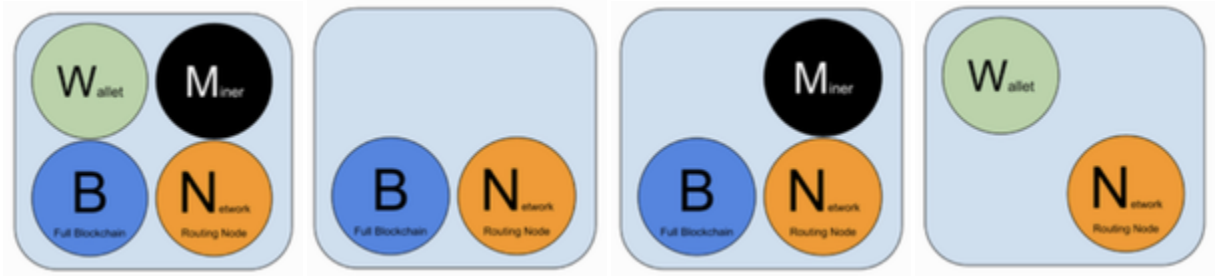
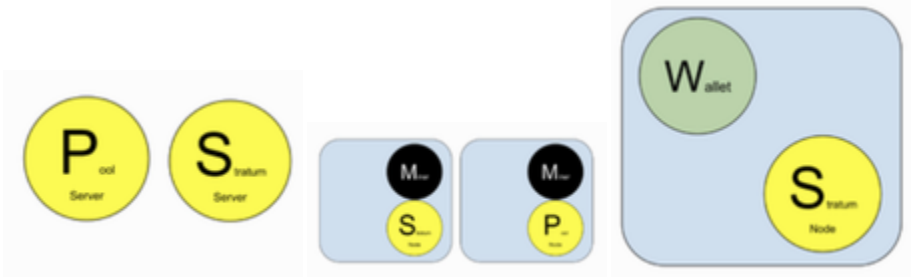
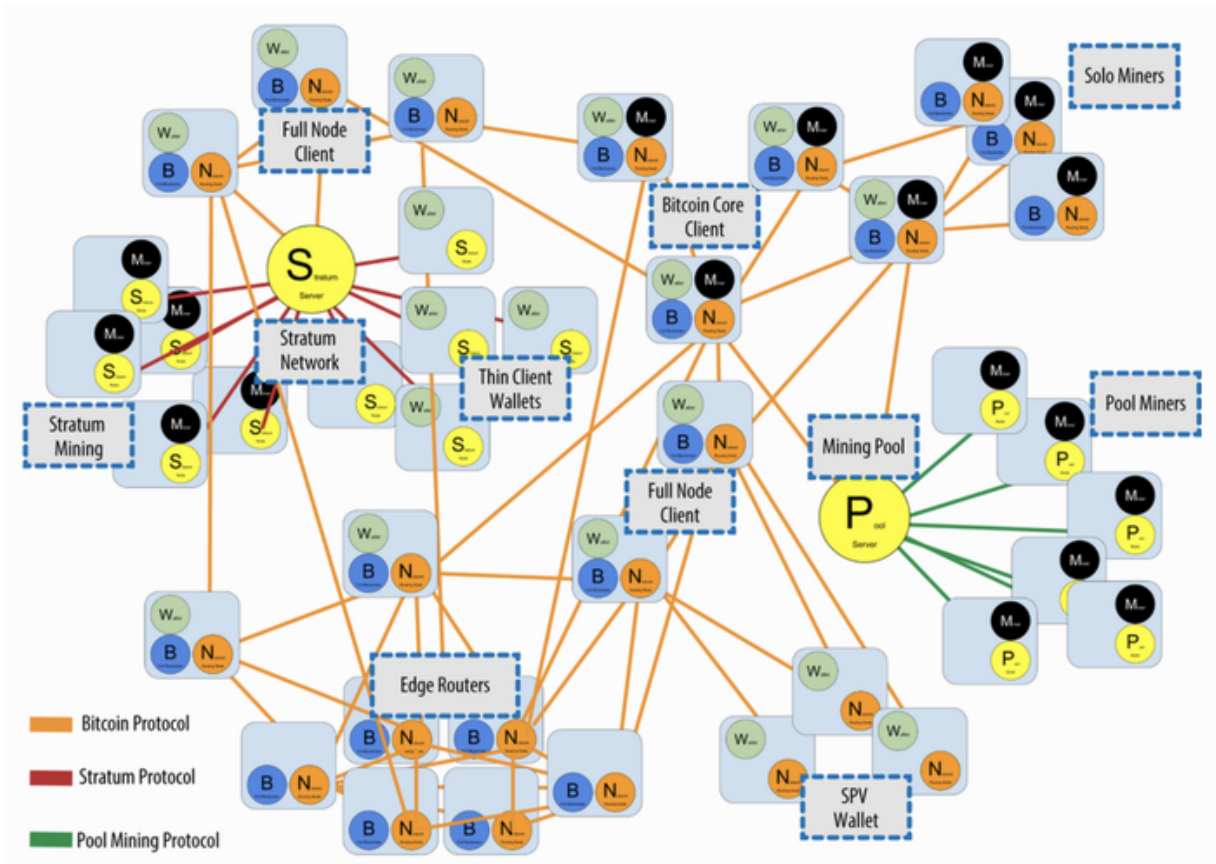
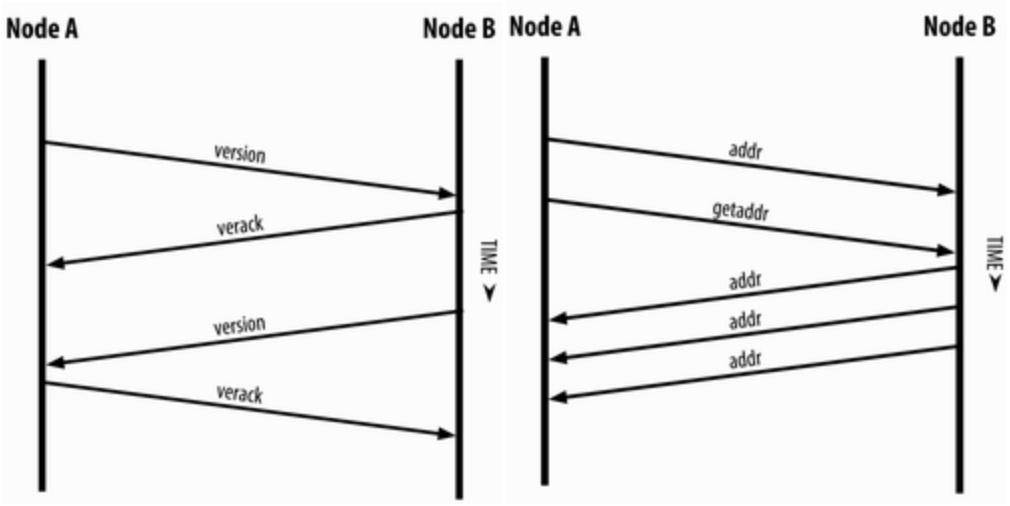
6. 비트코인 네트워크

- 비트코인 네트워크 - 비트코인 P2P 프로토콜을 실행하는 노드의 집합
    - P2P 네트워크 아키텍쳐 - 네트워크 상의 모든 노드는 '동등한' 토폴로지를 가지면서 그물망 네트워크에 서로 연결되어 있다.
    - 

    - Wallet - 키 페어를 저장하고 있는 지갑
    - Network - 네트워크 라우팅 기능
    - (Full)Blockchain - 완전한 블록체인
    - Miner - 작업증명 알고리즘을 푸는 기능
- 확장 비트코인 네트워크
    - 비트코인 P2P 프로토콜, 풀 채굴 프로토콜, 스트라텀 프로토콜 등의 연관된 프로토콜등이 포함되어 있는 네트워크 전반
    - 

    - 스트라텀 - server/client 구조를 가지는 프로토콜. lightweight 지갑/채굴 등에 쓰임.
    - 마이닝풀 - server/client 구조. 블록체인 없이 채굴기능만 제공하기 위한 클라이언트의 서버
    - 

- 네트워크 동작
    - 노드 부팅 과정



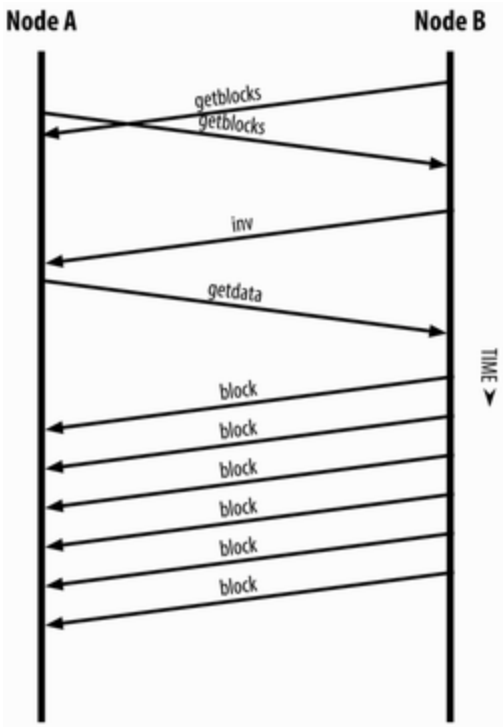
- 이미 클라이언트에 저장되어 있는 '종자 노드'에게 연결하거나 초기 실행시 하나 이상의 다른 노드의 IP를 입력해야 한다.
- 해당 노드에 연결 되었다면, 이후에는 자신의 IP를 전달하고, 연결된 노드가 알고 있는 다른 노드의 IP를 요청해서 받는다.
- 같은 과정을 일정 노드 수(약 10여개)에 연결될 때 까지 반복한다.
- 만약, 일부 노드가 90분 이상 트래픽이 없다면, 연결을 해제하고 위의 과정을 통해서 일정 노드 숫자를 유지한다.
- 비트코인 지갑 클라이언트를 이용해서 실제 연결중인 피어 정보 (피어는 8개 연결되어 있음. 중간중간 id가 비는게 있음.)

09:26:35 getpeerinfo

09:26:35

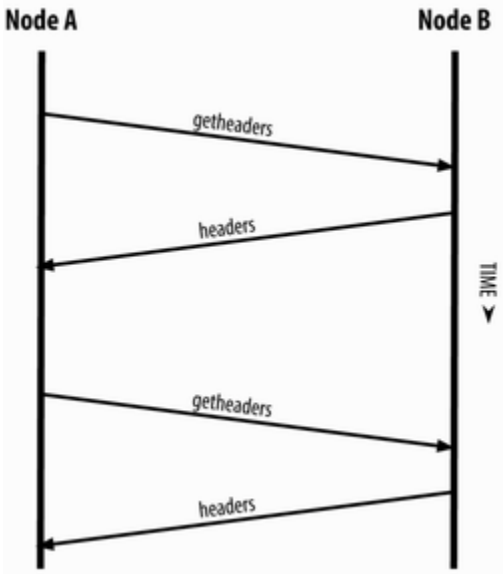

```
[
  {
    "id": 143,
    "addr": "176.9.155.246:8333",
    "addrlocal": "121.138.151.3:64187",
    "addrbind": "10.20.131.53:64187",
    "services": "000000000000000d",
    "relaytxes": true,
    "lastsend": 1510187194,
    "lastrecv": 1510187194,
    "bytessent": 89203,
    "bytesrecv": 34195653,
    "conntime": 1510186006,
    "timeoffset": 1,
    "pingtime": 0.314699,
    "minping": 0.301382,
    "version": 70015,
    "subver": "/Satoshi:0.15.0/",
    "inbound": false,
    "addnode": false,
    "startingheight": 493687,
    "banscore": 0,
    "synced_headers": 493688,
    "synced_blocks": 493688,
    "inflight": [
    ],
    "whitelisted": false,
    "bytesrecv_per_msg": {
      "addr": 30772,
      "block": 33761174,
      "cmpctblock": 12560,
      "feefilter": 32,
      "getheaders": 1021,
      "headers": 3103,
      "inv": 73244,
      "notfound": 3693,
      "ping": 320,
      "pong": 320,
      "sendcmpct": 66,
      "sendheaders": 24,
      "tx": 309174,
      "verack": 24,
      "version": 126
    }
  },
  {
    "id": 144,
    "addr": "104.129.28.170:8333",
    "addrlocal": "121.138.151.3:64197",
    "inflight": [
    ],
    "whitelisted": false,
    "bytessent_per_msg": {
      "addr": 960,
      "feefilter": 32,
      "getaddr": 24,
      "getdata": 32660,
      "getheaders": 1021,
      "headers": 25,
      "inv": 53286,
      "ping": 320,
      "pong": 320,
      "reject": 280,
      "sendcmpct": 99,
      "sendheaders": 24,
      "verack": 24,
      "version": 128
    },
    "bytesrecv_per_msg": {
      "addr": 30772,
      "block": 33761174,
```


- 인벤토리 교환 하기(풀 노드의 경우)



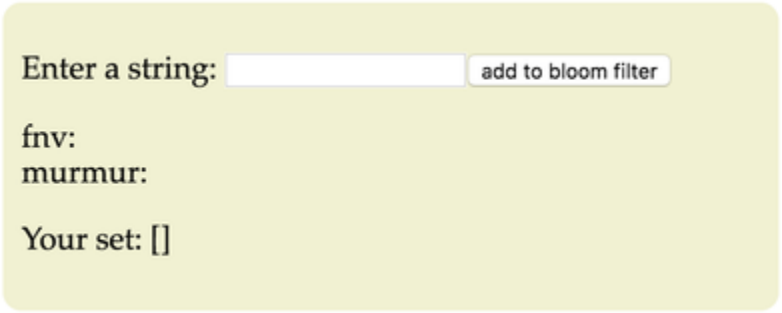
- version - 메시지 안의 BaseHeight 정보를 통해서 보유하고 있는 블록의 개수를 비교할 수 잇다.
- getblock - 메시지 안의 top 블록의 해시 정보가 있어서 이 해시가 자신의 블록 체인의 블록들의 해시값과 비교해보면서 자신의 블록체인이 더 긴지 확인해볼 수 있다.
- inv - 전송이 필요한 블록의 해시를 공유한다(500개씩)
- getdata - 더 적은 길이의 블록을 가지고 있는 노드는 getdata 메시지로 블록을 요청하고 inv에서 받았던 해시값과 비교해보면서 검증할 수 있다.

- 블록 헤더 동기화 (SPV 노드의 경우)

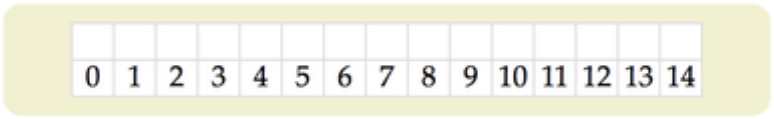


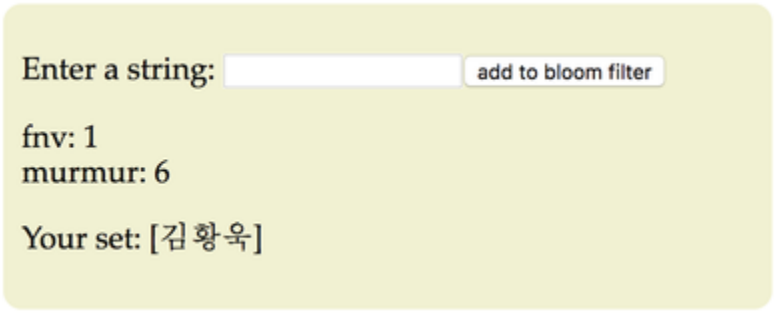
- 위의 풀 노드의 경우에서 getblocks 메시지 대신에 getheaders 메시지를 사용한다. 요청받은 노드는 2000개의 해시를 공유한다. 그 외의 과정은 위와 동일하다.
- SPV 노드는 블록자체는 저장하지 않고, 블록 헤더만 로컬에 저장하고 있다. 따라서, UTXO에 대한 데이터베이스나 메모리 풀을 구성할 수 없다.
- 단순 지불 검증
    - 블록 검증 방법
        - 이전의 블록과의 연결을 검증하는 것이 아니라(블록체인이 없으므로 불가능)
        - 해당 블록 이후에 추가로 생성된(전달받은) 블록의 깊이가 6개 이상인지 검증한다
    - UTXO 소비 여부 검증
        - 직접적인 검증은 불가능하다.
        - 머클 경로를 이용하여, 해당 거래와 해당 거래가 있는 블록을 연결한다.
        - 머클 경로는 거래의 해시와 머클 루트까지 계산할 수 있는 다른 거래의 해시들의 목록이다. 머클 루트는 블록헤더에 있으므로 해당 거래의 존재유무를 확인할 수 있다.

        - 단, SPV 노드 주변의 노드들이 작정하고 SPV 노드를 속이려고 하면 꼼짝없이 당할 수 있으므로, 신뢰 가능한 풀 노드를 확보하는 것이 중요하다.
    - 프라이버시 문제
        - SPV 노드는 계속해서 해당 거래에 대한 정보 요청을 계속 해야만 하므로, 제3자가 지갑의 사용자와 비트코인 주소를 연관시킬 수 있다.
        - 이를 블룸 필터를 통해서 해결한다.
- 블룸 필터
    - N개의 비트 어레이와 M개의 해시펑션(해시펑션의 결과값은 1 ~ N 사이)을 이용해서 어떤 패턴의 존재 여부가 'Maybe yes'인지 'Definitely Not'인지 확인한다.
    - 예제 - 김황욱, 이시용, 강순국 을 블룸 필터에 추가하고, 김황욱, 배민효를 검색하는 예제
        - 블룸 필터에 패턴 추가

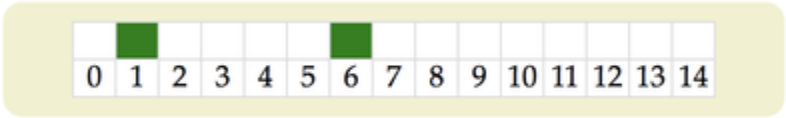


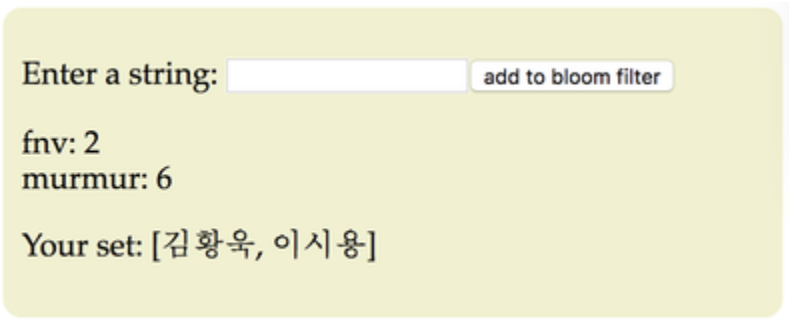
-



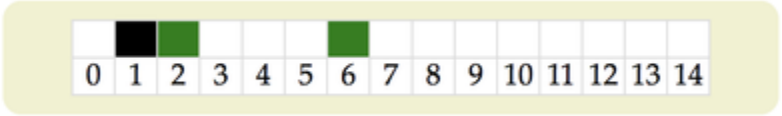


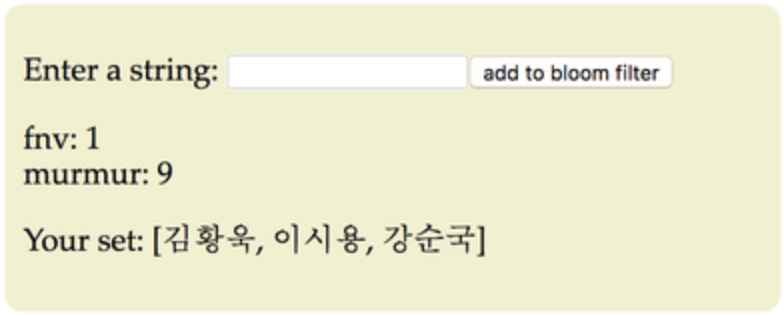
-

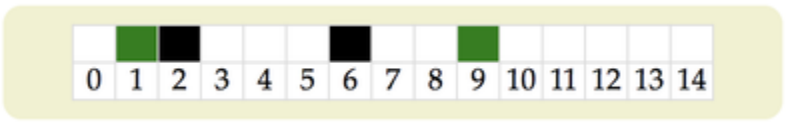




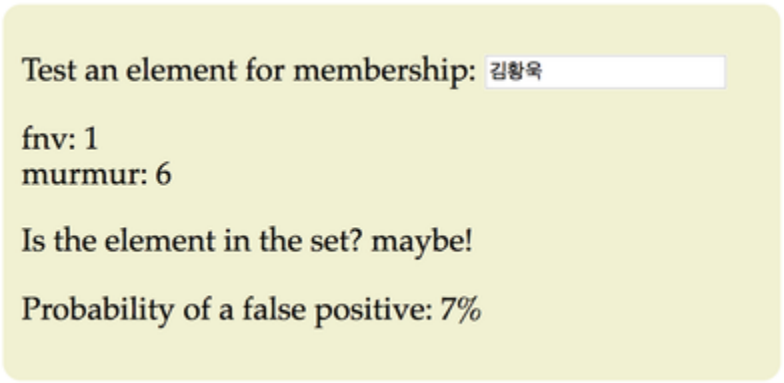
-

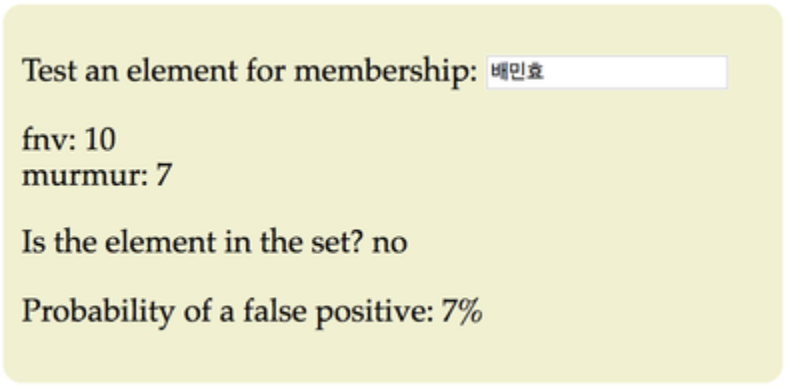




- 

- 블룸 필터에 테스트



- 

- 활용
  - SPV 노드가 블룸 필터를 초기화 한다.
  - SPV 노드가 지갑에 있는 모든 주소를 이용하여 잠금스크립트(P2PKH, P2SH 등)를 만들어 패턴을 만들어 블룸필터에 추가한다.
  - 업데이트 된 블룸 필터를 이웃들에게 전송한다.
  - 이웃들은 동일한 블룸 필터를 이용해서 'Maybe yes'의 결과값을 갖는 거래의 목록을 만들어 SPV 노드에게 전달한다.
  - SPV 노드가 찾고 있는 거래는 이웃에게 받은 거래 목록중에 있을 것이므로(거래가 올바르다면) 이로써 검증한다.