ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ

Курсовая работа

Криптографические протоколы с нулевым разглашением в криптовалютах

4 курс

Руководитель: С. Ю. Катышев
Выполнил: И. М. Ермилов

Москва, 2018 г.

Введение

В нынешних условиях люди все меньше доверяют государственным и коммерческим организациям. По этой причине не хотят, чтобы они были единственными доверенными элементами в процедуре обработки и передачи информации, принадлежащей частным лицам. Уйти от единого центра доверия можно путем использования распределенных реестров, в частности, технологии блокчейн.

На данный момент проблема анонимности или защиты персональных данных пользователя технологии блокчейн рассматривается через призму широко распространенных и активно обсуждаемых криптовалют, наиболее известной из которых является Биткоин. В этом контексте, мы будем понимать под персональными данными как сведения о пользователи (IP адрес, ФИО), так и о совершенных им транзакциях или иных действиях.

Хорошо известно, что при использовании криптовалюты Биткоин обеспечивается только "псевдонимность что подразумевает известную историю совершенных транзакций при отсутствии непосредственной идентификации пользователей.

Известны работы [2],[4],[5],[6],[12],[13] в которых представлены подходы к анализу системы Биткоин, с целью установления участников, указанных в транзакциях. В рамках данной работы мы рассмотрим "Байесовский подход"(раздел 3.1) к деанонимизации пользователя. Для защиты персональных данных предлагается использовать систему TOR, однако, в работе [2] показана слабость такого подхода. Подробнее этот аспект будет рассмотрен в разделе 3.2 работы.

С другой стороны, известны подходы, позволяющие обеспечить защиту персональных данных и передаваемой информации с использованием специального математического аппарата. В частности, использования групповых подписей, перемешивания и протоколов доказательства с нулевым разглашения.

Подходы к обеспечению защиты персональных данных, в основе которых лежит идея использования доказательств с нулевым разглашением, применяется в криптовалюте ZCash[17]. Именно последнему посвящена настоящая работа. В данной криптовалюте используется протокол zk-SNARKs[15]. Был проведен анализ кривой Баррето-Наерига, которая используется в zk-SNARKs. Для разных параметров кривой посчитана сложность решения задачи дискретного логарифмирования, необходимая для нахождения секретных параметров протокола.

1 Блокчейн и система Биткоин

Понятие Блокчейн было введено Сатоши Накамото в [1]. Именно эта статья и "породила"собой множество криптовалют, начиная от Биткоина и заканчивая малоизвестными криптовалютами. Стоит отметить, что блокчейн нашел себе применение не только в криптовалютах, но и в других сферах, таких как: деятельность банков, земельный реестр, удостоверение личности.

Многие связывают понятие Биткоина и блокчейна. В частности, это происхо-

дит потому что впервые об идее блокчейна заявил С. Накамото в своей статье [1], но конкретно понятие блокчейна он не вводил.

Блокчейн — выстроенная по определенным правилам непрерывная последовательная цепочка блоков, содержащих информацию. Применительно к разным сферам блокчейн может интерпретироваться по-разному. Чаще всего копии цепочек блоков хранятся на множестве разных компьютеров независимо друг от друга.

В различных системах генерировать и добавлять блоки могут как сами пользователи, так и сертифицированые органы. Так, например, в системе Биткоин блоки генерируются самими пользователями.

Рассмотрим основные принципы работы технологии блокчейн. Первоначальный блок обычно называют Initial Block. В системе пользователи осуществляют определенные действия, требующие подтверждения в сети. Далее, по заранее заданному правилу элементы объединяются в блоки. Затем пользователь, желающий добавить блок в цепочку осуществляет проверку на "корректность"совершенных действий. Например, достоверность осуществляемой транзакции. Данная проверка называется — "proof-of-stake". Помимо проверки на корректность зачастую пользователю необходимо совершить "доказательство работы"(proof-of-work), которая заключается в решении трудновычислимой задачи. Выполнив все проверки вновь полученный блок добавляется в общую цепочку блоков с указанием ссылки на предыдущий блок.

Если случается так, что два блока генерируются одновременно, а это возможно если в сети большое количество пользователей, то "верным"считается то ответвление, в котором содержится наибольшее число блоков. Пример цепочки приведен на Рис.1.

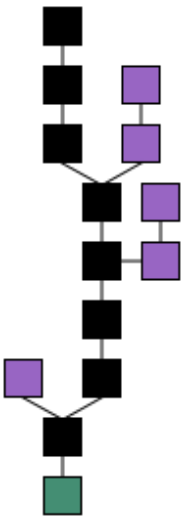


Рис. 1: Цепочка блоков. Начальный блок(нижний), за ним следуют подтвержденные блоки(черные), а дублированные блоки(когда несколько новых блоков в разных частях распределенной сети называют предыдущим один и тот же блок) отсеиваются.

1.1 Транзакции и объединение в блоки

1.1.1 Транзакции

Транзакция — это подтвержденная подписью секция данных, которая передается по сети Биткоин и собирается в блоки. Обычно она содержит ссылки на предыдущие транзакции и ассоциирует определенное количество Биткоинов с одним или несколькими публичными ключами (Биткоин адресами).

Данные транзакции Вход.

1)Предыдущая tx_1:
f5d8ee39a430901c91a5917b9f2dc19d6d1a0e9cea205b009ca73dd04470b9a6

Индекс: I_1

2)Предыдущая tx_2:
a592de9c1d17b9c99a4009ca74e31a0f6dea30901c105b2470b9a69f5d8e3dd0

Индекс: I_2

...

n)Предыдущая tx_n:
1d17b93c9c9da5ed74e0609a495b0a0370e3df249ead09a92ac161c100b89f5d

Индекс: I_n

n+1)Подпись:
304502206e21798a42fae0e854281abd38bacd1aeed3ee3738d9e1446618c4571
d10 90db022100e2ac980643b0b82c0e88ffdfec6b64e3e6ba35e7ba5fdd7d5d6c
c8d25c6b241501

Вход — это ссылки на выходы других транзакций. Значения этих ссылок суммируются, и общая сумма биткоинов может быть использован в выходе текущей транзакции. tx_i — это хеш предыдущей транзакции. Индекс — это определенный выход этой транзакции.

Выход.

1) Значение: 5000000000

2) Публичный ключ:
OP_DUP OP_HASH160 404371705fa9bd789a2fcd52d2c580b65d35549d
OP_EQUALVERIFY OP_CHECKSIG

Выход содержит инструкции по отправлению биткоинов. Значение — это количество сатоши (1 BTC = 100,000,000 сатоши), которое сможет использовать транзакция, для которой текущая будет входом. Может существовать более одного выхода, и они будут делить между собой сумму, пришедшую со входов.

Каждый выход транзакции может быть использован в качестве входа для следующей транзакции только один раз, поэтому сумма всех входов для текущей транзакции должна быть использована на еч выходах. В противном случае, оставшаяся сумма входов транзакции будет потеряна. Например, если ввод равен 50 BTC, а пользователю нужно отправить только 25 BTC, то биткоин создаст два выхода по 25 BTC каждый: один отправится в пункт назначения, а другой от-

правится еще раз владельцу этих средств (так называемая "сдача— транзакция, которую пользователь фактически отправляет сам себе).

Любая сумма входов биткоинов, не использованная в выходах, становится комиссией транзакции. Она достанется тому, кто сгенерирует блок.

1.1.2 Блоки

Блок — перманентно записываемые файлы в сети Биткоин, содержащие информацию о произошедших транзакциях. Блок — это запись части или всех недавних транзакций, которые еще не были записаны в предыдущие блоки. Когда блок добавлен в конец цепи, он не может быть изменен.

Все транзакции добавляются в список несовершенных транзакций. Пользователь, желающий добавить новый блок выбирает некоторое количество транзакций из этого списка(зависит от загруженности сети и размера самих транзакций) и формирует из них дерево Меркля, как показано на Рис.2

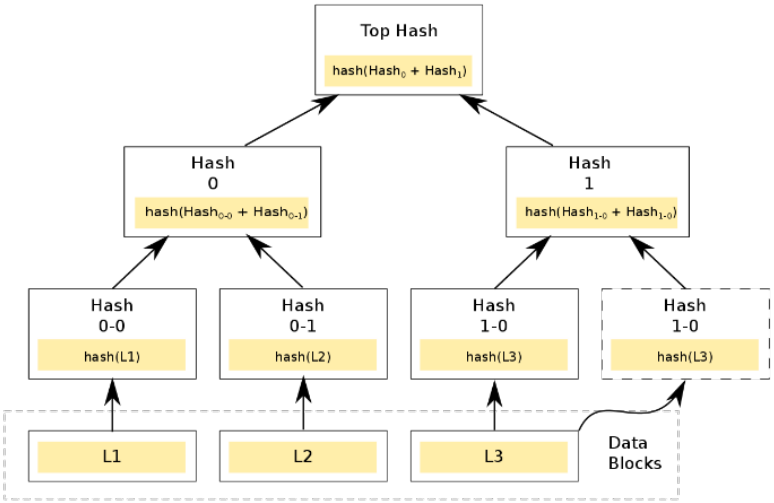


Рис. 2: Дерево Меркля

Построение дерева происходит следующим образом:

1. Вычисляются хэши транзакций, размещенных в блоке: hash(L1), hash(L2), hash(L3) и так далее.

2. Вычисляются хэши от результата конкатенации хэшей транзакций, например hash(hash(L1) | hash(L2)). Так как дерево Меркла является бинарным, то число элементов на каждой итерации должно быть четным. Поэтому если блок содержит нечетное количество транзакций, то последняя дублируется сама с собой: hash (hash(L3) | hash(L3)).

3. Далее, вновь вычисляются хэши от результата конкатенации хэшей. Процесс повторяется, пока не будет получен единый хеш — Top Hash, который является

корнем дерева Меркла (merkle root). Значение корня фиксируется в заголовке блока.

В системе Биткоин деревья Меркля строятся с использованием двойного хеширования $SHA-256$.

Заголовок(хэш) блока состоит из хэша предыдущего блока, хэша дерева(merkle root) и добавки(nonce), которая необходима для добавления блока в блокчейн.

Замечание. Такое хранение транзакций необходимо для экономии дискового пространства, так как заголовок пустого блока при таком хранении будет весить всего 80 байт. Стоит отметить, что на данный момент вес всей цепочки уже превысил 100Гб. Для рядового пользователя сети хранить всю цепочку не представляет никакой необходимости, достаточно хранить лишь небольшую часть и подгружать необходимую часть из сети.

1.1.3 Добавление блока в цепочку

В системе Биткоин реализован *сервер меток времени*(timestamp server). Его работа заключается в хэшировании блока данных, на который нужно поставить метку, и открытой публикации этого хэша(Рис.3). Каждый хэш включает в себя предыдущую метку: так выстраивается цепь, где очередное звено укрепляет все предыдущие.

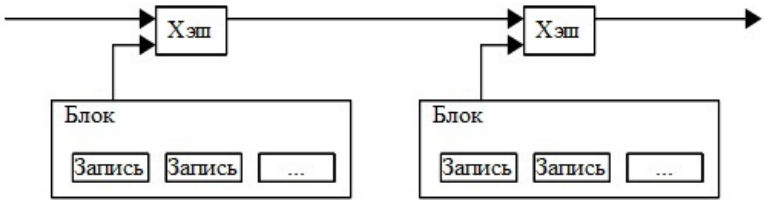


Рис. 3: Сервер меток времени

Пользователь, который хочет добавить транзакцию в цепочку(майнер) генерирует блок, затем проверяет этот блок на корректность(proof-of-stake), и осуществляет "доказательство работы"(proof-of-work). Суть доказательства проста: вычислять хэш от хэша заголовка блока и добавки(nonce).

В системе Биткоин используется хэш-функция $SHA-256$, таким образом мы получаем некоторое число, представленное в шестнадцатеричном формате. Автоматически устанавливается "цель— целое число, которое определяет необходимое число нулей для доказательства того, что блок может быть добавлен в цепочку блоков. Майнер перебирает добавку, пока хэш не будет содержать необходимое число нулей. И, наконец, добавляет блок в цепочку.

На данный момент, майнинг — крайне трудоемкий процесс, который практически невозможно реализовать на домашнем ПК. Для этих целей используют ви-

деокарты, ПЛИСы и специальные процессоры, заточенные на вычислении только $SHA-256$.

Может так случится, что два блока будут сгенерированы одновременно и тогда одна группа майнеров получит один блок, а другая группа второй(Рис.1). В таком случае, верным будет считаться тот блок, за которым первым последует следующий и все пользователи сети переключатся на это ответвление цепочки, "забывая"второстепенные.

1.2 Стимулы и эмиссия биткоинов

Для того, чтобы майнеры продолжали добавлять новые блоки в цепочку существует стимул.

Первая транзакция, содержащаяся в блоке, который сгенерировал майнер переводит на его адрес определенную сумму биткоинов. Изначально эта сумма равнялась 50 биткоинам и каждые 4 года она сокращалась вдвое. На данный момент награда составляет 12.5 биткоинов. Эта сумма будет уменьшаться, пока не достигнет значения 10^{-8}(минимальная единица, называемая Сатоши).

После чего эмиссия биткоинов прекратится, но стимул для майнеров не пропадет, так как помимо основной награды, майнеры также получают дополнительную награду(комиссию) с каждой транзакции.

2 Деанонимизация пользователей сети Биткоин

Проблемой анонимности в сети Биткоин занималось большое количество исследователей. В рамках курсовой работы я рассмотрю два варианта атаки на идентификацию пользователей сети: Байесовский метод и атаки при использовании помощью браузера Tor.

2.1 Байесовский подход

В статье[6] рассматривается вероятностная модель, основанная на распространении информации в сети Биткоин, которая позволяет нам идентифицировать пользователя, создавшего транзакцию.

Основная идея данного метода заключается в следующем:

1. Определяется вероятность для каждой транзакции быть созданной тем или иным клиентом (идентифицированному по IP адресу). Предполагая, что создатель транзакции контролирует кошелек(Биткоин адрес), с которого были посланы деньги, можно составить пары (Возможный IP адрес, Биткоин адрес).

2. Наиболее вероятные пары определяются путем комбинирования вероятностей из списка, составленного на предыдущем шаге.

3. Географическое местоположение IP адреса позволяет нам проводить анализ большого диапазона распределения и потока Биткоинов.

Для перехода к описанию метода необходимо ввести несколько вспомогательных понятий.

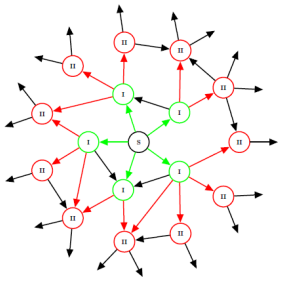


Рис. 4: Новая транзакция генерируется клиентом S. Клиенты, обозначенные как "I"извещаются о транзакции напрямую, через создателя. Затем, они распространяют транзакцию дальше.

Как было сказано ранее, по умолчанию, клиент устанавливает соединение с 8 другими пользователями. Когда пользователь хочет создать транзакцию, программа-клиент(далее — *создатель*) пересылает сообщение случайно выбранному клиенту каждые 100. Этот метод называется *триклинг*, его цель — скрыть источник транзакции. Помимо триклинга, они также пересылают сообщение другим клиентам с вероятностью 1/4(каждые 100).

Теперь перейдем к описанию метода.

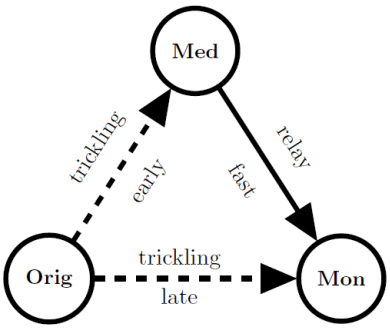
Распространяемые сообщения наблюдаются и записываются несколькими монитори клиентами с целью охватить большую часть сети, насколько это возможно. Для каждой транзакции мониторинг-клиенты записывают клиентов, переславших транзакцию в список "первостепенный сегмент"(*first time segment*). Они являются возможными создателями транзакции.

Затем, с помощью блокчейна мы сгруппируем Биткоин-адреса, принадлежащие одному владельцу. Группируя вероятности, полученные на первом шаге, пользователи(и их баланс) становятся в пару с клиентами, которые наиболее вероятно являются создателями транзакции.

Шаг 1: Собственные вероятности Рассмотрим отдельную транзакцию, наблюдаемую одним мониторинг-клиентом.

Мониторинг-клиент, подключенный к создателю не обязательно получает сообщение от создателя первым, потому что случается так, что оно может быть передано быстрее через медиатора.

Вычислим вероятность того, что создатель переслал сообщение мониторинг-клиенту через одну итерацию. Пусть к создателю подключено c_{orig} клиентов(среди которых и мониторинг-клиент).



$$P_1^{orig} = \frac{1}{c_{orig}} \tag{1}$$

В случае медиатора, он пересылает транзакцию мониторинг-клиенту с вероятностью $1/c_{med}$ из-за триклинга и с вероятностью $1/4$ при использовании иного механизма в одну итерацию. Получаем:

$$P_1^{med} = \frac{1}{c_{med}} + \frac{1}{4}(1 - \frac{1}{c_{med}}) = \frac{1}{4} - \frac{3}{4c_{med}} \tag{2}$$

Вероятность того, что клиент перешлет транзакцию на k-ом шаге будет равна

$$P_k^{orig} = P_1^{orig}(1 - P_1^{orig})^{k-1} \tag{3}$$

$$P_k^{med} = P_1^{med}(1 - P_1^{med})^{k-1} \tag{4}$$

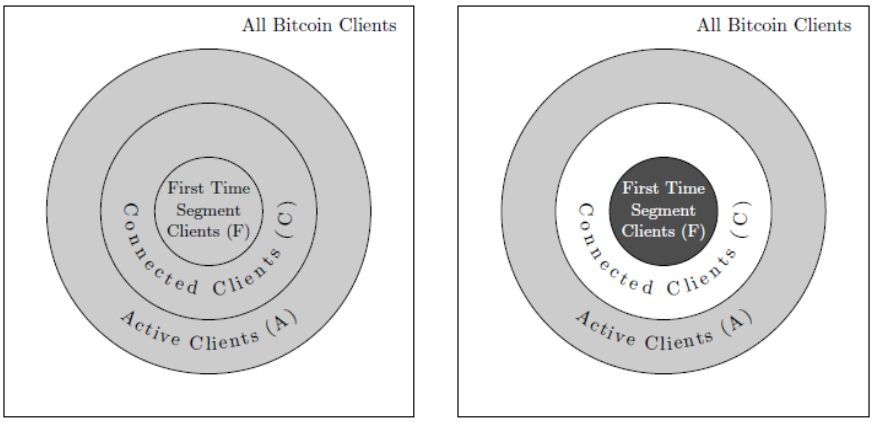
Теперь рассмотрим ситуацию, когда создатель отправляет сообщение медиатору, а медиатор пересылает мониторинг-клиенту.

$$P_k^{orig+med} = \sum_{i=1}^{k-1}(P_i^{orig} + P_{k-i}^{med}) = \frac{P_1^{orig} P_1^{med}}{P_1^{med} - P_1^{orig}}[(1 - P_1^{orig})^{k-1} - (1 - P_1^{med})^{k-1}] \tag{5}$$

Если считать пути независимыми, то вероятность того, что прямой маршрут окажется "короче"равняется 0.6334. Наша цель — определить временные рамки, необходимые мониторинг-клиенту, чтобы получить сообщение прямиком от создателя. Назовем этот интервал *первостепенным* и определим его равным $t_1 = 2$ сек, в таком случае вероятность будет равна 0.8071. Высока вероятность, что подключенные клиенты, которые не принадлежат первостепенному интервалу, не являются создателями транзакции. В условиях этого предположения оценим вероятность клиента быть создателем транзакции, основываясь на каждой полученной транзакции.

Смотря с позиции мониторинг клиента — другие Биткоин адреса могут быть разделены на группы (см. Рис.8).

До транзакции, мы не знаем никакой информации, поэтому лучшей оценкой, которую мы можем дать — признать равную вероятность для Биткоин клиента



(a) Before Transaction (b) After Transaction

Рис. 5: Отношение разных групп клиентов

быть создателем транзакции (левая сторона Рис.8). После транзакции, каждый Биткоин клиент в первостепенном интервале может являться как создателем транзакции, так и клиентом, ее переславшим. Более того, настоящий создатель может быть также среди остальной части сети, не подключенной к нашему мониторинг-клиенту. С другой стороны, основываясь на предыдущих утверждениях, мы предполагаем, клиенты не пересылающие транзакции в первостепенный интервал точно не являются создателями транзакции. Соответственно, вероятность клиентов первостепенного интервала растет, в то время как подключенные клиенты не из первостепенного интервала имеют нулевую вероятность(правая часть Рис.8).

Теперь вычислим вероятность быть создателем для каждой из групп. Обозначим за $\mathfrak{C}$ тот факт, что мониторинг клиент подключен к создателю транзакции, $\mathfrak{O}$ — создатель переслал сообщение из первостепенного интервала мониторинг клиенту и $\mathfrak{F}$ означает, что случайно выбранный клиент из первостепенного интервала является создателем транзакции. Используя введенные обозначения имеем:

$$\mathbb{P}(\mathfrak{C}) = \frac{|\mathfrak{C}|}{|\mathfrak{A}|} \tag{6}$$

Если мониторинг клиент подключен к создателю, он известит мониторинг клиент в первостепенном интервале. В то же время, все клиенты первостепенного интервала имеют ту же вероятность быть создателем

$\mathbb{P}(\mathfrak{O}|\mathfrak{C}) = 1$ $\mathbb{P}(\mathfrak{F}|\mathfrak{C}) = \frac{1}{|F|}$

Теперь выпишем вероятность $\mathbb{P}(\mathfrak{F})$, используя формулу полной вероятности:

$$\mathbb{P}(\mathfrak{F}) = \mathbb{P}(\mathfrak{F}|\mathfrak{C}) \cdot \mathbb{P}(\mathfrak{C}) + \mathbb{P}(\mathfrak{F}|\overline{\mathfrak{C}}) \cdot \mathbb{P}(\overline{\mathfrak{C}}) = \frac{|C|}{|A||F|} \tag{7}$$

Где мы используем тот факт, что клиент не может отправлять сообщения в первостепенном интервале, если он вообще не подключен к мониторинг клиенту.

Приведенная выше формула дает вероятность, отнесенную к клиентом первостепенного интервала. Подключенные клиенты не из первостепенного интервала

имеют нулевую вероятность. Остальная часть активных клиентов имеют ту же вероятность $1/|A|$.

До сих пор мы рассматривали только случай использования одного мониторинг клиента. В случае использования большего количества мониторинг клиентов все указанные группы определяются независимо друг от друга, то есть $F(tx_i) = \bigcup_{j=1}^{N} F_j(tx_j)$ и $C(tx_i) = \bigcup_{j=1}^{N} C_j(tx_i)$ для N мониторинг клиентов, где нижние индексы обозначают соответствующие наборы, наблюдаемые для транзакции tx_i j-ым мониторинг клиентом. Используя этот метод, мониторинг клиент Биткоин не должен быть синхронизирован по времени.

Шаг 2: Группировка транзакций одного пользователя Следующим нашим шагом является группировка транзакций по их создателям.

Для этого мы используем тот факт, что Биткоин адреса, появляющиеся на входах одинаковых транзакий обычно принадлежат одному и тому же пользователю. Более подробно о группировке адресов можно посмотреть в статьях [12], [13], [14].

Шаг 3: Байесовская классификация Будем предполагать, что транзакции, относящиеся к одному пользователю были созданы несколькими создателями (клиентами).

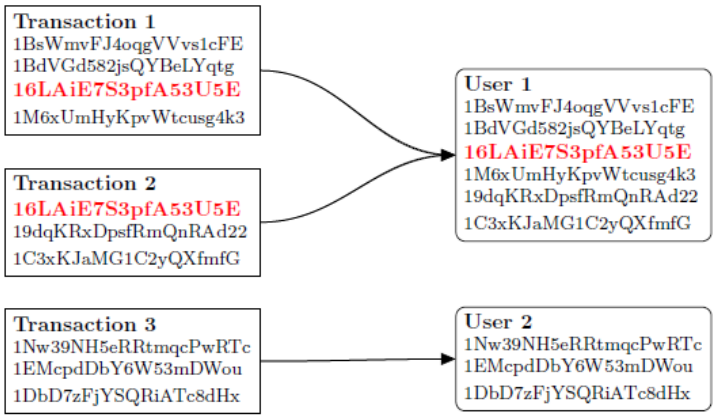


Рис. 6: Группировка адресов Биткоин: слева — три транзакции и входящие адреса этих транзакций, справа — то, как адреса группируются.

В таблице ниже представлены вероятность создания транзакции(tx_1) по отношению к IP адресу создателя.

Если доля подключенных клиентов невелика собственные вероятности в таблице также малы. Вероятность того, что IP адрес относится к разным транзакциям может быть вычислена с помощью Байесовской классификации.

IP адреса разобьем на два класса: "создатели"и "не создатели". Для каждой транзакции имеется хотя бы один IP адрес в классе создателей. Хотя, в итоге, в этом классе может лежать более одного адреса.

	IP_1	$\cdots$	IP_i	$\cdots$	IP_n
tx_1	$\mathbb{P}(\mathrm{IP}_1\|\mathrm{tx}_1)$		$\mathbb{P}(\mathrm{IP}_i\|\mathrm{tx}_1)$		$\mathbb{P}(\mathrm{IP}_n\|\mathrm{tx}_1)$
tx_2	$\mathbb{P}(\mathrm{IP}_1\|\mathrm{tx}_2)$		$\mathbb{P}(\mathrm{IP}_i\|\mathrm{tx}_2)$		$\mathbb{P}(\mathrm{IP}_n\|\mathrm{tx}_2)$
$\cdots$					
tx_j	$\mathbb{P}(\mathrm{IP}_1\|\mathrm{tx}_j)$		$\mathbb{P}(\mathrm{IP}_i\|\mathrm{tx}_j)$		$\mathbb{P}(\mathrm{IP}_n\|\mathrm{tx}_j)$
$\cdots$					
tx_m	$\mathbb{P}(\mathrm{IP}_1\|\mathrm{tx}_m)$		$\mathbb{P}(\mathrm{IP}_i\|\mathrm{tx}_m)$		$\mathbb{P}(\mathrm{IP}_n\|\mathrm{tx}_m)$

Используя Байесовскую классификацию выпишем вероятность того, что конкретный IP адрес (IP_i) является создателем транзакции(см. [6] приложение 7):

$$\mathbb{P}(IP_i \in C_0|\mathbf{tx}) = \frac{1}{1 + exp[(1-m)ln(|\bar{A}| - 1) + \sum_{k=1}^{m} ln(\frac{1}{\mathbb{P}(IP_i \in C_0|tx_k)} - 1)]} \quad (8)$$

Где $\mathbf{tx}$ — вектор из всех рассматриваемых транзакций, $|\bar{A}|$ — среднее количество активных клиентов, m — число транзакций.

Результаты Авторы в [6] приводят следующие результаты.

В период с 14.10.2013 по 20.12.2013 было сделано около 300 миллионов записей, в которых были идентифицированы 4155387 транзакций и 124498 IP адресов. Собранная информация была помещены в базу данных SQL ([20]).

2.2 Биткоин и TOR

Tor(The Onion Router) — это система прокси-серверов, позволяющая устанавливать анонимное сетевое соединение, защищенное от прослушивания. Для описания атаки необходимо сначала ввести некоторые понятия, существующие в Биткоине и Tor'е. В системе Биткоин не существует понятие сервер-клиент как такового, однако, мы все же будем различать эти понятия. Серверами мы будем называть узлы, которые принимают входящие подключения, а клиентами — узлы, которые не принимают. На момент написания статьи [2] существовало около 7000 серверов и 100000 клиентов. По умолчанию, каждый узел(пользователь) устанавливает соединение с 8 внешними узлами в сети. Если любое из 8 подключений прервется, система будет искать новые 8 подключений. Используя нашу терминологию, будем клиенты могут осуществлять подключения только к серверам.

Будем называть сервера, к которым подключился клиент "узлами ввода". Сервер может поддерживать до 117 подключений одновременно. Введем еще 4 базовых свойства, которыми обладает Биткоин:

1. В качестве анти-DoS защиты в системе существуют следующие правила. Каждый узел хранит, так называемый штрафной счет по отношению к каждому узлы всей сети(определяемому по IP адресу). Когда "некорректное"сообщение(весом 60 байт) отправляется пользователю — он увеличивает штрафной счет на 1.

Как только показатель достигнет 100 пользователь банит этот IP адрес на 24 часа.

2. Система Биткоин распознает только три типа адресов: $IPv4$, $IPv6$ и $OnionCat$.

3. Каждый узел Биткоин хранит базу данных IP адресов всех узлов, когда-либо появлявшихся в сети. При перезапуске клиента она сохраняется. Это реализуется путем обновления базы данных, хранящиеся на жестком диске, каждые 15 минут (это позволит нам настроить cookie на компьютере пользователя). Пользователи могут запрашивать адреса друг у друга путем отправления $GETADDR$ сообщений и самостоятельно публиковать адреса, используя $ADDR$ сообщения. В случае использования TORa, каждую секунду мы имеем подключения через DNS-сервера. Подключаясь к DNS, клиент запрашивает множество адресов и отключается.

   Каждый адрес сопровождается меткой времени, показывающий его свежесть.

4. Все адреса в базе данных распределяются по корзинам. Существует 256 корзин для "новых"адресов(к которым клиент еще ни разу не подключался) и 64 корзины для "опробованных"адресов (имело место быть хотя бы одно подключение). Каждая корзина способна хранить до 20480 адресов. Когда пользователь устанавливает соединение, он выбирает адрес из "опробованной"корзины с вероятностью $p = 0.9 - 0.1 \cdot n$, где n — это число уже установленных соединений(необходимо 8). Если адрес достаточно часто публикуется(см п.3), то у него есть хорошие шансы быть выбранным в качестве узла ввода и быть перемещенным в корзину "опробованных".

Теперь более подробно остановимся на устройстве Tor'a. Когда пользователь хочет подключится к сети Интернет, оставляя свой IP адрес в секрете он выбирает путь(строит цепь), состоящий из трех ретрансляторов, называемых "охранный"(Guard), "средний"(Middle) и "выходной"(Exit). В виду интуитивного понятия, будем придерживаться английского написания этих терминов. В целях ускорения работы, системой заранее установлены три Guard узла, на которые и совершается первый "прыжок"в цепи.

Список всех ретрансляторов представлен в документе, называемым "консенсус". Пользователь использует тот ретранслятор, которой соответствует необходимой пропускной способности, указанной в "консенсусе". Каждый ретранслятор определяется своим ID(fingerprinting).

Для подключения к Web-ресурсу анонимно через цепочку Tor'a, Exit ретранслятор (последний в цепи) должен осуществлять подключение к сети Интернет. В связи с этим, имеется список IP адресов и портов, к которым Exit узел может подключиться, а которым подключение запрещено. Двигаясь по цепи, клиент выбирает те Exit ретрансляторы, которые позволят ему достичь пункта назначения.

2.2.1 Человек по середине

Используя анти-DoS защиту Биткоина мы можем навязать пользователю, который подключается к сети Биткоин через Tor, подключиться к "нашему"Exit узлу или к нашим узлам Биткоин, изолируя пользователя от всей остальной сети. Рассмотрим подробнее как мы можем это сделать.

Атака состоит из четырех основных шагов:

1. Вод некоторого числа Биткоин узлов в сеть. Отметим, что Биткоин позволяет иметь только один узел на IP адрес.

2. Периодически публиковать новые введенные узлы в сеть так, чтобы они были включены в максимальное количество корзин со стороны клиента.

3. Ввести несколько Tor Exit ретрансляторов.

4. Заставить неатакующие узлы Биткоина заблокировать Tor Exit вершины неатакующего.

Теперь рассмотрим каждый шаг немного подробнее.

A. Ввод узлов Биткоин. Атакующий должен приобрести большое количество IP адресов, чтобы следовать принципу "один узел-один адрес". Самый легкий способ — аренда IP адресов. Средняя стоимость аренды составляет один цент в час. Важно заметить, что IP адрес не должен быть вовлечен ни в какую другую постороннюю активность(например, отправка спама или участие в DoS атаках).

B. Публикация узлов атакующего. Естественно, что атакующий заинтересован в том, чтобы его узлы выбирались клиентом как можно чаще. Соответственно, атакующий должен публиковать адреса своих узлов как можно чаще.

C. Ввод Tor Exit узлов. Exit узлы атакующего должны осуществлять подключения к любым двум портам из 80, 443, 6667. На практике же, атакующий может изменять политику подключения и осуществлять подключение через порт 8333.

D. Бан Tor Exit узлов. В этой части мы будем использовать анти-DoS защиту Биткоина. Атакующий выбирает неатакующий Биткоин узел и Tor Exit неатакующего, строит цепь через этот Exit узел и отправляет "некорректное"сообщение выбранному Биткоин узлу. Как только Биткоин узел получает такое сообщение, он анализирует IP адрес, который, очевидно, принадлежит Tor Exit, который выбрал атакующий.

E. Поражение onion узлов. Текущая конструкция скрытой службы Tor (позволяет клиентам подключаться не раскрывая физического расположения) позволяет проводить DoS-атаку. Перед подключением к скрытой службе клиент должен скачать соответствующий дескриптор из одной из шести доступных директорий, выбранных среди всех ретрансляторов Tor'a (однозначно определяются днем и onion адресом). Соответственно, атакующий должен добавить шесть ретрансляторов, которые станут доступными директориями. Авторы в [9] показывают как можно провести такую атаку. Атакующий может с легкостью ввести большое количество Биткоин узлов, доступных как скрытые службы Tor'a. Требуется запуск лишь одного BITCOIND экземпляра и привязка его к как можно большему количеству onion адресов. И пользователи с большей вероятностью будут подключаться к подконтрольным "onion"узлам.

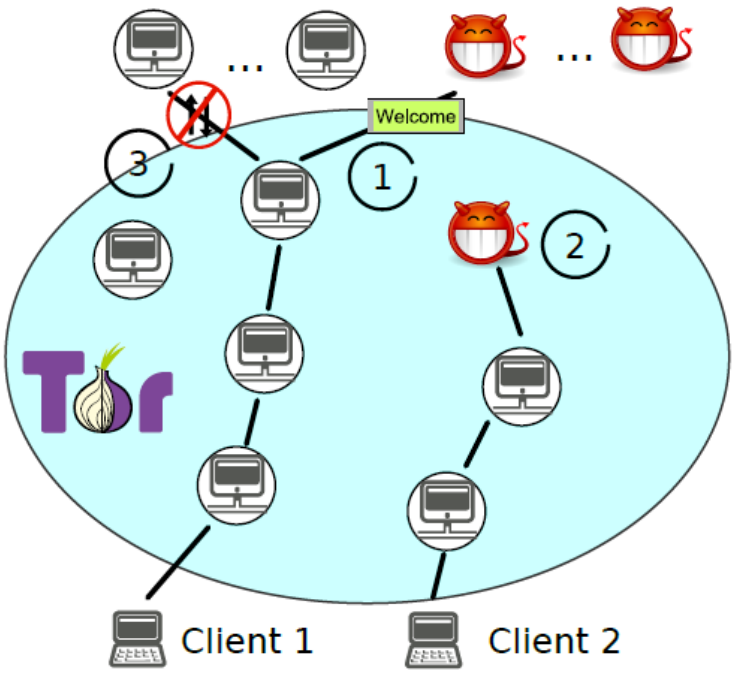


Рис. 7: Биткоин и Tor

2.2.2 Подсчет стоимости атак

Подсчет проводился в феврале 2018 года, опираясь на варианты атак, предложенные в статье [2].

Аренда подходящего ретранслятора Tor (не Exit) стоит около 2000 рублей в месяц. Также, необходимо доплачивать порядка 12500 рублей в месяц, чтобы обеспечить трафик в 180 терабайт в месяц (в то время, как на бесплатной основе предоставляется 10). На аренду, содержание и обеспечение необходимого трафика для шести таких ретрансляторов потребуется около 140000 рублей в месяц ($2500USD$).

3 ZCash

Перед тем как перейти к рассмотрению криптовалюты ZCash необходимо определить, что такое доказательства с нулевым разглашением.

Обычно, Алиса может доказать что-нибудь Бобу только рассказав ему все. Значит, Боб может выложить полученные сведения кому угодно, а Алиса ничего не сможет поделать.

Доказательство с нулевым разглашением позволяет нам добиться того, что Алиса докажет Бобу о том, что она владеет какой-либо информацией, не рассказывая саму информацию.

Существует много как интерактивных, так и неинтерактивных протоколов с нулевым разглашением, большую часть из них можно найти в [11].

ZCash — криптовалюта, основанная на криптографическом протоколе доказательства с нулевым разглашением. В логике работы протокола предполагается, что одна из взаимодействующих сторон способна убедиться в достоверности математического утверждения, не имея при этом никакой другой информации от второй стороны. Фактически ZCash является первой по-настоящему анонимной криптовалютой, так как можно подтвердить, что транзакция была совершена, но нет возможности узнать сумму, отправителя и получателя.

Протокол ZCash не сохраняет информацию о том, с какого адреса была проведена транзакция, на какой адрес было зачисление и какова сумма; виден только факт совершения транзакции. Информация о транзакции в блокчейне может быть открыта или скрыта по желанию пользователей, что является отличительной чертой ZCash.

ZCash шифрует содержимое защищенных транзакций. Для проверки достоверности использует криптографический протокол zk-SNARK (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge), разработанный создателями криптовалюты на основе доказательства с нулевым разглашением. Протокол позволяет формировать безопасный реестр остатков без раскрытия соответствующих сторон или сумм, но при этом можно доказать, что транзакция не является обманом или воровством. ZCash также позволяет пользователям создавать публичные транзакции, аналогично транзакциям в Биткоин.

Для того, чтобы обладать нулевым разглашением в ZCash, функция, определяющая истинность транзакции в соответствии с правилами сети, должна возвращать ответ о том, истинна ли транзакция или нет, без раскрытия какой-либо информации о выполненных действиях. На высоком уровне zk-SNARK превращает то, что вы хотите доказать, в эквивалентную форму знания решения некоторых алгебраических уравнений. В следующем разделе мы более подробно разберем работу протокола.

3.1 zero-knowledge Succinct Non-interactive ARgument of Knowle (zk-SNARK)

zk-SNARK — криптографический протокол с нулевым разглашением, использующийся в криптовалюте ZCash.

Zk-SNARK состоит из трех алгоритмов G, P, V определенных следующим образом:

Генератор ключей G принимает секретный параметр λ и программу C, и генерирует два публичных ключа: доказательный ключ pk, и ключ проверки vk. Эти ключи необходимо создать только один раз для данной программы C.

Алгоритм генерации доказательства P принимает в качестве входных значений: ключ pk, публичное значение x и секретное значение a. Алгоритм генерирует доказательство $\pi = P(pk, x, a)$ того, что доказывающий знает секретное значение a и что секретное значение удовлетворяет программе C.

Проверяющий алгоритм V вычисляет $V(vk, x, \pi)$ результатом которого будет $true$, если доказательство является верным, и $false$ в противном случае. Таким образом, эта функция возвращает $true$, если проверяющий знает, что секретное значение w удовлетворяет $C(x, w) == true$.

Обратите внимание на секретный параметр λ, используемый в генераторе. Этот параметр иногда затрудняет использование zk-SNARKs в реальных приложениях. Причина этого в том, что любой, кто знает этот параметр, может генерировать поддельные доказательства. В частности, при любой программе C и публичном значении x, человек, который знает λ, но не знает секретного a, может создать доказательство $\pi^{'}$, на которое алгоритм $V(vk, x, \pi^{'})$ вернет $true$.

Таким образом, запуск генератора должен являться безопасным процессом, защищенным от того, чтобы кто-то могу узнать или украсть параметр λ. Это стало причиной чрезвычайно сложной церемонии, проведенной командой ZCash для создания доказательного ключа и ключа проверки, при этом все "токсичные отходы"λ были уничтожены.

Давайте более детально разберем работу каждого алгоритма.

Алгоритм 1. ***Генератор ключей*** G**.**

$\triangleright$ Вход: *схема* $C : F_r^n \times F_r^h \to F_r^h$

$\triangleright$ Выход: *доказательный ключ* pk*, и ключ проверки* vk

Шаг 1: *Вычисление* $(\overrightarrow{A}, \overrightarrow{B}, \overrightarrow{C}, Z) := QAPinst(C)$*, где* $A_{m+1} = B_{m+2} = C_{m+3} = Z$*,* $A_{m+2} = A_{m+3} = B_{m+1} = B_{m+3} = C_{m+2} = 0$

Шаг 2: *Случайно выбрать* $\tau, \rho_A, \rho_B, \alpha_A, \alpha_B, \alpha_C, \beta, \gamma \in \mathbb{F}_r$

Шаг 3: *Определим* $pk := (C, pk_A, pk_A^{'}, pk_B, pk_B^{'}, pk_C, pk_C^{'}, pk_K, pk_H)$*, где для* $i = 0, 1, ..., m + 3$*:*

$pk_{A,i} := A_i(\tau)\rho_A P_1$, $pk'_{A,i} := A_i(\tau)\alpha_A\rho_A P_1$,

$pk_{B,i} := B_i(\tau)\rho_B P_2$, $pk'_{B,i} := B_i(\tau)\alpha_B\rho_B P_1$,

$pk_{C,i} := C_i(\tau)\rho_A\rho_B P_1$, $pk'_{C,i} := C_i(\tau)\alpha_C\rho_A\rho_B P_1$,

$pk_{K,i} := \beta(A_i(\tau)\rho_A + B_i(\tau)\rho_B + C_i(\tau)\rho_A\rho_B)P_1$,

и для $i = 0, 1, ..., d$, $pk_{H,i} := \tau^i P_1$.

Шаг 4: *Определим* $vk := (vk_A, vk_B, vk_C vk_\gamma, vk^1_{\beta\gamma}, vk^2_{\beta\gamma}, vk_Z, vk_{IC})$, *где*

$vk_A := \alpha_A P_2$, $vk_B := \alpha_B P_1$,$vk_C := \alpha_C P_2$,

$vk_\gamma := \gamma P_2$, $vk^1_{\beta\gamma} := \gamma\beta P_1$, $vk^2_{\beta\gamma} := \gamma\beta P_2$,

$vk_Z := Z(\tau)\rho_A\rho_B P_2$, $(vk_{IC})^n_{i=0} := (A_i(\tau)\rho_A P_1)^n_{i=0}$.

Шаг 5: *Выход* (pk, vk)

Алгоритм 2. ***Алгоритм генерации доказательства*** P.

▷ Вход: *доказательный ключ* pk, *вход* $x \in F^n_r$, *и свидетельство* $a \in F^h_r$

▷ Выход: *доказательство* π

Шаг 1: *Вычислить* $(\overrightarrow{A}, \overrightarrow{B}, \overrightarrow{C}, Z) := QAPinst(C)$

Шаг 2: *Вычислить* $\overrightarrow{s} := QAPwit(C, \overrightarrow{x}, \overrightarrow{a}) \in F^m_r$

Шаг 3: *Случайно выбрать* $\delta_1\delta_2\delta_3 \in F_r$

Шаг 4: *Вычислить* $\overrightarrow{h} = (h_0, ..., h_d) \in F^{d+1}_r$, *которые являются коэффициентами* $H(z) := \frac{A(z)B(z)-C(z)}{Z(z)}$, *где* $A, B, C \in F_r[z]$ *следующие:*

$A(z) := A_0(z) + \sum^m_{i=1} s_i A_i(z) + \delta_1 Z(z)$,

$B(z) := B_0(z) + \sum^m_{i=1} s_i B_i(z) + \delta_2 Z(z)$,

$C(z) := C_0(z) + \sum^m_{i=1} s_i C_i(z) + \delta_3 Z(z)$.

Шаг 5:

Положить $\tilde{pk}_A :=$ *"равным* pk_a, *но с* $pk_{A,i} = 0$ *для* $i = 0, ...n$ *".*

Положить $\tilde{pk}_A{}' :=$ *"равным* pk'_a, *но с* $pk'_{A,i} = 0$ *для* $i = 0, ...n$ *".*

Шаг 6: *Пусть* $\vec{c} := (1 \circ \vec{s} \circ \delta_1 \circ \delta_1 \circ \delta_1) \in \mathbb{F}^{4+m}_r$ *и вычислить:*

$\pi_A := \langle \vec{c}, \tilde{pk}_A \rangle$, $\pi'_A := \langle \vec{c}, \tilde{pk}_A{}' \rangle$, $\pi_B := \langle \vec{c}, pk_B \rangle$, $\pi'_B := \langle \vec{c}, pk'_B \rangle$,

$\pi_C := \langle \vec{c}, pk_c \rangle$, $\pi'_C := \langle \vec{c}, pl'_C \rangle$, $\pi_K := \langle \vec{c}, pk_K \rangle$, $\pi_H := \langle \vec{c}, pk_H \rangle$.

Шаг 7: *Выход* $\pi := (\pi_A, \pi'_A, \pi_B, \pi'_B, \pi_C, \pi'_C, \pi_K, \pi_H)$

Алгоритм 3. ***Проверяющий алгоритм*** V.

▷ Вход: *ключ проверки* vk, *вход* $x \in F^n_r$ *и доказательство* π

▷ Выход: *Бит выбора(true/false)*

Шаг 1:

Вычислить $vk_{\vec{x}} := vk_{IC,0} + \sum^n_{i=1} x_i vk_{IC,i} \in \mathbb{G}_1$

Шаг 2: *Проверка правильности знания для* A, B, C*:*

$e(\pi_A, vk_A) = e(\pi'_A, P_2)$, $e(vk_B, \pi_B) = e(\pi'_B, P_2)$, $e(\pi_C, vk_C) = e(\pi'_C, P_2)$.

Шаг 3: *Проверка использования тех же коэффициентов:*

$e(\pi_K, vk_\gamma) = e(vk_{\vec{x}} + \pi_A + \pi_C, vk^2_{\beta\gamma}) \cdot e(vk^1_{\beta\gamma}, \pi_B)$.

Шаг 4: *Проверка делимости QAP:*

$e(vk_{\vec{x}} + \pi_A, \pi_B) = e(\pi_H, vk_Z) \cdot e(pi_C, P_2)$.

Шаг 5: $True$, *если все проверки верны, иначе* $False$.

3.2 Пояснения

В этой секции будут даны теоретические пояснения к изложенным выше алгоритмам.

Определение 4. (Арифметическая схема)

Арифметическая схема представляет собой ациклический граф, который состоит из вычисляемых арифметических операций, называемых переходами, таких как сложение и умножение, с соединениями между ними. Например, рассмотрим многочлен $(c_1 \cdot c_2) \cdot (c_1 + c_3)$ его представление в виде арифметической схемы будет выглядеть следующим образом:

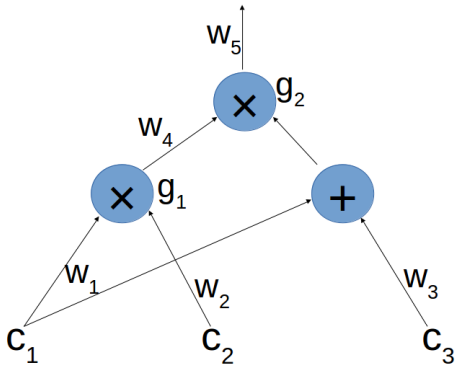


Рис. 8: Арифметическая схема

Допустимым набором присваивания для схемы, является присваивание значений для помеченных переходов, где выходное значение каждого перехода умножения является результатом произведением соответствующих входов.

Для нашей схемы допустимый набор присваивания будет иметь вид: $(c_1, ..., c_5)$, где $c_4 = c_1 \cdot c_2$ и $c_5 = c_4(c_1 + c_3)$

Определение 5. (Квадратичная арифметическая программа)

Квадратичной арифметической программой размера m и степени d называется набор $(\vec{A}, \vec{B}, \vec{C}, Z)$, где $\vec{A}, \vec{B}, \vec{C}$ — три вектора длины $m + 1$, состоящие из многочленов степени $\leq d$ над полем $\mathbb{F}[z]$ и $Z \in \mathbb{F}[z]$ степени d.

Спаривание Пусть $\mathbb{G}_1$ и $\mathbb{G}_2$ две циклические группы порядка r с образующими элементами P_1 и P_2 соответственно.

Спариванием называется отображение $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, где $\mathbb{G}_T$ также является циклической группой порядка r, которое удовлетворяет следующим условиям:

1. Билинейность. $\forall \alpha, \beta \in \mathbb{F}_r \backslash 0 \; e(\alpha P_1, \beta P_2) = e(P_1, P_2)^{\alpha\beta}$
2. Невырожденность.$e(P_1, P_2) \neq 1 \in \mathbb{G}_T$

В протоколе zk-SNARKs используются эллиптические кривые Барето-Наерига, определенные над полем $\mathbb{F}_q$, со степенью вложение k, порядка r $(E(\mathbb{F}_q))$. $\mathbb{G}_T$ является подгруппой корней r-ой степени из единицы в $\mathbb{F}_q^*$. $\mathbb{G}_1$ определяется как подгруппа порядка r группы $E(\mathbb{F}_q)$. $\mathbb{G}_2$ является подгруппой порядка r группы $E'(\mathbb{F}_{q^{k/d}})$, где $E' - d -$ кручение E.

Определение 6. (Кривая в форме Баррето-Наерига) Пусть $\mathbb{F}_q$ — конечное поле. Кривая в форме Баррето-Наерига над полем $\mathbb{F}_q$ задается уравнением $y^2 = x^3 + b$, где $b \in \mathbb{F}_q \backslash 0$

В протоколе zk-SNARKs используются следующие параметры (см.[17]):

$q = 18882428718392752222464057452572750886963111572978236626890378946452262085 83$

$r = 21888242871839275222246405745257275088548364400416034343698204186575808495617$

$b = 3$

$\mathbb{G}_1$ — группа точек эллиптической кривой Баррето-Наерига $E_{\mathbb{G}_1}$ над $\mathbb{F}_q$, задаваемое уравнением $y^2 = x^3 + b$. Кривая имеет степень расширения 12 порядка r.

$\mathbb{G}_2$ — подгруппа порядка r шестого кручения $E_{\mathbb{G}_2}$ группы $\mathbb{G}_1$ над $\mathbb{F}_{q^2}$, задаваемое уравнением $y^2 = x^3 + b/\xi$, где $\xi \in \mathbb{F}_{q^2}$.

В [15], сказано, что использование данной кривой с заданными параметрами гарантирует 128-битную стойкость. В работе будет показано, как можно понизить стойкость до 110 бит.

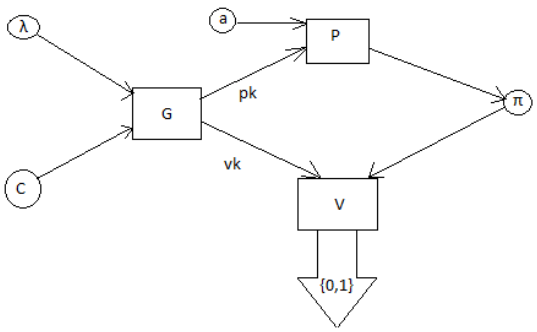
В качестве отображения e используется двухступенчатая функция: $e(P, Q) := FE(ML(P, Q))$, где $ML : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{F}_q^k$(Петля Миллера, Miller loop) и $FE : \mathbb{F}_q^k \to \mathbb{F}_q^k$ $(\alpha \mapsto \alpha^{(q^k-1)/r})$.

Параметр k (в нашем случае он равен 12) также называют MOV-степень или множитель секретности.

3.3 Анализ zk-SNARKs

3.3.1 Проблема дискретного логарифмирования(ПДЛ) на группе точек эллиптической кривой

Естественным встает вопрос о стойкости данной системы. Чтобы подделать сообщение нам необходимо найти секретный параметр $\lambda = (\tau, \rho_A, \rho_B, \alpha_A, \alpha_B, \alpha_C, \beta, \gamma)$. Наглядной демонстрацией является следующая диаграмма.



Зная λ мы можем сгенерировать с помощью pk доказательство π и используя vk и π успешно пройти проверяющий алгоритм V.

Найти секретные параметры $\tau, \rho_A, \rho_B, \alpha_A, \alpha_B, \alpha_C, \beta, \gamma$ можно воспользовавшись пунктом 4 Алгоритма 1. Например, рассмотрим равенство $vk_A := \alpha_A P_2$, где vk_A и P_2 известны. Требуется найти лишь $\alpha_A \in \mathbb{F}_r$. Таким образом, возникает задача дискретного логарифмирования на группе точек эллиптической кривой.

Напомним, стандартную оценку сложности для решения ПДЛ:
$L_Q(l, c) = exp((c + o(1))(\log Q)^l(\log\log Q)^{1-l})$

3.3.2 Оценка сложности решения ПДЛ для кривых Баррето-Наерига

Исследователи в своих работах ставят задачу максимально понизить параметры l и c для различных Q. Существует множество современных алгоритмов дискретного логарифмирования. Нас интересует случай, когда $Q = p^k$, где p — простое.

Авторы в [21], используя модификацию алгоритма ex-TNFS (Tower Number Field Sieve(см. [22])), предложили наименьшие из известных параметры l и c, равные 1/3 и 1.526 соответственно.

Также, в [21] было показано, что кривые Баррето-Наерига при использовании исходных параметров q, r и k гарантируют 110 бит безопасности, в отличии от заявленных 128 бит.

3.3.3 Анализ параметров zk-SNARKs

В своей работе я проведу сравнительный анализ параметров криптопротокола zk-SNARKs. Ниже будут приведена таблица зависимости параметров q и k от сложности решения ПДЛ. Все вычисления производились на персональном компьютере помощью программы Wolfram Mathematica 11.1.

t	k	n
254	11	103
384	11	133
512	11	157
254	12	110
384	12	140
512	12	169
254	13	113
384	13	147
512	13	170
254	14	120
384	14	150
512	14	177
254	15	123
384	15	157
512	15	183
254	16	130
384	16	163
512	16	190
254	17	133
384	17	170
512	17	196
254	18	140
384	18	173
512	18	203
254	19	143
384	19	180
512	19	210
254	20	146
384	20	183
512	20	220

Где $q \approx 2^t$, $L_{q^k}(l, c) \approx 2^n$

Список использованных источников и литературы

[1] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. - 2008

[2] Ivan Pustogarov, Alex Biryukov/ Bitcoin over Tor isnt a good idea. - 2015

[3] Bitcoin Wiki, https://en.bitcoin.it/wiki/ - 2014

[4] Venkatakrishnan SB, Fanti G, Viswanath P. "Dandelion: Redesigning the Bitcoin Network for Anonymity. 2017

[5] Biryukov A, Khovratovich D, Pustogarov I. "Deanonymisation of clients in Bitcoin P2P network."Proceedings of the - 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM - 2014

[6] Peter L. Juhasz, Jozsef Steger, Daniel Kondor, Gabor Vattay A Bayesian Approach to Identify Bitcoin Users. - 2014

[7] Артем Генкин, Алексей Михеев. Блокчейн. Как это работает и что ждет нас завтра. М.: Альпина Паблишер, 2017

[8] Michael Nielsen on December. How the Bitcoin protocol actually works,2017

[9] A. Biryukov, I. Pustogarov, and R.-P. Weinmann, "Trawling for tor hidden services: Detection, measurement, deanonymization,"in Proceedings of IEEE Symposium on Security and Privacy (SPЎ13). IEEE Computer Society, 2013.

[10] https://www.torproject.org/

[11] Брюс Шнайер. Прикладная криптография. 2-ое издание. Триумф. 2002

[12] Androulaki E et al. "Evaluating user privacy in bitcoin."International Conference on Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2013.

[13] Koshy P, Koshy D, McDaniel P. "An analysis of anonymity in bitcoin using p2p network traffic."International Conference on Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2014.

[14] Reid F, Harrigan M. "An analysis of anonymity in the bitcoin system."Security and privacy in social networks. Springer New York, 2013. 197-223.

[15] Succinct Non-Interactive Zero Knowledge for a von Neumann Architecture, Eli Ben-Sasson,Alessandro Chiesa, Eran Tromer, Madars Virza. May 19, 2015.

[16] Элементарное введение в эллиптическую криптографию. А.А. Болотов, С.Б. Гашков, А.Б. Фролов. КомКнига, 2006.

[17] Zcash Protocol Specifcation Version 2018.0-beta-12, Sean Bowe, Daira Hopwood, Taylor Hornby, Nathan Wilcox, March 6, 2018.

[18] P. Gaudry, F. Hess and N. Smart, Constructive and destructive facets ofWeil descent on elliptic curves Journal of Cryptology, 15 2002, 19-34.

[19] ФАТРМ. Технический комитет 026. кКриптографическая защита информациинь. Методические рекомендации по заданию параметров эллиптических кривых в соответствии с ГОСТ Р 34.10-2012

[20] J. Jeong and T. Kim, "Extended tower number field sieve with application to finite fields of arbitrary composite extension degree Cryptology ePrint Archive: Report 2016/526, 2016

[21] Challenges with Assessing the Impact of NFS Advances on the Security of Pairing-based Cryptography. Alfred Menezes, Palash Sarkar, Shashank Singh, Cryptology

ePrint, 2016

[22] R. Barbulescu, P. Gaudry and T. Kleinjung, "The tower number field sieve Advances in Cryptology - ASI- ACRYPT 2015, LNCS 9453 (2015), 31-55.

[23] Bitnodes.io, "https://getaddr.bitnodes.io"