

challenges.re Challenge #30

Challenge Author: Dennis Yurichev
Writeup by scaredandalone

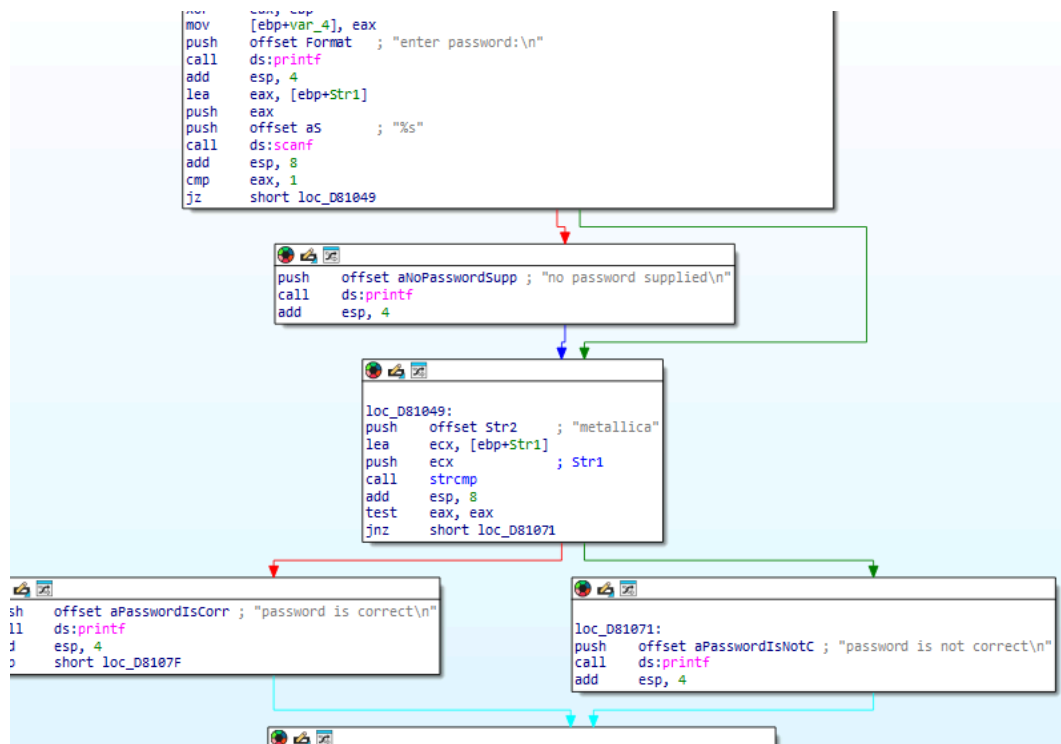
This program requires a password. Try to find it.

As an additional exercise, try to change the password by patching the executable file. Also try using one with a different length. What is the shortest possible password here?

Also try to crash the program using only string input.

Open the executable in your disassembler, I'm using IDA for this challenge.

Figure 1: IDA graph view



This is just a hard-coded password, as we can see above.

metallica

However, we are asked to patch the file, alter the length and the value of the password and try to crash the program with input.

Patching is pretty simple, we can just change the value of the Str2 variable (metallica) to a password of our choosing.

Figure 2: Password location in hex dump

```
00403020  6F 72 64 20 73 75 70 70 6C 69 65 64 0A 00 00 00  ord·supplied....
00403030  6D 65 74 61 6C 6C 69 63 61 00 00 00 70 61 73 73  metallica...pass
00403040  77 6F 72 64 20 69 73 20 63 6F 72 72 65 63 74 0A  word·is·correct.
00403050  00 00 00 00 70 61 73 73 77 6F 72 64 20 69 73 20  ....password·is·
00403060  6E 6F 74 20 63 6F 72 72 65 63 74 0A 00 00 00 00  not·correct....
00403070  4E E6 40 BB B1 19 BF 44 01 00 00 00 00 00 00 00  N.....D.....
```

Looking at the hex view, we can see where metallica is stored.

Figure 3: Changing the hex values

```
00403020  6F 72 64 20 73 75 70 70 6C 69 65 64 0A 00 00 00  ord·supplied....
00403030  6B 6F 72 6E 00 00 00 00 00 00 00 00 70 61 73 73  korn.....pass
00403040  77 6F 72 64 20 69 73 20 63 6F 72 72 65 63 74 0A  word·is·correct.
00403050  00 00 00 00 70 61 73 73 77 6F 72 64 20 69 73 20  ....password·is·
00403060  6E 6F 74 20 63 6F 72 72 65 63 74 0A 00 00 00 00  not·correct....
00403070  4E E6 40 BB B1 19 BF 44 01 00 00 00 00 00 00 00  N.....D.....
```

In this example, I am altering the value of the variable to a string that is smaller than the original. This means that we don't really need to adjust the array size, it's already aligned and the remaining bytes can just be null.

Figure 4: Running the program with patched password

```
PS C:\Users\malwarelab\Desktop > .\password1.exe
enter password:
korn
password is correct
```

The shortest possible password is 1 bytes long. This is because scanf (the function that is used to get the user input) expects non-empty input and loops until input is provided.

Figure 5: Shortest Password

```
PS C:\Users\malwarelab\Desktop > .\password1.exe
enter password:

input
password is not correct
```

As we can see, it does not allow us to put a null value, and it expects at least one char.

Now it's time to crash the program.

I first tried to overflow the buffer that stores the user input.

Figure 6: Program crash

```
PS C:\Users\malwarelab\Desktop > python -c "print('a' * 256)" | .\password1.exe
enter password:
password is not correct
FLARE-VM 08/31/2025 22:07:01
PS C:\Users\malwarelab\Desktop > Get-EventLog -LogName Application -Source "Application Error" -Newest 5
```

Index	Time	EntryType	Source	InstanceID	Message
12107	Aug 31 22:07	Error	Application Error	1000	Faulting application name: password1.exe, version: 0.0.0.0...

It looks like it worked and the program crashed. As expected.