

针对 51 步 RIPEMD-160 区分攻击的复杂度分析

崔斌斌, 王高丽

(华东师范大学 计算机科学与软件工程学院, 上海 200062)

摘 要: SASAKI Y 等人使用消息修改技术对 51 步 RIPEMD-160 算法进行区分攻击时 (Lecture Notes in Computer Science, Vol. 7341), 未考虑模减差分对差分路线的影响, 不能保证左右操作前半部分成立的概率为 1, 导致所得到的复杂度有误。为此, 通过 2-dimension sums 方法构建 RIPEMD-160 区分器, 在此基础上提出一种改进的区分攻击复杂度分析方法。给出保证差分路线中模减差分成立的充分条件, 使消息修改后左右操作差分路线前半部分成立的概率由 $2^{-7.717}$ 增加到 1, 从而提高区分器成立概率, 同时通过实验测试得到后半部分差分路线成立的概率。分析结果表明, 51 步 RIPEMD-160 算法区分攻击的复杂度为 $2^{152.672}$ 。

关键词: RIPEMD-160 算法; boomerang 区分器; 模减差分; 充分条件; 消息修改

开放科学(资源服务)标志码(OSID):



中文引用格式: 崔斌斌, 王高丽. 针对 51 步 RIPEMD-160 区分攻击的复杂度分析[J]. 计算机工程, 2019, 45(9): 147-152.

英文引用格式: CUI Binbin, WANG Gaoli. Complexity analysis of distinguishing attack on 51-step RIPEMD-160[J]. Computer Engineering, 2019, 45(9): 147-152.

Complexity Analysis of Distinguishing Attack on 51-step RIPEMD-160

CUI Binbin, WANG Gaoli

(School of Computer Science and Software Engineering, East China Normal University, Shanghai 200062, China)

[Abstract] SASAKI Y, et al used message modification technology to differentiate attack on 51-step RIPEMD-160 (Lecture Notes in Computer Science, Vol. 7341), but they did not consider the influence of modular subtraction difference on differential routes, and could not guarantee that the probability of the validation of the first half of the left and right operations is 1, which led to the error in the complexity of RIPEMD-160. For this reason, RIPEMD-160 differentiator is constructed by 2-dimension sums method and an improved distinguishing attack complexity analysis method is proposed. Sufficient conditions are given to ensure the validation of modular subtraction difference in differential routes. The probability of the validation of the first half of differential routes of left and right operations is increased from $2^{-7.717}$ to 1 after message modification. The probability of the validation of differential routes of the second half is obtained through experimental tests. The analysis results show that the complexity of distinguishing attacks on 51-step RIPEMD-160 is $2^{152.672}$.

[Key words] RIPEMD-160 algorithm; boomerang distinguisher; modular subtraction difference; sufficient condition; message modification

DOI: 10.19678/j.issn.1000-3428.0052363

0 概述

哈希函数 RIPEMD-160^[1] 基于 Markle-Damgard^[2-3] 结构, 是 RIPEMD 家族中的重要算法, 同时也是 ISO/IEC 国际标准之一。RIPEMD 家族的压缩函数采用双分支操作, 由于 RIPEMD 算法 2 个分支操作很相似, 因此降低了攻击难度。其中, RIPEMD-160 的 2 个分支操作使用了不同的常数、移位值、消息字

顺序和布尔函数, 增强了安全性, 其在文件校验、数字签名中被广泛应用。

目前对于 RIPEMD-160 算法的碰撞攻击、原像攻击和区分攻击研究有: 文献[4]提出一种对 36 步 RIPEMD-160 算法(从第 2 轮开始)的半自由起始碰撞攻击方案; 文献[5]提出一种自动搜索 RIPEMD-160 差分路线的改进方案, 建立了 48 步(从第 2 轮开始)和 36 步算法(从第 1 轮开始)的差分路线, 并在此基

基金项目: 国家密码发展基金(MMJJ20180201)。

作者简介: 崔斌斌(1993—), 男, 硕士研究生, 主研方向为信息安全、哈希函数; 王高丽(通信作者), 副教授。

收稿日期: 2018-08-10 修回日期: 2018-10-05 E-mail: glwang@sei.ecnu.edu.cn

基础上分别给出 42 步和 36 步 RIPEMD-160 算法的半自由起始碰撞攻击方案;文献[6]给出保证 RIPEMD-160 算法差分路线的模减差分成立的充分条件,通过消息修改技术使模减差分成立的概率为 1,同时给出了针对前 30 步 RIPEMD-160 算法的碰撞攻击,并改进前 36 步 RIPEMD-160 算法的半自由起始碰撞攻击结果;文献[7]给出一种 48 步 RIPEMD-160 算法的半自由起始碰撞攻击方案;文献[8]给出一种 31 步 RIPEMD-160 的原像攻击方案;文献[9]给出了 RIPEMD-160 的 31 步~34 步的原像攻击方案(从第 1 轮开始)和 35 步的原像攻击(从第 2 轮开始);文献[10]给出一种 51 步 RIPEMD-160 算法的区分攻击方案。

文献[11]提出对分组密码算法的 boomerang 攻击方法。在此基础上,文献[12-13]将这个方法应用到哈希函数的区分攻击中,文献[10]利用基于 boomerang 区分器的 2-dimension sums 方法,给出 51 步 RIPEMD-160 算法的区分攻击方法,复杂度为 2^{158} ,通过构建 RIPEMD-160 压缩函数左右操作的差分路线,并使用消息修改技术使左右操作差分路线的前半部分(左操作的 17 步~25 步,右操作的 17 步~22 步)以 1 的概率成立。然而由于 RIPEMD-160 的步操作不是一个标准的 T 函数(第 i 位输出只依赖于前 i 位的输入),并且文献[10]在使用消息修改技术时没有考虑模减差分对差分路线的影响,因此不能保证在消息修改之后左右操作前半部分成立的概率为 1,由此可知文献[10]中得到的区分攻击复杂度是有误的。

针对上述问题,本文给出保证 RIPEMD-160 算法差分路线模减差分成立的充分条件,通过消息修改技术使差分路线前半部分成立的概率为 1,以降低区分攻击的复杂度,同时在评估差分路线后半部分的概率时考虑差分路线的模减差分,通过实验测试得到后半部分差分路线成立的概率。

1 预备知识

1.1 基本符号说明

本文用到的符号说明如下:

1) $M = (m_0, m_1, \dots, m_{15})$ 和 $M' = (m'_0, m'_1, \dots, m'_{15})$ 分别代表 512 bit 的消息字。

2) X_i 和 X'_i 分别代表左分支压缩函数处理消息字 M 和 M' 在第 i ($1 \leq i \leq 80$) 步产生的输出值,其中 X_i 和 X'_i 都为 32 bit。

3) Y_i 和 Y'_i 分别代表右分支压缩函数处理消息字 M 和 M' 在第 i ($1 \leq i \leq 80$) 步产生的输出值,其中 Y_i 和 Y'_i 都为 32 bit。

4) $+$ 和 $-$ 分别表示模 2^{32} 加运算和模 2^{32} 减运算。

5) $x \lll s$ 表示 x 向左循环移动 s 位, $x \ggg s$ 表示 x 向右循环移动 s 位。

6) $\Delta x_i = x'_i - x_i$ 表示 x'_i 和 x_i 之间的模减差分。

7) $\Delta x_i = [j]$ 表示 $x_{i,j} = 0, x'_{i,j} = 1, x_{i,k} = x'_{i,k}, 0 \leq k \leq 31, k \neq j$ 。

8) $\Delta x_i = [-j]$ 表示 $x_{i,j} = 1, x'_{i,j} = 0, x_{i,k} = x'_{i,k}, 0 \leq k \leq 31, k \neq j$ 。

9) $C(i \sim j)$ 表示 C 的第 i 位到第 j 位的值 $0 \leq j \leq i \leq 31$ 。

1.2 RIPEMD-160 算法

哈希函数 RIPEMD-160 将任意长度的消息(小于 2^{64})压缩为 160 bit 的哈希值。首先将填充之后的消息分块,每个消息块的长度为 512 bit,然后压缩函数将每一个 512 bit 的消息块压缩成 160 bit 的输出。RIPEMD-160 的压缩函数包含 2 个分支,分别记为左分支和右分支,每个分支包含 5 轮运算,每轮包含 16 步操作,具体描述如下:

将 160 bit 的输入链变量 cv 记为 5 个字 cv_i ($i = 0, 1, 2, 3, 4$),其中 cv_i 的长度是 32 bit。首先对左右分支的 160 bit 的内部状态进行初始化:

$$X_{-4} = Y_{-4} = cv_0 \ggg 10, X_{-3} = Y_{-3} = cv_4 \ggg 10$$

$$X_{-2} = Y_{-2} = cv_3 \ggg 10, X_{-1} = Y_{-1} = cv_2, X_0 = Y_0 = cv_1$$

在第 j 轮 ($1 \leq j \leq 5$),左右操作的链接变量 X_i 和 Y_i ($1 \leq i \leq 80$) 的更新操作分别为:

$$X_i = (X_{i-4} \lll 10) + ((X_{i-5} \lll 10) + F_j(X_{i-1}, X_{i-2}, (X_{i-3} \lll 10)) + m_{\pi^l(i)} + k_j^l) \lll s_i^l$$

$$Y_i = (Y_{i-4} \lll 10) + ((Y_{i-5} \lll 10) + G_j(Y_{i-1}, Y_{i-2}, (Y_{i-3} \lll 10)) + m_{\pi^r(i)} + k_j^r) \lll s_i^r$$

其中,布尔函数 F_j 和 G_j , 轮常数 k_j^l 和 k_j^r 都依赖于轮数 j ($1 \leq j \leq 5$) 和左右分支,布尔函数 F_j 和 G_j 参照表 1,轮常数 k_j^l 和 k_j^r 的取值参照表 2,消息字 $m_{\pi^l(i)}$ 和 $m_{\pi^r(i)}$ 的顺序以及对应的移位值 s_i^l 和 s_i^r 参照表 3。

表 1 双分支的布尔函数

| X | $F_i(X, Y, Z)$ | $G_i(X, Y, Z)$ |
|---------|---------------------------------------|---------------------------------------|
| 1 ~ 16 | $X \oplus Y \oplus Z$ | $(Y \vee \neg Z) \oplus X$ |
| 17 ~ 32 | $(X \wedge Y) \vee (\neg X \wedge Y)$ | $(X \wedge Z) \vee (\neg Z \wedge Y)$ |
| 33 ~ 48 | $(X \vee \neg Y) \oplus Z$ | $(X \vee \neg Y) \oplus Z$ |
| 49 ~ 64 | $(X \wedge Z) \vee (\neg Z \wedge Y)$ | $(X \wedge Y) \vee (\neg X \wedge Y)$ |
| 65 ~ 80 | $(Y \vee \neg Z) \oplus X$ | $X \oplus Y \oplus Z$ |

表 2 轮常数取值

| round(j) | k_j^l | k_j^r |
|--------------|------------|------------|
| 1 | 0x00000000 | 0x50a28be6 |
| 2 | 0x5a827999 | 0x5c4dd124 |
| 3 | 0x6ed9eba1 | 0x6d703ef3 |
| 4 | 0x8f1bbcdc | 0x7a6d76e9 |
| 5 | 0xa953fd4e | 0x00000000 |

表 3 消息字的顺序和对应的移位值

| i | $\pi^l(i)$ | $\pi^r(i)$ | s_i^l | s_i^r | i | $\pi^l(i)$ | $\pi^r(i)$ | s_i^l | s_i^r |
|-----|------------|------------|---------|---------|-----|------------|------------|---------|---------|
| 1 | 0 | 5 | 11 | 8 | 41 | 2 | 11 | 14 | 12 |
| 2 | 1 | 14 | 14 | 9 | 42 | 7 | 8 | 8 | 13 |
| 3 | 2 | 7 | 15 | 9 | 43 | 0 | 12 | 13 | 5 |
| 4 | 3 | 0 | 12 | 11 | 44 | 6 | 2 | 6 | 14 |
| 5 | 4 | 9 | 5 | 13 | 45 | 13 | 10 | 5 | 13 |
| 6 | 5 | 2 | 8 | 15 | 46 | 11 | 0 | 12 | 13 |
| 7 | 6 | 11 | 7 | 15 | 47 | 5 | 4 | 7 | 7 |
| 8 | 7 | 4 | 9 | 5 | 48 | 12 | 13 | 5 | 5 |
| 9 | 8 | 13 | 11 | 7 | 49 | 1 | 8 | 11 | 15 |
| 10 | 9 | 6 | 13 | 7 | 50 | 9 | 6 | 12 | 5 |
| 11 | 10 | 15 | 14 | 8 | 51 | 11 | 4 | 14 | 8 |
| 12 | 11 | 8 | 15 | 11 | 52 | 10 | 1 | 15 | 11 |
| 13 | 12 | 1 | 6 | 14 | 53 | 0 | 3 | 14 | 14 |
| 14 | 13 | 10 | 7 | 14 | 54 | 8 | 11 | 15 | 14 |
| 15 | 14 | 3 | 9 | 12 | 55 | 12 | 15 | 9 | 6 |
| 16 | 15 | 12 | 8 | 6 | 56 | 4 | 0 | 8 | 14 |
| 17 | 7 | 6 | 7 | 9 | 57 | 13 | 5 | 9 | 6 |
| 18 | 4 | 11 | 6 | 13 | 58 | 3 | 12 | 14 | 9 |
| 19 | 13 | 3 | 8 | 15 | 59 | 7 | 2 | 5 | 12 |
| 20 | 1 | 7 | 13 | 7 | 60 | 15 | 13 | 6 | 9 |
| 21 | 10 | 0 | 11 | 12 | 61 | 14 | 9 | 8 | 12 |
| 22 | 6 | 13 | 9 | 8 | 62 | 5 | 7 | 6 | 5 |
| 23 | 15 | 5 | 7 | 9 | 63 | 6 | 10 | 5 | 15 |
| 24 | 3 | 10 | 15 | 11 | 64 | 2 | 14 | 12 | 8 |
| 25 | 12 | 14 | 7 | 7 | 65 | 4 | 12 | 9 | 8 |
| 26 | 0 | 15 | 12 | 7 | 66 | 0 | 15 | 15 | 5 |
| 27 | 9 | 8 | 15 | 12 | 67 | 5 | 10 | 5 | 12 |
| 28 | 5 | 12 | 9 | 7 | 68 | 9 | 4 | 11 | 9 |
| 29 | 2 | 4 | 11 | 6 | 69 | 7 | 1 | 6 | 12 |
| 30 | 14 | 9 | 7 | 15 | 70 | 12 | 5 | 8 | 5 |
| 31 | 11 | 1 | 13 | 13 | 71 | 2 | 8 | 13 | 14 |
| 32 | 8 | 2 | 12 | 11 | 72 | 10 | 7 | 12 | 6 |
| 33 | 3 | 15 | 11 | 9 | 73 | 14 | 6 | 5 | 8 |
| 34 | 10 | 5 | 13 | 7 | 74 | 1 | 2 | 12 | 13 |
| 35 | 14 | 1 | 6 | 15 | 75 | 3 | 13 | 13 | 6 |
| 36 | 4 | 3 | 7 | 11 | 76 | 8 | 14 | 14 | 5 |
| 37 | 9 | 7 | 14 | 8 | 77 | 11 | 0 | 11 | 15 |
| 38 | 15 | 14 | 9 | 6 | 78 | 6 | 3 | 8 | 13 |
| 39 | 8 | 6 | 13 | 6 | 79 | 15 | 9 | 5 | 11 |
| 40 | 1 | 9 | 15 | 14 | 80 | 13 | 11 | 6 | 11 |

1.3 模减差分及其概率计算

对于已知差分路线的 RIPEMD-128 算法,容易推导出保证差分路线成立的充分条件,并利用消息修改技术使得差分路线成立。由于 RIPEMD-160 的步操作不是标准的 T 函数,在计算差分路线的概率时不能忽略模减差分的影响,因此在计算差分路线中任意一步成立的概率时,既要考虑比特条件成立的概率,也要考虑模减差分成立的概率。文献[6]推导得到保证 RIPEMD-160 算法差分路线的模减差分成立的充分条件,通过消息修改技术^[14]使模减差分成立的概率为 1,同时给出前 30 步 RIPEMD-160 算法的碰撞攻击,并优化前 36 步 RIPEMD-160 算法的半自由起始碰撞攻击结果。下文介绍模减差分修改中用到的表达式以及修改方法。

令 $\Delta = X'_i - X_i, \Delta_{i-5} = (X'_{i-5} \lll 10) - (X_{i-5} \lll 10)$, $\Delta_{i-4} = (X'_{i-4} \lll 10) - (X_{i-4} \lll 10), \Delta F = \varphi_j^r(X'_{i-1}, X'_{i-2}, X'_{i-3} \lll 10) - \varphi_j^r(X_{i-1}, X_{i-2}, X_{i-3} \lll 10)$, 则根据计算 X_i 步操作的表达式可知模减差分成立的概率为 $\Pr(v) = \Pr[\Delta = \Delta_{i-4} + (\Delta_{i-5} + \Delta F + \Delta m_{\pi(i)} + T) \lll s_i - (T \lll s_i)]$, 其中, $T = (X_{i-5} \lll 10) + F_i(X_{i-1}, X_{i-2}, (X_{i-3} \lll 10)) + m_{\pi(i)} + k_j^l$ 。

令 $C_0 = \Delta_{i-5} + \Delta F + \Delta m_{\pi(i)}, C_1 = \Delta - \Delta_{i-4}$, 则模减差分成立的概率转换为: $\Pr(v) = \Pr[(T + C_0) \lll s_i = (T \lll s_i) + C_1]$ 。因为对于给定的差分路线, C_0 和 C_1 的值是确定的,所以可以通过在 T 上添加条件来保证 $\Pr(v) = 1$ 。又根据 X_i 的表达式知, $T = (X_{i-1} - (X_{i-4} \lll 10)) \ggg s_i$, 因此,在 T 上的条件可以转化为在 X_i 和 X_{i-4} 上的条件,即通过在 X_i 和 X_{i-4} 上添加条件,可以使得 X_i 和 X'_i 模减差分成立的概率为 1。具体添加条件的过程参见文献[6]。

2 文献[10]对 RIPEMD-160 的区分攻击分析

本节介绍文献[10]中 3 种构建 RIPEMD-160 区分器的方法,重点研究本文讨论的 2-dimension sums 方法,并分析其原理和实现方式。文献[10]提出的 3 种区分器,分别为 4-sum、2-dimension sums 和 n -dimension sums。

4-sum 属性是 4 个不同的输入 (I_0, I_1, I_2, I_3) 对应输出的异或和为 0,即 $CF(I_0) \oplus CF(I_1) \oplus CF(I_2) \oplus CF(I_3) = 0$, 其中, CF 为 RIPEMD-160 的压缩函数。构建 4-sum 区分器的一般复杂度为 $2^{n/3}$, 其中 n 为哈希值的长度。

通过对 4-sum 属性的输入 ∇ 添加约束,扩展出 2-dimension sums 属性,目的是寻找 4 个不同的输入 (I_0, I_1, I_2, I_3) 满足 $I_1 = I_0 \oplus \Delta, I_2 = I_0 \oplus \nabla, I_3 = I_0 \oplus \Delta \oplus \nabla$, 并且输出满足 $CF(I_0) \oplus CF(I_1) \oplus CF(I_2) \oplus CF(I_3) = 0$ (Δ 和 ∇ 分别代表 RIPEMD-160 左右分支的输入差分)。构建 2-dimension sums 区分器的一般复杂度是 2^n , 此方法适合用于构造双分支结构哈希函数的区分器。

n -dimension sums 区分器是 2-dimension sums 区分器的扩展,适用于构造 n 分支结构哈希函数的区分器。下文详细介绍 2-dimension sums 区分器。

如图 1 所示,文献[10]首先构建 RIPEMD-160 算法左右分支操作的差分路线,然后寻找输入链接变量 H 和消息 M ,使得 (M, H) 和 $(M + \Delta_M, H + \Delta_H)$ 之间以及 $(M + \nabla_M, H + \nabla_H)$ 和 $(M + \Delta_M + \nabla_M, H + \Delta_H + \nabla_H)$ 之间遵循左分支操作的差分路线,并且 (M, H) 和 $(M + \nabla_M, H + \nabla_H)$ 之间以及 $(M + \Delta_M, H + \Delta_H)$ 和 $(M + \Delta_M + \nabla_M, H + \Delta_H + \nabla_H)$ 之间遵循右分支操作的差分路线。对于这样的 (M, H) ,可以保证以下的关系式成立:

$$F(M + \Delta_M + \nabla_M, H + \Delta_H + \nabla_H) = CF(M, H) + CCF(M + \Delta_M, H + \Delta_H) + CF(M + \nabla_M, H + \nabla_H)$$

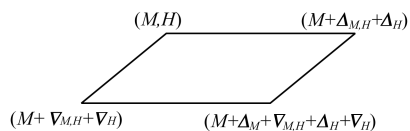


图 1 2-dimension sums 模型

3 51 步 RIPEMD-160 区分攻击分析

本节给出保证差分路线中模减差分成立的充分条件,并使用消息修改技术使左右操作差分路线的前半部分成立的概率为 1,改进了文献[10]中区分攻击的复杂度。本节修改了文献[10]中差分路线的错误,并通过实验测试给出差分路线后半部分的概率,最终给出 51 步 RIPEMD-160 区分攻击的正确复杂度。

3.1 差分路线中模减差分成立的充分条件

下文将通过一个实例说明如何在 T 上添加条件,使得在消息修改^[14-16]后,该步的模减差分成立的概率为 1。本节使用的差分路线如表 4 和表 5 所示,其中 * 表示取正号或负号均可。可以看出,从第 2 轮开始,差分路线与文献[10]略有不同,左分支第 56 步、59 步、60 步和 63 步为控制差分扩散进行了修改,第 67 步则修改了差分值。

表 4 差分路线(左分支)

| 步数 | s_i^r | $m_{\pi^r(i)}$ | $\Delta_{m_{\pi^r(i)}}$ | Δ_{y_i} |
|----|---------|----------------|-------------------------|----------------|
| 12 | — | — | — | [22, 24, 25] |
| 13 | — | — | — | [0, -31] |
| 14 | — | — | — | [5] |
| 15 | — | — | — | 0 |
| 16 | — | — | — | [-28] |
| 17 | 7 | 7 | — | [7] |
| 18 | 6 | 4 | — | [-16] |
| 19 | 8 | 13 | — | [23] |
| 20 | 13 | 1 | — | [-6] |
| 21 | 11 | 10 | — | 0 |
| 22 | 9 | 6 | — | 0 |
| 23 | 7 | 15 | — | 0 |
| 24 | 15 | 3 | — | 0 |
| 25 | 7 | 12 | 2^{16} | 0 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 47 | 7 | 5 | — | 0 |
| 48 | 5 | 12 | 2^{16} | [21] |
| 49 | 11 | 1 | — | 0 |
| 50 | 12 | 9 | — | 0 |
| 51 | 14 | 11 | — | 0 |
| 52 | 15 | 10 | — | [31] |
| 53 | 14 | 0 | — | 0 |
| 54 | 15 | 8 | — | 0 |
| 55 | 9 | 12 | 2^{16} | [25] |
| 56 | 8 | 4 | — | [-9, 10] |
| 57 | 9 | 13 | — | 0 |
| 58 | 14 | 3 | — | 0 |
| 59 | 5 | 7 | — | [-3, 4] |
| 60 | 6 | 15 | — | [-19, 20] |
| 61 | 8 | 14 | — | 0 |
| 62 | 6 | 5 | — | 0 |
| 63 | 5 | 6 | — | [-13, 14] |
| 64 | 12 | 2 | — | [29] |
| 65 | 9 | 4 | — | 0 |
| 66 | 15 | 0 | — | 0 |
| 67 | 5 | 5 | — | [23] |

表 5 差分路线(右分支)

| 步数 | s_i^r | $m_{\pi^r(i)}$ | $\Delta_{m_{\pi^r(i)}}$ | Δ_{y_i} |
|----|---------|----------------|-------------------------|----------------------|
| 12 | — | — | — | [11, -16] |
| 13 | — | — | — | [-20] |
| 14 | — | — | — | [1] |
| 15 | — | — | — | [-16] |
| 16 | — | — | — | [23] |
| 17 | 9 | 6 | — | [-3] |
| 18 | 13 | 11 | — | 0 |
| 19 | 15 | 3 | — | 0 |
| 20 | 7 | 7 | — | 0 |
| 21 | 12 | 0 | — | 0 |
| 22 | 8 | 13 | 2^{13} | 0 |
| 23 | 9 | 5 | — | 0 |
| 24 | 11 | 10 | — | 0 |
| 25 | 7 | 14 | — | 0 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| 47 | 7 | 4 | — | 0 |
| 48 | 5 | 13 | 2^{13} | [18] |
| 49 | 15 | 8 | — | 0 |
| 50 | 5 | 6 | — | 0 |
| 51 | 8 | 4 | — | 0 |
| 52 | 11 | 1 | — | [28] |
| 53 | 14 | 3 | — | 0 |
| 54 | 14 | 11 | — | 0 |
| 55 | 6 | 15 | — | 0 |
| 56 | 14 | 0 | — | [6] |
| 57 | 6 | 5 | — | 0 |
| 58 | 9 | 12 | — | 0 |
| 59 | 12 | 2 | — | 0 |
| 60 | 9 | 13 | 2^{13} | [16, 22] |
| 61 | 12 | 9 | — | 0 |
| 62 | 5 | 7 | — | 0 |
| 63 | 15 | 10 | — | 0 |
| 64 | 8 | 14 | — | [0, 26] |
| 65 | 8 | 12 | — | 0 |
| 66 | 5 | 15 | — | [*5, *31] |
| 67 | 12 | 10 | — | [*11, *16, *17, *22] |

本例考虑右分支的第 17 步操作 (Y_{17}), 根据 2.3 节结论可知, Y_{17} 和 Y'_{17} 的模减差分以 1 的概率成立, 当且仅当 $(T + C_0) \lll s_i = (T \lll s_i) + C_1$ 成立。根据 C_0 和 C_1 的计算公式可得 $C_0 = 0\text{xfc}200000$, $C_1 = 0\text{x}3\text{ffff}8$, 同时从表 2 可知 $s_{17}^r = 9$ 。 C_0 和 C_1 的二进制表示如表 6 和表 7 所示, 其中 t_i 表示 T 的第 i ($0 \leq i \leq 31$) 位的值。

表 6 C_0 的二进制表示

| t_i | C_0 | t_i | C_0 | t_i | C_0 |
|----------|-------|----------|-------|----------|-------|
| t_0 | 0 | t_{11} | 0 | t_{22} | 0 |
| t_1 | 0 | t_{12} | 0 | t_{23} | 0 |
| t_2 | 0 | t_{13} | 0 | t_{24} | 0 |
| t_3 | 0 | t_{14} | 0 | t_{25} | 0 |
| t_4 | 0 | t_{15} | 0 | t_{26} | 1 |
| t_5 | 0 | t_{16} | 0 | t_{27} | 1 |
| t_6 | 0 | t_{17} | 0 | t_{28} | 1 |
| t_7 | 0 | t_{18} | 0 | t_{29} | 1 |
| t_8 | 0 | t_{19} | 0 | t_{30} | 1 |
| t_9 | 0 | t_{20} | 0 | t_{31} | 1 |
| t_{10} | 0 | t_{21} | 1 | | |

表7 C_1 的二进制表示

| t_i | C_1 | t_i | C_1 | t_i | C_1 |
|----------|-------|----------|-------|----------|-------|
| t_0 | 0 | t_{11} | 0 | t_{22} | 1 |
| t_1 | 0 | t_{12} | 0 | t_{23} | 1 |
| t_2 | 0 | t_{13} | 0 | t_{24} | 1 |
| t_3 | 1 | t_{14} | 0 | t_{25} | 1 |
| t_4 | 1 | t_{15} | 0 | t_{26} | 1 |
| t_5 | 1 | t_{16} | 0 | t_{27} | 1 |
| t_6 | 1 | t_{17} | 0 | t_{28} | 1 |
| t_7 | 1 | t_{18} | 0 | t_{29} | 1 |
| t_8 | 1 | t_{19} | 0 | t_{30} | 0 |
| t_9 | 1 | t_{20} | 0 | t_{31} | 0 |
| t_{10} | 1 | t_{21} | 1 | | |

由表6和表7可知, $C_0(31 \sim 23) = C_1(8 \sim 0)$, $C_0(22 \sim 0) = C_1(31 \sim 9) + 1$ 。

为使 $(T + C_0) \lll 9 = (T \lll 9) + C_1$ 成立,需要在 T 上添加条件,过程如下:

首先,考虑 $(T \lll 9) + C_1$ 的第0位,为使 $t_{23} + C_{0,23} +$ 进位 $= t_{23} + C_{1,0}$ 成立,即为了让 $t_{23} + 0 +$ 进位 $= t_{23} + 0$ 成立,需要在计算 $(T + C_0)$ 时,第22位向第23位不能有进位。又根据 $C_{0,22} = 0$ 可知,添加条件 $t_{22} = 0$ 后,就可保证在计算 $(T + C_0)$ 时第22位向第23位没有进位。

其次,考虑 $(T \lll 9) + C_1$ 的第9位,为使 $t_0 + C_{0,0} = t_0 + C_{1,9} +$ 进位成立,即为使 $t_0 + 0 = t_0 + 1 +$ 进位成立,需要在计算 $(T \lll 9) + C_1$ 时,第8位向第9位有进位。又根据 $C_{1,8} = 1$ 知,添加条件 $t_{31} = 1$,就可保证在计算 $(T \lll 9) + C_1$ 时第8位向第9位有进位。

由RIPEMD-160算法可知: $T = (Y_{17} - (Y_{13} \lll 10)) \ggg 9$,因此,添加在 t_{22} 和 t_{31} 上的条件可转化为添加在 Y_{17} 和 Y_{13} 上。此时,需要结合 Y_{17} 和 Y_{13} 现有的比特条件(推导差分路线时添加的),在剩下没有条件的位置上加条件,如果出现了与已有条件矛盾的情况,还可以对公式 $T = (Y_{12} \lll 10) + F_i(Y_{16}, Y_{15}, (Y_{14} \lll 10)) + m_{\pi'(i)} + k_j'$ 中的 $Y_{12}, Y_{16}, Y_{15}, Y_{14}$ 中的任意一个链接变量加条件,使得 T 上的条件满足。所以,本例添加的条件是: $Y_{17,31} = 0, Y_{17,30} = 0, Y_{17,8} = 1, Y_{17,7} = 1, Y_{13,30} = 0, Y_{13,29} = 0$ 。再使用消息修改技术,让这些条件满足,从而可以保证右分支第17步的模减差分以1的概率成立。程序测试结果表明添加条件后的该步模减差分成立的概率始终为1。

3.2 与文献[10]分析结果的对比

本文通过对左右差分路线前面部分的每一步都采用3.1节方法添加条件,且在消息修改后保证每一步的模减差分成立的概率都为1。文献[10]中的消息修改没有保证差分路线的模减差分以1的概率成立,表8和表9分别比较了本文方法和文献[10]方法左右分支的差分路线前半部分模减差分成立的概率以及相应需要添加的条件,这部分成立的概率会增加区分攻击的复杂度。由于文献[10]在评估复

杂度时对此没有考虑,因此其给出的复杂度比实际正确的复杂度要低。

表8 模减差分成立概率比较(左分支)

| 步数 | 文献[10] 方法 | 本文 方法 | 添加条件 |
|----|--------------|----------|--|
| 17 | 0.999 999 | 1 | $X_{17,31} = 0, X_{17,30} = 0, X_{13,21} = Y_{13,21} = 1$ |
| 18 | 0.999 978 | 1 | $X_{18,31} = 0, X_{18,30} = 1, X_{14,21} = Y_{14,21} = 1$ |
| 19 | 0.997 980 | 1 | $X_{19,31} = X_{15,21} = Y_{15,21}, X_{19,30} = 1,$ $X_{15,20} = Y_{15,20} = 0$ |
| 20 | 0.998 627 | 1 | — |
| 21 | 0.998 326 | 1 | $X_{21,31} = 1, X_{21,30} = 1, X_{17,21} = 0, X_{17,20} = 0$ |
| 22 | 0.982 527 | 1 | $X_{22,31} = 0, X_{22,30} = 1, X_{18,21} = 0, X_{18,20} = 0$ |
| 23 | 0.971 012 | 1 | $X_{23,31} = 1, X_{23,30} = 1, X_{19,21} = 0, X_{19,20} = 0$ |
| 24 | 0.975 031 | 1 | $X_{24,31} = 0, X_{24,30} = 1, X_{20,21} = 0, X_{20,20} = 0$ |
| 25 | 0.971 959 | 1 | — |

表9 模减差分成立概率比较(右分支)

| 步数 | 文献[10] 方法 | 本文 方法 | 添加条件 |
|----|--------------|----------|--|
| 17 | 0.738 388 | 1 | $Y_{17,31} = 0, Y_{17,30} = 0, Y_{17,8} = 1$ $Y_{17,7} = 1, Y_{13,30} = X_{13,30} = 0, Y_{13,29} = X_{13,29} = 0$ |
| 18 | 0.469 738 | 1 | $Y_{18,12} = 1, Y_{18,11} = 1, Y_{14,2} = X_{14,2} = 0$ |
| 19 | 0.439 874 | 1 | $Y_{19,31} = 0, Y_{19,30} = 1, Y_{15,21} = X_{15,21} = 0$ |
| 20 | 0.336 268 | 1 | $Y_{20,6} = 0, Y_{20,5} = 1, Y_{16,27} = X_{16,27} = 0$ |
| 21 | 0.275 782 | 1 | $Y_{21,31} = 0, Y_{21,30} = 1, Y_{17,21} = 0, Y_{17,20} = 0$ |
| 22 | 0.274 668 | 1 | — |

3.3 51步RIPEMD-160区分攻击的复杂度

文献[10]中使用2-dimension sums的方法计算出51步的RIPEMD-160区分攻击的复杂度为 2^{158} ,但忽略了差分路线前面部分的模减差分的影响,认为消息修改之后的差分路线前面部分成立的概率为1,然而表8和表9的实验结果表明,左右分支的差分路线前半部分成立的概率为 $2^{-7.717}$ 。另一方面,文献[10]通过统计比特条件的数量,计算出差分路线后面部分的复杂度为 2^{158} ,按照其计算方法,51步RIPEMD-160区分攻击的复杂度应为 $2^{158} \times 2^{15.434} = 2^{173.434}$,该结果超出 2^{160} ,因此,该文给出的51步RIPEMD-160的区分攻击不成立。

本文考虑差分路线前面部分的模减差分,添加了保证模减差分成立的充分条件,并通过消息修改技术让这些条件成立,从而保证差分路线前面部分成立的概率为1,所以,在计算区分攻击的复杂度时,前半部分的复杂度可以忽略。同时本文也考虑了差分路线后面部分的模减差分,通过程序测试出差分路线后面部分成立的概率。实验结果表明,51步RIPEMD-160区分攻击的复杂度为 $2^{152.672}$; 52步RIPEMD-160区分攻击的复杂度为 $2^{161.995}$,该结果大于 2^{160} 。因此,对51步RIPEMD-160算法的2-dimension sums最优的区分攻击复杂度为 $2^{152.672}$ 。

4 结束语

本文考虑 RIPEMD-160 算法差分路线的模减差分,通过在链接变量上添加条件并使用消息修改技术,使差分路线中模减差分成立的概率为 1,从而降低 51 步 RIPEMD-160 区分攻击的复杂度。分析结果表明,对 51 步 RIPEMD-160 区分攻击的复杂度为 $2^{152.672}$,该结果为正确计算 RIPEMD-160 算法其他区分攻击的复杂度提供了参考。下一步将对左右分支消息字的选择进行比较,通过推理和编程找到最合适的消息字,再借助于模减差分技术得到更好的攻击方法。

参考文献

- [1] DOBBERTIN H, BOSSELAERS A, PRENEEL B. RIPEMD-160: a strengthened version of RIPEMD[C]//Proceedings of International Workshop on Fast Software Encryption. Cambridge, UK: [s. n.], 1996: 71-82.
- [2] DAMGÅRD I B. A design principle for Hash functions[C]//Proceedings of Conference on the Theory and Application of Cryptology. Santa Barbara, USA: [s. n.], 1989: 416-427.
- [3] MERKLE R C. One way Hash functions and DES[C]//Proceedings of Conference on the Theory and Application of Cryptology. Santa Barbara, USA: [s. n.], 1989: 428-446.
- [4] MENDEL F, NAD T, SCHERZ S, et al. Differential attacks on reduced RIPEMD-160[C]//Proceedings of International Conference on Information Security. Seoul, Korea: [s. n.], 2012: 23-38.
- [5] MENDEL F, PEYRIN T, SCHLÄFFER M, et al. Improved cryptanalysis of reduced RIPEMD-160[C]//Proceedings of International Conference on the Theory and Application of Cryptology and Information Security. Bengaluru, India: [s. n.], 2013: 484-503.
- [6] LIU Fukang, MENDEL F, WANG Gaoli. Collisions and semi-free-start collisions for round-reduced RIPEMD-160[C]//Proceedings of International Conference on the Theory and Application of Cryptology and Information Security. Paris, France: [s. n.], 2017: 158-186.
- [7] WANG Gaoli, SHEN Yanzhao, LIU Fukang. Cryptanalysis of 48-step RIPEMD-160 [J]. IACR Transactions on Symmetric Cryptology, 2017(2): 177-202.
- [8] OHTAHARA C, SASAKI Y, SHIMOYAMA T. Preimage attacks on step-reduced RIPEMD-128 and RIPEMD-160[C]//Proceedings of International Conference on Information Security and Cryptology. Singapore: [s. n.], 2010: 169-186.
- [9] 申延召. 约减轮 Hash 函数 HAS-160、RIPEMD-160 和 SM3 的原像攻击[D]. 济南: 山东大学, 2018.
- [10] SASAKI Y, WANG L. Distinguishers beyond three rounds of the RIPEMD-128/-160 compression functions[J]. Lecture Notes in Computer Science, 2012, 7341: 275-292.
- [11] WAGNER D. The boomerang attack [C]//Proceedings of International Workshop on Fast Software Encryption. Rome, Italy: [s. n.], 1999: 156-170.
- [12] BIRYUKOV A, NIKOLIĆ I, ROY A. Boomerang attacks on BLAKE-32 [C]//Proceedings of International Workshop on Fast Software Encryption. Lyngby, Denmark: [s. n.], 2011: 218-237.
- [13] 吴广辉, 于红波, 郝泳霖. 对约减轮数 Skein-1024 的 Boomerang 区分攻击[J]. 密码学报, 2016, 3(5): 492-504.
- [14] WANG Xiaoyun, LAI Xuejia, FENG Dengguo, et al. Cryptanalysis of the Hash functions MD4 and RIPEMD[C]//Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques. Aarhus, Denmark: [s. n.], 2005: 1-18.
- [15] WANG Xiaoyun, YU Hongbo. How to break MD5 and other Hash functions [C]//Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques. Aarhus, Denmark: [s. n.], 2005: 19-35.
- [16] WANG Xiaoyun, YIN Y L, YU Hongbo. Finding collisions in the full SHA-1 [C]//Proceedings of Annual International Cryptology Conference. Santa Barbara, USA: [s. n.], 2005: 17-36.

编辑 金胡考