A PARADIGM SHIFT IN XSS-DOM MITIGATION VIA DYNAMIC CONTENT SECURITY POLICY

REPORT
IT5712 - PROJECT I

Submitted by

Krishnaa S (2020506045)
Jawahar A S (2020506035)
Thamizharasi M (2020506102)

Guided by

Dr. P. Kola Sujatha

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

MADRAS INSTITUTE OF TECHNOLOGY

ANNA UNIVERSITY: CHENNAI 600044

25th October 2023

INTRODUCTION:

Cross-Site Scripting (XSS) and Document Object Model (DOM) attacks are critical security concerns for web applications. XSS vulnerabilities allow attackers to inject malicious scripts into web pages viewed by other users, potentially compromising their data or session information. These attacks exploit the trust a user has in a website, enabling unauthorized access or data theft. DOM-based XSS attacks specifically manipulate a webpage's Document Object Model, which represents the structure of a web page in a hierarchical manner. In DOM-based XSS, malicious scripts manipulate the DOM of a web page dynamically, often targeting client-side scripts and affecting the behavior of web applications. Detecting and mitigating DOM-based XSS attacks is essential for safeguarding user data, maintaining application integrity, and preventing unauthorized access to sensitive information. Security measures involve input validation, output encoding, and monitoring for unusual script behavior to thwart these attacks and ensure a secure web environment.

Our focus is on using pattern matching techniques to detect XSS-DOM vulnerabilities, adding an extra layer of security to web applications. This proactive approach helps identify potential attacks early, enhancing overall security. We also explore preventive strategies, such as input validation and Content Security Policy (CSP), in detail. These strategies are effective in reducing the risks associated with XSS-DOM attacks, making web environments more secure against potential breaches.

OBJECTIVES:

Understanding Vulnerabilities: Gain insight into the nature and mechanics of Cross-Site Scripting (XSS) and DOM-based attacks.

Detection Techniques: Investigate methods (Pattern matching) for identifying XSS-DOM vulnerabilities, enabling proactive defense against potential attacks.

Preventive Strategies: Explore strategies such as input validation and Content Security Policy (CSP) to mitigate the risks of XSS-DOM attacks.

Real-world Insights: Analyze real-world cases of notable attacks to grasp the potential consequences and implications of these vulnerabilities.

PROBLEM STATEMENT:

Cross-Site Scripting (XSS) vulnerabilities persist in web applications, allowing malicious scripts to compromise user data and privacy. Existing Content Security Policy solutions struggle to effectively protect dynamic web pages where content changes based on user interactions.

Lack of adaptive security measures leaves dynamic pages exposed to XSS attacks, necessitating a novel solution.

TECH STACK:

- Python
- Flask
- Javascript
- Sqlite

System Requirements:

- Windows 10 and above
- VS Code
- Chrome Browser

IMPLEMENTATION MODULES:

Module 1: DETECTION

- In this module, we have implemented a set of regular expression patterns designed to enhance the security system's ability to identify potential XSS-DOM vulnerabilities.
- This pattern integration allows for more robust detection of security threats related to client-side scripting and DOM manipulation.

Module 2: MITIGATION

- Under this module, we explore the integration of Content Security Policy (CSP) as a proactive measure to mitigate the risks associated with client-side scripting and DOM manipulation.
- We employ a specific whitelist framework to ensure no scripts or resources are loaded from a source that we are unaware of.
- Additionally, we receive alerts when such an attack is detected and blocked.
- These alerts include detailed information about the detected threats and recommended actions for effective incident response.
- This combined approach ensures that not only are vulnerabilities identified, but also steps are taken to mitigate and respond to potential threats effectively.

NOVELTY:

Traditionally, detecting XSS DOM attacks involved manual inspection of source code for potential vulnerabilities. We've introduced a novel approach that automates this process using pattern matching techniques and dynamic pattern learning.

Pattern Matching Techniques: We use predefined patterns to identify known attack vectors within user inputs.

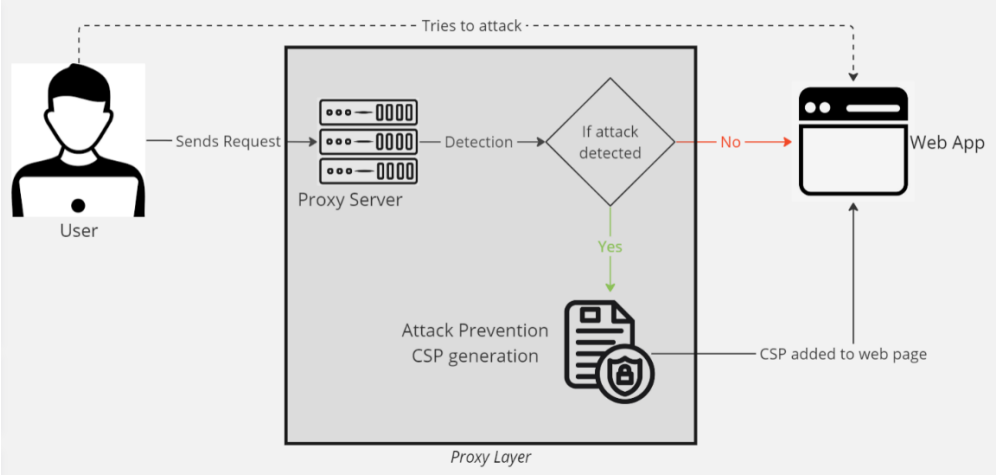
Dynamic Pattern Updation: Our innovation is a dynamic pattern updation algorithm that learns from the application's behavior. It employs taint analysis to track data flow and assesses inputs for suspicious keywords and context.

Risk Assessment: Inputs are assigned risk scores, and those exceeding a predefined threshold are considered potential new attack patterns.

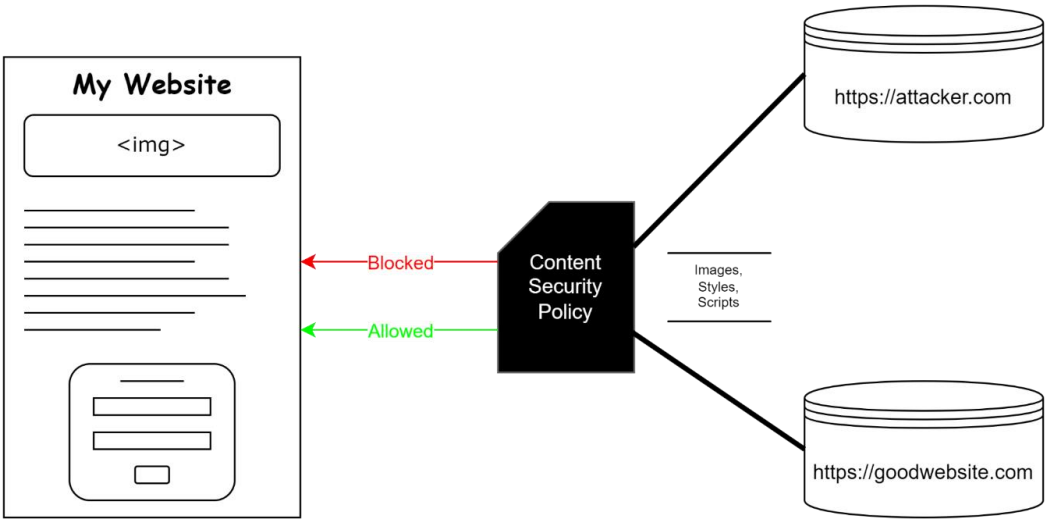
Pattern Addition: New patterns are added to the library, enabling real-time protection against emerging threats.

This approach streamlines XSS DOM attack detection, reducing manual effort and enhancing the application's security.

ARCHITECTURE DIAGRAM:

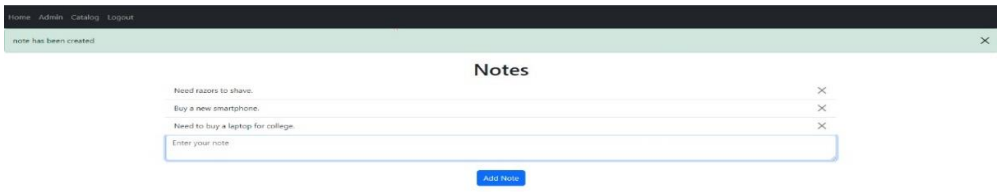


Project Architecture Diagram

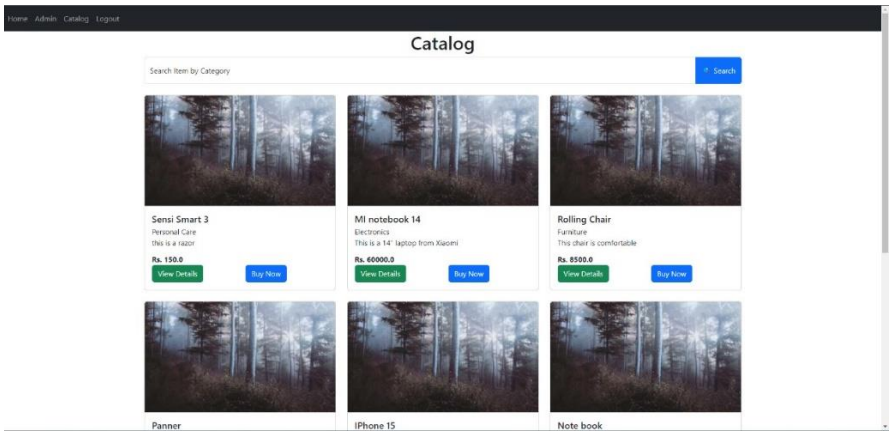


Content Security Policy working

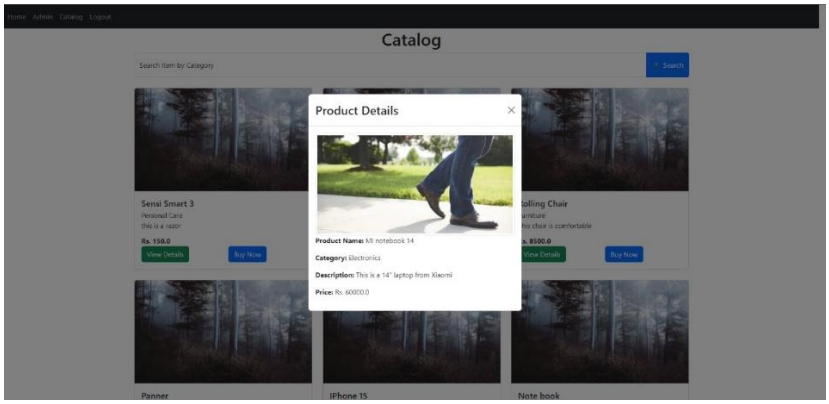
OUTPUT SCREENSHOTS:



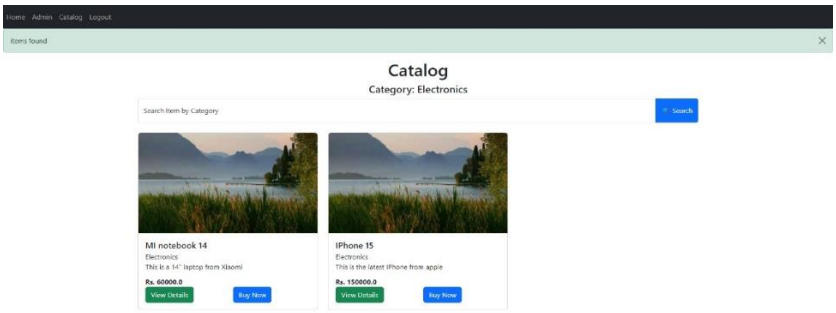
Notes page



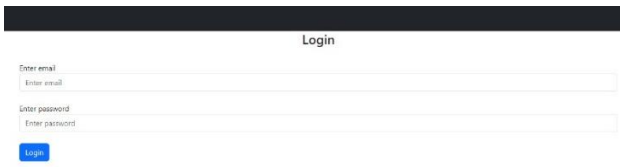
Catalog Page



Product details

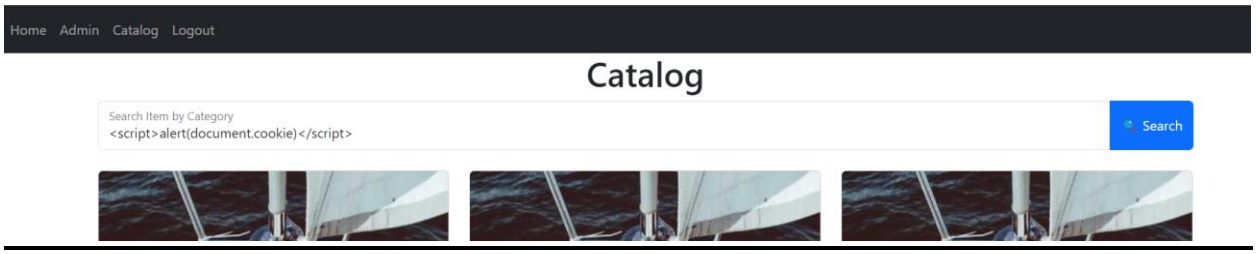


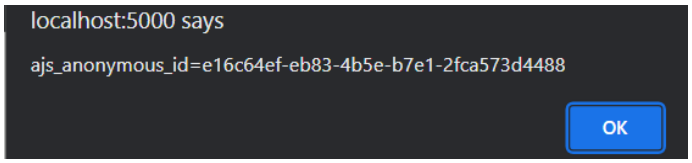
Product filtering based on category



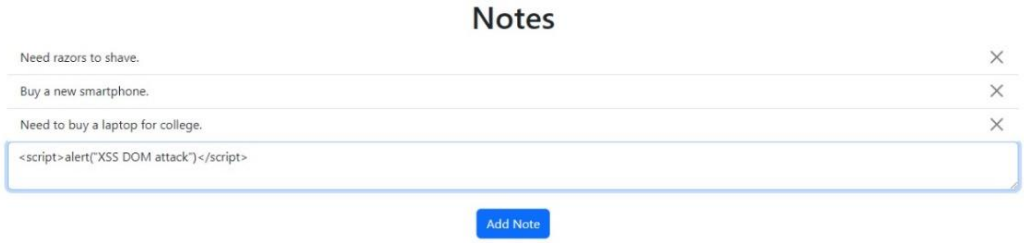
Admin, Login and Signup Pages

Injecting Script in various ways:

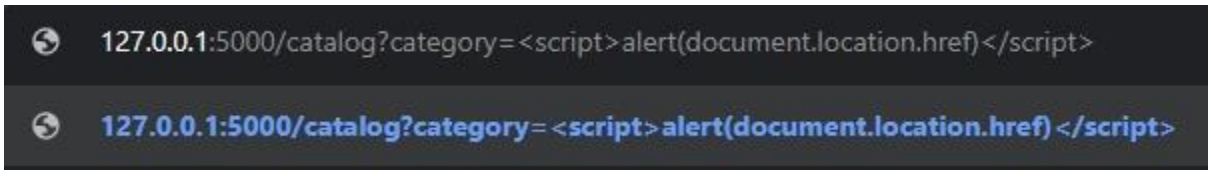


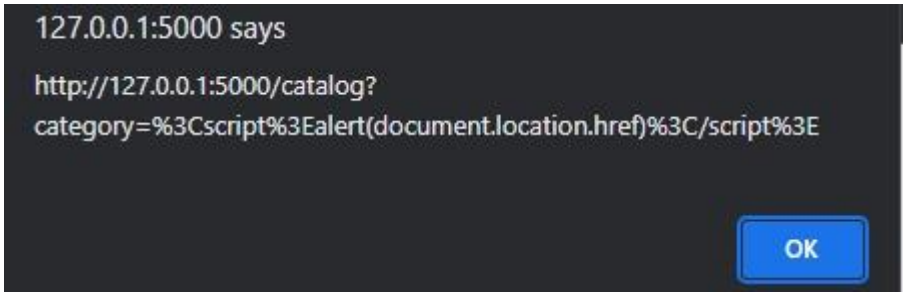


Via Search Bar



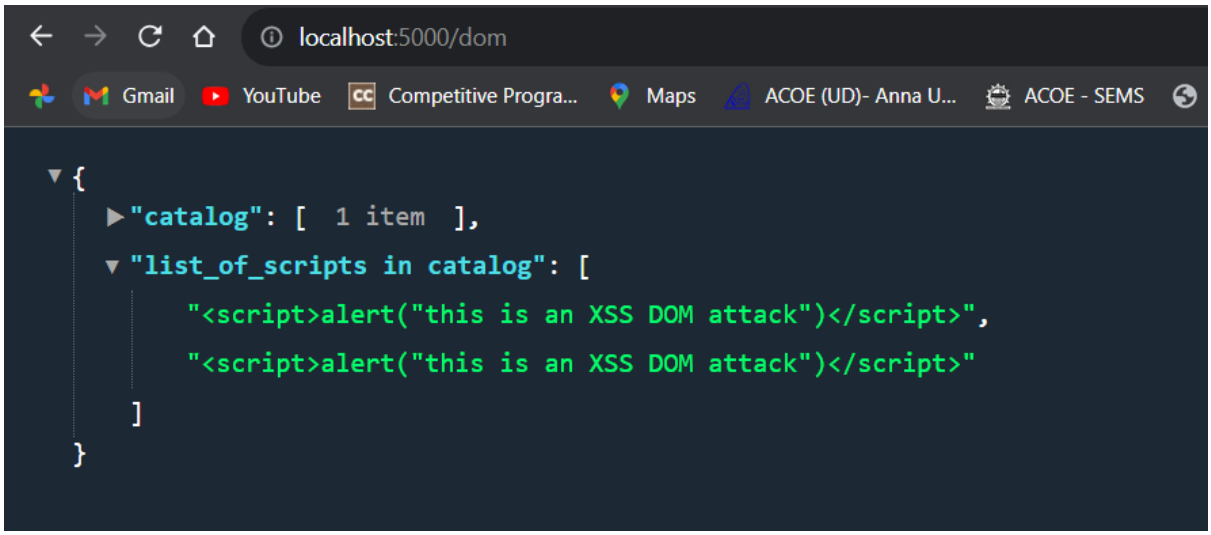
Via Notes Page





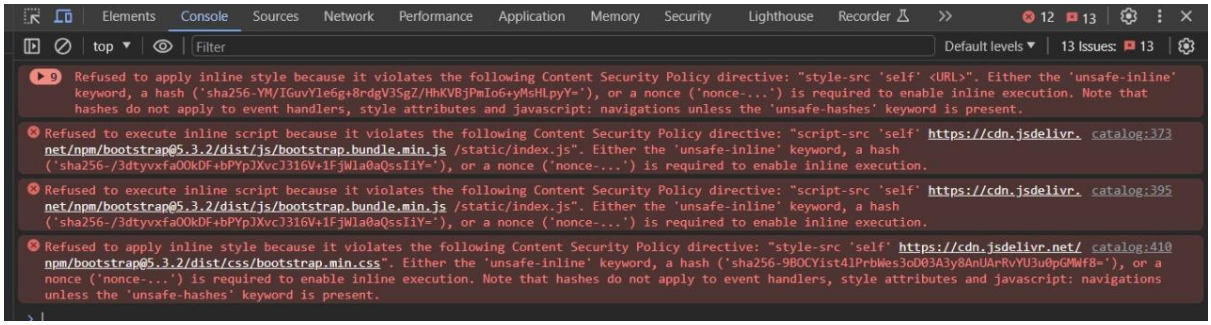
Via URL

Detection of XSS DOM attacks via RegEx (pattern matching):



Pattern Matching

Mitigation of XSS DOM attacks via CSP



Warning on prevention of XSS DOM Attack

REFERENCES:

1. G. Xu et al., "JSCSP: A Novel Policy-Based XSS Defense Mechanism for Browsers," in IEEE Transactions on Dependable and Secure Computing, vol. 19, no. 2, pp. 862-878, 1 March-April 2022, doi: 10.1109/TDSC.2020.3009472.
2. K. Ali, A. Abdel-Hamid and M. Kholief, "Prevention Of DOM Based XSS Attacks Using A White List Framework," 2014 24th International Conference on Computer Theory and Applications (ICCTA), Alexandria, Egypt, 2014, pp. 68-75, doi: 10.1109/ICCTA35431.2014.9521633.

3. S. K. Mahmoud, M. Alfonse, M. I. Roushdy and A. -B. M. Salem, "A comparative analysis of Cross Site Scripting (XSS) detecting and defensive techniques," 2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS), Cairo, 2017, pp. 36-42, doi: 10.1109/INTELCIS.2017.8260024.
4. J. C. Pazos, J. -S. Légaré and I. Beschastnikh, "XSnare: Application-specific client-side cross-site scripting protection," 2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, 2021, pp. 154-165, doi: 10.1109/SANER50967.2021.00023.
5. Z. Jingyu, H. Hongchao, H. Shumin and L. Huanruo, "A XSS Attack Detection Method Based on Subsequence Matching Algorithm," 2021 IEEE International Conference on Artificial Intelligence and Industrial Design (AIID), Guangzhou, China, 2021, pp. 83-86, doi: 10.1109/AIID51893.2021.9456515.
6. A. Shrivastava, S. Choudhary and A. Kumar, "XSS vulnerability assessment and prevention in web application," 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), Dehradun, India, 2016, pp. 850-853, doi: 10.1109/NGCT.2016.7877529.
7. G. Habibi and N. Surantha, "XSS Attack Detection With Machine Learning and n-Gram Methods," 2020 International Conference on Information Management and Technology (ICIMTech), Bandung, Indonesia, 2020, pp. 516-520, doi: 10.1109/ICIMTech50083.2020.9210946.
8. M. Obaidat, J. Brown and A. A. Hayajneh, "Web Browser Extension User-Script XSS Vulnerabilities," 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress

(DASC/PiCom/CBDCom/CyberSciTech), Calgary, AB, Canada, 2020, pp. 316-321, doi: 10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00062.

9. K. Santithanmanan, "The Detection Method for XSS Attacks on NFV by Using Machine Learning Models," 2022 International Conference on Decision Aid Sciences and Applications (DASA), Chiangrai, Thailand, 2022, pp. 620-623, doi: 10.1109/DASA54658.2022.9765122.
10. T. Wang, D. Zhao and J. Qi, "Research on Cross-site Scripting Vulnerability of XSS Based on International Student Website," 2022 International Conference on Computer Network, Electronic and Automation (ICCNEA), Xi'an, China, 2022, pp. 154-158, doi: 10.1109/ICCNEA57056.2022.00043.
11. V. K. Malviya, S. Saurav and A. Gupta, "On Security Issues in Web Applications through Cross Site Scripting (XSS)," 2013 20th Asia-Pacific Software Engineering Conference (APSEC), Bangkok, Thailand, 2013, pp. 583-588, doi: 10.1109/APSEC.2013.85.
12. T. Takeuchi, K. Mouri and S. Saito, "Mocha: Automatically Applying Content Security Policy to HTML Hybrid Application on Android Device," 2017 Fifth International Symposium on Computing and Networking (CANDAR), Aomori, Japan, 2017, pp. 503-509, doi: 10.1109/CANDAR.2017.48.
13. A. Shrivastava, V. K. Verma and V. G. Shankar, "XTrap: Trapping client and server side XSS vulnerability," 2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC), Waknaghat, India, 2016, pp. 394-398, doi: 10.1109/PDGC.2016.7913227.
14. T. K. Nguyen and S. O. Hwang, "Large-Scale Detection of DOM-Based XSS Based on Publisher and Subscriber Model," 2016 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2016, pp. 975-980, doi:

10.1109/CSCI.2016.0187.

15. P. Wang, J. Bangert and C. Kern, "If It's Not Secure, It Should Not Compile: Preventing DOM-Based XSS in Large-Scale Web Development with API Hardening," 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), Madrid, ES, 2021, pp. 1360-1372, doi: 10.1109/ICSE43902.2021.00123.
16. J. Pan and X. Mao, "DomXssMicro: A Micro Benchmark for Evaluating DOM-Based Cross-Site Scripting Detection," 2016 IEEE Trustcom/BigDataSE/ISPA, Tianjin, China, 2016, pp. 208-215, doi: 10.1109/TrustCom.2016.0065.
17. J. Pan and X. Mao, "Detecting DOM-Sourced Cross-Site Scripting in Browser Extensions," 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME), Shanghai, China, 2017, pp. 24-34, doi: 10.1109/ICSME.2017.11.
18. J. Harish Kumar and J. J Godwin Ponsam, "Cross Site Scripting (XSS) vulnerability detection using Machine Learning and Statistical Analysis," 2023 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2023, pp. 1-9, doi: 10.1109/ICCCI56745.2023.10128470.
19. I. Dolnák, "Content Security Policy (CSP) as countermeasure to Cross Site Scripting (XSS) attacks," 2017 15th International Conference on Emerging eLearning Technologies and Applications (ICETA), Stary Smokovec, Slovakia, 2017, pp. 1-4, doi: 10.1109/ICETA.2017.8102476.
20. Samer Attallah Mhana, Jamilah Binti Din and Rodziah Binti Atan, "Automatic generation of Content Security Policy to mitigate cross site scripting," 2016 2nd International Conference on Science in Information Technology (ICSITech), Balikpapan, Indonesia, 2016, pp. 324-328, doi: 10.1109/ICSITech.2016.7852656.
21. P. Wang, B. Á. Guðmundsson and K. Kotowicz, "Adopting Trusted Types

in ProductionWeb Frameworks to Prevent DOM-Based Cross-Site Scripting: A Case Study," 2021 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), Vienna, Austria, 2021, pp. 60-73, doi: 10.1109/EuroSPW54576.2021.00013.

22. A. F. Maskur and Y. Dwi Wardhana Asnar, "Static Code Analysis Tools with the Taint Analysis Method for Detecting Web Application Vulnerability," 2019 International Conference on Data and Software Engineering (ICoDSE), Pontianak, Indonesia, 2019, pp. 1-6, doi: 10.1109/ICoDSE48700.2019.9092614.