SPL-1 Project Report, 2018

Blockchain

Course: Software Project Lab I

Course No: SE 305

Submitted by

Md. Rakibul Islam

BSSE Roll No. : 0903

BSSE Session: 2016-2017

Supervised by

Nadia Nahar

Designation: *Lecturer*

Institute of Information Technology

Institute of Information Technology

University of Dhaka

30-05-2018

Table of Contents

Index of Figures

1.Introduction 1

"BlockChain" as it's name implies, it stores data transactions or any kind of data in blocks where the blocks are connected together in the form of a chain. The blockchain grows as the number of data transactions grow. Each block contains a "hash" and the hash of the previous block. The previous block's hash links the block together and prevent any block from being altered or inserted between two existing block.

In a block of a blockchain a "Hash"is generated from existing data. The hash is generated by using "Secured Hash Algorithm-256(SHA-256)" .In my project my target is to implement SHA-256 algorithm and to make blockchain by using data of files. Another target is to apply this algorithm for checking image and file security.

1.1 Background study

BLockchain

Stuart Haber and W.Scott Stonetta describe blockchain in 1991.But first it is conceptualized by a person or group known Satoshi Nakamato in 2008.A blockchain is decentralized, distributed that is used to record transactions across many computers so that record cannot be altered.Blockchain is inherently resistant to modification of data. Each block of a blockchain includes the cryptographic hash of the previous block which links two blocks together and form a chain.

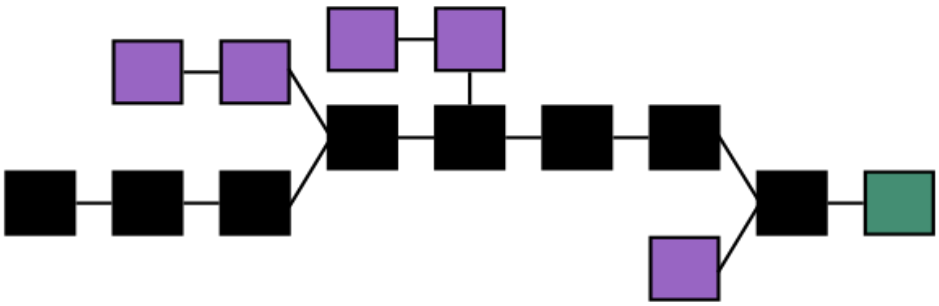

Figure 1: Blockchain(source:Wikipedia)

Hash Function

Hash Function is a mathematical function that converts a numerical input value into another compressed value. The input to the hash function is of arbitrary length but output is always of fixed length. Cryptographic hash has some features

- Fixed length output
- Efficiency of operation
- A small change to a message should change the hash value so extensively that new hash value appears uncorrelated with the old hash value

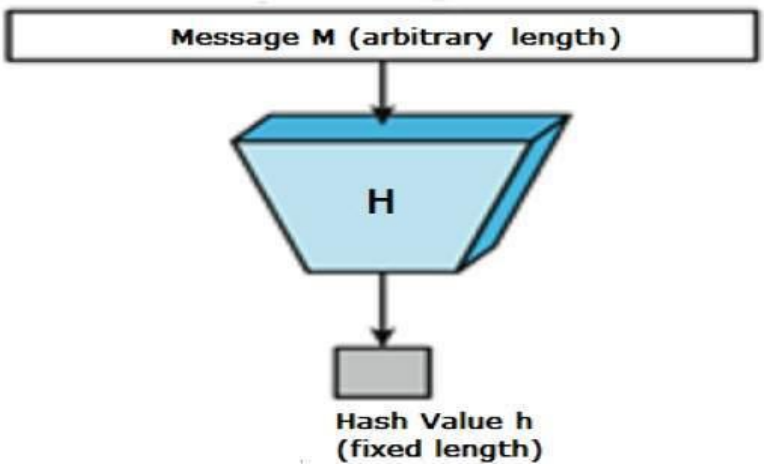

Figure 2: Hash Function(source:Tutorialspoint)

Secured Hash Algorithm-256

SHA-256 is a set of cryptographic hash functions designed by United States National Security Agency (NSA). It was first published in 2001 by National Institute of Standards and technology (NIST). By this algorithm a one way hash can be generated from any piece of data, but data cannot be generated from hash. An initial message is given as input. After being supplemented the message is broken into blocks where each block contains 16 words. Every block message is put by the algorithm through the cycle of 64 or 80 rounds. Two words are rebuilt on each iteration. The transformation function is set by other words. The result of each blocks are added, the sum being the value of the hash function. However inner state initialization is made by the result of the previous block processing. None can independently process blocks and summarize the result.

Bitwise Operations

Bits are individuals ones and zeros that make up everything we do with computers. All the data we use is stored in computer using bits. There are six bitwise operators.

- AND
- OR
- XOR
- Right Shift
- Left Shift
- One's compliment

I have worked with some other bitwise operations like circular left shift, right shift.

bitwise and	0	1	x
0	0	0	0
1	0	1	x
x	0	x	x

bitwise or	0	1	x
0	0	1	x
1	1	1	1
x	x	1	x

bitwise xor	0	1	x
0	0	1	x
1	1	0	x
x	x	x	x

bitwise xnor	0	1	x
0	1	0	x
1	0	1	x
x	x	x	x

bitwise negation	result
0	1
1	0
x	x

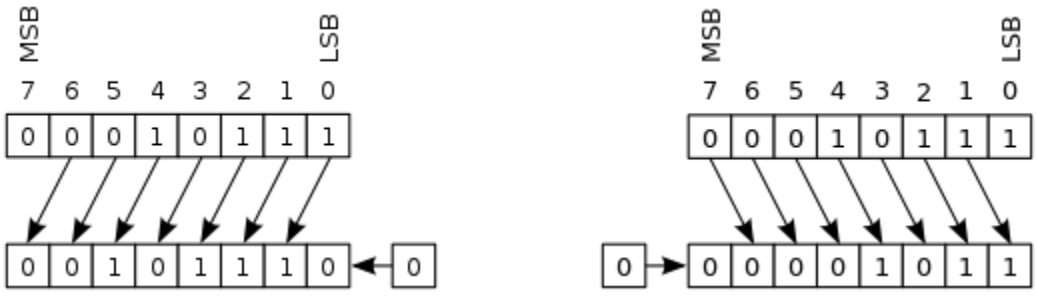

Figure 3: Bitwise operations(source: Wikipedia)

Raw File Reading 4

For reading data of files, I have to study some functions mentioned below

- Opening file
- Reading and writing
- Closing file

BMP File Format

BMP file format is used to store bitmap digital images. The BMP file format is capable of storing two dimensional digital images both monochrome and color. The bitmap image file consists of fixed size structures as well as variable size structure appearing in a predetermined sequence. Many different versions of some of these structures can appear in the file due to the long evolution of this file format.

1.2 Challenges

Implementing a new software solution carries with it a number of challenges. The process can be overwhelming, confusing and lenthy. For implementing this project there are lot of challenges that I have faced. Some of them are

- Handling large code for the first time
- Learning and understanding algorithm
- Designing bitwise operations
- Reading BMP file
- Linked list operations
- Matching digital signatures

2. Project Overview

I have divided my whole project into three different parts. They are

- Implementation of SHA-256 algorithm
- Application of SHA-256 in Blockchain
- Application of SHA-256 in Image security

2.1 Implementation of SHA-256 algorithm 5

This is the key part of my project. In a nutshell, this part can take a string as input and give a hash value as output. This hash value is unique for every message. Even if there is any small change in message it can detect by giving different hash value. This hash value can be generated through a complex process and through many functions. For getting the hash values some important functions are given below in details.

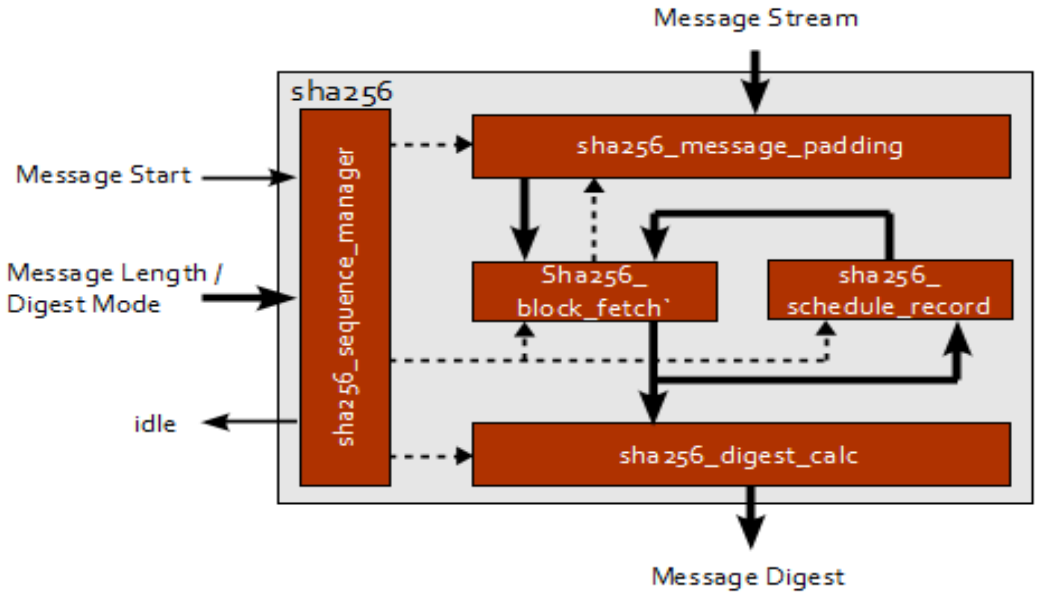

Figure 4: SHA-256 algorithm

CovertToBinary:

This function takes string as parameter and converts every character in binary. While converting it pushes the binary values in a vector and return the vector after finishing last convert.

```
113  vector< unsigned long> convertToBinary(string msg)
114  {
115          vector<unsigned long> converted;
116          for(int i=0;i<msg.length();i++)
117          {
118                  bitset<8>bits(msg.c_str()[i]);
119
120                  converted.push_back(bits.to_ulong());
121          }
122          return converted;
123  }
```


Figure 5: Function: convertToBinary

PaddingZeros:

Target of this function to make a 512 bit length of the binary converted vector. Using formula (k=447-length of converted vector), "k" zeros are added to the vector and at last the length of converted vector is added in binary format. This the total length of the new converted vector will be 512 bits.

```
126  vector<unsigned long> paddingZeros(vector<unsigned long> block)
127  {
128          int lengthOfString=block.size()*8;
129
130          int k=448-1-lengthOfString;
131
132          unsigned long pad1n70=0x80;
133          block.push_back(pad1n70);
134          //unsigned long o=0x00000000;
135
136          for(int i=0;i<k/8;i++)
137          {
138                  block.push_back(0x00000000);
139          }
140
141          bitset<64>len(lengthOfString);
142          string bitsetString= len.to_string();
143
144          bitset<8>temp1(bitsetString.substr(0,8));
145          block.push_back(temp1.to_ulong());
146
147          for(int i=8;i<63;i=i+8)
148          {
149                  bitset<8>len2(bitsetString.substr(i,8));
150
151                  block.push_back(len2.to_ulong());
152          }
153
154          return block;
155
```

Figure 6: Function PaddingZeros

ResizedInto16blocks: 7

This function resizes the 512 bits vector into 16 block where each block contains 32 bits of data.

```
159  vector<unsigned long> resizeInto16Blocks(vector<unsigned long>message)
160  {
161          vector<unsigned long>resizedMsgBlock(16);
162
163
164          for(int i=0;i<64;i=i+4)
165          {
166                  bitset<32> temp(0);
167
168                  temp=(unsigned long)message[i]<< 24;
169                  temp|=message[i+1]<< 16;
170                  temp|=message[i+2]<< 8;
171                  temp|=message[i+3];
172
173                  resizedMsgBlock[i/4]=temp.to_ulong();
174          }
175
176          return resizedMsgBlock;
177  }
```

Figure 7: Function: ResizeInto16Blocks

ComputeTheHashValue:

This is the key function of the code. There are 64 constant values which represent thirty-two bits of the fractional parts of the cube roots of the first sixty-four prime numbers. there are 8 initial hash values which represents the first thirty-two bits of the fractional parts of the square roots of the first eight prime numbers. This function uses bitwise functions and gives the required hash value.

```
193
194  string computeTheHashValue(vector<unsigned long>msg)
195  {
196          unsigned long k[64] = {
197                  0x428a2f98,0x71374491,0xb5c0fbcf,0xe9b5dba5,0x3956c25b,0x59f111f1,
198                  0x923f82a4,0xab1c5ed5,0xd807aa98,0x12835b01,0x243185be,0x550c7dc3,
199                  0x72be5d74,0x80deb1fe,0x9bdc06a7,0xc19bf174,0xe49b69c1,0xefbe4786,
200                  0x0fc19dc6,0x240ca1cc,0x2de92c6f,0x4a7484aa,0x5cb0a9dc,0x76f988da,
201                  0x983e5152,0xa831c66d,0xb00327c8,0xbf597fc7,0xc6e00bf3,0xd5a79147,
202                  0x06ca6351,0x14292967,0x27b70a85,0x2e1b2138,0x4d2c6dfc,0x53380d13,
203                  0x650a7354,0x766a0abb,0x81c2c92e,0x92722c85,0xa2bfe8a1,0xa81a664b,
204                  0xc24b8b70,0xc76c51a3,0xd192e819,0xd6990624,0xf40e3585,0x106aa070,
205                  0x19a4c116,0x1e376c08,0x2748774c,0x34b0bcb5,0x391c0cb3,0x4ed8aa4a,
206                  0x5b9cca4f,0x682e6ff3,0x748f82ee,0x78a5636f,0x84c87814,0x8cc70208,
207                  0x90befffa,0xa4506ceb,0xbef9a3f7,0xc67178f2
208          };
209
```

Figure 8: Function :ComputeHashValue

2.2 Application of SHA 256 in Blockchain

In this part any .txt file can be read and every line is considered as a block. For every block, a hash value is generated. One block is connected to the previous block through the previous block's hash value. Thus every blocks are connected to each other and make a chain. Any change in any block of the blockchain can be detected. Changing any data of block in blockchain and it's given hash value will not be the same as the previous hash value because every unique data or massage can give a unique hash value. So by comparing the hash values it can be detected whether the data in a block is changed or not.

2.3 Application of SHA 256 in Image Security Checking

Images can be slightly changed or can be concealed by other image internally but externally it can be viewed as same as real image. This is called steganography. Now a days', terrorists are very smart and they are using technologies for their harmful purpose. They can send information using steganography. For detecting steganography or any change in an image can be detected by using SHA 256 algorithm.

For this purpose, first I have read pixels of two image files which I have to detect. Every line of pixels is considered as a block and blocks are connected to their previous block by previous hash value. Then I have matched the hash values of corresponding image files. As there any change in data or pixel the hash value must be different, so if there any mismatch between the corresponding hash values, the program will give message where the mismatch is. If there is no change in data or pixels it will give a message that the images aren't different.

3. User manual

I have applied SHA-256 algorithm in two different areas. One is for file security checking and other is for image security checking.

3.1 File security checking:

- **Open executable file**
- **Then we will see an interface**

Figure 9: User Manual :Input for File security

- **Then we have to give a file name in .txt format which should stay with the executable file directory**
- **After that we have to give another file namein .txt format to store hash generated blocks**

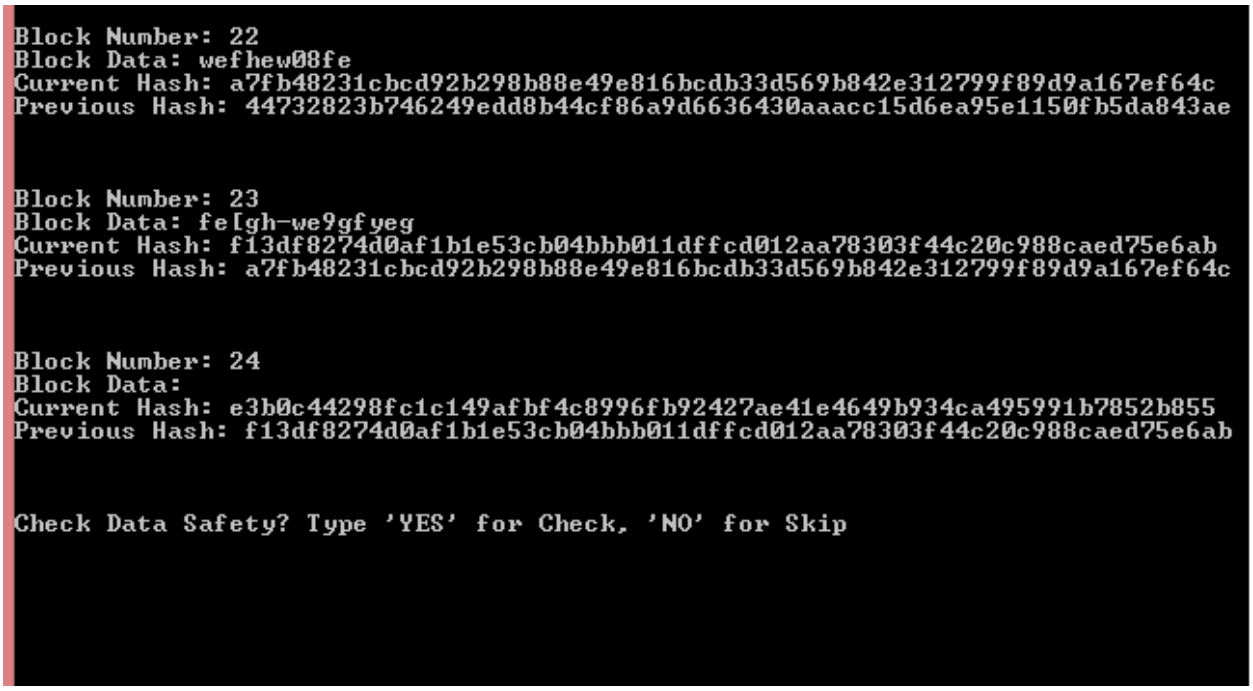

Figure 10: User Manual :Input for File security

- Then it will give output of the blockchain
- After that it will give a message if you want to check for data safety or not
- If we type "YES" it will check the block data and if it finds any change it will show message

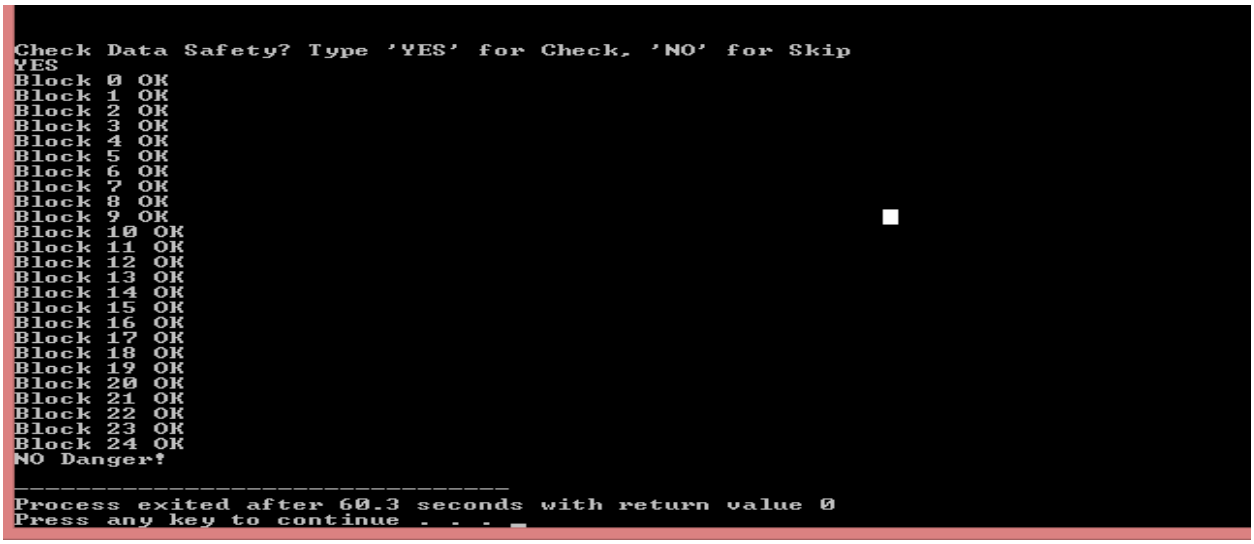

Figure 11:User Manual :Output for File security

3.2 Image security checking:

- Open the executable file
- Then we will see an interface

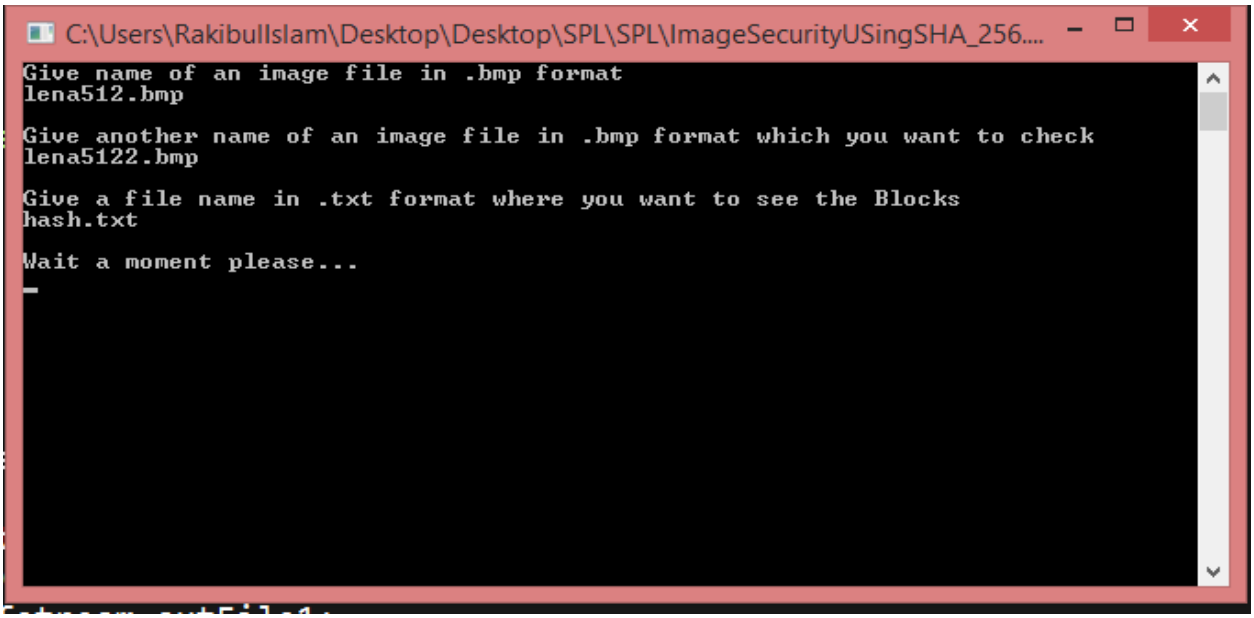

Figure 12:User Manual :Input for image security

- We have to give two image file in .bmp format which we want to check and they must be present in the executable file directory
- After that it will give a message for giving a file name where the blocks will be saved
- After giving file name we have wait for a moment as there wil be thousand of blocks for checking
- Then we will see if the hash of the two different image blocks are same or not.
- If same it will show a message “Matched”
- If different it will show a message “Pictures don’t match”

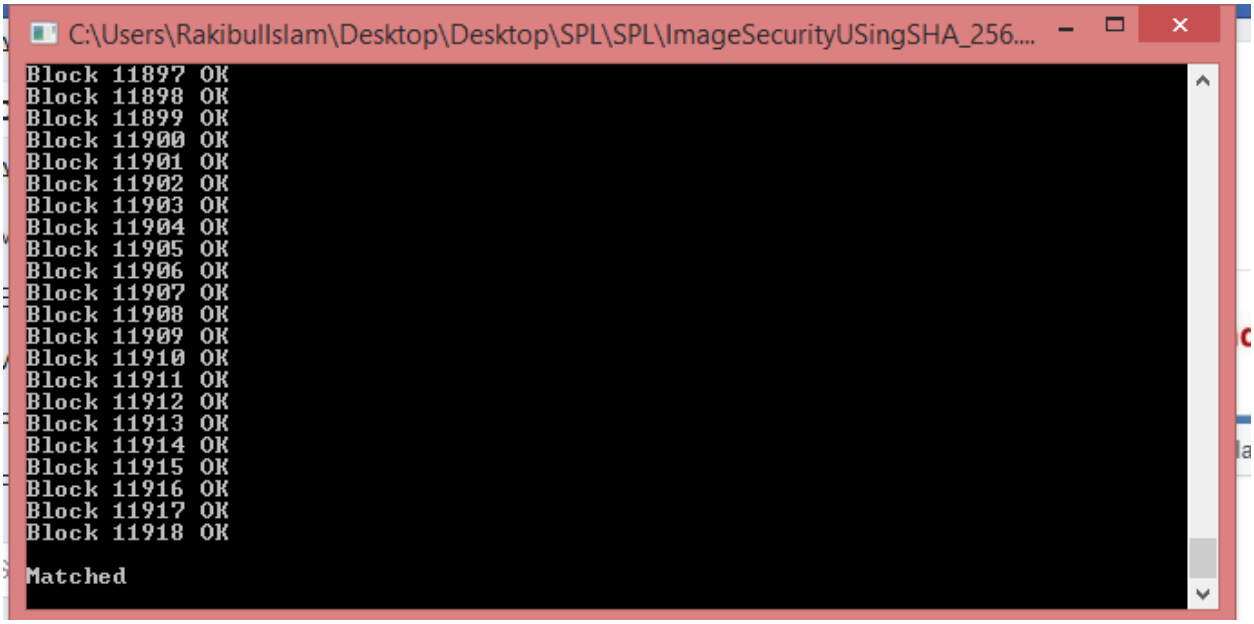

Figure 13:User Manual :Output for image security

4. Conclusion

Implementing SHA-256 algorithm helps me to improve my coding skill and I have learned to handle large code for the first time. I hope it will help me to deal with difficulties in future. This project was quiet challenging and I gained a lot of experience from it. I want to thank my supervisor for guiding me a lot during this project.

5. Appendix

In this project, I have implemented SHA 256 algorithm for .txt file and for .bmp file. In future I want to implement this algorithm for .jpeg, .jpg and .png file.

6. Reference

1. https://en.wikipedia.org/wiki/Blockchain_(database) [10/01/2018]

2.https://www.tutorialspoint.com/cryptography/cryptography_hash_functions.htm[20/05/2018]

3. https://en.wikipedia.org/wiki/Steganography [23/04/2018]

4. https://en.wikipedia.org/wiki/Bitwise_operation[29/01/2018]

5.https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Standards-and-Guidelines/documents/examples/SHA256.pdf[18/01/2018]