Journal Homepage: - www.journalijar.com
INTERNATIONAL JOURNAL OF ADVANCED RESEARCH (IJAR)
Article DOI: 10.21474/IJAR01/2081
DOI URL: http://dx.doi.org/10.21474/IJAR01/2081
ISSN NO. 2320-5407

RESEARCH ARTICLE

SMA CRYPTOGRAPHY ALGORITHM DECRYPT MD5 SOLUTION

Mr. Shady Ayesh and Dr. Mohammed Dweib.
1. Department of Computer Information System. Al - Quds Open University, Palestine.
2. Department of Computer Information System. Al - Quds Open University, Palestine.

Manuscript Info

Manuscript History

Received: 24 September 2016
Final Accepted: 26 October 2016
Published: November 2016

Key words:-
cryptography, MD5,RSA,encryption.

Abstract

In this article, we aim to discuss and analyze a new algorithm of cryptography which makes the application in any type more secure, and let all the users and administrators keep their critical data in safe. We depend on Md5 and RSA algorithms to populate the SMA algorithm for encryption. In SMA algorithm, the user only can decrypt his critical data according to special secure code that he is the only person know.

SMA Algorithm is a one way Encryption which means we can't decrypt the encrypted data without having a special secure code which is known by the user himself. We depend in this algorithm on some data such as two primary numbers, user secure code, salt string and generate five rounds depending on bitwise operations.

Introduction:-

Cryptography or cryptology (from Greek κρυπτός kryptós, "hidden, secret"; and γράφειν graphein, "writing", or -λογία -logia, "study", respectively[1]) is the practice and study of techniques for secure communication in the presence of third parties called adversaries [2]. More generally, cryptography is about constructing and analyzing protocols that prevent third parties or the public from reading private messages [3]; various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation[4] are central to modern cryptography. Modern cryptography exists at the intersection of the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.

In the last period , we found on the search engines that we have "Decrypt MD5" , where MD5 is very used in encrypting data on web application. This point put me on a spot of concentration to search for a solution to this big problem.

1. The first goal of my study is to solve the problem of the hacked algorithm (MD5). So, I started studying the concept of this algorithm, and analyzing it.
2. The second goal of this study is to develop this algorithm and use it in our applications that we develop [13].

MD5 Algorithm Analysis:-

The md5 algorithm is a widely used hash function producing a 128-bit hash value. although md5 was initially designed to be used as a cryptographic hash function, it has been found to suffer from extensive vulnerabilities. It can still be used as a checksum to verify data integrity, but only against unintentional corruption.

Corresponding Author:- Mr. Shady Ayesh.
Address:- Computer Information System. Al Quds Open University, Palestine.

Like most hash functions, MD5 is neither encryption nor encoding. It can be reversed by brute-force attack and suffers from extensive vulnerabilities as detailed in the security section below.

MD5 was designed by Ronald Rivest in 1991 to replace an earlier hash function MD4. The source code in RFC 1321 contains a "by attribution" RSA license.

The security of the MD5 has been severely compromised, with its weaknesses having been exploited in the field, most infamously by the Flame malware in 2012. The CMU Software Engineering Institute considers MD5 essentially "cryptographically broken and unsuitable for further use".

The security of the MD5 has been severely compromised, with its weaknesses having been exploited in the field, most infamously by the Flame malware in 2012. The CMU Software Engineering Institute considers MD5 essentially "cryptographically broken and unsuitable for further use"[5][11][14].

MD5 Steps:-
Step 1. Appending Padding Bits. The original message is "padded" (extended) so that its length (in bits) is congruent to 448, modulo 512. The padding rules are:

The original message is always padded with one bit "1" first.
Then zero or more bits "0" are padded to bring the length of the message up to 64 bits fewer than a multiple of 512.

Step 2. Appending Length. 64 bits are appended to the end of the padded message to indicate the length of the original message in bytes. The rules of appending length are:

The length of the original message in bytes is converted to its binary format of 64 bits. If overflow happens, only the low-order 64 bits are used.

Break the 64-bit length into 2 words (32 bits each).
The low-order word is appended first and followed by the high-order word.

Step 3. Initializing MD Buffer. MD5 algorithm requires a 128-bit buffer with a specific initial value. The rules of initializing buffer are:

The buffer is divided into 4 words (32 bits each), named as A, B, C, and D.
1. Word A is initialized to: 0x67452301.
2. Word B is initialized to: 0xEFCDAB89.
3. Word C is initialized to: 0x98BADCFE.
4. Word D is initialized to: 0x10325476.

Step 4. Processing Message in 512-bit Blocks. This is the main step of MD 5 algorithm, which loops through the padded and appended message in blocks of 512 bits each. For each input block, 4 rounds of operations are performed with 16 operations in each round. This step can be described in the following pseudo code slightly modified from the RFC 1321's version:

Input and predefined functions:
A, B, C, D: initialized buffer words

F(X,Y,Z) = (X AND Y) OR (NOT X AND Z)
G(X,Y,Z) = (X AND Z) OR (Y AND NOT Z)
H(X,Y,Z) = X XOR Y XOR Z
I(X,Y,Z) = Y XOR (X OR NOT Z)

T[1, 2, ..., 64]: Array of special constants (32-bit integers) as:
T[i] = int(abs(sin(i)) * 2**32)

M[1, 2, ..., N]: Blocks of the padded and appended message

R1(a,b,c,d,X,s,i): Round 1 operation defined as:
a = b + ((a + F(b,c,d) + X + T[i]) <<< s)

R2(a,b,c,d,X,s,i): Round 1 operation defined as:
a = b + ((a + G(b,c,d) + X + T[i]) <<< s)

R3(a,b,c,d,X,s,i): Round 1 operation defined as:
a = b + ((a + H(b,c,d) + X + T[i]) <<< s)

R4(a,b,c,d,X,s,i): Round 1 operation defined as:
a = b + ((a + I(b,c,d) + X + T[i]) <<< s)

Algorithm:-
For k = 1 to N do the following

AA = A
BB = B
CC = C
DD = D
(X[0], X[1], ..., X[15]) = M[k] /* Divide M[k] into 16 words */

/* Round 1. Do 16 operations. */
R1(A,B,C,D,X[0], 7, 1)
R1(D,A,B,C,X[1],12, 2)
R1(C,D,A,B,X[2],17, 3)
R1(B,C,D,A,X[3],22, 4)
R1(A,B,C,D,X[4], 7, 5)
R1(D,A,B,C,X[5],12, 6)
R1(C,D,A,B,X[6],17, 7)
R1(B,C,D,A,X[7],22, 8)
R1(A,B,C,D,X[8], 7, 9)
R1(D,A,B,C,X[9],12,10)
R1(C,D,A,B,X[10],17,11)
R1(B,C,D,A,X[11],22,12)
R1(A,B,C,D,X[12], 7,13)
R1(D,A,B,C,X[13],12,14)
R1(C,D,A,B,X[14],17,15)
R1(B,C,D,A,X[15],22,16)

/* Round 2. Do 16 operations. */
R2(A,B,C,D,X[1], 5,17)
R2(D,A,B,C,X[6], 9,18)
R2(C,D,A,B,X[11],14,19)
R2(B,C,D,A,X[0],20,20)
R2(A,B,C,D,X[5], 5,21)
R2(D,A,B,C,X[10], 9,22)
R2(C,D,A,B,X[15],14,23)
R2(B,C,D,A,X[4],20,24)
R2(A,B,C,D,X[9], 5,25)
R2(D,A,B,C,X[14], 9,26)
R2(C,D,A,B,X[3],14,27)
R2(B,C,D,A,X[8],20,28)
R2(A,B,C,D,X[13], 5,29)
R2(D,A,B,C,X[2], 9,30)

R2(C,D,A,B,X[7],14,31)
R2(B,C,D,A,X[12],20,32)

/* Round 3. Do 16 operations. */
R3(A,B,C,D,X[5], 4,33)
R3(D,A,B,C,X[8],11,34)
R3(C,D,A,B,X[11],16,35)
R3(B,C,D,A,X[14],23,36)
R3(A,B,C,D,X[1], 4,37)
R3(D,A,B,C,X[4],11,38)
R3(C,D,A,B,X[7],16,39)
R3(B,C,D,A,X[10],23,40)
R3(A,B,C,D,X[13], 4,41)
R3(D,A,B,C,X[0],11,42)
R3(C,D,A,B,X[3],16,43)
R3(B,C,D,A,X[6],23,44)
R3(A,B,C,D,X[9], 4,45)
R3(D,A,B,C,X[12],11,46)
R3(C,D,A,B,X[15],16,47)
R3(B,C,D,A,X[2],23,48)

/* Round 4. Do 16 operations. */
R4(A,B,C,D,X[0], 6,49)
R4(D,A,B,C,X[7],10,50)
R4(C,D,A,B,X[14],15,51)
R4(B,C,D,A,X[5],21,52)
R4(A,B,C,D,X[12], 6,53)
R4(D,A,B,C,X[3],10,54)
R4(C,D,A,B,X[10],15,55)
R4(B,C,D,A,X[1],21,56)
R4(A,B,C,D,X[8], 6,57)
R4(D,A,B,C,X[15],10,58)
R4(C,D,A,B,X[6],15,59)
R4(B,C,D,A,X[13],21,60)
R4(A,B,C,D,X[4], 6,61)
R4(D,A,B,C,X[11],10,62)
R4(C,D,A,B,X[2],15,63)
R4(B,C,D,A,X[9],21,64)

A = A + AA
B = B + BB
C = C + CC
D = D + DD
End of for loop

Output:
A, B, C, D: Message digest

Step 5. Output. The contents in buffer words A, B, C, D are returned in sequence with low-order byte first [6].

RSA Algorithm Analysis:-

RSA is one of the first practical public-key cryptosystems and is widely used for secure data transmission. In such a cryptosystem, the encryption key is public and differs from the decryption key which is kept secret. In RSA, this asymmetry is based on the practical difficulty of factoring the product of two large prime numbers, the factoring problem. RSA is made of the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who

first publicly described the algorithm in 1977. Clifford Cocks, an English mathematician working for the UK intelligence agency GCHQ, had developed an equivalent system in 1973, but it was not declassified until 1997 [7].

A user of RSA creates and then publishes a public key based on two large prime numbers, along with an auxiliary value. The prime numbers must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, if the public key is large enough, only someone with knowledge of the prime numbers can feasibly decode the message [8]. Breaking RSA encryption is known as the RSA problem; whether it is as hard as the factoring problem remains an open question.

RSA is a relatively slow algorithm, and because of this it is less commonly used to directly encrypt user data. More often, RSA passes encrypted shared keys for symmetric key cryptography which in turn can perform bulk encryption-decryption operations at much higher speed [9][12].

RSA encrypts messages through the following algorithm, which is divided into 3 steps:

Key Generation:-
Choose two distinct prime numbers p and q.

Find n such that $n = pq$.
n will be used as the modulus for both the public and private keys.

Find the totient function of n, $\phi(n)$
$\phi(n)=(p-1)(q-1)$.

Choose an e such that $1 < e < \phi(n)$, and such that e and $\phi(n)$ share no divisors other than 1 (e and $\phi(n)$ are relatively prime).
e is kept as the public key exponent.

Determine d (using modular arithmetic) which satisfies the congruence relation
$de \equiv 1 \pmod{\phi(n)}$.

In other words, pick d such that de - 1 can be evenly divided by (p-1)(q-1), the totient function, or $\phi(n)$.
This is often computed using the Extended Euclidean Algorithm, since e and $\phi(n)$ are relatively prime and d is to be the modular multiplicative inverse of e.

d is kept as the private key exponent.
The public key has modulus n and the public (or encryption) exponent e. The private key has modulus n and the private (or decryption) exponent d, which is kept secret.

Encryption:-
Person A transmits his/her public key (modulus n and exponent e) to Person B, keeping his/her private key secret.
When Person B wishes to send the message "M" to Person A, he first converts M to an integer such that $0 < m < n$ by using agreed upon reversible protocol known as a padding scheme.
Person B computes, with Person A's public key information, the cipher text c corresponding to $c \equiv me \pmod{n}$.
Person B now sends message "M" in cipher text, or c, to Person A.

Decryption:-
Person A recovers m from c by using his/her private key exponent, d, by the computation $m \equiv cd \pmod{n}$.
Given m, Person A can recover the original message "M" by reversing the padding scheme.
This procedure works since
$c \equiv me \pmod{n}$,
$cd \equiv (me)d \pmod{n}$,
$cd \equiv mde \pmod{n}$.

By the symmetry property of mods we have that
$m^{de} \equiv m^{de} \pmod{n}$.
Since $de = 1 + k\phi(n)$, we can write
$m^{de} \equiv m^{1 + k\phi(n)} \pmod{n}$,
$m^{de} \equiv m(m^{k})^{\phi(n)} \pmod{n}$,
$m^{de} \equiv m \pmod{n}$.

From Euler's Theorem and the Chinese Remainder Theorem, we can show that this is true for all m and the original message
$c^{d} \equiv m \pmod{n}$, is obtained [10].

SMA Algorithm Analysis:-

SMA algorithm stands for " Standard Maintain Algorithm". In this algorithm I depended on MD5 output then I made some functions to change the cipher text.

In this algorithm we concentrate on a user secure code which is used as input beside the plain text . This secure code doesn't save in any variable in the system or in the database of the system. It's also used in encryption and needed in decryption that will not be decrypted if we don't know the secure code. So, this is the basic steps of the algorithm:

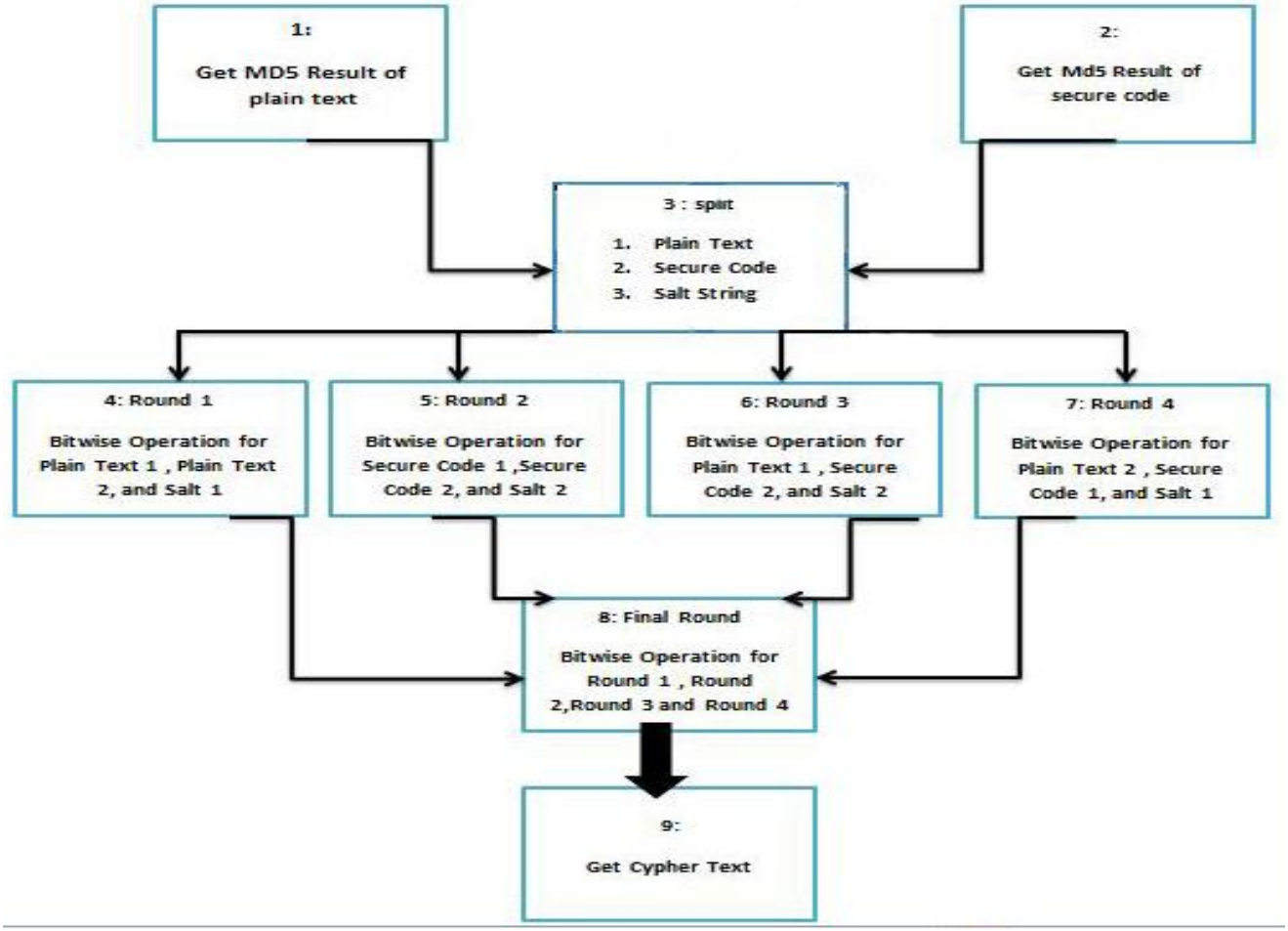

Figure 1:- SMA Steps.

SMA algorithm depends on some attributes, such that:

1. The user Secure Code : this code is used in encrypting operations, and used to generate the 3 functions mentioned above.

2. Two prime numbers: all the mathematical operations depends on these two prime numbers. So, we use these two prime numbers to make some calculations in the SMA algorithm.

Conclusion:-

As a conclusion of this article, we achieve the following:

- ❖ We solve the problem with MD5 decrypted algorithm.
- ❖ We get a new cipher text that is cannot be decrypted if we don't have the secure code.
- ❖ We get a new way of encryption depends on a secure code that nobody knows unless the user himself.
- ❖ The generated cipher text is with length of 224 bits, that depends on a secure code with at least length 40 bits.
- ❖ The most important point in SMA algorithm is how to keep the data more secure and non-decrypted by using a secure code with a good length to satisfy the goal of this algorithm.
- ❖ We test SMA algorithm in the graduation project "Tuning Connection To Database (TCTD)" for saving secure data in web page and Java application as a sample of financial system to test the system we developed.

Comparison:-

In this section we will compare between SMA, MD5 and RSA algorithms:[13][15]

SMA	MD5	RSA
512 BITS	128 BITS	330 – 2048 BITS
ASSYMETRIC	HASH	ASSYMETRIC
SECURE, PRIVATE and PUBLIC	NONE	PUBLIC and PRIVATE
NO	YES	NO

References:-

1. Liddell, Henry George; Scott, Robert; Jones, Henry Stuart; McKenzie, Roderick (1984). A Greek-English Lexicon. Oxford University Press.
2. Rivest, Ronald L. (1990). "Cryptography". In J. Van Leeuwen. Handbook of Theoretical Computer Science 1. Elsevier.
3. Bellare, Mihir; Rogaway, Phillip (21 September 2005). "Introduction". Introduction to Modern Cryptography. p. 10.
4. Menezes, A. J.; van Oorschot, P. C.; Vanstone, S. A. Handbook of Applied Cryptography. ISBN 0-8493-8523-7. Archived from the original on 7 March 2005.
5. Ciampa, Mark (2009). CompTIA Security+ 2008 in depth. Australia ; United States: Course Technology/Cengage Learning. p. 290.
6. Dr. Herong Yang - The MD5 Message-Digest Algorithmwww.herongyang.com/Cryptography/MD5-Message-Digest-Algorithm-Overview.html.
7. Smart, Nigel (February 19, 2008). "Dr. Clifford Cocks CB". Bristol University. Retrieved August 14, 2011.
8. Diffie, W.; Hellman, M.E. (November 1976). "New directions in cryptography". IEEE Transactions on Information Theory 22 (6): 644–654. doi:10.1109/TIT.1976.1055638. ISSN 0018-9448.
9. Wikipedia. https://en.wikipedia.org/wiki/RSA_(cryptosystem)#cite_note-rsa-2.
10. Sites.google. http://www.di-mgt.com.au/rsa_alg.html.
11. Shahzad Alam, Amir Jamil, Ankur Saldhi, Musheer Ahmad, "Digital Image Authentication and EncryptionUsing Digital Signature," International Conference on Advances in Computer Engineering & Applications, pp. 332-336, 2015
12. Keonwoo Kim, Un Sung Kyong, "Efficient Implementation of MD5 Algorithm in Password Recovery of a PDF File," Cyber Convergence Security Division, ETRI, Daejon, Korea, pp. 1080-1083
13. Anak Agung Putri Ratna, Ahmad Shaugi, Prima DewiPurnamasari, Muhammad Salman, "Analysis and Comparison of MD5 and SHA-1 Algorithm
14. Wang Xiayoun, "How to Break MD5 and other Hash Functions," 2005
15. Rashmi P.Sarode, Piyush Gupta, Neeraj Manglani, "A Comparative analysis of RSA and MD5 Algorithm," Journal of Computer Science and Applications, pp. 25-33,2014.