Efficient Implementation of DILH hash Algorithm in 8-bit Microcontroller

Rushdi A. Hamamreh

Computer Engineering Department, Al-Quds University

Abstract
Hash function is the foundation of digital signature and message authentication for the assurance of data integrity due to its collision-free and one-way properties. To receive correct data that sent as a plaintext between a remote terminal units through wired network or a wireless networks. Remote terminal unit is a device that responsible for retrieving occur data. In this paper, we describe the addition of the function signature, with the purpose of data integrity for terminal units. We use a programmable unit (Microcontroller) to implement DILH technique for one way hash algorithm. The Hill cipher requires the inverse of the matrix to recover the plaintext from cipher text. In the remote terminal unit developments DILH based on Hill cipher for generating only non invertible matrix to assure the accuracy of transmission data [1][12]. The result of this research implemented in an 8-bit microcontroller with 100% accuracy , data input using keypad matrix 4x4, DILH hash output is displayed on the LCD graphics 20x4 and can only enter data input capital letters only.

Keywords:
Hash function, Data integrity, DILH,information security, Networks, Microcontroller, Arduino kit.

1. Introduction

Message verification is important for information security, and the hash-based algorithm is a kind of implementation method.

A programmable unit is often implemented in world manufacturing industry either part of an electronic product, such as a remote terminal unit. Remote terminal unit (client) is an electronic device that responsible for retrieving data to a server, through a medium.

Problems can occur when the data is sent as a plaintext (not secure). Such data can be read by unauthorized persons in various ways [2].

So it needs to keep data secure and communicating in secure manner, It's led to development of Cryptography.

The main purpose of Cryptography is securing and enabling communication between two parties and protecting the data, sensitive data or information from outside attacks. In this context cryptography is based on four specific security requirements: authentication, integrity, privacy and non-repudiation. So the role of cryptography is not only data protection but also provide authentication, there are generally three cryptosystems used to achieve this: symmetric algorithms, asymmetric algorithms and hash algorithms. While symmetric and asymmetric cryptosystem is used for enciphering and deciphering the hash function are used for authentication. The original text is called plaintext and enciphered text is called cipher text. Otherwise the process of deciphering is inverse of enciphering, it has as input the enciphered text and gives as output the plaintext (original text). Both cryptosystems depend on a key, and the difference is that at symmetric cryptosystems one key is used for enciphering EK (P) = C and deciphering DK(C)=P, E enciphering function, D deciphering function , K key and P plaintext [3][6], cryptographic hash algorithms MD5, SHA , DLHI are one-way hashing functions which are easier to compute but are much harder to reverse [13].

In this research will discuss the third cryptosystem mentioned is hash algorithms , with the purpose of data integrity and implement it on 8-bit microcontroller-board, based on Arduino Uno kit.

2. Hash Functions

2.1. Definition and description

It is a function that accepts a variable input and give fixed output with non reversible property. These functions are called hash functions and in real world are built on the idea of a compression function. The inputs to the compression function are a message block mi and the output is hash value to that point vi. That is

$$H(m_i) = v_i$$

2.2. Hill Cipher

The core of Hill-cipher is matrix manipulations. It is a multi-letter cipher, is a type of monoalphabetic polygraphic substitution cipher. Hill cipher requires inverse of the key matrix while decryption. In fact that not all the matrices have an inverse and, therefore they will not be eligible as key matrices in the Hill cipher scheme [6][9].

Manuscript received August 5, 2017
Manuscript revised August 20, 2017

Moreover, Hill cipher has several advantages such as disguising letter frequencies of the plaintext (M), its simplicity because of using matrix multiplication and inversion for enciphering and deciphering, its high speed, and high throughput.

A. Hill Cipher (Encryption , Decryption)

To Encrypt Plaintext Block size of m [9], we need key matrix ($K_{n\times n}$) with entries are between $(0, q-1)$ included, but the determinant must be relatively prime to p, each entry in the plaintext block is between $(0, q-1)$, included each block of plaintext is then an n-dimensional vector m .We encrypt vector m simply to produce the cipher text vector $\boldsymbol{c}$ using the following linear algebra equation:

$$c = m \times k \; mod \; N$$

To Decrypt cipher text vector $\boldsymbol{c}$, we need first to find the inverse matrix k^{-1} to k , where that matrix must be invertible . Then can calculate m from the mathematical model

$$m = c \times k^{-1} \; mod \; N$$

In [5][9] they proposed new technique to convert any non-invertible matrix's to invertible ones. As a result, Hill cipher being a efficient algorithm because any encrypted text will decrypted using the key matrix [4][5][11].

B. Hill Cipher (Hashing algorithm)

The main point of one-way hash algorithm is that any encrypted text cannot be decrypted [1]. From this point, we need to choose the non-invertible matrix from the hill cipher to use it inside the practical one- way hash algorithm.

$$H(m) = m \times k \; mod \; N$$

$$H(m_i) = v_i$$

Where k is the non-invertible matrix. In [1] author works on an algorithm that generate non-invertible matrix and multiply it by plaintext as column vector with modular value N to generate the hash value v_i.

3. Proposed algorithm

Proposed algorithm "Data Integrity using Linear Combination for Hash Algorithm" (**DILH**) uses non-invertible matrix to produce hash value V. This algorithm selects and generates non-invertible matrixes using ***Linear combination***[1] of rows or columns of a matrix to ensure that the hash values are collision-free and one-way properties.

In this section, we plot the diagram for our proposed algorithm which showing each step inside it and how it works , It is also showing the mathematical proof of our work. Besides that, we have the DILH algorithm analysis and result and its comparison with other hashing algorithms including SHA-1,MD5 [6].

DILH algorithm

According to figure 2, the step of DILH algorithm structured as the followings:

Step1 (Input): input M.

Step2 (Padding): Pad (P) , $P(M)$.

Step3 (Splitting): M is split intoq, q is the number of blocks. blocks $(m_1, m_2, \ldots, m_i)$, each of length $n \times n$ suitable for the hashing block.

Step4 (Key generation): Key matrix generation k_i :

4.1 Generate a random matrix (R) with size $(n-1, n)$ and value in a interval (0,1).

4.2 Casting (R).

4.3 Introduce a new row in the matrix R so as the size of the resulting matrix k_i is $n \times n$. The added row is obtained by ***linear combination*** of row i and row j of matrix R according to:

$$\boldsymbol{K(N,:) = c_1 \times R(i,:) + c_2 \times R(j,:)}$$

where c_1 , c_2 are arbitrary integer constants. This ensure that the inverse of K denoted by K^{-1} does not exist.

Step5 (GenerationV_i): Hash value V_i generation using this formula:

$$\boldsymbol{H_{xi} = m_i \times k_i = v_i}$$

Step6 (Digest): Digest V_i.

4. System implementation

In this paper, we propose a circuit implementation scheme to realize message verification, which is based on DHLI, circuit module is reusable such that the circuit size is reduced and the processing speed is improved [5][14].

4.1. Components of the system:

- Arduino UNO
- LCD 20x4 pixel
- Matrix keypad 4x4

4.1.1 Arduino UNO

The Arduino Uno is a microcontroller board . It has 14 digital input/output pins , 6 analog inputs,
It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with battery to get started [5].

- Microcontroller
- Operating Voltage: 5V
- Input Voltage : 6-20V
- Digital I/O Pins: 14
- Analog Input Pins: 6
- DC Current per I/O Pin: 40 mA
- Flash Memory: 32 KB
- Clock Speed: 16 MHz

4.1.2 LCD 20x4 pixel

J204A is an industrial character type LCD, can also shows that 20 x04 namely 80characters, (20 column 4 line).

Liquid crystal display module is a slow display device, so in the execution each instruction before must affirm module mark is busy low electricity Flat, said were not busy, otherwise this instruction failure. To display character before display character input address, also is tell module in Where display character.

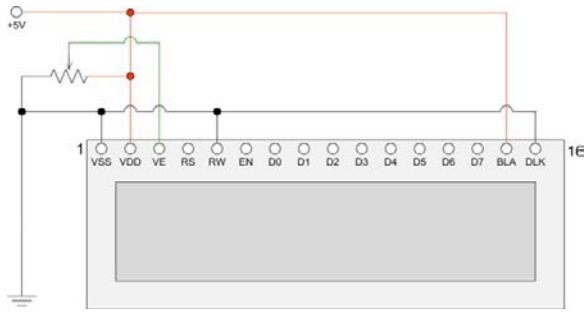

Fig. 1 LCD 20x4 pixel

4.1.3 Matrix keypad 4x4

The 4x4 Keypad is a general purpose 16 button (4x4) matrix keypad. It comes ready to work

simply peel-off the adhesive backing, stick it to your surface and plug it in [5]

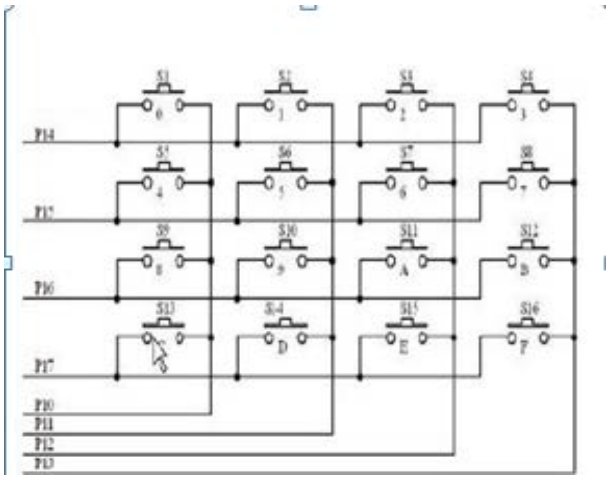

Fig. 2 Keypad matrix 4x4 schematic

4.2. Design and Scenario

4.2.1 Arduino open hardware

Arduino is an open source physical computing platform based on a simple input/output (I/O) board connected to software on your computer (such as Flash ,Processing) The boards can be assembled by hand or purchased preassembled; the open source IDE (Integrated Development Environment) can be downloaded for free from www.arduino.cc.".

4. 2.2 Hash DILH Algorithm

The DILH algorithm takes as input a message of arbitrary length and produces as output a 128-bit typically expressed in text format as a 16 digit hexadecimal number.

it is computationally infeasible to produce two messages having the same message The DILH algorithm is intended for digital signature applications, where a large file must be "compressed" in a secure manner before being encrypted.

4. 3 integration

In this research, 8-bit microcontroller use Arduino Uno Kit, display using graphic LCD 20x4 pixel, SPI shield, matrix keypad 4x4, connected to computer [7][9] .

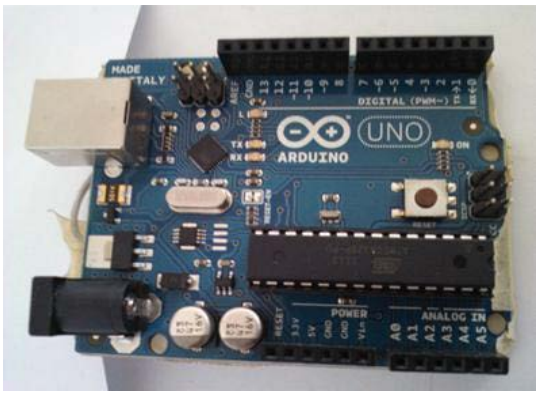

Fig. 3 Arduino Uno Kit

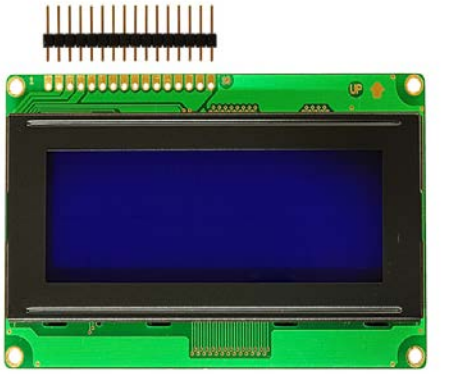

Fig.4 LCD 128x64 pixel

Fig. 5 Matrix keypad 4x4

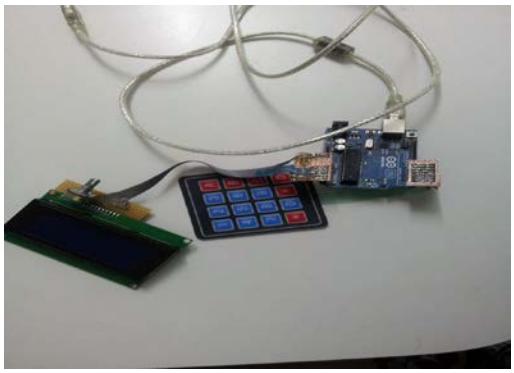

Fig. 6 Integration All Hardware

5. Implementation

Data input use the available keypad.

- First condition, booting (startup):

Fig. 7 LCD 20x4 pixel startup

- Splash screen, initialization condition:

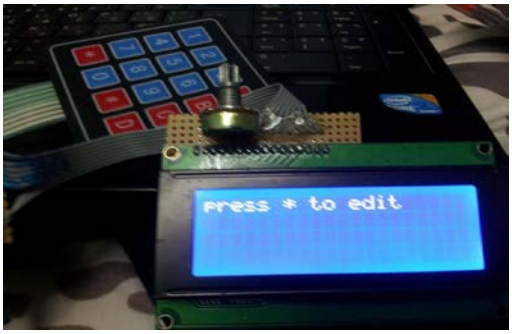

Fig. 8 LCD 20x4 pixel initialization

- Input field:

Fig. 9 LCD 20x4 pixel enter data

- Testing input data if don't equal # show in LCD:

Fig. 10 LCD 20x4 pixel Testing

- Result data if its equal # then hatch data by DILH:

Fig. 11 LCD 20x4 pixel Hatch

5.1 DILH Pseudo-Code for Microcontroller

There is the algorithm pseudo-code of DILH hash function that implement in microcontroller[8][10]:

```
#include <math.h>
#include <Keypad.h>
#include <LiquidCrystal.h>
LiquidCrystal lcd(A5, A4,A3, A2, A1, A0);
const byte ROWS = 4;
const byte COLS = 4;
char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {0,1,2,3};
byte colPins[COLS] = {4,5,6,7};
Keypad customKeypad =
Keypad( makeKeymap(hexaKeys), rowPins, colPins,
ROWS, COLS);
void setup()
{
 lcd.begin(20,4);
 lcd.setCursor(0, 0);
 lcd.print("Starting ...");
 delay(1000);
 lcd.clear();
}
void loop()
{
  lcd.setCursor(0, 0);
lcd.print("press * to edit");
 char a = customKeypad.getKey();
if (a=='*'){
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Enter data :");
char* data="";
while(a != '#'){
a = customKeypad.getKey();
while(! a){a = customKeypad.getKey();}
if (a !='#'){data = data + a; lcd.print(a);delay(300);}
}
unsigned char* hash=DILH ::make_hash(data);
char *
= DILH::make_digest(hash, 16);
free(hash);
lcd.setCursor(0,1);
lcd.print(dilhstr);
free(dilhstr);
}
}
```

6. Analysis and results

In this subsection we will analyze the time delay in the proposed DILH algorithm. Given a range of data files with sizes (8KB – 256KB), we first convert each file into matrix with size between (2x2 – 4x4). Then, we calculate the needed time in second to generate hash values. Table 1 shows the performance of the proposed DILH algorithm in

terms of the required time to generate hashes for different file sizes and with different matrix sizes. According to Table 1, increasing the file size will increase the required time for all matrix size considered.

Table 1: Required time in seconds for the proposed DILH algorithm with different file size and matrix size m=127

NxN	File sizes in kilobyte (*KB*) m=127					
	8	16	32	64	128	256
2x2	*20.0618*	*39.9580*	*84.0139*	*211.8587*	*360.6685*	*679.7405*
3x3	*9.3918*	*24.701*	*36.9153*	*73.4639*	*146.4697*	*306.6645*
4x4	*9.1095*	*18.3049*	*36.8379*	*75.5187*	*152.6781*	*303.3392*

According to the experimental data obtained by test, the efficiency of our hash algorithm is about 2.808 times slower than the efficiency of MD5 and about 5.813 times slower than SHA-1efficiency in 8KB file size. This ratio is increased in an positive relationship with file size, so you can find that our proposed algorithm is efficient 9.794 times that SHA-1 when the file size is 256KB. Figure 3 summarizes the experimental delay time for considered hash algorithms .

we can observe that our hash algorithm still has an advantage in the efficiency.

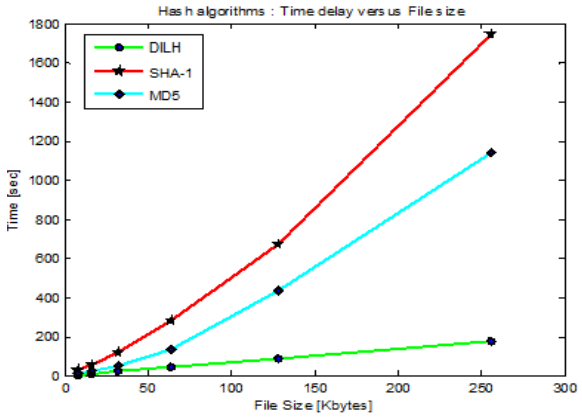

Fig. 12: Hash algorithms comparison (SHA-1,MD5 and DILH) in term of time delay versus file size where m=127.

7. Conclusion and future work

Hash DILH algorithm can be implemented at 8-bit microcontroller, with the result of 100% , data input 16 character (4x4 matrix) using keypad 4x4, DILH hash output is displayed on the LCD graphics 20x4.

Still has limitations on the issue among them, the data can be processed to a maximum of 16 characters, and data input using keypad. Based on these the limitations, on the next research must do improvement about user-interface, implement others encryption algorithm and create the enclosure for the microcontroller .

References

[1] Rushdi Hamamreh , Mohammed Jamoos, Data Integrity Mechanism Using Hashing Verification International Journal of Computer Science and Network Security, VOL.14 No.9, September 2014.

[2] Volodymyr Luzhetsky, Yurii Baryshev. Methods of Generic Attacks Infeasibility Increasing for Hash Functions. The 7th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications 12-14 September 2013, Berlin, Germany.

[3] Artan Berisha, Behar Baxhaku. A Class of Non Invertible Matrices in GF (2) for Practical One Way Hash Algorithm. International Journal of Computer Applications (0975 – 8887) Volume 54– No.18, September 2012.

[4] Songsheng Tang, Fuqiang Liu.A one-time pad encryption algorithm based on oneway hash and conventional block cipher, Conference on Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International, 21-23 April 2012 .

[5] Mochamad Vicky Ghani Aziz, Rifki Wijaya, HASH MD5 Function Implementation at 8-bit Microcontroller, Department Electrical Engineering, SEUS2013-11th

[6] Rivest, Roland. (1992), RFC 1321, The MD5 Message-Digest Algorithm. MIT Laboratory for Computer Science and RSA Data Security, Inc.

[7] Banzi, Masimo. (2011), Getting Started with Arduino, 2nd Edition.O'Reilly.

[8] Evans, Brian. (2008), Beginning Arduino Programming. Technology in Action.

[9] Datasheet. (2006), User's Guide DM12864HLCM (Liquid Crystal Display Module). XIAMEN OCULAR OPTICS CO.,LTD

[10] Coley, Gerald (2009-08-20). "Take advantage of open-source hardware". EDN, Retrieved October 13, 2011..

[11] James S. Plank, Adam L. Buchsbaum. s.l. Some Class of Invertible Matrices in GF(2).: University of Tennessee, 2007.

[12] Design of a robust cryptosystem algorithm for non-invertible matrices based on Hill cipher. Rushdi A. Hamamreh, Mousa Farajallah. 05, s.l. : International Journal of Computer Science and Network Security, 2009, Vol. i 9.

[13] Surbhi Aggarwa, Neha Goyal A review of Comparative Study of MD5 and SHA Security Algorithm, International Journal of Computer Applications (0975 – 8887) Volume 104 – No.14, October 2014

[14] Ning Li, Xiaojun Dang .Realizing High-Speed PBKDF2 Based on FPGA. 2015 International Conference on Intelligent Transportation, Big Data & Smart City