МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

УДК 519.7 DOI 10.17223/20710410/63/2

КЛЮЧЕВОЙ КРИПТОАЛГОРИТМ ПО СХЕМЕ «СЭНДВИЧ» НА ОСНОВЕ ХЕШ-ФУНКЦИИ «СТРИБОГ»

В. А. Кирюхин*,**, А. М. Сергеев*

**ООО «СФБ Лаб», г. Москва, Россия*
***АО «ИнфоТеКС», г. Москва, Россия*

E-mail: vitaly.kiryukhin@sfblaboratory.ru, andrey.sergeev@sfblaboratory.ru

Предложен способ преобразования хеш-функции «Стрибог» в ключевой криптоалгоритм, условно называемый Стрибог-С («сэндвич» — ключ в начале и ключ в конце) и не требующий изменений в самой хеш-функции. Особенности криптоалгоритма упрощают реализацию мер защиты от атак по побочным каналам. Доказано, что Стрибог-С, а также HMAC-Стрибог и Стрибог-К являются стойкими псевдослучайными функциями (PRF) и алгоритмами имитозащиты (MAC) в условиях, когда (почти все) их внутренние состояния становятся известными противнику. От функции сжатия, итеративно применяемой внутри хеш-функции, в таких условиях требуются дополнительные свойства — стойкость к атакам на построение коллизий и прообраза.

Ключевые слова: *Стрибог, PRF, HMAC, доказуемая стойкость.*

"SANDWICH"-LIKE KEYED ALGORITHM BASED ON THE "STREEBOG" HASH FUNCTION

V. A. Kiryukhin*,**, A. M. Sergeev*

**LLC "SFB Lab", Moscow, Russia*
***JSC "InfoTeCS", Moscow, Russia*

We propose a keyed cryptographic algorithm based on the "Streebog" hash function. We do not make any structural changes to the hash function itself, but only introduce a special type of padding. As a result, the key appears on both sides of the message in so-called "sandwich" manner — hence the name Streebog-S for our construction. "Sandwich" properties make it possible to simplify defenses against side-channel attacks while maintaining their effectiveness. We prove that Streebog-S and other algorithms based on "Streebog", HMAC-Streebog and Streebog-K, remain secure as pseudorandom functions (PRF) and message authentication codes (MAC) even when almost all internal states are leaked to the adversary. This leakage resistance requires additional properties from the underlying compression function, namely collision- and preimage-resistance.

Keywords: *Streebog, PRF, HMAC, provable security.*

Введение

Российская бесключевая хеш-функция «Стрибог» (ГОСТ 34.11-2018) [1] основана на модифицированной схеме Меркла — Дамгарда (МД) [2, 3]. Хешируемое сообщение M дополняется специальным образом и разбивается на блоки по $n = 512$ бит, затем к n-битному состоянию хеш-функции h и блоку сообщения m итеративно применяется функция сжатия $\mathsf{g}(h, m) = h'$. Начальное состояние хеш-функции является предопределённой константой, последнее состояние — результат хеширования (хеш-значение).

Примечательной особенностью отечественной хеш-функции, отличающей её от оригинальной схемы МД, является использование контрольной суммы (КС). Последний хешируемый блок — сумма всех блоков сообщения M по модулю 2^n (операция «$\boxplus$»).

Указанный механизм играет важную роль, когда «Стрибог» (H) используется в качестве основы для *ключевых* криптоалгоритмов. Примерами таковых могут служить HMAC-Стрибог [4] (двойное хеширование) и Стрибог-К [5] (ключ K перед сообщением — $\mathsf{H}(K||M)$). Эти алгоритмы являются стойкими псевдослучайными функциями (PRF) [5, 6] и применяются для защиты целостности (имитозащиты), т. е. служат для выработки кодов аутентификации сообщений (MAC — Message Authentication Code).

Поясним влияние КС на ключевые схемы. Пусть секретный ключ K является одним из хешируемых блоков, тогда последним блоком (назовём его финализирующим ключом) будет $K \boxplus \sigma$. Значение σ может выбираться противником, так как в типовой модели угроз у атакующего есть возможность выбора текста M. Сообщения $M_1, \ldots, M_q$ дают КС $\sigma_1, \ldots, \sigma_q$ и соответственно *связанные* ключи $K \boxplus \sigma_1, \ldots, K \boxplus \sigma_q$.

Первое следствие этого — от функции сжатия g требуется [5, 6] быть стойкой PRF в условиях атак со связанными ключами (PRF-RKA). Конструктивные исследования [7, 8] показывают, что g соответствует этим требованиям — на текущий момент не найдено атак лучше универсальных. Проблема в том, что в общем случае вероятность успеха универсального метода (тотального опробования) при r связанных ключах в r раз больше, чем при их отсутствии. До определённого объёма обрабатываемых данных (нагрузки на ключ), как показано в [6], специфика хеш-функции позволяет уйти от указанной проблемы, но значительное превышение этой границы приводит к эффективным атакам [9, 5]. Известны методы определения секретного ключа для HMAC-Стрибог со сложностью порядка $2^{4n/5}$ по времени и данным [9], аналогично и для Стрибог-К [5].

Второе следствие носит позитивный характер. Финализацию нельзя осуществить без ключа K (пусть и связанного). Если в результате некоторой утечки, например по побочным каналам, противнику становится известно внутреннее состояние криптоалгоритма (состояние хеш-функции после обработки первых блоков некоторого сообщения), то из-за ключа K в последнем блоке противник не сможет непосредственно вычислить имитовставку (хеш-значение). Здесь и далее считаем, что КС не является частью состояния, а ключ K складывается с σ однократно при вычислении последней функции сжатия. При «наивной» реализации хэш-функции частью состояния является блок вида $K \boxplus m_1 \boxplus \ldots \boxplus m_j$ (сумма ключа и первых j блоков сообщения), утечка которого немедленно приводит к раскрытию ключа.

Третье следствие — для защиты от атак по побочным каналам часто используется *маскирование*. В таких условиях попеременное использование операций «$\boxplus$» (при вычислении КС) и «$\oplus$» (сложение по модулю 2 внутри самой функции сжатия) приводит к дополнительным накладным расходам, требует применения специальных алгоритмов [10, 11], что снижает скорость работы и усложняет реализацию.

В настоящей работе предлагается простой способ построения ключевого криптоалгоритма, сохраняющий положительные свойства контрольной суммы и устраняющий отрицательные. К сообщению M приписывается специальный блок C так, чтобы КС от их конкатенации $M||C$ была равна нулю. При хешировании $\mathsf{H}(K||M||C)$ ключ K будет и первым блоком (в силу явного расположения), и последним (из-за КС). В саму хеш-функцию, как видно, не вносится никаких изменений. Получившийся криптоалгоритм по своей идее схож со схемой «Сэндвич-MAC» [12], отсюда условное наименование «Стрибог-С(эндвич)». Значение C по сути является «альтернативной» КС, не зависит от ключа, может вычисляться по ходу хеширования сообщения.

Стойкость схемы «сэндвич» требует более слабых предположений о функции сжатия (PRF вместо PRF-RKA), чем требуются для Стрибог-К и для HMAC-Стрибог. Эвристические оценки преобладания противника в задаче различения «криптоалгоритм или случайная функция» (а равно, оценки вероятности навязывания [13]) у трёх упомянутых криптоалгоритмов почти одинаковы.

Отсутствие связанных ключей позволяет доказать, что даже при утечке внутреннего состояния (но без утечки дополнительных сведений о ключе[1]) единственным эффективным способом определения секретного ключа для Стрибог-С является тотальное опробование (в предположении, что ключ функции сжатия также нельзя определить эффективнее тотального опробования). Для Стрибог-К и HMAC-Стрибог аналогичное утверждение верно только при длине ключа $k \leqslant n/2 = 256$ бит. При $k \leqslant n$ и q выбранных сообщениях определение ключа в таких условиях потребует порядка $(2^k/q + 2^{n/2})$ операций [5].

При раскрытии внутреннего состояния три рассматриваемых криптоалгоритма остаются стойкими PRF и стойкими к атакам на ключ (модель KR — Key Recovery). В настоящей работе описываются соответствующие формальные модели PRF-LEAK и KR-LEAK. От функции сжатия в таких условиях дополнительно требуется стойкость к атакам на построение коллизий (модель CR — Collision Resistance) и прообраза к зафиксированному значению (модель TPR — Target Preimage Resistance). Отдельное рассмотрение моделей KR (и KR-LEAK) обусловлено возможностью получения более точных оценок.

Реализация маскирования для Стрибог-С упрощается — последним обрабатываемым блоком является ключ K, а не сумма $K \boxplus \sigma$. При этом, в силу стойкости к утечке состояния, защищать необходимо только последний вызов функции сжатия, а точнее, используемый в ней LPSX-шифр. Соответствующие способы хорошо известны и рассматриваются, например, в [14–16].

Изложение результатов построено следующим образом: в п. 1 вводятся обозначения и приводятся общие сведения о математическом аппарате теории доказуемой стойкости. Пункт 2 содержит описание хеш-функции «Стрибог» и ключевых криптоалгоритмов, построенных на её основе. В п. 3 даётся формальное описание вычислительно сложных базовых задач, к которым сводится стойкость анализируемых алгоритмов. Пункт 4 посвящён описанию схемы «сэндвич», её основным эксплуатационным и криптографическим свойствам. Пункты 5 и 6 содержат соответственно описание моделей угроз PRF-LEAK и KR-LEAK. Для рассматриваемых криптоалгоритмов приводятся доказательства стойкости и соответствующие эвристические оценки. В п. 7 с учётом результатов анализа в моделях PRF-LEAK и KR-LEAK обсуждаются общие подходы

[1] Дополнительные сведения о ключе, например из побочных каналов, могут позволить действовать эффективнее тотального опробования.

к обеспечению защиты от атак по побочным каналам. Показывается, какие сведения о процессе вычислений должны быть защищены от утечек, а какие могут быть раскрыты противнику.

1. Обозначения и общие сведения

Обозначим:

— n, k, τ — битовый размер состояния/блока, ключа и выхода соответственно ($n = 512$, $k \leqslant n$, $\tau \leqslant n$);
— V^n — множество всех n-битных строк (элементы V^n могут естественным образом интерпретироваться как целые числа и наоборот);
— $V^{<2^n}$ — множество битовых строк длины менее 2^n;
— 0^u — строка из u нулевых бит;
— $||$ — конкатенация битовых строк;
— $\mathsf{msb}_u(X)$ и $\mathsf{lsb}_u(X)$ — усечение строки $X \in V^n$ до u старших и младших бит соответственно;
— $\oplus$ — сложение по модулю 2;
— $\boxplus$ и $\boxminus$ — сложение и вычитание по модулю 2^n соответственно;
— $\mathrm{Func}(\mathbf{X}, \mathbf{Y})$ — множество всех функций, отображающих конечное множество $\mathbf{X}$ в конечное множество $\mathbf{Y}$;
— $X \xleftarrow{\mathrm{R}} \mathbf{X}$ — случайный и равновероятный выбор элемента X из множества $\mathbf{X}$;
— $\mathsf{F} : \mathbf{X} \to \mathbf{Y}$ — детерминированный алгоритм, отображение из множества входов $\mathbf{X}$ в множество выходов $\mathbf{Y}$, $\mathsf{F} \in \mathrm{Func}(\mathbf{X}, \mathbf{Y})$.

Под противником будем понимать интерактивный вероятностный алгоритм $\mathcal{A}$, взаимодействующий с другими алгоритмами (оракулами) [17]. В рамках модели угроз TM для криптоалгоритма Alg количественную характеристику успешности противника $\mathcal{A}$ обозначаем $\mathrm{Adv}^{TM}_{\mathsf{Alg}}(\mathcal{A})$. В зависимости от модели TM значение $\mathrm{Adv}^{TM}_{\mathsf{Alg}}$ может определяться как преобладание в задаче различения реальной криптосхемы от идеальной или как вероятность реализации угрозы (пример — восстановление ключа). Вероятностное пространство каждой рассматриваемой модели угроз определяется равновероятным выбором заполнения случайной ленты у вероятностного алгоритма $\mathcal{A}$, ключа и «идеальных алгоритмов», соответствующий выбор обозначается символом «$\xleftarrow{\mathrm{R}}$».

Максимум значения $\mathrm{Adv}^{TM}_{\mathsf{Alg}}(\mathcal{A})$ среди противников, возможности которых ограничены вычислительно (числом операций t в некоторой модели вычислений) и информационно (числом запросов/ответов к оракулам и другими определяемыми моделью TM и алгоритмом Alg параметрами $prms$) обозначаем $\mathrm{Adv}^{TM}_{\mathsf{Alg}}(t, prms)$.

Результатом вычислений $\mathcal{A}$ после взаимодействия с оракулом $\mathcal{O}$ является значение x, обозначаем это $\mathcal{A}^{\mathcal{O}} \Rightarrow x$. Для моделей угроз, в которых целью противника является различение «реального» алгоритма от «идеального», результат работы $\mathcal{A}$ — бит $b \in \{0, 1\}$, где «реальному» алгоритму соответствует 1 ($\mathcal{A}^{\mathcal{O}} \Rightarrow 1$), а «идеальному» — 0 ($\mathcal{A}^{\mathcal{O}} \Rightarrow 0$). Противник $\mathcal{A}$ может формировать запросы к оракулу адаптивным образом, i-й запрос к оракулу может зависеть от ответа на запросы с номерами 1, 2, …, $i-1$, $1 \leqslant i \leqslant q$ (осуществляется атака с адаптивно выбираемыми сообщениями). Содержимое запроса полностью определяется противником, ограничение задаётся только на максимальную длину запроса. Без потери общности полагаем, что противник $\mathcal{A}$ всегда использует максимально возможное число запросов и среди них нет совпадающих (нет «бессмысленных действий»). Считаем, что размер описания алгоритма $\mathcal{A}$ (его ис-

ходного кода) ограничен некоторым малым значением, что позволяет исключить из рассмотрения отдельные атаки, основанные на «бесплатных» предвычислениях [18].

Неформально называем Alg стойким в модели угроз TM (TM-стойким), если $\mathrm{Adv}_{\mathsf{Alg}}^{TM}(t, prms) \leqslant \varepsilon$, где ε не превосходит некоторого малого значения, определяемого требованиями к стойкости криптосистемы, а ресурсы t и $prms$ сопоставимы с доступными противнику на практике.

Символ «$\lessapprox$» используем с целью демонстрации практической значимости результатов, подразумевая под ним: «меньше или равно, если соответствующие эвристические предположения истинны».

Приведём здесь определения часто упоминаемых моделей угроз, остальные дадим по тексту работы.

Определение 1. Преобладанием противника $\mathcal{A}$ в модели PRF (PRF-CMA — неотличимость от случайной функции при атаке с выбранными сообщениями) для ключевой функции $\mathsf{F} : \mathbf{K} \times \mathbf{X} \to \mathbf{Y}$ назовём

$$\mathrm{Adv}_{\mathsf{F}}^{\mathrm{PRF}}(\mathcal{A}) = \Pr\Big[K \xleftarrow{\mathrm{R}} \mathbf{K} : \mathcal{A}^{\mathsf{F}_K} \Rightarrow 1\Big] - \Pr\Big[\mathsf{R} \xleftarrow{\mathrm{R}} \mathrm{Func}(\mathbf{X}, \mathbf{Y}) : \mathcal{A}^{\mathsf{R}} \Rightarrow 1\Big].$$

Противник $\mathcal{A}$ делает к оракулу q запросов. Если $\mathbf{X} = V^{<2^n}$, то в ресурсы $\mathcal{A}$ включается l — максимальная длина запроса (в n-битных блоках).

Определение 2. Характеристикой успешности противника $\mathcal{A}$ в модели KR (KR-CPA, Key Recovery — восстановление ключа в условиях атаки с выбранными открытыми текстами) для ключевого алгоритма $\mathsf{F} : \mathbf{K} \times \mathbf{X} \to \mathbf{Y}$ назовём вероятность восстановления ключа

$$\mathrm{Adv}_{\mathsf{F}}^{\mathrm{KR}}(\mathcal{A}) = \Pr\Big[K \xleftarrow{\mathrm{R}} \mathbf{K} : \mathcal{A}^{\mathsf{F}_K(\cdot)} \Rightarrow K',\, K = K'\Big].$$

Ресурсы противника определяются как в модели PRF.

Хорошо известна связь этих моделей [17]:

$$\mathrm{Adv}_{\mathsf{F}}^{\mathrm{KR}}(t, q) \leqslant \mathrm{Adv}_{\mathsf{F}}^{\mathrm{PRF}}(t', q + u) + |\mathbf{Y}|^{-u}, \quad t' = t + q + u,$$

т. е. преобладание в задаче различения является верхней оценкой вероятности восстановления ключа (здесь и далее считаем, что вычислительная сложность передачи одного блока между противником и оракулом не превосходит сложности вычисления алгоритма F).

2. Хеш-функция «Стрибог»

Приведём описание хеш-функции «Стрибог» [1], пользуясь часто применяемым эквивалентным представлением [19], подробности можно найти также в [5].

Хешируемое двоичное сообщение $M \in V^{<2^n}$ дополняется битовой строкой $10\ldots0$, чтобы длина текста была кратна n: $M' = M||10\ldots0$. Дополнение выполняется, даже если длина M кратна n.

Далее M' разбивается на $(l+1)$ блоков по $n = 512$ бит, $M' = m_0||m_1||m_2||\ldots||m_l$. С помощью функции сжатия $\mathsf{g} : V^n \times V^n \to V^n$ выполняется их последовательная обработка:

$$h_{i+1} = \mathsf{g}(h_i, m_i) \oplus \Delta_i,\ 0 \leqslant i \leqslant l-1,$$
$$h_{l+1} = \mathsf{g}(h_l, m_l) \oplus \widetilde{\Delta}_l.$$

Здесь $\Delta_i, \widetilde{\Delta}_l \in V^n$ — некоторые константы, при этом $\Delta_i \neq \widetilde{\Delta}_l$, $i = 0, \ldots, l-1$. Начальное состояние хеш-функции $h_0 = IV_\tau \in V^n$ зависит от длины выхода $\tau \in \{256, 512\}$.

После обработки блоков из M' выполняется финализация, обрабатывается блок L (битовая длина сообщения M) и блок контрольной суммы $\Sigma = \mathsf{sum}_{\boxplus}(M) = m_0 \boxplus \ldots \boxplus m_l$:

$$H = \mathsf{g}(\mathsf{g}(h_{l+1}, L), \Sigma).$$

Результатом хеширования является $\mathsf{H}_\tau(M) = \mathsf{msb}_\tau(H)$, при $\tau = 512$ усечение фактически не выполняется. Если длина выхода не играет роли, то соответствующий индекс в обозначениях опускаем.

Функция сжатия реализована с помощью 12-раундового блочного шифра LPSX-типа $\mathsf{E} : V^n \times V^n \to V^n$ с использованием конструкциии Миагучи — Пренеля:

$$\mathsf{g}(h_i, m_i) = \mathsf{E}(h_i, m_i) \oplus h_i \oplus m_i = h_{i+1}.$$

Блочный шифр E состоит из 12 полных раундов и одного усечённого (всего 13 раундовых ключей)

$$\mathsf{E}(h, m) = \mathsf{X}_{13}\,\mathsf{L}\,\mathsf{P}\,\mathsf{S}\,\mathsf{X}_{12} \ldots \mathsf{L}\,\mathsf{P}\,\mathsf{S}\,\mathsf{X}_2\mathsf{L}\,\mathsf{P}\,\mathsf{S}\,\mathsf{X}_1(m),$$

а каждый раунд содержит четыре операции:

$\mathsf{X_j}$ — сложение блока по модулю 2 с j-м раундовым ключом $h^{(j)}$, $1 \leqslant j \leqslant 13$;

S — параллельное применение зафиксированной подстановки к каждому байту;

P — перестановка байт в блоке;

L — параллельное применение линейного преобразования к 64-битным подблокам.

Раундовые ключи формируются схожим образом:

$$h^{(1)} = \mathsf{LPS}(h),\ h^{(j+1)} = \mathsf{LPS}(h^{(j)} \oplus rc^{(j)}),\ 1 \leqslant j \leqslant 12.$$

Здесь $rc^{(j)} \in V^n$ — раундовые константы.

На основе хеш-функции «Стрибог» в [4] определено ключевое преобразование

$$\mathsf{HMAC}\text{-Стрибог-}\tau(K, M) = \mathsf{H}_\tau(\overline{K} \oplus \mathtt{opad} \,||\, \mathsf{H}_\tau(\overline{K} \oplus \mathtt{ipad} \,||\, M)),$$

где ключ $\overline{K} = K \,||\, 0^{n-k}$, $k \leqslant n$; $\mathtt{opad}$ и $\mathtt{ipad}$ — различные ненулевые константы.

Особенности хеш-функции позволяют построить ключевое преобразование более простым способом [5] за счёт однократного хеширования:

$$\text{Стрибог-K}(K, M) = \mathsf{H}(\overline{K} \,||\, M).$$

Для упрощения обозначений положим $\overline{K} = m_0$ и $M' = m_1||\ldots||m_l$ и определим каскадное преобразование следующим образом:

$$\mathsf{Csc}(K_{\mathsf{Csc}}, M) = \mathsf{g}(\ldots \mathsf{g}(\mathsf{g}(K_{\mathsf{Csc}} \oplus \Delta_0, m_1) \oplus \Delta_1, m_2) \ldots \oplus \widetilde{\Delta}_l, L).$$

Здесь каскадный ключ $K_{\mathsf{Csc}} \in V^n$. Полагаем, что входные данные $M \in V^{<2^n}$ дополняются в Csc строкой $10\ldots0$, а длина L увеличивается на n из-за приписывания ключа. Ключевая хеш-функция примет следующий вид (рис. 1):

$$\mathsf{H}(\overline{K}||M) = \mathsf{g}(\mathsf{Csc}(\mathsf{g}(IV, \overline{K}), M), \overline{K} \boxplus \sigma),$$
$$\sigma = \mathsf{sum}_{\boxplus}(M) = m_1 \boxplus m_2 \boxplus \ldots \boxplus m_l.$$

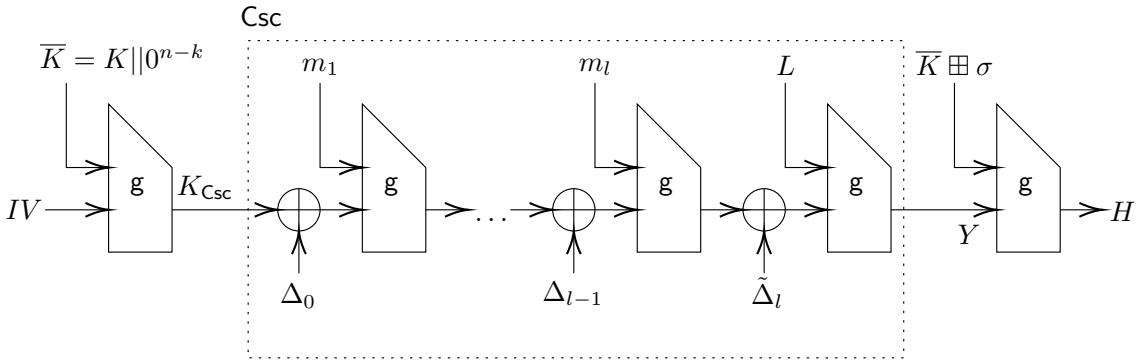

Рис. 1. Ключевая хеш-функция

Для краткости записи иногда сокращаем наименование Стрибог-K до KH, а HMAC-Стрибог — до HMAC; их областью определения считаем $V^k \times V^{<2^n} \to V^\tau$.

3. Базовые задачи

Стойкость рассматриваемых ключевых криптоалгоритмов сводится к ряду вычислительно сложных базовых задач, зависящих от функции сжатия g в соответствующих моделях угроз. Результаты конструктивного криптоанализа свидетельствуют, что g действительно является стойкой, а базовые задачи, следовательно, сложными. Приведём определения формальных моделей и дадим эвристические оценки преобладания противника, атакующего g.

Определение 3. Преобладанием противника $\mathcal{A}$ в модели PRF-RKA$_\circledast$ для ключевого криптоалгоритма $\mathsf{F} : \mathbf{K} \times \mathbf{X} \to \mathbf{Y}$ назовём

$$\mathrm{Adv}_{\mathsf{F}}^{\text{PRF-RKA}_\circledast}(\mathcal{A}) = \Pr\Big[K \xleftarrow{\mathrm{R}} \overline{\mathbf{K}} : \mathcal{A}^{\mathsf{F}_{K\circledast\cdot}(\cdot)} \Rightarrow 1\Big] -$$
$$- \Pr\Big[K \xleftarrow{\mathrm{R}} \overline{\mathbf{K}},\, \mathsf{R}_i \xleftarrow{\mathrm{R}} \mathrm{Func}(\mathbf{X}, \mathbf{Y}),\, \forall i \in \mathbf{K} : \mathcal{A}^{\mathsf{R}_{K\circledast\cdot}(\cdot)} \Rightarrow 1\Big],$$

где $\mathbf{K}, \mathbf{X}, \mathbf{Y}$ — множества ключей, входов и выходов соответственно; $\overline{\mathbf{K}} \subseteq \mathbf{K}$. Символом «$\circledast$» обозначена w-арная операция, определённая над $\mathbf{K}$ и являющаяся параметром модели. Запрос от $\mathcal{A}$ состоит из входа $x \in \mathbf{X}$ и «связи» $\kappa \in \mathbf{K}^{w-1}$. Ответом является значение $y = \mathsf{F}_{K\circledast\kappa}(x)$ (соответственно $y = \mathsf{R}_{K\circledast\kappa}(x)$). Противник $\mathcal{A}$ делает к оракулу q запросов, их содержимое ограничено по числу «связей» (r) и по максимальному числу различных «связей» (d), с которыми преобразуется одно и то же значение x $(d \leqslant r \leqslant q)$.

Определение 4. Характеристикой успешности противника $\mathcal{A}$ в модели KR-RKA$_\circledast$ для ключевого алгоритма $\mathsf{F} : \mathbf{K} \times \mathbf{X} \to \mathbf{Y}$ назовём вероятность восстановления ключа

$$\mathrm{Adv}_{\mathsf{F}}^{\text{KR-RKA}_\circledast}(\mathcal{A}) = \Pr\Big[K \xleftarrow{\mathrm{R}} \overline{\mathbf{K}} :\, \mathcal{A}^{\mathsf{F}_{K\circledast\cdot}(\cdot)} \Rightarrow K',\, K = K'\Big].$$

Параметры модели и ресурсы противника определяются как в PRF-RKA$_\circledast$.

Отметим, что при $\mathbf{K} = \overline{\mathbf{K}}$ и использовании в качестве «$\circledast$» унарного тождественного преобразования модель PRF-RKA$_\circledast$ эквивалентна модели PRF, а модель KR-RKA$_\circledast$ — модели KR. При одинаковых параметрах («$\circledast$» и $\overline{\mathbf{K}}$) модели PRF-RKA$_\circledast$ и KR-RKA$_\circledast$ соотносятся так же, как PRF и KR.

В качестве «$\circledast$» обычно используется бинарная операция $\circledast \in \{\boxplus, \oplus\}$. Анализ криптоалгоритма HMAC-Стрибог требует рассмотрения тернарной операции, определяемой композицией $\boxplus \circ \oplus$, в этом случае подразумеваем, что операндом для $\oplus$ служат

только два различных значения (`ipad` и `opad`). Множество $\overline{\mathbf{K}}$ введено исключительно из-за технического соображения — ключ может быть короче n-битного блока и в таком случае дополняется нулями.

Секретным ключом функции сжатия может выступать как блок сообщения, так и блок состояния, обозначаем это соответственно

$$\mathsf{g}^{\triangledown}_{\overline{K}}(\cdot) = \mathsf{g}(\cdot, \overline{K}),\ k \leqslant n,\ \overline{K} \in \overline{\mathbf{K}} = \{K \,||\, 0^{n-k} :\ K \in V^k\},$$
$$\mathsf{g}^{\triangleright}_{h}(\cdot) = \mathsf{g}(h, \cdot),\ h \in \overline{\mathbf{K}} = \mathbf{K} = V^n.$$

С учётом конструктивных результатов анализа [7, 8, 20] в работе [6] для $\mathsf{g}^{\triangledown}$ и $\mathsf{g}^{\triangleright}$ приводятся следующие эвристические оценки (по вероятности успеха универсальных атак):

$$\mathrm{Adv}^{\text{PRF-RKA}_{\boxplus}}_{\mathsf{g}^{\triangledown}}(t, q, r, d) \lessapprox \frac{t \cdot d}{2^k} \leqslant \frac{t \cdot r}{2^k} \leqslant \frac{t \cdot q}{2^k}; \tag{1}$$

$$\mathrm{Adv}^{\text{PRF-RKA}_{\oplus}}_{\mathsf{g}^{\triangleright}}(t, q, r = 2) \lessapprox \frac{2t}{2^n} + \frac{q(q-1)}{2^{n+1}}. \tag{2}$$

Оценку (1) используем в таком же виде для модели KR-RKA$_{\boxplus}$. Оценку для $\mathsf{g}^{\triangleright}$ в аналогичном случае можно несколько уточнить, так как второе слагаемое в (2) соответствует различителю по парадоксу дней рождения, а не атаке на ключ. Одной операцией (из t возможных) здесь и далее считается вычисление функции сжатия.

Предположение о стойкости $\mathsf{g}^{\triangleright}$ в модели PRF-RKA$_{\oplus}$ нужно для оценки каскадного преобразования [5]:

$$\mathrm{Adv}^{\text{PRF}}_{\mathsf{Csc}}(t, q, l) \leqslant q \cdot l' \cdot \mathrm{Adv}^{\text{PRF-RKA}_{\oplus}}_{\mathsf{g}^{\triangleright}}(t', q, r = 2), \quad t' = t + ql, \quad l' = l + 1.$$

В п. 5 показано, что анализируемые ключевые хеш-функции остаются PRF-стойкими даже при реализации угроз, связанных с утечкой внутреннего состояния криптоалгоритма (модель PRF-LEAK). От функции сжатия в таких условиях необходимо потребовать стойкости к атакам CR и TPR.

Определение 5. Характеристикой успешности противника $\mathcal{A}$ в модели CR для хеш-функции $\mathsf{F} : \mathbf{S} \times \mathbf{X} \to \mathbf{Y}$ назовём вероятность построения коллизии

$$\mathrm{Adv}^{\text{CR}}_{\mathsf{F}}(\mathcal{A}) = \Pr\Big[S \xleftarrow{\text{R}} \mathbf{S} :\ \mathcal{A}(S) \Rightarrow (M, M'),\ \mathsf{F}(S, M) = \mathsf{F}(S, M')\ \&\ M \neq M'\Big].$$

Определение 6. Характеристикой успешности противника $\mathcal{A}$ в модели TPR для хеш-функции $\mathsf{F} : \mathbf{S} \times \mathbf{X} \to \mathbf{Y}$ назовём вероятность построения прообраза

$$\mathrm{Adv}^{\text{TPR}}_{\mathsf{F}}(\mathcal{A}) = \Pr\Big[S \xleftarrow{\text{R}} \mathbf{S} :\ \mathcal{A}(S) \Rightarrow M,\ \mathsf{F}(S, M) \in \mathbf{T}\Big];$$

множество целевых образов $\mathbf{T}$ является параметром модели.

Причина введения множества $\mathbf{S}$ носит формальный характер [21]. Необходимо вывести из рассмотрения алгоритмы, в которых «зашита» найденная заранее пара сообщений (M, M'), порождающих коллизию. Такому алгоритму достаточно лишь предъявить эту пару. Указанный алгоритм, разумеется, существует, обладает малым описанием и требует пренебрежимо мало вычислительных ресурсов, но если смотреть с практической точки зрения, требует огромного объёма предварительно выполняемых операций для своего явного построения. Введение случайного выбора S делает подобные предвычисления бессмысленными. При этом реальные алгоритмы построения

коллизий обычно не используют существенным образом конкретное значение S и эффективны при любом выборе. Аналогичные соображения верны для модели TPR.

Для g множеством входов является $\mathbf{X} = (V^n \times V^n)$ — множество пар (блок состояния, блок сообщения). Множеством $\mathbf{S}$ можно условно считать все значения, которые могут принимать раундовые константы $rc^{(1)}, \ldots, rc^{(12)}$, $\mathsf{g} : \mathbf{S} \times (V^n \times V^n) \to V^n$.

С учётом отсутствия специфических методов построения коллизий в открытой литературе [22–26] (атакуется не более 9,5 из 12 полных раундов [25]), эвристическая оценка равна

$$\mathrm{Adv}^{\mathrm{CR}}_{\mathsf{g}}(t) \lessapprox \frac{t^2}{2^{n+1}} \tag{3}$$

— по вероятности успеха универсального метода — ρ-метода Полларда [27] (по парадоксу дней рождения).

Для полнораундовой функции сжатия не представлены нетривиальные методы построения прообраза [23, 24, 28–30] (атакуется не более 8,5 раундов [30]), поэтому эвристическая оценка вероятности успеха

$$\mathrm{Adv}^{\mathrm{TPR}}_{\mathsf{g}}(t) \lessapprox \frac{t \cdot |\mathbf{T}|}{2^n} = \frac{t}{2^n} \tag{4}$$

даётся по методу полного перебора, в качестве $\mathbf{T}$ используется только одноэлементное множество целей (образов), $\mathbf{T} = \{IV\}$.

Для каскадного преобразования, которое здесь для формальности определяется как $\mathsf{Csc} : \mathbf{S} \times (V^n \times V^{<2^n}) \to V^n$, верны следующие неравенства:

$$\mathrm{Adv}^{\mathrm{CR}}_{\mathsf{Csc}}(t) \leqslant \mathrm{Adv}^{\mathrm{CR}}_{\mathsf{g}}(t'); \tag{5}$$

$$\mathrm{Adv}^{\mathrm{TPR}}_{\mathsf{Csc}}(t) \leqslant \mathrm{Adv}^{\mathrm{TPR}}_{\mathsf{g}}(t'), \quad t' = t + 1. \tag{6}$$

Первое является классическим результатом [2, 3]. Последний хешируемый блок в Csc содержит длину сообщения L (МД-усиление), а значит, коллизия каскада — это коллизия и для функции сжатия. Требование к g может быть ослаблено с CR-стойкости до стойкости к коллизиям специального вида «Constrained CR» [31], но для упрощения изложения ограничимся оценкой (5).

Неравенство (6) для модели TPR следует из простого наблюдения. Пусть тройка (S, K_{Csc}, M) такова, что $\mathsf{Csc}(S, (K_{\mathsf{Csc}}, M)) \in \mathbf{T}$, тогда $\mathsf{g}(S, (x, L)) \in \mathbf{T}$, где x — состояние перед последним вызовом функции сжатия, которое легко вычислить, зная вход Csc.

Везде далее подразумеваем случайный выбор S и множество $\mathbf{S}$, опуская их явное обозначение.

4. Схема «сэндвич»

Стрибог-C (обозначаем для краткости SH) дополняет хешируемый текст M так, чтобы в последнем хешируемом блоке был именно ключ $\overline{K} = K \,||\, 0^{n-k}$, а не сумма $\overline{K} \boxplus \sigma$. Таким образом, ключ $\overline{K}$ является и первым хешируемым блоком, и последним — отсюда наименование «сэндвич». Напомним, что у хэш-функции «Стрибог» двумя последними хэшируемыми блоками являются битовая длина и контрольная сумма хэшируемых данных (рис. 2).

Различные варианты схемы «сэндвич» были предложены в [12] для «простой» конструкции Меркла — Дамгарда (т. е. без контрольной суммы). Сами схемы анализировались только в модели PRF, доказательства опирались на результат [32], практическая

неприменимость которого показана в [33]. Отметим также, что в предложенных вариантах схемы [12] первый и последний хешируемые блоки содержали ключ, но были заведомо различны.

Для хеш-функции «Стрибог» схема «сэндвич» реализуется за счёт специального блока $C \in V^n$:

$$\mathsf{SH}(K, M) = \mathsf{H}(\overline{K} \,||\, M \,||\, C), \quad C = \mathsf{cs}(M).$$

Алгоритм $\mathsf{cs} : V^{<2^n} \to V^n$ не зависит от секретного ключа и может выполняться по ходу обработки сообщения M.

Определим необходимую «поправку» к контрольной сумме

$$\tilde{C} \boxplus \mathsf{sum}_{\boxplus}(M) = 0, \quad \tilde{C} = 0 \boxminus \mathsf{sum}_{\boxplus}(M) = 0 \boxminus m_1 \boxminus m_2 \boxminus \ldots \boxminus m_l,$$

при расчёте $\mathsf{sum}_{\boxplus}$ текст M дополняется p-битной строкой $10\ldots0$ $(1 \leqslant p \leqslant n)$.

Если $p = n$, то C располагается в одном n-битном блоке, $C = \tilde{C}$, а если $p < n$, то биты $\tilde{C}$ попадают в два блока — старшие (msb) в последний, а младшие (lsb) — в предпоследний (рис. 2):

$$C = C_{lsb} \,||\, C_{msb} = \mathsf{lsb}_p(\tilde{C}) || \mathsf{msb}_{n-p}(\tilde{C}),$$

случай $p = n$ также описывается этой формулой: $C = \mathsf{lsb}_n(\tilde{C}) = \tilde{C} = C_{msb} \,||\, C_{lsb}$.

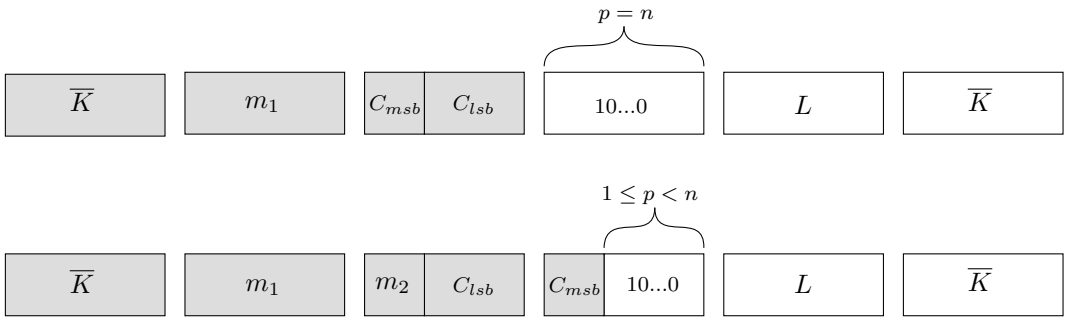

Рис. 2. Блоки на входе функции сжатия для случаев $p = n$ и $1 \leqslant p < n$. Серым выделены биты, которые являются входом хеш-функции, белым — биты, формируемые хеш-функцией. L – битовая длина входа хэш-функции

Формирование блока C не сказывается существенно на производительности криптоалгоритма. Вычисление КС в любом случае выполняется в рамках хеш-функции, Стрибог-С лишь заменяет сложение по модулю 2^n на вычитание — $\boxplus$ на $\boxminus$. Значение $\tilde{C}$ можно вычислить без непосредственного применения операции $\boxminus$ — за счёт побитовой инверсии (обозначаем «$\sim$») и сложения с единицей: $\tilde{C} = \sim \mathsf{sum}_{\boxplus}(M) \boxplus 1$.

Таким образом, на вычислительную эффективность существенно влияет лишь необходимость одного дополнительного обращения к функции сжатия. В таблице сравнивается число обращений к g для случая, когда длина сообщения M меньше n бит. Под предвычислениями понимается возможность заранее вычислить и хранить каскадный ключ $K_{\mathsf{Csc}} = \mathsf{g}(IV, \overline{K})$ (для HMAC-Стрибог соответственно два каскадных ключа, $K^I_{\mathsf{Csc}} = \mathsf{g}(IV, \overline{K} \oplus \mathtt{ipad})$ и $K^O_{\mathsf{Csc}} = \mathsf{g}(IV, \overline{K} \oplus \mathtt{opad})$).

Как можно видеть, «сэндвич» порождает меньше накладных расходов, чем двойное хеширование, но требует на один вызов g больше, чем Стрибог-К. Референсные реализации перечисленных в таблице криптоалгоритмов представлены в репозитории [34].

Минимальное число вызовов g, необходимое для обработки сообщения

Предвычисления	Стрибог-K	Стрибог-C	HMAC-Стрибог-256	HMAC-Стрибог-512
Нет	4	5	8	9
Есть	3	4	6	7

Необходимость использования «альтернативной» КС вызвана желанием оставить саму хеш-функцию «Стрибог» без каких-либо изменений, что, как представляется, позволяет использовать существующие реализации алгоритмов и в целом упрощает процесс внедрения. Безусловно, ключевую хеш-функцию по схеме «сэндвич» можно построить «с нуля» так, чтобы рассматриваемые криптографические свойства оставались такими же, а минимальное число обращений к g составляло 3 (ключ — блок с дополнением — ключ) и 2 (с предвычислениями).

Отметим, что аналогичный способ преобразования в схему «сэндвич» можно применить к хеш-функции ГОСТ Р 34.11-94 [35].

В [5, 6] показано, что Стрибог-K и HMAC-Стрибог являются PRF-стойкими, в частности, доказана

Теорема 1 [6]. Преобладание любого противника, атакующего Стрибог-K в модели PRF, ограничено следующим образом:

$$\mathrm{Adv}^{\mathrm{PRF}}_{\mathsf{KH}}(t,q,l) \leqslant \mathrm{Adv}^{\text{PRF-RKA}_{\boxplus}}_{\mathsf{g}^{\triangledown}}(t',q',r,d) + \mathrm{Adv}^{PRF}_{\mathsf{Csc}}(t',q,l') + \frac{q^2+q}{2^{n+1}}. \tag{7}$$

Здесь $t' = t + ql$; $r = q' = q+1$; $l' = l+1$; $d = 1$.

Эвристическая оценка преобладания противника

$$\mathrm{Adv}^{\mathrm{PRF}}_{\mathsf{KH}}(t',q,l) \lessapprox \frac{t'}{2^k} + \frac{2\cdot t'\cdot q\cdot l'}{2^n} + \frac{q^2+q}{2^n}, \quad t' \approx t, \quad l' = l+1, \tag{8}$$

получена в [6] в предположении об отсутствии для функции сжатия атак, которые были бы лучше универсальных. Оценка (8) является точной — каждому слагаемому оценки соответствует атака, действующая с сопоставимой вероятностью успеха: тотальное опробование ключа; атака за счёт деградации каскада; атака по парадоксу дней рождения.

PRF-стойкость схемы «сэндвич» является прямым следствием теоремы 1. В самом деле, из равенства

$$\mathsf{SH}(K,M) = \mathsf{KH}(K, M \,||\, \mathsf{cs}(M)) = \mathsf{H}(\overline{K} \,||\, M \,||\, C)$$

следует, что любой запрос к SH может быть выражен через запрос к KH. Пусть существует алгоритм $\mathcal{A}$, эффективно атакующий SH в модели PRF. Построим алгоритм $\mathcal{B}$, который будет атаковать KH с такой же эффективностью. На любой запрос M от $\mathcal{A}$, алгоритм $\mathcal{B}$ вычисляет $C = \mathsf{cs}(M)$, передаёт своему оракулу $\mathcal{O} \in \{\mathsf{KH}, \mathsf{R}\}$ запрос $M||C$, ответ оракула возвращает алгоритму $\mathcal{A}$. Результат работы алгоритма $\mathcal{B}$ равен результату работы $\mathcal{A}$. Алгоритм $\mathcal{B}$ идеально симулирует для $\mathcal{A}$ оракул SH или R, следовательно, в силу произвольности алгоритма $\mathcal{A}$ верно неравенство

$$\mathrm{Adv}^{\mathrm{PRF}}_{\mathsf{SH}}(t,q,l) \leqslant \mathrm{Adv}^{\mathrm{PRF}}_{\mathsf{KH}}(t',q,l+1), \quad t' = t + ql.$$

Кроме этого, у SH все блоки на входе функции $\mathsf{g}^{\triangledown}$ обрабатываются при ключе $\overline{K}$, а не при различных связанных ключах. Значение r снижается с $q+1$ до 1 — вместо модели $\text{PRF-RKA}_{\boxplus}$ фактически используется модель PRF.

Следствие 1. Преобладание любого противника, атакующего Стрибог-С в модели PRF, ограничено:

$$\mathrm{Adv}^{\mathrm{PRF}}_{\mathsf{SH}}(t,q,l) \leqslant \mathrm{Adv}^{\mathrm{PRF}}_{\mathsf{g}^{\triangledown}}(t',q') + \mathrm{Adv}^{\mathrm{PRF}}_{\mathsf{Csc}}(t',q,l') + \frac{q^2+q}{2^{n+1}}. \tag{9}$$

Здесь $t' = t + ql$; $q' = q + 1$; $l' = l + 2$.

Оценка (8) является точной и для схемы «сэндвич», для которой могут быть применены три соответствующие атаки.

5. Стойкость при раскрытии состояния

Рассматриваемые ключевые криптоалгоритмы (SH, KH, HMAC) обладают важным и полезным на практике свойством — они остаются стойкими псевдослучайными функциями, даже если противник получает доступ к внутреннему состоянию хеш-функции. Сам ключ, разумеется, не считается частью состояния. Контрольная сумма не зависит от ключа и сама по себе известна противнику (также не является частью состояния). Сложение КС с ключом происходит в KH однократно при вычислении последней функции сжатия (в HMAC соответственно двукратно), а в SH не происходит вообще.

Определение 7. Преобладанием противника $\mathcal{A}$ в модели PRF-LEAK для ключевого криптоалгоритма $\mathsf{F} : \mathbf{K} \times \mathbf{X} \to \mathbf{Y}$ при утечке, определяемой функциями $\mathsf{leak}_1 : \mathbf{K} \to \mathbf{L}_1$ и $\mathsf{leak}_2 : \mathbf{K} \times \mathbf{X} \to \mathbf{L}_2$, назовём

$$\mathrm{Adv}^{\text{PRF-LEAK}}_{\mathsf{F},\mathsf{leak}_1,\mathsf{leak}_2}(\mathcal{A}) = \Pr\Big[K \xleftarrow{\mathrm{R}} \mathbf{K} : \mathcal{A}^{\mathcal{L}}(\mathsf{leak}_1(K)) \Rightarrow 1\Big] -$$
$$- \Pr\Big[\mathsf{R} \xleftarrow{\mathrm{R}} \mathrm{Func}(\mathbf{X}, \mathbf{Y} \times \mathbf{L}_2),\, lk \xleftarrow{\mathrm{R}} \mathbf{L}_1 : \mathcal{A}^{\mathsf{R}}(lk) \Rightarrow 1\Big].$$

Оракул $\mathcal{L}$ на запрос $X \in \mathbf{X}$ возвращает ответ $(\mathsf{F}(K,X), \mathsf{leak}_2(K,X))$, состоящий из результата работы криптоалгоритма F и утечки, зависящей от ключа K и запроса X.

Нетрудно видеть, что при любом виде утечки модель PRF-LEAK расширяет модель PRF. При $\mathbf{L}_1 = \mathbf{L}_2 = \varnothing$ модели PRF и PRF-LEAK эквивалентны.

Функции leak_1 и leak_2 являются детерминированными, хотя в общем случае вместо них можно использовать вероятностные алгоритмы. В пользу противника считаем тем самым, что он получает достоверную «незашумлённую» информацию.

Согласно модели, раскрываемые промежуточные состояния должны быть вычислительно неотличимы от случайных. Это достаточно строгое требование, которому тем не менее соответствуют анализируемые криптоалгоритмы.

5.1. Стрибог-С и Стрибог-К

Для Стрибог-С и Стрибог-К в качестве leak_1 выберем $\mathsf{g}^{\triangledown}_{K}(IV)$ — до начала взаимодействия с оракулом раскрывается каскадный ключ K_{Csc}. Противник, таким образом, может вычислить любое состояние хеш-функции, исключение составляет последний вызов функции сжатия. Значение $H = \mathsf{g}^{\triangledown}_{\overline{K} \boxplus \sigma}(Y)$ даже при известных $Y = \mathsf{Csc}(K_{\mathsf{Csc}}, M)$ и $\sigma = \mathsf{sum}_{\boxplus}(M)$ ($\sigma = 0$ для SH) вычислить без знания $\overline{K}$ нельзя. Функция leak_2 тривиальна, $\mathbf{L}_2 = \varnothing$, так как знание K_{Csc} уже даёт противнику дополнительные возможности, релевантные для нашего анализа.

Покажем, что информация о промежуточных состояниях, полученная противником, не увеличивает существенным образом его преобладание в задаче различения, а следовательно, вероятность навязывания подделки и вероятность восстановления ключа остаются малыми.

Теорема 2. Преобладание любого противника, атакующего Стрибог-K или Стрибог-C в модели PRF-LEAK, при утечке каскадного ключа $K_{\mathsf{Csc}} = \mathsf{g}^{\triangledown}_{\overline{K}}(IV)$ ограничено соответственно

$$\begin{gathered}\mathrm{Adv}^{\text{PRF-LEAK}}_{\mathsf{KH},K_{\mathsf{Csc}}}(t,q,l) \leqslant \mathrm{Adv}^{\text{PRF-RKA}_{\boxplus}}_{\mathsf{g}^{\triangledown}}(t',q',r,d) + \mathrm{Adv}^{\mathrm{CR}}_{\mathsf{g}}(t') + \mathrm{Adv}^{\mathrm{TPR}}_{\mathsf{g}}(t'), \qquad (10)\\ \mathrm{Adv}^{\text{PRF-LEAK}}_{\mathsf{SH},K_{\mathsf{Csc}}}(t,q,l) \leqslant \mathrm{Adv}^{\mathrm{PRF}}_{\mathsf{g}^{\triangledown}}(t',q') + \mathrm{Adv}^{\mathrm{CR}}_{\mathsf{g}}(t') + \mathrm{Adv}^{\mathrm{TPR}}_{\mathsf{g}}(t'),\end{gathered}$$

где $t' = t + ql$; $r = q' = q + 1$; $d = 1$.

Доказательство. Докажем утверждение для KH, делая для SH оговорки по ходу изложения.

Назовём коллизией (C) совпадение любой пары элементов в ряду

$$IV, Y_1, Y_2, \ldots, Y_q, \text{ где } Y_i = \mathsf{Csc}(K_{\mathsf{Csc}}, M_i),\ 1 \leqslant i \leqslant q.$$

Противоположное событие обозначаем символом «$\overline{\mathrm{C}}$» . Событие C фактически означает, что противник смог создать прообраз к IV или непосредственно коллизию $Y_i = Y_j$, $i \neq j$. Коллизию, возникающую в результате взаимодействия противника $\mathcal{A}$ с KH, обозначаем $\mathcal{A}^{\mathsf{KH}} \Rightarrow (b, \mathrm{C})$. Значение бита $b \in \{0,1\}$ является результатом, возвращаемым противником, а коллизия — неявным «побочным эффектом» его вычислений и взаимодействий. Если символ b не указан в обозначениях, то подразумевается, что значение бита может быть любым, т. е. имеет место равенство

$$\Pr\bigl[\mathcal{A}^{\mathsf{KH}} \Rightarrow \mathrm{C}\bigr] = \Pr\bigl[\mathcal{A}^{\mathsf{KH}} \Rightarrow (1,\mathrm{C})\bigr] + \Pr\bigl[\mathcal{A}^{\mathsf{KH}} \Rightarrow (0,\mathrm{C})\bigr].$$

Представленные далее рассуждения повторяют ход доказательства теоремы о PRF-стойкости KH [6]. Главное отличие заключается в том, что при секретном каскадном ключе вероятность коллизии ограничивается за счёт PRF-стойкости каскада, а при утечке каскадного ключа — за счёт его стойкости в моделях CR и TPR.

Рассмотрим преобразование

$$\widetilde{\mathsf{KH}}(M_i) = \mathsf{f}_{\overline{K}\boxplus\sigma_i}(\mathsf{Csc}(\widetilde{K}_{\mathsf{Csc}}, M_i)),\ \widetilde{K}_{\mathsf{Csc}} = \mathsf{f}_{\overline{K}}(IV),\ \sigma_i = \mathsf{sum}_{\boxplus}(M_i),$$

полученное заменой в KH первого и последнего вызовов функции сжатия $\mathsf{g}^{\triangledown}$ на семейство из 2^n случайных функций f (для SH — на случайную функцию f). Если коллизии не происходит, то алгоритм $\widetilde{\mathsf{KH}}$ неотличим от случайной функции R, т. е.

$$\Pr\bigl[\mathcal{A}^{\mathsf{R}}(lk) \Rightarrow 1\bigr] = \Pr\Bigl[\mathcal{A}^{\widetilde{\mathsf{KH}}}(\widetilde{K}_{\mathsf{Csc}}) \Rightarrow (1,\overline{\mathrm{C}})\Bigr],$$

благодаря тому, что независимо от σ_i запрашиваемые у f значения $IV, Y_1, \ldots, Y_q$ не повторяются, а следовательно, результаты запросов к f (они же — результаты запросов $\mathcal{A}$ к $\widetilde{\mathsf{KH}}$) есть последовательность случайных значений. Значение lk случайно и равновероятно выбирается из V^n.

Согласно определению модели PRF-LEAK,

$$\mathrm{Adv}^{\text{PRF-LEAK}}_{\mathsf{KH},K_{\mathsf{Csc}}}(\mathcal{A}) = \Pr\bigl[\mathcal{A}^{\mathsf{KH}}(K_{\mathsf{Csc}}) \Rightarrow 1\bigr] - \Pr\bigl[\mathcal{A}^{\mathsf{R}}(lk) \Rightarrow 1\bigr].$$

По формуле полной вероятности

$$\Pr\bigl[\mathcal{A}^{\mathsf{KH}}(K_{\mathsf{Csc}}) \Rightarrow 1\bigr] = \Pr\bigl[\mathcal{A}^{\mathsf{KH}}(K_{\mathsf{Csc}}) \Rightarrow (1,\mathrm{C})\bigr] + \Pr\bigl[\mathcal{A}^{\mathsf{KH}}(K_{\mathsf{Csc}}) \Rightarrow (1,\overline{\mathrm{C}})\bigr].$$

Группируя слагаемые и используя неравенство треугольника, получим

$$\mathrm{Adv}^{\text{PRF-LEAK}}_{\mathsf{KH},K_{\mathsf{Csc}}}(\mathcal{A}) \leqslant \Big(\Pr\big[\mathcal{A}^{\mathsf{KH}}(K_{\mathsf{Csc}}) \Rightarrow (1,\overline{\mathrm{C}})\big] - \Pr\Big[\mathcal{A}^{\widetilde{\mathsf{KH}}}(\widetilde{K}_{\mathsf{Csc}}) \Rightarrow (1,\overline{\mathrm{C}})\Big]\Big) +$$
$$+ \Big(\Pr\big[\mathcal{A}^{\mathsf{KH}}(K_{\mathsf{Csc}}) \Rightarrow \mathrm{C}\big] - \Pr\Big[\mathcal{A}^{\widetilde{\mathsf{KH}}}(\widetilde{K}_{\mathsf{Csc}}) \Rightarrow \mathrm{C}\Big]\Big) + \Pr\Big[\mathcal{A}^{\widetilde{\mathsf{KH}}}(\widetilde{K}_{\mathsf{Csc}}) \Rightarrow \mathrm{C}\Big] = \epsilon + \epsilon_{\mathrm{C}} + p_{\mathrm{C}}.$$

Построим алгоритм $\mathcal{B}$, атакующий $\mathsf{g}^{\triangledown}$ в модели PRF-RKA$_{\boxplus}$ (для SH — в модели PRF), преобладание которого равно сумме ϵ и ϵ_{C} (полагаем, что каждое из этих значений неотрицательно). Алгоритм $\mathcal{B}$ делает запрос IV к оракулу $\mathcal{O} \in \{\mathsf{g}^{\triangledown}, \mathsf{f}\}$, получает каскадный ключ K_{Csc}, передаёт его алгоритму $\mathcal{A}$ (симулирует leak_1). При обработке запроса M_i от $\mathcal{A}$ алгоритм $\mathcal{B}$:

— самостоятельно вычисляет значения $\sigma_i = \mathsf{sum}_{\boxplus}(M_i)$ и $Y_i = \mathsf{Csc}(K_{\mathsf{Csc}}, M_i)$;
— проверяет условие коллизии: если $Y_i \in \{IV, Y_1, \ldots, Y_{i-1}\}$, то завершает работу $\mathcal{A}$ и возвращает 1;
— делает запрос (Y_i, σ_i) к оракулу $\mathcal{O} \in \{\mathsf{g}^{\triangledown}, \mathsf{f}\}$ и возвращает его ответ $\mathcal{A}$.

Если после q запросов от $\mathcal{A}$ коллизии не произошло, то результатом работы $\mathcal{B}$ является результат работы $\mathcal{A}$.

Для SH: $Y_i = \mathsf{Csc}(K_{\mathsf{Csc}}, M_i \,||\, \mathsf{cs}(M_i))$, а запрос к $\mathcal{O} \in \{\mathsf{g}^{\triangledown}, \mathsf{f}\}$ содержит только Y_i.

К оракулу выполняется не более $q' \leqslant q+1$ запросов, совокупное число связанных ключей такое же: $r' \leqslant q+1$. До возникновения коллизии первый аргумент в запросах не повторяется ($d = 1$), так как значения в ряду $IV, Y_1, \ldots, Y_i$ различны. Кроме того, если $\mathcal{O} = \mathsf{g}^{\triangledown}$, то $\mathcal{B}$ идеально симулирует для $\mathcal{A}$ оракул KH и утечку каскадного ключа $K_{\mathsf{Csc}} = \mathsf{g}^{\triangledown}_{\overline{K}}(IV)$. Аналогично, если $\mathcal{O} = \mathsf{f}$, то для $\mathcal{A}$ идеально симулируется оракул $\widetilde{\mathsf{KH}}$ и утечка $\widetilde{K}_{\mathsf{Csc}} = \mathsf{f}_{\overline{K}}(IV)$. Преобладание алгоритма $\mathcal{B}$ равно

$$\mathrm{Adv}^{\text{PRF-RKA}_{\boxplus}}_{\mathsf{g}^{\triangledown}}(\mathcal{B}) = \Pr\Big[\mathcal{B}^{\mathsf{g}^{\triangledown}} \Rightarrow 1\Big] - \Pr\big[\mathcal{B}^{\mathsf{f}} \Rightarrow 1\big] =$$
$$= \big(\Pr\big[\mathcal{A}^{\mathsf{KH}}(K_{\mathsf{Csc}}) \Rightarrow (1,\overline{\mathrm{C}})\big] + \Pr\big[\mathcal{A}^{\mathsf{KH}}(K_{\mathsf{Csc}}) \Rightarrow \mathrm{C}\big]\big) -$$
$$- \Big(\Pr\Big[\mathcal{A}^{\widetilde{\mathsf{KH}}}(\widetilde{K}_{\mathsf{Csc}}) \Rightarrow (1,\overline{\mathrm{C}})\Big] + \Pr\Big[\mathcal{A}^{\widetilde{\mathsf{KH}}}(\widetilde{K}_{\mathsf{Csc}}) \Rightarrow \mathrm{C}\Big]\Big) = \epsilon + \epsilon_{\mathrm{C}}.$$

Для завершения доказательства осталось оценить значение вероятности

$$p_{\mathrm{C}} = \Pr\Big[\mathcal{A}^{\widetilde{\mathsf{KH}}}(\widetilde{K}_{\mathsf{Csc}}) \Rightarrow \mathrm{C}\Big] \leqslant \Pr[Y_i = Y_j,\ 1 \leqslant i < j \leqslant q] + \Pr[IV \in \{Y_1, \ldots, Y_q\}].$$

Построим алгоритм $\mathcal{B}_{\mathrm{CR}}$, пытающийся сформировать коллизию для Csc в модели CR. Он имитирует семейство случайных функций f (для SH — случайную функцию f), формирует случайное значение $\widetilde{K}_{\mathsf{Csc}} = \mathsf{f}_{\overline{K}\boxplus 0}(IV)$, передаёт $\widetilde{K}_{\mathsf{Csc}}$ алгоритму $\mathcal{A}$. На любой запрос M_i от $\mathcal{A}$ алгоритм $\mathcal{B}_{\mathrm{CR}}$: вычисляет $Y_i = \mathsf{Csc}(\widetilde{K}_{\mathsf{Csc}}, M_i)$ и $\sigma_i = \mathsf{sum}_{\boxplus}(M_i)$; сохраняет (Y_i, M_i) в памяти; возвращает $H_i = \mathsf{f}_{\overline{K}\boxplus\sigma_i}(M_i)$ алгоритму $\mathcal{A}$ (для SH: $\widetilde{K}_{\mathsf{Csc}} = \mathsf{f}(IV)$; $Y_i = \mathsf{Csc}(\widetilde{K}_{\mathsf{Csc}}, M_i||\mathsf{cs}(M_i))$; $H_i = \mathsf{f}(M_i)$).

Если на каком-то шаге обнаружено равенство $Y_i = Y_j$, $1 \leqslant i < j \leqslant q$, то построена коллизия $((\widetilde{K}_{\mathsf{Csc}}, M_i), (\widetilde{K}_{\mathsf{Csc}}, M_j))$ и

$$\Pr[Y_i = Y_j,\ 1 \leqslant i < j \leqslant q] = \mathrm{Adv}^{\mathrm{CR}}_{\mathsf{Csc}}(\mathcal{B}_{\mathrm{CR}}).$$

Схожим образом построим алгоритм $\mathcal{B}_{\mathrm{TPR}}$, который вычисляет для Csc прообраз к значению IV:

$$\Pr[IV \in \{Y_1, \ldots, Y_q\}] = \mathrm{Adv}^{\mathrm{TPR}}_{\mathsf{Csc}}(\mathcal{B}_{\mathrm{TPR}}).$$

Вычислительные ресурсы алгоритмов $\mathcal{B}$, $\mathcal{B}_{\mathrm{CR}}$ и $\mathcal{B}_{\mathrm{TPR}}$ превосходят ресурсы t алгоритма $\mathcal{A}$ на значение, пропорциональное объёму обрабатываемых данных: $t' = t + ql$. Преобладание $\mathcal{B}_{\mathrm{CR}}$ и $\mathcal{B}_{\mathrm{TPR}}$ в соответствующих моделях оценивается неравенствами (5) и (6) соответственно.

В силу произвольности алгоритма $\mathcal{A}$ и неравенства треугольника доказываемое утверждение верно. ■

Подстановкой в (10) эвристических оценок (1), (3) и (4) получаем

$$\mathrm{Adv}^{\text{PRF-LEAK}}_{\mathsf{KH},K_{\mathsf{Csc}}}(t,q,l) \lessapprox \frac{t'}{2^k} + \frac{t'^2}{2^{n+1}} + \frac{t'}{2^n}, \quad t' \approx t, \tag{11}$$

и для SH идентично. Как и для оценки (8) в модели PRF, при длине ключа, не превосходящей половины блока ($k \leqslant n/2 = 256$ бит), единственным эффективным методом нарушения свойств безопасности является тотальное опробование, которому соответствует первое слагаемое в (11).

Второе слагаемое в (11) соответствует простой атаке. Противник, знающий K_{Csc}, строит самостоятельно коллизию из сообщений вида $M = P\,||(\sigma \boxminus P)$, $M' = P'\,||(\sigma \boxminus P')$:

$$\mathsf{Csc}(K_{\mathsf{Csc}}, M) = \mathsf{Csc}(K_{\mathsf{Csc}}, M'),$$
$$\mathsf{sum}_{\boxplus}(M) = \mathsf{sum}_{\boxplus}(M') = \sigma = \mathrm{const},$$

отправляет запрос M к оракулу, получает H, предъявляет пару (M', H) в качестве подделки. Альтернативно, делает два запроса M и M' и пользуется равенством $\mathcal{O}(M) \stackrel{?}{=} \mathcal{O}(M')$ в качестве критерия различения.

Схожим образом выполняется атака за счёт построения прообраза. Противник подбирает такое сообщение $M = P\,||(0 \boxminus P)$, чтобы $\mathsf{Csc}(K_{\mathsf{Csc}}, M) = IV$ и $\mathsf{sum}_{\boxplus}(M) = 0$. Хеш-значение для такого сообщения M равно K_{Csc}, что можно использовать для формирования подделки или как критерий для различения.

Следует заметить, что, в отличие от (8), оценка (11) зависит только от вычислительных ресурсов противника и, следовательно, в общем случае её гарантии значительно слабее. Например, при длине ключа $k = 512$ и ресурсах противника $t = 2^{256}$, $q = 2^{128}$, $l = 2^{64}$ криптоалгоритмы KH и SH будут стойкими PRF, но не будут таковыми при утечке каскадного ключа.

5.2. HMAC-Стрибог

Двойное хеширование мотивирует расширить возможности противника, ему даются оба каскадных ключа, а также раскрываются промежуточные состояния после первого хеширования:

$$\mathsf{leak}_1(K) = (\mathsf{g}^{\nabla}_{\overline{K}\oplus \mathtt{ipad}}(IV), \mathsf{g}^{\nabla}_{\overline{K}\oplus \mathtt{opad}}(IV)) = (K^I_{\mathsf{Csc}}, K^O_{\mathsf{Csc}}),$$
$$\mathsf{leak}_2(K, M_i) = \mathsf{H}(\overline{K} \oplus \mathtt{ipad} \,||\, M_i) = H^I_i, \quad 1 \leqslant i \leqslant q.$$

Противник может вычислить $Y^I_i = \mathsf{Csc}(K^I_{\mathsf{Csc}}, M_i)$ и $Y^O_i = \mathsf{Csc}(K^O_{\mathsf{Csc}}, H^I_i)$, $1 \leqslant i \leqslant q$.

Теорема 3. Преобладание любого противника, атакующего HMAC-Стрибог-τ в модели PRF-LEAK, ограничено:

$$\mathrm{Adv}^{\text{PRF-LEAK}}_{\mathsf{HMAC},\mathsf{leak}_1,\mathsf{leak}_2}(t,q,l) \leqslant \mathrm{Adv}^{\text{PRF-RKA}_{\boxplus\circ\oplus}}_{\mathsf{g}^{\nabla}}(t',q',r,d) + \frac{q^2}{2^{\tau+1}} + \mathrm{Adv}^{\mathrm{CR}}_{\mathsf{g}}(t') + \mathrm{Adv}^{\mathrm{TPR}}_{\mathsf{g}}(t') + 2^{-n},$$

где $t' = t + ql$, $r = q' = 2q + 2$, $d = 2$, $\tau \in \{256, 512\}$.

Доказательство. Структура доказательства аналогична [6, Theorem (PRF-security of HMAC-Streebog)]. Так же, как и для (7), вероятность коллизий вида $Y_i^I = Y_j^I$, $Y_i^O = Y_j^O$, $Y_i^I = Y_j^O$ ограничивается за счёт CR-стойкости каскада, а вероятность события $IV \in \{Y_1^I, \ldots, Y_q^O\}$ — за счёт стойкости к атакам TPR.

Обозначим значения, относящиеся к первому и ко второму хешированию, с помощью верхних индексов «I» и «O» соответственно (рис. 3):

$$K^I = \overline{K} \oplus \texttt{ipad},\ H^I = \mathsf{H}(K^I \,||\, M),\ \ \hat{H}^I = \mathsf{msb}_\tau(H^I),\ Y^I = \mathsf{Csc}(K^I_{\mathsf{Csc}}, M),\ \sigma^I = \mathsf{sum}_{\boxplus}(M),$$
$$K^O = \overline{K} \oplus \texttt{opad}, H^O = \mathsf{H}(K^O \,||\, \hat{H}^I), \hat{H}^O = \mathsf{msb}_\tau(H^O), Y^O = \mathsf{Csc}(K^O_{\mathsf{Csc}}, \hat{H}^I), \sigma^O = \mathsf{sum}_{\boxplus}(\hat{H}^I).$$

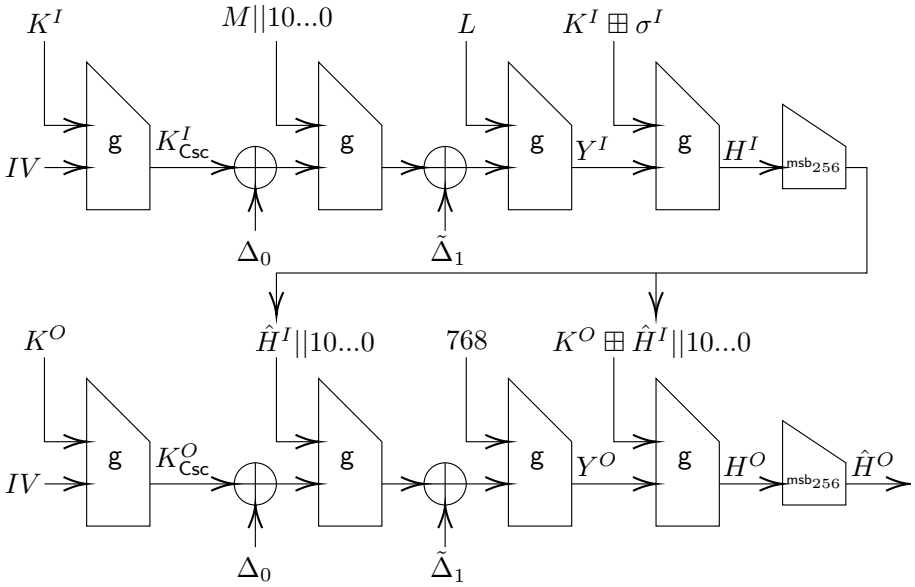

Рис. 3. HMAC-Стрибог-256, длина M равна $L < n$

Оракул $\mathcal{L}$ в ответ на запрос M возвращает $(\hat{H}^O, H^I) \in V^\tau \times V^n$ — непосредственное хеш-значение, сформированное криптоалгоритмом, и промежуточное состояние.

Рассмотрим «идеализированный» оракул $\widetilde{\mathcal{L}}$

$$\widetilde{\mathcal{L}}(M) = \left(\mathsf{msb}_\tau \mathsf{f}_{K^O \boxplus \sigma^O}\left(\ldots \mathsf{f}_{K^I \boxplus \sigma^I}(\mathsf{Csc}(\mathsf{f}_{K^I}(IV), M))\right), \mathsf{f}_{K^I \boxplus \sigma^I}(\mathsf{Csc}(\mathsf{f}_{K^I}(IV), M))\right),$$

где первое и последнее преобразование $\mathsf{g}^\triangledown$ в обоих обращениях к хеш-функции заменено на семейство из 2^n случайных функций, индексируемых связанными ключами вида $(\overline{K} \oplus \phi) \boxplus \sigma$, $\phi \in \{\texttt{ipad}, \texttt{opad}\}$.

Коллизия (C) — совпадение любой пары в ряду из $(2q+1)$ значений:

$$IV,\ Y_1^I,\ \ldots,\ Y_q^I,\ Y_1^O,\ \ldots,\ Y_q^O.$$

Если коллизия не реализуется ($\overline{\text{C}}$), то $\widetilde{\mathcal{L}}$ даже при известных противнику каскадных ключах неотличим от случайной функции $\mathsf{R} \in \mathrm{Func}(V^{<2^n}, V^\tau \times V^n)$:

$$\Pr\left[\mathcal{A}^{\mathsf{R}}(lk) \Rightarrow 1\right] = \Pr\left[\mathcal{A}^{\widetilde{\mathcal{L}}}(\mathsf{f}_{K^I}(IV), \mathsf{f}_{K^O}(IV)) \Rightarrow (1, \overline{\text{C}})\right], \quad lk \in V^n \times V^n,$$

так как все значения $(K^I_{\mathsf{Csc}}, K^O_{\mathsf{Csc}}, H^I_1, \ldots, H^I_q, \hat{H}^O_1, \ldots, \hat{H}^O_q)$, которые наблюдал противник, получены в результате различных запросов к f. Следовательно,

$$\mathrm{Adv}^{\text{PRF-LEAK}}_{\mathsf{HMAC}, \mathsf{leak}_1, \mathsf{leak}_2}(\mathcal{A}) = \left(\Pr\left[\mathcal{A}^{\mathcal{L}} \Rightarrow (1, \text{C})\right] + \Pr\left[\mathcal{A}^{\mathcal{L}} \Rightarrow (1, \overline{\text{C}})\right]\right) - \Pr\left[\mathcal{A}^{\widetilde{\mathcal{L}}} \Rightarrow (1, \overline{\text{C}})\right] \leqslant$$
$$\leqslant \left(\Pr\left[\mathcal{A}^{\mathcal{L}} \Rightarrow (1, \overline{\text{C}})\right] - \Pr\left[\mathcal{A}^{\widetilde{\mathcal{L}}} \Rightarrow (1, \overline{\text{C}})\right]\right) + \left(\Pr\left[\mathcal{A}^{\mathcal{L}} \Rightarrow \text{C}\right] - \Pr\left[\mathcal{A}^{\widetilde{\mathcal{L}}} \Rightarrow \text{C}\right]\right) + \Pr\left[\mathcal{A}^{\widetilde{\mathcal{L}}} \Rightarrow \text{C}\right] = \epsilon + \epsilon_{\text{C}} + p_{\text{C}},$$

противник $\mathcal{A}$ перед взаимодействием с оракулом $\mathcal{L}$ получает $(K^I_{\mathsf{Csc}}, K^O_{\mathsf{Csc}})$, а перед взаимодействием с $\widetilde{\mathcal{L}}$ — пару $(\mathsf{f}_{K^I}(IV), \mathsf{f}_{K^O}(IV))$.

Построим алгоритм $\mathcal{B}$, атакующий $\mathsf{g}^\triangledown$ в модели PRF-RKA$_{\boxplus\circ\oplus}$. Алгоритм $\mathcal{B}$ запрашивает каскадные ключи $K^I_{\mathsf{Csc}} = \mathcal{O}(IV, (\mathtt{ipad}, 0))$ и $K^O_{\mathsf{Csc}} = \mathcal{O}(IV, (\mathtt{opad}, 0))$ у оракула $\mathcal{O} \in \{\mathsf{g}^\triangledown, \mathsf{f}\}$, а затем запускает алгоритм $\mathcal{A}$, передавая ему $(K^I_{\mathsf{Csc}}, K^O_{\mathsf{Csc}})$.

При обработке запроса M_i от $\mathcal{A}$ алгоритм $\mathcal{B}$: вычисляет $Y^I_i = \mathsf{Csc}(K^I_{\mathsf{Csc}}, M_i)$ и $\sigma^I_i = \mathsf{sum}_{\boxplus}(M_i)$; сохраняет Y^I_i в памяти; проверяет условие коллизии; делает запрос $(Y^I_i, (\mathtt{ipad}, \sigma^I_i))$ к оракулу; получает H^I_i; вычисляет $Y^O_i = \mathsf{Csc}(K^O_{\mathsf{Csc}}, \hat{H}^I_i)$ и $\sigma^O_i = \mathsf{sum}_{\boxplus}(\hat{H}^I_i)$; сохраняет Y^O_i в памяти; проверяет условие коллизии; делает запрос $(Y^O_i, (\mathtt{opad}, \sigma^O_i))$; получает H^O_i; передаёт алгоритму $\mathcal{A}$ пару $(\hat{H}^O_i, H^I_i)$, $1 \leqslant i \leqslant q$.

Если проверка на каком-то из шагов показывает, что коллизия реализовалась, то $\mathcal{B}$ прекращает взаимодействие с $\mathcal{A}$ и возвращает 1. Если после обработки всех запросов от $\mathcal{A}$ коллизии нет, то результат работы $\mathcal{B}$ равен результату работы $\mathcal{A}$. К оракулу $\mathcal{O} \in \{\mathsf{g}^\triangledown, \mathsf{f}\}$ выполняется до $q' = (2q + 2)$ запросов, количество связанных ключей оценивается так же ($r = q'$). Только значение IV запрашивается при двух заведомо различных связанных ключах ($\mathtt{ipad} \neq \mathtt{opad}$), любой другой вход обрабатывается ровно один раз, следовательно, $d = 2$.

Преобладание алгоритма $\mathcal{B}$ равно

$$\mathrm{Adv}^{\text{PRF-RKA}_{\boxplus\circ\oplus}}_{\mathsf{g}^\triangledown}(\mathcal{B}) = \Pr\left[\mathcal{B}^{\mathsf{g}^\triangledown} \Rightarrow 1\right] - \Pr\left[\mathcal{B}^{\mathsf{f}} \Rightarrow 1\right] = \epsilon + \epsilon_{\mathrm{C}}.$$

Оценим сверху вероятность коллизии:

$$p_{\mathrm{C}} \leqslant \Pr\left[Y^I_i = Y^I_j,\ 1 \leqslant i < j \leqslant q\right] + \Pr\left[Y^O_i = Y^O_j,\ 1 \leqslant i < j \leqslant q\right] +$$
$$+ \Pr\left[Y^I_i = Y^O_j,\ 1 \leqslant i, j \leqslant q\right] + \Pr\left[IV \in \{Y^I_1, \ldots, Y^I_q, Y^O_1, \ldots, Y^O_q\}\right] = p^I_{\mathrm{C}} + p^O_{\mathrm{C}} + p^{I,O}_{\mathrm{C}} + p^{\mathrm{pre}}_{\mathrm{C}}.$$

Построим алгоритм $\mathcal{B}_{\mathrm{CR}}$, пытающийся сформировать коллизию для Csc. Он работает так же, как $\mathcal{B}$, но вместо запросов к оракулу $\mathcal{O} \in \{\mathsf{g}^\triangledown, \mathsf{f}\}$ самостоятельно имитирует семейство случайных функций f и сохраняет в памяти (Y^I_i, M^I_i) и $(Y^O_i, \hat{H}^I_i)$. Если на каком-то шаге обнаружена коллизия и при этом $IV \notin \{Y^I_1, \ldots, Y^O_q\}$, то возможны три случая:

1) $Y^I_i = Y^I_j$, $i \neq j$, даёт коллизию $((K^I_{\mathsf{Csc}}, M_i), (K^I_{\mathsf{Csc}}, M_j))$ в силу $M_i \neq M_j$;
2) $Y^O_i = Y^O_j$, $i \neq j$, даёт коллизию $((K^O_{\mathsf{Csc}}, \hat{H}^I_i), (K^O_{\mathsf{Csc}}, \hat{H}^I_j))$ при условии $\hat{H}^I_i \neq \hat{H}^I_j$;
3) $Y^I_i = Y^O_j$ даёт коллизию $((K^I_{\mathsf{Csc}}, M_i), (K^O_{\mathsf{Csc}}, \hat{H}^I_j))$, если хотя бы $K^I_{\mathsf{Csc}} \neq K^O_{\mathsf{Csc}}$.

Таким образом, вероятность успеха алгоритма $\mathcal{B}_{CR}$ оценивается как

$$\mathrm{Adv}^{\mathrm{CR}}_{\mathsf{Csc}}(\mathcal{B}_{\mathrm{CR}}) \geqslant (p^I_{\mathrm{C}} + p^O_{\mathrm{C}} + p^{I,O}_{\mathrm{C}}) - \Pr\left[K^I_{\mathsf{Csc}} = K^I_{\mathsf{Csc}}\right] - \Pr\left[\exists i \neq j\, (\hat{H}^I_i = \hat{H}^I_j)\right] \geqslant$$
$$\geqslant (p^I_{\mathrm{C}} + p^O_{\mathrm{C}} + p^{I,O}_{\mathrm{C}}) - 2^{-n} - \frac{q^2}{2^{\tau+1}}.$$

По аналогии с $\mathcal{B}_{\mathrm{CR}}$ построим алгоритм $\mathcal{B}_{\mathrm{TPR}}$, который ищет для Csc прообраз к значению IV. Искомым прообразом станет либо $(K^I_{\mathsf{Csc}}, M_i)$, либо $(K^O_{\mathsf{Csc}}, \hat{H}^I_i)$,

$$p^{\mathrm{pre}}_{\mathrm{C}} = \mathrm{Adv}^{\mathrm{TPR}}_{\mathsf{Csc}}(\mathcal{B}_{\mathrm{TPR}}).$$

Вычислительные ресурсы алгоритмов $\mathcal{B}$, $\mathcal{B}_{\mathrm{CR}}$ и $\mathcal{B}_{\mathrm{TPR}}$ превосходят ресурсы $\mathcal{A}$ на значение, пропорциональное объёму обрабатываемых данных, $t' = t + ql$. Преобладание $\mathcal{B}_{\mathrm{CR}}$

и $\mathcal{B}_{\mathrm{TPR}}$ в соответствующих моделях оценивается неравенствами (5) и (6) соответственно. Пользуясь неравенством треугольника, получаем доказываемое утверждение. ■

Не является стойким в модели PRF-LEAK, например, режим имитозащиты CMAC [36] — утечка промежуточного состояния сразу даёт противнику возможность сформировать подделку.

Раскрытие состояния приводит к потере стойкости у алгоритмов имитозащиты, основанных на однократном применении хеш-функций типа HAIFA [37], например Skein-MAC [38] и BLAKE2-MAC [39], а также у схем, построенных по схеме «губка» («sponge») — KMAC [40]. Вместе с тем из криптографической хеш-функции за счёт применения двойного хеширования или «сэндвича» обычно нетрудно сделать PRF-LEAK-стойкий криптоалгоритм.

6. Стойкость к атакам на ключ

Количественные оценки в модели PRF и PRF-LEAK одинаковы для трёх анализируемых схем. Кроме того, при $k \leqslant n/2 = 256$ единственным эффективным методом нарушения свойств безопасности является тотальное опробование ключа (пусть и в разных предположениях о функции сжатия $\mathsf{g}^{\triangledown}$).

При k, близком к n, и при отсутствии ограничений на объём обрабатываемого материала Стрибог-K и HMAC-Стрибог подвержены нетривиальным атакам на восстановление секретного ключа. Существуют методы с трудоёмкостью порядка $t \approx q \cdot l \approx 2^{4n/5}$ по времени и данным [9]. Простой подстановкой в (8) можно убедиться, что наличие таких атак не противоречит оценке, полученной с помощью доказательного подхода.

Атаки на ключ [9] дают по сути оценку снизу на вероятность успеха противника в модели KR. Получим верхние оценки в этой модели, а также в условиях утечки внутреннего состояния (KR-LEAK).

Схема «сэндвич» в моделях KR и KR-LEAK является более стойкой, чем два других криптоалгоритма.

Определение 8. Характеристикой успешности противника $\mathcal{A}$ в модели KR-LEAK для ключевого алгоритма $\mathsf{F} : \mathbf{K} \times \mathbf{X} \to \mathbf{Y}$ при утечке, определяемой функциями $\mathsf{leak}_1 : \mathbf{K} \to \mathbf{L}_1$ и $\mathsf{leak}_2 : \mathbf{K} \times \mathbf{X} \to \mathbf{L}_2$, назовём вероятность восстановления ключа:

$$\mathrm{Adv}^{\text{KR-LEAK}}_{\mathsf{F},\mathsf{leak}_1,\mathsf{leak}_2}(\mathcal{A}) = \Pr\Big[K \xleftarrow{\mathrm{R}} \mathbf{K} : \mathcal{A}^{\mathcal{L}}(\mathsf{leak}_1(K)) \Rightarrow K',\ K' = K\Big].$$

Оракул $\mathcal{L}$ на запрос $X \in \mathbf{X}$ возвращает $(\mathsf{F}(K, X), \mathsf{leak}_2(K, X))$.

При одинаковых функциях утечки модели PRF-LEAK и KR-LEAK связаны так же, как модели PRF и KR.

Утверждение 1. Вероятность успеха противника, пытающегося определить секретный ключ K криптоалгоритма Стрибог-C (при утечке K_{Csc}), или Стрибог-K (при утечке K_{Csc}), или HMAC-Стрибог (при утечке $\mathsf{leak}_1(K) = (K^I_{\mathsf{Csc}}, K^O_{\mathsf{Csc}})$ и $\mathsf{leak}_2(K, M_i) = H^I_i$, $1 \leqslant i \leqslant q$), ограничена сверху соответственно:

$$\mathrm{Adv}^{\text{KR-LEAK}}_{\mathsf{SH},K_{\mathsf{Csc}}}(t,q,l) \leqslant \mathrm{Adv}^{\text{KR}}_{\mathsf{g}^{\triangledown}}(t',q') \lessapprox \frac{t'}{2^k},$$

$$\mathrm{Adv}^{\text{KR-LEAK}}_{\mathsf{KH},K_{\mathsf{Csc}}}(t,q,l) \leqslant \mathrm{Adv}^{\text{KR-RKA}_{\boxplus}}_{\mathsf{g}^{\triangledown}}(t',q') \lessapprox \frac{t' \cdot q'}{2^k}; \tag{12}$$

$$\mathrm{Adv}^{\text{KR-LEAK}}_{\mathsf{HMAC},\mathsf{leak}_1,\mathsf{leak}_2}(t,q,l) \leqslant \mathrm{Adv}^{\text{KR-RKA}_{\boxplus\circ\oplus}}_{\mathsf{g}^{\triangledown}}(t',2q') \lessapprox \frac{2 \cdot t' \cdot q'}{2^k}, \tag{13}$$

где $t' = t + ql$, $q' = q + 1$.

Доказательство. Пусть противник $\mathcal{A}$ восстанавливает ключ алгоритма SH с вероятностью успеха p. Построим $\mathcal{B}$, атакующий $\mathsf{g}^{\triangledown}$. За счёт первого запроса получим $K_{\mathsf{Csc}} = \mathsf{g}^{\triangledown}_{K}(IV)$, передадим K_{Csc} алгоритму $\mathcal{A}$. На каждый запрос M_i от $\mathcal{A}$ к SH: самостоятельно вычисляем $Y_i = \mathsf{Csc}(K_{\mathsf{Csc}}, M_i||\mathsf{cs}(M_i))$; получаем у оракула $H_i = \mathsf{g}^{\triangledown}_{K}(Y_i)$ и возвращаем его $\mathcal{A}$. Для $\mathcal{A}$ идеально симулируется SH, а следовательно, вероятность успеха $\mathcal{B}$ равна p.

Для KH и HMAC аналогично, но $\mathsf{g}^{\triangledown}$ рассматривается относительно атак со связанными ключами. ■

Таким образом, единственным эффективным методом определения ключа схемы «сэндвич» даже при утечке K_{Csc} является метод тотального опробования (в предположении, что для функции сжатия эффективным является только тотальное опробование). Для KH и HMAC это не так, но утверждение показывает, что вероятность успеха атакующего растёт не более чем линейно с ростом числа сообщений (q) и практически не зависит от их длины (l), так как вычислительные мощности противника не меньше доступных ему данных ($t > q \cdot l$). Оптимум достигается при $t \approx q \approx 2^{k/2}$. Учитывая (11), при произвольном k получим

$$\mathrm{Adv}^{\text{KR-LEAK}}_{\mathsf{KH},K_{\mathsf{Csc}}}(t,q,l) \lessapprox \min\left(\frac{t'\cdot q'}{2^k}, \frac{t'}{2^k} + \frac{t'^2}{2^{n+1}} + \frac{t'}{2^n}\right),\ t' \approx t,\ q' = q+1,$$

и схожую оценку для HMAC.

В модели KR-LEAK оценки (12) и (13) являются точными, в [5] для схожих условий описана атака, основанная на построении мультиколлизий для каскадного преобразования. Оценки верны и в модели KR, но в этом случае говорить об их точности нельзя, атаки [9] требуют $t \approx 2^{4n/5}$, что много больше, чем $t \approx 2^{n/2}$.

7. Подходы к защите от атак по побочным каналам

Функции утечки в моделях KR-LEAK и PRF-LEAK никак не связаны с природой процессов, из-за которых противник получает дополнительные сведения. Используемый при доказательстве стойкости конкретный вид этих функций говорит о том, что полный доступ противника к содержащейся в них информации не приводит к нарушению свойств безопасности. Отсюда важное следствие: те сведения, которые не описаны функциями утечки (и не могут быть вычислены на их основе), являются критически важными и подлежат защите от раскрытия.

Конкретные меры защиты, в первую очередь маскирование [14, 41], существенным образом зависят от специфики реализации криптоалгоритма и физической модели рассматриваемых побочных каналов и не являются предметом настоящей работы. Здесь опишем части криптоалгоритмов, которые следует защищать, а также укажем на общие проблемы, возникающие при реализации соответствующей защиты.

Во всех представленных ранее результатах каскадный ключ K_{Csc} (пара ключей для HMAC-Стрибог) раскрывался противнику. Предполагаем, что каскадный ключ (ключи) вычисляется однократно и хранится в памяти.

Схема «сэндвич» и Стрибог-K схожи. Для любого сообщения M противник может вычислить:

$$\begin{aligned} \mathsf{SH}: Y &= \mathsf{Csc}(K_{\mathsf{Csc}}, M \,||\, \mathsf{cs}(M)), \quad \sigma = 0,\\ \mathsf{KH}: Y &= \mathsf{Csc}(K_{\mathsf{Csc}}, M), \qquad\quad \sigma = \mathsf{sum}_{\boxplus}(M). \end{aligned}$$

Сложение ключа $\overline{K}$ с блоками сообщения должно происходить в KH однократно: сначала вычисляется σ, затем $\overline{K} \boxplus \sigma$. В SH ключ вообще не должен складываться с блоками сообщения.

Последний вызов функции сжатия:

$$\begin{gathered} H = \mathsf{g}^{\nabla}_{\overline{K} \boxplus \sigma}(Y) = (\overline{K} \boxplus \sigma) \oplus Y \oplus \mathsf{E}(Y, \overline{K} \boxplus \sigma) = \\ = (\overline{K} \boxplus \sigma) \oplus Y \oplus \mathsf{X}_{13}\,\mathsf{L}\,\mathsf{P}\,\mathsf{S}\,\mathsf{X}_{12} \ldots \mathsf{L}\,\mathsf{P}\,\mathsf{S}\,\mathsf{X}_{1}(\overline{K} \boxplus \sigma), \quad \overline{K} = K \,||\, 0^{n-k}. \end{gathered} \tag{14}$$

Раундовые ключи шифра E формируются из состояния Y, а следовательно, также известны противнику. Функция сжатия сама по себе является стойкой в таких условиях [7, 8]. Однако если раскрывается промежуточное состояние, например $s^{(1)} = \mathsf{L}\,\mathsf{P}\,\mathsf{S}\,\mathsf{X}_1(\overline{K} \boxplus \sigma)$, то противник легко вычислит ключ:

$$\begin{gathered} s^{(13)} = \mathsf{X}_{13}\,\mathsf{L}\,\mathsf{P}\,\mathsf{S}\,\mathsf{X}_{12} \ldots \mathsf{L}\,\mathsf{P}\,\mathsf{S}\,\mathsf{X}_{2}(s^{(1)}), \\ \overline{K} = \left(H \oplus Y \oplus s^{(13)}\right) \boxminus \sigma. \end{gathered}$$

Необходима защита всех внутренних состояний шифра на протяжении 13 раундов.

Вычисления каскадного преобразования могут, следовательно, осуществляться «быстрым» незащищённым образом, а последний вызов шифра E — «медленным» защищённым, например на отдельном внешнем (по отношению к остальной вычислительной системе) модуле.

У Стрибог-С контрольная сумма всегда равна нулю ($\sigma = 0$), сложение по модулю 2^n отсутствует, что позволяет применять для защиты LPSX-преобразований (14) хорошо известные методы маскирования [14, 41, 42], в том числе использующие специфику нелинейного преобразования [16], а также методы, основанные на пороговой реализации (threshold implementation) [15, 43–45].

Для защиты криптоалгоритма Стрибог-К из-за произвольности значения σ может потребоваться применение вспомогательных алгоритмов, что снижает скорость работы и усложняет реализацию. Пусть, например, вместо ключа используется пара «маскированный ключ, маска» $(\overline{K} \boxplus W, W)$. Тогда после сложения с КС получим $(\overline{K} \boxplus W \boxplus \sigma, W)$. В шифре E используется операция $\oplus$, что потребует вычисления пары $(\overline{K} \boxplus \sigma \oplus W', W')$, например, с помощью [11]. Если одновременное хранение ключа под разными масками невозможно, то переход от $(\overline{K} \boxplus \sigma \oplus W', W')$ к $(\overline{K} \boxplus W'', W'')$ также потребует применения специального алгоритма [10].

У криптоалгоритма HMAC-Стрибог защищать требуется два вызова блочного шифра, при первом и втором хешировании. Противнику известны значения

$$\begin{aligned} Y^I &= \mathsf{Csc}(K^I_{\mathsf{Csc}}, M), & \sigma^I &= \mathsf{sum}_{\boxplus}(M), & H^I &= \mathsf{g}^{\nabla}_{\overline{K} \oplus \mathtt{ipad} \boxplus \sigma^I}(Y^I), \\ Y^O &= \mathsf{Csc}(K^O_{\mathsf{Csc}}, \mathsf{msb}_\tau(H^I)), & \sigma^O &= \mathsf{sum}_{\boxplus}(\mathsf{msb}_\tau(H^I)), & H^O &= \mathsf{g}^{\nabla}_{\overline{K} \oplus \mathtt{opad} \boxplus \sigma^O}(Y^O), \end{aligned}$$

а также раундовые ключи шифра E, формируемые из Y^I и Y^O. В общем случае $\sigma^I \neq 0$ и $\sigma^O \neq 0$, что приводит к необходимости двукратного применения таких же вспомогательных алгоритмов [10, 11], как и для Стрибог-К. Вычисление $Y^O = \mathsf{Csc}(K^O_{\mathsf{Csc}}, \ldots)$ требует два ($\tau = 256$) или три ($\tau = 512$) обращения к функции сжатия, что затрудняет реализацию E на защищённом внешнем модуле. К такому модулю потребуется либо делать два запроса, либо реализовывать вычисление Y^O внутри него.

Заключение

В работе предложен простой способ преобразования отечественной хеш-функции в ключевой криптоалгоритм по схеме «сэндвич» (Стрибог-С), к хешируемому тексту приписывается специальный блок, играющий роль «альтернативной» контрольной суммы, а в саму хеш-функцию никаких изменений не вносится.

Благодаря указанному приёму, сжимающее преобразование $\mathtt{g}^{\triangledown}$ (первое и последнее в хеш-функции) используется только с ключом K, а не со связанными ключами вида $K \boxplus \sigma$, как в алгоритмах HMAC-Стрибог и Стрибог-K.

Схема «сэндвич» является стойкой псевдослучайной функцией (PRF) и, следовательно, стойким алгоритмом имитозащиты, при этом функция $\mathtt{g}^{\triangledown}$ должна быть PRF, но стойкости к атакам со связанными ключами от неё не требуется. Аналогичное соображение позволяет показать, что при любом объёме обрабатываемых данных для Стрибог-С не существует более эффективного метода определения ключа, чем тотальное опробование, если то же самое верно для функции сжатия $\mathtt{g}^{\triangledown}$.

Предложены модели угроз PRF-LEAK и KR-LEAK, в рамках которых противник (решающий задачу различения или пытающийся восстановить ключ соответственно) получает непосредственный доступ к (почти всем) внутренним состояниям криптоалгоритма, происходит их раскрытие — утечка. Доказано, что Стрибог-С, Стрибог-K и HMAC-Стрибог являются стойкими в этих моделях при дополнительном предположении о стойкости функции сжатия к атакам на построение коллизий и прообраза. Справедливость предположения подтверждается представленными в открытой литературе конструктивными исследованиями. Стойкость в этих моделях делает указанные криптоалгоритмы предпочтительнее ряда схем, основанных на блочных шифрах, например режима имитозащиты ГОСТ 34.13-2018.

Выполненный анализ позволил выявить части хеш-функции, которые при реализации ключевых криптоалгоритмов подлежат защите от каких-либо утечек. Защищать требуется только последнюю функцию сжатия, а точнее, используемый внутри неё блочный шифр (без алгоритма развёртки ключа) и сам вход шифра (которым является секретный ключ K или $K \boxplus \sigma$). Для HMAC-Стрибог защита требуется и при первом, и при втором хешировании. Отсутствие у схемы «сэндвич» связанных ключей и, как следствие, попеременного использования операций $\boxplus$ и $\oplus$ в ряде случаев, как представляется, позволяет существенно упростить реализацию мер защиты.

ЛИТЕРАТУРА

1. ГОСТ 34.11-2018. Информационная технология. Криптографическая защита информации. Функция хэширования. М.: Стандартинформ, 2018.
2. *Damgard I.* A design principle for hash functions // LNCS. 1990. V. 435. P. 416–427.
3. *Merkle R.* One way hash functions and DES // LNCS. 1990. V. 435. P. 428–446.
4. Р 50.1.113-2016. Информационная технология. Криптографическая защита информации. Криптографические алгоритмы, сопутствующие применению алгоритмов электронной цифровой подписи и функции хэширования. М.: Стандартинформ, 2016.
5. *Kiryukhin V. A.* Keyed Streebog is a secure PRF and MAC // Матем. вопр. криптогр. 2023. Т. 14. № 2. С. 77–96.
6. *Kiryukhin V. A.* About "k-bit Security" of MACs Based on Hash Function Streebog. Cryptology ePrint Archive. Paper 2023/1305. 2023. https://eprint.iacr.org/2023/1305.
7. *Kiryukhin V. A.* Streebog compression function as PRF in secret-key settings // Матем. вопр. криптогр. 2022. Т. 13. № 2. С. 99–116.

8. *Kiryukhin V. A.* Related-key attacks on the compression function of Streebog // Матем. вопр. криптогр. 2023. Т. 14. № 2. С. 59–76.
9. *Dinur I. and Leurent G.* Improved generic attacks against hash-based MACs and HAIFA // LNCS. 2014. V. 8616. P. 149–168.
10. *Goubin L.* A Sound method for switching between Boolean and arithmetic masking // LNCS. 2001. V. 2162. P. 3–15.
11. *Coron J., Großschädl J., Tibouchi M., and Vadnala P. K.* Conversion from arithmetic to Boolean masking with Logarithmic complexity // LNCS. 2015. V. 9054. P. 130–149.
12. *Yasuda K.* "Sandwich" is indeed secure: How to authenticate a message with just one hashing // LNCS. 2007. V. 4586. P. 355–369.
13. *Bellare M., Goldreich O., and Mityagin A.* The Power of Verification Queries in Message Authentication and Authenticated Encryption. Cryptology ePrint Archive. Paper 2004/304. 2004. `https://eprint.iacr.org/2004/304`.
14. *Blömer J., Merchan J., and Krummel V.* Provably secure masking of AES // LNCS. 2004. V. 3357. P. 69–83.
15. *Nikova S., Rechberger C., and Rijmen V.* Threshold implementations against side-channel attacks and glitches // LNCS. 2006. V. 4307. P. 529–545.
16. *Lavrenteva T. A. and Matveev S. V.* Side-channel attacks countermeasure based on decomposed S-boxes for Kuznyechik // Матем. вопр. криптогр. 2021. Т. 12. № 2. С. 147–157.
17. *Bellare M. and Rogaway P.* Introduction to Modern Cryptography. 2005. `https://web.cs.ucdavis.edu/~rogaway/classes/227/spring05/book/main.pdf`.
18. *Bernstein D. J. and Lange T.* Non-uniform cracks in the concrete: The power of free precomputation // LNCS. 2013. V. 8270. P. 321–340.
19. *Guo J., Jean J., Leurent G., et al.* The usage of counter revisited: Second-preimage attack on new Russian standardized hash function // LNCS. 2014. V. 8781. P. 195–211.
20. *Abdelkhalek A., AlTawy R., and Youssef A. M.* Impossible differential properties of reduced round Streebog // LNCS. 2015. V. 9084. P. 274–286.
21. *Rogaway P. and Shrimpton T.* Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance // LNCS. 2004. V. 3017. P. 371–388.
22. *AlTawy R., Kircanski A., and Youssef A. M.* Rebound attacks on Stribog // LNCS. 2014. V. 8565. P. 175–188.
23. *Lin D., Xu S., and Yung M.* Cryptanalysis of the round-reduced GOST hash function // LNCS. 2014. V. 8567. P. 309–322.
24. *Ma B., Li B., Hao R., and Li X.* Improved cryptanalysis on reduced-round GOST and Whirlpool hash function // LNCS. 2014. V. 8479. P. 289–307.
25. *Wang Z., Yu H., and Wang X.* Cryptanalysis of GOST R hash function // Inform. Processing Lett. 2014. V. 114. P. 655–662.
26. *Kölbl S. and Rechberger C.* Practical attacks on AES-like cryptographic hash functions // LNCS. 2014. V. 8895. P. 259–273.
27. *Van Oorschot P. C. and Wiener M. J.* Parallel collision search with cryptanalytic applications // J. Cryptology. 1999. V. 12. No. 1. P. 1–28.
28. *AlTawy R. and Youssef A. M.* Preimage attacks on reduced-round Stribog // LNCS. 2014. V. 8469. P. 109–125.
29. *Ma B., Li B., Hao R., and Li X.* Improved (pseudo) preimage attacks on reduced-round GOST and Grostl-256 and studies on several truncation patterns for AES-like compression functions // LNCS. 2015. V. 9241. P. 79–96.

30. *Hua J., Dong X., Sun S., et al.* Improved MITM Cryptanalysis on Streebog. Cryptology ePrint Archive. Paper 2022/568. 2022. https://eprint.iacr.org/2022/568.
31. *Bellare M., Jaeger J., and Len J.* Better than advertised: Improved collision-resistance guarantees for MD-based hash functions // Proc. CCS'17. N.Y.: ACM, 2017. P. 891–906.
32. *Bellare M.* New proofs for NMAC and HMAC: Security without collision-resistance // LNCS. 2014. V. 4117. P. 602–619.
33. *Koblitz N. and Menezes A.* Another look at HMAC // J. Math. Cryptology. 2013. V. 7:3. P. 225–251.
34. Репозиторий «Ключевой Стрибог». https://gitflic.ru/project/vkir/streebog.
35. ГОСТ Р 34.11-94. Информационная технология. Криптографическая защита информации. Функция хэширования. М.: Издательство Стандартов, 1994.
36. ГОСТ 34.13-2018. Информационная технология. Криптографическая защита информации. Режимы работы блочных шифров. М.: Стандартинформ, 2018.
37. *Biham E. and Dunkelman O.* A Framework for Iterative Hash Functions — HAIFA. Cryptology ePrint Archive. Report 2007/278. 2007. https://eprint.iacr.org/2007/278.
38. *Ferguson N., Lucks S., Schneier B., et al.* The Skein Hash Function Family. 2009. https://api.semanticscholar.org/CorpusID:59739596.
39. *Aumasson J., Neves S., Wilcox-O'Hearn Z., and Winnerlein C.* BLAKE2: Simpler, Smaller, Fast as MD5. IACR Cryptology ePrint Archive. Report 2013/322. 2013. https://eprint.iacr.org/2013/322.pdf.
40. *Kelsey J., Chang S., and Perlner R.* SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash and ParallelHash. NIST Special Publication 800-185. 2016. https://doi.org/10.6028/NIST.SP.800-185.
41. *Goubin L. and Patarin J.* DES and differential power analysis. The "Duplication" Method // LNCS. 1999. V. 1717. P. 158–172.
42. *Oswald E., Mangard S., Pramstaller N., and Rijmen V.* A side-channel analysis resistant description of the AES S-Box // LNCS. 2005. V. 3557. P. 413–423.
43. *Bilgin B., Nikova S., Nikov V., et al.* Threshold implementations of all 3×3 and 4×4 S-boxes // LNCS. 2012. V. 7428. P. 76–91.
44. *Daemen J.* Changing of the guards: A simple and efficient method for achieving uniformity in threshold sharing // LNCS. 2017. V. 10529. P. 137–153.
45. *Piccione E., Andreoli S., Budaghyan L., et al.* An optimal universal construction for the threshold implementation of bijective S-boxes // IEEE Trans. Inform. Theory. 2023. V. 69. No. 10. P. 6700–6710.

REFERENCES

1. GOST 34.11-2018. Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Funktsiya kheshirovaniya [GOST R 34.11-2018. Information Technology. Cryptographic Data Security. Hash-function]. Moscow, Standartinform Publ., 2018. (in Russian)
2. *Damgard I.* A design principle for hash functions. LNCS, 1990, vol. 435, pp. 416–427.
3. *Merkle R.* One way wash functions and DES. LNCS, 1990, vol. 435, pp. 428–446.
4. R 50.1.113-2016. Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Kriptograficheskie algoritmy, soputstvuyushchie primeneniyu algoritmov elektronnoy tsifrovoy podpisi i funktsii kheshirovaniya [R 50.1.113-2016. Information Technology. Cryptographic Data Security. Cryptographic Algorithms Accompanying the Use of Electronic Digital Signature Algorithms and Hash Functions]. Moscow, Standartinform Publ., 2016. (in Russian)
5. *Kiryukhin V. A.* Keyed Streebog is a secure PRF and MAC. Mat. Vopr. Kriptogr., 2023, vol. 14, iss. 2, pp. 77–96.

6. *Kiryukhin V. A.* About "k-bit Security" of MACs Based on Hash Function Streebog. Cryptology ePrint Archive, Paper 2023/1305, 2023, https://eprint.iacr.org/2023/1305.
7. *Kiryukhin V. A.* Streebog compression function as PRF in secret-key settings. Mat. Vopr. Kriptogr., 2022, vol. 13, iss. 2, pp. 99–116.
8. *Kiryukhin V. A.* Related-key attacks on the compression function of Streebog. Mat. Vopr. Kriptogr., 2023, vol. 14, iss. 2, pp. 59–76.
9. *Dinur I. and Leurent G.* Improved generic attacks against hash-based MACs and HAIFA. LNCS, 2014, vol. 8616, pp. 149–168.
10. *Goubin L.* A sound method for switching between Boolean and arithmetic masking. LNCS, 2001, vol. 2162, pp. 3–15.
11. *Coron J., Großschädl J., Tibouchi M., and Vadnala P. K.* Conversion from arithmetic to Boolean masking with logarithmic complexity. LNCS, 2015, vol. 9054, pp. 130–149.
12. *Yasuda K.* "Sandwich" is indeed secure: How to authenticate a message with just one hashing. LNCS, 2007, vol. 4586, pp. 355–369.
13. *Bellare M., Goldreich O., and Mityagin A.* The Power of Verification Queries in Message Authentication and Authenticated Encryption. Cryptology ePrint Archive, Paper 2004/304, 2004, https://eprint.iacr.org/2004/304.
14. *Blömer J., Merchan J., and Krummel V.* Provably secure masking of AES. LNCS, 2004, vol. 3357, pp. 69–83.
15. *Nikova S., Rechberger C., and Rijmen V.* Threshold implementations against side-channel attacks and glitches. LNCS, 2006, vol. 4307, pp. 529–545.
16. *Lavrenteva T. A. and Matveev S. V.* Side-channel attacks countermeasure based on decomposed S-boxes for Kuznyechik. Mat. vopr. kriptogr., 2021, vol. 12, iss. 2, pp. 147–157.
17. *Bellare M. and Rogaway P.* Introduction to Modern Cryptography. 2005. https://web.cs.ucdavis.edu/~rogaway/classes/227/spring05/book/main.pdf.
18. *Bernstein D. J. and Lange T.* Non-uniform cracks in the concrete: The power of free precomputation. LNCS, 2013, vol. 8270, pp. 321–340.
19. *Guo J., Jean J., Leurent G., et al.* The usage of counter revisited: Second-preimage attack on new Russian standardized hash function. LNCS, 2014, vol. 8781, pp. 195–211.
20. *Abdelkhalek A., AlTawy R., and Youssef A. M.* Impossible differential properties of reduced round Streebog. LNCS, 2015, vol. 9084, pp. 274–286.
21. *Rogaway P. and Shrimpton T.* Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. LNCS, 2004, vol. 3017, pp. 371–388.
22. *AlTawy R., Kircanski A., and Youssef A. M.* Rebound attacks on Stribog. LNCS, 2014, vol. 8565, pp. 175–188.
23. *Lin D., Xu S., and Yung M.* Cryptanalysis of the round-reduced GOST hash function. LNCS, 2014, vol. 8567, pp. 309–322.
24. *Ma B., Li B., Hao R., and Li X.* Improved cryptanalysis on reduced-round GOST and Whirlpool hash function. LNCS, 2014, vol. 8479, pp. 289–307.
25. *Wang Z., Yu H., and Wang X.* Cryptanalysis of GOST R hash function. Inform. Processing Lett., 2014, vol. 114, pp. 655–662.
26. *Kölbl S. and Rechberger C.* Practical attacks on AES-like cryptographic hash functions. LNCS, 2014, vol. 8895, pp. 259–273.
27. *Van Oorschot P. C. and Wiener M. J.* Parallel collision search with cryptanalytic applications. J. Cryptology, 1999, vol. 12, iss. 1, pp. 1–28.
28. *AlTawy R. and Youssef A. M.* Preimage attacks on reduced-round Stribog. LNCS, 2014, vol. 8469, pp. 109–125.

29. *Ma B., Li B., Hao R., and Li X.* Improved (pseudo) preimage attacks on reduced-round GOST and Grostl-256 and studies on several truncation patterns for AES-like compression functions. LNCS, 2015, vol. 9241, pp. 79–96.
30. *Hua J., Dong X., Sun S., et al.* Improved MITM Cryptanalysis on Streebog. Cryptology ePrint Archive, Paper 2022/568, 2022, https://eprint.iacr.org/2022/568.
31. *Bellare M., Jaeger J., and Len J.* Better than advertised: Improved collision-resistance guarantees for MD-based hash functions. Proc. CCS'17, N.Y., ACM, 2017, pp. 891–906.
32. *Bellare M.* New proofs for NMAC and HMAC: security without collision-resistance. LNCS, 2014, vol. 4117, pp. 602–619.
33. *Koblitz N. and Menezes A.* Another look at HMAC. J. Math. Cryptology, 2013, vol. 7:3, pp. 225–251.
34. Repository "Keyed Streebog", https://gitflic.ru/project/vkir/streebog.
35. GOST R 34.11-94. Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Funktsiya kheshirovaniya. [GOST R 34.11-94. Information Technology. Cryptographic Data Security. Hash-function]. Moscow, Izdatelstvo Standartov, 1994. (in Russian)
36. GOST 34.13-2018. Informatsionnaya tekhnologiya. Kriptograficheskaya zashchita informatsii. Rezhimy raboty blochnykh shifrov [GOST 34.13-2018. Information Technology. Modes of Operation for Block Ciphers]. Moscow, Standartinform Publ., 2018. (in Russian)
37. *Biham E. and Dunkelman O.* A Framework for Iterative Hash Functions — HAIFA. Cryptology ePrint Archive, Report 2007/278. 2007, https://eprint.iacr.org/2007/278.
38. *Ferguson N., Lucks S., Schneier B., et al.* The Skein Hash Function Family. 2009, https://api.semanticscholar.org/CorpusID:59739596.
39. *Aumasson J., Neves S., Wilcox-O'Hearn Z., and Winnerlein C.* BLAKE2: Simpler, Smaller, Fast as MD5. IACR Cryptology ePrint Archive, Report 2013/322, 2013, https://eprint.iacr.org/2013/322.pdf.
40. *Kelsey J., Chang S., and Perlner R.* NIST Special Publication 800-185. SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash and ParallelHash. 2016, https://doi.org/10.6028/NIST.SP.800-185.
41. *Goubin L. and Patarin J.* DES and differential power analysis. The "Duplication" method. LNCS, 1999, vol. 1717, pp. 158–172.
42. *Oswald E., Mangard S., Pramstaller N., and Rijmen V.* A side-channel analysis resistant description of the AES S-box. LNCS, 2005, vol. 3557, pp. 413–423.
43. *Bilgin B., Nikova S., Nikov V., et al.* Threshold implementations of all 3×3 and 4×4 S-boxes. LNCS, 2012, vol. 7428, pp. 76–91.
44. *Daemen J.* Changing of the guards: A simple and efficient method for achieving uniformity in threshold sharing. LNCS, 2017, vol. 10529, pp. 137–153.
45. *Piccione E., Andreoli S., Budaghyan L., et al.* An optimal universal construction for the threshold implementation of bijective S-boxes. IEEE Trans. Inform. Theory, 2023, vol. 69, iss. 10, pp. 6700–6710.