Rekall Corporation

Penetration Test Report

Ultimate Pentesting, LLC

Confidentiality Statement

This document contains confidential and privileged information from Rekall Inc. (henceforth known as Rekall). The information contained in this document is confidential and may constitute inside or non-public information under international, federal, or state laws. Unauthorized forwarding, printing, copying, distribution, or use of such information is strictly prohibited and may be unlawful. If you are not the intended recipient, be aware that any disclosure, copying, or distribution of this document or its parts is prohibited.

Table of Contents

Contact Information

Company Name	Ultimate Pentesting, LLC
Contact Name	Collin Janecka
Contact Title	Penetration Tester

Document History

Version	Date	Author(s)	Comments
001	09/10/2023	Collin Janecka	Initial Draft v1
002	09/11/2023	Collin Janecka	Draft v1 Revision 01
003	09/12/2023	Collin Janecka	Draft v1 Revision 02
004	09/13/2023	Collin Janecka	Final Draft v1

Introduction

In accordance with Rekall policies, our organization conducts external and internal penetration tests of its networks and systems throughout the year. The purpose of this engagement was to assess the networks' and systems' security and identify potential security flaws by utilizing industry-accepted testing methodology and best practices.

For the testing, we focused on the following:

- Attempting to determine what system-level vulnerabilities could be discovered and exploited with no prior knowledge of the environment or notification to administrators.
- Attempting to exploit vulnerabilities found and access confidential information that may be stored on systems.
- Documenting and reporting on all findings.

All tests took into consideration the actual business processes implemented by the systems and their potential threats; therefore, the results of this assessment reflect a realistic picture of the actual exposure levels to online hackers. This document contains the results of that assessment.

Assessment Objective

The primary goal of this assessment was to provide an analysis of security flaws present in Rekall's web applications, networks, and systems. This assessment was conducted to identify exploitable vulnerabilities and provide actionable recommendations on how to remediate the vulnerabilities to provide a greater level of security for the environment.

We used our proven vulnerability testing methodology to assess all relevant web applications, networks, and systems in scope.

Rekall has outlined the following objectives:

Table 1: Defined Objectives

Objective
Find and exfiltrate any sensitive information within the domain.
Escalate privileges.
Compromise several machines.

Penetration Testing Methodology

Reconnaissance

We begin assessments by checking for any passive (open source) data that may assist the assessors with their tasks. If internal, the assessment team will perform active recon using tools such as Nmap and Bloodhound.

Identification of Vulnerabilities and Services

We use custom, private, and public tools such as Metasploit, hashcat, and Nmap to gain perspective of the network security from a hacker's point of view. These methods provide Rekall with an understanding of the risks that threaten its information, and also the strengths and weaknesses of the current controls protecting those systems. The results were achieved by mapping the network architecture, identifying hosts and services, enumerating network and system-level vulnerabilities, attempting to discover unexpected hosts within the environment, and eliminating false positives that might have arisen from scanning.

Vulnerability Exploitation

Our normal process is to both manually test each identified vulnerability and use automated tools to exploit these issues. Exploitation of a vulnerability is defined as any action we perform that gives us unauthorized access to the system or the sensitive data.

Reporting

Once exploitation is completed and the assessors have completed their objectives, or have done everything possible within the allotted time, the assessment team writes the report, which is the final deliverable to the customer.

Scope

Prior to any assessment activities, Rekall and the assessment team will identify targeted systems with a defined range or list of network IP addresses. The assessment team will work directly with the Rekall POC to determine which network ranges are in-scope for the scheduled assessment.

It is Rekall's responsibility to ensure that IP addresses identified as in-scope are actually controlled by Rekall and are hosted in Rekall-owned facilities (i.e., are not hosted by an external organization). In-scope and excluded IP addresses and ranges are listed below.

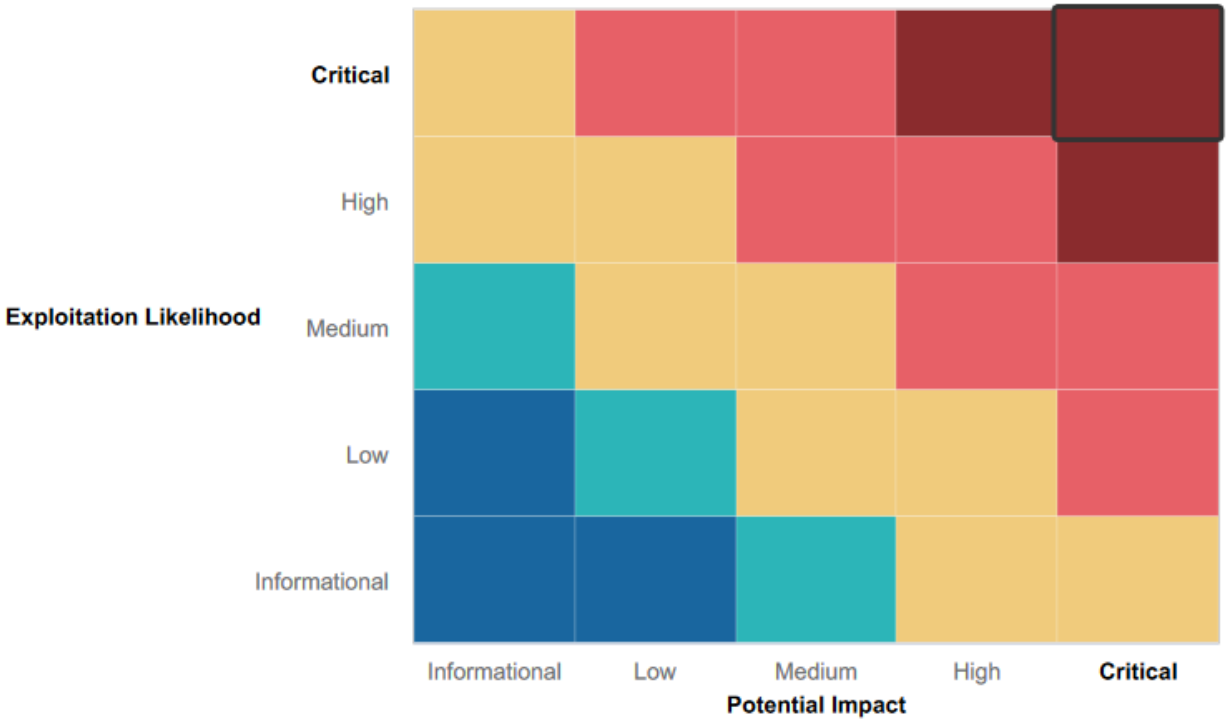
Executive Summary of Findings

Grading Methodology

Each finding was classified according to its severity, reflecting the risk each such vulnerability may pose to the business processes implemented by the application, based on the following criteria:

Critical: Immediate threat to key business processes.
High: Indirect threat to key business processes/threat to secondary business processes.
Medium: Indirect or partial threat to business processes.
Low: No direct threat exists; vulnerability may be leveraged with other vulnerabilities.
Informational: No threat; however, it is data that may be used in a future attack.

As the following grid shows, each threat is assessed in terms of both its potential impact on the business and the likelihood of exploitation:



Summary of Strengths

While the assessment team was successful in finding several vulnerabilities, the team also recognized several strengths within Rekall's environment. These positives highlight the effective countermeasures and defenses that successfully prevented, detected, or denied an attack technique or tactic from occurring.

- **User Authentication**: Authentication mechanisms were in place, reducing the risk of unauthorized access to critical systems and data.
- **Network Segmentation**: Network segmentation helped isolate some critical systems, limiting lateral movement for potential attackers and containing breaches.
- **Intrusion Detection and Prevention**: Intrusion detection and prevention systems effectively monitored network traffic within the majority of the network, providing real-time alerts and restricting unauthorized access to parts of the network.
- **Data Encryption**: The use of data encryption added a layer of protection to sensitive information, even in the event of unauthorized access.
- **Access Controls**: In several areas, strong access controls were implemented, limiting access to authorized personnel and reducing the attack surface.
- **Proactive Security Mindset**: Total Rekall has taken a proactive approach to maintaining a strong security posture through ongoing penetration testing for continuous vulnerability identification and mitigation.
- **Input Validation**: The web application demonstrates a strong use of input validation. Numerous deliberate attempts to inject SQL code into input fields were consistently thwarted, requiring over a dozen unsuccessful trials before any success was achieved. This enhances the security posture of the application significantly.

Summary of Weaknesses

We successfully found several critical vulnerabilities that should be immediately addressed in order to prevent an adversary from compromising the network. These findings are not specific to a software version but are more general and systemic vulnerabilities.

- **Inadequate Patch Management**: There is evidence of inadequate patch management, as multiple vulnerabilities were found that could have been mitigated through timely updates and patches.
- **Weak Password Policies**: Weak password policies were evident, as several vulnerabilities involved the successful guessing of passwords or exploitation of weak credentials.
- **Insufficient Access Controls**: Access controls were insufficiently enforced in some areas, allowing unauthorized users to gain access to critical systems and data.
- **Limited Intrusion Detection**: Some vulnerabilities were successfully exploited without detection, indicating gaps in intrusion detection capabilities.
- **Insufficient Training**: While there was evidence of a security-aware culture, there were instances where training gaps may have contributed to vulnerabilities, particularly the use of weak passwords.
- **Exposed Sensitive Data**: Sensitive data was found throughout the network directories, indicating a lack of data access controls and data classification.

- **Ineffective Password Hash Management**: Password hashes were not securely managed, allowing attackers to extract and crack them, leading to unauthorized access and further exploitation of Total Rekall's network..

- **Insufficient User Privilege Controls**: In some instances, users had excessive privileges. Enabling these accounts to access parts of the network that they do not need to have access to, increasing the potential impact resulting from compromised accounts.

- **Limited File Encryption**: Not all sensitive files were encrypted, or properly secured, leaving them vulnerable to unauthorized access.

- **Exposure of Credentials**: Exposure of credentials in public repositories highlighted a lack of awareness and controls regarding secure handling of sensitive information.

Executive Summary

Ultimate Pentesting, LLC, conducted a thorough evaluation of Total Rekall's network posture, accomplishing all the objectives defined within the scope of work for this engagement. Our team successfully compromised various aspects of the web application, Linux-based systems, and Windows-based systems. Furthermore, an exploit that facilitates elevated access privileges was utilized, enabling us to identify and extract any exposed sensitive data uncovered during the course of this assessment. The testing process was divided into three distinct phases. Phase one (1) concentrated on assessing the web application, while phase two (2) involved evaluating the Linux-based systems. Phase three (3) was dedicated to the examination of the Windows-based systems. Additionally, throughout this assessment, the following network scanning tools were utilized to enumerate Total Rekall’s network: Nmap, Zenmap, Nessus, and Burp Suite.

In Phase one of our assessment, we focused on the web application, identifying a range of critical security vulnerabilities that demand immediate attention. Cross-Site Scripting (XSS) vulnerabilities (Flags 1, 2, and 3) were discovered in the "Welcome," "Comments," and "VR Planner" pages, allowing for malicious script injections. These vulnerabilities were successfully exploited, compromising user data and the application's integrity. Insecure Direct Object Reference (IDOR) and Local File Inclusion (LFI) vulnerabilities (Flags 5, 6, and 7) were also found and exploited, granting unauthorized access to sensitive resources and exposing system files. Furthermore, sensitive data leakage via HTTP response headers (Flag 4) and plaintext administrator credentials (Flag 8) poses critical risks. Additionally, by simply adding "robots.txt" to the end of the website's URL, valuable information for future exploitation(s) is publicly displayed (Flag 9). A Command (PHP) Injection was discovered, allowing unauthorized access to administrative networking tools. We successfully exploited this vulnerability by manipulating inputs to access hidden data and sensitive files (Flags 11 and 13). A hidden administrator account with a weak password was found. This exposed the administrator's account through a relatively simple brute force attempt and provided us with access to the top-secret Legal Data page (Flag 12). A vulnerability related to Session Management was identified, allowing unauthorized access to restricted content by manipulating session IDs, specifically “87” in this instance (Flag 14). Lastly, a Directory Traversal vulnerability allowed unauthorized access to files and data not intended for public exposure (Flag 15).

Phase two of the assessment, which targeted the Linux systems, several critical issues were identified. In one instance, sensitive information was exposed through open-source data gathering tools and websites; specifically, Who.is, DNS Records, and Crt.sh (Flags 1 and 3). Additionally, we reviewed the network scan results, which allowed us to identify network assets and services (Flags 4, 5, and 6). Another vulnerability was found involving Apache Tomcat Remote Code Execution (RCE), providing us with unauthorized code execution and system compromise (Flag 7). We identified a vulnerability known as Shellshock, which allowed unauthorized access to a Linux system (Flags 8 and 9). We also encountered a concern related to Apache Struts RCE, discovered through a Nessus scan, which was exploited to provide unauthorized program execution and data exfiltration (Flag 10). The Drupal service in use was found to be susceptible to unauthorized access. By exploiting this vulnerability, we gained access to the target server (Flag 11). A Privilege Escalation exploit allowed us to escalate privileges, from standard user accounts to one with "root" level access (Flag 12). This unauthorized elevation of privileges can lead to full control over the system, jeopardizing data integrity, confidentiality, and system availability.

Phase three of this assessment focused on the windows systems. The first vulnerability involved the exposure of username and password hash on GitHub; which was exploited, using a password cracker called “John the Ripper”, to reveal the password "Tanya4life" (Flag 1). These credentials were used to gain unauthorized access the IP address “172.22.117.20” (Flag 2). The third vulnerability pertained to an anonymous FTP login, which we exploited to gain unauthorized access to sensitive information (Flag 3). The fourth vulnerability involved us attaining unauthorized access through the SLMail service found in a prior network scan (Flag 4). Subsequently, we utilized “Kiwi” to further exploit the compromised system. Dumping cached credentials from within the system revealed the user “ADMBob” and his hashed password, which we also cracked. Proceeded to use these credentials to launch a PsExec exploit and gain access to the Windows 2019 server. Going a step further, we launched a NTLM exploit which ultimately revealed and administrator user and its

hashed NTLM password (Flag 10). To tie things off, we performed a search, from a meterpreter session, which provided us with the location of sensitive data. In this case we exposed data within the “Documents” and “root” directories (Flags 7 and 9).

The comprehensive findings detailing the vulnerabilities can be found in the "Vulnerability Findings" section of this report. These vulnerabilities, if exploited, have the potential to inflict substantial and irreparable harm to Total Rekall's network, data, and reputation. Ultimate Pentesting, LLC has diligently presented a diverse array of remediation recommendations for each of the identified vulnerabilities. It is imperative that the critical vulnerabilities are addressed with the utmost urgency, while the remaining ones should be prioritized based on their severity and immediate need for mitigation. The suggested remediation strategies span a spectrum in terms of implementation costs and complexity, allowing for a tailored approach to enhance Total Rekall’s security measures.

Summary Vulnerability Overview

Vulnerability	Severity
Web Application: Cross Site Scripting (XSS) – Reflected & Stored	Critical
Web Application: Insecure Direct Object Reference (IDOR)	Critical
Web Application: Local File Inclusion (LFI)	Critical
Web Application: Sensitive Data Exposure via HTTP Response Headers	Medium
Web Application: Sensitive Data Exposure - Administrator Credentials	Critical
Web Application: Sensitive Data Exposure - Exclusion Standard Settings	Low
Web Application: Command (PHP) Injection	Critical
Web Application: Brute Force Attack	Critical
Web Application: Session Management	Medium
Web Application: Directory Traversal	Medium
Web Application: Open Source Data Exposure (WhoIs, Certificates, & DNS Records)	High
Linux OS: Network Scanning Susceptibility (Zenmap, Nmap, & Nessus)	High
Linux OS: Apache Tomcat Remote Code Execution (CVE-2017-12617)	Critical
Linux OS: Shellshock (Port 80)	Critical
Linux OS: Apache Struts (CVE-2017-5638) Remote Code Execution (RCE)	Critical
Linux OS: Drupal (CVE-2019-6340)	Critical
Linux OS: Privilege Escalation (CVE-2019-14287)	Critical
Web Application: Username & Password Hash Exposed (GitHub)	Critical
Windows OS: Unauthorized Network Scanning (Nmap)	High
Windows OS: Anonymous (FTP) Login	High
Windows OS: SLMail Service (POP3)	High
Windows OS: Task Scheduler	Medium

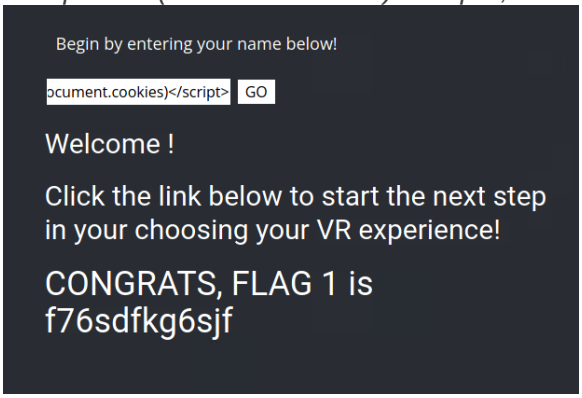
Windows OS: Exposed User/System Hashes (Kiwi)	**Critical**
Windows OS: Sensitive Data Exposure - Public Directory	**Medium**

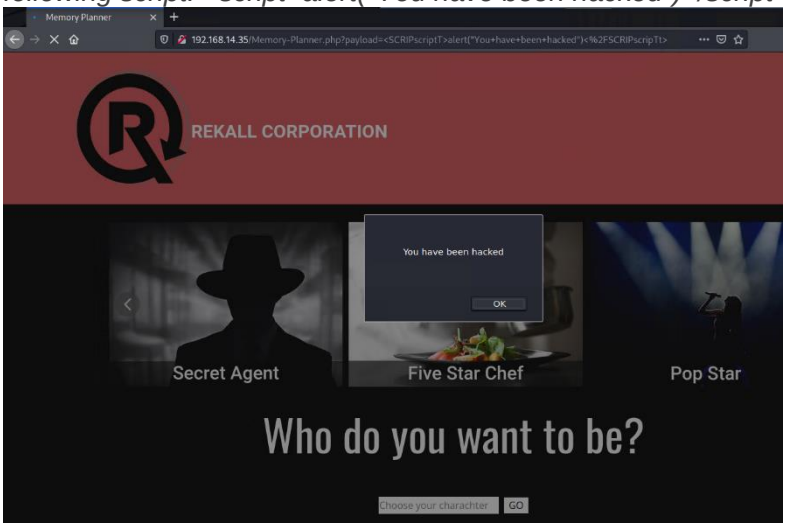
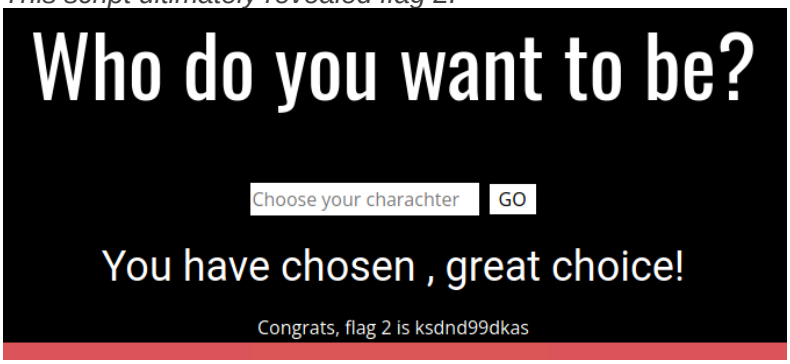
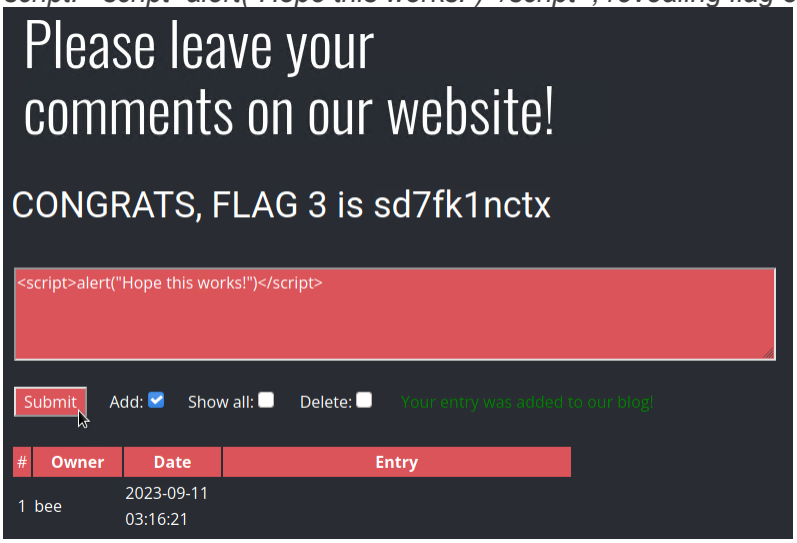
The following summary tables represent an overview of the assessment findings for this penetration test:

Scan Type	Total
Hosts	192.168.14.35 – totalrekall.xyz 15.197.148.33 – totalrekall.xyz 97.74.105.26 - ns51.domaincontrol.com 173.201.73.26 - ns52.domaincontrol.com 192.168.13.10 - Linux 192.168.13.11 - Linux 192.168.13.12 - Linux 192.168.13.13 - Linux 192.168.13.14 - Linux 192.168.13.1 – Linux 172.22.117.20 – Windows10 172.22.117.10 – Windows Domain Controller 172.22.117.100 – Windows Host
Ports	21 (FTP), 22 (SSH), 25 (SMTP), 80 (HTTP), 110 (POP3), 135 (RPC)

Exploitation Risk	Total
Critical	13
High	5
Medium	5
Low	1

Vulnerability Findings

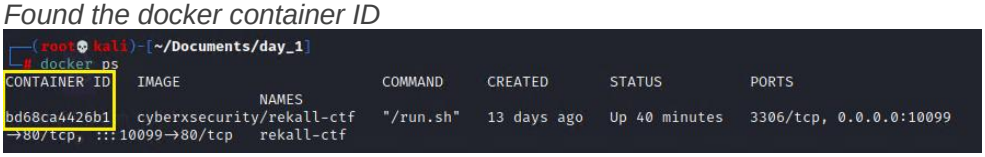
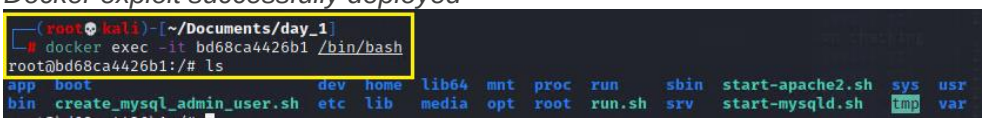
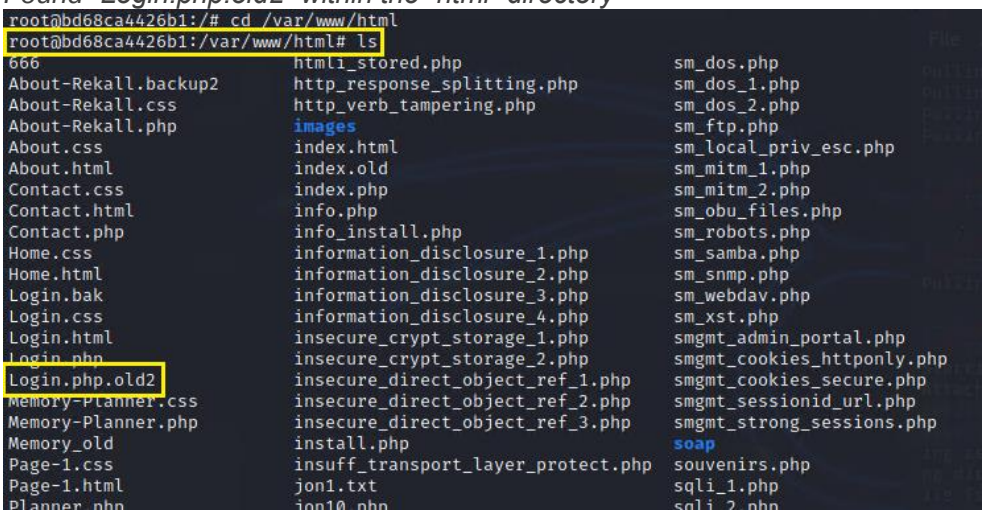
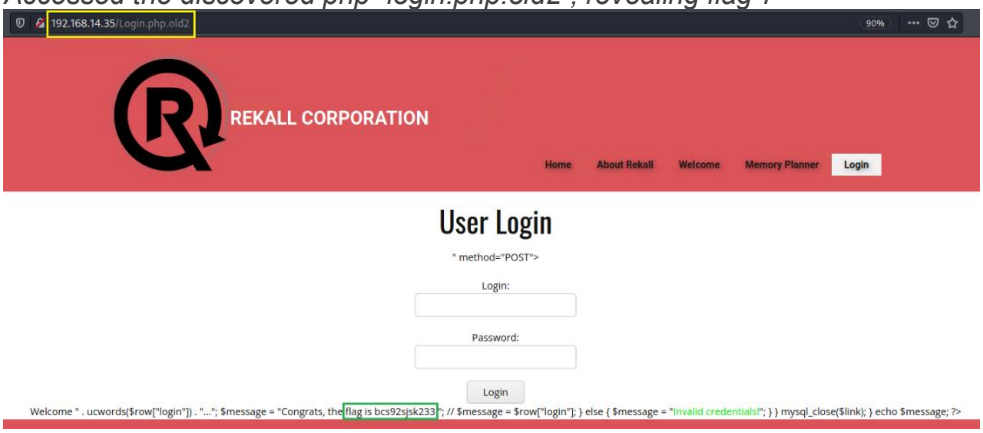
Vulnerability 1	Findings
Title	Cross Site Scripting (XSS) – Reflected & Stored
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Critical**
Description	Successfully implemented alert functionalities within the input fields designated as "Begin by entering your name below!" on the Welcome page, as well as in the "Choose Your Character" field of the Memory Planner page through the utilization of cross-site scripting (XSS) techniques. Although the Memory Planner page had certain input validation measures in place, we were able to circumvent them with minimal adjustments to our script input, employing "<SCRIPscriptT>" in place of "<script>". The ability to inject these scripts poses a significant security risk, as it could potentially enable an attacker to redirect your customers to fraudulent web pages, install keyloggers, or capture user cookies. This, in turn, would allow malicious actors to pilfer customer data and exploit it for unauthorized access to your system, potentially launching further damaging attacks.
Images	*The following script successfully reflected on the home page: <script>alert(Document.cookie)</script>, revealing flag 1.* 

	On the "Memory-Planner.php" page, we successfully implemented the following script: <script>alert("You have been hacked")</script>  *This script ultimately revealed flag 2.*  *Within the "Comments.php" page, successfully implemented the following script: <script>alert("Hope this works!")</script>, revealing flag 3.* 
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Input Validation and Sanitization:** Robust input validation and sanitization are fundamental to preventing XSS attacks. Ensure that all user inputs are validated and sanitized to reject malicious input.

	b) **Content Security Policy (CSP):** Implementing a well-defined CSP helps control which sources of content are allowed to execute, thereby mitigating the risk of unauthorized script execution. c) **Escape Output Data:** Encoding or escaping user-generated content before rendering it in web pages is a critical step to prevent browsers from interpreting it as executable code. d) **Secure Coding Practices:** Educate developers about secure coding practices, emphasizing input validation, output encoding, and avoiding direct script execution. e) **Regular Security Scans and Penetration Testing:** Conducting regular security scans and penetration testing helps identify and remediate vulnerabilities, including XSS issues. f) **Content-Security-Policy (CSP) Headers:** Properly configuring CSP headers to restrict sources and avoid allowing inline scripts is vital in preventing XSS attacks. g) **Security Headers:** Implementing security headers like HTTP Strict Transport Security (HSTS), X-Content-Type-Options, and X-XSS-Protection enhances web security. h) **Logging and Monitoring:** Setting up comprehensive logging and monitoring of web application activity enables the early detection and response to suspicious behavior, including potential XSS attacks. i) **Incident Response Plan:** Having a well-prepared incident response plan is crucial for addressing XSS incidents effectively when they occur.

Vulnerability 2	Findings
Title	Insecure Direct Object Reference (IDOR)
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Critical**
Description	In an IDOR vulnerability, an attacker can access or manipulate objects (such as files, directories, database records, or URLs) directly by changing parameters in the URL, without proper authentication or authorization. In this instance, we were able to access flag 7 at the URL "192.168.14.35/Login.php.old2" without the appropriate access rights. Initially, we identified the Docker container associated with the target web application by executing the "docker ps" command within the Kali environment. Subsequently, we successfully exploited this Docker container by employing the command "docker exec -it bd68ca4426b1 /bin/bash". Upon gaining access to the Docker container, a search for sensitive files and directories commenced, culminating in a significant discovery. Specifically, we located the file named "Login.php.old2" within the "/var/www/html" directory. To validate the existence of this identified file and assess its contents, we

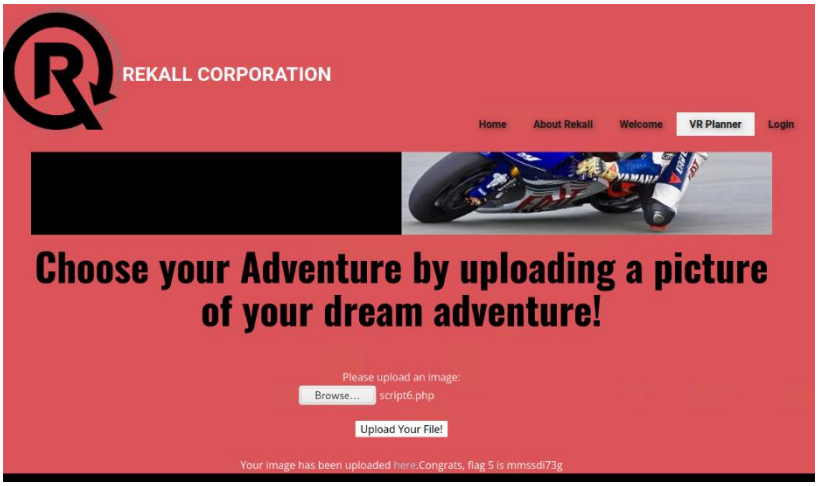
	manually adjusted the web URL to "192.168.14.35/Login.php.old2." This adjustment granted us access to the targeted web page, ultimately leading to the revelation of flag 7. These actions collectively exemplify the successful identification and exploitation of the IDOR vulnerability, underscoring the potential risks associated with unauthorized access to critical resources within the web application.

Images	*Found the docker container ID*
	 
	`# docker ps`
	`CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES`
	`bd68ca4426b1 cyberxsecurity/rekall-ctf "/run.sh" 13 days ago Up 40 minutes 3306/tcp, 0.0.0.0:10099->80/tcp, :::10099->80/tcp rekall-ctf`
	
	Docker exploit successfully deployed
	 
	`# docker exec -it bd68ca4426b1 /bin/bash`
	`root@bd68ca4426b1:/# ls`
	`app boot dev home lib64 mnt proc run sbin start-apache2.sh sys usr`
	`bin create_mysql_admin_user.sh etc lib media opt root run.sh srv start-mysqld.sh tmp var`
	
	Found "Login.php.old2" within the "html" directory
	 
	`root@bd68ca4426b1:/# cd /var/www/html`
	`root@bd68ca4426b1:/var/www/html# ls`
	`666 html1_stored.php sm_dos.php`
	`About-Rekall.backup2 http_response_splitting.php sm_dos_1.php`
	`About-Rekall.css http_verb_tampering.php sm_dos_2.php`
	`About-Rekall.php images sm_ftp.php`
	`About.css index.html sm_local_priv_esc.php`
	`About.html index.old sm_mitm_1.php`
	`Contact.css index.php sm_mitm_2.php`
	`Contact.html info.php sm_obu_files.php`
	`Contact.php info_install.php sm_robots.php`
	`Home.css information_disclosure_1.php sm_samba.php`
	`Home.html information_disclosure_2.php sm_snmp.php`
	`Login.bak information_disclosure_3.php sm_webdav.php`
	`Login.css information_disclosure_4.php sm_xst.php`
	`Login.html insecure_crypt_storage_1.php smgmt_admin_portal.php`
	`Login.php insecure_crypt_storage_2.php smgmt_cookies_httponly.php`
	`Login.php.old2 insecure_direct_object_ref_1.php smgmt_cookies_secure.php`
	`Memory-Planner.css insecure_direct_object_ref_2.php smgmt_sessionid_url.php`
	`Memory-Planner.php insecure_direct_object_ref_3.php smgmt_strong_sessions.php`
	`Memory_old install.php soap`
	`Page-1.css insuff_transport_layer_protect.php souvenirs.php`
	`Page-1.html jon1.txt sqli_1.php`
	`Planner.php jon10.php sqli_2.php`
	
	Accessed the discovered php "login.php.old2", revealing flag 7
	 
	192.168.14.35/Login.php.old2
	REKALL CORPORATION
	Home About Rekall Welcome Memory Planner Login
	User Login
	" method="POST">
	Login:
	Password:
	Login
	Welcome " . ucwords($row["login"]) . "..."; $message = "Congrats, the flag is bcs92sjsk233"; // $message = $row["login"]; } else { $message = "invalid credentials!"; } } mysql_close($link); } echo $message; ?>

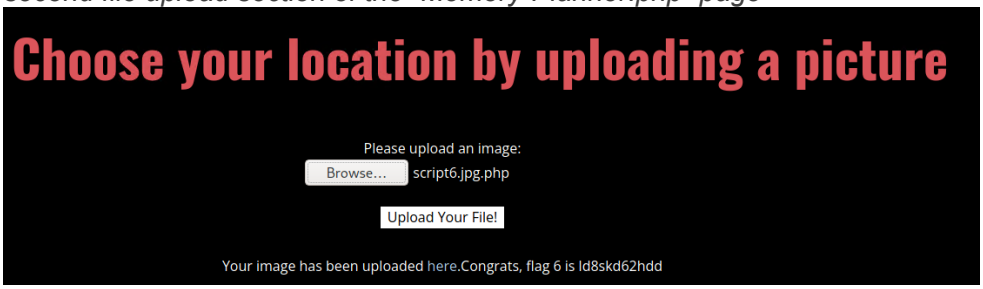
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Implement Proper Access Controls:** Strong access controls and proper authorization mechanisms are fundamental to preventing unauthorized access. Ensure that users can only access resources for which they have explicit permissions. b) **Use GUIDs or Unique Identifiers:** Employ globally unique identifiers

	(GUIDs) or random tokens for object references. This helps make it extremely difficult for attackers to guess or manipulate resource references. c) **Validate User Inputs:** Robust input validation is critical to ensure that user-supplied data is free from malicious or unauthorized content. Reject any input that does not meet defined validation criteria. d) **Secure URL Design:** Avoid exposing sensitive information or internal object references in URLs. Use indirect references or mapping mechanisms to abstract direct object references from URLs. e) **Audit and Monitoring:** Implement comprehensive logging and monitoring of user activities and access attempts. Regularly review logs to quickly detect and respond to unauthorized access. f) **Error Handling:** Customize error messages to avoid revealing sensitive system information or object references. Provide generic error messages to users to prevent information leakage. g) **Security Awareness Training:** Train developers, administrators, and users on the risks associated with IDOR vulnerabilities and the best practices for secure coding and resource access. h) **Regular Code Review:** Conduct code reviews to identify and rectify coding issues related to access controls and object references that may lead to IDOR vulnerabilities.

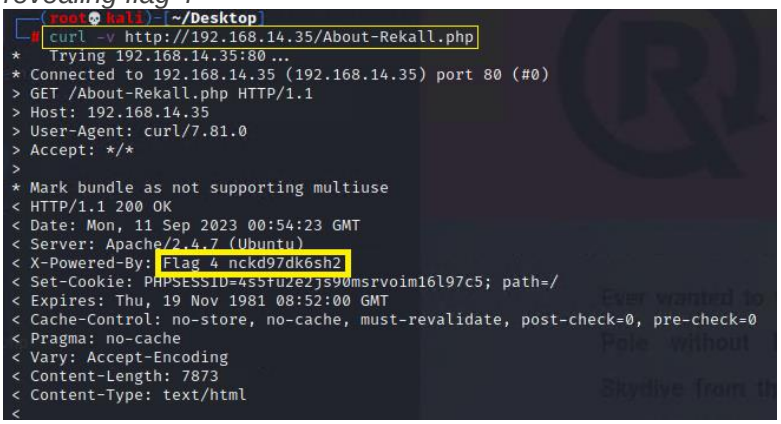
Vulnerability 3	Findings
Title	Local File Inclusion (LFI)
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Critical**
Description	We have successfully accomplished the creation of .php script files and their subsequent upload into the file upload section of the Memory Planner page. It is essential to recognize that .php pages inherently trigger a request for script execution on the backend server upon access. Permitting the uploading of .php script files introduces a significant security risk. This risk entails the potential execution of malicious scripts against the database, which could lead to unauthorized data modifications, deletions, and, in severe cases, system outages.
Images	*Flag #5 revealed through successful exploitation, “script.php”, on the first file upload section of the “Memory-Planner.php” page*



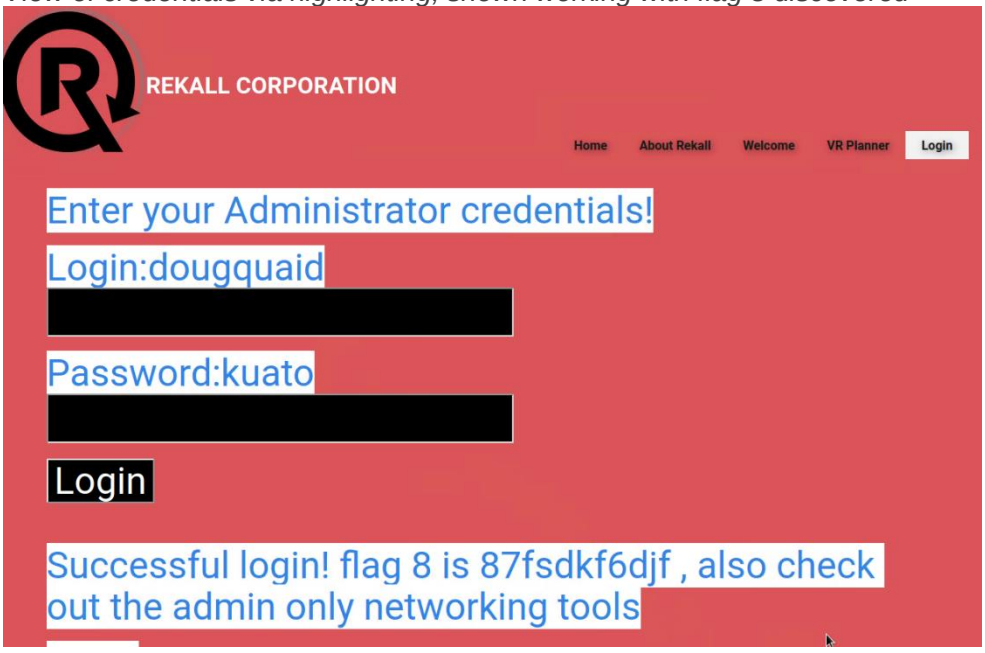
Flag #6 revealed through successful exploitation, "script.jpg.php", on the second file upload section of the "Memory-Planner.php" page



Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **File Upload Validation:** Implement rigorous validation and filtering of file uploads to only allow specific safe file types. b) **File Whitelisting:** Maintain a whitelist of allowed file extensions and restrict uploads to those extensions, denying all others. c) **File Storage Isolation:** Store uploaded files in a directory separate from the web root to prevent direct internet access. d) **Input Validation and Sanitization:** Ensure that all user inputs, including file names and paths, are rigorously validated and sanitized to prevent directory traversal attacks. e) **Access Controls:** Implement strong access controls to restrict access to sensitive parts of the application, reducing the risk of unauthorized local file access. f) **Web Application Firewall (WAF):** Deploy a Web Application Firewall with LFI detection capabilities to filter and block malicious requests. g) **Code Review:** Regularly review the application's codebase to identify and rectify potential LFI vulnerabilities stemming from improperly sanitized user inputs. h) **Logging and Monitoring:** Establish comprehensive logging and monitoring to detect and alert on suspicious file access or LFI attempts in real-time.

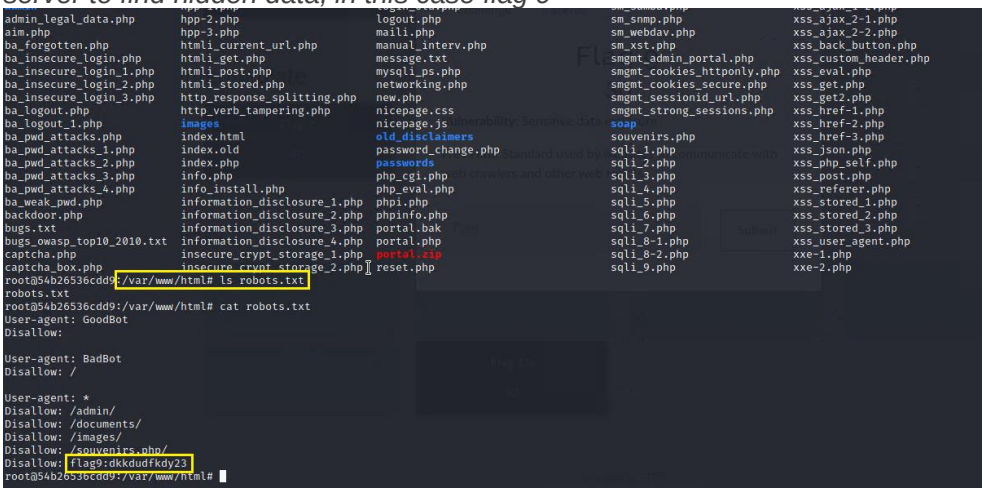
Vulnerability 4	Findings
Title	Sensitive Data Exposure via HTTP Response Headers
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Medium**
Description	To exploit this vulnerability, an adversary can employ specialized tools like BURP or initiate a cURL request directed at the "About-Rekall.php" page. In this particular instance, the cURL command, *"curl -v http://192.168.14.35/About-Rekall.php"* was employed. Subsequently, sensitive information was divulged within the HTTP response headers, leading to the revelation of flag 4. It is worth noting that the exploitation of this vulnerability does not demand an extensive technical skillset, but it does mandate the utilization of distinct tools or scripts designed for the purpose of accessing and capturing the exposed data. While this does involve sensitive data exposure, it is not immediately accessible through standard web browsing and necessitates additional tools or methods for exploitation. Nevertheless, the impact and potential consequences of this exposure should be assessed.
Images	*Results of the curl command "curl -v http://192.168.14.35/About-Rekall.php", revealing flag 4* 
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Content Redaction:** Immediately remove sensitive information from HTTP response headers to prevent unintentional exposure. b) **Header Configuration:** Review and configure HTTP response headers to minimize the disclosure of sensitive data. c) **Security Headers:** Implement security headers, such as Content Security Policy (CSP), to control content execution within web pages. d) **Response Filtering:** Implement content filtering and validation to ensure that sensitive data is not included in response headers. e) **Data Classification:** Classify and categorize data within the application to identify and protect sensitive data elements appropriately.

	f) **Access Controls:** Strengthen access controls to restrict unauthorized access to resources containing sensitive data. g) **Monitoring and Intrusion Detection:** Set up monitoring and intrusion detection to detect and alert on unusual or unauthorized access to sensitive data. h) **Incident Response Plan:** Develop and test an incident response plan specific to sensitive data exposure incidents, outlining the steps to take in case of an exposure event.

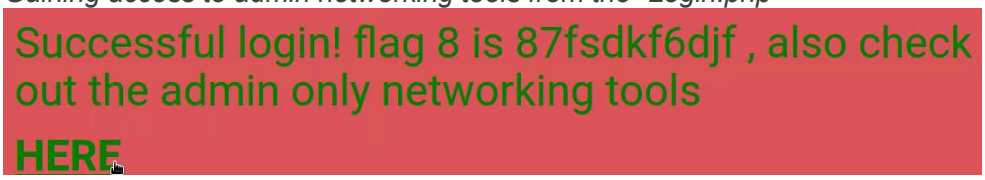
Vulnerability 5	Findings
Title	Sensitive Data Exposure – Administrator Credentials
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Critical**
Description	The identified vulnerability pertains to the exposure of sensitive data, namely the administrator's login credentials encompassing both username and password being displayed in plain-text view. Within the HTML code of the Login.php page or by simply highlighting the webpage content, access to these administrator credentials is readily attainable. This exposure poses a notable security risk, potentially resulting in unauthorized access to user accounts, data breaches, and other consequential security incidents. In this specific instance, the exposure of these credentials led to the successful access of flag 8. As well as access to Total Rekall's "admin only networking tools".
Images	*View of credentials via highlighting, shown working with flag 8 discovered*  *View of credentials via page source code (HTML)*

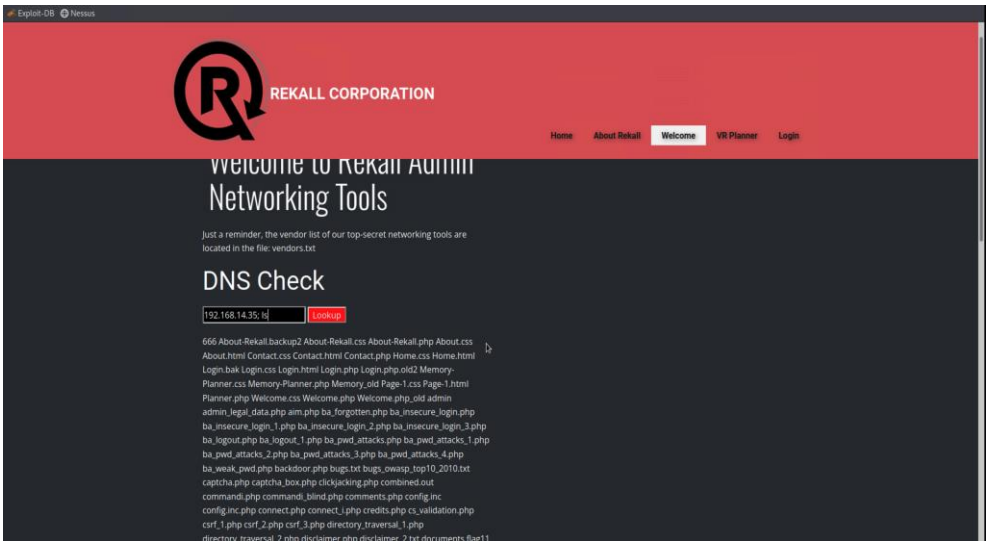
	
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Credential Encryption:** Prioritize encrypting administrator credentials to protect them from unauthorized access, even if exposed in source code or HTML. b) **Remove Hardcoded Credentials:** Ensure that no administrator login credentials are hardcoded in the application's source code. c) **Use Environment Variables:** Securely store sensitive data like credentials in environment variables or configuration files outside the web root to prevent exposure through source code or HTML. d) **Secure Coding Practices:** Educate developers about secure coding practices, emphasizing the importance of not embedding sensitive information directly in code. e) **Access Controls:** Implement robust access controls and authentication mechanisms to prevent unauthorized access to sensitive data and functions. f) **Web Application Firewall (WAF):** Deploy a WAF with content inspection capabilities to detect and block sensitive data exposure attempts. g) **Regular Security Testing:** Conduct routine security assessments, including vulnerability scanning and penetration testing, to identify and address sensitive data exposure vulnerabilities. h) **Code Review:** Regularly review the application's source code to identify and rectify potential sensitive data exposure issues, such as hardcoded credentials.

Vulnerability 6	Findings
Title	Sensitive Data Exposure - Exclusion Standard Settings
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	Low
Description	We successfully accessed the website's "robots.txt" file by appending it to the

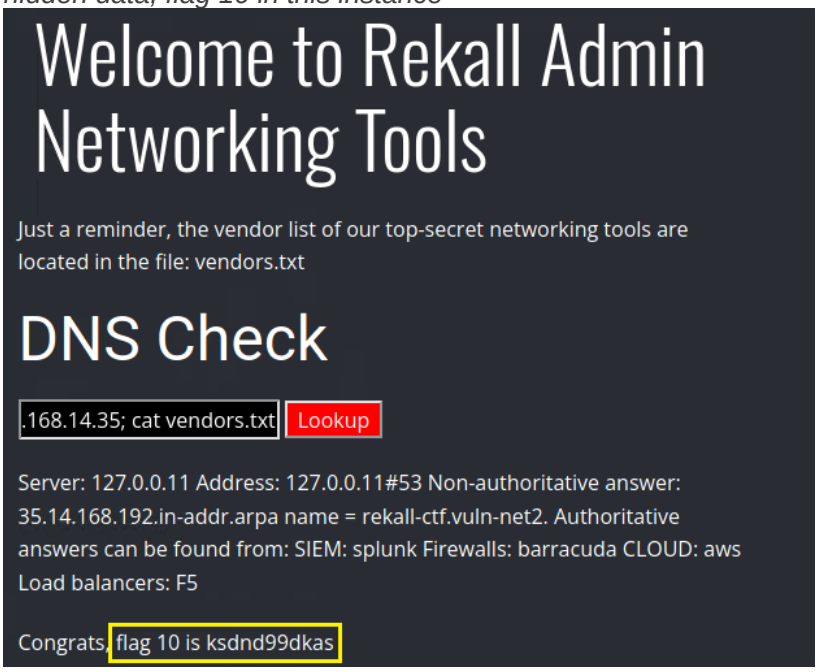
	end of the IP address. This file contains the robots exclusion standard settings for the website. Malicious actors could potentially utilize the "disallow" directives within the file as a guide to target areas for exploitation. To gain access to the Rekall web server (root), we executed the command "docker exec -it bd68ca4426b1 /bin/bash" via the Kali terminal. Following this, we navigated to the "html" directory, which can be accessed using the command "cd /var/www/html" and proceeded to list all files with "ls". These results yielded the “robots.txt” file. The "robots.txt" is a standard file used by websites to communicate with web crawlers and other automated agents. By executing the "cat robots.txt" command, we retrieved valuable information for future exploitations, including flag 9.
Images	*Changed the web address to view the “robots.txt”, revealing flag 9*  User-agent: GoodBot Disallow: User-agent: BadBot Disallow: / User-agent: * Disallow: /admin/ Disallow: /documents/ Disallow: /images/ Disallow: /souvenirs.php/ Disallow: flag9:dkkdudfkdy23 *(alternate method found later) Used root level access to the Rekall Corp.’s web server to find hidden data, in this case flag 9* 
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Secure Robots.txt Configuration:** Review and revise the "robots.txt" file to ensure it only exposes necessary information while concealing sensitive directories or areas that shouldn't be crawled by web robots. b) **Restrict Access to Robots.txt:** Restrict direct access to the "robots.txt" file from unauthorized users by configuring web server access controls or using authentication mechanisms. c) **Use Authentication for Web Server Access:** Implement authentication mechanisms for accessing sensitive areas of the web server to prevent unauthorized access, especially to administrative directories.

	d) **Regular Security Testing:** Conduct routine security assessments, including vulnerability scanning and penetration testing, to identify and address vulnerabilities like sensitive data exposure. e) **Access Controls:** Implement strong access controls to prevent unauthorized access to sensitive areas and files on the web server. f) **Logging and Monitoring:** Set up logging and monitoring systems to detect and alert on any unauthorized access or changes to the "robots.txt" file. g) **User Education:** Educate personnel about the risks associated with exposing sensitive information in the robots.txt file and the importance of properly configuring it. h) **Security Policies and Procedures:** Establish and document clear security policies and procedures regarding the handling of sensitive configuration files like "robots.txt".

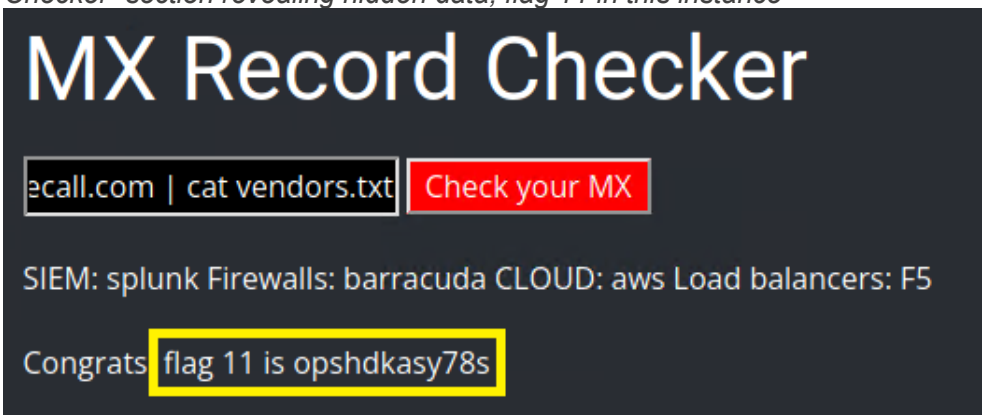
Vulnerability 7	Findings
Title	Command (PHP) Injection
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Critical**
Description	This vulnerability allows unauthorized access to administrative networking tools from the "Login.php" page, potentially leading to significant security breaches and data exposure. The vulnerability arises from a lack of input validation and proper security measures, enabling malicious actors to execute arbitrary PHP commands through manipulated inputs.
Images	*Gaining access to admin networking tools from the "Login.php"*  *Input "192.168.14.35; ls" into the DNS Check submission – Displays the contents of the "Welcome.php" page*



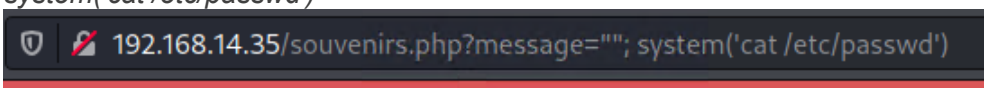
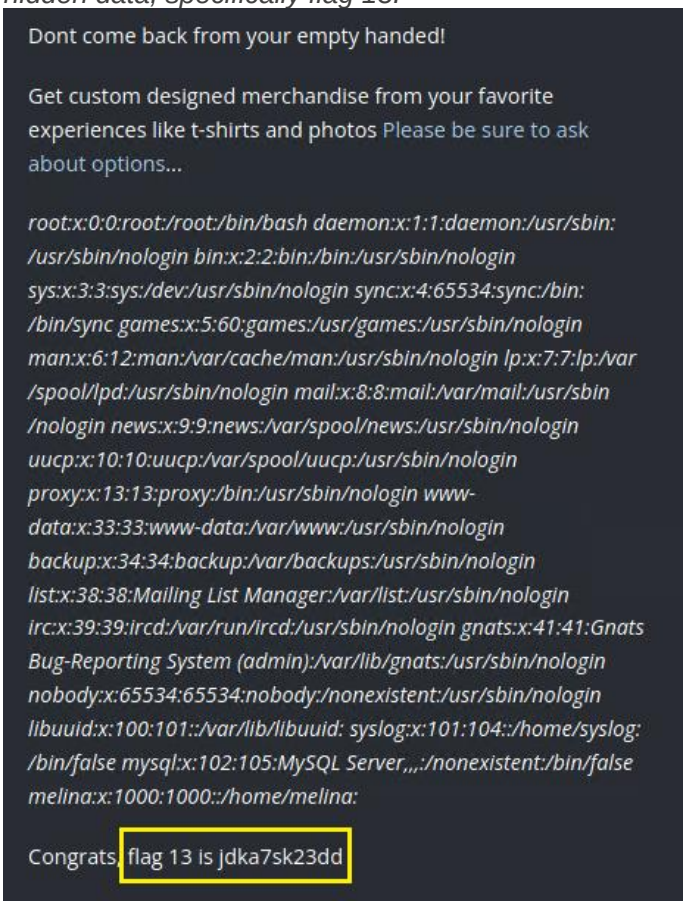
Input "192.168.14.35; cat vendors.txt" into the "DNS Check" section revealing hidden data, flag 10 in this instance



Input "www.welcometorecall.com | cat vendors.txt" into the "MX Record Checker" section revealing hidden data, flag 11 in this instance

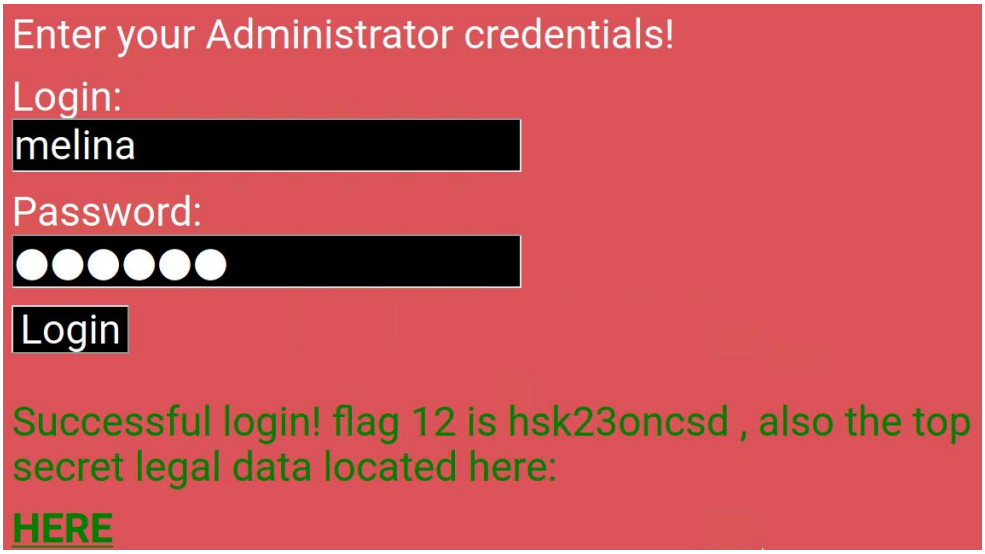


Altered the page's URL to "http://192.168.13.35/souvenirs.php?message="";

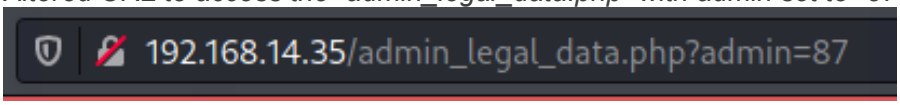
	system('cat /etc/passwd')"  *This caused the webpage to dump to contents of "/etc/passwd", revealing hidden data, specifically flag 13.* 
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Input Validation and Sanitization:** Implement robust input validation and sanitization mechanisms to filter out malicious input and prevent unauthorized command execution. b) **Use of Web Application Firewall (WAF):** Deploy a WAF with command injection detection capabilities to filter and block suspicious requests that may exploit this vulnerability. c) **Security Code Review:** Conduct a comprehensive code review of the affected application to identify and rectify insecure coding practices that lead to command injection vulnerabilities. d) **Least Privilege Principle:** Restrict access permissions and privileges to administrative networking tools to authorized personnel only. e) **Whitelist Allowed Commands:** Create a whitelist of allowed commands, and only permit those commands to be executed, thus preventing arbitrary command execution. f) **Error Handling and Logging:** Implement proper error handling and logging mechanisms to monitor and alert on any suspicious activities or

	unauthorized access attempts. g) **User Education:** Train application users and administrators about the risks associated with input manipulation and the importance of secure coding practices. h) **Regular Security Testing:** Conduct routine security assessments, including penetration testing and code scanning, to identify and remediate vulnerabilities proactively.

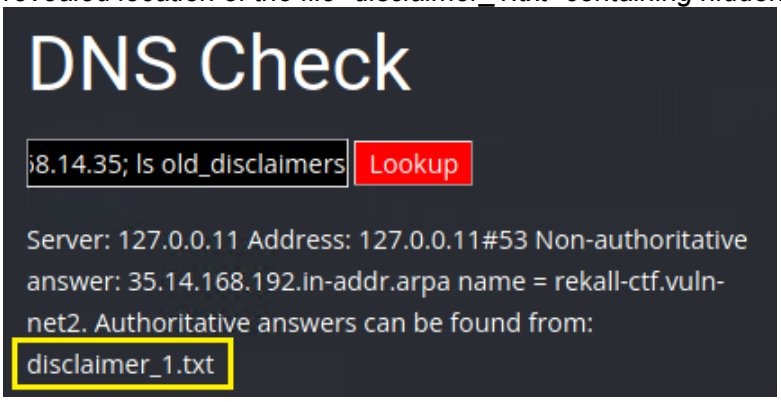
Vulnerability 8	Findings
Title	Brute Force Attack
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Critical**
Description	The vulnerability involves the discovery of a hidden administrator user named "Melina" through a brute force attack on the web application. During the attack, we injected "www.welcometorecall.com | cat /etc/passwd" on the “networking.php” page, which led to the exposure of the administrator user “melina”. The vulnerability is further exacerbated by the fact that the administrator "Melina" used a weak password; which was discovered, after only three (3) attempts, to be her own name, allowing unauthorized access to the administrator login, revealing flag 12.
Images	*Hidden administrator user “Melina” found after injecting “www.welcometorecall.com | cat /etc/passwd”*  *The administrator “Melina” was found to have utilized a weak password, with “melina” her name being used to access hidden data, in this case flag 12.*

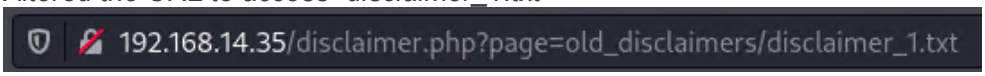
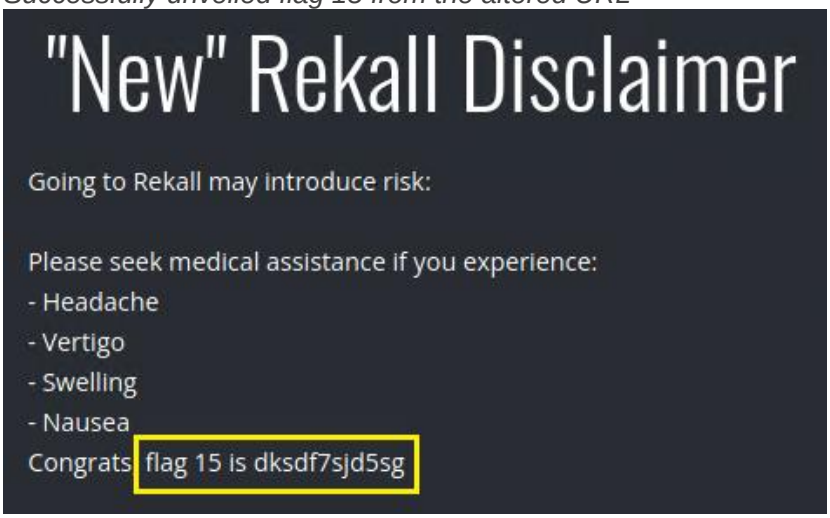
	
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Enforce Strong Password Policies:** Implement and enforce strong password policies for all users, requiring complex passwords that include a combination of uppercase and lowercase letters, numbers, and special characters. b) **Account Lockout Mechanism:** Implement an account lockout mechanism that temporarily locks out user accounts after a certain number of failed login attempts, preventing brute force attacks. c) **Multi-Factor Authentication (MFA):** Require the use of multi-factor authentication (MFA) for all user accounts, especially for administrators, to add an extra layer of security beyond passwords. d) **Security Awareness Training:** Educate administrators and users about the importance of strong passwords, password hygiene, and the risks associated with weak passwords. e) **Regular Password Changes:** Enforce regular password changes for all user accounts to reduce the risk of compromised credentials. f) **IP Whitelisting and Rate Limiting:** Implement IP whitelisting to restrict access to administrative interfaces only from trusted IP addresses. Additionally, enforce rate limiting to throttle login attempts from any source. g) **Implement Intrusion Detection Systems (IDS):** Deploy IDS solutions that can detect and alert on suspicious login attempts and brute force attacks. h) **Monitoring and Logging:** Set up monitoring and logging to track and alert on multiple failed login attempts, allowing for proactive identification of potential attacks.

Vulnerability 9	Findings

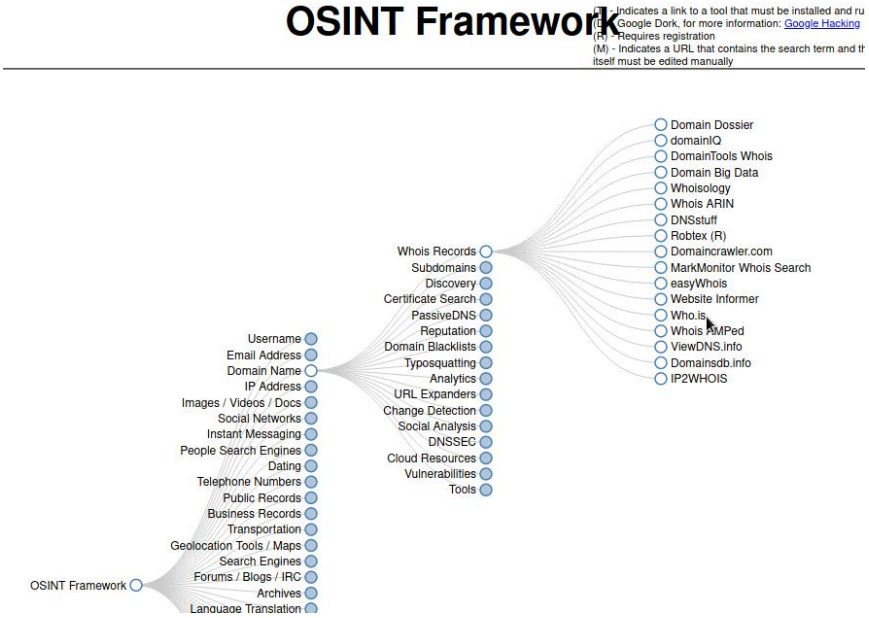
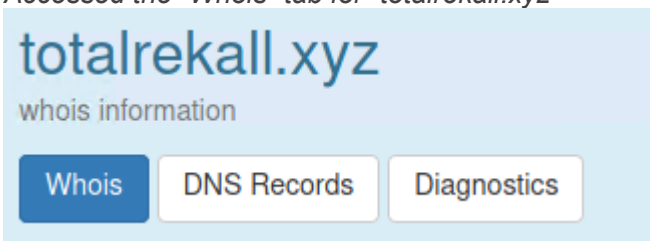
Title	Session Management
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Medium**
Description	This vulnerability in the web application pertains to session management. During testing with Burp Intruder, it was discovered that a specific session ID, namely "87," grants access to sensitive data. By altering the URL to "http://192.168.13.35/admin_legal_data.php?admin=87", we managed to gain access to the "Legal Documents – Restricted Area" page, revealing hidden data, specifically flag 14. This indicates a potential weakness in the session management mechanism, allowing unauthorized access to restricted areas. The vulnerability allows unauthorized access to restricted content. While it doesn't provide immediate access to critical or sensitive data, it exposes hidden information, which can have moderate implications. Exploiting this vulnerability requires knowledge of the specific session ID (87) and the ability to manipulate the URL, making it less severe than other vulnerabilities.
Images	*Altered URL to access the "admin_legal_data.php" with admin set to "87"*  *This granted access to the "Legal Documents – Restricted Area" page, revealing the hidden flag 14.* 
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Session Management Best Practices:** Implement robust session management practices, including secure session ID generation and storage, to prevent unauthorized access to sensitive areas. b) **Session Expiry and Timeout:** Configure sessions to expire after a reasonable period of inactivity, reducing the window of opportunity for attackers to exploit session IDs. c) **Randomized Session IDs:** Ensure that session IDs are sufficiently random and not predictable, making it difficult for attackers to guess

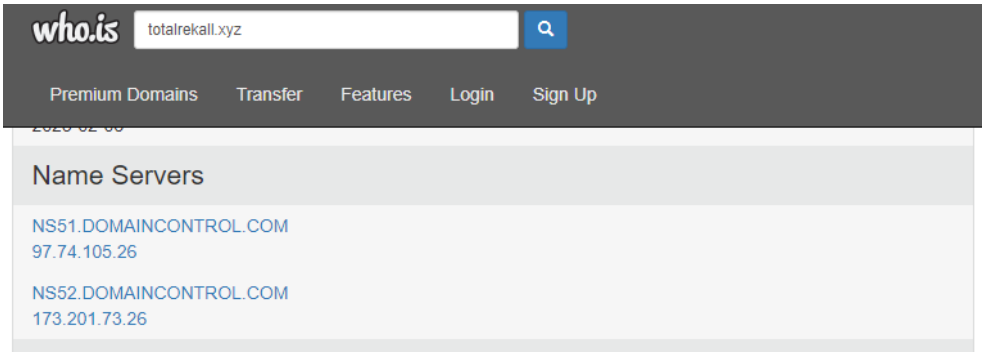
	valid IDs. d) **Access Controls:** Implement proper access controls to restrict user access to sensitive areas based on roles and permissions. e) **Error Handling and Logging:** Implement proper error handling and logging to monitor and alert on any suspicious access attempts, especially those involving manipulated session IDs. f) **Regular Security Testing:** Conduct routine security assessments, including session management testing, to identify and remediate vulnerabilities proactively. g) **User and Administrator Training:** Educate administrators and users about secure session management practices and the importance of not sharing session IDs. h) **Session Token Rotation:** Implement session token rotation to invalidate and replace session IDs periodically, reducing the risk of session fixation attacks.

Vulnerability 10	Findings
Title	Directory Traversal
Type (Web app / Linux OS / WIndows OS)	Web application
Risk Rating	**Medium**
Description	During testing, we successfully injected "192.168.14.35; ls old_disclaimers" into the "DNS Check" submission, which revealed the location of a file containing hidden data. We then altered the URL to access "disclaimer_1.txt" and successfully unveiled flag 15 from the altered URL. This vulnerability allows unauthorized access to files and data not intended for public exposure. While it can expose hidden data, it requires specific knowledge and manipulation of inputs, making it less severe than other vulnerabilities. Exploiting this can result in data exposure, but it may not immediately lead to critical security breaches.
Images	*Input "192.168.14.35; ls old_disclaimers" into the "DNS Check" submission, revealed location of the file "disclaimer_1.txt" containing hidden data* 

	Altered the URL to access "disclaimer_1.txt"  *Successfully unveiled flag 15 from the altered URL* 
Affected Hosts	192.168.14.35 – totalrekall.xyz
Remediation	a) **Input Validation and Sanitization:** Implement strict input validation and sanitization to prevent malicious input that may lead to directory traversal attacks. b) **Access Controls:** Enforce proper access controls to restrict access to files and directories based on user roles and permissions. c) **File Whitelisting:** Create a whitelist of allowed files and directories, and only permit access to those on the whitelist. d) **Path Normalization:** Implement path normalization to ensure that user-supplied input cannot be manipulated to traverse directories. e) **Content Security Policy (CSP):** Implement CSP headers to control which scripts can be executed, limiting the risk of directory traversal attacks. f) **Server Hardening:** Harden the web server configuration to minimize the attack surface and restrict file system access. g) **Security Awareness Training:** Train administrators and developers about the risks associated with directory traversal vulnerabilities and the importance of input validation. h) **Regular Security Testing:** Conduct routine security assessments, including vulnerability scanning and penetration testing, to identify and remediate directory traversal vulnerabilities proactively.

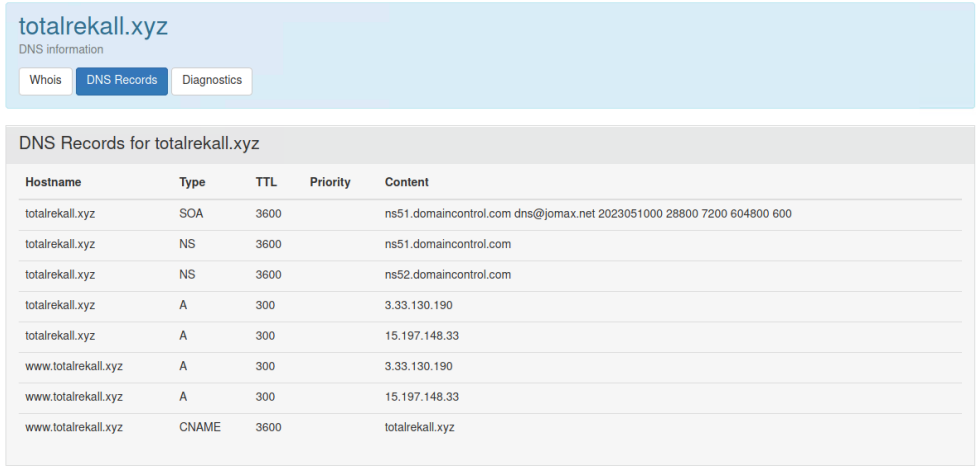
Vulnerability 11	Findings
Title	Open Source Data Exposure (WhoIs, Certificates, & DNS Records)

Type (Web app / Linux OS / WIndows OS)	Web Application
Risk Rating	**High**
Description	This involves the exposure of sensitive information through open-source data gathering tools and websites. Tools were utilized from the "dossier" Open Source Intelligence (OSINT) framework to access various methods for information gathering. Specifically, the "Whois" and "DNS Records" tabs were accessed for the domain "totalrekall.xyz", which revealed IP addresses, flag 1 and 2, along with a hint for accessing servers “sshUser alice”. Additionally, the "crt.sh" utility was used to inspect specific certificates and ultimately reveal flag 3. The exposure of IP addresses and sensitive data, including the identified flags, can potentially lead to unauthorized access to systems and data. The presence of plaintext hints for connecting to servers indicates a potential pathway for exploitation. Open-source data gathering tools can easily be employed, since they are readily available and widely used for reconnaissance activities.
Images	*Used the dossier (open source) tool [https://osintframework.com/] to access the “who.is” utilitiy*  *Accessed the “Whois” tab for “totalrekall.xyz”*  *This revealed the IP addresses for the domain controllers*



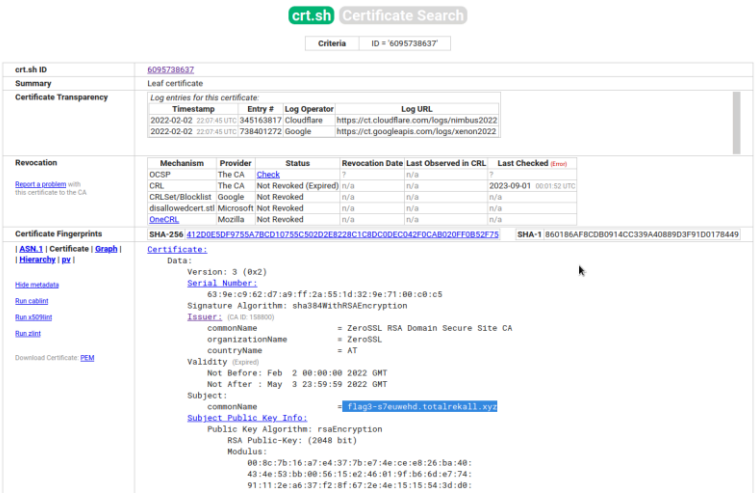
Scrolling down this page further revealed flag 1, as well as a plain text hint for connecting to the servers "sshUser alice" under the "Registrar Data" section

Accessed the "DNS Records" tab for "totalrekall.xyz", displaying the IP address "15.197.148.33" which was then used to reveal flag 2



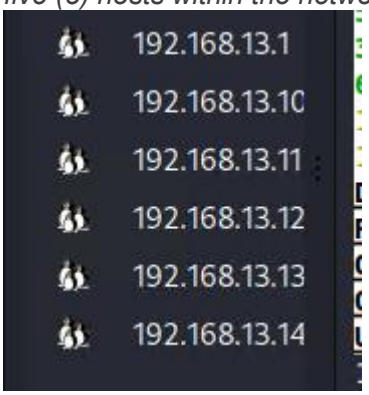
Returned to the dossier (open source) tool [https://osintframework.com/] to access the "crt.sh" utility, revealing a list of stored certificates

Inspected the contents of the crt.sh ID "6095738637", leading to the discovery of flag 3



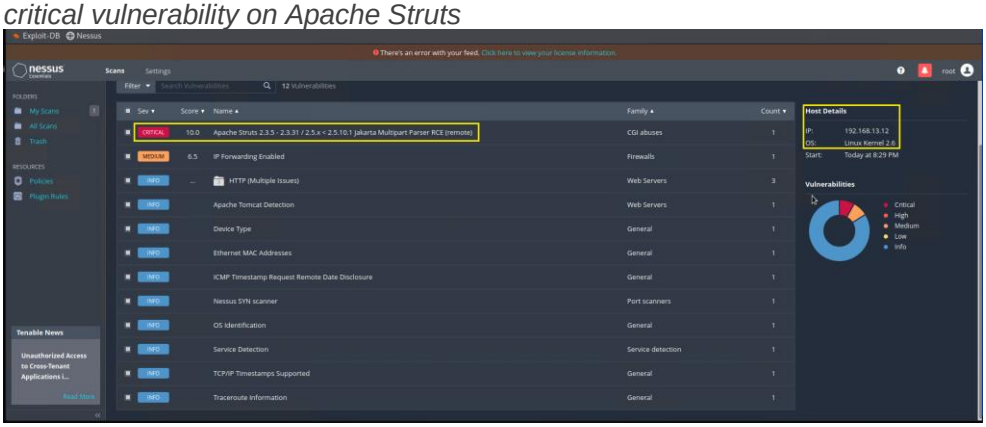
Affected Hosts	15.197.148.33 – totalrekall.xyz
Remediation	a) **Domain Privacy Protection:** Implement domain privacy protection services to hide sensitive information from public Whois databases, such as registrant names and contact details. This is essential to prevent the exposure of critical domain ownership information. b) **Remove Unnecessary Flags:** Identify and remove any exposed flags or sensitive information from public records, configurations, or websites to minimize the risk of unauthorized access. This is crucial for reducing potential attack vectors. c) **Review DNS Records:** Regularly review and audit DNS records to ensure that only essential information is exposed. Remove any unnecessary records that may disclose sensitive details. This helps in controlling the information available to potential attackers d) **Credential Hardening:** Ensure that plaintext hints or credentials are not exposed in public records or configuration files. Replace them with secure access mechanisms and enforce strong authentication. This is vital for preventing unauthorized access. e) **Monitor Certificate Transparency:** Continuously monitor Certificate Transparency logs to detect any unauthorized or unexpected certificates issued for your domain. Take prompt action to investigate and mitigate potential security incidents. This helps in identifying suspicious certificate activity. f) **Security Awareness Training:** Provide security awareness training to employees and staff members to educate them on the risks associated with open-source data exposure and the importance of safeguarding sensitive information. This is essential for creating a security-conscious workforce. g) **Access Control:** Implement access controls and authentication mechanisms to restrict access to sensitive systems and data. Use secure methods, such as SSH key authentication, to enhance server access security. Proper access control is crucial for limiting exposure. h) **Regular Scanning and Monitoring:** Conduct regular scans and security assessments of your web applications and infrastructure to identify vulnerabilities and exposed information. Employ intrusion

	detection systems (IDS) to detect suspicious activities. Ongoing monitoring is vital for proactive threat detection.

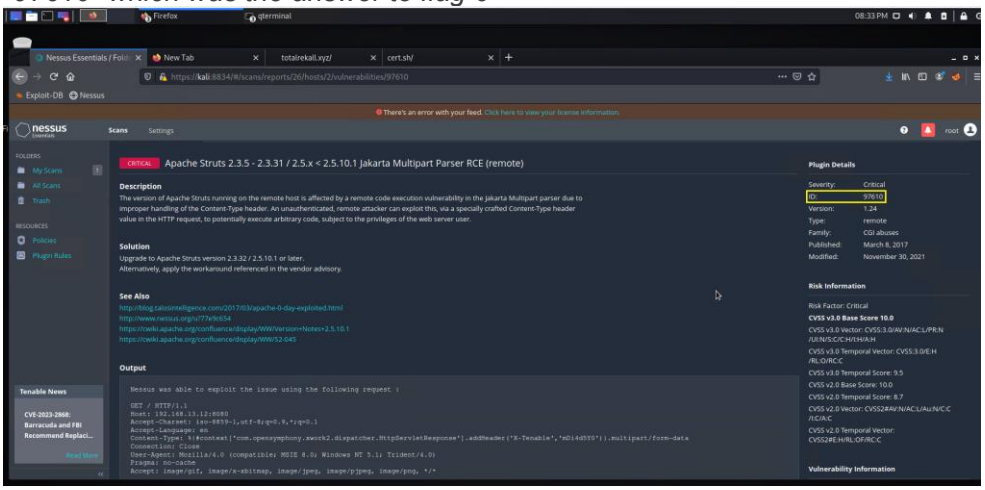
Vulnerability 12	Findings
Title	Network Scanning Susceptibility (Zenmap, Nmap, & Nessus)
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	**High**
Description	A Zenmap scan of the network (192.168.13.0/24) was conducted, revealing the presence of five hosts within the network. Flag 4 was derived from the discovery that there are five hosts, excluding the scanning host. By running an aggressive Nmap scan (-A) on the same network, it was determined that one of the hosts, specifically 192.168.13.13, was running the Drupal content management system. Flag 5 was obtained by identifying the host running Drupal. A Nessus scan was performed, targeting "192.168.13.12," which identified a critical vulnerability in Apache Struts. This vulnerability was associated with ID "97610", which was also the answer to flag 6. Unauthorized scanning can lead to the identification of network assets and services, which could be leveraged by malicious actors to launch targeted attacks. The presence of Drupal and Apache Struts on the hosts adds to the risk, as both vulnerabilities could be exploited.
Images	*Results of Zenmap (Intense Scan) on Network "192.168.13.0/24", revealing five (5) hosts within the network. "5" was also the answer to completing flag 4*  *Successful aggressive Nmap scan revealing the IP "192.168.13.13" is running "Drupal". This IP address was the answer to completing flag 6*

```
root@kali: ~
File  Actions  Edit  View  Help
PORT     STATE SERVICE VERSION
8080/tcp open  http    Apache Tomcat/Coyote JSP engine 1.1
|_http-server-header: Apache-Coyote/1.1
|_http-title: Site doesn't have a title (text/html;charset=UTF-8).
| http-methods:
|_  Potentially risky methods: PUT DELETE TRACE PATCH
|_http-favicon: Spring Java Framework
|_http-open-proxy: Proxy might be redirecting requests
MAC Address: 02:42:C0:A8:0D:0C (Unknown)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   0.01 ms 192.168.13.12

Nmap scan report for 192.168.13.13
Host is up (0.000012s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.25 ((Debian))
|_http-server-header: Apache/2.4.25 (Debian)
|_http-title: Hom | Drupal CVE-2019-6340
|_http-generator: Drupal 8 (https://www.drupal.org)
| http-robots.txt: 22 disallowed entries (15 shown)
| /core/ /profiles/ /README.txt /web.config /admin/
| /comment/reply/ /filter/tips /node/add/ /search/ /user/register/
| /user/password/ /user/login/ /user/logout/ /index.php/admin/
|_/index.php/comment/reply/
MAC Address: 02:42:C0:A8:0D:0D (Unknown)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop

TRACEROUTE
HOP RTT     ADDRESS
1   0.01 ms 192.168.13.13
```

Results of Nessus scan targeting the IP address “192.168.13.12”, revealing a critical vulnerability on Apache Struts



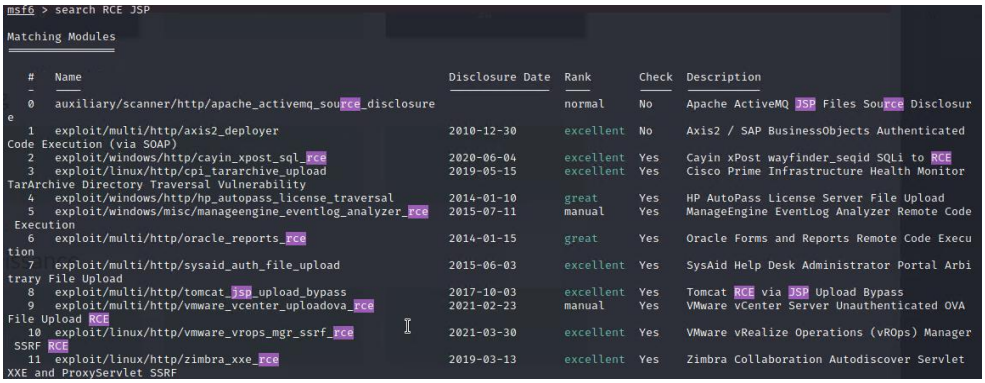
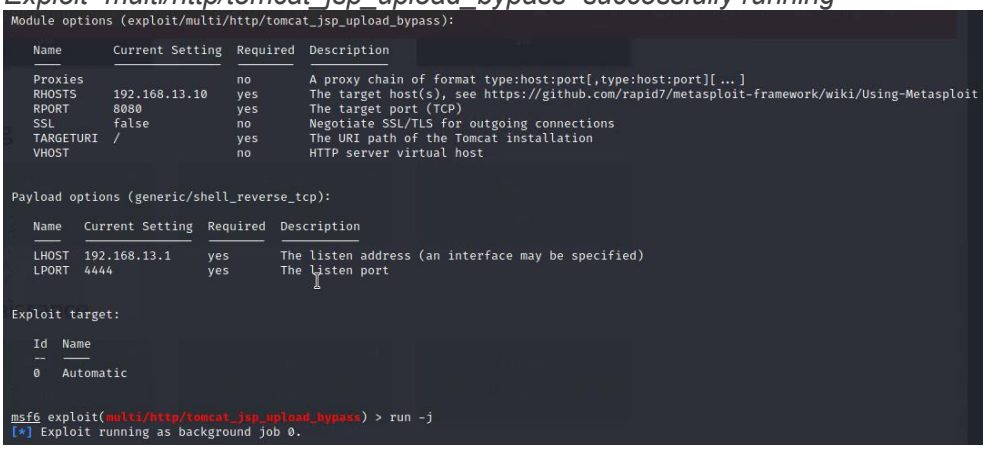
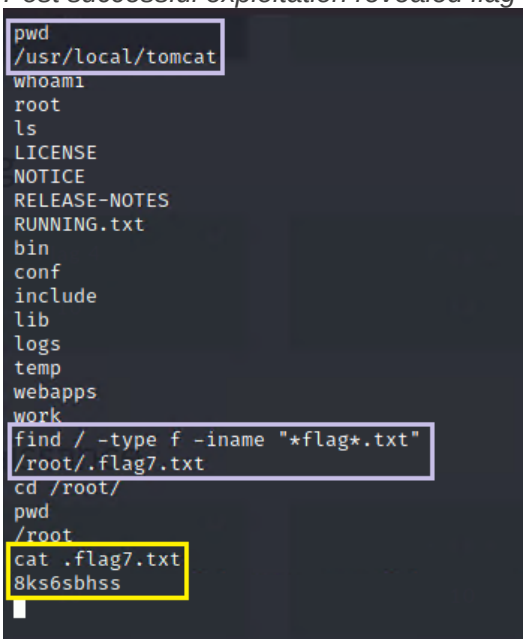
Accessed the Apache Struts vulnerability, revealing its corresponding ID “97610” which was the answer to flag 6



Affected Hosts	192.168.13.10 - Linux 192.168.13.11 - Linux 192.168.13.12 - Linux

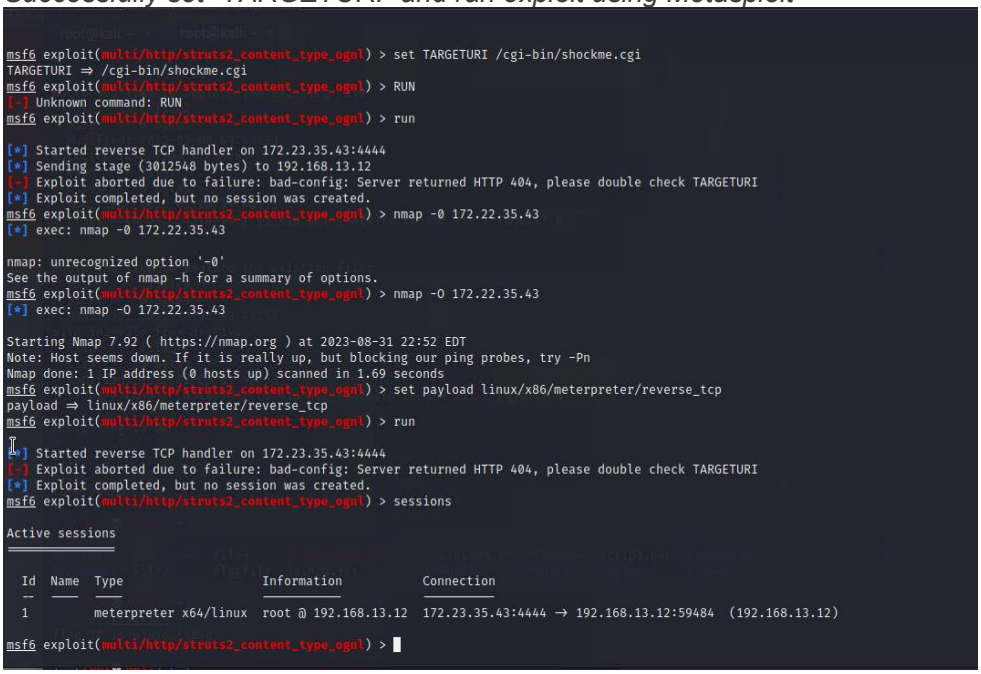
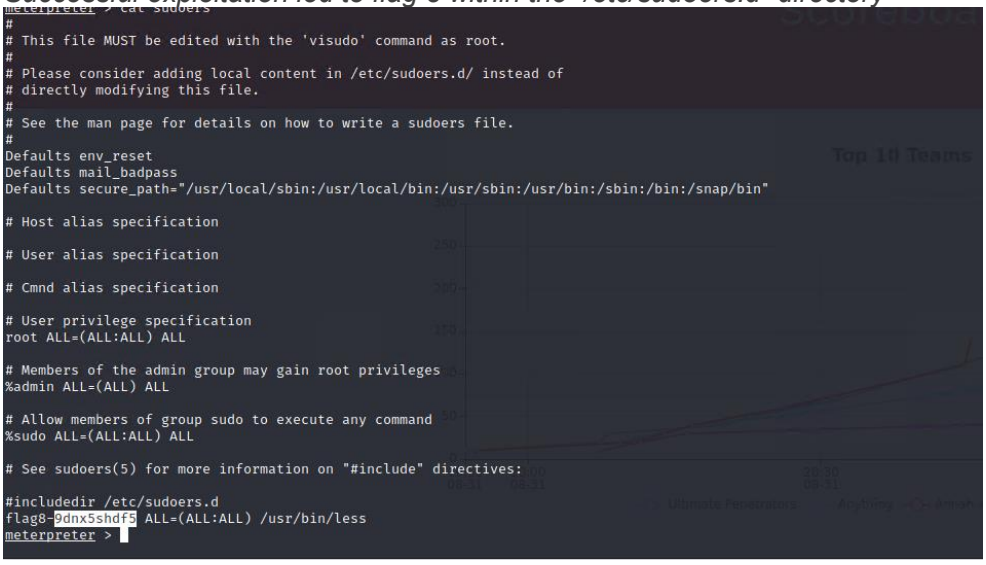
	192.168.13.13 - Linux 192.168.13.14 - Linux
Remediation	a) **Implement Network Segmentation:** Segregate the network into distinct segments based on trust levels and functions. This helps limit the exposure of hosts to unauthorized scanning and isolates critical assets from the general network. b) **Intrusion Detection System (IDS):** Deploy an IDS to monitor and detect suspicious scanning or reconnaissance activities. Configure the IDS to trigger alerts or block traffic patterns indicative of scanning attempts. c) **Access Control Lists (ACLs):** Implement ACLs on network devices to restrict access and communication between hosts. Only allow necessary traffic to and from specific hosts and services. d) **Regular Vulnerability Scanning:** Conduct regular vulnerability scans and assessments of the network to proactively identify and remediate security weaknesses before malicious actors can exploit them. e) **Update and Patch Drupal & Apache Struts:** Ensure that both Drupal and Apache Struts are installed, up to date with security patches and updates. Regularly monitor Drupal and Apache Struts security advisories and apply patches promptly. f) **Security Awareness Training:** Provide security awareness training to network users to educate them about the risks of unauthorized scanning and the importance of reporting suspicious activities. g) **Log Analysis:** Continuously monitor and analyze network logs for signs of scanning attempts. Promptly investigate and respond to any suspicious activity. h) **Network Monitoring:** Implement network monitoring tools to track network traffic and detect unusual or unauthorized scanning patterns. Configure alerts for anomalous behavior.

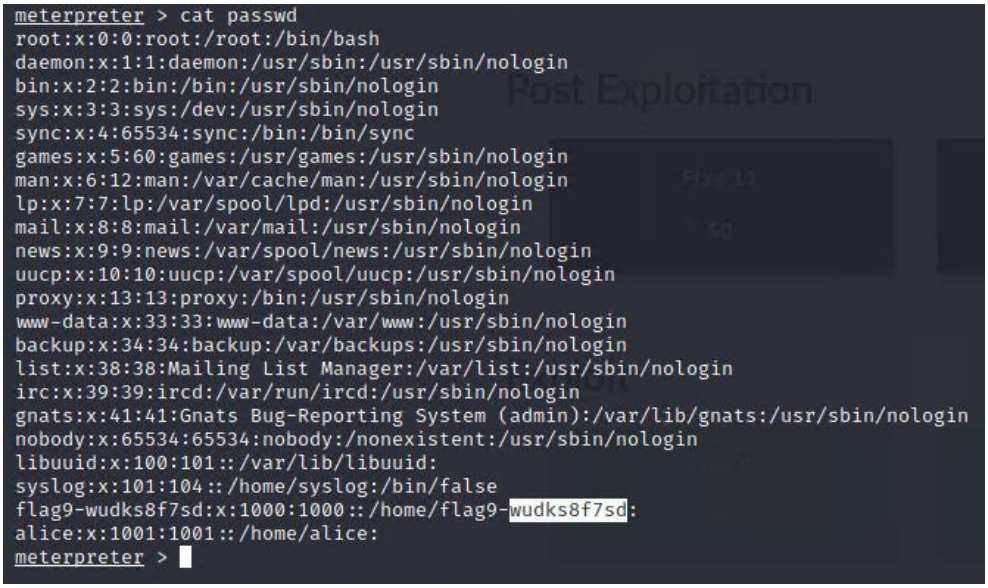
Vulnerability 13	Findings
Title	Apache Tomcat Remote Code Execution (CVE-2017-12617)
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	**Critical**
Description	Initiated MSFconsole, a specialized framework built for the rigorous practice of security penetration testing. “*search RCE JSP*” was then input, with the objective of identifying vulnerabilities linked to Tomcat and JSP components. Leading to the utilization of the exploit “multi/http/tomcat_jsp_upload_bypass”. Upon successful exploitation of the Meterpreter shell, “SHELL” was input to gain access to the command line. Proceeded to use the “find” command, “find / -type f -iname “*flag*.txt”, to search for the hidden flag file. The search yielded the location of flag 7 within the root directory. Navigated to the root directory, then displayed the contents of “flag7.txt” using the “cat” command.

	This poses a grave risk as it grants unauthorized access to the target system through the exploitation of a known Apache Tomcat vulnerability (CVE-2017-12617). An attacker could potentially execute malicious code remotely, compromising the integrity, confidentiality, and availability of the affected system. This could result in unauthorized data access, data manipulation, and even full system compromise, leading to severe consequences for the organization.
Images	*msfconsole results for "search RCE JSP"*  *Exploit "multi/http/tomcat_jsp_upload_bypass" successfully running*  *Post successful exploitation revealed flag 7 within the root directory* 

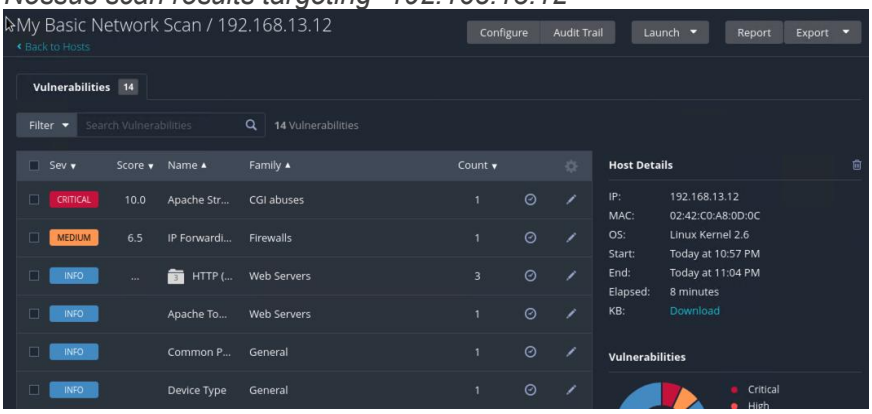
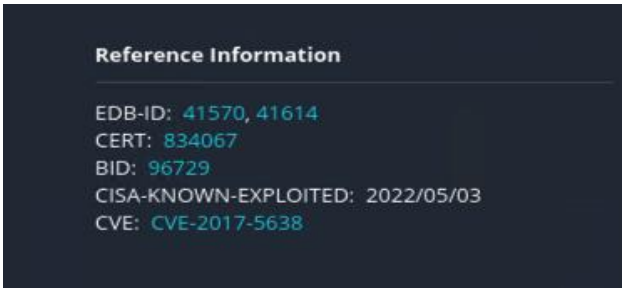
Affected Hosts	192.168.13.10 - Linux
Remediation	a) **Immediate Patching**: Apply the necessary security patches and updates to address the CVE-2017-12617 vulnerability in Apache Tomcat. Regularly monitor for security advisories and implement patches promptly. b) **Intrusion Detection System (IDS)**: Implement an IDS to detect and alert on suspicious activities or unauthorized attempts to exploit vulnerabilities. Configure the IDS to provide real-time alerts for potential threats. c) **Network Segmentation**: Isolate critical assets and systems from potentially vulnerable components to limit the potential impact of an exploit. Implement network segmentation to prevent lateral movement within the network. d) **Security Awareness Training**: Provide comprehensive security awareness training to personnel to educate them about the risks associated with remote code execution vulnerabilities and the importance of reporting suspicious activities. e) **Regular Vulnerability Scanning**: Conduct regular vulnerability scans and assessments to identify and remediate security weaknesses proactively. f) **Access Control**: Employ robust access control mechanisms to restrict access to critical systems and resources. Ensure that only authorized users can interact with sensitive components. g) **Log Monitoring**: Continuously monitor and analyze system logs for signs of unauthorized access or suspicious activities. Rapidly investigate and respond to any anomalies. h) **Incident Response Plan**: Develop and maintain an incident response plan to address security incidents promptly and effectively, minimizing potential damage in case of a breach.

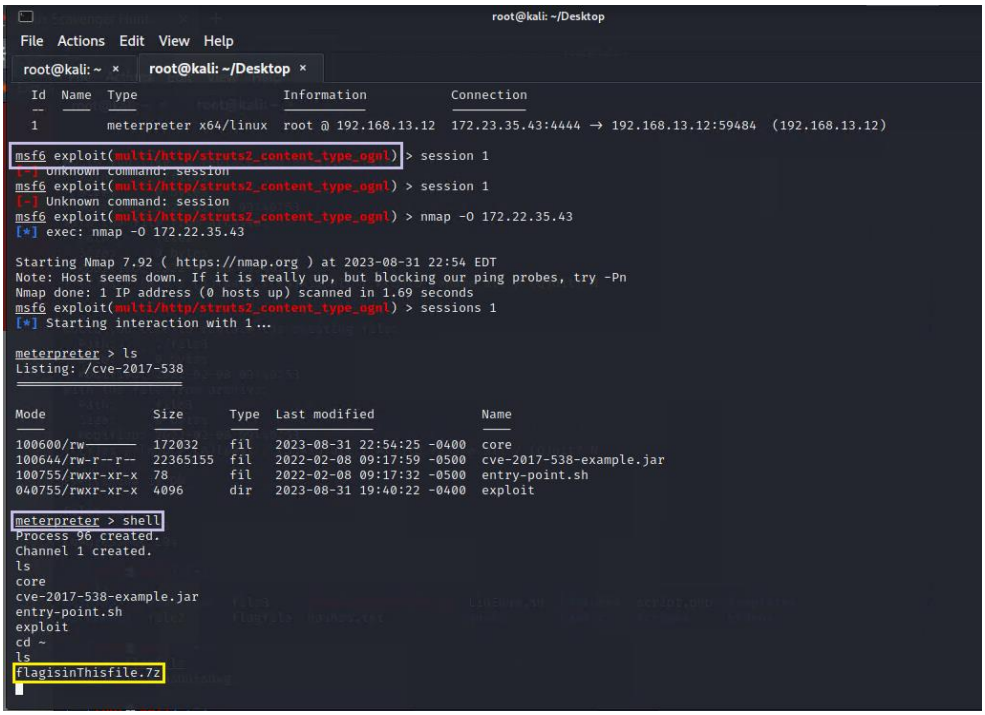
Vulnerability 14	Findings
Title	Shellshock (Port 80)
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	**Critical**
Description	Employed the metasploit "multi/http/apache_mod_cgi_bash_env_exec" exploit module to initiate the necessary actions. Set the "TARGETURI" to "/cgi-bin/shockme.cgi" to establish the required context for exploitation. Subsequently, executed the "shell" command to access the target system's shell. Examined the root privileges configuration of the "/etc/sudoers" file. This resulted in the exposer of flag 8, within the “/etc/sudoers.d” directory. From here we executed the command “cat passwd”, revealing flag 9 and its location in the “/home” directory. The Shellshock vulnerability, as exploited in this scenario, has the potential to

	result in unauthorized access, data exposure, and system compromise, making it a critical risk that demands immediate attention and remediation.
Images	*Successfully set "TARGETURI" and ran exploit using Metasploit*  *Successful exploitation led to flag 8 within the "/etc/sudoers.d" directory*  *Within the same meterpreter session, ran "cat passwd" revealing flag 9*

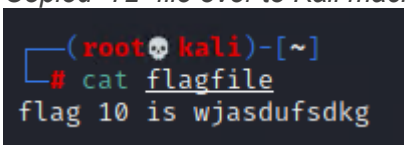
	
Affected Hosts	192.168.13.11 - Linux
Remediation	a) **Patch and Update Bash**: Addressing the root cause of the vulnerability by promptly updating the Bash shell to the latest patched version is critical. b) **Security Patch Management**: Establish a robust process for continuous monitoring and applying security updates and patches to all software components on the Linux system. c) **Access Control and Authentication**: Implement strong access controls and authentication mechanisms to restrict unauthorized access to sensitive system files, especially the "/etc/sudoers" file. d) **Web Server Hardening**: Review and strengthen the configuration of the web server (Port 80) to minimize exposure to potential vulnerabilities. e) **Intrusion Detection and Prevention**: Deploy intrusion detection and prevention systems (IDPS) to monitor network traffic for signs of Shellshock attacks or other suspicious activities. Configure alerts for immediate response. f) **Regular Vulnerability Scanning**: Conduct regular vulnerability scans and assessments to identify and remediate security weaknesses before they can be exploited. g) **Incident Response Plan**: Develop and maintain an incident response plan that outlines the steps to be taken in the event of a security breach. h) **Logging and Monitoring**: Enhance logging and monitoring capabilities on the Linux host. Maintain logs of all system activities and establish mechanisms to detect and respond to anomalies.

Vulnerability 15	Findings
Title	Apache Struts (CVE-2017-5638) Remote Code Execution (RCE)

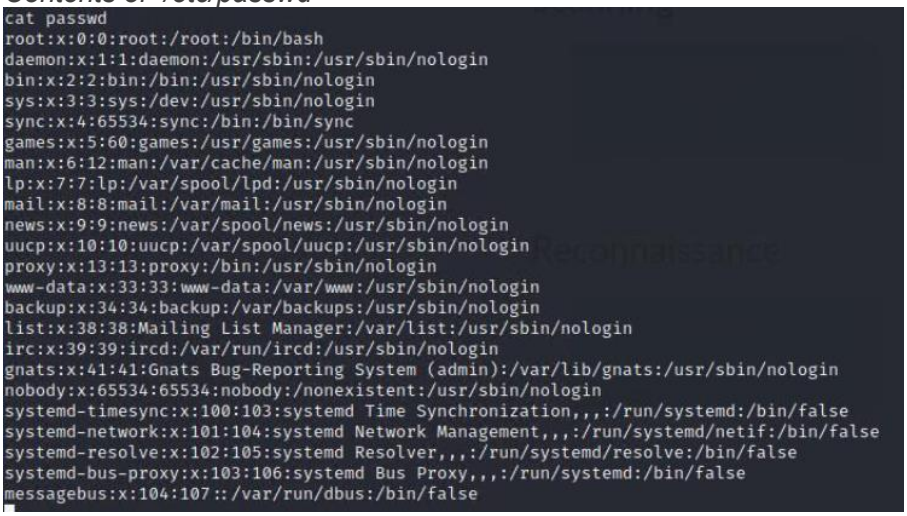
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	**Critical**
Description	Conducted network vulnerability assessments through Nmap and Nessus scans, which unveiled a critical security vulnerability residing within the Apache server, specifically identified as Apache Struts 2.3.5-2.3.31/2.5x<2.5.10.1 Jakarta Multipart parser Remote Code Execution (RCE). It is imperative to acknowledge that RCE exploits of this nature have the potential to grant malicious actors the capability to execute unauthorized programs, access confidential source code, and exfiltrate sensitive data. Leveraged a Remote Code Execution (RCE) script integrated into Metasploit. Specifically, the “multi/http/struts2_content_type_ognl” script, which allowed us to successfully exploit the identified vulnerability. Consequently, gaining unauthorized access to the Linux-based system located at IP address 192.168.13.12. Once inside the compromised system, critical system files were examined, including "/etc/passwd" and "/etc/shadow”. These files house sensitive user account information, including user credentials and permissions data. In this instance flag 10 was successfully found.
Images	*Nessus scan results targeting “192.168.13.12”*  *CVE reference information*  *Exploit successfully ran, revealing the location of flag 10*



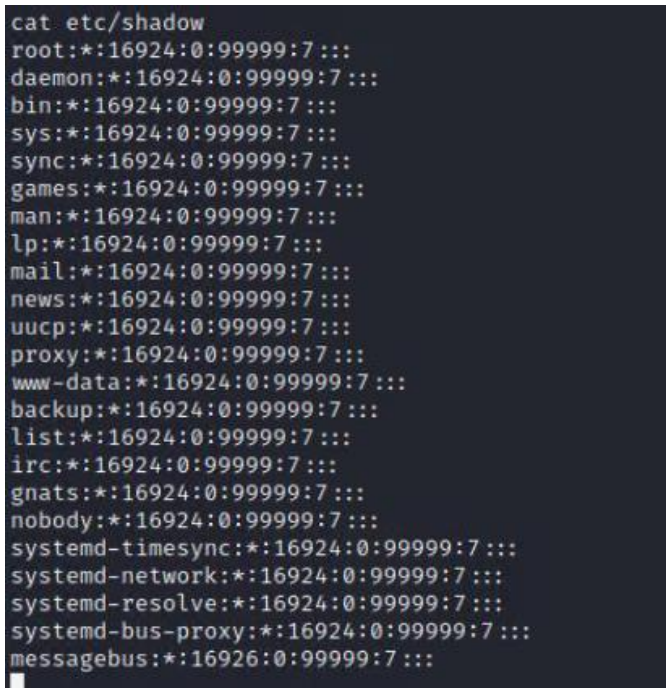
Copied "7z" file over to Kali machine to view its contents



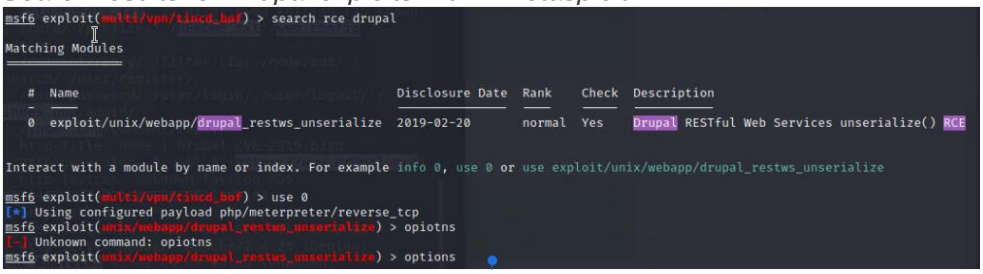
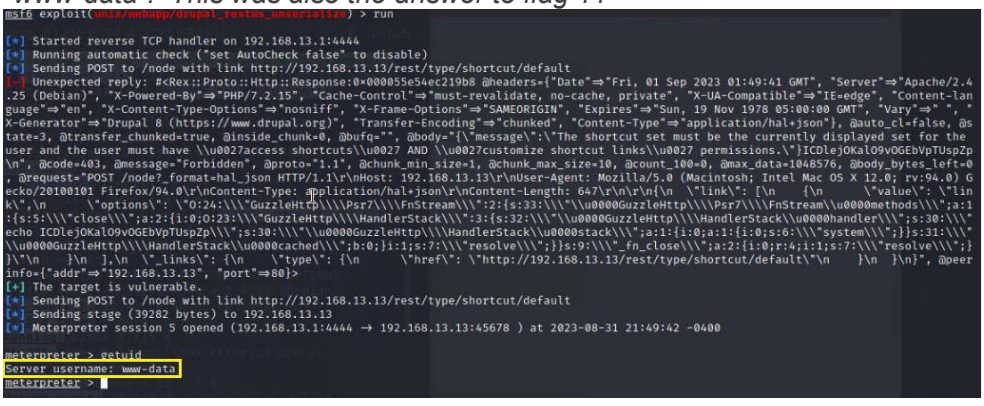
Contents of "/etc/passwd"



Contents of "/etc/shadow"

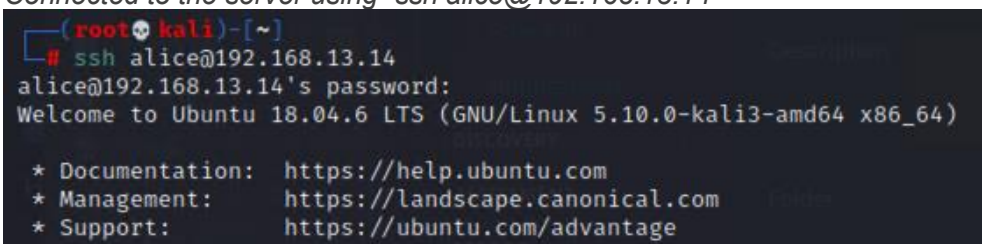
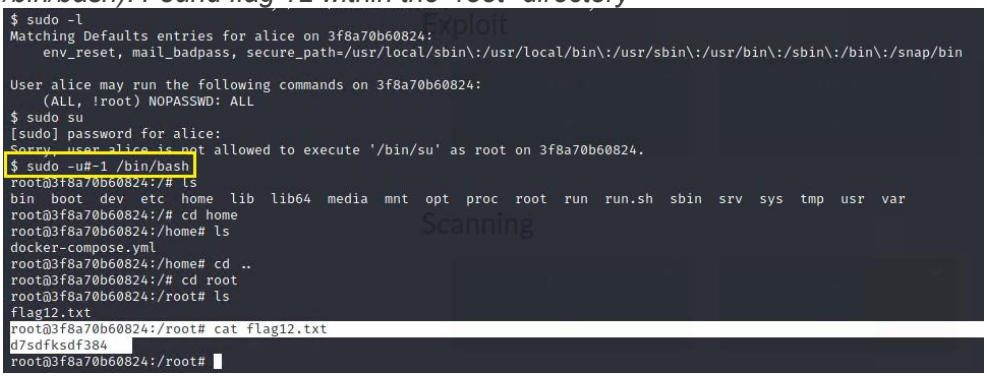


Affected Hosts	192.168.13.12 - Linux
Remediation	a) **Patch and Update Apache Struts**: This should be the highest priority. Ensure that you apply the latest security patches and updates to Apache Struts to address the known vulnerability. b) **Security Patch Management**: Establish a robust patch management process to monitor and apply security patches regularly, not just for Apache Struts but for all critical software and components in your environment. c) **Access Controls**: Enforce strict access controls and least privilege principles to limit who can access sensitive systems and data. This helps prevent unauthorized exploitation of vulnerabilities. d) **Network Segmentation**: Segment your network to isolate critical systems from public-facing networks, reducing the attack surface and limiting potential exposure. e) **Intrusion Detection/Prevention Systems (IDS/IPS)**: Implement IDS/IPS solutions to detect and block suspicious network traffic, including attempts to exploit Apache Struts vulnerabilities. f) **Web Application Firewall (WAF)**: Deploy a WAF to monitor and filter incoming web traffic, helping to detect and block malicious requests attempting to exploit the vulnerability. g) **Vulnerability Scanning**: Regularly conduct vulnerability scans to identify and remediate vulnerabilities proactively. Ensure that scan results are promptly reviewed and addressed. h) **Incident Response Plan**: Develop and maintain an incident response plan that outlines procedures for identifying and responding to security incidents, including those related to Apache Struts vulnerabilities.

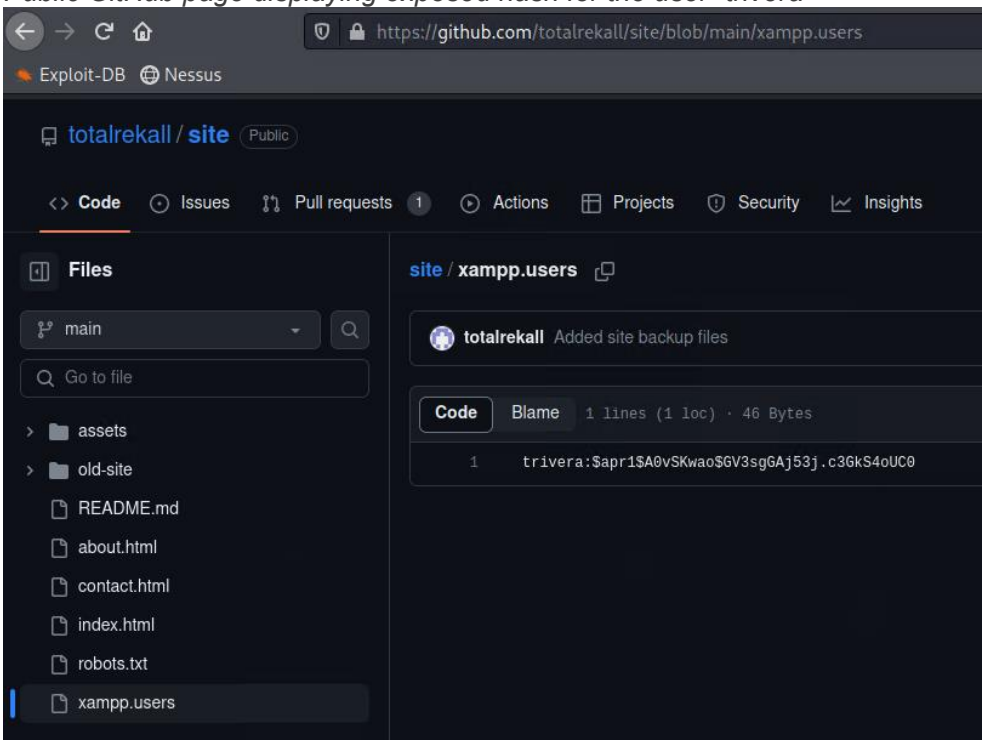
Vulnerability 16	Findings
Title	Drupal (CVE-2019-6340)
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	**Critical**
Description	Connected to MSFconsole, then conducted a search for Drupal-related exploits by employing the query "search rec drupal". This investigation yielded a single compatible exploit tailored to our objective. The specific exploit module, namely "unix/webapp/drupal_restws_unserialize", to facilitate the attainment of a Meterpreter shell. Carefully configuring the parameters, the RHOSTS value to was set to align with our target “192.168.13.13”. Executed the command "getuid" to discern the server's username “www-data”. This information doubled as the identification of flag 11. The potential risks associated with this vulnerability are multifaceted. By exploiting this vulnerability, unauthorized actors could gain access to the target server, potentially compromising data integrity, confidentiality, and system availability. Furthermore, the unauthorized acquisition of server credentials, as evidenced by the retrieval of flag 11, underscores the need for vigilant security measures to mitigate such risks effectively.
Images	*Search results for Drupal exploits within Metasploit*  *Upon successful exploitation, input “getuid” to display the server username “www-data”. This was also the answer to flag 11* 
Affected Hosts	192.168.13.13 - Linux
Remediation	a) **Patch and Update Drupal:** Apply the latest security patches and updates to the Drupal installation on the affected host (192.168.13.13). Regularly monitor Drupal security advisories and apply patches

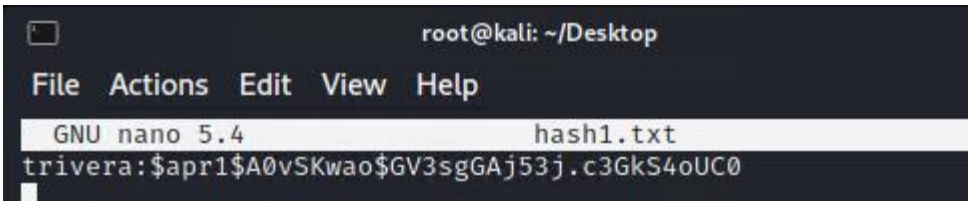
	promptly to address known vulnerabilities. b) **Access Control and Least Privilege:** Review and restrict user access and privileges to only those necessary for their roles. Implement the principle of least privilege to minimize potential damage from unauthorized access. c) **Intrusion Detection System (IDS):** Implement an IDS to monitor and detect suspicious activities on the network. Configure it to alert administrators to potential threats in real-time. d) **Regular Vulnerability Scanning:** Conduct regular vulnerability scans and assessments of your network and web applications to identify and remediate security weaknesses before malicious actors can exploit them. e) **Monitoring and Logging:** Implement comprehensive monitoring and logging of all network and server activities. Analyze logs for signs of unauthorized access or suspicious behavior and respond promptly. f) **Strong Authentication:** Enforce strong password policies and consider multi-factor authentication (MFA) to enhance the security of user accounts. g) **Security Patch Management:** Implement a robust patch management process across your network to ensure all systems and software are kept up-to-date with the latest security fixes and updates. h) **Incident Response Plan:** Develop and regularly update an incident response plan that outlines the steps to be taken in the event of a security breach. Ensure that your team is trained to respond effectively to security incidents.

Vulnerability 17	Findings
Title	Privilege Escalation (CVE-2019-14287)
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	**Critical**
Description	During an examination of WHOIS data related to Flag 1, it was observed that the associated username was "sshuser alice". To ascertain unauthorized access, an SSH connection was established to the server with the username "alice" at IP address 192.168.13.14. After two login attempts, the password "alice" was successfully guessed, allowing access to the system. The following command was executed, "sudo -u#-1 /bin/bash". This elevated the user privileges from "alice" to "root", providing unrestricted access to the system. Subsequently, this led to the discovery of flag 12 within the "root" directory. The risk associated with this vulnerability is the potential for unauthorized users to escalate their privileges from standard user accounts to "root" level access. This unauthorized elevation of privileges can lead to full control over the system, jeopardizing data integrity, confidentiality, and system availability.

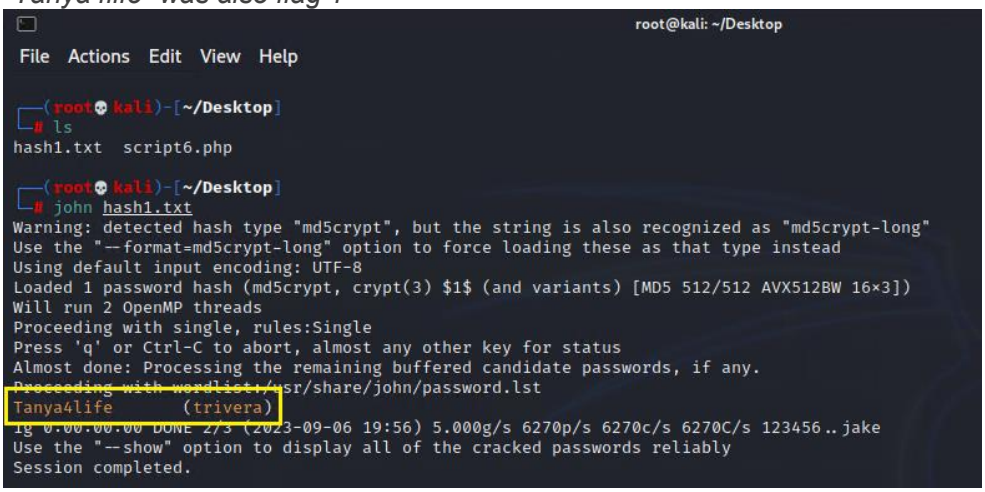
	To mitigate these risks, immediate remediation actions are essential.
Images	*Connected to the server using "ssh alice@192.168.13.14"*  *Successfully elevated privileges from "Alice" to "root" using: (sudo -u#-1 /bin/bash). Found flag 12 within the "root" directory* 
Affected Hosts	192.168.13.14 - Linux
Remediation	a) **Implement Strong Password Policies:** Enforce complex password requirements and regular password changes to make it more challenging for unauthorized users to guess or crack passwords. b) **Implement Multi-Factor Authentication (MFA):** Enable MFA for SSH access to add an extra layer of security. Even if passwords are compromised, MFA can prevent unauthorized access. c) **Disable Root Login:** Prohibit direct root logins via SSH. Instead, use sudo for privilege elevation to restrict access and reduce the attack surface. d) **Configure SSH Access Controls:** Limit SSH access to trusted IP addresses or networks. Use SSH keys for authentication when possible, and restrict user access based on the principle of least privilege. e) **Monitor Failed Login Attempts:** Implement an intrusion detection system (IDS) or monitoring solution to log and alert on multiple failed login attempts, helping detect and respond to potential attacks. f) **Regularly Update and Patch Software:** Keep the system and all software components, including the SSH server and operating system, up to date with security patches to address known vulnerabilities. g) **Audit User Privileges:** Regularly review and audit user privileges, including sudo configurations, to ensure that users have only the necessary permissions required for their tasks. h) **Use Stronger Authentication Methods:** Consider implementing public key authentication for SSH connections, which can provide a

	higher level of security compared to password-based authentication.

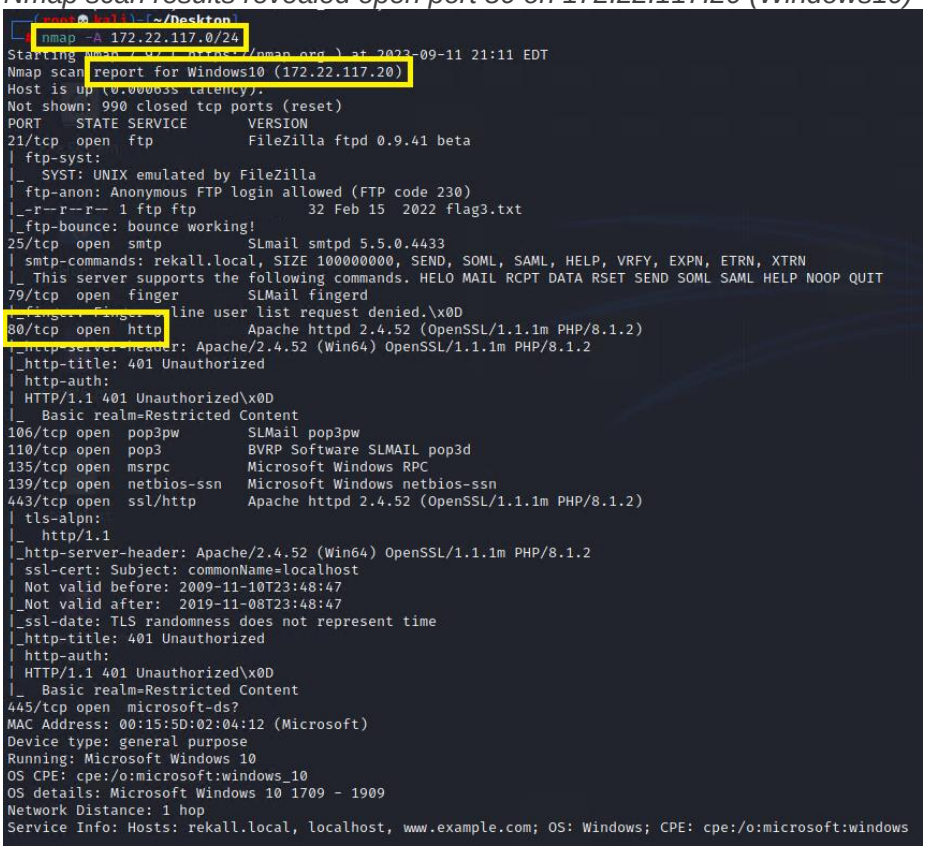
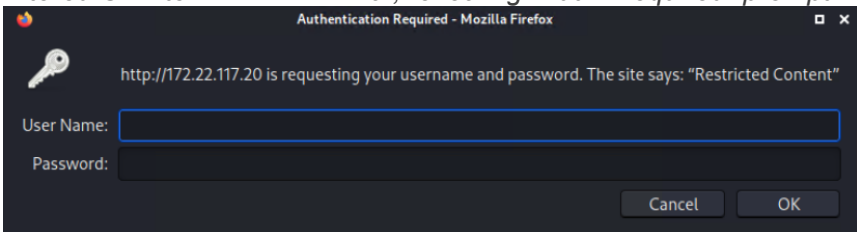
Vulnerability 18	Findings
Title	Username & Password Hash Exposed (GitHub)
Type (Web app / Linux OS / WIndows OS)	Web Application
Risk Rating	**Critical**
Description	An in-depth examination of GitHub was conducted, which ultimately led to the discovery of the totalrekall GitHub repository. Within this repository, particular attention was directed towards the "site" subdirectory, which contained a file named "xampp.users". This file yielded significant findings in the form of exposed credentials: "trivera:$apr1$A0vSKwao$GV3sgGAj53j.c3GkS4oUC0". A new text file was meticulously created containing a copy of the credentials. This involved the execution of the command "echo ' trivera:$apr1$A0vSKwao$GV3sgGAj53j.c3GkS4oUC0' > hash1.txt," effectively consolidating the exposed credentials into a structured document. The formidable password cracking tool, John the Ripper, was harnessed to perform an analysis of the captured hash. The successful outcome of this endeavor confirmed the password associated with the user "trivera" as "Tanya4life". This password was also flag 1. The revelation of this password underscores the critical need for fortified password management and access control mechanisms, as well as maintaining vigilance in safeguarding sensitive information.
Images	*Public GitHub page displaying exposed hash for the user “trivera”*  *“xampp.users” contents copied into text file*



Cracked the exposed user hash with "john the ripper", this password "Tanya4life" was also flag 1



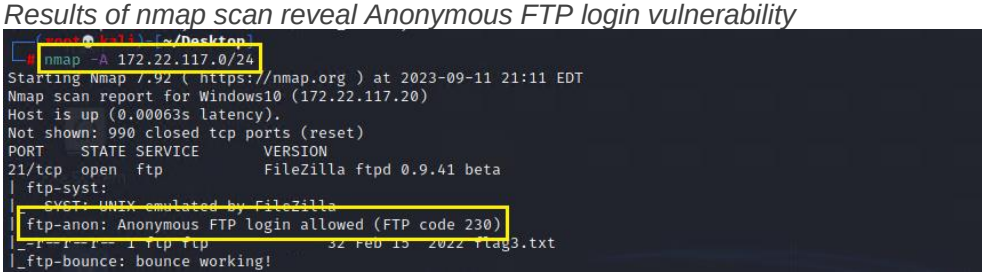
Affected Hosts	Total Rekall web server
Remediation	a) **Change Compromised Credentials**: Immediately change the compromised credentials associated with the user "trivera" to prevent unauthorized access. b) **Implement Multi-Factor Authentication (MFA)**: Enforce the use of MFA for all user accounts to provide an additional layer of security against unauthorized access, even if credentials are exposed. c) **Enhance Password Policies**: Strengthen password policies by enforcing longer and more complex passwords, regular password updates, and prohibiting the use of easily guessable passwords. d) **Regular Credential Scans**: Implement routine scans and checks for exposed credentials on public repositories, ensuring that sensitive information is promptly identified and secured. e) **Privilege Segmentation**: Restrict user privileges to the minimum necessary for their roles, limiting the potential impact of compromised credentials. f) **Continuous Monitoring**: Employ continuous monitoring and intrusion detection systems to identify and respond to unauthorized access attempts promptly. g) **Access Control**: Review and refine access control mechanisms to ensure that only authorized personnel have access to sensitive information. h) **Security Training and Awareness**: Conduct security awareness training for all personnel to educate them on the importance of strong password practices and the risks associated with credential exposure.

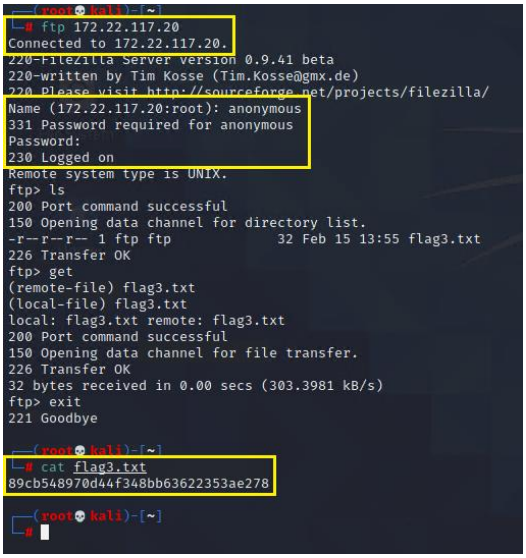
Vulnerability 19	Findings
Title	Unauthorized Network Scanning (Nmap)
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	**High**
Description	Unauthorized access was gained using credentials obtained from a GitHub repository. Within this unauthorized access, a single file named "flag2.txt", which contained sensitive information, was discovered. Executed an Nmap scan with the command "nmap -A 172.22.117.0/24". This scan unveiled the presence of an open port 80 on the system with the IP address 172.22.117.20. To exploit this finding, the IP address 172.22.117.20 was accessed through a web browser. Credentials obtained earlier (user: trivera | pass: Tanya4life) were employed to gain unauthorized access and retrieve flag 2.
Images	*Nmap scan results revealed open port 80 on 172.22.117.20 (Windows10)*  *Altered URL to "172.22.117.20", revealing "Auth. Required" prompt* 

	Accessed page using the compromised credentials found on GitHub  *Accessed "flag2.txt" revealing flag 2*
Affected Hosts	172.22.117.20 – Windows10
Remediation	a) **Implement Strong Access Controls**: Enhance access controls and enforce strict authentication mechanisms to prevent unauthorized access to sensitive systems and data. Strong access controls are essential for preventing unauthorized access, which was a critical aspect of the vulnerability. b) **Credential Management**: Regularly review and update credentials to ensure that compromised or weak credentials are promptly replaced with strong, unique passwords. Implement multi-factor authentication (MFA) where applicable. Proper credential management is crucial in preventing unauthorized access. c) **GitHub Security**: Strengthen security practices related to GitHub repositories by regularly auditing and removing sensitive information, including credentials, from public repositories. This is essential to prevent exposure of sensitive information. d) **Network Segmentation**: Employ network segmentation to isolate critical systems from unauthorized scanning and access. Limit communication between systems to only what is necessary. Network segmentation helps contain and isolate potential threats. e) **Intrusion Detection System (IDS)**: Deploy an IDS to monitor and detect suspicious network activities, including unauthorized scanning attempts. Configure the IDS to generate alerts for potential threats. An IDS helps in real-time threat detection. f) **Security Awareness Training**: Conduct security awareness training for personnel to educate them about the risks associated with unauthorized scanning and the importance of secure credential management. Human awareness is a critical defense against social engineering attacks. g) **Regular Vulnerability Scanning**: Conduct regular vulnerability scans on your network to proactively identify and remediate security

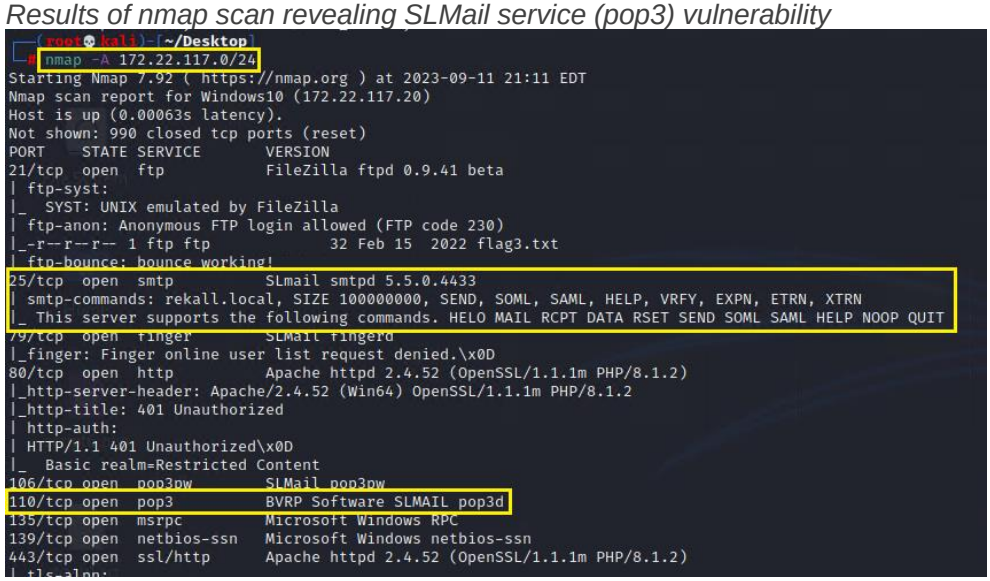
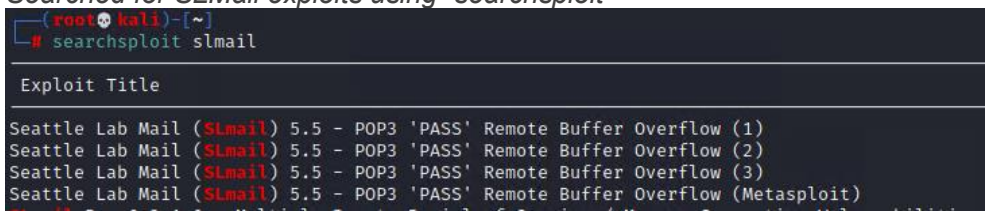
	weaknesses before malicious actors can exploit them. Regular scans help identify vulnerabilities before they can be exploited. h) **Access Logging and Monitoring**: Implement comprehensive access logging and monitoring to track user activities, especially those related to privileged accounts. Monitor logs for suspicious activities and respond promptly to anomalies. Monitoring helps in early detection of unauthorized access.

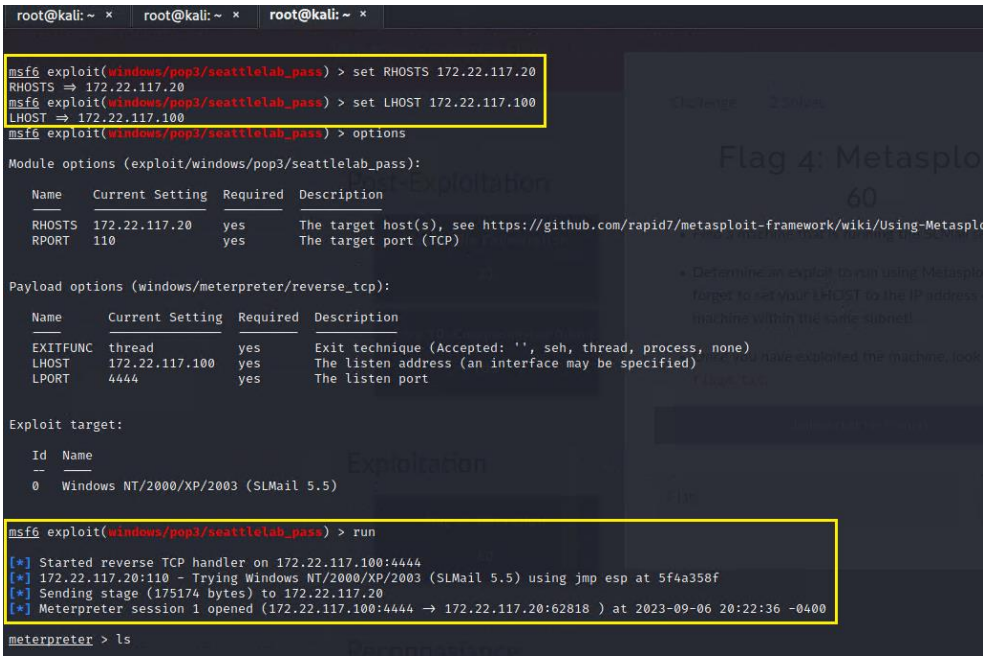
Vulnerability 20	Findings
Title	Anonymous (FTP) Login
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	**High**
Description	The risk associated with this vulnerability primarily pertains to unauthorized access and data exposure. Unauthorized users exploiting the anonymous FTP login could potentially compromise data integrity and confidentiality. Network assessment conducted via Nmap scanning, nmap –A 172.22.117.0/24, identified that the FTP server on the target system allows for anonymous access. An FTP connection was initiated with the command "ftp 172.22.117.20". Upon establishing the connection, using login credentials "anonymous" for both the username and password successfully, granting unauthorized access to the FTP server. A file retrieval command, "get", was executed, leading to the discovery of a text file named "flag3.txt". This file contained sensitive information and was retrieved during the unauthorized access. The FTP session was terminated, and the command "cat flag3.txt" was employed to display flag 3.

Images	*Results of nmap scan reveal Anonymous FTP login vulnerability*
	 
	`nmap -A 172.22.117.0/24`
	`Starting Nmap 7.92 ( https://nmap.org ) at 2023-09-11 21:11 EDT`
	`Nmap scan report for Windows10 (172.22.117.20)`
	`Host is up (0.00063s latency).`
	`Not shown: 990 closed tcp ports (reset)`
	`PORT STATE SERVICE VERSION`
	`21/tcp open ftp FileZilla ftpd 0.9.41 beta`
	`| ftp-syst:`
	`| SYST: UNIX emulated by FileZilla`
	`| ftp-anon: Anonymous FTP login allowed (FTP code 230)`
	`|_-r--r--r-- 1 ftp ftp 32 Feb 15 2022 flag3.txt`
	`|_ftp-bounce: bounce working!`
	
	FTP exploit successfully ran, revealing flag 3

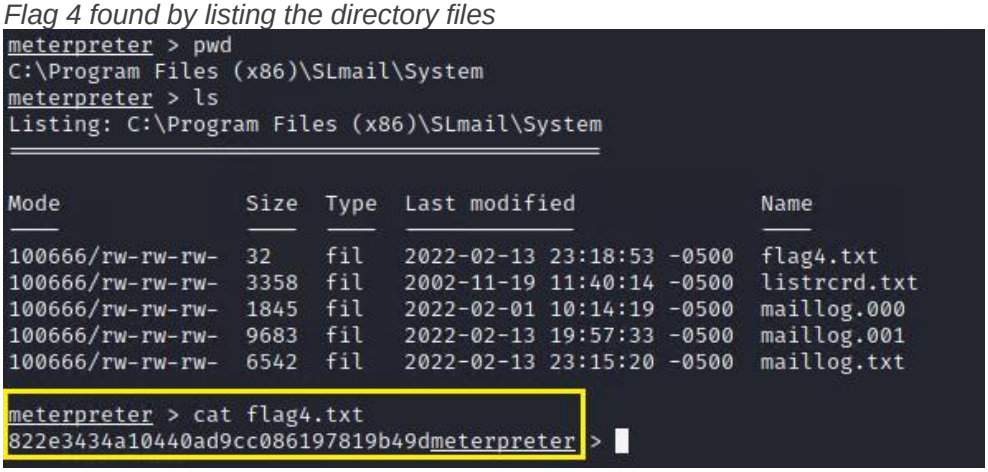
	
Affected Hosts	172.22.117.20 – Windows10
Remediation	a) **Disable Anonymous FTP Access:** Immediately disable anonymous FTP access on the FTP server to prevent unauthorized users from logging in with default credentials. b) **Implement Strong Authentication:** Enforce strong and unique usernames and passwords for FTP access, including complexity requirements and regular password changes to enhance security. c) **Regularly Update and Patch:** Keep the FTP server software up to date with the latest security patches and updates to address known vulnerabilities and protect against exploitation. d) **Network Segmentation:** Implement network segmentation to isolate the FTP server from the rest of the network, reducing the potential impact of unauthorized access. e) **Access Control Lists (ACLs):** Configure Access Control Lists (ACLs) on the FTP server to restrict access only to authorized IP addresses or IP ranges, blocking incoming connections from unauthorized sources. f) **Security Monitoring:** Establish robust security monitoring and logging for the FTP server, regularly reviewing and analyzing logs to detect and respond to suspicious or unauthorized access attempts. g) **User Training and Awareness:** Educate users and administrators about secure FTP practices, emphasizing the risks associated with anonymous access, and promoting strong password management. h) **Regular Vulnerability Scanning:** Conduct routine vulnerability scanning and testing of the FTP server to identify and proactively address security weaknesses before they can be exploited.

Vulnerability 21	Findings
Title	SLMail Service (POP3)

Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	**High**
Description	Network assessment was conducted using Nmap scanning (nmap -A 172.22.117.0/24), revealing the presence of the SLMail service running on SMTP port 25 and port 110 for POP3 on the target system. The "searchsploit" tool was employed to identify a Metasploit exploit module compatible with the identified version of SLMail, command used: "searchsploit slmail". Metasploit was then leveraged to execute the exploit "windows/pop3/seattlelab_pass", successfully gaining unauthorized access via port 110 on the Windows 10 machine. A directory listing command (ls) was executed within the Meterpreter session, revealing the existence of "flag4.txt." This file was subsequently accessed using the "cat" command from within the Meterpreter session, disclosing the contents of flag 4.
Images	*Results of nmap scan revealing SLMail service (pop3) vulnerability*  *Searched for SLMail exploits using "searchsploit"*  *Successfully ran exploit on the target "172.22.117.20", port 110*



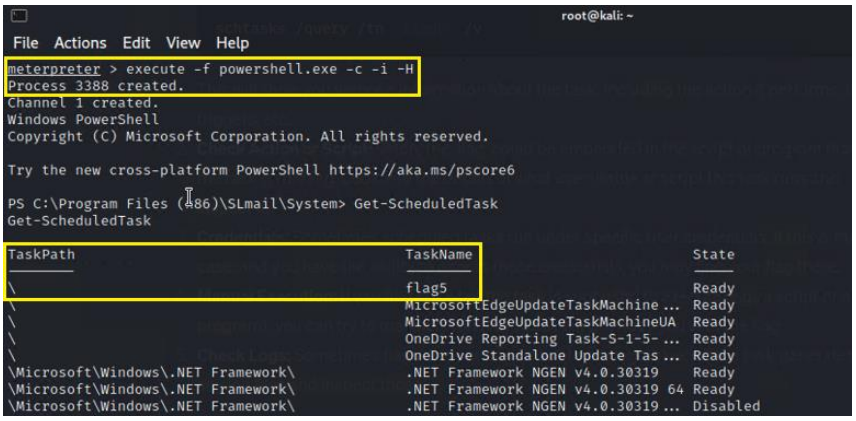
Flag 4 found by listing the directory files



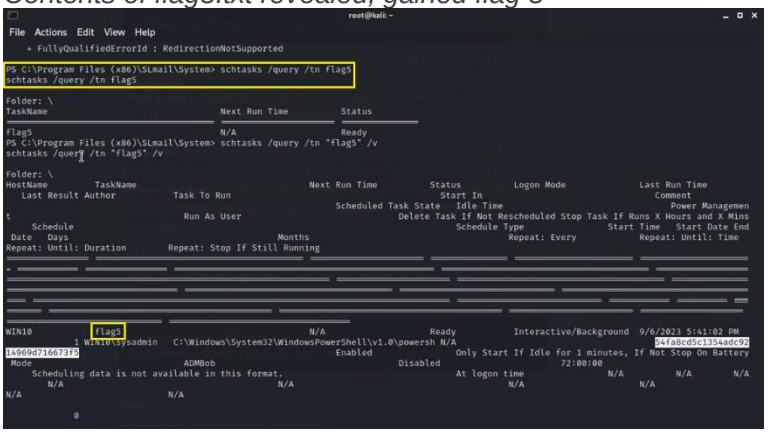
Affected Hosts	172.22.117.20 – Windows10
Remediation	a) **Apply Security Updates:** Regularly update the SLMail service and the underlying operating system to patch known vulnerabilities and enhance security. b) **Access Control:** Implement strict access controls and authentication mechanisms to restrict unauthorized access to the SLMail service. c) **Network Segmentation:** Use network segmentation to isolate critical systems from potentially vulnerable or exposed systems, limiting the scope of potential breaches. d) **Firewall Configuration:** Configure firewalls to filter and monitor traffic on ports 25 and 110, only allowing authorized and necessary traffic to reach the SLMail service. e) **Intrusion Detection and Prevention:** Deploy intrusion detection and prevention systems to monitor network traffic for suspicious activities and block potential threats in real-time. f) **Password Policies:** Enforce strong password policies for email

	accounts, including regular password changes and complexity requirements. g) **Security Training:** Provide security training for users and administrators to raise awareness of email security best practices, including recognizing phishing attempts. h) **Encryption**: Use email encryption protocols (e.g., TLS) to secure email communication between clients and servers.

Vulnerability 22	Findings
Title	Task Scheduler
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	**Medium**
Description	This vulnerability presents a moderate level of risk due to several factors. First, its exploitation demands a certain level of technical expertise and familiarity with specific commands within the Meterpreter framework, making it less accessible to less-skilled attackers. Second, while there is a risk of privilege escalation, achieving complete system control necessitates exploiting additional vulnerabilities or system weaknesses, complicating the attack process. Third, the potential impact varies based on system configuration and specific scheduled tasks, with risks including data exposure, malicious code execution, and system disruption, but the severity can be limited in some cases. Lastly, organizations can reduce this risk by proactively implementing measures such as task monitoring and permission restrictions. A hint alluding to "scheduled tasks" prompted an investigation into the system's scheduled tasks. A command shell was then initiated within the Meterpreter framework using the command "execute -f powershell.exe -c -i -H". The command s instructing the system to run the powershell.exe executable in an interactive mode, indicating that the subsequent part of the command should be treated as a PowerShell script or command. This revealed the path to flag 5. Additional information regarding scheduled tasks was accessed by employing the "schtasks /query /tn flag5" command. This provided the contents of flag 5.
Images	*Successful exploitation revealing path to flag 5*



Contents of flag5.txt revealed, gained flag 5

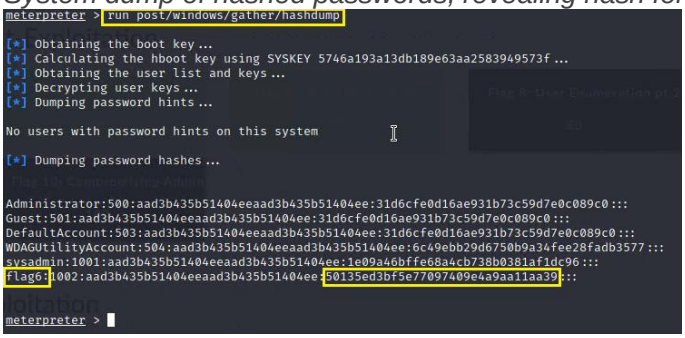


Affected Hosts	172.22.117.20 – Windows10
Remediation	a) **Review and Remove Unnecessary Tasks:** Conduct a thorough review of all scheduled tasks on the system. Identify and remove any tasks that are no longer required or appear suspicious. b) **Limit Permissions:** Restrict permissions for creating or modifying scheduled tasks to authorized users only. Avoid assigning overly permissive rights to users or groups. c) **Regularly Monitor Scheduled Tasks:** Establish a system for monitoring scheduled tasks for any unauthorized changes or suspicious activities. Implement logging and alerting mechanisms to detect and respond to unusual task-related events. d) **Enhance Authentication and Authorization:** Implement strong authentication mechanisms for users with the ability to create or modify scheduled tasks. Utilize role-based access control (RBAC) to limit access to task scheduling functions. e) **Implement Whitelisting:** Create a whitelist of trusted scripts and executables that can be executed via scheduled tasks. Block the execution of scripts or executables that are not on the whitelist. f) **Regular System Patching and Updates:** Keep the operating system and Task Scheduler software up to date with the latest security patches and updates. Apply vendor-recommended security configurations. g) **Educate and Train Users:** Provide training to system administrators

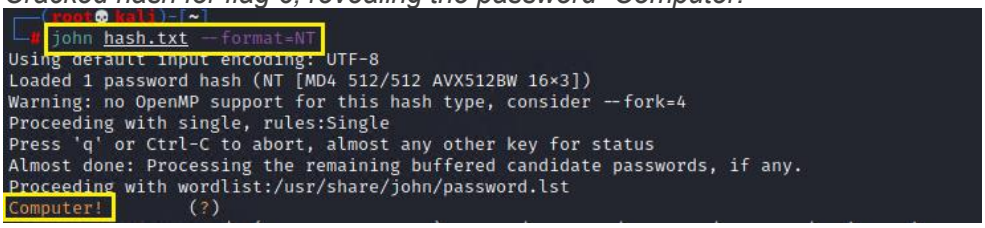
	and users about secure task scheduling practices. Highlight the risks associated with unauthorized or unsecured scheduled tasks. h) **Audit Scheduled Tasks Regularly:** Perform regular audits of scheduled tasks to ensure compliance with security policies and procedures. Document and track changes made to scheduled tasks.

Vulnerability 23	Findings
Title	Exposed User/System Hashes (Kiwi)
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	**Critical**
Description	Following the successful compromise of the SLMail service using Metasploit, access to the Meterpreter shell provided us with SYSTEM-level privileges. Subsequently, the "kiwi" tool was leveraged to further exploit the compromised system. Executed the command "run post/windows/gather/hashdump", which facilitated the extraction of password hashes residing on the compromised system. Among these hashes, one corresponding to flag 6 was identified and isolated, recorded in a text file named "hash.txt". To decipher the password associated with flag 6, the John the Ripper tool was employed using the command "john hash.txt --format=NT". This proved successful, as the hash was cracked, revealing the password: "Computer!". Proceeded to dump the cached credentials, using “kiwi_cmd lsadump::cache”. This revealed the user “administrator” and its hashed NTML. This data was copied into a text file called “hash.txt”, then cracked using “john hash.txt --format=mscash2”. Revealing the password of “Changeme!” for the user “ADMBob”. Deployed the PsExec exploit, “windows/smb/psexec” to successfully access the windows 2019 server using the compromised credentials for “ADMBob”. After gaining access; the command “\>net users” was ran, which displayed flag 8 as a listed user. Upon reentering the Meterpreter session, an exploit named "dcsync_ntlm administrator" was executed. This operation proved successful, resulting in the extraction of sensitive information denoted as "flag 10," specifically comprising the NTLM password hash.
Images	*Successfully launched “Kiwi”* 

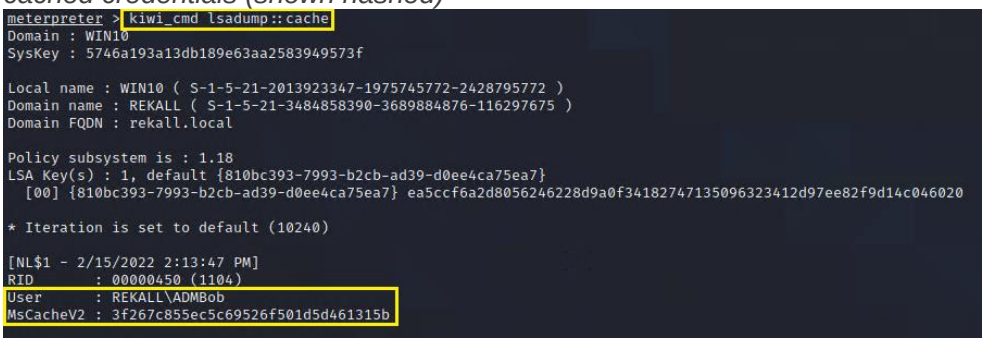
System dump of hashed passwords, revealing hash for flag 6



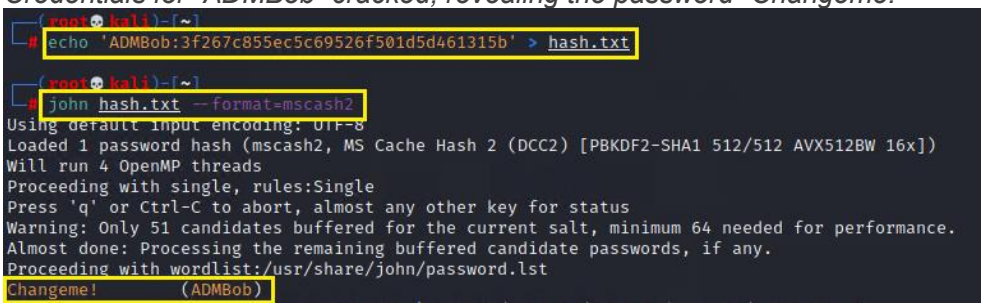
Cracked hash for flag 6, revealing the password "Computer!"



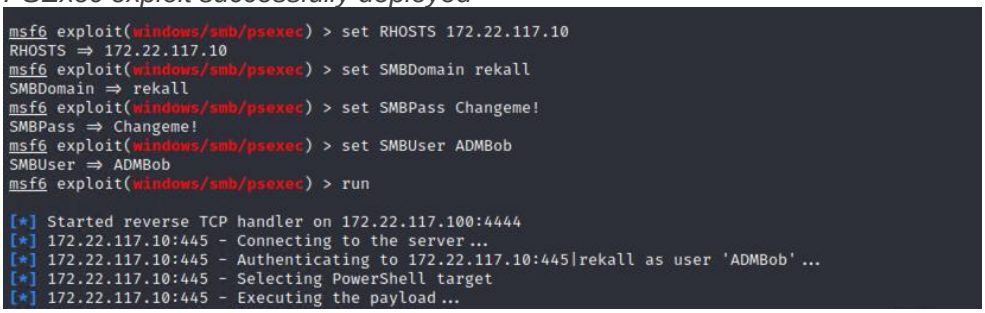
Dumped cached credentials, revealing admin user "ADMBOB" and their cached credentials (shown hashed)



Credentials for "ADMBob" cracked, revealing the password "Changeme!"



PSExec exploit successfully deployed



<table>
<tr><td></td><td>

```
meterpreter > shell
Process 3828 created.
Channel 2 created.
Microsoft Windows [Version 10.0.17763.737]
(c) 2018 Microsoft Corporation. All rights reserved.
```

Flag 8 found as a listed user with the command "\>net users"

```
C:\>net users
net users

User accounts for \\

-------------------------------------------------------------------------------
ADMBob                   Administrator            flag8-ad12fc2ffc1e47
Guest                    hdodge                   jsmith
krbtgt                   tschubert
The command completed with one or more errors.
```

DCsync exploit deployed, revealing user "administrator" and its NTLM hash. This NTLM hash was also flag 10

```
meterpreter > dcsync_ntlm administrator
[!] Running as SYSTEM; function will only work if this computer account has replication privileges (e.g. Domain Controller)
[+] Account  : administrator
[+] NTLM Hash : 4f0cfd309a1965906fd2ec39dd23d582
[+] LM Hash  : 0e9b6c3297033f52b59d01ba2328be55
[+] SID      : S-1-5-21-3484858390-3689884876-116297675-500
[+] RID      : 500

meterpreter >
```

</td></tr>
<tr><td>Affected Hosts</td><td>172.22.117.10 – Windows Domain Controller
172.22.117.20 – Windows10</td></tr>
<tr><td>Remediation</td><td>

a) **Isolate and Investigate Compromised Systems:** Immediately isolate the compromised systems to prevent further unauthorized access and conduct a thorough investigation to determine the extent of the breach.

b) **Password Changes and Account Lockouts:** Force password changes for all user accounts and service accounts on the compromised systems, implement account lockout policies, and review and update password policies to enforce strong, complex passwords.

c) **Remove Cached Credentials:** Audit and remove cached credentials from all systems to eliminate the possibility of unauthorized access using cached credentials.

d) **Patch and Update Systems:** Apply security patches and updates to address known vulnerabilities and security weaknesses on both the Windows 10 system and the Domain Controller. Regularly update and maintain all systems and software within the network.

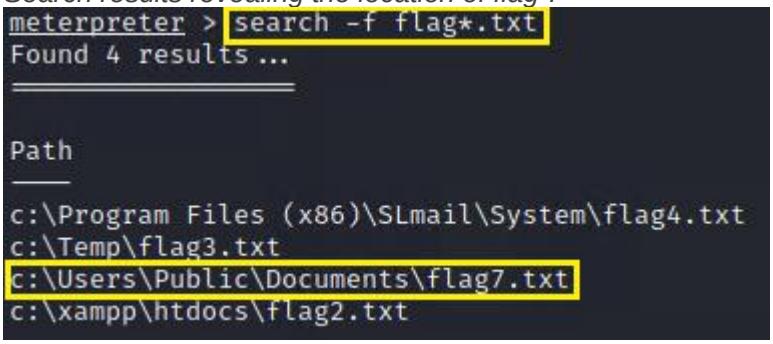
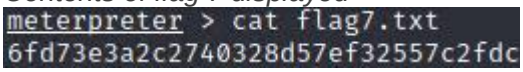
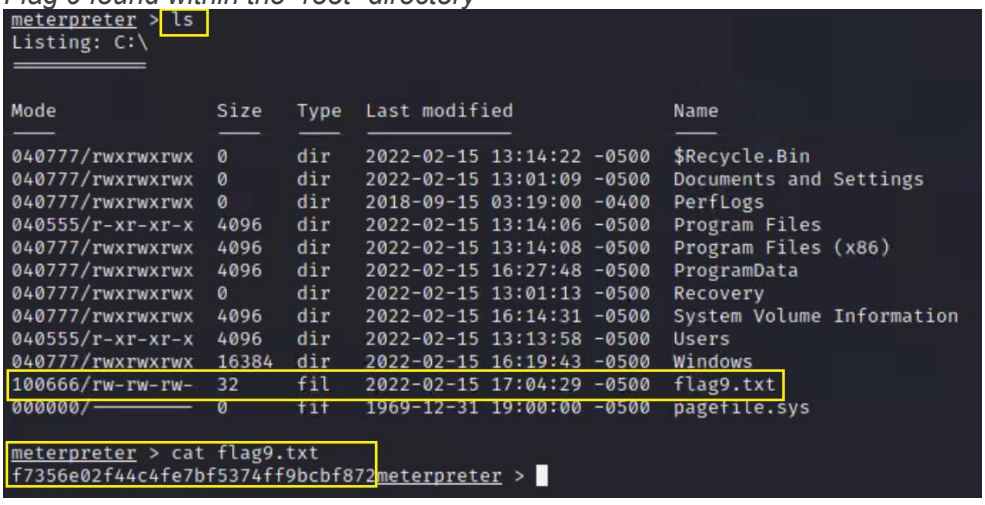
e) **Implement Network Segmentation:** Segment the network to isolate critical systems, especially Domain Controllers, from less critical systems. This limits lateral movement for attackers.

f) **Enhance Access Controls:** Review and revise user and group access controls to limit access to critical systems only to authorized personnel. Implement the principle of least privilege (PoLP) to restrict user access rights to the minimum necessary for their roles.

g) **Implement Advanced Threat Detection:** Deploy advanced threat detection and intrusion detection systems (IDS/IPS) to monitor network traffic for suspicious behavior and attacks. Set up real-time alerts and automated responses to potential threats.

</td></tr>
</table>

	h) **Incident Response Plan:** Develop and implement a comprehensive incident response plan that outlines the steps to take in case of a security incident. Ensure that relevant stakeholders are trained in incident response procedures.

Vulnerability 24	Findings
Title	Sensitive Data Exposure - Public Directory
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	**Medium**
Description	A search was conducted using the Meterpreter command "search –f flag*.txt" to identify files matching the pattern "flag*.txt". This search revealed the presence of "flag7.txt" located within the "C:\Users\Public\Documents" directory. Subsequently, an exploration of the system's directories led to the discovery of "flag9.txt" in the root directory "C:". To confirm the contents of this file, the command "cat flag9.txt" was executed within the Meterpreter session. Revealing flag 9 in the contents.
Images	*Search results revealing the location of flag 7*  *Contents of flag 7 displayed*  *Flag 9 found within the "root" directory* 

Affected Hosts	172.22.117.20 – Windows10
Remediation	a) **Access Control and Permissions Review:** Conduct a comprehensive review of access controls and permissions for sensitive directories and files. Ensure that only authorized users and groups have access to sensitive data. b) **Data Classification and Labeling:** Implement a data classification and labeling policy to identify and categorize sensitive data appropriately. Apply labels or tags to sensitive files to clearly indicate their importance. c) **File Encryption:** Implement file-level encryption for sensitive files stored on the system. This adds an extra layer of security, protecting data even if unauthorized access occurs. d) **Regular Scans for Sensitive Data:** Utilize scanning tools and software to regularly scan the system for sensitive data. These tools can identify and report the presence of sensitive files in public directories. e) **User Training and Awareness:** Provide training to users and administrators about the importance of handling sensitive data properly. Educate them on security best practices and the potential risks associated with exposing sensitive information. f) **Access Monitoring and Logging:** Set up access monitoring and comprehensive logging for sensitive directories. Regularly review access logs to detect unauthorized access attempts and anomalies. g) **Implement Data Loss Prevention (DLP) Solutions:** Deploy DLP solutions that can monitor and prevent the unauthorized transfer or sharing of sensitive data. These tools can help enforce data protection policies. h) **Regular Security Audits:** Conduct regular security audits and assessments of the system to identify and remediate vulnerabilities proactively. Ensure that sensitive data exposure risks are a focus of these audits.