

EOS.IO Technical White Paper v2

2018. 3. 16

개요: EOS.IO 소프트웨어(이하 EOS.IO)는 분산 애플리케이션의 수직/수평 확장을 가능하게 하기 위해 설계된 새로운 블록체인 아키텍처를 소개합니다. 이를 위해 애플리케이션이 동작할 수 있는 운영체제와 유사한 구조를 구현했습니다. EOS.IO는 계정을 비롯해 인증, DB, 비동기 통신, 다수의 CPU 코어/클러스터에서 애플리케이션을 스케줄링하는 기능 등을 제공합니다. 그 결과, 운영 차원에서 궁극적으로 초당 수백만 개의 트랜잭션까지 확장 가능하며, 무료로, 분산된 애플리케이션을 빠르고 쉽게 배포하고 유지 관리할 수 있는 블록체인 아키텍처가 태어났습니다.

주의 사항: 이 백서에 언급된 암호화된 토큰은 EOS.IO를 채택해 출시된 블록체인의 암호화된 토큰을 의미합니다. EOS 토큰 배포와 함께 Ethereum 블록체인에 배포된 ERC-20 호환 토큰을 의미하지 않습니다.

Copyright © 2018 block.one

원 출처와 해당 저작권을 고지할 경우 누구든지 허가 없이도 비영리적이며 교육적인 용도(즉, 무료나 비상업적 목적)로 본 백서의 내용을 사용, 또는 복제, 배포할 수 있습니다.

면책 조항: 이 EOS.IO 기술 백서 v2는 정보 제공 목적으로만 제공됩니다. block.one은 이 백서의 정확성이나 백서로 유도되는 결론에 대해 보장하지 않으며 이 백서는 "있는 그대로" 제공됩니다. block.one은 다음 항목을 포함해 이에 국한되지 않는 범위의 모든 보증이나 직접적, 묵시적, 법적, 혹은 기타 진술들을 명시적으로 부인합니다: (i) 상품성이나 특정 목적에의 적합성, 사용성, 소유권, 비침해성 보증; (ii) 본 백서의 내용에 오류가 없음; (iii) 그러한 내용이 제 3 자의 권리를 침해하지 않음. 아울러 block.one 및 그 계열사는 본 백서 또는 여기에 포함된 내용의 사용, 참조 또는 신뢰로 인해 발생하는 어떠한 종류의 손해에 대해서도 설명 사전에 손해의 가능성을 통보받았을지라도 어떠한 책임을 지지 않습니다. 불특정한 개인이나 단체가 본 백서 및 그 안에 포함된 내용을 사용하거나 참고하거나 의존하거나 하여 입은 사업이나 예산, 이익, 데이터, 사용, 영업권, 기타 무형의 손실이 포함된 직/간접적 혹은 결과적, 보상적, 우연적, 실제적, 모범적, 징벌적, 특수한 피해나 손실, 책임, 비용, 경비에 대해서 block.one 및 그 계열사는 어떠한 경우에도 책임이 없습니다.

- 배경
- 블록체인 애플리케이션 요구사항
 - 수백만 사용자 지원
 - 무료 사용
 - 손쉬운 업그레이드 및 버그 수정
 - 낮은 대기시간
 - 순차 처리 성능
 - 병렬 처리 성능
- 합의 알고리즘(Consensus Algorithm, BFT-DPOS)
 - 트랜잭션 확정
 - 지분 증명 기반 트랜잭션(Transaction as Proof of Stake, TaPoS)
- 계정
 - 액션 & 핸들러
 - 역할 기반 권한 관리
 - 명명된 권한 수준
 - 권한 매핑
 - 권한 평가
 - 기본 권한 그룹
 - 권한의 병렬 평가
 - 강제 지연과 관련된 액션
 - 도난당한 키의 복구
- 애플리케이션의 결정론적 병렬 실행

- 통신 대기시간 최소화
- 읽기 전용 액션 핸들러
- 여러 계정에 대한 원자적 트랜잭션
- 블록체인 상태의 부분 평가
- 주관적인 최선의 스케줄링
- 지연된 트랜잭션
- 문맥 무관 액션
- 토큰 모델 및 자원 사용
 - 객관적이지자 주관적인 측정
 - 수신자 부담
 - 용량 위임
 - 트랜잭션 비용을 토큰 가치와 구분
 - 상태 저장 비용
 - 블록 보상
 - 워커 제안 시스템
- 운영(Governance)
 - 계정 동결
 - 계정 코드 변경
 - 헌법
 - 프로토콜과 헌법의 업그레이드
 - 긴급 상황에서의 변경
- 스크립트 & 가상 머신
 - 스키마로 정의된 액션
 - 스키마로 정의된 DB
 - 일반적인 멀티 인덱스 DB API
 - 애플리케이션에서 인증을 분리
- 블록체인 간 통신
 - 경량 클라이언트 검증(Light Client Validation, LCV)을 위한 머클 증명
 - 체인 간 통신의 지연 시간
 - 완전성 증명
 - 증인 격리(분리)
- 결론

배경

블록체인 기술은 2008년, Bitcoin 통화의 출시와 함께 등장하였으며, 이후 기업과 개발자들은 단일 블록체인 플랫폼 상에서 보다 다양한 애플리케이션을 지원하는 기술을 일반화하려고 시도했습니다.

제대로 동작하는 분산 애플리케이션을 지원하기 위해 많은 블록체인 플랫폼이 어려움을 겪고 있지만, BitShares 분산형 거래소(2014) 및 Steem 소셜 미디어 플랫폼(2016)과 같은 애플리케이션형 블록체인은 수만 명의 일일 사용자를 가진 블록체인으로 성장했습니다. 이는 초당 수 천 개의 트랜잭션을 처리할 수 있도록 성능을 향상시키고, 지연 시간을 1.5 초로 단축하고, 트랜잭션 당 비용을 없애고 기존의 중앙 집중식 서비스에서 현재 제공하는 것과 유사한 사용자 경험을 제공하여 이뤄낸 결과입니다.

현존하는 블록체인 플랫폼은 막대한 수수료와 제한된 계산 용량으로 인해 광범위하게 사용되기 어렵다는 문제를 안고 있습니다.

블록체인 애플리케이션 요구사항

블록체인 애플리케이션을 널리 통용시키려면 다음 요구사항을 충족할 만큼 유연한 플랫폼이 필요합니다:

수백만 사용자 지원

eBay나 Uber, AirBnB, Facebook 등의 서비스와 경쟁하려면 수천만 명의 일일 사용자를 처리할 수 있는 블록체인 기술이 필요합니다. 경우에 따라 매우 많은 사용자가 사용하지 않는 한 원활히 동작하지 않는 애플리케이션이 있을 수 있기 때문에 매우 많은 수의 사용자를 처리할 수 있는 플랫폼은 무엇보다 중요합니다.

무료 사용

애플리케이션 개발자에게는 사용자에게 무료 서비스를 제공할 수 있는 유연성이 지원돼야 합니다; 사용자가 비용을 지불해야만 플랫폼을 사용하거나 서비스 혜택을 누릴 수 있어서는 안됩니다. 무료로 사용할 수 있는 블록체인 플랫폼일수록 더 널리 사용될 것입니다. (널리 사용되고 난) 이후에 개발자와 기업은 효과적인 수익 창출 전략을 만들어 낼 수 있습니다.

손쉬운 업그레이드 및 버그 수정

블록체인 기반 애플리케이션을 개발하는 기업이 유연하게 새 기능을 추가해 애플리케이션을 향상시킬 수 있어야 합니다. 플랫폼은 소프트웨어 및 스마트 컨트랙트 업그레이드를 지원해야 합니다.

모든 주요 소프트웨어는 아무리 엄격한 공식 검증을 거친다 해도 버그를 가질 수 있습니다. 불가피하게 버그가 발생했을 경우, 플랫폼은 안정적으로 버그를 수정할 수 있어야 합니다.

낮은 대기시간

좋은 사용자 경험에는 몇 초 내 지연시간으로 전달되는 안정적 피드백이 필수입니다. 지연시간이 길어질수록 사용자의 불만이 커져, 기존 애플리케이션에 대한 블록체인 애플리케이션의 경쟁력이 떨어집니다. 플랫폼은 낮은 대기시간의 트랜잭션을 지원해야 합니다.

순차 처리 성능

순차적으로 처리돼야만 해서 병렬 처리 알고리즘을 이용해 구현할 수 없는 애플리케이션도 있습니다. 거래소와 같은 애플리케이션은 많은 거래량을 순차적으로 처리할 수 있는 성능을 가져야 하기 때문에 플랫폼은 빠른 순차 성능을 지원해야 합니다.

병렬 처리 성능

대규모 애플리케이션은 여러 CPU와 컴퓨터에서 작업 부하를 나누어 처리할 수 있어야 합니다.

합의 알고리즘(Consensus Algorithm, BFT-DPOS)

EOS.IO는 블록체인 애플리케이션의 성능 요구사항을 충족할 수 있는 현재 유일한 분산 합의 알고리즘, 지분 위임 증명(Delegated Proof of Stake, DPOS)을 사용합니다. 이 알고리즘에 따르면, EOS.IO를 채택한 블록체인의 토큰 보유자는 반복되는 승인 투표 시스템을 통해 블록 프로듀서(Block Producer, BP)를 선택할 수 있습니다. 또한 모든 BP 입후보자는 토큰 보유자를 설득해 자신에게 투표하게끔 함으로서 블록을 생산할 수 있는 기회를 얻을 수 있습니다.

EOS.IO는 정확히 0.5 초마다 블록을 생산하도록 하며 단 하나의 BP 만이 그 시점에서 블록을 생산할 수 있습니다. 블록이 예정된 시점에 생산되지 않으면, 그 시점은 블록 생산 없이 지나갑니다. 블록이 하나 이상 생산되지 않고 지나갈 경우 블록체인에는 0.5초 이상의 생산 간격이 생기게 됩니다.

EOS.IO를 사용하면 라운드 당 126(21 명의 BP당 블록 6 개씩) 단계에 걸쳐 블록이 생산됩니다. 매 라운드가 시작될 때 21곳의 BP가 토큰 득표순에 따라 선택됩니다. 선정된 BP는 15 명 이상의 BP가 합의한 순서에 따라 생산을 진행합니다.

(역자 주: 2018년 4월 13일 현재 코드를 확인해보면, BP당 블록 12개씩 만듭니다. 그래서 라운드당 252개의 블록이 생산됩니다.)

자기 차례에 블록을 생산하지 않은 BP가 지난 24시간 내에 블록을 생산한 기록이 없었을 경우, 이 BP는 블록을 다시 생산하기 시작한다는 의사를 블록체인에 알릴 때까지 생산 순서에서 제외됩니다. 이것은 신뢰할 수 없다고 판명된 BP에게 계속 생산을 맡겨 누락될 블록 수를 최소화하여 네트워크를 원활하게 작동하도록 보장하기 위함입니다.

정상적인 상황에서 DPOS 블록체인은 BP 간 경쟁 없이 협력하여 블록을 생산하기 때문에 어떤 포크(fork)도 발생하지 않습니다. 포크가 발생할 경우 합의하에 자동으로 가장 긴 체인으로 전환됩니다. 블록이 블록체인 포크에 추가되는 속도는 동일한 합의를 공유하는 블록 프로듀서의 비율과 직접적으로 관련돼 있기 때문입니다. 즉, 더 많은 BP가 참여한 포크는 누락된 블록이 더 적을 것이기 때문에 참여한 BP가 적은 포크보다 빨리 길어집니다.

아울러, BP는 동시에 두 개의 포크에 블록을 생산할 수 없습니다. 이런 일을 자행한 BP는 표를 잃어 축출될 것입니다. 이런 규칙 위반자를 자동으로 제거하기 위해 이중 생산이 발생했다는 암호학적 증거(cryptographic evidence)를 사용할 수도 있습니다.

기존 DPOS에, 모든 BP가 모든 블록에 서명하여 동일한 타임스탬프나 높이(height)를 가지는 두 블록에 서명할 수 없게 하는 비잔티움 장애 허용(Byzantine Fault Tolerance, BFT) 기능이 추가되었습니다. 15 명의 BP가 블록에 서명하면 해당 블록은 확정된 것으로 간주됩니다. 어떤 비잔티움 BP가 동일한 타임스탬프나 높이를 가지는 두 블록에 서명하면, 이런 반역에 대한 암호학적 증거가 남게 됩니다. 이 모델 하에서는 1초 이내에 반복 불가능한 합의에 도달할 수 있게 됩니다.

트랜잭션 확정

일반적인 DPOS 블록체인에서 BP 참여율은 100% 가 됩니다. 브로드캐스트 시간 기준으로 평균 0.25초 이내에 99.9%의 확실성으로 트랜잭션이 확정된 것으로 간주할 수 있습니다. DPOS 외에도 EOS.IO는 비동기식 비잔티움 장애 허용(asynchronous BFT, aBFT) 기법을 추가하여 비가역성을 보다 빠르게 달성합니다. aBFT 알고리즘은 1초 이내에 100%의 비가역성을 보장합니다.

지분 증명 기반 트랜잭션(Transaction as Proof of Stake, TaPoS)

EOS.IO 내의 모든 트랜잭션에는 최근 블록 헤더의 해시 일부가 포함되어야 합니다. 이 해시는 두 가지 용도로 사용됩니다:

1. 참조된 블록을 포함하지 않는 포크에서 트랜잭션이 재생되는 것을 방지합니다
2. 특정 사용자와 그의 지분이 특정한 포크 상에 존재한다는 것을 네트워크에 알립니다

시간이 지나면 모든 사용자는 결국 직접적으로 블록체인을 확인하게 되는데, 이로 인해 합법적인 체인에서 발생하는 트랜잭션을 위조해 옮겨올 수 없기 때문에 위조 체인 생성이 어려워집니다.

계정

EOS.IO는 모든 계정이 최대 12문자 길이의 고유한, 사람이 읽을 수 있는 이름으로 생성될 수 있게 허용합니다. 이름은 계정 생성자가 선택합니다. 계정 생성자는 새로운 계정을 생성할 때 이 계정이 자체 RAM을 예약할 토큰을 보유할 때까지 RAM을 예약해야 합니다.

분산된 환경에서 애플리케이션 개발자는 새 사용자를 등록하기 위해 계정 생성을 위한 명목 비용을 지불하게 됩니다. 일반적인 기업 환경에서는 고객을 획득하기 위해 광고나 무료 서비스 등의 형태로 고객 당 상당 금액을 지출하고 있습니다. 이에 비하면 블록체인 계정에 대한 지출 비용은 미미한 수준일 것입니다. 다행히도 다른 애플리케이션에서 이미 등록한 사용자 계정을 또 만들 필요는 없습니다.

액션 & 핸들러

각 계정은 구조화된 액션을 다른 계정으로 보낼 수 있으며, 받은 액션을 처리할 스크립트(handler)를 정의할 수 있습니다. EOS.IO는 각 계정에 자체 액션 핸들러에서 접근할 수 있는 자체 데이터베이스를 제공합니다. 액션 핸들러가 다른 계정으로 액션을 보낼 수도 있습니다. EOS.IO는 액션과 자동화된 액션 핸들러의 조합으로 스마트 컨트랙트를 정의합니다.

병렬 실행을 지원하기 위해 각 계정은 데이터베이스 내에서 원하는 수 만큼의 범위를 정의할 수 있습니다. BP는 병렬 실행을 위해 범위값에 대한 메모리 접근 간 충돌이 생기지 않도록 트랜잭션을 스케줄링합니다.

역할 기반 권한 관리

권한 관리에는 액션이 적합한 권한을 가지고 있는지에 대한 판단이 포함됩니다. 가장 간단한 권한 관리는 트랜잭션이 실행에 필요한 서명을 보유하고 있는지 확인하는 것이지만, 이는 필요한 서명을 사전에 알고 있어야 한다는 뜻입니다. 일반적으로 자격(authority)은 개인이나 그룹에 구속되며, 대개의 경우 구분됩니다. EOS.IO는 누가, 무엇을, 언제 할 수 있는가를 세분화해, 고수준으로 제어할 수 있게 하는 선언적 권한 관리 시스템을 제공합니다.

인증 및 권한 관리를 표준화하고 애플리케이션의 비즈니스 로직과 분리하는 것은 중요합니다. 이렇게 하면 보편적인 방식으로 사용 권한을 관리하는 도구를 개발할 수 있으며, 성능을 최적화하기도 매우 용이합니다.

모든 계정은 다른 계정과 개인 키의 가중치 조합으로 제어할 수 있습니다. 이렇게 하면 사용 권한이 실제로 구성되는 방식을 반영한 계층적 자격 관리 구조가 구축돼 보다 쉽게 계정을 여러 사용자가 제어할 수 있게 됩니다. 이러한 다중 사용자 제어 구조는 단일 대처법 중 보안성을 높이는 데 가장 효과적인 대처법이며, 제대로 사용될 경우 해킹으로 인해 발생하는 도난 위험을 크게 줄일 수 있습니다.

EOS.IO에서는 계정에서 어떤 키 및/또는 계정 조합이 특정 유형의 액션을 다른 계정으로 보낼 수 있는지 정의할 수 있습니다. 예를 들어, 사용자의 소셜 미디어 계정 자체가 하나의 키를 가질 수 있으며 거래소에 접속하기 위한 키를 따로 가질 수 있습니다. 심지어는 키를 할당하지 않고도 다른 계정에 사용자 계정을 대신하여 액션을 보낼 수 있는 권한을 부여할 수도 있습니다.

명명된 권한 수준

EOS.IO에서 계정은 명명된 권한 수준을 정의할 수 있으며, 각 권한 수준은 보다 높은 수준의 명명된 권한에서 파생될 수 있습니다. 명명된 각 권한 수준(permission level)은 자격(authority)을 정의합니다; 자격(authority)은 키 및/혹은 다른 계정의 명명된 권한 수준으로 구성된 임계 다중 서명(threshold multi-signature) 검사입니다. 예를 들어, 계정의 액션에 "친구" 권한 수준을 설정하면 해당 계정의 친구는 이 액션을 동등하게 제어할 수 있습니다.

다른 예로는 Steem 블록체인이 있습니다. Steem 블록체인에는 3개의 명명된 권한 수준이 하드코딩돼 있습니다: 소유자, 활동, 게시. 게시 권한은 오직 소셜 활동(투표나 글 게시)만 허용됩니다. 반면 활동 권한은 소유자를 변경하는 것을 제외한 모든 일이 가능합니다. 소유자 권한은 cold storage를 위한 것이며 모든 일이 가능합니다. EOS.IO는 각 계정 소유자가 자신만의 계층 구조와 액션 그룹을 정의할 수 있게 함으로써 이 개념을 일반화했습니다.








권한 매핑

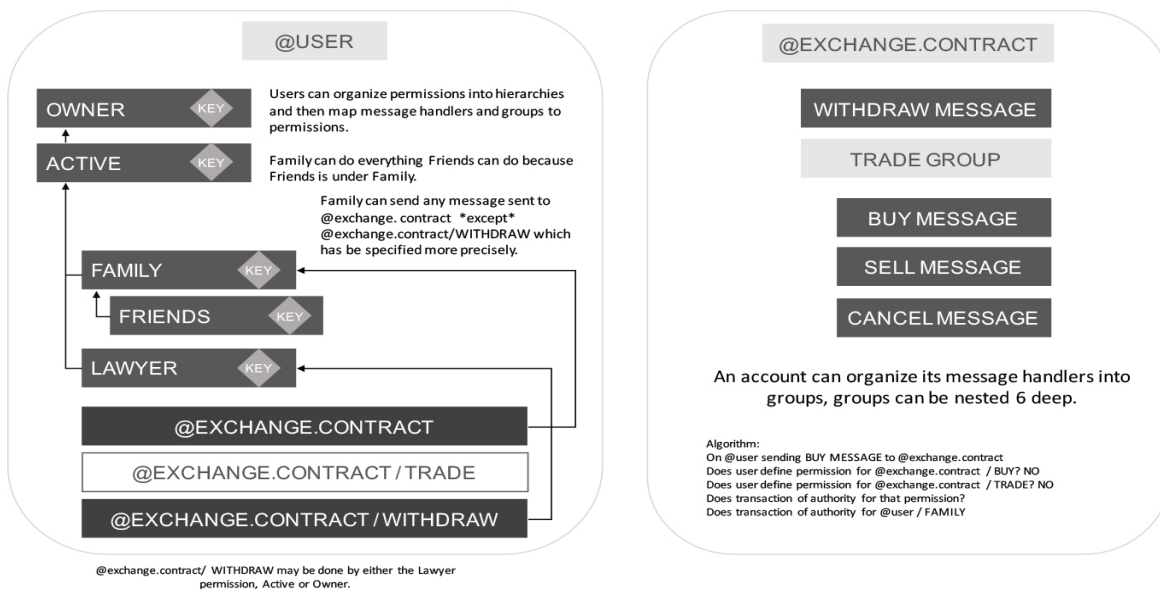
EOS.IO에서 각 계정은 컨트랙트/액션/다른 계정의 컨트랙트와 자신만의 명명된 권한 수준 간 매핑을 정의할 수 있습니다. 예를 들어, 계정 소유자는 자신의 소셜 미디어 애플리케이션을 계정 소유자의 "친구" 권한 그룹에 매핑할 수 있습니다. 이 매핑을 통해 모든 친구는 계정 소유자의 소셜 미디어에 계정 소유자 자격으로 게시할 수 있습니다. 계정 소유자 자격으로 게시하더라도 자신의 키를 사용해 액션에 서명하게 되기 때문에 항상 어떤 친구가 계정을 어떤 방법으로 사용했는지 알 수 있습니다.

권한 평가

@alice가 @bob에게 "액션" 타입의 액션을 보낼 때 EOS.IO는 @alice가 @bob.groupa.subgroup.Action에 대한 권한을 매핑했는지 먼저 검사합니다. 발견되지 않으면 @bob.groupa.subgroup, @bob.groupa 순으로 거슬러 올라가 마지막으로 @bob에 대한 매핑이 검사됩니다. 거기까지도 일치하는 매핑이 없으면 명명된 권한 그룹 @alice.active에 매핑되었다고 가정하게 됩니다.

매핑이 발견되면 이제 다중 서명 과정 및 명명된 권한에 부여된 자격을 사용하여 서명 자격의 유효성을 검증합니다. 실패하면 부모 권한으로 이동하고 최종적으로는 소유자 권한 @alice.owner까지 이동합니다.

Authorities	
Signing	
 STM7xh66F5ZHyfN9u4rNzTioBteZhvWdqDwaR2kR55tBxJCjTr2z	
Owner	
 STM71Tj8quuIqX7aUHZWryYkFAqqTDdpL8FwcoCuzEeKE745mvBY41	
Active	
 STM71Tj8quuIqX7aUHZWryYkFAqqTDdpL8FwcoCuzEeKE745mvBY41	
Posting	
 streemian	1 33.3%
 esteemapp	1 33.3%
 STM89k8Iwp1SR8CBrgAFkd5p5uayTvv57B6Zb4o8enWx8ChjElMTf	1 33.3%
Threshold	1 33.3%
Memo	
 STM75xvQgdvUQ8SFAD8vzFclSkoUdLqzEPbudCXuyokuz1rwzt6v	



기본 권한 그룹

또한 EOS.IO에서는 모든 계정이 모든 일을 할 수 있는 "소유자" 그룹과 소유자 그룹을 변경하는 일을 제외한 모든 일을 할 수 있는 "활동" 그룹을 가질 수 있습니다. 모든 다른 권한 그룹은 "활동" 그룹에서 파생됩니다.

권한의 병렬 평가

권한 평가 프로세스는 "읽기 전용"이며 트랜잭션이 수행한 권한 변경은 블록이 끝날 때까지 적용되지 않습니다. 즉, 모든 트랜잭션에 대한 모든 키와 권한 평가를 병렬로 실행할 수 있으며, 롤백될 수도 있는 값비싼 애플리케이션 로직을 시작하지 않고도 신속하게 권한의 유효성을 검사할 수 있다는 뜻입니다. 마지막으로, 보류중인 트랜잭션을 적용 시에 다시 평가할 필요 없이 수신할 때에만 트랜잭션 권한을 평가할 수 있습니다.

모든 것을 고려하면 권한 검증은 트랜잭션의 유효성을 검사하는 데 필요한 연산의 상당 부분을 차지합니다. 이 기능을 읽기 전용 및 병렬 처리가 가능한 프로세스로 만들면 성능이 크게 향상됩니다.

액션 로그에서 결정론적 상태를 재생성하기 위해 블록체인을 재생해 볼 때에는 권한을 다시 평가할 필요가 없습니다. 이미 정당하다고 알려진 블록에 포함돼 있는 트랜잭션은 이 단계를 건너 뛰어도 무방합니다. 이렇게 되면 끊임없이 커지는 블록체인을 재생할 때의 연산 부하가 크게 줄어듭니다.

강제 지연과 관련된 액션

시간은 보안의 핵심 요소입니다. 대부분의 경우, 개인 키의 도용 여부는 개인 키의 사용 시점 전까지는 알 수 없습니다. 일상적 사용을 위해 인터넷에 계속 연결돼 있는 컴퓨터에 키를 보관해야 하는 애플리케이션의 경우 시간 기반 보안이 특히 더 중요합니다. EOS.IO는 특정 액션이 블록에 포함되기 전까지 대기해야 할 최소 시간을 액션이 수행되기 전에 애플리케이션 개발자가 지정할 수 있습니다. 이 기간 내에 액션을 취소할 수 있습니다.

사용자는 이러한 액션 중 하나가 브로드캐스트될 때 전자 메일이나 텍스트 메시지로 알림을 받을 수 있습니다. 승인한 액션이 아니라면 계정 복구 프로세스를 통해 계정을 복구하고 액션을 철회할 수 있습니다.

얼마나 지연시킬지는 액션이 얼마나 민감한지에 따라 다릅니다. 커피 값을 지불하는 데에는 지연이 필요 없고, 몇 초만 지나도 반복되지 못하지만, 집을 사는 데에는 72시간의 청소 기간이 필요할 수 있습니다. 계정 전체를 새로운 권한으로 이전하는 데에는 최대 30일이 걸릴 수도 있습니다. 다. 정확한 지연 시간은 애플리케이션 개발자와 사용자가 선택합니다.

도난당한 키의 복구

EOS.IO는 사용자가 키를 도난당했을 때 계정을 복구할 수 있는 방법을 제공합니다. 계정 소유자는 지정된 계정 복구 파트너의 승인을 얻어, 지난 30일 동안 활성상태였던 소유자 키를 사용해 자신 계정의 소유자 키를 재설정할 수 있습니다. 계정 복구 파트너는 소유자의 도움이 있어야만 계정을 재설정할 수 있습니다.

해커가 계정 복구 프로세스를 시도해도 이미 계정은 해커에게 "제어"되고 있기 때문에 얻을 수 있는 것은 없습니다. 또한, 복구 프로세스 도중에 계정 복구 파트너는 식별 및 여러 요소를 사용한 인증(전화 및 전자 메일)을 요구할 수 있고, 이로 인해 해커의 정체가 탄로나거나 그 과정에서 어떠한 소득도 얻지 못할 수 있습니다.

이 프로세스는 단순 다중 서명 구성과도 매우 다릅니다. 다중 서명 트랜잭션을 사용하면, 다른 엔티티가 실행된 모든 트랜잭션의 당사자가 됩니다. 반대로 복구 프로세스를 사용하면 복구 파트너는 오직 복구 프로세스의 당사자일 뿐이며, 일반적인 트랜잭션에는 어떠한 권한도 가지지 못합니다. 이를 이용해 관련된 모든 사람의 비용과 법적 책임을 획기적으로 줄일 수 있습니다.

애플리케이션의 결정론적 병렬 실행

블록체인에서 합의는 (재현 가능한)결정론적인 행동 결과입니다. 즉, 모든 병렬 실행은 뮤텍스나 다른 락 기본 요소(lock primitives)를 사용하지 않아야 합니다. 락을 걸지 않기 때문에, 병렬로 실행될 수 있는 트랜잭션이 비 결정론적인 결과를 낳지 않게 보장하는 방법이 필요합니다.

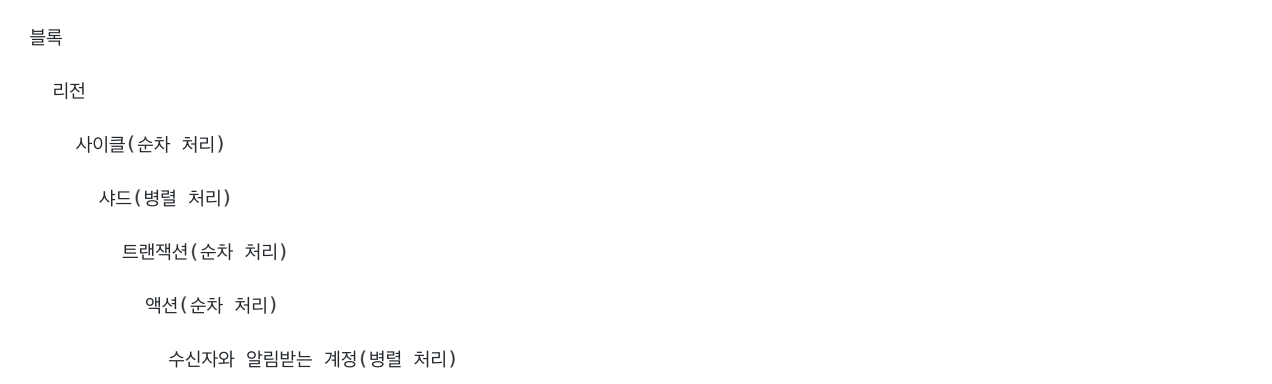
EOS.IO의 2018년 6월 릴리즈에는, 단일 스레드에서 실행되지만 향후 멀티스레드나 병렬 실행에서 필요한 데이터 구조가 포함됩니다.

EOS.IO 기반 블록체인에서, 병렬 실행이 가능해지면 BP는 서로 독립적인 샤드로 액션을 전달해 병렬로 실행시킬 수 있습니다. 실행 스케줄은 BP의 출력이 되며, 스케줄 자체는 결정론적으로 실행되지만 스케줄을 생성하는 프로세스까지 결정론적일 필요는 없습니다. 즉, 블록 프로듀서는 병렬 알고리즘을 사용해 트랜잭션을 스케줄링할 수 있습니다.

부분 병렬 실행(Part of parallel execution)은 스크립트가 새 액션을 생성할 때 즉시 전달되는 것이 아니라 대신 다음 사이클에 전달되도록 예약하는 것을 의미합니다. 수신자가 현재 다른 샤드에서 자신의 상태를 수정하고 있을 가능성이 있기 때문에 즉시 전달할 수 없습니다.

통신 대기시간 최소화

대기시간은 한 계정에서 다른 계정으로 액션을 전송한 다음 응답을 받는 데 걸리는 시간입니다. 목표는 두 계정이 액션 간 0.5초 이내의 대기시간으로 단일 블록 내에서 서로 액션을 교환할 수 있는 것입니다. 이것을 가능하게 하기 위해, EOS.IO는 각 블록을 사이클로 나눕니다. 각 사이클은 샤드로 나뉘며, 각 샤드에는 일련의 트랜잭션이 포함되며, 각 트랜잭션에는 전달될 일련의 액션이 포함됩니다. 이 구조를 그려 보면 계층 단위로 순차적이나 병렬적으로 처리되는 트리 형태가 될 것입니다.



한 사이클에서 생성된 트랜잭션은 후속 사이클이나 블록에 전달될 수 있습니다. BP는 지정된 제한시간이 다하거나 새로 생성돼 전달할 트랜잭션이 없을 때까지 블록에 사이클을 추가할 수 있습니다.

블록의 정적 분석을 이용해 주어진 주기 내에서 두 개의 샤드에 동일한 계정을 수정하는 트랜잭션이 없는지 검증할 수 있습니다. 이러한 불변성이 유지되는 한, 블록은 모든 샤드를 병렬로 실행해 처리할 수 있습니다.

읽기 전용 액션 핸들러

일부 계정은 내부 상태를 수정하지 않고도 액션의 성공/실패를 처리할 수 있습니다. 이러한 경우 특정 사이클 내에서 하나 이상 샤드에 해당 계정의 읽기 전용 액션 핸들러들만 포함돼 있다면 이 핸들러들은 병렬로 실행될 수 있습니다.

여러 계정에 대한 원자적 트랜잭션

액션이 원자적으로 여러 계정에 전달되어 수행되도록 보장해야 할 경우가 있습니다. 이런 경우 두 액션을 모두 하나의 트랜잭션에 배치하고 두 계정이 같은 샤드에 할당하여 작업을 순차적으로 적용합니다.

블록체인 상태의 부분 평가

블록체인의 확장성을 높이려면 구성 요소가 모듈화돼 있어야 합니다. 모든 사람들이 모든 것을 실행할 필요는 없으며, 애플리케이션의 작은 하위 기능만 사용하는 경우에는 특히 더 그렇습니다.

거래소 애플리케이션 개발자는 사용자에게 전체 거래 상태를 표시해주어야 하기 때문에 전체 노드를 실행시킵니다. 이런 거래소 애플리케이션이 소셜 미디어 애플리케이션과 관련된 상태를 알 필요는 없습니다. EOS.IO는 어떤 전체 노드가 일부 애플리케이션만 선택해 실행하도록 설정할 수 있습니다. 다른 컨트랙트의 상태에 연동되지 않는 애플리케이션에서 다른 애플리케이션에 전달되는 액션들은 무시됩니다.

주관적인 최선의 스케줄링

EOS.IO는 BP가 다른 계정에 액션을 전달해야만 한다고 강제하지 않습니다. 각 BP는 트랜잭션을 처리하는 데 필요한 연산 복잡도와 시간을 주관적으로 측정합니다. 이는 트랜잭션이 사용자에게 의해 생성되었건, 스마트 컨트랙트에 의해 자동으로 생성되었건 모두 적용됩니다.

EOS.IO를 채택해 출시된 블록체인을 네트워크 수준에서 보자면 모든 트랜잭션에는 실행하는 WASM 명령어 수에 따라 연산 대역폭 사용 비용이 청구됩니다. 하지만, 각 BP는 자신만의 알고리즘과 측정 기준을 이용해 리소스 사용량을 계산할 수 있습니다. BP가 트랜잭션이나 계정이 과도한 양의 연산 용량을 사용했다고 판단했다면, 간단히 자기 블록을 생성할 때 트랜잭션을 거부할 수 있습니다; 다만, 다른 BP가 유효하다고 판단했다면 거기서 트랜잭션이 처리됩니다.

일반적으로 적어도 하나의 BP가 트랜잭션이 자원 사용량 한계 내에서 유효하다고 간주했다면 다른 BP도 이를 받아들입니다만, 트랜잭션이 그런 BP를 찾는 데 최대 1분까지도 걸릴 수 있습니다.

경우에 따라서 BP는 허용 범위를 벗어난 규모를 가지는 트랜잭션을 포함하는 블록을 만들 수 있습니다. 이때 다음 BP가 블록을 거부할 경우, 동물이 되어 세 번째 BP의 선택에 따르게 됩니다. 큰 블록이 네트워크 전달 지연을 일으킬 때와 다를 바 없습니다. 커뮤니티는 악용 패턴을 발견하여 불량한 BP에게 투표하지 않을 것입니다.

이런 주관적인 계산비용 평가는 블록체인이 무엇인가를 실행하는 데 걸리는 시간을 정확히, 결정론적으로 측정하지 않아도 되게 합니다. 이와 같은 설계로, 합의를 깨지 않으면서 최적화 기회를 크게 증가시키는 명령어를 정확하게 셀 필요가 없어집니다.

지연된 트랜잭션

EOS.IO 소프트웨어는 향후에 실행되도록 트랜잭션을 지연할 수 있습니다. 이를 이용해 연산을 다른 샤드로 이동하거나, 연속된 트랜잭션을 반복해서 실행되는 장기 실행 프로세스로 만들 수 있습니다.

문맥 무관 액션

문맥 무관 액션에는 블록체인 상태에 의존하지 않고 트랜잭션 데이터에만 의존하는 연산이 포함됩니다. 예를 들자면, 서명 확인은 트랜잭션에 서명한 공개키를 확인하기 위해 트랜잭션 데이터와 서명만을 필요로 하는 연산입니다. 이것은 블록체인이 수행해야 하는 가장 비싼 단일 연산 중 하나이지만, 이 연산은 문맥과 무관하기 때문에 병렬로 실행될 수 있습니다.

문맥 무관 액션은 유효성 검사를 수행하기 위해 블록체인 상태에 액세스하지 않는다는 점을 제외하면 다른 사용자 액션과 동일합니다. 이것은 EOS.IO가 서명 확인과 같은 모든 문맥 무관 액션을 병렬로 처리할 수 있게 하고, 보다 더 중요하게는 일반화된 서명 검증을 가능하게 합니다.

문맥 무관 액션의 적용으로 샤딩(sharding) 및 Raiden, Plasma, State Channels 등등의 확장성 관련기술들은 훨씬 더 병렬화 가능해졌고 실용적이 되었습니다. 이것이 개발됨으로 인해 효율적인 블록체인 간 통신과 잠재적으로 무한한 확장이 가능해졌습니다.

토큰 모델 및 자원 사용

주의 사항: 이 백서에 언급된 암호화된 토큰은 EOS.IO를 채택해 출시된 블록체인인 암호화된 토큰을 의미합니다. EOS 토큰 배포와 함께 Ethereum 블록체인에 배포된 ERC-20 호환 토큰을 의미하지 않습니다.

모든 블록체인의 자원은 한정돼 있으며, 남용을 막기 위한 시스템이 필요합니다. EOS.IO를 사용하는 블록체인 상에는 애플리케이션이 소모하는 세 가지 광범위 클래스가 있습니다:

1. 대역폭 및 로그 저장소(디스크)
2. 연산 및 연산 백로그(CPU)
3. 상태 저장소(RAM)

대역폭과 연산에는 순간 사용과 장기 사용이라는 두 가지 구성 요소가 있습니다. 블록체인은 모든 액션에 대한 로그를 보관하며, 이 로그는 모든 전체 노드에 의해 저장되고 다운로드됩니다. 액션의 로그가 있으면 모든 애플리케이션의 상태를 재구성할 수 있습니다.

연산 부채는 액션 로그에서 상태를 재생성하기 위해 수행돼야 하는 연산입니다. 연산 부채가 너무 커지면 블록체인 상태의 스냅샷을 뜨고 난 뒤 블록체인의 히스토리를 버려야 할 필요가 있습니다. 연산 부채가 너무 빨리 커지면 1년분의 트랜잭션을 재생성하는 데 6개월이 걸릴 수도 있습니다. 이러한 일은 치명적이기 때문에 연산 부채는 주의깊게 관리되어야 합니다.

블록체인 상태 저장소는 애플리케이션 로직이 접근할 수 있는 정보입니다. 여기에는 주문서 및 계정 잔고와 같은 정보가 포함됩니다. 상태를 읽어 들이지 않는 애플리케이션이 상태를 저장해서는 안됩니다. 예를 들자면, 블로그 게시글과 댓글은 애플리케이션 로직이 읽어들이지 않습니다. 따라서 그것들은 블록체인 상태에 저장돼서는 안됩니다. 반면, 게시글이나 댓글의 존재 유무 및 투표 수나 다른 속성들같은 경우는 블록체인 상태의 일부로서 저장됩니다.

BP는 대역폭, 연산, 상태에 대한 사용 가능한 용량을 게시합니다. EOS.IO 상에서 각 계정은 지분을 사용한 3일간의 계약을 통해 보유하고 있는 토큰 량에 비례해 사용 가능 용량에서 일정 비율만큼 소비할 수 있습니다. 예를 들어, EOS.IO를 채택한 블록체인이 시작됐을 때 한 계정이 그 블록체인에서 배포 가능한 전체 토큰의 1%를 보유하고 있다면, 그 계정은 상태 저장 용량 전체의 1%를 사용할 수 있습니다.

이미 출시된 블록체인에 EOS.IO를 도입하게 되면 대역폭과 연산 용량은 일시적(미사용 용량을 미래에 사용하기 위해 저장해 둘 수 없음)이기 때문에 분할 예약 기준으로 할당됩니다. EOS.IO가 사용하는 알고리즘은 대역폭 사용 비율을 제한하기 위해 Steem에서 사용하는 알고리즘과 유사합니다.

객관적이지도 주관적인 측정

앞에서 설명했듯이, 연산능력 사용을 예측하는 것은 성능과 최적화에 중요한 영향을 끼칩니다; 따라서, 모든 자원 사용 제약조건은 궁극적으로 주관적이게 되며, BP가 각자의 알고리즘과 예상치를 이용해 적용합니다. 이들은 일반적으로 BP가 커스텀 플러그인을 개발해서 적용합니다.

아울러, 객관적으로 측정해도 무방한 사소한 것들이 있습니다. 전송되는 액션의 숫자나 내부 DB에 저장되는 데이터의 크기는 저렴하므로 객관적 측정이 가능합니다. EOS.IO는 BP가 이런 객관적 측정에 동일한 알고리즘을 쓸 수 있게 해 주지만, 주관적인 척도에 대해서는 더 엄격한 주관적인 척도를 선택할 수 있게도 할 수 있습니다.

수신자 부담

전통적으로 사무실 공간 및 연산 능력, 사업 운영에 필요한 기타 비용을 지불하는 것은 사업자입니다. 고객이 사업자에게서 특정 제품을 구입하면, 해당 제품 판매로 발생한 수익이 사업 운영 비용을 충당하는 데 사용됩니다. 마찬가지로 어떤 웹사이트도 호스팅 비용 명목으로 방문자에게 소액 결제를 강제하지 않습니다. 그렇기 때문에, 분산 애플리케이션은 고객에게 블록체인 사용 자체에 대한 비용을 직접 지불하도록 하게 해선 안 됩니다. EOS.IO를 채택해 출시된 블록체인은 블록체인 자체의 사용을 위해 사용자가 직접 지불할 필요가 없기 때문에 사업자가 자사 제품에 대한 자체 수익 창출 전략을 결정하는 것을 제한하거나 막지 않습니다.

수신자가 지불할 수 있기는 하지만, EOS.IO에서는 발신자가 대역폭 및 계산, 저장 비용을 지불할 수도 있습니다. 이를 이용하면 애플리케이션 개발자는 애플리케이션에 가장 잘 맞는 방법을 선택할 수 있습니다. 많은 경우 발신자 부담은 자체 배급 시스템(rationing system)을 구현하지 않으려 하는 애플리케이션 개발자들의 수고를 줄여줍니다. 애플리케이션 개발자는 대역폭과 연산용량을 사용자에게 위임한 다음 "발신자 부담" 모델로 사용하도록 할 수 있습니다. 최종 사용자의 관점에서 보면 무료이지만, 블록체인의 관점에서 보면 발신자 부담입니다.

용량 위임

EOS.IO를 채택한 블록체인의 토큰 보유자는 사용 가능한 대역폭의 전부나 일부를 즉시 사용할 필요가 없을 때 사용하지 않는 대역폭을 다른 사람들에게 위임하거나 대여할 수 있습니다; EOS.IO를 채택한 블록체인의 BP는 이와같은 용량 위임이 발생할 경우 이를 인지하고 그에 따라 대역폭을 할당합니다.

트랜잭션 비용을 토큰 가치와 구분

EOS.IO의 주요 장점 중 하나는 애플리케이션이 사용 가능한 대역폭 양이 토큰 가격과 완전히 독립적이라는 것입니다. 만약 애플리케이션 소유자가 EOS.IO를 채택한 블록체인에 적당한 양의 토큰을 보유하고 있다면, 고정된 상태 및 대역폭 사용량 내에서 애플리케이션을 무기한으로 실행할 수 있습니다. 이런 경우 개발자와 사용자는 토큰 시장의 가격 변동에 영향받지 않기 때문에 가격 공급에 의존하지 않게 됩니다. 즉, EOS.IO를 채택한 블록체인은 BP들이 자연스럽게 토큰의 가치와 상관없이 토큰 당 사용 가능한 대역폭 및 연산력, 스토리지를 늘릴 수 있게 합니다. EOS.IO를 채택한 블록체인은 BP가 블록을 생산할 때마다 보상으로 BP에게 토큰을 부여합니다. 토큰의 가치는 BP가 구매할 수 있는 대역폭, 저장 공간 및 연산력에 영향을 줍니다; 이 모델은 증가하는 토큰 가치를 자연적으로 레버리지해 네트워크 성능을 향상시킵니다.

상태 저장 비용

대역폭과 연산력을 위임할 수 있지만, 애플리케이션 상태를 저장하기 위해서 애플리케이션 개발자는 상태가 삭제되기 전까지 토큰을 보유하고 있어야 합니다. 상태가 삭제되지 않는다면, 토큰은 실질적으로 유통에서 배제됩니다.

블록 보상

EOS.IO를 채택한 블록체인은 블록을 생산할 때 마다 BP에게 새로운 토큰을 보상으로 지급합니다. 이렇게 생성된 토큰의 수는 모든 블록 프로듀서가 제시한 원하는 보상의 중간값을 따릅니다. EOS.IO 소프트웨어는 또한 연간 보상으로 발행되는 토큰 양이 전체의 5%를 초과하지 않도록 보상 상한을 설정할 수 있습니다.

워커 제안 시스템

EOS.IO 소프트웨어를 기반으로 한 블록체인 시스템에서 토큰 보유자는 BP를 선출하는 것 외에도 커뮤니티에 이익이 될 수 있는 다양한 워커 제안에 대해 투표할 수 있습니다. 투표에서 승리한 제안은 토큰 인플레이션이 발생할 때 BP가 받게 되는 일부의 보상만큼을 제외한 토큰에서 일정 퍼센트를 받게 됩니다. 이들 제안은 각 애플리케이션이 토큰 보유자에게 받은 표 수에 비례하여 토큰을 보상받게 되며, 최대로는 자신들이 역할을 수행하기 위해 필요하다고 제안한 만큼까지 받을 수 있습니다. 선출된 제안은 토큰 보유자가 새로이 선출한 제안으로 대체될 수 있습니다.

워커 제안을 구현하는 시스템 컨트랙트는 2018년 6월에 출시되지 못할 수 있지만, 자금 조달 메커니즘은 BP 보상이 시작됨과 동시에 시작될 것입니다. 워커 제안 시스템이 WASM으로 구현될 것이기 때문에 향후 포크 없이 적용될 수 있습니다.

운영(Governance)

운영은 커뮤니티 구성원에 의해 수행되는 다음과 같은 절차입니다:

1. 소프트웨어 알고리즘만으로 완전히 해결할 수 없는 집단 행동의 주관적 문제에 대한 합의를 도출하는 것
2. 도출된 합의를 수행하는 것
3. 헌법 수정안을 통해 운영 규칙 자체를 변경하는 것

EOS.IO 기반 블록체인은 현재 BP가 가진 영향을 효과적으로 행사할 수 있게끔 운영 절차를 구현하였습니다. 종전의 블록체인은 정의된 운영 절차 없이 임시적이고 비공식적이며 종종 논란이 되는 운영 절차에 의지하여 예측할 수 없는 결과를 초래했습니다.

EOS.IO 기반 블록체인에서 권한은 BP에게 권한을 위함한 토큰 보유자에게서 비롯된 것입니다. BP는 계정을 동결하고, 결함이 있는 애플리케이션을 업데이트하고, 기본 프로토콜에 대한 하드 포크 변경을 제안할 수 있는 제한적이지 확인된 권한을 부여받습니다.

EOS.IO에는 BP 선출기능이 내장돼 있습니다. 블록체인에 어떠한 변경이 이루어지려면, BP들의 승인을 얻어야 합니다. 만약 BP가 토큰 보유자가 원하는 결정을 내리지 않는다면, 그 BP는 표를 잃고 축출될 수 있습니다. 만약 BP가 토큰 보유자들의 허락 없이 변경을 일으킨다면, BP가 아닌 다른 모든 전체 노드 확인자(예: 거래소)가 이 변경을 거부할 수 있습니다.

계정 동결

이따금씩 스마트 컨트랙트는 비정상적이거나 예측하지 못했던 방식으로 작동하고 의도한 대로 수행되지 못할 수 있습니다; 다른 경우, 애플리케이션이나 계정이 비정상적인 양의 자원을 소모하게 하는 취약점을 발견할 수도 있습니다. 불가피하게 이와 같은 문제들이 발생하면, BP는 이런 상황을 바로잡을 수 있습니다.

모든 블록체인에서 BP는 블록에 어떤 트랜잭션들이 포함되는지 선택할 수 있는 권한을 가지며, 이를 이용해 계정을 동결할 수 있습니다.

EOS.IO를 채택한 블록체인은 활동중인 BP 중 15/21 표결을 통한 계정 동결 과정을 거쳐 이 권한을 공식화합니다. 만약 BP가 이 권한을 남용할 경우 표를 잃을 수 있으며, 계정은 동결에서 풀려납니다.

계정 코드 변경

다른 모든 대응책이 실패하여 "막을 수 없는 애플리케이션"이 예상치 못한 방식으로 작동하는 경우, EOS.IO를 채택한 블록체인은 BP가 전체 블록체인을 하드포크하지 않고 해당 계정의 코드만을 변경할 수 있게 합니다. 계정을 동결하는 과정과 유사하게, 이러한 코드 변경에는 선출된 BP의 15/21 표결이 필요합니다.

헌법

EOS.IO는 블록체인이 P2P 서비스 계약이나, 서명한 사용자 간 구속력을 가지는 컨트랙트, 이른바 "헌법"을 제정할 수 있게 합니다. 헌법은 단순히 코드만으로 강제될 수 없는 사용자 간 의무를 정의하고, 서로간에 상호 합의된 규칙을 통해 관할권을 확립하고 규정을 세워 분쟁 해결을 쉽도록 합니다. 네트워크 상의 모든 트랜잭션은 헌법의 해시를 서명의 일부로 포함시켜서 서명자를 컨트랙트에 명시적으로 구속되도록 합니다.

또한 이 헌법은 사람이 읽을 수 있도록 소스 코드 프로토콜의 의도를 정의합니다. 이 의도는 오류가 발생했을 때 버그와 기능의 차이를 구별하고, 수정된 내용이 적절한지 부적절한지 커뮤니티가 판정할 수 있도록 하는 데 사용됩니다.

프로토콜과 헌법의 업그레이드

EOS.IO는 정식 소스 코드 및 구성에 의해 정의된 대로 프로토콜을 업데이트할 수 있도록 다음과 같은 프로세스를 정의합니다:

1. BP는 헌법의 변경을 제안하고 15/21 표결을 거칩니다.
2. BP는 향후 30일간 새 헌법의 15/21 승인을 유지합니다
3. 모든 사용자는 새 헌법에 동의하였음을 명시해야만 향후 트랜잭션을 처리할 수 있습니다
4. BP는 헌법의 변경을 반영하기 위해 소스 코드를 변경하고 새 헌법의 해시를 이용해 블록체인에 제안합니다
5. BP는 향후 30일간 새 코드의 15/21 승인을 유지합니다
6. 코드 변경은 7일 후 효력이 발생되기 때문에, BP가 아닌 모든 노드는 그 기간 내에 업그레이드할 수 있습니다
7. 새 코드로 업그레이드하지 않는 모든 노드는 자동으로 종료됩니다

EOS.IO의 기본 설정에 따르면 블록체인에 새 기능을 추가해 업데이트하는 데 2~3개월이 걸립니다. 다만, 구조 변경 없이 중요하지 않은 버그를 수정하는 데에는 1~2개월이 걸립니다.

긴급 상황에서의 변경

만약 현재 사용자에게 해를 끼치는 버그나 보안 취약점을 수정하기 위해 소프트웨어 변경이 필요하다면 BP는 이 과정을 가속시킬 수 있습니다. 일반적으로는 새 기능을 도입하거나 무해한 버그를 수정하기 위해 업데이트를 가속화하는 것은 헌법에 위배될 수 있습니다.

스크립트 & 가상 머신

EOS.IO는 인증된 메시지(액션)를 계정에 전달하는 과정을 조율하는 최초이자 최고의 플랫폼입니다. 스크립트 언어와 가상 머신의 세부 내용은 EOS.IO 기술 설계와는 대부분 독립돼 구현됐습니다. 결정론적이고 정확한 샌드박스 처리를 제공하는 모든 언어 또는 가상 머신은 EOS.IO API와 통합될 수 있습니다.

스키마로 정의된 액션

계정간에 전송되는 모든 액션은 블록체인 합의 상태의 일부인 스키마에 의해 정의됩니다. 이 스키마는 액션의 이진 및 JSON 표현 간 원활한 변환을 허용합니다.

스키마로 정의된 DB

DB 상태 역시 유사한 스키마를 이용해 정의됩니다. 이것으로 모든 애플리케이션이 저장하는 모든 데이터가 사람이 읽을 수 있도록 JSON으로 변환될 수 있으면서도 바이너리로 저장되고 조작돼 효율성을 높일 수 있습니다.

일반적인 멀티 인덱스 DB API

스마트 컨트랙트를 개발하기 위해서는 데이터를 추적, 저장, 검색하기 위해 DB 스키마를 정의할 필요가 있습니다. 개발자는 일반적으로 여러 필드로 정렬되거나 인덱싱된 동일한 데이터를 필요로 하며, 모든 인덱스 간 일관성(consistency)이 유지돼야 합니다.

애플리케이션에서 인증을 분리

병렬화를 극대화하고 트랜잭션 로그에서 애플리케이션 상태를 재생성하는 연산 부채를 최소화하기 위해서 EOS.IO는 유효성 검증 로직을 세 섹션으로 분할하였습니다:

1. 액션의 내부 일관성 검증
2. 모든 선행조건의 타당성 검증
3. 애플리케이션의 상태 수정

액션의 내부 일관성 검증은 읽기 전용이며 블록체인 상태에 액세스할 필요가 없습니다. 즉, 최대한 병렬적으로 실행 가능하다는 뜻입니다. 필요 잔고 등과 같은 선행조건을 검증하는 것 역시 읽기 전용이며 병렬화의 수혜를 입을 수 있습니다. 오직 애플리케이션의 상태 수정만이 쓰기가 필요하며, 각 애플리케이션 간에 순차적으로 처리돼야 합니다.

인증은 액션이 적용될 수 있는지를 검사하는 읽기 전용 프로세스입니다. 애플리케이션이 실제로 이 작업을 수행합니다. 실시간으로 두 연산(인증, 작업)이 모두 수행돼야 하지만 일단 트랜잭션이 블록체인에 포함되고 나면 더이상 인증 작업을 수행할 필요는 없습니다.

블록체인 간 통신

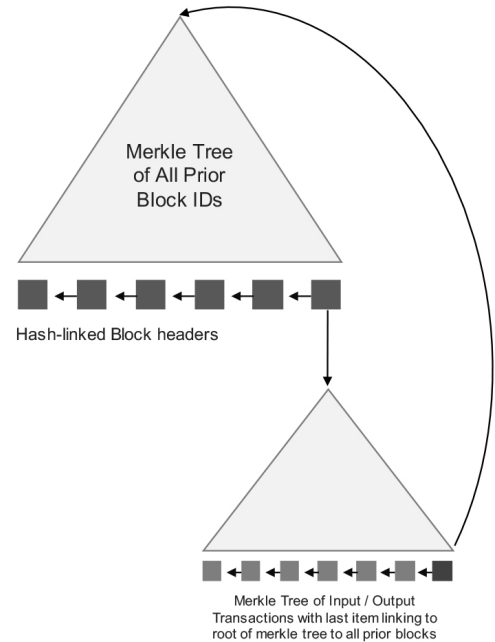
EOS.IO는 블록체인 간 통신을 지원하도록 설계되었습니다. 이것은 액션의 존재 증명과 액션의 순서 증명을 생성하기 쉽게 함으로서 가능해졌습니다. 이 증명과 액션 전송을 위해 고안된 애플리케이션 구조가 결합되면 블록체인 간 통신과 증명 타당성 검증이 애플리케이션 개발자로부터 은닉돼, 고수준의 추상화를 개발자에게 제공합니다.

경량 클라이언트 검증(Light Client Validation, LCV)를 위한 머클 증명

클라이언트가 모든 트랜잭션을 처리하지 않아도 된다면 다른 블록체인과 통합하는 것이 훨씬 쉬워집니다. 결국 거래소는 내/외부로 전달하는 것에만 관심이 있고 그 이외에는 관심이 없습니다. 또한, 거래 체인이 BP를 전적으로 의지하는 대신 예치에 대한 경량의 머클 증명을 사용할 수 있다면 이상적일 것입니다. 적어도 블록체인 내의 BP는 다른 블록체인과 동기화할 때 오버헤드를 가능한 한 최소화하기를 원합니다.

LCV의 목표는 상대적으로 경량의 데이터들을 추적하는 사람이 검증을 위해 사용할, 상대적으로 경량의 존재증명을 생성하는 것입니다. 목표는 특정 트랜잭션이 특정 블록에 포함돼 있고, 해당 블록이 특정 블록체인의 검증된 내역에 포함돼 있음을 증명하는 것입니다.

Bitcoin은 모든 노드가 연간 4MB에 달하는 블록 헤더의 전체 내역에 액세스할 수 있다고 가정하고 트랜잭션의 유효성 검사를 지원합니다. 초당 10회의 트랜잭션에서는 유효한 증명에 512바이트 가량이 필요합니다. 이것은 블록 생산 간격이 10분인 블록체인에서는 잘 동작하지만, 0.5초 생산 간격의 블록체인에서는 절대 "가벼운" 것이 아닙니다.



EOS.IO에서는 트랜잭션이 포함된 시점을 지나 반복할 수 없게 확정된 블록 헤더를 가진 사람이라면 누구나 경량 증명이 가능합니다. 그림과 같이, 링크된 해시 구조(hash-linked structure)를 사용하면 1024 바이트 미만의 데이터로 모든 트랜잭션의 존재 증명이 가능합니다.

블록체인 내의 블록에 대한 ID와 신뢰할 수 있는 비가역 블록의 헤더가 주어진다면, 블록이 블록체인에 포함돼 있음을 증명할 수 있습니다. 이 증명은 체인 내에 N개의 블록이 존재한다고 할 때 그 경로를 $\text{ceil}(\log_2(N))$ 로 요약합니다. SHA256 해시 기법으로 요약한다면, 체인 내의 10억개의 블록에서 특정한 블록의 존재를 864 바이트만으로 증명해 낼 수 있습니다.

블록을 생산할 때 이런 증명을 가능하게 하는 적절한 링크된 해시 구조를 생성하는 추가 오버헤드가 거의 없기 때문에 이런 식으로 블록을 생산하지 말아야 할 이유가 없습니다.

다른 체인에서 만든 증명을 검증할 때 많은 시간/공간/대역폭 최적화가 가능합니다. 모든 블록 헤더(매년 420MB)를 추적하면 증명 크기를 작게 가져갈 수 있습니다. 최신 헤더만을 추적하면 최소 장기 저장 공간과 증명 크기 간 절충이 필요합니다. 또한, 블록체인은 과거 증명의 중간 해시만을 기억하는 방식으로 지연 평가 접근방식을 사용할 수도 있습니다. 새 증명은 현재 알려진 희소 트리에 대한 링크만을 포함하면 됩니다. 정확히 어떤 접근 방식을 사용할 지는 필연적으로 머클 증명에 의해 참조된 트랜잭션을 포함하는 외부 블록의 비율에 따라 달라지게 됩니다.

상호 연관성의 밀도가 일정 수준을 넘어서면, 하나의 체인에 다른 체인의 전체 블록 히스토리가 포함되어 다함께 증명하지 않아도 될 수 있도록 단순화하는 것이 보다 효율적입니다. 성능상의 이유로 인해, 체인 간 증명의 빈도를 최소화하는 것이 이상적입니다.

체인 간 통신의 지연 시간

다른 외부 블록체인과 통신할 때, BP는 트랜잭션이 다른 블록체인 내에서 반복될 수 없게 확정되었다는 사실을 100% 확인할 때까지 기다린 이후에 이 트랜잭션을 정당하다고 판정해야 합니다. EOS.IO 기반 블록체인에 비잔티움 장애 허용 비가역성이 추가된 블록 생산간격 0.5초의 DPOS를 적용하게 되면 약 0.5초가 소요됩니다. 비가역성을 확인하기 위해 기다리지 않는 블록체인의 BP는 이후에 취소될 수 있는 입금을 승인하는 거래소처럼 돼 버리기 때문에 블록체인에서 발생하는 합의의 타당성에 영향을 줄 것입니다. EOS.IO는 DPOS와 aBFT를 사용해 빠른 비가역성을 제공합니다.

완전성 증명

외부 블록체인에서 머클 증명을 사용할 때, 처리된 모든 트랜잭션이 타당하다는 것을 인지하는 것과, 어떠한 트랜잭션도 건너뛰거나 생략하지 않았다는 점을 인지하는 것 사이에는 상당한 차이가 있습니다. 최근의 모든 트랜잭션이 알려졌다는 것을 증명하는 것은 불가능하지만, 트랜잭션 이력에 빈틈이 없다는 것을 증명하는 것은 가능합니다. EOS.IO는 모든 계정 간 전달되는 모든 액션에 일련번호를 할당하여 이를 용이하게 합니다. 사용자는 일련번호를 사용해 특정 계정을 대상으로 한 모든 액션이 성공적으로 처리되었고, 순서대로 처리되었음을 입증할 수 있습니다.

증인 격리(분리)

증인 격리(Segregated Witness, SegWit)의 개념은 트랜잭션 서명은 트랜잭션이 블록 체인에 불변하게 포함된 이후에는 무관해진다는 것에 서 비롯합니다. 일단 불변성을 획득하고 나면 서명 데이터를 잘라낸다 해도, 여전히 다른 사람들은 현재 상태를 파생시킬 수 있습니다. 서명은 대부분의 트랜잭션에서 많은 부분을 차지하기 때문에 SegWit은 디스크 사용량과 동기화 시간을 많이 절약해 줍니다.

이와 같은 개념은 블록체인 간 통신에서 사용되는 머클 증명에도 적용될 수 있습니다. 일단 증명이 이뤄져 블록체인에 비가역적으로 로깅되고 나면, 증명을 위해 SHA256으로 만들어 낸 2KB의 해시는 알맞은 블록체인 상태를 생성해 내는 데 더이상 필요하지 않습니다. 블록체인 간 통신에 이 방법을 적용하면 일반 서명을 절약하는 것보다 32배 더 절약할 수 있습니다.

SegWit의 또다른 예로는 Steem 블로그 게시물에 있습니다. 이 모델에서 게시물에는 오직 SHA256(블로그 내용) 해시만이 포함되며, 블로그 내용은 격리된 증인 데이터에 포함됩니다. BP는 게시물이 존재하고 주어진 해시가 맞는지 확인하지만, 현재 상태를 블록체인 로그에서 복구하기 위해 블로그 내용을 저장할 필요가 없습니다. 이로서 내용을 저장하지 않고도 내용이 이미 알려진 내용에서 변하지 않았다는 것을 증명할 수 있게 합니다.

결론

EOS.IO는 입증된 개념과 모범 사례를 활용한 경험을 토대로 설계되었으며, 블록체인 기술이 근본적으로 발전하였음을 상징합니다. EOS.IO는 전 세계적 규모 블록체인 상에서 분산 애플리케이션을 쉽게 배포하고 운영할 수 있게 하는 전체적인 청사진의 일부입니다.

번역 정보

- 원문 : <https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md>
- 번역 기준 리비전 : 68047653b0d96deb62f43e2113531c86cdc4247e