239

Б. Е. Пошерстник, М. Я. Пратусевич

БИТКОИН. БЛОКЧЕЙН. КРИПТОВАЛЮТЫ.

Лекции для старшеклассников

Полная документация.
От концепции до двоичного кода.

- Шифрование
- Хэширование
- ЭЦП
- Эллиптические кривые
- Публичные и приватные ключи
- Хэшчейн
- Блокчейн
- Proof of work
- Задача византийских генералов
- Lightning Network

Современная математика... не удовлетворяет критерию полезности.

Г. Х. Харди

ВСЕ ЖЕЛАЮЩИЕ ПРИОБРЕСТИ
КНИГУ МОГУТ ЭТО СДЕЛАТЬ
НА САЙТЕ ИЗДАТЕЛЬСТВА
WWW.SMIO.RU

Б. Е. Пошерстник, М. Я. Пратусевич

БИТКОИН. БЛОКЧЕЙН. КРИПТОВАЛЮТЫ

Лекции для старшеклассников

Санкт-Петербург
СМИО Пресс
2023

СМИО
ПРЕСС

УДК 519.719
ББК 22.184
П66

Пошерстник Б. Е., Пратусевич М. Я.
П66 БИТКОИН. БЛОКЧЕЙН. КРИПТОВАЛЮТЫ.
Лекции для старшеклассников. —
СПб: СМИО Пресс, 2023 — 200 с.

Книга «Биткоин. Блокчейн. Криптовалюты» возникла из цикла лекций, прочитанных для учеников 10 класса одним из авторов в школе, возглавляемой вторым автором.

В книге на доступном для школьников уровне рассмотрены криптовалюты с точки зрения экономики, математики, особенностей технической реализации. Подробно описаны идеи, совокупность которых легла в основу создания биткоина: асимметричное шифрование, шифрование на эллиптических кривых, электронная подпись, хэширование и т. д.

С одной стороны, авторы не устают восхищаться красотой идей, лежащих в основе решения «нерешаемой» задачи анонимной передачи ценностей без наличия единого центра, а с другой — предостерегают от бездумного и легковесного отношения к криптовалютам как светлому будущему развития мировой экономики.

Книга будет полезна всем желающим понять, что такое криптовалюты и как они устроены.

© Пошерстник Б. Е.,
Пратусевич М. Я., 2023

ISBN 978-5-7704-0396-1

© СМИО Пресс, 2023

ООО «Издательство «СМИО Пресс», включено в перечень организаций, осуществляющих издание учебных пособий, которые допускаются к использованию в образовательном процессе в имеющих государственную аккредитацию и реализующих образовательные программы общего образования образовательных учреждениях.
Приказ Минобрнауки РФ № 699, от 9 июня 2016 г.,
зарегистрирован в Минюсте 4 июля 2016 г., № 42729.

Художник *Кувакина Дарья* (*Kuvakina Daria*).
Художественный редактор *Соловьева Н. Д.*
Редактор *Золина Н. К.*
Дизайнер *Зайцев А. А.*
Директор издательства *Морозова И. С.*

Подписано в печать 18 декабря 2022. Формат 70S100 1/16. Бумага офсетная.
Гарнитура школьная. Усл.-печ. л. 1. Тираж 500 экз. Заказ № .
Издательство «СМИО Пресс»
Санкт-Петербург, ул. Камчатская д. 1.
Телефоны (812) 976-94-76, (911) 290-90-26 (МТС),
e-mail: smiopress@mail.ru, http://www.smio.ru
Отпечатано в ООО «Типография Лесник»
197183, г. Санкт-Петербург, ул. Сабировская, д. 37.

Посвящается, безусловно, лучшей школе
в мире — Президентскому физико-
математическому лицею № 239
г. Санкт-Петербурга.

Авторы выражают благодарность ученикам лицея, которые помогли в тестировании и проведении лекций, на основе которых написана данная книга. Это помогло нам объяснить сложнейший материал на уровне, доступном для понимания учеников 10 класса.

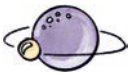
Авторы выражают отдельную благодарность ученикам 10 класса, которые внесли большой вклад в развитие книги:
Глебу Раткевичу и Арсению Морозову.

СМИО
ПРЕСС

Условные обозначения в книге

Примечание общего характера. То, что мы хотим выделить или обобщить.

Информация, которая будет интересна только тем, кто хочет глобально разобраться в вопросе. Не требуется для понимания остального материала.



Сторонняя информация, которая не требуется для понимания текущей темы и служит просто для расширения кругозора.

Очень часто мы будем пересказывать или напоминать уже изученные факты. Эти блоки созданы, чтобы напоминать вам об уже изученных концепциях. Могут раздражать внимательного читателя, но необходимы, чтобы держать вас в тонусе.

СМИО
ПРЕСС

Кому предназначена книга

Данная книга предназначена прежде всего для школьников, планирующих стать программистами, работающими в блокчейн-индустрии. Согласно исследованиям, знание блокчейн-технологии повышает зарплату на 40–70%, а также открывает новые вакансии. Согласно сайту hh.ru, зарплата программиста, знакомого с блокчейн-технологией, начинается от 150 000 р. и составляет в среднем 300 000 р. Также книга может быть интересна студентам математических вузов, ученикам математических кружков и школьникам старших классов специализированных школ.

Параллельно с изучением биткоина вы сможете начать знакомиться с огромным пластом современной криптографии, основанной на довольно сложной математике. Это серьёзно поднимет ваш общий уровень знаний в указанных областях. Хэширование, асимметричное шифрование, доказательство с нулевым разглашением и т. д. — всё это используется не только в криптовалютах, но и в огромном числе других задач.

Книга не требует специальных знаний, однако у вас не должно быть трудностей с пониманием школьного курса математики.

Мы не приводим в книге объяснения некоторых второстепенных или не нужных для понимания элементов. Некоторые стандартные математические концепции мы не включили в книгу просто потому, что они общеизвестны, не влияют на изложение, и эти знания вам не потребуются для понимания происходящего. К примеру, в главе 5 мы пишем, что для получения корней уравнения используется расширенный алгоритм Евклида, но не приводим его. В главе 7 мы упоминаем, что точки на эллиптической кривой образуют группу, и даём пример, как считается их сумма, но не углубляемся в дальнейшие рассмотрения, связанные с теорией групп.

Пропущенные алгоритмы и понятия полностью раскрываются в десятках и сотнях обычных учебников математики.

Все сравнения и аллегории были выбраны после теста на лекциях для школьников. Они могут вызывать улыбку у более подготовленного человека (как сравнение хэширования и отпечатка пальца), однако были специально выбраны, чтобы максимально понятно и подробно раскрыть суть явления.

Все важные концепции рассмотрены с нескольких сторон. Изложение протестировано на десятиклассниках Президентского физико-математического лицея № 239 г. Санкт-Петербурга. Авторы считают, что материал книги доступен любому человеку с основным общим образованием (9 классов).

СМИО ПРЕСС

Вступление и концепция книги

Книга подробно рассматривает все математические аспекты криптовалют и основные принципы их работы.

К сожалению, «хайп» вокруг данной темы приводит к огромной путанице в терминах. Блокчейном могут называть вообще любую базу данных, а криптовалютами любые электронные деньги (многие называют криптовалютами даже Yota или XRP). Поэтому при описании криптовалют мы используем гораздо более чёткий термин «биткоин», однако, за исключением мелких частностей, всё описанное в книге будет верно и для других криптовалют.

Таким образом, термин «биткоин» выбран нами для упрощения терминологии, так как этот термин гораздо более точен и понятен, чем термины «криптовалюты», «блокчейн» или «распределённый реестр».

Фактически, создатель биткоина Сатоши Накамото не создал ничего нового, просто гениально соединил уже созданные ранее элементы. Создание биткоина, это как создание свода храма, когда кирпичи, опираясь друг на друга, создают прочную конструкцию.

Условная схема концепций, из которых состоит биткоин

Каждая из этих концепций очень подробно рассмотрена в отдельных главах. После прочтения каждой главы у вас должно возникнуть чёткое понимание и умение производить необходимые действия и вычисления.

Мы специально не стали вставлять в книгу программный код, который обычно занимает около 30% подобных книг. Опыт показал, что он просто пропускается читателем. Мы решили сконцентрироваться на очень подробном описании концепции и технической реализации.

Также в книге вообще не рассмотрена конкретная работа с кошельками и другими программами. Мы в принципе не уверены в том, что экосистема биткоина готова к массовому принятию. Юридическая основа криптовалют до сих пор не определена, поэтому, используя их, вы можете преступить закон. К тому же информация о конкретных продуктах чрезвычайно быстро устаревает.

В первую очередь, мы стремились написать книгу, понятную для любого достаточно подготовленного читателя (знакомого со школьным курсом математики). Книга тестировалась на учениках 10 класса. Любая недостаточно очевидная концепция рассматривается с нескольких сторон.

К сожалению, многие концепции, которые описаны в этой книге, неочевидны. Часто возникает ощущение, что «автор ошибся» или «автор, наверняка, не это имел в виду». Для того чтобы люди проще принимали криптовалюты, их создатель оставил стандартную терминологию финансового мира, но теперь такие термины, как «адрес», «кошелёк», «владелец средств», означают на самом деле не то, что ожидает пользователь.

Мы, объясняя суть этих вещей, сталкиваемся с непониманием, почему знакомые термины теперь означают совсем не то, к чему все привыкли. Возможно, это самая трудная часть данной книги.

При прочтении глав, особенно в третьей части книги, мы постоянно призываем вернуться и ещё раз перечитать главы, посвящённые отдельным используемым терминам.

Пожалуйста, высылайте нам свои заметки и комментарии, которые, несомненно, помогут доработать книгу. Присылайте претензии и предложения на crypto@239.ru — и мы упомянем вас во второй версии книги!

СМИО
ПРЕСС

Краткое содержание книги

Часть 1. ФИНАНСОВАЯ И ЮРИДИЧЕСКАЯ СТОРОНА

В этой части мы опишем все основные концепции, которые использует биткоин. Здесь нет математики. Эту часть можно рассматривать как постановку задач, которые должны решить математики. Также мы постарались доказать, что биткоин — это не просто красивая математическая модель. У биткоина есть серьёзная идеологическая платформа и с финансовой, и с юридической точки зрения.

Глава 1. Идеология денег. Краткая глава, которая рассказывает об идеологии денег. Что мы можем называть деньгами? Обеспечение денег. Рассматривается альтернативное определение денег и зачем нужны деньги.

Глава 2. Идея отсутствия удостоверяющего центра. Описание самого смысла распределённых финансов и проблем, которые они решают.

Глава 3. Отсутствие доверия между сторонами сделки при передаче информации. Задача византийских генералов. Возможность передачи ценности при недоверии между сторонами, а также в случае недоверия к самому каналу передачи информации. Преимущества и недостатки такого подхода и почему блокчейн возможен только для криптовалют, но не для записи другой информации. Проблема была сформулирована как «задача о византийских генералах», и в этой главе мы подробно обсудим решение, которое предложил Сатоши Накамото.

Глава 4. А при чём тут биткоин. Мы очень подробно будем обсуждать биткоин в третьей части. В этой главе мы лишь очень кратко свяжем термины из реального мира, которыми оперировали, с терминами, которые вводит биткоин.

Часть 2. МАТЕМАТИЧЕСКАЯ ОСНОВА

Глава 5. Вступление в техническую часть. Мы хотим показать, что блокчейн — это вершина и соединение технологий, которые были созданы до его изобретения. По сути, он является лишь соединением известных до этого идей: асимметричное шифрование, хэшчейн, Proof of Work, доказательство с нулевым разглашением. Далее мы будем разбирать каждую из этих технологий подробно в каждой главе.

Глава 6. Симметричное и асимметричное шифрование. Разобрана идея симметричного шифрования, какие проблемы содержит и для чего

СМИО ПРЕСС

понадобилось асимметричное шифрование. Проблемы и идеология асимметричного шифрования. Использование метода асимметричного шифрования, основанное на факторизации числа. Метод RSA.

Глава 7. Доказательство с нулевым разглашением. Доказательство наличия решения, при котором не раскрывается само решение.

Глава 8. Хэширование. Идея обычного и криптографического хэширования. Объяснение смысла и примеры использования хэширования в различных ситуациях.

Глава 9. Электронно-цифровая подпись. В главе практически не содержится математических расчётов, но подробно изложен смысл цифровой подписи: для чего она создаётся и как используется в расчётах.

Глава 10. Асимметричное шифрование на эллиптических кривых. Одно из красивейших применений современной математики. Рассказывается об алгебраической операции на эллиптических кривых, использовании её в концепции криптографии на эллиптических кривых.

Глава 11. Создание приватного и публичного ключей [1]. Теория и практика создания приватного и публичного ключей на эллиптических кривых[2].

Глава 12. Proof of Work. Система доказательства работы и право включения в хэшчейн.

Часть 3. КОНЦЕПЦИЯ И KNOW-HOW БИТКОИНА

Глава 13. Создание биткоин-адресов. Двойная защита вашей информации.

Глава 14. Общая теория передачи ценности в криптовалютах. Глава, в которой описывается передача ценности от одного человека другому в ситуации, когда ценность не имеет физического выражения.

Глава 15. Хэшчейн. Идеология хэшчейна, цепочка хэшей и описание проблем, которые он решает. Почему необходимо создавать блокчейн, а хэшчейн не подходит для решения проблемы передачи ценности.

Глава 16. Проблема двойных трат. Постановка проблемы, которую решает биткоин. До этого мы фактически говорили о проблемах, которые

---

[1] *Мы используем термины «публичный» и «приватный», однако часто можно встретить другую пару терминов: «открытый» и «закрытый». Они абсолютно эквивалентны.*

[2] *В принципе, любая современная криптовалюта создаёт ключи аналогичным способом, поэтому данный способ относится к 90% криптовалют.*

были решены до биткоина. Объяснение, почему понадобился блокчейн и какую именно проблему решает.
Объяснение решения этой проблемы займёт следующие несколько глав.

Глава 17. Блокчейн. Решение проблемы двойных трат. Теоретическое и практическое описание проблемы консенсуса и технический разбор «проблемы византийских генералов».

Глава 18. Подробное описание блока на основе блока биткоина. Очень подробное описание блока, как он создаётся и что он содержит.

Глава 19. Консенсус на основе Proof of Work и необходимость майнинга.

Глава 20. Дальнейшее развитие. Подробное описание дальнейших идей биткоина. Куда движется экосистема и что можно ожидать с математической и практической точки зрения.

Приложение 1. **Биткоин и его особенности.**

Приложение 2. **Различные методы консенсуса.**

Приложение 3. **Альткоины и их различия.** Краткое описание, чем могут отличаться альткоины. Эфир, лайткоин, монеро, даш и тому подобное.

Приложение 4. **Различные идеи и термины в современном криптомире.**

Приложение 5. **Риски биткоина.**

СМИО
ПРЕСС

Часть 1

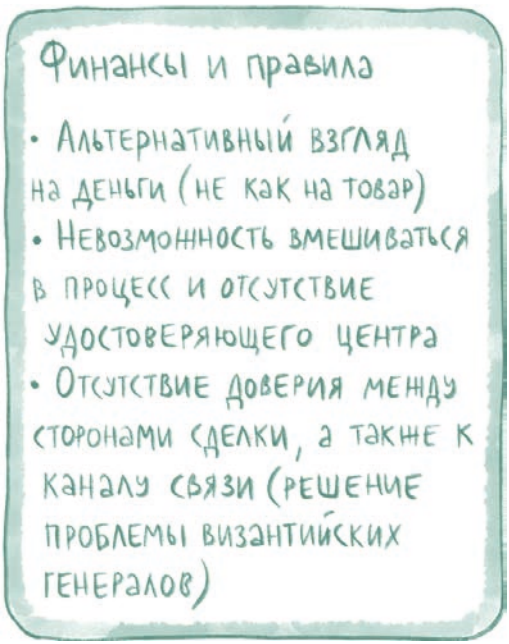
ФИНАНСОВАЯ И ЮРИДИЧЕСКАЯ СТОРОНА

Это единственная часть книги, где не будет использована математика. Мы хотим лишь показать, что биткоин — не просто красивая математическая модель. Биткоин является результатом долгого пути не только математической науки, но и финансовых инструментов.

Важно: мы не пытаемся ответить на все вопросы, в этой части мы даём лишь общую картину, список идей и основную реализацию концепции. Опыт лекций показал, что эта часть является самой сложной для понимания. Возможно, будет необходимо прочитать её несколько раз. Также мы советуем после прочтения второй части снова вернуться к этим главам.

Из-за очень большой сложности материала некоторые идеи мы объясняем несколько раз, с разных сторон. Такое подробное изложение многим может показаться избыточным.

В этой части мы рассматриваем идеологическую часть нашей большой карты.



Глава 1
ИДЕОЛОГИЯ ДЕНЕГ И БИТКОИНА

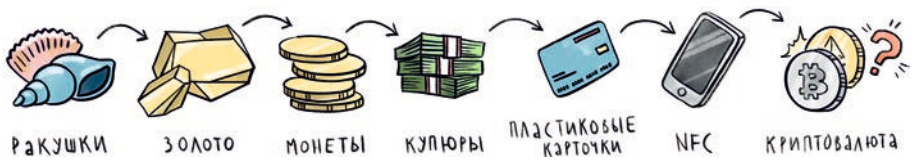
К теме криптовалют можно подойти с двух сторон — технической и идеологической. Книга на 90% состоит из описания технической части, чтобы вы могли досконально понимать идею, работать и даже изменять криптовалюты, создавая новые. Но в этой главе сделан больший упор на идею и финансовые принципы.

Финансисты обычно окружают свою теорию особыми терминами, которые могут выглядеть очень наукообразно, но на самом деле имеют очень простой и прозрачный смысл (кризис ликвидности, производная роста, эластичный спрос и т. д.).

Мы надеемся, что данная глава поможет понять, почему криптовалюты важны для будущего развития денег, почему их стоит изучить и почему это не мимолётная мода, а долгосрочный тренд в финансовом мире.

Уверены, что после прочтения этой главы вам будет проще заставить себя изучить и техническую сторону криптовалют.

Рис. 1



Постепенное развитие денег [1]

На протяжении своей истории человек пользовался разными видами денег. Как нам говорят историки, сначала это могли быть редкие и красивые в данной местности вещи — ракушки, красивые камни, шкурки енотов. Потом человечество перешло на куски драгоценных металлов — серебро, золото. Позже были изобретены бумажные деньги, электронные деньги и так далее. Всё это считают деньгами. Это важно отметить для следующего аргумента.

[1] *Разумеется, развитие условное. Видов денег было великое множество, а NFC это лишь способ связи – просто такая схема наиболее ярко отражает развитие способов передачи ценности.*

СМИО ПРЕСС

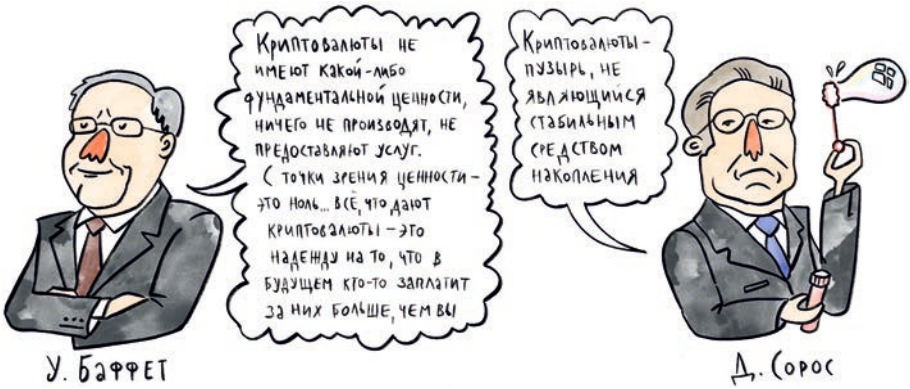


Рис. 2

Аргументы противников криптовалют

Согласно современной финансовой теории, деньги являются наиболее ликвидным товаром, который должен иметь внутреннюю ценность. По утверждению экономистов, исторически деньги возникли как наиболее нужный товар, он был нужен всем, а значит, его можно было обменять на любой другой товар.

То есть, когда один человек убивал оленя, а хотел получить копьё, он менял оленя на то, что нужно всем, а потом менял это что-то, нужное всем (ликвидный товар), на копьё.

Рис. 3

Классическая теория происхождения денег. Один создал копьё, другой убил оленя. Убивший оленя меняет его на некий «ликвидный товар». Потом за него приобретает копьё

Так многие экономисты подводят черту под использованием биткоина. Биткоин никому не нужен сам по себе — значит, у него нет внутренней ценности — значит, это не товар — значит, это просто пирамида (схема Понци, тюльпаномания и так далее). А значит, криптовалюты не являются деньгами и их нельзя использовать.

Однако в этой теории есть явные недостатки. Из рис. 1 видно, что деньги никогда не являлись товаром с внутренней ценностью. Никому из древних людей не были нужны ракушки и даже куски красивого, но бесполезного

СМИО ПРЕСС

золота. Тем более никому не нужны банкноты и цифры на банковском счету. По факту, они не является товаром. Тем более ликвидным.

Альтернативная теория происхождения денег состоит в том, что это долговая расписка. Если племя убивало мамонта, а у соседнего племени был голод, то соседи могли отдать красивые ракушки за мамонта, и эти ракушки были просто знаком, что они должны будут отдать часть своего мамонта, когда им повезёт на охоте.

Рис. 4

Альтернативная теория происхождения денег. Первый этап – запись о долге

Человек отдаёт кусок мяса за бесполезную ракушку только для того, чтобы, когда ему не повезёт на охоте, он смог отдать ракушку обратно за такой же кусок.

Рис. 5

Второй этап – возврат долга

Но если он считает, что мясо ему не понадобится, а ему нужно копьё, то происходит следующее:

Рис. 6

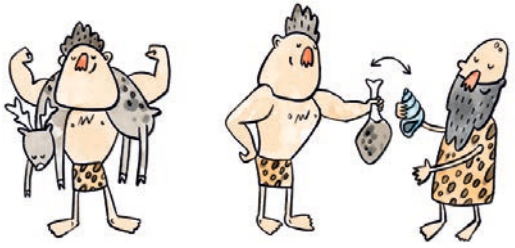


Рис. 7

Убив оленя, охотник отдаёт кусок за ракушку (рис. 6), которая является записью о долге «обладатель ракушки имеет право забрать у меня кусок мяса».

Получив ракушку, он меняет её на копье (рис. 7). Человек отдаёт копьё не за ракушку, а потому что знает: он может выменять её на мясо.

СМИО ПРЕСС

То есть если человек убивает оленя, он может отдать кусок за ракушку, идёт и покупает копьё за эту ракушку и говорит: «Острый Глаз должен мне кусок оленя — я передаю этот долг тебе за копьё». Третий человек идёт и получает за ракушку кусок оленя (рис. 8).

Рис. 8

В отличие от классической теории «деньги — это наиболее ликвидный товар», в альтернативной теории «деньги — это запись о долге» нет внутренних противоречий. Мы видим, что деньги ни в древности, ни в современном мире не содержат внутренней ценности. Так что же такое деньги?

Деньги практически всегда не являются чем-то нужным и самоценным. Они нужны лишь для того, чтобы договориться об обмене или записать информацию о долге.

Древнее племя могло убить мамонта, другое племя — прийти к ним и попросить у них кусок мяса. И дать за него что-то очень редкое, к примеру, удивительную ракушку. Главное, чтобы это легко было идентифицировать и очень сложно найти.

Когда возникала обратная ситуация, и второму племени удавалось убить мамонта, то первое племя могло обменять обратно ракушку на мясо. Так возникли деньги. У них никогда не было внутренней стоимости. И она им не нужна.

Можно услышать такой аргумент. Да, древнейшие деньги (ракушки и т. д.) не имели внутренней ценности и не являлись товаром. Современные деньги также не являются товаром — ни купюры, ни тем более символы на банковском счёте. Но в истории человечества был период, когда деньгами являлись куски драгоценных металлов, обычно золота или серебра. И они точно имели внутреннюю ценность. Ведь из золота можно делать украшения и посуду.

Но при более серьёзном изучении оказывается, что не золото ценное, потому что из него можно делать украшения. А украшения делаются из золота, потому что оно является ценностью.

В качестве примера можно вспомнить историю алюминия. Когда-то этот металл был очень дорог и посуду из алюминия император Наполеон III давал только самым дорогим гостям. Остальным же приходилось довольствоваться приборами из золота.

Но когда алюминий стал гораздо дешевле, вещи из него исчезли из статусных сервизов. Из него перестали делать монеты и украшения.

Совсем недавно, в связи с развитием электротехники, золото и серебро действительно стали очень нужными товарами. Однако нужно заметить, что именно в этот момент они перестали быть деньгами, что является ироничным

Рис. 9

Алюминий считался драгоценным металлом из-за редкости, и поэтому из него делали монеты, драгоценную посуду и украшения

свидетельством того, что деньги могут не быть товаром (и, по нашему мнению, почти никогда им не являются, вопреки расхожим представлениям).

Таким образом, мы полагаем, что в деньгах нет внутренней ценности. Как только нечто становится товаром с внутренней ценностью (к примеру, золото в эпоху электротехники), оно сразу перестаёт быть деньгами.

Что же такое деньги? Мы считаем, что это слово нельзя определить через более простые понятия и оно является базисным. Мы часто сталкиваемся с такими понятиями, которые знакомы всем, но которые нельзя определить более простыми словами. К примеру, время и пространство, точка и прямая. Мы определяем их через свойства. Например, время имеет свойства однородности, однонаправленности и одномерности. А пространство — трёхмерности, однородности и изотропности.

Так же и деньги. Деньгами становится то, что обладает в максимальной степени следующими свойствами:

Ограниченность
Транспортабельность
Долговечность
Делимость
Узнаваемость
Ценность
Защищённость

СМИО ПРЕСС

Разберём каждое из этих свойств.

• **Ограниченность.** Возможно, самое важное свойство денег. Денег должно быть мало. Нельзя считать деньгами то, что легко достать. К примеру, деньгами не может стать комок земли. При этом ограниченность может проявляться только в конкретной местности. Известны вулканические острова, где деньгами являлись камни, что невозможно для средней полосы России.

• **Транспортабельность.** Очевидно, что деньгами с большей вероятностью станет то, что легко транспортировать. Поэтому электронные деньги побеждают бумажные.

• **Долговечность.** Деньги должны храниться долго. Деньгами не могут стать куски мяса, потому что они быстро портятся. А золото имеет огромное преимущество перед железом благодаря отсутствию окисления.

• **Делимость.** Деньги нужно уметь делить. Поэтому ракушки очень быстро проигрывают куску металла.

• **Узнаваемость.** Должно быть легко определить, что это деньги (в том числе, отличить от подделок). Удивительно, но здесь ракушки выигрывают у золота и даже у банкнот, поэтому золотые монеты быстро сошли на нет, а банкноты вынуждены всё время эволюционировать.

• **Ценность.** Это свойство перекликается со свойством транспортабельности и редкости. Деньги должны содержать большую ценность в маленьком объёме. Потому железо не может быть деньгами, а золото — вполне.

• **Защищённость.** Ваши деньги должно быть тяжело украсть и завладеть ими.

Можно смотреть на историю денег как историю попыток нахождения или создания того, что обладает набором этих свойств. И как только появляется нечто, ещё лучше отвечающее этим требованиям, деньги сразу меняются.

В первобытном обществе деньгами становилось что-то очень редкое, но которое достаточно легко переносить, которое не испортится очень быстро, которое легко опознать и так далее. И деньгами становились редкие раковины, зубы особо страшных хищников и что угодно, обладающее нужными свойствами.

Когда люди научились добывать и обрабатывать редкие металлы, оказалось, что они гораздо удобнее всего, что было раньше, — их можно делить. Тем не менее металлы достаточно тяжёлые, их нелегко идентифицировать, понять их ценность не так просто. И их сложно делить.

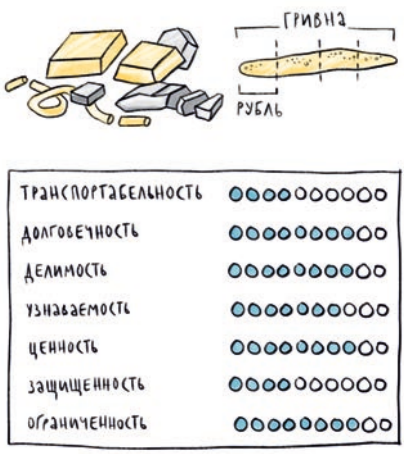
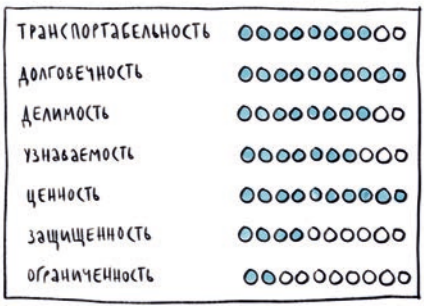


Рис. 10

Свойства металлических денег

ФИАТНЫЕ ДЕНЬГИ
(от лат. fiat — декрет, указание, «да будет так»)

Рис. 11



Банкноты практически идеальные деньги. Их легко транспортировать, делить, они долговечны, их очень легко идентифицировать, они обладают большой ценностью в маленьком объёме (рис. 11). Но есть только одна фундаментальная проблема — их выпуск ничем не ограничен. Любая страна может напечатать их в любом количестве.

Например, за несколько лет, во время «количественного смягчения», федеральная резервная система США напечатала несколько триллионов долларов.

Также есть проблемы с защищённостью.

БИТКОИН

Рис. 12

Теперь мы можем перейти к биткоину (рис. 12).

Биткоин — это настоящие деньги. Не эрзац, не замена, а настоящие деньги. Более настоящие, чем золото или доллары. Хотя, безусловно, в Российской Федерации не имеют статуса платёжного средства.

У биткоина есть недостаток — он не принимается в качестве законного средства платежа у государственных структур, тем не менее и здесь есть изменения.

Эквадор в конце весны 2021 г. сделал биткоин официальным платёжным средством. То есть его обязаны — не просто могут, а обязаны — принимать все. Конечно, там анархия и бедность, но уже сейчас существуют деревни, которые пользуются даже не биткоином, а Lightning Network, описанным в самой сложной части книги. Подробное описание Lightning Network смотрите в главе 20.

Теперь ещё немного о том, чем определяется ценность криптовалюты, в частности биткоина. Необходимо понимать, что есть только один параметр, который определяет ценность любой информационной структуры, в том числе

СМИО ПРЕСС

ценность фейсбука, яндекса или криптовалют, — это количество людей, которые этой структурой пользуются. Это и есть настоящая ценность денег.

У биткоина сложно подсчитать количество пользователей. Можно основываться только на косвенных признаках. К примеру, количество уникальных адресов.

Рис. 13

Этот график можно соотнести с изменением стоимости биткоина[1]:

Рис. 14

Можно заметить, что стоимость биткоина, как и любой другой современной системы, определяется её популярностью. Популярность, в свою очередь, во многом определяется удобством и нужностью системы.

[1] *Конечно, совпадение графиков нельзя считать корреляцией. Тем не менее мы утверждаем, что данные графики показывают зависимость стоимости биткоина от количества тех, кто его использует. Так как мы знаем только количество кошельков, а не их владельцев, то данная зависимость косвенная.*

Рис. 15

Древнейший блокчейн

Древнейший блокчейн

Закончить эту главу мы хотели бы примером древнейшего блокчейна, когда передача ценностей происходит не физически, а с помощью консенсуса.

На острове Яп деньгами были огромные камни. Остров коралловый и камней на нём нет, для их добычи и доставки необходимо преодолевать огромные расстояния и везти камни, масса которых достигает тонны, на лодках.

Понятно, что из рук в руки такую ценность передать сложно, поэтому когда кто-то кому-то платил, просто собирались жители деревни и хозяин камня провозглашал, что теперь камень принадлежит другому.

Есть истории о том, что иногда камни тонули до прибытия на остров. Тем не менее, эти камни всё равно могли участвовать в расчётах. У них был хозяин и он мог передавать ценности. Здесь мы видим достаточно яркий пример современного блокчейна, когда есть майнеры, добывающие камни, а передача ценности может осуществляться только её хозяином при помощи достижения консенсуса, который согласно современным терминам можно было бы называть Proof-of-Personal.

В этой главе мы попытались доказать, что вся история денег — это попытка найти то, что удовлетворяет списку условий (Транспортабельность, Долговечность, Делимость, Узнаваемость, Ценность, Защищённость, Ограниченность).

Биткоин наиболее полно соответствует этим свойствам, а значит, это настоящие деньги, которые лучше того, что было раньше. Таким образом, биткоин — это следующий этап эволюции денег.

Далее мы подробнее поговорим о консенсусе в передаче ценностей.

СМИО
ПРЕСС

Глава 2
ИДЕЯ ОТСУТСТВИЯ УДОСТОВЕРЯЮЩЕГО ЦЕНТРА

Когда деньги перестали иметь физический эквивалент, остро возникла проблема учёта. Ведь если деньги — это запись в компьютерной базе данных, становится возможным просто записать в эту базу любое число.

Для решения этой проблемы база данных должна находиться только на защищённом сервере под контролем удостоверяющего центра, которому все доверяют. Чаще всего это государственная структура, и только ей позволено менять числа в этой базе данных, что эквивалентно эмиссии («печати») денежных средств.

В Российской Федерации эта прерогатива принадлежит Центробанку, а, к примеру, в США — структуре под названием Федеральная Резервная Система. ФРС эмитирует («печатает») денежные средства, а далее даёт их в долг государству под определённый процент.

До создания биткоина считалось, что наличие удостоверяющего центра — это единственный вариант существования электронных денег. Создание биткоина перевернуло ситуацию. Появилась возможность передачи ценности от одного владельца к другому без удостоверяющего центра.

Деньги, у которых нет физического эквивалента и которые находятся лишь в памяти компьютера, переводятся напрямую между участниками сделки, что ранее считалось невозможным! Ведь без единого центра, которому все доверяют, реестр невозможно защищать от подделки или изменения. Казалось, в такой системе каждый может приписать себе любое количество денег.

Ещё раз отметим, когда деньгами являлся какой-то физический предмет, наличие общего центра было ненужным. Передать деньги — означало просто передать предмет. Когда деньги стали лишь записями в базе данных, то фактически передачи денег не происходит. При передаче ценности банк уменьшает сумму на одном счёте и увеличивает её на другом. То есть это не физическая передача — это просто уменьшение хранимого числа в одном месте и увеличение в другом (часто другой суммы с учётом комиссии).

Если мы хотим передавать ценность без наличия центра, необходимо снова сделать передачу денег именно передачей, а не удалением-зачислением, как сейчас происходит в любой банковской системе. То есть деньги

снова должны приобрести некий эквивалент. Разумеется, цифровой, а не физический.

Решение этой проблемы мы подробно рассмотрим в главе 13. Здесь же хотим наметить лишь общую схему.

Итак, наша цель — убрать необходимость удостоверяющего центра. Нам нужно избавиться от возможности кому-то постороннему увеличить или уменьшить сумму на счёте. Способ добиться этого — вообще убрать стандартную схему счетов с записями вида:

Клиент 1 — 1000 р.
Клиент 2 — 1300 р.
...
Клиент 1000 — 299 р.

И вернуться к системе, при которой происходит именно передача чего-то.

Биткоин решил проблему такой передачи следующим образом: любой кошелёк можно представить как сейф с замком.

Обладать ценностью — означает иметь ключ от этого замка. При этом нужно заметить, что сейф стоит в общедоступном для всех месте.

Рассмотрим очень подробно схему передачи денег (ещё раз напомним, что сам сейф общедоступен). Он был ранее кем-то создан, и мы пока не обсуждаем, как именно. В сейфе замок, от которого у Кости есть ключ (рис. 16).

Рис. 16

Сейф с магнитным замком. Уникальный ключ от него есть только у Кости

Рис. 17

Ключ и замок. Каждый создаёт свои собственные

Теперь Костя хочет передать эту ценность кому-то другому. Человек, которому Костя хочет её передать, должен создать ключ и замок — как создаётся ключ и замок, мы подробно разбираем в главе 11. Здесь мы можем просто схематично их изобразить (рис. 17).

Чтобы передать ценность, Костя просто берёт у человека его замок. Именно замок. Ключ от этого замка остаётся у создателя. Также Костя берёт СВОЙ ключ.

СМИО ПРЕСС

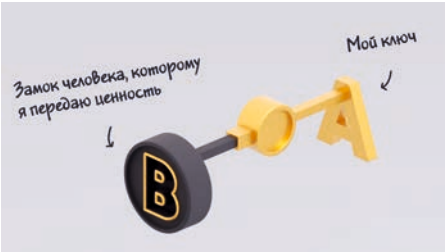


Рис. 18

Приваренный замок-ключ. Открепить одно от другого невозможно, об этом мы будем говорить в главах о ЭЦП – главе 9 и подробнее в главе 14

Далее, «приваривает» этот замок (рис. 18) к своему ключу. Эта очень важная концепция, к которой мы будем постоянно возвращаться.

У Кости получается ключе-замок[1].

Далее ключ можно вставить в сейф. На самом деле верификацию этого действия Костя провести не может, её проводит майнер, но об этом чуть позже. После того, как ключ вставлен в сейф, получается следующая картина (рис. 19).

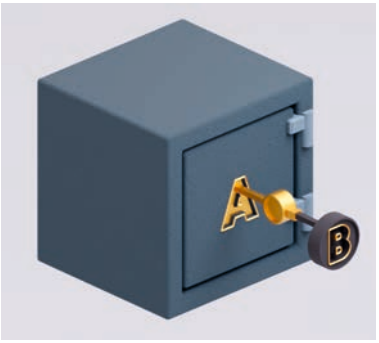


Рис. 19

Теперь на сейфе висит новый замок — замок того человека, которому Костя передал ценность. С этого момента сейф можно открыть только его ключом.

Сейчас Костин ключ уже не будет работать. В принципе невозможно уже ничего вставить в Костин первый сейфовый замок. Работает только замок, взятый у того, кому Костя перевёл деньги.

Это достаточно сложная для понимания концепция. Изучим процесс ещё раз, чуть изменив картинку. Опыт лекций показывает, что подобный путь наиболее эффективен. Теперь у нас не обычный замок со скважиной, а кодовый замок (рис. 20).

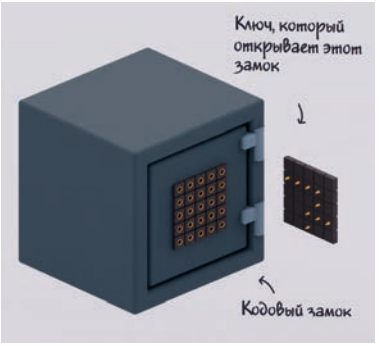


Рис. 20

Сейф с кодовым замком и ключ с уникальной комбинацией, который открывает этот замок

[1] *В терминах биткоина это называется «транзакция». Мы будем очень часто возвращаться к этой концепции, изучая её с разных сторон. Вначале мы будем называть это «ключе-замком», чтобы постоянно напоминать читателю, что это новый термин.*

СМИО ПРЕСС

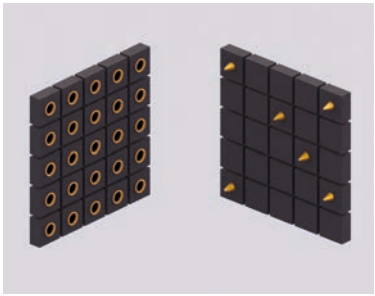


Рис. 21

Пусть Гриша — владелец сейфа. Теперь Гриша хочет передать ценность — то есть сделать так, чтобы сейф открывался тем ключом, который есть у другого человека.

Тот, кому Гриша передаёт ценность, создаёт замок и ключ к нему. Понятно, замок открывается другим ключом, чем у Гриши (другим кодом и набором иголок) (рис. 21).

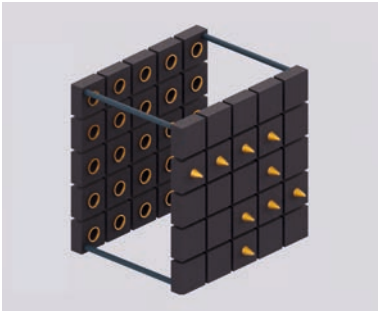


Рис. 22

Теперь Гриша сваривает свой ключ и чужой замок. После этого у него получился новый ключе-замок, который Гриша вставляет в сейф. Теперь сейф открывается ключом человека, который дал Грише замок.

Гриша приварил свой ключ (который открывает сейф) к чужому замку с прошлой картинки (рис. 22).

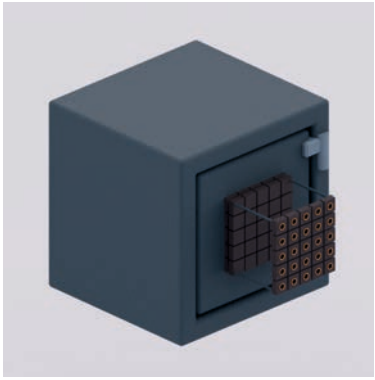


Рис. 23

Важно отметить, что прошлый замок и ключ никуда не пропадают. Фактически, открытия сейфа не происходит. Просто добавляется замок. Теперь у Гриши нет доступа к прошлому замку. Теперь доступен только новый замок (рис. 23).

У вас должен сразу же возникнуть вопрос: «А почему Гриша не может вытащить последний ключе-замок и снова получить доступ к своему замку?». Объяснение, почему нельзя «вытащить» этот ключе-замок, мы пока отложим до следующей главы.

В реальном мире эти ключи и замки не являются чем-то физическим. Это просто нули и единички в компьютере. То есть фактически мы имеем:

0101....0100*01101....1011***101....0110***110....1010***

Здесь, условно, выделенное жирным — это ключ, курсивный текст — это замок. Ключ и замок фактически связаны электронно-цифровой подписью (ЭЦП), поэтому разъединить их невозможно. Обладает ценностью тот, кто имеет ключ к последнему замку ***110....1010***.

Предположим, вы обладатель ключа к последнему замку — ***110....1010***. Когда вам нужно передать ценность, тот, кому вы её передаёте, даёт вам свой

СМИО ПРЕСС

замок. Из его замка и своего ключа вы создаёте ключе-замок и добавляете его в конец этой записи:

0101....0100*01101....1011***101....0110***110....1010***011....11011101....11110**[1]

Все этапы и схемы будут рассмотрены чуть позже — сейчас нам нужно лишь понять идею и схему передачи ценности. Обладание чем-то в наших терминах — это обладание ключом к последнему замку. Передать ценность — значит сделать так, чтобы ключ к последнему замку был у того, кому мы хотим передать эту ценность.

И нужно отметить, что таких сейфов — огромное множество (рис. 24).

Мы в самых общих чертах рассмотрели решение проблемы передачи ценности — мы не передаём саму ценность, мы передаём доступ к ней. Причём нам не нужен удостоверяющий центр — нам просто дают замок, к которому мы прикрепляем ключ. И все видят — сейф теперь открывается другим ключом (каким — никто не знает).

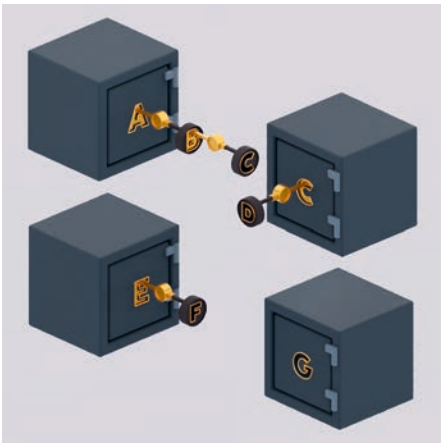


Рис. 24

Сейфов много, у каждого видно разное количество замков. И «открыт» сейф может быть только ключом, который подходит к последнему замку

Вопрос — ответ:

Можно ли разломать наш ключе-замок и «приварить» другой замок?

Нет, это технически невозможно из-за того, что «приварили» мы с помощью ЭЦП, а взломать её нельзя.

Что мешает снять последний ключе-замок и поставить другой?

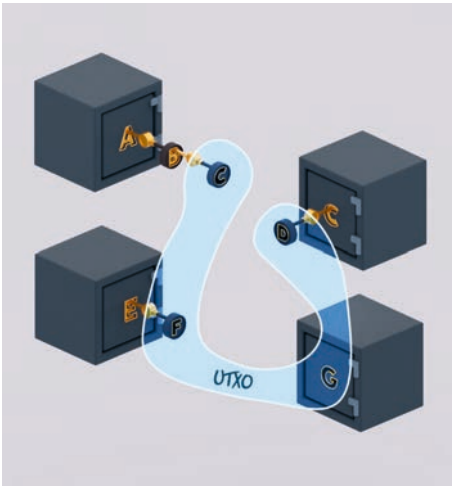
На этот вопрос мы ответим в конце следующей главы. Суть в том, что добавление ключей не может осуществляться каждым человеком. Обладатель ключа лишь заявляет о своём желании и показывает нужный ключе-замок.

Согласовывает новое состояние майнер, который решил сложную задачу. При этом он может только добавлять новые состояния. После чего новое «состояние» распространяется по всей сети. В главе 3 мы подробнее расскажем о распространении нового «состояния» сети. В главе 4 подробнее пройдём путь от создания ключе-замка до его распространения.

---

[1] *Мы обозначили последний замок красным цветом – владелец ключа от этого замка и обладает ценностью. А сам последний замок называется «выходом транзакции» и является частью UTXO, мы очень подробно обсудим это ниже.*

Рис. 25



«Состояние системы» – это список свободных замков

Для этого у нас есть специальный термин — UTXO (unspent transaction output), что обычно переводят как «выход неизрасходованных транзакций». Этот достаточно красивый термин просто обозначает, что эти транзакции «не израсходованы», т. е. это цепочки, в конец которых может быть добавлен код.

Изменение «состояния» (рис. 25) или UTXO — процесс, при котором в цепочку транзакции (её выход) добавляется код. Его добавить имеет право только майнер (иначе его не примут другие узлы). Как это происходит, подробнее рассмотрим далее.

Самостоятельно «добавить» или «снять» замок у себя на компьютере можно, ведь это просто нули и единицы в памяти компьютера, но распространить такую версию среди других узлов уже нельзя. Этому будет посвящена глава о майнинге. Другими словами, «снять» последнее состояние (ключе-замок) невозможно, т. к. это потребует изменения всех копий записей на всех узлах сети. А они запрограммированы только на добавление ключей[1].

Ещё раз обсудим, как происходит передача ценности. Есть «текущее состояние», которое можно представить как улицу с сейфами. В каждом сейфе лежит какая-то ценность. В каждом сейфе есть замок. Мы говорим «мне принадлежит ценность», если у нас есть ключ от замка на сейфе.

Чтобы передать ценность, мы просим замок у того, кому хотим её передать. Он даёт замок, мы привариваем к нему ключ и получаем «ключе-замок». Нужно помнить, что у нас есть просто последовательность нулей и единиц, а «приваривание» — это цифровая подпись, которую невозможно взломать. Защита от «вытаскивания» ключа из замка или «проблема двойных трат» будет рассмотрена в следующих главах.

Нужно отметить, что биткоин из-за отсутствия удостоверяющего центра сильно усложняет передачу денег. Поэтому он нужен только при отсутствии доверия между сторонами сделки.

В Российской Федерации нет проблемы доверия к традиционным финансовым институтам, однако нужно помнить, что мировая финансовая

[1] *Это не совсем так: злоумышленник, обладающий огромным запасом мощности, может изменить последние записи сети, мы это разбираем в главе 12 об атаке 51%.*

СМИО ПРЕСС

система в большой степени основана на долларе и неудивительно, что сам биткоин возник в 2008 году во время масштабного кризиса и падения экономики США. Биткоин не может заменить внутреннюю валюту стран, однако может заменить доллар США там, где он используется вне США. При торговле между странами, или если в стране отсутствует качественная финансовая система, биткоин выступает наилучшим платёжным решением.

К примеру, Эквадор, у которого фактически отсутствует собственная валюта и где граждане пользуются долларами в расчётах между собой, ввёл биткоины как обязательное платёжное средство именно в попытке заменить доллары.

СМИО
ПРЕСС

Глава 3

ОТСУТСТВИЕ ДОВЕРИЯ МЕЖДУ СТОРОНАМИ ПРИ ПЕРЕДАЧЕ ИНФОРМАЦИИ. ЗАДАЧА ВИЗАНТИЙСКИХ ГЕНЕРАЛОВ

В прошлой главе мы обсудили систему передачи ценности. Но на самом деле задача сложнее. Нам нужно не только навесить чужой замок, но и уведомить всех об этом. При этом мы не знаем, кого уведомлять, и не знаем, как уведомлять.

Сама возможность передачи важной информации, когда ни способу передачи, ни собеседнику нельзя доверять, кажется абсурдной и была сформулирована не так давно. В криптографии она красиво названа «задачей византийских генералов».

В общем случае (есть другие частные определения), есть неизвестное число генералов, которые изолированно друг от друга управляют своими отрядами. Какое-то число генералов может являться предателями. Кто именно предатель — неизвестно. Общение между ними осуществляется гонцами, при этом нельзя проверить, от кого пришёл гонец. Генералам нужно договориться об атаке. Если атакуют все вместе — они победят противника. Если по отдельности — противник их разобьёт. Если они не атакуют (отступят), то просто сохранят армию (приемлемый вариант).

Задача кажется невыполнимой и разрешимой только для частных случаев (к примеру, если канал связи абсолютно надёжен, всего генералов 4, а предатель только 1). Однако в экосистеме биткоина Сатоши Накамото смог соединить большое число интересных идей (см. общую схему, с. 6), чтобы решить эту задачу для любого числа генералов (узлов сети), число которых может динамически меняться. При этом канал связи может быть любым. Злоумышленник может его прослушивать и даже иметь возможность подмены данных.

Это нужно подчеркнуть: когда вы пользуетесь традиционными финансовыми институтами, вам необходима защита и шифрование канала. Когда вы передаёте в банк информацию о том, что хотите оплатить в магазине товар, вся информация шифруется и требует определённых каналов связи.

Экосистема биткоина позволяет передавать информацию через любые системы связи (интернет, коротковолновая связь, смс, бумагами с курье-

СМИО ПРЕСС

ром и т. д.). Злоумышленник может читать и видеть весь канал связи и всю переписку (её видит любой узел сети) и даже отправлять неверные данные в сеть. Вы можете переписать код узла или «ноды» сети, чтобы она, к примеру, передавала информацию о том, что все деньги отправляются вам. Это не поможет вам завладеть деньгами.

Если до создания системы биткоина задача византийских генералов считалась неразрешимой, то создатель биткоина смог решить даже более общую задачу. По сути, можно говорить о том, что в лесах прячутся не отряды с генералами, а просто много толп несвязанных между собой людей, которых никто не назначал, которые совершенно не знакомы и не доверяют друг другу. Более того, среди этих людей совершенно точно существуют предатели, которые не хотят, чтобы люди договорились.

О технической стороне мы будем говорить во второй части. Здесь мы просто попытаемся объяснить саму идею защиты.

Итак, вы являетесь сторонником Византии и вместе с другими гражданами должны договориться о едином времени атаки.

Все неверные сообщения вы должны отвергнуть. Но как же достигается защита от «предателей-генералов», которые изменят время атаки в последний момент (или от мошенников, которые хотят отменить свой перевод вам после получения услуги)?

Это достигается с помощью затрат энергии на расчёт сложной математической задачи. Представьте себе задачу, которую сложно решить (к примеру, огромная формула с неизвестным *X*), при этом задачу нельзя решить аналитически, а только подбором разных *X*. Но уже найдя *X*, ответ легко можно проверить, просто подставив ответ вместо *X*. Конечно, на практике всё сложнее, и мы поговорим об этом в главе 8 о хэшировании.

В терминах задачи о византийских генералах можно представить, что все генералы сажают своих солдат решать сложную математическую задачу и после её решения отправляют гонцов друг к другу, соединив это решение со временем атаки. Генералы (и отдельные ополченцы, и просто сторонники) подчиняются тому решению, которое придёт первым. Предполагается, что чем больше солдат её решают, тем быстрее кто-то из них её решит (умственные способности солдат не играют большой роли, т. к. задача решается перебором, и это важно).

Каждый солдат может как начать решать задачу (в терминах биткоина он будет являться майнером), так и просто транслировать полученные данные (в терминах биткоина он будет являться узлом сети).

Красным гребнем мы обозначим честных генералов. И у каждого свой размер отряда. С чёрным гребнем

Рис. 26

(предателей) заведомо меньше, но они могут высылать тоже огромное количество гонцов. Каждый посылает гонца к каждому (просто тут стрелок получается слишком много) (рис. 26).

К каждому генералу может прийти десятки сообщений от других генералов — и он не может понять, какие от предателей, а какие от честных. Одни требуют стоять на месте, другие атаковать (в разное время), третьи — отступать. Какому сообщению следовать?

И наша идея в том, что все генералы сажают своих солдат за подбор решения к задаче, которая потребует перебора тысяч (или миллионов, может, и миллиардов) вариантов, чтобы найти её решение. При этом время атаки будет в составе условия.

Таким образом у нас есть функция от двух аргументов: f(x,время атаки) = y.

Заранее известны функция и нужный результат *y*. Время атаки генерал ставит какое хочет, а *x* генералы приказывают подобрать. Такой вид задачи (чтобы её нельзя было решить аналитически) мы подробно разберём в главе 8 о хэшировании, сейчас просто представим, что она есть. Тысячи солдат подбирают *x*, чтобы функция имела решение, и когда первый из них найдёт такой *x*, генерал рассылает всем другим генералам гонцов.

*Если описание через функцию непонятно, можем добавить очередной уровень абстракции: представьте, что есть волшебная бумага (её много, она у каждого солдата), и после того как время атаки на ней написано, необходимо дописать к нему текст из 5 букв (т. е. вариантов — 33*33*33*33*33 и только 10 из вариантов будут верными). Они случайны и неизвестны заранее. Когда верный вариант найден, бумага начинает светиться.*

СМИО ПРЕСС

Рис. 27

Так как честных генералов больше, и солдат у них больше, то и решение найдёт кто-то из их солдат с гораздо большей вероятностью. Как только решение найдено, гонцы бегут к другим генералам и представляют найденный *x* и время атаки. Генералы следуют этому совету, а также рассылают на всякий случай гонцов к другим генералам (рис. 27).

Предатели могут поймать кого-то из гонцов, но их можно высылать неоднократно и по многу раз. А если кто-то из генералов получил сообщение, он тоже начинает рассылать гонцов.

Ещё раз напомним, что у нас компьютерная сеть, и «гонцы» в нашей ситуации — это просто сигналы другим узлам сети. По сути, их бесконечное количество. А если сеть устроена так, что «каждый связан с каждым», то сигнал нельзя остановить.

Также напомним, что никто не знает, сколько и где существует генералов, отдельных отрядов, отдельных повстанцев и так далее. Их никто не назначает, они не знают и не доверяют друг другу. Единственное, что все знают, — если задача решена, то этому решению можно доверять.

Между тем предатели, поймав гонца, ничего не могут сделать с сообщением. Они не могут его подменить, ведь если вписать другое время атаки, *x* не подойдёт. Они не могут составить разным генералам разное время атаки. А если предатель получит какое-то решение *x*, то он не сможет даже им воспользоваться, т. к. если он его вышлет, то другие генералы начнут высылать его решение с его временем атаки — и атака всё равно будет в одно и тоже время.

Единственный шанс у предателей сорвать план — это одновременно отослать разным генералам разное время атаки. Но чтобы получить несколько решений за то время, пока честные генералы не нашли и одного, предателей должно быть не просто большинство, их должно быть гораздо больше, чем честных генералов.

СМИО ПРЕСС

Из вышесказанного необходимо сделать несколько выводов.

Вывод 1. Сама система взаимодействия между генералами не касается управления войсками внутри подразделения. Даже теоретический взлом сети никак не повлияет на тех солдат, которые непосредственно в подчинении у генералов.

Вывод 2. Если злоумышленников окажется больше половины, то сеть может начать работать по их правилам. То есть, если у мятежных генералов окажется больше солдат, то они смогут (теоретически) сначала заставить атаковать в одно время, а потом в другое. Это называется «атака 51%», и защита от неё состоит в следующем: невозможно, чтобы злоумышленников (которые должны действовать сообща) оказалось больше, чем честных узлов сети (как не может быть в армии больше предателей, чем обычных солдат).

А если произошло такое невероятное событие, что у генералов-преда телей больше солдат, чем у всех других, то они могут захватить власть в Византии и им незачем проигрывать войну.

Вывод 3. Вся система контроля требует серьёзных затрат энергии. В случае, если у нас есть удостоверяющий центр и надёжные каналы связи, мы просто не захотим использовать такую систему. Генералам не будет смысла давать солдатам задачу, если канал связи надёжен, а предателей нет (или их можно вычислить).

И армия, которой приходится пользоваться такими сложными методами управления, будет, разумеется, в худшей позиции, чем армия с надёжными каналами связи и единым управлением.

Возвращаясь к прошлой главе, нужно отметить, что, в отличие от византийских генералов, нам необходимо согласовать не время атаки, а присоединение ключе-замков к замкам. Есть список ключе-замков, каждый генерал (майнер) выбирает определённый список, решает задачу и рассылает новое состояние всех сейфов. Подробнее об этом мы расскажем в следующей главе.

СМИО
ПРЕСС

Глава 4
А ПРИ ЧЁМ ТУТ БИТКОИН

В данном разделе мы описали большое количество аналогий и ситуаций, которые невозможны в реальной жизни. Мы просто попытались подвести вас к пониманию сути и системы блокчейна, и криптовалют.

Мы надеемся, что вы поняли все аналогии. Опираясь на них, мы можем объяснить общую концепцию экосистемы биткоина.

Передача ценности

В биткоине отсутствует реестр владельцев. В отличие от традиционных финансовых институтов, когда перевод денег означает списание в одном месте и зачисление в другом (часто другой суммы), биткоин возвращается именно к идее передачи ценности.

Обладать ценностью — значит иметь к ней доступ. Схематично это можно изобразить, как обладание кодом или ключом от сейфа. Передать ценность означает закрыть сейф на чужой замок.

Уведомление всех о передаче вами ценности

Если в обычной системе вам нужно уведомить лишь банк (центр) о передаче ценности, то теперь банка нет, а вам нужно уведомить об этом всех. При этом есть риск, что одним вы скажете, что передали ценность одному человеку, а другим — другому (проблема двойных трат). Нужен способ определить, кому именно вы передали деньги.

Схематично это можно изобразить как огромную улицу сейфов, коды (ключи) к которым есть неизвестно у кого, и при передаче ценности каждый может подойти и повесить на сейф новый ключе-замок. Улица сейфов не имеет физического выражения, может быть скопирована и изменена любым человеком, а потому есть необходимость согласования, какая из копий является верной.

Проблема византийских генералов и согласование копии

Здесь мы перешли к проблеме византийских генералов, которые должны согласовать единое время атаки, как сообщество биткоин согласовывает правильную версию «улицы сейфов». В примере про генералов мы использовали аналогию решения задачи. Солдаты объединяются в группы (или могут делать это в одиночку) и начинают решать очень сложную задачу подбором.

В нашей аналогии они на специальной «волшебной» бумаге пишут время атаки + случайный код. И нужно, чтобы бумага «засветилась волшебным цветом». Каждому времени соответствует свой код, который невозможно знать заранее. В сообществе биткоин это соответствует взятию хэша из данных о передаче ключей + код, который подбирается случайным образом.

Очень подробно о функциях хэширования мы будем говорить в главе 8.

Поиск кода называется майнингом. Нашедший правильный код вознаграждается. Понятно, что группа солдат, работая сообща, быстрее найдёт код, чем одиночка. Но и вознаграждение будет распределено между ними поровну. Одиночки в терминах биткоина называются майнерами. Сообщества, работающие сообща, — майнинг-пулами.

После того как решение найдено, тому, кто его получил, необходимо максимально быстро сообщить об этом всем. Они посылают гонцов (или в нашей системе — сигналы) по всем уголкам сети. Каждый может проверить решение быстро и легко, но не может его изменить, ведь если написать на бумаге что-то другое, то света не будет. Или, в нашей аналогии, если добавить этот же код к другой «улице сейфов», то решение будет неверным.

Разница между майнером и узлом сети

Ещё раз подчеркнём разницу: тот, кто ищет решение задачи (ищет время атаки), является майнером. Тот, кто просто получает и ретранслирует информацию о найденном решении, является узлом сети. У майнера всегда должен быть узел (иначе никто не узнает о его решении), но узел может не быть майнером.

Майнер получает вознаграждение. Узел ничего не получает. Он поддерживает сеть, а также может проверить правильность решения.

Продемонстрируем сказанное серией картинок. Представим, что всё началось с одного сейфа, который был создан когда-то Сатоши Накамото. Это единственный сейф в истории биткоина, который не требовал работы для создания.

Блок, который создал этот сейф, назван генезис-блоком и был создан 3 января 2009 г. В него был вписан заголовок газеты The Times:

The Times 03/Jan/2009 Chancellor on brink of second bailout for banks.

Возьмём нашу старую картинку (с. 22, рис. 16). Создан первый сейф.

Теперь Женя хочет получить эти деньги (в наших терминах — поставить на сейфе свой замок). Как вы помните из предыдущей главы, она даёт свой замок тому, кто создал этот сейф и у кого есть ключ.

СМИО ПРЕСС

Владелец текущего замка сваривает свой ключ и Женин замок вместе — и теперь ему надо согласовать замену замка (вставку ключа) (рис. 28).

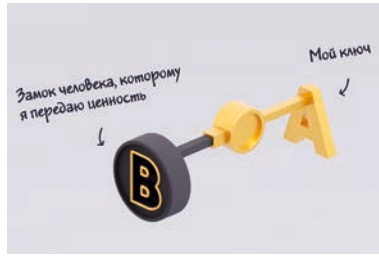


Рис. 28

Суть блокчейна в том, что другие узлы примут замену замка (вставку ключе-замка), только если будет проведена определённая работа — как в прошлой главе о византийских генералах.

Майнер забирает ключ и находит «волшебное число». Помните, как в прошлой главе генералы должны были найти ко времени атаки «волшебное число», чтобы «бумага засветилась»? Здесь мы делаем то же самое: наш ключе-замок — это всего лишь последовательность 0 и 1, как и всё в компьютерном мире. Нам нужно найти к нему «волшебное число».

Рис. 29

Создатель ключе-замка «кладёт» его на общий стол майнеров. В терминах биткоина это место называется «мемпулом»

Суть этого числа и как оно считается — мы обсудим в главе 8 о хэшировании. Сейчас просто можем представить, что к любому ключе-замку есть «волшебное число». И его приходится искать подбором.

Таким образом, майнер берёт ключе-замок и простым перебором ищет к нему «волшебное число». Как только он его найдёт, он пересылает всем узлам сети своё решение. Мы подробно об этом говорили в прошлой главе — даже наличие злоумышленников не предотвратит распространение этой информации.

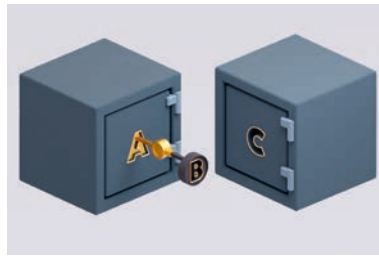


Рис. 30

Новое состояние

Согласно правилам сети, тот, кто нашёл «волшебное число», получает право создать сейф со своим замком «из ниоткуда», и в итоге получается наше новое состояние (или новый блок) — это два сейфа. В первом вставлен ключ, а второй — с замком от майнера.

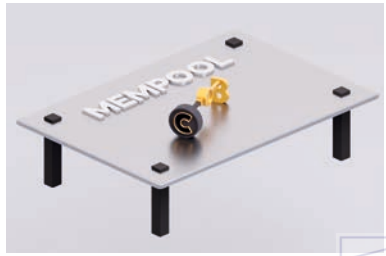


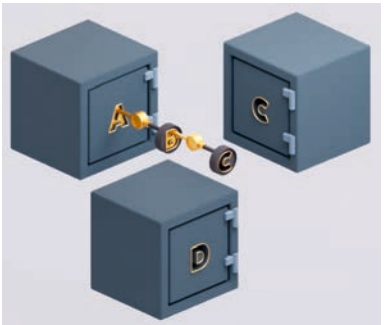
Рис. 31

Новый мемпул

Теперь представим, что владелец замка 1 тоже хочет передать ценность. Он берёт замок от того, кому хочет её передать, создаёт ключе-замок и тоже кладёт его в мемпул.

СМИО ПРЕСС

Рис. 32

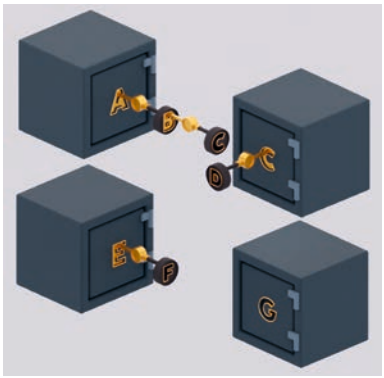


Теперь сейф 1 открывается тем же самым ключом, что и сейф 2. Это не противоречит логике. Получается, что владелец ключа от замка C владеет ценностью в первом и втором сейфах

Рис. 33

Теперь задача майнера – найти «волшебное число» сразу к двум ключам

Рис. 34



Сейф 1 без изменений, у сейфов 2 и 3 новые замки и появился сейф 4

Другой майнер находит к ключу «волшебное число» и распространяет новое состояние.

В первом сейфе уже два ключе-замка, второй сейф остался прежним и добавился третий сейф.

Теперь на следующем этапе владельцы замков к сейфам 2 и 3 хотят передать ценность. Они создают ключи и кладут их в мемпул.

Мы пока опустим до главы 8 конкретный способ. Отметим только несколько новых фактов:

1. Нахождение «волшебного числа» для одного ключе-замка или для ста ключе-замков — задача примерно одинаковой сложности. Поэтому майнерам неважно количество ключе-замков.

2. Майнер может получать доход не только от права на создание нового сейфа, но и «благодарность» от создателей ключе-замков (комиссия майнеру) — мы подробно рассмотрим этот механизм позже.

3. Теперь мы можем объяснить смысл фразы «невозможно вытащить ключ и заменить другим ключе-замком»: нам нужно считать для него другое «волшебное число» (повторить работу майнера), а другие узлы не примут нашу версию, если у них уже есть прошлая. Более подробно о согласовании мы будем говорить в главе 16 о двойных тратах.

Итак, майнер находит подпись к двум ключам сразу — и получается третье состояние (рис. 34).

Возникновение нового состояния. UTXO

После того как узел получает решение задачи и проверяет его правильность — возникает новая реальность. Теперь для узла появилась новая «улица сейфов», и он её

СМИО ПРЕСС

считает правильной. Совокупность всех состояний всех сейфов (по сути, всех последних замков) называется UTXO и ему будет посвящено много информации далее.

Схематично можно показать так (рис. 35). Мы обладаем ключом к одному из замков и хотим передать его другому. Нам дают замок, мы соединяем его с нашим ключом и распространяем для всех информацию, что в замок вставлен наш ключе-замок и теперь сейф закрыт на другой замок.

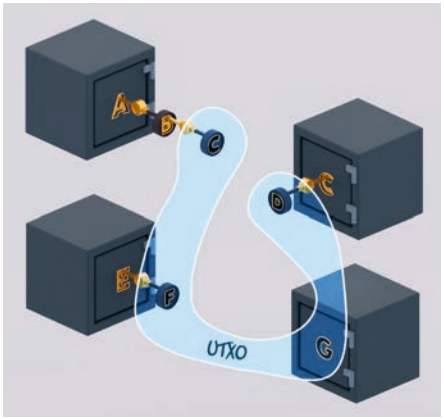


Рис. 35

Несколько других владельцев ключей также создают такие «ключе-замки» или в терминах биткоина «транзакции» и распространяют информацию о том, что обновляют замки в сейфах.

Этот список ещё несогласованных транзакций называется мемпулом. Мы расскажем об этом подробнее в главе 16.

Повторим. Майнер берёт из мемпула данные — кто и куда хочет перевести деньги, создаёт новую реальность — новую улицу сейфов, «подписывая» эту улицу с помощью решения задачи. Список новых использованных ключе-замков — это как время атаки из задачи про византийских генералов. Подробно система майнинга рассмотрена в главе 19.

Создав новую реальность и новую «улицу сейфов», точнее те же сейфы, но с новыми замками, в награду за выполненную работу майнер из ниоткуда получает сейф, который закрыт на замок, ключ которого находится у него.

Совокупность всех свободных замков (не только тех, что получились только что, а вообще всех свободных замков) называется UTXO.

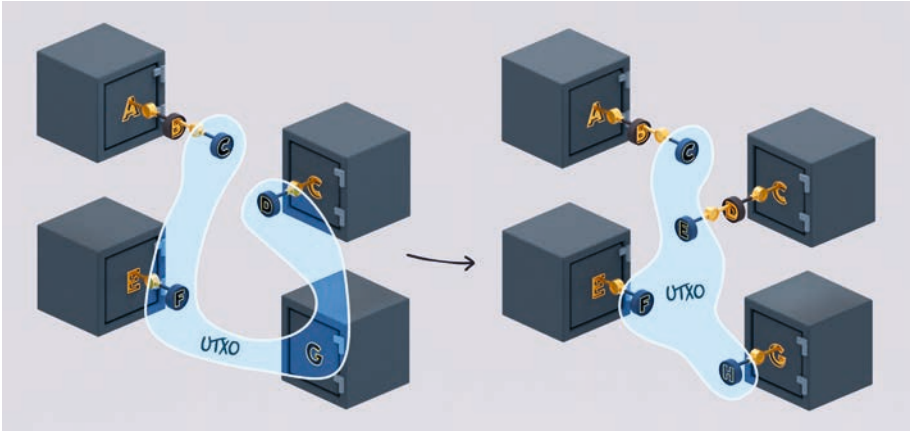
Теперь мы подошли к объяснению, почему тот, кто вставил новый ключе-замок в свой замок, не может вытащить его обратно. Потому что в этом случае он должен будет создать новую «улицу сейфов», а это возможно только при выполнении работы майнером. Новую транзакцию не примут, потому что в новом UTXO просто некуда вставить его ключе-замок. Тот замок, куда вставлялся ключ, уже занят. Подробнее об этом мы расскажем в главах 14 и 18.

Давайте ещё раз рассмотрим, как ключе-замки (транзакции) принимаются узлами сети. Как мы отметили, в любой момент времени состояние системы определяется замками, в которые можно вставить ключи. Совокупность этих замков (или выходов транзакций) называется UTXO или «состояние системы».

Право изменить эту систему получает майнер, который решит сложную математическую задачу (подробно в главе о майнерах). Он выбирает ключе-замки (транзакции) из мемпула (общего стола, куда все желающие могут по-

СМИО ПРЕСС

Рис. 36



ложить транзакции) и создаёт новое состояние, «вставляя» выбранные ключе-замки в свободные замки, которые для них предназначались.

Сам майнер не может создать ключе-замки, т. к. для их создания нужно обладать соответствующим приватным ключом.

После того, как новое состояние сделано, оно распространяется между узлами (рис. 36).

После того как узлы проверят решение задачи, а также, что ключе-замки действительно подходят к своим замкам, они примут новое состояние. И теперь «вытащить» ключе-замки уже невозможно просто потому, что узлы уже не примут это изменение.

Сделаем несколько выводов:

Вывод 1. Никто не сможет завладеть вашими биткоинами, даже если взломает сеть или захватит контроль над всеми нодами. Как у генерала никто не захватит контроль над его солдатами.

Вывод 2. Систему можно сломать, если в ней злоумышленников, работающих сообща, будет больше, чем честных узлов.

В терминах биткоина это будет означать возможность «отмены» транзакции злоумышленником. Однако «честность» в системе биткоина вознаграждается, а злоумышленник не получает ничего.

Если злоумышленников оказалось больше, то сеть попадает под их контроль со всеми преимуществами (возможность собирать все транзакции) и им становится невыгодно ломать сеть, которую они захватили. В ситуации с византийскими генералами это означает, что генералам проще захватить власть, чем проигрывать войну.

Вывод 3. Блокчейн биткоина всегда будет проигрывать по всем параметрам традиционной финансовой системе из-за необходимости поддержки такой инфраструктуры. То есть нет никакого смысла создавать «частный» блокчейн с ограничениями. Проще создать защищённую базу данных.

СМИО ПРЕСС

Смерть биткоина и других криптовалют будет неизбежна, если возникнет централизованная валюта, которой можно будет доверять. Если, к примеру, по всему миру можно будет легко и за секунды открывать счёт в условных рублях, блокировка счёта в них будет немыслима, а переводы окажутся дешёвыми, то биткоин потеряет смысл и не будет востребован.

Вывод 4. Блокчейн можно использовать только в криптовалютах. Совершенно бессмысленно говорить об использовании блокчейна и о распределённой системе принятия решений, если нет огромного количества энергии для её защиты. Любой разговор о том, что блокчейн можно использовать для контроля за сделками с недвижимостью или любым другим реестром, можно вести только если:

1) сама ценность должна находиться только внутри этой экосистемы. Если вы записываете в блокчейн своё право на владение недвижимостью, то при захвате этой недвижимости вы сможете защититься, только обратившись в суд, а значит, никаких преимуществ записи в блокчейне вы не получаете. Всё равно законы государства будут выше этой записи. Наоборот, если запись в блокчейне означает однозначную потерю вами права на владение, то злоумышленник, украв ваш цифровой код, сможет перевести недвижимость на себя, и вы не сможете её вернуть.

Проблема электронной регистрации сделок с реальной собственностью даже без блокчейна стоит настолько остро, что в 2019 г. Росреестр Российской Федерации запретил проведение сделок с недвижимостью через электронно-цифровую подпись;

2) защититься от «отмены» сделки можно только при огромных затратах на энергию. Тот, кто будет тратить эту энергию, должен получать деньги большие, чем мог бы получить, играя роль злоумышленника, что невозможно, если у вас «частный» распределённый реестр. Если условный Ситибанк создаст внутренний блокчейн без доступа к нему внешних источников, эта система будет проигрывать обычной базе данных из-за затрат на поддержку. А если даст доступ ко всеобщей записи транзакций любому желающему, то придётся создавать огромную систему, энергия которой будет больше, чем энергия, которую может затратить предполагаемый злоумышленник;

3) биткоин имеет смысл использовать только в случае, когда мы не можем (или не хотим) доверять ни контрагенту, ни единому центру (или этот центр вообще отсутствует). Или, если Центр запрещает вам переводить деньги или берёт слишком большие комиссии. Как уже говорилось, в Российской Федерации это не имеет смысла при расчётах внутри страны, т. к. при неисполнении обязательств можно подать в суд, а затраты на перевод в рублях небольшие. Но если речь идёт о трансграничных переводах, или, если очень большие комиссии, к примеру, если США наложили на вашу страну санкции, а вам нужно купить товар из Китая, это может иметь смысл.

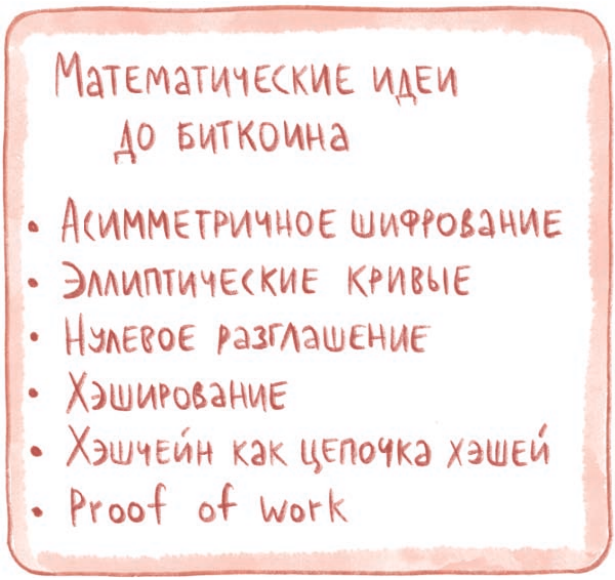
СМИО
ПРЕСС

Часть 2

МАТЕМАТИЧЕСКАЯ ОСНОВА

В прошлой части мы описывали идеи и концепции. Практически всегда мы пытались представить аналогии из реального мира, чтобы идею было проще понять.

Здесь мы описываем именно техническую реализацию. Эта глава объясняет, как именно работают те модели, которые мы описывали в первой части. Также нужно отметить, что все эти концепции были созданы до биткоина. Они являются стандартными математическими идеями, изучить которые можно в десятках и сотнях других книг. Для упрощения понимания, авторами книги был создан сайт crypto.239.ru, где вы можете самостоятельно проверить и лучше понять все изложенные здесь идеи.



СМИО
ПРЕСС

Глава 5
ВСТУПЛЕНИЕ В ТЕХНИЧЕСКУЮ ЧАСТЬ

В этой главе мы рассматриваем «красную» часть нашей общей схемы (см. с. 6). Мы опишем математические модели и решения, которые были изобретены до биткоина и с совершенно другими целями. Эти модели и решения использовались для создания биткоина. Напомним, что биткоин был создан в 2008 году (возможно, на несколько лет раньше).

Каждый из этих элементов был прорывом в решении определённого круга задач.

Элемент	Кем и когда создан	Цель
Асимметричное шифрование	Уитфилд Диффи и Мартин Хеллман, 1976 г.	Возможность передать ключ для зашифровки сообщения без возможности расшифровки самого сообщения, т. е. возможность зашифровать сообщение без возможности его расшифровки
Хэширование. Создание кода для любого текста	Идеология — 1953 г., первые примеры — 1980-е гг.	Огромное число целей: доказательство, что ты первый решил задачу, что ты скачиваешь правильную программу и т. д.

Мы будем очень подробно изучать и асимметричное шифрование, и хэширование, этому посвящены целые главы. Но прежде всего нужно понять, что между этими способами преобразования текстов нет ничего общего. При шифровании текста вы создаёте новый текст, который может быть потом обратно преобразован в исходный.

При хэшировании вы создаёте небольшой отпечаток текста. Как снятие отпечатка пальца у человека. Теоретически, отпечаток пальца может быть одинаковым у разных людей. И по отпечатку пальца невозможно восстановить человека. Так же из хэша невозможно восстановить изначальный текст, а вероятность совпадения хэшей у двух текстов исчезающе мала.

СМИО ПРЕСС

Элемент	Кем и когда создан	Цель
Электронная подпись	В 1976 г. предложено само понятие. Начало использования — конец 1980-х гг.	Обмен информацией с доказательством того, что текст создал определённый источник. Возможность расшифровать сообщение без возможности его зашифровать
Proof of Work	1997 г. — идея в системе Hashcash. В 1999 г. появляется и сам термин Proof of Work	Решение проблемы со спамом
Доказательство того, что у нас есть деньги (доказательство с нулевым разглашением)	Идея была сформирована ещё в древности. Современная система создавалась в 1986–1989 гг.	Передача паролей, доказательство того, что у нас есть информация, достоверность

СМИО
ПРЕСС

Глава 6

СИММЕТРИЧНОЕ И АСИММЕТРИЧНОЕ ШИФРОВАНИЕ

В этой главе мы хотим рассказать о шифровании, различии между симметричным и асимметричным шифрованием, представить вам программу и на конкретном примере показать, как шифруются и расшифровываются данные.

Первоочередная проблема любой передачи данных — это шифрование. Любая попытка передать секретную информацию (а финансовая информация секретна по определению) наталкивается на необходимость её шифровки.

Шифры известны столько времени, сколько существует письменность. С начала истории цивилизации люди использовали шифры. Их великое множество.

К примеру, шифром Цезаря вы можете зашифровать любое сообщение, заменяя букву другой, стоящей от неё в алфавите на определённом расстоянии. Например, *А* превращается в *Г*, если это расстояние 3 символа.

Соответственно, слово НАСТЯ превратится в РГФХВ (рис. 37).

Или другой, наверное, наиболее известный шифр — это замена символов на картинки (или на не связанные с ними другие символы). К примеру, в рассказе «Пляшущие человечки» Артура Конана Дойля Шерлок Холмс разгадывал такой шифр (рис. 38). Люди писали их на заборах или стенах домов, а не знающие тайны прохожие считали картинки с человечками просто играми ребят. Это, наверное, самый слабый из возможных шифров. Его легко раскрыть по словарям, частоте использования слов и так далее.

Рис. 37

Рис. 38

Такой шифр можно серьёзно улучшить, если намеренно писать в исходном тексте слова с грамматическими ошибками. Во время Великой Отечественной войны партизаны чаще всего передавали шифровку с ошибками, т. к. текст вида «гоммандер прекозал собротся у озира» будет расшифрован по словарю с гораздо большими трудностями, чем «командир приказал собраться у озера».

Тема шифров неисчерпаема. К примеру, прочитав шифрованное послание одного из преступников (здесь мы предлагаем свой перевод): «Житель Лондона, Хадсон, старый охотник, сдал щётку. Для вас цена небольшая. Советую купить и срочно помочь собаке линять»[1], — Ватсон ничего не понял, а Холмс сразу догадался, что тут просто надо читать каждое третье слово.

Все эти шифры являются примерами так называемых *симметричных шифров*, т. е. шифров, где для шифрования и дешифровки применяется один и тот же алгоритм, причём трудоёмкость его применения для шифровки и дешифровки одинакова.

Рис. 39

Трёхроторная машина «Энигма». По современной классификации её сложность составляла от 76 до 308 бит. Её также успешно взломали

Ещё во время Второй мировой войны единственным способом шифровки были симметричные шифры. Самой известной машиной шифровки была «Энигма». По сути, это было серьёзное улучшение того же шифра Цезаря, когда символ сдвигался не на одно и то же число знаков, а в зависимости от множества факторов. Сдвиг определялся начальным положением вращающихся роторов, часто новым для каждого сообщения.

Современные средства коммуникации также используют симметричные шифры. Обычно это просто усложнённые перестановки, когда сообщение пишется по строкам, а передаётся по столбцам. Так сообщение:

Н А С Т
Я Е С Т
К А Ш У

будет передано как НЯКАЕССШУТТ. Понятно, что это простейший пример, есть более сложные способы, такие, как «магический квадрат» и прочие.

Симметричные шифры могут обладать серьёзной криптостойкостью и, в связи с простотой и лёгкостью как шифровки, так и дешифровки, часто используются и сейчас. Самое главное для таких шифров — размер ключа шифрования (по сути, сложность преобразования) и количество «проходов», когда мы ещё несколько раз шифруем полученную строку.

---

[1] *В классическом переводе «в лоб»: «С дичью дело, мы полагаем, закончено. Глава предприятия Хадсон, по сведениям, рассказал о мухобойках всё. Фазаньих курочек берегитесь».*

СМИО ПРЕСС

Понятно, что шифр можно сделать очень сложным и его расшифровка может потребовать огромных вычислительных мощностей. Однако при увеличении сложности шифрования мы должны использовать всё большее количество энергии. Но это не главная проблема.

У симметричного шифрования есть два принципиальных недостатка, которые не позволяют его использовать не только в качестве шифра при передаче ценности, но и при более простом случае обыкновенной Электронной Цифровой Подписи.

Первая проблема, с которой мы сталкиваемся, — невозможность передать код шифровки через тот же канал связи. Код можно передать только в серьёзно зашифрованном или недоступном потенциальному противнику канале, чаще всего только при личной встрече.

Вторая проблема — если вы можете что-то зашифровать, значит, вы можете и расшифровать. И это может сделать любой человек, который знает способ шифровки.

Нужно подчеркнуть, что проблема симметричного шифрования не в сложности взлома — она может быть запредельной. Необходимость асимметричной расшифровки кроется именно в проблеме передачи ключа. Также асимметричное шифрование позволяет не дать возможности расшифровать сообщение тому, кто его зашифровал.

Возникла необходимость создания такого шифра, когда мы можем раздать код для шифровки, но расшифровка по этому коду невозможна.

Асимметричное шифрование

Решению этих проблем и посвящено асимметричное шифрование. Первые попытки создать систему, при которой шифровка сообщения происходит иным способом, чем расшифровка, относятся к началу 1970-х гг.

Изначально идея кажется невыполнимой. Ведь если мы знаем как зашифровать, значит, мы знаем и как расшифровать. И это кажется очевидным. Это одна из первых идей, положенная в основу криптовалют, которая сначала казалась невыполнимой.

Представим задачу и её исполнение следующим образом. У нас есть «шифрующий» робот. Если мы на вход подаём какую-то информацию, к примеру, слово «ШИФРОВАНИЕ»,

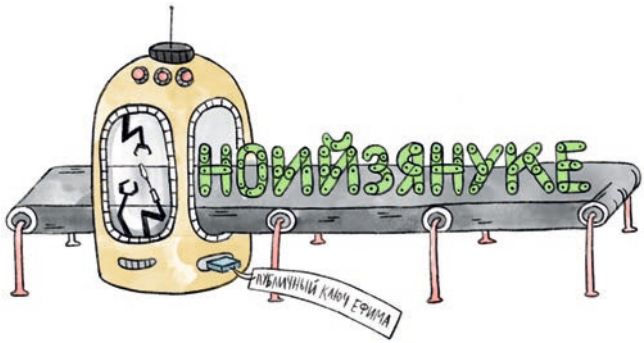
Рис. 40

то, постепенно пересобирая это слово:

Рис. 41

в конце концов мы получаем:

Рис. 42



Робот умеет пересобирать слова по определённым законам. К примеру, «взять деталь за элемент номер *X*, открутить болт номер *Y*, взять получившуюся деталь, открутить *V*, от неё открутить болт номер *J*, прикрутить болт *G*, открутить болт номер *Y*» и тому подобное. Все эти данные — *X Y V J G* записаны и называются «публичным ключом». Робот общеизвестен, но имея данного робота мы можем только зашифровать, а расшифровать — уже нет. Представьте, что данные настолько сложны, что обратная пересборка с этими данными невозможна.

Но если у нас есть так называемый «приватный ключ», то мы можем запустить совершенно другого (но тоже общеизвестного) робота приватных ключей, и он будет совершать обратную операцию:

Рис. 43

СМИО ПРЕСС

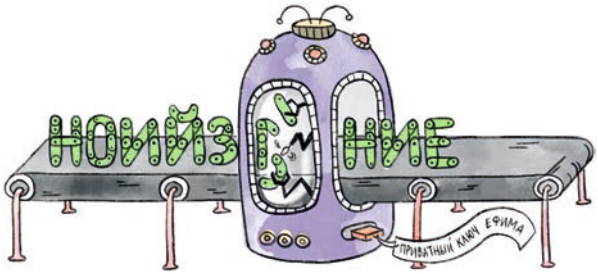


Рис. 44

И имея совершенно другие данные — какой брать болт, как откручивать и как переставлять, и совершенно другое устройство (рис.44), робот пересобирает получившийся код в исходные данные (рис. 45):

Рис. 45

Ещё раз давайте обсудим очень важные моменты:

1. Роботы для пересборки совершенно разные. Робот для шифровки не может расшифровывать и наоборот.
2. В роботов можно загружать совершенно разные данные, но приватный ключ должен быть парой к публичному ключу, т. е. они должны как бы дополнять друг друга. И один публичный ключ не подойдёт к иному приватному ключу — болты будут браться совершенно другие и исходные буквы не получатся.
3. Без пересборки невозможно понять, что именно шифровалось.

Ещё раз отметим, асимметричное шифрование не более криптостойкое, чем симметричное. Разница не в криптостойкости. Главное – мы не можем расшифровать то, что зашифровали. Представьте, у нас есть оба робота – и шифрующий, и расшифровывающий. Но если у нас есть только публичный ключ, мы можем только зашифровать. Но расшифровать мы не можем. Кроме робота, нам нужна программа для него.

От теории перейдём к практике.

Самый простой и часто используемый способ шифрования — это RSA. Криптостойкость этого метода шифрования основывается на сложности

СМИО ПРЕСС

факторизации (разложении на множители) чисел, множители которых являются большими простыми числами[1].

Мы предлагаем читателям зайти на наш сайт, где вы можете в интерактивном режиме создать коды, а также шифровать и расшифровывать данные с помощью получившихся кодов. Адрес страницы: crypto.239.ru

Подробно пройдём по всем шагам для получения ключа RSA.

Первый шаг. Мы берём два (небольших в нашем случае) простых числа, например, 239 и 241 (к слову, эти числа называются числами-близнецами — простые числа, отличающиеся на 2). В реальном шифре для перемножения могут использоваться простые числа, содержащие десятки и сотни знаков[2].

Сначала мы должны придумать два простых числа:

первое простое число P_1	239
второе простое число P_2	241

Второй шаг. Перемножаем их. При перемножении мы получаем число $239 \cdot 241 = 57599$. Проблема факторизации целых чисел состоит в том, что мы вынуждены перебирать подряд все числа при поиске делителя. То есть если нам известно число 57599 и нам нужно найти его делитель, то мы начинаем перебирать возможные делители: 2, 3, 5, 7, 11 и т. д., пока мы не дойдём до числа 239. Соответственно, если злоумышленник хочет взломать наш шифр, ему нужно будет перебирать огромное число простых чисел, которые могут оказаться делителями, и каждый раз честно делить одно число на другое. В случае больших простых чисел этот перебор занимает столько времени, что к моменту его окончания расшифровывать сообщение уже не актуально[3].

Это произведение мы назовём n.

$$n = P1 * P2 = 239 * 241 = 57599$$

Третий шаг. Перемножаем произведение этих чисел,уменьшенных на 1, т.е. $(P_1 - 1) \cdot (P_2 - 1) = \varphi$. Если говорить научным языком, то мы рассчитываем функцию Эйлера. Несложно посчитать, что в нашем случае это число:

$$(239 - 1) \cdot (241 - 1) = 238 \cdot 240 = 57120.$$

[1] *Шифр назван по первым буквам фамилий его создателей: Rivest, Shamir, Adleman.*

[2] *Именно с потребностями шифрования связано нахождение всё больших и больших простых чисел.*

[3] *Например, если самое маленькое простое число будет иметь порядка миллиона знаков, соответствующий перебор потребует примерно* 10^{500000} *операций. Самые современные суперкомпьютеры делают примерно* 10^{17} *операций в секунду. Поэтому самому современному суперкомпьютеру потребуется* $10^{500000} : 10^{17} = 10^{499893}$ *секунд. С учётом того, что в году чуть больше 2 миллионов секунд (округлим до 10 миллионов), это время составляет больше, чем* $10^{499893} : 10^{7} = 10^{499886}$ *лет, что во много раз больше возраста Вселенной!*

СМИО ПРЕСС

Далее мы используем формулу Эйлера:

$$\varphi(n) = (P1 - 1)(P2 - 1)$$

Подставив наши значения, мы получаем:

$$\varphi(57599) = (239 - 1)(241 - 1) = 57120$$

Здесь нужно подчеркнуть, что зная числа P_1 и P_2, вы легко находите и их произведение (которое мы будем использовать как модуль) и функцию Эйлера $\varphi(P_1 \cdot P_2) = (P_1 - 1) \cdot (P_2 - 1)$ от него. Однако, зная только $n = P_1 \cdot P_2$, мы не можем легко найти φ. Нам нужно сначала разложить наше произведение на множители (что, как мы видели, является существенно более сложной задачей), а только потом найти φ, которое нужно, чтобы узнать код расшифровки (об этом позже).

Четвёртый шаг. Выбираем небольшое простое число $e < \varphi$ — к примеру, 29 (это число может быть постоянным во всех ваших шифрах). Оно должно быть взаимно простым с φ. При этом подходит далеко не каждое такое число.

$$НОД\,(e, \varphi(n)) = 1$$

И при этом e находится в диапазоне от 3 до φ.

Наш открытый ключ — это пара чисел e и n.

$$(e, n) = (29, 57599)$$

Пятый шаг. Вычисляем число d, такое, чтобы при делении произведения $d \cdot e$ на φ остаток был бы 1,

$$d \equiv e^{-1}\ mod\ \varphi(n)$$

То есть в нашем случае мы ищем такое число d, чтобы при умножении его на 29 и после деления на 57120 мы получили бы остаток 1.

$$d \equiv 29^{-1}\ mod\ 57120$$

В более удобном виде:

$$d * 29\ mod\ 57120 = 1$$
$$d \equiv 29^{-1}\ mod\ 57120$$

А для программистов более привычен будет вид $d * 29 \,\%\, 57120 = 1$.

Для нахождения такого d есть специальные математические методы. Данное число вы получаете с помощью расширенного алгоритма Евклида. Особо не углубляясь в суть алгоритма, нужно сказать, что он помогает найти такие коэффициенты a и b, чтобы выполнялось равенство:

$$a * 29 + b * 57120 = 1$$

Вы можете получить искомое число на нашем сайте https://crypto.239.ru/, либо просто найти его на любом другом сайте, который предлагает такой расчёт онлайн.

Мы получаем:

$$a = 5909; b = -3$$
$$(5909) * 29 + (-3) * 57120 = 1$$

Очевидно, мы нашли наш коэффициент: 5909 — число, на которое нужно умножить 29, чтобы получившееся число по модулю 57120 давало остаток 1.

Пара чисел — d и наш множитель — и есть наш секретный ключ:

$$(d, n) = (5909, 57599)$$

Напоминаем, открытый ключ у нас:

$$(e, n) = (29, 57599)$$

Как ни удивительно, но по стандартам системы число n входит и в открытый, и в секретный ключи. Мы прошли все шаги для расчёта асимметричного кода. Два числа — e (мы его выбрали как 29 на четвёртом шаге) и произведение двух простых чисел из первого шага — это открытый ключ. Таким образом:

$$(e, n) = (29, 57599)$$

Пара чисел d, которое мы получили на последнем шаге, и то же число mod из первого — это секретный ключ:

$$(d, n) = (5909, 57599)$$

Если теперь наше сообщение зашифровать открытым ключом, то расшифровать его можно будет только закрытым ключом. Получить закрытый ключ d сложно, т. к. непросто найти исходные числа P_1 и P_2, зная только $P_1 \cdot P_2$.

Теперь кто угодно может зашифровать своё сообщение, но не может его расшифровать обратно. Он может послать это сообщение с помощью открытых каналов связи, но только вы можете его расшифровать.

СМИО ПРЕСС

Перейдём к практике. Возьмём любое число и зашифруем его общеизвестным открытым ключом.

$$c \equiv m^e \ mod \ n$$

Здесь m — число , которое мы шифруем; e, n — наш открытый ключ, а c — это результат. Попробуем зашифровать число 500 (надеемся, читатель возьмёт другое число и попробует зашифровать его на сайте). Подставив число 500 в формулу, мы получаем:

$$c \equiv 500^{29} \ mod \ 57599$$

Воспользуемся программой с нашего сайта crypto.239.ru.

Для возведения в степень по модулю есть достаточно быстрые алгоритмы, но мы решили не описывать их, чтобы не усложнять этим книгу.

Мы получили число 28970. Теперь расшифруем. Для этого нам понадобится формула:

$$m' \equiv c^d \ mod \ n$$
$$m' \equiv 28970^{5909} \ mod \ 57599$$

Всё это вы можете сделать на нашем сайте. И мы призываем попробовать зашифровать и расшифровать самостоятельно несколько чисел.

Как мы можем проверить на сайте (или на собственном калькуляторе), после данных действий у нас получается исходное число 500, т. е.:

$$m' \equiv m$$

так как $28970^{5909} \equiv (500^{29})^{5909} = 500^{29 \times 5909} = 500^{1+3 \times 5712} \equiv 500^1$, поскольку $500^{5712} \equiv 1$ по теореме Эйлера (все сравнения берутся по модулю 57599).

Вы попробовали найти открытый и закрытый ключи, а также с помощью них зашифровать и расшифровать число. Можно заметить, что асимметричное шифрование гораздо более сложное, чем симметричное, поэтому оно часто применяется в комплексе, либо шифруется только небольшой хэш (этот термин будет разобран в следующей главе). Либо шифруется только код для шифровки-расшифровки, а передав этот код другой стороне, далее используется симметричное шифрование.

Ещё раз отметим – RSA невозможно взломать, потому что невозможно найти множители (делители) числа, кроме как полным перебором. При этом на каждом шаге нам нужно проверить делимость числа, что также является нетривиальной задачей.

Ещё раз отметим глобальный принцип, который лежит в основе любого шифрования. Вам нужно придумать такую операцию, при которой прямое вычисление делается существенно легче, чем обратное.

В данном случае возвести число 500 в степень 29 и найти остаток этой степени от деления на 57599

$$c \equiv 500^{29} \; mod \; 57599$$

гораздо быстрее, чем найти число 5909, которое при умножении на известное нам 29 даст 1 по неизвестному нам модулю 5712 (мы не знаем число 5712 до тех пор, пока не разложили на множители известное нам число 57599).

$$m' \equiv 28970^{5909} \; mod \; 57599$$

Потому что мы не знаем, какое должно быть число, и нам приходится искать 5909 перебором, последовательно возводя результат шифрования – число 28970 степени в степени $1, 2, 3$ и т. д., пока не получится остаток 1 от деления на 57599, либо раскладывать 57599 на множители.

Приведём пример, как ассиметричное шифрование вместе с симметричным делают невозможным просмотр чужой переписки.

Упрощённая схема шифрования в современных мессенджерах:

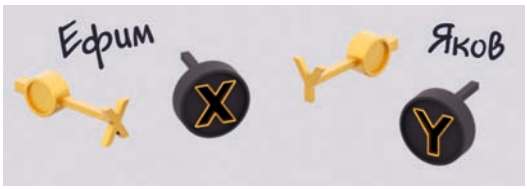


Рис. 46

1) Два пользователя, назовём их Ефим и Яков, договариваются создать закрытый канал связи. Обозначим их *X* и *Y* — каждый из них создаёт пару ключей (рис. 46).

2) Приватный ключ мы покажем как ключ, а публичный — как замок. Так же каждый из пользователей создаёт симметричный способ шифровки. Если вы помните, симметричный способ шифровки — это способ различных перестановок. Если вы можете провести перестановки в тексте, то вы сможете переставить всё обратно, получив исходное сообщение. Поэтому такое шифрование называется симметричным (рис. 47).

Рис. 47

СМИО ПРЕСС

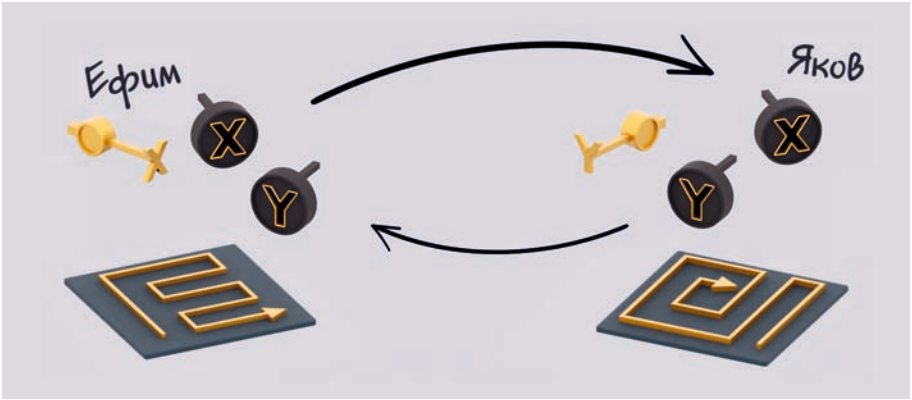


Рис. 48

3) Они передают свои публичные ключи друг другу. Теперь у каждого есть публичные ключи своего собеседника (рис. 48).

4) Теперь у каждого из пользователей есть возможность шифровать данные чужим публичным ключом так, что расшифровать их сможет только их собеседник, — шифровку мы показываем на картинке как наложение замка. Соответственно «раскрыть» этот замок может только тот, у кого есть приватный ключ.

5) Всё это время злоумышленник мог подсматривать и получать любые данные из канала, но это не поможет ему расшифровать сообщение. В лучшем случае, у злоумышленника есть только два публичных ключа (навешиваемые замки). Поэтому он может зашифровывать сообщение, но не может его расшифровать (рис. 49).

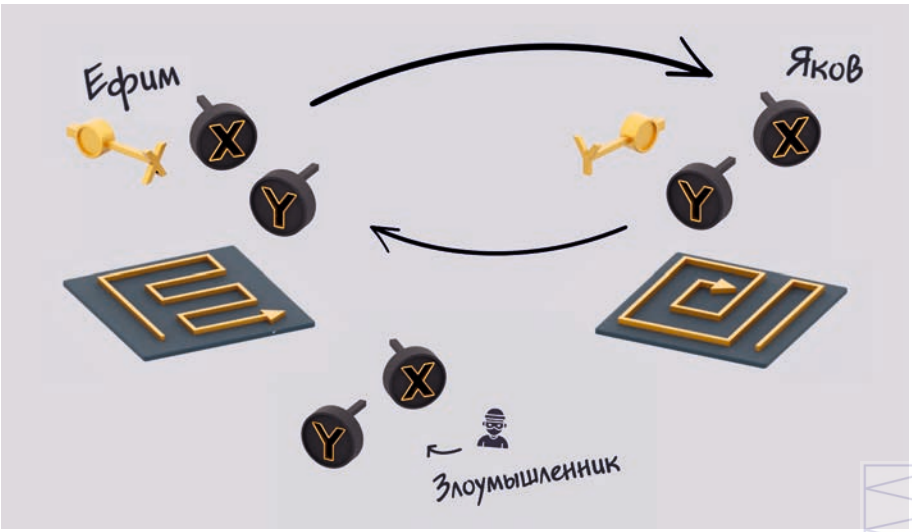


Рис. 49

СМИО ПРЕСС

6) Каждый из пользователей шифрует свой метод симметричного шифрования ЧУЖИМ замком. Злоумышленник видит эти данные, но не может расшифровать их и получить симметричные ключи шифрования.

Таким образом, оба кладут в сейфы свои симметричные способы шифрования (рис. 50):

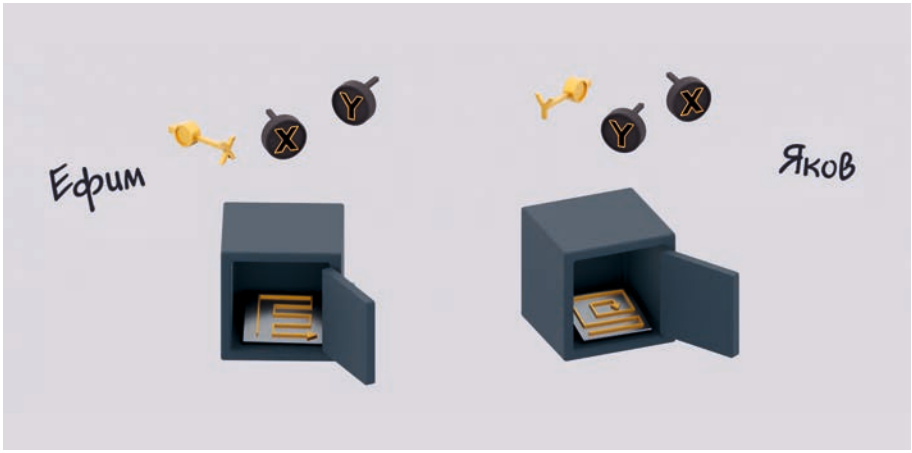


Рис. 50

После этого каждый из них закрывает этот сейф ЧУЖИМ публичным замком (рис. 51).

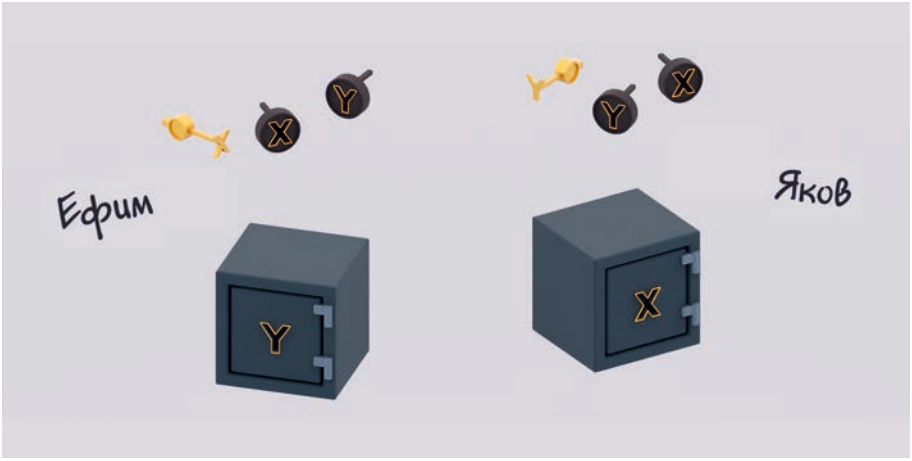


Рис. 51

Напоминаем: закрыть сейф на публичный замок мы можем, а потом открыть его мы уже не можем, т. к. у нас нет ключа.

7) Далее пользователи расшифровывают своими приватными ключами (их мы показали как ключи) эти данные. Теперь у каждого из них есть как свой, так и чужой метод симметричного шифрования — быстрый и простой. Тяжёлое

СМИО ПРЕСС

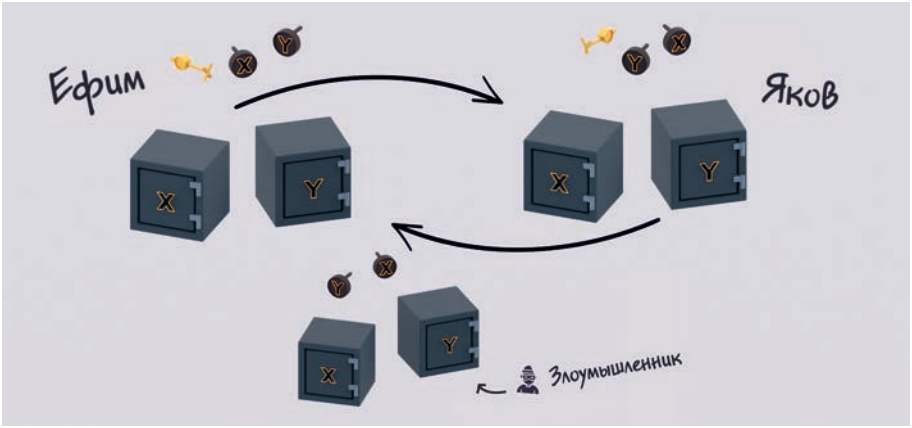


Рис. 52

ассиметричное шифрование нам нужно было только для того, чтобы передать напарнику эти методы (рис. 52).

8) Теперь Ефим может открыть сейф, который передал ему Яков, Яков может открыть сейф, который передал ему Ефим. У злоумышленника есть только два замка и два закрытых сейфа, но ни одного ключа — их просто не передавали в канале связи, поэтому перехватить их невозможно (рис. 53, 54).

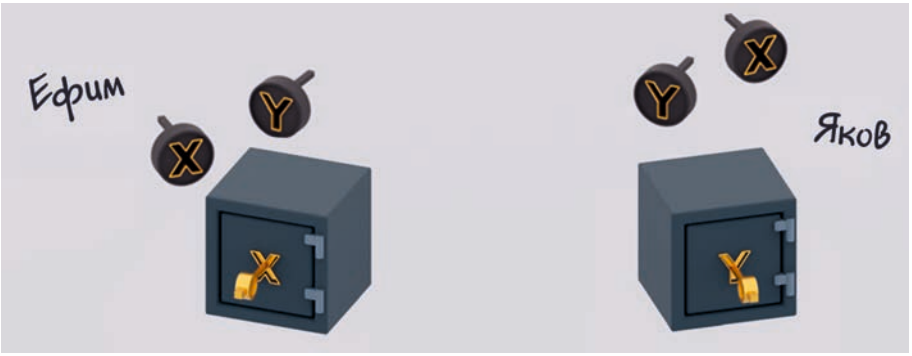


Рис. 53

Рис. 54

Так происходит во всех современных мессенджерах, где можно создать частную переписку. Каждый участник формирует свою пару приватного и публичного ключей, пересылает публичный ключ собеседнику, а собеседник шифрует им код симметричного шифрования, в котором будут пересылаться сообщения. Теперь каждый из них может шифровать свои сообщения симметричным ключом, который гарантированно неизвестен третьей стороне просто потому, что он не передаётся по каналам связи в открытом виде.

Напоминаем, симметричный способ шифрования тоже не поддаётся дешифровке, как и асимметричный. Асимметричный способ нужен не для увеличения криптостойкости, а для того, чтобы передавать ключи по открытому каналу связи.

Также асимметричный способ нужен для подтверждения, что расшифровать может только тот человек, которому сообщение предназначалось.

Возможно, вам сейчас не очевидно, как это связано с биткоином. Однако асимметричное шифрование сделало главное — была создана **возможность доказать, что ценность принадлежит вам, без необходимости раскрытия этой ценности**.

Вспомним, что главная проблема, с которой сталкивается биткоин, — как передать ценность, учитывая, что у нас не что-то физическое, а кодированная информация внутри компьютера. Как сделать так, чтобы эту ценность мог передать только тот человек, которому она принадлежит? И как её можно передавать по открытому каналу связи?

Эту проблему с помощью асимметричного шифрования решили задолго до биткоина. Когда вы заходите на любой безопасный сайт, он защищён протоколом *https*, что не позволяет злоумышленнику его подменить. Когда вы отправляете какой-либо запрос в банк или пользуетесь электронными деньгами, вы также пользуетесь стандартным асимметричным шифрованием. Суть шифрования состоит в том, что у вас есть ключ для шифрования и ключ для дешифровки сообщения. Ключ для дешифровки вы оставляете у себя, а ключ для шифрования вы можете дать кому угодно, используя открытые каналы связи.

Чтобы передать вам какую-либо ценность (данные), любому человеку достаточно зашифровать эти данные. Теперь только вы можете их расшифровать. А для передачи их кому-либо другому вам необходимо расшифровать их (это можете сделать только вы, так как только вы обладаете возможностью расшифровки), а потом зашифровать открытым ключом человека, которому вы передаёте ценности.

После зашифровки сообщения экосистема не знает, кому теперь данные (ценность) принадлежат. Этого можете не знать даже вы. Можно сказать, что теперь ценность «принадлежит тому человеку, открытым ключом которого вы эту ценность зашифровали».

В прошлой главе мы описывали эту систему как создание «ключе-замка».

СМИО
ПРЕСС

Глава 7

ДОКАЗАТЕЛЬСТВО С НУЛЕВЫМ РАЗГЛАШЕНИЕМ

Следующий пример взят из лекции академика РАН (кстати, выпускника 239 школы), автора решения X проблемы Гильберта Ю. В. Матиясевича lektorium.tv/node/32784.

Рассмотрим двух фокусников. Один хочет купить у другого секрет фокуса. Фокус состоит в том, что фокусник всегда угадывает, в какой коробке находится мячик. Но до покупки он хочет убедиться, что другой фокусник действительно обладает этим секретом.

Вероятность того, что фокусник случайно угадал, — $^1/_2$. Повторяем тот же эксперимент. Снова можно сказать, что во второй раз шанс угадывания $^1/_2$. Вероятность угадать 2 раза подряд – $^1/_4$. Если опыт повторить 10 раз, то вероятность угадать — $^1/_{1024}$. Эксперимент можно повторить любое количество раз. Вероятность того, что после k экспериментов коробка с мячиком будет угадана k раз, составляет $(^1/_2)^k$.

Любой скептик рано или поздно будет убеждён, что фокусник знает, как отгадывать, в какой коробке находится мячик. Но для каждого скептика потребуется своё количество экспериментов, чтобы его убедить.

При этом фокусник не раскрывает секрет своего фокуса, но доказывает, что этот секрет у него есть.

Пример условен, но тем не менее позволяет понять суть доказательства с нулевым разглашением, т. е. доказательства того, что у одной из сторон есть информация, не раскрывая содержание этой информации. Обычно под информацией понимают решение какой-либо задачи, описание которой есть у проверяющего. Например, человек, считающий, что у него есть доказательство Великой теоремы Ферма, мог бы с помощью подобной (хотя неизмеримо более сложной) процедуры показать, что оно у него есть, не раскрывая его сути.

Попытки показать наличие информации без разглашения этой информации предпринимались неоднократно.

Например, в Италии XVI века были популярны битвы математиков. Победивший получал призы, очень выгодные должности, почёт и уважение. Математикам нужно было составить 30 задач для конкурента и решить 30 задач, которые для них составил конкурент.

В 1512 году Scipione del Ferro нашёл первую формулу решения кубических уравнений (частный случай). Не выдавая формулу и способ решения, он выставил 30 нерешаемых для конкурента задач, сводившихся к уравнениям, которые дель Ферро умел решать[1]. Теперь он мог доказать, что он величайший математик Италии.

Однако вскоре другой математик — Никколо Тарталья смог найти решение для другого условия:

$$x^3 = mx + n.$$

И опять мог, не раскрывая формулу, решить то, что не удавалось другим.

Следующий математик, победивший Тарталью, — Лудовико (Луиджи) Феррари уже смог найти решение для уравнений 4-й степени.

Так, не раскрывая способов решения, каждый из этих математиков мог доказать, что он умеет делать то, что не умеют другие.

---

[1] *Частный вид кубического уравнения, которое научился решать Сципион дель Ферро:* $x^3 + ax = b$ *для* $a, b > 0$.

Вместо одного уравнения было несколько разных, поскольку в то время стремились избегать использования отрицательных чисел.

СМИО ПРЕСС

Другой пример, который мы рассмотрим в следующей главе о хэшировании, тоже переносит нас в эпоху Возрождения. Среди учёных также шла достаточно серьёзная борьба, связанная с приоритетом открытий. Поэтому перед публикацией открытия нужно было подтвердить, что оно есть, не раскрывая его сути. Сделав открытие, учёный часто публиковал фразу, в которой буквы были полностью переставлены, что практически не давало шансов понять смысл фразы.

Другими словами, доказательство с нулевым разглашением состоит в том, что одна из сторон должна доказать, что у неё есть решение задачи, суть которой известна обеим сторонам. К тому же полученная информация никак не поможет проверяющему приблизиться к пониманию решения этой задачи.

Опишем требования к доказательству с нулевым разглашением более полно:

1. Доказательство состоит из нескольких одинаковых раундов, в которых проверяющий играет активную роль.
2. Если у меня есть решение, то я могу доказать это проверяющему с любой заранее заданной вероятностью.
3. Если у меня нет решения, то вероятность обмануть проверяющего становится сколь угодно мала при достаточном числе раундов.
4. В результате приведения доказательства у проверяющего не появилось никакой дополнительной информации о сути доказательства, кроме самого факта, что я знаю доказательство.
5. Доказательство с нулевым разглашением адресовано конкретному проверяющему, и для любого другого проверяющего процедуру придётся проводить снова. В этом его отличие от математического доказательства (в чём-то оно больше похоже на доказательство в судебном процессе).

В современном мире криптографии идея доказательства с нулевым разглашением невероятно важна. Предположим, вам нужно доказать, что у вас есть приватный ключ. Для этого вам не нужно его предъявлять. Вашему оппоненту достаточно зашифровать что-нибудь открытым ключом, а вы должны выдать ему первоначальные данные.

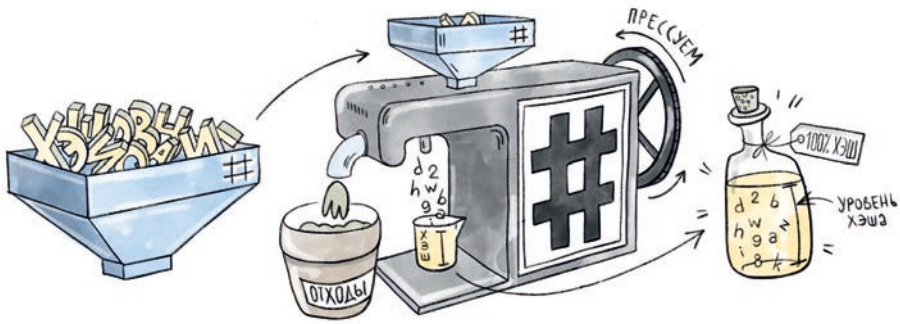
Если вы расшифруете — значит, у вас есть секретный ключ. При желании, если ваш оппонент считает, что вы могли угадать, вы можете повторить эксперимент любое количество раз. Причём, если вы предоставляете любое количество решений, это никак не приближает проверяющего к нахождению вашего секретного ключа.

СМИО
ПРЕСС

Глава 8
ХЭШИРОВАНИЕ

Хэширование — применение к тексту специальной операции (функции). При этом операция должна быть такой, чтобы по её результату нельзя было восстановить первоначальный текст.

Рис. 55



У нас есть некий текст (или вообще любые данные, которые всегда можно воспринимать как текст, то есть последовательность символов), мы его как-то меняем, и по получившемуся результату невозможно восстановить первоначальный текст

Термин «хэширование», как считается, возник в середине XX века, но и ранее данная операция применялась сотни лет.

Хэширование (от английского *hash*, что можно перевести как «перемешивание») изначально применялось некоторыми учёными, чтобы доказать своё первенство в каком-либо открытии, не раскрывая его сути. Особенно это было популярно в XVI–XVII веках среди астрономов.

Можно привести полуанекдотический случай с Галилеем, который, увидев у Сатурна кольца, решил, что это два спутника по бокам от планеты. Галилей не был в этом уверен. Тогда он опубликовал фразу

«Smaismrmielmepoetaleumibuvnenugttaviras».

Это была мешанина из букв фразы «Altissimum planetam tergeminum observavi». Её переводят по-разному, но суть фразы в том, что около Сатурна ещё 2 спутника [Сатурн тройной].

СМИО ПРЕСС

Кеплер потратил немало времени и расшифровал эту мешанину как «Salve, umbistineum geminatum Martia proles», что можно перевести как открытие двух спутников Марса. Современный читатель, разумеется, увидит иронию в том, что изначальная фраза была ошибкой, а перевод Кеплера оказался правдой. В самом деле, у Марса два спутника, в отличие от Сатурна.

Этот пример мы решили привести, потому что такое преобразование действительно можно назвать «перемешиванием» и сразу увидеть его недостатки. Во-первых, подобное «перемешивание» можно расшифровать, причём неправильно. Во-вторых, оно занимает столько же места, как и оригинальное сообщение. В третьих, нет чёткого алгоритма перемешивания, поэтому, перемешивая исходную фразу, можно получать разные результаты.

На нашем сайте вы можете написать фразу и её перемешать.

Мы постепенно будем усложнять примеры хэширования и расширять их применение, пока не придём к хэш-функциям, которые используют в современной криптографии (в том числе в биткоине).

Нужно сказать, что для дальнейших преобразований термин «перемешивание» не очень подходит. Для объяснения приведём несколько простейших примеров.

Так, хэшированием можно назвать получение остатка от деления двух чисел. То есть то, что в математике обозначается *mod*, а во многих языках программирования %.

$$X \bmod Y = Z \text{ или } X \,\%\, Y = Z.$$

Понятно, что зная Z и даже Y (делитель и частное), нельзя определить X (делимое).

В данном случае, зная число 38 и 40, невозможно сказать что делимое было 1398 (рис. 56).

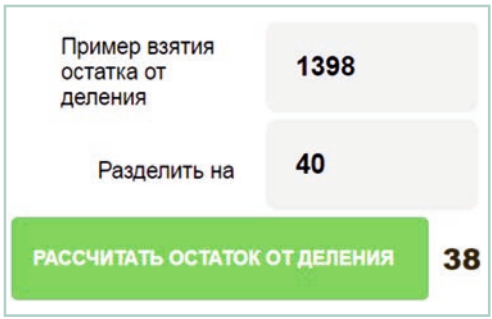


Рис. 56

Или другой пример: ряд чисел мы можем представить в виде их суммы. При таком преобразовании мы превращаем ряд чисел $1, 3, 8, 2, 5, 6$ в 25. И по числу 25 мы, разумеется, не сможем восстановить этот ряд.

С первого взгляда данное преобразование кажется бессмысленным. Тем не менее даже таким простым примерам можно найти применение.

Предположим, вам нужно передать числа по недостаточно надёжному каналу связи.

При передаче чисел $1, 3, 8, 2, 5, 6$ могут возникнуть искажения. Пускай, вы получите 1, 3, 8, 2, 5, 7. Ошибка может быть достаточно редкой (каждая 1000-я передача), однако она может стать серьёзной проблемой.

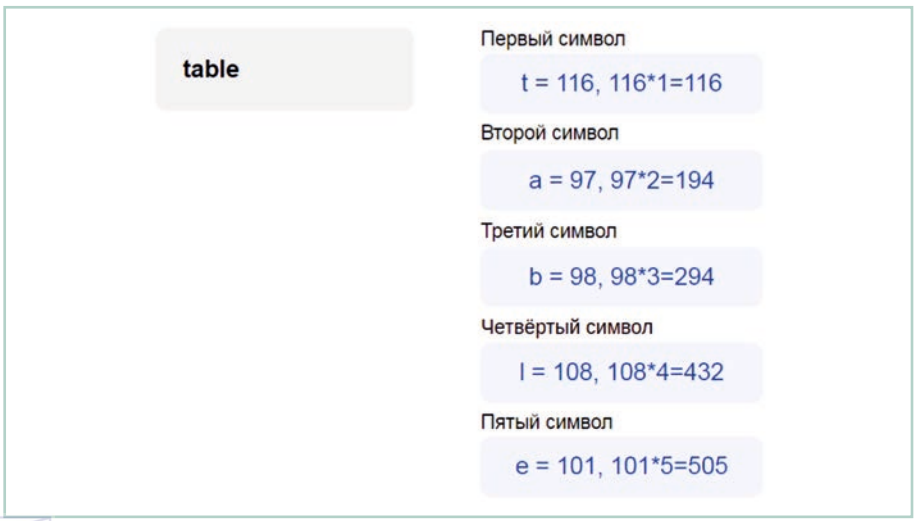
Можно передавать несколько раз, но это затратно с точки зрения объёма. Тогда вы можете передать число (например, 100-значное) и его остаток от деления, допустим, на $1\,000\,000$. Или вместе с группой чисел вы просто передаете их сумму. Это число называется «контрольной суммой».

Название «контрольная сумма» имеет чисто исторические причины. Ранее при передачи байта (8 бит), 7 бит передавали информацию, а восьмой бит был именно суммой прошлых бит (по модулю 2, т. е. если у вас было шесть 1 и один 0, то контрольная сумма — 0, а если три 1 и четыре 0 — то контрольная сумма 1). Сейчас контрольная сумма получается с помощью различных хэш-функций, которые мы и рассматриваем в данной главе, и совершенно не обязательно связана с суммированием.

При передаче большого числа можно выслать его остаток от деления, например, на 239, и оппонент на своей стороне может проверить полученное число, также разделив его с остатком. Если остатки от деления совпали, с большой вероятностью можно сказать, что потерь при передаче сообщения не было.

Данные примеры кажутся слишком простыми, чтобы использовать их на практике, но даже в случае с Галилеем их можно было бы применить с большей эффективностью. Например, буквы можно было превратить в числа, умножить каждое число на порядковый номер соответствующей буквы в тексте, сложить получившиеся — это и будет нашей контрольной суммой.

Вы можете сделать это на нашем сайте:



Контрольная сумма: (116+97*2+98*3+108*4+101*5) остаток от деления на 100 = 41

СМИО ПРЕСС

Вы можете попробовать написать другую фразу с такой же контрольной суммой и заметите, что, даже зная принцип хэширования, сделать это не так-то просто, ведь исходный текст должен быть адекватной фразой.

Остаток от деления на число — это один из основных способов хэширования в программировании, а сам термин hash можно (при большом желании) перевести как «отсечение».

Идея хэширования следующим образом была использована для облегчения поиска.

У вас есть огромный текст. Вам его нужно сохранить так, чтобы потом было легко найти в нём слова (или просто узнать, существуют ли в нём эти слова). Хранить все слова в памяти накладно. Тогда вы создаёте хэш-таблицу, где каждому слову соответствует хэш (слово превращаем в число и делим его с остатком на достаточно большое число, сравнимое с количеством слов в словаре), и записываем в базу получившийся хэш и адрес. Во-первых, мы сильно сократили базу данных, во-вторых, нам гораздо проще искать по хэшу.

Когда мы получаем запрос, мы превращаем его в хэш (не очень большое число, т. к. это остаток от деления) и ищем его в хэш-таблице.

На данном примере хорошо видны недостатки хэширования как «отсечения».

Они очевидны — это возможность так называемых коллизий. Предположим, у нас хэширование определено как деление на $10\,000$ с остатком. Тогда

$$123579 \bmod 10\,000 = 3579 \text{ и}$$
$$873579 \bmod 10\,000 = 3579.$$

Мы получили «коллизию» (запомните это слово) — одинаковый результат применения операции к разным данным. Также проблема в такой функции хэширования — что мы по результату можем легко придумать некоторое число, могущее быть исходным числом (хотя, скорее всего, оно не будет верным).

Этих недостатков лишено криптографическое хэширование.

Криптографическое хэширование

Криптографическое хэширование — это совершенно другой уровень хэширования с совершенно другими требованиями. Предыдущие способы хэширования работали «в одну сторону» — зная результат функции, нельзя определить изначальный код (аргумент функции), однако для криптографического хэширования требования гораздо более строгие. Опишем главные:

1. В результате преобразования должна получиться строка заранее известной длины.
2. Очень малая (исчезающе малая) вероятность коллизии.

3. Невозможность подобрать текст, который после хэширования даст тот же результат (коллизию) иным способом, кроме перебора[1].

Результат хэширования можно сравнить с дактилоскопической процедурой. Мы берём текст и получаем хэш, как мы получаем отпечаток пальца у человека. Мы заранее представляем вид и размер отпечатка, вероятность совпадения отпечатка у двух людей исчезающе мала (очень малая вероятность коллизии, разве что у однояйцевых близнецов). По отпечатку пальцев невозможно «восстановить» человека, кроме как перебором всех людей.

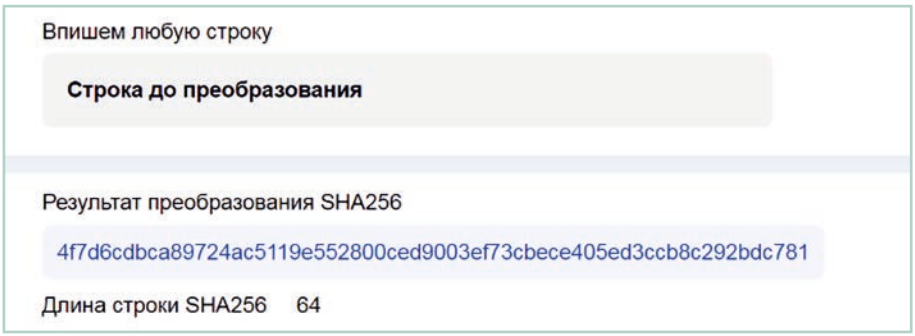
Мы превращаем текст в строку (под строкой мы подразумеваем двоичное число) заранее заданной длины — при этом у вас может быть один символ, а может быть огромная книга — но применение к этому тексту функции хэширования превратит их в строку одинаковой и заранее известной длины.

Опишем хэширование в математических терминах. У нас есть функция:

$$f(X) = Y.$$

То есть функция(аргумент) = результат.

Конкретный пример с сайта, где функция — это SHA256.



Функция хэширования должна обладать следующими свойствами:

1. Размер *Y* известен и всегда одинаков.
2. Результат *Y* сильно меняется при небольшом изменении аргумента *X*. Таким образом, нельзя провести подбор *X*, постепенно меняя его так, чтобы результат всё больше приближался к *Y*.

---

[1] *Нужно добавить, что пункты 2 и 3 не являются эквивалентными и не зависят друг от друга. У деления с остатком на 1 000 000 очень маленькая вероятность коллизии (лишь один из миллиона), тем не менее, если вы знаете, что у числа 4 000 239 остаток от деления на 1 000 000 это 239, то очень легко найти другое число, к примеру, 5 000 239, которое при делении на 1 000 000 также будет давать остаток 239 (пункт 2 выполняется, пункт 3 нет).*

Обратное также верно – ниже мы разберём метод хэширования, который потребует метод перебора, но если у нас в конце получается трёхзначное число, то примерно раз из 1000 попыток будет получаться одно и то же число, значит, пункт 2 не выполняется.

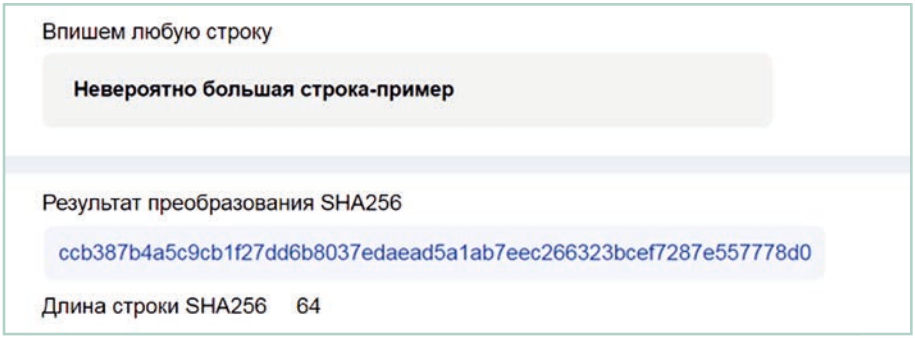
СМИО ПРЕСС

3. Минимальное количество коллизий, т. е. вероятность того, что $f(a) = b$, $f(v) = b$ при различных a и v достаточно малая. В приложении 5 мы объясняем, насколько это явление редкое.
4. Алгоритм создания хэша должен быть быстрым.
5. Преобразование одной и той же строки всегда даёт один и тот же результат (хотя это требование уже содержится в упоминании слова функция).
6. Зная результат, мы можем только полным перебором всех вариантов (что невозможно технически) подобрать такой аргумент A, чтобы $f(A) = Y$.

Рассмотрим все перечисленные пункты на примере криптографического хэширования, которое применяется в биткоине — SHA256. Мы советуем самостоятельно попробовать преобразовать строку на нашем сайте.

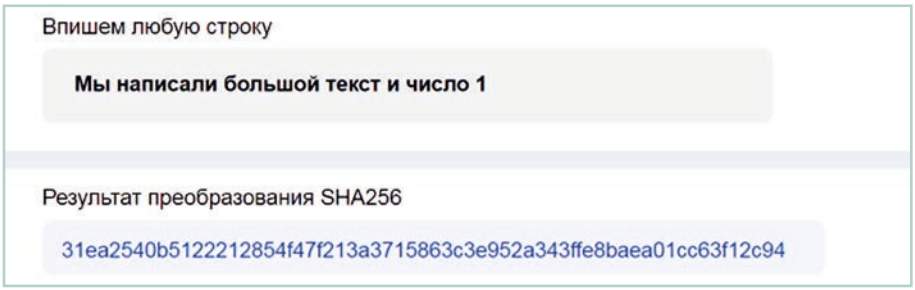
Давайте проверим, выполняются ли требования при преобразовании SHA256.

Попробуем преобразовать строку гораздо большей длины:

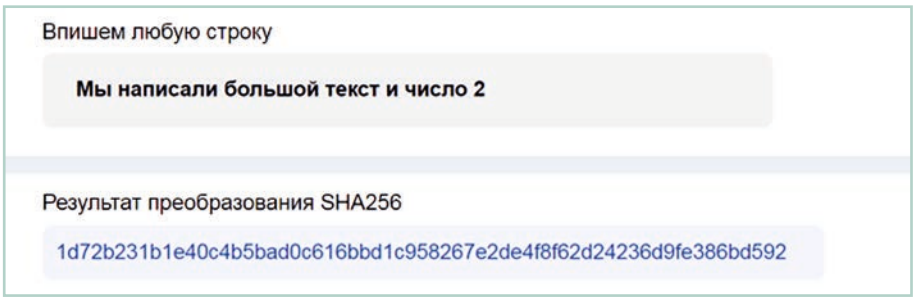


Как мы видим, длина результата не изменилась.

Проверим пункт 2 и сравним хэши у очень похожих строк.



Меняем цифру 1 на цифру 2 в конце:



Как мы видим, результат полностью изменился — буквально нет ничего похожего на предыдущий результат.

Так как при небольшом изменении у нас полностью меняется строка, то мы можем говорить о том, что у нас минимальное число коллизий, так как вариантов чрезвычайно много. Сколько же их? Длина строки-результата у нас 64 символа. Каждый символ может принимать 16 значений, т. е. число вариантов 16^{64} или 2^{256}, что сравнимо с количеством атомов в видимой части Вселенной.

Четвёртый пункт не так просто проверить, но мы можем запустить хэширование в небольшом цикле — скажем, провести 100 хэширований и оценить время. Преобразования происходят на вашем компьютере (не на сервере), и вы можете оценить скорость.

Всё происходит мгновенно.

СМИО ПРЕСС

Пятый пункт — попробовать получить преобразование SHA256 в другом месте и сравнить результаты. Мы не будем выставлять скриншоты других сайтов, вы можете попробовать в любом месте и сравнить результат.

Как видите, если вы зайдёте в другую программу и там хэшируете ту же строку, результат будет тот же самый.

Остался пункт 6 о том, что мы не сможем найти аргумент (т. е. любую строку), который после хэширования давал бы тот же результат.

То есть зная хэш, невозможно (за разумное время) определить первоначальную строку, как и подогнать аргумент под известный результат. Перефразируем известную сентенцию «фарш нельзя провернуть назад» более точно — по отпечатку пальца невозможно восстановить человека.

Единственный способ теоретически найти исходную строку (или любую строку, дающую тот же хэш) — это раз за разом меняя строку, хэшировать её, пока не получится известный нам результат, то есть просто полный перебор возможных строк. При отсутствии ограничения на длину исходной строки такое занятие видится бесперспективным.

Мы решили не приводить полного описания хэш-функции, чтобы не отпугнуть читателей, но опишем общий принцип. Для максимального упрощения мы будем хэшировать число.

Безусловно, любой символ (а значит, любой текст) мы можем представить в виде числа, поэтому это упрощение не ломает логику и гораздо проще для понимания.

Мы предлагаем читателю пройти каждый пункт вместе с нами. Здесь мы представим скриншоты нашей программы, расположенной на сайте crypto.239.ru.

1. Для простоты описания системы мы возьмём 4-значное число. Если нам нужно хэшировать меньшее число, мы дополняем его нулями, т. е. из 2 мы делаем 0002.

2. Мы берём 4 заранее известных числа. В нашем примере это будут числа, составленные из первых 5 цифр после запятой квадратных корней первых 4 простых чисел. Первые 4 простых числа — 2, 3, 5, 7.

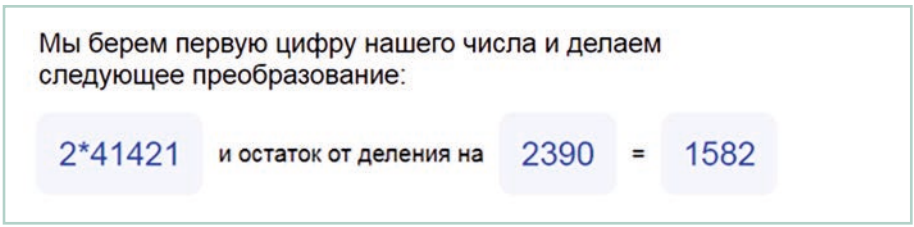
   $\sqrt{2}$ = 1,41421356... — даёт число 41421;

   $\sqrt{3}$ = 1,73205080... — даёт число 73205;

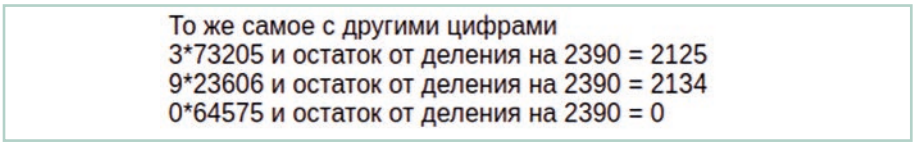
   $\sqrt{5}$ = 2,236067977... — даёт число 23606;

   $\sqrt{7}$ = 2,645751311... — даёт число 64575.

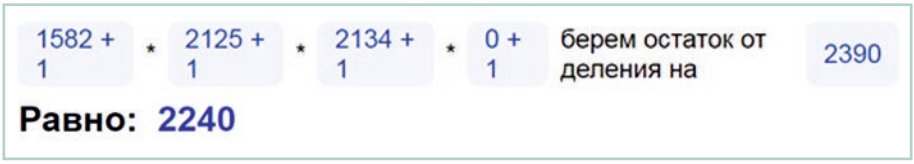
Мы берём первую цифру хэшируемого числа, умножаем её на первое из этих чисел и берём остаток от деления на наше число.



То же самое делаем с другими цифрами числа:



3. Получившиеся числа мы перемножаем и каждый раз берём остаток от деления на наше число, увеличенный на 1. $+1$, чтобы не допустить умножения на ноль:



4. Попробуем изменить всего один символ и сделаем число 2391.
5. Получилось 1546.
6. Попробуем менять по одной цифре:

   2381 даст 315; 2482 даст 2325;
   2382 даст 1742; 2483 даст 637.

   В общем, мы видим, что небольшие изменения резко меняют результат. При этом он совершенно непредсказуем.
7. Данный пример хэширования достаточно близок к реальному, однако, всё равно условен и полон недостатков. В частности, числа меньше 100 хэшируются слабее остальных. Это можно было бы исправить, к примеру, если бы мы перед пунктом 2 прибавляли определённое число.
8. Также очевидно, что примерно каждые 10 000 чисел будут выдавать одно и то же число после преобразования. Поэтому в реальном хэшировании результат показывается огромными числами. В SHA256 это число порядка 2^{256}.

   Может возникнуть вопрос: а если нам нужно хэшировать число с большим числом знаков?

СМИО ПРЕСС

Всё достаточно просто. Захэшировав первые 4 знака и получив новое 4-значное число, мы заменяем первое из наших первоначальных чисел (образованных первыми 5 знаками от корня простого числа) этим числом и продолжаем хэширование.

Очевидно, что при таком способе преобразования мы получаем строку заранее известной длины (так как в конце мы делим с остатком).

Невозможность получить исходные данные также очевидна. Каждое взятие остатка уже не даёт возможности установить исходное число — а мы взяли остаток несколько раз.

Остаётся вопрос: действительно ли мы специально не можем найти аргумент, который бы дал нам тот же результат (коллизию)?

Напомним: коллизией называется ситуация, при которой операция над разными числами даёт один и тот же результат. К примеру, при операции взятия остатка от деления на 1000, числа 10345 и 43345 дадут один и тот же результат 345.

Почему мы должны создать функцию, которая сводит коллизии к минимуму, делая их нахождение невозможным на практике?

В прошлой главе мы подробно рассмотрели доказательство с нулевым разглашением. Если при избранном способе доказательства возможны коллизии, то вы не сможете ничего доказать.

Помните пример с Галилеем? Кеплер нашёл коллизию (текст, который после перемешивания давал тот же результат), и в этом случае Галилей не может доказать, что наблюдал именно Сатурн, а не Марс.

Давайте попробуем восстановить исходное число по его хэшу.

Мы знаем, что число получилось из взятия остатка неизвестного числа от неизвестного числа, которое, в свою очередь, получилось перемножением двух неизвестных чисел. У нас одно уравнение с несколькими неизвестными. А значит, единственный способ найти решение — подставлять и пробовать решить.

Конечно, возможно, найдётся гениальный математик, который сможет найти способ получения коллизий, как нашли такой способ для MD5, но даже при такой теоретической возможности мы можем легко усложнить систему в десятки раз[1].

---

[1] *MD5 – на данный момент устаревший способ хэширования, на смену которого пришёл используемый в биткоине SHA256 (есть огромное число способов хэширования). На момент создания в 1991 году считался очень надёжным, но уже в 2005 году были продемонстрированы различные документы с одинаковой хэш-суммой, т. е. данное хэширование перестало удовлетворять п. 3 из нашего списка: «Невозможность подобрать текст, который после хэширования даст тот же результат (коллизию) иным способом, кроме перебора».*

Поэтому мы считаем, что наше хэширование не даёт возможности найти аргумент (начальный текст), а также не даёт возможности найти другое число, которое при преобразовании давало бы тот же результат, т. е. создать коллизию.

Внимательный читатель, правда, найдёт достаточно простой способ найти коллизию (или исходный текст). Нужно просто начать перебирать все числа от 0001 до 9999, пока не получим нужный результат.

Но в этом и состояла наша цель — чтобы единственный возможный способ нахождения коллизии был полным перебором вариантов. В биткоине, перебрав все числа до 2^{256}, вы действительно получите в собственное распоряжение все существующие биткоины. А почему это невозможно — и никогда не станет возможным — описано в Приложении 5.

Можно также проверить пункт «Алгоритм создания хэша должен быть быстрым». Хэширование должно применять только такие преобразования, которые легко проводят компьютеры. Проще всего компьютеры совершают битовые действия: сдвиг влево, сдвиг вправо, побитовое сложение и т.д. При обсуждении проблем хэшчейна и проблем специализированных ASICов мы затронем эту проблему, сейчас просто отметим эту особенность.

Наше преобразование не идеально выполняет этот пункт, однако в реальном мире оказывается, что излишне простое хэширование также имеет свои недостатки. Разбору этой проблемы посвящена глава 19 о майнинге.

Мы хотим также ещё раз отметить, что хэширование — это совершенно другой вид преобразования, чем шифрование. Шифрование — сложная процедура, которая требует больших вычислительных мощностей, а хранение шифра зачастую требует больше пространства, чем исходный код. А хэширование — это очень лёгкое для компьютера преобразование, результат которого легко хранить. Но невозможно по хэшу определить исходное значение.

Хэширование применяется повсеместно. Рассмотрим первое применение — хранение паролей. Когда вы заходите на какой-то сайт, вы вводите логин и пароль. Программа должна проверить ваш пароль — она должна взять то, что вы ввели, и сверить с тем, что хранится в базе данных. Но если мы будем хранить все пароли, то любой взлом приведёт к тому, что у хакера окажутся все пароли всех пользователей. Более того, любой системный администратор и любой программист компании будет знать ваши пароли.

Что же делать? Самый простой способ — хранить хэши паролей. То есть когда пользователь регистрируется, мы записываем в базу данных хэш его пароля. Когда происходит авторизация, мы берём хэш введённого пароля и сравниваем с хранящимся в базе данных.

СМИО ПРЕСС

Конечно, это не решает всех проблем. К примеру, если хакер получит в свои руки базу данных, он может начать поиск по известным ему хэшам. Например, чтобы узнать всех людей с паролём 123, он возьмёт hash (123) и найдёт все совпадения. Хакер найдёт тех, кто ввёл такой пароль, не задумываясь о безопасности. Компания может бороться с этим, добавляя к паролю так называемую «соль» (набор чисел, которые известны только компании) до хэширования, т. е. в базе данных хранится hash (123+«соль»), тогда хакер не сможет найти в базе известные ему хэши. Но соль будет для всех одна, поэтому хакер сможет найти одинаковые хэши, которые с высокой вероятностью означают очень простой пароль.

Следующий этап — добавлять в хэш не только единую для всех соль, но и логин, т. е. пароль хранится как hash (пароль+соль+логин). На этом примере можно показать, что само хэширование — это не панацея от всех бед, системы развиваются и необходимо применять хэширование с умом.

Также хэширование можно использовать для проверки файла на то, что злоумышленник не встроил в него вирус. Вы скачиваете файл с сайта, или вам был передан диск с программой, или вы скачали сайт не с сайта разработчика, а с какого-то общего сайта программ.

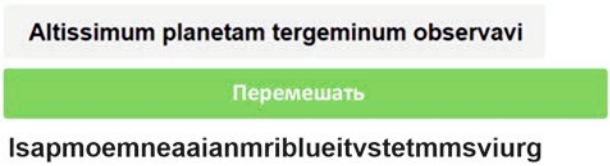
Как проверить, что в файл не встроился вирус? На сайте разработчика вы можете видеть хэш файла (который может быть прочитан просто как большая строка). Если злоумышленник поменял хотя бы один символ, хэш файла будет совершенно другой. Вы можете посмотреть хэш скачанного файла, и если он совпадает с хэшем со страницы разработчика, значит файл дошёл от создателя в неизменном виде[1].

Хэширование может применяться в совершенно неожиданных ситуациях. Предположим, что группа людей соревнуется в решении какой-либо задачи на время. Присылать решение в общий чат неправильно, потому что оно может оказаться неверным, а если оно верное, то остальным не останется возможности порешать задачу (занять второе, третье и т. п. места). После окончания соревнования необходимо будет понять, кто решил задачу первым, кто вторым и т. д. Написав решение, вы можете хэшировать его и скинуть хэш в общую переписку. Когда решит второй, он может сделать то же самое. При подведении итогов, вы можете скинуть целое решение, хэшируя которое, любой коллега может проверить, что вы решили его в тот момент, когда вы скинули решение в общий чат. Таким образом, вы можете доказать, что решили задачу, не раскрывая само решение.

---

[1] *К примеру, в системе Линукс есть специальная команда для проверки хэша файла* $ md5sum *опции файл.*

Помните пример Галилея? В современном мире он применил бы хэширование к своей фразе и получил бы код:



Другой пример — лотереи или конкурсы. К примеру, компания, производящая напитки, хочет разыграть призы. Для этого она под каждую крышечку ставит код и обещает победителю приз, если этот код будет выигрышным.

Через определённое время компания показывает выигрышный код. Но как доказать, что призы разыгрываются честно? Ведь компания может вообще не выпустить крышечки с выигрышными номерами.

Можно поступить следующим образом: компания придумывает выигрышные номера, добавляет к ним большой текст, хэширует и выставляет хэш на всеобщее обозрение.

К примеру, выигрышный код sYfeoGYdsfTweyer.

Компания берёт случайную строку

sXhlzqdrQvtX8QDPJFYcoMJCDC9jM0jgQnjoYgHpaazqHOxVQZy28ZmpxbFrndRQ2I1B

Тогда хэшировать нужно такую строку:

sYfeoGYdsfTweyersXhlzqdrQvtX8QDPJFYcoMJCDC9jM0jgQnjoYgHpaazqHOxVQZy28ZmpxbFrndRQ2I1B

И выдаёт хэш строки. Вы можете это сделать на нашем сайте.

Теперь каждый клиент может взять код под крышкой, зайти на сайт, проверить свой код, получив его хэш, и выставить на всеобщее обозрение этот хэш.

В определённый момент времени компания раскрывает последовательность чисел и «тайную» строку. Подделать строку невозможно, т. к. при изменении любого из символов (вы можете попробовать это на сайте) хэш полностью изменится.

СМИО ПРЕСС

Хэширование применяется в совершенно неожиданных местах, и его использование поистине безгранично. К примеру, если вы создаёте сайт, вы включаете в него картинки. Картинки редко меняются, и вы не хотите, чтобы при обращении к сайту пользователь каждый раз скачивал их заново.

Вы устанавливаете максимальное время кэширования (т. е. сохранения на компьютере пользователя картинки с сайта). Но как заставить компьютер скачать картинку, если она будет изменена? Вы можете вручную каждый раз устанавливать версию картинки, но это может быть достаточно сложно с точки зрения учёта.

Применив хэширование, вы легко решите задачу. Вам достаточно лишь добавить к имени файла картинки хэш от всего файла. В популярных сборщиках пакетов это будет выглядеть примерно так:

```
filename: '[name].[contenthash].js'
```

В биткоине хэширование применяется при построении хэшчейна, который рассмотрен в главе 15, а также для создания адреса, тоже рассмотренного отдельно. Также хэширование применяется в построении дерева Меркла, корень которого добавлен в заголовок блока. Об этом будет сказано также в отдельной главе о блокчейне.

Итак, в этой главе показано, что существует возможность создать «отпечаток» текста, который невозможно подделать. По отпечатку невозможно воссоздать текст иначе, как перебором (но мы не будем уверены, что нашли исходный текст, а не столкнулись с коллизией). И практически не существует текстов с идентичными отпечатками.

В прошлой главе мы изучили необходимость доказательства с нулевым разглашением. Теперь мы описали функцию, играющую ключевую роль для проведения такого доказательства.

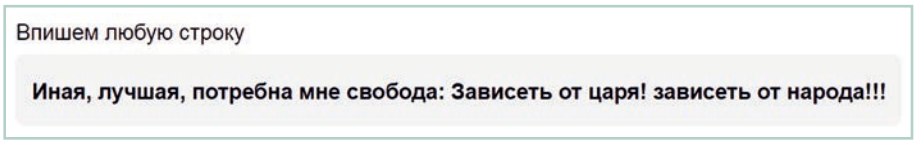
СМИО
ПРЕСС

Глава 9
ЭЛЕКТРОННО-ЦИФРОВАЯ ПОДПИСЬ

Мы хотели бы остановиться на такой важной части современной информационной экосистемы, как «электронно-цифровая подпись». Электронно-цифровая подпись (ЭЦП) служит доказательством того, что вы (и только вы) удостоверили этот текст. ЭЦП служит тем же целям, что и обычная подпись, при этом не только удостоверяет подписание текста, но и доказывает, что текст не был изменён.

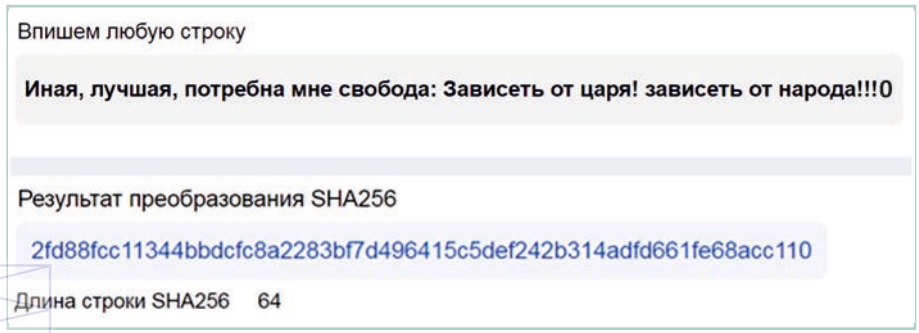
Представим себе любой текст.

На странице нашего сайта мы привели в пример стихотворение Пушкина. Далее, вы можете нажать на кнопку и получить хэш данного текста.



Получаем хэш данной строки:

Как мы помним из предыдущей главы о хэшировании, статистически невозможно придумать другой текст с таким же хэшем, а если изменить одну букву — хэш изменится полностью.



СМИО ПРЕСС

Теперь у нас есть возможность выслать контрагенту или государственному органу текст и хэш этого текста, который будет свидетельствовать о том, что текст не был изменён.

Получив от нас хэш текста, не нужно будет проверять каждый элемент. Ведь поскольку от изменения каждого символа хэш полностью меняется, любое изменение документа будет сразу же видно. Получателю достаточно провести хэширование текста и сравнить с полученным хэшем.

Сразу возникает вопрос: а если кто-то подменит и хэш данного текста? Поэтому хэш должен быть зашифрован или передан по защищённому каналу связи.

Но при пересылке получателю нужно будет убедиться, что текст написан именно нами. И мы это делаем с помощью асимметричного шифрования.

Помните, как мы шифровали и расшифровывали, используя публичный и приватный ключи? Если мы шифруем, используя публичный ключ, то расшифровать мы можем только с помощью приватного.

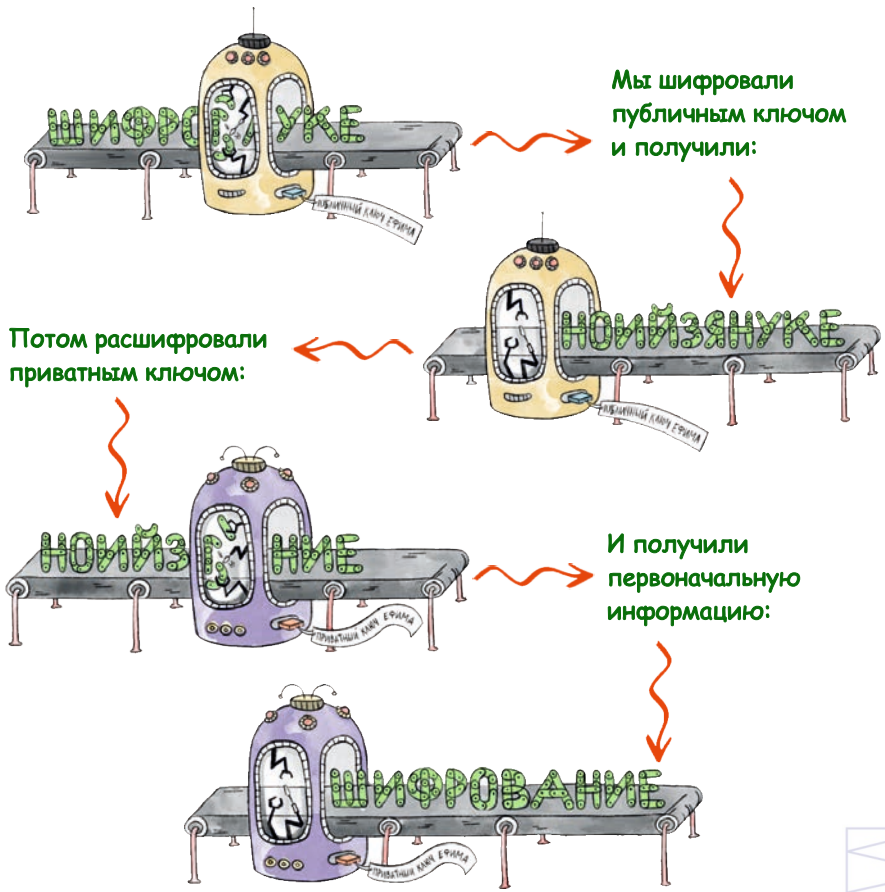
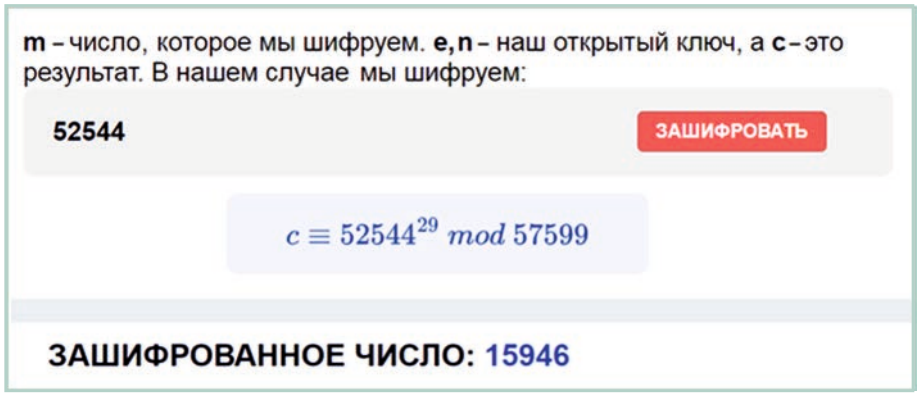
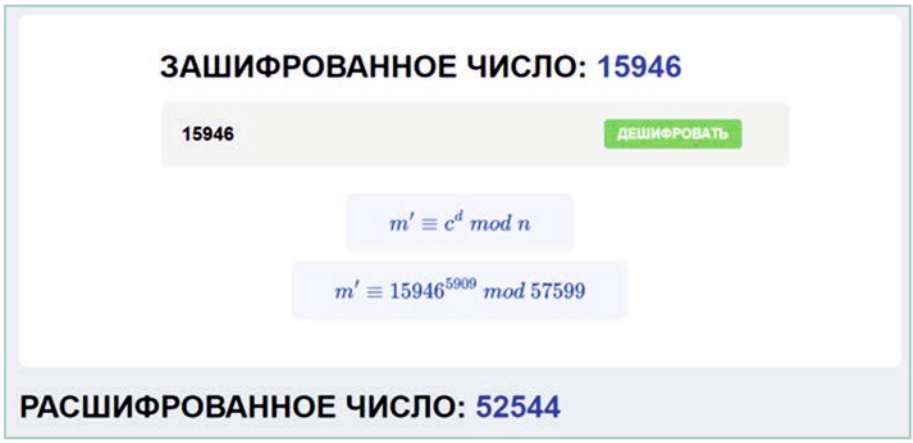


Рис. 57

Напомним главу 6. Читатель, разумеется, может ещё раз воспользоваться нашим сайтом crypto.239.ru.



А если расшифровываем:



Напомним: вы можете создать два ключа — для шифровки и дешифровки. Далее передать получателю только публичный ключ, который он использует для шифровки. Теперь отправитель может шифровать любые сообщения, передавать вам, но расшифровать их никто не может, так как вы никому не передавали ключ для дешифровки.

В случае с электронной подписью обратная ситуация: мы хотим, чтобы каждый мог расшифровать сообщение, но зашифровать его могли бы только мы!

СМИО ПРЕСС

Ещё раз обсудим, зачем это нужно: так как никто не может зашифровать сообщение, то значит, шифровали его точно мы. Это и является сутью Электронной Цифровой Подписи.

Как же это сделать? Оказывается, что по сути ничего не надо менять.

Мы просто запускаем в нашего робота приватных ключей незашифрованную информацию. Ведь по сути это робот, для него любая информация — это просто информация. И он по правилам приватного ключа начинает преобразование (рис. 58):

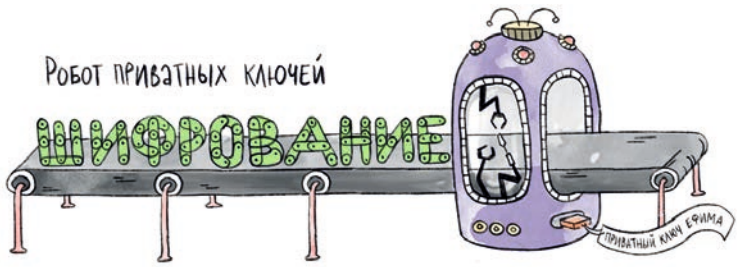


Рис. 58

В процессе шифрует наши незашифрованные данные (рис. 59):

Рис. 59

И получает (рис. 60):

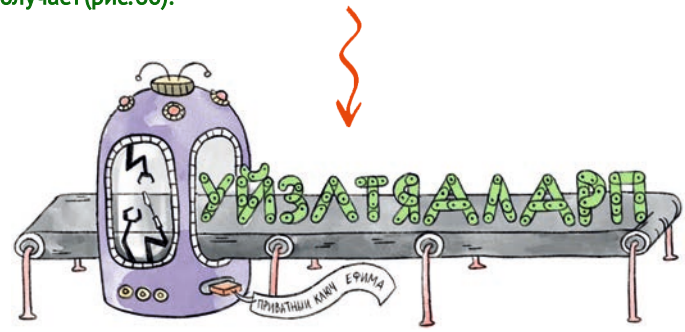
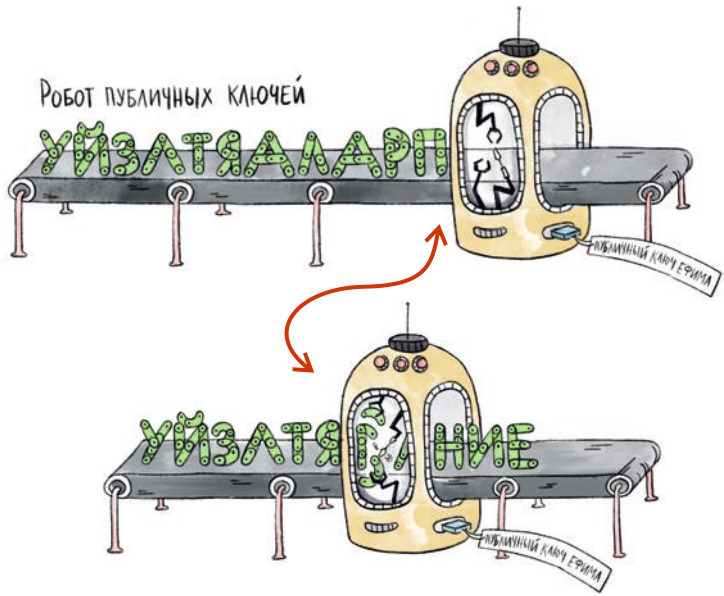


Рис. 60

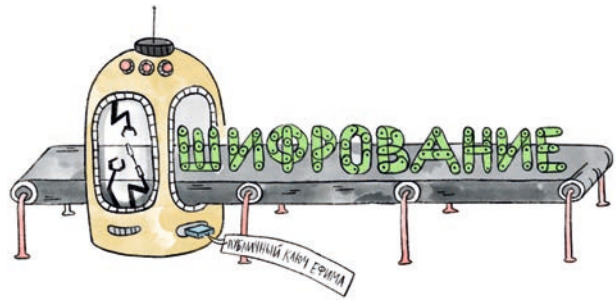
Заметьте: буквы совсем не те, что получил бы робот публичных ключей. Но мы помним, что робот публичных ключей делает просто обратную операцию, и если мы запустим это в него (рис. 61):

Рис. 61



Мы получим исходное (рис. 62):

Рис. 62



Мы шифруем с помощью приватного ключа, а получатель расшифровывает с помощью публичного ключа[1]. В нашем примере (на сайте вы можете самостоятельно шифровать-расшифровывать любые, но не очень большие числа) мы с помощью публичного ключа превратили 52544 в 15946.

[1] *Пусть f – просто выполняемая функция, а g – сложно выполняемая функция – две взаимно обратные функции. Пусть x – некоторое сообщение. Шифрование публичным ключом – это нахождение $f(x)$, расшифровка – нахождение $g(f(x)) = x$.*

Пусть теперь y – хэш подписываемого сообщения. Электронная подпись – это нахождение $g(y)$, а расшифровка – нахождение $f(g(y)) = y$.

СМИО ПРЕСС

Чтобы превратить (расшифровать) 15946, нам нужен ключ для дешифровки, который является приватным.

Опыт авторов подсказывает, что понять это можно, только производя несколько шифровок-расшифровок в прямом и обратном порядке (например, на нашем сайте crypto.239.ru).

Помните: только что мы шифровали 52544 и получили 15946? То есть (рис. 63):

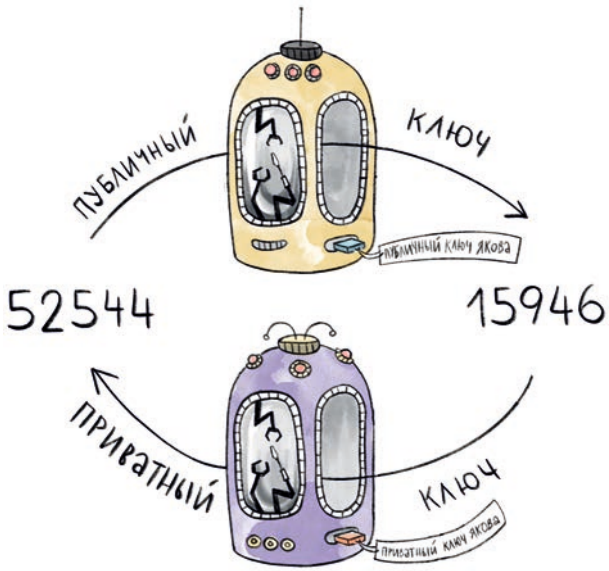


Рис. 63

Применяем публичный ключ (52544) — получили 15946.
Применяем приватный ключ (15946) — получили 52544.

Предположим, мы хотим, наоборот, чтобы каждый мог расшифровать зашифрованное нами сообщение. Мы взяли наше число 52544 и применили к нему приватный ключ.

И получили 26791. Так как применение публичного ключа — это обратная операция, то применив к 26791 публичный ключ, мы получим исходное число 52544.

Как мы видим, благодаря тому, что публичный ключ — это обратная операция к приватному ключу, мы получаем первоначальное число, т. е. мы можем шифровать приватным, а расшифровывать публичным ключом (рис. 64).

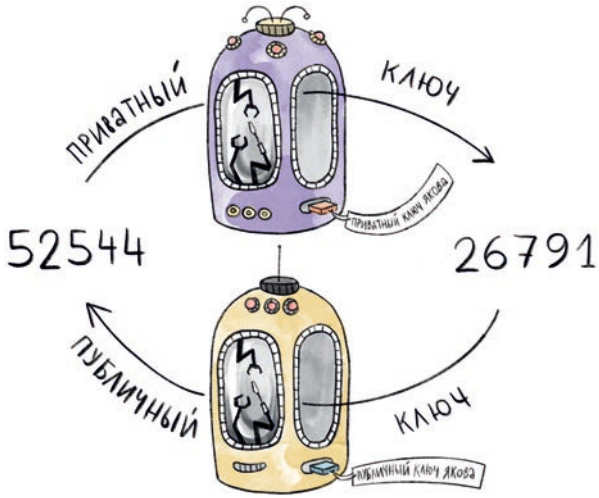


Рис. 64

Возвращаясь к формулам, $f(g(x)) = x$, а также $g(f(y)) = y$, нужно заметить, что, разумеется, $g(x)$ НЕ РАВНО $f(x)$ при равном x.

Как вы видите, с помощью той же страницы, где вы шифровали публичным, а расшифровывали приватным ключом, вы можете сделать обратную операцию. Зашифровать приватным (получив другой шифр), а расшифровать данные публичным ключом.

Неожиданно, но совершенно логично, не правда ли? Если раньше мы раздавали всем наш публичный ключ, чтобы они могли зашифровать данные,

СМИО ПРЕСС

а расшифровать могли только мы, теперь мы раздаём всем публичный ключ для того, чтобы они могли расшифровать зашифрованные нами данные.

Давайте ещё раз обсудим, для чего это нужно. Если человек смог расшифровать данные вашим публичным ключом, значит, зашифровали его вы и только вы, потому что только у вас есть приватный ключ!

Преобразование ваших данных вашим приватным ключом и называется электронно-цифровой подписью (ЭЦП). То есть когда мы говорим: «Я подписал», это означает: «Я зашифровал так, как могу зашифровать только я, вот мой шифр».

Напоминаем, на с. 55 (рис. 52) мы подробно объясняли, зачем мы шифруем публичным ключом, а расшифровываем приватным. Потому что публичный ключ нам передали по открытому каналу связи, злоумышленник тоже может его получить. Но т. к. расшифровка происходит приватным ключом, расшифровать данные невозможно.

Здесь обратная ситуация. Мы готовы к тому, что любой расшифрует наше сообщение. Нам важно доказать, что зашифровали именно мы. Поэтому мы шифруем недоступным никому приватным ключом, а расшифровка идёт публичным ключом, который может быть общедоступен.

Теперь рассмотрим полный алгоритм ЭЦП на схеме. До конца главы мы будем разбирать эту схему, но если она уже очевидна, можно переходить к следующей главе.

Процесс подписи (рис. 65):

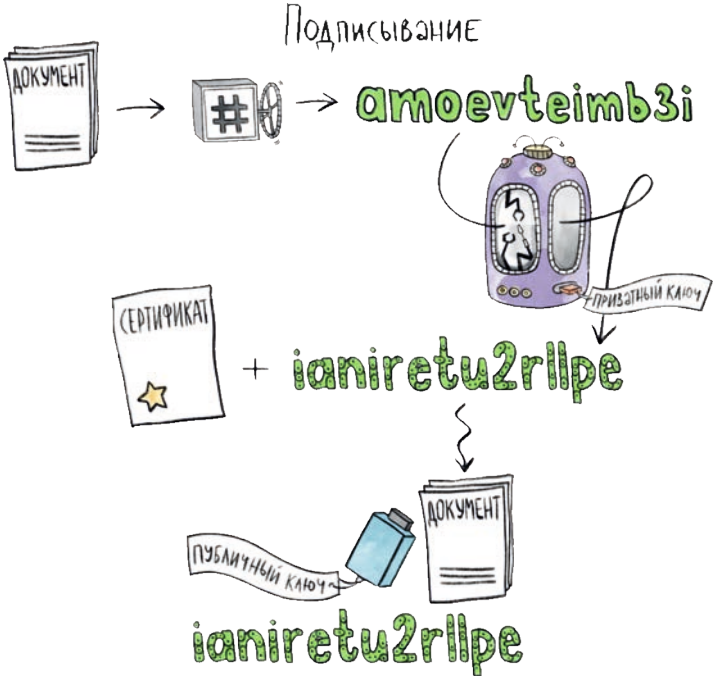


Рис. 65

Процесс проверки:

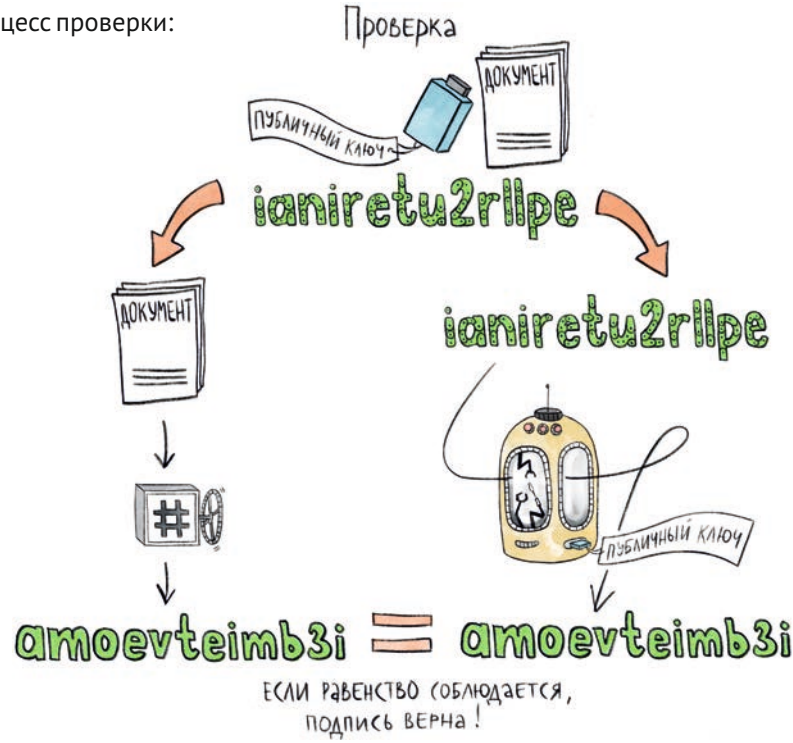


Рис. 66

Далее обсудим каждый шаг отдельно. Как мы видим, в схеме есть ещё один элемент, который мы не обсудили ранее. Это сертификат. Проблема состоит в том, что если вы не знакомы с владельцем подписи лично, вы не можете точно знать, чья подпись к вам пришла. Тогда вы должны обратиться в общий реестр, честность которого не должна подвергаться сомнению. Реестр по публичному ключу выдаст вам информацию о владельце.

Если не использовать реестр, то, во-первых, вы просто не будете знать, от кого пришла подпись. А если знаете просто о факте прихода подписи от определённого человека, то нет уверенности, что именно он его выслал.

Хакер может вклиниться в передачу данных следующим образом.

СМИО ПРЕСС

Борис высылает все свои данные. Хакер получает данные, расшифровывает публичным ключом Бориса, меняет данные, шифрует своим секретным ключом и высылает Максиму. Максим думает, что данные выслал Борис.

Покажем, как можно избежать приведённого выше плохого сценария, используя схему с сертификатом (рис. 67).

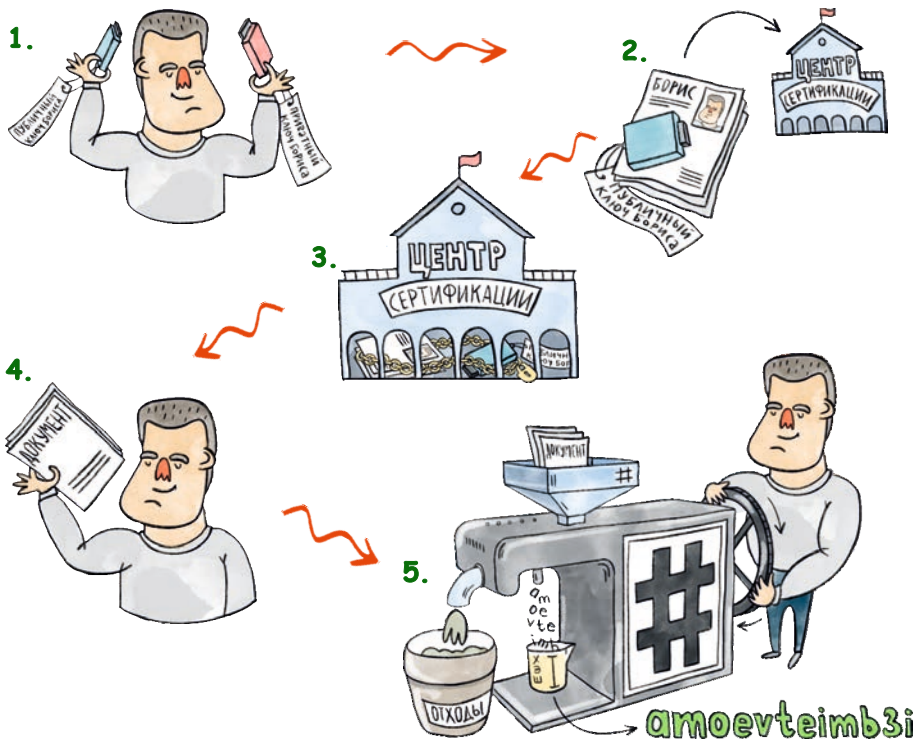


Рис. 67

1. Борис создаёт два ключа — публичный и приватный.

2. Он берёт публичный ключ, записывает о себе все данные и отправляет в сертификационный центр.

3. Борис получает сертификат, то есть уведомление о том, что его данные и публичный ключ хранятся в сертификационном центре.

4. Подготовка закончена, у Бориса есть документ, который он хочет заверить, т. е. доказать, что он согласен с документом (договором) и «подписывает» его.

5. Борис получает хэш из документа (стандарная процедура, которую может сделать каждый).

Он получает хэш из всех данных[1].

[1] *Реальный хэш примерно в 2–3 длиннее, чем* amoevteimb3i. *Здесь просто использован лишь фрагмент для более красивой картинки.*

Рис. 68

6. Борис шифрует хэш нашим приватным ключом (рис. 68).

7. Получается у нас зашифрованный хэш из наших данных. Как мы помним, если что-то зашифровано приватным ключом Бориса, оно может быть возвращено в первоначальный вид с помощью публичного ключа.

Рис. 69

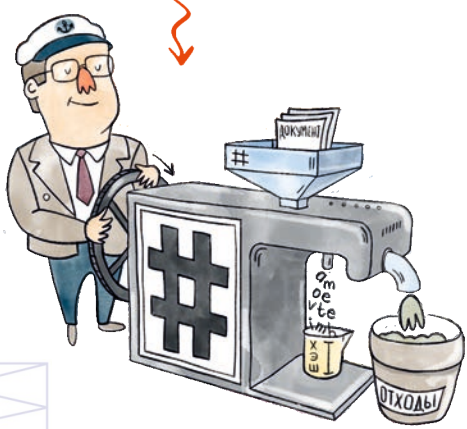
8. Борис высылает информацию и зашифрованный хэш этой информации (рис. 69).

9. Как мы видим, Максим получил три элемента: исходный документ, зашифрованный хэш из этого документа, публичный ключ Бориса.

Рис. 70

10. Максим запрашивает у сертификационного центра, кому принадлежит полученный публичный ключ. Сертификационный центр отвечает ему, что публичный ключ принадлежит Борису (рис. 70).

Рис. 71



11. Далее, Максим получает из данных хэш (функция хэширования общеизвестна и любой человек может получить хэш из любых данных). Понятно, что из документа Бориса и Борис и Максим получают один и тот же хэш — в нашем случае amoevteimb3i (рис. 71).

12. Максим расшифровывает хэш публичным ключом Бориса (рис. 72) и получает из

ianiretu2rllpe → **amoevteimb3i**.

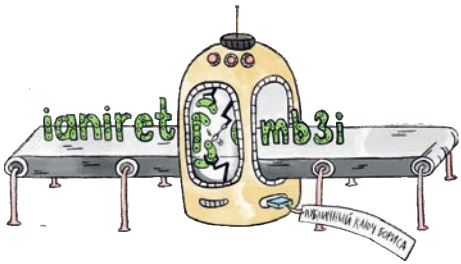


Рис. 72

13. Сравнивает два хэша — полученный из текста и расшифрованный нашим публичным ключом (рис. 73).

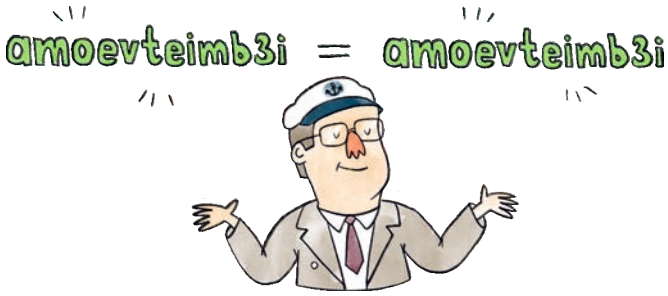


Рис. 73

14. Теперь Максим может быть уверен, что информация дошла до него в нужном виде и отправил его именно тот, кому принадлежит публичный ключ. Хакер не мог изменить информацию, т. к. тогда бы изменился хэш в п. 1, хакер не мог подменить шифрованный хэш, так как к нему бы тогда перестал подходить публичный ключ. Хакер не может поменять (теперь) публичный ключ, так как сертификационный центр в случае подмены ответит, что этот ключ не принадлежит Борису.

На всякий случай, ещё раз напоминаем, зачем нужно хэширование. Во-первых, асимметричное шифрование достаточно энергозатратная процедура, и если мы шифруем только хэш, мы серьёзно упрощаем работу. Во-вторых, саму проверку неизменности вашего текста проще проверить по хэшу, т. к. при изменении даже на один символ хэш будет полностью видоизменён, и это будет легко заметить, а изменение файла проверяется только посимвольно.

СМИО ПРЕСС

Глава 10
АСИММЕТРИЧНОЕ ШИФРОВАНИЕ НА ЭЛЛИПТИЧЕСКИХ КРИВЫХ

Биткоин не использует шифрование, которое основано на факторизации (разложении на множители) числа, о котором мы говорили в главе 3. Используется гораздо более красивая система, основанная на эллиптических кривых.

Если вам не интересна теория шифрования, то эта глава не обязательна для чтения. В следующей главе будет описана практика по созданию ключей, её также можно пропустить, если вам не интересно, как на самом деле создаются ключи. Эти главы не обязательны для изучения остальных.

Здесь описывается более сложная, нежели RSA, форма шифрования с помощью эллиптических кривых.

Два главных преимущества шифрования с помощью эллиптических кривых:

1) то, что вы зашифруете сейчас, нельзя будет расшифровать практически никогда (в отличие от факторизации чисел, которая теоретически может быть произведена с резким скачком производительности компьютера);

2) очень простое создание приватного ключа (нет необходимости искать большие простые числа).

Недостаток: *идеология шифрования гораздо сложнее*[1].

Первое преимущество вытекает из того, что, согласно концепциям современной физики, перебор не будет возможен никогда (см. Приложение 5).

Обсудим второе преимущество. Как вы помните, для создания приватного ключа RSA нужны два больших простых числа, само нахождение которых является сложной задачей. А при создании приватного ключа с помощью эллиптической кривой нужно случайно взять большое число (меньшее, чем 2^{256}).

Но обо всём по порядку.

Теория, связанная с эллиптическими кривыми, — наверное, самая красивая математическая теория. Именно с помощью них в конце XX века была доказана Великая Теорема Ферма.

[1] *Но мы настолько всё подробно и чётко описали, что всё должно пройти без проблем*:).

СМИО ПРЕСС

Рассматриваемая нами эллиптическая кривая задаётся равенством: $y^2 = x^3 + ax + b$.

Легко видеть, что эллиптическая кривая симметрична относительно оси абсцисс.

На рисунке (рис. 74) показано, как выглядит эллиптическая кривая.

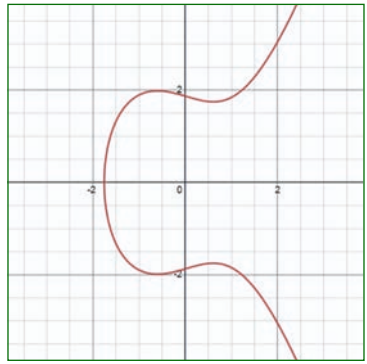


Рис. 74

С помощью эллиптической кривой определим действие, которое ставит в соответствие двум её точкам третью точку этой кривой, называемую суммой исходных двух точек.

Для наглядного представления можно использовать любой онлайн-калькулятор, например: https://www.desmos.com/calculator/ialhd71we3?lang=ru.

Если мы возьмём любые две точки на заданной эллиптической кривой, проведём через них невертикальную прямую, то прямая пересечёт кривую в третьей точке (так как уравнение для нахождения абсцисс точек пересечения прямой и кривой будет кубическим; два его корня — это абсциссы взятых двух точек, тогда у кубического уравнения должен быть и третий вещественный корень). Конечно, в случае касания точки могут «слипнуться». К примеру (рис. 75):

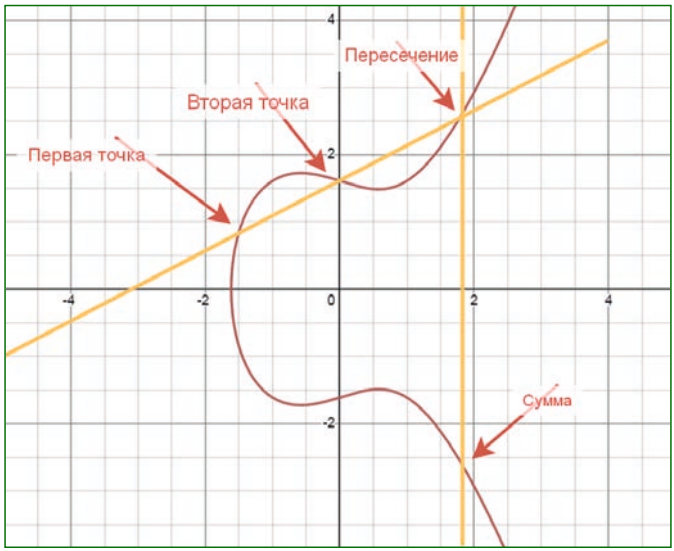


Рис. 75

Через найденную точку мы проводим вертикальную прямую, и вторая точка пересечения этой прямой с эллиптической кривой и есть сумма исходных точек (таким образом сумма двух исходных точек — это точка, симметричная относительно оси абсцисс третьей точке пересечения прямой, проведённой через исходные точки, с эллиптической кривой).

Нужно просто принять такое определение «суммы». Это не сумма в традиционном понимании (например, радиус-векторов этих точек), а особо определённая таким хитрым способом операция[1]. Так мы можем «складывать» любые точки на данной кривой. К примеру, другой вариант (рис. 76):

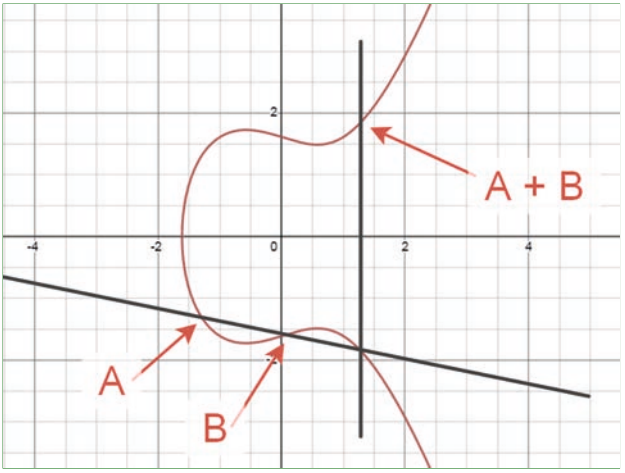


Рис. 76

Снова берём две точки, ищем третье пересечение с кривой, рисуем к ней вертикальную прямую и в её месте пересечения мы находим четвёртую точку, которую и называем суммой.

Мы можем взять совершенно другую точку *C* и суммировать её с получившейся точкой (рис. 77):

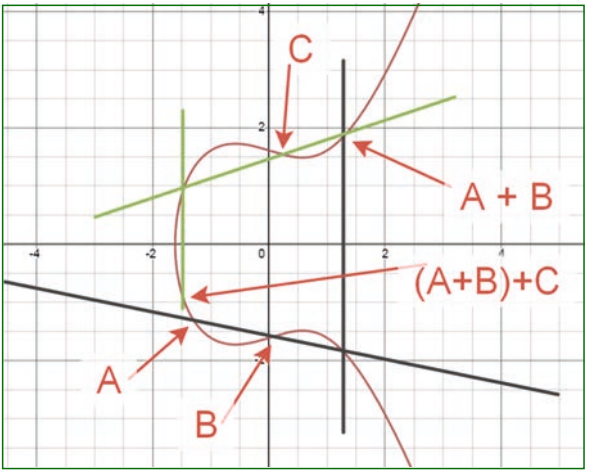


Рис. 77

[1] *Искушённый читатель может заметить, что множество точек эллиптической кривой с введённой на нём операцией суммы образует коммутативную группу (интересно, какая «точка» является нейтральным элементом :)).*

СМИО ПРЕСС

Так мы можем «складывать» любые точки, получая всё новые и новые. Но что будет, если сложить точку саму с собой? Получится следующая точка (разумеется, прямая, проведённая через точку и её саму, — это касательная к эллиптической кривой в данной точке) (рис. 78):

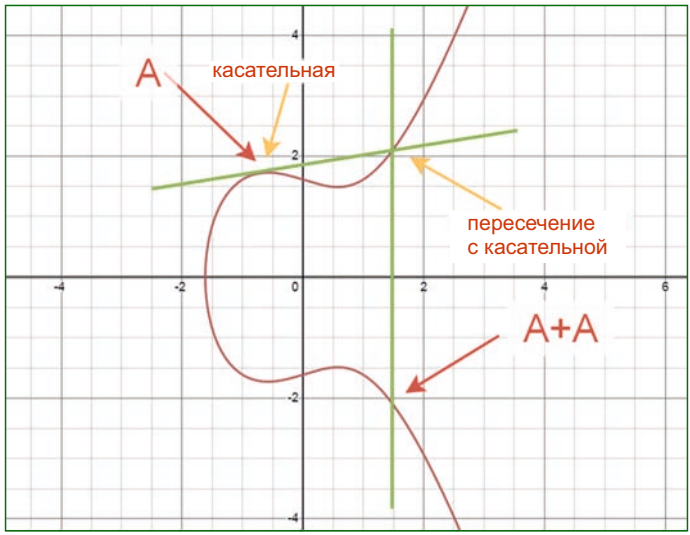


Рис. 78

Теперь мы можем сложить получившуюся точку $2A$ саму с собой (рис. 79).

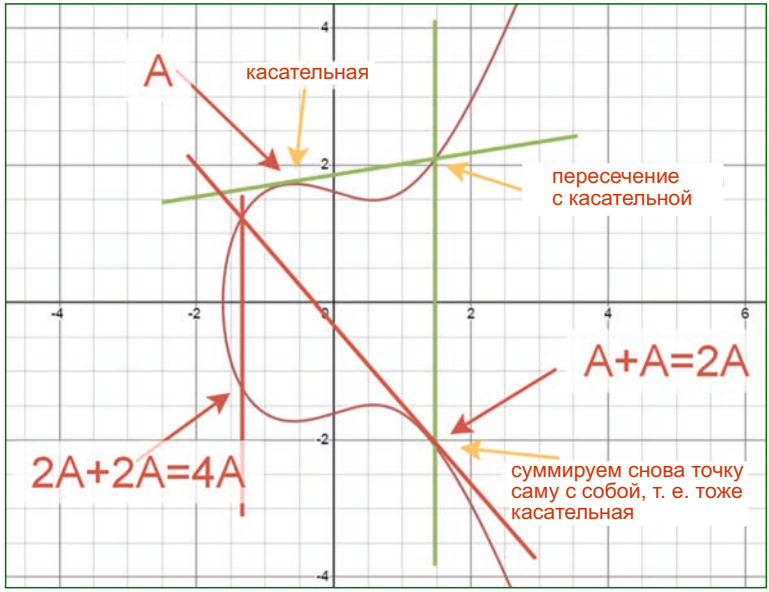


Рис. 79

Сложим точку *A* саму с собой n раз, где n — произвольное выбранное нами число. Координаты получившейся точки будут служить публичным ключом. А вот приватным ключом будет служить то самое число n.

Само шифрование немного сложнее, чем RSA, но подчиняется тем же правилам. Сам математический путь шифрования мы не включили в книгу, данную информацию вы можете прочесть дополнительно[1]. Идея приватного и публичного ключей точно такая же, как и с RSA. Например, приватным ключом мы будем шифровать сообщение, которое можно будет расшифровать только с помощью нашего публичного ключа (показав, как в электронной подписи, что сообщение зашифровано именно нами).

Для биткоина был взят стандарт криптографии Standards for Efficient Cryptography Group (SECG), который был предложен консорциумом Certicom in 1998: https://www.secg.org/sec2-v2.pdf

Этот стандарт берёт частный случай и просто постулирует некоторые параметры.

Так, формула кривой определена:

$$y^2 = x^3 + 7$$

(именно такой график мы и нарисовали в примере).

А базовой точкой (которую мы складываем определённое количество раз) принята (натуральные числа ниже записаны в шестнадцатиричной форме):

X-координата: 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798

Y-координата: 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8

Порядок (максимальное возможное число суммирований точки): FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141

Все эти данные вместе и называются стандартом secp256k1. Здесь эти параметры даны для полноты понимания. По сути, все эти числа (коэффициенты в формуле кривой, координаты точки, которую мы складываем саму с собой) просто постулированы и, как мы надеемся, взяты случайным образом в недрах Certicom Research и определены в http://www.secg.org/sec2-v2.pdf.

При создании биткоина Сатоши Накамото просто взял этот ранее созданный стандарт.

---

[1] *К примеру:*
Шифрование данных на базе эллиптических кривых. Д. Ф. Пастухов, Ю. Ф. Пастухов, П. Р. Синица.
Элементарное введение в эллиптическую криптографию. Алгебраические и алгоритмические основы. Болотов А. А., Гашков С. Б., Фролов А. Б., Часовских А. А.
Также, возможно, будет удобнее просто посмотреть какую-либо онлайн-лекцию.

СМИО ПРЕСС

Зная базовую точку (она указана выше), мы можем перейти к генерации двух ключей — приватного и публичного, которые работают совершенно так же, как ключи в описанной схеме RSA. Приватный ключ биткоина (как и практически любой другой криптовалюты) — это просто случайное число, меньшее, чем порядок[1] (порядок указан выше).

Мы можем придумать это число, например, бросая монетку 256 раз. Записав результат бросаний в виде последовательности нулей и единиц (например, решка — 0, орёл — 1), мы получим двоичное число, которое и будем использовать в качестве приватного ключа.

После того как мы его придумали, из этого числа мы можем получить публичный ключ. То есть публичный ключ — это точка на графике, которую мы получаем, складывая между собой базовую точку (*G*) число раз, равное приватному ключу.

То есть публичный ключ = $G + G + G + \ldots + G + G$ — и так мы складываем столько раз, чему у нас равен приватный ключ.

Как следует из названия, ваш публичный ключ общедоступен. Однако злоумышленнику это не поможет найти приватный ключ, который и нужен для шифровки сообщений.

Если злоумышленник захочет узнать наш приватный ключ, он должен будет взять базовую точку (которая известна) и складывать её между собой столько раз, сколько потребуется, чтобы получить публичный ключ (а это займёт миллиарды и триллионы лет, даже используя всю энергию Вселенной)[2].

Возникает вопрос: мы, чтобы получить публичный ключ, должны делать то же самое, и, соответственно, это займёт у нас столько же времени, как и у злоумышленника. Как этого избежать?

Мы определяем, что сумма $G+G$ — это умножение на два. То есть чтобы сложить G 8 раз, мы можем просто сложить два G, получить точку Y, далее сложить два Y, получить D, а после сложения двух D, получить нужную точку. Вместо 7 сложений $G + G + G + G + G + G + G + G$ мы сделали всего 3 сложения $(G + G) = Y$, $(Y + Y) = D$, $(D + D)$ = то что нужно = $G + G + G + G + G + G + G + G$.

Давайте ещё раз повторим. Помните, как на предыдущей картинке (рис. 79) мы сначала сложили $A + A$, затем полученную точку $2A$ можно сложить с A и получить точку, обозначаемую $3A$. Затем, сложив $3A$ и A, можно получить новую точку, обозначаемую $4A$.

---

[1] *Напоминаем, порядок – это максимальное возможное число суммирований базовой точки.*

[2] *С каждым следующим сложением точки-суммы появляются в совершенно непредсказуемых местах. Ошибившись в какой-либо координате точки на 1, через не очень много операций мы получим точки в совершенно другом месте, чем правильные. Поэтому все вычисления нужно производить точно до того момента, когда координаты получившейся суммы будут в точности равны координатам точки-публичного ключа. Тем самым, зная публичный ключ, мы не сможем оценить даже порядок величины приватного ключа. А прикидку числа необходимых операций мы провели в Приложении 5.*

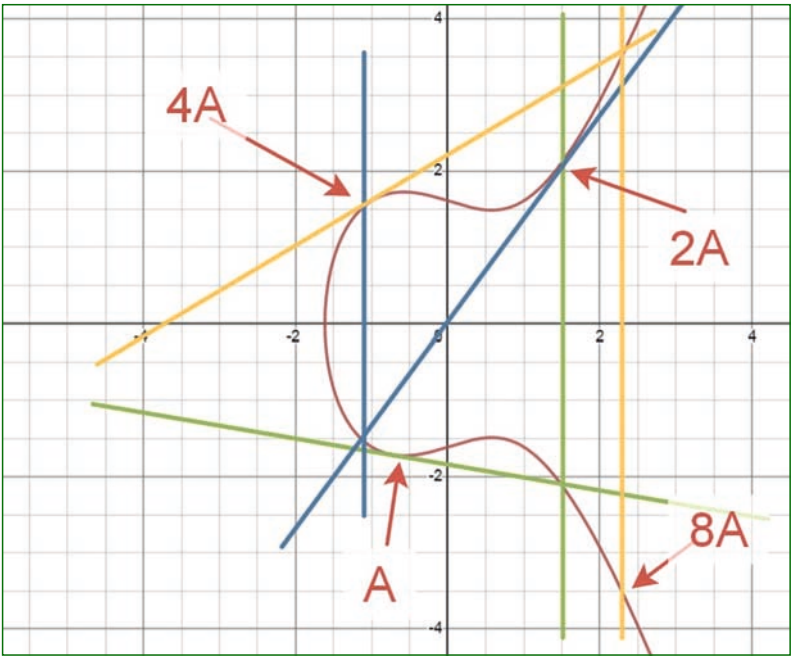


Рис. 80

Теперь мы умеем умножать точку и $4A$ мы получили лишь за два действия: $A+A$ и $2A+2A$. Чтобы получить $8A$ нам нужно только три действия (рис. 80).

Итак, мы умеем складывать точки между собой, а также умножать их на два (складывать саму с собой). Аналогично можно получить $4A$, сложив $3A$ и исходную точку A и так далее.

Оказывается, если мы знаем публичный ключ, нам лишь надо записать его в двоичной записи. К примеру, чтобы просуммировать 1024 раза, нужно найти всего 10 сумм. А чтобы получить $1040 = 2^{10} + 2^4$, нам нужно всего 15 операций[1]:

$$(A + A)*2*2*2*2*2*2*2*2*2*2 + (A + A)*2*2*2 = 1040A.$$

То есть вместо 1039 сложений у нас получается 15 сложений. Для огромных чисел, у которых сотни знаков, это принципиально уменьшает число операций[2]. То есть зная приватный ключ, мы легко находим публичный, представив приватный в виде суммы различных степеней двойки. Но, зная

[1] *Вдумчивый читатель заметит, что достаточно 11 операций, так как в процессе получения первого слагаемого было получено и второе, ещё раз его получать не надо* :). *Впрочем, для дальнейших оценок мы будем пользоваться грубыми прикидками.*

[2] *Число операций сложения точек для получения из точки A точки nA будет меньше* $(\log_2^2 n)$. *Таким образом, для чисел, меньших* 2^{256}, *число сложений будет меньше, чем* $256^2 < 10^5$.

СМИО ПРЕСС

только публичный ключ, нам придётся суммировать огромное количество раз по одной точке, что невозможно сделать на текущей стадии развития математической теории (и, скорее всего, невозможно будет сделать никогда).

Итак, мы получили пару ключей. Приватный (просто большое число) и публичный (координаты точки).

В данный момент мы лишь подробно описали систему шифрования. И у читателя может возникнуть вопрос: «При чём тут шифрование и передача денег?». Однако в системе биткоина (и других криптовалют) шифрование и есть передача денег. А расшифровка — это их получение.

Когда мы будем говорить про кошельки и адреса, нужно помнить, что в действительности это не кошельки и адреса, это возможность шифровать и расшифровывать любые данные. И если они зашифрованы вашим приватным ключом, значит, они принадлежат вам. А для передачи ценности вам необходимо расшифровать информацию вашим приватным ключом, а после этого мы только шифруем переданным нам публичным ключом.

Это очень важно помнить — мы не передаём кому-то деньги, мы шифруем так, чтобы кто-то другой мог расшифровать.

СМИО
ПРЕСС

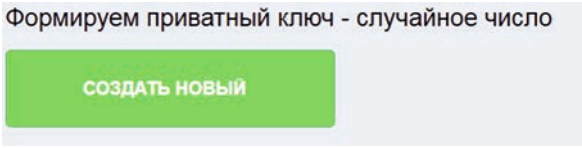
Глава 11
СОЗДАНИЕ ПРИВАТНОГО И ПУБЛИЧНОГО КЛЮЧЕЙ

В этой главе мы на практике изучим создание приватного и публичного ключей. Отметим, что это не настоящие ключи криптовалют и их нельзя использовать в криптовалютах напрямую. В криптовалютах обычно есть небольшие добавления, чтобы они отличались от обычных ключей, используемых в «классическом» шифровании на эллиптических кривых.

Но эти ключи, в теории, можно использовать для реальной подписи ваших сообщений и последующей их проверки адресатом.

Первое. Мы создаём приватный ключ. Как мы помним, это всего лишь случайное число. Попробуйте сформировать это число на нашем сайте, воспользовавшись кнопкой по адресу https://crypto.239.ru/makekeys (рис. 81).

Рис. 81



После нажатия на кнопку, вы получаете приватный ключ (рис. 82). Так как это просто число, его можно представить в классическом для человека виде. В десятичной системе:

Рис. 82

Как мы помним из предыдущей главы, чтобы максимально быстро подсчитать итоговую точку из базовой, нам нужно разложить это число по степе-

СМИО ПРЕСС

ням двойки (что то же самое: записав в двоичной системе счисления, т. е. представив его в удобном для компьютера двоичном виде) (рис. 83):

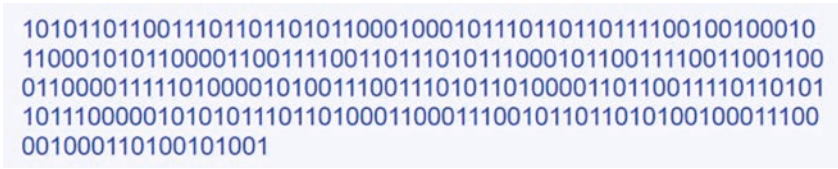


Рис. 83

Подчеркнём, это то же самое число, просто в двоичном виде. Давайте остановимся на этом моменте чуть подробнее. Система кажется очень простой, но нужно договориться о терминологии.

Число 5678, представленное в двоичной системе, будет: 1011000101110. Это то же самое, что сказать: число $5678 = 2^{12}+2^{10}+2^{9}+2^{5}+2^{3}+2^{2}+2^{1}$, то есть представить число в двоичном виде — это то же самое, что «разложить число по степеням двойки»[1].

Напоминаем, зачем нам нужно раскладывать число по степеням двойки. Помните предыдущую главу? Когда мы «складываем» точки на эллиптической кривой, то можем складывать только две точки, соответственно, мы можем умножать только на 2 ($2*A = A + A$).

Предположим, у нас стоит задача получить $5678*A$, т. е. точку A нужно сложить саму с собой 5677 раз. Как нам это сделать?

Можно просто «в лоб» посчитать $A + A + A + \ldots + A$ — и так 5677 раз, а можно разложить число 5678 по степеням двойки $5678 = 2^{12}+2^{10}+2^{9}+2^{5}+2^{3}+2^{2}+2^{1}$ — и тогда требуемое число сложений уменьшится:

$A*2*2*2*2*2*2*2*2*2*2*2*2*2$ (12 раз) + $A*2*2*2*2*2*2*2*2*2*2*2*2$ (10 раз) + + $A*2*2*2*2*2*2*2*2*2$ (9 раз) + $A*2*2*2*2*2$ (5 раз) + $A*2*2*2 + A*2*2 + A*2$.

За 48 сложений получится тот же результат, что и при складывании точки A 5677 раз.

Особое значение это имеет при огромных числах. В нашем примере число:
785287706964559270527249881758589605178022657996732943825837715836148719649 69

Именно такое количество раз нам нужно сложить точку саму с собой. На это не хватит всей энергии Вселенной[2]. Но если мы разложим это число по степеням двойки:

1010110110011101101101011000100010111011011011110010010001011 0001010110000011001111001101110101110001011001111001100110001100 0011111010000101001110011101011010000110110011111011010110111000 0001010101110110100011000111001011011010100100011100001000110100101001

[1] *В десятичной системе $5678 = 5 \cdot 10^{3} + 6 \cdot 10^{2} + 7 \cdot 10^{1} + 8$.*

[2] *Разумеется, число сложений точки на 1 меньше этого числа.*

то нужно будет гораздо меньше действий. Самый большой знак у нас 1, чтобы найти число, нам нужно будет умножить базовую точку *A* на 2^{256}, потом на 2^{255} и так далее. Даже если всё число будет состоять из единиц, нам нужно будет совершить не более $(256 + 1)*(256/2) = 32896$ операций. Это число, разумеется, несравнимо меньше, чем

78528770696455927052724988175858960517802265799673294382583771583614871964969

Второе. Теперь нам нужна базовая точка.

Напоминаем, базовая точка — это просто точка, заданная своими координатами, единая для всех, придуманная давно и не нами. Эта точка – часть стандарта secp256k1:

X-координата:

55066263022277343669578718895168534326250603453777594175500187360389116729240

Y-координата:

32670510020758816978083085130507043184471273380659243275938904335757337482424

Для подсчёта итоговой точки, как мы помним, мы складываем эту точку между собой определённое количество раз. Далее мы рассматриваем конкретно у нас получившийся приватный ключ, у вас, конечно, будут слегка другие данные при работе с сайтом.

Наш первый символ (справа, т. е. меньший)	1	Мы берём базовую точку, чтобы сравнить с общей суммой
Наш второй символ	0	Значит, тут 0
Наш третий символ	0	Значит, тут 0
Наш четвёртый символ	1	Мы берём базовую точку, умножаем её на 2 (складываем саму с собой), потом ещё раз на 2 (уже следующую получившуюся складываем саму с собой) и ещё раз на 2 и складываем с точкой на прошлом этапе. Получившуюся точку запоминаем

СМИО ПРЕСС

И так далее — вплоть до цифры в начале. Вручную, конечно, складывать невозможно, но на сайте программа легко с этим справляется, и в итоге у нас получится точка:

X-координата:

761857597937284670106773719842004081797248333577075594869125344060872760281 60

Y-координата:

291581024866864957959568336527233141664692006528587364556069818060913112677 75

Это очень непривычный вид публичного ключа, обычно он записывается в более удобном PEM-формате (это лишь удобство записи):

-----BEGIN PUBLIC KEY-----
MFYwEAYHKoZIzj0CAQYFK4EEAAoDQgAEgyX1VP2QRkyuDi+u1Dkwzab2E0E7OlUF 8CNUKo0FkPBPqT9u/LMn4NtZkX8EHluDTaLX+zF2w2KplDKKG10bDg==
-----END PUBLIC KEY-----

Этот же ключ можно представить в виде 16-ричной записи.

Ниже изображение публичного ключа электронной подписи одного из авторов (рис. 84).

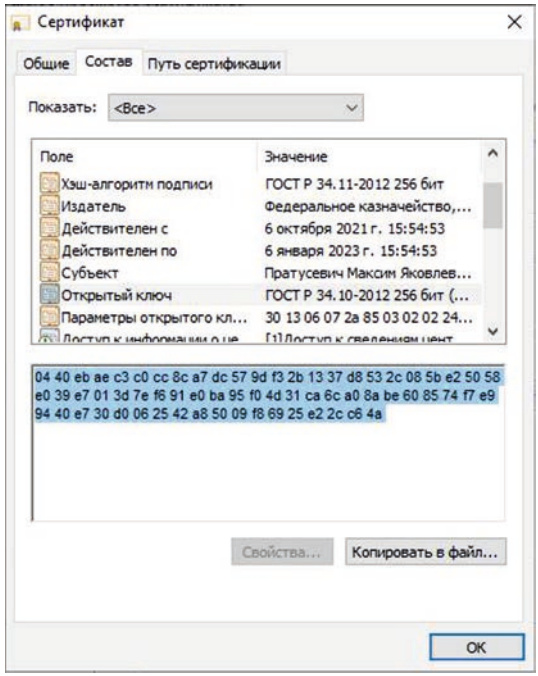


Рис. 84

Теперь мы можем получить подпись любого текста, имея наш приватный ключ (рис. 85):

Рис. 85

А также проверить, верна ли подпись под текстом, имея публичный ключ (рис. 86):

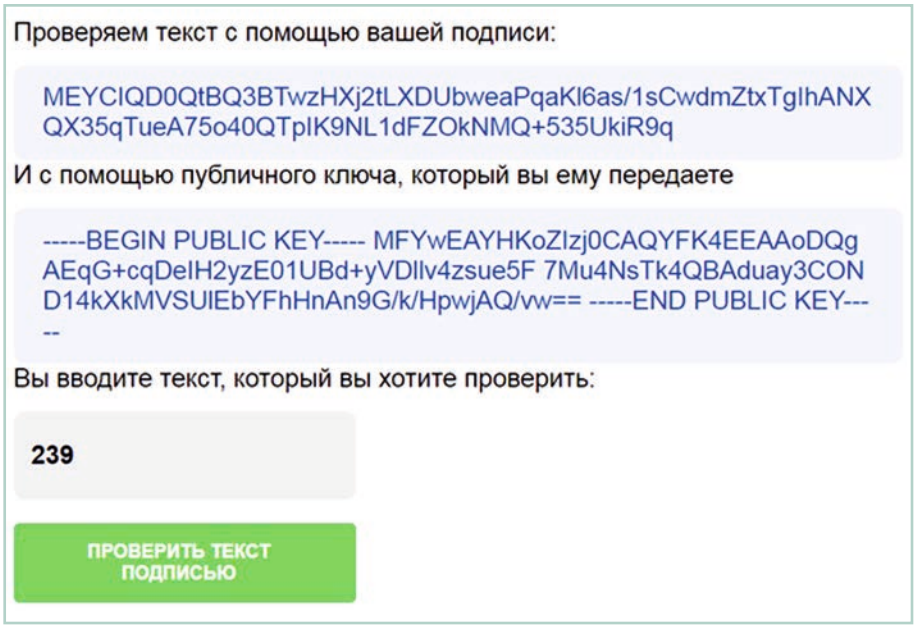
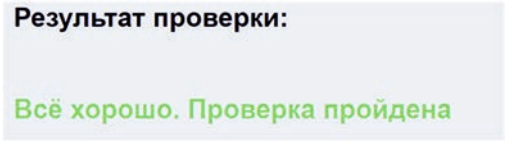


Рис. 86



Теперь вспомним главу 9 об электронной цифровой подписи. Имея приватный ключ, можно подписать любой текст и переслать его любому человеку. После этого получатель может подтвердить, что этот текст действительно написал отправитель, так как при расшифровке текста его публичным ключом получится исходный текст (или хэш), а значит, зашифровать его мог только он.

Можно ли взломать такую подпись? Мы знаем базовую точку и итоговую точку. Чтобы найти приватный ключ (сколько раз надо сложить базовую, чтобы получить итоговую), нет другого выхода, кроме как действовать перебором. Взять базовую точку, сложить её саму с собой, проверить, не получилась ли итоговая. Далее сложить ещё с базовой точкой, ещё раз проверить и так далее. Итак, в нашем примере:

78528770696455927052724988175858960517802265799673294382583771583614871964969

Вот столько раз! Как мы показываем в Приложении 5, это совершенно невозможно при современном уровне развития науки и техники.

СМИО
ПРЕСС

Глава 12
PROOF OF WORK

Идея может быть сформулирована следующим образом: для получения доступа к ресурсу сервера пользователь должен выполнить некоторую задачу, которая удовлетворяет следующим критериям:

1. Задача должна решаться долго, но должна быть разрешима за адекватное время.
2. Серия однотипных задач не должна приводить к облегчению решения каждой отдельной задачи. Иначе говоря, если необходимо решить 100 таких задач, то время на их решение будет примерно равно времени решения одной задачи, умноженному на 100.
3. Самое главное — проверка решения на стороне сервера (контрагента) должна быть очень быстрой, гораздо быстрее, чем решение этой задачи.

В 1997 году Адам Бэк предложил данную идею для решения проблемы спама. А в 1999 году появляется термин «Proof of Work» — «Доказательство работы», что означает: клиент решил данную задачу и может пользоваться ресурсами сервера.

Рассмотрим конкретный пример. Пользователю нужно отправить письмо. Но сервер должен отличить это письмо от спама или другой нежелательной массовой рассылки. Тогда мы вводим следующее правило: сервер принимает только такое письмо, хэш текста которого будет содержать 00 в начале.

Разберём на конкретном примере на нашем сайте.

У нас есть сообщение, в котором мы приглашаем Владимира Владимировича на конференцию, посвящённую теореме Эйлера.

ОБЩАЯ СТРОКА:

Уважаемый Владимир Владимирович, мы приглашаем вас на серию лекций, посвящённых проблемам и решениям теоремы Эйлера

Hash от общей строки:

367617abfc109d920e552e80d0f344241bcb1b265d609aa5d539cd989fef7bd3

Сервер не примет такое сообщение, потому что в начале нет 00.

Хэшируя просто письмо, вы, скорее всего, не добьётесь, чтобы хэш начинался с 00. Что же вам делать? А добавить любой символ в конец письма! И снова хэшировать. Здесь мы добавили 1 (рис. 87).

СМИО ПРЕСС

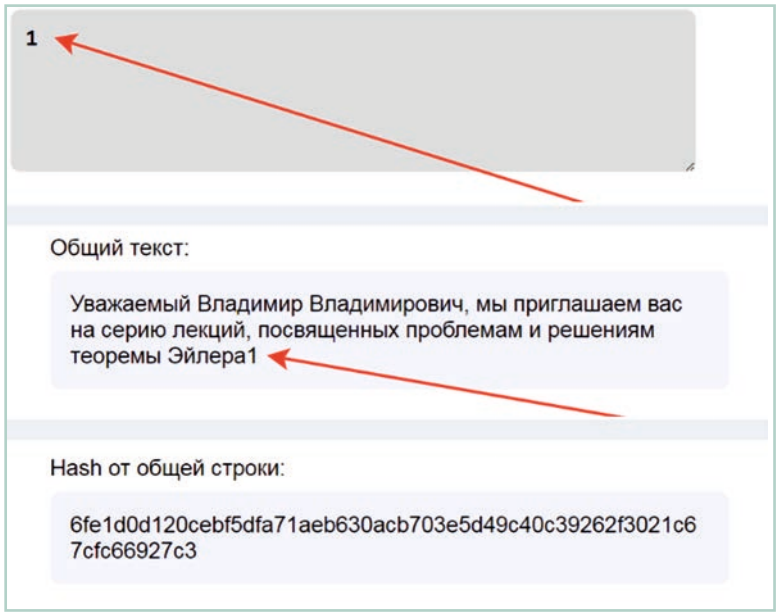


Рис. 87

Если снова не получилось — то снова хэшировать и так далее, пока мы не добьёмся результата.

Наконец получилось — оказалось, что чтобы в хэше в начале стояло 00, нужно было добавить в конец письма 1234564789 (рис. 88).

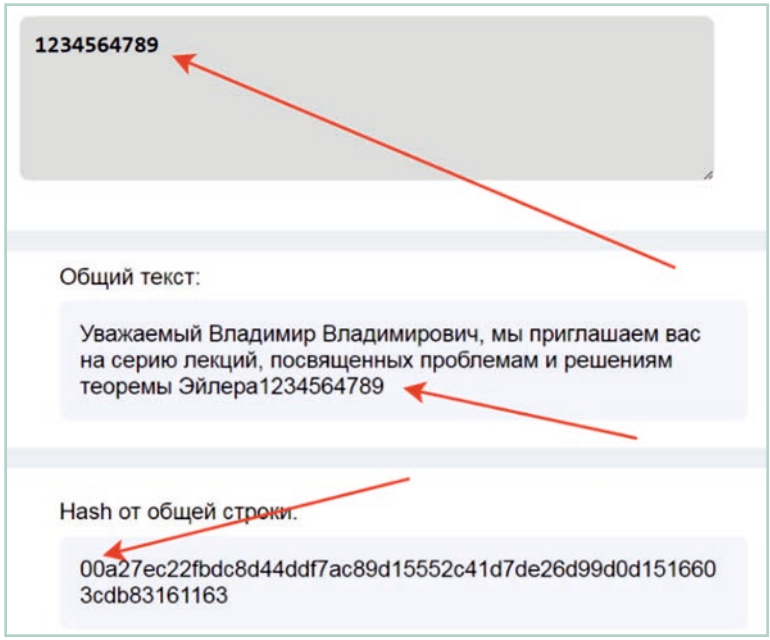


Рис. 88

Как видите, мы затратили достаточно продолжительное время. Но получив это письмо, сервер может взять хэш и получить 00 в начале, доказав тем самым, что мы выполнили работу и достойны выслать письмо. Мы предлагаем вам поэкспериментировать на нашем сайте.

Действительно, оказалось, что данная процедура полностью соответствует нашим целям:

1. Она достаточно быстрая и достаточно сложная одновременно. При этом мы можем менять её сложность (требовать разное количество нулей в начале хэша, с ростом производительности компьютера увеличивая её).
2. Расчёт хэша данного письма никак не поможет в расчёте хэша другого письма.
3. Серверу очень легко за минимальное время проверить ваш расчёт.

В реальной ситуации борьбы со спамом, разумеется, нужно хэшировать не текст письма, а текст письма вместе с адресом, куда письмо уходит, а также с примерным временем рассылки. В этом случае, даже если спамер захочет послать то же письмо тому же человеку через 10 минут, ему будет необходимо снова совершить такую же работу!

Забегая вперёд, отметим, что эта добавка к письму в терминах биткоина называется nonce, а количество нулей, необходимых для доказательства работы, приблизилось к 21 и называется сложностью.

Как вы помните, в главе 3 о задаче византийских генералах мы постулировали, что такая задача существует. Теперь мы конкретизировали задачу. Чтобы генералы приняли время атаки, кто-то из них должен придумать время, а потом к этому времени найти такую текстовую добавку, что хэш из «времени + добавка» давал в начале определённое количество нулей.

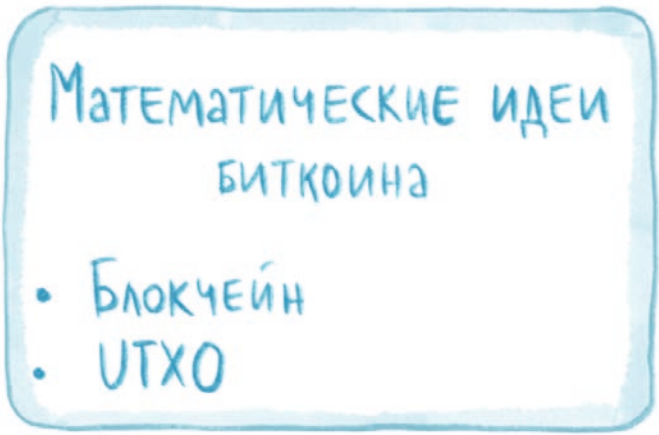
Задача решается лишь перебором, каждому времени соответствует своя добавка, поэтому злоумышленник не может поменять время. На нахождение добавки к определённому времени требуются огромные затраты времени, а проверка очень простая и происходит мгновенно (нам не нужно перебирать варианты). В случае биткоина в хэше, который представляет собой 16-ричное число, должен быть 21 ноль, т. е. чтобы надеяться с приличной вероятностью получить нужные нули в начале, в среднем необходимо 16^{21} хэшированій — огромное число.

СМИО
ПРЕСС

Часть 3

КОНЦЕПЦИЯ И KNOW-HOW БИТКОИНА

В этой части мы разбираем идеи, которые были добавлены именно создателями биткоина. Многих из них мы касались ранее, но сейчас обсудим всё максимально подробно.



СМИО ПРЕСС

Глава 13
СОЗДАНИЕ БИТКОИН-АДРЕСОВ

В главе 11 мы научились создавать приватный и публичный ключи для шифровки-дешифровки с помощью эллиптических кривых. Напомним, приватный ключ — это просто большое число, которое означает, сколько раз мы суммируем базовую точку. Публичный ключ — точка, получившаяся после этого суммирования. Базовая точка известна всем и является частью стандарта.

Познакомимся с термином «адрес». Вкратце, адрес — это просто захэшированный и представленный в более удобном для человека виде публичный ключ (на самом деле всё немного сложнее, но пока мы будем считать так).

«Адрес» — очень неудачный термин при попытке объяснить его суть применительно к биткоину. Но этот термин был выбран создателями, и теперь все вынуждены им пользоваться.

На самом деле адрес — это просто код (шифр) или замок, который вы ставите на «сейф», как мы обсуждали в главе 4.

Нужно напомнить, что в системе биткоина нет «перевода денег», так как в принципе отсутствует место, куда переводить (такое место мы и называем адресом в обычной жизни). Вы фактически добавляете в конец перевода добавку-шифр, которую может расшифровать только человек, обладающий соответствующим приватным ключом. В первой главе мы описали это как навешивание дополнительного замка на сейф или применение «ключе-замка».

Далее мы будем употреблять термин «адрес» лишь из-за устоявшегося термина, однако всегда нужно помнить, что ни адресов, ни счетов не существует. Мы лишь даём возможность другому человеку дешифровать зашифрованные нами сообщения.

Итак, что же такое адрес в системе биткоина? Как мы помним, у нас существует приватный и публичный ключи. Приватный ключ позволяет расшифровывать информацию и по сути является доступом к нашим деньгам. Публичный ключ обычно общедоступен и нужен для зашифровки сообщения. Но система биткоин идёт дальше и при зашифровке вообще не даёт доступа к публичному ключу.

Когда мы даём человеку «адрес», мы лишь говорим: «Укажите, что получить эти деньги может только тот, чей шифр будет давать хэш, который я вам дам».

СМИО ПРЕСС

Или можно сказать так: «Укажите, что получить эти деньги может только тот, кто предоставит такой набор цифр, хэш из которых будет давать число, которое я вам дам».

То есть мы не разглашаем публичный ключ — мы публикуем лишь его отпечаток (хэш). И в этой главе мы научимся получать такой отпечаток.

Рассмотрим всю схему максимально подробно. Рассмотрим ещё раз термин «транзакция». Мы можем считать это переводом со счёта на счёт. Раньше мы рассматривали «транзакцию» как «ключе-замок» Мы открываем наш замок нашим ключом и закрываем чужим замком.

Мы хотим перевести деньги Глебу.

Итак, у Глеба есть приватный ключ. Из приватного ключа он получает публичный ключ. Из публичного ключа он вычисляет хэш (рис. 89) (позже обсудим это подробнее на примере, пока схема).

Рис. 89

Как получить публичный ключ из приватного, мы подробно обсуждали в главах 10 и 11. Приватный ключ — это просто большое число. Публичный ключ получается сложением заранее известной и общей для всех исходной точки определённое количество раз, равное приватному ключу.

Как получить хэш из любых данных мы также обсуждали неоднократно. Мы как бы вырезаем из любых данных уникальный хэш. Это можно сравнить со взятием отпечатка пальца или с тем, как выжимают сок.

Захэшировав публичный ключ, мы получаем замок. Тот самый замок, о котором мы говорили с первой главы.

Этот хэш своего публичного ключа Глеб отдаёт нам, чтобы мы перевели ему деньги. Этот хэш называется «адресом» и, передавая его, Глеб говорит «переведите деньги на этот адрес».

В прошлом нам перевели деньги (согласно нашей терминологии «на сейфе наш замок»), и теперь нам нужно перевести деньги Глебу. То есть «составить транзакцию Глебу» означает, что нам нужно открыть наш замок своим ключом и закрыть замком Глеба.

СМИО ПРЕСС

Рассмотрим максимально подробно суть транзакции вплоть до компьютерного кода.

Схематично рисунок можно представить так.

После прошлой транзакции на сейфе висит наш замок — это хэш нашего публичного ключа. И сейчас нам нужно создать ключе-замок, где наш ключ и замок Глеба, который нам предоставили.

Рис. 90

Начнём с нашего замка (рис. 90).

Мы имеем дело с компьютерным кодом, поэтому замок — это набор 0 и 1.

Как пример (в двоичном виде):

01110101001010001100011001100100110011011100001101001100010111001110100000001001011101111100011101011011010001000110100110011011111000100110100101001110001100000010111000100001010110111001000010

Именно так он представлен в компьютере. Более краткая запись в 16-ричном виде (hex):

007528c664cdc34c5ce809778eb688d32f89a538c0b885b722

Как мы сказали, нам нужно открыть замок ключом и запереть чужим замком, то есть создать транзакцию (она же ключе-замок). При этом нам нужно сделать так, чтобы ключе-замок нельзя было сломать.

Первое, с чего начинается ключ, — это код, который при хэшировании даёт код замка. То есть наш замок — это хэш начала нашего ключа. Как уже говорилось, когда мы давали замок прошлому владельцу сейфа, чтобы он запер сейф на замок, в качестве замка был дан хэш нашего публичного ключа.

Таким образом, наш ключе-замок начинается с нашего публичного ключа. Напоминаем, у нас есть приватный ключ для шифровки сообщений и публичный для расшифровки (как в доказательстве ЭЦП, которое было описано в главе 9).

Давайте ещё раз повторим идею рис. 89. Чтобы получить наш «замок», мы хэшируем наш публичный ключ (рис. 91).

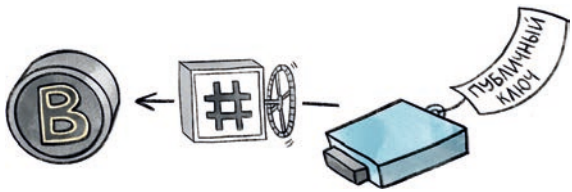


Рис. 91

То есть если у нас замок — это **01...10**, который мы указали выше, то начало ключа должно быть таким, чтобы хэш из него давал именно этот замок — **01...10**. Как мы помним, по хэшу нельзя восстановить исходные данные. И если в сети

СМИО ПРЕСС

все знают о нашем замке (хэш публичного ключа), то никто не может знать начало ключа (сам публичный ключ).

На нашем сайте вы можете видеть, как публичный ключ превращается в адрес. В данном случае, исходный публичный ключ — это:

0450863AD64A87AE8A2FE83C1AF1A8403CB53F53E486D8511DAD8A04887E5B23522C
D470243453A299FA9E77237716103ABC11A1DF38855ED6F2EE187E9C582BA6

После хэширования он как раз и даст нам адрес:

007528c664cdc34c5ce809778eb688d32f89a538c0b885b722

Далее, следующая часть транзакции состоит из подписи нашего замка нашим приватным ключом.

Напомним, что такое подпись. Наши данные (любые данные) — это по сути число, которое мы можем возводить в степень, брать остаток и так далее. Приватный ключ — это правила (в какую степень возвести, какой остаток взять и так далее), по которым мы из исходного числа (наши данные) получаем другое число (нашу подпись).

Превратить это в исходное число можно с помощью публичного ключа (это тоже набор чисел, и с помощью них (возведение в степень и так далее) мы получаем исходное число.

Мы берём замок (хэш публичного ключа) и шифруем (преобразовываем) его нашим приватным ключом. Таким образом из хэша публичного ключа мы получаем зашифрованный хэш публичного ключа (рис. 92).

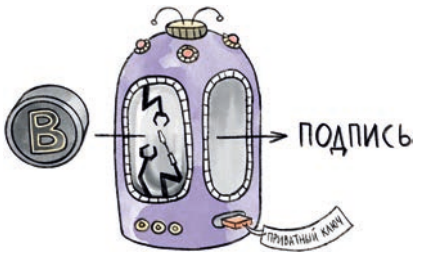


Рис. 92

Помните, что нашим публичным ключом мы можем расшифровать эти данные. Следородно, тот, кто обладает публичным ключом, может превратить подпись обратно в наш замок (хэш публичного ключа) (рис. 93).

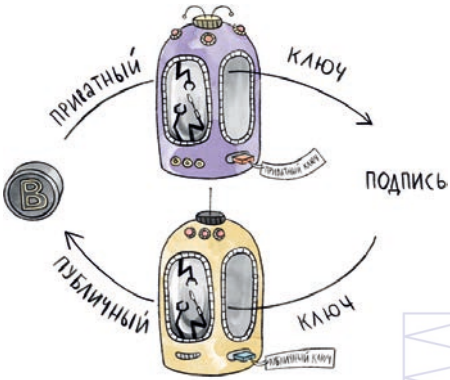
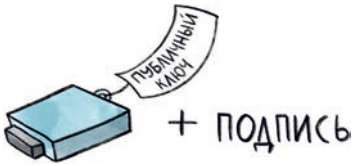


Рис. 93

Рис. 94

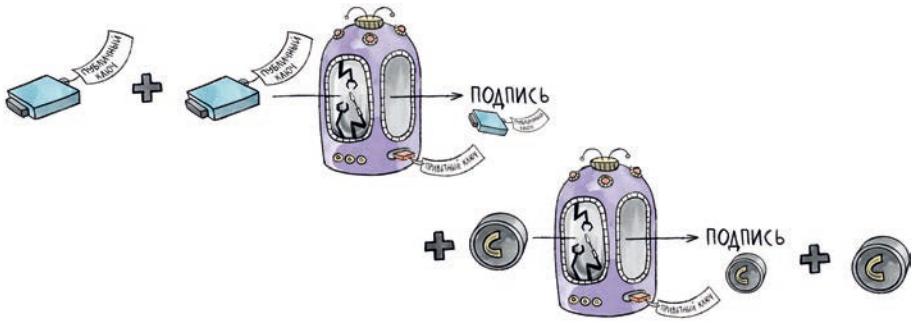


Теперь эти данные — публичный ключ и подпись — мы соединяем. Просто один идёт за другим (рис. 94).

Так как это ключ, то нам нужен замок. Мы берём замок у того, кому мы переводим деньги. Далее мы подписываем его нашим приватным ключом.

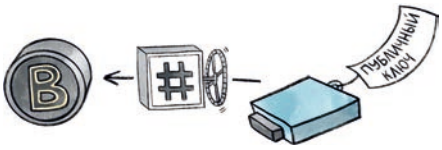
Получается такая схема (рис. 95): идущие друг за другом числа (коды) — публичный ключ, хэш публичного ключа, зашифрованный приватным ключом, замок (хэш чужого публичного ключа), зашифрованный нашим приватным ключом, и хэш чужого публичного ключа.

Рис. 95



Для подробного объяснения мы включили сюда лишний элемент — подпись нашего публичного ключа. На самом деле, он не нужен. Достаточно записать публичный ключ, а также подпись замка С. Если вы можете расшифровать подпись замка с помощью публичного ключа — это достаточное доказательство.

Рис. 96



Всё. Это и есть наш ключе-замок или транзакция. Теперь обсудим, как проверяется его валидность. После того, как узел получил данную транзакцию, он проверяет, является ли замок (выход прошлой транзакции) хэшем первой части нашего ключе-замка (рис. 96).

Если так и есть — мы доказали, что обладаем публичным ключом. Далее узел берёт подпись (зашифрованный замок) и расшифровывает его публичным ключом. Здесь давайте вспомним: мы брали публичный ключ, шифровали его приватным ключом и получали подпись.

Напоминаем: подпись в нашей терминологии — это шифр, получившийся после шифровки приватным ключом.

СМИО ПРЕСС

Мы предоставляем публичный ключ первым элементом нашего замка. И узел теперь его знает. Он теперь может взять второй элемент ключа — подпись публичного ключа и расшифровать его публичным ключом, получив публичный ключ. Получается, у нас намертво скреплены два элемента — публичный ключ и его подпись, потому что никакие другие данные не могут после расшифровки публичным ключом дать публичный ключ (рис. 97).

Рис. 97

Третий элемент — это подпись чужого замка, который нам дали. Это указано на рис. 95. Теперь узел проверяет, что данный замок в транзакцию вставил именно тот, кто обладал приватным ключом к первому элементу. Как мы показали на рис. 95, подпись замка С — это шифровка приватным ключом. Далее с помощью этого же публичного ключа (элемента 1) узел расшифровывает элемент 3. И получается элемент 4. Так мы намертво скрепляем элемент 4 и элемент 1. Это и есть то самое соединение ключе-замка, доказывающее, что никто не сможет сломать его и к нашему ключу присоединить другой замок (рис. 98).

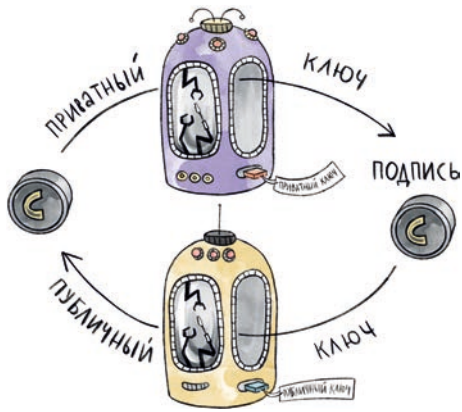


Рис. 98

Получается, что наш ключе-замок может создать только тот, кто обладает приватным ключом, а проверить его может любой узел. Также никто не сможет взять наш ключ и пристроить к нему чужой замок. Возможно, читатель помнит, в главе 4 мы говорили, что наш ключе-замок невозможно сломать.

Каждый элемент этой транзакции — это просто числа, т. е. 0 и 1 (рис. 99).

Рис. 99

Для подробного объяснения мы включили сюда лишний элемент. Другими словами, элемент номер 2 необязателен. Доказательством того, что у нас есть приватный ключ, служит элемент 3, подписанный «замок» следующего владельца, который можно расшифровать элементом 1. То есть наш ключ состоит из публичного ключа, зашифрованного замка, который можно расшифровать этим самым публичным ключом (элементом 1) и замком следующего владельца (на схеме замок С) – хэшем его публичного ключа, который уже будет началом его ключа.

Двоичный вид слишком длинный, сама транзакция будет выглядеть примерно так (в hex виде), где каждые 4 двоичных числа заменяются 16-ричным числом от 0 до 15 (10 – a, 11 – b, 12 – c, 13 – d, 14 – e, 15 – f).

0100000001592f3619bf215249ef7409a4a1192ef46f17d81d8b0ca06f1c1abcba
03444cb4000000006a4730440220796148d97d54803c8e9bb6be9f e7947ff24b
3077706d5fe1608f1a2caf8ccc23022047236ed76bcad41dbff959f959469db315
10d56ab2e8633fab73bcf20986a3a5012103433f46b3c8c92ef3a619a617b5a4d
d8e55c666f66391d0b692d5e0939a04c118ffffffff022047160000000001976a9
14fa4aa414a21e4b7251bc9c88fdd899a80c00342a88ac498a8d050000000019
76a914f8a8354d0ae73c40bd5215cdc7208bf3f75d4dea88ac00000000

В реальности всё ещё немного сложнее — сами команды записываются в замок, но это мы будем разбирать в следующих главах. Пока важно понять концепцию транзакции.

Теперь перейдём к более подробному описанию создания адреса (хэша).

На странице «Создание биткоин-адресов» мы начинаем с создания приватного ключа, но пока предположим, что мы уже создали приватный ключ и уже создали с помощью него публичный. Публичный ключ, как мы помним, — это две координаты результирующей точки. Записываем их одну за другой:

0450863AD64A87AE8A2FE83C1AF1A8
403CB53F53E486D8511DAD8A04887E
5B23522CD470243453A299FA9E77237
716103ABC11A1DF38855ED6F2EE187
E9C582BA6

Как мы помним, в работе с биткоином уровень шифрования максимальный. По публичному ключу фактически невозможно восстановить приватный. Но недостатком публичного ключа является то, что он слишком длинный.

Сокращая длину публичного ключа, как описано ниже, одновременно добавляют дополнительный уровень шифрования.

СМИО ПРЕСС

Получив публичный ключ, мы превращаем его в адрес следующим образом.

ЭТАП 1 — ХЭШИРУЕМ ПУБЛИЧНЫЙ КЛЮЧ С ПОМОЩЬЮ SHA256:

```
32511e82d56dcea68eb774094e25bab0f
8bdd9bc1eca1ceeda38c7a43aceddce
```

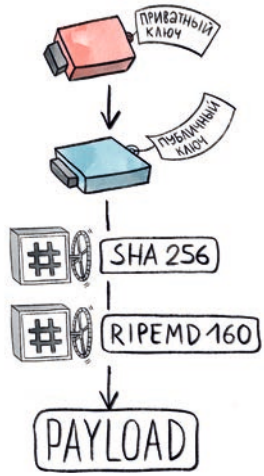
Напоминаем, это нужно, чтобы усложнить поиск приватного ключа, — мы не даём публичный ключ, мы даём только его хэш.

ЭТАП 2 — ЕЩЁ РАЗ ХЭШИРУЕМ ТО, ЧТО ПОЛУЧИЛОСЬ, НО ДРУГИМ МЕТОДОМ. НЕ SHA256, А RIPEMD160:

```
7528c664cdc34c5ce809778eb688d32f8
9a538c0
```

То есть мы ещё более (просто невероятно) усложняем возможность поиска исходного кода ключей.

Результат этого хэша меньше размером, чем у SHA256.



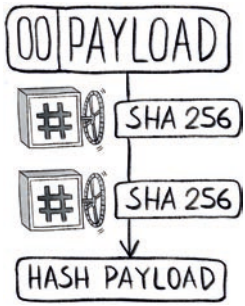
Этот получившийся хэш мы называем PayLoad.

ЭТАП 3 — ДОБАВЛЯЕМ 00 В НАЧАЛО
(это идентификатор сети, они могут быть разные)

```
007528c664cdc34c5ce809778eb688d32
f89a538c0
```

ЭТАП 4 — СОЗДАЁМ КОНТРОЛЬНУЮ СУММУ
(хэшируем ещё два раза)

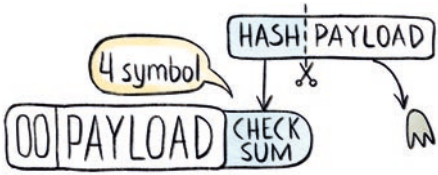
```
b885b7225b370e7ff27ee0afb4f89b52b8
675d5dc342d63de3abe7535f86cadb
```



ЭТАП 5 — СОЗДАЁМ КОНТРОЛЬНУЮ СУММУ —
этап 2 (из того, что получилось, берём первые 8 символов)

```
b885b722
```

Это контрольная сумма нашего PayLoad из третьего этапа.



Давайте вспомним, для чего нужна контрольная сумма. И что это означает. В главе о хэшировании мы приводили пример передачи байта (8 битов), где 7 битов содержат информацию, а восьмой — контрольный — в нём содержится чёт/нечёт единиц. Если в одном из 7 байтов будет ошибка (0 вместо 1 или 1 вместо 0), мы поймём это по несоответствию чёт/нечёт единиц и контрольной суммы.

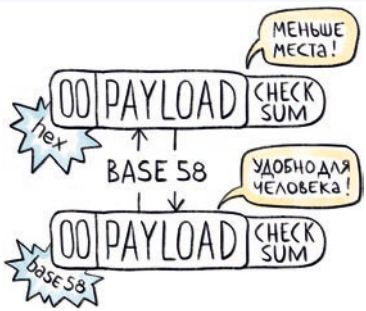
Здесь то же самое — контрольная сумма нужна для того, чтобы при вводе или пересылке сообщения в наш код не закралась ошибка. Если какой-то символ будет введён неправильно, то контрольная сумма не совпадёт.

ЭТАП 6 — СОЕДИНЯЕМ КОНТРОЛЬНУЮ СУММУ С РЕЗУЛЬТАТОМ ЭТАПА 3

```
007528c664cdc34c5ce809778eb688d32
f89a538c0b885b722
```

ЭТАП 7 — ПЕРЕВОДИМ В БОЛЕЕ УДОБНЫЙ ДЛЯ ЧЕЛОВЕКА ФОРМАТ BASE58

```
1BgUuYGea1GhDk7LZBGijzr3sp9Rz4y
Qs3
```



Мы не хотели делать на этой странице реальные адреса биткоина, чтобы вы не ошиблись и не перевели никуда деньги (или чтобы нельзя было перевести по этому адресу). Это лишь примеры, как работает система, но некоторые символы изменены.

Это сложная глава, и мы предлагаем несколько заданий на понимание. Если вы их смогли выполнить, то можно считать, что усвоили содержание этой главы.

1. Почему мы делаем замок в виде хэша публичного ключа?
2. Нарисуйте схему из чего состоит транзакция.
3. Опишите схему проверки узлом транзакции.
4. Опишите процесс создания адреса, начиная с создания приватного ключа.

Когда мы просим совершить перевод, мы не сообщаем наш открытый ключ, мы сообщаем его хэш (причём двойной). Таким образом, мы вообще не разглашаем никакой секретной информации.

Зачем двойной SHA256?

Мы знаем, что расшифровать хэш невозможно. Почему же мы хэшируем два раза? Так мы перестраховываемся от возможного аналитического решения SHA256 путем удлинения сообщения.

Двойной хэш не спасёт, если будет найдена возможность взломать SHA256, потому что если hash(x) = hash(y), то hash(hash(x)) = hash(hash(y)). Но против атаки «удлинения сообщения» это спасёт. Если злоумышленники знают длину входных данных хэша (а мы знаем, т. к. знаем длину публичного ключа), то могут заставить хэш-функцию взять определённую часть внутреннего состояния, добавив секретную строку к значению хэша. Чтобы работа злоумышленника не была облегчена, биткоин вычисляет хэш дважды. Подробнее смотрите в Интернете: «Атака удлинением сообщения».

СМИО ПРЕСС

На нашем сайте вы также можете изучить способ превращения приватного ключа из неудобного формата записи обычного числа в достаточно удобный WIF-формат. Мы не приводим его в книге ради уменьшения её объёма.

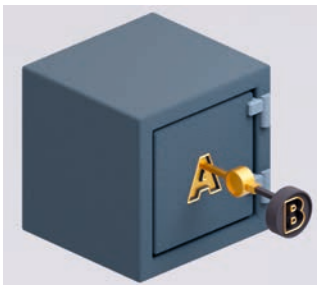


Рис. 100

Давайте ещё раз обсудим процесс «навешивания» ключе-замков. Итак, у нас есть сейф, который, как мы помним, создаёт майнер (подробнее о создании «сейфа» в главе 19). И в нём уже есть ключе-замок. И выход из этой транзакции — это замок B (рис. 100).

Здесь замок, напоминаем, — это хэш публичного ключа. Владельцем ценности считается обладатель ключа к этому замку. Говоря другими словами, это хэш публичного ключа, приватный ключ которого известен владельцу. Далее, если владелец хочет перевести деньги, он создаёт транзакцию.

Ещё раз напомним. Транзакция (передача денег) состоит из двух элементов: ключа, который «отпирает» прошлый замок, и нового замка.

Ключ состоит из двух элементов: 1) публичный ключ, который «крепится» к прошлому замку, — и только он подходит к этому замку, т. к. только из него, хэшируя вышеуказанным способом, можно получить прошлый замок; 2) подпись этим самым публичным ключом нашего нового замка.

И один элемент в транзакции нельзя отделить от другого. И только обладание приватным ключом, из которого можно получить публичный ключ, позволит создать транзакцию.

Только обладатель приватного ключа мог создать транзакцию, при этом сам приватный ключ не разглашается — в этом состоит суть доказательства с нулевым разглашением.

Создав этот ключ, владелец замка кладёт его в мемпул — место, откуда его могут забрать майнеры, которые пытаются составить блок (об этом в главе о майнинге).

Далее, после перевода, уже новый владелец замка может переводить кому-то деньги. У него есть приватный ключ, а значит, он может произвести транзакцию. И так далее. В итоге получается цепь ключе-замков (рис. 101).

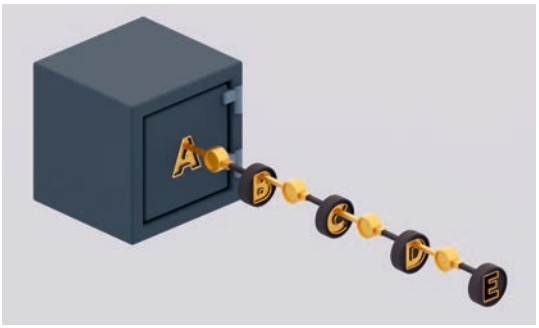


Рис. 101

СМИО ПРЕСС

Фактически, как мы говорили в главе 4, мы видим цепь «ключе-замков» или «транзакций» от первоначального замка на сейфе. Тот, кто знает приватный ключ, из которого можно сделать публичный ключ, из которого можно сделать хэш, который будет совпадать с кодом, тот и считается обладателем ценности.

Ещё более подробно мы поговорим о передаче ценности в следующей главе. В этой главе мы рассмотрели лишь идею, в следующей главе мы рассмотрим техническую реализацию. В том числе — техническую реализацию проверки каждым узлом транзакции.

СМИО
ПРЕСС

Глава 14
ОБЩАЯ ТЕОРИЯ ПЕРЕДАЧИ ЦЕННОСТИ В КРИПТОВАЛЮТАХ

Цель данной главы — объяснить, как в принципе можно передать ценность, которая выражается только в электронном виде. И как узлы сети могут узнать об этой передаче. Мы касались этой темы в первой части книги. Однако сейчас мы разберём техническую сторону данной проблемы.

Перейдём к самой сути и способу передачи ценности. Когда мы передаём что-то физическое, проблем с передачей не существует. Мы передаём это из рук в руки. Если у нас на руках кусок золота, он принадлежит нам. Если мы его передали, он принадлежит тому, кому мы его передали.

Перейдя на бумажные деньги, мы по-прежнему обладаем чем-то физическим, мы не можем его «размножить» по собственному желанию, но его может размножить финансовый центр. По сути, бумажные деньги — это просто право требования! Центробанк выпускает право требовать, продаёт его, и мы называем это деньгами. Любой субъект экономической деятельности может выпустить что-то подобное, и тогда это называется векселем.

Обыкновенная расписка тоже может являться деньгами. Не возникает проблем, чтобы понять, кому она принадлежит. У кого она на руках, тот ею и обладает.

Когда появились электронные деньги, проблем тоже не возникло. Это просто данные, у кого на данный момент находится что-то физическое. По сути, это просто записи в бухгалтерской книге, в которой чётко указано, сколько принадлежит Василию, а сколько Алёне.

Если вы хотите передать деньги, вы обращаетесь к центру и просите снять деньги с вашего счёта и зачислить на счёт другого человека. Центр верификации просто списывает у вас средства и зачисляет их на другой счёт. Ответственность за эти действия лежит на нём.

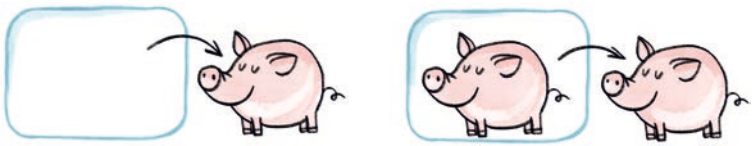
Это переход на другой уровень абстракции, но все к этому привыкли. При переводе со счёта на счёт фактически не происходит перевода денег. Банк убирает цифры с одного счёта и добавляет их на другой. По сути, происходит уничтожение, а после этого возникновение ценности из ниоткуда.

Это выглядит странным, но все к этому так привыкли, что не замечают. Концепция биткоина выглядит сложнее только на первый взгляд. На самом

СМИО ПРЕСС

деле передача ценности в экосистеме биткоина куда более логична и непротиворечива. Происходит именно передача.

Так как у нас нет финансового центра, которому мы можем доверять, невозможно уничтожить и создать заново некие записи. Нужно переводить деньги напрямую, без использования доверенной третьей стороны. Как это сделать, чтобы не было возможности копирования информации? Ведь если ценность — это последовательность нулей и единиц внутри компьютера, её можно скопировать.



Первый шаг на этом пути — это определить, что такое ценность и что такое принадлежность. Мы можем ввести определение, что «обладать» ценностью — это иметь возможность расшифровать что-то. Давайте вспомним путь, который мы прошли в двух прошлых главах.

Согласно принципу доказательства с нулевым разглашением, если мы можем расшифровать, значит, у нас есть ключ для расшифровки. Принцип ЭЦП говорит о том, что если вы можете расшифровать моим публичным ключом, значит, зашифровал его я — тот, кто обладает приватным ключом. Мы можем ввести определение:

деньги принадлежат тому, кто зашифровал (или, точнее, чьим ключом зашифровали).

Для начала попробуем объяснить схему, максимально её упростив. Пока не будем обсуждать, как появилась ценность, — предположим, кто-то принёс на главную площадь города сейф и запер его на особый кодовый замок. А ещё приклеил на сейф большую бумагу. Чтобы знать владельца сейфа (в том числе его содержимое), владельцу нужно доказать, что сейф закрыт на его ключ, а открывать сейф не обязательно!

Чтобы отпереть замок, нужно ввести код. Или, как мы объясняли более «физическими» терминами, в замок можно вставить ключе-замок и отпирать сейф можно будет уже только другим ключом.

На бумагу заносится информация о всех перепрограммированиях замка (все операции навсегда остаются в блокчейне).

При запирании замка вы вводите сумму, которая там находится, и хэш публичного ключа того человека, которому принадлежат деньги.

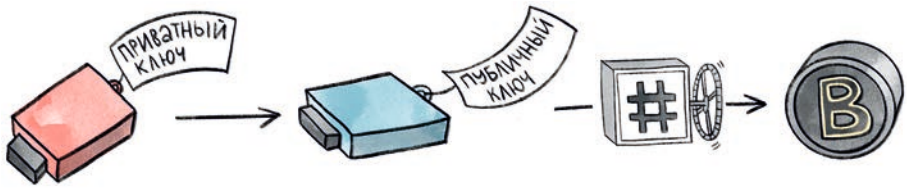
Итак, в кодовый замок введены данные:

Сумма в сейфе
Хэш из публичного ключа

СМИО ПРЕСС

Напомним единственный путь получения хэша публичного ключа (рис. 102):

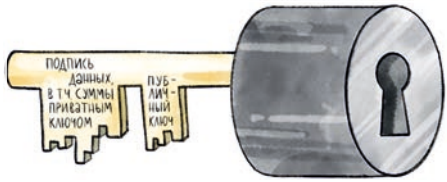
Рис. 102



В данном случае, нужно ввести строку, подписанную вашим приватным ключом (т. е. подпись этой суммы), и ваш публичный ключ. Мы очень подробно разобрали это в прошлой главе. Можно лишь добавить, что в подпись мы можем добавить и сумму передачи.

Вот примерно так будет выглядеть наш ключ (рис. 103).

Рис. 103



Подпись приватным ключом, публичный ключ

Ключ открывает замок и там остаётся. На нём уже прикреплена скважина нового замка, куда можно будет вставить ключ того, на кого настроена скважина нового замка.

Другое описание: представим, что мы вставляем не ключе-замки друг в друга, а добавляем зашифрованную надпись на сейфе с именем нового владельца[1]. Чтобы добавить новую надпись, нам нужно расшифровать предыдущую и добавить свой шифр.

При добавлении надписи, сейф проверяет, что ваш публичный ключ подходит к хэшу, а также, что ваш публичный ключ расшифровывает вашу же подпись.

То есть он берёт две полученные переменные, из публичного ключа берёт хэш и сверяет его с хранящимся — если ОК, то этим публичным ключом расшифровывает подпись. Если получилась сумма (на самом деле там большое количество данных, мы упрощаем, разумеется), значит, всё в порядке, ключ сработал. Мы тоже это подробно разобрали в прошлой главе.

На бумагу наносятся данные о переводе.

Теперь только обладатель нового шифра может его открыть, а это означает, что он теперь обладает этой ценностью.

Разумеется, любые упрощения и аллегории условны. Идея ключе-замка наиболее точно отражает идею передачи ценностей: у нас ничего не пропадает

[1] *Это напоминает переходящий спортивный кубок, на котором гравируют имена тех, кто им обладал. Для того чтобы получить кубок, надо много поработать!*

СМИО ПРЕСС

в блокчейне. У нас есть вся история ключе-замков, один вставлен в другой и всегда можно понять, через какие «руки» шла передача ценности.

Можно это представить как наложение нового кодового замка. При этом старый замок перестаёт действовать. До него просто «не достать».

Но не достать до него не потому, что он как-то склеен или приварен, ведь это просто нули и единицы, и мы можем разрезать код в любом месте. Защита состоит именно в том, что для разрезания-добавления нужно обладать возможностями майнеров.

И тем не менее, это очень грубое приближение (мы серьёзно его уточним в конце главы), и у читателя должно возникнуть множество вопросов. Самый главный из них: откуда вообще возникает ценность?

Вкратце мы можем ответить следующим образом.

Она возникает при «майнинге» как плата за решение задачи. Помните главу 10 — Proof of Work? Доказав, что он выполнил работу, майнер получает вознаграждение в качестве зачисления денег «из ниоткуда». Очень подробно мы рассматриваем принцип майнинга в главе 19. Это просто общий консенсус — если кто-то решил задачу, то возникает право на шифровку («получение») монет своим секретным ключом. Так возникает первая запись и ценность. Все биткоины были когда-то добыты таким образом.

Второй вопрос — как добиться того, чтобы только тот, кто действительно может расшифровать сообщение — только он мог что-то писать на бумаге? Ведь иначе нашу бумагу быстро «расчирикуют» хулиганы.

Право писать что-то на бумаге возникает только у майнеров. Только тот, кто решил задачу, наделяется таким правом. Контроль за этим правилом лежит на узлах сети. Помните наш рассказ о задаче византийских генералов? Тот, кто смог подобрать правильный код, начинает рассылать всем время атаки. Каждый может проверить, что код действительно подобран верно. Здесь, если вы хотите передать ценность, то пишете об этом во все узлы сети. Они проверяют, что ваш код действительно верен, и распространяют его дальше.

Все запросы на передачу попадают в очередь ожиданий (в экосистеме биткоина это называется мемпул). Далее они стоят в очереди к майнерам.

Рассмотрим проблему записей. Помните, в первой главе мы рассказывали про остров Яп и их денежную систему, когда ценностью являются большие камни, привезённые с далёких островов? Даже утопив камень по дороге к острову, его владелец не лишался его — просто по всеобщему мнению, камень лежащий на дне океана, принадлежал ему. Так же и в системе биткоина: по всеобщему мнению (консенсус) майнер получает вознаграждение, теперь «камень» принадлежит ему и он может его передавать.

Подойдём к проблеме сейфа с другой стороны. Кирилл, получив зашифрованную фразу «1000$», может расшифровать её своим ключом, а потом

СМИО ПРЕСС

зашифровать ключом Насти. И Настя будет ожидать, что её баланс вырастет на 1000$. А потом зашифровать ключом Оли, а потом зашифровать ключом Кати и так далее. Что же делать?

Здесь приходит know-how в системе биткоина под названием UTXO (unspent transaction output) — выход неизрасходованных транзакций.

В чём суть? Как мы упоминали, в обычном банке у нас есть реестр: владелец счёта — баланс на его счёте. Здесь этого реестра не существует. Есть лишь реестр транзакций, и ваш баланс определяется его суммой. В криптовалютах мы именно «передаём» ценность.

В физическом мире житель острова Яп обладал камнем (даже если не обладал им физически) и передавал его на общем собрании жителей острова, просто говоря: «Камень теперь принадлежит ей». Таким же образом Боря, обладая возникшей из ниоткуда ценностью (напоминаю, принцип возникновения ценности мы подробно рассмотрим в главе о майнинге), передаёт её, говоря всем: «Я передаю её». И все узлы сети записывают «Боря передал ценность Насте». Никто не записывает изменение баланса — просто в конец ценности (а именно, записи, которую Боря передаёт) ставится информация, что Боря её передал.

То есть под фразой: «На кошелёк Бори пришли биткоины» мы подразумеваем фразу: «В UTXO (реестре сделок) появились записи, в конце которых шифр, которые можно расшифровать с помощью секретного ключа, доступного Боре».

При этом вы можете часть денег дать одному адресату, часть денег — другому. Это как будто разломить камень и дать разные части разным людям. То есть на общем собрании острова Яп вы говорите: «Тот камень в океане я ломаю на 3 части — 10% отдаю Арсению, 20% — Алексею, 70% — Насте».

Если на острове есть реестр камней, то он выглядит следующим образом (рис. 104):

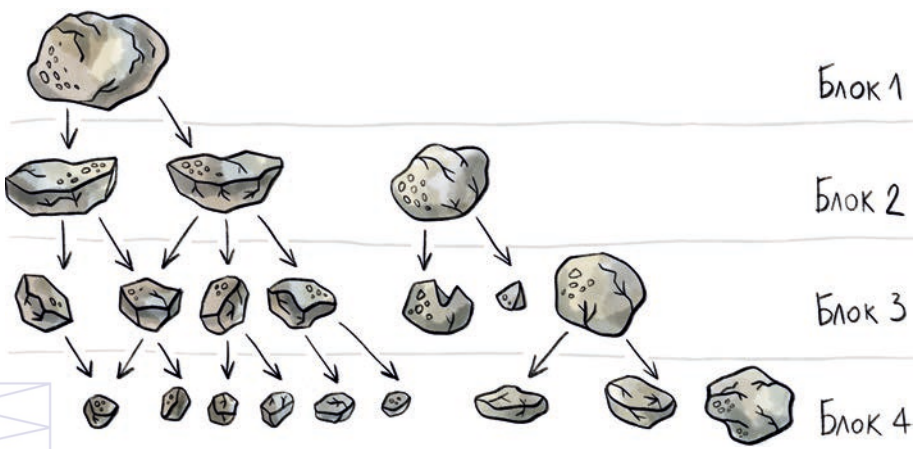


Рис. 104

СМИО ПРЕСС

Заметим ещё раз, что тут нет балансов, т. е. это не реестр владений каждого отдельного человека, а реестр камней. Каждый камень, как переходящий спортивный кубок, на котором написаны имена всех его обладателей!

Также отметим, что на каждом блоке у нас появляется камень.

Очень важно отметить, что информация обо всех транзакциях есть в конкретном блоке. Блок — это и есть набор транзакций. Кто-то может отдать только часть камня. Кто-то может объединить куски камней (на самом деле любые куски любых камней).

Это очень важный и принципиальный момент — на каждом собрании жителей острова они создают новый блок согласно своему консенсусу. Например, сегодня Александр получил право на новый камень, привезённый с далёкого острова. А этот камень разделили на три части.

В нашем случае камни — это коды, зашифрованные публичными ключами. Это можно представить, как будто на каждом новом блоке стоит печать.

Надеюсь, вы поняли аналогию и логику наших действий. Разумеется, мы слегка изменили систему жителей острова Яп, чтобы приблизить её к логике биткоина.

Этап	Действия жителей острова Яп	Действия держателей биткоина
Этап подготовки	Жители собирают друг у друга заявки и думают, хотят ли передать часть своих камней кому-то другому	Держатели биткоина думают и составляют заявки на передачу транзакций, которые у них есть (или частей транзакций). Все заявки добавляются в MemPool
Этап сбора	Один из жителей привёз на остров камень, что даёт сигнал на сбор жителей острова. Согласно правилам острова, собрание жителей происходит только, когда на остров привозят новый камень. Тот, кто привёз камень, получает право на выступление на острове — он же ведёт реестр. Ему же жители дают дары за то, что он впишет их переводы в реестр. Дары — это кусочки камней, которые они готовы передать выступающему	Один из майнеров смог найти такую добавку, что хэш из заявок + добавка имеет 20 нулей в начале (на конец 2022 года нужно 20 нулей). Этот майнер получает определённое количество биткоинов (6.25 биткоина на конец 2022 года), а также на комиссии за перевод. Каждый держатель сам определяет размер комиссии, который он хочет заплатить

СМИО ПРЕСС

Этап	Действия жителей острова Яп	Действия держателей биткоина
Этап включения в блок. Как показано на рис. 104, блок — это запись о переходах	Жители разбиваются самостоятельно по количеству даров, которые они готовы отдать выступающему (тому, кто привёз камень). Каждый выступает по очереди, показывает свою печать и говорит, кому передаёт камни с его печатью. Жители вместе обходят камни, стирают старую печать и ставят новую. Если нужно дать часть камня — стирают печать, ломают камень и ставят на обломке печать того, кому раньше принадлежал весь камень. При каждом стирании/установке печати бывший владелец даёт выступающему в дар кусок камня, согласованный заранее. Известно, что за встречу удаётся зафиксировать только 100 передач камней	Заявки разбиваются по размеру вознаграждения, размер блока ограничен, поэтому в него включены только заявки с максимальным размером комиссии. Майнер обновляет все записи, добавляя к каждой запирающие и отпирающие коды, которые передали ему владельцы записей. Эту схему мы опишем подробнее дальше
Этап распространения информации	Каждый житель переписывает себе новый реестр камней и хранит его у себя	Майнер передаёт нодам данные о блоке. Каждая нода может проверить исходную информацию о том что, майнер действительно выполнил нужную работу (Proof of Work), что каждый владелец транзакции действительно имел на неё право (смог расшифровать) и что данные переданы корректно

Это достаточно трудная для понимания таблица. Она демонстрирует очень необычный подход в передаче ценности и на самом деле содержит в себе несколько проблем, которые, скорее всего, не будут вам заметны, но которые мы обсудим далее.

СМИО ПРЕСС

Отдельно хотелось бы отметить, что весь консенсус достигается без голосования. Просто люди (в случае жителей острова Яп) или программы (в случае биткоина) следуют заранее выработанным правилам. Каждая транзакция проверяется независимо от других каждым человеком, право на создание блока проверяется также каждым элементом системы независимо.

Перейдём к сути транзакций в сети биткоина. Напоминаем, что в биткоине есть только транзакции. И любая транзакция должна иметь входы (это прошлые транзакции, которые пришли на наш кошелёк) и выходы — транзакции, которые мы отправляем с нашего кошелька.

Давайте ещё раз остановимся на этом подробнее. Представим, что Вася дал нам 2000 рублей, Петя дал 1000 рублей за выполненные работы. Мы хотим купить торт за 800 рублей и цветы за 1600 р.

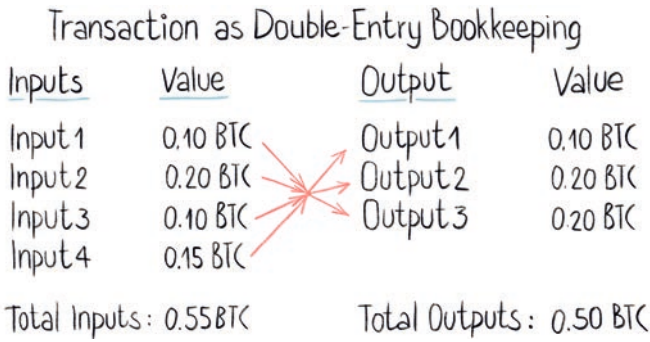
Тогда мы записываем следующую схему:

Наши входы:

1. Разбить 2000 на 1600 и 400.
2. Разбить 1000 на 400 и 600.

Наши выходы:

1. Отдать 400 + 400 = 800 за торт.
2. Отдать 1600 за цветы.
3. Вернуть себе 600 рублей.



И всё это — наша транзакция, и, кстати говоря, чем больше таких записей в одной транзакции, тем больше она стоит, т. к. мы платим майнерам за включение в блок (как вы помните) в зависимости от объёма нашего сообщения.

Когда Вася и Петя передавали нам транзакции, они вписали в свои транзакции наш адрес (надеюсь, вы помните из предыдущей главы, что адрес — это двойной хэш от публичного ключа).

Теперь в каждую транзакцию, которая отправлена на ваш адрес, вы записываете ваш публичный ключ, а потом подписываете транзакцию своим приватным ключом. Нужно отметить, что только в этот момент окружающий мир узнает о вашем публичном ключе.

СМИО ПРЕСС

То есть до этого в блокчейне был записан только адрес, и никто не знал ваш публичный ключ. И если до этого даже всей энергии Вселенной не хватило бы для расшифровки публичного ключа, то сейчас ваш публичный ключ известен и защита теперь осталась только в невозможности найти соответствие приватного и публичного ключа. Она достаточно надёжна, но адрес всё равно считается скомпрометированным.

К входящей транзакции владельцем предыдущего кошелька прикреплён ключ выхода. Вы можете добавить к её концу ключ входа, но только один раз. После этого транзакция считается погашенной и её уже нельзя использовать. Это очень важный момент. В этом суть системы UTXO. Входящая транзакция — как крупная купюра, с которой вы пришли в магазин и которая должна быть отдана кассиру полностью.

Ещё раз обратим внимание на очень важный момент: по сути транзакция — это не отправка на кошелёк или адрес, а постановка в конец предыдущей транзакции некоего кода.

Этап включения в блок

Как мы обсудили, майнер должен вставить транзакции в блок. Давайте ещё раз обсудим, что означает «открыть транзакцию». Суть состоит в том, что в конце каждой транзакции вписан некий код. Предположим, что в него вписано просто число 5. Наша задача написать скрипт, который после взятия и обработки транзакции, выдал бы «истина».

В языке биткоин зашит самый простой из возможных языков, так называемый стековый язык. Он был популярен в 60-е гг. XX века и на его основе работали самые сложные на тот момент вычислительные машины. Он очень ограничен и вставлен в биткоин сознательно, чтобы не было возможности взломать программу, которая на нём написана.

Мы стараемся рассказывать постепенно, «подтягивая» знания с разных сторон криптомира. Мы говорили, что у нас есть ключ и замок, что замок проверяет введённые данные. Так вот, в биткоине проверяет данные именно этот стековый язык, и мы очень подробно рассмотрим этот язык на примере.

Рис. 105

Стек — это структура, в которую можно положить что-то и достать из неё. При этом достать мы можем только «сверху», т. е. то, что положили последним, должно выйти первым. Это похоже на коробку или на детскую пирамидку.

Таким образом мы можем совершать операции только с верхними элементами.

СМИО ПРЕСС

Стековый язык очень простой. Он не позволяет совершать сложных расчётов, но зато очень надёжный.

Биткоин уже давно не является единственной криптовалютой. Многие криптовалюты пошли по пути усложнения внутреннего языка. Наиболее известный пример — эфириум.

Создатель эфириума разработал скриптовый язык, под названием «солидити», который является Тьюринг-полным и на его основе можно писать любые программы — так называемые полные смарт-контракты. Это привело к тому, что эти смарт-контракты стали постоянно взламывать и красть деньги. И если для эфира это обычное явление, то в биткоине его смарт-контракты пока никто взломать не смог.

Благодаря отсутствию циклов, каждая операция выполняется только один раз, и мы точно можем сказать, сколько времени займёт выполнение каждого скрипта.

Итак, обратим внимание на калькулятор 60-х гг. (называемый калькулятором с обратной польской записью) (рис. 106).

Рис. 106

Чтобы в нём сложить 120 + 150, нам нужно нажать 120, потом 150, потом +.

То есть операция сложения берёт два последних элемента стека, складывает их, а потом возвращает ответ 270 и кладёт его поверх стека.

Рассмотрим этот процесс в графическом формате (рис. 107):

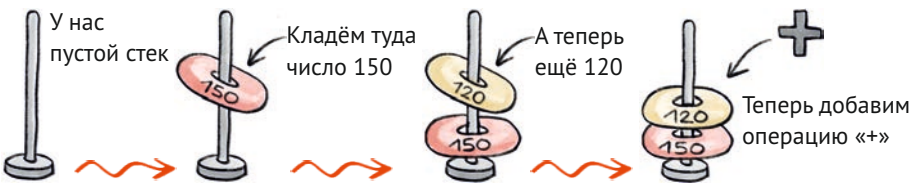
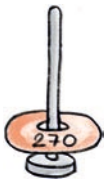


Рис. 107



Данная операция, как и любая другая стековая операция, может работать только поочерёдно с верхними элементами. Она берёт 120, потом берёт 150, выкидывает их из стека, а вместо выкинутых кладёт их сумму.

В этом суть работы стека и так работает стековый язык. Операции могут быть совершенно разные — к примеру, в языке биткоина это «взять хэш» или «взять верхний элемент и попытаться расшифровать им нижний» и так далее.

СМИО ПРЕСС

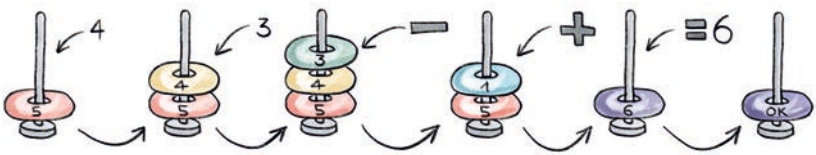
Представьте, что у нас есть набор готовых операций, которые будут поочерёдно применены к стеку. К примеру: 4, 3, –, +, =6. Этот список означает «положить в стек 4», «положить в стек 3» вычесть два верхних и заменить результатом. Сложить два верхних и заменить результатом. Проверить, что результат равен 6.

Наша задача состоит в том, чтобы подготовить такой стек, чтобы результат всех этих операций был бы «верно».

Нетрудно догадаться, что это будет стек всего с одним значением — 5.

Тогда результат применения операций будет (рис. 108):

Рис. 108



Так действует наш стековый язык и так работает наш замок. Мы должны представить такой ключ, чтобы при обработке его внутри замка в замке оставался бы только OK. Как мы только что разобрали, для замка 4, 3, –, +, =6 открывающим его ключом будет 5.

А к примеру, для замка «Хэшировать последний и заменить его хэшем», «Сравнить с СОХРАНЕННЫМ_ХЭШЕМ» — ключом будет только тот элемент, хэш которого будет давать СОХРАНЕННЫЙ_ХЭШ.

Перейдём к актуальному для темы главы примеру. Большинство цепочек наших ключе-замков в реальной сети биткоина заканчиваются следующим скриптом[1]:

OP_DUP
OP_HASH160
ЗДЕСЬ АДРЕС БИТКОИНА КУДА ОТПРАВЛЕНЫ ДЕНЬГИ В ВИДЕ HASH160
OP_EQUALVERIFY
OP_CHECKSIG

В реальности, разумеется, запись внутри транзакции гораздо короче. Команды заменяются короткими hex-записями. Запись

OP_DUP
OP_HASH160
6367b386808de90413833cf69f3baf2d80bd0fc6
OP_EQUALVERIFY
OP_CHECKSIG

будет эквивалентна hex-записи

76a914**6367b386808de90413833cf69f3baf2d80bd0fc6**88ac

[1] *С 2018 г. данная запись постепенно заменяется записью SegWit. Но по сути, это то же самое, только записи команд вынесены из самой записи во внешнюю часть и сокращены.*

СМИО ПРЕСС

Или в двоичном коде (hex to bin):

01110110101010010001010**0110001101100111101100111000011010000000100011011110100100000100000100011100000011001111001111011010011111001110111101011110010110110000000010111101000011111110000110**1000100010101100

Жирным мы отметили хэш публичного ключа. Остальное — команды.

Для простоты, мы покажем ключ, который открывает данный замок. Ключ, как мы помним, состоит из двух элементов: подписи и публичного ключа.

Мы очень подробно описали транзакцию в прошлой главе. Транзация состоит из ключа и замка. А ключ состоит из двух элементов.

Ключ: *<Подпись>*
<Публичный ключ>

Ещё раз напомним, что это просто самый частый вид замка. Замок, как мы приводили пример, может быть даже «дайте число, которое в сумме с 4 даст 6».

Итак, берём наш ключ и кладём его в стек. Напомним: стековая структура состоит в том, что мы работаем с элементами, которые положили сверху. Также особенность ситуации состоит в том, что мы замок кладём на ключ. После того, как мы положили ключ в стек, замок «проверяет» данный ключ.

Напомним ещё один важный момент: помним, что ключ и замок спаяны, т. е. майнер (или любой другой элемент системы) не может отделить ключ от замка, получается такой «ключе-замок», надёжность спайки определяется ЭЦП. Это важно, т. к. мы передаём ключ и замок неограниченному кругу лиц.

Тогда рисуем стек поэтапно:

<Подпись>

<Публичный ключ>
<Подпись>

Это наш «ключ». Теперь мы на наш ключ по одной накладываем команды из нашего «замка» и в результате наложения этих команд должны получить «OK». Так происходит на каждом узле. Если проверка не пройдёт, будет отвергнут весь блок.

Напоминаем наш замок с предыдущей страницы

OP_DUP
OP_HASH160
6367b386808de90413833cf69f3baf2d80bd0fc6
OP_EQUALVERIFY
OP_CHECKSIG

Если вы помните предыдущую главу, там у нас замок состоял только из хэша — здесь это третья строка. Остальные элементы — это способы проверки хэша (по сути, эта глава — описание стекового языка для проверки ключа).

СМИО ПРЕСС

Итак, у нас лежат в стеке два элемента ключа. И мы берём по одному элементу из замка и «накладываем» их на текущий стек, производя операции.

OP_DUP
<Публичный ключ>
<Подпись>

Операция OP_DUP берет верхний элемент стека и дублирует его. Т. е. у нас теперь стек меняется на:

<Публичный ключ>
<Публичный ключ>
<Подпись>

Кладём следующий элемент из нашего замка на получившийся стек. И стек превращается в:

OP_HASH160
<Публичный ключ>
<Публичный ключ>
<Подпись>

OP_HASH160 заменяет верхний элемент стека на его хэш
RIPEMD160(SHA256(x)).

То есть мы проводим операцию, которая была подробно изучена в прошлой главе — получения хэша из публичного ключа.

Теперь наш стек имеет вид:

Захэшированный публичный ключ
<Публичный ключ>
<Подпись>

Теперь мы добавляем следующий элемент — наш хэш. Мы чуть выше дали конкретный хэш — 6367b386808de90413833cf69f3baf2d80bd0fc6, но тут пока опишем его словами: это «адрес» биткоина (хэш), куда вы «отправляете» деньги.

ЗДЕСЬ АДРЕС БИТКОИНА КУДА ОТПРАВЛЕНЫ ДЕНЬГИ В ВИДЕ HASH160
Ваш Адрес в виде HASH160
<Публичный ключ>
<Подпись>

Следующий элемент замка — операция OP_EQUALVERIFY. И наш стек теперь:

OP_EQUALVERIFY
ЗДЕСЬ АДРЕС БИТКОИНА КУДА ОТПРАВЛЕНЫ ДЕНЬГИ В ВИДЕ HASH160
Ваш Адрес в виде HASH160
<Публичный ключ>
<Подпись>

СМИО ПРЕСС

OP_EQUALVERIFY удаляет два верхних элемента стека, если они равны, а если они не равны, то исполнение программы прерывается с ошибкой. Это именно та проверка, о которой мы говорили в начале прошлой главы, — узел проверяет, что ваш публичный ключ после хэширования даёт замок.

Если всё правильно, то наш стек превратился в:

<Публичный ключ>
<Подпись>

Теперь последний элемент. OP_CHECKSIG расшифровывает публичным ключом подпись и получает данные, которые были ранее зашифрованы приватным ключом.

OP_CHECKSIG
<Публичный ключ>
<Подпись>

То есть на последнем этапе узел проверяет, что вашим публичным ключом открывается эта подпись, и тогда возвращает «OK».

Теперь транзакция в полном вашем распоряжении, и вы можете ставить на неё свой замок. Если вы хотите просто перечислить средства на какой-нибудь адрес, то ставите аналогичный замок:

OP_DUP
OP_HASH160
ЗДЕСЬ АДРЕС БИТКОИНА, КУДА ВЫ ОТПРАВЛЯЕТЕ ДЕНЬГИ
В ВИДЕ HASH160
OP_EQUALVERIFY
OP_CHECKSIG

В конце транзакции мы должны получить «OK». Понятно, весь этот путь должен быть перепроверяемым. Каждый узел сети должен взять ваш публичный ключ, вашу подпись и пройти по всем запросам транзакции выхода.

Ещё раз: соединение ключа с замком будут проверять все узлы сети, прежде чем погасить все транзакции, которые ранее были неизрасходованными, и сделать ваши транзакции отправки неизрасходованными.

Как было описано в прошлой главе, проверка состоит из двух этапов:

1. Что вписанный вами публичный ключ соответствует адресу, куда отправлялись транзакции с указанными вами номерами.
2. Что ваша подпись приватным ключом расшифровывается именно этим публичным ключом.

Так мы передаём ценности в системе биткоина.

СМИО ПРЕСС

Ещё момент, о котором необходимо помнить: любая часть сети может передавать информацию по любым незащищённым каналам связи. Если информация о переводе по кредитной карте должна идти через особые системы, то информация о переводе может идти через абсолютно любые системы — не только Интернет, но и напрямую передаваться через смс или через текстовое сообщение, через пакетную спутниковую или коротковолновую связь.

При этом каждый узел проверяет транзакцию, чтобы передать её дальше. Поэтому невозможна атака на отказ обслуживания. Максимум можно заблокировать несколько нод, но вся сеть продолжит работу.

Табличка, показывающая, из чего состоит транзакция:

Размер	Название	Описание
4 байта	Версия	Каким правилам подчиняется транзакция
1–9 байтов	Количество входов	Количество входов у транзакции (стр. 123)
Переменный	Входы	Сами входы транзакций (количество мы указали прошлым пунктом, и это позволяет узлу заранее знать, сколько их будет)
1–9 байтов	Количество выходов	Количество выходов у транзакции (стр. 123). Напоминаем: это число никак не связано с количеством входов. Это как вы можете потратить одну большую купюру на много покупок или, наоборот, мелочью заплатить за одну покупку
Переменный	Выходы	Сами выходы
4 байта	Locktime	Unix-время

Некоторые условности и сложности

В нашей системе возможно «объединение» камней, т. е. объединение транзакций. Для человека, пользующегося кошельком, это может быть не так очевидно, но это важно, если вы программируете систему. При отправке средств кошелёк проверяет, есть ли какая-то входящая транзакция, большая этой суммы. Если есть, то отправляет её целиком, получая обратно «сдачу». Если такой суммы нет, соединяет несколько транзакций, которые были ранее, и опять же, если их сумма больше транзакции, получает часть обратно.

Майнер может не захотеть включать транзакции в блок. Это похоже на то, как человек, который привёз камень, отказывается от участия в собрании, либо согласился на участие в собрании не 100 самых крупных жертвователей, а ещё ограничил минимальным размер дара. Тогда участники системы не могут совершить транзакции. Раньше эта проблема стояла более остро, так как расчёт хэша без транзакций был проще, а значит, выигрывал тот, кто не брал транзак-

СМИО ПРЕСС

ции. Проблема решилась сама собой: со временем выросли комиссии за включение транзакции в блок, а ранние участники системы были в большинстве своём идейными людьми и не позволяли себе облегчать жизнь подобным образом.

Наша аналогия с печатями на камнях, их стиранием и выставлением новой, конечно, очень условна. Разумеется, в реальном мире мы сам камень шифруем и расшифровываем. Более того, в экосистеме биткоина мы не шифруем публичным ключом — мы ставим адрес.

В экосистеме криптовалют фактически нет перевода средств. Если вы имеете доступ к транзакции, вы просто добавляете к ней чужой код. Это и считается переводом денег.

Ещё раз о цепочке транзакций.

Отметим, что каждый полноценный узел сети проверяет все транзакции и строит новую картину UTXO после каждого блока. В простейшем виде такая цепь выглядит так.

Сейф — это коинбейз транзакция (то, что получает майнер, создавший блок), потом замок, туда ключ-замок, ключ-замок. Каждый блок приносит в себе ключ-замок для всех транзакций, включённых в блок.

В общем виде (рис. 109):

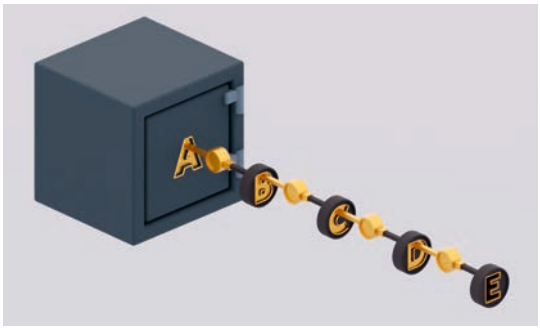


Рис. 109

Нужно помнить, что мы лишь в начале описания переводов в системе биткоина. Но главное, что вы должны понять, — что существует возможность создать «замок», который будет проверять «ключ» с помощью функций стекового языка программирования. Далее нас ограничивает только наша фантазия.

СМИО
ПРЕСС

Глава 15
ХЭШЧЕЙН

В главе 8 мы познакомились с хэшированием и узнали, что каждому тексту можно представить его «отпечаток», который можно считать однозначно идентифицирующим весь текст.

Берём случайную строку

Любой текст

Получаем хэш из неё:

cd02a072b88ab4b007994e4a2b8e4feaa57e3f416358227ec1c33ec69e2c9558

Разобравшись с хэшированием, мы можем перейти к понятию «хэшчейна» — это цепочка хэшей, которая нужна для того, чтобы невозможно было изменить предыдущие записи без изменения следующих.

Предположим, у нас есть система транзакций (переводов) или абсолютно любой другой информации. К примеру, цепочка транзакций: Боря перевёл Насте 1000 р., Настя перевела Ефиму, Ефим перевёл Ане, Аня перевела Соне.

На странице «Хэшчейн» нашего сайта вы можете, как и в главе 6, получить хэш из любой строки. Теперь представьте, что у нас несколько важных записей, из которых мы получаем хэш.

Первый перевод

Вася перёвел Пете 500р.

Получаем хэш из неё:

fdde2f241980c43853fd5146a14ba85691c42947b52d88d78177450720b98d78

СМИО ПРЕСС

Второй перевод

Александр перевёл Маше 1000р

Получаем хэш из неё + fdde2f241980c43853fd5146a14ba85691c42947b52d88d78177450720b98d78

2d1285c625817503731bc3ff2f1e5abf6ce4cc2747c111f6b41215ea1ece4d0c

Третий перевод

Настя перевела Боре 20р.

Получаем хэш этой строки + 2d1285c625817503731bc3ff2f1e5abf6ce4cc2747c111f6b41215ea1ece4d0c

983463d520ddaf3d211e603d8b2e6ddf1f90a237ad6d7d26a2fae1cace46c418

Пожалуйста, попробуйте изменять записи на нашем сайте: https://crypto.239.ru/hashchain.

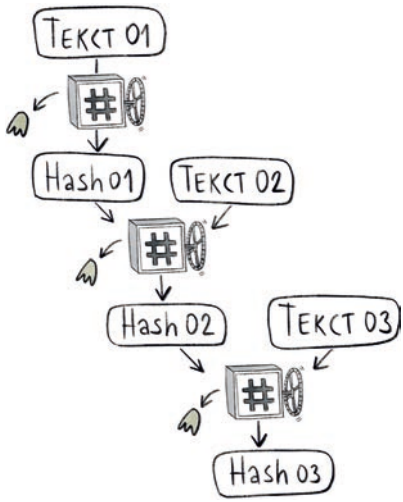


Рис. 110

Это три независимые записи. Вы можете менять каждую и видеть, что меняется её хэш. Но в отличие от прошлых глав, теперь мы вводим новое правило — хэш мы берём не от самой записи, а от строки, которая получается прибавлением хэша прошлой записи к текущей. Это следующий пример на сайте — как вы видите, хэш у нас получается из суммы строк (рис. 110).

Если вы попробуете изменить первую запись, соответственно изменится её хэш; т. к. изменится хэш, изменится вторая строка; т. к. первый хэш является её частью, соответственно изменится хэш второй строки. Соответственно изменится третья запись, т. к. второй хэш является её частью. И будет изменён третий хэш.

Давайте изучим это по шагам. Сейчас первая строка:

Вася перевёл Пете 500р.
Её хэш:
fdde2f241980c43853fd5146a14ba85691c42947b52d88d78177450720b98d78

СМИО ПРЕСС

Вторая строка:

Александр перевёл Маше 1000р.

ВНИМАНИЕ! Мы берём хэш не из строки «Александр перевёл Маше 1000р.», а из строки: «Александр перевёл Маше 1000р.fdde2f241980c43853fd5146a14ba85691c4294788d7 8177450720b98d78», т. е. мы перед взятием хэша из строки добавляем прошлый хэш.

Получается, что хэш этой строки зависит от хэша прошлой строки.

Следовательно, любое изменение любой промежуточной строки приведёт к изменению всех последующих хэшей. Мы предлагаем поэкспериментировать на нашем сайте.

Добавим один символ к первой строке — и все хэши изменятся. Пожалуйста, потратьте 5 минут на изменения строк на нашем сайте. Добавьте символ к первой строке, и вы увидите, что изменятся все последующие.

Если возникнет вопрос о каком-либо переводе, можно будет проверить все последующие хэши, а злоумышленнику придётся изменить не только эту запись и её хэш, но и абсолютно все последующие хэши.

Разумеется, это не очень серьёзная защита. Злоумышленник, получив в своё распоряжение базу данных, может изменить в ней все записи за приемлемое время и разослать её дальше. И тот, кто получит эту базу, не будет понимать, где верная запись, а где нет.

Но теперь мы введём новое правило: к каждой записи нужно добавить символы, чтобы хэш имел 0 в начале. Надеюсь, вы помните главу Proof of Work?

То есть, чтобы принять первый перевод «Боря перевёл Насте 1000р.», нам нужно найти добавку XXX, чтобы хэш получился нужного вида — с 00 в начале.

Далее то же самое сделать остальными переводами. Таким образом, каждый перевод включается в цепь, только если отправитель нашёл такую добавку, чтобы хэш из неё содержал два первых нуля.

По сути, то же самое происходит и в биткоине — каждый блок образуется в тот момент, когда один из участников найдёт такую добавку, что в начале хэша будет определённое количество нулей (на конец 2022 года – 20 нулей).

СМИО ПРЕСС

Если злоумышленник вознамерится сломать систему и отменить свой перевод , ему нужно будет пересчитать все хэши, начиная с этого перевода. Но не просто пересчитать, а сделать Proof of Work абсолютно для каждой записи.

К примеру, на шаге 2 Александр перевёл Маше 1000 р. А через какое-то время Александр хочет, чтобы в системе было записано, что он перевёл только 500 р. Он создаёт такую запись, считает её хэш, совершает Proof of Work, добавляет этот хэш к следующему элементу, дальше совершает Proof of Work для этого элемента и получает новый хэшчейн. И распространяет его, говоря, что это верная последовательность и верные блоки, а те, что распространяют другие — с переводом в 1000 р., это ложь.

По сути, блокчейн и создан, чтобы решить эту проблему хэшчейна — определить, какой из переводов верный. Этому будет посвящена следующая глава.

Проблема двойных трат

Мы надеемся, что читатель понял главную мысль хэшчейна: мы можем взять и упорядочить все блоки, связав их хэшами. Тогда мы получили цепь блоков, где последовательность жёстко задана. При этом, если мы произведём изменение в прошлых транзакциях, изменятся хэши во всех последующих. Далее это будет важно для контроля записей.

Реальный хэшчейн в биткоине, который мы разберём позже в главе 18: у нас есть список транзакций, из которых мы получаем хэш, из суммы хэшей мы получаем глобальный хэш и записываем его в блок. У нас есть предыдущий хэш, метка времени, хэш всех сделок и наша добавка (рис. 111).

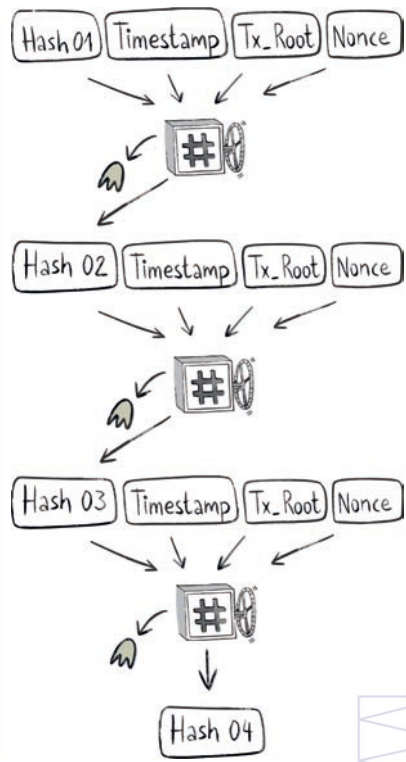


Рис. 111

Как мы помним, в хэшчейн может писать кто угодно. Отсюда возникает главная проблема: как определить, какая из цепочек верная. Как определить, какую цепь блоков в нашем связанном хэшчейне можно использовать, а какая является ложной? Та, где Александр переводит Маше 1000 р., или та, где он переводит ей 500 р.?

Напоминаем, что эту задачу можно свести к «задаче о византийских генералах», которой была посвящена вторая глава. Как с приемлемым уровнем риска подтвердить, какое из решений верное?

СМИО ПРЕСС

Глава 16
ПРОБЛЕМА ДВОЙНЫХ ТРАТ

По сути, все предыдущие главы были лишь подготовкой, а все идеи, которые в них заложены, были известны задолго (за годы) до создания биткоина в 2008 году. Инновация системы биткоин состоит не в том, что передача ценности — это шифрование чужим публичным ключом и не в идее P2P[1], которая была реализована ещё в торрентах.

Инновация и суть работы Сатоши Накамото состоит в решении проблемы двойной траты. Двойная трата — это попытка пользователя, которая может запутать сеть, указав не одного, а нескольких получателей одних и тех же средств.

Помните пример из предыдущей части? Александр записывает в блок информацию «я передаю 1000 р. Кате». А чуть позже или даже одновременно с этим он пишет «я передаю 1000 р. Маше». Подписывает оба сообщения своим ключом и отправляет одно сообщение в одну ноду (узел), а другое — в другую ноду.

При этом в наш хэшчейн могут писать все, кто выполняет работу. Как же защититься от двойной траты?

В этой статье мы кратко перескажем статью Сатоши Накамото и объясним содержание сделанного им изобретения.

Ещё раз напомним суть финансовой системы: когда вы переводите кому-то деньги, вы пишете доверенному центру, и он фактически уничтожает ваши монеты и создаёт новые. Как мы уже неоднократно писали, это дешёвый и удобный подход. Но его минусы в том, что вся денежная система находится в руках эмитента, который является монополистом. Внутри России это не является проблемой, однако, есть множество стран, где нет серьёзной денежной политики, а особенно большие сложности это создаёт для международной торговли.

В такой централизованной системе проблемы двойной траты решаются просто: какой запрос эмитент получил первым, тот платёж и будет обработан. Если у вас на счёте есть 10 000 р. и вы с двух компьютеров пошлёте одновременно запрос в вашем личном кабинете «Перевести 9000р. Кате» и «Перевести 10000 р. Маше», то будет обработан тот запрос, который по каким-либо причинам пришёл в банк первым.

---

[1] *Термин P2P расшифровывается как peer-to-peer (равный к равному) и означает, что все узлы сети равны, нет единого центра. В данной сети необходимы правила согласования (или консенсуса), по которому она работает.*

СМИО ПРЕСС

В распределённой системе вы можете послать один запрос одной ноде и другой запрос — другой ноде. Что в этом случае делать системе?

Вспомним, что мы описали цикл хэшчейна. Таким образом, есть последовательность блоков, которые благодаря использованию хэша из предыдущего блока спокойно выстраиваются в цепь (рис. 112).

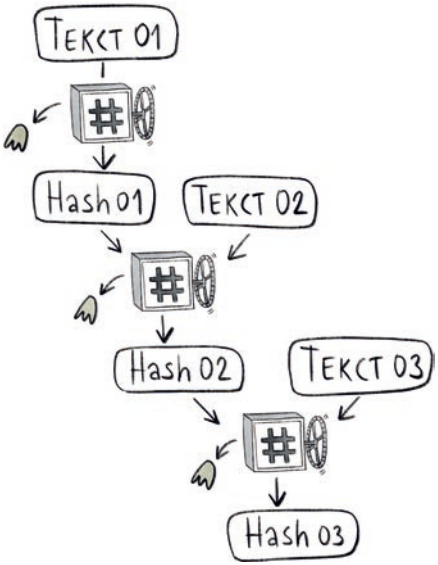


Рис. 112

Мы также решили проблему с тем, что в одном блоке не может быть две разные транзакции. В статье про UTXO мы разобрали, что транзакции наслаиваются друг на друга, помните рисунок с камнями или ключе-замки?

Невозможно одновременно на входящую в ваш кошелёк транзакцию наставить и транзакцию Кате, и транзакцию Маше.

Но у нас есть две разных транзакции — и распределённая система. Есть сеть нод — узлов сети.

На одну ноду Саша послал информацию, что хочет перевести Кате, а на другую — Маше. Это можно рассмотреть, как будто Саша создал два ключе-замка — с замком под Катин и с замком под Машин ключи (рис. 113).

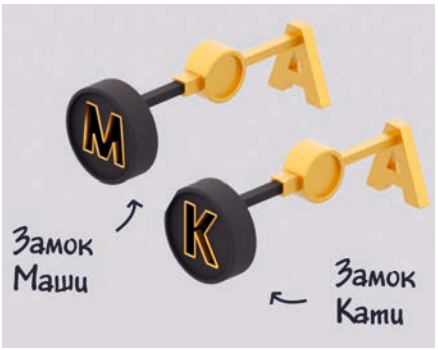


Рис. 113

Один из майнеров включил одну транзакцию (ключе-замок) в свой блок, другой майнер — другую транзакцию.

По каким правилам каждый отдельный узел принимает решение, какой блок включать в собственную базу?

В своей главной статье, т. н. «Белой книге» (полностью скачать её можно по ссылке http://satoshinakamoto.me/bitcoin.pdf) Сатоши описал это следующим образом (единственный блок на английском в нашей книге):

Nodes always consider the longest chain to be the correct one and will keep working on extending it. If two nodes broadcast different versions of the next block simultaneously, some nodes may receive one or the other first. In that case, they work on the first one they received, but save the other branch in case it becomes longer. The tie will be

СМИО ПРЕСС

broken when the next proof-of-work is found and one branch becomes longer; the nodes that were working on the other branch will then switch to the longer one.[1]

В этих строках — решение главной проблемы распределённых реестров. Как сделать так, чтобы убрать расхождения в распределённой сети, если одна часть сети считает верным одну транзакцию, а другая часть – верной другую?

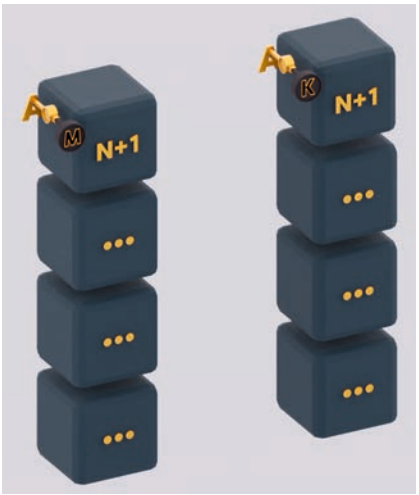


Рис. 114

Верной считается та цепочка, в которой больше блоков. Вернёмся к нашему примеру. У нас две цепи с блоками, транзакции в которых отличаются. В том числе отличаются транзакции у вас: в одной цепочке блок с переводом Кате, в другой — Маше (рис. 114).

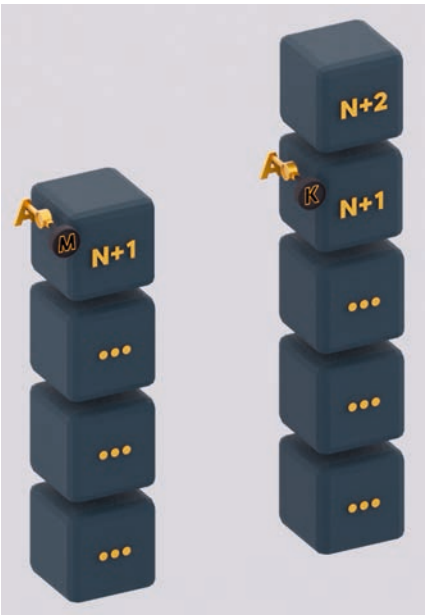


Рис. 115

Верной будет считаться та цепь, которую продолжит следующий майнер. То есть, в какой-то момент будет добыт следующий блок, в который (как мы помним) включается хэш прошлого, — и чей хэш будет включён, тот и будет считаться верным, т. к. принадлежит более длинной цепи.

Поэтому после включения транзакции в блок ещё нельзя сказать, кто является владельцем переданной суммы — Катя или Маша. Нужно ждать несколько блоков (рис. 115).

[1] *Ниже мы расшифровываем эту идею максимально подробно.*

СМИО ПРЕСС

Вы можете возразить: а что будет, если один майнер добавил блок в эту цепь, а вдруг произошло такое, что другой майнер добавил к другой. А потом третий майнер добавил в ту же цепь? Иначе говоря, что делать, если более короткая цепь в результате крайне маловероятного стечения обстоятельств вдруг стала длиннее (на рис. 115 цепь с транзакцией Маше была короче, а на рисунке ниже (рис. 116) она же стала длиннее).

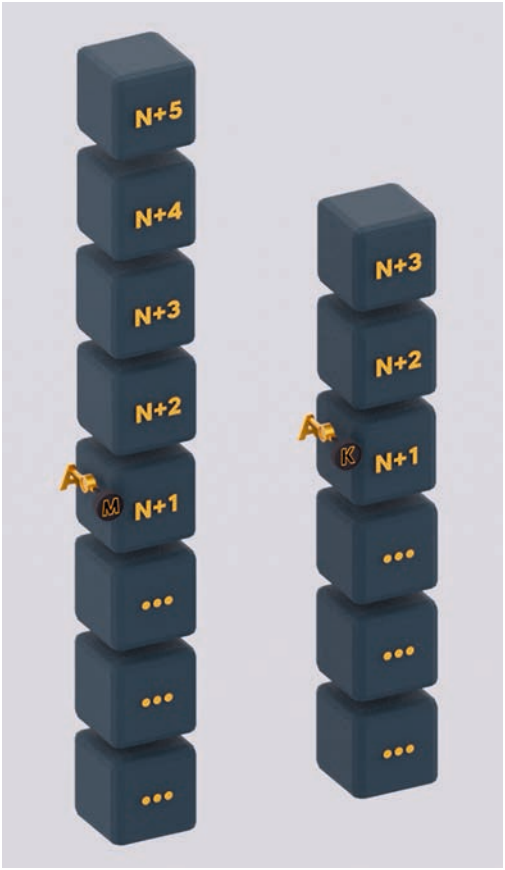


Рис. 116

Тогда то, что раньше было верным, станет неверным. Это правда. Если раньше мы могли сказать: «Перевод Кате получил одно подтверждение», то теперь: «Перевод Маше получил четыре подтверждения» (рис. 116).

Можно ли быть уверенным, что это когда-нибудь закончится? Да, конечно, потому что только случайность может заставить майнера добавить свой блок в ту цепочку, в которой меньше блоков. Майнеру важно получить деньги, а если его блок окажется в меньшей цепочке, то эта цепочка не будет поддержана нодами, а значит, он не получит вознаграждение за включение в блок.

Теперь вам должно быть понятно выражение «деньги считаются поступившими на счёт после *X* подтверждений». На заре биткоина надёжным считалось 6 подтверждений. Сейчас пользователь может считать, что деньги ему

СМИО ПРЕСС

поступили, если его перевод получил два подтверждения. Но, разумеется, в крайне маловероятном случае его перевод может быть отменён, если некий майнер сделает другую цепь длиннее.

Также мы решили проблему «а что делать, если один из узлов сети вышел и вернулся через какое-то время — как ему продолжать работу». Всё решается просто: узел берёт наиболее длинную на этот момент цепь. Больше блоков — значит, она верная[1].

Надеюсь, у читателя также возник вопрос: «А что будет, если мы намеренно хотим добавлять блоки только в нашу цепочку и обладаем такими ресурсами?» Да, это называется атакой 51%. Действительно, тот кто обладает 51% ресурсов (больше половины), может отменять предыдущие транзакции. Поэтому так опасна централизация в биткоине.

Но нужно отметить, что добыча биткоина носит вероятностный характер. Если у вас 51% ресурсов, то нет никакой гарантии, что отстав на 1–2 блока, вы сможете когда-нибудь «догнать» цепочку. А если ваша цепь отстала на 10 блоков (это происходит примерно за 1,5 часа), то догнать даже с 60% ресурсов будет проблематично, и вряд ли потраченные ресурсы будут окупаться возможностью отменить свою сделку. Отмена же чужих переводов вообще выглядит бессмыслицей[2].

Тем не менее, атака 51% действительно подвергает опасности всю систему. И целенаправленные действия очень богатого человека действительно могут подорвать эту систему. В Приложении 5 мы описали опасности, которые могут подстерегать биткоин. Там рассмотрена в том числе эта атака.

Эта глава завершает описание всей системы, дальше у нас только главы, уточняющие некоторые аспекты.

---

[1] *Здесь нужно сделать оговорку. На самом деле считается верной не та цепь, где было больше блоков, а та цепь, сложность которой оказалась самой большой. Если бы мы ориентировались только на количество блоков, кто-то мог создать миллион блоков с минимальной сложностью (только одним нулём в начале хэша), а после – создать только один блок с действительно правильной сложностью и перетянуть всю цепь на себя.*

[2] *Бессмысленно отменять чужие транзакции, потому что чужой может включить эту транзакцию в следующий блок буквально через секунду, и эту транзакцию тоже придётся отменять.*

СМИО
ПРЕСС

Глава 17
БЛОКЧЕЙН

Познакомившись с хэшчейном — цепочкой хэшей, мы можем перейти к блокчейну — цепочке блоков.

Блокчейн — это по сути хэшчейн, снабжённый распределённой сетью, в которую может писать любой выполнивший нужную работу, т.е. хэшчейн, обладающий набором дополнительных свойств. Конечно, многие люди по-разному интерпретируют термин «блокчейн», называя блокчейном любую цепочку блоков, однако мы определим блокчейн более узко.

Во-первых, новый блок может добавиться любым узлом — по определённым правилам, которые обычно называются Proof of XXX — правила выбора узла, который имеет право добавить блок в блокчейн. Ранее мы познакомились с методом консенсуса Proof of Work, в книге есть Приложение 2, которое рассматривает другие методы консенсуса. Эти методы не такие надёжные, но потребляют меньшее количество ресурсов.

В блоке находятся список транзакций, метка времени, нонс, хэш дерева Меркла, список всех транзакций. Очень подробно состав блока мы разбираем в следующей главе.

Проблема с термином «блокчейн» в том, что он стал просто модным словом. Сейчас под блокчейном могут подразумевать и метод достижения распределённого консенсуса, и систему, при которой нет удостоверяющего центра. А некоторые, видимо для рекламы, вообще называют блокчейном систему, где нет разделения на блоки. К примеру, Corda или Ripple.

Иногда блокчейном уже называют просто распределённые базы данных, которые хранятся на ограниченном количестве серверов. Появился даже термин «частный блокчейн». В это же время, разумеется, термин блокчейн обрастает огромным количеством терминов-спутников, которые обычно ничего не значат, а просто созданы из маркетинговых соображений. Блокчейн 2.0, даже Блокчейн 3.0, можно слышать термин сандчейн или парачейн (параллельный блок). К примеру, для Rootstock.

В общем, по сути, блокчейн стал очень размытым термином, поэтому мы практически не употребляем его в книге. Тем не менее, мы не можем пройти мимо этого термина — и определим его, как нам кажется, единственно правильным образом.

СМИО ПРЕСС

Когда мы рассказывали о камнях в первой главе, то консенсусом являлось просто общее согласие конкретных людей. Все знают, где чей камень расположен, все согласны, если его владелец говорит «вот этот камень — теперь его». По сути, это и есть консенсус и добавление блока.

Здесь можно вновь обратиться к задаче византийских генералов. Суть в том, что генералам необходимо достигнуть общего консенсуса — атаковать или отступать. При этом любой из них, руководитель или ординарцы (гонцы), может быть предателем (их меньше, чем честных генералов, но их может быть много).

Мы подробно говорили о том, как теоретически может достигаться консенсус в этом случае: солдаты садятся решать задачу, и чей солдат первым её решит, тот и посылает к остальным генералам ординарца с решением, как действовать остальным. Так как солдат у честных генералов больше, значит, с высокой долей вероятности можно говорить о том, что задачку решит солдат у честного генерала. По крайней мере, решение будет единообразно, что вполне устроит всех. Главное, что оно едино (вместе атакуем, вместе отступаем).

Напоминаем, что термин «генерал» всего лишь дань моде — в экосистеме биткоина вы сами себя назначаете «генералом», просто создавая узел. Вас никто не назначает, и вы можете полностью работать по своим правилам, в т.ч. попытаться стать «предателем», т. е. попытаться работать против сети.

Но главное преимущество блокчейна — это разрушение эвристического утверждения[1], называемого теоремой CAP[2], которая гласит, что **невозможно** одновременное выполнение трёх условий:

1. Во всех узлах данные не противоречат друг другу.
2. На запрос к любому узлу мы получаем корректный ответ.
3. Возможно разделение сети на несколько отдельных секций, но при этом каждая из секций выдаёт правильный ответ при запросе, а при воссоединении сети есть лёгкий способ согласовать данные.

Блокчейн ломает эту теорему, т. к. предлагает единый стандарт определения правильного ответа — полученная цепочка должна быть самой длинной (сложной). Если часть сети отпадает или выключается, а при соединении получает разные данные от разных частей сети, то эта часть просто выбирает более длинную цепочку, что было разобрано в главе о хэшчейне.

---

[1] *Эвристическое утверждение – утверждение, истинность которого многократно проверена на практике, но не доказана.*

[2] *CAP – это аббревиатура от английских слов Consistency (Согласованность, Целостность), Availability (Доступность) и Partition tolerance (Устойчивость к разделению).*

СМИО ПРЕСС

Если в какой-то момент цепь разъединилась и две её части росли самостоятельно, то при соединении выбирается более длинная цепь, а всё, что было приставлено к меньшей сети, выкидывается. Все транзакции, которые были включены в удалённые блоки, отменяются (рис. 117).

Рис. 117

СМИО
ПРЕСС

Глава 18
ПОДРОБНОЕ ОПИСАНИЕ БЛОКА БИТКОИНА[1]

В предыдущих главах мы рассмотрели все идеи, которые позволяют биткоину работать. Здесь мы подробно изучим реализацию этих идей в составе блока. Как мы помним, по сути, любая криптовалюта (по крайней мере, мы так определяем) — это набор связанных блоков.

Вернёмся немного назад и напомним, что такое блок. Блок — это записи о том, как переходят ценности от одного человека другому. Напомним о нашей главе о блоках на острове (рис. 118).

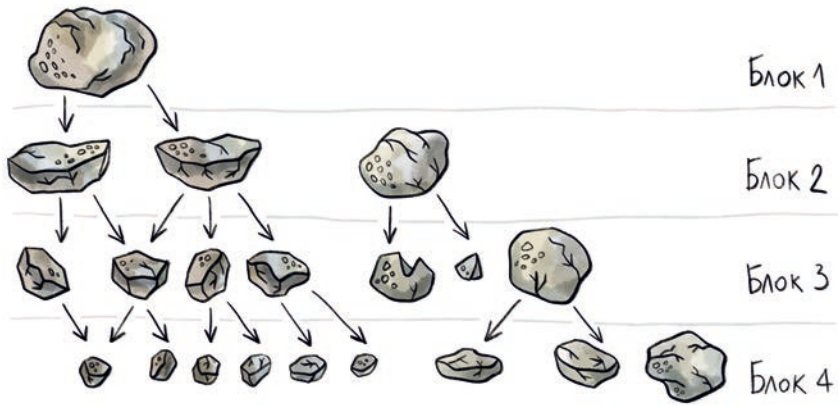


Рис. 118

Все блоки связаны между собой хэшами прошлого блока, принцип связи через хэши мы подробно описали в главе о хэшчейне.

Мы напоминаем, что блоки создаются майнерами. Мы описали процесс майнинга с разных сторон. В задаче византийских генералов — это отдельные бойцы (майнеры) или бойцы, объединённые одним генералом (майнинг-пулы), которые решают общую задачу. И тот кто решил, получает вознаграждение. В задаче византийских генералов — назначает время атаки. В экосистеме биткоина — приобретает возможность получить «ценность» из ниоткуда (подробнее о майнинге в следующей главе).

[1] *Далее мы просто говорим «блок», подразумевая под этим термином «блок блокчейна биткоина».*

СМИО ПРЕСС

После нахождения кода майнер отправляет эту информацию максимальному количеству узлов (нод), т. е. перенаправляет следующим узлам (нодам) и так далее.

Изначально термины «блокчейн» и «криптовалюты» вообще были синонимами, но как всегда бывает, оба термина всё больше размываются, и ваш собеседник может под термином «блокчейн» понимать совсем не то, что вы ожидаете.

Именно поэтому мы избегаем этих терминов и на всем протяжении книги используем термин «биткоин». И здесь мы будем говорить о «блокчейне биткоина» или «блоках биткоина».

Разберём строение блока биткоина с технической стороны.

Блок состоит из трёх частей:

1. Технические поля.
2. Заголовок блока.
3. Список транзакций.

Технические поля вообще обычно не рассматриваются в документации. Но мы их перечислим:

1. Магический номер (Magic no.). Просто число размером 4 байта, которое идентифицирует сеть. И всегда равно 0xD9B4BEF9. Данное число — просто знак того, что мы работаем в сети биткоина[1].
2. Размер блока (blocksize). Тоже просто число размером 4 байта, которое показывает размер блока в байтах.
3. Количество транзакций (transaction counter). Смысл очевиден, отметим только, что размер числа может быть от 1 до 9 байт.

Технические поля

Название	Смысл	Размер
Magic no.	Идентификатор сети. В сети биткоина 0xD9B4BEF9	4 байта
Transaction counter	Количество транзакций	1—9 байт
Blocksize	Размер блока в байтах	4 байта

Второй пункт — это **заголовок блока**. Это наиболее важная часть блока, но многие его поля основываются на списке транзакций и формируются в последнюю очередь, поэтому сначала мы рассмотрим список транзакций.

[1] *В другой сети этот номер будет другим.*

Список транзакций. Как мы помним, слово «транзакция» лишь дань моде. Никаких переводов средств в традиционном банковском понимании у криптовалют нет, так как, по сути, нет счетов. Здесь под транзакцией мы понимаем добавление нового кода в конец существующей транзакции.

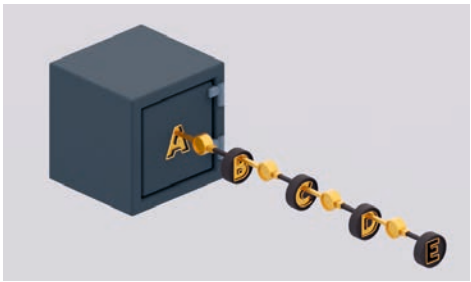


Рис. 119

Мы описывали это как добавление нового кодового замка на сейф или ключе-замка (рис. 119).

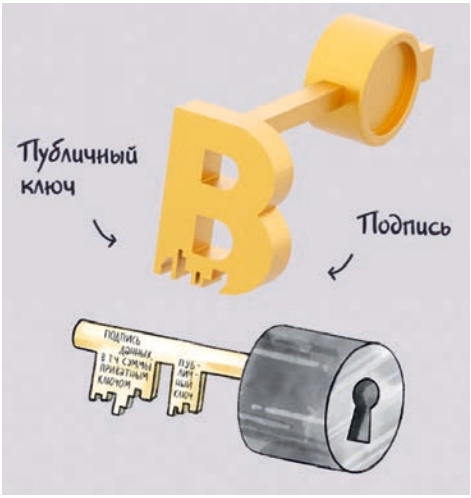


Рис. 120

В главе 14 мы полностью изучили, как создаётся наш ключе-замок (рис. 120).

Вот эти новые записи (или наложение кодовых замков на сейфы) мы и называем «транзакциями» Совокупность всех добавлений мы называем «списком транзакций».

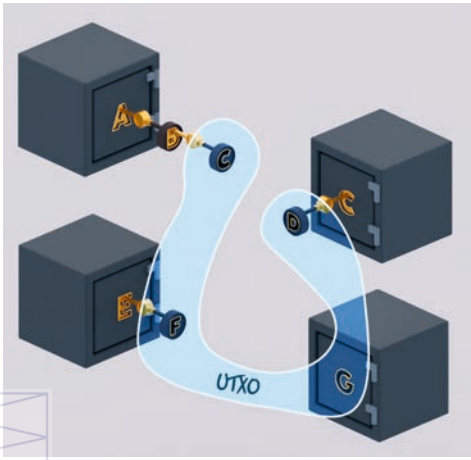


Рис. 121

Нужно упомянуть важнейший термин, который вынесен на картинку третьей части. UTXO (unspent transaction output) — список выходов транзакций. Под UTXO мы понимаем полный список всех текущих кодовых замков, которые наложены на сейфы или, более точно, все последние записи на всех цепочках записей.

То есть, если у нас есть 4 сейфа, то совокупность всех последних «замков» и есть UTXO (рис. 121).

СМИО ПРЕСС

Транзакции в каждом блоке определяют новые UTXO, т. е. часть выходов в глобальной картине меняется. Часть выходов транзакций остаётся, а некоторые остались на месте с момента создания (к примеру, почти все транзакции Сатоши Накамото).

Есть только одно исключение — generation transaction. Это транзакция «зачисления» на счёт майнера (или просто данные адреса майнера). У этой транзакции нет входа, а её размер определяется текущим состоянием сети. Это как «родной» замок у сейфа. Такую транзакцию можно понимать как создание сейфа или привоз нового камня для жителей острова.

Размер этой транзакции, т. е. сумма, которую получает майнер каждые 210 000 блоков (примерно каждые 4 года), уменьшается в два раза. Мы подробнее расскажем о таких транзакциях в следующей главе, посвящённой майнингу.

Кроме этой «порождающей» транзакции, которая всегда стоит на первом месте, каждая транзакция, как вы помните, должна содержать:

1. Ключ. 2. Замок.

Причём замок — это список функций стекового языка, а ключ — данные, к которым применяется прошлый замок (в который вставляется ключ).

Напомним, что все данные в блокчейне открыты. Вы можете самостоятельно открыть узел сети и узнать все данные обо всех транзакциях за всю историю биткоина. А информацию о новом блоке будете получать сразу же, как только он будет сформирован. Ещё раз отметим, что узел не майнит валюту, т. е. не создаёт новые блоки. Это лишь контроль за сетью.

Есть большое количество специализированных сайтов со своим узлом и веб-интерфейсом для просмотра всех транзакций. Возьмём, например, скрин транзакции с сайта https://www.blockchain.com/. На этом сайте вы можете выбрать любой блок и любую транзакцию для изучения. Мы взяли 500 000-й добытый блок: https://www.blockchain.com/explorer/blocks/btc/500000. Рассмотрим несколько транзакций. Для примера возьмём вторую транзакцию, у неё 2 входа и один выход (рис. 122). Как мы видим, у каждой транзакции есть хэш.

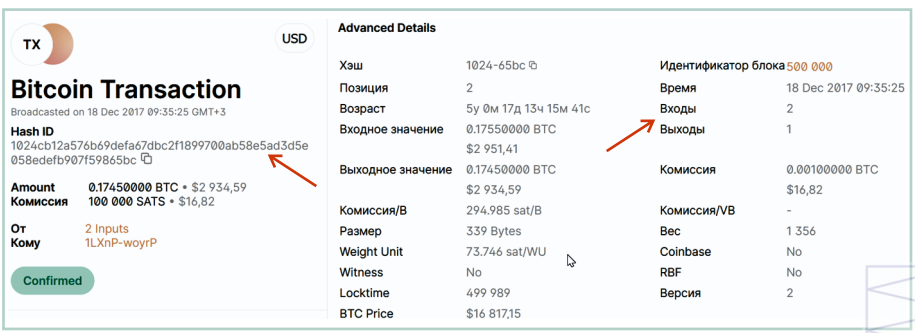


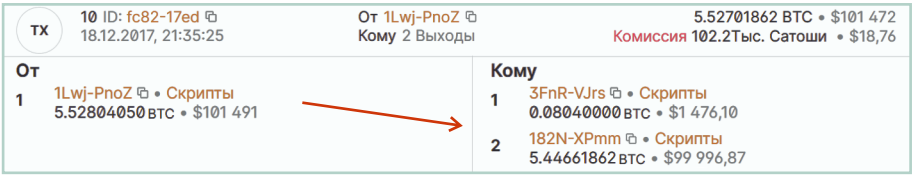
Рис. 122

СМИО ПРЕСС

Это просто строчка, которая определяет эту транзакцию. Как мы помним, хэш можно получить из любой информации, он будет заранее известной длины и будет однозначно определять текст, как отпечаток пальца.

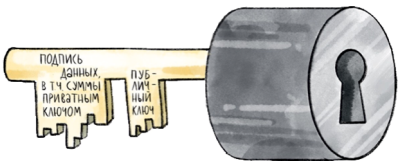
Мы ещё поговорим о хэше транзакций позже, когда зайдёт речь о дереве Меркла. Также есть входы транзакции (в этой транзакции их несколько), а также выход транзакции. Как мы помним, может быть наоборот — один вход и множество выходов. Так выглядит десятая транзакция в блоке:

Рис. 123



Ещё раз обсудим, что означает несколько выходов. Чтобы транзакция была совершена, владелец кошелька должен потратить всё. Он хотел перевести часть денег на один адрес, часть — на другой. При этом часть денег он может вернуть. Это похоже на ситуацию, когда у вас всего одна купюра. Когда вы приходите в магазин, вы должны отдать её полностью. Если вы не хотите потратить всё, вы просто указываете, сколько хотите перевести на адрес продавца, а сколько — на свой другой (или тот же самый) адрес.

Рис. 124



Ключ-замок из главы об UTXO

Далее, необходимо вспомнить главу об UTXO.

Каждая транзакция содержит ключ, которым отпирается предыдущий замок, а также замок, который может открыть только адресат. Мы пытались объяснить этот принцип тремя разными способами: наложением нового кода на старый, новой записью или ключе-замком (рис. 124).

Ясно, что каждая транзакция должна иметь ссылку на прошлую, иначе в принципе непонятно, к какой транзакции (какому замку) она относится и к какой транзакции должна быть подсоединена. Ссылкой является хэш прошлой транзакции. Ссылка на прошлую транзакцию (Previous tx):

g5d8ee39a430902c92a5927b9g1dc29d6d2a0e9cea105b009ca73dd04470b9a6

Index (внутри транзакции. Здесь транзакция входа была одна, поэтому 0.): 0.

Ключ (scriptSig) состоит из подписи и публичного ключа.

Напоминаем сам принцип открытия-закрытия сейфа (глава 13): мы просто в конец нашей транзакции ставим новую запись. Каждый узел проверяет её валидность и обновляет запись у себя.

Концепция показала свою сложность, поэтому опишем её ещё раз с конкретной, случайно выбранной нами транзакцией.

СМИО ПРЕСС

Можно увидеть все эти элементы на скриншоте со страницы сайта по ссылке https://www.blockchain.com/ (500 000-й блок, десятая транзакция с одним входом и несколькими выходами, рис. 123).

Напоминаем, что транзакция в нашей терминологии — это ключ плюс замок. Часто на сайтах используется другая терминология. На данном сайте «вход» — это прошлый замок (замок прошлой транзакции) плюс ключ к нему. Выходы — это замки, которые мы поставили.

Нажав на ссылку «Скрипты» в поле «От», получим (рис. 125):

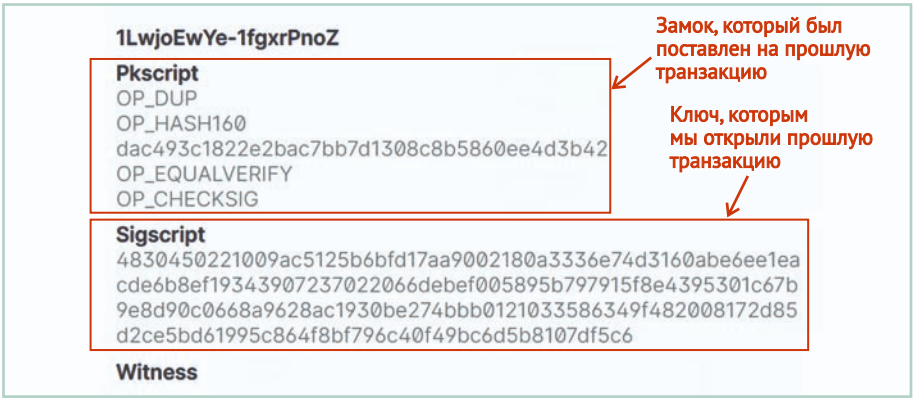


Рис. 125

Итак, на сайте мы видим конкретный Sigscript — это та часть, что «отпирает» транзакцию. Это наш ключ. Немного упростив, мы уже представляли его как зашифрованный нами код, который каждый может расшифровать уже полученным нами публичным ключом.

То есть ключом мы «открыли» прошлую транзакцию (прошлый замок транзакции), а теперь мы запираем новым замком. Новый замок — это «выход транзакции».

Совокупность активных выходов (незанятых замков) называется UTXO.

Нажав на кнопку «Скрипты» около второй транзакции выхода в поле «Кому», увидим:

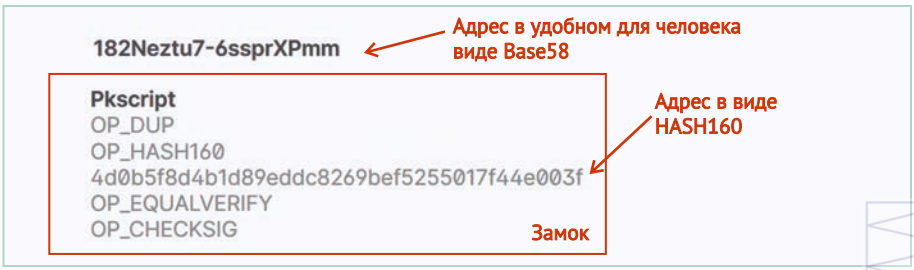


Рис. 126

О том, что адреса (рис. 126) совпадают и просто написаны в разных форматах, мы говорили в главе 13, где формировали такие же адреса.

Напоминаем, что каждая транзакция может иметь несколько выходов. То есть это как разделение камней на примере острова Яп. А может быть наоборот — «склейка». Это можно также описать, как покупку в магазине. При входе в магазин у вас может быть купюра 5000 р., а вы хотите купить что-то за 1500 р.

Вы отдаёте 5000 р, получаете то, что вам нужно, и 3500 р. на сдачу.

А может быть ситуация, когда у вас две купюры по 1000 р.; вы отдаёте обе купюры (два входа), получаете, что вы хотите, и ещё сдачу 500 р. (у вас было две купюры, стала одна).

Напоминаем, что выход — это только навешенный замок, который будет открыт и сразу закрыт только при переводе средств. Очень важно понимать, что мы можем поставить любой замок, который нам нравится. В мире программирования вы ограничены только языком.

Фактически, никакого «перевода» в обычном понимании не существует. Вы ставите в конец транзакции любой код. Но какой замок поставить — решает только тот, кто имеет доступ к последнему «выходу».

Напомним, как мы отбираем транзакции: майнер, в принципе, может выбрать транзакции, какие захочет, но так как комиссии платятся майнеру, то он обычно отбирает те транзакции, где комиссии больше всего.

Итак, у нас есть список транзакций. И мы его передаём в блок.

Теперь рассмотрим последний элемент блока — его **заголовок**. Это самый интересный и важный элемент. Рассмотрим его максимально подробно.

Сначала покажем весь заголовок сразу.

Состав заголовка блока:

ВерсияХэшПрошлогоБлокаКореньДереваМеркла МеткаВремени ЦелевойУровеньСложности Nonce

Пример данных — никакой магии, просто подряд данные в 16-ричном формате[1]:

00023029C2D5479759CB2BBF3A6B8350D2B3D7D30B92EE81E23A000000000000000008BB1B50152C8EA76F0B9ADAB8C162DF82E60FBC71C56AFF553CF2DD9CE64A7076189F015B2F58E0A47HF281A

Кратко опишем каждый элемент.

[1] *Мы сделаем полный разбор конкретного блока в Приложении 1.*

СМИО ПРЕСС

Элементы блока

00023029	Версия (не номер блока, именно версия)	4 байта
C2D5479759CB2BBF3A6B8350D2B3D7D30B92EE81E23A0000000000000000000	Хэш прошлого блока	32 байта
8BB1B50152C8EA76F0B9ADAB8C162DF82E60FBC71C56AFF553CF2DD9CE64A707	Корень дерева Меркла	32 байта
6189F015	Метка времени	4 байта
B2F58E0A	Целевой уровень сложности (по сути, сколько нулей должно быть в хэше) — тут всё немного сложнее, мы упоминали, что мы не пишем количество нулей, мы пишем: меньше чего должно быть итоговое число	4 байта
47HF281A	Nonce — то, что должен подобрать майнер	4 байта

В Приложении 1 мы повторим весь этот путь на конкретном блоке, и мы надеемся, что вы пройдёте весь путь вместе с нами.

Рассмотрим каждый из этих элементов по отдельности.

Номер версии. 00023029. Как видно, всё в блоке биткоина очень серьёзно сжато, буквально нет ни одного лишнего символа. Поэтому эти 4 байта (соответственно, 8 символов, так как 1 байт кодирует 2 символа) стали использоваться огромным числом способов.

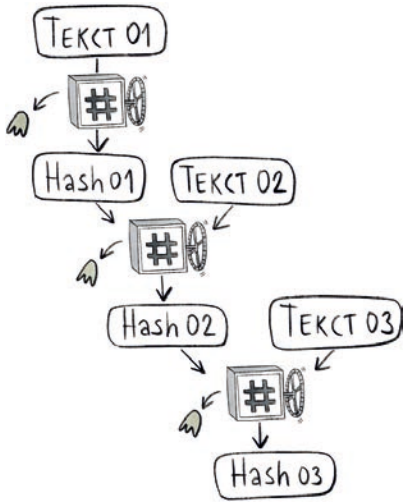
Как известно, компьютер работает только с 0 и 1. Один бит может принимать два вида: 0 или 1. Два бита могут принимать значения 00 или 01, или 10, или 11 — т. е. 4 разных вида. 4 бита могут принимать 16 разных значений, это наиболее удобный вид для пользователя и называется 16-ричной системой счисления. От 0 до F — называется символом.

Майнеры стали использовать эту часть в качестве «голосования» или «уведомления» о принятии или непринятии различных протоколов обновления (они называются BIP). Первая версия была 0x01, версия на конец 2021 г. — 0x20000004.

СМИО ПРЕСС

Версия — это первое, что майнер кладёт в свой блок. О сборке блока поговорим в следующей главе, посвящённой майнингу.

Хэш прошлого блока.
C2D5479759CB2BBF3A6B8350D2B3D7D30B92EE81E23A0000000000000000000



Помните идею хэшчейна? Мы считаем хэш прошлого блока частью нашего сообщения, чтобы при изменении прошлого блока хотя бы на один символ, изменилось и наше сообщение, а значит изменился хэш, а значит изменилось следующее сообщение. Напомним это рисунком 110 из главы 15.

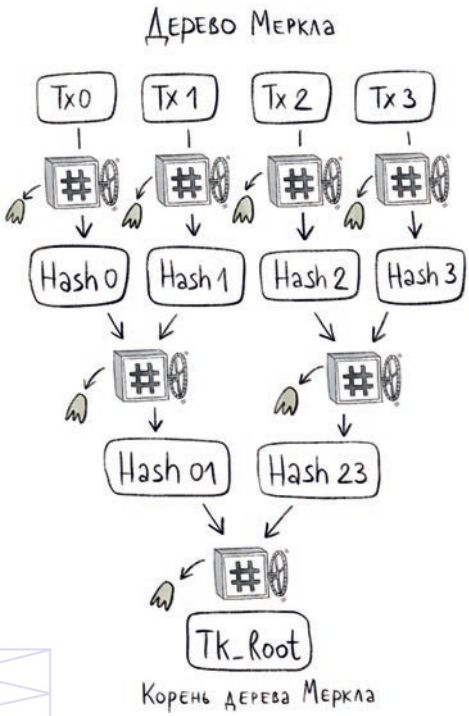


Рис. 127

Корень дерева Меркла
8BB1B50152C8EA76F0B9ADAB8C162DF8
2E60FBC71C56AFF553CF2DD9CE64A707

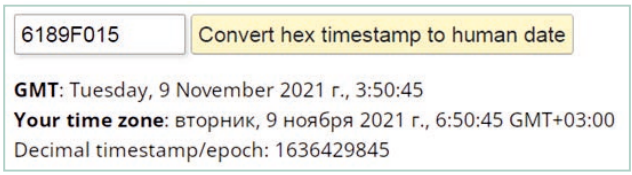
Схема дерева Меркла приведена на рис. 127.

Здесь в самом верху таблички мы видим Tx0… Tx3 — это транзакции, из которых мы берём хэш (который мы обсудили ранее, хэш — это отпечаток любого текста и фактически уникален). Затем мы связываем хэши попарно, т. е. хэш из Tx0 с хэшем из Tx1, Tx2 с Tx3 и так далее.

Связанные хэши мы также связываем попарно и так далее. В итоге у нас получается некий единый хэш — он и называется корнем дерева Меркла. Так как в формировании этого хэша участвуют все транзакции, он уникален именно для этого набора транзакций.

СМИО ПРЕСС

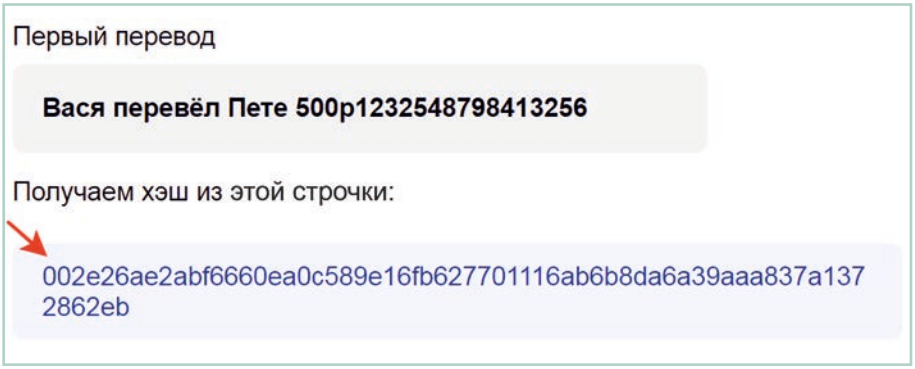
Далее обсудим метку времени.



Вид времени выглядит необычно — но это просто число. Вы можете его сконвертировать в более удобный формат на множестве сайтов. Нужно отметить, что метка времени, по сути, выставляется майнером, но выбор ограничен. Во-первых, время должно быть больше, чем среднее время последних 11 блоков (с учётом средней по времени генерации блока в 10 минут это даёт около часа в выборе времени).

Также время не должно быть больше, чем среднее время последних 11 блоков + 2 часа. Таким образом, метка времени имеет диапазон примерно в три часа.

Обсудим целевой уровень сложности. Здесь необходимо ещё раз вспомнить главы о хэшчейне и Proof of Work. У нас была цель: для того чтобы доказать серверу, что мы выполнили работу, нам нужно, чтобы хэш из письма + добавка имел несколько нулей в начале.



У майнера та же цель. Ему нужно найти такой блок, значение хэша которого будет меньше определённого числа (или, что то же самое, будет иметь определённое количество нулей в начале). В целевом уровне сложности мы кодируем именно эту информацию. То есть хэш блока, чтобы быть принятым сетью, должен быть меньше либо равен этому целевому значению.

Целевое значение автоматически рассчитывается всеми узлами сети в зависимости от времени создания последних блоков и сложности прошлого блока. Если блоки начинают добываться слишком медленно, то целевая сложность уменьшается. Если слишком быстро — увеличивается.

Сложность блока рассчитывается таким образом, чтобы каждый блок добывался около 10 минут. Целевой уровень сложности обновляется каждые 2016 блоков (т.е. примерно каждые две недели). Внутри этого периода все блоки будут иметь одинаковую сложность.

У читателей может возникнуть вопрос: как хэш из 32 байтов может быть меньше числа, в котором 8 байтов? Мы решили очень кратко рассмотреть этот момент, т. к. полный рассказ занял у нас около 4 страниц. Вкратце: мы сжимаем число, максимально удаляя из него информацию и оставляя, по сути, только порядок. То есть, вместо хранения всего числа, мы храним только количество значащих цифр числа (т. н. мантиссу).

Nonce. Единственная часть блока, которую мы не знаем до майнинга. Как мы только что сказали, майнер должен найти такой блок, хэш которого будет содержать определённое количество нулей в начале (на начало 2022 года — 21 ноль).

Выбрав транзакции, сформировав из них дерево Меркла, присоединив к нему другие данные, майнер может получить из них хэш. Маловероятно, что он будет содержать нужное количество нулей, так как хэш выдаёт, по сути, случайное число.

Тогда майнер начинает добавлять любые символы, соединяя их с блоком, чтобы итоговый хэш обладал нужными характеристиками[1].

Именно так мы делали в главе о Proof of Work. Мы советуем повторить этот опыт на нашем сайте.

То есть расчёт данного Nonce и является смыслом майнинга. При создании биткоина предполагалось, что только перебор возможных значений Nonce позволит находить нужный хэш.

Мы полностью разобрали состав блока и надеемся, что у читателя больше не возникнет вопросов по его составу.

---

[1] *Однако в связи с огромной текущей сложностью зачастую перебор всего Nonce не даёт нужного результата. Перебрав все варианты (все 4 байта), майнер может либо изменить транзакцию, что приведёт к изменению корня дерева Меркла, а значит, заголовка блока, что слишком дорого с ресурсной точки зрения, либо просто обновить время и начать перебор Nonce заново.*

СМИО
ПРЕСС

Глава 19
КОНСЕНСУС НА ОСНОВЕ PROOF OF WORK И НЕОБХОДИМОСТЬ МАЙНИНГА

Все аспекты майнинга мы рассмотрели в предыдущих главах, и по сути, в этой главе мы лишь ещё раз повторим пройденный материал, соединив всё то, о чём мы говорили раньше, а также тонкости и неожиданные проблемы майнинга.

Состав блока описан в предыдущей главе, а эта глава — тоже рассмотрение блока, но чуть под другим углом.

Майнер собирает блок из всех описанных выше частей.

Как мы помним, блок состоит из следующих элементов. Три технических поля (magic no, размер блока и количество транзакций) очевидны, опустим их в этом описании. Также, ещё до сборки блока, майнер знает практически все элементы будущего блока:

1. Текущую версию.
2. Целевое значение сложности.
3. Хэш предыдущего блока.
4. Время (как мы уже упоминали, майнер может менять его в интервале трёх часов).

Далее нам нужно разбираться с транзакциями. Мы включаем в блок те транзакции, которые нам наиболее выгодны (то есть те, в которых указана наибольшая комиссия). После этого мы можем начать считать дерево Меркла (напомним, что это хэш из хэшей транзакций).

Первое, что делает майнер, — создаёт «порождающую» (coinbase) транзакцию. Это то, ради чего мы пытаемся включить наш блок в блокчейн. Ценность, которая получается как бы «из ниоткуда».

Мы уже описывали coinbase-транзакцию, как создание сейфа с «родным» замком, на который позже будут навешиваться ключе-замки. Вся цепочка транзакций «выходит» из этой порождающей транзакции. Майнеру необходимо включить в блокчейн свой блок именно для того, чтобы в нём была его coinbase-транзакция!!!

То есть если ваш блок попадёт в блокчейн, то в него попадёт ваша coinbase-транзакция, и вы получите вознаграждение, т. е. условно «возникнет

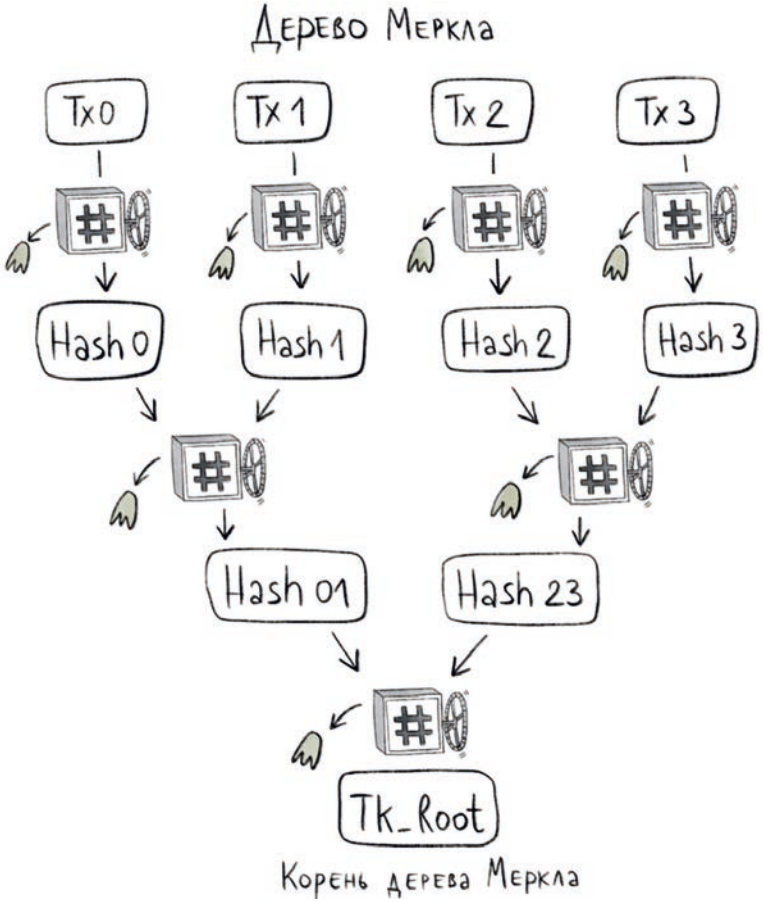
СМИО ПРЕСС

сейф», к которому у вас будет ключ доступа. На начало 2022 г. coinbase-транзакция — это 6.75 биткоина (или около 400 000$).

Создав свою порождающую транзакцию, вам необходимо выбрать из мемпула (всех ожидающих транзакций) транзакции, которые вы хотите включить в свой блок. Вы можете выбрать любые, но, как было сказано, ваша цель — заработать деньги, а значит, вы включаете те транзакции, за которые вы можете получить наибольшую комиссию. Мы уже обсуждали это в главе-сравнении о блокчейне острова Яп.

После того как вы сформировали список транзакций, которые вы хотите включить в свой блок:

1. Берём каждую транзакцию и берём из неё хэш.
2. Попарно связываем все получившиеся хэши, получая каждый раз хэш.
3. Повторяем процедуру, пока не находится последний хэш.
4. Этот хэш называется корнем дерева Меркла и записывается в заголовок блока.



Если мы добавляем (или меняем) даже один символ, хэш текста меняется непредсказуемым образом. Можем ещё раз повторить этот опыт на нашем сайте.

Данные	Хэш
Наша строка	db819ca4fd270acc33f75f545d7d1bd0a5742f64b7a6e036fc2025c361338698
Наша строка 1	a7dfa3c191705d824f7e0f34afe22acdf1c743963592ca141a294825cfda829e
Наша строка 2	50fb5fdfe89c1172d24d7234931522a7dc5363079cc9d475c2b9503b8c5d5158

Мы получили все данные из заголовка и теперь можем взять из него хэш. Очевидно, он не будет соответствовать правилу для включения в блок (у него не будет в начале нужного количества нулей), и тогда мы начинаем подбирать добавку к тексту, пытаясь добиться, чтобы хэш из этого текста начинался с 0 (или большего количества нулей). Возможно, вы помните наши опыты из главы о хэшчейне.

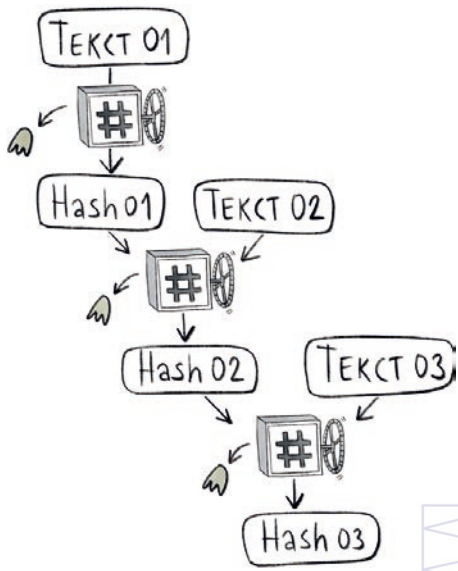
Первый перевод

Вася перевёл Пете 500р.

Получаем хэш из неё:

fdde2f241980c43853fd5146a14ba85691c42947b52d88d78177450720b98d78

Этот подбор называется работой Proof of Work и при большом количестве нулей может занять продолжительное время. Между тем проверка, что такая добавка действительно даёт определённое количество нулей в хэше, — очень быстрое и простое действие.



В главе 14 мы рассмотрели хэшчейн — цепочку блоков, связанных хэшами. Хэш предыдущего текста (блока) добавляется в текст следующего, что, разумеется, влияет на его хэш.

Майнер — это человек, который пытается подобрать такую добавку

(в терминах биткоина это называется Nonce), чтобы хэш содержал нужное количество нулей (на конец 2021 г. их должно быть 20).

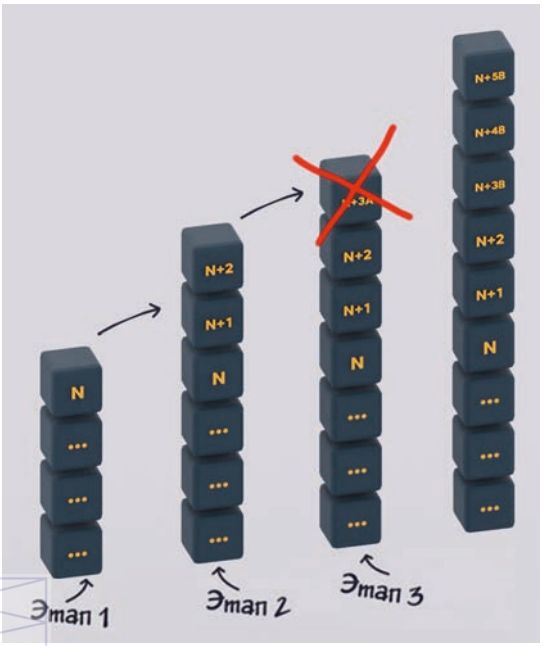
Ещё раз отметим — подобрать эту добавку, чтобы получить правильный хэш, можно только перебором. Нет никакой возможности создать аналитическое решение задачи.

Как мы упомянули в главе 17, правильным считается самая длинная цепочка блоков.

Сейчас мы можем уточнить, что важна не длина, а сложность этой цепочки. Нельзя создать миллион блоков с маленькой сложностью и переключить этим цепь на себя.

Как мы сказали, майнер подбирает Nonce с учётом поля «хэш предыдущего блока». Майнер может присоединить его только к блоку с этим хэшем. Если кто-то уже присоединил свой блок к этому блоку и распространил его по сети, то майнер должен бросить всю работу (т. к. узлы всё равно не примут его блок, ведь «место» уже занято) и пытаться майнить следующий блок после уже добытого кем-то.

Если же продолжить майнить этот блок, то, скорее всего, никто уже не будет майнить поверх вашего блока, и ваша цепочка не будет эталонной, а значит, никто не узнает о вашем решении задачи, и вы не получите вознаграждение. Ведь мы разбирали, что настоящей считается цепочка с самой большой суммарной сложностью.



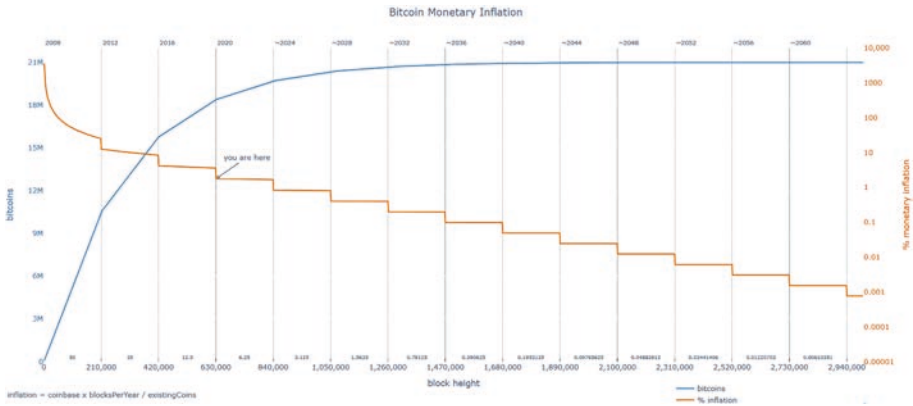
В этой ситуации никто не будет майнить блок для левой цепочки, который будет N + 4-м по порядку, а все будут майнить N + 6-й по порядку для правой.

СМИО ПРЕСС

Майнер получает за свою работу два вида вознаграждения.

1. Он получает плату от каждой транзакции (по их размеру он и включает транзакции в блок).
2. Его coinbase-транзакция включается в блок (фактически, он получает деньги). «Из ниоткуда» берётся определённое количество биткоинов, которое он шифрует своим закрытым ключом. Каков размер этого вознаграждения?

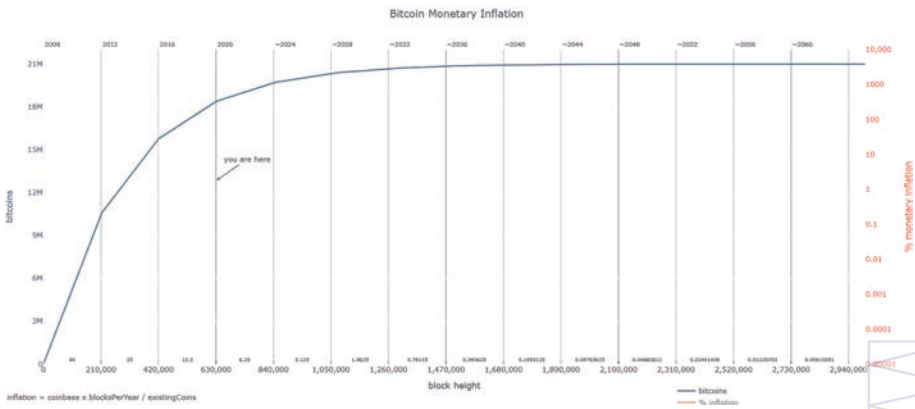
Отметка времени дана на начало 2022 г.



Примерно каждые 4 года размер вознаграждения падает вдвое.

Расписание	Месяц (спрогнозированно)	Вознаграждение
2009	03.01.2009	50 BTC
2012	28.11.2012	25 BTC
2016	09.07.2016	12,5 BTC
2020	11.05.2020	6,25 BTC
2024	11.05.2024	3,125 BTC
2028	01.05.2028	1,56 BTC

График количества добытых монет. Отметка времени дана на начало 2022 г.



Если на начало 2022 г. добыто около 17 млн монет, то за следующие 150 лет будет добыто ещё около 4 млн.

Возможность включить собственную coinbase-транзакцию — это не единственное преимущество. Также майнер получает вознаграждение от каждой транзакции, точнее, комиссии от всех, чьи транзакции он включил в блок. По сути, так транзакции и включаются в блок — те, у которых размер комиссии больше, чем у других. Напомним, комиссия устанавливается в качестве разницы между размером входящих и исходящих транзакций (размер исходящих должен быть меньше), как в нашем условном примере с камнями жителей острова Яп.

Глобальная идея при создании биткоина состояла в том, что если в начале майнеры получают фактически только вознаграждения, включённые в coinbase-транзакции, то далее их заработок будет состоять почти только из комиссий. Именно поэтому система биткоина настроена таким образом, что каждые 210 000 блоков (примерно раз в 4 года) награда сокращается вдвое. Эффект сокращения награды за блок называется халвингом.

Первые годы при создании биткоина (в том числе, когда майнингом занимался сам Сатоши Накамото) размер coinbase-транзакции составлял 50 биткоинов — и единственный майнер получал их каждый час (всего на кошельках Сатоши накоплено около миллиона биткоинов). На момент написания книги вознаграждение составляет 6,25 биткоина. При сохранении системы неизменной последний биткоин будет добыт в середине 2140 года, когда награда будет составлять 1 сатоши или 0.00000001 биткоина — далее награда будет менее 1 сатоши, т. е. сама система не будет её видеть.

При создании биткоина был установлен минимальный размер возможной суммы. Это одна стомиллионная или 0.00000001 биткоина. Позже этот размер был назван в честь основателя биткоина Сатоши.

В этом случае всего будет добыто 21 млн монет. Некоторые системы, к примеру, эфириум, не имеют такого ограничения, т. к. не имеют халвинга[1]. А некоторые имеют совершенно иную систему вознаграждений, и даже наоборот — изначально добыты все монеты, но они постепенно сгорают во время транзакций.

Рассмотрим вознаграждение за транзакцию на конкретном примере. Когда вы хотите перевести что-то в криптовалюте (к примеру, в биткоине), вы задаёте определённую сумму, которую вы готовы отдать майнеру. В одном из популярных кошельков — Электруме — это выглядит так (рис. 128).

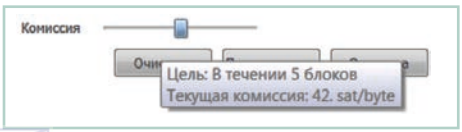


Рис. 128

[1] *Халвинг (см. выше) – периодическое уменьшение размера вознаграждения.*

СМИО ПРЕСС

Мы можем менять размер комиссии (рис. 129), и нам говорят, когда примерно наша транзакция попадёт в блокчейн биткоина.

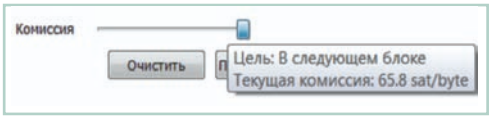


Рис. 129

Как мы видим, стоимость транзакции определяется её размером. Размер блока ограничен, и майнер хочет получить максимальную сумму, поэтому транзакции большего размера стоят дороже, соответственно, стоимость указана за включение байта информации в блок.

Напоминаем, что для выгоды майнеров, в блокчейн попадают не те транзакции, которые были предложены раньше, а те, на которых установлена наибольшая комиссия.

Как уже было сказано, в биткоине рассчитывают хэши от SHA256 — самым известным способом хэширования в криптомире. На нём работает биткоин и его форки, а также большинство других криптовалют.

Форк, или «вилка», — производная криптовалюта, за основу которой взята другая криптовалюта. В более широком значении, это любое разделение программного продукта на две несвязанные ветки, которые могут «жить» совершенно обособленно и над которыми могут работать разные команды программистов.

Просто приведём неполный список, чтобы вы оценили количество таких криптовалют (хотя большинство из них являются так называемыми шиткоинами, т. е. не имеющими никакого практического смысла):

AmoveoVEO, Argentum, (SHA256)ARG, Auroracoin, (SHA256)AUR, AurumCoinAU, BADcoin, (SHA256)BAD, Bitcoin, AtomBCA, Bitcoin, Cash, ABCBCHA, Bitcoin, CashBCH, Bitcoin, SVBSV, BitcoinBTC, BitcoinRXBR, BitcoinVaultBTCV, BitcoinVGBVG, BitflateBFL, BitTokenBXT, BlastBLAST, ButKoin, (SHA256)BUT, CheetahcoinCHTA, DanecoinDANE, Deutsche, eMarkDEM, DeVaultDVT, DigiByte, (SHA256)DGB, ElastosELA, EmercoinEMC, ErgonXRG, FreeCashFCH, FxTC, (SHA256)FXTC, Gleec, CoinGLEEC, Globaltoken, (SHA256)GLT, HathorHTR, iBitcoiniBTC, JouleCoinXJO, KatzcoinKAT, LanaCoinLANA, CashLCC, Long, coinLONG, LucentLCNT, MartkistMARTK, MazaCoinMZC, Myriad, (SHA256)XMY, NamecoinNMC, NEXT.ChainNEXT, ParallelCoin, (SHA256)DUO, PeercoinPPC, Pyrk, (SHA256)PYRK, Quebecoin, (SHA256)QBC, Smileycoin, (SHA256)SMLY, StrongHandsSHND, Super, BitcoinSBTC, SusucoinSUSU, SyscoinSYS, TDCoinTDC, TerraCoinTRC, TitcoinTIT, United, BitcoinUBTC, UnobtaniumUNO, VCashVCASH, Veil, (SHA256)VEIL, Veles, (SHA256)VLS, WidecoinWCN, Xaya, (SHA256)CHI, Xazab, (SHA256)XAZAB, ZetaCoinZET

Помните, в главе о хэшировании мы перечисляли требования в хэш-функциям, и одним из требований было максимально простой расчёт хэш-функций? Для простоты SHA256 состоит только из битовых операций.

СМИО ПРЕСС

Рис. 130

Пример ASICa за 1000 000 р.

Сейчас одно из главных преимуществ — простота — становится недостатком. Если изначально предполагалось, что майнить будут люди на собственных компьютерах, то сейчас созданы специальные устройства (называемые ASIC[1], или у нас ласково — *асиками*), которые ничего не умеют делать, кроме битовых операций, поэтому делают их гораздо качественнее, хотя и не предназначены ни для чего другого (рис. 130).

Компьютеры не могут конкурировать с такими устройствами, поэтому сейчас майнят только на специальных майнинг-фермах, а главными затратами стали не покупка компьютера, а электричество, что ещё более усугубляет централизацию. Майнинг-фермы располагаются там, где дешёвое или вообще бесплатное электричество. К примеру, до запрета майнинга в Китае, основные мировые майнинг-фермы располагались на избыточных мощностях около китайских ГЭС.

https://ccaf.io/cbeci/mining_map

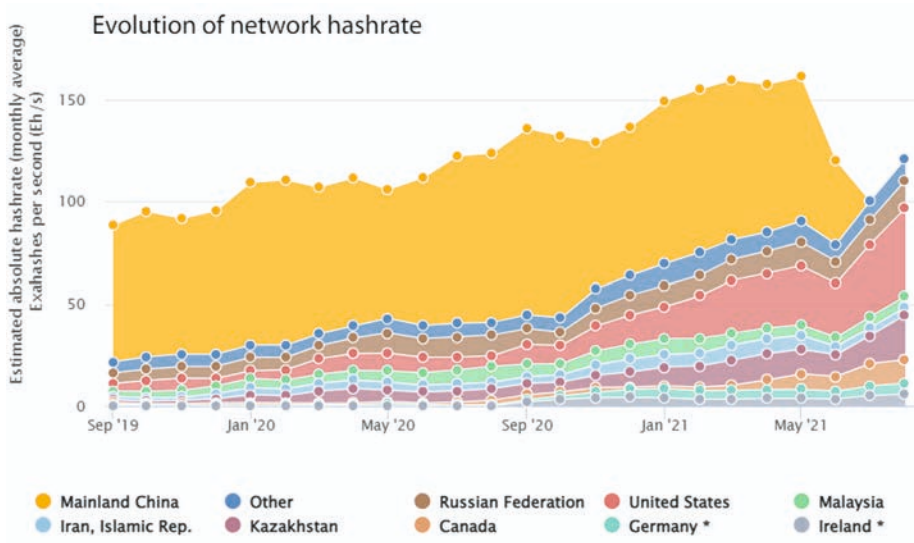


График изменения майнинга по странам (по вертикальной оси отложено число переборов возможных значений Nonce, рассчитанное в эксо-хэшах[2])

[1] *ASIC (аббревиатура от англ. application-specific integrated circuit, «интегральная схема для конкретного применения»).*

[2] *Эксо – это 10^{18} или 1000 000 000 000 000 000.*

СМИО ПРЕСС

Создатели других криптовалют сразу увидели проблемы централизации и попытались создать что-то такое, что будет препятствовать созданию «асиков». Лайткоин, к примеру, перешёл на Scrypt, расчёт которого требует оперативной памяти. Эфириум — на модифицированную версию SHA3 (KECCAK256).

Некоторые криптовалюты используют сразу несколько хэш-функций. А некоторые распределяют вознаграждение не только между майнерами, но и между владельцами нод (узлов).

В общем, многие пытаются создать такую систему, майнинг которой наиболее выгодно мог бы происходить на обычном домашнем компьютере. Тогда майнерами могли бы стать десятки миллионов людей (мы помним, что ценность денег тем больше, чем больше людей признают эти деньги), которые бы оставляли майнинг-программу на компьютере, не выключая его на ночь. Тогда сеть была бы полностью децентрализована, а атака на неё со стороны даже самого могущественного государства или организации стала бы невозможной.

Минусы централизации. Биткоин возник в ответ на запрос людей. Если бы такого запроса не было, биткоин не смог бы реализоваться. Так в чем состоит этот запрос? Чтобы никто не мог произвольно блокировать переводы, а также снимать деньги со счетов. Также был необходим надёжный, неблокируемый канал связи. Биткоин не создавался как анонимная платёжная система и не приспособлен к этому, вся суть биткоина именно в децентрализации. В первую очередь биткоин создавался как способ перевода между странами и неэффективен при оплате внутри страны.

К 2008 году (а тем более в 2022) доллар перестал быть удобным средством платежа из-за серьёзных ограничений и непредсказуемых правил, в том числе возможности отключения от переводов целых стран.

Соответственно, минусы централизации состоят в том, что существует сила, способная заблокировать или отменить транзакции.

Правда, у децентрализации есть и противники. Если майнить будет выгодно на компьютерах, то увеличится количество ботнет-писателей, которые будут пытаться получить доступ к компьютерам.

Ещё одна интересная идея состоит в том, что если мощность компьютера будет легко монетизировать, то тогда будет легко создать платные сайты, при посещении которых вы сможете не платить за его посещение, а будете всего лишь должны разрешить программе работать в фоновом режиме (к примеру, пока вы смотрите фильм), используя незадействованную вычислительную мощность вашего компьютера. В общем, развитие технологии может иметь совершенно неожиданные результаты.

Авторам хотелось бы напомнить о том, что не стоит без подготовки идти в майнеры. Сейчас это большой бизнес, и если у вас нет опыта, то не надо тратить деньги на создание домашней майнинг-фермы и пытаться «быстро заработать» — скорее всего, вы только потеряете время и деньги.

СМИО ПРЕСС

Известно много случаев, когда люди покупали в кредит видеокарты, а потом не могли вернуть эти кредиты. При этом разные криптовалюты используют разные хэши, а значит, для майнинга каждой криптовалюты необходимо использовать разное оборудование. Приведём некоторый список валют и хэшей, которые в них используются:

Bitcoin — SHA256.
Ethereum — Ethash
Bitcoin Cash — SHA256
Dash — X11
Litecoin — Scrypt
Monero — CryptoNight
Ethereum Classic — Ethash
Zcash — Equihash
Hshare — X13
BitConnect — Scrypt
Komodo — Equihash
Bytecoin — CryptoNight
Decred — Blake256r14
MonaCoin — Lyra2REv2
Dogecoin — Scrypt
Vertcoin — Lyra2REv2
Siacoin — Blake (2b)
BitcoinDark —SHA256
Metaverse ETP —Ethash
GameCredits — Scrypt

И так далее. Мы можем видеть, что разнообразие очень большое. И устройство для майнинга одного типа хэшей не подойдёт для другого.

На этом сайте вы можете узнать примеры разных асиков и прогнозируемую прибыль по ним на разных криптовалютах:

https://minerstat.com/hardware

Хотелось бы также обратиться к любителям лотерей. Если вы верите в свою удачу, то вместо покупки лотерейных билетов, вы можете просто запустить на своём компьютере программу майнинга биткоина (если очень верите в удачу) или другой валюты (чем меньше верите в удачу, тем менее популярную валюту берите). В случае если ваш компьютер найдёт нужный Nonce (а это зависит только от удачи), вы получите около 400 000$ (на конец 2021 года). Розыгрыш происходит примерно раз в 10 минут, а затраты — только электроэнергия на работу вашего компьютера. Шанс же выиграть — примерно как в лотерее.

СМИО ПРЕСС

Учитывая современную левую повестку, нельзя не упомянуть о главной проблеме майнинга Proof of Work — работа больших мощных устройств фактически тратится впустую на расчёт ненужных хэшей. С точки зрения экологии, это не очень хорошо (хотя оценки тут разные). Существует запрос на поиск полезных расчётов, которые можно делать при майнинге, — таких, как нахождение простых чисел или даже расчёт специальных белков. Но пока эти идеи реализовать не удалось.

Выходя за рамки главы (этому посвящено одно из Приложений), можно упомянуть о попытках построить консенсус на других принципах. Их уже существует несколько десятков. Proof of Work можно понимать как награду тому, кто лучше всех работал. Но можно давать возможность формировать блок тому, у кого больше всех денег, или кто больше всех совершил транзакций, или ещё по какому-нибудь принципу. Но все они подвержены атакам гораздо больше, чем Proof of Work. Также только Proof of Work не даёт возможности взломать сеть. Остальные системы делают взлом сети невыгодным, но всё равно возможным.

СМИО
ПРЕСС

Глава 20
ДАЛЬНЕЙШЕЕ РАЗВИТИЕ

В результате изучения различных криптовалют напрашивается вывод, что развитие криптовалют застопорилось, а главный недостаток блокчейна — необходимость записи и хранения всех транзакций.

Все попытки создать что-то новое, предложить какой-то иной принцип и улучшить систему биткоина, не приводят к прорыву. Эфир пытался предложить любые смарт-контракты (то, что в книге мы называем замками в UTXO), но попытки выпуска акций (ICO) практически полностью встали. Надёжность таких смарт-контрактов под большим вопросом.

Переход на иные типы майнинга, по сути, приводит к возврату централизации. Теоретически невозможно представить ситуацию, в которой можно переводить деньги так же быстро и просто, как в централизованных системах.

Биткоин полностью решает проблему перевода крупных сумм. Транзакции невозможно отменить, перехватить, заблокировать. Но перевод небольших сумм приводит к серьёзным проблемам. Во-первых, достаточно высокая стоимость транзакции (в среднем около 10$), во-вторых, очень длительное время подтверждения (минимум 10 минут, если мы верим переводу, сразу же включённому в блок), что очень дорого и медленно. Конечно, есть вариант, когда мы верим транзакции ещё до попадания в блок, но тогда нам необходимо хотя бы немного доверять человеку. И это не решает проблему высокой цены даже минимального перевода денег.

То есть теперь от криптовалют требуется создать систему, когда любой человек сможет перевести маленькую сумму мгновенно. Но это кажется невозможным, т. к. любая транзакция должна попасть в блокчейн и храниться там вечно. А значит, невозможно сделать её быстрой и очень дешёвой. Подсчёты показывают, что блокчейн биткоина не сможет обеспечить больше 7 транзакций в секунду. А, например, Visa может обрабатывать тысячи транзакций в секунду (официально 24 000, хотя это число многие ставят под сомнение).

Попытки увеличить количество транзакций приводят или к огромному росту блокчейна (как у Bitcoin Cash), или к сильнейшей централизации, как у NEO или других подобных валют. Попытки перейти на Proof of Stake, ког-

СМИО ПРЕСС

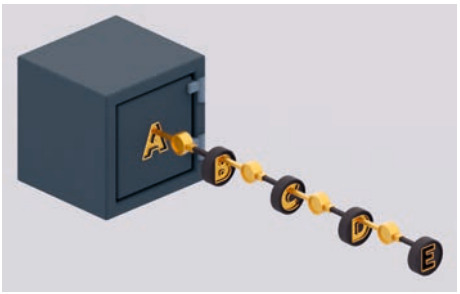
да новый блок создаётся тем, у кого больше денег, или даже Proof of Authority, когда новый блок создаётся «уважаемым» человеком, приводят к ещё большей централизации.

Ещё какое-то небольшое время назад казалось, что выхода нет. Но на сцену выходит изобретение, которое переворачивает саму идею перевода средств внутри блокчейна. На наш взгляд, это ярчайший пример того, как человечество находит ответ на, казалось бы, неразрешимые вопросы.

Наша история начиналась с того, как человек придумал невозможное, — сделал систему шифрования, когда человек знает, как зашифровать, но не может расшифровать. Далее биткоин показал, что можно передать ценность, которая составляет лишь нули и единицы внутри компьютера, но эту ценность нельзя скопировать. Только передать!!!

Завершим наше повествование очередным изобретением, которое ранее казалось невозможным: как передать ценность внутри блокчейна (и опираясь на блокчейн), но не записывая в блокчейн информацию об этой транзакции. Иначе говоря, как передать что-то мгновенно и практически бесплатно.

В этой главе мы пройдём через несколько решений этой проблемы — сначала не очень выгодным (но простым) способом, но далее усложняя нашу систему и добиваясь лучшего результата.



Надеемся, читатель хорошо запомнил идею UTXO: любая транзакция опирается на прошлую транзакцию, и всю систему можно представить как линии, которые начинаются с coinbase-транзакций, а заканчиваются замками, которые вешает тот, кто имел ключ к предыдущему замку.

Тем не менее, мы также помним, что наш замок — это набор правил, по которым будет проверяться ключ. Мы подробно разбирали это в главе 14 о скриптовом языке. Но если у нас проверка данных ключа идёт с помощью скриптового языка, мы можем вписать в этот язык некоторые дополнительные возможности.

Итак, представим себе сначала простейшую ситуацию, когда Соня хочет сделать множество покупок у Ани, но отдавая деньги за каждую покупку отдельно и имея возможность в любой момент прекратить делать покупки. Каждая запись требует включения в блокчейн, поэтому стоит достаточно дорого, поэтому Соня идёт другим путём.

Она имеет доступ к сейфу с 1 биткоином (1000 миллибиткоинов) и отпирает его (т.е. она создаёт транзакцию или вставляет ключе-замок). Но при этом замок к своему ключу делает не обычным, а с двумя скважинами.

Ведь Соня — полная хозяйка замка, а тот, чей ключ подошёл к прошлому замку, может создать любой новый замок (в рамках языка программирования).

И Соня создаёт два ключа, такие, что замок можно открыть в следующих случаях: либо с момента вставки ключа прошло определённое количество времени (и тогда достаточно любого из ключей Ани или Сони), либо нужны сразу оба ключа. При этом после «открывания замка» любым из этих способов контроль над всеми деньгами транзакции получает Соня (тот, кто её отправил). Таким образом, пока отправки денег не произошло.

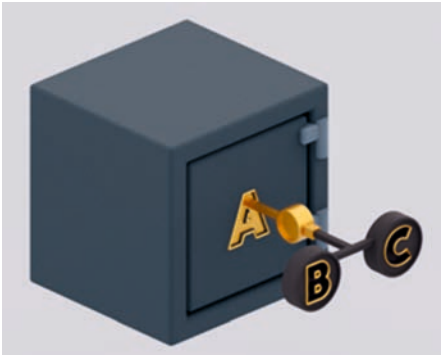


Рис. 131

Итак, у нас замок с двумя скважинами и два ключа, которые подходят каждый под свою скважину (рис. 131).

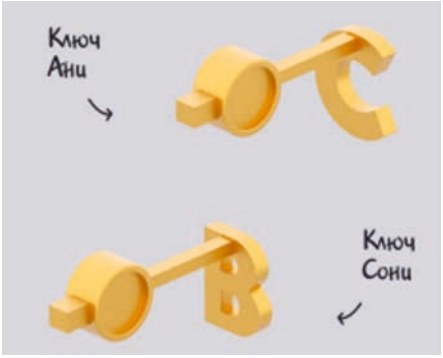


Рис. 132

Замок откроется в случае, если (рис. 132):

1. Будут вставлены оба ключа.
2. Будет вставлен один ключ — но тогда нужно ждать 1000 блоков для открытия.

При этом тот, кто вставил ключ, получит свои деньги только после возникновения 1000 блоков. А тот, чей ключ не вставлялся, получит их сразу же.

Всегда нужно помнить, что мы сейчас упрощаем то, что происходит на самом деле: замок — это программа, которая обрабатывает данные (а ключ — это данные). Данные должны быть такими, чтобы после обработки программа выдала OK. И программа полностью открыта для просмотра любому желающему.

То есть, если транзакция будет создана, но потом никаких действий не будет совершено, то (по условиям транзакции), к примеру, через 1000 блоков или 7 дней транзакция будет завершена (Соня потеряет комиссию за включение в блок).

Пока всё выглядит бессмысленным — Соня открыла транзакцию на 1 биткоин и, что бы ни случилось, она получает его обратно. Но теперь мы переходим к сути процедуры.

СМИО ПРЕСС

Соня совершает покупку на 0.001 биткоин. В нашей ситуации это означает, что Соня создаёт новый ключе-замок. Если он будет вставлен в скважину, то Соня получит уже не 1 биткоин, а лишь 0.999, а Аня через 999 блоков получит «0.001 биткоин».

То есть теперь у Ани два ключа — по первому она получает 0 через 1000 блоков (Соня сразу же — 1 биткоин), по второму она получает 0.001 биткоина через 999 блоков, Соня — 0.999 биткоина сразу же.

У Сони тоже два ключа: по первому она получает 1 биткоин через 1000 блоков (Аня 0 сразу же), по второму — 0.999 через 999 блоков (Аня получает 0.001 сразу же).

То есть, если раньше у Ани (тот, кто не создавал транзакцию, транзакция и деньги были Сониными) был только один ключ, который просто разрушал транзакцию, то теперь ей передали ключ Сони, имеющий особые настройки, — при его использовании Соня получает только часть того, что хранится под замком.

Далее Соня передаёт Ане ключ, который при использовании вернёт Соне только 0.998 биткоина через 998 блоков, а 0.02 биткоина Ане и так далее.

Каждый раз при покупке Соня передаёт возможность Ане получить определённую сумму раньше, чем будет доступно предыдущее разрешение.

При этом помним, что у Сони остаётся первоначальный ключ, вставив который она даёт замку сигнал «Ане ничего не дай, а мне верни всё через 1000 блоков». Почему она не может им воспользоваться? Потому что эта запись будет видна в блокчейне, и Аня сможет успеть воспользоваться своим ключом.

То есть Соня при каждой покупке передавала ключи Ане — каждый раз новый ключ и каждый раз новое правило срабатываний (рис. 133).

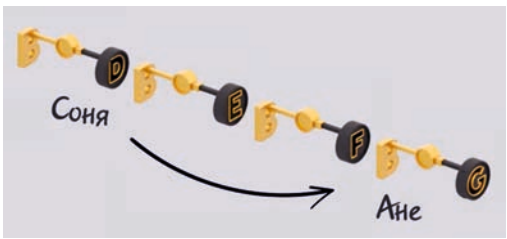


Рис. 133

Ане (тому, кому идут мелкие транзакции) не выгодно пользоваться полученным ключом. Таким образом она разрушит канал и лишит Соню возможности постоянно платить деньги. Соне невыгодно разрушать канал по той же причине.

При попытке смошенничать (т. е. использовать не самый выгодный Ане замок), Аня успеет воспользоваться тем замком, который самый выгодный, потому что он по своим настройкам быстрее всего вернёт девушкам их доли денег.

Понятно, что, с одной стороны, ситуация не сильно улучшается. С другой стороны, мы получили представление, что можно создать транзакции с мгновенной передачей средств и которые полностью бесплатны. То есть платится только за создание «канала» и «перевод на 1 биткоин», который записывается в блокчейн.

СМИО ПРЕСС

Можно привести следующую аналогию.

Боря и Максим приходят в банк, и Максим кладёт в ячейку 10 000 рублей. Банкир даёт Максиму документ, по которому он может получить содержимое ячейки через 100 дней после обращения с этим документом, а также даёт пачку бланков-приказов ***отдать Борису*** *определённую сумму через определённое время (и время, и сумма вписываются вручную, а имя Бориса уже впечатано в бланк).*

Максим теперь может создать приказ (и послать его Борису), по которому Борису отдадут деньги. Например, он пишет «отдать 100 рублей через 90 дней». Теперь Борис может прийти в банк, отдать полученный от Максима приказ и получить 100 рублей через 90 дней.

Казалось бы, у Максима остался документ, выданный банком, по которому он может получить 10 000 рублей через 100 дней после обращения с этим документом за деньгами. Но в действительности Максим не может их получить, т. к. если он отдаст этот документ банкиру, то Борис придёт на следующий день и потребует 100 р., которые ему отдадут через 90 дней (раньше, чем Максим получит 10 000).

Далее, совершая очередной перевод 200 рублей Борису, Максим просто отдаёт ему документ, по которому он может получить 300 (= 200 + 100) рублей через 88 дней (что меньше, чем 90 дней в первом приказе) и так далее. Банк исполняет только один приказ. То есть если Борис принёс приказ о получении 300 рублей, то он получит эти 300 рублей через 88 дней. Максим же получит остаток немедленно. И договор в этот момент прекращается.

Фактически, Максим совершает переводы Борису, даже не уведомляя об этом банк. А значит, не нужно платить банку за этот перевод. Банк получит за транзакцию только один раз при создании договора.

Эту систему можно использовать в любых ситуациях, когда известно, что один человек постоянно пересылает деньги другому человеку (и при этом не требуется доверия). Эта идея была реализована очень давно в попытке создать сервис потокового видео с оплатой на блокчейне. И данный инструмент вполне подходил для этой цели.

Однако минусы очень значительны.

1. Если один из контрагентов вдруг пропадёт, то второму для получения денег потребуется ждать окончания установленного времени (*срока контракта*).
2. Каждая транзакция уменьшает срок контракта.
3. Если Аня замешкается и не будет запрашивать свои деньги через 998 блоков, через 1000 Соня получит все деньги назад. Соне, в свою очередь, ничего не угрожает. Если она пропадёт, Аня получит свои 0.02 биткоина, а к ней вернутся её 0.98.

И в этих пунктах кроется противоречие: длинный срок будет ухудшать первый пункт (если человек пропадёт, второму дольше ждать своих денег).

СМИО ПРЕСС

Короткий срок будет уменьшать количество возможных транзакций (т.к. чем меньше срок, тем меньше транзакций можно сделать).

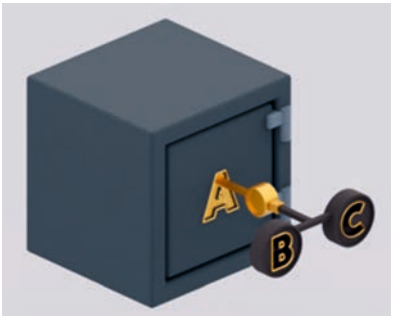
Как же решить проблему? Хотелось бы отменять вообще предыдущее состояние.

В предыдущем примере у нас был «умный» замок, который просто смотрел подпись и передавал Ане столько, сколько разрешила Соня и когда разрешила. А если ничего не было предъявлено, то Соня получала всё назад.

А теперь мы хотим, чтобы предыдущего состояния не было. Но это, казалось бы, полностью противоречит идее блокчейна, когда то, что попало в блокчейн, невозможно изменить или удалить!

Решается вопрос очень интересно. Предположим, у нас есть два субъекта, Боря и Настя, которые постоянно пересылают друг другу деньги. Они создают платёжный канал, вкладывая каждый в него по 10 биткоинов.

Они создают транзакцию, выходы которой (замки) вновь будут выглядеть как на рис. 131.



У замка два отверстия

У Бори есть ключ, который, будучи вставлен в замок, делает следующее:

1. Настя сразу же получает свою долю (при создании канала это 10 биткоинов) назад.
2. Боря (хозяин ключа) получает свою долю назад через 1000 блоков ПОСЛЕ ВСТАВКИ КЛЮЧА.
3. Если у Насти есть Борин секретный код к этой транзакции, а Боря вставил ключ, она получает все деньги Бори (пока кода у Насти нет, но она может получить его от Бори).

Соответственно, у Насти есть ключ, который делает обратное:

1. Боря сразу же получает свою долю (при создании канала это 10 биткоинов) назад.
2. Настя (хозяин ключа) получает свои биткоины назад через 1000 блоков.
3. Если у Бори есть секретный код Насти к этой транзакции, и она вставила свой ключ, он получает все деньги Насти (пока его у Бори нет, но он может получить его от Насти).

Понятно, что если Настя не выходит на связь, Боря вставляет свой ключ в замок и разрывает транзакцию. Через 1000 блоков получает свои деньги назад. И наоборот.

Теперь Боря хочет переслать 1 биткоин Насте. Он отдаёт Насте свой секретный код и этим лишает себя возможности вставить свой ключ в замок (если он вставит ключ в замок, Настя отберёт у него все деньги).

СМИО ПРЕСС

Настя передаёт Боре новый ключ, который он может вставить в замок, и при этом:

1. Настя сразу же получает свою долю (теперь это 11 биткоинов) назад.
2. Боря (хозяин ключа) получает свою долю (9 биткоинов) назад через 1000 блоков.
3. Если у Насти есть Борин секретный код к этому новому ключу, она получает все деньги Бори (это новый ключ и к нему создаётся новый секретный код).

Боря передаёт Насте вместе со своим секретным кодом тоже новый ключ, тогда:

1. Если Настя использует свой новый ключ, Боря получает свои 9 биткоинов.
2. Настя получает назад 11 биткоинов через 1000 блоков.
3. Новый секретный код, который позволит Насте получить все деньги Бори (если был использован его ключ).

Таким образом, есть замок, в который можно вставить ключ, и он распределит деньги определённым образом, тем самым создав два выхода UTXO.

Рассмотрим попытку мошенничества на этом этапе. К примеру, Боря решил, что Настя — глупая девочка, которая ничего не понимает. Она видит у себя в канале 11 биткоинов и не посмотрит в кошелёк.

Боря берёт свой старый ключ, который создавался при первоначальной транзакции. А именно тот, который даст ему 10 биткоинов. И вставляет свой ключ в замок, и замок создаёт две UTXO-транзакции. Одна UTXO в 10 биткоинов — Насте и другая — на 10 биткоинов — там тоже замок с двумя ключами — ожидает 1000 блоков или код, который создал Борис (этот код он отдал Насте, и при вводе этого кода все деньги уйдут Насте).

Но, как всегда, он просто недооценил девушку! Разумеется, Настя всё поняла, увидела проблему и вставила в этот замок ключ с кодом, который она получила к этой транзакции. В итоге, все деньги Бори оказались у Насти.

То же самое у Насти. Если Настя пересылает деньги Боре — соответственно, происходит зеркальная ситуация.

Насте тоже не имело смысла вставлять ключ предыдущей транзакции, так как в этом случае она получила бы меньше, чем ключом, который у неё есть (логичнее вставить последний ключ).

Ещё раз отметим: после первой транзакции у Бори есть новый ключ, чтобы получить 9 биткоинов, у Насти ключ, чтобы получить 11 биткоинов, а также секретный код, который при попытке использовать более выгодный Боре ключ (первый ключ, который создавался первоначально) отнимает у него все деньги!

Это очень непростая идея. В блокчейне остаётся право у каждого получить по 10 биткоинов, но фактически это становится невозможным, так как при попытке вставить любой ключ, кроме последнего, оппонент сможет отобрать

СМИО ПРЕСС

все деньги. Либо, в случае первой транзакции, это просто невыгодно, т. к. Настя получит только 10 биткоинов, да и то только через 1000 транзакций, хотя по новым обязательствам ей принадлежит уже 11 биткоинов.

Теперь передача денег будет означать передачу друг другу секретных кодов к предыдущей транзакции + передачу ключей к текущей. И так любое количество раз.

Но неужели нам придётся создавать для каждой пары свой канал? Нет.

Предположим, Боря хочет дать денег Фиме и у него нет общего с ним канала. Но есть общий канал с Настей. А у неё — общий канал с Фимой. И в каждом канале по 1 биткоину с каждой стороны (рис. 134).

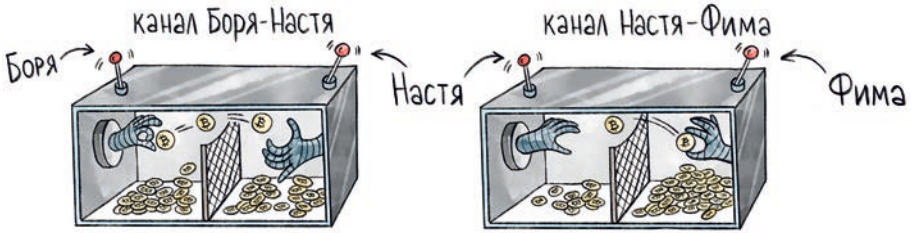


Рис. 134

Что тогда делать? Просто добавить в наш «замок» ещё один уровень.

Сначала Фима генерирует большое случайное число и получает из него хэш (мы помним, что по хэшу невозможно восстановить исходное число, а перебор никогда не станет возможен для чисел больше 2^{256} (см. Приложение 5). Этот хэш он передаёт Боре (рис. 135).

Рис. 135

Боря получает хэш от числа и создаёт внутри уже знакомого нам условия «я передаю Насте 1 биткоин» дополнительное условие «я передаю Насте 1,002 биткоина, если она скажет число, хэш которого равен этому числу до блока (номер текущего + 10)». То есть в наших терминах замков и ключей Боря создаёт не тот замок, о котором мы говорили ранее, а замок с дополнительным требованием.

Транзакция переходит к Насте в случае, если она предоставит число с таким хэшем. Настя внутри канала с Фимой создаёт новую транзакцию с правилом: «Я передаю Фиме 1,001 биткоина, если он скажет число, хэш которого

СМИО ПРЕСС

равен этому числу до блока (номер текущего + 8)», внутри канала Настя создаёт условие: «Если Фима скажет мне число, хэш которого равен этому числу до блока (номер текущего + 7), я передаю ему 1 биткоин».

Если Фима не скажет число — ничего, после 7 блоков Настино условие исполниться не может, а значит не будет выполнено условие Бори. Все останутся при своих. А если Фима передаст Ане число, он получит 1 биткоин. Настя передаст число Боре и получит 1.001 биткоин (рис. 136).

Распишем этапы ещё раз:

Рис. 136

1. Фима создаёт большое случайное число и берёт из него хэш.
2. Даёт хэш Боре.
3. Боря вставляет в автомат программу «Если ввести число, которое при хэшировании даёт то, что передал Фима, значит, передаём Насте 1.001 биткоин».
4. Даёт хэш Насте.
5. Настя видит, что Боря создал такое условие, и вводит в «свою» руку робота такое же условие «Если ввести число, которое при хэшировании даёт то, что передал Боря, значит, передаём Фиме 1 биткоин».
6. Фима видит условие, вводит число в робота (его видит и Настя и получает 1 биткоин).
7. Настя вводит число в автомат и получает 1.001 биткоин.

Транзакция проходит мгновенно, через любое количество «узлов», при этом каждый узел получает какую-то минимальную оплату. Недостаток лишь в том, что в каждом узле сети должно быть достаточно денег до транзакции, поскольку сначала деньги от Ани идут Фиме — у Ани должен быть 1 биткоин в совместном с Фимой узле. Далее у Насти должен быть 1.001 биткоин в совместном с Аней узле, у Бори 1.002 биткоина в совместном с Настей узле.

В будущем, с приходом на этот рынок корпораций, ситуация может измениться. Может быть создан один узел, с которым каждый человек сможет создать канал. К примеру, какой-нибудь Илон Маск создаст канал, к которому захочет подключиться каждый.

У нас есть общий узел, который создал каналы с 5 людьми. Каждый вложил в него определённое количество биткоинов. Предположим, Максим платит

СМИО ПРЕСС

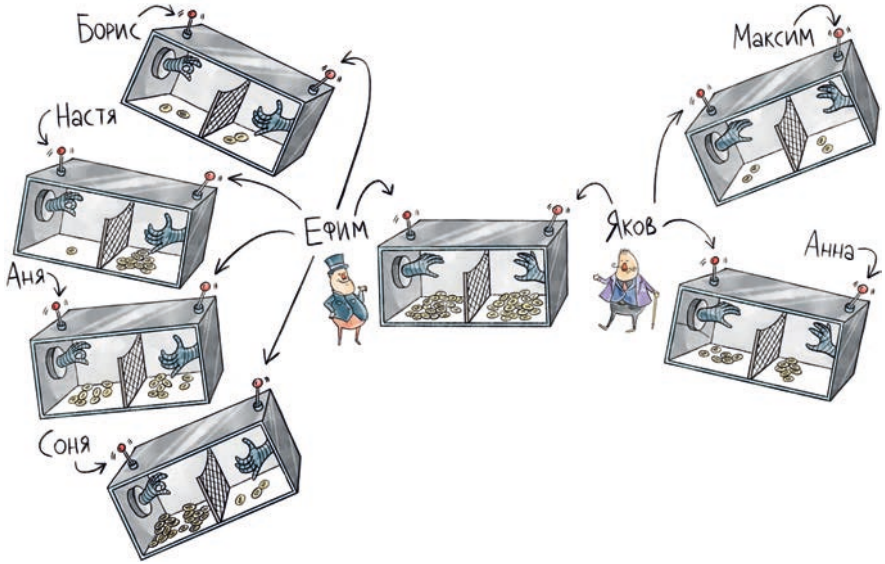


Рис. 137

Соне 1 биткоин. Тогда он переводит Якову 1.00002 биткоина (внутри их общего узла), Яков переводит Ефиму 1.00001 биткоин, 0.00001 биткоин взимает в качестве комиссии. Далее Ефим переводит Соне 1 биткоин, соответственно также беря комиссию 0.00001 биткоин.

Каждая транзакция защищена и доверие к общему узлу не требуется.

Данная сеть называется Lightning Network. На конец 2021 года топология сети выглядела следующим образом (рис. 138):

Рис. 138

Твиттер сети:
https://twitter.com/LNstat

Сеть, по сути, работает в тестовом режиме, поэтому её данные невелики. Более 30000 узлов сети, более 80000 каналов. Суммарный запас на начало 2022 г. внутри каналов составляет более 3200 BTC (более 200 000 000$).

С одной стороны, мы видим, как транзакции происходят мгновенно, с минимальными комиссиями (или вовсе без комиссии). В отличие от предыдущего примера, время жизни канала может быть вечным.

Но, с другой стороны, мы видим новые проблемы: каждое создание платёжного канала сопряжено с тратами (оплата блокчейну), значит, или нужно каждый раз при исчерпании средств в определённое время создавать новый канал (получил зарплату, создал канал, потратил, потом создал новый канал с новой зарплаты), либо полностью переходить в блокчейн, т. е. выгоднее искать работу, где будут платить именно биткоинами.

Но главное, биткоины всегда должны будут работать. То есть возникнет необходимость в наличии биткоинов. Чем больше у вас биткоинов, тем больший канал вы сможете создать с большим количеством людей. А значит, с вами будет сотрудничать выгоднее всего. Это может привести к резкому переходу на биткоины даже в тех отраслях, где они не требуется, а также к резкому росту стоимости биткоинов.

Возможно, именно этого и ждут те немногие, у кого сейчас на кошельках лежит огромное количество биткоинов и которые, к сожалению, смогут стать олигархами, которые по могуществу смогут превзойти нынешних IT-олигархов, затыкающих даже президентов США. Ведь если существует только 10 популярных каналов для передачи денег, которые аккумулируют в себе практически все средства, то больше всего вы будете бояться бана с их стороны, что повлечёт невозможность покупать что-либо и пользоваться услугами других людей.

Надеемся, общество сможет этому противостоять. Если потребуется — даже создать новый биткоин с заранее распределёнными между людьми деньгами. В любом случае, авторы надеются, что эта книга позволит резко увеличить количество людей, серьёзно разбирающихся в криптовалюте, чтобы в дальнейшем приложить усилия по противодействию любым, самым неожиданным угрозам. Ведь при наличии знания любые угрозы можно будет обойти.

СМИО
ПРЕСС

Приложение 1

Биткоин и его особенности

XIX век. После пятичасовых попыток объяснить крестьянину суть работы паровоза:

– Ну что, всё понял?

– Всё, полностью всё понял. Очень хорошо объяснили. Одно непонятно, куда тут лошадь впрягать?

Главная сложность при обучении математике состоит в том, что ученик не хочет копать вглубь. Чаще всего, математическое обучение заканчивается тем, что ученик выучивает несколько «магических» приёмов и с их помощью решает задачи.

Это большая проблема, ведь при любом «шаге в сторону» обучающийся совершенно не способен решить задачу. Всё потому, что он не понял суть явления, не разобрался до конца. В 239 школе мы всегда пытаемся заставить ученика или, точнее, помочь ему действительно разобраться в предмете. Вы должны помнить, что никакой «магии» не существует, а все математические приёмы, если понять их суть, достаточно просты.

Данная книга разбирает очень сложную тему — строение биткоина и других криптовалют. И главная сложность состоит в том, что ученик ожидает увидеть большую магию. Столкнувшись с тем, что здесь всё достаточно просто, чаще всего возникает эффект «отрицания», ученик считает, что ему что-то недоговаривают.

Данное приложение призвано показать, что в книге полностью разобраны и блокчейн, и система транзакций, и правило включения блока, и остальные составляющие биткоина. Мы предлагаем пройти с нами по шагам разбор блока и транзакций.

Также мы предлагаем каждому читателю попробовать подробно разобрать любой другой блок биткоина. Главное, чтобы было ощущение, что больше никакой магии нет. Что всё просто и понятно. Пожалуйста, если вы всё ещё считаете, что мы не разобрали какой-то пункт или элемент, пишите нам на почту crypto@239.ru.

Шаг первый. Блокчейн. Нужно понять, что в блокчейне биткоина нет ничего, кроме блоков. То есть всё, что нужно разобрать — это состав блока. Блоки идут друг за другом просто как предложения в этом тексте. Нет никакого

СМИО ПРЕСС

специального «соединения». Нет ничего, кроме последовательности из нулей и единиц, которые идут друг за другом.

Каждый узел проверяет блок на правильность данных и затем дозаписывает данные из блока в конец файла. Больше ничего. Нет никаких связующих звеньев либо чего-то ещё.

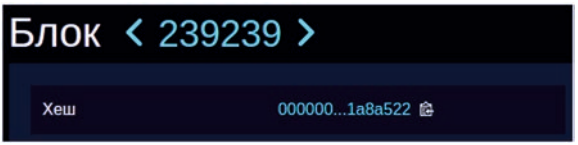
Шаг второй. Выбор номера блока для изучения. Для простоты мы предлагаем читателю выбрать любой номер от 1 до 742000 (блок, добытый во время написания книги) и изучить именно его. Мы, разумеется, выбрали блок 239239.

Шаг третий. Поиск блока. Любой полный узел содержит все блоки, однако мы понимаем, что для читателя гораздо удобнее получить блок онлайн. Сначала нужно узнать его хэш (т. к. потом мы будем искать именно по хэшу). Можем выбрать любой сайт, который используется для изучения блокчейна. Пусть это будет mempool.space. Далее мы вводим номер блока (рис. 139).

Рис. 139

И после поиска, получаем его хэш (рис. 140):

Рис. 140



Копируем данный хэш и переходим на сайт api.blockchair.com — это онлайн-сервис для подробного изучения блокчейна (вы можете выбрать любой другой). Далее вводим этот хэш, чтобы найти нужный нам блок:

https://api.blockchair.com/bitcoin/raw/block/000000000000000138baab2f35bf2d6a1b75e772597481919a149448d9f1a8a522

Мы видим первоисточник — буквально компьютерный код (рис. 141):

Рис. 141

"Raw_block":"02000000 4a0239275f80583f8c8dcebc8b7b28a5df1f4c8c8c 60f23f0a000000000000000 793b4e9f6089ca4f54e765f88c9f3e471965340a0e5 a0385f7274d27d87d1fce 58ffaa51 6461011afd510101 00000001..... (тут сотни килобайт текста).

Это 16-ричный код[1].

[1] *Напомним читателю, что в компьютере данные делятся на байты. Байт – это 8 битов. Бит – это 0 или 1. То есть байт – это 8 элементов (элемент 0 или 1). Так как стандартной (наиболее удобной для человека) является 16-ричная система счисления, каждый элемент занимает 4 бита. То есть в одном байте – два 16-ричных числа.*

СМИО ПРЕСС

Как мы уже разбирали, блок состоит из транзакций и заголовка блока. На рисунке мы специально выделили заголовок. Начнём именно с него. В принципе, мы повторяем главу 18, только в данный момент мы просим вас пройти весь путь вместе с нами на конкретном блоке. Мы надеемся, что вы выбрали другой блок, чтобы эксперимент был полным.

Далее, вы вместе с нами выписываете заголовок блока и делите его на байты:

02 00 00 00	Версия (не номер блока, именно версия)	4 байта
4a 02 39 27 5f 80 58 3f 8c 8d ce bc 8b 7b 28 a5 df 1f 4c 8c 8c 60 f2 3f 0a 00 00 00 00 00 00 00 Мы уже упоминали, что более привычным является «перевёрнутый» формат. То есть **00 00 00 00 00 00 00 0a 3f f2 60 8c 8c 4c 1f df a5 28 7b 8b bc ce 8d 8c 3f 58 80 5f 27 39 02 4a** Именно так, в перевёрнутом виде, нужно вводить в поиск: 	Хэш прошлого блока	32 байта
793b4e9f6089ca4f54e765f88c9f3e471965340a0e5a0385f7274d27d87d1fce	Корень дерева Меркла	32 байта
58ffaa51 — здесь тоже надо выделить байты **58 ff aa 51** и перевернуть в **51 aa ff 58**. Время можно проверить на любом сайте, к примеру: https://www.epochconverter.com/hex	Метка времени создания блока: 2013 год. 2 июня 12:16	4 байта
6461011a, — т. е. 1a016164 Мы помним, что это просто число в 16-ричном виде. Вы можете превратить его в 10-ричное число (например, в инженерном калькуляторе) — получится 436298084	Целевой уровень сложности. Здесь достаточно сложное понятие — мантисса числа должна быть меньше целевого уровня. Можно просто запомнить, что это целевая сложность (чем меньше, тем сложнее)	4 байта
fd510101	Nonce — то, что должен подобрать майнер	4 байта

В главе 18 мы подробно описали каждый из этих элементов. Здесь мы просто разбираем с вами конкретный пример блока.

СМИО ПРЕСС

Мы с вами разобрали выделенный элемент текста:

Как мы видим, заголовок блока занимает очень мало места. А далее идут транзакции, которые майнер включил в блок.

Теперь мы можем приступить к разбору конкретной транзакции. Разберём только одну — первую транзакцию, разбор остальных примерно такой же.

Выделим первую транзакцию в нашем списке (рис. 142):

Рис. 142

Как видите, абсолютно никакой магии. Закончился заголовок — начались транзакции.

Они просто идут друг за другом и, по сути, никак не связаны. Отдельно отметим, что транзакции в блоке никак не связаны не только технически, но и идейно. Майнер может набрать абсолютно любой набор транзакций, но обычно выбирает по принципу «кто больше заплатит». Мы говорили об этом в главе 19.

Как и в блоке, данные в транзакции просто идут одно за другим. Мы не будем выделять их цветом, просто опишем подряд, какие элементы в ней идут. Также не будем напоминать о визуальном «перевороте» по байтам.

Есть множество ресурсов, которые делят блокчейн на транзакции для удобного чтения. Например, мы упоминали о live.blockcypher.com/btc/decodetx.

Данная транзакция в удобном для человека виде будет выглядеть так (жирным — наш комментарий):

«fees»: 0, — **данного поля нет в транзакции, просто мы показываем его для удобства. Это первая транзакция, т. е. coinbase-транзакция, ради которой майнер совершал работу. Поэтому, комиссия — 0. Но вообще, блок был добавлен в 2013 г., тогда блоки даже до конца не заполнялись, так мало было переводов. И комиссию майнеры не брали — мемпула не хватало даже до полного заполнения блока.**

«hash»: «cf250d07097e8366e184179b5bc0c23501767dbd4402da45216a37dad6b0e277» — **как мы помним, хэш-идентификатор транзакции.**

«inputs»: — **список входов. Как вы помните, входов может быть несколько. Но здесь он только один.**

СМИО ПРЕСС

«script»: — **здесь должен быть зашит «ключ» к прошлой транзакции. Так как это coinbase-транзакция, здесь нет предыдущей, здесь информация от майнера.**

«0386a60304000157c0124d696e6564206279204254432047756c640800164e1400000002»,

```
    ],
      «outputs»: [
        {
          «addresses»: [
            «14cZMQk89mRYQkDEj8Rn25AnGoBi5H6uer» — этого элемента нет в шифровке, здесь показан только для удобства.
          ],
          «script»: «76a91427a1f12771de5cc3b73941664b2537c15316be4388ac»,
```

Именно тут зашит «замок» в нашей терминологии, к которому должен быть «приставлен» ключ впоследствии. Мы разобрали его ниже.

```
          «script_type»: «pay-to-pubkey-hash»,
          «value»: 2547355916
        }
      ],
      «preference»: «low»,
      «received»: «2022-06-26T01:11:39.710351513Z»,
      «relayed_by»: «3.221.158.139»,
      «size»: 122,
      «total»: 2547355916,
      «ver»: 1,
      «vin_sz»: 1,
      «vout_sz»: 1,
      «vsize»: 122 — технические поля. Размер комиссии, даты, размеры, версии и т. д.
    }
```

Мы теперь можем подробнее разобрать элемент «script». Даже сейчас он мало понятен. Код 76a91427a1f12771de5cc3b73941664b2537c15316be4388ac, согласно спецификации биткоина, разбивается следующим образом:

76a914
27a1f12771de5cc3b73941664b2537c15316be43
88ac
Расшифровывается следующим образом:
OP_DUP
OP_HASH160
OP_PUSHBYTES_20 27a1f12771de5cc3b73941664b2537c15316be43
OP_EQUALVERIFY
OP_CHECKSIG

Мы очень подробно разбирали схему стека. Вот в этой транзакции «замок» или запирающий стек именно такой.

Если мы поинтересуемся, была ли данная транзакция «потрачена» или, точнее, было ли такое, чтобы кто-то «присоединил» код к её выходу, мы можем это посмотреть в системе (у себя, если у вас есть узел, или на одном из сайтов) — и да, такая транзакция случилась на следующий день (рис. 143).

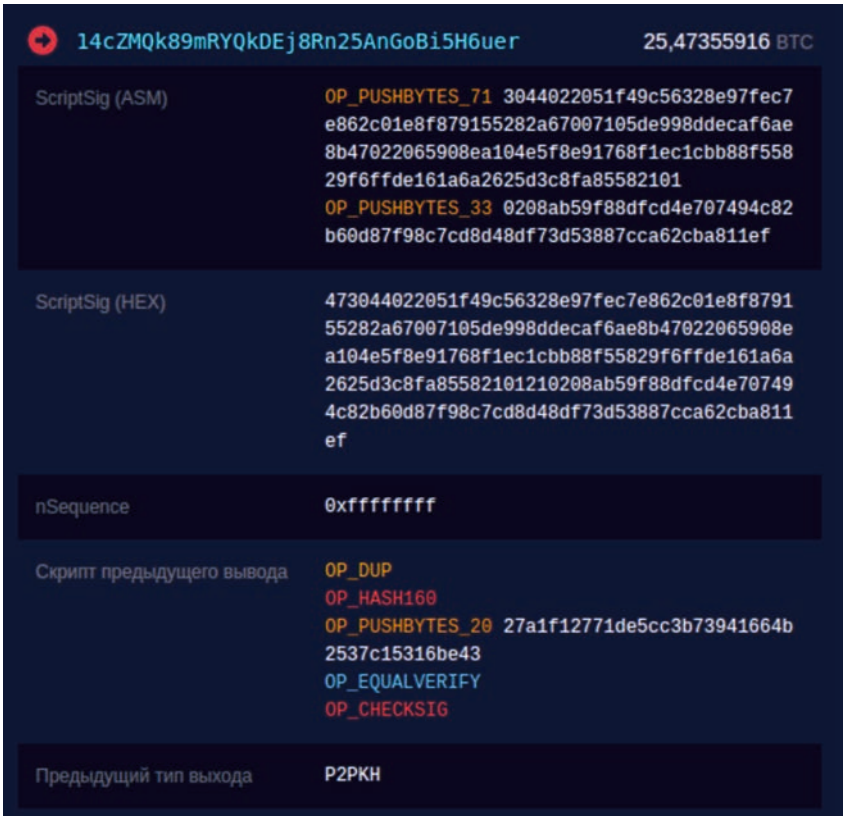


Рис. 143

И как мы видим, она была погашена, как того и требовалось, двумя элементами. Публичным ключом:

0208ab59f88dfcd4e707494c82b60d87f98c7cd8d48df73d53887cca62cba811ef

И подписью:

3044022051f49c56328e97fec7e862c01e8f879155282a67007105de998ddecaf6ae8b47022065908ea104e5f8e91768f1ec1cbb88f55829f6ffde161a6a2625d3c8fa85582101

Это погашение шло уже в блоке 239377.

Пожалуйста, изучите любой другой блок. Это докажет вам, что никакой магии нет, а вы полностью поняли всю идеологию и систему блокчейна биткоина.

СМИО ПРЕСС

Приложение 2

Различные методы консенсуса

Криптовалюты стали модной темой, и множество людей пытается привнести в них что-то своё. Придумано огромное количество концепций, но, к сожалению, им очень далеко по надёжности до основы биткоина — Proof of Work. Другие консенсусы включения в блок обычно не дают возможности полностью оградить цену от манипулирования. Кроме того, многие системы фактически централизованы, либо могут иметь тенденцию к централизации. В этом Приложении мы описали несколько видов консенсусов.

Proof of Work (PoW)

Доказательство работы. Метод, при котором ноды соглашаются на новый блок, если в нём решена криптографическая задача. В случае биткоина — вычисление такого хэша, который меньше определённого числа (или, что почти то же самое, — у которого в самом начале стоит определённое количество нулей).

Proof of Burn

Доказательство сжигания — майнер (если в данном случае его можно так назвать) отправляет валюту на заранее известный адрес, откуда он не сможет её достать. Этот протокол используют Binance Coin (BNB), Slimcoin (SLM), Counterparty (XCP) и Factom (FCT).

Proof of History

Цепочка хэшей связана. Мы можем это определить как хэшчейн. Мы знаем, какой блок был вычислен раньше, какой позже. Это похоже на связанные песчинки в песочных часах. Точно известно, какой блок был раньше, а какой позже.

Это просто метод замены timestamp на внутренние хэшчейн часы, поэтому, строго говоря, вообще не Proof. Должен работать вместе с Proof of Stake.

1. Есть список валидаторов, т. е. реально работающих нод. Каждый из них вычисляет верифицируемую функцию задержки (VDS).
2. Один из них становится лидером на период до 1000 блоков.
3. Вычисляет PoH и отправляет её валидаторам, которые также считали VDS и могут подтвердить, что прошло время, проверяют её и отправляют её каждый ещё двоим валидаторам. Валидаторы получают вознаграждение.
4. Через 1000 блоков выбирается другой лидер.

Пример: Solana.

Proof of Importance

Иногда ноды не только принимают, но и обслуживают платежи, — к примеру, создают микс переводов для более сложного отслеживания. Данная система реализована в валюте NEM и позволяет производить выплаты за то, что нода является активным пользователем сети.

Proof of Stake (PoS)

Доказательство владения. Метод, при котором ноды соглашаются на новый блок, если его добавляет пользователь, у которого максимальное денежное вознаграждение на счёте. Есть отдельный вид — DPOS — который реализован, к примеру, в криптовалюте EOS. Разница состоит в том, что ноды могут, объединившись, выбрать того, кто подтверждает транзакции в сети.

Proof of Authority (PoA)

Очень похожая на прошлую систему. Только здесь майнить даётся только «избранным», т. е. не каждый узел, на котором есть средства, может стать майнером, а только утверждённые неким центром.

Proofs of Replication, Proof of Storage, Proof of retrievability

Пример: FileCoin.

Очень интересная идея, когда майнером становится тот, кто дал в пользование часть объёма собственного жёсткого диска. Это позволит зарабатывать на ненужной его части. Но остаётся открытым вопрос, можно ли данную систему считать криптовалютой.

Proof of Capacity

Похожая на прошлую идею, майнинг тоже основан на наличии жёсткого диска, но он не отдаётся в аренду на полезное хранение, а забивается всеми возможными nonce, чтобы при включении в блок отобрать тот, что подходит.

СМИО
ПРЕСС

Приложение 3

Альткоины и их различия

Согласно сайту https://coinmarketcap.com/

Сейчас на рынке существует более 14 тысяч криптовалют. Разумеется, большинство из них является шиткоинами, но даже среди топ-валют сложно найти действительно важные инструменты, а многие, на наш взгляд, вообще нельзя назвать криптовалютами.

Нужно помнить, что биткоин не зря до сих пор доминирует на рынке. Новые криптовалюты не смогли предложить ничего содержательно нового и отличаются лишь небольшими нововведениями. Мы взяли 25 валют с самым большим объёмом торгов по оценке coinmarketcap на 2022 г. и буквально в одну строчку описали их отличия от биткоина.

Dogecoin (догикоин) https://dogecoin.com/ — мем-криптовалюта, которая обладает собственным блокчейном. В 2021 г. приобрела второе рождение благодаря поддержке Илона Маска. На данный момент с технической точки зрения является копией лайткоина, каким он был примерно в 2015 г. Из-за отсутствия халвинга общее количество монет не ограничено, время создания блока составляет одну минуту. Создатель блока получает вознаграждение в 10 тыс. DOGE.

Ethereum (эфир) https://ethereum.org/en/ — самый известный альткоин. Главным образом отличается от биткоина языком, на котором можно написать «замок» UTXO. Надеюсь, читатель помнит, что все транзакции в криптовалютах представляют цепочку, когда для совершения транзакции необходимо добавить в конец цепочки раскрывающий скрипт.

Для биткоина мы подробно описали скриптовый стековый язык, который позволяет создавать только очень простые замки. Язык эфира не ограничен ничем и является тьюринг-полным, т. е. позволяет писать любые программы. На эфире возможно создать любые правила, а также создавать валюты, которые будут иметь собственные UTXO-цепочки, но перевод будет осуществляться через блокчейн эфира. Самым известным токеном[1] ERC-20 является Tether ERC-20

[1] *Единица учёта, которая для перевода использует блокчейн криптовалюты.*

(есть и другие tether). Халвинг у эфира отсутствует, поэтому их количество неограниченно. В качестве хэша используется более современная версия, чем у биткоина, — SHA3 — Keccak. Это не влияет на надёжность, но не позволяет АСИКам полностью захватить майнинг.

Binance USD и Binance Coin https://www.binance.com/en. Самая известная и крупная криптобиржа Binance запустила собственный блокчейн «BEP-2». Создавался как токен для ICO Binance, на котором собрали около $15 миллионов, в 2019 году после создания своего блокчейна были обменены 1:1 на токены BEP-2. Их нельзя считать отдельной криптовалютой, т. к. вся система основана на доверии к бирже. Каждый месяц биржа скупает токены, расходуя 20% от своей прибыли, и сжигает их.

Также токенами можно оплачивать комиссию самой Binance по обмену, а также некоторые другие услуги. Их список есть на сайте.

XRP (бывший Ripple) https://xrpl.org/. Данную «валюту» часто включают в список криптовалют, однако это очень спорное решение. Можно сказать, что это последняя «добиткоиновская» попытка создать распределённые деньги. Первый вариант был представлен ещё в 2004 году. Сейчас система тестируется в нескольких финансовых компаниях финансового сектора.

Ethereum Classic — эфир, как он изначально задумывался. Как было сказано, эфир отличается от биткоина тем, что язык, которым пишется «замок» UTXO, не ограничен и можно писать любые программы (умные контракты), однако буквально первая же серьёзная программа была взломана хакерами, которые похитили большую сумму (сложно оценить сейчас эту сумму, тогда сумма потраченных эфиров была около 50 миллионов долларов). После этого создатель эфира прямым указанием решил «отрезать» цепочку блоков и начать «с чистого листа», будто транзакций не было.

Это привело к серьёзным противодействиям многих майнеров и держателей эфира. По их мнению, если несколько человек в «руководстве» эфира могут отменять транзакции, это нельзя считать настоящим блокчейном. Этот конфликт и привёл к разделению эфира на две цепочки.

Litecoin (лайткоин) https://litecoin.org/. Возможно, самая успешная на данный момент «копия» биткоина. Часто называемая цифровым серебром (в противовес «цифровому золоту» — биткоину). Создана именно как облегчённая версия. Блок собирается каждые 2,5 минуты, используется система противодействия ASIC, что до сих пор позволяет майнить на видеокартах, гораздо дешевле перевод средств. В 2017 году лайткоин был в авангарде прогресса (к примеру, в нём первым был имплементирован SegWit), однако позже практически лишился главного разработчика и не смог конкурировать дальше с биткоином. Тем не менее, если вам не нравится идея смартконтрактов, эта валюта стоит на втором месте после биткоина.

СМИО ПРЕСС

Chainlink. Токен эфира (не имеет собственного блокчейна). Не имеет большого веса в криптосреде и фактически создан только для оплаты услуг компании, которая предлагает делать «оракулов» — программы, которые находясь внутри смарт-контракта, могут получать различные данные из Интернета, а они могут влиять на работу самого смарт-контракта.

Bitcoin Cash. Криптовалюта с собственным блокчейном. В какой-то момент оказалось, что разные люди по-разному смотрят на то, каким биткоин должен стать. В 2017 г. произошёл очередной конфликт. Часть разработчиков считала, что объём блока должен серьёзно вырасти, другая — что надо создать способ сокращения веса транзакции. Bitcoin Cash — классический хардфорк (разделение валюты на две или более).

Каждый, у кого был биткоин, получил один биткоин кэш. У классического биткоина было имплементировано сокращение размера транзакции (SegWit), а Bitcoin Cash сделал размер блока в 8 мегабайт (у биткоина было 1 мб с возможностью расширения до 2 мб). Далее хардфорки происходили не один раз. Создавался Bitcoin SV, Bitcoin Gold, Diamond и т. д. Но если разделение на Bitcoin и Bitcoin Cash ещё можно было считать следствием идеологического спора, то другие хардфорки были просто попыткой заработать на шумихе.

EOS. Криптовалюта с собственным блокчейном, которая делала попытку стать «вторым эфиром». Идеология совершенно такая же, но по популярности уступает даже Ethereum Classic.

Polkadot https://polkadot.network/. Ситуация с данной «криптовалютой» похожа на Chainlink — это попытка расширения возможностей смарт-контрактов. В данном случае — попытка объединить разные блокчейны «первого поколения». По сути, это токен команды, которая пытается решить задачу и пока её не решила.

Cardano https://cardano.org/ — криптовалюта с собственным блокчейном. Первая в нашем списке, которая работает на Proof of Stake (на момент написания книги эфир ещё не перешёл на него). Читатель сам вправе решать, считать ли такой способ «настоящим» блокчейном, но на данный момент Cardano очень популярна и имеет серьёзный вес в криптосреде.

Dash — «анонимная» криптовалюта. Во многом показывает, что будет с крипторынком при попытках его запретить. Если в биткоине вы можете легко отследить все транзакции, следуя по их выводам, то в Dash применено огромное количество способов скрыть получателя и отправителя. Также интересна тем, что для майнинга требуется считать 11 видов хэшей, что усложняет работу АСИКам.

TRON https://tron.network/ — сложно описать «криптовалюту» в двух словах, но она является ярким примером того, когда криптовалютой называет что

СМИО ПРЕСС

угодно. Теоретически создана для ICO, но система не предусматривает даже майнинга.

Uniswap https://uniswap.org/ — фактически «токен» для так называемой децентрализованной криптовалютной биржи. Нужен для работы с этой биржей и выплаты комиссий. На его примере тоже можно оценить, насколько разные инструменты называются «криптовалютами».

Neo — очередная попытка создать «второй эфириум», который был ориентирован на ICO токенов. Но в связи с окончанием бума ICO ушёл на второй план.

Filecoin https://filecoin.io/. Один из действительно необычных проектов, который пытается дать возможность монетизировать свободное пространство на вашем жёстком диске. Использует консенсусы Proof-of-Spacetime (PoST) и Proof-of-Replication (PoRep). Проект пока на начальной стадии разработки и полностью зависит от команды, которая его предложила.

Zcash https://z.cash/ — возможно самая «анонимная» криптовалюта, но из-за отсутствия давления на биткоин и другие валюты сейчас не очень популярна. Отследить транзакции в ней невозможно, поэтому в случае давления на валюты типа биткоина может выйти на передний план.

Solana https://solana.com/ — одна из самых популярных платформ, основанная на системе консенсуса Proof of Stake (PoS). При этом создатели пошли дальше: чтобы создать блок, вы должны не только обладать деньгами, но и иметь определённый статус. Доверие только избранным делает её достаточно централизованной. Это и недостаток (с точки зрения идеалов криптомира), и преимущество (можно обрабатывать огромное число транзакций).

СМИО
ПРЕСС

Приложение 4

Различные идеи и термины в современном криптомире

Альткоин

Уже достаточно устаревший термин. По сути, он означает «любая криптовалюта, кроме биткоина».

AMM

Automated market maker. Автоматическая биржа.

ASIC

Специальное устройство, которое позволяет с меньшей затратой энергии рассчитывать специфические криптографические задачи. К примеру, чтобы рассчитывать биткоин-хэши SHA256 нет необходимости использовать сложное устройство, такое, как CPU или GPU, достаточно гораздо более простых устройств, которые, правда, ничего кроме расчёта SHA256 делать не умеют.

ASIC используют для майнинга различных криптовалют. При этом практически все криптовалюты стараются бороться с майнингом на ASICах, чтобы избежать излишней централизации майнинга.

Атака 51%

Атака на криптовалюту, при которой один майнер имеет большее количество голосов, чем все остальные, и в связи с этим может отменять транзакции в системе. В разных системах голосом могут считаться разные системы — в алгоритме биткоина и других систем Proof of Work (PoW) — это уровень хэшрейта.

То есть если один майнер обладает большим хэшрейтом, чем остальные, его цепочка блоков может быть главенствующей.

Биткоин-краны

Самый лёгкий способ получить первую криптовалюту. Вы совершаете определённое действие (часто просмотр рекламы), и вам выплачивают вознаграждение в биткоинах. Сегодня это считанные сатоши, но в 2010 году краны могли раздавать несколько биткоинов.

В момент создания краны использовались для привлечения новых участников системы, как первый шаг, чтобы любой человек мог получить первую криптовалюту.

СМИО ПРЕСС

DEFI

Decentralized Finance — просто собирательное название любых децентрализованных финансовых решений и инструментов.

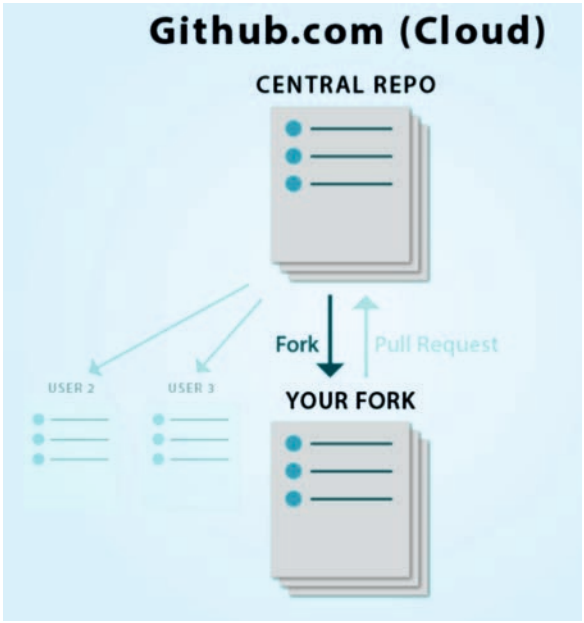
ERC-20

Экосистема второй по популярности валюты, Ethereum позволяет вносить предложения по изменению экосистемы. Токен называется Ethereum Request for Comments. Соответственно, ERC-20 — это, по сути, двадцатое предложение по улучшению, которое было принято и в котором ввели первый стандарт смарт-контракта. Смарт-контракты позволяют фактически собственную валюту — токен, который будет использовать блокчейн эфира. Токен должен соответствовать этой спецификации и фактически является токеном стандарта ERC-20 или смарт-контрактом на блокчейне Ethereum.

Токены (смарт-контракты) можно было создавать и до этого, но единый стандарт позволил делать это гораздо лучше и проще. Также нужно отметить, что смарт-контракты возможны не только в эфире, но и на других криптовалютах, к примеру в биткоине.

Fork

Или «форк» (hard fork, soft fork) — это любое ответвление в программном продукте. К примеру, на гитхабе оно выглядит следующим образом:



То есть вы можете скачать чужую программу (код), изменить его, и это будет ваш форк. Если вы обладаете определёнными правами, вы можете закачать его обратно после изменения. И это станет основной веткой.

СМИО ПРЕСС

А кто-то может изменить по-другому, «запуллить» на свой аккаунт, и это станет отдельной программой. В текущих терминах криптовалюты, если заливаете после изменения в основную ветку, то это софтфорк, а если создаёте отдельную программу, — это хардфорк.

При софтфорке пользователь, если хочет работать в текущей программе, может легко сохранять и обмениваться файлами с теми, кто использует более новую программу. При хардфорке чаще всего это невозможно. Если вы в старой программе можете работать с файлами, которые созданы в новой программе, — это совместимость. Если вы в новой программе можете работать с файлами, созданными в старой программе, — это обратная совместимость.

В ситуации с криптовалютой, при софтфорке биткоина вы можете пользоваться старыми кошельками, не переходя на новые. Даже если изменения глобальны — как ввод Segregated Witness.

Хардфорки, как уже было сказано, это фактически отдельные программы, и если сообщество принимает изменения, все просто переходят на новую версию. К примеру, так произошло при хардфорке Monero, поддерживающем Ring Confidential Transactions.

Иногда часть сообщества отвергает хардфорк, например, когда часть эфира была украдена у создателей Дао, и Бутерин откатил транзакции с помощью хардфорка, часть сообщества отказалась это сделать из идеологических соображений и создала Эфир Классик.

А иногда хардфорк в принципе делается для разделения, как было, например, при создании Bitcoin Cash: создатели изначально планировали разделение, при этом с целью продвижения валюты было введено правило, что сам блокчейн наследуется. И все, кто владел биткоинами на момент разделения, стали владеть также биткоин-кэшами в той же пропорции.

При хардфорках, понятно, вам необходимо пользоваться различными кошельками.

Harvesting и стекинг

В разных системах, например, в Proof-Of-Importance и Proof of Stake (PoS), майнеры не получают или получают не все выплаты. Часть выплат идёт нодам, которые не занимаются активной добычей (майнингом), и процесс получения выплат называется Harvesting. Часто Харвестинг также заменяют термином стекинг.

Халвинг

Уменьшение в два раза размера выплаты майнерам за добавление нового блока в блокчейн. Изначально халвинг был заложен Сатоши Накамото программно в сеть биткоина через каждые 210 тысяч блоков, то есть раз в 4 года. Благодаря халвингу количество биткоинов ограничено примерно 21 млн монет (предположительно в 2140 году вознаграждение за блок станет меньше 1 сатоши).

СМИО ПРЕСС

ICO

Initial Coin Offering (первичное размещение токенов) и фактически противопоставляется термину IPO (Initial Public Offering — первичное публичное предложение акций), которое используется на обычных биржах.

ICO — это выпуск токенов, которые иногда также называют купонами. Токены могут оцениваться как угодно. Иногда они твёрдо привязываются к какой-нибудь оффлайн ценности — например, к доллару (тезер) или золоту. Могут быть определены как акция или любым другим способом. Токены обычно привязаны к какой-нибудь криптовалюте и используют её блокчейн для передачи.

IEO

Initial exchange offering (первичное биржевое предложение) — по сути, вариант ICO, проходящего под контролем определённой криптобиржи. То есть, если при ICO создатели сначала создавали токен, а потом пытались (иногда безуспешно, иногда за отдельную, большую плату) провести ICO, то при IEO токен размещается сразу на бирже и его просто проще купить (достаточно зарегистрироваться на бирже и оплатить).

IPFS

InterPlanetary File System, («межпланетная файловая система») — это гипермедийный протокол связи с открытым кодом, с помощью которого одноранговые узлы осуществляют хранение и распространение данных в единой распределённой файловой системе.

Lightning Network

Сандчейн, или «второй уровень» биткоина, который позволяет ограниченному кругу лиц создать сеть для пересылки денег друг другу минуя блокчейн, в который записывается лишь первая транзакция. Её суть в том, что мы создаём «замок» на краю UTXO, который не позволяет получателю получить все деньги без дополнительного кода. Мы как бы можем сделать один перевод на большую сумму. Но передавать коды частями.

Нода (Узел, node)

Нода, или узел, — персональный компьютер, на котором есть полный блокчейн и который может участвовать в проверке (но не создании) новых блоков. Нода принимает новый блок, если он соответствует определённым правилам, т. е. доказана работа или владение, или любой другой вид доказательств.

Иногда нода также участвует в жизни сети (в некоторых валютах — к примеру Dash, NEM и других). В таких системах ноды (их называют мастернодами) получают вознаграждение вместе с майнерами.

NFT

Как известно, биткоин вводит блокчейн для того, чтобы не было возможности произвести двойную трату. То есть, если у вас есть некий программный

СМИО ПРЕСС

продукт, вы могли передать его только один раз. Эта идея и привела к созданию NFT (Non-Fungible Token) — цифровые предметы, обладающие уникальностью. Их нельзя копировать. Всё это достаточно условно, но тем не менее работает. Логика здесь в том, что вы, конечно, можете скопировать картину Рафаэля — но всё-таки это не будет картина Рафаэля.

Для покупки NFT вам необходим кошелёк Метамаск.

Oracle Оракул

Программа, которая даёт смарт-контракту информацию, которой нет в блокчейне.

OTC-трейдинг

Трейдинг напрямую, не через биржи — т. е. вы просто договариваетесь с кем-то и продаёте/покупаете.

Pump & Dump

Манипулятивный способ повышения цены на заранее купленные токены, криптовалюты или любой другой товар. Известен с давних времён. По сути, является мошенничеством.

1. Маленькими партиями скупаются токены, криптовалюта или акции, на данный момент рост котировок нежелателен.
2. Различными способами идёт реклама — через телеграм-каналы, сайты, а раньше через форумы.

При повышении заинтересованности организаторы продают ранее купленные товары по более высокой цене.

Satochi Nakamoto (Сатоши Накамото)

Ник основателя или группы основателей биткоина, первого программного клиента. Настоящее имя неизвестно

Segregated Witness (SegWit2x)

Увеличение количества подтверждённых транзакций в каждом блоке. В 2017 г. впервые был введён в качестве софтфорка биткоина, но основная масса нод перешла на него лишь в 2018 г. Также в этот софтфорк вводилось увеличение блока с 1 до 2 МБ.

https://forklog.com/chto-takoe-segregated-witness/
https://cryptor.net/tech/segregated-witness
https://cryptor.net/bitkoin-dlya-chaynikov/chto-takoe-segwit-segregated-witness

Смарт-контракт

Смарт-контракт — это компьютерная программа, которая создаётся при запуске новой криптовалюты (токена) и не может быть изменена. Он контролируется каждой нодой, и поэтому злоумышленники не смогут забрать средства, которые в нём хранятся (и им используются), — в таком случае транзакции просто будут отменены.

В качестве примера: мы можем использовать умный контракт в качестве краудфандинговой платформы, вбив в него код: если сумма на кошельке

достигнет в течение 2000 блоков 10 000$, то отправить эти деньги на определённый кошелёк. А если не достигнет, отправить деньги обратно отправителям.

Любой блокчейн может использовать смарт-контракты, в том числе и биткоин. Разница в смарт-контрактах биткоина и эфира состоит в том, что программирование на эфире является тьюринг-полным. То есть на нём можно писать абсолютно любые программы — а в биткоине заранее запрещены многие опасные, по мнению разработчиков, функции. Смарт-контракты на эфире гораздо более функциональны, однако в них гораздо чаще обнаруживаются уязвимости.

Стейблкоины

- равны фиатной валюте, чаще всего доллару (пример: Tether);
- которые можно обменять на цифровую валюту или корзину (BitUSD, токены Dai);
- необеспеченные (Seigniorage Shares).

Токен

Это единица учёта, которая для перевода использует чужой блокчейн.

СМИО
ПРЕСС

Приложение 5

Риски биткоина

С момента создания биткоина сразу же возникли прогнозы по очень скорой его смерти. Многие сайты даже подсчитывают биткоин-некрологи в различных изданиях. Например, по данным сайта https://99bitcoins.com/bitcoin-obituaries/ на конец 2021 г. вышло уже более 430 биткоин-некрологов.

Биткоину предсказывали смерть, когда он несколько месяцев не поднимался выше 0.23$, когда он упал с 5 до 3 долларов, когда упал с 1000 до 200 и так далее. Биткоин пережил десятки сайтов, которые говорили о его скорой гибели. Возможно, это случайность, и он может умереть в ближайшее время?

Опасность: криптовалюты станут не нужны.

Напомним, для чего нужен биткоин. Ведь, как мы показали, система биткоина гораздо более сложна и ресурсоёмка, чем традиционные финансы. Вместо того чтобы во время транзакции просто вычесть из одного баланса и прибавить к другому балансу, мы создаём огромную систему UTXO, каждая транзакция ожидает минимум несколько минут до включения в блок. Суммарный расход электроэнергии майнерами сравним с расходом небольшой страны. В общем, единый доверенный центр гораздо лучше справляется с финансовыми потоками.

Там, где есть единый финансовый центр, которому можно доверять, к примеру, внутри Российской Федерации, никто не пересылает деньги в биткоинах. Ведь любой человек легко откроет счёт в рублях, может перевести их другому по номеру карты или даже по номеру телефона. Есть ещё более выгодные способы перевода со счёта на счёт, например, электронные платежи.

Однако в мировом объёме такой системы не существует. Доллар, который так долго выполнял подобную функцию, практически убивает себя. Печать денег в огромных объёмах, запредельная стоимость перевода между счетами, очень сложная система открытия счёта (особенно для компаний), требование огромного числа документов и так далее.

Валюты других стран ещё хуже приспособлены к тому, чтобы служить мировым эквивалентом. Евро ничем не лучше доллара, переводы в юанях ещё более сложные, чем в долларах, и могут быть заблокированы в любой момент.

В общем, нужно прямо сказать, что в случае появления лёгкого способа переводить средства без вероятности блокировки, вся система биткоина умрёт

СМИО ПРЕСС

довольно быстро. Однако в возможность появления такой системы в современном раздираемом противоречиями мире верится с трудом. Поэтому можно говорить о долгосрочном существовании биткоина и других криптовалют.

Опасность: биткоин и другие криптовалюты запретят во всех странах.

Опыт говорит о том, что чем больше запретов, тем больше и противодействие. После запрета Napster появился bitTorrent, который умирает только из-за того, что появились очень дешёвые стриминговые сервисы и почти бесплатные онлайн-кинотеатры. Этот пример показывает, что запреты на любую популярную экосистему приводят только к её развитию. А биткоин может умереть, только если традиционные финансы будут лучше, чем он.

Опасность: появится криптовалюта лучше, чем биткоин.

Это наиболее вероятная смерть биткоина. Если появится валюта, которая будет явно лучше, она станет и популярнее. Однако пока никто не смог предложить ничего принципиально нового. В приложении 3 мы рассмотрели десятки криптосистем, но они по большей части лишь клоны. У кого-то другой метод хэширования, у кого-то другой принцип консенсуса, другой скриптовый язык и т. д.

Возможно, в будущем появится что-то, что будет несравненно лучше биткоина. К примеру, майнеры будут считать не бессмысленные хэши, а что-то полезное. И люди просто из гражданских чувств будут переходить в эту систему. Это и приведёт к смерти и биткоина, и всех современных криптовалют.

Опасность: биткоин взломают.

Биткоин основан на двух математических принципах: шифровании с помощью эллиптических кривых и хэшировании SHA256. Взлом одной из этих систем приведёт к тому, что система биткоина потерпит крах, если не успеет измениться.

Насколько это реально? Во-первых, сразу нужно уточнить, что на этих же основах стоит и классическая финансовая система. И не только при пересылке денег, но и при пересылке документов (электронные подписи). Биткоину, в отличие от современных централизованных систем, будет гораздо проще адаптироваться к изменениям. Один программист предложит решение, его примут почти мгновенно и просто перейдут на новую систему хэширования или шифрования.

Квантовый или другой супер-компьютер с огромной вычислительной мощностью.

Но возможен ли в принципе взлом этих систем? Возможно, квантовый компьютер сможет уничтожить биткоин? Надеюсь, читатель помнит, как происходит расчёт открытого ключа в эллиптических кривых. Мы берём число, меньшее 2^{256}, и такое количество раз складываем саму с собой базовую точку. Чтобы найти приватный ключ, зная его публичный ключ, нужно запустить сложения этой точки и просто складывать, пока не получится приватный ключ.

СМИО ПРЕСС

Сможет ли компьютер будущего это сделать? Нет, не сможет. Представим себе, что квантовый компьютер совершает сложение двух точек за самое маленькое возможное время. Современная физика называет это время планковским временем, ни один процесс не может происходить быстрее:

$$5{,}391247(60) \times 10^{-44} \text{ с.}$$

У нас количество операций 2^{256} или примерно (очень примерно) 10^{75}, т. е. если компьютер сможет одну операцию совершать за немыслимо короткое время, быстрее которого он не может из-за физических законов, он будет вынужден считать 10^{31} секунды, или около 10 000 лет.

Но это не самое страшное. В конце концов, вдруг мы создадим 100000 таких компьютеров и сможем за месяц взломать систему! Однако есть ограничение на количество энергии. Очень хорошо это рассмотрено в книге «Прикладная криптография» Брюса Шнайера[1]. Термодинамика накладывает минимальное ограничение на изменения кванта — по сути, это минимальная энергия, которую должен потреблять процесс. Эта энергия составляет при температуре, близкой к абсолютному нулю, $4{,}4 \times 10^{-16}$ эрга. Для 10^{75} операций это будет означать, что потребуется $4{,}4 \times 10^{59}$ эрга.

Солнце выделяет за год $1{,}21 \times 10^{41}$ эрга, если мы соберём за 1000 лет всю энергию, которую создало Солнце, то получим $1{,}21 \times 10^{44}$ эрга энергии. При взрыве сверхновой выделяется 10^{54} эрга, т. е. нам понадобится энергия взрыва 100000 сверхновых, которую мы соберём без остатка.

В общем, взлом перебором невозможен в принципе никаким квантовым компьютером.

Но если найдётся гениальный математик, который взломает шифр? Теоретически да, но маловероятно, что это возможно. Хотя возможности человека безграничны, скорее найдётся хакер, который найдёт ошибку в программе. Но гораздо более вероятно, что найдётся хакер, который найдёт ошибку в современных банковских системах.

[1] *Шнайер Брюс. Прикладная криптография. Издательство Вильямс, 2016.*

СМИО ПРЕСС

Дисклеймер

Авторы книги напоминают, что на территории Российской Федерации единственное законное платёжное средство — российский рубль. Авторы призывают не использовать никакие другие способы покупки товаров и услуг, а также не принимать в качестве оплаты криптовалюты (в частности, биткоин). Вы должны соблюдать законы страны, в которой проживаете.

Отдельно мы хотели бы предостеречь от вложения в криптовалюты. Особенно рискованны краткосрочные вложения и торговля «с плечом». Скорее всего, вы просто потеряете деньги.

Книга предназначена прежде всего для обучения и глубокого понимания сути блокчейна биткоина, как он работает. Криптовалюты для нас это прежде всего очень красивая математическая модель, однако решение о её применимости в реальной финансовой жизни страны может принять только высшее политическое руководство.

Доскональные знания распределённого реестра нужны в первую очередь людям, которые планируют работать программистами в самой высокооплачиваемой отрасли — IT и блокчейна.

Мы призываем вас не пользоваться биткоином и другими криптовалютами как платёжным средством, если это запрещено законодательством вашей страны.

Хотелось бы также объяснить, почему в книге мы чаще всего используем термин «биткоин», а не «криптовалюты» или «блокчейн». Проблема в том, что эти понятия в последнее время стали очень размыты. К криптовалютам можно причислить и Yota, и другие централизованные валюты, которые имеют в своём составе криптографию (под это описание подходит и система Visa), а термин «блокчейн» вообще можно интерпретировать как угодно — и как сложную систему с доказательством работы, и просто как цепочку блоков, которая может строиться по любому принципу.

Термин биткоин, в свою очередь, предельно конкретен и понимается одинаково в любой литературе. Однако объяснение на основе биткоина не умаляет общности. А в конце книги указаны отличия главных альткоинов от биткоина. Они очень невелики.

СМИО
ПРЕСС

Содержание

Условные обозначения в книге 4
Кому предназначена книга 5
Вступление и концепция книги 6
Краткое содержание книги 8

Часть 1
ФИНАНСОВАЯ И ЮРИДИЧЕСКАЯ СТОРОНА 11
Глава1
Идеология денег и биткоина 12
Глава 2
Идея отсутствия удостоверяющего центра 21
Глава 3
Отсутствие доверия между сторонами при передаче информации.
Задача византийских генералов 28
Глава 4
А при чём тут биткоин 33

Часть 2
МАТЕМАТИЧЕСКАЯ ОСНОВА 40
Глава 5
Вступление в техническую часть 41
Глава 6
Симметричное и асимметричное шифрование 43
Глава 7
Доказательство с нулевым разглашением 57
Глава 8
Хэширование 60
Глава 9
Электронно-цифровая подпись 74
Глава 10
Асимметричное шифрование на эллиптических кривых 86
Глава 11
Создание приватного и публичного ключей 94
Глава 12
Proof of Work 100

СМИО ПРЕСС

Часть 3
КОНЦЕПЦИЯ И KNOW-HOW БИТКОИНА 103
Глава 13
Создание биткоин-адресов 104
Глава 14
Общая теория передачи ценности в криптовалютах 114
Глава 15
Хэшчейн 132
Глава 16
Проблема двойных трат 136
Глава 17
Блокчейн 141
Глава 18
Подробное описание блока на основе блока биткоина 144
Глава 19
Консенсус на основе Proof of Work и необходимость майнинга 155
Глава 20
Дальнейшее развитие 166

Приложение 1
Биткоин и его особенности 177
Приложение 2
Различные методы консенсуса 183
Приложение 3
Альткоины и их различия 185
Приложение 4
Различные идеи и термины в современном криптомире 189
Приложение 5
Риски биткоина 195

Дисклеймер 198

СМИО
ПРЕСС

«Я давно мечтал, чтобы кто-то написал о биткоине и его функционировании русским языком (то есть, во-первых, по-русски, и во-вторых, доступным людям образом). Кажется, моя мечта сбылась: книга, которую вы держите в руках, мною в общих чертах освоена и понята. Значит, и любым матшкольником тоже может быть освоена и понята. И всяким сильным любителем математики тоже.

Упростить материал ещё дальше без потери содержательных моментов – невозможно. Авторы написали максимально простой текст на заданную тему. Я всегда говорил, что на свете в 2008 году жило лишь несколько человек, которые понимали программирование, теорию чисел и теорию игр на том уровне, который бы позволил спроектировать криптовалюту.

С появлением этой книги мы получим тысячи таковых в одной России! Возможно, кто-то из читателей кардинально упростит всю схему, дав миру новый шедевр системы электронного доверия. А пока, друзья, читайте эту книгу и умнейте!»

А. В. Савватеев,

член-корреспондент РАН,
руководитель Кавказского математического центра,
профессор МФТИ и Российской экономической школы

ISBN 978-5-7704-0396-1

СМИО
ПРЕСС