Collision Attacks on MD5 and SHA-1: Is this the "Sword of Damocles" for Electronic Commerce?

Praveen Gauravaram, Adrian McCullagh and Ed Dawson
Information Security Institute (ISI)
Queensland University of Technology (QUT)
2 George Street, GPO Box 2434, Brisbane
Queensland-4001, Australia.

Abstract

Since Wang *et al.* announced their results regarding the susceptibility of MD5 (Crypto'04) and SHA-1 (Crypto'05) hash functions to collision attacks, there have been many papers advancing further aspects of these attacks. What has been lacking is an analysis of the legal effect of these attacks upon electronic commerce transactions. As technological advancements are made, the law will need to adjust so as to take account of these attacks so that there does not arise a total undermining of the electronic commerce environment. The legal implications of these attacks need to be understood so that the courts do not over react and thus destroy any confidence commerce currently has in operating in the electronic commerce environment. This paper explores the legal implications of these attacks where certain software applications rely, in part, upon either MD5 or SHA-1.

1 Introduction

By the early 1990's, the National Science Foundation in the USA permitted the Internet to be open to commercial entities for non-research activities to transact business and thus was born the modern electronic commerce environment [7]. This has resulted in a massive uptake by commercial entities in exploiting this new environment. Traditional boundaries have been dissipated and from a legal perspective new and varied conundrums have confronted legislators and regulatory authorities in managing the various policy/legal issues that have arisen out of this new frontier. The electronic commerce environment is like the hydra which has many heads that need to be addressed. One of these heads involves the so called trusted technologies which can be used to effect non-face-to-face transactions. A fundamental underlying technology for non-face-to-face transactions is the use of cryptographic hash functions.

Cryptographic hash functions (hash functions) serve an essential role within a wide range of information security applications. These applications provide certain security services such as data integrity, authentication and to a partial extent non-repudiation [47]. These applications also include non-exhaustively, (a) digital signature generation and verification, (b) session key establishment in key agreement protocols, (c) management of password schemes and (d) commitment schemes in cryptographic protocols such as electronic auctions and electronic voting.

In many applications, it is significantly important from a legal perspective that a hash function should be collision resistant. The effectiveness of the collision resistance of the hash function can in some applications support or undermine the legal position of the application that relies upon hash function technology. As will be discussed later in this paper, this is especially so where authentication is a fundamental element underpinning the legal validity of the transaction.

There are a myriad of hash functions that have been developed over the last 20 years. Of these, MD5 [63] and SHA-1 [5, 43] are the most widely deployed hash functions with many applications. The recent collision attacks on MD5 [39, 40, 46, 66, 71, 75] show that it should no longer be used in any application as a collision resistant function. The collision attacks on the hash function SHA-1 [10, 74, 76] call into question the long-term future use of SHA-1 as a collision resistant function. See [5, 63] for the functionality of MD5 and SHA-1 hash functions. This paper focuses on the legal implications of the collisions associated with MD5 and SHA-1 because this issue has been lacking and further there are some fundamental implications that could substantially undermine the general layman's trust [30] that currently exists in the electronic commerce environment. In a related work, Gauravaram *et al.* [25] have studied the legal and practical implications of hash function collisions only on 128-bit hash functions. The analysis provided in this paper is more extensive than that and takes the collision attacks on SHA-1 and other advances in the cryptanalysis of hash functions into consideration. For the practical implication of hash function collision attacks on various protocols see [35, 68].

The paper is organised as follows: In Section 2, fundamental concepts on hash functions are discussed. In Section 3, an overview on the recent collision attacks on the hash functions MD5 and SHA-1 is given. In Section 4, the impact of these attacks on some important applications with a particular emphasis on the legal and regulatory issues is provided. Section 5 comprises some concluding remarks.

2 Fundamentals of hash functions

Hash functions compress an arbitrary finite length m-bit input message, without the provision of any secret parameter, into a fixed n-bit output value called a message digest or hash. A hash computation on the message x is expressed as $h(x) = y$ where h is a publicly known hash function and the computation of y from x must be easy.

At Crypto'89, Damgård [12] and Merkle [51] independently proposed a similar iterative structure to construct a collision resistant cryptographic hash function. Since then, this iterated design has been called Merkle/Damgård construction which influenced the design of modern hash functions. For more concepts on hash functions see [50, 58].

The principal security properties [50] of cryptographic hash functions are:

1. **Pre-image resistance:** A hash function h is said to be pre-image resistant if for any given message digest y, it is "computationally infeasible" or "hard" to find an input message x. In other words, it should be hard to invert or reverse the hash function from y to get x.

2. 2^{nd} **pre-image resistance:** A hash function h is said to be 2^{nd} pre-image resistant if for a given message x and its corresponding message digest y, it is hard to find another input message $x' \neq x$ such that $h(x) = h(x') = y$

3. **Collision resistance:** A hash function h is said to be collision resistant if it is hard to find any two distinct input messages x and x' such that $h(x) = h(x')$.

See [64] for the formal definitions of above properties. A hash algorithm that satisfies the first two properties mentioned above is said to be a "one-way" hash function (OWHF); whereas a hash algorithm that satisfies all three properties is said to be a "collision resistant" hash function (CRHF) [58]. Notwithstanding the general position concerning the three properties for CRHF, due to some technical reasons, pre-image resistance is not a compulsory requirement for a hash function to be classified as CRHF [50]. The practical security of any hash function lies in its output bit size to resist a successful birthday attack to find collisions [78]. Collisions in hash functions are easier to find than finding pre-images or 2^{nd} pre-images. It requires about $2^{n/2}$ hash function computations to find a collision for an n-bit hash function due to the birthday attack; whereas it requires about 2^n effort to find either pre-images or 2nd pre-images by the brute-force technique [50].

A collision attack is said to be applicable on a hash function if the computational effort required to find collisions in a hash function is significantly less than both the strength conjectured by the designer and that of hash functions of similar parameters with ideal strength. Informally, an attack requiring such a reduced number of operations is said to break the hash function with no bar on the feasibility of such an attack [50].

3 Overview of recent cryptanalysis on MD5 and SHA-1

MD4 [62] is the first hash function which follows the design principles of Merkle/Damgård construction. It was designed by Rivest in 1990. Soon after its design, MD4 was shown to be weak [14] and was replaced by MD5. The MD5 hash function [63] which is also based on Merkle/Damgård structure was designed by Rivest in 1991. While the earlier attacks [19, 22, 23] on the MD4 hash function have shown that it was practically dead as a collision resistant hash function, the earlier collision attacks on MD5 [15, 20] were not that practical in nature. Nevertheless, these attacks on MD5 hinted that in the long run, MD5 is not good enough to be used in applications that require collision resistance property from it [21]. The cryptanalysis of MD4 has been vastly improved in the recent past [38, 52, 71, 72].

The first practical collisions on MD5 were found in 2004 by Wang *et al.* [71]. The detailed description of their collision attack on MD5 was given in 2005 [75]. Their attack on MD5 is a 2-block collision attack using the differential cryptanalysis approach [11] with a complexity of 2^{39} hashing operations of the algorithm. Clearly, this factor is much much smaller than the birthday attack effort of 2^{64} hashing operations of the algorithm for finding collisions on MD5. Their technique involves finding nearly collided values for the first message block and then converting these values to collisions after processing the second message block. This is possible due to the iterative structure of the MD5 hash function.

Since Wang *et al.* have published their results on MD5, several improvements have been made for finding collisions on the algorithm. See [39, 40, 46, 66, 77] for the improved cryptanalysis on the MD5 algorithm. The latest cryptanalysis on MD5 [46,66] shows that collisions can be found on MD5 with a complexity about 2^{33}-2^{34} hashing operations of the algorithm.

SHA-1 [43] is another popular hash function deployed in many applications. It was designed by the National Security Agency (NSA) in 1995 as a replacement of its earlier version SHA-0 [53] and was issued as Federal Information Processing Standard (FIPS PUB 180-1) by the National Institute of Standards and Technology (NIST). SHA-1 is also part of FIPS PUB 180-2 standard [5] which includes SHA-224, SHA-256, SHA-384 and SHA-512 hash algorithms. Like MD5, SHA-1 is also based on the Merkle/Damgård iterative structure.

The reduced versions of SHA-1 were analyzed in [10, 61]. Biham *et al.* [10] have found a 2-block collision on the 40-round SHA-1 with a complexity of 2^{57}. Rijmen and Oswald [61] have estimated theoretically that a collision can be found on the 53-round SHA-1 with a complexity of 2^{71} hashing operations of the algorithm. The first full collision attack on SHA-1 was presented by Wang *et al.* [74]. Wang *et al.* have shown techniques that can be used to find full collisions on 2-block SHA-1 with a complexity of about 2^{69} hashing operations which is less than the 2^{80} theoretical bound of finding collisions in SHA-1. Very recently, Wang *et al.* have improved their attack [76] to a complexity of 2^{63} hashing operations of SHA-1.

Although it is clear that the techniques to find collisions in SHA-1 [74, 76] are viable, Wang *et al.* only estimated the difficulty of an attack, rather than showing any real collision as they had shown on MD4, MD5, RIPEMD, HAVAL and SHA-0 hash functions [71–73, 75]. Notwithstanding this, 2^{63} hashing operations of SHA-1 is within the reach of a distributed computing effort [69]. Further, it is likely that further improvements to this attack will occur in the foreseeable future. A hardware architecture to break the SHA-1 algorithm was proposed by Satoh [67] using Wang's cryptanalytical methods [74]. The estimated $10 million system built with current hardware technology would consist of 303 personal computers with 16 SHA-1 attacking boards with a USB interface each, and each board would have 32 chips. The system consists a total of 9,928,704 SHA-1 macros, and could find a real collision for the full-round SHA-1 in 127 days.

4 Legal affect of collision attacks on MD5 and SHA-1

This section focuses on the legal impact of the above mentioned collision attacks on MD5 and SHA-1 as they are embodied in certain important applications where either of these functions are widely deployed. The applications include digital signatures on messages and digital certificates, software protection and message authentication codes based on hash functions.

4.1 Digital Signatures on messages

Theoretically, digital signatures are used to authenticate the signers of electronic messages. In fact, digital signatures are based upon the substantial underlying assumption that the private key has not been compromised. Digital signatures do not actually establish who affixed the digital signature. They only establish that a particular private key was used to affix the signature. Digital signatures substantially depend upon the document that is being digitally signed, as well as the private key and the algorithms used to affix the digital signature. If the document changes in any way then the digital signature will also substantially change. Consequently, a digital signature is the result of applying the digital signature algorithm to the hash of the document that is being digitally signed, using a private key allegedly held by the person causing the digital signature to be created. Anyone, with the public key that corresponds with the private key used to affix the digital signature, should be able to verify the signature of the signer. The basic premise behind Public Key Technology (PKT) [18] and its ability to provide authentication and restricted non-repudiation services is that the private key remains private and has not been compromised by its disclosure to third parties.

Consider the following scenario between two parties Alice and Bob, wherein Alice wishes to send a digitally signed message to Bob. Assume that the parties use the MD5 hash function as part of the signing technology.

The following steps would be performed by Alice.

1. Alice hashes the message x that she wishes to send to Bob using MD5. MD5(x) is the value of the message digest. Let MD5(x) be h.
2. Alice then transforms h using her private key $kpriv_A$ using some public key algorithm such as RSA to compute the digital signature sig_A. This is expressed as follows.

   $sig_A(x) = E_{RSA}(h, kpriv_A)$
3. Alice sends message x and the signature on x, sig_A, to Bob. Further, Alice informs Bob as to what algorithms were used to generate sig_A [1].

The following steps would be performed by Bob after receiving x and sig_A to verify the digital signature of Alice.

1. Bob hashes the message x that he received from Alice using MD5. MD5(x) is the value of the message digest. Let MD5(x) be h'. Note that this is the same initial step performed by Alice above.

[1] This is actually achieved through an X 509 V. 3 certificate

2. Bob verifies the signature sig_A using the public key of Alice $kpub_A$ to get the message digest h''.

   This is expressed as follows.

   $h'' = D_{RSA}(sig_A, kpub_A)$

3. Bob compares the digests (h' and h'') obtained in steps 1 and 2 and if they are equal then the integrity of the message is established and provided the private key remains private, the person attributed as signing the message is authenticated.

Considering the feasible nature of finding many collisions on MD5 [39, 40, 75], it is now possible for both the signer and the verifier of a digitally signed message that relies upon the MD5 hash algorithm to cheat each other and thus obviate the non-repudiation property that is widely regarded as being an essential property of digital signature technology. This will be shown in the following two scenarios.

Scenario 1: The collision attack on the MD5 algorithm described in Section 3 can easily be used by Alice to cheat Bob. Let the innocuous message x, that Alice wants to sign, be something like:

> I, Alice, am selling my property of Blackacre comprising 5 acres to Bob for a price of AUD \$ 123, 456.

Assume that Alice can come up with an alternate message x' that gives the same digest as x. Let x' be:

> I, Alice, am selling my property of Blackacre comprising 2 acres to Bob for a price of AUD \$ 223, 456.

Note the decrease in the area of Blackacre that is being offered and the increase in the associate price. Let it also be assumed that these two messages have the characteristic that MD5(x) = MD5(x'). Alice then sends x and the signature sig_A computed on MD5(x) - (noting that it would be the same on MD5(x')) to Bob. Bob believes that the message x is the legitimate message but later Alice claims x' as the legitimate message by producing the same signature sig_A on x'. Furthermore, Alice has destroyed all evidence regarding the digital signing of x. Now Alice may have an opportunity to sell Blackacre post signing with Bob to another party "Trent" and she decides that she needs to get out of the first contract of sale with Bob so that she can enter into the more lucrative contract with Trent. Hence, she goes to the elaborate exercise of creating the later message x' that has the same message digest as x.

From an evidentiary perspective this raises some serious doubts as to the probity of the two messages and in particular the fact that the same digital signature can be verified for both messages. At common law the onus of proof is generally placed upon the plaintiff. That is, it is the plaintiff who has the onus of proving his/her case. In this case, if Alice raises a dispute over the contents of the electronic communications that have transpired between them, she will accordingly inform Bob of the problem. If Bob commences proceedings to prove his case in order to obtain specific performance of the contract, he will have the onus of proving the value of the contract; but in doing so he will encounter the defense that Alice has evidence that the contract was different to that which Bob claims. The difficulty for the arbiter of fact, namely the judge in this case, is that the judge will need to decide which message is the correct reflection of the contract. Other evidence will need to be presented by Bob, which could cause substantial expense being placed upon Bob in proving his case. Remembering that Bob, does not have access to Alice's private key; so Bob cannot generate a new digital signature for x'. Alice would argue that she does not know how this occurred. She can also show that her public key can verify the digital signature attached to x' that is in her possession.

Alice could argue that x' is the original message and that Bob has somehow developed x such that MD5(x) = MD5(x'). Further, she may claim that Bob has stripped the original signature from

x' and attached it to x [48]. Bob may have other evidence such as time of receipt of x that has the digital signature of x, but this kind of evidence can be easily spoofed or altered without leaving a trace by the recipient of an electronic communication. Such a case could be decided solely upon the oral evidence of the parties and not upon the technology that underpins the case. Of course, a court in this case could rightly make some substantial disparaging remarks about the technology and its lack of the so called non-repudiation and authentication properties.

Scenario 2: The collision attack on MD5 can also be used by Bob to cheat Alice. This is possible when Alice signs the messages as directed by Bob. Let x and x' be the two messages on which Bob get a collision before getting into contract with Alice. Let x and x' be respectively as follows:

> I, Alice, am selling my property of Blackacre comprising 5 acres to Bob for a price of AUD $\$223,456$.
>
> I, Alice, am selling my property of Blackacre comprising 10 acres to Bob for a price of AUD $\$123,456$.

Once Alice signs the message x at the will of Bob, Bob strips the signature of the message x and attaches it to the message x'. Bob, later claims that the message signed by Alice is x' not x.

In this case, if Bob raises a dispute over the contents of the electronic communications that have transpired between them, he will accordingly inform Alice of the problem. If Alice commences proceedings to prove her case, she will have the onus of proving the value of the contract; but in doing so she will encounter the defense that Bob has evidence that the contract was different to that which Alice claims. Further, Bob does not have access to the private key of Alice as it is known only to her and hence the signature on x' could only have been signed by Alice. That is, since Bob is able to verify the digital signature affixed to x' using Alice's public key and Bob has no access to Alice's private key then Bob will argue that Alice is trying to defraud him of the contract value.

Bob may be able to successfully force Alice to admit that her private key has not been compromised, which means that theoretically she is the only person who had access to the relevant private key and was the only person capable of activating its use to digitally sign x'. Since Bob had no access to Alice's private key, and it is Alice's public key that is used to verify the digital signature affixed to the electronic communication that is being relied upon by Bob, it will be very difficult for Alice to discredit this evidence in the court proceedings, even though the original message x has been substituted by Bob for x'.

The collision for both x and x' is a substantial flaw in the digital signature technology, where the hash algorithm used such as MD5 is not collision resistant. Such a flaw will completely undermine the concept of non-repudiation regarding forged digital signatures. The concept of non-repudiation has been a principal attribute promoted by a number of digital signature technology providers.

In Scenario 2, a court, at a minimum could come to the conclusion that there is some uncertainty regarding the validity of the two messages x and x' and at a maximum the electronic communication in the possession of Alice that has Alice's digital signature affixed, has been altered by Alice, which could give rise to an allegation of giving false evidence under oath or to an allegation of tampering with evidence.

It can be seen that these attacks on MD5 greatly undermine the evidential value of digital signature technology where MD5 is used for digital signature purposes. The collision attacks on MD5 can be used to construct ASCII message sequences which was demonstrated in [13].

There are no legal cases where digital signatures have been specifically disputed, though it is generally accepted that a digital signature will not be non-repudiatable because there are many legal reasons where a party may be able to successfully repudiate a digital signature attributed to them, such as unconscionable conduct or undue influence or duress [47]. What has been generally taken as the base position is that digital signature technology will greatly reduce the incidence of forgeries, but since the successful attacks by Wang *et al.* even the issue of forgeries has now become an issue, which undermines the concept of non-repudiation even further. When the underlying hash function technology is weak, it could result in the compromise of the non-repudiation security property.

A further effect of the undermining of the non-repudiation property is the long term archiving of digitally signed documents. It is not unusual for a dispute to take a substantial amount of time to elapse before it will be heard by a judge. During this intervening time both parties have to ensure that the evidence they have in their possession does not become tainted and maintains its integrity. In Australia, section 11(3) of the Electronic Transactions Act 1999 (Cth) [ETA] [16] provides that an electronic document can be endorsed by a third party for the purposes of integrity. The Uniform Electronic Transactions Act [UETA] [55] that has been adopted by many US States does not have a similar provision.

If the PKT used to affix a digital signature is undermined due to some technological advancement, it is not correct to get the parties to resign the document as this would alter the document in a substantial manner due to the effluxion of time. It may also be impractical for this to occur as one of the parties may not be available or may have even died in the mean time. The better approach is to get a trusted third party to endorse the document by affixing another digital signature which uses a more robust PKT. This approach is akin, though not completely analogous, to what a notary will undertake when endorsing a document. It is not uncommon for a notary to endorse a document by affixing the notary's seal to an original document. The affixation of the notary's signature and seal does not cause the notary to be bound by the contents of the document as a signatory. Instead, the notary is endorsing the document for some special purpose which must fit within the notary's powers of authenticating documents presented. Therefore, even though the notary may sign a document, this process does not make the notary bound to the document in the same way as the signatory is bound to the contents of the document. This position also arises when a witness affixes his/her signature to a document. The difference between the paper based environment and the digital environment is that in the paper based environment it is very difficult to alter a signature or any other aspects of the document without some visible trace after the signature has been affixed to a document, especially if the signature has been affixed using some indelible ink as opposed to some pencil markings.

An important aspect of the digital endorsement mechanism is that it can be undertaken multiple times. The original document with its original digital signature or signatures is maintained and thus the integrity of the document is preserved. From an evidential perspective, the long term preservation of the integrity of the document, which must include the digital signatures embodied or associated with the digital document is paramount. If there are any questions that adversely call into question the integrity of the digital document then a Court having jurisdiction could decide to not admit the document into evidence or if admitted into evidence, gives the document such a low weighting due to its unreliability that its evidential value is useless for the presenting party. The trusted third party will need to be noted as an endorser so as not to be confused as an original signatory. When the case is finally heard by a Judge, each endorsement of a digital signature will be verified until such time as the original signatures can be verified. For long term archiving purposes this approach is recommended. The UETA that has been adopted by a substantial number of US States does not specifically recognise the endorsement mechanism as provided in the Australian Electronic Transactions Act. However, UETA does make provision for security procedures which are to be used to maintain the integrity of digitally signed documents and therefore the procedure could be similar as noted above for endorsement purposes.

The issue of obviating the non-repudiation property has an even more catastrophic affect when digital certificates are the documents that are being digitally signed. One of the difficulties in using PKT is the deployment of the corresponding public keys that are used to verify digitally signed documents. It should be remembered that it is the public key that corresponds to the private key that is used to verify the digital signature. To distribute public keys, digital certificates were proposed [41] and developed, which are currently based upon the ISO standard X.509 v. 3. [36].

4.2 Digital signatures on digital certificates

A digital certificate is a structure used to convey information about a user for identification purposes [29]. It is issued by a certification authority (CA) and attempts to bind an identity to the public key and ip so facto the possessor of the corresponding private key. An entity in need of a digital certificate presents to the CA its identification details that evidences their identity, the public key and other required information. It contains among other things the name, a unique serial number for the certificate, expiration date, a copy of the certificate holder's public key (the holder should solely have the corresponding private key), the public key algorithm and the hash algorithm used and the digital signature of the CA so that a recipient can verify that the certificate is authentic. The most widely used digital certificate standard is X.509 version 3. Hence, the implications of MD5 collisions on X.509 version 3 digital certificate will be discussed in this section.

The X.509 version 3 digital certificate issued to an end user by a CA contains the following fields in order: Version number (v3), Unique serial number, Name of the signature algorithm (for example, MD5 with RSA), the name of the issuer (for example, secure server CA, Verisign CA), Validity period of the certificate, Subject of the certificate which is the end user, Public key of the subject, Optional set of extensions, Signature algorithm and Signature value.

The impact of MD5 collisions on the X.509 version 3 digital certificates depends on the order of the fields.

Case 1: Attack on a new certificate request

The concept of this attack is simple. The attacker prepares in advance a genuine certificate request whose hash value collides with that of a fraudulent certificate (which may have for example, a different name). The CA's signature on the genuine certificate is transferred to the fraudulent one. To generate the colliding request data, the attacker proceeds as follows.

Given any fixed prefix message p and two different desired messages $m1$ and $m2$, create two suffix messages $s1$ and $s2$ that provide a collision. That is concatenating the prefix message p with $m1$ and then concatenating the resultant with $s1$. In like manner, p,$m2$ and $s2$ are also concatenated. This is visualised as $h(p||m1||s1) = h(p||m2||s2)$. In the case of digital certificates, the serial number and validity period are not fixed. Serial numbers are generally harder to predict than the validity period. For example, Verisign the most widely used CA sets the validity period in the certificate based on the current time. This can be predicted to within a few minutes.

A malicious attacker can generate two certificate requests with two different subjects as follows. "Digital certificate request for **www.centralpark.com** and here are my personal details" ($m1$) and "Digital certificate request for **www.centralbank.com** and here are my personal details"($m2$). These two are the desired messages $m1$ and $m2$ of the attacker. Following the above order of field selection in the digital certificate, version number, name of the signature algorithm, name of the issuer, validity period (under the assumption that it can be easily predicted) are the fixed prefix fields. Public key field containing a public key (there would be corresponding private key to each public key) that comes after the subject field forms the suffix.

Once the attacker crafts the field patterns for the different name fields containing two distinct messages in advance, the attacker sends the initial request to get the digital certificate signed by the CA and later inserts the signature on to the fake second certificate thereby making a perfect forgery. Any browser can easily trust the **www.centralbank.com**'s digital certificate as the genuine certificate and masquerades as a bank thus sneaking the personal details of a customer. This attack follows the same principle as the one described in Section 2.1 where the non-repudiation property of security is violated. This type of attack could be used to great effect in the incidence of the email "phishing" scam that has become so prevalent in recent times.

The mathematical equation describing this attack on digital certificate is

$$\mathrm{h(v3||serialnumber1||p||m1||s1) = h(v3||serialnumber2||p||m2||s2)} \tag{1}$$

The practical possibility of this attack is still in doubt and further research is required. The reason is that when the CA signs a certificate, CA specifies a unique serial number in the certificate. It depends on how much control the attacker has on the serial number field. X.509 version 3 certificates use a serial number of length 128 bits. The probability of predicting a correct serial number is 2^{-128}. The collision attacks demonstrated on MD5 [39, 40, 46, 66, 71, 75] have 1024-bit freedom for choosing the message patterns. But in this scenario the degree of freedom is really short which is 128 bits.

A more devastating attack would be one in which an attacker can obtain a legitimate server certificate with a collision to the one containing a wild card[2] for the domain name and an expiration date far in the future [4]. It appears that the use of unpredictable serial numbers may prevent such attacks.

Lenstra *et al.* [44, 45] have shown a method to construct a pair of valid X.509 certificates in which the "to be signed" parts form a collision when MD5 is used in the X.509 digital certificates. The attack relies on the ability of the attacker to create two different public keys that would cause the body of the certificate to have collisions when hashes with MD5. For the real possibility of this attack, the attacker has to predict the contents and structure of the certificate before it is issued, including the identity, the serial number and the start and finish dates of the validity period of the certificate. This attack demonstrates that a relying party using a public key certificate based on MD5 can not be certain that the alleged owner actually possesses the corresponding private key. Since the identity in the two certificates is the same, there are probably no real-world scenarios where this kind of attack would get the attacker any advantage.

It was suggested in [35] that if a trusted third party who issues the digital certificates wants to avoid the above kind of attack, they can prevent the attack by making other signed parts of the certifcate random enough (for example, by adding a randomly chosen component to the identity) to eliminate any advantage gained the attack.

Daum and Lucks [13] have constructed a pair of postscript documents that look different but hash to the same digest using the random collisions in MD5 [75] and exploiting the iterative structure of the Merkle/Damgård construction. Postscript permits this attack to work by binding two different documents in the same file revealing only one document for signing purposes and at the same time hiding the other. In a practical sense, this attack would allow an intern at a company to create a file where one document is simply a reference that the employer would digitally sign while the other hidden document could contain unauthorized permissions such as the viewing of the employer's private data [9]. Considering the practical significance of these attacks, it is not recommended that MD5 continue to be used when collision-resistance property of MD5 is required by the application.

It is expected that CAs will move away from using MD5 for signing new digital certificates due to the threat of the described attack.

Case 2: Attack on an existing certificate

At this point of time, the attacks on MD5 and SHA-1 cannot be used to tamper with existing certificates, for example Secure Socket Layer (SSL) web-server certificates, as it needs an attack violating the 2^{nd} pre-image resistance property of these functions. This is important, as all Banking sites utilise the SSL protocol. If there is developed a feasible attack that could tamper with existing SSL server certificates then the trust currently reposed in banking sites would be substantially undermined and this could be a catastrophic effect upon the economy.

So far, the best known attack to violate this property is the brute-force attack which requires 2^{128} hash computations for the MD5 algorithm and 2^{160} hash computations for the SHA-1 algorithm respectively. Obviously, performing these tasks is computationally infeasible. In this case, the attacker cannot find a collision to any arbitrary existing digital certificate. The current attacks on MD5 do not violate the 2^{nd} pre-image resistance property [33].

Bellovin and Rescorla [68] have observed that the collision attacks on MD5 and SHA-1 cannot be translated into demonstrable attacks on real-world certificate-based protocols such as S/MIME, TLS

[2]A wild card is a character which can be substituted for another character

and IPsec. Their results show that the transition to stronger hash functions is necessary, though not immediate. Moreover, their analysis reveals that major internet protocols such as S/MIME and TLS were not designed properly for such a transition. In particular, if the signature algorithm is linked to a particular hash function, as Digital Signature Algorithm (DSA) is tied to SHA-1, both the signature algorithm and the hash algorithm need to be changed. Hoffman and Schneier [35] also in an independent work have indentified that most protocols use hash functions in a way that makes them immune to harm from collision attacks.

As stated, Lenstra *et al.* [44,45] have shown that a relying party will not have sufficient evidence to satisfy itself as to the named subscriber in the X.509 v.3 certificate as actually possessing the private key corresponding to the public key noted in the certificate. The effect of the collision attacks on MD5 substantially undermine the value of any certificate where MD5 hash algorithm has been used. From a legal perspective, these attacks bring into the question the commercial value of PKI and the technical/legal value of any certificates that utilize the MD5 hash function. So far no practical attacks on the certificates based on SHA-1 have been reported and hence the legal status of certificates based on SHA-1 is maintained.

4.3 Message authentication

Hash functions are used in constructing message authentication codes (MACs) as in the construction of HMAC [8]. A MAC function is said to be forged when an attacker finds the MAC value for a previously unseen message without knowing the secret key, which is one of the inputs to the MAC function. HMAC construction is provably secure under certain assumptions on the security of the underlying hash function. An attacker that tries to forge the MAC function HMAC would also be able to break the underlying hash function in one of the following two ways:

1. The attacker would be able to find collisions to the underlying hash function keyed through the IV of the hash function.

2. The attacker would be able to find an output of the external hash function which is keyed with a random and secret initial value.

The Transport Layer Security (TLS) protocol which is defined as a proposed Internet standard version for SSL version 3 in [17] uses HMAC algorithm defined in [8,42] as a MAC function and also as a Pseudo Random Function (PRF). HMACs are protected by a shared secret key, and PRFs do not require collision resistance as their security property [32]. The HMAC function used in the TLS protocol is instantiated either with MD5 or SHA-1 hash functions.

The collision attacks on MD5 and SHA-1 would not enable an attacker either to find collisions starting from random secret initial values or to produce known outputs from random secret initial values. Hence, the ability to find collisions in the hash functions MD5 and SHA-1 do not lead to forgery attacks on the HMAC function, and thus the legal status of the TLS structures is maintained. This is highly relevant as TLS is in many cases used as the underlying security technology for Internet banking. We also note that the collision attacks on MD5 and SHA-1 would not enable an attacker to forge the other variants of HMAC [26, 56].

4.4 Software protection

Software vendors want to protect the integrity of the software they sell to the users [58]. Hash functions are used to check the integrity of the software that the user receives from the vendor and make sure that user would not receive virus or malicious programs that infect the user's PC. A user can get the software or any application from the vendor through the CDs/DVDs, as downloads from the vendor's website or from vendor's mirror website or through some other means. Sometimes users may also download the software or files for free from the Internet and expect them to be free from viruses or malicious content.

There are so many free tools available on the Internet today [3] that use MD5 algorithm for data integrity control and verification of network file transfer, E-mail messages and files or software downloaded from the Internet. For example, the MD5 algorithm is used to check the integrity of a Cisco Internetworking Operating System(IOS) software image [1]. The MD5 file validation feature of the Cisco IOS uses MD5 to create a 128-bit checksum of the Cisco IOS software image on some of the Cisco released products and compares that with the MD5 checksum of the images on those releases posted on the Cisco website. Apache web-server [2], the most popular web-server on the Internet, develops and maintains an open-source Hyper Text Transfer Protocol (HTTP) server for the Unix and Windows NT operating systems. It uses MD5 hashes as one of the options, the other being the Pretty Good Privacy (PGP) signatures, to ensure the verifiability of the downloads from its home page and other mirror sites [6]. It uses appropriate MD5 embedded programs on the Unix and Windows distribution sites to achieve this. The Solaris Fingerprint Database (sfpDB), a free Sun Microsystems security tool, uses MD5 to verify the integrity of the files distributed with the Solaris Operating Environment [70]. The sfpDB ensures that its official binary distributions contain authentic files but not adapted ones that compromise the system security. The sfpDB uses MD5 to compare the digest of the binary distributions with the trusted hashes stored on its homepage and hence identify any mismatches if present.

In the wake of attacks on MD5, there would not be any threat to the above existing applications where MD5 is used as a CRHF to achieve data integrity [49]. Using the current collision attack techniques on MD5, it is impossible for a malicious attacker to generate the same digest on a new software with a back door as already exists for a certified software and then replacing the later with the former [59]. This attack requires violation of 2^{nd} pre-image resistance property of the hash function. It is still unsure whether the attack strategy on MD5 violates this property when there is a larger degree of freedom for the attacker to manipulate message formats as in this example. This case is quite different from the case of attacking existing digital certificates. In digital certificates, the degree of freedom available to the attacker is quite limited unlike here. The analysis on MD5 collisions given by Hawkes *et al.* [33] shows that MD5 is still 2^{nd} pre-image resistant and the attacks by Wang *et al.* [71,75] would not compromise this property of MD5. Hence we can say that at present the collision attacks do not allow tampering with arbitrary programs available on the Internet.

If a successful attack is developed then a malicious attacker can masquerade as a genuine vendor and develop collisions for the genuine code and malicious code using the attack technique on the MD5 algorithm. The attacker could place the malicious program (for example code with virus) and the corresponding hash code on the Internet. A similar attack is possible where an attacker uses the collision attack technique on MD5 and can enable a trusted compiler/verifier to accept and sign the innocuous program, which could then be substituted for the malicious one [4]. Hence, when the hash function used for the integrity checking is not robust, the end user cannot identify the infected code from the true code. The infected code could contain a deliberate security vulnerability. It is not out of the question for a malicious attacker to implement this attack on software patch mechanisms that are frequently released.

This then raises the duty of care that software manufacturers have to their end users in distributing software that could possibly be altered with out authority by third parties so as to insert a virus/trojan horse or some back-door code that can later be used by such third parties for nefarious activity [24]. Due to the substantial publication of the attacks on MD5 in the press both generally and computer specific it is unlikely that a software manufacturer could successfully argue that it was not aware of these attacks and their ramifications [37]. The legal status of software manufacturers would it is submitted be no different to other manufacturers where such manufacturers have to take reasonable care to ensure that their products do not cause damage to their constituent end users [24]. This duty of care issue has recently arisen in the USA in the Sony-BMG matter [65]. SONY-BMG was identified as having distributed a substantial number of music CDs that contained an alleged technological protection measure (TPM). The TPM was a root kit mechanism that included the property of hiding itself within the kernel. Further, it was not possible to easily remove the root kit from an infected system and if removed could adversely affect the operations of the infected computer system. Finally it has been identified that the SONY-BMG root kit created a security vulnerability for window users which can be exploited by third parties [65]. Further, in support of this duty of care requirement is the recent determination in Australia by the administrative Appeals Tribunal that non customised software (COTS) is goods for the purposes of the Export Markets Development grants Act [60]. The ratio of this decision is highly persuasive and it is likely that it would be accepted by a court having jurisdiction. The effect of this decision is that if software is legally determined to be goods then software manufacturers will now be within the scope of the product liability provisions of the Trade Practices Act 1974 (CTH). In order to better protect the distribution of such software, software manufacturer's should ensure that their software is digitally signed using an application that utilised an appropriately secure hash function. Thus based upon the effects of the recent attacks mentioned in this paper, MD5 should not be relied upon. This recommendation does not currently apply to SHA-1 as the attacks on SHA-1 are of limited value, though it is possible that future developments of these current attacks could undermine the security of SHA-1 when utilised in the digital signing of software applications. In fact, NIST has already recommended that stronger hash functions be used for digital signing purposes and other purposes (such as SHA-224, SHA-256, SHA-384, SHA-512) and the phasing out of SHA-1 by 2010 [54].

In light of the attacks to MD5, it is recommended that MD5 not be used as a CRHF to achieve data integrity. The continued use of MD5 for software protection measures can no longer be assured of security and therefore should be immediately be stopped and a more secure hash mechanism as identified above should be used instead. From a legal perspective, any software manufacturers who do not so change are potentially exposing themselves to an unnecessary risk.

5 Conclusion

The legal implications arising out of the successful identification of the cryptographic hash function attacks need to be properly understood so that the law does not over react to such attacks and thus undermine the economic benefits that commerce has taken advantage of. Notwithstanding this, it is clear that electronic commerce currently relies upon many software applications that in turn are dependent upon secure hash functions. Even if the NIST recommended hash functions [5] are used which necessiates the immediate phasing out of MD5 and the gradual phasing out of SHA-1 by 2010, there is no guarantee that other yet to be discovered attacks are developed on the strong hash functions like SHA-256 [3]. This then brings commerce back to the issue of whether the use of such technology will always result in a sword of damocles situation which is a risk that is ever present.

Table 1 summarises the technical implications of the collision attacks on MD5. As noted above, the legal implications are affected by how MD5 is used. For example, where a digital signature application utilises MD5 then the authentication and data integrity properties could be greatly undermined from the evidentiary perspective. This position is also applicable where the integrity and authentication of software is important.

Finally, it is recommended to move away from using hash functions based on the Merkle/Damgård design (as was anticipated [27] even before these attacks have actually happened) to new hash function designs [27, 28] keeping in mind the long term security of various applications that deploy hash functions.

[3]It should be noted that the analysis of SHA-256 has already started [31, 34, 57]

Table 1: Summary of impact of collisions in MD5 on some applications

Application	Attack possibility	Property violated
Message Signatures	Yes	Non-repudiation Integrity
Existing certificate signatures	No	
New certificate signatures	Yes	Authentication Non-repudiation
HMAC applications	No	
Tampering existing software	No	
Tampering new software	Yes	Non-repudiation Authentication Integrity

References

[1] MD5 File Validation. A document on MD5 File Validaion available on Cisco Website, 2002. http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/ Last access date: 17 September,2004.

[2] Apache HTTP Server Project. This is the homepage of Apache HTTP server, 2004. http://httpd.apache.org/ Last access date: 17 September,2004.

[3] Data Invariability and Integrity Control. Internet article on a tool which MD5 for integrity purposes, 2004. http://www.fastsum.com/ Last access date: 16 September, 2004.

[4] Hash collision question and answer. Crypto News on Attacks on Hash Functions, 2004. http://www.cryptography.com/cnews/hash.html/ Last access date: 13 September, 2004.

[5] National Institute of Standards and Technology (NIST) , Computer Systems Laboratory. Secure Hash Standard. Federal Information Processing Standards Publication (FIPS PUB) 180-2, August 2002.

[6] Apache HTTP Server Source Code Distributions. This download page of source code is on the Apache website, 2004. http://www.apache.org/dist/httpd/ Last access date: 17 September,2004.

[7] Barry M. Leiner and Vinton G. Cerf and David D. Clark and Robert E. Kahn and Leonard Kleinrock and Daniel C. Lynch and Jon Postel and Larry G. Roberts and Stephen Wolff. A Brief History of the Internet. `http://www.isoc.org/internet/history/brief.shtmlLastaccessdate:9January,2006`.

[8] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *Advances in Cryptology—CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag, 18–22 August 1996. Full version of the paper is available at `"http://www-cse.ucsd.edu/users/mihir/papers/hmac.html"`.

[9] Celeste Biever. Hashing exploit threatens digital security. An article in the New Scientist Magazine, 2005. The note is available at `http://www.newscientist.com/article.ns?id=dn7519`. Last access date: 13 January 2006.

[10] Eli Biham, Rafi Chen, Antonie Joux, Patrick Carribault, Christophe Lemuet, and William Jalby. Collisions of SHA-0 and reduced SHA-1. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 36–57. Springer, 2005.

[11] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems (extended abstract). In A. J. Menezes and S. A. Vanstone, editors, *Advances in Cryptology—CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 2–21. Springer-Verlag, 1991, 11–15 August 1990.

[12] I.B. Damgard. A design principle for hash functions. *Lecture Notes in Computer Science*, 435:416–427, 1990.

[13] Magnus Daum and Stefan Lucks. Hash Collisions (The Poisoned Message Attack) "The Story of Alice and her Boss". Presented at the Rump Session of EUROCRYPT 2005, May 2005. The result is available at `http://th.informatik.uni-mannheim.de/people/lucks/HashCollisions/`. Last access date: 10 November 2005.

[14] Bert den Boer and Antoon Bosselaers. An attack on the last two rounds of MD4. In J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 194–203. Springer-Verlag, 1991.

[15] Bert den Boer and Antoon Bosselaers. Collisions for the compression function of MD5. In *Advances in Cryptology – EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 293–304. Springer-Verlag, 1994.

[16] Australian Government Attorney-General's Department. Electronic Transactions Act (1999). The pdf version of the document is available at `http://scaleplus.law.gov.au/html/comact/10/6074/pdf/162of99.pdf` Last access date: 13 January, 2006.

[17] Tim Dierks and Christopher Allen. The TLS protocol version 1.0. Internet Request for Comment RFC 2246, Internet Engineering Task Force, January 1999. Proposed Standard.

[18] Whitfield Diffie. The first ten years of public-key cryptography. *Proceedings of the IEEE*, 76:560–576, 1988.

[19] Hans Dobbertin. Cryptanalysis of MD4. In Dieter Grollman, editor, *Fast Software Encryption: Third International Workshop*, volume 1039 of *Lecture Notes in Computer Science*, pages 53–69, Cambridge, UK, 21–23 February 1996. Springer-Verlag.

[20] Hans Dobbertin. Cryptanalysis of MD5 compress, May 27 1996. Presented at the Rump Session of Eurocrypt'96.

[21] Hans Dobbertin. The status of MD5 after a recent attack. *CryptoBytes*, 2(2):1, 3–6, Summer 1996.

[22] Hans Dobbertin. Cryptanalysis of MD4. *Journal of Cryptology*, 11(4):253–271, 1998.

[23] Hans Dobbertin. The first two rounds of MD4 are not one-way (extended abstract). In Serge Vaudenay, editor, *Fast Software Encryption: 5th International Workshop*, volume 1372 of *Lecture Notes in Computer Science*, pages 284–292, Paris, France, 23–25 March 1998. Springer-Verlag.

[24] [1932] All ER In The locus classicus case for the modern day action for negligence in the UK Donoghue v. Stevenson (nee Macalister), [1932] AC 532 and 162 N.E. 99 (1928) in the US the locus classicus case is Palsgraf v. Long Island Railroad Co. 248 N.Y. 339.

[25] Praveen Gauravaram and Adrian McCullagh and Ed Dawson. The legal and practical implication of recent attacks on 128-bit hash functions. First Monday Journal, Volume 11. Number 1-2, January 2006. `http://www.firstmonday.org/issues/issue11_1/gauravaram/index.html`.

[26] Praveen Gauravaram, William Millan, Juanma Gonzalez Nieto, and Edward Dawson. 3C- A Provably Secure Pseudorandom Function and Message Authentication Code. A New mode of operation for Cryptographic Hash Function. Cryptology ePrint Archive, Report 2005/390, 2005. `http://eprint.iacr.org/`.

[27] Praveen Gauravaram, William Millan, and Lauren May. CRUSH: A New Cryptographic Hash Function using Iterated Halving Technique. International workshop on cryptographic algorithms and uses . pages 28-39, 2004.

[28] Praveen Gauravaram, William Millan, Ed Dawson and Kapali Viswanathan. Constructing secure hash functions by enhancing Merkle-Damgård Construction. Cryptology ePrint Archive, Report 2006/061, 2006. `http://eprint.iacr.org/`.

[29] E. Gerck. Overview of certification systems: X.509, CA, PGP and SKIP, August 25 1999. `http://citeseer.ist.psu.edu/252354.html`. Last access date: 15 October 2005.

[30] Anthony Giddens. The consequences of modernity. Stanford University Press, 1990.

[31] Henri Gilbert and Helena Handschuh. Security Analysis of SHA-256 and Sisters. *Lecture Notes in Computer Science*, 3006:175–193, 2004.

[32] Peter Gutmann, David Naccache, and Charles Palmer. When Hashes Collide. *Security & Privacy Magazine, IEEE*, 3(3):68–71, May-June 2005.

[33] Philip Hawkes, Michael Paddon, and Gregory G. Rose. Musings on the wang et al. MD5 collision. Cryptology ePrint Archive, Report 2004/264, 2004. `http://eprint.iacr.org/`.

[34] Hirotaka Yoshida and Alex Biryukov. Analysis of a SHA-256 variant. Appeared in SAC'2005.

[35] Paul Hoffman and Bruce Schneier. Attacks on Cryptographic Hashes in Internet Protocols (RFC 4270). A memo providing information to the Internet Community. `http://www.rfc-archive.org/getrfc.php?rfc=4270`.

[36] R. Housley, W. Ford, W. Polk, and D. Solo. RFC 2459: Internet X.509 public key infrastructure certificate and CRL profile, January 1999. Status: PROPOSED STANDARD.

[37] In Re Verizone, Maine Public Utilities Commission. Docket No. 2000-849.

[38] Jongsung Kim, Alex Biryukov, Bart Preneel, and Sangjin Lee. On the Security of Encryption Modes of MD4, MD5 and HAVAL. Cryptology ePrint Archive, Report 2005/327, 2005. `http://eprint.iacr.org/`.

[39] Vlastimil Klima. Finding MD5 collisions A toy for a notebook. Cryptology ePrint Archive, Report 2005/075, 2005. `http://eprint.iacr.org/`.

[40] Vlastimil Klima. Finding MD5 Collisions on a Notebook PC using Multi-message Modifications. Cryptology ePrint Archive, Report 2005/102, April 2005. `http://eprint.iacr.org/`.

[41] Loren M. Kohnfelder. Towards a practical public-key cryptosystem. B.S. Thesis, supervised by L. Adleman, May 1978.

[42] Hugo Krawczyk, Mihir Bellare, and Ran Canetti. RFC 2104: HMAC: Keyed-hashing for message authentication, 1997. Status: INFORMATIONAL.

[43] National Institute of Standards and Technology (NIST) Computer Systems Laboratory. Secure hash standard. Federal Information Processing Standards Publication (FIPS PUB) 180-1, April 1995.

[44] Arjen Lenstra and Benne de Weger. On the possibility of constructing meaningful hash collisions for public keys. In Colin Boyd and Juan M. Gonzalez Nieto, editors, *Information Security and Privacy*, volume 3574 of *Lecture Notes in Computer Science*, pages 267–279. Springer, 4–6 2005. Complete paper is available at "http://www.win.tue.nl/~bdeweger/CollidingCertificates/".

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

[46] Jie Liang and Xuejia Lai. Improved collision attack on hash function md5. Cryptology ePrint Archive, Report 2005/425, 2005. http://eprint.iacr.org/.

[47] Adrian McCullagh and William Caelli. Non-Repudiation in the Digital Environment. *First Monday–Peer-Reviewed Journal On the Internet*, 5, August 2000. This article is available at http://www.firstmonday.dk/issues/issue5_8/mccullagh/ Last access date: 5 January 2006.

[48] Adrian McCullagh, William Caelli, and Peter Little. Signature stripping: A digital dilemma. *Journal of Information, Law and Technology*, 1. http://www2.warwick.ac.uk/fac/soc/law/elj/jilt/2001_1/mccullagh/Last access date: 9 January, 2006.

[49] Declan McCullagh. Crypto Researchers Abuzz over Flaws. This report was published at the CNET News, 2004. http://news.com.com/Crypto5313655.html/ Last access date: 17 September,2004.

[50] Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*, chapter Hash Functions and Data Integrity, pages 321–383. The CRC Press series on discrete mathematics and its applications. CRC Press, 1997.

[51] Ralph C. Merkle. One way hash functions and DES. *Lecture Notes in Computer Science*, 435:428–446, 1990.

[52] Yusuke Naito, Yu Sasaki, Noboru Kunihiro, and Kazuo Ohta. Improved collision attack on MD4. Cryptology ePrint Archive, Report 2005/151, 2005. http://eprint.iacr.org/.

[53] National Institute of Standards and Technology (NIST). Secure Hash Standard. Federal Information Processing Standards Publication (FIPS-180), May 1993.

[54] National Institute of Standards and Technology (NIST). NIST Brief Comments on Recent Cryptanalytic Attacks on SHA-1, February 2005. This short notice by NIST is available at http://csrc.nist.gov/news-highlights/NIST-Brief-Comments-on-SHA1-attack.pdf. Last access date: 5 January 2006.

[55] National Conference of Commissioners On Uniform State Laws. Uniform Electronic Transactions Act (1999), 1999. The pdf version of the document is available at http://www.law.upenn.edu/bll/ulc/fnact99/1990s/ueta99.pdf Last access date: 13 January, 2006.

[56] Sarvar Patel. An Efficient MAC for Short Messages. Cryptology ePrint Archive, Report 2001/097, 2001. http://eprint.iacr.org/.

[57] Norbert Pramstallar, Christian Rechberger, and Vincent Rijmen. Preliminary analysis of the SHA-256 Message Expansion. Technical report, National Institute of Standards and Technology NIST, October 2005. This paper is available at http://www.csrc.nist.gov/pki/HashWorkshop/program.htm. Last access date: 6 December 2005.

[58] Bart Preneel. *Analysis and design of Cryptographic Hash Functions.* PhD thesis, Katholieke Universiteit Leuven, 1993. Electronic copy of the thesis is available at http://homes.esat.kuleuven.be/~preneel/phd_preneel_feb1993.pdf. Last access date: 14 January 2006.

[59] James Randall and Michael Szydlo. Collisions for SHA0, MD5, HAVAL, MD4, and RIPEMD, but SHA1 Still Secure. A technical note on the RSA website, August 2004. The note is available at http://www.rsasecurity.com/rsalabs/node.asp?id=2738. Last access date: 13 January 2006.

[60] Re Amlink Technologies Pty Ltd and Australian Trade Commission [2005] AATA 359.

[61] Vincent Rijmen and Elisabeth Oswald. Update on SHA-1. In Alfred J. Menezes, editor, *CTRSA: The Cryptographers' Track at RSA Conference*, volume 3376 of *Lecture Notes in Computer Science*, pages 58–71. Springer-Verlag, 2005.

[62] Ronald Rivest. The MD4 message digest algorithm. In *Advances in Cryptology – CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 303–311, 1991.

[63] Ronald Rivest. The MD5 Message-Digest Algorithm. RFC 1321, MIT, RSA Data Security, April 1992.

[64] Phillip Rogaway and Thomas Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In Bimal K. Roy and Willi Meier, editors, *Fast Software Encryption (FSE)*, volume 3017 of *Lecture Notes in Computer Science*, pages 371–388. Springer-Verlag, 2004.

[65] M Russinovich. Marks sysinternals blog. See http://www.sysinternals.com/blog Last access date: 9 November, 2005.

[66] Yu Sasaki, Yusuke Naito, Noboru Kunihiro, and Kazuo Ohta. Improved collision attack on md5. Cryptology ePrint Archive, Report 2005/400, 2005. http://eprint.iacr.org/.

[67] Akashi Satoh. Hardware Architecture and Cost Estimates for Breaking SHA-1. In Chi-Ming Hu and Wen-Guey Tzeng, editors, *ISC*, volume 3650 of *Lecture Notes in Computer Science*, pages 259–273. Springer, 2005.

[68] Steven Bellovin and Eric Rescorla. Deploying a New Hash Algorithm. Technical report, National Institute of Standards and Technology NIST, October 2005. This paper is available at `http://www.csrc.nist.gov/pki/HashWorkshop/program.htm`. Last access date: 5 January 2006.

[69] Michael Szydlo. SHA-1 collisions can be found in 2^{63} operations. RSA Laboratories Technical report, 2005. `http://www.rsasecurity.com/rsalabs/node.asp?id=2927`. Last access date: 14 January 2006.

[70] The Solaris Fingerprint Database- A Security Tool for Solaris Operating Environment Files. This pdf document is published at the Sun Microsystem's BluePrints section, May 2001. http://www.sun.com/blueprints/0501/Fingerprint.pdf Last access date: 17 September,2004.

[71] Xiaoyun Wang, Dengguo Feng, Xuejia Lai, and Hongbo Yu. Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD. Cryptology ePrint Archive, Report 2004/199, 2004. `http://eprint.iacr.org/`.

[72] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the Hash Functions MD4 and RIPEMD. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2005.

[73] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Efficient collision search attacks on SHA-0. In Victor Shoup, editor, *Advances in Cryptology—CRYPTO '05*, volume 3621 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2005, 14–18 August 2005.

[74] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full SHA-1. In Victor Shoup, editor, *Advances in Cryptology—CRYPTO '05*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer, 2005, 14–18 August 2005.

[75] Xiaoyun Wang and Hongbo Yu. How to Break MD5 and Other Hash Functions. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2005.

[76] Xiaoyun Wang and Andrew Yao and Frances Yao. Cryptanalysis of SHA-1 hash function. Technical report, National Institute of Standards and Technology NIST, October 2005. The slides of this work are available at `http://www.csrc.nist.gov/pki/HashWorkshop/program.htm`. Last access date: 14 January 2006.

[77] Jun Yajima and Takeshi Shimoyama. Wang's sufficient conditions of MD5 are not sufficient. Cryptology ePrint Archive, Report 2005/263, 2005. `http://eprint.iacr.org/`.

[78] Gideon Yuval. How to swindle Rabin. *Cryptologia*, 3(3):187–189, July 1979.