PUBLICATIONS DE L'INSTITUT MATHÉMATIQUE
Nouvelle série, tome 93 (107) (2013), 109–115 DOI: 10.2298/PIM1307109L

COMPARISON OF RANDOM S-BOX GENERATION METHODS

Dragan Lambić and Miodrag Živković

Communicated by Žarko Mijajlović

ABSTRACT. Random bijective S-box generation methods are considered. An alternative S-box generation method by forming compositions of permutations from some fixed set is proposed. Experiments show that the rate of acceptable S-boxes for all the methods considered is essentially the same. The advantage of the composition method is an obvious parametrization, with the potentially large key space.

1. Introduction

A number of known block ciphers are of substitution-permutation (SP) type. S-boxes are used in such cipher systems as the important nonlinear component. A strong block cipher should be resistant to various attacks, such as linear and differential cryptanalysis. In SP networks this is generally achieved if the S-boxes used satisfy a number of criteria, such as strict avalanche criterion (SAC), bit independence criterion (BIC), nonlinearity, XOR Table Distribution and maximum expected linear probability (MELP, see for example [**1**, **2**]).

The S-box used in encryption process could be chosen under the control of key, instead of being fixed. For example in [**1**, **3**, **4**] random key-dependent S-boxes are generated once per encryption without discarding any S-boxes, and in [**5**] the S-boxes are generated until a good one is found.

Here we consider random bijective S-box generation methods satisfying chosen selected criteria. Jakimoski and Kocarev [**6**] generated the chaotic S-boxes by discretizing the exponential and logistic maps; the numbers of iterations of the discretized exponential map were considered as the keys, making the S-boxes key-dependent. The S-box generation method based on a 2D discretized chaotic Baker map was proposed in [**7**]. Afterwards, the 2D was further extended to a 3D one [**8**].

2010 *Mathematics Subject Classification:* 15A21; 15A36.
Key words and phrases: S-box, random permutation, Walsh transform.
Partially supported by Ministry of Science and Technology of Serbia, Grant 174021.

Yin et al. [**9**] presented S-box generation method based on the iteration of continuous chaotic maps, with the starting points depending of the key. In [**10**], a generation method is proposed, which uses the Lorenz system and special shifting method to generate the S-box. The S-boxes obtained by various methods are checked afterwards, and discarded if they do not satisfy common set of criteria (see for example [**2**, **11**–**13**]).

We propose another simple method to obtain random S-boxes. After choosing some fixed set of starting S-boxes, output S-boxes are obtained by making various compositions of the starting S-boxes. The sequence of the indices of starting S-boxes used is key-controlled. These methods are experimentally compared, by generating a number of $n \times n$ S-boxes, $n = 8, 10, 12$. It turns out that the rate of S-boxes satisfying all the criteria does not depend substantially on a generation method. The advantage of the proposed generation method is the size of the key space.

2. Notation

Let $B = \{0, 1\}$. S-box of the type $m \times n$ is a function $f : B^m \to B^n$. S-boxes appearing inside block ciphers are expected to satisfy some standard criteria. Here we consider only bijective S-boxes, where $m = n$. The vector $x = (x_1, x_2, \dots, x_n) \in B^n$ naturally corresponds to the integer $\sum_{i=1}^{n} x_i 2^{n-i}$, which is also denoted by x, without causing misunderstanding. Bijective S-box $f : B^n \to B^n$ is a permutation of the set $\{0, 1, \dots, N-1\}$, where $N = 2^n$. It is represented by the integer vector $\{f(0), f(1), \dots, f(N-1)\}$ which is also denoted by f, because there is no danger of confusion. A cycle is a permutation $(c_1, c_2, \dots, c_k)$ mapping c_i into c_{i+1}, $1 \leqslant i < k$, and mapping c_k into c_1, leaving the other elements in place. Transposition is a cycle of length two. The composition $h = fg$ of two permutations f and g of the same set A, is the permutation mapping each $x \in A$ into $h(x) = f(g(x))$.

Let $e_i = [\delta_{i,1}\ \delta_{i,2} \dots \delta_{i,n}]^T$, where

$$\delta_{i,j} = \begin{cases} 1, & i = j \\ 0, & i \neq j, \end{cases}$$

and let $(\cdot)^T$ denote a matrix transpose. Let $\oplus$ denote exclusive or operation.

The nonlinearity of a function $\phi : B^n \to B$ is defined by

$$N(\phi) = 2^{n-1} - \frac{1}{2} \max_{a \in B^n} \left| \sum_{x \in B^n} (-1)^{f(x) + a \cdot x} \right|$$

(see [**14**] for example). The complexity of computation of $N(\phi)$ using Walsh transform is $O(n2^n)$. Let $P_{i,j}(f) = 2^{-n} \sum_{x \in B^n} f_j(x) \oplus f_j(x \oplus e_i)$.

Linear probability is defined by $LP(a, b) = \left(2^{-n} \sum_{x \in B^n} (-1)^{a \cdot x + b \cdot f(x)}\right)^2$.

Given a function $f : B^n \to B^n$, $f = (f_1, f_2, \dots, f_n)$ (where $f_i : B^n \to B$), let

$$N(f) = \min\{N(f_i) \mid 1 \leqslant i \leqslant n\},$$

$$B(f) = \min\{N(f_i \oplus f_j) \mid 1 \leqslant i < j \leqslant n\},$$

$$S(f) = \frac{1}{n^2} \sum_{1 \leqslant i \leqslant n} \sum_{1 \leqslant j \leqslant n} \left| \frac{1}{2} - P_{i,j}(f) \right|$$

$$X(f) = \max_{\Delta x \in B^n \setminus \{0\},\ \Delta y \in B^n} \{x \in B^n \mid f(x) \oplus f(x \oplus \Delta x) = \Delta y\}$$
$$L(f) = \max_{a,b \in B^n \setminus \{0\}} LP(a,b).$$

The value $N(f)$ is a measure of the nonlinearity of f. Furthermore, $S(f)$, $B(f)$, $X(f)$ and $L(f)$ measure degree to which f satisfy strict avalanche criterion (SAC), output bits independence criterion (BIC), equiprobable input/output XOR distribution criterion (XOR) and maximum expected linear probability (MELP), respectively. Equiprobable input/output XOR distribution criterion (XOR) is also known as maximum expected differential probability (MEDP).

Let a, c, d be integers, $0 < a, c, d < 2^{n-1}$, and let $0 < b, e < 1$. We say that the S-box $f : B^n \to B^n$ is (a, b, c, d, e) acceptable if

$$N(f) \geqslant a,\ \ S(f) \leqslant b,\ \ B(f) \geqslant c,\ \ X(f) \leqslant d,\ \ L(f) < e. \tag{1}$$

These five criteria will be denoted by C_1, C_2, C_3, C_4, C_5, respectively. The values $N(f), S(f), B(f), X(f), L(f)$ can be computed in time $O(n^2 2^n)$, $O(n^2 2^n)$, $O(n^3 2^n)$, $O(n4^n)$, $O(n4^n)$, respectively. Therefore, it is most efficient to check if f satisfy the conditions 1, following the order C_i, $i = 1, 2, 3, 4, 5$; if f fails to satisfy C_i for some i, $1 \leqslant i \leqslant 5$, the computation is stopped, and f is discarded.

3. Random S-box generation methods

For given n, after choosing appropriate values of a, b, c, d, e, all $n \times n$ S-box generation algorithms that will be considered, generate a sequence of permutations of $\{0, 1, \ldots, N-1\}$, $N = 2^n$, leaving only (a, b, c, d, e) acceptable ones. There have been recently proposed a number of random bijective S-box generation algorithms, differing mainly in a random permutation generation method used. Each permutation is then checked to see if it satisfies conditions 1.

For the sake of completeness, the two "classical" algorithms to obtain random permutation of $\{0, 1, \ldots, N-1\}$ are included here. They both use a random number generator rnd(), giving the pseudorandom numbers uniformly distributed on $[0, 1]$. The algorithm *perm1* obtains the i-th element $f[i]$ of a permutation f repeatedly computing $a = \lfloor N \cdot \mathrm{rnd}() \rfloor$ until a is different from all $f[j]$, $0 \leqslant j < i$. The average number of calls to rnd() is $\sum_{i=0}^{N-1} \frac{N}{N-k} = O(N \log N)$.

The algorithm *perm2* (Knuth shuffle, see [**15**]) gives the permutation equal to the product of $N-1$ transpositions $(N-i, r[i])$, $i = 0, 1, \ldots, N-1$, where $r[i] = \lfloor (N-i)\,\mathrm{rnd}() \rfloor$. The number of calls to rnd() in the worst case is $O(N)$, explaining why *perm2* is much more efficient than *perm1*.

Denote by $S0$ the $n \times n$ S-box generating algorithm using *perm2*, and the standard RNG (random number generator) rnd(), the part of C language.

The method proposed in [**7**] (which will be denoted by $S1$) uses the random number generator based on recurrent sequence $x_m = \tau(x_{m-1})$, $m \geqslant 1$, $x_0 \in [0, 1]$, where $\tau(x) = \mu x(1-x)$, $\mu \in [4 \cdot \frac{2^n - \frac{1}{2}}{2^n}, 4]$. Here μ is chosen so that the length of the interval $[0, 1] \setminus \{\tau(x) \mid x \in [0, 1]\}$ is at most $1/(2N)$. Given x, the state of the RNG, the next random number generated rnd() is $\tau(x)$, which also replaces the state x. After obtaining the permutation π by *perm1*, the permutation $\pi\beta^9$

is returned, where β is the Baker 2D-map [**7**]. Experiments show that the rate of (a, b, c, d, e)-acceptable S-boxes is not changed if the permutations are generated by more efficient algorithm *perm2* .

The method proposed in [**8**] (which will be called $S2$) uses similar RNG, based on the recurrent sequence, where $\tau(x) = \cos(k \arccos(x))$ $(k \in R)$; instead of Baker 2D map β, the Baker 3D-map [**8**] is used.

The method from [**9**] (which will be called $S3$) gives random permutations of B^n depending on the key — the sequence o of $N = 2^n$ integers from the interval $[0, N-1]$. The permutation returned is the product of $N-1$ transpositions $(i, r[i])$, $i = 0, 1, \ldots, N-1$, where

$$r[i] = \left\lfloor N\tau^m \left(\frac{i}{2^n} + \frac{o(i)}{4^n} \right) \right\rfloor,$$

and $m = 9$, for example.

4. Proposed S-box generation method

We now describe the proposed simple algorithm, which will be called $S4$. The set of $k > 1$ fixed bijective starting S-boxes $f_1, f_2, \ldots, f_k$ is used. For an arbitrary sequence of $m > 1$ indexes $i_1, i_2, \ldots, i_m \in \{1, 2, \ldots, k\}$, $S4$ returns the S-box $\prod_{j=1}^{m} f_{i_j}$. For example, if $n = 8$, $k = 5$, and f_1, f_2, f_3, f_4 are the bijective chaotic S-boxes from [**7**], [**8**], [**16**], [**17**] respectively, and if f_5 is the AES S-box, then the S-box $f_5 f_4 f_5 f_3 f_3 f_1 f_4 f_3 f_5$ is good enough — it is $(106, 0.02856, 100, 10, 0.071)$-acceptable.

The S-box generation method could be incorporated in practical system as follows. Communicating parties A and B share the set of starting S-boxes, the parameter (a, b, c, d, e) values and the key for chosen PRNG (pseudo random number generator). The PRNG is used to generate index sequences, until the first (a, b, c, d, e)-acceptable S-box is found. Let K be the total number of S-boxes to be generated by the system, i.e., the number of different keys; we suppose that keys are not to be repeated. The key space size $\log_2 K$ could be estimated as follows. Denote by δ the probability that the randomly chosen S-box is (a, b, c, d, e)-acceptable. Denote by ϵ the probability that all S-boxes generated are different. Starting from the approximate expression (*birthday problem*, see for example [**18**], ϵ small)

$$\epsilon \simeq \frac{K^2}{2\delta(2^n)!},$$

the size of key space is approximately $\frac{1}{2}\log_2 N! + \frac{1}{2}\log_2(2\epsilon\delta)$. If, for example, $\epsilon = \delta = 2^{-20} \simeq 10^{-6}$, then the size of key space is approximately $\frac{1}{2}\log_2 N! - 9.5 \simeq 832, 4375, 21615$ bits for $n = 8, 10, 12$ respectively. The speed of the S-box generation is proportional to the selection rate δ. It is determined mostly by the efficiency of checking the conditions C_4 and C_5. A small change in key bits causes substantially different sequence of index sequences, and therefore substantially different output S-boxes.

5. Experimental results

In order to compare the rates of (a, b, c, d, e)-acceptable S-boxes that could be obtained by algorithms $S0$, $S1$, $S2$, $S3$ and $S4$, each of these algorithms is used

TABLE 1. The bounds (a, b, c, d, e) used for corresponding characteristics of S-boxes.

	a	b	c	d	e
8	106	0.030	100	10	0.079
10	456	0.015	444	12	0.025
12	1920	0.008	1900	14	0.006

TABLE 2. The rates of S-boxes satisfying one of five criteria, among those satisfying previous ones.

$n = 8,\ M = M_0 = 10^7$										
	M_1	$100\frac{M_1}{M_0}$	M_2	$100\frac{M_2}{M_1}$	M_3	$100\frac{M_3}{M_2}$	M_4	$100\frac{M_4}{M_3}$	M_5	$100\frac{M_5}{M_4}$
$S0$	1455	0.015	257	17.7	43	16.7	14	32.6	9	64.3
$S1$	1452	0.015	246	16.9	48	19.5	15	31.2	9	60.0
$S2$	1431	0.014	282	19.7	52	18.4	21	40.4	12	57.1
$S3$	1514	0.015	273	18.0	48	17.6	10	20.8	7	70.0
$S4$	1340	0.013	236	17.6	33	14.0	16	48.5	9	56.2
$n = 10,\ M = M_0 = 10^5$										
	M_1	$100\frac{M_1}{M_0}$	M_2	$100\frac{M_2}{M_1}$	M_3	$100\frac{M_3}{M_2}$	M_4	$100\frac{M_4}{M_3}$	M_5	$100\frac{M_5}{M_4}$
$S0$	2336	2.3	87	3.7	48	55.2	17	35.4	16	94.1
$S1$	2273	2.3	100	4.4	40	40.0	16	40.0	15	93.7
$S2$	2265	2.3	82	3.6	33	40.2	11	33.3	11	100.0
$S3$	2282	2.3	87	3.8	41	47.1	14	34.1	13	92.9
$S4$	2267	2.3	84	3.7	39	46.4	14	35.9	14	100.0
$n = 12,\ M = M_0 = 10^4$										
	M_1	$100\frac{M_1}{M_0}$	M_2	$100\frac{M_2}{M_1}$	M_3	$100\frac{M_3}{M_2}$	M_4	$100\frac{M_4}{M_3}$	M_5	$100\frac{M_5}{M_4}$
$S0$	672	6.7	52	7.7	21	40.4	6	28.6	6	100.0
$S1$	619	6.2	47	7.6	21	44.7	5	23.8	5	100.0
$S2$	663	6.6	57	8.6	26	45.6	10	38.5	10	100.0
$S3$	624	6.2	52	8.3	25	48.1	9	36.0	9	100.0
$S4$	676	6.8	50	7.4	18	36.0	6	33.3	6	100.0

to generate and check $M = M(n) = 10^7, 10^5, 10^4$ random S-boxes for $n = 8, 10, 12$ respectively.

Let $M_0 = M$, and let M_i denote the number of S-boxes satisfying all the criteria $C_0, \ldots, C_i$, $1 \leqslant i \leqslant 5$. The values of bounds (a, b, c, d, e) are chosen so that the ratio M_i/M_{i-1} is not unreasonably small, $1 \leqslant i \leqslant 5$. The bounds for $n = 8$ are stronger than or equal to those from [**7**,**8**,**10**,**16**,**17**,**19**] except for nonlinearity bound in [**19**] obtained with the heuristic S-box generation method. No S-box satisfying C_1 with $a = 108$ was found, so the value $a = 106$ was chosen. The bounds chosen are shown in Table 1.

The results are given in Table 2. The numbers M_i and the ratios $100M_i/M_{i-1}$, $i = 1, 2, 3, 4, 5$, are shown for $n = 8, 10, 12$. It is seen that the ratios $100M_i/M_{i-1}$ do not vary substantially for different S-box generation methods.

6. Conclusion

Several methods for random S-box generation have been proposed in recent years. The results show that the rates of good S-boxes among those generated by various methods do not depend substantially on the method of generation. The consequence of the large complexity of the S-box testing is that the total generation times per one good S-box obtained do not differ substantially, also. The advantage of the proposed composition method is a possibility to achieve a large key space.

Acknowledgement. The authors are grateful to anonymous referee for useful comments leading to improvement of exposition.

References

1. L. Keliher, H. Meijer, S. Tavares, *A new substitution-permutation network cryptosystem using key-dependent s-boxes*, In: Proc. SAC'97, Canada, (1997), 13–26
2. L. Keliher, *Refined analysis of bounds related to linear and differential and linear cryptanalysis for the AES*, (In: H. Dobbertin et al., eds. Advanced Encryption StandardŮAES Š04, Bonn, 2004, Lect. Notes Comput. Sci. 2005, 42–57)
3. K. Kazlauskas, *Key-dependent S-box generation in AES block cipher system*, Informatica 20 (2009), 23–34
4. B. Schneier, *Description of a new variable-length, 64-bit block cipher (Blowfish)*, (In: Proc. Fast Software Encryption, Springer, 1994, pp. 191–204)
5. P. Mroczkowski, *Generating Pseudorandom S-Boxes Ű– a Method of Improving the Security of Cryptosystems Based on Block Ciphers*, J. Telecommun. Inform. Technol. (2009), 74–79
6. G. Jakimoski, L. Kocarev, *Chaos and cryptography:block encryption ciphers based on chaotic maps*, IEEE Trans Circuits SystŰI 48 (2001), 163–70
7. G. Tang, X. F. Liao, Y. Chen, *A novel method for designing S-boxes based on chaotic maps*, Chaos Solitons Fractals 23 (2005), 413–419
8. G. Chen, Y. Chen, X. F. Liao, *An extended method for obtaining S-boxes based on* 3*-dimensional chaotic baker maps*, Chaos Solitons Fractals 31 (2007), 571–579
9. R. Yin, J. Yuan, J. Wang, X. Shan, X. Wang, *Designing key-dependent chaotic S-box with larger key space*, Chaos Solitons Fractals 42 (2009), 2582–2589
10. F. Ozkaynak, A. B. Ozer, *A method for designing strong S-Boxes based on chaotic Lorenz system*, Phys. Lett., A 374 (2010), 3733–3738
11. C. Adams, S. Tavares, *Good S-boxes are easy to find*, (In: Advances in Cryptology: Proc. of crypto'89. Lect. Notes Comput. Sci., 1989, 612–615)
12. A. Webster, S. Tavares, *On the design of S-boxes*, (In: Advances in Cryptology: Proc. of CRYPTO'85. Lect. Notes Comput. Sci., 1986, 523–534)
13. E. Biham, A. Shamir, *Differential cryptanalysis of DES-like cryptosystems*, J. Cryptol. 4 (1991), 3–72
14. T. Cusick, P. Stanica, *Cryptographic Boolean Functions and Applications*, Elsevier, 2009
15. D. E. Knuth, *The Art of Coputer Programming; Vol 2: Seminumerical Algorithms*, Addison-Wesley, Reading, MA, 1969, 124–125
16. G. Chen, *A novel heuristic method for obtaining S-boxes*, Chaos Solitons Fractals 36 (2008), 1028–1036
17. M. Asim, V. Jeoti, *Efficient and simple method for designing chaotic S-boxes*, ETRI J. 1 (2008), 170–172

18. A. A. J. Menezes, P. C. Van Oorschot, S. A. Vanstone *Handbook of Applied Crytography*, CRC Press, 1997
19. Y. Wang, K. W. Wong, C. Li, Y. Li. *A novel method to design S-box based on chaotic map and genetic algorithm*, Phys. Lett., A 376 (2012), 827–833

(Received 25 09 2011)
(Revised 21 12 2012 and 20 02 2013)

Faculty of Education
University of Novi Sad
Sombor
Srbija
draganposao@yahoo.com

Matematički Fakultet
Belgrade University
Beograd
Srbija
ezivkovm@matf.bg.ac.rs