改进的MD4有意义碰撞攻击

周永鹏[1] 王高丽[2]
(东华大学计算机科学与技术学院 上海 201620)[1]
(华东师范大学计算机科学与软件工程学院 上海 200062)[2]

摘 要 在FSE 1996上,Hans Dobbertin给出了一个基于ASCII编码且前20个字符是随机字符的有意义的MD4碰撞。贾珂婷和王小云教授于2009年给出了一个基于Latin-1字符集的有意义的MD4碰撞。以王小云教授的模差分方法为基础,采用于红波等在CANS 2005上给出的碰撞路线,给出了两个有意义的MD4碰撞实例,其中一个是基于GBK编码的汉语的有意义碰撞,另一个是基于UTF-8编码的英语的有意义的碰撞。同时给出了一个python脚本被篡改的实例。

关键词 MD4算法,模差分方分析,有意义的碰撞,GBK编码,UTF-8编码
中图法分类号 TN918.4 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.09.032

Improved Meaningful Collision Attack on MD4

ZHOU Yong-peng[1] WANG Gao-li[2]
(Department of Computer Science and Technology, Donghua University, Shanghai 201620, China)[1]
(School of Computer Science and Software Engineering, East China Normal University, Shanghai 200062, China)[2]

Abstract In FSE'1996, Hans Dobbertin gave a meaningful collision on MD4 based on ASCII, which contains meaningless words at the beginning of the text. In 2009, Jia and Wang presented a meaningful collision on MD4 based on Latin-1 character set, which contains meaningless words at the end of the text. In this paper, based on the modular differential method proposed by Wang, we gave two concrete meaningful collisions by using the differential characteristic proposed by Yu et al. in CANS 2005. One example of the meaningful collision is in Chinese and based on GBK, an other example is in English and based on UTF-8. Moreover, an example of tampered python script was proposed.

Keywords MD4 algorithm, Modular differential cryptanalysis, Meaningful collision, GBK, UTF-8

1 引言

杂凑算法在现代密码学中具有重要地位,它可以将不定长度的消息压缩成固定长度的消息摘要,广泛用于电子签名、文件校验等。本文研究的杂凑算法MD4[1]是美国密码学家Rivest于1990年针对32位操作系统设计的高速压缩算法,其虽然已经被破解,但是在一些对压缩效率要求比较高的场景依然被使用,例如ed2k使用MD4算法[1]计算出的Hash ID为文件的唯一标识,通过ed2k下载文件时需使用Hash ID对文件的正确性和完整性进行校验[14]。

杂凑算法的研究在各类国际密码组织的推动下得到了长足的发展,最早对MD4算法进行分析的有den Boer, Bosselaers[2]和Vaudenay[3]。Dobbertin[4]于1996年给出了MD4算法的破解方法,并且利用这个方法找到了MD4算法的有意义的碰撞。王小云等人于2004年首次公开了杂凑算法MD4,MD5,HAVAL-128,RIPEMD[11]的碰撞实例[5],引起了密码学界对杂凑算法研究的高度关注。王小云等人提出的模差分方法破解了包括MD5[6]和SHA-1[7]在内的一系列国际通用杂凑算法,促进了杂凑算法分析和设计技术的进一步发展。于红波等人于2005年提出了对于MD4算法的第二原像攻击方法[8],给出了条件数更少的MD4算法的碰撞路线,提高了第二原像攻击的效率。贾珂婷和王小云教授于2009年通过利用高效的碰撞路线[8]给出了MD4算法基于Latin-1编码格式的有意义碰撞实例[9]。

贾珂婷等人在构造MD4算法有意义的碰撞[9]时用到了特殊的编码方式,即文献[9]构造的是基于Latin-1字符集的有意义的碰撞。文献[9]为构造有意义的碰撞语句,首先根据第一轮条件逆推满足条件的消息的固定位,然后再对剩余位进行构造以保证第一轮的条件不被修改。这种方法要求设计者充分理解碰撞路线,利用差分路线中每一步的充分条件构造有意义的碰撞,但没有充分理解碰撞路线的研究人员很难使用这种方法。这是一种基于白盒模式设计的破解方法。

本文介绍了一种MD4算法有意义碰撞的方法,该方法适用多种字符编码,在构造有意义的碰撞时不需要深刻理解碰撞路线,只需要保证程序中的目标语句固定位前的自由位个数大于固定位个数,利用MD4轮函数的传导效应保证固定位不被修改,这是一种基于黑盒模式设计的破解方法。本文给出了MD4算法有意义的纯文本碰撞。

到稿日期:2016-08-17 返修日期:2016-11-30 本文受国家自然科学基金(61572125,61373142),上海市"科技创新行动计划"高新技术领域项目(16511101400)资助。

周永鹏(1989—),男,硕士,主要研究方向为密码学、信息安全,E-mail:zhouyongpeng1234@163.com;王高丽(1982—),女,博士,副教授,主要研究方向为密码学、信息安全。

2　预备知识

2.1　基本符号说明

(1)消息差分定义如下：

$M=(m_0,m_1,\cdots,m_{15})$，$\Delta M=(0,0,0,0,2^i,0,\cdots,0)$，$M'=M+\Delta M$

$M=(m_0,m_1,\cdots,m_{15})$和$M'=(m_0',m_1',\cdots,m_{15}')$分别表示 512 比特的消息分组。$M$和$M'$的差分记为$\Delta M=M'-M=(\Delta m_0,\Delta m_1,\cdots,\Delta m_{15})$，其中每个$\Delta m_i=m_i'-m_i$，$i=1,2,\cdots,15$。

(2)消息碰撞的定义

如果M满足表 1 中的所有充分条件，则M和M'由相同的 hash 值构成碰撞。

表 1　本文使用的 MD4 差分路径[8]

Step	Output for M	m_i	s_i	Δm_i	Output difference	Output for M'	Sufficient conditions
1	a_1	m_0	3				
2	d_1	m_1	7				
3	c_1	m_2	11				
4	b_1	m_3	19				
5	a_2	m_4	3	2^0	$2^{0\boxplus 3}$	$a_2[0\boxplus 4]$	$a_{2,0\boxplus 4}=0$
6	d_2	m_5	7			d_2	$b_{1,0\boxplus 4}=c_{1,0\boxplus 1}$
7	c_2	m_6	11			c_2	$d_{2,0\boxplus 4}=0$
8	b_2	m_7	19			b_2	$c_{2,0\boxplus 4}=1$
9	a_3	m_8	3		$2^{0\boxplus 6}$	$a_3[-(0\boxplus 7),(0\boxplus 8)]$	$a_{3,0\boxplus 7}=1,a_{3,0\boxplus 8}=0$
10	d_3	m_9	7			d_3	$b_{2,0\boxplus 7}=c_{2,0\boxplus 7},b_{2,0\boxplus 8}=c_{2,0\boxplus 8}$
11	c_3	m_{10}	11		$-2^{0\boxplus 17}$	$c_3[-(0\boxplus 18)]$	$c_{3,0\boxplus 18}=1,d_{3,0\boxplus 7}=1,d_{3,0\boxplus 8}=0$
12	b_3	m_{11}	19			b_3	$c_{3,0\boxplus 7}=1,c_{3,0\boxplus 8}=1,d_{3,0\boxplus 8}=a_{3,0\boxplus 18}$
13	a_4	m_{12}	3		$2^{0\boxplus 9}$	$a_4[0\boxplus 10]$	$a_{4,0\boxplus 10}=0,b_{3,0\boxplus 18}=0$
14	d_4	m_{13}	7				$b_{3,0\boxplus 10}=c_{3,0\boxplus 10},a_{4,0\boxplus 18}=0$
15	c_4	m_{14}	11		$-2^{0\boxplus 28}$	$c_4[-(0\boxplus 29)]$	$c_{4,0\boxplus 29}=1,d_{4,0\boxplus 10}=0$
16	b_4	m_{15}	19				$c_{4,0\boxplus 10}=1,d_{4,0\boxplus 29}=a_{4,0\boxplus 29}$
17	a_5	m_0	3		$2^{0\boxplus 12}$	$a_5[0\boxplus 13]$	$a_{5,0\boxplus 13}=0,b_{4,0\boxplus 29}=d_{4,0\boxplus 29}$
18	d_5	m_4	5		$2^{0\boxplus 5}+2^{0\boxplus 17}$	$d_5[0\boxplus 6,0\boxplus 18]$	$d_{5,0\boxplus 6}=0,d_{5,0\boxplus 18}=0,a_{5,0\boxplus 29}=b_{4,0\boxplus 29},b_{4,0\boxplus 13}=c_{4,0\boxplus 13}+1$
19	c_5	m_8	9		$-2^{0\boxplus 5}$	$c_5[-(0\boxplus 6)]$	$c_{5,0\boxplus 6}=1,a_{5,0\boxplus 18}=b_{4,0\boxplus 18},d_{5,0\boxplus 13}=b_{4,0\boxplus 13},a_{5,0\boxplus 6}=b_{4,0\boxplus 6}$
20	b_5	m_{12}	13			b_5	$c_{5,0\boxplus 13}=d_{5,0\boxplus 13},c_{5,0\boxplus 18}=a_{5,0\boxplus 18}$
21	a_6	m_1	3		$2^{0\boxplus 10}$	$a_6[(0\boxplus 16)]$	$a_{6,0\boxplus 16}=0,b_{5,0\boxplus 18}=c_{5,0\boxplus 18}$
22	d_6	m_5	5		$2^{0\boxplus 22}$	$d_6[(0\boxplus 23)]$	$d_{6,0\boxplus 23}=0,b_{5,0\boxplus 16}=c_{5,0\boxplus 16},a_{6,0\boxplus 6}=b_{5,0\boxplus 6}+1$
23	c_6	m_9	9		$-2^{0\boxplus 14}$	$c_6[0\boxplus 15),-(0\boxplus 16)]$	$c_{6,0\boxplus 15}=0,c_{6,0\boxplus 16}=1,d_{6,0\boxplus 16}=b_{5,0\boxplus 16},a_{6,0\boxplus 23}=b_{5,0\boxplus 23}$
24	b_6	m_{13}	13			b_6	$d_{6,0\boxplus 15}=a_{6,0\boxplus 15},c_{6,0\boxplus 23}=a_{6,0\boxplus 23}$
25	a_7	m_2	3			a_7	$b_{6,0\boxplus 15}=d_{6,0\boxplus 15},b_{6,0\boxplus 23}=c_{6,0\boxplus 23},b_{6,0\boxplus 16}=d_{6,0\boxplus 16}+1$
26	d_7	m_6	5		$2^{0\boxplus 27}$	$d_7[0\boxplus 28]$	$a_{7,0\boxplus 15}=b_{6,0\boxplus 15},d_{7,0\boxplus 28}=0,a_{7,0\boxplus 16}=b_{6,0\boxplus 16}$
27	c_7	m_{10}	9		$-2^{0\boxplus 23}$	$c_7[-(0\boxplus 24)]$	$c_{7,0\boxplus 24}=1,a_{7,0\boxplus 28}=b_{6,0\boxplus 28}$
28	b_7	m_{14}	13			b_7	$d_{7,0\boxplus 24}=a_{7,0\boxplus 24},c_{7,0\boxplus 28}=a_{7,0\boxplus 28}$
29	a_8	m_5	3			a_8	$b_{7,0\boxplus 24}=d_{7,0\boxplus 24},b_{7,0\boxplus 28}=c_{7,0\boxplus 28}$
30	d_8	m_7	5		2^0	$d_8[0\boxplus 1]$	$d_{8,0\boxplus 1}=0,a_{8,0\boxplus 24}=b_{7,0\boxplus 24}$
31	c_8	m_{11}	9		-2^0	$c_8[-(0\boxplus 1)]$	$c_{8,0\boxplus 1}=1,a_{8,0\boxplus 1}=b_{7,0\boxplus 1}$
32	b_8	m_{15}	13			b_8	
33	a_9	m_0	3			a_9	
34	d_9	m_8	9			d_9	$a_{9,0\boxplus 1}=b_{8,0\boxplus 1}$
35	c_9	m_4	11			c_9	
36	b_9	m_{12}	15			b_9	

(3)$a_{i,j}$，$b_{i,j}$，$c_{i,j}$，$d_{i,j}$和$m_{i,j}$分别表示a_i，b_i，c_i，d_i，m_i的第j比特，规定从第 1 比特开始。

(4)$\wedge$，$\vee$，$\oplus$，$\lll$，$\ggg$，$\boxplus$和$+$运算分别表示逻辑与运算、逻辑或运算、逻辑与或运算、循环左移运算、循环右移运算、模 32 加法运算和模2^{32}加法运算。

2.2　MD4 算法

MD4 算法能够把任意长度的消息压缩成 128 比特的消息摘要，压缩过程包含消息分组和迭代运算两个过程。后来在 MD4 的基础上又发展出被广泛应用的 MD5，HAVAL，SHA-1，SHA-2 系列和 RIPEMD 系列，以及我国商用的 SM3 杂凑标准[10]。在 MD4 算法的计算过程中首先计算消息的比特数，把消息填充为 512 比特的整数倍，然后按 512 比特一组进行分组。MD4 算法中使用的布尔函数分别是：

$F(X,Y,Z)=(X\wedge Y)\vee(\neg X\wedge Z)$

$G(X,Y,Z)=(X\wedge Y)\vee(Y\wedge Z)\vee(X\wedge Z)$

$H(X,Y,Z)=X\oplus Y\oplus Z$

初始值为：

$(a_0,b_0,c_0,d_0)=$(0x67452301，0xefcdab89，0x98badcfe，0x10325476)

定义 3 个函数：

$FF(a,b,c,d,k,s)=(a+F(b,c,d)+X[k])\lll s$

$GG(a,b,c,d,k,s)=(a+G(b,c,d)+X[k]+$0x5a827999$)\lll s$

$HH(a,b,c,d,k,s)=(a+H(b,c,d)+X[k]+$0x6ed9ebal$)\lll s$

其中，s表示移位数，a,b,c,d表示中间值，$X[k]$表示消息字。

第一轮的 16 步计算过程如下：

For $i=0,1,2,3$

$a_{i+1}=FF(a_i,b_i,c_i,d_i,4i,3)$

$d_{i+1}=FF(d_i,a_i,b_i,c_i,4i+1,7)$

$c_{i+1}=FF(c_i,d_i,a_i,b_i,4i+1,11)$

$b_{i+1}=FF(b_i,c_i,d_i,a_i,4i+3,19)$

第二轮的 16 步计算过程如下：

For $i=0,1,2,3$

$a_{i+5}=GG(a_{i+4},b_{i+4},c_{i+4},d_{i+4},i,3)$

$d_{i+5}=GG(d_{i+4},a_{i+4},b_{i+4},c_{i+4},i+4,5)$

$c_{i+5}=GG(c_{i+4},d_{i+4},a_{i+4},b_{i+4},i+8,9)$

$b_{i+5}=GG(b_{i+4},c_{i+4},d_{i+4},a_{i+4},i+12,13)$

第三轮的 16 步计算过程如下：

For $i=0,1,2,3$

$a_{i+9}=HH(a_{i+8},b_{i+8},c_{i+8},d_{i+8},i,3)$

$d_{i+9}=HH(d_{i+8},a_{i+8},b_{i+8},c_{i+8},i+8,9)$

$c_{i+9}=HH(c_{i+8},d_{i+8},a_{i+8},b_{i+8},i+4,11)$

$b_{i+9}=HH(b_{i+8},c_{i+8},d_{i+8},a_{i+8},i+12,15)$

最后的返回值计算过程如下：

$AA=a_0+a_{12}$，$BB=b_0+b_{12}$，$CC=c_0+c_{12}$，$DD=d_0+d_{12}$，则(AA，BB，CC，DD)即为该消息分组的压缩值。

3 寻找有意义的 MD4 碰撞

3.1 选择合适的差分路线

对于差分路线的选择，根据构造碰撞的目的不同，选择的标准也不相同。在构造 MD4 随机碰撞时选择第二轮和第三轮条件较少的差分路线，即使第一轮条件较多，通过简单消息修改技术也不会影响寻找随即碰撞的时间复杂度。但这种方法不适用于寻找有语义的碰撞，因为有语义的语句受限于字符的编码格式，其必须符合特定的编码格式才能被显示，同时还要保证消息具有一定的语义。第一轮条件越多，在修改过程中每一位要满足的条件也就越多，对消息的要求也就越苛刻，从而增加了构造有意义碰撞的难度。因此本文使用文献[8]的差分路线(见表 1)，该差分路线的特点是第一轮的条件较少，有利于保证消息的语义不被破坏。以下给出了使用该差分路线分别构造基于 GBK 和 UTF-8 编码的有意义的碰撞。根据字符 UTF-8 编码的二进制表示可知差分在 $\pm 2^7 \pm 2^{15} \pm 2^{23} \pm 2^{31}$，$M$ 和 M' 中有一个是不能显示的字符。为了方便构造，本文选择 $\Delta m_4=1$(从 0 开始计数)，同时使用 GBK 编码作为构造的元素的编码格式，其所受的限制比 UTF-8 编码受到的限制更少。

3.2 构造合适的消息

有意义的碰撞不仅要满足表 1 中的充分条件，还要满足特定的语义要求，本文介绍了一种高效且通用的方法来构造有意义的碰撞。此方法可以屏蔽差分路线的特征，类似于把所有充分条件设定好并放在一个盒子中，研究人员首先确定差分位，因为差分的选择会影响充分条件[13]；然后确定语句中自由位的个数和固定位的个数，为保证固定位不被修改，必须使得每个固定位前有足够的自由位。例如：“张小三借给刘小二 21186216 元。担保人：孙利。公证人：赵浴”。首先确定差分位于第十七个字符数字‘2’，然后对其分析划分，“张小三”为名字，因此是自由位，“刘小二”同样是自由位，而“借给”为特定动作且不能被替换，是固定位，因此可以使用枚举的方法通过遍历名字来保证“借给”不被修改。“21186216”可以为任意数字，总共有 $9*10^8$ 种组合为自由位，足够保证后面的固定位“元。担保人：”、“。公证人：”和“。”不被修改。

本文使用的方法提高了使用已知的差分路线伪造可执行程序的效率。由于在可执行程序中变量名和中间变量名都能被用来作为自由度，因此可以通过枚举变量名来保证特定的功能段不被修改。目前随着 MD5 和 SHA-1 的差分分析的不断深入，越来越多的高效差分路径被提出，选择一个高效的修改和构造方法会使得理论上的破解给实际应用带来更大的危害和风险。众所周知，在开源的 Linux 和 Unix 系统中采用源码验证编译的过程进行本地安装，如果被伪造源码经过验证并安装在操作系统中，则会带来严重的安全问题。

为了构造基于 GBK 编码的有意义的碰撞，首先确定消息的语意，然后统计消息的结构中自由位和固定位的个数及相对位置。如表 2 所列，把固定位标记出来，枚举自由位使得消息满足碰撞条件，同时也能保证消息的固定位不被修改。

表 2 固定位设计

****	****	****	****	+***	****	****	****
	借	给		1111	1111	元。	担保
****	****	****	****	****	****	****	****
人：		。公	证人	：	。		

修改的具体方法：首先，通过遍历数字部分寻找所有符合的情况，算法如下。

算法 1 枚举构造算法

输入：不同的数字组合 m_n

输出：所有可能的结果

算法描述：

INPUT 语句(m_n)

经过第一轮条件判断修改

IF OUTPUT 语句满足固定位未改变

输出 修改后的语句

ELSE

$m_n=m_n+1$

END

遍历数字自由位如表 3 所列。

表 3 遍历数字自由位

张小三借给刘小二 21186216 元。担保人：孙利。公证人：赵浴。sss3s 驲 q
张小三借给刘小二 22106511 元。担保人：孙利。公证人：赵浴。s3?硪 3q
张小三借给刘小二 25100415 元。担保人：孙利。公证人：赵浴。{s 髂 3 髁 i
张小三借给刘小二 26186417 元。担保人：孙利。公证人：赵浴。s3 驲碡 3q
张小三借给刘小二 27033266 元。担保人：孙利。公证人：赵浴。{s?sq3y
张小三借给刘小二 27035266 元。担保人：孙利。公证人：赵浴。{ss3 硪 sy
张小三借给刘小二 27045266 元。担保人：孙利。公证人：赵浴。{ss3 绸 ry

算法 1 可以灵活使用，在对数字进行枚举无效的情况下，可以对其他自由变量进行枚举。

然后，使用高级明文修改技术对第二轮条件进行修改以提高碰撞发生的概率，保证高级明文修改时补充的条件不与第一轮包含的充分条件发生冲突，能够使用高级明文修改的条件包括：

$a_{5,6}=b_{4,6}$，$a_{5,13}=0$，$a_{5,18}=b_{4,18}$，$a_{5,29}=b_{4,29}$，$d_{5,6}=0$，$d_{5,13}=b_{4,13}$，$c_{5,18}=a_{5,18}$

以 $a_{5,6}=b_{4,6}$ 为例，首先通过轮函数逆运算对 c_4 和 b_4 进行修改：

$c_4=(c_4\wedge(c_4\&0x320042b5)\wedge(d_4\&0x00000081)\wedge((\sim d_4)\&0x22004034))|0x10000200$；

$b_4=(b_4\wedge(b_4\&0x10001100)\wedge(d_4\&0x10000100)\wedge((\sim c_4)\&0x00001000))$；

以上修改能保证 c_4 和 b_4 满足所有条件(包括附加条件和原充分条件)。

然后利用表 4，通过修改 $b_{4,3}$ 对 $a_{5,6}$ 进行修改，对于其他条件的修改见表 5－表 10。

表 4 $a_{5,6}$ 的条件修改

15	m_{14}	11	c_4,d_4,a_4,b_3
16	m_{15}	19	$b_4'=b_4\oplus 4,c_4,d_4,a_4 m_{15}\leftarrow b_4'>>>19-b_3-F(c_4,d_4,a_4)$
17	m_0	3	a_5',b_4',c_4,d_4 $c_{4,3}\neq d_{4,3}$

表 5　$a_{5,13}$的条件修改

15	m_{14}	11	c_4,d_4,a_4,b_3
16	m_{15}	19	$b_4'=b_4\oplus 2^9,c_4,d_4,a_4 \quad m_{15}\leftarrow b_4'>>>19-b_3-F(c_4,d_4,a_4)$
17	m_0	3	$a_5',b_4',c_4,d_4\ c_{4,10}\neq d_{4,10}$

表 6　$a_{5,18}$的条件修改

15	m_{14}	11	c_4,d_4,a_4,b_3
16	m_{15}	19	$b_4'=b_4\oplus 2^{14},c_4,d_4,a_4 \quad m_{15}\leftarrow b_4'>>>19-b_3-F(c_4,d_4,a_4)$
17	m_0	3	$a_5',b_4',c_4,d_4 \quad c_{4,15}\neq d_{4,15}$

表 7　$a_{5,29}$的条件修改

15	m_{14}	11	c_4,d_4,a_4,b_3
16	m_{15}	19	$b_4'=b_4\oplus 2^{25},c_4,d_4,a_4 \quad m_{15}\leftarrow b_4'>>>19-b_3-F(c_4,d_4,a_4)$
17	m_0	3	$a_5',b_4',c_4,d_4c_{4,26}\neq d_{4,26}$

表 8　$d_{5,6}$的条件修改

15	m_{14}	11	c_4,d_4,a_4,b_3
16	m_{15}	19	$b_4'=b_4\oplus 1,c_4,d_4,a_4m_{15}\leftarrow b_4'>>>19-b_3-F(c_4,d_4,a_4)$
17	m_0	3	$a_5,b_4',c_4,d_4 \quad c_{4,1}=d_{4,1}$
18	m_4	5	$d_5',a_5,b_4',c_4 \quad a_{5,1}\neq c_{4,1}$

表 9　$d_{5,13}$的条件修改

15	m_{14}	11	c_4,d_4,a_4,b_3
16	m_{15}	19	$b_4'=b_4\oplus 2^7,c_4,d_4,a_4 \quad m_{15}\leftarrow b_4'>>>19-b_3-F(c_4,d_4,a_4)$
17	m_0	3	$a_5,b_4',c_4,d_4 \quad c_{4,8}=d_{4,8}$
18	m_4	5	$d_5',a_5,b_4',c_4 \quad a_{5,8}\neq c_{4,8}$

表 10　$c_{5,18}$的条件修改

15	m_{14}	11	$c_4'=c_4\oplus 2^8,d_4,a_4,b_3 \quad m_{14}\leftarrow c_4'>>>11-c_3-F(d_4,a_4,b_3)$
16	m_{15}	19	$b_4,c_4',d_4,a_4 \quad m_{15}\leftarrow b_4>>>19-b_3-F(c_4',d_4,a_4)$
17	m_0	3	$a_5,b_4,c_4',d_4 \quad b_{4,4}=d_{4,4}$
18	m_4	5	$c_{4,8}=d_{4,8} \quad a_{5,4}\neq b_{4,4}$
19	m_8	9	$d_5',a_5,b_4',c_4 \quad a_{5,8}\neq c_{4,8}$

其原理基于第二轮的变换函数 $G(X,Y,Z)=(X\wedge Y)\vee(X\wedge Z)\vee(Y\wedge Z)$。通过分析以上变换发现存在以下性质:$Y!=Z$当且仅当$G(X,Y,Z)!=G(\neg X,Y,Z)$;$X!=Z$ 当且仅当$G(X,Y,Z)!=G(X,\neg Y,Z)$;$X!=Z$ 当且仅当 $G(X,Y,Z)!=G(X,\neg Y,Z)$。利用此性质,通过轮函数的传递性,借助 m_{14}和 m_{15}达到修改高级明文的目的。

完成对以上列出的所有条件的修改以后,通过枚举 m_{14}和 m_{15}找到满足所有充分条件的语句 M,通过筛选得到可以被字符编码格式识别的语句。为了提高碰撞的概率,在枚举 m_{14}和 m_{15}时最好采用不同的随机过程分别生成 m_{14}和 m_{15}。最后通过公式 $M'=M+\Delta M$ 计算出 M'。

使用以上方法得到基于不同编码格式的碰撞实例,表 11 是以 GBK 为编码的碰撞实例。

表 11　16 进制表示无消息填充的碰撞 M 和 M'

M	张小三借给刘小二 21186216 元。担保人:孙利。公证人:赵浴。 校? ☼}嶟 a1d0c5d5 e8bdfdc8 f5c1f8b8 feb6a1d0 38313132 36313236 a3a1aad4 a3b1a3b5 baa3cbc8 fbc0efcb abb9a3a1 cbc8a4d6 d4d5baa3 a3a1a1d4 2eb78096 fd8d7d0f
M'	张小三借给刘小二 31186216 元。担保人:孙利。公证人:赵浴。 校? ☼}嶟 a1d0c5d5 e8bdfdc8 f5c1f8b8 feb6a1d0 38313133 36313236 a3a1aad4 a3b1a3b5 baa3cbc8 fbc0efcb abb9a3a1 cbc8a4d6 d4d5baa3 a3a1a1d4 2eb78096 fd8d7d0f
H	66fdf604adbf35b1703d8f50f9662270

表 12 列出了基于 UTF-8 的编码格式的碰撞实例。

表 12　16 进制表示无填充消息 M 和 M'的碰撞

M	Joy Smish spent 22242227 euros on jack's jewelry store. @鈥? 20796f4a 73696d53 70732068 20746e65 34323232 37323232 72756520 6f20736f 616a206e 73276b63 77656a20 79726c65 6f747320 202e6572 40e3fe00 8455e209
M'	Joy Smish spent 32242227 euros on jack's jewelry store. @鈥? 20796f4a 73696d53 70732068 20746e65 34323232 37323232 72756520 6f20736f 616a206e 73276b63 77656a20 79726c65 6f747320 202e6572 40e3fe00 8455e209
H	e65df39f49c5fd27e2790b33493f4899

4　其他应用实例

本节介绍一个基于 UTF-8 编码的 python 脚本被篡改的实例。

构造的 python 脚本如下:

```
b=138376
if b==138376:print('OK')
else:print('NO') #di■{4ZlwW
```

由于在 python 中'#'后面表示注释为不可执行语句,因此其可以作为冗余位进行构造。运行以上脚本输出结果"OK"。分析这个脚本,首先确定自由变量,其中变量名 b 可以是任意的小写字母,数字可以是首位不为零的任意组合,同时要保证两个数字相等,第 17 位必须为数字。根据碰撞路线可知差分出现在第 17 位。利用第 3 节的方法最终确定脚本的前 54 个字符为:

```
b=138376
if b==138376:print('OK')
else:print('NO') #
```

由于'#'后面为注释,因此不影响运行结果,通过遍历'#'后面的语句得到的结果如表 13 和表 14 所列。

表 13　原消息 M

```
b=138376
if b==138376:print('OK')
else:print('NO') #di■{4ZlwW
```

33313d62 36373338 66690a0d 3d3d6220 33383331 703a3637 746e6972 4b4f2728 0a0d2927 65736c65 6972703a 2728746e 29274f4e 69642320 1d347b08 57776c5a

表 14　差分消息 M'

```
b=138376
if b==238376:print('OK')
else:print('NO') #di■{4ZlwW
```

33313d62 36373338 66690a0d 3d3d6220 33383332 703a3637 746e6972 4b4f2728 0a0d2927 65736c65 6972703a 2728746e 29274f4e 69642320 1d347b08 57776c5a

将两份脚本保存在 txt 文件中,后缀名改为. py,第一个脚本输出结果为"OK",第二个脚本输出结果为"NO",但是两个脚本的 hash 值是相同的,其 hash 值为:c21b9904 be58cdfa 2c2e1ecc 7ab28f92。

这个实例表明,在实际应用中利用构造有意义碰撞的方法有可能伪造数字证书甚至应用程序,从而针对特定的软件进行修改、插入危害性的代码,绕过官方的验证,其危害性非常大。

(下转第 177 页)

[7] LI Y,SUN L,ZHAO C,et al. A digital self-interference cancellation algorithm based on spectral estimation in co-time co-frequency full duplex system[C]// 2015 10th International Conference on Computer Science & Education (ICCSE). IEEE,2015:412-415.

[8] MUKHERJEE A,FAKOORIAN S A A,HUANG J,et al. Principles of physical layer security in multiuser wireless networks:A survey[J]. IEEE Communications Surveys & Tutorials,2014,16(3):1550-1573.

[9] LI L,HUANG C,CHEN Z. Cooperative secrecy beamforming in wiretap interference channels[J]. IEEE Signal Processing Letters,2015,22(12):2435-2439.

[10] GOEL S,NEGI R. Guaranteeing secrecy using artificial noise[J]. IEEE Transactions on Wireless Communications,2008,7(6):2180-2189.

[11] CHEN J,ZHANG R,SONG L,et al. Joint relay and jammer selection for secure decode-and-forward two-way relay networks[C]// IEEE International Conference Proc. of communications (ICC). 2011.

[12] CHEN J,ZHANG R,SONG L,et al. Joint relay and jammer selection for secure two-way relay networks[J]. IEEE Transactions on Information Forensics and Security,2012,7(1):310-320.

[13] FRAGKIADAKIS A G,TRAGOS E Z,ASKOXYLAKIS I G. A survey on security threats and detection techniques in cognitive radio networks[J]. IEEE Communications Surveys & Tutorials,2013,15(1):428-445.

[14] PEI Y,LIANG Y C,ZHANG L,et al. Secure communication over MISO cognitive radio channels[J]. IEEE Transactions on Wireless Communications,2010,9(4):1494-1502.

[15] PEI Y,LIANG Y C,TEH K C,et al. Secure communication in multiantenna cognitive radio networks with imperfect channel state information[J]. IEEE Transactions on Signal Processing,2011,59(4):1683-1693.

[16] WANG C,WANG H M. On the secrecy throughput maximization for MISO cognitive radio network in slow fading channels[J]. IEEE Transactions on Information Forensics and Security,2014,9(11):1814-1827.

[17] ALAVI F,SAEEDI H. Radio resource allocation to provide physical layer security in relay-assisted cognitive radio networks[J]. IET Communications,2015,9(17):2124-2130.

[18] ZOU Y,ZHU J,YANG L,et al. Securing physical-layer communications for cognitive radio networks[J]. IEEE Communications Magazine,2015,53(9):48-54.

[19] ZHU F,YAO M. Improving Physical-Layer Security for CRNs Using SINR-Based Cooperative Beamforming[J]. IEEE Transactions on Vehicular Technology,2016,65(3):1835-1841.

[20] LUO Z Q,MA W,SO A M C,et al. Semidefinite relaxation of quadratic optimization problems[J]. IEEE Signal Processing Magazine,2010,27(3):20.

[21] GRANT M,BOYD S,YE Y. CVX:Matlab software for disciplined convex programming[J]. Golbal Optimization,2008:155-210.

[22] SIDIROPOULOS N D,DAVIDSON T N,LUO Z Q. Transmit beamforming for physical-layer multicasting[J]. IEEE Transactions on Signal Processing,2006,54(6):2239-2251.

(上接第 171 页)

结束语　本文给出了快速寻找 MD4 算法有意义碰撞的通用方法,并给出了有意义碰撞的实例,为构造其他已经被破解的杂凑算法有意义的碰撞提供了思路。

参 考 文 献

[1] RIVEST R L. The MD 4 message-digest algorithm[C]// CPYPTO 1990. LNCS,1990:303-312.

[2] BOER B D,BOSSELAERS A. An attack on the last two rounds of MD4[C]//CRYPTO 1991. LNCS 576,1991:194-203.

[3] VAUDENAY S. On the need for multipermutations:Cryptanalsis of MD4 and SAFER[C]//FSE 1995. LNCS 1008,1995:286-297.

[4] DOBBERTIN H. Cryptanalysis of MD4[J]. Journal of Cryptology,1998,11(4):253-271.

[5] WANG X,FENG D,LAI X,et al. Collisions for hash functions MD4,MD5, HAVAL-128 and RIPEMD[OL]. http://eprint.iacr.org/2004/199.pdf.

[6] WANG X,YU H. How to break MD5 and other hash functions[C]//EUROCRYPT 2005,LNCS. 2005:19-35.

[7] WANG X,YIN Y L,YU H. Finding Collisions in the Full SHA-1[C] // International Cryptology Conference on Advances in Cryptology-CRYPTO. Springer-Verlag,2005:17-36.

[8] YU H B,WANG G L,ZHANG G Y,et al. The Second-Preimage Attack on MD4[C]//CANS 2005. LNCS 3810,2005:1-12.

[9] JIA K,WANG X. Meaningful Collision Attack on MD4[J]. Journal of Frontiers of Computer Science & Technology,2010,3:202-213.

[10] BAI D X. Safety analysis of some block cipher and hash function[D]. Beijing:Tsinghua University,2015. (in Chinese)
白东霞. 几个分组密码和杂凑函数的安全性分析[D]. 北京:清华大学,2015.

[11] LANDELLE F,PEYRIN T. Cryptanalysis of Full RIPEMD-128[J]. Journal of Cryptology,2015,7881:1-25.

[12] CHENG K,HAN W B. Automatic construction algorithm of MD4 differential path [J]. Journal of Information Engineering University,2014,15(2):129-133. (in Chinese)
程宽,韩文报. MD4 差分路径的自动化构造算法[J]. 信息工程大学学报,2014,15(2):129-133.

[13] WANG G L. Collision Attack on the Full Extended MD4 and Pseudo-Preimage Attack on RIPEMD[J]. Journal of Computer Science and Technology,2013,28(1):129-143.

[14] LI Q,TANG B,YANG J. Key Technology Research for Content Supervision Based on KAD Network[C]// International Conference on Multimedia & Image Processing. IEEE Computer Society,2016:72-77.