

MD5 摘要压缩算法

Illusionna

2024 年 4 月 30 日

参考文献

- [1] Rivest, Ronald. “RFC1321: The MD5 message-digest algorithm”. (1992).
- [2] <https://www.rfc-editor.org/info/rfc1321>
- [3] Python 和 C 语言代码在主页上: <https://Illusionna.github.io/cryptography.github.io>

问题

我们先从一个实例开始: 诺亚方舟是《圣经》中记载的史前世界遭受大洪水时代袭击时, 诺亚为拯救他和家人以及各种地球上雌雄生物而建造的大木船, 它的英语单词是“Ark”. 有一天诺亚将一只鸽子放飞出去, 想让鸽子带着消息“Ark”去西奈山见上帝. 但与此同时, 他害怕外面的世界有人发现鸽子喙里的消息, 所以诺亚对“Ark”进行单向散列函数消息摘要加密. 把“Ark”变成一个固定长度的散列信息, 在不知原文的情况下, 任何人都无法由散列信息逆推出原文, 当然, 只有上帝能根据摘要密文知晓原文“Ark”.

原理

“Ark” 可以看作 3 个 utf-8 编码下的字符共同构成的, $\text{Ark} = \text{A} + \text{r} + \text{k}$. 把 “A” 转化成 utf-8 编码下的 0-1 比特流:

$$\text{“A”} \mapsto 65(\text{Decimal}) \mapsto 41(\text{Hexadecimal}) \mapsto 01000001(\text{Binary}) \quad (1)$$

同样的, 得到整个单词在 utf-8 编码下的 0-1 比特流:

$$\text{“Ark”} \mapsto 01000001, 01110010, 01101011(\text{Binary}) \quad (2)$$

MD5 算法大致分为三步:

1. 处理原文比特流: 进行零比特填充和原文消息长度填充;
2. 初始化全局幻数: 用于迭代输出最后的 MD5 加密结果;
3. 迭代加密运算: 通过比特流非线性位运算不断更新迭代.

1. 处理原文

MD5 算法是每 512 比特为一块进行分组处理, 而原文消息 “Ark” 在 utf-8 编码下只有 $3 \times 8 = 24$ 比特, 因此为了凑齐 512 比特, 需要对原文进行一定规则的填充.

MD5 规定: 有效信息原文的下一字节的首比特位值为 1, 其余比特位全为 0, 并且最后 8 字节 (64 比特位) 存储有效原文比特流长度, 除此之外, 还要保证填充完成后的比特流长度是 512 的正整数倍.

$$(3 \times 8 + 1 \times 8 + 52 \times 8 + 8 \times 8) \% 512 = 0 \quad (3)$$

$$\underbrace{01000001, 01110010, 01101011}_{\text{Ark 3bytes}}, \underbrace{10000000, 00000000, \dots, 00000000}_{53\text{bytes}}, \underbrace{00011000, 00000000, \dots, 00000000}_{8\text{bytes}} \quad (4)$$

最后 8 字节的第一字节 00011000 对应十进制就是 24, 也就是有效原文比特流长度. 若设有效原文字节流长度为 \mathcal{N} , 零比特填充的零字节流长度为 \mathcal{M} , 由 MD5 约束条件得:

$$(\mathcal{N} \times 8 + 1 \times 8 + \mathcal{M} \times 8) \% 512 = 448 \quad (5)$$

$$\implies \mathcal{M} = 64\mathcal{K} + 55 - \mathcal{N} \geq 0, \quad \mathcal{K} = 0, 1, 2, 3, \dots \quad (6)$$

因为有效原文字节流长度 \mathcal{N} 是可以通过遍历而得知的, 是一个已知量. 为了使得式 6 最小 \mathcal{K} 恒成立, 从计算机编程角度而言, 只需要进行一次取模 (mod) 运算即可确定下来零比特填充的字节流长度 \mathcal{M} , 让时间复杂度降低为 $\mathcal{O}(1)$.

2. 初始化全局幻数

RFC-1321 规定 [1] 标准 MD5 迭代的四个初值 (亦称幻数) 分别为 32 比特位, 十六进制整型取值依次为: 01 23 45 67, 89 ab cd ef, fe dc ba 98, 76 54 32 10, 正好构成一个轮回. 由于计算机中字节小端位在前, 因此在编程中可以表示为整型值:

```
1 >>> A = 0x67452301 = 1732584193
2 >>> B = 0xefcdab89 = 4023233417
3 >>> C = 0x98badcfe = 2562383102
4 >>> D = 0x10325476 = 271733878
```

转化为十进制流和 0-1 比特流:

```
1 >>> 0x67452301 -> 103 69 35 1 -> 01100111 01000101 00100011 00000001
2 >>> 0xefcdab89 -> 239 205 171 137 -> 11101111 11001101 10101011 10001001
3 >>> 0x98badcfe -> 152 186 220 254 -> 10011000 10111010 11011100 11111110
4 >>> 0x10325476 -> 16 50 84 118 -> 00010000 00110010 01010100 01110110
```

3. 定义辅助函数

在迭代前, 定义以下几种计算机非线性的位运算辅助函数:

- a) $F(X, Y, X) = (X \& Y) \mid ((\sim X) \& Z)$
- b) $G(X, Y, Z) = (X \& Z) \mid (Y \& (\sim Z))$
- c) $H(X, Y, Z) = X \wedge Y \wedge Z$
- d) $I(X, Y, Z) = Y \wedge (X \mid (\sim Z))$
- e) LoopShiftLeft 循环左移函数 = $\mathcal{L}(X, s) = (X \ll s) \mid (X \gg (32 - s))$
- f) $T(x) = \lfloor |\sin(x + 1)| \times 4294967296 \rfloor$, $x = 1, 2, \dots, 64$ 弧度制

其中 4294967296 (Decimal) = 100000000, 000000000, 000000000, 000000000 (Binary), 并且作者 Ronald L. Rivest [1] 规定循环左移偏量 s 是迭代次数的函数, 也就是已知的定值.

$$\begin{aligned} S &= 4 * [7, 12, 17, 22] + 4 * [5, 9, 14, 20] + 4 * [4, 11, 16, 23] + 4 * [6, 10, 15, 21] \\ &= [7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 5, 9, 14, 20, \dots, 6, 10, 15, 21]_{1 \times 64} \end{aligned} \quad (7)$$

4. 循环迭代

算法如下表 1 所示.

表 1: MD5 迭代算法

Algorithm 1: MD5 迭代过程 (列表索引按照数学思维从 1 开始, 大多数编程索引从 0 开始)

Input: 式 4 的十进制列表 $\Omega = \underbrace{65, 114, 107}_{\text{Ark 3bytes}}, \underbrace{128, 0, 0, \dots, 0}_{53\text{bytes}}, \underbrace{24, 0, 0, \dots, 0}_{8\text{bytes}}$ 与幻数

Output: 长度为 32 的十六进制字符串

```

1 for  $i = 1 \rightarrow ((\mathcal{N} + 1 + \mathcal{M} + 8) \bmod 64)$  /*  $\mathcal{N} + 1 + \mathcal{M} + 8 = \Omega$  的长度 */ do
2    $AA = A; BB = B; CC = C; DD = D;$ 
3   for  $j = 1 \rightarrow 64$  do
4      $aa = DD; bb = AA; cc = BB; dd = CC;$ 
5      $a = AA; b = BB; c = CC; d = DD;$ 
6     if  $1 \leq j \leq 16$  then
7        $k = 64(i - 1) + 4(j - 1) + 1;$ 
8        $X(k) = \mathbb{H}(\Omega[k], \Omega[k + 1], \Omega[k + 2], \Omega[k + 3]);$ 
9       /* 例如:  $\mathbb{H}(65, 114, 107, 128) \mapsto \underline{0x41\ 0x72\ 0x6B\ 0x80} = 0x806B7241$  */;
10       $a = b + \mathcal{L}(a + F(b, c, d) + X(k) + T(j), \mathcal{S}[j]);$ 
11    end
12    if  $17 \leq j \leq 32$  then
13       $k = 64(i - 1) + 4\{[5(j - 1) + 1] \% 16\} + 1;$ 
14       $X(k) = \mathbb{H}(\Omega[k], \Omega[k + 1], \Omega[k + 2], \Omega[k + 3]);$ 
15       $a = b + \mathcal{L}(a + G(b, c, d) + X(k) + T(j), \mathcal{S}[j]);$ 
16    end
17    if  $33 \leq j \leq 48$  then
18       $k = 64(i - 1) + 4\{[3(j - 1) + 5] \% 16\} + 1;$ 
19       $X(k) = \mathbb{H}(\Omega[k], \Omega[k + 1], \Omega[k + 2], \Omega[k + 3]);$ 
20       $a = b + \mathcal{L}(a + H(b, c, d) + X(k) + T(j), \mathcal{S}[j]);$ 
21    end
22    if  $49 \leq j \leq 64$  then
23       $k = 64(i - 1) + 4\{[7(j - 1)] \% 16\} + 1;$ 
24       $X(k) = \mathbb{H}(\Omega[k], \Omega[k + 1], \Omega[k + 2], \Omega[k + 3]);$ 
25       $a = b + \mathcal{L}(a + I(b, c, d) + X(k) + T(j), \mathcal{S}[j]);$ 
26    end
27     $bb = a;$ 
28     $AA = aa; BB = bb; CC = cc; DD = dd;$ 
29  end
30   $A = A + AA; B = B + BB; C = C + CC; D = D + DD;$ 
31 end
32  $A = 0x1e23a4ef; B = 0xd556c324; C = 0xf059a225; D = 0x4e4004b2;$ 
33 “Ark”  $\mapsto \mapsto \rightsquigarrow$  md5 = “efa4231e24c356d525a259f0b204404e”

```
