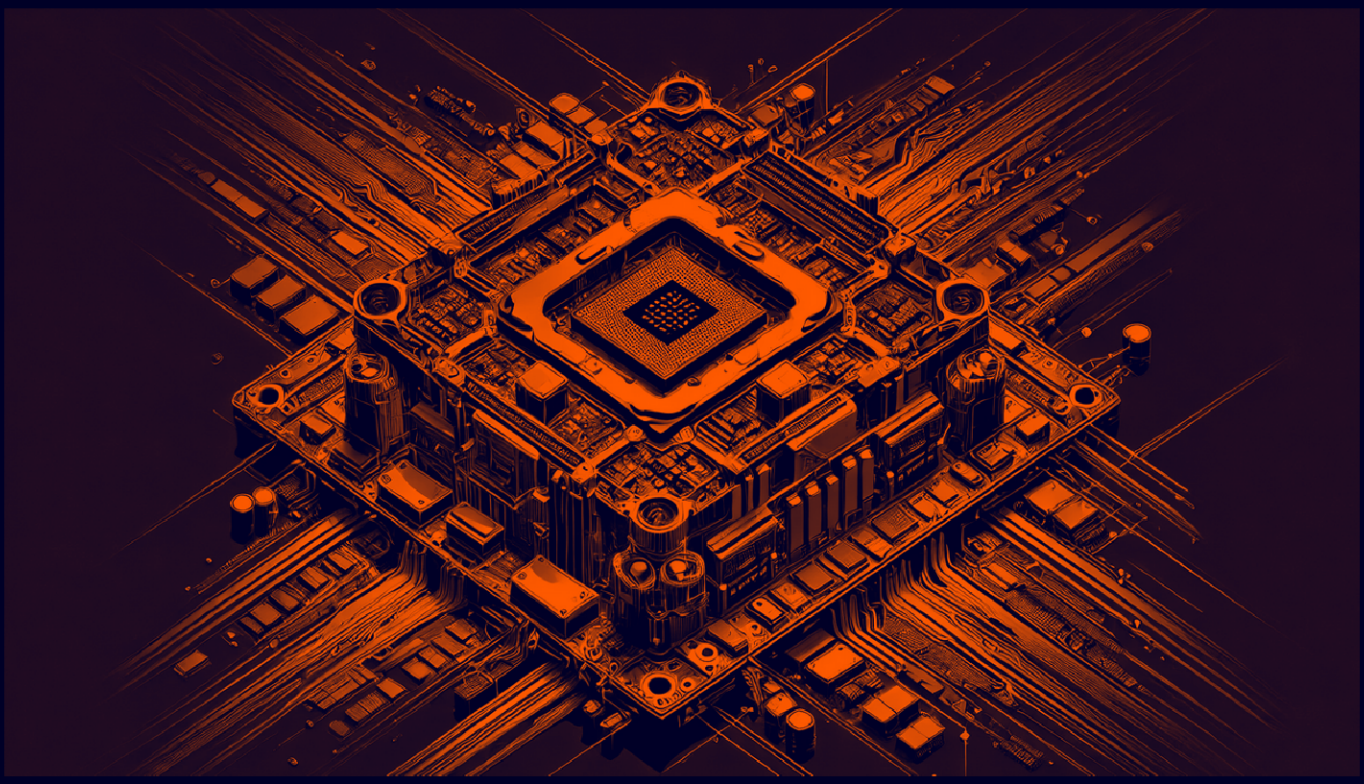
Reversing ELF

By Praveen Kumar Sharma



Crack Me - 1

So lets see what kind of file is this (Obviously ELF)

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±2 (0.026s)
ls -al crackme1

-rw-r--r-- 1 pks pks 7192 Nov  9 17:43 crackme1
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±2 (0.038s)
file crackme1

crackme1: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked,
d3c33f190c060c09b11e9ffd007f34, not stripped
```

Now lets make this executable and run it

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±2 (0.023s)
chmod +x crackme1
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±3 (0.025s)
./crackme1
flag{not_that_kind_of_elf}
```

And we get the flag

Crack Me - 2

Lets see this file now

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.026s)
ls -al crackme2
-rw-r--r-- 1 pks pks 5884 Nov  9 17:43 crackme2
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.027s)
file crackme2
crackme2: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked, 5f6b08b3c37f8feb269a60aba7, not stripped
```

Lets make this executable and run it

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.026s)
chmod +x crackme2
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±2 (0.029s)
./crackme2
Usage: ./crackme2 password
```

Now lets run strings on this binary to see if we can find something on it

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±2 (0.031s)
strings crackme2

/lib/ld-linux.so.2
libc.so.6
_IO_stdin_used
puts
printf
memset
strcmp
__libc_start_main
/usr/local/lib:$ORIGIN
__gmon_start__
GLIBC_2.0
PTRh
j3jA
[^_]
UWVS
t$,U
[^_]
Usage: %s password
super_secret_password
Access denied.
Access granted.
```

And we get the password here lets run the binary with this password

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±4 (0.025s)
./crackme2 super_secret_password

Access granted.
flag{if_i_submit_this_flag_then_i_will_get_points}
```

And we get the flag

Crack Me - 3

Lets see this file now

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.024s)
ls -al crackme3
-rw-r--r-- 1 pks pks 9632 Nov  9 17:43 crackme3

~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.027s)
file crackme3
crackme3: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked,
9f0f1a01cc543fbf5ba6204a73, stripped
```

Now lets change the permission here and run it

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.022s)
chmod +x crackme3

~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.024s)
./crackme3
Usage: ./crackme3 PASSWORD
```

Lets just run strings on this to see if we can find something

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.026s)
strings crackme3

/lib/ld-linux.so.2
__gmon_start__
libc.so.6
_IO_stdin_used
puts
strlen
malloc
stderr
fwrite
fprintf
strcmp
__libc_start_main
GLIBC_2.0
PTRh
iD$$
D$,;D$
UWVS
[^_]
Usage: %s PASSWORD
malloc failed
ZjByX3kwdXJfNWVjMG5kX2xlNTVvbl91bmJhc2U2NF80bGxfN2gzXzdoMW5nNQ==
Correct password!
Come on, even my aunt Mildred got this one!
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
```

So base64 lets decode this like so

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±4 (0.029s)
echo ZjByX3kwdXJfNWVjMG5kX2xlNTVvbl91bmJhc2U2NF80bGxfN2gzXzdoMW5nNQ== | base64 -d
f0r_y0ur_5ec0nd_le55on_unbase64_4ll_7h3_7h1ng5%
```

Now lets put this in

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±3 (0.024s)
./crackme3 f0r_y0ur_5ec0nd_le55on_unbase64_4ll_7h3_7h1ng5
Correct password!
```

So this password is our flag then

Crack Me - 4

Lets see this file here

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±3 (0.024s)
ls -al crackme4

-rw-r--r-- 1 pks pks 8740 Nov  9 17:44 crackme4
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±3 (0.027s)
file crackme4

crackme4: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked,
af334043b423ba50ec91cfa132260a, not stripped
```

Now lets make this executable and run this file

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.023s)
chmod +x crackme4
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.024s)
./crackme4

Usage : ./crackme4 password
This time the string is hidden and we used strcmp
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.024s)
./crackme4 test

password "test" not OK
```

It literally says it is using strcmp lets use `ltrace` here

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.029s)
ltrace ./crackme4 test

__libc_start_main(0x400716, 2, 0x7ffde834b278, 0x400760 <unfinished ...>
strcmp("my_m0r3_secur3_pwd", "test")
printf("password "%s" not OK\n", "test"password "test" not OK
)                                                                  = 23
+++ exited (status 0) +++
```

Lets put this password in

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.024s)
./crackme4 my_m0r3_secur3_pwd

password OK
```

So this is our flag

Crack Me - 5

Lets see this file

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.025s)
ls -al crackme5

-rw-r--r-- 1 pks pks 8968 Nov  9 17:44 crackme5
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.03s)
file crackme5

crackme5: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked,
3de8cb02f3ee4f38ee36b4ed568519, not stripped
```

Lets make this executable and run it

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.024s)
chmod +x crackme5
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±4 (2.257s)
./crackme5

Enter your input:
Hello
Always dig deeper
```

Lets run ltrace on this to see if this is comparing our input to something

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±4 (2.869s)
ltrace ./crackme5

__libc_start_main(0x400773, 1, 0x7ffe1ec3a618, 0x4008d0 <unfinished ...>
puts("Enter your input:"Enter your input:
)
__isoc99_scanf(0x400966, 0x7ffe1ec3a4a0, 0, 0x79e2cf4467a4test
)                                                 = 1
strlen("test")
strlen("test")
strlen("test")
strlen("test")
strlen("test")
strncmp("test", "OfdlDSA|3tXb32~X3tX@sX`4tXtz\342y", 28)
puts("Always dig deeper"Always dig deeper
)
+++ exited (status 0) +++
```

Lets put this in

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.814s)
./crackme5

Enter your input:
OfdlDSA|3tXb32~X3tX@sX`4tXtz
Good game
```

So this is our flag then

Crack Me - 6

Lets see this file right here

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.024s)
ls -al crackme6

-rw-r--r-- 1 pks pks 8635 Nov  9 17:44 crackme6
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.026s)
file crackme6

crackme6: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked,
9cab9f7263af75bcdbb328bda7f291, not stripped
```

Now lets make it executable then run it

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.024s)
chmod +x crackme6
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.021s)
./crackme6

Usage : ./crackme6 password
Good luck, read the source
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.026s)
./crackme6 test

password "test" not OK
```

Lets test the low hanging fruits here so here is `ltrace`

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.033s)
ltrace ./crackme6 test

__libc_start_main(0x400711, 2, 0x7ffcb93d9fa8, 0x400760 <unfinished ...>
printf("password "%s" not OK\n", "test"password "test" not OK
)                                                                 = 23
+++ exited (status 0) +++
```

And nothing in strings as well
Lets just disassemble it in ghidra

Decompile: main - (crackme6)

```
1
2 undefined8 main(int param_1,undefined8 *param_2)
3
4 {
5   if (param_1 == 2) {
6     compare_pwd((char *)param_2[1]);
7   }
8   else {
9     printf("Usage : %s password\nGood luck, read the source\n",(char *)*param_2);
10  }
11  return 0;
12 }
13
```

Lets check out his compare_pwd function

Decompile: compare_pwd - (crackme6)

```
1
2 void compare_pwd(char *param_1)
3
4 {
5   undefined8 uVar1;
6
7   uVar1 = my_secure_test(param_1);
8   if ((int)uVar1 == 0) {
9     puts("password OK");
10  }
11  else {
12    printf("password \"%s\" not OK\n",param_1);
13  }
14  return;
15 }
16
```

I guess lets see this my_secure_test function here

Decompile: my_secure_test - (crackme6)

```
1
2 undefined8 my_secure_test(char *param_1)
3
4 {
5   undefined8 uVar1;
6
7   if ((*param_1 == '\0') || (*param_1 != '1')) {
8     uVar1 = 0xffffffff;
9   }
10  else if ((param_1[1] == '\0') || (param_1[1] != '3')) {
11    uVar1 = 0xffffffff;
12  }
13  else if ((param_1[2] == '\0') || (param_1[2] != '3')) {
14    uVar1 = 0xffffffff;
15  }
16  else if ((param_1[3] == '\0') || (param_1[3] != '7')) {
17    uVar1 = 0xffffffff;
18  }
19  else if ((param_1[4] == '\0') || (param_1[4] != '_')) {
20    uVar1 = 0xffffffff;
21  }
22  else if ((param_1[5] == '\0') || (param_1[5] != 'p')) {
23    uVar1 = 0xffffffff;
24  }
25  else if ((param_1[6] == '\0') || (param_1[6] != 'w')) {
26    uVar1 = 0xffffffff;
27  }
28  else if ((param_1[7] == '\0') || (param_1[7] != 'd')) {
29    uVar1 = 0xffffffff;
30  }
31  else if (param_1[8] == '\0') {
32    uVar1 = 0;
33  }
34  else {
35    uVar1 = 0xffffffff;
36  }
37  return uVar1;
38 }
39
```

So password is : `1337_pwd` lets put this in

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.024s)
./crackme6 1337_pwd
password OK
```

So this is our flag here

Crack Me - 7

Lets see this file right here

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.023s)
ls -al crackme7
-rw-r--r-- 1 pks pks 6372 Nov  9 17:45 crackme7

~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.027s)
file crackme7
crackme7: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked,
7b0bef16a7c03f8fa49c4a39e7, not stripped
```

Lets make it executable and run it

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.024s)
chmod +x crackme7
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±4 (17.79s)
./crackme7
Menu:

[1] Say hello
[2] Add numbers
[3] Quit

[>] 1
What is your name? Fakechips
Hello, Fakechips!
Menu:

[1] Say hello
[2] Add numbers
[3] Quit

[>] 2
Enter first number: 10
Enter second number: 20
10 + 20 = 30
Menu:

[1] Say hello
[2] Add numbers
[3] Quit

[>] 3
Goodbye!
```

Skipping `strings` and `ltrace` here lets just disassemble this in ghidra

I changed the variable names a bit so its easier to understand

```
    if (input != 1) break;
    printf("What is your name? ");
    puVar1 = local_80;
    for (i = 25; i != 0; i = i + -1) {
      *puVar1 = 0;
      puVar1 = puVar1 + (uint)bVar2 * -2 + 1;
    }
    i = __isoc99_scanf(&DAT_0804883a,local_80);
    if (i != 1) {
      puts("Unable to read name!");
      return 1;
    }
    printf("Hello, %s!\n",(char *)local_80);
  }
  if (input != 2) {
    if (input == 3) {
      puts("Goodbye!");
    }
    else if (input == 0x7a69) {
      puts("Wow such h4x0r!");
      giveFlag();
    }
```

So we gotta put in 0x7a69 or 31337 as the input to run this giveFlag() function, Im not gonna bother with this function rn lets just run this with this new input

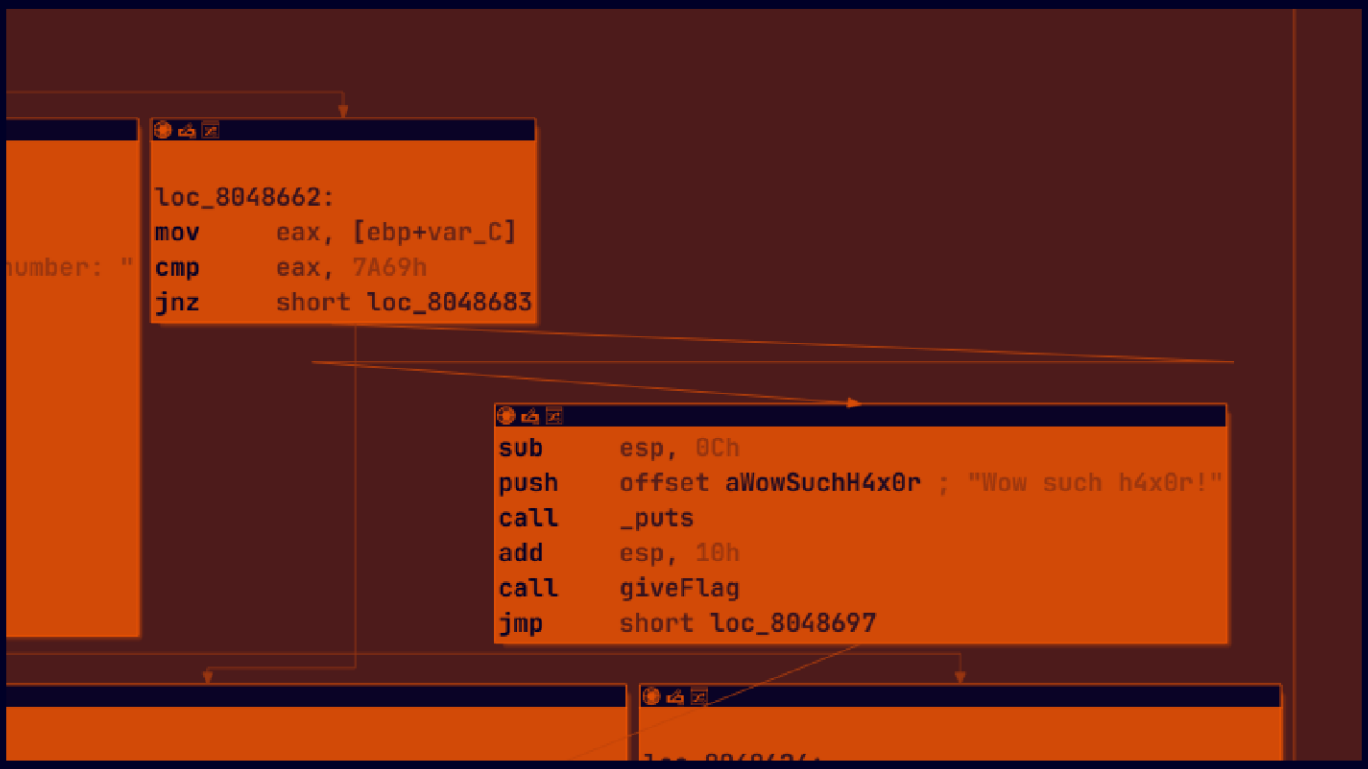
```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±3 (7.079s)
./crackme7

Menu:

[1] Say hello
[2] Add numbers
[3] Quit

[>] 31337
Wow such h4x0r!
flag{much_reversing_very_ida_wow}
```

Here is the flag, also yes we could've use ida for this
Im just gonna show how to do this in ida as well



We can change this last comparison to a `jz` instead of `jnz` to get the flag if we put a number that is not recognized cuz we trigger the block that says "unknown choice" and we are replacing that with this block to get us the flag

U can change the opcode to do that same right here if u like im not gonna do it but if u want to u can here

```
08048640  83 C4 10 E9 85 FE FF FF  8B 45 F4 83 F8 03 75 12  .........E....u.
08048650  83 EC 0C 68 B3 88 04 08  E8 13 FD FF FF 83 C4 10  ................
08048660  EB 35 8B 45 F4 3D 69 7A  00 00 75 17 83 EC 0C 68  ...E.....u.....
08048670  BC 88 04 08 E8 F7 FC FF  FF 83 C4 10 E8 25 00 00  ................
08048680  00 EB 14 8B 45 F4 83 EC  08 50 68 CC 88 04 08 E8  ....E....Ph.....
08048690  CC FC FF FF 83 C4 10 B8  00 00 00 00 8D 65 F8 59  .............e.Y
```

Change this 75 to 74 to convert this to a `jz` and make sure NOT to change the 17 cuz that is the jumping distance after the conditional jump

Anyway! Hope u enojoyed that tangent lets move on

Crack Me - 8

Lets see this file here

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±6 (0.023s)
ls -al crackme8

-rw-r--r-- 1 pks pks 5884 Nov  9 17:45 crackme8
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±7 (0.025s)
file crackme8

crackme8: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked,
ed0d08870ac523f9f3f8925a40, not stripped
```

Now lets make this file executable and run it

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±7 (0.023s)
chmod +x crackme8
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±9 (0.024s)
./crackme8

Usage: ./crackme8 password
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±9 (0.026s)
./crackme8 test

Access denied.
```

Skipping `ltrace` and `strings` again lets just disassemble this in ghidra

Decompile: main - (crackme8)

```
1
2 undefined4 main(int param_1,undefined4 *param_2)
3
4 {
5   undefined4 uVar1;
6   int iVar2;
7
8   if (param_1 == 2) {
9     iVar2 = atoi((char *)param_2[1]);
10    if (iVar2 == -0x35010ff3) {
11      puts("Access granted.");
12      giveFlag();
13      uVar1 = 0;
14    }
15    else {
16      puts("Access denied.");
17      uVar1 = 1;
18    }
19  }
20  else {
21    printf("Usage: %s password\n",(char *)*param_2);
22    uVar1 = 1;
23  }
24  return uVar1;
25 }
26
```

Pretty simple here it is just running atoi() function on our input basically, atoi converts numeric string to an integer value

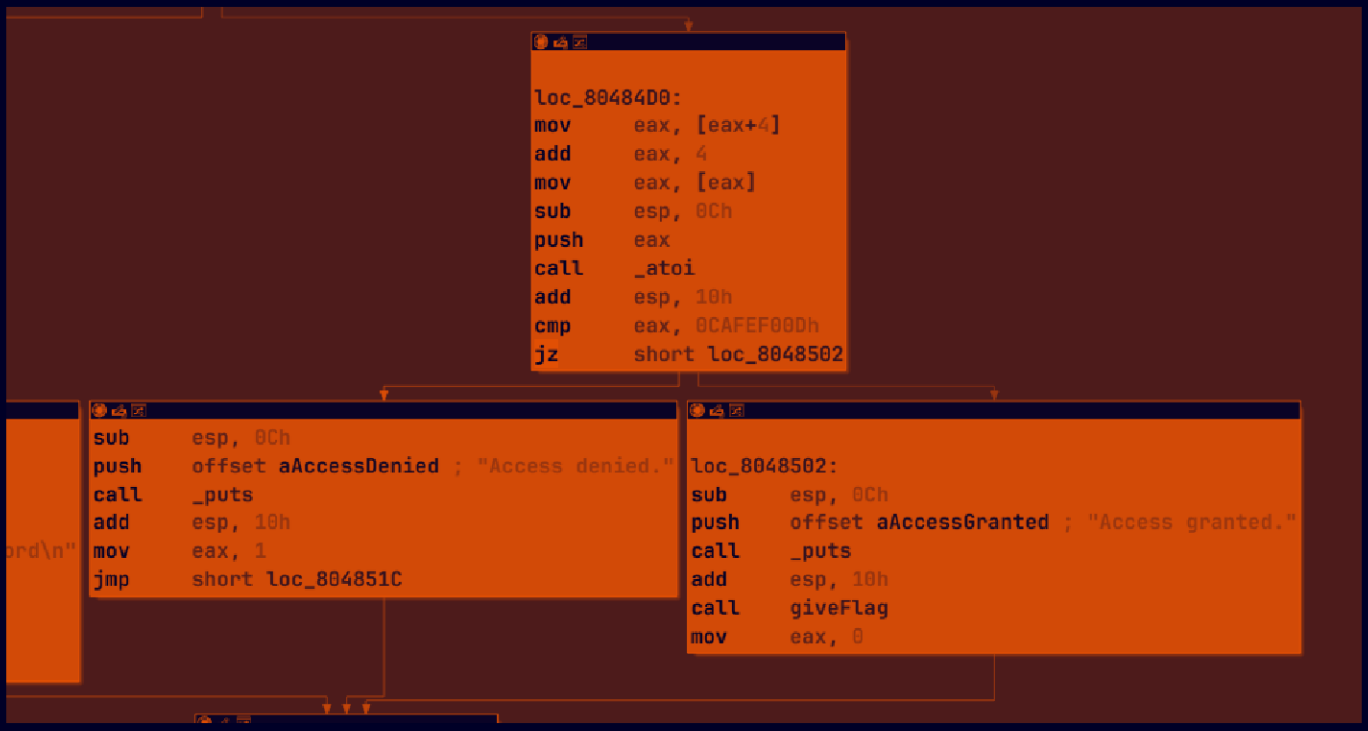
So this `-0x35010ff3` is our answer right here lets just convert this in ghidra

Decompile: main - (crackme8)

```
1
2 undefined4 main(int param_1,undefined4 *param_2)
3
4 {
5   undefined4 uVar1;
6   int iVar2;
7
8   if (param_1 == 2) {
9     iVar2 = atoi((char *)param_2[1]);
10    if (iVar2 == -889262067) {
11      puts("Access granted.");
12      giveFlag();
13      uVar1 = 0;
14    }
15    else {
16      puts("Access denied.");
17      uVar1 = 1;
18    }
19  }
20  else {
21    printf("Usage: %s password\n",(char *)*param_2);
22    uVar1 = 1;
23  }
24  return uVar1;
25 }
26
```

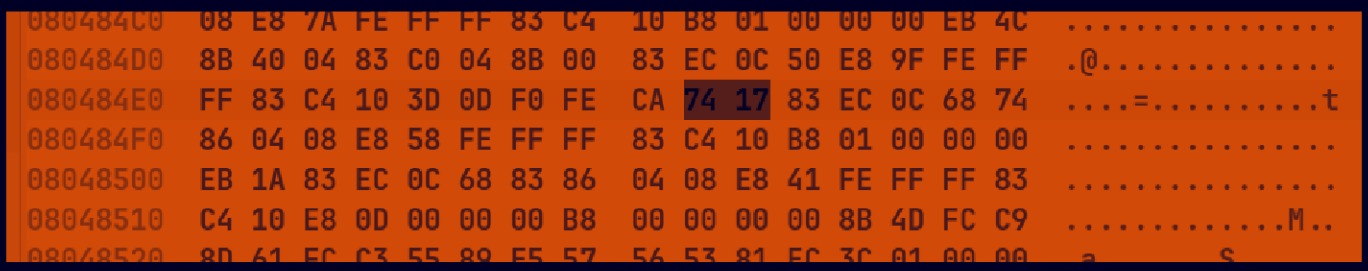
If u put this in u should have your flag but i want to show how to do this in ida as well lets do it

So loading this in ida



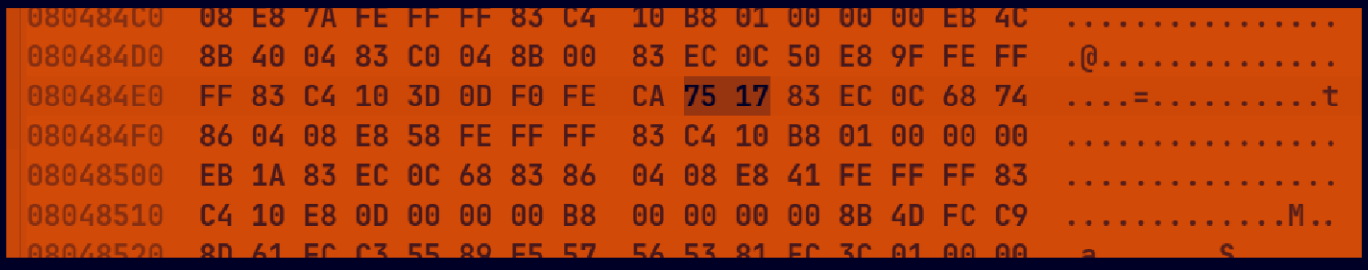
So this `jz` is deciding that if this is granted or not so we need to just reverse this

So click on the jz and go to `Hex view-1` tab

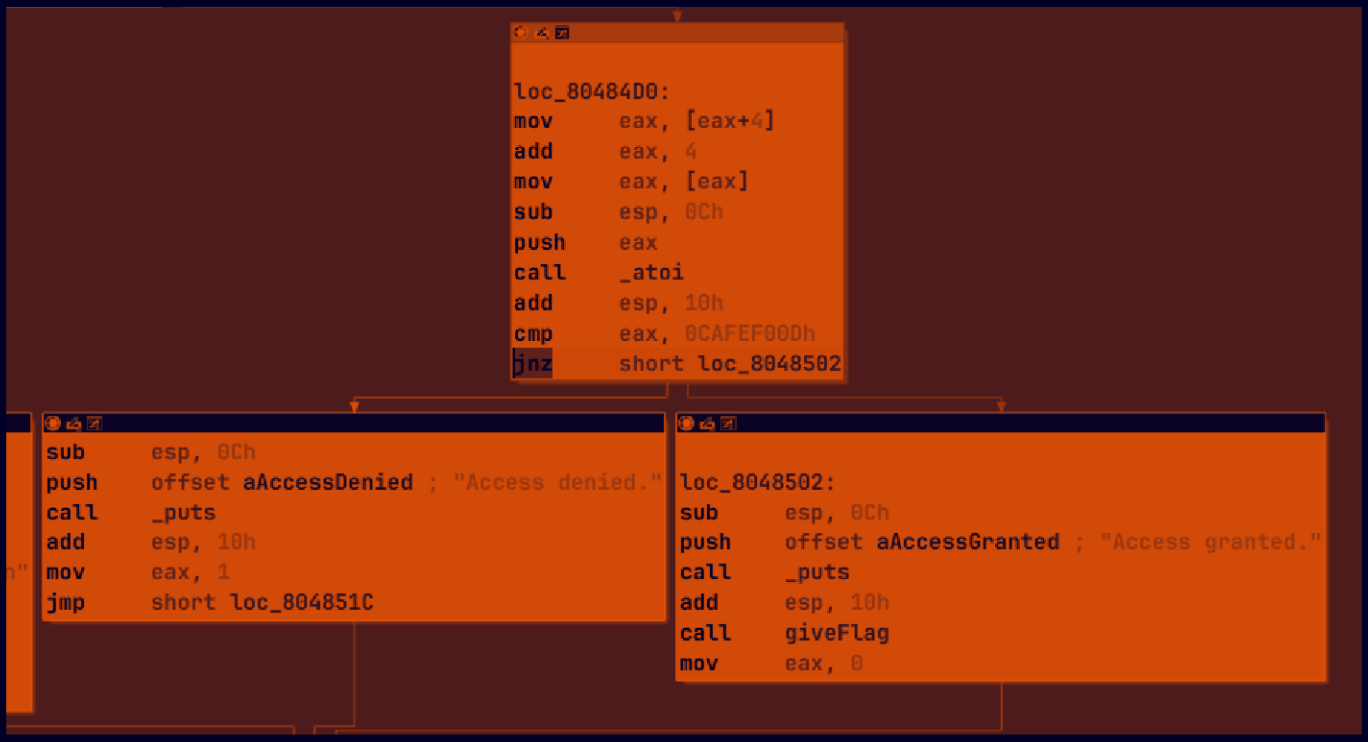


So we need to change this 74 to 74 to convert this `jz` to a `jnz`

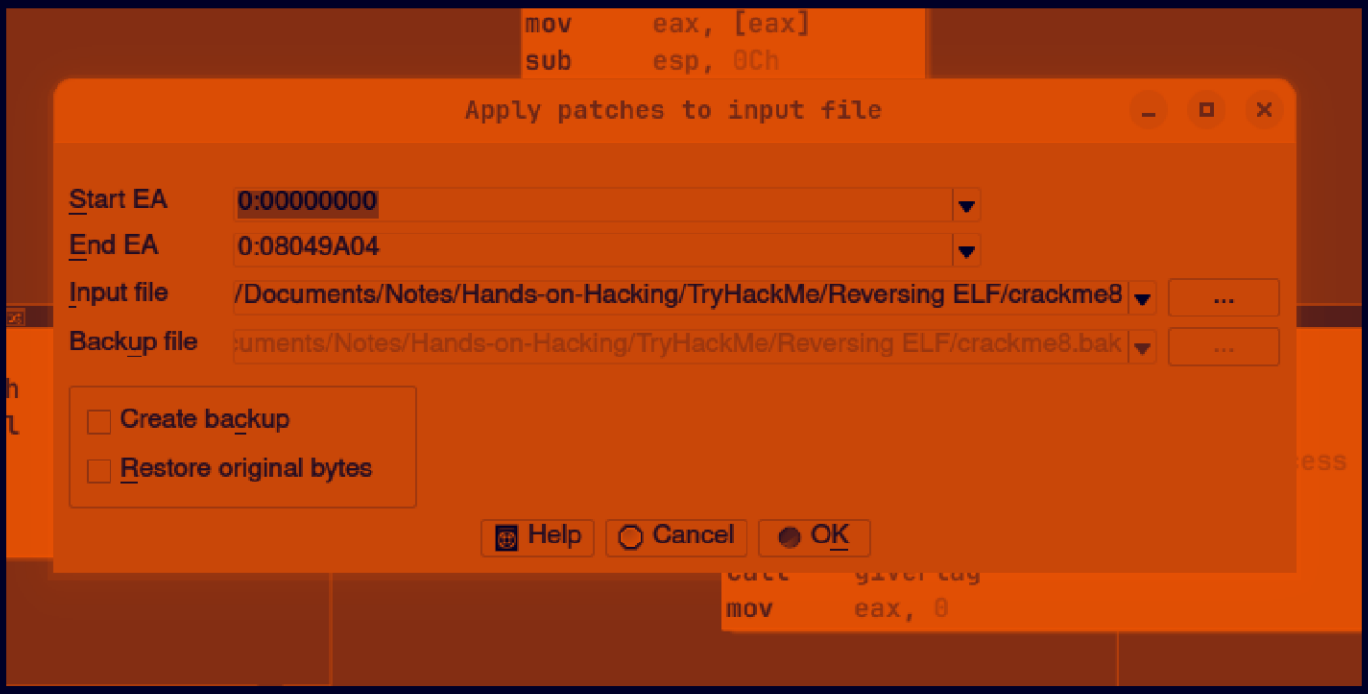
- Here hit right click then hit edit
- Change the 4 to 5 by just overwriting it then hit F2 to save it



This is not saved to the binary yet we'll do it next but lets verify if the instruction flip



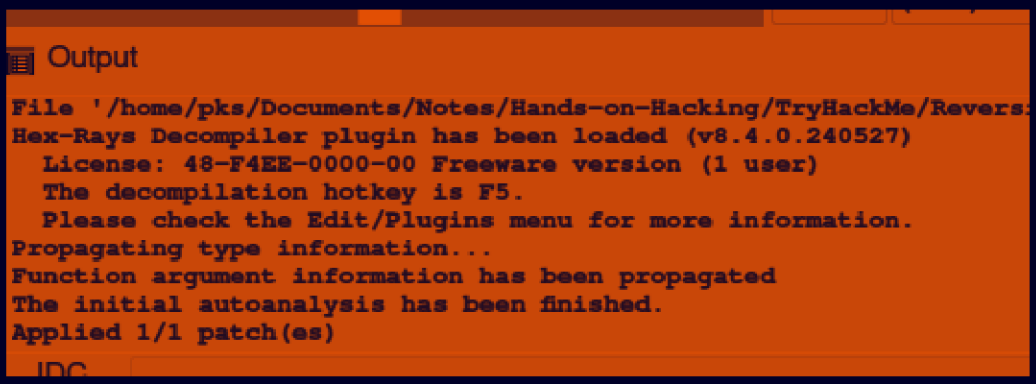
And it did change
To save this change to the binary hit go to : Edit(Top-Left) → Patch Program → Apply patches to input file



I do recommed that u make a backup file but in not gonna do it cuz im cool like that 😎

So just hit OK

Now make sure there is no it says applied patches in the bottom console



Now we can just run the binary with some random input (Tested it with both number and text and both will work)

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±1 (0.028s)
./crackme8 123

Access granted.
flag{at_least_this_cafe_wont_leak_your_credit_card_numbers}
```

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.024s)
./crackme8 test

Access granted.
flag{at_least_this_cafe_wont_leak_your_credit_card_numbers}
```

Changed this crackme8 binary name to test the original number we found in ghidra

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±7 (0.022s)
mv crackme8 crackme8.patched
```

I got a new copy of binary here

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main)±8 (0.023s)
ls -al

total 96
drwxr-xr-x 1 pks pks  192 Nov  9 19:14  .
drwxr-xr-x 1 pks pks  690 Nov  9 17:41  ..
-rwxr-xr-x 1 pks pks 7192 Nov  9 17:43  crackme1
-rwxr-xr-x 1 pks pks 5884 Nov  9 17:43  crackme2
-rwxr-xr-x 1 pks pks 9632 Nov  9 17:43  crackme3
-rwxr-xr-x 1 pks pks 8740 Nov  9 17:44  crackme4
-rwxr-xr-x 1 pks pks 8968 Nov  9 17:44  crackme5
-rwxr-xr-x 1 pks pks 8635 Nov  9 17:44  crackme6
-rwxr-xr-x 1 pks pks 6372 Nov  9 17:45  crackme7
-rwxr-xr-x 1 pks pks 5884 Nov  9 19:14  crackme8
-rwxr-xr-x 1 pks pks 5884 Nov  9 19:08  crackme8.patched
-rw-r--r-- 1 pks pks 5659 Nov  9 19:12 'Reversing ELF.md'
```

Lets run this with our that number

```
~/Documents/Notes/Hands-on-Hacking/TryHackMe/Reversing ELF git:(main) (0.023s)
./crackme8 -889262067

Access granted.
flag{at_least_this_cafe_wont_leak_your_credit_card_numbers}
```

I guess that's it, Thanks for reading I guess :)