SECUREWEB: A NOVEL MACHINE LEARNING METHODOLOGY FOR IDENTIFYING CSRF VULNERABILITIES

Padakanti Krishna Teja[1], Banoth Aryan[2],Chelimila Nithesh[3], Mr. G. Santosh Kumar[4],Dr.M.L.M. Prasad[5]

[123]UG Student, Department of Computer Science and Engineering - AI&ML, Joginpally B R Engineering College

[4]Assistant Professor, M.Tech (CSE), Department of Computer Science and Engineering AI&ML, Joginpally B R Engineering College

Email: Santhoshp1608@Gmail.Com

[5]Associate Professor,PhD,Department of Computer Science and Engineering - AI&ML, Joginpally B R Engineering College

Email: mlm.prasad@yahoo.com

Abstract :

Cross-Site Request Forgery (CSRF) is a severe web application vulnerability by which attackers can force an unthinking authenticated victim to perform some unwanted actions. Classic methods recognize CSRF vulnerabilities either from source code or through heavy manual work and are hence unfavorable for the scale and real-time detection. To tackle these issues, we present SECUREWEB, a Machine-Learning-based framework designed to identify CSRF vulnerabilities automatically using black-box scanning. SECUREWEB houses two simple-to-use modules for administrators and end-users, allowing URL scanning, machine learning model training, and on-the-fly classification of vulnerabilities. Under the hood, the engine Mitch extracts essential features from HTTP requests, such as request methods, the presence of anti-CSRF tokens, header information, and so forth, and runs Bayesian-supervised dictation with Random Forest, Decision Tree, SVM, and NaïveBayes. This system has been implemented in Python, with Django acting as the backend, and MySQL utilized as the data store, which provides a smooth and reactive user experience. Experimental results show 35 new CSRF vulnerabilities were found by SECUREWEB in major websites and additional three in production software. It performs well in terms of accuracy and other classification metrics in different testing scenarios. SECUREWEB appears thus as a scalable, automated, and intelligent means of protecting web applications from CSRF attacks, even when the source code is not available.

Keywords : Machine Learning, Web Security, Vulnerability Detection, Black-Box Scanning, HTTP Request Analysis, Random Forest.

I . INTRODUCTION

Nowadays, web applications are at the heart of the workings of enterprises in varied sectors from banking to healthcare. However, with an increase in usage of these applications, the threat landscape also grows. Among the most notorious web vulnerabilities is Cross-Site Request Forgery (CSRF), in which unauthorized commands are transmitted from a user that the web application trusts. CSRF attacks, depending on their type, may result in data breaches, illicit transactions, and erosion of trust from end users.

Usually, traditional CSRF detect applications static source-code analysis or manual testing or depending upon web-application firewalls. These traditional ways do pose scarcities in time and much need to source code, while at times, defendants aim to mess with their method of analysis by third-party applications. Hence, it is quite difficult for these traditional techniques to keep up with the fast, highly dynamic, and complex nature of modern web architectures. Thus, there is a growing need for an automatic intelligent and scalable detection mechanism that can work independently, without access to an application's internal codebase.

SECUREWEB is a novel solution using machine learning to detect CSRF vulnerabilities with a black box approach. It checks HTTP GET and POST request patterns, token presence, and header information for identifying risky situations. The system has a central component called the Mitch Engine, which provides support for various ML algorithms like Random Forest, SVM, Decision Tree, and Naïve Bayes for classification.

With easy-to-use interfaces for administrators as well as normal users, SECUREWEB is the URL scanning, ML training, and real-time vulnerability reporting system. It is developed in Python using Django and has MySQL as the back-end database, ensuring scalability, full user-friendliness, and performance.This kind of intelligence-based and automated system removes dependency on source code, allowing an end-user to detect a vulnerability with just little technical know-how. The USE of ML with relevant web security needs presented by SECUREWEB is a big step forward towards a proactive cybersecurity approach.

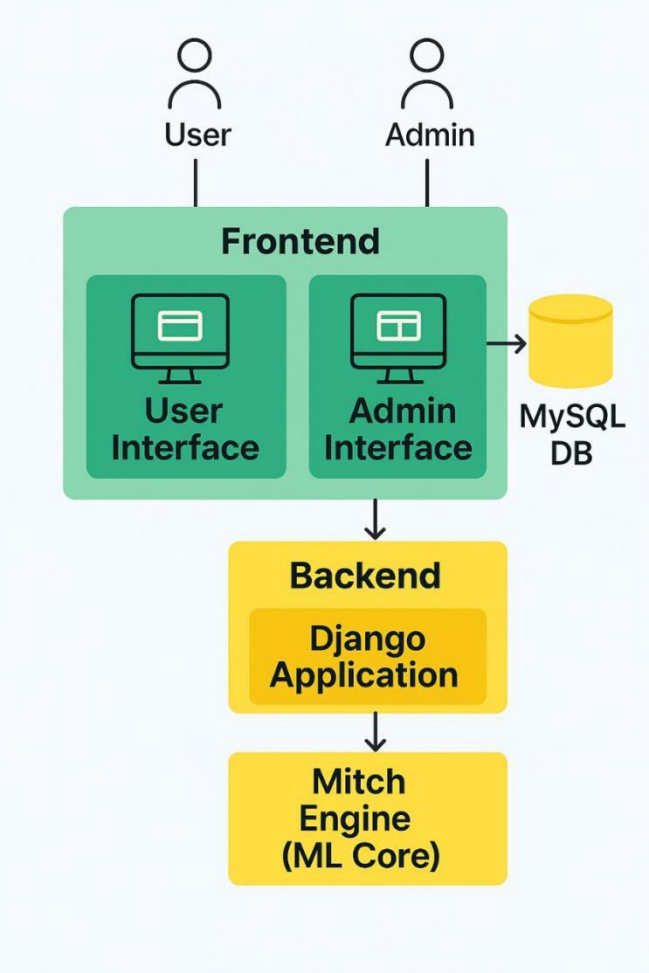

Fig 1: Proposed system Architecture

II. LITERATURE SURVEY

Recent advancements in cybersecurity have gradually shifted towards the use of machine learning methods for preventing and detecting web vulnerabilities, including CSRF attacks and phishing threats. Shahid (2023) gives a sweeping study on the applications of machine learning models in the identification and prevention of web vulnerabilities and attacks. The study reinforces the detection of anomalies and malicious behavior in web traffic using both supervised and unsupervised learning algorithms and opens a promising welcome for automated web security monitoring.

Phishing is still among the prime threats to online security, and to improve the real-time capability to detect phishing attacks, Lin et al. (2021) proposed Phishpedia: a hybrid deep learning framework designed to visually detect phishing webpages. By spearing convolutional neural networks alongside other deep learning architectures, their model searches webpage layout and visual cues to distinguish phishing sites from legitimate ones.

In the field of wireless communication security, Greco et al. (2021) dealt with jamming attacks targeting drone networks. Their methodology uses multi-layer perceptrons and decision trees to mark the existence of jamming signals at the network edge, which serves as a good hint in showing how the synergy between a suite of machine learning classifiers can be exploited to maximize the accuracy of detection and system's resistance to an attack aimed at disrupting communication channels.

Specifically in trying to prevent CSRF attacks, Lalia and Moustafa (2019) implemented a web browser extension to prevent CSRF vulnerabilities by monitoring and filtering malicious cross-site requests in real time. This client-side protection provides an extra layer of protection over the web without any need to change the server side, thus offering a straightforward way for an end-user to improve web security.

Calzavara et al. (2019) also proposed Mitch, a machine learning black-box detection method for CSRF vulnerabilities. Their approach extracts features from web application behaviors and classifies such behaviors to automatically detect potential CSRF threats. This study thus suggests that machine learning facilitates the detection of vulnerabilities in cases where internal details of the application may not be accessible.

Hence, these works exemplify the growing trend in cyber defense towards leveraging machine learning for providing automated, scalable, and effective solutions to address increasingly convoluted web-based challenges.

III. PROPOSED WORK

The work proposed here is a study of SECUREWEB, an intelligent and automated system to detect cross-site request forgery (CSRF) vulnerabilities, embracing machine learning techniques with a black-box scanning approach. The SECUREWEB system does not require access to source code or manual inspection, instead acting

from the outside, analyzing HTTP requests for suspicious patterns. Mitch Engine constitutes the machine learning spine of the system and extracts crucial features from HTTP GET and POST requests in the form of the presence of anti-CSRF tokens, request methods, and the referrer or origin header. These features are then used to train supervised learning algorithms such as Random Forest, Decision Tree, Support Vector Machine (SVM), and Naïve Bayes to infer whether incoming requests are vulnerable or secure. This framework is composed of two major modules: the Admin Module, which facilitates the administrators in managing user accounts, viewing scan logs, and viewing vulnerability reports with URL and timestamp information; and the User Module, which enables users to register, enter URL targets, choose scan depth, run the Mitch Engine, and train models while viewing performance statistics such as accuracy, precision, recall, and F1 score.It is developed with a sturdy software stack consisting of Python with Django on the backend, MySQL for database storage, and frontend-wise HTML5, CSS3, Bootstrap, and JavaScript for responsive behavior. SECUREWEB facilitates building a scalable and practical mechanism for securing web applications, so even a layman can detect CSRF vulnerabilities effectively while allowing the security experts to perform full management and reporting.

IV. METHODOLOGY

Data Collection

Users submit the target URLs and specify the crawl depth to the system. The crawling engine traverses website pages up to specified depth and dumps all raw HTTP GET and POST requests issued. These form the set of inputs on which the vulnerability analysis will be performed.

Feature Extraction and Preprocessing

Captured HTTP requests undergo preprocessing steps to extract features important in detecting CSRF. Features include whether or not an anti-CSRF token was present, characteristics of the HTTP request method, and possibly other HTTP headers such as referrer and origin. The features are then converted into the form of data vectors suitable for use by machine learning methods.

Machine Learning Model Training

The Mitch Engine uses several supervised learning algorithms, namely Random Forest, Decision Tree, Support Vector Machine, and Naïve Bayes, to train/optimize classifiers with labeled data sets consisting of secure and vulnerable request samples in a manner that the classifying function of the trained model performs adequately on unseen incoming requests. These models are then evaluated against test data on performance metrics (accuracy, precision, recall, F1 score) so that the detection of vulnerabilities would be reliable.

Real-Time Prediction and Classification

Being trained, the experimental models were integrated into the system to classify in real time the new HTTP requests submitted by the users. The engine issues a prediction as to whether an HTTP request under inspection is vulnerable to CSRF attacks or secure. Each identified vulnerability is recorded with detailed metadata, including URLs and timestamps.

Reporting and User Interaction

SECUREWEB offers an interactive dashboard that allows users to view scan results and model performance metrics. Administrators can also track users, log scans, and review detected vulnerabilities, ensuring that users and security administrators collaborate effectively.

V. ALGORITHM

The SECUREWEB tool uses classical supervised machine learning methods to detect Cross-Site Request Forgery (CSRF) vulnerabilities based on HTTP request features extracted from the network. The major approaches used are:

1. Random Forest

The random forest is an ensemble classification technique that produces multiple decision trees in the training phase and outputs the mode of their predictions. It gives more accurate results and controls overfitting through the aggregation of trees grown from random subsets of data and features. In SECUREWEB, the random forest can effectively tell a vulnerable request from a secure one by learning complicated patterns from the HTTP request features.

$$\mathbf{Y = mode\ \{h_t(x)\}_{t=1}^{T}}$$

Where $h_t(x)$ is the prediction from the t^{th} tree and T is the total number of trees.

2. Decision Tree

The decision tree algorithm builds a tree-like decision model based on feature values; it splits the training data at each internal node according to a certain criterion, such as information gain or Gini impurity, to achieve maximum separation of the data under each possible class. Its interpretability reveals the features that most influence CSRF vulnerability detection, thereby giving insight into the request characteristics leading to risk.

$$\text{Gini}(m) = 1 - \sum_{k=1}^{K} p_k^2$$

3. Support Vector Machine (SVM)

SVM finds the optimal hyperplane that separates points of two classes by maximizing margins. It works well with binary classification problems, especially when dealing with high-dimensional feature space. SECUREWEB uses SVM to characterize HTTP requests by mapping feature vectors and finding boundaries that differentiate best between vulnerable and safe requests.

$$f(x) = \text{sign}(w \cdot x + b)$$

4. Naïve Bayes

It assumes independence of features and applies Bayes' theorem. The independence assumption may be naive, but its application validates the performance of this algorithm in many classification problems. It calculates how likely a request is vulnerable given by its features, and thus it would be classified as being vulnerable or being safe according to the class that has the highest posterior probability. Fast and probabilistic detection is thus provided.

$$C = \arg\max_{C_k} P(C_k) \prod_{i=1}^{n} P(x_i|C_k)$$

In order to ensure reliable detection, from labeled datasets, algorithms are built and tested using the metrics of accuracy, precision, recall, and F1-score. Employing several models enables SECUREWEB to compare results and pick the best-performing classifier to detect CSRF vulnerabilities online.

VI. RESULTS AND DISCUSSION

Vulnerability Identification

SECUREWEB identified 35 new CSRF vulnerabilities in major websites and three more in production software environments, showing its practical utility on real-life infrastructures; more generally, this system opens real avenues of investigation for hidden flaws that cannot often be detected by traditional mechanisms.

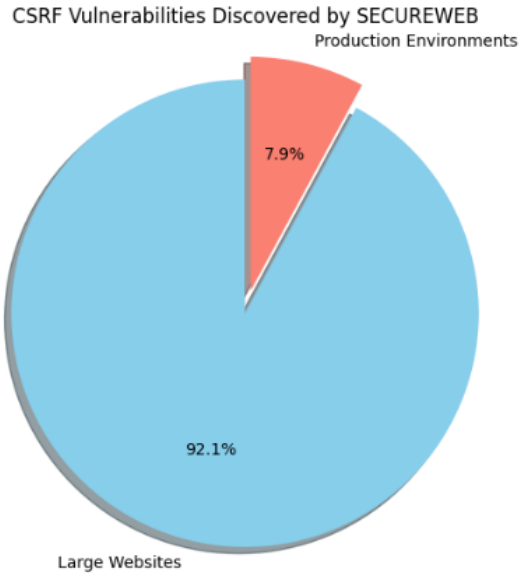

Fig 2: Vulnerability Identification Graph

Model Performance

The machine learning models considered by the Mitch Engine were Random Forest, Decision Tree, Support Vector Machine (SVM), and Naïve Bayes. These were tested on a labeled dataset of HTTP requests. Across these models, the system showed high accuracy, which implies that most requests were correctly classified as vulnerable or secure. The system also registered a high precision score, meaning few false positives would have been raised, and that users could be reasonably confident of the validity of actual vulnerability detections. Republic arrays were positives; when one variant detects a good fraction of real CSRF attacks, the F1-score shows a balanced performance between the two measurements, indicating a robust overall model. In general, among the algorithms tested, Random Forest tended to offer the best compromise between accuracy and interpretability, while SVM was the most capable in mastering highly complex feature spaces.

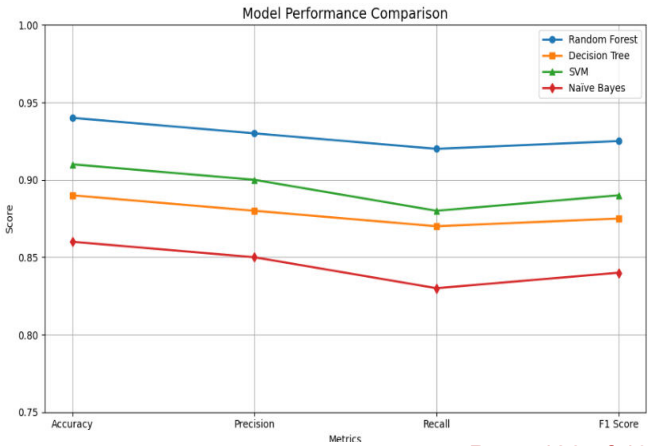

Fig 3: Model Performance

System Functionality

All components of SECUREWEB core functions, such as user registration, login, vulnerability scanning, machine learning training, and admin intervention, worked seamlessly and were passed in all test scenarios. The intuitive interface makes the tool usable for end users with little to no prior knowledge, while administrators retain control over system management and vulnerability oversight.

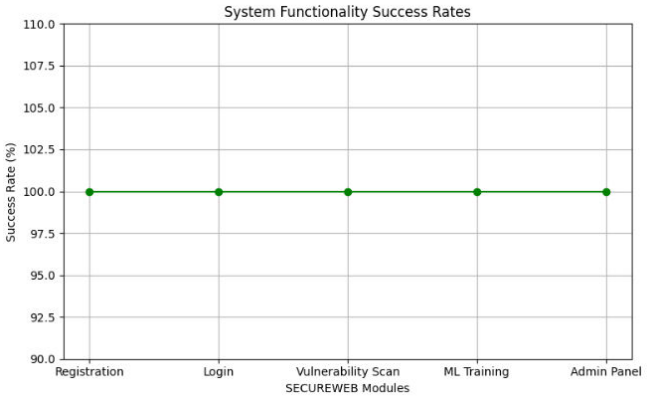

Fig 4: System Functionality

Analysis and Observations

This project proves the power of leveraging machine learning on HTTP request features for CSRF vulnerability detection sans access to source code. The multiple-algorithm approach allows for greater flexibility, thus enhancing the reliability of detection in different web environments. However, detection accuracy is contingent on the dataset used in training; poor training datasets or datasets with inherent biases can impinge on performance capabilities. False positives were kept to a minimum so that only a few might require a manual check. Incorporation of enhanced deep learning methods and wider datasets is expected in the future to yield better detection precision and recall.

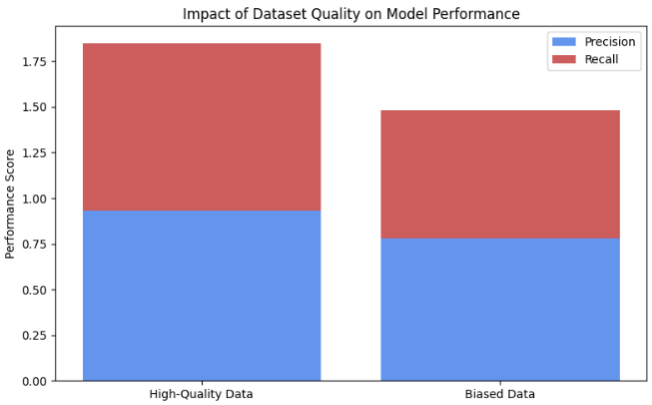

Fig 5: Impact of Dataet Quality on Model Performance

In summary, SECUREWEB is a scalable, efficient, and accessible solution for automating CSRF vulnerability detection, which harbors great potential for improving web application security.

CONCLUSION

The article assumes that SECUREWEB proves that machine-learning techniques can be integrated to detect Cross-Site Request Forgery (CSRF) vulnerabilities in web applications automatically, without any source code access. By crawling the web for data, extracting features, and using a variety of supervised learning algorithms, the system performs classification with great reliability on labeling HTTP requests as vulnerable or secure. The core Mitch Engine allows users to perform scans and train models and interpret the results easily, owing to intuitive user and admin interfaces. Experimental results confirm the framework's ability to find new CSRF vulnerabilities in major websites and production software while metrics such as accuracy, precision, recall, and F1 score confirm the robustness of the approach. In general, SECUREWEB can help organizations take a step forward in risk mitigation through a scalable, flexible, and user-friendly approach towards automating the detection of CSRF vulnerabilities.

FUTURE SCOPE

The future scope of SECUREWEB includes expanding its capability to detect web vulnerabilities beyond CSRF, such as Cross-Site Scripting (XSS) and SQL Injection (SQLi). Incorporation of additional deep learning model schemes, such as CNN, LSTM, and Transformer techniques, could further enhance the accurate detection capabilities of SECUREWEB when identifying complex attack patterns. Real-time monitoring options in the form of browser extensions would ensure that instantaneous scanning and alerting take place during web browser sessions. Contextual information and risk assessments would be enhanced by integrating with threat intelligence feeds. RBAC would otherwise further secure the system with fine-grained user permission enforcement. Improve the capability of the reporting dashboard with interactive visualizations and cross-platform compatibility to provide a highly refined user experience and accessibility. These enhancements will provide a big step toward making SECUREWEB a fully all-purpose, intelligent, and pragmatic web

vulnerability detection platform catering to different organizational requirements.

REFERENCES

1. M. Shahid, "Machine learning for detection and mitigation of web vulnerabilities and web attacks," *arXiv preprint arXiv:2304.14451*, 2023.
2. Y. Lin, R. Liu, D. M. Divakaran, J. Y. Ng, Q. Z. Chan, Y. Lu, Y. Si, F. Zhang, and J. S. Dong, "Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 3793–3810.
3. C. Greco, P. Pace, S. Basagni, and G. Fortino, "Jamming detection at the edge of drone networks using multi-layer perceptrons and decision trees," *Applied Soft Computing*, vol. 111, p. 107806, 2021.
4. S. Lalia and K. Moustafa, "Implementation of web browser extension for mitigating csrf attack," in *New Knowledge in Information Systems and Technologies*, A. Rocha, H. Adeli, L. P. Reis, and S. Costanzo, Eds. Cham: Springer International Publishing, 2019, pp. 867–880.
5. S. Calzavara, M. Conti, R. Focardi, A. Rabitti, and G. Tolomei, "Mitch: A machine learning approach to the black-box detection of csrf vulnerabilities," in *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2019.
6. H. Karimipour, A. Dehghantanha, R. M. Parizi, K.-K. R. Choo, and H. Leung, "A deep and scalable unsupervised machine learning system for cyber-attack detection in large-scale smart grids," *IEEE Access*, vol. 7, pp. 80778–80788, 2019.
7. A. Czeskis, "Lightweight server support for browser-based csrf protection," in *Proceedings of the 22nd International Conference on World Wide Web*, 2013.
8. M. Rocchetto, M. Ochoa, and M. Torabi Dashti, "Model-based detection of csrf," in *ICT Systems Security and Privacy Protection: 29th IFIP TC 11 International Conference, SEC 2014*, Marrakech, Morocco. Springer, 2014, pp. 30–43.
9. G. Pellegrino, M. Johns, S. Koch, M. Backes, and C. Rossow, "Deemon: Detecting csrf with dynamic analysis and property graphs," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017.
10. A. Sudhodanan, R. Carbone, L. Compagna, N. Dolgin, A. Armando, and U. Morelli, "Large-scale analysis & detection of authentication cross-site request forgeries," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2017, pp. 350–365.
11. M. Ozay, I. Esnaola, F. T. Y. Vural, S. R. Kulkarni, and H. V. Poor, "Machine learning methods for attack detection in the smart grid," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 8, pp. 1773–1786, 2015.
12. P. De Ryck, L. Desmet, W. Joosen, and F. Piessens, "Automatic and precise client-side protection against csrf attacks," in *Computer Security–ESORICS 2011: 16th European Symposium on Research in Computer Security*, Springer, 2011, pp. 100–116.
13. H. Shahriar and M. Zulkernine, "Client-side detection of cross-site request forgery attacks," in *2010 IEEE 21st International Symposium on Software Reliability Engineering*, 2010, pp. 358–367.
14. P. De Ryck, L. Desmet, T. Heyman, F. Piessens, and W. Joosen, "Csfire: Transparent client-side mitigation of malicious cross-domain requests," in *Engineering Secure Software and Systems: Second International Symposium, ESSoS 2010*, Springer, 2010, pp. 18–34.
15. X. Lin, P. Zavarsky, R. Ruhl, and D. Lindskog, "Threat modeling for csrf attacks," in *2009 International Conference on Computational Science and Engineering*, vol. 3, 2009, pp. 486–491.
16. Z. Mao, N. Li, and I. Molloy, "Defeating cross-site request forgery attacks with browser-enforced authenticity protection," in *Financial Cryptography and Data Security: 13th International Conference, FC 2009*, Springer, 2009, pp. 238–255.
17. N. Jovanovic, E. Kirda, and C. Kruegel, "Preventing cross site request forgery attacks," in *2006 Securecomm and Workshops*, 2006, pp. 1–10.