O'REILLY®

Ethereum — окно в мир всемирных децентрализованных вычислений. Эта платформа позволяет реализовывать децентрализованные приложения (DApps) и смарт-контракты без центральных точек доступа или контроля, интегрироваться с платежной сетью и работать с открытым блокчейном. В этом практическом пособии Андреас Антонопулос и Гэвин Вуд дают всю информацию, необходимую для построения смарт-контрактов и децентрализованных приложений в Ethereum и других блокчейн-системах.

Узнайте, почему IBM, Microsoft, NASDAQ и сотни других организаций экспериментируют с Ethereum. С помощью этого незаменимого руководства вы приобретете навыки, необходимые любому новатору в этой новой развивающейся и захватывающей индустрии.

- **Запуск клиента Ethereum, создание и передача простейших транзакций и написание смарт-контрактов**
- **Понимание процессов хранения цифровых ключей в «кошельках», которые управляют криптоактивами и смарт-контрактами**
- **Программное взаимодействие с клиентом Ethereum с помощью библиотек JavaScript и интерфейсов для удаленного вызова процедур**
- **Создание цифровых токенов, которые представляют собой активы, акции, голоса или права доступа**
- **Построение децентрализованных приложений с помощью нескольких пиринговых (P2P) компонентов**

Андреас Антонопулос заслужил признание как автор бестселлеров, докладчик, преподаватель и один из главных экспертов в мире по криптовалюте Bitcoin и открытым блокчейн-технологиям. Андреас делает сложные темы доступными и понятными.

Доктор **Гэвин Вуд** является соучредителем и бывшим техническим директором Ethereum (Ethereum Foundation), автором языка программирования для написания смарт-контрактов Solidity, основателем и президентом Web3 Foundation, а также основателем и ведущим разработчиком Parity Technologies.

«Я рекомендую эту книгу всем тем, кто хочет погрузиться в изучение криптографии, транзакций и безопасности, а также узнать об основах технологии Ethereum. Авторы раскрыли эти темы максимально широко. Книга также может стать надежным проводником для аналитиков, разработчиков, технических архитекторов и всех криптоэнтузиастов».

Глеб Костарев,
директор Binance по России и СНГ

БОМБОРА
ИЗДАТЕЛЬСТВО
БОМБОРА – лидер на рынке полезных и вдохновляющих книг.
Мы любим книги и создаем их, чтобы вы могли творить, открывать мир, пробовать новое, расти. Быть счастливыми. Быть на волне.
bomborabooks bombora.ru

ISBN 978-5-04-106781-6

ОСВАИВАЕМ Ethereum
Андреас Антонопулос
Гэвин Вуд
O'REILLY®
БОМБОРА

O'REILLY®

ОСВАИВАЕМ Ethereum

СОЗДАНИЕ СМАРТ-КОНТРАКТОВ И ДЕЦЕНТРАЛИЗОВАННЫХ ПРИЛОЖЕНИЙ

Андреас Антонопулос
Гэвин Вуд

Краткий глоссарий

Этот краткий глоссарий содержит многие термины, которые используются в контексте работы с Ethereum. Эти термины можно встретить на страницах данной книги, поэтому сделайте себе закладку, чтобы они всегда были у вас под рукой.

— *Assert*[1]. В языке Solidity инструкция `assert(false)` компилируется в недействительный опкод `0xfe`, который расходует весь оставшийся газ (gas) и откатывает все изменения (т. е. возвращает ошибку). Когда выполнение инструкции `assert()` завершается неудачей, это говорит о наличии существенной ошибки, это означает, что необходимо исправить код. assert() помогает убедиться, что условия, которые никогда не должны произойти, не произойдут.

— *BIP*. Предложение по улучшению Bitcoin (англ. Bitcoin Improvement Proposal). Такие предложения подаются участниками сообщества и направлены на улучшение прокотола Bitcoin. Например, BIP-21 предлагает улучшить унифицированные идентификаторы ресурсов (URI).

— *DAO*. Децентрализованная автономная организация (англ. Decentralized Autonomous Organization, сокр. DAO). Компания или другая организация, которая функционирует без иерархической модели управления. Может также обозначать одноименный смарт-контракт, запущенный 30 апреля 2016 и взломанный в июне того же года; это событие в итоге послужило мотивацией для хардфорка (англ. hardfork, см. далее по тексту определение), т. е. ветвление блокчейн-сети (с кодовым именем DAO) от блока № 1192000, которое откатило взломанный контракт и привело к разделению на две независимые системы: Ethereum и Ethereum Classic.

— *DApp* (англ. decentralized application). Децентрализованное приложение. В сущности, это смарт-контракт с пользовательским веб-интерфейсом. В более широком смысле DApp — это веб-приложение, построенное поверх открытых децентрализованных пиринговых инфраструктурных сервисов. Кроме того, многие DApp-приложения содержат децентрализованное хранилище и/или протокол с платформой для обмена сообщениями.

[1] Assert или assertion (утверждение). — *Прим. ред.*

— *Deed*. Стандарт уникальных токенов (англ. non-fungible token, или NFT), предложенный в документе ERC721. В отличие от ERC20 токены deed содержат доказательство владения и не являются взаимозаменяемыми, хотя они не имеют никакой юридической силы ни в одной из стран мира, по крайней мере, на сегодня (см. также NFT).

— *ECDSA*. Алгоритм создания цифровой подписи на основе эллиптической кривой (англ. Elliptic Curve Digital Signature Algorithm, сокр. ECDSA). Криптографический алгоритм, благодаря которому средства в Ethereum могут быть потрачены только их владельцем.

— *EIP*. Предложение по улучшению Ethereum (англ. Ethereum Improvement Proposal, сокр. EIP). Проектный документ, предоставляющий информацию сообществу Ethereum и описывающий предлагаемую реализацию нового функционала, или процесса, или среды. Подробности ищите на странице *github.com/ethereum/EIPs* (см. также ERC).

— *ENS*. Сервис имен Ethereum (англ. Ethereum Name Service, сокр. ENS). Дополнительную информацию можно получить по адресу *github.com/ethereum/ens/*.

— *EOA*. Учетная запись с внешним владельцем (англ. Externally Owned Account, или EOA). Учетная запись, созданная пользователем-человеком в блокчейн-сети Ethereum (или для него).

— *ERC*. Запрос на комментарии в Ethereum (англ. Ethereum Request for Comments, сокр. ERC). Метка, которой маркируются некоторые документы EIP в попытке определить конкретный способ использования стандарта Ethereum.

— *Ethash*. Алгоритм доказательства выполнения работы для Ethereum 1.0. Больше информации по адресу *github.com/ethereum/wiki/wiki/Ethash*.

— *EVM*. Виртуальная машина Ethereum (англ. Ethereum Virtual Machine, сокр. EVM). Виртуальная машина, основанная на стековой архитектуре и выполняющая байт-код. В Ethereum модель выполнения определяет способ изменения состояния системы в зависимости от последовательности инструкций в байт-коде и небольшого массива данных источника в виртуальной среде. Это происходит в рамках формальной модели виртуального конечного автомата[1].

— *Faucet*. Сервис (контракт faucet), распределяющий средства в виде бесплатного пробного эфира (ether), который может использоваться в тестнете.

— *Finney*. Деноминация эфира. 10^{15} finney = 1 эфир.

[1] Виртуальный конечный автомат предоставляет метод спецификации программного обеспечения для описания поведения системы управления с использованием присвоенных имен свойств элемента управления вводом и действий вывода. — *Прим. ред.*

— *Frontier*. Начальная тестовая стадия разработки Ethereum, которая началась в июле 2015 года и продолжалась до марта 2016 года.

— *Ganache*. Персональный блокчейн Ethereum, который можно использовать для проведения тестов, выполнения команд и исследования состояния с сохранением контроля над формированием цепочки блоков.

— *Geth*. Go Ethereum. Одна из самых известных реализаций протокола Ethereum, написанная на языке Go.

— *HD*-кошелек. Кошелек, в котором используются иерархически детерминированное (англ. hierarchical deterministic, сокр. HD) создание ключа и протокол передачи данных (BIP-32).

— *Homestead*. Второй этап разработки Ethereum, запущенный в марте 2016 года на блоке № 1150000.

— *ICAP*. Протокол совместного обмена клиентскими адресами (англ. Inter-exchange Client Address Protocol, или ICAP). Кодировка адресов в Ethereum, частично совместимая с IBAN (International Bank Account Number — международный номер банковского счета). Обладает гибкостью, переносимостью и поддержкой контрольных сумм. Адреса ICAP используют новый национальный псевдокод IBAN — XE, который расшифровывается как eXtended Ethereum (расширенный Ethereum) и применяется в цифровых валютах, не являющихся юридически значимыми[1] на уровне финансовых систем стран (таких как XBT, XRP и XCP).

— *Ice Age*. Хардфорк (или ветвление) сети Ethereum от блока № 200000 с целью введения экспоненциального повышения сложности (так называемой бомбы сложности, англ. difficulty bomb), которое послужило мотивацией для перехода на алгоритм Po S.

— *IDE*. Интегрированная среда разработки (англ. Integrated Development Environment, сокр. IDE). Пользовательский интерфейс, который обычно сочетает в себе редактор кода, компилятор, среду выполнения и отладчик.

— *IPFS*. Межпланетная файловая система (англ. InterPlanetary File System или сокр. IPFS). Протокол, сеть и проект с открытым исходным кодом, нацеленный на создание контентно-адресуемого, пирингового[2] метода хранения и обмена гипермедийными данными в распределенной файловой системе.

---

[1] Для цифровых валют (криптовалют) в различных странах существует определенный режим, они могут быть как запрещены, так и (условно) не запрещены, или может быть разрешено их использование. — *Прим. ред.*

[2] От англ. peer-to-peer. — *Прим. ред.*

— *KDF*. Функция формирования ключа (англ. Key Derivation Function, сокр. KDF), известная также как «алгоритм растяжения для пароля». Используется в форматах хранения ключей для защиты от взлома паролей, основанного на переборе (простом и по словарю) и радужных таблицах, и заключается в многократном хешировании кодовой фразы.

— *Keccak-256*. Криптографическая хеш-функция, используемая в Ethereum. Стандартизирована как алгоритм хеширования SHA-3[1].

— *LevelDB*. Хранилище с открытым исходным кодом вида «ключ-значение», размещаемое на диске. Реализуется в виде легковесной, узкоспециализированной библиотеки с возможными интеграциями с разными платформами.

— *METoken*. Расшифровывается как Mastering Ethereum Token. Токен ERC20 используется в этой книге в демонстрационных целях.

— *Metropolis*. Третья стадия разработки Ethereum, запущенная в октябре 2017 года.

— *Mist*. Браузер с поддержкой Ethereum, впервые созданный организацией Ethereum Foundation. Содержит кошелек в браузере, который являлся первой реализацией стандарта токенов ERC20 (Фабиан Фогельштеллер — автор ERC20, главный разработчик Mist). Кошелек Mist первым реализовал поддержку контрольных сумм в формате camelCase (EIP-55; подробнее см. раздел «Шестнадцатеричная кодировка EIP-55» на с. 125). Mist запускает полноценную ноду (узел сети) и является настоящим децентрализованным (DApp) браузером с поддержкой хранилища на основе Swarm и ENS-адресов.

— *NFT*. Невзаимозаменяемый (англ. non-fungible) токен (известный также как deed). Этот стандарт предложен в документе ERC721. Токены NFT можно отслеживать и обменивать, но каждый из них является уникальным, в отличие от взаимозаменяемых токенов ERC20. Токены NFT могут быть представлением владения цифровыми или физическими активами.

— *Parity*. Одна из самых известных интероперабельных реализаций программного обеспечения клиентской части Ethereum.

— *RLP*. Рекурсивный префикс длины (англ. Recursive Length Prefix, сокр. RLP). Стандарт, созданный разработчиками Ethereum для кодирования и сериализации объектов (структур данных) произвольной сложности и длины.

— *Seed HD*-кошелька. Значение, с помощью которого генерируются главный приватный (закрытый) ключ и код цепочки для HD-кошелька. Значение seed кошелька может быть представлено мнемоническими словами, что упрощает

[1] Подробнее см. стандарт «SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions», FIPS 202 (2–15); http://dx.doi.org/10.6028/NIST.FIPS.202. — *Прим. ред.*

ручную процедуру клонирования, резервного копирования и восстановления закрытых ключей.

— *Serenity*. Четвертая, заключительная, стадия разработки Ethereum[1]. У стадии Serenity пока нет запланированной даты выпуска.

— *Serpent*. Процедурный (императивный) язык программирования смарт-контрактов с синтаксисом, похожим на Python.

— *SHA*. Алгоритм криптографического хеширования (англ. Secure Hash Algorithm, сокр. SHA). Семейство криптографических хеш-функций, опубликованных Национальным институтом стандартов и технологий (англ. National Institute of Standards and Technology).

— *Solidity*. Процедурный (императивный) язык программирования с синтаксисом, похожим на JavaScript, C++ или Java. Является самым популярным и востребованным языком для написания смарт-контрактов в Ethereum. Создан доктором Гэвином Вудом (соавтором данной книги).

— *Spurious Dragon*. Хардфорк в блокчейне Ethereum, который произошел на блоке № 2675000 и был направлен на защиту от дополнительных векторов DoS-атак и очистку состояния (см. также Tangerine Whistle). Также механизм защиты от атаки повторного воспроизведения.

— *Swarm*. Децентрализованная (пиринговая) сеть для хранения данных, которая вместе с Web3 и Whisper используется для построения приложений DApp.

— *Szabo*. Деноминация эфира (ether). 10^{12} s*zabo* = 1 эфир.

— *Tangerine Whistle*. Хардфорк в блокчейне Ethereum, который произошел на блоке № 2463000. Был призван изменить способ подсчета газа для определенных операций, чувствительных к вводу/выводу, и очистить состояние, накопившееся в результате DoS-атаки, которая использовала низкую газовую стоимость этих операций.

— *Testnet*. Тестнет, или тестовая сеть, сокр. от англ. test network. Сеть, в которой запускаются симуляции основной сети Ethereum.

— *Truffle*. Один из самых часто используемых фреймворков для разработки в Ethereum.

— *Vyper*. Язык программирования высокого уровня, похожий на Serpent, синтаксис которого напоминает Python. Его основная цель — стать как можно более чистым функциональным языком. Создан Виталиком Бутериным.

[1] Serenity, или Ethereum 2.0 (ETH2). На дату публикации книги уже произошел переход на Ethereum 2.0 1 декабря 2020 года. — *Прим. ред.*

— *Web3*. Третья версия Интернета, изначально предложенная Гэвином Вудом и нацеленная на веб-приложения — как с централизованным механизмом владения и управления, так и построенных на основе децентрализованных протоколов.

— *Wei*. Самая мелкая деноминация эфира (ether). 10^{18} wei = 1 эфир.

— *Whisper*. Децентрализованный (пиринговый) сервис обмена сообщениями. Используется в сочетании с Web3 и Swarm для построения децентрализованных приложений (DApps).

— Адрес. Представляет собой EOA (см. определение выше. — *Авт.*) или контракт, который может принимать (конечный адрес) или отправлять (исходный адрес) транзакции в блокчейн-сети. Если быть более точным, это последние 160 бит хеша Keccak в публичном ключе ECDSA.

— Ассемблер *EVM*. Язык *Ассемблер* EVM. Байт-код EVM в человеко-читаемой форме.

— Ассемблерная вставка в *Solidity*. Ассемблер EVM внутри программы на Solidity. Поддержка ассемблерных вставок в Solidity облегчает написание кода для выполнения определенных операций.

— Атака повторного воспроизведения (англ. re-entrancy attack). Атака, при которой контракт злоумышленника вызывает контрактную функцию жертвы таким образом, чтобы во время выполнения та вызывала его рекурсивно. Это, например, может привести к хищению денежных средств путем пропуска участков контракта жертвы, которые обновляют баланс или вычисляют объем суммы списания средств.

— Байт-код *(англ.* bytecode). Набор абстрактных инструкций, предназначенных для эффективного выполнения программным интерпретатором или виртуальной машиной. В отличие от исходного человеко-читаемого кода, байт-код выражен в числовом формате.

— Библиотека. Контракт особого типа, у которого нет платежных функций и нет хранилища данных. Таким образом, он не может принимать (хранить) эфир или содержать данные. Библиотека играет роль заранее развернутого кода, который другие контракты могут использовать в своих вычислениях без права на запись.

— Блок (*англ.* block). Набор необходимой информации о транзакциях (заголовок блока), входящих в блок, и набор заголовков других блоков, известных как оммеры (англ. ommers). Блоки добавляются в сеть Ethereum майнерами.

— Блокчейн (*англ.* blockchain). В Ethereum это последовательность блоков, проверенных системой алгоритмов консенсуса PoW (Proof of work), каждый из которых указывает на цепочку предыдущих блоков вплоть до генезиз-блока

(genesis block). В отличие от протокола Bitcoin данная сеть не ограничивает максимальный размер блоков; вместо этого используются изменяющиеся лимиты на газ (gas).

— Византийский форк (или «Византийское ветвление»*)*. Первое из двух хардфорков на этапе разработки Metropolis. Включало в себя EIP-649: задержку бомбы сложности Metropolis и уменьшение награды за блок, когда форк Ice Age (см. ниже) было отложено на год, а награда за блок была уменьшена с 5 до 3 эфиров.

— Виталик Бутерин. Русско-канадский программист и писатель, известный в основном как соучредитель Ethereum и журнала *Bitcoin Magazine*.

— Внутренняя транзакция (или «сообщение»). Транзакция, отправленная с учетной записи контракта на другую учетную запись или EOA.

— Вызов сообщения. Передача сообщения от одной учетной записи к другой. Если конечная учетная запись связана с кодом EVM, виртуальная машина запустится в состоянии переданного объекта и с сообщением, с которым нужно произвести какие-то действия.

— Газ (англ. gas). Виртуальное «топливо»[1], которое используется в протоколе Ethereum для выполнения смарт-контрактов. EVM поддерживает учетный механизм, который измеряет расход газа и ограничивает потребление вычислительных ресурсов (см. «Полнота по Тьюрингу»).

— Генезис-блок (*англ.* genesis block*)*. Первый блок в блокчейн-сети, который используется для инициализации конкретной блокчейн-сети и ее внутренней криптовалюты.

— Гэвин Вуд. Британский программист, соучредитель и бывший технический директор Ethereum. В августе 2014 года он предложил язык Solidity — контрактно-ориентированный язык программирования для написания смарт-контрактов.

— Дерево Меркла, или дерево Патриции-Меркла (англ. Merkle Patricia tree). Структура данных, которая используется в Ethereum для эффективного хранения пар «ключ-значение».

— Доказательство доли владения*, или алгоритм доказательства (доли) владения* (англ. proof of stake *или сокр.* PoS). Метод, с помощью которого криптовалютный протокол блокчейна пытается достичь распределенного консенсуса. PoS просит пользователей доказать, что они владеют определенным объемом

[1] Используется как внутренняя единица системы смарт-контрактов. — *Прим. ред.*

криптовалюты (их «долей» в блокчейн-сети), чтобы они могли участвовать в проверке транзакций.

— Доказательство работы, или алгоритм доказательства работы (англ. proof of work *или сокр.* PoW). Фрагмент данных (доказательство), нахождение которого требует существенных вычислительных ресурсов. В Ethereum майнеры (англ. miners) ищут цифровое решение алгоритма Ethash, которое соответствует общесетевому уровню сложности.

— Квитанция (англ. receipt). Данные, возвращаемые клиентом Ethereum для представления результата отдельной транзакции, включая ее хеш, номер ее блока, объем расходованного газа (gas) и адрес контракта, если речь идет о его развертывании.

— Компиляция. Преобразование кода, написанного на языке программирования высокого уровня (таком как Solidity), в низкоуровневый язык (например, байт-код EVM).

— Консенсус (англ. consensus). Когда многочисленные узлы (обычно большинство узлов в сети) содержат одни и те же блоки в своих локально проверенных эталонных версиях блокчейна. Не стоит путать с правилами консенсуса.

— Константинопольский форк, или константинопольское ветвление (англ. Constantinople fork). Вторая часть этапа Metropolis, которая изначально планировалась на середину 2018 года. Среди прочего предусматривала переход на гибридный алгоритм консенсуса PoW/ Po S.

— Кошелек (англ. wallet). Программное обеспечение, хранящее секретные ключи. Используется для доступа к учетным записям Ethereum и их администрирования, а также для взаимодействия со смарт-контрактами. Чтобы улучшить безопасность, ключи могут находиться не в самом кошельке, а в автономном хранилище (например, на карте памяти или на записке на бумаге). Несмотря на свое название, кошельки никогда не хранят цифровые монеты или токены.

— Легкий клиент (англ. lightweight client). Клиент для Ethereum, который не хранит локальную копию блокчейна и не проверяет блоки и транзакции. Он предоставляет ПО с функциями кошелька, способен создавать и распространять транзакции.

— Лимит на газ. Максимальное количество газа, которое может быть использовано при выполнении транзакции или формировании блока.

— Майнер (англ. miner). Сетевой узел, который путем многократного хеширования находит доказательство работы (PoW) для новых блоков.

— Вознаграждение, или награда (англ. reward). Объем эфира, который включается в каждый блок как вознаграждение от блокчейн-сети для майнера за поиск решения PoW[1].

— Нода (узел) (*англ.* node). Клиент[2] программного обеспечения (программный клиент), который участвует в работе сети.

— Нулевой адрес. Специальный адрес в Ethereum, состоящий из одних нулей. Указывается в качестве адреса назначения для транзакций, создающих контракты.

— Одноразовый код (или *по англ.* nonce). В контексте криптографии — значение, которое можно использовать только один раз. В Ethereum такие значения применяются в двух местах: в учетных записях и в счетчиках транзакций для каждой учетной записи, с помощью которых предотвращаются атаки на основе воспроизведения; одноразовый код в алгоритме PoW — это случайное значение в блоке, который был использован для доказательства работы.

— Оммер*, или оммер-блок* (англ. ommer). Дочерний блок предка, который сам не является предком. При нахождении корректного блока другой майнер уже мог опубликовать альтернативный блок, добавленный в конец блокчейна. В отличие от Bitcoin, в Ethereum «осиротевшие блоки» могут становиться оммер-блоками в процессе майнинга и получать частично награду за формирование блока. Термин «оммер» является гендерно-нейтральным и обозначает узел, который находится на одном уровне с родительским узлом; иногда его называют «дядей» (англ. uncle).

— Публичный (открытый) ключ (англ. public key). Число, выведенное из приватного (закрытого) ключа с помощью однонаправленной функции. Доступен для публичного обмена и позволяет кому угодно удостовериться в том, что цифровая подпись сделана с использованием соответствующего приватного ключа.

— Полнота по Тьюрингу. Концепция, названная в честь английского математика и специалиста в области информатики Алана Тьюринга: система правил для манипуляции данными (такая как набор компьютерных инструкций, язык программирования или клеточный автомат) считается «тьюринг-полной» или «вычислительно универсальной», если с ее помощью можно симулировать любую машину Тьюринга.

---

[1] Майнер получает вознаграждения за выполнение вычислений согласно алгоритму PoW, его получает тот майнер, который сделал быстрее других вычисления (т. е. нашел доказательство работы — PoW), необходимые для формирования нового блока в сети. — *Прим. ред.*

[2] «Клиент» — это аппаратный или программный компонент вычислительной системы. — *Прим. ред.*

— Порядок от старшего к младшему. Позиционное число, указывающее на то, что старшая цифра должна находиться в начале (известное также как *big-endian*). Противоположным ему является порядок от младшего к старшему, при котором число начинается с младшей цифры.

— Правила консенсуса (англ. consensus rules). Правила проверки блока, которым следуют полные узлы, чтобы оставаться в консенсусе с другими узлами. Не стоит путать с определением «консенсуса».

— Приватный (закрытый) ключ (*англ.* private key). См. определение «секретный ключ» (англ. secret key).

— Проблема неизменяемости развернутого кода (англ. immutable deployed code problem). После развертывания код контракта (или библиотеки) становится неизменяемым. Стандартные методики разработки ПО предусматривают возможность исправления потенциальных ошибок (bugs) и добавления новых функций, что представляет определенный челлендж для развития смарт-контрактов.

— Публичный (открытый) ключ (англ. public key). Число, выведенное из приватного (закрытого) ключа с помощью однонаправленной функции. Доступен для публичного обмена и позволяет кому угодно удостовериться в том, что цифровая подпись сделана с использованием соответствующего закрытого ключа.

— Резервная функция, или функция fallback. Функция по умолчанию, которая вызывается в случае отсутствия данных или функции с задекларированным именем.

— Сатоши Накамото (англ. Satoshi Nakamoto). Имя (никнейм), использованное лицом или группой лиц, которые спроектировали Bitcoin, создали его первую эталонную реализацию и предложили первое решение проблемы двойной траты[1] для цифровой валюты. Его (их) личность остается неизвестной.

— Секретный (или закрытый) ключ (англ. secret key). Секретное число, с помощью которого пользователи Ethereum могут доказать, что они владеют учетной записью или контрактами. Для этого генерируется цифровая подпись (см. определения «публичный (открытый) ключ», «адрес», ECDSA).

— Сеть (англ. network). Отсылка к сети Ethereum. Пиринговая сеть, которая распространяет транзакции и блоки между всеми своими нодами (узлами) участниками блокчейн-сети.

— Синглтон (англ. singleton). Термин из области программирования, описывающий объект, который может существовать лишь в единственном экземпляре.

[1] В оригинале: double spending problem (англ.). — *Прим. ред.*

— Сложность. Общесетевой параметр, который определяет, какой объем вычислений требуется для доказательства работы (PoW)[1].

— Смарт-контракт, или контракт (англ. smart contract). Программа, которая выполняется в рамках вычислительной инфраструктуры Ethereum.

— Событие (англ. event). Позволяет использовать средства ведения журнала EVM. Децентрализованные приложения (DApps) могут отслеживать события и инициировать с их помощью функции обратного вызова в пользовательском интерфейсе, написанные на JavaScript. Подробнее об этом см. *solidity.readthedocs.io/en/develop/contracts.html#events*.

— Сообщение. Внутренняя транзакция, которая никогда не сериализуется и передается лишь внутри EVM.

— Транзакция по созданию контракта. Специальная транзакция с «нулевым адресом» в качестве получателя. Используется для регистрации контракта и записи его в блокчейн Ethereum (см. «нулевой адрес»).

— Транзакция. Данные, зафиксированные в блокчейне Ethereum, инициированные и подписанные с помощью учетной записи, направленные на определенный адрес. Транзакция содержит метаданные, такие как максимальный объем газа, который может пойти на ее выполнение.

— Учетная запись (англ. account). Объект, содержащий адрес, баланс, одноразовое число и, при необходимости, хранилище и код. Учетная запись может принадлежать контракту или иметь внешнего владельца (англ. externally owned account, или EOA).

— Учетная запись контракта (англ. contract account). Учетная запись с кодом, который выполняется при получении транзакции от другой учетной записи (EOA или контракта).

— Файл *keystore*. Файл в формате JSON с одним (сгенерированным произвольным образом) закрытым ключом. Для дополнительной защиты шифруется с помощью кодовой фразы.

— Форк (англ. fork)(*или ветвление в блокчейн-сети. — Ред.*). Изменение в протоколе, которое приводит к созданию альтернативной цепочки (блоков), или временное расхождение в двух потенциальных маршрутах блока во время майнинга.

— Хардфорк (англ. hardfork) *(дословно «жесткое ветвление» блокчейн-сети. Постоянное расхождение в блокчейне. — Ред.)*. Обычно происходит, когда необновленные ноды (узлы сети) не могут проверить блоки, созданные

[1] То есть для работы алгоритма PoW и формирования блока в сети. — *Прим. ред.*

обновленными нодами, которые следуют более новым правилам консенсуса. Хардфорк не стоит путать со «софтфорком» (англ. softfork, или «мягкое ветвление»), ветвлением ПО или ветвлением в Git.

— Хеш (*англ.* hash). Отпечаток ввода произвольного размера, который имеет фиксированную длину и генерируется хеш-функцией.

— Цифровая подпись (*англ.* digital signature). Короткая строка данных, которую пользователь генерирует на основе документа с помощью приватного (закрытого) ключа. Любой, у кого есть соответствующий публичный (открытый) ключ, подпись и документ, может удостовериться в том, что (а) документ «подписан» владельцем конкретного приватного ключа; и (б) он не был изменен после создания подписи.

— Энтропия. В контексте криптографии — нехватка предсказуемости или степень хаотичности. При генерации случайных данных, таких как приватные ключи, обычно алгоритмы используют источник высокой энтропии, чтобы сделать результат (вычислений) непредсказуемым.

— Эфир (англ. ether). Стандартизированная криптовалюта, которая используется в экосистеме Ethereum и покрывает стоимость газа (gas) при исполнении смарт-контрактов. Обозначается символом Ξ — прописной греческой буквой кси.

ГЛАВА 1

Что такое Ethereum?

Ethereum часто называют «всемирным компьютером». Что же это означает? Давайте начнем с определения, которое ближе к информатике, а затем попытаемся провести более практичный анализ возможностей и характеристик этой технологии, сравнивая ее с Bitcoin и другими децентрализованными платформами для обмена информацией (сокр. «блокчейнами»).

С точки зрения информатики Ethereum является детерминистическим, но, в сущности, неограниченным конечным автоматом[1], состоящим из состояния-синглтона, доступного глобально, и виртуальной машины, которая вносит изменения в это состояние.

С более практической точки зрения Ethereum представляет собой открытую, глобально децентрализованную вычислительную инфраструктуру, которая выполняет программы, так называемые смарт-контракты. Она использует блокчейн для синхронизации и хранения изменений состояния системы, а также криптовалюту под названием эфир (англ. ether) для измерения и ограничения стоимости вычислительных ресурсов.

Платформа Ethereum позволяет разработчикам создавать мощные децентрализованные приложения со встроенными экономическими функциями. Она предоставляет высокие доступность, подотчетность, прозрачность и нейтральность, но при этом ограничивает или устраняет цензуру и снижает определенные риски, связанные с контрагентами.

Сравнение с Bitcoin

Многие люди приходят в мир Ethereum с некоторым опытом работы с криптовалютами, в частности с биткойн (Bitcoin). Ethereum имеет много общего с другими открытыми блокчейнами: пиринговая (peer-to-peer, или P2P) сеть, соединяющая пользователей (участников сети); византийский отказоустойчивый алгоритм

[1] Из теории алгоритмов «конечный автомат» является математической абстракцией, моделью дискретного устройства, имеющего один вход, один выход и в каждый момент времени находящегося в одном состоянии из множества возможных. — *Прим. ред.*

консенсуса для синхронизации обновлений состояния (блокчейн с алгоритмом PoW); криптографические концепции — цифровые подписи и хеши; цифровая валюта (эфир).

Несмотря на это, по своему назначению и структуре платформа Ethereum разительно отличается от публичных блокчейнов, которые были ее предшественниками, в том числе и протокол Bitcoin.

Основная цель Ethereum состоит не в том, чтобы быть платежной сетью с цифровой валютой. Хотя эфир (ether) является неотъемлемой частью платформы и необходим для ее работы, он задумывался как утилитарная валюта[1] для оплаты использования сети Ethereum в качестве глобального компьютера.

В отличие от Bitcoin с его очень ограниченным сценарным (скриптовым) языком, платформа Ethereum задумывалась как программируемый блокчейн общего пользования, который работает на виртуальной машине, способной выполнять код произвольной и неограниченной сложности. Если язык скриптов в протоколе Bitcoin был намеренно сделан малофункциональным, способным лишь проверять условия расходования средств, то язык Ethereum является тьюринг-полным языком[2]. Это означает, что Ethereum может играть роль универсального компьютера.

Компоненты блокчейна

Публичный блокчейн (обычно) состоит из следующих компонентов.

— Пиринговая (peer-to-peer, или P2P) сеть, объединяющая участников и поддерживающая транзакции и блоки проверенных транзакций на основе протокола Gossip[3].

— Сообщения в виде транзакций, описывающих изменения их состояния.

— Набор правил для достижения консенсуса (правил консенсуса), которые определяют то, чем именно является транзакция и что такое корректный переход состояния.

— Конечный автомат, который обрабатывает транзакции в соответствии с правилами консенсуса.

[1] Служебная криптовалюта для внутреннего пользования в сети Ethereum. — *Прим. ред.*

[2] Язык обладает Тьюнинг-полнотой (Turing complete). — *Прим. ред.*

[3] Протокол для распределенных систем, состоящих из равноправных узлов. — *Прим. ред.*

ГЛАВА 4

Криптография

Одна из фундаментальных технологий, используемых в Ethereum, — криптография, которая является разделом математики, широко применяемым в компьютерной безопасности. Термин «криптография» происходит от греческого κρυπτός «скрытый» + γράφω «пишу», но данная область более широкая, чем просто изучение скрытой записи, она скорее близка к изучению шифрования. С помощью криптографии, к примеру, можно доказать, что вы знаете секрет, не раскрывая его сути (например, с помощью цифровой подписи), или убедиться в аутентичности данных (например, с помощью цифровых отпечатков, известных также как «хеши»). Эти виды криптографических доказательств являются ключевыми математическими инструментами, от которых зависит работа Ethereum (как и любых других блокчейнов), широко применяющихся в приложениях на платформе.

Стоит отметить, что на момент написания этой книги ни одна из частей протокола Ethereum не использовала шифрования; иными словами, все коммуникации с платформой Ethereum и между нодами (включая транзакционные данные) являются открытыми (публичными) и при необходимости могут быть прочитаны кем угодно. Это позволяет любым участникам проверить корректность обновлений состояния и достичь консенсуса. В будущем станут доступными такие продвинутые криптографические инструменты, как доказательство с нулевым разглашением (англ. zero knowledge proofs) и гомоморфное шифрование (англ. homomorphic encryption), которые позволят записывать в блокчейн некоторые зашифрованные вычисления, сохраняя при этом возможность консенсуса; но пока все это существует лишь в планах, хотя определенные шаги в этом направлении уже сделаны.

В этой главе мы познакомимся с некоторыми элементами криптографии, которые используются в Ethereum — в частности, с криптографией с использованием публичного (открытого) ключа (англ. public key cryptography, сокр. PKC), которая применяется для контроля за владением средствами в виде приватных ключей и адресов.

Ключи и адреса

В предыдущих главах мы узнали, что протокол Ethereum поддерживает два разных типа учетных записей: учетная запись с внешним владельцем (EOA) и учетная запись контракта. Владение эфиром со стороны OAE обеспечивается с помощью приватных ключей, адресов *Ethereum* и цифровых подписей. Приватные ключи лежат в основе всех взаимодействий пользователя с Ethereum. На самом деле адреса учетных записей выводятся непосредственно из приватных ключей: приватный ключ определяет один уникальный адрес Ethereum, который также называют учетной записью (англ. account).

Приватные ключи никоим образом не используются напрямую в системе Ethereum; они никогда в ней не передаются и не хранятся. Иными словами, приватные ключи должны оставаться закрытыми и никогда не должны фигурировать в сообщениях, передаваемых по сети, или храниться в блокчейне; для хранения и передачи в системе Ethereum предназначены исключительно адреса и цифровые подписи учетных записей. Больше информации о том, как безопасно хранить свои приватные ключи, можно найти в разделе «Контроль и ответственность» в главе 5.

Доступ и управление средствами достигается с помощью цифровых подписей, которые тоже создаются на основе приватного ключа. Чтобы транзакцию можно было добавить в блокчейн Ethereum, она должна иметь действительную цифровую подпись. Любой, у кого есть копия приватного ключа, имеет контроль над соответствующей учетной записью и всем эфиром, который на ней хранится. Исходя из того, что пользователь хранит свой приватный ключ в безопасном месте, цифровые подписи в транзакции Ethereum служат доказательством того, что он является подлинным владельцем средств, поскольку они доказывают владение приватным ключом.

В криптографических системах с открытым ключом, которые в том числе используются в Ethereum, приватные (секретные) ключи всегда идут в паре с публичными (открытыми). Публичный ключ — это как бы номер банковского счета, а приватный — секретный PIN-код; первый является публично доступным идентификатором учетной записи, а второй предоставляет контроль над ней. Приватные ключи как таковые очень редко встречаются пользователям Ethereum; они в основном хранятся (в зашифрованном виде) в специальных файлах и управляются программными кошельками.

В той части транзакции в сети Ethereum, которая отвечает за платеж, получатель средств представлен адресом, использующимся таким же образом, как реквизиты банковского счета при переводе средств. Как вы вскоре узнаете, адрес Ethereum для учетных записей EOA генерируется на основе публичного ключа.

Однако не все адреса в Ethereum представлены парами открытых-закрытых ключей; они также могут принимать форму контрактов, которые, как будет показано в главе 7, не всегда подкреплены связкой с приватными ключами.

В оставшейся части этой главы мы сначала подробнее исследуем основы криптографии и объясним математические модели, используемые в Ethereum. Затем рассмотрим генерацию, хранение и управление ключами. В конце будет дан краткий обзор различных форматов кодирования, которые применяются для представления приватных и публичных ключей и адресов.

Криптография с публичным ключом и криптовалюта

Криптография с публичным (открытым) ключом, известная также как «асимметричная криптография», является неотъемлемой частью современной информационной безопасности. Протокол обмена ключами, впервые опубликованный Мартином Хеллманом, Уитфилдом Диффи и Ральфом Мерклом в 1970-х годах, стал громадным прорывом, породившим первую большую волну общественного интереса к сфере криптографии. До этого момента знания о надежности криптографии были прерогативой правительств и хранились в тайне.

Для защиты информации асимметричная криптография использует уникальные ключи. Эти ключи основаны на математических функциях с особым свойством: их легко вычислить в одном направлении, но сложно в обратном. Благодаря им можно создавать цифровые секретные ключи и неподделываемые подписи, защищенные законами математики.

Например, умножение двух больших простых чисел является тривиальной задачей. Но, имея результат умножения, очень сложно подобрать подходящие множители (это называют проблемой факторизации простых чисел). Представьте, что мы дали вам число 8 018 009 и сказали, что оно является результатом умножения двух простых чисел. Их поиск окажется намного сложнее, чем изначальное умножение с целью получения 8 018 009.

Некоторые из этих математических функций можно легко инвертировать при наличии некоторой секретной информации. Если бы в предыдущем примере вы знали, что один из множителей равен 2003, вы могли бы легко найти другой множитель путем простого деления: 8 018 009 ÷ 2003 = 4003. Такие функции часто называют функциями с потайным входом (англ. trapdoor functions), поскольку их очень сложно инвертировать без наличия секретной информации.

Более продвинутая категория математических функций, которая применяется в криптографии, основана на арифметических операциях с эллиптической кривой. В эллиптической арифметике умножение по модулю простого числа

является тривиальным, однако деление (обратная операция) оказывается практически невозможным. Это называется проблемой дискретного логарифмирования (англ. discrete logarithm problem), и у нее нет «потайных входов», известных на сегодня. Эллиптическая криптография широко используется в современных компьютерных системах и служит фундаментом для применения приватных ключей и цифровых подписей в протоколе Ethereum (и других криптовалютных протоколах).

Если вы хотите узнать больше о современной криптографии и математических функциях, которые в ней используются, можете ознакомиться со следующими ресурсами:

— *ru.wikipedia.org/wiki/Криптография*
— *ru.wikipedia.org/wiki/Односторонняя_функция_с_потайным_входом*
— *ru.wikipedia.org/wiki/Факторизация_целых_чисел*
— *ru.wikipedia.org/wiki/Дискретное_логарифмирование*
— *ru.wikipedia.org/wiki/Эллиптическая_криптография*

В Ethereum для создания пар открытых-закрытых ключей, о которых мы говорили в этой главе, используется асимметричная криптография, известная как криптография с публичным (открытым) ключом. Речь идет о «парах», поскольку публичный (открытый) ключ генерируется из приватного (закрытого). Вместе они представляют учетную запись Ethereum, являясь публично доступным идентификатором (адресом) и, соответственно, средством приватного контроля за доступом к любому эфиру на этой учетной записи и за любой аутентификацией, которая может понадобиться при работе со смарт-контрактами. Приватный ключ управляет доступом, являясь уникальным фрагментом информации, который необходим для создания цифровых подписей; последние требуются для траты любых средств на счете учетной записи. Как вы узнаете в главе 7, цифровые подписи также применяются для аутентификации владельцев или пользователей контрактов.

В большинстве кошельков приватный и публичный ключи для удобства хранятся вместе в виде пары ключей. Однако публичный ключ можно легко вычислить из приватного, поэтому иногда его не хранят.

Цифровую подпись можно создать для любого сообщения. В Ethereum в качестве сообщения используются подробные сведения о самой транзакции. Математические концепции, стоящие за криптографией (в данном случае

эллиптической), позволяют объединить сообщение (то есть подробности о транзакции) с приватным ключом и создать тем самым код, который может быть сгенерирован только при наличии приватного ключа. Этот код называется цифровой подписью. Стоит отметить, что в Ethereum транзакция, по сути, является запросом доступа к определенной учетной записи с заданным адресом. Когда вы отправляете транзакцию в сеть Ethereum, чтобы переместить денежные средства или выполнить смарт-контракты, вы должны ее подписать с помощью закрытого ключа, который соответствует заданному адресу. Благодаря математике эллиптической кривой кто угодно может проверить подлинность транзакции, сопоставив цифровую подпись с деталями транзакции и Ethereum адресом, доступ к которому запрашивается. Проверка никоим образом не полагается на приватный ключ, остающийся приватным. Вместе с тем она однозначно подтверждает, что транзакция могла прийти только от того, кто владеет приватным ключом, который соответствует публичному ключу, стоящему за Ethereum адресом. Это «магия» асимметричной криптографии.

Шифрование не является частью протокола Ethereum — все сообщения, отправляемые в рамках сети Ethereum, могут (в случае необходимости) быть прочитаны кем угодно. Таким образом, закрытые ключи используются только для создания цифровых подписей, которые аутентифицируют транзакции.

Приватные (закрытые) ключи

Приватный (закрытый) ключ — это всего лишь число, подобранное случайным образом. Владение и контроль над закрытым ключом лежит в основе пользовательского контроля над всеми денежными средствами, связанными с соответствующим адресом Ethereum, а также доступа к контрактам, которые авторизуют этот адрес. Приватный ключ используется для создания подписей, требующих расходования эфира для подтверждения владения средствами, используемыми при проведении транзакции. Приватный ключ всегда должен оставаться секретным, поскольку его раскрытие третьим лицам эквивалентно передаче им контроля над эфиром (находящимся на адресе) и над контрактами, защищенными приватным ключом. Кроме того, он должен иметь резервные копии и быть защищен от случайной потери. В случае утраты ключа его нельзя будет восстановить, и все средства, которые связаны с данным ключом, тоже будут утеряны навсегда.

ГЛАВА 10

Токены

Слово «токен» (англ. token) происходит от староанглийского tācen, которое означает знак или символ. Оно часто используется по отношению к эмитируемым частным образом монетам (coins), сущностям, которые сами по себе имеют незначительную ценность. Это могут быть транспортные билеты с балансом (номинированные в токенах), жетоны для прачечных или игровых автоматов[1].

В наши дни под «токеном» на блокчейне[2] все чаще понимают основанную на блокчейне абстрактную единицу, которой можно владеть (пользователю) и которая может представлять активы (физические активы), валюту (средство обмена) или права доступа[3].

Незначительная ценность «токенов» во многом связана с ограниченным использованием их физических аналогов[4]. В реальном мире токены часто регистрируются для деятельности определенных бизнес-компаний, организаций или территорий, обычно имеют только одно функциональное назначение и не могут легко обмениваться (между собой)[5]. В блокчейне эти ограничения не действуют — или, если быть точным, их можно полностью поменять. Многие токены имеют несколько назначений в глобальном масштабе и подлежат обмену между собой или на другие валюты[6] на мировых рынках ликвидности. Вместе с ограничениями на использование и владение в прошлом осталась и идея «незначительной ценности».

---

[1] Англ.: transportation tokens, laundry tokens, arcade game tokens. — *Прим. ред.*

[2] Или токеном, выпущенным с использованием блокчейн-технологий. — *Прим. ред.*

[3] Англ.: assets, currency, access rights. — *Прим. ред.*

[4] То есть использование тех же токенов на блокчейне в «реальном» физическом мире. — *Прим. ред.*

[5] От англ.: exchangeable. — *Прим. ред.*

[6] Под валютами имеются в виду различные криптовалюты и токены. Криптовалюты — это активы, выпущенные на распределенном реестре или блокчейне. Токены — это активы, выпущенные в рамках блокчейн-сети, но не являющиеся основным платежным токеном, как например, эфир в рамках блокчейн-сети Ethereum является криптовалютой, а активы стандарта ERC20 — токенами. — *Прим. ред.*

В этой главе мы рассмотрим разные способы применения токенов и то, как они создаются. Мы также обсудим их характеристики, такие как взаимозаменяемость и свойства. В конце будут представлены стандарты и технологии, на которых они основаны, а также примеры построения, создания собственных токенов.

Способы применения токенов

Наиболее очевидной областью применения токенов являются частные цифровые валюты[1]. Но это лишь один из возможных вариантов. Токены можно запрограммировать для выполнения множества разных функций, которые часто имеют много общего. Например, токен может одновременно выражать право голоса, право доступа и владение ресурсом[2]. В следующем списке функционального применения токенов валюта занимает лишь один из пунктов.

— Валюта (или цифровая валюта). Токен может служить разновидностью валюты, ценность которой определяется в ходе частной торговли (обмена).

— Ресурс. Токен может представлять ресурс, добытый или произведенный в экономике совместного потребления[3] или среде с общими ресурсами[4]; например, токен хранилища[5] или токен процессора[6] представляет ресурсы, которыми можно делиться по(в) сети.

— Актив (или физический актив)[7]. Токен может представлять владение внутренним или внешним, материальным или виртуальным активом — например, золотом, недвижимостью, автомобилем, нефтью, энергией, вещами в многопользовательской игре[8] и т. д.

— Доступ[9]. Токен может предоставлять право владения или права доступа к цифровой или физической собственности, такой как форум

[1] Или «криптовалюты». — *Прим. ред.*

[2] Англ.: voting right, access right, ownership of a resource. — *Прим. ред.*

[3] Англ.: sharing economy. — *Прим. ред.*

[4] Англ.: resource-sharing environment. — *Прим. ред.*

[5] Англ.: storage token. — *Прим. ред.*

[6] Англ.: CPU token. — *Прим. ред.*

[7] Англ.: asset. — *Прим. ред.*

[8] Англ.: MMOG items. — *Прим. ред.*

[9] Англ.: access. — *Прим. ред.*

с обсуждениями, эксклюзивный веб-сайт, отдельный номер в гостинице или арендованный автомобиль.

— Капитал (или акции)[1]. Токен может представлять долю акционера (владельца) в цифровой организации (англ. Digital Organization), например в DAO, или юридическом лице, например корпорации.

— Голосование[2]. Токен может представлять право голоса в цифровой или правовой системе.

— Предмет коллекционирования. Токен может представлять цифровой (например, CryptoPunks) или физический предмет коллекционирования (например, картину).

— Идентификатор[3]. Токен может представлять цифровой идентификатор[4] (например, аватар) или юридически значимое удостоверение, в т. ч. личности[5] (например, национальный паспорт).

— Свидетельство. Токен может представлять удостоверение (сертификат) или подтверждение факта, выданное некими органами власти или децентрализованной системой репутации (например, запись о браке, свидетельство о рождении, диплом колледжа).

— Утилитарность[6]. Токен может быть использован для оплаты услуг или доступа к ним (внутри информационной[7]системы . — *Ред*.).

Часто один токен несет в себе несколько из этих функций. Иногда их сложно различить, поскольку их физические эквиваленты всегда были неразрывно связаны между собой. Например, в реальности водительские права (свидетельство) одновременно являются удостоверением личности (идентификатором), и эти два свойства данного документа нельзя разделить. В цифровом же мире функции, которые прежде были неразделимы, можно использовать по отдельности и развивать независимо друг от друга (например, использовать анонимное свидетельство).

---

[1] Англ.: equity. — *Прим. ред.*

[2] Англ.: voting. — *Прим. ред.*

[3] Англ.: identity. — *Прим. ред.*

[4] Англ.: digital identity. — *Прим. ред.*

[5] Англ.: legal identity. — *Прим. ред.*

[6] Англ.: utility. — *Прим. ред.*

[7] Или полезность для использования внутри информационной системы. — *Прим. ред.*

Токены и взаимозаменяемость

В «Википедии»[1] говорится: «В экономике взаимозаменяемость (fungibility) — такое свойство общего блага[2] или товара, когда отдельные единицы, по сути, являются (взаимно)заменяемыми друг на друга».

Токены являются взаимозаменяемыми, если мы можем заменить любую отдельную единицу (unit) токена на другую единицу без какого-либо изменения его ценности (value) или функции.

Строго говоря, если историю происхождения токенов можно проследить, их нельзя считать полностью взаимозаменяемыми. Возможность отслеживания происхождения может привести к созданию черных или белых списков, в результате чего их взаимозаменяемость понижается или вовсе устраняется.

Невзаимозаменяемыми (non-fungible) называют такие токены, которые представляют уникальные вещи материального или виртуального характера и, следовательно, не являются взаимозаменяемыми. Например, токен, представляющий владение определенной картиной Ван Гога, не является эквивалентом токена, который представляет владение картиной Пикассо, хотя оба они могут быть частью системы «токенов владельцев» (предметов искусства). Точно так же токен, представляющий определенный цифровой коллекционный предмет, такой как CryptoKitty, не является взаимозаменяемым по отношению к другому предмету CryptoKitty. У каждого уникального токена есть уникальный идентификатор, такой как серийный номер.

Примеры взаимозаменяемых и невзаимозаменяемых токенов будут представлены позже в этой главе.

Стоит отметить, что «взаимозаменяемые» токены часто используются в значении «непосредственного обмена их на деньги»[3] (например, токен в казино можно «обналичить», а токены в прачечной — как правило, нет). Это не то, что мы понимаем здесь под этим термином.

Риск контрагента

Риск контрагента заключается в том, что другая сторона транзакции может не выполнить свои обязательства. Некоторые типы транзакций подвержены

[1] *ru.wikipedia.org/wiki/Взаимозаменяемость_(экономика). — Прим. ред.*

[2] Англ.: commodity. — *Прим. ред.*

[3] Англ.: directly exchangeable for money. — *Прим. ред.*

дополнительному риску, поскольку в них участвует больше двух сторон. Например, если вы обладаете сертификатом (депозитным сертификатом) на драгоценный металл и хотите его кому-нибудь продать, в такой транзакции будет как минимум три участника: продавец, покупатель и хранитель (custodian) ценного металла. Тот, кто хранит физический актив, вынужден выступать одной из сторон совершения сделки; это добавляет риск контрагента к любой транзакции, связанной с этим активом. В целом, когда активом торгуют опосредованно на бирже, путем обмена токенами владения, хранитель актива привносит дополнительный риск — риск контрагента. Действительно ли у него есть этот металл? Признает (и позволит) ли он передачу владения путем отправки токена (например, сертификата, акта покупки-продажи, удостоверения права собственности или цифрового токена)? В мире цифровых токенов, представляющих активы, как и в реальном мире, важно понимать, кто хранит (держит) физический актив, представленный токеном, и какие правила на него распространяются.

Токены и их свойства (назначение)

Англ. слово intrinsic (свойственный) происходит от латинского intra, что означает «изнутри» (внутри . — *Ред.*).

Некоторые токены представляют цифровые сущности, которые являются внутренними по отношению к блокчейну. Такие цифровые активы регулируются правилами консенсуса, точно так же, как и сами токены. Это имеет важные последствия: токены, представляющие внутренние активы (intrinsic assets), не подвержены риску контрагента. Если у вас есть ключи к CryptoKitty, никакая другая сторона не хранит этот экземпляр (CryptoKitty) за вас — вы владеете им напрямую. Срабатывают правила консенсуса в блокчейне, и ваше владение (то есть контроль над) приватными ключами эквивалентно владению самим активом без какого-либо посредника.

С другой стороны, многие токены используются для представления внешних предметов, таких как объекты недвижимости, корпоративные голосующие акции, торговые марки и золотые слитки. Владение этими предметами, которые не находятся «внутри» блокчейна, регулируется законами, обычаями и нормами, отделенными от правил консенсуса, которые управляют токеном. Иными словами, эмитенты и владельцы токенов могут по-прежнему зависеть от реальных контрактов без приставки «смарт»[1]. Как результат, эти внешние активы (по отношению к блокчейну . — *Ред.*) подвержены дополнительному риску контрагента,

[1] Англ.: non-smart contracts. — *Прим. ред.*

поскольку они удерживаются хранителями, записываются во внешние реестры или регулируются законами и нормами за пределами среды блокчейна.

Одной из важнейших свойств токенов, основанных на блокчейне, является возможность преобразования (внешних) активов во внутренние, что устраняет риск контрагента. Хорошим примером этому может служить переход от внешнего, капитала (equity) корпорации к токенам-активам или токенам голосования в *DAO* или похожей (внутренней) организации.

Назначение токенов: утилитарные токены и токены-акции

На сегодня почти все проекты в Ethereum запускаются с каким-нибудь токеном. Но всегда ли это оправданно? Имеет ли использование токенов какие-либо недостатки или мы с вами увидим то, как воплощается в жизнь лозунг «токенизируй все»? В принципе, токены могут рассматриваться как основной управленческий или организационный инструмент. Но на практике интеграция блокчейн-платформ, включая Ethereum, с существующими общественными институтами пока показывает, что их применение (и их способность быть применимыми) имеет множество ограничений.

Для начала давайте проясним роль токенов в новых проектах (вновь запускаемых на блокчейн-платформе . — *Ред.*). Большинство проектов используют две основные формы токенизации, где токены обозначают «утилитарность» (это утилитарные токены) или «капитал» (это токены-акции). Очень часто эти две роли совмещены.

Утилитарным называют токен, использование которого необходимо для получения доступа к услуге, приложению или ресурсу. Например, он может предоставлять доступ к общему хранилищу или таким сервисам, как социальные сети.

Токены-акции представляют долю (акции) в управлении или владении корпоративным капиталом[1], например в стартапе. Данные токены могут быть неголосующими акциями, использоваться для распределения дивидендов и прибыли; но они также могут подразумевать участие в управлении децентрализованной автономной организацией с помощью комплексной системы управления на основе решений, принимаемых держателями токенов.

[1] В редакции научного редактора. — *Прим. ред.*

Это утка![1]

Многие стартапы сталкиваются со сложной проблемой: токены отлично подходят для реализации механизма фандрайзинга (сбора средств)[2], наравне с этим выпуск публичных ценных бумаг (акций) регулируется законами в большинстве стран. Выдавая токены-акции за утилитарные, многие стартапы пытаются обойти эти ограничения и собрать деньги в ходе публичного размещения (имеется в виду процедура публичного размещения монет (или токенов), часто называемая сокр. ICO. — *Ред.*) они оформляют это в виде предварительной продажи «ваучеров для доступа к услуге» или утилитарных токенов (utility tokens), как мы их называем. Время покажет, удастся ли этим тонко замаскированным предложениям убедить регуляторов «не реагировать»[3].

Как говорится в популярной пословице: «если нечто ходит как утка и крякает как утка, то это утка». Вряд ли регуляторов получится запутать этими семантическими приемами; напротив, они, скорее всего, воспримут такую правовую нечистоплотность как попытку ввести общественность в заблуждение.

Утилитарные токены: кому они нужны?

Реальная проблема состоит в том, что утилитарные токены создают для стартапов существенные риски и барьеры на пути их адаптации. Возможно, в далеком будущем все действительно будет токенизировано, но в настоящее время люди, которые разбираются в токенах и хотят их использовать, являются относительно небольшим сообществом и без того небольшого рынка криптовалют[4].

С позиции стартапа каждая инновация представляет собой риск и рыночный фильтр. Это выбор наименее известного пути, «в обход» традиций. Это одинокое путешествие. Если стартап старается внедрить инновации в новую технологическую нишу, такую как хранилище данных с пиринговыми сетями, то он уже одинок. Если добавить к этому использование утилитарных токенов в качестве инновации и предоставление услуг с помощью них, то это создает дополнительные риски и увеличивает барьеры их адаптации. Это все равно что сойти с и без

---

[1] В ориг.: It's a Duck! — *Прим. ред.*

[2] И краудфандинга. — *Прим. ред.*

[3] Практика работы, например, Комиссии по ценным бумагам США (US SEC) показала, что при наличии у выпущенных токенов признаков «капитала» (акций) они однозначно квалифицируется регулятором как ценные бумаги (security). — *Прим. ред.*

[4] В январе 2021 года капитализация криптовалюты ETH составила 84–155 млрд долл. США. — *Прим. ред.*

того одинокой тропинки инновации пирингового хранилища и начать пробираться через дикие заросли.

Представьте, что каждая инновация — это фильтр. Она ограничивает выход на рынок определенным сегментом, на котором могут появиться первые пользователи (early adopters) данной инновации. Добавление второго фильтра усиливает эффект, ограничивая доступность целевого рынок еще сильнее. Вы просите первых пользователей перейти сразу на две совершенно новые технологии: новое, только что созданное приложение (платформу и сервис) и экономику на основе токенов (токеномику, реализованную в данном приложении. — *Ред.*).

С точки зрения стартапа каждая инновация создает риски, которые повышают вероятность его провала. Если взять и так рискованную идею стартапа и добавить к ней утилитарный токен, вы получите совокупность рисков, связанных с платформой (Ethereum), экономикой в целом (биржей, ликвидностью), нормативно-правовой базой (регулирующие органы) и технологией (смарт-контрактами, стандартами токенов). Это слишком большой набор рисков для одного стартапа.

Сторонники концепции «токенизации всего» могут возразить, что вместе с этим проект получает рыночный энтузиазм, ранних пользователей, технологию, инновацию и ликвидность всей токеномики. Это тоже верно. Вопрос лишь в том, смогут ли эти преимущества и энтузиазм перевесить риски и неопределенность.

Тем не менее в мире криптовалют действительно зарождаются некоторые из самых инновационных бизнес-идей. Если регуляторы не успевают адаптировать законодательство и организовать поддержку новых бизнес-моделей, предприниматели и окружающие их талантливые специалисты будут искать возможности работы в юрисдикциях других стран, более дружелюбных к криптовалютам. Это уже происходит.

В начале этой главы во время знакомства с токенами мы использовали общеупотребительное определение «токенов»: «сущности, которые сами по себе имеют незначительную ценность». Причина незначительной ценности большинства токенов связана с тем, что их можно использовать только в очень узком функциональном контексте: в автобусах одной компании, в одной прачечной, в одном зале игровых автоматов, в одном отеле или одном магазине. Ограниченные ликвидность и применимость, высокие расходы их обмена сводят ценности токена к его стоимости. Поэтому, если вы добавите в свою платформу утилитарный токен, который можно использовать только в пределах данной платформы и небольшого рынка, вы тем самым воссоздадите условия, которые нивелируют ценность физических токенов. Это и в самом деле может быть правильным подходом к инкорпорированию токенизации в ваш проект. Однако если для

использования вашей платформы пользователю необходимо обменять что-то на ваш утилитарный токен, использовать его, затем по окончании совершить обратный обмен, чтобы получить что-то более полезное, то вы фактически создаете частную валюту. Комиссии при обмене цифровых токенов на порядки ниже, чем в реальном мире, где у их аналогов нет своего рынка, но платить все же приходится. Утилитарные токены, применяемые в целой отрасли народного хозяйства, могут оказаться очень интересными и, вероятно, довольно ценными. Но если для успеха вашего стартапа вы понимаете, что необходимо внедрить новый отраслевой стандарт, то это может означать то, что вы уже потерпели неудачу.

Одним из преимуществ развертывания сервисов на платформах общего пользования, таких как Ethereum, является возможность подключения смарт-контрактов (и, следовательно, утилитарных токенов) ко всем проектам, что повышает потенциальную ликвидность и (степень утилитарного) применения токенов.

Это решение должно иметь подходящие предпосылки. Внедряйте и адаптируйте токен, если ваше приложение не может без него работать, если он устраняет фундаментальный рыночный барьер или решает проблемы с доступом. Токен не следует использовать, если для вас это единственный способ быстрого сбора денег, если вы вынуждены делать вид, что это не публичное размещение ценных бумаг.

Токены в Ethereum

Блокчейн-токены существовали до появления Ethereum. Первая валюта, выпущенная на блокчейн, это Bitcoin, — сама в некотором роде является токеном. На основе Bitcoin и других криптовалют разработано много разных платформ с токенами, которые предшествовали Ethereum. Однако прорыв в этой области произошел в тот момент, когда на платформе Ethereum был представлен первый стандарт токенов.

Виталик Бутерин предложил токены в качестве одного из наиболее очевидных и полезных применений программируемого блокчейна общего пользования, такого как блокчейн Ethereum. На самом деле в первый год существования данной платформы Виталик и другие разработчики носили футболки с логотипом Ethereum и примера смарт-контракта на спине. Существовало несколько

ГЛАВА 13

Виртуальная машина Ethereum

В самом сердце платформы Ethereum, ее протоколе и выполняемых операциях, находится виртуальная машина — Ethereum Virtual Machine (сокр. EVM). Как можно было догадаться по ее имени, это вычислительная система, которая не сильно отличается от (других) виртуальных машин фреймворка .NET (Microsoft) или интерпретаторов языков программирования, компилируемых в байт-код, таких как Java. В этой главе мы подробно рассмотрим EVM, включая набор инструкций для установки, структуру и принципы ее работы при обновлении состояний в сети Ethereum.

Что такое EVM?

EVM — это часть Ethereum, которая отвечает за развертывание и выполнение (вычислений) смарт-контрактов. На самом деле вычисления EVM не требуются в простых транзакциях (операциях) по передаче средств от одной учетной записи EOA к другой, но все остальные операции так или иначе связаны с обновлением состояния, которое вычисляется с помощью EVM. По большому счету виртуальную машину, запущенную в блокчейне Ethereum, можно представить в виде глобального децентрализованного компьютера с миллионами исполняемых объектов, у каждого из которых есть собственное временное хранилище данных.

EVM представляет собой квази-тьюринг-полный конечный автомат[1]; приставка «квази» здесь из-за того, что любой исполняемый процесс имеет конечное количество вычислительных операций, зависящее от объема газа, доступного для выполнения смарт-контракта. Таким образом была «решена» проблема завершения работы[2] (любая программа когда-нибудь будет остановлена), и удалось избежать ситуации, когда выполнение продолжается вечно (случайно или в результате атаки) и угрожает работоспособности всей платформы Ethereum.

[1] Англ.: quasi–Turing-complete state machine. — *Прим. ред.*

[2] Англ.: halting problem. — *Прим. ред.*

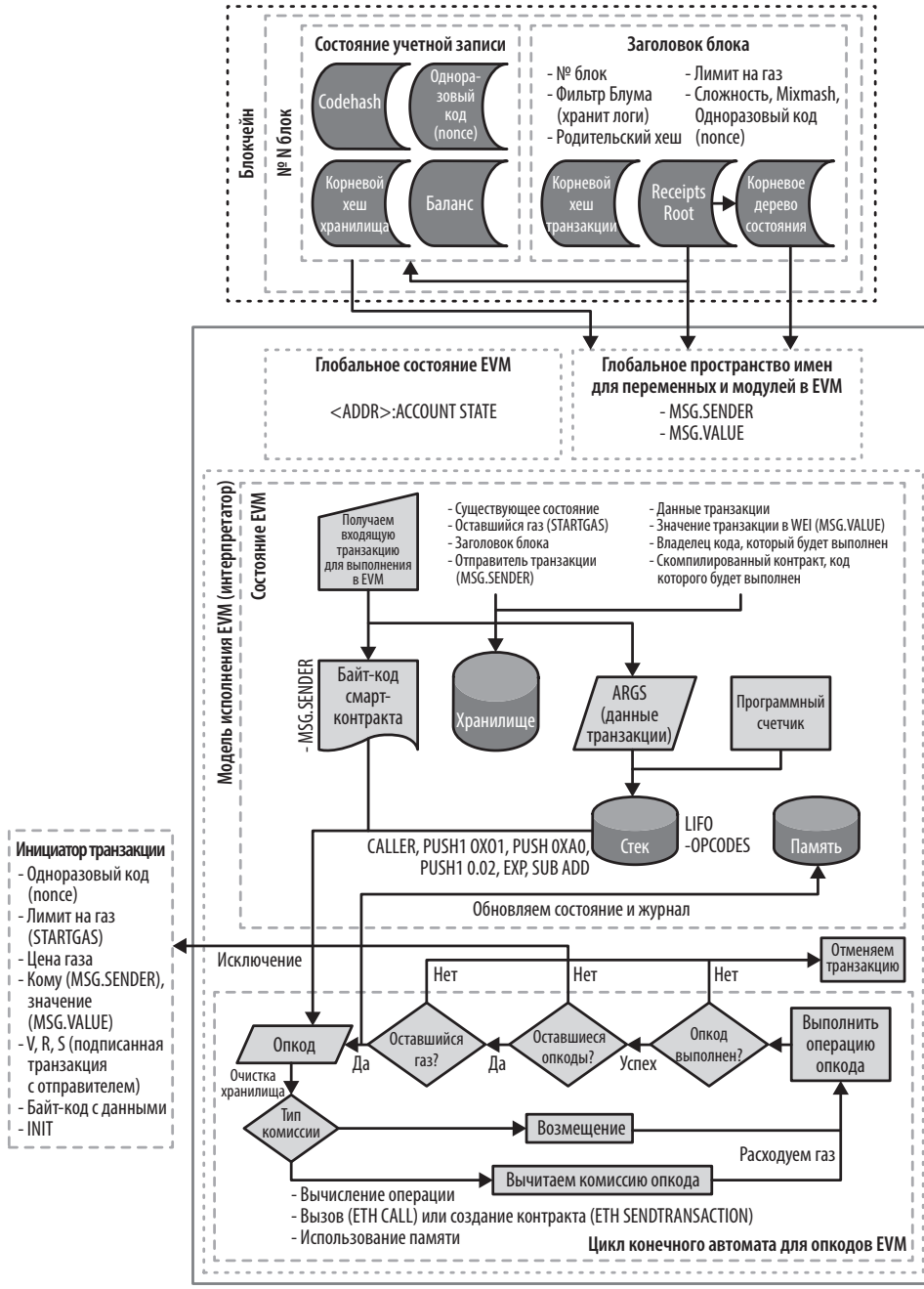

Рис. 13.1. Архитектура и контекст выполнения виртуальной машины Ethereum (EVM)

Архитектура EVM подразумевает, что все значения, которые записываются в память, попадают в стек. Она поддерживает лексемы[1] размером 256 бит (в основном для встроенной поддержки хеширования и операций с эллиптической кривой) и имеет несколько адресуемых элементов данных:

— постоянная, неизменяемая память, в которую загружается исполняемый байт-код смарт-контракта;
— временная память, каждый участок которой обнуляется во время инициализации;
— постоянное хранилище, которое является частью состояния все сети Ethereum, которое также обнуляется при инициализации.

Существует также набор переменных окружения и данных, которые доступны во время выполнения (вычислений). Мы пройдемся по ним подробнее позже в этой главе.

На рис. 13.1 показаны архитектура и операции выполнения вычислений EVM.

Сравнение с существующими технологиями

Термин «виртуальная машина» часто применяется по отношению к виртуализации компьютеров (обычно с помощью «гипервизоров», таких как VirtualBox или QEMU) или полноценных экземпляров операционной системы, таких как KVM на Linux. Что обеспечивается ПО для действующего аппаратного обеспечения (hardware) и для выполнения системных вызовов другой функциональностью ядра операционной системы[2].

Операции EVM работают в очень узкой области, где EVM продемонстрирован вычислительной машиной («движком»[3]), которая предоставляет только абстракцию для операций вычисления и хранилища, спецификация виртуальной машины схожа с Java Virtual Machine (сокр. JVM). С высокоуровневой позиции основной задачей JVM является предоставление программной среды выполнения, инкапсулирующей операционную систему и аппаратное обеспечение, на которых она запускается; это обеспечивает совместимость с множеством разнообразных информационных систем. Языки программирования высокого уровня, такие как Java, Scala (используют JVM) или C# (использует .NET), компилируются

[1] Лексема — последовательность допустимых символов языка программирования, имеющая смысл. — *Прим. ред.*

[2] Англ.: kernel functionality. — *Прим. ред.*

[3] Англ.: computation engine. — *Прим. ред.*