will occur when an infected website is run on the victim's computer and the malicious script will run on the victim's computer without his/her knowledge, the script will communicate with the attacker's web server and the attacker will have the access to victim's personal information.

There are three types of XSS attacks; they are reflected XSS, stored XSS and DOM-based XSS attacks.

A. Reflected XSS

It is a type of attack where the malicious script is delivered to the victim through different means like email, through another website, through messages. It tricks the victim in such a way that the victim will click the malicious link, or visit the malicious site or form on the victim's browser where the attack is reflected and the browser will execute the malicious script. It is also known as a non-persistent XSS attack.

B. Stored XSS

In this type of attack, the malicious script is stored on the target server permanently for example database. So whenever a user requests for the data stored on that database, the victim will receive the malicious script from the server. This attack is also known as a persistent XSS attack. Stored XSS attack affects more when compared to reflected XSS attack.

C. DOM-based XSS

Here the malicious script is run on the victim's browser by modifying the DOM environment so that the code on the client-side will run in an unpredicted manner.

The study on XSS attacks is not done in a long time because they are considered harmless however, this opinion is changed in recent years and there are several studies related to XSS attacks. Detecting XSS attacks is important and there exist many methods to detect the attack like machine learning methods, IDS, deep learning methods etc.

This paper is organized as follows. Section II explains the related work where we discuss different methods used to detect XSS attacks and section III describes the proposed methodology, section IV shows the experimental results and finally section V gives the conclusion.

II. RELATED WORK

To detect the attack capturing and analysing network traffic becomes very important because, the network traffic contains much useful information like network packets, session id, cookie details, visited websites, login attempts, login credentials etc., all this information can be extracted from the network flow. Many researchers have tried to detect XSS attacks using different methods like IDS, n-grams, machine learning.

In Ref. [5], the authors make use of IDS to detect the XSS attack. Here they used snort IDS to create rules and then to filter out the packets. For the experiment they used a compromised server and 3 pc to attack the server and packets were captured and analyzed using Wireshark, the snort rules were introduced and deep packet inspection is used to detect the attack and the details of the attack were stored in an alert file and logged in terms of dumps as metadata.

The authors in Ref. [6] also used the snort rule along with honeypot to detect the attack. They used the auto-generate IDS rule method to generate the rules. They experimented using 200 XSS attack samples and experimented in two stages and in the second stage out of 200 attacks 174 attacks were blocked by IDS. 87% of attacks were successfully blocked by IDS.

In Ref. [5], Ref. [6], both used IDS but failed to give accurate results due to case sensitive limitations on the snort rule.

In Ref. [7], the authors explained that by using different type of machine learning methods the XSS attacks can be detected effectively.

The authors in Ref. [8] demonstrated that SVM, k-NN, and RF can be used to build classifiers for XSS. The complete set of features used in the java script language and the scripts associated with malicious code is considered as feature set. Here SVM gave better results with the accuracy of 97%.

In Ref. [9], the authors proposed a feature set extraction technique to extract the malicious features in the HTTP traffic dataset. They used J48, one rule (OneR) and NB machine learning classifiers to detect the attack. When compared to OneR and NB, J48 performed well with the highest detection rate of 94.5%.

In Ref. [10], Ref. [11], Ref. [12], Ref. [13], Ref. [14], the authors of these papers used different machine learning methods for the experiments. They extracted different features in java script and in HTML to give better results.

In Ref. [15], the authors used SVM, NB and KNN with n-gram to detect the attack. N-gram is used to find similarities in script tags between malicious scripts. 400 URL data is used for evaluation purposes. SVM with n-gram showed a better result with an accuracy of 98%.

In Ref. [16], the authors used improved n-gram and alternating decision tree (ADTree) as classifier to detect the attack in online social network. The authors implemented two approaches & second approach showed better results by identifying malicious and benign script with precision of 0.918 and 0.967.

The authors in Ref. [17] used convolution neural network (CNN), SVM and multinomial naïve Bayes (MNB) to detect the attack.

From the literature survey we observed that the experiments conducted using machine learning methods gave better result when compared to IDS. So in this paper we make use of machine learning classifiers and propose a combined classifier that will detect the XSS attack accurately.

III. PROPOSED METHODOLOGY

The proposed methodology is discussed in four stages:

A. Stage 1: Dataset

To create a balance in the dataset, the dataset used in the experiment consists of both malicious and benign URL. The dataset is downloaded from github Ref. [18]. The dataset is divided into two parts and one is used for training and another one is used for testing.

Feature selection

While doing feature selection, both structural and behavioural features are considered. In this experiment total of 62 features were considered for the experiment.

Structural features

The structural feature consists of all non-alpha numeric characters that can possibly occur in a JavaScript. By using these characters in the URL, the attacker can add unnecessary characters, spaces in between and trick the user. Table I shows the structural feature characters:

TABLE I. STRUCTURAL FEATURES

Features Group	Terms
Punctuation	` , ! , @ , # , \$, % , ^ , & , * , < , > , ? , / , [, { , } , ; , : , " , ' , ' , \ , , ~ , + , _ , (,) , = , ~ , , , space ,
Punctuation combination	>< , ' " ">< , [] , = , & #

Behavioural features-

Behavioural features include certain keywords, commands and functions that are used in the JavaScript. The attacker can use different behavioural features in the URL and can trick the user. The Table II shows the behavioural features:

TABLE II. BEHAVIOURAL FEATURES

Features	Description
Readability	Is the script readable - the number of alphabetical characters
Objects	Document, window, iframe, location, This
Events	Onload, Onerror
Methods	createelement, String.fromCharCode, Search
Tags	DIV, IMG, <script
Attributes	SRC, Href, Cookie
Reserve	Var
Functions	Eval ()
Protocol	HTTP
External File	.js file

B. Stage 2: Classifiers

Here we consider four classifiers they are SVM, KNN, Discriminant and our proposed combined classifier.

Support vector machine (SVM)

SVM is used to perform supervised learning for regression and classification for datasets. It is used to sort two types of data by classification. So it makes use of hyper planes to sort the groups to their patterns. It requires some group of data to be trained and labelled for classification. The training group is separately kept and trained.

It builds a new model that separates the data to their respective groups. It can also be known as non-probabilistic classifier. Figure 1 shows the workflow of SVM classifier.

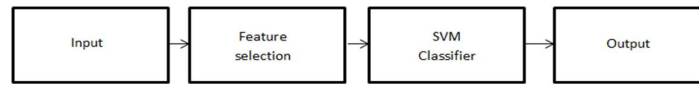


Figure 1. SVM Workflow

K – nearest neighbour (KNN):

This is one of the basic methods of classification which is non-parametric and very efficient in classification. KNN can be used for classification and regression as well. The data is classified majorly by plurality vote from the neighbours and that data is classified to the group with nearest k value. The KNN performance is dependent on K parameter which is found by the user. K training points are selected by considering the nearest distance to the test data and hence the performance of KNN is greatly dependent on the parameter K. Figure 2 shows the workflow of KNN classifier.

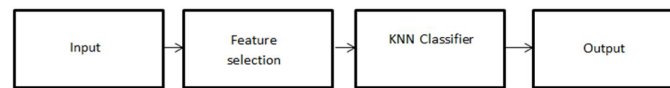


Figure 2. KNN Workflow

Discriminant classifier

Discriminant classifier is one of the classifier method used for classification Ref. [19]. It works on the assumption that based on different Gaussian distributions different classes generate their data. To estimate the parameters of Gaussian distributions of every class, the fitting function is used to train the classifier. While predicting the class for new data, the trained classifier makes use of the class with the smallest misclassification. Figure 3 shows the workflow of discriminant classifier.

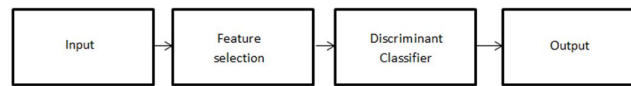


Figure 3. Discriminant Workflow

Proposed combined classifier

In proposed classifier we make use of all the three classifiers. The dataset is loaded and after the feature selection process the data is sent to classifiers and we can see their result individually. After that in proposed classifier the majority voting system is used. The output (1-malicious or 2-benign) of three classifiers is stored in an array. Whenever a particular URL is selected from the dataset, for that one if any two classifiers predict their output as malicious then the proposed classifier will also predict its output as malicious and raise an alert showing malicious detected. Figure 4 shows the workflow of proposed combined classifier.

While checking the output, whenever we select the particular URL, we can see the URL after the data is loaded into the model. By using this classifier we can reduce the amount of wrong predictions that can happen while using single classifiers. So the accuracy of the classifier also increases.

C. Stage 3: Algorithm

URL is used as input to classifiers, from which features are extracted. Based on the training data, the classifier will give the class of the output as Benign or Malicious.

Step 1: Get the benchmark dataset.

Step 2: Store the dataset in an Excel sheet

Step 3: Extract the important features from Dataset

Step 4: Divide Dataset 1 and Dataset 2 in two parts, one for training and other for testing.

Step 5: Train machine learning classifiers using training dataset.

Step 6: Test the dataset and if malicious found raise alert.

D. Stage 4: Metrics

To check the performance of the classifier we make use of confusion matrix and six metrics they are accuracy, precision, prevalence, F1 score, True positive rate and false positive rate.

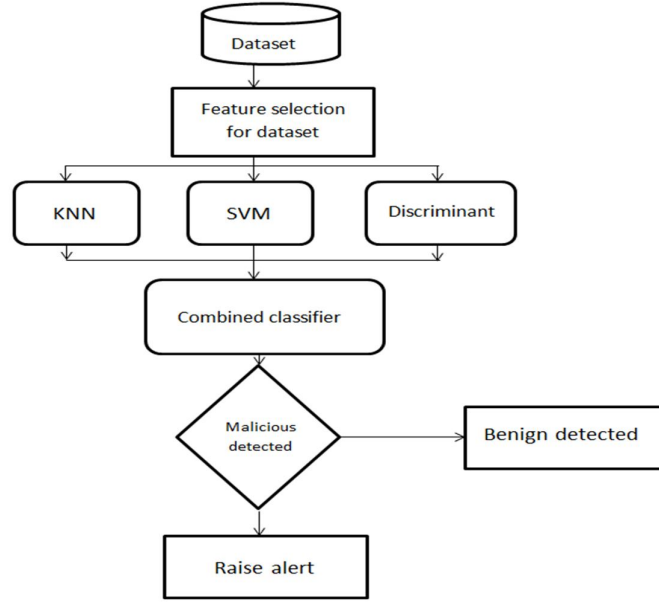


Figure 4. Combined classifier Workflow

Accuracy: To decide how often is the classifier correct in overall. It is given by the formula

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

Precision: To decide, how often is the classifier correct when it predicts benign.

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

Prevalence: To decide how often does the benign condition actually occur in our sample. It can be calculated as

$$Prevalence = \frac{Actual\ benign}{Total\ dataset} \quad (3)$$

True positive Rate: To decide how often the classifier predict benign, when it is actually benign. It is calculated as

$$TPR = \frac{TP}{TP+FN} \quad (4)$$

False positive Rate: To decide, how often the classifier predicts benign when it is actually malicious. It is calculated as

$$FPR = \frac{FP}{FP+TN} \quad (5)$$

F1 score: F1 score is nothing but the harmonic mean of recall and precision.

IV. EXPERIMENTAL RESULT

For the experiment we considered SVM, KNN, discriminant and combined classifiers. The prepared dataset is divided into 80% training and 20% testing and total 4800 URL dataset is considered out of which 4000 dataset is used for training the classifiers and 800 dataset is used for testing. The experiment is carried out using Matlab R2021a version. The dataset is loaded into the classifiers to check whether it is malicious or not. To properly balance the test dataset, we considered 400 malicious and 400 benign URL's. The experiment is conducted and after the experiment we observed that out of 800 dataset, our proposed combined classifier correctly predicted 788 dataset and wrongly predicted only 12 dataset.

From the table III and Figure 5, we can see that our proposed combined classifier is giving highest accuracy of 98.5% when compared to other classifiers.

TABLE III. EXPERIMENTAL RESULT

Metrics	Method evaluation %			
	KNN	SVM	Discriminant classifier	Proposed Combined classifier
Accuracy	88.5	96.5	96	98.5
Precision	89.69	94.28	93.39	97.08
Prevalence	0.5	0.5	0.5	0.5
True positive rate	87	99	99	1
False positive rate	0.1	0.06	0.07	0.03
F1 score	83.32	96.58	96.11	98.5

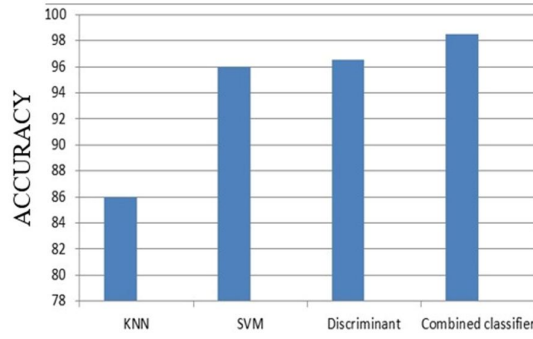


Figure 5. Accuracy comparison

V. CONCLUSION

XSS attack is one of the most famous vulnerability attacks for quite some time and there is a necessity to detect these types of attacks and take measures accordingly. From the experiment we can clearly see that our proposed combined classifier is giving accurate result of 98.5% when compared to individual classifiers (SVM, KNN and discriminant). From this we can say that instead of using single classifiers to detect the attack, we can use combined classifier so that it can predict the result with highest accuracy.

Detection of XSS attack is quite a task and with different kinds of URL's, it will become even difficult. So for future work, dataset can be increased and after the attack detection, we can implement some measures that will help in prevention of these kinds of attacks.

REFERENCES

- [1] "Cybersecurity Threatscape: Q1 2021," *Positive Technologies*, Dec. 30, 2021.
- [2] "OWASP Top 10 Security Risks & Vulnerabilities," OWASP Foundation, 2020. [Online]., *Sucuri*, Dec. 30, 2021.
- [3] M. Liu, B. Zhang, W. Chen, and X. Zhang, "A Survey of Exploitation and Detection Methods of XSS Vulnerabilities," *IEEE Access*, vol. 7, pp. 182004–182016, 2019, doi: 10.1109/ACCESS.2019.2960449.
- [4] "Cross site scripting (XSS)," *Cross Site Scripting (XSS) | OWASP Foundation*, Dec. 30, 2021.
- [5] K. Gupta, R. Ranjan Singh, and M. Dixit, "Cross site scripting (XSS) attack detection using intrusion detection system," in *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, Jun. 2017, pp. 199–203. doi: 10.1109/ICCONS.2017.8250709.
- [6] S. Syaifuddin, D. Risqiwati, and H. A. Sidharta, "Automation Snort Rule for XSS Detection with Honeypot," in *2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, Oct. 2018, pp. 584–588. doi: 10.1109/EECSI.2018.8752961.
- [7] J. Kaur and U. Garg, "A Detailed Survey on Recent XSS Web-Attacks Machine Learning Detection Techniques," in *2021 2nd Global Conference for Advancement in Technology (GCAT)*, Oct. 2021, pp. 1–6. doi: 10.1109/GCAT52182.2021.9587569.
- [8] F. A. Mereani and J. M. Howe, "Detecting Cross-Site Scripting Attacks Using Machine Learning," 2018, pp. 200–210. doi: 10.1007/978-3-319-74690-6_20.
- [9] S. Sharma, P. Zavarsky, and S. Butakov, "Machine Learning based Intrusion Detection System for Web-Based Attacks," in *2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, May 2020, pp. 227–230. doi: 10.1109/BigDataSecurity-HPSC-IDS49724.2020.00048.
- [10] S. Kascheev and T. Olenchikova, "The Detecting Cross-Site Scripting (XSS) Using Machine Learning Methods," in *2020 Global Smart Industry Conference (GloSIC)*, Nov. 2020, pp. 265–270. doi: 10.1109/GloSIC50886.2020.9267866.

- [11] R. Banerjee, A. Bakshi, N. Singh, and S. K. Bishnu, "Detection of XSS in web applications using Machine Learning Classifiers," in *2020 4th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*, Oct. 2020, pp. 1–5. doi: 10.1109/IEMENTech51367.2020.9270052.
- [12] S. Perumal and K. Sujatha P, "Stacking Ensemble-based XSS Attack Detection Strategy Using Classification Algorithms," in *2021 6th International Conference on Communication and Electronics Systems (ICCES)*, Jul. 2021, pp. 897–901. doi: 10.1109/ICCES51350.2021.9489177.
- [13] H.-C. Chen, A. Nshimiyimana, C. Damarjati, and P.-H. Chang, "Detection and Prevention of Cross-site Scripting Attack with Combined Approaches," in *2021 International Conference on Electronics, Information, and Communication (ICEIC)*, Jan. 2021, pp. 1–4. doi: 10.1109/ICEIC51217.2021.9369796.
- [14] F. A. Mereani and J. M. Howe, "Preventing Cross-Site Scripting Attacks by Combining Classifiers," in *Proceedings of the 10th International Joint Conference on Computational Intelligence*, 2018, pp. 135–143. doi: 10.5220/0006894901350143.
- [15] G. Habibi and N. Surantha, "XSS Attack Detection With Machine Learning and n-Gram Methods," in *2020 International Conference on Information Management and Technology (ICIMTech)*, Aug. 2020, pp. 516–520. doi: 10.1109/ICIMTech50083.2020.9210946.
- [16] R. Wang, X. Jia, Q. Li, and D. Zhang, "Improved N-gram approach for cross-site scripting detection in Online Social Network," in *2015 Science and Information Conference (SAI)*, Jul. 2015, pp. 1206–1212. doi: 10.1109/SAI.2015.7237298.
- [17] S. Akaishi and R. Uda, "Classification of XSS Attacks by Machine Learning with Frequency of Appearance and Co-occurrence," in *2019 53rd Annual Conference on Information Sciences and Systems (CISS)*, Mar. 2019, pp. 1–6. doi: 10.1109/CISS.2019.8693047.
- [18] "Cross-site-scripting-xss/xssdatasets at master · fmereani/cross-site-scripting-XSS," *GitHub*, Jan. 15, 2022.
- [19] "FITCDISCR," *Discriminant Analysis Classification - MATLAB & Simulink - MathWorks India*, Feb. 28, 2022.