УДК 004.056

П. Б. Демочкина

МАСШТАБИРУЕМЫЕ zk-SNARK

Описан общий принцип работы (этапы, используемые параметры) схемы доказательства с нулевым разглашением zk-SNARK, возможность и перспективы его улучшения. Реализации zk-SNARK, которые известны на данный момент, имеют ограничения масштабируемости, обусловленные величиной доказываемого вычисления. Во-первых, размер доказывающего ключа по крайней мере линейно зависит от верхней границы структуры, в которой мы работаем. Во-вторых, при доказательстве требуется запись всех предыдущих шагов. В статье описывается алгоритм достижения новой реализации zk-SNARK с использованием криптографии на эллиптических кривых, особенностей структуры поля и цельности доказательств. Практически такая реализация является рекурсивной композицией доказательства, при этом генерация ключей для любых размеров вычислений несет постоянные расходы по памяти. В дальнейшем в процессе доказательства осуществляются только постоянные мультипликативные по времени и аддитивные по памяти расходы. Таким образом, описанная реализация zk-SNARK наделяется двумя важными свойствами: емкость и инкрементальная вычислимость.

This article describes the General principle of operation (stages, parameters used) of the ZK-SNARK zero-knowledge proof scheme, the possibility and prospects for its improvement. Implementations of zk-SNARK that are currently known have scalability limitations that depend on the magnitude of the computation being proved. First, the size of the proving key depends at least linearly on the upper bound of the structure in which we work. Second, the proof requires a record of all previous steps. The article describes an algorithm for achieving a new implementation of zk-SNARK using elliptic curve cryptography, field structure features, and proof integrity. In practice, this implementation is a recursive composition of the proof, while generating keys for any size of calculations carries constant memory costs. Subsequently, the entire process of proof is solely multiplicative constant costs overtime and additivecosts in memory. Thus, the described implementation of zk-SNARK has two important properties: capacity and incremental computability.

Ключевые слова: zk-SNARK, Zcash, доказательство с нулевым разглашением, масштабируемость, инкрементально вычислимый.

Keywords: zk-SNARK, Zcash, zero-knowledge proof, scalability, incrementally computable.

Введение

С течением времени появляются все новые криптографические методы для обеспечения анонимности пользователей компьютерных систем. Однако, создавая новые, имеет смысл использовать и усовершенствовать

уже, казалось бы, хорошо изученные. Так, чтобы обеспечить доказательство с нулевым разглашением, возможно использовать zk-SNARK, который является механизмом работы криптовалюты Zcash. Насколько эффективна данная схема доказательств и как можно ее улучшить, мы рассмотрим в представленной статье.

Актуальность этого вопроса обусловлена тем, что в настоящее время криптовалюты активно внедряются в финансово-экономический спектр мира, и повышение их безопасности и скорости использования улучшит их характеристики и позволит расширить круг их пользователей.

Сущность и механизм работы zk-SNARK

Zk-SNARK — доказательство с нулевым разглашением, основанное на неинтерактивной аргументации, то есть доказательство о владении определенной информацией производится без раскрытия самой информации, при этом взаимодействие между проверяющим (verifier) и доказывающим (prover) минимально. Такое доказательство возможно проверить в течение нескольких миллисекунд, а само оно может иметь размер лишь несколько сотен байтов даже относительно очень большой программы.

Доказательство zk-SNARK используется в криптовалюте Zcash, платежной системе JP Morgan Chase, основанной на блокчейне, и в качестве средства безопасной аутентификации клиентов на серверах. Фокус нашего внимания направлен на zk-SNARK в Zcash. Строгая конфиденциальность, гарантированная Zcash, исходит из того факта, что экранированные транзакции (поддающиеся проверке, но не со стороны третьих лиц) в Zcash могут быть полностью зашифрованы с помощью блокчейна, но только если их можно проверить относительно правил согласования сети с помощью доказательств zk-SNARK [2].

В настоящее время доказательства zk-SNARK зависят от первоначальной доверенной установки открытых параметров. В Zcash эта начальная фаза настройки называется *церемонией генерации параметров* [3]. Предварительная настройка является важной процедурой, так как если злоумышленник получит доступ к механизму генерации параметров, то он сможет создать фальшивые доказательства. Для Zcash это означает, что злоумышленник может использовать поддельные монеты. Чтобы этого не случилось, Zcash генерирует публичные параметры посредством сложного многоуровневого процесса.

Протокол zk-SNARK состоит из трех алгоритмов: G, P и V.

G — генератор ключей, λ — параметр безопасности, C — функция от публичного значения x (например, факт совершения транзакции) и секретного значения w (то, что нужно доказать, например сумма транзакции). G получает на вход λ и C. Затем производится генерация общедоступных ключей: доказательного ключа pk и ключа проверки vk. То есть $(pk, vk) = G(\lambda, C)$.

P использует в качестве входных данных проверяющий ключ pk, случайный x (общедоступный) и значение w, которое требует доказательства без его раскрытия. Алгоритм P создает доказательство $prf = P(pk, x, w)$.

В качестве входных данных верификатор V принимает ключ доказательства, значение x и доказательство prf, а затем возвращает *True* или *False*.

В настоящее время в основе zk-SNARK лежит редуцированное эйт-спаривание [4, с. 21] на кривой BLS12-381.

Ограничения масштабируемости zk-SNARK

Существующие реализации zk-SNARK имеют серьезные ограничения масштабируемости с точки зрения пространственной сложности, которые зависят от размера доказываемого вычисления. Во-первых, размер доказывающего ключа квазилинеен в верхней границе размера вычислений. Во-вторых, получение доказательства требует записи всех промежуточных значений на протяжении всего вычисления, а затем проведения глобальных операций, таких как быстрое преобразование Фурье.

Можно показать, что, используя новые криптографические методы, основанные на эллиптических кривых и конечных полях, и неопределенности систем доказательства, возможно получить реализацию zk-SNARK, которая практически достигает рекурсивной композиции доказательства. Для генерации ключей, поддерживающих все размеры вычислений, требуется постоянное время. Впоследствии процесс доказательства несет только постоянные мультипликативные расходы по сравнению с исходным временем вычисления и существенные постоянные аддитивные расходы по памяти.

Рассмотрим ограничения масштабируемости подробнее.

Дорогостоящая предварительная доверенная установка. Как упоминалось выше, zk-SNARK требует одноразовой доверенной настройки открытых параметров. В большинстве конструкций zk-SNARK эта настройка является дорогостоящей. Генератор ключей принимает в качестве входных данных верхнюю границу размера вычислений. В таком случае пространственная сложность генератора ключей и размер выходного ключа доказательства по крайней мере линейно зависят от этой верхней границы [1].

Чтобы снизить затраты на дорогостоящую предварительную генерацию параметров, можно сделать схему универсальной, чтобы она могла обрабатывать более одного входного значения. Известные реализации zk-SNARK ограничены пространством даже для скромных вычислений: например, имеющие доказывающий ключ более 4 Гб используются для схем всего с 16 миллионами входов.

Таким образом, дорогостоящая предварительная установка сильно ограничивает масштабируемость zk-SNARK.

Пространственно-интенсивная генерация доказательств. Это ограничение частично связано с первым. Генерация доказательств во всех опубликованных реализациях zk-SNARK имеет большую пространственную сложность. По существу, процесс доказательства требует записи всего вычисления сразу, а затем проведения на его основе глобальных вычислений (таких, как быстрые преобразования Фурье или возведения в степень). В частности, если схема показывает выполнение программы, то доказательство требует записи полного следа промежуточных состояний [1].

Решить данную проблему можно путем использования отдельных частей глобальных алгоритмов в качестве самостоятельных алгоритмов, что позволит повторять вычисления и не запоминать промежуточные состояния. Это значительно уменьшит пространственную сложность, но увеличит временную, поэтому такое решение нельзя считать оптимальным.

В идеале нужно реализовать zk-SNARK, который не страдает ни одним из вышеупомянутых ограничений масштабируемости, то есть удовлетворяющий следующим требованиям:

1. Генерация ключей является дешевой операцией (время зависит только от параметра безопасности) и подходит для всех вычислений (полиномиальной сложности). Такой zk-SNARK называется *полностью емким*.

2. Генерация доказательств производится за счет доказательств правильности вычислений в данный момент. Такой zk-SNARK называется *инкрементально вычислимым*.

Bootstrapping

Bootstrapping (рекурсивная композиция доказательств) zk-SNARK — это система с устойчивой к столкновениям хеш-функцией и предварительной обработкой zk-SNARK.

В описываемом методе каждый шаг требует доказательства утверждения, которое:

1) проверяет достоверность доказательств zk-SNARK на предыдущих шагах;

2) проверяет достоверность на текущем шаге.

Необходимо реализовать верификатор V в виде F-арифметической схемы C_V (представляющей собой подсхему C_{pcd}).

Основной подход: использование PCD-дружественных циклов эллиптических кривых. В предварительной обработке zk-SNARK верификатор V содержит в основном операции на эллиптической кривой над полем F_0, поэтому и выражается наиболее эффективно в виде арифметической схемы F_0. Конечно, если $F_0 = F$, то композиция может быть еще эффективнее, но это равенство математически невозможно.

«Запасной» метод: недетерминированная проверка спариваний. Верификатор zk-SNARK включает в себя, в частности, несколько проверок на

основе спаривания над заданной эллиптической кривой. Тем не менее каждая оценка спаривания является очень дорогой, если не выполняется тщательно.

PCD и PCD-дружественные циклы

Proof-carrying data (PCD) — это криптографический примитив, который сохраняет гарантии безопасности, предоставляемые рекурсивной композицией доказательств [1]. Построив систему PCD, можно использовать ее для получения нового zk-SNARK, который является масштабируемым (то есть полностью кратким и инкрементально вычислимым).

Пусть F — конечное поле, (G, P, V) — предварительная обработка zk-SNARK для F-арифметической схемы. Рекурсивная композиция доказательств проверяет выполнимость F-арифметической схемы C_{pcd}, которая, в свою очередь, проверяет достоверность предыдущих доказательств. Задача состоит в реализации верификатора V в виде F-арифметической схемы C_V — подсхемы C_{pcd}.

Способ записи C_V зависит от алгоритма V, обусловленного выбором эллиптической кривой в реализации zk-SNARK. Для доказательства утверждения о выполнимости FR-арифметической схемы для простого r, экземпляр (G, P, V) использует кривую E, которая определена над конечным полем F_q, где группа $E\left(\mathbb{F}_q\right)$ (группа $\mathbb{F}_q$-рациональных точек) имеет такой порядок r, что r делит $\#E(\mathbb{F}_q)$, в частности $r = \#E(\mathbb{F}_q)$. В таком случае все вычисления V выполняются над F_q или его расширениями до k-й степени, где k является степенью вложения E относительно r.

Для данного способа не подходят следующие случаи:

1. $q = r$. Это идеальный случай, ведь тогда арифметика V выполняется над тем же полем, для которого определен язык NP. Но данное условие невыполнимо, потому что если E имеет степень вложения k относительно r, то r должен делить $q^k - 1$, из чего следует, что $q \neq r$. Даже если $E(\mathbb{F}_q)$ имеет не простой порядок n, но простое r, которое делит n, условие также невыполнимо.

2. *«Длинная арифметика»*, то есть работа с битовыми фрагментами для целочисленных вычислений и взятие по модулю q. Но такой подход способствует увеличению порядка до $\log q$, что очень дорого.

Можно предложить использовать подход *цикл через несколько кривых*. Идея состоит в том, чтобы основывать рекурсивную композицию доказательств не на одном zk-SNARK, а на нескольких, при этом все экземпляры созданы на разных эллиптических кривых, которые совместно удовлетворяют специальному свойству.

Для примера рассмотрим простой случай. Пусть мы имеем два простых числа q_α и q_β, а также эллиптические кривые $E_\alpha / \mathbb{F}_{q_\alpha}$ и $E_\beta / \mathbb{F}_{q_\beta}$

такие, что $q_\alpha = \#E_\beta(\mathbb{F}_{q_\beta})$ и $q_\beta = \#E_\alpha(\mathbb{F}_{q_\alpha})$, то есть размер базового поля одной кривой равен порядку группы другой кривой и наоборот. Построим две предварительные обработки zk-SNARK $(G_\alpha, P_\alpha, V_\alpha)$ и $(G_\beta, P_\beta, V_\beta)$, созданные соответственно на кривых $E_\alpha / \mathbb{F}_{q_\alpha}$ и $E_\beta / \mathbb{F}_{q_\beta}$.

Отметим, что $(G_\alpha, P_\alpha, V_\alpha)$ работает для $\mathbb{F}_{q_\beta}$-арифметической выполнимости схем, но все арифметические вычисления V_α выполняются над $\mathbb{F}_{q_\alpha}$ (или над его расширением). По аналогии это верно и для $(G_\beta, P_\beta, V_\beta)$. Вместо того чтобы каждый zk-SNARK обрабатывал утверждения о своем верификаторе, он обрабатывает утверждения о верификаторе другого zk-SNARK. То есть V_α — $\mathbb{F}_{q_\beta}$-арифметическая схема C_{V_α}, а V_β — $\mathbb{F}_{q_\alpha}$-арифметическая схема C_{V_β}.

Затем можно выполнить рекурсивную композицию доказательств, чередуя две системы доказательств. Грубо говоря, можно использовать P_α, чтобы доказать успешную проверку доказательств по C_{V_β}, и, наоборот, P_β для успешной верификации доказательства по C_{V_α}. Выполнение этого чередования гарантирует, что поля «совпадают», и никакая длинная арифметика не требуется.

В случае когда две кривые E_α и E_β таким образом облегчают построение PCD, мы говорим, что (E_α, E_β) — PCD-дружественный 2-цикл (длины 2) эллиптических кривых. Таким образом, можно получить обобщенное определение.

Определение. Пусть $E_0, \ldots, E_{l-1}$ — эллиптические кривые, определенные над конечными полями $F_{q_0}, \ldots, F_{q_{l-1}}$ соответственно, где каждое q_i — простое число. Будем говорить, что $(E_0, \ldots, E_{l-1})$ — *PCD-дружественный цикл* длины *l*, если каждая E_i является дружественной к спариванию и, кроме того, для любого $i \in \{0, \ldots, l-1\}$ имеет место $q_i \neq \#E_{i+1 \bmod l}(\mathbb{F}_{q_{i+1 \bmod l}})$ [1].

Масштабируемые zk-SNARK

Теоретически построение можно описать следующим образом: нужно установить предварительную обработку zk-SNARK (G, P, V) и устойчивую к столкновениям хеш-функцию *H*. Цель состоит в том, чтобы построить полностью емкий вычислимый zk-SNARK (G^*, P^*, V^*) для доказательства / проверки правильности выполнения на данной машине произвольного доступа *M*. Преобразование можно описать в четыре этапа.

Шаг 1: от zk-SNARK до PCD. Первым шагом, не зависящим от *M*, является построение PCD $(\mathbb{G}, \mathbb{P}, \mathbb{V})$ с использованием zk-SNARK (G, P, V). Этот шаг включает в себя рекурсивную композицию доказательств zk-SNARK.

Шаг 2: делегирование памяти машины. Второй шаг предполагает уменьшение размера занимаемой памяти машины M путем делегирования ее оперативной памяти надежному хранилищу при помощи деревьев Меркла. Так M будет загружать значения из памяти как недетерминированные предположения вместе с их алгоритмами аутентификации, которые проверяются с помощью корня дерева Меркла и хеш-функции H. Таким образом, все состояния M включают только состояние процессора и корень дерева Меркла, который «суммирует» память.

Шаг 3: разработка предиката $\Pi_{M,H}$ *для пошаговой проверки*. Третий шаг заключается в разработке предиката соответствия $\Pi_{M,H}$, гарантирующего, что единственными $\Pi_{M,H}$-совместимыми сообщениями z будут те, которые представляют собой результат правильного выполнения (модифицированной) машиной M, по одному шагу за раз, что соответствует свойству инкрементальной вычислимости. Но поскольку $\Pi_{M,H}$ вызывается для проверки только одного шага за один раз, его можно использовать лишь в схемах постоянного размера.

Шаг 4: построение новой системы доказательств. (G^*, P^*, V^*) — это новая реализация zk-SNARK, построение которого заключается в следующем: G^* устанавливается на место G, вызываемого на $\Pi_{M,H}$; P^* использует PCD доказывающего P для пошагового доказательства правильного выполнения M, проводя инкрементальные распределенные вычисления. V^* использует V для установления $\Pi_{M,H}$-соответствия. В общем, поскольку $\Pi_{M,H}$ мал и достаточен для всех вычислений, новый zk-SNARK масштабируется: он полностью емок; более того, поскольку новый проверяющий вычисляет доказательство для каждого нового шага на основе предыдущего, он также является инкрементально вычислимым.

Таким образом zk-SNARK можно улучшить и использовать более эффективно.

Заключение

Мы описали способ избавления от проблем масштабируемости в zk-SNARK, что позволило сделать вывод: криптографические методы, в частности схема доказательств zk-SNARK, поддаются улучшениям эффективности их работы. Есть ли еще какие-либо возможности по повышению эффективности работы данной схемы?

Известно, что доказательства zk-SNARK осуществляются на основе спаривания на эллиптических кривых. Возникает вопрос: будет ли более эффективным применить к этой схеме гиперэллиптическую кривую, не уступающую по уровню безопасности? Таким образом, актуальной задачей является поиск нового математического объекта, использование которого в zk-SNARK будет более эффективным, чем применение эллиптической кривой.

Список литературы

1. *Sasson E., Chiesa A., Tromer E., Virza M.* Scalable Zero Knowledge via Cycles of Elliptic Curves // Algorithmica. 2017. №79 (4). P. 1102—1160.

2. *Что такое* zk-SNARK? // Z.CASH.RU : офиц. сайт криптовалюты Zcash. URL: https://z.cash/ru/technology/zksnarks (дата обращения: 07.11.2020).

3. *Что такое* zk-SNARKs и zk-STARKs? // Binance Academy : [сайт]. URL: https://academy.binance.com/ru/articles/zk-snarks-and-zk-starks-explained (дата обращения: 07.11.2020).

4. *Мельничук Е. М., Новоселов С. А.* Характеристические многочлены некоторых гиперэллиптических кривых родов 2, 3 и p-ранга // Прикладная дискретная математика. Приложение. 2019. №12. С. 21—24.

Об авторе

Полина Борисовна Демочкина — студ., Балтийский федеральный университет им. И. Канта, Россия.

E-mail: polly.evseeva@gmail.com

The author

Polina B. Demochkina, Student, Immanuel Kant Baltic Federal University, Russia.

E-mail: polly.evseeva@gmail.com