

라이브피어(Livepeer)

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.

탈중앙화된 라이브 비디오 스트리밍 네트워크를 위한 프로토콜 및 경제적 인센티브

Doug Petkanics doug@livepeer.org

Eric Tang eric@livepeer.org

초록 (Abstract)

****Livepeer 프로젝트는 완전한 탈중앙화, 고확장성, 암호 토큰 인센티브 기반의 라이브 비디오 스트리밍 네트워크 프로토콜을 제공하는 것을 목표로 합니다. 이 솔루션은 탈중앙화된 개발(Web3) 스택에서 라이브 미디어 계층(Live Media Layer) 역할을 수행할 수 있도록 설계되었습니다.**

또한, Livepeer는 기존 방송사들이 사용하는 중앙화된 송출 인프라에 대한 경제적으로 효율적인 대안을 제공합니다.

이 문서에서는 Livepeer 프로토콜을 설명합니다. 이는 위임 지분 기반(Delegated Stake-based) 구조를 통해, 게임 이론적으로 안전한 방식으로 방송 네트워크 참여자들에게 인센티브를 제공하는 프로토콜입니다. 본 문서에서는 탈중앙화된 작업 검증의 확장 가능한 솔루션과, 인플레이션 기반 토큰 시스템에서 발생할 수 있는 무의미한 작업(useless work)을 방지하기 위한 방안도 제시합니다.

참고: 본 백서는 2017년 4월에 최초로 발표되었습니다.

이후 2018년 12월에 “Streamflow”라 불리는 확장성 제안이 추가로 제시되어, 본 문서의 여러 개념을 개선 보완하고 있습니다. 자세한 내용은 [Streamflow Proposal](#)에서 확인할 수 있습니다.

소개 및 배경 (Introduction and Background)

탈중앙화 웹(Web3)의 비전은 최근 몇 년간 현실화되기 시작했습니다. [Ethereum](#)은 신뢰 없는 컴퓨팅(trustless computing)을, [Swarm](#)과 [IPFS/Filecoin](#)은 탈중앙화된 저장 및 콘텐츠 배포를 가능하게 했습니다. 또한 Bitcoin과 다양한 토큰 프로젝트들은 가치의 P2P 전송을 실현하고 있으며, [Blockstack](#) 및 [ENS](#)와 같은 탈중앙화 네임 레지스트리는 콘텐츠 및 아이덴티티에 접근 가능한 인간 친화적 이름을 제공합니다.

이러한 요소들은 탈중앙화 애플리케이션(DApp)이 구축될 수 있는 기반을 형성하며, 현재는 주로 정적이거나 업데이트가 드문 웹 또는 모바일 콘텐츠 형태로 구현되고 있습니다. 그러나 아직까지 DApp은 스트리밍 미디어 및 데이터를 개방적이고 탈중앙화된 방식으로 포함시킬 수 있는 능력을 갖추지 못했습니다.

Livepeer 프로젝트의 목표는 인터넷 상의 라이브 비디오 방송을 탈중앙화하는 것입니다.

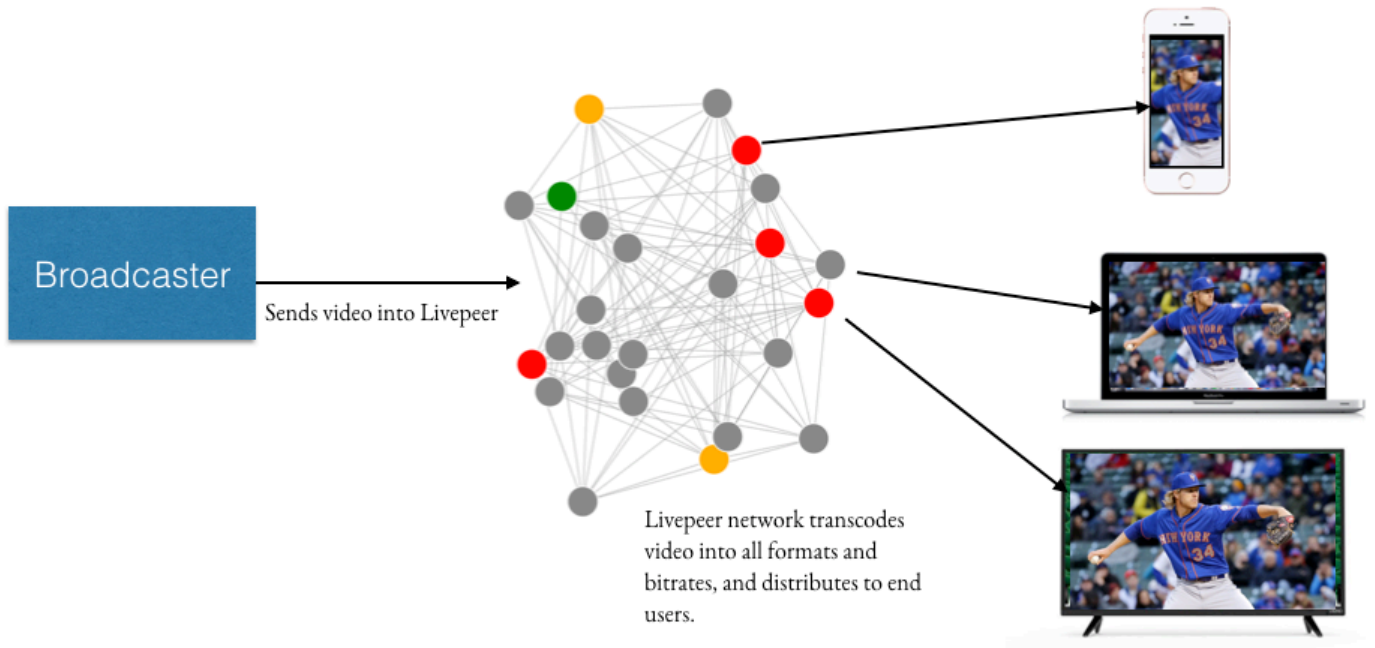
[Livepeer Project Overview](#)는 현재 인터넷 라이브 비디오의 상태를 잘 설명하고 있습니다. 본 백서는 주로 Livepeer의 암호경제적 프로토콜(cryptoeconomic protocol) 구조에 초점을 맞추며, 비즈니스적 측면보다는 기술적 세부사항을 다룹니다. 요약하자면, 현재 라이브 스트리밍은 빠르게 성장하고 있으나 중앙화되어 있으며 비용이 높습니다. 반면, 완전 탈중앙화된 P2P 솔루션 — 즉, 각 노드가 자신의 연산력과 대역폭을 제공하여 라이브 비디오 스트리밍을 지원하는 방식 — 은 더 개방적이고 확장 가능한 구조를 가질 수 있습니다. 이 경우, 동시에 처리할 수 있는 연결 수에 사실상 한계가 없어집니다.

현재 이러한 기술은 일정 부분 존재하지만, 이 기능을 제공하는 노드를 사용자들이 직접 운영하도록 유인할 인센티브가 없었으며, 또한 인터넷 전체가 혜택을 받을 수 있는 오픈 프로토콜 개발에 필요한 자금 지원도 충분히 이루어지지 않았습니다.

그러나 최근 등장한 암호 토큰 기반 프로토콜(Crypto-token powered protocols) [2,3] 덕분에, 이제는 사용자들이 연산력과 대역폭을 기여하도록 인센티브를 제공하면서, 동시에 최신 표준과 포맷을 지원하는 오픈 미디어 서버 솔루션 개발 자금을 확보할 수 있는 기회가 생겼습니다. 게다가 토큰 기반 프로토콜에서 나타나는 경제적 메커니즘을 고려하면, Livepeer 네트워크를 사용하는 방송자의 비용이 기존 중앙화 솔루션보다 저렴해질 가능성도 존재합니다.

Livepeer 기술과 프로토콜이 완성되면, 사용자는 다음과 같은 과정으로 참여할 수 있습니다:

1. 카메라, 휴대폰, 화면 또는 웹캠으로 영상을 캡처하여 Livepeer 네트워크로 전송합니다.
2. 네트워크 내 노드들은 해당 영상을 모든 지원 기기에서 재생 가능한 포맷으로 인코딩합니다. 이 노드 운영자들은 방송자로부터 ETH로 수수료를 받거나, 프로토콜 토큰을 통해 평판을 쌓아 향후 더 많은 작업을 수행할 권리를 얻게 됩니다.
3. 네트워크의 어떤 사용자든 스트림을 요청할 수 있으며, 영상은 거의 실시간(near-realtime)으로 자동 배포됩니다.



라이브 비디오 스택 (The Live Video Stack)

라이브 비디오를 방송하기 위한 기술 스택은 오랜 시간에 걸쳐 발전해 왔으며, 여러 계층으로 구성되어 있습니다.

방송자는 소스에서 비디오를 캡처하고, 미디어 서버(media server)를 통해 비디오를 다양한 포맷으로 인코딩·트랜스코딩한 후, 이를 네트워크를 통해 배포하며, 최종 소비자가 높은 품질로 영상을 시청할 수 있도록 해야 합니다.

또한, 이 스택을 설계할 때는 누가 대역폭 비용을 부담해야 하는가(방송자 vs 시청자)와 같은 경제적 문제도 함께 고려해야 합니다.

현재의 일반적인 라이브 스트리밍 플랫폼은 RTMP, HLS, MPEG-Dash 등의 비디오 포맷을 지원해야 하며, 코덱은 H.264와 VP8을 기본적으로 포함합니다. 근래에는 H.265/HEVC, VP9, AV1과 같은 신형 코덱이 등장하면서, 소비자들이 고화질 비디오에 익숙해짐에 따라 이러한 포맷이 점점 보편화될 것입니다.

예를 들어, HLS만 보더라도 [Apple의 가이드라인](#)에 따르면 비트레이트는 145kb/s에서 7800kb/s까지 다양하게 설정되어야 합니다. 이는 다양한 기기와 네트워크 조건을 모두 지원하기 위함이지만, 결과적으로 라이브 비디오 방송의 복잡성과 비용을 상당히 증가시키는 요인이 됩니다.

현재의 탈중앙화 개발 스택(Web3)은 파일 전송(file transfer), 결제(payments) 등 일부 계층에 대한 솔루션은 보유하고 있지만, 라이브 비디오의 캡처, 인터페이스, 트랜스코딩, 처리, 전송 계층을 위한 솔루션은 존재하지 않습니다.

이 문제를 해결하기 위해 Livepeer는 [Livepeer Media Server \(LPMS\)](#)를 도입했습니다. LPMS는 오픈소스 미디어 서버 구현체로, DApp 개발자 및 기존 방송사가 자신들의 애플리케이션에 라이브 비디오 기능을 손쉽게 통합할 수 있도록 지원하는 필수 기능을 제공합니다. [자세한 내용은 여기에서 확인할 수 있습니다.](#)

단독 애플리케이션 형태로도 LPMS 위에 라이브 비디오 서비스를 구축할 수 있지만, 이 경우 여전히 중앙화된 구조를 가지며 전통적인 방식으로만 확장할 수 있습니다. 그러나 Livepeer 네트워크의 모든 노드가 LPMS를 실행하고, 프로토콜의 경제적 인센티브 구조가 노드들이 트랜스코딩 및 비디오 전송을 위해 연산 자원과 대역폭을 기여하도록 유도한다면, 자동 확장(Self-scaling)형, 사용량 기반(Pay-as-you-go) 네트워크가 개발자에게 제공됩니다.

즉, 개발자는 단지 자신의 라이브 스트림을 네트워크로 전송하기만 하면, 확장성, 결제, 미디어 호스팅 등 복잡한 구현 세부사항이 추상화되어 자동으로 처리됩니다.

Livepeer 프로토콜 (Livepeer Protocol)

Livepeer 프로토콜은 라이브 스트리밍 생태계의 다양한 참여자들이 보안적이고 경제적으로 합리적인 방식으로 참여할 수 있도록 정의합니다.

프로토콜이 해결해야 할 주요 영역은 소스에서 다수의 소비자에게 실시간으로 라이브 비디오를 성능 좋고 확장 가능한 방식으로 배포하는 문제, 네트워크 참여를 보안적이고 게임 이론적으로 합리적으로 유도하기 위한 경제적 인센티브 구조 등 두가지 입니다.

이 백서는 경제 프로토콜 측면에 초점을 맞추며, 프로토콜의 보안성과 경제적 정렬을 설명합니다. 프로토콜의 상위 목표는 다음과 같습니다.

- 어떤 노드든 네트워크에 라이브 비디오를 전송하고, 필요 시 여러 포맷과 비트레이트로 트랜스코딩되도록 비용을 지불할 수 있게 함
- 어떤 노드든 네트워크에서 비디오를 요청할 수 있게 함
- 참여자가 자신의 연산력과 대역폭을 제공해 비디오의 트랜스코딩과 배포를 지원하고 이에 대한 보상을 받도록 함

탈중앙화 네트워크에서 참여자가 기여한 작업량에 비례해 보상을 받는 구조에서는 보안을 위해 다음 두 가지 문제를 해결해야 합니다.

- 노드가 수행한 작업이 올바르게 처리되었는가?
- 노드가 네트워크에 실제로 기여한 “진짜 작업”에 대해 보상을 받고 있는가, 아니면 부당한 방식으로 토큰을 얻고 있는가?

Livepeer 프로토콜은 작업의 검증과 가짜 작업의 방지를 모두 해결하도록 설계되었으며, 네트워크의 자동 확장성과 시간이 지남에 따른 프로토콜 거버넌스 기능을 함께 제공합니다.

비디오 세그먼트 (Video Segments)

Livepeer 내에서 미디어의 핵심 단위는 세그먼트(segment)입니다. 세그먼트는 시간 길이 t 를 갖는 오디오와 비디오가 다중화된(time-sliced, multiplexed) 데이터 조각입니다.

Livepeer 네트워크의 각 세그먼트는 고유하며, 해당 세그먼트가 특정 방송자(broadcaster)에 의해 생성되었음을 검증할 수 있는 암호학적 증거(cryptographic evidence)를 포함합니다. 하나의 스트림(stream)은 여러 개의 연속적인 세그먼트로 구성되며, 각 세그먼트에는 올바른 순서를 식별할 수 있는 시퀀스 번호(Sequence Number)가 포함됩니다.

세그먼트에는 다음과 같은 필드가 포함됩니다.

비디오 세그먼트 필드 (Video Segment Field)	설명 (Description)
StreamID	해당 세그먼트가 속한 원본 노드와 스트림을 식별합니다.
SequenceNumber	세그먼트가 원본 스트림에서 속하는 순서를 나타냅니다.
DataPayload	세그먼트 내의 오디오/비디오를 나타내는 바이너리 메타데이터 및 데이터입니다.
DataHash	데이터 페이로드(DataPayload)의 해시 값입니다.
BroadcasterSignature	방송자의 서명으로, $\text{Priv}(\text{StreamID}, \text{SequenceNumber}, \text{hash}(\text{StreamID}, \text{SequenceNumber}, \text{DataHash}))$ 에 대한 서명입니다. 이 서명은 방송자가 해당 세그먼트를 실제 데이터로 주장함을 검증하는 데 사용됩니다.

Livepeer 프로토콜은 일반적으로 트랜스코딩(transcoding), 배포(distribution), 결제(payments) 작업 단위로 세그먼트를 사용합니다.

라이브피어 토큰 (Livepeer Token)

Livepeer Token (LPT)은 Livepeer 네트워크의 프로토콜 토큰이지만, **교환 매개(currency token)**로 사용되지는 않습니다.

방송자(Broadcaster)는 네트워크에서 영상을 송출할 때 **Ethereum의 Ether(ETH)**를 사용하며, 연산력과 대역폭을 제공하는 노드들은 방송자로부터 수수료(fees)의 형태로 **ETH**를 보상받습니다.

LPT는 네트워크에서 작업을 수행하고자 하는 참여자들이 스테이킹(staking)하는 **지분 토큰(staking token)**으로, 이를 통해 네트워크 내 작업 분배를 조정하고, 작업이 정직하고 올바르게 수행되도록 보안을 제공합니다.

- 위임 지분 증명(Delegated Proof of Stake, DPoS) 시스템 내에서의 본딩 메커니즘(bonding mechanism) 역할을 합니다. 스테이크는 비디오 트랜스코딩(transcoding)과 작업 검증

(validation)에 참여하는 트랜스코더(Transcoder) 또는 검증자(Validator)에게 위임됩니다. 프로토콜 위반 시 발생하는 슬래싱(Slashing)은 네트워크를 공격으로부터 보호하기 위한 핵심 요소입니다.

- 네트워크 내 작업 분배를 조정하는 라우팅 메커니즘으로 작동합니다. 스테이킹 및 위임된 LPT의 양에 비례해 작업이 네트워크를 통해 분배됩니다.
- Livepeer 생태계에 특화된 회계 단위(Unit of Account)로서 기능합니다. 이는 SectorCoin 개념의 기반을 이루며, 향후 DVR, 자막(Closed Captioning), 광고 삽입 및 수익화(Ad Insertion/Monetization), 분석(Analytics) 등의 부가 서비스가 Livepeer 생태계에 통합될 때 스테이킹된 LPT를 기반으로 한 보안성을 활용할 수 있게 합니다 [4].

LPT의 초기 분배(Initial Allocation)는 다양한 참여자가 네트워크 내 역할을 수행할 수 있도록 배포되며, 이후에는 알고리즘적으로 프로그래밍된 발행 정책에 따라 추가 발행됩니다. 자세한 내용은 [토큰 분배\(Token Distribution\)](#) 섹션을 참조하세요.

Ethereum 및 다수의 ERC-20 토큰 [16]과 동일하게, LPT는 10^{18} 단위로 분할 가능하며, 큰 단위(LPT)는 스테이킹 등 사용자 수준의 트랜잭션에, 작은 단위는 프로토콜 내부 회계(accounting)에 사용됩니다.

프로토콜 내 역할 (Protocol Roles)

다음으로, 프로토콜을 이해하기 위해 네트워크 내에서 수행되는 다양한 역할을 정의합니다.

Livepeer 노드(Livepeer Node)란 Livepeer 소프트웨어를 실행하는 모든 컴퓨터를 의미합니다.

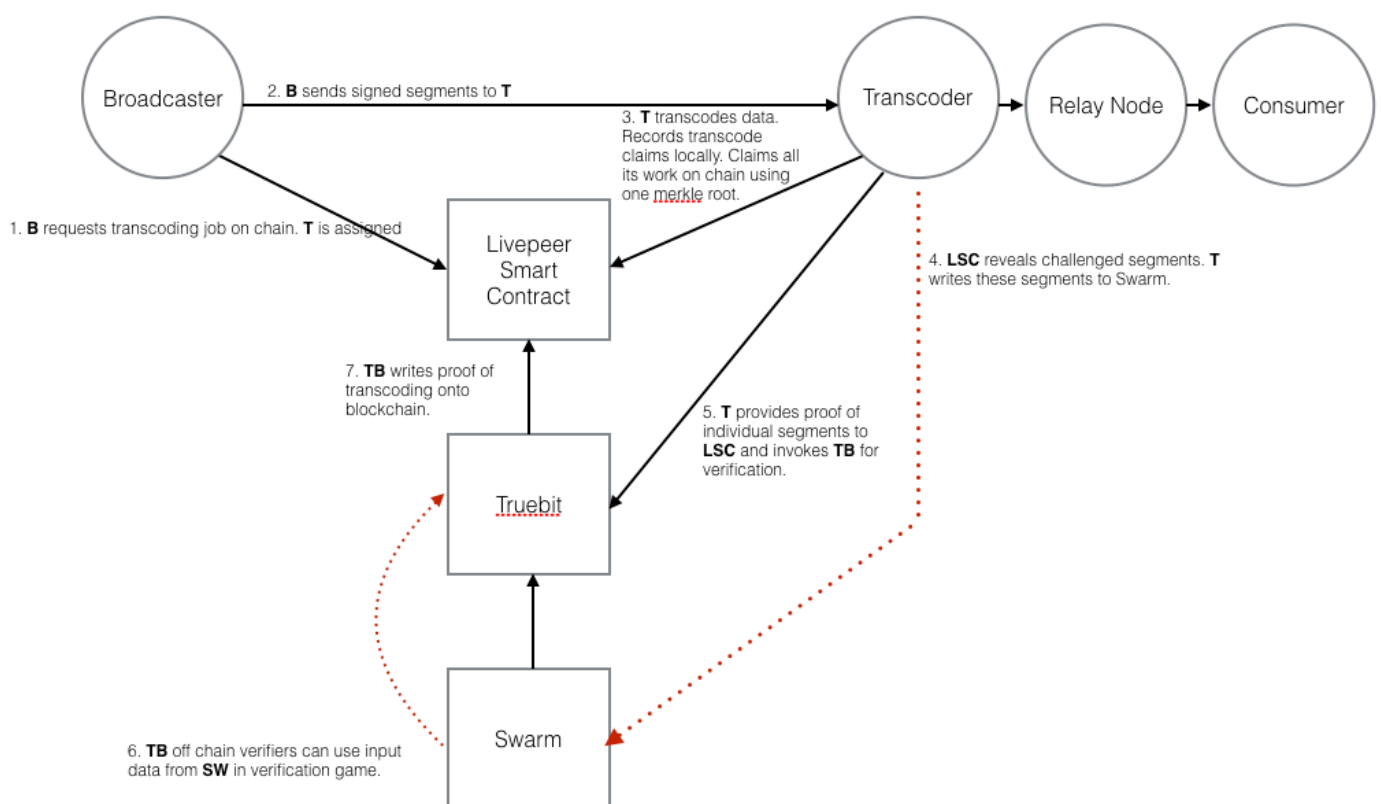
노드 역할 (Node Role)	설명 (Description)
Broadcaster	원본 스트림을 발행하는 Livepeer 노드
Transcoder	스트림을 다른 코덱, 비트레이트, 또는 패키징 포맷으로 트랜스코딩(transcoding)하는 Livepeer 노드
Relay Node	라이브 비디오의 전송 및 프로토콜 메시지 전달에 참여하지만, 반드시 트랜스코딩을 수행하지는 않는 노드
Consumer	스트림을 요청하는 Livepeer 노드로, 영상을 시청하거나 앱 또는 DApp 사용자에게 게이트웨이를 통해 제공하는 역할을 수행

위의 노드 역할 외에도, Livepeer 프로토콜에서는 다음과 같은 **시스템 역할(System Roles)**이 함께 정의됩니다. 이 문서에서는 특정 시스템을 예시로 들었지만, 동일한 기능과 암호경제적 보증(cryptoeconomic guarantees)을 제공하는 다른 시스템으로 대체될 수도 있습니다.

설명 (Description)

Swarm	콘텐츠 주소화(content-addressed) 저장 플랫폼입니다. SWEAR 프로토콜 [7, 12]을 통해 검증 과정 중 데이터가 일시적으로 저장되어 있음을 보장할 수 있습니다. (본 문서에서는 Swarm을 언급하지만, 높은 확률로 데이터 가용성을 보장할 수 있는 다른 콘텐츠 주소화 저장 시스템으로 대체할 수 있습니다.)
Livepeer Smart Contract	Ethereum 네트워크 상에서 실행되는 스마트 계약 [1]
Truebit	체인 상에서 수행된 연산의 정확성을 보장하는 블랙박스 검증 프로토콜 [6] (http://truebit.io)

다음은 아래에서 설명할 **작업 검증 절차(Work Verification Process)**에서 각 역할이 서로 어떤 방식으로 상호작용하는지를 시각적으로 보여줍니다.



세그먼트는 Broadcaster에서 Transcoder를 거쳐 Consumer로 전달됩니다.

Transcoder는 작업 검증 절차에 참여하기 위해 서명(Signature)과 작업 증명(Proof of Work)을 확보합니다.

트랜스코더(Transcoder)에 대한 참고 사항

트랜스코더는 Livepeer 생태계에서 가장 핵심적인 역할을 수행합니다. 이들은 입력 스트림을 받아, 이를 다양한 포맷으로 신속히 변환하여 저지연(low latency) 환경에서 배포할 수 있도록 합니다.

따라서 트랜스코더는 높은 가용성(High Availability), 효율적이고 강력한 하드웨어 (GPU 가속 트랜스코딩 포함 가능), 안정적인 고대역폭 네트워크 연결, 견고한 DevOps 운영 능력과 같은 조건을 갖추수록 더 큰 이점을 얻습니다.

트랜스코더는 스트리밍 트랜스코딩 작업을 맡을 때, 네트워크에서 이탈(churn)하는 일이 최소화되어야 합니다. 트랜스코딩 중에 네트워크에서 이탈하면 서비스 품질이 저하되기 때문입니다.

Livepeer 네트워크는 다수의 참여자가 트랜스코더 역할을 수행하도록 확장할 수 있으며, 이들은 그에 따른 토큰 보상을 받을 수 있습니다.

그러나 트랜스코더 역할은 모든 참여자에게 자동으로 부여되는 것이 아니라, 대부분의 네트워크 참가자로부터 위임(delegation)되는 특수한 역할입니다. 이러한 위임 구조는 방송자에게 신뢰할 수 있는 고품질 네트워크 서비스를 보장하기 위해 설계된 것입니다. 자세한 위임 구조는 이후 섹션에서 다룹니다.

합의 (Consensus)

Livepeer는 2계층 합의 시스템(two-layer consensus system)을 사용합니다. LPT 원장(ledger)과 트랜잭션은 기초 블록체인(예: Ethereum)에 의해 보안이 보장됩니다. 즉, LPT 토큰의 전송이나 시스템 내 모든 트랜잭션은, 해당 기초 체인의 작업증명(Proof of Work) 또는 지분증명(Proof of Stake) 블록체인과 동일한 수준의 보안으로 확인된 것으로 간주됩니다.

두 번째 계층은 새롭게 생성된 LPT의 분배(distribution)를 결정합니다. 이 부분은 Livepeer 스마트 계약(Livepeer Smart Contract)과 프로토콜에 참여하는 다양한 행위자(actor)에 의해 관리됩니다. 이 계층에서는 블록의 수용이나 검증을 위한 별도의 합의 과정이 필요한 것은 아니지만, 프로토콜은 각 참여자들의 역할과, 역할을 수행하지 못했을 경우 페널티(슬래싱, slashing)가 적용되는 조건을 명확히 정의합니다.

이 두 번째 합의 계층**은 Bitshares, Steem, Tendermint, Casper 등의 시스템 [5, 9, 10, 11]에서 영감을 받은 위임 지분 증명(Delegated Proof of Stake, DPoS) 모델을 기반으로 합니다.

네트워크 내 검증자(Validator)의 역할은 트랜스코더(Transcoder)가 수행합니다.

모든 사용자는 자신의 스테이크를 특정 트랜스코더에게 위임(delegate)할 수 있으며, 위임받은 트랜스코더는 네트워크 내에서 트랜스코딩 작업을 수행하고, 작업 검증 프로토콜(Work Verification Protocol)에 참여하며, 지정된 주기마다 온체인 함수(on-chain function)를 호출하여 자신의 작업을 검증해야 합니다.

프로토콜은 수수료(fees)와 새롭게 생성된 토큰을 분배하며, 부정행위를 한 참여자(actor)의 스테이크를 슬래싱합니다. 검증 결과(validation result)는 Truebit을 통해 온체인에 기록되며, 이를 통해 방송자(broadcaster)와 트랜스코더 간의 분쟁(dispute) 가능성을 제거합니다.

본딩 및 위임 (Bonding + Delegation)

Livepeer에서 네트워크 내 지분(stake)을 표시하기 위해, 노드는 일정량의 LPT를 본딩(Bond)해야 합니다. 이는 Bond () 트랜잭션을 통해 수행되며, 본딩된 LPT는 Unbond () 명령이 실행될 때까지 스마트 계약에 묶여(lock-up) 있게 됩니다.

Unbond ()가 호출되면, 노드는 언본딩 상태(Unbonding State)로 전환되며, 이 상태는 UnbondingPeriod 동안 지속됩니다. UnbondingPeriod가 완료되면 LPT를 인출(withdraw)할 수 있습니다.

본딩된 LPT는 트랜스코더에게 위임(delegate)되는 데 사용됩니다. 네트워크는 동시에 N개의 활성 트랜스코더(active transcoders)를 지원하며, N 값은 조정 가능한 네트워크 파라미터입니다. 모든 노드는 Transcoder () 트랜잭션을 통해 자신이 트랜스코더가 되기를 원한다고 표시할 수 있으며, 프로토콜은 각 라운드의 시작 시점에 (자신의 스테이크 + 다른 노드로부터 위임받은 스테이크)의 합이 가장 큰 상위 N개의 트랜스코더를 선택합니다. 또한 대기열(waitlist)에서 무작위로 한 명의 트랜스코더를 추가로 선정합니다.

Livepeer에서 새롭게 생성된 토큰은, 본딩된 노드가 프로토콜에 따라 올바르게 동작하는 트랜스코딩 노드에 위임한 경우, 그 본딩된 양과 작업량(work amount)에 비례하여 (수수료 제외) 분배됩니다. 만약 위임받은 트랜스코더가 프로토콜을 위반하거나 비정상적인 행동을 하면, 해당 트랜스코더에 위임한 본딩 노드의 스테이크는 일정 비율만큼 슬래싱(slashed)됩니다.

트랜스코더에게 본딩 및 위임을 수행한 노드는, 트랜스코더가 네트워크 상에서 트랜스코딩 작업을 통해 얻은 수수료(fees)의 일부를 함께 받습니다. 즉, 실제 작업을 수행하는 노드는 방송자로부터 받은 작업 수수료를 통해 보상을 얻게 됩니다.

이후 본 문서에서 “Delegator(위임자)”라는 용어는, 자신의 스테이크를 트랜스코더 후보에게 위임(delegate)한 본딩된 노드를 의미하며, 자신을 트랜스코더로 등록하는 대신 타 노드에게 위임한 참여자를 지칭합니다.

요약하자면, 참여자가 자신의 스테이크를 본딩하는 이유는 다음과 같습니다.

- 네트워크에 높은 품질의 서비스를 제공하는 효율적인 트랜스코더에게 위임하여, 방송자에게 가치를 제공하고 네트워크의 신뢰성을 높이기 위함
- 스테이크 비율에 따라 평판(reputation)과 향후 작업 할당(future work allocation)을 쌓기 위함
- 트랜스코더가 생성한 수수료(fees)를 공유하여 수익을 얻기 위함
- 직접 트랜스코더가 되어 네트워크에서 작업을 수행하기 위함

Transcoder() 트랜잭션

노드는 Transcoder () 트랜잭션을 제출함으로써 자신이 트랜스코더로 활동할 의사가 있음을 표시합니다. 이 트랜잭션에는 다음 세 가지 속성이 공개됩니다.

- **PricePerSegment**: 비디오의 한 세그먼트를 트랜스코딩할 때 수락 가능한 최소 가격
- **BlockRewardCut**: 본딩된 노드가 트랜스코더에게 트랜스코딩 서비스 대가로 지불하는 블록 보상의 비율(%)
(예: 2%. 본딩된 노드가 100 LPT를 블록 보상으로 받는다면, 트랜스코더에게 2 LPT를 지불)
- **FeeShare**: 트랜스코더가 방송 작업(broadcasting jobs)으로부터 받은 수수료 중 본딩된 위임자에게 공유할 비율(%) (예: 25%. 트랜스코더가 100 ETH의 수수료를 받는다면, 그중 25 ETH를 위임자에게 분배)

트랜스코더는 다음 트랜스코딩 라운드가 시작되기 전, RoundLockAmount 기간(라운드의 일정 비율, 예: 10% = 2.4시간) 동안 자신의 가용성 및 정보를 업데이트할 수 있습니다. 즉, 트랜스코더는 1일 (RoundLength) 동안 지속되는 라운드가 시작되기 2.4시간 전까지 가격 및 수익 배분 정보를 변경할 수 있습니다. 이 기간 동안 본딩된 노드들은 각 트랜스코더의 수수료 비율 및 토큰 보상 비율, 트랜스코더가 제시한 가격 및 네트워크 수요에 따른 예상 수익을 비교 검토한 후, 필요 시 위임된 스테이크를 다른 트랜스코더로 이동할 수 있습니다.

트랜스코딩 라운드가 시작되면 (InitializeRound () 트랜잭션 호출로 트리거됨), 해당 라운드의 활성 트랜스코더는 각 트랜스코더에게 위임된 총 스테이크를 기준으로 결정됩니다. 이 시점에서 스테이크 및 효율(Price/Reward 비율)은 해당 라운드 동안 고정됩니다.

RoundLockPeriod 동안 허용되는 단 한 가지 변경 사항: 후보 트랜스코더 중에서 가장 낮은 세그먼트 가격(PricePerSegment)은 고정되어 변경할 수 없습니다. 하지만 다른 트랜스코더 후보는 자신의 가격을 낮추는 방향으로만 조정할 수 있습니다. 이를 통해 네트워크 내 최저가와 동일하게 조정함으로써, 자신의 스테이크 비중에 따른 네트워크 작업 할당량을 확보할 수 있습니다. 이 기간에는 가격 인상은 허용되지 않습니다.

다음은 위임자가 트랜스코더를 선택할 때 참고할 수 있는 트랜스코더 옵션의 예시입니다.

Transcoder ID	PricePerSegment	BlockRewardCut	FeeShare
1	22 wei	1%	25%
2	30 wei	2%	40%
3	10 wei	4%	1%
...	...	...	...
N	14 wei	0%	2%

가격에 대한 참고 사항

본 문서에서는 가격/세그먼트(price/segment) 단위를 사용하지만, 실제 Livepeer는 가스 회계(gas accounting) 개념을 기반으로 한 모델을 사용할 계획입니다. 이 모델에서는 세그먼트의 비트레이트(bitrate), 인코딩(encoding), 프레임 크기(frame size) 등 작업 매개변수에 따라 필요한 가스 단위(gas units)가 정의됩니다. 현재의 “가격/세그먼트” 표기는 이를 단순화한 형태이며, 실제 인센티브 구조는 가격/가스(price/gas) 형태로 적용될 예정입니다.

방송 및 트랜스코딩 작업 (Broadcast + Transcoding Job)

네트워크에서 트랜스코딩 서비스를 제공할 준비가 된 트랜스코더는 `TranscodeAvailability()` 트랜잭션을 제출하여 자신의 가용성을 알리고, 새로 제출된 작업(Job)을 수행할 수 있는 트랜스코더 풀에 포함됩니다.

방송자가 자신의 스트림을 Livepeer 네트워크에 제출하면, 해당 스트림은 고유한 `StreamID`를 부여받습니다. 이 `StreamID`는 고유 식별자 역할을 할 뿐만 아니라, 원본 노드의 주소(origin node address)를 포함하고 있어 다른 노드들이 해당 스트림을 요청하거나 원본으로 라우팅하는 방법을 알 수 있도록 합니다.

각 스트림은 [비디오 세그먼트\(Video Segments\)](#) 섹션에서 설명한 것처럼 여러 개의 연속적인 세그먼트(Segments)로 구성됩니다.

만약 방송자가 자신의 스트림을 네트워크가 자동으로 트랜스코딩하여, 모든 기기와 사용자에게 적합한 다양한 포맷과 비트레이트로 변환되기를 원한다면, 첫 단계는 온체인에서 트랜스코딩 작업(Job)을 제출하는 것입니다.

이때 생성되는 작업은 고유한 Job ID를 가지며, 입력 데이터는 다음과 같이 구성됩니다.

`Job(StreamID, TranscodingOptions, PricePerSegment)`

`TranscodingOptions`은 출력 비트레이트, 포맷, 인코딩 등의 설정을 정의합니다.

`PricePerSegment`은 방송자가 세그먼트당 지불할 의향이 있는 가격을 명시합니다.

이 트랜잭션이 블록에 포함(mined)되면, 다음 블록의 해시(Blockhash)가 해당 작업을 수행할 트랜스코더를 의사난수(pseudo-random) 방식으로 결정하는 데 사용됩니다. 방송자가 제시한 가격 이하로 작업을 수행할 의향이 있는 모든 트랜스코더가 후보로 고려되며, 블록 해시 값(mod) 후보 트랜스코더의 수(스테이크 가중치 포함)에 따라 선택된 트랜스코더의 인덱스가 결정됩니다.

이 시점에서 방송자는 트랜스코더로 영상 세그먼트를 스트리밍하기 시작할 수 있으며, 이후 아래 프로토콜 절차에 따라 상호작용이 진행됩니다. 또한, 이 프로토콜은 작업 검증(Work Verification) 과정의 일부로 Swarm과 같은 지속형 스토리지 솔루션(persistent storage solution)을 함께 사용합니다.

사전 처리 (Preprocessing)

1. Broadcaster → Livepeer Smart Contract

방송자는 전체 트랜스코딩 작업의 비용을 충당하기 위해 온체인에 예치금(deposit)을 제출합니다. 이 예치금은 언제든지 추가 충전(refill)할 수 있지만, 트랜스코더가 이미 완료한 작업에 대한 보상을 점진적으로 청구(cash in)하는 동안 예치금이 소진될 경우, 트랜스코더는 작업을 중단할 수 있습니다.

다.

작업 수행 (The Job)

2. Broadcaster → Livepeer Smart Contract

`Job(streamID, options, price/segment)`

- 온체인에 작업 요청(Job Request)을 생성하고, 작업 비용을 지불하기 위해 일정량의 ETH를 에스크로(Escrow) 형태로 예치합니다.

3. 프로토콜은 다음 블록의 해시(Blockhash)를 사용하여 해당 작업을 수행할 트랜스코더를 결정론적(deterministic) 방식으로 선택합니다.

4. Transcoder → Broadcaster

- 출력 스트림 ID(output streamID)와 작업이 수락되었음을 증명하는 영수증(receipt)을 전송합니다.

5. Broadcaster → Transcoder

- 서명(signature)이 포함된 스트림 세그먼트(stream segments)를 전송합니다. 이 서명은 입력 데이터의 무결성을 검증하는 데 사용됩니다.

6. Transcoder

- 트랜스코딩을 수행하고, 새롭게 생성된 출력 스트림(output stream)을 네트워크 상에 사용할 수 있도록 제공합니다.

7. Transcoder

- 트랜스코딩된 각 세그먼트에 대해 트랜스코드 영수증(Transcode Receipt)을 저장합니다. 트랜스코드 영수증에는 다음 필드가 포함됩니다.

트랜스코드 영수증 필드 (Transcode Receipt Field)

설명 (Description)

StreamID	해당 세그먼트가 속한 원본 노드와 스트림을 식별합니다.
Sequence Number	세그먼트가 원본 스트림 내에서 속하는 순서를 나타냅니다.
Input Data hash	입력 세그먼트 데이터 페이로드의 해시 값입니다.
Transcoded Data hash	트랜스코딩된 출력 데이터의 해시 값입니다.
Broadcaster segment signature	방송자가 <code>Priv(StreamID, Seq#, Dhash)</code> 에 대해 생성한 서명으로, 이 세그먼트가 해당 방송자의 실제 데이터임을 증명합니다.
Transcoder segment signature	트랜스코더가 위 모든 필드에 대해 생성한 서명으로, 특정 입력 데이터에 대해 특정 트랜스코딩 결과를 수행했음을 증명합니다.

트랜스코더가 더 이상 세그먼트를 수신하지 않는 상황을 감지하면, `ClaimWork()` 함수를 호출하여 자신이 수행한 작업에 대한 보상을 청구할 수 있습니다.

작업 종료 (End Job)

10. Transcoder → Livepeer Smart Contract

`ClaimWork(JobID, StartSegmentSeq#, EndSegmentSeq#, MerkleRoot)` 호출, 트랜스코더는 온체인에서 특정 세그먼트 범위에 대해 작업을 수행했음을 청구합니다. 이때 트랜스코드 영수증(Transcode Receipt) 데이터 전체에 대한 머클 루트(Merkle Root)를 함께 제출하여, 해당 인코딩된 세그먼트들의 내용을 커밋(commit)합니다.

11. 해당 트랜잭션이 블록에 포함될 때까지 대기한 후, 다음 블록의 블록 해시(Blockhash)를 관찰합니다. 프로토콜은 `VerificationRate` 값에 따라 어떤 세그먼트들이 검증 대상이 될지를 결정합니다.

12. Transcoder → Swarm

검증(Verification)을 위해 선택된 세그먼트들의 입력 데이터 페이로드를 Swarm에 업로드합니다. 이때 **SWEAR 파라미터**를 사용하여 데이터가 검증이 완료될 때까지(VerificationPeriod 동안) 충분히 유지되도록 보장합니다.

13. Transcoder → Livepeer Smart Contract

검증이 필요한 각 세그먼트에 대해 온체인에 트랜스코드 청구(Transcode Claim)를 제출합니다. 이때 각 세그먼트의 영수증(Receipt)에 대한 머클 증명(Merkle Proof)도 함께 제공합니다. 스마트 계약은 방송자(Broadcaster)와 트랜스코더의 서명을 검증하여 검증에 필요한 모든 데이터가 존재함을 확인하고, `ClaimWork()`에서 제출된 머클 루트에 대해 머클 증명이 올바른지도 확인합니다.

14. Transcoder → Truebit

`Verify()` 호출, 트랜스코더는 Truebit 스마트 계약에 온체인 호출을 수행하며, 검증 대상 세그먼트의 Swarm 입력 해시(Input Hash)를 제공합니다. (자세한 검증 과정은 다음 섹션에서 설명됩니다.)

15. Truebit → Livepeer Smart Contract

Truebit은 작업의 검증 결과를 온체인에 기록합니다. Livepeer 스마트 계약은 이 결과를 트랜스코더가 제출한 트랜스코딩 청구 결과와 비교합니다.

16. Livepeer Smart Contract

이 시점에서 Livepeer 스마트 계약은 트랜스코더의 작업이 검증되었는지를 판단하는 데 필요한 모든 정보를 갖추게 됩니다. 검증 성공 시: 토큰 할당 알고리즘(token allocation algorithm)에 따라 보상을 계산하고, 에스크로(Escrow)에 보관된 수수료(fees)를 트랜스코더에게 지급합니다. 검증 실패 시: 트랜스코더 및 그에게 스테이킹한 위임자(delegators)는 `FailedVerificationSlashAmount`만큼 슬래싱(slashed)되며, 방송자는 환불(refund)을 받습니다.

방송자는 언제든지 세그먼트 전송을 중단할 수 있으며, 이는 사실상 `EndJob()` 호출과 동일한 효과를 가집니다. 이 시점에서 트랜스코딩이 수행되었고, 작업 증명이 온체인에 청구되었으며, 검증 성공 또는 실패 결과가 보고되었습니다. 모든 관련 정보는 온체인에 기록되어, 수수료 및 토큰 보상 분배, 트랜스코더 및 위임자에 대한 보상 또는 슬래싱 여부를 결정할 수 있습니다. 이제 실제로 작업이 어떻게 검증되는지를 살펴보겠습니다.

작업 검증 (Verification of Work)

트랜스코더가 자신이 트랜스코딩 작업을 수행했다고 주장할 때, 프로토콜은 해당 작업이 실제로 올바르게 수행되었는지를 높은 확률(high probability)로 판단할 수 있어야 합니다. 이를 위해 Livepeer는 [Truebit Protocol](#) [6]의 연구를 확장하여 활용합니다.

Truebit은 한 참가자(솔버, solver)가 실제 작업(이 경우 트랜스코딩)을 수행하고, 다른 참가자들(검증자, verifiers)이 그 결과를 검증하여 오류나 부정행위를 탐지하는 방식으로 작동합니다. 작업은 매우 작은 단계로 분할되며, 검증자들은 솔버의 결과와 기대값이 처음으로 불일치하는 단계를 찾아냅니다. 이 단계만 스마트 계약(판사 역할을 하는 judge)에 의해 온체인에서 재현되어, 어떤 쪽이 올바른 계산을 수행했는지를 판별할 수 있습니다.

경제적 인센티브 설계 또한 포함되어 있으며, 검증자들에게 일부러 오류를 발생시키는 ‘강제 오류(forced errors)’를 통해 검증을 수행하는 행위가 경제적으로 이익이 되도록 유도합니다. 이로 인해 잘못된 검증이나 부정행위를 하는 것은 이익이 되지 않으며, 대신 정직하게 검증을 수행하는 것이 가장 수익성 있는 선택이 됩니다.

Truebit 프로토콜의 단점은 모든 작업을 검증하려면 원래 작업 비용의 약 5~50배의 비용이 소요된다는 점입니다. Livepeer는 Truebit을 블랙박스 검증 시스템(black box verification)으로 활용하지만, 모든 세그먼트를 검증하지는 않습니다.

대신, 무작위로 일부 세그먼트만 검증(random sampling)하여 검증 비용을 줄이고, 검증 실패 시 슬래싱(Slashing)으로 보안을 유지합니다.

Livepeer 내의 VerificationRate 파라미터는 특정 세그먼트가 Truebit 검증 대상으로 선택될 확률을 결정합니다. 작업이 블록체인에 커밋된 이후, 미래 블록의 해시 값(Block Hash)의 무작위성을 기반으로 검증 대상 세그먼트가 선택됩니다.

예를 들어

작업이 블록 N에서 ClaimWork() 호출을 통해 커밋되었다면, 다음 조건을 만족하는 세그먼트는 검증 대상이 됩니다.

만일 $\text{Sha3}(N, \text{BlockHash}(N), \text{Seg\#}) \% \text{VerificationRate} == 0$, 세그먼트 번호 (Segment #)가 이 수식을 만족하면 Truebit 검증을 수행해야 합니다.

트랜스코더는 Verify() 트랜잭션을 호출하여, 후보 세그먼트에 대한 트랜스코딩 청구(Transcode Claim)를 온체인에 제출합니다. Livepeer 스마트 계약은 내부 서명(Signature) 및 머클 증명(Merkle Proof)을 통해 이 청구의 진위(authenticity)를 검증한 후, 해당 세그먼트를 Truebit에 전달하여 실제 검증을 수행합니다.

Truebit의 솔버와 검증자들은 Swarm과 같은 콘텐츠 주소 기반 저장 시스템(content-addressed storage)에서 세그먼트 입력 데이터를 가져옵니다. 트랜스코더는 Swarm 내에서 해당 세그먼트 데이터가 유효하게 존재함을 확인할 책임이 있으며, 선택적으로 SWEAR 프로토콜 [5]의 영수증(receipt)을 확인하여 데이터가 Truebit 검증이 완료될 때까지 유지될 것임을 보장할 수 있습니다. 또한 트랜스코더는 직접 Swarm 노드를 운영하여 데이터가 Truebit 검증 과정에서 접근 가능하도록 보장할 수도 있습니다. 만약 데이터가 Swarm에서 이용 불가능하다고 판단되면, 트랜스코더는 직접 데이터를 제공하거나 기존에 확보된 데이터를 기준으로 ClaimWork()를 호출할 수 있습니다.

Truebit은 계산 결과(성공 또는 실패)를 Livepeer 스마트 계약으로 다시 기록합니다. 이 정보는 프로토콜 내에서 보상 및 슬래싱 계산에 사용됩니다. 트랜스코더는 어떤 세그먼트가 검증 대상이 될지를 사전에 예측할 수 없으며, 부정행위 또는 트랜스코딩 실패 시 다음과 같은 패널티를 받게 됩니다.

- **FailedVerificationSlashAmount:** Truebit 검증에서 실패할 경우 이 금액만큼 슬래싱 됨.
- **MissedVerificationSlashAmount:** 검증 대상 세그먼트에 대해 트랜스코드 청구 및 Truebit 호출을 수행하지 않았을 경우 슬래싱됨.
- 방송자로부터 받을 수수료 상실
- 트랜스코더뿐 아니라, 해당 트랜스코더에게 스테이킹한 모든 위임자(delegators)도 함께 슬래싱됨. 위임자들은 이러한 위험 요소를 고려하여 위임 대상을 선택하게 되며, 트랜스코더는 신뢰를 잃을 경우 수익성 높은 작업 기회를 상실할 수 있음.

정직하게 수행하는 트랜스코더에게 LPT를 스테이킹하는 것이 부정행위로 인한 슬래싱 위험을 감수하면서 수수료를 받는 것보다 항상 더 수익성이 높아야 합니다. 따라서 슬래싱 파라미터(slash parameters)와 검증 비율(verification rate)을 신중히 조정하여 이러한 균형을 유지하는 것이 중요합니다.

Truebit 관련 참고 사항 (A Note On Truebit)

프로토콜은 **Truebit**을 활용하여 완전한 신뢰 불필요(trustless) 검증을 제공하도록 설계되었지만, Truebit이 아직 개발 및 테스트 중인 현실적인 상황에서는 동일한 수준의 **완전한 탈신뢰 검증**을 제공하지 못하더라도, 실제 사용 가능한 대체 검증 솔루션을 활용해야 할 수도 있습니다.

다음은 신뢰 불필요성(trustlessness)의 정도에 따라 정렬된 몇 가지 대안입니다.

1. Livepeer API 기반 오라클 (Livepeer API Based Oracle)

Livepeer 자체에 계산 검증을 위임하는 방식입니다. 매우 중앙화되어 있으며, 테스트 환경 이상의 용도로는 적합하지 않습니다.

2. Oraclize 계산 서비스 (Oraclize Computation Service)

계산 증명(proofs of computation)을 제공하는 회사를 신뢰하는 방식입니다. 해당 회사의 평판은 외부 데이터를 블록체인에 위변조 없이 제공했다는 증명에 의존합니다.

3. 보안 하드웨어 인클레이브 (Secure Hardware Enclaves)

Intel SGX 또는 TownCrier와 같은 서비스는 신뢰할 수 있는 컴퓨팅 환경(trusted computing environment)을 제공합니다. 하드웨어 구현이 올바르고 안전하다는 가정을 신뢰해야 하지만, 이러한 방식은 탈중앙화가 가능하며 외부 감사를 통해 검증될 수 있습니다.

토큰 생성 (Token Generation)

Livepeer는 인플레이션 모델을 기반으로 하며, 새로운 토큰이 시간이 지남에 따라 발행되고 할당됩니다. 발행 일정은 아래 토큰 분배 (Token Distribution) 섹션에 명시된 스케줄에 따라 진행됩니다.

모든 참여자가 프로토콜에 따라 정상적으로 동작할 경우, 새로 생성된 토큰은 본딩된 스테이크(Bonded Stake) 비율에 따라 사용자에게 할당됩니다 (수수료 제외). 트랜스코더(Transcoder)는 온체인 데이터를 기반으로 새로운 토큰의 할당 및 슬래싱을 계산하는 `Reward()` 함수를 호출하여 토큰 발행 과정을 트리거(trigger)하는 역할을 담당합니다.

각 트랜스코더는 매 라운드(Round)마다 한 번씩 `Reward()` 함수를 호출해야 합니다.

`Reward()` 호출 시 수행되는 절차는 다음과 같습니다:

1. **활성 트랜스코더(Active Transcoder)**가 호출했는지 확인합니다.
2. 해당 트랜스코더가 현재 라운드에서 이미 `Reward()`를 호출하지 않았는지 확인합니다.
3. `InflationRate`에 따라 발행할 토큰 수를 계산하고, 그 수만큼 토큰을 신규 발행(Mint)합니다.
4. 트랜스코더의 **BlockRewardCut** 비율에 따라 트랜스코더의 보상 비중을 계산합니다.
5. 계산된 트랜스코더 몫을 트랜스코더의 **본딩된 스테이크(Bonded Stake)**에 분배합니다.
6. 나머지 토큰은 위임자(Delegator) 보상 풀(Reward Pool)에 분배합니다.
7. 트랜스코더의 본딩된 토큰 수량을 업데이트합니다.

`Reward()` 함수를 호출하지 않을 경우 트랜스코더는 일부 토큰 할당량(token allocation)을 상실하게 됩니다. 또한, 해당 트랜스코더의 평판(Reputation)이 하락하여, 향후 위임자(Delegator)들로부터 트랜스코더로 선출될 가능성이 낮아집니다.

슬래싱 (Slashing)

앞서 언급했듯이, 슬래싱이 적용되는 조건은 다음과 같습니다.

- 검증에 실패한 경우
- 검증이 요구될 때 이를 호출하지 않은 경우
- 위임된 스테이크에 비례하여 플랫폼 내에서 요구되는 작업을 수행하지 않은 경우

Ethereum 생태계 위에서 구축함으로써 얻는 이점 중 하나는 Truebit, Swarm/SWEAR와 같은 다른 프로토콜 위에 구축할 수 있다는 **네트워크 효과**입니다. 그러나 이러한 외부 시스템(각각의 외부 의존성과 인센티브를 가짐)에 의존함에 따라, 해당 프로토콜의 결함이나 취약점이 Livepeer 내 슬래싱으로 이어질 가능성도 존재합니다.

예를 들어, Truebit 검증 작업이 오랜 시간 큐에 머무르며 어떤 솔버나 검증자도 이를 청구하지 않는다면, `Reward()`가 호출되기 전에 Livepeer가 해당 검증 결과를 제때 확인하지 못할 수 있습니다. 또는 Swarm 네트워크에 파티션이 발생하여 Truebit 검증자에게 파일이 제때 전파되지 못하는 경우에도 문제가 생길 수 있습니다.

이러한 위험은 Livepeer 프로토콜 참여자들이 자체적으로(in-house) 해당 역할을 수행하도록 인센티브를 제공함으로써 완화할 수 있습니다. 예컨대, 참여자가 Truebit 검증자나 Swarm 노드로 활동하는 것이 자신의 이익과 부합할 수 있습니다. 또 다른 접근으로는 슬래싱 파라미터에 확률적 임계값(probability thresholds) 개념을 도입하는 것입니다. 예를 들어, `VerificationFailureThreshold`와 같은 선

택적 프로토콜 변수를 설정하여, 노드가 검증의 99%를 통과한다면 슬래싱되지 않도록 할 수 있습니다. 이는 네트워크 배포 이전에 추가 연구가 필요한 영역으로 남습니다.

검증 호출 실패에 대한 슬래싱 조건은 Livepeer 프로토콜의 어떤 참여자라도 확인 및 실행할 수 있습니다. 이때 성공적으로 슬래싱 조건을 실행한 참여자는 슬래시 금액의 일정 비율을 보상으로 받는 FinderFee가 적용됩니다.

슬래싱된 나머지 자금은 CommonPool로 유입됩니다. 이 풀의 자금은 프로토콜의 거버넌스 메커니즘에 따라 소각되거나, 생태계 개발과 같은 공통 목적에 할당될 수 있습니다.

토큰 분배 (Token Distribution)

DPoS 스테이킹 알고리즘을 통해 네트워크에서 참여하고 작업을 수행할 수 있는 능력을 나타내는 토큰의 성격상, 초기 Livepeer 토큰 분배는 넓게 분산된 제네시스 상태를 요구하는 다른 DPoS 시스템의 패턴을 따릅니다.

네트워크 제네시스 시점과 초기 단계에 커뮤니티에 대한 초기 할당이 이뤄지며, 수령자는 이를 사용해 트랜스코더(Transcoder) 또는 위임자(Delegator) 역할로 스테이킹할 수 있습니다. 일부는 제네시스 이전에 프로토콜에 기여(사전 작업 및 자금)한 그룹에 할당되고, 일부는 코어 프로젝트의 장기 개발을 위해 출연됩니다.

네트워크 런칭 시점부터는 인플레이션 스케줄에 따라 토큰 발행이 계속되며, 유통 토큰 잔량에 대해 라운드마다 $Inflation \times Rate$ 비율로 토큰이 생성됩니다. 토큰은 프로토콜 내 본딩된 모든 참여자의 스테이크 비율에 따라 발행되므로, 활성 참여를 유도합니다. 참여자는 자신의 비율만큼을 획득하므로 인플레이션으로부터 “보호”됩니다. 반면, 토큰을 보유만 하고 본딩하지 않는 비활성 참여자는 인플레이션으로 인해 네트워크 내 소유 비율이 희석됩니다.

$InflationRate$ 의 초기 목표치는 약 $ParticipationRate$ 비율의 LPT가 본딩되어 활성 참여하도록 유도되도록 설정됩니다 [19]. 예를 들어, $ParticipationRate$ 가 50%라면, 유통 토큰의 절반이 본딩되도록 유인이 설계됩니다. 인플레이션율은 각 라운드마다 알고리즘적으로 조정되어 목표 참여율을 유도합니다. 높은 인플레이션율은 더 많은 본딩을 유도하고, 낮은 인플레이션율은 참여보다 유동성 선호를 증가시킵니다. 네트워크 내 다양한 경제적 요인으로 인해 유동성 선호 vs 네트워크 지분율 사이의 균형점이 형성되어야 합니다.

거버넌스 (Governance)

Livepeer 프로토콜 내 거버넌스의 역할은 다음 세 가지입니다.

1. 규칙을 위반한 노드로부터 슬래싱된 공용 자금의 **소각 또는 사용처 결정**
2. 방송자에게 가치 있는 건강하고 번성하는 네트워크 유지를 위해 **네트워크 파라미터 조정**
3. **탈중앙화 방식**으로 제안된 프로토콜 업데이트 실행

이 문서에서 언급된 `UnbondingPeriod`, `RoundLength`, `ParticipationRate`, `VerificationRate` 등 다수의 네트워크 파라미터는 조정 가능합니다. 이러한 파라미터 조정 제안은 제출될 수 있으며, 위임된 스테이크에 비례한 트랜스코더의 투표를 포함한 거버넌스 프로세스를 통해, 해당 변경의 채택 여부가 프로토콜 내에서 자동으로 결정됩니다. 상세한 거버넌스 명세는 별도 문서에 있습니다. [자세히 보기](#).

공격 (Attacks)

본 섹션은 악의적 행위자가 Livepeer 네트워크를 공격하려 할 수 있는 다양한 방식을 개괄합니다. 우리는 공격자가 자신의 경제적 이익을 기준으로 의사결정을 하는 합리적 공격자 모델을 사용합니다. 여러 공격은 수익성이 없도록 설계되어 완화되며, 설사 수익성이 없는 공격이 지속되더라도 네트워크 실패 없이 효율 저하 수준에 그치도록 보장하고자 합니다.

합의 공격 (Consensus Attacks)

앞서 언급했듯이, Livepeer 생태계의 합의는 기초 블록체인 플랫폼(예: Ethereum)이 제공합니다. 51% 공격, Livepeer 토큰의 이중지불, 네트워크 포크는 Ethereum 자체와 동일한 자원과 공격 비용이 필요합니다.

Livepeer는 스테이크 기반 프로토콜이며, 트랜스코더는 작업 검증 프로세스와 토큰 보상 분배 프로세스에 참여하지만, 다른 트랜스코더의 작업을 검증하거나 승인하는 역할은 아닙니다. 체인 자체의 개념이 없고, 이전 블록의 검증도 수행하지 않습니다. 각 참여자는 자신의 작업을 검증하고, 자신의 차례에 자신의 토큰 할당분을 분배하기 위한 경제적 인센티브만 존재합니다. 따라서 PoS 프로토콜에서 흔히 거론되는 Long Range Attack, Nothing at Stake, Bribe Attack 등은 적용되지 않습니다. 여러 블록에 서명하거나 과거 상태에서 더 긴 체인을 만들 기회가 없기 때문입니다. 다만, 기초 블록체인이 지분증명으로 전환할 경우, Livepeer에서의 이득이 Ethereum에서의 공격 비용을 초과하는 상황이 된다면 이들 공격이 위협이 될 수 있음을 인지해야 합니다.

기초 블록체인의 보안에 의존하는 것은 합의 공격을 방지하는 데 유용하지만, 그 외에도 Livepeer 네트워크의 품질과 효율성을 해칠 수 있는 공격 유형이 존재합니다.

DDoS

Livepeer에서의 서비스 거부(DoS)는 두 가지 방향으로 발생할 수 있습니다.

1. 트랜스코더가 작업을 수락한 뒤 트랜스코딩을 거부하여, 방송자가 인코딩된 스트림을 네트워크에 내보내는 것을 방해하거나 지연시키는 경우
2. 방송자가 세그먼트 전송을 거부하여, 트랜스코더가 자신에게 할당되었다고 믿는 작업을 수행하지 못하게 하는 경우

두 공격 모두 비용이 수반되며, 다소의 불편함은 있더라도 완화 가능합니다.

첫 번째 경우, 트랜스코더는 온체인에서 가용성 청구 비용을 지불해야 합니다. 일을 하지 않아 수수료를 받지 못한다면, 이는 곧 ETH를 낭비하는 것입니다. 방송자는 작업을 재제출하여 새로운 트랜스코더에게 할당받을 수 있습니다. 확장성 옵션으로, 프로토콜이 단일 트랜스코더가 아니라 우선순위 순으로 여러 유효 트랜스코더를 식별하도록 하여, 방송자가 추가 온체인 트랜잭션 없이 다음 후보로 이동할 수 있게 할 수도 있습니다. 또한 수락된 작업 수, 작업당 평균 트랜스코딩 세그먼트 수 등의 통계는 온체인 데이터로부터 계산될 수 있으며, 위임자들은 이를 참고하여 위임 대상을 선택합니다. 품질이 낮게 행동하면 역할을 잃게 됩니다.

두 번째 경우(방송자가 트랜스코더의 작업을 막는 경우)는 용량 계획(capacity planning) 문제입니다. 트랜스코더 노드는 자신이 동시 처리 가능한 작업 수, 작업이 활성화/비활성일 확률 등을 기록·관리하여, 자신이 청구한 작업을 처리할 수 있는 용량을 항상 확보하도록 할 수 있습니다. 세그먼트를 보내지 않는 노드를 무시하거나 EndJob()을 호출하는 것은 트랜스코더에 큰 피해를 주지 않습니다.

무의미하거나 자기거래(Self Dealing)형 트랜스코더 (Useless or Self Dealing Transcoder)

충분한 스테이크를 가진 트랜스코더가 이론적으로 BlockRewardCut을 100%, FeeShare를 0%로 설정하고, PricePerSegment를 매우 높게 책정하여 실제 작업 없이도 토큰 할당을 받으려 할 수 있습니다. 이는 CompetitivenessTolerance로 방지됩니다. 해당 파라미터는 일정 수준의 유효 작업 기여를 요구합니다. 또한 프로토콜 참여에 따른 트랜잭션 비용을 고려하면, 수수료를 공유하는 유효 트랜스코더에게 토큰을 스테이킹하는 편이, 수수료를 받지 못하는 무의미한 트랜스코더로 남는 것보다 수익성이 높습니다.

잘못된 출력을 내는 비정상 트랜스코더는 빠르게 슬래싱되어 스테이크가 줄어들고, 결국 작업을 유지하거나 배정받기 어려운 수준으로 감소합니다.

트랜스코더 그리핑 (Transcoder Griefing)

방송자가 트랜스코더 운영 비용을 크게 늘리고자 한다면, 비연속적인 세그먼트 번호를 전송할 수 있습니다. 트랜스코더는 연속 범위의 세그먼트에 대해서는 단일 트랜잭션으로 작업을 청구할 수 있지만, 무작위 범위에 걸친 작업을 청구하려면 여러 트랜잭션이 필요하기 때문입니다. 방어 방안은 다음과 같습니다.

1. 트랜스코더가 EndJob()을 호출하여 작업을 수행하지 않거나 수수료 청구를 시도하지 않음
2. 프로토콜 차원에서 온체인 파싱 또는 더 나은 세그먼트 청구 인코딩을 도입해, 비연속 세그먼트를 단일 호출로 청구할 때의 수수료를 절감
3. 해당 세그먼트를 단순히 무시하고 작업을 청구하지 않음

이 공격은 방송자에게 높은 비용이 듭니다. 방송자는 트랜스코더 할당을 받기 위해 예치금을 두고, 온체인으로 작업을 제출해야 하기 때문입니다. 방송자는 트랜스코더를 귀찮게 하거나 비효율을 초래할 수는 있어도, 네트워크에 실질적 피해를 주기는 어렵습니다.

체인 재조직 (Chain Reorg)

방송자가 Livepeer 스마트 계약에 작업을 제출할 때, 프로토콜은 현재 블록 해시를 사용해 해당 작업을 할 당받을 트랜스코더를 결정합니다. 기초 블록체인의 재조직(reorg)은 이 상황에서 혼란을 야기할 수 있습니다. 이는 직접적인 “공격”은 아니지만, 트랜스코더가 어느 순간에는 유효하고, 재조직 이후에는 유효하지 않게 될 수 있습니다. 리오그가 감지되면 방송자는 스트림을 새로운 유효 트랜스코더로 리디렉션할 수 있으며, 또는 프로토콜이 메인 체인에 포함된 잉클 블록(uncle)을 감지하여, 특정 임계값 내의 잉클 블록이 해당 트랜스코더를 유효하게 만들었을 경우 유효로 간주할 수도 있습니다.

라이브 비디오 배포 (Live Video Distribution)

본 백서는 적응형 비트레이트 스트리밍(Adaptive Bitrate) 및 모든 기기 도달을 위해 필요한 라이브 비디오의 올바른 트랜스코딩을 보장하는 경제적 인센티브와 프로토콜에 주로 초점을 맞추었습니다. 그러나 비디오를 네트워크 전반에 고품질·저지연으로 배포하는 것 또한 동일하게 중요합니다. 배포의 경제학은 BitTorrent로 유명해진 맞교환(tit-for-tat) 대역폭 회계에 기반하며, SWAP 등의 프로토콜로 확장됩니다 [13]. 단순화하면, 노드는 비디오 세그먼트를 요청할 때 비용을 지불하고, 제공할 때 비용을 받습니다. 한 노드가 이미 세그먼트를 보유하고 있고 다수의 요청자에게 제공할 수 있다면, 이는 수익성이 있습니다. 이러한 노드를 Relay 노드라고 부릅니다.

네트워크 내 역할에 따라 대역폭과 관련된 인센티브는 다릅니다.

- **소비자(Consumers):** 자신이 영상을 무료로 소비하는 대가로, **업스트림 대역폭**을 다른 소비자에게 콘텐츠 제공에 활용하도록 교환할 의사가 있을 수 있습니다. Webtorrent 시스템을 참고 [14].
- **방송자(Broadcasters):** **오리진 노드**의 역할을 하며, 영상 소비에 대해 비용을 청구하거나, 모두가 무료로 접근할 수 있도록 **대역폭 비용을 보조**할 수 있습니다.
- **트랜스코더와 Relay 노드:** **수익성이 있는 한** 비디오 배포를 위해 대역폭을 제공합니다. 이는 전통적 CDN의 역할과 유사합니다.

네트워크를 흐르는 데이터의 핵심 단위를 Segments로 삼으면, ETH를 결제의 기준 통화로 하는 맞교환 대역폭 회계를 구현할 수 있습니다. Swarm에서 차용한 Chequebook Contract 추상화를 사용하여, 오픈체인 결제 전달 + 온체인 정산 방식을 적용합니다 [6]. 향후 Raiden Network [15] 등의 생태계 발전은 결제 채널을 이 목적에 활용할 수 있게 할 것입니다. 또한 토큰 전송이 프로토콜에 네이티브이므로, 콘텐츠 가격을 프로토콜에 직접 내장하는 것도 가능합니다. 방송자는 시간 또는 콘텐츠에 대해 직접 요금을 청구할 수 있고, 노드들은 방송자에게 환류되는 더 높은 가격/세그먼트를 지불하는 방식으로 가치 이전에 동의할 수 있습니다.

중요한 점은, 대역폭 회계는 비디오 세그먼트를 네트워크 전반에 전달하여 용량을 늘리는 Relay 노드 운영의 수익성을 확보할 수 있지만, 이 노드들은 새로운 토큰 할당으로 인센티브를 받지 않는다는 것입니다. 실제로 Livepeer의 출력은 전통적 CDN(Amazon S3, Cloudflare 등)이나 탈중앙 CDN(IPFS, Swarm 등)에 삽입될 수 있습니다. 피어-투-피어 비디오 세그먼트 배포 프로토콜의 개발 자체는 성능 최적화와 개선을 위한 지속적 기회를 제공합니다.

피어-투-피어 CDN은 오리진 CDN 서버의 대역폭 요구를 80-98%까지 줄이는 것으로 나타났으며 [17], 탈중앙 네트워크의 토큰 메커니즘은 현재의 독점적 P2P CDN의 오픈 버전 개발·유지에 이해관계자들의 정렬을 가능하게 합니다. PPSP 프로토콜 [18]은 라이브 콘텐츠 전송에 초점을 둔 오픈 구현의 유력한 후보입니다.

Livepeer의 암호경제학과 직접적 연관이 없는 영역이므로 이 문서에서는 세부를 생략하지만, 관심 있는 독자는 [여기](#)에서 개발 진행을 따라가고, 향후 비디오 배포 프로토콜만을 다루는 별도 문서를 참조할 수 있습니다.

활용 사례 (Use Cases)

Livepeer 프로젝트는 일대다 라이브 비디오 방송(multicast)의 탈중앙화에 중점을 둡니다. 이는 방송자가 중간 왜곡이나 사후 해석 없이, 청중과 직접 연결할 수 있도록 하는 가장 순수한 형태의 미디어 배포입니다. 모든 이에게 목소리를 낼 플랫폼을 제공합니다. 기존 중앙화 솔루션은 검열, 제3자의 사용자 데이터/관계/수익화 통제, 비효율적 비용 구조 등의 문제를 겪을 수 있습니다. 다음은 Livepeer 위에서 구축될 수 있는 논리적 응용 사례들입니다.

사용량 기반 콘텐츠 소비 (Pay-As-You-Go Content Consumption)

가치 이전 트랜잭션이 프로토콜에 내장되어 있으므로, 방송자는 신용카드, 계정, 중앙화된 플랫폼의 신원 통제 없이도 시청자에게 라이브 방송 소비에 대한 비용을 직접 청구할 수 있습니다. 이는 교육(온라인 강의 유료 수강), 이벤트(콘서트·스포츠 경기 유료 시청), 엔터테인먼트(게이머·공연자 라이브 스트림 유료 시청) 등 다양한 분야에 적용 가능하며, 시청자의 프라이버시를 보존하면서 소비한 만큼만 방송자에게 직접 지불할 수 있습니다.

자동 확장형 소셜 비디오 서비스 (Auto-scaling Social Video Services)

오늘날 소비자 비디오 서비스를 구축하는 데 있어 가장 큰 과제 중 하나는, 증가하는 스트림 수와 시청자 수요를 지원하기 위한 인프라 확장입니다. Livepeer 네트워크 위에서 손쉽게 비디오 솔루션을 구축하게 해 주고, 스트림·시청자 수에 따라 자동으로 확장되는 서비스 레이어는, 원래라면 서버 증설, 미디어 서버 라이선스, 스파이크 대응 자원 관리 등을 계속해야 하는 인프라 개발자들에게 환영받는 해결책이 될 것입니다.

검열 불가능한 라이브 저널리즘 (Uncensorable Live Journalism)

Twitter, Facebook 등 기존 플랫폼은 대규모 청중에게 도달할 수 있는 훌륭한 라이브 비디오 솔루션을 제공하지만, 정치적 갈등 상황에서 가장 먼저 차단·검열될 수 있습니다. Livepeer와 같은 탈중앙 네트워크를 사용하면, 현장에서 실시간으로 일어나는 사실이 알려지는 것을 막기가 거의 불가능해집니다.

비디오 기반 DApp (Video Enabled DApps)

Ethereum 생태계를 중심으로 탈중앙 앱(DApp)이 등장하고 있지만, 현재로서는 중앙화 솔루션 없이 DApp 내에 라이브 비디오를 임베드하기가 쉽지 않거나, WebRTC 제약으로 수용 가능한 클라이언트 수가 제한됩니다. 스택에 Livepeer를 도입하면, 애플리케이션은 완전 탈중앙을 유지하면서도, 규모에 맞게 라이브 비디오를 원하는 만큼의 사용자에게 제공할 수 있습니다.

요약 (Summary)

요약하면, Livepeer 프로토콜은 트랜스코딩 및 라이브 비디오 배포를 위해 노드가 자신의 연산력과 대역폭을 네트워크에 기여하도록 인센티브를 제공합니다. 작업 검증은 Truebit 프로토콜 상의 확장 가능한 메커니즘으로 해결되며, 이를 통해 노드가 올바르게 트랜스코딩해야 수수료와 토큰 할당을 획득하고, 트랜스코더로서의 가치 창출 역할을 유지할 수 있습니다. 네트워크의 게이미피케이션과 무의미한 작업 문제는 위임 지분 증명(Delegated Proof of Stake) 기반 블록 보상 회계의 경제학으로 해결됩니다. 실제 수요가 없는 작업을 수행하면서 수수료를 지불하여 다른 위임자에게 분배되도록 만드는 것보다, 가치를 창출하는 노드에 토큰을 스테이크하는 편이 더 경제적으로 합리적입니다.

그 결과, Livepeer는 웹3 스택에서 누락된 레이어를 메우는, 확장 가능하고 사용량 기반(pay-as-you-go)의 탈중앙 라이브 비디오 방송 네트워크를 제공합니다.

부록 (Appendix)

Livepeer 프로토콜 파라미터 참조 (Livepeer Protocol Parameter Reference)

Parameter Name	Description	Example Value
T	Segment length in seconds	2 seconds
N	Number of active transcoders	144
RoundLength	Length of time between election of a new round of transcoders	1 day
InflationRate	The current target inflation rate per round of LPT. (Moves algorithmically).	.04% (equivalent to 15%/year)
ParticipationRate	The target percent of token bonded vs liquid.	50%
RoundLockAmount	Transcoders rates lock in for this percentage of a round at the end of a round so that delegators can review and delegate accordingly without worrying about last minute rate changes.	10% == 2.4 hours

UnbondingPeriod	Time between entering unbonding state, and ability to withdraw the funds.	1 month
VerificationPeriod	The deadline for verifying a job claim after submission of the job claim. This also serves as the minimum period that a receipt of data persistence must be provided in the decentralized storage solution.	6 hours
VerificationRate	The % of segments that will be verified.	1/500
FailedVerificationSlashAmount	% to slash in the case of a failed verification (beyond the potential allowed failure threshold)	5%
MissedRewardSlashAmount	% to slash in the case of missing a block reward round (Maybe only do this in the case of n consecutive misses)	3%
MissedVerificationSlashAmount	% to slash in the case the transcoder didn't call verification	10%
CompetitivenessTolerance	If all transcoders were always available and set the same price and fees, they would receive work in proportion to their stake. This parameter sets a % that they have to be within this target work % to be eligible for token allocation. This prevents transcoders from doing very little share of work relative to their stake.	90% (extreme example. With 100 transcoders and 100,000 segments, this means I am ok if I only did 100 segments (10% of the 1000 I was supposed to do)).
*SlashingThresholds (TBD)	Placeholder to indicate that we may not slash on all failures, only if they exceed some threshold % of failure rate.	
VerificationFailureThreshold	% of verifications you can fail without being slashed. Useful because of external dependencies like Swarm/Truebit that could cause sporadic failure.	1%
FinderFee	% of slash amount that the finder will receive as compensation.	5%
SlashingPeriod	The deadline for invoking a slashing condition after the VerificationPeriod has	1 hour

completed.

Livepeer 프로토콜 트랜잭션 유형 (Livepeer Protocol Transaction Types)

Transaction	Description
<code>Bond()</code>	Bond stake towards a transcoder.
<code>Unbond()</code>	Enter the unbonding state for the fixed <code>UnbondingPeriod</code> .
<code>Transcoder()</code>	Declare your intentions as a transcoder.
<code>ResignAsTranscoder()</code>	Resign your intentions as a transcoder.
<code>TranscodeAvailability()</code>	This transcoder is currently open to accepting another job. They're in the pool to be assigned randomly on new job submissions.
<code>Job()</code>	Submit a transcoding job on chain.
<code>EndJob()</code>	End the job to relinquish transcoding responsibility.
<code>Deposit()</code>	Submit a deposit on chain that will be used and drawn against to pay for jobs.
<code>Withdraw()</code>	Withdraw from deposit and unbonded stake.
<code>ClaimWork()</code>	End the transcode job and make the claim of which segments you can prove you've transcoded via segment range and merkle root.
<code>DistributeFees()</code>	Transcoder claims the fees for a particular claim after verification.
<code>Reward()</code>	Does all the verifications on chain to either slash or distribute token allocations. Can only be invoked by a transcoder who is active in the current round, once per round.
<code>Verify()</code>	Transcoder provides the transcode claims for segments which will be verified along with merkle proofs for comparison with merkle root from <code>ClaimWork()</code> . Explicitly call <code>Truebit</code> to

perform verification.

<code>InitializeRound()</code>	This transaction needs to be invoked once after the new round's start block to initialize the new active transcoder pool.
<code>UpdateDelegatorStake()</code>	This allows a delegator to claim their fees + token allocation from previous rounds. It's invoked automatically through unbonding and bonding, but it serves as a failsafe in case the delegator would like to update without changing state.
<code>*GovernanceTransactions()</code>	TBD

참고문헌 (References)

1. Ethereum White Paper - Vitalik Buterin - Ethereum Wiki - <https://github.com/ethereum/wiki/wiki/White-Paper>
2. Fat Protocols - Joel Monegro - USV Blog - <http://www.usv.com/blog/fat-protocols>
3. Crypto Tokens and the Coming Age of Protocol Innovation - Albert Wenger - <http://continuations.com/post/148098927445/crypto-tokens-and-the-coming-age-of-protocol>
4. The Case For SectorCoins - Eric Tang - <https://medium.com/@ericxtang/case-for-sectorcoins-b70a7c820c2d#.7892n4a57>
5. Delegated Proof-of-Stake Consensus - Daniel Larimer - <https://bitshares.org/technology/delegated-proof-of-stake-consensus/>
6. Truebit - Jason Teutsch and Christian Reitweisner - <https://people.cs.uchicago.edu/~teutsch/papers/truebit.pdf>
7. swap, swear and swindle, incentive system for swarm - viktor trón, aron fischer, daniel a. nagy, zsolt felföldi, nick johnson - <https://ethersphere.github.io/swarm-home/ethersphere/orange-papers/1/sw%5E3.pdf>
8. Kademlia: A Peer-to-peer Information System Based On The XOR Metric - Petar Maymounkov and David Mazieres <https://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf>
9. Steem Whitepaper - Daniel Larimer, Ned Scott, Valentine Zavgorodnev, Benjamin Johnson, James Calfee, Michael Vandeberg - <https://steem.io/SteemWhitePaper.pdf>
10. Introducing Casper "the Friendly Ghost" - Vlad Zamfir - <https://blog.ethereum.org/2015/08/01/introducing-casper-friendly-ghost/>
11. Tendermint Docs - Jae Kwon and Ethan Buchman - <https://tendermint.com/docs>
12. Swarm - <http://swarm-gateways.net/bzz:/theswarm.eth/>

13. Incentives Build Robustness in BitTorrent – Bram Cohen – <http://bittorrent.org/bittorrentecon.pdf>
14. WebTorrent – <https://webtorrent.io/>
15. Raiden Network – <http://raiden.network/>
16. ERC20 Token Standard – <https://github.com/ethereum/EIPs/issues/20>
17. Peer5 leverages viewers' devices for a P2P approach to streaming video – <http://techcrunch.com/2017/01/26/peer5-y-combinator/>
18. Peer-to-Peer Streaming Peer Protocol – <https://tools.ietf.org/html/rfc7574>
19. Inflation and Participation in Stake Based Protocols – Doug Petkanics – <https://medium.com/@petkanics/inflation-and-participation-in-stake-based-token-protocols-1593688612bf>