Etherium

Маковский Виктор, Дыбнов Никита, Неудачина Ева
Россия, Москва, 2022

1 Введение

В Тихом океане на острове Яп есть одна интересная традиция — несколько веков назад там в качестве денег использовали огромные каменные диски [1]. Перевозить и таскать их с собой было абсолютно невозможно, поэтому обычно они просто лежали где-то возле дома или на берегу океана, а все жители острова знали, где лежит каждая "монета" и кто ее владелец. Таким образом, любая транзакция должна была быть обозначена публично, чтобы все внесли изменения в "бухгалтерию" у себя в памяти. Заметим, что обмануть такую систему невозможно, ведь даже если вор физически украдет одну из монет, то он не сможет ей воспользоваться, ведь абсолютно каждый житель острова знает, что он на самом деле ей не владеет. Так же, жители острова в какой-то момент пришли к идеи, что эти огромные монеты на самом деле физически могут быть потеряны или повреждены, ведь в них как в физических объектах после их добычи и ввода в платежную систему нет никакой ценности.

Это простая и довольно интересная система, но, к сожалению, она не масштабируется и работала в тех условиях лишь из-за небольшого размера острова и количества транзакций — всем хватало памяти, чтобы держать в голове историю последних операций и обновлять свои бухгалтерии. Но такая идея, очевидно, не работает на глобальном уровне.

Но в современном мире эта проблема давно решена — мы доверяем каким-то третьим лицам: банкам или государствам, и передаем им всю ответственность за проверку и валидацию транзакций. Но банк может не заметить какие-то ошибки, потерять деньги или намеренно обмануть, например, держатели сберкнижек в СССР практически лишились своих вкладов после распада союза. Все это делает систему ненадежной.

Так давайте же как-то обойдем все упомянутые минусы и придумаем систему, которая сможет сделать доступным на мировом уровне надежное решение жителей острова Яп.

2 Blockchain

Пусть у каждого участника системы будет своя локальная бухгалтерия, которая полностью синхронизирована с бухгалтериями других. Мы всегда будем знать, сколько у кого денег, кто какие операции провел и поэтому нас никто не сможет обмануть.

В современном мире, в отличие от островитян, мы можем использовать математический и программный алгоритм, лежащий в открытом доступе, и который сам будет за счет существующих криптографических технологий верифицировать все транзакции, то есть определять, действительно ли нам хватает необходимых денег и согласилась ли на это вторая сторона, а так же возьмет на себя обязательства по поддержанию одинакового состояния всех бухгалтерий.

Все это становится возможным за счет использования криптографических ключей, которыми мы "подписываем" операции, уникальных номеров транзакций, чтобы их нельзя было скопировать, и добавим в информацию о операции какие-то данные предыдущих транзакций, чтобы убедиться, что на всех компьютерах информация находится в правильном порядке. Все это и распределение системы на всех ее участников позволяет убрать единый центр принятия решений, что делает систему более надежной.

Подобными размышлениями люди и пришли к создания технологии блокчейна.

Если коротко описать, то его главная задача — это хранить единую версию какой-то информации на разных узлах и прилагать все усилия, чтобы эта версия оставалась синхронизированной на всех компьютерах по всему миру при каждом изменении ее состояния.

Заметим, что как и жители острова Яп могли в итоге отказаться от физических монет, так и мы можем пользоваться не реальными деньгами, а криптовалютами, за счет которых мы сможем реализовать анонимные платежи, работающие на распределенной системе из многих равнозначных компьютеров.

Одной из первых популярных криптовалют был биткоин, основными алгоритмами которого был описанный выше блокчейн и "доказательство работы" (proof-of-work, PoW) — о нем будет подробно рассказано дальше.

3 Надежность Сети

Поговорим о некоторых используемых криптографических алгоритмах, без деталей их работы, но с описанием интерфейса, который они предоставляют. Подробнее про них можно почитать, например, в Ethereum Book [2]. Во всех этих алгоритмах будут фигурировать 2 объекта (можно считать, что это числа) — приватный ключ и публичный. Приватный знает только 1 человек (компьютер), публичный же лежит в открытом доступе и генерируется по приватному.

- Цифровая подпись — с помощью приватного ключа можно подписать файл, с помощью публичного можно проверить что это именно ты подписал.

Математика гарантирует, что подпись подделать очень сложно.
- Шифрование с открытым ключом — зашифровать можно публичным ключом, расшифровать только приватным.
- Хеш — необратимая функция. Хеш от любого значения вычислить легко, по значению хеш-функции угадать какое-нибудь число, хеш которого равен этому значению, вычислительно сложно.

Блокчейн использует специальные принципы консенсуса, позволяющие синхронизировать распределнную систему на всех ее узлах, поддерживает выполнение правил, проверяет, что транзакция верна, а все протоколы соблюдены. Для этого алгоритм консенсуса должен отвечать на несколько вопросов:

- Кто может изменять прошлое и как? Принцип immutability — неизменности
- Кто может изменять будущее и как? Принцип finality — заключительность
- Сколько стоит сделать такие изменения?
- Насколько децентрализованной должна быть власть, чтобы делать такие изменения?
- Кто узнает об изменениях и как они об этом узнают? В настоящий момент есть два самых популярных алгоритма, мы их опишем в общих словах, рассказывая общим идеи, но опустим детали реализации, ведь они могут отличаться в зависимости от той или иной криптовалюты

3.1 Доказательство Работы

В доказательстве работы (proof-of-work, PoW) есть два типа пользователей сети, обычные пользователи и майнеры. Обычный пользователь хранит у себя только свой баланс и адреса некоторых майнеров. Майнер же хранит всю историю всех транзакций в сети в виде блокчейна (на момент написания статьи размер блокчейна биткоина порядка нескольких сотен гигабайт).

Разберем вначале как этот алгоритм работает если все участники сети действуют честно. Вначале обычный пользователь отправляет в сеть запрос на транзакцию (обычный перевод денег или какое-то взаимодействие с смарт контрактом). Затем, майнеры получив этот запрос, проверяют валидный ли он и если да, то распространяют запрос далее остальным майнерам. Майнер, набрав некоторое количество запросов на транзакцию, объединяет их в блок и пытается подобрать специальное число для этого блока по определенным правилам. В биткойне это подбор такого числа чтобы общий хеш блока, который зависит от хеша предыдущего блока и транзакций в этом блоке, начинался на некоторое количество нулей, которое растет пропорционально суммарной мощности майнеров. В эфириуме более сложный алгоритм с графом. Подобрав такое число, тем самым доказав работу, майнер рассылает остальным майнерам свой блок. Они проверяют его на корректность, а именно

- во всех транзакциях у отправителя хватает денег
- эти транзакции не обработаны ранее (у каждой транзакции есть уникальный номер)
- все транзакции подписаны отправителями
- доказательство работы выполнено (подобрано число дающее верный хеш)

После этого они добавляют блок в свой блокчейн, удаляют из списка необработанных операций обработанные в этом блоке и начинают подбирать число для нового блока.

Что будет если мы решим обмануть систему? Если мы не майнер, а простой пользователь – ничего, майнер просто вернет нам нашу транзакцию, если она некорректна (например, попытались потратить денег больше, чем у нас есть). Если мы майнер, и пытаемся вставить блок с несуществующей транзакцией, то другие майнеры не примут его, так как он не подписан владельцем кошелька.

3.2 Доказательство Владения

Второй популярный алгоритм консенсуса в блокчейнах — доказательство владения (proof-of-stake, PoS). В этом случае блоки, объединяющие несколько транзакций, валидируются за счет общего голосования. Валидатором может стать любой пользователь, у которого есть криптовалюта, отправив специальный вид транзакции, блокирующая указанный в ней депозит. Участники голосования ставят свой депозит на блоки, которые они считают корректными. Если человек проголосовал за непопулярный в итоге вариант, то этот депозит списывается с его счета, что как раз и является системой наказания за попытку валидации ошибочного блока. В случае выигрыша варианта, за который мы проголосовали, аккаунт получает вознаграждение, пропорциональное размеру депозита. Заметим, что, если в нашей пласти находится бóльшая часть компьютеров сети, которые мы можем подговорить проголосовать за некорректный блок, чтобы украсть деньги из системы, то это будет означать, что мы владеем какой-то значительной суммой криптовалюты и в наших же интересах поддерживать корректность ее работы. Так что обмануть такую систему, конечно, можно, но невыгодно.

4 Смарт Контракты

Заметим, что идеей распределенной и синхронизированной системы можно пользоваться не только на уровне проведения денежный транзакций, но и для хранения любой другой информации пользователей.

Например, реализовывать открытые и честные голосования или заменить блокчейном работу нотариуса. Создатели платформы Ethreum увидели этот потенциал и придумали смарт-контракты — это некий код, который может быть загружен в Эфириум любым пользователем и исполняться на каком-то узле нашей распределенной системы. Да, в этой системе используется криптовалюта ether, но быть сетью для цифровых систем оплаты не является главной задачей Ethereum'a, ether в основном используется как валюта "полезности", то есть для оплаты использования платформой Ethereum как глобальным компьютером в процессе исполнения смарт-контрактов.

Использование слова "контракт" в названии может вводить вас в заблуждение, потому что на самом деле это всего лишь неизменяемая и детерминированная компьютерная программа, выполняемая на EVM — Ethreum Virtual Machine, о которой мы поговорим позже. Немного расширим определение:

- "Неизменяемая" — означает, что код смарт-контракта остается фиксированным после его создания. единственный способ его как-то изменить — создать новый
- "Детерминированная" — поведение программы зависит исключительно от транзакции, которая инициировала ее исполнение, и состояния блокчейна в этот момент. При сохранении этих вводных, сохраняется и поведение программы при разных запусках

Приведем пример использования смарт контрактов — сложные автоматизированные платежи: группа людей занимается бизнесом, завела себе смарт-контракт, клиент отправил им деньги за услугу, а контракт автоматически эти деньги по определнному заранее принципу распределил между людьми, заплатил за аренду помещения, электроэнергию и т.д. Таким образом, десятки операций произошли автоматически, при этом задать логику для этих операций можно абсолютно любую (смарт-контракты Тьюринг-полны, то есть может быть выполнен абсолютно любой набор инструкций).

Второй пример применения смарт контрактов — популярные осенью 2021 года NFT (non fungible token, не взаимозаменяемые токены). Это некие предметы, (теоретически, ничто не мешает им быть физическими) про которые смарт-контракт знает их владельца и правила, по которым их можно продавать: например, создатель токена может установить правило, что 10% от каждой продажи идут на его счет.

5 Ethereum Virtual Machine

На самом деле что такое Ethereum Virtual Machine достаточно понятно из названия. Единственное её отличие от привычных всем виртуальных машин, таких как JVM - это гораздо более скромный функционал. EVM не эмулирует работу полноценной операционной системы, она лишь предоставляет возможность исполнять элементарные действия, меняя состояние самой сети Ethereum.

Что касается деталей реализации EVM, её устройство схоже со строением примитивного компьютера. Есть несколько видов памяти с которой можно работать:

- Стек - большую часть времени низкоуровневые инструкции EVM работают именно с этой памятью. (Далее об этом подробно)
- Куча (Memory) - память используемая для хранения логов и состояния системы (находится в самом блокчейне)
- Storage - память используемая для хранения данных напрямую связанных со смарт контрактом и необходмых для его выполнения (загружается в EVM из самого смарт контракта)
- program code ROM - неизменяемая память загружаемая со смарт контрактом, хранящая код, который предстоит выполнить.

Можно заметить, что всё это очень похоже на обычный ПК, и это действительно так. Принцип работы низкоуровневых инструкций тоже максимально схож. Большая часть опкодов EVM представляет собой адаптированные опкоды ассемблера. Реализованы они похожим образом: необходимые аргументы берутся со стека, результат работы кладётся обратно на стек (в большинстве случаев). Стек, само собой, LIFO. Существенным отличием является то, что вычисления происходят в однопоточном режиме. Причина ясна, у EVM нет возможности иметь какой-либо sheduler, ведь порядок выполнения транзакций диктуется из вне (само собой однопоточность локальная, иначе было бы невозможно совершить две транзакции одновременно).

Такая сильная схожесть с работой примитивного компьютера делает неудивительным ещё один факт о работе EVM: используемый язык является Тьюринг полным. Как для любой вычислительной машины работающей с Тьюрингом полным языком, для EVM является актуальной проблема остановки (halting problem). Для того чтобы понять как она решена, нужно понять основные принципы работы Gas. Gas - это валюта, которой исчесляется стоимость всех операций, производимых EVM. В отличие от биткоина, где стоимость выполненной работы просто пропорциональна объёму (в килобайтах) выполненного кода, Gas тесно связан с затраченным временем на вычисление, затраченной памятью и текущим состоянием системы. Отсюда и вытекает решение проблемы остановки. Gas, который пользователь может потратить на выполнение смарт контракта не бесконечен. Конечно можно доплатить, чтобы продолжить работу "бесконечной программы но рано или поздно

мы всё равно упрёмся в лимит для одного блока. Таким образом язык является квази-Тьюринг-полным, что означает что любую ограниченную каким-то конкретным числом шагов программу возможно вычислить.

6 Возможные оптимизации

Существует опитимизация выполнения смартконтрактов iBatch. Её суть заключается в добавлении смарт контракта, выполняющего функцию компановщика (batcher). Основным предназначением компановщика является объединение транзакций в более "компактный дешёвый для вычисления вид. Однако трудно скомпановать что либо эффективно когда необходимо производить трансфер эфира сразу по поступлении запроса. Таким образом данная схема приспособлена лишь для экономии в процессе внутренних вычислений. Но что если отойти от идеи реализации компановщика в виде смарт контракта? Таким образом мы получим другую оптимизацию.

OCIA (Optimization Scheme for Institutional Accounts) это динамическая оптимизация вне блокчейна, созданная на основе iBatch, способная поддерживать трансфер эфира. Основная идея заключается в жертвовании безопасностью. Если использовать в качестве компоновщика не смарт контракт, а некоторый внешний сервис, не находящийся в блокчейне, это влечёт за собой множество рисков, но позволяет сэкономить до 47% средств в отдельных случаях. Один из вариантов работы компановщика: амортизирование транзакций во время верификации блокчейном получателя. До тех пор пока нет необходимости в трансфере эфира, транзакции сохраняются внутри компановщика в виде разностей конечных и начальных состояний отправителя. Грубо говоря, вместо того чтобы при появившейся возможности совершить 3 транзакции по отпрвке 100 эфира, мы совершим лишь одну по отправке 300 эфира, сэкономив тем самым GAS, который должен был быть израсходован на выполнение 3 транзакций вместо 1. Очевидным минусом данной схемы является отсутствие какого-либо выигрыша в случае низкой плотности транзакционного трафика. Более того, если данные получателя уже находятся в блокчейне, то оптимизация является просто небезопасной разновидностью iBatch из-за внешнего компановщика. На самом деле это не является как таковым минусом, ведь внедряя внешний сервис-компановщик в качестве посредника между аккаунтом и блокчейном, пользователь берёт на себя ответственность за его корректное функционирование. В пративном случае лишь сам пользователь понесёт убытки. (Подробнее про OCIA можно почитать здесь [3])

Распределение работы внутри команды

Мы все читали одни и те же источники и ознакомились со статьей, но писали разные части отчета:

- Маковский Виктор — надежность сети и принципы консенсуса
- Дыбнов Никита — EVM и оптимизации
- Неудачина Ева – введение, blockchain и смарт-контракты

Список литературы

[1] 2017. Камни Раи - огромные каменные диски, используемые в качестве валюты на островах Яп. https://kulturologia.ru/blogs/060217/33350/

[2] Gavin Wood Andreas M. Antonopoulos. 2018. Mastering Ethereum. https://github.com/ethereumbook/ethereumbook/blob/develop/04keys-addresses.asciidoc

[3] Yibo Wang and Yuzhe Tang. 2022. Enabling Cost-Effective Blockchain Applications via Workload-Adaptive Transaction Execution. https://doi.org/10.48550/ARXIV.2210.04644