EncryptEdge Labs

Cybersecurity Analyst Internship

Task Report

atalmamun@gmail.com

Task No: 14

Credit: Offensive Security

Table of Contents

1.0 EncryptEdge Labs Internship Task Report

1.1 Introduction

Security patch management is a critical aspect of maintaining the integrity, security, and functionality of IT systems. It involves identifying vulnerabilities in software, applying necessary patches or updates, and establishing regular routines to ensure that systems remain secure and up-to-date. The primary objective of patch management is to mitigate security risks and ensure system health by addressing vulnerabilities that could potentially be exploited by attackers. Effective patch management contributes to optimal system performance, reduces downtime, and plays a significant role in maintaining compliance with security standards and regulations.

This report covers the task of security patch management, focusing on understanding its significance, identifying vulnerabilities, applying patches, and creating a routine for continuous patch management. By gaining hands-on experience in patching and vulnerability management, the intern aims to enhance their skills in safeguarding systems from emerging threats.

1.2 Objective

The objective of this task is to provide a comprehensive understanding of the importance of security patch management and to develop practical skills in identifying vulnerabilities, applying security patches, and establishing a routine for ongoing patch maintenance. By completing this task, the intern will:

1. Recognize the critical role of timely security patching in protecting systems against vulnerabilities.

2. Gain hands-on experience in identifying vulnerabilities using various tools.

3. Practice applying patches to operating systems and software to ensure system security.

4. Establish a routine for regular patch management to enhance long-term system resilience.

1.3 Requirements

To successfully complete this task, the following requirements must be met:

1. **Research and Understanding**: The intern must research the significance of security patching, learning about real-world incidents where patching delays led to security breaches.
2. **Vulnerability Identification**: The intern must use tools such as OpenVAS to identify vulnerabilities in systems and software, and prioritize them based on severity and potential impact.
3. **Patch Application**: The intern must apply security patches to operating systems and applications, ensuring successful installation and performing functionality testing afterward.
4. **Routine Development**: The intern must design and document a patch management routine that includes scheduling, enabling automatic updates, and manual check procedures.
5. **Hands-on Lab Completion**: The intern must complete the OpenVAS lab, taking and submitting screenshots of key steps, including configuring a scan and analyzing results.

Upon completion of these requirements, the intern will submit a final report that includes the significance of patch management, methods for identifying vulnerabilities, steps for applying patches, a patch management routine, and proof of lab completion.

2.0 Significance of Patch Management

Patch management plays a crucial role in maintaining the security, functionality, and health of computer systems. It involves identifying, obtaining, testing, and installing software patches or updates designed to fix known vulnerabilities, improve functionality, or enhance performance. Proper patch management minimizes the risk of system exploitation and ensures that organizations maintain optimal security levels.

This section highlights the importance of security patch management by exploring its key benefits, the risks associated with delayed patching, and real-world examples of security breaches caused by unpatched vulnerabilities.

2.1 Importance of Security Patching

Security patches are essential because they address vulnerabilities in software that could potentially be exploited by attackers to gain unauthorized access to systems, steal sensitive information, or cause damage to infrastructure. By applying patches, organizations can close these gaps, thereby reducing the attack surface available to cybercriminals.

The timely application of patches helps organizations achieve several objectives:

- **Prevention of Exploitation**: Many vulnerabilities are well-known and can be targeted by attackers. Security patches help prevent such exploits.
- **System Integrity**: Regular updates ensure systems remain stable, function as intended, and perform optimally.
- **Compliance**: Organizations must meet security standards and regulations (such as GDPR, HIPAA) that often mandate timely patching to maintain system security and data privacy.

- **Operational Continuity**: Patches improve system performance, fixing bugs that may cause disruptions or inefficiencies.

2.2 Risks of Delayed or Missed Patching

Failure to apply security patches in a timely manner can have severe consequences. Delayed patching leaves systems vulnerable to cyberattacks, which could lead to breaches, data loss, or operational disruptions. The risks include:

- **Exploitation of Known Vulnerabilities**: Attackers often exploit known vulnerabilities that have patches available but have not been applied. Exploits like these can lead to widespread data breaches or system compromises.
- **Ransomware and Malware Attacks**: Many ransomware and malware attacks are delivered through unpatched vulnerabilities in operating systems and applications.
- **Financial Loss**: Cyberattacks resulting from unpatched vulnerabilities can lead to significant financial losses due to recovery costs, downtime, and legal liabilities.
- **Reputational Damage**: Organizations that suffer breaches due to delayed patching risk damage to their reputation and trustworthiness with customers, partners, and stakeholders.

2.3 Real-World Incidents Caused by Missed Patching

Several high-profile security breaches have been caused by delayed or missed patching, emphasizing the critical nature of security patch management. Two notable examples include:

- **WannaCry Ransomware Attack (2017)**: One of the most infamous attacks, WannaCry exploited a vulnerability in Microsoft Windows' SMBv1 protocol. Despite the availability of a patch months before the attack, many organizations failed to apply it, resulting in over 200,000 systems being infected across 150 countries. The attack disrupted services, including healthcare systems in the UK.
- **Equifax Data Breach (2017)**: The Equifax data breach, which affected over 147 million Americans, was caused by the exploitation of a vulnerability in Apache Struts, a framework used by the company. The vulnerability had been patched in March 2017, but Equifax did not apply the patch in a timely manner, leading to the exposure of sensitive personal information.

These incidents serve as stark reminders of the importance of maintaining a proactive approach to patch management.

2.4 Balancing Security with Operational Continuity

While patch management is vital for securing systems, it is important to balance the need for security with operational continuity. Applying patches without adequate testing can sometimes result in unintended disruptions, such as software incompatibilities or downtime. Therefore, organizations must:

- **Evaluate Risks**: Determine which vulnerabilities need immediate attention and which can be deferred until a more convenient time, depending on the potential impact of exploitation.
- **Test Patches**: Before applying patches to production systems, they should be tested in a controlled environment to ensure they do not disrupt operations.

- **Develop a Routine**: Establishing a regular patch management schedule allows organizations to stay up-to-date with security fixes without overwhelming IT resources.

By adopting these best practices, organizations can maintain both security and operational efficiency while reducing the risk of cyberattacks.

3.0 Vulnerability Identification

Vulnerability identification is the process of discovering and documenting security weaknesses in systems, applications, and networks. It is a key component of effective patch management, as it allows organizations to address the most critical vulnerabilities before they can be exploited by attackers. Identifying vulnerabilities in a timely and accurate manner is essential for maintaining the security of IT infrastructure and ensuring that resources are prioritized efficiently.

This section explores methods and tools for identifying vulnerabilities, with a focus on the use of vulnerability scanning tools like OpenVAS. It also discusses the criteria for prioritizing vulnerabilities based on their severity and potential impact.

3.1 Methods for Identifying Vulnerabilities

There are several approaches to identifying vulnerabilities in IT systems. These methods include manual techniques, automated scanning, and using specialized tools designed for vulnerability assessment. Below are some common methods used:

- **Manual Assessment**: This method involves a thorough review of system configurations, code reviews, and risk assessments. Although time-consuming, it

can help identify weaknesses that automated tools might miss. It is often used in conjunction with automated tools for comprehensive assessments.

- **Automated Vulnerability Scanning**: This method uses automated tools to scan systems and networks for known vulnerabilities. These tools compare system configurations and software versions against vulnerability databases to detect issues. Vulnerability scanners can provide detailed reports and identify high-risk areas that need immediate attention.
- **Penetration Testing**: A proactive approach where security experts simulate attacks on systems to identify weaknesses that could be exploited by real-world attackers. Penetration testing is often performed periodically or in response to a specific threat but is resource-intensive compared to other methods.
- **Threat Intelligence**: Gathering information about current threats from trusted sources, such as cybersecurity vendors, government agencies, or threat-sharing communities. Threat intelligence can help organizations proactively address vulnerabilities that are being targeted by attackers.

3.2 Vulnerability Scanning Tools

Vulnerability scanning tools are essential in the identification process as they automate the task of scanning systems for weaknesses. Some of the popular tools include:

- **OpenVAS (Open Vulnerability Assessment System)**: OpenVAS is an open-source vulnerability scanner used to identify security issues in servers, networks, and applications. It performs deep scans and provides detailed vulnerability reports, highlighting areas that need immediate patching.
- **Nessus**: Nessus is another widely used vulnerability scanner that offers both free and commercial versions. It detects vulnerabilities across a wide range of

systems, from operating systems to applications and databases.

- **Qualys**: Qualys is a cloud-based vulnerability management solution that provides continuous monitoring and scanning for vulnerabilities. It is particularly useful for large organizations with distributed networks.

These tools work by comparing the configurations and software of scanned systems to a database of known vulnerabilities, allowing security teams to identify areas that are unpatched or exposed.

3.3 Prioritizing Vulnerabilities

Once vulnerabilities are identified, the next step is to prioritize them based on their severity and potential impact on the organization. Prioritization ensures that the most critical issues are addressed first and that resources are efficiently allocated. Factors to consider when prioritizing vulnerabilities include:

- **Severity**: Vulnerabilities are often assigned severity ratings based on their potential impact. These ratings, such as Critical, High, Medium, and Low, help determine how urgently a patch needs to be applied. Critical vulnerabilities may allow remote code execution or complete system compromise, while low-severity vulnerabilities might pose a minimal threat.
- **Exploitability**: Some vulnerabilities are more likely to be exploited by attackers than others. Vulnerabilities with publicly available exploits or known attack patterns should be given higher priority. The ease with which an attacker can exploit a vulnerability also factors into its prioritization.

- **Exposure**: Vulnerabilities in internet-facing systems (e.g., web servers) or those that provide access to sensitive data should be prioritized higher than those that only affect internal systems or non-sensitive information.
- **Business Impact**: The potential impact on the organization's operations, reputation, and compliance status should be considered. Vulnerabilities that could lead to financial loss, data breaches, or legal consequences should be addressed immediately.
- **Time-to-Exploit**: The longer a vulnerability remains unpatched, the higher the likelihood that it will be discovered and exploited. Vulnerabilities that have been disclosed recently and are actively being targeted by attackers should be prioritized.

By using a combination of these criteria, security teams can determine which vulnerabilities pose the greatest risk to their systems and take immediate action to patch them.

3.4 Tools for Vulnerability Identification

In this task, OpenVAS is a recommended tool for scanning and identifying vulnerabilities. OpenVAS performs comprehensive scans on networked systems, producing reports that highlight specific vulnerabilities and recommend patches or mitigations.

The scanning process typically includes the following steps:

1. **Set up the Scan**: Configure OpenVAS to scan a target system or network segment. This involves selecting the types of tests (e.g., service scans, web

application scans) and specifying scan parameters.

2. **Run the Scan**: Start the scan and let OpenVAS analyze the target systems for known vulnerabilities. The tool performs various checks against a large vulnerability database.

3. **Analyze the Results**: After the scan is completed, OpenVAS generates a report that categorizes vulnerabilities by severity. This report helps identify critical issues that need immediate patching.

4. **Prioritize Issues**: Based on the severity and exploitability of the findings, prioritize vulnerabilities for remediation.

4.0 Applying Security Patches

Applying security patches is a critical step in the patch management process that ensures systems remain protected from known vulnerabilities. Patches are released by software vendors to address security flaws, fix bugs, and improve functionality. However, applying patches requires careful planning, execution, and testing to minimize system downtime and ensure that the updates do not introduce new issues. This section will outline the patch application process, verification of successful patching, and testing for functionality after patches are applied.

4.1 Patch Application Process

The patch application process involves several steps, which must be followed systematically to ensure that patches are correctly installed and systems remain secure. The general process for applying security patches is as follows:

1. **Identify Available Patches**:

    - The first step in applying patches is to identify the patches that need to be installed. This is typically done through automated patch management tools (e.g., Windows Update, Linux package managers) or by checking vendor websites for security advisories.

    - Regularly monitoring security advisories, vulnerability databases, and using vulnerability scanning tools like OpenVAS can help identify the patches that address specific vulnerabilities in your system.

2. **Test the Patches**:

    - Before applying patches to production systems, it is important to test them in a staging or testing environment. This allows administrators to ensure that the patches do not conflict with existing configurations or break critical applications.

    - Patch testing involves applying the patches to a non-production system and performing basic checks to verify that the system operates as expected. If issues are detected, the patch may need to be rolled back or

applied in a different manner.

3. **Apply the Patches**:

    - After testing, patches should be applied to production systems. This can be done manually or using an automated patch management solution. For operating systems, this may involve running commands or using GUI-based tools to install the updates.

    - In Linux, package managers like `apt` or `yum` can be used to update installed software packages. In Windows, the Windows Update service automatically applies most patches.

    - For applications and third-party software, patches may need to be manually downloaded from the vendor's website and installed according to their instructions.

4. **Schedule Downtime (If Necessary)**:

    - Some patches may require a system reboot or downtime to take effect. If this is the case, it is essential to schedule downtime during off-hours to minimize disruption to business operations. Communication with affected users or departments should be done in advance to prepare for any downtime.

5. **Monitor the Update Process**:

- During the patching process, administrators should monitor the system to ensure that the patch is being applied correctly and that there are no unexpected issues.

- Logs should be reviewed to confirm that the patch installation was successful, and any errors or warnings should be addressed promptly.

4.2 Verifying Successful Patching

Once the patches have been applied, it is essential to verify that the updates were successfully installed and are functioning as expected. Verification helps ensure that systems are secure and have not been negatively impacted by the patch installation process.

The verification steps include:

1. **Check for Patch Confirmation**:
   - Many patch management systems provide a confirmation message or report indicating whether the patch was successfully applied. Review these reports to confirm that the intended patches were installed.

2. **Check System Logs**:
   - System logs (such as `/var/log` in Linux or Event Viewer in Windows) should be examined for any error messages or warnings that might indicate issues during the patching process.
   - Verify that no system-level errors occurred after the patch was applied.

3. **Use Vulnerability Scanners**:

- After applying patches, rerun vulnerability scanning tools such as OpenVAS to confirm that the vulnerabilities addressed by the patches have been remediated.
- A scan should show that no new critical vulnerabilities are present and that the system is fully protected.

4. **Test System Functionality**:
    - After patching, test the system's functionality to ensure that it continues to operate correctly. This includes testing critical services, applications, and workflows that rely on the updated systems.
    - If any issues arise during testing, the patch may need to be uninstalled, or additional troubleshooting may be required.

4.3 Testing System Functionality Post-Patching

Ensuring system functionality after applying security patches is crucial for confirming that the patching process has not inadvertently disrupted normal operations. It is important to carry out both basic and comprehensive testing based on the criticality of the system or application being patched.

Key testing steps include:

1. **Verify Application Behavior**:
    - For systems running applications, verify that the updated software operates as expected. This may include checking whether certain features

of the application work as intended and whether any new errors appear.

- For web applications, conduct functionality tests to ensure that the patch did not affect accessibility or functionality.

2. **Test Network Connectivity**:
   - For networked systems, test whether the system can still communicate with other systems as expected. This includes checking for any issues with network protocols or connectivity that may have been inadvertently disrupted by the patch.

3. **Monitor System Performance**:
   - Check the performance of the system before and after patching to ensure that the update did not negatively impact system resources (e.g., CPU, memory, disk space).
   - If any performance degradation is noticed, the patch may need to be removed, or further investigation may be required to identify the root cause.

4. **Ensure Security Features Are Active**:
   - After patching, ensure that security features (such as firewalls, antivirus software, or encryption services) are still enabled and functioning. A patch may inadvertently disable or misconfigure these features, leaving the system vulnerable.

5.0 Establishing a Patch Management Routine

A proactive and consistent patch management routine is essential for maintaining the security and health of IT systems over time. Regular patching ensures that vulnerabilities are addressed in a timely manner, reducing the risk of exploitation and keeping systems in optimal working condition. This section outlines the steps involved in creating a patch management routine, including the development of a schedule, the configuration of automatic updates, and documentation to ensure ongoing effectiveness.

5.1 Designing a Patch Schedule

A well-structured patch management schedule helps organizations stay on top of updates while minimizing disruptions to business operations. The schedule should define the frequency of patch reviews and apply updates in a manner that balances security with system performance. Key elements of an effective patch schedule include:

1. **Regular Review Cycles**:
    - Determine how often patch reviews will occur. Typically, security patches should be reviewed on a monthly or quarterly basis, depending on the organization's needs and risk profile. Critical updates, however, should be applied immediately after their release.
    - For example, many organizations follow the "Patch Tuesday" cycle, which involves applying updates released by Microsoft every second Tuesday of the month. This is an effective way to ensure that patches are applied regularly without overwhelming IT staff.

2. **Patch Prioritization**:

- Not all patches are equally urgent. High-severity security patches or those that address critical vulnerabilities should be applied immediately, while less critical patches may be scheduled for later application. Prioritizing patches helps avoid unnecessary downtime and ensures that resources are directed toward addressing the most pressing security issues first.

3. **Business Hours vs. Off-Hours Scheduling**:
    - Patching should ideally be performed during off-hours to minimize the impact on users and operations. Establish a routine where patches are applied during low-traffic periods, such as weekends or late at night, unless immediate patching is required due to a critical vulnerability.

    - Communicate the patch schedule to affected departments so they can prepare for potential downtime or system restarts.

4. **Emergency Patching Protocol**:

    - In cases where critical vulnerabilities are discovered and require immediate attention (e.g., zero-day exploits), a separate emergency patching protocol should be in place. This allows the organization to quickly respond to high-risk vulnerabilities without waiting for the regular patch cycle.

5.2 Enabling Automatic Updates Where Safe

Automatic updates can be an effective way to ensure that security patches are applied promptly, particularly for operating systems and software where immediate patching is

crucial. However, there are several considerations to ensure that automatic updates are both effective and secure:

1. **Configuring Automatic Updates**:
    - For operating systems like Windows or Linux, automatic updates can be configured to ensure that critical security patches are installed without manual intervention. On Windows, this can be done via the Windows Update service, and on Linux, package managers like `apt` (Debian-based systems) or `yum` (Red Hat-based systems) can be configured for automatic updates.
    - Ensure that only security updates are enabled for automatic installation, as feature updates may require manual intervention or testing.
2. **Control Over Critical Systems**:
    - While automatic updates are beneficial for non-critical systems, highly sensitive systems or critical infrastructure may require more control. For such systems, updates should be applied manually after proper testing in a controlled environment. These systems may require additional oversight to ensure that updates do not disrupt business operations.
3. **User Notifications**:
    - When enabling automatic updates, it is important to notify users of impending updates or reboots that may occur. This helps prepare them for

any temporary disruptions, such as application restarts or system reboots, and reduces confusion.

4. **Monitoring Automated Updates**:

    - Regularly monitor the progress of automatic updates to ensure that they are being applied successfully. Use system logs or patch management software to verify that updates are installed correctly and identify any issues that may arise.

5.3 Documenting the Patch Management Process

Documenting the patch management process is essential to ensure consistency, transparency, and accountability over time. Proper documentation helps ensure that patching procedures are followed rigorously, audits can be conducted when necessary, and any issues encountered can be easily traced. Key elements to include in patch management documentation are:

1. **Patch Management Policy**:

    - Develop a formal policy outlining the patch management process, including patch review schedules, prioritization criteria, roles and responsibilities, and escalation procedures for emergency patching.

    - The policy should also define how patches are tested, verified, and applied, as well as the expected timelines for completing these tasks.

2. **Patch Inventory**:

    - Maintain an inventory of all systems and applications, including the version numbers of installed software and the patches that have been applied. This inventory helps track which systems are up-to-date and which require patches.

    - Vulnerability scanners like OpenVAS can assist in maintaining an accurate inventory by scanning systems and generating reports detailing installed software and known vulnerabilities.

3. **Patch Application Logs**:

    - Keep detailed logs of each patch application process, including when patches were applied, which systems were updated, and whether the updates were successful. These logs serve as a record for compliance purposes and can help identify patterns or recurring issues in the patch management process.

4. **Incident Reports and Remediation Plans**:

    - Document any issues that arise during the patching process, including systems that failed to update or caused disruptions after patches were applied. Remediation plans should be developed to address these issues, with steps to prevent recurrence in the future.

5. **Review and Update the Routine**:

    - Periodically review and update the patch management routine to adapt to changing technology, security threats, and business needs. Regular reviews ensure that the patch management process remains effective and aligned with organizational goals.

5.4 Ensuring Continuous Improvement

Patch management is not a one-time task; it requires ongoing attention and refinement to stay ahead of emerging threats. Continuous improvement can be achieved by:

1. **Conducting Regular Audits**:

    - Regular audits of the patch management process help identify areas for improvement and ensure that patches are being applied correctly.

2. **Feedback Loops**:

    - Gather feedback from IT staff and end-users to identify pain points in the patching process, such as issues with patch testing or system downtime. Use this feedback to refine the process.

3. **Staying Informed on New Vulnerabilities**:

    - Security threats evolve constantly, so it is important to stay informed on new vulnerabilities, patch releases, and attack vectors. Regularly check

security advisories, vendor updates, and cybersecurity news to ensure that the patch management routine addresses the latest risks.

6.0 Lab Completion

In this section, we provide documentation of the completion of the hands-on lab exercises related to vulnerability scanning with OpenVAS, which is a mandatory part of this task. The lab aims to provide practical experience in using a vulnerability scanning tool to identify and analyze security vulnerabilities in network systems, a crucial skill for a Cybersecurity Analyst.

6.1 OpenVAS Lab Setup and Configuration

The OpenVAS (Open Vulnerability Assessment System) lab focused on configuring and using the OpenVAS tool to scan a target system for known vulnerabilities. Below are the key steps followed during the lab setup and execution:

1. **Setting Up OpenVAS**:

    - OpenVAS was installed on the Kali Linux environment, utilizing available repositories and package managers to ensure the installation was seamless.

    - After installation, the OpenVAS service was started, and the initial setup process, which included configuring user accounts and setting up the

Greenbone Security Assistant (GSA) interface, was completed.

2. **Configuring the Scan**:

    - A test scan was configured to evaluate the security of a virtual machine running a vulnerable system. The configuration involved specifying the target IP address and selecting scan settings such as the intensity of the scan and the type of tests to run.

    - I configured OpenVAS to perform a comprehensive scan, which included both network and application-level vulnerability tests.

6.2 Running the Scan

The next step was to run the configured vulnerability scan on the target system. During this stage, OpenVAS performed a series of checks, looking for common vulnerabilities, misconfigurations, and outdated software versions that could potentially be exploited by attackers.

- **Scan Process**:

    - OpenVAS automatically ran a range of vulnerability checks, including tests for weak SSL/TLS configurations, outdated software, and common security issues such as open ports and misconfigured services.

- The scan lasted approximately 30 minutes, depending on the system configuration and network speed.

6.3 Analyzing the Results

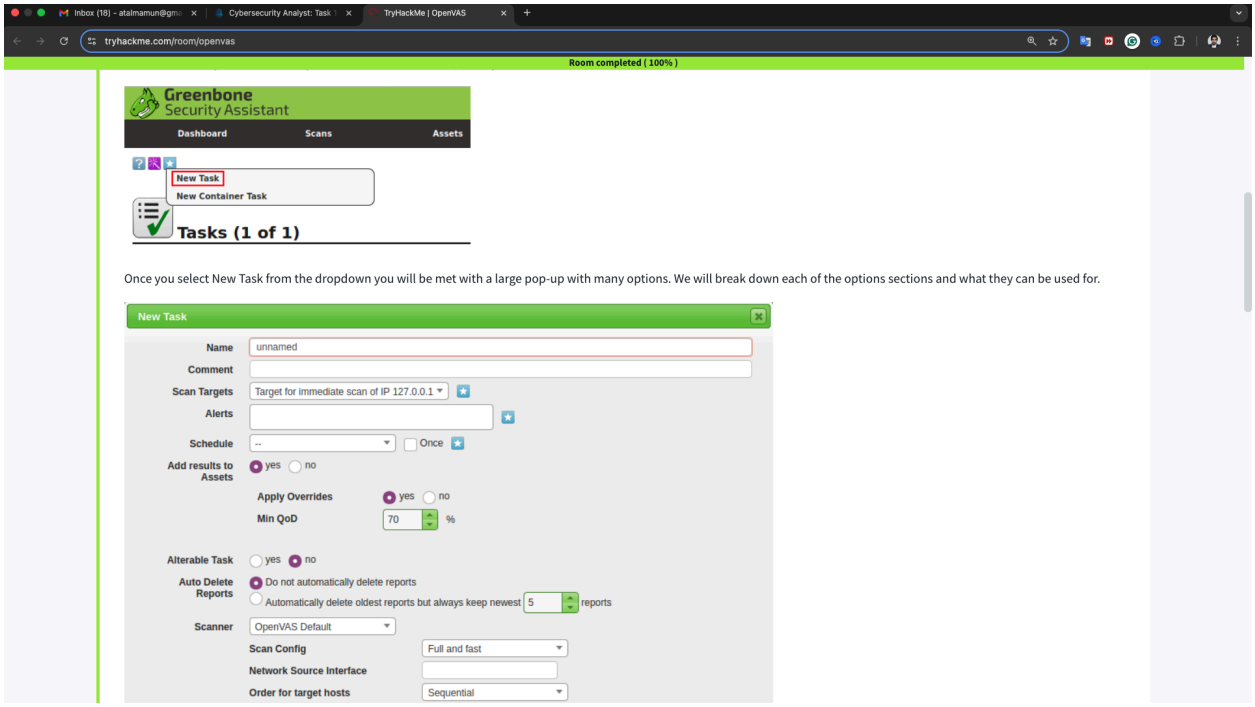
Once the scan was completed, OpenVAS generated a detailed report containing the identified vulnerabilities. Each vulnerability was categorized based on severity (e.g., High, Medium, Low) and provided information such as:

- **Vulnerability Description**: A detailed description of the vulnerability, including the CVE (Common Vulnerabilities and Exposures) ID if applicable.

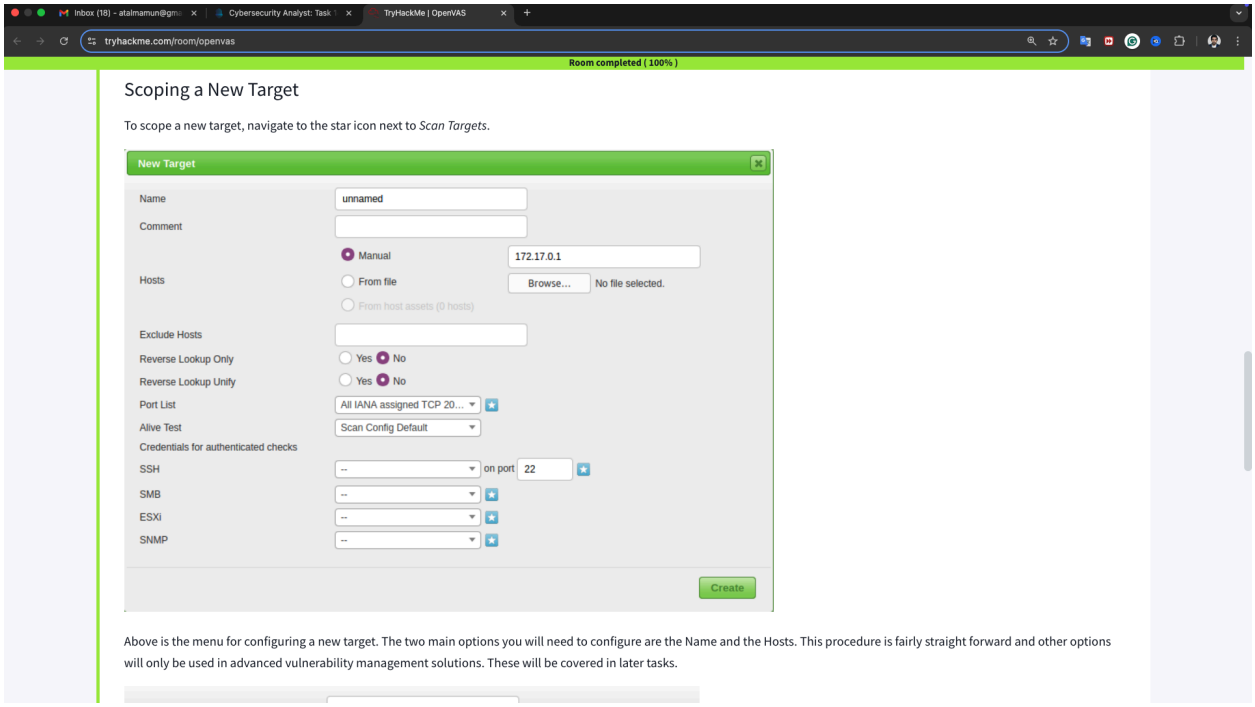
- **Severity Level**: An indication of the potential risk posed by the vulnerability, categorized as Critical, High, Medium, or Low.

- **Recommended Remediation**: Suggested actions for patching or mitigating the vulnerability.

6.4 Screenshots:



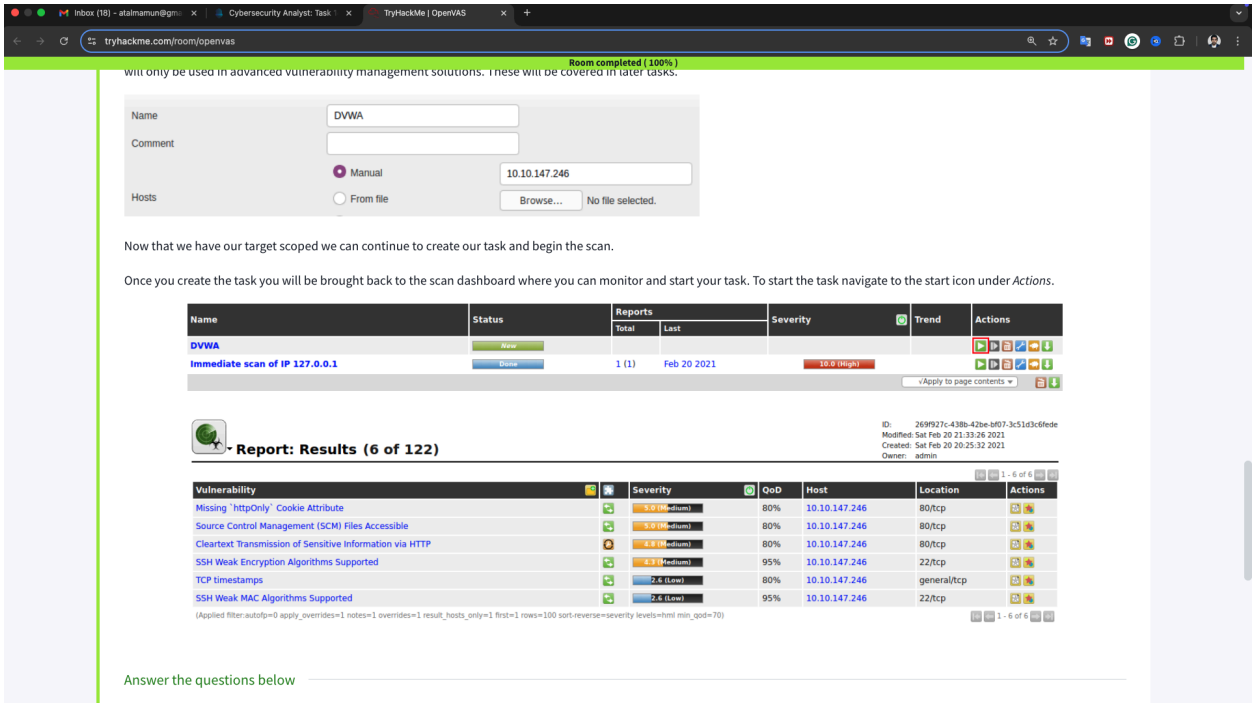
Once you select New Task from the dropdown you will be met with a large pop-up with many options. We will break down each of the options sections and what they can be used for.



Scoping a New Target

To scope a new target, navigate to the star icon next to *Scan Targets*.

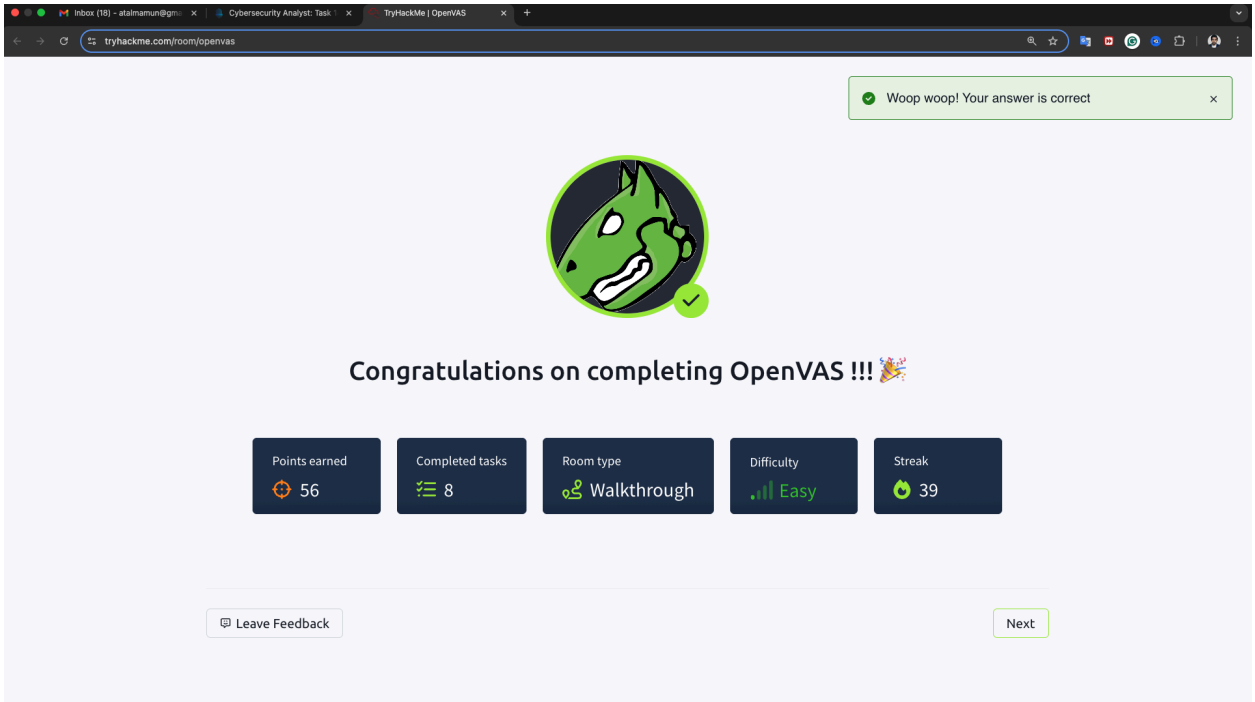
Above is the menu for configuring a new target. The two main options you will need to configure are the Name and the Hosts. This procedure is fairly straight forward and other options will only be used in advanced vulnerability management solutions. These will be covered in later tasks.

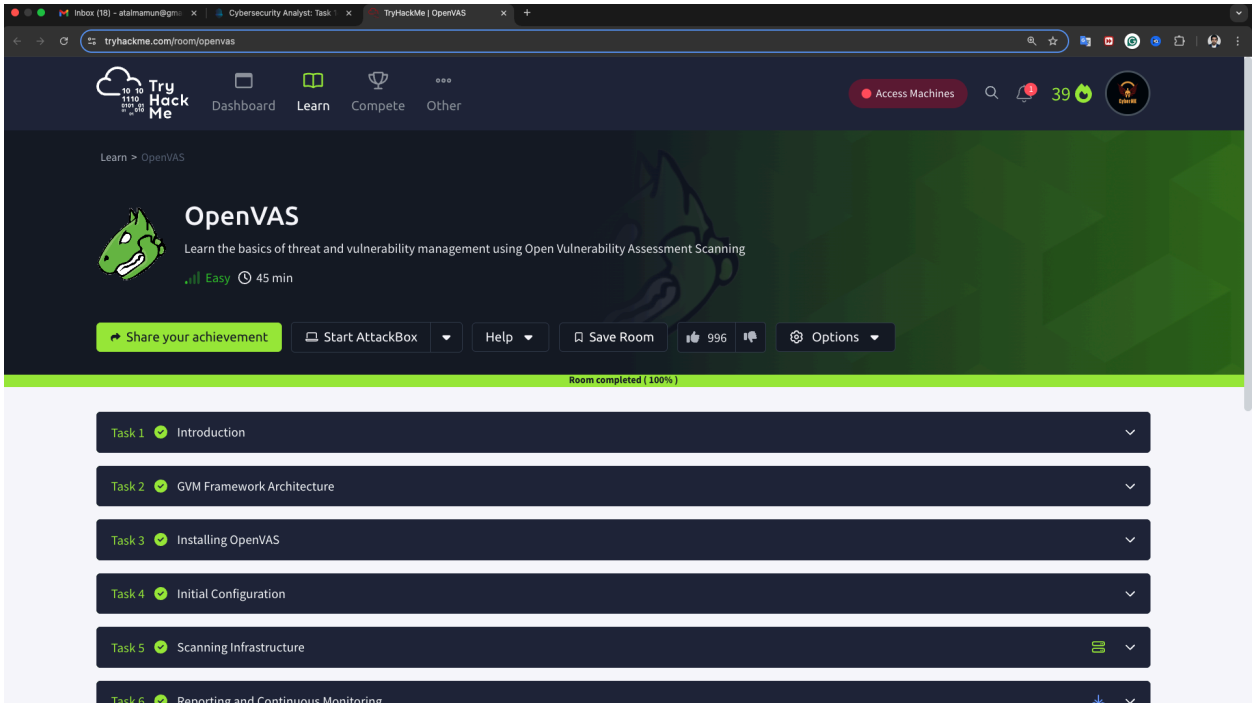


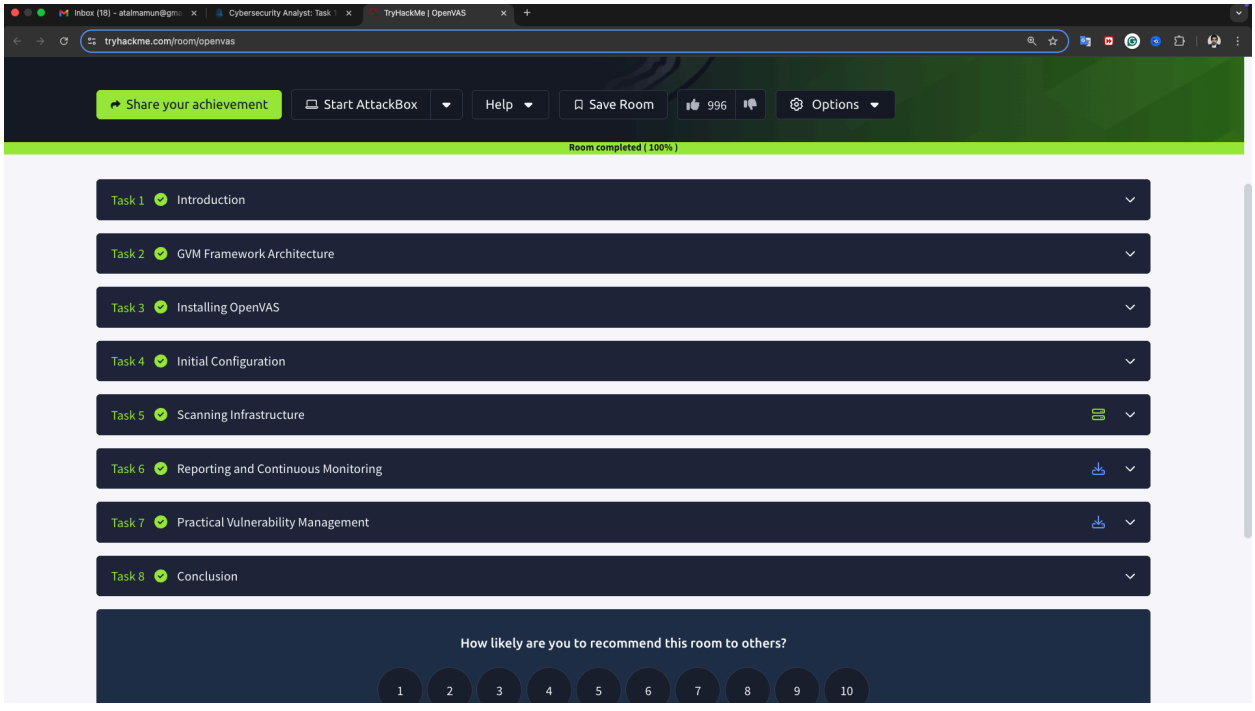
Now that we have our target scoped we can continue to create our task and begin the scan.

Once you create the task you will be brought back to the scan dashboard where you can monitor and start your task. To start the task navigate to the start icon under *Actions*.

Answer the questions below




tryhackme.com/room/openvas
Try Hack Me
Dashboard
Learn
Compete
Other
Access Machines
39
Learn > OpenVAS
OpenVAS
Learn the basics of threat and vulnerability management using Open Vulnerability Assessment Scanning
Easy
45 min
Share your achievement
Start AttackBox
Help
Save Room
996
Options
Room completed (100%)
Task 1 Introduction
Task 2 GVM Framework Architecture
Task 3 Installing OpenVAS
Task 4 Initial Configuration
Task 5 Scanning Infrastructure


tryhackme.com/room/openvas
Share your achievement
Start AttackBox
Help
Save Room
996
Options
Room completed (100%)
Task 1 Introduction
Task 2 GVM Framework Architecture
Task 3 Installing OpenVAS
Task 4 Initial Configuration
Task 5 Scanning Infrastructure
Task 6 Reporting and Continuous Monitoring
Task 7 Practical Vulnerability Management
Task 8 Conclusion
How likely are you to recommend this room to others?
1 2 3 4 5 6 7 8 9 10

6.5 Lab Completion and Reflection

The OpenVAS vulnerability scanning lab provided hands-on experience with identifying and analyzing security vulnerabilities. Through this lab, I was able to:

- Gain practical experience in configuring and using a vulnerability scanner.
- Understand how to interpret vulnerability reports and prioritize patches based on severity.
- Familiarize myself with common vulnerabilities that pose risks to IT systems and how they can be remediated through patching.

This lab was instrumental in reinforcing the theoretical knowledge of vulnerability identification and patch management by providing real-world context and practical skills in handling security assessments.

EncryptEdge Labs

This Internship Task report was developed on [April, 10, 2025]

By:

atalmamun@gmail.com