github.com

CTFs/2019_picoCTF/reverse_cipher.md at master · Dvd848/CTFs

~3 minutes

Reverse Engineering, 300 points

Description:

We have recovered a binary and a text file. Can you reverse the flag.

Solution:

Let's inspect the attached files:

```
root@kali:/media/sf_CTFs/pico/reverse_cipher# file rev
rev: ELF 64-bit LSB pie executable x86-64, version 1 (SYSV),
dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux
3.2.0, BuildID[sha1]=523d51973c11197605c76f84d4afb0fe9e59338c,
not stripped
root@kali:/media/sf_CTFs/pico/reverse_cipher# file rev_this
rev_this: ASCII text, with no line terminators
root@kali:/media/sf_CTFs/pico/reverse_cipher# cat rev_this
picoCTF{w1{1wq84>654f26}
```

In order to understand `rev`'s logic, let's check the main function using Ghidra's decompiler:

```
void main(void)
{
 FILE *flag_stream;
 FILE *output_stream;
 size_t num_elements_read;
 char flag_buf [24];
 int j;
 int i;
 char c;

 flag_stream = fopen("flag.txt","r");
 output_stream = fopen("rev_this","a");
 if (flag_stream == (FILE *)0x0) {
  puts("No flag found, please make sure this is run on the server");
 }
 if (output_stream == (FILE *)0x0) {
  puts("please run this on the server");
 }
 num_elements_read = fread(flag_buf,24,1,flag_stream);
 if ((int)num_elements_read < 1) {
          /* WARNING: Subroutine does not return */
  exit(0);
 }
 i = 0;
 while (i < 8) {
  fputc((int)flag_buf[i],output_stream);
  i = i + 1;
 }
 j = 8;
 while (j < 23) {
  if ((j & 1U) == 0) {
      c = flag_buf[j] + '\x05';
    }
    else {
      c = flag_buf[j] + -2;
    }
    fputc((int)c,output_stream);
    j = j + 1;
  }
  fputc((int)flag_buf[23],output_stream);
  fclose(output_stream);
  fclose(flag_stream);
  return;
}
```

We can easily reverse the encryption logic using the following script:

```
import os
import mmap

def memory_map(filename, access=mmap.ACCESS_READ):
    size = os.path.getsize(filename)
    fd = os.open(filename, os.O_RDONLY)
    return mmap.mmap(fd, size, access=access)

with memory_map("rev_this") as bin_file:
    for i in range(8):
        print(chr(bin_file[i]), end = '')
    for i in range(8, 23):
        if (i & 1) == 0:
            print(chr(bin_file[i] - 5), end = '')
        else:
            print(chr(bin_file[i] + 2), end = '')
    print (chr(bin_file[23]))
```

Output:

```
root@kali:/media/sf_CTFs/pico/reverse_cipher# python3 solve.py
picoCTF{r3v3rs369806a41}
```