발표하기에 앞서 A가 B에게 송금을 하는 상황을 상상해보겠습니다. 손으로 직접 거래하는 것이 아니라 인터넷 송금이 사용자끼리 거래가 가능할까요 ?

**(넘김)**

아닙니다 다음 그림과 같이 A가 은행에 보내고 은행이 B에게 다시 보내는 즉, 은행을 거쳐서 꼭 지나쳐야하는 중앙집중화 시스템을 지니고 있습니다.

하지만, 말을 반대로 하여 중앙시스템에서 은행과 같이 한 곳에 집중되어있는 기관이 없어지거나 문제가 생긴다면, 각 개인은 아무것도 할 수 없게 됩니다.

그러한 일은 실제로 일어났었습니다.

**(넘김)**

2008년에 세계금융위기가 발발하며 금융 시장에 큰 충격을 주었습니다. 다수의 사람들은 자산의 가치인 기본적인 화폐에 대한 신뢰도를 잃고 중앙기관의 정책이 자산에 영향을 주는 것에 불만을 가졌습니다. 중앙에서 관리하는 화폐가 아닌 모든 사람이 동등하게 관리할 수 있는 화폐는 없을까?의 생각으로

**(넘김)**

블록체인을 기반으로 한 암호화폐 비트코인이 처음으로 나오게 되었습니다.

**(넘김)**  
기존의 은행과 다른 기관들은 중앙집중화된 시스템을 사용하여 모든 작업을 한 노드에서 처리하는 방식으로 운영되었습니다. 이러한 방식은 작업이 증가할 경우 비효율적인 작업 처리 문제를 야기할 수 있습니다. 이에 대한 대안으로 **(넘김)** 분산 시스템을 도입하였습니다.

**(넘김)**

분산 시스템은 한 작업을 단일 노드가 아닌 여러 노드에 분산시켜 처리하는 방식입니다. 작업을 병렬적으로 처리함으로써 매우 효율적인 결과를 얻을 수 있었습니다. 이러한 분산 시스템은 중앙 서버가 존재하며, 다수의 노드로 작업을 분산시켜 실행하는 방식으로 동작합니다.

**(넘김)**

이로 인해 효율성은 증대되었지만, 여전히 중앙화를 기반으로 하기 때문에 신뢰성의 문제는 해결되지 못하였습니다.

**(넘김)**

그래서 우리는 탈중앙화라는 개념을 도입하였습니다. 분산과 탈중앙화는

**(넘김)**

작업 처리 방식에서 다릅니다. 분산은 작업을 다수의 노드에 분산하여 각각 다른 작업을 수행

이에 반해 탈중앙화는 모든 노드가 동일한 작업을 중복하여 수행합니다.

**(넘김)**

분산 시스템은 통제 서버가 개입되어 작업을 분배하고 종합하는 역할을 수행합니다. 반면에 탈중앙화 시스템은 네트워크 내 모든 노드가 동등한 지위를 가지며, 중요한 역할을 수행하는 특정 서버가 없습니다.

**(넘김)**

분산은 효율성은 높지만 신뢰성에 제약이 있고 지배적인 시스템입니다. 탈중앙화는 효율성은 희생하지만 결과의 신뢰성을 향상되고 동등한 P2P시스템입니다.

**(넘김)**

정리하면, 블록체인은 탈중앙화된 분산 데이터베이스 시스템으로, 중앙 기관 없이 사용자간의 거래를 체인 형태로 기록하여 안전하고 신뢰할 수 있는 서비스를 가능케 합니다.

**(넘김)**

먼저, 블록 구조는 블록 헤시, 블록헤더, Transaction 개수, 블록 바디로 구성되어 있습니다.

거래 카운트는 바디에 존재하는 트랜잭션의 개수입니다.

**~~(해당 부분이 존재하는 이유는 트랜잭션의 수수료 계산 및 블록의 크기 제한 문제 이유 때문에 기록해야합니다.)~~**

**(넘김)**

블록 헤시는 블록의 header 정보인 버전, 이전 블록 해시, 머클루트, 타임, bits/target, Nonce 를 모두 더한 합을 SHA-256으로 변환한 결과값입니다.

**(넘김)**

그렇다면 왜 블록 헤시값이 존재해야하는지 의문점이 듭니다.

이유로는 블록체인의 신뢰성을 위해 데이터 무결성을 위해서입니다. 헤더중에 하나의 값만 바뀌어도 전체의 값이 바뀌기 때문입니다.

**(넘김)**

예를들어, A -B – C 블록 순이라면 C의 블록 해시값은 A 블록해시 값의 영향을 받기에 데이터 무결성이 유지될 수 있는 것입니다.

**(넘김)**

블록 헤더는 블록 헤시값을 결정하는 부분입니다.

먼저 해당 블록의 버전 정보가 있습니다. 버전 정보는 각 블록이 사용하는 프로그램의 버전 정보인데, 버전마다 적용하는 규칙이 다르기 때문에, 정말 중요한 요소입니다.

이전 블록 해시값 입니다. 이전 블록의 해시값은 데이터 무결성 확인과 블록체인의 검증에 있어서 중요한 값입니다.

Time은 해당 블록의 생성시각 입니다.

Bit/target, Nonce는 작업증명에 사용되는 값이다.

마지막으로 머클루트 입니다. **머클루트를 쉽게 설명하는 그림이 있어 보여드리겠습니다.**

**(넘김)**

블록체인에서 거래내역을 트랜잭션이라고 부르는데 해당 값은 블록의 바디에 저장되어 있습니다. 머클루트는 트랜잭션을 머클트리 형태로 쌓아서 다수의 트랜잭션 해시 값을 하나의 해시값으로 만들어 머클루트 필드에 저장합니다. 그래서 하나라도 변조되면 변조 유무를 쉽게 파악하여 데이터 무결성을 유지할 수 있습니다.

**(넘김)**

블록 구조 최종 Flow입니다. Ppt 보고 설명

**(넘김)**

**블록체인 연결입니다.** 피피티 보고

----------------------------넘기기

**(넘김)**

작업증명 방식은 비트코인 네트워크에 합의 알고리즘으로 사용되며, 풀기 어려운 문제를 가장 빨리 해결한 사람에게 블록 생성 권한과 코인 보상을 제공하는 방식입니다. 이를 통해 비트코인 블록체인은 분산된 네트워크에서 안전하고 신뢰할 수 있는 거래 기록을 유지합니다.

**(넘김)**

비트코인에서는 해시퍼즐을 작업 증명 방식으로 도입합니다. 해시퍼즐은 입력 데이터를 해시 함수에 적용하여 원하는 결과를 얻는 것이 어렵지만, 결과를 검증하는 것은 간단한 작업입니다

**(넘김)**

네트워크 서버에서 목표 해시값을 받고 해시 퍼즐을 풀기 위해 두 값을 합하여 해시를 실행하여 해시숫자를 얻습니다. 해시값이 목표값보다 작으면 성공한 것이므로 작업증명이 달성되어 블록이 하나 추가합니다.

하지만, 논스값이 작을수록 해시값은 크기때문에, 논스값에 따라 목표값보다 해시값이 큰 경우가 있습니다. 이런 경우는 퍼즐을 풀지 못한것이고, 논스값을 늘려서 다시 도전합니다.

**(넘김)**

해시함수를 푸는 것은성능이 좋은 컴퓨터로 빠르게 풀 수 있는 것이 아니라 무차별 대입으로 풀어야합니다. 그렇기 때문에 비트코인에서 작업 증명은 해시함수를 이용하는 방식이므로 그 자체로 안전합니다.

블록을 수정하려면 이전 블록부터 현재까지의 모든 작업 증명을 다시 수행해야 하므로, 이는 많은 자원과 시간이 필요합니다. 이로 인해 공격자는 블록체인을 조작하기 어려워지며, 비트코인 블록체인은 작업 증명 방식을 통해 안전하게 보호될 수 있습니다. 현실적으로 공격자는 공격하려면 10분내로 블록체인의 모든 노드의 해시값을 변경할 수 있어야 하기에 현실적으로 불가능합니다.

**(넘김)**

단점으로는 처리능력을 최적화하는 특수 장비를 사용해야하며, 이러한 하드웨어 구입을 위한 초기 투자 그리고 가장 최악의 단점인 많은 에너지가 소비됩니다.

----------이더리움-------------

**(넘김**)

비트코인과 이더리움은 다양한 차이점을 갖고 있습니다.

**(넘김**)

크게 3가지가 있는데요

**(넘김)**

첫번째로는 비트코인의 발행량은 2100만개로 제한되어 있고(**19,427,075 BTC 93%)** 이더리움은 무한개로 개수 제한의 차이가 있습니다.

**(넘김)**

두번째는 스마트 컨트랙트의 유무입니다. 스마트 컨트랙트는 기존 비트코인을 생각해보면 단순히 사용자 간의 거래만 허용되었는데, 그것을 넘어 프로그램 코드를 추가한 것입니다. 더하여 계약조건이 충족되면 자동으로 계약 내용이 실행되도록 하는 시스템입니다.

**(넘김)**

마지막으로, 블록체인은 UTXO 모델을 사용하되, 이더리움은 계정 모델을 사용합니다. 비트코인은 외부 프로그램내에서 사용하지 않은 잔액인 UTXO인 암호화폐의 출처를 모아 특정 사용자의 잔액을 표현해주었는데, 이더리움은 각 계정의 잔액을 외부의 프로그램이 아닌 이더리움 자체 내에서 지원해주는 성격인 계정 모델을 채택하여 사용합니다. 이러한 차이를 유지하기 위해 이더리움은 계정이라는 개념을 추가하였습니다.

**(넘김)**

이더리움의 계정은 기본 거래 계정인 EOA, 계약 계정인 CA가 있습니다.

EOA는 개인키가 있어 풀 수 있고, CA는 개인키가 없습니다.

EOA는 스마트 컨트랙트 코드가 없고 CA는 코드가 있습니다.

EOA는 개인키로 제어가 가능하고, CA는 스마트 컨트랙트 로직에 의해 제어됩니다.

EOA는 자금송금 or 컨트랙트 실행이 가능하지만 CA는 개인키가 없어서 혼자서는 트랜잭션 실행이 불가합니다.

그래서 EOA는 개인키를 사용하여 혼자서도 트랜잭션 실행이 가능하며, CA는 외부 트랜잭션의 응답으로만 실행이 됩니다.

**(넘김)**

논스 : EOA의 경우, 해당 계정의 주소로부터 시작한 트랜잭션들의 수를 의미한다.

CA의 경우, 해당 계정에 의해 생성된 컨트랙트의 수를 의미한다.

Balance : 해당 계정이 소유하고 있는 암호화폐의 양

CodeHash : EOA의 경우, 코드를 저장할 수 없으므로 empty 문자열의 해시값이다.

StorageRoot : 계정에 관한 정보들을 머클 패트리샤 트리로 하나의 해시값으로 저장함

**(넘김)**

이더리움의 블록 구조에는 다양한 헤더 요소와 바디로 이루어져있습니다.

헤더를 알아보겠습니다.

**(넘김)**

**이더리움에서 가스는 트랜잭션과 스마트 컨트랙트 실행할 때 사용되는 작업의 비용입니다.**

**(넘김)**

머클루트는 비트코인에서 사용되었던 필드와 비슷한 필드입니다. 이더리움에서는 머클루트의 종류는 3가지입니다.

stateRoot는 이더리움의 상태 트리를 나타내는 값으로써 모든 계정의 현재 상태를 저장하는 데이터 구조로, 계정의 잔액, 상태 변수, 스마트 컨트랙트 등의 정보가 담겨있습니다.

transactionRoot는 Transaction을 하나의 해시값으로 만든 값입니다

reciptsRoot는 트랜잭션 실행 후 결과를 요약한 데이터 해시 값입니다.

이후 이더리움 트리 구조에 대해서 설명할 때 다루겠습니다.

**(넘김)**

이러디움 전체 구조를 보기 전에 반드시 알아야할 트리 구조에 대해서 설명 드리겠습니다.

비트코인에서는 철수 (20,80) 에서 영희(10)에게 30을 주면

철수는 80-30이 아닌 80을 삭제하고 50을 생성하고 영희는 30을 그냥 생성하는 교체를 사용합니다.

**(넘김)**

하지만 이더리움은 모든 계정의 잔액이 한번에 표현됩니다. 철수 (100), 영희 (10) 에서 다음 블록에서 철수가 영희에게 30을 주면 다음 블록 모든 계정 정보는 철수 (70) 영희 (40)이 되는 상태만 전이되는 현상을 상태 전이라고 합니다.

**(넘김)**

이진 머클트리는 기존 비트코인에서 머클루트를 도입할 때 사용하였던 개념입니다. 모든 데이터의 각 해시값을 저장하면 데이터가 많으니 모든 데이터를 이진 트리로 두어 각 데이터를 모아서 하나의 해시값으로 저장하는 것입니다.

물론, 하단에 있는 모든 데이터 값들을 저장하면 되는데 굳이 중간에 트리값들도 저장해야하기 때문에 데이터 구조의 비효율성처럼 보일 수 있습니다. 하지만 루트 해시값이 하나의 헤더에만 담기기 때문에 바디를 신경쓰지 않아도 될 때는 효율성이 증대됩니다.

**(넘김)**

상태전이와, 데이터값을 해시값으로 만드는 머클트리가 합쳐져서 상태전이 일반 머클트리가 됩니다. 비트코인은

해당 계정정보들을 아예 삭제한 후 생성하는 교체를 사용할 것입니다. 하지만, 이더리움에서는 상태전이를 할 수있으므로 변경된 값이 어떻게 변경되었는지 바꿔주는 상태전이를 사용할 수 있습니다.

**(넘김)**

패트리샤 트리: 피피티보고 설명

**(넘김)**

하지만, 이더리움은 조금 다르게 패트리샤 트리를 사용합니다.

리프노드는 키 값에 대한 패스가 종결되었을 때 value에 해당 주소에 대한 정보들을 포함한다. 현재 찾아보면A77d337 은 value로 1wei를 갖습니다.

브랜치 노드는 가지치기가 시작되는 노드입니다.

확장 노드는 공유되어질 키 값들이 저장합니다. 이 노드로는 종결되지 않으므로 확장노드의 value값은 이후의 키 값을 책임질 브랜치 노드를 갖게 됩니다.

이처럼 세 종류의 노드가 합쳐져서 확장 패트리샤 트리를 이루게 됩니다.

**(넘김) (머클 패트리샤 트리)**

지금까지 배웠던 개념들을 합쳐 머클 패트리샤 트리가 됩니다. 조금 다른점이 있다면 stateRoot 트리는 상태전이를 포함한 트리이며, 트랜잭션과 영수증 루트는 상태전이를 포함하지 않은 트리입니다.

**(넘김)**

블록 생성 때마다 이런식으로 별도의 프로그램이 UTXO를 찾는 비트코인과 다르게 계정의 상태 전이를 이더리움 자체에서 지원하는 점이 다릅니다.

또한, 영수증 루트에서 영수증은 거래의 유효성을 검증하기 위한 중요한 정보를 제공하며 이더리움 네트워크 참여자들은 영수증을 통해 거래가 올바르게 실행되고 결과가 예상한대로 상태에 반영되었는지 확인할 수 있습니다. 계정의 상태 변화 검증에도 이러한 영수증 루트를 사용할 수 있습니다.