Analysis and Performances of Different Hashing Algorithms

S Govinda Rao
Associate Professor in CSE
GRIET,Hyderabad

ABSTRACT
Now a day with the invention of new technologies communication of information is very fast and quickly sending of information even for long distances is made easy in the world. But this online communication had brought many vulnerabilities like authenticity and integrity of data. Though there are many cryptographic ciphers, they are vulnerable to different types of attacks. In this paper, the authors focus on different types of attacks and how authenticity and integrity of data is lost. The different algorithms that are used to maintain the authenticity and integrity of data and the comparative study on different algorithms with their vulnerability are considered. The main focus is done with respect to hashing techniques.

KEYWORDS: MD5, SHA, Message Digest, Hash value, Hashing.

INTRODUCTION
Cryptography is a Greek word which mean secret writing. Cryptography is defined as a science of writing secret message. The process of enciphering and deciphering of messages is called cryptography. On who cracks the cryptographic messages is called cryptanalysis. The process of doing both writing the secret messages and cracking the secret message is cryptology which is treated as a separate field of study. Cryptography have many uses like encryption, decryption, authentication, digital signatures, hashing and many more .
Encryption is the process of converting a plain text to cipher text with the intention of nobody can understand the coded message. Decryption is the process of converting a cipher text to plaintext with the intent of generating original message. Decryption is a reverse process of encryption. Authentication is the process of providing a proof that something is genuine, real, true, and authentic. Digital signatures are used for online validation and verification of something is genuine which is authentication online. Hashing is a technique which accepts a variable length message as input and produces a fixed length and unique string. It is very difficult to have a reverse process to generate original message. In cryptography whatever the new techniques are adopted to secure the data, by one or other way that techniques are venerable to various types of attacks. The security mechanisms in cryptography are encipherment that is encryption and decipherment that is decryption. The other mechanism of security in cryptography is hashing mechanism which uses a hash function and generates the hash values.
The security of cryptographic algorithms is dependent on various key ingredients like Plaintext, ciphertext, encryption and decryption process, the length of the key. If the length of the key is more then that algorithm is considered to be more secure. There are two ways of using the key in cryptography. One is using the same key for encryption and decryption process which is called as symmetric key cryptography. Other is using different key one is for encryption and other is for decryption which is called as asymmetric key cryptography.

RELATIVE STUDY

Cryptographic algorithms enable a secure communication between two communicating parties without eavesdropping the third-party. Cryptography ensures that the recipient of message by receiver that the message is not tampered, modified, added, or deleted the message. It protects it from unauthorized access. In order to prevent from unauthorized access, we generally use passwords. Passwords play a very important role in our online communication where we need to get authenticated. But as the advent of new technologies in hardware and software the cracking of passwords are possible or the passwords are theft. So there are various types of hashing techniques where we can hash our passwords and create a secure password. All the cryptographic hash algorithm uses a hash function which ensures the integrity of transmitted data or stored data. A hash function accepts a message or data of variable length and converts it into a fixed length data. This is called as message digest. These message digest is treated as signature of the message. The hash functions have the following properties:
1) One-way function: it is computationally infeasible to generate a message from its hash value because it is only one-way function and no reverse process is available. Only one option available is trying all the possibilities of message. 2) Weak collision resistance: it is computationally infeasible to compute a hash value where two hash values are same and 3) Strong Collision resistance: it is computationally infeasible to compute a pair of hash value where two hash values are same[1].

In this section we study about various different types of hashing algorithm.

MD-2 It is a cryptographic hash algorithm which generates a message digest of 128 bits. It was published in august 198 vulnerable various types of attack like pre-image attack and collision attacks. This MD2 algorithm is no longer considered as secure algorithm.

MD-4 This cryptographic hash algorithm, generates a fixed 128 bits message digest. It takes 48 rounds of its compression function. It was published in 1990.

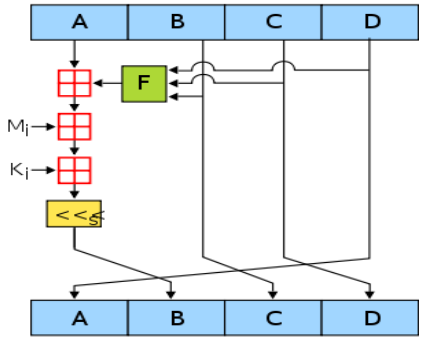

Fig 1: Single round of MD4

A collision attack published in 2007 can find collisions for full MD4 in less than 2 hash operation.

MD-5 MD-5 generates a message digest of fixed 128 bits. It takes 64 rounds. It was published in 1992. Its design is very popular as it is used in many hashing algorithms. The design is given by Markle-Damgard. It is called as Markle-damgard construction or Merkle-damgard hash function.

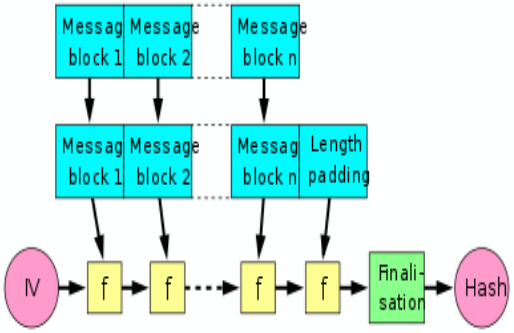

Fig 2:Markle-damgard construction or Merkle-damgard hash function

A 2013 attack by Xie Tao, Fanbao Liu, and Dengguo Feng breaks MD5 collision resistance in 218 times.

MD6: MD6 is designed by Ronald Rivest in the year 2008. It has a Merkle tree like structure. It allows a parallel computation of hashes for very long inputs. Merkle tree is also called hash tree. In Merkle tree every leaf node is labelled with the hash of the data block and every non leaf node is with cryptographic hash.

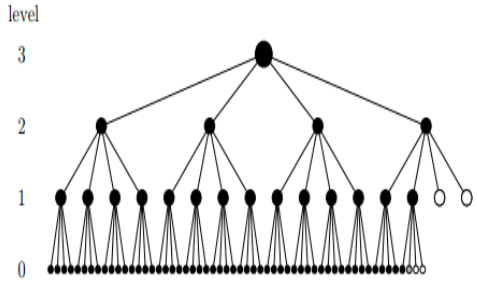

Fig 3: standard structure of MD6

SHA-0 SHA-0 belongs from SHA family; it is another cryptographic hash algorithm generates a message digest of fixed 160 bits. It takes 80 rounds. It was published in 1993. A 2004 attack by Bihamet.Al breaks SHA-0 collision resistance at 241.

SHA-1 SHA-1 generates a message digest of 160 bits. It takes 80 rounds and was published in 1995. It is the most widely used algorithm for integrity. Reason for its popularity among existing algorithms is its time efficiency and its robustness. Later on, a 2011 attack by Marc Stevens can produce hash collisions with a complexity of 261 operations.

SHA-2 Secure Hash Algorithm (SHA-2) is a cryptographic hashing algorithm that was developed by NSA. It has two variations namely SHA-256 and SHA-512. The primary difference between the two variants are the size of the words used. While SHA-256 uses 32-bit words, SHA512 uses 64-bit words. Although, neither SHA-256 nor SHA-512 have been proved to be flawed, they are still not preferred for integrity verification as they are not as efficient as SHA-1 in terms of time complexities. Also, as SHA-2 is derived from SHA-1 which in turn is based on the MerkleDamgård structure, that was exploited to break the SHA-1 cryptographic hashing algorithm, thus, theoretically SHA-2 can also be broken.

SHA-3 Secure Hash Algorithm 3 (SHA-3) is a cryptographic hashing algorithm that was chosen by the NSA in 2012 after a public competition among non-NSA designers. The prior name of the SHA-3 hashing algorithm prior to the results of the competition was keccak. When keccak emerged as the winner of the SHA-3 competition, it was renamed to SHA-3. While SHA-3 supports the same hash lengths as SHA-2, the internal structure very different and is invulnerable to attacks like length extension which both the MD5 and SHA-1 were proved to be susceptible to. The main reason for the creation of the SHA-3 algorithm is due to the theoretical attacks that are possible against SHA-2. While there no practical proof has been submitted exposing the flaws of SHA-2, one cannot deny that it is indeed possible [3].

COMPARATIVE STUDY:

Table 1: Secure Hash Algorithm Properties [10]

srno	Algorithm	Message size(bits)	Block Size (bits)	Words Size(bits)	Message Digest Size (bits)	Rounds
1	SHA -1	$<2^{64}$	512	32	160	80
2	SHA - 254	$<2^{64}$	512	32	224	64
3	SHA - 256	$<2^{64}$	512	32	256	64
4	SHA - 384	$<2^{128}$	1024	64	384	80
5	SHA - 512	$<2^{128}$	1024	64	512	80
6	SHA – 512/224	$<2^{128}$	1024	64	224	80
7	SHA – 512/256	$<2^{128}$	1024	64	256	80
8	MD2	$<2^{128}$	128	32	128	18
9	MD4	$<2^{128}$	512	32	128	3
10	MD5	$<2^{128}$	128	32	128	64
11	MD6	$<2^{64}$	1024	644	512	80
12	BLAKE-256	$<2^{64}$	512	32	256	12 or14
13	BLAKE-224	$<2^{64}$	512	32	224	12 or14
14	BLAKE-384	$<2^{128}$	1024	64	384	12 or14
15	BLAKE-512	$<2^{128}$	1024	64	512	12 or14

Table 2: Secure Hash Algorithm Properties [11]

srno	Algorithm	Collision status	speed	security	Successful attacks reported
1	SHA -1	YES	Slower, 80 iteration	More secure	YES
2	SHA - 254	YES	Slower,	More secure	YES
3	SHA - 256	YES	Slower,	More secure	YES
4	SHA - 384	YES	Slower,	More secure	YES
5	SHA - 512	YES	Slower,	More secure	NO
6	SHA – 512/224	YES	Slower,	More secure	NO
7	SHA – 512/256	Theoretical	Slower,	More secure	-
8	MD2	YES	Faster	Less secure	YES
9	MD4	YES	Faster	Less secure	YES
10	MD5	YES	Faster	Less secure	YES
11	MD6	NO	Faster, 60 iterations	More secure	NO
12	BLAKE-256	NO	Faster than SHA and MD5	More secure than MD5 and similar to SHA-3	NO
13	BLAKE-224	NO	Faster	More secure thanMD5 and SHA -3	NO
14	BLAKE-384	NO	Faster	More secure thanMD5 and SHA -3	NO
15	BLAKE-512	NO	faster	More secure thanMD5 and SHA -3	NO

RESULTS:

Table 3: calculation of Hash values for message [12]

srno	Algorithm	Hash value	Message Digest Size (bits)
Message :		welcome to hash coding	
1	SHA -1	ac5a02a45bb4af337236a052be798f694f8b7100	160
2	SHA - 224	2df461c9c64dbdbc44b8c76a6901737d6a7bf15f 197df072c204b471	224
3	SHA - 256	e986d508289222244d3a1a4b8a4b9c000d777e19 ae989afdc5b0aef4ab99d5cc	256
4	SHA - 384	5848b9b8cc830b6db1d630fcb4fe73e1f6b4293c4c 3ba570e8e33ee9ab656c6dfa289dbf6d43657daf7da 2cbdf4b363f	384
5	SHA - 512	da06fdbd388edac722a8f601e3313b2971f0c58d34d be06ecf383969615db673e0df7e42242c22a4423b82c aca1dfd968a861f4949cdc36be7c6a2859e63b405	512
6	SHA – 512/224	fbc5b8100d70fea2a68399b2eca93d1c8d9421ac2868 43cb5aa91e88	224
7	SHA – 512/256	b00601bca5a5654399fcb11f8ad3f00b0f1335f78890b 258b3b8e0df4f5c45ff	256
8	MD2	48a794cd1a51c25a62bdfd82a08c92ce	128
9	MD4	0ccf5a534fad1341316e0dc8440c21af	128
10	MD5	75b62e1511bc007ab2086b03a90efd38	128
11	MD6-128	019a04c9b088aac7599ce88d20346310	512
12	MD6-256	6b1563ea279d705e072872a47f0683df07ea9c8bb7d382 78ee5e63db1a2fe26a	256
13	MD6-512	dd15f90462b9981bb01e7e2942d6133a8d3ac970a0914ea 8cc4776850667358133b32a0f23782f767e76359228262117c534218772db4f0451a04a6e0e868010	512

Table 4 : calculation of Hash values for message [13]

srno	**Algorithm**	**Hash value**	**Message Digest Size (bits)**
Message :		**welcome to happy hash coding in cryptography**	
1	SHA - 1	409e3cc2ddab6333c2fb7251853a5e1f02663ab2	
2	SHA - 224	50e9fe69ecc7a192c9e9ff509051f5a992551fa171a05166390d3f3b	
3	SHA - 256	05e514a69f70be961ec5b49741e7902563ec81876185f74ee93efdaa02a29370	
4	SHA - 384	9b35a62163b563f9825b9d692bb35ad4f840612e09dc4fa64189a194890248320fb9746e75f8fe4296119e0ecb4d7c2c	
5	SHA - 512	af8d95a92c70a3af9d1affd106c12fcd1a4f2d744c821942bb9d360dc3e5d2d9fe7dce11338dd2471a7e7e1f6e152b5006a7def46b3cf58264210d8c2a3cde2c	
6	SHA – 512/224	195eb76e453d3be46f6d8a1b78920d6228c309d0d4d8a679f51c1333	
7	SHA – 512/256	2196a8448e4e7a0cc3bdba5b5f96dcbed9772ced18493c2e3b31f0f4c073df11	
8	MD2	3919aae8c8032cf62ea601daf370abc0	
9	MD4	d9d557d05f5ef0bf0e3416dd60cb7c05	
10	MD5	e8ec8ea5855f25e397181c7d9b7f3865	
11	MD6-128	3ade4b0ee28ae9ff87f36edad72ff763	
12	MD6-256	944bd8937ae39d9f7ff571507ce6ddd1b76448edafb3820fa137f0f76a4b272d	
13	MD6-512	40329529cfeec63800f75ba5c092d317e27eea2cff96591971144fd5e895f67ab191e02c372b130539099c3099e56722f15ad60bdb289adeea19a978a5fc4b5f	

[ ]https://www.browserling.com/tools/all-hashes

Issues with the above hashing algorithms

From the comparative study of various hashing algorithm it is found that as the length of hash increases then that algorithm is more secure, as its hash length is more and it becomes difficult to generate and crack the hash values. In order to make all the hash algorithm there are different version of the algorithm are upgraded.

Each algorithm is vulnerable to some type of attack. The authors Neha Kishore and Bhanu Kapoor.[2], Have surveyed the literature related to cryptographic Hash function [CHF] role of CHFs. They also studied the different types of attacks on SHA family algorithms. Their third part of student includes the parallelization of CHFs on both hardware and software's and proved that MD6 algorithm is computationally faster when compared with other algorithms.

Though there are various types of algorithms to generate a hash value and secure the process of authentication with the help of password and other method. But, still there is need of securing the password or hash values that are generated by any Hashing algorithm. The reason is that when a hash for a password is generated it must be stored in a database. If anyone who gets an access to database by unauthorized way where the hashed password is stored, he is unable to get back the original password. This is due to one-way property of hash. But, still there is a problem of generating the same password. When a user tries to get logon, the personalized module gets the password and performs a similar one-way hash and generated the hash value which is similar to one stored in the database. By doing this process he/she is able login successfully. These is due to the password is not encrypted before storing in to the database.

What is a secure password hash? A secure password hash is one which is encrypted and stored in database by applying certain cryptographic hash algorithms.

What if a user forgets his own password? When a user forgets his own password, where he/she gets a temporary password to set the new password but the same password is not provided. Here user can generate the same password again but system is not able to convert the hashed password to its original password as there is no reverse process of hashed passwords[5]. This is the most common techniques adopted by much application.

Even the secure algorithm MD5 is widely used, but it is also susceptible to brute-force and dictionary attacks.

What is a **brute force attack**? This attack is nothing but trying all the possibilities until we get a correct match of password (which is exhaustive search).

Dictionary attack: This attack is similar to brute force attack where a cracker tries all the word in the dictionary to crack a password. This dictionary attack is used by many cryptanalyst in computer security to find the decryption key or crack authentication mechanism. A good dictionary is that which contains more words whose possibilities of matching with password is more. A good dictionary can crack a password very fast.

Rainbow tables: Rainbow table is that which maintain the table of password and their hashes. It is a linked list of hashes. All the computed hashes are stored as chain so that it can be used for reverse cryptographic hash function. In order to crack any password it first generates a hash and looking for its hash value in the Rainbow table to find the corresponding Plaintext.

Solution for above types of attack

From the above types of attack it is clear that whatever the hash value is generated by any type of hashing algorithm but they are vulnerable to certain types of attack with some collision complexity. For example SHA 512 which is considered to be more secure is also vulnerable to attack with collision complexity of 57/80 rounds with 2^{511}complexity.

The SHA and MD5 algorithms are compared for performance and security point of view by many authors. It is concluded that MD5 algorithm is faster than the SHA - 512 algorithms but the security point of view the SHA-512 is more secure [4]. So these algorithms are considered for more secure hash generation purpose and performance purpose. There are different types of solution are provided to secure the hash in the database.

One of the solutions for the different types of attacks faced by the authentication algorithm is to do **salting. A salt** is a randomly choosen text which is added to the password before generating the hash of the password and storing in the database [6]

The main advantage of salting is that

1. It ensures that the outputs of all the hashes with some inputs are not same.
2. It is important when same password is used for a prolonged period of time at different places, it makes more secure by generating random passwords differently.
3. It make passwords more complex to crack and difficult to crack them.
4. It protects against rainbow table, dictionary and brute force types of attacks.
5. It slows down the process of comparison of hashes against the hashes of password guess.

Working procedure of password salting:

Usually the passwords are stored in databases as plaintext. Since passwords are stored in database and same password is used at different places, then that password is vulnerable to attack. The use of salting the password helps to protects from the attack by storing the salted password as hash in the database along with salt. Now it is difficult to know the password of the user as the password is not stored in the database. Even if the user uses the same password for different places. because every time user logon to the system the salt and the entered password are combined to generate the password. This generated password should be matched with the password that is stored in the database. If both the passwords and hash stored is matched then only authentication is done.

TABLE: EXAMPLE OF SALTING

Password	Salt	String for hash	Algo.	HASH
Hello123	ABCDEF0123456789	Hello123+ABCDEF0123456789	MD5	446f07ef8db3e9d76557633173238099
Hello123	AB12CD34EF567890	Hello123+AB12CD34EF567890	MD5	46c9daa89f149be2634280f9f5cd1e9c
Hello123	ABCDEF0123456789	Hello123+ABCDEF0123456789	SHA-1	1bc7ccfcf273392ab66aed96315706414811320e
Hello123	AB12CD34EF567890	Hello123+AB12CD34EF567890	SHA-1	760530f5486ad7c9884fd75acfefed2f8243ea4f

From the table password is same and the salts are different, the string given for hashing is combination of both password+salt and is given to any desirable hashing algorithm (here MD5 and SHA-1) to generate the Salted Hash.

Other solution for the different types of attack faced by authentication algorithms is using PBKDF2 with HmacSHA1 algorithm. From password salting and creating

secure hash it is possible to secure the password but, as the technology of hardware is also rapidly growing because of which any password hash can be cracked with brute force type of attack, rainbow table attack, and dictionary attacks. To solve this problem, a common idea of making this types of attacks slower that is to create method which make all the different types of attack slower. This feature can be implemented by using CPU intensive algorithms like **PBKDF2, Bcrypt** or **Scrypt**. These algorithms take the security factor as the number of iteration count. If the hardware configuration is increases next year then its iteration argument is going to be increased as challenge against the hardware. This iteration count argument makes the algorithm slower[7].

PBKDF2: (Password-Based Key Derivation Function 2) PBKDF2 is part of RSA Laboratories Public-Key Cryptography Standards (PKCS) series, RFC 8018 published in the year 2017 recommends PBKDF2 for secure password hashing. The main operation of PBKDF2 is as a Pseudorandom function. PBKDF2 accepts the input string as password and salt repeatedly to create a derived key. This key can be used as cryptographic key for encryption and decryption [8]. **Bcrypt** or **Scrypt**: These two concepts are similar to PBKDF2. The main of these functions is to make the attacker tougher to crack the password.

Conclusion:

The authors in this paper studied about different hasing algorithms and concludes that, as the length of the hash increase the security of hash increase. It is found that MD5 algrithm is computationally faster than the SHA 512 algorithm. But the secure hash algorithm provide better security than Message Digest algorithm. It is not secure to use only text only password because they can easily cracked. MD5 algorithm is basically secure but adding salting to it makes it more stronger and secure. SHA 512 is considered more secure but, as the advent of hardware technology, it is easily possible to crack the hashes generated by it. To overcome this problem and to make the attacks slower various attack slowering techniques are considered PBDKF2, B Crypt and SCrypt.

REFERENCES

[1] S. William and W. Stallings, "Cryptography and Network Security, 4/E," 2006

Kishore, N., & Kapoor, B. (2016). Attacks on and advances in secure hash algorithms. IAENG International Journal of Computer Science.

[3] https://en.wikipedia.org/wiki/SHA-1

[4] Aggarwal, S., Goyal Astt Professor, N., & Aggarwal Astt Professor MRCE, K. (2014). A review of Comparative Study of MD5 and SHA Security Algorithm. International Journal of Computer Applications.

[5] Aggarwal, S., Goyal Astt Professor, N., & Aggarwal Astt Professor MRCE, K. (2014). A review of Comparative Study of MD5 and SHA Security Algorithm. International Journal of Computer Applications.

[6] https://docs.oracle.com/cd/E26180_01/Platform.94/ATGPersProgGuide/html/s0506passwordhashing01.html

[7] https://www.commonlounge.com/discussion/eb7138e22a5f49aca8b388e4a475f040.

[8] https://howtodoinjava.com/security/how-to-generate-secure-password-hash-md5-sha-pbkdf2-bcrypt-examples/

[9] https://en.wikipedia.org/wiki/PBKDF2

[10] Department of Commerce United States of America. (2015). Secure Hash Standard (SHS) - FIPS PUB 180-4. Federal Information Processus Standards Publication. http://doi.org/10.6028/NIST.FIPS.180-4

[11] Commerce, U. S., & Technology, N. I. and. (2012). Secure Hash Standard (SHS). Federal Information Processing Standards Publication 180-4. http://doi.org/10.6028/NIST.FIPS.180-4

[12] Commerce, U. S., & Technology, N. I. and. (2012). Secure Hash Standard (SHS). Federal Information Processing Standards Publication 180-4. http://doi.org/10.6028/NIST.FIPS.180-4

[13] https://www.browserling.com/tools/all-hashes