НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
май–июнь 2025 Том 25 № 3 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
May–June 2025 Vol. 25 No 3 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)

НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

doi: 10.17586/2226-1494-2025-25-3-428-437
УДК 004.056

Анализ криптографической стойкости хеш-функции SHA-256 при помощи SAT-подхода

Вадим Валерьевич Давыдов[1]✉, Михаил Денисович Пихтовников[2], Анастасия Павловна Кирьянова[3], Олег Сергеевич Заикин[4]

[1] ООО «КуАпп», Москва, 121205, Российская Федерация
[1] Санкт-Петербургский государственный университет аэрокосмического приборостроения, 190000, Санкт-Петербург, Российская Федерация
[1,3] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
[2] Южный федеральный университет, Таганрог, 347922, Российская Федерация
[4] Новосибирский государственный университет, Новосибирск, 630090, Российская Федерация
[4] Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук, Иркутск, 664033, Российская Федерация

[1] vadimdavydov@outlook.com✉, https://orcid.org/0000-0002-5544-2434
[2] pikhtovnikov347@mail.ru, https://orcid.org/0009-0000-2173-0844
[3] anastaciakosanovskaya@gmail.com, https://orcid.org/0009-0006-0344-5111
[4] oleg.zaikin@icc.ru, https://orcid.org/0000-0002-0145-5010

Аннотация

Введение. В современных системах обеспечения информационной безопасности криптографические хеш-функции играют значительную роль и выполняют такие важные задачи, как обеспечение целостности данных и их эффективное сжатие. Одной из наиболее значимых и широко применяемых криптографических хеш-функций является SHA-256 из семейства SHA-2. Исследование криптографической стойкости SHA-256 является актуальной научной задачей и решается с применением современных подходов криптоанализа к атакам нахождения прообразов и коллизий с акцентом на практическую осуществимость таких атак. **Метод.** В представленной работе для поиска прообразов неполнораундовых версий функции сжатия SHA-256 применен логический криптоанализ, согласно которому исходная задача криптоанализа сводится к проблеме булевой выполнимости (SAT). Для поиска коллизий совместно применены логический и дифференциальный криптоанализы. **Основные результаты.** Выполнено сравнение эффективности различных способов сведения функции сжатия SHA-256 к SAT. Впервые найдены прообразы для 17- и 18-раундовых функций сжатия SHA-256, а также прообразы для ослабленной 19-раундовой функции сжатия. Построены базовые дифференциальные пути, с помощью которых быстрее найдены коллизии 18-раундовой функции сжатия. В результате сведения к SAT известных дифференциальных путей найдены коллизии 19-раундовой функции сжатия. **Обсуждение.** Продемонстрирована возможность комбинирования двух методов криптоанализа с целью повышения эффективности анализа криптографических алгоритмов. Результаты исследования подтвердили, что полнораундовая хеш-функция SHA-256 остается устойчивой к атакам, направленным на нахождение прообразов и коллизий, в рамках примененного SAT-подхода.

Ключевые слова

криптографическая хеш-функция, SHA-256, SAT, логический криптоанализ, дифференциальный криптоанализ

Благодарности

Работа Вадима Валерьевича Давыдова и Анастасии Павловны Кирьяновой выполнена в рамках государственного задания (проект FSER-2025-0003). Олег Сергеевич Заикин выполнил свою часть работы при поддержке Математического центра в Академгородке, соглашение с Министерством науки и высшего образования Российской Федерации № 075-15-2025-349. Работа является расширенной версией результатов, полученных в рамках летней школы-конференции «Криптография и информационная безопасность» в 2024 году.

Ссылка для цитирования: Давыдов В.В., Пихтовников М.Д., Кирьянова А.П., Заикин О.С. Анализ криптографической стойкости хеш-функции SHA-256 при помощи SAT-подхода // Научно-технический вестник информационных технологий, механики и оптики. 2025. Т. 25, № 3. С. 428–437. doi: 10.17586/2226-1494-2025-25-3-428-437

Analysis of the cryptographic strength of the SHA-256 hash function using the SAT approach

Vadim V. Davydov[1]✉, Michail D. Pikhtovnikov[2], Anastasia P. Kiryanova[3], Oleg S. Zaikin[4]

[1] QApp, Moscow, 121205, Russian Federation
[1] Saint Petersburg State University of Aerospace Instrumentation (SUAI), 190000, Saint Petersburg, Russian Federation
[1,3] ITMO University, Saint Petersburg, 197101, Russian Federation
[2] Southern Federal University, Taganrog, 347922, Russian Federation
[4] Novosibirsk State University, Novosibirsk, 630090, Russian Federation
[4] Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of RAS, Irkutsk, 664033, Russian Federation

[1] vadimdavydov@outlook.com✉, https://orcid.org/0000-0002-5544-2434
[2] pikhtovnikov347@mail.ru, https://orcid.org/0009-0000-2173-0844
[3] anastaciakosanovskaya@gmail.com, https://orcid.org/0009-0006-0344-5111
[4] oleg.zaikin@icc.ru, https://orcid.org/0000-0002-0145-5010

Abstract
Cryptographic hash functions play a significant role in modern information security systems by ensuring data integrity and enabling efficient data compression. One of the most important and widely used cryptographic hash functions is SHA-256 that belongs to the SHA-2 family. In this regard, the study of SHA-256 cryptographic resistance using modern cryptanalysis approaches to preimage and collision attacks with an emphasis on the practical feasibility of such attacks is an urgent scientific task. To search for preimages of round-reduced versions of the SHA-256 compression function, the logical cryptanalysis was applied, i.e., cryptanalysis problems were reduced to the Boolean satisfiability problem (SAT). For collision attacks, a combination of logical and differential cryptanalysis was utilized. The work presents a comparison between various methods for reducing the SHA-256 compression function to SAT and evaluates their efficiency. As a result of the work, preimages for 17- and 18-round SHA-256 compression functions were found for the first time as well as preimages for a weakened 19-round compression function. Basic differential paths were constructed, which facilitated faster finding of collisions for the 18-round compression function. Known differential paths were reduced in SAT leading to finding collisions for the 19-round compression function. The work demonstrates the possibility of combining two cryptanalysis methods to enhance the efficiency of analyzing cryptographic algorithms. The results of the study confirm that the full-round SHA-256 hash function remains resistant to preimage and collision attacks within the scope of the applied SAT-based approach.

Keywords
cryptographic hash function, SHA-256, SAT, logical cryptanalysis, differential cryptanalysis

Acknowledgements
The work of Vadim Davydov and Anastasia Kiryanova was performed within the framework of the State Assignment (project No. FSER-2025-0003). Oleg Zaikin was funded by the Mathematical Center in Akademgorodok under the Agreement No. 075-15-2025-349 with the Ministry of Science and Higher Education of the Russian Federation. The present paper is an extension of results obtained during the summer school-conference “Cryptography and information security” in 2024.

For citation: Davydov V.V., Pikhtovnikov M.D., Kiryanova A.P., Zaikin O.S. Analysis of the cryptographic strength of the SHA-256 hash function using the SAT approach. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2025, vol. 25, no. 3, pp. 428–437 (in Russian). doi: 10.17586/2226-1494-2025-25-3-428-437

Введение

Хеш-функция — алгоритм, преобразовывающий входные данные и позволяющий получить их «отпечаток» (хеш; в большинстве случаев — битовую строку меньшей длины, чем входная). Для использования в задачах криптографии хеш-функция должна обладать свойствами стойкости к поиску первого прообраза, второго прообраза и коллизий. Криптографические хеш-функции всегда детерминированы, при этом небольшое изменение входа очень сильно влияет на выход — такое свойство называется «лавинным эффектом». На практике очень сложно найти коллизию, т. е. два разных сообщения с одинаковым хеш-значением, однако, так как пространство хешей много меньше пространства сообщений (если нет ограничений по длине входа), коллизии у любой хеш-функции существуют всегда.

Актуальность анализа криптографических хеш-функций обуславливается их широкой применимостью и популярностью: с их помощью можно генерировать и проверять электронные подписи, формировать блокчейн, обеспечивать безопасное хранение и выполнять проверку верности паролей, проверять целостность файлов, использовать в качестве основы для генератора псевдослучайных чисел и многое другое. Из широкой распространенности вытекает необходимость в анализе уязвимостей используемых криптографических

хеш-функций. Криптоаналитики используют различные методы и атаки для нахождения прообразов и коллизий, такие как атака дней рождения, метод «встречи посередине», дифференциальные пути, логический криптоанализ и многое другое. В логическом криптоанализе исходная задача сводится к проблеме булевой выполнимости (satisfiability или SAT), поэтому он также называется SAT-криптоанализом.

В представленной работе рассмотрена и проанализирована криптографическая хеш-функция SHA-256 [1]. Несмотря на то, что существует более новый стандарт США, хеш-функция SHA-256 все еще используется в различных приложениях, например, в криптовалюте Bitcoin в процессе майнинга и генерации bitcoin-адресов, для аутентификации пакетов программного обеспечения Debian, а также в протоколе TLS 1.3.

Криптографическая стойкость SHA-256 является предметом обширных исследований в научных работах [2]. Атаки направлены на поиск прообраза и поиск коллизий (и их вариаций) функции сжатия SHA-256. На сегодняшний день полнораундовая функция сжатия (64 раунда) остается криптографически стойкой; основной исследовательский интерес сосредоточен на попытках обращения и поиска коллизий неполнораундовых версий. Атаки можно условно разделить на теоретические и практические. Теоретическая атака свидетельствует о снижении асимптотической сложности по сравнению со сложностью полного перебора, однако, на практике такие атаки остаются неприменимыми. Практическая атака подтверждает возможность осуществления реального нахождения прообраза или коллизии в условиях ограниченных вычислительных мощностей.

Для поиска прообраза лучшей теоретической атакой на сегодняшний день является атака, предложенная в работе [3], служащая для нахождения прообраза 45-раундовой версии SHA-256 со сложностью $2^{255,5}$ вызовов функции сжатия. Лучшей практической атакой является работа [4], где был применен логический криптоанализ и найден прообраз 16-раундовой версии функции сжатия.

Лучшей теоретической атакой по поиску коллизий является работа [5], где рассмотрена возможность использования противником квантового компьютера, верхняя граница сложности атаки для нахождения коллизии 38 раундов SHA-256 — 2^{116} вызовов функции сжатия. Лучший практический результат по поиску коллизий был представлен в 2024 году на конференции Eurocrypt [6], где продемонстрировано нахождение коллизии для 31-раундовой SHA-256.

В настоящей работе приведены описание принципа работы SHA-256 и краткий аналитический обзор атак. Представлены результаты применения логического криптоанализа для поиска прообразов и коллизий неполнораундовых версий SHA-256. Впервые найдены прообразы 17- и 18-раундовых функций сжатия SHA-256. При поиске коллизий для достижения большей эффективности продемонстрировано применение гибридной атаки, использующей дифференциальные пути и логический криптоанализ. Приведены ключевые результаты проведенного исследования, также показаны направления дальнейших исследований.

Хеш-функция SHA-256

Описание и принцип работы. Хеш-функция SHA-256 принадлежит семейству SHA-2. Помимо нее в семейство также входят еще пять различных вариантов, полное описание которых можно найти в [1]. На вход принимается сообщение произвольной длины, выходом является хеш-значение, длина которого варьируется от 160 до 512 бит. Все хеш-функции семейства SHA-2 строятся на основе структуры Меркла–Дамгора [7, 8]. Входное сообщение дополняется до определенной длины, после чего разбивается на блоки заданного размера, затем происходит итеративная обработка этих блоков с применением функции сжатия. В зависимости от используемого алгоритма функция сжатия может состоять из 64 или 80 раундов.

Самой первой функцией SHA, созданной Агентством Национальной Безопасности США и Национальным институтом стандартов и технологий, была SHA-0. Она разрабатывалась в качестве стандарта для безопасной генерации и проверки цифровой подписи, а также в целях обеспечения надежного алгоритма хеширования для федеральных приложений. Однако первая открытая публикация SHA-0 была быстро отозвана из-за неназванного недостатка, и была выпущена обновленная версия алгоритма хеширования — SHA-1. Через несколько лет на смену SHA-1 пришла SHA-256.

В качестве входного сообщения SHA-256 принимает битовую строку произвольного размера (максимальная длина — (2^{64} – 1 бит), которая после дополнения разбивается на N блоков по 512 бит, выходом является 256-битное значение на последнем этапе. Функция сжатия, являющаяся основой SHA-256, состоит из 64 раундов и работает с 512-битными блоками сообщения, которые представлены в виде 16 слов по 32 бита. Основные операции для работы со словами: логические операции сложения, умножения, XOR, отрицания; сложение по модулю 2^{32}; циклический и нециклический битовые сдвиги.

Первым этапом функции сжатия является расширение 16-ти 32-битных входных слов m_i до 64 слов:

$$W_i \leftarrow \begin{cases} m_i, & 0 \le i < 16 \\ \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16}, & 16 \le i < 64, \end{cases}$$

где функции σ работают следующим образом:

$$\sigma_0(x) \leftarrow (x >>> 7) \oplus (x >>> 18) \oplus (x >> 3),$$
$$\sigma_1(x) \leftarrow (x >>> 17) \oplus (x >>> 19) \oplus (x >> 10),$$

где >>> и >> — циклический и нециклический сдвиги вправо.

Основа работы функции сжатия с сообщениями W_i заключена в 8-ми 32-битных инициализационных значениях $H^0_{(i)}(i = 0, \ldots, 7)$ и восьми внутренних регистрах a, b, c, d, e, f, g, h. Изначально эти переменные равны инициализационным значениям $H^0_{(i)}$, на каждом из по-

следующих 64 раундов функции сжатия выполняются следующие операции:

$$h = g,\ d = c,$$
$$g = f,\ c = b,$$
$$f = e,\ b = a,$$
$$e = d + T_1,\ a = T_1 + T_2,$$

где

$$T_1 = h + \Sigma_1(h) + Ch(e, f, g) + K_t + W_t,$$
$$T_2 = \Sigma_0(a) + Maj(a, b, c),$$
$$Ch(x, y, z) = (x \wedge y) \oplus (\bar{a} \wedge z),$$
$$Maj(x, y, z) = (x \wedge y) \oplus (y \wedge z) \oplus (x \wedge z),$$
$$\Sigma_1(x) = (x >>> 6) \oplus (x >>> 11) \oplus (x >>> 25),$$
$$\Sigma_0(x) = (x >>> 2) \oplus (x >>> 13) \oplus (x >>> 22),$$

где K_t — раундовая константа; W_t — слово на раунде t, $t = 0, \ldots, 63$. В конце каждого раунда происходит обновление промежуточных значений $H(i)$, $i = 0, \ldots, N$.

Таким образом, на последнем 64-ом раунде, после вычисления значений регистров $a - h$ и обновления состояний $H^{(N)}$, финальное значение хеша равно конкатенации:

$$H_0^{(N)}||H_1^{(N)}||H_2^{(N)}||H_3^{(N)}||H_4^{(N)}||H_5^{(N)}||H_6^{(N)}||H_7^{(N)}.$$

Методы атак на SHA-256. На рассматриваемую хеш-функцию существует ряд теоретических и практических атак. SHA-256, как и любая другая криптографическая хеш-функция, должна обладать определенными свойствами, которые необходимо проанализировать в первую очередь при доказательстве безопасности, а именно — стойкость к поиску первого и второго прообразов и коллизий.

— Поиск первого прообраза (односторонность): при наличии хеш-значения $H(m)$ сообщения m необходимо вычислить само сообщение m.
— Поиск второго прообраза: при наличии сообщения m_1 и его хеш-значения $H(m_1)$ необходимо найти другое сообщение m_2, такое что $H(m_1) = H(m_2)$.
— Поиск коллизии: необходимо найти два различных сообщения $m_1 \neq m_2$, таких что их хеш-значения совпадают $H(m_1) = H(m_2)$.

На криптографические хеш-функции существует множество атак, но так или иначе все они являются вариациями поиска прообразов или коллизий с дополнительными требованиями или упрощениями. Так, при поиске псевдопрообразов или псевдоколлизий необходимо наложить условие, что атакующий может изменять значение вектора инициализации IV, а при поиске частичного прообраза атакующему будет достаточно восстановить часть сообщения [9, 10]. При реализации биклик-атаки (одна из разновидностей атак «встречи посередине») необходимо использовать структуру полного двудольного графа для того, чтобы увеличить количество атак «человека посередине». Еще одна уязвимость, присущая SHA-256, выявляется атакой удлинением сообщения. Данной атаке подвержены все итеративные конструкции, у которых финальное преобразование внутреннего состояния не отличается от всех предыдущих итераций. Зная длину исходного сообщения и его хеш-значение, злоумышленник может реализовать дополнительную $n + 1$ итерацию функции сжатия, подставив хеш-значение исходного сообщения и дополнив его своим сообщением с дополнением (паддингом). После проведения всех манипуляций хеш-функция выдаст валидное новое хеш-значение. Данная атака работает, даже если атакующий не знает самого исходного сообщения.

Не все эти атаки являются общими и не могут применяться ко всем хеш-функциям. Для оценки безопасности любой криптографической хеш-функции, вне зависимости от ее конструкции, можно использовать различные виды криптоанализа, например, алгебраический, логический, дифференциальный или линейный криптоанализы.

В работе [11] предложена алгебраическая атака на основе сбоев для анализа безопасности шифров или хеш-функций. При таком подходе происходит объединение атак по сторонним каналам с алгебраическими методами — для начала нужно внести сбои в расчетах аппаратного устройства, после чего построить алгебраические уравнения для хеш-функции, а в качестве новых значений использовать новые данные с ошибками. После этого применяются автоматические инструменты для решения уравнений и восстановления необходимой информации. В работе [12] алгебраическая атака на основе сбоев совмещена с SAT-решателями для восстановления секретных битов из аппаратных реализаций семейства хеш-функций SHA-1 и SHA-2.

В настоящей работе были применены методы дифференциального и логического криптоанализов. В основе логического криптоанализа лежит проблема выполнимости булевых формул, записанных в конъюнктивной нормальной форме (КНФ).

Необходимо определить, принимает ли формула, записанная в КНФ, значение «истина», и если да, то найти значения переменных, входящих в нее. На практике это означает найти прообраз имеющегося хеш-значения (т. е. обратить хеш-функцию), либо найти несколько сообщений с одним и тем же хеш-значением (коллизия) — все зависит от формулировки и кодирования задачи.

Логический криптоанализ является универсальным средством криптоанализа хеш-функций. Он не позволяет получить теоретические оценки сложности, однако, дает возможность оценить сложность с практической точки зрения. Примером применения логического криптоанализа для поиска прообразов криптографических хеш-функций является работа [13], где использован SAT-решатель Kissat для анализа стойкости криптографических хеш-функций — финалистов конкурса National Institute of Standards and Technology. В работе [14] применен SAT-решатель совместно с системой компьютерной алгебры CAS для поиска коллизий с полусвободным стартом неполнораундовой версии SHA-256. Интересный подход с применением SAT-решателя представлен в работе [15], где были найдены так называемые «квантовые» коллизии (поиск

коллизий с помощью алгоритма Гровера) хеш-функций семейства SHA-3; ученые использовали решатели SAT для более эффективного поиска коллизий, а также для оптимизации подхода в общем.

Идея дифференциального криптоанализа выдвинута в работе [16], где был предложен оригинальный метод анализа, позволяющий на тот момент времени взломать неполнораундовый шифр DES. Суть дифференциального анализа заключается в исследовании зависимости, как небольшая разница между двумя входными сообщениями влияет на их выходные значения. В случае криптографических хеш-функций это помогает при поиске коллизий: необходимо найти два различных сообщения, которые дают один и тот же хеш, при условии наложения ограничения на разницу внутренних состояний на каждом раунде. В качестве примера применения дифференциального криптоанализа можно привести работы [17–19], где был представлен анализ хеш-функций SHA-0, MD5, MD4 и RIPEMD.

Оценка стойкости SHA-256. Ральф Меркл и Иван Дамгар на конференции CRYPTO в 1989 году предложили структуру, в рамках которой стойкость криптографической хеш-функции основывается на стойкости ее функции сжатия. Потому зачастую при оценке безопасности всей хеш-функции достаточно проанализировать стойкость функции сжатия в ее основе. Верхняя граница сложности поиска коллизий на полную версию SHA-256 — около 2^{128} операций вызова функции (полный перебор требует 2^{256} операций вызова, парадокс дней рождения позволяет сократить это количество). Верхняя оценка сложности поиска прообраза в худшем случае — полный перебор, т. е. 2^{256} операций вызова функции. С помощью различных методов и атак можно понизить верхние оценки сложности, однако, в общем случае сделать это затруднительно, а такие оценки для SHA-256 сегодня носят чисто теоретический характер. Это означает, что, хотя с помощью различных атак сложность поиска прообраза и коллизий снижается в сравнении с методом полного перебора, она все еще остается за пределами вычислительных возможностей современных компьютеров.

Атаки нахождения прообраза. Первая атака на поиск первого и второго прообразов SHA-256 основывалась на технике «встречи посередине»; авторы реализовали атаку на урезанную версию, состоящую из 24 раундов, и смогли снизить сложность до 2^{240} вызовов функции сжатия [20]. Впоследствии было проведено множество других атак и их вариаций для поиска прообраза или псевдопрообраза. В работе [21] предложена теоретическая вариация атаки «встречи посередине» для нахождения прообраза 42-раундовой версии SHA-256 со сложностью $2^{248,4}$ вызовов функции сжатия по времени. В [4] применен логический криптоанализ и найден прообраз 16-раундовой версии функции сжатия. Последней успешной на сегодняшний день работой является [3], в которой предложены атаки нахождения прообраза 45-раундовой версии SHA-256 и псевдопрообраза 52-раундовой версии SHA-256. В обоих случаях это были теоретические биклик-атаки по двудольному графу: сложность была снижена до $2^{255,5}$ и 2^{255} вызовов функции сжатия соответственно. Отметим, что в настоящий момент большинство атак на поиск прообраза SHA-256 являются теоретическими.

Атаки нахождения коллизий. На практике получить коллизии хеш-функции SHA-256 впервые удалось в работе [22], в результате были получены коллизии для 18 раундов. В дальнейшем подходы к поиску коллизий были улучшены. В [23] была реализована вариация атаки «встречи посередине», в результате которой удалось найти коллизию для 28-раундовой функции сжатия и коллизию с полусвободным стартом для 31-раундовой функции сжатия SHA-256 со сложностью $2^{65,5}$ по времени. В работе [6] использован инструмент на основе SAT/SMT для эффективного поиска дифференциальных характеристик SHA-256; в итоге были найдены коллизии для 31 раунда (сложность по времени — $2^{49,5}$ операций вызова); псевдоколлизии для 52 раундов (сложность по времени — $2^{127,5}$ операций вызова) и коллизии с полусвободным стартом для 39 раундов (сложность по времени зависит от реализации атаки и начальных условий, например, если рассматривать только одну стартовую точку, то сложность по времени будет равна $2^{115,6}$ операциям вызова). Также в 2024 году в работе [14] был использован логический криптоанализ для нахождения коллизий с полусвободным стартом для 38-раундовой версии SHA-256.

Поиск прообразов неполнораундовых версий функции сжатия SHA-256

Применим метод логического криптоанализа для поиска прообразов усеченных (неполнораундовых) версий функции сжатия SHA-256. Для проведения экспериментов была выделена функция сжатия SHA-256 и построены ее SAT-кодировки, т. е. были сгенерированы соответствующие КНФ (четыре случая). Характеристики КНФ и время нахождения прообразов представлены в табл. 1, 2. Во всех случаях задачей являлось нахождение прообраза нулевого выхода функции сжатия, т. е. выхода, состоящего из 256 нулевых битов. Все эксперименты проводились на персональном компьютере со следующими характеристиками: 16-ядерный процессор AMD Ryzen 3950X, оперативная память — 64 ГБ.

При проведении экспериментов для получения КНФ использовались следующие программы: C Bounded Model Checker (CBMC), Transalg и SAT-encoding.

Транслятор CBMC [24] — транслятор общего назначения, предназначенный для преобразования программ на языках C и C++ в булевы формулы, которые затем анализируются для выявления ошибок и нарушения корректности исполнения. Процесс создания SAT-кодировок криптографических хеш-функций с помощью CBMC подробно описан в работе [13]. Транслятор Transalg [25] — специализированный транслятор, предназначенный для описания криптографических алгоритмов на предметно-ориентированном языке и формирующий файл с соответствующей КНФ. SAT-encoding — специальная программа, доступная в открытом репозитории [26], предназначенная для построения SAT-кодировок прикладных задач. SAT-encoding позволяет формировать SAT-кодировки функ-

ций сжатия криптографических хеш-функций MD4, SHA-1 и SHA-256. Программа также предоставляет возможность настройки дополнительных параметров: выбора алгоритма хеширования, типа кодирования целочисленного сложения, указания количества раундов для кодировки, определения цели генерации КНФ (поиск прообраза или коллизии), а также задания фиксированного размера входного сообщения (по умолчанию размер сообщения не фиксирован).

Для решения полученных КНФ применялся SAT-решатель Kissat [27], реализующий алгоритм CDCL [28].

В первом случае использована готовая программная реализация хеш-функции SHA-256 [1] на языке C. Для создания кодировки программный код был адаптирован: выделена функция сжатия, убран весь ненужный функционал, проверена корректность реализации и сформулирована задача поиска прообраза неполнораундовой версии функции сжатия. После этого файл с программным кодом подавался на вход транслятору CBMC для получения файла, содержащего КНФ. Таким образом, были сделаны КНФ для нахождения прообразов 16-, 17-, 18- и 19-раундовых функций сжатия SHA-256.

На построенных файлах был запущен SAT-решатель Kissat. Установлено ограничение по времени решения в 5000 с.

Во втором случае использован транслятор Transalg. Программа для функции сжатия SHA-256 доступна в репозитории этого транслятора [29]. Для экспериментов в этой программе варьировалось число раундов и задавался нулевой выход. Результаты, полученные с помощью Kissat, представлены в табл. 1.

В третьем случае была использована программа SAT-encoding. Для работы программы нужно задать вход и выход функции сжатия. При этом выход функции сжатия либо вводится вручную, либо автоматически генерируется случайным образом. Программа была модифицирована таким образом, чтобы была возможность задать нулевой выход.

Таблица 1. Характеристики КНФ, построенные с помощью CBMC и Transalg, а также время нахождения прообразов с помощью решателя Kissat

Table 1. Characteristics of CNF constructed using CBMC and Transalg as well as the time for finding preimages using the Kissat solver

Раунды	Переменные	Дизъюнкты	Время, с
CBMC			
16	35 136	148 609	0,05
17	35 966	152 848	0,25
18	36 794	157 080	12,53
19	37 622	161 312	не решено
Transalg			
16	22 212	143 478	0,06
17	22 774	147 154	0,36
18	23 334	150 823	5,76
19	23 894	154 492	не решено

Примечание. «не решено» — решатель не смог найти выполняющий набор за выделенное время.

В программе SAT-encoding предусмотрено использование различных алгоритмов кодирования в SAT целочисленного сложения: counter chain — использование счетчиков (суммирование битов операндов в одном столбце) в режиме каскадного суммирования, dot matrix — обработка и уменьшение матрицы операндов при помощи дерева Уоллеса, espresso — аналог counter chain с использованием логического минимизатора Espresso, two operand — одновременное внесение двух операндов. В программе SAT-encoding отсутствовала реализация Espresso, поэтому был подключен данный минимизатор в качестве внешней программы, взятой в открытом репозитории[1]. После сборки и запуска модифицированной программы были получены КНФ для 16, 17, 18 и 19 раундов функции сжатия для каждого вида кодирования сложения — итого 16 КНФ. Характеристики КНФ и результаты работы Kissat на них представлены в табл. 2.

В результате выполненных экспериментов можно сделать вывод, что нахождение прообраза для 18-раундовой функции сжатия SHA-256 на сегодняшний день — простая задача, и быстрее всего SAT-решатель Kissat ее решил на кодировке, сделанной с помощью SAT-encoding с использованием Espresso.

Так как задача нахождения прообраза 19-раундовой функции сжатия SHA-256 оказалась слишком сложной для решателя Kissat, были рассмотрены ослабленные версии данной задачи. Ослабление заключалось в том, что был известен не весь 256-битный выход, а только его часть. В такой постановке у каждого выхода функции сжатия увеличивается число соответствующих прообразов, что упрощает поиск хотя бы одного из них. Была построена кодировка 19 раундов функции сжатия SHA-256 с помощью SAT-encoding и Espresso. Затем сделаны 32 КНФ: в первой известен только первый бит выхода и он равен нулю; во второй известны первые два нулевых бита и так далее. На каждой из этих КНФ запущен Kissat с ограничением времени в 5000 с. В итоге удалось найти прообраз максимум для 24-битного нулевого выхода. Для сравнения, если закодированы все 64 раунда функции сжатия SHA-256, то в тех же условиях Kissat находит решение максимум для КНФ, в которой заданы нулями первые 17 битов выхода.

Существуют разные подходы к распараллеливанию трудных SAT-задач. Некоторые из них основаны на варьировании значений множества переменных КНФ [30]. Для решения трудных задач был выбран другой подход, который называется Cube-and-Conquer. В рамках этого подхода на первом этапе с помощью lookahead-решателя исходная задача упрощается и разбивается на независимые подзадачи, а затем на втором этапе эти подзадачи решаются с помощью CDCL-решателя. На

[1] University of California/Espresso-logic-minimizer [Электронный ресурс]. Режим доступа: https://github.com/classabbyamp/espresso-logic, свободный. Яз. англ. (дата обращения: 24.07.2024).

Таблица 2. Характеристики КНФ, построенные с помощью алгоритмов SAT-encoding, а также время нахождения прообразов с помощью решателя Kissat

Table 2. Characteristics of CNF constructed using algorithms SAT-encoding, as well as the time for finding preimages using the Kissat solver

Раунды	Перем.	Дизъюнкты	Время, с	Перем.	Дизъюнкты	Время, с
	Алгоритм Counter chain			Алгоритм Dot matrix		
16	11 936	70 504	0,04	10 760	66 088	0,05
17	12 918	76 651	0,14	11 604	71 727	0,13
18	13 900	82 798	7,52	12 448	77 366	8,89
19	14 882	88 945	не решено	13 292	83 005	не решено
Раунды	Алгоритм Two operand			Алгоритм Espresso		
16	11 400	66 536	0,02	7920	14 336	0,01
17	12 286	72 210	0,14	8495	140 442	0,76
18	13 172	77 884	20,60	9070	151 508	2,38
19	14 058	83 558	не решено	9645	162 574	не решено

Примечание. «не решено» — решатель не смог найти выполняющий набор за выделенное время; «Перем.» — число переменных.

8-ми 19-раундовых КНФ был запущен параллельный SAT-решатель EnCnC, реализующий метод Cube-and-Conquer [31]. Решатель был запущен с ограничением времени в 24 часа на 16-ядерном процессоре. На втором этапе использован Kissat, т. е. он был запущен на соответствующих подзадачах. В итоге для всех КНФ, кроме самой сложной (с 32-битным нулевым выходом), удалось найти решение. Все полученные КНФ доступны онлайн [32].

Можно сделать вывод, что нахождение прообраза для 19 раундов функции сжатия SHA-256 является слишком сложной задачей даже если тщательно подобрать SAT-кодировку и использовать параллельные вычисления. Отметим также, что по сравнению с работой [4], в которой был найден прообраз максимум для 16 раундов функции сжатия SHA-256 — достигнут явный прогресс.

Поиск коллизий для неполнораундовых версий функции сжатия SHA-256

Объединим методы и подходы логического криптоанализа для поиска коллизий в неполнораундовых версиях функции сжатия SHA-256. Выполним проверку эффективности совмещения двух предложенных подходов.

Задача поиска коллизий в данном случае формулируется следующим образом. Пусть M и M' — произвольные сообщения, $h(\cdot)$ — функция сжатия. Необходимо найти такие $M \neq M'$ что $H(M) = H(M')$.

Сначала применим только логический криптоанализ для поиска коллизий. Для этого закодируем поставленную задачу следующим образом. Зададим два произвольных входных сообщения M_1 и M_2 при этом наложим на них ограничения таким образом, чтобы они были различны. Также укажем, что выход функции сжатия после применения к обоим сообщениям будет одинаков: $H(M_1) = H(M_2)$. После чего попробуем решить задачу поиска коллизий с помощью Kissat.

Задача относительно легко решается для 18 раундов функции сжатия — поиск коллизии занял 34 с. Для 19 раундов задачу данным методом решить не удалось. Для более эффективного поиска коллизий можно попробовать объединить логический и дифференциальный криптоанализы. На каждом раунде используются 8 регистров: a, b, c, d, e, f, g, h, где каждый имеет размерность 32 бит, при этом выходом функции сжатия являются эти же регистры. Чтобы сократить пространство возможных сообщений для поиска коллизий, можно наложить ограничения на разность части регистров на некоторых раундах: таким образом можно построить дифференциальный путь. Например, для регистра a дифференциальный путь в общем виде будет выглядеть следующим образом:

$$\Delta a_{r_1} \to \Delta a_{r_2} \to \cdots \to \Delta a_{r_{n-1}} \to \Delta a_{r_n},$$

где $\{r_1, \ldots, r_n\}$ — раунды (всего n); на последнем раунде $\Delta a_{r_n} = 0$, так как выход регистра должен совпадать для сообщений M и M'. Таким образом, ограничения на значения регистров позволят уменьшить сложность перебора. Однако следует отметить, что сам поиск нужных ограничений — нетривиальная задача.

Для проверки зададим единственное значение дифференциального пути $\Delta a_{r_1} = 1$, сократим число раундов до 18, закодируем задачу поиска коллизии, получим КНФ с помощью CBMC и попробуем применить Kissat для решения задачи. Задача решается за 19 с. Если задано единственное значение дифференциального пути $\Delta e_{r_1} = 1$, то Kissat находит решение за 3 с. Таким образом, путем добавления всего лишь одного условия удалось достичь значительной оптимизации.

Важно заметить, что это не всегда так. В некоторых случаях, напротив, ограничения могут приводить к ухудшению производительности, а в некоторых коллизию найти и вовсе не удастся. Иногда совместное применение ограничений может привести к моментальному UNSAT, что означает — для таких ограничений

Таблица 3. Коллизия для 19-раундовой функции сжатия SHA-256
Table 3. Collision for the 19-round SHA-256 compression function

0x71646011	**0x5479452**	**0x3791dfb5**	0x1d1338de
0x9900a4c7	0x60745f6b	0x1e0ac5de	0x9b286266
0xef7f4755	**0x65ad1cba**	**0xab32b24**	0x9dc00b20
0xb6f13681	**0xcd20048c**	**0xdc967d4e**	**0xb1e8b4b**
0x71646011	**0x5479452**	**0x3791dfb5**	0xf51f13c9
0xd4c569f1	0xb9e206a2	0x5df3b728	0xfad4f80e
0xd46a748	**0x65ad1cba**	**0xab32b24**	0xc5b43035
0xb6f13681	**0xcd20048c**	**0xdc967d4e**	**0xb1e8b4b**

Примечание. Совпадающие 32-битные слова в сообщениях выделены полужирным шрифтом.

формула невыполнима, и они не могут быть использованы. Например, если совместно задать $\Delta a_{r_1} = 1$ и $\Delta e_{r_1} = 5$ (для 19-раундовой функции сжатия), Kissat выдает UNSAT (время выполнения 0,06 с), что доказывает — коллизий при указанных двух ограничениях не существует.

Для проверки корректности подхода закодируем все дифференциальные пути из работы [33] для 19-раундовой функции сжатия SHA-256. Kissat решает данную задачу за 2 мин 2 с. Найденная коллизия представлена в табл. 3.

Для 20-раундовой функции сжатия с теми же дифференциальными путями коллизию найти не удалось. Отметим, что полученный результат уступает существующим научным работам, например, в [23] была найдена коллизия для 28-раундовой версии SHA-256. Эффективное нахождение дифференциальных путей для полнораундовой версии хеш-функции остается на сегодняшний день открытой проблемой.

Заключение

В работе приведен логический криптоанализ алгоритма криптографической хеш-функции SHA-256. Исследована сама хеш-функция и описаны возможные атаки. В качестве эксперимента для нахождения прообразов и коллизий были использованы различные трансляторы для получения конъюнктивных нормальных форм и SAT-решатели для решения проблемы булевой выполнимости. В результате работы были найдены прообразы и коллизии 18-раундовой функции сжатия SHA-256. Изучена техника совместного использования дифференциального и логического криптоанализов. В результате для 18-раундовой функции сжатия удалось значительно ускорить время выполнения логического криптоанализа, а также получилось найти коллизии для 19-раундовой функции сжатия. В рамках дальнейших исследований планируется усовершенствовать предложенные в работе методы поиска прообразов и коллизий, а также изучить дополнительные подходы к криптоанализу алгоритма SHA-256 для достижения лучших результатов.

Литература

1. Secure hash standard (shs) // Fips pub. 2012. V. 180. N 4.
2. Alamgir N. Programmatic SAT for SHA-256 Collision Attack. 2024 [Электронный ресурс]. URL:https://scholar.uwindsor.ca/etd/9525 (дата обращения: 12.07.2024).
3. Khovratovich D., Rechberger C., Savelieva A. Bicliques for preimages: attacks on Skein-512 and the SHA-2 family // Lecture Notes in Computer Science. 2012. V. 7549. P. 244–263. https://doi.org/10.1007/978-3-642-34047-5_15
4. Homsirikamol E., Morawiecki P., Rogawski M., Srebrny M. Security margin evaluation of SHA-3 contest finalists through SAT-based attacks // Lecture Notes in Computer Science. 2012. V. 7564. P. 56–67. https://doi.org/10.1007/978-3-642-33260-9_4
5. Hosoyamada A., Sasaki Y. Quantum collision attacks on reduced SHA-256 and SHA-512 // Lecture Notes in Computer Science. 2021. V. 12825. P. 616–646. https://doi.org/10.1007/978-3-030-84242-0_22
6. Li Y., Liu F., Wang G. New records in collision attacks on SHA-2 // Lecture Notes in Computer Science. 2024. V. 14651. P. 158–186. https://doi.org/10.1007/978-3-031-58716-0_6
7. Damgard I.B. A design principle for hash functions // Lecture Notes in Computer Science. 1990. P. V. 435. P. 416–427. https://doi.org/10.1007/0-387-34805-0_39
8. Merkle R.C. A certified digital signature // Lecture Notes in Computer Science. 1990. V. 435. P. 218–238. https://doi.org/10.1007/0-387-34805-0_21
9. Al-Kuwari S., Davenport J.H., Bradford R. J. Cryptographic hash functions: Recent design trends and security notions // Proc. of the 6th

References

1. Secure hash standard (shs). *Fips pub*, 2012, vol. 180, no. 4.
2. Alamgir N. *Programmatic SAT for SHA-256 Collision Attack*. 2024 Available at: https://scholar.uwindsor.ca/etd/9525 (accessed: 12.07.2024).
3. Khovratovich D., Rechberger C., Savelieva A. Bicliques for preimages: attacks on Skein-512 and the SHA-2 family. *Lecture Notes in Computer Science*. 2012, vol. 7549, pp. 244–263. https://doi.org/10.1007/978-3-642-34047-5_15
4. Homsirikamol E., Morawiecki P., Rogawski M., Srebrny M. Security margin evaluation of SHA-3 contest finalists through SAT-based attacks. *Lecture Notes in Computer Science*, 2012, vol. 7564, pp. 56–67. https://doi.org/10.1007/978-3-642-33260-9_4
5. Hosoyamada A., Sasaki Y. Quantum collision attacks on reduced SHA-256 and SHA-512. *Lecture Notes in Computer Science*, 2021, vol. 12825, pp. 616–646. https://doi.org/10.1007/978-3-030-84242-0_22
6. Li Y., Liu F., Wang G. New records in collision attacks on SHA-2. *Lecture Notes in Computer Science*, 2024, vol. 14651, pp. 158–186. https://doi.org/10.1007/978-3-031-58716-0_6
7. Damgard I.B. A design principle for hash functions. *Lecture Notes in Computer Science*, 1990, vol. 435, pp. 416–427. https://doi.org/10.1007/0-387-34805-0_39
8. Merkle R.C. A certified digital signature. *Lecture Notes in Computer Science*, 1990, vol. 435, pp. 218–238. https://doi.org/10.1007/0-387-34805-0_21
9. Al-Kuwari S., Davenport J.H., Bradford R. J. Cryptographic hash functions: Recent design trends and security notions. *Proc. of the 6th*

China International Conference on Information Security and Cryptology (Inscrypt ‘10). 2010. P. 133–150.
10. Gauravaram P. Cryptographic Hash Functions: Cryptanalysis, Design and Applications. PhD thesis. Queensland University of Technology. 2007. 298 p.
11. Courtois N.T., Jackson K., Ware D. Fault-algebraic attacks on inner rounds of DES // Proc. of the E-Smart’10. 2010. P. 22–24.
12. Nejati S., Horacek J., Gebotys C., Ganesh V. Algebraic fault attack on sha hash functions using programmatic SAT solvers // Lecture Notes in Computer Science. 2018. V. 11008. P. 737–754. https://doi.org/10.1007/978-3-319-98334-9_47
13. Заикин О.С., Давыдов В.В., Кирьянова А.П. Применение алгоритмов решения проблемы булевой выполнимости для анализа финалистов конкурса SHA-3 // Вычислительные методы и программирование. 2024. Т. 25. С. 259–273. https://doi.org/10.26089/NumMet.v25r320
14. Alamgir N., Nejati S., Bright C. SHA-256 collision attack with programmatic SAT // CEUR Workshop Proceedings. 2024. V. 3717. P. 91–110.
15. Guo J., Liu G., Song L., Tu Y. Exploring SAT for cryptanalysis:(Quantum) collision attacks against 6-round SHA-3 // Lecture Notes in Computer Science. 2022. V. 13793. P. 645–674. https://doi.org/10.1007/978-3-031-22969-5_22
16. Biham E., Shamir A. Differential cryptanalysis of DES-like cryptosystems // Journal of Cryptology. 1991. V. 4. N 1. P. 3–72. https://doi.org/10.1007/BF00630563
17. Wang X., Yu H. How to break MD5 and other hash functions // Lecture Notes in Computer Science. 2005. V. 3494. P. 19–35. https://doi.org/10.1007/11426639_2
18. Wang X., Lai X., Feng D., Chen H., Yu X. Cryptanalysis of the hash functions MD4 and RIPEMD // Lecture Notes in Computer Science. 2005. V. 3494. P. 1–18. https://doi.org/10.1007/11426639_1
19. Wang X., Yu H., Yin Y.L. Efficient collision search attacks on SHA-0 // Lecture Notes in Computer Science. 2005. V. 3621. P. 1–16. https://doi.org/10.1007/11535218_1
20. Isobe T., Shibutani K. Preimage attacks on reduced Tiger and SHA-2 // Lecture Notes in Computer Science. 2009. V. 5665. P. 139–155. https://doi.org/10.1007/978-3-642-03317-9_9
21. Guo J., Ling S., Rechberger C., Wang H. Advanced meet-in-the-middle preimage attacks: First results on full Tiger, and improved results on MD4 and SHA-2 // Lecture Notes in Computer Science. 2010. V. 6477. P. 56–75. https://doi.org/10.1007/978-3-642-17373-8_4
22. Mendel F., Pramstaller N., Rechberger C., Rijmen V. Analysis of step-reduced sha-256 // Lecture Notes in Computer Science. 2006. V. 4047. P. 126–143. https://doi.org/10.1007/11799313_9
23. Mendel F., Nad T., Schläffer M. Improving local collisions: New attacks on reduced SHA-256 // Lecture Notes in Computer Science. 2013. V. 7881. P. 262–278. https://doi.org/10.1007/978-3-642-38348-9_16
24. Clarke E., Kroening D., Lerda F. A tool for checking ANSI-C programs // Lecture Notes in Computer Science. 2004. V. 2988. P. 168–176. https://doi.org/10.1007/978-3-540-24730-2_15
25. Semenov A., Otpuschennikov I., Gribanova I., Zaikin O., Kochemazov S. Translation of algorithmic descriptions of discrete functions to SAT with applications to cryptanalysis problems // Logical Methods in Computer Science. 2020. V. 16. N 1. P. 29. https://doi.org/10.23638/LMCS-16(1:29)2020
26. Nejati S. SAT Encoding [Электронный ресурс]. URL: https://github.com/saeednj/SAT-encoding (дата обращения: 12.07.2024).
27. Biere A. The Kissat SAT Solver [Электронный ресурс]. URL: https://github.com/arminbiere/kissat.git (дата обращения: 12.07.2024).
28. Marques-Silva J.P., Sakallah K.A. GRASP: A search algorithm for propositional satisfiability // IEEE Transactions on Computers. 1999. V. 48. N 5. P. 506–521. https://doi.org/10.1109/12.769433
29. Otpuschennikov I. Programs for SHA-256 [Электронный ресурс]. URL: https://gitlab.com/satencodings/satencodings/-/tree/master/SHA2?ref_type=heads (дата обращения: 12.07.2024).
30. Semenov A., Zaikin O., Kochemazov S. Finding effective SAT partitionings via black-box optimization // Springer Optimization and Its Applications. 2021. V. 170. P. 319–355. https://doi.org/10.1007/978-3-030-66515-9_11
31. Zaikin O. Inverting cryptographic hash functions via Cube-and-Conquer // Journal of Artificial Intelligence Research. 2024. V. 81. P. 359–399. https://doi.org/10.1613/jair.1.15244
32. Заикин О. КНФ для SHA-256 [Электронный ресурс]. URL: https://github.com/olegzaikin/sha256sat.git (дата обращения: 20.02.2025).

China International Conference on Information Security and Cryptology (Inscrypt ‘10), 2010, pp. 133–150.
10. Gauravaram P. *Cryptographic Hash Functions: Cryptanalysis, Design and Applications*. PhD thesis. Queensland University of Technology, 2007, 298 p.
11. Courtois N.T., Jackson K., Ware D. Fault-algebraic attacks on inner rounds of DES. *Proc. of the E-Smart’10*, 2010, pp. 22–24.
12. Nejati S., Horacek J., Gebotys C., Ganesh V. Algebraic fault attack on sha hash functions using programmatic SAT solvers. *Lecture Notes in Computer Science*, 2018, vol. 11008, pp. 737–754. https://doi.org/10.1007/978-3-319-98334-9_47
13. Zaikin O.S., Davydov V.V., Kiryanova A.P. SAT-based analysis of SHA-3 competition finalists. *Numerical Methods and Programming*, 2024, vol. 25, pp. 259–273. (in Russian). https://doi.org/10.26089/NumMet.v25r320
14. Alamgir N., Nejati S., Bright C. SHA-256 collision attack with programmatic SAT. *CEUR Workshop Proceedings*, 2024, vol. 3717, pp. 91–110.
15. Guo J., Liu G., Song L., Tu Y. Exploring SAT for cryptanalysis:(Quantum) collision attacks against 6-round SHA-3. *Lecture Notes in Computer Science*, 2022, vol. 13793, pp. 645–674. https://doi.org/10.1007/978-3-031-22969-5_22
16. Biham E., Shamir A. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 1991, vol. 4, no. 1, pp. 3–72. https://doi.org/10.1007/BF00630563
17. Wang X., Yu H. How to break MD5 and other hash functions. *Lecture Notes in Computer Science*, 2005, vol. 3494, pp. 19–35. https://doi.org/10.1007/11426639_2
18. Wang X., Lai X., Feng D., Chen H., Yu X. Cryptanalysis of the hash functions MD4 and RIPEMD. *Lecture Notes in Computer Science*, 2005, vol. 3494, pp. 1–18. https://doi.org/10.1007/11426639_1
19. Wang X., Yu H., Yin Y.L. Efficient collision search attacks on SHA-0. *Lecture Notes in Computer Science*, 2005, vol. 3621, pp. 1–16. https://doi.org/10.1007/11535218_1
20. Isobe T., Shibutani K. Preimage attacks on reduced Tiger and SHA-2. *Lecture Notes in Computer Science*, 2009, vol. 5665, pp. 139–155. https://doi.org/10.1007/978-3-642-03317-9_9
21. Guo J., Ling S., Rechberger C., Wang H. Advanced meet-in-the-middle preimage attacks: First results on full Tiger, and improved results on MD4 and SHA-2. *Lecture Notes in Computer Science*, 2010, vol. 6477, pp. 56–75. https://doi.org/10.1007/978-3-642-17373-8_4
22. Mendel F., Pramstaller N., Rechberger C., Rijmen V. Analysis of step-reduced sha-256. *Lecture Notes in Computer Science*, 2006, vol. 4047, pp. 126–143. https://doi.org/10.1007/11799313_9
23. Mendel F., Nad T., Schläffer M. Improving local collisions: New attacks on reduced SHA-256. *Lecture Notes in Computer Science*, 2013, vol. 7881, pp. 262–278. https://doi.org/10.1007/978-3-642-38348-9_16
24. Clarke E., Kroening D., Lerda F. A tool for checking ANSI-C programs. *Lecture Notes in Computer Science*, 2004, vol. 2988, pp. 168–176. https://doi.org/10.1007/978-3-540-24730-2_15
25. Semenov A., Otpuschennikov I., Gribanova I., Zaikin O., Kochemazov S. Translation of algorithmic descriptions of discrete functions to SAT with applications to cryptanalysis problems. *Logical Methods in Computer Science*, 2020, vol. 16, no. 1, pp. 29. https://doi.org/10.23638/LMCS-16(1:29)2020
26. Nejati S. *SAT Encoding*. Available at: https://github.com/saeednj/SAT-encoding (accessed: 12.07.2024).
27. Biere A. *The Kissat SAT Solver*. Available at: https://github.com/arminbiere/kissat.git (accessed: 12.07.2024).
28. Marques-Silva J.P., Sakallah K.A. GRASP: A search algorithm for propositional satisfiability. *IEEE Transactions on Computers*, 1999, vol. 48, no. 5, pp. 506–521. https://doi.org/10.1109/12.769433
29. Otpuschennikov I. *Programs for SHA-256*. Available at: https://gitlab.com/satencodings/satencodings/-/tree/master/SHA2?ref_type=heads (accessed: 12.07.2024).
30. Semenov A., Zaikin O., Kochemazov S. Finding effective SAT partitionings via black-box optimization. *Springer Optimization and Its Applications*, 2021, vol. 170, pp. 319–355. https://doi.org/10.1007/978-3-030-66515-9_11
31. Zaikin O. Inverting cryptographic hash functions via Cube-and-Conquer. *Journal of Artificial Intelligence Research*, 2024, vol. 81, pp. 359–399. https://doi.org/10.1613/jair.1.15244
32. Zaikin O. *CNF for SHA-256*. Available at: https://github.com/olegzaikin/sha256sat.git (accessed: 20.02.2025). (in Russian)

33. Sanadhya S.K., Sarkar P. Attacking reduced round SHA-256 // Lecture Notes in Computer Science. 2008. V. 5037. P. 130–143. https://doi.org/10.1007/978-3-540-68914-0_8

33. Sanadhya S.K., Sarkar P. Attacking reduced round SHA-256. *Lecture Notes in Computer Science*, 2008, vol. 5037, pp. 130–143. https://doi.org/10.1007/978-3-540-68914-0_8

Авторы

Давыдов Вадим Валерьевич — кандидат технических наук, криптограф-исследователь, ООО «КуАпп», Москва, 121205, Российская Федерация; доцент, Санкт-Петербургский государственный университет аэрокосмического приборостроения, 190000, Санкт-Петербург, Российская Федерация; научный сотрудник, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, sc 57203909696, https://orcid.org/0000-0002-5544-2434, vadimdavydov@outlook.com

Пихтовников Михаил Денисович — студент, Южный федеральный университет, Таганрог, 347922, Российская Федерация, https://orcid.org/0009-0000-2173-0844, pikhtovnikov347@mail.ru

Кирьянова Анастасия Павловна — аспирант, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, https://orcid.org/0009-0006-0344-5111, anastaciakosanovskaya@gmail.com

Заикин Олег Сергеевич — кандидат технических наук, научный сотрудник, Новосибирский государственный университет, Новосибирск, 630090, Российская Федерация; ведущий научный сотрудник, Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук, Иркутск, 664033, Российская Федерация, sc 56786079600, https://orcid.org/0000-0002-0145-5010, oleg.zaikin@icc.ru

Authors

Vadim V. Davydov — PhD, Cryptography Researcher, QApp, Moscow, 121205, Russian Federation; Associate Professor, Saint Petersburg State University of Aerospace Instrumentation (SUAI), 190000, Saint Petersburg, Russian Federation; Scientific Researcher, ITMO University, 197101, Saint Petersburg, Russian Federation, sc 57203909696, https://orcid.org/0000-0002-5544-2434, vadimdavydov@outlook.com

Michail D. Pikhtovnikov — Student, Southern Federal University, Taganrog, 347922, Russian Federation, https://orcid.org/0009-0000-2173-0844, pikhtovnikov347@mail.ru

Anastasia P. Kiryanova — PhD Student, ITMO University, 197101, Saint Petersburg, Russian Federation, https://orcid.org/0009-0006-0344-5111, anastaciakosanovskaya@gmail.com

Oleg S. Zaikin — PhD, Researcher, Novosibirsk State University, Novosibirsk, 630090, Russian Federation; Leading Researcher, Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of RAS, Irkutsk, 664033, Russian Federation, sc 56786079600, https://orcid.org/0000-0002-0145-5010, oleg.zaikin@icc.ru

Статья поступила в редакцию 18.02.2025
Одобрена после рецензирования 21.04.2025
Принята к печати 27.05.2025

Received 18.02.2025
Approved after reviewing 21.04.2025
Accepted 27.05.2025