Computer Security hw0 Write Up

陳鴻智 b04901025 ID:b04901025

October 2018

Buffer overflow(Pwn)

In order to exploit the binary, first we'll use objdump. Running this we would get the entire assembly code of the dms file. We found that there is a function named "hidden", apparently it must be the target function we want to execute. from main function:

```
40057f: 48 8d 45 f0           lea    rax,[rbp-0x10]
```

We found that 0x10 (in hex) bytes are preserved for buffer. And because the machine is 64 bit, the next 8 bytes are reserved for %ebp, and the next 8 bytes store the return address (the addreass for %eip to jump).

Here's our payload: the first 16+8 bytes would be any random characters, and the next four bytes would be the address of the hidden function(0x400566):

```
0000000: 6161 6161 6161 6161 6161 6161 6161 6161  aaaaaaaaaaaaaaaa
0000010: 6161 6161 6161 6161 6605 4000 0000 0000  aaaaaaaaf.@.....
```

Then, we'll use pwntools to send the payload to the server.

Pusheeeeen(Web)

Open the website with Safari. When I used the mouse to click, I found that the original link url was

```
http://kaibro.tw/hw0/hw0.php
```

However, after I clicked, the url became

```
http://kaibro.tw/hw0/hw0-0.php
```

I think maybe there are redirections. Then I used OWASP ZAP to set break on all requests and responses, starting from hw0.php. I checked all responses, and found the flag from the response of

```
http://kaibro.tw/hw0/kaibro_big_gg.php
```

MdRsRcXt(Crypto)

After tracing the code, several steps are summarized:

1. md5 hash.

2. XOR.

3. RSA

So We use a reverse procedure to get the flag.

babystego(Misc)

In this section, we use zsteg to exploit the picture.

```
b1,b,lsb,xy          .. file: MPEG ADTS, layer III, v2,  64 kbps, 24 kHz, Monaural
```

Found that there is an ADTS file, extract it using zsteg command. Then, open the sound file. After I played many times, I still can't recognize where's the flag. Then I use Audacity to reverse the file, it works! The file contains a lot of numbers. By using the online hex to ascii converter, we caught the flag.

notbabyjava(Rev)

First, unzip the jar file. Open the Main.class file, it said:

```
Welcome to our homework 0 reverse challenge.
If you see this message in error, then you are the right track.
Read it *CAREFULLY* to figure out what's wrong.
```

Then we use the java decompiler ”jad” to decompile the Main.class file. In the jad file, we can see the source code. Just reverse the algorithm(note that the range of ascii is 0-255)