

제로지(0G)

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.

0G: 완전한 탈중앙화 AI 운영체제를 향하여

제로 그래비티 연구소
og@og.ai
2024년 11월 28일
초안 버전

초록

대규모 언어 모델의 출현과 병렬 컴퓨팅 하드웨어(GPU, TPU, NPU)의 급속한 발전은 AGI를 향한 진보를 크게 가속화시켰다. 최근 인터넷 전역에서 수집된 방대한 데이터를 기반으로 학습된 고도화된 모델들은 많은 측면과 실제 시나리오에서 인간보다 현저히 우월한 성능을 보여주고 있다. 그러나 현재의 중앙집중적이고 독점적인 모델 학습 과정은 이러한 강력한 모델을 적용하는 데 있어 보안, 프라이버시, 공정성에 대한 많은 우려를 불러일으켰다. 이에 따라 최근에는 AI 제품의 생산 및 배포 과정에서 투명성과 민주화에 대한 강한 요구가 나타나고 있다. 블록체인 기술이 뒷받침하는 탈중앙화는 이를 달성하기 위한 유망한 경로로 인식되고 있다. 불행히도 현재 블록체인 산업의 인프라는 실제 세계의 AI 시스템과 응용 시나리오에 요구되는 인터넷급 확장성을 달성하는 데 여전히 막대한 확장성 문제에 직면해 있다. 0G는 모듈형 및 계층형 아키텍처 설계를 통해 탈중앙화된 AI 운영체제를 제공함으로써 궁극적인 해결책을 향한 첫걸음을 내딛는다. 이는 스토리지 네트워크, 데이터 가용성 네트워크, 데이터 제공 네트워크로 구성되며, 이들 네트워크의 컴퓨팅 및 스토리지 자원을 퍼미션리스 방식으로 관리하고, 별도의 0G 합의 네트워크와 유기적으로 오케스트레이션한다. 이러한 모든 네트워크 구성 요소는 무한한 수평 확장성을 달성하기 위해 정교하게 설계된 혁신적인 샤딩 메커니즘을 활용

하며, 이를 통해 AI의 진정한 민주화를 가로막는 장애물을 제거한다.

1 서론

인공지능(AI)은 산업을 변화시키고 우리의 일상생활을 재구성하는 강력한 도구로 빠르게 진화해왔다. 그러나 AI 개발과 배포의 여정은 종종 중앙집중형 플랫폼에 갇혀 있으며, 이는 프라이버시, 공정성, 정합성에 대한 중대한 우려를 초래한다. 우리가 AI의 확산을 계속 목격하면서, 하나의 중요한 흐름이 나타나고 있다: 탈중앙화를 통한 AI의 민주화이다.

중앙집중형 AI 플랫폼은 편리하고 강력하지만, 여러 중대한 문제를 야기한다. 소수의 주체가 AI 인프라를 통제하면, 그들은 데이터와 알고리즘 또한 통제하게 된다. 이러한 중앙집중화는 사용자의 프라이버시가 데이터 유출이나 개인정보 오용을 통해 손상될 수 있는 상황을 만든다. 더 나아가, AI 시스템의 의사결정 과정은 불투명해져 공정성과 책임성을 보장하기 어렵게 된다. AI가 인간의 가치와 의도에 부합하도록 작동하는지(정합성, alignment)를 확보하는 것은 특히 중앙집중적 환경에서는 어려워진다. 만약 AI 개발이 몇몇 기관에 의해 독점된다면, 이는 더 넓은 대중의 가치와의 불일치 위험을 높이고, 일부에게는 이익이 되지만 다른 이들에게는 해가 되는 결과로 이어질 수 있다.

탈중앙화는 유망한 대안을 제공한다. AI 워크플로우를 탈중앙화된 환경으로 이동시킴으로써 전체 과정이 더욱 투명해진다. 데이터 수집과 처리에서부터 모델 학습과 배포에 이르는 AI 생애주기의 모든 단계가 추적되고, 감사되며, 검증될 수 있다. 이러한 투명성은 데이터의 출처(데이터가 어디에서 왔는지)와 기여(누가 데이터를 소유하거나 모델에 기여했는지)에 대한 우려를 해소하는 데 도움이 된다. 또한 탈중앙화된 접근법은 단일 주체가 협소한 의제를 가지고 통제할 가능성이 적기 때문에, 더 넓고 다양한 가치 집합에 맞추어 AI 시스템을 정렬하기가 쉬워진다.

그러나 AI를 탈중앙화하는 이점이 명확한 반면, 도전과제 또한 만만치 않다. AI 워크플로우는 본질적으로 방대한 데이터 처리와 신경망 연산을 수반하는 복잡한 과정이다. 이 워크플로우를 탈중앙화된 환경에서 복제하는 것은 간단하지 않다. 이를 위해서는 AI 워크로드의 요구를 감당하면서도 탈중앙화의 이점을 유지할 수 있는 잘 설계된 추상화와 초확장성 인프라가 필요하다.

oG는 모듈형 및 계층형 아키텍처 설계를 통해 탈중앙화된 AI 운영체제를 제공함으로써 궁극적인 해결책을 향한 첫걸음을 내딛는다. 이는 스토리지 네트워크, 데이터 가용성(DA) 네트워크, 데이터 제공 네트워크로 구성되며, 이들 네트워크의 컴퓨팅 및 스토리지 자원을 퍼미션리스 방식으로 관리하고, 별도의 oG 합의 네트워크와 유기적으로 오케스트레이션한다. 이러한 모든 네트워크 구성 요소는 무한한 수평 확장성을 달성하기 위해 정교하게 설계된 혁신적인 샤딩 메커니즘을 활용하며, 이를 통해 AI의 진정한 민주화를 가로막는 장애물을 제거한다.

oG 스토리지 네트워크는 피어 투 피어 gossip 네트워크로 연결된 다수의 스토리지 노드로 구성된다. 스토리지 네트워크에 기록되는 각 데이터 블록은 데이터의 커밋먼트와 순서를 기록하기 위해 합의 네트워크에서의 트랜잭션을 동반한다. 데이터 처리량을 무한히 확장하기 위해, 스토리지 네트워크는 파티션 방식으로 구성되며, 병렬적이고 독립적으로 실행되는 임의의 수의 합의 네트워크와 연결된다. 서로 다른 독립적 애플리케이션의 데이터 요청은 서로 다른 스토리지 파티션에 기록될 수 있으며, 그 데이터 커밋먼트는 동시에 서로 다른 합의 네트워크에 기록될 수 있다. 모든 합의 네트워크는 동일한 스테이킹 상태를 가진 동일한 검증자 집합을

공유하므로 동일한 수준의 보안을 유지한다. 각 스토리지 노드는 특정 데이터 조각의 접근 가능성에 대한 증명을 합의 네트워크에 배포된 스마트 컨트랙트에 제출함으로써 마이닝 과정에 적극적으로 참여한다. 증명이 스마트 컨트랙트에 의해 검증되면 스토리지 노드는 그에 따라 보상을 받는다. 이러한 인센티브 기반 메커니즘은 오작동에 대한 처벌이 아니라 기여에 대한 보상을 통해 노드가 네트워크 유지에 참여하도록 장려하며, 결과적으로 실제로 더 나은 확장성을 달성하도록 네트워크를 촉진한다. 또한 oG 스토리지는 범용 스토리지 시스템으로 설계되어 있으며, 비정형 데이터를 보관하기 위한 append-only 로그 계층과 변경 가능한 구조적 데이터를 관리하기 위한 키-값 계층을 포함하는 다중 추상화 및 구조를 제공한다. 이를 통해 oG는 신뢰할 수 있는 데이터 인덱싱과 AI 처리를 위한 더 다양한 데이터 유형을 지원한다.

oG DA 네트워크는 낙관적 메커니즘을 갖춘 Layer2 또는 탈중앙화 AI 네트워크에서 실행된 상태의 오프체인 검증 수요에서 비롯되었다. 구체적으로, 오프체인 검증자는 보통 라이트 클라이언트(light client)이며, 트랜잭션 실행을 검증하기 위해 전체 트랜잭션 이력 데이터에 접근할 수 있어야 한다. Layer2 시나리오에서는 Layer2 네트워크의 실행된 트랜잭션을 포함하는 블록들이 라이트 클라이언트가 추가 검증을 수행할 수 있도록 어딘가에 게시되고 저장되어야 한다. 유사한 요구는 탈중앙화 AI 인프라에서도 존재하는데, 네트워크 내 디바이스에서 수행된 학습 또는 추론 작업의 결과가 사용자 요구나 시스템 인센티브로 인해 추가 검증을 필요로 하기 때문이다. 예를 들어, ORA/OpML 시나리오에서는 참가자들이 특정 AI 작업에 대해 챌린지 기간 동안 사기 증명(fraud proof)을 제공해야 하며, 이 증명에는 해당 작업에서 사용된 데이터와 모델이 포함될 수 있다. 일부 보상 메커니즘은 또한 오래된 과거 작업을 무작위로 선택해 검증하고 그에 따라 보상을 지급해야 하며, 이를 통해 검증 비용과 효과성 간의 균형을 달성한다.

다른 대체 DA 솔루션들(예: Celestia, EigenDA)과 달리, oG DA는 궁극적인 확장성과 보안을 목표로 한다. 이는 데이터 가용성 워크플로우를 데이터 게시 레인(Data Publishing Lane)과 데이터 저장 레인(Data Storage Lane)으로 분리하는 아이디어를 채택한다. 대량의 데이터 전송은 DA 노드가 담당하는 데이터 저장 레인에서 발생하며, 이는 소거 코드 기반의 데이터 슬라이싱을 통해 수평 확장성을 지원한다. 반면 데이터 게시 레인은 합의 네트워크에서 해당 DA 노드들의 집계 서명을 확인함으로써 데이터 가용성 속성을 보장하며, 이는 합의 프로토콜을 통해 흐르는 데이터량을 최소화하여 브로드캐스팅 병목 현상을 피한다.

네트워크의 DA 노드들은 또한 oG 합의에서 스테이커가 되어 다수 정직성 가정을 가진 일련의 퀴럼(quorum)을 형성해야 한다. 이것이 데이터 가용성의 보안 기반이다. oG의 퀴럼은 합의 시스템이 검증 가능한 난수 함수(VRF(검증 가능한 난수 함수, Verifiable Random Function))를 통해 무작위로 구성하며, 이론적으로는 전체 합의 네트워크 검증자 집합에서의 정직한 참가자 분포와 동일한 분포를 보장하므로 데이터 가용성 클라이언트가 그들과 담합할 수 없다. 다시 말해, 전체 검증자의 부분집합으로서 퀴럼은 전체 합의 네트워크와 동일한 보안 속성을 공유한다. 퀴럼의 집계 서명은 oG 합의에 제출되어 데이터 가용성 확인을 받으며, 이는 이더리움보다 수십 배 빠르고 효율적일 수 있다. 무한 확장성을 가진 oG의 다중 합의 설계와 결합되면, 합의 프로토콜은 데이터 처리량의 병목이 되지 않는다. 또한 EigenDA와 마찬가지로 데이터 가용성 클라이언트는 데이터를 청크로 나누기 위해 소거 코딩을 수행해야 한다. 실제로 이 과정은 비용이 많이 들며 단일 클라이언트의 처리량 병목이 될 수 있다. oG는 GPU 가속 솔루션을 제공하여 소거 코딩 과정을 크게 가속화한다.

나아가 효율적이고 확장 가능한 AI 연산과 데이터 접근을 지원하기 위해, oG는 탈중앙화된 서비스 네트워크를 제안한다. 이는 데이터 검색, AI 추론, 심지어 AI 학습 작업을 포함한 모든 종류의 서비스 워크로드를 지원할 수 있는 범용 프레임워크이다. 이는 모든 유형의 서비스가 탈중앙화 네트워크에 통합될 수 있도록 일련의 스마트 컨트랙트와 분산 소프트웨어로 구성된다. oG 서비스 프레임워크 SDK가 결합되면, 서비스 제공자는 서비스 유형, 가격, 검증 방법을 스마트 컨트랙트에 등록하고 게시할 수 있다. 사용자는 플랫폼을 통해 자신이 사용하고자 하는 서비스를 발견하고, 특정 금액의 oG 토큰을 스마트 컨트랙트에 선결제할 수 있다. 이후 사용자는 서비스와 직접 상호작용하여 요청을 보내고, 응답을 수신 및 검증하며, 응답을 서명으로 확인한다. 서비스 제공자는 사용자 확인과 함께 요청/응답 시퀀스를 유지하고, 언제든지 이를 스마트 컨트랙트에 제출하여 사용자 수수료에서 해당 보상을 받을 수 있다. 이러한 방식으로 서비스 워크로드는 완벽히 파티션되어 수평 확장성을 달성할 수 있으며, 블록체인 오버헤드는 최소화된다. 모든 서비스 프로세스의 추적 기록은 정산을 위해 스마트 컨트랙트에 제출되므로, 이 정보는 네트워크 전체에서 서비스의 기여도와 품질을 평가하는 데도 사용될 수 있으며, 이를 통해 더 나은 서비스 제공자에게 적절하고 공정한 인센티브를 제공할 수 있다.

2 시스템 설계 개요

oG 시스템은 탈중앙화 스토리지 시스템(oG Storage) 위에 구축된 데이터 가용성 계층(oG DA)으로 구성된다. 하나 또는 여러 개의 독립된 합의 네트워크가 존재하며, 이는 oG DA와 oG Storage 모두의 일부를 이룬다. oG Storage의 경우 각 합의 네트워크는 업로드된 데이터 블록의 순서를 결정하고, 스마트 컨트랙트를 통해 스토리지 마이닝 검증과 그에 따른 보상 메커니즘을 구현하는 역할을 맡는다. oG DA의 경우 합의는 해당 데이터 가용성 쿼럼으로부터 집계된 서명을 검증함으로써 각 데이터 블록의 데이터 가용성 속성을 보장한다.

그림 1은 단일 합의 네트워크를 배치했을 때의 oG 시스템 아키텍처를 보여준다. 데이터 블록이 oG DA에 들어오면, 먼저 소거 코딩을 통해 여러 연속적인 청크로 조직된다. 인코딩된 데이터 블록의 커밋먼트로서 머클 루트가 합의 계층에 제출되어 시스템에 들어오는 데이터의 순서를 유지한다. 이후 청크들은 oG Storage의 쿼럼 내 서로 다른 스토리지 노드에 분산된다. 스토리지 노드들은 합의 네트워크와 상호작용하여 시스템으로부터 보상을 얻기 위해 주기적으로 마이닝 과정에 참여한다. DA 클라이언트는 쿼럼으로부터 집계된 서명을 수집한 뒤, 이를 합의에 제출해 검증을 받는다.

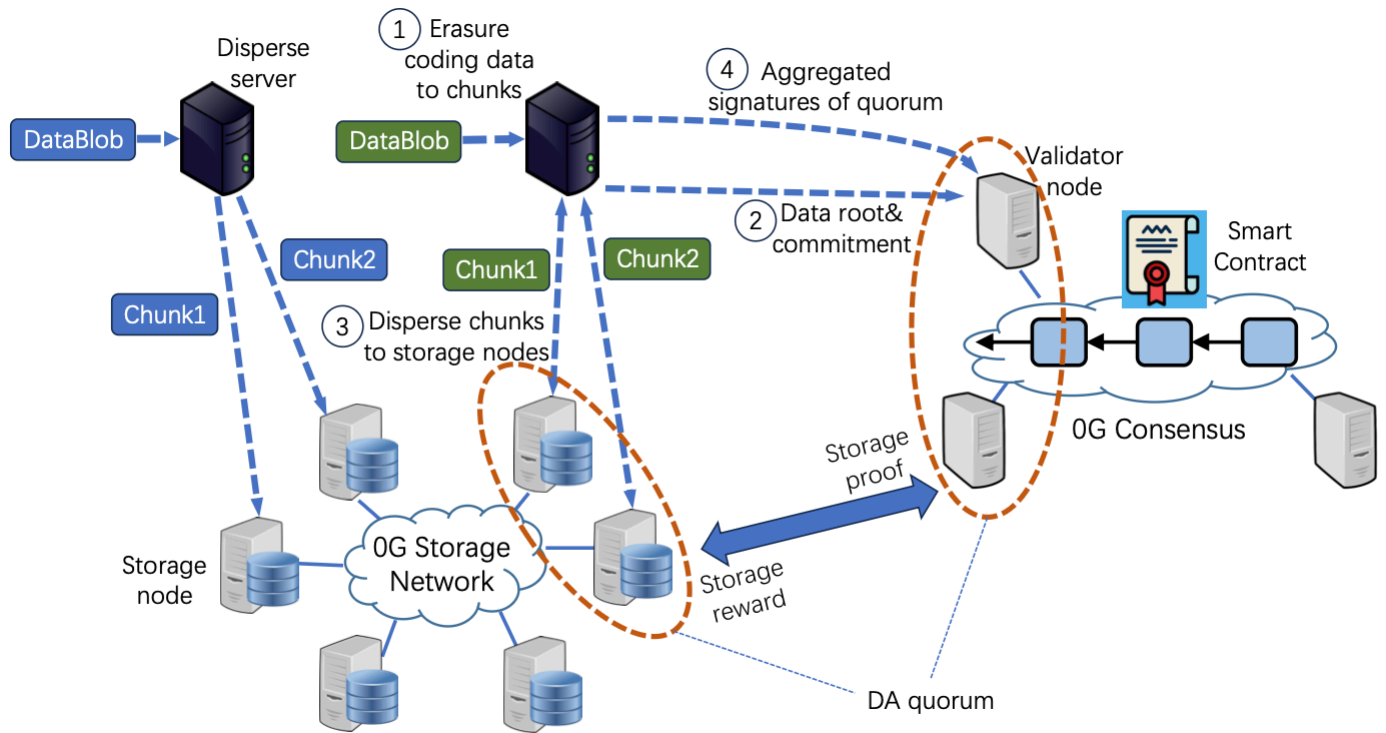


그림 1: oG 시스템의 아키텍처.

여러 POS 기반 합의 네트워크로 배치될 경우, 각 검증자는 공유 스테이킹 상태를 사용하여 모든 합의 네트워크의 유지에 참여한다. 공유 스테이킹 상태는 여러 합의 네트워크 중 하나의 스마트 컨트랙트에 기록되거나, oG 합의 네트워크 집합 외부의 블록체인 네트워크(예: 이더리움)에 기록될 수 있다. C_0 가 공유 스테이킹 상태를 유지하는 합의 네트워크라고 가정하면, C_0 의 토큰 T_0 는 POS 투표를 위해 스테이킹된다. 네트워크 C_i ($i \neq 0$)에서 인센티브를 통해 토큰 T_i 가 발행되면, 이는 C_i 에서 T_i 를 소각하고 보안이 확보된 크로스체인 브리지 채널을 통해 C_0 의 T_0 로 매핑될 수 있다. 검증자가 네트워크 C_i 에서 POS 프로토콜을 유지할 때는 항상 C_0 에 스테이킹된 T_0 의 양을 C_i 에서의 투표권으로 사용한다. 이에 따라 모든 네트워크 C_i 는 동일한 수준의 POS 보안을 공유한다. 이더리움을 네트워크 C_0 로 사용하면, 토큰 T_0 를 표준 ERC20 토큰으로 구현할 수 있는 이점이 있으며, EigenLayer와 같은 기존의 인기 있는 리스테이킹(restaking) 프레임워크와 손쉽게 통합할 수도 있다. 또한 공유 스테이킹 상태의 원천은 EigenLayer 및 Babylon과 같은 리스테이킹 프레임워크 내의 고가치 토큰(예: BTC, ETH)일 수도 있으며, 이를 통해 oG 합의가 비트코인 또는 이더리움 수준의 보안에 도달할 수 있다. 그림 2에 나타난 바와 같이, 이러한 공유 스테이킹 설계는 oG 시스템에 무제한 확장성을 도입한다.

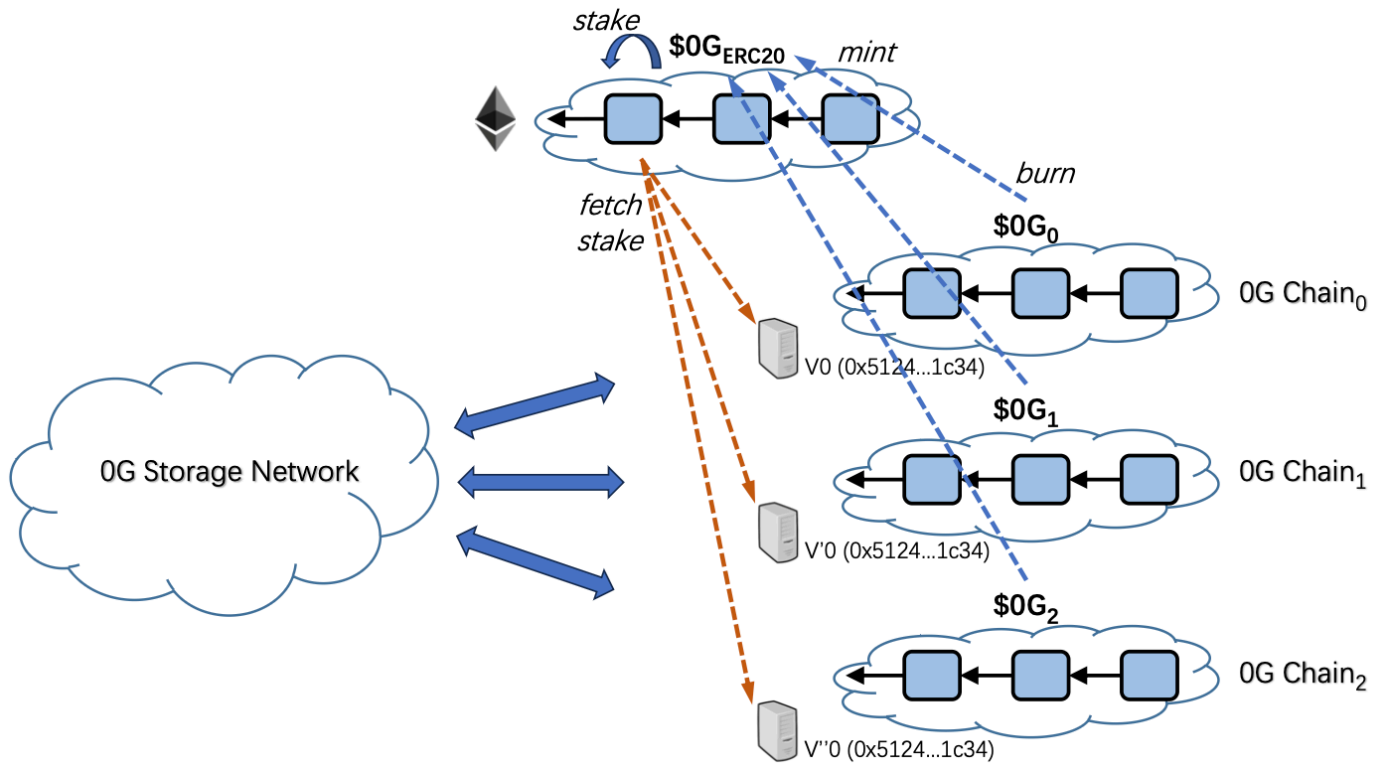


그림 2: oG 시스템의 무한 확장성.

3 OG 스토리지 설계

OG 스토리지는 다양한 유형의 탈중앙화 애플리케이션을 지원하기 위해 계층적 설계를 채택한다. 그림 3은 OG 스토리지의 풀스택 계층 개요를 보여준다.

가장 하위에는 로그 계층이 있으며, 이는 데이터 저장에 대해 보상을 제공하는 보상 메커니즘이 내장된 스토리지 네트워크를 구성하는 여러 스토리지 노드로 이루어진 탈중앙화 시스템이다. 업로드된 데이터의 순서는 로그 기반 의미론과 추상화를 제공하는 시퀀싱 메커니즘에 의해 보장된다. 이 계층은 영구 보존을 위한 비정형 원시 데이터를 저장하는 데 사용된다.

Transaction Processing	ACID for applications with concurrent update, e.g., Google doc with collaborative editing
Key-Value Store	Enables the mutability of the data with access control and allows applications like decentralized social network
Log System	Archive system which provides permanent persistency and append-only property

그림 3: oG 스토리지의 풀스택 솔루션.

로그 계층 위에서, oG 스토리지는 구조화된 데이터를 가변적으로 관리하기 위한 키-값 스토어 런타임을 제공한다. 여러 키-값 스토어 노드들은 기반 로그 시스템을 공유하며, 구조화된 키-값 데이터를 로그 항목에 삽입해 로그 시스템에 추가한다. 이들은 공유 로그의 로그 항목을 재생하여 키-값 스토어의 일관된 상태 스냅샷을 구성한다. 키-값 스토어의 처리량과 지연 시간은 로그 시스템에 의해 제한되므로, 로그 계층의 효율성은 시스템 전체 성능에 결정적으로 중요하다. 키-값 스토어는 데이터의 갱신 권한을 관리하기 위해 키에 접근 제어 정보를 연계할 수 있다. 이를 통해 사용자가 생성한 메시지의 소유권 유지를 필요로 하는 탈중앙화 트위터와 같은 소셜 네트워크 애플리케이션을 구현할 수 있다.

oG 스토리지는 또한 키-값 스토어 런타임 위에서 트랜잭션 의미론을 제공하여, 쓰기 권한을 가진 여러 사용자가 키에 대해 동시에 갱신할 수 있도록 한다. 기반 로그 시스템이 보장하는 로그 항목의 전체 순서는 키-값 스토어 상에서의 트랜잭션 실행 동시성 제어의 토대가 된다. 이러한 기능을 통해 oG 스토리지는 협업 편집과 심지어 데이터베이스 워크로드와 같은 탈중앙화 애플리케이션을 지원한다.

4 로그 시스템 프로토콜

oG 스토리지의 로그 계층은 퍼미션리스 네트워크로 연결된 스토리지 노드들을 통해 탈중앙화 스토리지 서비스를 제공한다. 이 스토리지 노드들은 보관된 데이터를 협력적으로 서비스하며, 각 노드는 자신이 로컬 스토리지에 유지하는 데이터의 일부를 선택적으로 지정할 수 있다.

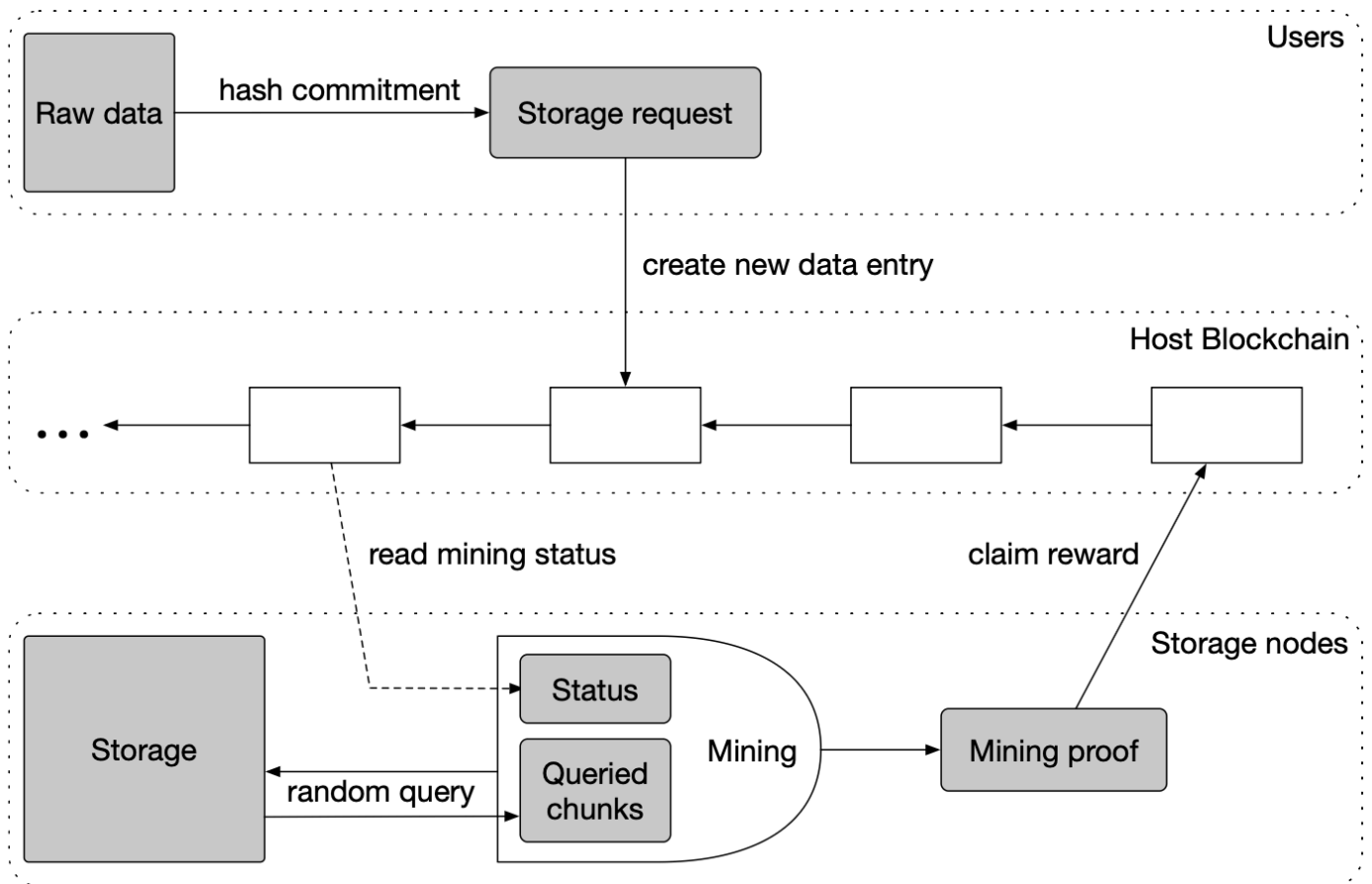


그림 4: oG 스토리지 프로토콜 개요. 데이터 생성과 보상 분배는 완전히 분리되어 있다. 사용자는 호스트 블록체인에 배포된 oG 스토리지 컨트랙트에 직접 저장 요청을 제출하며, 스토리지 노드는 보관된 데이터에 대해 무작위 질의에 응답할 수 있음을 컨트랙트에 증명하여 보상을 청구한다. oG 토큰은 별도의 독립된 컨트랙트에서 처리되어, 비차단 토큰 전송을 가능하게 한다.

4.1 프로토콜 개요

0G 스토리지 네트워크의 저장 상태는 0G 블록체인에 배포된 스마트 컨트랙트에 의해 유지된다. 이 컨트랙트는 0G 스토리지 컨트랙트라 불리며, 해당 블록체인은 호스트 블록체인 또는 줄여서 호스트 체인이라 불린다.

0G 스토리지 네트워크의 설계는 데이터 생성, 보상 분배, 토큰 유통을 완전히 분리한다.

0G 스토리지 컨트랙트는 데이터 저장 요청 처리, 데이터 항목 생성, 보상 분배를 담당한다. 데이터 저장 요청은 0G 스토리지 네트워크에 데이터를 저장하고자 하는 사용자에게 의해 제출되며, 각 요청에는 데이터 크기와 커밋먼트와 같은 필요한 메타데이터가 포함되고, 저장 서비스에 대한 결제가 함께 이루어진다. 승인된 데이터 요청에 대해 데이터 항목이 생성되어 저장된 데이터를 기록하며, 보상 분배는 마이닝 과정을 통해 독립적으로 처리된다. 스토리지 노드들은 0G 스토리지 네트워크 유지에 대한 보상을 청구하기 위해 마이닝 증명을 0G 스토리지 컨트랙트에 제출한다.

4.2 저장 단위

0G 스토리지의 로그 계층은 로그 항목 단위로 업데이트(append-only)되며, 각 항목은 0G 스토리지 컨트랙트에 전송된 저장 요청 트랜잭션에 의해 생성된다. 로그 계층은 파일시스템과 유사하며, 각 로그 항목은 하나의 파일에 해당한다.

로그 시스템은 고정 크기 섹터 단위로 동작하며, 각 섹터는 256B의 데이터를 저장한다. 서로 다른 로그 항목이 동일한 섹터를 공유하지 않도록 하기 위해, 모든 로그 항목은 여러 섹터 단위로 패딩되어야 한다.

0G 스토리지의 마이닝 과정은 무작위 챌린지 질의에 대한 데이터 접근성을 증명할 것을 요구한다. SSD 스토리지의 경쟁 우위를 극대화하기 위해, 챌린지 질의는 256KB 청크(즉, 1024 섹터) 단위로 설정된다. 각 챌린지 질의는 채굴자가 전체 청크 데이터에 대한 접근 가능성을 증명해야 하므로, 스토리지 노드들은 청크 단위로 데이터를 유지한다.

4.3 데이터 흐름(data flow)

0G 스토리지에서는 커밋된 데이터가 순차적으로 조직된다. 이러한 데이터의 순차열을 데이터 흐름이라 부르며, 이는 데이터 항목들의 목록 또는 고정 크기 데이터 섹터들의 순차열로 해석될 수 있다. 따라서 0G의 모든 데이터는 보편적인 오프셋으로 편리하게 인덱싱될 수 있으며, 이 오프셋은 PoRA의 마이닝 과정에서 챌린지 샘플링에 사용된다.

기본 데이터 흐름은 0G의 “메인 플로우”라 불리며, 모든 새로운 로그 항목을(별도 지정되지 않는 한) append-only 방식으로 통합한다. 특정 애플리케이션과 관련된 데이터와 같이 특정 범주의 로그 항목만을 수용하는 특수화된 플로우도 존재한다.

특수화된 플로우의 가장 중요한 장점은 연속적인 주소 공간으로, 이는 일부 사용 사례에서 매우 중요할 수 있다. 더 나아가, 특수화된 플로우는 맞춤형 저장 가격을 적용할 수 있으며, 이는 일반적으로 기본 플로우의 최저 가격보다 훨씬 높고, 따라서 더 나은 데이터 가용성과 신뢰성을 달성할 수 있다.

5 인센티브 메커니즘

이 절에서는 사용자와 채굴자(즉, “스토리지 노드”)라는 두 가지 유형의 참여자로 구성된 oG 스토리지의 보상 메커니즘 설계를 설명한다. 사용자는 로그에 데이터 항목을 생성하고 네트워크에 데이터를 추가하기 위해 oG 토큰을 지불하며, 채굴자는 데이터 서비스를 제공하고 네트워크로부터 보상으로 oG 토큰을 받는다. 사용자에서 채굴자로의 지불은 특정 채굴자가 아닌 네트워크 전체에 의해 서비스가 유지되기 때문에 oG 합의 네트워크에 의해 중개된다.

oG 스토리지는 “일정 기간에 대해 일회성 결제” 방식으로 저장 서비스를 구현한다. 사용자는 각 데이터 항목을 생성할 때 예를 들어 3년과 같은 특정 기간에 대한 스토리지 기부금을 지불하고, 이후 이 기부금은 해당 데이터 항목을 유지하는 채굴자를 선형적으로 분배하여 인센티브를 제공한다. 사용자는 스토리지 기간을 연장하기 위해 저장 기간 갱신할 수 있다.

스토리지 기부금은 데이터 항목별로 유지되며, 채굴자는 자신이 접근할 수 있는 데이터 항목에서만 저장 보상을 받을 수 있다. 특정 데이터 항목에 지급되는 총 저장 보상은 해당 항목의 인기도와 무관하다. 예를 들어, 많은 채굴자에 의해 저장된 인기 있는 데이터 항목은 자주 채굴되지만, 보상은 그 채굴자들 사이에 분할된다. 반면, 거의 채굴되지 않는 비인기 데이터 항목은 저장 보상이 축적되어 해당 희소한 데이터 항목을 저장하는 채굴자에게 더 높은 수익을 제공한다.

5.1 저장 요청 가격 책정

각 oG 스토리지 요청의 비용은 다음 두 부분으로 구성된다: a) 수수료, b) 스토리지 기부금. 수수료 부분은 저장 요청을 처리하고 새로운 데이터 항목을 로그에 추가하기 위해 oG 컨트랙트를 호출하는 데 대해 호스트 체인 채굴자/검증자에게 지급되며, 이는 다른 스마트 컨트랙트 호출 트랜잭션과 동일하게 가격이 책정된다. 이후에서는 해당 데이터에 서비스를 제공하는 oG 스토리지 채굴자에게 지속적인 보상을 지원하는 스토리지 기부금 부분에 초점을 맞춘다.

특정 기부금 $SR_{endowment}$ 과 커밋된 데이터 크기 $SR_{data\ size}$ (256B 섹터 수로 측정됨)을 가진 데이터 저장 요청 SR 이 주어졌을 때, SR 의 단위 가격은 다음과 같이 계산된다:

$$SR_{unit\ price} = \frac{SR_{endowment}}{SR_{data\ size}}$$

이 단위 가격 $SR_{unit\ price}$ 은 로그에 추가되기 위해 전역적으로 지정된 하한보다 커야 하며, 그렇지 않으면 요청은 하한이 $SR_{unit\ price}$ 보다 낮아질 때까지 대기 상태가 된다(그동안 채굴자들은 이 미지급 데이터를 저장하지 않을 가능성이 크다). 사용자는 더 높은 단위 가격 $SR_{unit\ price}$ 를 자유롭게 설정할 수 있으며, 이는 더 많은 스토리지 노드가 해당 데이터 항목을 채굴하도록 동기를 부여하고 결과적으로 더 나은 데이터 가용성을 이끌어낸다.

5.2 랜덤 접근 증명(PoRA)(PoRA)

oG 네트워크는 채굴자에게 데이터를 저장하도록 인센티브를 제공하기 위해 Proof of Random Access(PoRA) 메커니즘을 채택한다. 채굴자에게 보관된 데이터 청크에 대한 무작위 질의에 응답하도록 요구함으로써, PoRA 메커니즘은 마이닝 증명 생성 능력과 데이터 저장 사이의 관계를 보장한다. 채굴자는 질의에 반복적으로 응답하며 각 로드된 청크에 대해 출력 다이제스트를 계산하다가, 채굴 난이도(즉, 충분한 선행 0을 가진 해시)를 만족하는 다이제스트를 찾을 때까지 이를 수행한다. PoRA는 채굴자의 디스크 I/O를 강하게 사용하게 되어 사용자 질의에 응답하는 능력을 줄인다. 따라서 oG 스토리지는 간헐적 마이닝 방식을 채택하며, 여기서 마이닝 에포크는 호스트 체인의 특정 블록 높이에서 블록 생성으로 시작하고, 유효한 PoRA가 oG 스토리지 컨트랙트에 제출되면 종료된다.

가장 단순한 설계에서, PoRA 반복은 계산 단계와 로딩 단계로 구성된다. 계산 단계에서는 채굴자가 임의로 선택한 난스와 호스트 체인에서 읽은 마이닝 상태를 기반으로 무작위 회수 위치(플로우 내 보편적 오프셋)를 계산한다. 로딩 단계에서는 채굴자가 지정된 회수 위치의 보관된 데이터 청크를 로드하고, 마이닝 상태와 데이터 청크의 튜플을 해싱하여 출력 다이제스트를 계산한다. 출력 다이제스트가 목표 난이도를 만족하면, 채굴자는 선택한 난스, 로드된 데이터 청크, 그리고 데이터 청크의 정확성 증명을 포함한 합법적인 PoRA를 구성하여 마이닝 컨트랙트에 제출할 수 있다.

이후에서는 PoRA 마이닝의 공정성을 개선하기 위한 주요 고려 사항을 소개하고, PoRA 마이닝 메커니즘을 형식화한다.

5.2.1 소규모 채굴자에 대한 공정성

oG 스토리지 네트워크의 저장 용량이 단일 기계의 저장 용량을 크게 초과할 경우, 단일 기계는 사용 가능한 회수 위치를 찾는 데 거의 모든 시간을 소모한다. 이는 PoRA가 작업 증명(PoW)으로 변질되게 만든다. PoRA 마이닝 과정을 단일 기계만 가진 소규모 채굴자에게도 친화적으로 만들기 위해, 마이닝 범위는 8TB로 제한된다. 보관된 데이터 청크의 크기가 8TB를 초과하면, 채굴자는 데이터 흐름 시퀀스에서 8TB 크기의 마이닝 범위를 지정해야 한다. 모든 데이터를 저장할 충분한 기계를 가진 대규모 채굴자는 서로 다른 데이터 범위에서 동시에 마이닝할 수 있다.

5.2.2 스토리지 아웃소싱 억제

네트워크 전체가 데이터 청크의 충분한 복제본을 저장하도록 촉진하기 위해, oG 스토리지는 단일 스토리지 복제본만 가진 채굴자의 보상 비율을 제한한다. PoRA 메커니즘은 각 채굴자별로 데이터를 서로 다른 방식으로 봉인(seal)하고, 봉인된 데이터 접근을 챌린지하며, 타인의 데이터 청크를 사용한 마이닝을 방지한다. 채굴자가 데이터를 아웃소싱하여 PoRA 마이닝에서 보관된 데이터 청크를 질의할 경우, 응답 데이터를 봉인하여 출력 다이제스트를 계산해야 하므로 큰 비용이 든다. 채굴자는 단일 스레드로 수 MB에서 수십 MB의 데이터 청크를 봉인 및 해제할 수 있으므로 여전히 사용자 질의에는 효율적으로 응답할 수 있다. SSD에서 데이터를 읽을 때 최대 7GB/s 속도에 도달할 수 있으므로, 데이터 봉인은 PoRA 마이닝에 무거운 작업이다. 데이터 봉인 비용이 디스크 구매와 데이터 동기화 비용을 초과하는 한, 채굴자들은 PoRA 마이닝에서 데이터를 봉인하기보다는 데이터 청크의 복제본을 저장하는 쪽을 선호하게 된다.

보다 구체적으로, 각 4KB 데이터 청크는 채굴자 ID 및 기타 컨텍스트 데이터로 봉인된다. 봉인 시드가 채굴자 ID와 컨텍스트 데이터의 32바이트 다이제스트라고 가정하면, 그림 5는 봉인 과정을 정의한다.

Input : seal_seed, unsealed_data

Output: sealed_data

- 1 Regard unsealed_data in length of 4 KB as an array of 32-byte elements $\vec{d}$
- 2 $h \leftarrow \text{seal_seed}$
- 3 **for** i **from** 0 **to** 127 **do**
- 4 $\vec{d}[i] \leftarrow \vec{d}[i] \text{ XOR } h$
- 5 $h \leftarrow \text{Keccak256}(\vec{d}[i])$
- 6 Regard $\vec{d}$ as a 4-KB data chunk sealed_data

그림 5: 4KB 데이터 청크 봉인

5.2.3 분산 마이닝 억제

데이터 저장에 인센티브를 제공하는 마이닝 메커니즘으로서, 단일 스토리지 복제본은 공정성을 위해 유사한 해시레이트를 생성해야 한다. PoRA 마이닝 과정에서 해시레이트는 스토리지 입출력(I/O)에 의해 병목된다. 따라서 모든 채굴자가 단위 저장소당 유사한 스토리지 I/O를 갖기를 기대한다. 그러나 마이닝 팜은 인메모리 파일시스템을 갖춘 분산 시스템에 데이터 청크를 저장함으로써 일반 채굴자보다 I/O를 크게 증가시킬 수 있다. DDR 메모리는 SSD보다 훨씬 더 큰 대역폭을 가지므로, 마이닝 팜은 데이터 저장에는 거의 기여하지 않으면서 훨씬 더 많은 해시레이트를 생성할 수 있다.

OG 스토리지는 채굴자가 두 단계를 동일한 기계에서 완료하도록 유도하기 위해 계산 단계에서 로딩 단계로의 대량 데이터 전송을 부과함으로써 이러한 행위를 방지한다. 따라서 계산 단계와 로딩 단계가 서로 다른 기계로 분리되면, 해시레이트는 일반적으로 스토리지 I/O 대역폭보다 작은 네트워크 대역폭에 의해 제한된다. 계산 단계에서는 회수 위치와 함께 무작위 스크래치패드가 생성되며, 이는 로드될 데이터 청크와 동일한 길이를 가진다. 로드된 데이터 청크의 해시를 계산하기 전에, 데이터 청크는 XOR 연산으로 스크래치패드와 혼합되어야 한다.

그러나 읽기 대역폭과 호스트 체인에 PoRA를 제출하는 트랜잭션 수수료 사이에는 딜레마가 존재한다. 로컬 디스크에서 데이터 청크를 로드하는 것이 네트워크를 통해 다운로드하는 것보다 더 효율적이므로, 로딩 단계에서는 순차적 접근에서 SSD의 빠른 읽기 속도를 활용하여 데이터 청크 로딩 대역폭을 극대화해야 한다. 벤치마크에 따르면, 무작위로 256KB 데이터 청크를 로드할 때 순차 접근의 80% 읽기 속도에 도달할 수 있다. 그러나 데이터 청크를 호스트 체인에 제출할 때, 256KB 트랜잭션은 지나치게 커서 상당한 트랜잭션 수수료를 소모한다. 데이터 청크 크기를 줄이면 이 문제를 완화할 수 있지만, 읽기 속도는 저하된다.

PoRA 메커니즘은 이 딜레마를 해결하기 위해 “배치 데이터 로딩”을 채택한다. 각 반복에서 채굴자는 저장소에서 256KB 데이터 청크를 로드하고, 이를 4KB 데이터 청크로 나눈 뒤 각 4KB 데이터 청크에 대해 개별적으로 해시를 계산한다. 64개 출력 중 하나가 목표 품질을 만족하면 채굴자는 유효한 답을 찾은 것이다. 따라서 이 답에 기여한 4KB 데이터 청크만 호스트 체인에 제출하면 된다.

5.2.4 PoRA 메커니즘의 형식적 정의

정확히 말하면, 마이닝 과정은 다음 단계로 구성된다:

1. 마이닝 컨트랙트에 채굴자 ID를 등록한다.
2. 각 마이닝 에포크마다 다음 단계를 반복한다:
 - (a) 주어진 에포크 높이에서 레이어1 블록체인이 블록을 발행할 때까지 대기한다.
 - (b) 이 블록의 블록 해시 $block_hash$ 와 당시 관련 컨텍스트(머클 루트, 데이터 길이, 컨텍스트 데이터 포함)를 가져온다.
 - (c) 채굴 가능한 항목 수를 계산한다: $n = \lfloor \text{data length} / 256\text{KB} \rfloor$.
 - (d) 각 반복마다 다음 단계를 수행한다:
 1. 임의의 32바이트 난스를 선택한다.
 2. 마이닝 범위 파라미터 $start_position$ 과 $mine_length$ 를 결정한다. $mine_length$ 는 $\min(8\text{ TB}, n \cdot 256\text{KB})$ 와 같아야 한다.
 3. 그림 6의 알고리즘에 따라 회수 위치 τ 와 스크래치패드 $\vec{s}$ 를 계산한다.
 4. 위치 $h \cdot 256\text{KB}$ 에서 시작하는 256KB 봉인된 데이터 청크 $\vec{d}$ 를 로드한다.
 5. $\vec{w} = \vec{d} \oplus \vec{s}$ 를 계산하고 $\vec{w}$ 를 64개의 4KB 조각으로 분할한다.
 6. 각 조각 $\vec{v}$ 에 대해 튜플 $(miner_id, nonce, context_digest, start_position, mine_length, \vec{v})$ 의 Blake2b 해시를 계산한다.
 7. Blake2b 해시 출력 중 하나가 목표값보다 작으면, 채굴자는 합법적인 PoRA 해답을 찾은 것이다.

Input : miner_id, nonce, context_digest, start_position, mine_length

Output: Recall position τ and the scratchpad $\vec{s}$

- 1 $h \leftarrow \text{Blake2b}(\text{miner_id}, \text{nonce}, \text{context_digest}, \text{start_position}, \text{mine_length})$
- 2 Initialize an empty array $\vec{s}$ with 4096 64-byte elements
- 3 **for** $i \in \{x \in \mathbb{Z} \mid 0 \leq x < 4095\}$ **do**
- 4 $h \leftarrow \text{Blake2b}(h)$
- 5 $\vec{s}[i] \leftarrow h$
- 6 $r \leftarrow \text{Keccak256}(h)$
- 7 $n \leftarrow \text{mine_length} / 256\text{KB}$
- 8 $\tau \leftarrow \text{start_position} + r \bmod n$

그림 6: 회수 위치와 스크래치패드 계산

5.3 복제본 수의 균형

OG 스토리지의 모든 채굴자는 자신이 유지하는 데이터 항목의 범위를 명시적으로 선언해야 하며, 이후에는 오직 선언한 데이터만을 마이닝 과정에서 사용할 수 있다. 즉, 채굴자는 선언하지 않은 데이터에서 마이닝 보상을 받을 수 없으며, 반대로 선언한 범위 내의 챌린지에 응답하지 못하면 마이닝 출력은 감소한다. 결과적으로, 채굴자가 마이닝에 사용하는 데이터 범위에 대해 거짓말할 유인은 없다.

채굴자들이 자신이 유지하는 데이터 항목 목록을 진실되게 공개하기 때문에, 특정 데이터에 대해 복제본 수를 관찰하고 추정하기는 쉽다. 또한, 스토리지 기부금이 해당 데이터를 유지하는 채굴자에게 지급된다는 점을 상기하면, 채굴자들은 희귀 데이터를 저장하도록 인센티브를 받게 되고, 그 결과 복제본 수는 균형을 이루게 된다.

6 키-값(Key-Value) 런타임

OG 스토리지는 로그 계층 위에 키-값 런타임을 제공한다. 각 키-값 노드는 런타임 인터페이스를 통해 키-값 스토어 상태에 접근할 수 있다. 키-값 런타임은 Put () 및 Get ()과 같은 표준 인터페이스를 제공하며, 애플리케이션 특화 구조에서 직렬화된 키-값 쌍을 수용한다.

키-값 스토어 노드의 정상 실행 동안, 노드는 최신 키-값 상태를 로컬에 유지한다. Put () API를 통해 특정 키의 값을 갱신하며, 이는 갱신된 키-값 쌍을 포함하는 로그 항목을 구성하고 이를 로그에 추가한다. 런타임은 로그의 새로운 로그 항목을 지속적으로 모니터링하고 이를 키-값 노드로 가져와, 로그 항목의 내용에 따라 로컬 키-값 상태를 갱신한다. 이런 의미에서 여러 키-값 스토어 노드들은 본질적으로 공유된 탈중앙화 로그를 통해 서로 동기화된다. 로그 항목의 원시 내용을 애플리케이션 특화 키-값 구조로 역직렬화하기 위해 사용자 정의 함수가 사용된다.

애플리케이션은 Get () API를 사용하여 특정 키의 최신 값을 조회할 수 있다. 작은 키-값 쌍의 갱신 효율성을 높이기 위해, Put ()은 여러 키-값 쌍을 한 번에 배치 갱신할 수 있다. 그림 7은 탈중앙화 키-값 스토어의 아키텍처를 보여준다. 접근 제어를 관리하기 위해 각 키의 소유권 정보 또한 로그 항목에 유지될 수 있다. 모든 정직한 키-값 노드들은 소유권에 기반한 동일한 갱신 규칙을 따르며 상태 일관성을 달성한다.

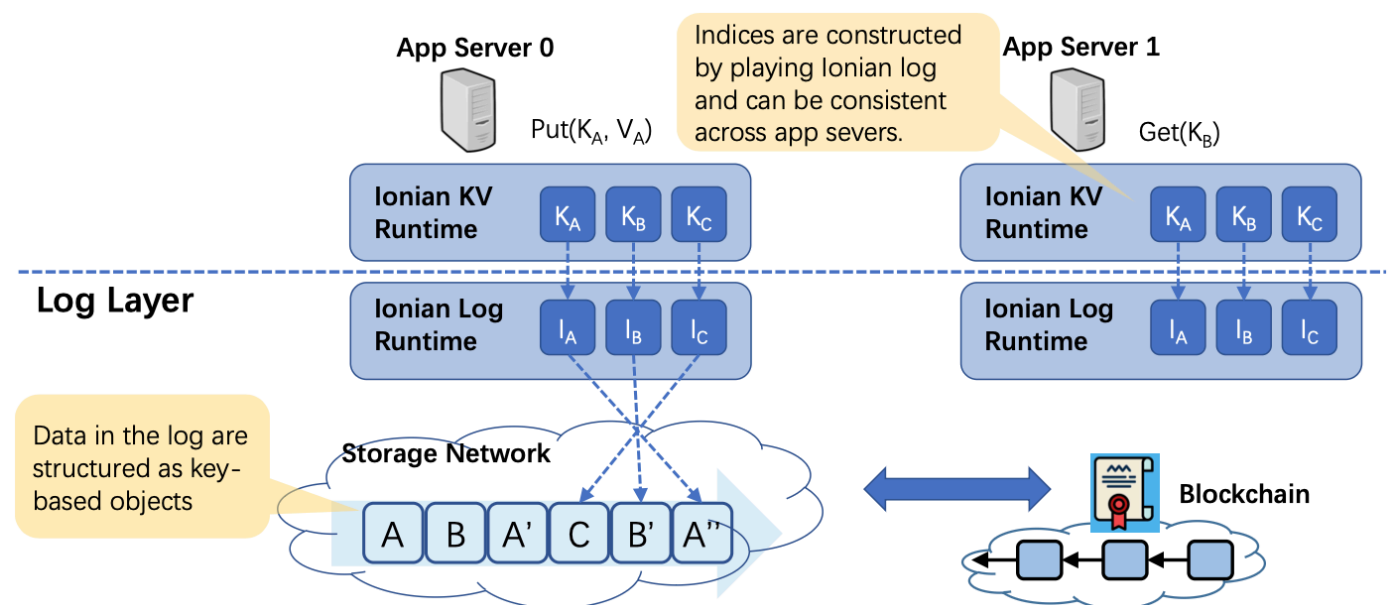


그림 7: 탈중앙화 키-값 스토어

새로운 키-값 노드가 네트워크에 합류할 때, 이는 로그 계층에 연결하여 로그 항목을 처음부터 끝까지 재생함으로써 키-값 스토어의 최신 상태를 구성한다. 로그 항목을 재생하는 동안, 애플리케이션 특화 키-값 노드는 자신이 관심을 갖지 않는 스트림 ID를 포함하지 않는 로그 항목은 건너뛸 수 있다.

7 키-값 스토어에서의 트랜잭션 처리

oG 스토리지는 키-값 런타임에서 동시성 제어를 적용하여 여러 키에 대한 동시 작업을 위한 트랜잭션 처리를 지원한다. 기반 로그 시스템이 보장하는 로그 항목의 전체 순서는 키-값 스토어 상의 트랜잭션 실행에 대한 동시성 제어의 토대를 제공한다. 이 메커니즘은 낙관적(optimistic) 방식이며, 그림 8에 설명되어 있다.

애플리케이션 서버가 oG 스토리지 키-값 런타임과 연결되어 `BeginTx()` 인터페이스를 사용해 트랜잭션을 시작할 때, 런타임에 이 트랜잭션이 현재 로그를 끝까지 재생하여 구성된 상태 스냅샷에서 동작할 것임을 알린다. `EndTx()` 호출 이전의 추가적인 키-값 연산은 서버에서 로컬로 키-값을 갱신하지만, 이 갱신은 로그에는 노출되지 않는다. `EndTx()`가 호출되면, 런타임은 트랜잭션이 시작된 로그 위치와 트랜잭션의 읽기-쓰기 집합을 포함하는 커밋 레코드를 작성한다. 이 커밋 레코드는 로그에 추가된다.

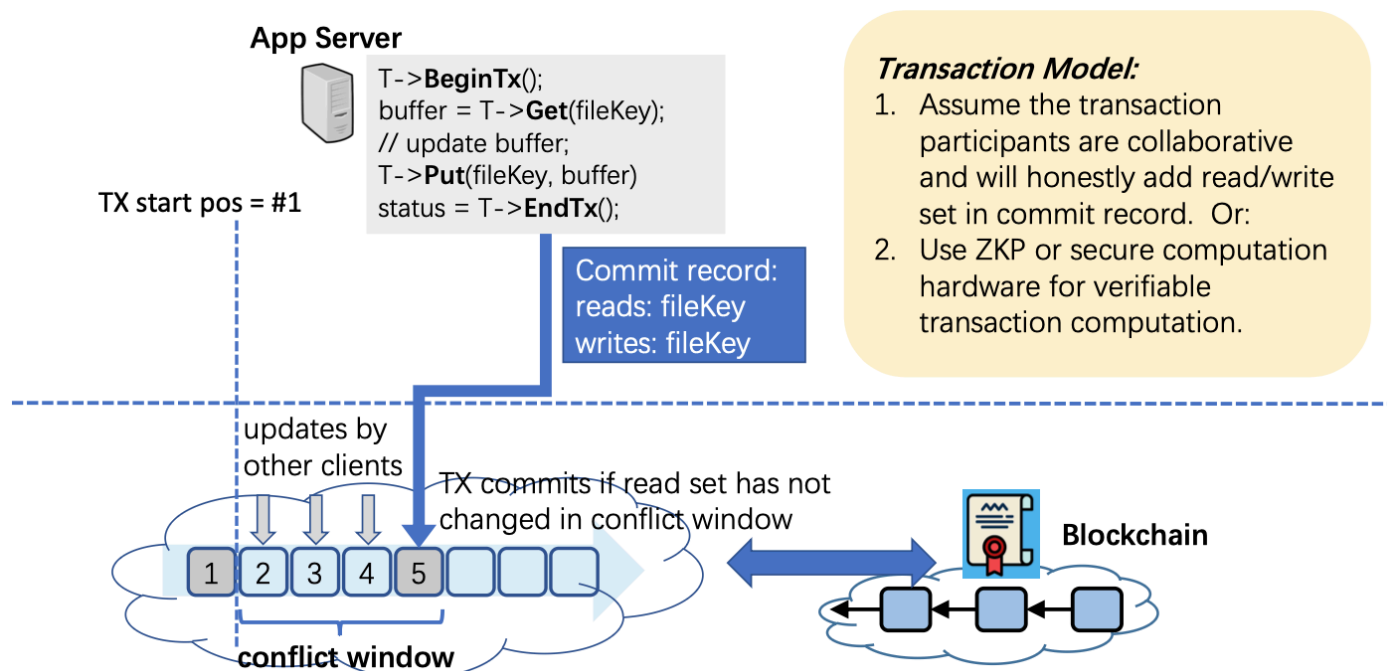


그림 8: oG 스토리지 키-값 스토어에서의 트랜잭션 처리

애플리케이션 서버가 키-값 런타임과 함께 로그를 재생하다가 커밋 레코드를 만나면, 트랜잭션 시작 로그 위치와 커밋 레코드 위치 사이의 모든 로그 항목으로 구성된 충돌 윈도우를 식별한다. 충돌 윈도우 내의 로그 항목들은 따라서 커밋 레코드를 제출한 트랜잭션과 동시에 발생한 키-값 연산을 포함한다. 런타임은 이러한 동시 연산들이 트랜잭션의 읽기 집합에 속한 키들에 대한 갱신을 포함하는지 탐지한다. 만약 그렇다면, 해당 트랜잭션은 중단(abort)된다. 그렇지 않으면, 성공적으로 커밋된다.

이 트랜잭션 모델은 트랜잭션 참여자들이 협력적이며, 올바른 내용을 담은 커밋 레코드를 정직하게 작성한다고 가정한다. 이러한 가정은 탈중앙화 환경에서는 지나치게 강할 수 있으나, 특정 애플리케이션에서는 여전히 실현 가능하다. 예를 들어, 구글 독스와 같은 애플리케이션에서는 사용자가 일반적으로 신뢰할 수 있는 다른 사용자와 접근 권한을 공유한다. 이 가정이 성립하지 않을 경우, 트랜잭션의 코드를 oG 로그에 저장하고, 영지식 증명(zero-knowledge proof)과 같은 검증 가능한 계산 메커니즘이나 신뢰 실행 환경(TEE)을 갖춘 하드웨어를 트랜잭션 실행자가 활용하여 커밋 레코드의 유효성을 탐지할 수 있다.

8 OG 연산 네트워크 위의 서비스 마켓플레이스

탈중앙화 AI 운영체제로서 OG는 또한 탈중앙화 컴퓨팅 자원을 서비스로 관리하는 별도의 플랫폼을 제공한다. 이는 컴퓨팅 자원을 연산 네트워크로 조직하고, 서비스가 검증 가능성(verifiability)을 갖춘 상태로 배포·등록되도록 하여, 사용자가 탈중앙화 환경에서 신뢰할 수 있는 서비스를 선택할 수 있게 한다.

OG 연산 네트워크는 사용자와 서비스 제공자가 피어 투 피어 방식으로 서비스 가격을 결정하는 자유로운 마켓플레이스로 설계되었다. 서비스 제공자는 자유롭게 서비스를 견적할 수 있고, 사용자는 자신에게 적절하게 책정된 서비스를 선택할 수 있다. 시스템은 모든 서비스 결과가 요청을 제출한 사용자에게 의해 검증될 수 있다고 가정한다. 신뢰 모델에서 서비스 제공자는 사용자보다 신뢰 가능하지만 시빌 공격(sybil attack)을 가할 정도로 강력하지는 않은 주체로 취급된다. 따라서 사용자는 단일 요청의 비용을 잃을 위험에 직면할 수 있다. 즉, 사용자가 요청을 제출하고 수수료를 스마트 컨트랙트에 보냈는데 서비스 제공자가 악의적이라면, 사용자는 수수료를 지불하고도 서비스 결과를 받지 못할 수 있다. 이 경우 사용자는 네트워크의 다른 서비스 제공자를 선택해 요청을 재시도할 수 있다.

서비스 제공자는 먼저 자신이 제공하는 서비스 유형, 각 유형의 가격, 특정 검증 메커니즘에 대한 정보를 스마트 컨트랙트에 등록해야 한다. 사용자가 서비스를 이용하고자 하면, 해당 서비스 제공자를 위해 일정 금액의 수수료를 스마트 컨트랙트에 선결제한다. 그 후 사용자는 서비스 제공자에게 요청을 보낸다할 수 있고, 서비스 제공자는 잔여 수수료가 충분한지 여부에 따라 서비스 결과를 반환할지 결정한다. 모든 요청과 응답에는 각각 사용자와 서비스 제공자의 서명이 포함된다. 사용자는 각 응답을 검증하고, 검증에 실패하면 더 이상 요청을 보내지 않기로 결정할 수 있다. 서비스 제공자는 언제든지 사용자의 서명이 포함된 요청 기록을 스마트 컨트랙트에 제출해 정산할 수 있다. 정산이 완료되면, 선결제된 수수료의 해당 부분이 서비스 제공자의 계정으로 전송된다.

성능상의 절충점은 서비스 제공자가 스마트 컨트랙트에 정산 요청을 얼마나 자주 제출하느냐에 있다. 지연 시간이 긴 서비스의 경우 제공자는 사용자 요청마다 정산을 제출해 보상을 받는 다할 수 있다. 그러나 지연 시간이 짧은 서비스의 경우 제공자는 여러 사용자 요청을 하나의 정산으로 묶어야 할 수 있다. 또 다른 고려사항은 사용자 요청 기록을 정산을 위한 스마트 컨트랙트 입력으로 어떻게 구성하느냐이다. 단순히 모든 요청을 기록으로 제출하면 지나치게 많은 온체인 가스 소비가 발생할 수 있다. 따라서 OG는 온체인 정산 비용을 최적화하기 위해 영지식 증명(zero-knowledge proof) 메커니즘을 채택한다.

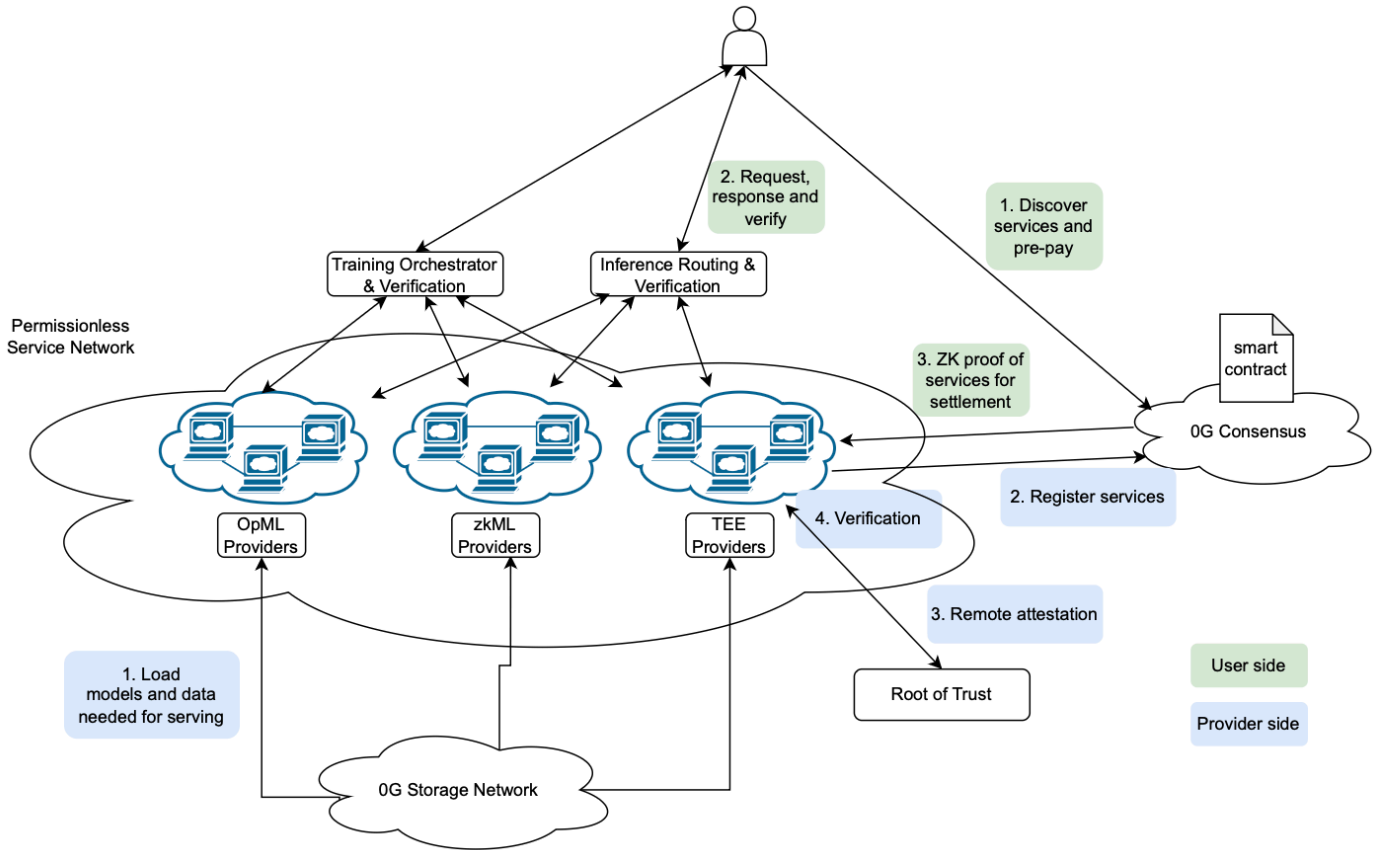


그림 9: oG 연산 네트워크와 서비스 마켓플레이스의 아키텍처.

oG 연산 네트워크는 서비스 제공자가 자신의 서비스를 마켓플레이스 스마트 컨트랙트에 쉽게 등록할 수 있도록 유용한 SDK와 API를 제공하며, 사용자가 적절한 서비스를 편리하게 발견하고 원활히 상호작용할 수 있게 한다. 서비스에는 AI 모델 추론 및 학습 작업자(worker), 데이터 캐싱과 데이터 가져오기(fetching) 서비스 등이 포함될 수 있다.

oG 스토리지와 DA 네트워크와 결합할 때, oG 연산 네트워크와 서비스 마켓플레이스는 dAIOS의 인프라 기초를 완성하여 AI 애플리케이션과 플랫폼이 Web2에서 Web3로 쉽게 이전하도록 돕고, AGI 민주화를 향한 길을 닦는다.

9 로드맵

추후 공개 예정(TBA).

참고문헌

[1] Mustafa Al-Bassam, Alberto Sonnino, and Vitalik Buterin. Fraud and data availability proofs: Maximising light client security and scaling blockchains with dishonest majorities, 2019.

[2] Ethereum White Paper. <https://ethereum.org/en/whitepaper/>, 2020.

[3] ORA-OpML. <https://docs.ora.io/doc>.

- [4] Optimism. <https://www.optimism.io/>.
- [5] Celestia. <https://celestia.org/>.
- [6] EigenDA. <https://www.blog.eigenlayer.xyz/tag/eigenda/>.
- [7] EigenLayer. <https://www.eigenlayer.xyz/>.
- [8] BabylonChain. <https://babylonchain.io/>.