Как устроен Ethereum и смарт-контракты

Децентрализованная машина Тьюринга с гарантиями блокчейна

Это продолжение поста «Блокчейн изнутри: как устроен биткоин», где я объяснял базовые понятия блокчейна: пула транзакций, цепочек блоков и майнинга. Рекомендую всем не читавшим проследовать для начала туда. Этот текст посложнее, в нём больше программерской терминологии и много отсылок к предыдущему посту.

Ethereum — второй по популярности блокчейн-проект в мире. И на мой взгляд самый интересный с точки зрения технологий.

Биткоин был точкой, с которой всё началось. Он оказался не просто системой денежных переводов, он показал миру новый способ организации сети, где гарантии завязаны не на посредников и «соглашения пользователя», а на голую математику. Так мир узнал про блокчейн — неизменяемый список с математической гарантией того, что никто не сможет подделать, изменить или удалить записанные в нём данные.

Ethereum взял идею блокчейна за основу, и применил для решения более широкого класса задач. Гарантировать не только валидность денежных переводов, но и вообще любых условий и сделок. И даже автоматизировать создание таких условий.

Сделкой можно назвать любую операцию, которая строится по шаблону *«если ..., то...»*. То же самое можно обозвать договором, контрактом, условиями — как кому привычнее. В жизни мы заключаем сделки постоянно и не только про деньги: *«Если я помогу тебе написать диплом — ты мне дашь свой PlayStation поиграть»*, *«Если я похудею к лету, то куплю себе билет в Сочи»*, и.т.д.

Главная проблема сделок — никто не может гарантировать выполнение их условий. Помог другу с дипломом, а PlayStation поиграть не дали — сиди, обижайся, пиши в спортлото. Купил билет в Сочи и полетел туда жирным — сделки с самим собой нарушать вообще одно удовольствие.

В бизнесе эту проблему решают контрактами: специальные люди пишут специальные слова на специальных бумажках с печатями, которые в случае чего можно будет отнести в какой-нибудь Арбитражный Суд. Там сидят специальные посредники, за определенную плату накажут виновника другой специальной бумажкой. В конце акта все весело платят налоги.

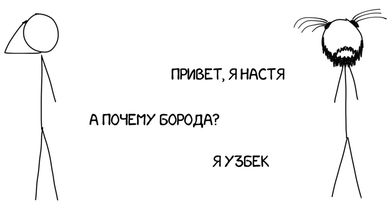
Напоминает нелепую игру, в которой я должен доверять кому-то только потому что считаю его честным. Ничего не напоминает? Снова посредники, снова комиссии, споры и обещания. Всё это очень похоже на ту проблему, которую уже решает блокчейн.

Так подумали и создатели Ethereum. И назвали это смарт-контрактами.

Олег и смарт-контракт

Снова нам поможет теоретический Олег. На этот раз Олег хочет поехать в другой город и арендовать там квартиру. Знакомая многим проблема.

В интернете Олег находит подходящий вариант — девушка Настя сдаёт свою квартиру в центре за 20000 рублей, только Славянам (Олег — Славян). Настя и Олег не знакомы, поэтому не могут доверять друг другу. Настя боится, что у Олега в последний момент изменятся планы и он не заплатит. Олег же резонно опасается, что под аккаунтом Насти в интернете сидит какой-нибудь Назарбек с жаждой наживы.



НЕ КАЖДОЙ НАСТЕ В ИНТЕРНЕТЕ МОЖНО ДОВЕРЯТЬ

Есть два варианта решения проблемы:

1. Олег и Настя подписывают длинный договор, где прописывают паспортные данные, права и обязанности сторон и другие страшные вещи. В итоге Олег обязуется заплатить 5000 рублей предоплаты и надеяться, что Настя — не Назарбек.
2. Использовать посредников, которые берут на себя ответственность с обоих сторон и хотят комиссию в 50% от сделки. Олег и Настя получают гарантии, но теряют кучу денег.

Оба так себе. Вот если бы у Олега и Насти была такая система, в которой они бы смогли прописать строгие логические правила и условия. Не просто на липовых бумажках, а на чем-то гарантированном. Как-нибудь так:

1. СОЗДАЁМ НЕЗАВИСИМОЕ ХРАНИЛИЩЕ, КУДА КАЖДЫЙ МОЖЕТ ПОЛОЖИТЬ, НО НЕ МОЖЕТ ВЗЯТЬ

2. ОЛЕГ КЛАДЁТ В ЭТО ХРАНИЛИЩЕ ДЕНЬГИ ЗА АРЕНДУ

3. НАСТЯ КЛАДЁТ ТУДА КОД ОТ ДВЕРИ СВОЕЙ КВАРТИРЫ

4. ОЛЕГУ ВЫСЫЛАЕТСЯ ЭТОТ КОД, НАСТЕ – ПОДТВЕРЖДЕНИЕ АРЕНДЫ НА ВЫБРАННЫЕ ДАТЫ

5. ЕСЛИ ОЛЕГ ПРИЕЗЖАЕТ И ВВОДИТ КОД, НАСТЕ ПЕРЕЧИСЛЯЕТСЯ СУММА ПРЕДОПЛАТЫ

6. ЕСЛИ КОД НЕ ПОДХОДИТ, ОЛЕГУ ВОЗВРАЩАЕТСЯ ВСЯ СУММА И КОНТРАКТ АННУЛИРУЕТСЯ

7. ЕСЛИ ОЛЕГ НЕ ПРИЕЗЖАЕТ, НАСТЕ ПЕРЕЧИСЛЯЕТСЯ СУММА НЕУСТОЙКИ, А ОЛЕГУ ВОЗВРАЩАЕТСЯ ОСТАТОК

8. ЕСЛИ ВСЁ ХОРОШО, ТО ПО ОКОНЧАНИЮ СРОКА АРЕНДЫ НАСТЕ ПЕРЕЧИСЛЯЕТСЯ ОСТАТОК СУММЫ

9. ХРАНИЛИЩЕ УНИЧТОЖАЕТСЯ, КОНТРАКТ СЧИТАЕТСЯ ИСПОЛНЕННЫМ

(ЛОГИКА ОПИСАНА УСЛОВНО И СОДЕРЖИТ ОЧЕВИДНЫЕ НЕДОЧЕТЫ)

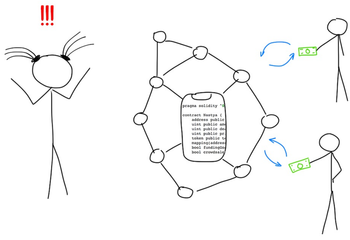
В нашей схеме предоплата становится даже лишним этапом, и оставлена исключительно для душевного спокойствия Насти. Бездушный алгоритм, написанный на языке логики, гарантирует исполнение всех заложенных в него условий. Олег не сможет внезапно забрать с виртуального кошелька свои деньги, а Настя не сможет подсунуть Олегу неправильный код или адрес квартиры — потому что тогда она не получит ничего.

Этот набор условий и есть простейший одноразовый смарт-контракт. В сети Ethereum вместо судов и бумажек с печатями, его исполнение гарантирует блокчейн — открытый и неподделываемый. Когда Олег создаст в блокчейне транзакцию «вот мои 5000 рублей предоплаты» с условием «вернуть обратно, если Настя меня обманет», то никто уже не сможет подделать блокчейн и изменить эту логику.

Нужно только как-то добавить в блокчейн поддержку вот этих «условий» — именно это и сделал Ethereum. Там такой смарт-контракт реализуется на языке Solidity (похож на JavaScript) за десяток строк кода.

Когда есть такая система, Насте уже не хочется писать и загружать новые смарт-контракты под каждого Олега. В реальности тоже никому не хочется. Ведь можно сразу описать общую логику аренды для любого желающего. Присылаешь деньги на виртуальный кошелек, тебе генерируется код от квартиры если она не занята на выбранные даты — профит. Такой контракт будет жить в системе вечно, можно даже заложить в него скидки, зависимость цены от времени года или спроса, а так же изменение условий по прямому запросу Насти.

Можно даже выложить его на гитхаб и создать универсальный смарт-контракт прозрачной аренды любой квартиры от любой Насти. Перечисляя деньги на такой контракт, любой Олег будет иметь гарантии неизменности логики блокчейна.



Языки смарт-котрактов хоть и упрощены до безобразия, но при этом обладают полнотой по Тьюрингу. Другими словами, на них можно реализовать любую логику, которая поддаётся программированию. В них есть переменные, функции, условия, циклы и даже некое подобие классов и наследования.

Такие смарт-контракты в теории можно написать для любого алгоритма. Вот это уже похоже на дивный новый мир без непонятных договоров и бумажек.

Но с небольшими оговорками.

Кто и как исполняет смарт-контракты?

Сложно понять весь Ethereum сразу, потому что многие вещи в нём циклически зависят друг от друга, в отличии от того же Биткоина. Так и у смарт-контрактов есть куча ограничений, которые связаны с особенностями Ethereum-блокчейна, который, в свою очередь, гарантирует исполнение этих смарт-контрактов.

Так что сначала давайте разберемся в смарт-контрактах, а про измененный блокчейн Ethereum и другие непонятные пока вещи я расскажу ниже. Обещаю.

Технически смарт-контракты лучше воспринимать не как подписание договора, а как исполнение кусков кода. По сути контракт — это и есть тупо код, результат исполнения которого навсегда фиксируется в блокчейне.

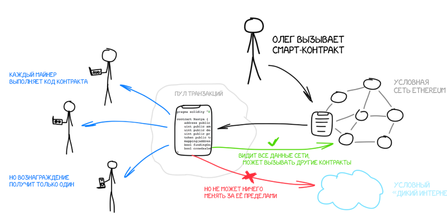
Контракт можно вызвать как функцию, совершив любую транзакцию в сети на его адрес — он вернет вам результат или ошибку.

> Пару раз я встречал целые статьи ненависти к словам «тупо код». Авторы говорят, мол, это «умный код, который дает гарантии и блаблабла». Не слушайте их. Это вам льют в уши покакавшие маркетологи, которые хотят выглядеть умными и впарить вам свои услуги для ICO. Для технарей это тупо код, исполняемый внутри своей виртуальной машины. Никакой магии.

Однако, смарт-контракт нельзя написать на вашем любимом языке программирования. На то есть две причины:

1 У каждой операции в контракте должна быть возможность в любой момент забыть/откатить все изменения, как будто их не существовало. Когда кто-то вызывает функцию смарт-контракта, все майнеры в сети одновременно пытаются исполнить код этой функции, чтобы включить её результат в свой новый блок. Но блок сможет добавить только один, а остальным придется забыть все изменения. Об этом будет подробнее рассказано в разделе Майнинг.

Так вот если вы сложите два числа на компьютере — вы с легкостью сможете удалить и забыть результат. Но если у вас будет возможность сделать HTTP-запрос, то эта операция уже необратима. Получается каждый майнер сделает этот HTTP-запрос на своём компьютере и для сервера с сайтом это будет DDoS.

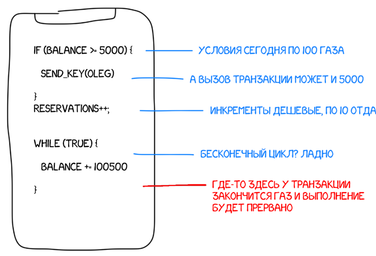


2 За выполнение операций в контракте, будь то условие, сравнение или вызов функции, нужно платить. Платит всегда тот, кто вызывает контракт. В нашем случае и Олег, и Настя заплатят за свои вызовы по копеечке майнерам.

Так сделано, чтобы избежать бесконечных циклов и чрезмерно сложных вычислений. Ведь код исполняется на компьютерах майнеров, те просто зависнут и не смогут майнить дальше.

Для этого в Ethereum используют так называемый Газ (Gas) — это маленький кусочек Эфира (ETH) — внутренней валюты. Газом оплачивается CPU майнеров, но реальные копеечки достаются только нашедшему блок — он включает их как свою комиссию.

Каждая операция внутри виртуальной машины имеет свою «цену». Можно условно представить себе это так: исполнение 1 строчки стоит 1 рубль, поэтому чтобы исполнить 15 строчек надо положить в транзакцию-вызов 15 рублей. Запоминать цены не нужно, при создании смарт-контракта редактор всё считает автоматически.



ГАЗ В ETHEREUM ОПЛАЧИВАЕТ ВЫЧИСЛЕНИЯ МАЙНЕРОВ И ЗАЩИЩАЕТ ОТ DDOS В КОДЕ

Получается код смарт-контрактов имеет доступ только к данным и вызовам внутри блокчейна Ethereum. Вы можете вызвать функцию из другого смарт-контракта, но не можете прочитать файл с диска или сходить в интернет посмотреть курс доллара.

Любой, кто хочет вызвать функцию смарт-контракта, обязан приложить к вызову немного денежек (Газа). Обычно эта сумма минимальна и её можно заработать просто включив в приложении Ethereum Wallet майнинг на пару минут.

Каждая строчка кода тратит прикрепленный к транзакции Газ. Если он внезапно кончается — исполнение прекращается и транзакция аннулируется. Если код успешно выполнился, но Газ еще остался, он возвращается отправителю как лишний. Всё честно.

Смарт-контракты в Биткоине

— инфа для самых любознательных, можно пропустить

На самом деле основы смарт-контрактов были заложены еще в блокчейне Биткоина. Помните, что каждый майнер должен верифицировать подпись для каждой транзакции, чтобы убедиться, что отправитель не пытается расплатиться чужими деньгами?

Так вот это вычисление хешей в биткоине реализовано с помощью вызова набора инструкций, который возвращают 0 или 1 в зависимости от результата. В этот набор теоретически можно добавить свою логику — там есть операторы ветвления, переменные и всякое такое. Он немудрено называется Script и похож на доисторический язык Forth.

Если заморочиться, смарт-контракт с арендой можно сделать на биткоине. Но в языке Script нет циклов и рекурсии — что лишает его полноты по Тьюрингу, а Ethereum есть. И целая виртуальная машина в придачу.

Два вида аккаунтов

До сих пор у нас были только кошельки, транзакции и блоки. Так вот смарт-контракт — это кошелек, тут его называют аккаунтом.

Если обычный кошелек управляется связкой публичного и приватного ключа к нему, то смарт-контракт — хешем от собственного кода. Изменение хотя бы одного символа в смарт-контракте, даже комментария в коде — это уже другой смарт контракт. Так гарантируется их уникальность.

Смарт-контракты создаются один раз и навсегда. Блокчейн помнит всё и в нём нельзя что-то изменить.

Классические кошельки пользователей тут принято называть — externally owned account, а созданные в сети смарт-контракты — contract account. Я буду их называть «кошелек» и «контракт», для краткости.

Общение с обоими типами аккаунтов возможно только с помощью транзакций

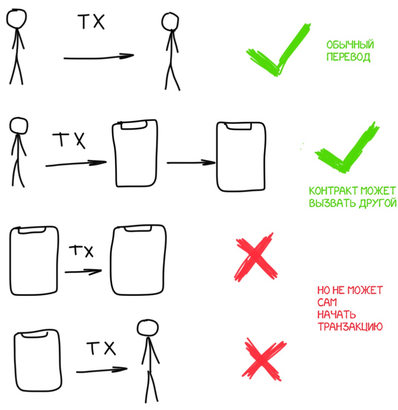
Транзакция на кошелек пользователя — это перевод средств. Полный аналог с биткоином. Перевод включает в себя количество перечисляемых ETH и адрес получателя.

Транзакция на контракт — это вызов его метода, потому её принято называть «сообщением». В неё, кроме количества и адреса контракта, включаются еще и дополнительные параметры вызова и Газ за исполнение кода.

Транзакция без получателя — это создание смарт-контракта. В такой транзакции обязательно нужно передать скомпилированный байт-код контракта и Газ за исполнение кода создания контракта (по сути конструктора).

Важная особенность: в контрактах невозможны таймеры, срабатывающие по истечению какого-то времени. Контракт может быть вызван только транзакцией, а их всегда запускает живой человек. «В фоне» контракт работать не умеет, но если его вызвали — он вполне может вызвать и другой контракт.

Получается в нашем примере про возврат предоплаты, Олегу должен сам запросить её у контракта.



Состояние вместо истории

— плавно переходим к понимаю блокчейна Ethereum

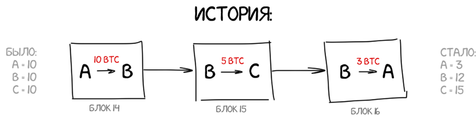
В посте про биткоин я рассказывал, что блокчейн (упрощенно) состоит из длинной цепочки всех изменений — своей истории. Чтобы посчитать текущий баланс кошелька, надо пробежать по ней и всё сложить. *Получил 5 BTC, заплатил 3 BTC, потом получил еще 4 BTC, итого баланс 5 - 3 + 4 = 6 BTC* — это и будет текущее состояние кошелька.

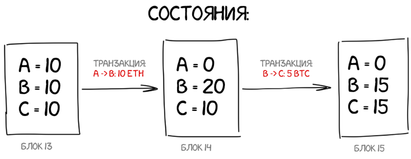
Ethereum в своем вайтпейпере примерно треть текста тратит на объяснение, почему цепочка изменений и цепочка состояний по сути одно и то же.

Состояние — это слепок всех изменений на определенный момент

История и Состояния — это не какие-то там разные сущности, а два подхода к пониманию одного и того же. Технически даже биткоин-кошельки внутри себя превращают историю в состояние для простоты.

> Похоже на срач между функциональным и императивным программированием. Два категорически разных подхода в описании одного и того же — логики задачи. Может для кого-то эта аналогия будет понятнее.



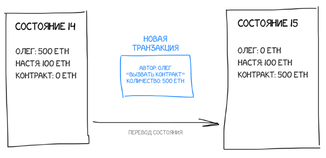


Понимание блокчейна как истории состояний сильно упрощает картину. Больше не надо бегать по истории и искать непотраченные транзакции, чтобы показать необходимый баланс, можно просто посмотреть состояние сети на текущий момент.

Ethereum — это транзакционная машина состояний. Набор текущих балансов кошельков и данных контрактов, который изменяется путём создания новых транзакций.

Создатели Ethereum модифицировали классический блокчейн, добавив в него одну важную особенность — хранилище (сторадж). Проще всего его представить как единый для всех GitHub-репозиторий, который скачивается вместе с блокчейном.

Каждая транзакция записывает изменения в этот репозиторий. Напоминает коммиты в гите. Когда Олег вызовет контракт Насти, в блокчейн будет записана транзакция «Олег вызвал контракт Насти», а в сторадж — «на контракте теперь лежит 5000 рублей от Олега + у Олега на балансе N - 5000 рублей». Это и будет новое состояние после изменений.

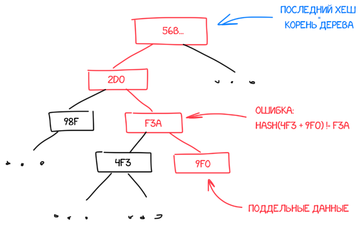


Внутри этот сторадж реализован как дерево Меркла (Merkle Tree), о котором мы говорили в прошлый раз. Тут оно немного модифицировано для оптимизации размеров и называется Patricia Tree, но суть остаётся та же — кешируем соседей пока не получим один главный хеш. Этот главный хеш — хеш корня дерева, всегда включается в каждый новый блок.

Получается, что зная один единственный блок, мы можем полностью узнать состояние системы на этот момент, прочитав из блока корень дерева хранилища.

Но это не отменяет того факта, что при первом запуске Ethereum нам придется выкачать весь блокчейн и проверить правильность всей цепочки блоков. Децентрализация — никому нельзя доверять. Вдруг нам подсунули левый блок. А вот потом уже можно и удалить лишнее.

Хранилище специально реализовано так, что новый блок записывает в него только новые изменения. Старые остаются на своих местах, на них просто создаются ссылки. Это сильно экономит место.



Транзакции — это сообщения

Когда у нас есть два вида аккаунтов и оба могут принимать транзакции, может возникнуть путаница. Для удобства транзакции между кошельками пользователей называют переводами, а транзакции с вызовом контрактов — сообщениями.

Технически это всё еще одни и те же объекты.

Для обывателя транзакция в Биткоине состоит из пяти главных элементов:

1. АДРЕС ПОЛУЧАТЕЛЯ
2. СУММА ПЕРЕСЫЛАЕМЫХ BTC
3. СПИСОК ИНПУТОВ НА ЭТУ СУММУ
   НАДО НАБРАТЬ ИЗ БЛОКЧЕЙНА ВХОДЯЩИХ ТРАНЗАКЦИЙ НА НУЖНУЮ СУММУ
4. РАЗМЕР КОМИССИИ
   ВЫЧИТАЕТСЯ ИЗ СУММЫ ИНПУТОВ, ЛИШНЕЕ ВОЗВРАЩАЕТСЯ
5. ПУБЛИЧНЫЙ КЛЮЧ ДЛЯ ПРОВЕРКИ ПОДПИСИ
   НУЖЕН ЧТОБЫ УДОСТОВЕРИТЬСЯ, ЧТО ВЫ ДЕЙСТВИТЕЛЬНО АВТОР ТРАНЗАКЦИИ

Транзакции Ethereum наследуют эту логику, но немного изменяют состав транзакций. Нам больше не надо собирать инпуты, чтобы доказать наличие средств. Каждый и так знает текущее состояние, а значит и баланс всех кошельков. Если пользователь пытается перевести деньги, которых в текущем состоянии у него нет, такая транзакция будет просто отброшена майнерами как ошибочная.

Вот список основных составляющих транзакции в Ethereum:

1. АДРЕС ПОЛУЧАТЕЛЯ
2. СУММА ПЕРЕВОДА
   ПРИ СОЗДАНИИ КОНТРАКТА ЭТО БУДЕТ ЕГО ПЕРВОНАЧАЛЬНЫЙ БАЛАНС
3. GAS LIMIT И GAS PRICE
   ТОТ САМЫЙ ГАЗ ЗА ВЫПОЛНЕНИЕ КОДА И ЕГО ЦЕНА (ПОДРОБНОСТИ НИЖЕ)
4. БАЙТ-КОД КОНТРАКТА
   ИСПОЛЬЗУЕТСЯ ТОЛЬКО ПРИ СОЗДАНИИ НОВОГО КОНТРАКТА
5. ДАННЫЕ ДЛЯ ВЫЗОВА КОНТРАКТА
   ДЛЯ ОБЫЧНЫХ ПЕРЕВОДОВ ПОЛЕ ПУСТОЕ
6. ДАННЫЕ ДЛЯ ПРОВЕРКИ ПОДПИСИ
   РАНЬШЕ ХВАТАЛО ПРОСТОГО ПУБЛИЧНОГО КЛЮЧА, НО У КОНТРАКТОВ ЕГО НЕТ ПОЭТОМУ ДЛЯ НИХ ЕСТЬ ОСОБЕННОСТИ, КОТОРЫЕ НЕ ОЧЕНЬ ИНТЕРЕСНЫ

Gas Limit и Gas Price

Именно сочетание этих двух значений теперь отвечает за комиссию, а не явное количество BTC как в Биткоине. Прикрепленный к транзакции Газ оплачивает вычисления майнеров как будто вы им заплатили.

Gas Limit — количество единиц Газа, которым оплачивается исполнение каждой строчки кода в смарт-контрактах. Цена каждой операции фиксирована в этих единицах Газа и одинакова на всех машинах. Условно: сравнение двух переменных стоит 10 Газа, а создание новой транзакции — 100. Помимо этого, Газом оплачивается и запись новых данных в сторадж системы. Сам вызов смарт-контракта тоже стоит фиксированное количество Газа, ведь загрузка его байт-кода в виртуальную машину — тоже операция. Но существуют операции, которые намеренно сделаны бесплатными. К таким относится, например, очистка временных данных, грубо говоря деструктор. Это сделано, чтобы мотивировать создателей контрактов меньше засирать глобальное хранилище.

Выполнение контракта может и не израсходовать весь приложенный Газ, тогда неиспользованный остаток просто вернется отправителю. Может случиться и обратное — газа не хватит и выполнение контракта будет прервано. В таком случае майнер получит весь приложенный газ как оплату своей работы впустую, а вы — ценный опыт, что лучше прикладывать больше Газа.

Gas Price — это цена одной единицы газа. Им можно установить более высокую или низкую цену за каждую операцию. Повышая Gas Price относительно его рыночной стоимости, можно мотивировать майнеров быстрее обработать вашу транзакцию. Вы как бы говорите «плачу за каждую строчку кода на 10% выше рынка».

Иногда повышение Gas Price заведомо необходимо. Помните ICO у Brave, которые собрали $36 млн за 24 секунды? Это меньше, чем два блока. То есть покупатели с рыночным Gas Price могли просто не успеть вскочить на хайптрейн за два блока. Майнеры бы оставили их транзакции «на потом» как дешевые, а контракт уже закрылся. Если Gas Limit измеряется просто в «штуках», то Gas Price уже в реальной криптовалюте. Цена Газа всегда ничтожно мала по сравнению с основной валютой, поэтому для таких ничтожных «копеечек» придумали свои названия:

- 1 Wei — минимальная единица расчета в системе
- 10^{12} Wei = 1 Szabo
- 10^{15} Wei = 1 Finney
- 10^{18} Wei = 1 Ether (тот самый Эфир или ETH)

GAS LIMIT		GAS PRICE		МАКС КОМИССИЯ ЗА ТРАНЗАКЦИЮ
80 000	X	500K WEI	=	0.00000004 ETH

Как любая финансовая система, Ethereum внутри оперирует только целыми числами. Проблемы с float и double известны всем нормальным студентам-айтишникам. Другими словами, Ethereum обрабатывает ETH с точностью до 18 знаков после запятой. Намного точнее, чем BTC с его 8 знаками.

Не сразу очевидная деталь: *для чтения данных из контракта не обязательно платить Газ и даже совершать транзакцию. Например, чтобы прочитать свой баланс из имеющегося контракта, можно просто скачать актуальную версию блокчейна и найти значение нужной переменной в глобальном хранилище. Такая возможность встроена в клиенты Ethereum.*

Блоки, которые...

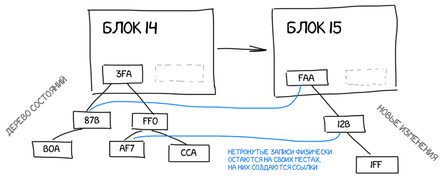
Освежаем в памяти. *Блокчейн всегда состоит из блоков. Майнеры собирают транзакции из пула и пытаются составить из них новый блок, чтобы включить в общую цепочку. Для этого они ищут какое случайное число в него добавить, чтобы блок удовлетворял неким глобальным правилам сложности.*

В Ethereum блоки немного отличаются от классического Биткоина.

Знают состояние системы

В блок всегда добавляется хеш от всего текущего хранилища. Как если бы вы сделали хеш от всех данных на вашем жестком диске. Такой слепок всегда отражает точное состояние системы на текущий момент: все данные контрактов и балансы аккаунтов. В Ethereum это просто корень дерева Меркла, о котором говорилось выше.

Хеш хранилища становится одним из важнейших компонентов блока. Он обязательно включается в каждый найденный блок и верифицируется всеми участниками сети, чтобы никто не пытался подделать состояние системы. Благо дерево Меркла позволяет делать это быстро: нужно проверить только измененные части дерева, хеши остальных остаются прежними.



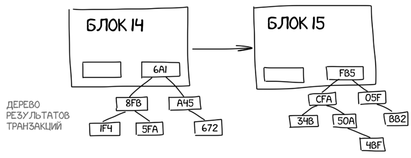
Выписывают чеки на транзакции

Представьте, что вы послали запрос на исполнение кода смарт-контракта, а тот вернул ошибку (ICO закончилось, например). Или же вы прикрепили к ней недостаточно газа и выполнение было прервано.

В обоих этих случаях транзакция будет считаться успешно выполненной с точки зрения блокчейна — ведь код был исполнен, газ был перечислен, а состояние изменено. Однако для отправителя эта транзакция должна быть помечена как неудачная. Но где? Для этого в Ethereum вводят еще одну сущность:

Помимо списка транзакций, в блок добавляется информация о результатах исполнения каждой. Так называемые «чеки», или receipts.

Теперь если я даже прикреплю 1000 единиц Газа, а на исполнение кода уйдет всего 800 — в чеке будет записано, что 800 я реально потратил, 200 мне вернут, а в транзакции будет 1000, как я её и послал.



Не забывают свои корни

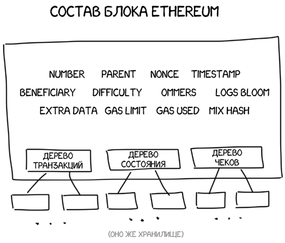
Майнеры включают в новый блок не только ссылку на его родителя (parent block), но и список ссылок на блоки, чей родитель равен родителю родителя текущего блока. Их называют «дядями» (uncles).

Тут надо закричать «Сложна!»

Проще представить это как своих реальных дядь и тёть — этот блок МОГ БЫ БЫТЬ моим родителем, если бы я майнил соседнюю цепочку (родился в соседней семье).

Зачем это нужно, расскажу чуть ниже в разделе про Майнинг и алгоритм GHOST.

В итоге блок в Ethereum выглядит как-то так:



Майнинг

— проблема 15 секунд и устойчивость к ASIC

Освежаем в памяти. *Майнеры — это куча компьютеров по всеми миру. Они одновременно собирают из транзакций новый блок, который пытаются добавить в блокчейн.*

Если майнеры начнут все одновременно анонсировать свои блоки в сеть, получится гонка, в которой невозможно определить победителя. Поэтому каждый майнер должен решить сложную задачу с легко проверяемым ответом, который он записывает в найденный блок. Первый, кто нашел ответ и анонсировал блок, получает вознаграждение 3 ETH.

Блок за 15 секунд

Сложность задачи устанавливается сетью автоматически. В биткоине сложность специально установлена огромной, чтобы в среднем по сети блоки находились раз в 10 минут. В Ethereum же новые блоки вставляются в блокчейн раз в 15 секунд.

Помните я говорил, что в биткоине нужно решить задачу поиска такого хеша, который начинается на N нулей, например 0000000000000000000523... Даже интерактив по поиску таких хешей был.

Так вот, я сильно упростил. Для человека проще понять фразу «начинается на 10 нулей», чем «имеет сложность ниже предела». На самом деле эти нули появляются в хешах блоков именно потому что в биткоине установлена такая высокая сложность, что достичь её можно только хешем с кучей нулей в начале.

Хеш блока, а это тупо число, должен быть меньше определенного установленного числа. Так гарантируются те 10 минут, за которые вся сеть находит новый блок. 15 секунд Ethereum — это ничто. Потому в Ethereum сложность такая, что нулей в начале блоков вы почти не встретите. Но она всё равно есть.

GHOST: проблема конкуренции

В сети Ethereum блоки майнятся за 15 секунд, а распространяются по всей сети примерно за 12 секунд. Приводит это к тому, что блокчейн чаще обычного находится в расщепленном состоянии — никто не может с уверенностью сказать какой из последних блоков верный, пока не найдут следующий.

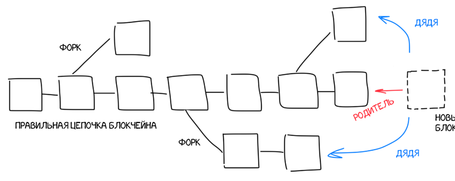
Правило самой длинной цепочки никто не отменял, и как только одна из цепочек блокчейна становится длиннее остальных, она принимается как единственно верная. Только при 15-секундном майнинге таких конкурирующих цепочек может появиться настолько много, что расщепленный блокчейн может жить часами, и в итоге откатывать и заново майнить большую часть транзакций.

Это не только неудобно когда вы пытаетесь расплатиться эфирами и ждете подтверждения по пол часа, но и таит серьезную опасность. Если майнеры большую часть времени тратят на то, чтобы майнить блоки в «ненужных» цепочках, то у них появляется мотивация объединиться в пул, где майнить вместе одну определенную цепочку, тем самым повышая шансы на успех и вознаграждение.

Мотивация объединяться в пулы приводит блокчейн к возможности «атаки 51%» — когда больше 50% мощности сосредоточено в руках одного управляющего пулом. Имея такую мощность, он сможет изменить историю блокчейна и откатить транзакции, о чем простые майнеры узнают только из постов на Реддите.

Поэтому в 2013 году был предложен модифицированный алгоритм GHOST — Greedy Heaviest Observed Subtree (Жадный поиск самой большой цепочки). Кроме понятия предыдущего блока («родителя»), он вводит понятие «дяди» блока (uncle или ommer).

GHOST несёт простой смысл: давать небольшое вознаграждение в том числе тем майнерам, которые нашли «дядю» — логически верный блок, которому просто не повезло оказаться в соседней цепочке. Дяде дают 12.5% от цены полноценного блока — это мотивирует майнеров продолжать майнить самостоятельно, ведь на поиске «дядь» тоже можно неплохо заработать.



РАНЬШЕ БЫЛИ БЫ ПРОСТО ОТБРОШЕНЫ, НО ТЕПЕРЬ ИМЕЮТ ШАНС ПОЛУЧИТЬ 12.5% ВОЗНАГРАЖДЕНИЯ

Устойчивость перед ASIC

В древние времена трава была зеленее и биткоины майнили на CPU. Потом биткоин стал расти и под майнинг приспособили видеокарты. Они не настолько универсальны как CPU и реализуют ограниченный набор операций, зато умеют делать их намного быстрее и в тысячи потоков.

Гонка вооружений продолжалась придумали ASIC — Application-Specific Integrated Circuit (интегральная схема специального назначения). Умельцы в подвалах собирали «процессоры», которые не умели вообще ничего, кроме как с огромной скоростью перебирать хеши под майнинг биткоина.

Многие считают ASIC'и читерством, но это реальность.

В Ethereum в процессе майнинга надо не только подбирать хеши, но и выполнять код смарт-контрактов. А это универсальные тьюринг-полные вычисления. Если попытаться собрать ASIC под Ethereum — это и будет CPU. Потому что нужен стек, память и всё остальное.

А в преддверии перехода на Proof-of-Stake собирать ASIC под Ethereum вообще нет никакого финансового смысла.

Proof-of-Stake

В прошлый раз я рассказывал, что классический майнинг с поиском ответа на сложную задачу называется Proof-of-Work. Главный его минус в том, что человечество тратит огромные ресурсы на его работу. Точнее оно даже не должно, но каждый хочет урвать своё вознаграждение, что приводит к постоянной гонке вооружений и майнинг-фермам, размером с половину Индии.

Ethereum сейчас тоже использует Proof-of-Work-майнинг, алгоритм зовётся Ethash, но сейчас он доживает свои последние дни. Ethereum давно собирается перейти на новый алгоритм майнинга, в котором не нужно будет скупать видеокарты и строить фермы — Proof-of-Stake.

Майнерам больше не нужно будет со скоростью света перебирать случайные числа, чтобы найти нужный хеш. Среди майнеров начинается «лотерея», когда они фиксируют у себя на счете транзакции на определенную сумму, хешируют только их и проверяют победил или не победил.

Нашел только одну короткую статью с нормальным описанием канонического Proof-of-Stake: вот эту.

В Ethereum назвали свою реализацию Casper и про неё много написано в официальном FAQ. Здесь я не привожу описание Casper, потому что сам его не до конца понимаю. Если кто-то из читателей сможет его объяснить в двух словах — напишите в комментарии здесь, будет полезно.

Собираем всё вместе

Попробуем наконец составить единую картину работы сети. Пользователи подключаются к сети как обычно скачав приложение, например Ethereum Wallet, и могут начинать делать транзакции.

Любая транзакция, будь то перевод средств, вызов или создание контракта, точно так же подписывается и уходит в общий пул неподтвержденных транзакций, где ждет пока её смайнят. Здесь нет никаких отличий от биткоина, даже несмотря на то, что в Ethereum принято мыслить «состояниями». Я почти уверен, что кошельки биткоина внутри себя тоже превращают историю в состояния.

Если транзакция — перевод между пользователями, её майнинг почти идентичен биткоину. Разве что вместо проверки инпутов здесь проверяется состояние кошельков пользователей.

Отличия начинаются, когда транзакция вызывает смарт-контракт. Тогда компьютер майнера находит этот смарт-контакт в скачанном сторадже и запускает его код с переданными параметрами внутри своей виртуальной машины (EVM — Ethereum Virtual Machine). Так делает каждый майнер, но результат в сеть сможет анонсировать только один. Поэтому все операции в контракте должны быть детерминированы и легко забываемы.

В каждом вызове контракта прописан Gas Limit — количество вычислений, которые нужно произвести для его выполнения. Если в биткоине майнеры набирали транзакции пока не достигнут нужного размера блока, здесь они решают сколько включить в блок по их суммарной сложности. Условно можно быстро выполнить сотню простеньких смарт-контрактов, а можно взять один жирный.

По итогам вычислений майнеры кроме составления дерева транзакций, перестраивают дерево состояний, выписывают receipt'ы на каждую выполненную транзакцию и включают всё это в новый блок. И только потом начинают пытаться подобрать хеш нужной сложности, чтобы включить свой блок в общий блокчейн. Кстати вместо стандартного SHA-256 в Ethereum используется хеш KECCAK-256.

Итого блокчейн Ethereum — это тот же самый классический блокчейн, к которому добавили сторадж и виртуальную машину EVM, которая позволяет исполнять произвольный код смарт-контрактов на каждом компьютере майнеров. Это стоило закрепить еще раз.

Да, по сравнению с алгоритмом биткоина всё это сначала выглядит как ад и матан. Но потом начинаешь понимать, что каждая деталь тут действительно необходима и решает определенную задачу.

Работа сети в реальном времени: ethstats.net

Две истории о смарт-контрактах

Новая технология в руках программиста — как новая бензопила для маньяка. Не терпится сразу её испробовать на всех подряд.

Расскажу две реальные истории про смарт-контракты, которые позволят лучше понять практические возможности смарт-контрактов и осознать риски от их ошибок в них.

История успеха: ICO

Сейчас появились десятки способов проведения ICO. Некоторые даже проводятся на самодельных биржах и никак не связаны с блокчейном. Но мы разберем пример классического ICO, какими они изначально технически задумывались — через смарт-контракты.

ICO — красивый пример полезного применения смарт-контрактов, и к этому моменту вы скорее всего догадываетесь, в чем он состоит. Разберем на Олегах.

Теперь Олег — мастер по производству милых плюшевых акул. Вот таких:

(Оригинал: KATYUSHKA ART DOLLS)

Олег делает плюшевых акул дома и продаёт их в интернете. Акулы стали очень популярны и Олег решил расширять бизнес, но он не может родиться в богатой семье или взять кредит в банке. Олег решает провести ICO — выпустить собственные токены, или другими словами валюту — акула-коин, продать их и собрать на этом немного денег.

Олег создаёт смарт-контракт, в котором описывает акула-коин, для этого уже есть даже готовый стандарт — ERC20. Смарт-контракт для токена абсолютно банален, в нём описаны функции «купить», «продать», «передать» и «баланс». Когда кто-то посылает на контракт транзакцию с деньгами (Эфирами), внутри смарт-контракта в простой словарь записывается «такой-то кошелек владеет столькими-то акула-коинами». Этот словарь хранится прям в блокчейне, в том самом «хранилище Ethereum», то есть виден всем желающим.

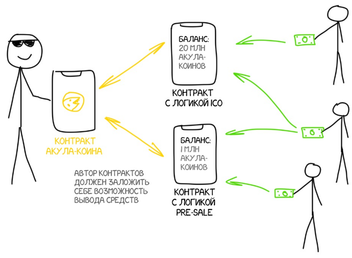
Олег загружает его в блокчейн через Ethereum Wallet и уже может идти писать на форумы «ребята, покупайте мои токены». На хайпе кто-нибудь даже купит себе парочку, как в истории с Useless ICO, где чувак решил собрать себе денег на новый компьютер, а собрал десятки тысяч долларов. Это похоже на выпуск акций, но еще не является полноценным ICO. Это скорее донаты за виртуальное вознаграждение.

> ICO-профессионалы сейчас возмутятся, мол, где же Вайтпейпер, где модный Лендинг со списком Адвайзеров, договор с Биржей и тред с краденого аккаунта на БиткоинТолке. Да, всё это пригодится, но к технической стороне не имеет никакого отношения.

Для полноценного ICO не хватает логики самой продажи: сколько стоит токен, какой лимит продажи, есть ли скидки ранним покупателям и другие особенности. Олег может заложить всю эту логику сразу в акула-коин, но тогда он уже никогда не сможет её изменить. Не сможет установить новые лимиты или устроить распродажу.

Тут я подскажу Олегу, что эту задачу решают созданием второго смарт-контракта — контракта продажи. В нём и закладывается вся логика проведения ICO: даты начала и конца, первоначальная цена, ограничения количества продаваемых токенов, и.т.д. Теперь инвесторы будут присылать деньги на него, а он уже будет решать сколько акула-коинов выдать каждому и по какой цене.

Такой контракт тоже нельзя изменить после загрузки в сеть, ведь этим гарантируется честность ICO. Зато теперь он работает только в установленные автором сроки и потом можно создать новый, устроив новый раунд или распродажу. При проведении ICO стандартной практикой является загрузка контракта в сеть заранее, чтобы пользователи успели с ним ознакомиться. Еще популярно делать pre-sale, создавая временный контракт со скидкой, который можно будет вызвать только в ограниченный срок перед продажей. Это подогревает активность участников.



После ICO акула-коины существуют по сути только как записи внутри контракта. Технически как банальный словарь (Иван: 20 коинов, Олег: 100 коинов), который теперь навсегда записан в глобальном хранилище. У циферок в словаре нет никакой плавающей цены, потому что их нельзя покупать или продавать — можно только передать или подарить кому-то.

Чтобы у проданных токенов появилась цена, Олегу надо добавить их на любую биржу. В этом случае Олег как бы переводит все собранные токены на счет биржи (со всеми очевидными опасностями её закрытия), на которой уже начинает формироваться их цена в зависимости от спроса и предложения.

Вот так в двух словах устроено ICO. Году в 2016 его даже можно было провести по этому описанию. Сейчас же на этот рынок прибежали Важные Дяди[tm] и правила игры стали сильно сложнее. Но как пример полезных смарт-контрактов ICO офигенен.

История провала: The DAO

У успеха контрактов есть и обратная сторона: история The DAO. Показательный пример того, как сделать всё правильно, но всё равно обосраться, разделить блокчейн и потерять половину коммьюнити.

Шел 2016 год, ICO еще не было массовым явлением, но Ethereum-коммьюнити уже было вдохновлено идеей тотального краудфаундинга через смарт-контракты. Тогда родился проект Decentralized Autonomous Organization. The DAO был по сути большой смарт-контракт, в котором были заложены механизмы классического инвестиционного фонда: участники вносят свои деньги, получают свою долю, с помощью голосования выбирают в какие проекты вложить собранные средства, чтобы потом срубить профит.

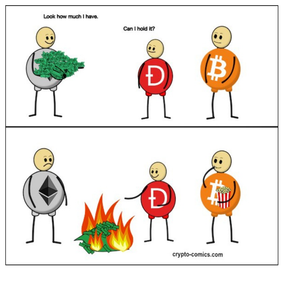
Смарт-контракт был гарантией, что никто никого не обманет и вообще будет полная демократия. В нём учли даже вариант, что некоторые участники захотят выйти из DAO и организовать собственные фонды. Например, если они не согласны с выбором проектов или просто хотят сами поиграть в инвесторов. Децентрализация внутри децентрализации — криптоанархисты в восторге.

Запуска The DAO ждали почти все, так что сразу после запуска в «фонд» прислали около $165 млн (а по текущему курсу Эфира это более $4.3 млрд). Это было большим событием в коммьюнити, даже сам Виталик поручился за его успех.

Через неделю после запуска, в коде смарт-контракта на GitHub нашли ошибку, в том самом месте, где реализовывалась логика «выйти и забрать свою долю из фонда». ~~Всем нам знакомый Race Condition — операция возврата средств выполнялась не атомарно, а в языке смарт-контрактов отсутствуют Mutex'ы~~. **fixed:** На самом деле суть дыры заключалась в том, что вместо адреса получателя доли можно было передать адрес другого смарт-контракта, который внутри себя мог попробовать еще раз запросить возврат средств до того, как главный контракт зафиксирует первый возврат у себя. И так рекурсивно вывести всё.

Так злоумышленники вывели на свои счета более $65 млн. Началась паника. Даже несмотря на то, что дыра была не в Ethereum, а в коде смарт-контракта, истерия обрушилась на самих создателей Ethereum. Толпа требовала «всё закрыть и откатить», а это просто невозможно в блокчейне. Безвыходная ситуация — ты вроде сделал хорошо, а всё равно виноват.

Это краткий пересказ истории с The DAO, за подробностями можно пройти в статью крах The DAO и разделение Ethereum.



Выхода было два: подарить злоумышленникам украденные деньги и смириться, либо «остановить» блокчейн, обновив все клиенты, откатиться до ранних блоков и запустить всё заново — по сути разделить блокчейн, сделать хард-форк. Был выбран второй вариант. Так появилось два блокчейна — Ethereum и Ethereum Classic.

Это был сильный удар по сообществу. Многие до сих пор не могут принять такое стороннее вмешательство в их независимый децентрализованный мир. *«Если создатели в любой момент могут поступить так с нашими деньгами — как мы можем доверять такому блокчейну»*, кричали они на форумах и реддите, и были в чем-то правы. Представляю каково было простым пользователям, которые в это время купили ETH где-нибудь в обменнике.

Очень неприятная история. Зато теперь, когда вы захотите отделаться от приставшего на улице криптоанархиста, просто прокричите ему в лицо пару раз «The DAO» и «ХАРДФОРК». Он упадёт и заплачет.

А вот про «пузырь», «пирамиду» и «МММ» не кричите. Это давно никого не задевает вообще.

Заключение и ссылки

Есть популярная аналогия: *«Bitcoins is Gold, Ethereum is Oil»*.

Биткоин — это золото. Редкий мягкий металл, бесполезный в быту. Из него нельзя построить дом или сделать оружие, но оно приобретает ценность когда весь мир договаривается использовать его как универсальную валюту для расчетов между собой. Поэтому золото стоит очень дорого.

Ethereum — это нефть (хотя мне больше по вкусу аналогия с Электричеством). Нефть можно добывать не только ради продажи. Вы всегда сможете использовать её для отопления, получения энергии или бензина для работы техники и заводов. Даже если никто не захочет покупать вашу нефть, она решает реальные прикладные задачи — помогает людям существовать и выживать. Как и электричество.

Мне нравится эта аналогия. Она показывает, что между Биткоином и Эфиром нет конкуренции — никто ведь не скажет, что завтра нефть сможет заменить золото или наоборот. Это два независимых ресурса, которые могут двигать всех нас вперед. А могут и рухнуть завтра как Римская Империя. Кто знает.

Спасибо каждому, кто дочитал пост до конца. Поздравляю, вы стали немного лучше понимать будущее.

- How does Ethereum work, anyway? — самая полезная статья, которую я смог нарыть. Некоторые вещи взял оттуда, автор молодец
- CoinDesk: A Beginner's Guide to Blockchain Technology — большой гайд по блокчейну вообще
- Ethereum Whitepaper — главный документ по устройству Ethereum. На удивление легко читается, моего английского даже без словаря хватило
- Перевод Ethereum Whitepaper — для тех, кому совсем тяжело в английский. Перевод правда неполный, немного устаревший и со странными комментариями переводчика
- Ethstats — залипательная страничка, где можно понаблюдать за работой сети в прямом эфире
- Что такое Ethereum и как он работает? — короткий поверхностный видос, но на редкость нормальный
- Обзор альтернатив Proof of Work. Часть 1. Proof of Stake — краткое объяснение принципов работы Proof-of-Stake

Я обновил свою страничку донатов. Там можно сказать «спасибо» за пост и стать официальным спонсором следующего. Донаты в Биткоинах и Эфирах тоже принимаются.

Ну и подпишитесь на Вастрик.Инсайд — там хорошо и не зашкварно.

Поддержать автора пивасом

Читайте Вастрик.Инсайд

Новые посты, заметки о технологиях и выживании в том киберпанке, что творится вокруг. Раз в две-три недели по почте. Лучший способ подписки, который не заблокирует Роскомнадзор. Прошлые выпуски здесь.

Эл. адрес:

Подписаться

Еще? Тогда вот

Телеграм-канал | Твиттер | Патреон | me@vas3k.ru