White Paper

(Информация может быть добавлена/изменена)

Децентрализованная финансовая система

CREDITS

Version 1.6/26 nov 2017

Оглавление

Абстракт 3
Введение 3

1. Реестр сети 4
Определения 4
Узлы сети 4
Последний сохраненный блок 4
Синхронизация узлов 5

2. Консенсус сети 5
Сравнение консенсусов 5
Понятие главного узла сети 6
Оборудование узлов сети 7
Достижение консенсуса 7
Формирование и инициация реестра 8
Транзакции, не вошедшие в реестр 8

3. Обработка транзакций 8
Транзакции 8
Нахождения консенсуса 8
Обработка транзакций 9
Структура записей реестра 9
Структура реестра CREDITS 9
Размер блока 9
Поиск участников сделки 10
Канал передачи данных 10
Действие в системе 11
Добавление транзакции на валидацию 11
Стоимость транзакций 12

4. Смарт-контракты 12
Введение 12
Сущности 12
Метод смарт-контракта 13
Виртуальная исполняемая машина 13
Стоимостное выражение 13
Выполнения условий смарт контракта 13
Источники данных 14

5. План реализации 14
Технический план реализации проекта 14
Криптовалюта CREDITS 15

Абстракт

Платформа CREDITS - это децентрализованная финансовая система для прямого взаимодействия между участниками на принципах peer-to-peer (P2P — равный к равному). Платформа расширяет потенциал использования финансовых услуг на базе распределенного реестра, самоисполняемых смарт контрактов и криптовалюты CREDITS. Система призвана объединить всех участников в одном месте, предоставляя им платформу для создания и использования финансовых услуг, где каждый может, как предложить услугу, так и воспользоваться ей. Благодаря четко определенной и сбалансированной технологической системе платформа CREDITS способна предложить новое техническое решение и новую концептуальную модель взаимодействия участников сети для развития современных децентрализованных финансовых услуг.

Введение

Полностью одноранговое устройство систем предоставления услуг позволяет формировать финансовые услуги: денежные переводы, обмены валют и ценностей, кредитование, фондирование и другие услуги, напрямую между участниками. Все осуществляется без дополнительных посредников, по принципу - один из равных участников - другим участникам системы. В итоге все получают услуги дешевле, быстрее и качественнее.

Мир движется в сторону прямого взаимодействия между людьми на принципах peer-to-peer – равный к равному. Произошла революция! Это наглядно видно по перевороту в средствах-массовой информации: газеты, журналы и ТВ до 1990-х годов являлись основными поставщиками информации. Сегодня лидерами мнений являются блогеры, каналы Youtube и социальные сети, деньги вкладывают в краудфандинг и ICO, а информацию хранят в децентрализованных облачных системах.

Финансовая отрасль, пожалуй, одна из немногих, которая отстает, сопротивляется внедрению децентрализации и прямого взаимодействия между участниками. Хотя технически создать систему децентрализованных финансовых услуг гораздо легче, чем беспилотные автомобили.

Но, для создания системы децентрализованных финансовых продуктов и услуг на базе распределенных реестров необходима соответствующая технологическая среда:

1. Высокая скорость исполнения (в секундах) вместе со способностью обрабатывать одновременно огромное количество транзакций (сотни тысяч в секунду) при не большой стоимости каждой транзакции (для микроплатежей и не денежных операций).

2. Создание системы, в которой объединяться все участники и элементы, необходимые для существования качественных финансовых децентрализованных услуг: персонализация пользователей, KYC, бюро кредитных историй, расчетные центры фиатных денег, вывод и обналичивание криптовалюты и прочее.

Это две большие и основные задачи, которые в настоящий момент препятствуют развитию peer-to-peer финансовых продуктов.

Представляем вам решение данных задач, которое мы осуществили с помощью финансовой системой CREDITS.

Единая технологическая децентрализованная платформа CREDITS способна объединить всех участников финансовых услуг, безопасно и быстро исполнять все транзакции работающие на принципах распределенного реестра. Самоисполняемые смарт контракты и принципы федеративной системы голосования дают неограниченные возможности для всех участников создания уникальных взаимодействий разных финансовых продуктов. Платформа открывает новый огромный рынок и новый потенциал использования блокчейн проектов и сервисов в

финансовой и других отраслях, ранее которые не могли использоватся в силу ограничений скорости и цены транзакций.

1. Реестр сети

Определения

1. Система - совокупность децентрализованных узлов сети осуществляющих обработку, сохранение транзакций, выполнение и подтверждение условий смарт контрактов, обработку запросов от сторонних систем, предоставление информационных данных при запросе.

2. Узел сети - компьютер, на котором установлен полный клиент сети, соединенный с общей системой, проверяющий транзакции и записывающий их в реестр.

3. Реестр - перечень транзакций, подтвержденных системой и сохраненный на всех узлах сети.

4. Транзакция - единица системы, обозначающая запрос на выполнение метода смарт контракта или любое действие в сети и запись результатов в блокчейн

5. Смарт контракт - единица системы, компьютерные протоколы, которые облегчают, проверяют или обеспечивают соблюдение условий взаимодействия. Они обычно имеют пользовательский интерфейс и часто эмулируют логику договорных отношений. Определяющее свойство умного контракта — это его децентрализованность и независимость от центрального источника.

6. Метод смарт контракта - программный код, отвечающий за расчет результата работы условий смарт контракта и запись его в реестр.

7. Сторона контракта - конечный участник сети и пользователь системы.

Узлы сети

Для построения децентрализованной сети, основанной на свободном доступе и подключении узла, мы используем несколько типов узлов в зависимости от их назначения:

1. Обычный узел - узел, участвующий в проверке транзакции на валидность, но при этом имеет минимальный коэффициент доверия, он же является кандидатом на получение роли доверенного узла и узла текущей обработки при следующем цикле выбора роли узлов в сети.

2. Доверенный узел - узел, участвующий в проверке транзакции и имеющий максимальный коэффициент доверия (единицу), является кандидатом на роль узла текущей обработки и обычного узла. Узел не может стать доверенным в течение математически рассчитанного количества циклов выбора и голосования среди узлов. Математический расчет зависит от количества узлов и сложности сети.

3. Главный узел сети – узел, участвующий в проверке и отвечающий за добавление в блок реестра транзакций. Данный узел не может стать доверенным или узлом текущей обработки в течение математически рассчитанного количества циклов голосования, математический расчет которого зависит от количества узлов и сложности сети.

В системе используется коэффициент доверия - абсолютное долевое числовое значение от 0 до 1, выраженное в математическом отношении количества доверенных узлов + 1 к общему количеству узлов в сети. Максимальное количество доверенных узлов не может превышать 50% узлов сети.

Последний сохраненный блок

Общий реестр блоков (ОРБ) - это синхронизированное состояние всего общего реестра блоков на всех узлах системы.

Под содержанием блока реестра мы понимаем единицу хранящейся информации, которая содержит хэш-код предыдущего блока и перечень данных относящихся к настоящему реестру со связанным номером предыдущего блока. После получения блока от другого узла, он в соответствии с номером занимает свое место в общем реестре блоков. Тем самым экономится пропускная способность сети.

В процессе синхронизации сначала проверяется только номер блока. Если блок отсутствует на данном узле, то он скачивается и сохраняется.

В результате в любой момент система содержит последнюю актуальную копию реестра. Мы даем ей название – последний реестр ПР. Он автоматически создается узлом, отвечающим за формирование реестра после нахождения консенсуса. Этот блок отправляется всем узлам системы, для того чтобы на всех узлах системы поддерживалось актуальное единообразие состояния реестра.

Каждый узел связан со всеми остальными узлами в сети и постоянно обменивается с ними новыми блоками с транзакциями, для того чтобы везде была актуальная информация. Все блоки формируют набор транзакций в кандидаты ждущих добавления в реестр. В это же время каждый сервер формирует предлагаемые наборы в кандидаты для других серверов и предложенного набора транзакций. После проверки принимается решение – добавлять ли их в реестр.

В итоге достигается многократное сохранение данных реестра на множестве серверов – узлов системы, вся информация находится под защитой. Чем больше узлов в системе, тем она надежнее и независимей.

Синхронизация узлов

Каждый новый узел запускается и синхронизируется после определения и тщательной проверки на доверие. Для повышения скорости обработки информации, все процессы обрабатываются одновременно, независимо друг от друга. Если входящих переменных нет, то создается пустое хранилище реестра – резервируется пространство в оперативной памяти для дальнейшего упрощенного доступа к ней. В случае, если искомого реестра нет, на доверенные узлы отправляется запрос на получение всех транзакций, которые были произведены для синхронизированного аккаунта.

Если в качестве входящего параметра указывается объект, характеризующий транзакцию, то запускается поиск по всем запущенным потокам синхронизации. Результатом операции является цифровой код – номер позиции в реестре доверенного узла для текущего потока или номер ошибки, если значение меньше нуля. Если метод потока завершается ошибкой соединения, то поток завершается полностью.

2. Консенсус сети

Консенсус CREDITS - это метод группового принятия решений, целью которого является приход к выработке окончательных решений, приемлемым для всех узлов сети.

Сравнение консенсусов

Для сравнения различных видов консенсуса определим принципы децентрализованного реестра CREDITS:

- доступность реестра (узлы могут записывать данные в реестр и читать их из него в любой момент времени);
- модифицируемость всеми участвующими узлами в сети;
- согласованность всех узлов системы (все узлы видят абсолютно одинаковую версию реестра, которая обновляется после изменений);
- устойчивость к разделению (если один узел становится неработоспособным, это никак не отражается на работе всего реестра).

Сравниваемый параметр	Credits specific DPOS	PoW	PoS
Принцип выявления узла, создавшего блок.	Расчет математической функции. Подтверждение хранения последней копии реестра.	Выполнение итеративного расчета математической функции, с изменяющейся сложностью.	Поиск среди участников (конкурирующих узлов) максимального стека.
Атака 51%.	Маловероятна, так как необходимо иметь полный реестр на ресурсах и вычислительную мощность для расчета, а выбор доверенных узлов происходит динамически.	Вероятна, но очень дорого обойдется по использованию ресурсов.	Вероятна, но дорого стоит, из-за необходимости увеличить свой собственный стек.
Компенсация за проделанную работу на узле для добавления в реестр/блокчейн.	Рассчитывается автоматически, зависит от комиссии за операцию.	Предоставляется фиксировано за майнинг блока.	Предоставляется фиксировано за майнинг блока.

Понятие главного узла сети

Все узлы сети децентрализованы и ни один из них не имеет приоритета перед другими узлами. При этом необходимо определить узел сети, который будет производить операцию обработки очереди транзакций, хранящихся на разных узлах сети. После этого он должен вносить новый сформированный блок транзакций в реестр.

Для увеличения скорости обработки транзакций, полной безопасности хранения данных, обработки и передачи транзакций, платформа CREDITS использует свой собственный комбинированный протокол. Протокол основан на расчете математической функции всех транзакций реестра, применяя принципы Proof of Work. Он точно определяет хранение последней актуальной копии реестра и программного обеспечения на данном узле (Proof of Capacity), вычисляя контрольную сумму значений всего содержимого – хеш-кода. Также определяется размер файлов, доказательство – что это последняя, актуальная копия и хеш-код последней, записанной в систему транзакции.

Чтобы стать главным узлом сети, узел осуществляет поиск значения хеш-функции, которую рассчитывает на основании последнего сохраненного реестра. Мы организовываем

между узлами сети здоровую конкурентную среду, за возможность стать главным, сформировать и сохранить новый реестр.

После расчета функции и получения результата, он отправляется всем узлам сети для проверки. Результат содержит временную метку даты расчета и значение на основе расчета функции файлов реестра и программного обеспечения. Все узлы получают рассчитанное значение, сравнивают время расчета, выделенное на поиск главного сервера сети, проверяют его и подтверждают доверенность узла, а также подтверждают его возможность участия в конкурсе – стать главным узлом сети.

После получения одобрения от всех узлов сети, формируется список из тех, кто правильно рассчитал значение функции и содержит метку времени. Узел, получивший правильный результат и одобривший его со всеми узлами быстрее всех, становится главным узлом сети в данный момент.

Для расчета хеш-суммы файла используется концепция алгоритма SHA-2.

Хеш-функции семейства SHA-2 построены на основе структуры Меркла — Дамгарда.

Исходное сообщение после дополнения разбивается на блоки, каждый блок — на 16 слов. Алгоритм пропускает каждый блок сообщения через цикл с 64 или 80 итерациями (раундами). На каждой итерации 2 слова преобразуются, а функцию преобразования задают остальные слова. Результаты обработки каждого блока складываются, сумма является значением хеш-функции. Тем не менее, инициализация внутреннего состояния производится по результатам обработки предыдущего блока. Поэтому независимо обрабатывать блоки и складывать результаты нельзя.

Оборудование узлов сети

Мы стремимся создать платформу с максимально быстрыми характеристиками обработки транзакций, поэтому предлагаем использовать материальное стимулирование поддержания узлов сети в наилучших условиях: мощное серверное оборудование и высокая пропускная способность сети интернет.

В качестве материальной компенсации владелец главного узла сети получит ½ вознаграждения в валюте CREDITS от суммы комиссий транзакций данного обработанного реестра. Оставшаяся часть ½ распределяется между доверенными узлами (узлы участвующие в решении BFT консенсуса). Величина процентного соотношения может быть изменена, а также вынесена в систему формирование ставок путем федеративного голосования узлами сети, после проведения первичного размещения монет после проведения первичного размещения монет в течение не менее трех лет.

В итоге мы стимулируем владельцев серверов - содержать данный сервер на наиболее мощном оборудовании и поддерживать качественный, скоростной канал связи.

Достижение консенсуса

В итоге мы имеем главный узел сети, выбранный всеми узлами. Основными задачами главного узла являются: получение транзакций в статусе кандидата на добавление в реестр от всех узлов, их обработка, формирование последнего актуального реестра и отправка всем узлам сети вновь сформированного реестра. Процесс обработки транзакции и формирование последнего актуального реестра как раз и является поиском решения консенсуса. Результат формирования последнего актуального реестра - решение консенсуса.

Весь процесс можно разделить на следующие этапы:

1. Поиск главного узла сети;
2. Формирование списка доверенных узлов;
3. Получение списка транзакций и формирование списка кандидатов на добавление в реестр;

4. Обработка списка кандидатов, голосование узлов (доверенные и обычные узлы имеют разные весовые коэффициенты (доверительный коэффициент);
5. Удаление из списка кандидатов не подтвержденных транзакций не прошедших проверку или по которым пришло отрицательное подтверждение;
6. Формирование списка подтвержденных транзакций для добавления в реестр;
7. Добавление транзакций в реестр с меткой времени и хеш-кодом блока, который содержал транзакцию;
8. Отправление блока с транзакциями всем узлам сети, при получении он добавляется в реестры на всех узлах.

Формирование и инициация реестра

Весь процесс можно описать в следующей последовательности:
1. Конечный участник сети в системе, формирует транзакцию.
2. При выполнении всех условий смарт-контракта, указанных в нем, пользователь инициирует действие (транзакцию) через вызов нужного метода, используя программное обеспечение платформы.
3. Для следования основополагающим принципам блокчейна, ядро валидаторов следит за синхронизацией и неизменностью последней версии реестра.
4. В момент решения консенсуса, все транзакции, поступившие во время цикла, собираются в блок.
5. Блоку присваивается номер, состоящий из временной метки и идентификатора узла, преобразованного в хеш-код, после чего блок помещается в модуль нахождения консенсуса.
6. После составления белого списка кандидатов, в реестр записывается не только хеш транзакции, но и хеш блока, на основании которого всегда можно удостоверить источник.
7. Данный хеш является своего рода подписью блока и того, кто этот блок с транзакциями создал.
8. После нахождения консенсуса с помощью федеративного алгоритма поиска, транзакции, добавленные в блок, передаются в ядро валидатора на запись в реестр.

Транзакции, не вошедшие в реестр

Транзакции, не вошедшие в список готовых транзакций, помечаются как отклоненные. Информация об этом сразу же выводится у отправителя (инициатора) транзакции.

Транзакции, не включенные в реестр, остаются в наборе кандидатов и хранятся в узлах сети. Туда же попадают и все новые транзакции, полученные сервером во время консенсуса, а затем процесс поиска начинается заново. Такая непрерывная цикличная работа сети позволяет проводить транзакции за достаточно небольшой промежуток времени с сохранением высокой степени достоверности и актуальности информации.

3. Обработка транзакций

Транзакции

Транзакция - это минимальная единица системы, информирующая платформу о выполнении методов контракта или прямых переводов между аккаунтами без создания смарт контракта, с последующим размещением результата выполнения в пиринговой сети.

Нахождения консенсуса

В системе используется федеративная модель поиска решения консенсуса - голосование доверенных узлов-валидаторов. А также алгоритм поиска решения консенсуса - алгоритм прохода конечного автомата. Консенсус работает по циклам (тактам), на каждый такт из сети происходит извлечение транзакций и помещение их в пул (одномерный массив). После помещения в пул, все транзакции отправляются по доверенным узлам с целью получения ответа. Если ответ получен, то транзакция, по которой был отправлен запрос на добавление может быть добавлена в реестр данного валидатора. После этого она отправляется следующим валидаторам в сети. При достижении консенсуса - конечной точки цепи и полного подтверждения легальности перевода, транзакция передается на валидацию с пометкой на запись и сохранение в реестр.

Обработка транзакций

Для достижения децентрализованного характера системы каждый сервер должен обладать не только хранилищем реестра, но и быть полноценным обработчиком всех транзакций.

В системе используется понятие - ядра системы. По ядрам мы понимаем обработчика данных, который выполняет определенную производственную задачу, вне зависимости от доступности и работоспособности остальных составляющих системы. Каждое ядро на входе в момент выполнения задачи получает перечень переменных для обработки. А на выходе всегда получает результат – положительный, любой другой или ошибку. В результате выполнения ядро системы всегда содержит код ответа, помимо основного набора данных. Данная структура необходима для максимально высокой скорости каждого из процессов, которые должны работать независимо друг от друга.

Структура записей реестра

Для достижения существенной производительности работы реестра, но в тоже время без ущерба безопасности, мы предлагаем использовать базу данных реестра без формирования дерева Меркла из хеш-кода предыдущего блока и результата транзакции.

Дерево Меркла (древовидное хеширование) (TTH - Tiger Tree Hashing) — тип хеш-функции, использующийся для того, чтобы проверять целостность данных, получать уникальный идентификатор цепочки, а также восстанавливать последовательность. Данные делятся на маленькие части — блоки, которые индивидуально хешируются при помощи Leaf Tiger Hash, затем из каждой пары хешей, поочерёдно вычисляется Internal Tiger Hash. Если хешу нет пары, то он переносится в новую цепочку без изменений. Далее в цепочке для каждой пары снова вычисляется Internal Tiger Hash. Эта процедура повторяется до тех пор, пока не останется один хеш.

При работе реестра с использованием деревьев Меркла скорость обработки транзакций очень низкая, а нагрузка на вычислительные ресурсы – очень высокая. На наш взгляд, это не рациональное использование хранилища данных.

Структура реестра CREDITS

Мы предлагаем отказаться от деревьев Меркла и использовать в системе CREDITS реестр транзакций, каждая запись которого состоит из хеш-кода блока транзакций для добавления в список кандидатов добавления в реестр. Также в записи указывается идентификатор узла и временная метка формирования. Запись реестра содержит направление транзакции, ее

начальный и конечный аккаунты, тип списания, количество единиц списания, тип зачисления и количество единиц зачисления. Данный принцип позволяет увеличить скорость обработки транзакций, повысить сложность нелегитимного изменения реестра, исключить возможность изменения записи в реестре задним числом.

Размер блока

Единицей времени принимается цикл поиска главного и доверенных узлов, а время длительности цикла рассчитывается в зависимости от сложности сети. За единицу времени, в сети находится N транзакций, сформированных и переданных на обработку в сеть с момента завершения предыдущего цикла, до момента начала следующего цикла, для получения статуса “Кандидат на добавление в реестр”. Выбранные из сети N транзакций, помещаются в блок. Размер блока зависит от количества транзакций в нем.

Поиск участников сделки

Пиринговую сеть CREDITS можно представить в виде графа, с аккаунтами пользователей в виде вершин и множеством возможных транзакций в виде направленных ребер, соединяющие две вершины (аккаунта). Так как все ребра имеют начальную и конечную вершины, значит всегда можно построить ориентированный граф (орграф).

Если принять за тождество следующие условия:

- любая транзакция всегда имеет отправителя и получателя;
- любая вершина (аккаунт) всегда может быть соединена с другой вершиной (аккаунтом) с помощью направленного ребра (транзакции);
- любая вершина графа (аккаунт) имеет конечное количество направленных ребер (входящих и исходящих транзакций).

В связи с вышесказанным, мы можем сказать, что орграф содержит искомый маршрут выполнения необходимых условий транзакции и построить простую цепь. Так как представляет собой конечную последовательность вершин, в которой каждая вершина (кроме последней) соединена со следующей в последовательности вершиной ребром.

А простая цепь - маршрут в орграфе без повторяющихся вершин.

Так как заранее граф не известен, то исходя из теории графов - маршрут придется строить по неизвестному ориентированному графу. Как известно, на классе таких графов длина обхода $\Theta(nm)$, где n – число вершин, а m – число ребер графа. Для любого графа существует обход длиной˘ $O(nm)$ и существуют графы с минимальной˘ длиной˘ обхода $\Omega(nm)$.

Обход неизвестного графа означает, что его топология заранее неизвестна, и мы узнаем ее только в процессе движения по графу. В каждой˘ вершине видно, какие ребра из нее исходят, но в какую вершину ведет ребро можно узнать, только пройдя по ней˘. Это аналогично задаче обхода лабиринта роботом, находящемся внутри него и не имеющем плана лабиринта. Если число состояний ограничено, то робот – это конечный˘ автомат. Такой робот является аналогом машины Тьюринга: лента заменена графом, а ее ячейки привязаны к вершинам и рёбрам графа.

Канал передачи данных

Каждый канал связи между главным узлом сети и обычным узлом сети CREDITS является отдельным потоком (multithreading), внутри которого данные при выполнении транзакции отправляются в зашифрованном виде.

Для обеспечения безопасности сети, все данные между узлами-валидаторами передаются в зашифрованном виде, а каждый коннект между узлами обеспечивается низкоуровневым соединением на основании сетевой библиотеки. Если передача данных происходит с ошибкой,

то поток автоматически должен быть прерван, соответствующая запись помещается для записи в систему логирования, а затем и в лог-файл. Данные передаются через типизированные переменные. Шифрование передаваемых данных происходит по симметричному алгоритму RC4. Так как работа данного алгоритма осуществляется по общему секретному ключу, то передача этого ключа происходит при создании соединения между узлами и передается в шифрованном виде в соответствии с алгоритмом Диффи-Хеллмана.

Алгоритм RC4, как и любой потоковый шифр, строится на основе генератора псевдослучайных битов. На вход генератора записывается ключ, а на выходе читаются псевдослучайные биты. Длина ключа может составлять от 40 до 2048 бит. Генерируемые биты имеют равномерное распределение.

Алгоритм Диффи — Хеллмана позволяет двум сторонам получить общий секретный ключ, используя незащищенный от прослушивания, но защищённый от подмены канал связи. Полученный ключ можно использовать для обмена сообщениями с помощью симметричного шифрования. Алгоритм основан на трудности вычислений дискретных логарифмов. В нем, как и во многих других алгоритмах с открытым ключом, вычисления производятся по модулю некоторого большого простого числа P.

Вначале специальным образом подбирается некоторое натуральное число A, меньшее P. Если мы хотим зашифровать значение X, то вычисляем

$Y = A^X \bmod P$.

Причем, имея X, вычислить Y легко. Обратная задача вычисления X из Y является достаточно сложной. Экспонента X как раз и называется дискретным логарифмом Y. Таким образом, зная о сложности вычисления дискретного логарифма, число Y можно открыто передавать по любому каналу связи, так как при большом модуле P исходное значение X подобрать будет практически невозможно. На этом математическом факте основан алгоритм Диффи-Хеллмана для формирования ключа.

Любые действия в системе привязываются к метке времени, номеру предыдущего блока, логину пользователя и идентификатору смарт контракта. Это позволяет найти дубли при выполнении. Если дубль найден, то брать первую транзакцию из пула, остальные считать нелегитимными.

Действие в системе

Действие в системе - это транзакция, характеризующая простейший перевод или трансфер ценности от аккаунта к аккаунту или передачу результата выполнения метода контракта валидатору, для последующего поиска решения в подсистеме поиска консенсуса.

Чтобы не допустить дублирования транзакции в одном блоке с одинаковым идентификатором, в системе принято соглашение, что единственной верной и правильной считается та транзакция, которая пришла первая на обработку в подсистему валидатора. Так как в системе валидатора уже записано, что с текущего аккаунта уже была совершена транзакция и на аккаунте не осталось ценностей для проведения транзакции, то консенсус найден быть не может. Таким образом, проблема двойной траты решена.

При выполнении транзакции, поступлении информации к валидатору и ее подтверждении, информация об изменении состояния реестра автоматически рассылается по всем узлам из доверенного списка, после чего реестр синхронизируется.

Для того чтобы всегда иметь актуальный реестр транзакции среди всех доверенных узлов для текущего узла-валидатора, необходимо каждый раз проводить синхронизацию вновь поступившей транзакции в реестре всех узлов. Для решения этой задачи необходимо использовать отдельный порт для синхронизации (если есть такая возможность). Данная возможность позволит увеличить скорость обработки входящей информации в ядро валидатора за счет распределения нагрузки на порт. Поток синхронизации выполняется всегда, он цикличен. Приоритет по распределению оперативной памяти и процессорной нагрузки (использование

тактов процессора) ниже среднего. В памяти хранятся последние 1000 операций и состояние аккаунтов по ним (в зашифрованном виде по синхронному алгоритму) это позволяет повысить скорость предоставления ответа на запрос от других узлов-валидаторов.

Добавление транзакции на валидацию

Добавление транзакций в реестр вызывается только из подсистемы валидатора сразу после нахождения консенсуса и составления белого списка с результатом сохранения транзакций в реестре. Вызов из сторонних систем невозможен, с целью повышения безопасности.

Входящие параметры - объект, характеризующий транзакцию. Результирующий параметр - ResultValue<0 - выполнение прервано с ошибкой, результирующее значение - код возможной ошибки/0 < ResultValue - выполнение функции произошло без ошибок, результат - номер записи в реестре.

Входящий параметр - объект, содержащий уникальную метку транзакции, отправителя, адресата, передаваемую ценность, стоимостное соответствие, искомую ценность, количество передаваемой ценности, количество искомой ценности и другую системную информацию, изменение которой допускается при необходимости.

Стоимость транзакций

В системе используется денежная единица CREDITS, которая служит:

- внутренним платежным средством за пользование системой;
- для операций по обмену разных валют внутри системы;
- для операций по обмену различных ценностей внутри системы;
- для создания и обработки операций по смарт-контрактам;
- для покупки информации из сторонних источников для сервисов внутри системы.

Стоимость транзакции может меняться в зависимости от нагрузки сети, от конкретного пользователя системы, который теоретически может направить огромный поток транзакций в определенный пиковый момент времени. Мы предлагаем использовать материальный способ и воздействие на пользователей системы в качестве управления нагрузкой на сеть.

Стоимость выполнения транзакций в первые три года работы системы будет устанавливаться индивидуально для разного типа транзакций и операций. В последующем будет разработан алгоритм автоматического формирования цены транзакции.

4. Смарт-контракты

Введение

Смарт-контракт в системе CREDITS - это электронный алгоритм, описывающий набор условий, с помощью которого можно связывать действия и события в реальном мире или цифровых системах.

Для реализации самоуправляемых смарт-контрактов требуется децентрализованная среда, полностью исключающая человеческий фактор, а для возможности использования в умном контракте передачи стоимости требуется независимая от центрального органа криптовалюта.

Сущности

Смарт-контракт в CREDITS состоит из следующих сущностей:

1. Свойство (public переменные) - сущность системы, позволяющая хранить публичные данные, необходимые для работы контракта в системе CREDITS.

2. Метод - сущность системы CREDITS, отвечающая за соблюдение логики и последовательности действий при проведении транзакции (действий по контракту).

Участники системы CREDITS подписывают смарт-контракты, используя используя вызов методов, изменяющих свойства контракта, запуская процессы проверки выполнения условий и согласования..

Смарт-контракт после подписания сторонами вступает в силу. Для обеспечения автоматизированного исполнения обязательств непременно требуется среда существования, которая позволяет полностью автоматизировать выполнение пунктов контракта. Это означает, что смарт-контракты смогут существовать только внутри среды, имеющей беспрепятственный доступ исполняемого кода к объектам смарт-контракта.

Все условия контракта должны иметь математическое описание и ясную логику исполнения. Таким образом, основной принцип смарт-контракта состоит в полной автоматизации и достоверности исполнения договорных отношений между сторонами.

Метод смарт-контракта

Методом смарт-контрактов CREDITS называется сущность системы, отвечающая за соблюдение логики и последовательности действий при проведении транзакции (действий по контракту).

Логика и последовательность действий описывается программным кодом (модулем), содержащем команды, последовательное выполнение которых позволяет получить искомый результат. Данный код может оперировать системными командами (например, команда присваивания), пользовательскими командами (отдельно написанными функциями), свойствами контракта (статически или динамически проинициализированный переменные, доступные из любого метода контракта), а также методами из любого другого стороннего контракта, использование которого доступного не только владельцу подключаемого (стороннего) контракта. Для большей популяризации - разработка ведется на скриптовых языках (например, JavaScript).

Метод (программный код) допускает использование всех широко распространенных операторов (команд) скриптовых языков (операторы присваивания, условных и безусловных переходов), создание функций и процедур (подпрограмм), подключение сторонних библиотек.

Виртуальная исполняемая машина

Метод контракта системы CREDITS выполняется в виртуальной среде системы (Virtual Machine, далее VM). При вызове метода для определенного контракта в VM выделяется область памяти и в нее загружается байт-код контракта, содержащий в себе методы и проинициализированные (или переопределенные при вызове других методов контракта) переменные. VM начинает обрабатывать байт-код метода, в момент выполнения переменные и код загружаются в ее область памяти, и происходит последовательное выполнение команд, результат выполнения которых передается в пиринговую сеть для последующего помещения в реестр.

Инициатором выполнения метода является пользователь системы, от имени которого происходит запуск этого метода.

Стоимостное выражение

Криптовалюта CREDITS также является показателем стоимостного выражения единицы контракта, что позволяет производить стоимостные сравнения двух абсолютно разных единиц и достигать консенсуса при выполнении или принятии контракта сторонами. Вместо того чтобы регистрировать каждую отдельную комбинацию ценность/шлюз, криптовалюта CREDITS

служит связкой для осуществления меж ценностных переводов. Это возможно потому, что любая ценность ликвидна по отношению к валюте CREDITS, а значит, любая ценность может быть ликвидна по отношению к любой другой ценности.

Выполнения условий смарт контракта

Условие контракта в системе CREDITS - это значения триггерных (проверяемых) полей, необходимых для закрытия (завершения) контракта.

Выполнение условий смарт-контракта - это процедура, при которой проверяются триггерные (искомые) поля на эквивалентное искомое значение. Существуют три возможных пути поиска решения для выполнения условий контракта:

1. Контракт заключен между двумя и более сторонами на передачу ценности. В данном случае выполнение контракта - это предоставление передающей стороне стоимостного эквивалента ценности от принимающей стороны.
2. Контракт заключен между сторонами на передачу ценности, но оплата должна быть осуществлена по выполнению определенного количества условий (например, доставка ценности к принимающей стороне).
3. В систему помещается контракт на конверсию одной ценности в другую со стоимостным эквивалентом в виде CREDITS. В этом случае платформа начинает искать максимально короткий путь по обмену одной ценности на другую через конверсию в других контрактах. Любое выполнение контракта может быть обеспечено за одну транзакцию, либо за несколько, что даст возможность набрать искомое количество единиц ценности для завершения контракта

Источники данных

Для правильной и полной работы, проверки и предоставления дополнительной информации, для принятия более взвешенного и оптимального решения система CREDITS использует сторонних поставщиков данных. Необходимость ввода в систему дополнительных источников данных обусловлено недостаточностью публичной информации об одной или нескольких сторон контракта (например, получение кредитного статуса заемщика для принятия решения о выдачи кредита).

Для работы со сторонними информационными системами платформа обладает возможностью вызывать интеграционную шину, которая по удаленному доступу генерирует запрос к сторонней системе (сайту) в формате представления данных на платной основе для участников системы с оплатой денежными единицами CREDITS.

Запрос отправляется в зашифрованном виде, на порты и адреса, предоставляемые информационными системами, отличными от стандартных. Результатом запроса может считаться любой ответ сервиса, содержащий необходимую информацию для принятия решения, либо код ошибки, характеризующий невозможность получения необходимого ответа и возможные шаги для исправления возникшей ошибки.

5. План реализации

Технический план реализации проекта

	S1	S2	S3	S4	S5

	Pre-Alpha	Alpha	Beta	Release candidate	Release
Storage, Consensus mFA Consensus	FA : Key Design Implementation	mFA : Key Design Implementation PoW (Proof-of-Work) and PoC (Proof-of-Capacity)	mFA Optimization	–	–
Data Store	Decentralization Ledger, NoSQL Store implementation	MessagePack History	–	Blockchain backup	–
CVM (Credits virtual machine)	Design and Implementation	Integration with ecosystem	Optimization	check errors	–
Third-party system	–	Design and Implementation	integrate to full system	–	optimisation
Inference Engine	Formal Specification and Key Design Elements	Reasoner Integration with Blockchain	Reasoner Optimization	–	–
User interface	Implementation	Web UX design	–	–	–
Wallet	Wallet Formal specification		UX design Application Test	–	Android, iOS, Desktop Wallets
RPC & REST API	Formal specification	Blockchain Explorer	–	Third-party explorer	–

Криптовалюта CREDITS

После запуска рабочей версии системы, будет выпущено фиксированное количество – 1 000 000 000 денежных единиц CREDITS. Их обменяют на токены стандарта ERC20, выпущенные на первичной продаже токенов. Обмен будет произведен по зафиксированному курсу: 1 токен стандарта ERC20 = 1 денежная единица CREDITS.