2017/03/20, 2017 동계 세미나-블록체인

A Next-Generation Smart Contract and Decentralized Application Platform (White paper)

Ethereum, 2015

이부형 (boohyung@pel.smuc.ac.kr)

상명대학교 프로토콜공학연구실

Contents

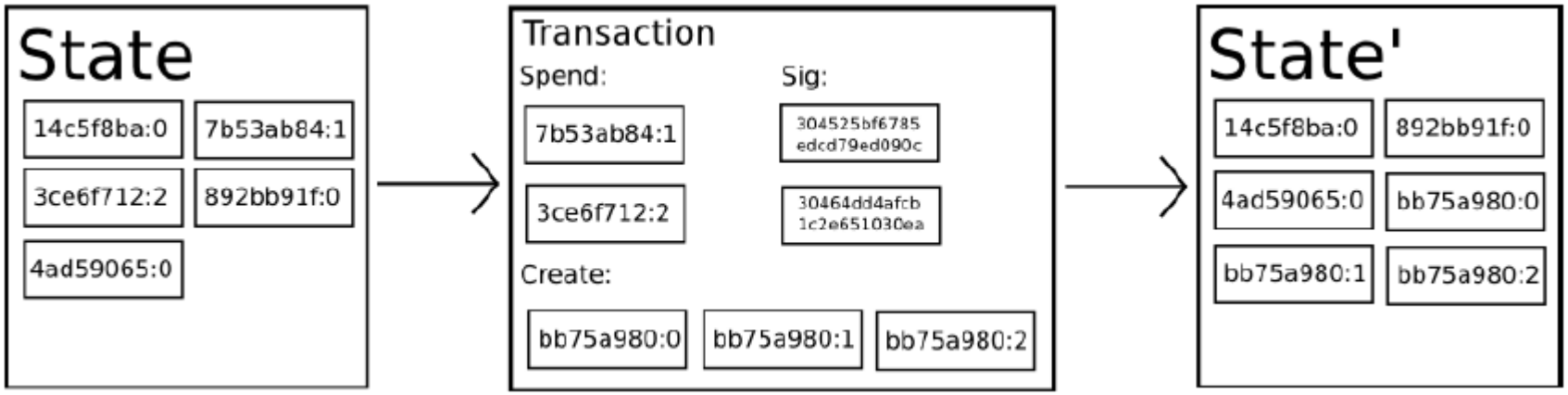
- Introduction to Bitcoin
- Ethereum

Introduction to Bitcoin (1/6)

- History
  - 분산 디지털 통화의 발전 과정
    - “익명의 e-cash 프로토콜”, 1980~90년대: 최초의 분산 디지털 통화 개념
      - 중앙집권적인 중개인에 의존한 거래 방식을 가짐
    - Wei Dai, “b-money”, 1998: 분산합의와 퍼즐을 해결하는 방식을 통한 화폐 발행 최초 제안
      - 실제 구현 방법을 제시하지 못함
    - Hall Finney, “Reusable proofs of work”, 2005: b-money + hashcash
      - 외부의 신뢰를 필요로 하는 컴퓨팅이 기반이 됨 (trusted computing)
    - Satoshi, “Bitcoin”, 2009: 실제적으로 구현된 최초의 탈중앙화 디지털 화폐
      - 공개키 암호화 방식 + Proof of Work

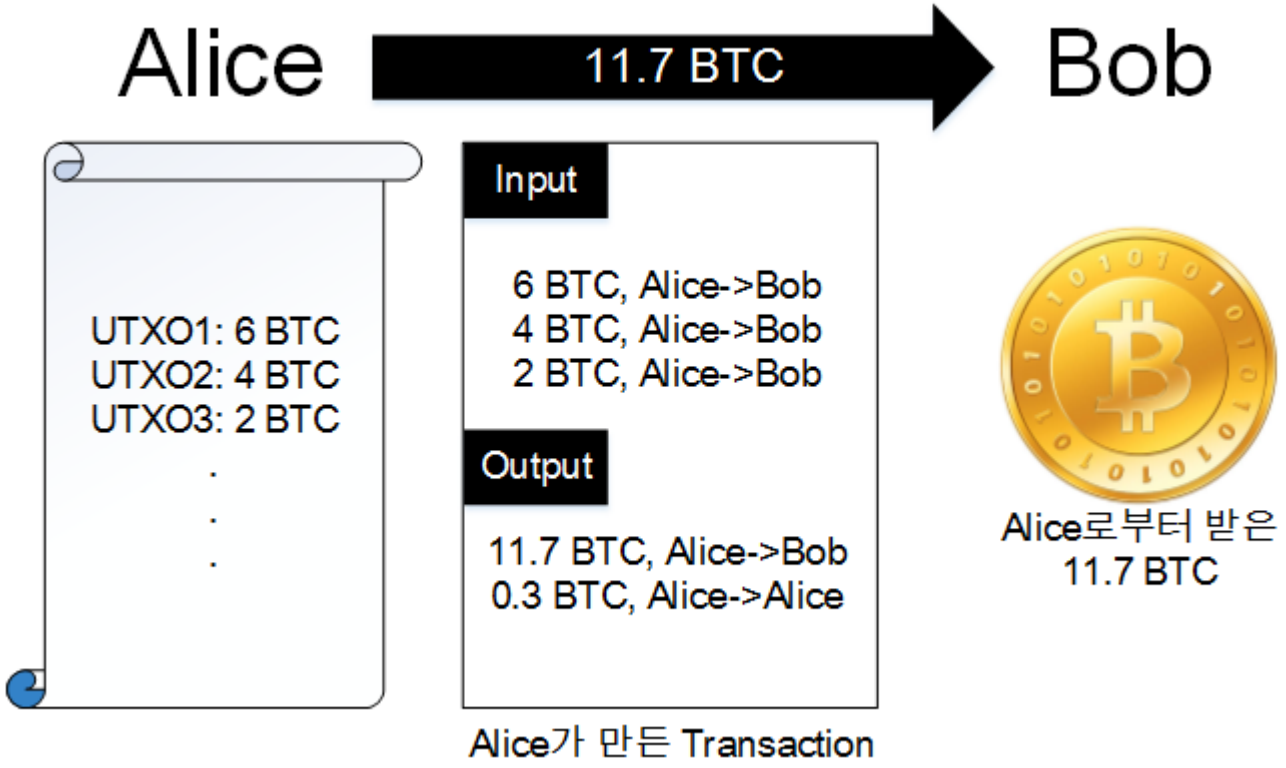
Introduction to Bitcoin (2/6)

- Bitcoin As A State Transition System
  - 상태(State)와 상태변환함수(State transition function)로 구성된 시스템
    - 상태: UTXO (Unspent Transaction Outputs)의 집합
      - UTXO: 금액과 소유자의 비트코인 주소
    - 상태변환함수: APPLY(S, TX) -> S' or ERROR
      - TX (트랜잭션): 입력 + 출력
        - 입력: 발신자의 UTXO 참조 정보, 발신자의 서명
        - 출력: 새로운 상태에 추가될 UTXO 정보



Introduction to Bitcoin (3/6)

- Bitcoin As A State Transition System
  - 예. “Alice [11.7BTC] -> Bob”



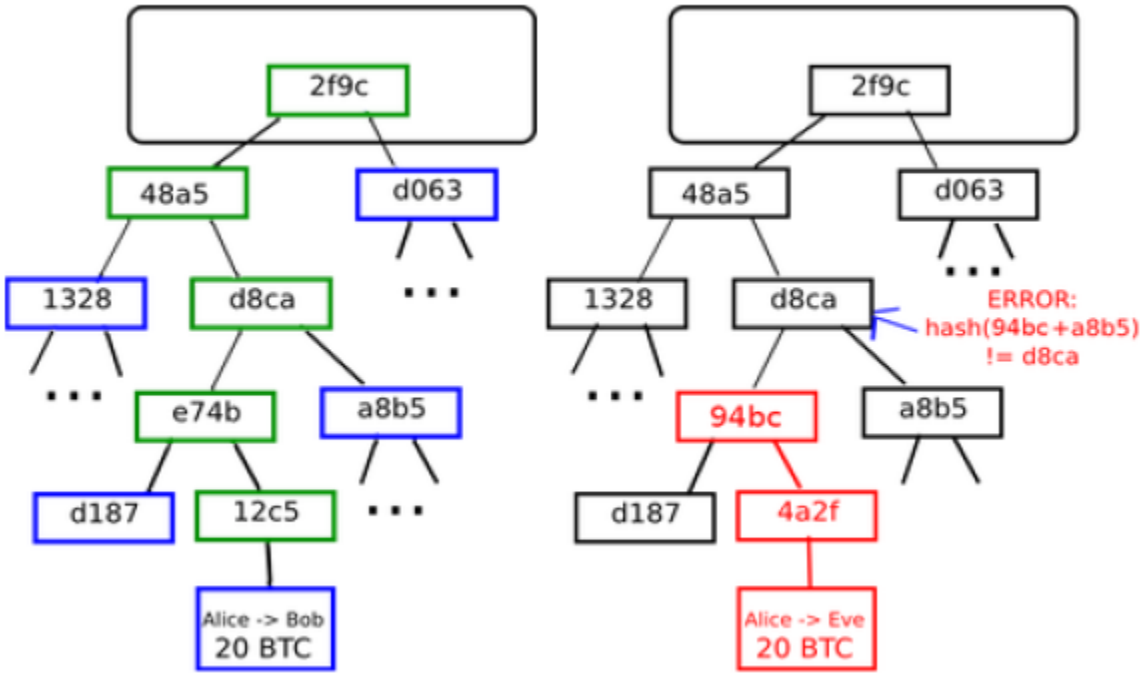
  - “Alice [0.3 BTC] -> Alice”는 거스름돈을 표시하는 Output

Introduction to Bitcoin (4/6)

- Mining
  - Proof of Work을 통한 블록 생성
    - 목적: 블록 생성을 계산적으로 어렵게 만들어서 공격자들이 마음대로 블록체인을 조작하는 것을 방지
  - 마이너는 블록이 생성되는 시점에 채굴 수수료를 획득
    - 생성한 블록에 포함된 트랜잭션 중 출력금액보다 입력금액이 큰 트랜잭션이 있다면 그 차액을 '거래 수수료'로 획득
      - 수수료는 사용자가 자유롭게 정할 수 있으나, 수수료를 0으로 설정하면 마이너가 블록에 포함시키기를 선호하지 않아 확정되는데 보통 1-7일이 걸림 (권장 수수료: 0.0005 BTC, 약 500원)
  - 6-confirmation
    - 예. 20000번째 블록에서 fork를 일으켜도 20005번째 블록이 생성되었을 때 거래를 확정하면 이중 지불 공격 방어가능
      - 만약, 공격자 노드가 51% Attack이 가능할 정도의 hashrate를 가졌다면 6-confirmation도 보안책이 될 수 없음

Introduction to Bitcoin (5/6)

- Merkle Tree
  - 블록헤더의 Merkle root를 통해 변조된 트랜잭션이 있는지 확인 가능



Introduction to Bitcoin (6/6)

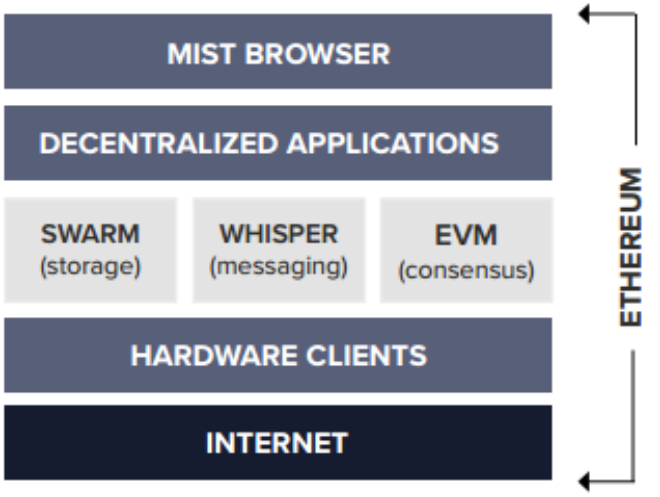
- Merkle Tree
  - SPV (Simplified Payment Verification)
    - Full node: 비트코인 네트워크에서 각 블록의 모든 정보를 저장하고 처리하는 노드
      - 전체 블록체인을 저장: 많은 디스크 공간을 필요로 함
      - 트랜잭션의 유효성을 검증 (Confirmation)
    - Light node (Thin client): 원하는 블록의 블록헤더를 다운로드하여 작업증명 여부를 검증하는 노드
      - 전체 블록체인을 저장하지 않고, 블록의 헤더만을 저장하여 Merkle root를 통한 특정 트랜잭션의 유효성 확인
      - 평소에는 트랜잭션의 유효성을 검증하지 않음
      - 헤더를 저장하기 때문에 특정 트랜잭션의 검증은 가능
  - 강한 안전성을 보장하면서도, 임의의 트랜잭션에 대한 상태와 잔고를 알 수 있음

Ethereum (1/13)

- Introduction
  - 2015년 7월 30일, 비탈릭 뷰테린(Vitalik Buterin)이 개발
  - 블록체인 기반의 분산 어플리케이션 개발 플랫폼
  - API 형태로 다양한 어플리케이션의 구현을 지원
    - https://github.com/ethereum/
    - C++, Java, Python, GO 등
    - 스마트 컨트랙트, 소유권에 대한 임의의 규칙, 트랜잭션 형식, 상태 변환 함수 등을 생성할 수 있음
  - 현재 이더리움 백서는 누구나 수정할 수 있도록 되어 있음
    - https://github.com/ethereum/wiki/wiki/White-Paper

Ethereum (2/13)

- Structure
  - Mist Browser
  - Decentralized Applications (DApps)
  - Swarm/Whisper/EVM
    - EVM (Ethereum Virtual Machine)
      - 스마트 컨트랙트 구현을 위해 4가지 언어를 사용
      - 상위 언어인 Serpent와 Solidity로 만들어진 코드가 컴파일되어 생성되는 바이트코드를 실행하기 위한 런타임 환경
  - Hardware Clients



Ethereum (3/13)

- Advantages (versus Bitcoin)
  - 튜링완전성 (Turing-Completeness)
    - 비트코인: Script를 사용하기 때문에 if 문만을 지원
    - 이더리움: 튜링완전언어를 지원하여 if 문 뿐만 아니라 반복/조건 명령문도 지원
      - 이더리움이 지원하는 튜링완전언어: Serpent, Solidity, LLL, Mutan
  - 개발 플랫폼을 이용한 응용성
    - 비트코인: 전자화폐 서비스
    - 이더리움: 서비스를 창조할 수 있는 거대한 플랫폼
  - 스마트 컨트랙트
    - 블록체인에 실행가능한 코드 형태 업로드, 특정 조건이 달성되면 자동적으로 코드가 실행되어 계약이 이행됨
    - 자기 강제적 언어 (Self-Enforcing Language)

Ethereum (4/13)

- Terms
  - account (어카운트): 사용자의 계좌 역할
  - ether (이더): 이더리움에서 통용되는 화폐의 단위
  - gas (가스): 트랜잭션 내 코드 실행에 드는 비용
    - 명령어의 종류에 따른 비용이 산정되어 있음
      - Gas cost (v1.0)
        - https://docs.google.com/spreadsheets/d/1m89CVujrQe5LAFJ8-YAUCcNK950dUzMQPMJBxRtGCqs/edit#gid=0
    - 같은 코드를 무한히 실행할 수 없도록 함
  - contract (컨트랙트): 트랜잭션이 도착하면 실행되는 특정한 코드
    - 코드 실행을 통해 새로운 컨트랙트를 생성할 수도 있음

Ethereum (5/13)

- Account
  - 어카운트의 필드 구성
    - nonce: 트랜잭션 카운터
    - 현재 어카운트에 남아있는 이더 잔고
    - 어카운트의 컨트랙트
    - 어카운트의 크기
  - 어카운트의 종류
    - 외부 소유 어카운트 (EOA, Externally Owned Account): 트랜잭션을 만들어서 서명하는 역할
      - 비트코인의 지갑과 같은 개념
    - 컨트랙트 어카운트: 메시지를 받을 때마다 자신의 컨트랙트를 활성화시키는 역할
      - 코드를 포함하거나 내부에서 코드를 실행
      - 데이터를 저장

Ethereum (6/13)

- Transaction
  - 상대방에게 보낼 메시지를 가지고 있는 서명된 데이터
  - 트랜잭션의 필드 구성
    - 수신자의 주소
    - 발신자의 서명
    - 발신자가 수신자에게 보내는 이더의 양
    - STARTGAS: 최대 트랜잭션 수행 횟수
    - GASPRICE: 매 계산마다 발신자가 지불하는 비용
  - 컨트랙트는 트랜잭션을 구성하는 필드의 데이터를 읽어서 어플리케이션에 맞는 코드를 실행

Ethereum (7/13)

- Transaction
  - STARTGAS와 GASPRICE
    - 2016년 9월, 이더리움 네트워크에 지속적인 DoS 공격 사례가 발생
      - 서로 다른 계좌에 다수의 빈 트랜잭션을 발생시킴
      - 빈 어카운트를 무한정 생성하여 메모리 소비
    - DoS 공격을 막기 위해 이더리움에서 제시한 방안
      - 발신자가 소비하는 모든 리소스에 비례하여 강제로 수수료를 지불
      - 코드 내의 우연적이거나 악의적인 무한루프, 계산 낭비를 방지하는 역할을 수행

Ethereum (8/13)

- Message
  - 컨트랙트에 의해 생성됨
    - CALL opcode
  - 트랜잭션과 다르게 따로 저장될 필요가 없음
  - 메시지의 필드 구성
    - 발신자 주소
    - 수신자 주소
    - 메시지와 함께 전달되는 이더
    - STARTGAS
    - 선택적 데이터 필드

Ethereum (9/13)

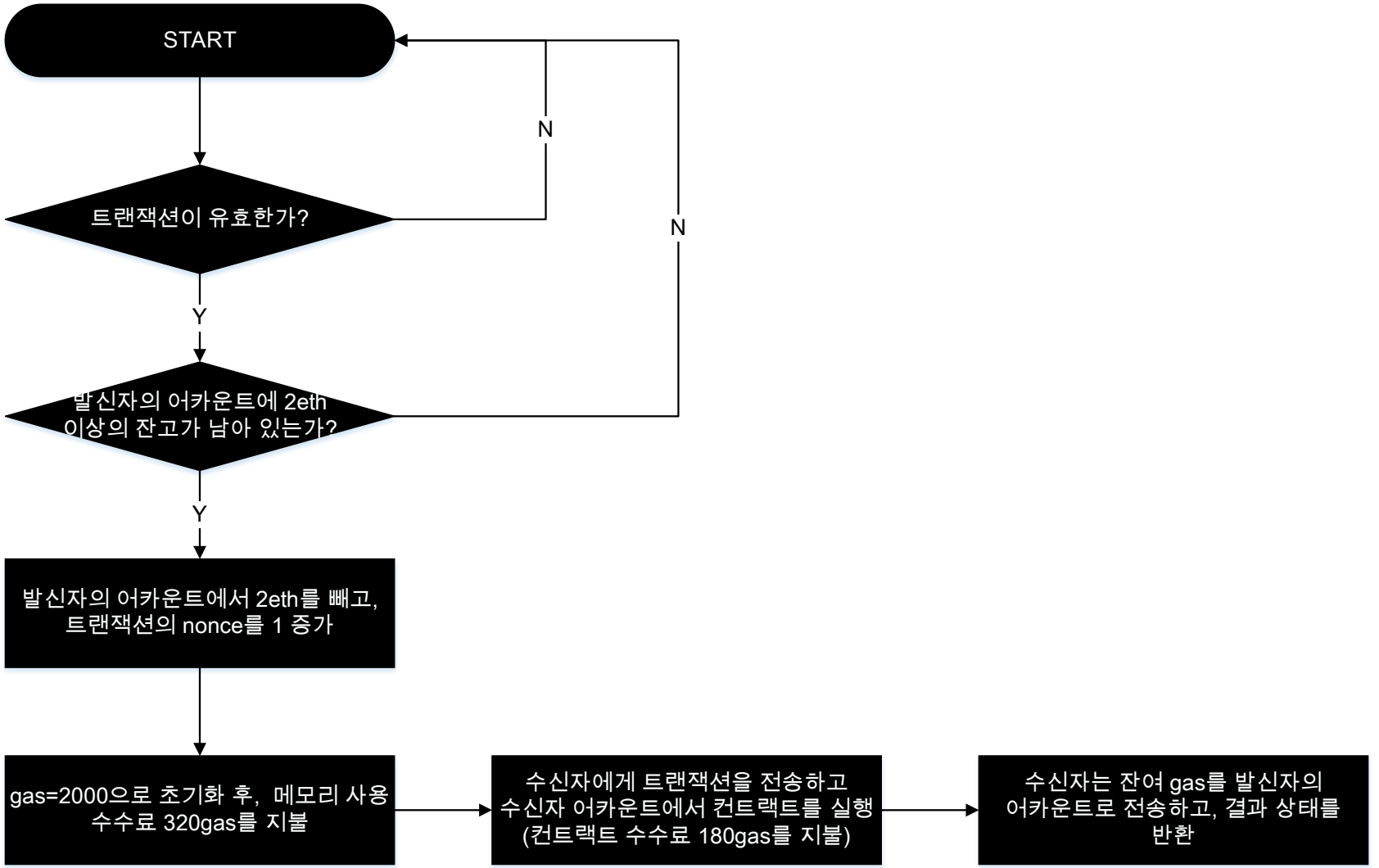
- State Transition System
  - APPLY(S, TX) -> S’
    - 기본사항 확인
      - 트랜잭션 형식, 서명의 유효성 여부
    - 발신자의 어카운트 잔고에서 수수료를 빼고 nonce를 1 증가
      - 트랜잭션 수수료 = STARTGAS * GASPRICE
    - gas = STARTGAS로 초기화 후, 트랜잭션에서 사용된 메모리에 대한 수수료를 지불
      - 1 바이트 당 5 gas
    - 수신자에게 트랜잭션을 전송
      - 만약, 수신자 어카운트가 컨트랙트라면 코드를 끝까지 실행
    - 결과 상태 반환
      - 잔여 gas를 이더로 환전하여 발신자에게 전송

Ethereum (10/13)

- State Transition System
  - 예제
    - 조건 1: 트랜잭션에 2,000gas (STARTGAS), 0.001eth GASPRICE, 64 바이트의 데이터를 사용
    - 조건 2: 발신자가 수신자 어카운트에서 컨트랙트를 실행할 때 180gas가 소요됨

Ethereum (11/13)

- State Transition System
  - 예제



Ethereum (12/13)

- Mining
  - Ethash: 수정된 작업증명 (PoW) 방법
    - 약 12초에 하나의 블록이 생성될 수 있도록 알고리즘이 설계되어 있음
    - 2차원 배열 데이터인 DAG 파일이 사용됨
  - 고속으로 생성되는 블록으로 인해 생기는 엉클 블록이 이더리움의 보안성을 저하시키는 문제가 발생
    - 엉클 블록: 블록생성에 성공하였고 검증에 오류가 없어서 네트워크를 통해 전파되었으나, 더 빨리 전파된 다른 채굴자들에 의해 순위가 밀려 주체인에 들어가지 못한 블록
      - 비트코인에서는 Stale block으로 표현함
    - 수정된 GHOST 프로토콜을 도입
      - 7세대까지의 엉클 블록을 주 체인에 포함
      - 엉클 블록 중 유효한 블록에 수수료를 지급 (기본 보상의 93.75%)

Ethereum (13/13)

- DApps
  - 금융 DApps: 자산을 블록체인 위에 올리고 스마트 컨트랙트의 대상으로 사용
    - 채권, 주식, 파생상품, 보험, 복권, 도박 등
  - 준/비금융 DApps: 토큰, 쿠폰, 네임코인, 투표 등
  - 탈중앙화 조직/회사: 회사나 조직을 블록체인 상에 올려서 운영
    - 월급지급, 금전거래, 회계장부기록, 지분표시 등
  - 탈중앙화 자율조직/회사 (DAO/DAC): 블록체인 상의 알고리즘으로 의사를 결정하여 영업, 회계, 구매, 판매 및 수익 분배 등을 실현

감사합니다!

이부형 (boohyung@pel.smuc.ac.kr)

보충 1: Blockchain Scalability Issues

- 블록체인의 확장성에 영향을 주는 파라미터
  - Block interval: 하나의 블록이 전파되는데 걸리는 시간
  - Block size: 블록의 크기
    - 트랜잭션의 개수에 따라 변할 수 있음
    - 한 개의 트랜잭션은 보통 500 bytes
- Case: Bitcoin
  - Block interval: 10 min (average)
  - Block size: 1MB (maximum)
  - 3.5 tps (TPS: Transactions per second)
    - 시스템이 초당 처리할 수 있는 트랜잭션의 양
  - block interval이 적고, block size가 클수록 확장성은 커짐

보충 2: Transaction fee

- 마이너가 블록을 생성하면 받을 수 있는 수수료
  - Block fee (Mining fee): 현재 12.5 BTC
  - Transaction fee: 생성한 블록 안에 포함되는 모든 트랜잭션들이 가지는 수수료의 합

- 트랜잭션 만으로 채울 수 있는 최대 크기는 0.75 MB (= 750,000 bytes)
  - 마이너는 하나의 블록에 최대 1500개의 트랜잭션을 넣어서 블록을 만들 수 있음
    - 한 개의 트랜잭션 크기는 평균 500 bytes
  - transaction pool에서 수수료가 높은 순/우선 순위가 높은 순으로 트랜잭션을 축적

보충 3: gas (ethereum)

- gas: 이더리움에서 코드 내 연산의 기본 단위
  - 트랜잭션이나 컨트랙트 실행에 사용
    - 보통, 하나의 연산에 1 gas가 소요됨
    - 특정한 연산, 상태의 일부분으로 저장되어야 하는 데이터의 양이 많을 경우에는 더 많은 gas가 소요됨
  - 이더리움에서 사용하는 화폐 단위 Ether로 코드 실행을 위한 gas를 구입하는 개념