НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
Навчально-науковий фізико-технічний інститут
Кафедра математичних методів захисту інформації

«На правах рукопису»
УДК 004.056.5

«До захисту допущено»
В.о. завідувача кафедри
_____ Сергій ЯКОВЛЄВ
«__» _____ 2023 р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою
«Математичні методи криптографічного захисту інформації»

зі спеціальності: 113 Прикладна математика
на тему: **«Аналіз фази ініціалізації потокового шифру «Струмок»»**

Виконав:
студент IV курсу, групи ФІ-93
Папишев Олег Ігорович _____

Керівник:
к. ф.-м. н., ст. викл. каф. ММЗІ
Фесенко Андрій Вячеславович _____

Рецензент:
посада, степінь, звання
Прізвище Ім'я По-батькові _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ — 2023

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
Навчально-науковий фізико-технічний інститут
Кафедра математичних методів захисту інформації

Рівень вищої освіти — перший (бакалаврський)
Спеціальність — 113 Прикладна математика,
ОПП «Математичні методи криптографічного захисту інформації»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Сергій ЯКОВЛЄВ
«__» _____ 2023 р.

ЗАВДАННЯ
на дипломну роботу

Студент: Папишев Олег Ігорович

1. Тема роботи: «*Аналіз фази ініціалізації потокового шифру «Струмок»*», науковий керівник роботи: к. ф.-м. н., ст. викл. каф. ММЗІ Фесенко Андрій Вячеславович,

затверджені наказом по університету №__ від «__» _____ 2023 р.

2. Термін подання студентом роботи: «__» _____ 2023 р.

3. Об'єкт дослідження: процес перетворення інформації в фазі ініціалізації шифру «Струмок».

4. Предмет дослідження: стійкість до атак на фазу ініціалізації шифру «Струмок».

5. Перелік завдань: розглянути особливості побудови шифрів сімейства «SNOW», включаючи шифр «Струмок», та особливості їх фаз ініціалізації; дослідити наявні вразливості фази ініціалізації шифру «Струмок–512» та наявні атаки, які використовують ці вразливості; побудувати модифікацію шифру «Струмок–512» з метою зменшення розміру внутрішнього стану для спрощення аналізу фази ініціалізації; вдосконалити фазу ініціалізації шифру «Струмок–512» з метою підвищення криптографічної стійкості шифру; проаналізувати

застосовність атаки раундового зсуву до запропонованої вдосконаленої фази ініціалізації; проаналізувати застосовність атаки пов'язаних ключів до запропонованої вдосконаленої фази ініціалізації.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: презентація доповіді.

7. Орієнтовний перелік публікацій: планується доповідь на всеукраїнській конференції

8. Дата видачі завдання: 10 вересня 2022 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	01-15 вересня 2022 р.	Виконано
2	Огляд опублікованих джерел за тематикою дослідження	Вересень-жовтень 2022 р.	Виконано
3	Аналіз структури «SNOW» подібних шифрів	Жовтень-грудень 2022 р.	Виконано
4	Дослідження вразливостей фази ініціалізації потокового шифру «Струмок–512»	Грудень-лютий 2023 р.	Виконано
5	Побудова нової фази ініціалізації шифру «Струмок–512»	Лютий-квітень 2023 р.	Виконано
6	Дослідження стійкості запропонованої фази ініціалізації шифру «Струмок–512»	Квітень-травень 2023 р.	Виконано
7	Оформлення та захист дипломної роботи	Травень-червень 2023 р.	Виконано

Студент _____ Олег ПАПИШЕВ

Керівник _____ Андрій ФЕСЕНКО

РЕФЕРАТ

Дипломна робота містить: 59 сторінок, 6 рисунків, 21 джерело.

Мета дослідження: аналіз стійкості фази ініціалізації шифру «Струмок».

Об'єкт дослідження: процес перетворення інформації в фазі ініціалізації шифру «Струмок».

Предмет дослідження: стійкість до атак на фазу ініціалізації шифру «Струмок».

В роботі проведено порівняльний аналіз фази ініціалізації потокового шифру «Струмок» з іншими шифрами сімейства «SNOW», зокрема зі «SNOW 2.0». Для спрощення аналізу фази ініціалізації шифру «Струмок–512» побудована модифікація «Струмок–512–32» внутрішній стан якої зменшено. Побудована нова вдосконалена фаза ініціалізації шифру «Струмок–512». Доведено, що атака пов'язаних ключів, та атака раундового зсуву, є незастосованими до шифру «Струмок–512» з вдосконаленою фазою ініціалізації. Доведено, що шифр «Струмок–512–32» з вдосконаленою фазою ініціалізації більш стійкий до статистичної атаки на фазу ініціалізації, ніж шифр «Струмок–512–32» з не зміненою фазою ініціалізації.

ПОТОКОВІ ШИФРИ, ФАЗА ІНІЦІАЛІЗАЦІЇ, КРИПТОГРАФІЧНА СТІЙКІСТЬ, СТРУМОК

ABSTRACT

The thesis consists of 59 pages and includes 6 figures and 21 references.

Research objective: Analysis of the initialization phase stability of the «Strumok» cipher.

Research object: The process of information transformation in the initialization phase of the «Strumok» cipher.

Research subject: Resistance to attacks on the initialization phase of the «Strumok» cipher.

The thesis presents a comparative analysis of the initialization phase of the «Strumok» stream cipher with other ciphers from the "SNOW"family, particularly with "SNOW 2.0". To simplify the analysis of the initialization phase of the «Strumok–512» cipher, a modification called «Strumok–512–32» was constructed, reducing its internal state. A new improved initialization phase for the «Strumok–512» cipher was developed. It has been proven that attacks such as related-key attacks and round-recovery attacks are not applicable to the «Strumok–512» cipher with the improved initialization phase. It has also been demonstrated that the «Strumok–512–32» cipher with the improved initialization phase is more resistant to statistical attacks on the initialization phase compared to the «Strumok–512–32» cipher with the unchanged initialization phase.

STREAM CIPHERS, INITIALIZATION PHASE, CRYPTOGRAPHIC SECURITY, STRUMOK

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

АНФ — алгебраїчна нормальна форма.

$\biguplus$ — операція конкатенації

$+_{32}$ — операція додавання за модулем 2^{32}.

$\otimes$ — операція множення на многочлен.

$\oplus$ — операція побітового додавання

$\mathbb{GF}(2^x)$ — поле Галуа порядка 2^x

LFSR (linear feedback shift register) — регістр зсуву з лійнійним зворотнім зв'язком

FSM(finite state machine) — скінченний автомат

IV (initialization vector) —- вектор ініціалізації

ВСТУП

Актуальність дослідження.Актуальність дослідження полягає у тому, що шифр «Струмок» затверджений в Україні у якості стандарту потокового шифрування ДСТУ8845 : 2019[18], але наразі цей шифр не є достатньо проаналізованим. Для забезпечення криптографічної стійкості шифру «Струмок», потрібно провести дослідження вразливостей шифру, проаналізувати наявні атаки, що базуються на використанні виявлених вразливостей та запропонувати вдосконалення алгоритму, яке збільшить його криптографічну стійкість до цих атак.

Метою дослідження є забезпечення криптографічної стійкості потокового шифру «Струмок».

Для досягнення мети необхідно вирішити такі завдання:

1) розглянути особливості побудови шифрів сімейства «SNOW», включаючи шифр «Струмок», та особливості їх фаз ініціалізації;

2) дослідити наявні вразливості фази ініціалізації шифру «Струмок–512» та наявні атаки, які використовують ці вразливості;

3) побудувати модифікацію шифру «Струмок–512» з метою зменшення розміру внутрішнього стану для спрощення аналізу фази ініціалізації;

4) вдосконалити фазу ініціалізації шифру «Струмок–512» з метою підвищення криптографічної стійкості шифру;

5) проаналізувати застосовність атаки раундового зсуву до запропонованої вдосконаленої фази ініціалізації;

6) проаналізувати застосовність атаки пов'язаних ключів до запропонованої вдосконаленої фази ініціалізації.

Об'єктом дослідження є процес перетворення інформації в фазі ініціалізації шифру «Струмок».

Предметом дослідження є стійкість до атак на фазу ініціалізації шифру «Струмок».

При розв'язанні поставлених завдань використовувались такі *методи дослідження*: теорії імовірностей, математичної статистики, комбінаторного аналізу, теорії кодування, теорії складності алгоритмів.

Наукова новизна отриманих результатів полягає в побудові нової вдосконаленої фази ініціалізації шифру «Струмок–512» та доведенні, що вдосконалена фаза ініціалізації є не вразливою до відомих наявних атак на фазу ініціалізації шифру «Струмок–512>.

Практичне значення результатів полягає в покращенні стійкості фази ініціалізації потокового шифру національного стандарту ДСТУ8845 : 2019 завдяки заміні на запропоновану вдосконалену фазу ініціалізації цього шифру.

1 ОСОБЛИВОСТІ ФАЗИ ІНІЦІАЛІЗАЦІЇ ПОТОКОВИХ ШИФРІВ СІМЕЙСТВА «SNOW»

У цьому розділі буде досліджено будову потокового шифру «Струмок» та «SNOW 2.0». Наведено алгебраїчну модель шиифру «Струмок», в термінах якою буде виконуватись аналізу шифру. Розглянуто фази ініціалізації інших шифрів сімейства «SNOW». Виділено особливості їх структури, які можуть впливати на стійкість шифру.

1.1 Фаза ініціалізації «SNOW 1.0»

У 2000 році Ekdahl P. та Johansson T. опублікували роботу [6] в якій був вперше запропонован потоковий шифр «SNOW» (далі в роботі ця версія буде використовуватись під назвою «SNOW–1.0»).

Не зважаючи на те, що «SNOW–1.0» мав деякі проблеми з криптостійкістю, шифр зміг отримати широку популярність та стати пробатьком сімейства шифрів «SNOW», до якого в подальшому війде і «Струмок».

В даному розділі основна увага буде приділена аналізу того, які правила в фазі ініціалізації потокового шифру «SNOW–1.0» закріпились в наступних версіях, а які були помилковими.

Потоковий шифр «SNOW–1.0» (дивись рис.1.1) складається з регістру зсуву з лінійним зворотнім зі'язком, який містить 16 32–бітних комірки s_i, $i \in \{1, \ldots, 16\}$ та скінченого автомату який містить дві 32–бітних комірки R_1, R_2. У алгоритмі виконються операції XOR, додавання за модулем 2^{32}, побітовий зсув, множення на α та операція S.

Аналогічно до шифру «Струмок» існує дві версії шифру «SNOW–1.0»: з ключем $K = \{k_0, k_1, k_2, k_3\}$ довжиною 128 бітів і ключем $K = \{k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7\}$ довжиною 256 бітів. Вектор ініціалізаціїї

$IV = \{iv_0, iv_1\}$ завжди має довжину 64 біти.

Бітовий розмір векторів k_i, iv_0, iv_1 дорівнює по 32 біти кожен, і не залежить від розміру ключа K.

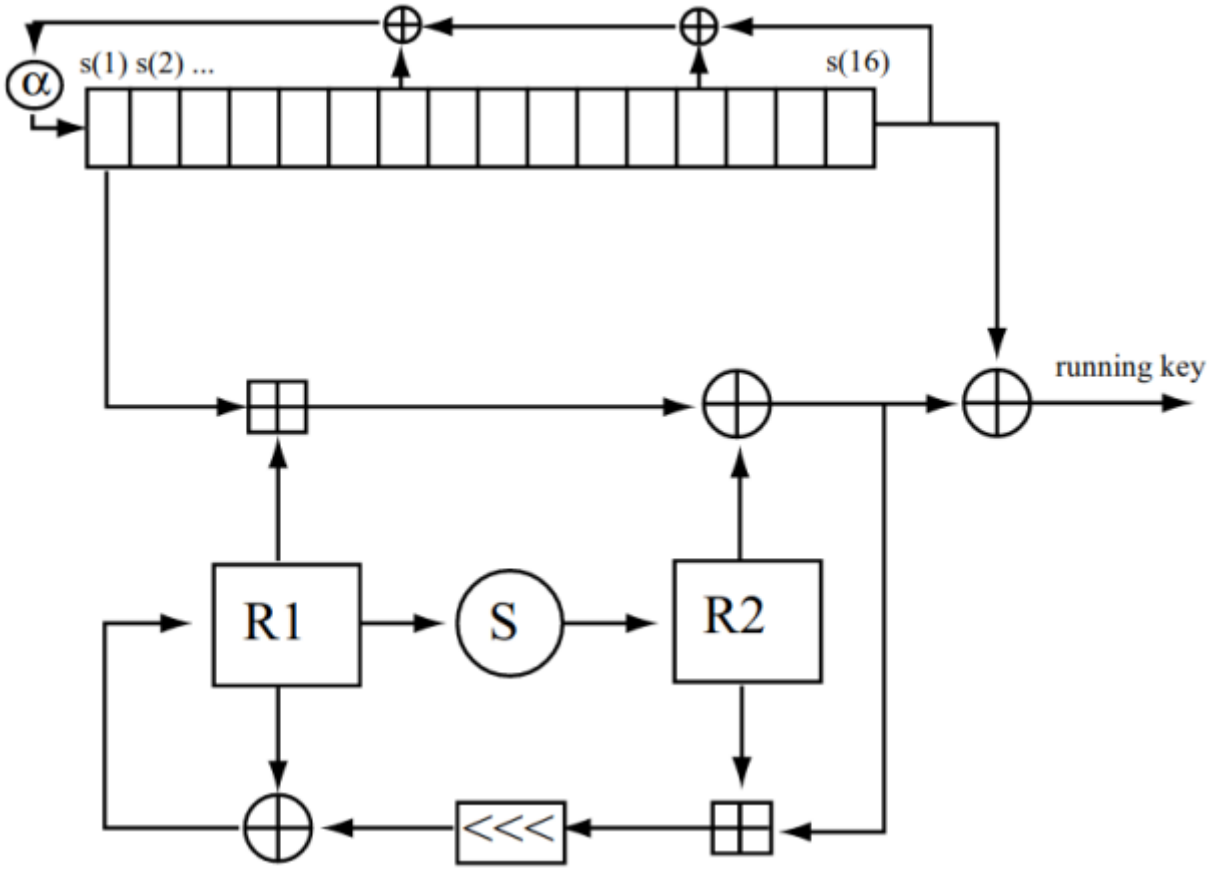

Рисунок 1.1 – Схема роботи шифру «SNOW 1.0»

Алгоритм починається з фази ініціалізаціїї, правила якої залежать від розміру ключа.

У випадку 128–бітового ключа:

$$s_0 = s_8 = \neg k_0, \quad s_1 = s_9 = \neg k_1, \quad s_2 = s_{10} = \neg k_2,$$
$$s_3 = s_{11} = \neg k_3, \quad s_4 = k_0, \quad s_5 = s_{13} = k_1,$$
$$s_6 = s_{14} = k_2, \quad s_7 = k_3, \quad s_{12} = IV_0,$$
$$s_{15} = IV_1,$$

Значення R_1 і R_2 встановлюються рівними $\{0\}^{32}$.

У випадку 256–бітового ключа:

$$
\begin{aligned}
&s_0 = \neg k_0, && s_1 = \neg k_1, && s_2 = \neg k_2, && s_3 = \neg k_3,\\
&s_4 = \neg k_4, && s_5 = \neg k_5, && s_6 = \neg k_6, && s_7 = \neg k_7,\\
&s_8 = k_0, && s_9 = k_1, && s_{10} = k_2, && s_{11} = k_3,\\
&s_{12} = k_4 \oplus IV_1, && s_{13} = k_5, && s_{14} = k_6, && s_{15} = k_7 \oplus IV_0,
\end{aligned}
$$

Значення R_1 і R_2 встановлюються рівними $\{0\}^{32}$.

Після цього виконується 64 раунди ініціалізації, під час яких перемішується ключ та вектору ініціалізації. Це потрібно для ускладнення зв'язку між бітами виходу ти початковим заповненням.

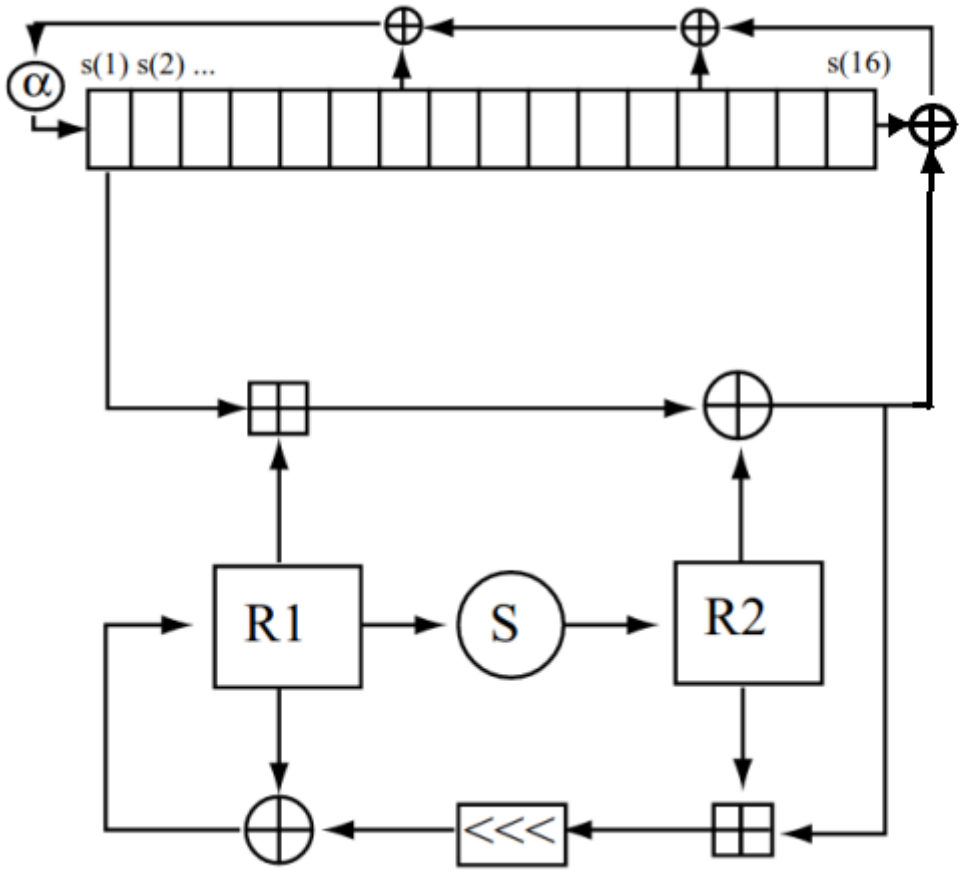

Рисунок 1.2 – Схема роботи шифру «SNOW 1.0» під час фази ініціалізації.

Тобто почергово, для кожного $t \in (0{,}64)$ обраховується

$$R1_{t+1} = (((s_{15_t} + 32R1_t) \oplus R2_{t+32R2_t}) \ll 7) \oplus R1_t,$$

$$R2_{t+1} = S(R1_t),$$

$$s_{j_{t+1}} = s_{j+1_t}, \quad j = 0, \ldots, 14,$$

$$s_{15_{t+1}} = \alpha(s_{0_t} \oplus s_{3_t} \oplus s_{9_t} \oplus (s_{15_t} + 32R1_t) \oplus R2_t)$$

Під час раундів ініціалізації не генеруються слова з ключового потоку, а структура шифру відрізняється (дивись рис.1.2) від структури під час шифрування. Ця властивість притаманна усім шифрам з сімейства «SNOW».

Наостанок потрібно виконати ще один раунд без генерації ключової гамми, але вже в стандартному режимі, а саме:

$$s_{15_{t+1}} = \alpha(s_{0_t} \oplus s_{3_t} \oplus s_{9_t})$$

Після цього фаза ініціалізації вважається завершеною і можна приступати до генерування ключового потоку.

Слід зазначити, що в подальших модифікаціях шифру кількість раундів в фазі ініціалізації було скорочено, а вектор ініціалізації, натомсть збільшено.

1.2 Будова потокового шифру «SNOW 2.0»

Оскільки Потоковий шифр «SNOW 2.0» був попередником «Струмка» вони мають практично однакову структуру.Це підтверджується схемою роботи «SNOW 2.0» (дивись рис.1.3). Зважаючи на це, увага в даному підрозділі буде зосереджена на відмінностях між цими шифрами.

Основна відмінність викликана тим, що «SNOW 2.0» розроблявся

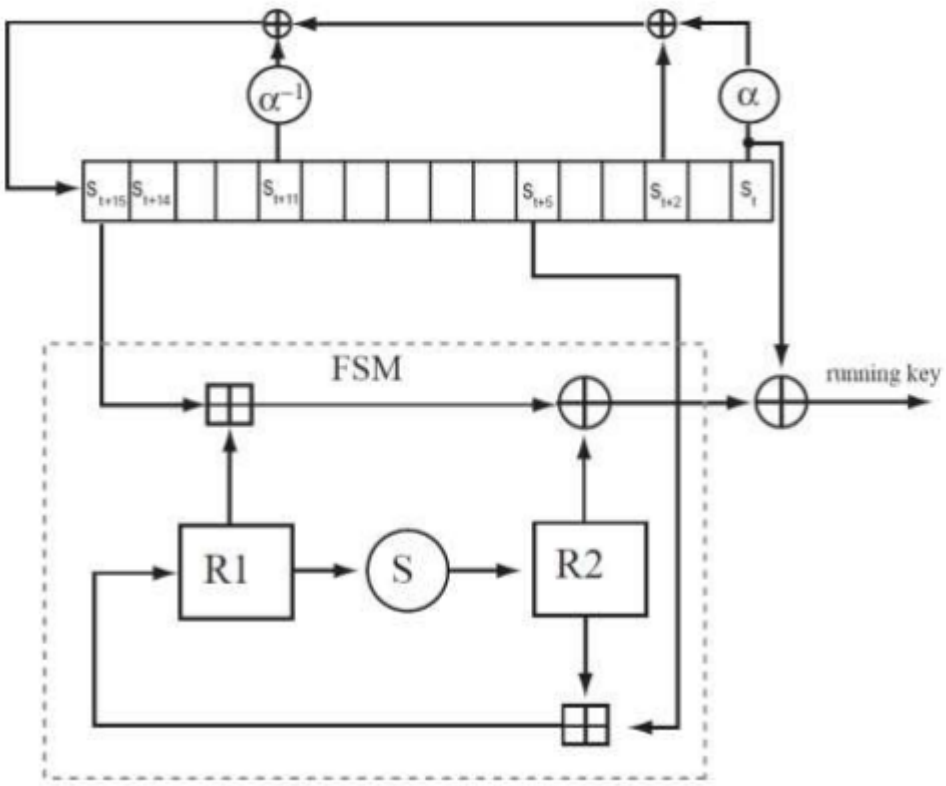

Рисунок 1.3 – Схема роботи шифру «SNOW 2.0»

для використання на 32 розрядних процесорах[1], відповідно розмірності комірок s_i, r_0, r_1 дорівнють 32 біти. З цього випливає, що регістр зсуву визначено над полем Галуа $\mathbb{F}_{2^{32}}$ з поліномом зворотного зв'язку

$$f(x) = \alpha x^{16} + x^{14} + \alpha^{-1} x^5 + 1,$$

де $\alpha \in \mathbb{F}_{2^{32}}$ є коренем поліному над полем $\mathbb{F}_{2^8}$

$$g(z) = z^4 + \beta^{23} z^3 + \beta^{245} z^2 + \beta^{48} z + \beta^{239},$$

де β — це корінь примітивного над полем $\mathbb{F}_2$ поліному

$$p(y) = y^8 + y^7 + y^5 + y^3 + 1.$$

Разом з цим, вектор ініціалізації $IV = \{iv_3, iv_2, iv_1, iv_0\}$ та ключ шифрування $K = \{k_7, k_6, k_5, k_4, k_3, k_2, k_1, k_0\}$ для 256 бітового ключа або $K = \{k_3, k_2, k_1, k_0\}$ для 128 бітного, складаються з тієї ж кількості слів що

й у «Струмку», хоча їх розмір у бітах в два рази менший.

Доцільно очікувати, що правила ініціалізації залишаться без змін, але це не так. Ініціалізація «SNOW 2.0» більш впорядкована.

Функція INIT для «SNOW 2.0»: Вхід: вектор ініціалізації $IV = \{iv_3, iv_2, iv_1, iv_0\}$, ключ шифрування $K = \{k_3, k_2, k_1, k_0\}$ довжиною 128 бітів, або ключ шифрування $K = \{k_7, k_6, k_5, k_4, k_3, k_2, k_1, k_0\}$ довжиною 256 бітів. Довжини кожного слова векторів – 32 біти

Вихід: стан FSM та LFSR після 32 раундів ініціалізації.

На першому етапі ініціалізації виконується заповнення стану регістр зсуву з лінійним зворотнім зв'зком.

Для ключа довжиною 128 біт:

$$
\begin{aligned}
&S_{15} = K_3 \oplus IV_0, && S_{14} = K_2, && S_{13} = K_1,\\
&S_{12} = K_0 \oplus IV_1, && S_{11} = \neg K_3, && S_{10} = \neg K_2 \oplus IV_2,\\
&S_9 = \neg K_1 \oplus IV_3, && S_8 = \neg K_0, && S_7 = K_3,\\
&S_6 = K_2, && S_5 = K_1, && S_4 = K_0,\\
&S_3 = \neg K_3, && S_2 = \neg K_2, && S_1 = \neg K_1,\\
&S_0 = \neg K_0.
\end{aligned}
$$

Для ключа довжиною 256 біт:

$$
\begin{aligned}
&S_{15} = K_7 \oplus IV_0, && S_{14} = K_6, && S_{13} = K_5,\\
&S_{12} = K_4 \oplus IV_1, && S_{11} = K_3, && S_{10} = K_2 \oplus IV_2,\\
&S_9 = K_1 \oplus IV_3, && S_8 = K_0, && S_7 = \neg K_7,\\
&S_6 = \neg K_6, && S_5 = \neg K_5, && S_4 = \neg K_4,\\
&S_3 = \neg K_3, && S_2 = \neg K_2, && S_1 = \neg K_1,\\
&S_0 = \neg K_0.
\end{aligned}
$$

Регістри R_1 та R_2, в обох випадках, ініціалізуємо нульовими словами.

Далі, аналогічно до «Струмку», виконується 32 раунди ініціалізації(дивись рис.1.4). Під час процедури ініціалізації обчислення

кожного наступного 32–бітного слова, яке має бути вставлене до LFSR, виконується згідно з формулою:

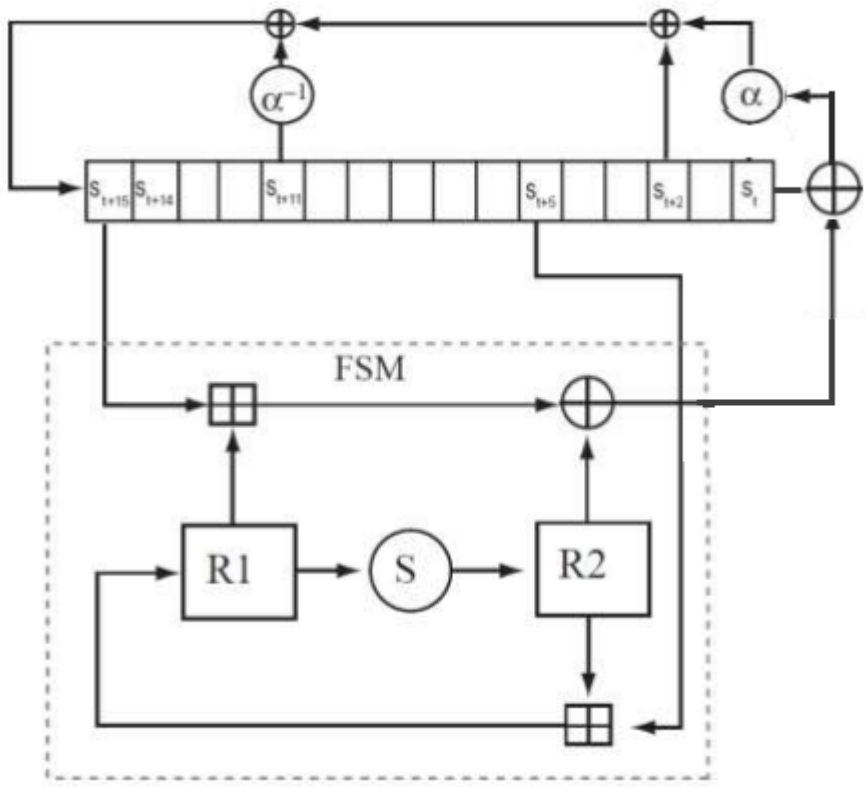

Рисунок 1.4 – Схема роботи шифру «SNOW 2.0» у фазі ініціалізації

$$S_{t+16} = \alpha^{-1} S_{t+11} \oplus S_{t+2} \oplus \alpha S_t \oplus \text{FSMout}, \quad 0 \leqslant t < 32$$

$$\text{FSMout} = (S_{t+15} +_{32} R_{1_t}) \oplus R_{2_t}, \quad t > 0$$

Перед кожною ітерацією шифру відбувається оновлення внутрішнього стану FSM за наступними формулами:

$$R_{1(t+1)} = S_{t+5} +_{32} R_{2t}, \quad R_{2(t+1)} = S(R_{1t}), \quad t > 0.$$

Де S — нелінійна перестановка на $F_{2^{32}}$, яка буде описана в подальшому, а $+_{32}$ додавання за модулем 2^{32}.

Вихідна послідовність при ініціалізації не утворюється. Після цього

виконується ще один додатковий такт без генерації вихідного слова.

$$S_{48} = \alpha^{-1} S_{43} \oplus S_{34} \oplus \alpha S_{32} \oplus \text{FSMout}$$

Функція NEXT:

Вхід: $S(t) = (s(t), r(t)), t > 48$.

Вихід: $S(t+1) = (s(t+1), r(t+1))$ Спочатку виконуємо оновлення внутрішнього стану FSM за наступними формулами:

$$R_{1(t+1)} = S_{t+5} +_{32} R_{2t}, \quad R_{2(t+1)} = S(R_{1t}), \quad t > 0.$$

Значення комірки s_{t+1} обчислюємо за правилом:

$$S_{t+1} = \alpha^{-1} S_{t-4} \oplus S_{t-13} \oplus \alpha S_{t-15} \oplus \text{FSMout}, t > 48$$

Функція Strm:

Функція генерації ключового потоку за станом LFSR, FSM.

Вхід: $S_t(s(t), r(t))$

Вихід: Z_t

$$z_t = \text{FSMout} \oplus S_t, \quad t > 1.$$

Функція нелінійної підстановки S:

На вхід функція отримує 32–бітове значення комірки R_1, яке представлене як вектор $w = \{w_3, w_2, w_1, w_0\}$ кожне слово котрого май розмір 8 бітів.

$$w = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{pmatrix},$$

У векторі w, компонента w_3 відповідає найбільш значущому байту.

Вихід функції — 32–бітове значення комірки $R_2 = \{r_3, r_2, r_1, r_0\}$. Тобто перетворення S — це перестановка на $F_{2^{32}}$

На першому кроці, для кожної компоненти вектору w застосовуємо раундову функцію шифру «Rijndael» [5] SR, що призводить до отримання вектору

$$\overline{SR} = \begin{pmatrix} SR(w_0) \\ SR(w_1) \\ SR(w_2) \\ SR(w_3) \end{pmatrix}$$

У перетворенні $SR(w_i)$ кожне 8–бітове слово (w_i) представлене у вигляді моному y над полем F_{2^8}, яке визначене над незвідним поліномом над полем F_2: $x^8 + x^4 + x^3 + x + 1 \in F_2[x]$. Вектор $\overline{SR}$ можна трактувати як поліном, що визначений у полі F_{2^8}.

На другому кроці, виконується множення за модулем $y^4 + 1 \in F_{2^8}[y]$ поліному $\overline{SR}$ на поліном $c(y)$, який має вигляд:

$$c(y) = (x+1)y^3 + y^2 + y + x \in F_{2^8}[y]$$

Результатом цього множення буде шуканий вектор $R_2 = \{r_3, r_2, r_1, r_0\}$. Виконаємо множення у матричному вигляді:

$$\begin{pmatrix} r_0 \\ r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} x & x+1 & 1 & 1 \\ 1 & x & x+1 & 1 \\ 1 & 1 & x & x+1 \\ x+1 & 1 & 1 & x \end{pmatrix} \cdot \begin{pmatrix} SR(w_0) \\ SR(w_1) \\ SR(w_2) \\ SR(w_3) \end{pmatrix},$$

Повертаємо вектор $R_2 = (r_3, r_2, r_1, r_0)$ як результат роботи функції.

1.3 Особливості побудови потокового шифру «Струмок»

«Струмок» — потоковий шифр, який був представлений у 2016 році в роботі [14]. Він базується на структурі потокового шифру «SNOW 2.0», але призначений для роботи з 64-бітовими словами. У роботі автори рекомендували використовувати 512–бітові та 1024–бітові ключі `K`, а також 256–бітові та 512–бітові вектори ініціалізації `IV`. Але в подальшому в роботі [2] були впроваджені зміни довжини ключа на 256 або 512 бітів, а довжину `IV` залишити лише 256 бітів. Ця варіація шифру була затверджена національним стандартом `ДСТУ`8845 : 2019[18] «Алгоритм симетричного потокового перетворення», і в роботі буде розглянута саме вона.

Потоковий шифр «Струмок» приймає на вхід вектор ініціалізації `IV`, який має довжину 256 бітів, та ключ шифрування `IV`, який може мати довжину 256 або 512 бітів. В залежності від довжини ключа ці шифри поділяються на «Струмок–256» та «Струмок–512», відповідно. На виході алгоритму отримуємо послідовність 64-бітових псевдовипадкових чисел `Z`.

Основними компонентами шифру «Струмок» є регістр зсуву з лінійним зворотнім зв'язком (LFSR — linear feedback shift register) та скінчений автомат (FSM — finite state machine) (дивись рис. 1.5). Скінчений автомат базується на двох 64–бітових регістрах, у ньому виконується нелінійне перетворення T, детальний опис якого наведено в роботі[16]. Регістр зсуву складається з 16 комірок, кожна з яких містить 64–бітове слово. Він побудований за допомогою примітивного над полем $\mathbb{GF}(2^{64})$ полінома $f(x)$

$$f(x) = x^{16} + x^{13} + \alpha^{-1}x^{11} + \alpha$$

Де α є коренем примітивного над полем $\mathbb{GF}(2^8)$ полінома $g(\gamma)$.

$$g(\gamma) = \gamma^8 + \beta^{170}y^7 + \beta^{166}\gamma^6 + \beta^2\gamma^5 + \beta^{224}\gamma^4 + \beta^{70}\gamma^3 + \beta^2$$

Де β, у свою чергу, є коренем примітивного над полем $\mathbb{GF}(2)$ полінома $k(z)$

$$k(z) = z^8 + z^4 + z^3 + z^2 + 1$$

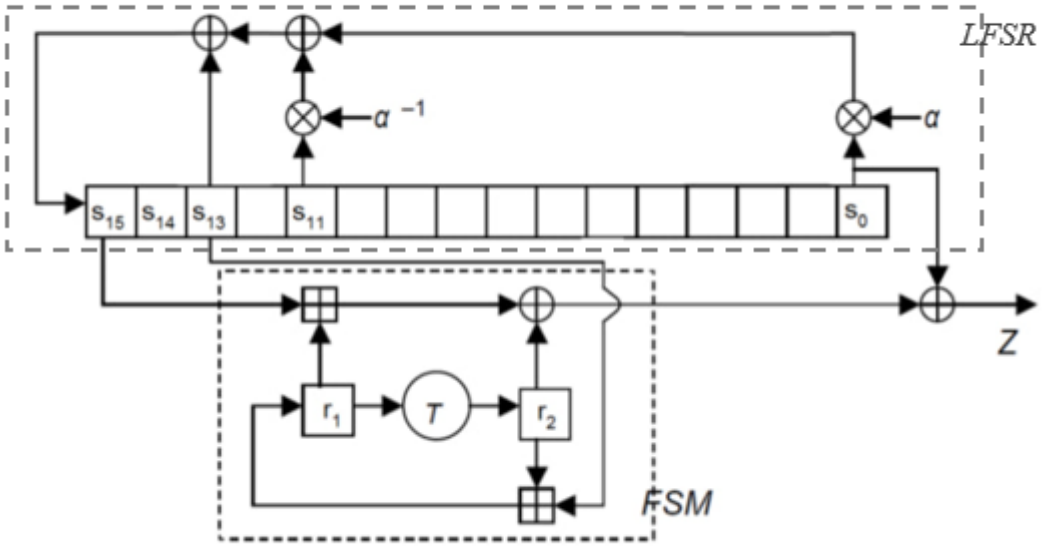

Рисунок 1.5 – Схема роботи шифру Струмок

LFSR генерує послідовність максимального періоду, а саме $2^{1024} - 1$.

Алгоритм «Струмок» можна структурно розділити на такі функції:

1) Кожна з функцій INIT–256, та INIT–512 відповідає за ініціалізацію алгоритму, приймає на вхід вектор ініціалізації IV та ключ шифрування K. Ця функція встановлює початкові значення регістрів та інших змінних, необхідних для правильної роботи алгоритму.

2) Функція Next приймає на вхід поточний стан шифру $S(i) = (s(i), r(i))$, де $s(i)$ представляє стан комірок регістру зсуву, а $r(i)$ - стан регістрів скінченного автомата. Результатом роботи функції є оновлений стан шифру $S(i + 1)$ для наступного моменту часу $i + 1$.

3) Функція ключового потоку STRM приймає на вхід зміну стану $S(i)$ і повертає 64-бітове слово Z_i в якості ключової гами. Використовує

FSM як оракула.

4) Функція скінченного автомата FSM приймає на вхід три 64-бітові змінні. Результатом роботи функції є 64-бітове слово.

5) Функція нелінійної підстановки T реалізує перестановку елементів скінченного поля $\mathbb{GF}(2^{64})$, використовуючи компоненти національного стандарту блокового симетричного шифрування ДСТУ 7624:2014 «Калина». Функція приймає на вхід слово довжиною 64 біти та повертає слово довжиною 64 біти.

Функція INIT–256 для «Струмок–256» :

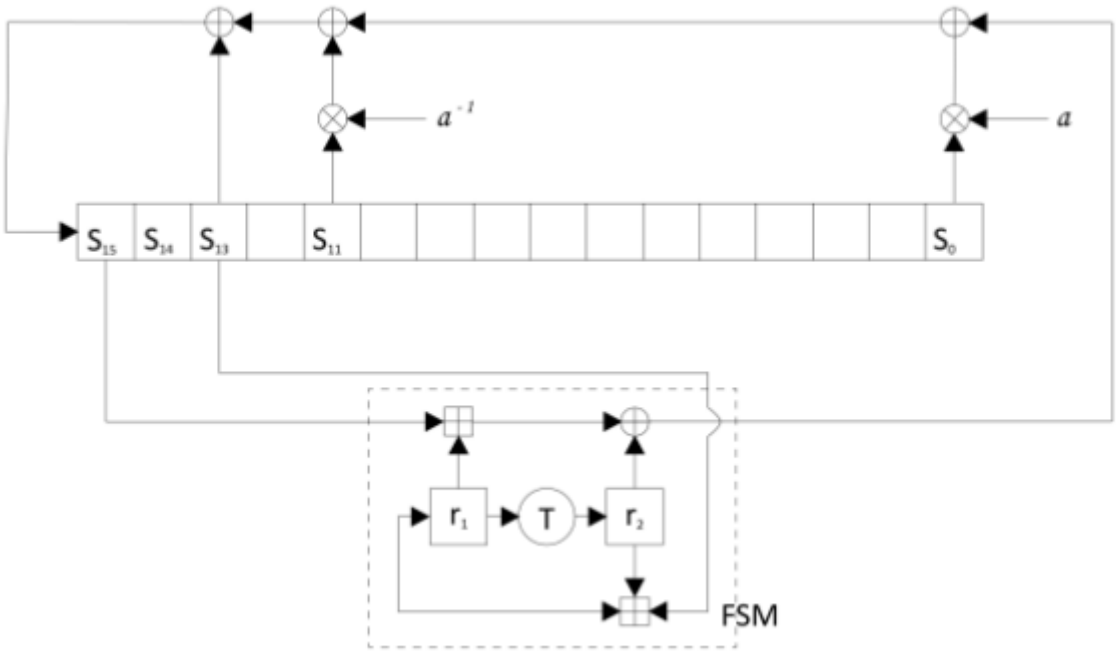

Рисунок 1.6 – Шифр «Струмок» у фазі ініціалізації

Вхід: вектор ініціалізації $IV = \{iv_3, iv_2, iv_1, iv_0\}$, ключ шифрування $K = \{k_3, k_2, k_1, k_0\}$ довжиною 256 бітів. Де довжини кожного слова векторів – 64 біти

Вихід: початковий стан шифру $S^(0) = (s(0), r(0))$.

На першому етапі ініціалізації виконується заповнення стану регістру зсуву з лінійним зворотнім зв'язком (LFSR).

$$s_{15}^{-33} = \neg K_0, \quad s_{14}^{-33} = K_1, \quad s_{13}^{-33} = \neg K_2, \quad s_{12}^{-33} = K_3,$$
$$s_{11}^{-33} = K_0, \quad s_{10}^{-33} = \neg K_1, \quad s_{9}^{-33} = K_2, \quad s_{8}^{-33} = K_3,$$
$$s_{7}^{-33} = \neg K_0, \quad s_{6}^{-33} = \neg K_1, \quad s_{5}^{-33} = K_2 \oplus IV_3, \quad s_{4}^{-33} = K_3,$$
$$s_{3}^{-33} = K_0 \oplus IV_2, \quad s_{2}^{-33} = K_1 \oplus IV_1, \quad s_{1}^{-33} = K_2, \quad s_{0}^{-33} = K_3 \oplus IV_0$$

Регістри r_1 та r_2 ініціалізуємо нульовими словами.

На другому етапі виконуємо 32 такти ініціалізації, застосовуючи процедуру NEXT в режимі ініціалізації:

$$S(-1) = \mathrm{NEXT}_{32}(s^{-33}, r(-33); \mathrm{INIT})$$

Далі, розраховуємо початкове значення стану, використовуючи функцію NEXT в звичайному режимі:

$$S(0) = \mathrm{NEXT}(S(-1))$$

Функція INIT–512 для «Струмок–512»:

Вхід: вектор ініціалізації $IV = \{iv_3, iv_2, iv_1, iv_0\}$, ключ шифрування $K = \{k_7, k_6, k_5, k_4, k_3, k_2, k_1, k_0\}$ довжиною 512 бітів. Де довжини кожного слова векторів – 64 біти

Вихід: початковий стан шифру $S(0) = (s(0), r(0))$.

На першому етапі ініціалізації виконується заповнення стану регістру зсуву з лінійним зворотнім зв'зком (LFSR).

$$s_{15}^{-33} = K_0, \quad s_{14}^{-33} = \neg K_1, \quad s_{13}^{-33} = K_2, \quad s_{12}^{-33} = K_3,$$
$$s_{11}^{-33} = \neg K_7, \quad s_{10}^{-33} = K_5, \quad s_{9}^{-33} = \neg K_6, \quad s_{8}^{-33} = K_4 \oplus IV_3,$$
$$s_{7}^{-33} = \neg K_0, \quad s_{6}^{-33} = K_1, \quad s_{5}^{-33} = K_2 \oplus IV_2, \quad s_{4}^{-33} = K_3,$$
$$s_{3}^{-33} = K_4 \oplus IV_1, \quad s_{2}^{-33} = K_5, \quad s_{1}^{-33} = K_6, \quad s_{0}^{-33} = K_7 \oplus IV_0,$$
$$r_1 = 0, \quad r_2 = 0$$

Слід зазначити, що при обчисленні комірок S_8^{-33} та S_3^{-33}, використовувалась однакова частина вектору K, а саме K_4, та вектори ініціалізації. Це непритаманно для інших SNOW–подібних шифрів.

На другому етапі виконуємо 32 такти ініціалізації, застосовуючи процедуру NEXT в режимі ініціалізації:

$$S(-1) = \mathrm{NEXT}_{32}(s^{-33}, r(-33); \mathrm{INIT})$$

Далі, розраховуємо початкове значення стану, використовуючи функцію NEXT в звичайному режимі:

$$S(0) = \mathrm{NEXT}(S(-1))$$

Тобто ініціалізація шифру «Струмок–512» відрізняється від ініціалізації «Струмок-256» лише правилом початкового заповнення комірок s_i

Функція NEXT:

Вхід: $S(i) = (s(i), r(i))$.

Вихід: $S(i+1) = (s(i+1), r(i+1))$

Перш за все оновлюється значення регістру r_2^{i+1} за формулою:

$$r_2^{i+1} = T(r_1^i)$$

Далі оновлюється значення регістру r_1^{i+1}:

$$r_1^{i+1} = r_2^i + s_{13}^i$$

Значення комірок регістру зсуву оновлюються за правилом:

$$s_j^{i+1} = s_{j+1}^i \quad \text{для } j \in \{0, \ldots, 14\}$$

Обчислюється значення комірки s_15 LFSR за правилом:

$$s_{15}^{i+1} = (s_0^i \oplus \alpha) \oplus (s_{11}^i \oplus \alpha^{-1}) \oplus s_{13}^i$$

якщо було встановлено звичайний режим роботи функції `NEXT`, або за правилом:

$$s_{15}^{i+1} = (s_0^i \oplus \alpha) \oplus (s_{11}^i \oplus \alpha^{-1}) \oplus s_{13}^i \mathrm{FSM}(s_{15}^i, r_1^i, r_2^i)$$

якщо `NEXT` викликано в режимі ініціалізації.

На вихід подається змінна стану $S(i+1) = (s(i+1), r(i+1))$.

Функція Strm:

Вхід: $S_i(s(i), r(i))$

Вихід: Z_i

$$Z_i = FSM(S_i(15), r(i,1), r(i,2)) \oplus s(i,0)$$

.

Функція FSM:

$$\mathrm{FSM}(x,y,z) = ((x+y) \mathrm{mod}\ 2^{64}) \oplus z$$

Функція нелінійної підстановки T:

Вхід: 64-бітове слово w

Вихід: 64-бітове слово q

Функція нелінійної підстановки `T` реалізує перестановку елементів скінченного поля $\mathbb{GF}(2^{64})$ відповідно до національного стандарту блокового симетричного шифрування ДСТУ 7624 : 2014 «Калина».

На першому етапі слово w розбивається на 8 блоків по 8 бітів: $w = (w_7, w_6, w_5, w_4, w_3, w_2, w_1, w_0)$. Потім кожен з блоків замінюється за допомогою чотирьох табличних перестановок $\pi_0, \pi_1, \pi_2, \pi_3$ за правилом:

$$r_j = \pi_j \mod 4[w_j]$$

На другому етапі обчислюємо вектор q, як добуток матриці T на вектор $r = (r_7, r_6, r_5, r_4, r_3, r_2, r_1, r_0)$. Де матриця T представляється як:

$$T = \begin{pmatrix} 01 & 01 & 05 & 01 & 08 & 06 & 07 & 04 \\ 04 & 01 & 01 & 05 & 01 & 08 & 06 & 07 \\ 07 & 04 & 01 & 01 & 05 & 01 & 08 & 06 \\ 06 & 07 & 04 & 01 & 01 & 05 & 01 & 08 \\ 08 & 06 & 07 & 04 & 01 & 01 & 05 & 01 \\ 01 & 08 & 06 & 07 & 04 & 01 & 01 & 05 \\ 05 & 01 & 08 & 06 & 07 & 04 & 01 & 01 \\ 01 & 05 & 01 & 08 & 06 & 07 & 04 & 01 \end{pmatrix}$$

Тобто $q_i = ((01, 01, 05, 01, 08, 06, 07, 04) \ggg i) \cdot r$

Таблиці підстановок та матрицю можна передобчислити заради скорочення часу витраченого на шифрування.

1.4 Алгебраїчна модель шиифру «Струмок»

Розглянемо алгебраїчну модель шифру «Струмок», запропоновану в роботі[12]. Позначення для початкових заповнень, станів, та елементів які наведені в цьому підрозділі будуть використовуватись при подальшому аналізі.

Для позначення елементів множини $\{0,1\}^{64}$ використовуються маленькі латинські літери з можливими індексами, наприклад $s_0^{(-33)}$, а для позначення рядків, що є результатом конкатенації елементів цієї множини, використовуються великі латинські літери з додатковим позначенням, можливо з індексами S^{-33}.

Для значень станів автомату LFSR під час раунду t будемо використовувати позначення $s_{15}^t, \ldots, s_0^t \in \{0,1\}^{64}$; для значень регістрів FSM —- $r_2^t, r_1^t \in \{0,1\}^{64}$; для позначення внутрішнього стану шифру —-

$\hat{W}^t \in \{0,1\}^{1152}$. Внутрішній стан $\hat{W}$ складається зі значень внутрішнього стану та регістрів скінченного автомату: $\hat{W} = s_{15}||\dots||s_0||r_2||r_1$, $s_{15},\dots,s_0,r_2,r_1 \in \{0,1\}^{64}$. Використаємо z^t, $z \in \{0,1\}^{64}$, для означення вихідного потоку алгоритму під час раунду t.

Операції над 64–бітовими комірками $\{0,1\}^{64}$ будуть позначені з використанням наступних символів: $\oplus$ — бінарна операція додавання за модулем 2; $\neg$ — унарну операція побітового заперечення; через $+_{64}$ — бінарна операцію додавання за модулем 2^{64}.

Через $\mathbf{0}$ будемо позначати елемент множини $\{0,1\}^{64}$, який складається з 64 символів 0, відповідно, через $\mathbf{1}$ будемо позначати елемент, який складається з 64 символів 1.

Для шифру Струмок–256 використовується відображення запропоноване в роботі [15] $\mathrm{SetKey}_{256} : \{0,1\}^{256} \times \{0,1\}^{256} \to \{0,1\}^{1152}$. Це відображення використовується для обчислення внутрішнього стану шифру $\hat{W}$, який складається з послідовності бітів $s_{15},\dots,s_0$ та значень регістрів r_2 та r_1. Внутрішній стан $\hat{W}$ обчислюється на основі значень ключа K та вектору ініціалізації IV.

$$\text{SetKey}_{256}(\hat{K}, I\hat{V}) = \hat{W} = s_{15} \parallel \ldots \parallel s_0 \parallel r_2 \parallel r_1,$$

$$s_{15} = \neg k_0, \qquad r_2 = 0,$$
$$s_{14} = k_1, \qquad r_1 = 0,$$
$$s_{13} = \neg k_2,$$
$$s_{12} = k_3,$$
$$s_{11} = k_0,$$
$$s_{10} = \neg k_1,$$
$$s_9 = k_2,$$
$$s_8 = k_3,$$
$$s_7 = \neg k_0,$$
$$s_6 = \neg k_1,$$
$$s_5 = k_2 \oplus iv_3,$$
$$s_4 = k_3,$$
$$s_3 = k_0 \oplus iv_2,$$
$$s_2 = k_1 \oplus iv_1,$$
$$s_1 = k_2,$$
$$s_0 = k_3 \oplus iv_0.$$

Для шифру Струмок–512 використовується відображення $\text{SetKey}_{512} : \{0,1\}^{512} \times \{0,1\}^{256} \to \{0,1\}^{1152}$. Це відображення також використовується для обчислення внутрішнього стану шифру $\hat{W}$ на основі значень ключа K та вектору ініціалізації IV.

$$\begin{aligned}
\mathrm{SetKey}_{512}(\hat{K}, \hat{IV}) = \hat{W^{-33}} = s_{15} \parallel \ldots \parallel s_0 \parallel r_2 \parallel r_1, \\
s_0 &= k_7 \oplus IV_0, & r_1 &= 0, \\
s_1 &= k_6, & r_1 &= 0, \\
s_2 &= k_5, \\
s_3 &= k_4 \oplus IV_1, \\
s_4 &= k_3, \\
s_5 &= k_2 \oplus IV_2, \\
s_6 &= k_1, \\
s_7 &= \neg k_0, \\
s_8 &= k_4 \oplus IV_3, \\
s_9 &= \neg k_6, \\
s_{10} &= k_5, \\
s_{11} &= \neg k_7, \\
s_{12} &= k_3, \\
s_{13} &= k_2, \\
s_{14} &= \neg k_1, \\
s_{15} &= k_0
\end{aligned}$$

За умови, що пройшло менше 32 раундів, тобто $t < 32$, для обчислення наступного внутрішнього стану $\hat{W}^{t+1}$ на кожному такті роботи шифру використовується функція $Next_{INIT} : \{0,1\}^{1152} \to \{0,1\}^{1152}$, яка виконує операції над послідовністю бітів $s_{15}, \ldots, s_0$ та значень регістрів r_2 та r_1.

$$Next_{INIT}(\hat{W}) = s_{16}||s_{15}||\dots||s_1||r_2||r_1, \quad s_{16}, r_2, r_1 \in \{0,1\}^{64},$$
$$s_{16} = s_{13} \oplus (s_{11} \otimes \alpha^{-1}) \oplus (s_0 \otimes \alpha) \oplus r_2 \oplus (r_1 + s_{15}),$$
$$r_{2'} = T(r_1),$$
$$r_{1'} = r_2 + s_{13}$$

За умови, що пройшло більше 32 раундів, тобто $t > 32$, для обчислення наступного внутрішнього стану $\hat{W}^{t+1}$ на кожному такті роботи шифру використовується функція $Next : \{0,1\}^{1152} \to \{0,1\}^{1152}$, яка виконує операції над послідовністю бітів $s_{15}, \dots, s_0$ та значень регістрів r_2 та r_1.

$$Next(\hat{W}) = s_{16}||s_{15}||\dots||s_1||r_2||r_1, \quad s_{16}, r_2, r_1 \in \{0,1\}^{64},$$
$$s_{16} = s_{13} \oplus (s_{11} \otimes \alpha^{-1}) \oplus (s_0 \otimes \alpha),$$
$$r_2 = T(r_1),$$
$$r_1 = r_2 + s_{13}$$

Також, після раундів ініціалізіції, для генерування ключового потоку z^t використовується відображення $Strm : \{0,1\}^{1152} \to \{0,1\}^{64}$

$$Strm(\hat{W}) = s_0 \oplus r_2 \oplus (s_{15} + r_1), \quad s_{15}, \dots, s_0, r_2, r_1 \in \{0,1\}^{64}$$

1.5 Структура потокового шифру SNOW–V

Потоковий шифр «SNOW–V» був розроблений у 2018 році і наразі він є одним з новітніх шифрів сімейства «SNOW». Завдяки значним змінам у будові(дивись рис.1.7), швидкодійність шифру «SNOW–V» значно збільшилась [7] відносно свого попередника «SNOW–3G»[4].

Не зважаючи на ускладнення конструкції схема «SNOW–V» все ще

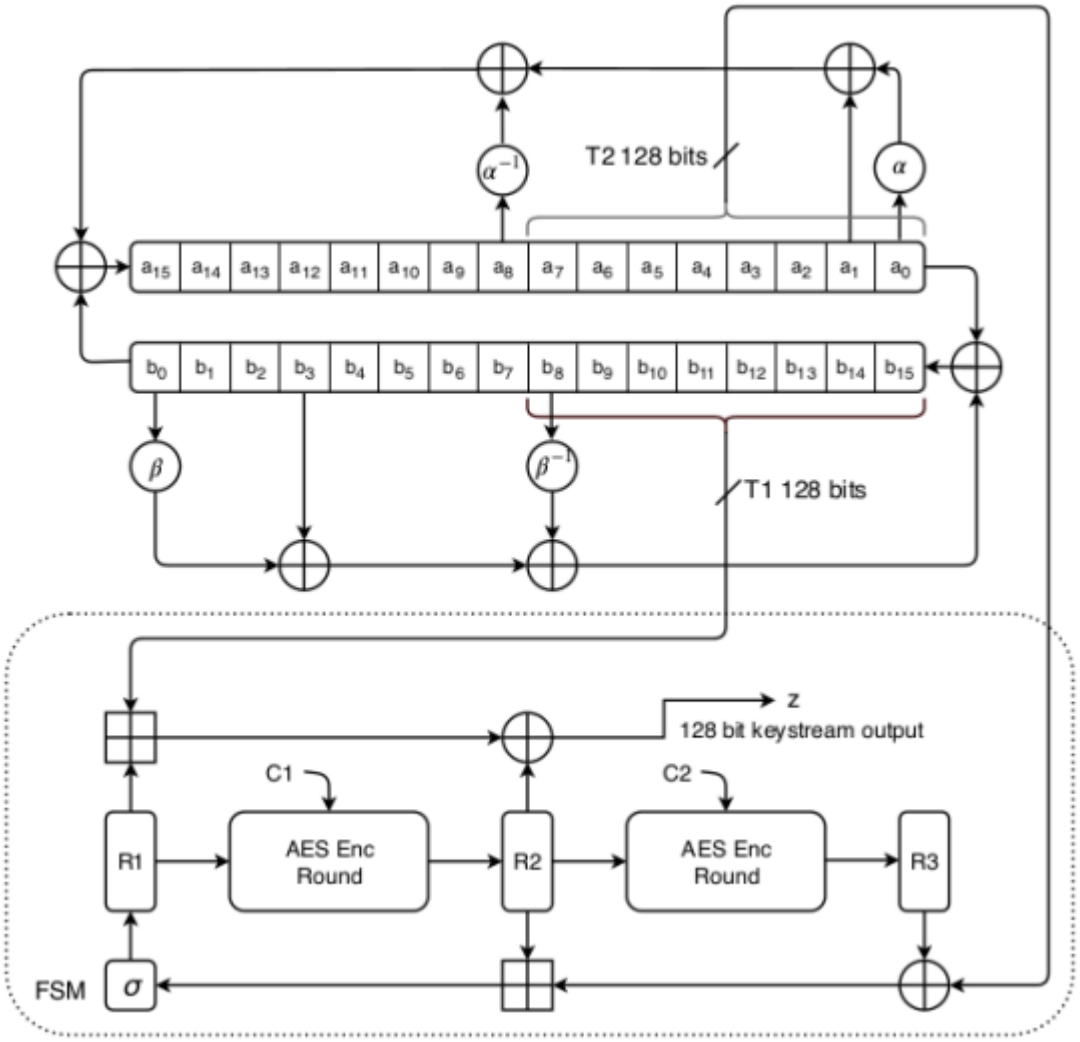

Рисунок 1.7 – Схема роботи шифру «SNOW-V».

поділяється на частини LFSR та FSM.

Будова LFSR змінилась на циклічний апарат, який складається з двох регістрів зсуву з лінійним зворотнім зв'язком «LFSR–A» і «LFSR–B». Кожен з регістрів містить по 16 8–бітових комірок $(a_{15}, \ldots, a_0)$ та $(b_{15}, \ldots, b_0)$, відповідно. Під час роботи алгоритму, слова перемішуються не тільки всередені свого регістру зсуву, але й впливають на створення слов іншого регістру.

Регістри LFSR-A і LFSR-B мають різні породжуючі поліноми.

Регістр LFSR-A:

$$g_A(x) = x^{16} + x^{15} + x^{12} + x^{11} + x^8 + x^3 + x^2 + x + 1, \quad g_A \in \mathbb{F}_2[x]$$

Регістр LFSR-B:

$$g_B(x) = x^{16} + x^{15} + x^{14} + x^{11} + x^8 + x^6 + x^5 + x + 1, \quad g_B \in \mathbb{F}_2[x]$$

Ці поліноми визначають послідовності бітів, що генеруються відповідними регістрами.

Скінченний автомат FSM складається з трьох 128–бітових комірок R_1, R_2, R_3 та двох нелінійних перетворень, які побудовані на основі раундової функції шифрування AES. На вхід FSM отримує блоки з восьми комірок «LFSR–A» і «LFSR–B»: $\{a_0, \ldots, a_7\}$ які впливають на стан R_1 та $\{b_{15}, \ldots, b_8\}$ які впливають на генерацію ключового потоку.

Розгялнемо фазу ініціалізації цього шифру наведену в роботі[9].

На відміну від попередників, «SNOW–V» може приймати на вхід лише 256–бітовий секретний ключ K та 128-б-ітовий вектор ініціалізації IV, $K = \{k_{15}, \ldots, k_0\}$, $IV = \{iv_7, \ldots, iv_0\}$, де кожне значення k_i, $0 \leqslant i \leqslant 15$ та iv_j, $0 \leqslant j \leqslant 7$ є 16–бітним словом.

На першому етапі ініціалізації комірки «LFSR–A» та «LFSR–B» заповнюються за правилом:

Значення комірок a:

$$\begin{aligned}
a_{15} &= k_7, & a_{14} &= k_6, & a_{13} &= k_5 \\
a_{12} &= k_4, & a_{11} &= k_3, & a_{10} &= k_2 \\
a_9 &= k_1, & a_8 &= k_0, & a_7 &= iv_7 \\
a_6 &= iv_6, & a_5 &= iv_5, & a_4 &= iv_4 \\
a_3 &= iv_3, & a_2 &= iv_2, & a_1 &= iv_1 \\
a_0 &= iv_0
\end{aligned}$$

Значення комірок b:

$$
\begin{array}{lll}
b_{15} = k_{15}, & b_{14} = k_{14}, & b_{13} = k_{13} \\
b_{12} = k_{12}, & b_{11} = k_{11}, & b_{10} = k_{10} \\
b_9 = k_9, & b_8 = k_8, & b_7 = 0 \\
b_6 = 0, & b_5 = 0, & b_4 = 0 \\
b_3 = 0, & b_2 = 0, & b_1 = 0 \\
b_0 = 0 & &
\end{array}
$$

На другому єтапі виконується 16 раундів в режимі ініціалізації, в яких аналогічно до інших шифрів сімейства «SNOW», не відбувається генерування ключового потоку, а 128-бітовий вихід z замішується до «LFSR» за допомогою операції XOR. На п'ятнадцятом та шістнадцятом раундах ініціалізації, до регістру R_1 побітово додається значення ключа K.

Висновки до розділу 1

У розділі досліджено будову потокового шифру «Струмок». Розглянуто відмінності фази ініціалізації шифру «Струмок» від шифрів «SNOW 1.0», «SNOW 2.0», «SNOW V». Наведена алгебраїчна модель шифру струмок, застосування якої спростить подальший аналіз.

За структурою «Струмок» найбільш схожий на «SNOW 2.0». Оперування 64–бітовими векторами робить його більш швидким, але зміна фази ініціалізації породжує недоліки:

- Вектор ініціалізації IV використовується не повністю.
- Частина вектору ініціалізації iv_3 не впливає на перші вісім раундів ініціалізації

Атаки, які базуються на використанні цих вразливостей буде

рогзглянуто у другому розділі.

2 АНАЛІЗ ВРАЗЛИВОСТЕЙ ФАЗИ ІНІЦІАЛІЗАЦІЇ ШИФРУ «СТРУМОК–512»

В даному розділі досліджено як відмінності у фазі ініціалізації шифру «Струмок» від шифрів сімейства «SNOW» впливають на криптографічну стійкість шифру до атаки «пов'язаниї ключів», атаки «раундового зсуву», та статистичної атаки на вектор ініціалізації.

2.1 Аналіз ефективності використання вектору ініціалізації у шифру «Струмок–512»

Як можна побачити з попереднього розділу, відображення SetKey_{256} та SetKey_{512} шифру «Струмок» та інших шифрів сімейства «SNOW» відрізняються.

Відмінність «Струмка» від «SNOW 1.0» полягає не тільки в розмірності векторів s_i, k_j, iv_t, r_f, z, яка дорівнює 32 біти, але й в відношенні розмірності вектора IV до розміру комірки. Тобто в «SNOW 1.0» використовувався вектор ініціалізації який поділявся лише на дві частини. Для забеспечення високого впливу вектору ініціалізації на ключовий потік, кількість раундів ініціалізації була вдвійчи більше ніж у «Струмка» та «SNOW 2.0» (64 раунди замість 32).

Оскільки в початковому заповненні комірок LFSR і «Струмок» і «SNOW 2.0» використовують вектор ініціалізації, який поділяється на чотири частини $IV = \{iv_0, iv_1, iv_2, iv_3\}$, можна припустити, що він використовується повністю. Але у відображенні SetKey_{512} «Струмок–512» вектори iv_1 та iv_3 побітово додаються до однакового значення ключа k_4. Це призводить до можливості побудувати бієктивне відображення ϕ_{512}, яке не буде використовувати iv_1 або iv_3 при побудові початкового заповнення W.

Розглянемо відображення неведене в[8] $\phi_{512} : \{0,1\}^{512} \cdot \{0,1\}^{256} \to \{0,1\}^{512} \cdot \{0,1\}^{256}$ вектору W що складається з ключа K сконкатинованого з вектором ініціалізації IV $W = \{k_i, iv_j\}$, де $i \in \{0,\ldots,7\}, j \in \{0,\ldots,3\}$ в вектор W_2 відповідно $W_2 = \{\tilde{k}_i, \tilde{iv}_j\}$, де $i \in \{0,\ldots,7\}, \tilde{iv}_j, j \in \{0,\ldots,3\}$ наступним чином:

$$\begin{array}{lll}
\tilde{k}_0 = k_0 & \tilde{k}_1 = \neg k_1 & \tilde{k}_2 = k_2 \oplus \mathrm{iv}_2 \\
\tilde{k}_3 = k_3 & \tilde{k}_4 = \neg(k_4 \oplus \mathrm{iv}_1) & \tilde{k}_5 = k_5 \\
\tilde{k}_6 = \neg k_6 & \tilde{k}_7 = k_7 \oplus \mathrm{iv}_0 & \tilde{\mathrm{iv}}_0 = k_4 \oplus k_7 \oplus \mathrm{iv}_0 \oplus \mathrm{iv}_3 \\
\tilde{\mathrm{iv}}_1 = k_4 \oplus k_7 \oplus \mathrm{iv}_1 & \tilde{\mathrm{iv}}_2 = \mathrm{iv}_2 & \tilde{\mathrm{iv}}_3 = \mathrm{iv}_1
\end{array}$$

Твердження 2.1. *Відображення* $\phi_{512} : \{0,1\}^{512} \cdot \{0,1\}^{256} \to \{0,1\}^{512} \times \{0,1\}^{256}$ *є бієкцією.*

Доведення. Побудуємо зворотнє відображення ϕ_{512}^{-1}:

$$\begin{array}{lll}
k_0 = \tilde{k}_0 & k_1 = \neg\tilde{k}_1 & k_2 = \tilde{k}_2 \oplus \tilde{\mathrm{iv}}_2 \\
k_3 = \tilde{k}_3 & k_4 = \neg(\tilde{k}_4 \oplus \tilde{\mathrm{iv}}_3) & k_5 = \tilde{k}_5 \\
k_6 = \neg\tilde{k}_6 & k_7 = \neg(\tilde{k}_7 \oplus \tilde{\mathrm{iv}}_1) & \mathrm{iv}_0 = \neg(\tilde{k}_4 \oplus \tilde{k}_7 \oplus \tilde{\mathrm{iv}}_1) \\
\mathrm{iv}_1 = \tilde{\mathrm{iv}}_3 & \mathrm{iv}_2 = \tilde{\mathrm{iv}}_2 & \mathrm{iv}_3 = \neg(\tilde{k}_4 \oplus \tilde{k}_7 \oplus \tilde{\mathrm{iv}}_3 \oplus \tilde{\mathrm{iv}}_0)
\end{array}$$

Перевіримо правильність побудови зворотнього відображення, підставивши його у початкові рівняння:

$$s_0 = \tilde{k}_7 = k_7 \oplus \tilde{\text{iv}}_0 \qquad s_1 = \tilde{k}_6 \oplus 1 = k_6 \qquad s_2 = \tilde{k}_5 = k_5$$
$$s_3 = \tilde{k}_4 \oplus 1 = k_4 \oplus \tilde{\text{iv}}_1 \qquad s_4 = \tilde{k}_3 = k_3 \qquad s_5 = \tilde{k}_2 = k_2 \oplus \tilde{\text{iv}}_2$$
$$s_6 = \tilde{k}_1 \oplus 1 = k_1 \qquad s_7 = \tilde{k}_0 \oplus 1 = \neg k_0 \qquad s_8 = \tilde{k}_7 \oplus \tilde{\text{iv}}_0 = k_4 \oplus \tilde{\text{iv}}_3$$
$$s_9 = \tilde{k}_6 = \neg k_6 \qquad s_{10} = \tilde{k}_5 = k_5 \qquad s_{11} = \tilde{k}_4 \oplus \tilde{\text{iv}}_1 = \tilde{\text{iv}}_1 \oplus k_4$$
$$s_{12} = \tilde{k}_3 = k_3 \qquad s_{13} = \tilde{k}_2 \oplus \tilde{\text{iv}}_2 = k_2 \qquad s_{14} = \tilde{k}_1 = \neg k_1$$
$$s_{15} = \tilde{k}_0 = k_0 \qquad r_1 = 0 \qquad r_2 = 0$$

□

Знайдемо вигляд початкового заповненння автомату LFSR шифру «Струмок–512», після використання бієктивного відображення ϕ_{512}:

$$\text{SetKey}_{512}(W_2)$$

$$s_0(-33) = k_7 \qquad s_1(-33) = \neg k_6 \qquad s_2(-33) = k_5$$
$$s_3(-33) = \neg k_4 \qquad s_4(-33) = k_3 \qquad s_5(-33) = k_2$$
$$s_6(-33) = \neg k_1 \qquad s_7(-33) = \neg k_0 \qquad s_8(-33) = k_7 \oplus \text{iv}_0$$
$$s_9(-33) = k_6 \qquad s_{10}(-33) = k_5 \qquad s_{11}(-33) = k_4 \oplus \text{iv}_1$$
$$s_{12}(-33) = k_3 \qquad s_{13}(-33) = k_2 \oplus \text{iv}_2 \qquad s_{14}(-33) = k_1$$
$$s_{15}(-33) = k_0 \qquad r_1(-33) = 0 \qquad r_2(-33) = 0$$

Наслідок 2.1. *Для кожного значення* $c \in \{0,1\}^{64}$ *та будь яких ключів* $K \in \{0,1\}^{512}$ *і векторів ініціалізації* $IV \in \{0,1\}^{256}$*, де* $K = k_7||\ldots||k_0$ *і* $IV = iv_3||\ldots||iv_0$*, можемо побудувати відповідність:*

$$SetKey_{512}(K, IV) = SetKey_{512}(K_2, IV_2)$$

Де $K_2 = k_7||\ldots||k_5||(k_4 \oplus c)||k_3||\ldots||k_0$ *і* $IV_2 = iv_3 \oplus c||iv_2||iv_1 \oplus c||iv_0$.

При аналізі бієктивного відображення ϕ_{512} було виявлено[13], що 64 – бітнове слово iv_4 з вектору ініціалізації IV може бути прибрано з алгоритму без втрати будь яких властивостей. Тобто ефективна довжина IV дорівнює 192 біти замість заявлених 256.

З наслідку 2.1 випливає, що для кожної пари ключа K довжиною 512 бітів та вектору ініціалізації IV, можна підібрати пару з ключа K_2 довжиною 448 (комірка $k_4 = \{0\}^{64}$) бітів та вектора ініціалізації iv_2 які б після ініціалізації видавали б той самий ключовий потік. Ключі K, K_2 називаються «пов'язаними ключами», а атака, що їх використовє відповідно називається «атака пов'язаних ключів».

Тобто шифр «Струмок–512» має вразливість до атаки «пов'язаних ключів». Детальніше про цю атаку в роботі[20].

2.2 Перевірка стійкості шифру «Струмок–512» до атаки раундового зсуву

Під час порівняння фази ініціалізації шифру «Струмок–512», з фазою ініціалізації шифру «SNOW 2.0», були виявлені відмінності у правилах початкового заповнення комірок $s_i, i \in \{0, \ldots, 15\}$. Оскільки атака раундового зсуву, є ефективно застосовною до шифру «SNOW 2.0» [21]доцільно перевірити вразливість «Струмка» від неї.

Атака раундового зсуву з кількістю раундів $r \in \mathbb{N}_1$, полягає[10] у побудові системи рівнянь між $s_i^{(-33+r)}$, де $i \in \{0, \ldots, 15\}$ до умов початкового заповнення $s_i^{(-33)}$, для деякого значення ключа K та вектору ініціалізації IV. Вирішення цієї системи дозволяє відкинути з розглядання перші r раундів ініціалізіції. Детальніше про побудову атаки на можна причитати у роботі[19].

Необхідною умовою для проведення атаки раундового зсуву з кількістю раундів $r \in \mathbb{N}_1$ є вирішуваність наведеної вище системи для деякого значення ключа K та вектору ініціалізації IV.

Перевіримо виконання цієї умови для атак раундового зсуву, з кількістю ранундів від одного до семи, стосовно шифру «Струмок–512». Не має сенсу перевіряти атаки з кількістю ранудів більше ніж сім, бо вони будуть не простішими за повний перебір.

Твердження 2.2. *Шифр «Струмок–512» є стійким до до атаки раундового зсуву з кількістю раундів від одного до сьоми.*

Доведення.

Розглянемо $r = 1$:

Тоді $\tilde{s}_9^{(-33)} = \tilde{k}_6$ і $\tilde{s}_1^{(-33)} = \tilde{k}_6$, але $\tilde{s}_9^{(-33)} = S_9^{(-33+1)} = S_{10}^{(-33)} = k_5$ і $\tilde{s}_1^{(-33)} = S_1^{(-33+1)} = S_2^{(-33)} = k_5$.

Отримали $\tilde{k}_6 = k_5$ і $\tilde{k}_6 = \neg k_5$ з чого слідує, що неможливо знайти певні значення k_5 і $\tilde{k}_6$ які задовольняють обидва рівняння одночасно. Це означає, що атака на «Струмок–512» з одним раундом зсуву не може бути застосовною, оскільки вона потребує таких значень, які не існують.

Розглянемо $r = 2$:

Тоді $\tilde{s}_4(-33) = \tilde{k}_3$ і $\tilde{s}_{12}(-33) = \tilde{k}_3$, але $\tilde{s}_4(-33) = s_4(-33 + 2) = s_6(-33) = \neg k_1$ і $\tilde{s}_{12}(-33) = s_{12}(-33 + 2) = s_{14}(-33) = k_1$,

Отримали $\tilde{k}_3 = \neg k_1$ і $\tilde{k}_3 = k_1$з чого слідує, що неможливо знайти певні значення k_1 і $\tilde{k}_3$, які задовольняють обидва рівняння одночасно. Це означає, що атака на шифр з двома раундами зсуву не може бути застосовною, оскільки вона потребує таких значень, які не існують.

Розглянемо $r = 3$:

Тоді $\tilde{s}_4(-33) = \tilde{k}_3$ і $\tilde{s}_{12}(-33) = \tilde{k}_3$, але $\tilde{s}_4(-33) = s_4(-33 + 3) = s_7(-33) = \neg k_0$ і $\tilde{s}_{12}(-33) = s_{12}(-33 + 3) = s_{15}(-33) = k_0$,

Отримали $\tilde{k}_3 = \neg k_0$ і $\tilde{k}_3 = k_0$з чого слідує, що неможливо знайти певні значення k_0 і $\tilde{k}_3$, які задовольняють обидва рівняння одночасно. Це означає, що атака на шифр з трьома раундами зсуву не може бути застосовною.

Розглянемо $r = 4$:

Тоді $\tilde{s}_2(-33) = \tilde{k}_5$ і $\tilde{s}_{10}(-33) = \tilde{k}_5$, але $\tilde{s}_2(-33) = s_2(-33 + 4) = s_6(-33) = k_1$ і $\tilde{s}_{10}(-33) = s_2(-33 + 4) = s_{14}(-33) = \neg k_1$,

Отримали $\tilde{k}_5 = \neg k_1$ і $\tilde{k}_5 = k_1$з чого слідує, що неможливо знайти певні значення k_1 і $\tilde{k}_5$, які задовольняють обидва рівняння одночасно. Це означає, що атака на шифр з чотирьма раундами зсуву не може бути застосовною.

Розглянемо $r = 5$:

Тоді $\tilde{s}_2(-33) = \tilde{k}_5$ і $\tilde{s}_{10}(-33) = \tilde{k}_5$, але $\tilde{s}_2(-33) = s_2(-33 + 5) = s_7(-33) = \neg k_0$ і $\tilde{s}_{10}(-33) = s_2(-33 + 5) = s_{15}(-33) = k_0$,

Отримали $\tilde{k}_5 = \neg k_0$ і $\tilde{k}_5 = k_0$, з чого слідує, що неможливо знайти певні значення k_0 і $\tilde{k}_5$, які задовольняють обидва рівняння одночасно. Це означає, що атака на шифр з пь'ятьма раундами зсуву не може бути застосовною.

Розглянемо $r = 6$:

Тоді $\tilde{s}_4(-33) = \tilde{k}_3$ і $\tilde{s}_{12}(-33) = \tilde{k}_3$ але $\tilde{s}_4(-33) = s_4(-33+6) = s_{10}(-33) = k_5$ і $\tilde{s}_{12}(-33) = s_{12}(-33+6) = s_2(-33) = k_5$

Отримали $\tilde{k}_3 = k_5$ і $\tilde{k}_3 = k_5$з чого слідує, що неможливо знайти певні значення k_5 і $\tilde{k}_3$, які задовольняють обидва рівняння одночасно. Це означає, що атака на шифр з шістьма раундами зсуву не може бути застосовною.

Розглянемо $r = 7$:

Тоді $\tilde{s}_2(-33) = \tilde{k}_5$ і $\tilde{s}_{10}(-33) = \tilde{k}_5$, але $\tilde{s}_2(-33) = s_2(-33 + 7) = s_9(-33) = \neg k_6$ і $\tilde{s}_{10}(-33) = s_2(-33 + 7) = s_1(-33) = k_6$,

Отримали $\tilde{k}_5 = \neg k_6$ і $\tilde{k}_5 = k_6$з чого слідує, що неможливо знайти певні значення k_6 і $\tilde{k}_5$, які задовольняють обидва рівняння одночасно. Це означає, що атака на шифр з сім'ю раундами зсуву не може бути застосовною. □

Фаза ініціалізації шифру «Струмок–512» побудована таким чином, що шифр є стійким до до атаки раундового зсуву. Атаки з кількостю раундів меншою за вісім не можуть бути застосовані до нього. Виявлено, що на стійкість до атаки впливає разташування операцій заперечення у

функції SetKey_{512}.

Тобто, для будь якого r необхідною умовою того, що система рівнянь $s_i^{(-33+r)} = s_i^{(-33)}$ буде невирішуваною, є наявність комірок

$$s_i^{-33} = k_j,\ s_{(i+xr)mod16}^{-33} = \neg k_j,\ \text{де } i \in \{0,\ldots,15\},\ j \in \{0,\ldots,7\},\ x \in \text{N}$$

При побудові нової фази ініціалізаціїї потрібно виконати умову необхідості, та змінювати розташування операцій заперечення, якщо застосування атаки раундового зсуву буде можливим.

2.3 Дослідження статистичних атак на вектор ініціалізації

У шифрі «Струмок–512» значення iv_3 з вектору IV впливає лише на заповнення комірки $s_8^{(-33)}$, у чому можна переконатись з відображення SetKey_{512}.

$$s_8 = k_4 \oplus IV_3$$

Під час фази ініціалізації, на створювання нових значень які записуються в регістр зсуву з лінійним зворотнім зв'язком впливають комірки $s_0, s_{11}, s_{13}, r_1, r_2$, та значення α.

$$s_{15}^{i+1} = (s_0^i \oplus \alpha) \oplus (s_{11}^i \oplus \alpha^{-1}) \oplus s_{13}^i \text{FSM}(s_{15}^i, r_1^i, r_2^i)$$

Вектор iv_3 не впливає на перші вісім раундів ініціалізації, оскільки значення $s_8^{(-33)}$ буде вперше застосовано функцією NEXT для генерування s_{15}^{-33+8}. Наслідком цього є вразливысть шифру «Струмок–512» до статистичних атак які встановлюють залежність між ключовим потоком та початковим $W^{(-33)}$, оскільки вектор ініціалізації не встигає достатньо перемішатись із ключем, після ініціціалізації.

Для статистичних атаки на вектор ініціалізації зазвичай використовуються детектори випадковості послідовності[11]. Завдяки ним можливо відрізнити ключовий потік від випадкової послідовності.

Існує декілька статистичних способів відстежити невипадковість ключового потоку:

1) Тест перевірки наявності мономів розміру d

2) Узагальнений підхід перевірки наявності мономів розміру d

3) MDM(maximum degree monomial) підхід

Детальніше про кожен:

1)Відстежити невипадковість ключового потоку можливо за кількістю входжень мономів розміру d у функцію δ, що записана в алгебраїчній нормальній формі, d–мономів можна наступним чином:

Фіксуємо $c = |IV| - n$ значень вектору ініціалізації IV як константу, де n – обране натуральне число. Далі будуємо функцію $\delta : |K| \times \{0,1\}^n \to |z|^{64}$ з ключа та розглядаємої частини вектору ініціалізації у ключовий потік. Записуєм цю функцію у алгебраїчній нормальній формі. Підраховуємо кількість мономів ваги d, які входять до АНФ. Порівнюємо отриману кількість з математичним очікуванням за допомогою критерію χ^2–згоди з одним ступенем свободи.

$$\chi^2 = \sum_{k=0}^{n} \left(\frac{m_k - \frac{C_k^n}{2}}{\sqrt{\frac{C_k^n}{2}}} \right)^2 \xrightarrow{d} \chi^2_{n+1, C_k^n} \xrightarrow{d} \infty$$

Оскільки обрати n бітів можливо 2^n різними способами, а для побудови полної залежності потрібно n^2 операцій, то складність становить $O(n^2 \cdot 2^n)$ операцій і вимагає $O(n^2 \cdot 2^n)$ бітів пам'яті.

2) Узагальнений підхід відрізняється від попереднього тесту тим, що будується не одна, а P різних функцій δ. $\delta_1, \delta_2, ..., \delta_P$, де для кожної пари функцій є хоча б одне не спільне вхіде значення iv_i. Далі, для кожної з цих функцій виконується статистичний тест перевірки наявності мономів розміру d.

Складність та затрати пам'ті відповідно зростають у P разів і дорівнюють $O(n^2 \cdot 2^n \cdot P)$

3)Для виконання MDM (від англ. maximum degree monomial) тесту,

запропонованого в роботі[3], потрібно обрати простір B з розмірністю b як конкатенацію секретного ключа K та вектору ініціалізації IV. $b = |K|+|IV|$. Згенеруємо 2^b різних вхідних значень та виконаємо ініціалізацію шифру для кожного значення. Потім потрібно обчислити значення функції f яке задане як

$$f = \biguplus_{y\in\{0,1\}^{|K|}} \bigoplus_{x\in\{0,1\}^{|IV|}} z_0(x,y)$$

Де $\biguplus$ – конкатенація, $\bigoplus$ – додавання за модулем 2, $z_0(x,y)$ – перший біт ключового потоку при при $K = y, IV = x$. Послідовність f називається MDM–сигнатурою, бо це коефіцієнти перед мономами максимального степеня.

Оскільки моном максимального степеня може бути присутній у АНФ функції з ймовірністю 1/2, то нулі та одиниці в MDM-сигнатурі мають з'являтися з однаковою ймовірністю. Тобто MDM–сигнатура має виглядати як випадкова послідовність, в ідеальному шифрі.

Складність тесту становить $O(2^b)$ операцій.

Існує вдосконалення цього способу, за рахунок зменшення простору B до його підмножини S, $S \subset, B$, де елементи що входять в множину S обираються жадімним алгоритм. Усі елементи, які залишилися у множині $B \setminus S$, вважаються рівними константі. Але слід зауважити, що від вибору підмножини S залежить релевантність результатів тесту. Детальніше про реалізацію цього методу можна прочитати в роботі [17].

2.4 Побудова модифікації «Струмок–512–32» та перевірка її стійкості до статистичних атак

В попередньому підрозділі була зазначена можливість вразливості у фазі ініціалізації шифру «Струмок–512», яка полягає в слабкому впливу вектору iv_3 на вихідний потік z.

Оскільки статистичні тести вимагають значних обчислювальних

потужностей потоковий шифр «Струмок–512» був змінений. Модифікація шифру буде використовуватись під назвою «Струмок–512–32».

Ідея цієї модифікації полягає в збереженні структури шифру «Струмок–512» та правил початкового заповнення комірок LFSR, але зменшення бітового розміру усіх векторів, які використовуються в алгоритмі. Це призводить до можливості екстраполяції висновків що до стійкості «Струмок–512–32» на «Струмок–512».

Структура «Струмок–512—32»:

Вдвічи зменшений бітовий розмір комірок $s_i, i \in \{0,\dots,15\, r_1, r_2$. Наразі вони дорівнюють 32 біти.

Зменшен розмір векторів $k_i, i \in \{0,\dots,7\, iv_j, j \in \{0,\dots,3\}$ вони дорівнюють 32 біти. Відповідно змінилась довжина ключа K та вектору ініціалізації IV. Однак ці веккотри все ще поділяються на чотири та вісім частин, відповідно: $IV = \{iv_3, iv_2, iv_1, iv_0\}$, $K = \{k_7, k_6, k_5, k_4, k_3, k_2, k_1, k_0\}$.

Функція заповнення стартових комірок

$$\text{SetKey}_{512\text{–}32} : \{0{,}1\}^{256} \times \{0{,}1\}^{128} \to \{0{,}1\}^{576}$$

Функція $\text{NEXT}_{\text{INIT}}$ змінюється на:

$$s_j^{i+1} = s_{j+1}^i \quad \text{для } j \in \{0,\dots,14\}$$

$$s_{15}^{i+1} = (s_0^i) \oplus (s_{11}^i) \oplus s_{13}^i \text{FSM}(s_{15}^i, r_1^i, r_2^i)$$

Функція NEXT змінюється на:

$$s_j^{i+1} = s_{j+1}^i \quad \text{для } j \in \{0,\dots,14\}$$

$$s_{15}^{i+1} = (s_0^i) \oplus (s_{11}^i) \oplus s_{13}^i$$

Функція оновлення станів скінченного автомату залишається тією ж самою що й у не модифікованому шифрі $r_2^{i+1} = T(r_1^i)$, $r_1^{i+1} = r_2^i + s_{13}^i$, але змінюється функція нелінійної перестановки T. Вона реалізує перестановку елементів скінченного поля $\mathbb{GF}(2^{32})$. Тобто, вектор r_1 розбивається на вісім

блоків по чотори біти, а не по вісім бітів.

Тест перевірки кількості мономів розміру 5 **в шифрі «Струмок–512—32»:**

Для виконання статистичного тесту на перевірку кількості мономів розміру d, потрібно обрати d з проміжку натуральних числел від одного, до довжини найбільшого можливого моному, не включно. Оберемо $d = 5$.

Побудуємо функції $\delta_i : W^{-33} \to z_{1_i}, i \in \{0, \ldots, 3\}$, де 120 значень вектору IV зафіксовано як константний нуль. Запис z_{1_i} означає, що розглядається i–тий байт вектору z_1. $\delta_i : |K| \times |IV| \to |z_{1_i}|$. Незафіксованими лишаються перші вісім бітів вектору iv_3. За допомогою S – блоків побудуємо АНФ функцій δ_i. Вони досить громіздкі, тому у додаток3.3 перенесено лише приклад АНФ δ_0.

Обчислимо очікувану кількість мономів розмірністю $d = 5$. Існує C_8^5 різних 5–мономів, імновірність зустріти кожен з яких у отриманій АНФ 0.5. Математичне очікування входження моному степеня 5, до одної АНФ дорівнює $\frac{C_8^5}{2} = 28$. Тоді у 32 АНФ очікувана кількість 5–мономім буде $E = 896$.

Підраховуємо фактичну кількість 5–мономів, $O = 844$

Проводимо χ^2 тест, та отримємо p–значення $= 0.073468$

Це означає, що гіпотеза: кожен з 5–мономім входіть в АНФ з імовірнюстю 0.5 є істиною з імовірністю $p = 0.073468$.

Тобто фаза ініціалізації «Струмок–512—32» та відповідно і «Струмок–512» вразлива до статистичних атак на вектор ініціалізації.

Висновки до розділу 2

В другому розділі досліджені атаки, що використовують вразливості фази ініціалізації.

Виявлено, що використання не повного вектору ініціалізації призводить до можливості застосування атаки пов'язаних ключів.

Перевірка стійкості модифікації «Струмок–512–32» до статистичних

атак на вектор ініціалізації показала, що відсутність використання вектору iv_3 на перших раундах ініціалізації зменшує його вплив на ключовий потік та породжує вразливості до статистичних атак.

Перевірка стійкості шифру «Струмок–512» до атак раундового зсуву з кількістю раундів від одного до сьоми показала показала, що «Струмок–512» захищен від цієї атаки. При будуванні вдосконаленої фази ініціалізації потрібно зберегти цю властивість.

3 ПРОПОЗИЦІЯ ПОКРАЩЕННОЇ ФАЗИ ІНІЦІАЛІЗАЦІЇ ШИФРУ «СТРУМОК–512»

В цьому розділі буде запропоноване початкове заповнення $Set_{new}(K,IV)$ замість $Set(K,IV)$ в шифрі «Струмок–512». Його застосування має позитивний вплив на безпеку шифру, оскільки зменшує його вразливість від статистичних атак на вектор ініціалізації, позбавляється вразливості до атак пов'язаних ключів, а також зберігає стійкість від атак раундових зсувів з кількістю ранудів від одного до сьоми, включно.

3.1 Побудова покращеної фази ініціалізації шифру «Струмок-512»

Зважаючи на наведені в попередньому розділі вразливості фази ініціалізації шифру «Струмок-512» пропонується змінити правила початкового заповнення комірок $s_0, \ldots, s_15$ таким чином, що б:

• Вектор ініціалізації IV використовувся повністю.

• Кожна частина вектора ініціалізаціїiv_0, iv_1, iv_2, iv_3 мала вплив на створюване значення s на як можливо ранішому раунді ініціалізації.

• Зберілась властивість стійкості до атаки раундового зсуву щонайменше до семи раундів.

• Вектори $k_0, \ldots, k_7$ були впорядковані за зростанням.

Побудуємо наступне початкове заповнення, яке назвемо $Set_{new}(K,IV)$:

$$
\begin{array}{llll}
s_0^{-33} = K_0 & s_4^{-33} = K_4 & s_8^{-33} = K_0 & s_{12}^{-33} = \neg K_4 \\
s_1^{-33} = K_1 \oplus IV_0 & s_5^{-33} = K_5 & s_9^{-33} = K_1 & s_{13}^{-33} = K_5 \oplus IV_3 \\
s_2^{-33} = K_2 \oplus IV_1 & s_6^{-33} = K_6 & s_{10}^{-33} = K_2 & s_{14}^{-33} = \neg K_6 \\
s_3^{-33} = K_3 \oplus IV_2 & s_7^{-33} = K_7 & s_{11}^{-33} = K_3 & s_{15}^{-33} = \neg K_7 \\
r_1 = \{0\}^{64} & r_2 = \{0\}^{64} & &
\end{array}
$$

Оскільки була застосовано кожна з частин вектору ініціалізації iv_0, iv_1, iv_2, iv_3 та не існє жодного значення k_i, яке б використовувалось в двох різних операціях $k_i \oplus iv_j$, то початкове заповнення $Set_{new}(K,IV)$ повністю використовує вектор ініціалізації $IV = (iv_0, iv_1, iv_2, iv_3)$.

Тобто не існує значення $c \in \{0,1\}^{64}$, яке б для довільних ключів K і векторів ініціалізації IV, задовольняло відповідності:

$$\text{SetKey}_{512}(K, IV) = \text{SetKey}_{512}(K_2, IV_2)$$

Де $K = k_7||\ldots||k_0$ і $IV = iv_3||\ldots||iv_0$, $K_2 = k_7||\ldots||k_5||(k_4 \oplus c)||k_3||\ldots||k_0$ і $IV_2 = iv_3 \oplus c||iv_2||iv_1 \oplus c||iv_0$.

З цього випливає, що $Set_{new}(K,IV)$ не має вразливості до атаки пов'язаних ключів.

3.2 Перверіка стійкості шифру «Струмок–512» з новою фазою ініціалізації до статистичних атак

У шифру «Струмок–512» з початковим заповненням W_{new}^{-33} виконаним функцією $Set_{new}(K,IV)$ кожна частина вектора ініціалізації iv_0, iv_1, iv_2, iv_3 має вплив на створюване значення s на початкових раундах ініціалізації.

Застосувавши до W_{new}^{-33} функцію $NEXT_{INIT}$ легко побачити, що iv_0 має вплив на створюване s на першому рануді, iv_1 на другому раунді, iv_2

на третьому раунді, iv_3 — на нуловому.

Після проходження 16 раундів ініціалізації при оригінальному заповненні, вектор ініціалізації впливає на стан восьми комірок s. Тоді як при заповненні W_{new}^{-33}, після 16 раундів ініціалізації, десять комірок s будуть створені під впливом вектору iv_i. Тому у заповненні W_{new}^{-33} кожен з векторів iv_0, iv_1, iv_2, iv_3 має більший вплив на створювані слова s, ніж у оригінальному заповненні.

Виконаємо статистичний тест на кількість входження d–мономів але з використанням функції $Set_{new}(K,IV)$. Оскільки статистичні тести вимагають значних обчислювальних потужностей, аналогічно до тесту описаному в розділі[2.4] скористаємось модифікацією «Струмок–512–32».

Покращення результатів тесту буде свідчити про те, що зміна фази ініціалізації призведе до збільшення стійкості не лише модифікації «Струмок–512–32», а й шифру «Струмок–512».

Побудуємо функції $\delta_{new_i} : W_{new}^{-33} \to z_{1_i}, i \in \{0, \ldots, 3\}$, де 120 значень вектору IV зафіксовано як константний нуль. Запис z_{1_i} означає, що розглядається біт і вектору z_1. $\delta_{new_i} : W_{new}^{-33} \to |z_{1_i}|$. Незафіксованими лишаються перші вісім бітів вектору iv_3. За допомогою S – блоків побудуємо АНФ функцій δ_{new_i}. Приклад δ_{new_i} наведено у додатку3.3.

Як було обчислено в розділі[2.4], очікувана кількість 5–мономім у 32 АНФ буде $E = 896$.

Підраховуємо фактичну кількість 5–мономів, $O_{new} = 912$

Проводимо χ^2 тест, та отримємо p–значення $= 0.596242$

Це означає, що гіпотеза: кожен з 5–мономім входіть в АНФ з імовірнюстю 0.5 є істиною з імовірністю $p_{new} = 0.596242$.

Порівнюючи результат з попереднім тестом бачимо, що залежність між вектором iv_3 та вихідним потоком зменшилась.

3.3 Перверіка стійкості шифру «Струмок–512», з новою фазою ініціалізації, до атаки раундового зсуву

Як було доведено в попередньому підрозділі, початкове заповнення $Set_{new}(K,IV)$ вирішує деякі проблеми фази ініціалізації, проте важливо переконатись, що це не приведе до інших, не менш значущих вразливостей. Атака, що найбільш тісно пов'язана з початковим заповненням – атака раундового зсуву. Мета атакуючого – побудувати систему з шістнадцяти рівнянь, яка має рішення, де в лівій частині рівнянь буде значеняя комірок $s_i^{(-33)}, i \in \{0,\dots,15\}$, а в правій – значення тих самих комірок після r раундів ініціалізації $s_i^{(-33+r)}, i \in \{0,\dots,15\}$. Слід зазначити, що вектор ініціалізації IV, для атаки раундового зсуву, вважається обраним зловмисником, а ключ K — довільним.

Перевіримо виконання цієї умови для атак раундового зсуву, з кількістю ранундів від одного до сьоми, стосовно шифру «Струмок–512» з початковим заповненням $Set_{new}(K,IV)$.

Твердження 3.1. *Атака раундого зсуву з кількістю раундів $r \in \{1,\dots,7\}$ не застосовна до шифру «Струмок–512» з початковим заповненням $Set_{new}(K,IV)$.*

Доведення. При r = 1:

Розглянемо пару $s_0^{-33} = k_0, \quad s_8^{-33} = k_0$.

Тоді обрахуємо значення комірок після зсуву:

$$s_0^{-33+1} = s_1^{-33} = k_1 \oplus iv_0, \quad s_8^{(-33+1)} = s_9^{-33} = k_1$$

Рівність $k_1 \oplus iv_0 = k_1$ винконується лише за умови $iv_0 = 0^{64}$.

Якщо $iv_0 = 0^{64}$, то $s_1^{-33} = k_1 \oplus iv_0 = k_1, \quad s_9^{-33} = k_1$.

Але $s_1^{-33+1} = s_2^{-33} = k_2 \oplus iv_1, \quad s_9^{(-33+1)} = s_{10}^{-33} = k_2$.

Рівність $k_2 \oplus iv_1 = k_2$ винконується лише за умови $iv_1 = 0^{64}$.

Якщо $iv_1 = 0^{64}$, то $s_2^{-33} = k_2 \oplus iv_1 = k_2, \quad s_{10}^{-33} = k_2$

Але $s_2^{-33+1} = s_3^{-33} = k_3 \oplus iv_2, \quad s_{10}^{(-33+1)} = s_{11}^{-33} = k_3.$

Рівність $k_3 \oplus iv_2 = k_3$ винконується лише за умови $iv_2 = 0^{64}$.

Нарешті, якщо $iv_2 = 0^{64}$, то $s_3^{-33} = k_3 \oplus iv_2 = k_3, \quad s_{11}^{-33} = k_3$

Але $s_3^{-33+1} = s_4^{-33} = k_4, \quad s_{11}^{(-33+1)} = s_{12}^{-33} = \neg k_4.$

Отримали $k_4 = \neg k_4$ з чого слідує, що неможливо знайти певне значення k_4, яке б задовольняють обидва рівняння одночасно. Це означає, що атака на «Струмок–512», з початковим заповненням $Set_{new}(K,IV)$, з одним раундом зсуву не може бути застосовною, оскільки побудована система рівнянь — не вирішувана.

При r = 2:

Розглянемо пару $s_0^{-33} = k_0, \quad s_8^{-33} = k_0.$

Тоді обрахуємо значення комірок після зсуву:

$$s_0^{-33+2} = s_2^{-33} = k_2 \oplus iv_1, \quad s_8^{(-33+2)} = s_{10}^{-33} = k_2$$

.

Рівність $k_2 \oplus iv_1 = k_2$ винконується лише за умови $iv_1 = 0^{64}$.

Якщо $iv_1 = 0^{64}$, то $s_2^{-33} = k_2 \oplus iv_1 = k_2, \quad s_{10}^{-33} = k_2$

Але $s_2^{-33+2} = s_4^{-33} = k_4, \quad s_{10}^{(-33+2)} = s_{12}^{-33} = \neg k_4.$

Отримали $k_4 = \neg k_4$ з чого слідує, що неможливо знайти певне значення k_4, яке б задовольняють обидва рівняння одночасно. Це означає, що атака на «Струмок–512», з початковим заповненням $Set_{new}(K,IV)$, з двома раундами зсуву не може бути застосовною, оскільки побудована система рівнянь — не вирішувана.

При r = 3:

Розглянемо пару $s_0^{-33} = k_0, \quad s_8^{-33} = k_0.$

Тоді обрахуємо значення комірок після зсуву:

$$s_0^{-33+3} = s_3^{-33} = k_3 \oplus iv_2, \quad s_8^{(-33+3)} = s_{11}^{-33} = k_3$$

.

Рівність $k_3 \oplus iv_2 = k_3$ винконується лише за умови $iv_2 = 0^{64}$.

Якщо $iv_2 = 0^{64}$, то $s_3^{-33} = k_3 \oplus iv_2 = k_3, \quad s_{11}^{-33} = k_3,$

Але $s_3^{-33+3} = s_6^{-33} = k_6, \quad s_{11}^{(-33+3)} = s_{14}^{-33} = \neg k_6.$

Отримали $k_6 = \neg k_6$ з чого слідує, що неможливо знайти певне значення k_6, яке б задовольняють обидва рівняння одночасно. Це означає, що атака на «Струмок–512», з початковим заповненням $Set_{new}(K,IV)$, з трьома раундами зсуву не може бути застосовною, оскільки побудована система рівнянь — не вирішувана.

При r = 4:

Розглянемо пару $s_0^{-33} = k_0, \quad s_8^{-33} = k_0.$

Тоді обрахуємо значення комірок після зсуву:

$$s_0^{-33+4} = s_3^{-33} = k_4, \quad s_8^{(-33+4)} = s_{12}^{-33} = \neg k_4$$

.

Отримали $k_4 = \neg k_4$ з чого слідує, що неможливо знайти певне значення k_4, яке б задовольняють обидва рівняння одночасно. Це означає, що атака на «Струмок–512», з початковим заповненням $Set_{new}(K,IV)$, з чотирьма раундами зсуву не може бути застосовною, оскільки побудована система рівнянь — не вирішувана.

При r = 5:

Розглянемо пару $s_0^{-33} = k_0, \quad s_8^{-33} = k_0.$

Тоді обрахуємо значення комірок після зсуву:

$$s_0^{-33+5} = s_5^{-33} = k_5, \quad s_8^{(-33+5)} = s_{13}^{-33} = k_5 \oplus iv_3$$

.

Рівність $k_5 \oplus iv_3 = k_5$ винконується лише за умови $iv_3 = 0^{64}$.

Якщо $iv_3 = 0^{64}$, то $s_5^{-33} = k_5, \quad s_{13}^{-33} = k_5 \oplus iv_2 = k_5$

Але

$s_5^{-33+5} = s_{10}^{-33} = k_2, \quad s_{13}^{(-33+5)} = \alpha \otimes s_2^{-33} \oplus \alpha^{-1} \otimes s_{13}^{-33} \oplus FSM_{out} \oplus s_{15}^{-33} = \neg k_2 \oplus iv_1.$

Рівність $k_2 \oplus iv_1 = k_2$ винконується лише за умови $iv_1 = 0^{64}$.

Якщо $iv_1 = 0^{64}$, то $s_2^{-33} = k_2 \oplus iv_1 = k_2, \quad s_{10}^{-33} = k_2$

Але $s_2^{-33+5} = s_7^{-33} = k_7, \quad s_{10}^{(-33+5)} = s_{15}^{-33} = \neg k_7$.

Отримали $k_7 = \neg k_7$ з чого слідує, що неможливо знайти певне значення k_7, яке б задовольняють обидва рівняння одночасно. Це означає, що атака на «Струмок–512», з початковим заповненням $Set_{new}(K,IV)$, з п'ятьма раундами зсуву не може бути застосовною, оскільки побудована система рівнянь — не вирішувана.

При r = 6:

Розглянемо пару $s_0^{-33} = k_0, \quad s_8^{-33} = k_0$.

Тоді обрахуємо значення комірок після зсуву:

$$s_0^{-33+6} = s_6^{-33} = k_6, \quad s_8^{(-33+6)} = s_{14}^{-33} = \neg k_6$$

.

Отримали $k_6 = \neg k_6$ з чого слідує, що неможливо знайти певне значення k_6, яке б задовольняють обидва рівняння одночасно. Це означає, що атака на «Струмок–512», з початковим заповненням $Set_{new}(K,IV)$, з шістьма раундами зсуву не може бути застосовною, оскільки побудована система рівнянь — не вирішувана.

При r = 7:

Розглянемо пару $s_0^{-33} = k_0, \quad s_8^{-33} = k_0$.

Тоді обрахуємо значення комірок після зсуву:

$$s_0^{-33+7} = s_7^{-33} = k_7, \quad s_8^{(-33+7)} = s_{15}^{-33} = \neg k_7$$

.

Отримали $k_7 = \neg k_7$ з чого слідує, що неможливо знайти певне значення k_7, яке б задовольняють обидва рівняння одночасно. Це означає, що атака на «Струмок–512», з початковим заповненням $Set_{new}(K,IV)$, з сім'ю раундами зсуву не може бути застосовною, оскільки побудована система рівнянь — не вирішувана. □

Висновки до розділу 3

В третьому розділі запропоновано використання функції $Set_{new}(K,IV)$ замість $Set(K,IV)$ у фазі ініціалізації шифру «Струмок–512». Доведено що ця зміна підвищить криптографічну стійкість шифру «Струмок–512». Наведено будову цієї функції та досліджено її вплив на криптографічну стійкість шифру.

Доведено, що функція $Set_{new}(K,IV)$ на відміну від $Set(K,IV)$ не має вразливості до атаки пов'язаних ключів, бо вдосконалена фаза ініціалізації шифру «Струмок–512» $Set_{new}(K,IV)$ побудована таким чином, що не існує пари векторів iv_0, iv_1, iv_2, iv_3 які додаються до однакового значення $k_i, i \in \{0,\ldots,7\}$

Крім того доведено, що запропоноване початкове заповнення $Set_{new}(K,IV)$ менш вразливе до статистичної атаки d–мономів, при $d = 5$. Отримали, що ключовий потік шифру «Струмок–512–32» з початковим заповненням $Set_{new}(K,IV)$ є більш схожим на випадковий, ніж ключовий потік шифру «Струмок–512–32» з початковим заповненням $Set(K,IV)$.

Побудовано доведення незастосовності атак раундового зсуву з кількістю раундів $r = 1,\ldots,7$ до запропонованої вдосконаленої фази ініціалізації, оскільки не існує таких значеннь k, які б задовольняли усі рівняння з системи $s_i^{(-33)} = s_i^{(-33+r)}, i \in \{0,\ldots,15\}$ одночасно.

ВИСНОВКИ

В роботі проведено дослідження структури потокового шифру «Струмок» та шифрів «SNOW 1.0», «SNOW 2.0», «SNOW V». Проаналізовані особливості фази ініціалізації шифру «Струмок–256» та шифру «Струмок–256».

В ході аналізу виявлено, що в фазі ініціалізації шифру «Струмок–512» додавання частин вектора ініціалізації iv_3, iv_1 до однієї частини ключа k_4, при початковому заповненні комірок s_8 та s_3 регістру зсуву з лінійним зворотнім зв'язком, призводить до вразливості шифру «Струмок–512» до атаки пов'язаних ключів. Окрім того, частина вектора ініціалізації iv_3 починає впливати на генерування ключового потоку лише на восьмому раунді. Ця вразливість призводить до можливості побудови статистичних атак.

1) Побудовано модифікацію шифру «Струмок–512» під назвою «Струмок–512–32», яка спрощує аналіз фази ініціалізації, шляхом зменшення розміру внутрішнього стану LFSR. Застосовано модифікацію «Струмок–512–32» для аналізу вразливості шифру до статистичної атаки d–мономів, при $d = 5$.

2) Побудована вдосконалена фаза ініціалізації шифру «Струмок–512» з метою підвищення криптографічної стійкості шифру. Відмінність від оригінальної фази ініціалізації полягає в зміні правила початкового заповнення $Set(K,IV)$ на $Set_{new}(K,IV)$ в якому вектор ініціалізації $IV = \{iv_0, iv_1, iv_2, iv_3\}$ додається до комірок s_1, s_2, s_3, s_{13} відповідно. Це призводить до того, що кожен з векторів iv починає впливати на ключовий потік не пізніше четвертого раунду. Порівняно стійкість модифікації «Струмок–512–32» до статистичної атаки d–мономів, при $d = 5$ з початковими заповненнями $Set(K,IV)$ та $Set_{new}(K,IV)$. Доведено, що початкове заповнення $Set_{new}(K,IV)$ я більш стійким до цієї атаки.

3) Доведено незастосовність атак раундового зсуву з кількістю раундів $r = 1, \ldots, 7$ до запропонованої вдосконаленої фази ініціалізації, оскільки не існує таких значень k, які б задовольняли усі рівняння з системи $s_i^{(-33)} = s_i^{(-33+r)}, i \in \{0, \ldots, 15\}$ одночасно. Це означає, що побудована система рівнянь — не вирішувана, а вдосконалена фаза ініціалізації є стійкою до цієї атаки.

4) Доведено, що атака пов'язаних ключів не є застосовною до запропонованої вдосконаленої фази ініціалізації шифру «Струмок–512», бо $Set_{new}(K,IV)$ побудована таким чином, що не існує пари векторів iv_0, iv_1, iv_2, iv_3 які додаються до однакового значення $k_i, i \in \{0, \ldots, 7\}$

В подальшому планується перевірити стійкість запропонованої вдосконаленої фази ініціалізації шифру «Струмок–512» до інших відомих атак на фазу ініціалізації.

ПЕРЕЛІК ПОСИЛАНЬ

[1] Kircanski A. та Youssef A. *On the sliding property of SNOW 3G and SNOW 2.0.* IET Information Security, 2011.

[2] A. N. Alekseychuk and. *Security Justification for Strumok Stream Cipher Against Correlation Attacks Over Finite Fields of Characteristic 2.* 19. Kamianets-Podilskyi National Ivan Ohiienko University, черв. 2019, с. 114—119. DOI: 10.32626/2308-5916.2019-19.114-119.

[3] Catherine Berbain та Henri Gilbert. *On the security of IV dependent stream ciphers.* За ред. Alex Biryukov. Т. 4593. Springer, Heidelberg, 2007, с. 254—273.

[4] Alex Biryukov та Christophe de Canniere. *Block ciphers and systems of quadratic equations.* 2003.

[5] Joan Daemen та Vincent Rijmen. *The Design of Rijndael.* Springer Berlin Heidelberg, 2002. DOI: 10.1007/978-3-662-04722-4.

[6] P. Ekdahl та T. Johansson. *SNOW. A new stream cipher.* Proceedings of the First NESSIE Workshop., 2000.

[7] Patrik Ekdahl та ін. *A new SNOW stream cipher called SNOW-V.* Universitatsbibliothek der Ruhr-Universitat Bochum, вер. 2019, с. 1—42. DOI: 10.46586/tosc.v2019.i3.1-42.

[8] I. Gorbenko та ін. *Strumok Keystream Generator.* Kyiv, Ukraine, трав. 2018, с. 292—299.

[9] I. Gorbenko та ін. *The research of modern stream ciphers.* Kharkiv, 2017, с. 207—210.

[10] I. Gorbenko та ін. *Потоковий шифр «Струмок».* Т. 9. 1. 2018, с. 4—14.

[11] I.D. Gorbenko та ін. *Methods of Information Protection in Communications Systems and Methods of Their Cryptoanalysis*. Т. 52. 4. 1998.

[12] Ivan Gorbenko та ін. *Strumok keystream generator*. IEEE, трав. 2018. DOI: 10.1109/dessert.2018.8409147.

[13] A. Kuznetsov та ін. *Analysis of block symmetric algorithms from international standard of lightweight cryptography ISO/IEC 29192-2*. Kharkiv, 2017, с. 203—206.

[14] O. Kuznetsov та D. Lutsenko M.and Ivanenko. *Strumok stream cipher: Specification and basic properties*. Англ. 2016.

[15] V. Martin. *Algebraic attack on stream ciphers*. 2007.

[16] Roman Oliynykov, Ivan Gorbenko, Oleksandr Kazymyrov та ін. *A New Encryption Standard of Ukraine: The Kalyna Block Cipher*. 2015/650. 2015. URL: https://eprint.iacr.org/2015/650.

[17] А.Н. Алексейчук, С.Н. Конюшок та А.Ю. Сторожук. *Обобщенная статистическая атака на синхронные поточные шифры*. Т. 17. 3. 2015, с. 54—65.

[18] *ДСТУ 8845:2019 Інформаційні технології. Криптографічний захист інформації. Алгоритм симетричного потокового перетворення*. 2019.

[19] О.О. Кузнецов та ін. *Математична структура потокового шифру струмок*.

[20] А.М. Олексійчук. *Достатня умова стійкості SNOW 2.0-подібних потокових шифрів відносно певних атак зі зв'язаними ключами*. Т. 18. 3. 2016, с. 261—268.

[21] А.М. Олексійчук та М.В. Поремський. *Загальна схема побудови кореляційних атак на SNOW 2.0-подібні потокові шифри*. Т. 1. 35. 2018, с. 70—79.

ДОДАТОК А ОТРИМАНІ АНФ

Пприклад АНФ δ_0

$$\delta_0 = x_0x_1x_2x_3x_4x_5x_6 \oplus x_0x_1x_2x_3x_4x_5x_7 \oplus x_0x_1x_2x_3x_4x_6x_7 \oplus$$
$$\oplus x_0x_1x_2x_3x_4x_6 \oplus x_0x_1x_2x_3x_4x_7 \oplus x_0x_1x_2x_3x_5x_6x_7 \oplus x_0x_1x_2x_3x_5 \oplus$$
$$\oplus x_0x_1x_2x_3x_6x_7 \oplus x_0x_1x_2x_3x_7 \oplus x_0x_1x_2x_4x_5x_6x_7 \oplus x_0x_1x_2x_4x_6x_7 \oplus$$
$$\oplus x_0x_1x_2x_4x_6 \oplus x_0x_1x_2x_4x_7 \oplus x_0x_1x_2x_5x_6x_7 \oplus x_0x_1x_2x_5x_7 \oplus$$
$$\oplus x_0x_1x_2x_5 \oplus x_0x_1x_2x_6 \oplus x_0x_1x_2x_7 \oplus x_0x_1x_3x_4x_5x_6 \oplus$$
$$\oplus x_0x_1x_3x_4x_5 \oplus x_0x_1x_3x_5x_6 \oplus x_0x_1x_3x_5x_7 \oplus x_0x_1x_3x_5 \oplus$$
$$\oplus x_0x_1x_3x_6 \oplus x_0x_1x_3x_7 \oplus x_0x_1x_3 \oplus x_0x_1x_4x_5x_6x_7 \oplus$$
$$\oplus x_0x_1x_4x_5x_6 \oplus x_0x_1x_4x_6x_7 \oplus x_0x_1x_4x_6 \oplus x_0x_1x_4x_7 \oplus$$
$$\oplus x_0x_1x_5x_6x_7 \oplus x_0x_1x_6 \oplus x_0x_1 \oplus x_0x_2x_3x_4x_5x_6x_7 \oplus$$
$$\oplus x_0x_2x_3x_4x_5x_6 \oplus x_0x_2x_3x_4x_6x_7 \oplus x_0x_2x_3x_4x_7 \oplus x_0x_2x_3x_4 \oplus$$
$$\oplus x_0x_2x_3x_6x_7 \oplus x_0x_2x_3 \oplus x_0x_2x_4x_5x_6x_7 \oplus x_0x_2x_4x_5x_6 \oplus$$
$$\oplus x_0x_2x_4x_5x_7 \oplus x_0x_2x_4x_6x_7 \oplus x_0x_2x_5x_6 \oplus x_0x_2x_6 \oplus$$
$$\oplus x_0x_2x_7 \oplus x_0x_3x_4x_5x_7 \oplus x_0x_3x_4x_5 \oplus x_0x_3x_4x_6x_7 \oplus x_0x_3x_4 \oplus$$
$$\oplus x_0x_3x_6x_7 \oplus x_0x_3x_7 \oplus x_0x_4x_5x_7 \oplus x_0x_4 \oplus x_0x_5x_6x_7 \oplus$$
$$\oplus x_0x_5 \oplus x_0x_6x_7 \oplus x_0x_6 \oplus x_0 \oplus x_1x_2x_3x_4x_5x_7 \oplus x_1x_2x_3x_4 \oplus$$
$$\oplus x_1x_2x_3x_5x_7 \oplus x_1x_2x_3x_6x_7 \oplus x_1x_2x_3x_7 \oplus x_1x_2x_3 \oplus$$
$$\oplus x_1x_2x_4x_5x_7 \oplus x_1x_2x_4x_7 \oplus x_1x_2x_5x_6x_7 \oplus x_1x_2x_5 \oplus$$
$$\oplus x_1x_2x_6x_7 \oplus x_1x_2x_7 \oplus x_1x_2 \oplus x_1x_3x_4x_5x_7 \oplus x_1x_3x_4x_5 \oplus$$
$$\oplus x_1x_3x_4x_6 \oplus x_1x_3x_4x_7 \oplus x_1x_3x_4 \oplus x_1x_3x_5x_6 \oplus x_1x_3x_5x_7 \oplus$$
$$\oplus x_1x_3x_6x_7 \oplus x_1x_4x_5x_6x_7 \oplus x_1x_4x_5x_7 \oplus x_1x_4x_5 \oplus x_1x_5x_6 \oplus$$
$$\oplus x_1x_6 \oplus x_1 \oplus x_2x_3x_4x_5x_6x_7 \oplus x_2x_3x_4x_5x_6 \oplus x_2x_3x_4x_5x_7 \oplus$$
$$\oplus x_2x_3x_4x_5 \oplus x_2x_3x_4x_6 \oplus x_2x_3x_4x_7 \oplus x_2x_3x_5x_6 \oplus x_2x_3x_6x_7 \oplus$$
$$\oplus x_2x_4x_5x_6x_7 \oplus x_2x_4x_6x_7 \oplus x_2x_4x_6 \oplus x_2x_4x_7 \oplus x_2x_5x_6x_7 \oplus$$
$$\oplus x_2x_5x_7 \oplus x_2x_5 \oplus x_2x_6x_7 \oplus x_2x_6 \oplus x_2x_7 \oplus x_2 \oplus x_3x_4x_5x_7 \oplus x_3x_4x_6 \oplus$$
$$\oplus x_3x_4x_7 \oplus x_3x_5x_6x_7 \oplus x_3x_5x_7 \oplus x_3x_5 \oplus x_3x_7 \oplus x_4x_5x_6x_7 \oplus$$
$$\oplus x_4x_5x_7 \oplus x_4x_6 \oplus x_5x_7 \oplus x_5 \oplus x_7 \oplus 1$$

Приклад АНФ δ_{new_0}

$$\delta_{new_0} = x_0x_1x_2x_3x_4x_5x_6x_7 \oplus x_0x_1x_2x_3x_4x_5x_6 \oplus x_0x_1x_2x_3x_4x_6 \oplus$$

$$\oplus\, x_0x_1x_2x_3x_4x_7 \oplus x_0x_1x_2x_3x_5x_6 \oplus x_0x_1x_2x_3x_5 \oplus x_0x_1x_2x_3x_6x_7 \oplus$$
$$\oplus\, x_0x_1x_2x_4x_5x_7 \oplus x_0x_1x_2x_4x_5 \oplus x_0x_1x_2x_4x_6x_7 \oplus x_0x_1x_2x_4x_7 \oplus$$
$$\oplus\, x_0x_1x_2x_4 \oplus x_0x_1x_2x_5x_6 \oplus x_0x_1x_2x_5 \oplus x_0x_1x_2x_6x_7 \oplus$$
$$\oplus\, x_0x_1x_2 \oplus x_0x_1x_3x_4x_5x_6x_7 \oplus x_0x_1x_3x_4x_5x_7 \oplus x_0x_1x_3x_4x_5 \oplus$$
$$\oplus\, x_0x_1x_3x_4 \oplus x_0x_1x_3x_5x_6x_7 \oplus x_0x_1x_3x_5x_6 \oplus x_0x_1x_3x_7 \oplus$$
$$\oplus\, x_0x_1x_4x_5x_6 \oplus x_0x_1x_4x_5 \oplus x_0x_1x_4x_6 \oplus x_0x_1x_4x_7 \oplus$$
$$\oplus\, x_0x_1x_4 \oplus x_0x_1x_5x_6x_7 \oplus x_0x_1x_5x_6 \oplus x_0x_1x_5 \oplus x_0x_1x_7 \oplus$$
$$\oplus\, x_0x_2x_3x_4x_5x_6x_7 \oplus x_0x_2x_3x_4x_5x_6 \oplus x_0x_2x_3x_4x_5x_7 \oplus x_0x_2x_3x_4 \oplus$$
$$\oplus\, x_0x_2x_3x_5x_6x_7 \oplus x_0x_2x_3x_6x_7 \oplus x_0x_2x_3x_6 \oplus x_0x_2x_4x_5x_6x_7 \oplus$$
$$\oplus\, x_0x_2x_4x_5x_7 \oplus x_0x_2x_4x_6x_7 \oplus x_0x_2x_5x_6 \oplus x_0x_2x_7 \oplus x_0x_2 \oplus$$
$$\oplus\, x_0x_3x_4x_5x_6 \oplus x_0x_3x_4x_6x_7 \oplus x_0x_3x_4x_7 \oplus x_0x_3x_5 \oplus x_0x_3 \oplus$$
$$\oplus\, x_0x_4x_5x_7 \oplus x_0x_4x_6x_7 \oplus x_0x_4x_6 \oplus x_0x_7 \oplus x_0 \oplus$$
$$\oplus\, x_1x_2x_3x_4x_6x_7 \oplus x_1x_2x_3x_4x_7 \oplus x_1x_2x_3x_5x_6x_7 \oplus x_1x_2x_3x_5x_7 \oplus$$
$$\oplus\, x_1x_2x_3x_5 \oplus x_1x_2x_3x_6 \oplus x_1x_2x_3x_7 \oplus x_1x_2x_3 \oplus$$
$$\oplus\, x_1x_2x_4x_5x_6x_7 \oplus x_1x_2x_4x_5x_6 \oplus x_1x_2x_4x_5x_7 \oplus x_1x_2x_4x_6x_7 \oplus$$
$$\oplus\, x_1x_2x_4x_6 \oplus x_1x_2x_4x_7 \oplus x_1x_2x_4 \oplus x_1x_2x_5x_7 \oplus$$
$$\oplus\, x_1x_2x_5 \oplus x_1x_2x_6 \oplus x_1x_2 \oplus x_1x_3x_4x_5x_6x_7 \oplus x_1x_3x_4x_5x_6 \oplus$$
$$\oplus\, x_1x_3x_4x_5x_7 \oplus x_1x_3x_4x_5 \oplus x_1x_3x_4x_6x_7 \oplus x_1x_3x_4x_6 \oplus$$
$$\oplus\, x_1x_3x_4x_7 \oplus x_1x_3x_5x_6x_7 \oplus x_1x_3x_5x_6 \oplus x_1x_3x_5 \oplus x_1x_3x_6 \oplus$$
$$\oplus\, x_1x_4x_5x_6x_7 \oplus x_1x_4x_5x_7 \oplus x_1x_4x_6 \oplus x_1x_4x_7 \oplus x_1x_6 \oplus x_1 \oplus$$
$$\oplus\, x_2x_3x_4x_5x_6x_7 \oplus x_2x_3x_4x_5x_7 \oplus x_2x_3x_4x_6 \oplus x_2x_3x_5x_6x_7 \oplus$$
$$\oplus\, x_2x_3x_5x_7 \oplus x_2x_3x_6x_7 \oplus x_2x_3x_7 \oplus x_2x_3 \oplus x_2x_4x_5x_6x_7 \oplus x_2x_4x_5x_6 \oplus$$
$$\oplus\, x_2x_4x_5x_7 \oplus x_2x_4x_6 \oplus x_2x_4x_7 \oplus x_2x_4 \oplus x_2x_5x_6x_7 \oplus x_2x_5x_6 \oplus x_2x_5 \oplus$$
$$\oplus\, x_2x_6x_7 \oplus x_2x_6 \oplus x_3x_4x_5 \oplus x_3x_4x_6x_7 \oplus x_3x_5x_6x_7 \oplus x_3x_5x_6 \oplus x_3x_5x_7 \oplus$$
$$\oplus\, x_3x_5 \oplus x_4x_5x_7 \oplus x_4x_6x_7 \oplus x_4x_6 \oplus x_4x_7 \oplus x_4 \oplus x_5x_6x_7 \oplus x_5 \oplus x_6 \oplus x_7 \oplus 1$$