하이퍼레저 패브릭 거래 처리

박승철교수

하이퍼레저 패브릭 거래

❖ 거래

- 블록체인에 스마트 계약 프로그램인 체인코드(chaincode)를 설치하고, 기존에 설치된 체인코드를 실행하기 위해 호출하는 동작(operation)
- 전체 네트워크의 피어들에게 전달되어 블록체인에 기록되기 전에 먼저 보증 피어들에 의해 보증
- 보증 피어에 의해 보증된 거래들만 확정(commit)되어 블록체인에 기록 가능

하이퍼레저 패브릭 거래

❖ 거래 유형

- 배치 거래(deploy transaction)
- 호출 거래(invoke transaction)

하이퍼레저 패브릭 거래

❖ 배치 거래(deploy transaction)

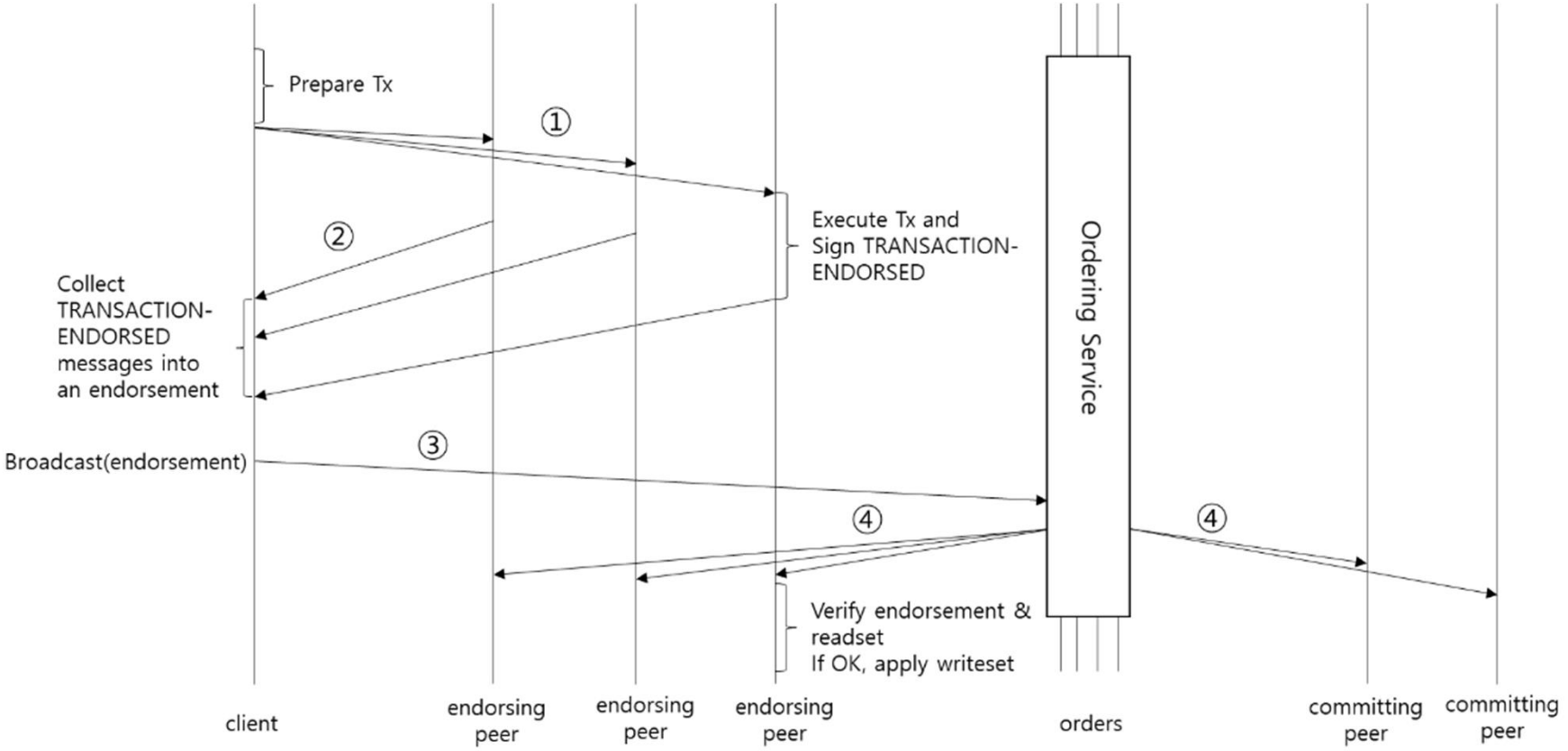
- 체인코드를 생성하여 블록체인에 설치하는 거래
- 체인코드 프로그램을 매개변수로 수신
- 체인코드는 해당 체인코드를 실행하기 위한 조건을 설정하는 보증 정책(endorsing policy) 지정을 포함
- 배치 거래가 성공적으로 실행되면 설치된 체인코드의 ID(chaincodeID) 반환

하이퍼레저 패브릭 거래

❖ 호출 거래(invoke transaction)

- 블록체인상의 체인코드의 특정 동작(함수)을 호출하여 실행
- 특정 체인코드와 해당 체인코드의 특정 함수를 참조
- 지정된 함수가 성공적으로 실행되면 블록체인의 상태를 갱신할 수 있는 결과 반환
- 블록체인 상태의 실제 갱신은 순서화 서비스를 거쳐 검증 및 확정 단계에서 수행

거래 처리 과정



① <PROPOSE, clientID, chaincodeID, txPayload, timestamp, clientSig>
② <TRANSACTION-ENDORSED, tid, epID, chaincodeID, txContent, readset, writeset, epSig>
③ broadcast(endorsement)
④ deliver(seqno, prevhash, blob)

거래 처리 과정

❖ 처리 단계

- 거래 보증(transaction endorsing)
- 거래 순서화(transaction ordering)
- 거래 확정(transaction committing)의 단계

거래 처리 과정

❖ PROPOSE 메시지 :

- <PROPOSE, clientID, chaincodeID, txPayload, timestamp, clientSig>
  - clientID : 거래를 제출하는 클라이언트의 ID
  - chaincodeID : 거래가 적용되는 체인코드의 ID
  - txPayload : 거래 수행에 필요한 정보. 호출 거래의 경우 체인코드의 함수와 매개변수, 그리고 호출 거래를 설명하는 메타데이터(metadata)가 포함되고, 배치 거래의 경우 체인코드의 소스 코드(source code), 체인코드와 응용을 설명하는 메타데이터, 그리고 해당 체인코드의 보증 정책
  - timestamp : 새로운 거래가 생성될 때마다 증가하는 정수 값, 클라이언트에 의해 유지됨.
  - clientSig : 거래를 제출하는 클라이언트의 서명

거래 처리 과정

❖ TRANSACTION-ENDORSED 메시지

- <TRANSACTION-ENDORSED, tid, epID, chaincodeID, txContent, readset, writeset, epSig>
  - tid : 거래의 ID
  - epID : 보증 피어의 ID
  - txContent : 거래의 페이로드 정보
  - readset : 거래에 의해 호출된 체인코드가 읽는 키(key)와 키 버전(key version) 쌍의 집합. readset은 거래가 실행되기 전에 생성
  - writeset : 거래 실행 과정에서 값이 변경되는 키(key)와 값(value) 쌍의 집합
  - epSig : 보증 피어의 서명

거래 보증 정책

- 전형적인 거래 보증 정책
  - 보증 노드의 서명을 사용하여 작성
  - 보증 피어 집합 E = {Alice, Bob, Charlie, Dave, Eve, Frank, George}를 지정하는 것으로 가정
- 보증 정책 예
  - E의 모든 보증 피어들로부터 동일한 내용에 대한 적합한 서명을 수신하여야 한다.
  - E에 속한 임의의 보증 피어로부터 적합한 서명을 수신하여야 한다.

거래 보증 정책

❖ 보증 정책 예

- (Alice OR Bob) AND (any two of (Charlie, Dave, Eve, Frank, George))의 조건을 충족시키는 보증 피어로부터 동일한 내용에 대한 적합한 서명을 수신하여 한다.
- 전체 7개의 보증 피어들 중에 임의의 5개의 보증 피어들로부터 동일한 내용에 대한 적합한 서명을 받아야 한다.

거래 보증 정책

❖ 보증 정책 설정

- 체인코드에 대해 설정하되 보증 정책은 시스템에 의해 미리 작성
- 체인코드 개발자가 특정 체인코드를 선택하고 매개변수 설정을 통해 보증 정책을 설정

거래 확정과 블록체인 갱신

- 현재 블록체인의 전체 상태**(world state)**
  - (k1, 1, v1), (k2, 1, v2), (k3, 1, v3), (k4, 1, v4), (k5, 1, v5), (k6, 1, v6)
- 현재 블록의 거래 순서
  - T1 → Write(k1, v11), Write(k2, v22)
  - T2 → Read(k1), Write(k3, v33)
  - T3 → Write(k2, v222)
  - T4 → Write(k2, v2222), read(k2)
  - T5 → Write(k6, v66), read(k5)

거래 확정과 블록체인 갱신

❖ 거래 확정 및 갱신

① T1의 reaset = { }, writeset = {(k1, v11), (k2, v22)}이다. readset이 비어 있으므로 읽기 의존성은 자동으로 충족된다. 따라서 T1 거래의 결과 블록체인의 전체 상태는 (k1, 2, v11), (k2, 2, v22)로 갱신된다. 갱신과정에서 writeset의 값으로 상태가 수정되고 대응되는 키의 버전 번호가 증가된다.

② T2의 readset은 {(k1, 1)}이다. 그런데 현재 블록체인의 전체 상태의 k1의 버전은 거래 T1에 의해 이미 2로 갱신되어 있기 때문에 읽기 의존성 검증에 실패하고, 따라서 T2 거래는 부적합 거래로 처리되어 블록체인 갱신이 적용되지 않는다.

③ T3의 reaset = { }, writeset = {(k2, v222)}이다. readset이 비어 있으므로 읽기 의존성은 자동으로 충족된다. 블록체인이 전체 상태는 (k2, 3, v222)로 갱신된다.

거래 확정과 블록체인 갱신

❖ 거래 확정 및 갱신

④ T4의 readset은 {(k2, 1)}이다. 그런데 현재 블록체인의 전체 상태의 k1의 버전은 거래 T1과 T3에 의해 이미 3으로 갱신되어 있기 때문에 읽기 의존성 검증에 실패하고, 따라서 T4 거래의 부적합 거래로 처리되어 블록체인 갱신이 적용되지 않는다.

⑤ T5의 reaset = {(k5, 1)}, writeset = {(k6, v66)}이다. 키 k5의 값은 이전의 거래 T1~T4에 의해 갱신되지 않았으므로 버전이 1로 유지가 되고 있고, 읽기 의존성이 충족된다. 블록체인이 전체 상태는 (k6, 2, v66)로 갱신된다.

합의 프로토콜

❖ 장착형 합의 프로토콜

- 응용의 요구사항에 따라 적합한 합의 프로토콜을 선택하여 사용
- SOLO – 하나의 중앙 집중형 순서화 노드를 통해 거래의 순서를 결정
- Kafka - f개의 노드가 고장(crash)이 나서 동작을 멈추더라도 분산된 순서화 서비스를 제공할 수 있는 합의 프로토콜. (2f+1)개 이상의 노드를 사용
- PBFT(Practical Byzantine Fault Tolerant) – f개의 노드가 고장으로 인한 멈춤을 포함한 임의의 오동작(Byzantine fault)에도 분산된 순서화 서비스 제공. (3f+1)개 이상의 노드 사용