BDD-based Cryptanalysis of LFSR Stream Ciphers

Srđan Đorđević, S. Bojanić and O. Nieto-Taladriz

Abstract - Binary Decision Diagram (BDD) is binary variant of the data structure called Decision diagram that is aimed to discrete function representation. BDDs algorithms are used as an effective way to represent Boolean functions and are very efficient in terms of space and time complexity. They raised a lot of interest in cryptanalysis of Linear Feedback Shift Register (LFSR) that is one of the most important and widely used building blocks for keystream generators. LSFR is well suited for hardware and software implementations and produce very uniformly distributed output streams. Best known LFSR-based stream ciphers are E_0 used in Bluetooth wireless LAN, A5/1 and A5/2 used in GSM standard of cell phones, the shrinking generator, etc. In this paper, different cryptanalytic variants of BDD like Free Binary Decision Diagram (FBDD), Ordered BDD (OBDD), Zero-suppressed BDD (ZBDD), etc. are discussed and further research directions are outlined.

Keywords –Cryptanalysis, Binary Decision Diagrams.

I. INTRODUCTION

A stream cipher generates bit by bit a keystream, which is used to encrypt the plaintext. The keystream produced by a stream cipher should be as random looking as possible in order to make it more resistant to attacks. The difficult task is to make stream cipher secure and at the same time provide excellent software or hardware performance. Much recent cryptography research has been focused on stream ciphers that offer faster performance in some architecture (8, 16, 32 or 64-bit) and smaller hardware implementation in terms of gates, area or power consumption.

In 2002, Krause introduced the concept of BDD-based attacks [1]. Attacks on several generators were presented, including *A5/1*, E_0 and the self-shrinking generator. Leater Shaked and Wool introduced their OBDD-based attack to E_0 key stream generator.

The time complexity of the algorithms is determined by the space complexity of the synthesized Binary Decision Diagrams throughout the entire process of construction.

Srđan Đorđević is with the Department of Electronics, Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia, E-mail: srdjan.djordjevic@elfak.ni.ac.rs

Slobodan Bojanić and Octavio Nieto-Taladriz are with Universidad Politecnica de Madrid, Departamento de Ingenieria Electronica, Ciudad Universitaria s/n, 28040 Madrid, Spain
E-mail: slobodan@die.upm.es (S. Bojanić), nieto@die.upm.es (O. Nieto-Taladriz)

In Section 2 we concern LFSR-based keystream generators. Section 3 is dedicated to the Binary Decision Diagram (BDD) and its variants used in cryptanalsyis. Brief introduction to BDD-attack against LFSR-based key straem generators is given in section 4. Two adapted and optimized attacks of Krause algorithm for the specific details of E_0 system, OBDD-attack and ZBDD-attack are described in Sections 5 and 6 respectively.

II. LFSR-BASED STREAM CIPHERS

A stream cipher is a symmetric key cipher which generates a pseudorandom bit stream (keystream) used to encrypt the plaintext. Digits of the plaintext (usually bits or bytes) are combined with a keystream by an exclusive-or operation to produce ciphertext.

Linear feedback shift register (LFSR) is a shift register whose input bit is a linear combination of its previos state. Implementation of LFSR, shown in Fig. 1, consists of a simple shift register in which a binary-weighted modulo-2 sum of the taps is fed back to the input. For any given tap, weight g_i is either 0, meaning "no connection," or 1, meaning it is fed back.

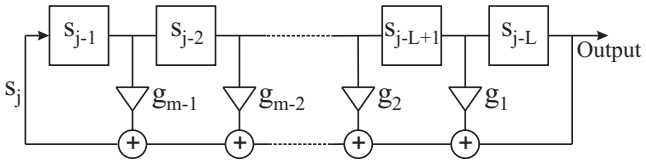

Fig. 1. Model of the linear feedback shift register (LFSR)

LFSR is widely used in stream ciphers as a pseudo-random number generator. It is well suited for hardware and software implementations and produce very uniformly distributed output streams with long periods.

The LFRSR-based keystream generators consists of two components, a linear bitstream L and a nonlinear compression function C. They generate keystream according to the rule $y=C(L(k))$ for the cipher key k. Linear bitstream generator produce a linear bitstream $L(k)$ by one or more parallel LFSRs.

Since LFSR is a linear system, cryptanalysis of their output sequences is very simple. Additional block of LFSR-based keystream generators is a nonlinear compression function, C that provide cryptographic security.

Best known LFSR-based stream ciphers are E_0 used in Bluetooth wireless LAN, *A5/1* and *A5/2* used in GSM standard of cell phones, the shrinking generator. Most of

the know cryptographic attacks are key recovery attacks. There are two categories of key stream attacks, short and long, according to the required known key-stream.

E_0 is a LFSR-based key stream generator used in the Bluetooth protocol. It employs 4 shift registers of differing lengths (25, 31, 33, 39 bits) and a nonlinear combiner logic, that consists of summation combiner logic and blend machine. Nonlinear combiner logic is usually represent as a 4 bit finite state machine.

The sum of the four output bits of the LFSR's is input into the finite state machine to update the state of the machine. At each clock tick E_0 generates output bit of the encryption system stream using outputs of the shift registers and two internal states, each 2 bits long. Practically, the output of the four LFSRs is xor-ed with the output bit of the nonlinear combiner logic. The secret key length is generally 128 bits.

III. BINARY DECISION DIAGRAM BDD

A binary decision tree (BDT) is a tree data structure that is used to represent a Boolean function. Each nonterminal node has exactly two children. Nonterminal nodes are labeled by boolean variables, while sink nodes, or leaves are labeled by 0 or 1. Every two different vertices on a single path are labeled by distinct variables. BDT determines a unike boolean function from the variables in nonterminal nodes, while consumes as much space as truth table.

Binary Decision Diagrams (BDDs) [2] are canonical directed acyclic graphs. A graph is canonical if multiple identical nodes are not allowed. Two nodes are identical if they have the same label, and their respective child nodes are also identical. Important property of the BDD is that computation results are stored for future reuse.

A BDD is derived from a BDT by elimination two obvious kinds of redundancies. Transformations to eliminate redundancies of BDT are:
(1) *elimination of redundant tests*: Nonterminal vertex whose two children vertices are the same can be removed.
(2) *merging isomorphic subdags*: Repeated appearance of the same subtrees are merged into one subtree.

Fig. 2 shows an example of a BDT and corresponding BDD, that is obtained by implementation of the above transformations.

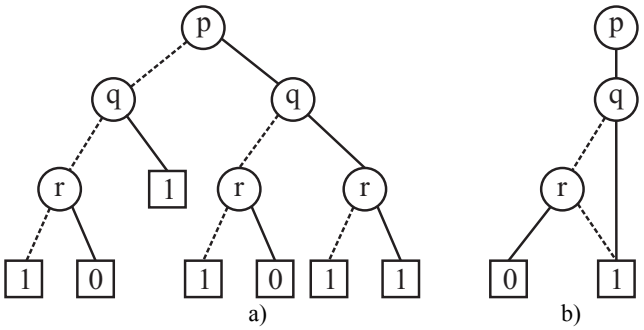

Fig. 1. a) Binary Decision Tree b) Binary Decision Diagram

It is obvious improvement of the data structure compactness.

Variants of BDD used in cryptography analysis so far are Free Binary Decision Diagram (FBDD), Ordered Binary Decision Diagram (OBDD) and Zero Decision Diagram (ZDD).

FBDD is a BDD in which along each path variables appears at most once. An Ordered BDD is a BDD in which each variable is encountered no more than once in any path and the order of variables is same along each path.

A Reduced BDD (ROBDD) is an OBDD that is reduced by two reduction rules: deletion rule and merging rule. These Reduction rules remove redundancies from the OBDD.

Zero-suppressed Binary Decision Diagrams (ZBDDs) are a special type of BDDs which were introduced for efficient manipulation of sparse item combinations [3].

An itemset p can be represented by a n-bit binary vector $(x_1, x_2, \ldots, x_n)$, where $x_i = 1$ if item i is contained in p. A set S of itemsets can be represented by a characteristic function $X_S(p): \{0,1\}^n \to \{0,1\}$ where $X_S(p) = 1$ if $p \in S$ and 0 otherwise. More specifically, a ZBDD is a BDD with two reduction rules:

1. Merging rule: merge identical subtrees (to obtain canonicity);
2. Zero-suppression rule: delete nodes whose 1-child is the sink-0, and replace them with their 0-child.

By utilising these rules, a sparse collection of item combinations, which can be seen as a boolean formula, can be represented with high compression.

IV. FBDD BASED CRYPTANALYSIS OF LFRS-BASED KEYSTREAM GENERATORS

The cryptanalysis of the keystream generators consists of finding a secret key k fulfilling $y=E(k)$, for a given keystream y and given encryption algorithm E. In the algorithm proposed by Krause [1] the problem of finding a secret key k is reduced to problem of finding the minimal FBDD P for the decision whether k fulfils $y = E(k)$. The important properties of FBDD is that they can be efficiently minimized and allow an efficient enumeration. The algorithm consists of three steps:

- Construction of minimal FBDD Q_m which decided whether bit stream Z, $z \in \{0,1\}^m$ for all $m \geq 1$ produce a prefix of keycipher k_{cipher}.
- Construction of minimal FBDD R_m based on the feedback polynomials that decided whether Z can be produced by Linear bitstream generator.
- Construction of third set of FBDD's P which are result of intersection of the Q_m and R_m.

Kraus algorithm incrementally computes FBDD for increasing number of bits. The cryptanalsys algorithm proposed by Kraus is very efficient against LFSR-based

keystream generators. The weakness of LFSR-based generators can be explained by small memory of the compressor *C* as a result of online manner of keystream generation.

Synthesis operation bounds the size of the synthesis result as:

$$|SYNTH(P,Q)| \le |R| \cdot |Q| \quad (1)$$

where $|Q|$ is size of the OBDD representing the compressor $|R|$ is an LFSR consistency check OBDD.

The authors of the algorithm reported, based on some estimation, that their attack against E_0 requires $O(2^{77})$ space complexity and $O(2^{81})$ time complexity.

V. OBDD BASED CRYPTANALYSIS OF E_0 KEY STREAM GENERATOR

Optimization of general OBDD-based attack for specific details of E_0 system was introduced by Shaked and Wool [4], which proposed OBDD-based cryptanalysis. This algorithm uses OBDD instead of FBDD and propose a new composable BDD for the compressor.

OBDD based attack uses regularities of E_0 key generator, reflected in fact that every clock tick each LFRS is stepped once and from each LFSR one output bit is input to the compressor. Variable ordering of the internal bitstream is expressed in terms of the clock tick index m $(0 \le m \le 127)$ and index of the LFSR $(1 \le i \le 4)$, as $j = 4 \cdot m + i - 1$. This indexing method leads to separate four equations for linear bitstream assigned to each of the four shift generators. Consequently algorithm includes construction of OBDD's for each bit of the bitstream and associated to a distinct LFSR. Different length of the shift generators requires adjustment of the algorithm.

Each internal bit is produced by one of the LFSR's, and depends on four earlier bits of the same shift register. Algorithm produces BDD, according to the LFSR's feedback polynomial, that make decision whether the internal bit z_k is consistent with the prefix $\{z_j\}_{j=1}^{k-1}$. Table I summarizes the basic consistency equations in the case of two indexing methods for each of the LFSR's.

TABLE I
CONSISTENCY RELATIONS

LFSR	Basic consistency equation
1	$z_i = z_{i-8} \otimes z_{i-12} \otimes z_{i-20} \otimes z_{i-25}$
2	$z_i = z_{i-12} \otimes z_{i-16} \otimes z_{i-24} \otimes z_{i-31}$
3	$z_i = z_{i-4} \otimes z_{i-24} \otimes z_{i-28} \otimes z_{i-33}$
4	$z_i = z_{i-4} \otimes z_{i-28} \otimes z_{i-36} \otimes z_{i-39}$
LFSR	Normalized consistency equation
1	$z_i = z_{i-32} \otimes z_{i-48} \otimes z_{i-80} \otimes z_{i-100}$
2	$z_i = z_{i-48} \otimes z_{i-64} \otimes z_{i-96} \otimes z_{i-124}$
3	$z_i = z_{i-16} \otimes z_{i-96} \otimes z_{i-112} \otimes z_{i-132}$
4	$z_i = z_{i-16} \otimes z_{i-112} \otimes z_{i-144} \otimes z_{i-156}$

An OBDD representing an LFSR consistency relation contains 5 variables and 11 nodes.

The second step in the OBDD-attack against E_0 system is construction of the OBDD that represents the compressor unit. This OBDD is built according to the transfer function of the compressor and known keystream bits.

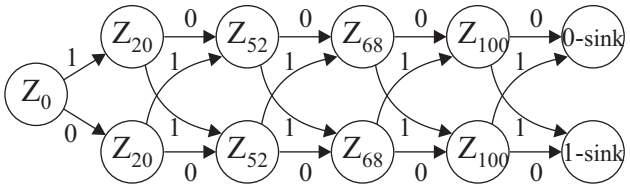

Fig. 2. OBDD representing consistency check for z_{100}

The value of the compressor unit is updated by the sum of the LFSR's output bits. The obtained BDD structure is called basic chain and practically represents the sum of 4 bits. This OBDD structure is illustrated on Fig. 3. The compressor unit consists of 16 identical basic chains, for each of states.

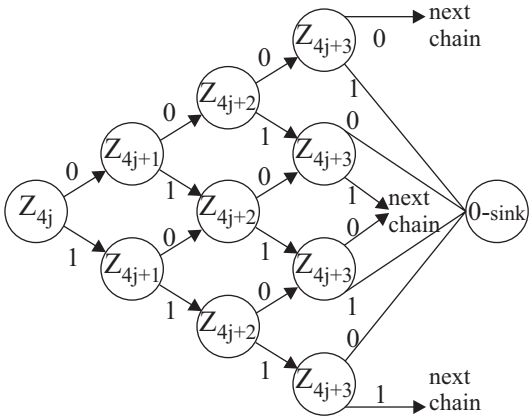

Fig. 3. Basic chain representing sum of 4 bits

The analysis of the algorithm complexity consists of the OBDD space complexity estimation. The size of synthetized OBDD $|P|$ is limited by two bounds. First is the number of satisfying assignments

$$|P| \le m \cdot |One(P)| \quad (2)$$

where $One(P)$ denotes the set of satisfying assignments of the BDD *P*, and *m* is the number of variables that BDD containts. Linear bitstream generator introduces 4 new variables each clock tick. There is one constraint because the output bit is known. It can be conclude that the number of satisfying assignments is multiplied by 2^3 per clock tick.

The second bound is synthesis operation bound

$$|P| \le |Q(m)| \cdot 2^{m-n} \quad (3)$$

where $|Q(m)|$ is size of the OBDD representing the compressor, *m* is the number of variables, *n* is size of the given keystream. This bound assumes that the number of introduced OBDD nodes during LFSR consistency check is duplicated by each new variable. The maximal number of the OBDD nodes during attack, $|P| \approx 2^{86}$, is find as intersection point of two bounds.

VI. ZDD BASED CRYPTANALYSIS OF E_0 KEY STREAM GENERATOR

ZDD-attack against E_0 generator [5] is based on the general FBDD-attack but with different data structure. ZDD is a variant of BDD obtained in such a way that one of the reduction rule is changed. Each path from the root to the terminal vertex 1 corresponds to one of the combinations.

The motivation of using ZDD data structure in cryptanalysis is that it is better suited for representation of sets than BDD. This data structure is especially efficient in manipulation with set of combination, that is represented by an binary vector. Each bit in this vector denotes presence or absence of an item in the combination. The set of combination can be shown with Boolean function called characteristic function.

Graph that decided whether a bit stream *Z* can be produced by linear stream generator, denoted as R_m, is constructed using ZDD data structure instead of OBDD or FBDD.

Data structure that represents the compressor unit and decide whether a bit stream $C(Z)$ produce a prefix of keycipher is denoted as Q_m. Since finite state machine of E_0 generator has 16 state it can be represent with 4 one bit variables q_i^n for $(1 \le i \le 3)$. The following function for Q_m should be computed after riding *m*+1 input symbols z_i:

$$F\left(q_3^{m+1}, q_2^{m+1}, q_1^{m+1}, q_0^{m+1}, z_{4m+3}, z_{4m+2}, z_{4m+1}, \ldots, z_0\right) \quad (4)$$

Algorithm mapped the problem to a combinatorial set problem. The computing of Q_m is reduced to checking of all possible combinations of input bits and final states. Most operations on sets are defined and implemented for ZDD data structure.

The number of variables and constraints during some steps of the algorithms is changeable and can be expressed in function of the length of each shift register, L_i for $0 \le i \le 3$. During first $|L_0|$ steps algorithm introduce 4 new variables and one constraint, then the number of assignments is multiplied by 2^3. After $|L_1|$ steps the output of the first register L_0 is known and represents additional constraint, then the number of assignments is multiplied by 2^2. In the same way after $|L_2|$ steps the number of assignments is multiplied by 2^1. After $|L_3|$ steps, the number of constraints is equal the number of variables, then the number of assignments will be constant.

The overall time complexity of the algorithm is 2^{82} and spice complexity of 2^{23}.

VII. CONCLUSION

We have presented a short overview of the BDD-based algorithms in cryptography. The paper provides description of algorithms and their complexity analysis.

The attacks are based on a backtracking approach [6], that build a binary search tree according to the feedback polynomials of LFSR and nonlinear compression function. BDD data structure is successfully performed in cryptography analysis as a compact and canonical presentation of the Binary Decision Tree.

One possible direction of future research is improvement of space consuming of FBDD-attack, which needs a lot of space for all constructed intermediate diagrams. Another open question is check whether FBDD-attack could be combined with other methods of cryptanalysis.

REFERENCES

[1] Krause, M., „*BDD-Based Cryptanalysis of Keystream Generators*“, In EUROCRYPT, Vol. 2332 of LNCS, 2002, pp. 222–237.

[2] Bryant, R. E., “*Graph-based algorithms for boolean function manipulation*“, IEEE Transactions on Computers, Vol. C-35, No. 8, Aug., 1986, pp. 677–691.

[3] Minato, S., “*Zero-suppressed BDDs and their applications*“, International Journal on Software Tools for Technology Transfer (STTT), Vol. 3, No. 2, 2001, pp. 156–170.

[4] Shaked, Y., Wool, A., “*Cryptanalysis of the Bluetooth E0 cipher using OBDD's*“, In Proceedings of 9th Information Security Conference, LNCS 4176, 2006, pp. 187–202.

[5] Ghasemzadeh, M., Meinel, Ch., Shirmohammadi, M., Shazamanian, M. H., “*ZDD-Based Cryptanalysis of E0 Keystream Generator*“, In Proceedings of 3th International Conference on Mathematical Sciences (ICM 2008), Mar., 2008.

[6] Zenner, E., Krause, M., Lucks, S.,“*Improved cryptanalysis of the self-shrinking generator*“ In V. Varadharajan and Y. Mu, editors, Australasian Conference on Information Security and Privacy ACISP'01, Lecture Notes in Computer Science, Vol. 2119, 2001, pp. 21-35.