**CTF crackme**

<https://www.youtube.com/watch?v=pF_prX9ZEHg>

<https://medium.com/@jeanmichel.amblat/first-steps-with-ghidra-crackme01-319827a2e80b>

[**http://www.cs.ucr.edu/~csong/cs165/17/lab1.html**](http://www.cs.ucr.edu/~csong/cs165/17/lab1.html)

<https://meowmeowxw.gitlab.io/ctf/easy-crack-me/> //Bem parecido com o nosso ctf

<https://maxkersten.nl/binary-analysis-course/assembly-basics/practical-case-crack-me-0x03/> // static e dinamic analysis

<https://medium.com/@Asm0d3us/part-2-crackmes-one-beginner-friendly-reversing-challenges-5e58a8a42e26>

**RADARE2**

<https://reverseengineering.stackexchange.com/questions/18085/radare2-find-command-line-arguments-and-location-in-stack>

**Comandos File, lreader, strings, objdump, gdb.**

<https://www.codementor.io/@packt/reverse-engineering-a-linux-executable-hello-world-rjceryk5d>

**Bufferoverflow**

* **Stack canary (ver mais sobre)**

[**https://ctf101.org/binary-exploitation/stack-canaries/**](https://ctf101.org/binary-exploitation/stack-canaries/)

[**https://www.mentebinaria.com.br/forums/topic/949-usar-buffer-overflow-para-fazer-leak-de-canary/**](https://www.mentebinaria.com.br/forums/topic/949-usar-buffer-overflow-para-fazer-leak-de-canary/)

* **Using python**

[**https://ctf101.org/binary-exploitation/buffer-overflow/**](https://ctf101.org/binary-exploitation/buffer-overflow/)

# **Pwndbg, Ghidra, Shellcode**

[**https://tc.gts3.org/cs6265/2019/tut/tut02-warmup2.html**](https://tc.gts3.org/cs6265/2019/tut/tut02-warmup2.html)

* **Using the little endian format**

[**https://bufferoverflows.net/medium-reverse-engineering-challenge-solving/**](https://bufferoverflows.net/medium-reverse-engineering-challenge-solving/)

[**https://www.tutorialspoint.com/how-do-i-convert-between-big-endian-and-little-endian-values-in-cplusplus**](https://www.tutorialspoint.com/how-do-i-convert-between-big-endian-and-little-endian-values-in-cplusplus)

* **Root me challenges**

[**https://dev.to/denisnutiu/root-me-exploitme-basic-buffer-overflow-4l1e**](https://dev.to/denisnutiu/root-me-exploitme-basic-buffer-overflow-4l1e)

[**https://github.com/s1syphu5/RootMe-Challenges**](https://github.com/s1syphu5/RootMe-Challenges)

**Documentação:**

**./Zodiac\_linux**

**sudo chmod +x ./Zodiac\_linux**

**Fazer o executavel**

strings willucrackme

**Mostra as strings dentro do ELF file**

gdb ./willucrackme

**analisar o codigo em assemble**

* **disas main / disas strcpy**
* **delete (para tirar o break)**
* **r (start the program)**
* **x/10i 0x187 (Mostra o codigo)**

**ltrace**

**---------------------------------------------------------------------------------------------------------**

1. **Reading the whole binary first (e.g., try objdump -d crackme0x00);**
2. **Starting with a gdb session (e.g., gdb ./crackme0x00) and setting a breakpoint on a well-known entry (e.g., luckily main() is exposed, try nm crackme0x00(Aparece os hexa e as funções na frente));**

**Endereço da main 0000000000001165 T main**

* **file willcrackme**

**-----------------------------------------------------------------------------------------------------------------------**

**Estudar esse código**

[**https://stackoverflow.com/questions/13487821/the-use-of-char/13487867**](https://stackoverflow.com/questions/13487821/the-use-of-char/13487867)

**int main(int iParm1,long lParm2)**

**{**

**char local\_58 [76];**

**uint local\_c;**

**if (iParm1 == 1) {**

**errx(1,"Por favor, argumente\n");**

**}**

**local\_c = 0;**

**y.3618 = 5;**

**z.3619 = 0xf;**

**strcpy(local\_58,\*(char \*\*)(lParm2 + 8));**

**if (z.3619 \* local\_c \* y.3618 == 0x2bffff5b) {**

**puts("Você escreveu o valor correto na variavel, parabens!");**

**}**

**else {**

**printf("Errrrrrrrrrrrou, 0x%08x\n",(ulong)local\_c);**

**}**

**return 0;**

**}**

**-----------------------------------------------------------------------------------------------------------------------------**

**int main(int iParm1,long lParm2)**

**{**

**char local\_58 [76];**

**long int local\_c = 9842631.19;**

**unsigned char y\_3618 = 5;**

**unsigned char z\_3619 = 0xf;**

**int sim = z\_3619 \* y\_3618 \* local\_c;**

**printf("value of a: %X\n",sim);**

**if (z\_3619 \* local\_c \* y\_3618 == 0x2bffff5b) {**

**puts("Você escreveu o valor correto na variavel, parabens!");**

**}**

**else {**

**printf("Errrrrrrrrrrrou, 0x%08x\n",local\_c);**

**}**

**return 0;**

**}**