Base64编码和其隐写术

原创

Shennoter伸 已于 2022-04-13 01:58:41 修改 180 收藏
分类专栏： CTF 文章标签： 安全
于 2022-04-13 01:58:20 首次发布

0x01 Base64编码

0x01 简述

Base64编码是目前应用十分广泛的一种编码方式。其作用在于将非ASCII字符转换为ASCII字符进行传输，使得消息能顺利完整地从文本通道通过。

数据在传输过程中会经过许多各式各样的网关，而且不同地区的计算机使用的文本编码可能不同，其中含有的非ASCII字符可能会在传输途中发生错误，从而导致数据出错或丢失。比如电子邮件的附件中往往含有许多非ASCII字符，为了让这些数据安全地到达目的地，人们于是使用Base64对其进行编码。

0x02 原理

假设有这样一个字符串"El Psy Kongroo"，我们根据其对应的ASCII码转换为二进制数。

> 01000101 01101100 00100000 01010000 01110011 01111001 00100000
> 01001011 01101111 01101110 01100111 01110010 01101111 01101111

转换后的二进制数若不满8位则用0补满。将得到的各二进制数拼接起来，再每6位划为一组，若不满6位则用0补满。

于是得到

> 010001 010110 110000 100000 010100 000111 001101 111001 001000 000100
> 101101 101111 011011 100110 011101 110010 011011 110110 1111***00 000000*** （斜体的0为补位用）

最后将每组二进制数分别转换为十进制数，并按照编码表转换为对应的ASCII字符。

那么这个字符串对应的Base64编码结果就是

> RWwgUHN5IEtvbmdyb28=

编号	字符	编号	字符	编号	字符	编号	字符
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

0x02 Base64隐写术

0x01 原理

我们知道，一段数据在进行Base64编码时，若最后的二进制数不满6位则需要用0补满，而在解码时这些用来填充空位的0会被舍弃，也就是说这些占位符并不会影响解码的结果。因此可以对其随意更改，解码的结果都是不变的。

再拿这个字符串为例子“**El Psy Kongroo**”，其二进制数最后两组补满后为1111***00 000000***。1111***00***在编码表中对应的字符为’8‘，***000000***对应的是 ‘**=**’，我们不能对000000进行修改，否则 ‘**=**’ 会变为其他字符。但我们可以对1111***00***斜体的部分进行修改，修改这个二进制数的最后两位只会让编码后的结果发生改变，而不会影响复原得到的原文。

例如，将1111***00***改为1111***10***，整段字符串的编码结果变成了“**RWwgUHN5lEtvbmdyb2+=**”，而对这段新得到的字符串进行Base64解码，得到的原文仍是“**El Psy Kongroo**”。

1111***01***，1111***11***同理。

从上述操作我们可以得知，当一段Base64编码结果末尾有一个 ‘**=**’ 时，这段编码有2比特可以用来隐藏数据；当末尾有两个 ‘**=**’ 时，这段编码有4比特可以用来隐藏数据；而当末尾没有 ‘**=**’ 时，这段编码无法用来隐藏数据。

这里就是可以让我们隐藏数据的好地方。我们可以将一大段文字分段进行Base64编码，然后将想要隐藏的信息转换为比特流，按顺序藏在在每段末尾有 ‘**=**’ 的编码中。

当想要复原隐写的信息时，就需要将有‘=’的编码筛选出来并进行如下操作：如果是只有一个‘=’的编码，则提取最后一个非‘=’字符对应编码表的十进制数，将其转换为二进制并提取最后两位比特；如果是有两个‘=’的编码，则提取最后一个非‘=’字符对应Base64编码表的十进制数，将其转换为二进制并提取最后四位比特。最后，将提取出来的比特拼接，再每八位划为一组，转换为十进制数后按照ASCII码表还原出隐写的信息。

0x02 示例

本例题来自于攻防世界MISC009

题目给出一个base64.txt文件，其内容为

```
U3RlZ2Fub2dyYXBoeSBpcyB0aGUgYXJ0IGFuZCBzY2llbmNlIG9m
IHdyaXRpbmcgaGlkZGVuIG1lc3NhZ2VzIGluIHN1Y2ggYSB3YXkgdGhhdCBubyBvbmV=
LCBhcGFydCBmcm9tIHRoZSBzZW5kZXIgYW5kIGludGVuZGVkIHJlY2lwaWVudCwgc3VzcGU=
Y3RzIHRoZSBleGlzdGVuY2Ugb2YgdGhlIG1lc3M=
YWdlLCBhIGZvcm0gb2Ygc2VjdXJpdHkgdGhyb3VnaCBvYnNjdXJpdHkuIFS=
aGUgd29yZCBzdGVnYW5vZ3JhcGh5IGlzIG9mIEdyZWVrIG9yaWdpbiBhbmQgbWVhbnMgImNvbmNlYW==
bGVkIHdyaXRpbmciIGZvb20gdGhlIEdyZWVrIHdvcmRzIHN0ZWdhbm9zIG1YW5pbmcgImNv
```

bGVkIHdyaXRpbmciIGZyb20gdGhlIEdyZWVrIHdvcmRzIHN0ZWdhbm9zIG1lYW5pbmcgImNv
dmVyZWQgb3IgcHJvdGVjdGVkIiwgYW5kIGdyYXBoZWluIG1lYW5pbmcgInRvIHc=
cml0ZSIuIFRoZSBmaXJzdCByZWNvcmRlZCB1c2Ugb2YgdGhlIHRlcm0gd2FzIGluIDE0OTkgYnkgSm9o
YW5uZXMgVHJpdGhlbWl1cyBpbiBoaXMgU3RlZ2Fub2dyYXBoaWEsIGEgdHJlYV==
dGlzZSBvbiBjcnlwdG9ncmFwaHkgYW5kIHN0ZWdhbm9ncmFwaHkgZGlzZ8==
dWlzZWQgYXMgYSBib29rIG9uIG1hZ2ljLiBHZW5lcmFsbHksIG1lc3P=
YWdlcyB3aWxsIGFwcGVhciB0byBiZSBzb21ldGhpbmcgZWxzZTogaW1hZ2VzLCBhcnRp
Y2xlcywgc2hvcHBpbmcgbGlzdHMsIG9yIHNvbWUgb3R=
aGVyIGNvdmVydGV4dCBhbmQsIGNsYXNzaWNhbGx5LCB0aGUgaGlkZGVuIG1lc3NhZ2UgbWF5IGJlIGluIGludmm=
c2libGUgaW5rIGJldHdlZW4gdGhlIHZpc2libGUgbGluZXMgb2YgYSBwcml2YXRlIGxldHRlci4NCg0KVGhl
IGFkdmFudGFnZSBvZiBzdGVnYW5vZ3JhcGh5LCBvdmVyIGNy
eXB0b2dyYXBoeSBhbG9uZSwgaXMgdGhhdCBtZXNzYWdlcyBkbyBub3QgYXR0cmFjdCBhdHRlbnRpb25=
IHRvIHRoZW1zZWx2ZXMuIFBsYWlubHkgdmlzaWJsZSBlbmNyeXB0ZWQgbWVzc2FnZXOXbm8gbWF0dGVyIF==
aG93IHVuYnJlYWthYmxlI3dpbGwgYXJvdXNlIHN=
dXNwaWNpb24sIGFuZCBtYXkgaW4gdGhlbXNlbHZlcyBiZSBpbmNyaW1pbmF0aW5nIP==
aW4gY291bnRyaWVzIHdoZXJlIGVuY3J5cHRpb24gaXMgaWxsZWdhbC4gVGhlcmVmb3JlLH==
IHdoZXJlYXMgY3J5cHRvZ3JhcGh5IHByb3RlY3RzIHRoZSBjb250ZW50cyBvZj==
IGEgbWVzc2FnZSwgc3RlZ2Fub2dyYXBoeSBjYW4gYmUgc2FpZCB0byBwcm90ZWN0IGJ=
b3RoIG1lc3NhZ2VzIGFuZCBjb21tdW5pY2F0aW5nIHBhcnRpZXMuDQoNCIN0ZWdhbm9ncmFwaHkgaW5jbHW=
ZGVzIHRoZSBjb25jZWFsbWVudCBvZiBpbmZvcm1hdGlvbiB3aXRoaW4gY29t
cHV0ZXIgZmlsZXMuIEluIGRpZ2I0YWwgc3RlZ2Fub2dyYXBoeSwgZWxlY3Ryb25pYyBjb21tdW5pY2F0aW9u
cyBtYXkgaW5jbHVkZSBzdGVnYW5vZ3JhcGhpYyBjb2RpbmcgaW5zaZ==
ZGUgb2YgYSB0cmFuc3BvcnQgbGF5ZXIsIHN1Y2ggYXMgYSBkb2N1bWVudCBmaWxlLCBpbWFnZSBmaWx=
ZSwgcHJvZ3JhbSBvciBwcm90b2NvbC4gTWVkaWEg
ZmlsZXMgYXJlIGlkZWFsIGZvciBzdGVnYW5vZ3JhcGhpYyB0cmFuc21pc3Npb+==
biBiZWNhdXNlIG9mIHRoZWlyIGxhcmdlIHNpemUuIEFzIB==
YSBzaW1wbGUgZXhhbXBsZSwgYSBzZW5kZXIgbWlnaHQgc3RhcnQgd2I0aCBh
biBpbm5vY3VvdXMgaW1hZ2UgZmlsZSBhbmQgYWRqdXN0IHRoZSBjb2xvciBvZiBldmVyeSAxMDB0aCBwaXhlbCD=
dG8gY29ycmVzcG9uZCB0byBhIGxldHRlciBpbiB0aGUgYWxwaGFiZXQsIGF=
IGNoYW5nZSBzbyBzdWJ0bGUgdGhhdCBzb21lb25lIG5vdCBzcGVjaWZpY2FsbHkgbG9va2luZyBm
b3IgaXQgaXMgdW5saWtlbHkgdG8gbm90aWNlIGI0Lg0KDQpUaGU=
IGZpcnN0IHJlY29yZGVkIHVzZXMgb2Ygc3RlZ2Fub2dyYXBoeSBjYW4gYmUgdHJ=
YWNlZCBiYWNrIHRvIDQ0MCBCQyB3aGVuIEhlcm9kb3R1cyBtZW50aW9ucyB0d28gZXhhbXBsZXMgb+==
ZiBzdGVnYW5vZ3JhcGh5IGIuIFRoZSBIaXN0b3JpZXMgb2Yg
SGVyb2RvdHVzLiBEZW1hcmF0dXMgc2VudCBhIHdhcm5pbmcgYWJvdXQgYSB=
Zm9ydGhjb21pbmcgYXR0YWNrIHRvIEdyZWVjZSBieSB3
cml0aW5nIGI0IGRpcmVjdGx5IG9uIHRoZSB3b29kZW4gYmFja2luZyBvZiBhIHdheCB0YWJsZXQgYmVm
b3JlIGFwcGx5aW5nIGI0cyBiZWVzd2F4IHN1cmZhY2UuIFdheCB0YWJsZXRzIHdlcmUgaW4gY29tbW9uIHVzZV==
IHRoZW4gYXMgcmV1c2FibGUgd3JpdGluZyBzdXJmYWNlcywgc29tZXRpbWX=
cyB1c2VkIGZvciBzaG9ydGhhbmQuIEFub3RoZXIgYW5jaWVudCBleGFtcGxlIGIzIHRoYXQgb9==
ZiBIaXN0aWFldXMsIHdobyBzaGF2ZWQgdGhlIGhlYWQgb2YgaGIzIG1vc3QgdHJ1c3RlZCBz
bGF2ZSBhbmQgdGF0dG9vZWQgYSBtZXNzYWdlIG9uIGI0LiBBZnRlciBoaXMgaGFpciBoYWQgZ5==
cm93biB0aGUgbWVzc2FnZSB3YXMgaGlkZGVuLiBUaGUgcHVycG9zZSB3YXMgdG+=
IGluc3RpZ2F0ZSBhIHJldm9sdCBhZ2FpbnN0IHRoZSBQZXJzaWFucy4NCg0KU3RlZ2Fub2dyYXBoeSBoYXMgYm==
ZWVuIHdpZGVseSB1c2VkLCBpbmNsdWRpbmcgaW4gcmVjZW50IGhpc3RvcmljYWwgdGltZXMgYW5kIHT=
aGUgcHJlc2VudCBkYXkuIFBvc3NpYmxlIHBlcm11dGF0aW9ucyBhcmUgZW5kbGVzcyBhbmT=
IGtub3duIGV4YW1wbGVzIGluY2x1ZGU6DQoqIEhpZGRlbiBtZXNzYWdlcyB3aXRoaW4gd2F4IHRh
YmxldHM6IGluIGFuY2llbnQgR3JlZWNlLCBwZW9wbGUgd3JvdGUgbWV=
c3NhZ2VzIG9uIHRoZSB3b29kLCB0aGVuIGNvdmVyZWQgaXQgd2I0aCB3YXggdXBvbiB3aGljaCBhbiBpbm5vY2Vu
dCBjb3ZlcmluZyBtZXNzYWdlIHdhcyB3cml0dGVu
Lg0KKiBIaWRkZW4gbWVzc2FnZXMgb24gbWVzc2VuZ2VyJ3MgYm9keTogYWxzbyB1c2VkIGluIGFuY2llbt==
dCBHcmVlY2UuIEhlcm9kb3R1cyB0ZWxscyB0aGUgc3Rvcnkgb1==
ZiBhIG1lc3NhZ2UgdGF0dG9vZWQgb24gYSBzbGF2ZSdzIHNoYXZlZCBoZWFkLCBoaWRkZW4gYnkgdGhl
IGdyb3d0aCBvZiBoaXMgaGFpciwgYW5kIGV4cG9zZWQgYnkgc2hhdmluZyBoaXMgaGVhZM==
IGFnYWluLiBUaGUgbWVzc2FnZSBhbGxlZ2VkbHkgY2FycmllZCBhIHdhcm5pbmcgdG8gR3JlZWNlIGFib5==
dXQgUGVyc2lhbiBpbnZhc2lvbiBwbGFucy4gVGh=
aXMgbWV0aG9kIGhhcyBvYnZpb3VzIGRyYXdiYWNrcyz=
IHN1Y2ggYXMgZGVsYXllZCB0cmFuc21pc3Npb24gd2hpbGUgd2FpdGluZyBmb3IgdGhlIHP=
bGF2ZSdzIGhhaXIgdG8gZ3JvdywgYW5kIHRoZSByZXN0cmljdGlvbnMgb3==
biB0aGUgbnVtYmVyIGFuZCBzaXplIG9mIG1lc3M=

YWdlcyB0aGF0IGNhbiBiZSBlbmNvZGVkIG9uIG9uZSBwZXJzb24=
J3Mgc2NhbHAuDQoqIEluIFdXSUksIHRoZSBGcmVuY2ggUmVzaXN0YW5jZSBzZW50IHNvbWUgbWVzc2FnZXMgd2==
cml0dGVuIG9uIHRoZSBiYWNrcyBvZiBjb3VyaWVycyD=
dXNpbmcgaW52aXNpYmxlIGluay4NCiogSGlkZGVuIG1lc3NhZ2VzIG9uIHBhcGVyIHdy
aXR0ZW4gaW4gc2VjcmV0IGlua3MsIHVuZGVyIG90aGVyIG1lc3NhZ2Vz
IG9yIG9uIHRoZSBibGFuayBwYXJ0cyBvZiBvdGhlct==
IG1lc3NhZ2VzLg0KKiBNZXNzYWdlcyB3cml0dGVuIGluIE1vcnNlIGNvZGUgb24ga25pdHRpbmcgeWFybiBhbmQg
dGhlbiBrbml0dGVkIGludG8gYSBwaWVjZSBvZiBjbG90aGluZyB3b3K=
biBieSBhIGNvdXJpZXIuDQoqIE1lc3NhZ2VzIHdyaXR0ZW4gb24gdGhlIGJhY2sgb5==
ZiBwb3N0YWdlIHN0YW1wcy4NCiogRHVyaW5nIGFuZCBhZnRlcm==
IFdvcmxkIFdhciBJSSwgZXNwaW9uYWdlIGFnZW50cyB1c2VkIHBob3RvZ3JhcGhpY2FsbHkgcO==
cm9kdWNlZCBtaWNyb2RvdHMgdG8gc2VuZCBpbmZvcm1hdGlvbiBiYWNrIGFuZH==
IGZvcnRoLiBNaWNyb2RvdHMgd2VyZSB0eXBpY2FsbHkg
bWludXRlLCBhcHByb3hpbWF0ZWx5IGxlc3MgdGhhbiB0aGUgc2l6ZSBvZiB0aGUgcGVyaW9kIHByb2R=
dWNlZCBieSBhIHR5cGV3cml0ZXIuIFdXSUkgbWljcm9kb3RzIG5lZWRlZCB0byBiZSBlbWJlZGRlZB==
IGluIHRoZSBwYXBlciBhbmQgY292ZXJlZCB3aXRoIGFuIGFkaGVzaXZlIChzdWNoIGFzIGNvbGxvZGlvbikuIFR=
aGlzIHdhcyByZWZsZWN0aXZlIGFuZCB0aHVzIGRldGVjdGFibGUg
Ynkgdmlld2luZyBhZ2FpbnN0IGdsYW5jaW5nIGxpZ2h0LiBBbHRlcm5hdGl2ZSB0ZWNobmlxdWVzIGluY2x1ZGVk
IGluc2VydGluZyBtaWNyb2RvdHMgaW50byBzbGl0cyBjdXQgaW50byB0aGUgZWRnZSBvZv==
IHBvc3QgY2FyZHMuDQoqIER1cmluZyBXb3JsZCBXYXIgSUksIGEgc3B5IGZvciB=
SmFwYW4gaW4gTmV3IFlvcmsgQ2l0eSwgVmVsdmFsZWW=
IERpY2tpbnNvbiwgc2VudCBpbmZvcm1hdGlvbiB0byBhY2NvbW1vZGF0aW9=
biBhZGRyZXNzZXMgaW4gbmV1dHJhbCBTb3V0aCBBbWVyaWO=
YS4gU2hlIHdhcyBhIGRlYWxlciBpbiBkb2xscywgYW5kIG==
aGVyIGxldHRlcnMgZGlzY3Vzc2VkIGhvdyBtYW55IG9mIHRoaXMgb3IgdGhhdCBkb2xs
IHRvIHNoaXAuIFRoZSBzdGVnb3RleHQgd2FzIHRoZSBkb2xsIG9yZGVycywgd2hpbGUgdGhl
IGNvbmNlYWxlZCAicGxhaW50ZXh0IiB3YXMgaXRzZWxmIGVuY2+=
ZGVkIGFuZCBnYXZlIGluZm9ybWF0aW9uIGFib3V0IHNoaXAgbW92ZW1lbnRzLF==
IGV0Yy4gSGVyIGNhc2UgYmVjYW1lIHNvbWV3aGF0IGZh
bW91cyBhbmQgc2hlIGJlY2FtZSBrbm93biBhcyB0aGX=
IERvbGwgV29tYW4uDQoqIENvbGQgV2FyIGNvdW50
ZXItcHJvcGFnYW5kYS4gSW4gMTk2OCwgY3JldyBtZW1iZW==
cnMgb2YgdGhlIFVTUyBQdWVibG8gKEFHRVItMikgaW50ZWxsaWdlbmNlIHNoaXAgaGVsZCBhcyBwcm==
aXNvbmVycyBieSBOb3J0aCBLb3JlYSwgY29tbXVuaWNhdGVkIGluIHNpZ25=
IGxhbmd1YWdlIGR1cmluZyBzdGFnZWQgcGhvdG8gb3Bwb3J0
dW5pdGllcywgaW5mb3JtaW5nIHRoZSBVbml0ZWQgU3RhdGVzIHRoZXkg
d2VyZSBub3QgZGVmZWN0b3JzIGJ1dCByYXRoZXIgd2VyZSBiZWluZyBoZWxkIGNh
cHRpdmUgYnkgdGhlIE5vcnRoIEtvcmVhbnMuIEluIG90aGVyIHBob3Rv
cyBwcmVzZW50ZWQgdG8gdGhlIFVTLCBjcmV3IG1lbWJlcnMgZ2F2ZSAidGhlIGZpbmdlciIgdG8g
dGhlIHVuc3VzcGVjdGluZyBOb3J0aCBLb3JlYW5zLCBpbiBhbiBhdHRlbXB0IHRvIE==
ZGlzY3JlZGl0IHBob3RvcyB0aGF0IHNob3dlZCB0aGVtIHNtaQ==
bGluZyBhbmQgY29tZm9ydGFibGUuDQoNCi0tDQpodHRwOi8vZW4ud2lraXBlZGlhLm9yZw==
L3dpa2kvU3RlZ2Fub2dyYXBoeQ0K

直接进行Base64解码得到的结果是这样一段文字

> Steganography is the art and science of writing hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message, a form of security through obscurity.The word steganography is of Greek origin and means “concealed writing” from the Greek words steganos meaning “covered or protected”, and graphein meaning “to write”. The first recorded use of the term was in 1499 by Johannes Trithemius in his Steganographia, a treatise on cryptography and steganography disguised as a book on magic. Generally, messages will appear to be something else: images, articles, shopping lists, or some other covertext and, classically, the hidden message may be in invisible ink between the visible lines of a private letter.

按照上一节所阐述的原理，可以写出对应的隐写复原脚本

```
#将字符按照Base64编码表转换为十进制数
def b64ord(char):
    if char in 'abcdefghijklmnopqrstuvwxyz':
        return ord(char) - 71
    elif char in 'ABCDEFGHIJKLMNOPQRSTUVWXYZ':
        return ord(char) - 65
    elif char in '0123456789':
        return ord(char) + 4
    elif char == '+':
        return 62
    elif char == '/':
        return 63
    else: return

with open('C:/Users/shenn/Desktop/base64.txt', 'r') as f:
    ostr = f.read()

info = ''
flag = ''
count = 0

for i in ostr:
    if i == '\n':
        count += 1
ostr = ostr.splitlines()

for i in range(0, count):
    tmp = ''

    if '==' in ostr[i]:
        code = b64ord(ostr[i][-3])
        tmp = str(bin(code))[-4::]
        #由于python会在二进制数前加上'0b'，所以采取以下操作去掉
        if code <= 1:
            tmp = '000' + tmp[-1]
        elif 2<= code <= 3:
            tmp = '00' + tmp[-2::]
        elif 4<= code <= 7:
            tmp = '0' + tmp[-3::]

    elif '=' in ostr[i]:
        code = b64ord(ostr[i][-2])
        tmp = str(bin(code))[-2::]
        if code <= 1:
            tmp = '0' + tmp[-1]

    info += tmp

#将得到的结果拼接起来
for i in range(0, len(info), 8):
    flag += chr(int(info[i:i+8], 2))
print(flag)
```

脚本运行结果为：

```
Base_sixty_four_point_five
```

至此，flag已到手。