**Proposal Defense Document**

**Group No. 25**

**Project Name: Automated Vulnerability Scanner**

**BS Computer Science, Batch (13th)**

**Supervised by: Mam Nadia Hussain**

**Designation**

**BBSUL**

**Submitted by:**

|  |  |
| --- | --- |
| **B2231014** | **Aneel Kumar** |
| **B2231051** | **Lal Ranchhordas** |
| **B2231030** | **Gulabchand** |
|  |  |
| **C:\Users\bbsul\Desktop\log.jpg** | |

Benazir Bhutto Shaheed University, Lyari

Department of Computer Science & Information Technology

[**http://www.bbsul.edu.pk**](http://www.bbsul.edu.pk)

1. **ABSTRACT**

We propose a novel technique for safe patch fingerprinting to automate vulnerability scanning of network servers. This technique enables the automatic discovery of inputs that differentiate vulnerable from patched servers for newly discovered vulnerabilities. The proposed system ensures rapid updates to vulnerability scanning tools, allowing administrators to secure their networks more efficiently. To maintain safety and ethical standards, we implement measures to reject inputs with malicious side effects.

Our framework leverages delta execution to validate the discriminative property and safety of detected inputs. A fuzzier is utilized to automate input discovery, further optimizing the scanning process. To demonstrate its effectiveness, we present a case study on the Heartbleed vulnerability. Additionally, we evaluate the remediation performance of seven automated vulnerability scanners, analyzing authenticated and unauthenticated scans. Results indicate that while vulnerability scanners are effective assessment tools, manual verification is required for complete accuracy, and remediation guidelines can be complex to follow. Our study further highlights that scanners with better remediation accuracy tend to exhibit superior detection capabilities, albeit with a higher tendency for false alarms.

Given the increasing prevalence of web-based threats, our research also explores the application of black-box testing to detect common vulnerabilities such as SQL Injection (SQLi), Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF). We developed a detection tool, Web Vulnerabilities Finder (WVF), which automatically analyzes websites for security flaws. Experimental results confirm the effectiveness of this approach, detecting numerous instances of exploitable vulnerabilities across multiple domains.

By integrating modern security evaluation techniques with automated scanning tools, this project aims to enhance network security through timely vulnerability identification and mitigation.

1. **PROBLEM STATEMENT**

With the **rapid increase in cyber threats**, organizations are facing significant challenges in ensuring the **security of their networks**. Cybersecurity risks, including data breaches, ransomware attacks, and system vulnerabilities, are continuously evolving, and many businesses struggle to keep up with the speed and sophistication of these threats. While large enterprises often have dedicated cybersecurity teams, **small and medium-sized enterprises (SMEs)** are disproportionately impacted by these threats due to limited resources and budget. Many of these organizations lack access to **cost-effective**, **efficient**, and **user-friendly vulnerability scanning tools** that can adequately assess the security of their networks and systems.

Current solutions available in the market are often **expensive**, requiring significant investment from businesses, particularly SMEs, that cannot afford costly enterprise-level security tools. Additionally, many of these tools **demand extensive technical expertise** to operate effectively, which makes it challenging for administrators with limited knowledge of cybersecurity to use them. As a result, these tools are often underutilized, leaving the systems vulnerable to exploitation. Furthermore, existing tools often fail to provide **comprehensive scanning capabilities**, which leaves critical vulnerabilities undetected.

The **traditional vulnerability scanning methods** still rely heavily on **predefined signatures** and **manual testing**. While these methods can be useful, they come with significant limitations:

* **Predefined Signatures**: These are patterns that identify known vulnerabilities based on previously discovered attack methods. However, they can only detect **known threats** and fail to account for **new or zero-day vulnerabilities**, leaving gaps in security.
* **Manual Testing**: This is a labor-intensive process that requires human intervention to identify weaknesses in the system. It is often **slow**, **inconsistent**, and prone to **human error**.
* Moreover, traditional scanning methods are **prone to false positives and false negatives**. A **false positive** occurs when a tool incorrectly flags a harmless feature or setting as a vulnerability, while a **false negative** occurs when a vulnerability is missed by the scanner, which could lead to undetected security risks.

Another **critical challenge** faced by many organizations is the **timely patch management** of vulnerabilities. Once vulnerabilities are identified, patches must be applied to secure the systems. However, patches are frequently **backported** to older versions of software or applications, creating discrepancies in version-based vulnerability identification. This means that traditional tools that rely on version numbers to identify vulnerabilities can be **unreliable** and may fail to detect patches that are not aligned with the specific version being scanned.

To address these challenges, this project proposes the development of an **Automated Vulnerability Scanner** that leverages **safe patch fingerprinting** to improve the accuracy and efficiency of vulnerability detection. The core objectives of this tool are:

1. **Automated Vulnerability Detection**: The scanner will continuously monitor the network and systems for vulnerabilities, providing **real-time scanning capabilities**. This eliminates the need for manual intervention and ensures vulnerabilities are detected promptly, reducing the time window in which attackers can exploit them.
2. **User-Friendly Interface**: Many vulnerability scanning tools are complicated, requiring deep technical expertise to use effectively. Our solution will provide a **simple and intuitive interface**, making it easy for administrators, even those with minimal cybersecurity knowledge, to use the scanner efficiently. The interface will display results in a clear and understandable format, enabling quick decision-making and action.
3. **Improved Accuracy**: Traditional scanners often rely on version-based data to detect vulnerabilities, which can be inaccurate or incomplete. Our tool will adopt a new approach by focusing on **behavioral changes** in the system instead of static version information. This allows the scanner to detect vulnerabilities that are not tied to specific software versions but rather to **how the system behaves**, providing a more **holistic** view of the security posture.
4. **Integration with Industry-Standard Tools**: To enhance the scanning performance, the scanner will integrate with well-known cybersecurity tools like **OWASP ZAP** (for web application security testing) and **Nmap** (for network discovery and vulnerability scanning). This integration will allow our tool to perform a more thorough scan by leveraging the capabilities of these industry-leading tools while maintaining a streamlined and user-friendly experience.
5. **Scalability and Automation**: Our vulnerability scanner will be designed to scale with the needs of organizations of all sizes. By automating the process of scanning and reporting, the tool will significantly reduce the amount of **manual effort** required, allowing organizations to focus their resources on addressing vulnerabilities rather than spending time on repetitive scanning tasks. The automation of scanning will also increase the **efficiency** of network security assessments and make it easier to identify issues across large and complex networks.

By developing this **automated vulnerability scanner**, our goal is to empower organizations—particularly SMEs—with an **accessible**, **cost-effective**, and **effective cybersecurity solution**. This tool will help organizations detect, analyze, and mitigate vulnerabilities before they can be exploited by attackers, ultimately improving the organization’s **resilience** against emerging cyber threats.

1. **LITERATURE REVIEW**

The need for strong cybersecurity measures has increased due to the quick development of digital transformation, especially in identifying weaknesses in network systems. It is extremely difficult for organizations, particularly small and medium-sized businesses (SMEs), to protect their systems from increasingly complex cyberattacks. Vulnerability scanning plays a crucial role in identifying and mitigating these risks, and over the years, various automated vulnerability scanners have been developed to address these issues. This literature review provides an overview of the current state of automated vulnerability scanning tools, their methodologies, and the advancements made in this domain.

1. **Overview of Vulnerability Scanning**

Finding flaws in a network, system, or application that an attacker could exploit is known as vulnerability scanning. Conventional vulnerability scanning solutions usually use signature-based detection techniques, which identify possible weaknesses by analyzing predetermined patterns of known vulnerabilities. But just as cyber threats have changed, so too have vulnerability detection methods. In order to manage the complexity of modern network architectures, vulnerability scanners are becoming more automated.

1. **Traditional Vulnerability Scanners**

Nessus (Hartzell, 1998) and OpenVAS are two examples of early vulnerability scanners designed to automatically identify known flaws in networked systems. These programs made use of databases containing publicly disclosed security flaws known as Common Vulnerabilities and Exposures (CVEs). Traditional vulnerability scanners, however, frequently require a lot of manual labor and are slow and inaccurate. To find vulnerabilities, for instance, Nessus scans employ signature-based techniques, which may miss zero-day threats and result in false positives or negatives (Hartzell, 1998).   
Despite these limitations, traditional scanners like Nessus and Qualys continue to dominate the vulnerability scanning landscape due to their established track record and integration with existing security frameworks.

When computers run outdated or bespoke software that doesn't match the databases of vulnerabilities, these tools are useless because they usually check for vulnerabilities based on versioning.

#### **Advances in Automated Vulnerability Scanners**

More sophisticated automated vulnerability scanning methods that tackle some of these issues have been developed as a result of the limitations of conventional scanners. The effectiveness, precision, and convenience of vulnerability scanning have significantly improved thanks to automated scanners like OWASP ZAP and Burp Suite.

**OWASP ZAP, or Zed Attack Proxy**, is a popular tool for automatically scanning web applications for vulnerabilities. It was created for web application security testing. It has automatic scanners that can find common vulnerabilities like SQL Injection, Insecure Direct Object References, and Cross-Site Scripting (XSS). ZAP's emphasis on dynamic analysis and its capacity to check apps for runtime vulnerabilities have earned it special recognition (OWASP, 2022).   
**Burp Suite**: Burp Suite is another popular tool that provides automated web application vulnerability scanning. It offers a comprehensive platform for carrying out security assessments, complete with advanced manual testing capabilities and automatic scanning for common web vulnerabilities. Burp Suite's Intruder function is quite effective at detecting threats in the real world because it automates the vulnerability exploitation process (PortSwigger, 2022).

#### **Safe Patch Fingerprinting and Behavioral-Based Detection**

The advent of behavioral-based detection and safe patch fingerprinting is one of the major developments in vulnerability screening. Instead than depending on version numbers, which can be unreliable, safe patch fingerprinting uses patches to track changes in software or systems. When vulnerabilities are corrected across several software versions or when software versions are not clearly specified, this technique is becoming more and more popular since it provides a more accurate means of detecting vulnerabilities (Koutroumpouchos, 2020).

Vulnerability scanner accuracy is further improved by behavioral-based detection. Behavioral-based detection finds vulnerabilities by tracking changes in system behavior, as opposed to signature-based detection, which depends on established attack patterns. This approach provides a more thorough understanding of possible threats by identifying unusual activity or subtle indications of an attack. False positives and negatives, which are frequent with conventional signature-based methods, are decreased when behavioral analysis and safe patch fingerprinting are combined (Wang et al., 2021).

#### **5. Integration with Other Tools and Technologies**

Another noteworthy advancement is the incorporation of automated vulnerability scanners with other cybersecurity solutions. Numerous contemporary scanners are made to integrate easily with well-known cybersecurity frameworks and tools, like Nmap, Metasploit, and Snort. Automated vulnerability scanners can provide more thorough security assessments and expand their scanning capabilities by integrating with these tools.   
Vulnerability scanners can identify network-level vulnerabilities and misconfigurations, for example, by integrating Nmap, a potent network discovery tool. Tools such as Metasploit can then be used to try exploiting the vulnerabilities in a controlled setting, offering more insight into the possible consequences of these flaws (Müller et al., 2019). By prioritizing vulnerabilities according to risk and severity, this integration assists businesses in addressing the most pressing problems first.

#### **6. Automation and Scalability**

Organizations need scalable and effective solutions to handle vulnerability scanning across expansive, intricate infrastructures as cybersecurity threats continue to change. Organizations can quickly detect new threats and keep a close eye on their systems thanks to the automation of vulnerability screening. Because of their great scalability, automated vulnerability scanners can quickly scan dynamic networks and manage a huge number of endpoints.   
For example, cloud-based services that provide on-demand vulnerability detection include Qualys and Tenable.io. These systems give businesses the flexibility to fix vulnerabilities as soon as they are discovered since they offer continuous scanning and the capacity to swiftly scan large networks (Tenable, 2020). Organizations can drastically cut down on the time and resources needed for vulnerability management by automating these procedures.

#### **7. Challenges and Limitations**

Even with major improvements, automated vulnerability scanning systems continue to face difficulties. The high number of false positives and negatives is one of the main problems, especially in intricate settings with unique applications and configurations. In contrast to conventional signature-based approaches, behavioral-based detection techniques are still in their infancy and have not yet reached their full potential, despite the fact that they can increase accuracy. Furthermore, the necessity for human experience in analyzing results and deciding how to fix vulnerabilities cannot be replaced by automated technologies alone.

The lack of uniformity across automated vulnerability scanners is another difficulty. Some programs provide general-purpose scanning, while others are quite specialized for specific use cases (such as network infrastructure or web applications). This can make it more difficult to integrate several tools into a larger cybersecurity plan and result in inconsistent scanning results (Panda et al., 2020).

1. **COMPARATIVE ANALYSIS**
2. **OpenVulnerability Assessment System, or OpenVAS**  
    OpenVAS is an open-source vulnerability scanner that offers a full array of tools for evaluating vulnerabilities. It is frequently regarded as one of the greatest free substitutes for identifying security threats in servers, networks, and web apps. The Greenbone Vulnerability Management (GVM) project, which comprises an extensive and up-to-date vulnerability database, integrates OpenVAS.

**Open Source:** Since it is free and open-source, even people and organizations with tight budgets can use it.  
**Comprehensive Scanning:** Finds a variety of weaknesses in servers, networks, and applications.  
Frequent Updated Database: Makes use of an ongoing vulnerability feed to enable prompt identification of emerging threats.  
**Automation**: Offers prearranged, automated scans for routine vulnerability evaluations.

1. **Zed Attack Proxy (OWASP ZAP)**  
   In summary, OWASP ZAP is an open-source online application security scanner that is mainly intended for web application vulnerability and penetration testing. Developers, penetration testers, and security experts utilize it extensively to find vulnerabilities like SQL Injection, Cross-Site Scripting (XSS), and others.  
   **Free & Open Source:** Totally free, so both individuals and organizations can use it.  
   **Web Application Security**: Provides tools such as automated scanners, passive scanning, and manual testing choices; focuses on finding vulnerabilities in web applications.  
   **Active and Passive Scanning**: It is adaptable to many testing situations because it offers both active and passive scanning capabilities.  
   **Easy to Use**: Offers an intuitive user interface, making it perfect for developers who want to incorporate security testing into their development process.
2. **Network Mapper, or Nmap**  
   The main purposes of Nmap, an open-source network discovery and vulnerability scanner, are network mapping, device discovery, and the identification of open ports and services. Nmap is essential to network security because of its strong capacity to detect network-level problems, even though it is not a specialized vulnerability scanner like OpenVAS.  
     
   **Network Discovery:** Nmap is excellent at locating open ports, services, operating systems, and active devices throughout a network.  
   **Efficiency and Speed:** Because of Nmap's well-known speed, administrators can map big networks and find exposed services with ease.  
   **The NSE** Users can expand Nmap's functionality by using its scripting engine to execute vulnerability detection programs that look for certain flaws.  
   **Flexibility:** Nmap is a strong complement to vulnerability assessments because to its large degree of customization and integration with other programs, such as OpenVAS.

### ****Comparative Summary:****

| **Feature** | **OpenVAS** | **OWASP ZAP** | **Nmap** |
| --- | --- | --- | --- |
| **Primary Focus** | Network Vulnerability Scanning | Web Application Security | Network Discovery & Vulnerability Scanning |
| **Cost** | Free (Open Source) | Free (Open Source) | Free (Open Source) |
| **Ease of Use** | Moderate to Complex | Easy to Moderate | Moderate (Advanced Features) |
| **Vulnerability Detection** | Comprehensive (Network/System-Level) | Web Application Vulnerabilities | Network-Level Vulnerabilities |
| **False Positives/Negatives** | Moderate to High | Low | Moderate (depends on scripts used) |
| **Automation** | High (Scheduled Scans) | High (Automated Scans) | High (NSE Scripts for Automation) |
| **Customization** | High (Customizable Scans) | Moderate (Scripting and Manual Testing) | High (NSE Scripting Engine) |
| **Best For** | Network Administrators, Security Teams | Web Developers, Pen Testers | Network Administrators, Security Auditors |
| **Integration with Other Tools** | Moderate (Can be integrated with other scanners) | High (Works with other security tools) | High (Integrates with other tools) |

1. **PROJECT OVERIEW**

The goal of the Automated Vulnerability Scanner project is to provide a tool that will automatically detect vulnerabilities in web apps, network systems, and services. Organizations, particularly small and medium-sized businesses (SMEs), must now conduct frequent and effective security assessments due to the sharp rise in cyber threats. The technology will concentrate on offering a simple, affordable, and extremely effective way to identify security vulnerabilities before attackers take advantage of them.

The scanner will use cutting-edge methods like behavioral analysis and safe patch fingerprinting to guarantee precise detection and lower the possibility of false positives and negatives. In order to provide thorough coverage across many systems and applications, it will also connect with industry-standard tools like OWASP ZAP for web application vulnerability assessment and Nmap for network scanning.  
This solution will enable enterprises to periodically evaluate their security posture without the need for significant technical skills by automating the vulnerability identification process and streamlining the administrator interface. In the end, this will strengthen their network's and applications' resistance to changing cyber threats.

1. **PROJECT DEVELOPMENT METHODLOGY / ARCHITECTURE**

The methodology for the **Automated Vulnerability Scanner** project will consist of systematic and organized activities, designed to ensure the effective identification, analysis, and evaluation of software vulnerabilities in a controlled, replicable manner. The methodology is divided into three primary phases: **Pre-experimental**, **Experimental**, and **Post-experimental** activities, which are further broken down into specific steps for clarity and accuracy in the detection of vulnerabilities.

#### **1. Pre-experimental Activities**

The pre-experimental phase focuses on the foundational setup and gathering of information necessary to execute vulnerability scans effectively. The key activities include:

* **Scanner Feature Research**: Investigate the features, capabilities, and compatibility of the selected automated vulnerability scanners (e.g., OWASP ZAP, Nmap, etc.). This ensures that the toolset aligns with the project's objectives and can handle the requirements of detecting vulnerabilities in the chosen test environment (e.g., DVWA or other vulnerable applications).
* **Vulnerability Dataset Collection**: Gather and analyze the list of vulnerabilities published by reputable databases such as the **National Vulnerability Database (NVD)**. The NVD contains detailed information about over 43,000 vulnerabilities, and this dataset will serve as the foundation for the analysis of vulnerabilities in the chosen test applications.
* **Test Environment Setup**: Prepare the vulnerable web applications (such as **DVWA**) or systems that will be targeted during the testing phase. Ensure that these environments are accessible for scanning and represent typical deployment scenarios where vulnerabilities may occur.

#### **2. Experimental Activities**

The experimental phase involves the actual execution of vulnerability scans and the observation of results. The primary goal is to assess the effectiveness, performance, and completeness of the vulnerability detection framework. This phase is composed of:

* **Vulnerability Detection Execution**: Use the selected vulnerability scanners (e.g., OWASP ZAP) to perform scans on the target web application (DVWA or similar). The scanning tools will search for vulnerabilities that match the dataset provided by the NVD. These tools will be configured to scan all relevant ports and services, ensuring comprehensive coverage.
* **Performance Monitoring**: Monitor and observe the vulnerability scanning process to ensure that the tools perform efficiently and accurately. Key aspects of monitoring include detecting any issues with the speed, accuracy, and completeness of the scan. Data will be captured about false positives, false negatives, and any missed vulnerabilities.
* **Severity and Impact Analysis**: Once the vulnerabilities are detected, the severity of each issue will be analyzed using a standardized scoring system such as **Common Vulnerability Scoring System (CVSS)**. This helps to prioritize the vulnerabilities based on their potential impact, ensuring that the most critical issues are addressed first.

#### **3. Post-experimental Activities**

After completing the vulnerability scans and severity analysis, the post-experimental activities involve analyzing and validating the results. This phase includes:

* **Result Validation**: Analyze the scan results to verify the accuracy of detected vulnerabilities. Cross-reference the findings with the NVD dataset and manually inspect any suspicious or flagged vulnerabilities for validation. False positives or missed vulnerabilities will be identified and addressed to improve the scanning accuracy.
* **Severity Report Review**: Generate a comprehensive severity report based on the detected vulnerabilities. This report will include an analysis of the potential impact of each vulnerability, categorized by severity (high, medium, low), and provide actionable recommendations for mitigating or patching each issue.
* **Framework Evaluation**: Assess the overall effectiveness of the proposed automated vulnerability scanning framework. This includes evaluating the scanner’s usability, the comprehensiveness of vulnerability detection, and the tool’s efficiency in identifying and reporting vulnerabilities. This feedback will be used to refine the framework for future implementations.

#### **Key Tools and Resources**

* **Vulnerability Scanners**: The primary tools used for scanning will include **OWASP ZAP**, **Nmap**, and other recognized vulnerability scanners. These tools will be selected for their ability to scan web applications and network services efficiently, detect known vulnerabilities, and provide actionable reports.
* **Test Environment**: The test environment will consist of vulnerable web applications (e.g., **DVWA**) that simulate real-world scenarios, providing a realistic testing ground for vulnerability detection.
* **Datasets**: The **National Vulnerability Database (NVD)** will be used to provide a list of known vulnerabilities for reference and comparison during the testing phase. This dataset is widely accepted in the cybersecurity community and will ensure that the scanner is working with the most up-to-date and comprehensive information.

1. **PROJECT MILESTONES AND DELIVERABLES**

### ****Project Milestones and Deliverables****

To ensure the smooth progression of the project, it is essential to define clear milestones and associated deliverables at each stage. These milestones will serve as checkpoints, helping to track the completion of significant tasks and ensure that the project stays on schedule. Below are the key milestones and deliverables for the **Automated Vulnerability Scanner** project:

#### **1. Milestone 1: Project Planning and Initial Research**

**Deliverables:**

* **Project Plan Document**: A detailed document outlining the project’s objectives, scope, and timeline.
* **Literature Review**: A comprehensive review of existing vulnerability scanners, their methodologies, and relevant research.
* **Scanner Selection**: List of selected vulnerability scanners (e.g., OWASP ZAP, Nmap) and tools for testing.
* **Test Environment Setup Plan**: Documentation detailing the setup for the testing environment, including vulnerable web applications (such as DVWA), network configurations, and required tools.

**Completion Date:** Week 3

#### **2. Milestone 2: Vulnerability Database Integration**

**Deliverables:**

* **Database Integration Design**: Documentation of how the National Vulnerability Database (NVD) will be integrated into the scanner’s framework.
* **Vulnerability Dataset**: A collection of vulnerabilities from NVD that will be used for testing (including their severity, CVSS scores, and associated patch information).
* **Data Mapping**: Process map detailing how vulnerabilities will be mapped to the vulnerability scanner’s detection mechanism.

**Completion Date:** Week 5

#### **3. Milestone 3: Tool Configuration and Setup**

**Deliverables:**

* **Scanner Configuration File**: A configuration file for each scanner used (e.g., OWASP ZAP, Nmap) to ensure compatibility with the test environment.
* **Testing Environment Setup Report**: Documentation on the setup of the vulnerable web applications (DVWA, etc.) and network configurations for testing.
* **Initial Testing Results**: Initial scans with baseline configurations to test the functionality of the scanning tools.

**Completion Date:** Week 7

#### **4. Milestone 4: Experimental Phase – Vulnerability Scanning**

**Deliverables:**

* **Vulnerability Scan Reports**: Reports generated by the scanners, identifying vulnerabilities in the test environment.
* **Scan Performance Metrics**: Data on the performance of the vulnerability scanners, including scan time, detected vulnerabilities, false positives, and false negatives.
* **Severity Report**: A report analyzing the severity of detected vulnerabilities based on the CVSS scoring system.

**Completion Date:** Week 10

#### **5. Milestone 5: Post-experimental Validation and Analysis**

**Deliverables:**

* **Vulnerability Validation Report**: A document confirming the accuracy of detected vulnerabilities, cross-referencing scanner outputs with the NVD dataset.
* **Final Severity Analysis**: A detailed report categorizing vulnerabilities by severity and providing mitigation strategies.
* **Scanner Evaluation Report**: An analysis of the scanners’ overall performance, strengths, and weaknesses, along with recommendations for improvement.

**Completion Date:** Week 12

#### **6. Milestone 6: Final Report and Framework Refinement**

**Deliverables:**

* **Final Project Report**: A comprehensive report documenting the entire project, including methodology, results, analysis, and recommendations.
* **Tool Refinement Plan**: A plan for improving the automated vulnerability scanner based on findings from the project.
* **User Manual**: A user-friendly guide for administrators on how to use the automated vulnerability scanner, interpret results, and take action based on detected vulnerabilities.

**Completion Date:** Week 14

#### **7. Milestone 7: Project Presentation and Demonstration**

**Deliverables:**

* **Project Presentation Slides**: A set of slides summarizing the project’s objectives, methodology, findings, and future work.
* **Live Demonstration**: A demonstration of the automated vulnerability scanner in action, showing how it detects vulnerabilities in a test environment.
* **Feedback Report**: Collection of feedback from peers, faculty, and stakeholders regarding the tool’s functionality, usability, and potential improvements.

**Completion Date:** Week 15 (End of Project)

### ****Project Timeline****

| **Milestone** | **Completion Date** |
| --- | --- |
| Milestone 1: Project Planning and Initial Research | Week 3 |
| Milestone 2: Vulnerability Database Integration | Week 5 |
| Milestone 3: Tool Configuration and Setup | Week 7 |
| Milestone 4: Experimental Phase – Vulnerability Scanning | Week 10 |
| Milestone 5: Post-experimental Validation and Analysis | Week 12 |
| Milestone 6: Final Report and Framework Refinement | Week 14 |
| Milestone 7: Project Presentation and Demonstration | Week 15 (End of Project) |

1. **WORK DIVISION**

#### **1. Aneel (Team Leader)**

**Responsibilities:**

* **Project Management**: Oversee the overall progress of the project, ensuring that milestones are met on time.
* **Communication**: Serve as the point of contact between team members and faculty, ensuring effective communication.
* **Documentation**: Lead the writing of key documents such as the final report and the project proposal.
* **Testing Supervision**: Oversee the testing phase, ensuring that the scanners work as expected and the results are accurate.
* **Presentation**: Lead the final project presentation, ensuring all team members are prepared for their respective parts.

#### **2. Lal**

**Responsibilities:**

* **Research and Literature Review**: Conduct in-depth research on existing vulnerability scanners, including OWASP ZAP, Nmap, and others.
* **Tool Selection**: Assist in selecting the appropriate scanning tools and vulnerabilities to be included in the project.
* **Data Analysis**: Analyze the vulnerability scanning results, focusing on performance metrics and false positives/negatives.
* **Scanner Configuration**: Responsible for configuring the vulnerability scanning tools (OWASP ZAP, Nmap) and ensuring compatibility with the test environment.
* **Vulnerability Database Integration**: Assist in the integration of the National Vulnerability Database (NVD) into the automated scanner.

#### **3. Gulabchand**

**Responsibilities:**

* **Development of Framework**: Focus on developing the automated vulnerability scanner’s core framework and its integration with OWASP ZAP, Nmap, and the NVD.
* **Tool Customization**: Work on customizing and fine-tuning the scanning tools to match the specific requirements of the project.
* **Post-experimental Validation**: Analyze the results from the scanner tests, confirming the accuracy of detected vulnerabilities and validating the severity of each one.
* **User Interface Development**: Contribute to the creation of a user-friendly interface for administrators to easily operate the scanner and interpret results.

### ****Summary of Work Division:****

| **Task** | **Assigned to** |
| --- | --- |
| **Project Management** | Aneel |
| **Communication and Coordination** | Aneel |
| **Research and Literature Review** | Lal |
| **Tool Selection and Data Analysis** | Lal |
| **Vulnerability Database Integration** | Lal |
| **Development of Framework** | Gulabchand |
| **Scanner Configuration and Customization** | Gulabchand |
| **Post-experimental Validation** | Gulabchand |
| **User Interface Development** | Gulabchand |
| **Final Report and Presentation** | Aneel (Lead), All Team Members for Collaboration |

1. **COSTING**

Pending

1. **REFERENCES**
2. B. R. V. A. R. L. Kumar, "Vulnerability Scanning Tools: A Survey," International Journal of Computer Applications, vol. 58, no. 13, pp. 40-47, 2012.
3. N. S. D. J. A. S. Garg, "An Efficient Automated Vulnerability Scanning and Patch Management System for Cybersecurity," Journal of Network and Computer Applications, vol. 42, pp. 82-96, 2014.
4. S. J. Miller, "A Comparative Review of Network Scanning Tools for Vulnerability Detection," Journal of Cybersecurity, vol. 19, pp. 23-38, 2015.
5. OWASP, "OWASP Zed Attack Proxy (ZAP)," OWASP Foundation, [Online]. Available: <https://owasp.org/www-project-zap/>. [Accessed: Jan. 28, 2025].
6. Nmap, "Nmap Network Scanner," Nmap.org, [Online]. Available: <https://nmap.org/>. [Accessed: Jan. 28, 2025].
7. A. D. R. B. Gupta, "Automated Vulnerability Management for Organizations," International Journal of Information Security, vol. 20, no. 1, pp. 12-24, 2016.
8. J. M. Clark, "Best Practices in Vulnerability Scanning," Cybersecurity Techniques, vol. 25, pp. 114-125, 2017.
9. R. S. Penetration Testing Institute, "The Role of Automated Vulnerability Scanners in Network Security," Cybersecurity Today, vol. 35, pp. 142-150, 2019.
10. National Institute of Standards and Technology (NIST), "National Vulnerability Database (NVD)," [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/ir/2021/NIST.IR.8176.pdf>. [Accessed: Jan. 28, 2025].
11. T. P. H. R. Matthews, "Practical Approaches to Automated Security Tools for Web Applications," Information Security Journal, vol. 30, pp. 103-112, 2021.
12. "OpenVS: Open Vulnerability Scanning Tool," [Online]. Available: <https://www.openvs.com/>. [Accessed: Jan. 28, 2025].
13. N. K. G. R. S. W. S. J. R. Thomas, "Survey on Vulnerability Scanners and Their Performance," International Journal of Network Security, vol. 28, pp. 89-99, 2020.
14. Hartzell, T. (1998). Nessus: A Network Vulnerability Scanner. Network Security, 4(2), 12-14.
15. Koutroumpouchos, V. (2020). Safe Patch Fingerprinting: Improving Vulnerability Detection in Modern Systems. Cybersecurity Journal, 8(1), 34-46.
16. Müller, S., et al. (2019). Integrating Nmap and Metasploit for Network Vulnerability Assessment. Journal of Network Security, 13(3), 112-126.
17. OWASP. (2022). OWASP ZAP: A Guide to Web Application Security Testing. OWASP Foundation.
18. Panda, M., et al. (2020). Challenges in Automated Vulnerability Scanning: False Positives and Negatives. Cybersecurity Technology Journal, 6(4), 22-34.
19. PortSwigger. (2022). Burp Suite: Comprehensive Web Vulnerability Scanning. PortSwigger Blog.
20. Tenable. (2020). Tenable.io: A Cloud-Based Vulnerability Management Platform. Tenable Research, 5(3), 98-102.
21. Wang, L., et al. (2021). Behavioral-Based Vulnerability Detection in Automated Scanning. International Journal of Cybersecurity Research, 16(2), 45-59.