World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:11, No:6, 2017

Secure Hashing Algorithm and Advance Encryption Algorithm in Cloud Computing

Jaimin Patel

***Abstract*—**Cloud computing is one of the most sharp and important movement in various computing technologies. It provides flexibility to users, cost effectiveness, location independence, easy maintenance, enables multitenancy, drastic performance improvements, and increased productivity. On the other hand, there are also major issues like security. Being a common server, security for a cloud is a major issue; it is important to provide security to protect user's private data, and it is especially important in e-commerce and social networks. In this paper, encryption algorithms such as Advanced Encryption Standard algorithms, their vulnerabilities, risk of attacks, optimal time and complexity management and comparison with other algorithms based on software implementation is proposed. Encryption techniques to improve the performance of AES algorithms and to reduce risk management are given. Secure Hash Algorithms, their vulnerabilities, software implementations, risk of attacks and comparison with other hashing algorithms as well as the advantages and disadvantages between hashing techniques and encryption are given.

***Keywords*—**Cloud computing, encryption algorithm, secure hashing algorithm, brute force attack, birthday attack, plaintext attack, man-in-the-middle attack.

I. Introduction

CLOUD computing is also referred as "The Cloud". Cloud computing is nothing but shared data and computing processing resources among all computers available on networks. Cloud computing allows users to store data at a third party's data center. Cloud computing depends on the sharing of resources to gain coherence and effective usage of electricity. It is one of the most drastic and important turning points in the field of computer science, as it reduces the amount of data, while increasing the efficiency of data, and reducing costs, etc. On the other hand, it has also given birth to certain issues such as the security of user's data. Algorithms have developed to gain maximum security. Data is secured only if it fulfills the availability, integrity and confidentiality of data. Confidentiality is the most important requirement because a user has to be confident and satisfied if s/he keeps data on a third party data center. To obtain this level of security, many cryptographic algorithms are developed [4].

Cryptography is a technique to convert data into an unreadable form, which makes it useless for any intruder who tries to obtain the data without permission. A common approach to encrypt any data is to cipher it using a key, and on the other side using same key in reverse to decrypt it [4].

Cryptographic algorithms are divided into two forms: Encryptions (Symmetric and Asymmetric) algorithms and Hashing, which uses a key to cipher data. Symmetric algorithms use a single key at both ends and asymmetric algorithms use both public and private keys to cipher data at both ends. Hashing is a process which encrypts data using a particular algorithm known as hash function which consists of hash values. The difference between encryption and hashing of data is that Hashing cannot decrypt data as encryption does [15].

Jaimin Patel is with the Department of Computer Science and Engineering, The Maharaja Sayajirao University of Baroda, Vadodara, India (e-mail: jp30101995@gmail.com).

II. Hashing

Various types of hashing have developed. One common approach among all hashing algorithms is to cipher data using a hash function. The MD5 hashing algorithm was designed by Professor Ronald Rivest at MIT in 1991. This algorithm uses a hash function which generates a hash value of 128 bits; thus, there are 2^{128} combinations which can be easily broken by brute force attack using a current Intel 2.6 GHz Pentium 4 processor. Collision attack also breaks this hash value within seconds. More than 15 million hashes per second can be computed on NVIDIA GeForce graphics. GeForce 8800 Ultra can break around 200 million hashes per second [1], [8].

Secure Hash algorithms have been found to generate more than 128 Bit keys to overcome brute force and collision attacks. These algorithms are mainly known as SHA with different versions of SHA-0, SHA-1, SHA-2 and SHA-3 designed by the National Security Agency in US Federal Information Processing Standard. SHA-0, -1, -2, -3 are published by United States National Institute of Standards and Technology (NIST). The major differences among these are the number of rounds and output size, which increases the complexity to break the algorithm as the number of round increases. Brute Force Attack takes 2^1 rounds and Collision Attack takes $1.2*2^{l/2}$ rounds, where l is the number of rounds. Because of the increase in the number of rounds, it takes more time to break as compared to the MD series [8].

A. SHA

SHA-1 produces a 160-bit hash value which is renders in a 40-digit-long hexadecimal form. It has a message digest strength of 80-bit [2], [4]. Several examples are given to understand how actual data is encrypted.

For example, the statement for SHA-1 is given as "The quick brown fox jumps over the lazy dog" then hexadecimal form will be "2fd4e1c67a2d28fced84fced849ee1bb76e7391b93eb12".

Any kind of string less then size of 2^{64}-1 bits size can be converted into a hexadecimal form of 40 digits. Even a blank

string can also be converted into a 40-digit hexadecimal form.

Notes:

- Each variable are unsigned 32-bit.
- Message length is 64-bit and message digest is 160-bit.
- Constants are assumed in Big-endian format.
- H0, H1, H2, H3, H4 are variables which are initialized with some hex value.
- msgL = message length.

The process for the Algorithm is given below, which describes the bit process of the SHA algorithm in general form: SHA – 1 and SHA – 2.

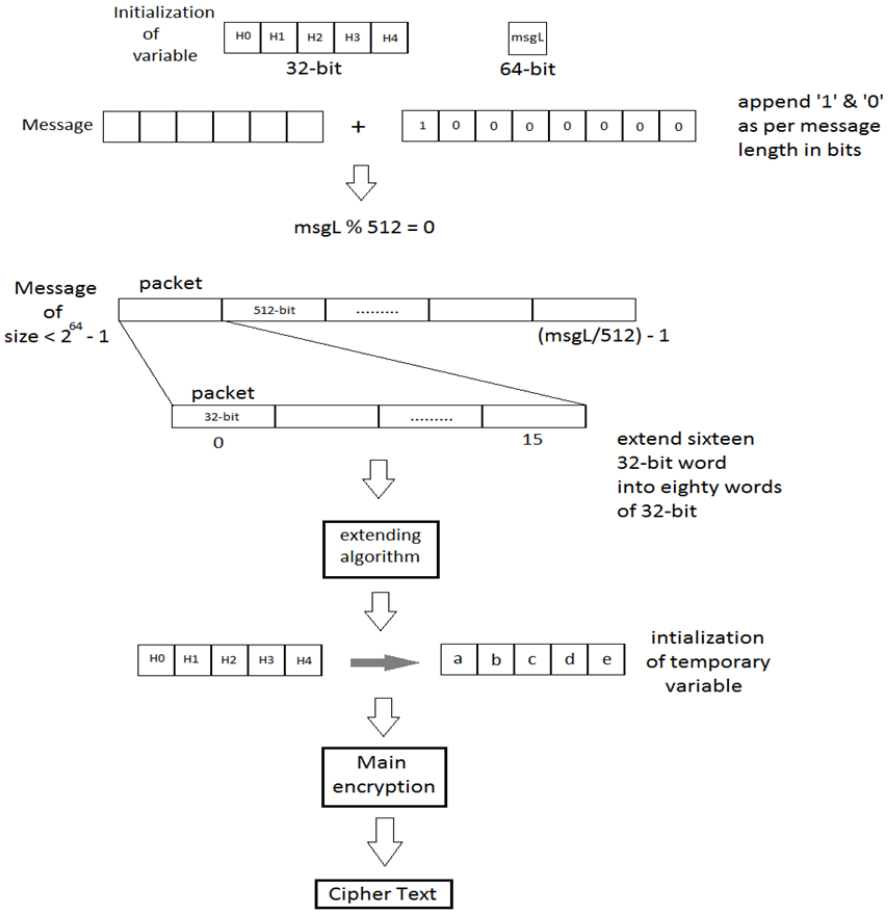

Fig. 1 Hashing algorithm

TABLE I
SHA-1 AND SHA-2 HASHING ALGORITHMS

	Number of Blocks used for main encryption algorithm	Extending Algorithm	Main Encryption algorithm	Final hash value
SHA - 1	80	Simple common formula for each is used with left-rotate and XOR operation	Different formula used according to number of each blocks	Left-rotate and or operation used for combination of each temporary variable
SHA - 2	64	Temporary variables are used as per packet number	Common formula is used for each packet using mathematical operations	Simple append operation of all temporary variable

- Number of Blocks used for the main encryption algorithm describes the block size used for the encryption.
- Extending Algorithms informs of the flow of algorithm expansion, and how small packets will be expanded into larger packets.
- Main Encryption Algorithm, given below, is for the main algorithm which encrypts data.
- Final hash value is a combination of all hash values to gather.

B. SHA – Analysis

Expanding nature and compression functions are different for Both SHA-1 and SHA-2 hash algorithms. This makes difference in their efficiency, vulnerability to attacks as well as block size. Highest message size which can be encrypted is also different. SHA-1 uses more number of packets with short number of operation than SHA-2. This makes SHA-2 more efficient than SHA-1.

Bits provided for security purpose are less than 80 which makes attacks easy in SHA-1. It is also easy to break it using

brute force attack and collision attack [2]. SHA – 2 has more than 80 security bits, especially in SHA – 384 and SHA – 512. SHA – 512 has a packet size of 1024 bits and 80 packets, and the Rotation and Shift operations are also different from other SHA – 2 algorithms. The bits for security are less than any other SHA – 2 algorithm in SHA – 512. On the other hand, it has maximum efficiency with comparatively high vulnerability [2], [3], [5].

III. Encryption

Encryption is also used to store data in encrypted form. A common approach for encryption is to use a single cipher function on both ends to encrypt as well as to decrypt; however, it is beneficial as well as harmful. It is beneficial in the sense that single a cipher key is or methodology is used to encrypt and decrypt. On the other hand, this makes it easy for an intruder to find that cipher key simply by checking the encrypted and decrypted form. Elements to be considered for an efficient encryption algorithm include:

Key should not be too large

If a key size is too large then the encryption speed will decrease and the algorithm will be inefficient.

Key should not be too small

If a key is small then it would be very easy to break using brute force or birthday attack. For example, if a block size is 'm', then as total number of permutation (2^M) will also be small.

Key should be in multiples of 8-bit

It is recommended because the register size is 8-bit. For example, if a key size is 47-bit, then we have to pad on one more '0' bit at the end to make it in multiple of eight. If the size is in a multiple of 8-bits, there is no need to pad extra.

A. Data Encryption Standard (DES)

The Data Encryption Standard was developed by IBM in 1970 by Horst Feistel for the US government's needs for computer security. It is considered as a weak algorithm, as its key size is 56-bit, which is too small to break and easy to break using today's processors [7].

The entire encryption process depends on the Feistel structure, which is also known as the F function in DES.

Note. F (K, R) is the Feistel function, where K is the key which depends on the requirements of the algorithms. The numbers of rounds are also variable, as per the requirement.

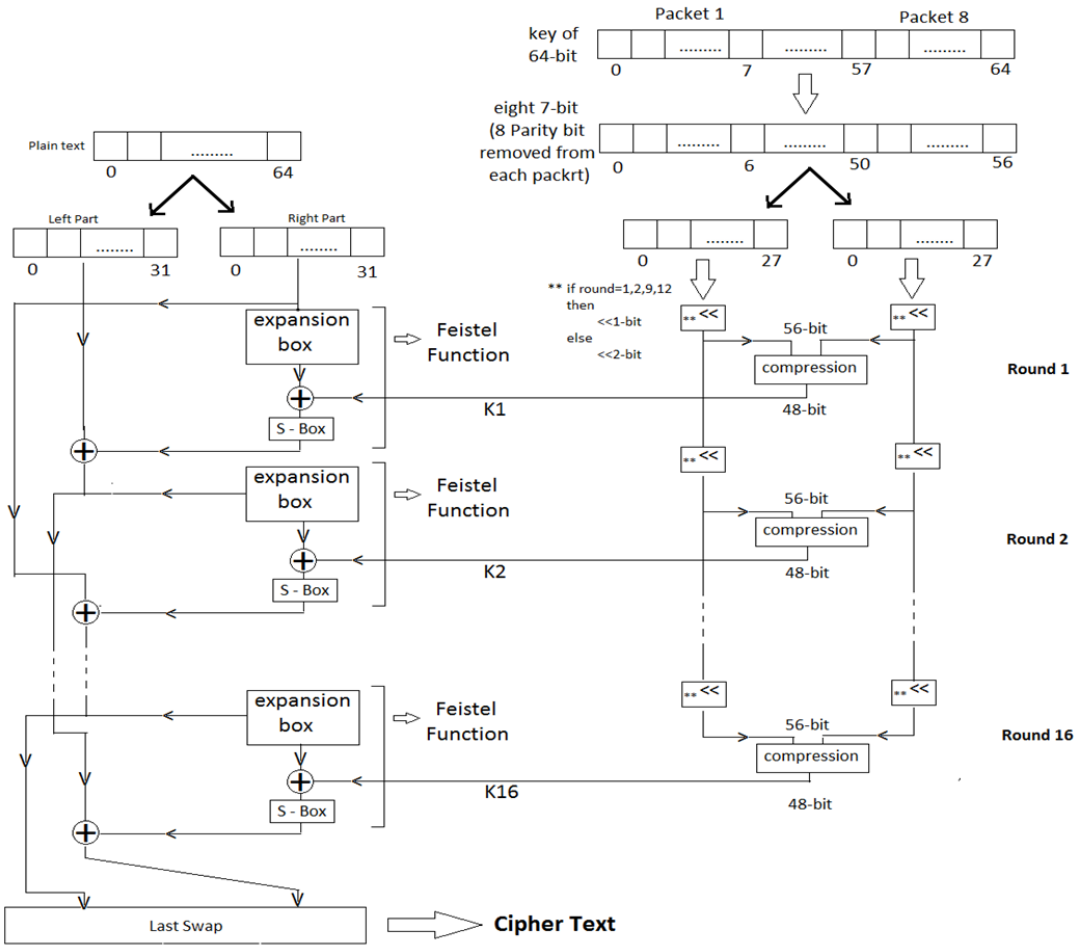

Fig. 2 DATA encryption algorithm

Decryption is also similar but in the reverse direction. Cipher text is input and plaintext is output. All Feistel functions are applied in the reverse order. The 16th swap shown in Fig. 2 between L and R is important here, because if there is an odd number of swaps with the XOR operation, then same thing cannot be reversed. There must be even number of steps to get the original data back from the encrypted data, as shown in Fig. 2. There must be an even number of swaps to decrypt any cipher text using the Feistel function.

DES uses 16 rounds of Feistel structure with a 64-bit block size, where 8-bits from these 64 are not used for the encryption process. It is not fixed that which 8 bits will not be used in encryption process. This makes an initial $\binom{64}{56}$ = 4,426,165,368 permutations and the same number of final permutations. Message and variables are assumed 32 bit unsigned. The key size is 48-bits. To match the size of the message and key, the algorithm uses an expanding box which takes the 32-bit as the input and 48-bit as the output. The expanding box uses a specific table or algorithm to pad extra 16 bits. In reverse, for the contraction it also uses a contraction box which uses the same algorithms in reverse to get plaintext. This takes the 48-bit as input and 32-bit as output.

DES uses 16 keys for each round of the Feistel function. It uses a Round-key generator to generate 16 different keys from a given single key [6], [9].

1. DES Analysis

DES uses a 56-bit round-key generator to generate 16 different keys, which gives: $2^{56} * \binom{56}{48} * \binom{48}{16} * 16!$ (16 factorial) rounds to get which round of 16 keys are used. Base clock speed of current processor is 4 GHz. This means that clock can complete 4,000,000,000 rounds per second. Therefore, with this processing power it is very easy to break such encryptions within a matter of minutes. Even back in 1977, Diffie and Hellman designed a machine that could break this encryption in a single day [12].

To improve its performance, IBM and NSA designed an algorithm which requires at least 2^{49} plaintext samples to break. However, this is still not enough. RC5, IDEA, SAFER, FEAL also required around 2^{49} samples to break. Improvements to DES give rise to a new version known as Triple DES [12].

B. Triple DES

Similar to DES, the major difference of the Triple DES is the key length. It uses three 56-bit keys at three different levels, for a total key size is 168-bit. Thus, the total number of permutations will be increased from 2^{56} to 2^{168}.

The decryption process is also same in reverse with key order k3, k2, k1. Process will be same as shown in Fig. 2. It also provides backward compatibility to DES which is one of its major benefits. Normal DES is considered as a 168-bit key, which has the same three 56-bit keys like k1, k1, k1 of Fig. 2. Sometimes, only two 56-bit keys are used in order of k1, k2, k1 to increase the encryption performance. Decryption speed will also be increased because of same order of keys. This is called the 2-DES algorithm [14]. The 2-DES is used when high security is not required. It is weaker than the 3-DES in performance.

C. 3-DES Analysis

The main advantage of 3-DES is that it encrypts data in three levels, therefore, reducing the risk of a meet-in-the-middle attack. 3-DES uses a 168-bit key. Meet-in-the-middle attack can only break up to 112-bit keys. The 3-DES algorithm can provide security for 112-bits, while 2-DES cannot. Thus, it is not recommended to use the 2-DES algorithm when high security is major issue.

Brute Force Attack requires 2^{88} bit memory, 2^{90} DES decryption, which is almost impossible for the current processor specification [12].

The drawback is that 3-DES algorithm uses levels and that is why it takes more time than simplifies DES. Chances of errors are also high due to linear computation. It is around three times higher for both encryption and decryption. For this reason, use of the 2-DES algorithm is preferred, as it requires almost the same time and processing for both encryption and decryption. One common drawback for DES is security or speed (in terms of efficiency), which must compromised. To solve this problem of linear calculation (number of rounds) and complexity, other algorithm standards were developed known as, Advanced Encryptions Standards, which encrypt data in levels and also do not have any specific number of rounds; it varies according to the key size [12], [13].

IV. Advanced Encryption Standards (AES)

Advanced Encryption Standards also include the well-known Rijndael Algorithms. They were designed by NIST in 2001. AES is part of the Rijndael cipher, which was design by Joan Daemen and Vincent Rijmen, and its major advantage is that it uses variable key sizes of 128-bit, 192-bit and 256-bit. The AES encryption function uses four different operations which are comparatively more efficient than DES; almost all levels have O(n) complexity.

The number of cycle of repetitions depends on the key size, which are:

- 10 cycles - 128-bit keys
- 12 cycles – 192-bit keys
- 14 cycles – 256-bit keys

Analysis for the algorithm is described here for 128-bit keys with a repetition of 10 cycles. The encryption process consists of three major rounds as shown in Figs. 3 and 4.

- Initial round: The AddRound Key, K0, is combined with plaintext.
- Rounds: SubBytes – bytes are replaced according to Rijndael's S-Box lookup table.
  - ShiftRows – Bytes are shifted left using a pattern
  - MixColumns – mixing operations are applied on each Byte of any column.
  - AddRound Key – Exactly same process as the initial round using the next key.
- Final round: All rounds are the same as the aforementioned rounds, with the exception of Mix

Columns [9].

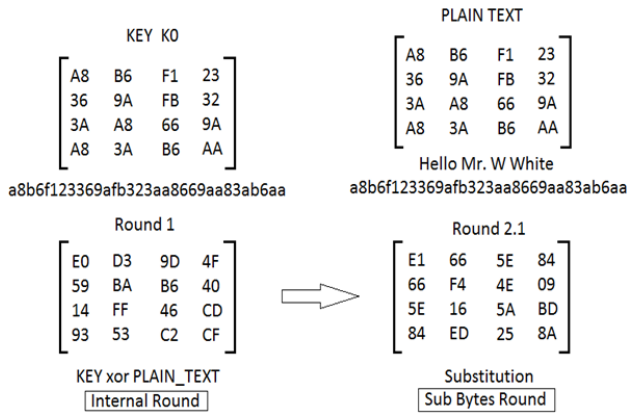

Fig. 3 Advance encryption algorithm part 1

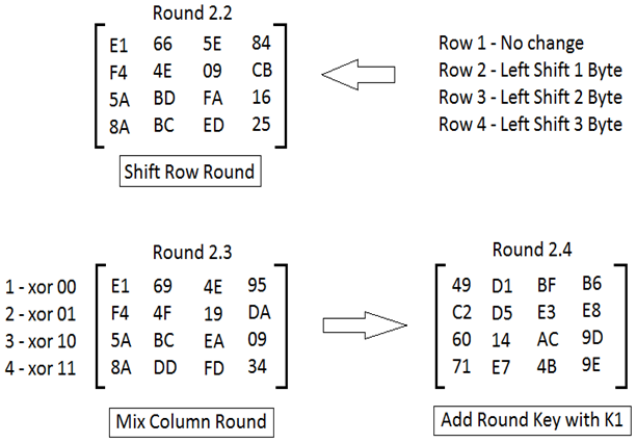

Fig. 4 Advance encryption algorithm part 2

A. AES – Analysis

- The major advantage of AES is its variable key size, and variable behavior in rounds like Mix Column. Thus, it is difficult to assume which column is used for the Mix Column sub-round. It is impossible to break a Mix Column sub-round using Brute Force attack.
- The complexity of AES is also efficient. The Sub Byte sub-round also has lower complexity of O(n), where n is the number of bytes. AES is six times faster than DES.
- Each round in AES is independent from each other. So that it is impossible to find relationship between rounds. As a result, the avalanche effect will be seen less when compared to DES.
- AES can be implemented on both hardware as well as software. The decryption process is also the same, but in the reverse direct of rounds.
- The drawbacks of AES are as follow: once algebraic computations of an encryption are known, it is very easy to break it. That is why Side Channel attack is more harmful than any other attack. An XSL-attack can also break AES into small amount of time [10], [11].

REFERENCES

[1] Marc Stevens (June 2012). "Attacks on Hash Functions and Applications" - PhD thesis.
[2] Schneier on Security–Cryptanalysis of SHA–1 http://www.schneier.com 17th July, 2016.
[3] Dmitry Khovratovich, Christian Rechberger & Alexandra Savelieva. "Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 family". IACR Cryptology.
[4] National Institute on Standards and Technology Computer Security Resource Center, NIST's Policy on Hash Functions.
[5] Diffie, Whitfield; Hellman, Martin E. "Exhaustive Cryptanalysis of the NBS Data Encryption Standard". Computer. 10 (6): pp 74–84.
[6] Konheim. Computer Security and Cryptography. pp. 301.
[7] William E. Burr, "Data Encryption Standard", in NIST's anthology "A Century of Excellence in Measurements, Standards, and Technology": A Chronicle of Selected NBS/NIST Publications, 1901–2000 Link: http://nvl.nist.gov/pub/nistpubs/sp958-lide/250-253.pdf - 1st July, 2016.
[8] Schneier. Applied Cryptography (1st edition). Page number - 271. Link - https://www.schneier.com/books/applied_cryptography/1errv159.html
[9] "Efficient software implementation of AES on 32-bit platforms". Lecture Notes in Computer Science: 2523. 2003.
[10] Bruce Schneier, AES Announced, October 15, 2000.
[11] Nikolic, Ivica (2009). "Distinguisher and Related-Key Attack on the Full AES-256".Advances in Cryptology – CRYPTO 2009. Springer Berlin / Heidelberg. pp. 231–249.doi: 10.1007/978-3-642-03356-8_14. ISBN 978-3-642-03355-1.
[12] Biham, Eli and Shamir, Adi (1991). "Differential Cryptanalysis of DES-like Cryptosystems". Journal of Cryptology. 4 (1): 3–72. doi:10.1007/BF00630563.
[13] New Comparative Study between DES, 3DES and AES within Nine Factors. Journal of computing, volume 2, issue 3, march 2010, issn 2151-9617. Retrieved 2012-12-01. Link - http://arxiv.org/pdf/1003.4085.pdf
[14] "Triple DES Encryption". IBM. Retrieved 2014-05-17.
[15] A Classical introduction to cryptography by Serge Vaudenay - Introduction