IST-2002-507932

ECRYPT

European Network of Excellence in Cryptology

Network of Excellence

Information Society Technologies

Recent Collision Attacks on Hash Functions: ECRYPT Position Paper

Start date of project: 1 February 2004 Duration: 4 years

Lead contractor: Katholieke Universiteit Leuven (KUL)

Revision 1.1

Project co-funded by the European Commission within the 6th Framework Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission services)	
RE	Restricted to a group specified by the consortium (including the Commission services)	
CO	Confidential, only for members of the consortium (including the Commission services)	

Recent Collision Attacks on Hash Functions: ECRYPT Position Paper

ECRYPT Network of Excellence
www.ecrypt.eu.org

17. February 2005
Revision 1.1

The work described in this report has in part been supported by the Commission of the European Communities through the IST program under contract IST-2002-507932. The information in this document is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Contents

1 Introduction

At the Crypto 2004 conference in August 2004, a number of attacks on different cryptographic hash functions were presented. These results have gained some attention within the community, the media, and Internet discussion forums. Partners within the ECRYPT Network of Excellence have observed some confusion as to what the *practical implications* of these results might be, and so the purpose of this document is to provide some perspective to these results and to make recommendations about the continued use of certain hash functions.

NEWS FLASH: On February 15 2005 it was announced on the Internet, [21], that new attacks on SHA-1 had been found, obtaining collisions in 2^{69} complexity. The present version of this document briefly discusses impact of this new attack, and supersedes the earlier version 1.0 of Nov 29, 2004.

2 Cryptographic Hashing—A Short Background

Cryptographic hash functions are widely used in computer and network security applications. Depending on the application, some or all of a number of security properties are required from the hash function $h(\cdot)$. Potential properties that we might appeal to are:

Pre-image resistance: Given an output $y = h(x)$ (but not a corresponding input x) it is practically infeasible to find x. Such a property might be useful when one wishes to commit to the value x at some point in time while keeping the value of x secret until later.

2nd pre-image resistance: Given an output $y = h(x)$ *and* a corresponding input x it is practically infeasible to find another input $z \neq x$ such that $h(z) = h(x)$. This property might be used to prevent some party from changing from a committed value.

Collision resistance: It is practically infeasible to find *any* pair of distinct inputs x and z such that $h(x) = h(z)$. This property is often required to protect electronic signature schemes against forgeries. In such schemes the hash of a message is typically signed as a representation of that message. Thus if an attacker can find two inputs x and z that collide with some hash function, then the attacker might be able to re-use a legitimate signature on x as a correct, but falsely-obtained, signature on z.

Random oracle property: The function $h(\cdot)$ "behaves" as a randomly chosen function. Assuming this property holds sometimes makes it possible to formally prove the security of public key encryption and signature schemes.

While one can construct contrived examples of functions that are collision resistant but not pre-image resistant, the hash function properties have been ordered in terms of the difficulty faced by an opponent, with the task of finding a pre-image being the hardest. In the absence of any analytic weaknesses, only brute force methods are available to the attacker. More precisely, if n is the size of the hash outputs, one would from a secure hash expect around 2^n operations to be required to break the first two properties (though this decreases as the number of available targets increase) and around $2^{n/2}$ operations to break the property of collision resistance (due to the birthday paradox). Consequently, one needs to choose a secure h with a large enough n so that these numbers meet application-dependent requirements on “practical infeasibility”.

As we will see, the attacks on hash functions presented at Crypto 2004 (and the more recent, February 2005 attacks) are attacks against the collision resistance property for some hash functions defined for $n = 128$ respectively $n = 160$. Strictly speaking, collision attacks also invalidate the random oracle property, but it should be noted that it was already known that no fixed function can really meet “random oracle requirements” in a generic sense [3].

3 Hash Function Collisions at Crypto 2004

Various design techniques are used to implement hash functions. A popular one, used by all the recently attacked functions, is to base the design on a *compression function* which maps m-bit blocks to n-bit blocks ($m > n$). This use of a “secure” compression function is often referred to as the Merkle-Damgård construction, and the initial value for the first iteration of the compression function is fixed and publicly known. The compression function is, in turn, built up by iterating simpler operations over a number of *rounds*. An important feature within the compression function is the way that the message is processed (the message schedule) with some of the earlier hash function designs using simpler methods than those adopted in later hash function designs. For more information on hash functions see [11, 16].

When analysing a hash function, a starting point for cryptanalysts would likely be some modification of the compression function, perhaps by reducing the number of rounds of computation. While attacks against heavily round-reduced variants are usually unavoidable, as the number of compromised rounds begins to approach the full number of rounds in the compression function, security doubts about the hash function may well appear. Before describing the attacks presented at Crypto 2004, we feel that it is important to appreciate the relation between different hash function proposals.

- MD4 [19] is viewed as the starting point for the hash function families considered in this note. Collisions and results compromising the one-way property of MD4 have been known for some time.

- MD5 [20] is a successor to MD4. While contemporary applications might no longer use MD5, it is still widely used in fielded applications and products. MD5 is the subject of recent results.

- SHA [12] (equivalently SHA-0) is the starting point for the SHA-family of NIST-standardised hash functions and contains some design features found in MD4 and MD5. However SHA was withdrawn by NIST in 1995. SHA is the subject of recent results but we do not expect SHA to feature in any fielded application or product.

- SHA was replaced by an algorithm referred to as SHA-1 [13] . Since then, more complicated hash functions with longer hash outputs have been published in FIPS 180-2. These later hash functions are sometimes casually referred to as SHA-2, but we find this nomenclature confusing and will refer to them instead as being algorithms within FIPS 180-2.

- SHA-1 has been, and continues to be, used widely. Some of the recent results apply to **weakened variants** of SHA-1. These results do not apply directly to the algorithms in FIPS 180-2 though some work on these hash functions has been taking place [7]. As mentioned, at the time of finalizing this report, credible reports on attacks applicable to (non-weakend) SHA-1 have surfaced.
- RIPEMD [18] is the starting point for the RIPEMD-family of hash functions. RIPEMD was superseded by the hash functions RIPEMD-128 and RIPEMD-160 [6]. Recent results apply to the original RIPEMD and not to these strengthened successors.

At Crypto 2004, Biham and Chen published a new cryptanalytic approach that built on earlier results by Chabaud and Joux [4]. This technique, called the neutral bit technique, leads to near-collisions on the compression function of SHA [2]. This technique also provides collisions on reduced-round versions of SHA-1 (attacking up to 53 of the 80 rounds).

At the rump session of the same conference, two independent groups of researchers also announced that they had found examples of collisions for some important hash functions. First, Joux, Carribault, Jalby and Lemuet [9] provided a full collision on SHA using two four-block (2048-bit) messages and requiring a complexity of 2^{51} compression function computations. Second, Wang *et al.* [22] published full collisions on MD4, MD5, HAVAL-128 and RIPEMD each using two two-block (1024-bit) messages and requiring very low computational complexity (just a matter of hours at most). Wang *et al.* also announced an estimated computational complexity of 2^{40} compression function computations to find a full collision on SHA but did not provide an example. However, as mentioned, in mid February 2005, reports have started to appear about attacks on full SHA-1 by the same researchers, see [21] for a summary. While no details are yet available, it is claimed that collisions on SHA-1 can be found in 2^{69} complexity, and we believe these reports to be credible. Considering the complexity of this attack, it is unlikely that an explicit collision has (yet) been produced.

Finally, from Joux's paper [8] (also from Crypto 2004) together with the results of [22] it follows that, in the case of MD5, it is easy to construct millions of longer messages that hash to the same value.

4 Current Implications

As a consequence of these breakthroughs, we believe hash functions using a simple message schedule such as those derived from the MD4 type construction are at risk for use in real-life implementations. These include MD4, MD5, RIPEMD, HAVAL and SHA.

With the recent announcements of attacks on full SHA-1, while not as efficient as those on e.g. MD5, there is also some reason to be cautious with using SHA-1.

More complex hash functions such as those listed in FIPS 180-2 do not seem to be at immediate risk as their message schedule is more complex and similar to a key schedule for block ciphers.

4.1 Implications for MAC constructions

Hash functions might be used in a MAC (Message Authentication Codes) construction such as HMAC [1]. This construction is provably secure under certain assumptions on the security of the underlying hash function.

The security proof for HMAC focuses on the role of the compression function. While today's hash functions use a fixed (and publicly known) initial value for the first iteration of the compression function, the security of the HMAC construction depends on the security of the underlying compression function when the fixed and public initial value is replaced by a random secret value. HMAC also depends on certain reasonable assumptions about the predictability of the output from the compression function when secret text and/or a random and secret initial value is used. The proof of security for the HMAC construction shows that an attacker who is able to find a valid authentication tag (MAC) for a previously unseen input, would also be able to break the underlying hash function in one of two ways:

1. The attacker would be able to find collisions on the underlying hash function used with a random and secret initial value.
2. The attacker would be able to find an output of the compression function used with a random and secret initial value.

It is currently doubtful that recent attacks on hash functions would enable the attacker to find collisions starting from random secret initial values, or even to produce known outputs from random secret initial values. In the current setting, the ability to find collisions on standard hash functions does not imply the ability to forge MACs and the HMAC construction remains secure unless more sophisticated attacks are found that allow additional malleability in the attacks on the hash function.

4.2 Implications for Digital Signatures

For both performance and security reasons, digital signature primitives are never used alone, but in combination with an initial encoding step. In many schemes, e.g. PKCS#1, this encoding involves processing the message through a hash function. While additional pre-processing operations such as padding may well take place between the hash function and the signature primitive, such operations have no impact on the attacks presented in this note. A collision attack against a hash function might be exploited as follows:

1. An attacker finds a pair of messages x and y such that $h(x) = h(y)$ (intuitively, x is an "innocent" and y is a "compromising" message).
2. The attacker convinces her victim to sign the innocent message x.
3. The attacker then exhibits message y and the signature of x; since both messages hash to the same value, this is also a valid signature for y.

Collision attacks are thus a real concern in the context of digital signatures. Theoretically, the attack of Wang *et al.* can produce thousands of pairs of messages having the same signature at a reasonable cost.

However, some practical considerations limit the potentials of this attack in real-life situations. Most importantly, colliding pairs currently require a very specific difference between them, and the message value in the two 512-bit blocks cannot be freely chosen (but the opponent has some very limited control over them). While this constraint may make it difficult in many applications to find meaningful colliding messages, one should be aware that the two 512-bit blocks can be preceded and followed by much longer blocks that can be chosen freely as long as they are equal for the two messages. Moreover, further research may result in the removal of some of these constraints. Therefore, ECRYPT does not recommend the continued use of hash functions for which any collisions have been demonstrated when collision-resistance is required by the application.

It is important to mention that the integrity of existing signatures (for instance within digital certificates) need not rely on the collision resistance of a hash function. For existing signatures, an input and output for the hash function have already been established and fixed. An attacker attempting to re-use such a signature for a second message would then be required to find a second pre-image, and not a general collision. This is much harder for the adversary since the colliding value has been fixed ahead of time. The new attacks do not apply to this problem.

5 ECRYPT Recommendations

Other statements on the impact of the Crypto 2004 results are available [10, 15, 17]. The ECRYPT Network of Excellence makes the following recommendations:

1. In general, hash functions with outputs shorter than 160 bits, are not recommended unless the consequences of an attack resistance less than 2^{80} operations have been fully considered.

2. We see no immediate need to be concerned about the security of the HMAC construction used with either MD5 or SHA-1. However, cryptanalytic advances on MD5 suggest that it might be prudent to replace HMAC-MD5 with HMAC-SHA-1, or preferably HMAC-h for some still “collision resistant” hash function h, as soon as convenient.

3. Even though new attacks cannot currently produce highly controlled message collisions, ECRYPT does not recommend the continued use of MD5 in signature applications with medium to high security requirements. In light of the recently announced attacks on SHA-1, though we have no information on the severity/impact of the collisions that may be obtained, we also recommend to be cautious with new deployments of SHA-1, in particular since it cannot be excluded that the announced attacks will be improved in the near future. It seems unlikely that already existing SHA-1 based signatures are threatened.

4. ECRYPT sees no immediate need to replace RIPEMD-160 for current and near term (3-5 years) applications.

5. In the longer term, ECRYPT recommends, where possible, that the newer family of hash functions specified in FIPS 180-2, or alternatives such as Whirlpool [16], be considered.

References

[1] M. Bellare, R. Canetti and H. Krawczyk. Keying hash functions for message authentication. In *Advances in Cryptology - CRYPTO 96,* LNCS 1109, pages 1–15, Springer-Verlag, 1996.

[2] E. Biham and R. Chen. Near-Collisions of SHA-0. In *Advances in Cryptology - CRYPTO 2004,* LNCS 3152, pages 290–305, Springer-Verlag, 2004.

[3] R. Canetti, O. Goldreich, and S. Halevi. The Random Oracle Methodology, Revisited In *Proceedings of 30th Annual ACM Symposium on the Theory of Computing*, pages 209–218, May 1998, ACM.

[4] F. Chabaud and A. Joux. Differential Collisions in SHA-0. In *Advances in Cryptology - CRYPTO'98,* LNCS 1462, pages 56–71, Springer-Verlag, 1998.

[5] H. Dobbertin. Cryptanalysis of MD4. In D. Gollmann, editor, Proceedings of Fast Software Encryption 1996, LNCS 1039, pages 53–70, Springer-Verlag, 1996.

[6] H. Dobbertin, A. Bosselaers, and B. Preneel. RIPEMD-160: A Strengthened Version of RIPEMD. In D. Gollmann, editor, Proceedings of Fast Software Encryption 1996, LNCS 1039, pages 71–82, Springer-Verlag, 1996.

[7] P. Hawkes and G. Rose. On corrective patterns for the SHA-2 family. Available via `http://eprint.iacr.org/2004/207/`.

[8] A. Joux. Multicollisions in iterated hash functions. Application to cascaded constructions. Proceedings of Crypto 2004, LNCS 3152, pages 306–316.

[9] A. Joux, P. Carribault, W. Jalby and C. Lemuet. Collisions in SHA-0. Presented at the rump session of CRYPTO 2004, August 2004.

[10] A. Lenstra. Progress in hashing cryptanalysis. Available via `cm.bell-labs.com/who/akl/hash.pdf`, September 2004.

[11] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. The Handbook of Applied Cryptography. CRC Press. 1996.

[12] National Institute of Standards and Technology (NIST), *FIPS Publication 180: Secure Hash Standard*, May 11, 1993.

[13] National Institute of Standards and Technology (NIST), *FIPS Publication 180-1: Secure Hash Standard*, April 17, 1995. Available via `www.itl.nist.gov/fipspubs/fip180-1.htm`

[14] National Institute of Standards and Technology (NIST), *FIPS Publication 180-2: Secure Hash Standard*, August 1, 2002. Available via `csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf`

[15] National Institute of Standards and Technology (NIST), *Brief Comments on Recent Cryptanalytic Attacks on Secure Hashing Functions and the Continued Security Provided by SHA-1*, Available via `csrc.nist.gov/hash_standards_comments.pdf`.

[16] NESSIE consortium, NESSIE Security Report, version 2.0. Available via `www.cosic.esat.kuleuven.ac.be/nessie/deliverables/D20-v2.pdf`

[17] J. Randall and M. Szydlo. Collisions for SHA0, MD5, HAVAL, MD4, and RIPEMD, but SHA1 still secure. RSA Laboratories Technical Note, available via `www.rsasecurity.com/rsalabs/node.asp?id=2738`

[18] RIPE. *Integrity Primitives for Secure Information Systems, Final Report of RACE Integrity Primitives Evaluate (RIPE-RACE 1040)*, LNCS 1007, Springer-Verlag, 1995.

[19] R. Rivest. The MD4 Message Digest Algorithm. RFC 1320.
Available via `http://www.faqs.org/rfcs/rfc1320.html`.

[20] R. Rivest. The MD5 Message Digest Algorithm. RFC 1321.
Available via `http://www.faqs.org/rfcs/rfc1321.html`.

[21] SHA-1 announcement, available via
`www.schneier.com/blog/archives/2005/02/sha1_broken.html`

[22] X. Wang, X. Lai, D. Feng and H. Yu. Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD. Presented at the rump session of CRYPTO 2004, August 2004.