IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

Cryptography Machine for Encryption and Decryption

Om Gautam, Aditya Sabale, Harshal Sawant, Prof. A.Y. Kadam
Department of Information Technology
Smt. Kashibai Navale College of Engineering, Pune, India

Abstract: *Computers are most valuable when they are used to solve problems that humans cannot easily solve for themselves. Charles Babbage, for example, wanted to automate the production of mathematical tables, partly because it was a tedious task, but mostly because the people who undertook the necessary calculations made so many mistakes. Computers, however, are also useful when they solve problems faster than human beings. If you face a situation in which timeliness is essential, you may not be able to wait for results generated at human speeds. In such cases, it may be necessary to develop a technological solution to get the answers you need when you need them. In World War II, the Allies faced precisely this situation. The shipping lanes of the North Atlantic were under such threat from German U-boats that Britain was in danger of being starved into submission. Breaking the U-boat code was a critical turning point in the war and may have changed its outcome. Faced with a code that changed every day, the British had to develop mechanical tools that would allow them to read German military dispatches quickly enough to act on that information. Breaking the German military codes was an early application of cryptography, which is the science of creating and decoding messages whose meaning cannot be understood by those who intercept the message. In the language of cryptography, the message you are trying to send is called the plaintext; the message you actually send is called the ciphertext. Unless your adversaries know the secret of the encoding system, which is usually embodied in some privileged piece of information called a key, intercepting the ciphertext should not make it possible for them to discover the original plaintext version of the message. On the other hand, the recipient, who is presumably in possession of the key, can easily translate the ciphertext back into its plaintext counterpart. Cryptographic Hash functions are used to achieve a number of security objectives. In this paper, we bring out the importance of hash functions, its various structures, design techniques, attacks and the progressive recent development in this field.*

Keywords: Cryptography, Hash function, compression function

I. INTRODUCTION

Cryptographic techniques mainly encryption & decryptions have been used for centuries to protect military and political secrets and D. Kahn in has given comprehensive study of this history. Throughout this history of cryptology, confidentiality has taken the primary seat and it was believed that if the secrecy is maintained (using symmetric encryption and secret key) then the authentication will automatically be achieved. The logic was if decryption of an encrypted text results in a meaningful message it must have been constructed by someone who knows the secret key. During all this period the field of cryptology was kingdom of selected few i.e. it was studied and practiced by few. The trend changers were Diffie and Hellman, who are credited for advent of public key cryptography in mid 70s. Their seminal paper “New Directions in Cryptography” introduced a number of relevant concepts like Digital Signatures and differentiated Confidentiality from Authentication and to quite an extent initiated the development of cryptographic schemes for the protection of authenticity. These schemes use a very important cryptographic primitive named ‘Cryptographic Hash Functions’. However cryptographic hash functions have received much less attention from the cryptologic community than encryption schemes in the past. Bert Rompay in his thesis [quoted the example of NESSIE (New European Scheme for Signature Integrity and Encryption) project to illustrate how cryptographic hash functions have been ignored in the past. In NESSIE project, seventeen block ciphers and six stream ciphers were submitted as candidates (both are categories of encryption schemes), but only one un- keyed hash function and two keyed hash

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

functions (also known as MAC – Message Authentication were submitted. Rompay also gave example of open competition used by the National Institute of Standards and Technology (NIST)in the United States to decide on the block cipher to be used as Advanced Encryption Standard. This competition had fifteen candidates out of which the Rijndael block cipher finally chosen. On the other hand, for its hash function standard NIST simply chose the SHA hash functions, designed by the NSA without disclosure of their design strategy or any supporting cryptanalytic results. However the trend has changed in recent years because of the wide range of applications areas of cryptographic hash functions. Cryptographic Hash Functions are used to achieve a number of Security Goals like Message Authentication, Message Integrity, and are also used to implement Digital Signatures (Non-repudiation), Entity Authentication and Digital Steganography. Considerable research has been undergoing in the field of Cryptographic Hash Functions. Hash Functions are being generated from existing primitives like Block ciphers (e.g. Whirlpool, Skein) as well as being explicitly and specially constructed from scratch like MD family and SHA family of hash functions.

1.1 Organization of the paper

This paper will present the detailed study of Cryptographic Hash Functions. Organization of the paper is as follows. In Section 2 and 3 the basic concepts like definitions, properties and applications of Hash functions are detailed. Section 4 discusses the basic as well as currently used iterative structures of Hash functions. In Section 5 and 6 security properties and possible attacks are detailed. In Section 7 various design techniques of underlying compression functions have been explained. Section 8 throws light on the current scenario in Hash functions.

II. CRYPTOGRAPHIC HASH FUNCTIONS

The term hash function has been used in computer science from quite some time and it refers to a function that compresses a string of arbitrary input to a string of fixed length. However if it satisfies some additional requirements (as detailed further), then it can be used for cryptographic applications and then known as Cryptographic Hash functions. Cryptographic Hash functions are one of the most important tools in the field of cryptography and are used to achieve a number of security goals like authenticity, digital signatures, pseudo number generation, digital steganography, digital time stamping etc. Gaurav ram in his thesis has suggested that the usage of cryptographic hash functions in several information processing applications to achieve various security goals is much more widespread than application of block ciphers and stream ciphers.

Rompay has given the following formal definition of hash functions

Definition: *A* ***hash function*** *is a function* ***h****: D □ R, where the domain* ***D*** *= {0,1}* * *and* ***R*** *= {0,1}* ^{n} *for some n >= 1*

Cryptographic Hash Functions are broadly of two types i.e. Keyed Hash functions; the one which uses a secret key, and Un-keyed Hash Functions; the other one which does not use a secret key. The keyed Hash functions are referred to as Message Authentication code. Generally the term hash functions refer to un- keyed hash functions and, in this paper, **we will concentrate on Un-keyed Hash functions only.** Un- keyed or simply Hash functions (some time also known as MDC – Manipulation Detection Code) can further classified into OWHF (One Way Hash Functions), CRHF (Collision Resistant Hash Functions) and UOWHF (Universal One-way Hash Functions) depending on the additional properties it satisfies.

III. SECURITY SERVICES OF CRYPTOGRAPHIC HASH FUNCTIONS

3.1 Achieving Integrity & Authentication

Verifying the integrity and authenticity of information is a prime necessity in computer systems and networks. In particular, two parties communicating over an insecure channel require a method by which information sent by one party can be validated as authentic (or unmodified) by the other.

Message Integrity & Authentication may be implemented in multiple ways. Symmetric Encryption based mechanisms may be used but they have their own drawbacks. Drawbacks like speed, cost factor, optimization for data sizes etc. have been highlighted by Tsudik. Such methods combine the Confidentiality and Authentication functions. However there are scenarios where encrypting full message (confidentiality) is not required. For such applications keeping message secret is not the concern but authenticating it is important. For example in SNMP (Simple Network Management

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

Protocol), it is usually important for a managed system to authenticate incoming SNMP commands (like changing the parameters at the managed system), but concealing the SNMP traffic is not required.

In order to implement message authentication and integrity, the alternative techniques (other than the methods mentioned in last paragraph) are *MAC* or *hash functions*. MACs may be constructed out of block ciphers like DES. More recently, however, there has been a surge of interest in the idea of constructing MACs from cryptographic Hash Functions. In addition to using Hash Functions for implementing MAC, Hash functions can be used to achieve message authentication and integrity goals without the use of symmetric encryption. Tsudiac has detailed a protocol based on the same idea. Rompay has also detailed the ways of ensuring authentication using hash functions alone as well as using hash functions with encryption. The usage of Hash Functions for Message Authentications and ensuring message integrity has surged because majority of hash functions are faster than block ciphers in software implementation and these software implementations are readily and freely available.

3.2 Implementing Efficient Digital Signatures

Digital signature is a security goal of a cryptosystem which intends to achieve the goal of authenticity and a security service or property of non-repudiation. MAC and Hash Functions alone do not implement the Security goal of Digital Signatures. It was Diffie and Hellman who first realized the need for a message dependent electronic signature (fingerprint) to avoid disputes between sender and receiver. RSA was the first public key crypto systems with digital signature capabilities. However there has been an interesting part of this invention. James Ellis, Clifford Cocks, and Malcolm Williamson from GCHQ (Government Communication Head Quarters), Cheltenham, Britain perhaps invented the idea of public key in 1972. The three Britons had to sit back and watch as their discoveries were rediscovered by Diffie, Hellman, Merkle, Rivest, Shamir and Adleman over the next three years because of the polices of GCHQ that all work is top secret and cannot be shared with anyone.

Hash functions are used to optimize the digital signature schemes. Without the use of Hash, the signature will be of same size as message. The fundamental concept here is instead of generating the signature for the whole message which is to be authenticated; the sender of the message only signs the digest of the message using a signature generation algorithm. The sender then transmits the message and the signature to the intended receiver. The receiver verifies the signature of the sender by computing the digest of the message using the same hash function as the sender and comparing it with the output of the signature verification algorithm. It is obvious that this approach saves a lot of computational overhead involved in signing and verifying the messages in the absence of hash functions.

3.3 Authenticate Users of Computer Systems

Hash functions may be used to authenticate the users at the time of login. The passwords are stored in the form of message digest to avoid access of the same even to Database Administrators (because of Pre-Image resistance of Hash digest). Whenever user tries to login and enter the password, the message digest of the entered password is computed and compared with the digest stored in the database. If it matches, then login is successful, otherwise user is not authenticated.

3.4 Digital Time Stamping

Majority of text, audio and video documents are available in digital format and a number of simple techniques and tools are available to change digital documents. So some sort of mechanism is required to certify when such a document was created or last modified. Digital timestamp solves the purpose and provide a temporal authentication Rompay in his thesis work has suggested the multiple ways like simple scheme based on trusted third party, scheme that links timestamps into temporal chain and the other one that make use of Merkle Tree. Rompay highlighted that Digital time stamp helps in protecting intellectual property rights, ensuring strong auditing procedures, and implementing true non-repudiation services. Before, Haber and Stornetta has also detailed how One way hash functions and digital signatures can be used to implement the digital time stamping.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

3.5 Hash functions as PRNG

Hash functions as one-way functions can be used to implement PRNG (Pseudo random number generator). A very simple technique can be to start from an initial value (s) known as seed and computer H(s) and then H(s+1), H(s+2) and so on has given some other ways of constructing Pseudo random strings from Hash functions.

3.6 Session Key Derivations

Hash functions as one-way functions can be used to generate sequence of session keys that are used for the protection of successive communication sessions. Starting from a master key K0, the first session key can be K1 = H(K0) and second session key can be K2 = H(K1) and so on. Matyaset.al. described the key management scheme based on control vectors which makes use of hash functions and Encryption functions for generating session keys.

3.7 Constructions of Block Ciphers

Block ciphers can be used to construct a cryptographic hash function however the inverse is also true and there has been block ciphers designed using Hash functions. In Hand schuh and Naccache proposed to use the compression function of cryptographic hash function SHA-1 in encryption mode. The name of the cipher was SHACAL. SHACAL-1 (originally named SHACAL) and SHACAL-2 are block ciphers based on SHA-1and SHA-256 respectively. SHACAL-1 (originally named SHACAL) is 160-bit clock cipher and SHACAL-2 is 256-bit block cipher. Both were selected for the second phase of NESSIE project. In 2003 SHACAL-1 was not recommended for NESSIE portfolio because of concerns about its key schedule, while SHACAL-2 was finally selected as one of the 17 NESSIE finalists. SHACAL-1 used the compression function of SHA-1 and turned it into a block cipher by using the state input as the data block and using the data input as the key input. In other words SHACAL-1contemplated the SHA-1 compression function as an 80-round, 160-bit block cipher with a 512-bit key. Keys shorter than 512 bits are supported by padding them with zero up to 512. SHACAL-1 was not intended to be used with keys shorter than 128-bit.

3.8 Other Applications

Hash Functions can also be used to index data in hash tables, for fingerprinting, to detect duplicate data or uniquely identify files, and as checksums to detect accidental data corruption and for generating random numbers also. Looking at this wide range of applications, it is not correct to say that Hash Functions belong to one particular cryptographic sub branch. These cryptographic tools deserve a separate status for themselves. They are used in almost all places in cryptology where efficient information processing is require.

Security Properties of Hash Functions

Basic Security Properties

Basic notion of security of Hash functions revolves around preimage resistance, second-preimage resistance and collision resistance as defined in Section 2. In literature Collision resistance property is referred to as collision freeness or strong collision resistance, second pre-image resistance is called as weak collision resistance and preimage resistance is referred to as one-wayness. It is easy to see that collision resistance implies second-preimage resistance i.e. if a hash function is collision resistant then his also second pre-image resistant. However second-preimage resistance and one-wayness are incomparable (the properties do not follow/imply one another), although hash functions which are one-way but not second- preimage resistant are quite contrived. In practice, collision resistance is the strongest property of all three, hardest to satisfy and easiest to breach, and breaking it is the goal of most attacks on hash functions.

Rogaway and Shrimpton extended the notion of hash function security and defined seven different security notions, three on pre-image resistance, three on second pre-image resistance and one on collision resistance. The work of Rogaway and Shrimpton is based on generic concept of a hash function family that is a finite set of hash functions with common domain and range. The security of hash function and probability of success of an adversary depends on the manner in which one chooses a particular hash function from the hash function family for example the hash function can be chosen on random or may be fixed element. Based on these variations, seven different security notations and relation between them are given in.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

Avalanche Criterion and Completeness

From a good hash function it is desired that for two different inputs, the output of hash function should be completely different, regardless of difference in inputs. The same can be formalized with two properties of hash functions i.e. Completeness and Avalanche effect. Strong Avalanche effect represents a property when small change in input result in a significant change in message digests. Completeness represents a property when each input bit affects all output bits. Strict Avalanche Criterion combines both the avalanche effect and the completeness and represent a property when a change in one bit of input results in changing every bit of the output (message digest) with a probability of ½. If these criterions are not satisfied then the probability of successful attack on the hash functions increases considerably.

Certificational Properties of Hash functions

Near Collision Resistance: A hash function is said to be Near Collision resistant if it is hard to find two messages x and x' such that the hamming distance between h(x) and h(x') is small (typically a few bits). Near collision may also be termed as almost collision and can be defined for underlying compression function also. With respect to underlying compression function, almost / near collision means that two message blocks are found for which the difference between the outputs has a low Hamming weight. Gauravram quoted the example of how near collisions in case of hash functions with truncated outputs can lead to full collision. If we have a truncated hash function that makes use of leftmost 224 bit of output after chopping rightmost 32 bits then if near collision is found such that message digests only in the rightmost 32 bits then such a near collisions are practically full collisions only.

Certificational Properties on the Compression Function

Certificational properties or weaknesses on the compression functions used in the MerkleDamgard structure or similar other iterative structures are classified based on the IV / H0 (Initial value) used. These classifications and nomenclature have varied from author to author. For example Pseudo collision resistance as defined in is termed as Special pseudo (type-3) collision resistance. Similarly for an attack, Rompay in has used the nomenclature as Random IV collision and for the same attack Gauravram in has used the nomenclature as Semi free start collision. Furthermore Mironovin defined Pseudo Collision resistance and Free Start collision resistance as two separate properties on the other side Gauravram and Knudsen termed pseudo collision resistance and free start collision resistance as one and the same thing. In this sub section we use the terminology and classification done by Gauravram in as it has been found most exhaustive and clear but at the same time, we also list the alternative nomenclature used by different authors.

IV. METHODS OF ATTACK ON HASH FUNCTIONS

Attacking a hash function means breaking one of the security properties (basic, extended or certificational property) of hash functions. For example breaking pre- image resistance means adversary is able to break the pre-image property i.e. an adversary is able to create a message that hashes to a specific hash. Breaking certificational properties may not yield a practical attack but are an important warning to reflect weakness in the hash / compression function. Gauravram [16] recommended switching to a strong hash function when an attack on certificational properties is observed. In an iterated hash function, if a pre-image or collision (Type-1 collision only) can be found for compression function (f), the same can be extended and an attack on hash function can be derived. So attacks may focus on structure of hash function or on algorithm of compression function. In this sub section we will review different types of attacks on hash functions. Attacks on Hash functions can be classified into two broad categories - Brute Force Attacks and CryptanalyticalAttack.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

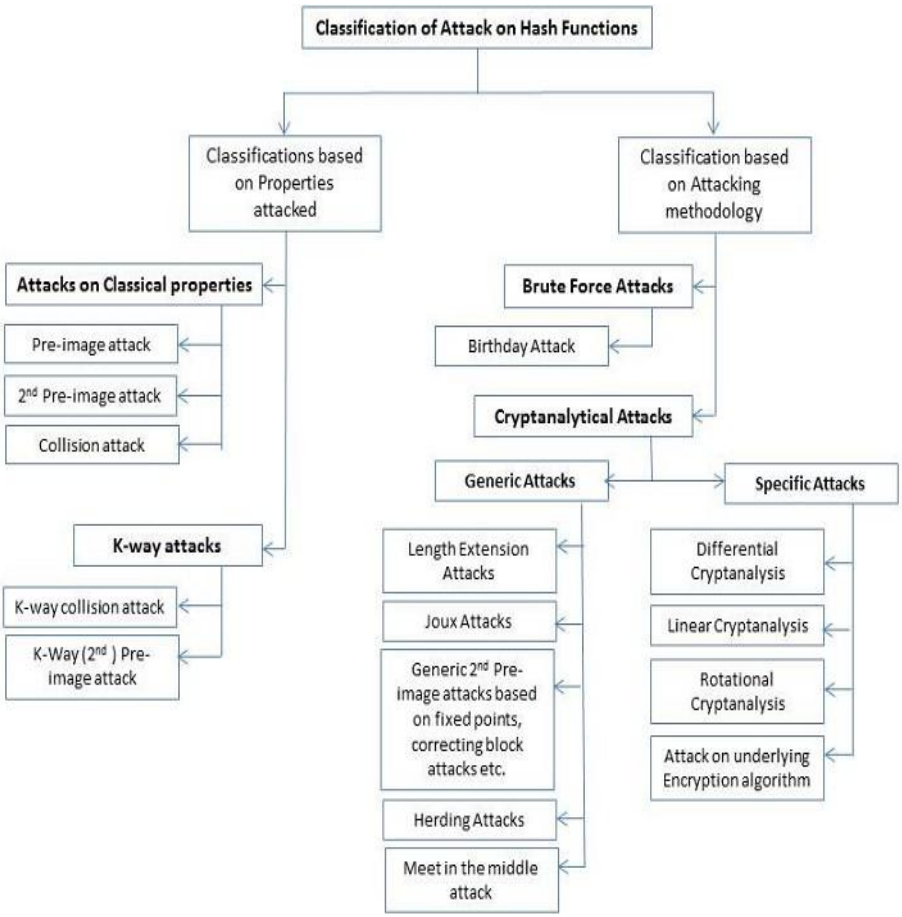


Fig. 2 Classification of attacks on Hash Functions

4.1 Brute Force Attack

Brute force attacks work on all hash functions independent of their structure andany other working details. They are similar to exhaustive search or brute- forcekey recovery attacks on the encryption schemes to extract the secret key of the encryption scheme. The security of any hash function lies in its output bit size. For a hash code of length n, the level of effort required to resist different brute force classical attacks on hash functions is as follow:

Pre-image attack: Effort required for brute force attack = 2n. In this attack, for a given n-bit digest h ofthe hash function H( ), the attacker evaluates H( ) with every possible input message M until the attacker obtains the value h.

2nd Pre-image attack: Effort required for brute force attack = 2n.In this attack, for a given message M and the hash function H( ), the attacker tries H( ) with every possible input message' ≠ M until the attacker obtains the value H(M).

Collision attack: Effort required for brute force attack

= 2n/2. In this attack, for a given hash function H, the attacker tries to find two messages M and M' such that M ≠ M' and H(M) = H(M'). On average the opponent would have to try 2n / 2 (= 2n-1) messages to find one that matches the hash code of the intercepted message However a chosen plain text attack (based on Birthday Paradox) is possible and in that case the effort required for collision in a Hash function is 2n/2 in place of 2n-1 [29]. It is also referred as Birthday attack.

In addition to the above discussed classical attacks, the following natural extensions have also been studied by different authors.

K-Way Collision attack for K >=2: Find K different messages Mi such that H(M1) = … = H(MK).

K-Way (2nd) pre-image attack for K>=1: Given Y (or M with H(M) = Y), find K different messages Mi, with H(Mi) = Y and Mi ≠ M.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

4.2 Cryptanalytical Attack

Cryptanalysis of Hash functions focuses on the underlying structure of hash function and/or on the algorithm of Compression Function. Due to fixed size of the hash values compared to much larger size of themessages, collisions must exist in hash functions. However, for the security of thehash function, they must be computationally infeasible to find. Collisionsin hash functions are much easier to find than pre-images or 2nd pre-images.

Informally, a hash function is said to be "broken" when a reduced number ofevaluations of the hash function compared to the brute force attack complexitiesand the strengths estimated by the designer of the hash function are used toviolate at least one of its properties immaterial of the computational feasibilityof that effort. For example, assume that it requires 290 evaluations of the hashfunction to find a collision for a 256-bit hash function. Though it is impracticalto generate this amount of computational power today, the hash function is saidto be broken as this factor is less than the 2128 evaluations of the hash functionrequired by the Birthday attack. It should be noted that hash functions are easier to attack practically thanencryption schemes because the attacker does not need to assume any secrets andthe maximum computational effort required to attack the hash function is onlyupper bounded by the attacker's resources not users gullibility. This is not thecase with block ciphers where the maximum practical count of executions of theblock algorithm is limited by how much computational effort the attacker can getthe user to do [16].

Collision finding algorithm and attacks may be classified as single block attacks or multi block attacks depending on whether that attack uses single block (i.e. one compression function) or more than one block (i.e more than one iteration of compression function) for finding collision or pre-images.

A. Specific Attacks

The attacks that work on specific hash function or the algorithm of its compression function are called specificattacks. For example, collision attacks on the specific hash functions MD4 [30],MD5 [31,32], SHA-0 [33,34] and SHA-1 [33,35]. Attacks using differential cryptanalysis, linear cryptanalysis, rotational cryptanalysis &attack on the underlying encryption algorithms are type of specific cryptanalysis attacks. The most successful of these are the attacks based on differential cryptanalysis.

- Differential Cryptanalysis: Differentialcryptanalysis was introduced by Biham and Shamir [59] and the technique was mainly devised to analyse block ciphers. In differential cryptanalysis the correlation between the difference in input and output is studied. If X and X' are two inputs then the difference between them is defined as $\Delta X = X$ op X'. If H and H' are two corresponding message digests then the difference between them is defined as $\Delta H = H$ op H'. The difference operation op canbe XOR operation or integer subtraction or any other operation. For differential cryptanalysis attack, the attacker searches for specific difference in inputs (ΔX)that result in specific difference in output (ΔH) with high probability. In case of hash function, the difference in output should be zero to result in collisions. Examples of specific attacks using differential cryptanalysis are [30, 31, 32,33, 34, 35, 60, 61].
- Linear Cryptanalysis: Linear cryptanalysis was proposed by Matsui [62]. S. Bakhtiariet. al. in [58] quoted that for Block ciphers like DES, better results have been obtained with Linear Cryptanalysis compared to Differential Cryptanalysis. Hash functions based on the Encryption algorithm can be susceptible to linear cryptanalysis, but till date not much successful attack on Hash functions using linear cryptanalysis has been reported.
- Rotational Cryptanalysis: The term Rotational cryptanalysis was coined by in February 2010 by Dmitry Khovartovich and Ivica Nikolic in [64]. The attack may also be classified as generic attack because as per [64] it may be applied on all the algorithms that are based on three operations modular addition, rotation and XOR (ARX for short). However we have placed it under the category of specific attacks as this attack has been demonstrated by Khovartovich and Nikolic against reduced round Threefish cipher – part of Skein hash function [66], a SHA3 competition [45] candidate only. Secondly as per our classification, the generic attacks are applicable to all the hash functions falling under a particular structure like MerkleDamgard, so it is better to consider rotational cryptanalysis as a specific attack. In October 2010, a followup attack that combines rotational cryptanalysis with the rebound attack was presented by the same authors along with Christian Rechberger in [65].

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

- Attacks on underlying Encryption Algorithm: Ifthe underlying compression function of hash function is implemented using the Encryption algorithm, then the weakness in encryption algorithm can be exploited to attack hash functions. Encryption function may have complementation property or weak keys or may have fixed points and the same may be used to attack complete hash function based on encryption algorithm. Miyaguchiet. al. in[63] analyzed the hash functions from the standpoint of the complementation property and weak keys of the block ciphers used in them and notified their weaknesses.

V. TYPE OF HASH FUNCTIONS BASED ON DESIGN OF UNDERLYING COMPRESSION FUNCTION

From the discussion in section 4, it is evident that for processing arbitrary length of input the iterative structure of hash function (may be MerkleDamgard or any other) is desired and the crucial part of this iterative structure is Compression function and thus designer can view of all these approaches have been given in this section.

5.1 Hash Functions based on Block Cipher as Compression functions

One of the possible approaches that have been studied by the authors is to design a compression function from an existing cryptographic primitive like block ciphers. The advantage is that the existing implementations in hardware or software can be reused. Secondly some existing block ciphers like DES [67] or AES [7] have received a lot of scrutiny, and thus there is a lot of trust in their security properties [3]. At the same time a number of drawbacks of block cipher based hash functions have also been observed. One of the arguments is that the block ciphers do not possess the properties of randomizing functions. For example they are invertible. This lack of randomness may lead to weakness that may be exploited [85]. Secondly the differential cryptanalysis is easier against block operations in hash functions than against block operations used for encryption; because the key is known so several techniques can be applied. [68, 69] suggest the various techniques of using differential cryptanalysis for attacking hash functions based on clock ciphers. Thirdly it has been suggested that block cipher based on hash functions are significantly slower than hash functions based on compression function specially designed for hash functions. It is also felt that use of a block cipher for a purpose for which it was not designed may reveal some other weaknesses which may not be relevant in case of encryption. However with the adoption of AES, there has been renewed interest in developing a secure hash function based on strong bock cipher and exhibiting good performance [85]. Hash functions based on Block ciphers can be further classified as follows:

A. Single block length construction

These are the schemesin which size of hash code equals the block size of underlying block cipher. A number of proposals have been made and the basic concept to construct compression function ffromblock cipher as described in [15] is as follow:

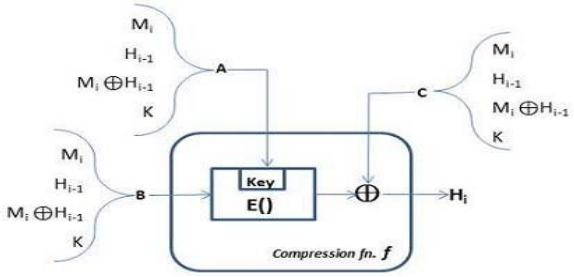


Fig. 3 Compression function based on block cipher

E is the clock cipher that takes two inputs A and B and produces an output that is XOR with variable C. Variable A, B and C can be either Mi, H_{i-1}, ($M_i \oplus H_{i-1}$) or a constant K (K may be assumed to be zero also). Message M is divided in to blocks and padding is done as illustrated in Section 4 and in each round one block M_i is processed in the compression function ***f***as per follow:

H_0 = Initial value

$H_i = E_A(B) \oplus C$

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

The three different variables A, B and C can take on one of four possible values, so there are 64 total schemes of this type. Prennel*et. al.*[72] studied them all and showed that 12 of them (as given in the *Table 1*) are secure.

Table 1: Secure Hash Functions as per [72] based on Block Cipher

Secure Schemes based on Block cipher to generate Compression function	**Other Common Name for the scheme as per the Literature**
$H_i = E_{Hi-1}(M_i) \oplus M_i$	Matyas-Meyer-Oases Scheme [70]
$H_i = E_{Hi-1}(M_i \oplus H_{i-1}) \oplus M_i \oplus H_{i-1}$	--
$H_i = E_{Hi-1}(M_i) \oplus H_{i-1} \oplus M_i$	Miyaguchi – Preneel Scheme Independently proposed by Miyaguchi[71] and Preneel[73]
$H_i = E_{Hi-1}(M_i \oplus H_{i-1}) \oplus M_i$	--
$H_i = E_{Mi}(H_{i-1}) \oplus H_{i-1}$	Davies-Meyer Scheme [70, 74]
$H_i = E_{Mi}(M_i \oplus H_{i-1}) \oplus M_i \oplus H_{i-1}$	--
$H_i = E_{Mi}(H_{i-1}) \oplus M_i \oplus H_{i-1}$	--
$H_i = E_{Mi}(M_i \oplus H_{i-1}) \oplus H_{i-1}$	--
$H_i = E_{Mi \oplus Hi-1}(M_i) \oplus M_i$	--
$H_i = E_{Mi \oplus Hi-1}(H_{i-1}) \oplus H_{i-1}$	--
$H_i = E_{Mi \oplus Hi-1}(M_i) \oplus H_{i-1}$	--
$H_i = E_{Mi \oplus Hi-1}(H_{i-1}) \oplus M_i$	--

For formal proof of the security of these 12 schemes refer to [75] and for various other schemes proposed in literature that have been shown to be insecure refer [15, 72].

B. Double block length construction

A Hash function generating digest of 64 bits (or 128 bits) is insecure as brute force collision will require 2^{32}(or 2^{64}) operations only. Using the Single block length construction schemes as mentioned in previous sub-section, we will get a 64 bit digest with DES as underlying block or 128 bit digest with AES as underlying block cipher. To increase the digest size of hash function and to make it more secure double length block constructions is suggested. It is schemes in which size of hash code doubles the block size of underlying block cipher. This means, DES will result in a 128-bit hash function, and AES in a 256-bit hash function. The best known scheme in this class as suggested by Rompay [3] is MDC2 and MDC4 designed by B. Bracht l*et. al.* [76, 77].MDC-2 is sometime called as Meyer-Schilling scheme. The compression function of MDC2 makes uses of two parallel computations of Matyas-Meyer-Oases scheme [70]. Explanation of MDC-2 as given in [3] is reproduced here using the terminology used in previous subsection.

Let C^L and C^R denote the left and right halves of b-bit block length of underlying block cipher. Then the compression function of MDC-2 can be described by $H_i \| H_i' = f(H_i \| H_i', M_i)$, which depends on the following computations:

The compression function of MDC-4 consists of two sequential executions of MDC-2 compression function. For the second MDC-2 compression, the keys are derived from the outputs (Chaining variables) of the first MDC-2 compression, and the plaintext inputs are the outputs (Chaining variables) from the opposite sides of the previous MDC-4 compression.

For details of few of the other double length construction schemes studied in literature like Quisquarter-Girault, LOKI Double Block, Parallel Davies Meyer, Tandem and Abreast Davies – Meyer schemes, refer [15, 78, 79, 80, 81]

Few of the famous hash functions based on block ciphers are listed below:

- **GOST Hash Function** – This hash function comes from Russia, and is specified in the GOST R.34.11-94. It uses the GOST block encryption algorithm. For details refer [82]

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

- **AR Hash Function**: AR Hash function was developed by Algorithmic Research, Ltd. and has been distributed by the ISO for information purposes only. Its basic structure is a variant of underlying block cipher (DES in the reference) in Cipher Block Chaining mode. For details refer [83]
- **Whirlpool Hash Function**: Whirlpool is one of the only two hah functions endorsed by NESSIE (New European Scheme for Signatures, Integrity and Encryption). Unlike virtually all other proposals for a block-cipher based hash function, Whirlpool uses a block cipher that is specifically designed for use in the hash functions and that is unlikely ever to be used as a standalone encryption function. For details refer [84] **Skein Hash Function**: Skein hash function is one out of five finalists in the NIST hash function competition [45] to design SHA-3 standard that will replace SHA-1 and SHA-2 [4, 5, 6, 8]. The algorithm is based on Three fisht weakable Block Cipher. For details refer [66]
- **Grøstl Hash Function:** Just Like Skein, Grøstl also is a SHA-3 final round candidate algorithm. Its compression functions is not exactly uses existing block cipher but Grøstl uses the same S-Boxes as AES. Its compression function ***f*** is based on a pair of permutation functions P and Q and these permutation functions are heavily based on AES [7] block cipher.

C. Hash functions based on Modular Arithmetic

Compression function can also be designed using modular arithmetic. This allows the reuse of existing implementations of modular arithmetic such as in asymmetric cryptosystems. The idea of cryptosystems based on modular arithmetic is to reduce the security of a system to the difficulty of solving the problems in number theory. Two important hard problems in number theory which can act as a base for generating cryptosystems are factorization and Discrete logarithm. Rompay in [3] has referred to design of two variants of MASH hash functions based on modular arithmetic. The advantage of such hash functions is that the level of security can be easily enhanced by choosing Modulus M of appropriate length but hash functions based on modular arithmetic are very slow, even slower than block cipher based hash functions. Also many such constructions have been broken in the past.

D. Dedicated Hash Functions

Dedicated hash functions are the one which are designed for the explicit purpose of hashing. Compression functions of dedicated Hash functions are not based on the existing cryptographic primitives like block ciphers and are not constrained to reuse existing components such as block ciphers or modular arithmetic. This means that they can be designed with optimised performance in mind. A number of such hash functions have been designed. Few of the famous dedicated hash functions and the status of attacks on these hash functions are as follows:

MDx Familyof hash functions: MD2, MD4 and MD5 are three hash functions from MDx family. Compared to other two, MD2 is slower and has not obtained much success. Dedicated hash functions which have received the most attention in practice are those based on the MD4 algorithm [3]. MD4 is a hash function proposed by R. Rivest in 1990 [9]. It was designed specifically towards software implementation on 32-bit platforms. Because of security concerns, Rivest in 1991 came up with a conservative version namedMD5
[10] to replace the earlier Hash MD4. MD5 became a milestone in the development of Hash. It was a widely- used well-known 128-bit iterated hash function, used in various applications including SSL/TLS, IPsec, and many other cryptographic protocols. It was also commonly-used in implementations of time stamping mechanisms, commitment schemes, and integrity- checking applications for online software and random- number generation. Type-2 (Semi free start collision) and Type-3 (Pseudo collision) attacks on MD5 were reported in [47, 50]. Strong collisions (Type-1 collisions) on MD4 and MD5 have been reported by Wang *et. al.* in [30, 31, 32] and these attacks make the further usage of these hash functions questionable.

SHA family of Hash Functions: Secure Hash Algorithm (SHA) developed by the National Institute of Standards and Technology (NIST) was also designed on the same principle as MD4 and was published as Federal Information Processing Standard (FIPS 180)in 1993 [4]. A revised version was issued as FIPS180-1 in 1995 and is generally referred to as SHA-1 [5]. When revised version of SHA-1 was published no details of the weaknesses found in SHA-0 (originally SHA) were provided [33]. SHA-1 produces a hash value of 160 bit. In 2002, NIST produced a revised version of the standard known as FIPS180-2

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

[6] and defined three new versions of SHA with digest lengths of 256, 384 and 512 and known as SHA-256, SHA-384, and SHA-512 respectively. So total SHA versions becomes four including SHA-1 (160 bit). In October 2008, FIPS 180-2 has been replaced by FIPS 180-3 [8] and in new standard SHA-224 has been added which is same as other SHA algorithm producing 224 bits of message difest. All these SHA versions are based on the same principle of MD4 and hash length has changed and certain other improvements have been carried from one version to next. Attacks on SHA-0 and SHA-1 have been reported in [33, 34, 35]. Till date no practical attack has been reported on SHA-2.

RIPEMD family of Hash Functions: RIPEMD family of hash functions consists of RIPE MD, RIPEMD-128, RIPEMD-160, RIPEMD-256, RIPEMD-320. RIPE MD, a 128 bit hash function, based on MD4 algorithm, was developed in the framework of the EU (European Union) project RIPE (RACE Integrity Primitives Evaluation) by Hans Dobbertin, Antoon Bosselaers, Bart Preneel.. RIPEMD- 160 [87] was an improved version of RIPE MD. The 128 bit version was intended only as a drop-in replacement for the original RIPEMD, which had been found to have questionable security. The 256 and 320 bit versions diminish chance of accidental collision, and don't have higher level of security compared to RIPEMD-160. A collision on RIPEMD was reported in [30] but that does not affect RIPEMD-160. Till date no practical attack has been observed on RIPEMD- 160.

HAVAL Hash functions Yuliang Zeng, *et. al* invented HAVAL hash function in 1992 [86]. To certain extent it takes the motivation from MD4 hash function only. However HAVAL can produce hashes of different length i.e. 128, 160, 192,224 or 256 bits. In addition, HAVAL has a parameter that controls the number of passes a message block (of 1024 bits) is processed. A message block can be processed in 3, 4 or 5 passes. By combining output length with pass, authors provided fifteen (15) choices for practical applications where different levels of security are required. Algorithm was designed for 32-bit computers Experiments showed that HAVAL is 60%faster than MD5 when 3 passes are required, 15% faster than MD5 when4 passes are required, and as fast as MD5 when full 5 passes are required. Research has uncovered weaknesses which make further use of HAVAL (at least the variant with 128 bits and 3 passes) questionable. The strong collision attack on HAVAL was reported by Wang et. al. in [31].

All the above dedicated hash functions are somehow designed with motivation from MD4 algorithm only and thus are sometime collectively known as MDx type hash functions

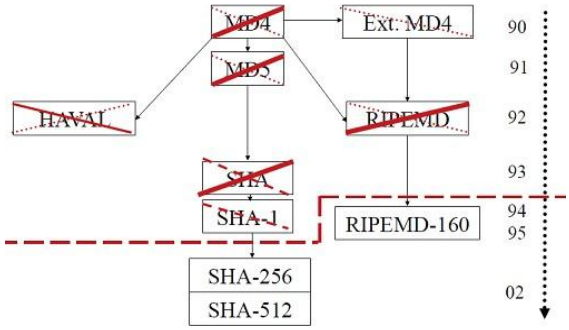


Fig. 4MDx-type hash function history [106]. Vertical line refer year when hash function was invented and functions Crossed with red lines have been attacked

Few other famous dedicated hash functions reported in literature are SNEFRU [88], Tiger [89], JH [90], Keccak[46], Blake 91]. **Snefru**; designed by Ralph Merkle in 1990, like Khufu and Khafre block ciphers was an Egyptian Pharaoh. Snefru's initial design as well as modified design has been shown to to be insecure against differential cryptanalysis [93]. **Tiger** hash function was designed by Anderson and Biham in 1995mainly for 64-bit platforms. It is quite efficient on Software but because of its inherent use of large S- Boxes, implementation in hardware or small microcontrollers is difficult. Tiger hash function is frequently used in Merkle Hash tree form, where it is referred to Tiger Tree hash (TTH). TTH is used by many clients on Direct Connect and Gnutella file sharing networks. The last two in the list i.e. JH, Keccakand Blake are among the five finalists in the NIST hash function competition [45] to design SHA-3 standard. JH hash function makes use of S-boxes and is well suited for bit slicing. Keccak on the hand make use of sponge construction as detailed in Section 4. Blake does not fit exactly into the category of dedicated hash functions because it is based on ChaCha Stream Cipher.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

Few Other approaches

There has been few hash functions that have not been based on existing cryptographic primitives like block ciphers or modular arithmetic but rather are based on some hard problems like knapsack problem, cellular automata or Discrete Fourier transformations. Hash function based on knapsack was proposed by Ivan Damgard in but the same was shown to be broken in.Cellular automata-based hash function was proposed in by Wolram and in by Daeman*et.al.*Claus Schorr has proposed hash functions based on discrete Fourier transformations called FFT- hash. Three modifications of FFT-Hash have been proposed. First two modifications, FFT-Hash I and FFT – Hash II, was broken few weeks after the proposal. Third modification is quite slower. As a whole, all these approaches (based on knapsack or cellular automata or FFT) have not found much success and are not generally used these days.

VI. CURRENT SCENARIO: PROGRESSING TO SHA-3

The current Secure Hash Standard as developed by NIST (National institute of Standards and Technology) is FIPS 180-3. This standard suggests five hash functions SHA-1, SHA -224, SHA-256 SHA-384, and SHA-512. All these are dedicated hash functions as explained in Section 6 and to certain extent are based on MDx family. The practical attack on MDx family, followed by attack on SHA-0 and SHA-1 has been discussed in section 6.3. Majority of these attacks have been carried out in year 2004 and 2005 by a team of researchers from the Shandong University in Jinan China, led by Xiaoyun Wang. The same team also broke HAVAL-128 and RIPEMD. Looking at the variety of hash functions attacked by this team, it seemed likely that their approach may prove effective against all cryptographic hashes in the MD family, including all variants of SHA.

Burr from US National Institute of Standards and Technology in his paper reviewed the scenarios of Cryptographic Hash Functions. Burr pointed out that with SHA-1 and SHA-2 in its cryptographic toolkit, NIST had hoped to be done with hash functions for a long time. Aside from a near break of MD5 by Dobbertin in 1996, researchers made little progress in hash function analysis until mid-2004.

Since then, Wang, AntonineJoux, and Eli Biham have attacked nearly all the early hash functions, including SHA-1. Given that SHA-2 functions are in the same family as the earlier broken functions, these attacks shook cryptographers' long term confidence in nearly all hash functions designed to date. Cryptographers have learned much about hash functions and how to attack them in the past couple of years, and yet cryptanalysts 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 (2^{112}, 2^{128}, 2^{192}, and 2^{256} operations for SHA-224, SHA-256, SHA-384, and SHA-512, respectively) [104].

Hoch and Shamir in year 2006 [105], studied the multi collisions on Iterated Concatenated Expanded (ICE) Hash Functions. Hoch and Shamir extended the idea presented by Joux. Joux in 2004 showed that in any iterated hash function it is relatively easy to find exponential sized multi-collisions, and thus the concatenation of several hash functions does not increase their security. But Joux [31] Attack does not work on ICE i.e. when in addition to Iterated and Concatenated Hash Function technique message Expansion is also added i.e. each iterated function process message block more than once. Hoch *et al.*[105]considered the general case (ICE) and proved that even if we allow each iterated hash function to scan the input multiple times in an arbitrary expanded order, their concatenation is not stronger than a single function. Finally, authors extended their result to tree- based hash functions with arbitrary tree structures. Hoch *et al.* showed that a large class of natural hash functions (ICE and its generalization TCE) is vulnerable to a multi-collision attack, and hoped that the techniques developed here will help in creating multicollision attacks against even more complicated types of hash functions. **Such a conclusion was perhaps hinting to probable attack on SHA 2 family of hash functions.**

Looking at the current scenarios, In **Nov 2007** NIST (National Institute of Standards and Technology) announced a **public competition** [45] to develop a new cryptographic hash algorithm to replace the older SHA-1 and SHA-2. The competition was NIST's response to advances in the cryptanalysis of hash algorithms. The winning algorithm will be named "SHA-3", and will augment the hash algorithms currently specified in the Federal Information Processing Standard (FIPS) 180-3, Secure Hash Standard [8]. As per NIST website "NIST is initiating an effort to develop one or more additional hash algorithms through a public competition, similar to the development process for the Advanced Encryption Standard (AES)." [45]

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

By October 31, 2008, NIST received **sixty-four** entries; and **selected fifty-one** candidate algorithms to advance to the **first round on December 10, 2008**, and fourteen advanced to the **second round on July 24, 2009**. A year was allocated for the public review of the fourteen second-round candidates. NIST received significant feedback from the cryptographic community. Based on the public feedback and internal reviews of the second-round candidates, NIST selected five SHA-3 finalists – **BLAKE [91], Grøstl [92], JH [90], Keccak [46],** and **Skein [66]** to advance to the third (and final) round of the competition **on December 9, 2010**, which ended the second round of the competition. A one-year public comment period is planned for the finalists. NIST also plans to host a final SHA-3 Candidate Conference in the spring of 2012 to discuss the public feedback on these candidates, and select the SHA-3 winner later in 2012.

VII. BLAKE – IMPLEMENTED IN OUR PROJECT

BLAKE2 is a cryptographic hash function **faster than MD5, SHA-1, SHA-2, and SHA-3**, yet is at least as secure as the latest standard SHA-3. BLAKE2 has been adopted by many projects due to its high speed, security, and simplicity. BLAKE2 comes in two flavors:

BLAKE2b (or just BLAKE2) is optimized for 64-bit platforms—including NEON-enabled ARMs—and produces digests of any size between 1 and 64 bytes

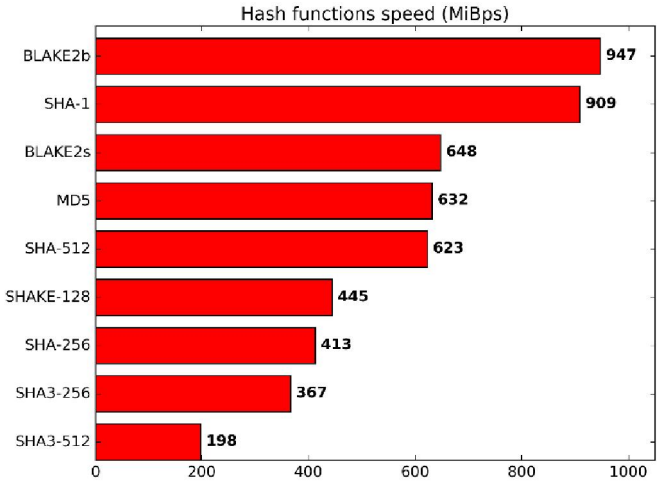
BLAKE2s is optimized for **8- to 32-bit** platforms and produces digests of any size between 1 and 32 bytes

BLAKE2 includes the 4-way parallel **BLAKE2bp** and 8-way parallel **BLAKE2sp** designed for increased performance on multicore or SIMD CPUs. BLAKE2 offers these algorithms tuned to your specific requirements, such as **keyed** hashing (that is, MAC or PRF), hashing with a **salt**, updatable or incremental **tree-hashing**, or any combination thereof.

BLAKE2 also includes the **BLAKE2x** variants, which can produce digests of arbitrary length.

BLAKE2 shines on 64-bit CPUs: on an Intel Core i5-6600 (Skylake microarchitecture, 3310MHz), BLAKE2b can process **1 gibibyte per second**, or a speed rate of 3.08 cycles per byte.

The plot below shows how BLAKE2 outperforms MD5, SHA-1, SHA-2, and SHA-3 on a Skylake Intel CPU (speeds are for hashing using a single core; using multiple cores, BLAKE2 can be even faster):



Specifications

b2sum

BLAKE2 is based on the SHA-3 proposal BLAKE, designed by Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and Raphael C.-W. Phan. BLAKE2, like BLAKE, relies on a core algorithm borrowed from the ChaCha stream cipher, designed by Daniel J. Bernstein.

Users - Non-exhaustive list of systems using BLAKE2:

- Linux kernel RNG: The Linux kernel's RNG uses BLAKE2s as its entropy extractor
- OpenSSL: OpenSSL includes BLAKE2b and BLAKE2s

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

- WireGuard: The WireGuard VPN uses BLAKE2s for hashing and as a MAC
- WolfSSL: WolfSSL includes BLAKE2b
- Botan: The Botan library includes BLAKE2b
- Crypto++: The Crypto++ library includes BLAKE2s and BLAKE2b
- Noise: The Noise protocol (used in WhatsApp and WireGuard) uses BLAKE2s and BLAKE2b
- Cifra Extrema: Cifra Extrema products use several versions of BLAKE2 in its servers and satellite modules
- Bouncy Castle: Includes BLAKE2b-160, BLAKE2b-256, BLAKE2b-384, and BLAKE2b-512
- Peerio: BLAKE2s is used to generated IDs and for integrity checks
- 8th: BLAKE2s is the default hash in the 8th cross-platform development system
- librsync: BLAKE2b is the default hash un this popular remote delta-compression library
- checksum: BLAKE2s is one of the three hash functions supported with MD5 and SHA-1
- Password hashing schemes:
- Argon2 (by Biryukov, Dinu, Khovratovich; PHC winner)
- Catena (by Forler, Lucks, Wenzel; PHC candidate)
- Lanarea (by Mubarak; PHC candidate)
- Lyra and Lyra2 (by Simplicio Jr., Barreto, Almeida, Andrade; PHC candidate)
- Neoscrypt (by Doering)
- RIG (by Chang, Jati, Mishra, Sanadhya; PHC candidate)
- TwoCats (by Cox; PHC candidate)
- Yarn (by Capun; PHC candidate)

VIII. CONCLUSION

In this paper, we have shown how cryptographic hash functions slowly gained its importance in the field of cryptology. We have made all attempts to give a complete picture of cryptographic hashes, its design techniques and vulnerabilities. This paper would really help budding researchers who would take up research in this particular field

REFERENCES

[1]. D. Kahn,TheCodebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet, Scribner, 1996.
[2]. W. Diffie, and M. Hellman, “New Directions in Cryptography”,IEEE Transactions on Information Theory, vol. 22, No. 6, 1976, pp. 644-654.
[3]. B. V. Rompay, “Analysis and Design of Cryptographic Hash functions, MAC algorithms and Block Ciphers”, Ph.D. thesis, Electrical Engineering Department, KatholiekeUniversiteit, Leuven, Belgium, 2004.
[4]. FIPS 180, Secure Hash Standard (SHS), National Institute of Standardsand Technology, US Department of Commerce, WashingtonD. C., 1993.
[5]. FIPS 180-1, Secure Hash Standard (SHS), National Institute of Standards and Technology, US Department of Commerce, WashingtonD. C.,1995.
[6]. FIPS 180-2, Secure Hash Standard (SHS), National Institute of Standards and Technology, US Department of Commerce, WashingtonD. C.,2002.
[7]. FIPS 197, Advanced Encryption Standard, National Institute of Standards and Technology, US Department of Commerce, WashingtonD. C.,2001.
[8]. FIPS180-3, Secure Hash Standard (SHS), National Institute of Standards and Technology, US Department of Commerce, Washington D. C., 2008.
[9]. R. Rivest, “The MD4 Message Digest Algorithm”, IETF RFC 1320, 1992.
[10]. R. Rivest, “The MD5 Message Digest Algorithm”, IETF RFC 1321, 1992.
[11]. S. Lucks, “Design Principled for Iterated Hash Functions”, in IACR Cryptology ePrint Archive, 2004, pp. 253.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 1, June 2023

[12]. G. Bertoni, J.Daemen, M. Peeters, and G. Van Assche, "Cryptographic Sponges", [online] http://sponge.noekeon.org/.
[13]. National Institute of Standard and Technology (NIST): Cryptographic Hash Algorithm Competition. [online] http://csrc.nist.gov/groups/ST/hash/sha-3/
[14]. [46]G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche, "The Keccak Reference", Submission to NIST (Round 3),2011.[online] http://csrc.nist.gov/groups/ST/hash/sha-3/Round3/submissions_rnd3.html.
[15]. B.den Boer, and A. Bosselaers, "Collisions for the compression function of MD5", in EUROCRYPT, 1993, pp. 293-304.
[16]. L. Knudsen. "Block Ciphers: Analysis, Design and Applications", Ph.D.thesis, Aarhus University, Aarhus,
[17]. Denmark, 1994
[18]. O. Mikle, "Practical Attacks on Digital Signatures Using MD5 Message Digest", IACR Cryptology ePrint Archive, 2004, pp.356.
[19]. H. Dobbertin, "Cryptanalysis of MD5 compress", inEUROCRYPT, 1996
[20]. KUL15-RMC- 1.0, private communications, 2006.
[21]. E. Andreeva, G. Neven, B. Preneel, and T. Shrimpton, "Seven-Property-Preserving Iterated Hashing: ROX", IACR Cryptology ePrint Archive, 2007, pp.176.
[22]. M. Bellare, and T. Ristenpart, "Multi-Property- Preserving Hash Domain Extension and the EMD Transform", in ASIACRYPT, 2006, pp.299-314 .
[23]. T. Duong, and J. Rizzo, "Flickr's API Signature Forgery Vulnerability", 2009 [online] http://netifera.com/research/flickr_api_signature_forgery.pdf
[24]. B. Kaliski, and M. Robshaw. "Message Authentication with MD5". RSA Labs' CryptoBytes, Vol. 1, No. 1, Spring 1991.