ТЕХНИЧЕСКАЯ ЗАПИСКА MRL-00XX.0

Оценка безопасности криптовалюты и её анонимности в постквантовом мире

Адам Корбо (Adam Corbo)*, Митчел “Isthmus” Кравиц-Тайер (Mitchell Krawiec-Thayer)†,
Брэндон Дж. Гуделл (Brandon G Goodell)‡
Insight и Исследовательская лаборатория Monero (Monero Research Lab)
Сентябрь 2020

Аннотация

Мы анализируем некоторые свойства безопасности протокола Monero в контексте возможностей квантовых компьютеров и используемых ими алгоритмов, включая алгоритм Шнорра, алгоритм Гровера, алгоритм Саймона, квантовый дифференциальный криптоанализ и квантовый доступ к случайным оракулам. Нами подробно описаны несколько теоретических уязвимостей: возможность выведения приватных ключей на основе публичных ключей, возможность выведения ключей на основе одноразовых адресов, возможность выявления действительного входа в кольцевых подписях, нарушение сбалансированности транзакций, раскрытие сумм транзакций, связывание множества транзакций с одним и тем же адресом, расшифровка идентификаторов платежей, а также конфликты хешей. Нами также рассматриваются несколько возможных постквантовых настроек: использование криптографии на основе решёток, многовариантной криптографии, криптографии на основе хеширования, а также криптографии на эллиптических кривых с суперсингулярной изогенией. Среди альтернативных протоколов рассматриваются ZK-STARKs, MatRiCT, Raptor-512 и RingRainbow. На основе размера подписи/доказательства, размера публичных ключей, времени генерирования и верификации нами исследуются общие компромиссы и сравниваются возможные протоколы. Для заинтересованного читателя нами предлагается простой вариант реализации алгоритмов Шнорра, Гровера и Саймона на языке Python, а также результаты, полученные нами для коммерческого квантового аппаратного обеспечения.

`https://github.com/insight-decentralized-consensus-lab/post-quantum-monero`

1 Вступление

В основе безопасности существующих на сегодняшний день криптовалют, таких как Bitcoin, Ethereum, Monero и Zcash, лежит решение вычислительно сложных задач. К таким способам защиты принадлежит, как кажется, повсеместно используемый метод, основанный на сложности решения задачи дискретного логарифмирования, который применяется по крайней мере со времени публикации работы [1]. Все перечисленные криптовалютные протоколы как минимум частично опираются на сложность решения задачи дискретного логарифмирования, которая является базисом, обеспечивающим защиту на уровне классических компьютеров (см. работы [2], [3], [4] и [5]). Но существуют квантовые алгоритмы, позволяющие решить эту задачу за полиномиальное время при наличии достаточно производительного квантового аппаратного обеспечения. Технически эти алгоритмы можно использовать и с классическим компьютером, но это будет неэффективно при отсутствии доступа к такому аппаратному обеспечению.

Мы анализируем безопасность протоколов блокчейна в контексте нескольких квантовых алгоритмов, используя Monero в качестве практического примера. Отмеченные уязвимости, такие как возможность генерирования ключей, характерны для многих монет, в основе которых лежит криптография на эллиптических кривых, и было бы ошибочно считать их присущими исключительно Monero с точки зрения полученных результатов.

*`adamryancorbo@gmail.com`
†`Isthmus@getmonero.org` или `IsthmusCrypto@protonmail.com`
‡`surae@getmonero.org`

В течение нескольких последних десятилетий теория квантовых алгоритмов и методов стремительно развивалась. Мы считаем, что формальный анализ влияния на безопасность существующих протоколов, а также обсуждение долгосрочного ухода от использования протоколов, не защищённых от атак обладающего квантовыми мощностями злоумышленника, уже представляются запоздалыми. Настоящая техническая записка не может считаться исчерпывающей. Появляется всё больше квантовых алгоритмов и соответствующих возможностей. Кроме того, уже могут существовать методы, пока не известные широкой публике.

Криптографами были разработаны протоколы для использования с классическими компьютерами, которые в равной степени устойчивы к злонамеренному применению квантовых методов, и эти протоколы могут быть адаптированы для Monero. У каждого подхода есть свои преимущества, недостатки, и для каждого из них характерны сложности, связанные с требуемым пространством/временем. Некоторые из них описываются нами в Разделе 5.

Рассуждения на тему, появятся ли когда-нибудь квантовые компьютеры в реальности, и если да, то когда это произойдёт, не являются темой этой технической записки. Насколько нам известно, в настоящее время не существует приемлемых постквантовых анонимных криптовалют, а это значит, что на уровне сегодняшней технологии можно говорить лишь о краткосрочной или среднесрочной финансовой анонимности. Первый протокол, который обеспечит анонимность при условии применения злоумышленником квантовых методов, займёт сильные позиции даже в предквантовом мире.

1.1 Предварительные условия, информация и терминология

Мы обозначаем сцепление строк как $||$. В случае с конечным неупорядоченным множеством X, где $|X| = n$, следует отметить, что мы можем произвольно определять элементы X и допускать, что $X = [n] = \{1, 2, \ldots, n\}$, без какого-либо ущерба для общности. Мы обозначаем криптографические хеш-функции, используя $\mathcal{H}$.

Мы говорим, что две транзакции являются *несвязываемыми*, если стороне, не являющейся получающей стороной, трудно определить, имеют ли две транзакции одно и того же получателя или нет. Мы говорим, что подписанта транзакции *определить нельзя*, если стороне, не являющейся отправляющей стороной, трудно определить отправителя транзакции. Мы называем транзакции *конфиденциальными*, если стороне, не являющейся получающей и не являющейся отправляющей стороной, трудно определить сумму транзакции.

Если мы говорим о *классическом злоумышленнике*, которого сокращённо обозначаем CA, то мы имеем в виду, что у такого злоумышленника есть доступ только к вероятностной машине Тьюринга. Если мы говорим о *квантовом злоумышленнике*, которого сокращёно обозначаем QA, мы имеем в виду, что такой злоумышленник имеет доступ к любому устройству (устройствам), способному эффективно выполнять квантовые алгоритмы и обладающему достаточной степенью масштабируемости, чтобы конкретно угрожать безопасности криптографических примитивов, лежащих в основе протокола Monero. Это не определяется каким-то волшебным количеством кубитов или какой-либо определённой конфигурацией; скорее, это касается способности выгодно использовать такие методы, как алгоритм Шнорра (см. работу [6]), алгоритм Гровера (см. работу [7]) и алгоритм Саймона (см. работу [8]), и делать это достаточно эффективно, чтобы обойти протокол Monero.

Во всех случаях мы предполагаем, что QA и CA имеют доступ к публичным данным блокчейна. В некоторых разделах мы отмечаем, что проведение атаки требует от злоумышленника наличия неполной базы данных адресов. Такие данные можно получить в результате утечек, путём сбора информации на основе открытых источников (OSINT), путём сговора с представителями бирж / предпринимателями или из любого другого источника.

2 Возможности квантового злоумышленника

В этом разделе нами приводится краткое и информативное описание процесса квантовых вычислений, а также некоторых возможностей QA.

Классический бит является двоичным числом, принимающим одно или два значения для указания логического значения. По определению этими значениями являются 0 и 1. С другой стороны, кубиты представляют собой множество логических значений, которые приводятся к классическому биту всякий раз, когда кубит взаимодействует со средой. Фактически измерение состояние кубита эквивалентно его приведению в классическое состояние. Перед измерением кубит остаётся в состоянии суперпозиции значений 0 и 1. Состояния кубитов являются элементами $\mathbb{C}^2$, и мы обозначаем их ортогональными базисными векторами $|1\rangle$ и $|0\rangle$. Суперпозиция кубита $|\psi\rangle$ представляется в виде линейной комбинации этих базисных векторов $|\psi\rangle = a_0 |0\rangle + a_1 |1\rangle$, где $a_0 \in \mathbb{C}$ является комплексной скалярной амплитудой состояния наряду с направлением $|0\rangle$ в $\mathbb{C}^2$, а a_1 является амплитудой по $|1\rangle$ в $\mathbb{C}^2$.

Амплитуды могут рассматриваться в качестве «квантовых вероятностей» - амплитуда наряду с ортогональным базисным вектором связана с вероятностью, согласно которой свёрнутое состояние будет соответствовать этому вектору. Другими словами, амплитуда по $|0\rangle$ связана с вероятностью, согласно которой свёрнутое состояние будет 0, а амплитуда по $|1\rangle$ связана с вероятностью, согласно которой свёрнутое состояние будет 1. Тем не менее следует быть осторожными. Амплитуды представлены сложными числами, в то время как традиционные вероятности описываются действительными числами. Точно так же, как вероятности в классической системе должны объединяться в 1 при наличии определённой меры вероятности μ, чтобы сформировать функцию распределения, квадраты величины амплитуд состояния в квантовой системе должны соответствовать $\int |a_0|^2 + |a_1|^2 \, d\mu = 1$.

2.1 Нарушение сложности задачи дискретного логарифмирования при помощи алгоритма Шора

Допустим, G является группой эллиптической кривой в некотором поле $\mathbb{F}$. Мы можем утверждать, что задача дискретного логарифмирования является вычислительно сложной для G, если соблюдается следующее условие.

Assumption 2.1 (Дискретные логарифмы в группах эллиптической кривой)**.** *Не существует такого алгоритма $\mathcal{A}$, который бы брал в качестве вводных данных пару $(g, h) \in G^2$, выбранную единообразно случайным образом, запускал полиномиальное время t и с не являющейся незначительной вероятностью ϵ выдавал такое некоторое значение $x \in \mathbb{F}$, чтобы $g^x = h$ или $g = h^x$.*

Алгоритм Шора, выполняемый на квантовом компьютере, может использоваться с некоторой не являющейся незначительной вероятностью ϵ за время t, являющееся полиномиальным для $|G|$, для решения соответствующей задачи, *задачи скрытой подгруппы*. В случае с конечной абелевой группой G, подгруппой $H \subseteq G$, конечным множеством X и функцией $f : G \to X$ мы можем утверждать, что f *скрывает* H если и только если любое значение $g_1, g_2 \in G$, $f(g_1) = f(g_2)$, и только если $g_1 H = g_2 H \in G/H$. Например, если установить X как фактор-группу $X = G/H$, мы увидим, как канонический групповой эпиморфизм $f : G \to G/H$ скрывает H. Мы всегда можем описать f, используя $O(\log |G| + \log |X|)$ бит. Довольно просто описать, как можно взломать дискретные логарифмы, решив задачу скрытой подгруппы. Если $G' = \langle g \rangle$ является какой-либо группой порядка p, мы можем вычислить дискретный логарифм предположительно случайного $h = g^x$ с неизвестным x, используя $G = \mathbb{F} \times \mathbb{F}$, $H = \langle (-x, 1) \rangle$ и функцию $f : G \to \mathbb{Z}_p$, определяемую преобразованием $(a, b) \mapsto g^a h^b$. Эта функция f является гомоморфизмом групп с ядром H; нахождение H эквивалентно вычислению дискретного логарифма для генератора $X = g^x$. Более подробное техническое описание того, как алгоритм Шора можно использовать для нарушения допуска сложности решения задачи дискретного логарифмирования, приводится в Разделе A.1. Более подробную информацию также можно найти в работах [9] и [10].

Пока что считается, что не существует какого-либо PPT-алгоритма, позволяющего нарушить Условие 2.1. Согласно работе [11] асимптотически нижней границей классических алгоритмов решения задачи дискретного логарифмирования является $(\frac{2}{3} + o(1))\sqrt{|G|}$ операций, в то время как среднее количество операций, необходимых для выполнения алгоритма Шора, составляет $O((\log |G|)^3)$, что экспоненциально быстрее. В случае с

группой G, где $|G| \geq 2^{256}$, СА требуется по крайней мере 2^{127} операций, однозначно трудновыполнимо. С другой стороны, существует некоторая постоянная c, в соответствии с которой QA требуется $c \cdot 2^{24}$ операций, вплоть до значений низшего порядка.

2.2 Неструктурированный поиск при помощи алгоритма Гровера и преобразов хешей

Некоторые аспекты безопасности зависят от того факта, что нахождение преобраза хеш-суммы является сложным. Для вычисления преобразов хеш-сумм можно использовать алгоритм Гровера, который находит неупорядоченные записи в базе данных, соответствующие критерию поиска, за время $O(\sqrt{n})$, где n является размером базы данных. Подробную информацию можно найти в работе [7] и Приложении А.2. Применение алгоритма Гровера является асимптотически оптимальным решением даже для злоумышленника, что описано в работе [12]. QA, использующий алгоритм Гровера, способен найти помеченное значение в неупорядоченном наборе данных. Несмотря на то, что это лишь квадратичное ускорение, решение является асимптотически оптимальным, так как любому квантовому алгоритму требуется $\Omega(\sqrt{n})$ времени для решения этой задачи (см. работу [12]). Одним из способов не допустить это является удвоение длины сообщений, используемых хеш-функцией, что затруднит выполнение такой задачи. Допустим, $f : [n] \to \{0, 1\}$ является функцией, описывающей, соответствует ли индекс критерию поиска, а также к какому оракулу имеет доступ алгоритм Гровера. Для любой неотрицательной функции $f : [n] \to \mathbb{Z}^+$, такой, чтобы $\sum_{x\in[n]} f(x) > 0$, следует отметить, что её неотрицательное значение подразумевает существование такого $w \in [n]$, что $f(w) > 0$. Более того, областью значений f является $\{0, 1\}$, поэтому $f(w) = 1$. Алгоритм Гровера делает приблизительно $O(\sqrt{n})$ запросов f, пользуясь доступом к оракулу, и выдаёт такое решение w, что $f(w) = 1$. При известной хеш-функции $\mathcal{H}$ мы можем определить $f(x)$ как 1, если $\mathcal{H}(x) = y$, и как 0 в противном случае. Затем алгоритм Гровера можно использовать для нахождения преобразов хеш-суммы, соответствующих необходимым параметрам.

Очень часто целое множество возможных преобразов сохраняется в открытом доступе в блокчейне, поэтому применение алгоритма Гровера в подобных случаях в течение времени $O(\sqrt{n})$ может быть очень быстрым. При этом алгоритм Гровера по-прежнему может применяться для нахождения произвольных преобразов, которые вовсе необязательно были опубликованы, но пространство поиска будет гораздо большим. В этом случае $O(\sqrt{n})$ также остаётся ограничивающим условием.

Техническое описание алгоритма Гровера приводится в Разделе А.2. Некоторые ограничения алгоритма Гровера описаны в работе [13], где демонстрируется, что ожидаемое количество запросов (по одному на итерацию), которые требуется сделать алгоритму к базе данных, составляет по крайней мере $\lfloor \sin(\pi/8)\sqrt{n} \rfloor$. Чтобы методом грубого перебора найти 256-битную не конфликтующую с другой хеш-сумму, по сути, пропуская каждый преобраз через хеш-функцию до тех пор, пока не всплывёт нужная хеш-функция, классическому компьютеру потребуется в среднем 2^{255} непрактичных операций. Такой вид атаки при области сообщений, равной примерно 2^{256}, представляется действительно непрактичным и односторонним подходом. С другой стороны, QA способен провести атаку методом поиска преобраза, используя алгоритм Гровера.

Для нахождения 256-битной хеш-суммы квантовому компьютеру, использующему алгоритм Гровера, потребуется сделать по крайней мере $\lfloor \sin(\pi/8)2^{128} \rfloor \approx 2^{126.6}$ запросов базе данных по одному на каждую итерацию. Даже если каждая итерация алгоритма Гровера будет происходить за постоянное время, составляющее самое большее одну пикосекунду, что является для нас до смешного щедрым допуском в отношении QA, поиск базы данных с 2^{256} записей займёт примерно 10^{26} секунд. Область будет существовать уже примерно 10^{17} секунд, поэтому такой подход будет более чем непрактичным. С другой стороны, поиск базы данных с 2^{64} записей требует менее одного миллиарда итераций, а поиск базы с 2^{32} записей - примерно $25,000$ итераций.

Очевидно, что алгоритм Гровера будет более эффективен, если использовать его с небольшой базой, а не осуществлять поиск по криптографически большому пространству. Фактически сокращение объёма записей в базе данных наполовину сокращает и время поиска на 75%. Если пространство поиска нельзя сократить в значительной мере, применение этого алгоритма требует больше времени, чем требуется алгоритму Шора для

взлома ECC, RSA и т. д.

Алгоритм Гровера также можно использовать вместе с другими методами для обнаружения преобразов хеш-суммы, например, вместе с методом квантового дифференциального криптоанализа (см. Раздел 2.3).

Monero использует хеш-функцию Keccak, которая в работе [14] и других работах характеризуется как безопасная с точки зрения проведения постквантовых атак. Тем не менее в работе [15] такая оценка оспаривается и говорится об атаках, проведённых на Keccak с более высоким уровнем воздействия на предполагаемую защиту, чем подразумевает данная функция. В интересах полноты исследования наши описания уязвимостей, характерных для Monero, включают в себя поверхность атаки, которую представляет собой уязвимая для квантовой атаки хеш-функция Keccak (даже несмотря на то, что мы надеемся на её постквантовую неуязвимость).

2.3 Алгоритм Саймона, квантовый дифференциальный криптоанализ и прочие возможности

Алгоритм Саймона, описанный в работе [8], может использоваться для выделения масок XOR из функций, под которыми они остаются неизменяемыми. Поскольку в основе многих хеш-функций лежат чередующиеся маски XOR, это делает возможным проведение дифференциального криптоанализа.

Допустим, $f : \{0,1\}^n \to \{0,1\}^n$ является любой функцией, которая будет неизменяемой под некоторой маской a. Другими словами, существует такая $a \in \{0,1\}^n$, что для каждого $x, y \in \{0,1\}^n$ будет $f(x) = f(y)$, если и только если $x \oplus y \in \{0, a\}$. При наличии у оракула доступа к f алгоритм Саймона делает $O(n)$ запросов оракулу и в качестве результата выдаёт маску a. Как и в случае с алгоритмом Гровера, это является оптимальным в том плане, что любой квантовый алгоритм должен сделать $\Omega(n)$ запросов. Это также происходит экспоненциально быстрее, чем в случае с любым классическим алгоритмом, которому потребовалось бы $\Omega(2^{n/2})$ запросов. Более подробную информацию можно найти в работе [16].

Жёсткое ограничение вероятности успеха использования в качестве функции количества запросов, сделанных оракулу f, приводится в работе [17], где предлагается следующее эвристическое решение: выполнение алгоритма Саймона в среднем считается успешным после $n + 3$ запросов, сделанных f, а вероятность успеха после $n + k$ запросов составляет примерно $1 - 2^{-k}$.

Техническое описание алгоритма Саймона приводится в Разделе A.3.

2.3.1 Квантовый дифференциальный анализ

Помимо методов атаки путём грубого перебора существуют и некоторые другие методы, которые QA и CA могут использовать, чтобы восстановить или раскрыть скрытую информацию, связанную с преобразом хеш-суммы, при помощи различных техник дифференциального криптоанализа, подобных тем, которые впервые были описаны в работе [18], и в реализации этих техник может помочь квантовый компьютер. Используя алгоритм Саймона, можно создать систему линейных уравнений, которые затем можно использовать для проведения дифференциального криптоанализа с целью расшифровки маски XOR, проведения атаки на симметричные примитивы ключей, как это описано в работе [19], или атаки на планы смены ключей, о чём говорится в работе [18]. Также в работе [20] рассматривается вариант реализации блочного шифра, устойчивого к дифференциальному криптоанализу.

Считается, что функция губки, используемая для создания Keccak, хеш-функции, используемой в Monero, устойчива к атакам, проводимым квантовым злоумышленником (более подробную информацию можно найти в работе [14]. Субъективно маловероятно, что нахождение преобразов функции губки будет происходить быстрее в случае применения алгоритма Гровера, по крайней мере, при значениях низшего порядка. Исключение могут составить случаи плохого выбора генерирования случайного преобразования. Следует отметить, что уже была представлена некоторая информация о возможности проведения атак по сторонним каналам (например, в работах [21] и [22], а позднее были предложены и способы избежать этого. В рамках конкурса NIST в работе [23] был предложен подход на базе алгоритма Гровера, позволяющий снизить сложность нахождения преобразов

Keccak (и многих других возможных хеш-функций SHA-3).

Если кратко подытожить, то проведение атак на Keccak методом грубого перебора при помощи алгоритма Гровера представляется трудной задачей даже для QA, за исключением определённых обстоятельств, когда речь идёт о небольших преобразах. Это также трудно реализуемо даже с точки зрения квантового дифференциального криптоанализа. Сужение области поиска преобраза делает такой вид атаки практичным если не для CA, то для QA точно, но всё же требует значительных затрат.

2.3.2 Квантовый доступ к случайным оракулам и метод Фиата-Шамира

Как модель случайного оракула, так и эвристический метод Фиата-Шамира подвергаются критике даже с точки зрения классических параметров (подробную информацию можно найти в работах [24], [25], [26] и [27]). В настоящей технической записке мы везде указываем на то, что эвристический метод Фиата-Шамира, позволяющий сделать интерактивные протоколы неинтерактивными, может быть улучшен для QA путём применения эвристического метода Унру, описанного в работе [26]. Это затратно в плане размера доказательства и времени верификации. При наличии квантового доступа к случайным оракулам квантовый компьютер с определённой версией хеш-функции может вычислить хеш суперпозиции экспоненциально большого количества входов, что позволяет определить суперпозицию хеш-сумм. Если кратко, QA может получить информацию о хеш-суммах экспоненциально многих выходов при наличии информации о полиномиальном количестве запросов. Доказательства безопасности, в основе которых лежит моделирование, при классических параметрах опираются на то, что «перемотка» не осуществляется напрямую. Например, если злоумышленник делает запрос с суперпозицией всех входов, случайный оракул нельзя смоделировать «на ходу» случайным бросанием монеты.

Тем не менее нами отмечается малая вероятность того, что эвристический метод Унру позволит осуществлять практические замены, которые можно будет использовать в Monero. Фактически подписи Monero являются кольцевым вариантом подписей Шнорра, описанных в работе [28] и являющихся протоколом интерактивной идентификации, который становится неинтерактивным при использовании эвристического метода Фиата-Шамира и который уже не считается устойчивым к проведению атак квантовым злоумышленником. А это означает, что подписи, подобные подписям Шнорра, можно подделать при наличии QA. Безусловно, фальсификация компонента Фиата-Шамира в этих подписях в целом требует больше квантовых ресурсов, чем вычисление дискретного логарифма ключа, который затем можно использовать для классического построения подписей по желанию.

В данной записке мы не перечисляем последствия получения квантового доступа к случайному оракулу, но отмечаем, что это может вызвать определённые проблемы, которые будет экспоненциально проще создать, чем в классическом случае (подробности изложены в работе [29]). В случае с уязвимостями, описанными ниже, нет никакой необходимости в подробной информации, касающейся случайного оракула.

3 Техническое описание уязвимостей

Если бы такой злоумышленник существовал, целый ряд фундаментальных механизмов Monero стал бы уязвим с точки зрения появления возможности вредоносной фальсификации. Ниже нами описано, как применение различных известных механизмов может повлиять на эти механизмы, и предлагаемый список не следует считать исчерпывающим.

В целях обеспечения криптографического контекста мы кратко характеризуем соответствующие криптографические механизмы, упрощая некоторые аспекты для ясности и сжатости изложения. Подробную спецификацию протокола Monero и описание его реализации читатель найдёт в работе [30].

3.1 Выведение приватных ключей кошелька из данных, сохранённых в блокчейне

В этих разделах мы объясняем, как QA может вывести ключи кошелька на основе публично доступной информации, такой как адреса, подадреса и другие данные, которые публично публикуются в блокчейне. В этом разделе предполагается, что у QA имеется база данных известных адресов.

Прежде всего, вспомним сам процесс создания ключей Monero. При наличии группы G порядка p с генератором g, процесс создания кошелька Monero подразумевает использование двух пар ключей: одной для траты и другой для просмотра. Тем не менее приватный ключ траты k_s используется для детерминированного вычисления всех остальных ключей кошелька Monero Core.

Приватный ключ траты $k_s \in \mathbb{Z}_{p-1}$ является случайным целым числом, выбираемым генератором псевдослучайных чисел. Публичный ключ траты K_s является групповым элементом, получаемым путём вычисления $K_s = g^{k_s}$. Мнемоническая фраза, состоящая из 25 слов и используемая для восстановления кошелька, это просто ключ k_s (с контрольной суммой), для удобства зашифрованный при помощи кодировки base-1626.

Генерируется приватный ключ просмотра k_v и вычисляется соответствующий публичный ключ $K_v = g^{k_v}$ В случае с кошельком Monero Core приватный ключ просмотра выводится на основе приватного ключа траты путём вычисления хеша $k_v = \mathcal{H}(k_s)$. Для проведения нашего анализа нет никакой необходимости в дальнейших подробностях, но следует отметить, что k_s и k_v могут независимо генерироваться при помощи генератора псевдослучайных чисел, что не позволит QA позднее воспользоваться этой связью для проведения атаки. Но в контексте данного раздела это бесполезно с точки зрения защиты.

3.1.1 Выделение ключей из первичного адреса

Первичный публичный адрес кошелька W является зашифрованным в кодировке base-58 прямым произведением сетевого префикса N, обоих публичных ключей K_v и K_s, и контрольной суммы C, а именно, $W = N||K_s||K_v||C$. Предположим, QA известен ваш основной адрес. Публичный ключ траты K_s и публичный ключ просмотра K_v могут быть выведены напрямую из адреса W, что позволит QA, которому известен любой адрес Monero, применить алгоритм Шора, чтобы вывести соответствующий приватный ключ траты k_s. Исходя из этого, CA может вычислить k_v (если k_v детерминировано выведен из k_s, как в случае с вариантом реализации кошелька Monero Core). Даже если пользователь не использует ключ просмотра k_v, детерминировано выведенный из k_s (например, вместо выбора двух независимых псевдослучайных чисел для k_s и k_v), адрес попрежнему будет уязвим для QA, который способен вычислить k_v, инвертировав преобразование $k_v \mapsto g^{k_v} = K_v$ повторно используя алгоритм Шора.

После этого злоумышленник, по сути, может завладеть кошельком: он может вывести остальные ключи, просмотреть всю историю кошелька, потратить любые средства и так далее. Monero не гарантирует какой-либо защиты от злоумышленника, которому удалось завладеть вашим приватным ключом. Поэтому даже публикация вашего публичного ключа (то есть вашего адреса) может оказаться опасной. Более того, подобное выведение ключей может произойти в любой момент в будущем, так как ключи будут взломаны задним числом, если кошелёк W будет найден позднее (например, в данных интернет-архива).

Одним из способов снизить вероятность взлома ключей является простое создание нового кошелька для каждой транзакции с полным опустошением предыдущего кошелька. Таким образом, вы сможете безопасно сообщать адреса кошельков отправителям, а не публиковать адрес в открытую. Очевидно, что это плохой способ. Атак, о которых говорится в данном разделе, можно в принципе избежать при помощи некоторой устойчивой к квантовой атаке схемы инкапсуляции ключей. Такой подход определённо в перспективе найдёт широкое применение в рамках протокола Monero, и пока совершенно не ясно, как этот протокол будет выглядеть в целом. Наиболее практичной альтернативой выглядит полный уход от использования инфраструктуры создания ключей на базе дискретного логарифма.

3.1.2 Выведение ключей на основе подадресов

Monero позволяет создавать множество подадресов для одного кошелька, а выходы для всех адресов могут расшифровываться при помощи основного приватного ключа просмотра кошелька k_s, но CA не сможет связать подадреса. Алгоритм Шора также позволяет извлечь ключи и из подадресов. Как в предыдущем разделе, предположим, что QA известен ваш подадрес, например, из данных OSINT или вследствие сговора с биржей.

Подадрес W_i, являющийся подадресом i^{th}, содержит публичный ключ траты $K_{s,i} := K_s \cdot g^{\mathcal{H}(k_v,i)}$, являющийся публичным ключом траты i^{th}, и публичный ключ просмотра $K_{v,i} := K_{s,i}^{k_v}$, также являющийся ключом i, где $\mathcal{H}$ является хеш-функцией. Таким образом, если QA становится известен подадрес $(K_{s,i}, K_{v,i})$, он может дважды использовать алгоритм Шора, чтобы вычислить $(k_{s,i}, k_{v,i})$, и применить этот алгоритм в третий раз, чтобы вычислить дискретный логарифм $K_{v,i}$ относительно $K_{s,i}$, получив таким образом k_v. Затем QA может воспользоваться классическим методом грубого перебора $k_s^{(i)} = k_{s,i} + \mathcal{H}(k_v, i)$ для каждого i в некотором малом диапазоне, чтобы найти предполагаемый приватный ключ траты $k_s^{(i)}$. С высокой вероятностью по крайней мере один из них будет являться ключом траты, соответствующим K_s, что можно проверить классическим методом возведения g в степень.

Кроме того, следует отметить, что подадреса используют хеш-функцию $\mathcal{H}$. В соответствии с моделью случайного оракула, $g^{\mathcal{H}(k_v,i)}$ является равномерно распределённым случайным элементом группы. Доказательства безопасности, основанные на моделировании, в схемах, использующих ключи на базе подадресов Monero, не могут считаться действительными, если существует возможность квантового доступа к случайному оракулу, несмотря на то, что мы так и не выявили какой-либо определённой уязвимости.

3.1.3 Выведение ключей на основе одноразовых адресов

Одноразовые ключи Monero вычисляются при помощи функции генерирования ключей $f_{otk} : \mathbb{Z} \times \mathbb{Z}_p \times \mathbb{G}^2 \to \mathbb{G}$, преобразующей (i, r, K_s, K_v) в одноразовый ключ $P = K_s g^{\mathcal{H}(K_v^r||i)}$.Предположим, у QA нет никакой информации, кроме публичных данных, сохранённых в блокчейне, то есть нет никакой базы данных ключей, собранной из открытых источников. В этом случае одного одноразового адреса P даже для QA будет недостаточно, чтобы вычислить приватный ключ траты k_s. Одной из классических особенностей Monero с точки зрения безопасности является то, что адрес можно безопасно использовать повторно. Это так благодаря одноразовым адресам Monero, которые не позволяют CA связывать транзакции с одним и тем же получателем или идентифицировать реальный адрес, стоящий за одноразовым адресом. Но то, как вычисляются одноразовые ключи Monero, может быть небезопасным при проведении атаки QA.

Одноразовые ключи Monero имеют форму $(R, P) = (g^r, g^p)$ для некоторого ключа транзакции $R = g^r$ и $P = K_s g^{\mathcal{H}((K_v)^r,i)}$. Дискретным логарифмом будет $p = k_s + \mathcal{H}((K_v)^r, i)$. В соответствии с моделью случайного оракула результатом применения $\mathcal{H}$ является случайная величина с равномерным распределением, и, таким образом, в рамках модели случайного оракула p прекрасно скрывает k_s при помощи $\mathcal{H}((K_v)^r, i)$.

Мы предполагаем, что QA собирается проанализировать отдельный одноразовый публичный ключ (R, P), который видит в блокчейне; мы можем предположить (без ущерба для общности), что QA не является отправителем данной транзакции, в противном случае ему уже был бы известен (K_s, K_v). получателя. Тот факт, что злоумышленник обладает квантовым доступом к случайному оракулу, не имеет значения, поскольку публикуется только один P с дискретным логарифмом $p = k_s + h$, где h является хеш-суммой. Поскольку как k_s, так и h не известны QA, он даже не сможет применить алгоритм Гровера, чтобы найти преобраз h (и, как упоминается в Разделе 2.2, применение алгоритма Гровера к хеш-сумме 256-битного входа занимает слишком много времени и не может считаться практичным решением).

3.1.4 Выведение ключей на основе выборки множества одноразовых адресов

Если QA получает контроль над значениями r, k_v и i в хеше, то вполне возможно использовать эту выборку из нескольких одноразовых адресов для построения варианта задачи скрытой подгруппы, уязвимой с точки

зрения применения алгоритма Шора. Скрытая подгруппа является ядром естественного эпиморфизма группы $\pi : \mathbb{G} \to \mathbb{G}/\langle K_s \rangle$. Нахождение скрытой подгруппы позволяет получить генератор K_s. Следовательно, если QA может получить выборку одноразовых ключей $P_1, \ldots, P_n \in \mathbb{G}$ для некоторого значения $n = O((\log |\mathbb{G}|)^3)$, то он сможет получить и K_s. Ещё одно применение алгоритма Шора даст k_s.

Тем не менее, если QA не контролирует r, k_v и i, то у него нет и доступа к случайному оракулу, необходимому для выполнения алгоритма Шора. Поэтому вероятность, с которой QA может получить K_s, не контролируя полностью эти значения в запросах оракула, неизвестна.

На данный момент мы не можем гарантировать безопасность повторного использования ключей с точки зрения возможности выведения ключей гипотетическим QA в будущем. Если на какой-нибудь адрес или подадрес (скажем, с парой ключей (K_s, K_v)) будет принято более одной транзакции, что отразится в истории блокчейна, то эти одноразовые ключи попадут в выборку одноразовых ключей, которую можно будет использовать, как описано выше. Поскольку абсолютно любой человек, которому известен ваш адрес, может отправить множество выходов на один из ваших адресов, то повторное использование ключей вполне может позволить QA вывести приватные ключи, и это будет вероятным до тех пор, пока не будет доказано обратное.

Следует отметить, что в соответствии с текущим вариантом реализации основного протокола Monero выход сдачи каждой транзакции отправляется с индексом 0 текущего счёта. Следовательно, отправка двух (или большего количества) транзакций на любой счёт со сдачей приведёт к повторному использованию адреса, независимо от количества входящих транзакций. Тем не менее этого можно избежать, если изменить поведение кошелька, например, зарезервировав поднабор подадресов для разового получения сдачи.

3.2 Нарушение неопределённости подписанта с использованием данных блокчейна

В этом разделе мы описываем, как QA может определить, кем из участников кольца была построена кольцевая подпись, используя только данные, сохранённые в блокчейне. Также в данном разделе не предполагается наличия у QA базы данных известных ключей или адресов.

Чтобы исключить возможность двойной траты, транзакции Monero требуют публикации всех образов действительных подписывающих ключей, используемых во всех кольцевых подписях, для транзакции в соответствии с односторонней функции. Взяв данные блокчейна, QA может воспользоваться кольцом публичных одноразовых ключей и связывающим тегом для выведения соответствующего приватного одноразового ключа.

Для каждого входа транзакции подписант включает публичный связующий тег J и кольцо одноразовых ключей выходов, допустим, $\{P_1, \ldots, P_n\}$, где n является размером кольца. На момент написания данной работы, протоколом предусматривал размер кольца $n = 11$. Подписанту сообщения известен приватный ключ p_π, соответствующий некоторому публичному ключу P_π одного из участников кольца, а также известны публичные ключи для других $n - 1$ ложных ключей, которые были выбраны подписантом из блокчейна.

Monero использует связующие теги $J := (\mathcal{H}(P))^p$. В соответстви с допуском дискретного логарифмирования, CA не может установить, какой из индексов кольца π соответствует ключу P_π, используемому подписантом для вычисления связующего тега J, несмотря на то, что он может вычислить $\mathcal{H}(P_i)$ для каждого i. С другой стороны, QA также может вычислить дискретный логарифм $\widehat{p}_i$ тега J относительно каждого $\mathcal{H}(P_i)$. Для некоторого индекса π, $J = (\mathcal{H}(g^{\widehat{p}_\pi}))^{\widehat{p}_\pi}$. QA приходит к заключению, что действительным подписывающим ключом был P_π и, приходя к такому заключению, QA узнаёт p_π.

Это, скорее, является неэффективным применением квантовых ресурсов, требующим множества выполнений алгоритма Шора. Если QA обладает достаточным количеством ресурсов, он может просто вычислить дискретный логарифм каждого нового P, когда он публикуется в блокчейне, при помощи алгоритма Шора, и может использовать его для вычисления соответствующего связующего тега J. Это снижает степень неопределённости подписанта Monero до анонимной группы с кардинальностью 1.

QA задним числом может сделать так, чтобы подписант всех транзакций, опубликованных в блокчейне, перестал быть скрытым. Необходимым, но недостаточным для защиты Monero от QA изменением является ис-

пользование связующих тегов, которые будет нельзя детерминировано вычислить в рамках задачи дискретного логарифмирования.

3.3 Нарушение баланса транзакций

В этом разделе предполагается, что QA заполучил несколько Monero и может строить транзакции по собственному желанию, а также, что он использует при этом вариант протокола Monero, где обязательства Педерсена по суммам не имеют детерминировано вычисляемых масок. Следует отметить, что вариант реализации Monero Core использует детерминировано вычисляемые маски, но не в обязательном порядке, что допускает возможность обмана.

На создание кольцевых конфиденциальных транзакций Monero в стиле описанных в работе [31] повлияли конфиденциальные транзакции в стиле Bitcoin (см. работу [32]), где обычные цифровые подписи заменяются кольцевыми. Сбалансированность таких транзакций доказывается при помощи доказательств диапазона сумм транзакций. В этом разделе мы продемонстрируем, как QA двумя способами может нарушить сбалансированность транзакций, а также опишем способы избежать обоих этих вариантов.

Если бы сообщество Monero узнало о существовании QA на практике до того, как избежать подобной атаки, было бы невозможно верифицировать денежную массу при помощи классического компьютера.

3.3.1 Атака: взлом масок в обязательствах Педерсена

В случае, если QA удастся заполучить сколько-нибудь Monero, он сможет произвольным образом манипулировать денежной массой Monero, взломав маски обязательств Педерсена. Суммы Monero шифруются совершенными с точки зрения сокрытия и обязательными к вычислению обязательствами Педерсена, описанными в работе [33] и используемыми как часть входящих ключей. Схема обязательств Педерсена использует две базовые точки, g_1 и g_2, дискретные логарифмы которых неизвестны относительно друг друга. В варианте реализации Core Monero $g_1 = g$ (то же, что и базовая точка публичного ключа), а $g_2 = \mathcal{H}(g)$ для некоторой хеш-функции $H : \{0,1\}^n \to G$. Чтобы создать обязательство по сумме b с маской y, мы вычисляем $C(y,b) = g_1^y g_2^b$. Чтобы раскрыть некоторое C, мы раскрываем (y,b) и проверяем, чтобы $C = C(y,b)$.

Следует отметить, что преобразование с использованием (y,b) для C является «многим к одному»: для любых (y,b) существует множество таких вариантов выбора $(y',b') \neq (y,b)$, чтобы $C(y',b') = C(y,b)$.

У CA есть несколько вариантов раскрытия некоторого $C = C(y,b)$ для некоторых $(y',b') \neq (y,b)$, но, чтобы методом грубого перебора найти любые (y',b'), придётся пойти на хитрость. Если CA работает в PPT, это является сложной задачей. В этом смысле схема обязательств Педерсена является вычислительно обязательной. Тем не менее QA может нарушить это, используя алгоритм Шора для вычисления дискретного логарифма g_2 относительно g_1 или, *наоборот*, скажем, $g_2 = g_1^\gamma = g^\gamma$. Чтобы раскрыть $C(y,b)$ для другого значения $b' \neq b$, QA может просто классическим образом вычислить $y' = y + \gamma(b - b')$. Затем $C = C(y,b) = C(y',b')$, но уже для $b \neq b'$.

Это может само собой разрешиться в последующую атаку. QA получает выход Monero с суммой b и некоторой маской y, которая является общим секретом между QA и предыдущим отправителем, то есть обоим известно, как открыть $C(y,b)$. QA решает раскрыть $C(y,b)$ по сумме $C(y',2b)$ для некоторой другой y', чтобы построить следующую транзакцию, таким образом удвоив свои деньги.

Поскольку QA известно такое значение γ, что $g_2 = g_1^\gamma$, и поскольку $C(y,b) = g_1^y g_2^b = g_1^{y+\gamma b}$, QA решает $y + \gamma b = y' + \gamma(2b)$ для $y' = y - \gamma b$. Тогда $g_1^{y'} g_2^{2b} = g_1^{y-\gamma b} g_2^{2b} = g_1^y g_2^{2b-b} = C(y,b)$, и QA может использовать y' и $2b$ для классического вычисления фиктивно действительной транзакции.

3.3.2 Способ защиты: ограничение масок

Маски сумм в обязательствах Педерсена в рамках реализации Monero Core вычисляются детерминировано при помощи хеш-функции. Это не позволяет QA выбрать случайную маску y' для раскрытия обязательства

при отправке третьей стороне.

По сути, маски Monero являются двойными хешами общего секрета Диффи-Хеллмана K_v^r, то есть $y = \mathcal{H}(\texttt{pre} \mid\mid \mathcal{H}(K_v^r, i))$ для префикса `pre`. QA может попытаться нарушить сбалансированность транзакции, как было описано в предыдущем разделе. Однако атака будет успешной только в том случае, если получатель откроет транзакцию с суммой, отличной от той, по которой давалось оригинальное обязательство со значением b, например, с отрицательным значением. У QA могут быть ключи жертвы (K_v, K_s), и он может применить алгоритм Гровера, чтобы найти преобраз для y', скажем, $y' = \mathcal{H}(x)$, где $x = \mathcal{H}(\texttt{pre} \mid\mid y)$. Затем QA может применить алгоритм Гровера во второй раз, чтобы найти преобраз для y в форме $y = \mathcal{H}(z \mid\mid i)$. Затем QA может вычислить r, чтобы z являлось представлением K_v^r в форме битовой строки для целевого K_v. Это, определённо, крайне неэффективно.

Вместо этого QA может просто вычислить (k_v, k_s) на основе известных ключей (K_v, K_s) полностью завладеть кошельком жертвы, а не пытаться протолкнуть транзакцию с ложным балансом. Несомненно, несмотря на то, что QA может попытаться провести атаку, описанную выше, квадратично быстрее, чем CA, повторное применение алгоритма Гровера для нахождения преобразов без каких-либо ограничений по пространству поиска делает этот подход с точки зрения времени вычислений гораздо более долгим, чем возраст вселенной. Это особенно точно в том случае, если преобраз детерминированных масок является таким длинным, как описано в Разделе 2.2. Тем не менее злоумышленник, проводящий атаку, может сделать это так, как описано в Разделе 3.3.1, просто отправляя средства самому себе и последовательно удваивая их (или более того) с каждой транзакцией. Он может раскрыть своё окончательное обязательство по желанию и в конечном счёте использовать обычную детерминированную маску, чтобы отправить только что созданные деньги третьей стороне.

Ещё один альтернативный вариант заключается в вычислении масок на основе верифицируемой случайной функции и включении доказательств с нулевым разглашением (соответствующей надёжности), согласно которым маски были вычислены правильно внутри транзакции. Преимущество данного подхода заключается в том, что он не даёт злоумышленнику последовательно множить свои средства при условии, что верифицируемая случайная функция устойчива к квантовым вычислениям.

Интересным направлением будущих исследований может стать поиск более быстрого квантового алгоритма, позволяющего нарушить обязательную составляющую обязательств Педерсена при помощи детерминировано вычисленных масок в рамках уязвимой для квантовых вычислений модели доступа к случайному оракулу.

3.3.3 Способ защиты: переключаемые обязательства

Денежную массу Monero можно эффективно защитить путём изменения протокола и использования переключаемых обязательств вместо обязательств Педерсена. Такие обязательства обеспечат надёжную защиту в случае появления квантового компьютера. Схема переключаемых обязательств была предложена в Работе [34]. Будучи основанными на обязательствах Эль Гамаля, переключаемые обязательства уже содержат обычные обязательства Педерсена и поэтому могут рассматриваться в качестве расширения той схемы обязательств, которую уже использует Monero. Таким образом, модификация протокола потребует внесения минимальных изменений.

Переключаемые обязательства являются гомоморфными обязательствами, основанными на обязательствах Эль Гамаля, и делают арифметические выступления, связанные с обязательствами по сумме, совместимыми с арифметическими вычислениями, связанными с суммами, выраженными простым текстом. В отличие от обязательств Педерсена раскрывающая сторона может убедить верификатора, что некоторое обязательство C соответствует определённому значению путём *частичного* или *полного* раскрытия. Если раскрывающая сторона раскрывает обязательство частично, схема является вычислительно обязательной и идеально скрывающей сумму. Если же раскрывающая сторона раскрывает обязательство полностью, схема является статистически обязательной и вычислительно скрывающей сумму.

Благодаря этим свойствам протокол Monero может быть модифицирован для использования переключаемых обязательств с частичным раскрытием до тех пор, пока появление квантовых компьютеров не станет

неизбежным, после чего протокол можно изменить так, чтобы обязательства раскрывались полностью. Такие изменения не являются тривиальными с точки зрения кодовой базы Monero, но представляют собой кратчайший путь к обеспечению защиты от этой конкретной уязвимости.

3.3.4 Атака: взлом доказательств диапазона Bulletproofs

Bulletproofs, система доказательства, используемая для построения доказательств диапазона в Monero, основана на условии сложности решения задачи дискретного логарифмирования. Интерактивный протокол Bulletproof, представленный в Работе [35], является идеальным доказательством с нулевым разглашением для честного верификатора и является совершенно полным. Следовательно, даже QA не сможет использовать транскрипт передаваемого доказательства Bulletproof, чтобы выделить секретные данные, к которым оно относится, и не сможет найти свидетельство того, что оно не прошло верификацию.

Тем не менее, насколько нам известно, Bulletproofs соответствует только вычислительной эмуляции с расширением свидетельства. Квантовый компьютер мог бы применить алгоритм Шора ко всем базовым точкам, используемым Bulletproofs, и выдать такое приемлемое доказательство Bulletproof, что эмулятор не смог бы выделить свидетельство. Более того, проблемы, связанные с использованием QA эвристического подхода Фиата-Шамира, подробно описанные в Работе [27], указывают на то, что при наличии QA недостатком Bulletproofs могут оказаться совершенно непредвиденные проблемы.

3.3.5 Способ защиты: устойчивые к квантовой атаке доказательства диапазона

Доказательства с нулевым разглашением, основанные на задачах, решение которых будет сложным даже для квантовых компьютеров, могут заменить собой Bulletproofs в качестве доказательств диапазона Monero, усилив тем самым Monero против QA. Так, например, в Работе [36] предлагается подход, основанный на использовании решёток. Тем не менее авторы указывают на то, что, насколько им известно, до сих пор не было предложено никаких многовариантных доказательств диапазона.

3.4 Нарушение свойства несвязываемости

В этом разделе мы допускаем, что у QA имеется база данных известных публичных адресов кошельков. То, о чём говорится в данном разделе, не следует путать с атаками, описанными в Разделе 3.1, когда злоумышленник, проводящий атаку, выводит ключи. В данном случае он желает связать одноразовые ключи, используемые общим получателем. Следует отметить, что злоумышленник может воспользоваться техниками, о которых говорится в Разделе 3.1, чтобы вывести приватные ключи на основе базы данных и связать транзакции напрямую, что подразумевает наличие последствий, описанных в настоящем разделе.

Одноразовые адреса Monero призваны исключить возможность повторного использования ключей, что позволило бы связать транзакции по получателю. Одноразовые адреса маскируют публичные ключи получателя. Эти одноразовые адреса могут быть сформированы либо из подадресов, либо основных адресов, и они вычисляются при помощи хеш-функций.

Используя комбинацию из алгоритма Шора и алгоритма Гровера, квантовый компьютер может раскрыть подлинные публичные ключи, стоящие за каждым одноразовым адресом, используемым в транзакции. Хитрость состоит в том, чтобы сузить область возможных значений преобразов, чтобы сократить размер сообщений, необходимых для итераций алгоритма Гровера.

Увидев новую транзакцию, транслируемую в сети Monero, скажем, с одноразовым ключом $P = K_s g^{\mathcal{H}(K_v^r||i)}$ и ключом транзакции $R = g^r$ для некоторых неизвестных K_s, K_v, r, i, QA может вычислить r, применив алгоритм Шора. Затем путём итерации через i QA может применить алгоритм Гровера, чтобы найти в базе данных любую пару ключей $(K_v^{(j)}, K_s^{(j)})$, соответствующую $P = K_s^{(j)} g^{\mathcal{H}((K_v^{(j)})^r||i)}$. Таким образом, QA может связать все транзакции, отправленные для одной и той же пары ключей, даже несмотря на то, что были использованы одноразовые ключи. Несмотря на то, что описанный подход является довольно медленным из-за того, что в

отношении каждой проводимой транзакции применяется алгоритм Шора, он всё же экспоненциально быстрее любого схожего подхода, который может использовать CA. Более того, этот подход безошибочен в том смысле, что либо он сработает, либо ключ получателя попросту будет отсутствовать в базе данных.

Это совсем не обязательно наилучший пример использования квантовых ресурсов. Если у QA имеется база данных ключей, вместо этого QA может выделить все имеющиеся у него ресурсы на вычисление всех соответствующих дискретных логарифмов. Как только ключ будет взломан, а QA, по сути, владел ключом, он сможет классическим способом проверить все входящие транзакции, как и любой другой пользователь Monero, не вычисляя логарифм ключа каждой проводимой транзакции.

Пользователи, всегда генерирующие новые кошельки, могут избежать проблем, связанных с несвязываемостью. Эта мера защиты демонстрирует, что одноразовые ключи Monero, предназначенные для решения проблем со связываемостью в Monero, не обеспечивают полной защиты от QA.

Полной замены процесса обмена ключами в соответствии с протоколом Диффи-Хеллмана на устойчивый к квантовым атакам процесс обмена ключами будет недостаточно, чтобы избежать угрозы, о которой говорится в этом разделе. Безусловно, поскольку общий секрет хешируется, это по-прежнему позволит QA применить алгоритм Гровера с целью нахождения преобраза путём поиска в известной базе данных, даже при большом размере ключей. Если база данных будет достаточно маленькой, QA по-прежнему довольно быстро сможет найти правильные решения всякий раз, когда их будет нужно найти. По этой причине способы защиты от атак, связанных с несвязываемостью, которые может совершить QA, будут найдены в области устойчивых к квантовым атакам схем одноразовых подписей.

3.5 Расшифровка идентификаторов платежей

Транзакции Monero опционально[§] содержат идентификаторы платежей, состоящие из XOR, между сообщением битовой строки и маской. Маска генерируется на основе хеша K_v^r, где ключом транзакции является $R = g^r$. Таким образом, в любом из предыдущих разделов, где QA может вычислить k_v при помощи алгоритма Шора, QA может применить алгоритм повторно в отношении R, чтобы вычислить r и вычислить маску X напрямую.

Кажется, что шифрования идентификаторов платежей по постквантовой схеме с некоторой постквантовой инфраструктурой ключей будет достаточно, чтобы решить проблему, о которой говорится в данном разделе, с внесением довольно консервативных изменений в кодовую базу.

3.6 Конфликт идентификаторов транзакций

Первый этап вычисления идентификатора транзакции $\mathcal{T}$ состоит в разбиении различных полей транзакции Monero на три набора:

1. d_1 содержит версию, входы, выходы и `дополнительные` поля транзакции;
2. d_2 содержит тип подписи, комиссию, обязательства по псевдовыходам, зашифрованные суммы транзакции и обязательства по выходам;
3. d_3 содержит кольцевые подписи и доказательства диапазона.

Эти наборы хешируются, чтобы получить идентификатор транзакции $\mathcal{T} = \mathcal{H}(\mathcal{H}(d_1)||\mathcal{H}(d_2)||\mathcal{H}(d_3))$. Чтобы создать конфликт, который сохранит структуру, которой будет достаточно для надлежащего разбиения, QA закрепляет два входа за конечной хеш-функцией и ищет конфликт в подполе третьего набора.

[§]В целях обеспечения единообразия транзакций версия Core кошелька использует зашифрованные идентификаторы (ID) платежей во всех транзакциях с двумя выходами, а ID платежей, указываемые простым текстом, якобы обрезаются. Тем не менее консенсус не предусматривает никакого строгого правила, и мы по-прежнему можем видеть транзакции без каких-либо ID платежей, а также транзакции с ID платежей, указанными простым текстом. Примером может служить транзакция с 1 входом и двумя выходами, зафиксированная в сентябре 2020 [37], без какого-либо ID платежа, а также транзакция [38] с ID платежа, указанным простым текстом.

Начнём с действительной транзакции $T_\circ$ с идентификатором транзакции $\mathcal{T}_\circ$ и выберем одно из подполей f для модификации, и отметим набор входов, содержащий его как d_f. Если QA действительно способен находить преобразы с фиксированными префиксом и суффиксом, то он может создавать и такую полезную информацию g, что $g \neq f$ (при этом подразумевается, что $d_g \neq d_f$), но $\mathcal{H}(d_g) = \mathcal{H}(d_f)$, а это приводит к конфликту. Рассмотрим изменение поля `extra` транзакции в d_1 и отметим, что $\mathcal{H}(\mathcal{H}(d_f)||\mathcal{H}(d_2)||\mathcal{H}(d_3)) = \mathcal{T}_\circ = \mathcal{T}_g = \mathcal{H}(\mathcal{H}(d_g)||\mathcal{H}(d_2)||\mathcal{H}(d_3))$. Транзакция с альтернативной полезной информацией T_g будет иметь тот же идентификатор транзакции, что и оригинальная транзакция $T_\circ$, и будет надлежащим образом форматироваться как транзакция, но не пройдёт верификацию.

Если узел злоумышленника ретранслирует действительный блок, ссылающийся на $\mathcal{T}$ вместе с T_g (версией транзакции, содержащей альтернативную полезную информацию), узел жертвы ошибочно пометит весь блок (и все последующие блоки в этом блокчейне) как недействительный. Этому могут сопутствовать махинации сетевого уровня (например, отправка одним узлам $T_\circ$, а другим - T_g), вызывающие расхождения.

Следует отметить, что наши предыдущие претензии в отношении применения алгоритма Гровера не следует относить к этому разделу. Фактически даже нахождения второго преобраза g с небольшим количеством бит достаточно для появления расхождений, позволяющих QA успешно провести эту атаку быстрее, чем может показаться на первый взгляд. Например, нахождение 32-битного преобраза для некоторого $h \in \{0,1\}^{256}$, выбранного единообразно случайным образом, занимает в среднем от 2 до 3 итераций алгоритма Гровера, если такой преобраз существует и приблизительно от 1 до 8 хеш-сумм в $\{0,1\}^{256}$ содержит преобраз. Поскольку нахождения даже одного плохого преобраза достаточно для совершения мошенничества, по самой скромной оценке 24 выполнений алгоритма Гровера будет достаточно, чтобы найти второй 32-битный преобраз (по крайней мере, в обозримой перспективе).

Также следует отметить, что если ввести требование, чтобы целевые данные содержали 256-битный корневой хеш дерева Меркла, то даже это не защитит от атаки, о которой говорится в данном разделе, если этот корневой хеш будет фиксированным. По сути, каноническое внесение строки в сообщение перед хешированием эквивалентно разделению домена, которое позволяет злоумышленнику, проводящему атаку, найти преобраз с низкой энтропией.

3.7 Конфликт хешей блоков

Как и в прошлом разделе QA получает дополнительные возможности за счёт использования хешей блоков, а не хешей транзакций. Фактически хеш блока $\mathcal{B}$ производится путём хеширования данных блока d, как и в предыдущем разделе, что позволяет QA найти преобраз d', как и до этого. Тем не менее QA может ограничить поиск теми значениями d', которые соответствуют значениям пустого блока. Напомним, что включение префикса в хеш разделяет домен хеш-фнукции, что приводит к появлению отдельных хеш-функций для каждого префикса. Например, при наличии хеш-функции $\mathcal{H}$ можно получить две отдельные хеш-функции, вычислив $\mathcal{H}(0 \mid\mid d)$ или $\mathcal{H}(1 \mid\mid d)$.

Таким образом, если кто-то опубликует хеш блока $\mathcal{B} = \mathcal{H}(d)$ для некоторого непустого блока d, QA сможет найти некоторые данные d', соответствующие пустому блоку и (с высокой степенью вероятности) другому нонсу. Затем QA сможет ретранслировать новый пустой блок с хешем блока $\mathcal{B}$ другим узлам. Узлы, которые в этом случае становятся жертвами, ошибочно сочтут, что блок на этой высоте является пустым, и любые транзакции, ссылающиеся на транзакции, находящиеся внутри этого блока, будут рассматриваться этими узлами в качестве недействительных. Данный подход также можно объединить с махинациями на сетевом уровне, что приведёт к расхождениям и ошибкам на уровне консенсуса или к разбиению блокчейна.

Тем не менее следует отметить, что блок с данными d, содержит корневой хеш дерева транзакций Меркла, находящегося внутри него. В случае Monero это 256-битная хеш-сумма, служащая преобразом для идентификатора блока. Это предполагает, что алгоритма Гровенра по-прежнему будет недостаточно для проведения атаки, о которой говорится в данном разделе, путём замены хорошего блока на пустой блок. Следует отметить, что если дерево Меркла состоит из пустых (плохих) транзакций, то корневой хеш можно рассматривать в ка-

честве фиксированного разделителя домена, и злоумышленник, проводящий атаку, всё ещё может попытаться обмануть узел жертвы.

Однако, как и в предыдущем разделе, преобраз с небольшим количеством бит может быть найден гораздо быстрее, что позволит QA совершить обман, отправив узлам жертв некие плохие данные, которые как будто бы будут соответствовать требованиям к хешам.

3.8 Нарушение конфиденциальности транзакций

Обязательства Педерсена идеальны с точки зрения сокрытия. При наличии набора транзакций, использующего маски, выбранные с высокой степенью энтропии, QA не сможет взять одно из обязательств Педерсена по сумме и вычислить сумму транзакции.

Тем не менее, как уже говорилось в Разделе 3.3.2, в Monero маски вычисляются детерминировано. Если у QA будет база данных известных ключей, QA сможет применить алгоритм Шора, чтобы получить ключ транзакции r для каждой транзакции, вычислить K_v^r для каждого K_v в этой базе данных и напрямую вычислить их маску y, соответствующую каждому K_v.

Следует отметить, что при наличии новой транзакции каждый ключ K_v в базе данных можно использовать для вычисления детерминированной маски, а обязательство Педерсена по сумме в транзакции может быть «демаскировано» при помощи этой маски, чтобы вычислить предполагаемую сумму транзакции. Если K_v не является ключом получателя, эта предполагаемая сумма транзакции является равномерно распределённым элементом $\mathbb{Z}_p$.

В случае с Monero действительные обязательства, вычисляемые получестными сторонами, содержат суммы по некоторому малому множеству $\{0, 1, \ldots, 2^n - 1\}$, где 2^n гораздо меньше, чем порядок группы p. Каждая предполагаемая сумма транзакции, однако, является равномерно распределяемой случайной переменной из скалярного поля, и вероятность, что предполагаемая сумма, соответствующая ключу K_v, не принадлежащему получателю, находится в этом небольшом множестве, крайне мала.

Даже при наличии базы с очень большим количеством ключей в целом только одна пара ключей (K_v, K_s) максимум будет соответствовать действительной сумме транзакции в диапазоне $\{0, \ldots, 2^n - 1\}$, и эта пара (K_v, K_s) с высокой степенью вероятности будет принадлежать получателю транзакции, что раскрывает как получателя, так и сумму. Вероятность ложноположительного результата (наличия несоответствующего ключа, которое случайно приведёт к появлению предполагаемой суммы в пределах действительного диапазона) приблизительно равна отношению размера диапазона из доказательства диапазона к размеру группы: $\sim 2^{64}/2^{255} = 2^{-191}$.

Безусловно, зная K_v и K_s, QA может проверить, принадлежит ли эта пара ключей получателю, вычислив k_v при помощи алгоритма Шора. Это позволит QA вычислить сумму транзакции напрямую, как и в предыдущих разделах.

4 Альтернативы криптографии на эллиптических кривых

4.1 Криптография на основе решёток

Криптография на основе решёток является основным претендентом на замену широко распространённой сегодня системы протоколов RSA и поэтому также должна рассматриваться в качестве возможной замены системы ECC, используемой Monero. Несмотря на то, что как RSA, так и ECC зависят от решения теоретико-числовых задач с использованием квантового алгоритма за полиномиальное время, в основе криптографии на основе решёток лежит решение задач, которые доказано являются NP-полными или NP-сложными. Оба класса сложности представлены задачами, лежащими за пределами BQP, важного класса задач, решаемых квантовым компьютером.

Задачи теории решёток очень хорошо изучены. Они изучены настолько хорошо, что некоторые из задач, датированные Диофантом III веком н.э., могут быть отнесены к задачам теории решёток, например. Задачи теории решёток всё чаще встречаются в криптографической литературе, в частности, начиная с работы [6]. Криптография на основе решёток представляет собой один из наиболее тестируемых подходов среди существующих в настоящее время вариантов постквантовой криптографии.

Если кратко, то уже только вышеуказанных факторов достаточно, чтобы прийти к заключению, что криптография на основе решёток является самым первым претендентом, который сможет усилить Monero против квантового злоумышленника. Гибкость основанных на теории решёток позволяет заменять компоненты постепенно, а не в рамках разового перехода.

4.1.1 С геометрической точки зрения

Криптография на основе решёток имеет несколько разновидностей, некоторые из которых можно визуализировать. Читатель может составить себе соответствующее понимание, представив эксперименты по сокрытию низкоразмерных линейных пространств в высокоразмерном пространстве. Произвольный 0-мерный объект (точка) в 1-мерном пространстве с соответствующей мерой (такой как вещественная числовая ось) имеет «длину», равную 0. Подобным образом 0-мерная точка или 1-мерный объект (такой как линия или кривая) в 2-мерном пространстве (таком как плоскость или поверхность сферы) имеет «площадь» (меру), равную 0. Произвольное 0-, 1- или 2-мерное пространство в 3-мерном пространстве имеет «объём» (меру), равную 0.

Подобная формулировка сохраняется и тогда, когда множества являются событиями, а мера является соответствующей мерой вероятности. Для понимания достаточно принять, что низкоразмерное событие в высокоразмерном пространстве происходит с ничтожно малой вероятностью.

Рассмотрим равномерно распределяемый случайный выбор из скалярного поля $\mathbb{Z}_p$, где $G = \langle g \rangle$ является некоторой циклической группой эллиптической кривой с генератором g и порядка p. Для заданного публичного ключа g^x равномерно распределяемый случайный выбор скалярной величины $y \leftarrow \mathbb{Z}_p$ даёт $g^y = g^x$ с вероятностью $1/p$; если G выбирается так, чтобы быть порядка 2^n для некоторого параметра безопасности n, то вероятность успешного угадывания приватного ключа подобным образом будет пренебрежимо малой функцией n. Фактически это аналогично нахождению 0-мерной скрытой точки в 1-мерном пространстве.

В частности, криптосистема публичных ключей может быть построена на базе предположительно случайных высокоразмерных (публичных) матриц и выбора секрета и основания некоторого подпространства.

Например, краткое целочисленное решение задачи сводится к представлению участнику публичной матрицы A и постановки перед ним задачи по нахождению в нормированном пространстве такого $\mathbf{x}$, чтобы $A\mathbf{x} = \mathbf{0}$, и чтобы такой $\mathbf{x}$ имел достаточно малое нормирование.

4.1.2 Различия между криптографией на основе решёток, RSA и ECC

Криптография на основе решёток, RSA и ECC имеют критически важные качественные различия.

В частности, в отличие от RSA и ECC и подобно многовариантной криптографии условия сложности в случае криптографии на основе решёток основаны на решении доказуемо NP-полных и NP-сложных задач, каждая из которых находится за пределами BQP.

На момент написания этой работы не существовало какого-либо квантового алгоритма, позволяющего решить такие задачи теории решёток и дать какое-либо значительное преимущество в сравнении с известными классическими алгоритмами, и любой подобный алгоритм, который бы выполнялся за полиномиальное время, мог бы быть использован для решения любой NP-задачи за полиномиальное время.

4.1.3 Переход

Криптография на основе решёток не является самым прорывным среди всего диапазона методов, которые, предположительно, будут использоваться в будущем. Важно то, что криптография на основе решёток под-

разумевает очевидно хорошую масштабируемость (по крайней мере, в большинстве случаев), и протоколы на её основе предполагают один из самых малых размеров ключей, если сравнивать с другими протоколами постквантового семейства. Это могло бы сделать их гораздо менее сложными при реализации относительно существующих систем, чем определённые другие варианты.

И всё же, несмотря на то, что криптография на основе решёток имеет множество многообещающих преимуществ, у неё есть и свои недостатки. Например, хотя решение на основе теории решёток может обеспечить гораздо меньшую длину ключей, если сравнивать с некоторыми другими перспективными методами, ключи на основе решёток всё же будут больше чем те, что используются существующими протоколами RSA, и время доказательство тоже может стать проблемой.

4.2 Многовариантные решения

Многовариантные криптосистемы в качестве приватных ключей, по сути, используют многовариантные полиномиальные инверсии, а также объединение соответствующих многовариантных многочленов в качестве публичных ключей. В результате криптосистема может быть построена на основе инверсии публичного ключа. Публичные ключи здесь являются случайными функциями, и всё же предполагаемый владелец может обратить функцию. Многовариантные схемы обычно быстры и вычислительно эффективны, что позволяет их использовать с устройствами с меньшей вычислительной мощностью, такими как смарт-карты и микросхемы RFID (см. Работы [39] и [40], например).

Условие сложности, лежащее в основе многовариантной криптографии, в свою очередь, основано на решении NP-полной задачи, которая находится за пределами BQP.

Безопасность многовариантных протоколов фундаментально зависит от задачи решения нелинейных многочленов в конечном поле, которая является NP-полной. Многовариантные квадратные многочлены являются линейными комбинациями одночленов с общей степенью равной, равной 2. То есть при наличии переменных $\{x_1, \ldots, x_n\}$ многовариантный квадратный многочлен в конечном поле $\mathbb{F}$ будет линейной комбинацией

$$p(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} \cdot x_i x_j + \sum_{i=1}^{n} b_i \cdot x_i + c$$

где $\mathbf{x} = (x_1, \ldots, x_n)^\top$ является вектором неизвестных в поле $\mathbb{F}$ и коэффициентов $a_{i,j}, b_i, c \in \mathbb{F}$. Название происходит от слова «много», что означает «более одного», и слова «вариант», означающего наличие «случайных переменных». Сумма, содержащая линейные члены, в данном случае может быть записана как скалярное произведение $\mathbf{x} \cdot \mathbf{b}$, где $\mathbf{b} = (b_1, \ldots, b_n)^\top \in \mathbb{F}^n$. Сумма, содержащая суперлинейные члены, также может быть представлена как скалярное произведение $p = \mathbf{x} \cdot (A\mathbf{x})$, где $W \in \mathbb{F}^{n \times n}$ является матрицей, в которой элементом $(i,j)^{th}$ является $a_{i,j}$. Таким образом, квадратный многовариантный многочлен можно записать как $p(\mathbf{x}) = \mathbf{x} \cdot (A\mathbf{x} + \mathbf{b}) + c$. Используя это, мы можем воспроизвести систему из m квадратных многовариантных многочленов по R со следующим вектором столбцов:

$$\mathbf{p}(\mathbf{x}) = \begin{pmatrix} \mathbf{x} \cdot \left(A^{(1)}\mathbf{x} + \mathbf{b}^{(1)} \right) + c^{(1)} \\ \mathbf{x} \cdot \left(A^{(2)}\mathbf{x} + \mathbf{b}^{(2)} \right) + c^{(2)} \\ \vdots \\ \mathbf{x} \cdot \left(A^{(m)}\mathbf{x} + \mathbf{b}^{(m)} \right) + c^{(m)} \end{pmatrix}.$$

Проверить, является ли некоторый $\mathbf{x}$ решением $\mathbf{p}(\mathbf{x}) = 0$, так же просто, как оценить многочлены m, но нахождение такого решения при наличии случайного p будет NP-полным.

В основе многовариантной криптографии лежат хорошо изученные задачи, так же как и в случае с криптографией на основе решёток. Криптосистемы, основанные на многовариантности публичных ключей, используют преимущества многих математических областей, среди которых теория операторов, нелинейная динамика, алгебра и комплексный анализ. Как таковая криптография многовариантных публичных ключей изобилует

различными техниками, основанными на фундаментальной математике. Подобно криптографии на основе решёток многовариантные задачи являются многообещающими и несут в себе новые перспективы с точки зрения постквантовой криптографии, что делает возможным построение совершенно гомоморфных протоколов шифрования, защищающих от квантовых атак, но обладающих при этом очень высоким уровнем гибкости. Кроме того, как ожидается, мультивариантные протоколы обеспечат возможность реализации довольно компактного размера ключей, равно как и короткого времени шифровки/расшифровки.

Судя по количеству работ, опубликованных по каждой из тем, многовариантная криптография вызывает относительно меньший интерес, чем криптография на основе решёток. По крайней мере, разработка протокола ограничена недостаточным количеством схем, из которого можно было бы выбирать. Информация о всех протоколах транзакций, обеспечивающих безопасность подобно Monero, находится в общем доступе (см. [41], [42], [43]).

Многовариантная криптография обладает множеством многообещающих свойств, превышающих уровень криптобезопасности, который будет актуален в будущем. Пожалуй, самым перспективным из них является относительно небольшой размер ключей и подписей, если сравнивать с другими постквантовыми криптографическими схемами. Тем не менее относительная недостаточность качественных исследований, в сравнении с другими постквантовыми криптографическими схемами, может оказаться неблагоприятным фактором, но в равной степени может и дать определённые возможности разработчикам и исследователям в будущем.

4.3 Криптография на основе хешей

Постквантовая криптография на основе хешей, как и следует из названия, предусматривает наличие криптографических протоколов, базирующихся на принципе генерирования ключей посредством безопасных хеш-функций. Основная сложность взлома такого рода криптографических схем в первую очередь объясняется устойчивостью преобразов указанных хеш-функций. Как уже подробно говорилось выше, гипотетический QA (квантовый злоумышленник), эффективно использующий алгоритм Гровера, вполне может сделать схемы хеш-функций уязвимыми, но тщательный выбор параметров хешей и конструкции определённых протоколов на базе дерева могут защитить от этого и обезопасить схемы от воздействия со стороны QA. Так мера, в которой может быть задействован квантовый дифференциальный криптоанализ, очень зависит от архитектуры протокола и выбора хеш-функции.

Протокол расширенной схемы подписи Меркла XMSS (eXtended Merkle Signature Scheme) является постквантовой криптографической схемой на основе хешей, которая в настоящее время используется в QRL (Quantum Resistant Ledger). QRL - это децентрализованная криптовалюта на базе блокчейн-технологии, которая обеспечивает, по сути, те же функции, что и публичный реестр Bitcoin, но с дополнительным «бонусом» в виде устойчивости к квантовым атакам. Несмотря на то, что протоколы на основе хешей довольно устойчивы к атакам QA, и они имеют свои недостатки: большой размер ключей и подписей, например.

4.4 Криптография на суперсингулярных эллиптических кривых на изогениях

Криптография на суперсингулярных эллиптических кривых на изогениях использует сложные задачи на изогении эллиптических кривых. Многие из существующих криптографических протоколов опираются на сложность решения задач, связанных с эллиптическими кривыми, но если ECC основан на сложности инвертирования канонического преобразования из поля в группу эллиптической кривой по этому полю, то криптография на базе SSEI использует сложность нахождения изогении между суперсингулярными эллиптическими кривыми. Изогения представляет собой эпиморфизм алгебраических групп с конечным ядром. Также этот класс задач можно рассматривать с той точки зрения, что он работает путём нахождения регулярных преобразований между сгенерированными эллиптическими кривыми. Эти классы сложных математических задач выглядят многообещающе с точки зрения их устойчивости как к квантовым, так и классическим атакам.

5 Альтернативные протоколы

В этом разделе нами будут рассмотрены некоторые альтернативные схемы, имеющие отношение к проводимому нами оценочному сравнению. В частности, нами будет представлена общая система доказательства, ZK-STARKs, протокол транзакций, считающийся устойчивым к квантовым атакам, MatRiCT и две схемы кольцевой подписи, которые также считаются устойчивыми к квантовым атакам. Несмотря на то, что эти схемы и не служат одной и той же цели, мы приводим их требования к занимаемому месту и времени в Таблице 1.

5.1 ZK-STARKs

Как указано в Работе [44], все задачи в NP имеют доказательства с нулевым разглашением. Краткий прозрачный аргумент знания с нулевым разглашением ZK-STARKs (Zero-Knowledge Scalable Transparent ARguments of Knowledge) является общей системой доказательства с нулевым разглашением для языков NP. ZK-STARKs не требует доверенных настроек и может использоваться в качестве замены доказательств диапазона и подписей Monero. Доказательства с нулевым разглашением позволяют одной стороне доказать свои утверждения (такие, как наличие достаточного количества средств для проведения транзакции) третьей стороне, не раскрывая скрытой информации такой третьей стороне (такой, как наличие общего количества средств при проведении транзакции). Данный тип протокола представляет собой фундаментальную основу криптовалюты Monero, и ZK-STARKs является альтернативой, поскольку также является безопасным с точки зрения возможной квантовой атаки.

ZK-STARKs состоит из схем, которые считаются устойчивыми относительно QA, и это хорошо описано в работе [45].

5.2 MatRiCT

Было предложено более одного варианта реализации протокола кольцевых конфиденциальных транзакций на основе решёток, например, в работах [41] и [43]. Единственным протоколом, демонстрирующим свою практическую эффективность, является протокол MatRiCT (Matrix Ring Confidential Transactions), описанный в работе [41]. Данный протокол предлагает потрясающие возможности и новаторские подсхемы, но при этом имеет по крайней мере один критический недостаток, препятствующий его реализации: отсутствие одноразовых адресов, используемых для сокрытия получателей.

В основе протокола Monero лежат связываемые кольцевые подписи, маскирующие отправителя транзакции, схема гомоморфных обязательств, позволяющая скрыть сумму транзакции, схема доказательства диапазона, доказывающая, что скрытая сумма является положительной и не станет причиной появления ошибок, связанных с превышением/переносом баланса транзакции, и одноразовые адреса, скрывающие получателя транзакции. MatRiCT использует задачи краткого целочисленного решения по модулю (M-SIS) и обучения с ошибками по модулю (M-LWE), чтобы обеспечить большую часть этих функций, не прибегая к нормальному распределению Гаусса, наряду с новой схемой извлекаемых обязательств, но не предусматривает наличия одноразовых адресов.

Длина доказательства, в соответствии с протоколом, примерно на два порядка величины меньше, чем в существующем постквантовом предложении, и хорошо масштабируется при наличии больших анонимных групп. В частности, транзакция может быть сгенерирована за время, составляющее от 250 до 3600 мсек, и верифицирована в период от 23 до 250 мсек в зависимости от уровня анонимности и аппаратного обеспечения. В случае MatRiCT размер публичного ключа составляет примерно 4 Кбайта, то есть примерно столько же, сколько и в случае RSA. Верификация транзакций, подобных транзакциям Monero, занимает около 25 мсек, а сами они занимают примерно 100 Кбайт.

За счёт относительно коротких ключей и относительно быстрого времени верификации MatRiCT является основным претендентом на то, чтобы сделать Monero устойчивой к воздействию QA, несмотря на то, что в

предложении, изложенном в работе [41], отсутствует функция использования одноразовых ключей, что позволяет увидеть получателя каждой транзакции. Но этого можно избежать следующим образом: если получатели не станут принимать более одной транзакции с использованием одного и того же публичного ключа (то есть ключи не будут использоваться повторно), то MatRiCT будет обеспечивать функции безопасности схожие с теми, что обеспечивает Monero. Однако такое решение оставляет желать лучшего, поскольку пользователи склонны прибегать к тому режиму, который был предложен им по умолчанию. Вспомним, что автор(ы) работы [46] рекомендовали никогда не использовать ключи повторно, даже в случае с Bitcoin, но при этом указывали на то, что повторное использование адресов является обычным делом.

В целом MatRiCT может заменить RingCT и большинство других механизмов Monero, которые окажутся уязвимыми для QA, за исключением одноразовых адресов. Если не учитывать этого недостатка, MatRiCT (или другой подобный протокол) представляется довольно многообещающим кандидатом.

5.3 Raptor-512

Протокол Raptor, предложенный в работе [47], является самым практичным протоколом (связываемых) кольцевых подписей на основе решёток. Он может похвастать небольшим (но, к сожалению, линейным) размером подписей, составляющим примерно $1,3$ Кбайт на участника кольца, и обеспечивает основу для реализации практически всех существующих функций обеспечения безопасности Monero.

В своих конструкциях Raptor использует так называемые «хеш-функции хамелеоны плюс» в качестве основных строительных блоков. Такие функции могут быть реализованы в рамках решётчатой схемы как устойчивые к конфликтам защищённые ключами односторонние хеш-функции с потайным входом. За счёт такого потайного входа появляется возможность без труда обнаружить конфликт для любого входа. Хеш-функции хамелеоны плюс являются именно тем фундаментальным строительным блоком, который отличает Raptor от остальных протоколов и схем на основе решёток.

Linkable-Raptor-512 - это название связываемой кольцевой подписи на базе протокола Raptor, обеспечивающей тот же размер подписи на одного участника кольца, что и Raptor-512. Размер публичных ключей Linkable-Raptor-512 составляет $0,9$ Кбайт (сравните с 64 байтами в случае Monero).

Как утверждается, время создания подписи составляет порядка 50 мсек в зависимости от варианта реализации и аппаратного обеспечения, а время верификации меньше 50 мсек. Небольшой размер кольца без связующих тегов обеспечивает примерно то же время верификации, что и у Monero, и занимает менее 3 мсек.

5.4 RingRainbow

Пока что многовариантные криптографические схемы обеспечивают самый малый размер подписи среди всех остальных предлагаемых постквантовых криптографических схем, обеспечивающих безопасность. Схема RingRainbow, описанная в работе [48], представляет собой многовариантную схему кольцевой подписи, позволяющую создавать подписи небольшого размера, а также обеспечивающую небольшое время верификации и устойчивость к атакам QA. Более того, RingRainbow строится при помощи простых и эффективных техник, применимых к схеме подписи Rainbow. Многовариантная криптография подразумевает высокий уровень гибкости, особенно в отношении кольцевых подписей (см. работы [49], [50], [51] и [52].

В этом разделе нами кратко описаны техники, используемые для построения RingRainbow. Получаемая схема демонстрирует идеальный уровень анонимности среди участников кольца, обеспечивает малый размер ключей, а также быструю верификацию подписей.

Схема подписи Rainbow предполагает генерирование ключей на основе двух обратимых линейных функций и квадратичной функции. Публичный ключ вычисляется на основе этих приватных ключей таким образом, что доступа к любому из приватных ключей достаточно, чтобы обратить публичный ключ. Чтобы подписать сообщение M, вычисляется хеш $h = \mathcal{H}(M)$. Подписант использует приватную часть пары ключей (sk, pk) для вычисления подписи σ таким образом, чтобы $pk(\sigma) = h$; вспомним, что pk является функцией, а нахождение

преобразов для случайной pk эквивалентно нарушению условия сложности, лежащего в основе NP-полной многовариантной задачи.

Чтобы построить кольцевую подпись для сообщения M с кольцом публичных ключей $\{pk_1, \ldots, pk_R\}$, используя приватный ключ sk_ℓ для некоторого индекса ℓ, подписант, как и прежде, сначала вычисляет $h = \mathcal{H}(M)$, случайным образом и с равномерным распределением выбирает некоторые случайные значения ложных выходов $\{\sigma_i\}_{i\neq\ell}$ и вычисляет $\widehat{h} = h - \sum_{i\neq\ell} pk_i(\sigma_i)$. Затем подписант использует свой приватный ключ для вычисления такого z_ℓ, чтобы $pk_\ell(z_\ell) = \widehat{h}$. Кольцевой подписью будет $(z_1, \ldots, z_R)$. Чтобы верифицировать подпись, верификатор вычисляет $\sum_i pk_i(z_i)$. В том случае, если приватный ключ sk_ℓ использовался для вычисления подписи получестным образом, мы разбиваем эту сумму на $pk_\ell(z_\ell) + \sum_{i\neq\ell} pk_i(z_i)$. Следует отметить, что в случае с получестным вычислением $pk_\ell(z_\ell) = \widehat{h} = h - \sum_{i\neq\ell} pk_i(z_i)$, поэтому $\sum_i pk_i(z_i) = h = \mathcal{H}(M)$. Более того, вычисление решения для $\sum_i pk_i(z_i) - h = 0$ без знания соответствующего sk_i является сложным, поэтому даже квантовый компьютер не сможет подделать подпись.

Схема кольцевой подписи при наличии надлежащей формализации является доказуемо безопасной как в постквантовом, так и в классическом варианте, а также в данном случае отмечается самый малый размер подписи из всех постквантовых методов, рассмотренных до этого момента.

Подпись RingRainbow занимает всего 43 байта (является самой малой из всех подписей, до этого представленных в этой работе), а размер публичного ключа составляет примерно 47 Кбайт. Время создания равно примерно 13 мсек, и каждая подпись может быть верифицирована менее чем за 25 мсек. Таким образом, основной особенностью, выделяющей RingRainbow среди других схем, является самый малый размер подписи..

6 Сравнительная таблица размеров ключей/времени верификации

В Таблице 1 представлены самые малые размеры ключей и самое короткое время верификации, обеспечиваемые каждым протоколом, включая RSA и ECC. Аппаратное обеспечение, которое было задействовано для определения времени генерирования и верификации, работало с тактовой частотой в диапазоне от 20 МГц до 3 ГГц. В случае RSA время верификации на медленном аппаратном обеспечении (ниже 20 МГц) для подписи с размером 1 Кбайт составило 2 сек, в то время как на более быстром аппаратном обеспечении (более 3 ГГц) оно же было равно 13 мсек. Большая часть значений времени генерирования ключей и верификации была получена с использованием аппаратного обеспечения, работающего с тактовой частотой, составлявшей примерно 3 ГГц.

	Размер подписи / доказательства (Кбайт)	**Размер публичного ключа** (Кбайт)	**Время генерирования** (Кбайт)	**Время верификации** (Кбайт)	**Применение**
Raptor-512	1.2	0.9	29 - 57	3 - 50	Схема кольцевой подписи
RingRainbow	0.043 - 0.43	47 - 460	13 - 3100	10 - 31000	Схема кольцевой подписи
Monero	0.25	0.064	50 - 70	10 - 20	Протокол транзакций
MatRiCT	0.65 - 6.5	4 - 10	240 - 3500	20 - 220	Протокол транзакций
ZK-STARK	0.064 - 100	10 - 120	100 - 60000	50 - 70	Система доказательства

Таблица 1: Сравнение криптографических протоколов и соответствующих требований к занимаемому месту и времени. Время верификации и генерирования артефактов в случае с различными схемами должно рассматриваться в качестве примерной оценки, поскольку в связи с другими протоколами и различным аппаратным обеспечением оно будет разным. В таблице приводятся лучшие и худшие значения. Требования к занимаемому месту в случае со схемами кольцевых подписей отражают размеры подписей и публичных ключей при размере анонимных групп, составляющем от 5 до 50 участников.

7 Заключение

Разработка масштабируемых квантовых компьютеров, способных эффективно использовать множество кубитов для реализации алгоритма Шора, представляет собой экзистенциальную угрозу для большинства криптовалют и банковских систем, включая Monero. По большей части это обстоит так из-за условия, согласно которому вычисление дискретного логарифма на эллиптических кривых является сложным, и это небезопасно, если будут задействованы квантовые компьютеры. Причиной этой угрозы также являются несколько уязвимостей в практической конструкции децентрализованных реестров, поскольку создание сообщений, которые могут вызвать конфликт хешей, позволяет разрушить защиту идентификаторов транзакций и блоков. Что ещё хуже, вследствие недавнего прогресса в области управления кубитами удалось сократить ожидаемое количество кубитов, необходимых для выполнения подобных трюков.

Идеи, представленные в этой работе, не следует толковать как избыточный список уязвимостей или техник, которыми однажды сможет воспользоваться QA, чтобы подорвать функционал Monero или других криптовалют.

Нами отмечается, что возможность ретроактивной деанонимизации может сделать сегодняшних пользователей Monero жертвами завтрашних (квантовых или классических) злоумышленников. Если когда-то в будущем на практике появятся квантовые компьютеры, способные взломать шифр Monero, то вся история транзакций всех пользователей станет публично доступной и будет с удовольствием поглощена AdTech-промышленностью, охотниками за информацией, преступниками и правительствами. И не имеет значения, какая из сторон первой опубликует деанонимизированную копию блокчейна Monero - уничтожение всяческой анонимности в глобальном масштабе станет необратимым.

Большинство функций протокола Monero и реализации Core уязвимы к воздействию со стороны квантового злоумышленника. К счастью, для решения проблем, связанной с этим рядом угроз, уже существуют схемы, которые вполне можно рассматривать в качестве защиты от атак квантовых злоумышленников и достаточно эффективной структурной замены существующих на сегодня схем Monero. Альтернативы криптографии на эллиптических кривых, перечисленные в этой технической записке, включают в себя схемы и протоколы, которые позволят эффективно заменить значительное количество функций и механизмов обеспечения безопасности Monero.

Большинство аспектов Monero, связанных с обеспечением безопасности, основано на криптографии на эллиптических кривых. Если что-то и следует почерпнуть из этой технической записки, так это то, что когда-нибудь эти аспекты будет необходимо перенести в область какого-либо другого вида криптографии, устойчивого к применению алгоритма Шора.

Вопреки общему заблуждению, согласно которому вся постквантовая криптография будет непрактичной с точки зрения требуемого места или времени верификации, данные, которые приводятся в Таблице 1, демонстрируют наличие некоторых альтернативных решений, близких к практической реализации. Поскольку атаки, о которых говорится в данной работе, в большинстве своём имеют ретроактивный характер и представляют собой лишь возможную при некоторых обстоятельствах угрозу анонимности пользователей Monero, мы рекомендуем тщательно взвешивать компромисс между производительностью и вероятными последствиями для сегодняшних пользователей, если их финансовые данные впоследствии можно будет раскрыть. Время, к которому эти механизмы будет необходимо заменить на их постквантовые безопасные версии, является предметом свободного обсуждения сообществом Monero. В ходе такого обсуждения мы рекомендуем сообществу учесть последние усовершенствования в области квантовых вычислений, равно как и то, что алгоритм Шора, алгоритм Гровера и алгоритм Саймона уже были реализованы экспериментально на действующих квантовых компьютерах.

Некоторые из мер защиты, о которых говорится в настоящей работе, представляют собой «низко висящий плод» с точки зрения их реализации. Например, переключаемые обязательства всего на один групповой элемент больше обязательств Педерсена и требуют внесения минимальных модификаций в доказательства диапазона Bulletproofs. Поэтому переключаемые обязательства представляются идеальным, не требующим

глобальных изменений кандидатом для защиты денежной массы Monero от угрозы, которую представляют собой квантовые компьютеры.

С другой стороны, применение некоторых из видов атак, представленных в данной работе, может привести к сбою в работе критически важных криптографических механизмов, таких как выведение приватных ключей на основе публичных ключей, что катастрофически скажется на методах генерирования ключей, используемых практически каждой криптовалютой. Решение некоторых из этих вопросов может потребовать внесения более радикальных изменений в кодовую базу Monero. Эти атаки также являются ретроактивными, поэтому отсутствие решения данных проблем в период до появления квантовых компьютеров (в нашу сегодняшнюю эру, когда мы понимаем возможности квантовых компьютеров, но пока не можем строить их в масштабе практического взлома криптографических схем) ставит ныне живущих пользователей Monero перед лицом неизвестности в будущем. Переход не должен начаться тогда, когда уже появятся квантовые компьютеры; существование реестра будет иметь смысл только в том случае, если переход будет завершён до того, как квантовый злоумышленник выйдет на сцену. Мы настоятельно рекомендуем сообществу Monero начать рассматривать возможность перехода к постквантовой безопасной инфраструктуре ключей и безопасным протоколам транзакций в будущем..

8 Благодарность

Мы высоко ценим тщательную редакцию и глубокие замечания, которые были получены нами от Саранга Ноезера. Примеры транзакций были любезно предоставлены нам Neptune из Noncesense Research Lab. Нам бы хотелось отдельно поблагодарить десятки членов сообщества, которые приняли участие в финансировании этого исследования посредством системы CCS Monero. Также мы благодарим Insight за предоставление места для проведения открытых технических исследований проблемы анонимности.

Мы благодарим Алексея Кальвина (Aleksey Calvin), Кунала Марваха (Kunal Marwaha), Шайенн Нельсон (Cheyenne Nelson) и Исследовательскую лабораторию Monero (`#monero-research-lab`) за полезные технические советы и исправления.

Мы выражаем отдельную личную благодарность Джуди Янг (Judy Young), Тиму Корбо (Tim Corbo), Кёртису Робинсону (Curtis Robinson), Джесси Корсон (Jesse Korson), Дженифер Айяла (Jennifer Ayala) и Кэмерон Уэллс (Cameron Wells). И, конечно же, большое спасибо всем нашим собакам, кошкам, друзьям, родственникам и партнёрам, которые поддерживали нас при реализации этого проекта.

A Техническая реализация квантовых алгоритмов

Во вступлении к данной технической записке кратко описаны некоторые возможности, которые теоретически мог бы применить квантовый злоумышленник, чтобы воспользоваться уязвимостями существующей инфраструктуры безопасности Monero. Это приложение призвано обеспечить более глубокое техническое понимание алгоритмов, которые будут работать на квантовом аппаратном обеспечении для обеспечения указанных возможностей. Тремя алгоритмами, представленными в этом приложении, являются алгоритм Шор, алгоритм Гровера и алгоритм Саймона. Несмотря на то, что, как было отмечено, эти три алгоритма представляют самый большой риск для Monero с точки зрения безопасности, возможности квантового компьютера не ограничиваются их выполнением. В эту группу не включено множество алгоритмов, которые могут использоваться в целях квантового дифференциального криптоанализа (QDC), таких как алгоритм Бернштейна-Вазирани и прочие. Основной угрозой с точки зрения QDC является обращение или раскрытие скрытой информации, связанной с преобразом хеш-суммы.

Ниже нами приводится рабочее описание хрестоматийных вариантов реализации квантовых алгоритмов, представленных в данной технической записке. Примеры кода и результаты, представленные в приложениях, используют открытую библиотеку qiskit python и были получены при помощи квантового аппаратного

обеспечения, доступ к которому был получен через облачный сервис IBM Quantum Experience.

A.1 Техническая реализация алгоритма Шора

Алгоритм Шора позволяет нарушить условие сложности RSA (см. Работу [53]) и условие сложности решения задачи дискретного логарифмирования (см. Работу [9]), оба из которых предполагают сложность решения за полиномиальное время.

Эллиптические кривые над конечными полями формируют абелевы группы. Для (необязательно циклической) группы $\mathbb{G}$ с некоторым генератором $g \in \mathbb{G}$, g^n будет n-кратным применением групповой операции с g, то есть

$$g^n = \underbrace{g \cdot g \cdot \ldots \cdot g}_{n}. \tag{1}$$

Если $g' = g^x$ для некоторого $g \in \mathbb{G}$ и элемента поля x, мы говорим, что x является дискретным логарифмом g' относительно g. Группа $\mathbb{G}$ является циклической, то есть $\mathbb{G} = \langle g \rangle$ так, что $g^p = \mathrm{id}_{\mathbb{G}}$, а значит, любой элемент $g' \in \mathbb{G}$ может быть записан как $g' = g^x$ для некоторого $x \in \mathbb{Z}_p$.

Условие сложности решения задачи дискретного логарифмирования заключается в том, что в случае с группой эллиптической кривой алгоритму будет сложно выдать x, если известно только значение g, и $g' = g^x$. Далее следует описание алгоритма Шора применительно к эллиптическим кривым, в первую очередь основанное на Работе [9].

После приведения кубитов в нулевое состояние к регистрам применяется унитарное преобразование, после чего квантовое состояние системы можно описать следующим образом:

$$|\psi\rangle = \frac{1}{2^n} \sum_{x=0}^{2^n-1} \sum_{y=0}^{2^n-1} |x, y, g^x (g')^y\rangle \,. \tag{2}$$

Чтобы вычислить $g^x (g')^y$, сначала мы циклически возводим в квадрат групповые элементы g и g', получая таким образом множители $P_i = g^{2^i}$ и $Q_i = (g')^{2^i}$. Затем мы можем вычислить произведения $P_i Q_i$ и записать

$$g^x (g')^y = \prod_i g^{x_i} (g')^{y_i}. \tag{3}$$

где x_i и y_i являются битами i^{th} в двоичном расширении x и y, соответственно. Эти групповые элементы могут быть классическим образом предварительно вычислены, а $|x, y\rangle$ могут быть описаны одним кубитом. Это выполняется при помощи полуклассического квантового преобразования Фурье, описанного в Работе [54], и аналогично алгоритму факторизации, используемому для взлома RSA. Таким образом, регистр сумматора R, мы можем представить $|x, y, g^x (g')^y\rangle = |x, y, R\rangle$, описав $|x, y\rangle$ одним кубитом.

После выполнения этих шагов задача сводится к умножению фиксированной, предварительно вычисленной и классически известной точки P_i на наложение точек. Это произведение, по сути, смещает дискретный логарифм каждого компонента в наложении на одну и ту же величину. Это, в свою очередь, приводит к появлению необходимости в унитарном преобразовании U_{P_i}, U_{Q_i}, работающем по состоянию $|S_i\rangle$, представляющему точку на эллиптической кривой, то есть $U_{P_i} : |S\rangle \to |S \cdot P_i\rangle$ и $U_{Q_i} : |S\rangle \to |S \cdot Q_i\rangle$.

Эти шаги применяются n раз в отношении g^x и n раз в отношении $(g')^y$, где $n \approx log_2(q)$. Этот ряд шагов разбивает квантовый алгоритм дискретного логарифмирования на последовательность групповых смещений на постоянные классически известные элементы. После того как это будет выполнено, измеренный результат должен будет соответствовать распределению с пиками вокруг скалярных величин (например, приватных ключей), равных Nk/q и Ndk/q, где $N = 2^n$. Умножение на q/N даёт k и dk, где d является значением дискретно логарифма, которое мы пытаемся найти.

Алгоритм Шора, выполняемый на квантовом компьютере, работает экспоненциально быстрее, чем любой другой известный классический алгоритм, предназначенный для вычисления дискретных логарифмов.

```
"""Далее следует код python, использующий библиотеку qiskit, работающую на существующем
квантовом аппаратном обеспечении и требующую 5 кубитов для факторизации целого числа 15 на 3 и 5.
При наличии периода обнаружения для a^r mod N = 1, где a = 11 a N = 15 (целое число, подлежащее
факторизации), задача состоит в нахождении r значений для этого равенства так, чтобы можно было
найти простые множители N. В случае с 11^r mod(15) =1,
результаты (как показано на рис. 1) соответствуют периоду r = 4 (|00100>) и r = 0 (|00000>).
Чтобы найти множитель, используется равенство a^r mod 15. В результате получаем:
(a^r -1) mod 15 = (a^(r/2) + 1)(a^(r/2) - 1) mod 15. В этом случае, a = 11. Включение двух
значений r для этого значения даёт нам (11^(0/2) +1)(11^(4/2) - 1) mod 15 = 2*(11 +1)(11-1) mod 15.
Таким образом, мы находим (24)(20) mod 15. Нахождение самого наибольшего общего множителя
между двумя коэффициентами gcd(24,15) и gcd(20,15), даёт нам 3 и 5, соответственно. Это простые
множители 15. Таким образом мы демонстрируем результат работы алгоритма Шора при нахождении
простых множителей целого числа с использованием квантового аппаратного обеспечения. Следует
отметить, что это не то же самое, что техническая реализация алгшоритма Шора, описанная в
данном разделе для нарушения условия сложности решения задачи дискретного логарифмирования,
хотя доказательство концепции и сохраняется."""

# Импорт библиотек

from qiskit.compiler import transpile, assemble
from qiskit.tools.jupyter import *
from qiskit.visualization import *
from numpy import pi
from qiskit import IBMQ, Aer, QuantumCircuit, ClassicalRegister, QuantumRegister, execute
from qiskit.providers.ibmq import least_busy
from qiskit.visualization import plot_histogram

# Инициализация регистров кубитов
qreg_q = QuantumRegister(5, 'q')
creg_c = ClassicalRegister(5, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

circuit.reset(qreg_q[0])
circuit.reset(qreg_q[1])
circuit.reset(qreg_q[2])
circuit.reset(qreg_q[3])
circuit.reset(qreg_q[4])

# Применение Адамара к регистрам кубитов
circuit.h(qreg_q[0])
circuit.h(qreg_q[1])
circuit.h(qreg_q[2])

# Применение первого QFT, модульного возведения в степень и ещё одного QFT
circuit.h(qreg_q[1])
circuit.cx(qreg_q[2], qreg_q[3])
circuit.crx(pi/2, qreg_q[0], qreg_q[1])
```

```
circuit.ccx(qreg_q[2], qreg_q[3], qreg_q[4])
circuit.h(qreg_q[0])
circuit.rx(pi/2, qreg_q[2])
circuit.crx(pi/2, qreg_q[1], qreg_q[2])
circuit.crx(pi/2, qreg_q[1], qreg_q[2])
circuit.cx(qreg_q[0], qreg_q[1])

# Измерение регистров кубитов 0-2
circuit.measure(qreg_q[2], creg_c[2])
circuit.measure(qreg_q[1], creg_c[1])
circuit.measure(qreg_q[0], creg_c[0])

# Запуск наименее занятой квантовой вычислительной машины базы данных
provider = IBMQ.load_account()
device = least_busy(provider.backends(filters=lambda x: x.configuration().n_qubits >= 3 and
                                    not x.configuration().simulator and x.status().operational==True))
print("Running on current least busy device: ", device)

# Запуск схемы на доступном квантовом компьютере и построение гистограммы
from qiskit.tools.monitor import job_monitor
job = execute(circuit, backend=device, shots=1024, optimization_level=3)
job_monitor(job, interval = 2)

results = job.result()
answer = results.get_counts(circuit)
plot_histogram(answer)

# Результаты с самой большой амплитудой соответствуют значениям r, использованным для вычисления
простого множителя N.
```

Репозиторий GitHub с данным кодом можно найти по следующей ссылке:

https://github.com/hamburgerguy/Quantum-Algorithm-Implementations/blob/master/Shor.py

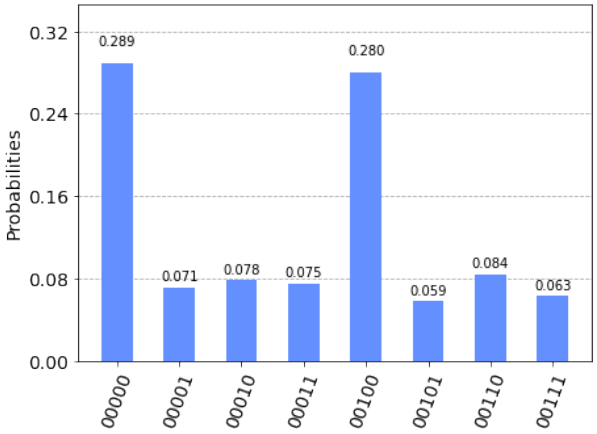

Рис. 1: Гистограмма, полученная после выполнения кода алгоритма Шора для факторизации 15 на квантовом компьютере ibmq-16-melbourne. Результаты соответствуют теории.

A.2 Техническая реализация алгоритма Гровера

При наличии набора данных целевой вариант реализации алгоритма Гровера позволяет получить на основе этих данных маркированный элемент x_0, который может быть конфиденциальным преобразом хеш-суммы, например. Далее описаны шаги, которые необходимо предпринять для этого:

1. Инициализация кубитов.
$$|\psi\rangle = |0\rangle^{\otimes n}$$

2. Перевод кубитов в состояние равномерной суперпозиции.
$$|s\rangle = H^{\otimes n}|0\rangle^n$$

3. Применение отражения оракула $|U_f\rangle$, являющегося унитарным преобразованием, к маркированному элементу x_0 кубитов.

4. Применение дополнительного отражения
$$U_s = 2|s\rangle\langle s| - \mathbb{1}$$
так, чтобы это отражало состояние $U_s U_f|s\rangle$.

5. Повторение шагов 3 и 4 до тех пор, пока состояние системы можно будет описать как $|\psi_t\rangle = (U_s U_f)^t|s\rangle$. Для этого понадобится примерно $\sqrt{n} = \mathrm{t}$ раз, где n является количеством записей в базе данных. Кубиты измеряются, и соответствующая амплитуда должна соответствовать эквивалентному количеству классических бит целевой записи.

```
"""Вариант реализации алгоритма Гровера на языке Python посредством использования библиотеки
Qiskit для нахождения 3 (|11>) из четырёх возможных значений."""

# Импорт numpy и библиотеки графиков

import matplotlib.pyplot as plt
import numpy as np
```

```
# Импорт Qiskit
from qiskit import IBMQ, Aer, QuantumCircuit, ClassicalRegister, QuantumRegister, execute
from qiskit.providers.ibmq import least_busy
from qiskit.quantum_info import Statevector

# Импорт базовых инструментов работы с графиками
from qiskit.visualization import plot_histogram

# Определение уязвимостей, 1) инициализация кубитов в нулевом состоянии
n = 2
grover_circuit = QuantumCircuit(n)

# Определение функции инициализации
def initialize_s(qc, qubits):
    '''Apply a H-gate to 'qubits' in qc'''
    for q in qubits:
        qc.h(q)
    return qc

### Запуск схемы Гровера ###

# 2) Перевод кубитов в состояние равномерной суперпозиции
grover_circuit = initialize_s(grover_circuit, [0,1])

# 3) Применение отражения оракула к маркированному элементу x_0 = 3, (|11>)
grover_circuit.cz(0,1)
statevec = job_sim.result().get_statevector()
from qiskit_textbook.tools import vector2latex
vector2latex(statevec, pretext="|\\psi\\rangle =")

# 4) Применение дополнительного отражения (диффузионный оператор)
grover_circuit.h([0,1])
grover_circuit.z([0,1])
grover_circuit.cz(0,1)
grover_circuit.h([0,1])

# При выполнении на существующем квантовом аппаратном обеспечении

# Загрузить учётную запись IBM Q и получить доступ к наименее занятому компьютеру базы данных
provider = IBMQ.load_account()
device = least_busy(provider.backends(filters=lambda x: x.configuration().n_qubits >= 3 and
                                    not x.configuration().simulator and x.status().operational==True))
print("Running on current least busy device: ", device)

# 5) Измерить кубиты
grover_circuit.measure_all()
```

```
# При работе с симуляторами кубитов
qasm_simulator = Aer.get_backend('qasm_simulator')
shots = 1024
results = execute(grover_circuit, backend=qasm_simulator, shots=shots).result()
answer = results.get_counts()

# При выполнении на реальном аппаратном обеспечении

from qiskit.tools.monitor import job_monitor
job = execute(grover_circuit, backend=device, shots=1024, optimization_level=3)
job_monitor(job, interval = 2)

results = job.result()
answer = results.get_counts(grover_circuit)
plot_histogram(answer)

# Наибольшая амплитуда должна соответствовать маркированному значению x_0 (|11>)
```

Ссылка Github для выполнения алгоритма Гровера на квантовом аппаратном обеспечении:

https://github.com/hamburgerguy/Quantum-Algorithm-Implementations/blob/master/Grover.py

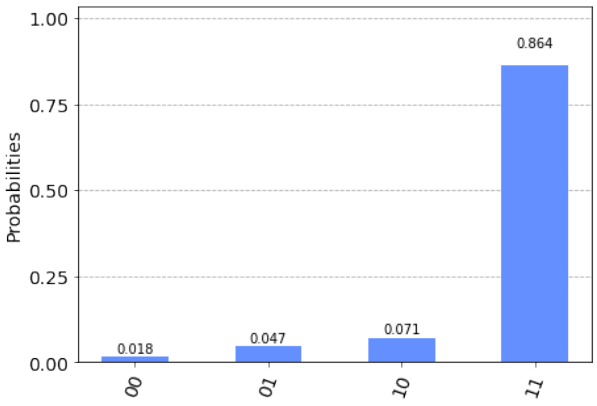

Рис. 2: Гистограмма, полученная после выполнения кода алгоритма Гровера на квантовом компьютере ibmqx2. Результаты соответствуют теории.

A.3 Техническая реализация алгоритма Саймона

Задача алгоритма Саймона состоит в том, чтобы при наличии функции $f : \{0,1\}^n \to \{0,1\}^n$, которая считается инвариантной при наличии некоторой n-битной XOR маски a, вычислить a. Другими словами, при известной $f(x) = f(y)$ и только если $x \oplus y \in \{0, a\}$, вычислить a.

Это может использоваться для построения системы линейных уравнений, которые можно будет использовать для нахождения результатов и вычисления маски XOR определённых функций, которые пимеют многочисленное количество способов применения в криптографии.

1. Инициализация двух регистров n-кубитов в нулевом состоянии.

$$|\psi_1\rangle = |0\rangle^{\otimes n} |0\rangle^{\otimes n} \tag{4}$$

2. Применение преобразования Адамара к первому регистру.

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle|0\rangle^{\otimes n} \tag{5}$$

3. Применение функции запроса Q_f.

$$|\psi_3\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle|f(x)\rangle \tag{6}$$

4. Измерение второго регистра, в результате чего первый регистр приобретает следующую форму:

$$|\psi_4\rangle = \frac{1}{\sqrt{2}} \left(|x\rangle + |y\rangle\right). \tag{7}$$

5. Применение преобразования Адамара к первому регистру.

$$|\psi_5\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{z\in\{0,1\}^n} \left[(-1)^{x\cdot z} + (-1)^{y\cdot z}\right] |z\rangle \tag{8}$$

6. Измерение первого регистра, которое даёт следующий результат:

$$(-1)^{x\cdot z} = (-1)^{y\cdot z}. \tag{9}$$

Результат должен соответствовать строке z, которая преобразуется в $b \cdot z = 0 \pmod 2$ и после применения метода исключения Гаусса может использоваться для вычисления маски XOR для функции $f(x)$.

```
"""Код Qiskit для выполнения алгоритма Саймона на квантовом аппаратном обеспечении при наличии 2 кубитов и

# Импорт Qiskit
from qiskit import IBMQ, BasicAer
from qiskit.providers.ibmq import least_busy
from qiskit import QuantumCircuit, execute

# Импорт базовых инструментов работы с графиками
from qiskit.visualization import plot_histogram
from qiskit_textbook.tools import simon_oracle

# Присвоение b значения '11'
b = '11'

# 1) Инициализация кубитов
n = 2
simon_circuit_2 = QuantumCircuit(n*2, n)

# 2) Применение вентиля Адамара перед передачей запроса оракулу
simon_circuit_2.h(range(n))

# 3) Передача запроса оракулу
simon_circuit_2 += simon_oracle(b)

# 5) Применение вентиля Адамара к входному регистру
simon_circuit_2.h(range(n))

# 3) и 6) Измерение кубитов
simon_circuit_2.measure(range(n), range(n))

# Загрузка сохранённых учётных записей IBMQ и получение доступа к наименее занятому компьютеру базы данных
IBMQ.load_account()
provider = IBMQ.get_provider(hub='ibm-q')
backend = least_busy(provider.backends(filters=lambda x: x.configuration().n_qubits >= n and
                                    not x.configuration().simulator and x.status().operational==True))
print("least busy backend: ", backend)

# Выполнение и отслеживание работы
from qiskit.tools.monitor import job_monitor
shots = 1024
job = execute(simon_circuit_2, backend=backend, shots=shots, optimization_level=3)
job_monitor(job, interval = 2)

# Получение графиков и построение расчётов
```

```
device_counts = job.result().get_counts()
plot_histogram(device_counts)

# Дополнительное применение функции для вычисления скалярного произведения результатов
def bdotz(b, z):
    accum = 0
    for i in range(len(b)):
        accum += int(b[i]) * int(z[i])
    return (accum % 2)

print('b = ' + b)
for z in device_counts:
    print( '{}.{} = {} (mod 2) ({:.1f}%)'.format(b, z, bdotz(b,z), device_counts[z]*100/shots))

# Самыми значащими результатами будут те, для которых b dot z=0(mod 2).

'''b = 11
11.00 = 0 (mod 2) (45.0%)
11.01 = 1 (mod 2) (6.2%)
11.10 = 1 (mod 2) (6.4%)
11.11 = 0 (mod 2) (42.4%)'''


Ссылка Github для этого кода:

https://github.com/hamburgerguy/Quantum-Algorithm-Implementations/blob/master/Simon.py
```

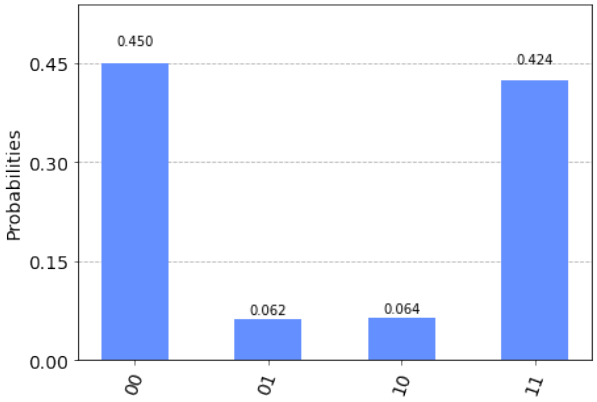

Рис. 3: Результат выполнения кода алгоритма Саймона на квантовом компьютере ibmqx2. Результаты соответствуют теории.

Список литературы

[1] Whitfield Diffie and Martin Hellman. New directions in cryptography (Новые направления в криптографии). *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[2] Andreas Antonopoulos. *Mastering Bitcoin (Овладевая Bitcoin)*. O'Reilly Media, Inc., 2017.

[3] Andreas Antonopoulos and Gavin Wood. *Mastering Ethereum (Овладевая Ethereum)*. O'Reilly Media, Inc., 2018.

[4] SerHack. *Mastering Monero (Овладевая Monero)*. Lernolibro LLC, 2018.

[5] Daira Hopwood, Sean Bowe, Taylor Hornby, and Nathan Wilcox. Zcash protocol specification, version version 2020.1.14 (Спецификация протокола zcash. Версия 2020.1.14). `https://raw.githubusercontent.com/zcash/zips/master/protocol/protocol.pdf`, 2020.

[6] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring (Алгоритмы для квантовых вычислений: дискретные логарифмы и факторизация). In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.

[7] Lov K. Grover. A fast quantum mechanical algorithm for database search (Быстрый квантово-механический алгоритм поиска в базах данных). In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[8] Daniel R. Simon. On the power of quantum computation (По вопросу мощности квантовых вычислений). *SIAM journal on computing*, 26(5):1474–1483, 1997.

[9] John Proos and Christof Zalka. Shor's discrete logarithm quantum algorithm for elliptic curves (Применение квантового алгоритма решения задачи дискретного логарифмирования Шора к эллиптическим кривым). *arXiv preprint quant-ph/0301141*, 2003.

[10] Michele Mosca. Quantum algorithms (Квантовые алгоритмы). *arXiv preprint arXiv:0808.0369*, 2008.

[11] Steven D Galbraith and Pierrick Gaudry. Recent progress on the elliptic curve discrete logarithm problem (Последние успехи в области решения задач дискретного логарифмирования на эллиптических кривых). *Designs, Codes and Cryptography*, 78(1):51–72, 2016.

[12] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing (Сильные и слабые стороны квантовых вычислений). *SIAM journal on Computing*, 26(5):1510–1523, 1997.

[13] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching (Жёсткие границы квантового поиска). *Fortschritte der Physik: Progress of Physics*, 46(4-5):493–505, 1998.

[14] Jan Czajkowski, Leon Groot Bruinderink, Andreas Hülsing, and Christian Schaffner. Quantum preimage, 2nd-preimage, and collision resistance of sha3 (Квантовый прообраз, 2-ой прообраз и устойчивость к конфликтам sha3). *IACR ePrint*, 302:2017, 2017.

[15] Daniel J. Bernstein. "quantum attacks against blue midnight wish, echo, fugue, grøstl, hamsi, jh, keccak, shabal, shavite-3, simd, and skein (Квантовые атаки на blue midnight wish, echo, fugue, grøstl, hamsi, jh, keccak, shabal, shavite-3, simd и skein). `cr.yp.to/papers.html#quantumsha3`, 2010.

[16] Pascal Koiran, Vincent Nesme, and Natacha Portier. A quantum lower bound for the query complexity of simon's problem (Квантовая нижняя граница сложности решения задачи Саймона). In *International Colloquium on Automata, Languages, and Programming*, pages 1287–1298. Springer, 2005.

[17] Xavier Bonnetain. Tight bounds for simon's algorithm (Жёсткие границы алгоритма Саймона).

[18] Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems (Дифференциальный криптоанализ криптосистем, подобных des). *Journal of CRYPTOLOGY*, 4(1):3–72, 1991.

[19] Thomas Santoli and Christian Schaffner. Using simon's algorithm to attack symmetric-key cryptographic primitives (Использование алгоритма Саймона для проведения атак на криптографические примитивы симметричных ключей). *arXiv preprint arXiv:1603.07856*, 2016.

[20] Shengbao Wu and Mingsheng Wang. Security evaluation against differential cryptanalysis for block cipher structures (Оценка безопасности с точки зрения возможности применения дифференциального криптоанализа структур с блочным шифрованием). *IACR Cryptol. ePrint Arch.*, 2011:551, 2011.

[21] Matthias J. Kannwischer, Peter Pessl, and Robert Primas. Single-trace attacks on keccak (Атаки на keccak с одиночной трассировкой). *IACR Cryptol. ePrint Arch.*, 2020:371, 2020.

[22] Nathaniel Graff. Differential power analysis in-practice for hardware implementations of the keccak sponge function (Дифференциальный анализ мощности при практической аппаратной реализации функции губки keccak). 2018.

[23] Daniel J. Bernstein. Quantum attacks against blue midnight wish, echo, fugue, grøstl, hamsi, jh, keccak, shabal, shavite-3, simd, and skein.

[24] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (Снова о методике применения случайного оракула). *Journal of the ACM (JACM)*, 51(4):557–594, 2004.

[25] Shafi Goldwasser and Yael Tauman Kalai. On the (in) security of the fiat-shamir paradigm (По вопросу безопасности в парадигме протокола Фиата-Шамира). In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 102–113. IEEE, 2003.

[26] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model (Неинтерактивные доказательства с нулевым разглашением в рамках квантовой модели случайного оракула). In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 755–784. Springer, 2015.

[27] Dominique Unruh. Post-quantum security of fiat-shamir (Постквантовая безопасность протокола Фиата-Шамира). In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 65–95. Springer, 2017.

[28] Claus-Peter Schnorr. Efficient identification and signatures for smart cards (Эффективная идентификация и создание подписей с применением смарт-карт). In *Conference on the Theory and Application of Cryptology*, pages 239–252. Springer, 1989.

[29] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world (Применение случайных оракулов в квантовом мире). In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 41–69. Springer, 2011.

[30] Koe, Kurt M. Alonso, and Sarang Noether. *Zero to Monero: Second edition v2.0.0 (Monero с нуля: второе издание v2.0.0)*. 2020.

[31] Shen Noether, Adam Mackenzie, et al. Ring confidential transactions (Кольцевые конфиденциальные транзакции). *Ledger*, 1:1–18, 2016.

[32] Andrew Poelstra, Adam Back, Mark Friedenbach, Gregory Maxwell, and Pieter Wuille. Confidential assets. In *International Conference on Financial Cryptography and Data Security (Материалы Международной конференции по финансовой криптографии и безопасности данных)*, pages 43–63. Springer, 2018.

[33] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing (Неинтерактивное и информационно-теоретическое совместное безопасное использование общих секретов с возможностью верификации). In *Annual international cryptology conference*, pages 129–140. Springer, 1991.

[34] Tim Ruffing and Giulio Malavolta. Switch commitments: A safety switch for confidential transactions (Переключаемые обязательства: безопасный переход от конфиденциальных транзакций). In *International Conference on Financial Cryptography and Data Security*, pages 170–181. Springer, 2017.

[35] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more (bulletproofs: компактные доказательства конфиденциальных транзакций и многое другое). In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 315–334. IEEE, 2018.

[36] Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications (Эффективные аргументы с нулевым разглашением на базе решёток со стандартным уровнем надёжности: построение и применение). In *Annual International Cryptology Conference*, pages 147–175. Springer, 2019.

[37] Транзакция из основной сети Monero c62c6707f5f12939456b3df3839b6b8a6102fe331bdaaf3241ae2d7f5fdf4c2b. `https://xmrchain.net/tx/C62C6707F5F12939456B3DF3839B6B8A6102FE331BDAAF3241AE2D7F5FDF4C2B`, block 2180052 on 2020-09-05.

[38] Транзакция из основной сети Monero 6d2bcb7a61b002cf8ad695f459cf56546c55809d719d96d48ae48ea74c98be21. `https://xmrchain.net/tx/6D2BCB7A61B002CF8AD695F459CF56546C55809D719D96D48AE48EA74C98BE21`, block 2176830 on 2020-09-01.

[39] Andrey Bogdanov, Thomas Eisenbarth, Andy Rupp, and Christopher Wolf. Time-area optimized public-key engines: Mq-cryptosystems as replacement for elliptic curves? (Оптимизированные по времени инструменты генерирования публичных ключей: могут ли mq-криптосистемы заменить системы на основе эллиптических кривых?). In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 45–61. Springer, 2008.

[40] Anna Inn-Tung Chen, Ming-Shing Chen, Tien-Ren Chen, Chen-Mou Cheng, Jintai Ding, Eric Li-Hsiang Kuo, Frost Yu-Shuang Lee, and Bo-Yin Yang. Sse implementation of multivariate pkcs on modern x86 cpus (sse реализация многовариантных pkcs на современных компьютерах x86). In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 33–48. Springer, 2009.

[41] Muhammed F. Esgin, Raymond K. Zhao, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. Matrict: efficient, scalable and post-quantum blockchain confidential transactions protocol (matrict: протокол эффективных, масштабируемых и постквантовых конфиденциальных транзакций). In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 567–584, 2019.

[42] Wilson Abel Alberto Torres, Ron Steinfeld, Amin Sakzad, Joseph K. Liu, Veronika Kuchta, Nandita Bhattacharjee, Man Ho Au, and Jacob Cheng. Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain (lattice ringct v1. 0) (Постквантовая одноразовая связываемая кольцевая подпись и её применение в кольцевых конфиденциальных транзакциях в блокчейне (решётка ringct v1.0)). In *Australasian Conference on Information Security and Privacy*, pages 558–576. Springer, 2018.

[43] Wilson Alberto Torres, Veronika Kuchta, Ron Steinfeld, Amin Sakzad, Joseph K. Liu, and Jacob Cheng. Lattice ringct v2. 0 with multiple input and multiple output wallets (Применение решётки ringct v2.0 к кошелькам со множеством входов и выходов). In *Australasian Conference on Information Security and Privacy*, pages 156–175. Springer, 2019.

[44] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems (Доказательства, не доказывающие ничего, кроме собственной действительности, или все языки в np имеют доказательства с нулевым разглашением). *Journal of the ACM (JACM)*, 38(3):690–728, 1991.

[45] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity (Целостность масштабируемых, прозрачных постквантовых вычислений). *IACR Cryptol. ePrint Arch.*, 2018:46, 2018.

[46] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system (bitcoin: одноранговая электронная денежная система).

[47] Xingye Lu, Man Ho Au, and Zhenfei Zhang. Raptor: a practical lattice-based (linkable) ring signature (raptor: практическаяч реализация (связываемых) кольцевых подписей на основе решёток). In *International Conference on Applied Cryptography and Network Security*, pages 110–130. Springer, 2019.

[48] Mohamed Saied Emam Mohamed and Albrecht Petzoldt. Ringrainbow–an efficient multivariate ring signature scheme (ringrainbow - Эффективные схемы многовариантных кольцевых подписей). In *International Conference on Cryptology in Africa*, pages 3–20. Springer, 2017.

[49] Albrecht Petzoldt. Efficient key generation for rainbow (Эффективное генерирование ключей в rainbow). In *International Conference on Post-Quantum Cryptography*, pages 92–107. Springer, 2020.

[50] Albrecht Petzoldt, Stanislav Bulygin, and Johannes Buchmann. A multivariate based threshold ring signature scheme (Схема многовариантных пороговых кольцевых подписей). *Applicable Algebra in Engineering, Communication and Computing*, 24(3-4):255–275, 2013.

[51] Murat Demircioglu, Sedat Akleylek, and Murat Cenk. Efficient gemss based ring signature scheme (Эффективная схема кольцевой подписи на базе gemss). *Malaysian Journal of Computing and Applied Mathematics*, 3(1):35–39, 2020.

[52] Mohamed Saied Emam Mohamed, Albrecht Petzoldt, and C RingRainbow. Efficient multivariate ring signature schemes. *IACR Cryptol. ePrint Arch.*, 2017:247, 2017.

[53] Edward Gerjuoy. Shor's factoring algorithm and modern cryptography. an illustration of the capabilities inherent in quantum computers (Алгоритм факторизации Шора и современная криптография. Иллюстрация возможностей квантовых компьютеров). *American journal of physics*, 73(6):521–540, 2005.

[54] Richard Cleve, Artur Ekert, Chiara Macchiavello, and Michele Mosca. Quantum algorithms revisited (Снова о квантовых алгоритмах). *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969):339–354, 1998.