<블록체인 기본 개념 #2>

1. 블록체인 정의
    A. P2p(peer to peer) 네트워크를 이용한 분산 데이터 베이스 기반의 저장 기술
    B. 거래내용이 분산되어 저장된다.
    C. 블록체인 네트워크에 연결된 여러 컴퓨터에 거래 정보가 담긴 원장데이터를 저장 및 보관하는 기술
2. 블록체인 거래 방식
    A. 블록체인에 참여한 모든 노드들이 거래한 사실을 증명함 -> 분산형태로 저장하기 때문에
    B. 블록체인 안에서 합의 알고리즘에 의해 중복사용(double spending)을 감시한다.
3. 블록체인의 역사
    A. 초기 블록체인 Digicash 설립 (1989) : 암호화, 개인키 및 공개키, 블라인드 서명 기술 적용한 전자결재
4. 블록체인 활용
    A. 비트코인
    B. 자산 기록
    C. 물류 유통
    D. 스마트 컨트랙트(도박, 중개거래, 이커머스, 물품거래)
    E. 전자투표,으료보험
5. 블록체인 기본 특징
    A. 분산원장 : 모든 노드들이 동일한 거래내용을 가지고 있어 과거 거래를 조작할 수 없다.
    B. 암호화 : 블록체인의 핵심 기능으로, 해시값을 통해 무결성을 보장한다.
        i. 블록이 해시체인으로 연결되어 있어 조작하지 못한다.
        ii. 거래에 있어서 전자서명 알고리즘을 통해 거래의 정당성을 보증한다.
    C. 합의 알고리즘
        i. 블록체인 네트워크에서 각 노드간 중요 정보 도달에 대한 시간차이가 있더라도, 새롭게 생성된 블록에 대한 정당성을 검토하여 동의를 얻기 위한 알고리즘
        ii. P2p 네트워크에서 정보지연 및 수신 문제점을 해결하기 위해 사용
        iii. 거래를 검증하기 위해, 블록체인 네트워크의 각 노드에서 만들 블록의 정당성을 검토
    D. 스마트 컨트랙트
        i. 여러 사람이 합의한 계약을 사람이 없어도 자동으로 실행
        ii. 다수의 노드가 내용을 공유하고 계약을 모두가 다 확인하고 가지고 있는 것, 무결성이 보장되서 스마트 계약이 가능하다.
6. 암호화폐에서 보장되야 할 것
    A. 계좌 및 신원 관리
        i. 비트코인 전송시 공개키에 대한 개인키가 필요
        ii. 공개키로 접근하고, 개인키로 제어함

B. 화폐지급

C. 신용관리

D. 신뢰성

7. 비트코인

A. pow라는 합의 알고리즘을 사용한다.

B. 합의 알고리즘 (다수의 참여자들이 통일된 의사결정을 위해 사용하는 알고리즘)

i. 이중지불 문제를 해결하기 위해 합의 알고리즘이 필요.

ii. 블록체인의 분산형 원장을 활용한 거래 이력 공유 및 검증을 수행

iii. 종류 : 1.작업증명(pow) , 2.지분증명(pos), 3.위임지분증명(대표선출방식), 등등

C. Pow (작업증명) = 채굴

i. 블록을 생성하는 것

ii. 먼저 문제를 풀어서 블록을 만들어야 그 내용이 전파되고, 확정된다.

iii. 문제를 푸는 것, 특정 타겟보다 작은 hash값을 구해야한다.

iv. 비트코인은 긴 블록을 선택하는 longest

<블록체인 기본 #3>

8. 블록체인 네트워크
    A. Trustless network : 신용이 필요없는 네트워크로, 시스템 안에서 신용을 만든다.
9. 해쉬함수
    A. 일방향 함수로, input data 를 x, hash function 을 h, output 을 z 라고 할 때,
    B. x 는 z 의 프리 이미지이다
    C. z 는 이미지 이다.
    D. 동일한 z 를 가지는 x 가 많이 존재할 수 있다.
10. 해시함수 조건
    A. 프리 이미지 조건 : output 으로 input data 를 찾는 것이 어려워야 한다.
    B. 제 2 프리이미지 저항성 : x 가 주워졌을 때, 해시값 z 를 가지는 또다른 값을 찾는 것이 어려워하 함
    C. 충돌 저항성 : z 가 주워졌을 때, x 와 x'의 값이 달라야 한다(찾기 어려워야함).
    D. 눈사태 효과 : 비슷한 값이더라도 해쉬값에는 차이가 많이 나는 것
11. 전자서명 (개인키로 data 를 암호화한 것)
    A. 인증 : 전자서명을 통해 서명자를 검증할 수 있다.
    B. 메시지 무결성 : 메시지는 서명 후 변경될 수 없음
    C. 부인인방지 : 발신자는 서명한 사실을 부인할 수 없음
    D. 거래할 사람에게 공개키를 보내고, 보낼 데이터를 2 개로 나눠 보낸다. 그냥 데이터와 개인키로 암호화(전자서명)한 데이터를 전송한다. 보낼 사람은 받은 공개키를 이용해서 암호화 한 데이터를 복호화 한다. 복호화한 데이터와 그냥 전송받은 데이터를 비교한다.
12. 대칭키
    A. 암호화와 복호화에 같은 키가 사용되는 것
    B. 비대칭키보다 속도가 빠르다. 하지만 배포 과정이 까다롭다.
13. Timestamp (시간 관리)
    A. 분산 네트워크 에서는 hash-chain 을 이용해 타임 스퀀스를 맞춘다.
    B. 절대시간가 분리를 시켜 거래 내용을 관리한다.
    C. 거래한 내용과 시간을 해쉬 함수에 넣어서 다음 거래의 해쉬함수 input 값으로 넣어서 연결
14. 거래 생성
    A. 누군가에게 보냈다는 것을 개인키로 암호화(전자서명) 한다. 서명한 값과 거래 내용을 네트워크로 보낸다.
15. 블록 생성
    A. 이전 block 의 내용을 해쉬 함수로 매핑하면 block ID 가 됨
    B. 이전 block ID 와 거래들의 내용을 묶어서 블록을 생성한다.
    C. 따라서 block 안에 존재하는 거래들 중, 하나만 바뀌어도 해쉬값이 바뀌기 때문에 다른 노드들이 틀린 것을 증명할 수 있다.
16. 블록체인 종류

A. 개방형 블록체인

i. 비트코인같이 네트워크로 누구나 참여할 수 있는 블록체인

B. 허가형 블록체인

i. 기업에서 사용하며, 권한을 부여받아야 사용할 수 있는 블록체인

17. 블록체인 포크

A. 합의 알고리즘을 사용하는 이유로, 악의적인 트랜잭션 발생 또는 데이터 동기화 이슈가 있을경우 포크가 발생함.

B. 일어나는 경우 : 유효하지 않은 블록(이중지불), 데이터 동기화(통신지연)

C. 종류 : 1.소프트포크(포크가 일어나도 이전 버전과 호환 가능) 2.(포크가 일어나면 이전과 호환불가)

<역사 #4> ~ <비트코인 #5-1>

18. 거래내역 = 트랜잭션

19. 노드종류

    A. 거래내용을 full blockchain 으로 보냄 -> full blockchain 에서 solo miner 로 보냄 -> 블록을 생성해서 full blockchain 에 보냄 -> 유효성 검사를 위해 SPV 로 보냄 -> 유효성검사 완료되면 full blockchain 에서 다른 노드로 내용을 전파

    B. Full Blockchain Node : 블록체인 데이터 전체를 관리하는 노드 (모든 노드가 모든 거래내용을 가지기엔 너무 크기 때문에 특정 노드가 관리함)

        i. 새로운 블록을 받으면, 유효성 검증 후, 기존 블록체인에 연결하고, 다른 노드들에게 전파함

    C. Lightweight wallet(SPV) : 블록의 헤더 정보만 가지고 있는 노드로, 트랜잭션의 유효성을 검증함

        i. 새로운 블록이 생성되면, Full Blockchain 으로부터 블록헤더를 받아 자신의 헤더 체인에 연결해서 유효성을 검증

    D. Solo Miner : 해쉬값을 구하는 것으로, 수학적 퍼즐을 풀고 먼저 해답을 제시

        i. 채굴자의 블록이 Full Blockchain Node 로 전송되며, 이 해답에 대해서 유효성 검사가 끝나면, 그 대가로 보상을 받는다.

    E. Reference Client(Bitcoin Core)

        i. 모든 기능을 다 활용하는 노드로써, wallet, miner, full blockchain, network 가 포함된 노드

    F. Pool protocol server

        i. 다른 노드들이 필요한 노드가 있으면 연결시켜주는 노드

    G. Mining Node

        i. 블록체인 없이 채굴만 담당(pool 에 들어가서 채굴만 함)

    H. Lightweight(spv) Strartum wallet

        i. Stratum 프로토콜을 이용해 네트워크 기능이 있는 노드와 연결하고 이웃 노드에 트랜잭션 전달

20. 머클 트리

    A. 모든 트랜잭션을 하나의 블록에 가지고 있는 것이 비효율 적이기 때문에 트랜잭션을 합쳐 머클루트를 만들고, 머클루트만 블록 생성에 저장함

    B. 블록내에서 다수의 트랜잭션들을 암호화 한다.

    C. 무결성 검증과 거래내용 확인을 위해 머클트리가 사용된다.

    D. 특정거래내용을 찾기위해서 Log2 만큼만 검색하면 확인가능

    E. 하나의 거래를 확인하기 위해서 머클경로(머클루트를 구하기 위해 필요한 HASH 값)만 있으면 알 수 있다.

    F. 장점 : 특정 거래에 대한 위변조 가능성과 빠른 탐색을 용이하게 함

    G. 이렇게 볼 수 있습니다. 여기에서 특정 거래 Tx1 의 거래를 검증하기 위해서는, 즉 최종 머클 루트를 구하기 위해서는 H2, H10, H14 값만 필요하게 됩니다. (기존의 Tx1 에 대한 H1 은 가지고 있을 테니깐요.)
    그렇다면, 이러한 H2, H10, H14 이 값들을 머클 루트를 구하기 위한 머클 경로 라고 부르고, 단순한 검증 노드인 SPV (Simplified payment verification) 노드는 위의 머클 경로를 풀 블록체인 네트워크 노드 한테 전달받게 됩니다.
    그리고, 자신이 가지고 있는 H1 과 전달받은 H2 를 계산하여, H9 를 얻고 이와 H10 을 계산하여 H13 을 얻고, H13 과 H14 를 이용하여 최종 머클루트 값을 얻게 되어, 현재 Tx1 이 현재 확인하고 싶은 블록에 있는지를 검증하게 됩니다.

    수업시간에도 설명을 했던것과 같이, SPV 는 경량화된 지갑을 소유하고 있는 녀석들이기 때문에, 모든 블록들을 가지고 있지도 않거니와 한 블록내의 모든 Transaction 들도 가지고 있지 않습니다.(한개 블록에 Transaction 의 수가 1000 개에 가까운 경우도 있기 때문이지요.)

그래서 특정 블록에서의 검증이 필요한 Transaction 에 대해서 풀 블록체인 네트워크 노드에게 검증을 위한 요청을 하게되고, 풀 블록체인 노드는 해당 정보를 확인하여 검증하는데 필요한 최소한의 해시 리스트인 머클 경로를 전달하게 되어, SPV 노드가 빠르고 간결하게 검증을 할 수 있도록 도와줍니다.

21. PoW (작업증명)

    A. 블록을 만드는 것,

    B. 이전 블록의 해쉬값보다 커야하기 때문에 , nonce 의 값을 1 씩 증가시키며 해쉬값을 찾는다.

    C. 블록을 만들고 full blockchain node 에 전송 -> 생성된 블록에 대한 유효성 검증 진행 -> 검증 끝나면 다른 노드로 해당 블록을 전파

<비트코인 -2 #5-2>

22. 블록구조
    A. Block = 헤더 + 트랜잭션 => 블록크기 : 최대 4MB
    B. 헤더 :
        i. Version : 노드들은 다 같은 버전을 가져야 한다.
        ii. previous block hash : 이전 블록 해쉬값
        iii. merkle root : 머클 트리의 루트 노드값
        iv. timestamp : 블록이 생성됐을때의 시간이 아닌, 유추할 정도로 이용
        v. difficulty tree : 결과를 얻는데 있어 현재의 난이도를 나타냄
        vi. nonce : 해쉬를 만들어서 이 해쉬값이 이전의 해쉬값보다 커질때까지 증가해 가면서 만든다.
23. 거래를 위한 요구 조건 => 거래증명에 필요한 것
    A. 소유권 증명
    B. 사용 가능한 자금
    C. 이중거래 유무
24. UTXO 모델
    A. 거래 기반 방식을 사용 -> 더 큰 돈을 주고 거스름돈을 받는 송금구조로, 내가 가진 자산을 쪼개서 보관하는 것
    B. 합쳐진 형태가 아닌 각각의 거래로 온 형태로 저장되어 있다.
    C. 다른 사용자에게 일정량의 암호화폐를 받을 때 생성된다.
    D. 받은 금액 그대로를 UTXO 로 저장한다. (ex -> 6BTC 를 1BTC, 2BTC, 3BTC 처럼 각각의 UTXO 로 저장)
    E. 일부 금액을 송금할 경우, 새로운 UTXO 를 생성하고 기존 UTXO 는 파기한다.
25. 트랜잭션 검증
    A. 트랜잭션은 입력부와 출력부로 나뉜다.
    B. 개인키로 거래에 대한 내용을 암호화 하면, 공개키로 복호화 할 수있다. = 전자서명
    C. 공개키 = 암호화폐의 접근 권한을 잠금 -> 보내기 위해 잠근다.
    D. 개인키 = 암호화폐 이용 권한을 얻음 -> 권한을 얻기위해 해제한다.
    E. 복합스크립트 = 잠금 스크립트(공개키, output) + 해제 스크립트(개인키, input)
    F. P2PKH (pay -to – pubkey – hash) :
        i. 다른 사람으로부터 받은 bitcoin 을 다른 거래에 이용할 경우 사용하는 것
        ii. 비트코인 내에서 가장 일반적인 스크립트 형식으로, 비트코인 프로토콜에 대한 지불 거래 유형이다.
        iii. 잠금스크립트로 거래를 잠구고, 해제 스크립트로 받은 거래에 대한 권한을 얻는다.
        iv. 스택을 이용한다.
26. 비트코인 주소는 개인키 -> 공개키 생성 -> 비트코인 주소 생성
27. Pow 하는 이유 : 블록만드려고,

A. 채굴수익 = 블록생성 보상 + 거래 수수료

28. Mining pool : 개별적으로 참여할수 있고, 소프트 업그래이드에 용이하다. 단점은 중앙화 되어 있어서 공격받을 수 있고, 관리자를 신뢰해야한다.

<wallet #6>

29. Wallet
    A. 비트코인 주소와 비밀키가 담겨있는 소프트웨어
    B. 종류 :
        i. 브레인 월렛 : 개인키를 나의 의지로 생각해낸 단어로 만드는 것
        ii. 베니티 월렛 : 특정 위치에 원하는 문자열이 나올때까지 해쉬함수를 돌리는 것
30. 지갑 유형
    A. 비결정적 방식 : 그냥 랜덤하게 생성하는 것
        i. 보완적으로는 좋지만, 잃어버리면 찾을 수 없음
    B. 결정적 방식 (type -1 형지갑): 브레인 월렛 ,베니티 월렛
        i. 시드값을 이용해 계속 해쉬함수 적용시키는 것
    C. 계층 구조의 결정적 방식(HD WALLETS)
        i. 상위 노드의 공개키로 하위 계층의 키와 주소를 생성
        ii. 하나로 하는 것보다 나눠서 작업하는 것이 더 안전함 (은행 여러 개에 자신 분배해서 저장)
        iii. Master key 를 이용해 공개키인 child key 를 생성
    D. 연상 기호 코드 워드
        i. 브레인 월렛과 비슷
        ii. 차이점은 내가 생각하는 단어가 아닌 연상기호 코드를 이용해 단어를 선택

<이더리움 #7>

31. 이더리움과 비트코인 차이점 -> #7 번 프린트 보기

	이더리움	비트코인
사용 목적	스마트 계약 블록체인	결제용 블록체인
화폐	Ether 사용	Bitcoin 사용
블록체인 모형	계정 기반 (외부 계정, 계약 계정)	UTXO 모형
채굴 알고리즘	현재는 Ethash Proof-of-Work (PoW)로 블록 생성 (향후 Proof-of-Stake (PoS)으로 변경 예정)	SHA-256-기반 Proof-of-Work
블록 생성 시간	3 ~ 30 sec	~ 10 min
블록 크기	30KB (2KB/s)	1MB (1.67KB/s)

32.

33.

A. 계정기반 vs 거래기반(UTXO) 작업방식

34. 이더리움 블록체인의 모든 기능은 항상 EOA 가 날린 트랜잭션으로부터 시작된다.

35. 먼저 "소유계정"은 오직 거래만을 담당하고, 거래에 대한 기록은 "계약계정"에 저장된다.

A. 이때 계약계정은 소유 계정에서 요구하는 조건을 수행할 수 있다. 예를 들어 "이더의 가격이 100 만원이 되면 B 는 C 에게 이더를 보내라 " 라는 조건이 있으면, 먼저 소유계정에서 보낸 이더가 pow 를 통해 블록으로 생성되고, B 에게 이더를 보낸다. 그리고 조건이 만족되면 B 는 C 의 외부계정으로 이더를 보내게 된다.

36. 계정 구조

A. "계약관련 계정(CA,조건으로 거래내용이 저장되는 곳)"과 "소유금액 계정(EOA,실제 돈 거래가 이루어지는 계정)"으로 나눠진다.

B. 소유계정 (EOA) or 외부계정

i. 개인키를 사용해 거래를 생성하고 서명함

ii. Ca 를 활성화 하기 위해서 eth 가 필요

iii. 이더리움 사용자를 위한 계정

iv. 주소와 연결하여 잔액정보를 기록

v. 개인키로 제어되는 것으로서, 코드를 저장할 수 없다.

vi. EOA 에서의 value 는 ether 를 뜻함

C. 계약 계정 (CA)

i. 계약서들을 저장하는 계정으로, 자신 스스로는 활성화 될 수 없다.

ii. 계약 증명이 있는 계정으로, 프로그램의 일종이다.

iii. 코드정보 및 잔액 정보를 기록

iv. 스마트 컨트랙트 코드에 의해 제어되며, 특정 CA 코드를 저장할 수 있다.

D. EOA -> EOA : 금액이 왔다갔다 할 때 사용

E. EOA -> CA : 계약관련으로 일을 수행

37. 상태 트리

A. 계정 정보, 이더 잔액 정보를 저장하는 곳

B. 상태 트리는 블록 외부에 보관하고, 각 블록에는 상태 트리의 루트 노드 값만 저장한다.

C. Nonce : 이중거래를 방지(얼마의 거래가 발생한지 알 수 있는 것) -> 동일한 값이 나오면 동일지불인 것이다. (중요함)

D. 블록 외부에 보관하고, 블록에는 상태트리의 루트 노드값만 저장

38. 이더리움의 스마트 계약

A. CA 가 EOA 로부터 명령을 받으면 EVM 에 의해 자동으로 수행되는 것을 "스마트 계약 " 이라고 한다.

B. 자율 에이전트와 같다 (자동적으로 기능을 수행하는 것)

C. 스마트 컨트랙트를 활성화 하기 위해서는 EOA 가 필요하다.

D. EVM 에 의해 스마트 컨트랙트(자율 에이전트) 가 실행된다.

E. 스마트 계약의 목적

i. 데이터 저장 및 유지 (CA 의 일종으로, 계약 내용 저장)

39. EVM (이더리움 버츄얼 머신)

A. 연산을 수행하는 곳

B. 블록들의 검증 절차로 실행되는 것(마이닝이 EVM 을 실행)

C. 마이닝이 EVM 코드를 실행하고, 경쟁적으로 블록을 생성

D. 생성된 이더리움 노드들은 생성된 블록들의 검증 절차의 일부로써 EVM 을 실행

E. EVM 이 contract code 를 EVM Bytecode 로 바꿔줌

F. EVM 은 코드기 때문에 무한루프가 생길수 있는데, 이때 gas 라는 개념을 통해 문제를 해결한다.

G. Solidity 특징 -> 튜링 완전 : 무한 반복이 가능한 것 -> 이것을 해결하기 위해 gas 를 사용

40. 이더리움 화폐 단위

A. 10^9 wei = 1gwei

B. 1Eth = 10^{18} wei

C. Gas 수수료 설정

i. 최대값 = gas price * gas limit

ii. Ex) 10^6gwei = 10^6 * 10^9 wei = 10^{15} wei = 10^{-3} Eth

41. 수수료 gas

A. 코드 명령어는 gas 를 필요로 한다. 이때 gas 부족하면 상태 변경이 취소되고, 사용된 gas 도 돌려받지 못한다.

B. 수수료 gas 는 채굴자의 이더리움 주소로 들어간다.

42. 이더리움은 Ghost 프로토콜을 사용한다.

A. 블록 생성자 삼촌 블록(전파시간보다 블록생성 시간이 더 빠를 경우 발생)에게도 보상을 준다. (fork 일어난 노드에 대해서도 보상을 지급)

43. POS(지분증명)

A. Eth 라는 금액에 대해 기간과 양에 따라 블록 생성 권한을 주는 것

B. 많은 Eth 를 가질수록 더 많은 블록을 생성할 수 있다. -> 중앙화가 될 수 있다.