DOI 10.52209/1609-1825_2022_4_486 УДК 004.056.55

Исследование алгоритма малоресурсной криптографии PRESENT

[1]***МАХАУРИ Артур Султанович,*** *магистрант, arthurmakhauri@gmail.com,*
[1]***СТОЛПОВСКИЙ Никита Владимирович,*** *магистрант, nikita.stolpovskiy@gmail.com,*
[1]****ТАЙЛАҚ Бибігүл Елжасқызы,*** *старший преподаватель, tailak_b@mail.ru,*
[1]***КОККОЗ Махаббат Мейрамкызы,*** *к.п.н., зав. кафедрой, Makhabbat_k@bk.ru,*
[1]НАО «Карагандинский технический университет имени Абылкаса Сагинова», Казахстан, Караганда, пр. Н. Назарбаева, 56,
**автор-корреспондент.*

Аннотация. *В связи с высокими темпами развития приложений, использующих недорогие и энергоэффективные устройства, возникла необходимость обеспечения безопасности передаваемой информации в устройствах микроконтроллерного типа с использованием малоресурсных алгоритмов. В статье исследована реализация алгоритма «PRESENT», который на сегодняшний день обладает высокой криптостойкостью и имеет характеристики, которые делают его пригодным для программирования устройств с низкой вычислительной способностью. Рассмотрены основные операции блочного шифра, структура алгоритма PRESENT, принцип его работы. Реализация алгоритма осуществлена на языке программирования Python 3.9. Проведена проверка алгоритма PRESENT на нескольких микроконтроллерах с разной разрядностью процессоров. Исследование было проведено с целью выявления необходимых условий для использования алгоритма в малоресурсных устройствах.*

Ключевые слова: *малоресурсная криптография, легковесная криптография, блочные шифры, алгоритм PRESENT, блочное шифрование, малоресурсные микроконтроллеры.*

Введение

Различные параметры устройств с ограниченной пропускной способностью влияют на выбор встроенных платформ микроконтроллеров, на основе которых можно реализовывать «облегченные» алгоритмы шифрования, при этом одним из главных преимуществ таких устройств является низкая цена [1].

Большинство микроконтроллерных платформ ориентированы на использование в беспроводной сети с низкой пропускной способностью. Например, это свойственно популярным сенсорным сетям PAN, RFID, XBEE.

В статье рассмотрен блочный алгоритм малоресурсной криптографии PRESENT, также проведено тестирование его реализации при использовании во встраиваемых микроконтроллерных системах.

Алгоритм PRESENT является алгоритмом блочного шифрования уменьшенного размера, т.к. он использует блоки подстановки и перестановки всего по 4 бита, а также сравнительно небольшой размер ключа по сравнению с другими алгоритмами того же типа. Эта особенность, вместе с его уменьшенным количеством раундов, делает его алгоритмом, который уравновешивает использование внутренней памяти микроконтроллера с достигнутой пропускной способностью [2].

Для соблюдения информационной безопасности в системах такого типа, важно выбрать независимую и понятную схему шифрования данных, с четко определенным алгоритмом, секретным ключом и его передачей [3].

В нашем случае была выбрана симметричная схема шифрования, в ней используется один секретный ключ, который известен отправителю и получателю, в результате получаем преимущества в виде низкой сложности выполнения, что позволяет использовать в высокоскоростных системах, например, телематических системах или в системах с ограниченными вычислительными ресурсами.

Основные операции блочного шифра

Существует два варианта шифров [4]:

- потоковое шифрование: данный вид генерирует непрерывный поток закодированных битов посредством непрерывного процесса преобразования непрерывного потока битов из открытого текстового сообщения;

- блочное шифрование: в этом типе шифрования сообщение разделяют на блоки по n бит, таким образом все блоки зашифрованы одинаково,

независимо от порядка передачи сообщения. Для того чтобы гарантировать «разрыв» возможных зависимостей, которые были в исходном сообщении, все биты блока сообщения задействованы во всех раундах шифрования.

Шифр PRESENT реализован по базовой методологии блочных шифров и имеет структуру, показанную на рисунке 1. Все блочные шифры основаны на генерации следующих комбинаций: ByteSub, ShiftRow, MixedAllColums и AddNewKeyRound или их эквивалентов.

Комбинация таких операций называется раундом. Операции раунда должны быть основаны на однородных и обратимых преобразованиях, называемых слоями, разработанными для стойкости от линейного и дифференциального криптоанализа, а именно:

- нелинейный слой: состоит в применении S-блоков параллельно с оптимальными свойствами нелинейности;
- линейный слой смешивания: гарантирует высокий уровень диффузии на протяжении нескольких раундов;
- слой добавления ключа: это уникальная операция OR (*ИЛИ*) между промежуточным этапом и уникальным ключом каждого раунда [5].

Структура алгоритма PRESENT

На рисунке 2 показана базовая структура алгоритма PRESENT, которая представлена функциональными блоками, из которых возможно легко написать в исходный код в программном и аппаратном обеспечении. Для шифрования сообщения потребуется выполнить 31 раунд.

Слой подстановки байтов: NewLayerSBox. Слой SBOX состоит из блока нелинейной подстановки, используется на каждом фрагменте матрицы этапа независимо, пример представлен в таблице 1. Этот слой генерирует новую информацию, никак не связанную с исходным сообщением. Этот блок замещения применяется к 16 полубайтам, завершающим 64 бита информации, что является стандартным размером блоков шифра. Для создания таблицы подстановки используется поиск в таблице с маской в 4-бита, основанной на информации в памяти микроконтроллера [6].

Битовая перестановка: newLayerP. Данный слой смешивает побитовую замену, 64-битный блок сообщения, где бит i-го раунда перемещается в позицию P(i). Порядок такой замены продемонстрирован в таблице 2.

Функция расширения ключа: addNewKeyRound. Этот блок отвечает за генерацию новых ключей для каждого раунда. В этой реализации была использована методология для 80-битных ключей, по этой причине был сгенерирован вектор K из 80 позиций, но в каждом раунде будут задействованы только 64 бита, имеющие высокий приоритет из нового ключа K_i следующего раунда. Способ, которым должен быть сделан обратный оборот, показан ниже:

$K_i = K_{63}K_{62}...K_0 = K_{79}K_{78}...K_0$.

Затем необходимо выполнить следующие операции в определенном порядке:

1. Битовая ротация входного ключа: $[K_{79}K_{78}...K_0] = [K_{18}K_{17}...K_{20}K_{19}]$.
2. Замена с помощью S-Box полубайта (4 бита) K_{78} на K_{76} ключа: $[K_{79}K_{78}K_{77}K_{76}] = S[K_{79}K_{78}K_{77}K_{76}]$.
3. Полубайт, добавляющий или смешивающий K_{19}-K_{15} ключа со счетчиком цикла, посредством операции XOR: $[K_{79}K_{78}K_{77}K_{76}] = S[K_{79}K_{78}K_{77}K_{76}]$.

Операция addRoundKey должна выполняться в каждом из необходимых раундов для шифрования сообщения, в противном случае необходимо учитывать, что в процессе дешифрования блоки

Рисунок 1 – Основные операции блочного шифра

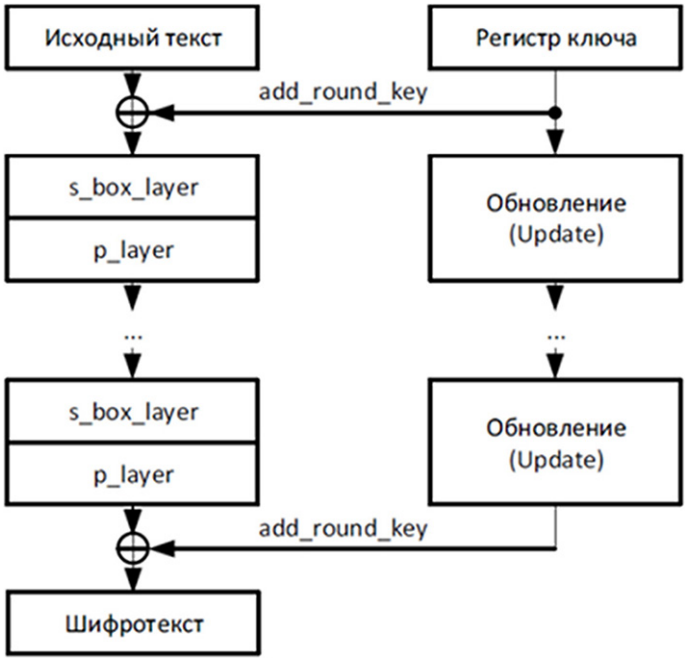

Рисунок 2 – Структура раунда алгоритма

Таблица 1 – Таблица S-BOX алгоритма PRESENT

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S(x)	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Таблица 2 – Таблица подстановки алгоритма PRESENT

ì	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
P(i)	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
ì	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
P(i)	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
ì	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
P(i)	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
ì	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
P(i)	12	28	55	60	13	29	45	61	14	30	46	62	15	31	47	63

берутся из списка ключей K_i до тех пор, пока не будет достигнуты начальные значения. Это означает, что последний ключ, использованный для шифрования, будет первым, используемым для дешифрования. Поэтому перед началом процесса дешифрования должны быть выполнены все раунды addNewKeyRound до тех пор, пока не будет получен последний K_i, выполняя инверсивный процесс до достижения оригинального ключа [7].

Реализация алгоритма

Большинство блочных или потоковых алгоритмов обычно имеет открытый доступ к деталям их реализации. При этом имеют высокую криптостойкость, связанную с тем, что знание реализации алгоритма не подразумевает знание данных для его дешифровки, открытых/закрытых ключей, раундов шифрования и т.д. [8].

На рисунке 3 представлен псевдокод алгоритма шифрования, который описывает принцип действия алгоритма, а также дает возможность реализовать его в качестве алгоритма шифрования для малоресурсных устройств [9].

Для замен LayerP и LayerSBox использовались таблицы из памяти малоресурсных контроллеров, что значительно сокращает скорость шифрования данных, но при этом увеличивает память, необходимую для реализации алгоритма.

Для реализации алгоритма были рассмотрены аналоги доступных платформ на рынке с учетом

их особенностей: архитектуры, цены, языков программирования, памяти и производительности.

По результатам этого анализа было решено провести тесты с несколькими устройствами от компании Microchip, которые на текущий момент имеют лидерство на мировом рынке. Для 8-битной проверки были использованы микроконтроллеры семейства 16F и 18F. Для 16-битной проверки были использованы микроконтроллеры семейства 24F. Данные устройства имеют возможности компиляции и симуляции с помощью систем виртуализации, а также инструменты проверки и тестирования. Выбором для 32 бит стала платформа ARM семейства Cortex-M0, в связи с тем, что она составляет непосредственную конкуренцию 8-битным микроконтроллерам. Основная работа была проведена на платформе KL25Z, в виду того, что она имеет доступный онлайн компилятор, который поддерживает языки программирования C и C++, динамическое выделение памяти и систему контроля версий. Малоресурсные контроллеры ARM используют онлайн платформу, что позволяет выполнять сборку и компиляцию независимо от аппаратного обеспечения или разработчика данного устройства, что является большим преимуществом, т.к. контроллеры являются полностью универсальными и подходят для решения задач, связанных с реализацией алгоритмов шифрования.

Безопасность, обеспечиваемая алгоритмом, зависит от сложности этих операций или количества раундов, необходимых для шифрования [10].

В случае алгоритма PRESENT перемешивание происходит непосредственно во время выполнения. Пример работы показан на рисунке 4, функционал реализован на языке Python 3.9.

Для алгоритма реализованы все необходимые функции, перечисленные ниже:

- def addNewKeyToRound(self, status, keyI):
- def newLayerSbox(self, status);
- def newLayerP(self, status);
- def getKeyI(self, keyI);
- def RoundKeys(self);
- def read_bit_n_bits(self);
- def write_bit_n_bits(self).

Для выполнения операций подстановки и перемешивания были разработаны специальные функции чтения и записи, написанные по-разному для каждой ширины шины процессора. Кроме того, они реализованы с учетом того, что если процессор имеет аппаратное обеспечение для управления битами, эти функции выполняют соответствующее шифрование в соответствии с разрядностью процессора.

После проверки правильности работы алгоритма на каждой из платформ были проведены тесты в реальном времени, первоначально для микроконтроллеров 8 бит (семейств 16F и 18F), затем для 16-битных микроконтроллеров (24FJ) и в конце в 32-битном микроконтроллере архитектуры ARM. Эти тесты проводились путем изменения данных с выходного PIN-контакта микроконтроллера каждый раз, когда алгоритм выполнял новое шифрование.

```
#Генерация ключей
def RoundKeys(self):
    for i in range(1, 31):
        KeyI = self.getKeyI(i)
        self.addNewKeyToRound(status=STATUS, keyI=KeyI)
        self.newLayerSbox(status=STATUS)
        self.newLayerP(status=STATUS)

    self.addNewKeyToRound(status=STATUS, keyI=32)
```

Рисунок 3 – Реализация алгоритма PRESENT на языке Python

```
ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  ТЕРМИНАЛ  КОНСОЛЬ ОТЛАДКИ  Python Debug Console
7\pythonFiles\lib\python\debugpy\launcher' '56086' '--' 'c:\Users\makhauri.artur\Untitled-1.py'
PRESENT Алгоритма - реализация

Введенный текст для шифрования: 0xFFFFFFFF FFFFFFFF

Шифрованный текст : 0x3333DCD3 213210D2

Дешифровать? - ДА

Дешифрованный текст : 0xFFFFFFFF FFFFFFFF
```

Рисунок 4 – Скриншот работы алгоритма на компьютере

После проведения тестов на различных микроконтроллерах было установлено, что алгоритм осуществляет функцию шифрования независимо от разрядности устройства. В результате тестирования реализации алгоритма были определены параметры его производительности – пропускная способность флэш-памяти и пропускная способность оперативной памяти. В таблице 3 приведены результаты для различных устройств.

Реализовано описание алгоритма на программном языке Python 3.9, который имеет библиотеки для работы на любой из выбранных платформ и имеет хорошую совместимость с различными архитектурами, особенно с 32-битными архитектурами ARM, которые имеют бит-ориентированные операции и благодаря этому достигают и превышают в 20 раз пропускную способность 8-битных микроконтроллеров при очень похожей стоимости.

Заключение

Малоресурсный шифр PRESENT представляет собой алгоритм, который обладает одним из самых компактных методов шифрования. Благодаря своим характеристикам, он используется в микроконтроллерах с низкой энергоэффективностью и производительностью. Работа алгоритма на различных платформах была изучена с целью выявления необходимых условий для использования алгоритма в малоресурсных устройствах. Алгоритм реализован и протестирован на четырех платформах различного типа. Также проведен анализ выполнения реализаций алгоритма в аппаратных микроконтроллерах с низкой производительностью.

Реализаций данного алгоритма на 32-битных микроконтроллерах в процессе исследования не выявлено, поэтому описанная реализация является основным достоинством данного исследования и является уникальной.

Таблица 3 – Результаты тестирования на выбранных микроконтроллерах

Микроконтроллер	Флэш память (байт)	Данные памяти RAM (байт)	Пропускная способность (мс)	Быстродействие (MIPS)
8 бит	2210	73	12	5
8 бит	2129	768	12,12	5
16 бит	3276	245	18,8	10
32 бит	20,5 k	600	234	48

СПИСОК ЛИТЕРАТУРЫ

1. Poschmann A. Y. Lightweight Cryptography: Cryptographic Engineering for a Pervasive World // Cryptology ePrint Archive. Report 2009/516, 2009.
2. Кнудсен Л.Р., Лэндер Г. PRESENT Сверхлегкий алгоритм. URL: www.lightweightcrypto.org/present/present_ches2007.pdf (дата обращения 28.03.22).
3. Адаменко М.В. Основы классической криптологии. Секреты шифров и кодов. – М.: ДМК Пресс, 2012. – 978 с.
4. Молдовян Н.А. Криптография: от примитивов к синтезу алгоритмов / Н. Молдовян, А. Молдовян, М. Еремеев. – М.: БХВ-Петербург, 2014. – 448 с.
5. Bogdanov A., Knudsen L.R., Leander G., Paar C., Poschmann A., Robshaw M.J.B., Seurin Y., Vikkelsoe C. PRESENT: An ultra-lightweight block cipher // CHES 2007. LNCS, vol. 4727, pp. 450-466. Springer, Heidelberg, 2007.
6. Жукова И.Ю. Стохастические методы и средства защиты информации в компьютерных системах и сетях. – М.: КУДИЦ-Пресс, 2019.
7. Амели Р. Шифр Present. Шифрование IP ядра. URL: https://opencores.org/websvn/filedetails?repname=present_encryptor&path=%2Fpresent_encryptor%2Ftrunk%2Fdoc%2Fpresent.pdf (дата обращения 28.03.22).
8. Бабенко Л.К., Беспалов Д.А., Макаревич О.Б., Чесноков Р.Д., Трубников Я.А. Разработка и исследование программно-аппаратного комплекса шифрования по алгоритму Present для решения задач малоресурсной криптографии // Известия ЮФУ. Технические науки. № 2, 2014. С. 174-180.
9. Жуков А.Е. Легковесная криптография. Ч. 1. // Вопросы кибербезопасности. № 1 (9). С. 26-43, 2015.
10. Simon and Speck: Block Ciphers for the Internet of Things. NIST Lightweight Cryptography Workshop (9 July 2015). URL: https://csrc.nist.gov/csrc/media/events/lightweightcryptography-workshop2015/documents/papers/session1-shors-paper.pdf (дата обращения 25.03.22).

PRESENT төмен ресурстық криптография алгоритмін зерттеу

[1]***МАХАУРИ Артур Султанович,*** *магистрант, arthurmakhauri@gmail.com,*
[1]***СТОЛПОВСКИЙ Никита Владимирович,*** *магистрант, nikita.stolpovskiy@gmail.com,*
[1]****ТАЙЛАҚ Бибігүл Елжасқызы,*** *аға оқытушы, tailak_b@mail.ru,*

[1]***КОККОЗ Махаббат Мейрамкызы,*** *п.ғ.к., кафедра меңгерушісі, Makhabbat_k@bk.ru,*
[1]*«Әбілқас Сағынов атындағы Қарағанды техникалық университеті» КеАҚ, Қазақстан, Қарағанды, Н. Назарбаев даңғылы, 56,*

**автор-корреспондент.*

Аңдатпа. *Қымбат емес және энергия үнемдейтін құрылғыларды қолданатын қосымшалардың жоғары даму қарқынына байланысты аз ресурсты алгоритмдерді қолдана отырып, микроконтроллер түріндегі құрылғыларда берілетін ақпараттың қауіпсіздігін қамтамасыз ету қажеттілігі туындады. Мақалада PRESENT алгоритмінің іске асырылуы зерттелген, ол қазіргі уақытта жоғары криптотөзімділікке ие және оны есептеу қабілеті төмен құрылғыларды бағдарламалау үшін қолайлы ететін сипаттамаларға ие. Блок шифрінің негізгі операциялары, PRESENT алгоритмінің құрылымы, оның жұмыс принципі қарастырылған. Алгоритм Python 3.9 бағдарламалау тілінде жүзеге асырылды. PRESENT алгоритмі әртүрлі процессорлық сыйымдылығы бар бірнеше микроконтроллерлерде тексерілді. Зерттеу алгоритмді аз ресурстық құрылғыларда қолдану үшін қажетті жағдайларды анықтау мақсатында жүргізілді.*

Кілт сөздер: *аз ресурсты криптография, жеңіл салмақты криптография, блоктық шифрлар, PRESENT алгоритмі, блоктық шифрлау, аз ресурсты микроконтроллерлер.*

Research of the Low-resource Cryptography Algorithm PRESENT

[1]***MAKHAURI Artur,*** *master student, arthurmakhauri@gmail.com,*
[1]***STOLPOVSKY Nikita,*** *master student, nikita.stolpovskiy@gmail.com,*
[1]****TAILAK Bibigul,*** *Senior Lecturer, tailak_b@mail.ru,*
[1]***KOKKOZ Mahabbat,*** *Cand. of Ped. Sci., Head of Department, Makhabbat_k@bk.ru,*
[1]*NPJSC «Abylkas Saginov Karaganda Technical University», Kazakhstan, Karaganda, N. Nazarbayev Avenue, 56,*

**corresponding author.*

Abstract. *Due to the high rate of development of applications using inexpensive and energy-efficient devices, it became necessary to ensure the security of transmitted information in microcontroller-type devices using low-resource algorithms. The article investigates the implementation of the «PRESENT» algorithm, which today has high cryptographic resistance and has characteristics that make it suitable for programming devices with low computing power. The main operations of the block cipher, the structure of the PRESENT algorithm, the principle of its operation are considered. The implementation of the algorithm is carried out in the Python 3.9 programming language. The PRESENT algorithm was tested on several microcontrollers with different bit depths of processors. The study was conducted to identify the necessary conditions for the use of the algorithm in low-resource devices.*

Keywords: *low-resource cryptography, lightweight cryptography, block ciphers, PRESENT algorithm, block encryption, low-resource microcontrollers.*

REFERENCES

1. Poschmann A. Y. Lightweight Cryptography: Cryptographic Engineering for a Pervasive World // Cryptology ePrint Archive. Report 2009/516, 2009.
2. Knudsen L.R., Lander G. PRESENT Sverhlegkij algoritm [PRESENT Ultralight algorithm]. URL: www.lightweightcrypto.org/present/present_ches2007.pdf (accessed 03/28/22).
3. Adamenko M.V. Osnovy klassicheskoj kriptologii. Sekrety shifrov i kodov [Fundamentals of classical cryptology. Secrets of ciphers and codes]. – Moscow: Publ. DMK Press, 2012. – 978 p.
4. Moldovyan N.A., et al. Kriptografija: ot primitivov k sintezu algoritmov [Cryptography: from primitives to the synthesis of algorithms]. – Moscow: Publ. BHV-Petersburg, 2014. – 448 p.
5. Bogdanov A., Knudsen L.R., Leander G., Paar C., Poschmann A., Robshaw M.J.B., Seurin Y., Vikkelsoe C. PRESENT: An ultra-lightweight block cipher // CHES 2007. LNCS, vol. 4727, pp. 450-466. Springer, Heidelberg, 2007.
6. Zhukova I.Yu. Stohasticheskie metody i sredstva zashhity informacii v komp'juternyh sistemah i setjah [Stochastic methods and means of information protection in computer systems and networks]. – Moscow: Publ. KUDITs-Press, 2019.
7. Amelie R. Shifr Present. Shifrovanie IP jadra [Cipher Present. Core IP Encryption]. URL: https://opencores.org/websvn/filedetails?repname =present_encryptor&path=%2Fpresent_encryptor%2Ftrunk%2Fdoc%2Fpresent.pdf (accessed 03/28/22).
8. Babenko L.K., Bespalov D.A., Makarevich O.B., Chesnokov R.D., Trubnikov Ya.A. Razrabotka i issledovanie programmno-apparatnogo kompleksa shifrovanija po algoritmu Present dlja reshenija zadach maloresursnoj kriptografii [Development and research of a hardware-software complex for encryption using the present algorithm for solving problems of low-resource cryptography]. – Moscow: Publ. Proceedings of the Southern Federal University. Technical science. 2014.
9. Zhukov A.E. Legkovesnaja kriptografija. Ch. 1. Voprosy kiberbezopasnosti [Lightweight cryptography. Part 1. Cybersecurity Issues]. Moscow, 2015. No. 1 (9). pp. 26-43.
10. Simon and Speck: Block Ciphers for the Internet of Things. NIST Lightweight Cryptography Workshop (9 July 2015). URL: https://csrc.nist.gov/csrc/media/events/lightweightcryptography-workshop2015/documents/papers/session1-shors-paper.pdf.