기술 백서

(다른 세부 사항이 추가될 수 있습니다)

탈중앙화 금융 시스템

CREDITS

1.5 버전/2017년 9월 12일

목차

요약

CREDITS 플랫폼은 피어투피어(P2P) 원리를 이용해 참여자들 간 직접적인 트랜잭션이 가능한 탈중앙화 금융 시스템입니다. 우리의 플랫폼은 분산된 장부, 자동 실행되는 스마트 컨트랙트, 그리고 CREDITS만의 가상화폐를 이용하여 금융 서비스의 한계를 넘고 있습니다. 시스템의 목적은 모든 참여자를 한 사이트로 모아, 금융 서비스를 창조하고 사용할 수 있는 플랫폼을 통해 모두가 서비스를 이용할 뿐 아니라 제안할 수 있도록 하는 것입니다. CREDITS 플랫폼은 명확하고 균형 잡힌 기술 시스템을 확립하여, 온라인 참여자들이 상호작용하는 새로운 금융 모델이자 기술적 솔루션을 갖춘 첨단 탈중앙화 금융 서비스를 시작합니다.

소개

자금 이전, 통화와 가치 간 교환, 신용 거래, 펀딩과 참여자들 간의 직접적인 트랜잭션를 가능하게 하는 서비스 전달 체제 간의 완전한 피어투피어 방식을 채택합니다. 우리의 원칙에 따라 서로 다른 시스템의 참여자들 간에도 중개인의 개입이 필요 없습니다. 결과적으로 모두가 더욱 저렴하고, 빠르고 개선된 서비스를 이용할 수 있게 됩니다.

우리의 세상의 상호작용은 점점 사람들 간의 직접적이고 평등한 소통으로 변화하고 있습니다. 이것은 가히 혁신이라고 할 수 있습니다! 이 현상은 미디어에서도 뚜렷이 나타납니다: 1990년대까지는 정보의 주요 전달자는 신문, 잡지와 TV였습니다. 오늘날 여론을 주도하는 것들은 유투브 채널과 소셜 네트워크를 이용하는 블로거들, 크라우드펀딩과 ICO에 투자한 자금, 그리고 탈중앙화 클라우드 시스템입니다.

금융 업계는 이러한 추세와 달리 아직까지 탈중앙화 방식과 참여자들 간 직접적인 소통을 도입하길 꺼려하는 몇 남지 않은 부문 중 하나입니다. 기술적으로는 탈중앙화 금융 시스템의 개발이 무인 자동차를 만드는 것 보다 어렵지도 않습니다.

분산 장부를 기반으로 하는 탈중앙화 금융 상품과 서비스 시스템을 만들기 위해서는 그를 위한 기술적 환경이 갖추어져야 합니다.

1. 빠른 실행 속도(수 초 내)로 거대한 양의 트랜잭션를 동시에 다룰 수 있는 처리 능력(1초 당 몇 만 건)를 각 트랜잭션을(소액 결제와 비 현금 트랜잭션 등) 저렴한 가격에 제공할 수 있어야 합니다.

2. 이에 더해 양질의 탈중앙화 서비스 제공에 필요한 참여자와 아이템이 모두 확보된 시스템을 개발해야하며, 사용자들의 개인화, KYC, 개인 신용 조회 기관, 신용 화폐 거래소, 가상화폐의 인출과 현금화 기능도 필요합니다.

현재 피어투피어 금융 상품의 개발을 방해하는 것은 기본적이지만 큰 과제 두 가지입니다.

우리는 CREDITS 금융 시스템으로 이러한 과제를 해결합니다.

CREDITS 단일 기술적 탈중앙화 플랫폼은 금융 서비스의 모든 참여자를 통합하고, 분산 장부의 원리를 이용하여 모든 트랜잭션을 빠르고 안전하게 처리합니다. 자동 실행되는 스마트 컨트랙트와 연합 투표 제도의 원리를 이용하면 어떤 참여자던 다양한 금융 상품을 만들어 독특한 상호작용을 이어갈 수 있습니다. 이 플랫폼은 블록체인 프로젝트와 금융 서비스, 그리고 속도와 거래 비용의 한계로 이용하지 못했던 다른 부문들에게도 거대한 잠재력이 있는 시장을 열어줍니다.

1. 네트워크 장부

정의

1. 시스템이란 처리 능력을 제공, 트랜잭션 저장, 스마트 컨트랙트의 조항을 실행하고 확인하며 외부 시스템의 프로세싱 요청을 처리하며 요청된 정보를 제공하는 역할을 수행하는 탈중앙화된 노들의 집합입니다.

2. 네트워크 노드란 완전한 네트워크 클라이언트가 설치되어 공유 시스템에 연결되어 있으며, 트랜잭션을 식별하고 장부에 기록하는 컴퓨터입니다.

3. 장부란 시스템이 검증한 트랜잭션의 목록이며 모든 네트워크 노드에 저장되어 있습니다.

4. 트랜잭션이란 시스템의 아이템으로, 스마트 컨트랙트 기능 실행 요청이나 네트워크 상의 모든 행위를 의미하며 블록체인 시스템에 그 결과가 기록됩니다.

5. 스마트 컨트랙트 역시 시스템의 아이템으로, 참여자들 간 상호작용의 조건을 식별하고 지켜지는지 확인하는 컴퓨터 프로토콜입니다. 일반적으로 사용자 인터페이스를 가지며 계약성 관계의 논리를 따릅니다. 스마트 컨트랙트의 주요한 특징은 그 탈중앙화된 성격과 중심적인 소스로부터 독립적으로 존재한다는 것입니다.

6. 스마트 컨트랙트 메소드란 스마트 컨트랙트 조항의 처리 결과를 계산하고 장부에 기록하는 프로그램 코드입니다.

7. 계약 당사자는 네트워크 최종 참여자와 시스템의 이용자입니다.

네트워크 노드

우리는 접근이 무료이며 노드 간 서로 연결된 탈중앙화 네트워크를 만들기 위해 고려하여 다양한 종류의 노드를 사용합니다.

1. 공통 노드(OУ)는 트랜잭션의 유효성을 조회하는 역할에 참여하지만 최소한의 신뢰 요인만을 가지고 있습니다. 또한 네트워크가 수행하는 차기 노드 역할 배정 사이클 시 현재 처리 노드와 신뢰된 노드의 후보입니다.

2. 신뢰된 노드(ДУ)란 트랜잭션 조회에 참여하는 동시에 최대한의 신뢰 요인(1)을 가지며, 현재 처리와 공통 노드 역할의 후보 노드입니다. 이 노드 간 노드 간 수학적으로 계산된 배정과 투표 사이클 중에는 트러스티드가 될 수 없습니다. 수학적 계산은 노드의 수와 네트워크의 복잡도에 따라 상이합니다.

3. 네트워크의 메인 노드(ГУ)란 조회에 참여하는 노드이자 트랜잭션 장부 블록에 트랜잭션를 더하는 역할을 합니다. 이 노드는 노드의 숫자와 네트워크의 복잡성에 따라 수학적으로 계산된 횟수의 사이클 동안은 신뢰된 노드나 현재 처리 노드가 될 수 없습니다.

시스템은 신뢰 요인(네트워크의 전체 노드 +1의 숫자에서 신뢰된 노드의 갯수 비율인, 0과 1 사이의 분수로 표현된 가치의 절댓값)을 이용합니다. 신뢰된 노드의 수는 네트워크의 모든 노드 중 50%를 넘길 수 없습니다.

마지막으로 저장되는 블록

블록의 공통 장부(CRB)는 모든 시스템 노드 간의 전체 공통 장부가 동기화된 상태입니다.

장부 블록 컨텐츠란, 이전 블록의 해시 코드를 담은 정보의 단위와, 이전 블록의 관련 번호를 지닌 이전 장부에 관한 데이터의 리스트입니다. 다른 노드에서 블록을 받은 후에는, 번호에 따라 블록의 공통 장부에 자리를 잡습니다. 이는 네트워크 대역폭을 절약합니다.

동기화 과정 중 블록 번호만 먼저 검사합니다. 이 노드에서 블록이 누락되어 있으면 다운로드하여 저장합니다.

결과적으로, 어떤 시점에서던 장부의 최신 업데이트 본을 가지고 있게 됩니다. 우리는 이것을 마지막 장부(LR)라고 합니다. 합의가 이루어지고 나면 이는 장부를 생성한 노드가 이를 자동으로 생성합니다. 이 블록은 시스템의 노드가 모두 일제히 가장 최신의 장부를 가지고 있을 수 모든 시스템 노드에 전송됩니다.

각 노드는 네트워크의 다른 노드와 연결되어 있으며 트랜잭션이 있는 새로운 블록을 지속적으로 교환하여 관련 정보를 관리합니다. 모든 블록은 장부에 추가 되기를 기다리는 트랜잭션 후보를 선정합니다. 동시에 각 서버는 다른 서버와 제안된 트랜잭션의 집합을 위해 후보를 가정하여 집합을 구성합니다. 확인 후에는 장부에 추가할지 결정합니다.

결과적으로, 다수의 서버에 장부 데이터를 여러 번 저장하는 것이 가능해지고, 시스템 노드와 정보를 모두 보호할 수 있습니다. 시스템에 노드가 많을수록 안정적이고, 독립적입니다.

노드의 동기화

각 노드는 정의가 결정되고 신뢰성 확인을 거친 뒤 동기화 됩니다. 정보의 처리 속도를 높이기 위해 모든 프로세스는 서로 독립적으로 동시에 처리됩니다. 외부에서 유입되는 변수가 없으면 빈 장부가(간소화된 접근을 위해 보유된 RAM 공간) 생성됩니다. 필요한 장부가 없는 경우에는 신뢰할 수 있는 노드에 요청하여 동기화된 계정의 트랜잭션을 처리하도록 합니다.

입력된 매개 변수가 트랜잭션을 정의하는 오브젝트일 경우, 모든 동기화 스레드가 검색을 시작합니다. 이러한 과정으로 신뢰된 노드 장부의 위치 번호를 나타내는 숫자 번호나, 가치가 0보다 작게 나올 때의 오류 번호를 출력합니다. 스레드 방식이 연결 오류로 끝나는 경우에는 스레드가 완전히 종료됩니다.

2. 네트워크 합의제

CREDIT의 합의제는 집단 의사 결정 방식을 채택합니다. 우리는 모든 네트워크가 허용하는 최종 솔루션을 개발 솔루션을 개발하는 것이 목표입니다.

합의제 비교

기타 합의제와 비교하기 위한 CREDITS 탈중앙화 장부의 원리 정의:

- 장부 가용성(노드는 장부에 언제든 데이터를 기록하고 받을 수 있습니다);
- 모든 참여 네트워크에 의해 변경 가능;
- 시스템 노드 사이의 일관성 (모든 노드는 모두 완전히 같은 버전의 장부를 보게 되며, 장부 변경 시에는 업데이트를 받습니다);
- 분리에 대한 저항성 (하나의 호드가 작동하지 않아도 전체 장부의 작동에는 영향을 주지 않습니다).

비교하는 매개 변수	CREDITS의 PoW와 PoC	PoW	PoS
블록을 생성한 노드의 식별 원리	수학 함수의 계산 마지막 장부 사본의 저장 확인	다양한 복잡성을 지닌 수학 함수의 반복 계산 수행	참여자(경합 노드) 간 최대 스택 검색

Attack 51%.	매우 희박합니다. 신뢰된 노드는 역동적으로 선택되고, 자원을 완전히 갖춘 장부와 계산을 할 만한 컴퓨터 처리 능력이 필요하기 때문입니다.	가능하지만, 자원 사용 측면에서 기회 비용이 많이 듭니다.	가능하지만 자신의 스택을 키워야하기 때문에 비용이 많이 듭니다.
장부/블록체인에 추가하는 작업을 수행한 사이트에 대한 보상	자동으로 산정되며, 작업당 요금에 따라 변합니다.	블록 마이닝에 부과하는 고정된 요금	블록 마이닝에 부과하는 고정된 요금

메인 네트워크 노드의 개념

모든 네트워크는 탈중앙화되어 있으며 노드 간 우선순위가 없습니다. 다른 네트워크 노드에 저장된 트랜잭션의 큐를 처리하기 위해 네트워크 노드를 정의해야 합니다. 그 이후에는 장부에 새롭게 생성된 트랜잭션 블록을 등록해야 합니다.

CREDITS 플랫폼은 자체 프로토콜을 이용하여 트랜잭션의 처리 속도를 높이고, 데이터 저장, 처리와 트랜잭션의 안전성을 높입니다. 이 프로토콜은 모든 장부 트랜잭션의 수학 함수의 계산 결과에 기반하며, 작업 증명 원리를 적용합니다. 해시 코드, 즉 가치의 검사합을 계산하여 가장 최신 버전의 장부와 노드의 소프트웨어를 정확히 밝힙니다. 이에 더해 파일의 용량도 알아내어, 가장 최근에 이루어진 시스템 상 트랜잭션에 대한 가장 최신에 업데이트된 사본과 해시 코드임을 증명합니다.

메인 네트워크 노드가 되기 위해서, 마지막으로 저장된 장부에 기반하여 노드는 해시 함수의 값을 검색합니다. 우리는 네트워크끼리 새로운 장부를 생성하고 저장할 수 있는 메인 노드가 되기 위해 건전한 경쟁을 펼치는 환경을 조성합니다.

함수를 계산하여 얻은 결과는 검증을 위해 모든 네트워크 노드에 전송됩니다. 결과는 계산의 타임스탬프와 장부 파일과 소프트웨어 함수의 계산을 한 값을 포함합니다. 모든 노드는 계산된 값을 받은 후, 메인 네트워크 서버를 위한 검색에 배당된 시간을 비교한 후 검증한 후 노드의 신뢰 요인을 확정하고, 메인 네트워크 노드로 경쟁에 참여할 수 있는 기회를 제공하기도 합니다.

모든 네트워크 노드의 승인을 받은 후에는, 타임스탬프가 포함되어 있고, 함수의 값을 정확히 계산한 노드의 리스트를 만듭니다. 올바른 값을 받고 가장 빠른 시간 내에 승인한 노드는 그 순간의 메인 네트워크 노드가 됩니다.

SHA2 알고리즘 개념을 이용하여 파일의 해시 합계를 계산합니다.

SHA2 계열의 해시 함수는 Merkle–Damgard 구조를 기초로 개발됩니다.

추가 뒤의 초기 메시지는 블록으로 나누어져 있으며, 각 블록은 다시 16개의 단어로 나누어집니다. 알고리즘은 각 메시지를 64개나 80개의 반복 과정이 있는 사이클에 돌립니다. 각 반복에서 두 개의 단어가 변환되고, 나머지 단어는 변환 함수를 정의합니다. 블록 프로세스의 결과가 요약됩니다. 그 합계가 해시 함수의 값이 됩니다. 그러나 내부 상태는 이전 블록 처리의 결과에 따라 초기화됩니다. 따라서 블록을 서로 독립적으로 처리하고 그 결과를 합하는 것은 불가능합니다.

네트워크 노드의 장비

우리가 가장 빠른 트랜잭션 처리 특성을 가진 플랫폼을 구축하기 위해, 가장 좋은 상태(고성능의 서버 장비와 가장 넓은 인터넷 대역폭)을 가진 메인 네트워크 노드에게는 물질적인 인센티브를 제공합니다.

메인 네트워크의 인센티브는 CREDITS 처리 장부의 트랜잭션 요금에 따라 책정되는 화폐 보상입니다. 나머지(1/2)는 사용자 지원, 현재 기능 및 신제품 개발 등을 위한 전반적인 프로젝트 개발 예산으로 사용됩니다. 이 비율은 변할 수 있으며, ICO 이후 3년 간 네트워크 노드의 연합 투표에 의한 비율 책정 시스템으로 쪼개질 수도 있습니다.

결　거으로, 우리는 서버 소유자들이 자신의 서버 하드웨어 성능을 관리하고, 양질의 고속 통신 채널을 유지할 것을 권장합니다.

합의 도출

따라서, 우리는 모든 노드가 합의하여 메인 네트워크 노드를 선택하도록 합니다. 메인 노드의 주요 작업은 다음과 같습니다: 후보 상태의 트랜잭션을 구하여 모든 노드의 장부에 추가하고, 처리하며, 마지막 유효 장부를 구축하며 모든 네트워크 노드에 새로운 장부를 전송합니다. 트랜잭션 처리의 과정과 마지막 유효 장부의 구축 모두 합의 솔루션을 요구합니다. 마지막 유효 장부의 구축 결과가 곧 합의로 도출된 솔루션입니다.

그 과정은 다음의 단계로 나눌 수 있습니다:

1. 메인 네트워크 노드 탐색;
2. 신뢰된 노드 구축;
3. 트랜잭션 목록을 수신하고 장부에 추가할 후보의 목록 생성;
4. 후보의 목록을 처리하고, 노드 투표 실시 (신뢰된 노드와 공통 노드는 신뢰 요인으로 인해 서로 다른 가중치를 부여받음);
5. 후보자의 목록에서 검증되지 않은 트랜잭션이나 거부된 트랜잭션 제거;
6. 장부에 추가될 검증된 트랜잭션 목록 생성;
7. 장부에 보유한 블록의 해시 코드와 타임스탬프가 붙은 트랜잭션 추가
8. 네트워크 노드 전체에 트랜잭션을 보유한 블록 전송 수신 후에는, 노드의 레지스트리에 추가됩니다.

장부의 개발과 개시

이 과정은 다음의 단계를 거칩니다:

1. 네트워크의 최종 사용자가 트랜잭션을 생성합니다.
2. 스마트 컨트랙트의 모든 조건이 충족되면, 올바른 방법으로 플랫폼 소프트웨어를 이용해 사용자는 동작(트랜잭션)을 개시합니다.
3. 블록체인의 기본 원리를 따르기 위해서, 유효성 검사기의 핵심이 동기화와 장부의 업데이트 상태를 계속하여 확인합니다.
4. 합의를 도출하는 과정에서는, 사이클 중 수신된 모든 트랜잭션은 블록에 수집됩니다.
5. 블록에는 타임 스탬프와 해시 코드로 변환된 노드 식별자가 붙여지며, 그 후 합의 모듈에 배치됩니다.
6. 후보자 화이트 리스트를 만든 후에, 트랜잭션의 해시 뿐 아니라 블록의 해시까지 장부에 기록하여 언제든지 트랜잭션의 소스를 찾을 수 있도록 합니다.

7. 이 해시는 블록과, 트랜잭션으로 이 블록을 만든 사람의 서명 역할을 합니다.
8. 연합 검색 알고리즘을 통해 합의를 구축한 후, 블록에 추가된 트랜잭션은 장부에 기록되기 위한 전 단계로, 유효성 검사기의 커널로 전송됩니다.

레지스터에 포함되지 않은 트랜잭션

준비된 트랜잭션 목록에 포함되지 않은 트랜잭션은 거부된 것으로 표시합니다. 이 정보는 즉시 트랜잭션의 발송자 (실행인) 단계에서 드러납니다.

장부에 포함되지 않은 트랜잭션은 후보군으로 남으며 네트워크 노드에 저장됩니다. 서버가 합의 도출을 할 동안 수신한 새로운 트랜잭션 또한 이 곳으로 보내지며, 검색 프로세스가 다시 시작됩니다. 네트워크의 이러한 지속적인 순환 형태는 짧은 시간 내 트랜잭션을 처리함과 동시에 높은 수준의 신뢰도와 정보의 적시성을 보장합니다.

3. 트랜젝션 프로세싱

트랜젝션

트랜잭션이란 컨트랙트 방식이나 계정 간 스마트 컨트랙트 없이 이루어진 거래를 플랫폼에 보고하는 시스템의 최소 단위로, 그 결과는 피어투피어 네트워크에 배치됩니다.

합의 도출

시스템은 합의를 도출하기 위해 연합 모델을(신뢰된 유효성 검사기 노드의 투표와, 유한 상태 자동화의 통로인 합의 도출 알고리즘) 사용합니다. 합의는 사이클로(시간 단계) 이루어지며, 각 시간 단계마다 트랜잭션은 풀에 배치됩니다(일차원 배열). 풀에 배치된 모든 트랜잭션은 응답을 받기 위해 신뢰된 노드로 보내집니다. 응답이 수신되면, 추가되길 요청한 트랜잭션은 이 유효성 검사기의 장부에 추가될 수 있습니다. 그 후에는, 네트워크의 다음 유효성 검사기로 전송됩니다. 전송의 유효성이 완전히 검증되는 체인의 끝에서 합의가 도출되고 나면, 트랜잭션은 타당성 검증이 되어 장부에 기록되고 저장되기 위한 표시를 붙여 전송됩니다.

트랜잭션 프로세싱

시스템이 탈중앙화되기 위해서는, 각 서버는 장부와 스토리지를 갖추어야 하며, 모든 트랜잭션을 완벽히 처리할 수 있어야 합니다.

시스템은 시스템 커널 개념을 사용합니다. 커널이란, 남은 시스템 요소의 가용성이나 작동성과 상관없이 특정 생산 태스크를 수행하는 데이터의 다루개를 의미합니다. 각 커널은 입력 단계에서, 태스크가 실행되는 시점에서 프로세싱 할 변수의 목록을 받습니다. 그리고 항상 출력 단계에서 긍정, 기타, 오류로 결과를 나타냅니다. 따라서 시스템 커널은 항상 메인 데이터 집합과 함께 응답 코드를 가지고 보유하고 있습니다. 이러한 구조는 서로 독립적으로 실행되는 각 프로세스가 최고 속도로 처리되기 위해 필요한 것입니다.

장부 입력 구조

보안을 유지하면서도 장부의 성능을 높이기 위해서, 우리는 이전 블록의 해시 코드와 트랜잭션 결과로 이루어진 Merkle 트리가 없는 장부 데이터 베이스를 제안합니다.

Merkle 트리 (TTH – Tiger Tree Hashing)는 데이터의 무결성을 검사하고, 체인의 고유 식별자를 구하거나 시퀀스를 복원하기 위해 사용되는 해시 함수의 일종입니다. Merkle 트리에서 데이터는 Leaf Tiger Hash를 이용해 개별적으로 해시된 블록으로 쪼개어지며, 각 해시 짝의 Internal Tiger Hash를 하나 하나 계산합니다. 해시가 짝이 없으면, 변경되지 않은 채 새로운 체인으로 전송됩니다. 그 다음, 각 짝의 Internal Tiger Hash가 다시 계산됩니다. 이 과정은 해시가 하나 남을 때 까지 반복됩니다.

Merkle 트리를 이용해 운영된 장부는 트랜잭션 처리 속도가 매우 느리며 컴퓨팅 처리 능력에 대한 부담은 매우 큽니다. 우리의 생각으로는, 이는 합리적인 데이터 저장 방법이 아닙니다.

CREDITS의 장부 구조

우리는 Merkle 트리를 사용하지 않고, 모든 입력 사항이 후보 목록과 장부에 추가되기 위한 트랜잭션 블록의 해시 코드를 포함하고 있는, 트랜잭션 블록 CREDITS 시스템의 트랜잭션 장부를 이용하고자 합니다. 또한 입력 사항은 노드의 식별자와 생성된 시기의 타임 스탬프를 가지고 있습니다. 장부 항목에는 트랜잭션의 방향, 초기 및 최종 계정, 대손상각 단위의 수, 입금의 유형 및 입금 단위의 수가 있습니다. 이 원리는 트랜잭션 처리의 속도를 높이고, 불법 장부 변경을 하기 어렵게하며, 장부 항목의 뒤늦은 변경을 불가능하게 만들어 줍니다.

블록 크기

시간의 단위는 메인 노드와 신뢰된 노드를 탐색하기 위한 사이클이며, 사이클의 시간은 네트워크의 복잡도에 따라 다르게 계산됩니다. 각 시간 단위 당, 네트워크는 생성되어 처리를 위해 네트워크에 전송된 트랜잭션 N개를 보유하며 이는 이전 사이클의 끝부터 다음 사이클의 시작까지 진행되어, "장부에 추가되기 위한 후보"의 상태가 됩니다. 네트워크 N에서 선택된 트랜잭션이 블록에 배치됩니다. 블록의 용량은 트랜잭션의 수에 따라 달라집니다.

트랜잭션 참여자 탐색

CREDITS 피어투피어 네트워크는 사용자 계정은 꼭지점으로, 가능한 수많은 트랜잭션을 두 꼭지점을 잇는 방향이 정해진 테두리 선으로 표현된 그래프로 나타낼 수 있습니다. 모든 테두리 선은 시작점과 끝점이 있으므로, 항상 방향 지향 그래프(ograph)를 만들 수 있습니다.

다음을 식별의 조건이라고 두면:

- 모든 트랜잭션에는 발신자와 수신자가 있습니다.
- 모든 꼭지점 (계정)은 항상 방향성 있는 테두리 선 (트랜재션)으로 다른 꼭지점으로 연결될 수 있습니다.
- 그래프의 모든 꼭지점(계정)은 유한한 수의 방향성 있는 테두리 선(들어오고 나가는 트랜잭션)을 가지고 있습니다

위의 내용으로 미루어보았을 때, ograph가 트랜잭션 조건을 만족시키고 단순한 체인을 형성하기 위한 루트를 가지고 있다는 것을 알 수 있습니다. 꼭지점은 유한한 시퀀스로 이루어져 있으므로, 각 꼭지점은 (마지막을 제외하고) 테두리 선에 의해 다음 꼭지점과 연결되어 있습니다.

데이터 전송 채널

CREDITS 네트워크에서 메인 네트워크 노드와 공통 노드 사이에 이루어지는 통신 채널은 서로 구분된 스레드로(멀티 스레딩 기법), 트랜잭션 실행 시 데이터는 암호화된 형태로 전송됩니다.

네트워크의 안전성을 보장하기 위해, 유효성 검사기 노드 간 모든 데이터는 암호화 형태로 전송되며, 노드 간의 연결은 일정 수준 네트워크 라이브러리를 기반으로 합니다. 데이터 전송 중 오류가 발생하면 자동으로 스레드가 방해되고, 해당 항목은 로깅 시스템, 그리고 그 이후에는 로그 파일에 쓰여집니다. 데이터는 전형적인 변수를 통해 전송됩니다. 전송된 데이터는 대칭 RC4 알고리즘을 이용해 암호화됩니다. 이 알고리즘은 공통 비밀 키 아래 움직이므로, 노드끼리 연결되면 키가 전송되며 DiffieHellman 알고리즘으로 암호화되어 출력됩니다.

RC4 알고리즘은 여느 스팀 암호화 같이 의사 랜덤 비트 생성기를 기반으로 구축됩니다. 키가 생성기의 입력으로 쓰이고, 의사 랜덤 비트가 결과로 출력됩니다. 키의 길이는 40에서 2048 비트까지 이릅니다. 생성된 비트는 동일하게 분포합니다.

DiffieHellman 알고리즘은 리스닝으로부터 자유롭지는 않지만 통신 채널 변경으로부터는 보호된 채널을 통해 두 참여자가 공통 비밀 키를 수신하게 해 줍니다. 수신된 키는 대칭 암호화를 이용하여 메시지를 교환하는 데 이용합니다. 알고리즘은 개별 로그 컴퓨팅의 복잡성에 기반합니다. 공개 키를 가진 많은 알고리들과 같이 그 안에서는, 특정 큰 소수 P까지 모듈로 계산이 실행됩니다.

첫번째로, P보다 작은 특정 자연수 A가 특별한 방식으로 선정됩니다. X를 암호화하기 위해서는

$$Y = AX \bmod P$$

를 계산합니다.

X를 통해 Y를 계산하는 것은 쉽습니다. 그러나 역으로 Y를 통해 X를 계산하는 것은 다소 복잡합니다. 지수 X는 이산대수 Y라고 합니다. 따라서, 이산대수 Y를 산출하는 것의 복잡성을 알면, 모듈러스 P가 크다면 초기값 X를 찾기 거의 불가능하기 때문에 Y를 어떠한 통신 채널에서도 공개적으로 전송할 수 있습니다. 키를 생성하기 위해 사용되는 DiffieHellman 알고리즘은 이러한 수학적 사실에 기반합니다.

시스템의 모든 동작은 타임스탬프, 이전 블록의 번호, 사용자의 로그인과 스마트 컨트랙드 ID와 연계됩니다. 이를 통해 실행 중 복제품을 찾을 수 있습니다. 복제품을 발견하면, 풀에서 트랜잭션을 제거하고, 나머지는 위법으로 간주합니다.

시스템의 동작

시스템의 동작이란 합의 탐색 하부조직의 솔루션을 찾기 위해 계정 간 교환이나 컨트랙트 방식을 유효성 검사기로 전송하는 트랜잭션입니다.

같은 블록 내 동일한 식별자를 가진 트랜잭션이 없도록 시스템은 처리를 위해 유효성 하부조직으로 보내진 트랜잭션이 진위의 유일한 트랜잭션이라는 협약을 받아들입니다. 이미 유효성 검사기 시스템에는 트랜잭션이 현재 거래에서 발생했다는 사실이 기록되어 있고, 트랜잭션을 수행할 값들이 계정에 남아있지 않기 문에 다시 합의를 도출할 수 없습니다. 이로써 이중 폐기물의 문제가 해결됩니다.

트랜잭션이 실행되면, 그 정보는 유효성 검사기로 보내져 검증을 받으며, 장부의 변화에 관한 정보를 자동으로 신뢰된 목록의 노드들에게 보낸 후, 장부가 동기화됩니다.

현재 유효성 검사기 노드를 위해 모든 신뢰된 노드가 최신의 트랜잭션 장부를 가지기 위해서는, 모든 노드끼리 새로 시작된 트랜잭션에 매번 동기화해야 합니다. 이 문제를 해결하기 위해, (그러한 기회가 있는 경우) 동기화를 위한 별도의 포트가 사용되어야 합니다. 이러한 기회는 포트에 가해지는 부하를 분산해 유효성 검사기 커널에 유입되는 정보의 처리 속도를 높입니다. 동기화 스레드는 항상 주기적으로 실행됩니다. RAM과 (CPU 사이클을 이용한) CPU 부하 할당의 우선 순위는 평균보다 낮습니다. 메모리가 대신하여 (동시 알고리즘으로 암호화된 형태인) 마지막 1,000개의 작업과 계정의 상태를 저장하며, 이는 다른 유효성 검사기 노드의 요청의 처리 속도를 높입니다.

유효성 검사를 위해 트랜잭션 추가하기

장부에 트랜잭션을 추가하는 것은 합의 도축과 장부에 추가할 트랜잭션의 결과가 담긴 화이트 리스트를 만든 후 즉시 유효성 검사 하부조직으로부터 불려옵니다. 보안성을 개선하기 위해 외부 시스템이 불러오는 것은 불가능합니다.

유입되는 매개변수– 트랜잭션을 특정짓는 개체입니다. 결과의 값인 ResultValue<0

– 실행은 오류로 중단되며, 결과값은 possible error code / 0 < ResultValue 함수는 오류 없이 실행되었으며, 결과는 장부 항목의 번호입니다.

유입 매개변수– 트랜잭션의 고유 레이블, 발신자, 수신자, 전송된 값, 가치의 대응값, 목표값, 전송된 가치의 양, 목표값의 양과 필요시 변경할 수 있는 시스템 정보를 보유한 개체입니다.

트랜잭션 비용

시스템이 사용하는 CREDITS 통화는 다음과 같이 쓰입니다:

- 시스템 이용에 대한 내부 지불 수단;
- 시스템 내 다른 통화와의 교환할 시;
- 시스템 내 다양한 가치를 교환할 시;
- 스마트 컨트랙트로 제어되는 작업을 생성하고 처리하기 위해;
- 시스템 내의 서비스를 위해 외부 소스에게서 정보를 구입할 때;

트랜잭션의 비용은, 이론적으로 피크 타임에 많은 양의 트랜잭션의 흐름을 관리할 수 있는 시스템의 특정 사용자가 부담하는 네트워크 부하에 따라 달라집니다. 우리는 시스템 사용자들에게 물질적인 혜택을 제공하여 네트워크 부하를 조정하도록 유도할 것입니다.

시스템 운영의 첫 3년 동안은 트랜잭션을 실행하는 비용은 트랜잭션과 작업의 종류에 따라 개별적으로 책정될 것입니다. 미래에는, 트랜잭션의 비용을 자동으로 계산하는 알고리즘이 개발될 것입니다.

4. 스마트 컨트랙트

소개

CREDITS 시스템의 스마트 컨트랙트는 실제 세계와 디지털 시스템의 동작과 사건을 연결한 조건들의 집합을 나타내는 전자 알고리즘입니다.

자동 제어되는 스마트 컨트렉트를 실행하기 위해, 인간의 개입을 완전히 배제한 탈중앙화 환경이 필요하며, 스마트 컨트랙트의 비용을 전송하기 위해서는 중앙 권력 기관이 없는 독립적인 가상화폐가 필요합니다.

개체

CREDITS 스마트 컨트랙트는 다음의 개체를 포함합니다:

1. 속성 (공개 변수) – CREDITS 시스템의 컨트랙트가 작동하기 위해 필요한 공개 데이터를 저장하는 개체입니다.
2. 메소드는 트랜잭션 수행 시 로직과 동작의 시퀀스를 관측하는 CREDITS의 시스템 개체입니다.

CREDITS 시스템의 참여자들은 조건 준수와 협력 사실을 검증하는 프로세스를 실행하여 컨트랙트 속성을 변경하는 메소드를 호출하여 스마트 컨트랙트에 서명을 하고,

스마트 컨트랙트는 참여자들이 서명한 후 효력을 발휘합니다. 명시된 의무가 자동으로 이행됨을 보장하기 위해, 컨트랙트 조건을 완전히 자동으로 실행되는 존재 환경이 필요합니다. 이는 스마트 컨트랙트가 스마트 컨트랙트 항목의 실행 가능한 항목에 방해받지 않고 접근할 수 있는 환경에서만 존재할 수 있다는 의미입니다.

컨트랙트의 모든 조건은 수학적 설명과 명확한 실행 로직을 갖추어야 합니다. 즉, 스마트 컨트랙트의 주요 원리는 완전한 자동화가 참여자 간 계약 관계에 대한 신뢰입니다.

스마트 컨트랙트 메소드

CREDITS 스마트 컨트랙트 메소드는 로직과 트랜잭션 중의 동작(컨트랙트의 동작)에 대한 준수를 책임지는 시스템 개체입니다.

동작의 로직과 시퀀스는 명령어를 가진 프로그램 코드(모듈)로 표현되며, 순차적 실행으로 목표값에 도달합니다. 이 코드는 시스템 명령(예를 들어 할당 명령), 사용자 명령(별도로 작성된 함수), 계약 속성(어떤 컨트랙트 메소드에서도 정적 혹은 동적으로 초기화된 변수, 그리고 주인과 연결된(제3자) 컨트랙트끼리만 연결하는 기타 외부 계약의 메소드를 다룹니다. 대중화를 위해 이는 다양한 스크립팅 언어(JavaScript)로 제공됩니다.

메소드(프로그램 코드)는 다양한 스크립팅 언어 연산자(명령어)(할당, 조건부 및 무조건 점프), 함수의 생성과 절차(서브루틴), 제3자 라이브러리 간 연결을 허용합니다.

가상 실행 서버

CREDITS 시스템의 컨트랙트 메소드는 시스템의 가상 환경(가상 서버, 이하 VM)을 통해 실행됩니다. 특정 컨트랙트를 위해 메소드가 호출되면, VM이 메모리 공간을 할당하고 메소드와 초기화된 변수(혹은 다른 컨트랙트 메소드를 불러올 때 재정의된 변수)를 가진 컨트랙트 바이트코드를 로드합니다. VM이 메소드 바이트코드를 처리하기 시작하고, 런타임 시 변수와 코드가 메모리 공간으로 로드되며, 명령이 실행되고 결과는 장부에 배치되기 전 피어투피어 네트워크로 전송됩니다.

실행 메소드를 개시하는 사람은 시스템의 사용자로, 그의 대행으로 메소드가 실행됩니다.

가치의 단위

CREDITS 가상화폐는 완전히 다른 두 단위의 가치를 비교할 때 상대적인 값을 나타내기 위해, 그리고 계약 당사자들 간 합의를 도출할 때 혹은 컨트랙트를 실행하거나 허락할 때 사용됩니다. 가치/게이트웨이 조합을 모두 등록하기보다 CREDIT 가상화폐를 사용하며 다른 단위 가치 간 교환을 할 때 용이합니다. 이는 CREDITS 화폐에 대해 어떤 가치도 유동성을 가질 수 있기 때문에 가능하며, 이는 곧 다른 단위의 가치끼리 모두 서로에 대해 유동성을 가칠 수 있다는 의미입니다.

스마트 컨트랙트 조건 수행

CREDITS 시스템의 컨트랙트 조건은 컨트랙트를 종결(완성)하기 위해 필요한 (확인된)트리거 필드의 값입니다.

스마트 컨트랙트 조건의 이행은 트리거 (목표) 필드가 목표값과 일치하는지 여부를 확인하는 절차입니다. 컨트랙트 조건을 충족하는 방법에는 3가지가 있습니다:

1. 컨트랙트가 가치의 이동 후 2명 이상의 당사자 사이에서 종결됩니다. 이 경우에, 컨트랙트의 이행은 이전하는 당사로부터 수신하는 당사자에게 전송되는 가치와 일치하는 비용의 제공입니다.
2. 컨트랙트는 가치의 이전 후 당사자들 간에는 종결되지만, 비용 지불은 몇 가지의 조건을 충족한 뒤 이루어져야 합니다(예를 들어, 수신하는 당사에게까지 전달되어야 합니다).
3. 시스템에는 한 가치를 CREDITS의 형태로 변환하는 컨트랙트가 배치되어 있습니다. 이 경우, 플랫폼은 다른 컨트랙트 내 변환이 이루어질 시 최단의 경로를 탐색합니다. 컨트랙트의 완료는 트랜잭션 하나 당, 혹은 트랜잭션 여러 개당 제공되며, 이는 컨트랙트를 완료하기 위한 단위 가치의 양의 채울 기회를 제공합니다.

데이터 소스

CREIDTS는 정확하고 최적화된 작업과 정보를 추가적으로 제공하고 확인하여 균형 잡힌 최적의 솔루션을 위해 외부 데이터 제공자를 사용합니다. 시스템에 추가 데이터 소스를 도입하는 이유는 계약 당사자들에 대해 공개된 정보가 적은 문제를 해결하기 위함입니다(예를 들어, 대출자의 신용 상태를 확인할 경우 필요합니다).

외부 정보 시스템과 협업하기 위해, 플랫폼은 integration bus를 이용하여 원격으로 외부 시스템의 데이터 전시 포맷으로 요청을 넣으며, 참여자들에게 CREDITS로 비용을 지불합니다.

이 요청은 표준의 것과 다르게 정보 시스템에서 제공하는 포트와 주소로 암호화된 형태로 보내집니다. 요청의 결과는 결정을 내리는 데 필요한 정보를 포함 서비스에 대한 모든 응답, 혹은 목표한 대답의 수신을 방해한 오류 코드와 그 오류를 해결하기 위한 절차에 대한 안내일 수 있습니다.

5. 시행 계획

프로젝트 시행의 기술적 계획

	S1	S2	S3	S4	S5
	프리 알파	알파	베타	Release candidate	출시
스토리지, 합의 mFA 합의	FA : 키 디자인 실행 n	mFA : 키 디자인 실행 Pow (Proof-of-Wor k) 와 PoC (Proof-of-C apacity)	mFA 최적화	-	-

데이터 저장소	장부 탈중앙화, NoSQL 스토어 시행	메시지팩 히스토리	–	블록체인 백업	–
CVM (Credits 가상 서버)	디자인과 실행	생태계와의 통합	최적화	오류 확인	–
제3자 시스템	–	디자인과 실행	완전한 시스템 통합	–	최적화
추론 엔진	형식 명세서와 키 디자인 요소	블록체인과 Reasoner 통합	Reasoner 최적화	–	–
사용자 인터페이스	실행	웹 UX 디자인	–	–	–
월렛	월렛 형식 명세서		UX 디자인 어플리케이션 시험	–	Android, iOS, 데스크톱 월렛
RPC & REST API	형식 명세서	블록체인 익스플로러	–	제3자 익스플로러	–

CREDITS 가상화폐

시스템의 릴리스 버전을 출시한 후, 1,000,000,000 CREDITS 라는 고정된 양이 발행될 것입니다. ERC20 표준 토큰으로 구매할 수 있으며, 초기 토큰 세일에서 지급됩니다. 이는 고정된 교환률로 거래될 것입니다:
1 ERC20 표준 토큰 = 1 CREDITS 화폐 단위