

# Cryptography and its hash function's security

**Amandeep Kaur**  
U.I.E.T  
Panjab University, Chandigarh  
[amanruby\\_deep7@yahoo.com](mailto:amanruby_deep7@yahoo.com)

**Amandeep Verma**  
Lecturer, U.I.E.T  
Panjab University, Chandigarh  
[verma\\_aman81@yahoo.com](mailto:verma_aman81@yahoo.com)

**Abstract-** Cryptography is commonly used for securing communications. It uses all kinds of security mechanisms and related techniques. It referred almost exclusively to encryption and decryption methods. One of the most important classes of cryptographic algorithms in use is the class of cryptographic hash functions. There are series of information about hash functions like MD5 and SHA are cracked. This paper will discuss in detail the broken MD5 and SHA-1 and describe that how cryptanalysis attack these hash functions. This paper also shows the directions for the development of the theory of hash functions and how can we overcome from this serious problem.

## I. INTRODUCTION

Cryptography referred almost exclusively to encryption, the process of converting ordinary information (plaintext) into unintelligible gibberish (ie, cipher text). Decryption is the reverse, moving from unintelligible cipher text to plaintext. A cipher (or cypher) is a pair of algorithms, which perform this encryption and the reversing decryption. The detailed operation of a cipher is controlled both by the algorithm and, in each instance, by a key. This is a secret parameter (ideally, known only to the communicants) for a specific message exchange context. Keys are important, as ciphers without variable keys are trivially breakable and therefore less than useful for most purposes. Historically, ciphers were often used directly for encryption or decryption, without additional procedures such as authentication or integrity checks. Cryptography can be defined as the study of mathematical techniques related to the security of transmission and storage of information.[6]

Cryptography was concerned solely with message confidentiality (i.e., encryption) — conversion of messages from a comprehensible form into an incomprehensible one, and back again at the other end, rendering it unreadable by interceptors or eavesdroppers without secret knowledge (namely, the key needed for decryption of that message). In recent decades, the field has expanded beyond confidentiality concerns to include techniques for message integrity checking, sender/receiver identity authentication, digital signatures, interactive proofs, and secure computation, amongst others.

The main classical cipher types are transposition ciphers, which rearrange the order of letters in a message (e.g. 'help me' becomes 'ehpl em' in a trivially simple rearrangement scheme), and substitution ciphers, which systematically replace letters or groups of letters with other letters or groups of letters (e.g., 'fly at once' becomes 'gmz bu podf' by replacing each letter with the one following it in the alphabet). Cryptographic algorithm and system designers must also sensibly consider probable future developments in their designs. For instance, the continued improvements in computer processing power have increased the

scope of brute-force attacks when specifying key lengths. Cryptography is necessary when communicating over any untrusted medium, which includes just about any network, particularly the Internet. Within the context of any application-to-application communication, there are some specific security requirements, including:

- **Authentication:** The process of proving one's identity. (The primary forms of host-to-host authentication on the Internet today are name-based or address-based, both of which are notoriously weak.)
- **Privacy/confidentiality:** Ensuring that no one can read the message except the intended receiver.
- **Integrity:** Assuring the receiver that the received message has not been altered in any way from the original.
- **Non-repudiation:** A mechanism to prove that the sender really sent this message.

Cryptography, then, not only protects data from theft or alteration, but can also be used for user authentication. There are, in general, three types of cryptographic schemes typically used to accomplish these goals: secret key (or symmetric) cryptography, public-key (or asymmetric) cryptography, and hash functions, each of which is described below. In all cases, the initial unencrypted data is referred to as plaintext. It is encrypted into cipher text, which will in turn (usually) be decrypted into usable plaintext.

Types of cryptography algorithms are:

- **Secret Key Cryptography (SKC):** Uses a single key for both encryption and decryption such as Data Encryption Standard (DES), AES (Advanced Encryption Standard). These methods focus on the encryption and decryption of the information.[1]
- **Public Key Cryptography (PKC):** Uses one key for encryption and another for decryption such as RSA, Elliptic Curve Cryptosystem. (ECC), focus on the public exchange of the key.[1]
- **Hash Functions:** Uses a mathematical transformation to irreversibly "encrypt" information. It a function, mathematical or otherwise, that takes a variable-length input string and converts it to a fixed length (generally smaller) output string[5]

Cryptography can be defined as the study of mathematical techniques related to the security of transmission and storage of information. Cryptography is currently one of the most

important tools in information security. Although it has been historically linked to confidentiality, modern cryptography covers a much wider range of issues, such as integrity, authentication and non-repudiation.

## II. HASH FUNCTIONS

One of the most important classes of cryptographic algorithms in use is the class of cryptographic hash functions. Hash functions are essentially easy-to-compute functions that produce a digital fingerprint of messages or data. Hash functions are ubiquitous in today's IT systems and have a wide range of applications in security protocols and schemes, such as providing software integrity, digital signatures and password protection. Furthermore, hash function algorithms are also used for constructing pseudo-random number generators, key derivation algorithms, message authentication codes [6].

Given the many applications of hash function algorithms, it is not surprising that secure hash functions need to feature a wide range of structural properties. In the following sections we provide a brief overview of cryptographic hash functions and their main use. Hash functions which have a "dedicated" design are fast and have considerable advantage over other algorithms which are based on block cipher[8]. Hash is mainly used to verify the accuracy of the information and to avoid any unauthorized changes during the communications. It is the authentication function of the security mechanism [6].

There are different types of well-known algorithms: MD5 and SHA-1, SHA-256, SHA-512.

### A. MD5

In 1990, Rivest designed Message Digest 4 (MD4). In 1991; MD5 was designed to replace the earlier Hash MD4. It became a milestone in the development of Hash. One of the characteristics of MD5 is that it processes a variable length inputted message into an output of fixed-length: 128-bit. MD5 digests have been widely used in the software world to provide some assurance that a transferred file has arrived intact. For example, file servers often provide a pre-computed MD5 checksum for the files, so that a user can compare the checksum of the downloaded file to it. Unix-based operating systems include MD5 sum utilities in their distribution packages, whereas Windows users use third-party applications. We refer to the output as digest (or message digest). The basic procedure required to process the digest is as follows:

- Divide an input message of any length by 264 and obtain the residue. This means that the 64-bit data is appended to the end of the message.

- Then check the input message to see if the length is congruent to 512 modulo 448. If not, add a sequence of bits as required. This is done by adding a sequence of 1000...0 bits[1]

The length of the sequence depends on the length of the original message. To check the result we use the standard test in the RFC1 321, as provided by Rives, and posted. If the input message is null, then the output of the 16-byte digest will be "d41d8cd98f00b204e9800998ecf8427e."

The MD5 algorithm has been specified as part of the IPv6 protocol. However, some questions have been raised as to the ability of MD5 to provide the throughput needed for high-speed networks. The cause of poor performance for MD5 is identified, and three areas are examined to determine where improvements can be made. The three areas examined are compiler technology, architectural changes, and implementation strategies. Of the three, only implementation strategies provide an avenue for significant improvement. However, the algorithm is still deemed too slow for good performance on high throughput networks [4].

### B. SHA

Secure Hash Algorithms (SHA) was published in 1993 by the National Security Agency (NSA) and the National Institute of Standards and Technology (NIST). This Hash is designed based on the principles similar to MD4, an earlier version of MD5. In 1995, some weakness was found and subsequently corrected, so the earlier version is usually called the SHA-0, and the successor - SHA-1.

The SHA-1 produces a 60-bit digest for the message; except the procedure is similar to MD5, and the usage of the initial value and function are different. Because the length of digest is longer than MD5, it is considered a highly secure Hash. The probability is 280 to find the same digest for the different messages. Use the description, which was published by NIST as a further explanation. By using the benchmark for testing the design of the Hash, the programmer can check if the Hash that he wrote is correct.

In the Hash, say  $H()$ , let  $M$  be the original message,  $M'$  is another message different from  $M$ . If the comparison of  $H(M)$   $H(M')$  does match, then  $M$  should equal  $M'$ . However, the probability of the result is rather low according to the property of the Hash. In other words, it is computationally impossible to reverse the input message  $M$  from the digest  $H(M)$  when using the Hash.

Message	Digest
Null	d41d8cd98f00b204e9800998ecf8427e
a	0cc175b9c0f1b6a831c399e269772661
abc	900150983cd24fb0d6963f7d28e17f72
message digest	f96b697d7cb7938d525a2f31aaf161d0
abcdefghijklmnopqrstuvwxyz	c3fcd3d76192e4007dffb496cca67e13b
ABCDEFGHIJKLMNOPQRSTUVWXYZ	d174ab98d277d9f5a5611c2c9f419d9f
abcdefghijklmnopqrstuvwxyz 0123456789	
12345678901234567890123456789	57edf4a22be3c955ac49da2e2107v67a
01234567890123456789012345678	
9012345678901234567890	

FIG. 1: The benchmark of RFC 1321 for MD4

In the past it was thought that the repeatability of the digests which were produced by MD5 and SHA-1 were lower than 264. Since it is believed that MD5 and SHA-1 provide high security, these two algorithms are employed widely. Although

any algorithm is going to be cracked someday, the researchers believed that it would take a very long time to do that. The only method of breaking them was "sheer force," but that could take a very long time

Message	Digest
abc	a9993e364706816aba3e25717850c26c9cd0d89d
abcdbcdecdefdefgefghfghighijhijk ijkljklmklmnlmnomnopnopq	84983e441c3bd26ebaae4aa1f95129e5e54670f1
A million of "a"	34aa973cd4c4daa4f61eeb2bdbad27316534016f

FIG 2: The benchmark for SHA-1

### C. SHA-2

NIST has published four additional hash functions in the SHA family, each with longer digests, collectively known as SHA-2. The individual variants are named after their digest lengths (in bits): SHA-224, SHA-256, SHA-384, and SHA-512. The latter three were first published in 2001 in the draft FIPS PUB 180-2, at which time review and comment were accepted. FIPS PUB 180-2, which also includes SHA-1, was released as an official standard in 2002. In February 2004, a change notice was published for FIPS PUB 180-2, specifying an additional variant, SHA-224, defined to match the key length of two-key Triple DES. These variants are patented in US patent 6829355.

SHA-256 and SHA-512 are novel hash functions computed with 32- and 64-bit words, respectively. They use different shift amounts and additive constants, but their structures are otherwise virtually identical, differing only in the number of rounds. SHA-224 and SHA-384 are simply truncated versions of the first two,

computed with different initial values. The data path architecture of SHA-384/512 is almost the same as that of SHA-224/256, except for the word size [9].

These new hash functions have not received as much scrutiny by the public cryptographic community as SHA-1 has, and so their cryptographic security is not yet as well established. Gilbert and Handschuh (2003) have studied the newer variants and found no weaknesses. These new hash functions have much more complex structure than SHA-1 [10]

## III. CRYPTANALYSIS OF HASH FUNCTIONS

### A. CRYPTANALYSIS OF SHA-0

At CRYPTO 98, two French researchers, Florent Chabaud and Antoine Joux, presented an attack on SHA-0 (Chabaud and Joux, 1998): collisions can be found with complexity 261, fewer than the 280 for an ideal hash function of the same size.

In 2004, Biham and Chen found near-collisions for SHA-0 — two messages that hash to nearly the same value; in this case, 142 out of the 160 bits are equal. They also found full collisions of SHA-0 reduced to 62 out of its 80 rounds. Subsequently, on 12 August 2004, a collision for the full SHA-0 algorithm was announced by Joux, Carribault, Lemuet, and Jalby. This was done by using a generalization of the Chabaud and Joux attack. Finding the collision had complexity 251 and took about 80,000 CPU hours on a supercomputer with 256 Itanium 2 processors.

### B. CRYPTANALYSIS OF SHA-1

In light of the results on SHA-0, some experts suggested that plans for the use of SHA-1 in new cryptosystems should be reconsidered. After the CRYPTO 2004 results were published, NIST announced that they planned to phase out the use of SHA than 280 operations.-1 by 2010 in favor of the SHA-2 variants. In early 2005, Rijmen and Oswald published an attack on a reduced version of SHA-1 --- 53 out of 80 rounds - which find collisions.

In February 2005, an attack by Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu was announced. The attacks can find collisions in the full version of SHA-1, requiring fewer than 269 operations. (A brute-force search would require 280 operations.)

The authors write: "In particular, our analysis is built upon the original differential attack on SHA0, the near collision attack on SHA0, the multiblock collision techniques, as well as the message modification techniques used in the collision search attack on MD5. Breaking SHA1 would not be possible without these powerful analytical techniques.

The authors have presented a collision for 58-round SHA-1, found with 233 hash operations. The paper with the full attack description was published in August 2005 at the CRYPTO conference. In an interview, Yin states that, "Roughly, we exploit

Table 1: Comparison of hash functions

Algorithm	Output size (bits)	Internal state size (bits)	Block size (bits)	Max message size (bits)	Word size (bits)	Rounds	Operations	Collision
SHA-0	160	160	512	264 – 1	32	80	+,and,or,xor,rotl	Yes
SHA-1	160	160	512	264 – 1	32	80	+,and,or,xor,rotl	263 attack
SHA-256/224	256/224	256	512	264 – 1	32	64	+,and,or,xor,shr,rotr	None yet
SHA-512/384	512/384	512	1024	2128 – 1	64	80	+,and,or,xor,shr,rotr	None yet

### V. FORUM OF CRYPTANALYSIS

In recent years the process of data assurance has come under attack. The nature of data and its usage require that users have a high degree of assurance of data

the following two weaknesses: One is that the file preprocessing step is not complicated enough; another is that certain math operations in the first 20 rounds have unexpected security problems. On 17 August 2005, an improvement on the SHA-1 attack was announced on behalf of Xiaoyun Wang, Andrew Yao and Frances Yao at the CRYPTO 2005 rump session, lowering the complexity required for finding a collision in SHA-1 to 263.

Christophe De Cannière and Christian Rechberger further improved the attack on SHA-1 in "Finding SHA-1 Characteristics: General Results and Applications, receiving the Best Paper Award at ASIACRYPT 2006. A two-block collision for 64-round SHA-1 was presented, found using unoptimized methods with 235 compression function evaluations.

As this attack requires the equivalent of about 235 evaluations, it is considered to be a theoretical break. To find an actual collision, however, a massive distributed computing effort is required. To that end, a collision search for SHA-1 using the distributed computing platform BOINC is currently being made.

At the Rump Session of CRYPTO 2006, Christian Rechberger and Christophe De Cannière claimed to have discovered a collision attack on SHA-1 that would allow an attacker to select at least parts of the message.

The new structures are carefully fine-tuned to make the current attacks on SHA-1 infeasible. We can reach better performance security trade-offs by considering radically different designs. At the very least, a substantial effort should be made to evaluate designs that have little in common with the current SHA family. One of the goals of follow-up workshops should definitely be to encourage the submission of new and original designs.[7]

### IV. SHA SIZES

In the table1 below, internal state means the “internal hash sum” after each compression of a data block;

integrity.Dependence upon data integrity affects users in all areas. While in many cases a change in the data may not have any contextual effects for users, in other cases a change may affect the data’s functionality or its usefulness. Application or

system code, which fails verification for data integrity, is suspect, and may in fact be malicious code. [2]

Other cases may involve changed data which, while not having any contextual differences (“white space” is altered in a text document), render the document unusable. This would result in a failure to verify a text document in a legal (forensic) setting.[2]

Data verification schemes have been developed in order to process data against a known result, which was produced against an original DataStream. Different algorithms have been developed in order to determine whether or not the DataStream is a bit-for-bit copy of the original. Comparing the original generated checksums against the checksums obtained by processing the suspect DataStream does this.

The assumption is that when a different result is obtained, the data are “suspect”. For example, in those cases in which the data are used by programs, results may be incorrect: dramatic instances may involve configuration data utilized in sensitive applications, causing applications to fail or radically depart from normal operations. Cryptographers have learned much about hash functions and how to attack them in the past couple of years, yet cryptanalysts at the workshop generally agreed that practical attacks on the SHA-2 hash functions are unlikely in the next decade. However, attacks and research results could reduce their strength well below theoretical work levels (2112, 2128, 2192, and 2256 operations for SHA-224, SHA-256, SHA-384, and SHA-512, respectively). Attendees thus agreed that starting work now to develop new hash functions would be prudent.[3]

Cryptanalysis means to break or analyze the ciphertext in cryptography. Generally speaking, this doesn't mean that it can recover the plaintext in a minute from the ciphertext in hand. But the algorithm to obtain the key of the ciphertext is measured in polynomial time complexity. The cracking of the Hash, either MD5 or SHA-1, means that they are not foolproof.[1] In Internet communication and the procedure of encryption and decryption, in order to see which part has been seriously affected by the cracking of MD5 and SHA-1. Generally speaking, after verifying each other's identity, both ends of the network communication can send the digital information with a digital signature attached. Basically, there are two steps in the digital signature procedure: Signature and Verification. The procedure for a signature is as follows:

First hash the original document to be a digest; then use the private key of the owner of the document to encrypt the digest. That is the basic procedure of a digital signature. After that, the signature is encapsulated along with the original document. Then send the encapsulated signature to the receiver. Once the encapsulated signature has been received, the receiver launches the verification procedure immediately. First, the receiver uses the sender's public key to decrypt the digital signature and unveil the digest, hash the transmitted plaintext, and obtain a new digest. [1]

These two digests, the decrypted digest and the new digest is compared. If the result matches, then the signature is verified, if not, the signature will be voided. Therefore, if the receiver successfully verifies the signature of the sender, it confirms two things:

- The digital document is certain to have been sent by the sender who signed it. The source of the digital document can be confirmed to come from the particular sender, because the most important part, the private key, is only controlled by the person who is (was?) signing it. In other words, this fact cannot be denied.
- The digital document that the receiver received has not been altered during the communicating process, and retains its complete information, since after the original document has been signed, any change will be noted.

Since the chance of collision has been raised in the event of MD5 being cracked, we have to seek another solution for the sake of security. It is quite possible that SHA-1 will be broken in a few years. Fortunately, except for MD5 and SHA-1, there are some substitutes proposed. They are more difficult to break than SHA-1 and have been considered for the candidate list. In August 2002, for example, NIST released a series of new algorithms including SHA-224, SHA-256, SHA-384, SHA-512, and Tiger, and made them into the FIPS. Tiger, is proposed by Anderson et al and the digest of Tiger is longer than the current SHA-1, being 192-bit-long. In addition, the SHA-512 can produce a digest of 512-bit-long, and the dimension of security. It is more difficult to obtain the same digest output for any two different string inputs. The list of the result of the testing for the string of "abc" in the newer hash algorithms.

## CONCLUSION

New cryptographic codes are created and broken every day. It is through this challenge that the cryptographic technology advances forward. Although the existing information shows that there is no immediate damage with SHA-1, we still need to find substitutes, secured Hash Algorithms, SHA-series, such as SHA-224, SHA-256, SHA-384, and SHA-512, in order to replace SHA-1 before the year 2010. The main point of this report on the cracking of SHA-1 is not to tell us how fragile SHA-1 is, but to remind us that we have to speed up the development and strengthening of our existing security mechanisms.

## REFERENCES

1. Concerns about Hash Cracking: Aftereffect on Authentication Procedures in Applications of Cyberspace  
Shiuh-Jeng Wang, Hung-Jui Ke  
IEEE, 2007.
2. Hashing and data integrity: Reliability of hashing and Granularity size reduction: Christopher Malinowski, Richard Noble  
2007 Elsevier Ltd..
3. Cryptographic Hash Standards: Where Do We Go from Here?  
Tim Grance, Rick Kuhn, Susan Landau  
IEEE, 2006
4. Problems using MD5 with IPv6  
Larry D. Barchett, Arindam Banerji, John M. Tracey, David L. Cohn
5. When Hashes Collide  
Peter Gutmann, David Naccache, Charles C. Palmer  
IEEE, 2005
6. Recent developments in cryptographic hash functions: Security implications and future directions  
Carlos Cid,  
2006 Elsevier Ltd.
7. The NIST Cryptographic Workshop on Hash Functions  
Carl Landwehr  
IEEE, 2006
8. Hardware implementation analysis of SHA-256 and SHA-512 algorithms on FPGAs  
Imtiaz Ahmad, A. Shoba Das  
Science direct, 2007
9. ASIC-hardware-focused comparison for hash functions MD5, RIPEMD-160, and SHS  
Akashi Satoh, Tadanobu Inoue  
Science direct, 2007
10. SHA: A design for parallel architectures?  
A. Bosselaers, R. Govaerts and J. Vandewall  
W. Fumy, Ed., Springer-Verlag, 1997