# base64及其隐写原理

原创

R0be11 于 2020-03-14 21:27:47 发布 881 收藏 2
分类专栏： CTF

本文链接：https://blog.csdn.net/robacco/article/details/104867279





## 一、base64编码

**base64就是一种基于64个可打印字符来表示二进制数据的表示方法，包含字符"A-Z、a-z、+、/"64个字符。编码表如下图：**

| 数值 | 字符 | 数值 | 字符 | 数值 | 字符 | 数值 | 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 | / |



**base64编码原理：**

base64由64个字符组成，使用6位即可表示，而一个字节由8位二进制组成，所以最后需要将8位拆分为6位二进制。

例如给定一个字符，首先需要对应ASCII码表的值，并将其转换为二进制，如“r”对应114，二进制为01110010。接着需要把8位二进制划分为6位，不足6位的在后面补零，使其是6的倍数。然后将其对应base64码表的字符，000000对应“=”即可。

例 01110010 （8位）

—>011100 100000 000000 000000 （6位）

—> 28 32 = = （十进制）

—> c g = = （base64码表符号）

即字符“r”进行base64编码后为“cg==”。

**解码就是编码的逆过程：**

首先去掉等号, 再根据base64码表将对应字符转为二进制数，然后从左到右, 每8 个位一组, 并将多余位扔掉, 转为对应的 ASCII 码即可。

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 | +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 | ^Back space |

注：表中的ASCII字符可以用:ALT + “小键盘上的数字键” 输入

**base64隐写原理：**

回顾上面的例子，图中红色部分就是隐写位（4位），因为解码过程中会丢掉多余的位（就是红色），所有在其中进行隐写不会影响解码的结果。

例 01110010 （8位）

—>011100 100000 000000 000000 （6位）

—> 28 32 = = （十进制）

—> c g = = （base64码表符号）

即字符“r”进行base64编码后为“cg==”。

若把红色的位改成如下：

例 01110010 （8位）

—>011100 101000 000000 000000 （6位）

—> 28 40 = = （十进制）

—> c o = = （base64码表符号）

即字符“r”进行base64编码后为“co==”，解码结果仍然和“cg==”一样是字符“r”，所以在隐写位进行隐写不会修改最终的解码值。

**接下来再举一例，编码字符“ro”**

r o

01110010 01101111

—>011100 100110 1111**00** 000000 （红色代表隐写位，2位）

—> 28 38 60 =

—> c m 8 = （base64码表符号）

即字符“ro”进行base64编码后为“cm8=”。

**从以上两个例子可以得出：若编码结尾有一个“=”，表示可隐写2位，两个“=”表示可隐写4位。**