

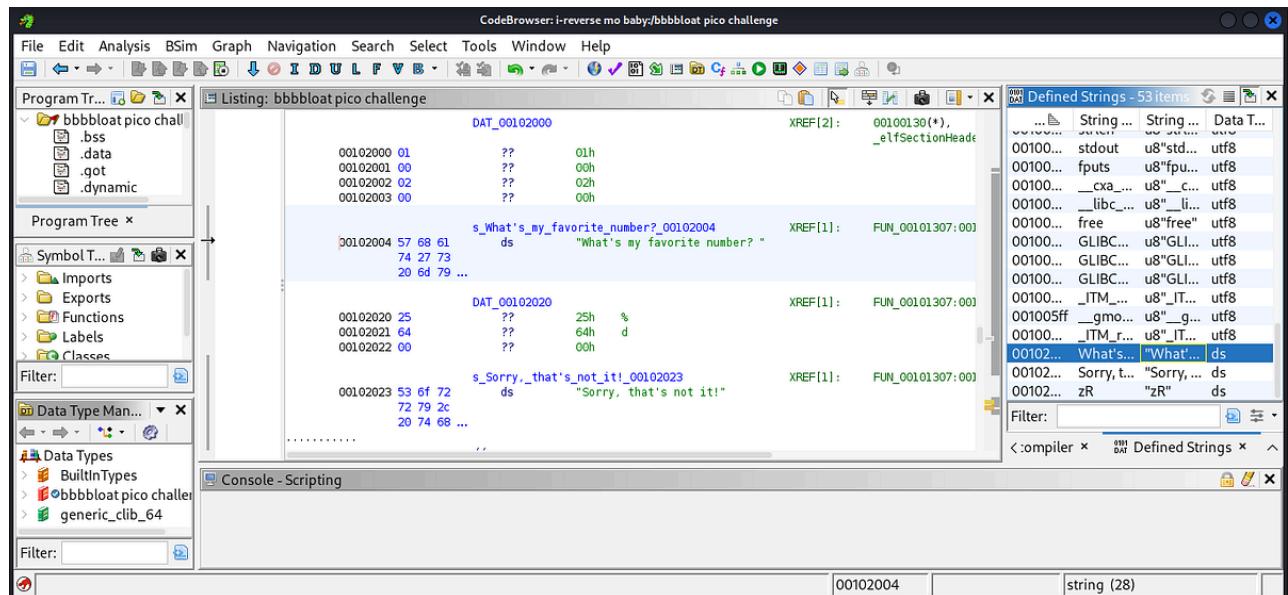
Pico-CTF : bbbbloat challenge Solution

Category: Reverse Engineering

Tool Used: Ghidra

I started by trying to understand how the program worked. Upon inspection, I noticed that the program contains this one **function**, and within it, a key piece of code stood out.

```
if(local_48 == 0x86187) {
    local_44 = 0xd2c49;
    local_40 = (char *)FUN_00101249(o, &local_38);
    fputs(local_40, stdout);
    putchar(10);
    free(local_40);
} else {
    puts("Sorry, that's not it!");
}
```



Here's what's happening: the program checks if local_48 matches the hexadecimal value 0x86187. If it does, it calls a function that unscrambles the flag and prints it. Otherwise, it simply outputs: "Sorry, that's not it!".

The screenshot shows the Ghidra interface with the following panes:

- Program Tree:** Shows the project structure with files like .bss, .data, .got, and .dynamic.
- Listing:** Displays assembly code for the function. A specific section of the assembly is highlighted in green, corresponding to the decompiled C code below.
- Decompile:** Shows the decompiled C code for the function. The highlighted assembly code corresponds to the following C code:


```

16 local_38 = 0x4c7525724034341;
17 local_30 = 0x3062396630646434;
18 local_28 = 0x65623066635f3d33;
19 local_20 = 0x4e326566623535;
20 local_44 = 0x2c49;
21 printf("What's my favorite number? ");
22 local_44 = 0x2c49;
23 __scanc(6DAT_00102020,&local_48);
24 local_44 = 0x2c49;
25 if (local_48 == 0x86187) {
26   local_44 = 0x2c49;
27   local_40 = (char *)FUN_00101249(0,&local_38);
28   fput(local_40,stdout);
29   putchar(10);
30   free(local_40);
31 } else {
32   puts("Sorry, that's not it!");
33 }
34 if (local_10 != *(long *)in_FS_OFFSET + 0x28) {
35   /* WARNING: Subroutine does not return */
      
```
- Defined Strings:** Shows the string "CMP EAX,0x86187" which is part of the assembly code.
- Console - Scripting:** An empty console window.

To get the correct input, I converted the hexadecimal value to decimal.

```
>>> hex_value = 0x86187
>>> int(hex_value)
549255
```

The terminal window shows the following session:

```

File Actions View Help
(loki@SolarisFortress)-[~]
$ python3
Python 3.13.6 (main, Aug 7 2025, 10:53:54) [GCC 14.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 0x86187
549255
>>> 
    tries to understand how the program works
    only one function in the program
    piece of code noticed

```

So, the answer to the prompt "What's my favorite number?" is 549255.

Running the program with this input reveals the flag:

Flag: picoCTF{cu7_7h3_blo47_36dd316a}

The screenshot shows a terminal window titled 'loki@SolarisFortress: ~'. The terminal has a dark background with light-colored text. It displays the following session:

```
(loki@SolarisFortress)-[~]$ ./bbbbloat\ pico\ challenge
What's my favorite number? 549255
picoCTF{cu7_7h3_b1047_36dd316a}

$ [REDACTED] to understand how the program works
$ [REDACTED] only one function in the program
$ [REDACTED] piece of code noticed
```

By [Alexander Sapo](#) on [November 15, 2025](#).

[Canonical link](#)

Exported from [Medium](#) on February 7, 2026.