An Innovative Approach for Implementation of One-Time Pads

Sagar Chandrakar
Department of CS & E
FET, SSTC, SSGI,
Junwani, Bhilai (C.G.)
India

Shrikant Tiwari
Department of CS & E
FET, SSTC, SSGI,
Junwani, Bhilai (C.G.)
India

Bhagya Shree Jain
Department of CS & E
FET, SSTC, SSGI,
Junwani, Bhilai (C.G.)
India

ABSTRACT
This paper presents a different way for implementation of one-time pad. The one-time pad are known as the mother of all encryption schemes, since it is known to be information-theoretically secure in contrast to other encryption schemes used in practice, which are at most computationally secure. Our proposed way will make encryption impossible to crack if used correctly. We are implementing one-time pad encryption technique using both block cipher and one-way hash. In this proposed method each bit or character from the plaintext is encrypted by a modular addition with a bit or character from a secret random key of the same length as the plaintext, resulting in a cipher text. If the key is truly random and at least as long as the plaintext also never reused in whole or part and kept secret the cipher text will be impossible to decrypt or break without knowing the key.

Keywords
One-Time Pad, Encryption, Block Cipher, Randomness, One–Way Hash, Decryption

1. INTRODUCTION
In theory, one-time pad is the best solution for data encryption, as system using one-time pad can't be deciphered [1]. One-time pad key system has a random sequence of keys where each key is used once for each block, the receiver uses the same sequence of keys to decrypt the blocks. Third parties can decrypt all the blocks only when they get the sequence of keys. If there is no sequence of keys, it would be impossible to decipher the cipher text. The important point of one-time pad algorithm is how to generate a random sequence [1] of keys. The way to attack one-time pad algorithm is same as to attack the algorithm of generate a random sequence of keys. There are number of algorithms to generate a random sequence of keys, such as using the Microsoft's Crypto API to generate a random sequence of keys using a random 64-bit binary number and a one-way hash function to generate a random sequence of keys, using a polynomial to encrypt the plaintext. These methods all show one-time pad's high security.

We can only talk about one-time pad if four important rules [2] are followed. When rules are applied correctly, the one-time pad is proved unbreakable However, even if only one of these rules is disregarded, the cipher is no longer unbreakable.

1. The key is as long as the plaintext.

2. The key is truly random (not generated by simple computer Rnd functions or whatever!).

3. There should only be two copies of the key: one for the sender and one for the receiver (some exceptions exist for multiple receivers).

4. The keys are used only once, and both sender and receiver must destroy their key after use.

2. ONE-TIME PAD IN THEORETICAL CONTEXT
Shannon's one-time pad [3] is one of the simplest cryptosystems. If suppose the length of a message is a n bits string, $m = b1\ b2 \ldots bn \in \{0, 1\}\ n$

The length of secret key is a n bits string also,

$k = k1\ k2 \ldots kn \in U\ \{0, 1\}\ n$

Notice: the sign"$\in U$" denote homogeneous random selection.

Encrypt one bit once, each message bits and the correspond of the secret key bits carries on the XOR operation to get a cryptograph string $c = c1\ c2 \ldots cn$。 $ci = biki$, $1 \leq i \leq n$. If the secret key string use once, the one-time pad Vernam cryptogram be called one-time pad. Strictly speaking, satisfy a conditional cryptogram of the following to be real one-time pad:

1. Secret key is random produced;

2. Secret key cannot be of repeated usage

3. The length of secret key must bigger than plaintext.

3. PROPOSED METHOD FOR ONE-TIME PAD ENCRYPTION
We are proposing one-time pad encryption technique using block cipher and one-way hash.

Conventional block cipher is symmetric encryption [4] here the sender's key is same to the receiver's key. Block cipher is the algorithm that can get a cipher text block from a plaintext block and both the two blocks have the same length. Conventional block cipher algorithms including AES, DES, IDEA and other commonly used encryption algorithm.

One-way hash function's [4] purpose is to generate fingerprints for data such as documents, message etc. A hash function for message authentication has some of the following nature: it can input the data of arbitrary length; the output data is fixed length; given the original data, the encrypted data easy to calculate; for a given encrypted data to looking for the original data is computationally difficult; for different original data to get the same encrypted data is computationally difficult [5, 6]. MD5, SHA-1 is excellent one-way hash algorithm. A message of arbitrary length can be MD5 algorithm's input.

3.1. Description of the algorithm
In this paper, we choose SHA-1 from one-way hash function and choose AES from conventional block cipher to describe

the algorithm [4]. The plaintext block is 64 bit in length, and the first block is A1, the second block is A2and so on; Each plaintext block has a key, and the first block's key is K1, the second block's key is K2... and so on; We can get one cipher text block from one plaintext block , and the first cipher text block is B1, the second cipher text block is B2and so on;[4] The key that the user input we call it user key K. Figure 1 shows the encryption process and Figure 2 shows the decryption process [4]. The reason for combination of block is the fundamental requirement of one-time pad to have length of message and key same.

3.1.1. Proposed Encryption Process:

1) First transform the user key K into B0 through the transformation function X (K). BO's length is same to the AES block length; both of them are 64 bit. X (K) contains some simple XOR and shift operations.

2) Then using one-way hash function SHA (B0, K) to get the first block's encryption key K1. The function SHA (B0, K) connects the B0 and K, and to do a XOR and shift operation, finally, calculates its SHA value. This SHA value is K1.

3) And call the function AES_EN (K1, A1) to encrypt the first block. The function AES_EN (K1, A1) use K1 as AES algorithm's key, and use A1 as its input, then get the cipher text block B1.

4) Calling SHA (B1, K) to get the second block's encryption key K2.

5) Calling AES_EN (K2, A2) to encrypt the second block and get B2.

6) Repeat 4) and 5) until you have encrypted all the blocks.

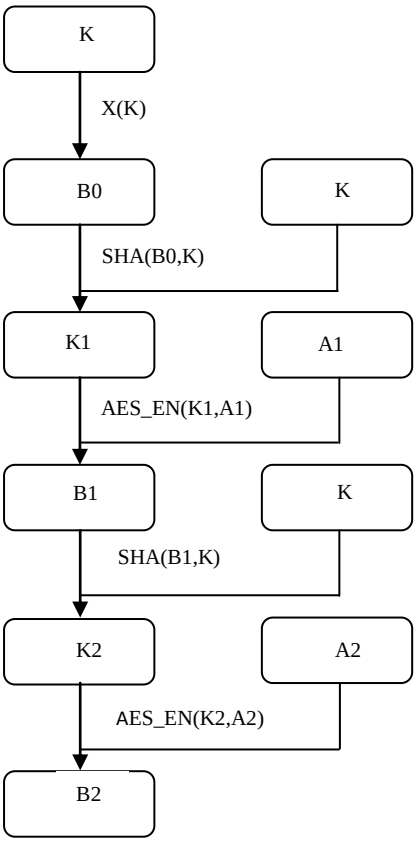

Figure 1: Encryption Process [4]

3.1.2. Proposed Decryption Process:

1) First transform the user key K into B0 through the transformation function X (K). BO's length is same to the AES block length; both of them are 64 bit. X (K) contains some simple XOR and shift operations.

2) Then using one-way hash function SHA (B0, K) to get the first block's encryption key K1. The function SHA (B0, K) connects the B0 and K, and to do a XOR and shift operation, finally, calculates its SHA value. This SHA value is K1.

3) And call the function AES_DE (K1, B1) to decrypt the first block. The function AES_DE (K1, B1) use K1 as AES algorithm's key, and use B1 as input, then get the plaintext block A1.

4) Calling SHA (B1, K) to get the second block's decryption key K2.

5) Calling AES_DE (K2, B2) to decrypt the second block and get A2.

6) Repeat 4) and 5) until you have decrypted all the blocks.

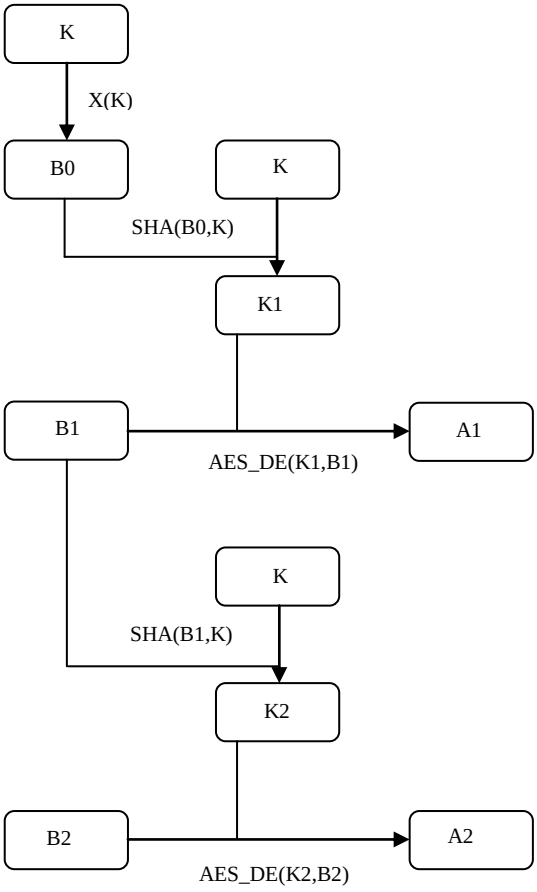

Figure 2: Decryption Process [4]

3.2. Analyse algorithm's security

This algorithm is an implementation of one-time pad, which calculate an encryption key for each block, while the sequence of keys to be determined entirely by the user encryption key and the plaintext. Before encrypt data, no one can predict or get the sequence of keys, because it is dynamically generated in the encryption process [7]. Through SHA transform we can get the first block's key from user key, and then all the other keys come from a SHA transform which based on the previous block and the user key. So this sequence of keys can't be

predicted. Thus making it secure from the perspective of one-time pad.

4. OBSERVATION

When we see the theoretical aspect as well as the Practical implementation in one-time pad we observe one important characteristic that is length of Key is equal to length of Message (as shown in Figure 3) so in other words as the length of message increases the complexity and the probability of use of brute force method to find out the encrypted messages reduces, but if the key length is small then there can be probability of finding the sequence of random key generated.

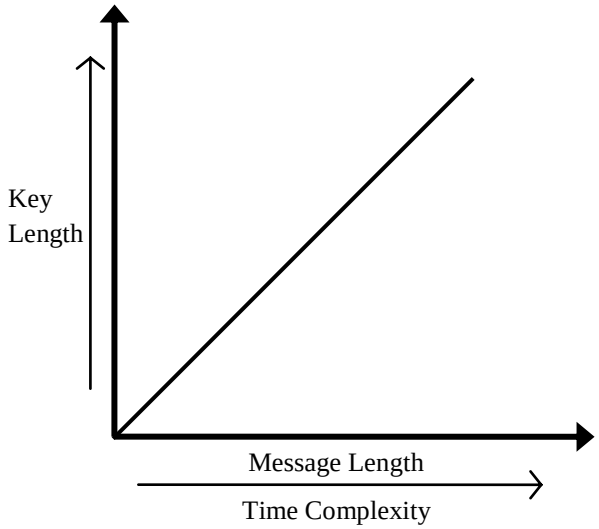

Figure 3: Basis Characteristics of One-Time pad

When we compare one-time pad with other major encryption algorithm as shown in Figure 4 we observe that practically for low volume of data the encryption rate of all encryption standard is more or less same, but as the volume of data increase's the time taken for encryption increase as complexity and data iteration overhead also increase's.

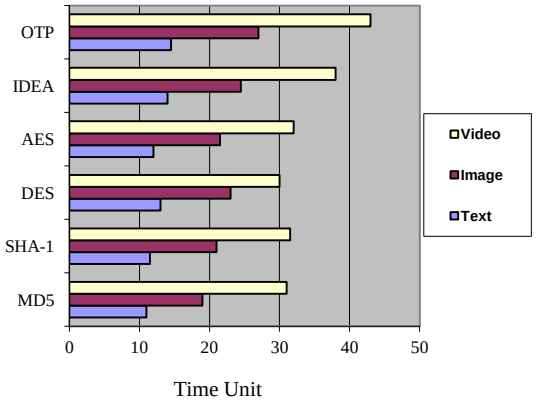

Figure 4: Speed Comparison of different encryption algorithm

In one-time pad when the encryption operation is simple that is may be only XOR operation is used the encryption rate is high but when it uses our proposed way then the encryption rate becomes less and thus for large volume of data the one-time pad is slower but much more secure as it would have a very large random key. The summary of our finding about the time requirement of One-Time pad is tabulated as shown in Table 1.

Table 1: Time requirement of One-Time pad compared with other major encryption algorithm.

Algorithm	Time taken for Encryption		
	Text	Image	Video
MD5	Low	Low	Low-Medium
SHA-1	Low	Low	Low-Medium
DES	Low	Medium	Medium-High
AES	Low	Medium	Medium-High
IDEA	Low	Medium	Medium-High
One-Time Pad	Low	Medium	High

5. CONCLUSION

This algorithm protects user key using one-way hash function since it is not reversible. We can calculate the encryption key from the last cipher text block used in this implementation. Using this simple method we will get the unpredictable and random sequence of keys. The algorithm used is based on conventional block cipher algorithms. This implementation allows the security of conventional block cipher to increase exponentially.

6. REFERENCES

[1] Christian Matt and Ueli Maurer, "The One-Time Pad Revisited" in proceeding of Information Theory Proceedings Conference, pages 2706-2710. July 2013.

[2] Mariusz Borowski and Marek Lesniewicz, "Modern usage of "old" one-time pad" in proceeding of Communications and Information Systems Conference, pages 1-5. October 2012.

[3] Jiao-Hongqiang, Tian-Junfeng and Wang-Baomin, "A Study on the One-Time Pad Scheme Based Stern-Brocot Tree" in proceeding of ISCSCT 08 Conference, pages 568-571. December 2008.

[4] Songsheng Tang and Fuqiang Liu, "A one-time pad encryption algorithm based on one-way hash and conventional block cipher" in proceeding of CECNet Conference, pages 72-74. April 2012.

[5] Jeyamala.C, GopiGanesh.S, Raman G.S, "An Image Encryption Scheme Based on One Time Pads – A Chaotic Approach" in proceeding of ICCCNT Conference, pages 1-6. July 2010.

[6] J. C. Bienfang, A. Mink, B. J. Hershman, A. Nakassis, X. Tang, R. F. Boisvert, D. H. Su,Charles W. Clark and Carl J. Williams, "Broadband Quantum Generated One-Time-Pad Encryption" in proceeding of Quantum Electronics and Laser Science Conference, pages 362-364. May 2005.

[7] Alan Mink, Lijun Ma, Tassos Nakassis, Hai Xu, Oliver Slattery, Barry Hershman and Xiao Tang, "A Quantum Network Manager That Supports A One-Time Pad Stream" in proceeding of Quantum, Nano and Micro Technologies Conference, pages 16-21. February 2008.