IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 3, March 2018

Anti Collision Data Ciphering of AES With Hash Function Enhancement and Implementation on LabView

Aya Qusay Dawood[1], Raaed Khalid Ibrahim[2]

Computer Deptartment, College of Engineering, Baghdad, Iraq[1,2]

Abstract: This paper presented an implementation of Advanced Encryption Standard (AES) and Secure Hash Algorithm (SHA) by using the LabView software environment. The proposed algorithm in this work is to combine the AES algorithm and the Secure Hash algorithm (SHA-1) together. The resulting hash code output was used as the input key for the AES in order to improve data security and also to make the SHA-1 two way encryption algorithm. SHA-1 is a 160 -bit key so, the AES key used is a192-bit where the SHA-1 would take 160bit from the AES key size and the rest of the 192-bit of the AES key would be used as a salt. From the results of SHA-1 and AES algorithms obtained in LabView, the suggested work showed the simplicity in modeling hashing and AES algorithms, generating hash codes and plain texts in English plain text(small and capital letters), Arabic plain text, symbols, and numbers.

Keywords: Advanced Encryption Standard AES; Secure Hash Algorithm SHA-1; Security; LabView.

I. INTRODUCTION

Advanced encryption standard (AES) was generated by two Belgian cryptographers, Vincent Rijmen and Joan Daemen. Advanced Encryption Standard use the same key for the encryption and decryption so it is a symmetric block cipher. AES is different from Rijndael where AES has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits, by disparity, the Rijndael creation per second is specified with a block and key sizes that may be any multiple of 32 bits, both with a minimum of 128 and a maximum of 256 bits [1]. Secure hash algorithm (SHA) is used to detect changes to messages and it is a one way cryptographic function [2]. SHA-1 also known as a cryptographic hash function is a primitive or building block utilized in the schemes that used to provide information security, a cryptographic hash function converts an input data of arbitrary length into a fixed-length output [3].

II. AES ALGORITHM USING LABVIEW

The AES algorithm was implemented by using LabView 2013, by using the ECB mode. The input message is processed and the key of the AES is processed through the SHA-1, where the code generated by the Hash function will be the input key of the AES algorithm so, the code generated by the AES gives a very secure code that cannot be breakable easily. The procedure of AES algorithm is shown in figure 1.

SHA-1 itself does not achieve confidentiality because it only achieves integrity and authentication, to make the hash more secure there is a need to merge it with any encryption method. The AES algorithm implemented in LabVIEW to realize the confidentiality property to support the security of the system [4]. The message that processed by the SHA-1 algorithm, represent the plain text. The hash code that produced via SHA-1represent the password of the AES algorithm to producwe another code which is very difficult to break as shown in the figure below.

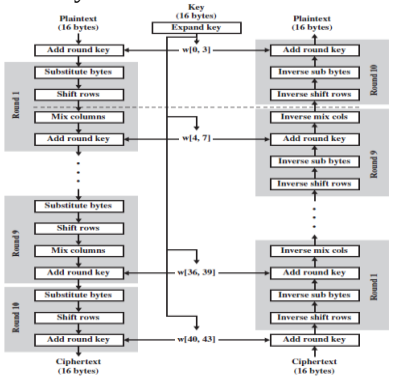


Fig. 1: AES algorithm [2]

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 3, March 2018

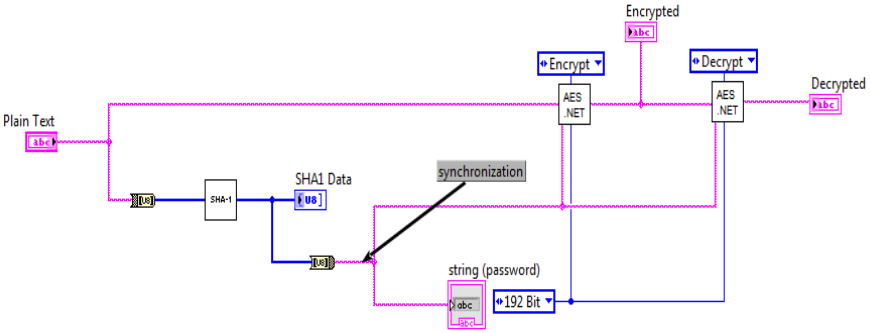


Fig. 2: Work design

Figure 2 shows that the plain text which is the original message entered to SHA-1 and AES is called plain text, it passes the whole system operations. Key size used for AES is 192 bit length, SHA-1 uses 160 bit key length. The hash code generated by the SHA-1 produced the key of the AES, the key then enters to AES encryption block as an input dynamic key to give an encrypted message, then the encrypted message enters to AES decryption block to return the plain text which is the original message that was entered first.

III. SHA-1 (128 BIT) ALGORITHM BY USING LABVIEW

Secure hash algorithm SHA-1 steps are implemented by using LabView which has determind the LabView environment capabilities for effective implementation of cryptographic algorithms. The procedure of SHA-1 is shown in figure 3.

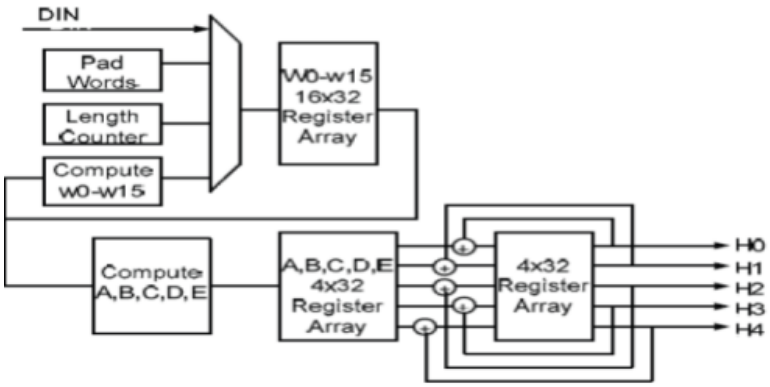
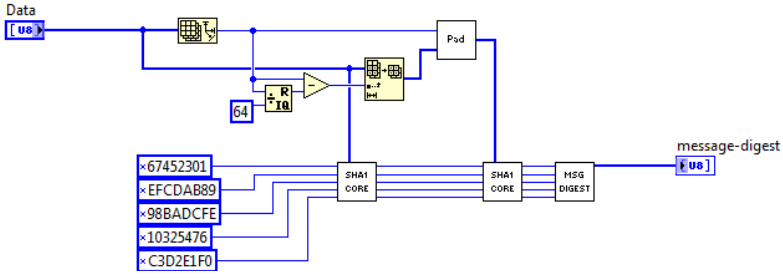


Fig. 3: SHA-1 algorithm [5]

From figure 3 SHA-1 is used to compute the hash code. The size of hash code 160 bits, the message length is divided into chunks, each with size of 512 bits. Every chunk is processed to produce five buffers which are A, B, C, D and E. The processed algorithm was implemented by LabView where the message was entered through SHA-1 block and processed to give a message digest, as in figure 4.

Fig. 4: SHA-1 by LabView

The SHA-1 block works as black box that produce many calculations that are divided into three stages which are padding, SHA-1 core and message digests as shown in figure 5.



IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 3, March 2018

Fig. 5: SHA-1 block diagram

A. Padding

The message is padded to make the total length of a padded message congruent to 448 modulo 512(length = 448 mod 512). The number of padding bits should be between 1 and 512. Padding made up of a 1-bit followed by the necessary number of 0-bits. Given an m-bit message, a 1 bit is appended as the m + 1th bit and then (448 (m + 1)) mod 512 (between 0 and 511) zero bits are appended. As a result, the message becomes 64-bit short of being a multiple of 512 bits long[6].

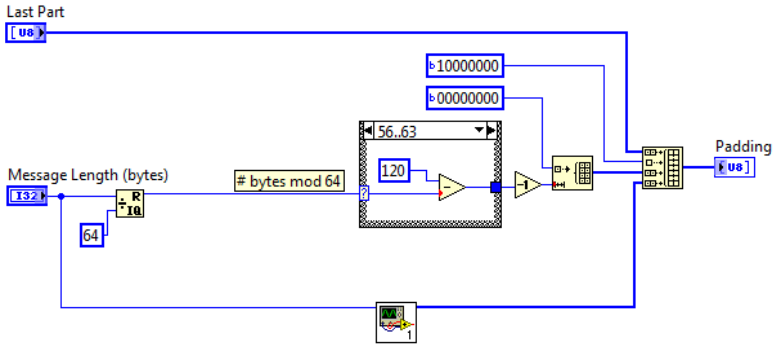


Fig. 6: SHA-1 padding

There are two stages in the padding procedure, as shown in figure 6. When the message enters the loop it is converted to bytes, the message enters chunk by chunk which means each 512 bits are processed alone in sequence, but enters as a byte "64 byte" and that is the first stage. While the second stage is that the message shifted left three times and then enters to the last block which has four options, only one released to produce the padded message [7]

B. SHA-1 core [7]

SHA-1 Core holds the whole system computation and functions, the initial vectors and the padded message are the input to SHA-1 core. The initial vectors (IV) are (H0, H1, H2, H3, H4) are initialized as fixed constants, as shown in figure 7.

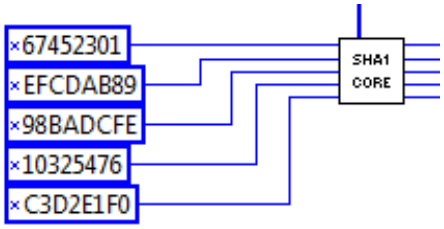


Fig. 7: Initial Vectors

The next step in the core is that SHA-1 consists of 4 rounds, each containing 20 iterations (80 iterations in total). The algorithm operates on a 128-bit state, divided into four 32-bit words [8].
The first operation is the allocation of 80 words array for the message schedule to the 80 rounds, as shown in figure 8.

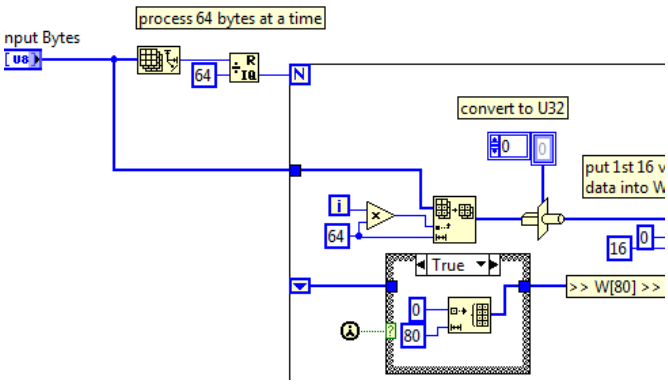


Fig. 8: 80 Words array allocation (true state)

The input bytes of figure 6 are the padded message bytes (512 bits but enters the loop as bytes). The block diagram in figure 8 built by LabView, it shows the case structure with boolean type, in the true case the output gives 80 words with

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 3, March 2018

array representation. In the false case the function inside the case structure replaced by a single wire and that means the structure works if and only if the first call function works to give a true state, figure 9 shows the false state.

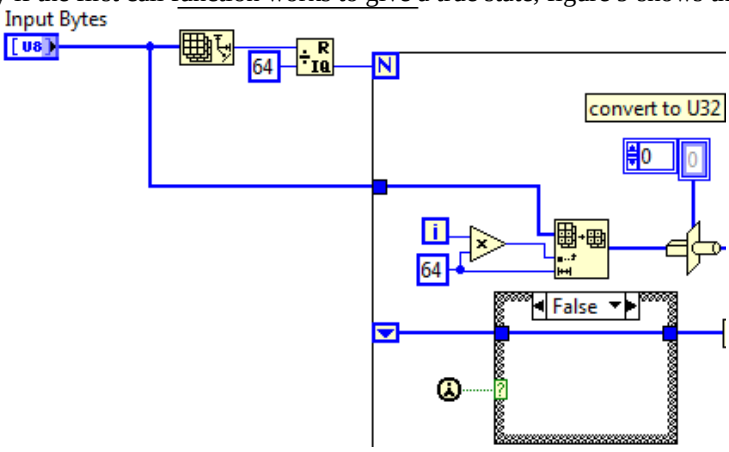


Fig. 9: 80 Words array allocation (false state)

After allocating the 80 words array, the procedure comes successively, set the first 16 words to be the 512 bit block split into 16 words, the rest of the words are generated using the logical function:

Word [i8] XOR word [i3] XOR word [i14] XOR word [i16], and then rotated 1 bit to the left, as shown in figure 10. The LabView block diagram shown in figure 10 shows that the 8th, 3rd, 14th, and 16th words XORed and rotated to left.

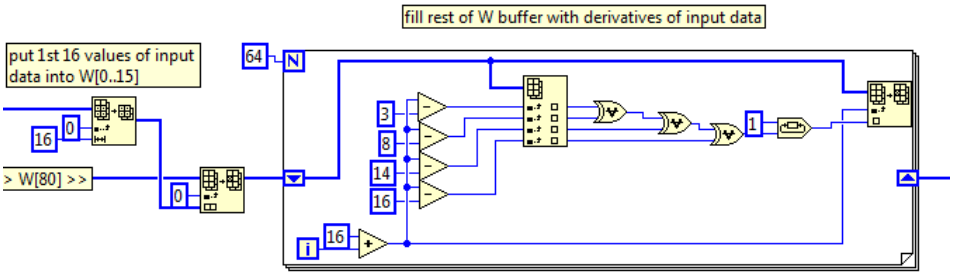


Fig. 10: words array iterations

Next the constants and the functions are identified, a sequence of logical functions f (0), f (1)... f (79) is used in SHA-1. Each f (t) where 0 <= t <= 79, operates on three 32-bit words B, C, D and produces a 32-bit word as output.f (t;B,C,D) is defined as follows [9] [10]:

$$F(t;\ B, C, D) = (B\ AND\ C) OR\ ((NOT\ B) AND\ D) (0 \leq\ t\ \leq 19) \quad (1)$$

$$F(t;\ B, C, D) = B\ XOR\ C\ XOR\ D\ (20 \leq\ t\ \leq 39) \quad (2)$$

$$F(t;\ B, C, D) = (B\ AND\ C) OR (B\ AND\ D) OR (C\ AND\ D) (40 \leq t \leq 59) \quad (3)$$

$$F(t;\ B, C, D) = B\ XOR\ C\ XOR\ D\ (60 \leq\ t\ \leq 79) \quad (4)$$

These operations are implemented in LabView blocks as shown in figure 11.

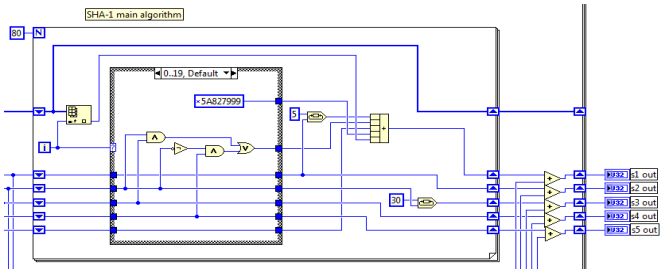


Fig. 11: SHA-1 Calculations

A sequence of constant words K (0), K (1), ... , K (79) is used in the SHA-1. In hex these are given by:

$$K(t) = CA62C1D6\ (60 \leq t \leq 79) \quad (5)$$

$$K(t) = 8F1BBCDC\ (40 \leq t \leq 59) \quad (6)$$

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 3, March 2018

K (t) = 6ED9EBA1 (20≤ t ≤39) (7)
K (t) = 5A827999 (0≤ t ≤19) (8)

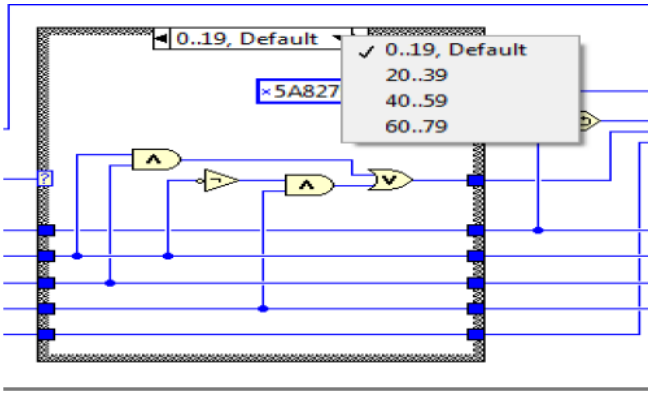


Fig. 12: SHA-1 functions with LabVIEW Case Structure

Figure 12 shows that as default the first function is:
f(t;B,C,D)= (B AND C)OR ((NOT B)AND D)(0≤ t≤ 19) (1) with constant value $K(t) = 5A827999$ for period (0≤ t≤ 19), the next case of function and constant changed by the second iteration from 19 to 20 based on t value, as shown in figure 13. The same procedure to the next iterations.

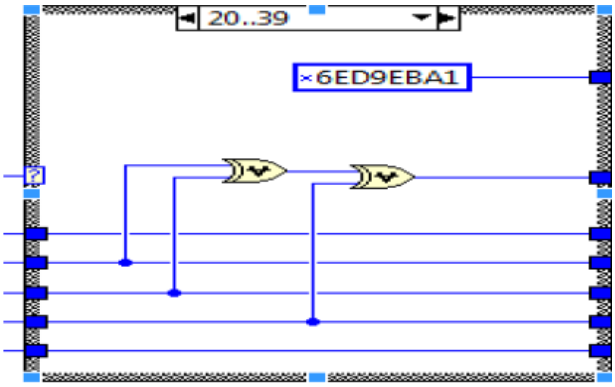


Fig. 13: the second round in the case structure

C. Message Digest: [7]
The message digest is the last stage of the SHA-1 core, which is the last calculation to give the final code (hash code), as shown in figure 5 the output vector is input of the message digest. The message digest is computed using the padded message. The message digest block diagram is shown in figure 14. Two buffers are used to describe the computation, each buffer consist of five 32-bit words, and a sequence of eighty 32-bit words. The first 5-word buffer are labeled A, B, C, D, and E. The second 5-word buffer are labeled H0, H1, H2, H3, H4. The 80-word sequence are labeled W(0),W(1),...,W(79). A single word buffer TEMP is also employed. [10]

The 16-word blocks M(1),M(2),...,M(n) are processed in order, to generate the message digest, the processing of each M(i) involves 80 step.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 3, March 2018

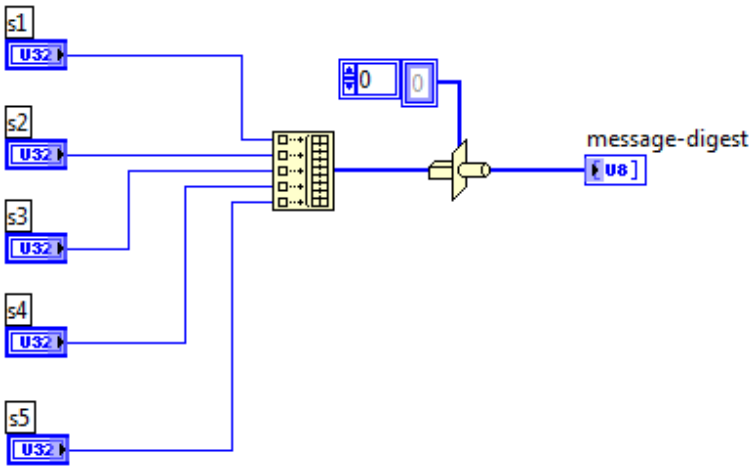


Fig. 14: Message Digest

Now M(1),M(2),...,M(n) are processed. To process M(i) proceed as follows:

1) Divide M(i) into 16 words W(0),W(1),...,W(15), where W(0) is the left-most word.
2) For t = 16 to 79 let W(t) = S^1(W(t-3) XOR W(t-14) XOR W(t-8) XOR W(t-16)).
3) Let A = H0,B = H1,C = H2,D = H3,E = H4
4) For t = 0 to 79 do TEMP = S^5(A) + f(t;B,C,D) + E + W(t) + K(t)
E = D; D = C; C = S^30(B) ,show figure 7.; B = A; A = TEMP
Let H0 = H0 + A,H1 = H1 + B,H2 = H2 + C,H3 = H3 + D,H4 = H4+ E

The message digest is the 160-bit string represented by the 5 words : H0 H1 H2 H3 H4 and this result achieved after processing M(n)[10]. From figure 14 it is distinct that the inputs are the output of SHA-1 core as illustrated in figure 11 ,these inputs will be processed by a function and changes its type to Hex to give the last representation of the code which is the hash code (digest) with 160 bits length.

IV. RESULT

From the implementation of cryptographic hash function and AES in LabVIEW 2013 and test the execution of all steps in AES and SHA-1 algorithm in different types of plain text such as English (small letters and capital letters), symbols and numbers to produce fixed 160 bits hash code and AES cipher text. Figure 15 shows the hash code.

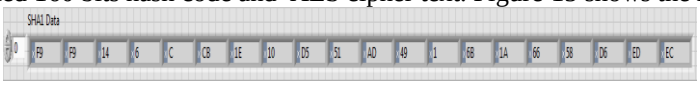


Fig. 15: Output of SHA-1 (message digests)

The testing of the proposed system in small letters is shown in figure 16.

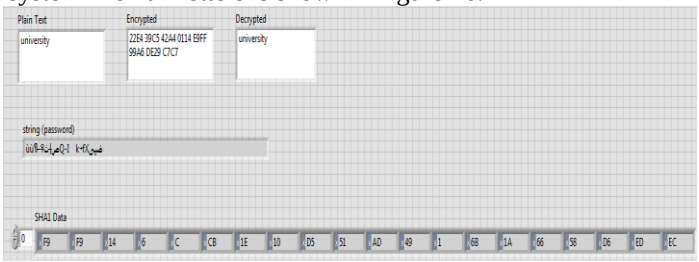
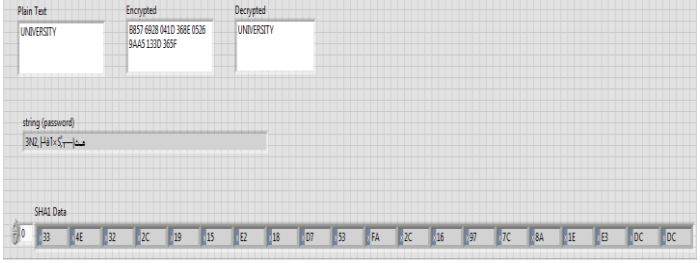


Fig. 16: AES and SHA-1 system(small letters)

The testing of the proposed system in capital letters is shown in figure17.



ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 3, March 2018

Fig. 17: AES and SHA-1 system(capital letters)

The testing of the proposed system in symbols and numbers is shown in figure 18.

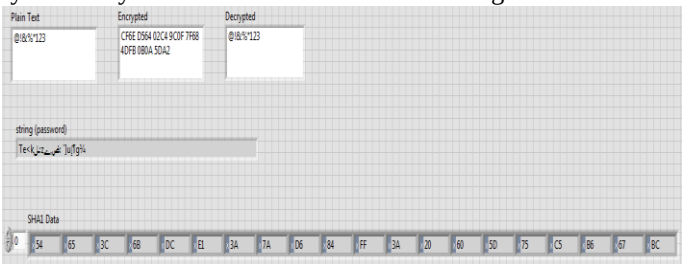


Fig. 18: AES and SHA-1 system(symbols and numbers)

After generating the 160bit Hash code, these bits are entered to the AES algorithm which is 192 bit key to both encryption and decryption to give a very strong code which is very difficult to break. The encrypted hash code gives more security to the system.

The text entered to the system is "university" which is 10 bytes in size, after processing via SHA-1 algorithm and AES-192bit the message digest will be:F9F9 146CCB1E10D551AD4916B1A6658D6EDEC, as shown in figure 15. The message digest change if any change happened to the text and thats the most important property of the system, and also if any letter changed from uppercase to lowercase and vice versa this is called breakdown. From figure 16 note that according to AES algorithm of 192 bits the cipher text is: 22E4 39C5 42A4 0114 E9FF 99A6 DE29 C7C7.This cipher text can be reversed to return back to the original plain text which is "university". From figures 16,17 and 18 the result show that any changes to message entered, the message digest changed so as key entered to the AES.

V. CONCLUSION

From the result of SHA-1 merged with AES built via LabView are concluded the following points:

(1) The process speed of LabView to implement SHA-1 algorithm and AES is acceptable with respect to other software tool and easy way to compute the run time as well.
(2) From the proposed implementation of SHA-1 as well as AES encryption and decryptipn method in LabVIEW it is easy to control and change the coefficient parameters in SHA-1 algorithm and AES algorithm due to the simplicity and flexibility of LabView tasks, so it is easy to compute the hash codes and generate encrypted plain texts of English and other languages in our proposed system.
(3) Merging between the AES and SHA-1 algorithms made the system powerfully built via LabView and disposes of the probability of collision that may occur with SHA-1seperately.This make the system very difficult to break and using of stream cipher add the two way property to the system and the merging between the hashing and cryptography support the security of any system apply the merging algorithms.
(4) Since SHA-1 used only 160 bit from the 192 bit of the AES key, the rest of the 192 bit was used as a salt to the AES, this made the system more secure.

REFERENCES

[1] Sahoo O.B., Kole D.K, and Rahaman H, "Optimized S-Box for Advanced Encryption Standard (AES) Design," Advances in Computing and Communications (ICACC), International Conference, pp. 154-157, 9-11Aug. 2012.
[2] William Stallings, Cryptography and Network Security Princibles and Practice 5th Edition.
[3] Olakunle Esuruoso, "High Speed FPGA Implementation of Cryptographic Hash Function," 2011.
[4] Raaed K. Ibraheem, Roula AJ. Kadhim, and Ali SH. A, "Anti-Collision Enhancement of a SHA-1 Digest Using AES Encryption By LABVIEW," pp. C27-C31, 2015.
[5] Ge F, Jain P, and Choi K, "Ultra-Low power and High Speed Design and Implementation of AES and SHA1 Hardware cores in 65 Nanometer CMOS," Electro/Information Technology, pp. 405-410, 2009, IEEE International Conference.
[6] Nalina H D, Shruthi T M, Spoorthi Y, and Thilagava , "ANTI COLLISION ENHANCEMENT OF SHA-1 USING AES ENCRYPTION," in 2nd Inernational Conference on"Innovative Trends in Science ,Engeneering and Management", New Delhi,India, 2016, pp. 252-266.
[7] Raaed K. Ibrahim, Ali SH. Hussain, and Roula A. Kadhim, "IMPLEMENTATION OF SECURE HASH ALGORITHM SHA-1 BY LABVIEW," International Journal of Computer Science and Mobile Computing, vol. 4, no. 3, pp. 61-67, march 2015.
[8] Rubina B. and PatelNaveen Chaudhary, "analyzing digital signature robustness witm message digest algorithm," IJCA Special Issue on Communication Security, 2012.
[9] Lawrence C. Washington, Introduction to cryptography with Coding Theory.Wade trappe:cmputer and network labrotary.
[10] D. Eastlake and P. Jones , "US Secure Hash Algorithm 1 (SHA1)," Network Working Group.3rd Motorola, Cisco Systems, September 2001.