Ранее на Рускрипто. . .

Обзор результатов анализа хэш-функций ГОСТ Р 34.11-2012

Лавриков И.В., Маршалко Г.Б., Рудской В.И., Смышляев С.В., Шишкин В.А.

РусКрипто 2015

18 марта 2015 года

Обзор результатов анализа хэш-функций ГОСТ Р 34.11-2012 10 лет спустя

Матюхин Д.В., Маршалко Г.Б., Фомин Д.Б., Столовник Д.А.

Технический комитет по стандартизации ТК 26 «Криптографическая защита информации»

19 марта 2025 года

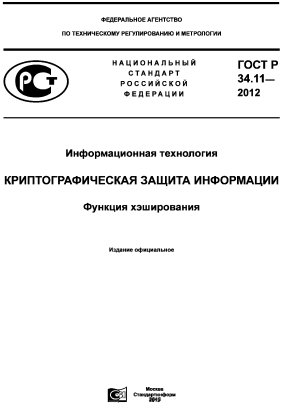
All characters appearing in this work are fictitious! Any resemblance to real persons, living or dead, is purely coincidental!

Дисклеймер:

Настоящая презентация представляет собой систематизированный обзор основных свойств хэш-функций «Стрибог» с акцентом на их интуитивное понимание и краткое описание. Все приведенные результаты основаны на существующих теоретических исследованиях и формальных обоснованиях, однако они не претендуют абсолютную строгость изложения.

Таким образом, материалы данной презентации *не могут быть использованы* в качестве строгой математической основы для обоснования криптографических и иных свойств механизмов, использующих хэш-функцию «Стрибог» в качестве примитива. Для таких целей рекомендуется обращаться к первичным научным источникам и официальным стандартам!

- В 2012 году в РФ принят новый национальный стандарт ГОСТ Р 34.11-2012, определяющий функции хэширования.
- В ходе разработки и стандартизации закрепилось название «Стрибог» (англ. «Stribog» или «Streebog»).
- В 2013 году спецификация хэш-функции ГОСТ Р 34.11-2012 опубликована в качестве рекомендаций IETF RFC 6986 «GOST R 34.11-2012: Hash Function».
- Хэш-функции из ГОСТ Р 34.11-2012 утверждены как межгосударственный стандарт ГОСТ 34.11-2018 Межгосударственным советом по стандартизации, метрологии и сертификации.
- В 2018 году включены в международный стандарт ISO/IEC 10118-3:2018.


ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р 34.11— 2012

Информационная технология

КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ

Функция хэширования

Издание официальное

Москва
Стандартинформ
2012

- Используются в 17 документах национальной системы стандартизации
- Более 30 механизмов используют хэш-функции в качестве примитива
- Необходимы в большом количестве проектов документов, в том числе при проектировании постквантовых криптографических механизмов
- Без хэш-функции сложно представить функционирование сложных современных информационных систем
- Опубликовано более 35 научных работ, посвященных описанию свойств

ОПРЕДЕЛЕНЫ ПОБЕДИТЕЛИ КОНКУРСА ПО ИССЛЕДОВАНИЮ ХЭШ-ФУНКЦИИ «СТРИБОГ»

Российский Технический комитет по стандартизации «Криптографическая защита информации» (ТК 26), Академия криптографии Российской Федерации и компания ИнфоТеКС сообщают об окончании Открытого конкурса научно-исследовательских работ, посвященных анализу криптографических качеств хэш-функции, определенной в национальном стандарте ГОСТ Р 34.11-2012 «Информационная технология. Криптографическая защита информации. Функция хэширования».

Победителями конкурса стали исследовательские коллективы из Сингапура, Канады и молодой ученый из России. Также Конкурсной комиссией было принято решение отметить поощрительным призом работу участников из Китая.

Экспертиза работ проводилось в период с 22 февраля по 4 марта. 5 марта состоялось заседание Конкурсной комиссии, в ходе которого с учетом результатов экспертизы были определены победители Конкурса в лице следующих исследователей и исследовательских групп:

Первая премия в размере 500 000 (пятьсот тысяч) рублей: J. Guo, J. Jean, G. Leurent, T. Peyrin и L. Wang за работу "The Usage of Counter Revisited: Second-Preimage Attack on New Russian Standardized Hash Function».

Две вторых премии в размере300 000 (триста тысяч) рублей: R. AlTawy и A.M.Youssef за цикл работ «A Study of Security Properties of GOST R 34.11-2012»; Г. Седов за работу «Стойкость ГОСТ 34.11-2012 к атаке поиска прообраза и к атаке поиска коллизий»

Дополнительно Конкурсной комиссией было принято решение поощрить работу «Improved Cryptanalysis on Reduced-Round GOST and Whirlpool Hash Function» группы исследователей B. Ma, B. Li, X. Li, and R. Hao премией в размере 150 000 (сто пятьдесят тысяч) рублей.

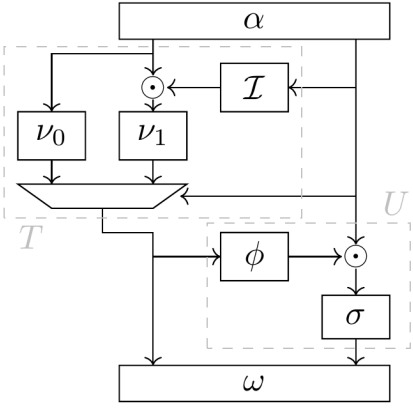
Исследования хэш-функций «Стрибог» (условно) можно разделить на следующие (вообще говоря пересекающиеся) классы

- Описание алгебраических свойств, поиск «эквивалентных представлений»
- Описание способов реализации хэш-функций (или примитивов)
- Получение оценок теоретической стойкости
  - хэш-функции
  - функции сжатия
- Уточнение оценок практической стойкости
  - хэш-функции
  - режимов работы
  - функции сжатия

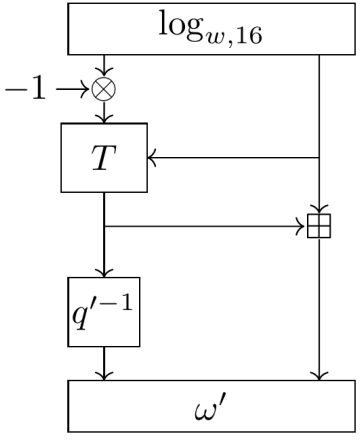
Известные результаты исследования особенностей примитивов, используемых в хэш-функции «Стрибог»

- «O. Kazymyrov и др., *Algebraic Aspects of the Russian Hash Standard GOST R 34.11-2012*, 2013» [19] – AES-подобное представление
- В работе «C. Beierle и др., *Decomposing Linear Layers*, 2022» [10] – продолжение изучения свойств линейного преобразования. Авторам удалось показать, что матрица линейного преобразования хэш-функций «Стрибог» является матрицей Коши.

$$A = \begin{bmatrix} \gamma^{1} & \gamma^{64} & \gamma^{66} & \gamma^{39} & \gamma^{133} & \gamma^{249} & \gamma^{94} & \gamma^{135} \\ \gamma^{249} & \gamma^{84} & \gamma^{150} & \gamma^{0} & \gamma^{210} & \gamma^{1} & \gamma^{221} & \gamma^{32} \\ \gamma^{100} & \gamma^{16} & \gamma^{155} & \gamma^{15} & \gamma^{167} & \gamma^{36} & \gamma^{182} & \gamma^{57} \\ \gamma^{220} & \gamma^{174} & \gamma^{246} & \gamma^{217} & \gamma^{216} & \gamma^{17} & \gamma^{90} & \gamma^{198} \\ \gamma^{116} & \gamma^{188} & \gamma^{217} & \gamma^{246} & \gamma^{124} & \gamma^{127} & \gamma^{237} & \gamma^{206} \\ \gamma^{37} & \gamma^{129} & \gamma^{147} & \gamma^{243} & \gamma^{36} & \gamma^{167} & \gamma^{154} & \gamma^{89} \\ \gamma^{77} & \gamma^{66} & \gamma^{64} & \gamma^{238} & \gamma^{206} & \gamma^{3} & \gamma^{136} & \gamma^{124} \\ \gamma^{135} & \gamma^{230} & \gamma^{73} & \gamma^{137} & \gamma^{164} & \gamma^{32} & \gamma^{134} & \gamma^{1} \end{bmatrix}, \qquad \gamma = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$



TU-представление подстановки из [11]



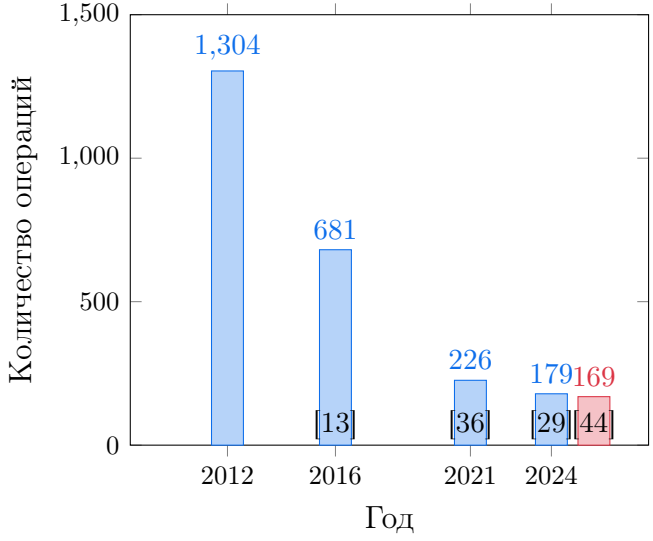
Логарифмическое представление подстановки из [28]

- В работе «L. Perrin, *Partitions in the S-Box of Streebog and Kuznyechik*, 2019» [27] авторы утверждают, что нашли такое представление и что подстановка была «намеренно спроектирована» с использованием определенной алгебраической структуры, которую они представили в статье и назвали TKlog
- Авторы объединили идеи предыдущих двух известных представлений и предложили свой вариант алгоритма вычисления подстановки

С момента публикации последней работы прошло 6 лет

На настоящий момент отсутствуют работы, показывающие хотя бы потенциальную возможность построения методов криптографического анализа хэш-функций, использующих любое представление подстановки. Однако, на основе эквивалентных представлений подстановки, стала возможна более эффективная ее реализация.

Изменение оценок комбинационной сложности реализации подстановки[1]



[1]Рассматривается базис $\Omega = \{\texttt{OR}, \texttt{XOR}, \texttt{AND}, \texttt{NOT}, 1\}$

Работа [17] стала прорывной в области анализа хэш-функций, что позволило в многочисленных последующих исследованиях уточнять ее оценки как теоретической, так и практической стойкости

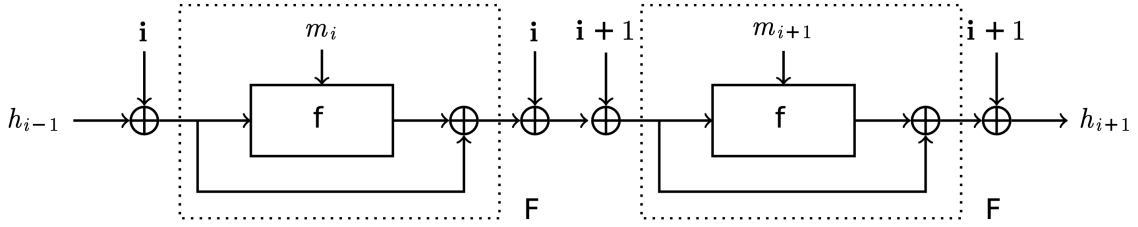


Рис.: Последовательное применение эквивалентного сжимающего преобразования к двум последовательным блокам

Обозначив $\Delta_i = \mathbf{i} \oplus (\mathbf{i}+\mathbf{1}) = (i \cdot 512) \oplus ((i+1) \cdot 512)$ получим эквивалентное представление формулы вычисления функции сжатия. В частности:

$$h_{i+1} = \mathsf{F}\left(\mathsf{F}\left(h_{i-1} \oplus \mathbf{i}, m_i\right) \oplus \Delta_i, m_{i+1}\right) \oplus \Delta_{i+1} \oplus (\mathbf{i}+\mathbf{2})$$

- Работа «R. AlTawy и др., *Watch your constants: malicious Streebog*, 2015», [4], авторы рассматривают хэш-функцию «Стрибог» с точки зрения возможности построения «алгоритма хэширования с закладкой», то есть возможности внедрения уязвимостей в алгоритм, которые позволят злоумышленнику «эффективно» находить коллизии
- В работе на основе метода «столкновения» (rebound attack) был представлен способ построения коллизии при условии «специального выбора констант»
- Способ выработки констант был опубликован в «В. И. Рудской, *Об алгоритме выработки констант функции хэширования «Стрибог»*, 2015» [41], что показывает невозможность применения предложенного метода построения коллизий

Оценки теоретической стойкости хэш-функции «Стрибог»

Теоретическая стойкость: от любви до ненависти и обратно?

- Теоретическая стойкость часто вызывает споры:
  - Она не всегда отражает реальную безопасность
  - Её оценки могут быть слишком абстрактными или идеализированными
  - Иногда кажется, что она «не помогает» в практическом плане
- Но... без неё зачастую невозможно (или очень сложно) обосновать базовые свойства криптосистем!

При обосновании теоретической стойкости конструкции Миягучи-Принеля в «J. Black и др., «An Analysis of the Blockcipher-Based Hash Functions from PGV», 2010» [12] вводится модель безопасности comp[2]

$$\mathsf{Adv}^{\mathrm{comp}}_{\mathsf{h}^{\mathcal{E}}}(\mathcal{A}) = \mathsf{P}\left(\mathcal{A}^{\mathcal{E},\mathcal{E}^{-1}} \to ((h,m),(h',m')) : \right.$$
$$\left.\big((h,m) \neq (h',m') \;\;\&\;\; \mathsf{h}^{\mathcal{E}}(h,m) = \mathsf{h}^{\mathcal{E}}(h',m')\big) \vee \mathsf{h}^{\mathcal{E}}(h,m) = h_0 \,\middle|\, \mathcal{E} \overset{\mathcal{U}}{\leftarrow} \mathbf{E}\right),$$

Для $\mathsf{h}^{\mathcal{E}}$ в модели comp оценка нестойкости определяется следующим образом:

$$\mathsf{Insec}^{\mathrm{comp}}_{\mathsf{h}^{\mathcal{E}}}(q) = \max_{\mathcal{A}(q')\colon q' \leq q} \mathsf{Adv}^{\mathrm{comp}}_{\mathsf{h}^{\mathcal{E}}}\left(\mathcal{A}(q')\right).$$

[2]Обоснование теоретической стойкости в [12] проводится в модели со случайным оракулом, когда используемый блочный шифр E заменяется на $\mathcal{E}$, выбранный случайно равновероятно из множества блочных шифров с соответсвующими длинами ключа, открытого текста и шифртекста $\mathbf{E}$.

- Нижняя оценка (для четных q, $q \leq 2^{(n+3)/2}$):

$$\frac{1}{4e} \cdot \frac{q(q-2)}{2^n} \leq \mathsf{Insec}^{\mathrm{Coll}}_{\mathsf{H}^{\mathcal{E}}}(q) \leq \mathsf{Insec}^{\mathrm{comp}}_{\mathsf{h}^{\mathcal{E}}}(q).$$

- Верхняя оценка (для любого $q \in \mathbb{N}$):

$$\mathsf{Insec}^{\mathrm{comp}}_{\mathsf{h}^{\mathcal{E}}}(q) \leq \frac{q(q+1)}{2^n}.$$

Для хэш-функций «Стрибог» косвенным образом применимы известные результаты для хэш-функций, построенных с использованием конструкции Меркла-Дамгорда [12][3]:

- Оценка трудоемкости построения коллизии для хэш-функции $\mathsf{H}^{\mathcal{E}}$ связана с трудоемкостью решения задачи comp для функции сжатия $\mathsf{h}^{\mathcal{E}}$:

$$\mathsf{Insec}^{\mathrm{Coll}}_{\mathsf{H}^{\mathcal{E}}}(q) \leq \mathsf{Insec}^{\mathrm{comp}}_{\mathsf{h}^{\mathcal{E}}}(q) \leq \frac{q(q+1)}{2^n}.$$

- Оценка трудоемкости построения прообраза:

$$\mathsf{Insec}^{\mathrm{Pre}}_{\mathsf{H}^{\mathcal{E}}}(q) \leq \mathsf{Insec}^{\mathrm{Pre}}_{\mathsf{h}^{\mathcal{E}}}(q) + 2^{-n} \leq \frac{q}{2^{n-1}}.$$

- Оценка трудоемкости построения второго прообраза:

$$\left|\mathsf{Insec}^{\mathrm{Pre}}_{\mathsf{H}^{\mathcal{E}}}(q) - \mathsf{Insec}^{\mathrm{Sec}^{l\cdot n}}_{\mathsf{H}^{\mathcal{E}}}(q)\right| \leq \frac{l}{2^{n-1}},$$

что указывает на близость этих задач при больших n.

[3]Все оценки верны при предположении, что используемый блочный шифр при фиксированном ключе является случайной подстановкой.

Среди результатов о теоретической стойкости хэш-функций «Стрибог» можно выделить следующее:

- Оценка теоретической стойкости в модели PrA (Pseudo-random Adversary Model) — модель с «экстрактором»
- Оценка теоретической стойкости относительно решения задачи отличения (Indifferentiability) — основанной на известной парадигме «идеальной функциональности»

В работе «Г. Седов, «Стойкость ГОСТ Р 34.11-2012 к атаке поиска прообраза и к атаке поиска коллизий», 2015» [43] нарушителю предоставляется доступ к так называемому «экстрактору» $\mathcal{Ex}$ — алгоритму, который для любого $C \in \mathbb{F}_2^n$ или выдает значение $M \in \mathbb{F}_2^\delta$, такое, что $\mathsf{H}^{\mathcal{E}}(M) = C$, или символ ошибки, если такого M не существует.

Задачей нарушителя является по выбранным им значениям $h_i \in \mathbb{F}_2^n$, $i \in \{1, 2, \ldots, q_e\}$, $q_e \in \mathbb{N}$, предъявить сообщение M', такое, что:

- экстрактор $\mathcal{Ex}$ получал в качестве аргументов все значения h_i и выдавал в качестве соответствующих ответов M_i, $i \in \{1, 2, \ldots, q_e\}$,
- $\mathsf{H}^{\mathcal{E}}(M') = h_i$ для некоторого $i \in \{1, 2, \ldots, q_e\}$ и $M' \notin \{M_1, M_2, \ldots, M_{q_e}\}$.

В [43] была получена следующая оценка стойкости параметризованной хэш-функции $\mathsf{H}^{\mathcal{E}}$ (названной в работе $\mathcal{GOST}$):

$$\mathsf{Insec}^{\mathrm{PrA}}_{\mathsf{H}^{\mathcal{E}}}(\mathcal{A}) \leq \frac{q_e(l+2)(q_p+l+2)}{2^n-(q_p+l+2)} + \frac{(q_p+l+2)(q_p+l+3)}{2(2^n-(q_p+l+2))},$$

где q_e — число запросов к экстрактору, q_p — число запросов к блочному шифру.

Из этой оценки следует, что преимущество противника «достаточно мало» как функция от n, а значит, семейство хэш-функций $\mathcal{GOST}$, рассматриваемое в работе, является стойким к атаке поиска прообраза, из чего следует и стойкость к атаке поиска коллизий.

Согласно «U. M. Maurer и др., «Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology», 2004» [26], хэш-функция $\mathsf{H}^{\mathcal{E}}$ называется $(t_D, q_H, q_E, \varepsilon)$-неотличимой от случайного оракула $\mathcal{H}$, если существует симулятор $\mathcal{S}$, такой, что выполняется:

$$\mathsf{Insec}^{\mathrm{PRO}}_{\mathsf{H}^{\mathcal{E}}}(t_D, q_H) = \max_{\mathcal{D}(t,q):\, t \le t_D, q \le q_H} \mathsf{Adv}^{\mathrm{PRO}}_{\mathsf{H}^{\mathcal{E}}}(\mathcal{D}(t,q)) \le \varepsilon,$$

где $\mathsf{Adv}^{\mathrm{PRO}}_{\mathsf{H}^{\mathcal{E}}}(\mathcal{D}) = \left|\mathsf{P}\left(\mathcal{D}^{\mathsf{H},\mathcal{E}} \to 1\right) - \mathsf{P}\left(\mathcal{D}^{\mathcal{H},\mathcal{S}} \to 1\right)\right|$.

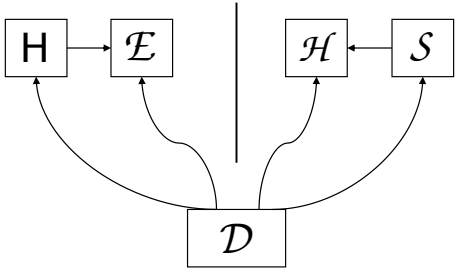


Рис.: Схематическое изображение эксперимента по отличению $\mathcal{D}$ хэш-функции $\mathsf{H}^{\mathcal{E}}$ от случайного оракула $\mathcal{H}$

В работе «L. R. Akhmetzyanova и др., *Streebog as a Random Oracle*, 2023» [2] был получен следующий результат:
Хэш-функция H с длиной хэш-значения $k \in \{256, 512\}$, представляющая собой параметризованную хэш-функцию «Стрибог», в которой параметром является используемый блочный шифр $\mathcal{E} : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$, является $(t_D, q_H, q_E, \varepsilon)$-неотличимой от случайного оракула для любого t_D, при

$$\varepsilon = \frac{(1 + l_m)q}{2^{n-4}} + \frac{(1 + n + l_m)q^2}{2^{n-7}}, \tag{1}$$

где $q = q_E + q_H(l_m + 2)$ и l_m — максимальная длина в блоках сообщения, подаваемого нарушителем на вход оракула хэширования.

- Нетрудно заметить, что в приведенной на предыдущем слайде формуле отсутствует зависимость от значения k. Возьмем $k = 256$ и $q = 2^{129}$, $l_m = 3$. Тогда нарушитель с вероятностью, близкой к 1, может решить задачу построения коллизии для хэш-функции с длиной хэш-значения 256 бит, однако значение ε равняется $5.7 \cdot 10^{-73}$.
- Иными словами, нарушитель $\mathcal{D}$ отличает хэш-функцию от случайного оракула «немногим лучше» случайного угадывания.

- Для корректного использования результатов работы «L. R. Akhmetzyanova и др., *Streebog as a Random Oracle*, 2023» [2] следует воспользоваться результатами работы «E. Andreeva и др., «Security Reductions of the Second Round SHA-3 Candidates», 2010» [8], в которой доказывается, что вероятность решения задачи ATK, $\mathrm{ATK} \in \{\mathrm{Coll}, \mathrm{Pre}, \mathrm{Sec}\}$, а также ряда других задач (построения мультиколлизии, мультипрообраза и др.) можно оценить сверху через вероятность решения соответствующей задачи для случайной функции и значение $\mathsf{Insec}^{\mathrm{PRO}}_{\mathsf{H}^{\mathcal{E}}}$:

$$\mathsf{Insec}^{\mathrm{ATK}}_{\mathsf{H}^{\mathcal{E}}}(q_H) \leq \mathsf{Insec}^{\mathrm{ATK}}_{\mathcal{H}}(q_H) + \mathsf{Insec}^{\mathrm{PRO}}_{\mathsf{H}^{\mathcal{E}}}(q_H).$$

- Иными словами, в работе [8] утверждается следующее: если $\mathcal{H}$ неотличима от $\mathcal{R}$, тогда в любом механизме, использующим $\mathcal{R}$ в качестве примитива, может быть осуществлена его замена на $\mathcal{H}$ без изменения величины теоретической стойкости механизма в целом.

В 2011 году последнее утверждение было частично опровергнуто в работе «T. Ristenpart и др., «Careful with Composition: Limitations of the Indifferentiability Framework», 2011» [30], в которой утверждается, что результаты работы «U. M. Maurer и др., «Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology», 2004» [26] не применимы для «любого механизма», и получение оценки теоретической стойкости с использованием такого подхода возможно только для так называемых «single-stage» нарушителей.

Как было замечено в работе «P. Rogaway, «Formalizing Human Ignorance», 2006» [31], формальное определение величины преобладания для оценки теоретической стойкости хэш-функции относительно решения задачи построения коллизии не имеет содержательного смысла:

$$\mathsf{Adv}_{\mathsf{H}}^{\mathrm{Coll}} = 1,$$

что парадоксально, так как исходя из «бытовых представлений» нарушитель должен произвести какие-то вычисления. Однако:

- Коллизия для криптографической функции хэширования всегда существует
- Наилучший алгоритм просто выводит пару значений (M, M'), $M \neq M'$, таких, что $\mathsf{H}(M) = \mathsf{H}(M')$, без дополнительных вычислений

При обосновании криптографических свойств проектов национальных стандартов иногда происходит сведение к задаче построения коллизии. Это может быть некорректно.

Согласно «P. Rogaway, «Formalizing Human Ignorance», 2006» [31] можно использовать сведение в случае, когда доказывается: *"Из эффективного алгоритма решения задачи* XXX *для протокола* П *конструктивно строится эффективный алгоритм нахождения коллизии."*

- Обоснование в модели ROM (сильные ограничения о возможностях нарушителя).
- Использование режимов работы хэш-функции (в частности, использование ключевых хэш-функций, например, «V. Kiryukhin, «Streebog compression function as PRF in secret-key settings», 2022, V. Kiryukhin, «Keyed Streebog is a secure PRF and MAC», 2023» [22, 20]).

Оценки практической стойкости хэш-функции «Стрибог»


Нет, доказуемая стойкость все доказывает!
Доказуемой стойкости не достаточно
Доказуемой стойкости не достаточно
34%
34%
68%
14%
14%
95%
0.1%
2%
2%
0.1%
IQ score
55
70
85
100
115
130
145

Методы и подходы к криптографическому анализу, применяемые авторами работ при получении оценок практической стойкости для функции сжатия:

- m-collision — построение мультиколлизии;
- ImpDiff — метод невозможных разностей;
- IntegralPrp — исследование интегральных характеристик;
- MILP — использование автоматических средств оптимизации (т.н. MILP-решателей от англ. Mixed-Integer Linear Programing);
- MitM — метод согласования;
- MultiDiff — многомерный разностный метод;
- Rebound — метод столкновения;
- RelatedKey — использование связанных ключей;
- SIM — использование того факта, что алгоритм развертывания ключа в блочном шифре хэш-функции «Стрибог» аналогичен самому блочному шифру (structural similarities);
- TruncDiff — метод усеченных разностей.

Построение прообраза для функции сжатия в «Стрибог» с длиной выхода 512 бит

Трудоемкость построения прообраза для идеального примитива: 2^{512}.

Раунды	Тр-ть	Память	Исп.	Год	Ист.	Прим.
5	2^{192}	2^{64}	MitM	2014	[24]	
	2^{208}	2^{12}	MitM	2015	[24]	
	2^{448}	2^{64}	MitM	2015	[6]	Псевдо-прообраз
6	2^{496}	2^{64}	MitM	2014	[35]	
	2^{496}	2^{112}	MitM	2014	[6]	Псевдо-прообраз
6.5	2^{232}	2^{120}	MitM	2015	[24]	
	2^{209}	2^{160}	MitM+MILP	2018	[18]	
6.75	$2^{399.5}$	2^{349}	ImpDiff	2014	[1]	Восст. неизв. h, данные: 2^{483}
	$2^{261.5}$	2^{205}	ImpDiff	2014	[1]	Восст. неизв. h, данные: $2^{495.5}$
7	2^{421}	2^{354}	MultiDiff	2022	[22]	Восст. неизв. h, данные: 2^{64}
	2^{240}	2^{20}	TruncDiff	2022	[22]	Восст. неизв. h, данные: 2^{113}
7.5	2^{496}	2^{64}	MitM	2015	[24]	
	2^{441}	2^{192}	MitM+MILP	2022	[18]	
	2^{433}	2^{177}	SIM+MitM	2024	[15]	
8.5	2^{481}	2^{288}	MitM+MILP	2022	[18]	
	2^{481}	2^{129}	SIM+MitM	2024	[15]	
10	2^{224}	2^{94}	TruncDiff+ RelatedKey	2022	[21]	Восст. неизв. m, данные: 2^{225}, необх.: 2^{198} связ-х ключей
	2^{232}	2^{91}	TruncDiff+ RelatedKey	2022	[21]	Восст. неизв. m, данные: 2^{168}, необх.: 2^{145} связ-х ключей
11	2^{224}	2^{68}	TruncDiff+ RelatedKey	2022	[21]	Восст. неизв. m, данные: 2^{225}, необх.: 2^{224} связ-х ключей

Трудоемкость построения прообраза для идеального примитива: 2^{512}.

Раунды	**Тр-ть**	**Память**	**Исп.**	**Год**	**Ист.**
5	2^{481}	2^{256}	MitM (х.ф.)+MitM	2014	[6]
6	2^{505}	2^{64}	MitM+m-collision	2014	[35]
	2^{505}	2^{256}	MitM (х.ф.)+MitM	2014	[6]
	2^{496}	2^{64}	MitM+m-collision	2014	[25]
	2^{504}	2^{11}	MitM+m-collision	2014	[25]
7.5	2^{496}	2^{64}	MitM+m-collision	2015	[24]
	2^{504}	2^{11}	MitM+m-collision	2015	[24]
	$2^{478.25}$	2^{256}	MitM+MILP	2022	[18]
	$2^{474.25}$	2^{256}	MitM+m-collision	2024	[15]
8.5	$2^{498.25}$	2^{288}	MitM+MILP	2022	[18]
	$2^{498.25}$	2^{256}	MitM+m-collision	2024	[15]

Трудоемкость построения прообраза для идеального примитива: 2^{256}.

Цель	Раунды	Тр-ть	Память	Исп.	Год	Ист.
Функция сжатия	6	2^{240}	2^{64}	MitM	2014	[35]
	6.5	2^{232}	2^{120}	MitM+MILP	2015	[24]
	7.5	2^{209}	2^{192}	MitM+MILP	2022	[18]
Хэш-функция	5	2^{192}	2^{64}	MitM+MILP	2015	[24]
		2^{208}	2^{12}	MitM+MILP	2015	[24]
	6.5	2^{232}	2^{120}	MitM+MILP	2015	[24]
		2^{209}	2^{160}	MitM+MILP+ m-collision	2022	[18]

Трудоемкость построения коллизии для идеального примитива: $\sqrt{\pi/2} \cdot 2^{256}$.

Цель	Раунды	Тр-ть	Память	Исп.	Год	Ист.	Прим.
Функция сжатия	4	$2^{19.8}$	2^{16}	Rebound	2014	[23]	псевдо-коллизия
	4.5	2^{64}	2^{16}	Rebound	2013	[34]	псевдо-коллизия
		$2^{19.8}$	2^{16}	Rebound	2015	[23]	псевдо-коллизия
	4.75	2^{8}	—	Rebound	2014	[3]	псевдо-коллизия
	5	2^{120}	2^{64}	Rebound	2014	[35]	
	5.5	2^{64}	2^{64}	Rebound	2013	[34]	псевдо-коллизия
	6.5	2^{64}	2^{16}	Rebound	2015	[23]	псевдо-коллизия
	7.5	2^{128}	2^{16}	Rebound	2013	[34]	
		2^{176}	2^{64}	Rebound	2014	[25]	
	7.75	2^{184}	2^{8}	Rebound	2013	[3]	псевдо-коллизия
		2^{72}	2^{8}	Rebound	2013	[3]	псевдо-почти-коллизия
	8.75	2^{128}	2^{8}	Rebound	2013	[3]	псевдо-почти-коллизия
	9.5	2^{176}	2^{128}	Rebound	2013	[34]	
		2^{240}	2^{16}	Rebound	2013	[34]	
		2^{176}	2^{64}	Rebound	2014	[25]	
	9.75	2^{184}	2^{8}	Rebound	2013	[3]	псевдо-почти-коллизия

Трудоемкость построения коллизии для идеального примитива: $\sqrt{\pi/2} \cdot 2^{256}$.

Цель	Раунды	Тр-ть	Память	Исп.	Год	Ист.	Прим.
Блочный шифр	5	2^8	2^8	Rebound	2013	[3]	условно-свободная коллизия
	8	2^{64}	2^8	Rebound	2013	[3]	условно-свободная коллизия
Хэш-функция	5	2^{122}	2^{64}	Rebound+m-collision	2014	[35]	
	7.5	2^{181}	2^{64}	Rebound+m-collision	2014	[25]	

Трудоемкость построения коллизии для идеального примитива: $\sqrt{\pi/2} \cdot 2^{128}$.

Цель	Раунды	Тр-ть	Память	Исп.	Год	Ист.
Идеальный примитив		$1.2 \cdot 2^{128}$	const $\cdot$ 2^{256}	—	—	—
		$2.4 \cdot 2^{128}$	const	—	—	[25]
Функция сжатия	5	2^{122}	2^{64}	Rebound	2013	[25]
		2^{120}	2^{64}	Rebound	2014	[35]
		2^{120}	2^{64}	Rebound	2014	[35]
	6.5	2^{120}	2^{64}	Rebound	2014	[25]
Хэш-функция	5	2^{122}	2^{64}	Rebound+m-collision	2014	[35]
	6.5	2^{125}	2^{64}	Rebound+m-collision	2014	[25]

Цель	Раунды	Тр-ть	Память	Исп.	Год	Ист.	Прим.
Подстановка	6.5	2^{64}	—	IntegralPrp	2013	[7]	Данные: 2^{64}
	7.5	2^{120}	—	IntegralPrp	2013	[7]	Данные: 2^{120}
	8.75	2^{128}	2^{7}	IntegralPrp	2019	[33]	—
	12	$2^{192.4}$	2^{129}	IntegralPrp	2019	[33]	—
Функция сжатия	6	2^{64}	—	IntegralPrp	2013	[7]	Данные: 2^{64}
		2^{56}	—	IntegralPrp	2018	[32]	Данные: 2^{56}
		2^{16}	—	IntegralPrp	2018	[32]	Данные: 2^{16}
	7	2^{120}	—	IntegralPrp	2013	[7]	Данные: 2^{120}
		2^{104}	—	IntegralPrp	2018	[32]	Данные: 2^{104}
		2^{64}	—	IntegralPrp	2018	[32]	Данные: 2^{64}
	8	2^{392}	—	IntegralPrp	2018	[32]	Данные: 2^{392}
		2^{112}	—	IntegralPrp	2018	[32]	Данные: 2^{112}
	8.75	2^{128}	2^{7}	IntegralPrp	2019	[33]	—
	9	2^{480}	—	IntegralPrp	2018	[32]	Данные: 2^{480}
		2^{440}	—	IntegralPrp	2018	[32]	Данные: 2^{440}
	9.5	2^{441}	2^{136}	Rebound	2014	[25]	—
	10	2^{176}	2^{16}	Rebound	2014	[34]	—
		2^{128}	2^{128}	Rebound	2014	[34]	—
		2^{508}	—	IntegralPrp	2018	[32]	Данные: 2^{508}
	12	2^{129}	2^{71}	IntegralPrp	2019	[33]	—

- В настоящее время известна единственная работа, посвященная анализу криптографических свойств полнораундовой функции сжатия и полнораундового блочного шифра «C. Tingting и др., «Distinguisher on full-round compression function of GOST R», 2020» [33].
- В данной работе используется метод интегрального криптоанализа с известным ключом[4], в котором противнику известен ключ, структура всех преобразований хэш-функции, а соответственно он имеет возможность вычислять промежуточные состояния хэш-функции.
- При таких условиях метод, предложенный в работе [33] – не более чем описание алгебраических свойств хэш-функции.
- Может влиять на корректность получения некоторых оценок теоретической стойкости. Например, в модели, рассмотренной в «L. R. Akhmetzyanova и др., *Streebog as a Random Oracle*, 2023», [2], возможно отличить реальный блочный шифр от выхода симулятора.

[4]от англ. «known-key integral cryptoanalysis»

- Следует отметить, что **все перечисленные методы**, применяемые авторами работ, указанных в таблицах, **на сегодняшний день применимы лишь к ограниченному числу раундов функции сжатия / хэш-функций «Стрибог»** или требуют больших объёмов вычислений для полного 12-раундового алгоритма (см. [33]).
- Таким образом, существующие исследования показывают, что применение методов криптографического анализа целой хэш-функции на основе «неидеальности» используемых примитивов (блочного шифра и функции сжатия) оказывается невозможным.
- Известны результаты, связанные с использованием «эквивалентного представления функции сжатия».

Методы криптографического анализа хэш-функций, построенных на основе конструкции Меркла-Дамгорда, напрямую неприменимы к хэш-функции «Стрибог», однако использование эквивалентного представления функции сжатия позволило адаптировать известные методы построения второго прообраза для хэш-функции «Стрибог», «J. Guo и др., *The Usage of Counter Revisited: Second-Preimage Attack on New Russian Standardized Hash Function*, 2014» [17].

- Построение второго прообраза с использованием древовидной коллизии и 2^n-мультиколлизии
- Построение второго прообраза с использованием модифицированного метода расширяемых сообщений

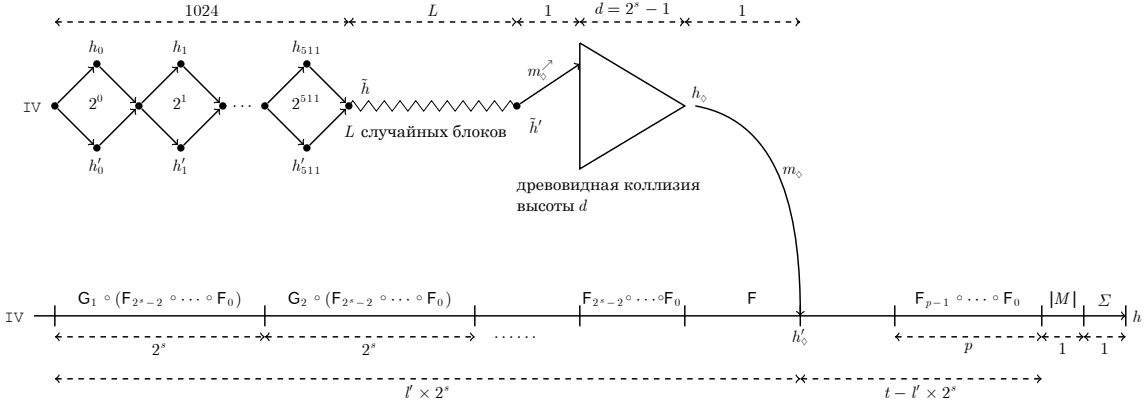


Рис.: Схематическое изображение алгоритма построения второго прообраза с использоваем древовидной коллизии и 2^n-мультиколлизии

Суммарная трудоемкость может быть оценена величиной (d, l — параметры)

$$\sum_{t=0}^{d-1} 2^{(n+d-t)/2} + 2n\sqrt{\pi 2^n} + 2^n/l + 2^{n-d} \approx 3.41 \cdot 2^{(n+d)/2} + 2^{267} + 2^n/l + 2^{n-d}$$

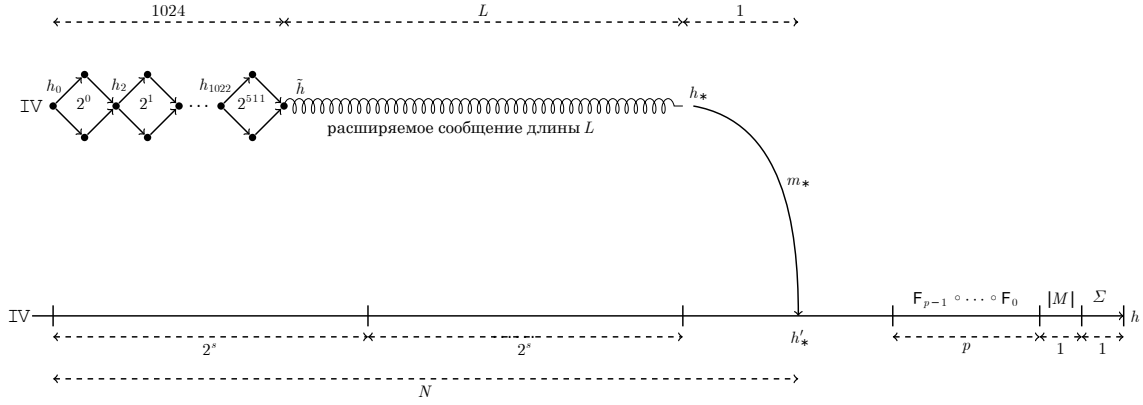


Рис.: Схематическое изображение алгоритма построения второго прообраза с использоваем модифицированного метода расширяемых сообщений

Суммарная трудоемкость может быть оценена величиной (s, t, l — параметры)

$$2 \cdot \sqrt{\pi 2^n}(t + 1 - s + n) + 2^{t+1} + 2^n/l$$

Заметим, что в случае более «коротких» сообщений, мажорирующим слагаемым будет являться величина $2^n/k$, где k — количество блоков в исходном сообщении, которая будет являться хотя и достаточно грубой, но все же **приемлемой** для практических приложений оценкой стойкости относительно решения задачи построения второго прообраза.

Оценки практической стойкости различных конструкций хэш-функций.

Задача	**Идеальный примитив**	**Конструкция Меркла-Дамгорда**	**Конструкция HAIFA**	**«Стрибог» (512)**[5]
Построение прообраза	2^n	$2 \cdot 2^n$	$2 \cdot 2^n$	$3 \cdot 2^n$
Построение прообраза для одного из k' сообщений	$2^n/k'$	$2 \cdot 2^n/k'$	$2 \cdot 2^n/k'$	$3 \cdot 2^n/k'$
Построение второго прообраза для сообщения из k блоков	2^n	$\max\left\{2^n/k, 2^{n/2}\sqrt{\pi/2}\right\}$	2^n	$\max\left\{2^n/k, 2^{n/2}\sqrt{\pi/2}\right\}$
Построение второго прообраза для одного из k' сообщений, каждое длины k блоков	$2^n/k'$	$\max\left\{2^n/(k \cdot k'), 2^{n/2}\sqrt{\pi/2}\right\}$	$2^n/k'$	$\max\left\{2^n/(k \cdot k'), 2^{n/2}\sqrt{\pi/2}\right\}$
Построение коллизии	$\sqrt{\pi/2} \cdot 2^{n/2}$	$\sqrt{\pi/2} \cdot 2^{n/2}$	$\sqrt{\pi/2} \cdot 2^{n/2}$	$2 \cdot \sqrt{\pi} \cdot 2^{n/2}$
Построение k-коллизии	$k/e \cdot 2^{n(k-1)/k}$	$\sqrt{\pi/2} \cdot \lceil \log_2(k) \cdot 2^{n/2} \rceil$	$\sqrt{\pi/2} \cdot \lceil \log_2(k) \cdot 2^{n/2} \rceil$	$\sqrt{\pi/2} \cdot \lceil \log_2(k) \cdot 2^{n/2} \rceil$

[5] Для второго прообраза — оценки снизу, для конструкций, отличных от идеального примитива — в количествах вычислений сжимающего отображения

Для хэш-функции с длиной выхода $m = 256$ (в вычислениях функции сжатия):

- Построение прообраза: $3 \cdot 2^m$
- Построение прообраза для одного из k' сообщений: $3 \cdot 2^m / k'$
- Построение второго прообраза для сообщения из k блоков: $3 \cdot 2^m$
- Построение второго прообраза для одного из k' сообщений, каждое длины k блоков: $3 \cdot 2^m / k'$
- Построение коллизии: $3 \cdot \sqrt{\pi/2} \cdot 2^{m/2}$
- Построение k-коллизии: $\min\{3 \cdot k/e \cdot 2^{m(k-1)/k}, \sqrt{\pi/2} \cdot \lceil \log_2(k) \cdot 2^{n/2} \rceil\}$

В работе [9] показывается, что методы, применимые для нарушителя, не имеющего доступа к квантовому вычислителю, тривиальным образом переносятся на случай нарушителя, имеющего такой доступ

Цель	Свойство	Сценарий $\mathcal{R}_1$		Сценарий $\mathcal{R}_2$		
		Время	Память (квантовая)	Время	Память (квантовая)	Память (классич.)
Конструкция Меркла-Дамгорда, «Стрибог»-512	Коллизия	$2^{n/3}$	$2^{n/3}$	$2^{2n/5}$	$O(n)$	$2^{n/5}$
	Прообраз	$2^{n/2}$	$O(n)$	$2^{n/2}$	$O(n)$	$O(1)$
	2-й прообраз для сообщения из 2^k блоков[6]	$\max\{2^{n/3}, 2^{(n-k)/2}\}$	$2^{n/3}$	$\max\{2^{2n/5}, 2^{(n-k)/2}\}$	$O(n)$	$2^{3n/7}$
Произвольное сжимающее отображение, «Стрибог»-256	Коллизия	$2^{n/3}$	$2^{n/3}$	$2^{2n/5}$	$O(n)$	$2^{2n/5}$
	Прообраз	$2^{n/2}$	$O(n)$	$2^{n/2}$	$O(n)$	$O(1)$
	2-й прообраз	$2^{n/2}$	$O(n)$	$2^{n/2}$	$O(n)$	$O(1)$

[6]Оценки снизу

- За рамками рассмотрения остались работы, посвященные криптографическому анализу механизмов, в которых хэш-функция «Стрибог» используется в качестве структурного элемента.
- Методы криптографического анализа хэш-функции «Стрибог», использующие информацию из побочных каналов утечки также остались за рамками данной работы (см., например, «R. AlTawy и др., «Differential Fault Analysis of Streebog», 2015» [5]).
- Однако, изучение данных направлений также является интересным и перспективным как с научной точки зрения, так и с точки зрения оценки безопасности современных информационных систем, использующих хэш-функцию «Стрибог».

О реализациях хэш-функции «Стрибог»

- Существует множество программных и аппаратных реализаций хэш-функции «Стрибог» ГОСТ Р 34.11-2012. Можно выделить следующие работы:
  - «I. Gafurov, «Высокоскоростная программная реализация алгоритма хэширования ‘Стрибог’», 2023» [16]
  - «А. Казимиров и др., «О создании эффективных программных реализаций отечественных криптографических стандартов», 2013» [39]
  - «П. Лебедев, «Сравнение старого и нового стандартов РФ на криптографическую хэш-функцию на ЦП и графических процессорах NVIDIA», 2013» [40]
  - «M. Borodin и др., «High-Speed Software Implementation of the Prospective 128-bit Block Cipher and Streebog Hash-Function», 2014» [14]
  - «В. Архипкин и др., «Устройство для реализации алгоритма шифрования "Кузнечик"и хэш-функции "Стрибог"», 2021» [37]
  - «А. Боршевников, «Об оптимизации скорости расчета криптографических хэш-функций ГОСТ Р 34.11-94 и ГОСТ Р 34.11-2012», 2019» [38]
  - «Ю. Северин и др., «Низкоресурсная оптимизация производительности на примере функции хэширования по ГОСТ Р 34.11-2012», 2017» [42]

- Зачастую многие реализации хэш-функции «Стрибог» содержат ошибки, приводящие к некорректной работе алгоритма
- Использование некорректных реализаций приводит к возникновению практически реализуемых уязвимостей (например, возможности построения коллизии).

Известные на настоящий момент результаты исследований позволяют утверждать, что стандартизированная в Российской Федерации хэш-функция удовлетворяет всем предъявляемым к ней требованиям.

[1] A. Abdelkhalek, R. AlTawy и A. M. Youssef. «Impossible Differential Properties of Reduced Round Streebog». В: *Codes, Cryptology, and Information Security - C2SI 2015*. Т. 9084. Lecture Notes in Computer Science. Springer, 2015, с. 274—286. URL: https://doi.org/10.1007/978-3-319-18681-8_22.

[2] Liliya R. Akhmetzyanova, Alexandra Babueva и Andrey Bozhko. *Streebog as a Random Oracle*. 2023. URL: https://eprint.iacr.org/2023/1075.

[3] R. AlTawy, A. Kircanski и A. M. Youssef. «Rebound Attacks on Stribog». В: *Information Security and Cryptology - ICISC 2013*. Т. 8565. Lecture Notes in Computer Science. Springer, 2013, с. 175—188. URL: https://doi.org/10.1007/978-3-319-12160-4_11.

[4] R. AlTawy и A. Youssef. *Watch your constants: malicious Streebog*. 2015. DOI: 10.1049/iet-ifs.2014.0540.

[5] Riham AlTawy и Amr M. Youssef. «Differential Fault Analysis of Streebog». В: *Information Security Practice and Experience - 11th International Conference, ISPEC 2015, Beijing, China, May 5-8, 2015. Proceedings*. Под ред. Javier López и Yongdong Wu. T. 9065. Lecture Notes in Computer Science. Springer, 2015, с. 35—49. DOI: 10.1007/978-3-319-17533-1_3. URL: https://doi.org/10.1007/978-3-319-17533-1%5C_3.

[6] Riham AlTawy и Amr M. Youssef. *Preimage attacks on Reduced-round Stribog*. Cryptology ePrint Archive, Paper 2014/319. https://eprint.iacr.org/2014/319. 2014. URL: https://eprint.iacr.org/2014/319.

[7] Riham AlTawy и Amr M. Youssef. «Integral Distinguishers for Reduced-round Stribog». В: *IACR Cryptol. ePrint Arch.* (2013), с. 648. URL: http://eprint.iacr.org/2013/648.

[8] E. Andreeva, B. Mennink и B. Preneel. «Security Reductions of the Second Round SHA-3 Candidates». В: *Information Security - ISC 2010*. Т. 6531. Lecture Notes in Computer Science. Springer, 2010, с. 39—53. URL: https://doi.org/10.1007/978-3-642-18178-8_5.

[9] Zhenzhen Bao и др. *Evaluating the Security of Merkle-Damgård Hash Functions and Combiners in Quantum Settings*. Cryptology ePrint Archive, Paper 2022/1058. 2022. DOI: 10.1007/978-3-031-23020-2_39. URL: https://eprint.iacr.org/2022/1058.

[10] Christof Beierle и др. *Decomposing Linear Layers*. Cryptology ePrint Archive, Paper 2022/1159. 2022. DOI: 10.46586/tosc.v2022.i4.243-265. URL: https://eprint.iacr.org/2022/1159.

[11] A. Biryukov, L. Perrin и A. Udovenko. «Reverse-Engineering the S-Box of Streebog, Kuznyechik and STRIBOBr1». В: *EUROCRYPT 2016*. Springer, 2016, с. 372—402.

[12] J. Black, P. Rogaway и T. Shrimpton. «An Analysis of the Blockcipher-Based Hash Functions from PGV». В: *J. Cryptol.* 23.4 (2010), с. 519—545. URL: https://doi.org/10.1007/S00145-010-9071-0.

[13] N.P. Borisenko, D.A. Vasinev и Khoang Dyk Tkho. «Method of forming S-blocks with minimum number of logic elements (RU 2572423 C2)». RU 2572423 C2. 2016. URL: https://patents.google.com/patent/RU2572423C2/ru.

[14] M. Borodin, A. Rybkin и A. Urivskiy. «High-Speed Software Implementation of the Prospective 128-bit Block Cipher and Streebog Hash-Function». В: *Proceedings of the Conference on Cryptographic Algorithms and Security Systems*. 2014.

[15] Shiyao Chen и др. «Diving Deep into the Preimage Security of AES-like Hashing». В: *IACR Cryptol. ePrint Arch.* (2024), с. 300. URL: https://eprint.iacr.org/2024/300.

[16] I.R. Gafurov. «Высокоскоростная программная реализация алгоритма хэширования ‘Стрибог’». В: *Ulyanovsk State Univ. Journal* (2023).

[17] Jian Guo и др. *The Usage of Counter Revisited: Second-Preimage Attack on New Russian Standardized Hash Function*. 2014. URL: https://api.semanticscholar.org/CorpusID:9933436.

[18] Jialiang Hua и др. «Improved MITM Cryptanalysis on Streebog». В: *IACR Cryptol. ePrint Arch.* (2022), с. 568. URL: https://eprint.iacr.org/2022/568.

[19] Oleksandr Kazymyrov и Valentyna Kazymyrova. *Algebraic Aspects of the Russian Hash Standard GOST R 34.11-2012*. Cryptology ePrint Archive, Paper 2013/556. 2013. URL: https://eprint.iacr.org/2013/556.

[20] V. A. Kiryukhin. «Keyed Streebog is a secure PRF and MAC». В: *Матем. вопр. криптогр.* 14.2 (2023), с. 77—96.

[21] V. A. Kiryukhin. «Related-key attacks on the compression function of Streebog». В: *Матем. вопр. криптогр.* 14.2 (2023), с. 59—76.

[22] V. A. Kiryukhin. «Streebog compression function as PRF in secret-key settings». В: *Матем. вопр. криптогр.* 13.2 (2022), с. 99—116.

[23] S. Kölbl и C. Rechberger. «Practical Attacks on AES-like Cryptographic Hash Functions». В: *Progress in Cryptology - LATINCRYPT 2014*. Т. 8895. Lecture Notes in Computer Science. Springer, 2014, с. 259—273. URL: https://doi.org/10.1007/978-3-319-16295-9%5C_14.

[24] B. Ma и др. «Improved (Pseudo) Preimage Attacks on Reduced-Round GOST and Grøstl-256 and Studies on Several Truncation Patterns for AES-like Compression Functions». В: *Advances in Information and Computer Security - 10th Int. Workshop on Security, IWSEC 2015, Nara, Japan*. Т. 9241. Lecture Notes in Computer Science. Springer, 2015, с. 79—96. URL: https://doi.org/10.1007/978-3-319-22425-1_6.

[25] B. Ma и др. «Improved Cryptanalysis on Reduced-Round GOST and Whirlpool Hash Function». В: *Applied Cryptography and Network Security - 12th Int. Conf., ACNS 2014, Lausanne, Switzerland*. Т. 8479. Lecture Notes in Computer Science. Springer, 2014, с. 289—307. URL: https://doi.org/10.1007/978-3-319-07536-5_18.

[26] U. M. Maurer, R. Renner и C. Holenstein. «Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology». В: *Theory of Cryptography - TCC 2004*. Т. 2951. Lecture Notes in Computer Science. Springer, 2004, с. 21—39. URL: https://doi.org/10.1007/978-3-540-24638-1%5C_2.

[27] Léo Perrin. *Partitions in the S-Box of Streebog and Kuznyechik*. Cryptology ePrint Archive, Paper 2019/092. 2019. URL: https://eprint.iacr.org/2019/092.

[28] Léo Perrin и Aleksei Udovenko. *Exponential S-Boxes: a Link Between the S-Boxes of BelT and Kuznyechik/Streebog*. 2017. URL: https://api.semanticscholar.org/CorpusID:21879521.

[29] O. C. Puente, R. F. Leal и R. A. de la Cruz R. A. de la Cruz Jiménez. *On the Bit-Slice representations of some nonlinear bijective transformations*. 2024.

[30] T. Ristenpart, H. Shacham и T. Shrimpton. «Careful with Composition: Limitations of the Indifferentiability Framework». В: *Advances in Cryptology - EUROCRYPT 2011, Tallinn, Estonia*. Т. 6632. Lecture Notes in Computer Science. Springer, 2011, с. 487—506. URL: https://doi.org/10.1007/978-3-642-20465-4_27.

[31] P. Rogaway. «Formalizing Human Ignorance». В: *Progress in Cryptology - VIETCRYPT 2006, Hanoi, Vietnam*. Т. 4341. Lecture Notes in Computer Science. Springer, 2006, с. 211—228. URL: https://doi.org/10.1007/11958239_14.

[32] L. Rongjia, J. Chenhui и F. Ruya. «Improved Integral Distinguishers on Compression Function of GOST R Hash Function». В: *Comput. J.* 62.4 (2019), с. 535—544. URL: https://doi.org/10.1093/comjnl/bxy123.

[33] Cui Tingting, Wang Wei и Wang Meiqin. «Distinguisher on full-round compression function of GOST R». В: *Inf. Process. Lett.* 156 (2020), с. 105902. URL: https://doi.org/10.1016/j.ipl.2019.105902.

[34] Z. Wang, H. Yu и X. Wang. «Cryptanalysis of GOST R hash function». В: *Inf. Process. Lett.* 114.12 (2014), с. 655—662. URL: https://doi.org/10.1016/j.ipl.2014.07.007.

[35] J. Zou, W. Wu и S. Wu. «Cryptanalysis of the Round-Reduced GOST Hash Function». В: *Information Security and Cryptology - Inscrypt 2013*. Т. 8567. Lecture Notes in Computer Science. Springer, 2013, с. 309—322. URL: https://doi.org/10.1007/978-3-319-12087-4%5C_20.

[36] О. Д. Авраамова и др. «A compact bit-sliced representation of Kuznyechik S-box». В: *Mathematical Aspects of Cryptography* (2021). URL: https://api.semanticscholar.org/CorpusID:238024730.

[37] В.Я. Архипкин, В.В. Ерохин и А.В. Иванов. «Устройство для реализации алгоритма шифрования "Кузнечик"и хэш-функции "Стрибог"». RU 2743412 C1. 2021. URL: https://elibrary.ru/item.asp?id=44759497.

[38] А.Е. Боршевников. «Об оптимизации скорости расчета криптографических хэш-функций ГОСТ Р 34.11-94 и ГОСТ Р 34.11-2012». В: (2019), с. 76—78.

[39] А. Казимиров и С. Смышляев. «О создании эффективных программных реализаций отечественных криптографических стандартов». В: *Рускрипто*. [online]. 2013. URL: http://www.ruscrypto.ru/resource/summary/rc2013/ruscrypto_2013_028.zip.

[40] П.А. Лебедев. «Сравнение старого и нового стандартов РФ на криптографическую хэш-функцию на ЦП и графических процессорах NVIDIA». В: *Математические вопросы криптографии* 4.2 (2013), с. 73—80.

[41] Владимир Игоревич Рудской. *Об алгоритме выработки констант функции хэширования «Стрибог»*. 2015.

[42] Ю.В. Северин и Ю.В. Гольчевский. «Низкоресурсная оптимизация производительности на примере функции хэширования по ГОСТ Р 34.11-2012». В: *Системный администратор* 1.2 (2017), с. 170—171.

[43] Г. К. Седов. «Стойкость ГОСТ Р 34.11-2012 к атаке поиска прообраза и к атаке поиска коллизий». В: *Математические вопросы криптографии* 6.2 (2015), с. 79—98.

[44] Д. И. Трифонов и Д. Б. Фомин. *Computational work for some TU-based permutations*. 2024.