Новый эмпирический анализ отслеживаемости в блокчейнах на базе CryptoNote

Зуокся Ю (Zuoxia Yu)[1], Мэн Хо О (Man Ho Au)[1,*], Джяньшань Ю (Jiangshan Yu)[2], Рупенг Янг (Rupeng Yang)[3,1], Кьюлянь Ксю (Qiuliang Xu)[4] и Вонг Фэт Ло (Wang Fat Lau)[1]

[1] Факультет вычислительной техники, Гонконгский политехнический университет, Гонконг
csallen@comp.polyu.edu.hk
[2] Университет Монаша, Австралия
[3] Школа вычислительной техники и технологии, Шаньдунский университет, Китай
[4] Школа программного обеспечения, Шаньдунский университет, Китай

Аннотация

Атаки, направленные на формирование каскадного эффекта (PETS' 18) на механизмы обеспечения неотслеживаемости Monero, можно предотвратить двумя способами. Первый подразумевает увеличение минимального размера кольца для каждого входа: с 3 (версия 0.9.0) до 7 при последнем обновлении (версия 0.12.0). Второй подход заключается в реализации кольцевых конфиденциальных транзакций, повышающих гарантии обеспечения анонимности. Тем не менее до настоящего времени не было проведено формального анализа уровня анонимности, обеспечиваемого новыми контрмерами, используемыми Monero. Кроме того, так как Monero является единственным примером ведущего блокчейна на базе CryptoNote, фактическая гарантия анонимности, обеспечиваемой другими похожими блокчейнами, на практике остаётся неизвестной.

В данной работе нами предлагается более сложный статистический анализ криптовалют на базе CryptoNote. В частности, нами описан новый вид атаки на механизмы обеспечения неотслеживаемости под названием атака замкнутым множеством (closed set attack). Мы доказываем, что описанный нами тип атаки является оптимальным при условии отсутствия какой-либо дополнительной информации. Другими словами, с точки зрения результата атака закрытым множеством эквивалентна атаке прямым подбором (brute force attack), когда перебираются все возможные входы и удаляются те, использование которых невозможно при условии наличия ограничений, накладываемых миксинами в каждой транзакции.

Чтобы проверить, как описанная нами атака работает в реальности, мы провели эксперимент с тремя лучшими (в соответствии с их рыночной капитализацией) криптовалютами на базе CryptoNote, а именно: с Monero, Bytecoin и DigitalNote. Так как вычислительные затраты проведения атаки замкнутым множеством непозволительно велики, нами предлагается эффективный алгоритм под названием алгоритм кластеризации (clustering algorithm), позволяющий (приблизительно реально) реализовать нашу атаку. Наш метод кластеризации в сочетании с каскадной атакой позволяет нам идентифицировать реально потраченные монеты в случае с 70,52% входов Monero, 74,25% входов Bytecoin и 96,56% входов DigitalNote.

Помимо этого, нами предлагается теоретический анализ указанной атаки замкнутым множеством, то есть, если у каждого входа в блокчейне на базе CryptoNote есть 3 миксина, и все миксины единообразно выбираются из всех существующих монет, доля успешных попыток проведения этой атаки довольно мала (примерно 2^{-19}). Учитывая, что атака замкнутым множеством эквивалентна наиболее удачной из возможных статистических атак, наши результаты демонстрируют два ключевых момента. Во-первых, текущая системная конфигурация Monero обеспечивает защиту от статистических атак, так как минимальное количество миксинов составляет 6. Во-вторых, нами определяется новый фактор повышения уровня анонимности, то есть количество непотраченных ключей. Наш анализ указывает на то, что количество миксинов во входе необязательно должно быть очень большим, если процент непотраченных ключей будет высоким.

1 Вступление

С момента появления Bitcoin в 2009 году [10] было предложено множество распределённых валют. Несмотря на это, большинство существующих валют не обеспечивают надёжной защиты анонимности. Например, в нескольких работах [11, 7, 13] было показано, что Bitcoin, самая популярная и масштабная на сегодняшний день криптовалюта, уязвим к атакам, направленным на

* *Соответствующий автор

деанонимизацию.

В рамках решения этой проблемы всё больше внимания привлекают к себе ориентированные на обеспечение анонимности криптовалюты, гарантирующий более высокий уровень конфиденциальности данных. Среди них криптовалюты на базе протокола CryptoNote, которые представляют собой наиболее удачную попытку решения указанной проблемы. Протокол CryptoNote впервые был описан в работе [14]. Акцент делался на защиту конфиденциальности и анонимности электронной валюты. С момента появления данного протокола была предложена масса вариантов его использования, включая Bytecoin, Boolberry, Dashcoin, DigitalNote, Monero и т. д. Как и в случае со многими другими распределёнными криптовалютами, CryptoNote также использует понятие транзакции для обозначения процесса траты монет. Каждая транзакция содержит несколько входов и выходов, при этом входы потребляют монеты от отправителя, а выходы передают монеты получателю. Общая сумма монет, отправляемая во входах, и общая сумма монет, передаваемых в выходы, должны быть равными. Кроме того, каждая транзакция должна быть подписана отправителем, чтобы подтвердить передачу. Это делается при помощи приватного ключа, связанного с публичным ключом (адресом)[5] монеты, которая должна быть потрачена. Более того, используется схема кольцевой подписи [12, 6], гарантирующая анонимность реальной траты каждого входа. Это криптографический примитив, позволяющий пользователю анонимно подписывать сообщение от лица группы пользователей. Следовательно, идентичность реальной траты скрыта. Все другие ложные монеты во входе называются миксинами.

Тем не менее на практике криптовалюты на базе CryptoNote страдают от недостаточности заявляемой анонимности. Недавно в двух независимых, но одновременно появившихся работах [8, 5][6], было продемонстрировано, что транзакции Monero могут быть деанонимизированы путём проведения статистического анализа. В частности, авторами было обнаружено, что большинство входов Monero имеет очень маленькое количество миксинов, а более половины выходов выплачивается вообще без каких-либо миксинов. Такие выходы без миксинов могут быть без каких-то сложностей деанонимизированы. И что ещё хуже, как только монета, выплаченная без миксина, будет выбрана в качестве миксина в другой транзакции, вход этой транзакции также подвергнется риску деанонимизации.

На основе этого простого, но всё же жизненно важного наблюдения, авторами этих двух работ были предложены сходные стратегии проведения эмпирической оценки, основанные на так называемом анализе «цепной реакции» [8] или каскадном эффекте [5]. Грубо говоря, атакующий сначала идентифицирует все входы с нулевым миксином. Так как каждый такой обнаруженный вход будет выплачен просто с использованием одного публичного ключа, публичный ключ укажет на реальное лицо, тратящее вход. Так как в случае с Monero каждый публичный ключ может использоваться лишь единожды, будет безопаснее удалить эти деанонимизированные публичные ключи в миксинах оставшихся входов. Это приведёт к появлению новых входов с нулевым миксином, и атаку можно будет проводить повторно. В соответствии с результатами экспериментов, описанных в работах [8, 5], к февралю 2017 года примерно 65% входов транзакций были входами с нулевым миксином, а каскадный эффект позволяет отследить ещё 22% входов, то есть примерно 87% всех входов Monero являются небезопасными с точки зрения анонимности пользователей.

Столкнувшись (и спрогнозировав) этот тип атак, Monero было предложено реализовать несколько противомер. Во-первых, в версии 0.9.0 (1 января 2016 г.) появилось обязательное требование, которое заключалось в том, что все входы транзакций должны иметь по крайней мере 2 миксина. Впоследствии в версии 0.10.0 (19 сентября 2016 г.) были реализованы кольцевые конфиденциальные транзакции (RingCT), которые должны были ещё больше повысить уровень анонимности пользователей за счёт сокрытия суммы транзакции. Дополнительным преимуществом RingCT являлось то, что все входы RingCT в качестве миксинов должны были использовать выходы RingCT, то есть ни один из публичных ключей, использованных до выхода версии 0.10.0, не мог быть выбран в качестве миксина для входа RingCT. Поэтому ни атака, направленная на вызов цепной реакции, ни каскадная атака не работают в случае с RingCT. Кроме того, после того как было определено, какое влияние оказывает количество миксинов, в версии 0.12.0 (29 марта 2018 г.) оно было увеличено с 2 до 6.

---

5 [5]В данной работе мы равнозначно используем термины монета, выход и публичный ключ.

6 [6]Также недавно появилась работа [9], являющаяся обновлённой версией работы [8]. Но как метод, так и результаты анализа отслеживаемости в этих двух работах схожи, поэтому мы фокусируем внимание на начальной версии.

Нет никакого сомнения в том, что известные виды атак были предупреждены Monero, но более фундаментальная проблема сохраняется, и она представляет собой угрозу для всех криптовалют на базе CryptoNote. А именно, может ли анонимность пользователей быть защищена в достаточной мере при текущем размере кольца, то есть каковы противомеры для предупреждения известных видов атак? Связанный с этим вопрос сводится к проблеме теоретического анализа уровня безопасности, достигнутого теми криптовалютами, которые для обеспечения неотслеживаемости используют кольцевые подписи. Кроме того, какой уровень анонимности достигается криптовалютами на базе CryptoNote на практике?

Наш вклад

В данной работе нами даются ответы на поставленные выше вопросы. Во-первых, мы демонстрируем, что существующие на данный момент противомеры, позволяющие противостоять известным видам атак, обеспечивают хороший уровень анонимности в рамках системы Monero. Тем не менее с отрицательной стороны мы показываем, что другие протоколы на базе CryptoNote по-прежнему уязвимы для тех же типов атак. Фактически описываемые нами комбинированные виды атак гораздо более эффективны в случае с Bytecoin и DigitalNote, так как позволяют деанонимизировать до 91,56% транзакций в блокчейне DigitalNote.

Нами описан новый вид атаки на механизмы обеспечения неотслеживаемости криптовалют на базе CryptoNote под названием *атака замкнутым множеством*. Атака основана на том факте, что *n* входов транзакций будет и должно использовать *n* уникальных публичных ключей при реальной трате, так как каждый публичный ключ может быть взят лишь единожды. Набор входов называется *замкнутым множеством*, если количество входов равно количеству уникальных используемых публичных ключей. Следовательно, мы можем сделать вывод, что все публичные ключи, включённые в *замкнутое множество*, должны являться миксинами в других входах за пределами *замкнутого множества*. Таким образом, поиск *замкнутых множеств* поможет отследить реальную трату некоторых других входов. В отличие от атаки, направленной на создание каскадного эффекта, которая полагается на «анализ цепной реакции», возможный благодаря входам с нулевым миксином, *атака замкнутым множеством* предполагает более глубокое отслеживание без использования каких-либо ранее отслеженных входов.

Вклад данной работы можно разделить по следующим аспектам:

1. Нами даётся определение *атаки замкнутым множеством* на механизмы обеспечения неотслеживаемости блокчейнов на базе CryptoNote, а также приводится доказательство того, что *атака замкнутым множеством* является *оптимальной*. В частности, она может определить минимальное количество миксинов для каждого входа, то есть она удаляет все публичные ключи, выплаченные где-либо ещё в миксинах, идентично тому, как это происходит в случае с результатами атаки прямым подбором.
1. Нами проверяются результаты проведения нашего типа атаки. Для этого используются данные реальных блокчейнов 3 ведущих (по рыночной капитализации) криптовалют на базе CryptoNote: Monero, Bytecoin и DiditalNote. Так как вычислительные затраты проведения предлагаемого вида атаки слишком велики из-за её крайней сложности, нами предлагается эффективный алгоритм под названием алгоритм кластеризации, позволяющий (приблизительно реально) реализовать *атаку замкнутым множеством*. Нами даётся нижняя граница для нашего алгоритма кластеризации при реализации *атаки замкнутым множеством*. В частности, мы доказываем, что наш алгоритм позволяет найти все *замкнутые множества*, размер которых будет меньше или будет равен 5. Результаты экспериментов с данными блокчейнов этих трёх криптовалют приводятся в Таблице 1.

Таблица 1. Результаты экспериментов. Все входы, рассмотренные в данной работе, это входы транзакций, не являющихся транзакциями coinbase, если не указано иное. Цифры в колонке «Каск.» (соответственно «Класт.») указывают на общее количество и процент входов, отслеженных при проведении каскадной атаки (соответственно атаки кластеризацией). В колонке «Кол-во З. М.» указывается общее количество замкнутых множеств, обнаруженных алгоритмом кластеризации.

Монета	Общее кол-во блоков	Общее кол-во входов	Прослеженных (%)	Каск. (%)	Класт. (%)	Кол-во З. М.
Monero	1541236 (30.03.2018)	23164745	16334967 (70,516%)	16329215 (99,96%)	5752 (0,04%)	3017
Bytecoin	1586652 (03.08.2018)	45663011	33902808 (74,25 %)	33822593 (99,763%)	80215 (0,237%)	5912

DigitalNote	699748 (13.08.2018)	8110602	7426036 (91,56%)	7425987 (99,9993%)	49 (0,0007%)	38

2 Кроме прочего, нами также предоставлены результаты теоретического анализа возможности существования *замкнутых множеств*. Нами было обнаружено, что если у всех входов будет по 3 миксина, а все миксины будут распределены единообразно, во всех случаях, но с очень малой вероятностью (примерно 2^{-19}), *замкнутого множества* не будет. Наш анализ показал, что частота использования выходов тесно связана с анонимностью Monero. Более того, если мы сможем гарантировать, что вероятность выбора непотраченного ключа в качестве миксина будет составлять 25%, то количество миксинов каждого входа может составлять всего 3, и это сделает атаку прямым подбором просто неэффективной.

Связь с сообществом

Мы полностью раскрыли наши результаты соответствующим исследовательским сообществам, включая сообщества CryptoNote, Monero, Bytecoin и DigitalNote. Мы узнали, что исследователи Monero одновременно с нами и независимо наблюдали подобные атаки [3], и метод отрицательной маркировки (blackball tool), разработанный членами сообщества Monero, позволяет идентифицировать часть замкнутых множеств, которые мы описываем в настоящей работе. В случае с Monero метод отрицательной маркировки и наш анализ не выявили ни одной пострадавшей транзакции RingCT. Мы выложим наш код в открытый репозиторий, а работа с Monero позволит улучшить их метод и повысить уровень анонимности пользователей.

1 Предварительные данные

Протокол CryptoNote

Протокол CryptoNote [14] направлен на повышение уровня анонимности пользователей криптовалют за счёт обеспечения двух следующих свойств:

- Неотслеживаемость. В случае с любой транзакцией реальная трата должна быть анонимной со всеми множествами выходов во входе.
- Несвязываемость. В случае с любыми двумя транзакциями невозможно доказать, что они были отправлены одному и тому же пользователю.

Чтобы гарантировать несвязываемость каждого выхода транзакции, протокол CryptoNote использует одноразовый случайный ключ в качестве адреса назначения, а этот ключ выводится из публичного ключа получателя и случайных данных отправителя. Таким образом, только получатель, у которого будет постоянный секретный ключ, сможет извлечь такой выход. Для обеспечения неотслеживаемости протокол CryptoNote использует кольцевые подписи, которые являются примитивами, позволяющими пользователю анонимно подписывать транзакцию от лица группы пользователей и называемыми обычно кольцом. Поэтому реальная трата будет скрыта при помощи других выходов, которые называются миксинами. Очевидно, что для входа с n публичных ключей количество миксинов должно составлять n-1.

Обозначения

Мы используем $[m]$ для обозначения целочисленного множества $\{1, 2, \dots, m\}$. Для любого множества S мы используем $|S|$ для обозначения его размера. В случае с транзакцией tx мы используем $tx.in$ для обозначения входа этой транзакции, который является множеством публичных ключей $\{pk_1, pk_2, \dots, pk_n\}$, используемым для создания кольцевой подписи. В данной работе мы также взаимозаменяем и называем каждый вход $tx.in$ транзакции кольцом R. В частности, мы используем $R = \{pk_1, pk_2, \dots, pk_n\}$ для обозначения входа транзакции, включающего публичные ключи $pk_1, pk_2, \dots, pk_n$.

Для нашего анализа также необходимо использовать границу Чернова. Существуют различные формы границы Чернова. В данной работе нами использована форма из работы [4].

Лемма 1 (границы Чернова)

Допустим, *[формула]*, где $X_i = 1$ с вероятностью p_i и $X_i = 0$ с вероятностью $1 — p_i$, а также все X_i являются независимыми. Допустим, *[формула]*, тогда

$$\Pr[X \geq (1+\delta)\mu] \leq e^{-\frac{\delta^2}{2+\delta}\mu} \; for \; all \; \delta > 0;$$

$$\Pr[X \leq (1-\delta)\mu] \leq e^{-\frac{\delta^2}{2}\mu} \; for \; all \; 0 < \delta < 1.$$

2 *Атака замкнутым множеством*

В данном разделе нами представлен наш вариант *атаки замкнутым множеством*. Все атаки, рассмотренные в этом разделе, подразумевают доступ к транзакциям в блокчейне валюты на базе CryptoNote без каких-либо дополнительных активных действий. Этот допуск является действительным, поскольку все транзакции в блокчейне находятся в свободном доступе.

Мы доказываем, что предлагаемый нами вид *атаки замкнутым множеством* является оптимальным, то есть атакой прямым подбором. Забегая вперёд, можно сказать, что атака прямым подбором проверяет все возможные назначения входов плательщиком и удаляет те, у которых присутствует конфликт данных. Как наш вариант атаки, так и атака прямым подбором возвращают минимальное множество кандидатов в реальные плательщики для каждого входа.

3.1 *Атака прямым подбором*

Атака прямым подбором является атакой, при проведении которой проверяются все возможные последовательности уникальных публичных ключей, чтобы выяснить, являются ли они действительными для назначения в качестве реальной траты для всех входов транзакции. В то время как последовательность публичных ключей является действительной, если она удовлетворяет следующие требования: 1) размер последовательности равен общему количеству входов транзакции в наборе данных; 2) все публичные ключи в последовательности являются уникальными; 3) для всех индексов в этой последовательности публичный ключ i принадлежит соответствующему входу i в наборе данных. Другими словами, атака прямым подбором представляет собой процесс поиска всех действительных последовательностей среди перестановок публичных ключей определённой длины в соответствии с вышеуказанными требованиями. Мы называем все элементы, входящие в индекс i ($i \leq$ количеству всех входов) всех действительных последовательностей, кандидатами на то, чтобы являться реальной тратой входа транзакции i. Следовательно, полученные действительные последовательности являются комбинациями всех возможных реальных трат каждого входа транзакции. Кроме того, если у входа транзакции есть только один кандидат на то, чтобы являться реальным плательщиком, то кандидат и должен быть его плательщиком.

Нетрудно заметить, что атака прямым подбором является совершенной атакой, позволяющей выявить все возможные реальные траты для каждого входа транзакции. Допустим, в нашем наборе данных есть n уникальных ключей и m входов транзакций, и без потери универсальности n больше, чем m. Допустим, *[формула]* обозначает количество перестановок элементов m среди элементов n. Количество действительных последовательностей после проведения атаки прямым подбором составляет *[формула]*, где Conflicts означает множество удалённых перестановок, которые не соответствовали вышеуказанным требованиям.

3.2 *Наша атака*

Несмотря на то, что описанная выше атака прямым подбором совершенна, её сложность невероятно высока для реализации атаки на практике, а именно *[формула]*. Учитывая неэффективность и непрактичность атаки прямым подбором, нами предлагается атака под названием *атака замкнутым множеством*, более эффективная и позволяющая добиться того же результата.

Предлагаемая *атака замкнутым множеством* основана на том наблюдении, что если количество уникальных публичных ключей, входящих во множество входов транзакций, равно количеству входов во множестве, то мы можем прийти к выводу, что каждый включённый публичный ключ должен являться реальной тратой определённого входа в этом множестве, а также миксином в других внешних входах. Таким образом, нахождение *замкнутого множества* имеет как минимум два значительных последствия. Во-первых, другие входы становятся отслеживаемыми после удаления публичных ключей *замкнутого множества*. Во-вторых, средний размер входов будет снижаться, что поможет при выполнения последующей операции.

Атака замкнутым множеством представляет собой итеративный процесс, позволяющий найти

все возможные *замкнутые множества*, удалить включённые в них публичные ключи, а также найти отслеживаемые входы. В сравнении с предшествующей атакой, направленной на создание каскадного эффекта, о которой говорилось в работе [5], или же анализом «цепной реакции» из работы [8] *атака замкнутым множеством* делает отслеживаемым большее количество входов. Если быть более точными, атака, направленная на создание каскадного эффекта, использует тот факт, что входы с нулевыми миксинами повлияют на отслеживаемость других входов, которые возьмут их публичные ключи в качестве миксинов. Другими словами, эта атака основана на множестве ранее отслеженных входов и использует их для отслеживания оставшихся анонимных входов. В то же самое время наша атака может быть начата с любого анонимного входа.

Если донести суть нашей атаки, ниже мы приводим короткий пример. В данном случае мы рассмотрим четыре входа, включённых в транзакции *[формула]*, и допустим, что есть четыре уникальных публичных ключа *[формула]*, включённых во множества входов, состоящих из них, то есть:

$$tx_1.in = \{pk_1, pk_2, pk_3\};$$
$$tx_2.in = \{pk_2, pk_3\};$$
$$tx_3.in = \{pk_1, pk_3\};$$
$$tx_4.in = \{pk_1, pk_2, pk_3, pk_4\}.$$

Следует отметить, что не должно существовать никакого другого входа транзакции, который бы состоял только из публичных ключей среди *[формула]*. В противном случае будет нарушен принцип построения Monero, предполагающий, что выход может быть возмещён лишь единожды. В то же самое время оригинальная каскадная атака, описанная в работах [8, 5], в данном случае не работает, так как входы с нулевыми миксинами отсутствуют.

Несмотря на то, что мы не можем сделать все вышеупомянутые входы отслеживаемыми, мы можем отследить реальную трату одного из них. В частности, рассмотрим множество *[формула]*. В нём заключено единое множество уникальных публичных ключей $\{pk_1, pk_2, pk_3\}$. Очевидно, что размер S равен количеству уникальных публичных ключей, которые в нём содержатся, таким образом, оно является *замкнутым множеством*. Так как каждый выход может быть потрачен только один раз, выход *[формула]* должен быть реальной тратой определённой транзакции *[формула]*. Таким образом, мы можем прийти к выводу, что реальной тратой tx_4 является pk_4.

Простой вариант реализации

Простой метод нахождения всех *замкнутых множеств* состоит в переборе всех возможных подмножеств входов транзакций. В случае с каждым подмножеством мы проверяем, является оно *замкнутым множеством* или нет, сравнивая количество входов с количеством уникальных публичных ключей, содержащихся в нём. Если является, то мы проводим операции удаления и отслеживания для этого *замкнутого множества*. В противном случае процесс продолжается до тех пор, пока не будут проверены все подмножества. Из-за ограниченности места этот алгоритм приводится нами в Приложении А.

Теоретически этот алгоритм позволяет найти все замкнутые множества, включённые во все входы транзакций. Тем не менее его реализация в реальности требует больших затрат из-за сложности перебора всех подмножеств входов, которая может быть выражена как $\theta(2^m)$, где m является общим количеством всех входов транзакций, содержащихся в блокчейне. Например, до блока 1541236 Monero количество неотслеживаемых входов составило 6 835 530 после проведения каскадной атаки. Начиная с этих входов, на этапе поиска подмножеств, размер которых составлял 5, сложность алгоритма составит *[формула]*.

Анализ атаки замкнутым множеством

Мы доказываем, что описываемая нами *атака замкнутым множеством* является оптимальной. Другими словами, наша *атака замкнутым множеством* эквивалентна атаке прямым перебором. В частности, мы доказываем, что после проведения нашей *атаки замкнутым множеством* каждый вход транзакции является множеством кандидатов на реальную его трату, обнаруженным при реализации атаки прямым перебором. Анализ заканчивается следующей теоремой.

Теорема 1. *Вышеупомянутая атака замкнутым множеством эквивалентна атаке прямым перебором. Другими словами, для любого множества транзакций результат реализации нашей*

атаки будет идентичен результату реализации атаки прямым перебором.

Из-за ограниченности места мы предлагаем читателю ознакомиться с полной версией этой работы, в которой приводится доказательство данной теоремы.

3.3 По вопросу существования замкнутых множеств: теоретическая перспектива

Как было сказано выше, атака замкнутым множеством является оптимальной. Скажем, мы можем прийти к выводу, что анонимность входов не может быть снижена при отсутствии *замкнутого множества*. В данном разделе нами оценивается вероятность существования по крайней мере одного *закрытого множества* при идеальном сценарии, а именно, когда у всех входов есть (небольшое) постоянное количество миксинов, и все миксины единообразно выбираются из всех ключей.

Более конкретно нами рассматривается сценарий, при котором:

- существует $6 \cdot 2^{20}$ входов с $6 \cdot 2^{20}$ реально расходуемыми публичными ключами;
- также существует 25% (то есть, $2 \cdot 2^{20}$) непотраченных публичных ключей;
- каждый вход имеет 3 миксина;
- каждый миксин единообразно выбирается из всех $8 \cdot 2^{20}$ ключей;

где первые два условия взяты из реальных данных Monero после проведения каскадной атаки, а третье условие основано на том факте, что средний размер кольца после каскадной атаки составляет 4,62.

Лемма 2. *Со всей, но малой вероятностью* 2^{-19} *в вышеуказанном наборе данных нет никакого замкнутого множества, если у всех выходов имеется по 3 миксина, и все миксины единообразно выбираются из всех ключей*.

Доказательство данной леммы приводится в полной версии данной работы.

3 Наш алгоритм кластеризации

Учитывая непрактичность алгоритма на базе подмножеств, о котором говорилось выше, нами предлагается приблизительный, но эффективный алгоритм поиска *замкнутых множеств* под названием «алгоритм кластеризации». Забегая вперёд, несмотря на то, что алгоритм кластеризации является только приблизительным алгоритмом, мы демонстрируем, что нижняя граница размера *замкнутых множеств*, которые он позволяет обнаружить, равна 5. Другими словами, все *замкнутые множества* размером менее или равном 5 будут найдены. Кроме того, нами были проведены эксперименты, которые показали, что наш алгоритм кластеризации позволяет достигнуть более высоких результатов при фактическом выполнении.

Интуитивные свойства нашего алгоритма кластеризации

Вспомним, что основным свойством *замкнутого множества* является то, что оно включает в себя одинаковое количество входов и уникальных публичных ключей. Следовательно, наша цель состоит в том, чтобы найти множество входов с вышеуказанными характеристиками. Для этого существует один интуитивный способ, который заключается в формировании множества на основе определённого входа с последующим поглощением другого входа. Это помогает получить *замкнутое множество*. Ключевая проблема заключается в том, как выбрать другие кольца.

Нами было подмечено, что поскольку конечная цель состоит в том, чтобы две цифры в данном множестве были равными, можно выбирать кольца на основе последовательности добавления входов во множество. Например, допустим, что рассматриваемое нами множество, обозначенное как S, начинается со входа R. Всякий раз, когда во множество S будет добавляться вход R', будет существовать три варианта возможных последствий:

- Случай 1. Если все публичные ключи в R' являются подмножеством всех публичных ключей, содержащихся в S, то для множества S количество включённых входов транзакций будет увеличиваться на единицу, а количество уникальных публичных ключей будет оставаться тем же. Таким образом, добавление R' определённо повысит вероятность того, что S является *замкнутым множеством*. Мы называем такой вход «полезным

входом».

- Случай 2. Если добавление R' приведёт к добавлению только одного уникального публичного ключа в S, то добавление этого входа не изменит текущего соотношения количества уникальных публичных ключей с количеством входов во множестве S. Такой вид входов увеличивает множество публичных ключей в S, что может быть полезно для поглощения других входов. Мы называем такой вход «неопределённым входом».
- Случай 3. Если добавление R' приведёт к добавлению двух или большего количества уникальных публичных ключей во множество S, то количество входов увеличится всего на один, но количество публичных ключей возрастёт на 2 или более. Так как это совершенно не поможет нам в нашем анализе, мы назовём этот вид входа «плохим входом».

Кроме всего прочего, если мы включаем в множество только относительно полезные и неопределённые входы, то мы можем найти замкнутое множество быстрее и с более высокой вероятностью.

Определение кластера

Кластер *Clus* определяется как множество входов, а именно $Clus = \{R_1, R_2, \dots, R_n\}$. Каждый кластер представляет множество *PK_Clus*, которое определяется как *[формула]*. Другими словами, *PK_Clus* является множеством, используемым для сбора всех уникальных публичных ключей, входящих во входы *Clus*.

Расстояние от входа до кластера определяется как количество публичных ключей, входящих во вход, но не в кластер. Формальное определение приводится ниже:

$$Dist(R, Clus) = Dist(R, PK\_Clus) = |R| - |PK\_Clus \cap R|,$$

где R является входом, который должен быть добавлен, а *Clus* обозначает кластер со множеством публичных ключей *PK_Clus*. Примечательно, что данное определение не симметрично. Согласно нашему определению, расстояние от входа до кластера, то есть *Dist(R, Clus)*, отличается от расстояния от кластера до входа, то есть *Dist(Clus, R)*.

Рассмотрим, например, кластер *Clus* и вход R, составленные следующим образом:

$$Clus = \{\{pk_1, pk_2\}, \{pk_1, pk_3\}, \{pk_2, pk_4\}\},$$
$$R = \{pk_1, pk_3, pk_5\}.$$

Очевидно, что множеством публичных ключей *Clus* является $PK\_Clus = \{pk_1, pk_2, pk_3, pk_4\}$. Размер R равен 3, а количество общих публичных ключей — 2. Следовательно, в соответствии с нашим определением, расстоянием от R до *Clus* будет $Dist(R, Clus) = Dist(R, PK\_Clus) = 3 - 2 = 1$. Таким образом, если мы добавим R в *Clus*, то в кластер *Clus* также будет добавлен только один вход, то есть pk_5.

Начиная с определённого входа, построение кластера представляет собой динамический процесс поиска других подходящих входов. Чтобы прояснить, какой вид входов может быть поглощён кластером, мы связываем каждый кластер с расстоянием. Если говорить более точно, мы называем кластер *Clus* с расстоянием 1, только если эти входы, удовлетворяющие условию $Dist(R, Clus) \leq 1$, могут быть добавлены в него. Так как операция добавления может вызвать изменения в кластере, нам всегда необходимо подтверждать существующий кластер, чтобы вычислять расстояние от входа до него. Алгоритм построения кластера, начиная с определённого входа, показан ниже.

Алгоритм 1. *Cluster_Form*(R)

1: Начать со входа R и определить кластер как $Clus = \{R\}$
2: Допустим, *DataSet* является всеми входами транзакций в блокчейне
3: **для** каждого *[формула]* **выполнить**
4: **если** $Dist(R', Clus) \leq 1$, **то**
5: *[формула]*
6: **результат** *Clus*

Для каждого кластера нами используется два дополнительных параметра, позволяющих

проверить, является ли он *замкнутым множеством*. Одним из параметров является количество входов, входящих в него, а другим — количество входящих в него же уникальных публичных ключей. Формально, если количество входов равно количеству уникальных публичных ключей, входящих в кластер, то мы можем сказать, что данный кластер является *замкнутым множеством*. Кроме того, в некоторых случаях *замкнутое множество* может содержать другие *замкнутые подмножества*. Чтобы найти все *замкнутые множества*, всякий раз, когда мы используем для этого данный алгоритм, мы выполняем операцию поиска *замкнутых подмножеств*. Важно, что если публичный ключ лишь единожды появляется в *замкнутом множестве*, то он будет реальной тратой входа, содержащего его. Для простоты мы используем этот метод для проверки на предмет наличия *замкнутого подмножества* в *замкнутом множестве*, так как сложность прямого перебора всех подмножеств данного *замкнутого множества* довольно велика.

Затем мы применяем наш алгоритм кластеризации ко всем кластерам с расстоянием равным 1. Основная идея состоит в том, чтобы повторно пройти все входы транзакций за множество итераций. При каждой итерации алгоритм выбирает вход и использует его для создания кластера *Clus*. Затем мы запускаем алгоритм построения кластера (Алгоритм 1), чтобы добавить подходящие входы в *Clus*. Очередная итерация продолжается до тех пор, пока полученный кластер не окажется *замкнутым множеством*. В противном случае перед началом очередной итерации, алгоритм должен закончить следующие операции. Удалить все публичные ключи, содержащиеся в этом кластере, из остающихся входов и найти отслеживаемые. После этого мы проверяем, содержит или нет текущее *замкнутое множество* публичный ключ, появляющийся только в одном входе. Если да, мы продолжаем деанонимизацию входов внутри *замкнутого множества*.

Алгоритм поиска всех кластеров с расстоянием равным 1 среди входов всех транзакций в блокчейне приводится ниже (Алгоритм 2). Следует отметить, что все кольца, которые рассматриваются в случае с нашим алгоритмом, являются анонимными. Как только будет обнаружена реальная трата входа, мы не совершаем какой-либо операции с таким входом. Кроме того, наш алгоритм концентрируется на полученных после проведения каскадной атаки данных. Таким образом, в Алгоритме 2 мы неправильно используем концепцию, предполагающую использование прежде всего вызова, создающего каскадный эффект.

Алгоритм 2. Алгоритм кластеризации

```
1:  Допустим, DataSet является всеми входами транзакций в блокчейне
2:  Cascade-Effect (Dataset)
3:  Flag = true
4:  если Flag = = true, выполнить
5:          Flag = false
6:      для каждого [формула] выполнить
7:              Clus_Form(R) → Clus
8:              если Clus является замкнутым множеством, то
9:                  Remove(Clus) → Flag
10:                 если Flag = = true, то
11:                     найти отслеживаемые входы
12:                    проверить, являются ли кольца внутри Clus отслеживаемыми
```

Анализ точности

Точность алгоритма кластеризации анализируется при помощи следующей теоремы, которая определяет нижнюю границу алгоритма кластеризации. К слову, все *замкнутые множества*, размер которых меньше или равен 5, могут быть найдены после реализации алгоритма кластеризации.

***Теорема 2**. После выполнения нашего алгоритма кластеризации с поиском расстояния 1 все неделимые замкнутые множества с размером менее или равным 5 могут быть возвращены нашим алгоритмом.*

Мы предлагаем читателю ознакомиться с полной версией этой работы, в которой приводится доказательство данной теоремы.

Анализ сложности

Допустим, что общее количество входов транзакций в блокчейне является N. Количество итераций по нашему алгоритму будет $\theta(N)$. Предположим, что средней длиной входа является ℓ. При каждой итерации в худшем случае мы вычисляем *[формула]* раз расстояние между всеми входами и текущими кластерами. Следовательно, в худшем случае сложность будет $\theta(\ell N^2)$.

4 Результаты эксперимента

Чтобы оценить уровень анонимности криптовалют на базе CryptoNote, а также оценить вероятность существования *замкнутых множеств* в реальности, мы реализовали наш алгоритм на языке C++, и программа работала на компьютере с процессором Intel Core i5 3,1 ГГц, 16 Гб ОЗУ и SSD-накопителем объёмом 256 Гб. Следует отметить, что нами были проанализированы только три лучшие (согласно их рыночной капитализации) криптовалюты на базе CryptoNote, а именно, Monero, Bytecoin и DigitalNote. В случае со всеми тремя валютами нами были экспортированы все необходимые данные напрямую из соответствующей базы данных блокчейна путём изменения её исходного кода.

5.1 Анализ Monero

Так как уже существует две предшествующие работы [8, 5], в которых было рассмотрено влияние атак, направленных на создание каскадного эффекта, на неотслеживаемость транзакций Monero, мы в основном сконцентрируемся на анализе данных после проведения известных атак.

Сбор данных

Мы собрали все блоки из блокчейна Monero, начиная с первого блока (18 апреля 2014 г.) и заканчивая блоком 1541236 (30 марта 2018 г.). Помимо этого, все необходимые данные были напрямую экспортированы из соответствующей базы данных блокчейна путём изменения исходного кода Monero [2]. Наш набор данных в совокупности включил в себя 4 153 307 транзакций. Среди них 2 612 070 не являлись coinbase-транзакциями, что составило всего 23 164 745 входов транзакций и 25 126 033 уникальных публичных ключа. Следует отметить, что в данной работе нами рассмотрены только транзакции, не являющиеся coinbase-транзакциями, если не указано иное.

Результаты эксперимента

В Таблице 2 нами приводятся результаты применения алгоритма кластеризации к вышеупомянутому набору данных. Оказывается, 16 334 967 входов становятся отслеживаемыми. В частности, 16 329 215 входов стали отслеживаемыми в результате проведения каскадной атаки, а оставшиеся входы, то есть всего 5752, были отслежены при поиске *замкнутого множества*. Всего отслеживаемыми являются 70,52% входов транзакций Monero. В то же самое время после проведения каскадной атаки с использованием этого же набора данных отследить удалось только 0,084%

Таблица 2. Отслеживаемость Monero

Кол-во миксинов	Всего	Выводимых	Каскадная атака	Алгоритм кластеризации	%
0	12 209 675	12 209 675	12 209 675	0	100
1	707 786	625 641	625 264	377	88,39
2	4 496 490	1 779 134	1 776 192	2942	39,57
3	1 486 593	952 855	951 984	871	64,10
4	3 242 625	451 959	451 230	729	13,94
5	319 352	74 186	73 980	206	23,23
6	432 875	202 360	202 100	260	46,75
7	21 528	4296	4282	14	19,96
8	30 067	3506	3490	16	11,66
9	17 724	2178	2162	16	12,29
≥ 10	200 030	29 177	28 856	321	14,59
Итого	23164745	16334967	16329215	5752	70,52

Кроме того, всего 6 829 778 входов транзакций по-прежнему невозможно отследить. Для всех таких остающихся входов мы определили частоту количества миксинов до и после выполнения алгоритма кластеризации, как показано на рисунке 1.

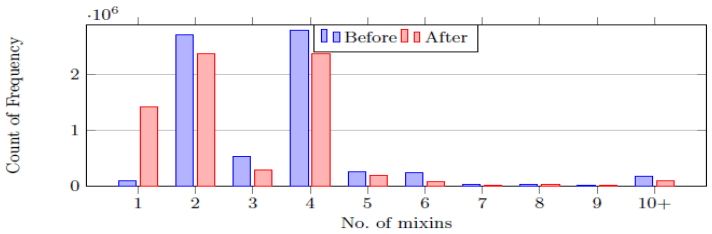

Рис. 1. Частота количества миксинов анонимных входов до и после выполнения алгоритма кластеризации

Count of frequency	Частота
No. of mixins	Количество миксинов
Before	До
After	После

Алгоритм кластеризации также позволил найти 3017 уникальных *замкнутых множеств*, размер которых находился в диапазоне от 2 до 55, и которые включали в себя всего 7478 уникальных публичных ключей. Как нами уже упоминалось ранее, эти 7478 публичных ключей должны являться реальной тратой определённого входа, который содержится в этом *замкнутом множестве*. Другими словами, мы можем прийти к выводу, что они были потрачены, несмотря на то, что нам неизвестно, в какой конкретно транзакции они использовались. Тем не менее с точки зрения анонимности бесполезно, если какой-либо другой новый вход возьмёт из них публичные ключи.

Кого-то может удивить расхождение между вероятностью нахождения *замкнутого множества*, составляющей 2^{-19}, и наличием 3017 *замкнутых множеств*, обнаруженных при проведении эксперимента. Причиной этого является тот факт, что наш анализ подразумевает, что миксины выбираются единообразно, и что у каждого входа есть по три миксина. Однако на практике выборка распределения и количества миксинов для всех входов не является единообразной. Это немного повышает вероятность нахождения *замкнутого множества*.

5.2 Анализ Bytecoin

Нами приводится анализ неотслеживаемости Bytecoin после проведения каскадной атаки и атаки кластеризацией.

Сбор данных

Нами были собраны все блоки в блокчейне Bytecoin, начиная с первого блока (4 июля 2012 г.) и заканчивая блоком 1586652 (3 августа 2018 г.). В этом наборе данных содержалось всего 3 782 566 транзакций, не являвшихся coinbase-транзакциями, и они в совокупности включали в себя 45 663 011 входов транзакций. Кроме того, в набор данных входило 48 613 764 уникальных публичных ключа.

Результаты эксперимента

В Таблице 3 приводятся результаты эксперимента, проведённого с набором данных Bytecoin. В частности, 33 902 808 входов транзакций Bytecoin стали отслеживаемыми, что составило 74,25% от всех входов из нашего набора данных. Среди них 28 591 486 входов являлись входами с нулевым миксином, и каскадный эффект, вызванный ими, сделал отслеживаемыми ещё 5 231 107 входов, что составляет 99,763% от всех отслеживаемых входов. Кроме того, наш алгоритм кластеризации позволил отследить ещё 80 215 входов транзакций из остающихся, что составляет 0,68% от тех неотслеживаемых входов, которые остались после проведения каскадной атаки. Всего было обнаружено 5912 *замкнутых множества*, размер которых находился в диапазоне от 2 до 55.

Таблица 3. Отслеживаемость Bytecoin

Кол-во миксинов	Всего	Выводимых	Каскадная атака	Алгоритм кластеризации	%
0	28 591 486	28 591 486	28 591 486	0	100
1	5 751 268	3 281 500	3 240 142	41 358	57,06
2	2 840 745	1 133 602	1 112 648	20 954	39,91
3	1 442 133	261 197	260 298	899	18,11
4	2 516 851	276 237	275 172	1065	10,98
5	617 041	59 922	59 493	429	9,71
6	3 145 092	270 355	255 156	15 199	8,60

7	388 759	26 434	26 160	274	6,80
8	81 504	1231	1220	11	1,51
9	65 379	397	389	8	0,61
≥ 10	222 753	447	429	18	0,2
Итого	45663011	33902808	33822593	80215	74,25

5.3 Анализ DigitalNote

Также нами впервые был проведён анализ неотслеживаемости DigitalNote.

Сбор данных

Нами были собраны все 633 548 транзакций, не являющиеся coinbase-транзакциями, входившими в блоки, начиная с блока 1 (31 мая 2014 г.) по блок 699748 (13 августа 2018 г.), входящих в блокчейн DigitalNote. Вышеуказанные транзакции включали в себя всего 8 110 602 входов и 8 396 472 уникальных публичных ключа.

Результаты эксперимента

В Таблице 4 приводятся результаты эксперимента, проведённого с набором данных DigitalNote. Удалось отследить 91,56% всех входов транзакций, при этом 60,39% из них не имели миксинов. Кроме того, в результате каскадной атаки добавилось 39,60% отслеживаемых входов. Наш алгоритм кластеризации сделал отслеживаемыми ещё 49 входов, что составило 0,007% от неотслеживаемых входов, оставшихся после проведения каскадной атаки. Это было сделано за счёт выявления 38 *замкнутых множеств*.

Таблица 4. Отслеживаемость DigitalNote

Кол-во миксинов	Всего	Выводимых	Каскадная атака	Алгоритм кластеризации	%
0	4 484 726	4 484 726	4 484 726	0	100
1	2 087 295	1 847 151	1 847 132	19	88,49
2	1 194 410	895 480	895 472	8	74,97
3	129 700	101 872	101 872	0	78,54
4	6225	4362	4358	4	70,07
5	193 669	85 941	85 939	2	44,38
6	3071	1840	1837	3	59,92
7	844	442	440	2	52,38
8	1686	856	853	3	50,77
9	1288	682	681	1	52,95
≥ 10	7688	2684	2677	7	34,91
Итого	8110602	7426036	7425987	49	91,56

5 Наблюдения и рекомендации

В этом разделе мы делимся своими наблюдениями и даём рекомендации в соответствии с результатами, проведённых нами экспериментов.

- ***Наблюдение 1***. *Частота использования выходов является важным фактором обеспечения анонимности криптовалют на базе CryptoNote*. Частота использования выходов относится к проценту потраченных публичных ключей, которые легко вычислить, используя общую сумму входов в наборе данных по общему количеству уникальных выходов (то есть публичных ключей), так как каждый выход может быть выплачен лишь единожды. Как уже было сказано в Разделе 3.3, такие непотраченные публичные ключи играют важную роль с точки зрения предотвращения формирования *замкнутого множества*. Поэтому будет честным сказать, что в некоторой степени снижение частоты использования повысит уровень анонимности.
- ***Наблюдение 2***. *Замкнутые множества тесно связаны с анонимностью входов*. В данной работе нами было продемонстрировано, что нахождение *замкнутых множеств* может помочь идентифицировать реальные траты или сократить размер кольца (а также уровень анонимности) во входах. Несмотря на то, что вероятность существования *замкнутых множеств* невысока, они существуют и угрожают анонимности входов.
- ***Рекомендация 1***. *Снизить частоту использования выходов путём создания большего*

количества выходов. Следует помнить о том, что низкая частота использования выходов способствует обеспечению анонимности входов Monero. Следовательно, чтобы снизить частоту использования выходов, мы рекомендуем пользователям генерировать несколько дополнительных выходов с нулевой суммой, что сделает множество непотраченных выходов больше.

- ***Рекомендация 2***. *Не используйте бесполезные миксины*. Возьмём Monero в качестве примера. Наш алгоритм кластеризации позволил найти 3017 уникальных *замкнутых множества*, содержавших 7478 уникальных публичных ключей. Эти 7478 уникальных публичных ключа должны являться реальной тратой определённого входа, входящего в эти *замкнутые множества*. Следовательно, при создании любого нового входа использование этих ключей в качестве миксинов не повысит уровня анонимности. Поэтому мы рекомендуем пользователям не использовать эти бесполезные миксины. Тем не менее обычному пользователю сложно определить, содержится выход в *замкнутом множестве* или нет. Поэтому мы опубликуем наш код, который позволит реализовать атаку.

Благодарность

Мы благодарим анонимных редакторов за их работу и ценные предложения. Часть данной работы была поддержана Национальным фондом естественных наук Китая (Грант № 61602396, U1636205, 61572294, 61632020), Объединённой лабораторией блокчейн технологий и криптовалют MonashU-PolyU-Collinstar Capital, Советом по исследовательским грантам Гонконга (Грант № 25206317) и Национальным исследовательским фондом Люксембурга (FNR) за Грант PEARL FNR/P14/8149128.

Ссылки

1 Cryptocurrencies market capacity (Объемы криптовалютного рынка). https://coinmarketcap.com/all/views/all/, онлайн ресурс, использован 16 апреля 2018 г.

2 Monero source code (Исходный код Monero). https://github.com/monero-project/monero, онлайн ресурс, использован 30 марта 2018 г.

3 Sets of spent outputs (Множества потраченных выходов). https://ww.getmonero.org/resources/research-lab/pubs/MRL-0007.pdf, онлайн ресурс, использован 26 ноября 2018 г.

4 М. Гёманс (M. Goemans). Lecture notes in chernoff bounds, and some applications (Примечания к лекции по границам Чернова и некоторые варианты их применения), февраль 2015 г. http://math.mit.edu/~goemans/18310S15/chernoff-notes.pdf.

5 А. Кумар (A. Kumar), С. Фишер (C. Fischer), С. Топл (S. Tople) и П. Саксена (P. Saxena). A traceability analysis of monero's blockchain (Анализ отслеживаемости блокчейна Monero). In European Symposium on Research in Computer Security (материалы Европейского симпозиума по исследованиям в области компьютерной безопасности), страницы 153-173. Springer, 2017 г.

6 Дж. К. Лю (J. K. Liu), В. Сусило (W. Susilo) и Д. С. Вонг (D. S. Wong). Ring signature with designated linkability (Кольцевая подпись с обозначенной связываемостью). In International Workshop on Security (материалы Международного семинара по безопасности), страницы 104-119. Springer, 2006 г.

7 С. Мекледжон (S. Meiklejohn), М. Помароле (M. Pomarole), Дж. Джордан (G. Jordan), К. Левченко (K. Levchenko), Д. Маккой (D. McCoy), Дж.. М. Вёлькер (G. M. Voelker) и С. Сэведж (S. Savage). A fistful of bitcoins: characterizing payments among men with no names (Пригоршня Bitcoin: идентификация платежей, производимых людьми без имени). In Proceedings of the 2013 conference on Internet measurement conference (Протоколы конференции по проведению измерений в сети Интернет 2013), страницы 127-140. ACM,

2013 г.

8 А. Миллер (A. Miller), М. Мёзер (M. Moser), К. Ли (K. Lee) и А. Нараянан (A. Narayanan). An empirical analysis of linkability in the monero blockchain (Эмпирический анализ связываемости в блокчейне Monero). arXiv сигнальный экземпляр arXiv:1704.04299, 2017 г.

9 М. Мёзер (M. Moser), К. Соска (K. Soska), Е. Хейлман (E. Heilman), К. Ли (K. Lee), Х. Хин (H. Hean), С. Сривастава (S. Srivastava), К. Хоган (K. Hogan), Дж. Хеннеси (J. Hennessey), А. Миллер (A. Miller), А. Нараянан (A. Narayanan) и др. An empirical analysis of traceability in the monero blockchain (Эмпирический анализ отслеживаемости в блокчейне Monero). Proceedings on Privacy Enhancing Technologies (Доклады по технологиям повышения анонимности), 3:143{163, 2018 г.

10 С. Накамото (S. Nakamoto). Bitcoin: A peer-to-peer electronic cash system (Bitcoin: электронная одноранговая денежная система). 2008 г.

11 Ф. Рейд (F. Reid) и М. Харриган (M. Harrigan). An analysis of anonymity in the bitcoin system (Анализ анонимности в системе Bitcoin). In Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on (материалы конференции по анонимности, безопасности, рискам и доверительному управлению (PASSAT) и третьей международной конференции IEEE по социальным вычислительным технологиям 2013 (SocialCom), а также третьей международной конференции по... 2011), страницы 1318-1326. IEEE, 2011 г.

12 Р. Л. Ривест (R. L. Rivest), А. Шамир (A. Shamir) и Ю. Тауман (Y. Tauman). How to leak a secret (Как украсть секрет). In International Conference on the Theory and Application of Cryptology and Information Security (материалы Международной конференции по прикладной криптографии и информационной безопасности), страницы 552-565. Springer, 2001 г.

13 Д. Рон (D. Ron) и А. Шамир (A. Shamir). Quantitative analysis of the full bitcoin transaction graph (Количественный анализ полного графа транзакций Bitcoin). In International Conference on Financial Cryptography and Data Security (материалы Международной конференции по финансовой криптографии и Безопасности данных), страницы 6-24. Springer, 2013 г.

14 Н. Ван Саберхаген (N. Van Saberhagen). Cryptonote v 2.0, 2013 г.

A Алгоритм на базе подмножеств

Здесь нами приводится простой алгоритм поиска всех *замкнутых множеств* путём обнаружения подмножеств входов транзакций. Забегая вперёд, мы используем Cascade-Effect(inputs) для обозначения функции, которая реализует атаку, направленную на создание каскадного эффекта. Допустим, Remove (*closed set CS*) → *flag* является функцией, которая удалит все публичные ключи, содержащиеся в *closed set CS* из других входов за пределами *CS*, и выводит переменную *flag* = *true*, если происходит какая-либо операция удаления. Алгоритм приводится ниже.

Алгоритм 3. Алгоритм поиска подмножеств

1: Допустим, *DataSet* является всеми входами транзакций в блокчейне
2: Допустим ℓ является размером текущего подмножества, и $\ell \geq 2$
3: Cascade-Effect(Dataset)
4: если $\ell \leq |Dataset|$ **выполнить**
5: Допустим, *[формула]* задано для всех входов, s.t., размер каждого входа равен или меньше, чем ℓ
6: Допустим, $\{Subset_{\ell,j}\}$ является всеми подмножествами Set_ℓ размером ℓ, где *[формула]*, и каждый *[формула]*
7: **для** $j = 1$ для *[формула]* **выполнить**

8: **если** $Subset_{\ell,j}$ является замкнутым подмножеством, **то**

9: Remove($Subset_{\ell,j}$) $\rightarrow$ *flag*

10: **если** flag = = true, **то**

11: найти отслеживаемые входы

12: **перейти** с ℓ ++