[MISC]Base64隐写

原创

Weird0_ 于 2021-02-17 19:08:52 发布 355 收藏 2

隐写原理

在我之前的文章里写过Base64编码的原理：Base64编码原理及Python实现

在对长度非3的倍数的字符串进行Base64编码过程中，进行转换为二进制字串这一步骤会在末尾添加0，而解码过程中之前添加的0则会被舍弃。

而base64隐写产生的原因就在于，添加的0字符在进行base64解码时会被舍弃，这意味着在这一步骤添加的二进制值可以不全为0，这并不影响解码结果。

`Terra`这一字符串的长度为5，非3的倍数，在转为6位二进制字串时添加了两个0（红色加粗部分）。编码后的结果为`VGVycmE=`：

原字符串	T	e	r	r	a	
ASCII码	84	101	114	114	97	
8位二进制	01010100	01100101	01110010	01110010	01100001	
6位二进制	010101	000110	010101	110010	011100	100110	000100
十进制	21	6	21	50	28	38	4	
base64码值	V	G	V	y	c	m	E	=

倘若添加的二进制值不全为0，虽然会改变“=”号前最后一个字符的值，使编码后的字符串变为`VGVycmH=`。但该字符串进行Base64解码的结果依然是`Terra`：

原字符串	T	e	r	r	a	
ASCII码	84	101	114	114	97	
8位二进制	01010100	01100101	01110010	01110010	01100001	
6位二进制	010101	000110	010101	110010	011100	100110	000111
十进制	21	6	21	50	28	38	7	
base64码值	V	G	V	y	c	m	H	=

末尾有两个“=”字符的编码字符串同样如此，`Lucy`字符串正常编码应为`THVjeQ==`

原字符串	L	u	c	y		
ASCII码	76	117	99	121		
8位二进制	01001100	01110101	01100011	01111001		
6位二进制	010011	000111	010101	100011	011110	010000		
十进制	19	7	21	35	30	16		
base64码值	T	H	V	j	e	Q	=	=

修改后为`THVjeV==`，同上，进行base64解码结果依然是`Lucy`

原字符串	L		u		c		y					
ASCII码	76		117		99		121					
8位二进制	0 1 0 0 1 1 0 0		0 1 1 1 0 1 0 1		0 1 1 0 0 0 1 1		0 1 1 1 1 0 0 1					
6位二进制	0 1 0 0 1 1	0 0 0 1 1 1	0 1 0 1 0 1		1 0 0 0 1 1		0 1 1 1 1 0	0 1 0 1 0 1				
十进制	19	7	21		35		30	21				
base64码值	T	H	V		j		e	V	=		=	

若像这样对多个base64编码字符串结尾进行修改，即可隐藏更多的信息，这就是base64隐写。

常见的Base64隐写题为一个txt文本文档，内含多个经过base64编码的字符串。解码规则是将所有被修改过的base64字符串结尾的二进制值提取出来组成一个二进制串，以8位分割并转为十进制值，最终十进制对应的ASCII字符串即为base64隐写结果。

解密代码

这个是网上找到的可用的解题代码，但是是python2版本的：

```
def get_base64_diff_value(s1, s2):
    base64chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'
    res = 0
    for i in xrange(len(s2)):
        if s1[i] != s2[i]:
            return abs(base64chars.index(s1[i]) - base64chars.index(s2[i]))
    return res


def solve_stego():
    with open('123.txt', 'rb') as f:
        file_lines = f.readlines()
        bin_str = ''
        for line in file_lines:
            steg_line = line.replace('\n', '')
            norm_line = line.replace('\n', '').decode('base64').encode('base64').replace('\n', '')
            diff = get_base64_diff_value(steg_line, norm_line)
            print diff
            pads_num = steg_line.count('=')
            if diff:
                bin_str += bin(diff)[2:].zfill(pads_num * 2)
            else:
                bin_str += '0' * pads_num * 2
            print goflag(bin_str)


def goflag(bin_str):
    res_str = ''
    for i in xrange(0, len(bin_str), 8):
        res_str += chr(int(bin_str[i:i + 8], 2))
    return res_str


if __name__ == '__main__':
    solve_stego()
```

在以上代码的基础上修改成了python3可用版本：

```python
# base64隐写
import base64
def get_diff(s1, s2):
    base64chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'
    res = 0
    for i in range(len(s2)):
        if s1[i] != s2[i]:
            return abs(base64chars.index(s1[i]) - base64chars.index(s2[i]))
    return res


def b64_stego_decode():
    file = open("flag.txt","rb")
    x = ''                                      # x即bin_str
    lines =  file.readlines()
    for line in lines:
        l = str(line, encoding = "utf-8")
        stego = l.replace('\n','')
        #print(stego)
        realtext = base64.b64decode(l)
        #print(realtext)
        realtext = str(base64.b64encode(realtext),encoding = "utf-8")
        #print(realtext)
        diff = get_diff(stego, realtext)        # diff为隐写字串与实际字串的二进制差值
        n = stego.count('=')
        if diff:
            x += bin(diff)[2:].zfill(n*2)
        else:
            x += '0' * n*2

    i = 0
    flag = ''
    while i < len(x):
        if int(x[i:i+8],2):
            flag += chr(int(x[i:i+8],2))
        i += 8
    print(flag)

if __name__ == '__main__':
    b64_stego_decode()
```

基于个人理解的base64隐写原理编写的python3代码如下：

```
def base64value(c):
    table = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
    for i in range(64):
        if table[i] == c:
            return i
    return 0

def base64stego():
    f = open("flag.txt","rb")
    lines = f.readlines()
    x = ''
    for line in lines:
        l = str(line, encoding = "utf-8").strip()
        if l[-1] == '=':
            if l[-2] == '=':
                x += bin(base64value(l[-3]))[-4:]
            else:
                x += bin(base64value(l[-2]))[-2:]

    flag = ''
    for i in range(0, len(x), 8):
        flag += chr(int(x[i:i+8],2))
    print(flag)

if __name__ == '__main__':
    base64stego()
```

这几个脚本都是能解出BUU上的两道base64隐写题的，题目是`[ACTF新生赛2020]base64隐写`和`[GXYCTF2019]SXMgdGhpcyBiYXNlPw==`

编写代码的时候踩了一点点小坑，原本在这个代码中使用的是`replace('\n','')`来替换文本结尾的换行符，但是只能解出ACTF的题不能解出另外一道。估计是行末除了换行可能还有其他空字符/空白符，百度查了下改成了用`strip()`就解决了。

参考：

神奇的 Base64 隐写

base64数据隐写实现原理分析

[GXYCTF2019]SXMgdGhpcyBiYXNlPw== 题解