О ВЛИЯНИИ КРИПТОГРАФИЧЕСКОЙ СТОЙКОСТИ ФУНКЦИЙ ХЕШИРОВАНИЯ НА УСТОЙЧИВОСТЬ СОВРЕМЕННЫХ БЛОКЧЕЙН-ЭКОСИСТЕМ И ПЛАТФОРМ

Ищукова Е. А.[1]

DOI: 10.21681/2311-3456-2025-3-63-71

Целью настоящей работы *является систематизация знаний по функциям хеширования современных блокчейн-экосистем и платформ, а также определение криптографической стойкости упомянутых функций с точки зрения времени, затрачиваемого на проведение криптоанализа.*

Методы исследования *основываются на использовании теории информации, теории устойчивости, теории криптографии и криптоанализа, математического аппарата теории вероятностей и математической статистики, технологии блокчейн, технологиях обеспечения киберустойчивости и информационной безопасности.*

Результаты: *в работе рассмотрены основные бесключевые криптографические примитивы, применяемые в современных блокчейн-системах – функции хеширования. Для них рассмотрены подходы к определению криптографической стойкости с точки зрения вычислительных затрат по отношению ко времени применения тактики полного перебора. Рассмотрено пять различных кейсов применения функций хеширования в составе блокчейн-систем и возможные сценарии атак на них.*

Практическая ценность *заключается в рассмотрении ряда кейсов, связанных с применением функций хеширования в современных блокчейн-системах. Для каждого кейса сформулирована постановка задачи, приведено возможное решение и дана оценка его сложности. Показано, что при правильном использовании функций хеширования обеспечивается достаточная стойкость блокчейн-систем, построенных на их основе. Также показано, что большинство встречающихся уязвимостей связано с ошибками реализации или применения функций хеширования внутри блокчейн-систем, а не со слабостью конструкций используемых функций.*

Ключевые слова: *киберустойчивость, блокчейн, криптографическая стойкость, алгоритм шифрования, функция хеширования, криптография, криптоанализ.*

Введение

Блокчейн технологии являются частным случаем построения систем распределенного реестра. Отличительной особенностью блокчейн технологий является выстраивание единого связанного списка, в котором каждая следующая запись зависит от предыдущей, а значит невозможно изменить какую-то запись в базе данных, не изменив все остальные, связанные с ней записи [1, 2]. Выстраивание единого связанного списка становится возможным за счет использования механизмов криптографии. Как правило, в качестве основных элементов любой блокчейн системы используется асимметричная криптография на эллиптических кривых (для формирования адресов в сети, формирования подписи, а также подтверждения своего права совершать те или иные операции в сети) и функции хеширования (для проверки целостности отдельных записей в системе, проверки целостности блоков, упаковки транзакций в блок, а также в случае необходимости для выстраивания цепочки блоков в соответствии с консенсусом).

Именно поэтому функции хеширования являются объектом пристального внимания ученых, которые изучают их стойкость, моделируя процессы хеширования и пытаясь выявить возможные уязвимости, а также оценить влияние таких уязвимостей на стабильность работы блокчейн системы в целом.

Рассмотрим кратко мировой опыт по анализу наиболее известных функций хеширования. Наибольшее количество работ посвящено анализу функции хеширования SHA-1, которая является предшественницей семейства функций хеширования SHA-2. Так, в 2005 году группа китайских ученых предложила алгоритм поиска коллизий, который оказался в 2000 раз быстрее, чем полный перебор всех возможных комбинаций [3]. В работе [4] рассмотрена успешная атака на алгоритм SHA-1. Авторы нашли способ подделать два разных PDF-документа с одинаковым хешем SHA-1, которые отображают разное произвольно выбранное визуальное содержимое. При этом сами авторы отмечают, что смогли обнаружить эту коллизию, объединив множество специальных криптоаналитических методов сложными способами. Также имеются результаты по дифференциальному анализу функции SHA-1 [5] и ее упрощенных версий [6]. Для текущих используемых функций хеширования

1 Ищукова Евгения Александровна, кандидат технических наук, ведущий научный сотрудник Научного центра информационных технологий и искусственного интеллекта НТУ Сириус. Россия. E-mail: ischukova.ea@talantiuspeh.ru, ORCID 0000-0002-6818-1608.

семейств SHA-2 и SHA-3 в открытой публикации сведения об успешном применении анализа отсутствуют [2]. Наиболее успешной атакой на сегодняшний день является применение метода дифференциального криптоанализа к функции хеширования SHA-256, сокращенной до 46 раундов [7].

Наряду с исследованием структуры функций хеширования с применением различных крипто аналитических техник, изучаются и другие вопросы: скорость работы в зависимости от реализации и используемой аппаратной части, возможность применения в устройствах специального назначения (например, IoT), а также другие вопросы. Так, в работе [8] исследуется влияние выбора другой хэш-функции на общую производительность блокчейна на примере платформы Ethereum. Авторы показали, что производительности блокчейна могут существенно зависеть от используемой хэш-функции. В работе [9] авторы предлагают новое семейство легковесных хэш-функций, предназначенных для приложений IoT в здравоохранении. Похожее исследование представлено в статье [10], где в качестве хеш-функции для систем блокчейн-IoT используется функция HashLEA. Исследование различных аспектов применимости функций хеширования в составе блокчейн-систем также рассматривается в ряде других работ [11–19].

Проведенный анализ показывает, что вопрос надежности и безопасности использования функций хеширования в составе блокчейн-технологий является актуальным. В данной статье предлагается рассмотреть основные подходы к применению функций хеширования в составе блокчейн систем с точки зрения их надежности.

1. Постановка задачи

Целью работы является систематизация знаний о функциях хеширования, используемых в современных блокчейн платформах, в том числе определение их криптографической стойкости с точки зрения времени, затрачиваемого на проведение анализа.

Для достижения поставленной цели необходимо:

1. Дать краткую характеристику для современных функций хеширования и подходов к их анализу, определить назначение их применения внутри блокчейн-систем.
2. Рассмотреть подходы к определению криптографической стойкости для каждого из случаев применения с точки зрения вычислительных затрат по отношению ко времени применения тактики полного перебора.

2. Объект исследования

Под функцией хеширования h = H(M) мы понимаем некоторое необратимое легко вычислимое математическое преобразование H() из сообщения M произвольной длины в сообщение фиксированной длины h. В силу того, что пространство возможных входов в функцию хеширования бесконечно, а пространство выходных значений конечно и ограничено размерностью выходного хеш-сообщения, неизменно будет существовать бесконечное множество коллизий. То есть таких ситуаций, при которых два различных значения M и M1 на входе функции H() приведут к образованию одного и того же выходного значения h = H(M) = H(M1). Криптографически стойкой функцией хеширования считают такую функцию хеширования, для которой не существует эффективного поиска коллизий. При этом различают два вида стойкости при поиске коллизий. Стойкость в слабом смысле сводится к тому, что для заданного сообщения M вычислительно трудно подобрать второе сообщение M1, имеющего такое же хеш-значение h = H(M) = H(M1). Стойкость в сильном смысле сводится к тому, что вычислительно трудно подобрать два произвольных сообщения M и M1, обладающих одним и тем же хеш-значением h = H(M) = H(M1).

Различают ключевые и бесключевые функции хеширования. Как правило, ключевые функции хеширования строятся на основе какого-либо симметричного алгоритма шифрования. В блокчейн-системах обычно используются бесключевые алгоритмы хеширования преимущественно для контроля целостности транзакций и блоков, а также иногда для обеспечения корректной работы механизма выстраивания блоков в цепочку (например, в случае с механизмом консенсуса «доказательство работы» от англ. proof-of-work). Кроме того, используются надстройки над алгоритмами хеширования. Например, схема вычисления HMAC используется для вычисления секретного ключа шифрования из стартовой сид-фразы или вычисления дочерних ключей из мастер-ключа, а схема хеширования с использованием дерева Меркля используется для упаковки выбранных транзакций в блок [2].

3. Подходы к определению стойкости функций хеширования в составе блокчейн систем

Как уже было отмечено ранее, функции хеширования бывают двух видов – ключевые и бесключевые. С точки зрения их применимости в современных блокчейн-системах, будем рассматривать в первую очередь бесключевые функции хеширования. По принципу построения все функции хеширования можно разделить на две категории: построенные на основе структуры Меркля-Дамгарда или построенные по принципу впитывающей губки. К наиболее часто используемым в современных блокчейн-системах можно отнести функции хеширования семейства RIPEMD, функции хеширования семейств SHA-2 и SHA-3. Для российских блокчейн-систем, построенных на основе отечественной

криптографии, используется функция хеширования ГОСТ Р34.11-2012. Так как объем статьи ограничен, то в настоящей работе ограничимся лишь краткой характеристикой по каждой из функций хеширования, применительно к которой будем рассматривать подходы к анализу. Детальное описание работы функций хеширования для всех упоминаемых семейств хеширования можно найти в работе [2].

Функция хеширования RIPEMD-160 построена на основе структуры Меркля-Дамгарда. За один раз она обрабатывает блок данных размерностью 512 бит в течение 80 раундов преобразований. На выходе образуется хеш-значение размерностью 160 бит.

Функция хеширования SHA-256 построена на основе структуры Меркля-Дамгарда. Хеширование выполняется блоками данных по 512 бит с обязательным дополнением последнего блока. Преобразование одного блока выполняется в течение 64 раундов преобразований. На выходе образуется хеш-значение размерностью 256 бит.

Функция хеширования Keccak-256 построена по принципу впитывающей губки. Функция хеширования Keccak лежит в основе семейства хеширования SHA-3. Важно отметить, что по сравнению с семейством SHA-3 оригинальный алгоритм Keccak имеет множество настраиваемых параметров (в то время как в стандарте SHA-3 эти параметры зафиксированы), а также для функции Keccak имеются отличия в том, как выполняется дополнение последнего блока хешируемых данных.

Для того чтобы одно и то же значение хешировалось по-разному, зачастую применяют так называемое «подсаливание» – добавление к исходной информации ничего не значащей последовательности, которая полностью изменит результат хеширования. Так, например, криптокошельки подсаливают сид-фразу, для того чтобы даже одинаковые сид-фразы давали разные секретные ключи в разных криптокошельках. При этом то, как выполняется подсаливание, полностью зависит от разработчика системы. Это может быть уникальное значение соли для каждого пользователя, а может быть глобальная соль, одинаковая для всех пользователей сервиса.

Рассмотрим наиболее известные подходы к анализу криптографических функций хеширования. Для функций хеширования задача анализа сводится либо к нахождению коллизии, либо к нахождению прообраза (восстановлению исходного сообщения).

Атака по словарю. С помощью данной атаки можно осуществлять как поиск коллизии, так и поиск прообраза. Составляется словарь с наиболее вероятными значениями, которые могут поступать на вход функции хеширования. Анализ выполняется поиском по составленной таблице. Успех анализа зависит от объема составленного словаря.

Атака полным перебором. Аналогична атаке по словарю. Только поиск ведется по всем возможным комбинациям символов для входного сообщения. Здесь сложность анализа возрастает с ростом длины входного сообщения.

Радужные таблицы используются для построения цепочек хеш-значений и, как правило, используются для поиска паролей.

Парадокс «дней рождений» заключается в том, что для получения из множества, содержащего N видов элементов, двух элементов одинакового вида требуется сделать O ($\sqrt{N}$) попыток. Благодаря парадоксу «дней рождения» атака методом полного перебора для поиска коллизий требует в $2^{n/2}$ раз меньше операций, чем для поиска прообраза.

Метод дифференциального криптоанализа рассматривает разность двух сообщений (определяемую с помощью операции сложения по модулю 2) и используется для поиска коллизий. Задача сводится к тому, чтобы определить наиболее вероятную разность dM для двух текстов M и M1 (dM = M xor M1), которая после применения функции хеширования h = H(M) и h1 = H(M1) на выходе преобразуется в разность, равную 0: dh = h xor h1 = 0, что фактически означает вариант наличия коллизии:

$$h = h1 = H(M) = h(M1).$$

Метод алгебраического криптоанализа применяется путем построения системы линейных уравнений, связывающих между собой биты исходного сообщения и биты вырабатываемой хеш-последовательности. На сегодняшний день активно развивается направление применения SAT-решателей к решению задач нахождения прообраза и поиска коллизий [20]. Кроме того, известны атаки на упрощенные хеш-функции с применением комбинаций методов дифференциального и алгебраического анализа [2].

Атака на основе использования слабости функции дополнения сообщения используется тогда, когда в хеш-функции отсутствует дополнение сообщения или в силу каких-то причин (например, неправильная программная реализация или слабость самой функции дополнения) дополнение выполняется неправильным образом. Рассмотрим упрощенный пример. Пусть функция хеширования за один раз обрабатывает блок размерностью 8 байт. Допустим, на вход поступает сообщение M = ABCD, которое по размеру меньше, чем обрабатываемый блок на входе функции хеширования. Такое сообщение будет дополнено какой-то битовой последовательностью, например, нулями в младших значащих разрядах до полного размера блока ABCD0000. В этом случае мы легко получаем коллизию. Так как и сообщение M = ABCD, и сообщение M1 = ABCD0000 приведут к одному и тому же хеш-значению.

DOI: 10.21681/2311-3456-2025-3-63-71

Атака удлинением сообщения заключается в том, что для функций хеширования, построенных на основе структуры Меркля-Дамгарда, существует возможность изменить хеш-значение, добавив дополнительный блок информации в конец исходного сообщения. При этом пользователю даже не обязательно знать исходное сообщение. Достаточно иметь выходное хеш-значение, которое будет использовано для инициализации стартовых регистров.

Метод встречи посередине можно применить в том случае, когда процесс хеширования можно разбить на независимые итерации. Или, например, в случае, когда выполняется двойное хеширование друг за другом. В этом случае процессы хеширования можно разделить и выполнять параллельно, после чего производить поиск совпадений.

Рассмотрим несколько кейсов, связанных с применением функций хеширования в современных блокчейн-технологиях. Для каждого из случаев определим возможный сценарий атак и оценим сложность анализа.

Кейс № 1.

Описание задачи: В разных блокчейн-платформах транзакции упаковываются по-разному. Однако при этом, как правило, для упаковки транзакций используется принцип хеширования с использованием дерева Меркля. При вычислении хеш-значения блока используются не сами транзакции, а только корень созданного дерева Меркля. Рассмотрим пример такого блока в случае, когда используется механизм консенсуса, не связанный с вычислением хеш-значения (отличный от PoW).

Постановка задачи: Блок данных M состоит из заголовочной информации и корня дерева Меркля RM, образованного транзакциями блока T1, T2, T3 ...TN. Для вычисления корня дерева Меркля используется функция хеширования h = H(Ti). Необходимо заменить в блоке M одну транзакцию так, чтобы корень дерева Меркля остался неизменным. При этом должны быть соблюдены все правила составления транзакций и блока.

Решение:

Вход: Транзакции T1, T2, T3...TN. Функция хеширования h = H(T). Корень дерева Меркля RM.

Выход: Транзакции T1, T2, T3...TNew.

Пример построения дерева Меркля для различного количества входных транзакций показан на рис. 1. Отличительной особенностью алгоритма является удвоение последнего хеш-значения, если количество листьев в обрабатываемом уровне дерева является нечетным. В современных блокчейн платформах ведется проверка на дублирование транзакций в блоке, чтобы избежать уязвимости CVE-2012-2459. Из рис. 1 наглядно видно, что самым простым вариантом решения поставленной задачи является поиск коллизии для самого нижнего уровня дерева Меркля. Только в этом случае все дальнейшие преобразования будут выполняться одинаково.

В случае нечетного количества транзакций задача сводится к замене последней транзакции TN на транзакцию TNew таким образом, чтобы выполнялось соотношение H(TN||TN) = H(TNew||TNew).

В случае четного количества транзакций задача сводится к замене последней транзакции TN на транзакцию TNew таким образом, чтобы выполнялось соотношение H(TN – 1||TN) = H(TN – 1||TNew).

В обоих случаях перебору (подбору) подлежит одна из транзакций. Основная проблема заключается в том, что новая транзакция TNew должна быть построена по всем правилам, которые предъявляются данной блокчейн системой к построению транзакций. Это означает, что в данном случае нельзя перебирать все возможные двоичные комбинации для второй части хешируемого сообщения. В данном случае можно применить несколько техник анализа.

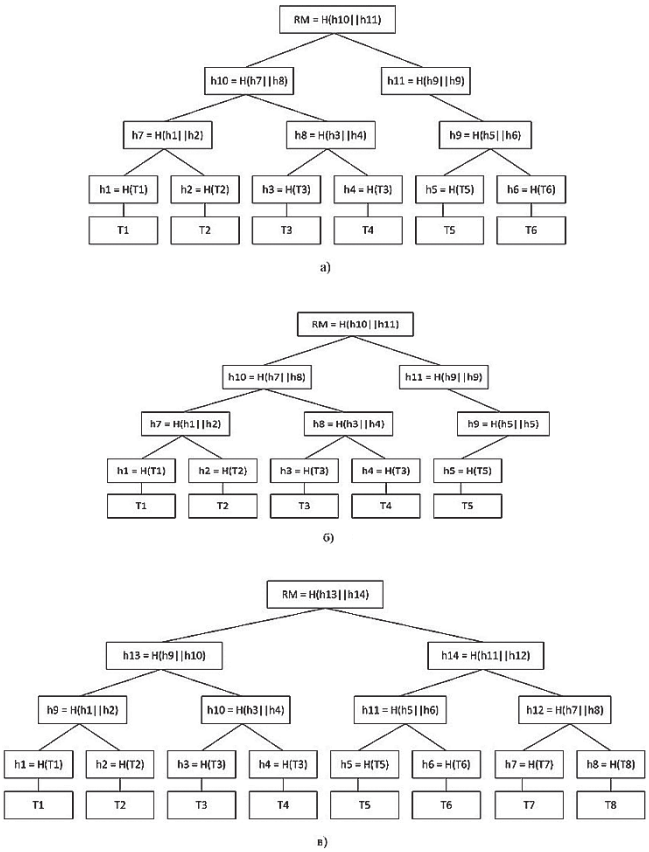

Рис. 1. Пример построения дерева Меркля

Вариант 1. Формирование новой транзакции TNew. Вычисление хеш-значения H(TNew||TNew) или H(TN – 1||TNew). Сравнение полученного хеш-значения с H(TN||TN) или H(TN – 1||TN) соответственно. Сложность анализа будет зависеть от длины n вырабатываемой хеш-последовательности и в среднем составит 2n/2 шагов.

Вариант 2. В случае четного количества транзакций можно попробовать применить метод дифференциального криптоанализа для случая, когда разность входов составляет dT = 0|| (TN xor TNew), а разность выходов dH = 0. Этот вариант, возможно, найдет решение в будущем, так как в настоящий момент сведений о применении метода дифференциального криптоанализа к полнораундовым алгоритмам SHA-1 и SHA-2 нет.

Кейс № 2.

Описание задачи: Для алгоритма консенсуса Proof-of-Work главным критерием выстраивания цепочки блоков является подбор такого значения nonce, при котором результат хеширования блока образует заданное количество нулевых бит в старших разрядах. Транзакции, как и в предыдущем случае, упаковываются с помощью дерева Меркля и при вычислении хеш-значения блока обычно учитывается только корень дерева Меркля. При этом изменение количества и состава транзакций может оказывать влияние на заголовочную информацию в блоке.

Постановка задачи: Блок данных M состоит из заголовочной информации IV, корня дерева Меркля RM и значения nonce. Для вычисления хеша блока используется функция хеширования h = H(M). Необходимо заменить блок M на блок M1 с соблюдением всех правил составления блока (в зависимости от блокчейн-платформы). Например, убрать или добавить одну или несколько транзакций, заменить в транзакции скрипт погашения и т.д. При этом хеш-значение нового блока M1 должно совпадать с хеш-значением блока M.

Решение:

Вход: Блок M = (IV||RM||nonce). Функция хеширования h = H(T).

Выход: Блок M1 = (IV1||RM1||nonce1), такой что h(M) = h(M1)

Данный кейс сводится к задаче поиска коллизии. В отличие от предыдущего кейса, здесь нет необходимости сохранять корень дерева Меркля неизменным. Достаточно переопределить набор транзакций в блоке и стартовую информацию блока (IV1||RM1). После чего необходимо перебирать значение nonce1 до тех пор, пока не будет получена коллизия h(IV||RM||nonce) = h(IV1||RM1||nonce1). Сложность анализа будет зависеть от длины n вырабатываемой хеш-последовательности и в среднем составит 2n/2 шагов.

Кейс № 3.

Описание задачи: В публичных блокчейн-сетях распространена практика, когда публичные ключи пользователей не хранят в системе в открытом виде. Считается, что публичный ключ можно использовать для вычисления приватного ключа, несмотря на то что это является вычислительно сложной задачей. Для платформы Эфириум адрес пользователя в сети составляет 20 байт и вычисляется как хеш от публичного ключа алгоритма ECDSA. В качестве функции хеширования используется алгоритм Keccak-256. Общая схема выработки адреса для сети Эфириум приведена на рис. 2.

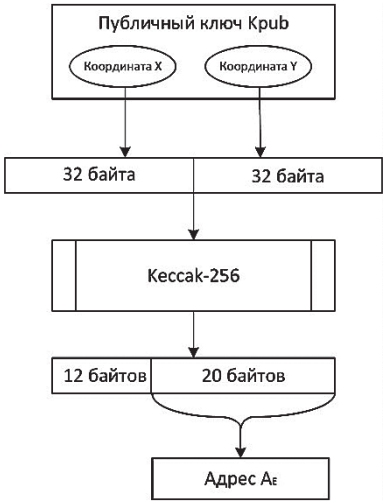

Рис. 2. Алгоритм выработки адреса в сети Эфириум

Постановка задачи: Известен адрес сети Эфириум AE. Определить публичный ключ пользователя.

Решение:

Вход: адрес сети Эфириум AE.

Выход: публичный ключ в виде координат точки Q = (x, y).

В данном случае задача сводится к поиску прообраза. Входными параметрами являются координаты точки Q = (x, y) для эллиптической кривой ECDSA. Забегая вперед (подробнее задачи, связанные с асимметричной криптографией на эллиптических кривых, будут рассмотрены в следующем разделе), отметим следующий факт. Не все возможные значения x в диапазоне от 1 до 2^{256} будут принадлежать заданной эллиптической кривой. Определить, является ли заданное квадратичное уравнение разрешимым для x можно с использованием символа Якоби. Для тех значений x, которые принадлежат заданной эллиптической кривой, всегда будут существовать два значения y. В общем случае алгоритм поиска публичного ключа будет выглядеть следующим образом.

Алгоритм 1:

Для всех x от 1 до $2^{256} - 1$:

1. Определить значение символа Якоби.
2. Если символ Якоби не равен 1, то вернуться к шагу 1.
3. Определить значения y_1 и y_2.
4. Вычислить h = Keccak-256(x||y_1).

DOI: 10.21681/2311-3456-2025-3-63-71

5. Если младшие 20 бит h равны AE, то решение точка (x||y_1).
6. Вычислить h = Keccak-256(x||y_2).
7. Если младшие 20 бит h равны AE, то решение точка (x||y_2).

В общей сложности потребуется 2^{257} операций хеширования для вычисления прообраза точки Q = (x, y).

Кейс № 4.

Описание задачи: Для платформы Биткоин адрес пользователя в сети составляет 25 байт и представляет собой кодировку Base58 от результата двойного хеширования публичного ключа алгоритма ECDSA с применением функций хеширования SHA-256 и RIPEMD-160 с добавлением 4 байтов контроля целостности, которые вырабатываются от старших 21 байта адреса путем двойного хеширования с использованием алгоритма SHA-256. Общая схема выработки адреса для сети Биткоин приведена на рис. 3.

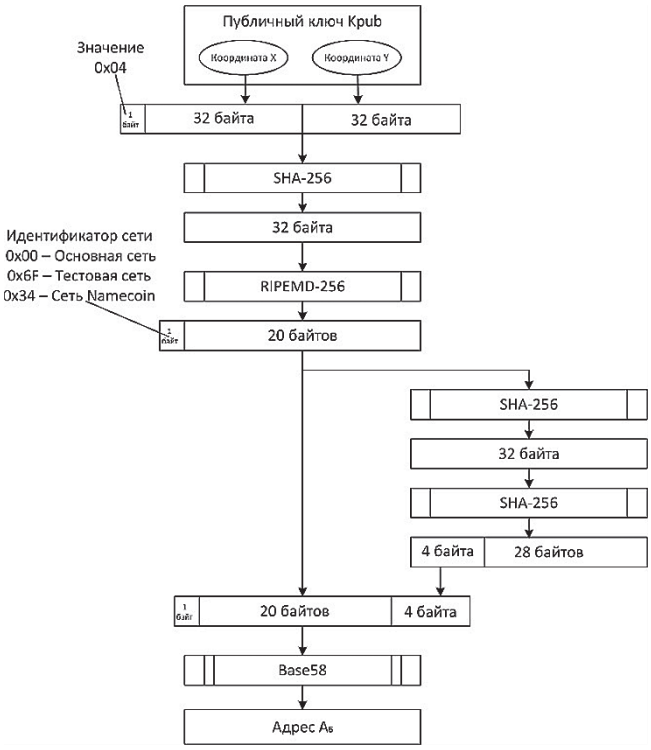

Рис. 3. Алгоритм выработки адреса в сети Биткоин

Постановка задачи: Известен адрес сети Биткоин АБ. Определить публичный ключ пользователя.

Решение:

Вход: адрес сети Биткоин $A_Б$.

Выход: публичный ключ в виде координат точки Q = (x, y).

В данном случае задача также сводится к поиску прообраза. Входными параметрами являются координаты точки Q = (x, y) для эллиптической кривой ECDSA. Первый байт устанавливается равным 0x04 всегда, что означает запись точки эллиптической кривой в развернутом виде. Аналогично предыдущему кейсу, не все возможные значения x в диапазоне от 1 до 2^{256} будут принадлежать заданной эллиптической кривой. Определить, является ли заданное квадратичное уравнение разрешимым для x можно с использованием символа Якоби. Для тех значений x, которые принадлежат заданной эллиптической кривой, всегда будет существовать два значения y. Прежде, чем приступать к поиску публичного ключа, необходимо преобразовать заданный адрес $A_Б$ в кодировку Base256. Обозначим старшие 21 байта адреса $A_Б$ как $MSB_{21}A_Б$, а младшие 4 байта адреса как $LSB_4A_Б$. Вычислить хеш-значение контрольной суммы: h = SHA-256(SHA-256($MSB_{21}A_Б$)). Сравнить младшие 4 байта полученного значения h с $LSB_4A_Б$. В случае совпадения этих значений, признать адрес правильным и приступить к поиску публичного ключа. В общем случае алгоритм поиска публичного ключа будет выглядеть следующим образом.

Алгоритм 2:

1. Для всех x от 1 до 2^{256} – 1:
   - 1.1. Определить значение символа Якоби.
   - 1.2. Если символ Якоби не равен 1, то вернуться к шагу 1.
   - 1.3. Определить значения y_1 и y_2.
   - 1.4. Вычислить h1 = SHA-256(x||y_1).
   - 1.5. Вычислить h2 = RIPEMD-160(h1).
   - 1.6. Если (00||h2) = $MSB_{21}A_Б$, то решение точка (x||y_1).
   - 1.7. Если (6F||h2) = $MSB_{21}A_Б$, то решение точка (x||y_1).
   - 1.8. Если (34||h2) = $MSB_{21}A_Б$, то решение точка (x||y_1).
   - 1.9. Вычислить h1 = SHA-256(x||y_2).
   - 1.10. Вычислить h2 = RIPEMD-160(h1).
   - 1.11. Если (00||h2) = $MSB_{21}A_Б$, то решение точка (x||y_2).
   - 1.12. Если (6F||h2) = $MSB_{21}A_Б$, то решение точка (x||y_2).
   - 1.13. Если (34||h2) = $MSB_{21}A_Б$, то решение точка (x||y_2).

В общей сложности потребуется 2^{258} операций хеширования для вычисления прообраза точки Q = (x, y).

Можно рассмотреть альтернативный вариант поиска на основе использования метода «встреча посередине». В этом случае необходимо создать базу предвычисленных значений, по которой будет осуществляется поиск. Сложность данному алгоритму будет добавлять необходимость хранения большой базы предвычисленных данных.

Алгоритм 3:

1. Для всех x от 1 до 2^{256} – 1:
   - 1.1. Определить значение символа Якоби.
   - 1.2. Если символ Якоби не равен 1, то вернуться к шагу 1.

1.3. Определить значения y_1 и y_2.
1.4. Вычислить h1 = SHA-256(x||y_1). Сохранить в Базе1 значения (x||y_1, h1).
1.5. Вычислить h2 = SHA-256(x||y_2). Сохранить в Базе1 значения (x||y_2, h2).
1.6. Вычислить h3 = RIPEMD-160(x).
1.7. Если (00||h3) = $MSB_{21}A_Б$, то сохранить в Базе2 значение x.
1.8. Если (6F||h3) = $MSB_{21}A_Б$, то сохранить в Базе2 значения x.
1.9. Если (34||h2) = $MSB_{21}A_Б$, то сохранить в Базе2 значения x.

2. Для всех x из Базы 2 провести сравнение со значениями h1 и h2 из Базы 1. В случае совпадения с h1 решением является точка (x||y_1) из советующей записи в Базе1. В случае совпадения с h2 решением является точка (x||y_2) из советующей записи в Базе1.

В случае использования Алгоритма 3 в общей сложности потребуется $3 * 2^{256}$ операций хеширования, а также около 2^{416} сравнений по базе. В худшем случае объем памяти, который потребуется для сохранения Базы 1 составит 2^{263}, что соответствует 10ПБ. Для сохранения Базы 2 потребуется еще дополнительно 2^{165} байт, что соответствует примерно 4ПБ. Исходя из конфигурации самого мощного суперкомпьютера мира по данным на ноябрь 2024 года (сайт top500.org), можно видеть, что в настоящее время задача поиска коллизий и прообразов является трудно вычислимой и практически не решаемой.

Выводы

В работе рассмотрен ряд кейсов, связанных с применением функций хеширования в современных блокчейн-системах. Для каждого кейса выполнено описание проблемы, сформулирована постановка задачи, приведено возможное решение и дана оценка его сложности. Показано, что при правильном использовании современных функций хеширования обеспечивается достаточная стойкость блокчейн-систем, построенных на их основе. Большинство встречающихся уязвимостей связано с ошибками реализации или ошибками применения функций хеширования внутри блокчейн-систем, а не со слабостью конструкций используемых примитивов.

Достоверность предлагаемого научного подхода подтверждается применением общенаучных методов исследования, достаточным информационным обеспечением, а также корректным применением методов криптографии, в том числе в построении формульных доказательств и выводов.

Результаты получены при финансовой поддержке проекта «Технологии противодействия ранее неизвестным квантовым киберугрозам», реализуемого в рамках государственной программы федеральной территории «Сириус» «Научно-технологическое развитие федеральной территории «Сириус» (Соглашение №23-03 от 27.09.2024 г.).

Литература

1. Satoshi Nakamoto Bitcoin: A Peer-to-Peer Electronic Cash System // https://www.ussc.gov/sites/default/files/pdf/training/annual-national-training-seminar/2018/Emerging_Tech_Bitcoin_Crypto.pdf
2. Ищукова Е.А., Панасенко С.П., Романенко К.С., Салманов В.Д. Криптографические основы блокчейн-технологий. – М.: ДМК Пресс, 2022. – 300 с.
3. Er-Rajy Latifa, El Kiram My Ahemed, El Ghazouani Mohamed, Achbarou Omar Blockchain: Bitcoin wallet cryptography security, challenges and countermeasures // Journal of Internet Banking and Commerce. – 2017. – V. 22. – n. 3.
4. Stevens, Marc & Bursztein, Elie & Karpman, Pierre & Albertini, Ange & Markov, Yarik. (2017). The First Collision for Full SHA-1. p. 570–596. DOI: 10.1007/978-3-319-63688-7_19.
5. A. Bakhtiyor, A. Orif, B. Ilkhom and K. Zarif, «Differential Collisions in SHA-1». In: 2020 International Conference on Information Science and Communications Technologies (ICISCT), Tashkent, Uzbekistan, 2020, pp. 1–5, DOI: 10.1109/ICISCT50599.2020.9351441.
6. Л.К. Бабенко, Е.А. Ищукова, Дифференциальный криптоанализ упрощенной функции хэширования SHA // Известия Южного федерального университета. Технические науки, 2010. – № 11. – С. 203 – 220.
7. Lamberger, Mario & Mendel, Florian. (2011). Higher-Order Differential Attack on Reduced SHA-256. IACR Cryptology ePrint Archive. 2011. 37.
8. Wang, Fuqin & Chen, Yijiang & Wang, Ruochen & Francis, Olusegun & Bugingo, Emmanuel & Zheng, Wei & Chen, Jinjun. (2019). An Experimental Investigation Into the Hash Functions Used in Blockchains. IEEE Transactions on Engineering Management. Vol. 67. No. 4. P. 1404–1424. DOI: 10.1109/TEM.2019.2932202.
9. Ramadan, Rabie A. and khalifa, Hany. S. and Dessouky, Mohamed and Aboshosha, Bassam W., Blockchain Technology for Enhanced Security of Iot Healthcare Devices: A Novel Lightweight Hash Function Approach and Secure Management System. Available at SSRN. http://dx.doi.org/10.2139/ssrn.4680105.
10. Sevin, A.; Osman Mohammed, A.A. Comparative Study of Blockchain Hashing Algorithms with a Proposal for HashLEA. Appl. Sci. 2024, 14(24), 11967. https://doi.org/10.3390/app142411967.
11. Cojocaru, A., Garay, J., Song, F. (2025). Generalized Hybrid Search with Applications to Blockchains and Hash Function Security. In: Chung, KM., Sasaki, Y. (eds) Advances in Cryptology – ASIACRYPT 2024. ASIACRYPT 2024. Lecture Notes in Computer Science, vol 15492. Springer, Singapore. P. 65-93. https://doi.org/10.1007/978-981-96-0947-5_3.

12. Fei Teng and Yong-zhen Li, Research on application of efficient hash function in blockchain technology, International Conference on High Performance Computing and Communication (HPCCE 2021), 2022. P. 121620Y. DOI: 10.1117/12.2628073.
13. Gençoğlu, M.Tuncay. (2022). Mathematical Analysis of The Hash Functions as a Cryptographic Tools for Blockchain. Turkish Journal of Science and Technology. 2022. Vol 17. Issue 2. p. 187–201. DOI: 17. 10.55525/tjst.1140811.
14. Alfaidi A., Semwal S. (2022) Privacy Issues in mHealth Systems Using Blockchain. Advances in Information and Communication. Vol. 438. P. 877–891. DOI: 10.1007/978-3-030-98012-2_61.
15. Wang, Maoning & Duan, Meijiao & Zhu, Jianming. (2018). Research on the Security Criteria of Hash Functions in the Blockchain. 47–55. DOI: 10.1145/3205230.3205238.
16. Fu, Jinhua & Qiao, Sihai & Huang, Yongzhong & Si, Xueming & Li, Bin & Yuan, Chao. (2020). A Study on the Optimization of Blockchain Hashing Algorithm Based on PRCA. Security and Communication Networks. 2020. Vol. 8. P. 1–12. DOI: 10.1155/2020/8876317.
17. F. Jahan, M. Mostafa, S. Chowdhury. Sha-256 in parallel blockchain technology: Storing land related documents. International Journal of Computer Applications. 2020. Vol. 175. No. 35. pp. 33–38. DOI:10.5120/ijca2020920911.
18. Z.A. Kamal, R.F. Ghani, R.F. Ghani A proposed hash algorithm to use for blockchain base transaction flow system. Original Research. 2021. Vol. 9. No. 4. pp. 657–673. DOI:10.13140/RG.2.2.31831.14249.
19. A.A.M.A. Ali, M.J. Hazar, M. Mabrouk, M. Zrigui Proposal of a Modified Hash Algorithm to Increase blockchain Security Procedia Computer Science. 2023. Vol. 225. pp. 3265–3275.
20. O. Zaikin Inverting Step-Reduced SHA-1 and MD5 by Parameterized SAT Solvers // 30th International Conference on Principles and Practice of Constraint Programming. – Leibniz International Proceedings in Informatics. – 2024. Volume 307, pp. 31:1–31:19. DOI: 10.4230/LIPIcs.CP.2024.31.

ON THE INFLUENCE OF CRYPTOGRAPHIC STABILITY OF HASHING FUNCTIONS ON THE STABILITY OF MODERN BLOCKCHAIN ECOSYSTEMS AND PLATFORMS

Ishchukova E. A.[2]

Keywords: *cyber resilience, blockchain, cryptographic strength, encryption algorithm, hashing function, cryptography, cryptanalysis.*

Purpose: *the aim of this work is to systematize knowledge on hashing functions of modern blockchain ecosystems and platforms, as well as to determine the cryptographic strength of the mentioned functions in terms of the time spent on cryptanalysis.*

Method: *Методы исследования основываются на использовании теории информации, теории устойчивости, теории криптографии и криптоанализа, математического аппарата теории вероятностей и математической статистики, технологии блокчейн, технологиях обеспечения киберустойчивости и информационной безопасности.*

Results: *the paper considers the main keyless cryptographic primitives used in modern blockchain systems – hashing functions. For them, approaches to determining cryptographic resistance are considered in terms of computational costs in relation to the time of applying the exhaustive search tactic. Five different cases of using hashing functions in blockchain systems and possible attack scenarios on them are considered.*

The scientific novelty *lies in the consideration of a number of cases related to the use of hashing functions in modern blockchain systems. For each case, a description of the problem is provided, a statement of the task is formulated, a possible solution is given and an assessment of its complexity is given. It is shown that with the correct use of hashing functions, sufficient stability of blockchain systems built on their basis is ensured. Most of the vulnerabilities encountered are associated with errors in the implementation or application of hashing functions within blockchain systems, and not with the weakness of the designs of the functions used.*

References

1. Satoshi Nakamoto Bitcoin: A Peer-to-Peer Electronic Cash System // https://www.ussc.gov/sites/default/files/pdf/training/annual-national-training-seminar/2018/Emerging_Tech_Bitcoin_Crypto.pdf
2. Ishchukova E.A., Panasenko S.P., Romanenko K.S., Salmanov V.D. Kriptograficheskie osnovy blokchejn-tehnologij. – M.: DMK Press, 2022. – 302 s.
3. Er-Rajy Latifa, El Kiram My Ahemed, El Ghazouani Mohamed, Achbarou Omar Blockchain: Bitcoin wallet cryptography security, challenges and countermeasures // Journal of Internet Banking and Commerce. – 2017. – V. 22. – n. 3.
4. Stevens, Marc & Bursztein, Elie & Karpman, Pierre & Albertini, Ange & Markov, Yarik. (2017). The First Collision for Full SHA-1. p. 570–596. DOI: 10.1007/978-3-319-63688-7_19.

2 Evgeniya A. Ishchukova, Ph.D. (of Tech.), Leading researcher of the Scientific Center of Information Technologies and Artificial Intelligence of NTU Sirius. E-mail: ischukova.ea@talantiuspeh.ru, ORCID 0000-0002-6818-1608.

5. A. Bakhtiyor, A. Orif, B. Ilkhom and K. Zarif, «Differential Collisions in SHA-1», 2020 International Conference on Information Science and Communications Technologies (ICISCT), Tashkent, Uzbekistan, 2020, pp. 1-5, doi: 10.1109/ICISCT50599.2020.9351441.
6. L. K. Babenko, E. A. Ishhukova, Differencial'nyj kriptoanaliz uproshhennoj funkcii hjeshirovanija SHA // Izvestija Juzhnogo federal'nogo universiteta. Tehnicheskie nauki, 2010. – № 11. – S. 203 – 220.
7. Lamberger, Mario & Mendel, Florian. (2011). Higher-Order Differential Attack on Reduced SHA-256. IACR Cryptology ePrint Archive. 2011. 37.
8. Wang, Fuqin & Chen, Yijiang & Wang, Ruochen & Francis, Olusegun & Bugingo, Emmanuel & Zheng, Wei & Chen, Jinjun. (2019). An Experimental Investigation Into the Hash Functions Used in Blockchains. IEEE Transactions on Engineering Management. PP. 1–21. DOI: 10.1109/TEM.2019.2932202.
9. Ramadan, Rabie A. and khalifa, Hany. S. and Dessouky, Mohamed and Aboshosha, Bassam W., Blockchain Technology for Enhanced Security of Iot Healthcare Devices: A Novel Lightweight Hash Function Approach and Secure Management System. Available at SSRN: https://ssrn.com/abstract=4680105 or http://dx.doi.org/10.2139/ssrn.4680105
10. Sevin, A.; Osman Mohammed, A. A. Comparative Study of Blockchain Hashing Algorithms with a Proposal for HashLEA. Appl. Sci. 2024, 14, 11967. https://doi.org/10.3390/app142411967.
11. Cojocaru, A., Garay, J., Song, F. (2025). Generalized Hybrid Search with Applications to Blockchains and Hash Function Security. In: Chung, KM., Sasaki, Y. (eds) Advances in Cryptology – ASIACRYPT 2024. ASIACRYPT 2024. Lecture Notes in Computer Science, vol 15492. Springer, Singapore. https://doi.org/10.1007/978-981-96-0947-5_3.
12. Fei Teng and Yong-zhen Li, Research on application of efficient hash function in blockchain technology, International Conference on High Performance Computing and Communication (HPCCE 2021), 2022 10.1117/12.2628073.
13. Gençoğlu, M.Tuncay. (2022). Mathematical Analysis of The Hash Functions as a Cryptographic Tools for Blockchain. Turkish Journal of Science and Technology. 17. 10.55525/tjst.1140811.
14. Alfaidi ASemwal S(2022)Privacy Issues in mHealth Systems Using BlockchainAdvances in Information and Communication. DOI: 10.1007/978-3-030-98012-2_61877-891 Online publication date: 8-Mar-2022 https://doi.org/10.1007/978-3-030-98012-2_61.
15. Wang, Maoning & Duan, Meijiao & Zhu, Jianming. (2018). Research on the Security Criteria of Hash Functions in the Blockchain. PP. 47–55. DOI: 10.1145/3205230.3205238.
16. Fu, Jinhua & Qiao, Sihai & Huang, Yongzhong & Si, Xueming & Li, Bin & Yuan, Chao. (2020). A Study on the Optimization of Blockchain Hashing Algorithm Based on PRCA. Security and Communication Networks. 2020. PP. 1–12. DOI: 10.1155/2020/8876317.
17. F. Jahan, M. Mostafa, S. Chowdhury. Sha-256 in parallel blockchain technology: Storing land related documents Int. J. Comput. Appl., 175 (35) (2020), pp. 33–38.
18. Z. A. Kamal, R. F. Ghani, R. F. Ghani. A proposed hash algorithm to use for blockchain base transaction flow system Original Research, 9 (4) (2021), pp. 657–673
19. A. A. M. A. Ali, M. J. Hazar, M. Mabrouk, M. Zrigui Proposal of a Modified Hash Algorithm to Increase blockchain Security Procedia Computer Science, 225 (2023), pp. 3265–3275.
20. O. Zaikin Inverting Step-Reduced SHA-1 and MD5 by Parameterized SAT Solvers // 30th International Conference on Principles and Practice of Constraint Programming. – Leibniz International Proceedings in Informatics. – 2024.