ISSN 2277-2685
IJESR/April-June. 2025/ Vol-15/Issue-2s/562-569
Tippa Lokesh *et. al.*, / International Journal of Engineering & Science Research

Web Vulnerability Using Machine Learning

Tippa Lokesh
PG scholar, Department of MCA, DNR college, Bhimavaram, Andhra Pradesh.
K.Sri Devi
(Assistant Professor), Master of Computer Applications, DNR college, Bhimavaram, Andhra Pradesh.

Abstract: Cross-Site Request Forgery (CSRF) is one of the oldest and simplest attacks on the Web, yet it is still effective on many websites and it can lead to severe consequences, such as economic losses and account takeovers. Unfortunately, tools and techniques proposed so far to identify CSRF vulnerabilities either need manual reviewing by human experts or assume the availability of the source code of the web application. In this paper we present Mitch, the first machine learning solution for the black-box detection of CSRF vulnerabilities. At the core of Mitch there is an automated detector of sensitive HTTP requests, i.e., requests which require protection against CSRF for security reasons. We trained the detector using supervised learning techniques on a dataset of 5,828 HTTP requests collected on popular websites, which we make available to other security researchers. Our solution outperforms existing detection heuristics proposed in the literature, allowing us to identify 35 new CSRF vulnerabilities on 20 major websites and 3 previously undetected CSRF vulnerabilities on production software already analysed using a state-of-the-art tool.

I. INTRODUCTION

Cross-Site Request Forgery (CSRF) is one of the simplest web attacks to understand, yet it has consistently been one of the top web security threats since its discovery in the early 2000s. In a CSRF attack, a malicious website forces the web browser to perform authenticated, security-sensitive operations at a target web application by means of cross-site requests, without any involvement of the browser's user. This can be done by using just standard HTML tags and JavaScript, making CSRF attempts trivial to perform and forcing securitysensitive web developers to implement solutions to filter out malicious cross-site requests abusing authentication. Robust defenses against CSRF are well-known [2], but still a significant number of modern web applications was shown to be vulnerable to this class of attacks [28], [34].

The main challenge to face when implementing protection against CSRF is that one has to strike a delicate balance between security and usability. As a matter of fact, the Web is built on top of cross-site requests, most of which are not malicious. Hence, web developers have to choose carefully which operations can only be made available to same-site requests without breaking the website functionality, and implement appropriate security checks on cross-site requests. It is thus easy to accidentally leave room for CSRF attacks, which motivated recent research on automated CSRF detection [28].

Deemon is the first research tool that automatically detects CSRF vulnerabilities [28]. Technically, Deemon is a modelbased security testing framework based on a runtime monitor implemented in the PHP interpreter. Although Deemon proved to be very effective on existing open-source web applications, it is a language-dependent analyzer, which only works on PHP applications whose source code is available for dynamic analysis.

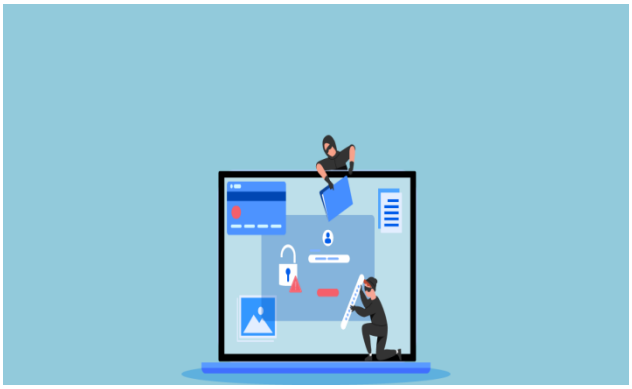

Fig.1.1 What is Website Vulnerabilities? | Indusface Blog

Overcoming this limitation is of paramount importance to advance the practical impact of the research on the automated detection of CSRF vulnerabilities, because existing web applications are often developed using several different technology stacks, and their source code might not be fully available for analysis in many

real-world cases. Black-box security testing is thus an appealing approach to detect CSRF vulnerabilities, but previous research highlighted that existing web application scanners are not effective in this task. For instance, [3] notes that an existing commercial tool does not report CSRF vulnerabilities "due to the difficulty of determining which forms in the application require protection from CSRF".

In fact, correctly detecting sensitive HTTP requests is the first major challenge to solve in order to propose an automated black-box detection tool for CSRF vulnerabilities. Unfortunately, previous heuristics from the literature [26], [30] give an unacceptably high number of false positives, as we show in Section IV-E. Thus, security practitioners are often forced to use manual penetration testing tools like Burp1 and ZAP2 to detect and test for CSRF vulnerabilities.

These tools require users to first manually identify the sensitive HTTP requests using their understanding of the web application, and then rely on techniques like those discussed in the OWASP Testing Guide3 to confirm the attack by running the generated CSRF proof of concept in a web browser to visually check its outcome. This is a complex and time-consuming manual task, which we aim to significantly automate and improve. To achieve this goal, we present Mitch, the first machine learning solution for the black-box detection of CSRF vulnerabilities. At the core of Mitch there is an automated detector of sensitive HTTP requests that outperforms existing detection heuristics proposed in the literature [26], [30].

1.4.1 Importance of Machine Learning in Web Application Security

Machine Learning offers a promising solution by utilizing manually labeled data to enhance automated analysis tools. By learning from labeled datasets that encapsulate known vulnerabilities and normal application behaviors, ML algorithms can effectively discern subtle patterns indicative of security threats. This capability bridges the gap between human understanding of web application semantics and automated detection tools, significantly bolstering web application security measures.

1.4.2 Introducing Mitch: The First ML Solution for CSRF Detection

As a cornerstone of our methodology, we introduce Mitch, the first ML-powered solution tailored for the black-box detection of Cross-Site Request Forgery (CSRF) vulnerabilities. CSRF attacks exploit the trust that a user has in a legitimate web application to perform unauthorized actions. Traditional methods often struggle to detect these vulnerabilities comprehensively in real-world scenarios, highlighting the need for advanced ML techniques.

1.4.3 Achievements and Impact of Mitch

Through the implementation of Mitch, we have achieved substantial results in identifying CSRF vulnerabilities across major websites and production software environments. Specifically, Mitch has successfully uncovered 35 previously unknown CSRF vulnerabilities on 20 prominent websites and identified an additional 3 vulnerabilities within critical production software systems. These findings underscore Mitch's effectiveness in enhancing the security posture of web applications through advanced ML-driven detection capabilities.

1.5 Web Vulnerabilities

The term vulnerability is a weakness, glitch, or loophole in web application development. An exploit is when the vulnerability is exploited, and the attack succeeds. Based on the research work [**2**], web application vulnerabilities can be classified into three categories (**Figure 2**): Improper input validation refers to an incorrect validation and sanitization of user input. SQL injection and Cross-Site Scripting (XSS) are examples of web attacks caused by improper input validation vulnerability.

II. LITERATURE SURVEY

[1] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, "The security of machine learning," Machine Learning, vol. 81, no. 2, pp. 121–148, 2010.

Machine learning's ability to rapidly evolve to changing and complex situations has helped it become a fundamental tool for computer security. That adaptability is also a vulnerability: attackers

can exploit machine learning systems. We present a taxonomy identifying and analyzing attacks against machine learning systems. We show how these classes influence the costs for the attacker and defender, and we give a formal structure defining their interaction. We use our framework to survey and analyze the literature of attacks against machine learning systems. We also illustrate our taxonomy by showing how it can guide attacks against SpamBayes, a popular statistical spam filter. Finally, we discuss how our taxonomy suggests new lines of defenses.

[2] A. Barth, C. Jackson, and J. C. Mitchell, "Robust defenses for crosssite request forgery," in Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008, 2008, pp. 75–88.

Cross-Site Request Forgery (CSRF) is a widely exploited web site vulnerability. In this paper, we present a new variation on CSRF attacks, login CSRF, in which the attacker forges a cross-site request to the login form, logging the victim into the honest web site as the attacker. The severity of a login CSRF vulnerability varies by site, but it can be as severe as a cross-site scripting vulnerability. We detail three major CSRF defense techniques and find shortcomings with each technique. Although the HTTP Referer header could provide an effective defense, our experimental observation of 283,945 advertisement impressions indicates that the header is widely blocked at the network layer due to privacy concerns. Our observations do suggest, however, that the header can be used today as a reliable CSRF defense over HTTPS, making it particularly well-suited for defending against login CSRF. For the long term, we propose that browsers implement the Origin header, which provides the security benefits of the Referer header while responding to privacy concerns.

[3] J. Bau, E. Bursztein, D. Gupta, and J. C. Mitchell, "State of the art: Automated black-box web application vulnerability testing," in 31st IEEE Symposium on Security and Privacy, S&P 2010, 16-19 May 2010, Berkeley/Oakland, California, USA, 2010, pp. 332–345.

Black-box web application vulnerability scanners are automated tools that probe web applications for security vulnerabilities. In order to assess the current state of the art, we obtained access to eight leading tools and carried out a study of: (i) the class of vulnerabilities tested by these scanners, (ii) their effectiveness against target vulnerabilities, and (iii) the relevance of the target vulnerabilities to vulnerabilities found in the wild. To conduct our study we used a custom web application vulnerable to known and projected vulnerabilities, and previous versions of widely used web applications containing known vulnerabilities. Our results show the promise and effectiveness of automated tools, as a group, and also some limitations. In particular, "stored" forms of Cross Site Scripting (XSS) and SQL Injection (SQLI) vulnerabilities are not currently found by many tools. Because our goal is to assess the potential of future research, not to evaluate specific vendors, we do not report comparative data or make any recommendations about purchase of specific tools.

[4] A. Bhandare, M. Bhide, P. Gokhale, and C. Rohan, "Applications of convolutional neural networks," International Journal of Computer Science and Information Technologies, vol. 7, pp. 2206–2215, September– October 2016.

In recent years, deep learning has been used extensively in a wide range of fields. In deep learning, Convolutional Neural Networks are found to give the most accurate results in solving real world problems. In this paper, we give a comprehensive summary of the applications of CNN in computer vision and natural language processing. We delineate how CNN is used in computer vision, mainly in face recognition, scene labelling, image classification, action recognition, human pose estimation and document analysis. Further, we describe how CNN is used in the field of speech recognition and text classification for natural language processing. We compare CNN with other methods to solve the same problem and explain why CNN is better than other methods.

[5] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, Oct. 2001.

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges a.s. to a limit as the number of trees in the forest becomes large. The generalization error of a forest

of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them. Using a random selection of features to split each node yields error rates that compare favorably to Adaboost (Y. Freund & R. Schapire, *Machine Learning*: *Proceedings of the Thirteenth International conference*, ***, 148–156), but are more robust with respect to noise. Internal estimates monitor error, strength, and correlation and these are used to show the response to increasing the number of features used in the splitting. Internal estimates are also used to measure variable importance. These ideas are also applicable to regression.

[6] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Classification and Regression Trees. Wadsworth, 1984.

The methodology used to construct tree structured rules is the focus of this monograph. Unlike many other statistical procedures, which moved from pencil and paper to calculators, this text's use of trees was unthinkable before computers. Both the practical and theoretical sides have been developed in the authors' study of tree methods. Classification and Regression Trees reflects these two sides, covering the use of trees as a data analysis method, and in a more mathematical framework, proving some of their fundamental properties.

[7] S. Calzavara, R. Focardi, N. Grimm, and M. Maffei, "Micro-policies for web session security," in IEEE 29th Computer Security Foundations Symposium, CSF 2016, Lisbon, Portugal, June 27 - July 1, 2016, 2016, pp. 179–193.

—Micro-policies, originally proposed to implement hardware-level security monitors, constitute a flexible and general enforcement technique, based on assigning security tags to system components and taking security actions based on dynamic checks over these tags. In this paper, we present the first application of micro-policies to web security, by proposing a core browser model supporting them and studying its effectiveness at securing web sessions. In our view, web session security requirements are expressed in terms of a simple, declarative information flow policy, which is then automatically translated into a micropolicy enforcing it. This leads to a browser-side enforcement mechanism which is elegant, sound and flexible, while being accessible to web developers. We show how a large class of attacks against web sessions can be uniformly and effectively prevented by the adoption of this approach. We also develop a proof-ofconcept implementation of a significant core of our proposal as a Google Chrome extension, Michrome: our experiments show that Michrome can be easily configured to enforce strong security policies without breaking the functionality of websites.

III. PROPOSED METHOD

4.1 Proposed Work

In this project, we propose a methodology to leverage Machine Learning (ML) for the detection of web application vulnerabilities. Web applications are particularly challenging to analyses, due to their diversity and the widespread adoption of custom programming practices. ML is thus very helpful for web application security: it can take advantage of manually labelled data to bring the human understanding of the web application semantics into automated analysis tools. We use our methodology in the design of Mitch, the first ML solution for the black-box detection of Cross-Site Request Forgery (CSRF) vulnerabilities. Mitch allowed us to identify 35 new CSRFs on 20 major websites and 3 new CSRFs on production software.

This project consists of two modules

1) Admin Module: admin can login to system using username and password as admin and admin. After login admin can activate new registered user account. Admin can view all CSRF list, view users, view POST and GET request
2) New User Sign up: user can sign up with the application
3) User Login: once account activated then user can login to system and then enter any URL along with depth value to scan URL to detect vulnerability and then apply mitch process to identify URL as vulnerable or not. User can train ML algorithms on different methods and then calculate accuracy and other metrics.

To run project install python 3.7.2 and then install all packages given in requirements.txt file. Install MYSQL database and then open MYSQL console and then copy content from 'database.txt' file and paste in MYSQL console to create database

4.2 DATASET CONSTRUCTION

In this section, we first give a practical definition of sensitive request and we describe how we collect and label a real-world dataset of sensitive and insensitve requests, taking also into account possible ethical implications.

RESULT

To run project double click on 'run.bat file to start python server and get below page

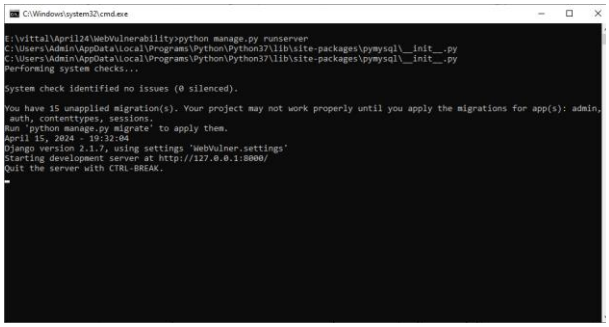

Fig.7.1 python server started

In above screen python server started and now open browser and enter URL as http://127.0.0.1:8000/index.html and press enter key to get below page

Fig.7.2 New User Sign up'

In above screen click on 'New User Sign up' link to get below page

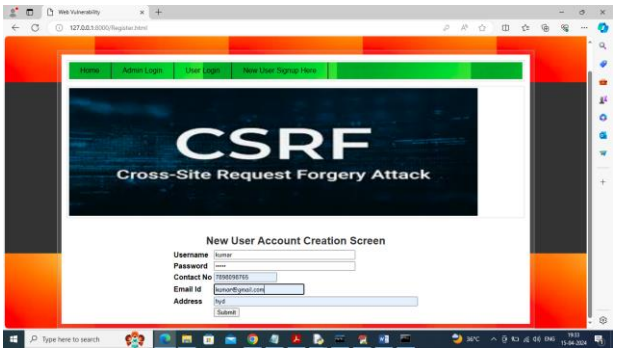

Fig.7.3 user is entering sign up data

In above screen user is entering sign up data and then press button to get below page

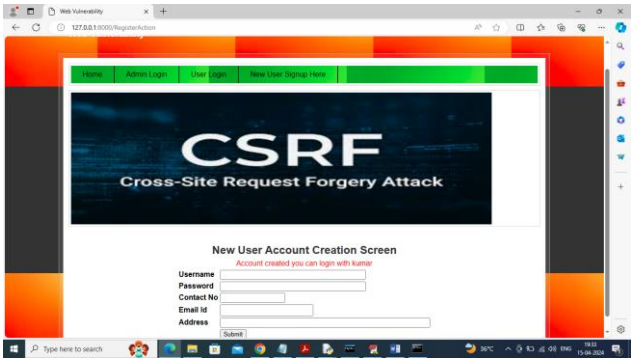

Fig.7.4 user account created

In above screen user account created and now login as admin to activate user account

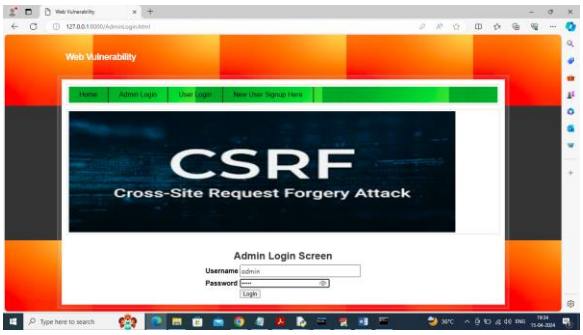

Fig.7.5 admin is login

In above screen admin is login and after login will get below page

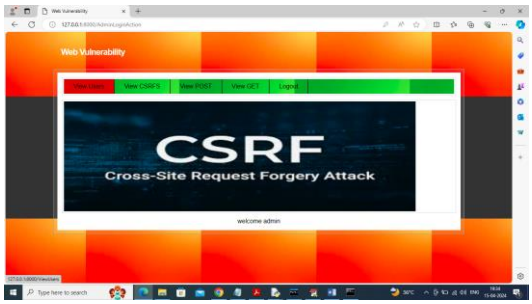

Fig.7.6 'View Users'

In above screen admin can click on 'View Users' link to get below page

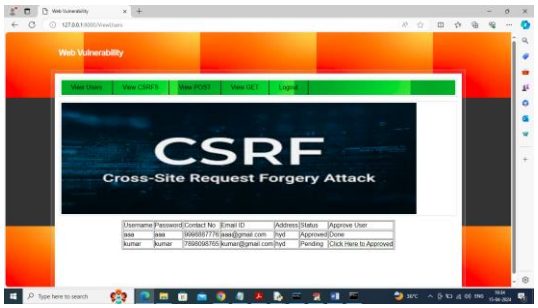

Fig.7.7. admin can view list of accounts

In above screen admin can view list of accounts and can click on 'Click Here to Approved' link on pending accounts to approve users and get below page

Fig.7.8 user account approved

In above screen user account approved and now logout and login as user

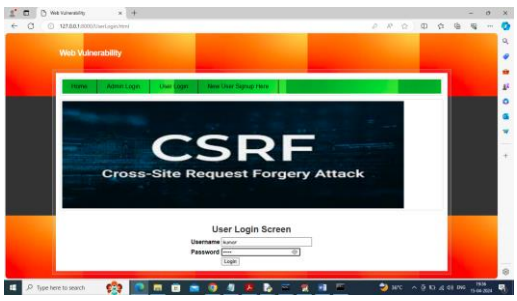

Fig.7.9 user is login

In above screen user is login and after login will get below page

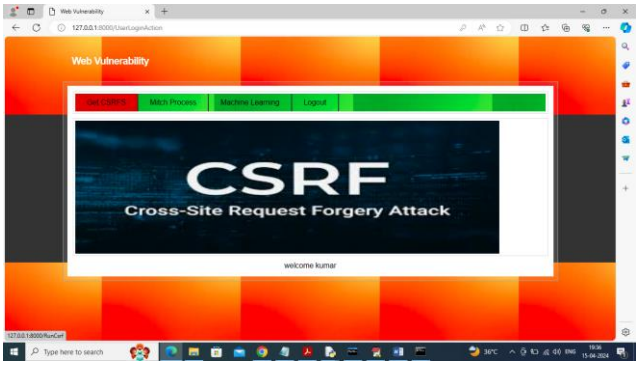

Fig.7.10'Get CSRFS' link

In above screen user can click on 'Get CSRFS' link to get below page

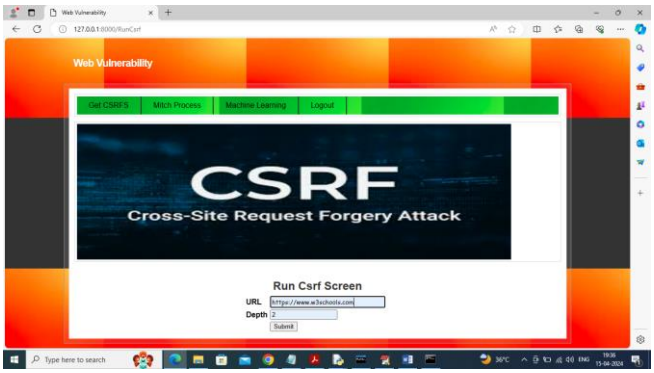

Fig.7.11 enter URL and depth value

In above screen enter URL and depth value and then click on button to get below URL scanning output

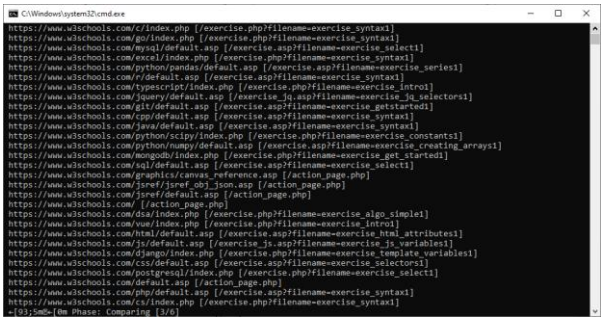

Fig.7.12 scanned URL

In above screen can see all scanned URL and then will get below page

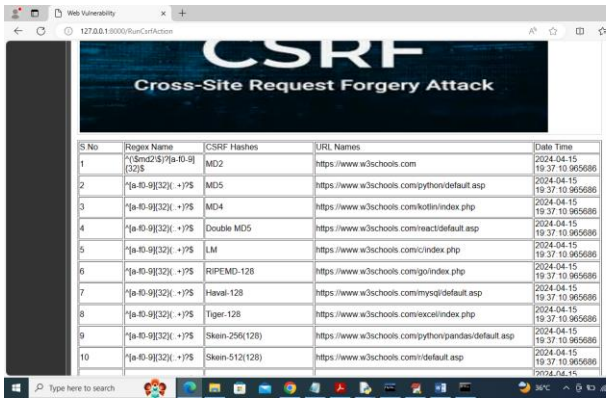

Fig.7.13 all CSRF list obtained from given URL

In above screen can see all CSRF list obtained from given URL and scroll down to view all details

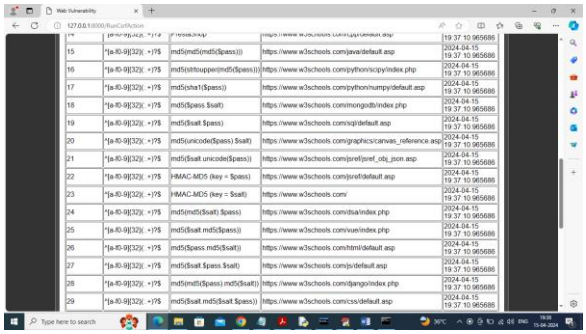

Fig.7.14 Mitch Process' link

Now user can click on 'Mitch Process' link to run mitch and get below output

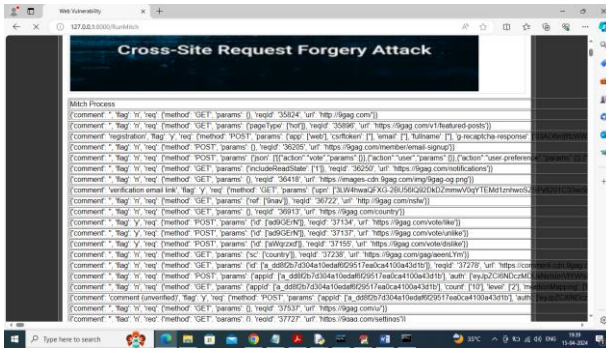

Fig.7.15 all MITCH process

In above screen can see all MITCH process and now click on 'Machine Learning' link to train ML algorithm and get below output

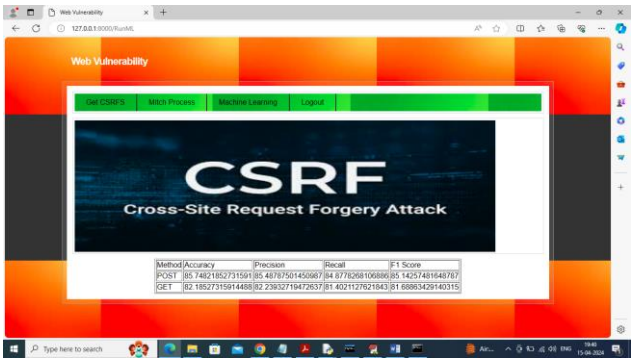

Fig.7.16 ML accuracy on both GET and POST methods

In above screen can see ML accuracy on both GET and POST methods and now logout and login as admin to view other process

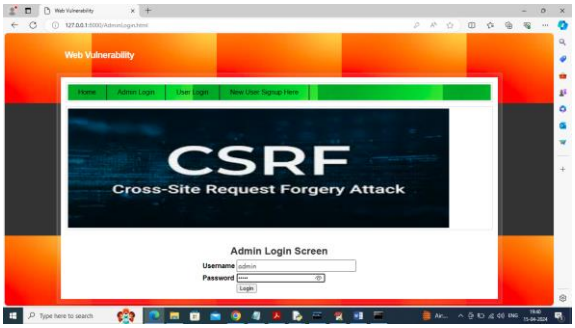

Fig.7.17 admin is login

In above screen admin is login and after login will get below page

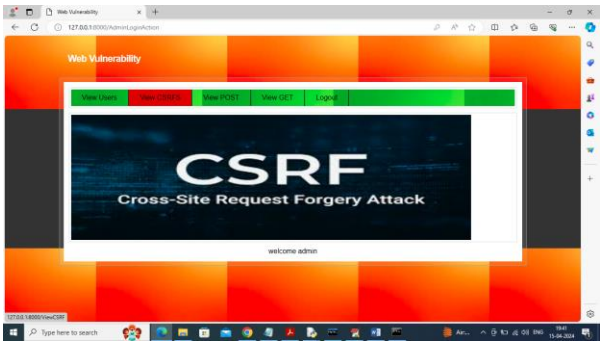

Fig.7.18 'View CSRFS' link to view all past CSRFS list

In above screen admin can click on 'View CSRFS' link to view all past CSRFS list and get below page

Fig.7.19 admin can view list of CSRF list

In above screen admin can view list of CSRF list and now click on 'View Post' link to view all post request

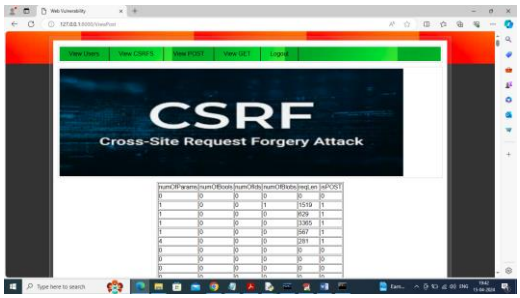

Fig.7.20 all POST request data

In above screen can see all POST request data and now click on 'View Get' link to get all get request data

Fig.7.21 list of all GET request

In above screen can see list of all GET request and similarly by following above screens you can run all modules of the project

IV. CONCLUSION

Mitch is the first machine learning solution for the black-box detection of CSRF vulnerabilities. It exploits supervised learning techniques to accurately identify HTTP requests which require protection against CSRF (sensitive requests) and relies on heuristics to find attacks abusing them. We experimentally showed that Mitch produces a small number of false positives in practice, and that it is effective enough to expose new CSRF vulnerabilities in existing websites and production software, some of which escaped previous approaches. By using Mitch, we observed that web application developers are aware of the dangers of CSRF and typically try to implement appropriate protection mechanisms, yet they accidentally leave room for attacks, e.g., because they forget to check anti-CSRF tokens which are sent back to the web application. There are a number of avenues for future work. Adversarial learning is an active research area whose main goal is designing classification algorithms which are robust to the presence of attackers who actively try to fool them into misprediction [1].

REFERENCES

[1] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, "The security of machine learning," Machine Learning, vol. 81, no. 2, pp. 121–148, 2010.

[2] A. Barth, C. Jackson, and J. C. Mitchell, "Robust defenses for crosssite request forgery," in Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008, 2008, pp. 75–88.

[3] J. Bau, E. Bursztein, D. Gupta, and J. C. Mitchell, "State of the art: Automated black-box web application vulnerability testing," in 31st IEEE Symposium on Security and Privacy, S&P 2010, 16-19 May 2010, Berkeley/Oakland, California, USA, 2010, pp. 332–345.

[4] A. Bhandare, M. Bhide, P. Gokhale, and C. Rohan, "Applications of convolutional neural networks," International Journal of Computer Science and Information Technologies, vol. 7, pp. 2206–2215, September– October 2016.

[5] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[6] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Classification and Regression Trees. Wadsworth, 1984.

[7] S. Calzavara, R. Focardi, N. Grimm, and M. Maffei, "Micro-policies for web session security," in IEEE 29th Computer Security Foundations Symposium, CSF 2016, Lisbon, Portugal, June 27 - July 1, 2016, 2016, pp. 179–193.

[8] S. Calzavara, R. Focardi, M. Squarcina, and M. Tempesta, "Surviving the web: A journey into web session security," ACM Comput. Surv., vol. 50, no. 1, pp. 13:1–13:34, 2017.

[9] S. Calzavara, G. Tolomei, M. Bugliesi, and S. Orlando, "Quite a mess in my cookie jar!: leveraging machine learning to protect web authentication," in 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, 2014, pp. 189– 200.

[10] S. Calzavara, G. Tolomei, A. Casini, M. Bugliesi, and S. Orlando, "A supervised learning approach to protect client authentication on the web," TWEB, vol. 9, no. 3, pp. 15:1–15:30, 2015.