# CTF编码方式

原创

人多吃冰棍 于 2021-09-08 15:18:14 发布 478 收藏 2

文章标签： unctf



本文链接：https://blog.csdn.net/m0_51345650/article/details/120169064



## CTF编码方式

### base家族编码

> base16 / base32 / base64 / base58 / base85 / base 100 （另附不常见的 base 36 / base62 / base92 加解密）

**简述：**

**Base16编码是将二进制文件转换成由16个字符组成的文本**

**base32的编码表是由（A-Z、2-7）32个可见字符构成，“=”符号用作后缀填充。**

**base64的编码表是由（A-Z、a-z、0-9、+、/）64个可见字符构成，“=”符号用作后缀填充。**

**base58的编码表相比base64少了数字0，大写字母I，O，小写字母 l (这个是L），以及符号‘+’和‘/’**

**base91的密文由91个字符（0-9，a-z，A-Z,!#$%&()*+,./:;<=>?@[]^_`{|}~”）组成**

**Base100编码/解码工具（又名：Emoji表情符号编码/解码），可将文本内容编码为Emoji表情符号；同时也可以将编码后的Emoji表情符号内容解码为文本。**

**举例：**

> 明文：hello，world.123456

```
base16: 68656C6C6F2C776F726C642E313233343635
特征：大写字母(A-Z)和数字(0-9)，不用‘=’补齐。

base32: NBSWY3DPFR3W64TMMQXDCMRTGQ3DK===
特征：大写字母(A-Z)和数字(2-7)，不满5的倍数，用‘=’补齐。

base64: aGVsbG8sd29ybGQuMTIzNDY1
特征：大小写字母（A-Z，a-z）和数字（0-9）以及特殊字符‘+’，‘/’，不满3的倍数，用‘=’补齐。

base58: 2smDFYXWKE8vc8XA8dadEYcSqcQb
特征：相比Base64，Base58不使用数字"0"，字母大写"O"，字母大写"I"，和字母小写"l"，以及"+"和"/"符号，最主要的是后面不会出现'='
。

base85: BOu!rDst>tGAhM<A1fSl1GgsI
特征：特点是奇怪的字符比较多，但是很难出现等号

base91: TPwJh>go2Tv!_,aRA2IbLmA
特征：由91个字符（0-9，a-z，A-Z,!#$%&()*+,./:;<=>?@[]^_`{|}~”）组成，不支持中文。

base100: 🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆🍆
特征：就是一堆Emoji表情
```

**在线编码：**

base16 / base32 / base64

base58
base85
base91

base100

不常见的：base36(加密整数/解密字符串) base62 base92

## MD5、SHA1、HMAC、NTLM等类似加密型

### 1、MD5

**简述：**

一般MD5值是32位由数字“0-9”和字母“a-f”所组成的字符串，字母大小写统一；如果出现这个范围以外的字符说明这可能是个错误的md5值，就没必要再拿去解密了。

16位值是取的是8~24位。

**特征：**

有固定长度，一般是32位或者16位

由数字“0-9”和字母“a-f”组成

**举例：**

```
明文：hello，world.123456
md5(hello，world.123456,32) = 5189503aae1b1c0a6fbf7ea9e3128ab0
md5(hello，world.123456,16) = ae1b1c0a6fbf7ea9
```

**在线加解密**

MD5解密1

MD5解密2

## 2、SHA1

**简述**

SHA1是一种密码散列函数，SHA1可以生成一个被称为消息摘要的160位，20字节的散列值，散列值通常的呈现形式为**40位十六进制数**。这种加密和MD5类似。

**特征：**

有固定长度，为40位的字符串

**举例：**

```
明文：hello，world.123456
sha1（hello，world.123456）= 0179303b8f08fbc3d16cd23a4be5828790e12375
```

**在线加解密**

SHA1加解密1

SHA1加解密2

## 3、HMAC

**简述：**

HMAC (Hash-based Message Authentication Code) 常用于接口签名验证，这种算法就是在前两种加密的基础上引入了秘钥，而秘钥又只有传输双方才知道，所以基本上是破解不了的 。

**特征：**

和MD5类似，但是有秘钥。

**在线加解密：**

HAMC加解密

## 4、NTLM

**简述：**

这种加密是Windows的哈希密码，是 Windows NT 早期版本的标准安全协议。与它相同的还有Domain Cached Credentials（域哈希）。

**在线解密：**

NTML加解密

## 5、类似加密穷举

| # | 算法 | 长度 |
| --- | --- | --- |
| 1 | md5 | 32/16 |

| # | 算法 | 长度 |
|---|---|---|
| 2 | sha1 | 40 |
| 3 | sha256 | 64 |
| 4 | sha512 | 128 |
| 5 | adler32 | 8 |
| 6 | crc32 | 8 |
| 7 | crc32b | 8 |
| 8 | fnv132 | 8 |
| 9 | fnv164 | 16 |
| 10 | fnv1a32 | 8 |
| 11 | fnv1a64 | 16 |
| 12 | gost | 64 |
| 13 | gost-crypto | 64 |
| 14 | haval128,3 | 32 |
| 15 | haval128,4 | 32 |
| 16 | haval128,5 | 32 |
| 17 | haval160,3 | 40 |
| 18 | haval160,4 | 40 |
| 19 | haval160,5 | 40 |
| 20 | haval192,3 | 48 |
| 21 | haval192,4 | 48 |
| 22 | haval192,5 | 48 |
| 23 | haval224,3 | 56 |
| 24 | haval224,4 | 56 |
| 25 | haval224,5 | 56 |
| 26 | haval256,3 | 64 |
| 27 | haval256,4 | 64 |
| 28 | haval256,5 | 64 |
| 29 | joaat | 8 |
| 30 | md2 | 32 |
| 31 | md4 | 32 |
| 32 | ripemd128 | 32 |
| 33 | ripemd160 | 40 |

| # | 算法 | 长度 |
| --- | --- | --- |
| 34 | ripemd256 | 64 |
| 35 | ripemd320 | 80 |
| 36 | sha224 | 56 |
| 37 | sha3-224 | 56 |
| 38 | sha3-256 | 64 |
| 39 | sha3-384 | 96 |
| 40 | sha3-512 | 128 |
| 41 | sha384 | 96 |
| 42 | sha512/224 | 56 |
| 43 | sha512/256 | 64 |
| 44 | snefru | 64 |
| 45 | snefru256 | 64 |
| 46 | tiger128,3 | 32 |
| 47 | tiger128,4 | 32 |
| 48 | tiger160,3 | 40 |
| 49 | tiger160,4 | 40 |
| 50 | tiger192,3 | 48 |
| 51 | tiger192,4 | 48 |
| 52 | whirlpool | 128 |
| 53 | mysql | 老MYSQL数据库用的，16位，且第1位和第7位必须为0-8 |
| 54 | mysql5 | 40 |
| 55 | NTLM | 32 |
| 56 | Domain Cached Credentials | 32 |

## AES、DES、RC4、Rabbit、3DES型加密

**简述：**

以上都是非对称性加密算法，就是引入了密钥，密文特征与Base64类似。

**在线解密：**

非对称加密

## Unicode编码

**简述：**

Unicode（统一码、万国码、单一码）是一种在计算机上使用的字符编码。
它用两个字节来编码一个字符,字符编码一般用十六进制来表示.

**举例：**

Unicode有以下四种编码方式：

> 明文：hello，world.

```
&#x [hex]：&#x0068;&#x0065;&#x006C;&#x006C;&#x006F;&#xFF0C;&#x0077;&#x006F;&#x0072;&#x006C;&#x0064;&#x002E;

&# [hex]：&#00104;&#00101;&#00108;&#00108;&#00111;&#65292;&#00119;&#00111;&#00114;&#00108;&#00100;&#00046;

\u [hex]：\U0068\U0065\U006C\U006C\U006F\U002C\U0077\U006F\U0072\U006C\U0064\U002E

\u+ [hex]：\U+0068\U+0065\U+006C\U+006C\U+006F\U+FF0C\U+0077\U+006F\U+0072\U+006C\U+0064\U+002E
```

**在线编码：**

四种方式都有

unicode16进制

unicode

常见\u方式

## HTML实体编码

**简述：**

字符实体是用一个编号写入HTML代码中来代替一个字符，在使用浏览器访问网页时会将这个编号解析还原为字符以供阅读。

**举例：**

```
明文：hello，world.
十进制：&#104;&#101;&#108;&#108;&#111;&#65292;&#119;&#111;&#114;&#108;&#100;&#46;
十六进制：&#x68;&#x65;&#x6C;&#x6C;&#x6F;&#xFF0C;&#x77;&#x6F;&#x72;&#x6C;&#x64;&#x2E;
```

**在线加解密：**

HTML实体加解密1

HTML实体加解密2

## Escape、Unescape编码（%u）

**简述：**

**Escape/Unescape加密解码/编码解码**,又叫%u编码，其实就是字符对应UTF-16 16进制表示方式前面加%u。**Unescape解码/解密**，就是去掉"%u"后，将16进制字符还原后，由utf-16转码到自己目标字符。如：字符“中”，UTF-16BE是：“6d93”，因此Escape是“%u6d93”，反之也一样！

**举例：**

```
明文：hello，world.
密文：%u0068%u0065%u006c%u006c%u006f%uff0c%u0077%u006f%u0072%u006c%u0064%u002e
```

**在线加解密：**

Escape编码/解码

## URL编码

**简述：**

url编码又叫百分号编码，是统一资源定位(URL)编码方式。URL地址（常说网址）规定了常用地数字，字母可以直接使用，另外一批作为特殊用户字符也可以直接用（/,:@等），剩下的其它所有字符必须通过%xx编码处理。 现在已经成为一种规范了，基本所有程序语言都有这种编码，如js：有encodeURI、encodeURIComponent，PHP有 urlencode、urldecode等。编码方法很简单，在该字节ascii码的的16进制字符前面加%. 如 空格字符，ascii码是32，对应16进制是’20’，那么urlencode编码结果是:%20。

**特征：**

编码前面都有%

**举例：**

原文：https://baike.baidu.com/item/ABC

URL转码：
%68%74%74%70%73%3A%2F%2F%62%61%69%6B%65%2E%62%61%69%64%75%2E%63%6F%6D%2F%69%74%65%6D%2F%41%42%43

**在线加解密：**

URL加解密

## Hex编码

**简述：**

Hex 全称 是Intel HEX。Hex文件是由一行行符合Intel HEX文件格式的文本所构成的ASCII文本文件。在Intel HEX文件中，每一行包含一个HEX记录。这些记录由对应机器语言码和/或常量数据的**十六进制编码数字**组成。

**特征：**

十六进制（Hexadecimal）

它是计算机中数据的一种表示方法，**由0-9，A-F组 成，字母不区分大小写**。

与10进制的对应关系是：0-9不变，A-F对应10-15。

**举例：**

```
明文：hello，world.
密文（带%）：%68%65%6c%6c%6f%ef%bc%8c%77%6f%72%6c%64%2e
密文（不带%）：68656C6C6FEFBC8C776F726C642E
```

**在线加解密：**

不带%

带%

## js专用加密

### 1、JS颜文字加密

**特征：**

一堆颜文字构成的js代码，在F12中可直接解密执行

**举例：**

原代码：

```
alert("Hello, www.oicqzone.com")
```

加密后：

```
ﾟωﾟ/= / ｀m´）ﾉ ~┻━┻   //*´▽｀*/ ['_']; o=(ﾟｰﾟ)  =_=3; c=(ﾟΘﾟ) =(ﾟｰﾟ)-(ﾟｰﾟ); (ﾟДﾟ) =(ﾟΘﾟ)= (o^_^o)/ (o^_^o);(ﾟДﾟ)={ﾟΘﾟ: '_' ,ﾟωﾟﾉ : ((ﾟωﾟ/==3) +'_') [ﾟΘﾟ] ,ﾟｰﾟﾉ :(ﾟωﾟﾉ+ '_')[o^_^o -(ﾟΘﾟ)] ,ﾟДﾟﾉ:((ﾟｰﾟ==3) +'_')[ﾟｰﾟ] }; (ﾟДﾟ) [ﾟΘﾟ] =((ﾟωﾟ/==3) +'_') [c^_^o];(ﾟДﾟ) ['c'] = ((ﾟДﾟ)+'_') [ (ﾟｰﾟ)+(ﾟｰﾟ)-(ﾟΘﾟ) ];(ﾟДﾟ) ['o'] = ((ﾟДﾟ)+'_') [ﾟΘﾟ];(ﾟoﾟ)=(ﾟДﾟ) ['c']+(ﾟДﾟ) ['o']+(ﾟωﾟﾉ +'_')[ﾟΘﾟ]+ ((ﾟωﾟ/==3) +'_') [ﾟｰﾟ] + ((ﾟДﾟ) +'_') [(ﾟｰﾟ)+(ﾟｰﾟ)]+ ((ﾟｰﾟ==3) +'_') [ﾟΘﾟ]+((ﾟｰﾟ==3) +'_') [(ﾟｰﾟ) - (ﾟΘﾟ)]+(ﾟДﾟ) ['c']+((ﾟДﾟ)+'_') [(ﾟｰﾟ)+(ﾟｰﾟ)]+ (ﾟДﾟ) ['o']+((ﾟｰﾟ==3) +'_') [ﾟΘﾟ];(ﾟДﾟ) ['_'] =(o^_^o) [ﾟoﾟ] [ﾟoﾟ];(ﾟεﾟ)=((ﾟｰﾟ==3) +'_') [ﾟΘﾟ]+ (ﾟДﾟ) .ﾟДﾟ/+((ﾟДﾟ)+'_') [(ﾟｰﾟ) + (ﾟｰﾟ)]+((ﾟｰﾟ==3) +'_') [o^_^o -ﾟΘﾟ]+((ﾟｰﾟ==3) +'_') [ﾟΘﾟ]+ (ﾟωﾟﾉ +'_') [ﾟΘﾟ]; (ﾟｰﾟ)+=(ﾟΘﾟ); (ﾟДﾟ)[ﾟεﾟ]='\\'; (ﾟДﾟ).ﾟΘﾟ/=(ﾟДﾟ+ ﾟｰﾟ)[o^_^o -(ﾟΘﾟ)];(oﾟｰﾟo)=(ﾟωﾟﾉ +'_')[c^_^o];(ﾟДﾟ) [ﾟoﾟ]='\"';(ﾟДﾟ) ['_'] ( (ﾟДﾟ) ['_'] (ﾟεﾟ+(ﾟДﾟ)[ﾟoﾟ]+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ (ﾟΘﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ ((o^_^o) - (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+((ﾟｰﾟ) + (ﾟΘﾟ))+ (c^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟｰﾟ)+ ((o^_^o) - (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟΘﾟ)+ (c^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ ((ﾟｰﾟ) + (o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟｰﾟ)+ (c^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ ((ﾟｰﾟ) + (o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ ((ﾟｰﾟ) + (o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ ((ﾟｰﾟ) + (o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+((ﾟｰﾟ) + (ﾟΘﾟ))+ ((o^_^o) +(o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ ((ﾟｰﾟ) + (o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟΘﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ (o^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ (ﾟΘﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (o^_^o))+ ((o^_^o) - (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ ((ﾟｰﾟ) + (o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ ((o^_^o) +(o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+((ﾟｰﾟ) + (ﾟΘﾟ))+ ((o^_^o) +(o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ (o^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ ((ﾟｰﾟ) + (o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ ((o^_^o) - (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟΘﾟ)+ (ﾟДﾟ)[ﾟoﾟ]) (ﾟΘﾟ)) ('_');
```

**在线加密：**

JS颜文字加密 解密在F12的console中

### 2、Jother编码

**简述：**

jother是一种运用于javascript语言中利用少量字符构造精简的匿名函数方法对于字符串进行的编码方式。

**特征：**

只用 **! + ( ) [ ] { } **这八个字符就能完成对任意字符串的编码。可在F12中解密执行

**举例：**


**解密：**

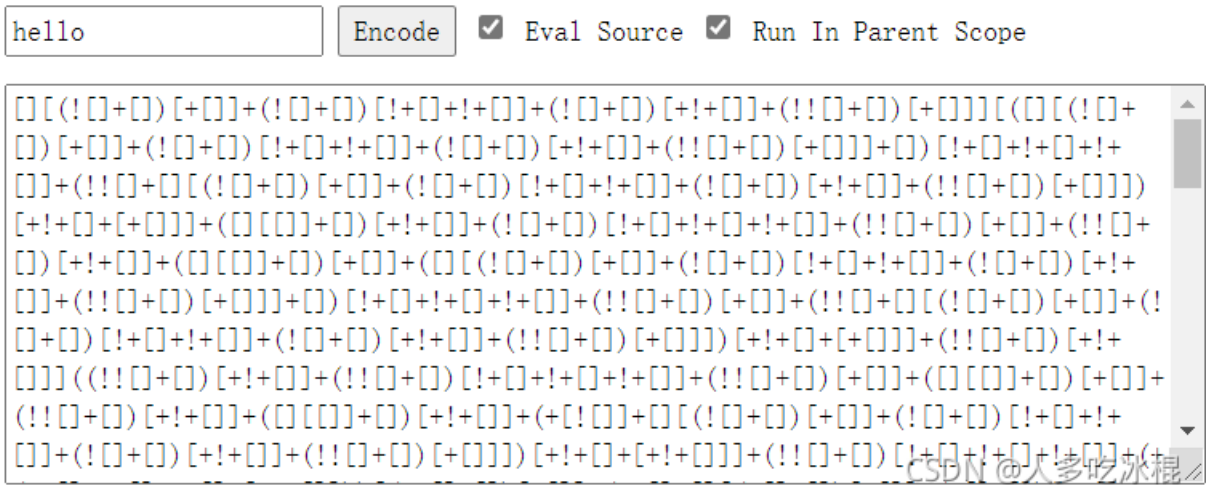
alert(xxx)、console(xxx)、document.write(xxx)即可（xxx为编码内容）。

### 3、JSFuck编码

**特征：**

**与jother很像，只是少了{}**

**举例：**




**在线加密：**

JSFuck加密 解密在F12的console中

## Quoted-printable编码

**简述：**

它是多用途互联网邮件扩展（MIME）一种实现方式。有时候我们可以邮件头里面能够看到这样的编码；

**特征：**

任何一个8位的字节值可编码为3个字符：一个等号”=”后跟随两个十六进制数字(0–9或A–F)表示该字节的数值.

**举例：**

```
明文：天上掉下了个猪八戒
密文：=E5=A4=A9=E4=B8=8A=E6=8E=89=E4=B8=8B=E4=BA=86=E4=B8=AA=E7=8C=AA=E5=85=AB=E6=88=92
```

**在线编码：**

在线编码

## XXencode

**简述：**

XXencode将输入文本以每三个字节为单位进行编码。如果最后剩下的资料少于三个字节，不够的部份用零补齐。这三个字节共有24个Bit，以6bit为单位分为4个组，每个组以十进制来表示所出现的数值只会落在0到63之间。以所对应值的位置字符代替。

**特征：**

字符范围是：

0-9，A-Z，a-z，

一共64个字符。跟base64打印字符相比，就是UUencode多一个“-” 字符，少一个”/” 字符。

**举例：**

```
明文：hello，world.
密文：BO4JgP4yXf5RjQalY9U++
```

**在线加解密：**

UUencode加解密

## UUencode

**简述：**

UUencode是一种二进制到文字的编码，最早在unix邮件系统中使用，全称：Unix-to-Unix encoding，UUencode将输入文本以每三个字节为单位进行编码，如果最后剩下的资料少于三个字节，不够的部份用零补齐。三个字节共有24个Bit，以6-bit为单位分为4个组，每个组以十进制来表示所出现的字节的数值。这个数值只会落在0到63之间。然后将每个数加上32，所产生的结果刚好落在ASCII字符集中可打印字符（32-空白…95-底线）的范围之中。

**举例：**

```
明文：hello,world.
密文：,:&5L;&\L=V]R;&0N
```

**在线解密&工具：**

UUencode1

UUencode2

## aaencode编码

**特征：**

将JS代码转换成常用的**网络表情**

**举例：**

Enter JavaScript source:

```
alert("Hello, JavaScript")
```

aaencode

```
ﾟωﾟﾉ= /｀ｍ´）ﾉ ~┻━┻   //*´∇｀*/ ['_']; o=(ﾟｰﾟ)  =_=3; c=(ﾟΘﾟ) =(ﾟｰﾟ)-(ﾟｰﾟ); (ﾟДﾟ) =(ﾟΘﾟ)= (o^_^o)/ (o^_^o);(ﾟДﾟ)=
[ﾟΘﾟ]= '_' ,ﾟωﾟﾉ : ((ﾟωﾟﾉ==3) +'_') [ﾟΘﾟ] ,ﾟｰﾟﾉ =(ﾟωﾟﾉ+ '_')[o^_^o -(ﾟΘﾟ)] ,ﾟДﾟﾉ=((ﾟｰﾟ==3) +'_')[ﾟｰﾟ] ]; (ﾟДﾟ) [ﾟΘﾟ]
=((ﾟωﾟﾉ==3) +'_') [c^_^o];(ﾟДﾟ) ['c'] = ((ﾟДﾟ)+'_') [ (ﾟｰﾟ)+(ﾟｰﾟ)-(ﾟΘﾟ) ];(ﾟДﾟ) ['o'] = ((ﾟДﾟ)+'_') [ﾟΘﾟ];(ﾟoﾟ)=(ﾟДﾟ)
['c']+(ﾟДﾟ) ['o']+(ﾟωﾟﾉ +'_')[ﾟΘﾟ]+ ((ﾟωﾟﾉ==3) +'_') [ﾟｰﾟ] + ((ﾟДﾟ) +'_') [(ﾟｰﾟ)+(ﾟｰﾟ)]+ ((ﾟｰﾟ==3) +'_') [ﾟΘﾟ]+((ﾟｰﾟ
==3) +'_') [(ﾟｰﾟ) - (ﾟΘﾟ)]+(ﾟДﾟ) ['c']+((ﾟДﾟ)+'_') [(ﾟｰﾟ)+(ﾟｰﾟ)]+ (ﾟДﾟ) ['o']+((ﾟｰﾟ==3) +'_') [ﾟΘﾟ];(ﾟДﾟ) ['_'] =
(o^_^o) [ﾟoﾟ] [ﾟoﾟ];(ﾟεﾟ)=((ﾟｰﾟ==3) +'_') [ﾟΘﾟ]+ (ﾟДﾟ) .ﾟДﾟﾉ+((ﾟДﾟ)+'_') [(ﾟｰﾟ) + (ﾟｰﾟ)]+((ﾟｰﾟ==3) +'_') [o^_^o -ﾟΘﾟ]+
((ﾟｰﾟ==3) +'_') [ﾟΘﾟ]+ (ﾟωﾟﾉ +'_') [ﾟΘﾟ]; (ﾟｰﾟ)+=(ﾟΘﾟ); (ﾟДﾟ)[ﾟεﾟ]='¥¥'; (ﾟДﾟ).ﾟΘﾟﾉ=(ﾟДﾟ+ ﾟｰﾟ)[o^_^o -(ﾟΘﾟ)];(oﾟｰﾟ
o)=(ﾟωﾟﾉ +'_')[c^_^o];(ﾟДﾟ) [ﾟoﾟ]='¥"';(ﾟДﾟ) ['_'] ( (ﾟДﾟ) ['_'] (ﾟεﾟ+(ﾟДﾟ)[ﾟoﾟ]+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ (ﾟΘﾟ)+
(ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +
(o^_^o))+ ((o^_^o) - (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+((ﾟｰﾟ) + (ﾟΘﾟ))+ (c^_^o)+ (ﾟДﾟ)
[ﾟεﾟ]+(ﾟｰﾟ)+ ((o^_^o) - (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟΘﾟ)+ (c^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟДﾟ)
[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+
((ﾟｰﾟ) + (o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟｰﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟｰﾟ)+ (c^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟΘﾟ)+ ((o^_^o) -
(ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ (ﾟΘﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ ((o^_^o) +(o^_^o))+ (ﾟДﾟ)[ﾟεﾟ]+(
ﾟΘﾟ)+ (ﾟｰﾟ)+ (ﾟΘﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((o^_^o) - (ﾟΘﾟ))+ (o^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ (ﾟｰﾟ)+ (o^_^o)+ (ﾟДﾟ)[ﾟεﾟ]+(
ﾟΘﾟ)+ ((o^_^o) +(o^_^o))+ ((o^_^o) - (ﾟΘﾟ))+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟΘﾟ)+ ((ﾟｰﾟ) + (ﾟΘﾟ))+ (ﾟΘﾟ)+ (ﾟДﾟ)[ﾟεﾟ]+(ﾟ
```



**在线加解密：**

aaencode1

aaencode2

aaencode3

## jjencode编码

**特征：**

将JS代码转换成**只有符号**的字符串

**举例：**

Hello, JavaScript

```
$=~[];$={___:++$,$$$$:(![]+"")[$],__$:++$,$_$_:(![]+"")[$],_$_:++$,$_$$:({}+"")[$],$$_$:($[$]+"")[$],_$$:++$,$$$_:(!""+"")[$],$__:++$,$_$:++$,$$__:({}+"")
[$],$$_:++$,$$$:++$,$___:++$,$__$:++$};$.$_=($.$_=$+"")[$.$_$]+($._$=$.$_[$.__$])+($.$$=($.$+"")[$.__$])+((!$)+"")[$._$$]+($.__=$.$_[$.$$_])+($.$=(!""+"")[$.__$])+($._=(!""+"")
[$._$_])+$.$_[$.$_$]+$.__+$._$+$.$;$.$$=$.$+(!""+"")[$._$$]+$.__+$._+$.$+$.$$;$.$=($.___)[$.$_][$.$_];$.$($.$($.$$+"\""+"\\"+$.__$+$.__$+$.___+$.$$$_+(![]+"")[$._$_]+(![]+"")
[$._$_]+$._$+",\\"+$.$__+$.___+"\\"+$.__$+$._$+$.$_$_+"\\"+$.__$+$.$$_+$.$$_+$.$_$_+"\\"+$.__$+$._$+$._$$+$.$$__+"\\"+$.__$+$.$$_+$._$_+"\\"+$.__$+$.$_$+$.__$+"\\"+
$.__$+$.$$_+$.___+$.__+"\\"+$.__$+$._$_+"\\"+$.__$+$.__$+"\\"+$.__$+$.__$+"\\"+$.__$+$.__$+"\"")())();
```

**在线加解密：**

jjencode加密

jjencode解密

## brainfuck编码

**简述：**

Brainfuck是一种极小化的计算机语言，按照"Turing complete（完整图灵机）"思想设计的语言，它的主要设计思路是：用最小的概念实现一种“简单”的语言。

**特征：**

BrainFuck 语言只有八种符号，所有的操作都由这八种符号 **(> < + - . , [ ])** 的组合来完成。

**举例：**

```
明文：hello,world.
密文：+++++ +++++ [->++ +++++ +++<] >++++ .---. +++++ ++..+ ++.<+ +++++ ++[->
----- ---<] >---. <++++ ++++[ ->+++ +++++ <]>++ +++++ ++++. ----- ---.+
++.-- ----. ----- ---.< +++++ ++[-> ----- --<]> ----- .<
```

**在线加解密：**

Brainfuck1

Brainfuck2

## 莫尔斯电码

**简述：**

摩尔斯电码(Morse Code)是由美国人萨缪尔·摩尔斯在1836年发明的一种时通时断的且通过不同的排列顺序来表达不同英文字母、数字和标点符号的信号代码，摩尔斯电码主要由以下5种它的代码组成：

1. 点（.）
2. 划（-）
3. 每个字符间短的停顿（通常用空格表示停顿）
4. 每个词之间中等的停顿（通常用 `/` 划分）
5. 以及句子之间长的停顿

莫尔斯电码对应表：

| 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 --.. | 0 ----- | 1 .---- | 2 ..--- | 3 ...-- |
| 4 ....- | 5 ..... | 6 -.... | 7 --... | 8 ---.. | 9 ----. | . .-.-.- | , --..-- | ? ..--.. | - -....- |

| A .- | B -... | C -.-. | D -.. | E . | F ..-. | G –. | H .... | I .. | J .— |
|---|---|---|---|---|---|---|---|---|---|
| = -...- | : —... | ; -.-.-. | ( -.–. | ) -.–.- | / -..-. | " .-..-. | $ ...-..- | ’ .----. | ¶ .-.-... |
| _ ..–.- | @ .–.-. | ! —. | ! -.-.– | + .-.-. | ~ .-... | # ..-.- | & . ... | ⁄ -..-. | |

**特征：**

由 **. - “空格” / **表示。

**在线加解密：**

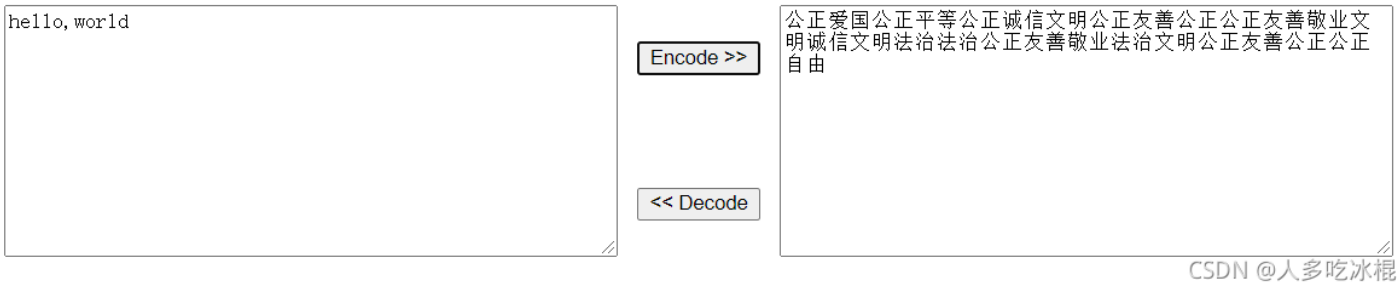
莫斯电码1

## 社会主义编码

**特征：**

字符全部是社会主义核心价值观。

**举例：**





**在线加解密：**

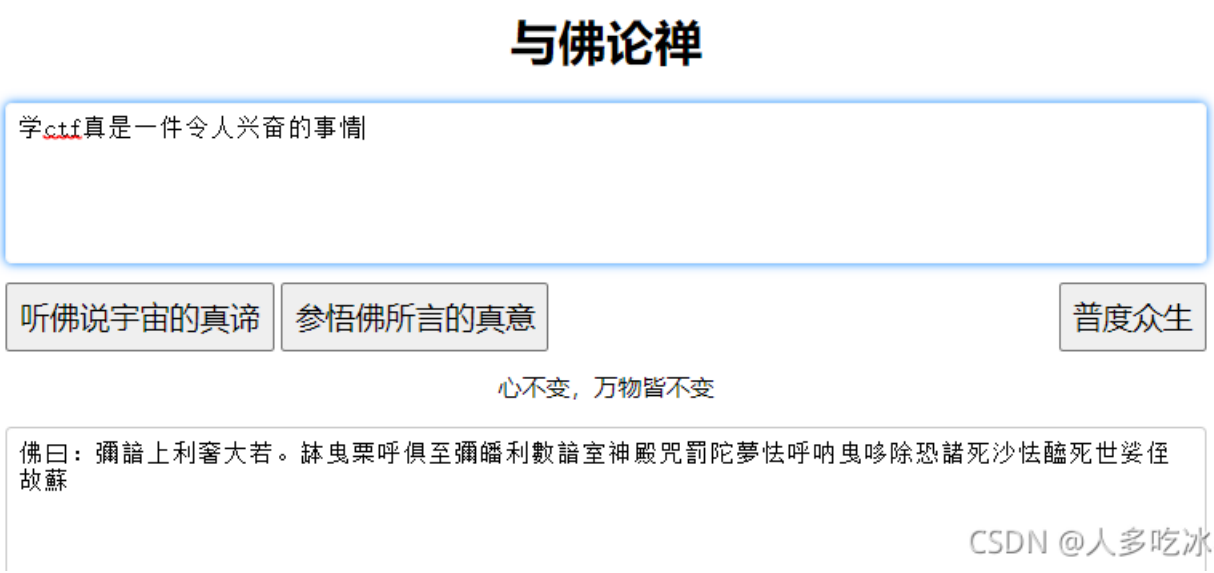
社会主义编码

## 与佛论禅

**特征：**

密文以”佛曰：如是我闻：”开头，密文一般是与关佛经的汉字

**举例：**




**在线工具：**

与佛论禅

## 猪圈密码

**特征：**

明文: abcdefghijklmnopqrstuvwxyz:<>?@;=[\]^_`{}|~%+



**在线工具：**

猪圈密码

## Ook!

**特征：**

密文都是Ook+（./?/!）的形式构成。

**举例：**

原文：nice

加密：

Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook! Ook! Ook. Ook? Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook? Ook. Ook? Ook! Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook! Ook! Ook! Ook!
Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook. Ook. Ook. Ook. Ook. Ook!
Ook. Ook? Ook.

**在线工具：**

Brainfuck/Text/Ook!