СОВРЕМЕННЫЕ ПОДХОДЫ К ПОСТРОЕНИЮ ХЕШ-ФУНКЦИЙ НА ПРИМЕРЕ ФИНАЛИСТОВ КОНКУРСА SHA-3

***Авезова Яна Эдуардовна**, г. Москва*

В данной статье в хронологическом порядке развития рассмотрены основные принципы построения алгоритмов хеширования. Описана классическая конструкция Меркла–Дамгарда и ее развитие в схемах Девиса–Мейера, Матиса–Мейера–Осеаса, Миагучи–Пренеля, выделены их различия. Перечислены основные типы атак на конструкцию Меркла–Дамгарда. Описаны ее модификации Wide Pipe, рандомизированная схема, HAIFA и криптографическая губка.

Приведен перечень требований к претендентам на победу в конкурсе SHA–3. На примере финалистов конкурса SHA–3 – Skein, JH, Grostl, BLAKE и Keccak – выделяются конструктивные особенности современных хеш–функций, способных обеспечить устойчивость к широким классам атак. Приведены последние результаты криптоанализа финалистов SHA–3, позволяющие сделать вывод о темпах дальнейшего развития в области построения криптографических примитивов, составляющих основу стойких хеш–функций.

***Ключевые слова**: хеш–функции, SHA–3, конструкция Меркла–Дамгарда, криптографическая губка.*

MODERN APPROACHES TO HASH FUNCTIONS DESIGN ON THE EXAMPLE OF SHA-3 COMPETITION FINALISTS

***Yana Avezova**, Moscow*

In this paper author give an overview on the basic principles of hash functions design, which are considered in chronological order of their development. The classical structure of Merkle–Damgard construction and its progress in Davies–Meyer, Matyas–Meyer–Oseas, Miyaguchi–Preneel schemes are described, their differences are highlighted. Generic attacks on Merkle–Damgard construction are presented. Some modifications such as Wide Pipe, randomized hashing, HAIFA and sponge construction are introduced.

The paper also contains a list of requirements for SHA–3 challenge. On the example of SHA–3 competition finalists Skein, JH, Grostl, BLAKE and Keccak design features of modern hash functions which can provide resistance to a wide class of attacks are considered. The most recent results of SHA–3 finalists cryptanalysis are provided, which leads to the conclusion about the rate of further development in the cryptographic primitives design field which form the foundation of strong hash functions.

***Keywords**: hash functions, SHA–3, Merkle–Damgard, sponge construction.*

Введение

Хеш-функции используются в качестве строительного блока во многих приложениях. В 2004 году серия атак показала наличие уязвимостей в широко распространенном алгоритме SHA-1. В связи с этим NIST обратился с рекомендацией перейти к использованию SHA-2 и в 2007 году объявил о конкурсе для нового стандарта хеширования SHA-3.

Конкурс состоял из трех этапов. В заключительный раунд вышли пять финалистов: Skein, JH, Grøstl, BLAKE и Keccak. Все хеш-функции относятся к классу итеративных алгоритмов. Рассмотрим основные принципы построения хеш-функций, после чего перейдем к обзору непосредственно финалистов конкурса.

1. От Меркла-Дамгарда до криптографической губки

В 1976 году Диффи и Хеллман впервые подчеркнули необходимость построения однонаправленной функции как составной части схемы цифровой подписи [1]. Этот год можно считать отправной точкой развития хеш-функций.

1.1. Основа основ: конструкция Меркла-Дамгарда

Конструкция Меркла-Дамгарда [2] была опи-

сана Ральфом Мерклом в его кандидатской диссертации в 1979 году. Суть конструкции заключается в итеративном процессе последовательных преобразований, когда на вход каждой итерации поступает блок исходного текста и выход предыдущей итерации. Входная строка x разбивается на t одинаковых по длине блоков, длина блока x_i равна r. Длина блока r должна соответствовать длине входного блока функции сжатия f. Рассмотрим подробнее шаги схемы (Рисунок 1).

Разбить входные данные x на блоки $x_1,\ldots,x_t$.

Дополнить последний блок x_t нулевыми битами так, чтобы его длина равнялась r.

Добавить (t+1)-й блок, содержащий информацию о длине входных данных x.

Используя блок x_i в качестве аргумента функции сжатия f, получить промежуточное значение H_i.

H_i обеспечивает обратную связь для f и вместе с блоком x_i+1 используется в следующей итерации. Это предполагает наличие вектора инициализации H_0, определенного предварительно.

После обработки всех блоков функция g отображает предварительное значение H_{t+1} в окончательное значение хеш-суммы необходимой длины. Часто g – тождественная функция.

Роль функции сжатия может осуществлять любой блочный шифр E. Данная идея легла в основу развития конструкции Меркла-Дамгарда в схемах Девиса-Мейера, Матиса-Мейера-Осеаса, Миагучи-Пренеля (Рисунок 2).

Поясним схемы, приведенные на рисунке 2. Блок E отождествляется с блочным алгоритмом шифрования с размером блока в n бит, $E_k(z)$ – результат применения шифра E к блоку открытого текста z с использованием ключа k. Вход блока E, помеченный символом «•», отвечает за ввод ключа. Все схемы предполагают наличие начального значения H_0.

В схеме Девиса-Мейера блок сообщения x_i и предыдущее значение хеш-функции H_{i-1} поступают в качестве ключа и блока открытого текста соответственно на вход блочного шифра E. Получившийся в результате шифрования блок закрытого текста суммируется (операция XOR) с результатом предыдущей итерации хеширования (H_{i-1}). Результат суммирования H_i является результатом i-ой итерации хеш-функции на основе схемы Девиса-Мейера:

$$H_i = E_{x_i}(H_{i-1}) \oplus H_{i-1}.$$

В схеме Матиса-Мейера-Осеаса блок сообщения x_i и предыдущее значение хеш-функции H_{i-1} поступают в качестве блока открытого текста и ключа соответственно на вход блочного шифра E. Из-за возможных различий в размерах хеш-суммы и размере ключа шифра E значение H_{i-1} подвергается предварительной обработке функцией g, ре-

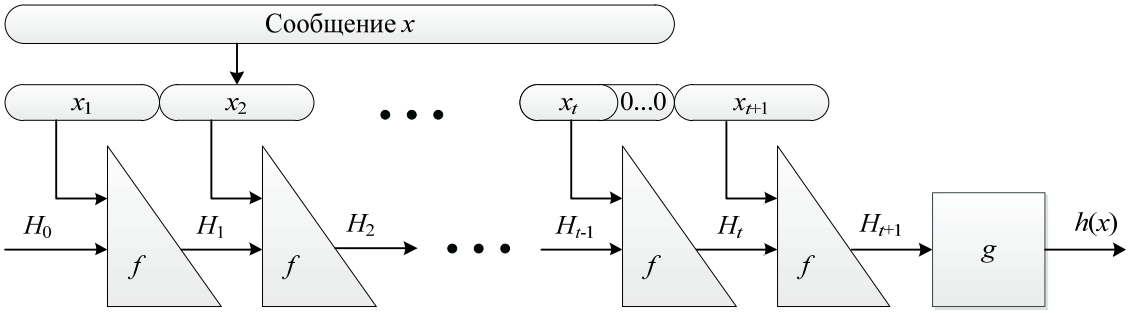

***Рис. 1**. Конструкция Меркла-Дамгарда*

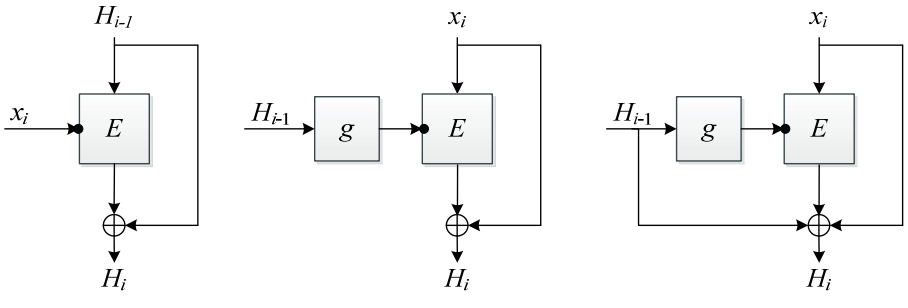

***Рис. 2**. Схемы Девиса-Мейера, Матиса-Мейера-Осеаса, Миагучи-Пренеля*

ализующей отображение n-битного значения хеш-функции в k-битный ключ шифра E. В результате применения операции шифрования, получается блок закрытого текста, который суммируется с соответствующим ему блоком открытого текста (x_i). Результат суммирования H_i является результатом i-ой итерации хеш-функции на основе схемы Матиса-Мейера-Осеаса:

$$H_i = E_{g(H_{i-1})}(x_i) \oplus x_i .$$

Схема Миагучи-Пренеля является расширенной версией схемы Матиса-Мейера-Осеаса. Отличие в том, что блок закрытого текста суммируется не только с соответствующим ему блоком открытого текста (x_i), но и с результатом предыдущей итерации хеширования (H_{i-1}).

$$H_i = E_{g(H_{i-1})}(x_i) \oplus x_i \oplus H_{i-1} .$$

Конструкция Меркла-Дамгарда используется в таких известных хеш-функциях, как MD5, SHA-1, SHA-2. Широкую популярность схема приобрела благодаря доказательству того, что устойчивость к коллизиям всей конструкции зависит только от устойчивости к коллизиям функции сжатия. Несмотря на популярность схемы Меркла-Дамгарда, ряд работ показал недостатки данной конструкции, связанные с множественными коллизиями [3], дополнением сообщения до нужной длины [4], нахождением второго прообраза [5]. Атаки направлены не на конкретный алгоритм, а применимы к любой хеш-функции, построенной по схеме Меркла-Дамгарда. Атака, основанная на множественных коллизиях [3], может быть осуществлена злоумышленником, даже если он имеет доступ к функции хеширования в целом, но не имеет возможности контролировать значения функции сжатия. Дополнение сообщения до нужной длины направлено на схему Девиса-Мейера, однако никак не затрагивает особенности самого блочного шифра, что делает эту атаку также достаточно общей. В связи с этим стали появляться предложения по модификации столь популярной конструкции.

1.2. Конструкция Wide Pipe

Если длина внутреннего состояния H_i совпадает с длиной хеш-суммы $\square(x)$, то коллизия $\square(x) = \square(\square)$ при входных данных $x \neq \square$ может быть обобщена на более длинные сообщения $(x||\square) \neq (\square||\square)$ добавлением одного и того же $\square$ к обоим исходным сообщениям. Используя это свойство, в работе [3] автор показал, что для поиска 2^k-коллизий требуется $k \cdot 2^{n/2}$ операций, где n – длина внутреннего состояния (она же – длина хеш-суммы).

Как меру борьбы с множественными коллизиями в 2004 г. Стефан Лакс предложил улучшенную версию схемы Меркла-Дамгарда [6]. Данная конструкция очень похожа на свою предшественницу. Отличие в том, что длина внутреннего состояния H_i больше (например, в два раза), чем длина хеш-суммы. Функция $\square$ осуществляет операции над блоком $H_{\square+1}$, в результате которых его длина изменяется до нужной длины хеш-суммы. Таким образом, идея Лакса была крайне простой, но она сделала поиск множественных коллизий ресурсоемким.

В 2010 году предложена модификация Fast Wide Pipe [7], которая позволила увеличить скорость вычислений по сравнению с Wide Pipe в два раза. Каждое значение H_i делится на две половины. Первая половина подается на вход функции сжатия, а вторая – суммируется с результатом той же итерации. Однако данная схема требует дополнительной памяти.

1.3. Рандомизированная схема

В 2006 году предложен способ усложнения поиска коллизий хеш-функций методом рандомизации входных данных для функции сжатия [8]. Такой способ позволил замаскировать коллизии в функции сжатия. Способ позиционируется авторами как отдельный режим работы криптосистемы хеширования без изменения самой ее конструкции. Может быть полезен в цифровых подписях для предотвращения сценария атаки нахождения второго прообраза.

1.4. Конструкция HAIFA

Конструкция HAsh Iterative FrAmework (HAIFA), предложенная в 2007 году, имеет ряд конструктивных особенностей [9]. Помимо нулей и информации о длине входного сообщения, входные данные дополняются *r* битами, кодирующими длину хеш-суммы. В качестве аргументов функции сжатия, кроме внутреннего состояния и очередного блока сообщения, используются количество бит, уже поступавших на вход функции сжатия на предыдущих итерациях, и «соль». Использование количества бит, которые уже поступали на вход функции сжатия на предыдущих итерациях, в качестве аргумента функции сжатия – мера противодействия атаке с использованием неподвижных точек. Добавление «соли» позволяет рассматривать данную конструкцию как экземпляр семейства рандомизированных хеш-функций и обеспечивает следующие преимущества:

- возможность оценить стойкость хеш-функции на теоретическом уровне;

- исключение атак, основанных на предварительных вычислениях;
- повышение стойкости цифровых подписей к атакам нахождения второго прообраза.

Данная конструкция позволяет получать хеш-суммы различных длин в рамках одного приложения. Используется универсальный вектор инициализации, предусмотренный разработчиком приложения, а вторичный вектор инициализации (длина которого соответствует желаемой длине хеш-суммы) получается как результат функции сжатия от универсального вектора и определенных значений остальных параметров.

1.5. Криптографическая губка

Конструкция «криптографическая губка» была разработана группой криптографов во главе с Джоном Даеменом с целью заменить устаревшую конструкцию Меркла-Дамгарда. Впервые представлена в 2007 году на симпозиуме ECRYPT. Представляет собой отображение входных данных переменной длины в выходные данные также переменной длины. Преобразование (или перестановка) f оперирует с фиксированным количеством бит $b = r + c$, где r называется битовой скоростью, а c мощностью. На начальном этапе, как и в конструкции Меркла-Дамгарда, входные данные расширяются в соответствии с заданным алгоритмом, после чего разбиваются на блоки по r бит. Далее b бит состояния инициализируются нулями.

Конструкция включает две фазы (Рисунок 3). В первой фазе (абсорбирование) r-битовые блоки сообщения суммируются (операция XOR) с первыми r битами внутреннего состояния – результата преобразования f. Когда эта операция проделана для всех блоков сообщения, фаза завершается. Далее, на фазе «отжима» первые r бит внутреннего состояния возвращаются в качестве выходных блоков f. Это действие повторяется, пока не будет получена желаемая длина хеш-суммы.

Внимательный взгляд на новую конструкцию позволяет заметить, что и она базируется на конструкции Меркла-Дамгарда, используя дополнительно операцию усечения.

2. SHA-3: лучшие из лучших

Активные исследования в области построения новых хеш-функций приходятся на 2007-2012 года, когда NIST проводил конкурсный отбор кандидатов на использование хеш-функции в качестве нового стандарта SHA-3. Новый стандарт SHA-3 должен поддерживать семейство алгоритмов, реализующих хеш-суммы длиной 224, 256, 384 и 512 бит. Как минимум, одна функция из семейства должна поддерживать хеш-код аутентификации сообщений (HMAC) и рандомизированное хеширование. Кроме того, для всех n бит хеш-суммы алгоритм должен обеспечивать выполнение следующих условий:

- устойчивость к нахождению прообраза для n бит;
- устойчивость к нахождению второго прообраза для $(n - L)$ бит, где первый прообраз имеет длину не более 2^L блоков;
- устойчивость к коллизиям для $n/2$ бит;
- устойчивость к атакам дополнением сообщения;
- для любого $m \leq n$ любое подмножество из m бит хеш-суммы длиной в n бит должно удовлетворять выше названным условиям.

2.1. Skein

Разработан в 2008 году группой авторов под руководством Брюса Шнайера. Алгоритм основывается на конструкции Меркла-Дамгарда с использованием в качестве функции сжатия настраиваемого блочного шифра Threefish по схеме

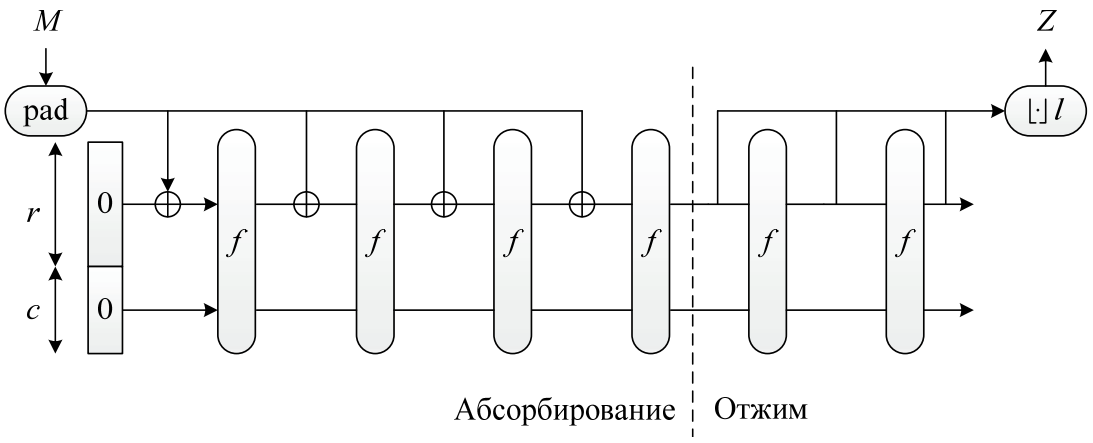

***Рис. 3**. Криптографическая губка*

Матиса-Мейера-Осеаса. Структура Threefish представляет собой 72-раундовую SP-сеть. Полное перемешивание входных данных достигается за 9-11 раундов. На последнем этапе соревнования была изменена константа в ключевом расписании с целью повысить устойчивость Threefish к криптоанализу.

Skein использует 64-битные операции, что позволяет легко распараллеливать его для вычисления на 64-разрядных процессорах. Кроме того, алгоритм хеширования может применяться в ограниченных в памяти и вычислительных мощностях устройствах (смарткарты, RFID-устройства и др.). В Skein не используются такие ресурсоемкие для процессора операции, как умножение или циклический сдвиг на переменное число бит.

Обращение к памяти – медленная операция, к тому же в ряде случаев она может привести к возникновению побочного канала для злоумышленника. Процессоры x64 содержат 15 регистров по 64 бита. Этого вполне достаточно для Threefish-256 и Threefish-512. Threefish-1024 необходимо 16 регистров, поэтому ему приходится делать небольшое количество обращений к памяти в каждом раунде.

Авторы использовали идею конструкции Wide Pipe как меру борьбы с множественными коллизиями: для хеш-функции Skein предусмотрены три возможные длины внутренних состояний: 256, 512 и 1024 бит.

В зависимости от параметров Skein можно использовать для различных приложений: цифровые подписи, создание кодов аутентификации, выработка ключей, генераторы псевдослучайных чисел, одноразовых паролей и cookies, проверка целостности, поточный шифр.

Дифференциальный криптоанализ Skein, проведенный в 2009 году, показал [10] существование псевдоколлизий плоть до 17 раунда и возможность проведения атак до 35 раунда. В 2010 году было показана [11] возможность проведения атаки на основе подобранного ключа на 57 раундов Threefish. На сегодняшний день нет данных в открытых источниках информации об успешных атаках на полный алгоритм хеширования Skein.

2.2. JH

Алгоритм разработан и представлен на конкурс SHA-3 в 2008 году сингапурским криптографом Wu Hongjun. Как и Skein, JH базируется на конструкции «криптографическая губка». Сжатие выполняется биективной функцией (блочный шифр с постоянным ключом). Прототипом для алгоритма шифрования стал AES. AES построен на основе SP-сети, входные данные представляют собой двумерный массив. В JH-алгоритме используется его модификация: AES обобщается до использования многомерных массивов, что позволило построить шифр с большой длиной блока на основе более маленьких компонентов. Изменяя параметр d, отвечающий за размерность массива входных данных, можно получать хеш-функции с различными характеристиками. В базовом варианте $d = 8$. Для эффективной аппаратной реализации авторы рекомендуют использовать $d = 6$. Размер блока сообщения при таком параметре составляет 128 бит, размер хеш-суммы – 256 бит. Обеспечивается устойчивость к атакам нахождения второго прообраза для сообщений длиной менее чем 2^{64} бит.

Рассмотрим подробнее, как происходит сжатие. Функции сжатия F_8 реализует фиксированную 1024-битную перестановку E_8 (Рисунок 4). Блок сообщения размером в 512 бит суммируется (операция XOR) с первой половиной входа перестановки E_8, а вторая половина – с выходом перестановки E_8. Перестановка E_8 включает два S-бокса размером 4×4, 8-битную линейную перестановку L и финальную перестановку P_8.

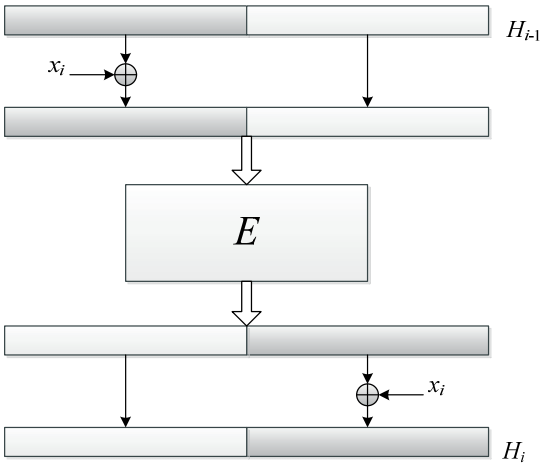

***Рис. 4**. Функция сжатия алгоритма JH*

Для борьбы с множественными коллизиями использована идея конструкции Wide Pipe (размер внутреннего состояния составляет 1024 бит).

Простота конструкция обеспечивается простой функцией сжатия (блочный шифр с постоянным ключом). Такая структура позволяет понизить сложность вычислений конструкции «криптографическая губка» за счет отсутствия необходимости усечения выходных данных.

Анализировать JH очень легко. Во-первых, поскольку ключ блочного шифра постоянен, нет внедрения дополнительных значений в функцию сжа-

тия, что позволяет провести анализ расхождения разностей достаточно легко. Во-вторых, блочный шифр основывается на AES, что упрощает дифференциальный криптоанализ. Наконец, в-третьих, конструкция обобщенного шифра на основе AES такова, что многомерные массивы могут быть оценены с помощью массивов меньшей размерности.

В 2011 году была предложена модификация алгоритма: количество раундов блочного шифра увеличилось до 42 для повышения эффективности реализации на аппаратных платформах и улучшения характеристик безопасности. Несмотря на это, в этом же году представлена атака на перестановку E_8, затронувшая все 42 раунда, а также атака на функцию сжатия, позволившая получить псевдоколлизии вплоть до 37 раунда [12]. Их временные сложности составляют 2^{304} и 2^{352} соответственно, что делает их непригодными для практики.

В 2013 году появились сведения о ряде атак на хеш-функции, среди которых оказалась и JH. Для нахождения прообраза JH-*s* злоумышленнику требуется $\frac{1}{e}2^{\frac{s}{2}+n}$ предвычислений и $2^{s/2}$ обращений к функции сжатия для нахождения прообраза.

2.3. Grøstl

Еще одним финалистом SHA-3 стал алгоритм Grøstl, разработанный в 2008 году датской командой криптографов. Базируется на конструкции Меркла-Дамгарда, используя идею Wide Pipe для устойчивости ко множественным коллизиям. Внутреннее состояние имеет длину 512 бит для Grøstl-256 и 1024 бит для Grøstl-512.

Функция сжатия (Рисунок 5) включает в себя две фиксированные $2n$-битные перестановки P и Q, основанные на конструкции AES. Используются S-боксы, расширенные до 512 бит для Grøstl-224, Grøstl-256 и до 1024 бит для Grøstl-384, Grøstl-512. Grøstl-256 включает 10 раундов, Grøstl-512 – 12 раундов.

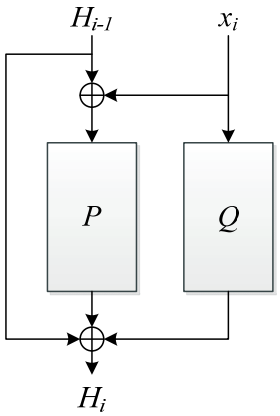

***Рис. 5**. Функция сжатия алгоритма Grøstl*

Финальная функция отсекает n бит от суммы перестановки P и ее аргумента (Рисунок 6).

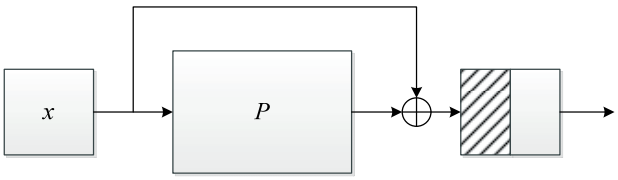

***Рис. 6**. Финальная функция алгоритма Grøstl*

Grøstl базируется на малом количестве перестановок (не использует блочный шифр), тогда как блочный шифр включает в себя большее количество операций. Может быть реализован с высокой производительностью на широком спектре платформ. Позиционируется как устойчивый к атакам по побочным каналам при небольших дополнениях и как устойчивый к атакам дополнением сообщения. Конструкция функции сжатия позволяет легко распараллелить алгоритм.

В 2011 году была показана атака, приводящая к коллизии за 2^{64} операций, но она применима только к 3 раундам Grøstl-224 и Grøstl-256 [13]. Атака на 10 раундов Grøstl-512, осуществленная в 2012 году, требует 2^{392} операций [14].

2.4. BLAKE

Алгоритм разработан командой из четырех криптографов из Швейцарии. В основе алгоритма BLAKE лежит конструкция HAIFA. Функция сжатия базируется на модифицированной схеме Девиса-Мейера. Блочный шифр оперирует внутренними состояниями, которые могут быть представлены квадратной матрицей слов порядка 4. BLAKE-224 и BLAKE-256 используют 512-битный блочный шифр, в котором блок представляется 32-битными словами, а BLAKE-384 и BLAKE-512 используют 1024-битную версию шифра с 64-битными словами. Функция сжатия G (Рисунок 7) основывается на поточном шифре ChaCha, она обновляет столбцы, используя ARX-операции. Слова входных данных и константы выбираются с помощью фиксированных перестановок σ_r в зависимости от номера раунда r.

В последнем раунде соревнования SHA-3 количество итераций функции сжатия было увеличено авторами с 10 до 14 для BLAKE-224 и BLAKE-256 и с 14 до 16 для BLAKE-384 и BLAKE-512.

Большинство работ по криптоанализу BLAKE были проведены еще до финального этапа соревнования, но несмотря на увеличенное количество итераций, они оставались актуальными. В 2010 году за 2^{21} удалось совершить атаку на 4 раунда

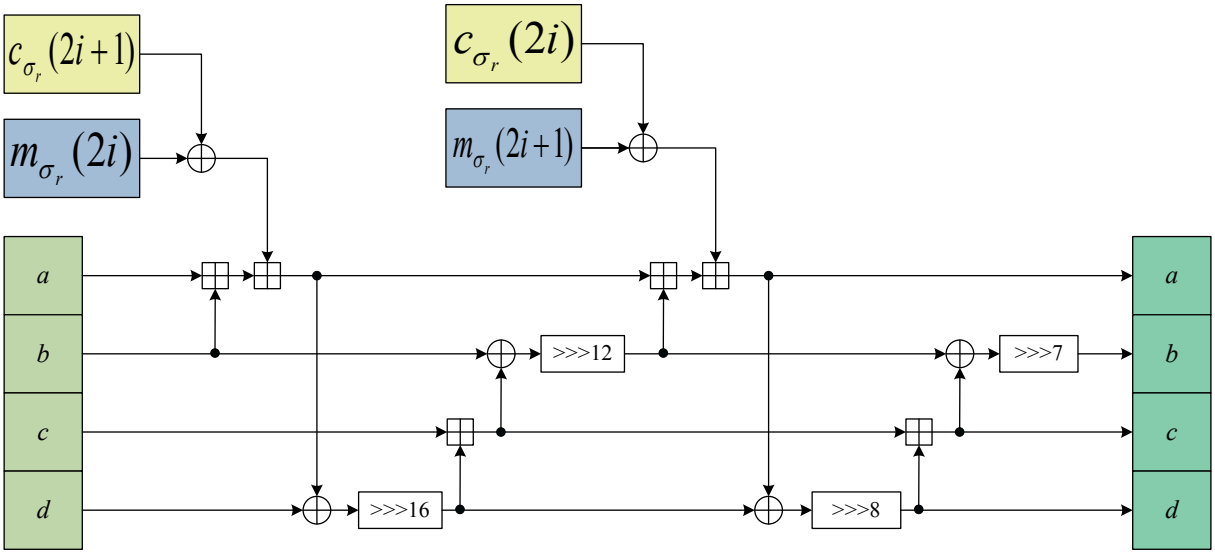

***Рис. 7**. Функция сжатия алгоритма BLAKE*

функции сжатия [15]. В 2011 году стала известна атака [16] на 8 раундов алгоритма шифрования, сложность которой оценивается в 2^{242}. Эти скромные показатели, пожалуй, лучшие из всех, которые были представлены в работах по криптоанализу BLAKE за годы проведения соревнования. В 2013 сложность атаки на 8 раундов удалось понизить до 2^{200} [17], а в прошедшем 2014 году – до 2^{198} [18].

Уже после завершения соревнования, в 2013 году, появился BLAKE2. Новая версия алгоритма имеет скорость, близкую к скорости MD5 на 64-битных платформах (в 2.53 раза быстрее, чем SHA-2), и требует на 33% меньше оперативной памяти, чем SHA-2, на устройствах с ограниченными ресурсами. Авторы достигли таких результатов, не сильно изменив конструкцию BLAKE. В частности, были изменены начальные установки для функции сжатия, оптимизированы под аппаратные платформы константы циклических сдвигов, исключены константы раундовой функции. Эти оптимизирующие преобразования привели к понижению теоретической стойкости. Успешная атака на все 12 раундов версии BLAKE2b имеет крайне высокую сложность 2^{876}. Анализ показал, что исключение констант из раундовой функции сделало стойкость хеш-функции сильно зависимой от правильного выбора вектора инициализации. В 2014 году появились теоретические атаки сразу на обе версии BLAKE2: BLAKE2s (7.5 раундов, сложность 2^{184}) и BLAKE2b (8.5 раундов, сложность 2^{474}). Такие показатели по-прежнему оставляют за BLAKE2 право считаться легковесной, быстрой и надежной хеш-функцией.

2.5. Keccak

Keccak, победитель конкурса SHA-3, имеет конструкцию «криптографической губки». Функция сжатия представляет собой 1600-битовую перестановку. Размер блока сообщения зависит от желаемого размера хеш-суммы: для Keccak-512, -384, -256, -224 блок сообщения имеет размер 576, 832, 1088 и 1152 бит соответственно. Также команда разработчиков Keccak предложила версии с уменьшенной длиной перестановки в 25, 50, 100, 200, 400 и 800 бит вместо 1600 бит и объявила об организации конкурса на поиск уязвимостей Keccak с параметром $c = 160$ бит.

Преобразование f может быть задано с помощью 5-битового S-бокса или композиции линейного перемешивающего преобразования и простого нелинейного перемешивающего преобразования. Изначально количество раундов равнялось 18, но было увеличено до 24 на последнем этапе конкурса.

Как и другие финалисты, Keccak – стойкая хеш-функция. Сложность атаки на все 24 раунда в 2010 году составила 2^{1590} [19] и была улучшена в 2011 до 2^{1579} [20]. В мае 2014 года на сайте конкурса по криптоанализу Keccak появилась новость об атаке с практической сложностью на 6 раундов Keccak в режиме поточного шифра или для вычисления кода аутентификации [21]. В режиме поточного шифра Keccak был атакован с параметрами $r = 1024$ и $c = 576$, сложность поиска 128-битного ключа составила 2^{36}. В режиме выработки имитовставки 80-битный ключ был взломан за 2^{35}. Организаторы конкурса предполагают, что атака может быть расширена еще на несколько раундов алгоритма.

Заключение

Развитие криптографических примитивов не стоит на месте. Как было показано, хеш-функции используются во многих приложениях и на различных платформах, что вынуждает нас предъявлять высокие требования к их стойкости. Поскольку большинство хеш-функций по-прежнему основываются на итеративных блочных шифрах, они тесно связаны: взлом шифра фактически ведет ко взлому хеш-алгоритма. Появление в 2014 году применимых на практике атак на редуцированный Keccak – победитель SHA-3 – позволяет сделать вывод о том, что в ближайшие несколько лет появятся новые, усовершенствованные криптографические алгоритмы.

Литература (References)

1. Whitfield Diffiee, Martin E. Hellman, «New directions in cryptography», IEEE Trans. on Information Theory, Vol. IT-22, No. 6, 1976, pp. 644-654.
2. Ivan Bjerre Damgård, «A design principle for hash functions», In G. Brassard, editor, Advances in Cryptology - CRYPTO '89, Vol. 435 of Lecture Notes in Computer Science, pp. 416-427, Springer, 1990.
3. Antoine Joux, «Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions», In M. Franklin, editor, Advances in Cryptology - CRYPTO 2004, Vol. 3152 of Lecture Notes in Computer Science, pp. 306–316, Springer, 2004.
4. John Kelsey, «A long-message attack on SHAx, MDx, Tiger, N-Hash, Whirlpool, and Snefru. Draft». Unpublished Manuscript.
5. Ueli Maurer, Renato Renner, Clemens Holenstein, «Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology», In M. Naor, editor, Theory of Cryptography, First Theory of Cryptography Conference, Vol. 2951 of Lecture Notes in Computer Science, pp. 21-39, Springer, 2004.
6. Stefan Lucks, «Design principles for iterated hash functions», Cryptology ePrint Archive, Report 2004/253, 2004.
7. Mridul Nandi, Souradyuti Paul, «Speeding Up the Widepipe: Secure and Fast Hashing», In Guang Gong and Kishan Gupta, editor, Indocrypt 2010, Vol. 6498 of Lecture Notes in Computer Science, pp. 144-162, Springer, 2010.
8. Shai Halevi, Hugo Krawczyk, «Strengthening Digital Signatures via Randomized Hashing», Advances in Cryptology - CRYPTO 2006, Vol. 4117 of Lecture Notes in Computer Science, pp. 41-59, Springer-Verlag, 2006.
9. Eli Biham , Orr Dunkelman «A Framework for Iterative Hash Functions - HAIFA», Cryptology ePrint Archive, Report 2007/278, 2007.
10. Jean-Philippe Aumasson, Çağdaş Çalık, Willi Meier, Onur Özen, Raphael C. -W. Phan, Kerem Varıcı, «Improved Cryptanalysis of Skein», In Mitsuru Matsui, editor, Advances in Cryptology - ASIACRYPT 2009, Vol. 5912 of Lecture Notes in Computer Science, 2009, pp. 542-559, Springer, 2009.
11. Dmitry Khovratovich, Ivica Nikolić, «Rotational Cryptanalysis of ARX», In Seokhie Hong, Tetsu Iwata, editors, Fast Software Encryption, Vol. 6147 of Lecture Notes in Computer Science, 2010, pp. 333-346, Springer, 2010.
12. María Naya-Plasencia, Deniz Toz, Kerem Varici, «Rebound Attack on JH42», Advances in Cryptology - ASIACRYPT 2011, Vol. 7073 of Lecture Notes in Computer Science, 2011, pp. 252-269, Springer, 2011.
13. Martin Schläffer, «Updated Differential Analysis of Grøstl», Grøstl website, 2011.
14. Jérémy Jean, María Naya-Plasencia, Thomas Peyrin, «Improved Rebound Attack on the Finalist Grøstl», In Anne Canteaut, editor, Fast Software Encryption, Vol. 7549 of Lecture Notes in Computer Science, 2012, pp. 110-126, Springer, 2012.
15. Bozhan Su, Wenling Wu, Shuang Wu, Le Dong, «Near-Collisions on the Reduced-Round Compression Functions of Skein and BLAKE», In Swee-Huay Heng, Rebecca N. Wright, Bok-Min Goi, editors, Cryptology and Network Security, Vol. 6467 of Lecture Notes in Computer Science, 2010, pp. 124-139, Springer, 2010.
16. Alex Biryukov, Ivica Nikolić, Arnab Roy, «Boomerang attacks on BLAKE-32», In Antoine Joux, editor, Fast Software Encryption, Vol. 6733 of Lecture Notes in Computer Science, 2011, pp. 218-237, Springer, 2011.
17. Dongxia Bai, Hongbo Yu, Gaoli Wang, Xiaoyun Wang, «Improved boomerang attacks on round-reduced SM3 and BLAKE-256», IET Information Security, 12 pp.
18. Yonglin Hao, «The Boomerang Attacks on BLAKE and BLAKE2», Cryptology ePrint Archive, Report 2014/1012, 2014.
19. Christina Boura, Anne Canteaut, Christophe De Cannière, «Higher Order Differential Properties of Keccak and Luffa», In Antoine Joux, editor, Fast Software Encryption, Vol. 6733 of Lecture Notes in Computer Science, 2011, pp. 252-269, Springer, 2011.
20. Ming Duan, XueJia Lai, «Improved Zero-Sum Distinguisher for Full Round Keccak-f Permutation», In Ming Duan, XueJia Lai, editors, Chinese Science Bulletin, Vol. 57, Issue 6, pp. 694-697, SP Science China Press, 2012.
21. Itai Dinur, Pawe Morawiecki, Josef Pieprzyk, Marian Srebrny, and Micha Straus, «Practical Complexity Cube Attacks on Round-Reduced Keccak Sponge Function», Cryptology ePrint Archive, Report 2014/259, 2014.