К.: МК-Пресс, 2006. – 288 с. ***2.*** *Грибунин В.Г., Оков И.Н., Туринцев И.В. Цифровая стеганография. – М.: СОЛОН-Пресс, 2002. – 272 с.* ***3.*** *Аграновский А.В., Девянин П.Н., Хади Р.А. и др. Основы компьютерной стеганографии. – М.: Радио и связь, 2003. – 151 с.* ***4.*** *Eyadat M., Vasikarla S. Performance evaluation of an incorporated DCT Block-Based Watermarking algorithm with Human Visual system Model // Pattern Recognition Journal. – 2005. – V. 26. – P. 1405-1411.* ***5.*** *Золотарев В.В., Овечкин Г.В. Помехоустойчивое кодирование. Методы и алгоритмы: Справочник / Под ред. чл.-кор. РАН Ю.Б. Зубарева. – М.: Горячая линия-Телеком, 2004. – 126 с.*

УДК 681.3.06

МЕТОД ПОБУДОВИ ВИСОКОШВИДКІСНОГО ПРОГРАМНО-ОРІЄНТОВАНОГО ПОТОКОВОГО ШИФРУ

Олександр Дирда
Державна служба спеціального зв'язку та захисту інформації України

Анотація: **Запропонована криптографічна схема нового программно-орієнтованого потокового шифру гамування з умовною назвою WSC, який базується на лінійному рекурентному регістрі довжини 32 над скінченним полем $GF(2^{32})$ та схемі ускладнення на основі чотирьох нелінійних регістрів зсуву над скінченним полем $GF(2^8)$. У схемі ускладнення пропонується використати 8×8 S-блоки з властивістю кореляційної імунності всіх координатних функцій.**
Summary: **This article proposes description of cryptographic scheme of new software-oriented stream cipher called WSC. It based on two items: linear feedback shift register length of 16 over the Galois field GF(2^{32}) and complication scheme which based on four nonlinear shift registers over the Galois field GF(2^8). 8x8 S-boxes with correlation immunity property of all coordinate functions are proposed to be used in this complication scheme.**
Ключові слова: **Потоковий шифр, генератор гами, лінійний рекурентний регістр, скінченне поле, S-блок, кореляційна імунність.**

I Вступ

Одним із важливих класів симетричних криптографічних алгоритмів є потокові шифри гамування. Вони містять у своєму складі генератор гами (ключової послідовності), що виробляє псевдовипадкову послідовність бітів, яка додається за модулем два до послідовності бітів відкритого тексту. Секретний ключ, як правило, використовується для ініціалізації початкового стану генератора гами.

До недавнього часу на практиці використовувались переважно біт-орієнтовані потокові шифри гамування, які містили один або декілька лінійних рекурентних регістрів (ЛРР) над скінченним полем $GF(2)$ та фільтруючі або комбінуючі булеві функції. Принципи синтезу та аналізу таких схем досить докладно наведені в монографії [1]. Математичні основи синтезу ЛРР над полем $GF(2)$ викладені в [2].

Прикладом біт-орієнтованого потокового шифру гамування є шифр A5, який був запропонований у 1987 році і використовується для криптографічного захисту інформації в стандарті GSM [3]. Ряд прикладів потокових шифрів наведені в монографії [4]. Біт-орієнтовані потокові шифри мають швидку апаратну реалізацію, однак, їх реалізація на сучасних процесорах є повільною. Такі шифри можна вважати апаратно-орієнтованими, хоча вони мають досить ефективну реалізацію на інтегральних логічних матрицях, що програмуються.

Першим широко відомим байт-орієнтовним потоковим шифром є шифр RC4, який був розроблений Рівестом у 1987 році [5]. В останнє десятиріччя криптографами розроблені низка високошвидкісних слово-орієнтованих потокових шифрів, найбільш відомими з яких є алгоритми SEAL, WAKE, SNOW 2.0, Sober-t32. Як байт-, так і слово-орієнтовані шифри є програмно-орієнтованими, що означає можливість їх ефективної реалізації саме на універсальних процесорах.

Взагалі, слід відзначити, що розробка високошвидкісних програмно-орієнтованих потокових шифрів є актуальною задачею сучасної прикладної криптографії. Це пояснюється, по-перше, поступовим витискуванням апаратних реалізацій складних електронних схем програмно-апаратними, по-друге, постійно зростаючою швидкістю передачі інформації у сучасних телекомунікаційних мережах передачі даних, що диктує необхідність розробки високошвидкісних алгоритмів шифрування. Про це свідчать ряд міжнародних

проектів щодо створення потокових шифрів, зокрема NESSIE [6] та eSTREAM [7]. В рамках тільки цих двох проектів було запропоновано понад 40 криптографічних схем потокових шифрів, більшість з яких є саме програмно-орієнтованими. Понад 20 шифрів переважають за швидкодією алгоритм AES у режимі OFB і майже всі алгоритми є більш ефективними, ніж алгоритм ГОСТ 28147-89 у режимі гамування. Як приклад, програмна реалізація алгоритму SNOW 2.0 [8] для процесору типу Intel Pentium IV потребує усього 18 тактів для зашифрування 32-розрядного слова, отже, при тактовій частоті процесора 2 ГГц швидкість шифрування даних дорівнює приблизно 400 Мбайт/с, що у десятки разів переважає швидкодію алгоритму ГОСТ 28147-89. Варто зауважити, що алгоритм ГОСТ 28147-89 у режимах гамування та гамування зі зворотним зв'язком досить часто використовується на практиці у програмно-апаратних засобах, які призначені для криптографічного захисту конфіденційної інформації.

II Опис криптографічної схеми потокового шифру WSC

У цьому розділі статті запропоновано метод побудови програмно-орієнтованого потокового шифру гамування, який засновано на ЛРР над розширенням поля $GF(2)$ та нелінійній схемі ускладнення. Криптографічна схема потокового шифру з умовною назвою WSC містить:

- ЛРР довжини 32 над скінченним полем $GF(2^{32})$;
- вузол ускладнення (ВУ) на основі чотирьох ідентичних нелінійних регістрів Галуа довжини 4 над скінченним полем $GF(2^8)$;
- вузол накладення гами (ВНГ), який є суматором за модулем 2.

За один такт роботи шифру WSC формується 32 біти гами, які у ВНГ додаються до 32 бітів відкритого тексту.

Перші два вузла є генератором гами потокового шифру WSC. Структурна схема генератора гами зображена на рис. 1.

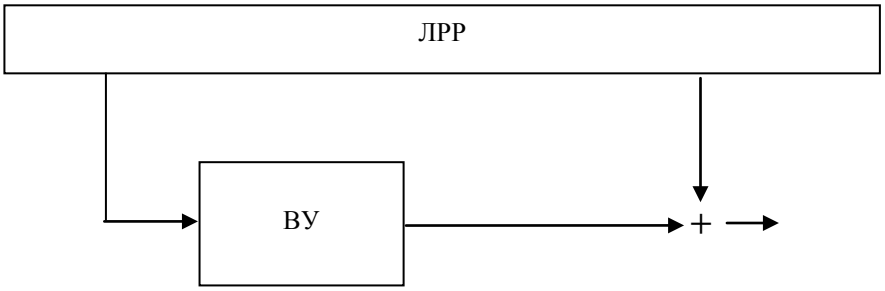

Рисунок 1 – Структурна схема генератора гами потокового шифру WSC

На рисунку 1 через $+$ позначена операція додавання за модулем 2^{32}. Інформація з ЛРР до ВУ знімається з 11-ї комірки ЛРР, а до суматора за модулем 2^{32} – з 2-ї комірки ЛРР.

Структурна схема ЛРР над полем $GF(2^{32})$ зображена на рис. 2.

Позначимо стан ЛРР в такті з номером t через $(s_{t+31},...,s_{t+1},s_t)$. Тоді стан ЛРР в такті з номером $t+1$ дорівнює $(\alpha^{-1}s_{t+21}+s_{t+6}+\alpha\, s_t, s_{t+31},...,s_{t+2},s_{t+1})$.

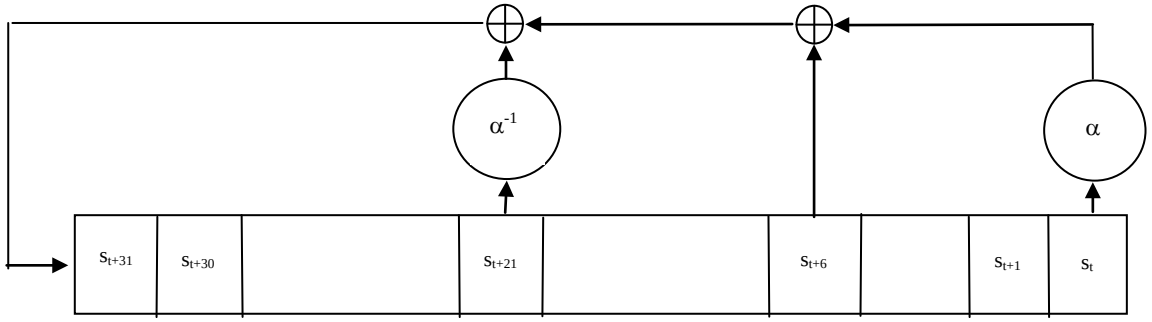

Рисунок 2 – Структурна схема ЛРР над полем $GF(2^{32})$

Структурна схема i-го нелінійного регістру Галуа над скінченним полем $GF(2^8)$, де $i = \overline{0,3}$, зображена на рис. 3. Через b_i позначено i-й байт 32-розрядного слова, яке надходить до ВУ з ЛРР, через S_i – 8×8 S-блок i-го регістру, через $\oplus$ – операцію додавання за модулем 2, через $+$ – операцію додавання за модулем 2^8. 32-розрядне слово x з виходу ВУ утворюється у результаті конкатенації чотирьох байтів x_i з виходу i-го регістру Галуа, де $i = \overline{0,3}$, тобто, $x = (x_3 \,||\, x_2 \,||\, x_1 \,||\, x_0)$.

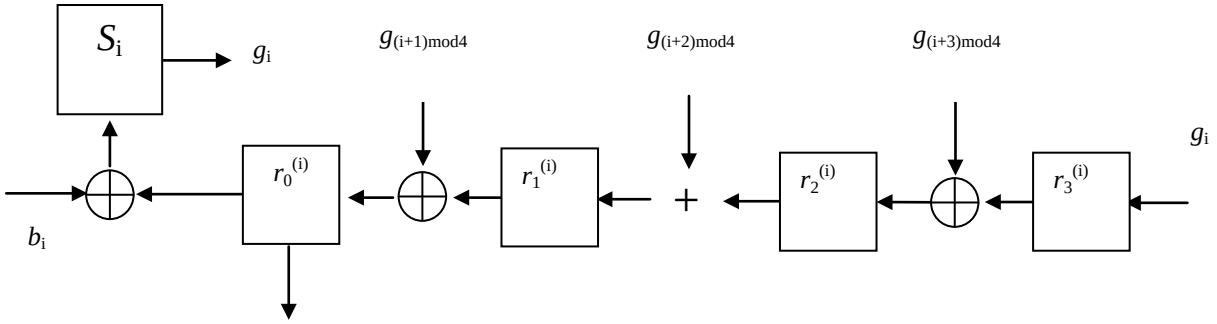

Рисунок 3 – Структурна схема i-го регістру Галуа ($i = \overline{0,3}$).

Аналітичний вираз функції переходів вузла ускладнення наведено нижче, а саме, стан чотирьох регістрів Галуа у наступному такті.

$$\left(r_1^{(0)} \oplus S_1(r_0^{(1)} \oplus b_1),\, r_2^{(0)} + S_2(r_0^{(2)} \oplus b_2),\, r_3^{(0)} \oplus S_3(r_0^{(3)} \oplus b_3),\, S_0(r_0^{(0)} \oplus b_0)\right),$$
$$\left(r_1^{(1)} \oplus S_2(r_0^{(2)} \oplus b_2),\, r_2^{(1)} + S_3(r_0^{(3)} \oplus b_3),\, r_3^{(1)} \oplus S_0(r_0^{(0)} \oplus b_0),\, S_1(r_0^{(1)} \oplus b_1)\right),$$
$$\left(r_1^{(2)} \oplus S_3(r_0^{(3)} \oplus b_3),\, r_2^{(2)} + S_0(r_0^{(0)} \oplus b_0),\, r_3^{(2)} \oplus S_1(r_0^{(1)} \oplus b_1),\, S_2(r_0^{(2)} \oplus b_2)\right),$$
$$\left(r_1^{(3)} \oplus S_0(r_0^{(0)} \oplus b_0),\, r_2^{(3)} + S_1(r_0^{(1)} \oplus b_1),\, r_3^{(3)} \oplus S_2(r_0^{(2)} \oplus b_2),\, S_3(r_0^{(3)} \oplus b_3)\right).$$

Структурна схема ВУ зображена на рис. 4.

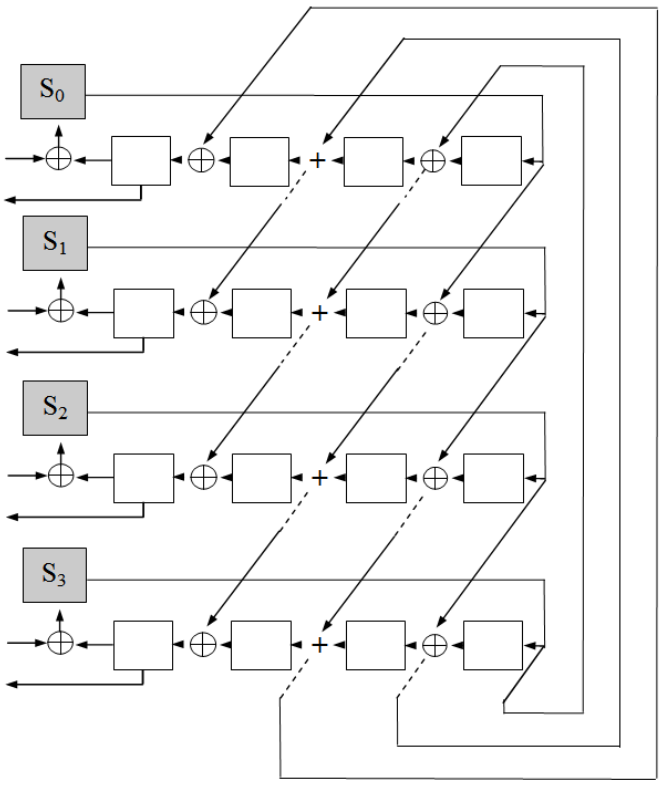

Рисунок 4 – Структурна схема ВУ

Секретним ключовим параметром шифру WSC є довгостроковий ключ (ДК) довжиною 256 бітів (32 байти), який заноситься до ЛРР та регістрів ВУ на етапі ініціалізації. Синхропосилка довжиною 64 біти (два 32-розрядних слова) є несекретним параметром, який також заноситься до ЛРР на етапі ініціалізації. Ініціалізація здійснюється наступним чином:

а) в усі комірки ЛРР записуються значення 0xFFFFFFFF;

б) 0-й байт ДК записується в $r_0^{(0)}$, 1-й байт ДК – в $r_1^{(0)}$, ..., 15-й байт ДК – в $r_3^{(3)}$;

в) схема працює 32 такти, при цьому гама, яка виробляється, додається за модулем 2 до 31-ої комірки ЛРР;

г) схема працює 1 такт;

д) перше 32-розрядне слово синхропосилки додається за модулем 2^{32} до 21-ї комірки ЛРР;

е) 16-й байт ДК записується в $r_0^{(0)}$, 17-й байт ДК – в $r_1^{(0)}$, ..., 31-й байт ДК – в $r_3^{(3)}$;

ж) схема працює 32 такти, при цьому гама, яка виробляється, додається за модулем 2 до 31-ї комірки ЛРР;

з) схема працює 1 такт;

и) друге 32-розрядне слово синхропосилки додається за модулем 2^{32} до 21-ї комірки ЛРР;

к) схема працює 32 такти.

На кроках г), з) та к) гама, що виробляється, не використовується.

III Шляхи ефективної програмної реалізації шифру WSC

Вибір поля $GF(2^{32})$ для реалізації ЛРР пояснюється ефективною програмною реалізацією операцій з елементами поля на сучасних універсальних 32-розрядних процесорах. ЛРР над полем $GF(2^{32})$ застосовуються також у потокових шифрах SNOW 2.0, SOBER-t32, SOBER-t128, NLS, Turing, Sosemanuk тощо.

У потоковому шифрі WSC використано ЛРР довжини 32 з поліномом зворотного зв'язку $\pi(x) = \alpha \cdot x^{31} + x^{26} + \alpha^{-1} \cdot x^{11} + 1$ над полем $GF(2^{32})$, де α є коренем поліному $x^4 + \beta^{23}x^3 + \beta^{245}x^2 + \beta^{48}x + \beta^{239}$ над полем $GF(2^8)$. У свою чергу елемент β є коренем поліному $x^8 + x^7 + x^5 + x^3 + 1$ над полем $GF(2)$. Такі елементи α та β використовуються у потоковому шифрі SNOW 2.0 [8]. Поліном $\pi(x)$ є примітивним поліномом над полем $GF(2^{32})$.

Поле $GF(2^{32})$ розглядається як розширення поля $GF(2^8)$, отже, будь-який елемент поля $\omega \in GF(2^{32})$ може бути поданий як поліном $\omega = c_3\alpha^3 + c_2\alpha^2 + c_1\alpha + c_0$ над полем $GF(2^8)$. Оскільки $\alpha^4 = \beta^{23}x^3 + \beta^{245}x^2 + \beta^{48}x + \beta^{239}$, то $\alpha \cdot \omega = c_3\alpha^4 + c_2\alpha^3 + c_1\alpha^2 + c_0\alpha =$ $(c_3\beta^{23} + c_2) \cdot \alpha^3 + (c_3\beta^{245} + c_1) \cdot \alpha^2 + (c_3\beta^{48} + c_0) \cdot \alpha + c_3\beta^{239}$. Це дозволяє множення на елементи α та α^{-1} здійснювати з використанням двох таблиць: $M_\alpha[c] = (c\beta^{23} \,||\, c\beta^{245} \,||\, c\beta^{48} \,||\, c\beta^{239})$ та $M_{\alpha^{-1}}[c] = (c\beta^{16} \,||\, c\beta^{29} \,||\, c\beta^{6} \,||\, c\beta^{64})$, де $c \in GF(2^8)$, $||$ – знак конкатенації.

Мовою програмування С множення елемента w на елемент α реалізується як `(w<<8)^Ma[w>>24]`, де Ma – таблиця $M_\alpha[c]$, а множення на елемент α^{-1} реалізується як `(w>>8)^Ma_inv[w&0xFF]`, де Ma_inv – таблиця $M_{\alpha^{-1}}[c]$.

Таким чином, множення на елементи α та α^{-1} може бути програмно реалізовано з використанням двох масивів розміру 256 32-розрядних слів кожний, та операцій XOR, AND і зсуву, які на сучасних процесорах виконуються за один такт.

Для програмної реалізації алгоритму WSC необхідно 144 байти для реалізації регістрів і 3072 байти для зберігання таблиць $M_\alpha[c]$, $M_{\alpha^{-1}}[c]$ та S-блоків S_0, S_1, S_2, S_3.

Запропонований потоковий шифр орієнтований на програмну реалізацію для 32-розрядних процесорів, наприклад, Intel, AMD, SPARC. Швидкодія програмної реалізації алгоритму WSC мовою програмування С на ПЕОМ на базі процесору Intel Pentium IV 2,4 ГГц складає приблизно 100 Мбайт/сек.

IV Аналіз криптографічних властивостей вузлів шифру WSC

Детальний аналіз криптографічних властивостей потокового шифру WSC виходить за рамки даної статті. Надалі розглядаються тільки деякі найважливіші властивості окремих вузлів, які визначають період вихідної послідовності, а також ряд характеристик нелінійної схеми ускладнення.

ЛРР, який використано у потоковому шифрі WSC, є еквівалентним 32 паралельно працюючим ЛРР довжини 1024 над полем $GF(2)$ з мінімальним поліномом зворотного зв'язку $f = x^{1024} + x^{1018} + x^{1003} + x^{1000} +$ $x^{994} + x^{992} + x^{979} + x^{976} + x^{970} + x^{968} + x^{958} + x^{955} + x^{952} + x^{949} + x^{946} + x^{944} + x^{943} + x^{940} + x^{934} + x^{932} + x^{925} + x^{923} + x^{920} + x^{919} +$ $x^{916} + x^{908} + x^{906} + x^{905} + x^{901} + x^{898} + x^{896} + x^{892} + x^{891} + x^{890} + x^{889} + x^{883} + x^{878} + x^{877} + x^{874} + x^{873} + x^{870} + x^{868} + x^{866} + x^{863} +$ $x^{862} + x^{861} + x^{859} + x^{857} + x^{856} + x^{855} + x^{853} + x^{851} + x^{850} + x^{848} + x^{847} + x^{845} + x^{844} + x^{839} + x^{838} + x^{836} + x^{833} + x^{831} + x^{828} + x^{827} +$ $x^{826} + x^{823} + x^{822} + x^{821} + x^{820} + x^{817} + x^{816} + x^{814} + x^{813} + x^{812} + x^{811} + x^{810} + x^{809} + x^{806} + x^{805} + x^{804} + x^{803} + x^{800} + x^{799} + x^{798} +$ $x^{795} + x^{794} + x^{791} + x^{790} + x^{789} + x^{788} + x^{787} + x^{783} + x^{782} + x^{779} + x^{777} + x^{770} + x^{767} + x^{766} + x^{762} + x^{761} + x^{760} + x^{757} + x^{755} + x^{754} +$ $x^{753} + x^{751} + x^{749} + x^{748} + x^{745} + x^{744} + x^{743} + x^{740} + x^{736} + x^{734} + x^{733} + x^{732} + x^{731} + x^{730} + x^{728} + x^{727} + x^{726} + x^{723} + x^{722} + x^{720} +$ $x^{719} + x^{718} + x^{716} + x^{714} + x^{713} + x^{712} + x^{710} + x^{709} + x^{702} + x^{700} + x^{695} + x^{693} + x^{690} + x^{689} + x^{687} + x^{686} + x^{684} + x^{680} + x^{678} + x^{675} +$ $x^{674} + x^{673} + x^{670} + x^{669} + x^{668} + x^{665} + x^{662} + x^{661} + x^{660} + x^{659} + x^{658} + x^{657} + x^{654} + x^{651} + x^{649} + x^{648} + x^{647} + x^{645} + x^{644} + x^{640} +$ $x^{639} + x^{637} + x^{636} + x^{635} + x^{633} + x^{631} + x^{630} + x^{624} + x^{623} + x^{622} + x^{619} + x^{618} + x^{616} + x^{614} + x^{612} + x^{610} + x^{607} + x^{604} + x^{603} + x^{602} +$ $x^{600} + x^{598} + x^{594} + x^{592} + x^{586} + x^{585} + x^{583} + x^{582} + x^{581} + x^{576} + x^{575} + x^{574} + x^{573} + x^{572} + x^{568} + x^{563} + x^{561} + x^{557} + x^{555} + x^{554} +$ $x^{551} + x^{550} + x^{549} + x^{548} + x^{543} + x^{539} + x^{538} + x^{535} + x^{532} + x^{530} + x^{527} + x^{526} + x^{525} + x^{522} + x^{521} + x^{519} + x^{518} + x^{517} + x^{514} + x^{513} +$ $x^{511} + x^{510} + x^{507} + x^{506} + x^{504} + x^{503} + x^{500} + x^{498} + x^{496} + x^{492} + x^{491} + x^{488} + x^{485} + x^{484} + x^{483} + x^{482} + x^{480} + x^{478} + x^{477} + x^{476} +$ $x^{475} + x^{474} + x^{472} + x^{470} + x^{469} + x^{467} + x^{462} + x^{460} + x^{459} + x^{458} + x^{457} + x^{455} + x^{451} + x^{450} + x^{449} + x^{446} + x^{445} + x^{442} + x^{441} + x^{439} +$ $x^{438} + x^{437} + x^{436} + x^{432} + x^{427} + x^{426} + x^{425} + x^{424} + x^{423} + x^{422} + x^{419} + x^{418} + x^{417} + x^{416} + x^{413} + x^{411} + x^{406} + x^{405} + x^{404} + x^{403} +$ $x^{398} + x^{394} + x^{392} + x^{390} + x^{388} + x^{385} + x^{384} + x^{382} + x^{381} + x^{378} + x^{376} + x^{374} + x^{372} + x^{371} + x^{369} + x^{367} + x^{365} + x^{362} + x^{361} + x^{353} +$ $x^{351} + x^{350} + x^{349} + x^{345} + x^{343} + x^{342} + x^{338} + x^{336} + x^{333} + x^{332} + x^{331} + x^{330} + x^{328} + x^{327} + x^{326} + x^{324} + x^{322} + x^{321} + x^{320} + x^{317} +$ $x^{316} + x^{313} + x^{311} + x^{310} + x^{306} + x^{305} + x^{300} + x^{299} + x^{297} + x^{296} + x^{292} + x^{290} + x^{289} + x^{288} + x^{286} + x^{282} + x^{280} + x^{277} + x^{276} + x^{273} +$ $x^{272} + x^{271} + x^{269} + x^{268} + x^{266} + x^{265} + x^{262} + x^{260} + x^{257} + x^{256} + x^{255} + x^{254} + x^{252} + x^{245} + x^{244} + x^{242} + x^{240} + x^{239} + x^{238} + x^{232} +$

$x^{230} + x^{229} + x^{228} + x^{227} + x^{226} + x^{224} + x^{223} + x^{221} + x^{220} + x^{215} + x^{214} + x^{213} + x^{210} + x^{209} + x^{208} + x^{204} + x^{203} + x^{199} + x^{195} + x^{193} + x^{192} + x^{191} + x^{189} + x^{188} + x^{182} + x^{181} + x^{179} + x^{177} + x^{175} + x^{173} + x^{171} + x^{169} + x^{168} + x^{167} + x^{166} + x^{165} + x^{163} + x^{162} + x^{161} + x^{160} + x^{158} + x^{154} + x^{153} + x^{151} + x^{148} + x^{146} + x^{145} + x^{143} + x^{142} + x^{141} + x^{140} + x^{136} + x^{133} + x^{132} + x^{131} + x^{129} + x^{125} + x^{124} + x^{121} + x^{120} + x^{119} + x^{115} + x^{110} + x^{107} + x^{102} + x^{101} + x^{99} + x^{98} + x^{95} + x^{89} + x^{88} + x^{87} + x^{84} + x^{81} + x^{80} + x^{76} + x^{75} + x^{63} + x^{58} + x^{55} + x^{54} + x^{44} + x^{43} + x^{37} + x^{32} + x^{11} + 1$, де + – операція додавання у полі $GF(2)$. Цей поліном було обчислено з використанням алгоритму Берлекемпа-Мессі [2]. Поліном f має 451 терм (одночлен) і є примітивним поліномом над полем $GF(2)$, отже, період вихідної послідовності ЛРР дорівнює $2^{1024} - 1 \approx 10^{308}$ [2].

Варто зауважити, що алгоритм WSC має найбільший період гами серед потокових шифрів, в яких використовуються ЛРР над полем $GF(2^{32})$.

У шифрі WSC пропонується використати S-блоки, всі координатні функції яких є збалансованими та задовольняють властивості кореляційної імунності першого порядку, тобто, є 1-усталеними функціями [9]. Для таких функцій рівноймовірними (збалансованими) є як сама функція, так і всі її підфункції, які утворюються шляхом фіксації однієї змінної. Властивість кореляційної імунності координатних функцій ускладнює застосування кореляційних методів криптографічного аналізу для потокового шифру WSC.

Приклад 8×8 S-блоку з властивістю кореляційної імунності усіх координатних функцій у шістнадцятковому виді наведено нижче. Цей S-блок позначимо S_{im}.

```
6f 33 df 6a 61 c8 d6 3c 5d b0 6e 44 92 22 de 4c
b7 75 94 39 e0 11 3a 8b 1b 46 8d d8 69 82 c5 a4
63 9a 4e a6 bd 41 a7 ab 78 00 4a 1c d3 59 65 f1
34 87 d0 55 cf 36 c9 fe 43 a8 ed 72 2c 3b 8e 17
3f bc 37 05 b2 58 a3 71 86 f4 9d 4b 8a 10 d9 bf
c0 fb 6d 16 1a a1 ef c4 f3 a2 48 a9 27 ec 7e 95
26 f8 49 9b 47 7d ba 64 9c ae 91 c3 f5 0f 60 52
30 0d 6b 42 d7 b6 14 b9 c1 cc aa 13 28 5f a5 5e
d1 0b c2 d5 0c 4f 98 29 9e 77 f6 4d 20 83 74 e1
e6 d2 dc 90 0a fa e8 0e b3 e5 07 19 5b fd 7f 2b
e4 2d 2a af f9 ac b1 96 70 e3 be 35 67 38 bb 32
54 2e 08 da 89 45 57 81 9f 93 1d 31 fc c6 04 e2
6c 2f 03 62 80 15 97 24 06 cd 09 f0 1e db 3d ea
5c 18 e7 73 76 5a 01 ff 85 eb b4 a0 79 25 12 ee
88 cb f7 66 51 84 b8 dd f2 1f 99 8c 53 ce 40 c7
7a d4 e9 7b ad 8f 02 50 21 b5 3e 68 ca 7c 23 56
```

Для синтезу S-блоків з властивістю кореляційної імунності координатних функцій розроблено спеціальний алгоритм, в якому поєднується евристичний алгоритм побудови 1-усталеної функції з алгоритмом послідовного спрямованого перебору координатних функцій. Час генерації одного 8×8 S-блоку з вимогою $N \geq 96$, де N – нелінійність S-блоку, на ПЕОМ на базі процесора Intel Pentium IV 2,4 ГГц дорівнює приблизно 8 годин.

Нелінійність N згідно з [10] може бути обчислена на основі таблиці лінійних апроксимацій (Linear Approximation Table) S-блоку. Для $n \times n$ S-блоку ця таблиця визначається як матриця розміру $2^n \times 2^n$, в якій елемент з номером (α, β) обчислюється за формулою

$$LAT(\alpha,\beta) = \#\{x \in V_n \mid \bigoplus_{i=1}^{n} x_i\alpha_i = \bigoplus_{i=1}^{n} y_i\beta_i\} - 2^{n-1}, \text{ де } y = S(x).$$

Параметр N обчислюється за формулою

$$N = 2^{n-1} - \max_{\alpha,\beta \in V \setminus \{0\}} |LAT(\alpha,\beta)|.$$

Для S-блоку S_{im} нелінійність координатних функцій дорівнює 116, 112, 112, 112, 112, 108, 108, 108 відповідно. Нелінійність S-блоку дорівнює 96.

Степені нелінійності всіх координатних функцій дорівнюють 6, що є максимальним значенням для 1-стійких функцій. Крім того, для всіх координатних функцій досягається нерівність Зігентайлера для кожної

з восьми змінних, отже, функції є оптимальними [9]. Порядок нелінійності S-блоку S_{im} дорівнює 6, що є максимально можливим значенням для S-блоків з властивістю 1-стійкості координатних функцій.

Згідно з [11] кращі показники нелінійності N мають тільки S-блоки, побудовані на основі мультипликативного звертання елемента у скінченному полі $GF(2^8)$, а також S-блоки алгоритмів E2, Skipjack, BelT. Аналогічний показник, а саме, 96, мають S-блоки алгоритмів Crypton, Snow 1.0, Twofish, Whirlpool, CS. Гірший показник нелінійності мають S-блоки алгоритмів MD2, RC2, Safer+, Anubis, Turing, DESX.

Максимальні значення в таблиці різниць S-блоку S_{im} відносно бінарних операцій додавання за модулем 2 ($\oplus$) та додавання за модулем 256 (+) дорівнюють $R_{\oplus\oplus}$=10, $R_{\oplus+}$=8, R_{++}=8, $R_{+\oplus}$=7. Нагадаємо, що відносно бінарних операцій o_1 та o_2, які задані на лінійному просторі V_n, максимальне значення в таблиці різниць підстановки S обчислюється за формулою

$$R_{o_1 o_2} = \max_{\alpha,\beta \in V_n, \alpha \neq 0} \sum_{x \in V_n} I\{S(x\, o_1\, \alpha) = S(x)\, o_2\, \beta\},$$ де $I\{\varepsilon\}$ – індикатор події ε [9].

Як видно S-блок S_{im} не є оптимізованим відносно методів диференційного криптоаналізу, однак, варто зауважити, що цей метод криптоаналізу для потокових шифрів, зазвичай, не застосовується.

За результатами аналізу S-блоку S_{im} можна зробити висновок, що він є стійким як відносно методу лінійного криптоаналізу, так і відносно методу кореляційного криптоаналізу.

Довжина ключа алгоритму WSC дорівнює 256 бітів. Відносно методу тотального перебору ключів криптографічна стійкість алгоритму WSC оцінюється величиною 10^{77} елементарних операцій. Аналіз криптографічної стійкості відносно інших методів криптоаналізу є напрямком подальших досліджень.

V Оцінка „статистичної безпеки" алгоритму WSC

Однією із важливих властивостей генераторів гами потокових шифрів є „статистична безпека". Вважається, що алгоритм є „статистично безпечним", якщо послідовність, яку він генерує, за своїми властивостями не поступається випадковій послідовності, тобто, статистично не відрізняється від послідовності випадкових величин Бернуллі з параметром $p = 0.5$.

Однім із підходів до оцінки статистичної безпеки криптографічних алгоритмів є підхід, який запропонований NIST у рекомендаціях NIST SP 800-22 [12]. Вони містять набір статистичних тестів та методику тестування. Назви статистичних тестів, які визначені в NIST SP 800-22, наведені в таблиці 1. "Дефекти" в послідовності, які виявляються тестами, наведені в таблиці 2.

Таблиця 1 – Назви статистичних тестів NIST SP 800-22

№	Назва тесту (англійською мовою)	Назва тесту (українською мовою)
1	Frequency (Monobit) Test	Частотний (монобітний) тест
2	Frequency Test within a Block	Частотний тест (всередині блоку)
3	Cumulative Sums Test	Тест накопичених сум
4	Runs Test	Тест серій
5	Test for the Longest Run of Ones in a Block	Тест серій одиниць у послідовностях
6	Binary Matrix Rank Test	Тест рангів бінарних матриць
7	Discrete Fourier Transform (Spectral) Test	Спектральний тест
8	Non-overlapping Template Matching Test	Тест шаблонів, що не перекриваються
9	Overlapping Template Matching Test	Тест шаблонів, що перекриваються
10	Maurer's 'Universal Statistical' Test	Універсальний статистичний тест Маурера
11	Approximate Entropy Test	Тест апроксимованої ентропії
12	Random Excursions Test	Тест випадкових відхилень
13	Random Excursions Variant Test	Тест випадкових відхилень (варіант)
14	Serial Test	Послідовний тест
15	Linear Complexity Test	Тест лінійної складності

Таблиця 2 – "Дефекти" в послідовності, які виявляються тестами NIST SP 800-22

№	Назва тесту	Дефект у послідовності, що виявляється тестом
1	Frequency (Monobit) Test	Занадто багато нулів або одиниць у послідовності
2	Frequency Test within a Block	Локалізовані відхилення частоти появи одиниць в блоці від ідеального значення 0.5
3	Cumulative Sums Test	Велика кількість одиниць або нулів на початку або наприкінці двійкової послідовності
4	Runs Test	Занадто швидка або занадто повільна зміна знаку в ході генерації послідовності
5	Test for the Longest Run of Ones in a Block	Відхилення від теоретичного закону розподілу максимальних довжин серій одиниць
6	Binary Matrix Rank Test	Відхилення емпіричного закону розподілу значень рангів матриць від теоретичного, що вказує на залежність символів у послідовності
7	Discrete Fourier Transform (Spectral) Test	Наявність періодичних складових (трендів) у двійковій послідовності
8	Non-overlapping Template Matching Test	Велика кількість заданих неперіодичних шаблонів, що не перекриваються, у послідовності
9	Overlapping Template Matching Test	Велика кількість m-бітових серій із одиниць у послідовності
10	Maurer's 'Universal Statistical' Test	Можливість стиснення послідовності
11	Approximate Entropy Test	Нерівномірність розподілу m-бітових слів у послідовності (регулярність властивостей джерела)
12	Random Excursions Test	Відхилення від теоретичного закону розподілу візитів у конкретний стан при випадковому блуканні
13	Random Excursions Variant Test	Відхилення від теоретично очікуваної загальної кількості візитів при випадковому блуканні в заданий стан
14	Serial Test	Нерівномірність розподілу m-бітових слів у послідовності
15	Linear Complexity Test	Відхилення емпіричного розподілу довжин еквівалентних ЛРР для послідовностей фіксованої довжини від теоретичного закону розподілу для випадкової послідовності, що вказує на недостатню складність послідовності, що тестується

Для алгоритму WSC проведене статистичне тестування 100 послідовностей довжиною 1000000 біт кожна із застосуванням 188 статистичних тестів (15 тестів із різними вхідними параметрами) згідно з рекомендаціями NIST SP 800-22 та побудовано „статистичний портрет" генератора гами. За результатами аналізу „статистичного портрету" згідно з методикою NIST SP 800-22 зроблено висновок, що послідовності з виходу генератора гами потокового шифру WSC задовольняють вимогам незалежності та рівноймовірності, отже, алгоритм є „статистично безпечним".

VI Висновки

В умовах поступового переходу від апаратних реалізацій засобів КЗІ до програмних та програмно-апаратних найбільш доцільним є розробка програмно-орієнтованих потокових шифрів гамування, які ефективно реалізуються на сучасних 32-розрядних процесорах.

Криптографічний алгоритм потокового шифрування WSC, запропонований у статті, побудований за „класичною схемою", яка містить ЛРР та нелінійну схему ускладнення. Особливістю алгоритму є застосування S-блоків, усі координатні функції яких задовольняють властивості кореляційної імунності першого порядку та оптимальності, що ускладнює застосування методів кореляційного криптоаналізу для цього шифру.

Принципи побудови потокового шифру WSC можуть бути використані під час синтезу криптографічних схем, призначених для реалізації в програмних або програмно-апаратних засобах криптографічного захисту інформації.

Література: ***1.*** *R. A. Rueppel. Analysis and Design of stream ciphers. Springer-Verlag, 1986.* ***2.*** *Лиддл Р., Нидеррайтер Г. Конечные поля. –М.: Мир, 1988.* ***3.*** *R. Anderson and M. Roe, "A5," Technical report, 1994.* ***4.*** *Schneier B. Applied cryptography. Second edition. Protocols, Algorithms and Source Code in C, John Wiley & Sons,*

Inc. 1996. ***5.*** *R. Rivest, The RC4 encryption algorithm, RSA Data Security, Inc., 1992.* ***6.*** *NESSIE. New European Schemes for Signatures, Integrity, and Encryption. Available at http://www.cryptonessie.org.* ***7.*** *eSTREAM – The ECRYPT Stream Cipher Project. Available at http://www.ecrypt.eu.org* ***8.*** *P. Ekdahl and T. Johansson. A new version of the stream cipher SNOW. In Selected Areas in Cryptography – SAC 2002, volume 2295 of Lecture Notes in Computer Science, pages 47–61. Springer-Verlag, 2002.* ***9.*** *Siegentaler T. Correlation-immunity of Nonlinear Combining Functions for Cryptographic Applications. // IEEE Trans. on Information Theory. 1984. Vol. IT-30.* ***5.*** *P. 776–780.* ***10.*** *M. Matsui. Linear cryptanalysis method for DES cipher. Abstracts of EUROCRYPT'93, May, 1993* ***11.*** *Л. Скрипник, О. Дирда. Порівняльний аналіз методів побудови та властивостей S-блоків ряду сучасних криптографічних алгоритмів. // Правове, нормативне та метрологічне забезпечення систем захисту інформації в Україні, –К.: 2005. –Вип.10. –С.85–98.* ***12.*** *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, COMPUTER SECURITY. National Institute of Standards and Technology (NIST), Special Publications 800-22, USA, 2000, http://csrc.nist.gov/rng/rng2.htm.*