第九届山东省赛科来杯:简单的JS、左上角的秘密writeup

原创

零号程序猿 于 2020-11-09 22:52:11 发布 280 收藏

分类专栏： ctf 山东省赛

本文链接：https://blog.csdn.net/King_W_G/article/details/109587971


ctf 同时被 2 个专栏收录

7 篇文章 2 订阅

订阅专栏


山东省赛

1 篇文章 0 订阅

订阅专栏

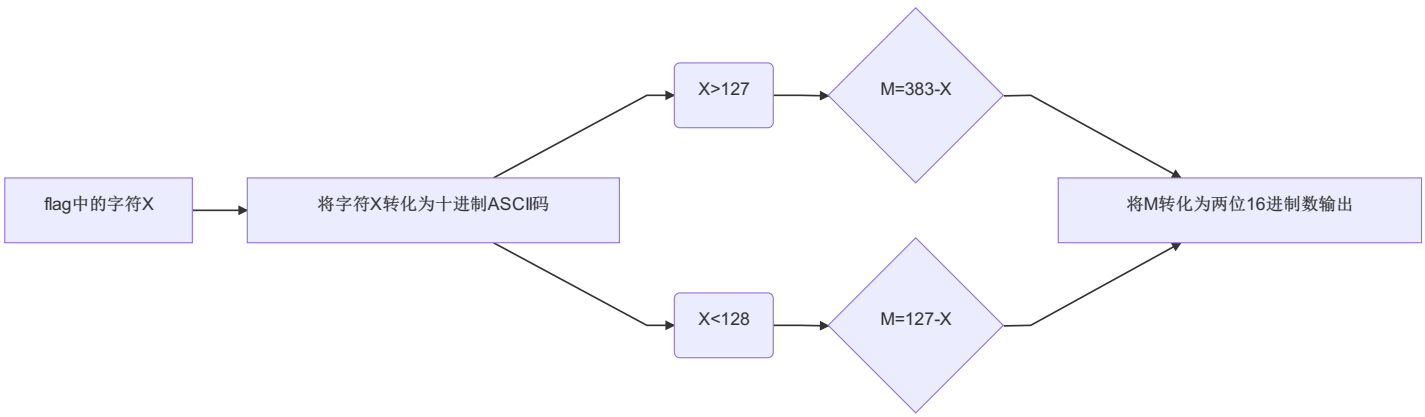
MISC

简单的JS

题目中给了一个js脚本，如下

```
function pseudoHash(string, method) {
  // Default method is encryption
  if (!('ENCRYPT' == method || 'DECRYPT' == method)) {
    method = 'ENCRYPT';
  }
  // Run algorithm with the right method
  if ('ENCRYPT' == method) {
    // Variable for output string
    var output = '';
    // Algorithm to encrypt
    for (var x = 0, y = string.length, charCode, hexCode; x < y; ++x) {
      charCode = string.charCodeAt(x);
      if (128 > charCode) {
        charCode += 128;
      } else if (127 < charCode) {
        charCode -= 128;
      }
      charCode = 255 - charCode;
      hexCode = charCode.toString(16);
      if (2 > hexCode.length) {
        hexCode = '0' + hexCode;
      }
      output += hexCode;
    }
    // Return output
    return output;
  } else if ('DECRYPT' == method) {
    // DECODE MISS
    // Return ASCII value of character
    return string;
  }
}
pseudoHash('19131e18041b1d4c47191d19194f1949481a481a1d4c1c461b4d484b191b4e474f1e4b1d4c02', 'DECRYPT');
```

这个脚本大体流程如下



简单的逆向一下，写出解题脚本

```
#f里是简单的js中的值
f = '19131e18041b1d4c47191d19194f1949481a481a1d4c1c461b4d484b191b4e474f1e4b1d4c02'
flag = ''
for h in range(0,len(f),2):
    tmp = int(f[h]+f[h+1],16)
    if (tmp<128):
        flag += chr(127-tmp)
    else:
        flag += chr(383-tmp)
print (flag)
#flag{db38fbff0f67e7eb3c9d274fd180a4b3}
```

STEGO

左上角的秘密

题目给出了一个python脚本和hex文件

```
# encoding=utf-8
flag_enc = open("flag_enc.hex", "wb")
def file_encode(flag):
    i = 1
    while True:
        byte_str = flag.read(1)
        if (byte_str == b''):
            exit()
        byte_str = hex_encode(byte_str)
        file_write(flag_enc, byte_str)
        # print(byte_str, end="")
        i = i + 1
def hex_encode(byte_str):
    tmp = int.from_bytes(byte_str, byteorder="big")
    if (tmp % 2 == 0):
        tmp = (tmp + 1) ^ 128
    else:
        tmp = (tmp - 1) ^ 128
    tmp = bytes([tmp])
    return tmp
def file_write(flag_enc, byte_str):
    flag_enc.write(byte_str)

if __name__ == '__main__':
    with open("./flag.png", "rb") as flag:
        file_encode(flag)
    flag_enc.close()
```

这个脚本将flag图片加密成为了一个16进制文件，即题目给出的16进制文件，
简单逆向，还原图片

```
flag_enc = open("flag.png", "wb")
with open("flag_enc.hex", "rb") as flag:
    while True:
        byte_str = flag.read(1)
        if(byte_str == b''):
            exit()
        tmp = int.from_bytes(byte_str, byteorder="big")
        if (tmp % 2 == 0):
            tmp = (tmp + 1) ^ 128
        else:
            tmp = (tmp - 1) ^ 128
        tmp = bytes([tmp])
        flag_enc.write(tmp)
    flag_enc.close()
```

还原出图片

将图片左上角剪下来，分析像素点，发现斜线像素中的G值隐藏信息，使用脚本分析

```python
from PIL import Image            # 导入图片处理的包
from base64 import b64decode     # 导入base64解码的包



im = Image.open('flag2.png')     # 打开图片，图片名为flag2
width = im.size[0]               # 返回图片的宽
height = im.size[1]              # 返回图片的高
pim = im.load()                  # 读取图片的像素信息
tmp = ''                         # 空字符串，接收base64
for h in range(height):          # 循环遍历每一个像素

    for w in range(width):
        # 除去白色的像素点[0][1][2]分别代表RGB三个值，只取其中的G值
        if(pim[w,h][0] == 255 & pim[w,h][1] == 255 & pim[w,h][2] == 255):
            # print(pim[w,h])   #  (R,G,B) 表示第一通道
            continue
        tmp += chr(pim[w,h][1]) # 将像素点的值(Ascii形式)转化为字符串

flag = b64decode(tmp)            # base64解这个字符串
print(flag.decode('utf-8'))      # 以utf-8的形式输出
# flag{c6e4c99a6388c5d2a9ae6ef6a843cea6}
```