IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Xss Attack Detection Using Machine Learning – Threatlens

[1]Prof. N.K. Patil, [2]Ms. Shubhanu Muthukumar, [3]Ms. Sheetal Surwase, [4]Ms. Tanvi Shetty, [5]Ms. Prajakta Pandit

[1]Lecturer, Department of Computer Engineering, Sinhgad Institute of Technology, Lonavala, Maharashtra, India

[2,3,4,5]Student, Department of Computer Engineering, Sinhgad Institute of Technology, Lonavala, Maharashtra, India

Abstract: XSS or Cross-site Scripting is one of the most damaging attacks done to web applications security wise, since it lets assailants insert harmful scripts to webpages which users can easily access. Moreover, these attacks can result in session hijacking, account takeover, and data breach. Detection using rules is a common technique; however, XSS detection is too complex for this static approach due to its evolving and concealed nature. More flexible methods are required. The contribution of this work is in employing machine learning algorithms (ML) to improve detection systems of XSS attacks while enhancing the accuracy and lowering false positives when compared to previous techniques. By utilizing supervised learning, our model is trained on an exhaustive dataset containing both malicious and benign labels making it capable of pattern detection and classification at suspicious levels. We study other ML techniques such as Support Vector Machines, Random Forest, and adaptable Deep learning and analyze them on their precision, recall and overall runtime efficiency. This project proposes a framework of machine learning algorithms that allows for the real-time detection and mitigation of XSS attacks. The proposed approach is based on the supervised learning techniques that contain pre-labeled data, which makes it feasible to distinguish between valid and malicious HTTP requests.

Keywords - XSS, benign, Machine learning, Support Vector Machines, Random Forest, Deep learning

I. INTRODUCTION

The priority of web security significantly increased due to over-dependence on web applications. A frequently occurring risk found within websites includes Cross Site Scripting (XSS), which permits attackers to load harmful scripts into safe websites. Scripts like these can be executed in the browsers of other users, leading to data theft, session hijacking, or other malicious activities. Considering the impacts XSS attacks can have, it is imperative to at the very least detect the presence of vulnerabilities and safeguard the web ecosystem throughout. Many components of web identity protection systems and service-oriented frameworks face immense problems dealing with the detection of Cross Site Scripting (XSS) attacks considering how multi-faceted and dynamic the threats posed are. Common methods of detection, especially signature-based ones, systematically fail to keep pace with the innovative and adaptable malicious intent of these hackers who use the simplest changes to unnoticeable scripts. A situation like this could expose web applications to attacks that breach security frameworks, enabling access to sensitive user information and data which, if mishandled, could result in disastrous outcomes such as identity theft or defacement of the website. There is a need for these types of flexible detection solutions. The development of these solutions can be simplified through machine learning (ML).

II. LITERATURE SURVEY

1.Detecting of Cross-Site Scripting (XSS) attacks using machine learning algorithms: a review

This study has the purpose of reviewing the reasons behind web attacks and analyzing existing research to design an optimized solution for the detection of XSS attacks. This review analyzes the recent developments concerning the detection of XSS attacks. Kaur, Garg and Bathla 2023 recounted why traditional approaches fail to mitigate XSS attacks. Our survey answers the questions framing detection and prevention of XSS attacks through Machine Learning. We assist decision makers in strengthening their approach to defending against vulnerabilities by outlining advanced techniques for vulnerability analysis compared to traditional attack detection techniques. Using machine learning techniques leads to greater detection accuracy for XSS attacks than with other approaches. This review addresses supervised, unsupervised, and semi-supervised techniques with their respective advantages and disadvantages.

2.AlgoXSSF: Detection and analysis of Cross site request forgery (XSRF) and cross site scripting (XSS) attacks via Machine learning

Following the work of Naresh Kshetri, Dilip Kumar, James Hutson, Navneet Kaur, and Omar Faruq Osama in 2024, this approach uses machine learning techniques to autonomously detect patterns of CSRF and XSS attacks for early detection and prevention. In the beginning, a web log dataset is gathered which consists of web requests that are both benign and malicious. A feature extraction step is performed that deals with relevant characteristics like request headers, tokens, and even JavaScript code which need to be included. These characteristics are what enable the machine learning algorithms to classify CSRF and XSS attacks as either benign or malicious. Any combination of algorithms are appropriate such as supervised learning classifiers like Support Vector Machine (SVM), Decision Trees, or even the Neural Networks Algorithm . The algorithm is trained using the labeled dataset, to understand and generalize the patterns that are present both in benign and malicious instances. There are some evaluation metrics that can be put in place such as the model's recall, precision, and F1 score for determining the model’s accuracy and robustness.

3.Detecting Cross Site Scripting Attack using Machine learning Algorithms

XSS attack detection is done through machine learning algorithms in the step-wise fashion as described by S. Karthika, G. Padmavathi, A. Roshni and S. Varshini (2024) which includes: Data Collection, Pre-processing, feature selection, model building and training, model evaluation, and deployment.

4.Cross Site Scripting Attack Detection in Web using Machine learning algorithm

According to Harishkumar S, Manikandan P. A, Mohana Sundaram K, V. P. Dhivya (2021), the dataset concerning the XSS attack is first loaded into our database. After loading, the database undergoes a pre-processing stage where irrelevant or noisy data is eliminated from the loaded dataset. The data set consists of different fields, among which only relevant ones are chosen. Once feature extraction is done data is classified by the CNN classification algorithm. This ensures that accurate detection of injected scripts being XSS attacks is detected during the final stage.

5.XSS detection using hybrid machine learning methods

According to Nagham Kamil Albusalih, Rana Jumaa Aljanabi (2022), “The proposed system consists of three stages.” First, the pre-processing stage has more than one step which involves preparing a large number of instances and features in the dataset for the subsequent stages. Second, the features selection (stage of constructing the feature vector is embedded in the next stage) In this work, one technique is applied for selecting features, RF. Third, the classification stage is for classifying the features with the NB classifier.

III. METHODOLOGY

The primary goal of this system is to create a model that can detect Cross-Site Scripting (XSS) by using a Machine Learning technique. The project adopts a multi-tier architecture with a frontend developed in WordPress and hosted on InfinityFree. The backend is developed locally using Python and Flask. The model of Machine Learning that has been used in this system is capable of distinguishing between benign inputs and malicious inputs (XSS). The methodology is predominantly focused on data collection, model training, backend integration, and communication between the WordPress frontend and Flask backend.

1. Data Gathering and Preparation

The dataset that is required should contain both attack payloads as well as benign inputs for usage and these should also be labeled. The dataset is curated from open-source repositories like OWASP, Kaggle, and GitHub, and covers common attack vectors such as <script>alert("XSS")</script> and onerror=alert(1),). A value of 1 is assigned to inputs considered malicious and 0 to safe inputs. Checking and cleansing the data will ensure that it is complete and of good quality and this is termed preprocessing. HTML entities are decoded, special characters are tokenized, and lowercase transformation is applied. This allows the model to concentrate on identifying patterns within the data rather than being sidetracked by differences in formatting or case sensitivity.

2. Feature Extraction

As with any other form of textual data, feature extraction will be done to transform the text into a numerical form amenable to machine learning algorithms. In the project at hand, we employ vectorization using TF-IDF (Term Frequency-Inverse Document Frequency) with character-level n-grams (3-grams) to extract short subsequences of suspicious patterns that tend to occur with XSS payloads. This ensures that attacks can still be recognized even when slight changes are made to the input.

3. Model Training

Then, a supervised learning algorithm is applied to the processed and vectorized dataset. We selected Logistic Regression because of its straightforward implementation and reasonable performance on text classification problems. The model is trained on 80% of the data and validated on the remaining 20% with accuracy, precision, recall, F1-score, and other metrics. The obtained model is stored using joblib (or pickle module) so it can be readily accessed from within the Flask application.

4.Integrating Flask with Backend Services

With regard to the backend, the Flask application hosts the XSS detection model and functions as an interface. A REST API is built with Flask, which contains the POST request that is sent to the /predict endpoint containing user input. The API fetches the pre-trained model, cleans the given input the same way it was done during the training phase, performs feature extraction, and runs the model against the provided data. The model's response as to whether or not the given data is safe or likely a XSS attack, is relayed back in JSON format.

5.Communication with WordPress on the Frontend

The project's frontend is built on WordPress and hosted on InfinityFree. Regarding the Flask backend, JavaScript sends AJAX requests to the local Flask server. On form submission or input on the WordPress site, a JavaScript function captures the input and makes a POST request to the Flask API. The response generated by the Flask model is rendered on the frontend, which either makes it possible to submit the form or issue a warning that the form contains a potential XSS attack.

6. Testing and Local Deployment

As the Flask backend runs locally, the system is validated on a local development environment. Ngrok or other tunneling utilities may be used to temporarily expose the local Flask server externally over the internet for testing interaction with the live WordPress site. Comprehensive testing is conducted using known XSS payloads and regular inputs to ensure the correctness of the system in real-time.

7. Results Management and Feedback

Depending on the prediction obtained from the Flask backend, either form submission continues or an error message is returned to the user advising that the input may be dangerous. The feedback mechanism prevents malicious scripts from being saved or executed on the website and thus protects users and site integrity.

IV.CONCLUSION

The detection of XSS attacks through machine learning provides an adaptable answer to an archaic problem in website security. Using machine learning's advanced model techniques to recognize patterns, developers and security specialists can better defend web applications against both previously documented and novel XSS attacks. The end model yielded an accuracy of 99 %, which means it appropriately classified 99 out of 100 inputs given to it. Other notable performance metrics were 98.7% precision and 99.3% recall, with F1-score being 99%, signifying that the model maintained equilibrium while malicious input detection, without excessively high false positive or negative rates. This project offers a scalable and smarter solution to XSS detection by using machine learning on a WordPress frontend with Flask backend. The system uses the capabilities of ML to identify both documented and concealed attack patterns. Without further intervention, the solution has the potential to adapt to new threats and strengthen security measures on web applications through ongoing model retraining with fresh data.

Threatlens's machine learning XSS detection website is illustrated in Figure 1.

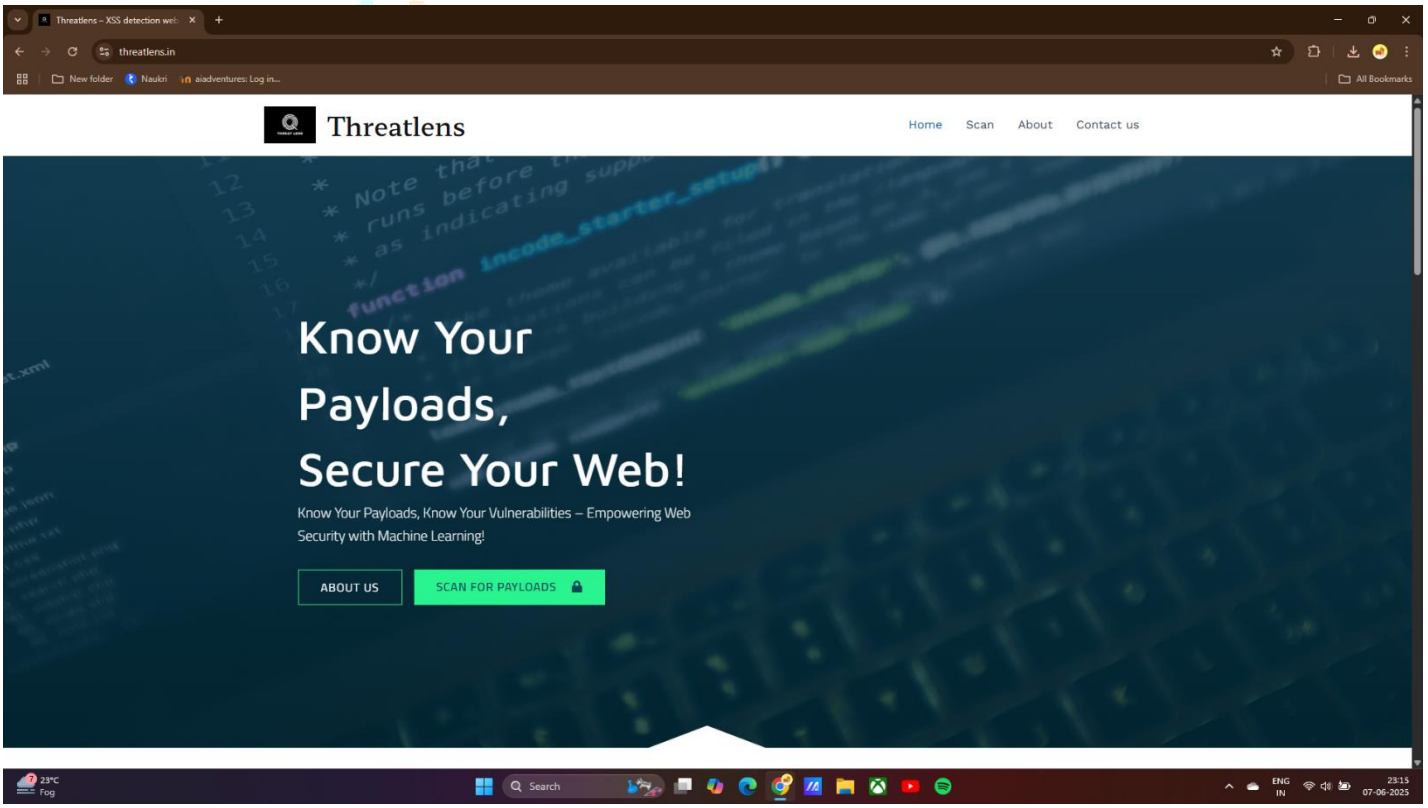

Figure 1: Threatlens Homepage

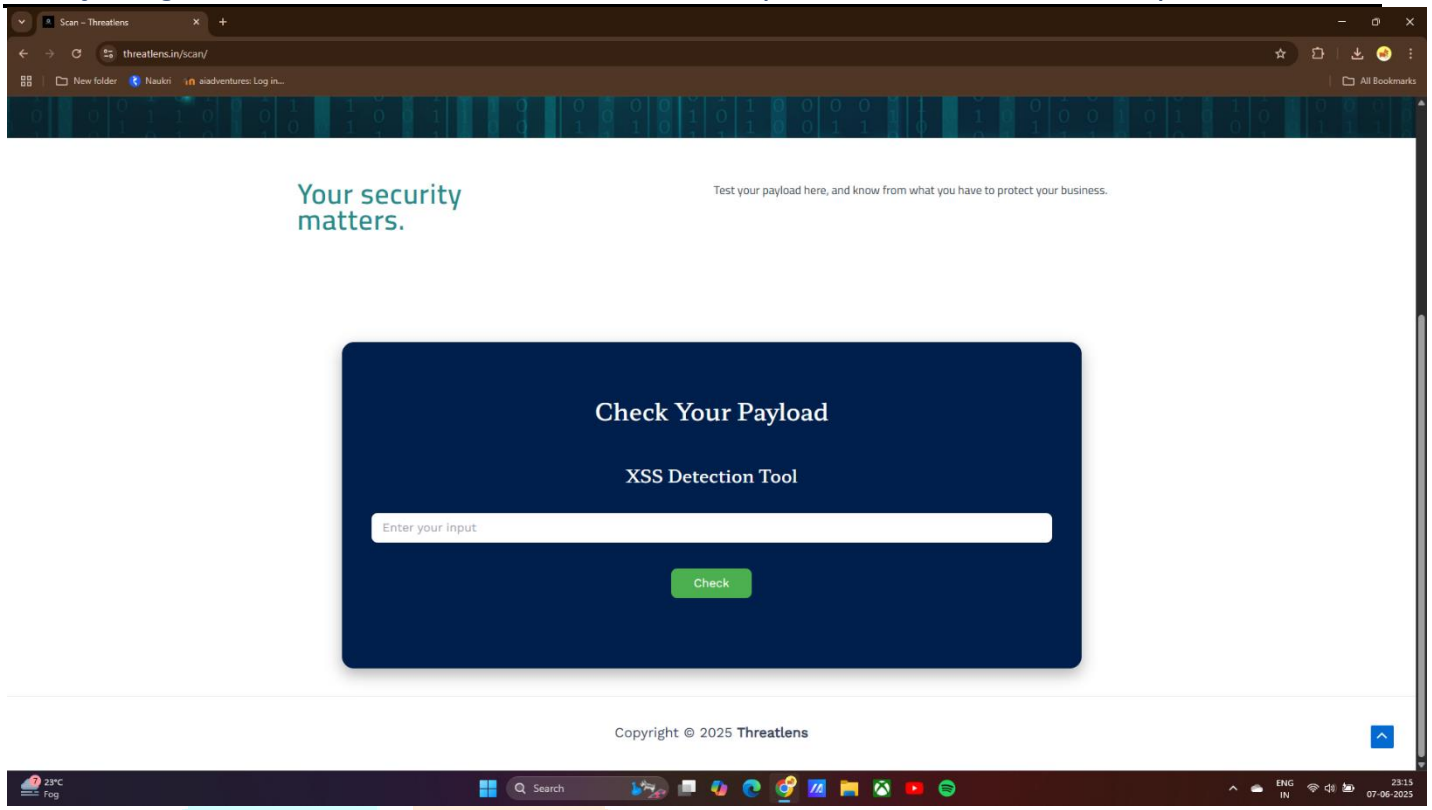

Figure 2: Threatlens Scan page

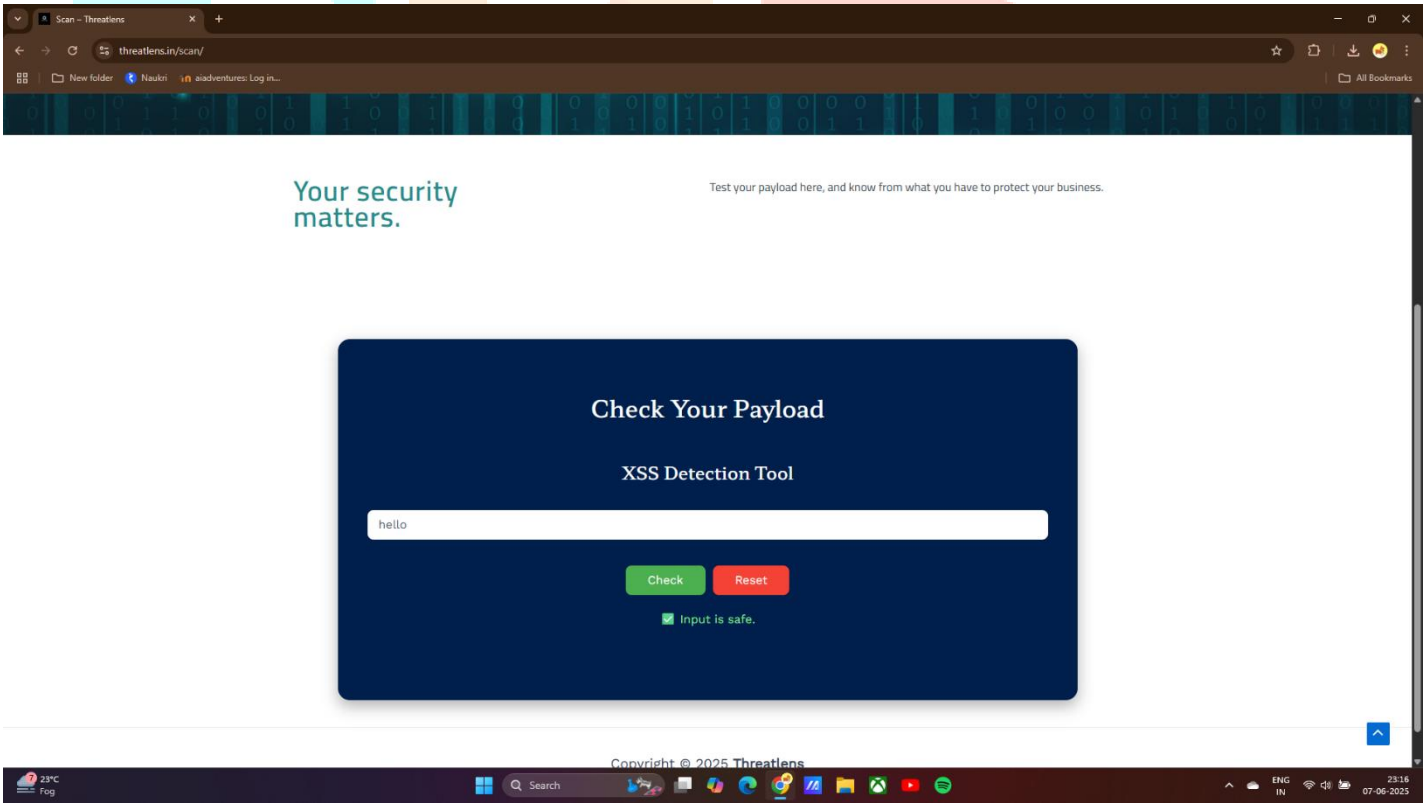

Figure 3: Input payload is safe

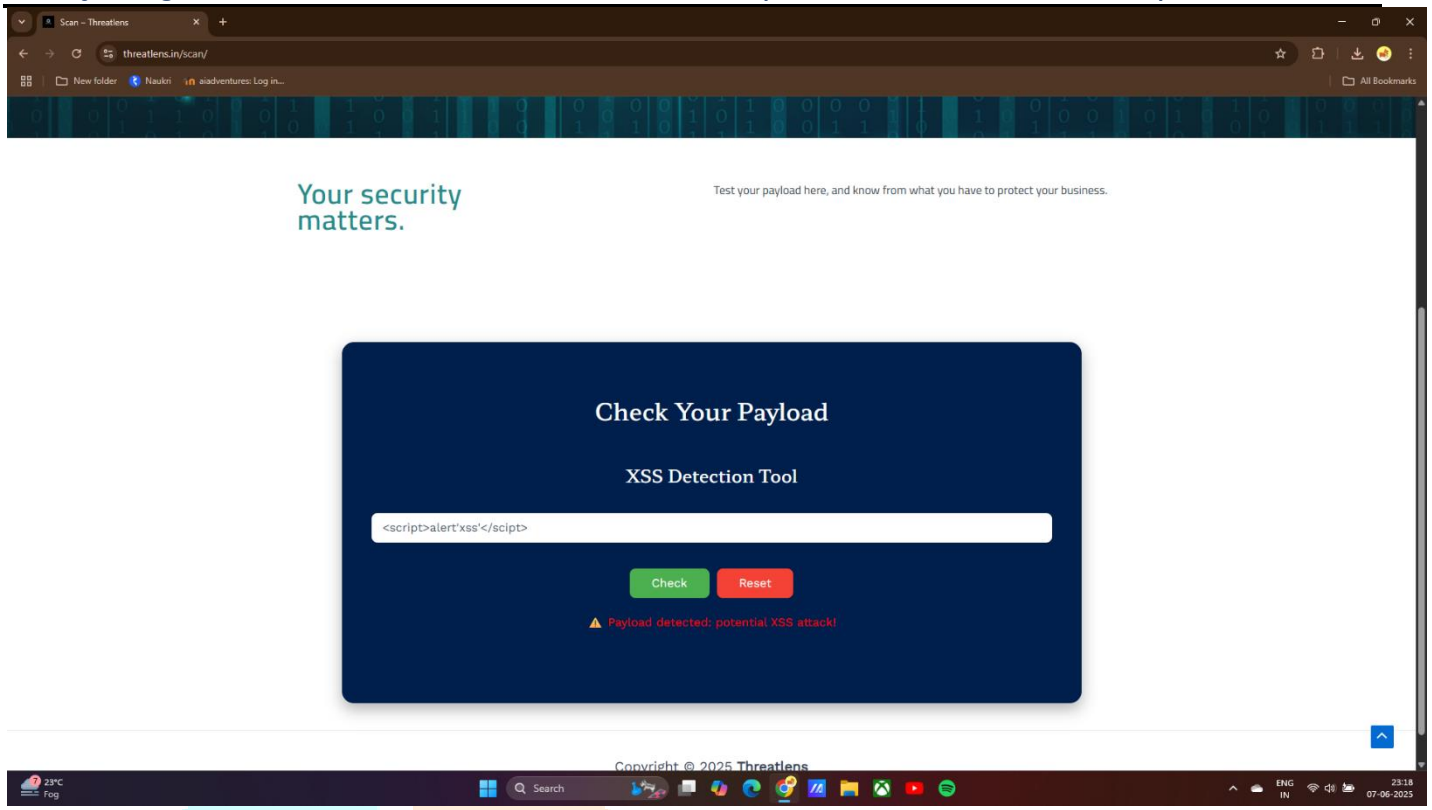

Figure 4: Input payload is detected

V. FUTURE SCOPE

The project done in this paper may be furthered within the scope of context augmentation and finding new features to add to the trained model's existing dataset. In this research, a single machine learning model was used for analysis. It can be done with other models, so it is more than reproducible. Since databases containing information on phishing and spam are publicly accessible, the same can be done for labeling phishing and spam datasets in order to improve the current dataset's adaptability to different attack types. As data will be supplied upon request. Later, we plan to incorporate more sophisticated transfer learning techniques. Moreover, our suggested model will be integrated into a real-time system as a web-based XSS attack detector.

Because the majority of known solutions ignore the application of new XSS attack vectors and payloads that could be detected by a proper machine learning model, it is evident that XSS attack detection models are also susceptible to exploitation. Consequently, that model must recalculate with proper learner algorithms and models and new advanced techniques. Also, we will augment the dataset that we have by obtaining more web scripts. Additionally, application of new technology is an urgent requirement to respond to such types of information system offensive maneuvers. In order to reach optimal precision, there has to be selected features that must be chosen based on some algorithm that should be selected.

VI. ACKNOWLEDGEMENT:

The authors are grateful to Prof. N.K. Patil, Prof. Karim Mulani, and Dr. Shubhangi Patil for their valuable guidance and ever-ready support during the course of this project. The authors are also grateful to the Department of Computer Engineering, Sinhgad Institute of Technology, for furnishing the required facilities and environment for the success of this work.

VII. REFERENCES:

[1] Jasleen Kaur, Urvashi Garg, Gourav Bathla(2023) :'Detection of Cross-Site Scripting(XSS) attacks using machine learning algorithms: a review' https://www.researchgate.net/publication/369476572

[2] Naresh Kshetri, Dilip Kumar, James Hutson, Navneet Kaur, Omar Faruq Osama(2024): "algoXSSF: Detection and analysis of cross-site request forgery (XSRF) and cross-site scripting (XSS) attacks via Machine learning algorithms" https://doi.org/10.48550/arXiv.2402.01012

[3] S. Karthika, G. Padmavathi, A. Roshni and S. Varshini (2024): "Detecting Cross-Site Scripting Attack using Machine Learning Algorithms,"- https://ieeexplore.ieee.org/document/10499119

[4] Nagham Kamil Albusalih, Rana Jumaa Aljanabi (2022): "XSS detection using hybrid machine learning methods."