УДК 004.056.5 **DOI:10.30837/rt.2022.2.209.07**

О.О. КУЗНЕЦОВ, д-р техн. наук, М.О ПОЛУЯНЕНКО, канд. техн. наук, С.О. КАНДІЙ, О.І. ПЕЛЮХ

ДОСЛІДЖЕННЯ НОВОЇ ФУНКЦІЇ ВАРТОСТІ ДЛЯ ГЕНЕРАЦІЇ ВИПАДКОВИХ ПІДСТАНОВОК СИМЕТРИЧНИХ ШИФРІВ

Вступ

Алгоритми блокового та потокового шифрування із секретним ключем застосовуються у різних додатках інформаційної безпеки [1, 2]. Зокрема, вони є головним компонентом безпеки в інтернеті та в сучасних телекомунікаційних мережах, використовуються для шифрування великих сховищ даних, тощо. Отже проєктування сучасних шифрів, які забезпечують високу швидкість перетворення та криптографічну стійкість, є актуальною та важливою задачею [1 – 3].

Сучасні погляди на проєктування шифрів із секретним ключем базуються на концепції substitution-permutation networks (SPN) [4, 5]. SPN використовує прості для реалізації криптографічні примітиви (підстановки та перестановки), які у сукупності забезпечують властивості confusion та diffusion [6]. Ці властивості перешкоджають застосуванню статистичного, диференціального, лінійного та інших методів криптоаналізу [7 – 10]. Зокрема підстановки (substitutions, S-boxes) вносять нелінійність у співвідношення відкритий текст-шифр текст та забезпечують властивість confusion. Для захищеності від алгебраїчного криптоаналізу S-boxes повинні бути також випадковими [11 – 13], тобто підстановки не повинні містити простих алгебраїчних конструкцій, як, наприклад, в S-box шифрі AES [4, 14, 15].

Слід зазначити, що генерація криптографічно стійких випадкових підстановок є складною обчислювальною задачею. Зазвичай генерацію здійснюють алгоритмами локального пошуку: Hill climbing Algorithm [16 – 19]; Simulated Annealing [20, 21]; Genetic Algorithm [22 – 24] та інші. Це ітеративні алгоритми, пошук цільового рішення якими здійснюється із застосуванням спеціальних функцій вартості. На кожній ітерації алгоритм пошуку змінює поточний стан доки не буде досягнуто умову виходу: знаходження цільового рішення або виконання певної кількості ітерацій. Фактично, пошук цільового S-box здійснюється шляхом мінімізації (інколи максимізації) функції вартості. Однак пошук екстремуму та генерація високонелінійних S-boxes є надзвичайно складним завданням. Наприклад, для найбільш швидкого відомого результату для генерації S-box із нелінійністю 104 необхідно виконати не менше 65 тисяч ітерацій [18, 25].

В статті пропонується нова функція вартості та досліджується ефективність генерації високонелінійних випадкових S-box. Реалізовано Hill climbing алгоритм та проведено серію експериментів з генерації підстановок. Показано, що складність пошуку можна суттєво зменшити. Зокрема, для генерації S-box із нелінійністю 104 необхідно виконати менше 50 тисяч ітерацій.

Пов’язані роботи

Алгоритми локальної оптимізації для генерації високонелінійних підстановок досліджуються багатьма авторами. Зокрема, у [16 – 19] досліджено Hill climbing алгоритм; у [7, 18, 20, 25] розглянуто Local Search Algorithm [7, 18, 20, 25]; у роботах [17, 26 – 28] та [20, 21] вивчено simulated annealing; роботи [22–24] присвячено Genetic Algorithm і т.д.

Ці алгоритми застосовують різні функції вартості. Зокрема, найбільш дослідженою та поширеною є функція вартості Кларка (Clark’s cost functions). Ця функція based on Walsh-Hadamard Spectra (WHS). Вперше її було запропоновано в [26]. Дослідження її поведінки та певну модифікацію виконано в [20, 21].

В роботах [18, 29] Picek та іншими авторами було запропоновано нову функцію вартості. Picek's cost functions для деяких алгоритмів виявилася більш ефективнішою у порівнянні із WHS.

В [18, 25] Freyre-Echevarría та іншими було запропоновано нову функцію WCF (Cost Function of the content of the Walsh-Hadamard spectrum). Як виявилося, вона дозволяє як найшвидше сформувати випадкові бієктивні 8-бітні S-boxes. Кращим алгоритмом пошуку виявився Hill climbing. Наприклад, для генерації S-boxes із нелінійністю 104 йому необхідно в середньому понад 65 тисяч ітерацій. Це кращий відомий результат. Далі показано, що запропонована в цій статті функція вартості дозволяє зменшити кількість ітерацій Hill climbing алгоритму до 50 тисяч.

Передумова

За визначенням S-box є нелінійною підстановкою $S:\{0,1\}^n \to \{0,1\}^m$, яку зазвичай подають у вигляді координатних булевих функцій $F(x)=(f_1, f_2,..., f_m)$ [10, 30]:

$$f_1(x_1, x_2,..., x_n) = y_1,$$
$$f_2(x_1, x_2,..., x_n) = y_2,$$
$$...$$
$$f_m(x_1, x_2,..., x_n) = y_m.$$

В цій роботі розглядаються 8-бітні бієктивні підстановки, тобто $n=m=8$.

Основним криптографічним показником S-box є нелінійність $N(S)$, яку розраховують за формулою [30]:

$$N(S) = \min_{v\in\{0,1\}^m\setminus\{0\}^m}\{N(v\cdot F(x))\} = \frac{1}{2}(2^8 - WHT_{\max}), \quad (1)$$

де

$$WHT_{\max} = \max_{v,u\in\{0,1\}^m\setminus\{0\}^m}\left|WHT(v\cdot F(x),u)\right|,$$
$$WHT(f(x),u) = \sum_{x\in\{0,1\}^n}(-1)^{f(x)\oplus u\cdot x}. \quad 2)$$

Таким чином, нелінійність $N(S)$ визначається через перетворення Уолша – Адамара $WHT(f(x),u)$ булевої функції $f(x)=v\cdot F(x)$. Саме коефіцієнти $WHT(f(x),u)$ визначають $N(S)$. Тому функції вартості повинні враховувати значення $WHT(f(x),u)$ з метою максимізації $N(S)$ з (1), тобто максимізації мінімуму нелінійності за всіма булевими функціями $v\cdot F(x)$.

Для пошуку випадкових S-boxes на сьогодні використовують декілька функцій вартості. Перша та найбільш досліджена функція вартості WHS запропонована у [31], вона базується на врахуванні коефіцієнтів Уолша – Адамара:

$$WHS = \sum_{v\in\{0,1\}^m}\sum_{u\in\{0,1\}^n}\left|\,|WHT(v\cdot F(x),u)| - X\right|^R, \quad (3)$$

де X и R – параметри функції, які потрібно підібрати для мінімізації ітерацій пошуку.

Функція WHS використовувалася в багатьох пов'язаних роботах. Наприклад, в одній з останніх публікацій показано, що із її використанням вдається формувати 8-бітну бієктивну підстановку із $N(S)=104$ [25]. Але складність такого пошуку занадто велика. В середньому генерація вимагає близько 3,8 мільйонів ітерацій. При цьому використовувався варіант генетичного алгоритму пошуку.

В роботі [29] Picek та іншими авторами запропоновано іншу функцію. Вона заснована на врахуванні лише позицій ненульових коефіцієнтів $WHT(f(x),u)$. Функція вартості обраховується за формулою

$$PCF = \sum_{i=1}^{N} 2^{-i} H(S)_{k-i}, \tag{4}$$

де $H(S)$ – вектор значень $|WHT(v \cdot F(x), u)|$, в якому на i-й позиції вказано число коефіцієнтів с значеннями $|4i|$, k – максимальний номер позиції з ненульовим значенням.

В роботі [29] проведено низку експериментів, які показали, що функція вартості (4) є значно ефективнішою за (3). З тим же алгоритмом пошуку вона вимагає лише 167 451 ітерацій для генерації S-boxes із $N(S) = 104$.

Останній варіант функції вартості WCF було запропоновано в [18, 25]. Вона обчислюється за формулою:

$$WCF = \sum_{v \in \{0,1\}^m} \sum_{u \in \{0,1\}^n} \prod_{z \in C} \| WHT(v \cdot F(x), u) | - z |, \tag{5}$$

где $C = \{0, 4, ..., 32\}$.

За результатами експериментів в [18, 25] ця функція вартості виявилася найбільш ефективною. Зокрема, в поєднанні із Hill climbing алгоритмом вона вимагає лише біля 65 тисяч ітерацій для генерації випадкових S-boxes із $N(S) = 104$. Це кращий із відомих на сьогодні результат.

Запропонована функція вартості WCFS

Для обґрунтування нової функції вартості розглянемо розподіл значень (2) для випадково сгенерованого S-box. Приклад такого розподілу наведено на рис. 1.

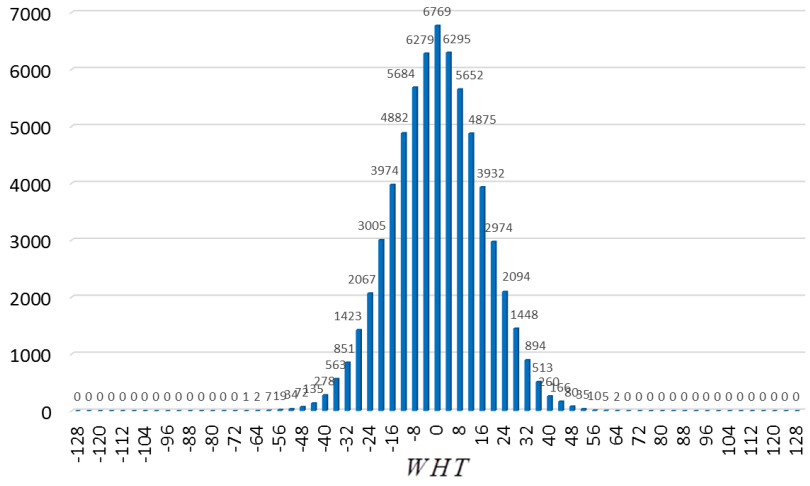

Рис. 1. Приклад розподілу значень спектральних коефіцієнтів Уолша – Адамара для випадково сформованого бієктивного S-блоку

За нашими дослідженнями виявилося, що для підрахунку функції вартості переважним є спосіб урахування окремих коефіцієнтів (2). Оскільки нелінійність (1) обраховується за максимальними значеннями $|WHT(v \cdot F(x), u)|$, то пріоритетними повинні бути крайні значення спектру з рис. 1. По мірі наближення коефіцієнтів $|WHT(v \cdot F(x), u)|$ до центральної частини розподілу їх вплив на функцію вартості повинен значно зменшуватись.

Таким чином, на кожній ітерації алгоритму оптимізації нас у першу чергу цікавить зменшення крайніх коефіцієнтів. Групування інших коефіцієнтів до центру розподілу буде полегшувати підвищення нелінійності на наступних ітераціях алгоритму. Отже, необхідно враховувати кожний спектральний коефіцієнт з деяким ваговим коефіцієнтом. Чим ближче коефіцієнт до нуля – тим нижче його вага. У функції (5) це реалізовано за рахунок добутку, при-

чому, спектральні коефіцієнти від нуля до 32 включно взагалі не ураховуються (внаслідок множення на нуль), а значення добутку зростає пропорційно позиції спектрального коефіцієнта у розподілу.

Схоже вагове урахування реалізовано у функції (3). Наприклад, при $R=12$ та $X=0$ коефіцієнти спектру Уолша – Адамара, які знаходяться на крайніх позиціях, мають більш вагомий внесок у значення функції вартості, ніж коефіцієнти, які знаходяться ближчі до центру у розподілу.

Базуючись на отриманих результатах, можна зробити висновок, що більш швидке знаходження S-блоку відбувається при урахуванні лише деяких крайніх значень спектру Уолша – Адамара. З цього приводу нами пропонується нова цільова функція WCFS, яка є деяким гібридним рішенням між функціями WHS та WCF. В ній враховуються лише коефіцієнти, які більш деякого значення X, що відповідає виразу $|WHT|>X$. Також доцільним є зменшення значень, що враховуються, на X та у 4 рази, що призведе до постійного та рівномірного порядку зростання (1,2,3,…) значень спектральних коефіцієнтів, що враховуються. Вагове урахування позицій спектральних коефіцієнтів у розподілі реалізуємо у вигляді возведення у деяку ступінь R отриманої позиції спектральних коефіцієнтів. Нижче наведено формальний опис запропонованої цільової функції:

$$WCFS=\sum_{\substack{b=1\\ |WHT[b,i]|>X}}^{255}\sum_{i=0}^{255}\left(\frac{|WHT[b,i]|-X}{4}\right)^{R}. \tag{6}$$

Параметри X та R повинні бути підібрані з метою підвищення ефективності генерації підстановок.

Тестування та оптимізація параметрів нової функції вартості

Для підбору оптимального значення параметрів X та R розглянемо, як буде змінюватися вплив запропонованої функції (6) на швидкість пошуку. Швидкість будемо вимірювати в кількості ітерацій.

У якості методу пошуку будемо використовувати Hill climbing алгоритм. Псевдокод цього алгоритму наведено у додатку роботи [18]. Пошук починається із випадково сформованої бієктивної підстановки S_0. Критерієм зупинки алгоритму є досягнення загальної кількості ітерацій N_1. Додатково введено ще два критерії зупинки алгоритму, а саме:

- досягнення максимальної кількості N_2 поспіль виконаних ітерацій, при яких не знайдено жодного покращення функції вартості;
- досягнення цільового значення нелінійності підстановки N_3, розрахованого за формулою (1).

Отже наш варіант Hill climbing алгоритму подаємо у наступному вигляді.

Pseudo-Code of the Hill Climbing Algorithm

Вхід: S_0, N_1, N_2, N_3.

$S \leftarrow S_0$, $n \leftarrow 0$;

While ($N_1 > 0$) and ($n < N_2$) and ($N(S) < N_3$) do:

$S' \leftarrow S$;

Select at random two different positions i and j and swap the outputs on S' corresponding to i and j;

if $WCFS(S') \le WCFS(S)$ then

$S \leftarrow S'$, $n \leftarrow 0$;

else

$n \leftarrow n+1$;

$N_1 \leftarrow N_1 - 1$;

Return S.

Таким чином, на кожній ітерації алгоритму модифікується поточне значення підстановки S, в результаті отримуємо S-box S'. Далі розраховуємо значення функції вартості $WCFS(S')$ за формулою (6) та порівнюємо його із значенням $WCFS(S)$ для поточного S-box S. Якщо значення функції вартості не збільшилося, тоді S' приймається за кращий поточний результат.

Початкова підстановка S_0 формувалася випадковим чином.

Параметри зупинки алгоритму обрано наступні:

$N_1 = 1\,000\,000$,

$N_2 = 100\,000$,

$N_3 = 104$.

При тестуванні параметри X та R змінювались у діапазоні:

$-32 \geq X \geq 32$ з кроком 4;

$5 \geq R \geq 18$ з кроком 1.

Зауважимо, що при $X = 48$ та $N(S) = 104$ маємо значення функції $WCFS = 0$.

Для кожного параметру X та R, з метою усереднення результатів, проводилось 100 запусків Hill climbing алгоритму.

Результати досліджень усередненої кількості ітерацій пошуку наведено у табл. 1 та візуалізовано на рис. 1. Крім наведених даних для кожного вдалого запуску (тобто коли був знайдений S-блок з нелінійністю 104) також фіксуємо кількість ітерації алгоритму пошуку, які було виконано для досягнення нелінійності 100 та 102. Усереднені кількості ітерації наведено у табл. 2 і 3 та візуалізовано на рис. 2 і 3 відповідно.

Символом «–» у табл. 1 – 3 позначено випадки, коли алгоритмом пошуку було знайдено цільовий S-блок менш ніж у 50 % випробувань та отримані результати не можуть характеризувати необхідну для пошуку кількість ітерації. У табл. 4 наведено відсоток окремих запусків, за результатом яких було знайдено бієктивний S-блок з нелінійністю 104.

Таблиця 1

Середньоарифметична кількість ітерацій, які було виконано
до знаходження бієктивного S-блоку з нелінійністю 104 при використанні функції WCFS

X	R													
	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
-32	–	–	–	–	–	231762	155358	119041	103349	75955	69554	61694	54840	55188
-28	–	–	–	–	–	157544	133267	108750	86979	70292	61496	58275	56007	57114
-24	–	–	–	–	184299	174331	113341	90294	74057	62517	58270	61259	**52476**	54384
-20	–	–	–	–	162916	126207	97274	74575	70495	61759	53146	**52547**	53902	**53678**
-16	–	–	–	–	151538	99170	84834	68011	61772	51375	56735	54452	53260	56557
-12	–	–	–	183600	123348	93124	67811	58808	**54800**	50732	**50934**	53289	58196	56962
-8	–	–	197322	157799	104289	77086	60823	53292	56088	**49399**	58393	59345	62675	60460
-4	–	–	179263	119941	92926	70066	52260	**50438**	56295	53192	55211	66722	69403	74278
0	–	–	141912	96838	79239	57095	53509	54924	55990	58531	62098	66465	84083	84002
4	–	184668	111097	77850	65663	56931	**50238**	56603	57259	63593	68088	88000	89250	109897
8	–	148482	90091	63921	56935	**50798**	56814	56123	65781	76237	79544	100726	120771	132152
12	171440	103464	66827	**53062**	**52809**	58960	60873	62813	75322	84133	116092	121023	145128	150432
16	131424	78297	63295	57519	55367	60144	71509	87545	97420	119710	138424	–	–	–
20	86714	60656	**51971**	56974	65052	78869	92354	115900	123319	123864	–	–	–	–
24	65010	**54329**	62361	67914	86958	105767	129225	150163	–	–	–	–	–	–
28	**56280**	63986	70787	95384	111413	146470	–	–	–	–	–	–	–	–
32	67049	80236	108002	134489	–	–	–	–	–	–	–	–	–	–

Таблиця 2

Середньоарифметична кількість ітерацій, які було виконано
до знаходження бієктивного S-блоку з нелінійністю 102 при використанні функції WCFS

X	*R*													
	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
-32	–	–	–	–	–	1985	1726	1476	1407	1195	1092	1075	1019	965
-28	–	–	–	–	–	1805	1574	1388	1199	1137	1064	1018	984	959
-24	–	–	–	–	2342	1794	1482	1267	1188	1096	1072	1036	996	924
-20	–	–	–	–	1919	1607	1333	1165	1130	1032	1007	957	1002	952
-16	–	–	–	–	1867	1475	1262	1162	1092	992	967	953	**905**	**894**
-12	–	–	–	2235	1557	1291	1195	1083	1024	1025	972	991	924	970
-8	–	–	2446	1672	1353	1257	1107	1008	998	949	**896**	**878**	936	958
-4	–	–	2023	1478	1271	1088	1072	**939**	998	964	949	985	974	979
0	–	–	1732	1384	1181	1049	981	964	**922**	**940**	972	983	991	1008
4	–	2127	1484	1193	1067	976	**943**	944	937	948	947	1010	1067	1081
8	–	1785	1346	1137	1024	952	965	960	964	967	1018	1070	1128	1091
12	2274	1336	1172	1037	968	**932**	953	972	1009	1050	1049	1046	1119	1353
16	1703	1289	1101	985	**952**	933	1012	996	1114	1022	1116	–	–	–
20	1446	1096	972	973	997	995	969	1015	1094	1196	–	–	–	–
24	1219	1063	923	**913**	971	1005	1053	1158	–	–	–	–	–	–
28	1020	**945**	**915**	966	1015	1204	–	–	–	–	–	–	–	–
32	**974**	961	1001	1151	–	–	–	–	–	–	–	–	–	–

Таблиця 3

Середньоарифметична кількість ітерацій, які було виконано
до знаходження бієктивного S-блоку з нелінійністю 100 при використанні функції WCFS

X	*R*													
	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
-32	–	–	–	–	–	276	231	208	189	179	178	166	151	157
-28	–	–	–	–	–	246	203	194	191	170	166	161	153	154
-24	–	–	–	–	287	230	213	189	181	169	162	153	153	138
-20	–	–	–	–	252	218	202	186	174	163	166	154	148	150
-16	–	–	–	–	232	200	202	170	164	157	164	161	148	149
-12	–	–	–	250	215	199	183	183	168	153	152	148	147	140
-8	–	–	270	233	207	184	173	161	154	151	146	145	145	147
-4	–	–	255	232	196	175	160	165	143	142	147	143	144	143
0	–	–	235	200	183	159	161	156	155	148	148	149	142	142
4	–	259	232	191	173	161	153	149	145	152	142	145	147	149
8	–	238	201	171	164	151	152	144	150	139	143	146	138	139
12	267	220	193	157	153	145	146	148	145	146	147	151	146	143
16	250	191	176	153	151	134	143	150	140	141	158	–	–	–
20	207	182	163	152	146	144	145	144	144	145	–	–	–	–
24	185	168	150	147	149	140	147	145	–	–	–	–	–	–
28	162	154	143	145	154	141	–	–	–	–	–	–	–	–
32	153	148	145	136	–	–	–	–	–	–	–	–	–	–

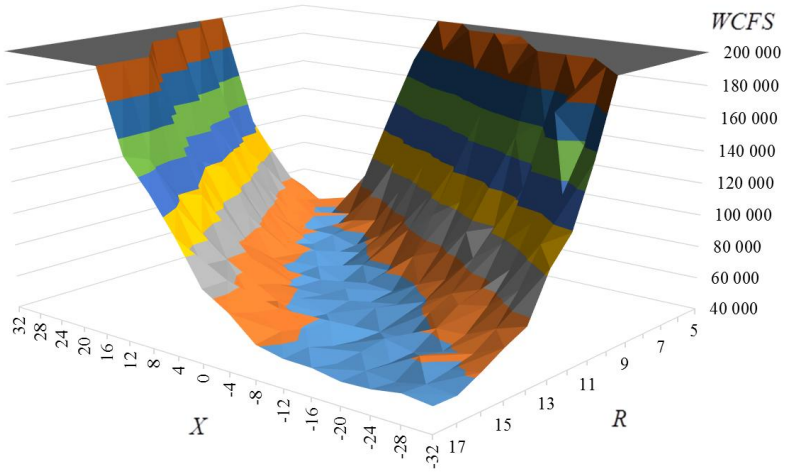

Рис. 2. Середньостатистична кількість ітерацій, які було виконано до знаходження бієктивного S-блоку з нелінійністю 104 при використанні функції $WCFS$ при різних параметрах X та R

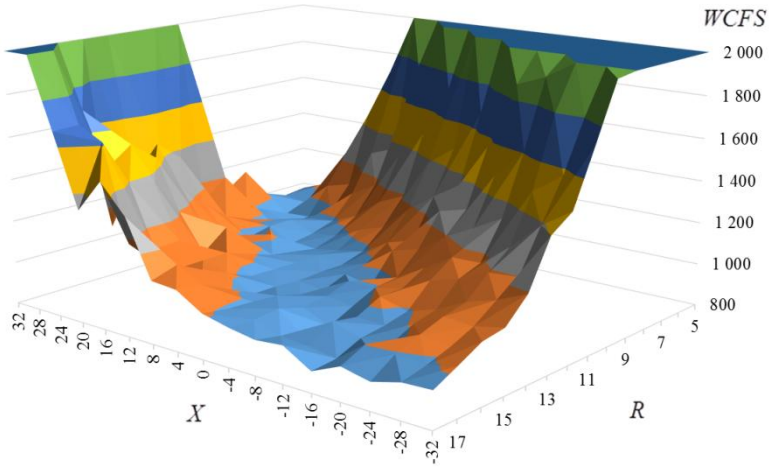

Рис. 3. Середньостатистична кількість ітерацій, які було виконано до знаходження бієктивного S-блоку з нелінійністю 102 при використанні функції $WCFS$ при різних параметрах X та R

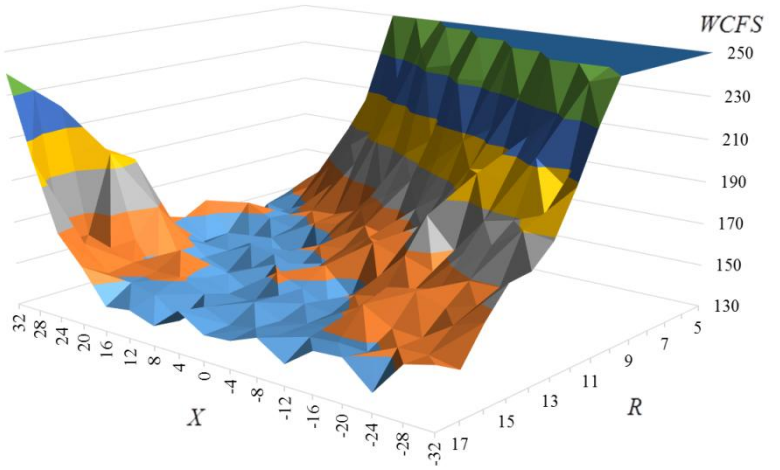

Рис. 4. Середньостатистична кількість ітерацій, які було виконано до знаходження бієктивного S-блоку з нелінійністю 100 при використанні функції $WCFS$ при різних параметрах X та R

Таблиця 4

Кількість знайдених (під час досліджень) бієктивних S-блоків з нелінійністю 104 при використанні функції WCFS

X	R													
	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
-32	1	0	2	10	48	61	86	95	98	99	100	99	100	100
-28	1	1	3	13	35	70	94	99	96	99	99	100	100	100
-24	1	0	7	26	51	87	94	95	100	100	100	100	100	100
-20	0	0	8	36	73	89	98	98	99	100	99	100	100	100
-16	0	3	18	47	84	92	99	100	99	100	100	100	100	100
-12	1	11	34	73	92	98	100	100	100	100	100	99	100	100
-8	2	9	50	85	99	98	100	99	100	100	99	100	99	99
-4	2	34	67	91	100	99	100	100	100	99	100	100	99	100
0	8	47	90	97	98	100	100	99	100	100	100	100	99	95
4	14	56	93	99	100	100	100	100	99	99	99	99	90	92
8	41	85	99	99	100	99	100	99	98	100	97	97	90	80
12	59	97	99	99	100	100	100	100	100	96	88	68	73	55
16	87	97	100	100	99	100	99	98	90	83	72	47	27	19
20	97	100	100	100	99	98	96	82	64	53	35	25	9	7
24	99	100	100	98	98	92	75	60	40	20	17	14	3	2
28	100	99	99	94	84	58	32	13	11	5	1	2	0	1
32	100	98	94	61	35	22	5	3	1	2	3	1	1	3

Обговорення результатів

Отримані результати тестування демонструють високу ефективність запропонованої функції вартості WCFS. Зокрема, для функції (6) існує великий діапазон значень параметрів X та R, для яких генерація підстановок є дуже швидкою. Наприклад, для параметрів $R=14$ та $X=$-8 середньостатистична кількість ітерацій алгоритму пошуку бієктивних S-блоків з нелінійністю 104 склала 49 399 ітерацій. Це суттєво менш ніж при використанні функції WCF (понад 65 тисяч операцій) з [18, 25]. Крім того, як бачимо з наведених результатів, майже при будь-якому значенні X можливо підібрати вагове значення R з дуже швидким знаходженням S-блоку.

Синім кольором у табл. 1 позначено випадки коли середня кількість ітерації скла менш за 60 000, а темно-синім – менш за 55 000 ітерації. Область мінімальних значень середньої кількості ітерації відповідає співвідношенню параметрів X та R, які емпірично можна визначити формулою

$$X = 48 - 4 \cdot R. \tag{7}$$

Значення параметрів X та R, які відповідають співвідношенню (7), у табл. 1 обведені рамкою. Експериментально встановлені мінімальні значення середньої кількості ітерації для кожного R позначено жирним шрифтом. Як бачимо, позиції, які відповідають співвідношенню (7), та знайдені мінімальні значення, у більшості випадках співпадають або знаходяться дуже близько один до одного (у межах обчислювальної похибки). Отже формулу (7) можна використовувати для швидкого підбору найбільш придатних співвідношень X та R.

Слід зазначити, що середня кількість ітерацій для параметрів X та R, яка відповідає співвідношенню (7), також не є однорядною та має мінімальне значення в області $R=12\pm3$. При зменшені або збільшенні значення R від області $R=12\pm3$ середня кількість ітерації починає зростати.

Для мінімізації обчислювальних ресурсів при розрахунку функції вартості необхідним є зменшення значення R. В цій роботі проведено тестування середнього часу обчислення функції *WCFS* в залежності від обраного значення R (при $X=48-4\cdot R$). Отримані результати наведено у табл. 5. Розрахунок виконувався на персональному комп'ютері Intel Core i5-3210M CPU 2.50GHz під керуванням 64-розрядної операційної системи

Windows 7. Компіляція коду, написаного на C++, виконувалась за допомогою Microsoft Visual Studio Community 2022 (64-розрядна версія) у Release конфігурації.

Таблиця 5

Середній час обчислення цільової функції *WCFS* при різних параметрах

Параметр	**Час виконання обчислення функції *WCFS*, с**
$R=5, X=28$	$1{,}06\cdot10^{-3}$
$R=6, X=24$	$1{,}09\cdot10^{-3}$
$R=7, X=20$	$1{,}13\cdot10^{-3}$
$R=8, X=16$	$1{,}22\cdot10^{-3}$
$R=9, X=12$	$1{,}24\cdot10^{-3}$
$R=10, X=8$	$1{,}32\cdot10^{-3}$
$R=11, X=4$	$1{,}30\cdot10^{-3}$
$R=12, X=0$	$1{,}28\cdot10^{-3}$
$R=13, X=-4$	$1{,}30\cdot10^{-3}$
$R=14, X=-8$	$1{,}32\cdot10^{-3}$
$R=15, X=-12$	$1{,}37\cdot10^{-3}$
$R=16, X=-16$	$1{,}38\cdot10^{-3}$
$R=17, X=-20$	$1{,}39\cdot10^{-3}$
$R=18, X=-24$	$1{,}41\cdot10^{-3}$

Наші дослідження показують, що при дотриманні вимоги (7) ймовірність знаходження бієктивного S-блоку з $N(S)=104$, при обраних параметрах алгоритму пошуку, близька до 1 (див. табл.4). Зі збільшенням значення R відповідно збільшується діапазон значень X, при яких ймовірність і середня кількість ітерації знаходяться у своїх кращих значень для обраних R.

В табл. 2 та 3 наведено результати генерації S-блоків з $N(S)=102$ та $N(S)=100$ відповідно.

У табл. 2 синім кольором позначено випадки, коли середня кількість ітерації скла менш за 1 000, а темно-синім – менш за 950 ітерації. Спостерігається схожий характер розподілу з областю мінімальної кількості ітерацій до розподілу з $N(S)=104$. Однак область мінімальних значень буде відповідати співвідношенню

$$X=52-4\cdot R. \tag{8}$$

Чарунки, які відповідають цьому співвідношенню, позначені рамкою. Мінімальні значення для кожного R виділені жирним шрифтом. Також спостерігається добрий збіг мінімальних значень з емпіричним співвідношенням (8).

Для генерації бієктивних S-блоків з $N(S)=100$ (див. табл. 3) також можна виділити область мінімальних значень які будуть відповідати емпіричному співвідношенню

$$X=56-4\cdot R. \tag{9}$$

Узагальнюючи область мінімальної кількості ітерації для різних значень нелінійності найкраще співвідношення параметрів X та R буде відповідати емпіричній залежності:

$$X=2\cdot(128-N(S))-4\cdot R \tag{10}$$

та імовірно найкраще обрати

$$R = \frac{128 - N(S)}{2}. \quad (11)$$

При обранні рекомендованих формулами (10) та (11) параметрів функції *WCFS* середньо-арифметична кількість ітерацій до знаходження цільового S-блоку буде складати:

- близько 52 200 ітерацій при $N(S) = 104$;
- близько 950 ітерацій при $N(S) = 102$;
- близько 148 ітерацій при $N(S) = 100$.

Кращий результат генерації S-блоків з $N(S) = 104$ дає функція *WCFS* з параметрами $X = -8$ and $R = 14$. При цьому Hill climbing алгоритму необхідно в середньому 49399 ітерацій.

Для порівняння отриманих результатів в табл. 6 наведено кращі відомі результати з генерації нелінійних підстановок з $N(S) = 104$.

Таблиця 6

Порівняння отриманих результатів з генерації нелінійних підстановок з $N(S) = 104$

Parameters	[31]	[29]	[18]	[25]	Our work
Generation Method	«Genetic and Tree»	«Genetic and Tree»	Hill climbing	Hill climbing	Hill climbing
Cost function, parameters	WHS, $X = 21$ and $R = 7$	PCF, $N_p = 10$	WCF	WCF	WCFS, $X = -8$ and $R = 14$
Average number of iterations	3 239 000	167 451	70 596	65 933	49 399

Як бачимо, запропонована функція вартості в наших експериментах показує значне покращення. Зокрема, середню кількість ітерацій зменшено на понад 20 % в порівнянні з кращим відомим результатом.

Висновки

Запропоновано та досліджено нову функцію вартості WCFS. Використання WCFS дозволяє підвищити ефективність евристичного пошуку нелінійних підстановок. Крім того, частка успішних запусків алгоритму генерації досягає 100 %. В наших тестуваннях застосовувався Hill climbing алгоритм. В порівнянні з кращим відомим результатом генерації S-блоків з $N(S) = 104$ нам вдалося більше ніж на 20 % скоротити кількість ітерацій.

Проведені тестування дозволили виділити область параметрів функції вартості, для якої спостерігається найменша кількість ітерацій. Введено емпіричну залежність між параметрами X та R. Результати тестування майже повністю збігаються із цією емпіричною залежністю. Отже маємо змогу швидко підбирати параметри нової функції вартості WCFS для генерації S-блоків.

Використання функції WCFS в алгоритмі Hill climbing дало кращі результати в порівнянні з функціями вартості *WCF* або *WHS*. Отримано такі результати:

- для знаходження бієктивного S-блоку з $N(S) = 104$ алгоритм генерації потребує в середньому 49 399 ітерацій (при $R = 14$ та $X = -8$), що на 23 % менш ніж кращий відомий результат (65 933 ітерацій);
- для знаходження бієктивного S-блоку з $N(S) = 102$ алгоритм генерації потребує в середньому 878 ітерацій (при $R = 16$ та $X = -8$);
- для знаходження бієктивного S-блоку з $N(S) = 100$ алгоритм генерації потребує в середньому 134 ітерації (при $R = 10$ та $X = 16$).

Список літератури:

1. Menezes A.J. et al. Handbook of Applied Cryptography. CRC Press, 2018.
2. Delfs H., Knebl H. Introduction to Cryptography. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.
3. Kuznetsov A.A. et al. Stream Ciphers in Modern Real-time IT Systems: Analysis, Design and Comparative Studies. Springer International Publishing, 2022. XVI. 611 p.
4. Daemen J., Rijmen V. Specification of Rijndael // The Design of Rijndael: The Advanced Encryption Standard (AES) / ed. Daemen J., Rijmen V. Berlin, Heidelberg: Springer, 2020. P. 31–51.
5. Daemen J., Rijmen V. AES proposal: rijndael. 1999.
6. Shannon C.E. Communication theory of secrecy systems // The Bell System Technical Journal. 1949. Vol. 28, № 4. P. 656–715.
7. Freyre Echevarría A. Evolución híbrida de s-cajas no lineales resistentes a ataques de potencia. 2020.
8. Gorbenko I. et al. Random S-Boxes Generation Methods for Symmetric Cryptography // 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON). 2019. P. 947–950.
9. Moskovchenko I. et al. HEURISTIC METHODS FOR THE DESIGN OF CRYPTOGRAPHIC BOOLEAN FUNCTIONS: 3 // International Journal of Computing. 2019. Vol. 18, № 3. P. 265–277.
10. Cusick T., Stănică P. Cryptographic Boolean Functions and Applications: Second edition // Cryptographic Boolean Functions and Applications: Second Edition. 2017. P. 2751 p.
11. Bard G.V. Algebraic Cryptanalysis. Boston, MA: Springer US, 2009.
12. Courtois N.T., Bard G.V. Algebraic Cryptanalysis of the Data Encryption Standard // Cryptography and Coding / ed. Galbraith S.D. Berlin, Heidelberg: Springer, 2007. P. 152–169.
13. Courtois N.T., Pieprzyk J. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations // Advances in Cryptology — ASIACRYPT 2002 / ed. Zheng Y. Berlin, Heidelberg: Springer, 2002. P. 267–287.
14. Technology N.I. of S. and. Advanced Encryption Standard (AES): Federal Information Processing Standard (FIPS) 197. U.S. Department of Commerce, 2001.
15. Daemen J., Rijmen V. Rijndael/AES // Encyclopedia of Cryptography and Security / ed. van Tilborg H.C.A. Boston, MA: Springer US, 2005. P. 520–524.
16. Burnett L.D. Heuristic Optimization of Boolean Functions and Substitution Boxes for Cryptography: phd. Queensland University of Technology, 2005.
17. Ivanov G., Nikolov N., Nikova S. Cryptographically Strong S-Boxes Generated by Modified Immune Algorithm // Cryptography and Information Security in the Balkans / ed. Pasalic E., Knudsen L.R. Cham: Springer International Publishing, 2016. P. 31–42.
18. Freyre-Echevarría A. et al. An External Parameter Independent Novel Cost Function for Evolving Bijective Substitution-Boxes: 11 // Symmetry. Multidisciplinary Digital Publishing Institute. 2020. Vol. 12, № 11. P. 1896.
19. Freyre-Echevarría A. et al. Evolving Nonlinear S-Boxes With Improved Theoretical Resilience to Power Attacks // IEEE Access. 2020. Vol. 8. P. 202728–202737.
20. Kuznetsov A. et al. Optimizing the Local Search Algorithm for Generating S-Boxes // 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S T). 2021. P. 458–464.
21. Kuznetsov A. et al. WHS Cost Function for Generating S-boxes // 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S T). 2021. P. 434–438.
22. Ivanov G., Nikolov N., Nikova S. Reversed genetic algorithms for generation of bijective s-boxes with good cryptographic properties // Cryptogr. Commun. 2016. Vol. 8, № 2. P. 247–276.
23. Kapuściński T., Nowicki R.K., Napoli C. Application of Genetic Algorithms in the Construction of Invertible Substitution Boxes // Artificial Intelligence and Soft Computing / ed. Rutkowski L. et al. Cham: Springer International Publishing, 2016. P. 380–391.
24. Mariot L., Leporati A. Heuristic Search by Particle Swarm Optimization of Boolean Functions for Cryptographic Applications // Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation. New York, NY, USA: Association for Computing Machinery. 2015. P. 1425–1426.
25. Freyre Echevarría A., Martínez Díaz I. A new cost function to improve nonlinearity of bijective S-boxes. 2020.
26. Clark J.A., Jacob J.L., Stepney S. The design of S-boxes by simulated annealing // New Gener Comput. 2005. Vol. 23, № 3. P. 219–231.
27. McLaughlin J. Applications of search techniques to cryptanalysis and the construction of cipher components: phd. University of York, 2012.
28. Wang J. et al. Construction Method and Performance Analysis of Chaotic S-Box Based on a Memorable Simulated Annealing Algorithm: 12 // Symmetry. Multidisciplinary Digital Publishing Institute, 2020. Vol. 12, № 12. P. 2115.
29. Picek S., Cupic M., Rotim L. A New Cost Function for Evolution of S-Boxes // Evolutionary Computation. 2016. Vol. 24, № 4. P. 695–718.
30. Carlet C. Vectorial Boolean functions for cryptography // Boolean Models and Methods in Mathematics, Computer Science, and Engineering. 2006.
31. Tesar P. A New Method for Generating High Non-linearity S-Boxes. Společnost pro radioelektronické inženýrství, 2010.

Надійшла до редколегії 11.05.2022

Відомості про авторів:

Кузнецов Олександр Олександрович – д-р техн. наук, професор, Харківський національний університет імені В.Н. Каразіна, професор кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; Україна; e-mail: kuznetsov@karazin.ua; ORCID: https://orcid.org/0000-0003-2331-6326

Полуяненко Микола Олександрович – канд. техн. наук, Харківський національний університет імені В.Н. Каразіна, доцент кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; Україна; e-mail: nlfsr01@gmail.com; ORCID: https://orcid.org/0000-0001-9386-2547

Кандій Сергій Олегович – АТ «Інститут інформаційних технологій», технік-конструктор, Україна; e-mail: sergey.kandy@gmail.com, ORCID: https://orcid.org/0000-0003-0552-8341

Пелюх Олександр Іванович – Харківський національний університет імені В.Н. Каразіна, студент кафедри безпеки інформаційних систем і технологій, факультет комп'ютерних наук; Україна; e-mail: oleksandrpelyukh@gmail.com; ORCID: https://orcid.org/0000-0003-0507-0262