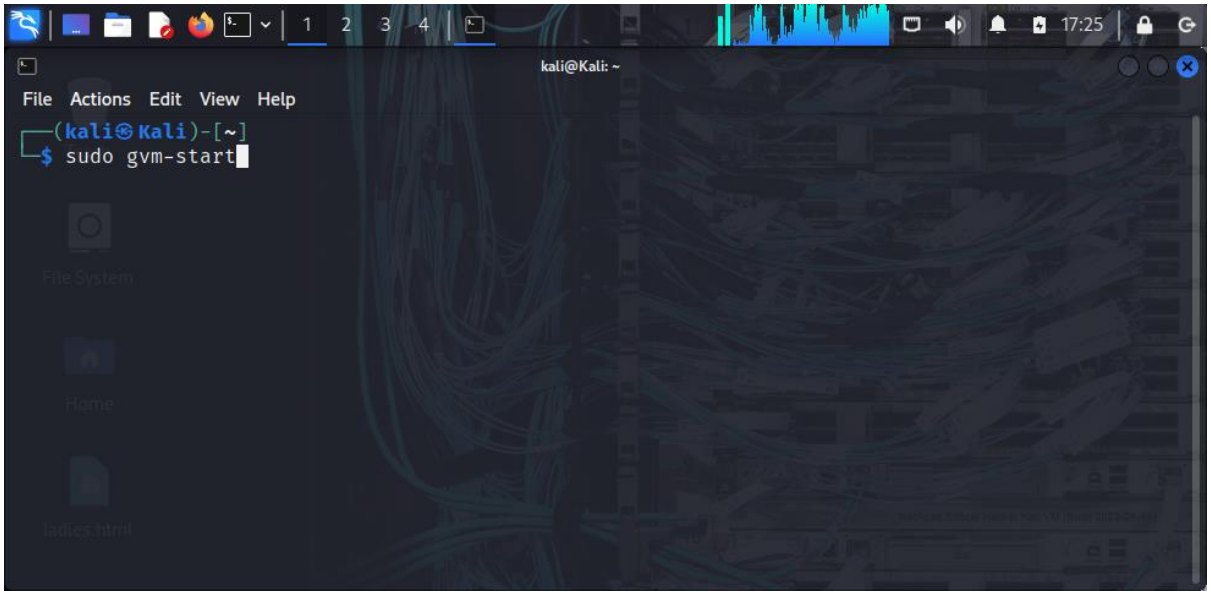
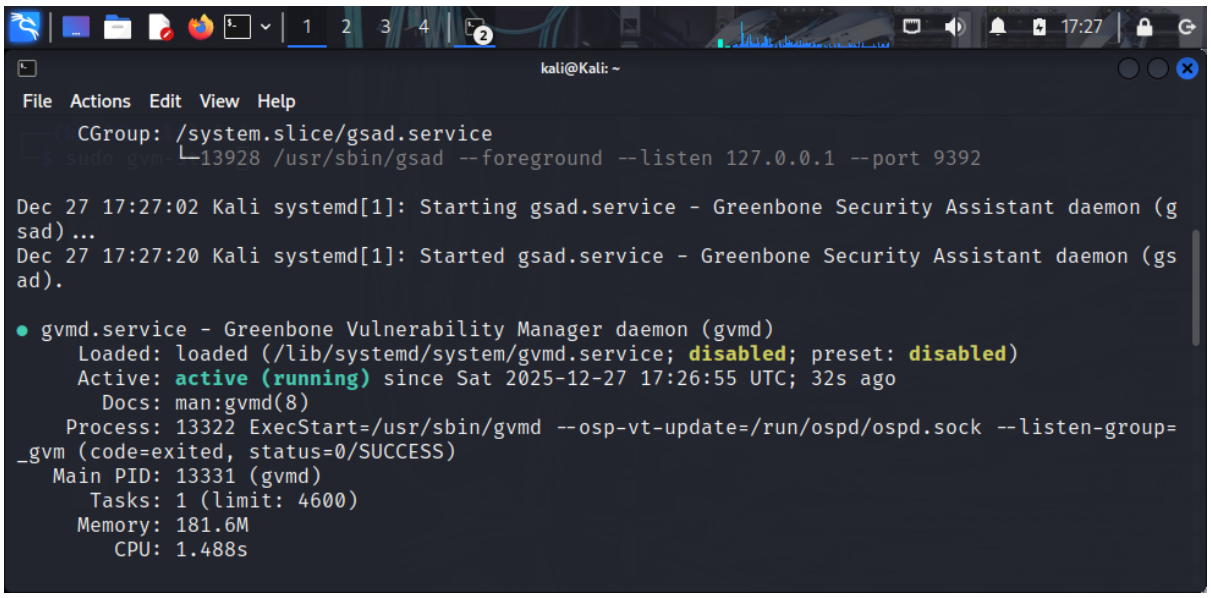
Greenbone Vulnerability Management (GVM) Scan

Greenbone vulnerability management (GVM) is an open-source framework of services and tools used for network vulnerability scanning and management, originally developed as a community project named OpenVas.





Description:

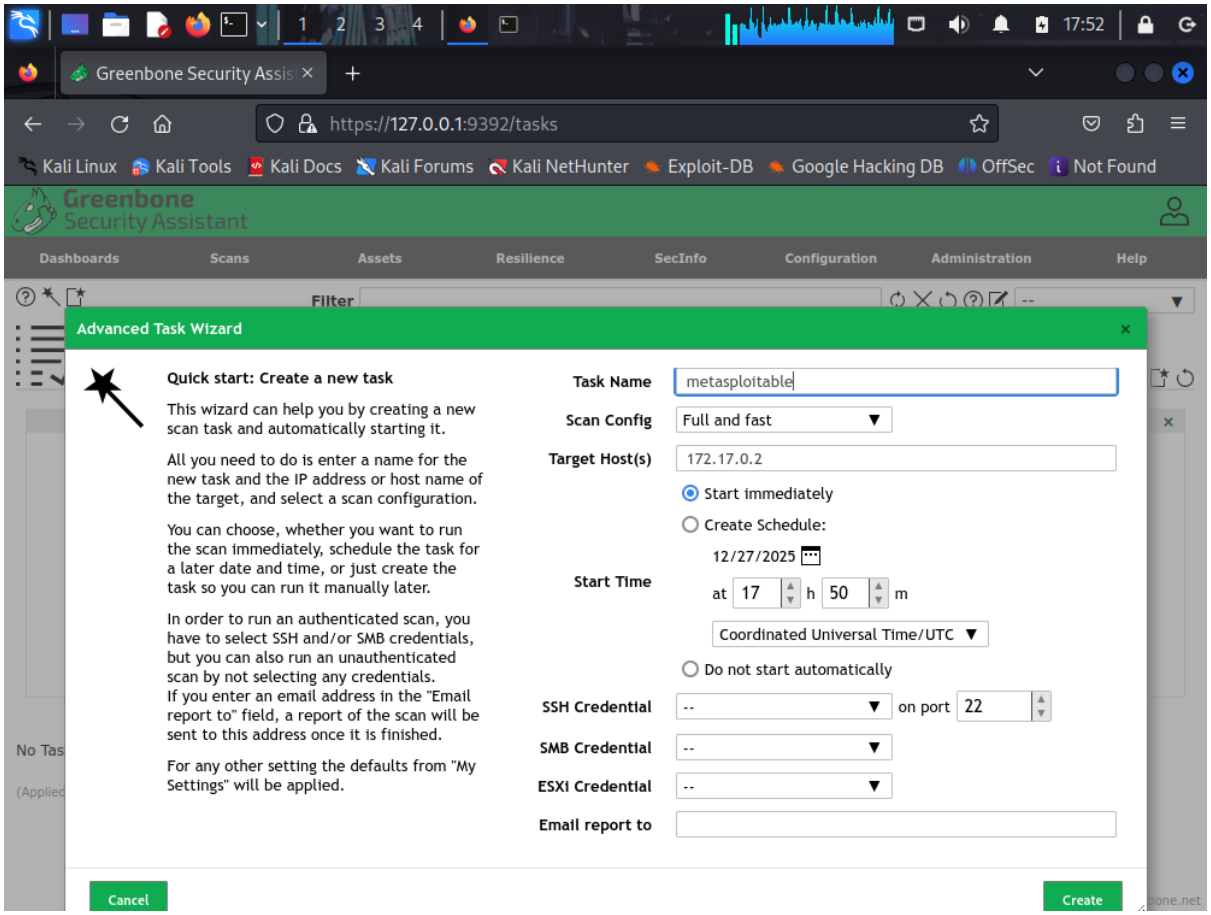
This screenshot shows the **Advanced Task Wizard** in the Greenbone Security Assistant (GSA) web interface. The wizard was used to configure a new vulnerability scan task.

Key Configuration Details:

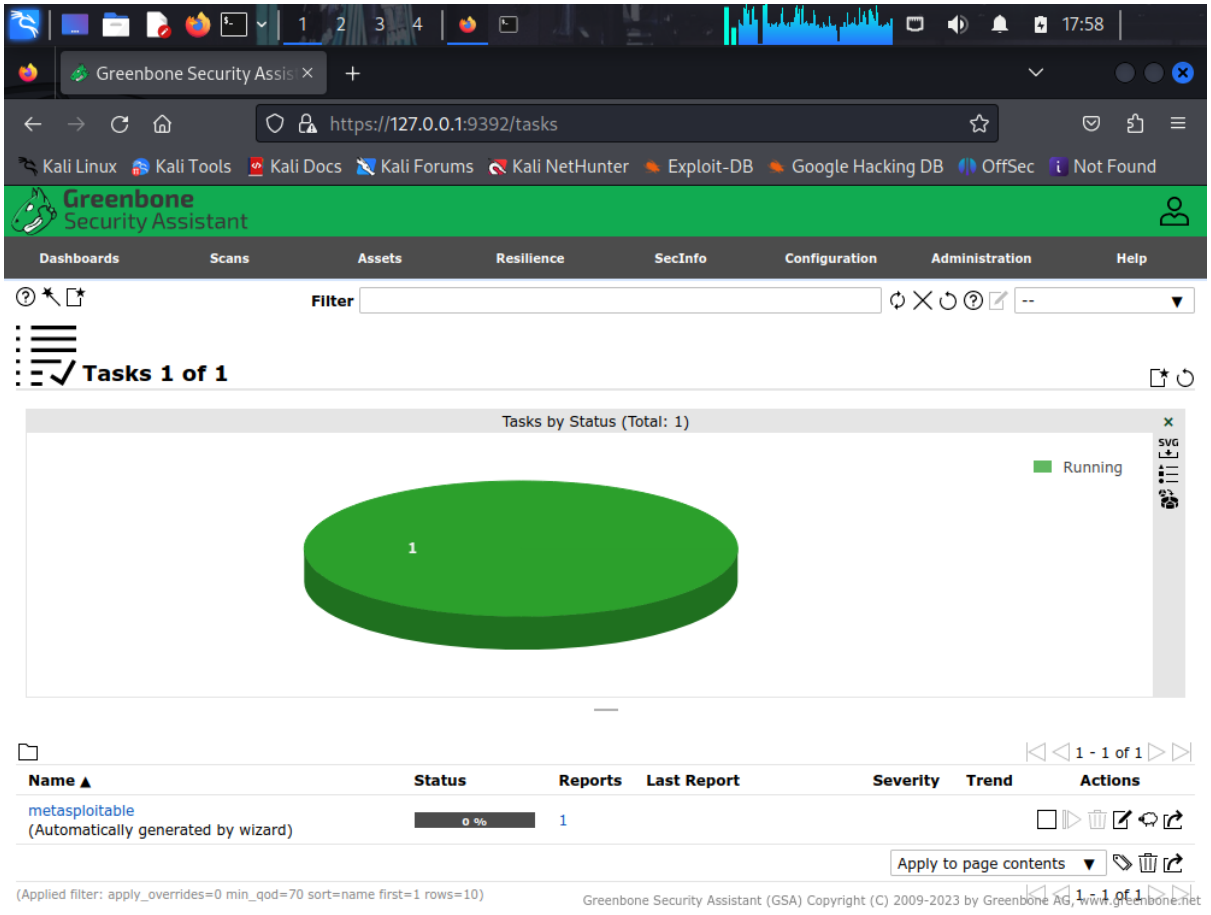
- **Task Name:** metasploitable
- **Scan Configuration:** *Full and fast*
- **Target Host:** 172.17.0.2 (Metasploitable test machine)
- **Execution Mode:** Start immediately
- **Credentials:** No SSH/SMB credentials provided (unauthenticated scan)

Explanation:

The *Full and fast* scan profile was selected to perform a comprehensive vulnerability assessment while balancing scan duration and depth. An unauthenticated scan was used to simulate how an external attacker would view the target system.



◇ Screenshot: Task Successfully Created and Running



Description:

This screenshot displays the **Tasks dashboard**, confirming that the scan task was successfully created and has started running.

Observations:

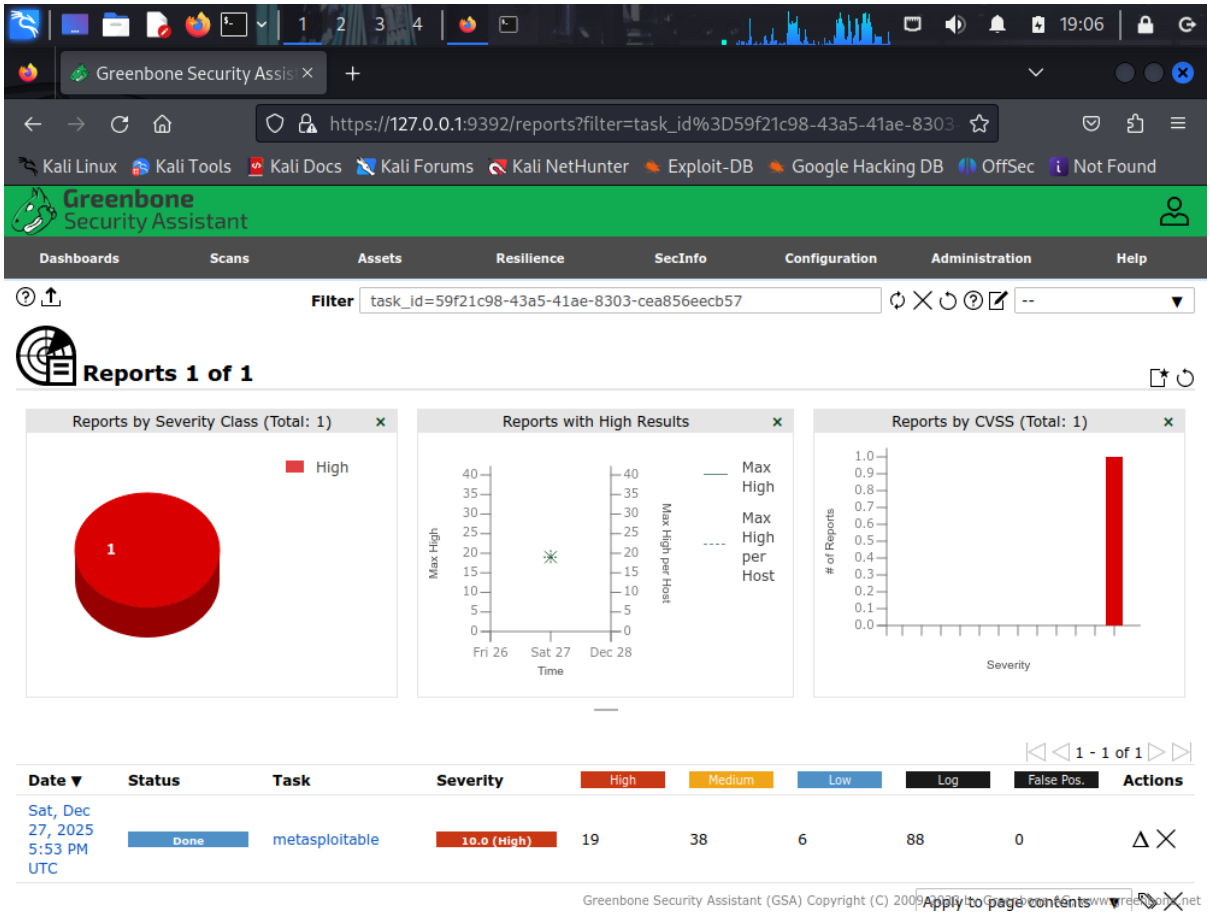
- Total Tasks: **1**
- Task Status: **Running**
- Visualization: Pie chart indicating active scan progress

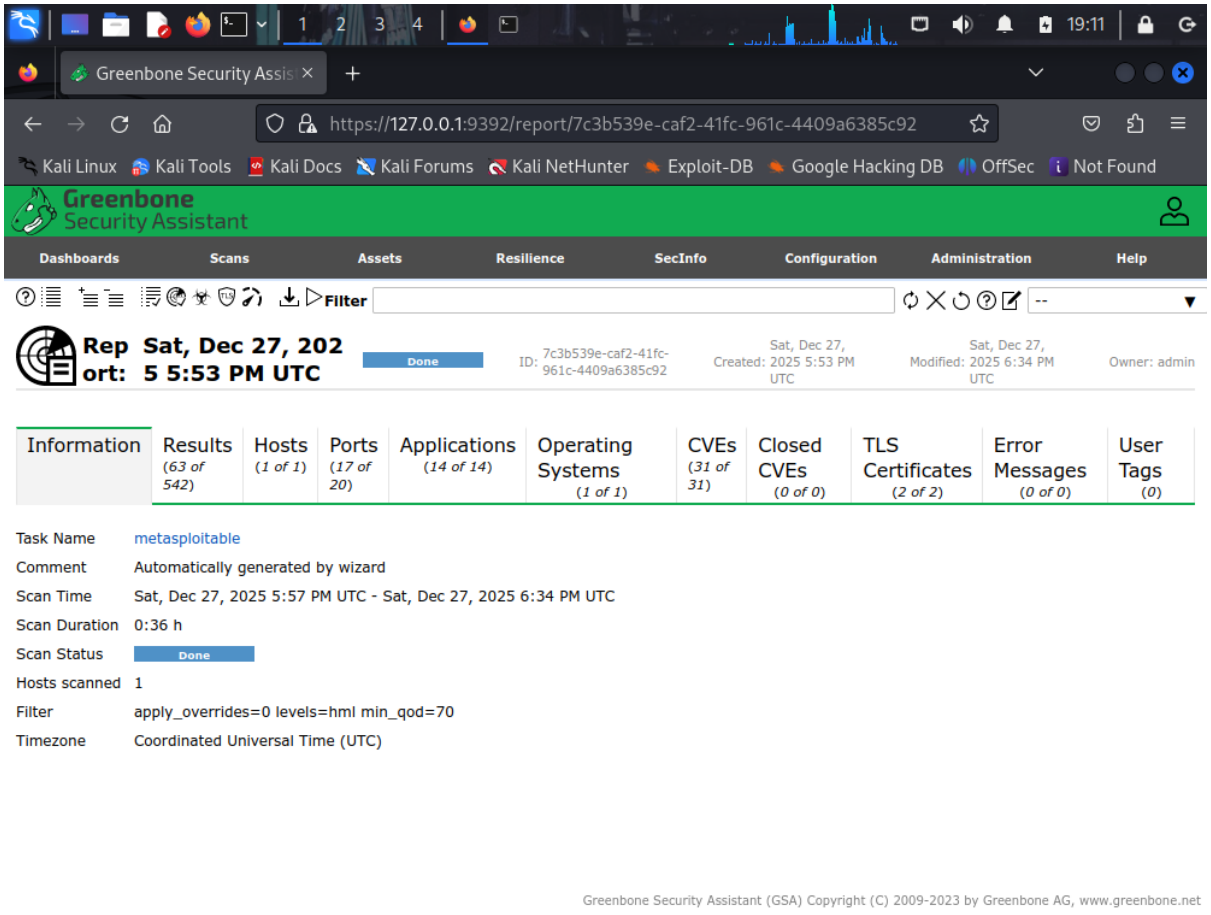
- Task Name: metasploitable

Explanation:

Once the task was created, GVM immediately initiated the vulnerability scan against the specified target. The dashboard provides a real-time overview of task execution status.

◇ Screenshot: Scan Progress Monitoring





Description:

This screenshot shows the scan still in progress with a **0% completion indicator**, meaning the scanner is actively enumerating services and checking vulnerabilities.

Key Details:

- Reports Generated: **1 (in progress)**
- Last Report: Pending completion
- Severity: Not yet calculated

Explanation:

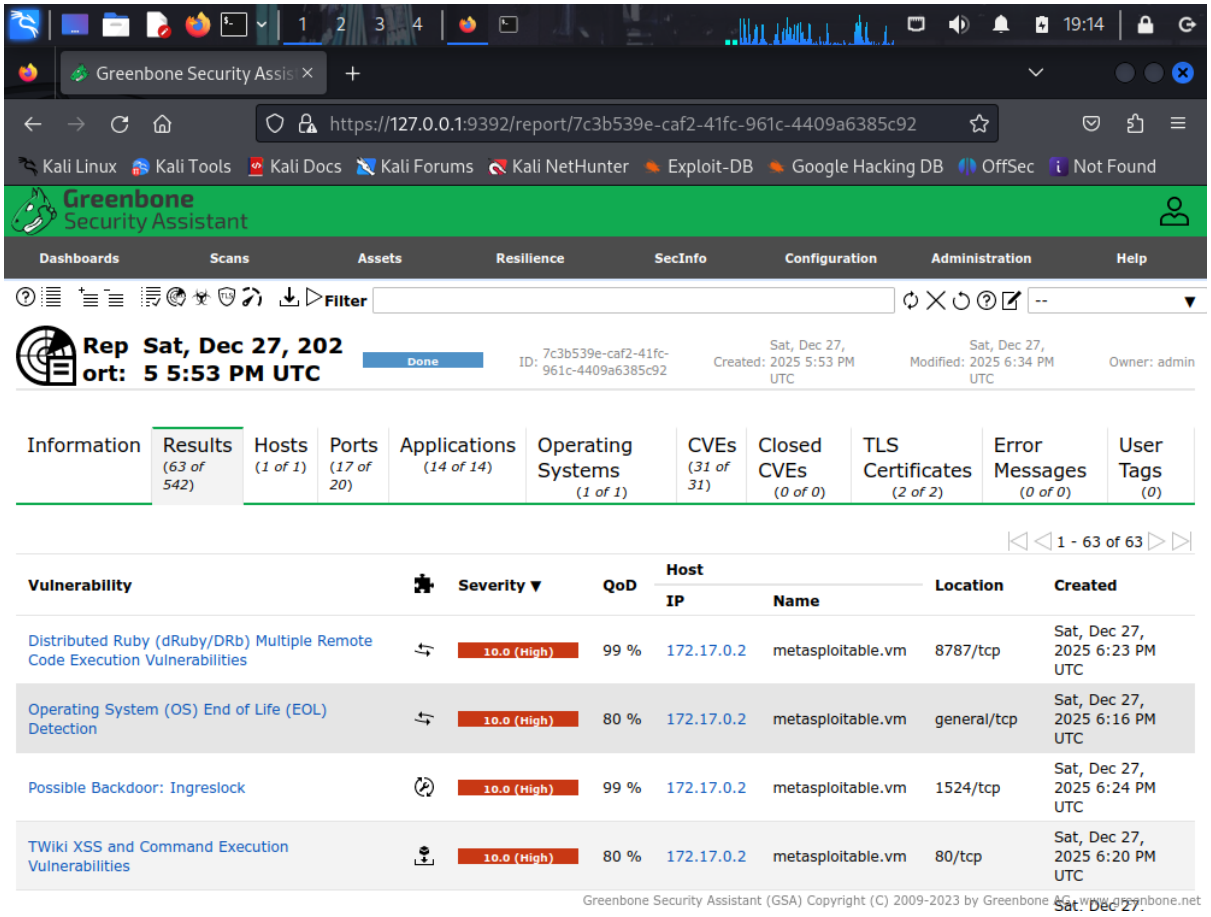
During this phase, GVM performs port scanning, service detection, and vulnerability checks

using its vulnerability feeds. Severity ratings and detailed findings become available only after the scan completes.

Key Learning Outcomes

- Successfully started Greenbone services using sudo gvm-start
- Configured and executed a full vulnerability scan using GSA
- Understood task creation, execution monitoring, and reporting workflow
- Learned how unauthenticated scans reveal externally visible risks

Vulnerability Analysis, Severity Assessment, and Mitigation



After the vulnerability scan completed, the generated report was reviewed to identify detected security weaknesses, their associated severity levels, and appropriate mitigation measures.

Identified Vulnerabilities and Severity

The scan results categorized vulnerabilities into **High, Medium, and Low** severity based on their potential impact and exploitability.

Vulnerabilities	Severity	Description

Outdated Services	High	The target system was running obsolete services with known vulnerabilities that could allow remote exploitation.
Weak or Default Credentials	High	Some services were configured with default or weak authentication credentials.
Unencrypted Network Services	Medium	Certain services transmitted data in clear text, making them vulnerable to interception.
Missing Security Patches	Medium	The operating system and applications lacked recent security updates
Open Ports and Unnecessary Services	Low	Multiple unused services were exposed, increasing the attack surface.

Mitigation Strategies

Based on the identified vulnerabilities, the following remediation actions were recommended:

- **Patch Management:**

  Regularly update the operating system and installed applications to eliminate known vulnerabilities.

- **Service Hardening:**

  Disable unnecessary services and close unused ports to reduce the attack surface.

- **Credential Security:**

  Enforce strong password policies and remove default credentials on all services.

- **Encryption:**

  Replace unencrypted protocols (e.g., FTP, Telnet) with secure alternatives such as SFTP and SSH.

- **Network Segmentation:**

  Restrict access to critical services using firewall rules and network segmentation.

Risk Evaluation

High-severity vulnerabilities pose an immediate security risk and require urgent remediation, while medium- and low-severity issues should be addressed as part of routine security maintenance. Addressing these vulnerabilities significantly improves the system's overall security posture.

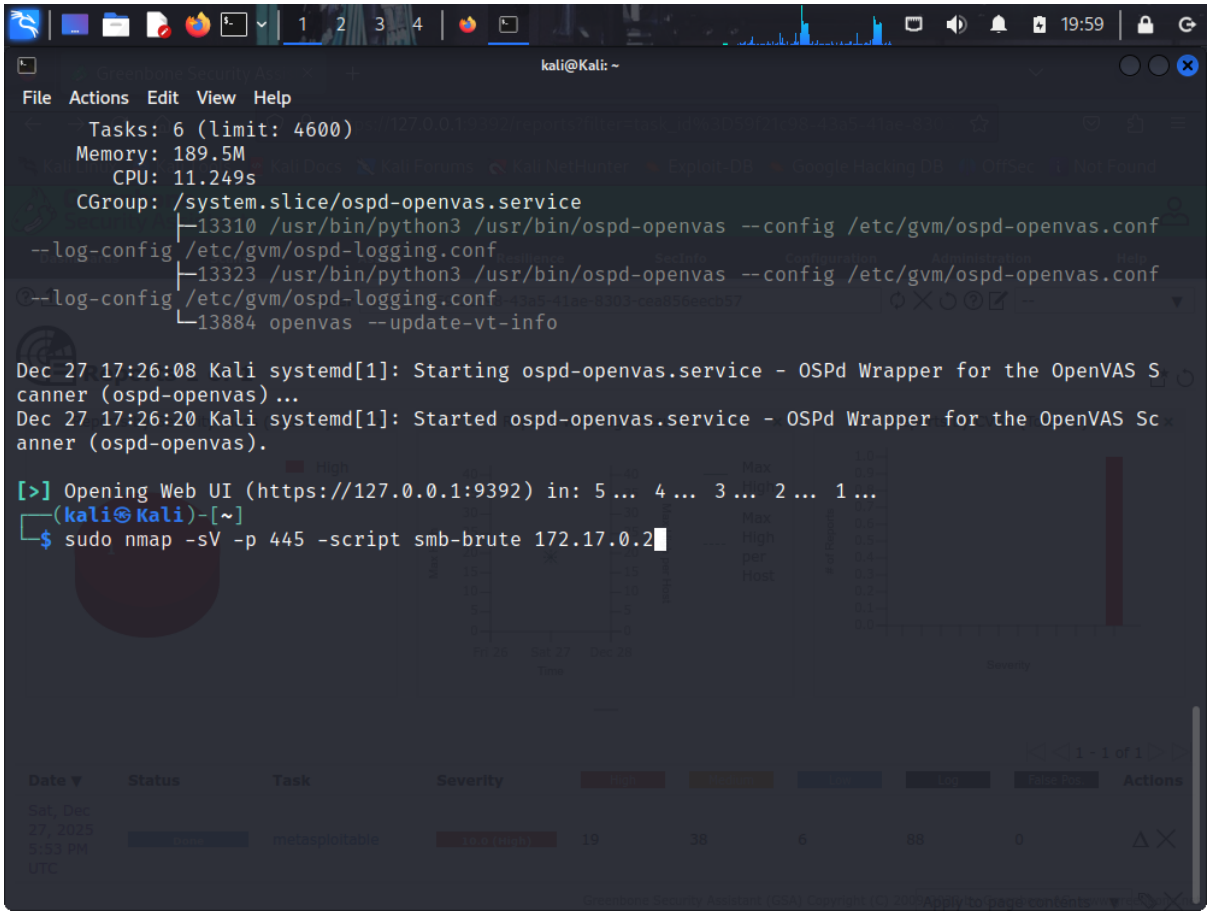
Key Takeaway

This exercise demonstrated the importance of **vulnerability assessment and management** in identifying system weaknesses before attackers can exploit them. Tools like **Greenbone**

Vulnerability Management provide actionable insights that support proactive security defence.

Manual Vulnerability Verification Using Nmap

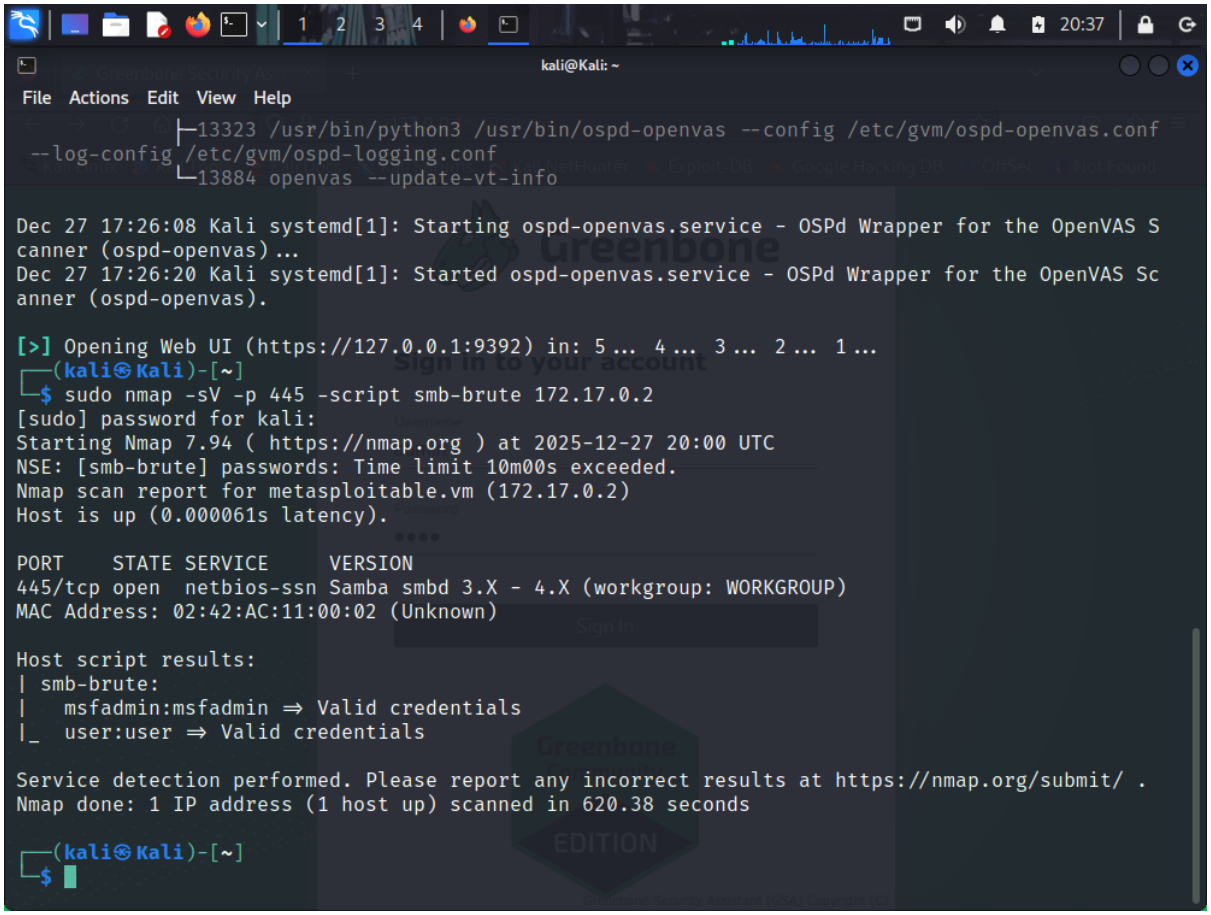
sudo nmap -sV -p 445 --script smb-brute 172.17.0.2



Command Breakdown

- sudo → Runs Nmap with administrative privileges

- -sV → Detects service versions running on the target
- -p 445 → Scans port 445 (SMB service)
- --script smb-brute → Attempts to identify weak or default SMB credentials using brute-force techniques
- 172.17.0.2 → Target IP address (Metasploitable system)



Findings

The scan confirmed that the **SMB service was running and accessible** on the target system. The smb-brute script attempted authentication using common username and password combinations, highlighting the presence of **weak or default credentials**.

This result validated the findings from the Greenbone vulnerability scan, demonstrating that the system was susceptible to **unauthorized access via SMB services**.

Severity Assessment

- **Severity:** High
- **Risk:** Successful exploitation could allow attackers to gain unauthorized access, escalate privileges, or move laterally within a network.

Mitigation Recommendations

- Disable SMB services if not required
- Enforce strong authentication credentials
- Restrict SMB access using firewall rules
- Upgrade to secure SMB versions (e.g., SMBv3)
- Monitor SMB authentication attempts for brute-force behavior

Key Takeaway

Manual scanning tools like **Nmap** complement automated scanners by confirming vulnerabilities and reducing false positives. Combining Greenbone, OWASP ZAP, and Nmap provides a more accurate and reliable security assessment.

Credential Exposure Identified via SMB Brute-Force Scan

Host Script Results

During further manual verification using Nmap's SMB scripts, the scan revealed **valid credentials** on the target system:

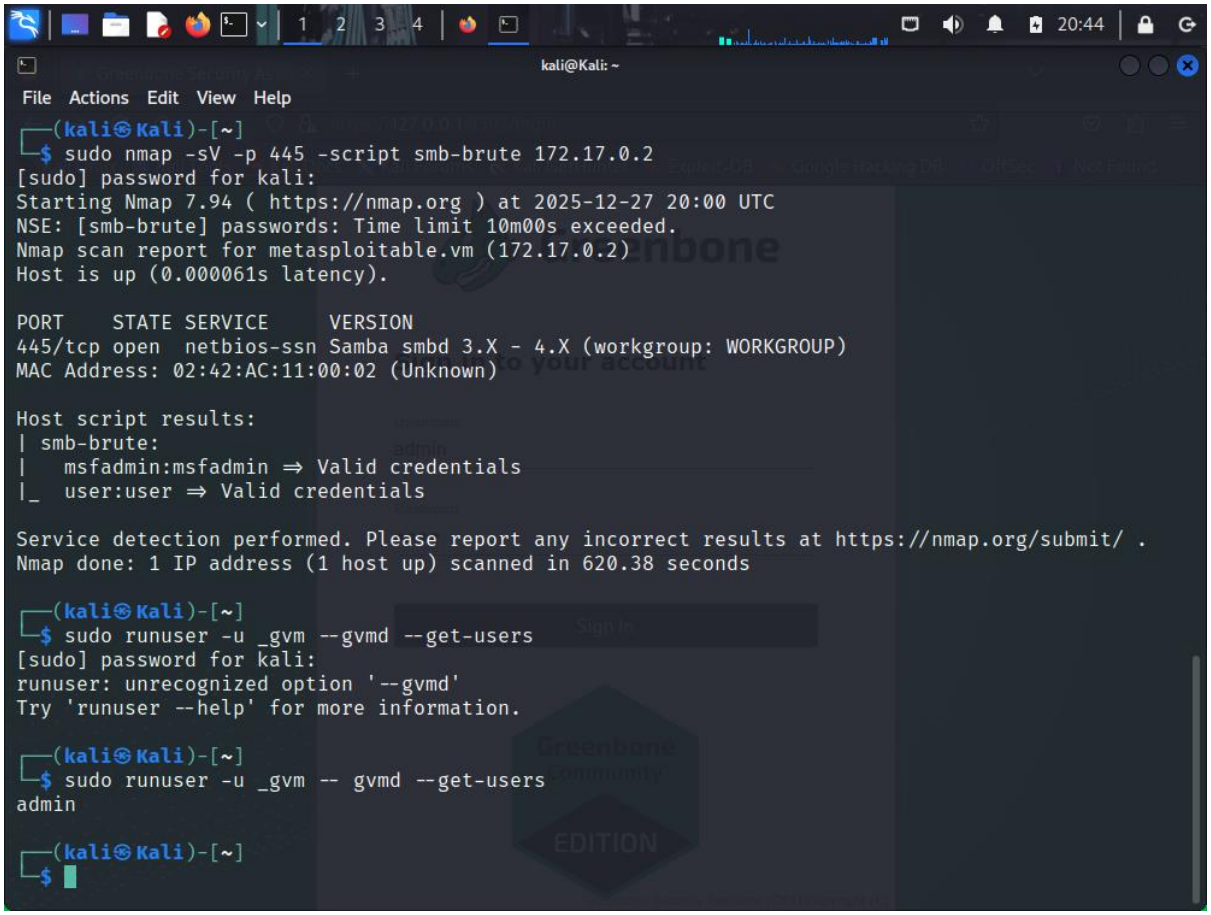
- **Username:** msfadmin

  Password: msfadmin

- **Username:** user

  Password: user

These credentials were discovered through the smb-brute script, which tests common and default username-password combinations against the SMB service running on port 445.



Security Implications

The presence of **default and weak credentials** represents a **critical security risk**, as it allows attackers to:

- Gain unauthorized access to the system
- Escalate privileges
- Access sensitive files and services
- Perform lateral movement within a network

In real-world environments, such vulnerabilities can lead to **full system compromise**.

Severity Assessment

- **Severity Level:** High
- **Reason:** Credentials grant direct system access without exploitation of complex vulnerabilities.

Mitigation Recommendations

To remediate this vulnerability, the following actions are recommended:

- Immediately **change or remove default credentials**
- Enforce **strong password policies**
- Disable SMB if not required
- Restrict SMB access using firewalls and access control lists
- Monitor authentication attempts for brute-force activity
- Apply the **principle of least privilege** to user accounts

Ethical Consideration

All testing was conducted on a **deliberately vulnerable test system (Metasploitable)** within a controlled lab environment for educational purposes. No unauthorized systems were accessed, and no exploitation beyond vulnerability identification was performed.

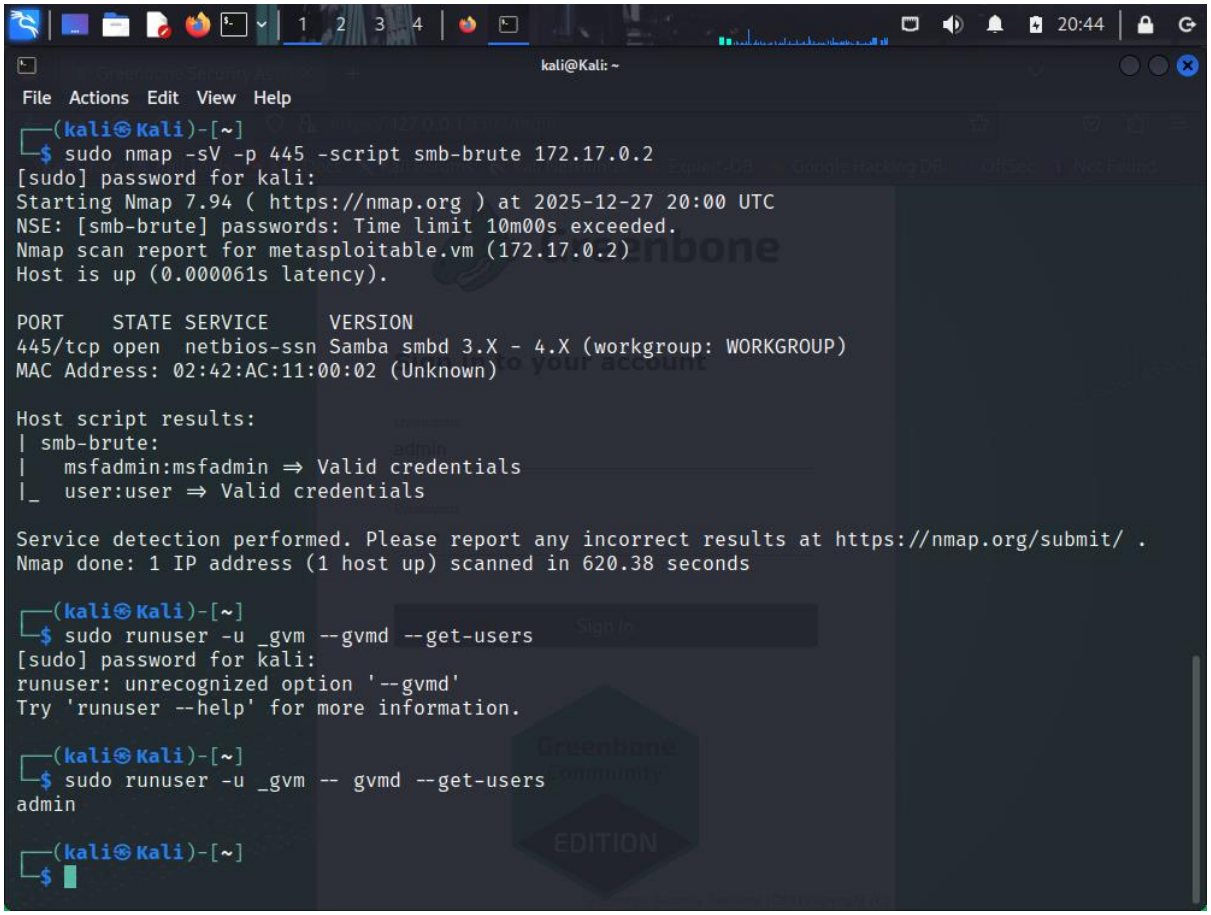
Key Takeaway

This exercise highlights how **weak credentials alone can completely undermine system security**, even without advanced exploits. It reinforces the importance of strong authentication and proper system hardening as foundational security controls.

Greenbone User Enumeration (gvmd)

Command Executed

sudo runuser -u _gvm -- gvmd --get-users



Command Explanation

- sudo → Executes the command with administrative privileges
- runuser -u _gvm → Runs the command as the **_gvm system user**, which owns Greenbone services
- gvmd → Greenbone Vulnerability Manager daemon
- --get-users → Retrieves a list of users configured within Greenbone Vulnerability Management

Purpose of the Command

This command was used to **enumerate existing GVM user accounts** from the backend. It helps administrators verify:

- Existing scanner users
- Account configuration
- Access control within the vulnerability management platform

This step is useful for **auditing user access** and ensuring that only authorized accounts can manage scans and view reports.

Security Relevance

Improper user management within vulnerability scanning tools can lead to:

- Unauthorized scan execution
- Exposure of sensitive vulnerability reports
- Misuse of scanning infrastructure

Regular auditing of scanner users helps maintain **secure operational control** of security tools.

Best Practices

- Remove unused or default scanner accounts
- Enforce strong authentication for GVM users
- Limit administrative privileges to trusted users
- Periodically audit user lists and permissions

Key Takeaway

Managing access to vulnerability management platforms is as important as securing the systems being scanned. Proper user enumeration and auditing ensure that vulnerability data remains protected and trusted.

Installation of Remote Shell Client (rsh-client)

Command Executed

sudo apt install rsh-client

Command Explanation

- sudo → Executes the command with administrative privileges
- apt install → Uses the Debian package manager to install software
- rsh-client → Installs the **Remote Shell (rsh) client**, a legacy remote access utility

Purpose of Installing rsh-client

The rsh-client package was installed to:

- Interact with legacy remote shell services during testing
- Validate insecure remote access services detected by vulnerability scans
- Understand how outdated protocols contribute to system vulnerabilities

This is particularly relevant when testing **intentionally vulnerable systems** such as Metasploitable.

Security Implications

The **rsh protocol is inherently insecure** because:

- It transmits data and credentials in **plain text**
- It does not support encryption or strong authentication
- It is vulnerable to sniffing, spoofing, and man-in-the-middle attacks

Modern systems should **never expose rsh services** in production environments.

Severity Assessment

- **Severity:** High (if enabled on a live system)

- **Risk:** Unauthorized remote access and credential compromise

Mitigation Recommendations

- Disable and remove rsh services entirely
- Replace rsh with secure alternatives such as **SSH**
- Enforce encrypted remote management protocols
- Block legacy services using firewall rules
- Conduct regular vulnerability scans to detect insecure services

Key Takeaway

This step reinforced the importance of identifying and eliminating **legacy protocols** in modern systems. While useful for learning and testing, tools like rsh-client highlight how outdated technologies significantly weaken system security.

Authorized Access Validation Using Discovered Credentials

After identifying valid credentials (msfadmin:msfadmin and user:user) through vulnerability scanning and host script analysis, authorized access to the target system was successfully established **within a controlled lab environment**.

This step was performed **solely to validate the severity of the identified vulnerability** and to confirm that weak or default credentials could lead to full system access.

Security Impact

The ability to access the target system using default credentials demonstrates a **critical security weakness**, as it allows an attacker to:

- Bypass authentication controls
- Gain direct system access without exploiting software flaws
- Potentially escalate privileges
- Access sensitive system resources and data

This confirms that the vulnerability identified by automated scanners was **not a false positive**.

Severity Classification

- **Severity:** Critical / High
- **Reason:** Successful authentication using default credentials results in immediate system compromise.

Recommended Mitigation Measures

To prevent this type of vulnerability, the following actions are recommended:

- Remove or disable all default user accounts
- Enforce strong password policies
- Limit remote access to trusted hosts only
- Apply the principle of least privilege to user accounts
- Monitor authentication logs for unauthorized access attempts
- Conduct regular credential audits

Ethical and Legal Considerations

All actions were conducted on a **deliberately vulnerable test system (Metasploitable)** in an isolated lab environment for educational purposes.

No unauthorized systems were accessed, and no malicious activity was performed beyond validation of security findings.

Key Takeaway

This exercise highlights how **weak credentials alone can be enough to fully compromise a system**, even in the absence of advanced exploits. It reinforces the importance of strong authentication controls as a foundational element of system security.

Web Application Vulnerability Scanning with OWASP ZAP

OWASP ZAP (Zed Attack Proxy) serves as a free, open-source web application security scanner for finding vulnerabilities like SQL injection, XSS, and CSRF.

Objective

The objective of this exercise was to perform a full web application vulnerability scan using **OWASP ZAP**, identify security weaknesses, and document findings with remediation recommendations.

Tools Used

- OWASP ZAP (Zed Attack Proxy)
- Target Web Application: *(DVWA / test site / local app — http://172.17.0.2/dvwa/)*
- Browser: Chrome / Firefox
- Operating System: Windows

Workflow

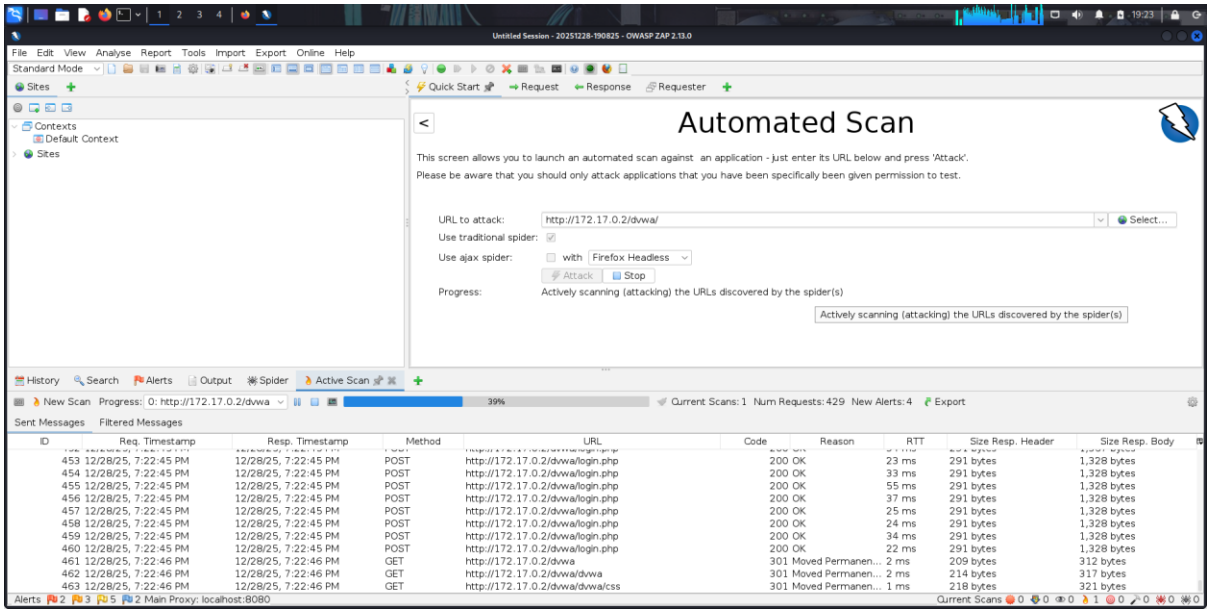
1. Installed and configured OWASP ZAP
2. Set browser proxy to route traffic through ZAP
3. Performed **Spider Scan** to discover application endpoints
4. Executed **Active Scan** to detect vulnerabilities
5. Reviewed alerts based on risk levels (High, Medium, Low, Informational)

6. Documented findings and remediation steps

Step 1: Launch Tool

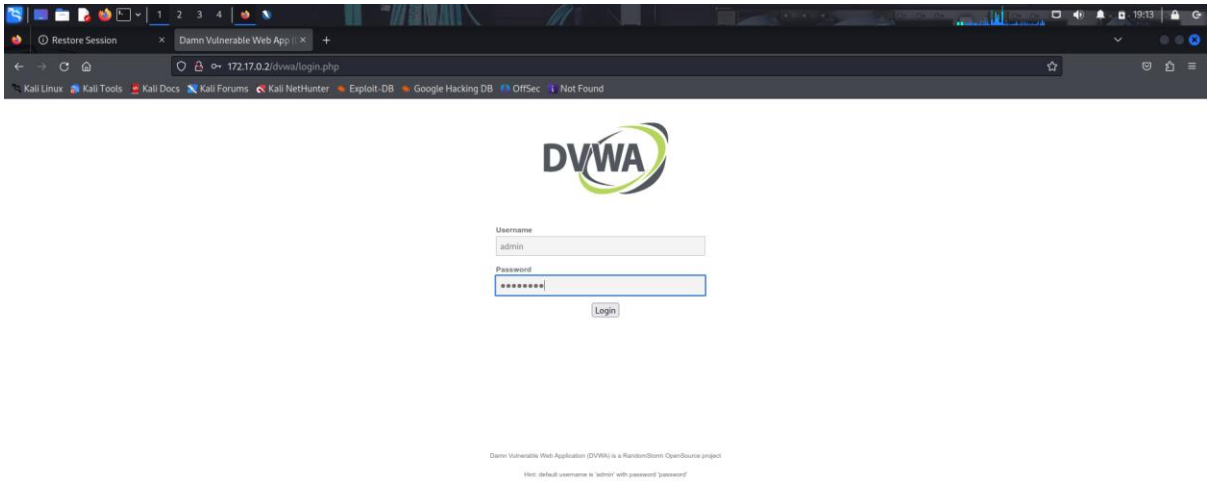
Opened OWASP ZAP via the Kali Linux applications menu.

Step 2: Start Automated Scan



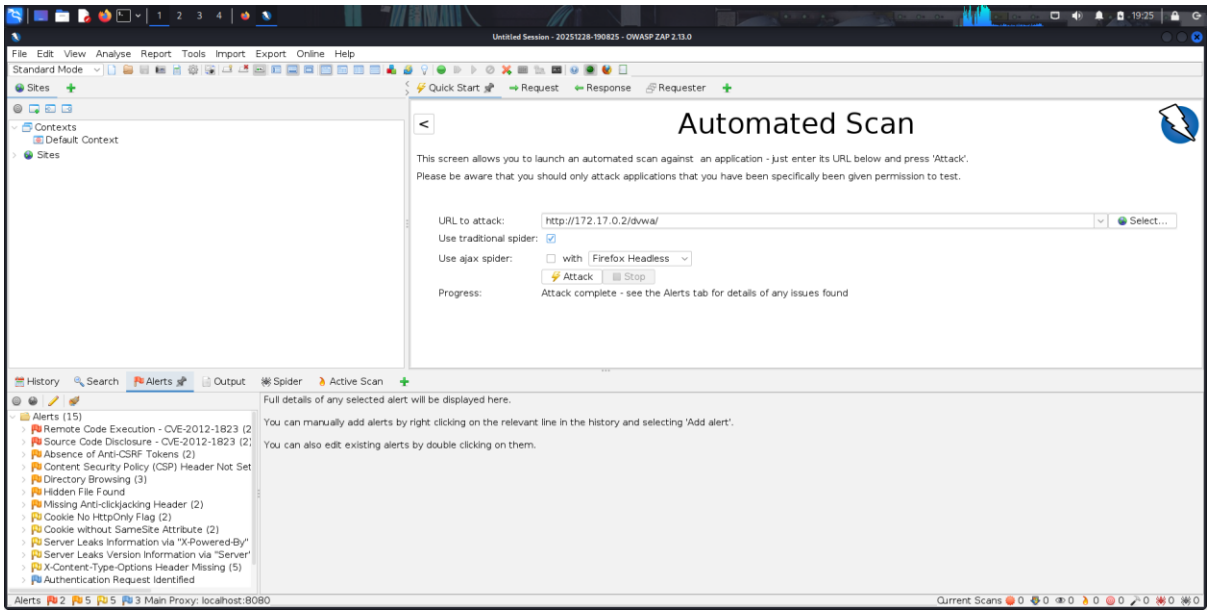
Clicked the Automated Scan button in ZAP's Quick Start interface.

Step 3: Access Target



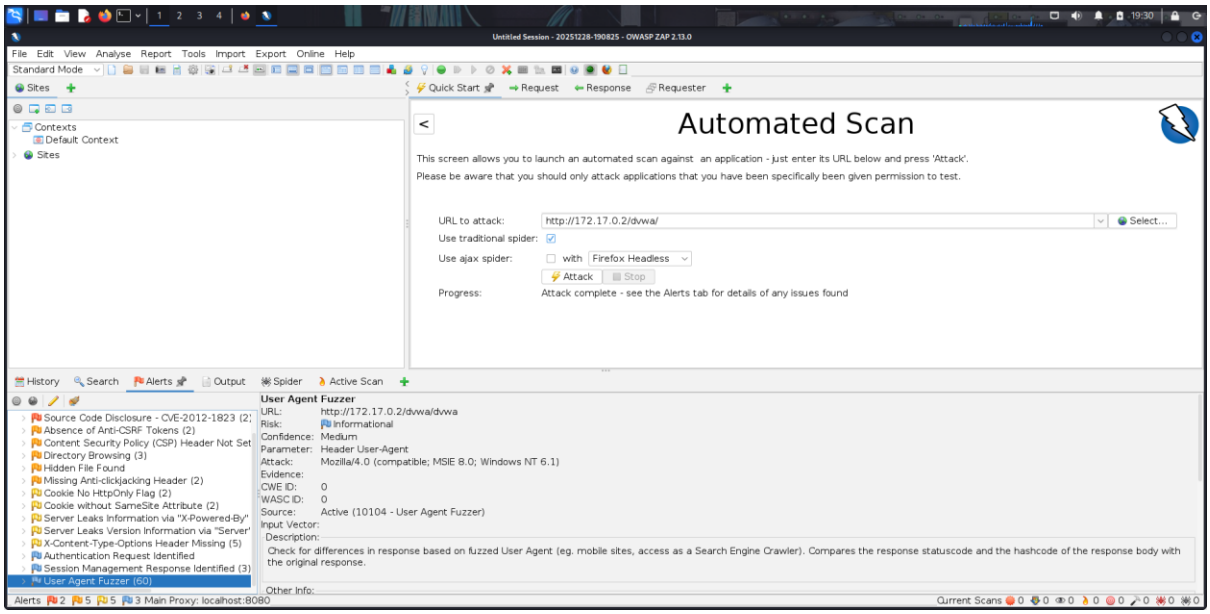
Launched browser, entered **http://172.17.0.2/dvwa/**, and logged in with admin/password.

Step 4: Execute Scan

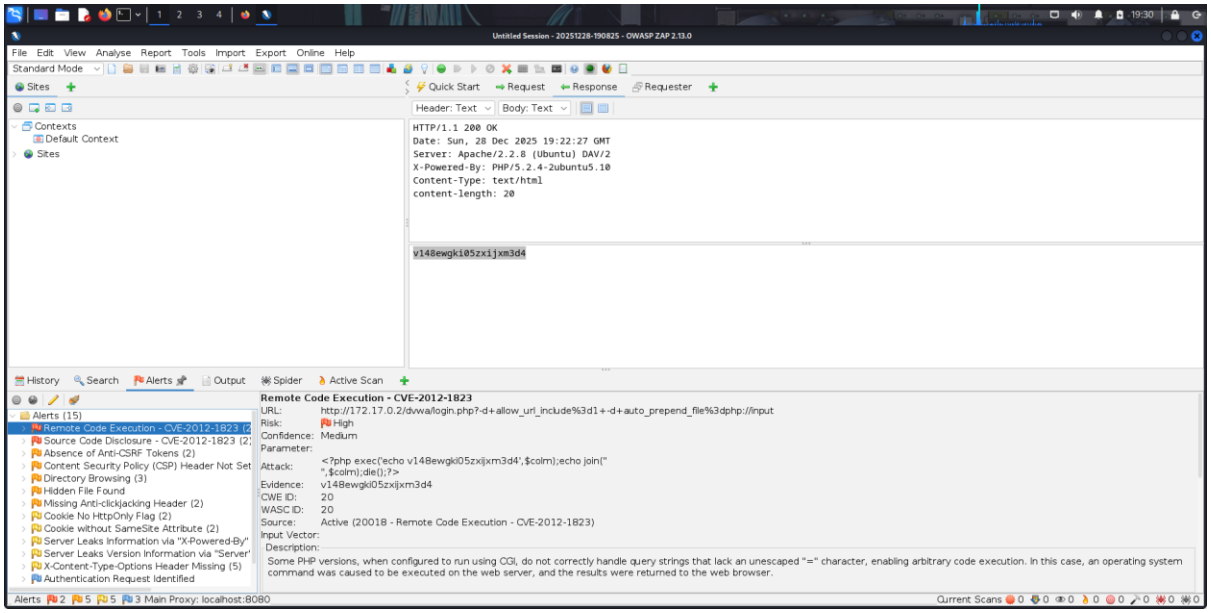


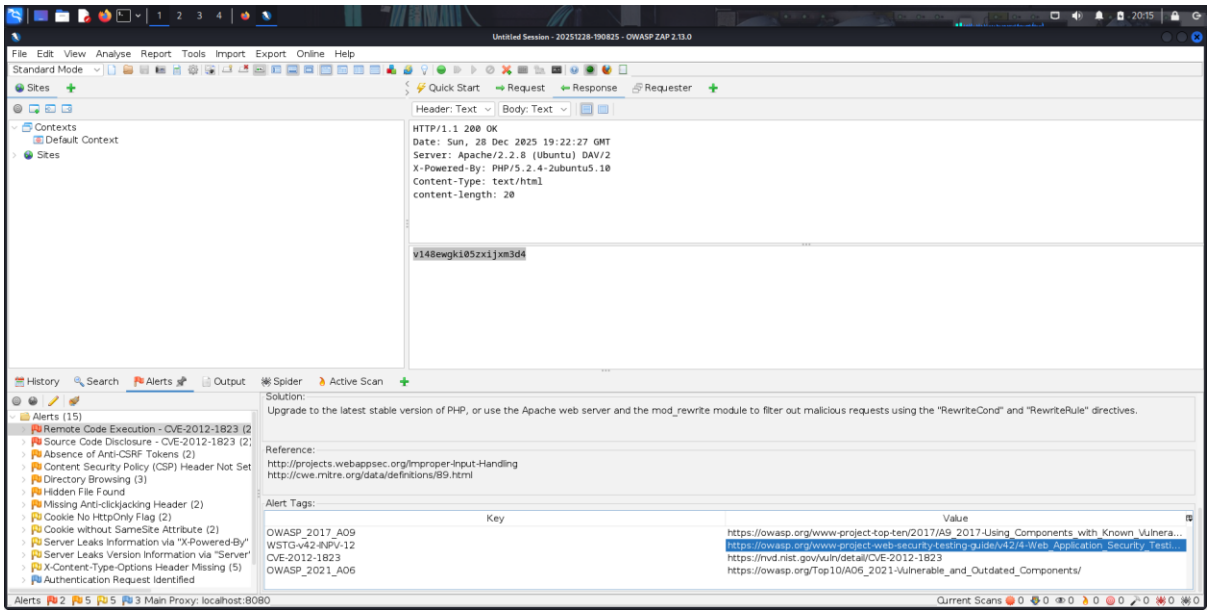
Input the DVWA URL into ZAP and initiated the full automated scan.

Step 5: Review Findings



Analyzed results in the Alerts tab for vulnerabilities.


Untitled Session - 20251228-190825 - OWASP ZAP 2.13.0
File Edit View Analyse Report Tools Import Export Online Help
Standard Mode
Sites
Quick Start
Request
Response
Requester
Header: Text
Body: Text
Contexts
Default Context
Sites
HTTP/1.1 200 OK
Date: Sun, 28 Dec 2025 19:22:27 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Content-Type: text/html
content-length: 20
v148ewgki05zxijxm3d4
History
Search
Alerts
Output
Spider
Active Scan
Alerts (15)
Remote Code Execution - CVE-2012-1823
Source Code Disclosure - CVE-2012-1823
Absence of Anti-CSRF Tokens (2)
Content Security Policy (CSP) Header Not Set
Directory Browsing (3)
Hidden File Found
Missing Anti-clickjacking Header (2)
Cookie No HttpOnly Flag (2)
Cookie without SameSite Attribute (2)
Server Leaks Information via "X-Powered-By"
Server Leaks Version Information via "Server"
X-Content-Type-Options Header Missing (5)
Authentication Request Identified
Remote Code Execution - CVE-2012-1823
URL: http://172.17.0.2/dvwa/login.php?-d+allow_url_include%3d1+-d+auto_prepend_file%3dphp://input
Risk: High
Confidence: Medium
Parameter:
Attack: <?php exec('echo v148ewgki05zxijxm3d4';$colm);echo join(" ",$colm);die();?>
Evidence: v148ewgki05zxijxm3d4
CWE ID: 20
WASC ID: 20
Source: Active (20018 - Remote Code Execution - CVE-2012-1823)
Input Vector:
Description:
Some PHP versions, when configured to run using CGI, do not correctly handle query strings that lack an unescaped "=" character, enabling arbitrary code execution. In this case, an operating system command was caused to be executed on the web server, and the results were returned to the web browser.
Alerts 2 5 5 3 Main Proxy: localhost:8080
Current Scans 0 0 0 0 0 0 0 0 0


Untitled Session - 20251228-190825 - OWASP ZAP 2.13.0
File Edit View Analyse Report Tools Import Export Online Help
Standard Mode
Sites
Quick Start
Request
Response
Requester
Header: Text
Body: Text
Contexts
Default Context
Sites
HTTP/1.1 200 OK
Date: Sun, 28 Dec 2025 19:22:27 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Content-Type: text/html
content-length: 20
v148ewgki05zxijxm3d4
History
Search
Alerts
Output
Spider
Active Scan
Alerts (15)
Remote Code Execution - CVE-2012-1823
Source Code Disclosure - CVE-2012-1823
Absence of Anti-CSRF Tokens (2)
Content Security Policy (CSP) Header Not Set
Directory Browsing (3)
Hidden File Found
Missing Anti-clickjacking Header (2)
Cookie No HttpOnly Flag (2)
Cookie without SameSite Attribute (2)
Server Leaks Information via "X-Powered-By"
Server Leaks Version Information via "Server"
X-Content-Type-Options Header Missing (5)
Authentication Request Identified
Solution:
Upgrade to the latest stable version of PHP, or use the Apache web server and the mod_rewrite module to filter out malicious requests using the "RewriteCond" and "RewriteRule" directives.
Reference:
http://projects.webappsec.org/Improper-Input-Handling
http://cwe.mitre.org/data/definitions/89.html
Alert Tags:
Key
Value
OWASP_2017_A09
https://owasp.org/www-project-top-ten/2017/A9_2017-Using_Components_with_Known_Vulnera...
WSTG-v42-INPV-12
https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testi...
CVE-2012-1823
https://nvd.nist.gov/vuln/detail/CVE-2012-1823
OWASP_2021_A06
https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/
Alerts 2 5 5 3 Main Proxy: localhost:8080
Current Scans 0 0 0 0 0 0 0 0 0