A SEMINAR REPORT ON

# ”An Integrated XSS-Based Attack Detection Technique”

SUBMITTED TO

SAVITRIBAI PHULE PUNE UNIVERSITY

As Per

COMPUTER ENGINEERING

Submited By

TAKE SWAPNIL RAJENDRA

Roll No : 38

Under The Guidance Of

Prof. Pachhade R.C.

DEPARTMENT OF COMPUTER ENGINEERING

VISHWABHARATI ACADEMY’S COLLAGE OF ENGINEERING

SAROLA BADDI

AHMEDNAGAR 414201

2021-2022

**DEPARTMENT OF COMPUTER ENGINEERING**
**Vishwabharati Academy's Collage Of Engineering**
**Sarola Baddi,Ahmednagar**

# CERTIFICATE

This is to certify that the Mr. Take Swapnil Rajendra from TE Computer Engineering, Roll no. 38 submitted his seminar report on **"An Integrated XSS-Based Attack Detection Technique"** under my guidance Prof.Pachhade R.C. and supervision. The work has been done to my satisfaction during the academic year 2021-2022 under Savitribai Phule Pune University.

| | | |
|---|---|---|
| Prof.Pachhade R.C. | Prof.Joshi S.G. | Prof.Dhongade V.S. |
| **Seminar Guide** | **H.O.D** | **Principal** |

Date :

Place :Ahmednagar

# Acknowledgement

This is a great pleasure & immense satisfaction to express my deepest sense of gratitude & thanks to everyone who has directly or indirectly helped me in completing my Seminar work successfully.

I express my gratitude towards seminar guide **Prof.Pachhade R. C.** and **Prof.Joshi S. G.** Head Of Department of Computer Engineering, Vishwabharati Academy's Collage Of Engineering, Sarola Baddi, Ahmednagar, who guided & encouraged me in completing the Seminar work in scheduled time.

I would like to thanks our Principal **Prof.Dhongade V. S.** , for his extended support. No words are sufficient to express my gratitude to my family for their unwavering encouragement. I also thank all friends for being a constant source of my support.

Mr. Take Swapnil Rajendra

T. E. Computer

Roll No: 38

# Contents

**1 Introduction** **1**

1.1 Types of XSS-based attacks . . . . . . . . . . . . . . . . . . . . . . 1

1.2 Scope . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

1.3 Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

**2 Literature Review** **3**

**3 System Architecture** **4**

3.1 System Architecture . . . . . . . . . . . . . . . . . . . . . . . . . 4

3.1.1 System Architecture Details . . . . . . . . . . . . . . . . . 5

**4 Domain Specific Information** **6**

4.1 What is cybersecurity? . . . . . . . . . . . . . . . . . . . . . . . . 6

4.2 Why is cybersecurity important? . . . . . . . . . . . . . . . . . . . 7

4.3 Types of cybersecurity threats: . . . . . . . . . . . . . . . . . . . 7

**5 Implementation Details** **9**

5.1 HTML-based Features Sub-Model . . . . . . . . . . . . . . . . . . . 9

5.1.1 Algorithm for Parsing HTML Documents . . . . . . . . 9

5.2 Javascript-Based Features Sub-Model . . . . . . . . . . . . . . . 10

5.2.1 Algorithm for Parsing Javascript Documents . . . . . . 10

5.3 Modules in the System. . . . . . . . . . . . . . . . . . . . . . . . 12

5.3.1 Collecting Raw Data . . . . . . . . . . . . . . . . . . . . 12

5.3.2 Dynamic Feature Extraction Model . . . . . . . . . . . . 12

5.3.3 Artificial Neural Network Model . . . . . . . . . . . . . . 12

5.4 Flowchart . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

5.4.1 Features Identification . . . . . . . . . . . . . . . . . . . 13

5.4.2 Collecting Webpages . . . . . . . . . . . . . . . . . . . . 14

5.4.3 Features Extraction and Training Dataset Construction . 14

5.4.4 Multilayer Perceptron Technique . . . . . . . . . . . . . 14

**6 Result Analysis** **16**

7 Future Scope 18

8 Conclusion 19

References 20

# List of Figures

3.1 **System Architecture** . . . . . . . . . . . . . . . . . . . . . . . . . 4

5.1 **The Framework of Proposed System** . . . . . . . . . . . . . 13
5.2 **Flowchart** . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

6.1 Model Accuracy On Training and Testing Dataset . . . . . . . 16
6.2 Model Log Loss Values Over Time . . . . . . . . . . . . . . . . . 17

# Abstract

Web technology has been increased exponentially in daily volume and interactions involving web-based services, such as self-driving finance in web banking, Chat bots /AI assistants and recommendation engines in e-commerce, social networking sites such as Facebook, Twitter, forums, blogs, and much more. In all the given examples dynamic web application plays an important role that enable Cyber-attacks to be executed. One of the common attack in web application is "Cross-Site Scripting" also familiar by the name "XSS". Nowadays, XSS is still dramatically increasing and Considered as one of the most severe threats for organizations. Therefore, it has the potentials to be applied for XSS based attack detection in either the client -side or the server-side. primarily, there are three types of XSS based attacks attacks.Stored-XSS, Reflected-XSS, DOM-based XSS. The defense mechanism against these attacks could be on either the Server-side, Client-side, or both.In terms of analyzing, there are three approaches to defense against XSS-based are:Static analysis approach, Dynamic analysis approach, The hybrid approach.

# Chapter 1

# Introduction

Web technology has been increased exponentially in daily volume and interactions involving web-based services, such as self-driving finance in web banking, Chatbots /AI assistants and recommendation engines in e-commerce, social networking sites such as Facebook, Twitter, forums, blogs, and much more. This technology has become an integral part of our daily and private lives, and at the same time, web applications became primary targets of cybercriminals. Cybercriminals exploit the poor code experiences of web developers, weaknesses within the code, improper user input sanitization, and non-compliance with security standards by the software package developers .

Additionally, the vulnerabilities may be found in reusable software components (i.e., third-party libraries, open-source software, etc.) which are heavily used to develop web applications, also vulnerable cyber-defense system where attackers regularly develop their offensive tactics by devising new ways to bypass the defense systems, and exploiting the vast sophistication of AI technology to facilitate their pernicious tasks . It mandates the development of more sophisticated web application cyber-defense systems, which can be accomplished using the latest AI concepts with high precision to tackle the new web-based attacks.

## 1.1 Types of XSS-based attacks

- Stored XSS : Stored XSS generally occurs when user input is stored on the target server, such as in a database, in a message forum, visitor log, comment field, etc. And then a victim is able to retrieve the stored data from the web application without that data being made safe to render in the browser. With the advent of HTML5, and other browser technologies, we can envision the attack payload being permanently stored in the victim's browser, such as an HTML5 database, and never being sent to the server at all.

- Reflected XSS : Reflected XSS occurs when user input is immediately returned by a web application in an error message, search result, or any other response that includes some or all of the input provided by the user as part of the request, without that data being made safe to render in the browser, and without permanently storing the user provided data. In some cases, the user provided data may never even leave the browser
- DOM Based XSS : DOM based XSS is a form of XSS where the entire tainted data flow from source to sink takes place in the browser, i.e., the source of the data is in the DOM, the sink is also in the DOM, and the data flow never leaves the browser. For example, the source (where malicious data is read) could be the URL of the page (e.g., document.location.href), or it could be an element of the HTML, and the sink is a sensitive method call that causes the execution of the malicious data (e.g., document.write)."

## 1.2 Scope

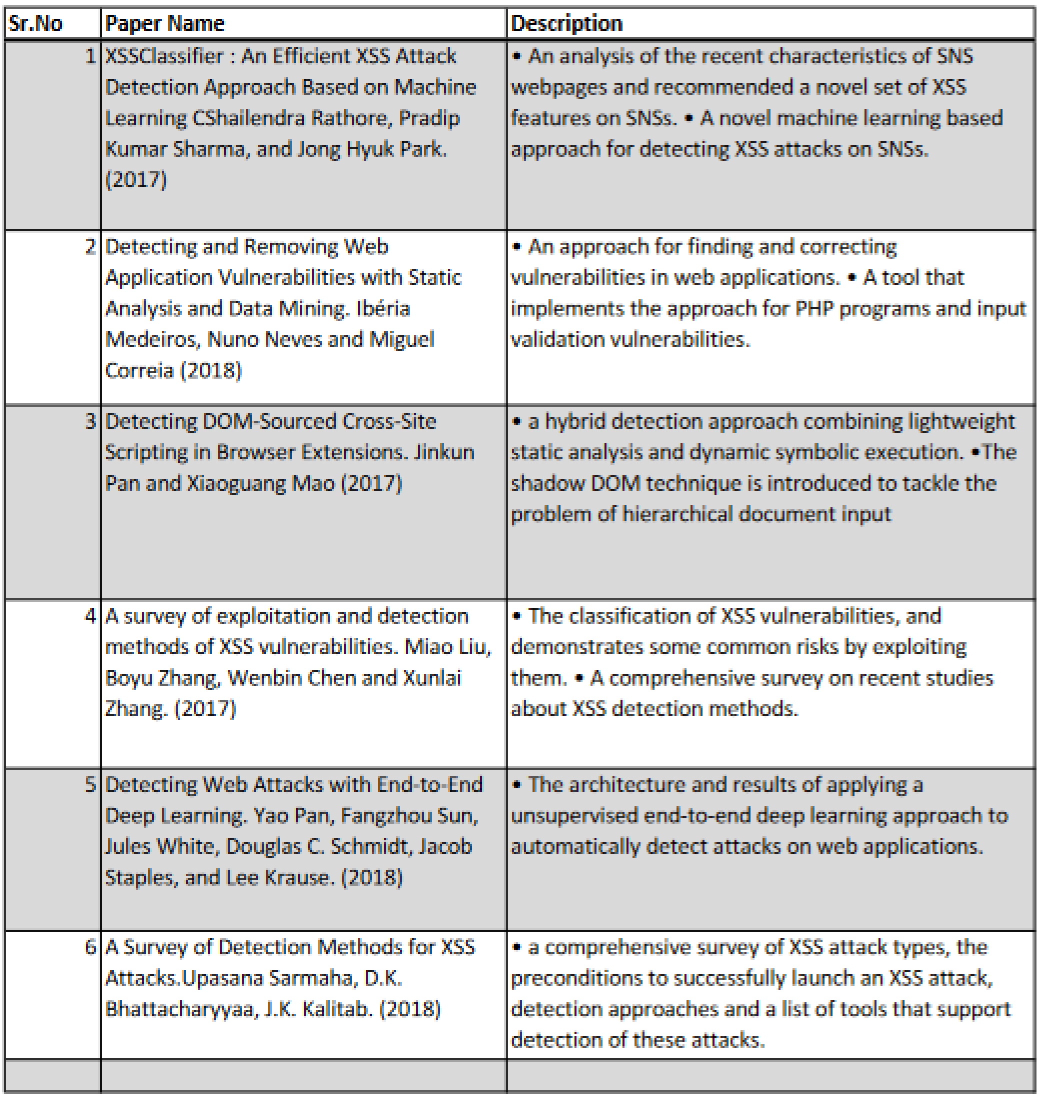
- The project analyzes the current situation on scripting attacks.
- The work covers the current problems associated with XSS attack at the browser side.
- The solution is geared towards ease of use of tools to detect XSS attacks.

## 1.3 Objectives

- To analyze, classify and discuss the current situation of XSS problem.
- To identify, build and test the components of the proposed tool.
- To determine the capability, accuracy and performance.

# Chapter 2

# Literature Review



| Sr.No | Paper Name | Description |
|---|---|---|
| 1 | XSSClassifier : An Efficient XSS Attack Detection Approach Based on Machine Learning CShailendra Rathore, Pradip Kumar Sharma, and Jong Hyuk Park. (2017) | • An analysis of the recent characteristics of SNS webpages and recommended a novel set of XSS features on SNSs. • A novel machine learning based approach for detecting XSS attacks on SNSs. |
| 2 | Detecting and Removing Web Application Vulnerabilities with Static Analysis and Data Mining. Ibéria Medeiros, Nuno Neves and Miguel Correia (2018) | • An approach for finding and correcting vulnerabilities in web applications. • A tool that implements the approach for PHP programs and input validation vulnerabilities. |
| 3 | Detecting DOM-Sourced Cross-Site Scripting in Browser Extensions. Jinkun Pan and Xiaoguang Mao (2017) | • a hybrid detection approach combining lightweight static analysis and dynamic symbolic execution. •The shadow DOM technique is introduced to tackle the problem of hierarchical document input |
| 4 | A survey of exploitation and detection methods of XSS vulnerabilities. Miao Liu, Boyu Zhang, Wenbin Chen and Xunlai Zhang. (2017) | • The classification of XSS vulnerabilities, and demonstrates some common risks by exploiting them. • A comprehensive survey on recent studies about XSS detection methods. |
| 5 | Detecting Web Attacks with End-to-End Deep Learning. Yao Pan, Fangzhou Sun, Jules White, Douglas C. Schmidt, Jacob Staples, and Lee Krause. (2018) | • The architecture and results of applying a unsupervised end-to-end deep learning approach to automatically detect attacks on web applications. |
| 6 | A Survey of Detection Methods for XSS Attacks.Upasana Sarmaha, D.K. Bhattacharyyaa, J.K. Kalitab. (2018) | • a comprehensive survey of XSS attack types, the preconditions to successfully launch an XSS attack, detection approaches and a list of tools that support detection of these attacks. |
| | | |

# Chapter 3

# System Architecture

## 3.1 System Architecture

The System Design and architecture gives the detail architecture for developing the proposed system.

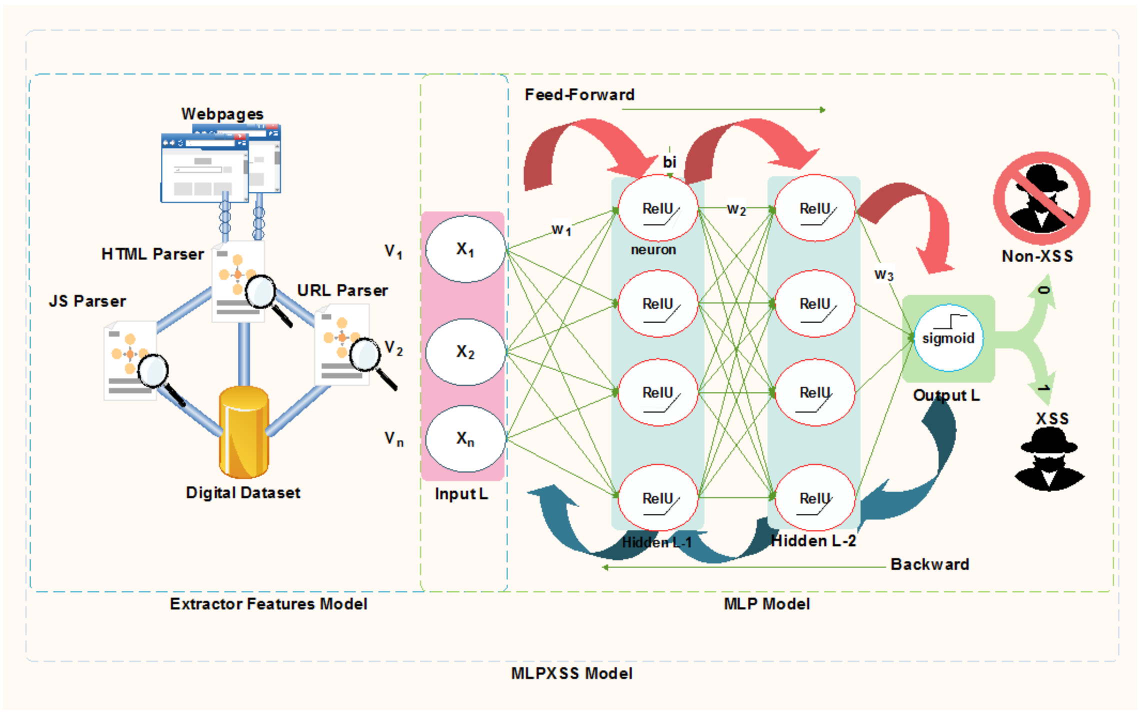


Figure 3.1: **System Architecture**

### 3.1.1 System Architecture Details

#### A. Features Identification

Features identification is an important step in the machine learning classification process. As the main classification ingredient, features correctly separate XSS-infected samples from the non-infected ones. These features are extracted from URLs,webpage content, and SNSs to create a feature vector, which is given as an input to the classifier.

#### B. Collecting Webpages

In this Step, a database is created by collecting malicious and benign webpages from various trusted Internet sources such as XSSed, Alexa and Elgg . The database consists of three kinds of webpages: benign webpages, malicious webpages containing obfuscated code, and SNSs webpages infected by XSS worm.

#### C. Features Extraction And Training Dataset Construction

This step is responsible for extracting and recording XSS features (identified in step 1) from each of the collected webpages. For features extraction, various tools are used. Each webpage is manually labeled as XSS or non-XSS based on the extracted features, and a training dataset is constructed.

#### D. Multilayer Perceptron Technique

The goal of this step is to categorize the webpages into XSS or Non-XSS. In other words, it is used to determine whether a webpage is infected with XSS or is legitimate. In order to achieve this objective, the training dataset (constructed in an earlier step) is supplied to the artificial neural network. That generates a predictive model that is further used to detect XSS-infected webpages.

# Chapter 4

# Domain Specific Information

## 4.1 What is cybersecurity?

Cyber security is the practice of defending computers, servers, mobile devices, electronic systems, networks, and data from malicious attacks. It's also known as information technology security or electronic information security. The term applies in a variety of contexts, from business to mobile computing, and can be divided into a few common categories.

- **Network security** is the practice of securing a computer network from intruders, whether targeted attackers or opportunistic malware.
- **Application security** focuses on keeping software and devices free of threats. A compromised application could provide access to the data it's designed to protect. Successful security begins in the design stage, well before a program or device is deployed.
- **Information security** protects the integrity and privacy of data, both in storage and in transit.
- **Operational security** includes the processes and decisions for handling and protecting data assets. The permissions users have when accessing a network and the procedures that determine how and where data may be stored or shared all fall under this umbrella.
- **Disaster recovery and business continuity** define how an organization responds to a cyber-security incident or any other event that causes the loss of operations or data. Disaster recovery policies dictate how the organization restores its operations and information to return to the same operating capacity as before the event. Business continuity is the plan

the organization falls back on while trying to operate without certain resources.

- **End-user education** addresses the most unpredictable cyber-security factor: people. Anyone can accidentally introduce a virus to an otherwise secure system by failing to follow good security practices. Teaching users to delete suspicious email attachments, not plug in unidentified USB drives, and various other important lessons is vital for the security of any organization.

## 4.2 Why is cybersecurity important?

In today's connected world, everyone benefits from advanced cyber defense programs. At an individual level, a cybersecurity attack can result in everything from identity theft, to extortion attempts, to the loss of important data like family photos. Everyone relies on critical infrastructure like power plants, hospitals, and financial service companies. Securing these and other organizations is essential to keeping our society functioning.

## 4.3 Types of cybersecurity threats:

- **Phishing**

  Phishing is the practice of sending fraudulent emails that resemble emails from reputable sources. The aim is to steal sensitive data like credit card numbers and login information. It's the most common type of cyber-attack. You can help protect yourself through education or a technology solution that filters malicious emails.

- **Ransomware**

  Ransomware is a type of malicious software. It is designed to extort money by blocking access to files or the computer system until the ransom is paid. Paying the ransom does not guarantee that the files will be recovered or the system restored.

- **Malware**

  Malware is a type of software designed to gain unauthorized access or to cause damage to a computer.

- **Social engineering**

Social engineering is a tactic that adversaries use to trick you into revealing sensitive information. They can solicit a monetary payment or gain access to your confidential data. Social engineering can be combined with any of the threats listed above to make you more likely to click on links, download malware, or trust a malicious source.

# Chapter 5

# Implementation Details

## 5.1 HTML-based Features Sub-Model

Three core features related to HTML are selected in this study, including tags, attributes, and events. This model is designed to extract features and JavaScript code from HTML raw files in combination with the html5lib parser. The html5lib parser gives us the optimal representation way and is extremely broad as it parses the pages in the same manner, as a web browser does. Furthermore, it also performs the HTML5 parsing algorithm, which is heavily impacted by modern browsers and based on the WHATWG HTML5 specification. Once the tree structure of HTML data for the webpage is created, the process of navigating and searching parse tree is required, i.e., tree traversal. For this task, a BeautifulSoup python library is used to create an object for each page which can be executed in parallel with html5lib. Since the object of this library contains all the data in a tree structure, the tags, attributes, and event of each webpage are programmatically extracted as shown in Algorithm 1.

### 5.1.1 Algorithm for Parsing HTML Documents

```
Input - set of web pages WP wp1,wp2...wpn
Output- Html-based Feature Vector (HFV)
    TG[ ] ⇐ list representing HTML tags
AT[ ] ⇐ list representing HTML attributes
EV[ ] ⇐ list representing HTML events
KE[ ] ⇐ list representing keywords_evil
P ⇐ Null
$H_{FV}$⇐ Null
for each $wp_i$ $\epsilon$ WP do // parsing each web page using html5lib and Beautiful-
```

```
Soup for tree traversal
P ⇐ parse_html $wp_i$ // Collect and count each TG,AT,EV,KE from each node
in tree
for each $node_i$ $\epsilon$ P do
for each $t_i$ $\epsilon$ TG do
H$_{FV}$[t$_i$]= $t_i$,$t_i$ in $node_i$
for each $a_i$ $\epsilon$ AT do
H$_{FV}$[a$_i$]= $a_i$,$a_i$ in $node_i$
for each $e_i$ $\epsilon$ EV do
H$_{FV}$[e$_i$]= $e_i$,$e_i$ in $node_i$
for each $t_i$ $\epsilon$ KE do
H$_{FV}$[k$_i$]= $k_i$,$k_i$ in $node_i$
end for
H$_{FV}$$hl = lenp$ // html length
end for
Returns features vector $H_{FV}$ for future model uses
```

## 5.2 Javascript-Based Features Sub-Model

The dynamically extracted code is distributed into a series of tokens and syntax tree using Esprima parser is produced, which acts in the same manner as JavaScript engine does. Hence, lexical and syntactical analysis of JavaScript code can be performed at the same time. However, it has been observed in some cases that the JavaScript used by XSS could be broken and cause parser stop. To deal with this, force a parser to continue parsing, and produce a syntax tree even the JavaScript does not represent a valid code. The parser is configured to a tolerant mode and set the tokens flag to be true in the configuration object to keep the tokens that found during the parsing process. To traversing the entire tree, the generator function was made to take Esprima node and yielding all child nodes at any level of the tree, which gives us the full ability to visit all branches of the tree. The complete processing of above steps are presented in the following Algorithm 2 which is used to get a JavaScript-based Feature vector (JSFV) for the future model.

### 5.2.1 Algorithm for Parsing Javascript Documents

```
Input - set of web pages WP wp1,wp2...wpn
Output- JS-based Feature Vector (JS-FV)
    DO[ ] ⇐ list representing domObjectss
```

```
JP[ ] ⇐ list representing JS properties
JM[ ] ⇐ list representing JS methods
TG[ ] ⇐ list representing HTML tags
EV[ ] ⇐ list representing HTML events
AT[ ] ⇐ list representing HTML attributes
P ⇐ Null
js ⇐ Null
JS_strings = []
JS$_{FV}$ ⇐ $\phi$
    for each $wp_i$ $\epsilon$ WP do
    P ⇐ parse_html $wp_i$ with P do
     for $t_i$ $\epsilon$ TG do
    // Extract JS code from script
    if ( t$_i$ $== scriptandAT[src] == false)then$
    js = t$_i.string$;
    if (js $\neq \phi$ ) then
    JS_strings.append(js)
    //Extract JS code from links
    if ( t$_i$ $== aandAT[href] == false)then$
    js = t$_i.string$;
    if (js $\neq \phi$ ) then
    JS_strings.append(js)
    //Extract JS code from form
    if ( t$_i$ $== formandAT[action] == false)then$
    js = t$_i.string$;
    if (js $\neq \phi$ ) then
    JS_strings.append(js)
    //Extract JS code from iframe
    if ( t$_i$ $== iframeandAT[src] == false)then$
    js = t$_i.string$;
    if (js $\neq \phi$ ) then
    JS_strings.append(js)
    //Extract JS code from frame
    if ( t$_i$ $== frameandAT[src] == false)then$
    js = t$_i.string$;
    if (js $\neq \phi$ ) then
    JS_strings.append(js)
    end for
end for
```

Returns features vector (JS$_{FV}$)$for futer model uses$

## 5.3 Modules in the System.

### 5.3.1 Collecting Raw Data

Building a new digital dataset of such attacks containing malicious and benign instances for training and testing a model is quite challenging. Therefore, to create a new dataset, one approach is to crawl all the web applications with different webpages. The size of the web applications precludes this strategy from being productive. Alternatively, one can crawl only a part of the web applications. However, crawling the web in some fixed, and settled manner is problematic because it potentially biases the dataset, which is not comprehensive nor miscellaneous, and the model could not generalize it. Besides, it is not an obvious way to argue that a sufficient subset of the websites is representative. For this, a novel approach is used to create a large real-world dataset and to make it comprehensive and miscellaneous to tackle the issues concerned with it.

### 5.3.2 Dynamic Feature Extraction Model

In several cases of Natural Language Processing using deep neural network technique, the text samples are represented in the form of word vectors, but in case of an XSS attack, malicious code is often a JavaScript code. Thus, JavaScript has non-standard encoding; there exist many senseless strings and Unicode symbols. Additionally, JavaScript uses data encapsulation, code rearrangement, rubbish strings insertion, and alternative techniques where the word vectors lead to generating large dimensions is the very time-consuming case of XSS. This research introduces a novel dynamic feature extraction model which is introduced to get proper feature vectors that characterize the XSS anomaly.

### 5.3.3 Artificial Neural Network Model

Artificial Neural Network (ANN) is inspired by a human brain operation . It is composed of many layers including, the input layer, the hidden layer(s) and the output layer. In this research, an ANN-based multilayer perceptron (MLP) algorithm which characterized by the dynamic features of XSS-based attack detection is used.

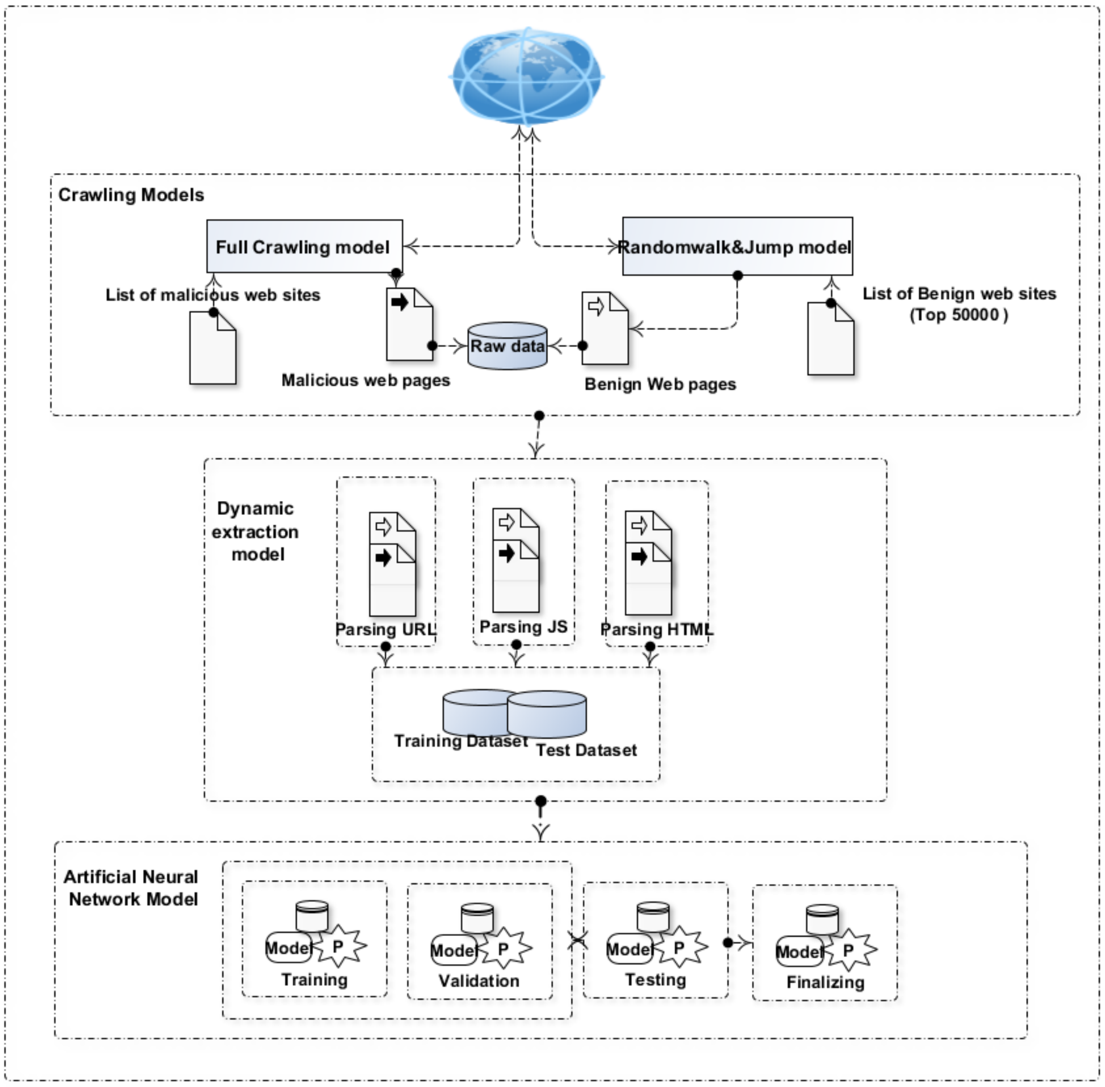


Figure 5.1: **The Framework of Proposed System**

## 5.4 Flowchart

This system consists of six steps viz. Features Identification, Collect Webpages, Feature Extraction, Training Dataset, Multilayer Perceptron Technique and Classification is done to categorize the webpages into XSS or Non-XSS.

### 5.4.1 Features Identification

Feature identification is an important step in the machine learning classification process. As the mainclassification ingredient, features correctly separate XSS-infected samples from the non-infected ones. These features are extracted from URLs, webpage content, and SNSs to create a feature vector, which is given as an input to the classifier.

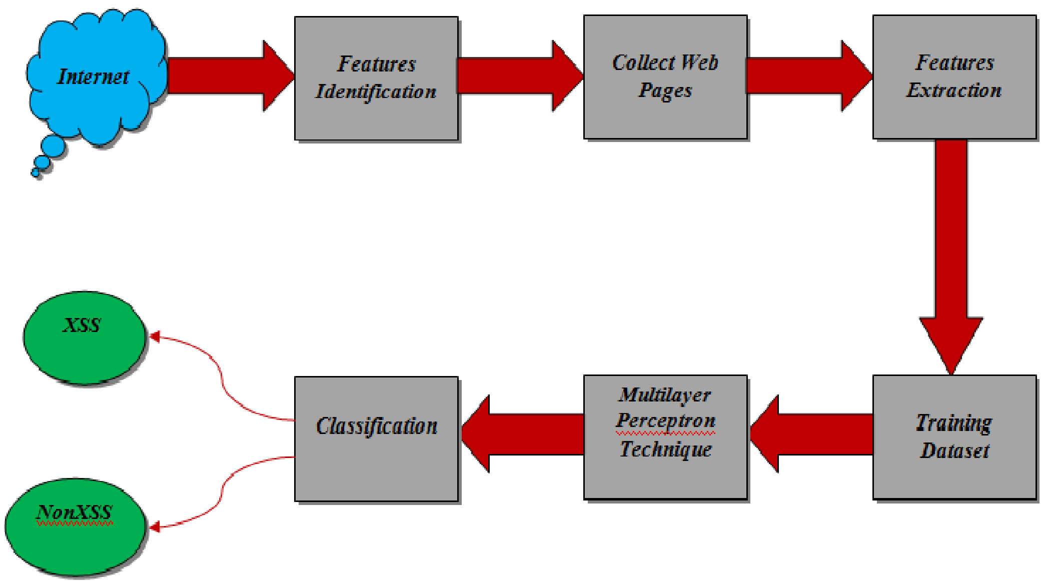


Figure 5.2: **Flowchart**

### 5.4.2 Collecting Webpages

In this Step, a database is created by collecting malicious and benign webpages from various trustedInternet sources such as XSSed, Alexa and Elgg. The database consists of three kinds ofwebpages: benign webpages, malicious webpages containing obfuscated code, and SNSs webpages infected by XSS worm.

### 5.4.3 Features Extraction and Training Dataset Construction

This step is responsible for extracting and recording XSS features (identified in step 1) from each of the collected webpages. For features extraction, various tools are used. Each webpage is manually labeled as XSS or non-XSS based on the extracted features, and a training dataset is constructed.

### 5.4.4 Multilayer Perceptron Technique

The goal of this step is to categorize the webpages into XSS or Non-XSS. In other words, it is used to determine whether a webpage is infected with

XSS or is legitimate. In orderto achieve this objective, the training dataset (constructed in an earlier step) is supplied to the artificial neural network. That generates a predictive model that is further used to detect XSS-infected webpages.

# Chapter 6

# Result Analysis

The results obtained on held-out test dataset achieved 99.32 % of accuracy, as shown in Fig.1 . While the precision is 99.21 % for XSS class and detection rate up to 98.35%. The false positive rate is tend-to-zero, which represent 0.31 %. We obtained the global value of quality for the complete taxonomy of our model using the weighted averages (micro average), and macro averages of both classes. Table 1 shows the full report generated based on the confusion matrix for testing model performance.

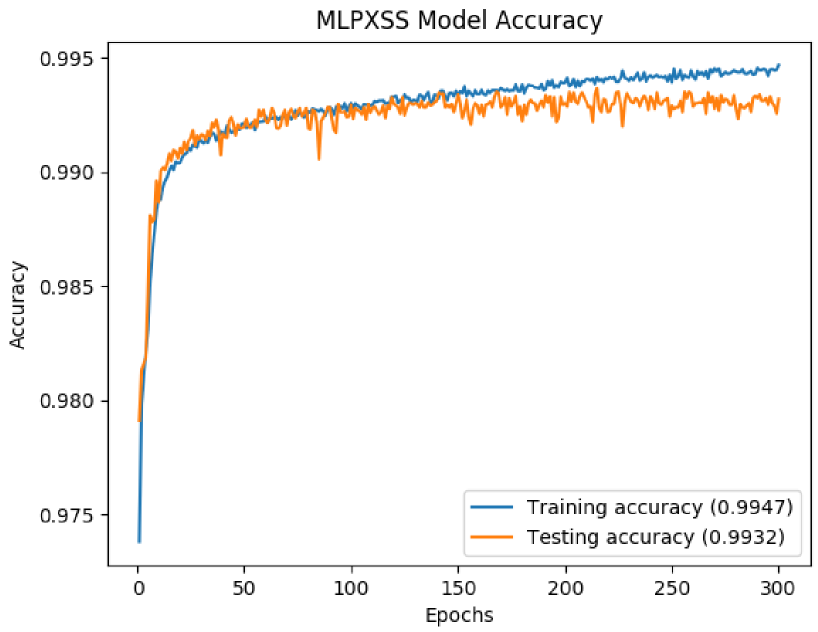


**Figure 6.1: Model Accuracy On Training and Testing Dataset**

For the strict verification of data quality, a new experiment is implemented using the same dataset with different machine learning algorithms such as nonlinear SVM Radial Basis Function (RBF) kernel with c value =1.0, k-nearest neighbor (k-NN) with neighbors =5 and ensemble method AdaBoost

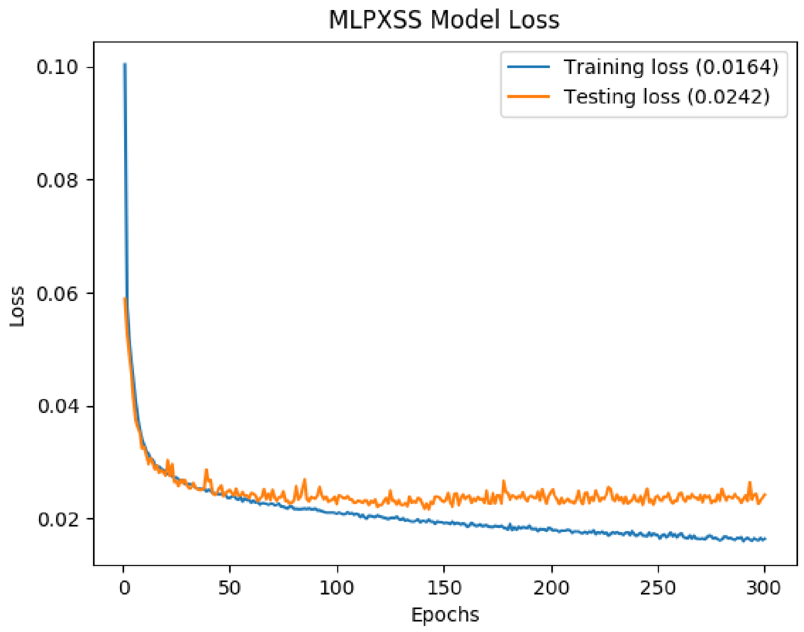


**Figure 6.2: Model Log Loss Values Over Time**

classifier along with proposed methodology. The goal is to verify whether the dataset employed by the proposed method has advantages and able to works on different techniques as well as to compare MPLXSS with different algorithms. The results of the various algorithms proved the efficiency and effectiveness of the proposed mechanism through which the data collection and extraction were obtained as shown in Table 6.1 and Table 6.2. Also, the results of the AdaBoost algorithm are approximate to the results of neural networks model in terms of accuracy and less than in terms of detection rate. Therefore, we can conclude that data quality and features extraction are the key components for precision and robustness of these classifiers since most of them are performing well on the same data set.

Table 6.1: Result Comparison MLPXSS With Other Classifiers

| **Classifier** | **Accuracy** | **FP Rate** | **Misclassification Rate** | **ROC** |
|---|---|---|---|---|
| MLPXSS | 0.9932 | 0.0031 | 0.0068 | 0.9902 |
| Gausian NB | 0.9716 | 0.0148 | 0.0284 | 0.9609 |
| SVM | 0.9798 | 0.0005 | 0.0202 | 0.9644 |
| K-NN | 0.9884 | 0.0043 | 0.0116 | 0.9826 |
| AdaBoost | 0.9921 | 0.0030 | 0.0079 | 0.9882 |

# Chapter 7

# Future Scope

Combined traditional approaches can detect XSS attacks with a higher accuracy.If combined the traditional method of using WAF with ML to detect XSS attacks in web applications.

The future scope of this approach is that ML approaches can be combined with other traditional approach like static analysis, dynamic and hybrid analysis to detect and prevent XSS attacks in web applications.

# Chapter 8

# Conclusion

The ANN-based scheme is used to detect the XSS-based web-applications attack. Three models are designed in a novel manner. The first model is concerned with the quality of the raw data and random crawling. The second model deals with extracting digital data as features of raw data and providing neural networks with these digital features, and the third is ANN-based multilayer perceptron model that takes the digital data, processes and classifies the final prediction result of XSS-attack problem.

This model performs a prediction of security threats such as XSS attack, which can reflect in the form of a warning to users who can cancel the subsequent treatment of the pages. It acts as a security layer either for the client-side or the server-side.

In our future work, we will improve and apply this scheme to detect XSS attacks in the real-time detection system.

# References

[1] XSSClassifier: Shailendra Rathore, Pradip Kumar Sharma, Jong Hyuk Park, "An Efficient XSS Attack Detection Approach Based on Machine Learning Classifier on SNSs," Journal of Information Processing Systems Vol. 13, No. 4, pp. 1014-1028, Aug. 2017 10.3745/JIPS.03.0079

[2] Ibéria Medeiros, Nuno Neves and Miguel Correia , “Detecting and Removing Web Application Vulnerabilities with Static Analysis and Data Mining ,” UID/CEC/00408/2016

[3] Yao Pan, Fangzhou Sun, Jules White, Douglas C. Schmidt, Jacob Staples, and Lee Krause, “Detecting Web Attacks with End-to-End Deep Learning,” in Vanderbilt University, Melbourne, FL, USA,Eighth International Conference on. IEEE, 2018, pp. 453– 455

[4] Miao Liu, Boyu Zhang, Wenbin Chen and Xunlai Zhang, “A survey of exploitation and detection methods of XSS vulnerabilities,"IEEE, 2017.

[5] Upasana Sarmaha, D.K. Bhattacharyyaa, J.K. Kalitab, “A Survey of Detection Methods for XSS Attacks,” IEEE Access, 2018

[6] Jinkun Pan and Xiaoguang Mao, “Detecting DOM-Sourced Cross-Site Scripting in Browser Extensions,” in Proceedings of the 17th IEEE/ACM, 2017. International Symposium on Cluster, Cloud and Grid Computing, ser. CCGrid 17. Piscataway, NJ, USA: IEEE Press, 2017, pp. 458–467. [Online]. Available: https://doi.org/10.1109/CCGRID.2017.111

[7] Fawaz Mahiuob Mohammed Mokbal1, Wang Dan1, Azhar Imran2, Lin Jiuchuan3 , Faheem Akhtar2,4,and Wang Xiaoxi5,"MLPXSS: An Integrated XSS-Based Attack Detection Scheme in Web Applications Using Multilayer Perceptron Technique,"Digital Object Identifier 10.1109/ACCESS.2019.Doi Number