

아이오에스티(IOST)

해당 번역본은 이용자가 디지털자산 관련 정보를 편리하게 확인할 수 있도록
단순 참고용으로 번역된 것이며, 투자 권유를 목적으로 하지 않습니다.
그 내용이 정확하지 않을 수 있으므로 정확한 정보 습득을 위해서는
원문을 참고하시거나 원문 작성 주체 측에 문의하시기 바랍니다.
두나무는 해당 번역본 내용의 진실성 및 정확성을 보장하지 않고,
제공된 정보에 의한 투자 결과에 대하여 어떠한 책임도 지지 않으며,
투자 손실은 투자자에게 귀속됩니다.

인터넷 오브 서비스: 온라인 서비스를 위한 차세대, 안전하고 고도로 확장 가능한 생태 계

인터넷 오브 서비스 재단

2017년 12월 31일

초록

최근 가상자산 시장의 과열에도 불구하고, 기반이 되는 블록체인 기술은 여전히 초기 단계에 있으며 대중적 채택과는 거리가 멀다.

현재 블록체인 기술에서 잘 알려진 문제 중 하나는 확장성이다.

대량의 트랜잭션을 신속하게 처리할 수 있는 능력이 없다면, 페이스북, 아마존, 가상자산 거래소와 같은 고사 용량 서비스는 블록체인에 배포하는 것이 사실상 불가능하다.

본 논문에서는 수평적 확장성과 높은 트랜잭션 처리량을 제공하도록 설계된 혁신적이고 안전한 블록체인 패 러다임인 인터넷 오브 서비스("IOS")를 제안한다.

우리는 새로운 샤딩 아키텍처와 합의 메커니즘을 구현함으로써, IOS 시스템이 초당 최대 100,000건의 안 전한 트랜잭션을 처리할 수 있도록 하였다.

이 연구는 다음과 같은 기여를 한다. 다음 항목들을 소개한다:

1. **효율적인 분산 샤딩 (EDS)** - 클라이언트-서버 무작위성 수집 메커니즘과 암호학적 정렬을 통한 리 더 선출을 결합하여 샤드를 충분히 크고 강한 편향 저항성을 갖도록 만드는 혁신적인 샤딩 방식.

2. **TransEpoch** - 트랜잭션 운용 가능성을 유지하면서 에폭 전환 중 검증자-샤드 할당을 안전하게 수행.
3. **Atomix** - 비잔틴 환경에서 트랜잭션 원자성을 보장하는 새로운 2단계 샤드 간 원자 커밋 프로토콜.
4. **신뢰도 증명 (PoB)** - 단일 샤드 단위에서 트랜잭션 처리량을 최대화하면서 시스템의 안전성과 생존성을 보장하는 신뢰 우선 접근 방식의 획기적인 비잔틴 합의 프로토콜.
5. **마이크로 상태 블록 (MSB)** - 검증자의 저장 및 부트스트래핑 비용을 최소화하는 새로운 메커니즘.

참고: IOS는 현재 개발 중인 프로젝트이다. 연구는 활발히 진행 중이며, 본 논문의 새로운 버전은 <http://iost.io> 에서 업데이트될 예정이다. 의견이나 제안은 team@iost.io 로 연락 바란다.

목차

- [배경](#)
- [관련 연구](#)
 - [상태 머신 복제](#)
 - [비트코인과 작업 증명](#)
 - [지분 증명](#)
- [블록체인 아키텍처](#)
- [분산 무작위성 프로토콜](#)
- [효율적인 분산 샤딩](#)
 - [알고리즘 - 리더 선출 및 백업 프로토콜](#)
 - [분석](#)
- [에폭 전환 중 운용성](#)
 - [노드-샤드 전환 할당 알고리즘 - TransEpoch](#)
 - [분석](#)
- [샤드 간 트랜잭션](#)
- [합의 메커니즘](#)
 - [토큰 및 동기 부여](#)
 - [신뢰도 증명](#)
- [블록체인 저장소 정리](#)
 - [알고리즘 - MSB 생성 프로토콜](#)
 - [분석](#)
- [참고문헌](#)

배경

과도한 수수료, 프라이버시 침해, 사기, 그리고 검열은 중앙화된 온라인 서비스 제공업체와의 일상적인 상호 작용에서 흔히 발생하는 문제들이다.

중앙집중화에 따른 이러한 널리 인식된 문제들을 고려할 때, 이를 해결하고자 하는 다양한 블록체인 기술들이 2008년 비트코인 [15]의 출범 이후 개발되었다.

SteemIt [29], Bitshares [30], Syscoin [21]과 같은 특화된 프로젝트들과 Ethereum [28], EOS [11]과 같은 보다 범용적인 프로젝트들이 그 예이다.

그러나 이러한 시도들 대부분은 특정 애플리케이션에 지나치게 특화되어 있거나 낮은 트랜잭션 처리량으로 부담을 준다.

유연성과 트랜잭션 처리량에 있어 이러한 한계들로 인해, 페이스북이나 아마존과 같은 대형 서비스를 블록체인에 올리는 것은 불가능하며, 가상자산 거래소처럼 더 복잡한 것은 말할 것도 없다.

확장성 문제의 핵심은 이러한 기존 블록체인 기술들의 근본적인 설계 — 즉, 그것들의 합의 프로토콜과 블록체인 아키텍처 — 에 있다.

기존 대부분의 블록체인 기술들은 확장하는 과정에서 두 가지 주요 문제에 직면한다:

- a) 모든 전체 노드는 참여하기 위해 전체 원장을 저장해야 한다;
- b) 네트워크에 참여하는 모든 노드는 모든 트랜잭션을 처리해야 한다.

모든 참여 노드가 본질적으로 동일한 작업을 수행하기 때문에, 시스템이 처리할 수 있는 트랜잭션 수는 단일 노드의 처리량을 초과하지 못한다.

더욱이, 블록체인의 크기가 커짐에 따라 노드가 네트워크에 완전하게 참여하기 위해 필요한 저장 공간, 대역 폭, 계산 자원의 요구사항과 비용이 증가한다.

채굴 비용의 증가는 결국 네트워크 참여를 소수에게만 허용되는 특권으로 만들게 되고, 이는 곧바로 중앙집중화 문제로 되돌아가게 만든다.

IOS는 이러한 공백을 메우기 위해 설계되었다.

우리의 비전에서 IOS는 서비스 지향 생태계를 지원하기 위한 네트워크 인프라를 제공하는 차세대 블록체인 기술이다.

IOS 플랫폼은 사용자에게 온라인 서비스와 디지털 상품을 완전히 탈중앙화된 방식으로 교환할 수 있도록 제공할 뿐만 아니라, 개발자가 대규모 사용자 수를 지원할 수 있는 대형 dApp을 배포할 수 있도록 한다.

효율적인 분산 샤딩("EDS")과 신뢰 우선 합의 접근 방식과 같은 일련의 획기적인 혁신들을 통해, 우리는 보안을 보장하면서 시스템의 처리량을 엄청나게 증가시킬 수 있었다.

우리는 샤딩 기법을 기반으로 EDS를 개발하였다.

샤딩은 병렬 트랜잭션 처리를 가능하게 하기 위해 분산 시스템과 데이터베이스에서 널리 채택된다.

컴퓨터 과학의 고전적인 "분할 정복" 원리에 영감을 받은 샤딩은 전체 IOS 네트워크를 샤드라 불리는 일정 개수의 하위 공간으로 분할하는 기술이다.

각 샤드는 자체 합의 프로토콜을 병렬로 실행하는 소형 네트워크로 간주할 수 있다.

전체 네트워크가 동일한 트랜잭션 집합을 검증하는 대신, 트랜잭션의 하위 집합은 다양한 합의 그룹들에 의해 동시에 처리될 수 있다.

따라서, 네트워크의 크기와 트랜잭션 수가 빠르게 증가하더라도 시스템의 처리량은 상당히 향상될 수 있다.

게다가, 네트워크가 균일하게 분할되도록 보장하기 위해 우리는 샤딩 과정에 편향되지 않고 투명한 무작위성을 도입하기 위한 편향 저항 분산 무작위성 프로토콜을 개발하였다.

EDS와 함께, IOS는 고성능과 유연성을 갖춘 대규모 dApp 배포를 가능하게 하는 매우 강력한 기술 도구 모음을 포함하고 있다.

이것은 개발자들이 전통적인 독점 온라인 서비스 제공업체들의 대응물부터, 이전에는 불가능하다고 여겨졌던 완전히 새로운 비즈니스 모델들까지 다양한 제품을 구축할 수 있도록 한다.

물론, IOS 네트워크의 크기가 비교적 작을 경우, 이러한 서비스를 실행하는 것은 꽤 비용이 많이 들 수 있다. 그러나 네트워크 내의 노드 수와 자원이 증가함에 따라, 이러한 대규모 dApp 운영 비용은 크게 감소할 수 있다.

추가적으로, 사이버 공격 회피, 높은 수준의 데이터 보안, 불변성 등 여러 이점들이 있다.

IOS 블록체인의 개발 과정에서 우리는 이전의 시도들로부터 배우기 위해 현재 사용 가능한 모든 솔루션을 철저히 조사하였다.

관련 연구

상태 머신 복제

간단히 말하면, 블록체인 기술은 상태 머신 복제 프로토콜이다.

모든 상태 머신 복제 프로토콜은 두 가지 중요한 속성을 만족해야 한다:

1. **안전성**, 즉 네트워크 내의 모든 서버가 동일한 트랜잭션 기록을 가지고 있어야 한다;
2. **생존성**, 즉 클라이언트의 트랜잭션이 빠르게 제출되고 로그에 기록되어야 한다.

상태 머신 복제를 달성하는 방식에는 근본적으로 두 가지가 있다: 고전 스타일 합의와 블록체인 스타일 합의 [18].

고전 스타일 합의는 일반적으로 Paxos 유사 알고리즘을 적용하며, 시스템이 사전에 알고 있는 합의 노드들이 존재하는 허가형 환경에서 사용된다.

이러한 응용의 예로는 아마존과 같은 소프트웨어 회사의 서버들을 들 수 있으며, 그들의 서버는 집합적으로 고전적 방식을 이용하여 정보를 복제하고 저장하며, 고전 알고리즘은 데이터의 순서를 합의하기 위한 기본 요소를 구성한다.

비트코인과 작업 증명

사토시 나카모토는 허가가 필요 없는 환경에서 합의를 형성하는 솔루션으로 비트코인을 처음 제안하였다.

즉, 어떤 노드든지 사전 정보 없이 자유롭게 네트워크에 참여하고 떠날 수 있는 환경이다.

비트코인을 기반으로 하는 네트워크인 블록체인은 서버들(비트코인에서는 채굴자라 불리는)에게 경제적 인센티브를 제공함으로써 사람의 개입 없이 분산 시스템의 확장성을 개선하였다.

비트코인 네트워크의 채굴자들은 일정 난이도의 부분 해시 충돌을 계산함으로써 합의를 형성한다.

누적 난이도가 가장 큰 체인은 다른 노드들에 의해 합의 결과로 인정된다.

이 솔루션은 **작업 증명(Proof-of-Work, PoW)**이라 불리며, 본질적으로는 네트워크 내 모든 노드가 계산 능력을 제공함으로써 인센티브를 획득하고, 시스템 전체의 트랜잭션 순서를 결정하는 방식이다.

PoW의 장점 중 하나는 허가가 필요 없는 환경에서 시빌 공격에 대응할 수 있다는 것이다 [15].

확장성과 보안 면에서의 진보에도 불구하고, 비트코인은 몇 가지 주요 단점들을 지닌다:

(1) 다른 현대 가상자산들과 달리, 구성상 트랜잭션을 확정하는 데 1시간 이상이 소요된다;
(2) 비트코인 네트워크 위에 다양한 애플리케이션을 개발하기 어렵다;
(3) 합의 메커니즘이 너무 많은 에너지를 낭비한다. 즉, 하루에 200만 달러 이상의 전력 비용이 발생한다.
더 중요한 점은, 이전 연구들에 따르면 비트코인 스타일 블록체인은 보안을 유지하기 위해 충분히 큰 간격을 가져야 한다는 것이다 [16][17].
따라서 비트코인은 현재 중앙집중형 시스템을 대체하여 일상적인 애플리케이션과 대량의 트랜잭션을 지원하는 데 적합하지 않다.

지분 증명

지분 증명의 개념은 온라인 블록체인 포럼 [31]에서 처음 논의되었고, PPcoin [22], PeerCoins [23], Nxt [5]와 같은 몇몇 가상자산에서 채택되었다.
PoS의 기본 아이디어는 단위 지분당 한 표로, 각 검증자는 더 많은 지분을 가질수록 더 높은 투표 권한을 갖게 된다.
따라서 검증자는 전체 블록체인 네트워크를 해치는 경제적 인센티브가 없다.
공격자 입장에서는 대다수 지분을 소유해야 하므로, 공격 비용이 매우 크다.
초기 개발 단계에서는, 지분 증명 합의 메커니즘은 “nothing-at-stake” 공격에 취약한 것으로 알려졌는데, 이는 서버들이 동시에 여러 블록에 투표할 수 있으며 수렴할 인센티브가 없기 때문에 블록체인의 보안이 훼손되는 문제다.
후속 연구에서는 slasher [3]를 이용하여 문제를 해결하였으며, 이는 규칙 위반 노드에게 벌칙을 부과한다.
여러 다른 연구들도 지분 증명의 특수한 적용 사례로 설명된다 [1-3,26][12][10].
PoS는 복제 상태 머신 프로토콜의 생존성을 충족시키지만, 중앙화 및 보안과 같은 문제에 여전히 직면해 있다.
예를 들어, 더 많은 토큰을 보유한 검증자는 새로운 블록을 생성할 가능성이 더 높고 더 부유해지기 때문에, 중앙화 문제로 이어질 수 있다.
또한, 이전 연구에 따르면 지분 증명 프로토콜이 증명 가능하게 안전하고 견고하게 구성되기 위해서는 해당 토큰이 자주 교환되지 않아야 하며 [19], 이는 보안을 유지하기 위해 지분 증명의 처리량에 상한이 있을 수 있음을 시사한다.

블록체인 아키텍처

IOSChain의 인프라는 비트코인이나 이더리움과 같은 기존의 잘 알려진 블록체인과 유사하며, 노드는 gossip 프로토콜을 통해 데이터를 전파한다.
시스템은 그 데이터와 상태를 샤드로 분할한다.
각 노드는 시스템의 하나의 샤드에 대해 책임을 진다.
사용되지 않은 트랜잭션(UTXO)은 해당 샤드에 속한 노드의 메모리에 저장된다.
이로 인해 몇 가지 새로운 과제가 발생한다.

- 시스템을 어떻게 샤드로 나눌 것인가.
- 각 샤드에서 어떻게 합의에 도달할 것인가.
- 샤드 간 트랜잭션을 어떻게 수행할 것인가.

위 과제들을 공정하고 안전하게 해결하기 위해, 우리는 여러 무작위 연산을 수행해야 한다. 예를 들어, 노드를 샤드에 할당하거나 각 샤드에서 리더를 선출하는 것이다.

따라서 우리는 먼저 위조 불가능하고 편향되지 않은(균등 무작위) 분산 난수 생성 프로토콜을 설계해야 한다. 이 분산 난수 생성 프로토콜을 통해 위 과제들을 하나씩 해결할 수 있다.

이 논문의 다음 부분에서는 이러한 과제들을 해결하기 위해 사용된 기술과 방법들을 제시한다.

- 4장에서는 **분산 무작위성 프로토콜 (DRP)**을 제시한다. 이 프로토콜은 악의적 노드의 비율이 사전에 정의된 임계값 이하일 때 위조 불가능하고 편향되지 않는다. DRP에서 생성된 난수는 시스템을 샤드로 분할하고, 노드를 서로 다른 샤드에 할당하며, 각 샤드에서 리더를 선출하는 데 사용된다.
- 5장에서는 **효율적인 분산 샤딩 (EDS)**을 제시한다. 이는 DRP와 암호학적 정렬을 통한 VRF 기반 리더 선출을 결합하여, 충분히 크고 강한 편향 저항성을 가진 샤드(상태를 기록하고 트랜잭션을 처리하는 검증자 집합)를 형성하는 새로운 방식이다.
- 6장에서는 **TransEpoch**을 제시한다. 이는 에폭 전환 중에도 시스템의 운용 가능성을 유지하면서 검증자를 샤드에 할당하는 안전한 프로토콜이다.
- 7장에서는 **Atomix**를 제시한다. 이는 비잔틴 환경에서 트랜잭션의 원자성을 보장하는 새로운 2단계 샤드 간 원자 커밋 프로토콜이다.
- 8장에서는 **신뢰도 증명 (PoB)**을 제시한다. 이는 신뢰 우선 접근 방식을 채택한 새로운 비잔틴 합의 프로토콜로, 시스템의 안전성과 생존성을 보장하면서도 단일 샤드 규모로 트랜잭션 처리량을 최대화한다.
- 9장에서는 **마이크로 상태 블록 (MSB)**을 제시한다. 이는 검증자의 저장 및 부트스트래핑 비용을 최소화하는 새로운 메커니즘이다.

분산 무작위성 프로토콜

작업 증명 기반 메커니즘 [13]이나 신뢰할 수 있는 비콘 [6]과 같은 전통적인 무작위성 생성 방식은 계산 자원의 낭비와 중앙화에 대한 우려를 가진다.

암호학적 도구를 사용하여 분산 무작위 수를 생성하는 것이 바람직하며, 이는 자원을 절약할 뿐만 아니라 증명 가능하게 안전하다.

IOSChain에서 노드-샤드 할당 및 리더 선출을 위한 분산 무작위 수를 생성하기 위해 여러 알고리즘이 존재한다.

여기에서는 우리가 알기로는 IOSChain 시나리오의 요구사항에 가장 잘 부합하는 알고리즘을 제시한다.

IOSChain에서 분산 무작위 수 생성기는 다음과 같은 요구사항을 가진다.

1. 특정 비율 내의 부정직한 참여자(클라이언트 및 서버 포함)에 대해 저항성을 가져야 한다. 자세히 말하자면, 부정직한 참여자의 비율이 임계값 이하일 경우 시스템은 진행 가능하며, 진행 중에 나쁜 일이 발생하지 않아야 한다.
2. 최종 무작위 수는 위조 불가능하고 편향되지 않아야 한다(균등 무작위), 단, 무시할 수 있는 확률은 예외로 한다.
3. 부정직한 참여자는 자신에게 유리한 무작위 수를 생성하기 위해 여러 번 시도할 수 없어야 하며, 복수의 부정직한 참여자가 공모하더라도 마찬가지여야 한다.
4. 제3자는 해당 출력이 프로토콜을 성실히 실행한 결과임을 검증할 수 있어야 한다(즉, 위의 모든 요구사항을 충족함을 검증할 수 있어야 한다).

이러한 요구사항을 달성하기 위해, 우리는 분산 무작위성 프로토콜(Distributed Randomness Protocol, DRP) [24]이라 불리는 클라이언트-서버 프로토콜의 사용을 제안한다.

이 프로토콜에서 클라이언트는 일련의 서버들과 통신하며, 비대화형 영지식 증명(NIZK)과 공개 검증 가능한 비밀 분산(PVSS)을 통해 위조 불가능하고 균등한 무작위 값을 생성한다.

하나의 프로토콜 실행에서, 프로토콜이 완료되고 최종 무작위 출력이 공개되기 전까지는 어떤 엔터티도 해당 출력에 대한 어떠한 정보도 알 수 없으며, 이는 부정직한 클라이언트가 자신에게 유리한 무작위 수를 생성하기 위해 여러 번 시도하는 것을 방지한다.

이 프로토콜은 두 단계로 구성된다 — 무작위성 생성과 무작위성 검증. 작동 방식은 다음과 같다.

초기에, 클라이언트는 무작위로 생성된 균형 잡힌 그룹 구성을 포함하는 메시지를 모든 서버에 브로드캐스트하여 프로토콜 실행을 시작한다.

첫 번째 단계에서는, 각 서버가 무작위 입력 값을 생성하고 같은 그룹 내 다른 멤버들만을 위한 세어를 PVSS를 사용하여 생성한다.

모든 서버로부터 NIZK [25] 증명이 포함된 암호화된 세어를 수신(또는 타임아웃)한 후, 클라이언트는 각 그룹에서 일부 서버 입력을 선택한다.

이것은 클라이언트가 각 그룹의 비밀과 프로토콜 출력을 고정할 수 있게 한다.

두 번째 단계에서는, 클라이언트가 집단 서명(CoSi) [25]의 글로벌 실행에서 입력 값에 대한 승인 서명을 수신하자마자, 서버들은 세어를 복호화하여 클라이언트에게 전송한다.

그런 다음, 클라이언트는 복원된 그룹 비밀들을 결합하여 최종 무작위 출력을 도출한다.

효율적인 분산 샤딩

위에서 제시한 분산 무작위성 프로토콜(DRP)을 통해 효율적인 분산 샤딩을 구현하는 것은 어렵지 않다.

그러나 이 프로토콜은 검증자들에 의해 집단적으로 수행되므로, 악의적 노드나 실패한 노드가 없을 경우에만 잘 작동한다.

따라서 우리는 악의적 또는 실패한 노드가 존재하는 시나리오를 위한 백업 프로토콜을 설계해야 한다.

이 문제를 해결하기 위해, 우리는 리더를 선출하기 위한 솔루션으로 Algorand [9]와 Omniledger [8]를 사용하는 방법을 제안한다.

알고리즘 - 백업 프로토콜을 포함한 리더 선출

입력값(Inputs):

1. v 는 뷰 카운터(view counter)이다.
2. i 는 검증자(validator)이다.
3. sk_i 는 검증자 i 의 개인 키(private key)이다.
4. e 는 현재 에포크(epoch)이다.
5. Δ 는 동기화 한계(synchrony bound)이다.

출력값(Output):

DRP를 실행할 최소값의 유효한 복권(lottery)을 가진 검증자 i .

1. 각 에포크 e 마다, 각 검증자 i 는 자신의 뷰 v 와 개인 키 sk_i 를 이용하여 검증 가능한 난수 함수 (Verifiable Random Function, VRF)를 통해 복권 $lottery_{i,e,v}$ 를 계산한다.
2. 그 후, 시간 한계 Δ 동안 검증자들은 이 복권 값을 서로 고십(gossip)한다. 각 검증자는 고십 과정에서 최소값 상위 3개의 복권을 수집한다.
3. 시간 한계 Δ 가 지난 후, 각 검증자는 자신이 본 복권 중 최소값의 유효한 복권을 확정한다.
4. 최소값의 유효한 복권에 해당하는 검증자가 리더로 선출되며, 두 번째 및 세 번째 최소값 복권에 해당하는 검증자들은 백업 리더 풀(back-up leaders pool)에 포함된다.
5. 선출된 검증자가 DRP를 성공적으로 실행하면, 그 결과값 rnd_e 를 올바른 증명(correctness proof)과 함께 다른 모든 검증자에게 브로드캐스트한다.
6. 각 검증자 i 는 rnd_e 를 이용해 순열(permutation)을 계산하고 결과를 동일한 크기의 m 개의 버킷(bucket)으로 나눈다. 이렇게 해서 노드에서 샤드로의 매핑이 결정된다.
7. 시간 한계 Δ 이후에도 선출된 리더가 DRP를 시작하지 못할 경우, 검증자들은 해당 실행을 실패로 표시하고, 해당 리더를 에포크 e 의 나머지 기간 동안 제외한다. 이 경우 백업 리더가 DRP를 실행하게 된다. 두 백업 리더 모두 연속으로 실패할 경우, 복권 절차는 1단계로 되돌아가며 전체 프로토콜이 다시 실행된다.

분석

리더 선출 메커니즘은 섹션 4에서 설명된 것과 동일한 필요한 속성을 제공한다. 각 검증자는 하나의 에포크 e 내에서 각 뷰 v 마다 단 하나의 유효한 복권(lottery)만 생성할 수 있다. DRP 설계는 확장성을 보장한다. 개인 키 sk_i 가 비밀로 유지되므로, VRF의 출력은 예측할 수 없다. 동기화 시간 한계 Δ 가 주어졌을 때, 해당 복권은 Δ 시간 내에 다른 모든 검증자들에게 전달된다. 악의적인 노드가 복권에 당첨되더라도 임의의 행동을 수행할 수는 없으며, DRP 프로토콜을 실행하여 협력하거나 에포크를 실패 상태로 만드는 두 가지 선택지 중 하나만 선택할 수 있다. 악의적이거나 비정상적인 상황이 발생할 경우, 해당 악의적 노드는 에포크의 나머지 기간 동안 참여에서 제외된다.

에폭 전환 중 운용성

샤드 구성 방식에는 정적 구성과 몇 가지 다른 롤링 구성들이 존재한다.

IOSChain은 동적 롤링 방식을 사용하며, 각 에폭 e 마다 검증자들을 배치 단위로 교체한다.

이 구성 방식은 IOSChain에 유희 기간을 제공하며, 충분한 수의 검증자가 적절히 부트스트랩된 후에야 네트워크가 트랜잭션 처리를 시작할 수 있게 된다.

많은 블록체인 설계는 이 기간 동안 시스템이 어떻게 운용 가능한지를 보장할 것인지를 문제를 고려하지 않았다.

이 전환 중 핵심 요인은 배치 크기이며, 이는 시스템의 안전성과 매우 밀접하게 관련되어 있다.

교체 배치 크기가 증가할수록, 남아 있는 정직한 검증자 수가 합의에 도달하기에 충분하지 않게 되어 위험이 증가한다.

교체 배치 크기가 증가할 때의 또 다른 단점은 다운로드 및 부트스트래핑 정보가 네트워크에 부하를 증가시킨다는 것이다.

우리는 최대 1/3의 악의적 노드가 존재한다는 위협 모델을 가정하므로, 교체 배치의 최대 크기는 1/3 노드 미

만이어야 한다.

전환/유휴 단계 동안 완전한 운용성을 유지하기 위해, 우리는 일부 검증자를 선택하여 교체하고 새로운 멤버로 대체하는 방법을 사용한다 [8,24].

이 방법은 Omniledger의 접근 방식 [8]에 기반한다.

이 방식은 새롭게 참여한 노드들이 과거 데이터를 다운로드하고 부트스트래핑하는 동안, 남아 있는 검증자들이 계속 서비스를 제공할 수 있도록 해준다.

다음은 노드-샤드 전환 할당 프로토콜 TransEpoch이다.

노드-샤드 전환 할당 알고리즘 - TransEpoch

입력값(Inputs):

1. n 은 전체 노드의 개수이다.
 2. m 은 각 샤드의 크기이다.
 3. k 는 스왑아웃 배치 크기(swap-out batch size)로, 주어진 에폭 내에서 한 번에 교체될 검증자 수를 의미한다.
-

출력값(Outputs):

1. 집합
 $k = \log(n / m)$
 2. 각 샤드 j 에 대해,
 - DRP에서 생성된 난수 출력을 사용하여 두 개의 시드 s_j, rnd_e 및 so, rnd_e 를 생성한다.
 - s_j, rnd_e 와 so, rnd_e 를 이용해 순열 $\pi_{j,e}$ 및 $\pi_{o,e}$ 를 구하고, 노드들을 크기 k 의 버킷(bucket)으로 나눈다. 이 과정을 통해 노드와 스왑 배치 간의 매핑이 결정된다.
 - $\pi_{j,e}$ 는 현재 노드에 사용되고, $\pi_{o,e}$ 는 새로 합류하는 노드에 사용된다.
 - 각 배치는 Δ 동안 대기한 후 스왑을 시작한다.
-

분석

위에서 제시한 알고리즘에서는, 각 샤드 전환에 있어 비잔틴 장애 허용(BFT) 합의의 안전성을 보장한다. 그 이유는 두 가지 측면에 있다.

첫째, 우리는 그룹 크기를 보장하였다. 최소한 샤드 크기의 $2/3$ 이상의 검증자가 합의를 수행한다.

둘째, 적대자에 대한 안전성 역시 보장되며, 이는 에폭 단위 무작위성이 검증자 배치 순열을 생성하는 데 사용되기 때문이다.

샤드 간 트랜잭션

샤드 간 트랜잭션을 지원하는 메커니즘은 시스템에서 매우 중요하다. 트랜잭션이 샤드를 넘나들 가능성이 높기 때문이다.

우리는 샤드 간 원자성을 보장하기 위해 비잔틴 샤드 원자 커밋(Atomix) 프로토콜을 도입한다.

이 프로토콜은 이중 지불을 방지하며 트랜잭션의 일관성을 유지한다.

우리의 설계는 Omniledger 알고리즘 [8]의 변형이다.

우리는 먼저 UTXO 상태 모델에서 Atomix를 제시한다.

이전 연구는 UTXO 모델이 지원된다면 Atomix가 스마트 계약 또한 샤드 간 트랜잭션 메커니즘에 의해 지원될 수 있음을 입증하였다 [27].

요약하자면, 샤드 A의 노드 a에서 샤드 B의 노드 b로의 샤드 간 트랜잭션이 발생할 때, 알고리즘은 다음과 같이 작동한다:

-
1. **샤드 A** 내에서 TX를 생성하고, 모든 노드가 해당 트랜잭션을 검증한다.
 2. 만약 **A**의 모든 노드가 TX를 승인하면, 트랜잭션은 **A**의 블록체인에 기록된다. 동시에 클라이언트는 트랜잭션을 승인했다는 증명을 gossip하며, **a**의 자금을 UTXO로 잠그고 이를 **B**로 전송한다.
 - 만약 **A**의 노드들이 TX를 거부하면, 자금은 **a**에게 반환된다.
 3. **A**의 블록체인은 TX를 **B**의 블록체인에 커밋하고, 수신자 샤드의 노드들이 TX를 검증한다.
 - 만약 **B**의 노드들이 TX를 거부하면, 자금은 **a**에게 반환된다.
 4. 만약 **B**의 모든 노드가 TX를 승인하면, 자금은 **b**에게 지급된다.
 - 만약 모든 노드가 거부하면, 자금은 **a**에게 반환된다.

합의 메커니즘

토큰과 동기 부여

IOS 시스템에서, IOS 토큰은 다른 블록체인 시스템의 토큰과 마찬가지로 모든 트랜잭션과 수수료의 교환 수단으로 사용된다.

더 중요한 것은, IOS는 사용자의 신뢰도 점수를 계산하는 데에도 중요한 역할을 한다.

모든 IOS 토큰은 제네시스 블록에서 생성될 것이다.

IOS 생태계에서, IOS 토큰은 다음과 같은 용도로 사용될 수 있다:

- **지불**: 상인 또는 다른 커뮤니티 구성원이 제공하는 서비스 및 상품에 대한 지불.
- **수수료**: 스마트 계약 실행, 메시지 및 트랜잭션 처리, 저장 공간, 연산 자원 등 일반 생태계에서 공유되는 자원을 사용하는 것에 대한 보상으로 검증자에게 지급되는 수수료.
수수료는 검증자에게 인센티브를 제공하고, 악의적인 사용자가 스마트 계약을 과도하게 배포하여 커뮤니티의 이익을 훼손하는 것을 방지한다.
- **신뢰도**: IOS 토큰은 사용자의 신뢰도 점수를 계산하는 데 사용된다(다음 섹션에서 설명될 예정이다).

또한 IOS 생태계의 구성원으로서, 각 사용자는 트랜잭션을 검증하고 자원을 제공함으로써 IOS 토큰을 획득할 수 있다 (예: 스마트 계약 실행, 저장 공간 제공 등).

앞서 언급했듯이, 기존 지분 증명(Proof-of-Stake) 합의 메커니즘이 직면한 주요 문제 중 하나는 중앙화 경향이다.

이러한 위험을 완화하기 위해, 우리는 사용자의 커뮤니티 기여도를 측정하고 IOSChain의 지속적 발전에 대한 기여를 장려하는 수단으로 Servi를 도입한다.

Servi는 다음과 같은 속성을 갖는다:

- **양도 불가능:** Servi는 교환 수단으로 설계되지 않았기 때문에, 어떤 방식으로든 거래되거나 교환될 수 없다.
- **자동 소멸:** 블록이 검증된 후, 시스템은 해당 검증자가 보유한 Servi 잔액을 자동으로 삭제한다. 이 방식으로, 높은 신뢰도를 가진 노드들이 블록 검증을 번갈아 수행할 수 있게 되어 공정한 블록 생성 프로세스를 보장한다.
- **자동 발행:** Servi는 커뮤니티 서비스 제공, 다른 개체가 제공한 서비스 평가, 기타 특별한 기여와 같은 행위를 수행한 후 자동으로 생성되어 사용자 계정에 입금된다.

신뢰도 증명

기존 블록체인 시스템은 샤드 크기에 따라 안전성과 처리량 사이에 내재된 트레이드오프가 존재한다.

작은 샤드가 많은 시스템은 성능은 뛰어나지만 악의적 행위자에 대한 복원력이 떨어지며, 그 반대도 마찬가지이다.

우리는 안전성을 유지하면서 처리량을 증가시킬 수 있는 방식으로 이 트레이드오프를 해결하기 위해, IOSChain을 위한 혁신적인 신뢰도 증명(Proof-of-Believability, PoB) 합의 프로토콜을 제안한다. PoB는 노드가 악의적으로 행동할 확률이 무시할 정도로 낮음을 보장하면서도, 단일 샤드 크기로 트랜잭션 처리량을 크게 증가시킨다.

신뢰도 증명 합의 프로토콜은 샤드 내 "신뢰 우선(Believable-First)" 접근 방식을 사용한다.

이 프로토콜은 모든 검증자를 두 그룹으로 나눈다: 신뢰도 리그(believable league)와 일반 리그(normal league).

신뢰도 검증자들은 1단계에서 트랜잭션을 빠르게 처리한다.

그 후, 일반 검증자들은 2단계에서 샘플링과 검증을 수행하여 종결성과 검증 가능성을 제공한다.

노드가 신뢰도 리그에 선출될 확률은 신뢰도 점수에 따라 결정되며, 이 점수는 토큰 잔액, 커뮤니티 기여도, 평가 등 여러 요인으로 계산된다.

신뢰도 점수가 높은 자가 신뢰도 리그에 선출될 가능성이 더 크다.

신뢰도 검증자들은 커밋된 트랜잭션의 집합과 순서를 결정하고, 해당 순서대로 트랜잭션을 처리하는 절차를 따른다.

신뢰도 검증자들은 소규모 그룹 — 그룹당 하나의 검증자 — 를 구성한다.

트랜잭션은 이들 신뢰도 검증자들 사이에 무작위로 분배된다.

그 결과, 이들은 매우 낮은 지연으로 소규모 블록을 생성한다.

그러나 단 하나의 노드가 검증을 수행하기 때문에 보안 문제가 발생할 수 있다.

그 결과, 일부 잘못된 트랜잭션이 악의적 검증자에 의해 커밋될 수 있다.

이 보안 문제를 해결하기 위해, 우리는 일반 검증자들이 트랜잭션을 샘플링하고 불일치를 감지하는 샘플링 확률 p 를 명시한다.

검증자가 악의적으로 행동한 것으로 감지되면, 해당 검증자는 시스템 내에서 모든 토큰과 평판을 잃게 되며,

피해 사용자들은 손실에 대해 보상받는다.

신뢰 우선 접근 방식은 단 하나의 (신뢰 가능한) 검증자만 검증을 수행하며, 이 검증자가 악의적일 가능성이 낮기 때문에 트랜잭션 처리를 매우 빠르게 만든다.

IOS 시스템에서는 샤딩 정책 파일이 신뢰도 리그와 일반 리그의 크기 및 샘플링 확률 p 를 각각 명시한다.

에폭이 시작될 때, 모든 검증자는 분산 무작위성 생성 프로토콜을 사용하여 샤드에 할당된다.

그들의 상태는 해당 샤드의 마지막 마이크로 상태 블록(MSB)에서 부트스트랩된다.

신뢰도 점수에 따라, 검증자는 샤드 내에서 신뢰 그룹(작음) 또는 일반 그룹(큼)에 할당된다.

1단계에서는, 신뢰도 리그가 처리한 트랜잭션이 낙관적으로(validated optimistically) 검증된 블록을 생성한다.

이 블록들은 동시에 실행되는 일반 리그에 의해 샘플링 재검증의 입력으로 사용된다.

일반 리그는 여러 낙관적 처리 그룹으로부터 입력을 결합하기도 한다.

이것은 시스템의 처리량을 극대화할 수 있다.

트랜잭션이 성공적으로 검증되면, 그것들은 종결된 블록에 포함되고, 해당 샤드의 블록체인에 추가되며, 최종적으로 MSB에 포함된다.

그러나 일반 리그가 불일치를 감지하는 경우, 해당 검증된 트랜잭션은 블록체인에서 제외되고, 잘못된 블록에 서명한 검증자는 감지되어 책임을 지게 된다.

우리는 검증자가 어떤 상황에서도 악의적으로 행동할 유인을 가지지 않도록 강력한 처벌 체계를 설계하였다.

검증자가 악의적으로 행동한 것으로 판명되면, 해당 검증자는 시스템에서 모든 토큰과 평판을 잃으며, 그가 이전에 검증한 모든 트랜잭션은 다시 점검된다.

잘못을 저지를 유인이 극히 적고, 검증의 보안에 대해 정량적인 확신을 가질 수 있기 때문에, 클라이언트는 실시간 처리 속도와 함께 확신을 얻을 수 있다.

일반 리그는 ByzCoin [7]에 기반한 비잔틴 합의 방식을 실행한다.

이는 수천 명의 합의 그룹 구성원까지 확장 가능하기 때문이다.

ByzCoin은 다중 서명 [20]을 사용하는 확장 가능한 암호학적 기초인 집단 서명(CoSi) [25]을 사용하여 PBFT [4]와 같은 전통적인 합의 알고리즘을 확장 가능하게 만든다.

ByzCoin은 성능을 위해 멀티캐스트 트리를 사용하여 블록을 배포하고, 장애 허용을 위해 별형 토폴로지(star topology)로 되돌아간다.

이는 일부 악의적인 노드가 샤드에 존재하더라도, 샤드 내의 모든 정직한 구성원들이 특정한 공통 작업 순서에 합의하도록 보장하면서, 안전성과 생존성을 보장한다.

견고함을 보장하기 위해, 우리는 신뢰 우선 프로토콜 내에 폴백(fall-back) 체계를 사용한다.

샤드가 일시적인 다운타임이거나 생태계의 부트스트래핑 단계에 있어 충분한 신뢰도 검증자들이 리그를 형성하지 못하는 경우, 이중 리그 위원회는 단일 리그로 폴백한다.

모든 트랜잭션은 PBFT 합의 프로토콜을 따르는 일반 리그에 의해 직접 처리된다.

블록체인 저장소 정리

현재 블록체인이 직면한 또 다른 문제는 블록체인 저장소 크기의 급격한 증가이다 [8].

이로 인해 새로운 검증자에게는 부트스트래핑에 큰 부담이 발생한다.

블록체인은 처음부터의 이력을 저장하는 동일한 패턴을 따른다.

그러나 이는 고처리량 블록체인 시스템에서는 저장소가 폭증할 수 있기 때문에 중요한 문제이다.

우리는 검증자의 저장소 및 부트스트래핑 비용을 최소화하기 위해, 샤드 블록체인의 전체 상태를 요약하는 **블록체인 저장소 정리(blockchain storage pruning)** 접근 방식을 사용한다.

우리는 **마이크로 상태 블록(MSB)**을 사용하며, 이는 State Block [8]에 기반한다.

아래에 MSB 생성 프로토콜을 제시한다.

알고리즘 - MSB 생성 프로토콜

입력값(Inputs):

- e 는 현재 에포크(epoch)이다.
- j 는 현재 샤드(shard)이다.

출력값(Output):

에포크 e 에서 샤드 j 의 마이크로 상태 블록(micro state block) $msb_{j,e}$

1. 에포크 e 가 종료될 때, 샤드 리더는 해당 에포크의 모든 트랜잭션을 머클 트리(Merkle tree) [14]에 저장한다.
 2. 샤드 리더는 머클 트리의 루트(root)를 해싱하여 h 로 표기하고, 이를 $Header(msb_{j,e})$ 에 포함시킨다.
 3. 검증자들은 대기 중인 일반 블록이 없을 때 $Header(msb_{j,e})$ 에 대한 합의(consensus)를 수행한다.
 4. $Header(msb_{j,e})$ 의 정확성이 검증되면, 샤드 리더는 승인된 헤더를 샤드의 블록체인에 저장한다.
 5. 에포크 $e + 1$ 의 종료 시점에서, 모든 노드는 $msb_{j,e-1}$ 의 본문(body)을 삭제하고 에포크 e 의 일반 블록만 유지한다.
-

분석

트랜잭션은 과거 블록체인을 참조함으로써 확인된다.

IOSChain의 각 샤드는 과거 MSB 헤더만 저장하고 블록체인 상태는 여러 샤드에 걸쳐 분산되어 있기 때문에, 클라이언트는 블록 내의 확인 정보를 제공함으로써 과거 트랜잭션의 존재를 증명할 수 없다.

우리는 저장 책임을 과거 블록체인으로부터 클라이언트로 이전함으로써 이 문제를 완화한다.

최신 에폭의 블록은 유지되므로, 클라이언트는 다음 에폭 동안 에폭 e 에서 검증된 트랜잭션에 대해 해당 샤드의 검증자에게 존재 증명을 생성해줄 것을 요청할 수 있다.

검증자들은 본질적으로 MSB에 대한 상위 수준의 체인을 생성하고 있으며, 이는 하나의 에폭의 MSB에서 다른 에폭의 MSB로 건너뛸 수 있도록 한다.

이 MSB 체인은 본문이 포함된 최신 MSB와 이전의 모든 MSB 헤더들을 유지한다.

이는 과거 트랜잭션을 검증하려는 클라이언트에게 기준점이 필요하므로 중요하다.

우리는 MSB가 증명 크기를 줄이기 위해 일반 블록의 헤더에 대한 여러 단계의 백포인터를 포함할 수 있음을

언급한다.

MSB를 통해, 새로운 검증자의 부트스트래핑과 충돌한 검증자의 최신 상태 동기화가 효율적으로 이루어진다.

검증자는 마지막으로 유효한 MSB에서 시작하여 블록체인의 마지막 부분만 재실행하면 되므로, 첫 번째 블록부터 또는 충돌 시점부터 전체 이력을 다시 재생할 필요가 없다.

비트코인이 IOSChain에 배포될 경우, 현재 부트스트래핑에 필요한 대역폭 비용은 두 자릿수(100배) 수준으로 줄어든 것이다.

이는 IOSChain에 새로운 샤드가 도입될 때 매우 중요하다.

무작위 샤드 할당 메커니즘으로 인해, 검증자들은 주기적으로 샤드를 변경하며 자주 업데이트되어야 하며, 이는 블록체인 저장소 정리 기술로부터 많은 혜택을 받는다.

참고문헌

- [1] Iddo Bentov, Ariel Gabizon, and Alex Mizrahi. 2016. Cryptocurrencies Without Proof of Work. In Lecture Notes in Computer Science. 142–157.
- [2] Iddo Bentov, Charles Lee, Alex Mizrahi, and Meni Rosenfeld. 2014. Proof of Activity. ACM SIGMETRICS Performance Evaluation Review 42, 3 (2014), 34–37.
- [3] Vitalik Buterin. 2014. Slasher: a punitive proof of stake algorithm. Retrieved January 9, 2018 from <https://blog.ethereum.org/2014/01/15/slasher-a-punitive-proof-of-stake-algorithm/>
- [4] Miguel Oom Temudo de Castro. 2000. Practical Byzantine Fault Tolerance.
- [5] Nxt Community. Nxt Whitepaper. Retrieved January 9, 2018 from <https://braveneecoin.com/assets/Whitepapers/NxtWhitepaper-v122-rev4.pdf>
- [6] George Danezis and Sarah Meiklejohn. 2016. Centrally Banked Cryptocurrencies. In Proceedings 2016 Network and Distributed System Security Symposium. DOI:<http://doi.org/10.14722/ndss.2016.23187>
- [7] E. Kokoris-Kogias, P. Jovanovic, N. Gailly, I. Khoffi, L. Gasser, and B. Ford. 2016. Enhancing Bitcoin Security and Performance with Strong Consistency via Collective Signing. In 25th USENIX Conference on Security Symposium.
- [8] Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser†, Nicolas Gailly, Ewa Syta, Bryan Ford. 2017. OmniLedger: A Secure, Scale-Out, Decentralized Ledger via Sharding.

- [9] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. 2017. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In Proceedings of the 26th Symposium on Operating Systems Principles – SOSP '17. DOI:<https://doi.org/10.1145/3132747.3132757>
- [10] G Maxwell And. 2015. On Stake and Consensus. Retrieved January 9, 2018 from <https://download.wpsoftware.net/bitcoin/pos.pdf>
- [11] Ian Grigg. EOS – An Introduction. eos.io. Retrieved from https://eos.io/documents/EOS_An_Introduction.pdf
- [12] J. Kwon. 2014. Tendermint: Consensus without mining. Retrieved January 9, 2018 from <http://tendermint.com/docs/tendermint.pdf>
- [13] Loi Luu, Viswesh Narayanan, Chaodong Zheng, Kunal Baweja, Seth Gilbert, and Prateek Saxena. 2016. A Secure Sharding Protocol For Open Blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security – CCS'16. DOI:<https://doi.org/10.1145/2976749.2978389>
- [14] Ralph C. Merkle. A Certified Digital Signature. In Lecture Notes in Computer Science. 218–238.
- [15] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. bitcoin.org. Retrieved from <https://bitcoin.org/bitcoin.pdf>
- [16] Rafael Pass, Lior Seeman, and Abhi Shelat. 2017. Analysis of the Blockchain Protocol in Asynchronous Networks. In Lecture Notes in Computer Science. 643–673.
- [17] Rafael Pass and Elaine Shi. 2017. The Sleepy Model of Consensus. In Lecture Notes in Computer Science. 380–409.
- [18] Phil Daian and Rafael Pass and Elaine Shi. 2016. Snow White: Provably Secure Proofs of Stake. (2016).
- [19] Phil Daian Rafael Pass. Snow White: Robustly Reconfigurable Consensus and Applications to Provably Secure Proofs of Stake.
- [20] C. P. Schnorr. 1991. Efficient signature generation by smart cards. J. Cryptology 4, 3 (1991). DOI:<https://doi.org/10.1007/bf00196725>
- [21] Jagdeep Sidhu. 2017. Syscoin: A Peer-to-Peer Electronic Cash System with Blockchain-Based Services for E-Business. In 2017 26th International Conference on Computer Communication and Networks (ICCCN). DOI:<https://doi.org/10.1109/icccn.2017.8038518>

- [22] Sunny King And. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. Retrieved 2012 from <https://peercoin.net/assets/paper/peercoin-paper.pdf>
- [23] Scott Nadal Sunny King. 2012. Peercoin. Retrieved January 9, 2018 from <https://peercoin.net/assets/paper/peercoin-paper.pdf>
- [24] Ewa Syta, Philipp Jovanovic, Eleftherios Kokoris Kogias, Nicolas Gailly, Linus Gasser, Ismail Khoffi, Michael J. Fischer, and Bryan Ford. 2017. Scalable Bias-Resistant Distributed Randomness. In 2017 IEEE Symposium on Security and Privacy (SP). DOI:<https://doi.org/10.1109/sp.2017.45>
- [25] Ewa Syta, Iulia Tamas, Dylan Visser, David Isaac Wolinsky, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ismail Khoffi, and Bryan Ford. 2016. Keeping Authorities “Honest or Bust” with Decentralized Witness Cosigning. In 2016 IEEE Symposium on Security and Privacy (SP). DOI:<https://doi.org/10.1109/sp.2016.38>
- [26] V Buterin And. 2015. Casper. Retrieved January 9, 2018 from <https://blog.ethereum.org/2015/08/01/introducing-casper-friendly-ghost/>
- [27] G. Wood. 2014. Ethereum: A Secure Decentralised Generalised Transaction Ledger. Ethereum Project Yellow Paper. (2014).
- [28] Gavin Wood. 2018. ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER. [ethereum.github.io/yellowpaper](https://ethereum.github.io/yellowpaper/paper.pdf). Retrieved from <https://ethereum.github.io/yellowpaper/paper.pdf>
- [29] 2017. Steem: An incentivized, blockchain-based, public content platform. [steem.io](https://steem.io/SteemWhitePaper.pdf). Retrieved from <https://steem.io/SteemWhitePaper.pdf>
- [30] Whitepapers. [bitshares.org](http://docs.bitshares.org/bitshares/papers/). Retrieved from <http://docs.bitshares.org/bitshares/papers/>
- [31] Proof of stake instead of proof of work. Retrieved January 9, 2018 from <https://bitcointalk.org/index.php?topic=27787.0>