SM3密码杂凑算法原理与数学推导

1. 算法概述

SM3是中国国家密码管理局于2010年发布的密码杂凑算法标准，收录于GB/T 32905-2016《信息安全技术 SM3密码杂凑算法》。该算法适用于数字签名、消息认证、数据完整性校验等场景，其安全性与SHA-256相当，输出长度为256位（32字节）。

SM3算法的处理过程主要包括：

1. 消息填充
2. 消息扩展
3. 压缩函数迭代

2. 数学基础

2.1 数据表示

SM3处理的消息为字节流，所有运算均在32位无符号整数域内进行，运算结果按模2^32处理。

2.2 常量与函数定义

常量：

- 初始向量IV：

```
IV = [0x7380166F, 0x4914B2B9, 0x172442D7, 0xDA8A0600,
      0xA96F30BC, 0x163138AA, 0xE38DEE4D, 0xB0FB0E4E]
```

- 常量T：

```
T_j = 0x79CC4519,  0 ≤ j ≤ 15
T_j = 0x7A879D8A, 16 ≤ j ≤ 63
```

函数：

1. 循环左移函数：

$$RL(x, n) = (x \ll n) \oplus (x \gg (32 - n))$$

   其中≪表示左移，≫表示右移，运算结果取32位。

2. 置换函数P0：

$$P_0(x) = x \oplus RL(x, 9) \oplus RL(x, 17)$$

3. 置换函数P1：

$$P_1(x) = x \oplus RL(x, 15) \oplus RL(x, 23)$$

4. 布尔函数FF：

$$FF_j(x,y,z)=\begin{cases}x\oplus y\oplus z, & 0\le j\le 15\\ (x\wedge y)\vee(x\wedge z)\vee(y\wedge z), & 16\le j\le 63\end{cases}$$

   其中$\oplus$表示异或，$\wedge$表示与，$\vee$表示或。

5. 布尔函数GG：

$$GG_j(x,y,z)=\begin{cases}x\oplus y\oplus z, & 0\le j\le 15\\ (x\wedge y)\vee(\neg x\wedge z), & 16\le j\le 63\end{cases}$$

   其中$\neg$表示非。

3. 算法步骤

3.1 消息填充

SM3要求输入消息的长度（以比特为单位）L满足L < 2^64。填充过程如下：

1. 在消息末尾添加一个比特"1"
2. 添加k个比特"0"，其中k是满足L + 1 + k ≡ 448 mod 512的最小非负整数
3. 添加一个64位的整数，该整数是原始消息长度L的二进制表示（小端序）

填充后的消息长度为512的整数倍，可以表示为：

$$m'=m||1||0^k||(L \mod 2^{64})$$

其中$||$表示比特串连接。

3.2 消息扩展

将填充后的消息按512比特分组，得到消息块$B^{(0)},B^{(1)},...,B^{(n-1)}$，其中n = 填充后消息长度 / 512。

对每个消息块$B^{(i)}$，进行如下扩展：

1. 将$B^{(i)}$分为16个32位字$W_0,W_1,...,W_{15}$
2. 扩展生成$W_{16}\sim W_{67}$：

$$W_j=P_1(W_{j-16}\oplus W_{j-9}\oplus RL(W_{j-3},15))\oplus RL(W_{j-13},7)\oplus W_{j-6}$$

   其中16 ≤ j ≤ 67
3. 生成$W'_0\sim W'_{63}$：

$$W'_j=W_j\oplus W_{j+4}$$

   其中0 ≤ j ≤ 63

3.3 压缩函数

压缩函数将当前的链接值$V^{(i)}$和消息块$B^{(i)}$压缩为新的链接值$V^{(i+1)}$。

初始链接值$V^{(0)} = IV$。

对于每个消息块$B^{(i)}$，压缩过程如下：

1. 初始化8个工作变量：

$$A = V_0^{(i)}, B = V_1^{(i)}, C = V_2^{(i)}, D = V_3^{(i)}$$

$$E = V_4^{(i)}, F = V_5^{(i)}, G = V_6^{(i)}, H = V_7^{(i)}$$

2. 进行64轮迭代：

$$SS1 = RL((RL(A, 12) + E + RL(T_j, j)) \mod 2^{32}, 7)$$

$$SS2 = SS1 \oplus RL(A, 12)$$

$$TT1 = (FF_j(A, B, C) + D + SS2 + W_j') \mod 2^{32}$$

$$TT2 = (GG_j(E, F, G) + H + SS1 + W_j) \mod 2^{32}$$

$$D = C$$

$$C = RL(B, 9)$$

$$B = A$$

$$A = TT1$$

$$H = G$$

$$G = RL(F, 19)$$

$$F = E$$

$$E = P_0(TT2)$$

3. 计算新的链接值：

$$V^{(i+1)} = (A \oplus V_0^{(i)}, B \oplus V_1^{(i)}, C \oplus V_2^{(i)}, D \oplus V_3^{(i)},$$

$$E \oplus V_4^{(i)}, F \oplus V_5^{(i)}, G \oplus V_6^{(i)}, H \oplus V_7^{(i)})$$

3.4 输出结果

处理完所有消息块后，最终的哈希值为：

$$SM3(m) = V_0^{(n)} || V_1^{(n)} || V_2^{(n)} || V_3^{(n)} || V_4^{(n)} || V_5^{(n)} || V_6^{(n)} || V_7^{(n)}$$

其中||表示32位字的连接，结果为256位的哈希值。

4. 测试向量

1. 空消息：

```
SM3("") = 1ab21d8355cfa17f8e61194831e81a8f794264c6
```

2. 消息"abc"：

```
SM3("abc") = 66c7f0f462eeedd9d1f2d46bdc10e4e24167c4875cf2f7a2297da02b8f4ba8e0
```

3. 长消息：

```
SM3("abcdabcdabcdabcdabcdabcdabcdabcdabcdabcdabcdabcdabcdabcdabcdabcdabcdabcd") =
debe9ff92275b8a138604889c18e5a4d6fdb70e5387e5765293dcba39c0c5732
```

5. 安全性分析

SM3的设计目标是提供128位的安全强度，能够抵抗以下攻击：

1. **碰撞攻击**：找到两个不同的消息m1和m2，使得SM3(m1) = SM3(m2)
2. **原象攻击**：给定哈希值h，找到一个消息m使得SM3(m) = h
3. **第二原象攻击**：给定消息m1，找到另一个不同的消息m2使得SM3(m1) = SM3(m2)

SM3通过以下设计确保安全性：

- 复杂的布尔函数FF和GG提供了良好的混淆特性
- 置换函数P0和P1提供了良好的扩散特性
- 64轮迭代确保了足够的安全性边际

- 精心设计的常量和消息扩展过程增强了算法的抗分析能力

截至目前，尚未有公开的有效攻击能够威胁SM3的安全性，使其成为商用密码应用的可靠选择。