Finding MD5 Collisions – a Toy For a Notebook

Vlastimil Klíma [1]
Prague, Czech Republic
http://cryptography.hyperlink.cz
v.klima@volny.cz

March 5, 2005

Abstract. In this short memo, we summarize the results achieved during a two and half months long research. Further details will be provided in a forthcoming paper.

One of the major cryptographic "break-through" of the recent years was a discovery of collisions for a set of hash functions (MD4, MD5, HAVAL-128, RIPEMD) by the Chinese cryptographers in August 2004 [1]. Their authors (Wang et al.) kept the algorithm secret, however. During October 2004, the Australian team (Hawkes et al.) tried to reconstruct the methodology in their great work [3]. The most important "Chinese trick" was not discovered, although they succeeded in describing a differential scheme of conditions that hold for the published collisions. Nevertheless, fulfilling the conditions of this scheme has been still more computationally difficult in comparison to what the results of [1] showed.

During our research, we also analyzed the available data using differential cryptanalysis. We have found a way to generate the first message block of the collision about 1000 - 2000 times faster than the Chinese team - that corresponds to reaching the first colliding block in 2 minutes using a common notebook (PC platform). The same computation phase took the Chinese team about an hour using an IBM P690 supercomputer. On the other hand, the Chinese team was 2 - 80 times faster when computing the second message block of their collisions. Therefore, our and the Chinese methods probably differs in several details in both parts of the computation. Overall, our method is about 3 - 6 times faster. More specifically, finding the first (complete) collision took 8 hours using a notebook PC (Intel Pentium 1.6 GHz). Note that our method works for any initialization vector. It can be abused in forging signatures of software packages and digital certificates as some papers show ([4], [5], [6]). We have shown that it is possible to find MD5 collisions using an ordinary home PC. That should be a warning towards persisting usage of MD5. In the appendix, we show new examples of collisions for a standard and chosen initialization vectors.

[1] This research has been done during Christmas vacation and during January and February 2005. At that time the author has been working for the company LEC, s.r.o., Prague, Czech Republic which supported the project by material and financial means.

Introduction

Hash functions are very useful cryptographic tool. To be one-way and collision-free, the hash functions have to be very robust and complex. Therefore, it is always exciting when a collision is found. One of the most important cryptanalytic articles of the last year was precisely the work of Chinese team [1]. MD5 was the most challenging hash function, so we will further focus only on this function.

Let us remind that there was no algorithm nor explanation in [1] as to how to find the collisions, only some brief data was provided which we shall recall now: The colliding pair of messages (M, N) and (M', N') consist of two message blocks. The first blocks differ only in a predefined constant vector C1 (M' = M + C1) and the second blocks also differ only in predefined constant vector C2 = -C1 mod 2^{32} (N' = N + C2) whereas MD5(M, N) = MD5(M', N').

Wang et al. stated that it takes about an hour to find the block M using their supercomputer IBM P690. Finding N then takes only 15 seconds to 5 minutes. In the first version of [1], two pairs of colliding messages were presented. The initialization vector (IV) value the authors chose, however, was not the one used in MD5 algorithm, since the order of bytes was opposite (little-endian vs. big-endian). In the corrected version of the paper, Wang et al. showed two pairs of colliding messages for MD5, with the right IV that time. They made a remark that their attack works for any initializing value IV.

After having published their results, we had only four pairs of colliding messages. Nevertheless, it has been shown that even these data could be used to mount a successful attack [4], [5]. It is shown in [4] that a single collision is enough to create a pair of different self-extracting archives with identical hash value. It can be abused, for example, to put backdoors into large software packages during their distribution. Furthermore, it has been shown in cooperation with one of the authors of [1], how to forge a digital certificate using the ability of making collision for any initialization vector, too.

A work by Hawkes et al. [3] has been published in October 2004. The authors tried to unveil the "Chinese method" basing on the raw data provided in [1]. In this work, they inspect inner differences and conditions for messages to hold in order to create a collision using the Chinese method. It was the first rigorous analysis and attempt to explain the Chinese method. Basing on one pair of colliding messages (with the correct IV), the authors described the differential scheme that the published collision (the one with correct IV) satisfies. This scheme was probably in the background of the collision design. However, they did not manage to explain how this schema was created. Furthermore, they described what conditions must hold for one message of the colliding pair so that the differential scheme is fulfilled. A long list of conditions was acquired. The first set of conditions (so called ft- and Tt-conditions) comes up by the first block passing the 64 rounds of MD5.

If the conditions are met in the first 16 rounds (more than 200 conditions) by well selecting M block, 39 ft-conditions and "3.2" Tt-conditions are left to be fulfilled in the remaining rounds. These conditions are met only on probabilistic basis. To sum it up, we need to generate about $2^{42.2}$ messages M to find the one, such that it meets all the ft- and Tt- conditions from rounds 17 - 64. Similarly, to fulfill the ft- and Tt- conditions for the second block N, it is necessary to generate $2^{42.2}$ messages. The overall complexity is then 2^{43}. Hawkes et al. judge that the complexity is too large for the collision to be computed in one hour. Basing on that observation, they concluded that Wang et al. must have used another trick. Obviously, this trick is the key trick.

In our research, we started by the results of [3] and observed the differential scheme from the point of additive differences (arithmetic difference modulo 2^{32}) and binary differences (xor, modulo 2), in the same way as in [3]. Furthermore, we examined other colliding pair, which was the one with wrong IV. We verified that the differential scheme holds for both colliding pairs, however there was no more data available. In our research, it emerged that some of the ft- and Tt- conditions could have been met in different ways than those Hawkes et al. chose. That could theoretically decrease the computational complexity. However, it would lead to increasing the complexity of the collision-finding program and its memory demands. Therefore, we did not go this way. Yet, the analysis of ft- and Tt- conditions has shown that the real complexity for finding collisions could be smaller than the one in the theoretical model. As the research went further, we found a way to generate the first blocks of colliding messages very quickly. Using a standard PC notebook it took 2 minutes to find the first block of the message whereas it took one hour using the supercomputer [1]. Due to the briefness of research we did not go further in speeding up the search for second blocks as we did for the first one, even though we reached the complexity significantly lower than 2^{42} (according to [3]). The fact that we are able to find the collision in 8 hours using the PC notebook attests that. According to [1], the search for the second block should be 12 - 240 times faster than searching for the first block. That would yield a collision in 2 minutes instead of 8 hours on a notebook.

We did not use any supercomputer to find the collisions, just ordinary desktop computers. The author conducted his experiments on his notebook where he found tens of thousands of collisions for the first block and subsequently complete MD5 collisions for both original IV and chosen IVs. To test the program functionality, the author asked a few friends to try it on their own computers. In this way, during a week of experimentation, tens of thousands first-block collisions and a few dozens of full collisions were found.

The results obtained using an ordinary notebook (Acer Travelmate 450 LMi, Intel Pentium 1.6 GHz) are as follows: During 8 hours, 331 collisions of first block were found and one complete collision was reached. According to the fact that it took the Chinese team one hour to find the first block collision, searching for 331 of these collisions would take 331 hours, which is 40 times more. It is hard to compare the power of a notebook and a supercomputer due to different architectures, but if we take to account that the IBM P690 is about 25 - 50 times faster than the notebook (estimate provided by Ondrej Mikle based on simple bogomips ratio) we get the result that our method of searching for first-block collision is 1000 - 2000 times faster than the one in [1]. On the other hand, the searching for second-block collision is 2 - 80 times slower. Overall, if we compare the time to find a complete collision by Chinese team (1 - 1.08 hour) with us (8 hours) on 25 - 50 times slower machine, our method is 3 - 6 times faster. All these comparisons are only for illustrational purposes and the author makes no claim of their accuracy (except for the time values).

It turns out that:

- MD5 collision can be found using a notebook,
- Our method and the Chinese method [1] are different in terms of speed and probably also an approach (in both parts of the computation),
- Our method is faster overall,
- The method works for any chosen IV.

Acknowledgements

I would like to thank my friends for their help. To Milan Nosál (LEC, s.r.o) for his help debugging the program, to Tomáš Rosa, Ondřej Pokorný and Milan Nosál for conducting experiments on their home computers, to Tomáš Jabůrek for technical help with experiments, to Ondrej Mikle for helping with the translation of the paper and to all for valuable comments.

Note

In the last experiment, provided by Ondřej Pokorný on his home PC (Intel Pentium, 1GHz), he obtained 14 collisions in 58 hours and 32 minutes. It gives even more optimistic time for finding a collision (1 collision per 4 hours 11 minutes) than on the author´s notebook.

Homepage of the project

http://cryptography.hyperlink.cz/MD5_collisions.html

Conclusion

The paper shows that nowadays, the MD5 collision can be found using only a PC notebook. The method works for any IV and is faster than the original Chinese method. It may be expected that after publishing the Chinese method the overall time for finding a complete collision can fall down to as less as 2 minutes on a PC notebook.

References

[1] Xiaoyun Wang, Dengguo Feng , Xuejia Lai, Hongbo Yu: Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD, rump session, CRYPTO 2004, *Cryptology ePrint Archive*, Report 2004/199, first version (August 16, 2004), second version (August 17, 2004), http://eprint.iacr.org/2004/199.pdf

[2] Ronald Rivest: The MD5 Message Digest Algorithm, RFC1321, April 1992, ftp://ftp.rfc-editor.org/in-notes/rfc1321.txt

[3] Philip Hawkes, Michael Paddon, Gregory G. Rose: Musings on the Wang et al. MD5 Collision, *Cryptology ePrint Archive*, Report 2004/264, 13 October 2004, http://eprint.iacr.org/2004/264.pdf

[4] Vlastimil Klima: Several observations regarding Chinese collisions of MD5, 3rd International Scientific Conference *Security and Protection of Information*, Brno, Czech Republic, May 3 - 5, 2005, http://www.unob.cz/spi/defaulten.asp, in preparation

[4] Ondrej Mikle: Practical Attacks on Digital Signatures Using MD5 Message Digest, *Cryptology ePrint Archive*, Report 2004/356, http://eprint.iacr.org/2004/356, 2nd December 2004

[5] Dan Kaminsky: MD5 To Be Considered Harmful Someday, *Cryptology ePrint Archive*, Report 2004/357, http://eprint.iacr.org/2004/357, 6 December 2004

[6] Arjen Lenstra, Xiaoyun Wang and Benne de Weger: Colliding X.509 Certificates, *Cryptology ePrint Archive*, Report 2005/067, http://eprint.iacr.org/2005/067

Appendix: Examples

Example: MD5 collision with the standard IV

```
IV according to [2]:
  context->state[0] = 0x67452301;
  context->state[1] = 0xefcdab89;
  context->state[2] = 0x98badcfe;
  context->state[3] = 0x10325476;
First message:
0xA6,0x64,0xEA,0xB8,0x89,0x04,0xC2,0xAC,
0x48,0x43,0x41,0x0E,0x0A,0x63,0x42,0x54,
0x16,0x60,0x6C,0x81,0x44,0x2D,0xD6,0x8D,
0x40,0x04,0x58,0x3E,0xB8,0xFB,0x7F,0x89,
0x55,0xAD,0x34,0x06,0x09,0xF4,0xB3,0x02,
0x83,0xE4,0x88,0x83,0x25,0x71,0x41,0x5A,
0x08,0x51,0x25,0xE8,0xF7,0xCD,0xC9,0x9F,
0xD9,0x1D,0xBD,0xF2,0x80,0x37,0x3C,0x5B,
0x97,0x9E,0xBD,0xB4,0x0E,0x2A,0x6E,0x17,
0xA6,0x23,0x57,0x24,0xD1,0xDF,0x41,0xB4,
0x46,0x73,0xF9,0x96,0xF1,0x62,0x4A,0xDD,
0x10,0x29,0x31,0x67,0xD0,0x09,0xB1,0x8F,
0x75,0xA7,0x7F,0x79,0x30,0xD9,0x5C,0xEB,
0x02,0xE8,0xAD,0xBA,0x7A,0xC8,0x55,0x5C,
0xED,0x74,0xCA,0xDD,0x5F,0xC9,0x93,0x6D,
0xB1,0x9B,0x4A,0xD8,0x35,0xCC,0x67,0xE3.

Second message:
0xA6,0x64,0xEA,0xB8,0x89,0x04,0xC2,0xAC,
0x48,0x43,0x41,0x0E,0x0A,0x63,0x42,0x54,
0x16,0x60,0x6C,0x01,0x44,0x2D,0xD6,0x8D,
0x40,0x04,0x58,0x3E,0xB8,0xFB,0x7F,0x89,
0x55,0xAD,0x34,0x06,0x09,0xF4,0xB3,0x02,
0x83,0xE4,0x88,0x83,0x25,0xF1,0x41,0x5A,
0x08,0x51,0x25,0xE8,0xF7,0xCD,0xC9,0x9F,
0xD9,0x1D,0xBD,0x72,0x80,0x37,0x3C,0x5B,
0x97,0x9E,0xBD,0xB4,0x0E,0x2A,0x6E,0x17,
0xA6,0x23,0x57,0x24,0xD1,0xDF,0x41,0xB4,
0x46,0x73,0xF9,0x16,0xF1,0x62,0x4A,0xDD,
0x10,0x29,0x31,0x67,0xD0,0x09,0xB1,0x8F,
0x75,0xA7,0x7F,0x79,0x30,0xD9,0x5C,0xEB,
0x02,0xE8,0xAD,0xBA,0x7A,0x48,0x55,0x5C,
0xED,0x74,0xCA,0xDD,0x5F,0xC9,0x93,0x6D,
0xB1,0x9B,0x4A,0x58,0x35,0xCC,0x67,0xE3.

Common MD5 hash:
0x2B,0xA3,0xBE,0x5A,0xA5,0x41,0x00,0x6B,
0x62,0x37,0x01,0x11,0x28,0x2D,0x19,0xF5.
```

Example: MD5 collision with a chosen IV

```
context->state[0] = 0xabaaaaaa;
context->state[1] = 0xaaacaaaa;
context->state[2] = 0xaaaadaaa;
context->state[3] = 0xaaaaaaea;

First message:
0x9E,0x83,0x2A,0x4C,0x95,0x64,0x5E,0x2B,
0x2E,0x1B,0xB0,0x70,0x47,0x1E,0xBA,0x13,
0x7F,0x1A,0x53,0x43,0x22,0x34,0x25,0xC1,
0x40,0x04,0x58,0x3E,0xB8,0xFB,0x7F,0x89,
0x55,0xAD,0x34,0x06,0x09,0xF4,0xB3,0x02,
0x83,0xE4,0x88,0x83,0x25,0x71,0x41,0x5A,
0x08,0x51,0x25,0xE8,0xF7,0xCD,0xC9,0x9F,
0xD9,0x1D,0xBD,0xF2,0x80,0x37,0x3C,0x5B,
0x89,0x62,0x33,0xEC,0x5B,0x0C,0x8D,0x77,
0x19,0xDE,0x93,0xFA,0xA1,0x44,0xA8,0xCC,
0x56,0x91,0x9E,0x47,0x00,0x0C,0x00,0x4D,
0x40,0x29,0xF1,0x66,0xD1,0x09,0xB1,0x8F,
0x75,0x27,0x7F,0x79,0x30,0xD5,0x5C,0xEB,
0x42,0xE8,0xAD,0xBA,0x78,0xCC,0x55,0x5C,
0xED,0xF4,0xCA,0xDD,0x5F,0xC5,0x93,0x6D,
0xD1,0x9B,0x0A,0xD8,0x35,0xCC,0xE7,0xE3.

Second message:
0x9E,0x83,0x2A,0x4C,0x95,0x64,0x5E,0x2B,
0x2E,0x1B,0xB0,0x70,0x47,0x1E,0xBA,0x13,
0x7F,0x1A,0x53,0xC3,0x22,0x34,0x25,0xC1,
0x40,0x04,0x58,0x3E,0xB8,0xFB,0x7F,0x89,
0x55,0xAD,0x34,0x06,0x09,0xF4,0xB3,0x02,
0x83,0xE4,0x88,0x83,0x25,0xF1,0x41,0x5A,
0x08,0x51,0x25,0xE8,0xF7,0xCD,0xC9,0x9F,
0xD9,0x1D,0xBD,0x72,0x80,0x37,0x3C,0x5B,
0x89,0x62,0x33,0xEC,0x5B,0x0C,0x8D,0x77,
0x19,0xDE,0x93,0xFA,0xA1,0x44,0xA8,0xCC,
0x56,0x91,0x9E,0xC7,0x00,0x0C,0x00,0x4D,
0x40,0x29,0xF1,0x66,0xD1,0x09,0xB1,0x8F,
0x75,0x27,0x7F,0x79,0x30,0xD5,0x5C,0xEB,
0x42,0xE8,0xAD,0xBA,0x78,0x4C,0x55,0x5C,
0xED,0xF4,0xCA,0xDD,0x5F,0xC5,0x93,0x6D,
0xD1,0x9B,0x0A,0x58,0x35,0xCC,0xE7,0xE3.

Common hash:
//value corrected on March 8, thanks to Jan Kasprzak
0xef,0x2e,0xae,0x54,0xe0,0x34,0x70,0x7c,
0xa2,0x6e,0xb0,0x9b,0x45,0xc7,0xe4,0x87.
```