http://dx.doi.org/10.35596/1729-7648-2021-19-3-89-95

Оригинальная статья
Original paper

УДК 004.056.55

ПРИМЕР ВНУТРЕННЕЙ ФУНКЦИИ ДЛЯ СХЕМЫ SPONGE, ПОСТРОЕННОЙ НА ОСНОВЕ ОБОБЩЕННОЙ МЕТОДОЛОГИИ ПРОЕКТИРОВАНИЯ AES

Р.М. ОСПАНОВ, Е.Н. СЕЙТКУЛОВ, Б.Б. ЕРГАЛИЕВА, Н.М. СИСЕНОВ

Евразийский национальный университет имени Л.Н. Гумилева,

НИИ Институт информационной безопасности и криптологии (г. Нур-Султан, Казахстан)

Поступила в редакцию 18 марта 2021

Аннотация. Целью данной статьи является построение внутренней функции, лежащей в основе схемы “Sponge” для построения криптографических хеш-функций. Внутренняя функция в схеме “Sponge” является преобразованием фиксированной длины или перестановкой, оперирующей с фиксированным числом битов, составляющих внутреннее состояние функции. Существуют различные конструктивные подходы к проектированию функции. Наиболее распространенным является подход, при котором используется перестановка, основанная на симметричном блочном алгоритме шифрования с константами в качестве ключа. В данной статье строится внутренняя функция с помощью обобщенной методологии проектирования AES. Эта методология позволяет легко проектировать блочные шифры для зашифровывания больших блоков открытого текста с помощью небольших компонентов, представляя обрабатываемые данные в виде многомерных массивов. Внутренняя функция является блочным шифром, который обрабатывает 2048 битов, представляемых в виде 9-мерного массива из 512 4-битовых элементов размера $2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$. Каждый раунд шифрования состоит из трех преобразований (S-блоки, линейное преобразование и перестановка), аналогичных трем раундовым преобразованиям AES SubBytes, MixColumns и ShiftRows. Построенная функция может быть использована в качестве внутренней функции в модифицированной схеме “Sponge” построения криптографических хеш-функций.

Ключевые слова: криптография, хеш-функция, Sponge, схема, модификация, шифрование, МДР код, *S*-блок, перестановка, линейное преобразование, AES.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Благодарности. Данная работа выполнена при финансовой поддержке грантового финансирования МЦРИАП РК, № AP06851124.

Для цитирования. Оспанов Р.М., Сейткулов Е.Н., Ергалиева Б.Б., Сисенов Н.М. Пример внутренней функции для схемы Sponge, построенной на основе обобщенной методологии проектирования AES. Доклады БГУИР. 2021; 19(3): 89-95.

EXAMPLE OF INTERNAL FUNCTION FOR SPONGE SCHEME BUILT ON THE BASIS OF THE GENERALIZED AES DESIGN METHODOLOGY

RUSLAN M. OSPANOV, YERZHAN N. SEITKULOV, BANU B. YERGALIYEVA, NURBEK M. SISENOV

Gumilyov Eurasian National University,

Research Institute of Information Security and Cryptology (Nur-Sultan, Kazakhstan)

Submitted 18 March 2021

Abstract. The purpose of this article is to construct an internal function underlying the “Sponge” scheme for constructing cryptographic hash functions. An internal function in the “Sponge” scheme is a fixed-length transformation or permutation that operates on a fixed number of bits that make up the internal state of the function. There are various constructive approaches to function design. The most common approach is to use a permutation based on a symmetric block encryption algorithm with constants as the key. This article builds an internal function using the generalized AES design methodology. This methodology makes it easy to design block ciphers to encrypt large blocks of plaintext with small components, representing the processed data as multidimensional arrays. The internal function is a block cipher that processes 2048 bits, represented as a 9-dimensional array of 512 4-bit elements with size $2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$. Each round of encryption consists of three transformations (S-blocks, linear transformation, and permutation), similar to the three round transformations of AES SubBytes, MixColumns, and ShiftRows. The constructed function can be used as an internal function in the modified “Sponge” scheme for constructing cryptographic hash functions.

Keywords: cryptography, hash function, Sponge, scheme, modification, encryption, MDS code, *S*-box, permutation, linear transformation, AES.

Conflict of interests. The authors declare no conflict of interests.

Gratitude. This work was carried out with the financial support of grant funding from MDDIAI RK, No. AP06851124.

For citation. Ospanov R.M., Seitkulov Ye.N., Yergaliyeva B.B., Sisenov N.M. Example of internal function for Sponge scheme built on the basis of the generalized AES design methodology. Doklady BGUIR. 2021; 19(3): 89-95.

Введение

Схему “Sponge” [1] для построения криптографического алгоритма хеширования можно описать как последовательность следующих основных преобразований[1], в результате которых вычисляется хеш-значение заданного сообщения:

1. Дополнение (padding), при котором входное сообщение дополняется некоторым количеством битов так, чтобы длина дополненного сообщения была кратна заданной длине блока сообщения.
2. Инициализация состояния, при котором задается некоторое начальное значение состояния.
3. «Фаза впитывания» (absorbing phase), при котором сообщение сжимается итеративно.
4. «Фаза выжимания» (squeezing phase), при котором в результате требуемое хеш-значение сообщения извлекается.

[1] Bertoni G., Daemen J., Peeters M., Van Assche G. *Cryptographic sponge functions, Version 0.1.* January 14, 2011. URL.: https://keccak.team/files/CSF-0.1.pdf.

Существующие различные модификации схемы отличаются друг от друга различными способами дополнения, вариантами реализации инициализации состояния. Но основным и важным компонентом схемы “Sponge” является внутренняя функция, являющаяся преобразованием фиксированной длины или перестановкой, оперирующей с фиксированным числом битов, составляющих внутреннее состояние функции.

Существуют различные конструктивные подходы к проектированию функции. Наиболее распространенным является подход, при котором используется перестановка, основанная на симметричном блочном алгоритме шифрования с константами в качестве ключа. Так, например, в алгоритме Keccak перестановка построена как итерационный блочный шифр, подобный Noekeon и Rijndael, в котором раундовые ключи заменяются некоторыми простыми раундовыми константами. В алгоритме SPONGENT используется перестановка, представляющая собой модифицированную версию блочного шифра PRESENT. В алгоритмах Luffa, JH используется перестановка, основанная на блочном шифре с константами в качестве ключа. В алгоритме PHOTON используется AES-подобная перестановка. В алгоритме Bash используется перестановка, относящаяся к классу симметричных криптографических схем LRX (Logical-Rotation-Xor). В алгоритме ACE используется перестановка, представляющая собой бесключевой блочный шифр Simeck с уменьшенным количеством раундов. В алгоритмах SPN-Hash, GAGE, KNOT, SYCON, Ascon-Hash, Coral используются перестановки, итеративно применяющие раундовые преобразования SPN вида (подстановочно-перестановочные сети). В алгоритмах Esch256, Esch384 (SPARKLE) используются перестановки, основанные на ARX подходе и тесно связанные с блочным шифром Sparx. В алгоритме Rijndael256-Hash используется перестановка, основанная на блочном шифре Rijndael256, в котором ключом является константа 0. В алгоритме SIV-TEM-PHOTON-hash используется перестановка, основанная на блочном шифре TEM-PHOTON с константами в качестве ключа. В алгоритме SKINNY используются перестановки, основанные на семействе настраиваемых (tweakable) блочных шифров. В алгоритмах PHOTON-Beetle, ORANGISH используется AES-подобная 256-битная перестановка PHOTON256. В данной статье рассматривается внутренняя функция, построенная на основе обобщенной методологии проектирования AES[2].

Обобщенная методология проектирования AES

Криптографический алгоритм блочного симметричного шифрования AES представляет 128-битный блок зашифровываемого открытого текста в виде двумерного байтового массива размера 4 × 4 и производит преобразования над отдельными байтами массива и независимыми строками и столбцами массива. Алгоритм построен с помощью подстановочно-перестановочной сети и использует МДР код для обеспечения диффузии. МДР код в AES применяется к столбцам массива в четных раундах и к строкам в нечетных раундах (нумерация раундов начинается с 0). В работе Hongjun Wu[3] обобщили метод проектирования AES, представляя блок зашифровываемого открытого текста в виде d-мерного массива $\prod_{i=0}^{d-1} a_i \ (a_i \geq 2)$ m-битовых элементов ($m \in \mathbb{N}$), так чтобы было легко проектировать блочные шифры для зашифровывания больших блоков открытого текста с помощью небольших компонентов. МДР код применяется к ряду элементов ($r \bmod d$)-го измерения в r-м раунде (нумерация раундов начинается с 0). Например, для зашифровывания 512-битного блока открытого текста можно увеличить размерность AES до 3, представляя входные данные в виде трехмерного байтового массива размера 4 × 4 × 4. Тогда МДР код будет применяться к ряду байтов ($r \bmod 3$)-го измерения в r-м раунде. А для зашифровывания 2048-битного блока открытого текста можно увеличить размерность до 4, представляя входные данные в виде четырехмерного байтового массива размера 4 × 4 × 4 × 4. И МДР код будет применяться к ряду байтов ($r \bmod 4$)-го измерения в r-м раунде. На основе этой обобщенной методологии проектирования AES построена внутренняя функция

[2] Daeman J., Rijmen V. *AES Proposal: Rijndael*. 2003. URL.: https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf.
[3] Hongjun Wu. *The Hash Function JH.* 2011. URL.: https://www3.ntu.edu.sg/home/wuhj/research/jh/jh_round3.pdf.

криптографической хеш-функции JH. Внутренняя функция в JH является блочным шифром, который обрабатывает 1024 битов, представляемых в виде 8-мерного массива из 256 4-битовых элементов размера $2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$. Раундовыми ключами шифра являются константы, сгенерированные с помощью 6-размерного блочного шифра. Каждый раунд состоит из трех преобразований: *S*-блоки, линейное преобразование и перестановка (подобны трем раундовым преобразованиям AES SubBytes, MixColumns и ShiftRows). Структура внутренней функции предполагает несколько вариантов путем изменения размерности. Например, в случае размерности 9 можно зашифровывать 2048-битный блок. Далее рассмотрим детально конструкцию внутренней функции на основе обобщенной методологии проектирования AES с размерностью 9.

Структура функции

Внутренняя функция *F* итеративно использует *S*-блоки, линейное преобразование и перестановку, а также константы, определенные так же, как в [4]. Обрабатывается 2048 битов, представляемых в виде 9-мерного массива из 512 4-битовых элементов размера $2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$.

S-блоки.

Внутренняя функция *F* использует два 4-битовых *S*-блока S_0 и S_1, заданные табл. 1, 2.

Таблица 1. Блоки S_0

x	0	1	2	3	4	5	6	7	8	9	*A*	*B*	*C*	*D*	*E*	*F*
$S_0(x)$	9	0	4	*B*	*D*	*C*	3	*F*	1	*A*	2	6	7	5	8	*E*

Таблица 2. Блоки S_1

x	0	1	2	3	4	5	6	7	8	9	*A*	*B*	*C*	*D*	*E*	*F*
$S_1(x)$	3	*C*	6	*D*	5	7	1	9	*F*	2	0	4	*B*	*A*	*E*	8

Линейное преобразование *L*.

Внутренняя функция *F* использует линейное преобразование *L*, реализующее (4, 2, 3) МДР код над $GF(2^4)$, где умножение определяется как умножение двоичных многочленов по модулю неприводимого многочлена $x^4 + x + 1$. *L* преобразовывает пару 4-битовых слов $X = (x_0, x_1, x_2, x_3)$, $Y = (y_0, y_1, y_2, y_3)$ следующим образом:

$$L(x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3) = (x_0 + (y_1 + x_2), x_1 + (y_2 + x_3 + x_0), x_2 + (y_3 + x_0) + (y_0 + x_1), x_3 + (y_0 + x_1), y_0 + x_1, y_1 + x_2, y_2 + x_3 + x_0, y_3 + x_0)$$

Перестановка P.

Внутренняя функция *F* использует перестановку *P*. *P* – перестановка 512 элементов, являющаяся композицией трех перестановок 512 элементов $P = p_0 \circ p_1 \circ p_2$.

p_0 – перестановка 512 элементов, определяемая следующим образом:

$p_0(x_i) = x_i, i = 0, \ldots, 255,$

$p_0(x_{2i+0}) = x_{2i+1}, i = 128, \ldots, 255,$

$p_0(x_{2i+1}) = x_{2i+0}, i = 128, \ldots, 255.$

p_1 – перестановка 512 элементов, определяемая следующим образом:

$p_1(x_i) = x_{2i}, i = 0, \ldots, 255,$

$p_1(x_{i+256}) = x_{2i+1}, i = 0, \ldots, 255.$

p_2 – перестановка 512 элементов, определяемая следующим образом:

$p_2(x_{4i+0}) = x_{4i+0}, i = 0, \ldots, 127,$

$p_2(x_{4i+1}) = x_{4i+1}, i = 0, \ldots, 127,$

$p_2(x_{4i+2}) = x_{4i+3}, i = 0, \ldots, 127,$

$p_2(x_{4i+3}) = x_{4i+2}, i = 0, \ldots, 127.$

Раундовое преобразование R.

Внутренняя функция *F* использует раундовое преобразование *R*, выполняемое над 2048-битовыми словами. Раундовое преобразование $R(A, C)$ определяется на основе определенных выше *S*-блоков S_0 и S_1, линейного преобразования *L* и перестановки *P* следующим образом.

Пусть $A = (a_0 \| a_1 \| \cdots \| a_{511})$ – 2048-битовое слово, где a_i $(i = 0, \ldots, 511)$ – 4-битовые слова.

Пусть $C = (C_0 \| C_1 \| \cdots \| C_{511})$ – 512-битовое слово.

1. К слову *A* применяются *S*-блоки S_0 и S_1. Для каждого $i = 0, \ldots, 511$ a_i заменяется на $a'_i = S_0(a_i)$, если $C_i = 0$, или a_i заменяется на $a'_i = S_1(a_i)$, если $C_i = 1$.
2. К полученному слову $A' = (a'_0 \| a'_1 \| \cdots \| a'_{511})$ применяется линейное преобразование *L*. Для каждого $i = 0, \ldots, 255$ пара 4-битовых слов (a'_{2i}, a'_{2i+1}) заменяется на $(a''_{2i}, a''_{2i+1}) = L(a'_{2i}, a'_{2i+1},)$.
3. К полученному слову $A'' = (a''_0 \| a''_1 \| \cdots \| a''_{511})$ применяется перестановка *P*.

Раундовые константы.

Внутренняя функция *F* использует в качестве раундовых констант 512-битовые слова $C_r, r = 0, 1, \ldots, 47$, определяемые следующим образом.

C_0 – целая часть числа $(\sqrt{2} - 1) \times 2^{512}$.

Для генерации остальных констант используются преобразование P_0 и перестановка P_0.

P_0 – перестановка 128 элементов, аналогичная перестановке *P*, является композицией трех перестановок 128 элементов $P_0 = p'_0 \circ p'_1 \circ p'_2$.

p'_0 – перестановка 128 элементов, определяемая следующим образом:

$p'_0(x_i) = x_i, i = 0, \ldots, 63,$

$p'_0(x_{2i+0}) = x_{2i+1}, i = 32, \ldots, 63,$

$p'_0(x_{2i+1}) = x_{2i+0}, i = 32, \ldots, 63.$

p'_1 – перестановка 128 элементов, определяемая следующим образом:

$p'_1(x_i) = x_{2i}, i = 0, \ldots, 63,$

$p'_1(x_{i+64}) = x_{2i+1}, i = 0, \ldots, 63.$

p'_2 – перестановка 128 элементов, определяемая следующим образом:

$p'_2(x_{4i+0}) = x_{4i+0}, i = 0, \ldots, 31,$

$p'_2(x_{4i+1}) = x_{4i+1}, i = 0, \ldots, 31,$

$p'_2(x_{4i+2}) = x_{4i+3}, i = 0, \ldots, 31,$

$p'_2(x_{4i+3}) = x_{4i+2}, i = 0, \ldots, 31.$

R_0 – преобразование, выполняемое над 512-битовыми словами, аналогичное преобразованию *R*. R_0 определяется следующим образом.

Пусть $A = (a_0 \| a_1 \| \cdots \| a_{127})$ – 512-битовое слово, где $a_i (i = 0, \ldots, 127)$ – 4-битовые слова.

1. К слову *A* применяется *S*-блок S_0. Для каждого $i = 0, \ldots, 127$ a_i заменяется на $a'_i = S_0(a_i)$.
2. К полученному слову $A' = (a'_0 \| a'_1 \| \cdots \| a'_{127})$ применяется линейное преобразование *L*. Для каждого $i = 0, \ldots, 63$ пара 4-битовых слов (a'_{2i}, a'_{2i+1}) заменяется на $(a''_{2i}, a''_{2i+1}) = L(a'_{2i}, a'_{2i+1})$.
3. К полученному слову $A'' = (a''_0 \| a''_1 \| \cdots \| a''_{127})$ применяется перестановка P_0.

Константы $C_r = R_0(C_{r-1}),\ r = 1, 2, \ldots, 47$.

Внутренняя функция *F* выполняет над входными битами следующие преобразования.

1. Входные 2048 битов $A = (a_0 \| a_1 \| \cdots \| a_{2047})$ группируются по 4 бита. Получается 512 4-битовых слов (512 полубайтов) $Q_0 = (q_{0,0} \| q_{0,1} \| \cdots \| q_{0,511})$, где

$q_{0,2i} = a_i \| a_{i+512} \| a_{i+1024} \| a_{i+1536}, i = 0, \ldots, 255,$

$q_{0,2i+1} = a_{i+256} \| a_{i+768} \| a_{i+1280} \| a_{i+1792}, i = 0, \ldots, 255.$

2. Далее 48 раз применяется раундовое преобразование *R* с раундовыми константами C_r, получая в результате $Q_{r+1} = R(Q_r, C_r), r = 0, 1, \ldots, 47$.
3. Полученные в результате последнего раунда 512 4-битовых слов (512 полубайтов) $Q_{48} = (q_{48,0} \| q_{48,1} \| \cdots \| q_{48,511})$ разгруппировываются. Получаются выходные 2048 битов $B = (b_0 \| b_1 \| \cdots \| b_{2047})$, где

$b_i \| b_{i+512} \| b_{i+1024} \| b_{i+1536} = q_{48,2i}, i = 0, \ldots, 255,$

$b_{i+256} \| b_{i+768} \| b_{i+1280} \| b_{i+1792} = q_{48,2i+1}, i = 0, \ldots, 255.$

Заключение

В данной статье рассмотрен новый вариант внутренней функции, лежащей в основе перспективной современной схемы построения криптографических хеш-функций “Sponge” («криптографическая губка»). Рассматриваемый пример внутренней функции построен на основе обобщенной методологии проектирования AES. Внутренняя функция является блочным шифром, который обрабатывает 2048 битов, представляемых в виде 9-мерного массива из 512 4-битовых элементов размера 2 × 2 × 2 × 2 × 2 × 2 × 2 × 2 × 2. Каждый раунд шифрования состоит из трех преобразований (*S*-блоки, линейное преобразование и перестановка), аналогичных трем раундовым преобразованиям AES SubBytes, MixColumns и ShiftRows. Построенная функция может быть использована в качестве внутренней функции из заданного множества внутренних функций в модифицированной схеме “Sponge” построения криптографических хеш-функций [2–5].

Список литературы

1. Bertoni G., Daemen J., Peeters M., Van Assche G. *Sponge Functions*. Ecrypt Hash Workshop, 2007.
2. Оспанов Р.М., Сейткулов Е.Н., Арапов Н.К., Ергалиева Б.Б. Модификация схемы построения криптографических хеш-функций SPONGE. *Вестник КазНИТУ*. 2020;5(141):520-525.
3. Оспанов Р.М., Сейткулов Е.Н. Киберщит: О различных реализациях схемы построения криптографических хэш-функций “Sponge”. *Материалы Международной научно-практической Web-конференции «Военно-техническое обеспечение деятельности вооруженных сил: мировой опыт и тенденции развития»*. Нур-Султан: Из-во НУО; 2020: 305-308.
4. Оспанов Р.М., Сейткулов Е.Н. О способах проектирования внутренней функции для схемы построения криптографических хеш-функций SPONGE. *Вестник КазНИТУ*. 2020;5(141):645-650.
5. Оспанов Р.М. Киберщит: О внутренней функции в схеме построения криптографических хеш-функций “Sponge”. *Материалы Международной научно-практической Web-конференции «Военно-техническое обеспечение деятельности вооруженных сил: мировой опыт и тенденции развития»*. Нур-Султан: Из-во НУО; 2020: 351-353.

References

1. Bertoni G., Daemen J., Peeters M., Van Assche G. *Sponge Functions*. Ecrypt Hash Workshop, 2007.
2. Ospanov R.M., Seitkulov Ye.N., Arapov N.K., Yergalieva B.B. [Modification of the scheme for constructing cryptographic hash functions SPONGE]. *Bulletin of KazNTU*. 2020;5(141):520-525. (In Russ.)
3. Ospanov R.M., Seitkulov Ye.N. [Cybershield: On various implementations of the “Sponge” cryptographic hash function construction scheme]. *Materials of the International Scientific and Practical Web-Conference “Military-technical support of the armed forces: world experience and development trends”.* Nur-Sultan: From NUD; 2020: 305-308. (In Russ.)
4. Ospanov R.M., Seitkulov Ye.N. [On the ways of designing an internal function for the scheme for constructing cryptographic hash functions SPONGE]. *Bulletin of KazNRTU*. 2020;5(141):645-650 (In Russ.)
5. Ospanov R.M. [Cybershield: On the internal function in the scheme for constructing cryptographic hash functions “Sponge”]. *Materials of the International Scientific and Practical Web-conference “Military-technical support of the activities of the armed forces: world experience and development trends”,* Nur-Sultan: From NUD. 2020: 351-353. (In Russ.)

Вклад авторов

Оспанов Р.М. осуществил постановку задачи для проведения исследования, подготовил рукопись статьи.

Сейткулов Е.Н. определил цели исследования, разработал общую концепцию.

Ергалиева Б.Б. разработала обобщенную методологию проектирования AES.

Сисенов Н.М. описал структуру внутренней функции *F*, подготовил заключение.

Authors' contribution

Ospanov R.M. carried out the formulation of the problem for the study, prepared the manuscript of the article.

Seitkulov Ye.N. defined the objectives of the study, the development of a general concept.

Yergalieva B.B. developed a generic AES design methodology.

Sisenov N.M. described the structure of the internal function F, prepared the conclusion of the article.

Сведения об авторах

Оспанов Р.М., научный сотрудник НИИ ИБиК ЕНУ им. Л.Н. Гумилева.

Сейткулов Е.Н., к.ф.-м.н., профессор, директор НИИ ИБиК ЕНУ им. Л.Н. Гумилева.

Ергалиева Б.Б., младший научный сотрудник НИИ ИБиК ЕНУ им. Л.Н. Гумилева.

Сисенов Н.М., младший научный сотрудник НИИ ИБиК ЕНУ им. Л.Н. Гумилева.

Адрес для корреспонденции

010000, Республика Казахстан, г. Нур-Султан, ЕНУ им. Л.Н. Гумилева;
тел.: +7-7172-70-95-00;
e-mail: erj@mail.ru
Сейткулов Ержан Нураханович

Information about the authors

Ospanov R.M., Researcher of the Institute IS&C at the Gumilyov ENU.

Seitkulov Ye.N., PhD, Professor, Director of the Institute IS&C at the Gumilyov ENU.

Yergalieva B.B., Junior Researcher of the Institute IS&C at the Gumilyov ENU.

Sisenov N.M., Junior Researcher of the Institute IS&C at the Gumilyov ENU.

Address for correspondence

010000, Republic of Kazakhstan, Nur-Sultan, Gumilyov ENU;
tel.: +7-7172-70-95-00;
e-mail: erj@mail.ru
Seitkulov Yerzhan Nurakhanovich