An Evaluation and Enhancement of Seredynski-Bouvry CA-based Encryption Scheme

Hossein Arabnezhad∗1 and Babak Sadeghiyan†2

1,2Department of Computer Engineering, Amirkabir University of Technology

Abstract

In this paper, we study a block cipher based on cellular automata, proposed by Seredynski and Bouvry in [1] against plain-text avalanche criteria and randomness tests. Our experiments show that Seredynski-Bouvry encryption scheme does not pass some NIST statistical tests by neighborhood radius less than three. It also showed that if the CA rule is selected carelessly, it weaken the security of scheme. Therefore, the selection of CA-rule as part of key cannot be left to the user. Hence, cryptographic properties such as balancedness and nonlinearity should be considered in the selection of CA-rules. This approach is more compliant with Kerckhoffs principle. So security should depend just on security of final data. We also improve Seredynski-Bouvry encryption scheme to satisfy strict avalanche criteria and NIST statistical test suite in about half number of iterations comparing to original scheme. This improvement is achieved by change in the definition of neighborhood.

1 Introduction

Nowadays private key cryptosystems have an important role in security of commercial communications. In this type of cryptosystems, sender and receiver share a secret key privately. Advanced Encryption Standard (AES) is
an example of private key block ciphers standardized for confidentiality purposes. Block ciphers are a subclass of private key cryptosystems, in which data is divided into blocks and each block encrypted individually. Design of these cipher is based on two key properties: confusion and diffusion. One of main features of cellular automata is to emerge complex behavior by application of simple rules, frequently, on array of cells. This feature has conformance to block cipher design objectives. In design of block ciphers we try to build a complex relation between plain-text and cipher-text by application of a simpler round function, iteratively. So, cellular automata can be a natural solution to design of block cipher.

Many CA-based cryptography schemes proposed in the literature, including public key, stream ciphers [2] and block ciphers [1, 3]. There are also schemes that build up a cipher based on CA-based components such as s-box or linear transformation [4].

One of these scheme is proposed by seredynski and bouvry in [1]. Their proposed scheme is based on reversible cellular automata. They studied their scheme only against strict avalanche criteria. In this article we verify the results presented in [1]. A secure block cipher also can be used as a pseudo random number generator. We study this scheme against NIST statistical test suite for testing randomness of generated bitstreams. We improve scheme to satisfy mentioned properties in half number of iterations comparing to original scheme.

In section 2 we present some preliminaries about elementary CA and reversible CA. In section 3 Seredynski-Bouvry encryption scheme is introduced. Our study of Seredynski-Bouvry scheme is presented in 4.

2 Preliminaries

2.1 Elementary Cellular Automata

Elementary CA(ECA) is an array of cells, in which, each cell is assigned a value from a set of state alphabet. All cells are updated synchronously by application of a rule $R$ in discrete times. An ECA can be explicitly expressed by its size, the rules to be applied, initial state and formation of neighborhood. Size of CA defines number of cells contained in CA. When CA size is finite, usually, it is considered as a ring. Each application of rule to all cell in CA at time $t$ is called an iteration. Before first iteration cells may be assigned values which is called initial state. Next state of each cell $S_{i}^{t+1}$ at time $t$ can be obtained by :

$$S_{i}^{t+1} = R(S_{i-r}^{t}, ..., S_{i-r-1}^{t}, S_{i}^{t}, S_{i+1}^{t}, ..., S_{i+r}^{t}).$$  (1)
In Eq. 1, \( r \) defines neighborhood radius and \( S_t^i \) shows the state of \( i \)-th cell. Number of possible rules is \( 2^k \) in which \( k = 2^{2r+1} \). Proposed CA in [1] is a one-dimensional uniform CA with binary alphabet \( \{0, 1\} \). Neighborhood radius is either 2 or 3, for size 32 and 64 cells.

### 2.2 Reversible Cellular Automata

By application of rules on each cell \( s_i \) in configuration \( q_t \), we get a new configuration \( q_{t+1} \). We can view this transformation as a global transition function which map configuration \( q_t \) to \( q_{t+1} \). A CA is reversible if only if its global transition function is one-to-one. Study of elementary CA showed only few rules have this property. For example, considering all 256 rule with neighborhood radius 1, only six rules are reversible. Hence, a new class of reversible introduced in [5]. In addition to current state of cell, rules in this reversible CA are depending on state of cell in the previous iteration. Eq. 2 defines how the rule applies to a cell and its neighbors.

\[
S_t^{i+1} = R(S_t^{i-r}, \ldots, S_t^{i-1}, S_t^i, S_t^{i+1}, \ldots, S_t^{i+r})
\]  

(2)

In Eq. 2, \( S_t^{i+1} \) depends on state of cell in iterations \( t \) and \( t - 1 \). These rules can be easily derived from ECA rules. The rule in RCA is comprised of two elementary rules. One is applied when the cell state is 1, otherwise second rule is applied. Figure 1 shows an example of such rules. If the first rule is known \( R_1 \), the second rule can be easily derived by following equation:

\[
R_2 = 2^d - R_1 - 1.
\]  

(3)

in which, \( d = 2^{2r+1} \) and \( r \) is the neighborhood radius. Considering that each of these rules are depending on state in the previous iteration, the initial configuration of CA is comprised of two initial configuration \( q_0 \) and \( q_1 \). These rules are also used to proceed in backward or forward direction.

Figure 1: reversible rule 236/19 [1]
3 Seredynski-Bouvry encryption scheme

In this section we introduce Seredynski-Bouvry CA-based block cipher \[1\]. In this scheme, plain-text is encoded as a part of initial configuration, \(q_1\). \(q_0\) part of initial configuration is filled by random data. Encryption process is done through iterating in forward direction. This process illustrated in figure 2. After \(n - 1\) iterations, configuration \(q_{n-1}\) contains cipher-text. In this scheme, \(q_n\) must be delivered to other party securely, by encryption or through a secure channel. If plain-text length is more than one block, it is divided into \(n\) blocks, and each block is encrypted separately. \(q_0\) of first block is filled by random data. \(q_0\) configuration of other blocks are filled by final data of previous block. Final data of last block, \(q_n\), must delivered to other party securely. Figure 3 illustrates the encryption process for multi-block data. Decryption process is depicted in figure 4. In decryption process initial configuration is composed of cipher-text and final data. After that \(q_0\) and \(q_1\) are filled by final data and cipher-text respectively and CA will be iterated \(n - 1\) times to obtain the plain-text.

So called avalanche property of a cipher, is that even a single change in plain-text or key must lead to significant changes in cipher-text. Strict avalanche criteria states that by any change in plain-text or key, each bit of cipher-text must be flipped with probability 0.5. In \[1\], Seredynski-Bouvry scheme is studied against this criteria for CA with size 32 and 64 cells. Neighborhood radius is taken 2 and 3 for CA size 32 and 64, respectively. Figures 5 and 6 show relation of number of iteration and behavior of the proposed cipher against strict avalanche criteria for CA with size 32 and 64, respectively. As stated in \[1\], initial data, rules are selected randomly.
Figure 3: Encryption process of multi-block data

Figure 4: Decrypting a block of data by Seredynski-Bouvry scheme
4 Our study of Seredynski-Bouvry scheme

We implemented Seredynski-Bouvry encryption scheme using C++ in Ubuntu. Just as [1], we also select CA size to 32 and 64 cells.

4.1 Strict Avalanche Criteria

For our experiments, we generate a random rule and obtain its complement by 3. This experience is done over 10000 random plain-texts and initial data. In our test, the rule is generated randomly and it is fixed for each neighborhood radius. Figures 7 and 8 illustrate results for CA with sizes 32 and 64 cells.

These result are compatible with results presented in [1]. As illustrated in figures 7 and 8 greater neighborhood radius leads to satisfaction of SAC in smaller number of iterations. So, we can establish an trade-off of between performance and security.

4.2 Statistical testing of randomness

Block cipher are also can be used as a pseudo random number generator for various cryptographic application. So sequence of bits generated by the
underlying block cipher, must be random. NIST statistical test suite, is a collection statistical test to verify whether a sequence of bits is whether random. NIST statistical tests is comprised of 15 different tests.

In table 1 we bring results of some of tests, which may not be passed for radius two, for different neighborhood radius. As it showed in table 1 CA with neighborhood size two can not pass statistical tests. So the size of neighborhood radius should be greater than 3.

Table 1: NIST Statistical Tests

| Statistical Test /neighborhood radius | $r = 2$ | $r = 3$ | $r = 4$ | $r = 5$ | $r = 6$ |
|--------------------------------------|---------|---------|---------|---------|---------|
| Frequency                            | -       | +       | +       | +       | +       |
| Frequency in block                   | -       | +       | +       | +       | +       |
| Run                                  | +       | +       | +       | +       | +       |
| Binary Matrix Rank                   | -       | +       | +       | +       | +       |
| Fast Fourier Transform               | -       | +       | +       | +       | +       |
| Universal Maurer Test                | -       | +       | +       | +       | +       |
Figure 7: strict avalanche criteria for CA with 32 cells and radius $r = 2$ to $r = 6$

Table 2: boolean function equivalent to three rules 0x1, 0x55555555 and 0x2B722D4.

| Rule       | Equivalent Boolean Function                                      |
|------------|-----------------------------------------------------------------|
| 0x1        | $\neg a \land \neg b \land \neg c \land \neg d \land \neg e$  |
| 0x55555555 | $\neg e$                                                        |
| 0x2B722D4  | $(a \land \neg b \land \neg c \land \neg d) \oplus (b \land \neg c \land \neg d \land \neg e) \oplus$  |
|            | $(\neg b \land \neg c \land \neg d \land \neg e) \oplus (a \land \neg b \land \neg c \land \neg d \land \neg e) \oplus$  |
|            | $(\neg a \land b \land c \land \neg d \land \neg e) \oplus (\neg a \land \neg b \land c \land \neg d \land \neg e)$  |

4.3 Effect of rules

In this scheme, rules are part of key and naturally selected randomly. Selected rules have direct effect on cryptographic properties of the cipher. We study the effect rule to SAC for three rules: 0x1, 0x55555555, 0x2B722D4. In 9 illustrates effect of these rule on satisfying SAC, for a CA with size 32 and neighborhood radius two. As it showed, selection of rule have direct effect on satisfying cryptographic properties, so rules should be selected carefully and cannot be selected as a part of key just by a random process.

5 Improvement of Seredynski-Bouvry Scheme

The neighborhood defined in [1] is a kind of normal neighborhood, that is, the propagation of changes in state of cells of array to other cells is not
fast enough. We can improve propagation by changing the neighborhood definition. The new neighborhood defined in Eq. 4

\[ S_{0i}^{t+1} = R(S_i^{t-r}, \ldots, S_{i-1}^{t}, S_i^{t}, S_{i+1}^{t-1}, \ldots, S_i^{t+r}) \] (4)

In Eq. 4 next state of \( S_{0i} \) is obtained through application of rule on cells \( i - r \) through \( i + r \). In figure 10 we compare two neighbourhood definitions. As we see our definition of neighbourhood, leads to satisfaction of SAC in smaller number of iterations (about half).

6 Conclusion

In this paper we studied Seredynski and Bouvry encryption scheme [1] against strict avalanche criteria and NIST randomness tests. We also studied effect rule in satisfying aforementioned criteria. It showed that randomly selected rules may weaken security of scheme. So, rules must be selected with respect to cryptographic properties such as balancedness and completeness. We changed neighbourhood definition, such that strict avalanche criteria can be satisfied in half number of iterations comparing to original scheme.

References

[1] M. Seredynski and P. Bouvry, “Block encryption using reversible cellular automata,” in *Cellular Automata* (P. Sloot, B. Chopard, and A. Hoek-
Figure 9: SAC for CA with three rules $0x1$, $0x55555555$ and $0x2B722D4$.

Figure 10: comparison of effect of neighborhood definition with [1] and ours

[1] F. Seredyński, P. Bouvry, and A. Y. Zomaya, “Cellular automata computations and secret key cryptography,” *Parallel Computing*, vol. 30, no. 5, pp. 753–766, 2004.

[2] S. Sen, C. Shaw, D. Chowdhuri, N. Ganguly, and P. Chaudhuri, “Cellular automata based cryptosystem (cac),” in *Information and Communications Security* (R. Deng, F. Bao, J. Zhou, and S. Qing, eds.), vol. 2513 of *Lecture Notes in Computer Science*, pp. 303–314, Springer Berlin Heidelberg, 2002.
[4] A. Ghaemi Bafghi, M. R. Meybodi, and B. Sadeghiyan, “Cellular automata and its application to cryptography,” in *Iranian Conference on Electrical Engineering*, no. 7, (Tehran, Iran), pp. 279–286, ITRC, May 17-19 1999.

[5] S. Wolfram, *A New Kind of Science and a New Kind of Science Explorer*. Wolfram Media Incorporated, 2002.