

# [CTF从0到1学习] 二、CTF 密码学

原创

南岸青栀\* 于 2021-12-05 00:11:08 发布 3416 收藏 4

分类专栏: [CTF wp](#) 文章标签: [安全](#) [web安全](#) [区块链](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/qq\\_43710889/article/details/121480596](https://blog.csdn.net/qq_43710889/article/details/121480596)

版权



[CTF wp](#) 专栏收录该内容

5 篇文章 1 订阅

订阅专栏

## 密码学

### 文章目录

#### 密码学

##### 概述

##### 密码学基本概念

##### 密码学的历史与发展

第一阶段 (1949年前) 古典密码发展阶段

第二阶段: 近代密码阶段 (1949~1976)

现代密码学阶段 (1976~至今)

现代密码学的主要发展方向

##### 密码体制分类

##### 密码攻击

算法的安全性

#### 编码与密码

##### 编码基础

ASCII

unicode

BASE64

##### 古典密码学

单表代换密码

凯撒密码

移位密码

仿射密码

多表代换密码

nobhius密码 (棋盘密码)

polybius密码 / 铁盘密码 /

vigenere密码

其他类型密码

培根密码

栅栏密码

摩斯密码

CTF中奇怪密码

倒序加密

电脑键盘密码

键盘密码

手机键盘加密

当铺密码

猪圈密码

对称加密

CTF Wp

编码解码

1.AAencode

2.brainfuck

3.Ook

4.Quoted-Printable编码

5.UUencode

6.XXencode

古典密码

简单换位密码

猪圈密码

埃特巴什密码

夏多密码

当铺密码

培根密码

九宫格密码

凯撒大帝

维吉尼亚密码

Rabbit密码

Rot13密码

栅栏之困

关于同余方程、欧拉定理、乘法逆元、定义在 $Z_m$ 上的矩阵求逆

希尔密码

仿射密码

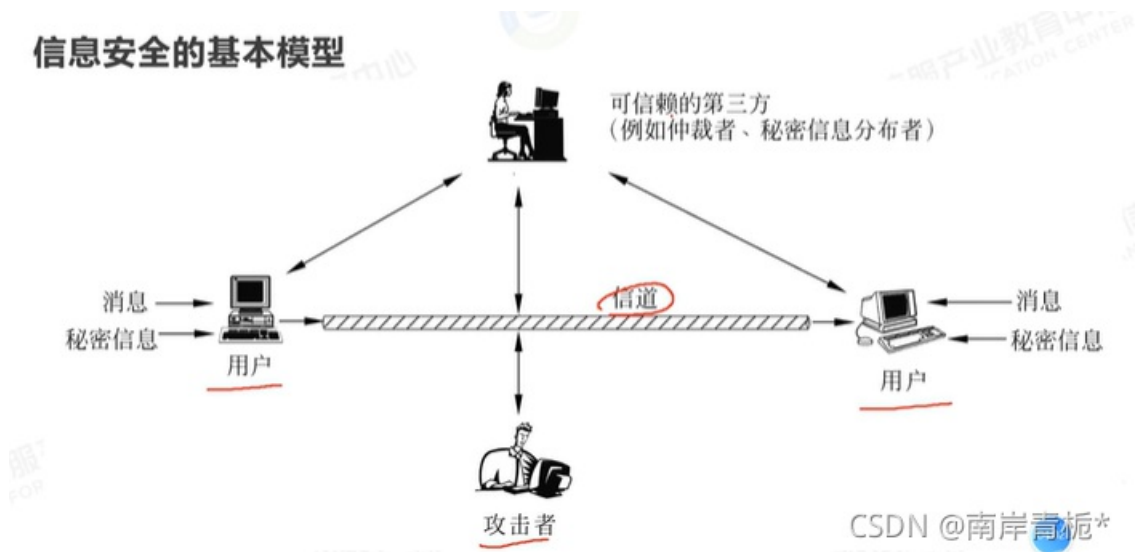
对称密码

DES解密

## 概述

- 密码学是研究编制密码和破译密码的技术科学
- 使信息保密的技术和科学学叫密码编码学
- 破译密文的科学与技术叫密码分析学

### 信息安全的基本模型



## 密码学基本概念

明文：没有加密的文字or字符串

密文：对明文加密之后的报文

加密算法：

解密算法

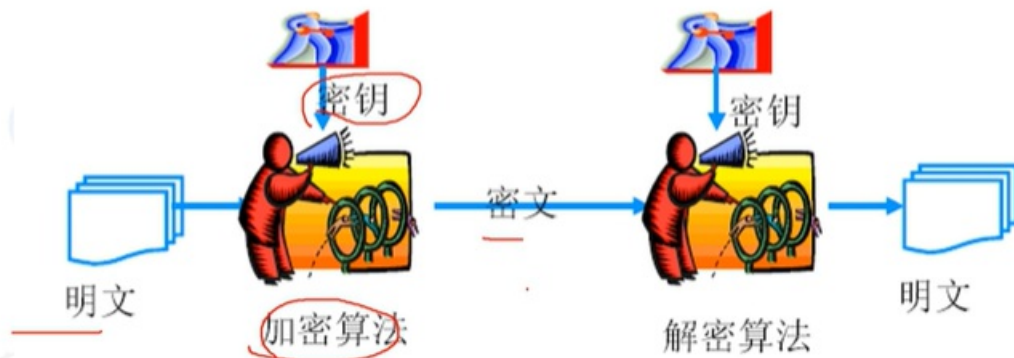
加密密钥

解密密钥

单钥密码体系

双钥密码体系

## 加解密码示意图



## 加解密过程示意图

CSDN @南岸青栀\*

## 密码学的历史与发展

### 第一阶段（1949年前） 古典密码发展阶段

隐写术，暗语，隐语，藏头诗等

采用手工或机械变换的方式实现

单表代换密码：Caesar密码、仿射密码

多表代换密码：Vigenere，Hill密码等

转轮密码：Enigma，Red密码等

### 第二阶段：近代密码阶段（1949~1976）

- 1949，Shannon发表了《保密系统的通信理论》，用信息论的观点分析了密码学的基本原理，奠定了密码学的基本理论。
- 1967年David Kahn出版了《破译者》一书

### 现代密码学阶段（1976~至今）

- 1976年，Diffie、Hellman发表了《密码学新方向》，开辟了公钥密码学的新领域
- 1976年，美国建立了DES为联邦标准

### 现代密码学的主要发展方向

混沌密码学：混沌加密的基本原理是利用混沌系统产生混沌序列作为密钥序列，接收方用混沌同步的方法将明文信号提取出来实现解密

量子密码学：量子密码学利用量子力学的特性来加密的科学。任何试图尝试读取量子态的行动都会改变量子态本身。

## 密码体制分类

受限制的 (restricted) 算法: 算法的保密性基于保密算法的秘密。

基于密钥 (key-based) 的算法: 算法的保密性基于对密钥的保密。

一个加密系统S可以用数学符号描述如下:

$$S = \{P, C, K, E, D\}$$

P --- 明文空间, 表示全体可能出现的明文集合  
C --- 密文空间, 表示全体可能出现的密文集合  
K --- 密钥空间, 密钥是加密算法中的可变参数  
E --- 加密算法, 由一些公式, 法则或程序构成  
D --- 解密算法, E的逆

当给定密钥k时, 各符号的关系

$C = E_k(P)$ , 对明文P加密后得到密文C  
 $P = D_k(C) = D_k(E_k(P))$ , 对密文C解密后得明文P  
加密设计主要是确定E, D, K

优秀密码算法应该是基于密钥的保密, 而非算法的保密

现代密码学用密钥解决问题, 密钥用K表示

密钥K的可能值的范围叫做密钥空间 (keyspace)

如加密和解密都用一个密钥, 加/解密函数变成:

$$E_k(M) = C$$

$$D_k(C) = M$$

$$D_k(E_k(M)) = M$$

单钥体制, 对称加密

流密码

分组密码

双钥体制

1976年, Diffie和Hellman首先引入

一对密钥: 公钥和密钥

## 密码攻击

攻击类型	攻击者掌握的内容
唯密文攻击	加密算法 截获的部分密文

攻击类型	攻击者掌握的内容
已知明文攻击	加密算法 截获的部分密文 一个或多个明文密文对
选择明文攻击	加密算法 截获的部分密文 自己选择的明文消息，以及由密钥产生的相对密文
选择密文攻击	加密算法 截获的部分密文 自己选择的密文消息，以及相应的被解密的明文

## 算法的安全性

如果破译算法的代价大于加密数据价值，那么加密算法是安全的

如果破译算法所需要的时间比加密数据保密的时间长，那么你可能是安全的

## 编码与密码

### 编码基础

#### ASCII

标准ASCII码，使用 7 位二进制数（剩下的一位二进制为0）来表示所有的大小写字母，数字，标点符号。

后128个称为扩展ASCII码，许多基于x86的系统都支持使用扩展ASCII。扩展ASCII码允许将每个字符的第八位用于确定附加的128个特殊符号字符，外来语字母和图形符号

ASCII 字符代码表 一

高四位 低四位		ASCII非打印控制字符								ASCII 打印字符													
		0000				0001				0010	0011	0100	0101	0110	0111								
		0				1				2	3	4	5	6	7								
	+进制	字符	ctrl	代码	字符解释	+进制	字符	ctrl	代码	字符解释	+进制	字符	+进制	字符	+进制	字符	+进制	字符	ctrl				
0000	0	0	BLANK NULL	^@	NUL	空	16	▶	^P	DLE	数据链路转意	32		48	0	64	@	80	P	96	`	112	p
0001	1	1	☺	^A	SOH	头标开始	17	◀	^Q	DC1	设备控制 1	33	!	49	1	65	A	81	Q	97	a	113	q
0010	2	2	☹	^B	STX	正文开始	18	↕	^R	DC2	设备控制 2	34	"	50	2	66	B	82	R	98	b	114	r
0011	3	3	♥	^C	ETX	正文结束	19	!!	^S	DC3	设备控制 3	35	#	51	3	67	C	83	S	99	c	115	s
0100	4	4	◆	^D	EOT	传输结束	20	⏏	^T	DC4	设备控制 4	36	\$	52	4	68	D	84	T	100	d	116	t
0101	5	5	♣	^E	ENQ	查询	21	♫	^U	NAK	反确认	37	%	53	5	69	E	85	U	101	e	117	u
0110	6	6	♠	^F	ACK	确认	22	■	^V	SYN	同步空闲	38	&	54	6	70	F	86	V	102	f	118	v
0111	7	7	●	^G	BEL	震铃	23	↑↓	^W	ETB	传输块结束	39	'	55	7	71	G	87	w	103	g	119	w
1000	8	8	◻	^H	BS	退格	24	↑	^X	CAN	取消	40	(	56	8	72	H	88	X	104	h	120	x
1001	9	9	○	^I	TAB	水平制表符	25	↓	^Y	EM	媒体结束	41	)	57	9	73	I	89	Y	105	i	121	y
1010	A	10	◻	^J	LF	换行/新行	26	→	^Z	SUB	替换	42	*	58	:	74	J	90	Z	106	j	122	z
1011	B	11	♂	^K	VT	竖直制表符	27	←	^[	ESC	转意	43	+	59	;	75	K	91	[	107	k	123	{
1100	C	12	♀	^L	FF	换页/新页	28	└	^\ FS	文件分隔符	44	,	60	<	76	L	92	\	108	l	124		
1101	D	13	♪	^M	CR	回车	29	↔	^] GS	组分隔符	45	-	61	=	77	M	93	]	109	m	125	}	
1110	E	14	🎵	^N	SO	移出	30	▲	^6 RS	记录分隔符	46	.	62	>	78	N	94	^	110	n	126	~	
1111	F	15	☼	^O	SI	移入	31	▼	^- US	单元分隔符	47	/	63	?	79	O	95	_	111	o	127	Δ <sup>Back space</sup>	

注：表中的ASCII字符可以用:ALT + “小键盘上的数字键”输入

扩展ASCII表

## ASCII 字符代码表 二

高四位 低四位		扩充ASCII码字符集															
		1000		1001		1010		1011		1100		1101		1110		1111	
		8		9		A/10		B/16		C/32		D/48		E/64		F/80	
		+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符	+进制	字符
0000	0	128	Ç	144	É	160	á	176	☒	192	Ł	208	⋈	224	α	240	≡
0001	1	129	Ü	145	æ	161	í	177	☒	193	⌊	209	≡	225	β	241	±
0010	2	130	é	146	Æ	162	ó	178	☒	194	⌋	210	⌋	226	Γ	242	≥
0011	3	131	â	147	ô	163	ú	179		195	⌋	211	⌋	227	Π	243	≤
0100	4	132	ä	148	ö	164	ñ	180	⌋	196	—	212	Ô	228	Σ	244	∫
0101	5	133	à	149	ò	165	Ñ	181	⌋	197	+	213	ƒ	229	σ	245	∫
0110	6	134	å	150	û	166	ª	182	⌋	198	⌋	214	⌋	230	μ	246	÷
0111	7	135	ç	151	ù	167	º	183	⌋	199	⌋	215	⌋	231	τ	247	≈
1000	8	136	ê	152	ÿ	168	¿	184	⌋	200	⌋	216	⌋	232	Φ	248	°
1001	9	137	ë	153	ÿ	169	⌋	185	⌋	201	⌋	217	⌋	233	Θ	249	•
1010	A	138	è	154	ÿ	170	⌋	186	⌋	202	⌋	218	⌋	234	Ω	250	•
1011	B	139	ï	155	ç	171	½	187	⌋	203	⌋	219	☒	235	δ	251	√
1100	C	140	î	156	£	172	¼	188	⌋	204	⌋	220	☒	236	∞	252	n
1101	D	141	ì	157	¥	173	¡	189	⌋	205	=	221	⌋	237	φ	253	²
1110	E	142	Ä	158	₤	174	«	190	⌋	206	⌋	222	⌋	238	ε	254	■
1111	F	143	Å	159	ƒ	175	»	191	⌋	207	⌋	223	☒	239	∩	255	BLANK FF

注：表中的ASCII字符可以用:ALT + “小键盘上的数字键” 输入

## unicode

因为ASCII只有8位，只能表达256种字符。所以出现unicode，unicode是国际组织制定的可以容纳世界上所有文字和符号的字符编码方案。使用16位的编码空间（每个字符占用2个字节）

UTF-8（8-bit Unicode Transformation Format）是一种针对Unicode 的可变长度字符编码。

UTF-8使用1~6个字节为每个字符编码

## BASE64

Base64是一种基于64个可打印字符来表示二进制数据的表示方法。

每6个比特位一个单元，对应某个可打印字符除了A-Z，a-z，0-9共62个字符还有“+”，“/”，最后用“=”填充不能被3整除的空位。

### • 编码 “Man”

文本	M				a				n														
ASCII编码	77				97				110														
二进制位	0	1	0	0	1	1	0	1	0	1	1	0	0	0	1	0	1	1	0	1	1	1	0
索引	19				22				5				46										
Base64编码	T				W				F				u										

## 古典密码学



古典密码的加密是将明文的每一个字母替换为字母表中的另一个字母。

## 单表代换密码

### 凯撒密码

凯撒密码加密时将明文中的每个字母按字母表顺序向前或向后移动固定数目作为密文

#### 如偏移量是左移3为例

- 明文: ABCDEFGHIJKLMNOPQRSTUVWXYZ
- 密文: DEFGHIJKLMNOPQRSTUVWXYZABC

### 移位密码

与凯撒密码类似，最早的凯撒密码是固定左移3位。

移位密码可以任意移动，后期不仅处理26个字母，还会处理数字和特殊字符。参照ASCII表进行位移

### 仿射密码

#### ■ 原理

仿射密码的加密函数是  $E(x) = (ax + b) \pmod{m}$ ，其中

- $x$  表示明文按照某种编码得到的数字
- $a$  和  $m$  互质
- $m$  是编码系统中字母的数目。

解密函数是  $D(x) = a^{-1}(x - b) \pmod{m}$ ，其中  $a^{-1}$  是  $a$  在  $\mathbb{Z}_m$  群的乘法逆元。

#### ■ 乘法逆元:

$$a * a^{-1} \pmod{m} = 1$$

CSDN @南岸青栀\*

例子:

加密

## 例子

下面我们以  $E(x) = (5x + 8) \bmod 26$  函数为例子进行介绍，加密字符串为AFFINE CIPHER，这里我们直接采用字母表26个字母作为编码系统

明文	A	F	F	I	N	E	C	I	P	H	E	R
x	0	5	5	8	13	4	2	8	15	7	4	17
$y = 5x + 8$	8	33	33	48	73	28	18	48	83	43	28	93
$y \bmod 26$	8	7	7	22	21	2	18	22	5	17	2	15
密文	I	H	H	W	V	C	S	W	F	R	C	P

其对应的加密结果是IHHWVCSWFRCP。

CSDN @南岸青栀\*

解密：

## 解密 $a = 5$ $b = 8$ , $a$ 对26的乘法逆元为21。 $5 * 21 \bmod 26 = 1$

对于解密过程，正常解密者具有a与b，可以计算得到  $a^{-1}$  为21，所以其解密函数是  $D(x) = 21(x - 8) \pmod{26}$ ，解密如下

密文	I	H	H	W	V	C	S	W	F	R	C	P
y	8	7	7	22	21	2	18	22	5	17	2	15
$x = 21(y - 8)$	0	-21	-21	294	273	-126	210	294	-63	189	-126	147
$x \bmod 26$	0	5	5	8	13	4	2	8	15	7	4	17
明文	A	F	F	I	N	E	C	I	P	H	E	R

可以看出其特点在于只有26个英文字母。

CSDN @南岸青栀\*

## 多表代换密码

- 该算法基于5\*5的字母矩阵，该矩阵使用一个关键词构造（即密钥）
- 从左到右、从上到下顺序，填入关键词的字母（去除重复字母）后，将字母表其余字母填入。（I=J）
- 将明文两个分为一组，若出现相同字母，则用X替代最后字母。
- 在每组中，查找矩阵替换：
  - 若两个字母同行，则用右方字母替换
  - 若两个字母同列，则用下方字母替换
  - 若即不同行也不同列，则用矩阵对角字母替换

CSDN @南岸青栀\*

查找矩阵替换：

若两个字母同行，则用右方字母替换

若两个字母同列，则用下方字母替换

若既不同行也不同列，则用矩阵对角字母替换

- 例子:
- 以 playfair example 为密钥，构造矩阵
- 明文为: **hide the gold in the tree stump**

HI DE TH EG OL DI NT HE TR EX ES TU MP

- 密文为:

BM OD ZB XD NA BE KU DM UI XM MO UV IF

P	L	A	Y	F
I	R	E	X	M
B	C	D	G	H
K	N	O	Q	S
T	U	V	W	Z

CSDN @南岸青栀\*

### polybius密码 (棋盘密码)

- 又称棋盘密码。
- 将给定明文加密为两两组合的特征
- 加密后结果只有5种字符
- ADFGX密码是德军在一战中使用的栏块密码

$$\begin{pmatrix} & A & D & F & G & X \\ A & b & t & a & l & p \\ D & d & h & o & z & k \\ F & q & f & v & s & n \\ G & g & j & c & u & x \\ X & m & r & e & w & y \end{pmatrix}$$

- 明文=A T T A C K A T O N C E
- 密文: AF AD AD AF GF DX AF AD DF FX GF XF

CSDN @南岸青栀\*

### vigenere密码

使用26个字母构成字母矩阵横行为明文列，纵向为密钥列

■ 使用26个字母构成字母矩阵  
横行为明文列，纵向为密钥列

■ 明文: come greatwall

■ 密钥: crypto

■ 扩充密钥与是明文一样长

■ 密文 efktzferrltzn

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

其他类型密码

培根密码

二进制思想：粗体字为B，正常字为A

■ 用两种不同字体，代表A和B（或者0和1），结合加密表进行加密。

■ 加密表如图：

a	<u>AAAAA</u>	g	AABBA	n	ABBAA	t	BAABA
b	<u>AAAAB</u>	h	AABBB	o	ABBAB	u-v	BAABB
c	<u>AAABA</u>	i-j	ABAAA	p	ABBBA	w	BABAA
d	AAABB	k	ABAAB	q	ABBBB	x	BABAB
e	AABAA	l	ABABA	r	BAAAA	y	BABBA
f	AABAB	m	ABABB	<u>s</u>	<u>BAAAB</u>	z	BABBB

■ 明文: steganography

■ 正常字体是A，粗体是B，加密结果如图

To encode a message each letter of the plaintext is replaced by a group of five of the letters 'A' or 'B'.

CSDN @南岸青栀\*

栅栏密码

将明文分成N个一组，然后每组的第一个连起来

- 把明文分成N个一组，然后每组的第1个字连起来，然后连第2个.....

- 例子

- 明文: THERE IS A CIPHER

- 分组: TH ER IS ACI PH ER

- 取出: TEESCPE HRIAHR

CSDN @南岸青栀\*

## 摩斯密码

用.和\_表示，以前用于发电报

INTERNATIONAL MORSE CODE		
A ..	N -. .	0 -----
B -... .	O ---	1 .----
C -.-. .	P -.-. .	2 ..---
D -.. .	Q ---. .	3 ...--
E .	R -. .	4 ....-
F ..-. .	S ...	5 ..... .
G ---. .	T -	6 -.... .
H ....	U -. .	7 ---... .
I .. .	V ...-. .	8 ----. .
J .-.- .	W -. .	9 ----- .
K -.- .	X -.-. .	. .-.-. .
L .-... .	Y -.-. .	, ---. .
M -- .	Z -. .	? ..... .

## CTF中奇怪密码

倒序加密

电脑键盘密码

- ABCDE加密后变成QWERT

- 按键盘位置转换字母表，红色是明文

Q	A	B	C	D	E	F	G	H	I	J
A	K	L	M	N	O	P	Q	R	S	
Z	X	U	V	W	X	Y	Z			

CSDN @南岸青栀\*

### 键盘密码

- 利用键盘上按键所在的行与列，进行编号加密码。下面是一种形态

- 也有忽略大小写的编号方法

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	~	!	@	#	\$	%	^	&	*	(	)	_	+	
2	`	1	2	3	4	5	6	7	8	9	0	-	=	\
3	Q	W	E	R	T	Y	U	I	O	P	{	}		
4	q	w	e	r	t	y	u	i	o	p	[	]		
5	A	S	D	F	G	H	J	K	L	:	"			
6	a	s	d	f	g	h	j	k	l	;	'			
7	Z	X	C	V	B	N	M	<	>	?				
8	z	x	c	v	b	n	m	,	.	/				

键盘密码加密的原理同棋盘密码，只是利用了键盘作为方阵。

例：  
密文：  
87 34 112  
55 47 87 410

明文：  
mR\_Gump

CSDN @南岸青栀\*

### 手机键盘加密

- 来源于以前的手机9键 键盘

- 输入6，得到字母m，编码为61
- 输入66，得到字母n，编码为62
- 输入666，得到字母o，编码为63



CSDN @南岸青栀\*

## 当铺密码

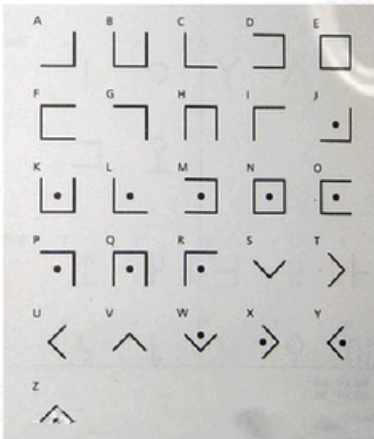
用汉子来表示数字，进行编码。汉子特点是出头数量

■ 如下:

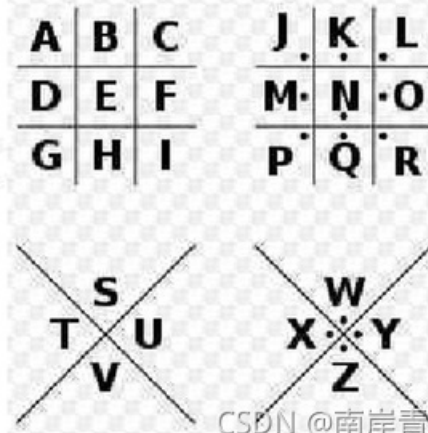
1	2	3	4	5	6	7	8	9	0
由	中	人	工	大	王	夫	进	羊	口

## 猪圈密码

■ 编码方式



解密方式



## 对称加密

### 基本概念

特征：加密解密使用相同的密钥（单密钥加密）

根据加密对象分为

流加密：每次加密都通过密钥生成一个密钥流，解密也是使用同一个密钥流，明文与同样长度的密钥流进行异或运算得到密文，密文与同样的密钥流进行异或运算得到明文。典型算法RC4。

- RC4:

RC4是典型的流加密算法，常用于SSL/TSL，及802.11和WAP中。

流加密会逐字节加密数据，RC4本质是以密钥为种子产生的随机数来对明文进行逐字节异或。

分组密码与流密码的区别就在于有无记忆性

### 加密过程

#### 1.初始化S表

(1) 对S表进行线性填充，一般为256个字节

(2) 用种子密钥填充另一个256字节的K表

(3) 用K表对S表进行初始置换

2. 密钥流的生成（为每一个待加密的字节生成一个伪随机数，用来异或，S表一旦完成初始化，种子密钥就不再被使用）。

## ■ 概述

- 每次加密明文中的一个字节
- 密钥长度可变，1-256字节

## ■ 基本流程

- 初始化S和T
- 计算排列S, j从0到255
- 产生与明文等长的密钥流
- 加密运算

CSDN @南岸青栀\*

## ■ 算法描述

- S和T的初始状态：S中元素的值按升序被置为0-255，同时建立一个临时向量T。将密钥的值循环复制到T向量中。

## ■ S的初始置换

用T产生S的初始置换，置换伪码如下

```
j = 0;
for (i = 0 ; i < 256 ; i++){
    j = (j + S[i] + T[i]) mod 256;
    swap(S[i] , S[j]);
}
```

CSDN @南岸青栀\*



密钥流生成伪码如下

$i, j = 0;$

while (true){

$i = (i + 1) \bmod 256;$

$j = (j + S[i]) \bmod 256;$

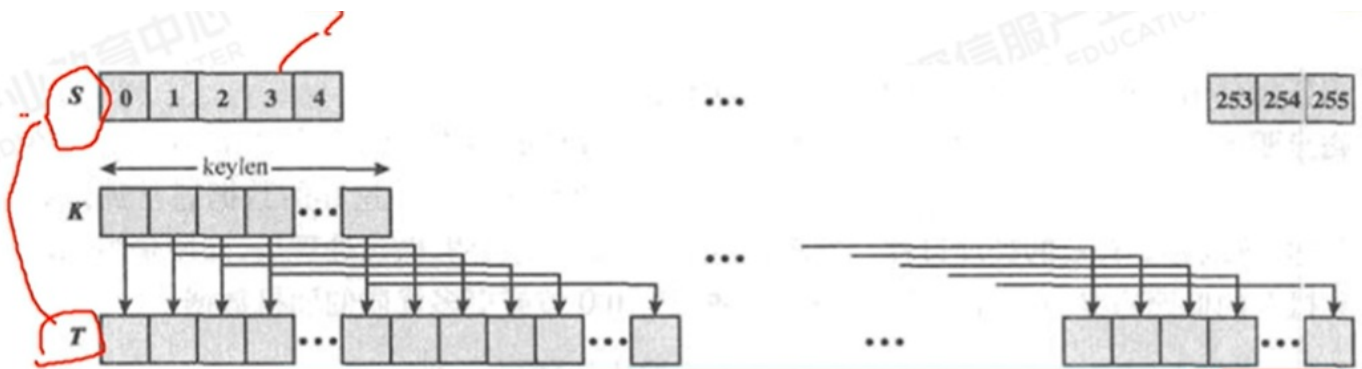
swap( $S[i], S[j]$ );

$t = (S[i] + S[j]) \bmod 256;$

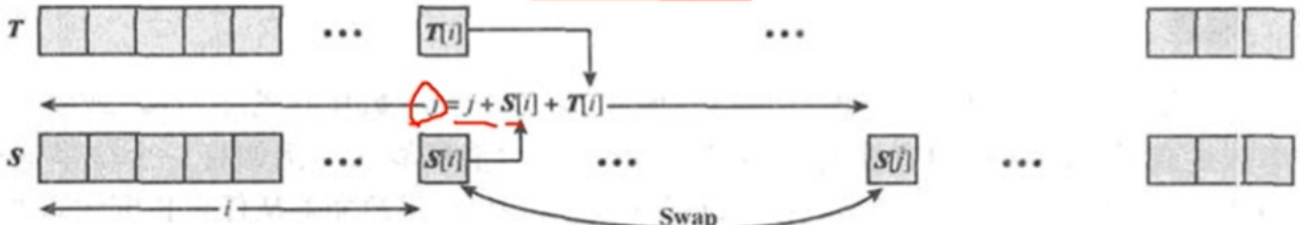
$k = S[t];$

}

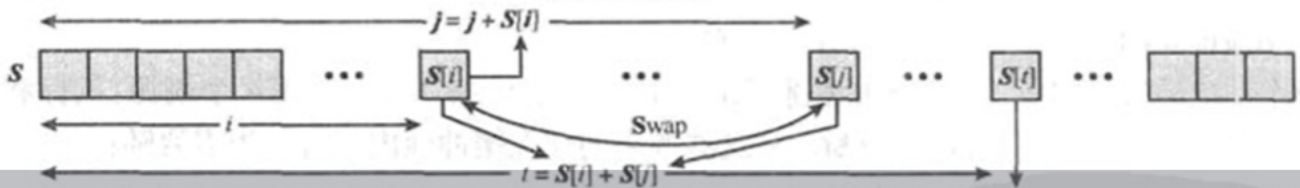
CSDN @南岸青栀\*



(a) S和T的初始状态



(b) S的初始置换



(c) 密钥流的生成

CSDN @南岸青栀\*

RC4加密实现

```
# RC4加密算法实现
```

```
# 初始化S表
```

```
def ini_S(K):
```

```
    S = [a for a in range(256)]
```

```
    j = 0
```

```
    for i in range(256):
```

```
        j = (j + S[i] + K[i]) % 256
```

```
        S[i], S[j] = S[j], S[i]
```

```

return S

# 种子密码生成临时表
def create_R(seeds):
    T = bytes(seeds, encoding='utf-8') # 将utf-8的字节码解码成Python默认的unicode的字符串
    T = list(T) # 转换成列表类型
    print('private seeds:', T) # 输出密钥
    len_key = len(T) # 确定密钥的长度
    R = [T[i % len_key] for i in range(256)] # 对T进行复制和填充, 并保存在R数组中

    return R

# 密钥流的生成
def steam_K(S,length):
    i = 0
    j = 0
    Sh = [] # Sh保存流
    length = int(length) # 保存明文的长度或者密文长度的一半
    for i in range(length):
        i = (i + 1) % 256 # 如果用完256个位置, 再从S[i]开始
        j = (j + S[i]) % 256 # 选择S[i]与S的另一个字节
        # 对S[i]和s的另一字节进行交换
        temp = S[i]
        S[j] = S[i]
        S[i] = temp
        h = (S[j] + S[i]) % 256 # 防溢出操作
        k = S[h]
        Sh.append(k) # 将k在Sh的末尾添加新的对象
    return Sh

if __name__ == '__main__':
    choose = input('choose 1--encryption, 2--decryption:') # 1为加密, 2为解密

    if choose == '1':
        key = input("input the key:")
        R = create_R(key)
        S = ini_S(R)
        plaintext = input("input the plaintext:")
        K = steam_K(S,len(plaintext))
        ciphertext = ''
        for i in range(len(plaintext)):
            ciphertext = ciphertext + '%02x' % (K[i] ^ ord(plaintext[i])) # 进行异或运算并保存在ciphertext
        print('ciphertext:', ciphertext) # 输出密文

# 解密操作
    if choose == '2':
        key = input('input the key:') # key保存密钥
        K = create_R(key) # K保存key的辅助表
        S = ini_S(K) # S的初始化
        ciphertext = input('input the ciphertext:') # ciphertext保存密文
        plaintext = '' # plaintext保存明文
        Sh = steam_K(S, len(ciphertext)/2) # 生成密文的流
        # Sh与密文的下一个字节进行异或运算
        for i in range(int(len(ciphertext)/2)):
            plaintext = plaintext + chr(int(ciphertext[0:2], 16) ^ Sh[i]) # 进行异或运算并保存在plaintext
            ciphertext = ciphertext[2:]
        print('plaintext:', plaintext) # 输出密文

```

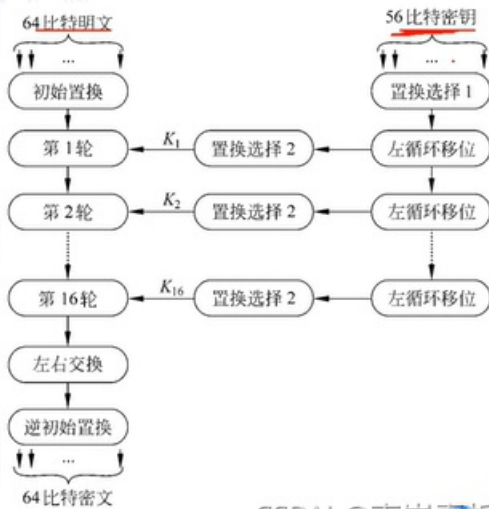
块加密（分组密码），加密和解密序列分为一个个分组，最后吧每一块序列合并到一起，形成明文或密文。根据不同的分组加密方式，每个分组之间可以有联系，也可以没有联系。典型算法DES和AES

DES（data encryption standard，数据加密标准）

- **数据加密标准（data encryption standard, DES）是迄今为止世界上最为广泛使用和流行的一种分组密码算法**
- 它的分组长度为64比特，密钥长度为56比特，它是由美国IBM公司研制
- DES在1975年3月17日首次被公布在联邦记录中，经过大量的公开讨论后，DES于1977年1月15日被正式批准并作为美国联邦信息处理标准
- 1998年5月美国EFF(electronics frontier foundation)宣布，他们以一台价值20万美元的计算机改装成的专用解密机，用56小时破译了56比特密钥的DES
- **块加密的解密流程和加密流程往往是不同的**

CSDN @南岸青栀\*

- **明文分组长64bit**
- **密钥56bit**
- **明文处理：3个阶段**
  - 初始之后IP
  - 16轮变换
  - 逆初始之后 $IP^{-1}$
- **密钥处理：**
  - 置换函数
  - 左循环移位+置换->子密钥



CSDN @南岸青栀\*

■ 初始转换，举例：64比特的输入M，置换：X=IP(M)

M <sub>1</sub>	M <sub>2</sub>	M <sub>3</sub>	M <sub>4</sub>	M <sub>5</sub>	M <sub>6</sub>	M <sub>7</sub>	M <sub>8</sub>
M <sub>9</sub>	M <sub>10</sub>	M <sub>11</sub>	M <sub>12</sub>	M <sub>13</sub>	M <sub>14</sub>	M <sub>15</sub>	M <sub>16</sub>
M <sub>17</sub>	M <sub>18</sub>	M <sub>19</sub>	M <sub>20</sub>	M <sub>21</sub>	M <sub>22</sub>	M <sub>23</sub>	M <sub>24</sub>
M <sub>25</sub>	M <sub>26</sub>	M <sub>27</sub>	M <sub>28</sub>	M <sub>29</sub>	M <sub>30</sub>	M <sub>31</sub>	M <sub>32</sub>
M <sub>33</sub>	M <sub>34</sub>	M <sub>35</sub>	M <sub>36</sub>	M <sub>37</sub>	M <sub>38</sub>	M <sub>39</sub>	M <sub>40</sub>
M <sub>41</sub>	M <sub>42</sub>	M <sub>43</sub>	M <sub>44</sub>	M <sub>45</sub>	M <sub>46</sub>	M <sub>47</sub>	M <sub>48</sub>
M <sub>49</sub>	M <sub>50</sub>	M <sub>51</sub>	M <sub>52</sub>	M <sub>53</sub>	M <sub>54</sub>	M <sub>55</sub>	M <sub>56</sub>
M <sub>57</sub>	M <sub>58</sub>	M <sub>59</sub>	M <sub>60</sub>	M <sub>61</sub>	M <sub>62</sub>	M <sub>63</sub>	M <sub>64</sub>

M <sub>58</sub>	M <sub>50</sub>	M <sub>42</sub>	M <sub>34</sub>	M <sub>26</sub>	M <sub>18</sub>	M <sub>10</sub>	M <sub>2</sub>
M <sub>60</sub>	M <sub>52</sub>	M <sub>44</sub>	M <sub>36</sub>	M <sub>28</sub>	M <sub>20</sub>	M <sub>12</sub>	M <sub>4</sub>
M <sub>62</sub>	M <sub>54</sub>	M <sub>46</sub>	M <sub>38</sub>	M <sub>30</sub>	M <sub>22</sub>	M <sub>14</sub>	M <sub>6</sub>
M <sub>64</sub>	M <sub>56</sub>	M <sub>48</sub>	M <sub>40</sub>	M <sub>32</sub>	M <sub>24</sub>	M <sub>16</sub>	M <sub>8</sub>
M <sub>57</sub>	M <sub>49</sub>	M <sub>41</sub>	M <sub>33</sub>	M <sub>25</sub>	M <sub>17</sub>	M <sub>9</sub>	M <sub>1</sub>
M <sub>59</sub>	M <sub>51</sub>	M <sub>43</sub>	M <sub>35</sub>	M <sub>27</sub>	M <sub>19</sub>	M <sub>11</sub>	M <sub>3</sub>
M <sub>61</sub>	M <sub>53</sub>	M <sub>45</sub>	M <sub>37</sub>	M <sub>29</sub>	M <sub>21</sub>	M <sub>13</sub>	M <sub>5</sub>
M <sub>63</sub>	M <sub>55</sub>	M <sub>47</sub>	M <sub>39</sub>	M <sub>31</sub>	M <sub>23</sub>	M <sub>15</sub>	M <sub>7</sub>

64比特输入

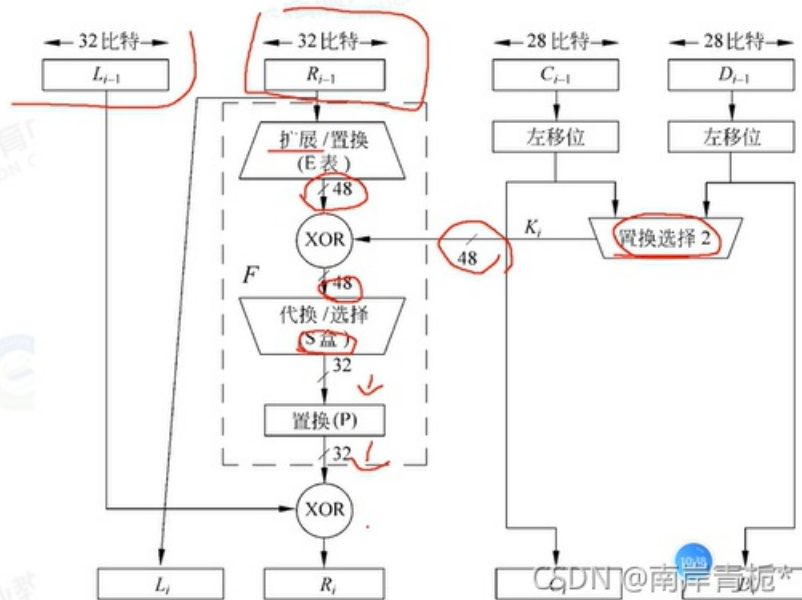
X=IP(M)

CSDN @南岸青栀\*

■ 轮变换公式：

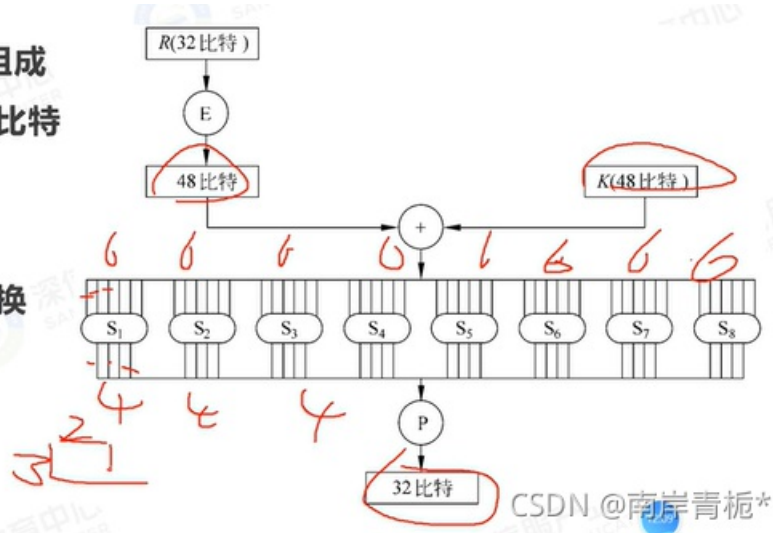
$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$$



CSDN @南岸青栀\*

- F中的代换由8个S盒组成
- 每个S盒的输入长为6比特
- 输出长为4比特
- 其变换关系由表定义
- 每个S盒给出了4个代换



加密模式

ECB (Electronic CodeBook)

是块加密中比较简单的加密模式。在ECB模式中，每一块明文数据都给独立的进行加密来生成加密块。这意味着如果你发现两个加密块有相同的内容，那么也就可以确定两个加密块的原文是相同的。

CBC (cipher-Block Chaining)

最常见的块加密模式。在CBC中，每个明文块都会自加密前被使用前一个明文快的密文进行异或；解密过程正好相反。其中第一块明文块会被使用IV及初始化向量进行异或。

密码分析

- 差分密码分析是现在攻击迭代密码最有效的方法之一，基本思想是：通过分析明文对的差值对密文对的差值的影响来恢复某些密钥比特。
- 线性密码分析是对迭代密码的一种已知明文攻击，利用的是密码算法中的“不平衡（有效）的线性逼近”。

AES (advanced encryption standard)

- Rijndael 由比利时的Joan Daemen和Vincent Rijmen设计，算法的原型是Square算法，它的设计策略是宽轨迹策略。
- 宽轨迹策略是针对差分分析和线性分析提出的，它的最大优点是可以给出算法的最佳差分特征的概率及最佳线性逼近的偏差的界。
- Rijndael密码的设计力求满足以下3条标准：
  - 抵抗所有已知的攻击。
  - 在多个平台上速度快，编码紧凑
  - 设计简单。

公钥密码体制算法条件

产生一对密钥是计算可行的

已知公钥、明文，产生密文是计算可行的

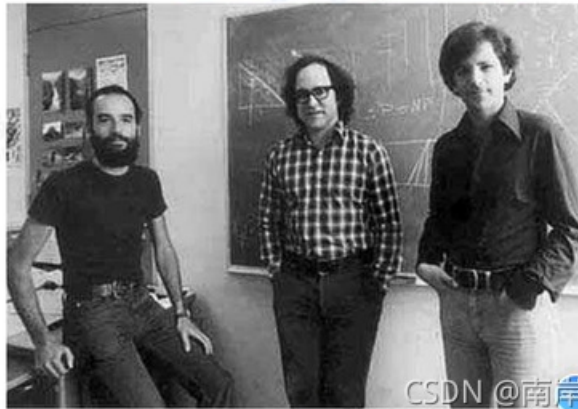
利用私钥、密文，得到明文是计算可行的

利用公钥来推断私钥是计算不可行的

利用公钥、密文，得到明文是计算不可行的

- RSA算法是1978年由R.Rivest, A.Shamir和L.Adleman提出的一种用数论构造的、也是迄今为止理论上最为成熟完善的公钥密码体制，该体制已得到广泛的应用。

RSA



CSDN @南岸青栀\*

RSA算法

## ■ 算法描述

- ① 选两个保密的大素数  $p$  和  $q$ 。
- ② 计算  $n = p \times q$ ,  $\varphi(n) = (p-1)(q-1)$ , 其中  $\varphi(n)$  是  $n$  的欧拉函数值, 令  $k = \varphi(n)$ 。
- ③ 选一整数  $e$ , 满足  $1 < e < k$ , 且  $\gcd(k, e) = 1$ 。即  $e$  与  $k$  互质。
- ④ 计算  $d$ , 满足  $d \cdot e \equiv 1 \pmod{k}$ , 即  $d$  是  $e$  在模  $k$  下的乘法逆元, 因  $e$  与  $k$  互质, 由模运算可知, 它的乘法逆元一定存在。即  $d \cdot e = k \cdot t + 1$ ,  $t$  为正整数
- ⑤ 以  $\{e, n\}$  为公钥,  $\{d, n\}$  为私钥。

CSDN @南岸青栀\*

## ■ 以上要求的本质之处在于要求一个陷门单向函数

### ■ 总结为——陷门单向函数是一族可逆函数 $f_k$ , 满足:

- ①  $Y = f_k(X)$  易于计算 (当  $k$  和  $X$  已知时)。
- ②  $X = f_k^{-1}(Y)$  易于计算 (当  $k$  和  $Y$  已知时)。
- ③  $X = f_k^{-1}(Y)$  计算上是不可行的 (当  $Y$  已知但  $k$  未知时)

- 上述表达②③中,  $k$  就是所指的陷门, 已知  $K$  时, 可以进行逆计算。

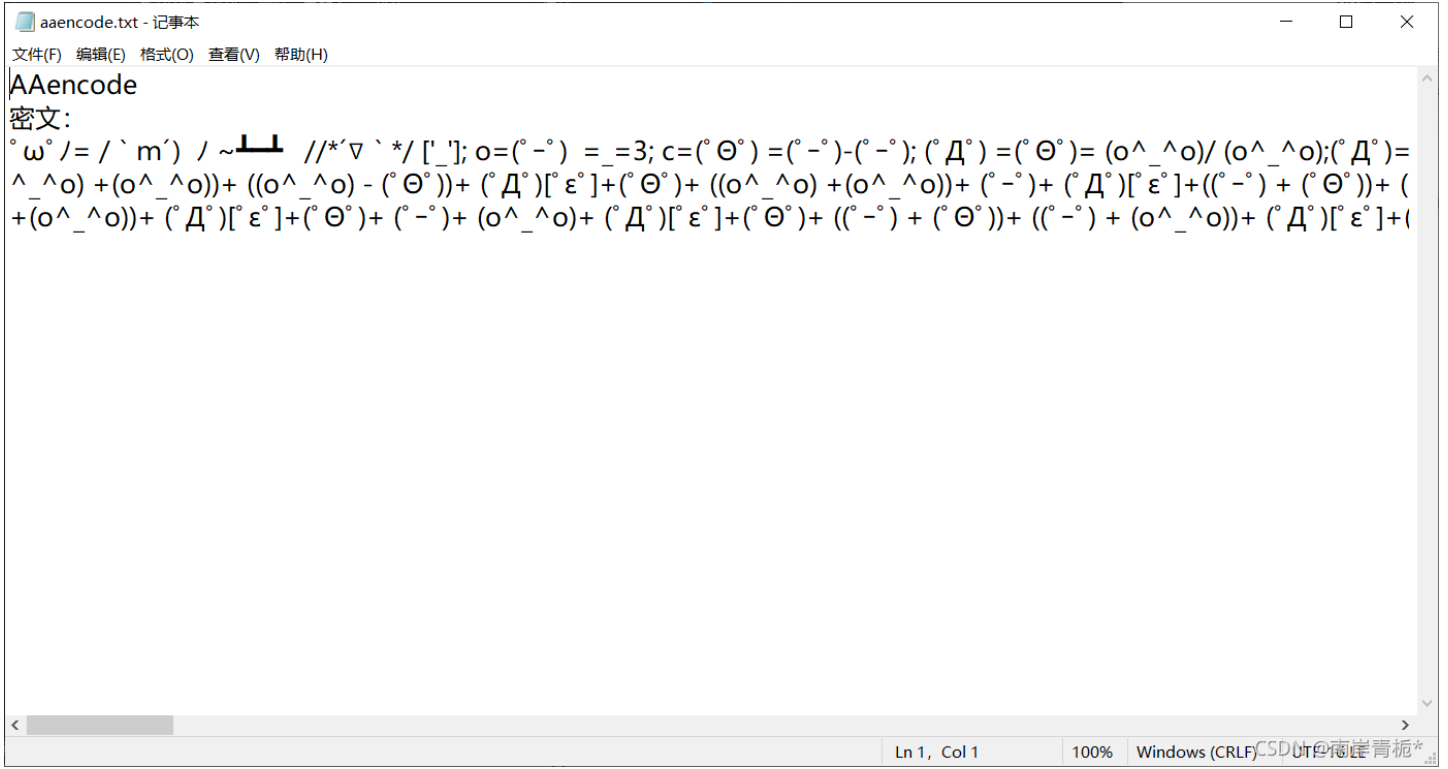
CSDN @南岸青栀\*

## CTF Wp

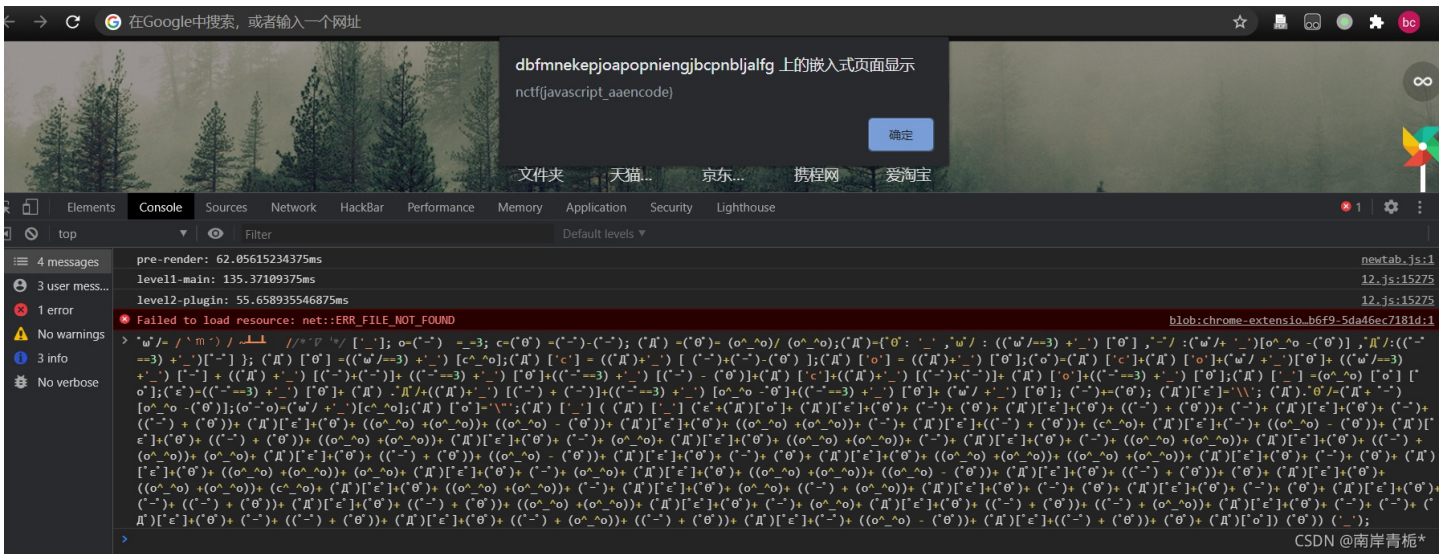
给的东西和题目不一样, 但是没关系, 因为前面几个题都是非常常规的编码方式

### 编码解码

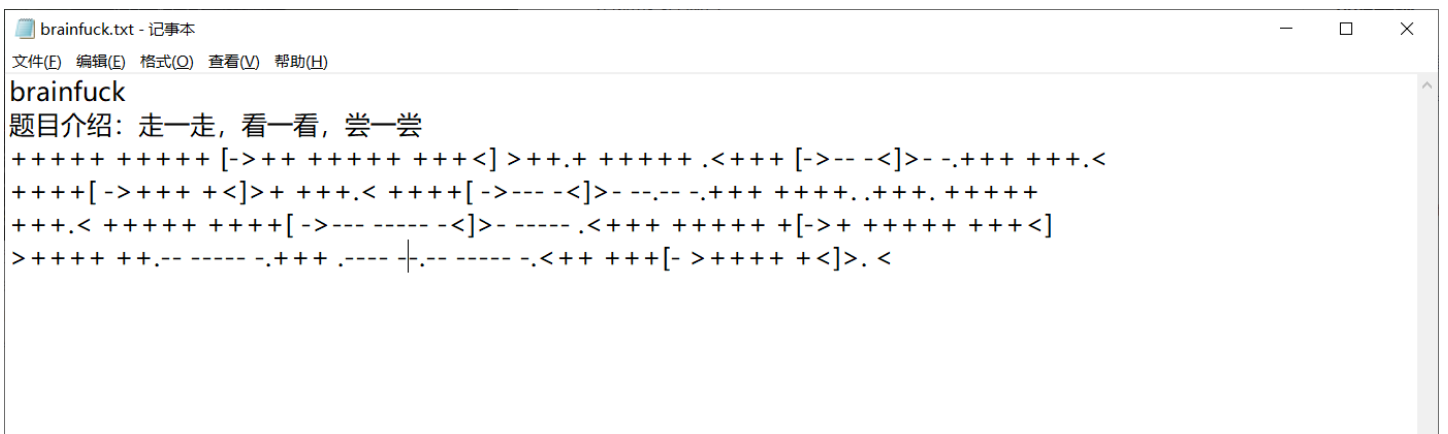
#### 1. AAencode



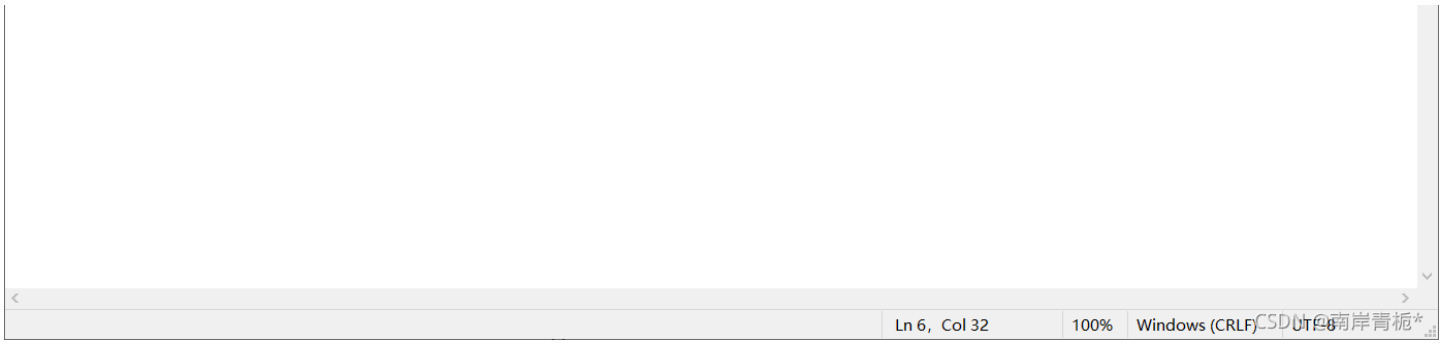
发现是一堆颜文字，AAencode主要是将js代码转换为网络表情。我们将密文直接放入浏览器控制台



## 2.brainfuck







题目提示brainfuck编码，百度一大堆

+ | ★ 收藏 | 👍 360 | 📄 1

# Brainfuck

🔊 语音 | ✎ 编辑 | 💬 讨论 | 📺 上传视频

Brainfuck是一种极小化的计算机语言，它是由Urban Müller在1993年创建的。由于fuck在英语中是脏话，这种语言有时被称为brain\*ck或brain\*\*k，甚至被简称为BF。

外文名	BrainFuck	类 型	计算机语言
发明者	Urban Müller	简 称	BF

目录	1 简介	3 条件指令	5 乘法
	2 字符标识	4 加法	6 除法

## 简介

Müller的目标是建立一种简单的、可以用最小的编译器来实现的、符合图灵完全思想的编程语言。这种语言由八种状态构成，为Amiga机器编写的编译器（第二版）只有240个字节大小!

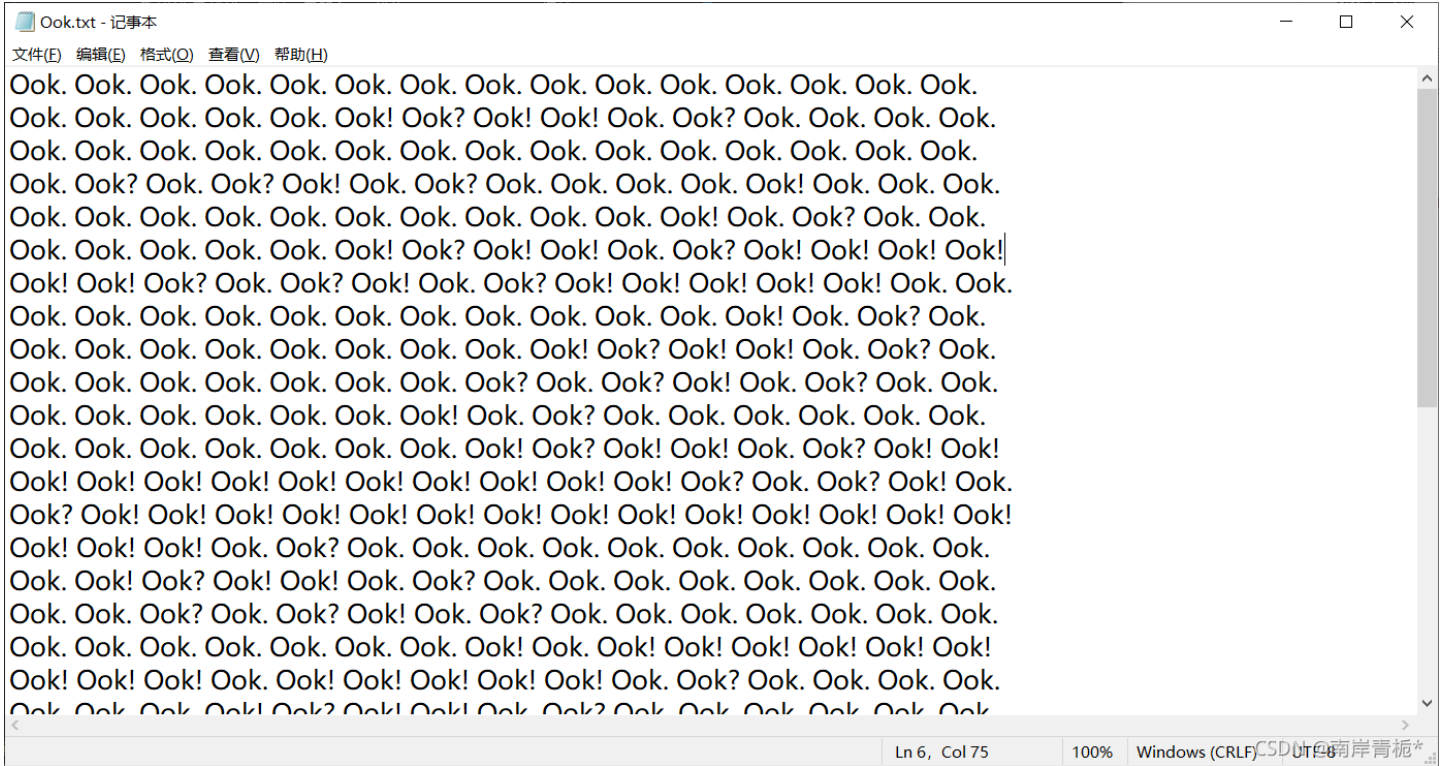
🔊 语音 | ✎ 编辑

CSDN @南岸青柁\*

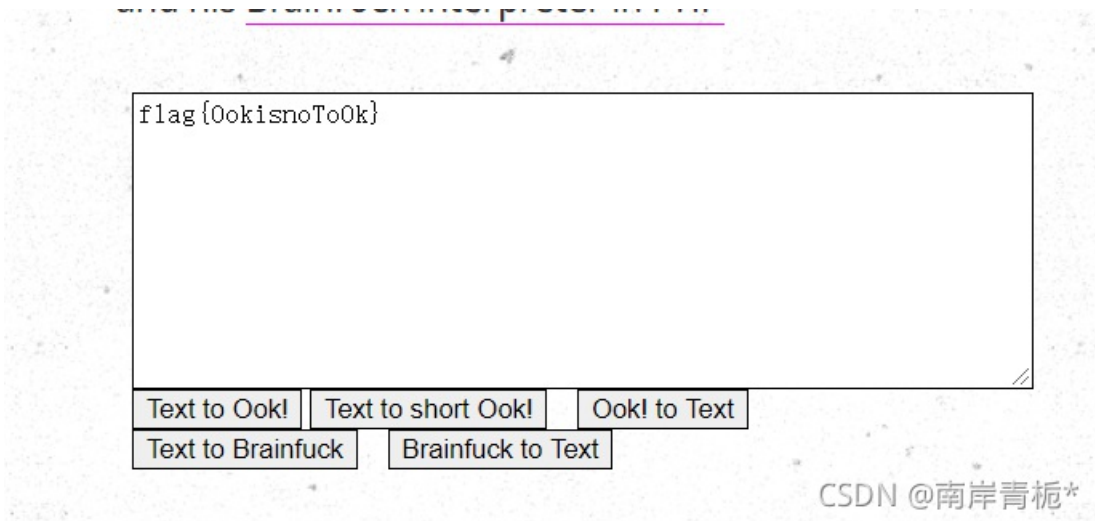
在线编解码网站：<https://www.splitbrain.org/services/ook>

CSDN @南岸青柁\*

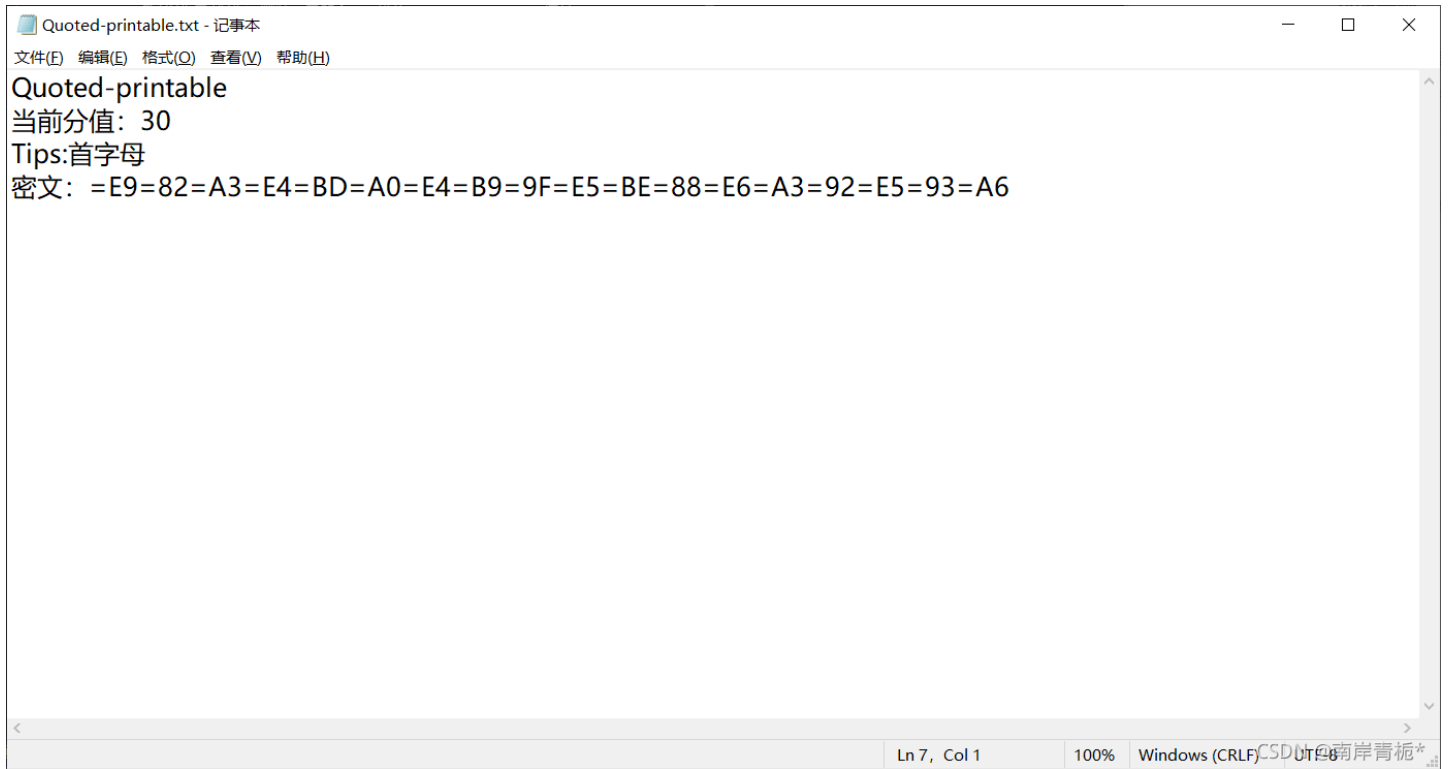
### 3.Ook



在线编解码工具



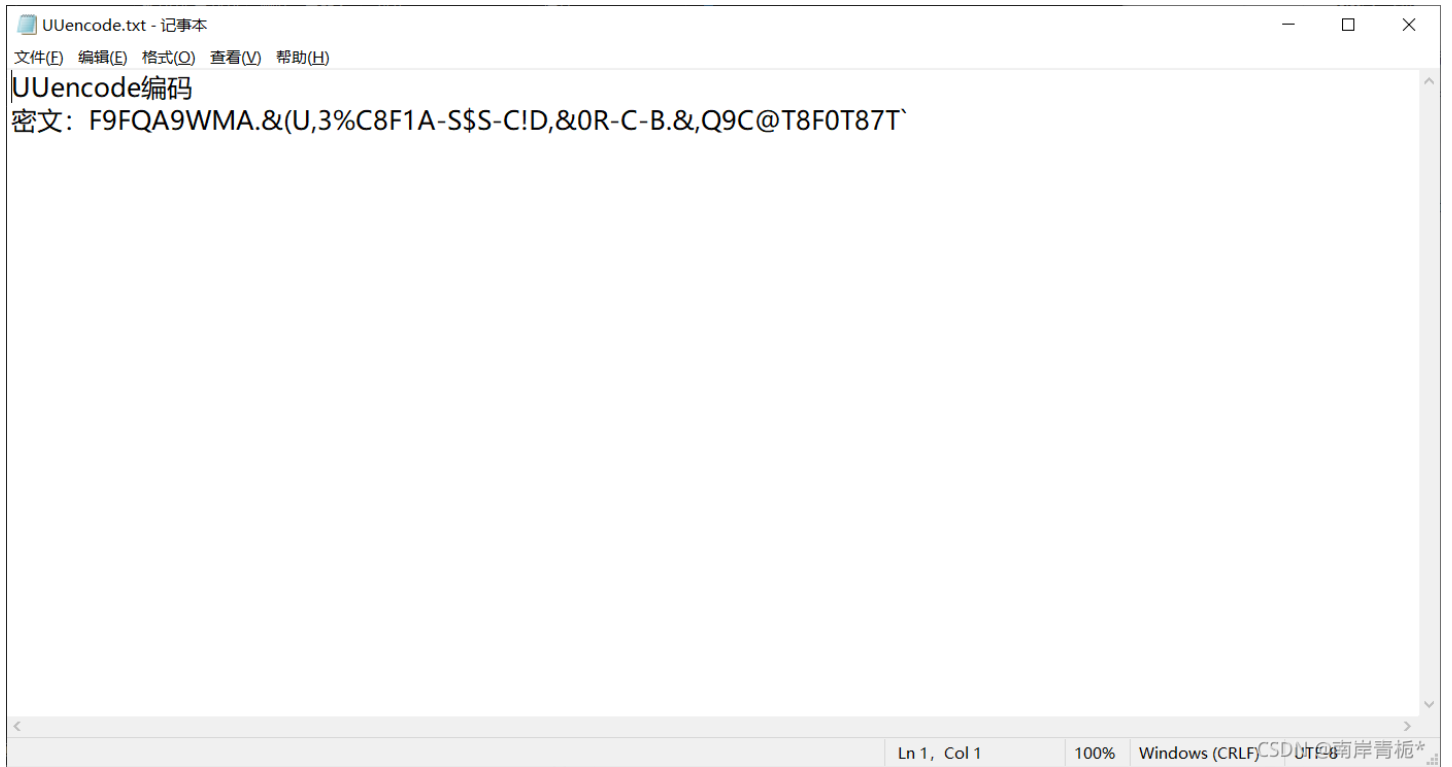
#### 4. Quoted-Printable 编码



Quoted-Printable编码表示“可打印字符引用”，它是多用途互联网邮件扩展（MIME）的一种实现方式。编码原理是：任何一个8位的字节值都可编码为3个字符，一个等号后跟随两个十六进制数字（0~9 or A~F）



## 5.UUencode



F9FQA9WMA. & (U, 3%C8F1A-S\$\$-C!D, &0R-C-B. &, Q9C@T8F0T87T`

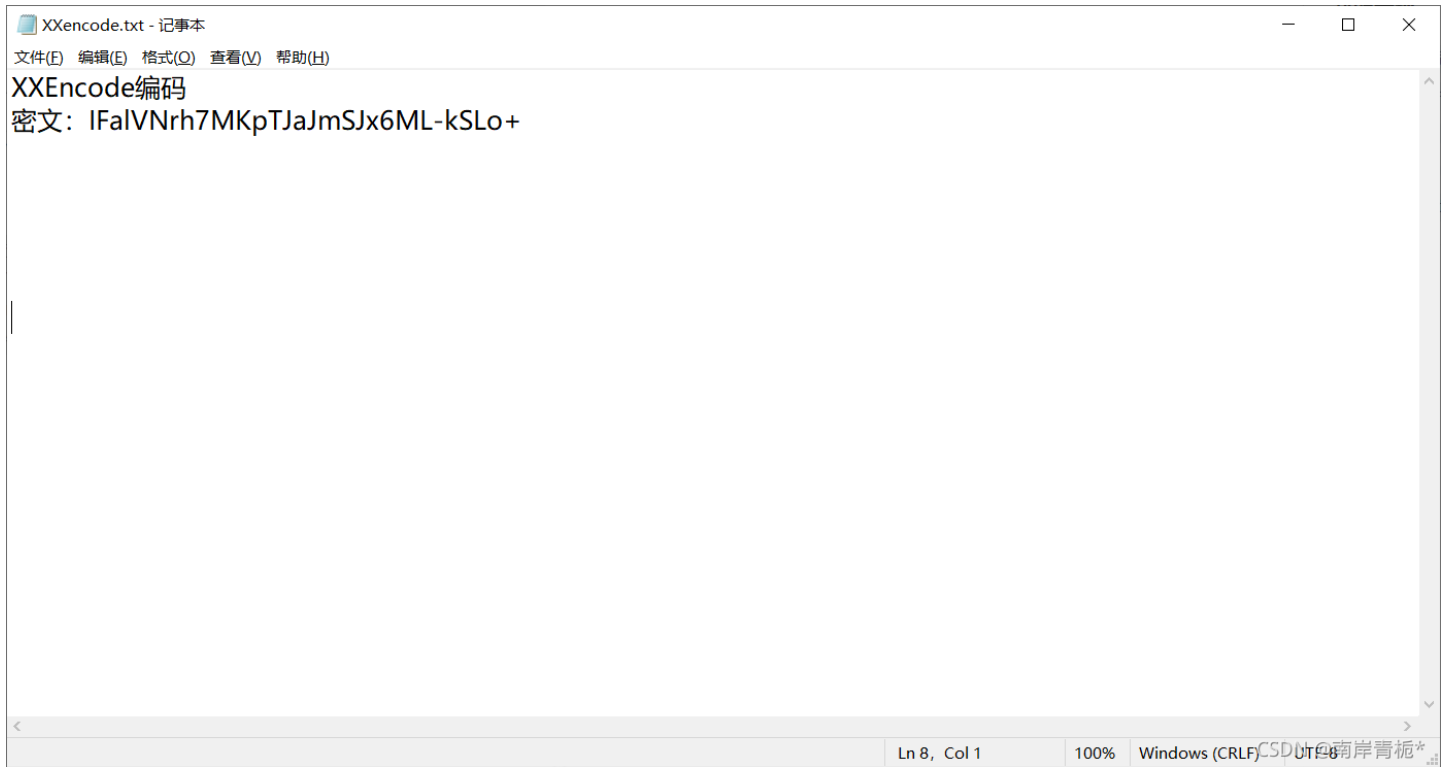
加密

解密

flag{a8b511cbda71360d0d263b8c1f84bd4a}

CSDN @南岸青栀\*

## 6.XXencode



古典密码

简单换位密码

简单换位.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

简单换位密码

密文: foerlu\_ia\_ssglue{ir!ykp}

根据题目提示编写解密脚本

```
ciphertext = "foerlu_ia_ssglue{ir!ykp}"

key = 6 # 几个一组
numrow = key
numcol = round(len(ciphertext)/key) # 4
plaintext = [""] * int(numcol)
# print(plaintext)

col = 0
row = 0
for symbol in ciphertext:
    plaintext[col] += symbol
    col += 1
    # print(plaintext)
    #
    if col == numcol:
        col = 0
        row += 1
        # print(row,plaintext)
print(''.join(plaintext))
```

```

pythonLearning - CTF脚本 - 3-19简单换位.py
1 ciphertext = "foerlu_ia_ssslue{ir!ykp}"
2
3 key = 6
4 numrow = key
5 numcol = round(len(ciphertext)/key) # 4
6 plaintext = [""] * int(numcol)
7 # print(plaintext)
8
9 col = 0
10 row = 0
11 for symbol in ciphertext:
12     plaintext[col] += symbol
13     col += 1

```

Run: 3-19简单换位

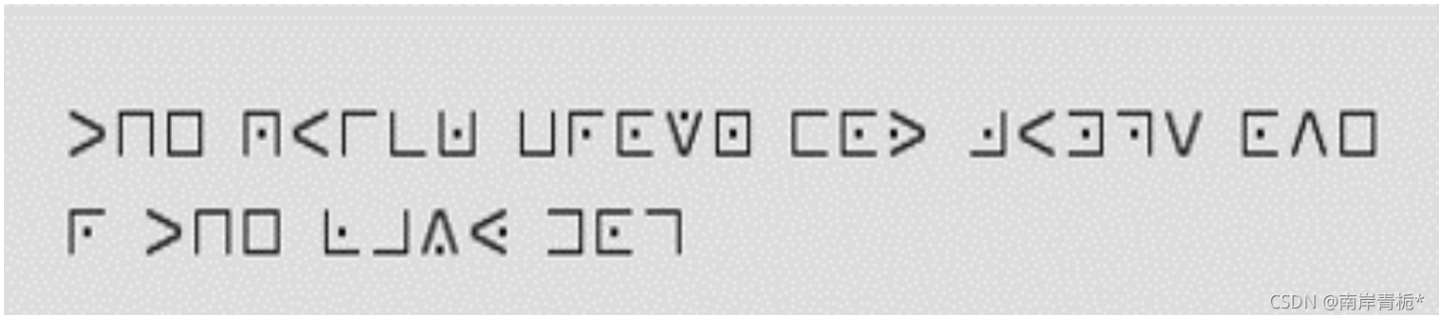
```

flag{you_like_surprise!
flag{you_like_surprise!
flag{you_like_surprise!}

```

Process finished with exit code 0

## 猪圈密码



看到题目想到猪圈密码，按照对应关系解密

凯撒密码加密 维吉尼亚密码计算 栅栏密码加密 猪圈密码加密 猪圈密码解密

摩斯密码翻译器

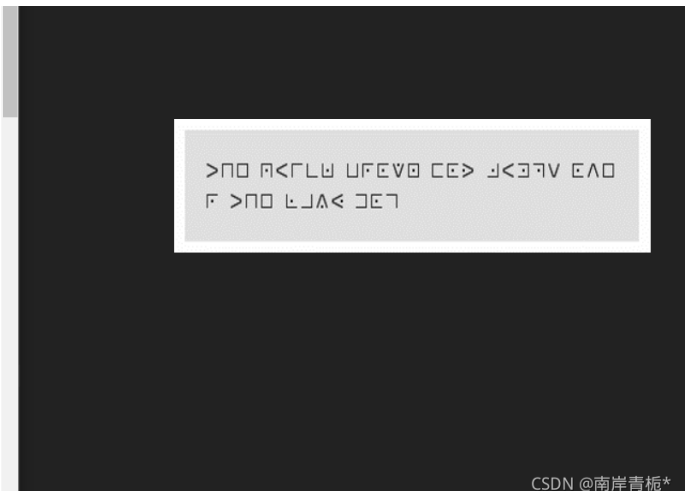
J	U	L	3	0	C	7	Π	Γ	J	Ω	E
3	0	E	7	Π	F	∨	>	<	^	∨	>
<	^										

加密的内容:

>ΠΟΑ<ΓΛΩUFEVΘCE> J<EΓV EΛΘF >ΠΟEJ^<EΓΓ

解密的内容:

thequickbrownfoxjumpsoverthelazydog



## 埃特巴什密码

埃特巴什.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

埃特巴什密码

分值: 50

密文: VU9aVFNaU1pYR1VSSE1SWFY=

CSDN @南岸青栀\*

首先看像base64编码

VU9aVFNaU1pYR1VSSE1SWFY=

清空

加密

解密

解密为UTF-8字节流

UOZTSZSZXGURHMRXV

CSDN @南岸青栀\*

常文: ABCDEFGHIJKLMNOPQRSTUVWXYZ

密文: ZYXWVUTSRQPONMLKJIHGFEDCBA

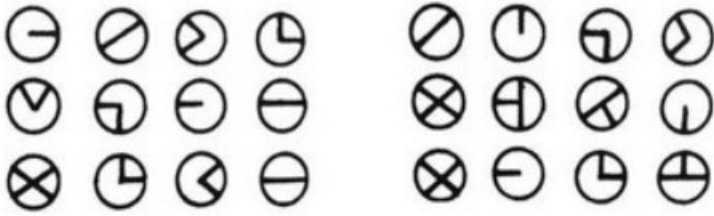
自己写一个脚本

```
arr1 = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K",  
        "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z"]  
arr2 = list(reversed(arr1))  
ciphertext = "UOZTSZSZXGURHMRXV"  
plaintext = ""  
for k in ciphertext:  
    plaintext += arr2[arr1.index(k)]  
print(plaintext)
```

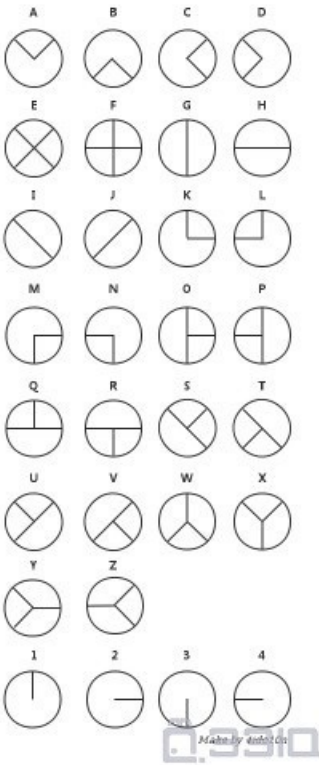


```
pythonLearning
Project
pythonLearning
External Libraries
Scratches and Consoles
Commit
Z-Structure
Z-Favorites
File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonLearning - 3-21埃特巴什解密.py - PyCharm
3-21埃特巴什解密
Git:
1 arr1 = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K",
2         "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z"]
3 arr2 = list(reversed(arr1))
4 ciphertext = "UOZTSZSZXGURHMRXV"
5 plaintext = ""
6 for k in ciphertext:
7     plaintext += arr2[arr1.index(k)]
8     print(plaintext)
for k in ciphertext
Run: 3-21埃特巴什解密
FLAGHAHACTFISNI
FLAGHAHACTFISNIC
FLAGHAHACTFISNICE
Process finished with exit code 0
PyCharm 2020.1.5 available
Update...
8:20 CRLF UTF-8 4 spaces CSDN@南岸青栀
```

## 夏多密码



对应解码就行



当铺密码





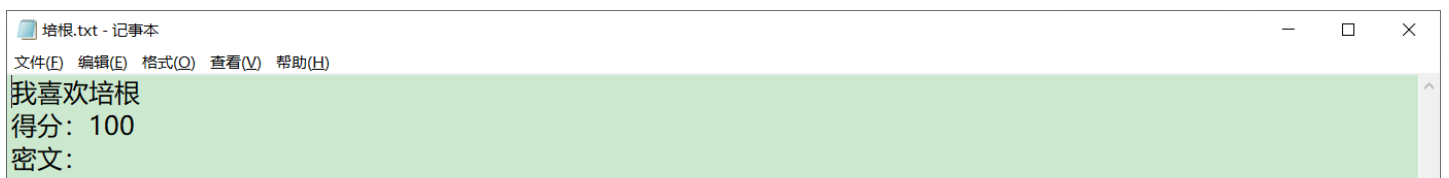
## unicode中文互转

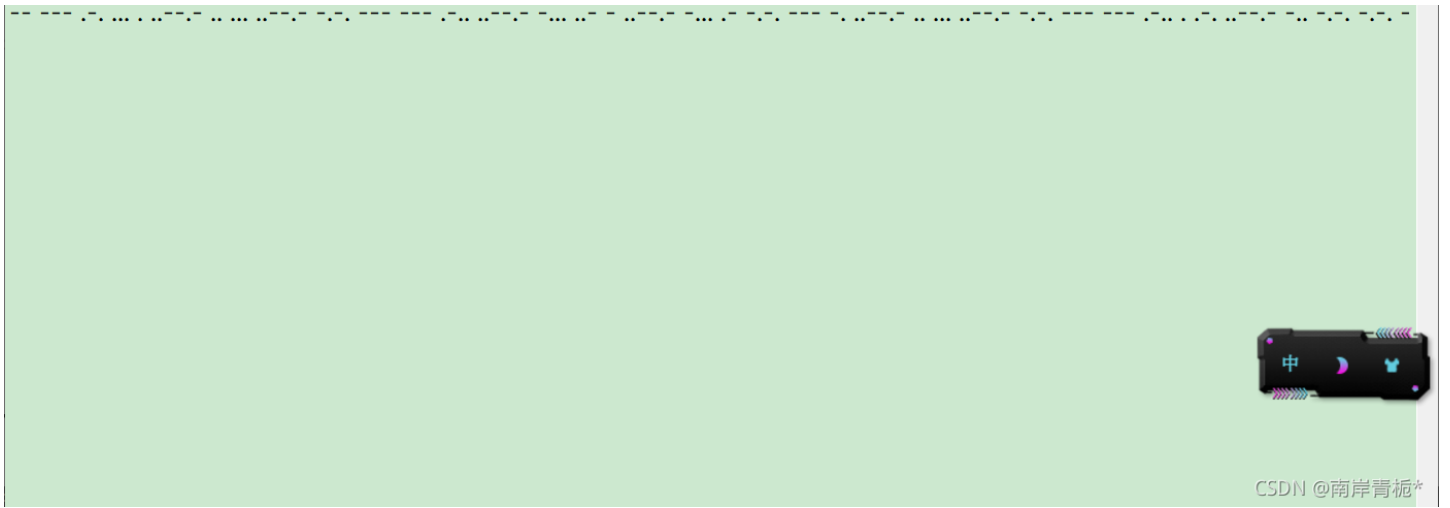
1 羊由大井夫大人王中工

得到 9 1 5 8 7 5 3 6 2 4, 不正确

通过binwalk发现图片中还有压缩文件

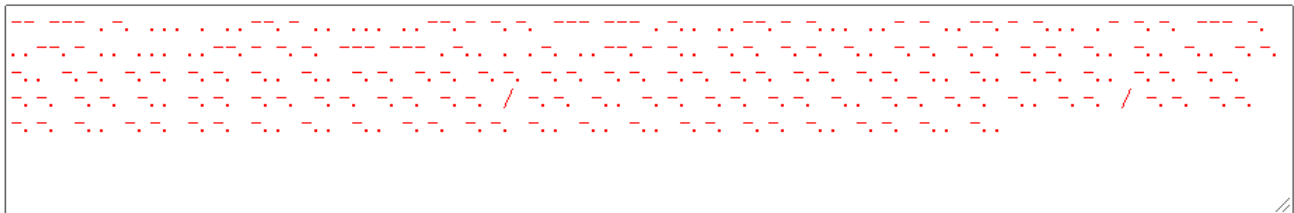
## 培根密码





### 1.首先看像是摩斯密码

输入摩尔斯电码，点击“解密”，即可将摩尔斯电码翻译成可识别的字符。



解密

**morse is cool but bacon is cooler**  
**dccdccdddcddccddccccccdddcddccddccccccddccddccdddcdddcdd**

CSDN @南岸青栀\*

### 2.因为题目说是培根，想到培根密码

Reverse Cipher

abbabbbaaababbbaabbbbbbbbaababbbbabbbbabbabbabbabbbaaabbaabbabaa

加密

解密

nyxhsxnrru

CSDN @南岸青栀\*

## 九宫格密码

收到一条奇怪的短信:

335321414374744361715332

你能帮我解出隐藏的内容嘛?!

格式: CTF{xxx}

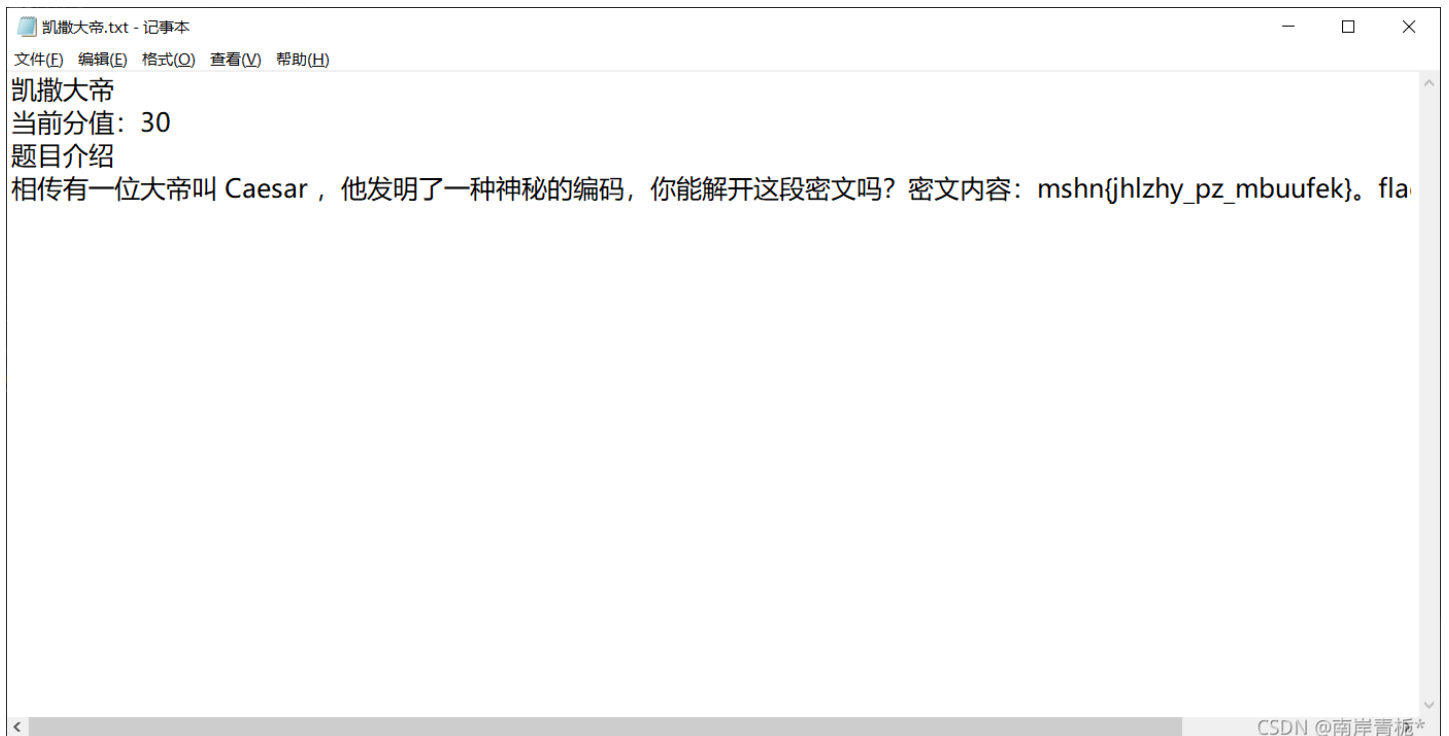
CSDN @南岸青栀\*

因为是手机收到的, 所以猜测和手机有关



书上说flagsimple, 但是我咋凑不出来呐?

## 凯撒大帝



写个脚本

```
cipher = "mshnjhlzhypzmbuufek" # mshn{jhlzhy_pz_mbuufek}
# flagcaesarisftnnxwd flag{caesar_is_ftnnxwd}
# print(ord("a"),ord("z")) 97 122
for n in range(1,26):
    tmp = []
    for i in cipher:
        # 使用了三目运算
        tmp.append(chr(ord(i)-n if ord(i)-n >= 97 else 122 - n + ord(i) - 97))
    print(''.join(tmp))
```

The screenshot shows a Python IDE with a file named '3-27凯撒加密.py'. The code in the editor is identical to the one in the first block. The output window shows the result of running the script:

```
E:\python编译软件\python.exe D:/桌面/pythonLearning/CTF脚本/3-27凯撒加密.py
lrgmigkygxoylattedj
kqflhfjxfwnxkyssdci
jpekgeiwevmwjxrrcbh
iodjfdhvdulviwqqbag
hnciecguctkuhvppayf
gmbhdbftbsjtguooyxe
flagcaesarisftnnxwd
ekyfbydryqhresmmwvc
```

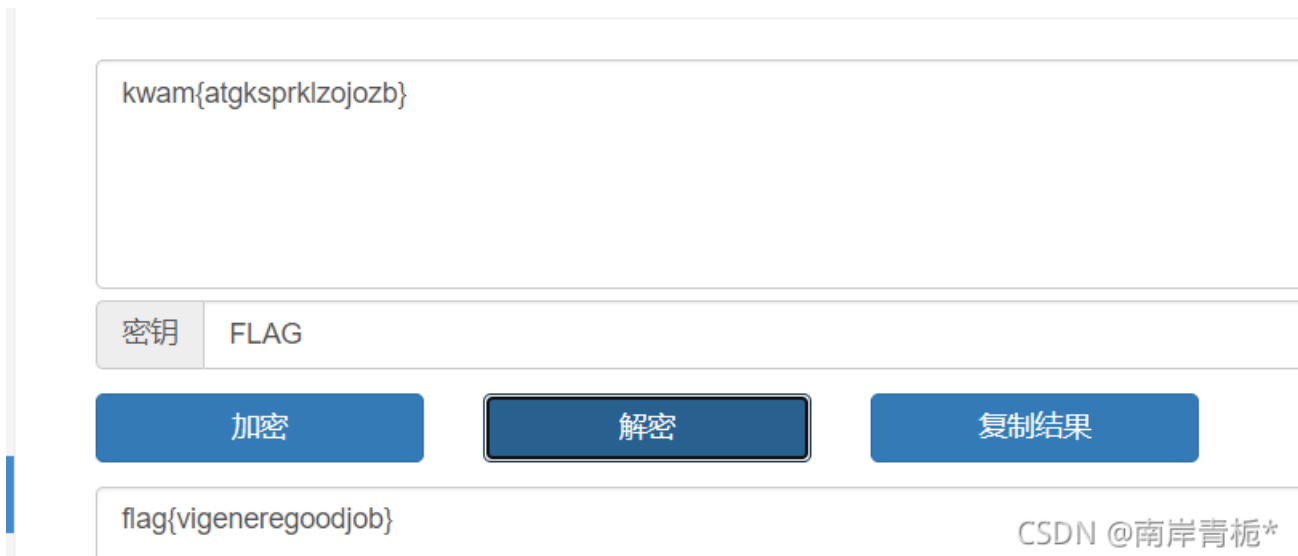
A red box highlights the output line 'flagcaesarisftnnxwd'. A red arrow points from the comment '# 使用了三目运算' in the code to the output, with the text '依次遍历解出flag' (Iteratively solve for flag) next to it.

CSDN @ 南岸青栀 2020

## 维吉尼亚密码



尝试了一下



Rabbit密码





U2FsdGVkX1+Dsr+FeBFLpdIXG4hw2/Q3eYTUMHvgr6WHxgb5

自定义密码, 例如: 123456, 如不需要密码时可以为空

Rabbit加密 Rabbit解密 清空输入框 复制结果文本

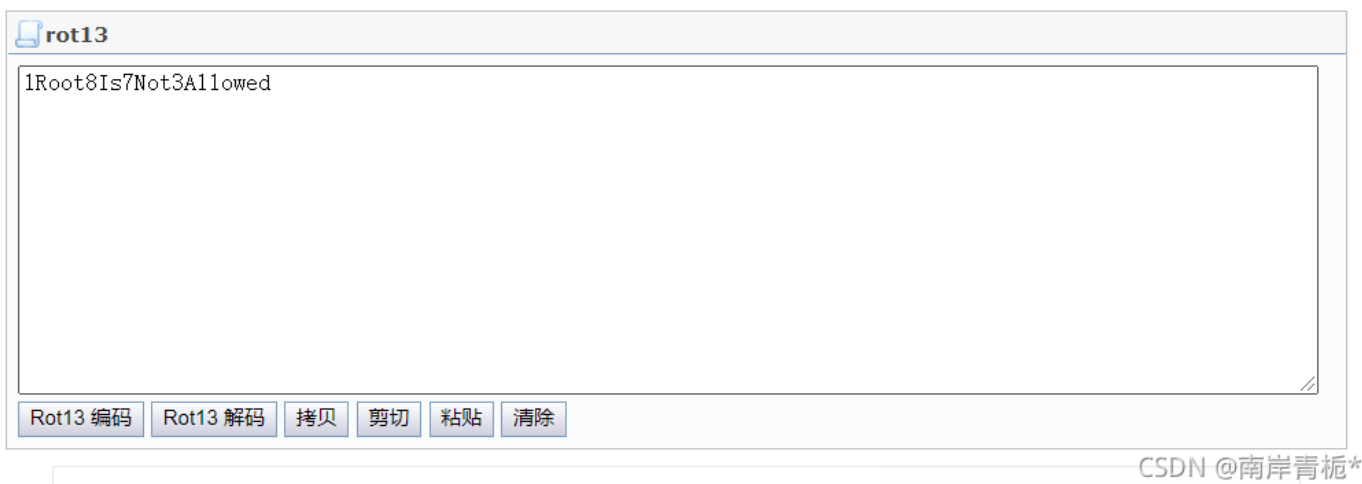
flag{rabbit\_123\_fun}

CSDN @南岸青栀\*

## Rot13密码



就是凯撒加密移位13



栅栏之困



看看题，凯撒被13栅栏用栅栏解密，间隔13

转换前: ✕

h{igr},aarclietflhf-\_peecirroc,eo\_fhlels\_caifnge

栏目数:

13

加密>

解密>

删除待加密内容空格

转换后: 📄

i,ctfer,flagishere-flaghrail\_cool\_fence\_cipher}{

CSDN @南岸青栀\*

## 关于同余方程、欧拉定理、乘法逆元、定义在 $Z_m$ 上的矩阵求逆

参考: [https://blog.csdn.net/Pioo\\_/article/details/110390802](https://blog.csdn.net/Pioo_/article/details/110390802)

这篇讲的还是蛮清楚的

### 1.模同余

模同余：给定一个正整数 $m$ ，如果两个整数 $a$ 和 $b$ 满足 $a-b$ 能够被 $m$ 整除，即 $(a-b)/m$ 得到一个整数，那么就称整数 $a$ 与 $b$ 模 $m$ 同余，记作 $a \equiv b \pmod{m}$ 。对模 $m$ 同余是整数的一个等价关系。

例如：

3被2除 余1

5被2除 余1

3, 5 被2除有相同的余数

所以 3 同余 5 模 1，记做： $3 \equiv 5 \pmod{1}$

## 2 若将 $a$ 和 $b$ 分别表示为

$$a = q_1 m + r_1, \quad b = q_2 m + r_2, \quad 0 \leq r_1, r_2 \leq m - 1,$$

则有

$$a \equiv b \pmod{m} \iff r_1 = r_2.$$

http://CSDN.net/@南岸青栀\*

同余的一些性质

### 同余的一些性质

如不特别声明，小写字母皆代表整数。

性质(1) 若 $x \equiv a \pmod{m}$ ，则 $x - a = hm$ 、 $x = a + hm$

性质(2) 同余是自反的， $a \equiv a \pmod{m}$

性质(3) 同余是对称的，若 $a \equiv b \pmod{m}$  则 $b \equiv a \pmod{m}$

性质(4) 同余是传递的，若 $a \equiv b \pmod{m}$ ， $b \equiv c \pmod{m}$ ，则 $a \equiv c \pmod{m}$

性质(5) 若 $x \equiv a \pmod{m}$ ， $y \equiv b \pmod{m}$ ，则 $x \pm y \equiv a \pm b \pmod{m}$

性质(6) 若 $x \equiv a \pmod{m}$ ，则 $x \pm b \equiv a \pm b \pmod{m}$

性质(7) 若 $x \equiv a \pmod{m}$ ，则 $nx \equiv na \pmod{m}$

性质(8) 若 $x \equiv a \pmod{m}$ ， $y \equiv b \pmod{m}$ ，则 $xy \equiv ab \pmod{m}$

性质(9) 若 $x \equiv a \pmod{m}$ ，则 $x^n \equiv a^n \pmod{m}$

性质(10) 若 $x \equiv a \pmod{m}$ 、 $d \mid m$ ，则 $x \equiv a \pmod{d}$  (其中 $d \mid m$  表示 $d$  整除 $m$ )

1

---

性质(11) 若 $ca \equiv cb \pmod{m}$ ， $d = \gcd(c, m)$  为 $c$  与 $m$  的最大公约数，则 $a \equiv b \pmod{\frac{m}{d}}$

性质(12) 若 $\gcd(c, m) = 1$ 、 $ca \equiv cb \pmod{m}$ ，则 $a \equiv b \pmod{m}$

http://CSDN.net/@南岸青栀\*

定理(1)

设 $a$ 、 $b$ 、 $m$  皆为整数，且 $\gcd(a, m) = d$ ，  
 (1) 若 $d \nmid b$ ，则 $ax \equiv b \pmod{m}$  无解；  
 (2) 若 $d \mid b$ ，则 $ax \equiv b \pmod{m}$  有解。

**定理(1)的证明**

(1)若 $d \nmid b$ ,

1. 若 $ax \equiv b \pmod{m}$  有解，存在整数 $p$ 、 $q$  使得 $ap - b = mq$ ,  $ap - mq = b$ ;
2. 因 $\gcd(a, m) = d$ ,  $d$  整除 $a$ 、 $m$ , 故 $d$  整除 $b = ap - mq$ , 是故与 $d \nmid b$  矛盾;
3. 故 $ax \equiv b \pmod{m}$  无解;

(2)若 $d \mid b$ ,

4. 由有关最大公因数的「裴蜀定理」：对于任意给定的正整数 $a, m$ , 必定存在整数 $p, q$  使得 $d = \gcd(a, m) = ap + mq$ ;
5. 因 $d \mid b$ ,  $b = rd = r(ap + mq) = a(rp) + rmq$ ;
6.  $a(rp) - b = -rmq$ ,  $a(rp) \equiv b \pmod{m}$ ,  $x = rp$  为同余方程的解。

**定理(2)**

若 $\gcd(a, m) = 1$ ,  $ax \equiv b \pmod{m}$  有对模 $m$  的唯一解。

**定理(2)的证明**

1. 显然,  $\gcd(a, m) = 1 \mid b$ , 由定理(1),  $ax \equiv b \pmod{m}$  有解;
2. 若 $x_0$ 、 $x_1$  为同余方程的两个解,
  - (a)  $ax_0 \equiv b \pmod{m}$ 、 $ax_1 \equiv b \pmod{m}$ ,
  - (b) 由性质(6),  $ax_0 - ax_1 \equiv b - b = 0 \pmod{m}$
  - (c) 因 $a(x_0 - x_1) \equiv 0 \pmod{m}$ ,  $m \mid a(x_0 - x_1)$ ,
  - (d) 因 $\gcd(a, m) = 1$ ,  $a$  与 $m$  互质,  $m \mid (x_0 - x_1)$ , 故 $x_0 \equiv x_1 \pmod{m}$
3. 故同余方程有唯一对模 $m$  的解。

<http://CSDN.net/南岸青栀>\*

**3.欧拉函数和欧拉定理**

## 乘法逆定义和性质

- ① 设  $a \in \mathbb{Z}_m$ , 若存在  $a' \in \mathbb{Z}_m$ , 使得

$$aa' \equiv 1 \pmod{m},$$

则称  $a$  模  $m$  可逆,  $a'$  为  $a$  的 (乘法) 逆元.  $a'$  可记为

$$a^{-1} \pmod{m},$$

在  $m$  固定的情形下, 简记为  $a^{-1}$ .

- ②  $a$  模  $m$  可逆, 当且仅当  $\gcd(a, m) = 1$ .  
③ 如果  $p$  为素数, 则  $\mathbb{Z}_p$  上任一非零元素均可逆.  
④ 如果  $\gcd(a, m) = 1$ , 则同余方程

$$ax \equiv b \pmod{m}$$

在  $\mathbb{Z}_m$  内有唯一解

$$x = (a^{-1}b) \pmod{m}.$$

<http://CSDN.net/@南岸青栀>

## 希尔密码



希尔.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

希尔  
当前分值: 80  
题目介绍  
密文: 22,09,00,12,03,01,10,03,04,08,01,17 (wjambkdeibr) ; 使用的矩阵是: 1 2 3 4 5 6 7 8 10; 请对密文解密。 f

Ln 1, Col 1 100% Windows (CRLF) ANSI 南岸青栀\*

## 加密:

密文向量 = 明文向量 \* 密钥矩阵 (mod 26)

1. 先将明文串对应英文字母编码表进行数字转化 4 0 18 19 2 7 8 13 0 13 14 17 12 0 11 20 13 8 21 4 17 18 8 19 24  
然后两两一组写成矩阵形式:

$$\begin{bmatrix} 4 & 18 & 2 & 8 & 0 & 14 & 12 & 11 & 13 & 21 & 17 & 8 & 24 \\ 0 & 19 & 7 & 13 & 13 & 17 & 0 & 20 & 8 & 4 & 18 & 19 & \square \end{bmatrix}$$

我去, 发现少了一个, 老师出题就不能凑个整吗??? 这样子, 我们做补0处理。

2. 接下来开始加密

$$\begin{aligned} X = E_k(x) &= x * K = \begin{bmatrix} 2 & 5 \\ 9 & 5 \end{bmatrix} \begin{bmatrix} 4 & 18 & 2 & 8 & 0 & 14 & 12 & 11 & 13 & 21 & 17 & 8 & 24 \\ 0 & 19 & 7 & 13 & 13 & 17 & 0 & 20 & 8 & 4 & 18 & 19 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 8 & 131 & 39 & 81 & 65 & 113 & 24 & 122 & 66 & 62 & 124 & 111 & 48 \\ 36 & 257 & 53 & 137 & 65 & 211 & 108 & 199 & 157 & 209 & 243 & 167 & 216 \end{bmatrix} \text{ mod } 26 \\ &= \begin{bmatrix} 8 & 1 & 13 & 3 & 13 & 9 & 24 & 18 & 14 & 10 & 20 & 7 & 22 \\ 10 & 23 & 1 & 7 & 13 & 3 & 4 & 17 & 1 & 1 & 9 & 11 & 8 \end{bmatrix} \end{aligned}$$

得到密文矩阵后, 按照分组对应的向量转成字母:

**IK BX NB DH NN JD YE SR OB KB UJ HL W**

CSDN @南岸青栀\*

## 仿射密码

参考: [https://blog.csdn.net/Pioo\\_/article/details/110235362](https://blog.csdn.net/Pioo_/article/details/110235362)



因为 $y = 3x + 9$

$$3x + 9 = y \pmod{26}$$

$$3x = (y - 9) \pmod{26}$$

$$x = 3^{-1}(y - 9) \pmod{26}$$

其中 $3^{-1}$ 是指3的乘法逆元, 不是3的导数

因为3的乘法逆元为9

$$\text{所以 } x = 9(y - 9) \pmod{26}$$

CSDN @南岸青枫\*

注意哈不是等号是三等奖

还是看不懂的就看看参考文章吧!!!



## 加密:

加密函数:  $E(x) = (7x + 3) \pmod{26}$

根据上面英文字母编码表得到X

明文	h	o	t
x	7	14	19
$7x+3$	52	101	136
$(7x+3) \pmod{26}$	0	23	6
密文	A	X	G

## 解密:

解密函数:

求得a的乘法逆元为15

$D(x) = 15(x - 3) \pmod{26}$

明文	a	x	g
x	0	23	6
$15(x-3)$	-45	300	45
$15(x-3) \pmod{26}$	7	14	19
解码	H	O	T

CSDN@南岸青栀\*

```
E:\python编译软件\python.exe D:
```

```
明文是: AFFINECRYPTO
```

```
密文是: JYYHWVPIDCOZ
```

代码

```

'''
仿射密码K = (a,b)
加密函数是E(x)= (ax + b) (mod 26)
解密函数为D(x) = (a^-1)(x - b) (mod 26)，其中a^-1是a的乘法逆元
'''

def fangsheEncode(a, b, e):
    cipher = []
    for i in e:
        tmp = chr(((ord(i) - 65) * a + b) % 26 + 65) # a ascii 97
        cipher.append(tmp)
    print("密文是: " + "".join(cipher).upper())
# 遍历求乘法逆元
def get_multiplicative_inverse(a):
    for i in range(0,100):
        if(a * i % 26 == 1):
            # print(i)
            return i

def fangsheDecode(a, b, e):
    ny = get_multiplicative_inverse(a)
    plaintext = []

    for i in e:
        tmp = ((ord(i) - 65) - b) * ny % 26 + 65 # A ascii 65
        plaintext.append(chr(tmp))
    print("明文是: " + "".join(plaintext).upper())

if __name__ == '__main__':
    a,b = 3,9
    e = "JYYHWVPIDCOZ"
    fangsheDecode(a, b, e)
    fangsheEncode(a,b,"AFFINECRYPTO")
    # print(ord("A")) 65

```

## 对称密码

 DES1.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

soulslayer:2aBl6E94luUfo

```
(root azb)-[~]
# hash-identifier
#####
#
#
#
#
#
#
#
#
#
#
#####
v1.2 #
By Zion3R #
www.Blackexploit.com #
Root@Blackexploit.com #
#####
-----
HASH: 2aBl6E94IuUfo

Possible Hashs:
[+] DES(Unix)
-----
HASH: █
```

### DES解密

## 解密DES

找遍了所有地方没有发现密钥。

据说给出的东西足够解出秘密了

U2FsdGVkX18fll8vjD2eBsbj7n77+YDHfY8mA9/B5fV7B6huFdkqlH4yqzAU/hCi

HaOLt3kKgCuBMv+9nzN5Eg==

答案格式: flag{xxx}

CSDN @南岸青栀\*

```
U2FsdGVkX18fll8vjD2eBsbj7n77+YDHfY8mA9/B5fV7B6huFdkqlH4yqzAU/hCi
HaOLt3kKgCuBMv+9nzN5Eg==
```

自定义密码, 例如: 123456, 如不需要密码时可以为空

DES加密

DES解密

清空输入框

复制结果文本

```
flag{DES_IS_ALSO_AN_INTRESTING_ENCRYPTO}
```

CSDN @南岸青栀\*

## DES-CBC

```
DES3.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
DES
当前分值: 50
题目介绍
DES CBC模式, IV为全0, key是abcd, 请解密0e97589c250e4ef717e9f9f74f3b7ea422c5b50d31ae9c62d8d6248700
Ln 1, Col 1 100% Windows (CRLF) CSDN@南岸青枫*
```

<https://gchq.github.io/CyberChef/> 解密网站