УДК 004.9 **DOI 10.52167/1609-1817-2022-122-3-360-375**

Р.М.Оспанов[1], Е.Н. Сейткулов[1], Б.Б. Ергалиева[1],
Балбаев Г.К.[1], А.Т. Ахмедиярова [2]
[1]Евразийский национальный университет им. Л.Н.Гумилева,
Астана, Казахстан
[2]Учреждение «Центр научных и научно-технических исследований «National Security»,
Алматы, Казахстан
E-mail: yerzhan.seitkulov@gmail.com

ОБ АНАЛИЗЕ БЕЗОПАСНОСТИ КРИПТОГРАФИЧЕСКОЙ ХЕШ-ФУНКЦИИ, ПОСТРОЕННОЙ НА ОСНОВЕ МОДИФИЦИРОВАННОЙ СХЕМЫ SPONGE

Аннотация. Данная работа посвящена анализу безопасности криптографической хеш-функции, основанной на модифицированной схеме «Sponge». Схема «Sponge» («криптографическая губка») - перспективная и популярная схема построения криптографических хеш-функций. Основным и важным компонентом схемы «Sponge» является внутренняя функция, являющейся преобразованием фиксированной длины или перестановкой, оперирующей с фиксированным числом битов, составляющих внутреннее состояние функции. Классическая схема «Sponge» и большинство ее модификаций предполагают в своем составе только одну внутреннюю функцию. Модифицированная схема «Sponge» предполагает использование уже множества внутренних функций. В работе приведено обоснование безопасности криптографической хеш-функции, основанной на модифицированной схеме «Sponge».

Ключевые слова. Информационная безопасность**,** криптографическая хеш-функция, схема Sponge, криптоанализ, стойкость к атакам.

Введение.

Понятие криптографических хеш-функций было впервые введено в работе [1] как важный строительный блок для повышения эффективности цифровых подписей. В настоящее время криптографические хеш-функции используются во множестве приложений, протоколов и криптографических схем, например, для цифровых подписей, для защиты с помощью паролей, для генерации случайных чисел, для получения ключей, для защиты целостности, для обнаружения вредоносного кода, для аутентификации сообщений и для многого другого.

Формально криптографические хеш-функции можно определить следующим образом.

Пусть Σ – некоторый алфавит.

Σ^n, $n \in \mathbb{N}$ – множество всех слов длины n над алфавитом Σ.

Σ^∞ - множество всех слов над алфавитом Σ.

Хеш-функцией называется эффективно вычислимая функция $H: \Sigma^\infty \to \Sigma^n$. Аргументы хеш-функции называются сообщениями. Значения хеш-функции называются хеш-значениями (соответствующих сообщений).

Далее пусть $\Sigma = \{0,1\}$.

Хеш-функция $H: \Sigma^\infty \to \Sigma^n$ называется стойкой к поиску первого прообраза, если вычислительно трудно найти для любого заданного хеш-значения его прообраз.

Хеш-функция $H: \Sigma^{\infty} \to \Sigma^{n}$ называется стойкой к поиску второго прообраза, если вычислительно трудно найти для заданного сообщения второе сообщение, хеш-значение которого равно хеш-значению заданного сообщения.

Хеш-функция $H: \Sigma^{\infty} \to \Sigma^{n}$ называется стойкой к поиску коллизий, если вычислительно трудно найти два различных сообщения, хеш-значения которых будут равны.

Хеш-функция $H: \Sigma^{\infty} \to \Sigma^{n}$ называется криптографической хеш-функцией, если

– она является стойкой к поиску первого прообраза;
– она является стойкой к поиску второго прообраза;
– она является стойкой к поиску коллизий.

Таким образом, криптографическая хеш-функция определяется следующими свойствами:

1) Сжатие данных (компрессия) (compression).

2) Простота вычислений (ease of computation).

3) Стойкость к поиску первого прообраза (необратимость) (Pre, preimage resistance) или однонаправленность (one-wayness).

4) Стойкость к поиску второго прообраза (Sec, 2nd preimage resistance) или стойкость к коллизиям первого рода (weak collision resistance).

5) Стойкость к коллизиям (CR, collision resistance) или стойкость к коллизиям второго рода (strong collision resistance).

С точки зрения необходимого уровня безопасности эти свойства часто связаны с длиной хеш-значения сообщения. Минимальные стандартные требования к безопасности современных криптографических хеш-функций с n-битным выходным размером определяются следующим образом.

Стойкость к коллизиям. Вычислительная сложность нахождения коллизий близка к $2^{\frac{n}{2}}$.

Стойкость к поиску первого прообраза (необратимость). Вычислительная сложность нахождения прообраза близка к 2^{n}.

Стойкость к поиску второго прообраза. Вычислительная сложность нахождения второго прообраза для любого сообщения длиной меньше $n - k$ бит близка к $2^{\frac{k}{2}}$.

Стойкость к атакам удлинением сообщения.

Псевдослучайность должно быть трудно отличить генератор псевдослучайных чисел на основе хеш-функции от генератора случайных чисел, например, он проходит обычные тесты на случайность.

Также между указанными выше свойствами существуют определенные связи. Например, известно, что свойство стойкости к коллизиям следует из стойкости к поиску второго прообраза, а при определенных условиях стойкость к коллизиям также следует из стойкости к поиску первого прообраза.

Кроме того существуют варианты стойкости к поиску первого прообраза (Pre) (everywhere preimage resistance) (ePre) и (always preimage resistance) (aPre) и варианты стойкости к поиску второго прообраза (Sec) (everywhere second-preimage resistance) (eSec) и (always second-preimage resistance) (aSec).

Существуют также более слабые версии стойкости к коллизиям: стойкость к почти-коллизиям (near-collision resistance), стойкость к коллизиям с полусвободным стартом (semi-free-start collision resistance), стойкость к псевдо-коллизиям (pseudo-collision resistance) (также называемая стойкостью к коллизиям со свободным стартом (free-start collision)). Эти версии стойкости к коллизиям определяются следующим образом.

Стойкость к почти-коллизиям (near-collision resistance). Хеш-функция $H: \Sigma^{\infty} \to \Sigma^{n}$ называется стойкой к поиску почти-коллизий, если вычислительно трудно найти два различных сообщения, хеш-значения которых будут равны только в некоторых частях.

Стойкость к коллизиям с полусвободным стартом (semi-free-start collision resistance). Хеш-функция $H: \Sigma^{\infty} \to \Sigma^{n}$ называется стойкой к поиску коллизий с полусвободным стартом, если вычислительно трудно найти два различных сообщения с одинаковыми начальными значениями, хеш-значения которых будут равны.

Стойкость к псевдо-коллизиям (pseudo-collision resistance) (также называемая стойкостью к коллизиям со свободным стартом (free-start collision)). Хеш-функция $H: \Sigma^{\infty} \to \Sigma^{n}$ называется стойкой к поиску коллизий со свободным стартом, если вычислительно трудно найти два различных сообщения с различными начальными значениями, хеш-значения которых будут равны.

Существует еще ряд различных свойств безопасности такие, как стойкость к поиску частичного прообраза (partial pre-image resistance), non-correlation (correlation freeness), Chosen Target Forced Prex (CTFP) pre-image resistance, сохранение кода аутентификации сообщения (message authentication code) (MAC) (unforgeability), сохранение псевдослучайной функции (pseudorandom function) (PRF), сохранение псевдослучайного оракула (pseudorandom oracle) (PRO), PrA (Pre-image Awareness), adaptive pre-image resistance, s-CR, s-Sec, s-aSec, s-eSec, s-Pre, s-aPre.

Еще одним ключевым понятием в определениях безопасности является понятие неразличимости [2]. Для доказательства безопасности криптографических хеш-функций можно использовать модель случайного оракула, т.е. идеализированную хеш-функцию. Понятие неразличимости формально фиксирует «расстояние» между двумя такими объектами (реальная хеш-функция и идеальная хеш-функция). Неформально это понятие можно описать следующим образом. Рассматривается противник, имеющий имеет доступ к любому из вышеуказанных двух объектов. Цель противника состоит в том, чтобы сделать различить реальную криптографическую хеш-функцию от идеальной. Противник обычно называется «различителем». Противник формализуется как некий эффективный алгоритм либо в теоретико-информационном смысле, либо в теоретико-сложностном.

Случайные оракулы (идеальные хеш-функции) являются монолитными объектами. В то же время реальные хеш-функции являются очень структурированными объектами. Понятие неразличимости не всегда отражает поведение структурированных функций, и взаимодействует с монолитными случайными оракулами. Более усовершенствованным родственным понятием, более распространенным в контексте криптографических хеш-функций является понятие недифференцируемости [3]. Недифференцируемость больше касается взаимодействия структурных компонентов, что делает это понятие более подходящей для анализа безопасности хеш-функций. Действительно, большинство последних хеш-функций используют в качестве строительных блоков внутренние функции, являющиеся либо функциями сжатия, либо внутренними перестановками. Отсутствие анализа безопасности, даже когда эти строительные блоки идеализированы, не только влияет на требования безопасности соответствующей хеш-функции, но также помогает указать на потенциальные недостатки в дизайне. Другими словами, недифференцируемость гарантирует, что режим работы не имеет конструктивных недостатков. Существуют также несколько вариантов недифференцируемости, где случайные оракулы заменены другими моделями. Граница недифференцируемости гарантирует, что до определенной степени криптографическая хеш-функция защищена от любой общей атаки, такой как поиск коллизий, прообразов, мультиколлизий, фиксированных точек и т. д.

Данная работа посвящена анализу безопасности криптографической хеш-функции, основанной на модифицированной схеме «Sponge», предложенной в работах [4,5]. Схема «Sponge» («криптографическая губка») - известная схема построения криптографических хеш-функций, предложенная в работах [6,7]. Основным и важным компонентом схемы «Sponge» является внутренняя функция, являющейся преобразованием фиксированной длины или перестановкой, оперирующей с фиксированным числом битов, составляющих внутреннее состояние функции. Классическая схема «Sponge» и большинство ее модификаций предполагают в своем составе только одну внутреннюю функцию. Модифицированная схема «Sponge» предполагает использование уже множества внутренних функций. В работе рассматриваются различные подходы к анализу безопасности криптографических хеш-функций и приводится обоснование безопасности криптографической хеш-функции, основанной на модифицированной схеме «Sponge».

Материалы и методы.

Существует два подхода к анализу безопасности криптографических хеш-функций - криптоаналитический подход и подход доказуемой безопасности.

Криптоаналитический подход состоит просто в попытке ее взломать. Это простой способ проанализировать криптографические хеш-функции. Этот подход представляет собой довольно значительное направление исследований в области симметричной криптографии. Основная идея криптоаналитического подхода состоит в том, чтобы попытаться найти коллизии, прообразы или вторые прообразы для рассматриваемой криптографической хеш-функции (упрощенной ее версии) быстрее, чем это ожидается для случайного оракула. В более общем случае попытка взлома криптографической хеш-функции состоит в нахождении некоторого неслучайного свойства хеш-функции.

Подход доказуемой безопасности состоит в попытке формального доказательства безопасности криптографической хеш-функции. При этом обычно основное внимание уделяется режимам работы. В частности, рассматривается способ конструирования криптографической хеш-функции на основе некоторой внутренней функции, в частности, функции сжатия, а также способ конструирования внутренней функции из некоторого симметричного блочного алгоритма шифрования или из некоторой перестановки. Существуют две модели доказательства безопасности: стандартная модель и идеальная модель. В случае стандартной модели безопасность всей схемы криптографической хеш-функции связывается с безопасностью используемого в схеме базового примитива. В случае идеальной модели предполагается, что используемый в схеме базовый примитив ведет себя случайным образом, и находится вероятность нарушения безопасности схемы злоумышленником. Также существует возможность комбинирования этих двух моделей. Обе модели имеют свои преимущества и недостатки. Стандартная модель не опирается на предположения об используемом базовом примитиве. В то же время доступные в рамках стандартной модели методы ограничены и сводятся в основном к поиску коллизий, прообразов и вторых прообразов функции сжатия. Идеальная модель опирается на строгие предположения об используемых базовых примитивах, но в то же время она дает гораздо больше возможностей, таких как оценка безопасности хеш-функции или функции сжатия в случае, когда в качестве базового примитива выступает блочный шифр или перестановка. Идеальная модель работает, когда стандартная не работает. Например, в работе [8] было показано, что не существует хеш-функции, доказуемо устойчивой к коллизиям, основанной только на односторонней перестановке. В частности, это означает, что стандартная модель безопасности для блочных шифров, псевдослучайных перестановок, недостаточна для доказательства безопасности хеш-функции на основе блочного шифра.

Если сравнивать с криптоаналитическим подходом, то иногда результаты, полученные при подходе доказуемой безопасности, могут быть спорными. С одной стороны, это связано с тем, что иногда очевидна безопасность используемого в рассматриваемой схеме режима работы, и для этого не требуется никаких доказательств. Хотя это может быть верным в случае достаточно простых конструкций, в то же время существуют более сложные конструкции, для которых наоборот не существует доказательства безопасности, или фактическая безопасность неизвестна. Также существуют конструкции, которые на первый взгляд кажутся безопасными, но при ближайшем рассмотрении оказываются ненадежными. В целом, несмотря на то, что доказуемые доказательства безопасности обеспечивают безопасность только режима работы, а не всей конструкции, они гарантируют, что конструкция не имеет структурных недостатков. Они подтверждают, что конструкция имеет надежный режим работы и что недостатки, если таковые имеются, могут возникать только из лежащего в основе примитива. Спорность результатов доказуемой безопасности также связано с утверждением, что идеальных примитивов не существует. Этот пункт можно разобрать на примере результатов, полученных в рамках доказуемой безопасности и при криптоанализе, в случае криптографической хеш-функции Keccak [9]. В рамках доказуемой безопасности показано, что эта хеш-функция ведет себя как идеальная хеш-функция вплоть до 2^n вычислений, где n – длина хеш-значения (в битах). Это дает гарантию, среди прочего, оптимальной защиты от успешного нахождения коллизий, прообразов и вторых прообразов, но только в том случае, когда базовая перестановка моделируется как случайная перестановка. Примерами криптоаналитических атак на Keccak являются атаки поиска коллизий с уменьшенным количеством раундов, где базовая перестановка ограничивалась тремя раундами (для n = 512), четыремя раундами (для n = 224, 384) или пятью раундами (для n = 256) (Базовая перестановка Keccak состоит из 24 раундов). Эти результаты, полученные в рамках двух разных подходов, не противоречат друг другу, так как относятся к разным моделям. Первый подход относится к модели идеальной перестановки, в которой предполагается, что базовая перестановка ведет себя как случайная перестановка. Второй подход связан с уменьшением количества раундов. Таким образом, в обоих случаях при исследовании безопасности рассматривается некая упрощенная версия хеш-функции Keccak, в которой базовая перестановка некоторым образом адаптирована для проведения анализа. С этой точки зрения оба подхода рассматривают алгоритм Keccak с разных сторон, и справедливо работать в абстрактной идеализированной модели по сравнению с криптоаналитическим подходом. Между рассматриваемыми подходами существует принципиальная разница. При криптоаналитическом подходе конечной целью является анализ всей полной конструкции, шаг за шагом увеличивая количество раундов. При подходе доказуемой безопасности же попытка отойти от идеальной модели приводят к техническим проблемам. Например, существуют примеры криптографических хеш-функций и внутренних функций (функций сжатия), являющиеся безопасными в идеальной модели, но не безопасные, когда идеальный примитив заменяется некоторой изобретенной безопасной функцией или даже некоторой известной используемой на практике функцией. Тем не менее, в этой области идеальная модель представляется наиболее подходящим способом анализа безопасности схем, основанных на блочных шифрах или перестановках.

Криптоанализ оценивает безопасность криптографических алгоритмов и пытается их «взломать». Функциональное назначение криптографического алгоритма определяет конкретные цели его криптоанализа. Так, например, в случае симметричных блочных и потоковых шифров, предназначенных в основном для шифрования, целью криптоанализа

является нахождение секретного ключа или получение информации в виде открытого текста. А в случае криптографических хеш-функций целью криптоанализа является поиск коллизий или восстановление части сообщения или всего сообщения из его хеш-значения. Эффективный алгоритм, целью которого является поиск слабых мест в криптографической схемы или в одном из ее основных примитивов, приводящих к нарушению свойств безопасности, называется атакой. Под атакой на криптографическую хеш-функцию понимают алгоритм, целью которого является нахождение уязвимости относительно одной из свойств безопасности хеш-функции. Атаки на хеш-функции делятся на группу общих атак, не зависящих от конкретной структуры алгоритма хеширования, и групп специальных атак, сосредоточенных на структурных уязвимостях и использующих определенные слабости алгоритма.

В случае общей атаки хеш-функция рассматривается как «черный ящик» и, таким образом, общая атака не зависит от структурных компонентов алгоритма. Сложность общей атаки зависит только от параметров криптографической хеш-функции. Для того, чтобы атаки такого типа были невозможны, разработчики криптографических хеш-функций должны выбирать подходящие параметры. В противном случае, невозможно предотвратить такие атаки.

В случае специальных атак рассматриваются свойства структурных компонентов хеш-функции или метод построения хеш-функции.

К общим атакам относятся атака исчерпывающего поиска, атака компромисса время-память, атака дней рождения, атака дней рождения без памяти, атака встреча посередине.

К специальным атакам относятся атака с фиксированной точкой, атака удлинением длины (сообщения), атака нахождения мультиколлизий, атака «herding».

Атака исчерпывающего поиска является самый простым и прямым способом поиска прообразов, а также применима для поиска вторых прообразов. Для нахождения прообразов злоумышленник осуществляет случайным образом перебор входных данных, вычисляя соответствующие хеш-значения, надеясь найти заданное хеш-значение. Вероятность успеха его поиска равна 1/|R|, где |R| - количество всех возможных хеш-значений. То есть, например, для n-битного хеша вероятность успеха равна 2^{-n} при условии, что каждое хеш-значение встречается приблизительно с одинаковой вероятностью. В случае поиска вторых прообразов, единственная разница в том, что злоумышленник заранее знает другое входное сообщение, хеш-значение которого является заданным значением. В случае нахождения прообразов (вторых прообразов) одновременно для 2^t сообщений сложность атаки снижается, и вероятность успеха становится равна 2^{-n+t}.

Атака компромисса время-память была впервые предложена в работе [10] в 1980 году. В случае этой атаки выполняются предварительные вычисления хеш-значений с сохранением их вместе с соответствующими сообщениями в таблице. Тем самым можно уменьшить временную сложность атака, вместо того, чтобы выполнять исчерпывающий поиск по всем возможным сообщениям. С другой стороны, увеличивается емкостная сложность атаки, поскольку предварительные вычисления требуют дополнительные затраты с точки зрения места для хранения данных. С помощью атаки компромисса время-память можно найти прообразы, а также она применима для поиска вторых прообразов. Злоумышленник выбирает определенное количество начальных точек из пространства сообщений и предварительно вычисляет хеш-цепочки для каждой из выбранных начальных точек (сообщений), сохраняя в памяти начальную и конечную точки каждой хеш-цепочки. Далее, выполняется та же итерация, до тех пор пока не будет найдена известная конечная точка. Таким образом, выполняется поиск прообраза. Однако,

возможно появление цепочек с разными начальными точками, совпадающих через какое-нибудь количество шагов, что приводит к уменьшению охвата значений. Существуют различные способы увеличения вероятности успеха.

Атака дней рождения основана на парадоксе дня рождения. Парадокс дней рождения утверждает, что в группе случайно выбранных людей по крайней мере у одной пары из них день рождения с высокой вероятностью будет совпадать. Существуют различные обобщения парадокса дней рождения. В случае криптографических хеш-функций эта задача эквивалентна задаче поиска коллизий. Злоумышленник выбирает набор из r случайных сообщений, сохраняет их хеш-значения в таблице и надеется, что, по крайней мере, два из них будут иметь одинаковое хеш-значение. Вероятность коллизии для n-битного хеша можно аппроксимировать следующим образом: $p \approx 1 - \exp\left(-\frac{r^2}{2|R|}\right) = 1 - \exp\left(-\frac{r^2}{2^{n+1}}\right)$.

Ожидаемое количество попыток, пока не будет найдено коллизия, равно $(\pi/2)^{1/2} \cdot 2^{n/2}$. Таким образом, временная сложность атаки дня рождения без учета константы составляет $2^{n/2}$ вычислений хеш-функции. Сложность памяти определяется размером таблицы, который также равен $2^{n/2}$. Существует вариант атаки дней рождения, в котором используется два набора различных сообщений для получения значимых коллизий. Злоумышленник выбирает r_1 сообщений и модифицирует их, чтобы получить результирующие r_2 сообщений. Если $r = r_1 = r_2 = 2^{n/2}$, вычислительная сложность этой атаки становится оптимальной.

У традиционной атаки дня рождения имеются большие требования к памяти, поскольку она требует хранения $2^{n/2}$ сообщений и соотвествующих хеш-значений. Существует вариант атаки дней рождения, при котором можно значительно уменьшить требования к большой памяти, используя алгоритм поиска цикла. Такая атака называется атакой дней рождения без памяти. Основная идея атаки основана на следующем факте. В случае конечных множеств, в частности, в случае множества $\{0,1\}^n$ при итеративном применении некой функции f, начиная со случайного значения x_0, получаемая последовательность значений x_1 = f(x_0), x_2 = f(x_1), … должна с некоторого момента повторяться, образуя тем самым с некоторого значения цикл. Следовательно, на входе в цикл функция f будет иметь коллизию.

Самый известный алгоритм поиска цикла – это алгоритм “черепахи и зайца” или алгоритм Флойда [11]. В алгоритме используется две последовательности, полученные итеративным применением функции f один раз и два раза за один шаг соответственно. Результаты каждого шага сравниваются. Когда на каком-то шаге два значения совпадают, $x_i = x_{2i}$, обнаруживается коллизия, но ни входные значения, приводящие к коллизии, ни точка входа в цикл неизвестны. Эту проблему можно решить, повторив еще раз, начиная с x_0 и x_i, но применяя f только один раз для обеих последовательностей.

Ожидаемая длина последовательности до входа в цикл и наибольшая длина цикла равны $(\pi/8)^{1/2} \cdot 2^{n/2}$. Временная сложность атаки примерно в три раза больше, чем временная сложность стандартной атаки дней рождения с незначительными требованиями к памяти.

Еще одним вариантом атаки дней рождения является атака встреча посередине. Эта атака применяется к хеш-функциям с легко обратимыми функциями сжатия. Она позволяет найти прообраз, второй прообраз или коллизию для промежуточных значений вместо хеш-значений. Злоумышленник выбирает r_1 начальных значений, вычисляет некоторые промежуточные соответствующие значения в прямом направлении и сохраняет

их в некотором списке. Затем он выбирает r_2 хеш-значений, вычисляет их некоторые обратные промежуточные значения и сравнивает со значениями, записанными в список. Вероятность совпадения промежуточного значения из двух разных наборов определяется выражением: $p \approx 1 - \exp\left(-\frac{r_1 \cdot r_2}{2^n}\right)$, где n — длина хеш-значений и промежуточных значений. Временная сложность атаки составляет r_1 вызовов функции, вычисляющей промежуточные значения в прямом направлении, и r_2 вызовов функции, вычисляющей обратные промежуточные значения, следовательно, она достигает минимального значения, когда $r_1 = r_2 = 2^{n/2}$, предполагая, что вычисления в обоих направлениях выполняются одинаковое количество времени. В сложности памяти преобладает размер списка, который также равен $2^{n/2}$.

Атака с фиксированной точкой, как следует из названия, предполагает существование фиксированных точек. Фиксированная точка — это пара (m_i, h_i), такая, что h_{i+1} = f (m_i, h_i) = h_i . Это означает, что значение сообщения m_i не влияет на значение цепочки h_{i+1} и, следовательно, на значение хеш-функции. Следовательно, можно построить вторые прообразы, удалив m_i. Точнее, при заданном сообщении m = m_1 m_2 . . . m_t, если злоумышленник найдет фиксированную точку для i-й итерации, то m = m_1 m_2 . . . m_{i-1} m_{i+1} ... m_t имеет такое же значение хеш-функции. Эта атака применима к хеш-функциям, имеющим функцию сжатия. Фиксированные точки можно легко найти для большинства функций сжатия.

Атака удлинением длины (сообщения) имеет своей целью хеш-значения сообщения m||m’ при условии знания хеш-значения сообщения m без знания самого сообщения m. Тривиальная атака на схему Меркла-Дамгарда, если нет выходного преобразования, может быть представлена следующим образом: последнее значение цепочки равно хеш-значению H(m), следовательно, злоумышленник может напрямую добавлять блоки сообщений и возобновлять вычисления, не зная об этом сообщении m.

При использовании усиленной схемы Меркла-Дамгарда злоумышленник больше не может напрямую применять атаку, описанную выше, но по-прежнему может выполнять атаку с расширением длины. При условии, что злоумышленник знает длину сообщения m, он может выбрать m’ таким образом, чтобы заполнение для m||m’ было закодировано в последних l битах m’.

Атака нахождения мультиколлизий является обобщением атаки поиска коллизии. Под мультиколлизией понимают множество сообщений, обладающих одним и тем же хеш-значением. В работе [12] было показано, что поиск мультиколлизий для итеративных хеш-функции не намного сложнее, чем поиск одной коллизии. Основная идея заключается в объединении цепочки внутренних коллизий. Злоумышленник вычисляет конфликтующие пары {m_i , m'_i} такие, что f (h_{i-1}, m_i) = f (h_{i-1}, m'_i) для t шагов. Тогда можно построить 2^t сообщений x_1 || . . . || x_t , где x i ∈ {m_i , m'_i}, которые хешируют до одного и того же значения.

Используя метод, представленный в работе [12], можно получить 2^t различных мультиколлизий для итеративных хеш-функции с вычислительной сложностью t × $2^{n/2}$. С другой стороны, для идеальной хеш-функции с n-битным хеш-значением считалось, что для нахождения t-кратной мультиколлии требуется $2^{n(t-1)/t}$ вызовов f. Однако в работе [13] было показано, что при таком количестве вызовов вероятность обнаружения t-кратной мультиколлизии составляет всего 1/t! что очень мало для больших t, и сложность должна быть увеличена в $(t!)^{1/t}$ раз, чтобы достичь вероятности успеха не менее 50%.

Атака «herding» [14], также известная как атака Chosen Target Forced Prefix (CFTP), позволяет злоумышленнику найти прообраз для еще неизвестного значения хеш-функции для схемы Меркла-Дамгарда с меньшей сложностью, чем атака прообраза. Он состоит из двух фаз. В автономной фазе злоумышленник выбирает 2^k значений цепочки и случайным образом генерирует блоки сообщений для вычисления новых значений цепочки, среди которых он ищет коллизии между парами. Процедура итеративно повторяется с новыми значениями цепочки и блоками сообщений до тех пор, пока не останется одна переменная цепочки, которую можно использовать в качестве хеш-значения. В результате злоумышленник находит множество коллизий, используя полный перебор, и формирует ромбовидную структуру, которая представляет собой структуру данных, напоминающую дерево. В онлайн-фазе злоумышленнику дается префикс P, и он тщательно ищет строку S', такую, что P || S' соответствует одному из промежуточных состояний этой структуры. Наконец, суффикс S, такой что H(P || S' || S) = h, может быть вычислен из структуры ромба.

Мультиколлизии, полученные этим методом, имеют ту же длину, но дороже, чем в работе [12]. Сложность создания ромбовидной структуры с 2^k хеш-значений в самом широком месте составляет не более $2^{\frac{k}{2}+\frac{n}{2}+2}$ вызовов функции сжатия f.

Дифференциальный криптоанализ, введенный в работе [15], является одним из самых мощных методов, используемых в случае блочных шифров, хеш-функций, потоковых шифров и т. д. Метод основан на проверке входных/выходных разностей и отношений между ними. В большинстве случаев эта разность является XOR, хотя можно использовать сложение по модулю 2^n, произвольные групповые операции или другие виды. С течением времени на основе дифференциального криптоанализа были разработаны многие другие методы криптоанализа, такие как, например, дифференциальный криптоанализ более высокого порядка, усеченный дифференциальный криптоанализ, невозможный дифференциальный криптоанализ и бумеранг-атака. Целью дифференциальной атаки на криптографические хеш-функции в основном является поиск коллизий. Например, для хеш-функций, основанных на блочных шифрах, где есть прямая связь открытого текста, разница на выходе должна быть равна разнице на входе, чтобы возникла коллизия. Все важные атаки коллизий на хорошо известные хеш-функции такие как MD4, MD5, RIPEMD, SHA-0, SHA-1, являются приложениями дифференциальной атаки. Еще одним важным применением дифференциальной атаки является поиск отличительных признаков для хеш-функций, их функций сжатия или даже построения. Введя разность на входе и наблюдая за ее распространением на выходе (или после некоторого количества раундов), злоумышленник может отличить функцию от случайного отображения; эта парадигма работает при условии, что вероятность получения ожидаемой разности выходных данных выше, чем вероятность наличия такой же разности выходных данных в случайной функции. Другими словами, если выходная разница с достаточно высокой вероятностью соответствует желаемому шаблону, злоумышленник может сказать, что хеш-значения генерируются заданной функцией.

Одним из алгоритмов криптоанализа криптографических хеш-функций, основанным на дифференциальном криптоанализе является так называемая Rebound-атака, введенная в работе [16]. Эта атака может быть рассмотрена как комбинация атаки «встреча посередине» и подхода «наизнанку» с использованием усеченных дифференциалов. Эта атака использовалась сначала для криптоанализа сокращенных версий криптографических хеш-функций Whirlpool и Grøstl, а затем была расширена для получения отличительных признаков полной функции сжатия алгоритма Whirlpool. Позже были представлены линеаризованные методы сопоставления посередине и начала с

середины для улучшения Rebound-атаки. Более того, разреженный усеченный дифференциальный путь использовался в атаке на алгоритм LANE, а не все активное состояние в соответствующей части атаки. Затем эти методы были использованы для улучшения результатов алгоритмов на основе AES (ECHO, Grøstl). Rebound-атака также успешно применялась к хеш-функциям, основанным на перестановках и на структурах Фейстеля.

Техника соединения и вырезания, представленная в работе [17], представляет собой усовершенствование атаки «встреча посередине». В этом методе первый и последний шаги целевого алгоритма рассматриваются как последовательные шаги. Затем алгоритм делится на две части, называемые порциями, таким образом, что каждая порция включает по крайней мере одно независимое слово сообщения от другой порции, называемой нейтральным словом. Наконец, псевдопрообразы можно найти с помощью атаки «встреча посередине». Этот метод позволяет злоумышленнику начать атаку «встреча посередине» с любого шага. Впервые он был использован для атаки на хеш-функции MD4 и MD5, после чего последовали атаки прообраза на HAVAL. Позже он был применен к хеш-функциям SHA-0 и SHA-1. Затем эта техника была усовершенствована и использовалась для атаки на полный алгоритм MD5.

Бумеранг-атака, введенная в работе [18], направлена на снижение сложности дифференциального криптоанализа. Основная идея заключается в использовании двух коротких дифференциальных характеристик с высокой вероятностью вместо одной длинной характеристики с меньшей вероятностью. Атака пытается создать структуру, называемую квартет, на полпути к шифру. Атака бумерангом применялась к внутренним функциям и версиям с уменьшенным числом циклов многих хеш-функций, таких как MD4, MD5 и HAVAL, SHA-1, BLAKE и Skein.

В то время как классический дифференциальный криптоанализ изучает распространение разностей между открытыми текстами, дифференциальный криптоанализ более высокого порядка использует распространение разностей между разностями. Понятие производных высших порядков было впервые введено в работе [19] и применено к дифференциальному криптоанализу в работе [20]. Дифференциальные атаки более высокого порядка применялись к хеш-функциям Keccak и Luffa, а коллизии второго порядка были обнаружены для хеш-функции SHA-256.

Линейный криптоанализ, предложенный в работе [21], является еще одним мощным методом, используемым при анализе криптографических систем. Он основан на изучении соотношения между линейными комбинациями битов открытого и зашифрованного текста. С момента своего появления линейный криптоанализ успешно применялся ко многим блочным и потоковым шифрам; с другой стороны, ему не уделялось особого внимания с точки зрения криптоанализа хеш-функций. Поскольку в хеш-функциях нет ключа для восстановления, линейный криптоанализ можно использовать только для оценки безопасности функции. Несколько примеров включают атаки на криптографические хеш-функции Hamsi и Cube Hash.

В алгебраическом криптоанализе шифр выражается в виде системы нелинейных уравнений над GF(2) или GF(2^n) с большим числом неизвестных переменных и ищется решение для системы. Хотя любой шифр можно описать в терминах многомерных уравнений, их решение представляет собой NP-сложную задачу даже для квадратных уравнений (известная как проблема MQ). Использование алгебраических методов в криптоанализе было впервые сформулировано Шенноном в 1949 г.: для взлома хорошего шифра требуется «такая же работа, как для решения системы одновременных уравнений с большим числом неизвестных сложного типа» [22]. В 1965 г. в работе [23] была представлена теория базиса Грёбнера, набора нелинейных многочленов от многих

переменных с определенными свойствами для решения алгебраических систем. Также был предложен алгоритм, который является обобщением алгоритма Евклида и исключения Гаусса для многомерных полиномиальных колец, для преобразования каждой полиномиальной системы в базисную форму Грёбнера. Позже в работах [24], [25] были представлены более эффективные алгоритмы для вычисления базисов Грёбнера, а именно алгоритмы F4 и F5. В работе [26] было показано, что сложность задачи MQ снижается, когда она становится переопределенной (т. е. существует больше уравнений, чем неизвестных переменных). Основная идея заключается в линеаризации системы путем замены любого произведения переменных новой переменной и решения полученной линейной системы. Этот метод, называемый релинеаризацией, направлен на решение систем квадратных уравнений за полиномиальное время. В работе [27] было показано, что многие уравнения, сгенерированные релинеаризацией, линейно зависимы, поэтому этот метод не так эффективен, как ожидалось, и они представили алгоритм расширенной линеаризации (XL). Основная идея алгоритма XL заключается в умножении каждого полиномиального уравнения на все возможные мономы (некоторой ограниченной степени) для создания вариантов более высокой степени, а затем линеаризации новой системы. Фактическая сложность алгоритма XL неизвестна (особенно для очень больших систем уравнений), и он дает небольшое преимущество по сравнению с методами базиса Грёбнера. В работе [28] утверждалось, что если MQ является разреженным и имеет регулярную структуру, то существует более эффективный метод поиска решения, называемый алгоритмом расширенной разреженной линеаризации (XSL). В этом алгоритме каждое уравнение умножается на «тщательно подобранные мономы» вместо всех возможных полиномов. Кроме того, существует последний шаг (называемый методом T), на котором пытаются получить новые линейно независимые уравнения, не создавая никаких новых мономов. Другим подходом, используемым в алгебраическом криптоанализе, является использование SAT-решателей, основанных на алгоритмах эвристического поиска. Основная идея заключается в том, чтобы угадать некоторые переменные и изучить последствия. Всякий раз, когда обнаруживается противоречие, может быть добавлено новое ограничение уравнения, говорящее, что в этом наборе ограничений одно из них ложно. В большинстве случаев SAT-решатели намного быстрее и могут сломать больше экземпляров, чем текущие методы баз Грёбнера.

Хотя решение больших систем многомерных полиномиальных уравнений для криптографических схем широко изучалось в алгебраическом криптоанализе, в [29] было заявлено, что эти уравнения не являются произвольными и несвязанными. Вместо этого они получаются из основного полинома путем фиксации значений некоторых общедоступных переменных, таких как биты сообщения и биты IV. Базовая кубическая атака представляет собой алгоритм решения таких уравнений. В кубической атаке этот алгоритм рассматривается как черный ящик, который вычисляет многочлен p над полем GF(2) из n+m входных битов (x_1,..., x_n; v_1,..., v_m), где x_i — секретные переменные (ключевые биты), а v_j — общедоступные переменные (биты открытого текста или биты IV). Далее для простоты различие между открытыми и закрытыми переменными игнорируется. Кубическую атаку можно рассматривать как расширение алгебраической IV дифференциальной атаки (AIDA) и дифференциалов более высокого порядка. Основная идея всех этих атак состоит в том, чтобы сгенерировать достаточное количество открытых текстов и суммировать соответствующие зашифрованные тексты, чтобы получить некоторые свойства, облегчающие атаку.

Результаты и обсуждение.

Классическая схема «Sponge» и большинство ее модификаций предполагают в своем составе только одну внутреннюю функцию. Модифицированная схема «Sponge» предполагает использование уже множества внутренних функций. Идея использования набора внутренних функций исходит из конструкции «Enhanced Sponge Function» [30]. Идея состоит в том, чтобы обеспечить псевдослучайное смешивание блоков сообщений для повышения уровня неопределенности или энтропии выходных битов. По мнению автора [30], такая схема позволяет сохранить базовую структуру схемы «Sponge». Соответственно, доказательства безопасности также сохраняются. В доказательствах безопасности, представленных в [6,7], единственной особенностью, отличающей случайную Sponge функцию от случайного оракула, является наличие внутренних коллизий. В работе [7] авторы предоставили верхнюю границу вероятности успеха злоумышленника, различающего случайный оракул (RO) и вывод Sponge функции. Эта граница покрывает противника, имеющего доступ к f и f^{-1}, и позволяет заменить случайный оракул случайной Sponge функцией в любом приложении. В случае же модифицированной схемы «Sponge», например, с тремя внутренними функцями f_0, f_1, f_2, злоумышленник должен иметь доступ к f_0, f_0^{-1}, f_1, f_1^{-1}, f_2, f_2^{-1} чтобы начать атаку. Другой важной характеристикой Sponge функции является вероятность того, что злоумышленник, не имеющий прямого доступа к перестановке, заданной f_0, f_1, f_2, сможет отличить случайный оракул от выходных данных «Sponge» функции.

Стоимость N запроса — это общее количество обращений к F, которые он выдаст, если X = S [F]. Это основано на том факте, что противник, которому представлена система X, является либо S [F], либо RO. Априорная вероятность того, что X является либо RO, либо S [F], равна 1/2. В работе [7] пришли к выводу, что отличительное преимущество противника, представленное алгоритмом A, определяется следующим образом: Adv (A) = Pr {A [S [F]] = 1} – Pr {A [RO] = 1}. Далее в работе [7] был продолжен анализ, чтобы определить верхнюю границу вероятности успеха типовых атак. Мы предполагаем, что это доказательство, развитое только для одной функции перестановки f, также применимо к случаю множества внутренних функций. На самом деле, в этом случае у противника будет более высокая стоимость запросов для успешной атаки. Идеальный случай, когда Adv (A) = 0, когда вероятности обнаружения равны. Рассматривая эту меру Adv (A), можно заметить, что она может давать отрицательную вероятность. Чтобы избежать этой ситуации, в работе [7] использовалось абсолютное значение уравнения. Однако можно визуализировать эту ситуацию, когда Pr {A [S [F]] = 1} < Pr {A [RO] = 1}, как дефектный алгоритм различения A, который возвращает RO выход как менее случайный, чем выход RO Sponge функции. Алгоритм A зависит от набора отправленных запросов и правила угадывания. Следовательно, для алгоритма A возможно, что набор запросов или правило угадывания, или и то и другое, могут отображать ситуацию, когда возникает отрицательная вероятность или Adv (A).

Результаты анализа позволяют говорить об отсутствии уязвимостей в рассматриваемом алгоритме хеширования. Таким образом, рассматриваемый алгоритм обладает высоким уровнем безопасности.

Заключение.

Данная работа посвящена анализу безопасности криптографической хеш-функции, основанной на модифицированной схеме «Sponge». Схема «Sponge» («криптографическая губка») – известная конструкция криптографических хеш-функций. Одним из основных составляющих схемы «Sponge» является внутренняя функция, являющейся преобразованием фиксированной длины или перестановкой, оперирующей с

фиксированным числом битов, составляющих внутреннее состояние функции. В данной работе приведены результаты статистического анализа криптографической хеш-функции, основанной на модифицированной схеме «Sponge». Проведенный анализ позволяет говорить, что рассматриваемый алгоритм обладает высоким уровнем безопасности.

Финансирование. Работа выполнена при финансовой поддержке МЦРИАП РК, № AP06851124.

ЛИТЕРАТУРА

[1] Diffie W., Hellman M. E. New Directions in Cryptography // IEEE Transactions on Information Theory 22, 6 (1976), 644–654.

[2] Maurer U. M. Indistinguishability of Random Systems // EUROCRYPT (2002), L. R. Knudsen, Ed., vol. 2332 of Lecture Notes in Computer Science, Springer, pp. 110–132.

[3] Maurer U. M., Renner R., Holenstein C. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology // TCC (2004), M. Naor, Ed., vol. 2951 of Lecture Notes in Computer Science, Springer, pp. 21–39.

[4] Оспанов Р.М., Сейткулов Е.Н., Ергалиева Б.Б. Пример криптографической хеш-функции, построенной на основе модифицированной схемы Sponge // Вестник КазНИТУ.- Алматы: КазНИТУ, 2021.-№143(1). - С. 247-259

[5] Ospanov R.M., Seitkulov Ye.N., Yergaliyeva B.B. A cryptographic hash function based on a modified Sponge scheme // Eurasian Journal of Mathematical and Computer Applications. Volume 10, Issue 2 (2022) pp. 55 –70

[6] Bertoni G., Daemen J., Peeters M., Van Assche G. Sponge functions // Ecrypt Hash Workshop 2007 (May 2007), http://www.csrc.nist.gov/pki/HashWorkshop/Public_Comments/2007_May.html

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

[8] Daniel R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? // Advances in Cryptology – EUROCRYPT'98, volume 1403 of Lecture Notes in Computer Science, pages 334–345, Espoo, Finland, May 31 – June 4, 1998. Springer, Berlin, Germany.

[9] Bertoni G., Daemen J, Peeters M., Van Assche G. The Keccak reference. SHA-3 competition (round 3), 2011, https://keccak.team/sponge_duplex.html.

[10] Hellman M. E. A Cryptanalytic Time-Memory Trade-Off // IEEE Transactions on Information Theory 26, 4 (1980), 401–406.

[11] Knuth D. E. The Art of Computer Programming, Volume II: Seminumerical Algorithms. Addison-Wesley, 1969.

[12] Joux A. Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions // CRYPTO (2004), M. K. Franklin, Ed., vol. 3152 of Lecture Notes in Computer Science, Springer, pp. 306–316.

[13] Suzuki K., Tonien D., Kurosawa K., Toyota K. Birthday Paradox for Multi-collisions // ICISC (2006), vol. 4296 of Lecture Notes in Computer Science, Springer, pp. 29–40.

[14] Kelsey J., Kohno T. Herding Hash Functions and the Nostradamus Attack // Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings (2006), vol. 4004 of Lecture Notes in Computer Science, Springer., pp. 183–200.

[15] Biham E., Shamir A. Differential Cryptanalysis of DES-like Cryptosystems // Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference,

Santa Barbara, California, USA, August 11-15, 1990, Proceedings (1991), vol. 537 of Lecture Notes in Computer Science, Springer., pp. 2–21.

[16] Mendel F., Rechberger C., Schläffer M., Thomsen, S. S. The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl // Fast Software Encryption, 16th International Workshop, FSE 2009, Leuven, Belgium, February 22-25, 2009, Revised Selected Papers (2009), vol. 5665 of Lecture Notes in Computer Science, Springer., pp. 260–276.

[17] Aoki K., Sasaki Y. Preimage Attacks on One-Block MD4, 63-Step MD5 and More // Selected Areas in Cryptography (2008), vol. 5381 of Lecture Notes in Computer Science, Springer, pp. 103–119.

[18] Wagner D. The Boomerang Attack // Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings (1999), vol. 1636 of Lecture Notes in Computer Science, Springer., pp. 156–170.

[19] Lai X. Higher Order Derivatives and Differential Cryptanalysis // Proc. "Symposium on Communication, Coding and Cryptography", in honor of J. L. Massey on the occasion of his 60'th birthday (1994), Kluwer Academic Publishers.

[20] Knudsen L. R. Truncated and Higher Order Differentials // FSE (1994), B. Preneel, Ed., vol. 1008 of Lecture Notes in Computer Science, Springer, pp. 196–211.

[21] Matsui M. 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 (1994), vol. 765 of Lecture Notes in Computer Science, Springer., pp. 386–397.

[22] Shannon C. Communication Theory of Secrecy Systems // Bell System Technical Journal, Vol 28 (October 1949), pp. 656–715.

[23] Buchberger B. Bruno Buchberger's PhD Thesis 1965: An Algorithm for Finding the Basis Elements of the Residue Class Ring of a Zero Dimensional Polynomial Ideal. Journal of Symbolic Computation 41, 3-4 (2006), pp. 475–511.

[24] Faugère J.-C.: A new efficient algorithm for computing Gröbner bases (F4) // Journal of Pure and Applied Algebra 139(1-3), pp. 61–88 (1999).

[25] Faugère J.-C. A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero (F5) // ISSAC'02: Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation (2002), pp. 75–83.

[26] Kipnis A., Shamir A. Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization // CRYPTO (1999), M. J. Wiener, Ed., vol. 1666 of Lecture Notes in Computer Science, Springer, pp. 19–30.

[27] Courtois N., Klimov A., Patarin J., Shamir A. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations // EUROCRYPT (2000), B. Preneel, Ed., vol. 1807 of Lecture Notes in Computer Science, Springer, pp. 392–407.

[28] Courtois N., Pieprzyk J. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations // ASIACRYPT (2002), Y. Zheng, Ed., vol. 2501 of Lecture Notes in Computer Science, Springer, pp. 267–287.

[29] Dinur I., Shamir A. Cube Attacks on Tweakable Black Box Polynomials // IACR Cryptology ePrint Archive 2008 (2008), 385.

[30] Saeb M.M., An Enhanced Sponge Function (ESP) // International Journal of Computer Science & Communications Security IJCSCS, July 2012. https://www.researchgate.net/publication/230646378_An_Enhanced_Sponge_Function_ESP

Руслан Оспанов, аға ғылыми қызметкер, Л. Н. Гумилев атындағы Еуразия ұлттық университеті, Астана, Қазақстан, ospanovrm@gmail.com

Ержан Сейтқұлов, ф.-м.ғ.к., профессор, Л. Н. Гумилев атындағы Еуразия ұлттық университеті, Астана, Қазақстан, yerzhan.seitkulov@gmail.com

Бану Ергалиева, докторант, Л. Н. Гумилев атындағы Еуразия ұлттық университеті, Астана, Қазақстан, banu.yergaliyeva@gmail.com

Ғани Балбаев, PhD, профессор, Л. Н. Гумилев атындағы Еуразия ұлттық университеті, Астана, Қазақстан, gani_b@mail.ru

Айнұр Ахмедиярова, PhD, "National Security" ғылыми және ғылыми-техникалық зерттеулер орталығы, Алматы, Қазақстан, aat.78@mail.ru

ӨЗГЕРІЛГЕН SPONGE СІЗБАСЫ НЕГІЗІНДЕ ҚҰРЫЛҒАН КРИПТОГРАФИЯЛЫҚ ХЭШ ФУНКЦИЯСЫНЫҢ ҚАУІПСІЗДІГІН ТАЛДАУ ТУРАЛЫ

Аңдатпа. Бұл жұмыс модификацияланған «Sponge» схемасы негізінде криптографиялық хэш функциясының қауіпсіздік талдауына арналған. «Sponge» схемасы («криптографиялық губка») криптографиялық хэш функцияларын құрудың перспективалы және танымал схемасы болып табылады. «Sponge» схемасының негізгі және маңызды құрамдас бөлігі - функцияның ішкі күйін құрайтын биттердің тіркелген санымен жұмыс істейтін тұрақты ұзындықты түрлендіру немесе ауыстыру болып табылатын ішкі функция. Классикалық «Sponge» схемасы және оның модификацияларының көпшілігі олардың құрамында тек бір ғана ішкі функцияны қамтиды. Өзгертілген «Sponge» схемасы қазірдің өзінде көптеген ішкі функцияларды пайдалануды қамтиды. Қағаз өзгертілген «Sponge» схемасы негізінде криптографиялық хэш-функцияның қауіпсіздігінің негіздемесін береді.

Түйінді сөздер. Ақпараттық қауіпсіздік, криптографиялық хэш-функция, Sponge схемасы, криптонализ, шабуылдарға төзімділік.

Ruslan Ospanov, senior researcher, L.N. Gumilyov Eurasian National University, Astana, Kazakhstan, ospanovrm@gmail.com

Yerzhan Seitkulov, candidate of physical and mathematical sciences, professor, L. N. Gumilyov Eurasian National University, Astana, Kazakhstan, yerzhan.seitkulov@gmail.com

Banu Yergaliyeva, doctoral student, L.N. Gumilyov Eurasian National University, Astana, Kazakhstan, banu.yergaliyeva@gmail.com

Gani Balbayev, PhD, professor, L.N. Gumilyov Eurasian National University, Astana, Kazakhstan, gani_b@mail.ru

Ainur Akhmediyarova, PhD, Center for Scientific and Scientific and Technical Research "National Security" Almaty, Kazakhstan, aat.78@mail.ru

ON THE ANALYSIS OF THE SECURITY OF A CRYPTOGRAPHIC HASH FUNCTION BUILT ON THE BASIS OF A MODIFIED SPONGE SCHEME

Annotation. This work is devoted to the security analysis of a cryptographic hash function based on a modified "Sponge" scheme. The "Sponge" scheme ("cryptographic sponge") is a promising and popular scheme for constructing cryptographic hash functions. The main and important component of the Sponge circuit is an internal function, which is a fixed-length transformation or permutation that operates on a fixed number of bits that make up the internal

state of the function. The classic "Sponge" scheme and most of its modifications involve only one internal function in their composition. The modified "Sponge" scheme already involves the use of many internal functions. The paper provides a justification for the security of a cryptographic hash function based on a modified "Sponge" scheme.

Keywords. Information security, cryptographic hash function, Sponge scheme, cryptanalysis, resistance to attacks.

****