Oasis Decentralized Database White Paper

Dmitry Kochin[1], Alexander Neymark[2]

April, 2017 - March, 2019

[1] linkedin.com/in/kochin

[2] linkedin.com/in/neymark

Проблема информации в современном мире

Хранение и обмен информацией - ключевая проблема интернета. Каждый день сотни миллионов человек по всему миру читают информацию в интернете, добавляют новые документы, видеофайлы, картинки, заводят профили в социальных сетях, пишут сообщения в мессенджерах.

Важные свойства информации:

1. Доступ к ней должен быть **быстрым**.
1. Желательно, чтобы доступ к информации был **постоянным** (без перерывов) и **бесплатным** или стоил разумных денег.
1. Желательно иметь доступ к **правдивой** информации (уметь блокировать СПАМ и ложные новости).
1. Доступ к информации должен быть **свободным** (не должно быть цензуры, запретов и ограничений).

К сожалению, в современном мире мы часто страдаем от ложных новостей и ограничений на доступ к информации - иногда информации слишком много и мы не понимаем, где правда, а где ложь, а иногда ее так мало, что мы начинаем верить слухам, что от нас специально что-то скрывают. Некоторые сайты заблокированы, переписку в мессенджерах хотят читать и контролировать, часть новостей не доходит до тех, кому они действительно интересны и нужны.

Это большая проблема, и современное общество не в состоянии ее решить без применения новых технологий.

Какие недостатки хранения и обмена информацией существуют в современном мире?

1. **Качество связи**. Скорость распространения для информации имеет ключевое значение. Но у всех источников информации разные каналы доступа в интернет и разные способы распространения информации. У кого-то компьютер работает, у кого-то нет, кто-то лег спать, кто-то заболел, кто-то не заплатил за хостинг и его временно отключили, у кого-то сломался мобильный телефон и так далее.
1. **Правдивость информации**. Очень часто мы не знаем, кто распространяет слухи, от кого происходит фальшивая новость и кто нам врёт. Но газеты активно

перепечатывают новость в погоне за сенсацией и рейтингом. Ролики на YouTube также активно нас дезинформируют , что иногда даже может привести к панике. Бороться с этим очень сложно, так как зачастую весьма трудно вычислить источник ложной информации.

1. **Деньги за информацию.** Вы создали уникальный реферат или видеофайл и хотите его продавать. Вы его активно продаете через посредников в интернете, но вы никогда не можете определить точно, сколько копий вашего реферата или видеофайла продано, поэтому потенциально вы теряете много денег. Продавцы вам просто недоплачивают. И Вы ничего не можете с этим сделать.
1. **Цензура и блокировки**. Что здесь скажешь? Вы сами все знаете. Телеграм заблокирован. Линкедин заблокирован. Сотни сайтов заблокированы. Переписку читают. За лайки и репосты в социальных сетях могут привлечь к уголовной ответственности. Цензура в современном мире набирает обороты. Классические технологии против нее бессильны.

Как же нам быть в этом мире обмана, ограничений, цензуры и лжи? К счастью, у нас с Вами есть ОАЗИС - сеть, где есть полная свобода доступа к информации.

Что такое Оазис и как это работает?

Оазис - это децентрализованная база данных. Она работает по тому же принципу, что и блокчейн. Каждый может стать участником сети и хранить у себя на компьютере информацию организаций, юридических лиц или других пользователей.

Поскольку Оазис не имеет одного центра - его невозможно запретить, закрыть, заблокировать или ограничить к нему доступ. Пока есть хоть одна нода (компьютер), доступ к документам и файлам, которые на ней хранятся, будет постоянным.

Дайте приведем простой пример, чтобы объяснить, что такое Оазис. Все знают, что такое торренты - это технология, которая позволяет обмениваться файлами через компьютеры пользователей. У Вас на компьютере есть фильм, мы можете его раздавать разным пользователям, у которых есть доступ к файлу-торренту. Почему торренты нельзя заблокировать? Потому что нельзя заблокировать все компьютеры всех людей на земле. Пока есть хотя бы один компьютер, который раздает - вы всегда сможете скачать свой фильм.

Но что можно заблокировать? Блокируется обычно сайт с каталогом-описанием, где хранится информация о том, какой фильм находится на каком компьютере. Вот в этом принципиальная разница между торрентами и Оазисом. Торренты - это просто файлы, они не содержат в себе описание того, что это за файл, какое в нем содержание. Иными словами, вы не знаете, на чьем компьютере хранится файл с вашим любимым фильмом. Вы должны для этого искать сайт со всеми описаниями, а он часто заблокирован.

Оазис решает эту проблему: он содержит и сами файлы и описание к ним. Но Оазис - это нечто большее, чем просто система хранения файлов. Это именно база данных, где хранятся файлы, документы, тексты и можно найти часть файла по ключевому слову или по описанию.

Таким образом, Оазис открывает огромные возможности для построения бизнеса:

- Информацию можно хранить не внедряя у себя новые технологии, принципы хранения для бизнеса останутся неизменными. Нужно просто подключиться к сети Оазис (все инструкции мы предоставим).
- Никакой цензуры, никаких запретов, ваши сервисы будет очень сложно заблокировать.
- Нет рисков, что компьютер с информацией сломается или будет выключен, его мгновенно заменят другие компьютеры - доступ к информации круглосуточный.
- Очень быстро и надежно. Скорость доступа - миллионы запросов в секунду.
- Информация может храниться как открыто, так и зашифровано.
- Учет хранения осуществляет блокчейн и оплата за хранения происходит в токенах проекта Оазис. Но бизнес может платить и в рублях и в долларах по договору с компанией Эфирус (чтобы бизнес не волновался, что ему нужно осваивать покупку криптовалюты). Компания Эфирус сама купит токены и сама оплатит ими владельцам ноды.
- Безопасность того же уровня, что и ключи к биткоин-кошелькам. Информация хранится надежно и очень удобно.

Оазис - идеальная сеть нового поколения для хранения информации.

Общие данные о проекте

Данный документ представляет проект Oasis.DDB - публичную полностью децентрализованную базу данных, с возможностью полнотекстового поиска и поддержкой вторичных индексов, при сохранении скорости, подобной существующим реализациям частных (внутрикорпоративных) noSql баз данных, где серверы по-умолчанию доверяют друг другу. Такая база данных может использоваться сообществом во многих проектах, поскольку она может работать в связке с любым блокчейном, поддерживающим Тьюринг-полные смарт контракты.

Сущности проекта

Фармеры

Oasis.DDB представляет собой сеть независимых узлов. Каждый узел имеет своего владельца, будем называть его фармером. Фармером стать достаточно просто - необходимо поставить на свой сервер программное обеспечение Oasis.DDB, зарегистрировать в блокчейне вступительный депозит, и узел включится в работу сети. Когда узел обрабатывает запрос пользователя, ему начисляются ODDB токены. Таким образом, фармеры зарабатывают на содержании узла.

Пользователи

Пользователь может посылать запросы в Oasis.DDB (связь с любым узлом сети обеспечивает связь со всей сетью) и получать ответы, размещать в Oasis.DDB свои данные. Для работы ему необходим зарегистрированный в блокчейне депозит в ODDB токенах, который постепенно списывается на оплату запросов. Плата направляется тем узлам, которые участвовали в обработке запроса пользователя.

ODDB токен

Валюта взаиморасчета между узлами и пользователями сети. Пользователи платят, а узлы (фармеры) зарабатывают ODDB токены. В ODDB токенах также делаются депозиты узлами для вступления в сеть, и пользователями для оплаты запросов в сеть.

Проблема византийских генералов

Поскольку узлы Oasis.DDB независимы и не могут доверять друг другу, для обеспечения устойчивости и целостности данных необходимо принять специальные меры.

Вообще, создание публичных сетей с открытым исходным кодом, к которой каждый может присоединиться, связано с рядом трудностей. Одна из таких трудностей - византийская проблема (или “Задача Византийских генералов”). Это в обобщенном виде задача по взаимодействию нескольких независимых объектов (получивших приказ из одного центра) с целью поддержания системы взаимодействия между этими объектами в жизнеспособном состоянии продолжительное время с единой, выигрышной для всех стратегией при том, что часть объектов могут оказывать на систему вредоносное воздействие.

Формулировка задачи

Византия. Ночь перед великим сражением с противником. Византийская армия состоит из *N* легионов, каждым из которых командует свой генерал. Также, у армии есть главнокомандующий, которому подчиняются генералы.
В то же самое время, империя находится в упадке, и любой из генералов и даже главнокомандующий могут быть предателями Византии, заинтересованными в её поражении.
Ночью каждый из генералов получает от предводителя приказ о варианте действий в 10 часов утра (время одинаковое для всех и известно заранее), а именно: «атаковать противника» или «отступать».
Возможные исходы сражения:

- Если все генералы атакуют — Византия уничтожит противника (благоприятный исход).
- Если все генералы отступят — Византия сохранит свою армию (промежуточный исход).
- Если некоторые генералы атакуют, а некоторые отступят — противник уничтожит всю армию Византии (неблагоприятный исход).

Также следует учитывать, что если главнокомандующий — предатель, то он может дать разным генералам противоположные приказы, чтобы обеспечить уничтожение армии. Следовательно, генералам лучше не доверять его приказам. Если же каждый генерал будет действовать полностью независимо от других (например, делает случайный выбор), то вероятность благоприятного исхода весьма низка. Поэтому генералы нуждаются в обмене информацией между собой, чтобы прийти к единому решению.

Следствия

При решении проблем децентрализованных автономных организаций, необходимо учитывать тот факт, что объекты (например, узлы в блокчейне) полностью автономны, получают приказы от пользователей (проведение транзакции), а значит, должны действовать по объективным и заранее описанным законам и вредоносные узлы (“нелояльные генералы”) не должны своими действиями заблокировать работу системы в целом или нарушить большинство ее бизнес-процессов. Данная проблема решена для децентрализованных систем, и было доказано[3], что если сообщения содержать цифровую подпись и не могут быть незаметно изменены в момент прохождения через вредоносные узлы, то $2m$ +1 (где m - количество вредоносных узлов) работающих узлов способны обеспечить согласованную работу системы в целом.

При создании распределенной базы данных, в которой информация хранится на серверах независимых узлов, византийская проблема очень актуальна, т.к. на данный момент не существует решения, которое бы гарантировало сохранность и верность данных при высокой скорости добавления, изменения и удаления информации, а также при постоянной доступности информации на серверах узлов.

Oasis.DDB: базовая архитектура проекта

[3] M. PEASE, R, SHOSTAK, AND L. LAMPORT. Reaching Agreement in the Presence of Faults. Journal of the Association for Computing Machinery, Vol 27, No 2, April 1980, pp 228-234. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/12/Reaching-Agreement-in-the-Presence-of-Faults.pdf.

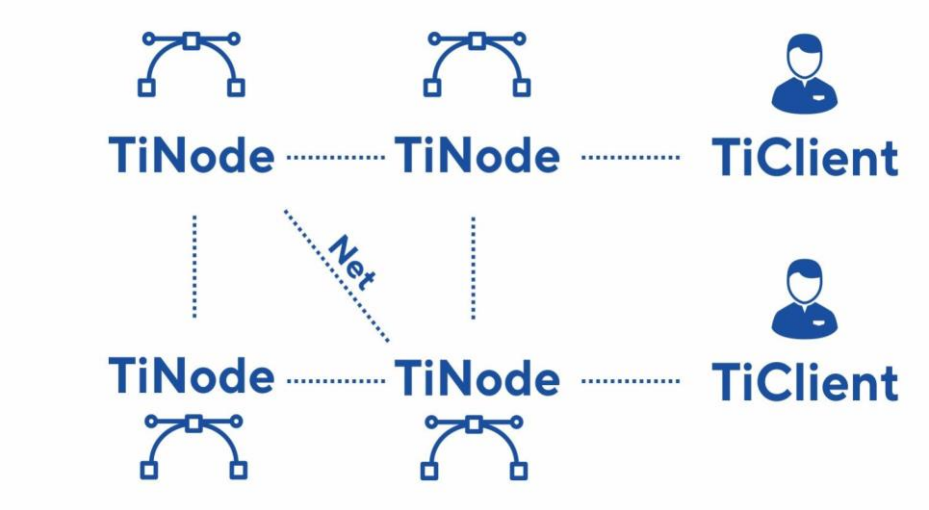

Рис. Очень укрупненный вид

Требования

Для того, чтобы платформа Oasis.DDB успешно обеспечивала выполнение возложенных на неё функций, её архитектура должна удовлетворять следующим требованиям.

1. **Децентрализация**. Платформа Oasis.DDB должна состоять из децентрализованной сети серверов (узлов). Клиентские приложения соединяются с узлами по сети.
2. **Устойчивость** - платформа должна быть устойчива к недобропорядочному поведению отдельных её компонент (устойчивость к Sybil attack, проблеме византийский генералов и т.д.).
3. **Анонимность** отдельных серверов и пользователей.
4. **Приватность** коммуникаций между серверами, пользователями и между серверами и пользователями.
5. **Объемы хранения**. Возможность хранить и осуществлять поиск по большим объемам структурированных данных.
6. **Масштабируемость** - с ростом числа пользователей и серверов скорость работы и максимальный объем хранимых данных не должны падать.
7. **Open source** - все компоненты платформы должны быть с открытым исходным кодом и опубликованы под свободными лицензиями.
8. **Публичность** - кто угодно может присоединиться к поддержке сети, установив открытое ПО системы.
9. **Выгода** - присоединяться к сети и поддерживать сеть должно быть экономически выгодно.
10. **Скорость**. Высокая скорость транзакций.
11. **Поиск**. Широкие возможности для проведения поиска данных

Далее подробно остановимся на каждом пункте.

Децентрализация и устойчивость

В основе каждой TiNode лежит блокчейн. Суть блокчейна состоит как раз в обеспечении устойчивости при децентрализации. На текущий момент на рынке есть несколько реализаций блокчейна, которые можно использовать для Oasis.DDB. Наиболее подходящим для реализации Oasis.DDB является блокчейн Etherus[4].

Блокчейн изначально обладает устойчивостью к недобропорядочному поведению отдельных участников сети. Однако, если кроме блокчейна в системе есть и другое программное обеспечение, то для обеспечения устойчивости требуются дополнительные усилия. Мы вернемся к этому вопросу, когда будем рассматривать хранение данных.

Анонимность и приватность

Нам необходимо обеспечивать анонимность и приватность не только для клиентов платформы, но и для отдельных узлов. В идеальном случае никто не должен иметь возможность узнать настоящий IP узла или клиента, который позволит установить его местонахождение. Для удовлетворения этих требований традиционно используются методы скрытия IP и шифрования трафика, такие как TOR[5] или I2P[6]. Эти методы имеют, к сожалению, существенные недостатки, главный из которых низкая скорость работы. Если работа через TOR или I2P ещё возможна между клиентом и сервером, то для синхронизация данных между серверами требуется высокая скорость соединения, которая невозможна при работе через указанные сети.

Для обеспечения требования анонимности и приватности мы предлагаем размещать все узлы и клиентов в mesh-сети Hyperboria[7], работающей по протоколу cjdns[8].

Использование mesh-сети Hyperboria имеет следующие преимущества по сравнению с другими сетями:

1. Работа как между устройствами с физической связью, так и поверх интернет.
2. Сильное end-to-end шифрование трафика - промежуточные узлы не могут прочитать или изменить трафик.
3. Поддержка IPv6 - существующее ПО будет работать в этой сети без модификаций.
4. Адреса IPv6, назначаемые узлам, никак не связаны с их местонахождением. То есть, по этому адресу нельзя узнать физическое местонахождение узла.
5. Высокая скорость работы

Есть также и недостатки

1. При работе через интернет непосредственно соединенные туннелем узлы знают IP друг друга (но только друг друга, а не любых других узлов, с которыми они не соединены туннелем непосредственно).

[4] https://etherus.org/
[5] https://www.torproject.org/
[6] https://geti2p.net/
[7] https://hyperboria.net/
[8] https://github.com/cjdelisle/cjdns

2. В угоду скорости трафик идет кратчайшим путем, соответственно, все промежуточные узлы знают, какие узлы между собой общаются (но не знают, о чем, и где они находятся).

Первый недостаток можно обойти следующими способами:

1. Узел при вступлении в сеть связывается туннелями только с доверенными узлами, если такие есть.
2. Создаётся собственный промежуточный узел Hyperboria, и узел связывается туннелем с ним. Вместо собственного узла можно использовать доверенный, если такой есть. Такой же способ могут использовать и клиенты Oasis.DDB для установления соединения с сетью.

Второй недостаток можно значительно уменьшить, динамически меняя IPv6 идентификатор и пару ключей собственного узла с течением времени. Тогда для промежуточных узлов проходящий трафик будет выглядеть как исходящий с разных узлов.

Несмотря на эти недостатки, Hyperboria удачно подходит для связывания в сеть узлов Oasis.DDB, потому что позволяет им показывать свой настоящий IP адрес только доверенным узлам или даже только доверенному существующему стороннему узлу Hyperboria - необязательно соединять узлы через прямые туннели. Любой трафик в сети автоматически шифруется. При этом скорость соединений остается высокой, и совершенно безопасно публиковать IPv6 адреса узлов для присоединения к ним клиентов и балансировки нагрузки.

Тем не менее, для клиентов, более серьезно озабоченных анонимностью можно обеспечить дополнительную возможность для соединения с узлами. Часть узлов будут опубликованы как TOR Hidden Service, и доступ к ним может быть осуществлен через TOR. Для ещё большей анонимности можно использовать TOR совместно с VPN.

Таким образом, анонимность платформы реализуется с помощью размещения её в Hyperboria, приватность обеспечивается за счет обязательного шифрования всего трафика сети, независимо от работающих поверх неё сервисов. Взыскательные клиенты могут для соединения с нодами использовать TOR + VPN.

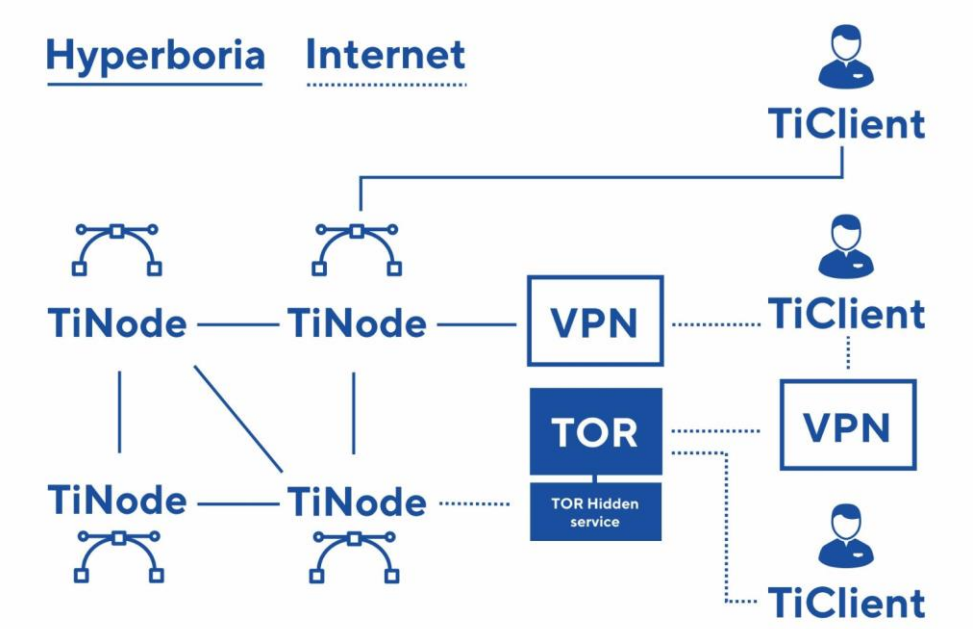

Рис. Анонимность и приватность

Хранение данных - Oasis.DDB

Oasis.DDB решает проблемы децентрализованного хранения данных, удовлетворяя при этом следующим требованиям:

1. Распределенность
2. Публичность
3. Устойчивость к проблеме византийских генералов и другим типам атак в публичной сети
4. Поддержка шардинга (возможность репликации только части данных на каждой ноде, чтобы увеличить максимальный суммарный объем данных)
5. Скорость
6. Возможность хранить структурированные данные
7. Возможность удаления данных
8. Язык запросов с возможностью осуществлять поиск не только по первичному ключу

Шардинг обеспечивает масштабируемость и высокие объемы хранения, возможность хранить структурированные данные и язык запросов позволяют проводить гибкий и быстрый поиск данных в базе.

Для децентрализации центра управления БД используется существующий блокчейн. При том, что в угоду скорости допустима нежесткая связь с блокчейном (то есть, не все транзакции будут проходить через блокчейн), она должна быть устойчива к злонамеренному поведению других узлов БД, обеспечивать достаточный уровень репликации, иметь механизмы мотивации участников на поддержку сети.

Что касается принципов хранения данных, предлагается наделить БД следующими свойствами:

1. База публичная, идентификация пользователя (клиента) базы осуществляется по его публичному ключу - это ID пользователя.
2. Каждый пользователь может слать транзакции в базу, каждая транзакция должна быть подписана этим пользователем.
3. Созданная пользователем новая запись запоминает, что он её владелец.
4. Изменять запись после создания может только её владелец (или пользователь, для которого установлено доверие через механизм разрешений, реализованный как смарт контракт на блокчейне).
5. Все могут читать все записи.
6. Чтобы между разными пользователями не возникало конфликтов ключей их записей, всем ключам записей пользователя делается префикс: ID пользователя.
7. Более сложные разрешения можно устанавливать с помощью смарт контракта в блокчейне (например, доверие между конкретными пользователями, права на создание/удаление таблиц) и т.д.
8. Все разрешения обязательно проверяются и при транзакциях, и при репликациях.

Обязательная криптографическая подпись каждой записи гарантирует, что её изменение или удаление злонамеренной стороной без знания приватного ключа владельца записи невозможно. То есть, таким образом построенное хранение данных устойчиво к византийской проблеме даже без механизма консенсуса. Это даёт основания полагать, что скорость работы такой схемы будет мало отличаться от скорости существующих реализаций noSql баз данных.

Однако злоумышленник может произвести Sybil attack, генерируя пары ключей и создавая мусорные записи в БД. Эта проблема решается введением мотивации и будет рассмотрена ниже.

Мотивация

Публичная сеть предполагает, что участники могут свободно присоединяться к ней, предоставляя оборудование, усиливающее вычислительную мощь, объемы хранения информации и распределенность сети. Для стимуляции такого поведения владельцы оборудования должны получать вознаграждение, мотивирующее их на честную работу.

Аналогично Ethereum Swarm[9] предполагаются следующие награды:

- Награда за извлечение данных
- Награда за хранение данных

Награды выделяются из средств пользователя, осуществляющего запросы. Так как платежи через блокчейн проходят весьма долго, для быстрых платежей можно использовать два метода - off-chain транзакции и “чековые книжки”. В случае off-chain транзакций пользователю будет необходимо создавать off-chain канал с каждой нодой БД (или использовать промежуточные каналы между нодами). Учитывая, что каждый такой канал требует резервирования средств на нём, это может быть довольно накладно. Поэтому мы примем в качестве основного подход “чековой книжки”. Перед обращением к базе данных пользователь должен зарезервировать часть средств на

[9] viktor trón et al. “Swap, swear and swindle incentive system for swarm”, http://swarm-gateways.net/bzz:/theswarm.eth/ethersphere/orange-papers/1/sw%5E3.pdf

специальном смарт контракте “чековая книжка”. Далее адрес этого контракта используется нодой для получения вознаграждения - контракт “чековая книжка” хранит деньги своего владельца и позволяет третьим сторонам обналичивать подписанные чеки, просто посылая транзакцию с чеком в качестве данных на метод cash контракта.

- Контракт отслеживает итоговую сумму, выписанную каждому получателю во время соединения
- Владелец при посылке чека должен обязательно запоминать общую посланную сумму

Чек обналичивается, если

- адрес контракта соответствует адресу на чеке
- чек подписан владельцем (ID пользователя - публичный ключ)
- полная сумма на чеке больше, чем в предыдущем погашеном чеке для данного получателя

Тогда при необходимости вознаградить ноду, пользователь просто шлет ей чек. Нода-получатель может сохранять только последний полученный чек от каждого пользователя и периодически обналичивать его, посылая на контракт “чековая книжка”, что позволяет при некотором доверии экономить на транзакциях блокчейна.

Награда за извлечение данных

Данные на нодах БД имеют определенный уровень репликации, то есть, данные с определенным ключом хранятся только на части нод, например, на N. Тем не менее, за данными пользователь может обратиться к любой ноде. Нода, к которой обратился пользователь, выступает далее в качестве “координатора”.

По значению ключа данных вычисляются N нод, ответственные за хранение этих ключей, и запросы направляются к ним. Возвращенные нодами данные проверяются координатором на соответствие электронным подписям, сравниваются по метке времени, и самая свежая запись возвращается пользователю.

Оплате подлежит работа координатора и реплик, хранящих данные. Пропорции оплаты подлежат более подробному расчету, но для стимуляции правильного поведения необходимо выполнять следующие принципы:

1. Чем быстрее нода вернула данные, тем большую часть оплаты она заслуживает
2. Если нода вернула старые данные, оплата уменьшается
3. Нода, не вернувшая данные, не получает ничего
4. Координатор получает фиксированную небольшую часть

Вместе с данными координатор выставляет счет, где указано, какой ноде сколько полагается. Пользователь выписывает чеки каждой. Координатор отправляет чеки нодам. А также обновленные данные, если нода не вернула ничего или вернула старые данные.

Чтобы защититься от злонамеренных координаторов и пользователей, которые не будут платить, каждая нода ведет список пользователей, от которых она ожидает оплаты, и координаторов, посылающих запросы от этих пользователей. Если уровень задолженности превышает некоторое пороговое значение, нода может перестать принимать запросы от указанных пользователей и координаторов. При получении чеков списки корректируются.

Награда за хранение

Награда за извлечение косвенным образом стимулирует и хранение, но работает только в отношении популярных и часто запрашиваемых данных. Для стимуляции долговременного хранения данных, особенно если они запрашиваются редко, требуется награда за хранение.

В статье об Ethereum Swarm[10] описана система наград за хранение. Ноды заключают с владельцем информации контракт на хранение данных в течение какого-то времени. Оплата хранения может происходить в момент сохранения (обновления) данных или через некоторое время при условии, что данные действительно хранятся. В случае обнаружения потери данных в течение действия контракта, нода может быть оштрафована, для чего каждой ноде требуется первоначальная регистрация с гарантийным депозитом.

При сохранении данных нода возвращает квитанцию, которая служит доказательством, что нода приняла файл на хранение. Эта квитанция впоследствии позволяет проверить, хранятся ли всё ещё соответствующие им данные, и если нет - открыть судебный смарт контракт, который позволит наказать провинившуюся ноду.

В нашем случае данные не статичны, запись с одним и тем же ключом может перезаписываться несколько раз. В этом случае предъявленной квитанции может соответствовать не только оригинальная запись, но и более новая в соответствии с меткой времени запись с тем же ключом.

При инициированной пользователем операции удаления данных, вместо физического удаления данные заменяются специальной “нулевой” записью. Запись может быть физически удалена после истечения контракта хранения на неё.

Полнотекстовый поиск, вторичные индексы

В noSql базах данных быстрый поиск с задействованием небольшого количества нод возможен только по первичному ключу. Oasis.DDB в этом отношении немногим отличается от них. Между тем, поиск делового партнера по ключевым словам как минимум, а также группировка записей по каким-то критериям трудно достижима без вторичных индексов и полнотекстового поиска. Для полнотекстового поиска, а также использования вторичных индексов предлагается использовать решение, аналогичное Elassandra[11]. Это решение представляет собой локальные полнотекстовые индексы ElasticSearch[12] на каждой ноде распределенной noSql базы Cassandra. Полнотекстовые запросы рассылаются координатором всем нодам, затем смешиваются и возвращаются клиенту. Поскольку дополнительные индексы создаются локально и независимо на каждой ноде, дополнительное предотвращение проблемы византийских генералов не требуется.

Заключение

Таким образом, Oasis.DDB можно отнести в следующему поколению баз данных, удовлетворяющему указанным выше принципам:

---

[10] viktor trón et al. “Swap, swear and swindle incentive system for swarm”, http://swarm-gateways.net/bzz:/theswarm.eth/ethersphere/orange-papers/1/sw%5E3.pdf

[11] http://www.elassandra.io/

[12] https://www.elastic.co/products/elasticsearch

1. **Распределенность**
   Oasis.DDB поддерживает неограниченное число реплик, каждая из которых может быть координатором. То есть, обращаясь к одной из них, пользователь получает доступ ко всем данным
2. **Публичность**
   Oasis.DDB разработан для работы в публичной среде. Новые ноды могут добавляться к сети и брать на себя часть нагрузки в любой момент.
3. **Устойчивость к проблеме византийских генералов и другим типам атак в публичной сети**
   Учитывая, что все данные, помещенные в Oasis.DDB, подписываются их владельцем, ноды не могут по своему усмотрению подменить данные, как и не могут испортить данные при репликации на других нодах. Попытки подмены сразу же обнаруживаются, используя механизм электронной подписи. За попытку подмены нода-нарушитель может быть лишена регистрационного депозита и исключена из сети. Для размещения депозита, настройки прав доступа, механизмов взаиморасчетов между нодами используется внешний (для Oasis.DDB) блокчейн, который должен поддерживать тьюринг полные смарт контракты.
4. **Поддержка шардинга** (возможность репликации только части данных на каждой ноде, чтобы увеличить максимальный суммарный объем данных) Каждая нода Oasis.DDB отвечает за определенный интервал первичных ключей данных, которые она хранит. Уровень репликации (число нод, хранящих копии данных с одним и тем же первичным ключом) задаётся отдельно и может расти с ростом сети.
5. **Скорость**
   Принципы хранения данных позволяют предположить, что скорость записи и чтения данных в Oasis.DDB не будет сильно отличаться от текущих реализаций подобных баз данных, например, Apache Cassandra.
6. **Возможность хранить структурированные данные**
   Данные в Oasis.DDB поддерживают структуру. Это может быть JSON документ со структурой, удобной для конкретного приложения.
7. **Возможность удаления данных**
   Удаление данных поддерживается в Oasis.DDB. Гарантировать мгновенное удаление нельзя, но в конце концов при добропорядочном поведении нод данные будут удалены. Злонамеренная нода может сознательно сохранять все данные, которые удаляются. Однако она не сможет это сделать для всех данных, потому что ей направляются запросы только в определенном интервале первичных ключей.
8. **Язык запросов с возможностью осуществлять поиск не только по первичному ключу**
   С помощью ElasticSearch, аналогично методам интеграции с Cassandra в проекте Elassandra возможно расширение языка запросов вторичными ключами и полнотекстовым поиском.

Приложение

Обзор распределенных хранилищ данных

Для обоснования новизны технологии Oasis.DDB необходимо рассмотреть существующие подходы к распределенному хранению данных.

Блокчейн

Блокчейн уже сам по себе является распределенным хранилищем данных. Современные реализации, например, Ethereum, Etherus, позволяют хранить в блокчейне не только информацию о транзакциях, но и смарт контракты, а также любую произвольную информацию. На основе смарт контрактов могут быть построены распределенные приложения (например, dApps в Ethereum), которым также необходимо где-то хранить пользовательскую информацию. Большинство приложений хранятся все пользовательские данные непосредственно в блокчейне. Однако, этот подход обладает рядом существенных недостатков.

1. **Блокчейн неизменяем**. Всё, что сохранено в блокчейн, остаётся там навсегда и не может быть удалено. Это серьезный недостаток, учитывая, что большая часть информации при взаимодействии пользователей может быть временной и её можно было бы удалить, когда надобность в ее хранении отпадет. Постоянное хранение информации также работает против анонимности.
2. **Каждая нода является полной репликой других нод**. В результате при взрывном росте популярности приложения на блокчейне размер блокчейна стремительно растет на всех нодах одновременно. В какой-то момент размер блокчейна может превысить емкость серийно выпускаемых жестких дисков и для работы узлов потребуется специальное оборудование, которое могут себе позволить только большие компании, что ведет к опасной централизации.
3. **Скорость транзакций в блокчейне очень низкая**. Из-за необходимости консенсуса между всеми узлами сети скорость транзакций весьма невелика. Увеличение размера блока и частоты генерации блоков позволяют не сильно повысить скорость транзакций, потому что в этом случае скорость транзакций упирается в скорость распространения блоков по сети.

Все эти проблемы представляют серьезную проблему для хранения данных в существующих реализациях блокчейна, особенно для реализаций, поддерживающих смарт контракты и распределенные приложения dApps.

Таким образом, блокчейн нецелесообразно использовать в качестве хранилища больших объемов нефинансовых данных.

IPFS

IPFS[13] (InterPlanetary File System) - технология распределенной файловой системы, основанная на DHT[14] (Distributed Hash Table) и протоколе BitTorrent[15]. Она

[13] https://ipfs.io/
[14] https://en.wikipedia.org/wiki/Distributed_hash_table
[15] https://en.wikipedia.org/wiki/BitTorrent

позволяет объединить файловые системы на различных устройствах в одну, используя контентную адресацию.

Достоинства:

1. Каждое устройство хранит только те файлы, которые ему нужны, плюс метаинформацию по расположению файлов на других устройствах. Поэтому не требуется доп. мотивация за хранение файлов.
2. Нет необходимости доверять пирам, потому что адресация файлов осуществляется по содержимому.
3. Устойчивость к флуду (загрузке ненужных файлов в сеть), потому что файлы размещаются только на собственное устройство.
4. Высокая пропускная способность (благодаря BitTorrent)

Недостатки:

1. Хранение только файлов (неструктурированной информации).
2. После размещения файла нельзя выходить из сети, пока он по ней не разойдется.
3. Хранение данных другими устройствами не гарантировано, для гарантированного предоставления своего файла другим нужно быть онлайн
4. Файлы статичны (неизменяемы)
5. Удаление файла в принципе не предусмотрено.

С использованием IPFS строится социальная сеть AKASHA[16] (Ethereum + IPFS) и торговая площадка OpenBazaar[17]. Они в полной мере наследуют указанные выше недостатки IPFS, основной из которых - нельзя разместить информацию в сети и выйти оффлайн, пока она не распространилась по пирам.

Таким образом, указанные недостатки IPFS не позволяют эффективно использовать эту технологию в качестве замены БД.

Распределенные файловые хранилища

Такие хранилища позволяют объединять отдельные устройства в общее облачное хранилище. В результате пользователи могут хранить там свои файлы так же, как они это могли бы делать в классическом централизованном хранилище, например, Dropbox[18], но дешевле. Владельцы устройств (“фермеры”), предоставляя место для хранения чужих файлов, получают за это деньги от пользователей соответственно своему вкладу. Чтобы измерить вклад, обеспечить надежность хранения и пресечь злоупотребления используются различные проверки, например, proof of storage (доказательство принятия файла), proof of retrievability (доказательство, что файл в наличии и может быть извлечен), основанные на криптографии. За успешное прохождение проверки пользователь платит, а фермер получает некоторую сумму в криптовалюте.

Строятся такие проекты, в основном, с использованием технологии DHT и контентной адресации. Некоторые дополнительно используют blockchain и смарт контракты.

[16] https://akasha.world/
[17] https://openbazaar.org/
[18] https://www.dropbox.com/

Таких проектов на рынке на текущий момент довольно много, например, Sia[19], Storj[20], Ethereum Swarm[21], MadeSAFE[22]. Они все построены по схожим принципам. Причем Ethereum Swarm задумывался в том числе и для обеспечения удобного хранилища файлов для dApps.

Достоинства:

1. Файлы хранятся в облаке и доступны независимо от доступности их владельца.
2. Высокая пропускная способность.
3. За счет финансовой мотивации обеспечивается надежность хранения и извлечения файлов.
4. Удаление ненужных файлов возможно

Недостатки:

1. Хранение только файлов (а не структурированной информации)
2. Файлы статичны
3. Хранение небесплатно

Для хранения файлов распределенные файловые хранилища выглядят привлекательно. Однако для хранения структурированной динамической информации, например, пользовательских данных социальной сети, хранение данных в статических файлах представляет собой серьезную проблему. Дело в том, что файловые хранилища ничего не знают о содержимом файла, а в нашем случае требуется искать информацию не только по идентификатору (или имени) файла, но и по его содержимому. Например, найти всех пользователей с ключевым словом blockchain. Или находящихся в определенном городе. Или даже осуществлять полноценный поиск по публикациям. В результате, распределенные файловые хранилища также неэффективны в качестве замены БД.

Распределенные базы данных

К сожалению, в силу теоремы CAP[23] нельзя получить полностью распределенную базу данных, которая одновременно обеспечивает согласованность, доступность и устойчивость к разделению.

В нашем случае требуется именно распределенная база данных, разумеется, устойчивая к разделению и доступная - нам необходимо быстро получать ответ из неё, хотя, возможно, и не самый свежий. Это ограничивает наш выбор рядом NoSQL баз данных, потому что ACID[24] SQL СУБД в первую очередь обеспечивают согласованность.

Реализаций распределенных NoSQL баз данных великое множество. Например, MongoDB[25], Cassandra[26], RethinkDB[27]. Все они способны работать с большим количеством реплик, объединяющихся в кластер. Клиент работает с одной из реплик, а данные автоматически синхронизируются с остальными. Для балансировки нагрузки

[19] http://sia.tech/
[20] https://storj.io/
[21] https://github.com/ethersphere/swarm
[22] https://maidsafe.net/
[23] https://en.wikipedia.org/wiki/CAP_theorem
[24] https://en.wikipedia.org/wiki/ACID
[25] https://www.mongodb.com/
[26] http://cassandra.apache.org/
[27] https://www.rethinkdb.com/

может использоваться шардинг, когда часть данных хранится только на части реплик. Добавление в кластер новой реплики практически линейно масштабирует кластер, причем некоторые реализации (например, Cassandra) позволяют реплике автоматически принять на себя часть работы кластера.

NoSQL базы данных обеспечивают “целостность в конечном итоге” (eventual consistency), то есть, данные становятся согласованы через некоторое время, когда отдельные реплики синхронизируются. В этом они похожи на блокчейн - подтверждение транзакции тем вероятнее, чем больше времени прошло.

NoSQL базы данных могут хранить как просто ключ-значение, так и поддерживать внутреннюю структуру значения, а также дополнительные индексы. Наиболее продвинутые имеют также базовую поддержку транзакций и SQL-подобный язык запросов (например, Cassandra).

По всему вышесказанному этот класс баз данных может показаться идеальным для использования в блокчейне. Но представьте, что в слаженный кластер такой баз данных кто-то добавил злонамеренную реплику, которая начинает сообщать другим репликам в кластере, что все данные необходимо удалить! Все остальные реплики послушно все данные удалят, и база данных будет безнадежно испорчена. То есть, такая слаженная работа реплик возможна сейчас только в доверенной среде (кластер таких баз данных неустойчив к проблеме византийских генералов). Если в кластер поместить злонамеренно работающую реплику, она может вызвать уничтожение данных всего кластера.

Преимущества

1. Высокая скорость
2. Линейное масштабирование скорости и размера хранилища
3. Устойчивость к недоступности отдельных реплик
4. Зрелые реализации

Недостатки

1. Централизация
2. Необходимость доверять пирам - неустойчивость к проблеме византийских генералов.

Распределенные базы данных нельзя использовать в децентрализованном проекте в силу их централизованной природы.

BigChainDB

Существует реализация блокчейна под названием BigChainDB[28] или, как она ещё называется, IPDB (InterPlanetary DataBase). Авторы заявляют очень высокую скорость транзакций (1 млн/сек), огромную емкость хранилища (за счет распределенного хранения с частичной репликацией). BigChainDB получает эти преимущества за счет упрощенного консенсуса при формировании блоков, а также за счет хранения всех блоков и транзакций в существующей реализации noSql базе данных - RethinkDB или MongoDB.

К сожалению, подобная архитектура имеет существенный недостаток - каждый узел имеет полные права на запись в общее хранилище данных, а значит, система в целом неустойчива к проблеме византийских генералов. Авторы этого проекта знают

[28] https://www.bigchaindb.com/

об этом, обещая подумать об этом позже[29]. Однако исправление фундаментальных проблем в базовой архитектуре после выпуска продукта весьма трудоёмко и чаще всего невозможно, потому что может привести к существенно другому продукту с другой архитектурой. Такое легкое отношение к фундаментальной проблеме вызывает критику проекта со стороны сообщества[30], потому что демонстрируемые высокие скоростные и объёмные характеристики BigChainDB в условиях отсутствия BFT (Byzantine Fault Tolerance) не так уж и отличаются от характеристик, демонстрируемых изначально noSql базами данных RethinkDB и MongoDB, которые используются ею для хранения данных. Но раз уж всё равно требуется полное доверие между узлами, то почему не пользоваться указанными базами данных напрямую?

Таким образом, реальное использование BigChainDB ограничивается приватными сетями. Чтобы не вводить людей в заблуждение, BigChainDB стоило бы назвать BigPrivateBlockChain, тогда никаких вопросов не возникало бы. Для публичных сетей она совершенно не подходит.

Достоинства:

1. Скорость и хранилище, сопоставимое с распределенными noSql базами данных

Недостатки:

1. По сути, это и есть обычная noSql база данных, дополненная недостатками блокчейна
2. Неизменяемость (данные нельзя удалить легально, но можно злонамеренно)
3. Неустойчивость к проблеме византийских генералов, следовательно, невозможность использования в публичной сети

Таким образом, BigChainDB также не подходит для децентрализованного хранения данных.

[29] https://docs.bigchaindb.com/en/latest/bft.html и https://github.com/bigchaindb/bigchaindb/issues/293

[30] https://reddit.com/r/Bitcoin/comments/4j7wjf/bigchaindb_a_prime_example_of_blockchain_bullshit/