27. *Tickoo M.L*. Simplification: Theory and Application. Anthology Series 31, *ERIC Clearinghouse,* 1993, pp. 254.
28. Big-O, *Big-O.io*. Available at: https://big-o.io (accessed 03 July 2024).
29. Complexity Cheat Sheet for Python Operations, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/complexity-cheat-sheet-for-python-operations/ (accessed 03 July 2024).

Статью рекомендовал к опубликованию д.т.н. В.Ф. Лубенцов.

Герасименко Евгения Михайловна – Южный федеральный университет; e-mail: egerasimenko@sfedu.ru; г. Таганрог, Россия; тел.: 88634371651; кафедра систем автоматизированного проектирования им В.М. Курейчика; доцент.

Кравченко Юрий Алексеевич – e-mail: yakravchenko@sfedu.ru; тел.: 88634371651; кафедра систем автоматизированного проектирования им В.М. Курейчика; д.т.н.; профессор.

Шаненко Дарья Андреевна – e-mail: shanenko@sfedu.ru; тел.: 88634371651; кафедра систем автоматизированного проектирования им В.М. Курейчика, ассистент.

Gerasimenko Evgeniya Mihailovna – Southern Federal University; e-mail: egerasimenko@sfedu.ru; Taganrog, Russia; phone: +78634371651; the Department of Computer Aided Design named after V.M. Kureychik; associate professor.

Kravchenko Yury Alekseevich – e-mail: yakravchenko@sfedu.ru; phone: +78634371651; the Department of Computer Aided Design named after V.M. Kureychik; dr. of eng. sc.; professor.

Shanenko Daria Andreevna – e-mail: shanenko@sfedu.ru; phone: +78634371651; the Department of Computer Aided Design named after V.M. Kureychik; assistant.

УДК 004.056.55 DOI 10.18522/2311-3103-2024-5-102-118

С.В. Поликарпов, В.А. Прудников, К.Е. Румянцев

СИНТЕЗ ПСЕВДО-ДИНАМИЧЕСКИХ ФУНКЦИЙ PD-sbox-ARX-32

Целью работы является разработка метода синтеза оптимальных псевдо-динамических функций PD-sbox-ARX-32, размерностью 32 бита, в соответствие с противоречивыми требованиями к криптографическим характеристикам, рассматриваемой структуры. Рассмотрены методы синтеза классических sbox, в том числе с использованием эволюционного и генетического методов. Представлены требования к криптографическим характеристикам, как к функциям PD-sbox, так и к их составным элементам (классические sbox и ARX-функции). Предложен метод синтеза псевдо-динамических функций PD-sbox-ARX-32, включающий два этапа: 1) эвристический поиск структуры, соответствующей противоречивым требованиям к результирующим криптографическим характеристикам, потребляемым программным и аппаратным ресурсам, а также скорости работы представленной функции; 2) поиск оптимальных параметров основного элемента PD-sbox-ARX-32 – ARX-функций, при помощи эволюционного метода, суть которого заключается в подборе значений циклических сдвигов в ARX-функциях. В результате получен набор четырёх ARX-функций для псевдо-динамического преобразования PD-sbox-ARX-32, имеющего вес линейных характеристик равный 2^{-13} и разностных характеристик равный 2^{-32} (при этом эмпирический вес составляет 2^{-26}). Для определения весов криптографических характеристик в работе применены методы на основе использования SAT-решателей. Приведены выводы о том, что подобранная структура 32-битной ARX-функции в составе PD-sbox позволяет обеспечить критический путь (максимальное количество последовательных операций сложения по модулю 2^{16}) в четыре раза меньше чем 8-итерационная 32-битная Alzette-подобная структура, при двухкратном увеличении количества операций и при сопоставимых максимальных значениях весов разностных и линейных характеристик. Аналогичный результат получается при сравнении 32-битной ARX-функции с 8-итерационным 32-битным преобразованием из блочного криптоалгоритма Speck32. Предложенный метод синтеза параметров 32-битной ARX-функции позволяет минимизировать количество затрачиваемых ассемблерных инструкций на операции циклического сдвига при реализации на малоресурсных 8-битных микроконтроллерах AVR (например, ATmega328P).

Криптографические свойства; псевдо-динамическая функция; PD-sbox-ARX-32; синтез псевдо-динамических функций.

S.V. Polikarpov, V.A. Prudnikov, K.E. Rumyantsev

SYNTHESIS OF PSEUDO-DYNAMIC FUNCTIONS PD-sbox-ARX-32

The aim of the work is to develop a method for synthesizing optimal pseudo-dynamic functions PD-sbox-ARX-32, 32-bit in size, in accordance with conflicting requirements for cryptographic characteristics of the considered structure. The methods for synthesizing classical sbox'es are considered, including those using evolutionary and genetic methods. The requirements for cryptographic characteristics are presented, both for the PD-sbox functions and for their constituent elements (classical sbox and ARX functions). A method for synthesizing pseudo-dynamic functions PD-sbox-ARX-32 is proposed, including two stages: 1) heuristic search for a structure corresponding to conflicting requirements for the resulting cryptographic characteristics, consumed software and hardware resources, as well as the speed of operation of the presented function; 2) search for optimal parameters of the main element of PD-sbox-ARX-32 – ARX functions, using the evolutionary method, the essence of which is to select the values of cyclic shifts in ARX functions. As a result, a set of four ARX functions was obtained for the pseudo-dynamic transformation of PD-sbox-ARX-32, having the weight of linear characteristics equal to 2^{-13} *and difference characteristics equal to* 2^{-32} *(in this case the empirical weight is* 2^{-26}*). To determine the weights of cryptographic characteristics, methods based on the use of SAT solvers were used in the work. The paper concludes that the selected structure of the 32-bit ARX function in the PD-sbox allows for a critical path (maximum number of sequential addition operations modulo* 2^{16}*) that is four times smaller than that of the 8-iteration 32-bit Alzette-like structure, with a twofold increase in the number of operations and comparable maximum values of the weights of the difference and linear characteristics. A similar result is obtained when comparing the 32-bit ARX function with the 8-iteration 32-bit transformation from the Speck32 block cryptographic algorithm. The proposed method for synthesizing the parameters of the 32-bit ARX function allows for minimizing the number of assembler instructions spent on cyclic shift operations when implemented on low-resource 8-bit microcontrollers AVR (for example ATmega328P).*

Cryptographic properties; pseudo-dynamic function; PD-sbox-ARX-32; synthesis of pseudo-dynamic functions.

Введение. Преобразование sbox (s-box) является сокращением от substitution box – блок подстановки/замены (узел замены, в терминологии ГОСТ 28147-89), под термином «подстановка» подразумевается операция в виде табличной замены значений, операция не обязательно должна быть взаимно однозначной, как, например, в криптоалгоритме DES [1]. Функция sbox является основным нелинейным элементом множества блочных криптоалгоритмов, обеспечивающим противодействие к различным видам криптоанализа. sbox обладает множеством параметров, напрямую влияющих на устойчивость, как самой функции, так и полноценного криптоалгоритма, в котором sbox применён. Следовательно, при синтезе этого нелинейного элемента должно учитываться множество противоречивых требований, связанных со стойкостью sbox к различным криптографическим атакам, потреблению программных и аппаратных ресурсов, а также его быстродействию при программной или аппаратной реализации. Функция sbox и различные её разновидности обладают следующими характеристиками: линейные, разностные, алгебраические, а также их производные. Одним из направлений развития sbox является их объединение в структуру псевдо-динамической функции PD-sbox, основной особенностью которой является возможность разрушения статистических взаимосвязей между входными и выходными значениями за счёт динамической трансформации их криптографических свойств, а также параллельная работа фиксированных нелинейных элементов, входящих в состав PD-sbox [2]. Псевдо-динамические функции, семейства PD-sbox-ARX являются перспективным направлением развития псевдо-динамических преобразований PD-sbox, являющихся основным нелинейным элементом псевдослучайной функции pCollapser. Отличие PD-sbox-ARX от PD-sbox заключается в использовании ARX-функций в составе псевдо-динамических преобразований в качестве фиксированного нелинейного элемента [3]. Стоит отметить, что применение в качестве фиксированных sbox линейных ARX-функции показало, что миниверсия типовой функции на основе SP-сети не способна обеспечить хоть какой-то уровень нелинейности. Однако, миниверсия псевдослучайной функции

pCollapser позволяет получить из набора 4 ARX-конструкций, обладающих крайне низкими криптографическими свойствами, качественную нелинейную функцию. Подобное проявление свойств pCollapser подтверждает правильность концепции псевдо-динамических подстановок PD-sbox (разрушение статистических взаимосвязей между входными и выходными значениями за счёт динамической трансформации их криптографических свойств) [4].

Цель работы – определение метода синтеза оптимальных псевдо-динамических функций PD-sbox-ARX-32, размерностью 32 бита, в соответствие с противоречивыми требованиями к криптографическим свойствам, затрачиваемым ресурсам и задержки преобразования рассматриваемой структуры.

Описание псевдо-динамических функций PD-sbox. Псевдо-динамическая функция (преобразование) PD-sbox представляет собой структуру, включающую в свой состав набор фиксированных sbox или иных нелинейных элементов, в частности – ARX-функций [5, 6].

Аргумент каждого фиксированного sbox параметризован значением состояния S_i, где i – номер фиксированного sbox или иного нелинейного элемента (от 0 до $N-1$). Текущее значение состояния $S = \{S_0, S_1, S_2, \dots, S_{N-1}\}$ задаёт один вариант преобразования из набора возможных PD-sbox. Преобразование, полученное на основе определённого значения состояния S следует называть эквивалентным (сгенерированным) преобразованием для PD-sbox. Число эквивалентных преобразований ограничено количеством возможных значений состояния S, которые могут динамически изменяться в ходе обработки блока информации. При этом предполагается, что вероятностные свойства состояния S соответствуют равномерному распределению.

Общий вид выражения, описывающего структуру псевдо-динамической функции PD-sbox:

$$\mathrm{Y} = \bigoplus_{\mathrm{i}=0}^{\mathrm{N}-1} \mathrm{sbox_i}(\mathrm{X} \oplus \mathrm{S_i}),$$

где sbox – фиксированный блок замены (обычно описывается в виде табличной замены значений, при этом операция не обязательно должна быть взаимнооднозначной, как, например, в криптоалгоритме DES [1]); N– количество фиксированных sbox; X – биты на входе; Y – биты на выходе; S – биты состояния псевдо-динамической функции; $\oplus$ – операция сложения по модулю 2.

Псевдо-динамическая функция PD-sbox может функционировать в статическом (ключезависимом) и динамическом (зависимом от значений ключа и промежуточных состояний). В случае динамического равновероятного изменения состояний S, как дифференциальные усреднённые свойства, так и линейные, могут быть близки к идеальным (при усреднении характеристик по всем эквивалентным преобразованиям). Это потенциально позволяет нейтрализовать существующие методы дифференциального и линейного криптоанализа [5, 6].

Описание псевдо-динамических функций PD-sbox-ARX. В [3] предложен вариант применения специально подобранных ARX-функций в составе псевдо-динамических операций sbox, что позволяет обеспечить как параллелизм обработки информации, так и стойкость к статистическим методам криптоанализа и возможность эффективной программной реализации. Структура используемых ARX-функций приведена на рис. 1. Выбор подобной архитектуры функции обусловлен обеспечением криптографических свойств и оптимальным использованием возможностей современных процессоров и аппаратных платформ.

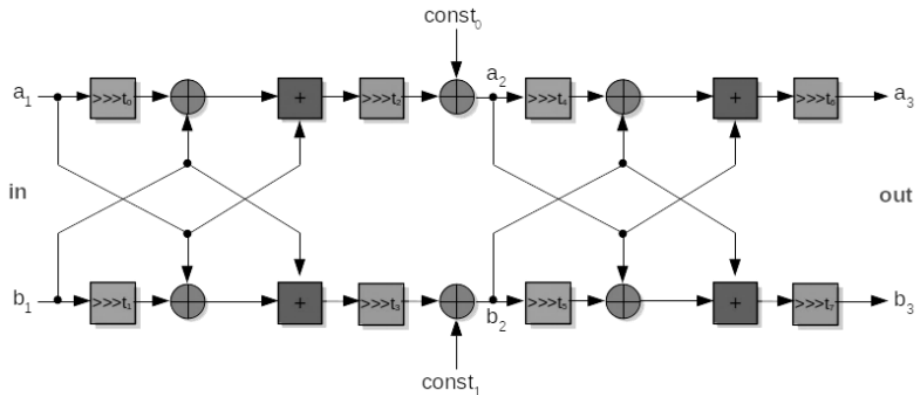

Рис. 1. Структура используемых ARX-функций

PD-sbox-ARX состоит из четырёх параллельно включённых в её структуру ARX-функций. Размерность входа-выхода PD-sbox-ARX соответствует размерности используемых ARX-конструкций. На рис. 2 представлена псевдо-динамическая sbox, включающая в свой состав четыре параллельно интегрированных ARX-функции. Размерность входа-выхода PD-sbox-ARX соответствует размерности используемых ARX-конструкций [2].

Выражение, описывающее значение на выходе:

$$c_i = \bigoplus_{j=0}^{3} funcARX_j(m_i \oplus s_j^i),$$

где i – индекс-битного слова из входного/выходного вектора и далее индекс PD-sbox-ARX; j – индекс компонента PD-sbox-ARX; m_i – n-битные слова из входного вектора; c_i – n-битные слова из выходного вектора; $funcARX$ – ARX-функция; s_j^i – n-битные слова из входного вектора управляющего состояния [3].

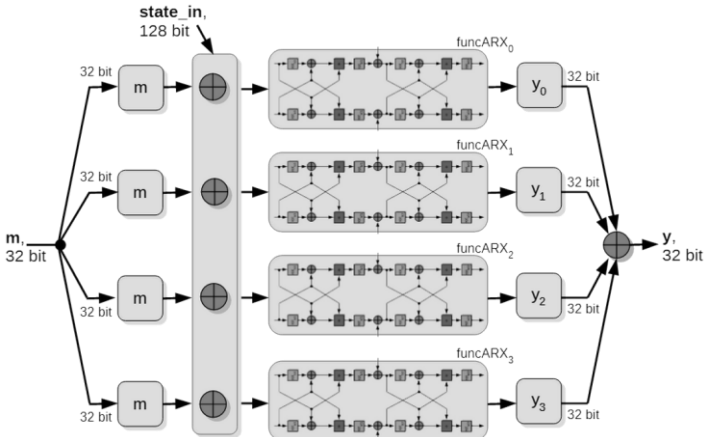

Рис. 2. Псевдо-динамическая операция sbox на основе ARX-конструкций

Выражение, описывающее индивидуальные управляющие состояния на выходе PD-sbox-ARX:

$$g_n^i = c_i \oplus funcARX_j\left(m_i \oplus s_j^i\right) = \bigoplus_{n=0, n \neq i}^{3} funcARX_j(m_i \oplus s_j^i).$$

Описание sbox Alzette. В работе [7] представлен sbox размерностью 64 бит, реализованный на основе ARX-конструкций – Alzette. Данное преобразование может быть вычислено с использованием 12 инструкций на современных процессорах, а параллельная реализация sbox Alzette может использовать векторные (SIMD) инструкции. За одну итерацию Alzette достигает разностных и линейных свойств, сравнимых со свойствами sbox AES [7]. Alzette обладает следующим преимуществом – использование операций размерностью 32 бит, следовательно, согласно [8], его эффективная реализация возможна на множестве различных архитектур, благодаря использованию регистров сдвига (barrel shift registers), если они доступны, а также благодаря подобранным значениям операций циклического сдвига.

Рассмотрим структуру Alzette, представленную на рис. 3. Функция параметризована константой c размерностью 32 бит, используемой 4 раза. На вход Alzette поступает два слова размерностью 32 бит. Далее над каждым из них выполняются операции циклического сдвига, сложения по модулю 2^{32}, а также XOR.

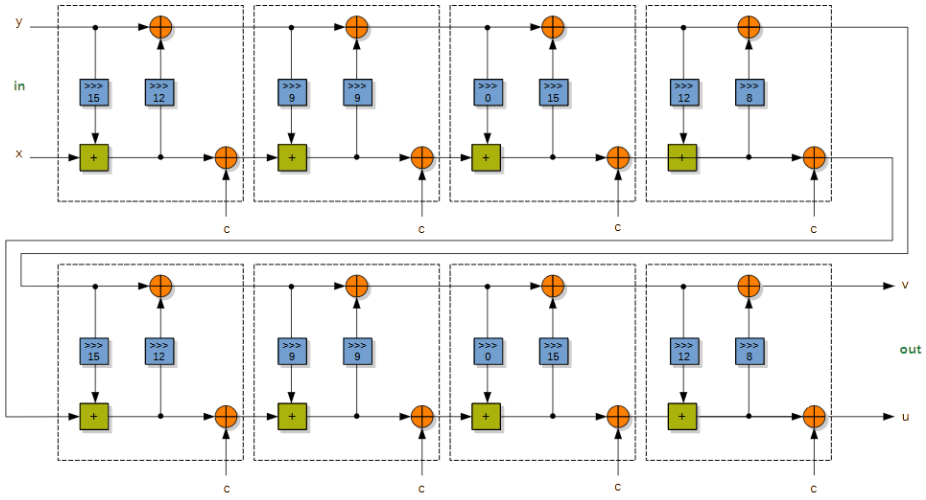

Рис. 3. Структура sbox Alzette (8 раундов)

При использовании Alzette в составе преобразования Sparkle, используются следующие константы: $c_0 = b7e15162$, $c_1 = bf715880$, $c_2 = 38b4da56$, $c_3 = 324e7738$, $c_4 = bb1185eb$, $c_5 = 4f7c7b57$, $c_6 = cfbfa1c8$, $c_7 = c2b3293d$. Sparkle – это семейство криптографических преобразований размерностью 256, 384 или 512 бит. Sparkle включает в свой состав операции сложения по модулю слова, циклического сдвига и XOR (ARX-конструкции). Основой перестановки являются sbox на основе ARX-функций Alzette [9].

Таким образом, применение ARX-операций позволило разработчикам Alzette создать функцию (блок замены) размерностью 64 бит. Это значительно больше, чем типовая размерность блоков замены в криптографических преобразованиях. Следовательно, прямой расчёт криптографических свойств (например, таблиц распределения разностей DDT и линейных приближений LAT) для sbox Alzette крайне затруднителен или вовсе невозможен, так как требует недоступного объёма вычислительных ресурсов. Поэтому, авторы Alzette применяли SAT-решатели для определения её криптографических свойств [7].

Применительно к sbox Alzette её авторы используют термин “раунд” (итерация), по аналогии с блочными криптографическими преобразованиями (например, как в криптоалгоритме Speck64, структура которого представлена на рис. 4). В табл. 1 приведены границы разностных и линейных свойств для версий Alzette с использованием от 1 до 12 раундов. Основная версия Alzette предполагает 4 раунда преобразования [10].

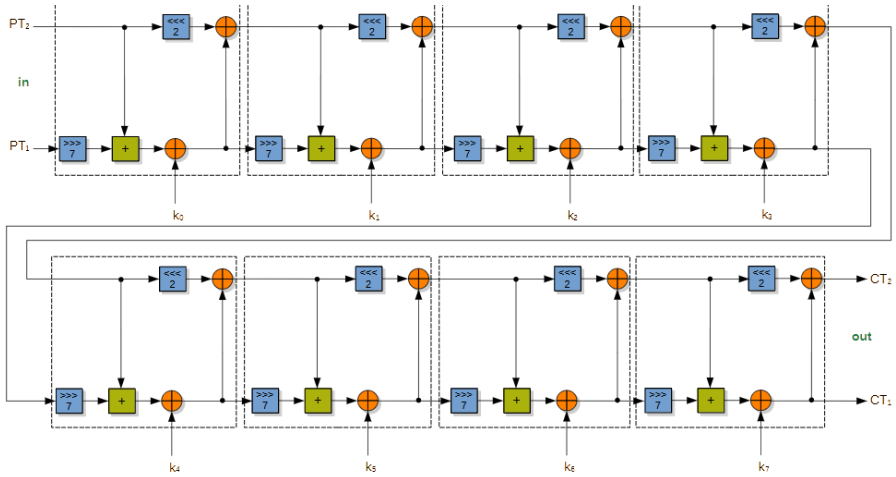

Рис. 4. Структура Speck32 (8 раундов)

Таблица 1

Нижние границы для разностных и линейных свойств sbox Alzette

$(r_0, r_1, r_2, r_3, s_0, s_1, s_2, s_3)$	1	2	3	4	5	6	7	8	9	10	11	12
(31, 17, 0, 24, 24, 17, 31, 16)	0 0	1 0	2 1	6 2	10 5	18 8	≥24 13(11.64)	≥32 17(15.79)	≥36 –	≥42 –	≥46 –	≥52 –
(17, 0, 24, 31, 17, 31, 16, 24)	0 0	1 0	2 1	6 2	10 5	17 9	≥25 13	≥31 16	≥37 –	≥41 –	≥47 –	– –
(0, 24, 31, 17, 31, 16, 24, 17)	0 0	1 0	2 1	6 2	10 6	18 8	≥24 13	≥32 15	≥36 –	≥42 –	– –	– –
(24, 31, 17, 0, 16, 24, 17, 31)	0 0	1 0	2 1	6 2	10 5	17 9	≥25 12	≥31 16	≥37 –	– –	– –	– –
Speck64	0 0	1 0	3 1	6 3	10 6	15 9	21 13	29 17	≥32 19	– 21	– 24	– 27

Для каждого смещения первая строка демонстрирует $\log_2 p$, где p – максимальная ожидаемая вероятность дифференциального следа для дифференциального случая. Вторая строка демонстрирует $\log_2 c$, где c – максимальная ожидаемая абсолютная линейная корреляция следа для линейного случая. Значение, указанное в скобках, соответствует максимальной абсолютной корреляции линейной оболочки с учетом кластеризации, полученной экспериментальной проверкой.

Как и в блочных преобразованиях, авторы Alzette используют широко распространённый метод поиска разностных и линейных характеристик – когда определяются разностные и линейные свойства отдельных раундов и результирующие свойства всего преобразования определяются через «принцип накопления» (piling-up principle) [11]. Данный принцип предполагает, что входные значения каждого раунда являются статистически независимыми.

Стоит отметить одну важную особенность – в Alzette между «раундами» не используется добавление раундовых ключей, что, предположительно, существенно нарушает «принцип накопления». Однако, это не помешало авторам Alzette получить валидные криптографические свойства используя SAT-решатели и применяя метод Мацуи для поитерационного поиска разностных и линейных характеристик Alzette [10].

Анализ криптографических свойств PD-sbox-ARX-32. Рассмотрим анализ криптографических характеристик синтезированной псевдо-динамической функции PD-sbox-ARX-32, в частности разностных и линейных, а также сравним их с аналогичными параметрами миниверсии sbox Alzette, размерностью 32 бита.

В силу размерности и сложности анализируемых конструкций, для поиска разностных и линейных свойств использован метод поиска криптографических характеристик, использующий SAT-решатели, в частности – фреймворк CASCADA [12]. Создатели дан-

ного фреймворка особое внимание уделили анализу ARX-функций. В настоящее время SAT-решатели являются одним из основных методов поиска и валидации криптографических свойств и характеристик криптографических преобразований [12–15].

Свойства, используемые в разностном криптоанализе, являются разностями (α, β) по функции шифрования E_k с высокой ожидаемой разностной вероятностью. При наличии разностей (α, β) по f его разностная вероятность определяется как

$$\#\{x: f(x\Delta\,\alpha)\nabla f(x) = \beta\}/2^n\,,$$

где $\Delta = \oplus = \nabla$. Оператор ∇ вычисляет разность пары значений (x, x'), а оператор Δ принимает в качестве входных данных значение x и разность α и выводит такое значение x', что пара (x, x') имеет разность α.

Ожидаемая разностная вероятность p — это разностная вероятность, усредненная по ключевому пространству K:

$$p = \frac{1}{|K|}\sum_{k\in K}\#\{x: f(x\Delta\alpha)\nabla\mathrm{f(x)} = \beta\}/2^n,$$

а сложность разностного криптоанализа составляет $O(1/p)$.

Свойство (α, β) над функцией f является действительным, если его вероятность распространения не равна нулю. В этом случае мы определяем вес распространения (α, β) как отрицательный двоичный логарифм его вероятности распространения:

$$PW_f(\alpha, \beta) = -\log_2\Big(PP_f(\alpha, \beta)\Big).$$

Результат работы CASCADA – веса найденных оптимальных криптографических характеристик:

$$w = -\log_2 P(.),$$

где $P(.)$ – вероятность появления входной/выходной разности для тестируемой функции (для разностного анализа) или значения корреляции (для линейного анализа).

Следующие обозначения и определения являются стандартными в линейном криптоанализе.

Определение 1. Итеративный блочный шифр – это алгоритм, который преобразует блок открытого текста фиксированного размера n в блок шифр-текста идентичного размера под воздействием ключа k путем применения итеративного обратимого преобразования p, называемого раундовым преобразованием. Обозначая открытый текст как x_0, а шифр-текст как x_R, операция зашифрования может быть записана как:

$$x_{r+1} = p_{k_r}(x_r), \quad r = 1, 2, \dots, R,$$

где k_r являются подключами, сгенерированными алгоритмом выработки ключей. Для простоты мы рассматриваем -битные ключи и подключи.

Определение 2. Пусть $F: \{0,1\}^n \to \{0,1\}^n$ – биективное преобразование, a, b – две маски $\in \{0,1\}^n$. Если $X \in \{0,1\}^n$ равномерно распределенная случайная величина, тогда смещение линейной аппроксимации $LB(a, b)$ определено, как:

$$LB(a, b) = \Pr_X\{a * X = b * F(X)\} - \frac{1}{2},$$

где $*$ – скалярное произведение. Если F параметризовано ключом K, запишем $LB(a, b, K)$ и ожидаемое линейное отклонение $ELB(a, b)$ определено, как:

$$ELB(a, b) = \underset{K}{E}\big(LB(a, b, K)\big).$$

Линейное отклонение может быть вычислено для различных преобразований, например, для одного *sbox*, раундовой функции или блочного шифра. Точное определение линейного отклонения является вычислительно сложной задачей по мере увеличения размерности преобразования.

Определение 3. Одноraундовая характеристика для раунда i итерационного блочного шифра представляет собой пару-битных векторов $\langle a_i, b_i\rangle$, соответствующих входным и выходным маскам для этого раунда. -раундовая характеристика для раундов $1 \dots R$ представляет собой $(R + 1)$-кортеж n-битных векторов $\Omega = \langle a_1, a_2, \dots, a_{R+1}\rangle$, где $\langle a_i, a_{i+1}\rangle$ соответствуют входным и выходным маскам для раунда i.

Определение 4. Шифр Маркова – блочный шифр, в котором линейные (и разностные) отклонения разных раундов независимы друг от друга, предполагая, что в разных раундах используются равномерно-случайные подключи [16].

В нашем случае PD-sbox-ARX является небиективным преобразованием, однако, возможным вариантом его применения является использование в качестве нелинейной функции Фейстель-подобного итерационного преобразования, представленного на рис. 5. Которое, как известно, является взаимооднозначным преобразованием. Возможность применения небиективного преобразования является важным преимуществом сети Фейстеля и такие известные криптоалгоритмы как DES и ГОСТ [1, 17] используют небиективную функцию. Стоит отметить, что изначально понятия итерационной характеристики было введено для анализа криптоалгоритма DES [18, 19].

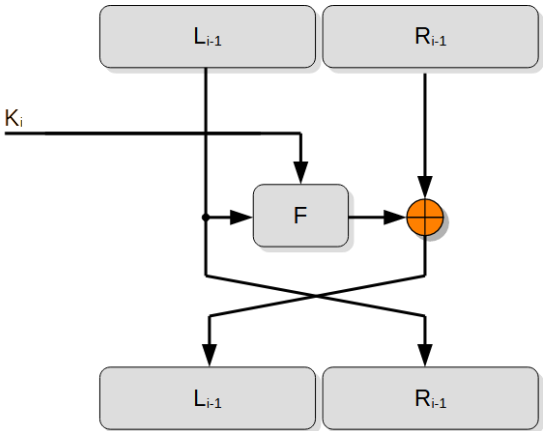

Рис. 5. Раунд сети Фейстеля

PD-sbox-ARX-32 является невзаимнооднозначной псевдо-динамической функцией, несмотря на это, её анализ с использованием SAT/SMT-решателей возможен и найденные характеристики будут валидны. В качестве примера следует привести работу [20], в которой представлены результаты первого криптоанализа шифра DES с использованием SAT-решателей.

Особенности программной реализации на малоресурсных процессорах. Реализация криптографических преобразований на микроконтроллерах (малоресурсных процессорах) является важнейшей задачей, обусловленной как повсеместным использованием криптографических преобразований в сетевых протоколах, так и значительной ресурсоёмкости этих преобразований, что может привести к дефициту вычислительных ресурсов для основных процессов.

Стоит отметить, что на различных процессорах/микроконтроллерах для ARX-операций может требоваться разное количество тактов на их выполнение. Однако, для относительно простых микроконтроллеров предполагается, что на выполнение операций ADD или XOR требуется один такт [21]. Но выполнение операции ROL может требовать значительного количества инструкций и тактов микроконтроллера.

В качестве таких процессоров/микроконтроллеров мы будем рассматривать широко распространённые 8-битные микроконтроллеры AVR фирмы Atmel, микроконтроллеры ATmega328P (применяется, например, в Arduino UNO R3) и микроконтроллеры на базе инструкций MIPS32/MIPS64 (самым известным является семейство микроконтроллеров PIC32 от Microchip).

Мы не будем рассматривать здесь более совершенные процессоры/микроконтроллеры на основе архитектур RISC-V и ARM, в которых уже реализована встроенная операция ROL, требующая небольшого количества тактов на выполнение. Что касается 8-битных микроконтроллеров семейства AVR (например, ATmega328P), то в них аппаратно реализована инструкция циклического сдвига только на 1 бит.

Для определения количества инструкций на операции циклического сдвига с разным количеством сдвигаемых бит мы воспользовались ресурсом godbolt.org, интерфейс которого представлен на рис. 6, позволяющим в интерактивном режиме, как вывести результат компиляции исходного кода в виде набора ассемблерных инструкций, так и легко выбрать компилятор и архитектуру/семейство целевого процессора или микроконтроллера.

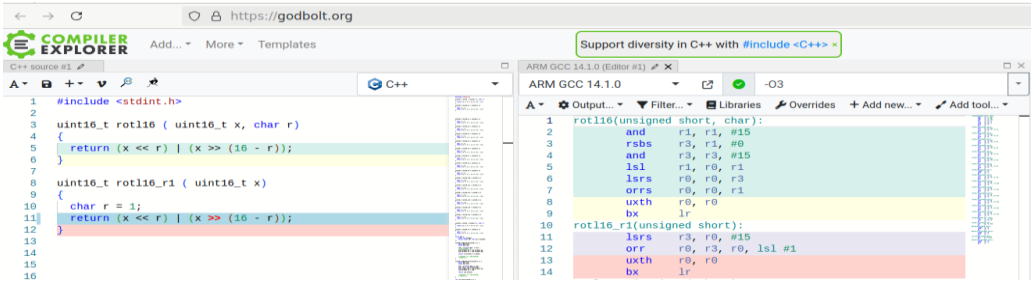

Рис. 6. Результат компиляции на ресурсе godbolt.org

Мы взяли типовой способ описания операции ROL на языке C в виде типовой конструкции из двух нециклических сдвигов (влево и вправо) и объединения результатов при помощи операции OR или XOR. Как известно, современные компиляторы, в том числе gcc, хорошо распознают такую типовую конструкцию и при компиляции заменяют её на соответствующую встроенную команду. Или, в случае отсутствия такой команды – на оптимальный набор инструкций, выполняющий эквивалентное преобразование.

Для каждого значения циклического сдвига мы реализовали отдельную функцию, представленную на рис. 7, что позволило в дальнейшем оценить количество затрачиваемых инструкций для ROL с разным значением сдвига.

```
1   uint16_t rotl16_r1 (uint16_t x) {char r = 1; return (x << r) | (x >> (16 - r));}
2   uint16_t rotl16_r2 (uint16_t x) {char r = 2; return (x << r) | (x >> (16 - r));}
3   uint16_t rotl16_r3 (uint16_t x) {char r = 3; return (x << r) | (x >> (16 - r));}
4   ...
5   uint16_t rotl16_r15 (uint16_t x) {char r = 15; return (x << r) | (x >> (16 - r));}
```

Рис. 7. Отдельные функции для значений циклического сдвига

После компиляции для каждой функции подсчитали количество требуемых на реализацию ассемблерных инструкций. При этом инструкцию RET (возврат из функции) мы не учитывали, так как при компиляции целиком всей ARX-функции обычно осуществляется встраивание кода ROL-функции непосредственно в точку вызова функции (вместо её фактического вызова, что позволяет снизить количество инструкций вызова и возврата). В табл. 2 приведён пример результата компиляции, использовался флаг «-O3».

Таблица 2

Пример результата компиляции 16-битовых операций ROL-1 и ROL-2 для различных целевых платформ

Операция	Архитектура / компилятор	
	x86-64 / gcc 14.2	AVR / gcc 14.1.0
ROL-1	rotl16 r1(unsigned short): mov eax, edi rol ax ret	rotl16 r1(unsigned int): .L__stack_usage = 0 lsl r24 rol r25 adc r24,__zero_reg__ ret

Окончание табл. 2

ROL-2	rotl16 r2(unsigned short):
	mov eax, edi
	rol ax, 2
	ret
		rotl16 r2(unsigned int):
		.L__stack_usage = 0
		mov r18,r24
		mov r24,r25
		mov r19,r18
		swap r19
		lsr r19
		lsr r19
		andi r19,lo8(3)
		lsl r25
		lsl r25
		swap r24
		lsr r24
		lsr r24
		andi r24,lo8(3)
		lsl r18
		lsl r18
		or r24,r18
		or r25,r19
		ret

Как можно увидеть, для архитектуры x86-64 требуется всего 2 инструкции для реализации ROL-1 и ROL-2, а вот для AVR требуется 3 инструкции для ROL-1 и 17 инструкций для ROL-2. При этом для x86-64 одна из инструкций – команда перемещения из одного регистра в другой (mov eax, edi), которая может быть убрана при оптимизации кода.

В табл. 3 приведены сводные значения по количеству инструкций на реализацию операции 16-битового циклического сдвига для рассматриваемых архитектур.

Таблица 3

Количество ассемблерных инструкций для реализации 16-битовой операций ROL в зависимости от значения циклического сдвига

Операция	**Количество инструкций для архитектуры/компилятора**				
	AVR (gcc 14.1.0)	Arduino Uno (1.8.9)	mips32/64 (gcc 14.1.0)	ARM (gcc 14.1.0)	x86-64 (gcc 14.2)
ROL-1	**3**	**3**	5	3	2
ROL-2	17	18			
ROL-3	17	18			
ROL-4	13	14			
ROL-5	17	18			
ROL-6	17	18			
ROL-7	13	14			
ROL-8	**3**	**3**			
ROL-9	13	14			
ROL-10	17	18			
ROL-11	17	18			
ROL-12	13	14			
ROL-13	17	18			
ROL-14	17	18			
ROL-15	**4**	**4**			

Следует обратить внимание на то, что иные операции (сложение по модулю, XOR) соответствуют одной ассемблерной инструкции и ими можно пренебречь.

Метод синтеза псевдо-динамической функции PD-sbox-ARX-32. В ходе экспериментальных исследований нами был выработан следующий метод синтеза PD-sbox-ARX-32:

1. Эвристический выбор структуры ARX-функции с учётом возможных особенностей программной и аппаратной реализаций.

2. Начальное заполнение параметров циклических сдвигов ARX-функций (всего по 8 значений на 4 функции) значением 8.

3. Последовательный выбор каждого параметра ARX-функции, замена его на случайное значение из допустимого диапазона (от 0 до 15), проверка криптографических свойств (вес разностных и линейных характеристик) и ожидаемого количества затрачиваемых ассемблерных инструкций для полученной версии ARX-функции. После обхода всех параметров первой ARX-функции выбирается наилучшая версия (при её наличии), которая заменяет исходную. Далее осуществляется переход к следующей ARX-функции для выполнения аналогичных действий.

4. Действия из пункта 3 повторяются до момента отсутствия улучшений в свойствах ARX-функций.

5. Формирование при помощи пунктов 2 и 3 набора наиболее удачных ARX-функций.

6. Выбор из набора наиболее удачных ARX-функций варианта с наименьшим количеством затрачиваемых ассемблерных инструкций для микроконтроллеров архитектуры AVR. В табл. 4 представлено количество ассемблерных инструкций для реализации 16-битовых операций ROL для 12 отобранных параметров PD-sbox-ARX-32.

Следует обратить внимание на следующее: по пункту 2 экспериментальные исследования показали, что если сразу задать «удачный» вариант значений циклического сдвига (для 16-битных сдвигов это значение равно 8), то значительно увеличивается вероятность того, что эти значения будут в результирующей ARX-функции после операций синтеза; по пункту 5 экспериментальные исследования показали, что предложенный пошаговый подбор параметров позволяет получать PD-sbox-ARX с достаточно близкими криптографическими свойствами. При синтезе 100 PD-sbox-ARX 73 варианта имели вес разностных характеристик Wd равный 32 и вес линейных характеристик Wl, равный 13 и 14. Такие характеристики очень близки 8-раундовым преобразованиям Speck32 и miniAlzette32. Поэтому варианты с более худшими характеристиками исключаются из набора.

В результате получилось 12 параметров, представленных в табл. 4, при этом наименьшее количество ассемблерных инструкций для архитектуры AVR составило $N = 360$, что в 1,4 раза и 1,7 раза больше, чем для наихудшего и наилучшего вариантов синтеза соответственно, для которых $N = 256$ и $N = 217$.

Таблица 4

Количество ассемблерных инструкций для реализации 16-битовых операций ROL для 12 отобранных параметров PD-sbox-ARX-32

№	Архитектура микроконтроллера		
	AVR	mips64	ARM
1	242	150	73
2	248		
3	256		
4	222		
5	242		
6	248		

Окончание табл. 4

7	240		
8	231		
9	**217**		
10	234		
11	234		
12	248		

В табл. 5 приведено сравнение свойств лучшей синтезированной PD-sbox-ARX-32 со свойствами 8-итерационной 32-битной Alzette-подобной структуры и 8-итерационным 32-битным преобразованием из блочного криптоалгоритма Speck32.

Таблица 5

Количество ассемблерных инструкций для реализации 16-битовой операций ROL

Преобразование	**Архитектура микроконтроллера**			**Криптографические свойства**			
	AVR	mips64	ARM	Wd	Wde	Wl	Wle
miniAlzette32 (8 раундов)	154	70	42	27	~27	13	~13
Speck32 (8 раундов)	240	80	48	24	~24	12	~12
PD-sbox-ARX-32	**217**	**150**	**73**	**32**	**~26**	**13**	**~13**

Сравнение разработанного метода синтеза с методом случайного поиска параметров псевдо-динамической функции PD-sbox-ARX-32. Для оценки эффективности предложенного метода нами сформировано 100 000 случайных наборов параметров ARX-функций для PD-sbox-ARX-32, для которых определено количество затрачиваемых ассемблерных инструкций и криптографические свойства – вес разностных характеристик Wd, вес линейных характеристик Wl. Данный универсальный метод применён нами для сравнения, так как иные методы синтеза параметров PD-sbox-ARX не представлены в открытой печати.

Ниже приведены результаты в виде гистограмм распределения по количеству затрачиваемых ассемблерных инструкций – на рис. 8, в виде гистограмм распределения по весам разностных Wd и линейных Wl характеристик – на рис. 9.

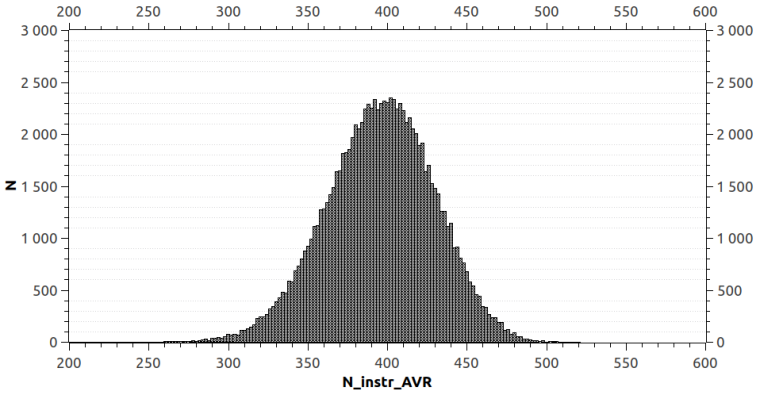

Рис. 8. Гистограмма распределения наборов параметров по количеству затрачиваемых ассемблерных инструкций

Полученная гистограмма имеет нормальное распределение, среднее значение составляет 395,92, стандартное отклонение 33,96, стандартная ошибка 0,1074, минимальное 230, максимальное 512, медиана 397.

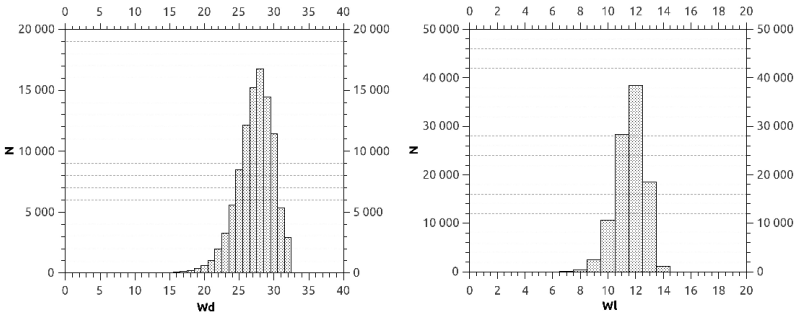

Рис. 9. Гистограмма распределения по весам разностных Wd и линейных Wl характеристик

Анализ гистограммы, представленной на рис. 8, позволяет выделить 5 комбинаций параметров, выделенных на рис. 10, обладающих минимальным количеством затрачиваемых ресурсов. Свойства параметров приведены в табл. 6.

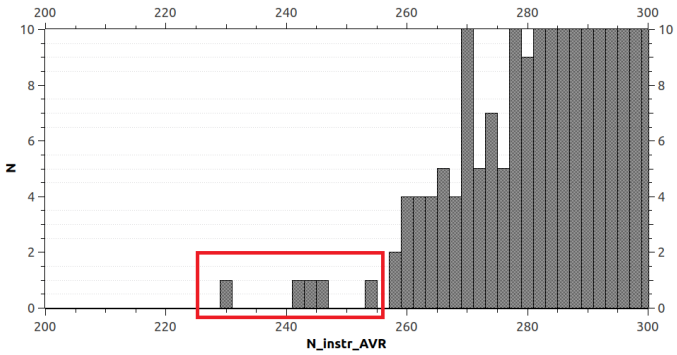

Рис. 10. Комбинации параметров, обладающих минимальным количеством затрачиваемых ресурсов при использовании метода случайного подбора

Таблица 6

Свойства 5 случайно подобранных комбинаций параметров

N	AVR	Wd	Wl
1	230	24	9
2	243	20	10
3	244	18	8
4	245	20	11
5	246	25	12
Синтезированный PD-sbox-ARX	**217**	**32**	**13**

Все случайно подобранные комбинации параметров ARX-функций с минимальным количеством затрачиваемых ассемблерных инструкций обладают неприемлемыми криптографическими характеристиками. Однако, даже в данном случае они существенно уступают синтезированному PD-sbox-ARX по количеству затрачиваемых ресурсов. При сравнении с вариантом 1 из табл. 6, разница составляет ~5%, учитывая его неудовлетворительные криптографические характеристики.

На рис. 11 приведены результаты в виде гистограмм распределения по количеству затрачиваемых ассемблерных инструкций, минимальное значение затрачиваемых ассемблерных инструкций равно 284 при $Wd > 29$ и $Wl > 10$.

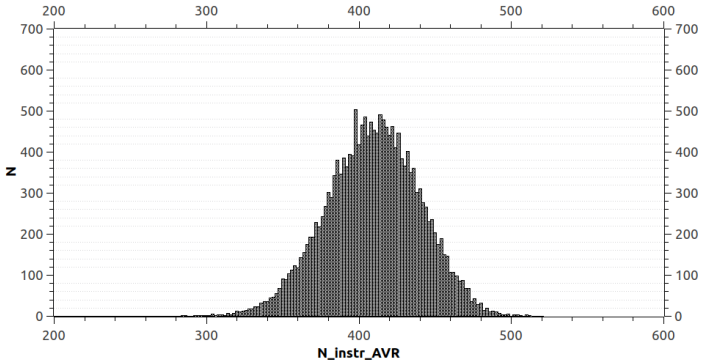

Рис. 11. Гистограмма распределения наборов параметров по количеству затрачиваемых ассемблерных инструкций при $Wd > 29$ и $Wl > 10$

Полученная гистограмма имеет нормальное распределение, среднее значение составляет 409,66, стандартное отклонение 0,57, стандартная ошибка 0,2264, минимальное 284, максимальное 553, медиана 410.

Анализ гистограммы, представленной на рис. 11, позволяет выделить 5 комбинаций параметров, выделенных на рис. 12, обладающих минимальным количеством затрачиваемых ресурсов при $Wd > 29$ и $Wl > 10$. Свойства параметров приведены в табл. 7.

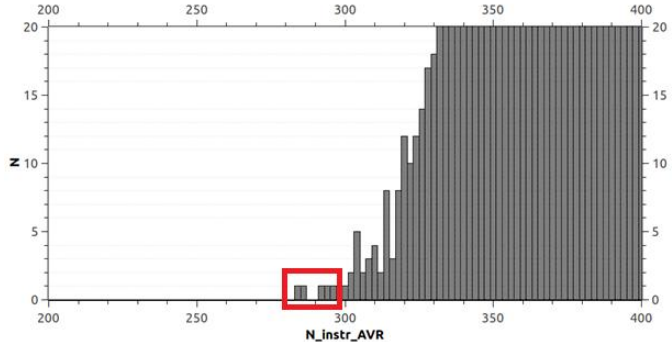

Рис. 12. Комбинации параметров, обладающих минимальным количеством затрачиваемых ресурсов при использовании метода случайного подбора при $Wd > 29$ и $Wl > 10$

Таблица 7

Свойства 5 случайно подобранных комбинаций параметров при $Wd > 29$ и $Wl > 10$

N	AVR	Wd	Wl
1	284	30	11
2	286	30	11
3	292	30	11
4	294	31	11
5	297	30	11
Синтезированный PD-sbox-ARX	**217**	**32**	**13**

Все случайно подобранные комбинации параметров ARX-функций с минимальным количеством затрачиваемых ассемблерных инструкций обладают удовлетворимыми криптографическими свойствами, однако существенно уступают синтезированному PD-sbox-ARX по количеству затрачиваемых ресурсов. При сравнении с вариантом 1 из таблицы 7, разница составляет ~24%, без учёта уступающих криптографических свойств.

Заключение. Подобранная структура 32-битной ARX-функции в составе PD-sbox позволяет обеспечить критический путь (максимальное количество последовательных операций сложения по модулю 2^{16}) в четыре раза меньше, чем 8-итерационная 32-битная Alzette-подобная структура, при двухкратном увеличении количества операций и при сопоставимых максимальных значениях весов разностных и линейных характеристик.

Аналогичный результат получается при сравнении 32-битной ARX-функции с 8-итерационным 32-битным преобразованием из блочного криптоалгоритма Speck32.

Практическая значимость: при аппаратной реализации ARX-функции данное свойство позволяет пропорционально уменьшить (до 4 раз) задержку при преобразовании блоков информации.

Предложенный метод синтеза параметров 32-битной ARX-функции позволяет получить параметры операций циклического сдвига, при которых обеспечивается максимальный вес разностной характеристики равный 2^{-32} (эмпирический вес 2^{-26}) и вес линейной характеристики 2^{-13} для результирующего PD-sbox-ARX, включающей в свой состав четыре 32-битные ARX-функции. Сопоставимые разностные и линейные характеристики имеют 8-итерационные 32-битная Alzette-подобная структура и 8-итерационное 32-битное преобразование из блочного криптоалгоритма Speck32.

Предложенный метод синтеза параметров 32-битной ARX-функции позволяет минимизировать количество затрачиваемых ассемблерных инструкций на операции циклического сдвига при реализации на малоресурсных 8-битных микроконтроллерах семейства AVR (например, ATmega328P).

Например, при параметрах циклических сдвигов [5, 8, 8, 8, 8, 11, 0, 0], [12, 3, 8, 8, 8, 15, 4, 4], [8, 12, 8, 8, 9, 8, 8, 8], [1, 8, 8, 8, 8, 3, 12, 12] на их реализацию потребуется 217 ассемблерных инструкций микроконтроллера AVR. Что ориентировочно на 10% меньше, чем для типовых получаемых параметров и соответствует количеству ресурсов 8-итерационного преобразования из криптоалгоритма Speck32 (240 ассемблерных инструкций микроконтроллера AVR) на циклические сдвиги.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Diffie W. and Hellman M.E.* Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard // in Computer. – June 1977. – Vol. 10, No. 6. – P. 74-84. – DOI: 10.1109/C-M.1977.217750.
2. *Поликарпов С.В., Румянцев К.Е., Кожевников А.А.* Псевдо-динамические таблицы подстановки: основа современных симметричных криптоалгоритмов // Научное обозрение. – 2014. – № 12. – С. 162–166. – URL: http://www.sced.ru/ru/files/7_12_1_2014/7_12_1_2014.pdf.
3. *Поликарпов С.В., Румянцев К.Е., Прудников В.А.* Высокопроизводительная псевдослучайная функция pCollapserARX256-32x2 // XXIV научно-практическая конференция «РусКрипто'2022». – 2022. – URL: https://ruscrypto.ru/resource/archive/rc2022/files/02_polikarpov_rumyantsev_prudnikov.pdf.
4. *Поликарпов С.В., Румянцев К.Е., Прудников В.А.* Исследование свойств миниверсии псевдослучайной функции pcollapser // Известия ЮФУ. Технические науки. – 2022. – № 6 (230). – С. 148-162. – DOI: 10.18522/2311-3103-2022-6-148-162.
5. *Поликарпов С.В., Румянцев К.Е., Кожевников А.А.* Исследование линейных характеристик псевдо-динамических подстановок // Известия ЮФУ. Технические науки – 2015. – № 5 (166). – С. 111-123. – URL: http://izv-tn.tti.sfedu.ru/wp-content/uploads/2015/5/11.pdf.
6. *Polikarpov S., Rumyantsev K., Petrov D.* Computationally efficient method for determining averaged distribution of differentials for pseudo-dynamic substitutions // International Conference on Electrical, Electronics, Materials and Applied Science, AIP Conf. Proc., 1952 / eds. V. Rao, A. Ben, S. Bhukya. Amer. Inst. Phys., 2018, UNSP 020091. – DOI: 10.1063/1.5032053.
7. *Beierle C. et al.* Alzette: A 64-Bit ARX-box / In: Micciancio, D., Ristenpart, T. (eds) //Advances in Cryptology – CRYPTO 2020: Lecture Notes in Computer Science. – 2020. – Vol. 12172. – Springer, Cham. – https://doi.org/10.1007/978-3-030-56877-1_15.

8. *Dinu D., Corre Y.L., Khovratovich D. et al*. Triathlon of lightweight block ciphers for the Internet of things // J Cryptogr Eng. – 2019. – 9. – P. 283-302. – https://doi.org/10.1007/s13389-018-0193-x.
9. *Beierle C., Biryukov A., Cardoso dos Santos L., Großschädl J., Perrin L., Udovenko A., Velichkov V., & Wang Q.* Lightweight AEAD and Hashing using the Sparkle Permutation Family // IACR Transactions on Symmetric Cryptology. – 2020. – 2020(S1). – P. 208-261. – https://doi.org/10.13154/tosc.v2020.iS1.208-261.
10. *Beierle C., Biryukov A., Cardoso dos Santos L.* Schwaemm and Esch: Lightweight Authenticated Encryption and Hashing using the Sparkle Permutation Family. University of Luxembourg. – 2019. – URL: https://sparkle-lwc.github.io/assets/sparkle-specification-latest.pdf.
11. *Beyne T.* A Geometric Approach to Linear Cryptanalysis / In: Tibouchi, M., Wang, H. (eds) // Advances in Cryptology – ASIACRYPT 2021: Lecture Notes in Computer Science. – Vol. 13090. – Springer, Cham, 2021. – https://doi.org/10.1007/978-3-030-92062-3_2.
12. *Ranea A., Rijmen V.* Characteristic Automated Search of Cryptographic Algorithms for Distinguishing Attacks (CASCADA) // IET Information Security. – 2022. – 16 (6). – DOI: 10.1049/ise2.12077. – URL: https://eprint.iacr.org/2022/513.pdf.
13. *Stachowiak S., Kurkowski M., & Soboń A*. New results in SAT – cryptanalysis of the AES // 2022 IEEE 16th International Scientific Conference on Informatics (Informatics). – 2022. – P. 280-286.
14. *Bellini E., Piccoli A.D., Formenti M., Gérault D., Huynh P., Pelizzola S., Polese S., & Visconti A.* Differential Cryptanalysis with SAT, SMT, MILP, and CP: A Detailed Comparison for Bit-Oriented Primitives // Cryptology and Network Security. – 2023.
15. *Shi J., Liu G., & Li C.* SAT-Based Security Evaluation for WARP against Linear Cryptanalysis // IET Information Security. – 2023.
16. *Collard Baudoin & Standaert François-Xavier.* Experimenting linear cryptanalysis // Cryptology and Information Security Series. – 2011. – 7. – 10.3233/978-1-60750-844-1-1.
17. ГОСТ 28147-89 Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. – М.: Стандартинформ, 1990.
18. *Matsui Mitsuru.* Linear Cryptoanalysis Method for DES Cipher // Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings. – 1993. – P. 386-397. – DOI: http://dx.doi.org/10.1007/3-540-48285-7_33.
19. *Biham E., & Shamir A.* Differential Cryptanalysis of the Data Encryption Standard. – Springer: New York, 1993.
20. *Massacci F., Marraro L.* Logical cryptanalysis as a SAT-problem: Encoding and analysis // In Journal of Automated Reasoning. – 2000. – 24. – P. 165-203.
21. 8-bit Atmel Microcontroller with 128Kbytes In-System Programmable Flash // ATmega128, ATmega128L. Rev. 2467X–AVR–06/11. 2011 Atmel Corporation. – URL: http://ww1.microchip.com/ downloads/en/devicedoc/doc2467.pdf.

REFERENCES

1. *Diffie W. and Hellman M.E*. Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard, *in Computer,* June 1977, Vol. 10, No. 6, pp. 74-84. DOI: 10.1109/C-M.1977.217750.
2. *Polikarpov S.V., Rumyantsev K.E., Kozhevnikov A.A.* Psevdo-dinamicheskie tablitsy podstanovki: osnova sovremennykh simmetrichnykh kriptoalgoritmov [Pseudo-dynamic substitution tables: the basis of modern symmetric cryptoalgorithms], *Nauchnoe obozrenie* [Scientific Review], 2014, No. 12, pp. 162-166. Available at: http://www.sced.ru/ru/files/7_12_1_2014/7_12_1_2014.pdf.
3. *Polikarpov S.V., Rumyantsev K.E., Prudnikov V.A.* Vysokoproizvoditel'naya psevdosluchaynaya funktsiya pCollapserARX256-32x2 [High-performance pseudorandom function pCollapserARX256-32x2], *XXIV nauchno-prakticheskaya konferentsiya «RusKripto'2022»* [XXIV scientific and practical conference "RusCrypto'2022"], 2022. Available at: https://ruscrypto.ru/resource/archive/rc2022/files/02_polikarpov_rumyantsev_prudnikov.pdf.
4. *Polikarpov S.V., Rumyantsev K.E., Prudnikov V.A.* Issledovanie svoystv miniversii psevdo-sluchaynoy funktsii pcollapser [Study of properties of miniversion of pseudo-random function pcollapser], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2022, No. 6 (230), pp. 148-162. DOI: 10.18522/2311-3103-2022-6-148-162.
5. *Polikarpov S.V., Rumyantsev K.E., Kozhevnikov A.A.* Issledovanie lineynykh kharakteristik psevdo-dinamicheskikh podstanovok [Study of linear characteristics of pseudo-dynamic substitutions], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2015, No. 5 (166), pp. 111-123. Available at: http://izv-tn.tti.sfedu.ru/wp-content/uploads/2015/5/11.pdf.
6. *Polikarpov S., Rumyantsev K., Petrov D.* Computationally efficient method for determining averaged distribution of differentials for pseudo-dynamic substitutions, *International Conference on Electrical, Electronics, Materials and Applied Science, AIP Conf. Proc., 1952*, eds. V. Rao, A. Ben, S. Bhukya. Amer. Inst. Phys., 2018, UNSP 020091. DOI: 10.1063/1.5032053.

7. *Beierle C. et al.* Alzette: A 64-Bit ARX-box, In: Micciancio, D., Ristenpart, T. (eds), *Advances in Cryptology – CRYPTO 2020: Lecture Notes in Computer Science,* 2020, Vol. 12172. Springer, Cham. Available at: https://doi.org/10.1007/978-3-030-56877-1_15.
8. *Dinu D., Corre Y.L., Khovratovich D. et al*. Triathlon of lightweight block ciphers for the Internet of things, *J Cryptogr Eng*., 2019, 9, pp. 283-302. Available at: https://doi.org/10.1007/s13389-018-0193-x.
9. *Beierle C., Biryukov A., Cardoso dos Santos L., Großschädl J., Perrin L., Udovenko A., Velichkov V., & Wang Q.* Lightweight AEAD and Hashing using the Sparkle Permutation Family, *IACR Transactions on Symmetric Cryptology,* 2020, 2020(S1), pp. 208-261. Available at: https://doi.org/10.13154/tosc.v2020.iS1.208-261.
10. *Beierle C., Biryukov A., Cardoso dos Santos L.* Schwaemm and Esch: Lightweight Authenticated Encryption and Hashing using the Sparkle Permutation Family. University of Luxembourg, 2019. Available at: https://sparkle-lwc.github.io/assets/sparkle-specification-latest.pdf.
11. *Beyne T.* A Geometric Approach to Linear Cryptanalysis, In: Tibouchi, M., Wang, H. (eds), *Advances in Cryptology – ASIACRYPT 2021: Lecture Notes in Computer Science,* Vol. 13090. Springer, Cham, 2021. Available at: https://doi.org/10.1007/978-3-030-92062-3_2.
12. *Ranea A., Rijmen V.* Characteristic Automated Search of Cryptographic Algorithms for Distinguishing Attacks (CASCADA), *IET Information Security,* 2022, 16 (6). DOI: 10.1049/ise2.12077. Available at: https://eprint.iacr.org/2022/513.pdf.
13. *Stachowiak S., Kurkowski M., & Soboń A.* New results in SAT – cryptanalysis of the AES, *2022 IEEE 16th International Scientific Conference on Informatics (Informatics),* 2022, pp. 280-286.
14. *Bellini E., Piccoli A.D., Formenti M., Gérault D., Huynh P., Pelizzola S., Polese S., & Visconti A.* Differential Cryptanalysis with SAT, SMT, MILP, and CP: A Detailed Comparison for Bit-Oriented Primitives, *Cryptology and Network Security,* 2023.
15. *Shi J., Liu G., & Li C.* SAT-Based Security Evaluation for WARP against Linear Cryptanalysis, *IET Information Security,* 2023.
16. *Collard Baudoin & Standaert François-Xavier.* Experimenting linear cryptanalysis, *Cryptology and Information Security Series,* 2011, 7. 10.3233/978-1-60750-844-1-1.
17. GOST 28147-89 Sistemy obrabotki informatsii. Zashchita kriptograficheskaya. Algoritm kriptograficheskogo preobrazovaniya [GOST 28147-89 Information processing systems. Cryptographic protection. Cryptographic transformation algorithm]. Moscow: Standartinform, 1990.
18. *Matsui Mitsuru.* Linear Cryptoanalysis Method for DES Cipher, *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings,* 1993, pp. 386-397. DOI: http://dx.doi.org/10.1007/3-540-48285-7_33.
19. *Biham E., & Shamir A.* Differential Cryptanalysis of the Data Encryption Standard. Springer: New York, 1993.
20. *Massacci F., Marraro L.* Logical cryptanalysis as a SAT-problem: Encoding and analysis, *In Journal of Automated Reasoning,* 2000, 24, pp. 165-203.
21. 8-bit Atmel Microcontroller with 128Kbytes In-System Programmable Flash, *ATmega128, ATmega128L. Rev. 2467X–AVR–06/11. 2011 Atmel Corporation*. Available at: http://ww1.microchip.com/ downloads/en/devicedoc/doc2467.pdf.

Статью рекомендовал к опубликованию д.т.н., профессор О.И. Шелухин.

Поликарпов Сергей Витальевич – Южный федеральный университет; e-mail: polikarpovsv@sfedu.ru, г. Таганрог, Россия; тел.: +79085159762; к.т.н.

Прудников Вадим Александрович – e-mail: pruvad@yandex.ru, тел.: +79198961427; старший преподаватель.

Румянцев Константин Евгеньевич – e-mail: rke2004@mail.ru; тел.: +79281827209; д.т.н.; профессор.

Polikarpov Sergey Vitalievich – Southern Federal University; e-mail: polikarpovsv@sfedu.ru; Taganrog, Russia; phone: +79085159762; cand. of eng. sc.

Prudnikov Vadim Aleksandrovich – e-mail: pruvad@yandex.ru; phone: +7 9198961427; senior lecturer.

Rumyantsev Konstantin Evgenyevich – e-mail: rke2004@mail.ru; phone: +79281827209; dr. of eng. sc.; professor.