2019/04/11 @ 상명대

Stella Consensus Protocol 분석

발표자: 임연주(yeonjoo@pel.smuc.ac.kr)

상명대학교 프로토콜공학연구실

목 차

- 개요
- 관련 연구(Related work)
  - FBA(Federated Byzantine agreement)
  - 쿼럼 슬라이스(Quorum slice)
- SCP(Stella Consensus Protocol)
  - Consensus protocol

개요

- 개요
  - 2015년 David Mazières가 FBA(Federated Byzantine Agreement) 프로토콜 기반 합의 알고리즘인 SCP 공개[1]
  - 현재 Stellar lumens(암호화폐)의 합의 알고리즘으로 사용
    - stellard에서 stellar-core으로 합의 알고리즘 변경
      - stellard: 리플의 합의 알고리즘 기반
      - stellar-core: SCP
  - 분산된 인터넷 인프라 구축을 위한 표준화 재정 중
    - 분산된 환경을 위해 SCP에 대한 연구가 진행되고 있음[2]
      - IRTF DINRG(Internet Research Task Force Distributed Internet iNfrastructure Research Group)

개요

- 특징

1. Decentralized control
   - 모든 노드가 합의 과정 참여하며, 합의를 제어하는 중앙기관 없이 동작함

2. Flexible trust
   - 사용자의 신뢰 노드(쿼럼 슬라이스)를 선택할 자유를 제공

3. Low latency
   - 빠른 시간(3~5초) 내에 합의 도달

4. Asymptotic Security
   - 컴퓨팅 파워를 통해 보안성을 제공하지 않음
   - 디지털 서명, 해시함수를 통해 안전성 제공

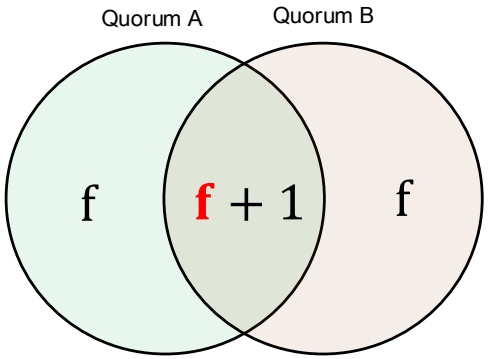
관련 연구(Related work)

- FBA(Federated Byzantine Agreement)
  - 기존 BA에서는 N개의 노드(최소 합의를 위한 노드의 수)에 의해 폐쇄적인 합의를 거침
    - $N = 3f + 1$
  - 개념
    - 리소스 기반의 합의 참여 조건이 없이 모든 노드가 합의 과정에 참여
      - 기존 PoW의 경우 해시 연산, PoS의 경우 담보금 등의 리소스를 기반으로 합의에 참여할 수 있었음
    - 각 노드는 네트워크 참여 시, 신뢰할 노드를 선택
      - 신뢰 노드 선택에 따라 쿼럼 슬라이스 형성

관련 연구(Related work)

- FBA(Federated Byzantine Agreement)
  - N 개의 노드 간의 합의
    - 두 개의 쿼럼이 있다면, 반드시 $f+1$의 교집합을 가져야 함
      - 예시
        - f = 비잔틴 행위를 하는 노드 (ill-behaved node)
        - $N-f$ = 올바른 노드 (well-behaved node)

$$N = 3f + 1$$

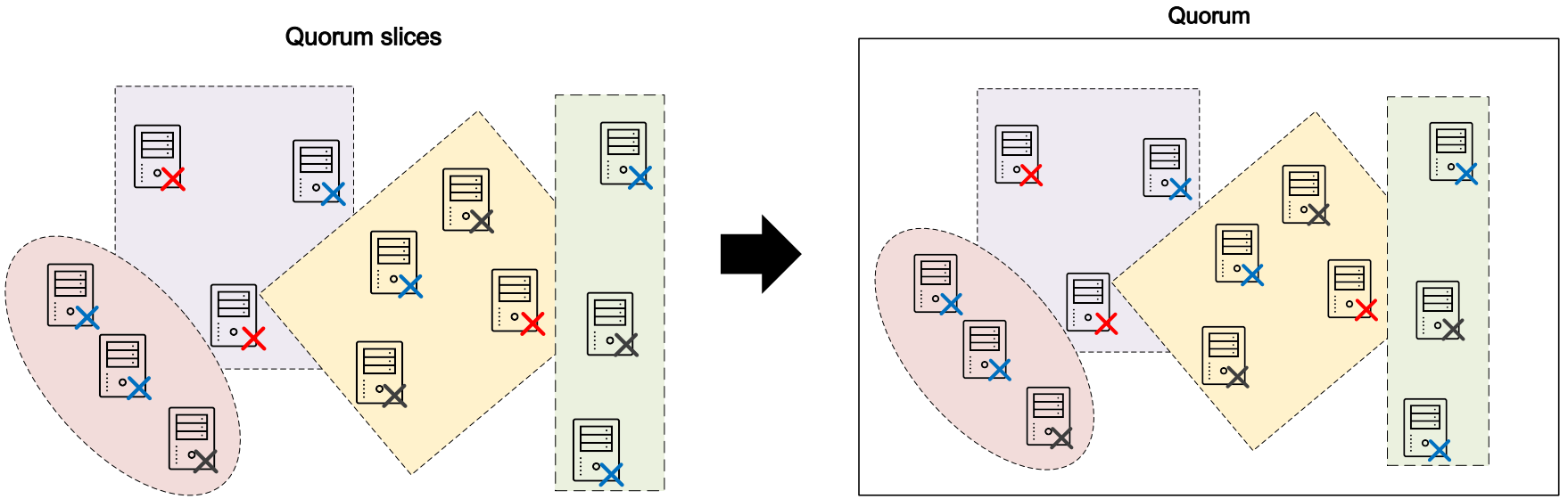


- **f** : ill-behaved node

    - 두 쿼럼 사이의 모순된 결론을 내리지 않을 수 있음

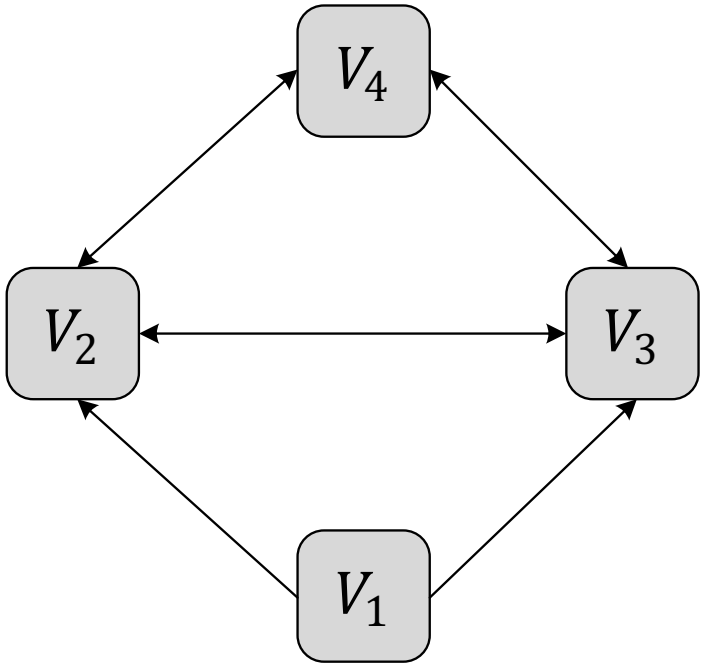
관련 연구(Related work)

- 쿼럼 슬라이스(Quorum slice)와 쿼럼(Quorum)
  - 쿼럼 슬라이스
    - 최소의 쿼럼 멤버십
    - 각 노드가 신뢰하는 피어를 선정해 쿼럼 슬라이스를 설정하여 합의를 진행
  - 쿼럼
    - 합의에 도달하는데 충분한 노드 집합



관련 연구(Related work)

- 쿼럼 슬라이스(Quorum slice)
  - 쿼럼 슬라이스의 다이어그램
    - v_i = 노드
    - $Q(v_i)$ = 노드 v_i 의 쿼럼 슬라이스
    - $\longrightarrow$ = 쿼럼 슬라이스 종속성(신뢰 노드)

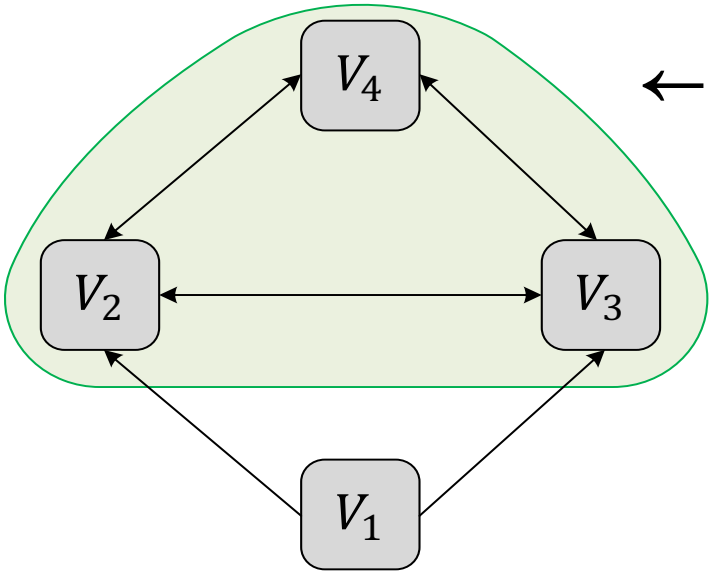


$$Q(v_1) = \{\{v_1, v_2, v_3\}\}$$

$$Q(v_2) = Q(v_3) = Q(v_4) = \{\{v_2, v_3, v_4\}\}$$

관련 연구(Related work)

- 쿼럼 슬라이스(Quorum slice)



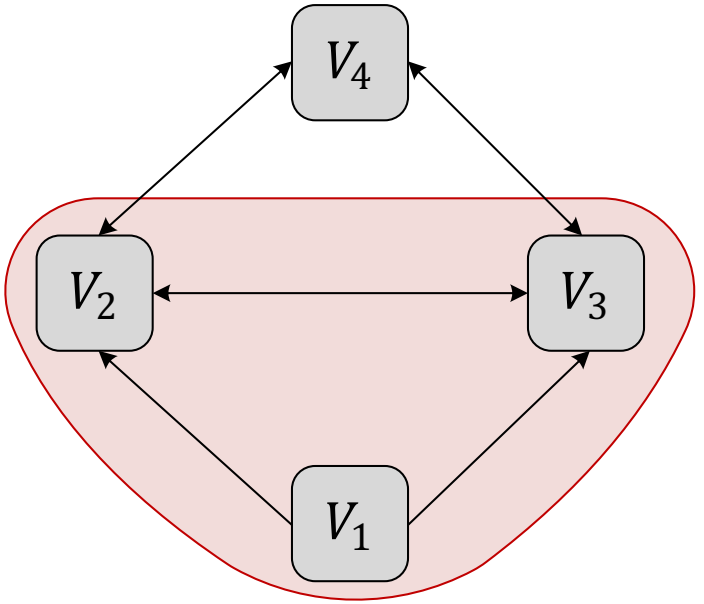
← 노드 v_2, v_3, v_4 의 쿼럼, 각 노드들의 쿼럼 슬라이스

$Q(v_1) = \{\{v_1, v_2, v_3\}\}$

$Q(v_2) = Q(v_3) = Q(v_4) =$

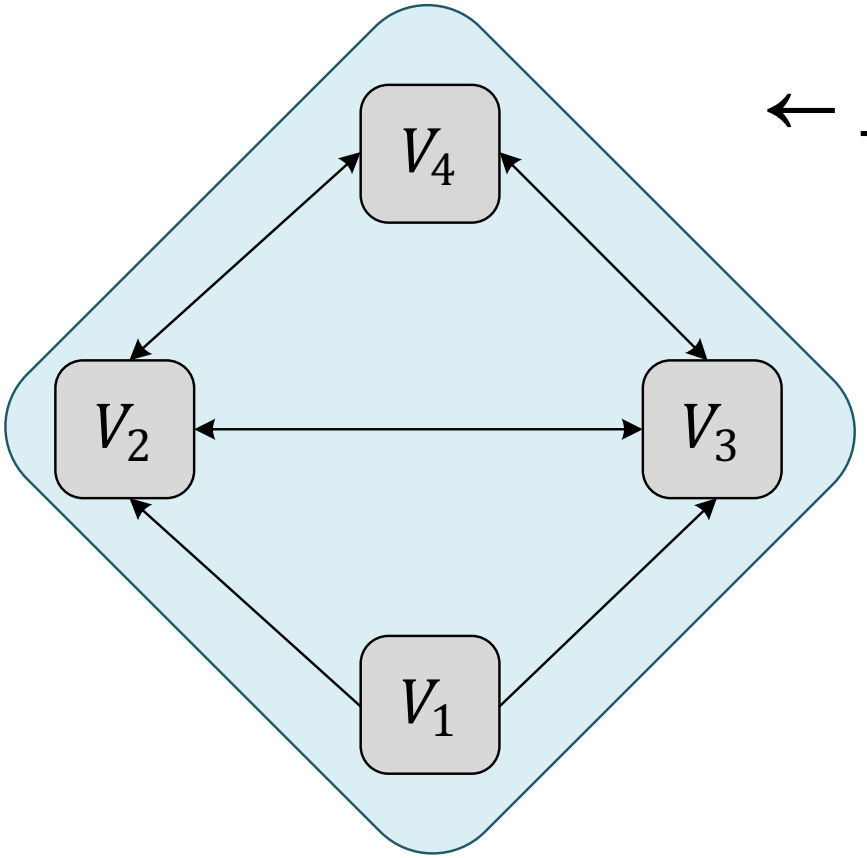
$\{\{v_2, v_3, v_4\}\}$

노드 v_1 의 쿼럼 슬라이스 ,쿼럼은 아님 →



관련 연구(Related work)

- 쿼럼 슬라이스(Quorum slice)



← 노드 v_1을 포함하는 최소의 쿼럼

$$Q(v_1) = \{\{v_1, v_2, v_3\}\}$$

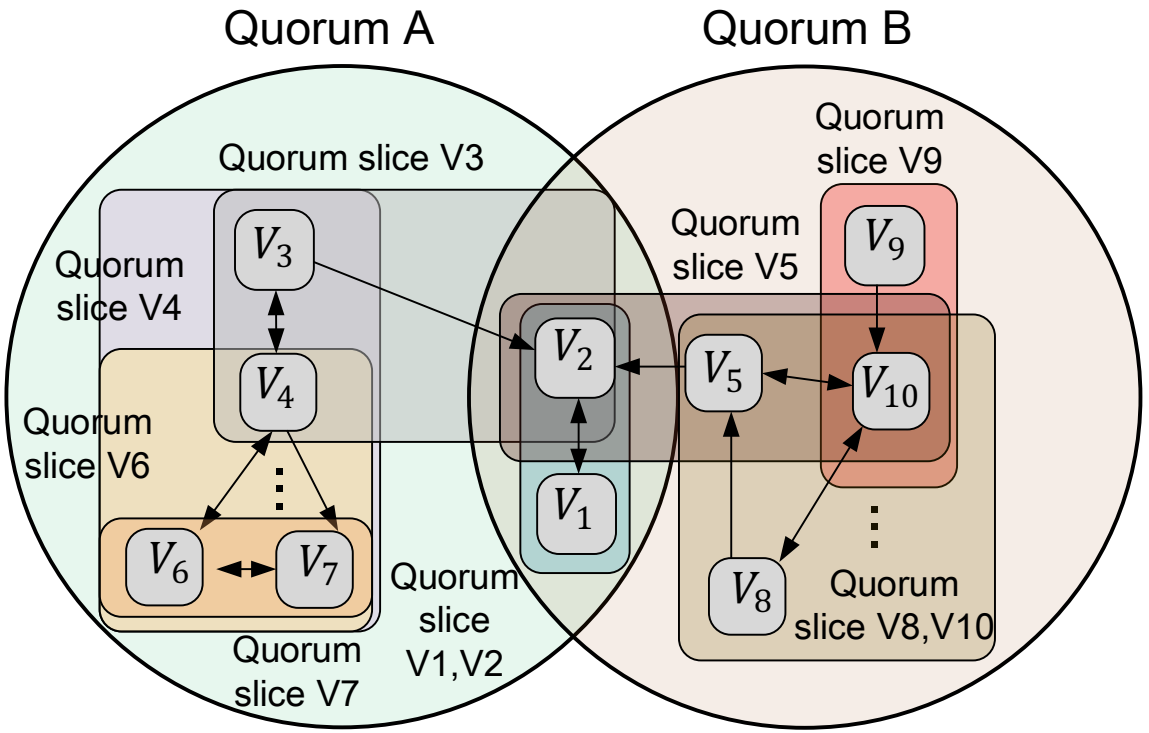
$$Q(v_2) = Q(v_3) = Q(v_4) = \{\{v_2, v_3, v_4\}\}$$

SCP(Stellar Consensus Protocol)

- 쿼럼 교차(Quorum Intersection)
  - 개념
    - 쿼럼들 중 어떤 두 쿼럼이 노드 하나를 공유하는 필요충분조건이 충족되는 경우를 쿼럼 교차라고 함
  - 쿼럼 간의 교차되는 부분을 통해 안정성 제공

SCP(Stellar Consensus Protocol)

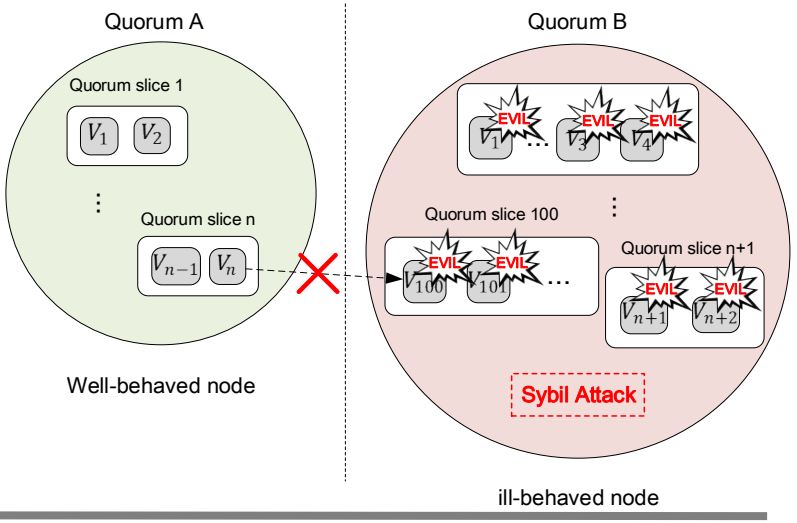
- 쿼럼 교차(Quorum Intersection)
  - 예제 그림



주체	신뢰 노드
V_1	V_2
V_2	V_1
V_3	V_4, V_2
V_4	V_3, V_6, V_7
V_5	V_2, V_{10}
V_6	V_7, V_4
V_7	V_6
V_8	V_5, V_{10}
V_9	V_{10}
V_{10}	V_5, V_8

SCP(Stellar Consensus Protocol)

- 쿼럼 교차(Quorum Intersection)
  - Sybil 공격
    - 합의 과정에서 발생하는 메시지는 디지털 서명을 하기 때문에 위조가 불가능함
    - 올바른 노드들은 Sybil 공격으로 생성된 ill-behaved 노드(악의적)들을 자신의 쿼럼 슬라이스로 선택하지 않음
      - 가짜 노드들은 쿼럼 A에 영향을 미치지 않음



SCP(Stellar Consensus Protocol)

- Consensus protocol
  - Nomination
    - 후보자 등록
      - Federated voting을 통해 후보 등록
  - Balloting
    - 당선자 투표
    - 3 단계로 진행됨
      1. Prepare
      2. Commit
      3. Externalize

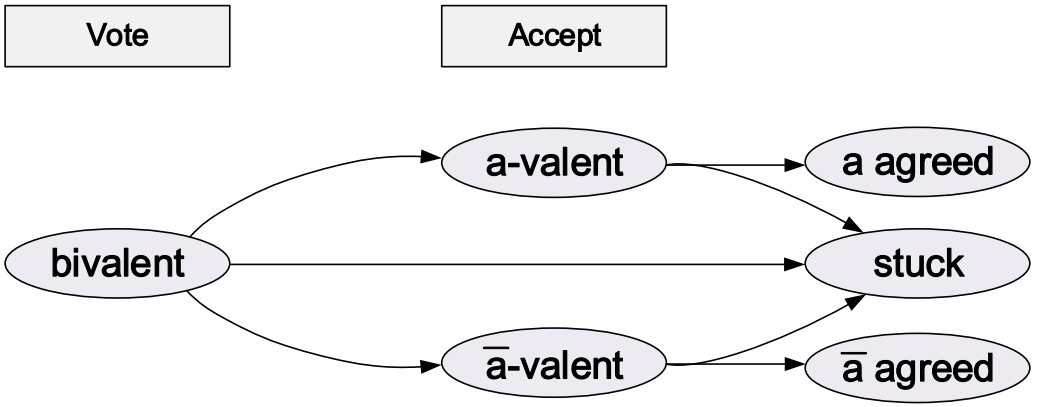
SCP(Stellar Consensus Protocol)

- Consensus protocol
  - 메인 서브루틴
    - Federated voting(연합투표)
      - 상태 변환을 위한 투표를 의미
      - 각 단계 마다 Federated voting을 수행함

      - 임계 값(Threshold)
        - Quorum threshold
          - 쿼럼 모든 노드가 메시지를 보낸 경우
        - Blocking threshold
          - 쿼럼 슬라이스 내 한 노드가 accept 메시지를 보낸 경우

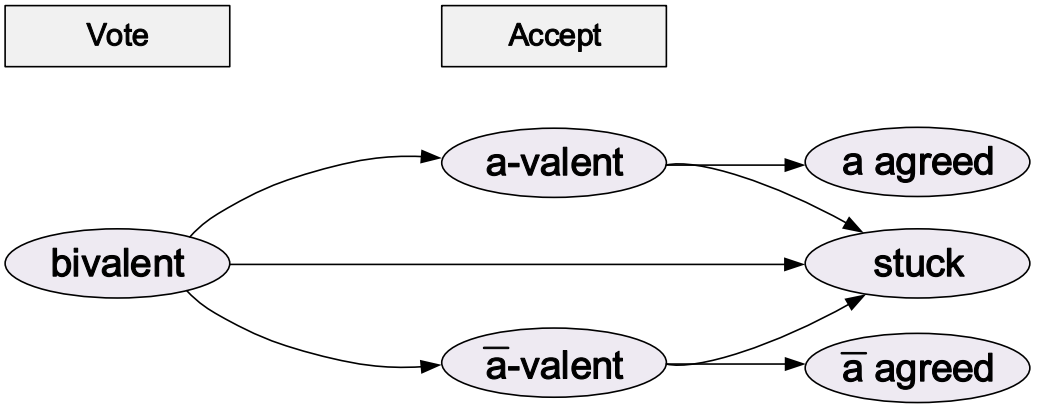
SCP(Stellar Consensus Protocol)

- Federated Voting(연합 투표)
  - 결과(Outcome)
    - bivalent
      - 이분법적인 투표 진행
    - valent
      - 노드가 a/!a를 승인(accept)하면 a-valent/!a-valent
        - 모순되는 선택을 할 수 없음



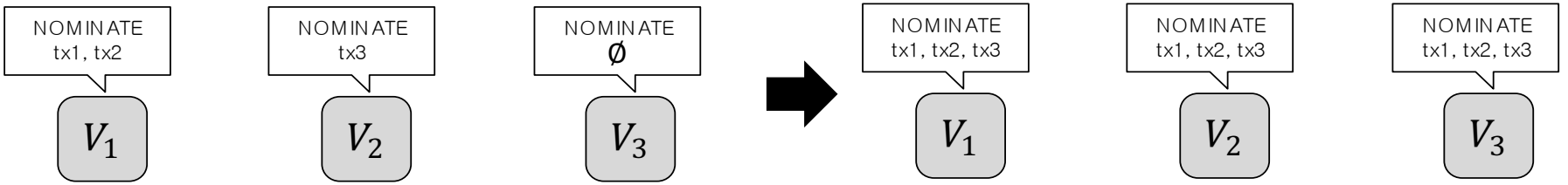
SCP(Stellar Consensus Protocol)

- Federated Voting(연합 투표)
  - 결과(Outcome)
    - agreed
      - accept 메시지에 동의함
    - stuck
      - a와 !a의 수가 같을 경우
      - 도중에 또는 타임 아웃 시, 고착상태가 될 수 있음



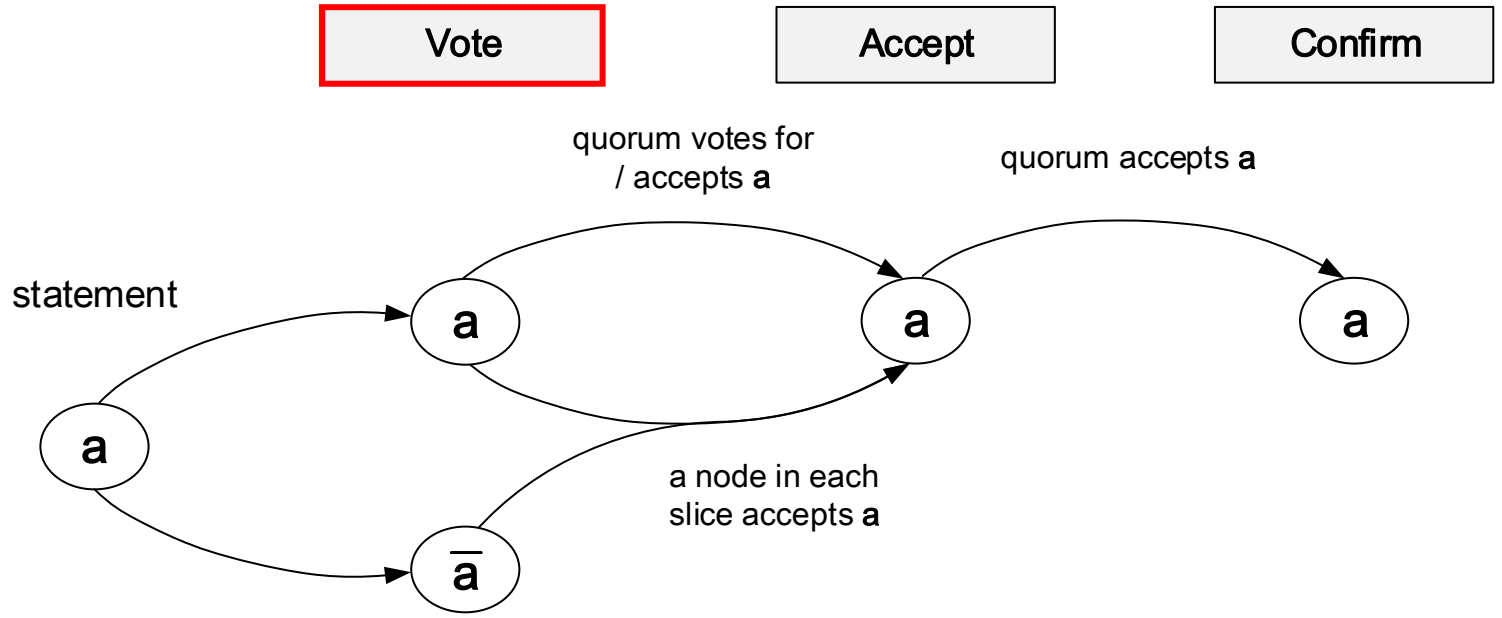
SCP(Stellar Consensus Protocol)

- Nomination Mechanism
  - 3 단계를 통해 후보군 정의
    1. Federated voting을 통해 후보자(candidate values) 등록
    2. NOMINATE x
       - x : 유효한 합의 후보의 집합(a set of valid candidate consensus value)
         - NOMINATE 문을 확인하면 새로운 후보자 등록을 중지
    3. 모든 노드가 동일한 후보자를 공유
       - $x = tx_i$
         - e.g., tx1, tx2, tx3



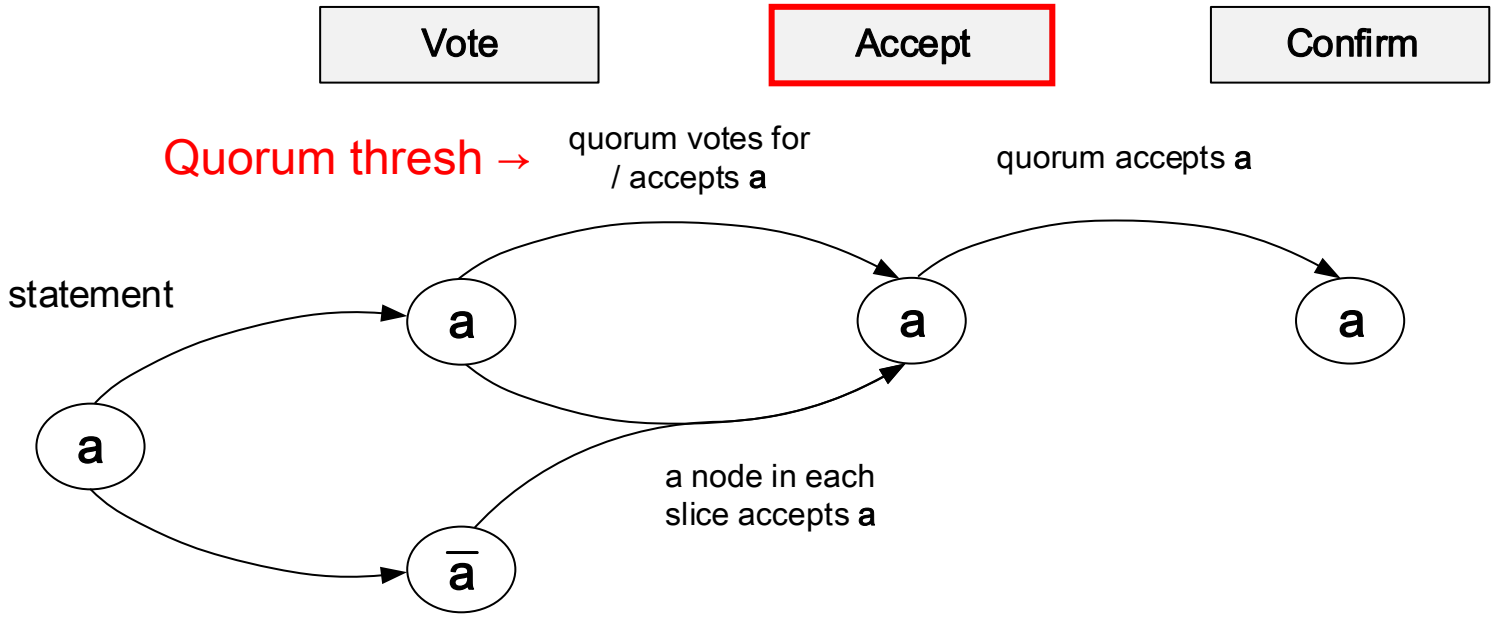
SCP(Stellar Consensus Protocol)

- Nomination
  - Federated Voting 과정
    1. Vote
       - 유효한 후보 값(statement)에 대해 투표를 진행
         - 모순된 투표를 하면 안됨
           - 하지만 !a를 투표한 뒤, a를 accept 할 수 있음



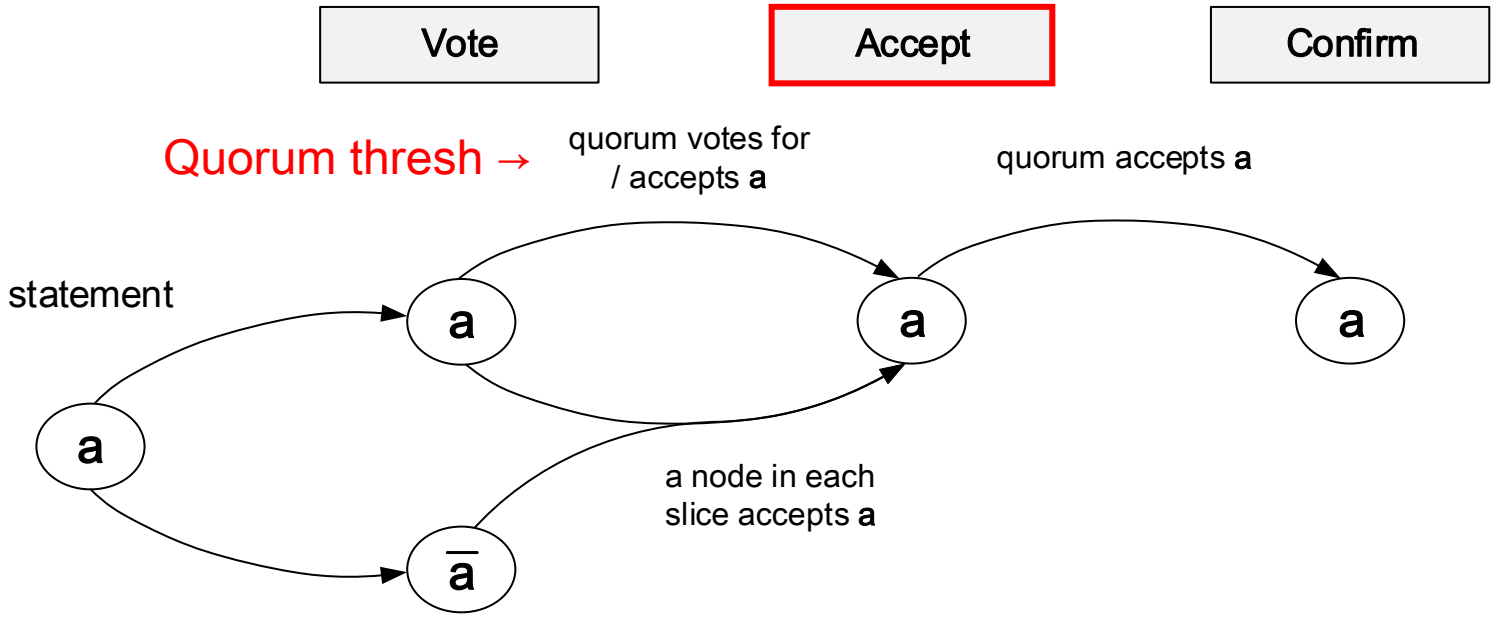
SCP(Stellar Consensus Protocol)

- Nomination
  - Federated Voting 과정
    2. Accept
       - 첫 번째 투표가 성공했다는 것에 대한 두번째 투표
         - accept(a)



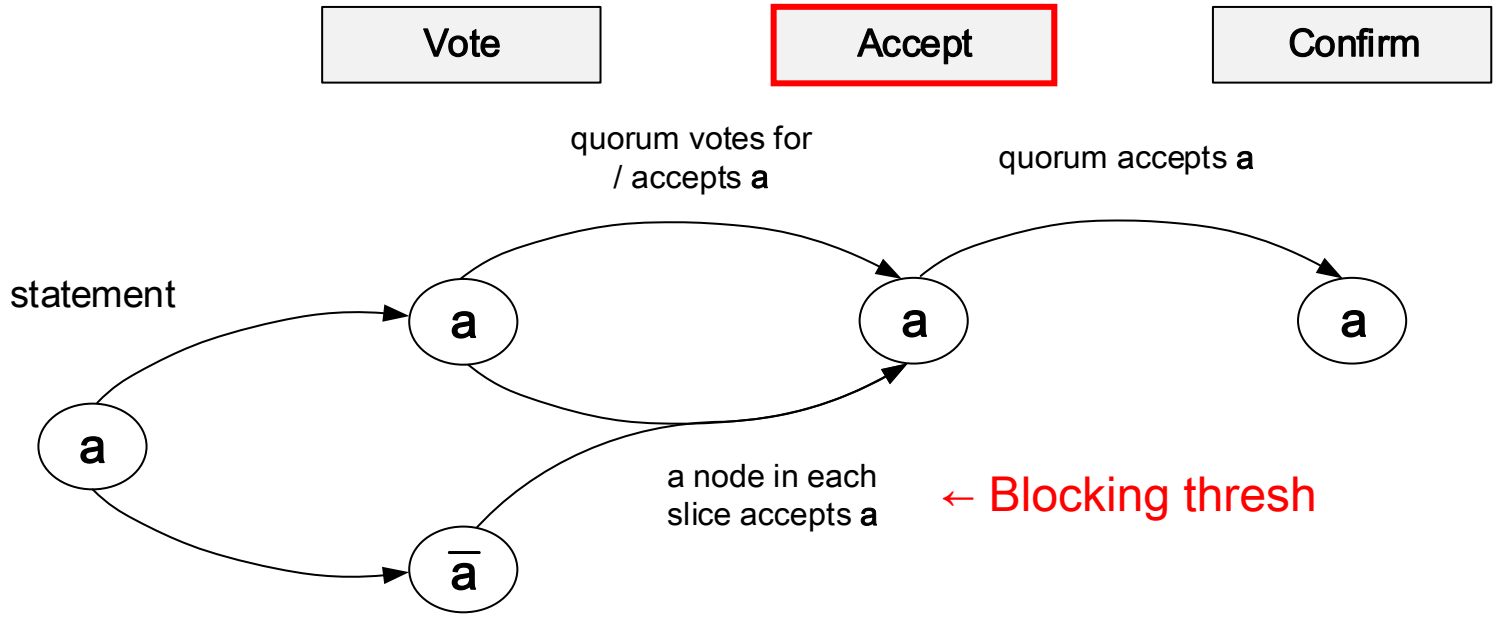
SCP(Stellar Consensus Protocol)

- Nomination
  - Federated Voting 과정
    2. Accept
       1) Quorum threshold
          - 노드 v가 속한 쿼럼 모두가 투표를 완료한 경우, Accept 단계로 넘어감



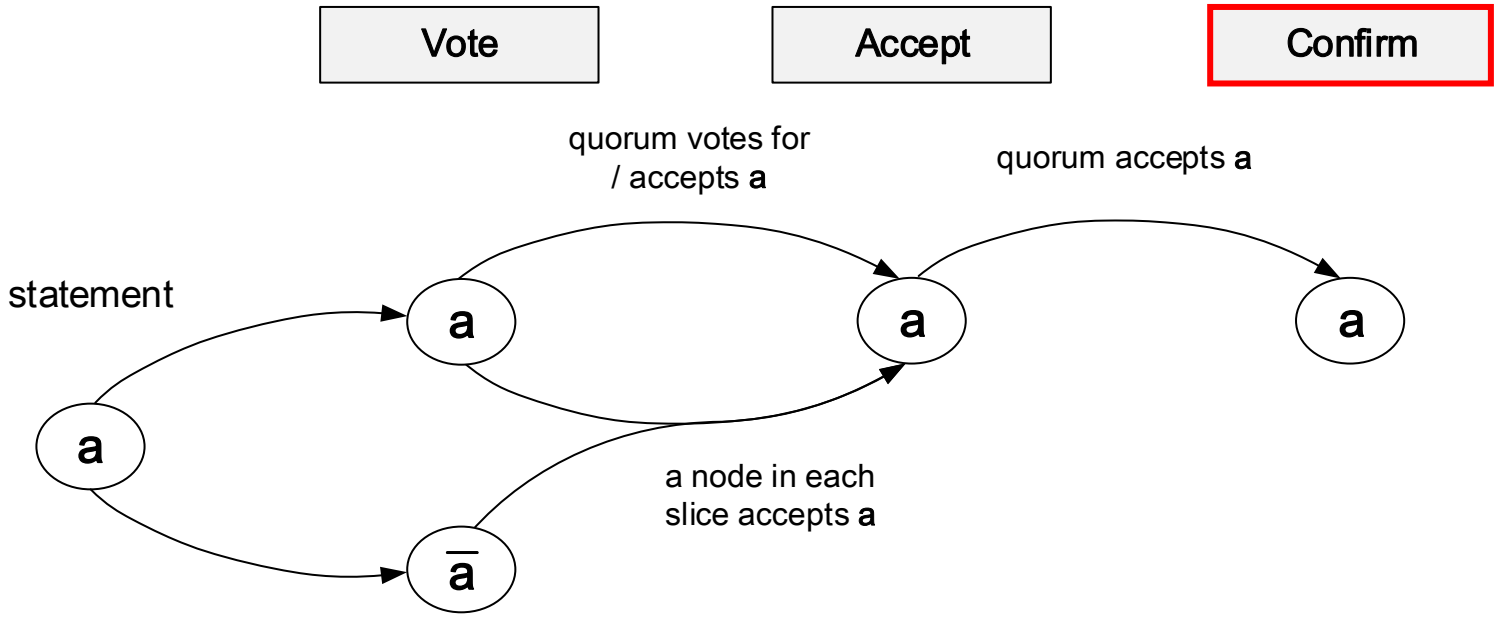
SCP(Stellar Consensus Protocol)

- Nomination
  - Federated Voting 과정
    2. Accept
       2) Blocking threshold
          - a에 반대 투표를 한 노드 v의 쿼럼 슬라이스에 a를 수락한 노드가 있으면 노드 v는 accept 수행
            - 이 때, 노드 v는 !a에 대한 투표를 무효화하기 위한 증명을 쿼럼 슬라이스에 알림



SCP(Stellar Consensus Protocol)

- Nomination
  - Federated Voting 과정
    3. Confirm
       - 쿼럼 구성원들의 수락된 투표 결과를 확인
         - Confirm(a)

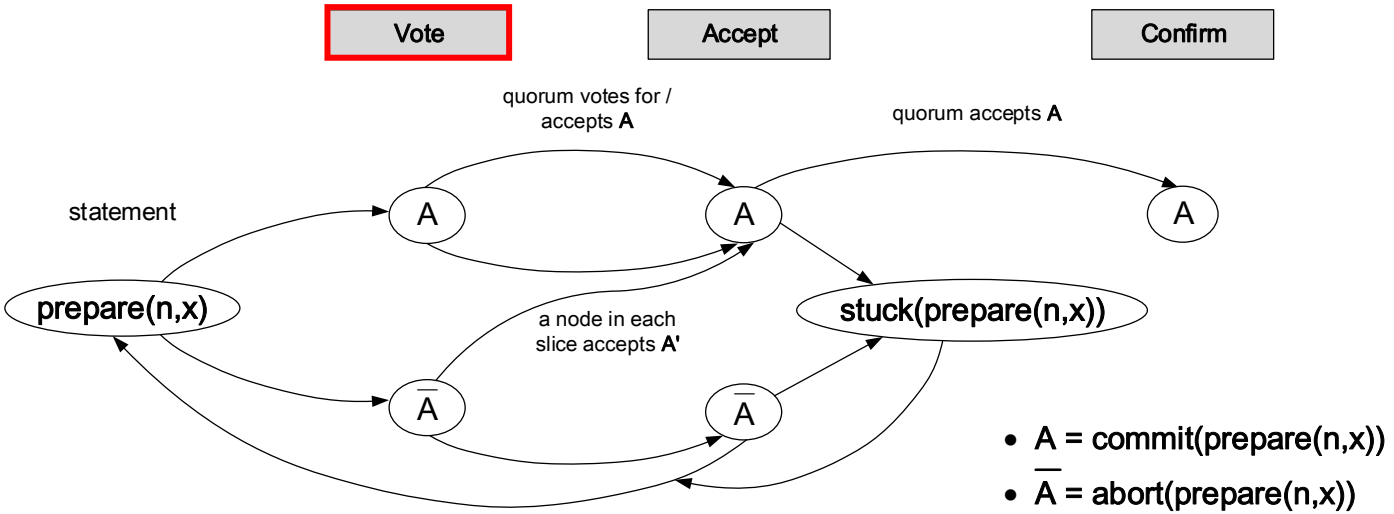


SCP(Stellar Consensus Protocol)

- Balloting Mechanism
  - 3 단계를 통해 투표를 진행
    1. PREPARE
       - 후보군(Candidate values)에 대한 투표 용지 준비(Prepare)
       - 하나의 후보가 선정 될 때까지 prepare 과정을 반복
         - 최종 선정: accept(commit(x))
    2. COMMIT
       - 준비된(prepared) 후보 값에 대한 투표
    3. Externalize
       - 최종 선정을 알림

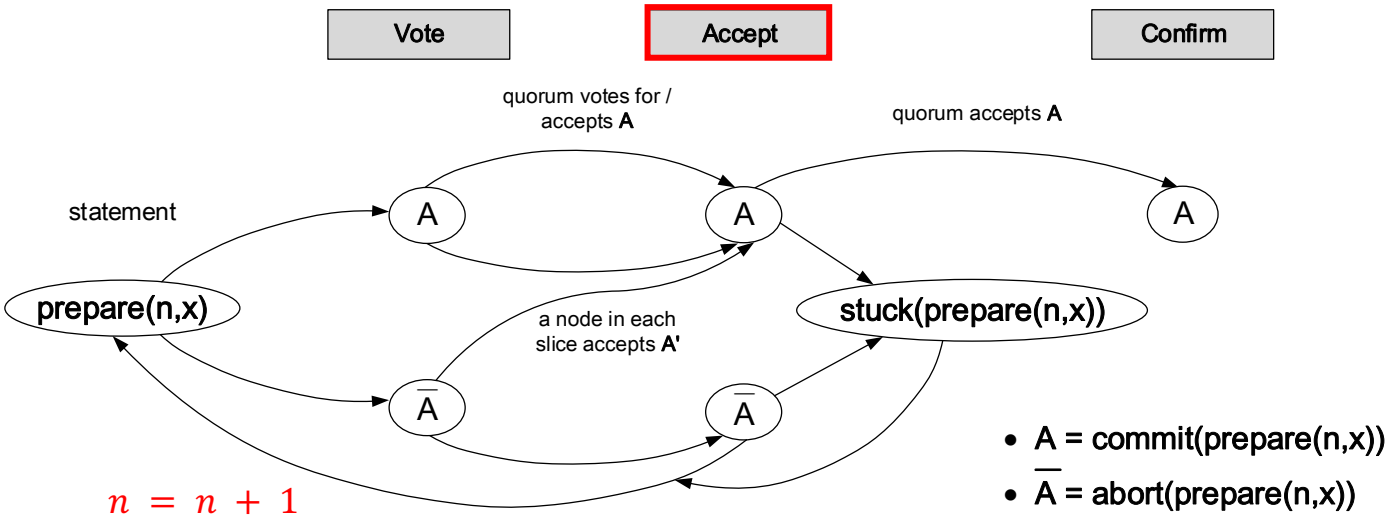
SCP(Stellar Consensus Protocol)

- Balloting - PREPARE
  - Federated Voting 과정
    1. Vote
       - 선택할 후보에 대한 투표를 진행
         - commit(x): x에 대한 수용(찬성)을 의미
         - abort(x): x에 대한 중단(반대)을 의미



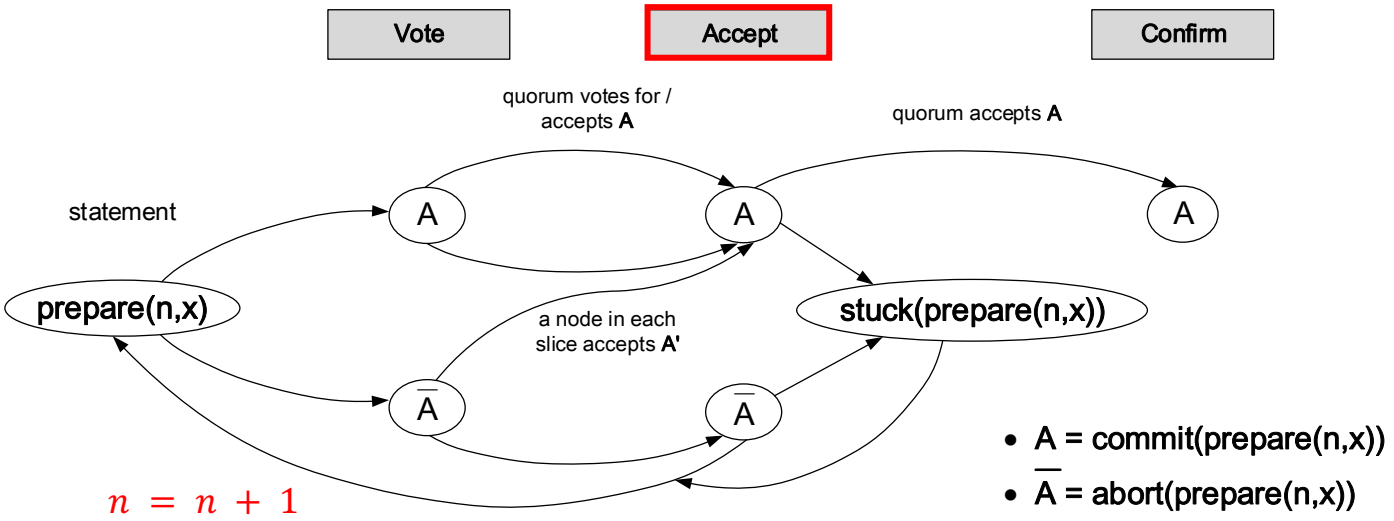
SCP(Stellar Consensus Protocol)

- Balloting - PREPARE
  - Federated Voting 과정
    2. Accept
       - commit(prepare(n,x))
         - Quorum threshold 에 부합할 시, Accpet(commit(prepare(n,x))) 로 넘어감
       - abort(prepare(n,x))
         - Quorum threshold 에 부합할 시, prepare(n+1,x)로 재투표 진행



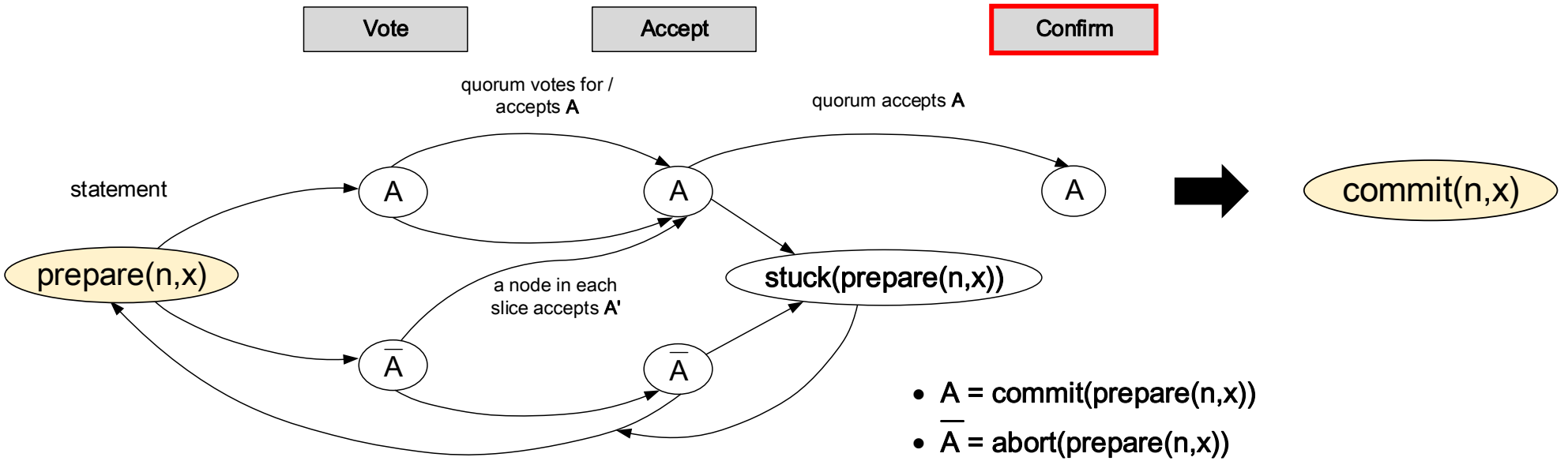
SCP(Stellar Consensus Protocol)

- Balloting - PREPARE
  - Federated Voting 과정
    2. Accept
       - Stuck 상태
         - 타임 아웃 시, 쿼럼의 모든 노드가 accpet A를 하지 않은 경우를 의미
         - 시간 초과 후 n+1 투표(재투표)를 실행



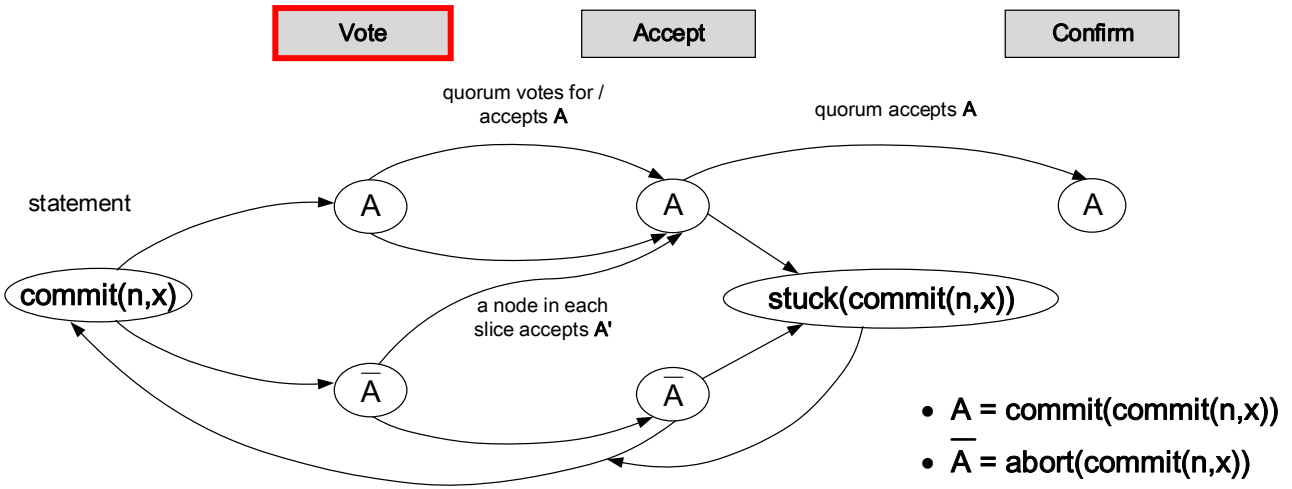
SCP(Stellar Consensus Protocol)

- Balloting - PREPARE
  - Federated Voting 과정
    3. Confirm
       - x 의 투표 용지가 준비 되었음을 의미
         - prepared(n,x)
       - prepare 메시지가 confirm 됨과 동시에 commit(n,x) 상태로 넘어감



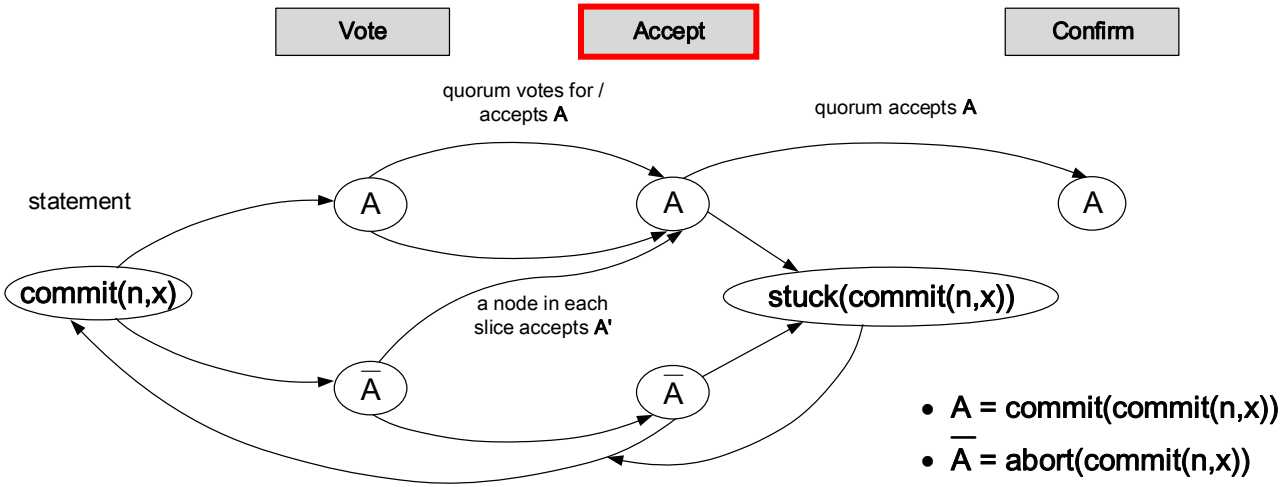
SCP(Stellar Consensus Protocol)

- Balloting - COMMIT
  - Federated Voting 과정
    1. Vote
       - 준비된 후보에 대한 찬반 투표를 진행
         - commit(x): x에 대한 수용(찬성)을 의미
         - abort(x): x에 대한 중단(반대)을 의미



SCP(Stellar Consensus Protocol)

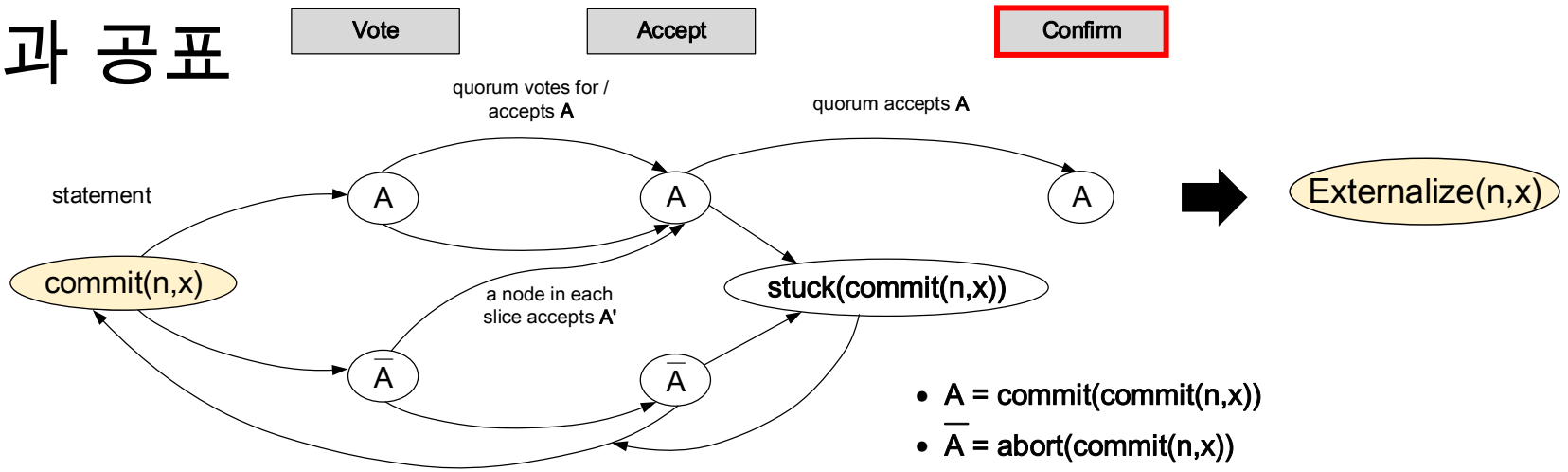
- Balloting - COMMIT
  - Federated Voting 과정
    2. Accept
       - commit(commit(n,x))
         - Quorum threshold 에 부합할 시, Accpet(commit(commit(n,x))) 로 넘어감
       - abort(commit(n,x))
         - Quorum threshold 에 부합할 시, commit(n+1,x)로 재투표 진행



SCP(Stellar Consensus Protocol)

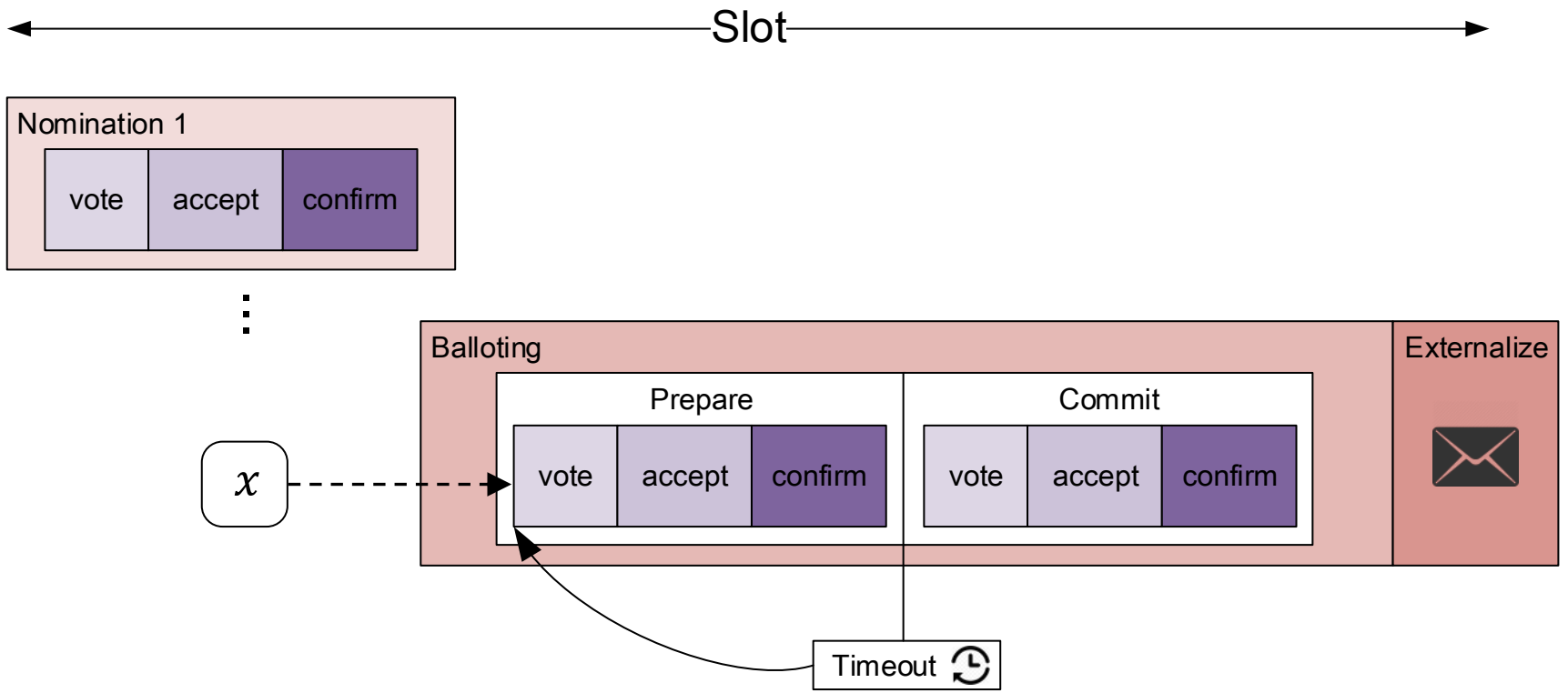
- Balloting - COMMIT
  - Federated Voting 과정
    3. Confirm
       - x 가 ballot n에서 최종 당선되었음을 의미
       - commit 메시지가 confirm 됨과 동시에 Externalize 단계로 넘어감

- Externalize
  - 투표 결과 공표



SCP(Stellar Consensus Protocol)

- 전체 동작 과정



- Federated voting(vote | accept | confirm) : 연합 투표 과정
- x : nomination의 결과로, 최종 후보군(a set of candidate values)

추가 연구 사항

- 시스템 안전성
  - Dset
- 리더 노드 선출 방법
- IETF 104
  - End system Multicast

참고 문헌

- [1] Mazieres, David. " The stellar consensus protocol: A federated model for internet-level consensus. " *Stellar Development Foundation* (2015).
- [2] “Official DINRG web page”, https://irtf.org/dinrg , 2019년 3월 방문
- [3] Malkhi, Dahlia, and Michael Reiter. "Byzantine quorum systems." Distributed computing 11.4 (1998): 203-213.

감사합니다!

백업1 - SCP 메시지 포맷

- Vote Message
  - Statement 구조체
    - `typedef opaque Hash[32];  // SHA-256`
    - `struct SCPStatement {`

```
        PublicKey      nodeID;        // v (node signing message)
        unit64         slotIndex;
        Hash           quorumSetHash;
        SCPStatement   pledges;
    }
```

    - `typedef opaque Signature<64>;`
    - `struct SCPEnvelope  {`

```
        SCPStatement   statement;
        Signature      signature;
    }
```

백업1 - SCP 메시지 포맷

- Nomination Protocol
  - Nominate 메시지 포맷
    - Typedef opaque Value<>;

    - struct SCPNominate {
      ```
      Value  voted<>;       // vote to nominate these values
      Value  accepted<>;    // assert that these are accepted
      }
      ```

    - union SCPStatement switch (SCPStatementType type) {
      ```
      case SCP_ST_NOMINATE:
            SCPNomination  nominate;
      };
      ```

백업1 - SCP 메시지 포맷

- Balloting Protocol
  - Ballot 구조체
    - ```
      struct SCPBallot {
            unit32 counter;       // n: 투표 용지수
            Value  value;         // x: 후보 값
        }
      ```
      ```

백업1 - SCP 메시지 포맷

- Balloting Protocol
  - Prepare 메시지 포맷
    - ```
      struct SCPPrepare   {
            SCPBallot     ballot;        // b.n, b.x
            SCPBallot     *prepared;     // p.n, p.x
            uint32        aCounter;
            uint32        hCounter;
            uint32        cCounter;
        }
      ```
      ```

백업1 - SCP 메시지 포맷

- Balloting Protocol
  - Commit 메시지 포맷
    - ```
      struct SCPCommit    {
            SCPBallot     ballot;
            uint32        preparedCounter;
            uint32        hCounter;
            uint32        cCounter;
        }
      ```
      ```

백업1 - SCP 메시지 포맷

- Balloting Protocol
  - Externalize 메시지 포맷
    - ```
      struct SCPExternalize      {
            SCPBallot      commit;
            uint32         hCounter;
        }
      ```
      ```