Journal of Mobile Communication 1 (4): 109-113, 2007
ISSN: 1990-794X

Security (GCSEC) Algorithm for non Real Time Data in Mobile Networks

[1]M. Ganaga Durga, [2]G. Chandrasekaran and [3]S. Arivazhagan
[1]Department of M.C.A., K.L.N. College of Engineering, Tamilnadu, India
[2]Department of Computer Applications and [3]Department of ECE,
Mepco Schlenk Engineering Collge, Tamilnadu, India

Abstract: The focal hub delves into conniving an proficient 256-bit hash utility to shield the non real time data and its hacking echelon is compared with the hash algorithms. This study investigates the recital of hashing algorithms by an untried loom. Hash functions have ample and imperative role in cryptography. They generate hash ethics, which succinctly embody longer messages or credentials from which they were computed. The focal task of cryptographic hash functions is in the stipulation of message veracity checks and digital signatures. This algorithm utterly secures the non real time data by encrypting the data what is sent or time-honored.

Key words: Refuge, hash utility, psychiatry, hacking tempo, digest

INTRODUCTION

Security-an ever-growing concern: Security requirements in any transaction are directly proportional to the transaction's value, sensitivity and volume. Mobile transactions in the cash management industry typically possess all these characteristics. Indeed, security is a *de facto* requirement for any transaction channel in the cash management business andthus most security requirements are fairly generic and independent of the channel. However, the mobile channel has unique characteristics that can result in several new security vulnerabilities.

Authentication is the remedy to the other attacks. User Authentication is defined as Provision of Assurance that the message is originated from authorized user (William). Message Authentication is defined as 'Provision of assurance that the message is not altered'. One type of Message Authentication is by hash algorithm. This provides an assurance to the destination that the message is not changed by the intruders (Nist, 2002). To get protected from eavesdropping, encryption in source and decryption in destination is also done with the help of a secret key. Encryption can be used to ensure secrecy, but other techniques are still needed to make communications secure, particularly to verify the integrity and authenticity of a message; for example, a Message Authentication Code (MAC) or digital signatures. Another consideration is protection against traffic analysis.

PROJECTED DESIGN

This proposal describes an proficient 256-bit hash function, GCSEC. It is designed not only to have higher seclusion but also efficiency. The recital of the new hash function is better than that of other algorithms in software.

For an idyllic hash utility with an m-bit output, finding a preimage or a second preimage requires about 2^m operations and the best ever way to unearth a conflict is a birthday attack which desires approximately $2^{m/2}$ operations. Most dedicated hash functions which have iterative procedure use the Merkle-Damgard edifice in order to hash inputs of arbitrary length. They work as follows. Let HASH be a hash function. The message X is padded to a multiple of the block length and subsequently divided into t blocks X1, • • •, Xt. Then HASH can be described as follows:

$$CV0 = IV;\ CVi = COMP(CVi\text{-}1, Xi),\ 1 \le i \le t;$$
$$HASH(X) = CVt,$$

Where, COMP is the solidity utility of HASH, CVi is the chaining capricious between stage i and stage i + 1 andIV denotes the preliminary estimation. The most trendy method of deceitful compression functions of obsessive hash functions is a serial successive iteration of a small step function, as like round functions of chunk ciphers. (Biham and Chen, 2004) specifies that many hash functions such as MD4, MD5, HAVAL, SHA-family,

Corresponding Author: M. Ganaga Durga, 119 Aruthu Pandiyar Street, Vasuki Nagar, Iii Street, Valluvar Colony, Madurai 625 017, Tamil Nadu, India

follow this. Attacks on hash functions have been paying attention on vanishing the difference of intermediate values caused by the difference of messages. MD4-type hash functions including SHA-1 are vulnerable to Wang collision-finding attack. RIPEMD-family has somewhat diverse approach for designing a secure hash function. The invader who tries to break members of RIPEMD family should aim simultaneously at two ways where the message difference passes. This design strategy is still successful because so far there is not any effective attack on RIPEMD-family except the first proposal of RIPEMD. However, Bellovin (1989) says that RIPEMD-family have heavier compression functions than hash functions with serial structure. Total number of steps is twice as many as that of MD4. Also, the number of steps of RIPEMD-160 is almost twice as many as that of SHA-0. In this study, we propose a dedicated hash function. According to the observation, we determined the design goals (of compression function) as follows:

- It should have a 256-bit output because the security of 2^{128} operations is recommended for symmetric key cryptography as the computing power increases.
- Its structure should be resistant against known attacks including Wang attack.
- The performance should be as competitive as that of SHA-256.

COMPUTATIONAL STEPS OF GCSEC

These are basic notations used in GCSEC:
+ : Addition mod 2^{32}
⊕ : XOR (exclusive OR)
<<<s : S-bit left rotation for a 32-bit string A

Input block length and padding: An input message is processed by 512-bit block. GCSEC pads a message by appending a single bit 1 next to the least significant bit of the message, followed by zero or more bit 0s until the length of the message is 448 modulo 512 andthen appends to the message the 64-bit original message length modulo 2^{64}.

Structure of GCSEC: The solidity function of GCSEC hashes a 512-bit string to a 256-bit string. It consists of five parallel branch functions, BRANCH1, BRANCH2, BRANCH3, BRANCH4 and BRANCH5 (Fig. 1). Let Cvi = (A, B, C, D, E, F, G, H) be the chaining variable of the density function. It is initialized to IV0 which is A = 6a09e667xB = bb67ae85xC = 3c6ef372xD = a54ff53axE = 510e527fxF = 9b05688cxG = 1f83d9abxH = 5be0cd19x.

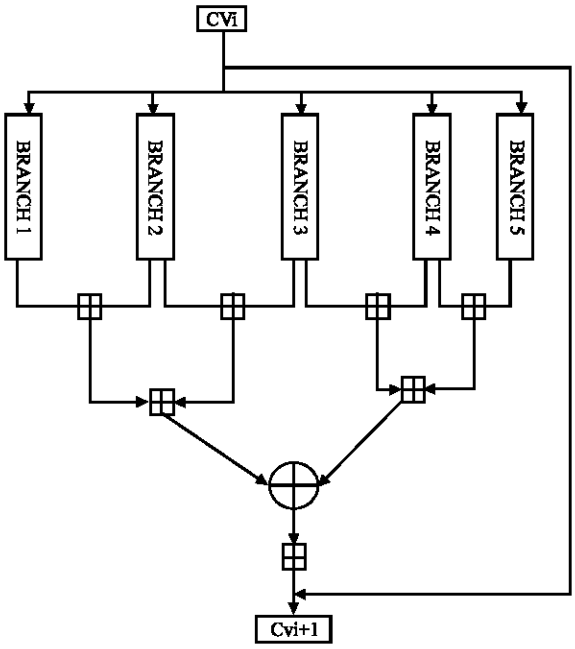

Fig. 1: Structure of GCSEC

Each consecutive 512-bit message block M is divided into sixteen 32-bit words M0, M1, • • •, M15 and the following computation is performed to update CVi to CVi+1:

$Z = [BRANCH1(CVi, \Sigma_1(M)) + BRANCH2(CVi, \Sigma_2(M))]$
$Y = [BRANCH2(CVi, \Sigma_3(M)) + BRANCH3(CVi, \Sigma_2(M))]$
$X = [BRANCH3(CVi, \Sigma_3(M)) + BRANCH4(CVi, \Sigma_4(M))]$
$X1 = [BRANCH4(CVi, \Sigma_3(M)) + BRANCH5(CVi, \Sigma_4(M))]$
$X2 = Z + Y$ and $X3 = X + X1$ and $CVi+1 = CVi + [X2 \oplus X3]$

Where, $\Sigma_j(M) = (M\sigma j(0), \bullet\bullet\bullet, M\sigma j(15))$ is the re-ordering of message words for j = 1, 2, 3, 4,5 given by Table 1.

ALGORITHM GUSH

Branch functions: Each BRANCHj is computed as follows:

- The chaining variable Cvi is copied to initial variables Vj, 0 for j-th branch.
- At k-th step of each BRANCHj ($0 \le k \le 7$), the step function STEPj, k is computed as follows:

Vj, k+1 = STEPj, k (Vj, k, M_j (2k), M_j (2k+1), αj, k, βj, k), where αj, k and βj, k are constants. Input Order of Message Words this table shows the input order of message words 0 to M 15 applied to BRANCHj ($1 \le j \le 5$) functions (Fig. 2).

Table 1: Ordering rule of message words

T	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\Sigma_1(t)$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\Sigma_2(t)$	15	16	12	10	9	11	4	5	3	14	1	6	7	8	13	2
$\Sigma_3(t)$	8	7	11	15	14	3	10	13	12	5	16	9	6	1	2	4
$\Sigma_4(t)$	6	13	2	9	16	1	14	12	4	11	10	3	8	15	5	7
$\Sigma_5(t)$	13	14	1	2	7	8	5	6	11	12	9	10	3	4	15	16

Table 2: Ordering of constants

Step k	$\alpha_{1,k}$	$\beta_{1,k}$	$\alpha_{2,k}$	$\beta_{2,k}$	$\alpha_{3,k}$	$\beta_{3,k}$	$\alpha_{4,k}$	$\beta_{4,k}$
0	δ_0	δ_1	δ_{15}	δ_{14}	δ_1	δ_0	δ_{14}	δ_{15}
1	δ_2	δ_3	δ_{13}	δ_{12}	δ_3	δ_2	δ_{12}	δ_{13}
2	δ_4	δ_5	δ_{11}	δ_{10}	δ_5	δ_4	δ_{10}	δ_{11}
3	δ_6	δ_7	δ_9	δ_8	δ_7	δ_6	δ_8	δ_9
4	δ_8	δ_9	δ_7	δ_6	δ_9	δ_8	δ_6	δ_7
5	δ_{10}	δ_{11}	δ_5	δ_4	δ_{11}	δ_{10}	δ_4	δ_5
6	δ_{12}	δ_{13}	δ_3	δ_2	δ_{13}	δ_{12}	δ_2	δ_3
7	δ_{14}	δ_{15}	δ_1	δ_0	δ_{15}	δ_{14}	δ_0	δ_1

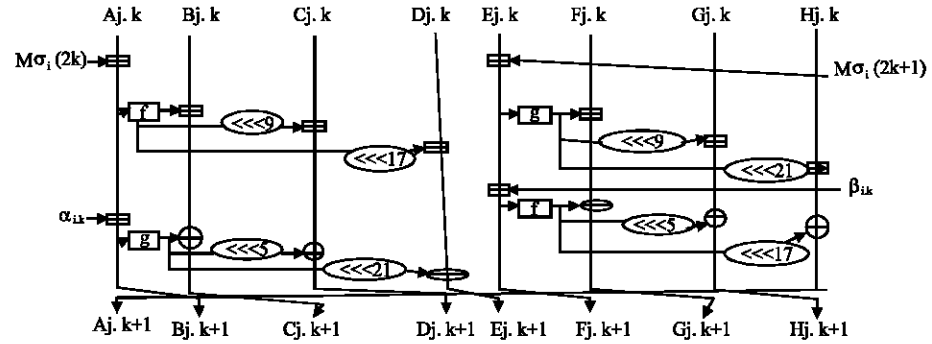

Fig. 2: The compression function of GCSEC

Branch computation: The compression function of GCSEC sixteen constants given by the following Table 2:

$\delta_0 = 428a2f98_x$ $\quad \delta_1 = 71374491_x$
$\delta_2 = b5c0fbcf_x$ $\quad \delta_3 = e9b5dba5_x$
$\delta_4 = 3956c25b_x$ $\quad \delta_5 = 59f111f1_x$
$\delta_6 = 923f82a4_x$ $\quad \delta_7 = ab1c5ed5_x$
$\delta_8 = d807aa98_x$ $\quad \delta_9 = 12835b01_x$
$\delta_{10} = 243185be_x$ $\quad \delta_{11} = 550c7dc3_x$
$\delta_{12} = 72be5d74_x$ $\quad \delta_{13} = 80deb1fe_x$
$\delta_{14} = 9bdc06a7_x$ $\quad \delta_{15} = c19bf174_x$

These constants are applied to each BRANCHj according to the ordering rule of them as follows:

f and g are nonlinear functions (https://www.cingular.com/media/text) as follows:

$$f(x) = x + (x<<<7 \oplus x<<<22),$$
$$g(x) = x \oplus (x<<<13 + x<<<27).$$

DEVISE THEORY

Structure GCSEC consists of 5 Branches. In the security facet, we can give the security against known attacks with the different message-ordering in branches. For example, RIPEMD, which consists of 2 branches, was fully attacked by Wang because RIPEMD has same message-ordering in 2 branches. On the other hand, in case of RIPEMD-128/160, there is no attack result because RIPEMD-128/160 have different message-ordering in branches. In the implementation aspect, GCSEC can be implemented efficiently because the message-ordering is simpler than the message expansion such as that of SHA-256 (Cellular Online, 2004) (Fig. 3).

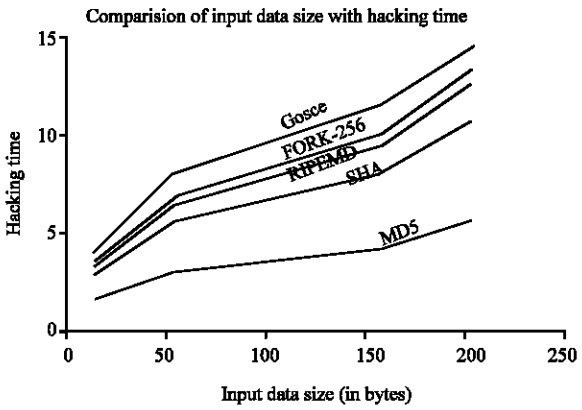

Fig. 3: Relationship of hacking speed

Constants: Each BRANCHi uses 16 different constants ái,j and âi,j for j =0, • • •, 7. By using constants we pursue the goal to disturb the attacker who tries to find a good differential characteristic with a relatively high probability.

Nonlinear functions: Nonlinear functions f and g output one word with one input word. Almost dedicated hash functions use boolean functions which output one word with three words at least. The boolean functions make it easy to control the output one word by adjusting the input several words. In addition, the output words of f and g functions are used to update other chaining variables. In almost dedicated hash functions output words of boolean functions are used to update only one chaining variable (CERT, 1996).

Ordering of message words: We espouse the message word ordering instead of the message word extension. If an attacker constructs an intended differential characteristics for one branch function, the ordering of message words will cause unintended differential patterns in the other branch functions (Cingubar Wireless). This is the core part of the security in the compression function.

GCSEC in hacking:

- Preimage and second preimage resistance is 2^{256}
- Birthday attack needs 2^{128}
- Probabilty to have Collision resistance is 2^{-256}
- Parallel operation is used to enhance the refuge.

SIMULATION

The hash algorithms effectively encrypt the messages and send them through mobile (CERT, 1996). The algorithm is developed in C and analyzed in MATLAB 6.1 for computers and mobile networks to encrypt and encrypt and decrypt the E-mail, Fax, SMS (Cisco Systems White Paper, 2004) and other real time data (Fig. 4).

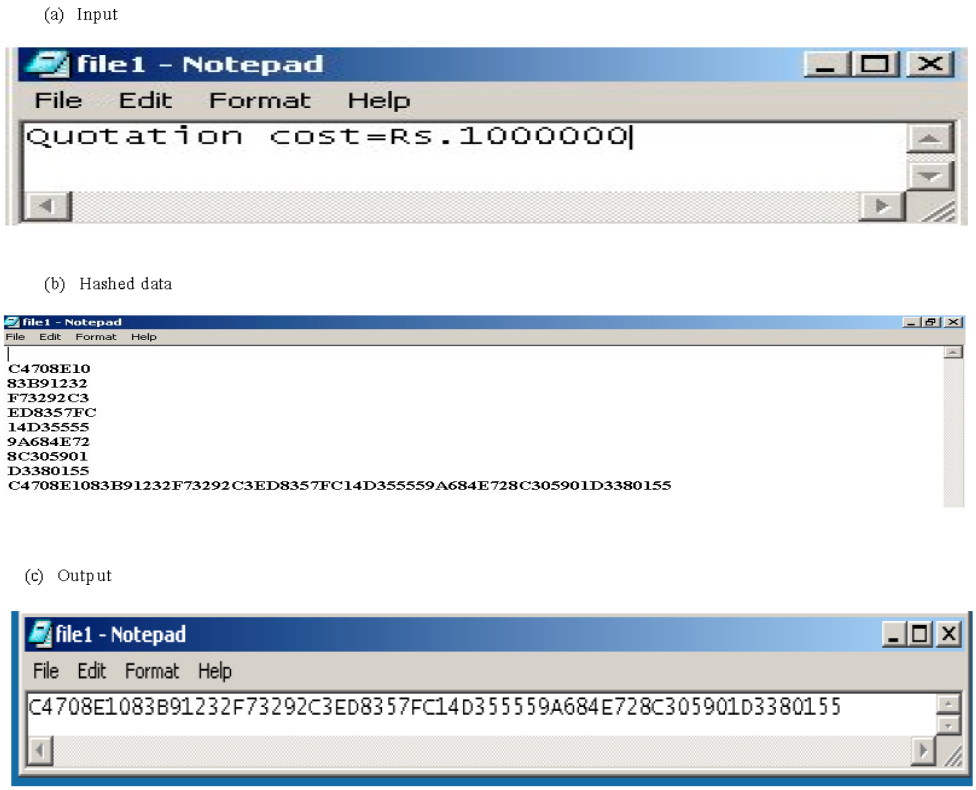

Fig. 4: The hash algorithms

CONCLUSION

All algorithms are compared with GCSEC algorithm. This algorithm works faster than others. The complexity of this algorithm is increased by introducing so many number of steps for shifting the left and right parts of the bits of the data. GCSEC is having highest level of privacy hence any hackers can't easily attack the non real time data. This algorithm can be used for sending or receiving non real time data in wireless or mobile networks.

ACKNOWLEDGMENT

The authors express their sincere thanks to the Principals of K.L.N. College of Engineering and MEPCO Schlenk Engineering College for their co-operation and constant encouragement.

REFERENCES

Biham, E. and R. Chen, 2004. Near Collisions of SHA-0. Advances in Cryptology CRYPTO 2004, LNCS 3152, Spriger-Verlag, pp: 290-305.

Bellovin, S., 1989. Security problems in the TCP/IP protocol suite. Comput. Commun. Rev., 19: 32-48.

Cellular Online, 2004. Uk sms traffic continues to rise. http://www.cellular.co.news.

CERT. Advisory CA-1996-26 'denial-of-service attack via ping'.http://www.cert.org/advisories/CA.

Cisco Systems Whitepaper, 2004. A study in mobile messaging: The evolution of messaging in mobile networks andhow to efficiently and effectively manage the growing messaging traffic. Technical report.

NIST/NSA, 2002. FIPS 180-2: Secure Hash Standard (SHS).

Rivest, R.L., 2004. The MD5 Message Digest Algorithm. IETF Request for Comments.