International Journal of Engineering and Technical Research (IJETR)
ISSN: 2321-0869, Volume-2, Issue-2, February 2014

Implementation of One-Time Pad over Web

Sagar Chandrakar, Bhagya Shree Jain, Shrikant Tiwari

Abstract— **This paper aims to present the implementation of One-Time Pad (OTP) which is a type of encryption that is impossible to crack if used properly and correctly. In this paper we are proposing one-time pad implementation over web using cookie encryption and cookie decryption. The basic encryption method of one-time pad is used, where characterfrom the plaintext is XOR with the character from a secret random key (or *pad*) of the same length as the plaintext, resulting in a cipher text. If the key is truly random then the cipher text would be impossible to decrypt or break without knowing the key. In our proposed method an index string is generated with is stored in remote servers. This information is used to generate cookie which can be used for secure communication over web using One-Time pad.**

Index Terms— **One-Time Pad, Encryption, Decryption, Randomness**

I. Introduction

One-Time Pad was first proposed in 1926 by G. Vernam to encrypt wire and radio communications. In 1949 C.Shannon [1] proved the perfect secrecy of one-time pad. The essence of one-time pad is to generate a random string of at least the length of the message to be the key of this message, and then perform any operation like XOR the key with the message to produce the cipher text. In order to decrypt the cipher text we have to reverse the operation XOR the cipher text with the key again in this example. It is impossible to crypt-analyse one-time pad, since every message has a different random key, and there is no way to distinguish the cipher text from a random string. Although one-time pad is simple, fast and very secure, it is not widely used. There are two major obstacles that prevent one-time pad from being used in generic applications [2].

The first obstacle being the generation of random numbers.Although we recognize that this is a big obstacle for using one-time pad encryption we do not address any solution for this particular problem in this paper. We believe that if the second obstacle, the key distribution problem, can be solved then pseudo random number generator [2]

with periodic reseeding can be used. Although the resulting algorithm would not be exactly one-time pad init's some sense, but it would provide a reasonable proximity to the one-time pad.

Manuscript received Feb. 20, 2014.

Sagar Chandrakar, Department of Computer Science and Engineering FET, SSTC, SSGI, Junwani, Bhilai (C.G.)
Bhagya Shree Jain, Department of Computer Science and Engineering FET, SSTC, SSGI, Junwani, Bhilai (C.G.)
Shrikant Tiwari, Department of Computer Science and Engineering FET, SSTC, SSGI, Junwani, Bhilai (C.G.)

In the usual scenario of secure data communication, encryption of a message is done by one party, and decryption

of the same message is done by another party, with both parties sharing the same key for the same message. However the one-time pad encryption requires a new key for every new message, which in turn requires a mechanism to distribute keys so that both parties can have the same key for a message while no third party could intercept morph or tamper with these keys. If such a mechanism exists for generic applications, then logically the encrypting party could have used this mechanism to send the message directly without encryption. In reality, no such practical mechanism exists, so this fear of leaking the keys out is keeping people from using one-time pad extensively. However, we should notice that in applications where encryption and decryption are performed by the same party, the key distribution problem [3] is not an issue anymore. This means that we can employ one-time pad on such type of encryptions tasks, and achieve the perfect secrecy.

II. Proposed Web Implementation of One-Time Pad

One Implementation that can be used in web is cookie encryption [4] where encryption and decryption of the cookies are done by the same party that is in server. The client simply stores and sends the cookie, but does not participate in cookie encryption or decryption also; the client does not use the information in the cookie. Therefore the keys are solely used by the server, instead of being shared by two parties. This characteristic shows up only when the party that performs both encryption and decryption is not able to or does not want to store the sensitive information after its use. Otherwise, if the encrypting or decrypting party wants to store the information, then this party could have simply generated a string to send to the other party, while storing the real information with the string in the local database, since the other party does not need the real information anyway. Another approach would be to use combination of different encryption algorithm [5]

in one-time pad.Figure1 shows the flow charts for cookie encryption and Figure2for decryption. When a web server needs to encrypt a cookie before sending it out, the web server follows the first chart. When the web server receives a cookie from a client, the web server follows the other chart to decrypt the cookie.

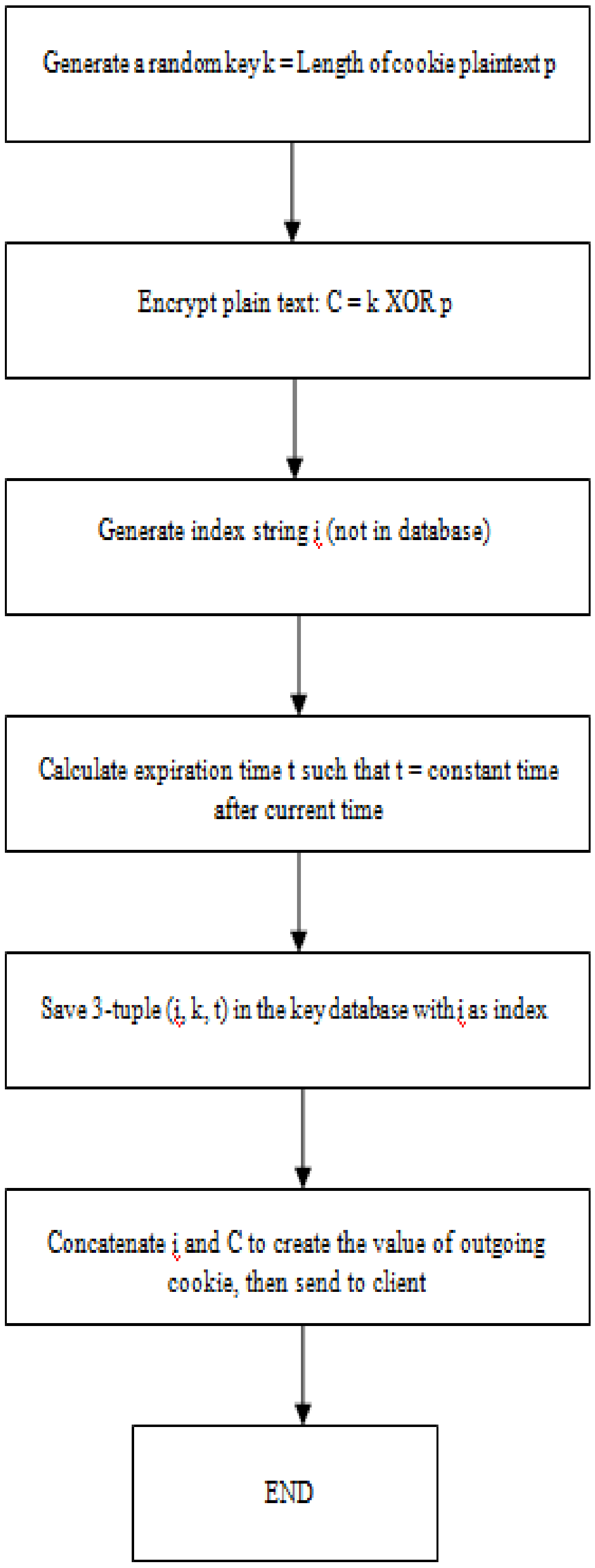
Generate a random key k = Length of cookie plaintext p
Encrypt plain text: C = k XOR p
Generate index string i (not in database)
Calculate expiration time t such that t = constant time after current time
Save 3-tuple (i, k, t) in the key database with i as index
Concatenate i and C to create the value of outgoing cookie, then send to client
END

Figure 1: Cookie Encryption [4]

What makes cookie encryption so special is the web sites' motive to respect users' privacy and reduce the web sites' own liability; this is the reason why the web sites do not want to store customers' credit card information after its use. Based on this observation, we can use one-time pad cookie encryption protocol to protect online users' privacy. In short, the web servers store the one-time pad keys in a local database and encrypt or decrypt the cookies using these keys.

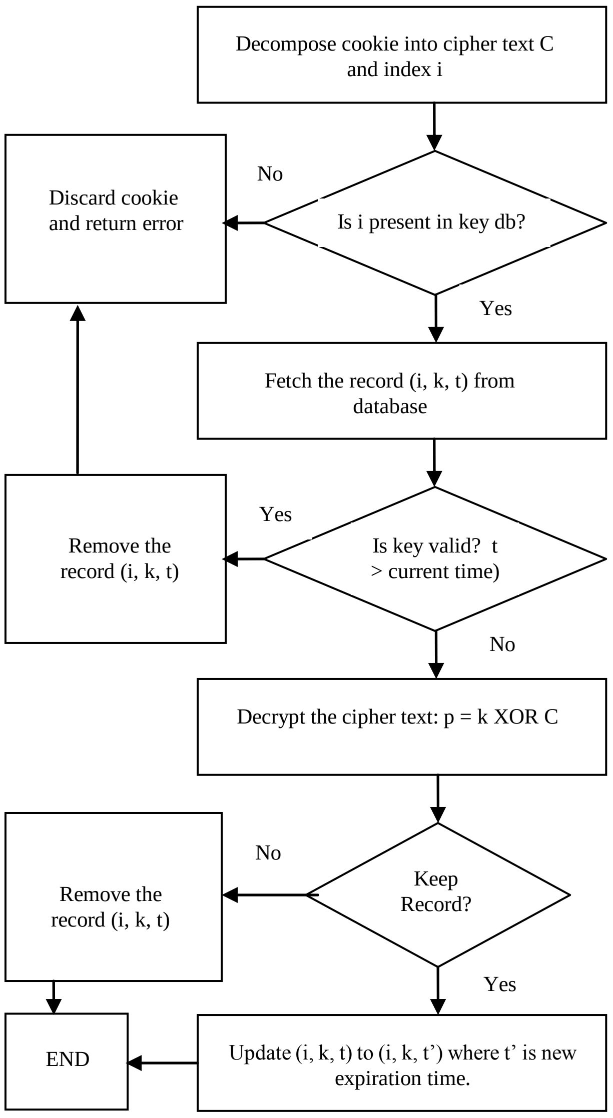
Decompose cookie into cipher text C and index i
Is i present in key db?
No
Discard cookie and return error
Yes
Fetch the record (i, k, t) from database
Is key valid? t > current time)
Yes
Remove the record (i, k, t)
No
Decrypt the cipher text: p = k XOR C
Keep Record?
No
Remove the record (i, k, t)
Yes
Update (i, k, t) to (i, k, t') where t' is new expiration time.
END

Figure 2: Cookie Decryption [4]

Notice that after the decryption, whether or not to remove the key from the key database would depend on the web designer's choice. The web server would check's the expiration times in the key database against the current time, and deletes all the expired keys. The main security related reason is to provide a mechanism to limit the time the server stores the key, hence limiting the web server's liability in case an attacker breaks into both the database and a user's computer. Another reason is that sometimes users may leave a web site and not come back anymore, or chooses to clean her cookies even if she comes back next time, or the users' system might crash, in which cases the cookies would not return to the web server anymore. So server should delete them.
The main factor that could impact one-time pad performance is not the encryption or decryption themselves, but the generation of the random keys [6]. The process of generating random numbers is usually far slower than the one-time pad encryption or decryption itself. Nevertheless, in case this becomes a major issue of the overall real-time performance, web servers can choose to use pre-computed random number table to save the time of random number generation.
One seeming drawback of one-time pad is that the keys should be as long as the plain texts and have to occupy the server's storage space that is as large as the plain texts. In addition,

since the keys are supposed to be random strings with very high entropy, compression will not work on the keys. However, the cookies that this scheme is encrypting are not that much large.

The perfect secrecy of any one-time pad relies on the randomness of key generation [7, 8]. If the keys are not random, then the algorithm cannot be proven to provide perfect secrecy. True random numbers can be obtained by measuring anyrandom physical process, such as cosmic radiation, or thermal noise [9]. Therefore we can be reasonably confident that one-time pad encryption on cookies is unbreakable. The user's privacy is hence strongly protected [9], and the web site's liability is substantially reduced by the one-time pad encryption.

III. Observation

When we observe the theoretical aspect as well as the Practical implementation in one-time pad as shown in Figure 3 we come across a very important aspect that the length of Key should be equal to length of Message. This implies that as the length of message increases proportionally the key length also increase's thus the complexity and the probability of brute force method to find out the encrypted messages reduces.

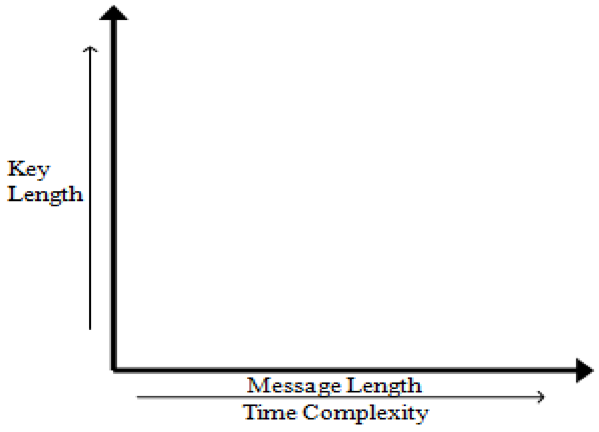

Figure 3: Basis Characteristics of One-Time pad

The performance comparison as shown in Figure 4 of one-time pad gives us very useful insight of encryption rate over different formats of different encryption algorithms which are used worldwide over web.

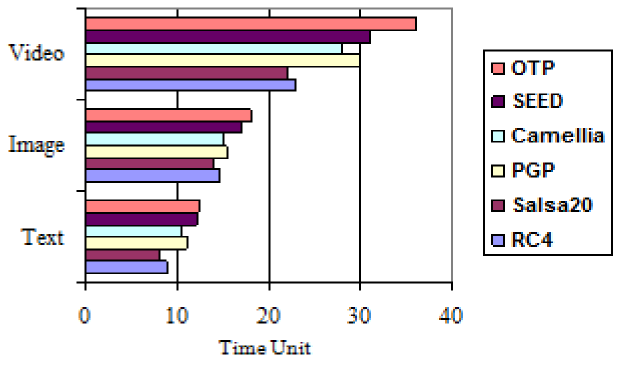

Figure 4: Performance Comparison of various encryption algorithms over web

When we analysis the speed or rather the time complexity of one-time pad with major other encryption algorithm as shown in Table: 1 we observe that speed or encryption rate is practically is more or less same for text data or rather for low volume, but as the volume of data increase's the time taken for encryption increase due to increased complexity and higher data iteration overhead. In one-time pad when the data size is small then we have faster encryption rate as the length of key is also small, but as the data size increase's the key length also increase's in same proportion so there is decrease in the data encryption rate.

Algorithm	Time taken for Encryption		
	Text	Image	Video
RC4	Low	Low	Low-Medium
Salsa20	Low	Low	Low-Medium
PGP	Low	Medium	Medium-High
Camellia	Low	Medium	Medium-High
SEED	Low	Medium	Medium-High
One-Time Pad	Low	Medium	High

Table 1: Time Complexity of major encryption algorithm used over web.

IV. Conclusion

This paper gives a really useful, real-world application of secure distributed storage of sensitive information using our implementation. A One-Time Pad method to achieve perfect secrecy of cookie encryption is proposed, based on the observation of the interesting characteristic of cookie encryption. The pros and cons of this approach are analysed, and the comparison with the existing approaches shows that our approach is able to protects users' privacy and reduce web sites' liability in a much stronger manner.

References

[1] 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.
[2] Fengling Han, Jiankun Hu and Kai Xi, “Highly Efficient One-time Pad Key Generation for Large Volume Medical Data Protection” in proceeding of ICIEA Conference, pages 330-335. June 2010.
[3] 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.
[4] Donghua Xu, Chenghuai Lu and Andre Dos Santos, “Protecting Web Usage of Credit Cards Using One-Time Pad Cookie Encryption” in proceeding of Computer Security Applications Conference, pages 51-58. April 2002.
[5] 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.
[6] 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.
[7] Samah M. H. Alwahbani and Eihab B. M. Bashier “Speech Scrambling Based on Chaotic Maps and One Time Pad” in proceeding of ICCEEE Conference, pages 128-133. August 2013.
[8] 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.
[9]Zhihua Chen and Jin Xu, “One-Time-Pads Encryption in the Tile Assembly Model” in proceeding of BICTA Conference, pages 23-30. October 2008.