

DawgCTF 2020 Reversing

Saturday, March 7, 2020 12:16 PM

Put your thang down flip it and reverse it.

Challenge 184 Solves >

Put your thang down flip
it and reverse it

150

Ra-ta-ta-ta-ta-ta-ta-ta-ta.

Authors: trashcanna and pleoxconfusa

 missyelliott

Downloaded the missyelliott binary and used **Ghidra**.

The functions had no symbols at first and they all looked like **FUN_XXXXXXXX**.
By studying then, it became clear that this one was main.
(and so I renamed it)

```
undefined8 main(void)
{
    size_t sVar1;

    puts("Let me search ya.");
    fgets(userdata,0x2c,stdin);
    sVar1 = strlen(userdata,0x2b);
    if (sVar1 != 0x2b) {
        lose();
        /* WARNING: Subroutine does not return */
        exit(1);
    }
    flipBits();
    scrambleUserData();
    doTest();
    return 0;
}
```

It wants a string of length 0x2b (43).

I renamed all the other functions to friendly names too (so this is a bit more readable than when I started).

```
void flipBits(void)
{
    int local_c;
```

```

local_c = 0;
while (local_c < 0x2b) {
    userdata[local_c] = ~userdata[local_c];
    local_c = local_c + 1;
}
return;
}

```

```

void scrambleUserData(void)
{
    char temp;
    byte byteValue;
    int j;
    uint i;
    int k;

    j = 0;
    while (byteValue = 0, j < 0x2b) {
        i = 0;
        while (i < 8) {
            if ((byte)((byte)(1 << ((byte)i & 0x1f)) & userdata[j]) != 0) {
                byteValue = byteValue | (byte)(1 << (7 - (byte)i & 0x1f));
            }
            i = i + 1;
        }
        userdata[j] = byteValue;
        j = j + 1;
    }
    k = 0;
    while (k < 0x15) {
        temp = userdata[k];
        userdata[k] = userdata[0x2a - k];
        userdata[0x2a - k] = temp;
        k = k + 1;
    }
    return;
}

```

```

void doTest(void)
{
    int iVar1;

    iVar1 = strcmp(userdata,rawdata,0x2c);
    if (iVar1 == 0) {
        win();
    }
    else {
        lose();
    }
}

```

```

}
return;
}

```

So, you enter a 43 char string, it flips all the bits, then scrambles them in some way and THEN compares it with the rawdata.



This is just a pointer to the data. We really want 102008

DAT_00102008				XREF[2]: doTest:001011cb(*),00104010(*)
00102008	41	??	41h	A
00102009	f5	??	F5h	
0010200a	51	??	51h	Q
0010200b	d1	??	D1h	
0010200c	4d	??	4Dh	M
0010200d	61	??	61h	a
0010200e	d5	??	D5h	
0010200f	e9	??	E9h	
00102010	69	??	69h	i
00102011	89	??	89h	
00102012	19	??	19h	
00102013	dd	??	DDh	
00102014	09	??	09h	
00102015	11	??	11h	
00102016	89	??	89h	
00102017	cb	??	CBh	
00102018	9d	??	9Dh	
00102019	c9	??	C9h	
0010201a	69	??	69h	i
0010201b	f1	??	F1h	
0010201c	6d	??	6Dh	m
0010201d	d1	??	D1h	
0010201e	7d	??	7Dh	}
0010201f	89	??	89h	
00102020	d9	??	D9h	
00102021	b5	??	B5h	
00102022	59	??	59h	Y
00102023	91	??	91h	
00102024	59	??	59h	Y
00102025	b1	??	B1h	
00102026	31	??	31h	1
00102027	59	??	59h	Y

00102028 6d	??	6Dh	m
00102029 d1	??	D1h	
0010202a 8b	??	8Bh	
0010202b 21	??	21h	!
0010202c 9d	??	9Dh	
0010202d d5	??	D5h	
0010202e 3d	??	3Dh	=
0010202f 19	??	19h	
00102030 11	??	11h	
00102031 79	??	79h	y
00102032 dd	??	DDh	
00102033 00	??	00h	

Selecting this region in Ghidra and doing **rightclick->Copy Special->As Byte String** yields:

```
41 f5 51 d1 4d 61 d5 e9 69 89 19 dd 09 11 89 cb 9d c9 69 f1 6d d1 7d 89 d9 b5 59 91 59 b1 31 59 6d d1 8b 21 9d d5
3d 19 11 79 dd 00
```

Luckily both the bit flipping and the scrambling are such that running it twice returns it back to normal.

So, I could just start with the **rawdata**, unscramble it, then flip all the bits.

The scramble function does two things.

1. reverses the bits in each byte
2. reverses the bytes in the array

So, to reverse, I need to:

1. reverse the bytes in the array
2. reverse the bits in each byte

Here is the python code:

```
userdata = [0x41, 0xf5, 0x51, 0xd1, 0x4d, 0x61, 0xd5, 0xe9, 0x69, 0x89, 0x19, 0xdd, 0x09, 0x11, 0x89, 0xcb, 0x9d, 0xc9,
0x69, 0xf1, 0x6d, 0xd1, 0x7d, 0x89, 0xd9, 0xb5, 0x59, 0x91, 0x59, 0xb1, 0x31, 0x59, 0x6d, 0xd1, 0x8b, 0x21, 0x9d, 0xd5,
0x3d, 0x19, 0x11, 0x79, 0xdd]
```

```
def reverseBytesInList():
    for k in range(0, 0x15):
        temp = userdata[k]
        userdata[k] = userdata[0x2a - k]
        userdata[0x2a - k] = temp
```

```
def reverseBitsInByte():
    for j in range(0, 0x2b):
        byteValue = 0
        for i in range(0, 8):
            if (userdata[j] & (1 << i)) != 0:
                byteValue = byteValue | (1 << (7 - i))
        userdata[j] = byteValue
```

```
def flipBitsInByte():
    for j in range(0, 0x2b):
        userdata[j] = userdata[j] ^ 0xff

reverseBytesInList()
reverseBitsInByte()
flipBitsInByte()

print(''.join([chr(n) for n in userdata]))
```

DawgCTF{.tlesreveRdnAtlpilF,nwoDgnihTyMtuP}