Data Availability in Blockchain

Wonjae Choi
Email | Github | X | LinkedIn

Table of Contents

1. Data Availability

- Understanding Data Availability
- Data Availability Problem

2. Data Availability Sampling

- Construction Framework

3. References

Understanding Data Availability

1. Understanding Data Availability

Definition

What is Data Availability?

Data Availability in General

- Ensuring that data is always
  - Accessible
  - Accurate, consistent, and up-to-date
- Can be easily achieved in centralized systems

Data Availability in Blockchain

- All data necessary for the block verification
  - is accessible at anytime
  - for every participants in the network
- Not containing the data accuracy or integrity
- Ensures that anyone can independently verify the blockchain
- **Responsible for the Security of Blockchain**

Common Misconception

Data Availability != Data Retrievability

Data Retrievability (or Data Storage)

- The ability for historical data to be retrieved
- Usage
  - Searching past transaction data
  - Syncing nodes
- Archive Node, Decentralized Storage, The Graph
- 1-of-N honesty assumption
- Not crucial for the security or execution of the blockchain

Difference With Data Availability

- Historical data vs Data for block verification
- DA doesn't require holding all data
- Pruning is possible with DA, but not with DR

Source: [codixon.eth on X]

1. Understanding Data Availability

Background

How Blockchain is updated in Current Ethereum

Block Proposal

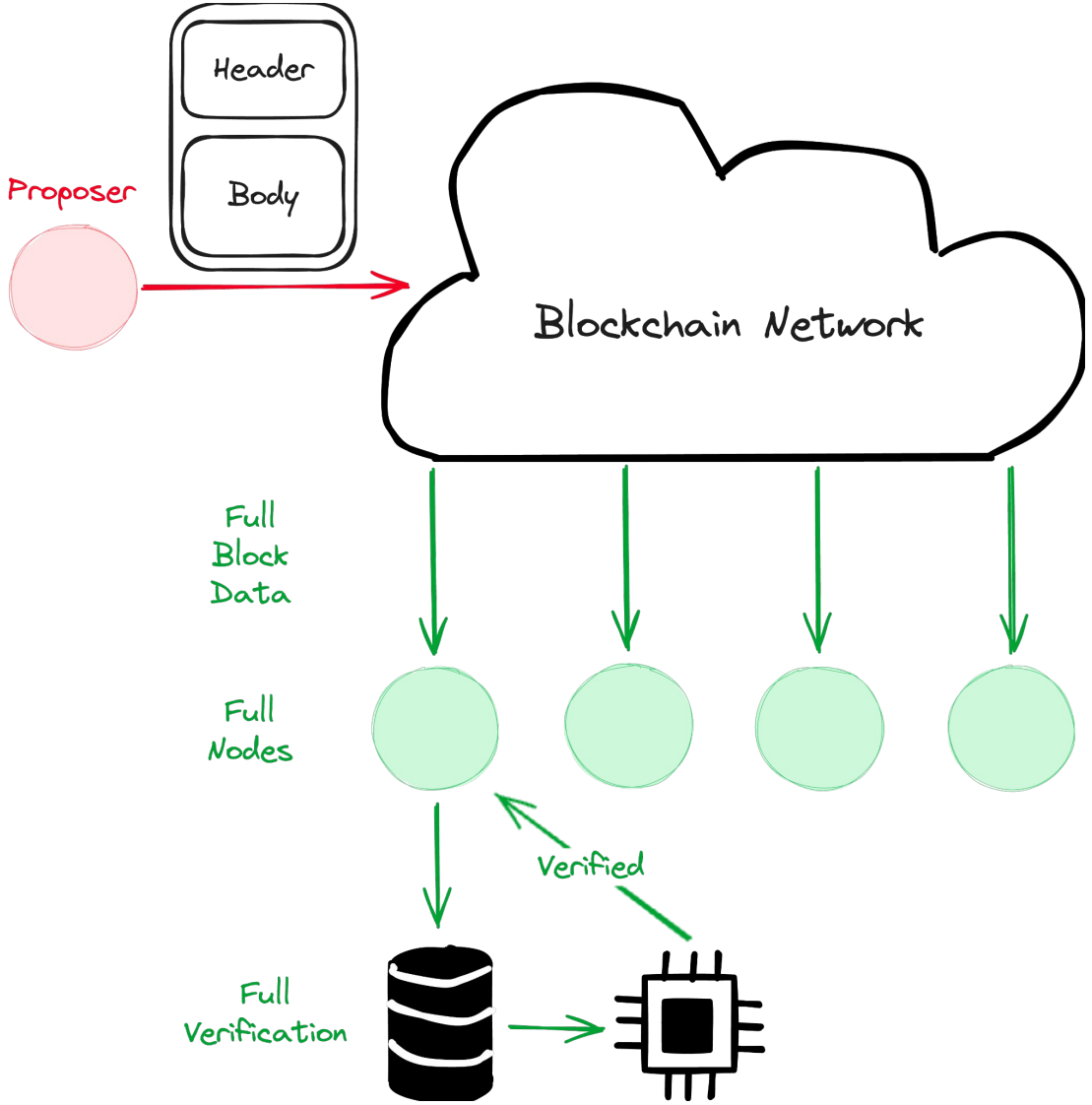
- Beacon chain selects a validator to be a block producer
- Block producer chooses transactions from the mempool and creates a block

Block Propagation

- Block producer propagate the block to the network

Block Verification & Update

- Each node executes the transactions independently and verifies the block
- If correct, each node accepts it and updates the blockchain state independently



1. Understanding Data Availability

Background

How Blockchain is updated in Current Ethereum

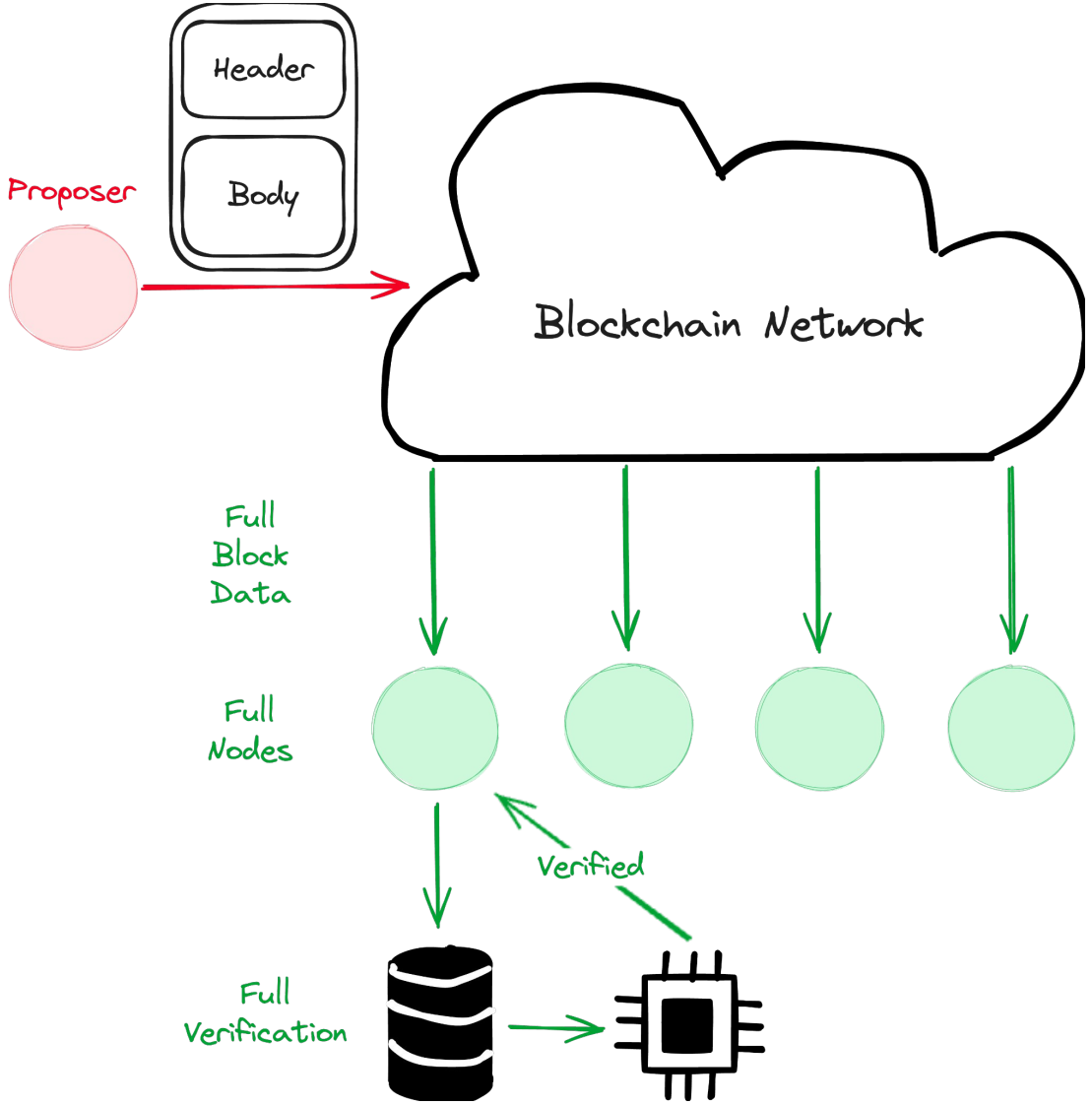
On-chain Data Availability

- Protocol enforces on-chain DA
- If they hide data, the block is rejected and get slashed

It looks fine so far …

- No problem regarding DA within the Ethereum protocol itself
- Then what is the Data Availability Problem?

Problem rises with Scaling Solutions



Data Availability Problem

1. Data Availability Problem

Problem

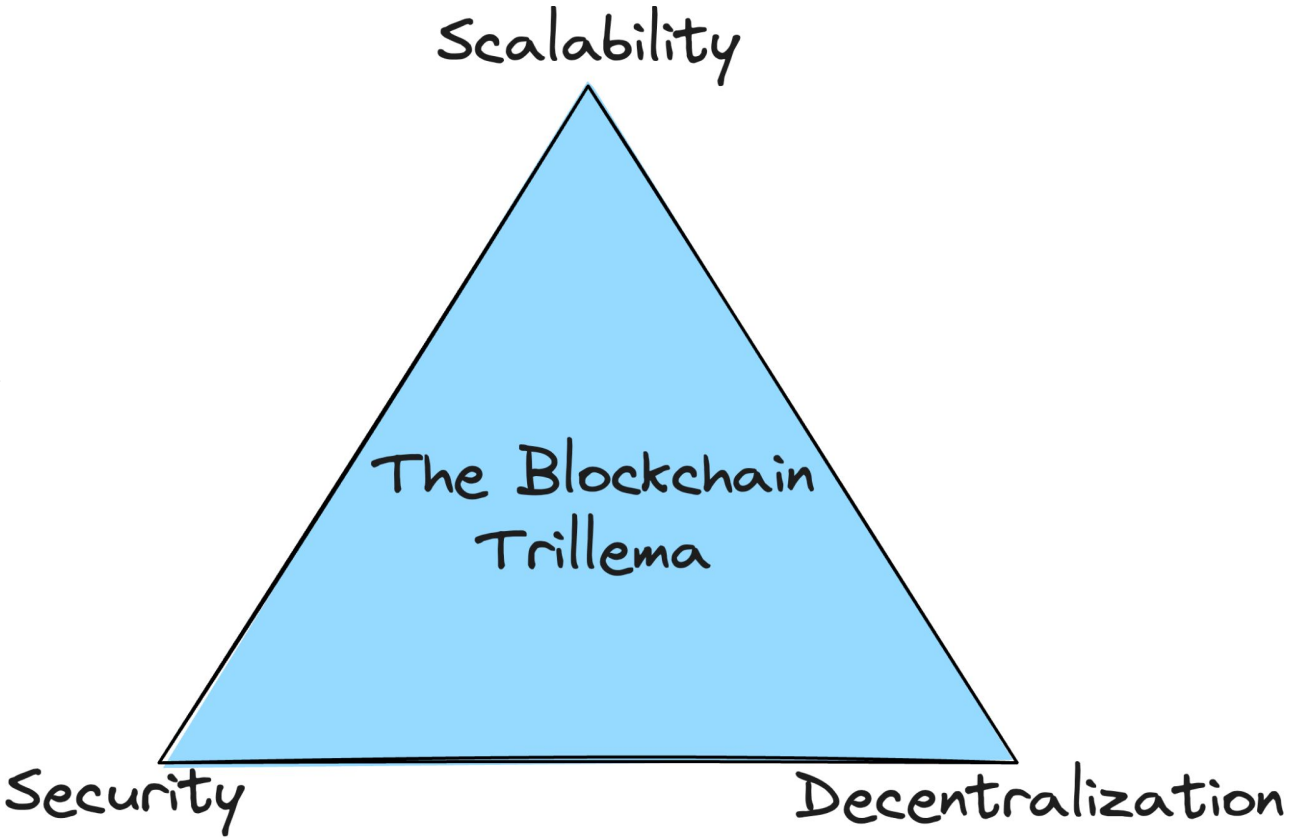
Scalability Dilemma

Scaling Solutions

- There are many solutions to address the scalability problem in Ethereum
- Notably, Light Client and Rollup (*Light Client is not yet implemented in Ethereum)

Scalability vs Security Trade-off

- Light Client and Rollup significantly enhances the scalability
- But both face security issues
- Let's see what are the security issues

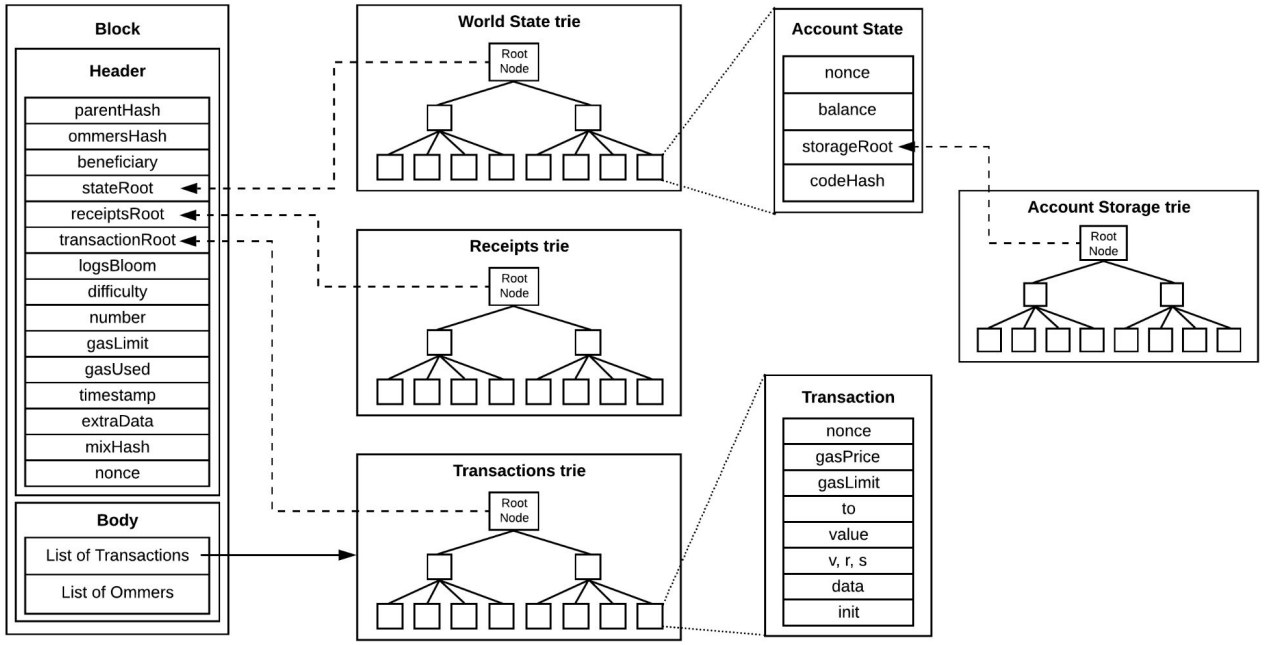


Light Node

What is Light Node?

Ethereum's Block Structure

- Block = Header + Body
- Block Header contains the Merkle Root of the transactions



Definition

- Light Node is a node that uses light client
- Only download the block headers and use them to verify the blocks
- Neither download the data nor execute the transactions
- Same as Light Node or SPV(Simple Payment Verification) Node in Bitcoin
- **Scalability & Decentralization**

Source: [Merkle Tree and Ethereum Objects - Ethereum Yellow Paper Walkthrough (2/7)]

Light Node

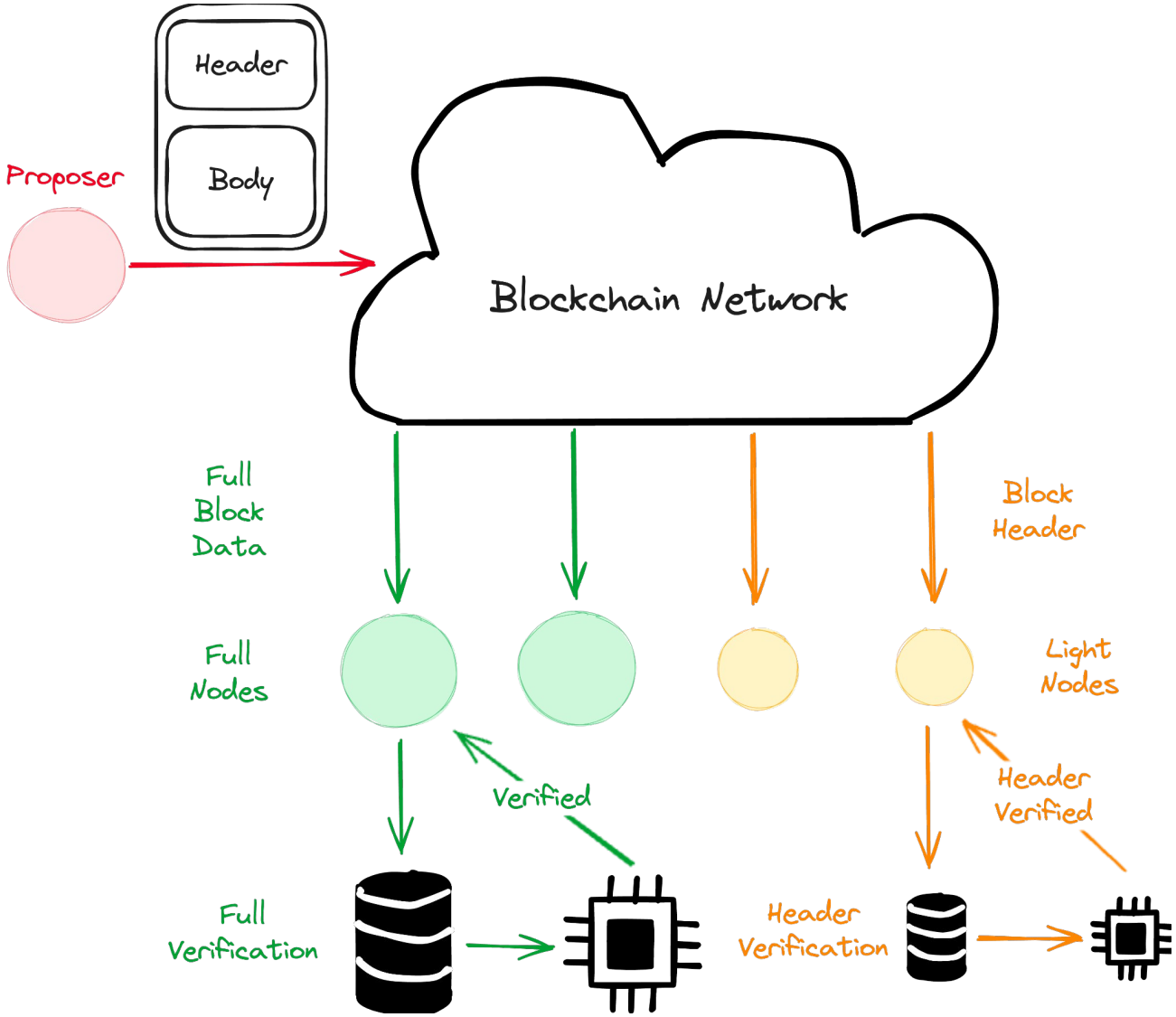
What is Light Node?

Ethereum's Block Structure

- Block = Header + Body
- Block Header contains the Merkle Root of the transactions

Definition

- Light Node is a node that uses light client
- Only download the block headers and use them to verify the blocks
- Neither download the data nor execute the transactions
- Same as Light Node or SPV(Simple Payment Verification) Node in Bitcoin
- **Scalability & Decentralization**



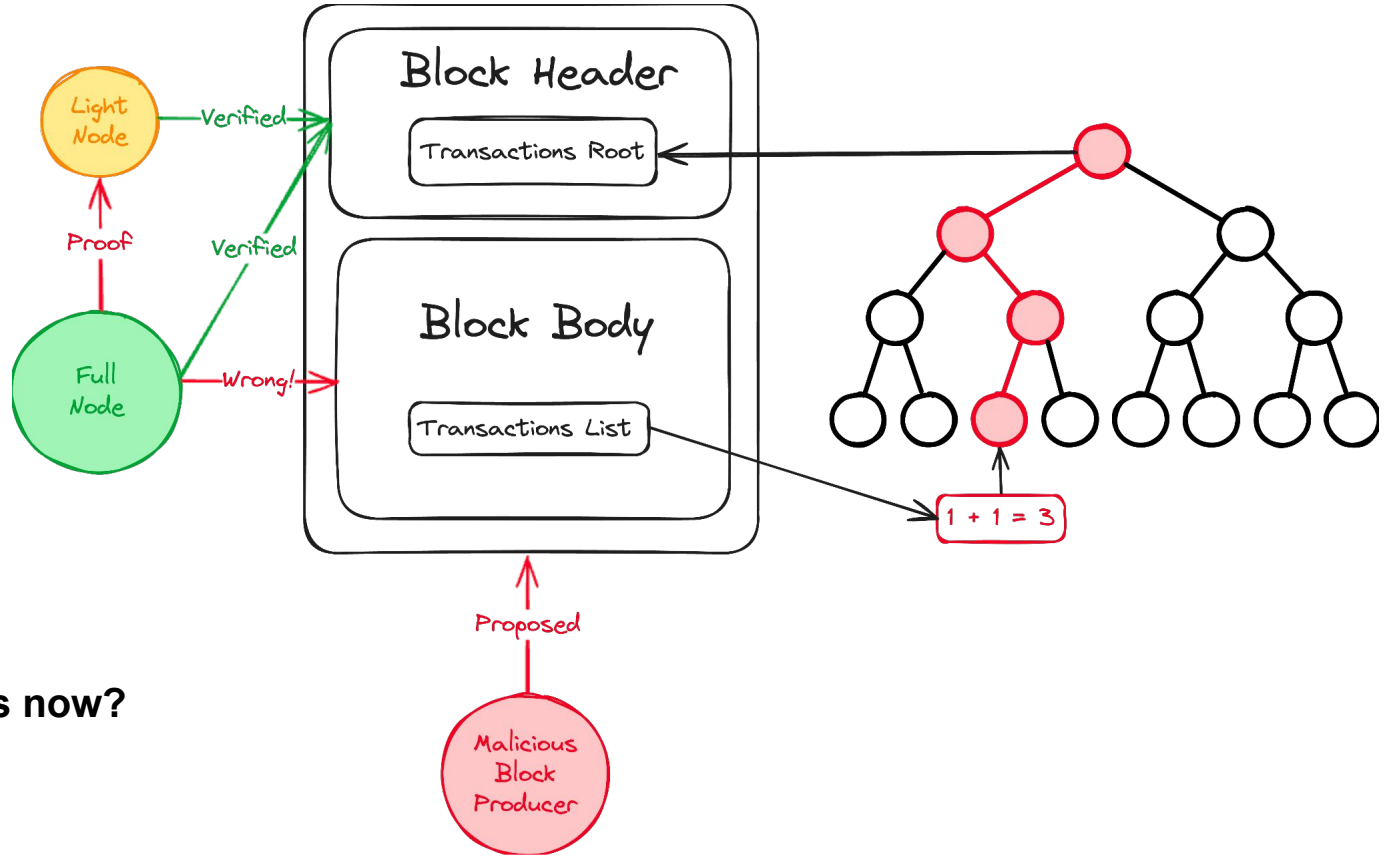
1. Data Availability Problem

Light Node

Attacking the Light Node

Block Header with Invalid Transactions

- Block headers alone cannot verify if they were created from valid transactions
  - Because what they contain are the merkle roots!
- Two approaches to fix this:
  - **Fraud Proof:** If a header is incorrect, full nodes inform light nodes (Celestia)
  - **Validity Proof:** Provide cryptographic proofs that block headers were derived from valid data (Ethereum 2.0, Avail)



So can light nodes can ensure the validity of the headers now?

> No!

Translations WIP

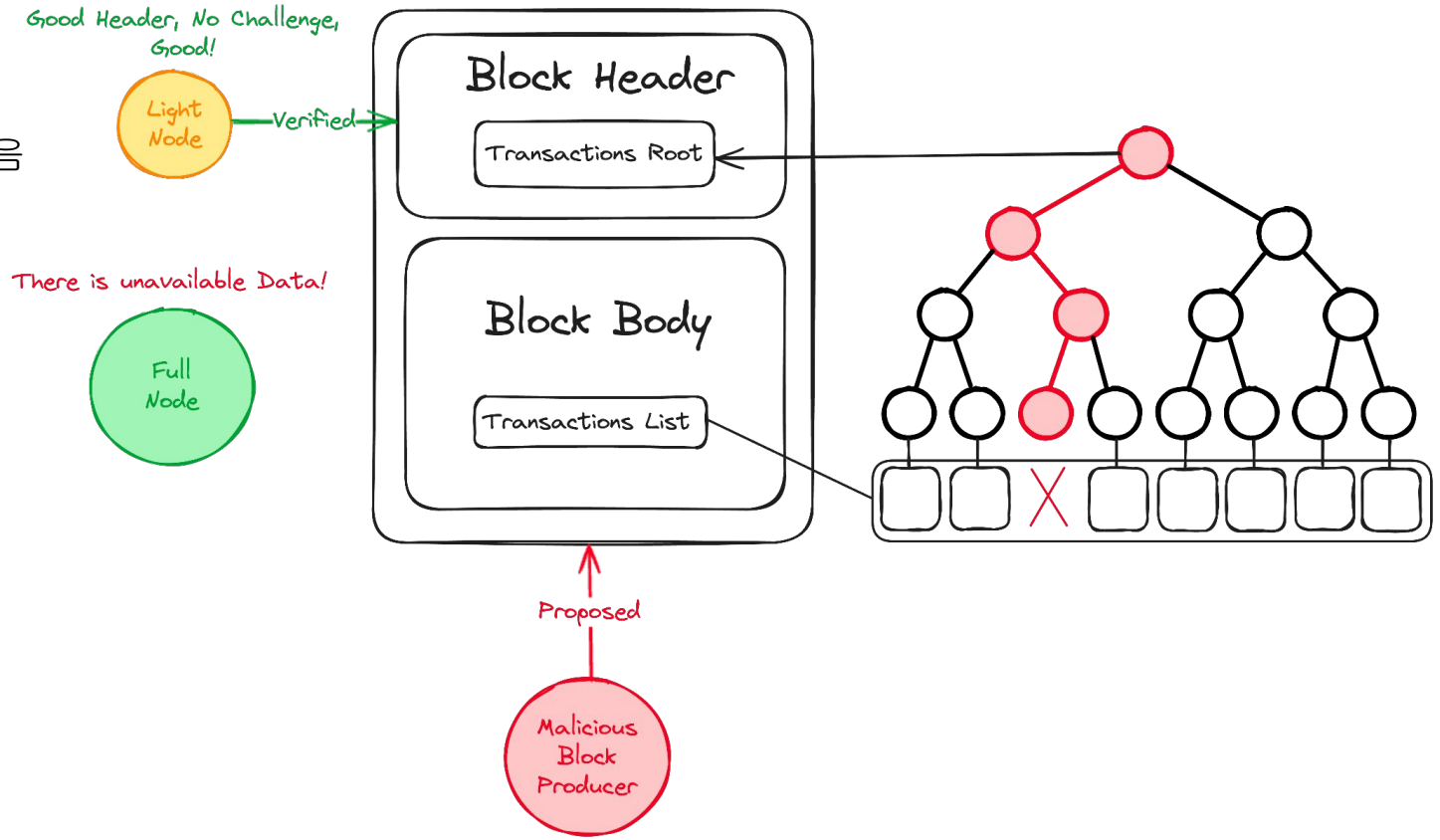
1. Data Availability Problem

Light Node

Data Availability Problem

Data Withholding Attack

- 블록 생성자가 새로 만들어진 블록에 포함된 일부 트랜잭션 데이터를 감춘다면?
- 즉, Data Availability를 제공하지 않는다면?
- 풀 노드는 이를 알지만, 라이트 노드는 이를 알 수 없음
- **Fraud Proof**
  - 틀렸음에 대한 증명을 해야 하는데, 데이터가 없으므로 증명을 만들 수 없음
  - 즉, DA가 없음을 증명하는 것은 불가능
  - 유효하지 않은 트랜잭션이 감추어 졌을 수 있음
- **Validity Proof**
  - 트랜잭션들이 전부 올바르고 블록 헤더가 올바르다는 것은 Validity Proof로 증명됨
  - 마찬가지로 DA가 없음을 증명할 수 없음
  - 유효하지 않은 트랜잭션을 감출 수는 없음



Light Node

Data Availability Problem

Data Withholding Attack

- 블록 생성자가 새로 만들어진 블록에 포함된 일부 트랜잭션 데이터를 감춘다면?
- 즉, Data Availability를 제공하지 않는다면?
- 풀 노드는 이를 알지만, 라이트 노드는 이를 알 수 없음
- **Fraud Proof**
  - 틀렸음에 대한 증명을 해야 하는데, 데이터가 없으므로 증명을 만들 수 없음
  - 즉, DA가 없음을 증명하는 것은 불가능
  - 유효하지 않은 트랜잭션이 감추어 졌을 수 있음
- **Validity Proof**
  - 트랜잭션들이 전부 올바르고 블록 헤더가 올바르다는 것은 Validity Proof로 증명됨
  - 마찬가지로 DA가 없음을 증명할 수 없음
  - 유효하지 않은 트랜잭션을 감출 수는 없음

Solutions

Solutions to Data Availability Problem in Light Node

Naive Solution - Download All the Data

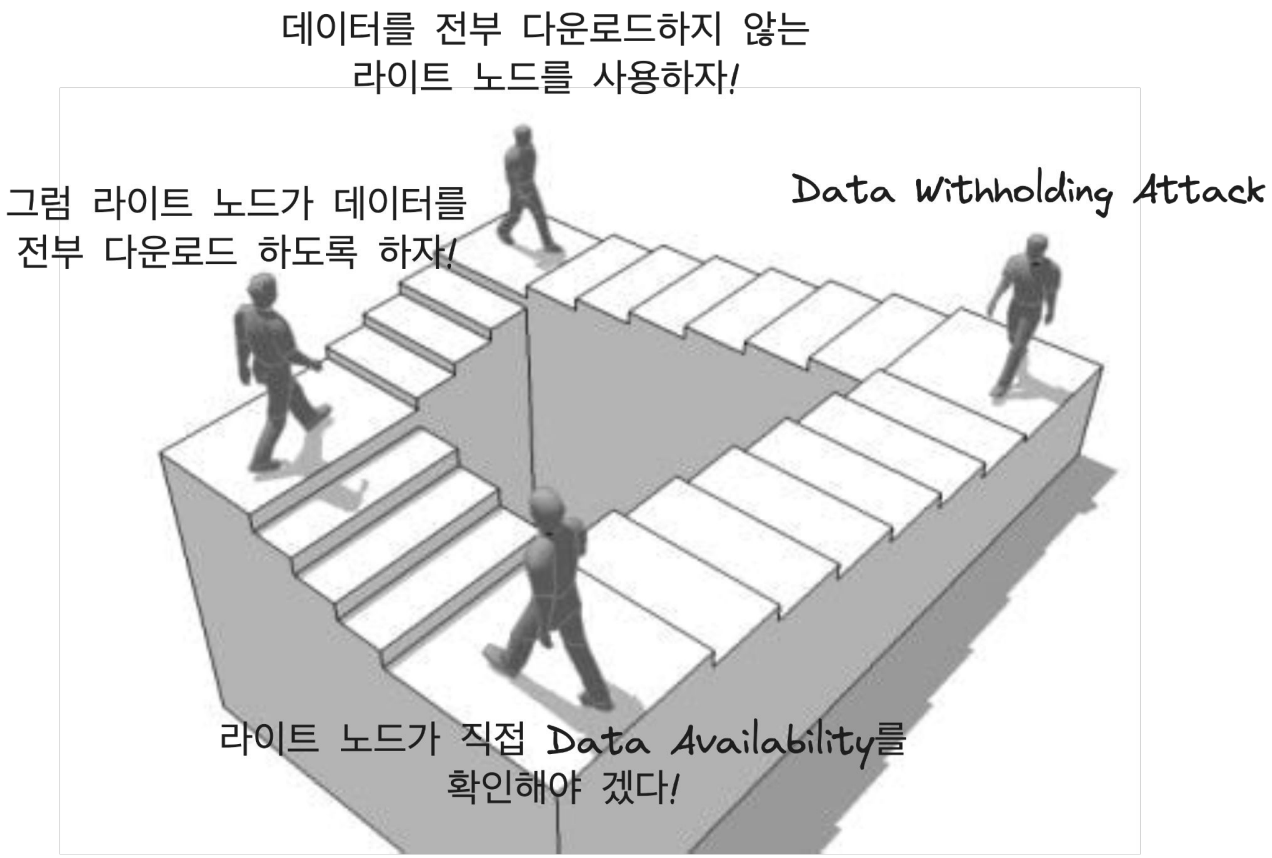
- 데이터가 실제로 전부 네트워크에 존재함을 알기 위해 모든 데이터를 다운로드
- 그러면 라이트 노드의 의미가 크게 하락하기 때문에 좋지 않은 해결책

The Best Solution - Data Availability Sampling

- 모든 데이터를 다운로드하지 않고도 데이터가 네트워크 상에 완전히 게시되어 있음을 확인하는 방법
- 정보이론, 확률론, 암호학에 기반

Another Alternatives - Data Availability Committee

- 중앙화된 혹은 허가된 노드들이 Data Availability를 제공
- Honest Majority Assumption
- 낮은 보안, 낮은 탈중앙성, 싼 가격



1. Data Availability Problem

Rollup

Why does Data Availability Matter in Rollups?

Data Withholding Attack of Sequencer

- 롤업에서는 시퀀서가 블록을 생성
- 롤업에서 DA를 제공하는 여부나 방식은 온전히 프로젝트의 선택
- 이더리움에 게시하여 온체인 DA를 제공할 수도 있고, 다른 프로토콜을 사용하여 오프체인 DA를 제공할 수도 있음
- 만약 L2 시퀀서가 악의적으로 행동하여, DA를 제공하지 않으면?

Case of Optimistic Rollup

- 옵티미스틱 롤업에서는 블록이 L1에 제출되고, Fraud Proof를 제출할 수 있는 Challenge 기간이 열림
- 하지만 Data Availability가 없으므로, Fraud Proof를 제출할 수 없음
- Challenge 기간이 지나면 유효하지 않은 트랜잭션들도

Case of ZK Rollup

- Validity Proof는 트랜잭션들의 유효성을 증명하기 때문에, 유효하지 않은 트랜잭션은 포함 불가능
- 즉, Validity Proof 자체는 Data Availability가 필요 없음
- 그럼에도 시퀀서가 Data Availability를 제공하지 않는 경우 문제가 발생
  - 블록체인의 state를 모르기 때문에 사용자가 롤업 블록체인과 상호작용할 수 없음
  - Prover가 탈중앙화 되었을 때는 Validity Proof 생성 과정에서도 Data Availability가 필요

1. Data Availability Problem

Rollup

Why does Data Availability Matter in Rollups?

Data Withholding Attack of Sequencer

- 롤업에서는 시퀀서가 블록을 생성
- 롤업에서 DA를 제공하는 여부나 방식은 온전히 프로젝트의 선택
- 이더리움에 게시하여 온체인 DA를 제공할 수도 있고, 다른 프로토콜을 사용하여 오프체인 DA를 제공할 수도 있음
- 만약 L2 시퀀서가 악의적으로 행동하여, DA를 제공하지 않으면?

Case of Optimistic Rollup

- 옵티미스틱 롤업에서는 블록이 L1에 제출되고, Fraud Proof를 제출할 수 있는 Challenge 기간이 열림
- 하지만 Data Availability가 없으므로, Fraud Proof를 제출할 수 없음
- Challenge 기간이 지나면 유효하지 않은 트랜잭션들도

Case of ZK Rollup

- Validity Proof는 트랜잭션들의 유효성을 증명하기 때문에, 유효하지 않은 트랜잭션은 포함 불가능
- 즉, Validity Proof 자체는 Data Availability가 필요 없음
- 그럼에도 시퀀서가 Data Availability를 제공하지 않는 경우 문제가 발생
  - 블록체인의 state를 모르기 때문에 사용자가 롤업 블록체인과 상호작용할 수 없음
  - Prover가 탈중앙화 되었을 때는 Validity Proof 생성 과정에서도 Data Availability가 필요

1. Data Availability Problem

Rollup

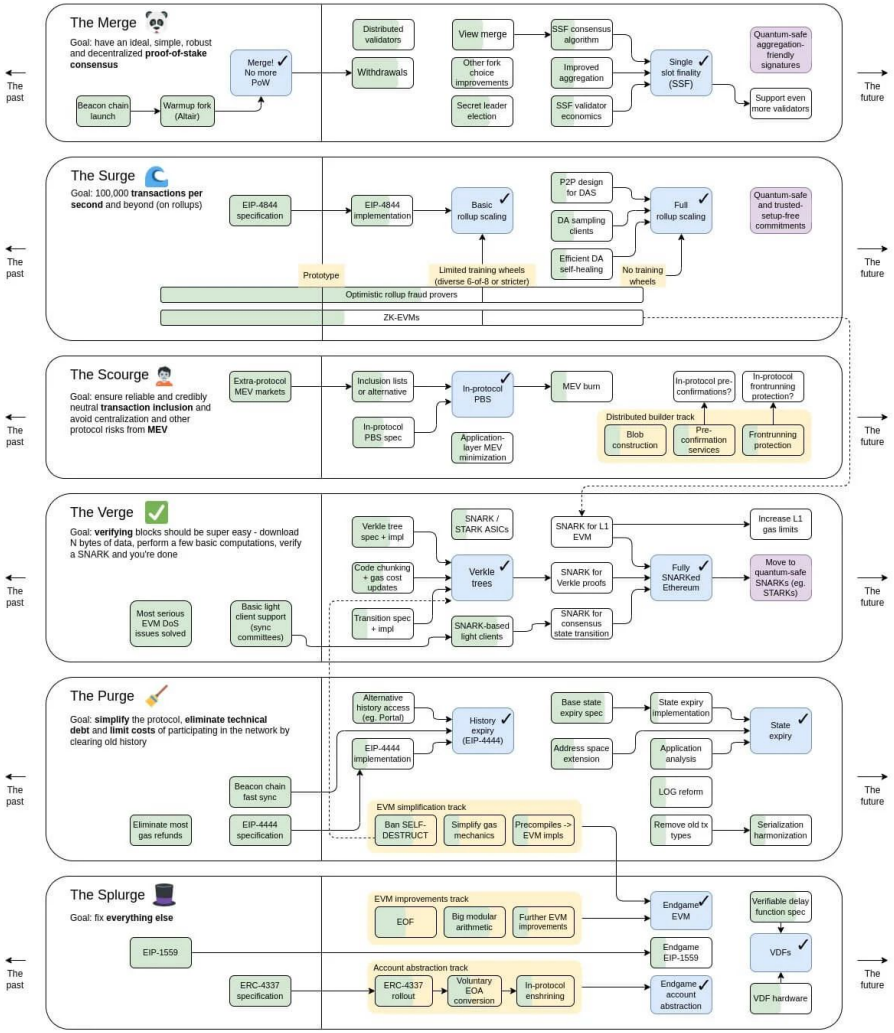
Solutions

On-chain Data Availability

- DA를 보장하기 위해 많은 롤업들은 블록 데이터를 이더리움에 게시함
- 이전에는 Calldata, EIP-4844 이후에는 Blob
- 이더리움 2.0 로드맵에서는 이더리움 프로토콜 내에서 DAS를 통해 DA 제공

Forced Inclusion

- 만약 DA가 없는 경우, L1에서 L2에 트랜잭션을 강제로 포함시키는 방법이 존재

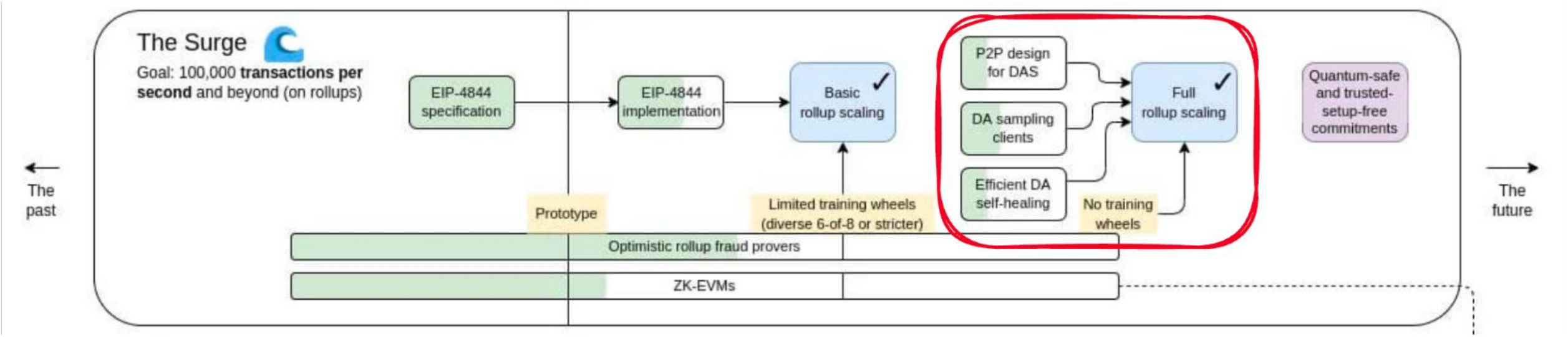


Source: [vitalik.eth on X]

1. Data Availability Problem

Rollup

Solutions



1. Data Availability Problem

Rollup

Solutions

Use Data Availability Layers

- 오프체인 프로토콜을 사용하여 DA를 보장할 수 있음
- DA Layer는 DA를 보장하기 위해 만들어진 블록체인
  - DA를 보장하기 위해 DAS나 DAC를 사용

Validium

- Validity Proof를 사용하고 DA를 오프체인에서 제공하면 Validium
- Validity Proof 자체는 DA가 필요 없으므로 보안 상의 큰 허점은 없음

Optimium (Plasma)

- Fraud Proof를 사용하고 DA를 오프체인에서 제공하면 Optimium
- Fraud Proof에는 DA가 필수적이므로, 보안 상 문제 있음

	Validity Proofs	Fraud Proofs
DA on-chain	ZK Rollup	Optimistic Rollup
DA off-chain	Validium	Plasma(Optimium)

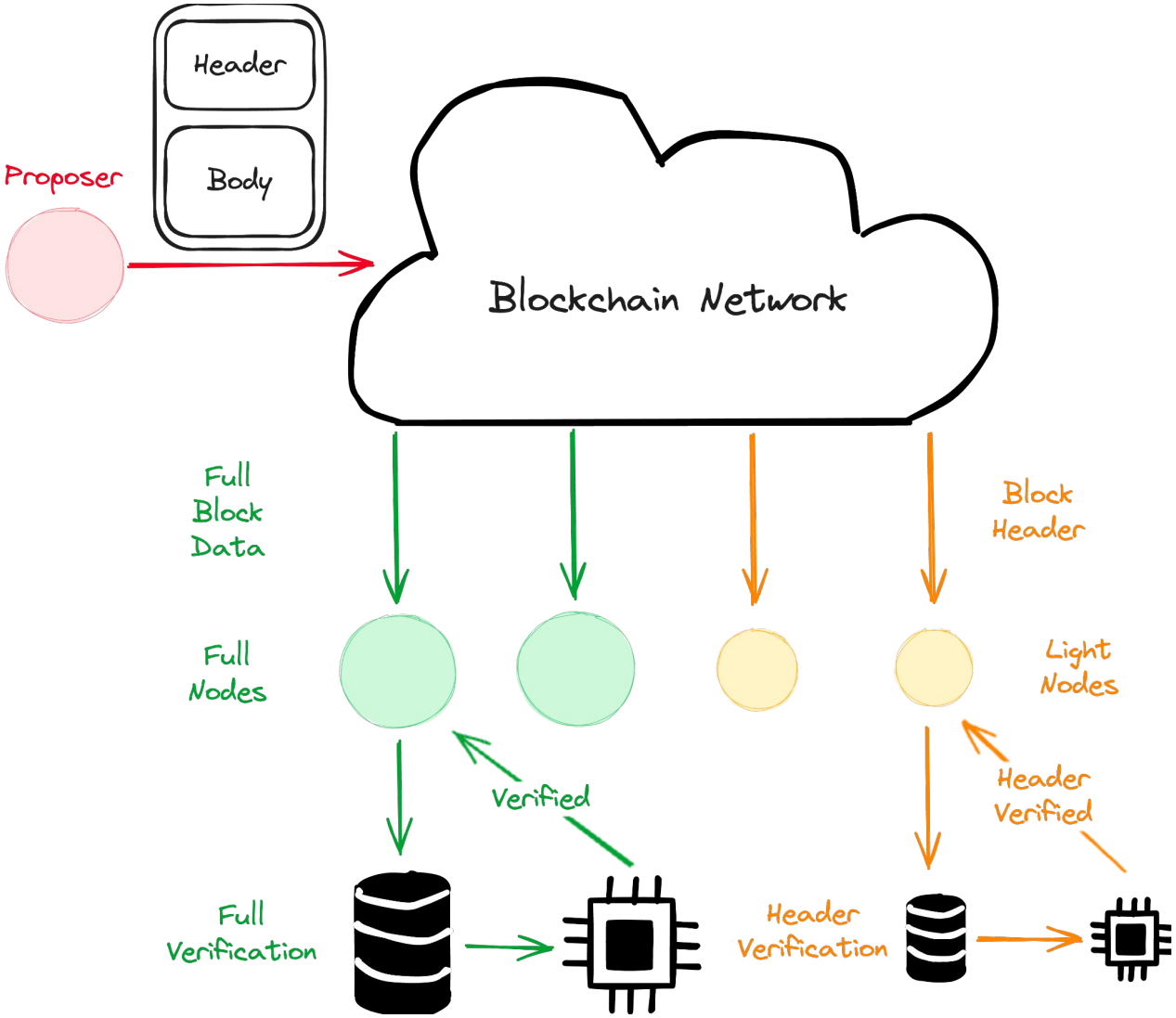
Data Availability Sampling

Settings

Example Setup

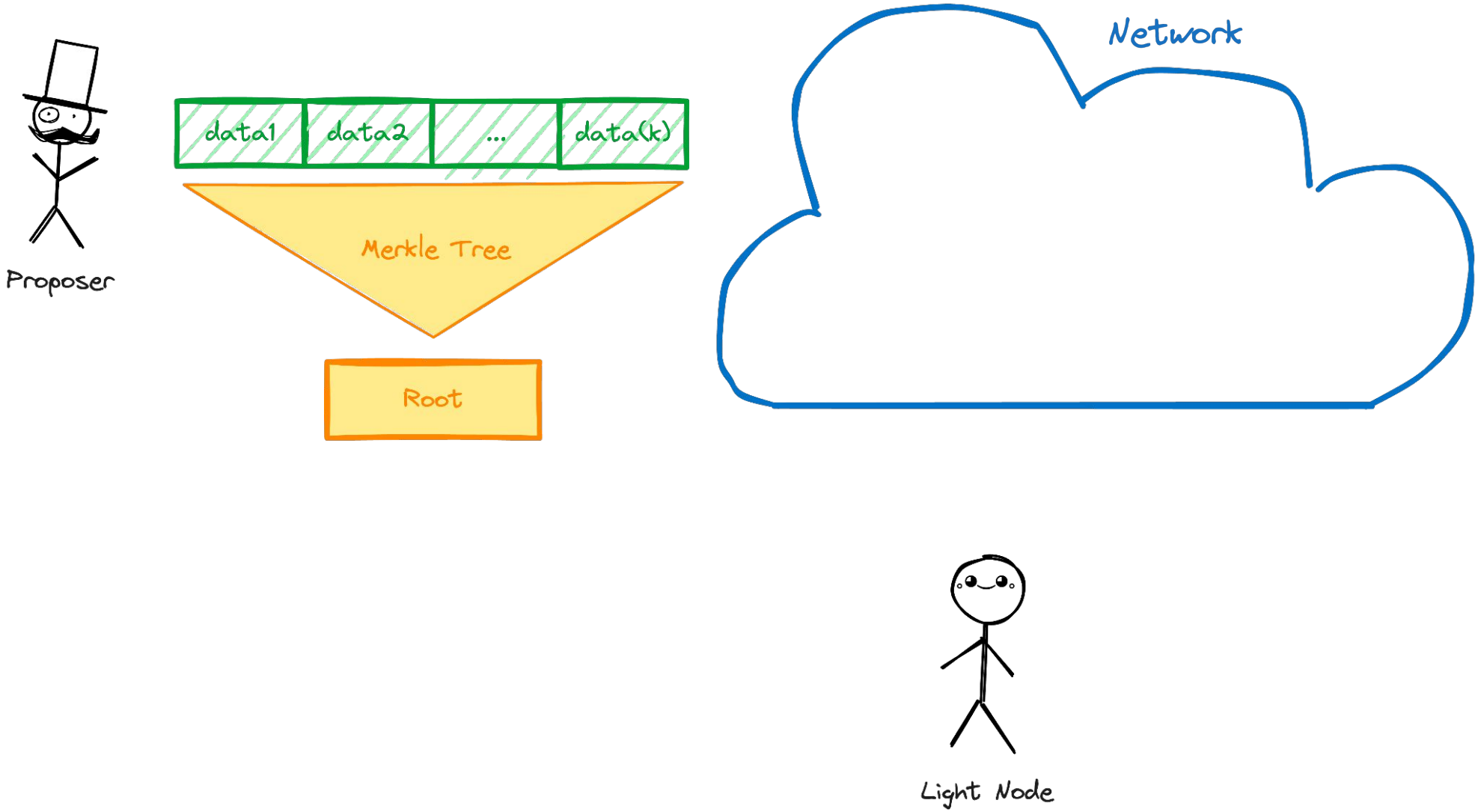
Parties

- *Proposer*
  - 블록을 만들고 네트워크에 전파하는 단일 주체
  - 악의적인 행동을 할 수 있음 (Data Withholding Attack)
  - DAS에서는 *Proposer*의 악의적인 행동을 방지하는 것이 주 목적
- *Full Node*
  - 트랜잭션 검증에 필요한 모든 데이터를 다운받아서 직접 모든 검증을 하는 노드
  - Minority Honesty Assumption
- *Light Node*
  - 블록 헤더만 다운받아 검증에 참가하는 노드
  - DAS를 통해 DA를 보장받으려는 주체



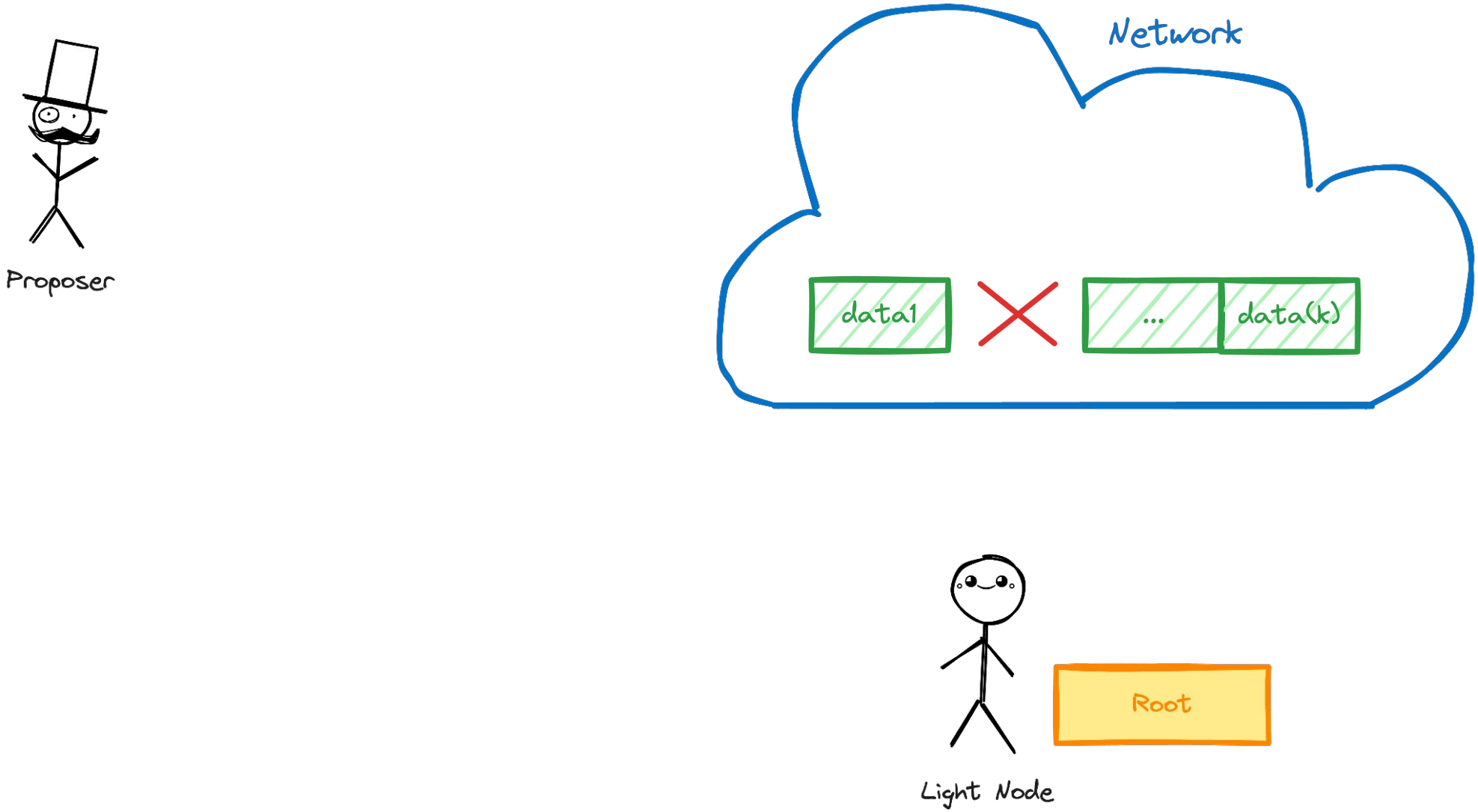
How to Check Data Availability

Without Erasure Coding



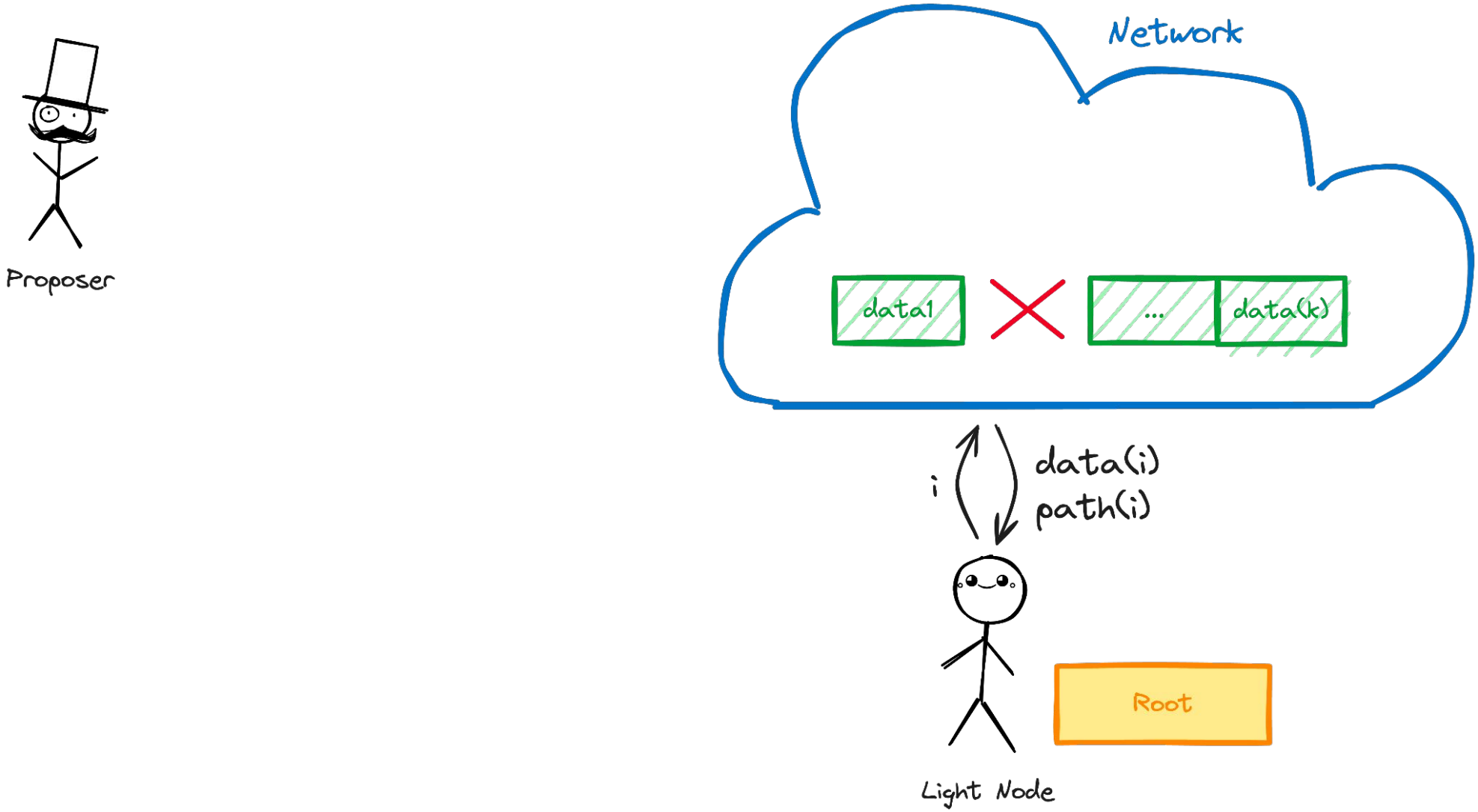
How to Check Data Availability

Without Erasure Coding



How to Check Data Availability

Without Erasure Coding



How to Check Data Availability

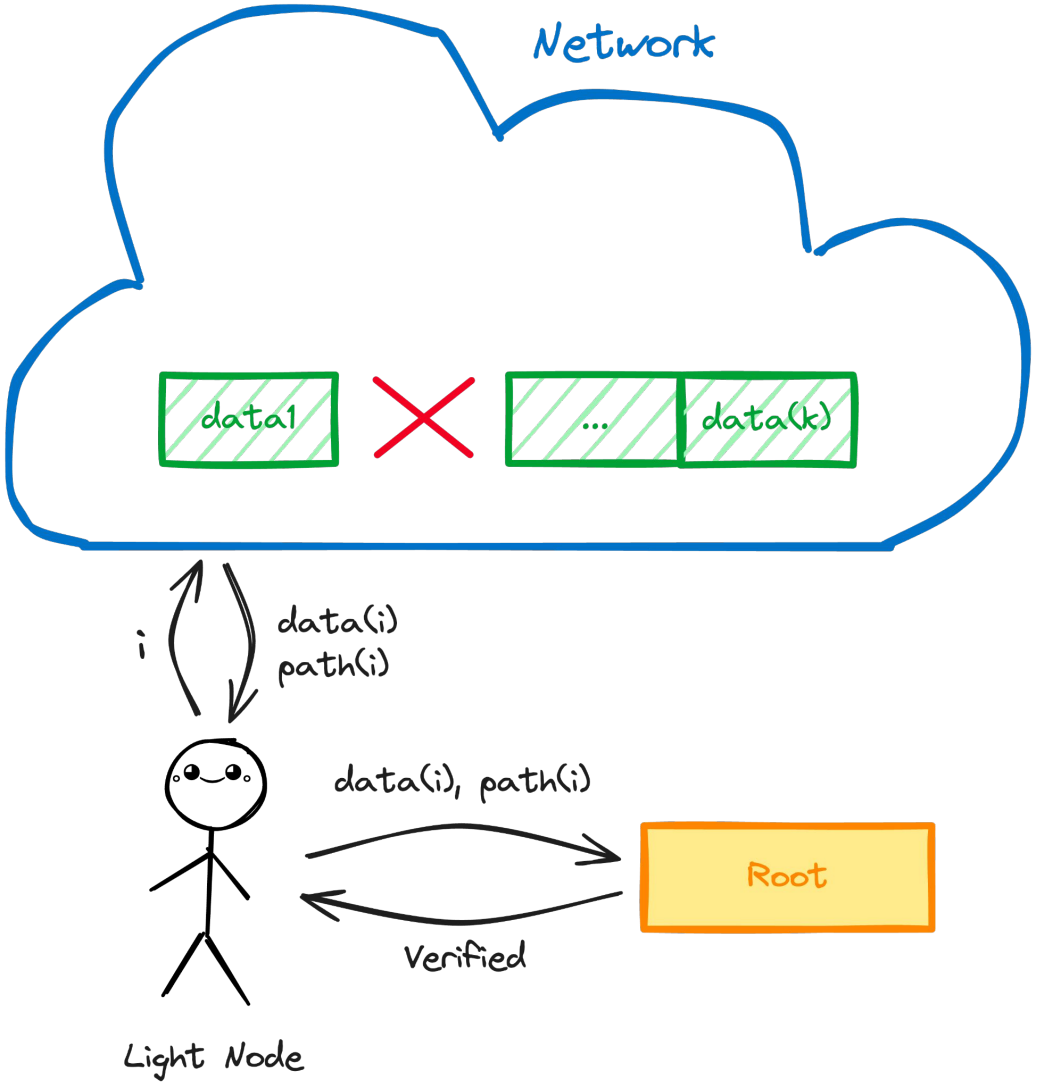
Why It's Bad

Data Withholding Attack

- Proposer가 할 수 있는 가장 효과적이고 강력한 공격은
- 전체 데이터 중 하나의 청크만 숨기는 것
- 그렇다면 이를 detect 하기 위해서는?

How Many Queries Needed?

- 전체 데이터 개수가 k에서 숨겨진 데이터가 1개라면
- 데이터를 q개 요청하면 그 중 숨겨진 데이터를 요청할 확률이 q/k
  - = DA가 없음을 검증할 확률
- 예를 들어, 전체 데이터가 10000개이고, 99.9%의 확신이 필요하다면 9990번 확인을 해보아야 함
- 매우 비효율적이고, 라이트 노드의 의미가 퇴색



Solution

First Introduced by This Paper

Fraud and Data Availability Proofs: Maximising Light Client Security and Scaling Blockchains with Dishonest Majorities

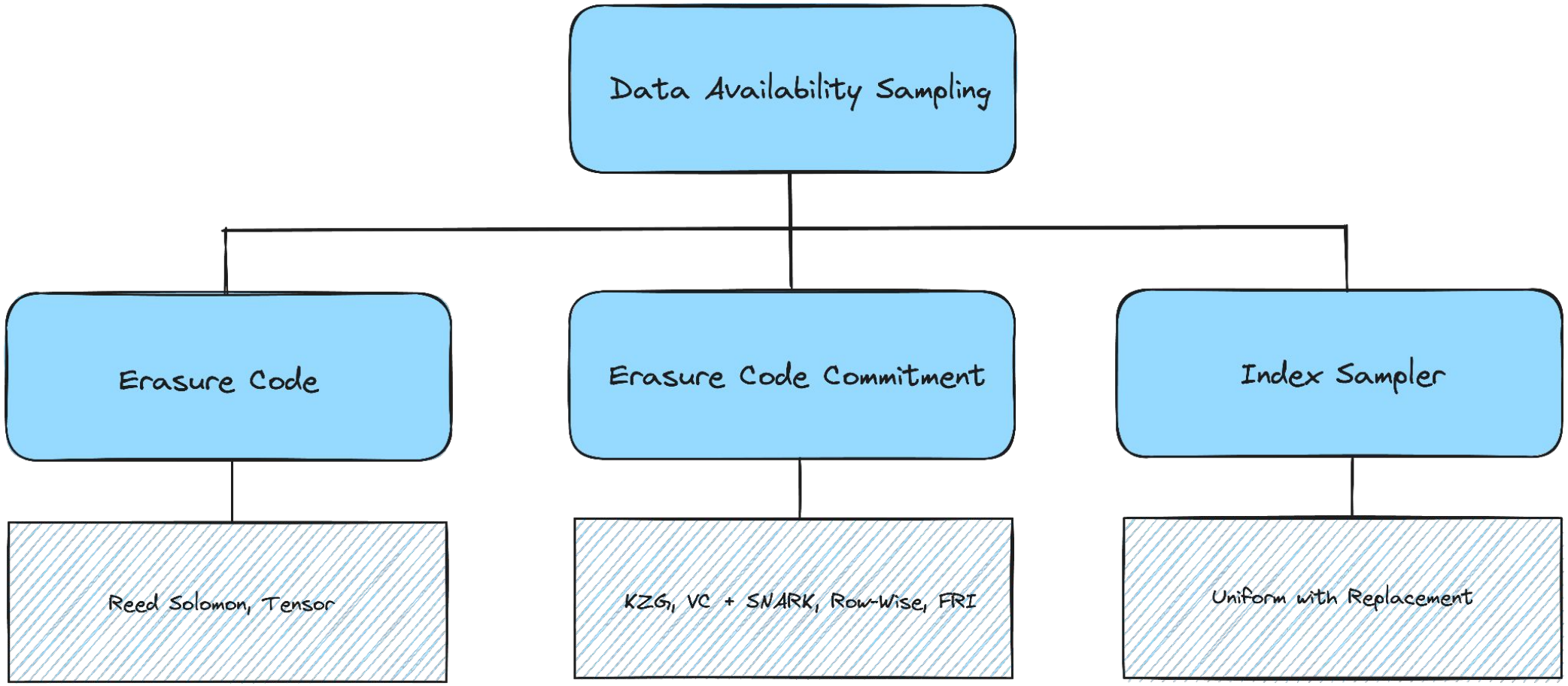
Mustafa Al-Bassam[1], Alberto Sonnino[1], and Vitalik Buterin[2]

[1] University College London
{m.albassam,a.sonnino}@cs.ucl.ac.uk
[2] Ethereum Research
vitalik@ethereum.org

Abstract. Light clients, also known as Simple Payment Verification (SPV) clients, are nodes which only download a small portion of the data in a blockchain, and use indirect means to verify that a given chain is valid. Typically, instead of validating block data, they assume that the chain favoured by the blockchain's consensus algorithm only contains valid blocks, and that the majority of block producers are honest. By allowing such clients to receive fraud proofs generated by fully validating nodes that show that a block violates the protocol rules, and combining this with probabilistic sampling techniques to verify that all of the data in a block actually is available to be downloaded, we can eliminate the honest-majority assumption for block validity, and instead make much weaker assumptions about a minimum number of honest nodes that rebroadcast data. Fraud and data availability proofs are key to enabling on-chain scaling of blockchains (*e.g.,* via sharding or bigger blocks) while maintaining a strong assurance that on-chain data is available and valid. We present, implement, and evaluate a novel fraud and data availability proof system.

Construction Framework

Three Main Components



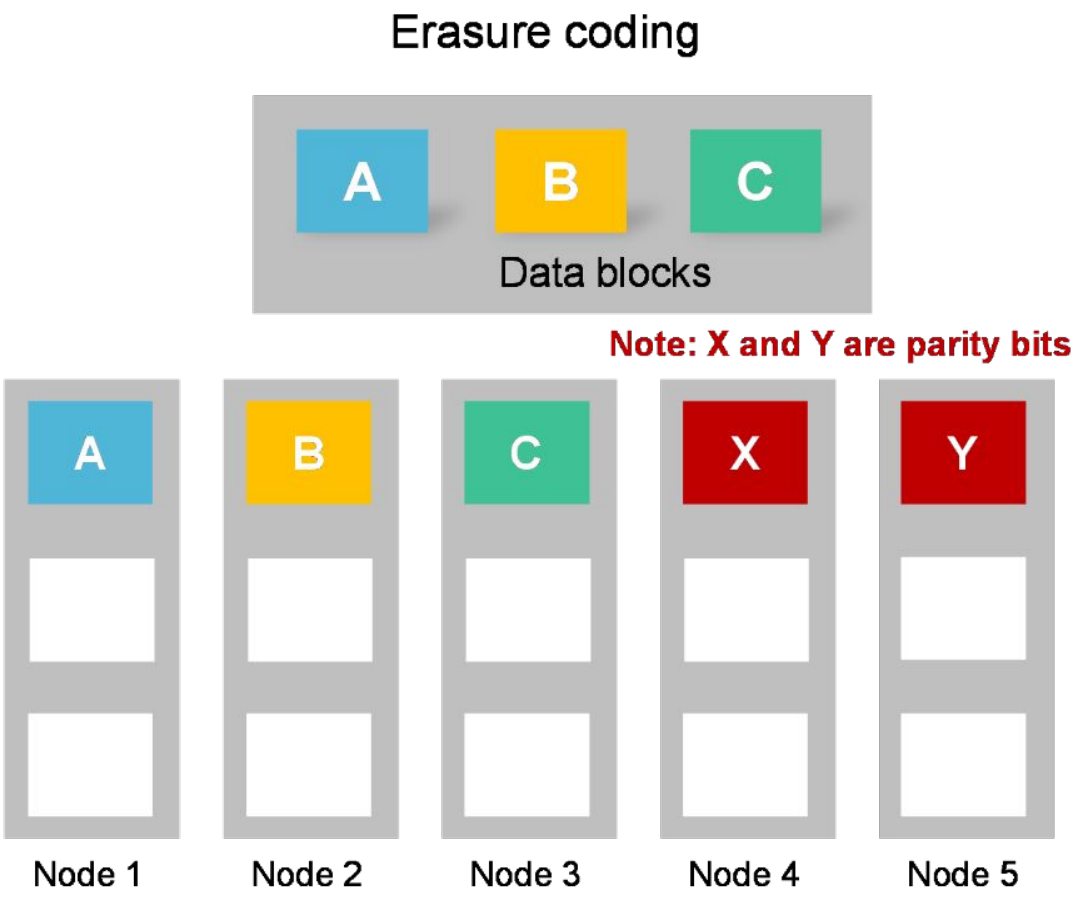
Erasure Code

What is Erasure Code?

- 오류 정정 코드의 일종
- 데이터에 Redundancy를 추가하여 변환
- 일부 데이터가 삭제되어도 이를 복구할 수 있도록 하는 코딩

Example of Error Correction Code (RAID)

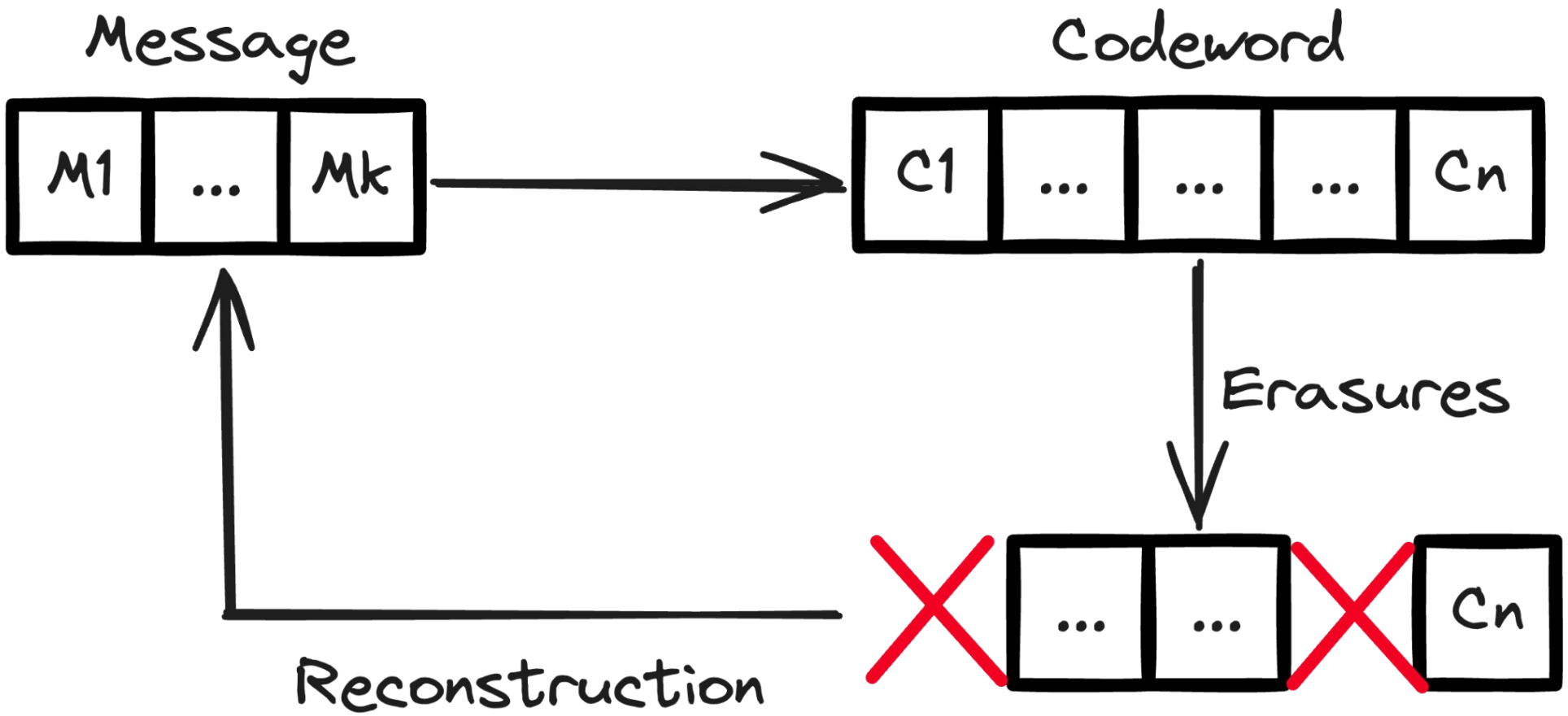
- A, B, C 라는 데이터를 저장하는 상황
- $P = A \oplus B \oplus C$ 인 패리티 데이터를 추가로 저장
- 예를 들어 만약 B가 손실 되면,
  $B = P \oplus A \oplus C$ 이므로 복구 가능
- 즉, 데이터 중 일부가 손실되어도 복구가 가능하도록 추가 데이터를 포함시키는 것



Source: [Replication and Erasure Coding]

Erasure Code

Objectives



Erasure Code

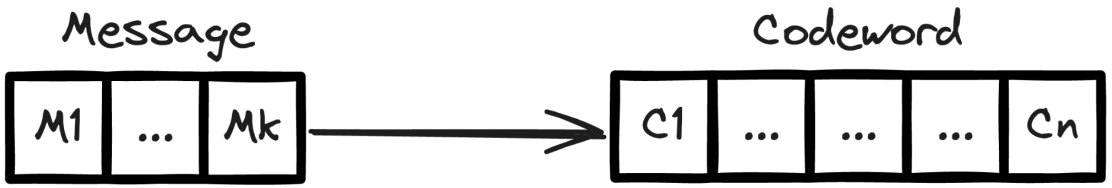
Reed Solomon Codes Construction Process (Original View)

1. 유한체와 평가점 선택

- 유한체는 보통 GF(2^m) 사용
- 평가점 n (코드워드의 길이)개 선$\{a_1, a_2, \ldots, a_n\} \subset F$
- 평가점은 유한체 내에 있으면 상관없지만, $p(a_i) = m_i \;\; \text{for } i = 1, 2, \ldots, k$)를 만족시키는 값을 선택 (빠른 Decoding을 위함)

2. 메시지 다항식 p(x) 생성

- 메세지의 각 데이터를 계수로 하는 k-1차 다항식 생성
- $p(x) = \sum_{i=1}^{k} m_i x^i \;\; \text{message} = [m_1, m_2, \ldots, m_k]$



3. 평가점을 넣어서 코드워드 계산

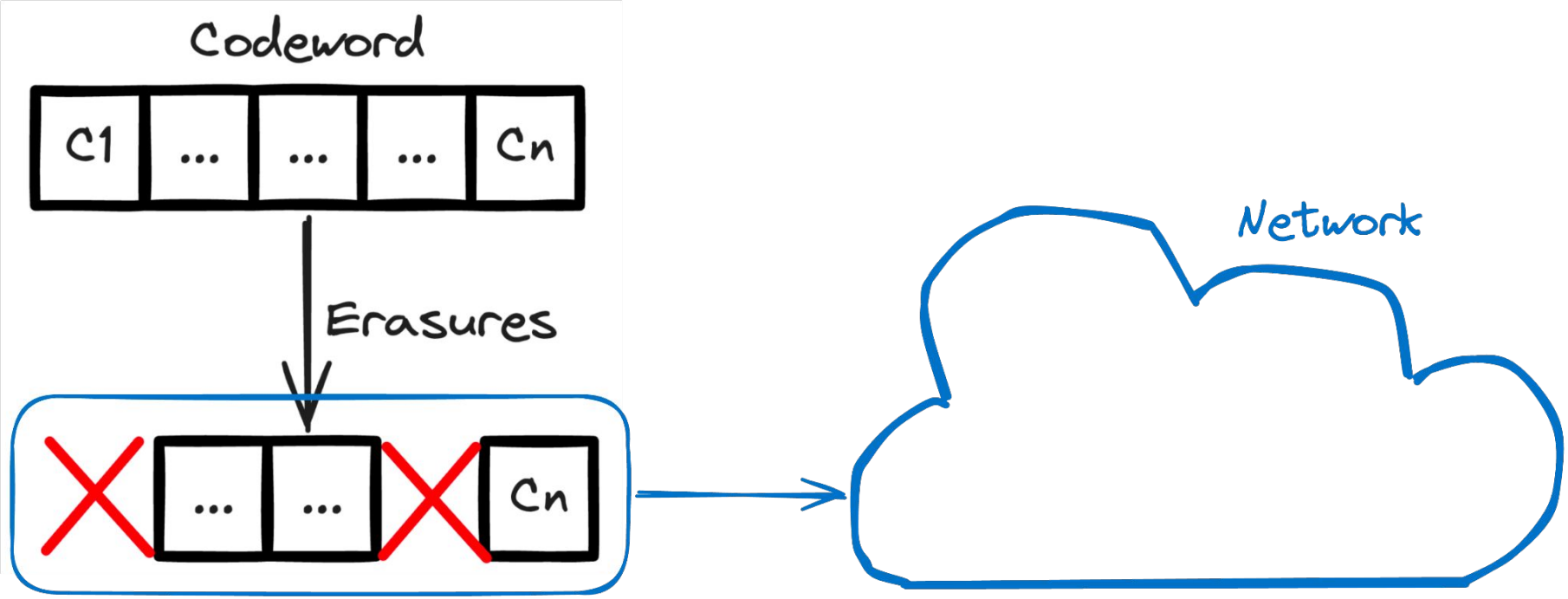
- $\mathbf{C} = \left\{ \left(p(a_1), p(a_2), \ldots, p(a_n)\right) \;\middle|\; p \text{ is a polynomial over } F \text{ of degree } < k \right\}$

Erasure Code

Reconstruction (Original View)

Current State After Coding

- 코드워드$\mathbf{C} = \left\{ \left(p(a_1), p(a_2), \ldots, p(a_n)\right) \,\middle|\, p \text{ is a polynomial over } F \text{ of degree } < k \right\}$ 가 네트워크 상에 공개
- 코드워드의 n개 데이터 중 일부가 소실 되었을 수 있음



Erasure Code

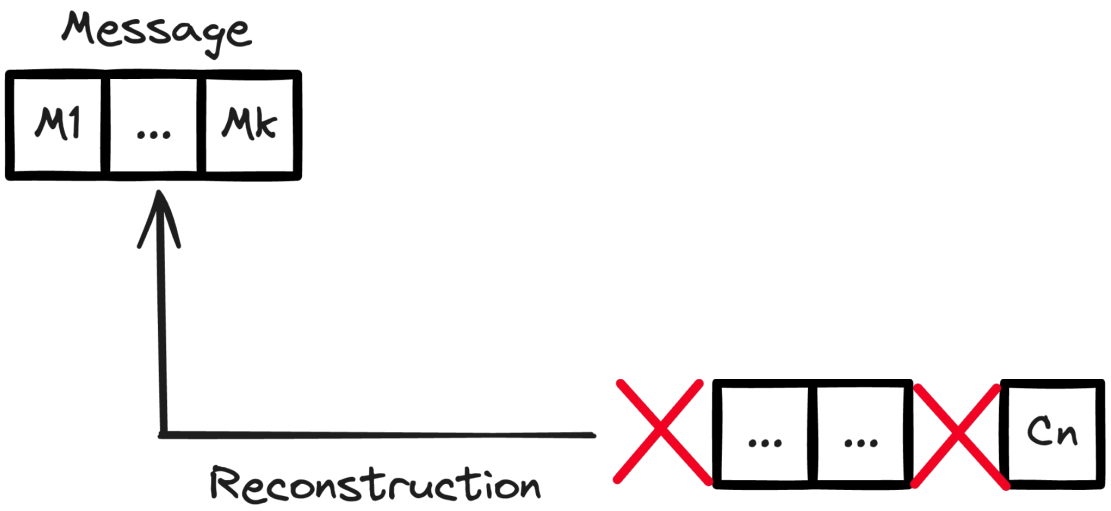
Reconstruction (Original View)

Objectives

- 코드워드$\mathbf{C} = \left\{ \left(p(a_1), p(a_2), \ldots, p(a_n)\right) \,\middle|\, p \text{ is a polynomial over } F \text{ of degree } < k \right\}$ 로부터 원래의 메시지$\text{message} = [m_1, m_2, \ldots, m_k]$ 를 복구

Mechanism

- 어떤 N-1차 다항식이 있을 때, 이 다항식 위의 점을 N개 이상 알고 있으면, 다항식을 계산해낼 수 있음 (Interpolation)
- 메세지 다항식 p(x)는 k-1차 다항식이므로 p(x) 위의 점을 k개 이상 알 수 있다면 p(x)를 알 수 있음
- 코드워드 C는 p(x) 위의 점의 집합이므로, 코드워드 C의 데이터 중 k개 이상이 남아있다면 p(x)를 구할 수 있음
- 다항식 p(x)의 계수가 원래의 메시지를 의미하므로, 원래의 메시지를 얻게 됨



Conclusion

- 네트워크에 코드워드 C의 데이터 n개 중 k개 이상 소실되지 않았다면, 원래의 메시지를 복구 가능 **=> Data Availability가 있음**

Erasure Code

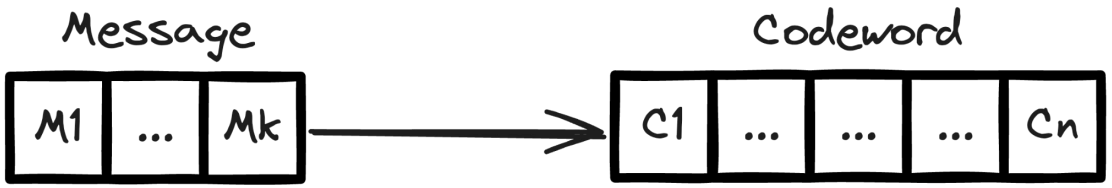
Reed Solomon Codes Construction Process (BCH View)

1. 유한체와 Generator 다항식 선택

- 유한체와 유한체의 원시원소 α를 선택
- 원시원소 α로 n-k차 다항식 생성
- $g(x) = \left(x - \alpha^{i}\right)\left(x - \alpha^{i+1}\right)\cdots\left(x - \alpha^{i+n-k-1}\right) = g_0 + g_1 x + \cdots + g_{n-k-1}x^{n-k-1} + x^{n-k}$

2. 메시지 다항식 p(x) 생성

- 메세지의 각 데이터를 계수로 하는 k-1차 다항식 생성
- $p(x) = \sum_{i=1}^{k} m_i x^i \quad \text{message} = [m_1, m_2, \ldots, m_k]$



3. 코드워드 다항식 s(x)와 코드워드 생성

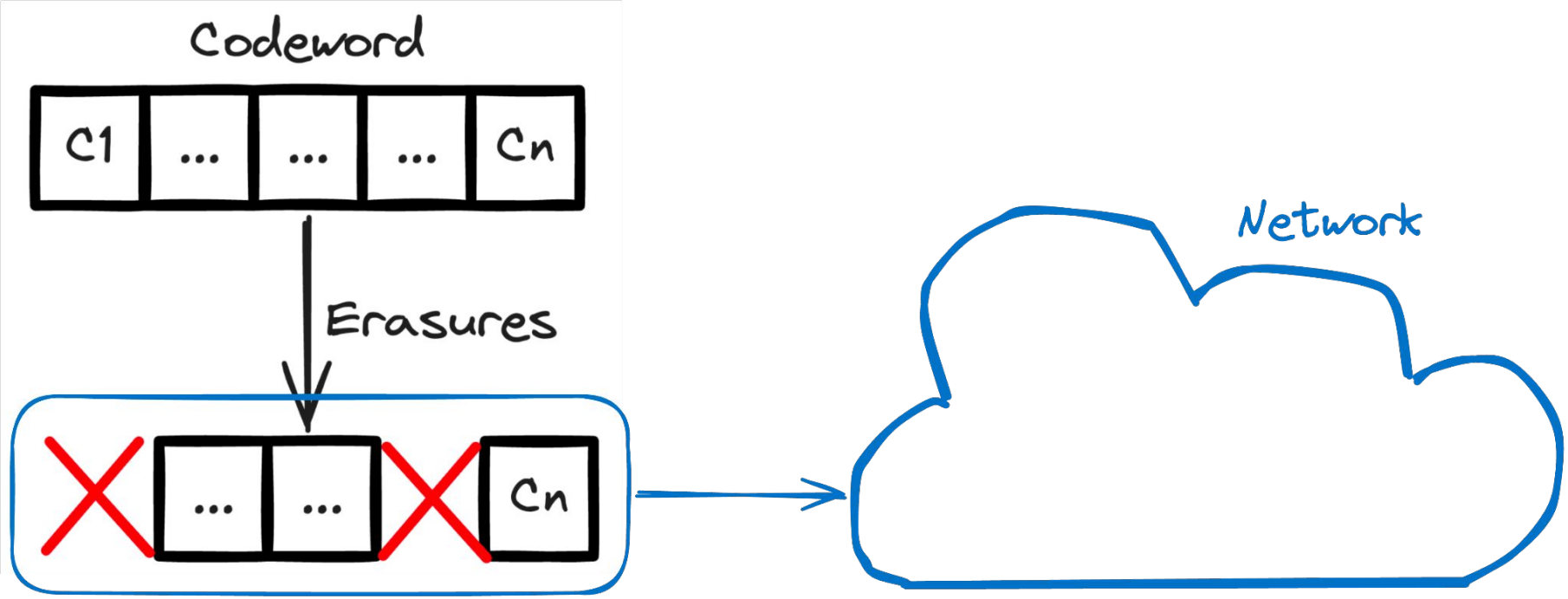
- $s(x) = p(x) \cdot g(x)$
- $s(x) = \sum_{i=1}^{n} c_i x^i \quad \text{codeword} = [c_1, c_2, \ldots, c_n]$

Erasure Code

Reconstruction (BCH View)

Current State After Coding

- 코드워드$\text{codeword} = [c_1, c_2, \ldots, c_n]$ 가 네트워크 상에 공개
- 코드워드의 n개 데이터 중 일부가 소실 되었을 수 있음

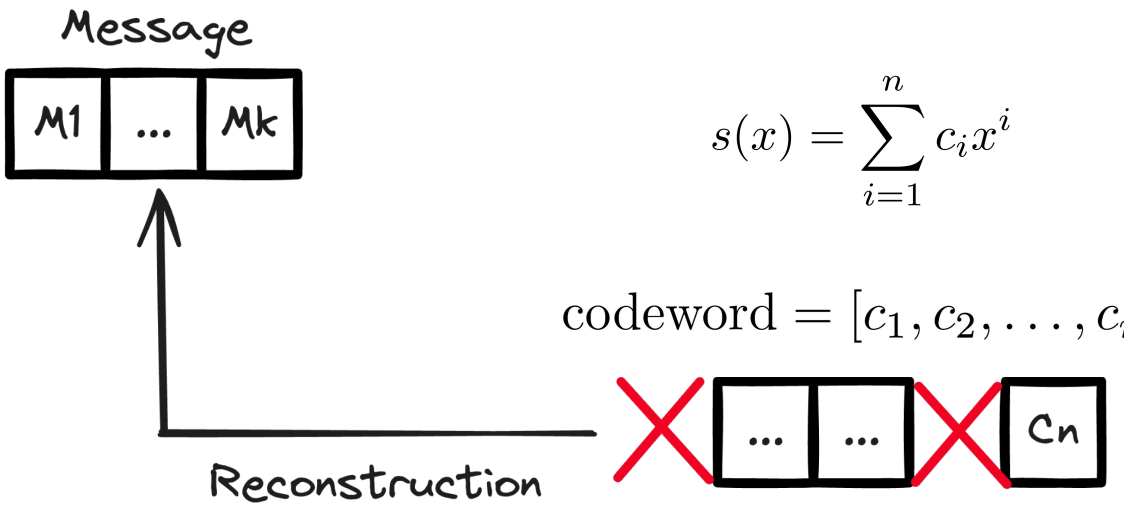


Erasure Code

Reconstruction (BCH View)

Objectives

$$\text{message} = [m_1, m_2, \ldots, m_k]$$



$$s(x) = \sum_{i=1}^{n} c_i x^i$$

$$\text{codeword} = [c_1, c_2, \ldots, c_n]$$

Mechanism

1. Codeword Reception & Error Detection
2. Syndrome Calculation
   - $S_j = r(\alpha^j)$
3. Error Locator Polynomial Calculation
   - $\Lambda(x) = \prod_{i=1}^{t} (1 - x\alpha^{i_i})$
   - Berlekamp-Massey algorithm
4. Finding Error Locations
   - $\text{find } i \text{ such that } \Lambda(\alpha^{-i}) = 0$
   - Chien search algorithm
5. Calculating Error Values
   - $e_i = -\dfrac{S(\alpha^{-i})}{\Lambda'(\alpha^{-i})}$
   - Forney algorithm
6. Codeword Correction $c(x) = r(x) - e(x)$

2. Data Availability Sampling

Erasure Code

How to Ensure Data Availability with Erasure Codes

Objectives

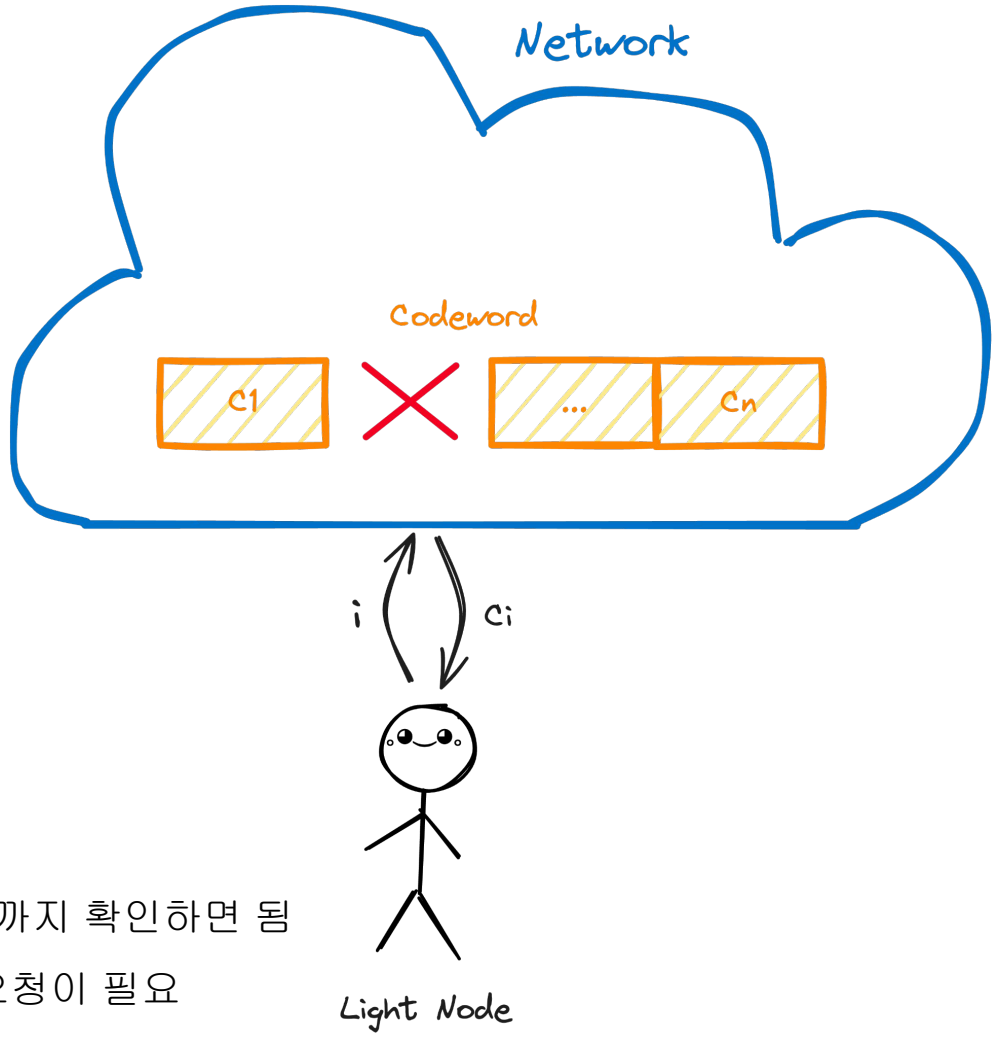
- 길이 n의 코드워드 중 k개 이상이 네트워크 상에 존재하는지 확인
- 위를 확인한다면 네트워크 상에 Data Availability가 있다고 확신할 수 있음

Example Settings

- k = 10,000, n = 20,000
- 코드워드 데이터 20,000개 중에 10,000개 이상만 존재하면 DA가 있음
- DA가 없다면, 네트워크에 데이터를 요청했을 때
  50% 이상의 확률로 데이터가 존재하지 않음

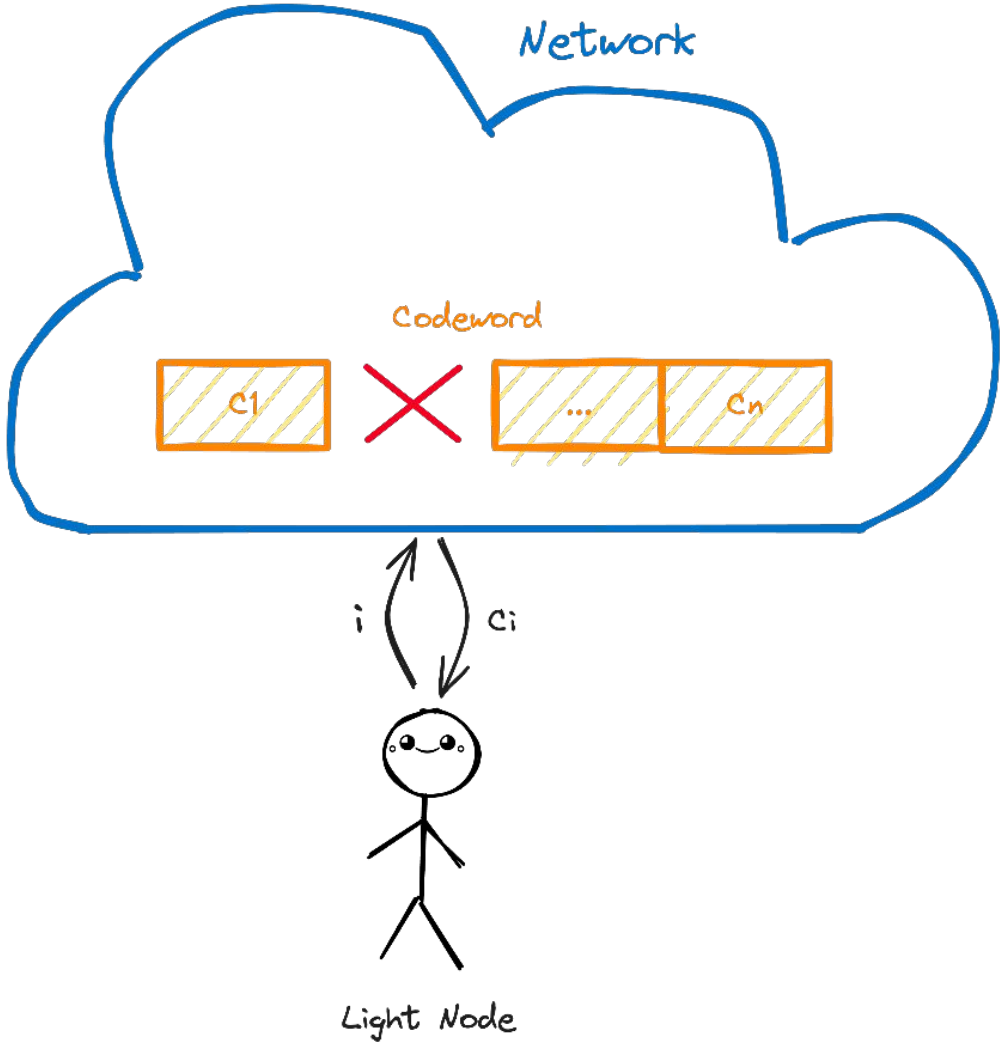
Sampling

- q번 요청해서 전부 데이터를 확인했다면 DA가 없을 가능성은 (½)^q 이하
- 즉, q번 요청해서 전부 존재했다면 1-(½)^q의 신뢰도로 DA가 있음
- q = 10일 때, 약 99.9% 신뢰도로 DA가 있음을 확인
- 중간에 데이터가 없었다면 시행 횟수를 늘리고, 원하는 신뢰도에 도달할 때까지 확인하면 됨
- Erasure Coding 없이 확인하면 99.9% 신뢰도를 얻기 위해서는 9990번의 요청이 필요
- k와 n의 비율이 동일하면 k값이 커져도 필요한 요청 횟수는 동일하므로 Scalable 함

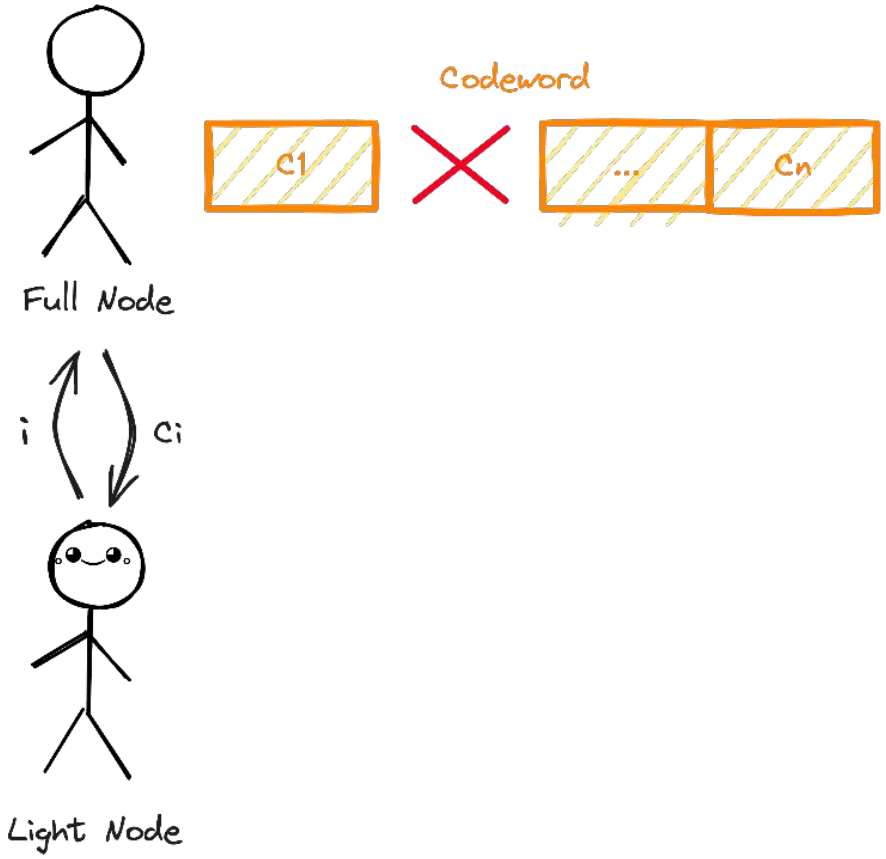


Erasure Code Commitment

But Can We Trust the Data Given From the Network?



==



Erasure Code Commitment

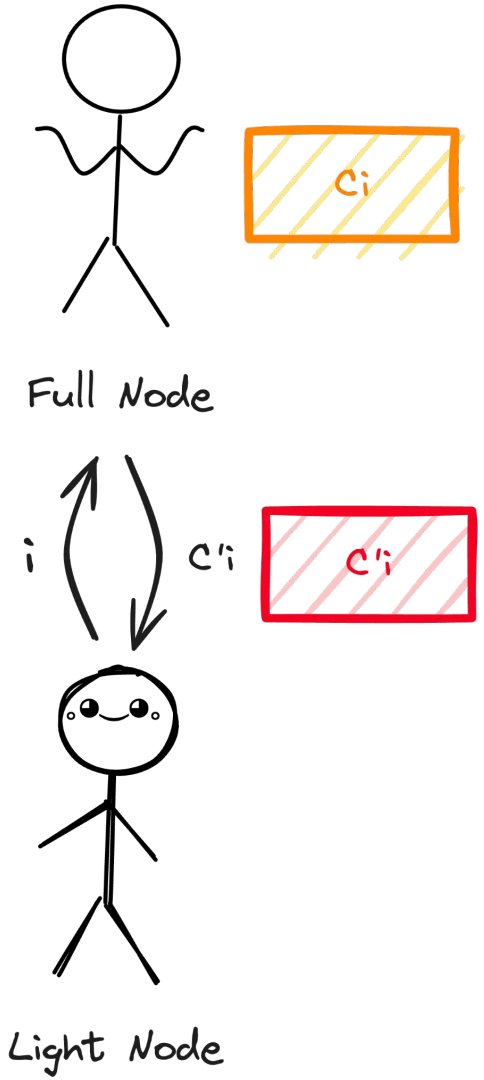
Giving the Wrong Data

Problem Statement

- 라이트 노드는 Data Availability Sampling을 위해 풀 노드에게 데이터를 요청
- 만약 풀 노드가 올바르지 않은 데이터를 준다면?

Verification Needed

- 풀 노드가 준 데이터가 올바르게 원래 메시지로부터 생성된 코드워드의 일부분인지 알 수 없음
- 풀 노드가 제공하는 데이터가 실제 코드워드의 일부분이라는 것에 대한 증명과 검증이 필요



Erasure Code Commitment

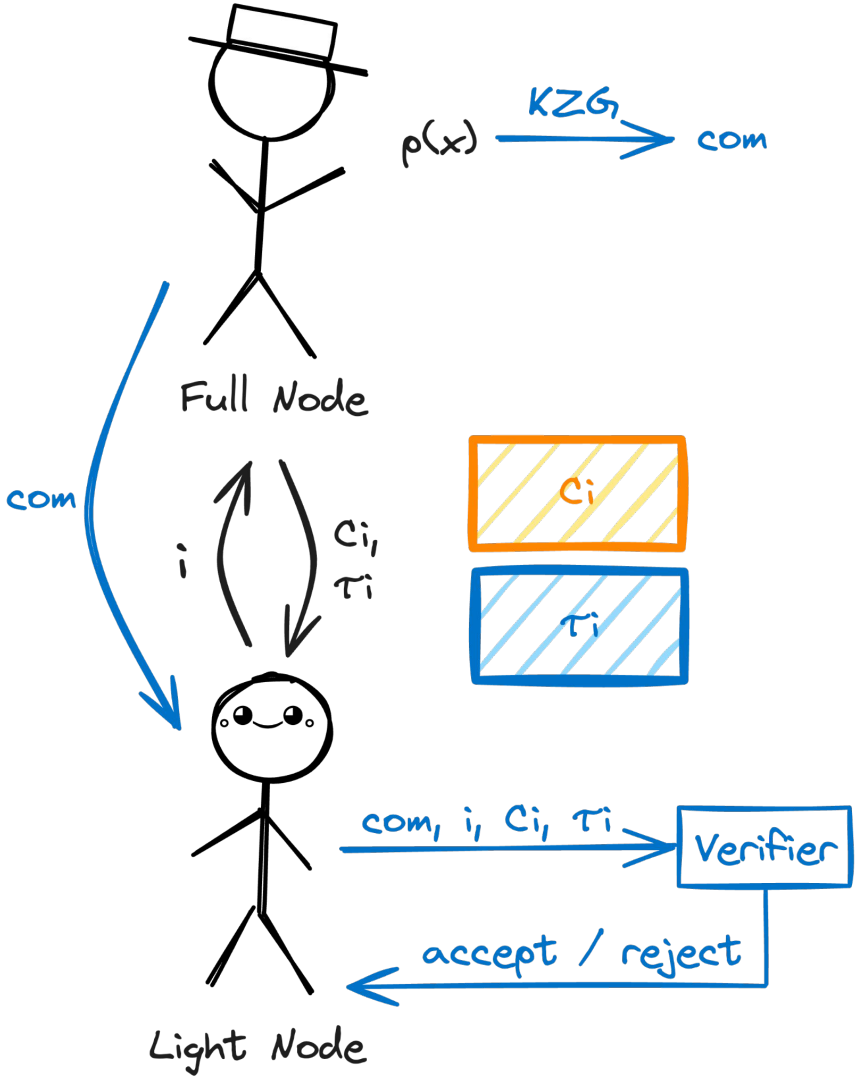
Code Binding

Polynomial Commitment

- 코드워드의 각 값은 메시지 다항식 p(x)의 평가값
  - $c_i = p(a_i)$
- 즉, 제공한 데이터가 실제로 다항식의 한 평가값임을 증명하면 됨
- 다항식 커밋먼트를 활용하면 이를 증명할 수 있음
- KZG Commitment, FRI 등 활용 가능
  - FRI는 다항식 커밋먼트는 아니지만 여기서 사용 가능

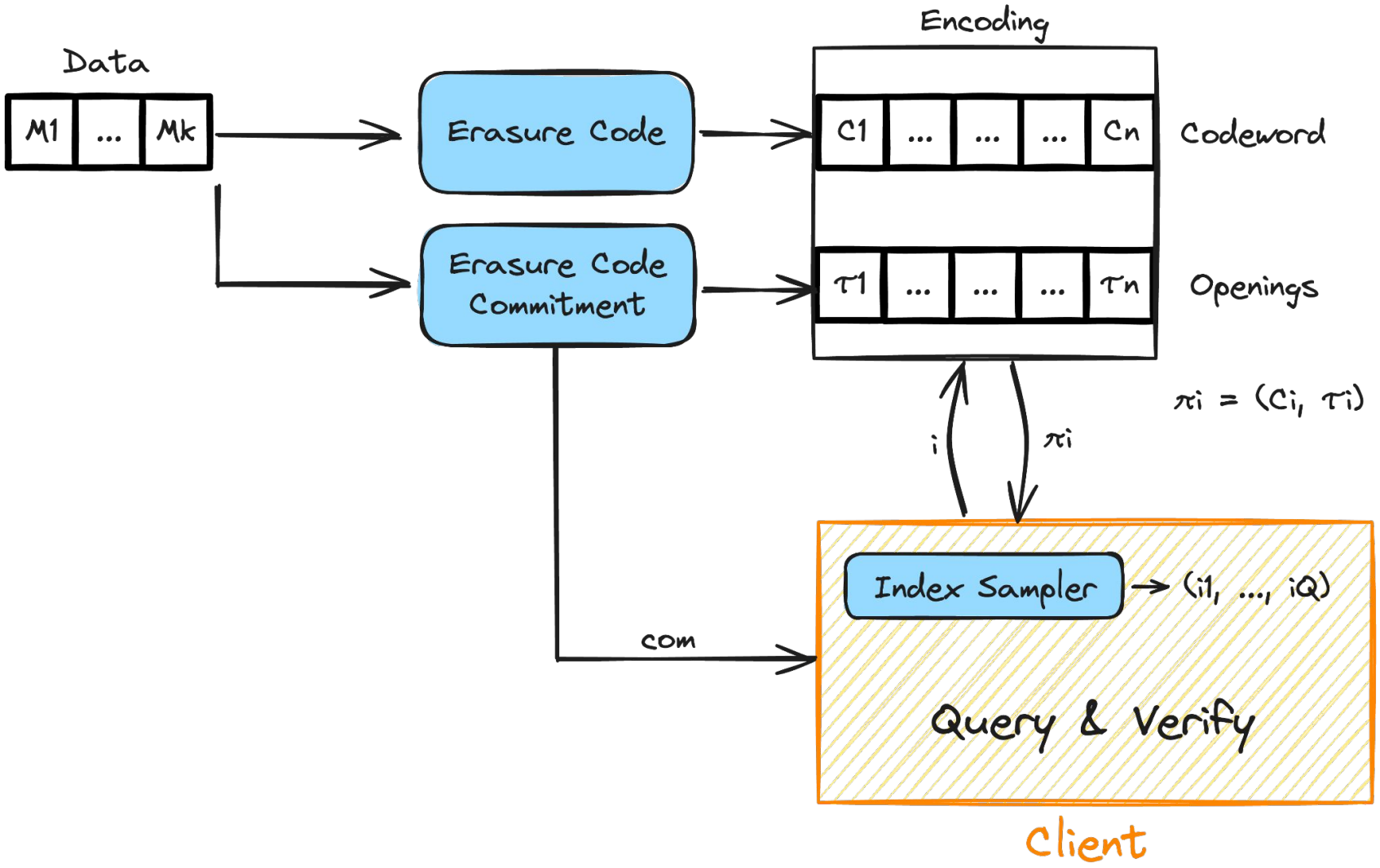
How It Works

- 메시지 다항식 p(x)에 대하여 commit, 커밋값 com을 생성하고 이를 사전에 네트워크에 공개
- 검증자가 코드워드의 i번째 자리에 대해 요청했을 때, 해당 position을 open하여 증명 τ_i를 생성하고 c_i와 함께 제공
- 검증자는 커밋값, 코드워드, 증명을 이용하여 다항식



Construction

All Together



Data Availability Projects

Comparison Table

	Ethereum (EIP 4844)	Celestia	Avail	EigenDA	DAC (Data Availability Committee)	FRIDA
Consensus Mechanism	GHOST + Casper 3	Tendermint	GRANDPA + BABE	Committee Based	Committee Based	TBD?
Decentralization	Excellent	Good	**Great?** (is it better than Celestia as they claim to be?)	Good	Poor	
Execution Environment	Yes	No	No	No	No	No
Ability to Scale	No	Yes	Yes	Yes	No	Yes
Cost	Very High	Low	Low	TBD	Very Low	
Block Time	12 sec	15 sec	20 sec	N/A	?	
Time to Finality	12 min	15 sec	20 sec	12 min	?	
DAS support	No	Yes	Yes	No	No	Yes
Light Node Security	Honest Majority	Honest Minority	Honest Minority	N/A	?	Honest Minority
Encoding Proof Scheme	Validity Proofs	Fraud Proofs	Validity Proofs	Validity Proofs	?	Validity Proofs

Source: Nethermind

References

3. References

Websites

Data Availability

- What is Data Availability?
- Data Availability Bottleneck
- Data Availability Checks

Data Availability Sampling

- Ethereum has blobs. Where do we go from here?

Reed-Solomon Codes

- Reed-Solomon Error Correction

KZG Commitment

- KZG Polynomial Commitments

Project Docs

- Celestia Docs
- Avail Docs

Papers

Data Availability Sampling

- Foundations of Data Availability Sampling
- Fraud and Data Availability Proofs: Maximising Light Client Security and Scaling Blockchains with Dishonest Majorities
- FRIDA: Data Availability Sampling from FRI

KZG Commitment

- Constant-Size Commitments to Polynomials and Their Applications

Thank You