Stream Ciphers: A Comparative Study of Attacks and Structures

Daniyal M. Alghazzawi Syed Hamid Hasan Mohamed Salim Trigui

Information Security Research Group
Faculty of Computing and Information Technology
Department of Information Systems
King Abdulaziz University
Kingdom of Saudi Arabia

ABSTRACT
In the latest past, research work has been done in the region of steam ciphers and as a answer of which, several design models for stream ciphers were projected. In Order to appreciate a international standard for data encryption that would prove good in the due course of time, endure the action of cryptanalysis algorithms and reinforce the security that will take longer to be broken, this research work was taken up. There is no standard model for Stream cipher and their design, in comparison to block ciphers, is based on a number of structures. We would not try to evaluate the different designs taken up by various stream ciphers, different attacks agreed out on these stream ciphers and the result of these attacks. During this exercise we would also have a better idea of the latest stream ciphers designs.

General Terms
Cryptography, Security

Keywords
Attacks, stream cipher, AES, LFSR

1. INTRODUCTION
Does improved security offer comfort to apprehensive people? Or does security offer some very fundamental protections that we are juvenile to believe that we do not require? Nowadays whilst millions of populace relies on Internet for crucial communication, for business between them, a secure system is a very important aspect to deal with. Under these circumstances Cryptography becomes a very essential aspect for secure communications. Cryptography deals with four major goals viz Confidentiality, Data integrity, Authentication and Non-repudiation and thus is widely used to secure telephonic messages, e-mails, credit card information, and corporate data[1] but even with all these applications, it must be kept in mind that alone cryptography is not enough for the requirements of security. Cryptography systems can in general classified into symmetric-key systems (AES,RC4,DES) that uses a single key available both with the sender and the recipient, and public-key or asymmetric systems (ElGamal, McEliece, RSA) which utilizes the two key system, where the public key is published and the private key provided only to the person the message is intended for [3]. Block cipher and stream cipher are combined to make the Symmetric cryptosystems. According to Rueppel.

Fixed transformation is applied on large blocks of plain-text data for the operation of Block cipher while for Stream cipher the transformation utilized are the time-varying transformations.
However, this is not an unconditional classification, block cipher is interchangeably uses as stream cipher for specific operation modes. S_0 & K_c are the initial S-bit and KC Bit cypher key, respectively that the stream cipher is laden with. The key stream Z_t is generated by the output function 'O' of the stream cipher at every time interval t. A plain-text digit P_t is encrypted by this key stream digit through E, an encryption function, which results in the cipher-text C_t. Then U, the states update function updates the state S_t. the three equations that state the rules of the encryption process are as follows:

$$Z_t = O(S_t, K_c)$$
$$C_t = E(P_t, Z_t)$$
$$S_{t+1} = U(P, S_t, K_c),$$

Where the decryption function *D,* can easily be constructed for the *E (*encryption function), We can describe the process of decryption as bellow:

$$Z_t = O(S_t, K_c)$$
$$P_t = D(C_t, Z_t)$$
$$S_{t+1} = U(P_t, S_t, Kc).$$

Stream Ciphers when compared to the block ciphers, offer a number of advantages[4] as:

- Stream cipher is faster than the block ciphers
  - The propagation of error is minimal
  - Complexity of Hardware is very low
  - We can generate the keystream even before encryption or decryption.

Further on, we can classify Stream Ciphers as self-synchronizing or synchronous based on their internal state. A cipher is categorized as a synchronous stream cipher, if the change occurring in the state is independent from plain-text/cipher-text message. On the other hand In contrast, if the update of the state is bases on the previous cipher text digits then we term as self-synchronizing. The key and its position *i* are the only dependencies for the generation of the keystream in synchronous ciphers, while the key & a specific amount of previous cipher-text, are the factors deciding the generation of the keystream in case of self-synchronous cipher. Synchronous ciphers are described as having no error transmission while error transmission is incomplete in self-synchronous With synchronous ciphers, synchronization is achieved with 'marker positions' in the transmission, however, decryption can be resumed if the keystream falls out of synchronization, in case of Self-Synchronous ciphers.
Though desirable properties are found in both the variations, various implications are found in both of these. During decryption, the synchronous cipher limits the opportunity of detecting an error and a more serious limitation that since the attacker is aware of the exact effect any changes in the chipper-text will have on the plain-text, he can easily make

controlled change to the cipher-text. Rueppel [5] suggests two implications with self-synchronizing ciphers. One is that, an enemy is conscious of the information related to the variables that are used in the format of input to the initiator, as it is in use from the ciphertext. Another is that since the keystream is dependent on the message these generators can be analyzed in a limited fashion only.

2. DIFFERENT DESIGN APPROACHES TO STREAM CIPHERS

Since there are no standard models, a number of structures can be found in literature. Some of these structures are:

2.1 Linear Feedback Shift Register based stream ciphers

Having the ability of being cost-effectively implemented in the Hardware, a commonly used part of stream cipher is LFSR (Linear feedback shift register), which also have well understood properties and possess simple structure that is easy to analyze mathematically. The basic motivation behind the frequent usage of LFSRs in stream cipher design is their simple structure. However, LFSRs do not guarantee adept security. Thus the security of LFSRs can be increased by using various general schemes like filtering function, clock controlling and non-linear combining function.

2.2 Non-linear combining functions

The current stream ciphers have an integral part in the form of feedback registers with non-linear functions. LFSRs are inherently linear. We can feed the output of multiple LFSRs parallely into a Non Linear Boolean Function so that is generates a combination, thus removing the linearity. Security of resulting scheme is dependent on the properties of the combining function and it required to avoid the correlating attack.

2.3 Clock-controlled generators

Clock controlled generators serve the function of introducing non-linearity in LFSRs. This non-linearity is obtained by having LFSR clocked unevenly by being motivated by the output of some other LFSR. Several generators like stop and go, Shrinking Generator and Alternating-Step-Generator are based on the above principle.

2.4 Filter generator

Filtering Boolean functions is yet another approach of removing the linearity of LFSR and thus improving its security. Although, not sufficient to be resistant enough against several attacks, certain characteristic like high algebraic degree, balance, high non linearity and algebraic immunity are termed essential in stream ciphers with this structure.

2.5 T-Function Usage

An invertible mapping with singular cycle on n bit was proposed by Klimov and Shamir in 2002 this mapping was known as T Function (Ttriangular-functions). The bit size of these T functions make them less useful but the expanded multi word version of the T function is being put to use in numerous stream ciphers that are treated as alternative of Linear feedback shift registers.

3. ANALYSIS OF SOME STREAM CIPHERS

In 1987, the most commonly used stream cipher, known as RC4 was designed by Ron Rivest.

3.1 RC4

It was originally designed with the intent of providing security to RSA, thus it was kept as a secret but in 1994 the source code was leaked to the email list of cyber Punks by someone anonymous. The Secure Socket Layer/ Transport Layer Security uses RC4 for communications amongst the browser and server. It is also utilized in the Wired Equivalent (WEP) protocol used for wireless communication in 802.11 WLAN

3.1.1 Implementation - Since it is required to manipulate bytes only for RC4, it is most suited for implementing in software. The inner state of the cypher stream contains a dynamic table S and variables with 2 byte lengths. 256 bytes are used for keeping the Array of states, from S[0] to S[255], The key is stored in k byte, from key[0] to key [k-1]. S is permuted by a variable length key K, and contain the value 0 to 255 in increasing sequence. Based on the state the output is produced, some element of S being permuted in all steps. Linear cryptanalysis is ruled out because of the enormous size of the log (256!)≈1684 bit.

3.1.2 Security - In 2001, Shamir & Mantin discovered a minor and distinctive attack, even though the cipher is studied and analyzed so much[8]. Information related to the ley is leaked by the initial few-hundred randomly outputed bytes, particularly the initial byte which are greatly prejudiced and the value of 0 with the probability of 2-7 in place of 2-8, is taken by the 2^{nd} byte of RC4. The cause of this flaw is the non-uniform distribution of Table S, post the preliminary rearrangement. It was demonstrated by Mossel [9] that a table of size n can be permuted in a uniform fashion by rearrangements through semi random transposition that has a computation complexity Θ (n log n).

3.2 Polar Bear

Kohan H Astad and Mats N Aslund designed Polar Bear with the claim that it was appropriate for hardware and software both. From the thirty five illustrations which was submitted to eSTREAM, one is Plar Bear.

3.2.1 Implementation - Polar Bear uses one Seven-worded LFSR R^0 and one Nine-worded LFSR R^1. Apart from the above mentioned register, the word-Quantity S and Dynamical arrangements of Byte D determine the internal state of the cipher. It is mainly intended for a 128 bit key and with an upper limit of 31 the IV can be of any length. The key is taken, IV is interpreted as plain text block and a slightly modified five round Rijndael encryption is applied with a block length of 256 for initializing the cipher for each message that needs processing .R^1 & R^0 are then utilized to load the resultant cipher text-block. Lastly, table T_8 is equated to D_8 by initializing the permutation, S is set to value '0' along with the Rijndael S-box. At a time a 4-byte result is generated. On the other, S determines the 2 LFSRs after being clocked irregularly. The 4 output Byte are generated as a result of 8 byte chosen from R^0 and R^1 being run through the permutation D_8. Finally to prepare for the next output cycle R^0 and S are

modified after swapping specific D_8 entries. Except the LFSR stepping all entry in R^1 remain un-modified.

In his research paper John Mattson [10] has shown certain weakness of the stream cipher polar bear. He has written that attacks are possible. It is evident that the key size of polar bear is not net by its attack resistance. E.g. If one could guess the value even of the smaller LFSR, it is easy for him to infer all the other values just by analyzing the output.
To solve this issue we could increase the length of one of both the LFSR, however to effectively counter such attacks the length need to be nearly doubled.

3.3 Sober t-32

Sober t-32 is an asynchronous stream cipher design for a secret key that is of 256 bits. The aim of design on the SOBER ciphers was to be used for mobile telecommunication. However, it would be devastating if there is loss of synchronization with the cipher stream while being used in mobile communication. To prevent such a loss the information is sent in small packets named Frames.

3.3.1 Implementation - Sober t-32 has 4 four constituents:

a. Stuttering,
b. LFSR,
c. Key loading,
d. and NLF (nonlinear filter) and

The seventeen word of the LFSR are set to the primary state, by the key loading based on the key. Sometimes during the keyloading the frame key or re-synchronization key gets utilized. A stream of words called LFSR stream is constructed by the LFSR utilizing a linear feedback function and this process by which a new word is produced is termed as LFSR cycle. The nonlinear filter hides the linearity in the stream. The non linear function is utilized for combining the words of the register by the NFL, after each LFSR cycle, the output forms the NLF stream. The NFL stream words to be used in the key are occasionally selected through stuttering from amongst the NLF stream words.
Steve Babbage & team[11], in the paper titled "Cryptanalysis of sober - t 32", have shown that few new attack on the stream cipher "sober – t 32", they include guess and determine attacks upon the un-stuttered "sober – t 32" and distinguishing attacks upon full "sober – t 32".

3.3.2 Security - Guess and determine attacks upon the un-stuttered "sober – t 32" .The guessable properties of stream cipher (t-class) which are present in its s - box construction is the cause of the attacks. This relation amongst the In & Out 8 bits is un-dispersed to other position in the word. The attack could have been foiled just by a Cyclic-shift towards the last of S-Box. Additionally, the entire word should be implicated in the "sober – t 32", instead of a part of the word by the NFL in the Word Based LFSR to prevent such attack. In such a case the guessing of certain bits of the words would prove to be useless for the attacker. In Stuttering, it is actually the fact that subsequent word don't appear in the stream that prevent the attack, rather than the uncertainty introduced by stuttering. Actually when stuttering is attacked by a timing attack [12] large sequences of subsequent word are revealed by Stuttering, this enables the guess and determine attack. Two more of the growing & unique attack on full "Sober – t 32" are found at [13]. One of them is a variation of the attacks on un-stuttered "Sober – t 32", and it works on full version of the "Sober – t 32" as well. It can distinguish the "Sober – t 32" key stream originating from an undeviating source with approximately 2^{200} output word.

4. CONCLUSION

Undoubtedly, the importance of stream ciphers in computer applications cannot be ignored. Therefore, a standardized model for the stream cipher design is certainly today's requisite. Though some new ideas are being practiced but the classical structures like LFSR, clock controlling etc can let us make a good start in this field. This review studies the standard structures and a few important stream ciphers with a hope to come up with a superior stream cipher that meets the standards of effectiveness in terms of security, implementation, and speed and error propagation in future.

5. REFERENCES

[1]. Robshaw, M.J.B., 1995. "Stream Ciphers" RSA Laboratories Technical Report TR-701Version 2.0

[2]. Rueppel, R. A. 1992. "Stream ciphers" The science of information integrity, IEEE press, New York.

[3]. Mattsson, J. 2006. "Stream Cipher Design" Master of Thesis Stockholm, Sweden.

[4]. Schafheutle, M. and Pyka S. 2003. Stream Ciphers. Technical report, NESSIE consortium.

[5]. Rueppel, R.A. 1984. New Approaches to Stream Ciphers. PhD thesis, Swiss Federal Institute of Technology, Zurich.

[6]. Kessler, G.C. 2009. "An Overview of Cryptography".

[7]. Afzal, M., Kausar, F., and Masood, A. 2006. "Comparative Analysis of the Structures of eSTREAM Submitted Stream Ciphers", IEEE-ICET 2006

[8]. Mantin, I. and Shamir, A. A Practical Attack on Broadcast RC4.

[9]. Mossel, E., Peres,Y. and Sinclair, A. 2004 "Shuffling by semi random transportations".

[10]. Bear, P. 2006. Stream Cipher Design-An evaluation of the eSTREAM candidate Polar Bear, John Mattson, PhD thesis, Sweden.

[11]. Babbage, S., Canniuere, C.D., Lano, J., Preneel, B., and Vandewalle. J. Cryptanalysis of SOBER-t32. In T.Johansson, editor, Fast Software Encryption, FSE 2003, number 2887 in Lecture Notes in Computer Science, pages 111{128. Springer-Verlag, 2003.

[12]. Ekdahl, P. and Johansson, T. Distinguishing attacks on SOBER-t16 and t32. In J.Daemen and V.Rijmen, editors, Fast Software Encryption, FSE 2002, number 2365 in Lecture notes in Computer Science, pages 210{224. Springer-Verlag, 2002.

[13]. Babbage, S., Canniuere, C.D., Lano, J., Preneel, B., and Vandewalle, J. Distinguishing attacks on SOBER-t32.

[14]. Wikipedia, the free Encyclopedia.