wallet 1, wallet2의 초기 지갑 정보는 다음과 같다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

1. **Hello + 학번뒤 5자리로 구성된 문자열을 포함하도록 rawtransaction을 생성**

rawtransaction을 생성하기 위해서 아직 사용하지 않은 코인에 대한 정보, UXTO (Unspent Transaction Output)를 먼저 지정해야 한다.

listunspent 명령을 통해 wallet1에 어떤 UTXO가 있는지 확인할 수 있다.  
여기서 39.999… 비트코인이 있는 첫 번째 UTXO의 txid와 vout 값을 새로 생성할 raw transaction의 input에 넣어준다.

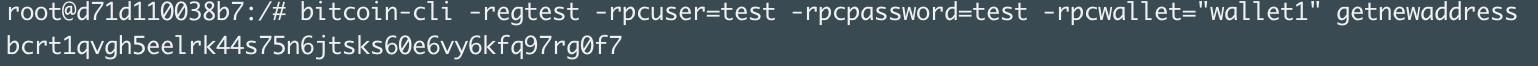
텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

raw transaction은 createrawtransaction 명령을 통해 생성할 수 있다.  
텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

Outputs에는 wallet2의 주소로 39.1 BTC을 전송하고, wallet1의 change address에 0.6 BTC을 전송하도록 하였다. UTXO는 항상 그 UTXO 전체를 소비해야 하므로, wallet2에 보내고 남은 트랜잭션 잔액은 Fee로 자동 지불하게 된다. 따라서 남은 금액 일부를 wallet1의 change address에 보냈다.



[wallet1의 새로운 주소]

이때, Hello 55615 문자열은 data value에 hex로 넣어줄 수 있다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

String 변환 시, 48656c6c6f203535363135이므로 이 값을 data value에 넣어준다.

raw transaction을 생성하면 hex string이 그 결과로 나오는데, 그 hex 값을 decode하여 생성된 트랜잭션 정보를 확인할 수 있다. vin 필드에서 scriptSig, sequence number를 알 수 있고,

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

vout 필드에서 bitcoin amount와 scirptPubKey, 새로 삽입한 문자열 정보를 알 수 있다.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

1. **해당 트랜잭션에 개인키로 서명**

생성한 raw transaction을 개인키로 서명하기 위해 signrawtransactionwithwallet 명령을 사용한다.

명령어 뒤에 raw transaction을 나타내는 hex 값을 지정하고 서명하면, 서명된 트랜잭션 데이터를 출력한다..

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

1. **서명된 트랜잭션을 regtest 네트워크로 전송**

서명한 raw transaction을 다른 노드에 broadcasting 하기 위해 sendrawtransaction 명령을 사용한다. 파라미터로 서명된 트랜잭션 hex 값을 가지며, broadcasting 후에 트랜잭션 id를 출력한다.

이때, 트랜잭션 수수료가 너무 높다는 오류가 발생하였다. raw transaction을 생성할 때 inputs과 outputs의 차이를 계산해보면 약 0.29999859 BTC이 나오는데, 이 값이 기준값보다 크기 때문이다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

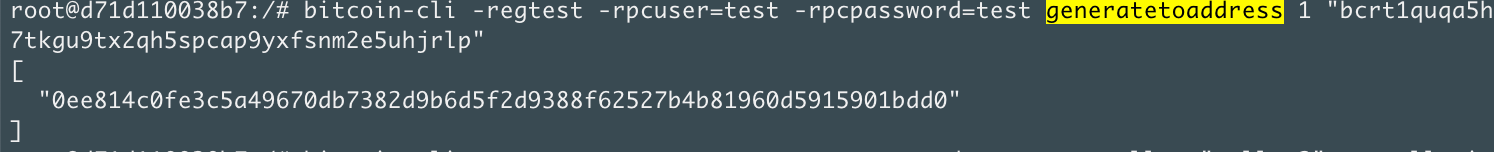
regtest 모드이므로, 수수료를 0으로 설정해도 문제 없이 트랜잭션은 전송되기 때문에 임시로 maxfeerate를 0으로 설정하여 트랜잭션을 전송하였다.

텍스트, 스크린샷, 폰트, 평행이(가) 표시된 사진

자동 생성된 설명

1. **해당 트랜잭션을 포함하는 블록 생성**

해당 트랜잭션을 포함하는 블록을 생성하기 위해 generatetoaddress 명령으로 wallet1에 블록을 생성한다.

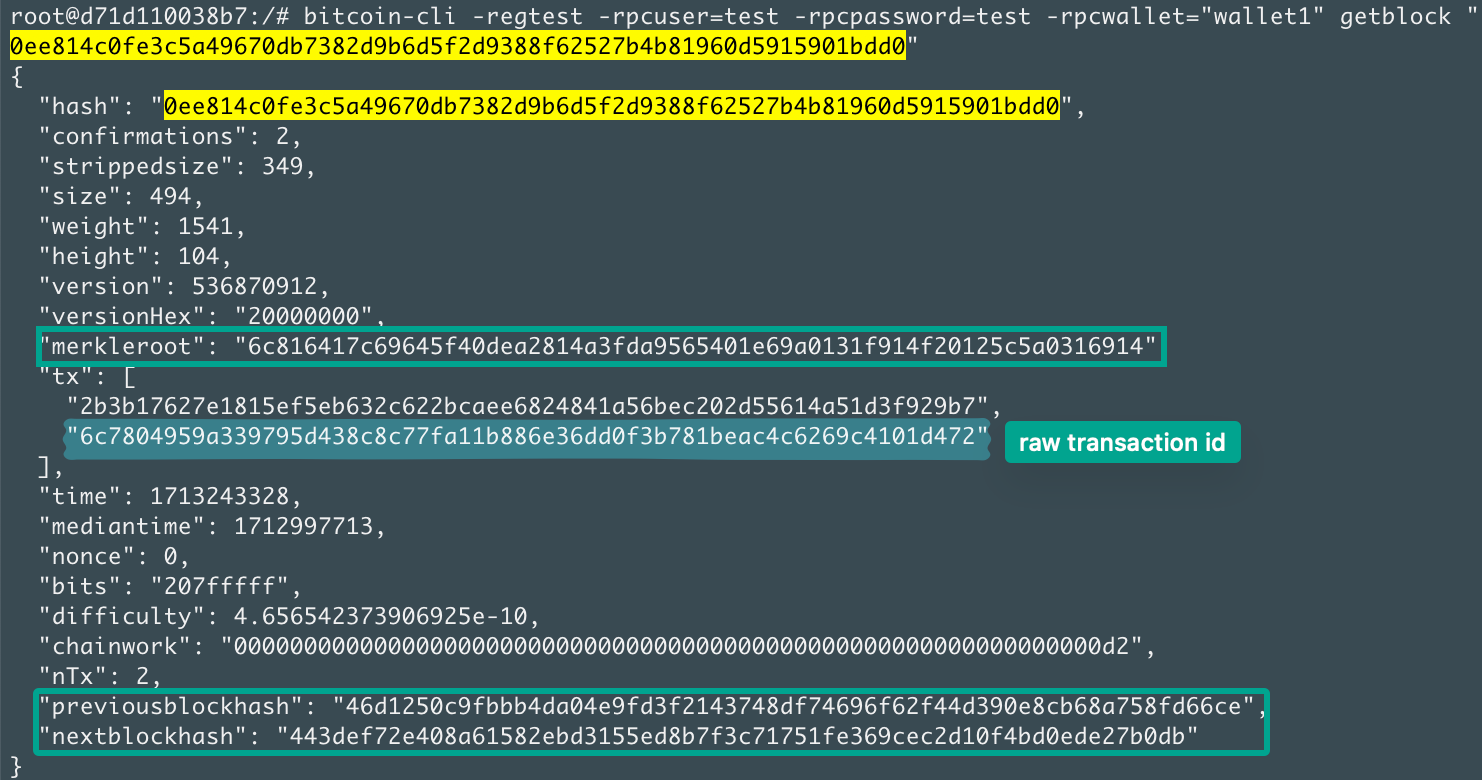


wallet2에 39.1 BTC가 잘 전달된 것을 확인할 수 있다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

같은 방식으로 raw transaction을 하나 더 만들고, 네트워크에 전달하여 서로 체인되어 있는 것을 확인해본다. 이때, 앞서 생성한 블록에 대한 정보를 출력해보면 다음과 같은데



생성했던 rawtransaction의 tx id 값이 있는 것을 확인할 수 있고, 서로 다른 두 개의 블록의 previous/next block hash 값을 통해 체인되어 있는 것을 확인할 수 있다.

