Base编码方式以及隐写原理

原创

A丶R 于 2021-12-08 16:06:42 发布 185 收藏

分类专栏： CTF-Misc 文章标签： python 网络安全 base64

本文链接：https://blog.csdn.net/weixin_51804748/article/details/121792094

前言

base编码是一种常见的编码方式，常见的有base64、base32、base16，最近在做题时碰到了一道base隐写的misc题目，于是上网学习了一下base的编码方式以及隐写原理

---

base64编码方式

Base64就是一种基于64个可打印字符来表示二进制数据的方法，具体的64个字符分别是A-Za-z0-9+/以及占位符=

索引	对应字符	索引	对应字符	索引	对应字符	索引	对应字符
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v		

14	O	31	f	48	w		
15	P	32	g	49	x		
16	Q	33	h	50	y		

base64编码首先将原字符串的每一个字符转换为8位的二进制ascii码，将所有的8位二进制数拼接后按照6个一组进行重组，再将每一组转换为10进制，去base64的索引表中寻找对应编码字符，以下图为例

字符	h								e								l							
ASCII值	104								101								108							
二进制	0	1	1	0	1	0	0	0	0	1	1	0	0	1	0	1	0	1	1	0	1	1	0	0
索引	26								6								21							
Base64编码	a								G								V							

原字符串为hel，将ascii码转化为二进制后一共3*8=24位，再6个一组，故编码后为aGVs四位字符。
但在实际过程中，并不是所有的字符串转化后长度都是6的倍数，这时就需要用0来补充未满的位数，并使用=来当作占位符，使最终的长度为8的倍数

字符	h								e															
ASCII值	104								101															
二进制	0	1	1	0	1	0	0	0	0	1	1	0	0	1	0	1	**0**	**0**						
索引	26								6								20							
Base64编码	a								G								U							

字符	h																							
ASCII值	104																							
二进制	0	1	1	0	1	0	0	0	**0**	**0**	**0**	**0**												
索引	26								0															
Base64编码	a								A				=								=			

base64解码方式

在对base64进行解码时，会将需要解码的字符串每四个分为一组（在有占位符=的情况下，待解码的内容长度总是4的倍数）
当末尾是`=`时，则将等于号去掉，并将最后一个字符的后两位二进制数去掉，剩下的二进制数长度即为8的倍数，再进行解码
当末尾是`==`时，则将两个等于号去掉，并将最后一个字符的后四位二进制数去掉，剩下的二进制数长度即为8的倍数，再进行解码

base64隐写原理

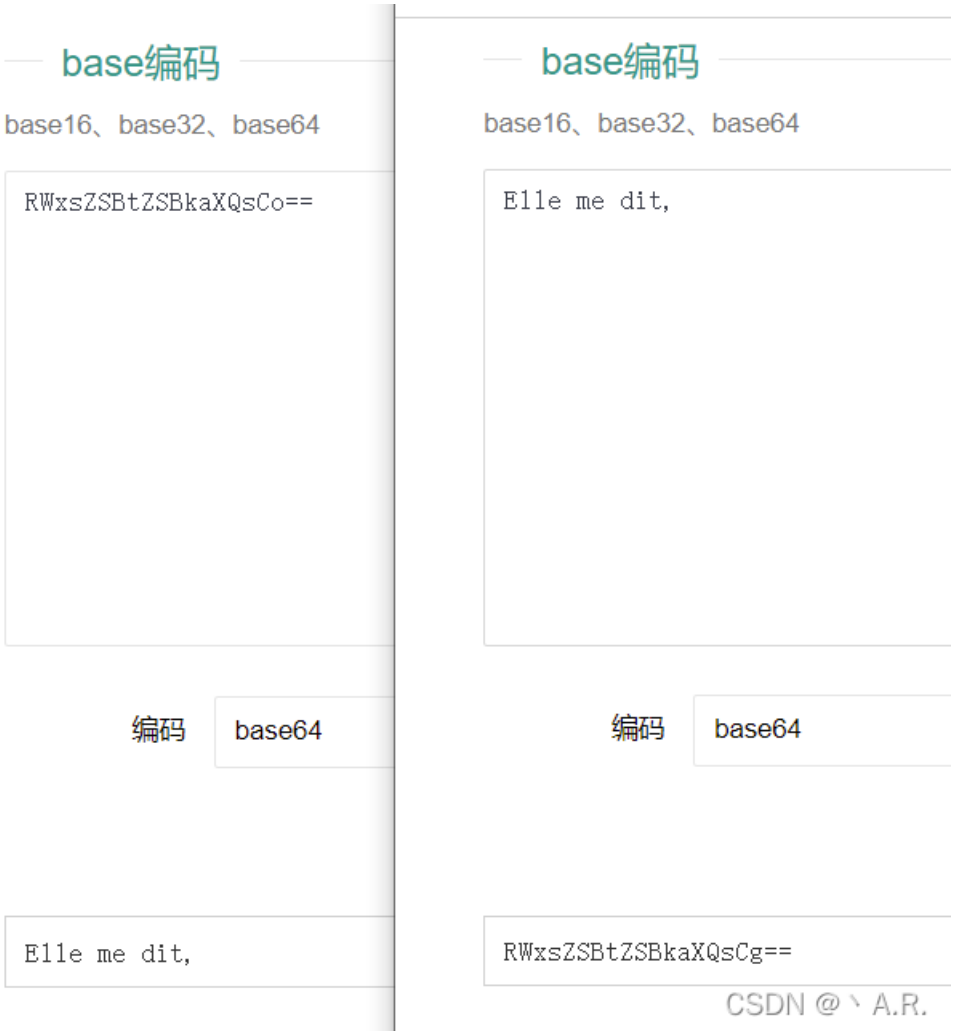
在存在占位符`=`的情况下，会将部分位的二进制数丢弃，而这些被丢弃的位中可以插入隐写的信息，即在那些用0补充的位上插入别的信息，同时又不影响base解码
可知base64编码中，末尾`=`的个数只可能是0个、1个、2个，而0个的时候无法隐写，故我们需要考虑末尾有1个或2个等号的情况

base64隐写例题：[GXYCTF2019]SXMgdGhpcyBiYXNlPw==

buu上有题目，附件可自行下载，

```
Q2V0dGUgbnVpdCwK
SW50ZW5hYmxlIGluc29tbmllLAp=
TGEgZm9saWUgbWUgZ3VldHRlLAo=
SmUgc3VpcyBjZSBxdWUgamUgZnVpcwp=
SmUgc3ViaXMsCt==
Q2V0dGUgY2Fjb3Bob25pZSwK
UXVpIG1lIHNjaWUgbGEgdOmUmnRlLAp=
QXNzb21tYW50ZSBoYXJtb25pZSwK
RWxsZSBtZSBkaXQsCo==
VHUgcGFpZXJhcyB0ZXMgZGVsaXRzLAp=
UXVvaSBxdSdpbCBhZHZpZW5uZSwK
T24gdHJh5Y2vbmUgc2VzIGNoYeWNr25lcywK
U2VzIHBlaW5lcywK
SmUgdm91ZSBtZXMgbnVpdHMsCm==
QSBsJ2Fzc2FzeW1waG9uaWUsCl==
QXV4IHJlcXVpZW1zLAr=
VHVhbnQgcGFyIGRlcGl0LAq=
Q2UgcXVlIGplIHNlbWUsCt==
SmUgdm91ZSBtZXMgbnVpdHMsCp==
QSBsJ2Fzc2FzeW1waG9uaWUsCp==
RXQgYXV4IGJsYXNwaGVtZXMsCo==
Sidhdm91ZSBqZSBtYXVkaXMsCl==
VG91cyBjZXV4IHF1aSBzJ2FpbWVudCwK
TCdlbm5lbWksCu==
VGFwaSBkYW5zIG1vbiBlc3ByaXQsCp==
RumUmnRlIG1lcyBkZWZhaXRlcywK
```


将附件中的base64编码正常解码后会得到一大堆英文语句，虽然有含义但是没有跟flag相关的信息，于是我们要考虑是否存在base64隐写，可用工具进行简单判断，把其中一句用base64解码后再编码



可以看到，解码后再编码的结果和原先不一样，则本题存在base64隐写，原本的00填充位一定隐藏了别的信息，我们可以取出所有隐写位的信息，全部拼接起来后8个一组计算10进制，再当作ascii码转化为字符串

```
import base64

table = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/'
file = open("flag.txt")
flag=''
tmpbin=''

for line in file.readlines():
    line=line.strip('\n')
    if(line[-1]=='='):
        if(line[-2]=='='):
            i=table.index(line[-3])
            b=bin(i)[2:]
            b=b.zfill(6)
            print(line)
            print(b)
            print(b[-4:]+'\n')

            tmpbin+=b[-4:]
        else:
            i = table.index(line[-2])
            b = bin(i)[2:]
            b = b.zfill(6)
            print(line)
            print(b)
            print(b[-2:]+'\n')

            tmpbin+=b[-2:]

length= len(tmpbin)/8
for i in range(int(length)):
    flag+=chr(int(tmpbin[i*8:i*8+8],2))

print(flag)
```

base32隐写

与base64隐写原理相同，通过改变填充位的0来隐藏信息，以下为base32的编码表，由于只有32个字符，故base32在编码时为5位一组来编码，末尾的`=`个数有可能为0个、1个、3个、4个、6个，下面直接给出脚本

数值	字符		数值	字符		数值	字符		数值	字符
0	A		8	I		16	Q		24	Y
1	B		9	J		17	R		25	Z
2	C		10	K		18	S		26	2
3	D		11	L		19	T		27	3
4	E		12	M		20	U		28	4
5	F		13	N		21	V		29	5
6	G		14	O		22	W		30	6
7	H		15	P		23	X		31	7
填充	=									

```
import base64

table='ABCDEFGHIJKLMNOPQRSTUVWXYZ234567'
file = open("xjj.txt")
flag=''
tmpbin=''

for line in file.readlines():
    line=line.strip('\n')
    if(line[-1]=='='):
        if(line[-3]=='='):
            if(line[-4]=='='):
                if (line[-6] == '='):
                    i=table.index(line[-7])
                    b = bin(i)[2:]
                    b = b.zfill(5)
                    tmpbin+=b[-2:]
                    print(line)
                    print(b)
                else:
                    i = table.index(line[-5])
                    b = bin(i)[2:]
                    b = b.zfill(5)
                    tmpbin += b[-4:]
                    print(line)
                    print(b)
            else:
                i = table.index(line[-4])
                b = bin(i)[2:]
                b = b.zfill(5)
                tmpbin += b[-1:]
                print(line)
                print(b)
        else:
            i = table.index(line[-2])
            b = bin(i)[2:]
            b = b.zfill(5)
            tmpbin += b[-3:]
            print(line)
            print(b)

length= len(tmpbin)/8
for i in range(int(length)):
    flag+=chr(int(tmpbin[i*8:i*8+8],2))

print(tmpbin)
print(flag)
```