A

Project Report On

"OWASP OPERATING SYSTEM"

Submitted in partial fulfillment of the requirement of the University of Mumbai for the Degree of **Bachelor of Engineering**

Computer Engineering (SEM VIII)

By

RAJ MHATRE (45)

YUVRAJ TODANKAR (70)

OM DHUMAL (15)

ATHARVA PATIL (12)

Under the guidance of

Prof. Anup Maurya

Department of Computer Engineering

Wilfred's Education Society

Chhatrapati Shivaji Maharaj Institute of Technology

Shedung, Panvel Dist: Raigad-410206

University of Mumbai

Academic Year 2023-2024

Chhatrapati Shivaji Maharaj Institute of Technology

Department of Computer Engineering

Academic Year 2023-24

CERTIFICATE

This is to certify that

Mr. RAJ MHATRE, Mr. YUVRAJ TODANKAR, Mr. OM DHUMAL,

Mr. ATHARV PATIL

SEM.VIII, BE Computer, Roll No: **45, 70, 15, 12** *has*

Satisfactorily completed the requirements of the Major Project-II entitled

"OWASP OPERATING SYSTEM"

As prescribed by the University of Mumbai Under the guidance of

Prof. Anup Maurya

Guide	HOD	Vice Principal
(Prof. Anup Maurya)	**(Prof. Anup Maurya)**	**(Dr. Manish Sharma)**

Internal Examiner **External Examiner**

ABSTRACT

This study demonstrates how we may create an operating system specifically for web application development and thorough penetration testing. A community-driven initiative called the OWASP Operating Systems project aims to provide a specialized operating system for web application development and penetration testing. The project team is developing and assessing the OS using a rangeof techniques, including user interviews, literature reviews, and prototyping. The creation of a specialized operating system for web application development and penetration testing has advanced significantly under the OWASP Operating Systems project. The project team has created a prototype operating system (OS) with an integrated SIEM system for monitoring and responding to security threats, a full set of integrated penetration testing tools, and a pre-configured development environment. A potential endeavor to create a safe and integrated environment for web development and penetration testing is the OWASP Operating Systems project. Developers may use it to create online apps that are more secure, while security experts could use it to more efficiently find and fix problems. Although theproject is still in its infancy, it has the potential to significantly alter how online applications are created and protected.

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise, does not depend solely on individual efforts buton the guidance, encouragement and cooperation of intellectuals, elders and friends. Several personalities, in their own capacities, have helped me in carrying out this project. We would like totake an opportunity to thank them all.

We would like to thank **Dr. Manish Sharma**, Vice Principal, Chhatrapati Shivaji Maharaj Instituteof Technology, Panvel, for his valuable suggestions and expert advice.

We would like to thank **Prof. Anup Maurya**, Professor and Head of the Department, Computer Engineering, Chhatrapati Shivaji Maharaj Institute of Technology, Panvel, for constant encouragement and support extended towards completing my Project.

We deeply express my sincere gratitude to my guide **Prof. Anup Maurya,** Department of Computer Engineering, Chhatrapati Shivaji Maharaj Institute of Technology, Panvel, for his able guidance, regular source of encouragement and assistance throughout my project period.

Last, but not the least, we would like to thank my peers and friends who provided me with valuablesuggestions to improve my project.

Acknowledge By: -

1. Raj Mhatre
2. Yuvraj Todankar
3. Om Dhumal
4. Atharva Patil

LIST OF FIGURES

Table of Contents

CHAPTER 1

INTRODUCTION

Modern life is impossible without web apps, but they are also a popular target for hackers. The complexity of the online development environment and the continuously changing threat landscape make it difficult to secure web applications. A community-driven initiative to create a specialized operating system for web application development and penetration testing is the OWASP Operating Systems project. This OS may make it simpler for programmers to create safe online applications and for security experts to identify and remedy flaws. The development and penetration testing phases of the OWASP Operating System project are still in their infancy. This OS may make it simpler for programmers to create safe online applications and for security experts to identify and remedy flaws.

A relevant and creative endeavour1 to address the expanding problems with online application security is the OWASP Operating Systems project. The increasing frequency and severity of web application attacks, the growing complexity of the web development environment, and the continuously changing threat landscape all point to the necessity for such a project. Current study emphasizes the difficulties in safeguarding online applications. For instance, 2022 research by Verizon revealed that over 40% of all data breaches involved online apps, and that the most frequent web application assaults were SQL injection, cross-site scripting, and weak authentication. Injection, faulty authentication, and unsafe direct object references were revealed to be the most frequent security flaws in online applications, according to another survey by the OWASP Foundation.

1.1 BACKGROUND

The OWASP Operating System Project (OSPT) is a critical initiative within the broader scope of the Open Web Application Security Project (OWASP). OWASP is a globally recognized, non-profit organization dedicated to improving the security of software. Its mission is to make software security visible so that individuals and organizations can make informed decisions about application security risks.

The OWASP Operating System Project is one of several projects under the OWASP umbrella, each focusing on different aspects of software security. The OSPT is particularly focused on addressing the security concerns and challenges associated with the underlying operating systems of modern computing environments.

Operating systems are the foundation upon which all software and applications run. As such, their security is paramount to the overall security of a computing environment. Vulnerabilities within the operating system can potentially be exploited to compromise the confidentiality, integrity, and availability of all software and data on a system. Operating system security is critical for protecting against a wide range of threats, including malware, privilege escalation attacks, and unauthorized access.

The significance of the OWASP Operating System Project becomes even more evident in today's landscape, where software and data are increasingly moved to cloud-based and virtualized environments. The project addresses these challenges by providing guidance, tools, and resources for improving the security of various operating systems, thereby helping organizations mitigate risks and protect their digital asset.

1.2 Relevance of the Project:

The OWASP Operating System Project (OSPT) holds significant relevance in the field of cybersecurity for several compelling reasons. First and foremost, operating systems serve as the foundational layer for all software and applications in a computing environment. They are the primary interface between hardware and higher-level software, making their security paramount. OSPT's emphasis on enhancing operating system gets contributes to safeguarding critical digital assets, reducing vulnerabilities, and mitigating the risk of malicious attacks that can compromise data integrity and system availability. Next is the importance of the Firewall. We should never turn off the firewall because by doing this malware or any virus can easily get into the PC. We are going to show how the system gets compromise when Laptop/PC firewall is off. We are going to create a payload which contains malicious content. When the system gets compromised the anonymous person gets all access of the victims PC. He can access all the files and camera and many more. A Firewall is simply a program or hardware device that filters the information coming through the internet connection into your private network or computer system.

Lastly, OSPT promotes a culture of collaboration and knowledge sharing within the cybersecurity community. By bringing together experts, administrators, and developers, it facilitates the exchange of insights and practical experiences related to operating system security. This collaboration not only ensures that the latest threats and vulnerabilities are addressed effectively but also fosters innovation in security practices and solutions. Ultimately, OSPT's relevance lies in its contribution to a safer digital world by making operating systems more resilient to attacks and more reliable in their operation, thus benefiting a wide range of stakeholders, from individual users to large enterprises.

1.3 Organization of Project Report:

The material presented in the Project Report is organized into five chapters: Chapter

1: Chapter 1 This is the Introductory chapter.

Chapter 2: Chapter 2 describes the Literature Survey and the Research we did before and after finalizing the project.

Chapter 3: Chapter 3 presents an overview of the proposed system which contains the Algorithms, Design details & Methodology of the system.

Chapter 4: Chapter 4 describes the software and hardware used details of the input received by the system.

Chapter 5: Chapter 5 presents us with the implementation of the project and the Experimental result or discussion.

CHAPTER 2

LITRATURE SURVEY

1. Web Application Security:

- "Web Application Security: Threats, Countermeasures, and Best Practices" by Li, S., & Faynberg, I.
- "A survey of web application security testing tools" by Wong, W., & Hui, L.
- "Web Application Vulnerabilities: A Comprehensive Study" by Nain, A., et al.

2. Penetration Testing:

- "Penetration Testing: The Ultimate Cybersecurity Assessment" by Rajabi, M.
- "Automated Web Application Security Testing" by Abdurakhmanov, F., et al.
- "Penetration Testing in Web Applications: Attacks, Countermeasures, and Trends" by Erkin, Z.

3. Operating System Development:

- "The Design and Implementation of the FreeBSD Operating System" by McKusick, M. K., et al.
- "Linux: A Portable Operating System" by Torvalds, L., & Diamond, D.
- "Operating System Concepts" by Silberschatz, A., et al.

4. Linux-Based Operating Systems:

- "An Overview of the Linux Operating System" by Smith, D. E.
- "Security in Linux and Linux Kernel Space" by Hu, W., & Arbaugh, W. A.
- "Enhancing Security of the Linux Operating System" by Tanenbaum, A. S., & Garfinkel, T.

5. Community Collaboration and Open-Source Development:

- "The cathedral and the bazaar: Musings on Linux and Open Source by an Accidental Revolutionary" by Raymond, E. S.
- "Collaborative Software Development: A new approach to community" by Lerner, J., & Tirole, J.

- "An Empirical Study of the Robustness of Windows NT Applications Using Random Testing " by Voas, J.

2.1 Existing System Survey:

An analysis of the existing system and a gap analysis for the "OWASP Web Penetration Testing Guide" project reveal several limitations in current approaches to web application development and penetration testing. The prevailing methodology typically involves using a combination of disparate tools and environments, which often results in inefficiencies, complexity, and potential security gaps.

Current platforms for web application development and penetration testing, such as Kali Linux, Parrot OS, and OWASP ZAP, are essential tools in the field but often lack some critical features. For instance, while Kali Linux is popular among penetration testers, it may present usability challenges for newcomers, and its capabilities can be overwhelming. Parrot OS, another notable platform, may not enjoy the same reputation and user base as Kali Linux. Additionally, many of these existing systems focus on individual aspects of security or testing, making them less comprehensive for users who need an all-encompassing solution.

Furthermore, there may be issues with tool integration, OWASP compliance, user-friendliness, and documentation in existing systems. Often, adapting to the ever-evolving web application security landscape is a challenge as many systems fall short in terms of providing updates and staying current with the latest security tools and libraries. These limitations emphasize the need for a new system, like the "OWASP Web Penetration Testing Guide," which aims to address these challenges by offering a complete, integrated, and user-friendly environment. By providing a unified platform that aligns with OWASP standards and offers thorough documentation, this project aims to bridge the existing gaps and offer a more efficient and secure solution for web application development and penetration testing.

2.2 Limitations of the Existing System:

The existing systems for web application development and penetration testing, such as Kali Linux, Parrot OS, and OWASP ZAP, exhibit several limitations that necessitate the development of a new and more comprehensive solution. One notable limitation lies in their usability, as Kali Linux, while popular, can be intimidating for newcomers to the field due to its extensive range of tools and complex interfaces. Parrot OS, on the other hand, while a valuable platform for penetration testing, may not have the same level of recognition and user base as Kali Linux, potentially limiting its appeal and support.

Moreover, many of these existing systems are designed to address specific aspects of security or testing, making them less versatile for users who require an all-encompassing solution. The integration of tools and libraries within these platforms can be ineffective, leading to a fragmented user experience and hindering productivity. Additionally, documentation and user support may be lacking, making it challenging for users to fully harness the potential of these systems and effectively address their web application security needs.

Another critical limitation is the difficulty of staying up-to-date with the rapidly evolving web application security landscape. Existing systems often struggle to provide timely updates and may lag behind in incorporating the latest security tools and libraries, leaving users vulnerable to emerging threats. In summary, the limitations of the existing systems emphasize the need for a more holistic and user-friendly solution, such as the "OWASP Web Penetration Testing Guide," to overcome these challenges and offer a more efficient, secure, and adaptable platform for web application development and penetration testing.

One key shortcoming of existing web application development and penetration testing methods is their inability to adapt to changing user demands and preferences. While these platforms provide a wealth of tools and features, they sometimes take a one-size-fits-all approach that may not meet the unique needs or workflows of individual users or organisations. This rigidity can lead to inefficiencies and frustrations, as users may be limited by the predetermined processes and toolsets provided by these platforms. As a result, there is an increasing need for customised solutions that allow users to personalise their development and testing environments to their own interests and goals.

Furthermore, interoperability difficulties provide a significant hurdle to users of existing web application security technologies. While these platforms strive to provide comprehensive solutions for development and testing tasks, they frequently fail to interface effectively with other tools and technologies routinely used in the software development lifecycle. This lack of compatibility might limit collaboration between development and security teams, as well as the adoption of safe development best practices. Furthermore, it complicates the challenge of implementing automated security testing into continuous integration and deployment pipelines, increasing the likelihood that vulnerabilities would go unnoticed until later stages of development or after release. Addressing these interoperability difficulties is critical to improving the efficacy and efficiency of web application security operations.

Moreover, a common drawback of existing web application development and penetration testing systems is the burden of resource-intensive requirements, both in terms of hardware and expertise. Many of these platforms demand substantial computing resources to operate efficiently, making them inaccessible to users with limited hardware resources or constrained budgets. Additionally, proficiency in navigating and utilizing the extensive toolsets provided by these systems often requires specialized training and expertise, posing a barrier to entry for newcomers to the field. As a result, organizations may struggle to allocate the necessary resources and personnel to effectively leverage these platforms for their security testing needs, potentially leaving their web applications vulnerable to exploitation. Simplifying the deployment and usage of web application security tools and reducing the expertise barrier are essential steps toward democratizing access to robust security testing capabilities.

2.3 Problem Statement:

The rapidly changing landscape of web application development and cybersecurity has given rise to a pressing demand for a holistic and unified operating system that can simplify processes and bolster security measures. Traditional methods, which involve employing separate tools and environments for web development and penetration testing, have proven to be less efficient and more complex, potentially leaving vulnerabilities unaddressed.

To address this issue, there is a need for an integrated operating system that can seamlessly support web application development and penetration testing, reducing the complexities of managing multiple tools and ensuring that security gaps are minimized. This proposed system aims to bridge the gap between these two critical domains, offering a comprehensive solution that enhances productivity, streamlines workflows, and ultimately fortifies the security posture of online applications.

2.4 Objectives:

The objectives of the proposed project, which aims to develop an integrated operating system for web application development and penetration testing, can be summarized as follows:

1. Enhanced Security: Develop an operating system that prioritizes security and compliance with industry standards, including OWASP guidelines, to create a secure environment for web application development and penetration testing.

2. Streamlined Workflows: Simplify and accelerate the processes involved in web application development and penetration testing by providing a unified and user-friendly environment, reducing the need for switching between disparate tools.

3. Comprehensive Tool Integration: Integrate a wide range of tools, libraries, and frameworks commonly used in web application development and penetration testing, ensuring that users have access to the resources they need in a cohesive manner.

4. User-Friendly Interface: Design an intuitive and efficient user interface that makes it easy for both beginners and experienced professionals to navigate and utilize the system effectively.

5. Documentation and Training: Create comprehensive documentation and training materials that empower users to understand and make the most of the operating system's capabilities, including guidance on web application security best practices.

6. Adaptability: Ensure the operating system is compatible with various hardware and software configurations, accommodating the evolving needs and preferences of users.

7. Regular Updates: Commit to providing timely updates to the operating system, incorporating the latest security tools, libraries, and addressing vulnerabilities to keep the system current and secure.

8. Cost-Effective Solution: Help users reduce costs associated with the procurement and maintenance of individual tools by offering a single, integrated solution.

9. Improved Security Testing: Enable users to conduct more thorough and effective security testing of web applications, lowering the risk of security breaches and vulnerabilities.

10. Community Building: Foster a community of users, contributors, and supporters to ensure the sustainability of the project, encourage collaboration, and gather feedback for continuous improvement.

11. Alignment with OWASP Standards: Ensure that the operating system aligns with the principles and guidelines provided by OWASP, supporting the organization's mission to enhance web application security.

12. Note Tracking Application: Integrate a web application with a checklist of attack techniques for website penetration testing, allowing users to track their progress and maintain notes on their testing activity

CHAPTER 3

PROPOSED SYSTEM

3.1 Introduction:

The proposed system, known as the "OWASP Web Penetration Testing Guide," is designed to revolutionize web application development and cybersecurity by offering an integrated and comprehensive operating system. By unifying the tools and resources required for web application development and penetration testing, it seeks to provide a more efficient and secure environment for professionals in these fields. The system is built on the foundation of security, aligning with OWASP standards and best practices to ensure robust protection against vulnerabilities and cyber threats.

One of the key features of the proposed system is its user-friendly interface, which aims to simplify the often complex and disparate processes involved in web development and security testing. With a pre-configured development environment, an integrated set of penetration testing tools, and an online notes application for tracking progress, users will find it easier to manage their tasks, enhance productivity, and maintain a detailed record of their testing activities.

Furthermore, the system is committed to staying up-to-date with the evolving landscape of web application security. It will provide regular updates, incorporating the latest security tools and libraries to keep pace with emerging threats. The integration of a web application with an attack technique checklist empowers users to conduct more thorough and effective penetration testing. In summary, the proposed system not only addresses the existing limitations in web application development and penetration testing but also provides a holistic solution that promotes efficiency, security, and adaptability in an ever-changing digital environment.

The proposed OWASP Web Penetration Testing Guide marks a big step forward in addressing the serious issues confronting both web developers and cybersecurity experts. With the rapid expansion of online applications, the necessity for strong security measures has become critical. Traditional approaches to security testing sometimes employ fragmentary solutions, requiring users to negotiate a maze of various tools and resources. This fragmented landscape not only reduces productivity, but also exposes systems to growing risks. Recognising this vital requirement, the OWASP Web Penetration Testing Guide strives to streamline the whole process,

providing a consistent and straightforward framework that allows users to confidently create and protect web applications.

At its heart, the suggested system embraces the characteristics of accessibility, efficiency, and flexibility. By integrating a full range of tools and resources into a single, user-friendly interface, users no longer need to switch between various programmes and workflows. This unified approach not only streamlines the development and testing processes, but it also promotes cooperation and knowledge exchange among cybersecurity professionals.

3.2 Framework / Algorithm:

The framework of the "OWASP Web Penetration Testing Guide" project is structured to provide a solid and organized foundation for web application development and security testing. At its core, it encompasses the development of a custom operating system that integrates a wide array of tools, libraries, and resources. This framework ensures that users have all the essential components at their fingertips, eliminating the need to juggle disparate software and environments. Additionally, the system emphasizes user-friendliness, offering an intuitive interface that caters to both newcomers and experienced professionals, streamlining workflows, and reducing the complexities associated with web development and security testing.

In terms of the algorithmic aspects, the project incorporates algorithms for various tasks, such as security testing, system monitoring, and user interaction. For example, it includes algorithms for vulnerability scanning, intrusion detection, and security assessment. These algorithms play a critical role in automating and optimizing the penetration testing process, making it more efficient and accurate. Furthermore, the system employs algorithms for data analysis and reporting, helping users interpret and act upon the results of security tests. These algorithms are designed to ensure that users can easily identify vulnerabilities and take appropriate measures to mitigate potential risks, all within a unified operating system.

The combination of a well-structured framework and powerful algorithms in this project is intended to not only simplify the complexities of web application development and penetration testing but also enhance security and productivity. By providing users with a comprehensive and integrated solution, the project aims to improve the overall security of web applications while streamlining the processes involved in testing and development.

The structure of the "OWASP Web Penetration Testing Guide" project has been deliberately built to create a smooth and coherent environment for both developers and security specialists. It promotes a comprehensive approach to web application development and security testing by integrating a wide range of tools, libraries and resources. This framework not only streamlines the setup procedure, but it also encourages uniformity and standardisation throughout the development and testing phases. Furthermore, by providing a customisable operating system designed specifically for penetration testers, it enables users to confidently and easily overcome complicated security issues.

3.3 Designing Details:

System Architecture:

The system architecture for the "OWASP Web Penetration Testing Guide" is designed to provide a unified and efficient platform for web application development and penetration testing. It comprises various components that work together to support these critical tasks. Here is an overview of the system architecture:

1. Operating System Core: At the heart of the architecture is the custom-built operating system. This Linux- based OS is configured to be secure and compatible with a wide range of hardware and software configurations. It serves as the foundation for the entire system, providing a stable and reliable environment for users.

2. Development Environment: The operating system includes a pre-configured development environment. This environment encompasses essential web development tools, programming languages, and libraries to facilitate web application creation. It simplifies the process of setting up a development environment, saving users time and effort.

3. Penetration Testing Tools: A central component of the system is the integration of a comprehensive set of penetration testing tools. These tools are carefully selected to cover a broad spectrum of security testing, including vulnerability scanning, network monitoring, and intrusion detection. Users can access these tools from a unified interface, eliminating the need to switch between different applications.

4. User Interface: The architecture incorporates a user-friendly interface that simplifies navigation and tool access. It is designed to be intuitive and efficient, ensuring that users can easily perform tasks related to both web development and penetration testing. The user interface supports a seamless workflow, enhancing productivity.

5. Documentation and Training Materials: To assist users in understanding and effectively utilizing the system, the architecture includes comprehensive documentation and training materials.

These resources provide guidance on best practices in web application security and offer tutorials on how to make the most of the system's features.

6. Online Notes Application: The architecture integrates an online notes application that enables users to track their progress through the OWASP online Penetration Testing Guide. Users can maintain notes on their penetration testing activities and have quick access to their records from within the development environment and penetration testing tools.

7. Community Collaboration: To ensure the sustainability of the project, the architecture includes mechanisms for building a user and contributor community. This community-driven approach encourages collaboration, feedback, and continuous improvement of the system.

8. Regular Updates: Keeping the system current with the evolving web application security landscape is a vital part of the architecture. It includes a mechanism for providing regular updates, incorporating the latest security tools and libraries to address emerging threats.

9. Modular Design Approach: The system architecture follows a modular design approach, allowing each component to function independently while seamlessly integrating with others. This modular structure enhances flexibility, scalability, and maintainability. Each module is designed with well-defined interfaces, enabling easy integration of additional tools or functionalities in the future without disrupting the existing system.

10. Security-Centric Design Principles: Security is prioritized throughout the architecture design process. The operating system core is hardened and configured with security best practices to mitigate common attack vectors. Development tools and penetration testing tools undergo rigorous evaluation to ensure they meet stringent security standards. Additionally, the user interface implements authentication and access control mechanisms to safeguard sensitive functionalities and data.

11. Optimized Performance and Resource Efficiency: The architecture is optimized for performance and resource efficiency to provide a responsive and seamless user experience. Development tools and penetration testing tools are carefully selected and configured to minimize resource consumption without compromising functionality. Moreover, the operating system is streamlined to eliminate unnecessary overhead, maximizing available resources for critical tasks such as application development and security testing.

12. Scalability and Extensibility: Scalability and extensibility are inherent characteristics of the architecture, allowing it to adapt to varying workloads and evolving requirements. The system can

scale horizontally to accommodate increased user demand by deploying additional instances or nodes. Furthermore, the architecture supports the integration of third-party tools and extensions, enabling users to customize their environment according to specific project needs. This flexibility ensures that the system remains relevant and effective in diverse scenarios, from small-scale projects to enterprise-level deployments.

The system architecture is meticulously designed to create a holistic and integrated environment for web application development and penetration testing. It aims to address the limitations of existing systems, promoting security, efficiency, and adaptability in a rapidly changing digital environment.

3.4 System Implementation

The implementation of the "OWASP Web Penetration Testing Guide" system involves several key steps and considerations. First and foremost, it requires the development of a customized operating system, which serves as the foundation of the entire project. This OS is designed to integrate a wide range of security tools, libraries, and development resources essential for web application development and penetration testing. Building the operating system involves selecting a suitable Linux-based platform, configuring hardware support, and creating a user-friendly interface that simplifies navigation and tool access.

A crucial aspect of system implementation is the integration of various algorithms and security mechanisms. These include algorithms for vulnerability scanning, intrusion detection, and data analysis. Implementing these algorithms requires extensive testing and validation to ensure their accuracy and reliability in identifying potential security vulnerabilities. The system should also support a web application that includes an attack technique checklist, allowing users to track their progress and maintain detailed notes on their testing activities.

Moreover, ongoing updates and maintenance are integral to the system's implementation. Staying current with the rapidly evolving web application security landscape is vital. Regular updates must be provided to incorporate the latest security tools, libraries, and patches to address vulnerabilities. The project should actively engage a community of users and contributors to ensure that the system remains robust, secure, and adaptable to emerging threats.

Finally, the implementation process involves creating comprehensive documentation and training materials to assist users in understanding and effectively utilizing the operating system. The availability of these resources is essential to ensure that users can make the most of the system's capabilities. With proper implementation, the project can fulfill its objectives of enhancing web application security, streamlining workflows, and simplifying the often-complex processes of web development and penetration testing.

3.5 Methodology

The development of the "OWASP Web Penetration Testing Guide" system follows a structured methodology to ensure its success and effectiveness. This methodology encompasses various phases and processes to achieve the project's objectives.

1. Requirements Analysis: The first step in the methodology involves gathering and analyzing the requirements of users, security experts, and developers. Understanding their needs and expectations is crucial to tailor the system to their specific needs, ensuring that it aligns with industry standards, OWASP guidelines, and user preferences.

2. System Design: Once the requirements are understood, the system's design phase commences. This involves planning the architecture, selecting the operating system, designing the user interface, and integrating the necessary penetration testing tools. The design phase aims to create a blueprint for the entire system, including its user-friendly interface, integration capabilities, and adaptability to evolving security needs.

3. Development: The development phase involves building the operating system, customizing it to include the pre-configured development environment, and integrating the selected penetration testing tools. It also includes the implementation of algorithms for vulnerability scanning, intrusion detection, and data analysis. Development efforts are guided by a commitment to security, user-friendliness, and alignment with OWASP standards.

4. Documentation and Training: This phase runs concurrently with development and focuses on creating comprehensive documentation and training materials. Users need access to instructional resources that empower them to understand and efficiently utilize the system's capabilities. These materials cover web application security best practices, tutorials, and guidelines.

5. Testing and Quality Assurance: Rigorous testing and quality assurance procedures are applied to verify the system's functionality, security, and reliability. This phase includes vulnerability testing, performance testing, and user testing to identify and address any issues or deficiencies.

6. User Training and Adoption: Before the system's release, training sessions and user familiarization are conducted to ensure that users can effectively navigate and utilize the system. Training sessions help users become proficient in web application development and penetration testing.

7. Deployment and Regular Updates: Once the system is ready for use, it is deployed to users. Ongoing maintenance and regular updates are integral to the methodology. The system must stay current with the evolving web application security landscape, ensuring that users have access to the latest tools, libraries, and security patches.

8. Community Building: The project actively encourages community building, involving users and contributors in discussions, feedback, and collaborative development efforts. A thriving community can provide

CHAPTER 4

EXPERIMENT SETUP

4.1 HARDWARE AND SOFTWARE

4.1.1 Hardware

Processor: Any Processor

RAM: more than 2GB

Hard Disk: 10 GB

Memory: 6 GB RAM

CHAPTER 5

IMPLEMENTATION

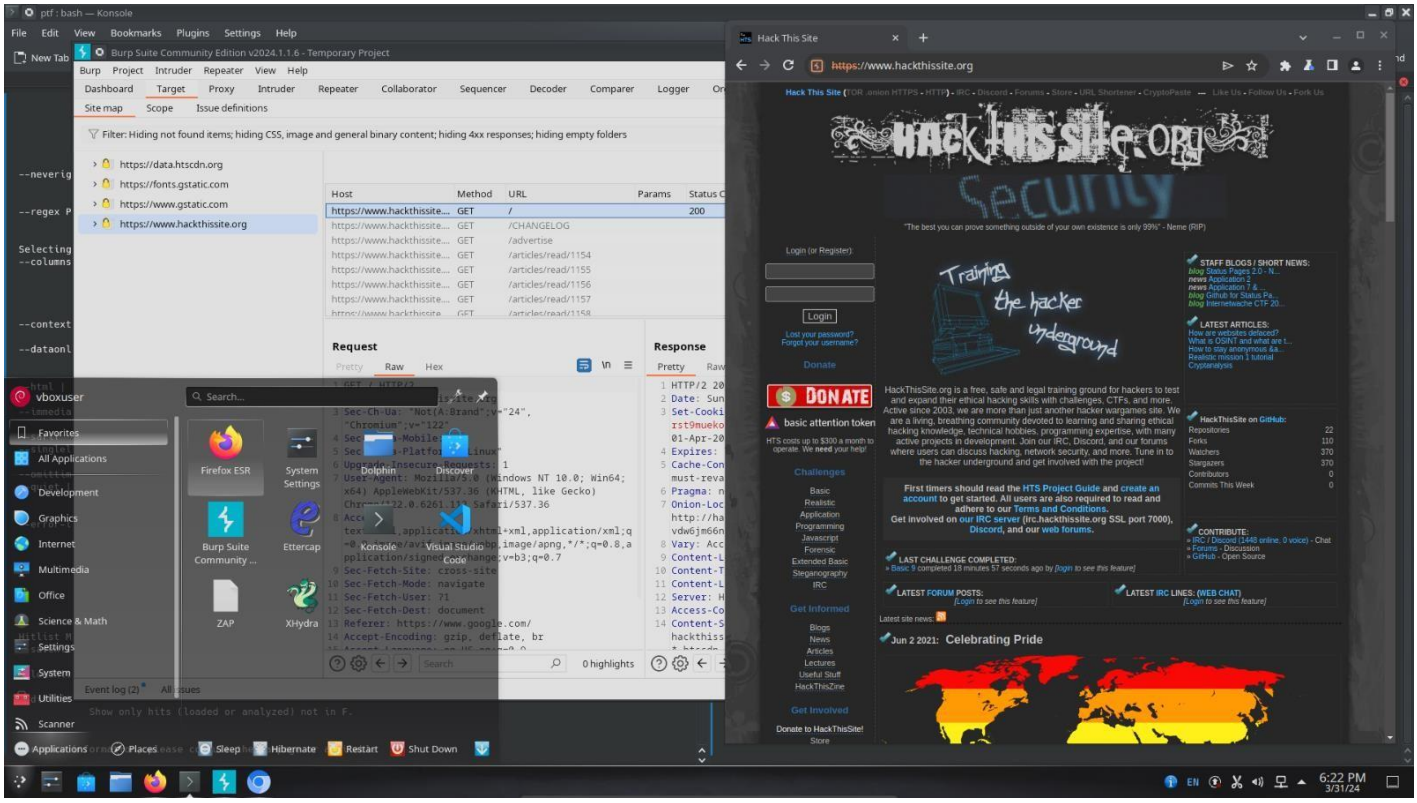

Fig 5.1. Burp suite software for action

- Burp Suite helps you find vulnerabilities and weaknesses in websites or web applications before bad actors do. It's like a detective tool for websites, sniffing out potential problems that could be exploited by hackers.
- In this we are going to perform various attack on the website www.hackthesite.com.
- On this website we are going to perform various attacks like XSS, CSRF, SSRF, SQL Injection, Session Hijacking, File Inclusion.
- With so much sensitive information being stored and transmitted online, it's crucial for websites to be secure. Burp Suite helps developers and security professionals identify and fix vulnerabilities before they can be exploited by hackers, ultimately making the internet safer for everyone.

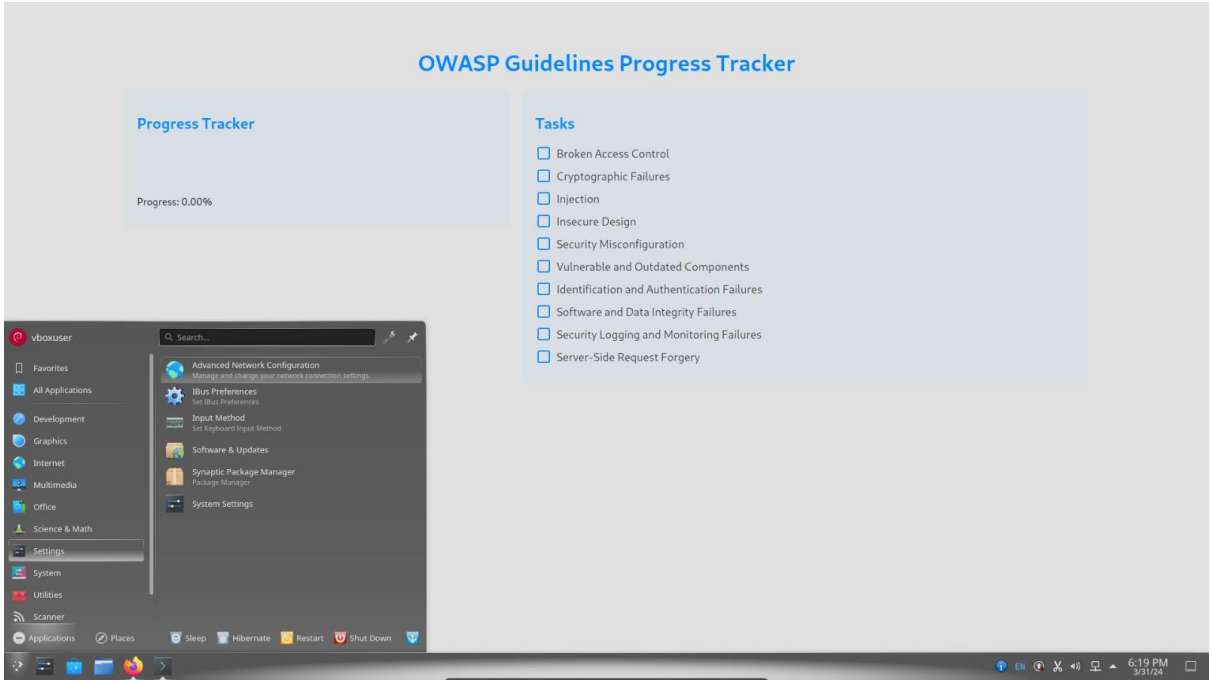

Fig 5.2 OWASP Guidelines Progress Tracker

- Creating an OWASP (Open Web Application Security Project) guideline progress tracker can be a valuable tool for organizations striving to improve the security of their web applications. Here's a simple outline of how you might structure it:
- Creating an OWASP (Open Web Application Security Project) guideline progress tracker can be a valuable tool for organizations striving to improve the security of their web applications. Here's a simple outline of how you might structure it:
- OWASP Guidelines: Start by listing the various guidelines provided by OWASP. These may include the OWASP Top 10, OWASP Application Security Verification Standard (ASVS), and other relevant OWASP documents.
- Organize the guidelines into categories and subcategories. For example, the OWASP Top 10 includes categories like Injection, Broken Authentication, Sensitive Data Exposure, etc.
- In this there all the OWASP Top10 web application security and checklist of progress tracker also gets when OWASP Top 10 gets updated.

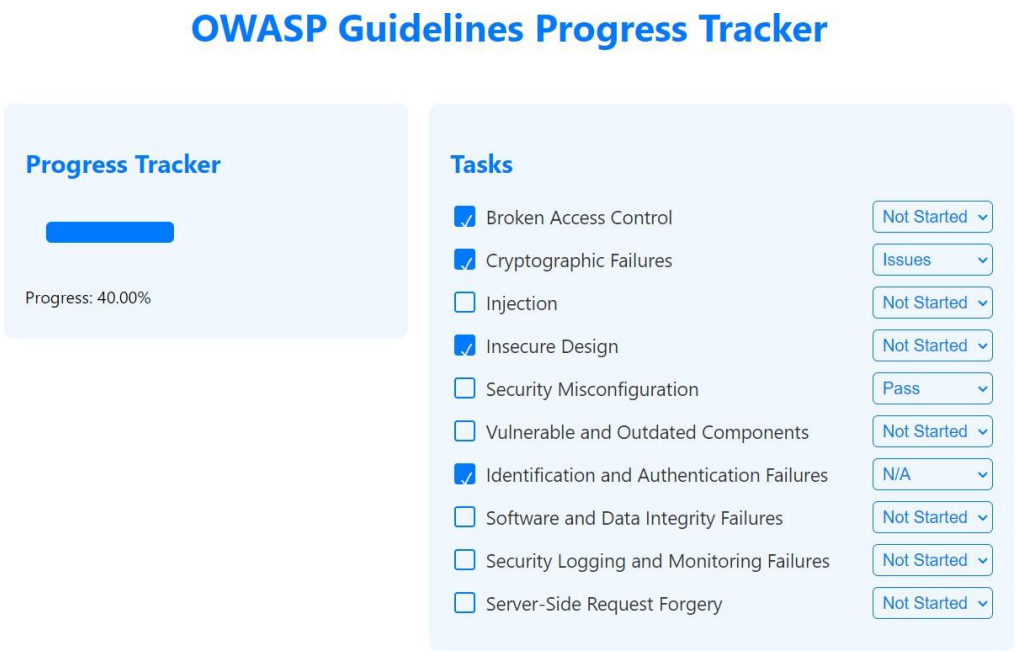

Fig 5.3 Working of Progress tracker

- Making a Checklist: First, you make a simple list of all the important security guidelines provided by OWASP. It's like jotting down all the things you need to do to keep your website safe.
- Then, you group these guidelines into categories like "Protecting Against Hacking" or "Keeping User Data Safe." This helps you see what areas of security you need to focus on.
- Next, you go through each guideline and mark whether it's already in place, halfway there, or not started yet. It's like ticking off items on a to-do list.
- You keep updating the tracker regularly to see how far you've come. It's like watching a progress bar fill up as you get closer to finishing a game level.

Fig No. 5.4 Use of Nmap (Network Mapper)

- Nmap, short for "Network Mapper," is a powerful open-source tool used for network exploration and security auditing. It's like a Swiss Army knife for network administrators, security professionals, and ethical hackers. Here's a breakdown of what Nmap is, how it works, and some of the attacks it can be used for:
- Network Exploration: Nmap allows users to discover hosts and services on a computer network. It can scan large networks to find active devices, identify open ports, and gather information about the services running on those ports.
- Port Scanning: One of Nmap's primary functions is port scanning. It sends packets to target hosts and analyses the responses to determine which ports are open, closed, or filtered. This information helps in understanding the network's topology and identifying potential entry points for attackers.
- Service Version Detection: Nmap can also determine the version and type of services running on open ports. This helps in identifying vulnerable services that may be outdated or misconfigured, making them susceptible to exploitation.
- OS Fingerprinting: Nmap can attempt to identify the operating system running on a target host based on subtle differences in how it responds to network probes. This information is valuable for understanding the target environment and tailoring attacks accordingly.

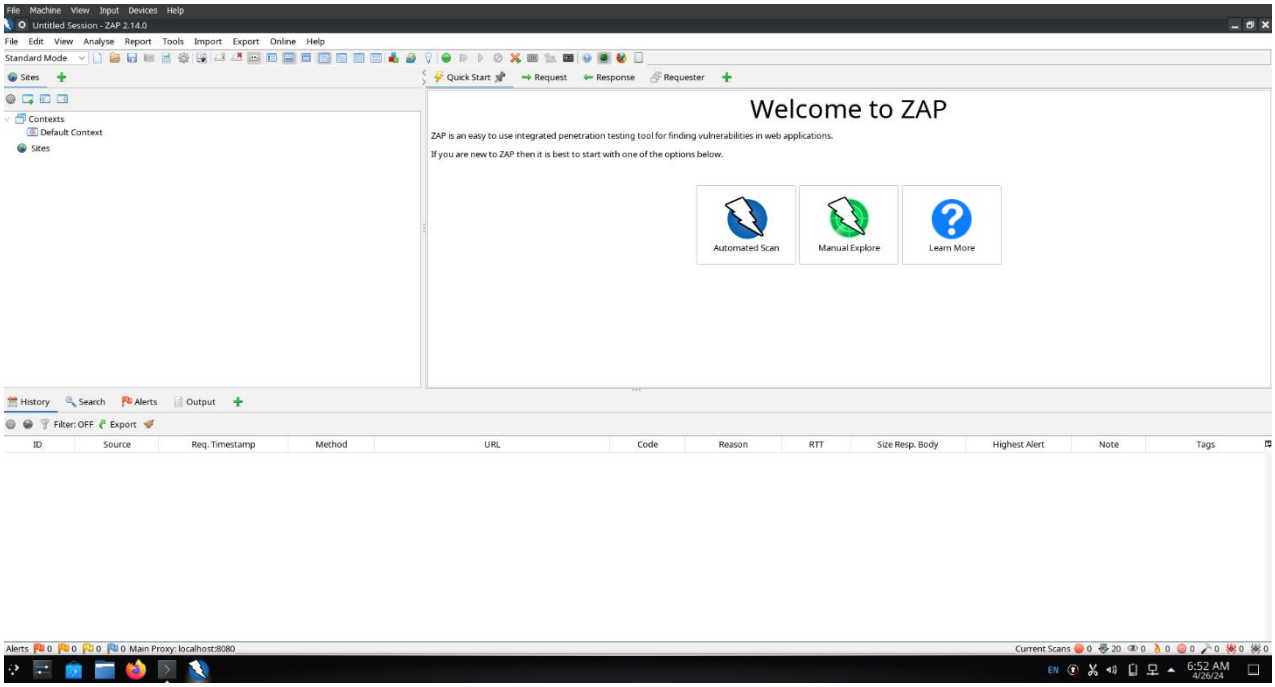

Fig No. 5.5 OWASP ZAP

- OWASP ZAP (Zed Attack Proxy) is a free and open-source web application security scanner and testing tool. It's designed to help developers and security professionals find security vulnerabilities in web applications during development and testing phases.
- OWASP ZAP is developed by the Open Web Application Security Project (OWASP) to provide developers and security professionals with an easy-to-use tool for finding vulnerabilities in web applications.
- ZAP offers a wide range of features, including automated scanning, passive scanning, manual testing, and more. It can be used to identify vulnerabilities such as SQL injection, cross-site scripting (XSS), security misconfigurations, and other common web application flaws.

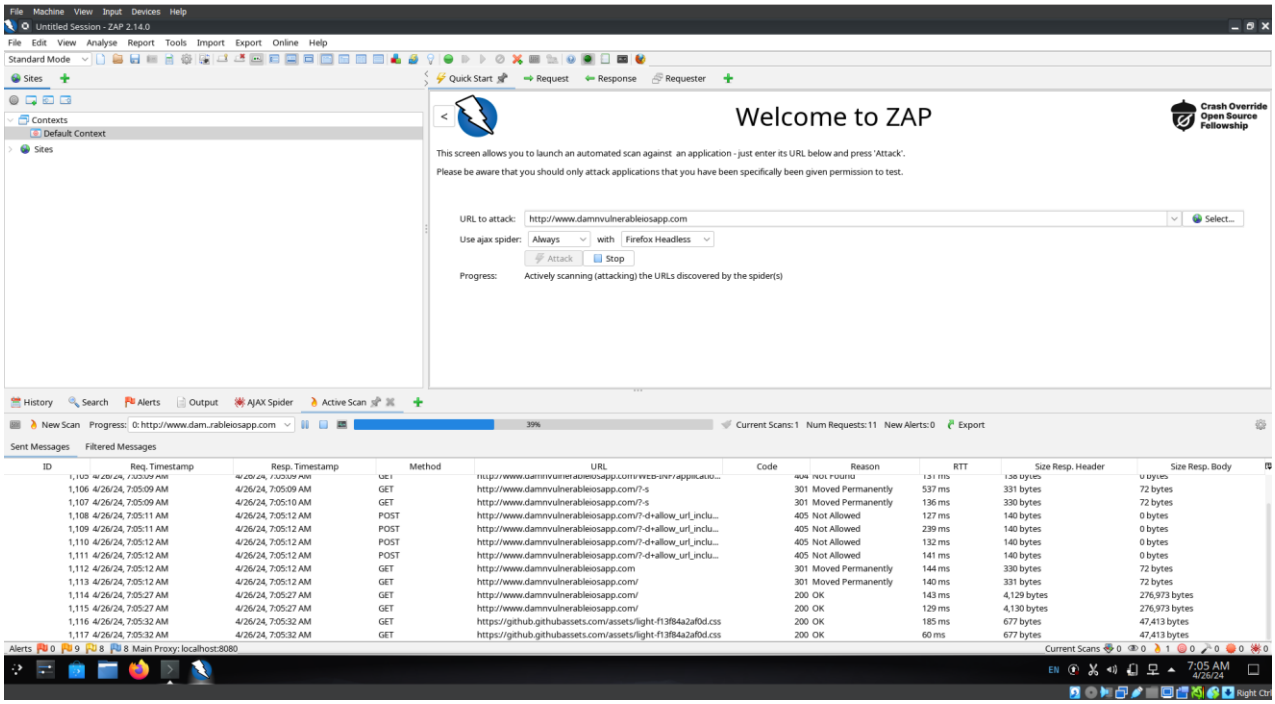

Fig No. 5.6 ZAP attack on a website

- In this scenario, we are conducting an automated scan on the website "www.damnvulnerableioapp.com". Before initiating the attack, we have configured the filter to utilize the Ajax Spider option set to "always", employing "Firefox headless" mode.

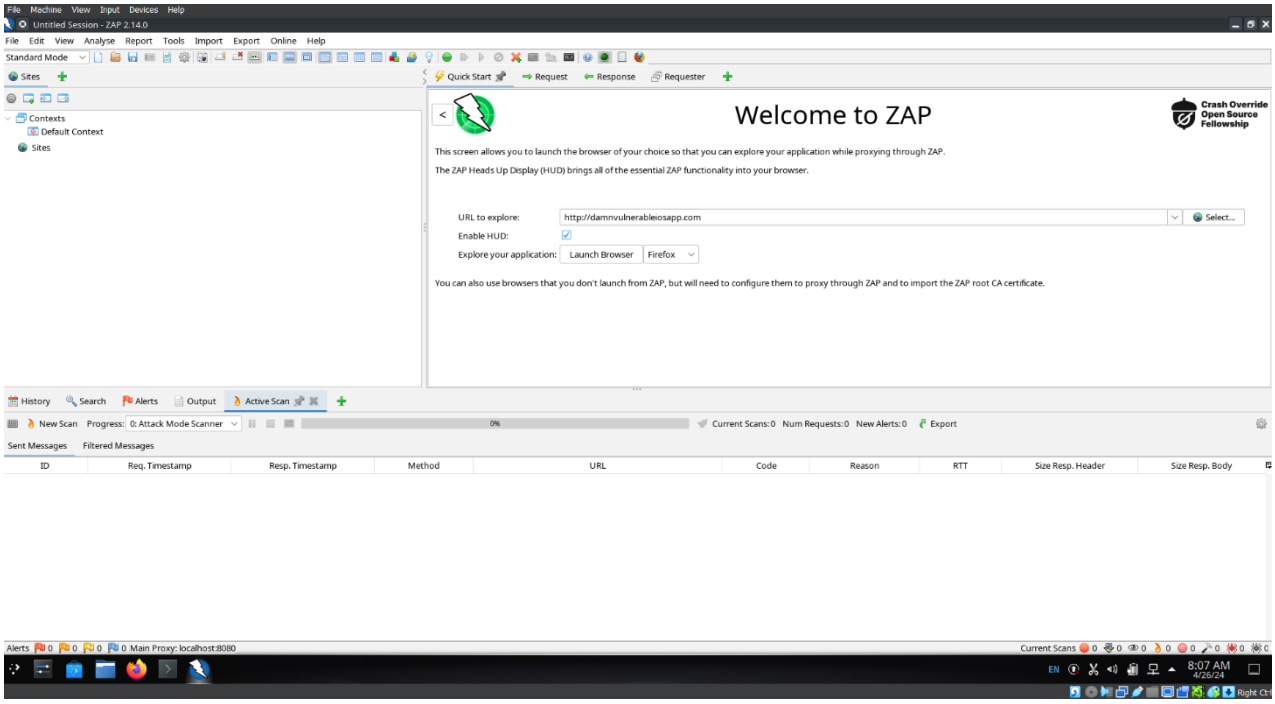

Fig No. 5.7 ZAP Manual Explore Mode

- In OWASP ZAP (Zed Attack Proxy), the Manual Explore mode is a feature that allows users to manually explore and interact with web applications to identify security vulnerabilities.
- In Manual Explore mode, users interact with the target web application through their browser, just like a regular user would. They can navigate through different pages, submit forms, and interact with various elements of the web application.
- Users can document their findings and observations in ZAP, including screenshots, notes, and annotations. This information can be used to generate comprehensive reports for further analysis and remediation.

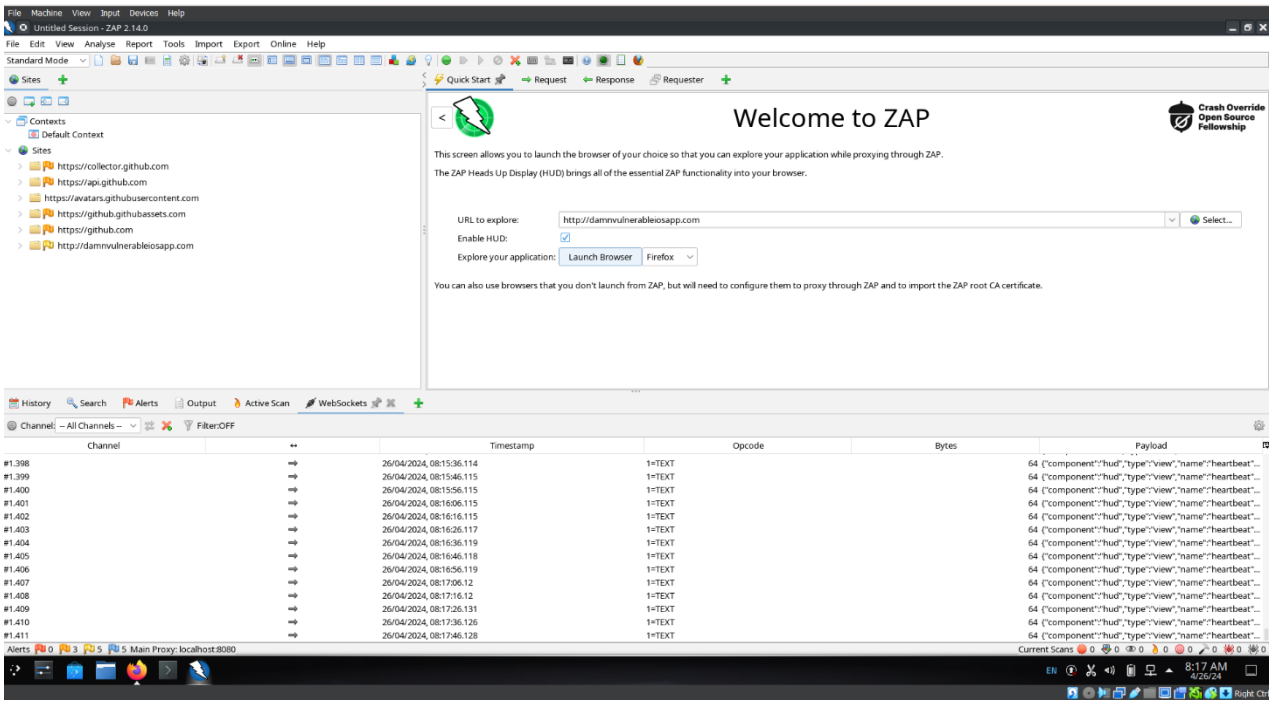

Fig No. 5.8 Manual Explore Attack Mode

- In Manual Explore mode in OWASP ZAP, when attacking the website "www.damnvulnerableiosapp.com," here are some points to consider:
- Target Analysis: Start by analysing the target website, "www.damnvulnerableiosapp.com," to understand its structure, functionality, and potential areas of vulnerability. This includes identifying different pages, input fields, forms, and functionalities available on the website.
- Configure your browser to route traffic through ZAP's proxy server to intercept HTTP and HTTPS requests and responses between the browser and the target website.

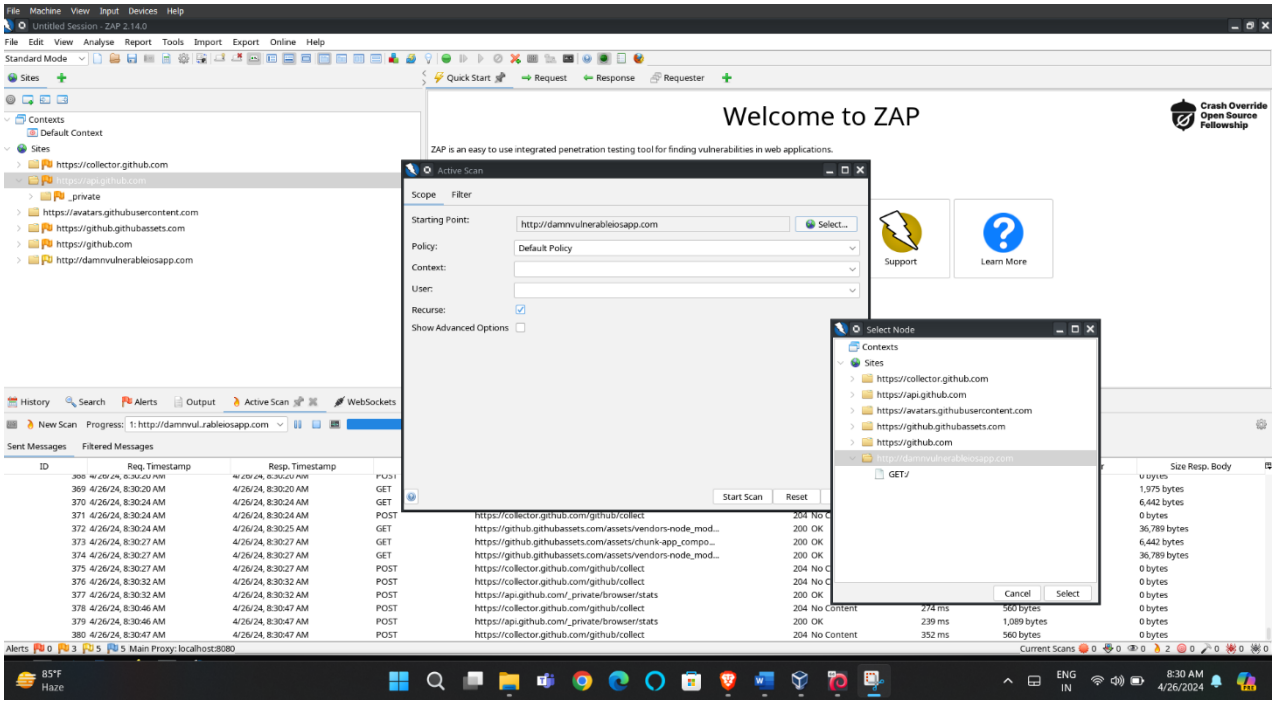

Fig No. 5.9 Active Scan in Manual Explore

- In OWASP ZAP's Manual Explore mode, the active scan feature allows you to perform automated security scans on specific parts of the target web application. Here's how it works:
- Interception of Requests: In Manual Explore mode, OWASP ZAP intercepts all HTTP and HTTPS requests and responses between your browser and the target web application.
- As you navigate through the web application manually, ZAP captures the requests and responses in real-time. This allows you to interact with the application and identify potential vulnerabilities.
- When conducting an active scan on the web application "http://damnvulnerableiosapp.com" using OWASP ZAP, here are some points to consider:
- If the application requires authentication, ensure that ZAP is configured to handle authentication properly. Provide necessary login credentials or configure session handling rules to authenticate with the application before initiating the scan.

5.2 Code

5.2.1 ProgressTracker.js

```
import React, { useState } from 'react';
import './CircularProgressTracker.css';

const CheckBoxProgressTracker = () => {
  const checkboxes = [
    { id: 1, label: 'Broken Access Control' },
    { id: 2, label: 'Cryptographic Failures' },
    { id: 3, label: 'Injection' },
    { id: 4, label: 'Insecure Design' },
    { id: 5, label: 'Security Misconfiguration' },
    { id: 6, label: 'Vulnerable and Outdated Components' },
    { id: 7, label: 'Identification and Authentication Failures' },
    { id: 8, label: 'Software and Data Integrity Failures' },
    { id: 9, label: 'Security Logging and Monitoring Failures' },
    { id: 10, label: 'Server-Side Request Forgery' },

  ];

  const [progress, setProgress] = useState(0);
  const [checkedCount, setCheckedCount] = useState(0);
```

```
  const handleChange = (isChecked) => {
    const newCount = isChecked ? checkedCount + 1 : checkedCount - 1;
    setCheckedCount(newCount);
    const newProgress = (newCount / checkboxes.length) * 100;
    setProgress(newProgress);
  };

  const completionMessage =
    progress === 100 ? <p className="completion-message">Progress Completed!</p> : null;

  return (
    <div className="wrapper">
      <h1 className="website-heading">OWASP Guidelines Progress Tracker</h1>
      <div className="container">
        <div className="progress-column">
          <h2 className="column-heading">Progress Tracker</h2>
          <div
            className="progress-bar-container"
            style={{
              backgroundColor: '#f0f7ff',
              borderRadius: '10px',
              padding: '20px',
              marginBottom: '20px',
```

```
        }}
      >
        <div
          className="progress-bar"
          style={{
            width: `${progress}%`,
            backgroundColor: progress === 100 ? 'green' : '#007bff',
            borderRadius: '5px',
            height: '20px',
            transition: `width 0.5s ${progress === 0 || progress === 100 ? 'ease' : 'linear'}`,
          }}
        />
      </div>
      <p className="progress-text">Progress: {progress.toFixed(2)}%</p>
      {completionMessage}
    </div>
    <div className="checkbox-column">
      <h2 className="column-heading">Tasks</h2>
      {checkboxes.map((checkbox) => (
        <div key={checkbox.id} className="checkbox-item">
          <input
            type="checkbox"
            id={`checkbox-${checkbox.id}`}
            onChange={(e) => handleChange(e.target.checked)}
```

```
            />
            <label htmlFor={`checkbox-${checkbox.id}`}>{checkbox.label}</label>
            <select className="dropdown">
              <option value="option1">Not Started</option>
              <option value="option2">Pass</option>
              <option value="option3">Issues</option>
              <option value="option4">N/A</option>
            </select>
          </div>
        ))}
      </div>
    </div>
  </div>
 );
};

export default CheckBoxProgressTracker;
```

5.2.2 ProgressTracker.css

```
.wrapper {
  display: flex;
  flex-direction: column;
  align-items: center;
```

```
  margin-top: 50px;
}
.website-heading {
  color: #007bff;
  font-size: 36px;
  margin-bottom: 20px;
  text-align: center;
}
.container {
  display: flex;
  justify-content: center;
  align-items: flex-start;
  width: 80%;
}
.progress-column {
  flex: 2; /* Increased width */
  background-color: #f0f7ff;
  border-radius: 10px;
  padding: 20px;
  margin-right: 20px;
  margin-top: 30px;
}
.checkbox-column {
  flex: 3; /* Increased width */
```

```
  background-color: #f0f7ff;
  border-radius: 10px;
  padding: 20px;
  overflow-y: auto; /* Enable scrolling if needed */
  margin-top: 30px;
}
.column-heading {
  color: #007bff;
  font-size: 24px;
  margin-bottom: 20px;
}
.checkbox-item {
  margin-bottom: 10px;
  display: flex;
  align-items: center;
}
input[type='checkbox'] {
  margin-right: 10px;
  appearance: none;
  width: 20px;
  height: 20px;
  border: 2px solid #007bff;
  border-radius: 3px;
  cursor: pointer;
```

```
  position: relative;
  transition: background-color 0.3s, border-color 0.3s;
}
input[type='checkbox']:checked {
  background-color: #007bff;
  border-color: #007bff;
}
input[type='checkbox']:checked::after {
  content: '\2713';
  font-size: 16px;
  color: white;
  position: absolute;
  top: 1px;
  left: 4px;
}
label {
  font-size: 18px;
  color: #333;
  cursor: pointer;
}
.progress-bar-container {
  background-color: #ddd;
  border-radius: 5px;
  height: 20px;
```

```
  margin-bottom: 10px;
  overflow: hidden; /* Hide overflowing progress */
}
.progress-bar {
  background-color: #007bff;
  border-radius: 5px;
  height: 100%;
}
.progress-text {
  margin-bottom: 10px;
}
.completion-message {
  color: green;
  font-weight: bold;
}
.checkbox-item {
  display: flex;
  align-items: center;
}
.dropdown {
  margin-left: auto; /* Align dropdowns to the right */
  background-color: #f0f7ff; /* Matching theme color */
  border: 1px solid #007bff; /* Matching theme color */
  border-radius: 5px;
```

```
  padding: 5px;
  color: #007bff; /* Matching theme color */
  font-size: 16px;
  cursor: pointer;
}
.dropdown:focus {
  outline: none; /* Remove focus outline */
}
/* CircularProgressTracker.css */
.wrapper {
  max-width: 1200px; /* Set maximum width */
  margin: 0 auto; /* Center the content */
}
.container {
  display: flex;
  flex-direction: column;
}
@media (min-width: 768px) {
  .container {
    flex-direction: row; /* Change to row layout on larger screens */
    justify-content: space-between; /* Space out the columns */
  }
  .progress-column {
    width: 60%; /* Adjust width for larger screens */
```

```
  }
  . checkbox-column {
    width: 35%; /* Adjust width for larger screens */
  }
}
```

CHAPTER 6

CONCLUSION

In conclusion, the "OWASP Web Penetration Testing Guide" project represents a promising endeavor to revolutionize web application development and cybersecurity. By providing a unified and integrated operating system, this project aims to streamline workflows, enhance security measures, and simplify the often-complex processes of web development and penetration testing. The holistic approach, incorporating a user-friendly interface, a wide array of penetration testing tools, and alignment with OWASP standards, sets the stage for a comprehensive and secure environment.

The project's methodology, encompassing requirements analysis, system design, development, documentation, testing, and community engagement, underscores its commitment to a systematic and user- focused approach. These methodologies are pivotal in achieving the project's objectives of improving web application security, efficiency, and user productivity.

With the right hardware and software requirements in place, including a powerful processor, adequate RAM, a suitable hard disk, and a Wi-Fi adapter for penetration testing, the project sets the stage for effective web application security testing. It is essential to continue to adapt and update the system to stay current with the evolving web application security landscape and to engage a community of users and contributors to ensure the project's sustainability and success.

In essence, the "OWASP Web Penetration Testing Guide" project holds great promise in enhancing the security and efficiency of web application development and penetration testing, ultimately contributing to the overall resilience of online applications in an ever-changing digital landscape.

CHAPTER 7

REFERENCE

1. Anjalee Sahu, Asst.prof.Shrikant Singh and HOD Rahul Chawda, “Research Paper On Operating System,” International Journal Of Creative Reserch Thoughts (IJCRT), | Volume 9, Issue 6 June 2021 | ISSN: 2320-2882

2. Michael Barabanov, “A Linux based Real Time Operating System”, New Mexico Institute of Mining and Technology Socorro New Mexico June 1, 1997.

3. Marko Boras, Josip Balen, Krešimir Vdovjak, “Performance Evaluation of Linux Operating Systems”, Conference Paper · October 2020 DOI: 10.1109/SST49455.2020.9264055.

4. Roshni Thangavel, Ankita Maiti, Karen Pinto and Prof. Tamil Priya D, “Comparative Research on Recent Trends, Designs, and Functionalities of Various Operating Systems”, International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181 Vol. 8 Issue 10, October-2019.

5. Sahar Badri and Daniyal Alghazzawi, “Security and Performance through Operating System Services; Development of an Anti-Hacking System”, Computer and Information Science; Vol. 15, No. 4; 2022 ISSN 1913-8989 E-ISSN 1913-8997.

6. Ouissem Ben Fredj , Omar Cheikhrouhou , Moez Krichen , Habib Hamam and Abdelouahid Derhab, “An OWASP Top Ten Driven Survey on Web Application Protection Methods”, Conference Paper · November 2020.

7. Virpal Kaur PG Student, Guru Kashi University, India, “Operating System -Review Paper”, International Journal of Research Publication and Reviews ISSN 2582-7421.

8. https://owasp.org/ The Open Worldwide Application Security Project.

9. https://www.linux.org/ Zenity is a command-line utility for Linux and Unix-like operating systems.