International Journal of Scientific Research in Engineering & Technology
Volume 4, Issue 1 (January- February 2024), PP: 26-29.
https://www.doi.org/10.59256/ijsreat.20240401005
www.ijsreat.com ISSN No: 2583-1240

Machine Learning for Web Vulnerability Detection: The Case of Cross-Site Request Forgery

N. Sravani[1], O.Sai Raju[2], Ch.Harish[3], B.Anil Kumar[4], S.Anirudh[5]
[1]*Assistant professor, Department of Information Technology, CMR Engineering College (UGC Autonomous), Hyderabad, Telangana, India.*
[2,3,4,5] *B.Tech IV-Year, Department of Information Technology, CMREC, (UGC Autonomous), Hyderabad, Telangana, India.*

To Cite this Article: *N. Sravani[1], O.Sai Raju[2], Ch.Harish[3], B.Anil Kumar[4], S.Anirudh[5], "Machine Learning For Web Vulnerability Detection: The Case of Cross-Site Request Forgery", International Journal of Scientific Research in Engineering & Technology Volume 04, Issue 01, January-February 2024, PP: 26-29.*

Abstract: *Cross-site request forgery (CSRF) vulnerabilities pose a significant threat to web application security, enabling attackers to execute unauthorized actions on behalf of authenticated users. Conventional CSRF detection methods, such as manual code review and static analysis, are often time-consuming, error-prone, and inefficient. Proposes Mitch, a novel machine learning (ML)-based solution for the black-box detection of CSRF vulnerabilities. Mitch employs supervised learning, trained on a comprehensive dataset of HTTP requests and responses, to effectively identify security-sensitive HTTP requests and uncover CSRF vulnerabilities within them. Rigorous evaluations on a diverse set of real-world web applications demonstrate Mitch's remarkable ability to detect CSRF vulnerabilities with high accuracy, outperforming traditional methods. Mitch's automated nature eliminates the need for manual code review and static analysis, saving time and effort while reducing the risk of human error. Additionally, Mitch's scalability allows seamless integration into continuous integration and continuous delivery (CI/CD) pipelines, enabling continuous security monitoring and vulnerability detection. Mitch's efficacy extends beyond detecting known CSRF vulnerabilities. Its ability to identify patterns and relationships enables it to uncover obscure CSRF vulnerabilities that may have been overlooked by traditional methods, including zero-day vulnerabilities. In conclusion, Mitch emerges as a powerful tool for enhancing web application security, offering a comprehensive and automated solution for detecting CSRF vulnerabilities. Its ability to handle complex web applications, uncover hidden CSRF vulnerabilities, and integrate into CI/CD pipelines makes it an indispensable tool for web security professionals. Mitch's adoption has the potential to significantly reduce the risk of CSRF attacks and safeguard sensitive user data. We propose a methodology to leverage machine learning (ML) for the detection of web application vulnerabilities. We use it in the design of Mitch, the first ML solution for the black-box detection of cross-site request forgery vulnerabilities. Finally, we show the effectiveness of Mitch on real software. 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 labeled data to bring the human understanding of the web application semantics into automated analysis toolsMitch allowed us to identify 35 new CSRFs on 20 major websites and 3 new CSRFs on production software.*

Keywords: *Mitch, CSRF, CI/CD pipelines, Security Token Service (STS), Same-Origin Policy (SOP).*

I.INTRODUCTION

Web applications are the most common interface to security sensitive data and functionality available nowadays. They are routinely used to file tax incomes, access the results of medical screenings, perform financial transactions, and share opinions with our circle of friends, just to mention a few popular use cases. On the downside, this means that web applications are appealing targets to malicious users (attackers) who are determined to force economic losses, unduly access confidential data or create embarrassment to their victims. Securing web applications is well known to be hard. There are several reasons for this, ranging from the heterogeneity and complexity of the web platform to the adoption of undisciplined scripting languages offering dubious security guarantees and not amenable for static analysis. In such a setting, black-box vulnerability detection methods are particularly popular. As opposed to white-box techniques which require access to the web application source code, black-box methods operate at the level of HTTP traffic, i.e., HTTP requests and responses. Though this limited perspective might miss important insights, it has the key advantage of offering a language-agnostic vulnerability detection approach, which abstracts from the complexity of scripting languages and offers a uniform interface to the widest possible range of web applications.

This sounds appealing, yet previous work showed that such an analysis is far from trivial. One of the main challenges there is how to expose to automated tools a critical ingredient of effective vulnerability detection, i.e., an understanding of the web application semantics. Example: Cross-Site Request Forgery (CSRF) Cross-Site Request Forgery (CSRF) is a well-known

web attack that forces a user into submitting unwanted, attacker controlled HTTP requests towards a vulnerable web application in which she is currently authenticated. The key concept of CSRF is that the malicious requests are routed to the web application through the user's browser, hence they might be indistinguishable from intended benign requests which were actually authorized by the user.Notice that CSRF does not require the attacker to intercept or modify user's requests and responses: it suffices that the Preventing CSRF.To prevent CSRF, web developers have to implement explicit protection mechanisms. If adding extra user interaction does not affect usability too much, it is possible to force re-authentication or use one-time passwords / CAPTCHAs to prevent cross-site requests going through unnoticed. In many cases, however, automated prevention is preferred: the recently introduced SameSite cookie attribute can be used to prevent cookie attachment on cross-site requests, which solves the root cause of CSRF and is highly recommended for new web applications. Unfortunately, this defense is not yet widespread and existing web applications typically filter out cross-site request by using any of the following techniques:

1) checking the value of standard HTTP request headers such as Referrer and Origin, indicating the page originating the request;
2) checking the presence of custom HTTP request headers like X-Requested-With, which cannot be set from a cross-site position;
3) checking the presence of unpredictable anti-CSRF tokens,set by the server into sensitive forms.

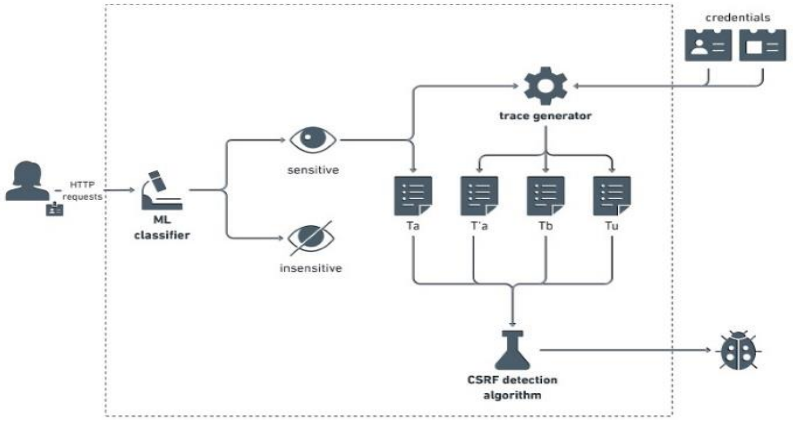
A recent paper discusses the pros and cons of these different solutions. However, all three options suffer from the same limitation: they require a careful and fine-grained placement of security checks. For example, tokens should be attached to all and only the security-sensitive HTTP requests, so as to ensure complete protection without harming the user experience.Using a token to protect a "like" button is useful to prevent the attack discussed above, yet having a token on the social network homepage is undesirable, because it might lead to rejecting legitimate cross-site requests, e.g., from clicks on the results of a search engine indexing the social network. In the end, finding the "optimal" placement of anti-CSRF defenses is typically a daunting task for web developers. Modern web application development frameworks provideAutomated support for this, yet CSRF vulnerabilities are still routinely found even in top-ranked websites. This motivates the need for effective CSRF detection tools. But how can we provide automated tool support for CSRF detection if we have no mechanized way to detect which HTTP requests are actually security-sensitive.are passed - No splits.

This work presents the most current and comprehensive understanding of a not very well understood web vulnerability known as the CSRF (Cross-Site Request Forgery) and provides specific solutions to identify and defend CSRF vulnerabilities. The immediate benefits of this work include tangible and pragmatic application framework for use by individuals, organizations and developers, either as consumers or providers of web services. This work responds directly to the challenges of keeping pace with the evolving cyber technologies and vulnerabilities that increasingly expose businesses towards privacy and identity theft specific attacks, where the traditional anti-virus and anti-spyware approaches fail. The urgency to come up with appropriate detection and defense mechanism against the lethal CSRF attacks is indicated due to expanding cloud based technologies, HTML5, Semantic Web, and various emerging security frameworks comprised of inchoate vestigial of "Big Data" that demand exceedingly evolved defense mechanisms. A methodical approach is used to investigate CSRF attacks and remedies are proposed by introducing a novel distinctive set of algorithms that use intelligent assumptions to detect and defend CSRF. In this work, design details of a CSRF Detection Model (CDM), implantation and experimentation results of CDM are elaborated to detect, predict and provide solutions for CSRF attacks on contemporary Web Applications and Web Services environment. Additionally, CDM based recommendations for users and providers of cyber security products and services are presented. Cross-Site Request Forgery (CSRF) attack causes actions on a web application without the knowledge of the user in an authenticated browser session.

CSRF attacks specifically target state-changing requests like transferring funds, changing email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application. CSRF, also known as the Sleeping Giant, was considered to be one of the top 5 web vulnerabilities only 4 years ago. Even so, at least 270 incidents of CSRF attacks have been reported as of 2016. Not much has improved in terms of new CSRF solutions since the CSRF problem appeared in the horizon in 2010. Cross-Site Reference Forgery (CSRF) and Cross-Site Scripting (XSS) vulnerabilities have received much attention recently. An XSS attack, one of the top 3 current cyber security challenges, occurs when an attacker injects malicious code (typically JavaScript), including a CSRF attack code, into a site for the purpose of targeting users of the site, e.g., sites that allow posting comments.

According to the Open Web Application Security Project (OWASP), an open web community dedicated to address cyber security challenges, CSRF is one of the top eight cyber security vulnerabilities in the world, today. While CSRF attacks are simple to create and exploit, amazingly, they are difficult to identify and mitigate.A search for "Cross Site Scripting" (which differs from CSRF) on the ACM Digital Library returned 117 papers, while a search for "CSRF" returned only four papers. A search for "XSS" on Safari Books Online (a collection of over 5000 books on technology) showed the term appeared in 96 books, while "CSRF OR XSRF" appeared in only 13 books. Very few CSRF solutions are developed and implemented. Even so, while current solutions still lack common applicability all the pieces for large scale massive CSRF attacks are already in place [53]. This state of the current relentless CSRF attacks and meager defenses dynamics is the primary motivation for undertaking this study.

II.SYSTEM ARCHITECTURE



III.INPUT DESIGN

The input design is the link between the information system and the user. It comprises thedeveloping specification and procedures for data preparation and those steps are necessary to puttransaction data in to a usable form for processing can be achieved by inspecting the computer toread data from a written or printed document or it can occur by having people keying the datadirectly into the system. The design of input focuses on controlling the amount of input required,controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. Theinput is designed in such a way so that it provides security and ease of use with retaining theprivacy.

IV.OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.
1. Convey information about past activities, current status or projections of the Future.
2. Signal important events, opportunities, problems, or warnings.
3. Trigger an action.
4. Confirm an action.

V.RESULT

In fig(a) , you can see that how precision and recall for the CSRFs has been detected

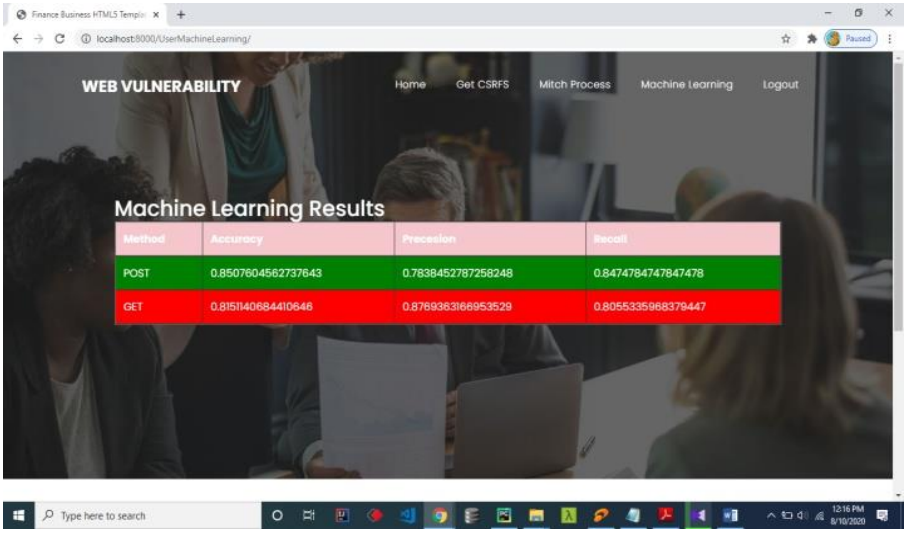


Fig (a)

VI.ACKNOWLEDGEMENT

1 . We are extremely grateful to Dr. A. Srinivasula Reddy, Principal and Dr. Madhavi Pingili, HOD, Department of IT, CMR Engineering College for their constant support.

2 .We are extremely thankful toN. Sravani, Assistant Professor, Internal Guide, Department of IT, for her constant guidance, encouragement and moral support throughout the project.

3. We express our thanks to all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time.

REFERENCES

1. *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.*
2. *A. Sudhodanan, R. Carbone, L. Compagna, N. Dolgin, A. Armando and U. Morelli, "Large-scale analysis & detection of authentication cross-site request forgeries", Proc. 2017 IEEE European Symp. Security and Privacy (EuroS&P 2017), pp. 350-365*
3. *S. Calzavara, A. Rabitti, A. Ragazzo and M. Bugliesi, "Testing for integrity flaws in web sessions", Proc. Computer Security 24rd European Symp. Research Computer Security (ESORICS 2019), pp. 606-624, Sept. 2019.*
4. *J. Bau, E. Bursztein, D. Gupta and J. C. Mitchell, "State of the art: Automated black-box web application vulnerability testing", Proc. 31st IEEE Symp. Security and Privacy (S&P 2010), pp. 332-345, May 2010.*
5. *A. Doupé, M. Cova and G. Vigna, "Why Johnny can't pentest: An analysis of black-box web vulnerability scanners", Proc. 7th Int. Conf. Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA 2010), pp. 111-131, July 2010.*
6. *Barth, C. Jackson and J. C. Mitchell, "Robust defenses for cross-site request forgery", Proc. 2008 ACM Conf. Computer and Communications Security (CCS 2008), pp. 75-88.*
7. *M. Mohri, A. Rostamizadeh and A. Talwalkar, Foundations of Machine Learning, Cambridge, MA:MIT Press, 2012*
8. *. Calzavara, G. Tolomei, A. Casini, M. Bugliesi and S. Orlando, "A supervised learning approach to protect client authentication on the web", ACM Trans. Web, vol. 9, no. 3, pp. 15:1-15:30, 2015.*
9. *S. Calzavara, M. Conti, R. Focardi, A. Rabitti and G. Tolomei, "Mitch: A machine learning approach to the black-box detection of CSRF vulnerabilities", Proc. IEEE European Symp. Security and Privacy (EuroS&P 2019), pp. 528-543, June 2019.*
10. *G. Pellegrino, M. Johns, S. Koch, M. Backes and C. Rossow, "Deemon: Detecting CSRF with dynamic analysis and property graph", Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security(CCS 2017), pp. 1757-1771, Oct. 2017.*