

## 对缩减杂凑函数 HAVAL 的原根攻击

王高丽

(东华大学 计算机科学与技术学院, 上海 201620)

(wanggaoli@dhu.edu.cn)

**摘 要:**根据杂凑函数 HAVAL 算法中消息字的顺序和第一圈中圈函数的性质,结合使用“中间相遇攻击”和穷举搜索等方法,给出从第 3 步到第 122 步 HAVAL 压缩函数的原根攻击。分别采用中间相遇方法和树方法,把对 120 步压缩函数的原根攻击扩展到对 120 步 HAVAL 算法的原根攻击。

**关键词:**杂凑函数;HAVAL 算法;算法安全性分析;原根攻击

**中图分类号:** TP309 **文献标志码:** A

## Preimage attack on step reduced hash function HAVAL

WANG Gao-li

(School of Computer Science and Technology, Donghua University, Shanghai 201620, China)

**Abstract:** According to the order of the message words in HAVAL algorithm and the property of the function in the first pass, a preimage attack of the compression function from step 3 to step 122 was proposed by using "meeting-in-the-middle attack" and the exhaustive search method. The preimage attack on the 120-step reduced compression function was extended to the preimage attack on the 120-step reduced HAVAL by using "meeting-in-the-middle method" and the tree method respectively.

**Key words:** hash function; HAVAL algorithm; security cryptanalysis; preimage attack

### 0 引言

杂凑函数是把任意长度(二进制表示时的长度)的消息压缩为固定长度杂凑值的函数,即杂凑函数  $h: \{0,1\}^* \rightarrow \{0,1\}^n$ ,其中  $\{0,1\}^*$  表示任意长度消息的集合, $\{0,1\}^n$  表示长度为  $n$  的比特串的集合。安全的杂凑函数  $h$  应满足以下三个属性:

**抗原根攻击** 对任意的杂凑值  $y$ ,找到一个消息  $x$ ,使得  $h(x) = y$  是计算上不可行的;

**抗第二原根攻击** 对任意消息  $x$ ,找到另一个异于  $x$  的消息  $x_0$ ,使得  $h(x) = h(x_0)$  是计算上不可行的;

**抗碰撞攻击** 找到两个相异的消息  $x$  和  $x_0$ ,使得  $h(x) = h(x_0)$  是计算上不可行的。

目前对杂凑函数安全性的分析主要集中在对它们进行碰撞攻击,适用于所有杂凑函数的碰撞攻击的方法是“生日攻击”。假设杂凑值的长度是  $n$  bit,用生日攻击找到碰撞的计算复杂度是  $2^{\frac{n}{2}}$  次杂凑运算;而寻找原根和第二原根理论上只能穷举搜索,其计算复杂度都是  $2^n$  次杂凑运算。如果找到碰撞的计算复杂度小于  $2^{\frac{n}{2}}$ ,或者找到原根的计算复杂度小于  $2^n$ ,或者找到第二原根的计算复杂度小于  $2^n$ ,则称该杂凑函数被理论上破解。

近来对杂凑函数安全性的分析得到了一系列重大结果:文献[1-4]对 MD4 系列杂凑函数 HAVAL、MD5、RIPEMD、SHA-0 和 SHA-1 等进行了碰撞攻击。有些应用环境要求杂凑函数抗原根攻击,而不需要抗碰撞攻击,所以对杂凑函数进行原根攻击有着很重要的意义。寻找杂凑函数的原根比寻找

碰撞要困难,文献[5]找到了前两轮 MD4 算法的原根,文献[6]对完整 MD4 算法给出了原根攻击,文献[7]给出了 96 步 HAVAL 和 47 步 MD5 算法的原根攻击。考虑到 HAVAL 算法<sup>[8]</sup>里消息字的顺序,本文给出对 120 步 HAVAL(第 3 步到第 122 步)压缩函数的原根攻击,然后将对压缩函数的原根攻击被扩展为对 120 步 HAVAL 算法的原根攻击。

### 1 HAVAL 算法

本文只对 120 步(第 3 步到第 122 步)的算法进行原根攻击,故只描述 128 步的算法。虽然 HAVAL 算法的杂凑值长可以为 128、160、192、224 和 256 bit,由于长为 128、160、192 和 224 bit 的输出都是从 256 bit 的输出截取得到的,所以本文就认为 HAVAL 的杂凑值长 256 bit。HAVAL 的结构和 MD4 类似,首先进行消息填充,使得填充后的消息长为 1024 的倍数,然后用压缩函数把长为 1024 bit 的消息分组压缩为长为 256 bit 的输出。HAVAL 的初始参数为:

$$(aa_0, bb_0, cc_0, dd_0, ee_0, ff_0, gg_0, hh_0) = \\ (0x243f6a88, 0x85a308d3, 0x13198a2e, 0x03707344, \\ 0xa4093822, 0x299f31d0, 0x082efa98, 0xec4e6c89)$$

128 步算法包括 4 圈,每圈包括 32 步,每圈采用一个非线性圈函数,分别为:

$$F_1(x_6, x_5, x_4, x_3, x_2, x_1, x_0) = x_1x_3 \oplus x_5x_6 \oplus x_2x_4 \oplus \\ x_0x_3 \oplus x_0 \\ F_2(x_6, x_5, x_4, x_3, x_2, x_1, x_0) = x_0x_1x_6 \oplus x_1x_2x_5 \oplus x_1x_6 \oplus \\ x_2x_6 \oplus x_1x_3 \oplus x_0x_5 \oplus x_2x_5 \oplus x_1x_4 \oplus x_4 \\ F_3(x_6, x_5, x_4, x_3, x_2, x_1, x_0) = x_0x_2x_6 \oplus x_2x_3 \oplus x_0x_4 \oplus$$

$$x_1x_6 \oplus x_5x_6 \oplus x_5 \\ F_4(x_6, x_5, x_4, x_3, x_2, x_1, x_0) = x_1x_2x_5 \oplus x_0x_2x_4 \oplus x_0x_5x_6 \oplus \\ x_0x_1 \oplus x_2x_6 \oplus x_0x_5 \oplus x_4x_5 \oplus x_5x_6 \oplus x_0x_4 \oplus x_0x_6 \oplus \\ x_0x_3 \oplus x_3$$

其中  $x_0, \dots, x_6$  是长 32 bit 的字。

HAVAL 的压缩函数: 对长 1 024 bit 的消息分组  $M = (m_0, \dots, m_{31})$ , 运算过程如下所示。

1)  $a_0 = aa, b_0 = bb, c_0 = cc, d_0 = dd, e_0 = ee, f_0 = ff, g_0 = gg, h_0 = hh$ ,  $(aa, bb, cc, dd, ee, ff, gg, hh)$  是分组  $M$  的输入参数, 如果  $M$  是第一个被压缩的分组, 则  $(aa, bb, cc, dd, ee, ff, gg, hh)$  是初始参数; 否则, 就是前一个消息分组的压缩值。

2) 对  $j = 1, 2, 3, 4$ , 完成以下 128 步运算:

$$i = 32(j-1), 32(j-1) + 1, \dots, 32(j-1) + 31$$

$$p_i = F_j(g_i, f_i, e_i, d_i, c_i, b_i, a_i)$$

$$r = (p_i >> 7) + (h_i >> 11) + m_{(j,i)} + k_{j,i}$$

$$h_{i+1} = g_i$$

$$g_{i+1} = f_i$$

$$f_{i+1} = e_i$$

$$e_{i+1} = d_i$$

$$d_{i+1} = c_i$$

$$c_{i+1} = b_i$$

$$b_{i+1} = a_i$$

$$a_{i+1} = r$$

$$3) aa = a_{128} + a, bb = b_{128} + b, \dots, hh = h_{128} + h_0$$

如果  $M$  是最后一个消息分组, 则原消息的压缩值为  $hh \parallel gg \parallel ff \parallel ee \parallel dd \parallel cc \parallel bb \parallel aa$ , 其中  $\parallel$  表示级联。每步运算中用到了一个常数  $k_{j,i}$  (其中  $k_{0,i} = 0, 0 \leq i \leq 31$ ); “ $>>$ ” 表示循环右移; “ $+$ ” 表示模  $2^{32}$  加运算。每圈中消息字的次序见表 1。

表 1 消息字的顺序

圈数	消息字的顺序
(1, $i$ )	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31
(2, $i$ )	5, 14, 26, 18, 11, 28, 7, 16, 0, 23, 20, 22, 1, 10, 4, 8, 30, 3, 21, 9, 17, 24, 29, 6, 19, 12, 15, 13, 2, 25, 31, 27
(3, $i$ )	19, 9, 4, 20, 28, 17, 8, 22, 29, 14, 25, 12, 24, 30, 16, 26, 31, 15, 7, 3, 1, 0, 18, 27, 13, 6, 21, 10, 23, 11, 5, 2
(4, $i$ )	24, 4, 0, 14, 2, 7, 28, 23, 26, 6, 30, 20, 18, 25, 19, 3, 22, 11, 31, 21, 8, 27, 12, 9, 1, 29, 5, 15, 17, 10, 16, 13

## 2 对 120 步压缩函数的原根攻击

首先介绍圈函数  $F_1$  的一些性质, 然后给出对 120 步 HAVAL 压缩函数 (记为 CHAVAL) 的原根攻击, 其计算复杂度是  $2^{224}$  次 CHAVAL 运算。

### 2.1 圈函数的性质

性质 1 设  $y_1 = F_1(x_6, x_5, x_4, x_3, x_2, x_1, x_0)$ ,  $y_{1,i} = F_1(x_6, \dots, x_{i+1}, \neg x_i, x_{i-1}, \dots, x_0)$ , 则:

$$1) y_1 = y_{1,0} \text{ 当且仅当 } x_3 = 1;$$

$$2) y_1 = y_{1,1} \text{ 当且仅当 } x_3 = 0;$$

$$3) y_1 = y_{1,2} \text{ 当且仅当 } x_4 = 0;$$

$$4) y_1 = y_{1,3} \text{ 当且仅当 } x_0 + x_1 = 0;$$

$$5) y_1 = y_{1,4} \text{ 当且仅当 } x_2 = 0;$$

$$6) y_1 = y_{1,5} \text{ 当且仅当 } x_6 = 0;$$

$$7) y_1 = y_{1,6} \text{ 当且仅当 } x_5 = 0.$$

其中  $x_i \in \{0, 1\}$  ( $0 \leq i \leq 6$ )。

### 2.2 对 CHAVAL 的原根攻击

考虑到 HAVAL 算法中消息字的顺序, 本文给出从第 3 步到第 122 步算法的原根攻击, 记算法的输入为  $(a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2)$ 。利用圈函数  $F_1$  的性质, 结合使用“中间相遇攻击”和穷举搜索来寻找原根。

为了得到 CHAVAL 的压缩值  $hh^* \parallel gg^* \parallel ff^* \parallel ee^* \parallel dd^* \parallel cc^* \parallel bb^* \parallel aa^*$ , 寻找一个 1 024 bit 的消息分组  $M = (m_0, \dots, m_{31})$  和输入参数  $(a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2)$ , 使得它们经过 CHAVAL 压缩的值等于  $hh^* \parallel gg^* \parallel ff^* \parallel ee^* \parallel dd^* \parallel cc^* \parallel bb^* \parallel aa^*$ 。算法 1 描述了该攻击过程。

算法 1 对 120 步压缩函数的原根攻击。

重复以下过程:

1) 取  $a_2 = 0, c_2 = 0, d_2 = 0, f_2 = 0$ , 随机取  $b_2, e_2, g_2$  和  $h_2$ 。

2) 取消息字  $m_2, m_3, m_5, m_6, m_7, m_9, m_{10}, m_{11}, m_{13}, m_{14}, m_{15}, m_{17}, m_{18}, m_{19}, m_{21}, m_{22}, m_{23}, m_{25}, m_{26}, m_{27}$  使得  $a_3 = 0, a_4 = 0, a_6 = 0\text{xffffffff}$ ,  $a_7 = 0, a_8 = 0, a_{10} = 0, a_{11} = 0, a_{12} = 0, a_{14} = 0\text{xffffffff}$ ,  $a_{15} = 0, a_{16} = 0, a_{18} = 0, a_{19} = 0, a_{20} = 0, a_{22} = 0\text{xffffffff}$ ,  $a_{23} = 0, a_{24} = 0, a_{26} = 0, a_{27} = 0, a_{28} = 0$  分别成立。

3) 随机选择消息字  $m_0, m_1, m_4, m_8, m_{12}, m_{16}, m_{20}, m_{24}, m_{28}, m_{29}, m_{30}, m_{31}$ 。

4) 对  $2^{64}$  个  $b_2$  和  $e_2$ :

① 计算得到  $(a_{94}, b_{94}, c_{94}, d_{94}, e_{94}, f_{94}, g_{94}, h_{94})$ , 存储到  $L$ 。

② 令  $a_{122} = aa - a_2, c_{122} = cc - c_2, d_{122} = dd - d_2, f_{122} = ff - f_2$ 。

5) 对  $2^{64}$  个  $b_{122}$  和  $e_{122}$ :

① 计算得到  $(a_{95}, b_{95}, c_{95}, d_{95}, e_{95}, f_{95}, g_{95}, h_{95})$ 。

② 与  $L$  中的变量作比较, 若  $L$  中存在中间变量满足  $b_{95} = a_{94}, c_{95} = b_{94}, d_{95} = c_{94}, e_{95} = d_{94}, f_{95} = e_{94}, g_{95} = f_{94}, h_{95} = g_{94}$ , 则:

6) 修正  $m_5$  从而得到:

$$a_{95} = (F_3(g_{94}, f_{94}, e_{94}, d_{94}, c_{94}, b_{94}, a_{94}) >> 7) + (h_{94} >> 11) + m_5 + k_{3,30}$$

7) 通过修改  $b_2$  和  $e_2$  来修改  $a_9, a_{17}, a_{25}$ , 并且保持其他的变量不变。

8) 计算杂凑值  $hh \parallel gg \parallel ff \parallel ee \parallel dd \parallel cc \parallel bb \parallel aa$ , 如果它等于  $hh^* \parallel gg^* \parallel ff^* \parallel ee^* \parallel dd^* \parallel cc^* \parallel bb^* \parallel aa^*$  则返回消息  $M = (m_0, \dots, m_{31})$  和输入参数  $(a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2)$ 。下面证明算法 1 可以对 CHAVAL 进行原根攻击。

1) 为了通过修改  $m_2$  来使得  $a_3 = 0$  成立, 修改  $m_2$  如下:

$$m_2 = 0 - (F_1(g_2, \dots, a_2) >> 7) - (h_2 >> 11) - k_{1,2}$$

同理可修正其他条件。

2) 为了让算法 1 的第 6) 步成立, 修改  $m_5$  如下:

$$m_5 = a_{95} - (F_3(g_{94}, \dots, a_{94}) >> 7) - (h_{94} >> 11) - k_{3,30}$$

3) 在 HAVAL 算法的第 1 圈和第 2 圈,  $m_5$  分别用在第 6 步和第 33 步, 下面证明  $m_5$  的改动不会影响最终的压缩值。

① 由算法可知:

$$a_{33} = (F_2(g_{32}, f_{32}, e_{32}, d_{32}, c_{32}, b_{32}, a_{32}) > 7) + (h_{32} > 11) + m_5 + k_{2,0}$$

且  $h_{32} = a_{25}$ , 为了“中和”在第33步  $m_5$  的变化, 修改  $a_{25}$  为:

$$a_{25} \leftarrow (a_{33} - (F_2(g_{32}, \dots, a_{32}) > 7) - m_5 - k_{2,0}) < 11$$

根据  $F_1$  的性质可知, 条件  $a_{22} = 0$ ,  $a_{23} = 0$ ,  $a_{24} = 0$ ,  $a_{26} = 0$ ,  $a_{27} = 0$ ,  $a_{28} = 0$  保证  $a_{25}$  的改变不会影响  $a_{26}$ ,  $\dots$ ,  $a_{32}$ , 从而不会影响其他的变量。

② 类似 ① 的情况, 通过修改  $a_{17}$  来修改  $a_{25}$ , 条件  $a_{14} = 0$ ,  $a_{15} = 0$ ,  $a_{16} = 0$ ,  $a_{18} = 0$ ,  $a_{19} = 0$ ,  $a_{20} = 0$  保证  $a_{17}$  的变化不会影响  $a_{18}$ ,  $\dots$ ,  $a_{24}$ 。通过修改  $a_9$  来修改  $a_{17}$ , 条件  $a_6 = 0$ ,  $a_7 = 0$ ,  $a_8 = 0$ ,  $a_{10} = 0$ ,  $a_{11} = 0$ ,  $a_{12} = 0$ , 保证  $a_9$  的变化不会影响  $a_{10}$ ,  $\dots$ ,  $a_{16}$ 。

③ 通过修改  $b_2$  来修改  $a_9$ 。为了“中和”在第6步  $m_5$  的变化, 修改  $e_2$  为:

$$e_2 \leftarrow (a_6 - (F_1(g_5, \dots, a_5) > 7) - m_5 - k_{1,5}) < 11$$

根据  $F_1$  的性质可知, 条件  $a_2 = 0$ ,  $c_2 = 0$ ,  $d_2 = 0$ ,  $f_2 = 0$ ,  $a_3 = 0$ ,  $a_4 = 0$  保证  $b_2$  和  $e_2$  的改变不会影响  $a_4$ ,  $\dots$ ,  $a_8$ , 从而不会影响其他的变量。

4) 因此, 可以通过修改  $b_2$  和  $e_2$  来“中和” $m_5$  的变化, 同时不影响变量  $a_{26}$ ,  $\dots$ ,  $a_{34}$ , 从而不影响压缩值。

算法1复杂度的估计: 一方面, 由第4)、5)步知, 共有  $2^{64} \times 2^{64} = 2^{128}$  个组合, 而在第10步“中间相遇”的概率是  $(2^{-32})^7 = 2^{-224}$ , 故需重复  $\frac{1}{2^{-224} \times 2^{128}} = 2^{96}$  次才能得到一次“中间相遇”, 由于做一次的计算量是  $2^{64}$  次杂凑运算, 故得到192 bit 压缩值的计算量是  $2^{64} \times 2^{96} = 2^{160}$  次杂凑运算; 穷举搜索其余的  $64 (= 256 - 192)$  bit 压缩值, 从而算法1的计算量是  $2^{160} \times 2^{64} = 2^{224}$  次杂凑运算。( $a_{94}, b_{94}, c_{94}, d_{94}, e_{94}, f_{94}, g_{94}, h_{94}$ ) 共256 bit, 即32 Byte, 故由第4)步知, 需要存储  $2^{64} \times 32 = 2^{69}$  Byte。

### 3 对120步 HAVAL 的原根攻击

把对压缩函数的原根攻击扩展为对 HAVAL 算法的原根攻击主要考虑两个方面: 一是 HAVAL 的填充算法, 另一个是 HAVAL 算法的初始值。填充算法只对最后几个消息字有限制条件, 特殊情况下, 对前29个消息字都有限制条件, 只对后3个消息字有限制条件, 在本文的攻击里, 对后3个消息字没有要求, 所以填充算法不影响把对压缩函数的原根攻击扩展为对 HAVAL 算法的原根攻击。但是, 本文攻击里所用的初始值与 HAVAL 算法的标准初始值不同, 下面给出两种方法来克服这个问题。

#### 3.1 基本的中间相遇方法

设杂凑值的长度是  $n$  bit, 对压缩函数原根攻击的计算复杂度是  $2^x$  次杂凑运算, 一方面, 在标准初始值下压缩  $2^{\frac{n-x}{2}}$  个随机的消息, 另一方面, 计算出压缩值  $hh \parallel gg \parallel ff \parallel ee \parallel dd \parallel cc \parallel bb \parallel aa$  的  $2^{\frac{n-x}{2}}$  个原根, 根据生日攻击知, 中间相遇的期望值的个数是  $\frac{2^{\frac{n-x}{2}} \times 2^{\frac{n-x}{2}}}{2^n} = 1$ , 找到原根的计算复杂度是  $2^{\frac{n-x}{2}} + 2^{\frac{n-x}{2}} \times 2^x = 2^{1+\frac{n-x}{2}}$  次杂凑运算。

本文  $x = 224$ , 在标准初始值下压缩  $2^{240}$  个随机消息, 需  $2^{240}$  次运算; 计算出  $hh \parallel gg \parallel ff \parallel ee \parallel dd \parallel cc \parallel bb \parallel aa$  的  $2^{16}$  个

原根, 需  $2^{240} = 2^{16} \times 2^{224}$  次运算。因此, 对120步 HAVAL 的原根攻击的计算复杂度是  $2^{240} + 2^{240} = 2^{241}$  次杂凑运算, 由算法1的复杂度估计知, 需存储  $2^{69}$  Byte。

#### 3.2 树方法

该方法是中间相遇方法和基于树结构方法的组合, 其细节可参考文献[7]。首先基于树方法计算出  $hh \parallel gg \parallel ff \parallel ee \parallel dd \parallel cc \parallel bb \parallel aa$  的  $2^{32}$  个多消息分组原根, 根据参考文献[7], 结合算法1的计算复杂度是  $2^{224}$  可知, 计算出  $2^{32}$  个多消息分组原根的计算复杂度是  $2^{230} = 32 \times 2^{225}$  次运算。然后在标准初始值下压缩  $2^{224}$  个随机消息, 根据生日攻击知, 中间相遇的期望值是  $\frac{2^{32} \times 2^{224}}{2^{256}} = 1$ , 因此, 对120步 HAVAL 的原根攻击的计算复杂度约是  $2^{230}$  次杂凑运算, 需存储  $2^{69}$  Byte。

### 4 结语

过去对杂凑函数 HAVAL 安全性的分析主要集中在碰撞攻击, 本文给出了对120步 HAVAL 算法的原根攻击, 其计算复杂度约是  $2^{230}$  次杂凑运算, 需存储  $2^{69}$  Byte。在实际应用中, 有些环境要求杂凑函数抗原根攻击, 本文的分析结果表明在这些环境中使用 HAVAL 算法存在较大的安全隐患。

#### 参考文献:

- [1] ELI B, CHEN R, ANTOINE J, *et al.* Collisions of SHA-0 and reduced SHA-1 [C]// 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, LNCS 3494: Advances in Cryptology-EUROCRYPT'05. Berlin: Springer-Verlag, 2005: 36-57.
- [2] WANG XIAO-YUN, LAI XUE-JIA, FENG DENG-GUO, *et al.* Cryptanalysis for hash functions MD4 and RIPEMD [C]// 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, LNCS 3494: Advances in Cryptology-EUROCRYPT'05. Berlin: Springer-Verlag, 2005: 1-18.
- [3] WANG XIAO-YUN, YU HONG-BO. How to break MD5 and other hash functions [C]// 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, LNCS 3494: Advances in Cryptology-EUROCRYPT'05. Berlin: Springer-Verlag, 2005: 19-35.
- [4] 王小云, 冯登国, 于秀源. HAVAL-128 的碰撞攻击[J]. 中国科学: E 辑, 2005, 35(3): 1-12.
- [5] DOBBERTIN H. The first two rounds of MD 4 are not one-way [C]// Proceedings of the 5th International Workshop on Fast Software Encryption, LNCS 1372: Advances in Cryptology-FSE'98. Berlin: Springer-Verlag, 1998: 284-292.
- [6] LEURENT G. MD4 is not one-way: Lausanne [C]// Proceedings of the 15th International Workshop on Fast Software Encryption, LNCS 5086: Advances in Cryptology-FSE'08. Berlin: Springer-Verlag, 2008: 412-428.
- [7] AUMASSON J-P, MEIER W, MENDEL F. Preimage attacks on 3-pass HAVAL and step-reduced MD5 [C]// The 15th Annual Workshop on Selected Areas in Cryptography: SAC'08. Berlin: Springer-Verlag, 2008.
- [8] ZHENG Y, PIEPRZYK J, SEBERRY J. HAVAL - A one-way hashing algorithm with variable length of output [C]// Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques, LNCS 718: Advances in Cryptology-AUSCRYPT'92. Berlin: Springer-Verlag, 1992: 83-104.