## horizontal line



Bitcoin Scripting Assignment

REPORT

**─**

TEAM : DeCentrix

Anmol Jain - 230008009

Priyanshu Patel - 230008026

Mitanshu Kumawat - 230008022

# Introduction

This report documents the process and findings of the Bitcoin Scripting assignment, which involves creating and validating Bitcoin transactions using both Legacy (P2PKH) and SegWit (P2SH-P2WPKH) address formats. The assignment required writing Python scripts to interact with bitcoind, create transactions, and analyze the scripts involved. The report also includes a comparison of transaction sizes and an explanation of the differences between Legacy and SegWit transactions.

# Part 1: Legacy Address Transactions (P2PKH)

## **Workflow**

## Setup Environment:

* Installed and configured **bitcoind** in regtest mode.
* Created a wallet named **DeCentrixStore**.
* Generated three legacy addresses: Address A, Address B, and Address C.

## Funding Address A:

* Funded Address A with 1.0 BTC using the **sendtoaddress** command.
* Mined a block to confirm the funding transaction.

## Transaction from Address A to Address B:

* Created a raw transaction sending 70% of the funds from Address A to Address B and 30% back to Address A.
* Signed and broadcasted the transaction.
* Mined a block to confirm the transaction.

## Transaction from Address B to Address C:

* Used the UTXO from the previous transaction (A to B) as input.
* Created a raw transaction sending 50% of the funds from Address B to Address C and 50% back to Address B.
* Signed and broadcasted the transaction.
* Mined a block to confirm the transaction.

## **Decoded Scripts**

## Transaction from A to B

##### Transaction ID:

c36f200aaf927ab0eeecaa2795bc287fe2b4aedee20972508c575ea19d8f8b35

##### ScriptPubKey (Locking Script) for Address B:

OP\_DUP OP\_HASH160 f3887322c62fe309bff0053bbbbee44b0de6f729 OP\_EQUALVERIFY OP\_CHECKSIG

##### ScriptSig (Unlocking Script):

304402204218331004fc22b7577931d31bea269b72d93a12e1e1910bbbd19c39ae99413902206df91614ea23220f677b9c58a5525dd6b8841dc58cae6ef28616b5c291126dff01 03b43e080a7fa94c06dde30e2edddc15ef1dc988ed9705c3125c5064d09b36e3ba

## Transaction from B to C

##### Transaction ID:

203d22842f5296eacc70e60bbaf22c41e8704965957ce787ff987aa917efebd5

##### ScriptPubKey (Locking Script) for Address C:

OP\_DUP OP\_HASH160 016d08d5e5d51cd3158ada26ae76303e5ced6e90 OP\_EQUALVERIFY OP\_CHECKSIG

##### ScriptSig (Unlocking Script):

304402204218331004fc22b7577931d31bea269b72d93a12e1e1910bbbd19c39ae99413902206df91614ea23220f677b9c58a5525dd6b8841dc58cae6ef28616b5c291126dff01 03b43e080a7fa94c06dde30e2edddc15ef1dc988ed9705c3125c5064d09b36e3ba

## **Challenge and Response Scripts**

##### Challenge Script (ScriptPubKey):

The locking script requires the recipient to provide a signature and a public key that matches the hash in the script.

##### Response Script (ScriptSig):

The unlocking script provides the signature and public key, which are validated against the challenge script.

## **Bitcoin Debugger Execution**

The Bitcoin Debugger (btcdeb) was used to validate the scripts. The debugger confirmed that the unlocking script successfully satisfied the locking script, allowing the transaction to be validated.

# Part 2: SegWit Address Transactions (P2SH-P2WPKH)

## **Workflow**

## Setup Environment:

* Loaded the existing wallet **DeCentrixStore**.
* Generated three P2SH-SegWit addresses: Address A', Address B', and Address C'.

## Funding Address A':

* Funded Address A' with 1.0 BTC using the **sendtoaddress** command.
* Mined a block to confirm the funding transaction.

## Transaction from Address A' to Address B':

* Created a raw transaction sending 70% of the funds from Address A' to Address B' and 30% back to Address A'.
* Signed and broadcasted the transaction.
* Mined a block to confirm the transaction.

## Transaction from Address B' to Address C':

* Used the UTXO from the previous transaction (A' to B') as input.
* Created a raw transaction sending 50% of the funds from Address B' to Address C' and 50% back to Address B'.
* Signed and broadcasted the transaction.
* Mined a block to confirm the transaction.

## **Decoded Scripts**

## Transaction from A' to B'

##### Transaction ID:

f534985fc48c2da409e56071644f961d55bcfcbe2c01d82b3f5d8f6954c978a9

##### ScriptPubKey (Locking Script) for Address B':

OP\_HASH160 cd5c75321d3ab5fc87ab7a8afc59a266c276e343 OP\_EQUAL

##### ScriptSig (Unlocking Script):

00147e60553a9218bcf7faf17d0f17ebb2f7c9dbbd5c

## Transaction from B' to C'

##### Transaction ID:

25e18b664ffeaf31e04b2247c917ba30ebbbb2e8e42b03b266660e31be215e71

##### ScriptPubKey (Locking Script) for Address C':

OP\_HASH160 e6dc4368773b2060bfc5567c91142c3f2fcb29a4 OP\_EQUAL

##### ScriptSig (Unlocking Script):

00147e60553a9218bcf7faf17d0f17ebb2f7c9dbbd5c

## **Challenge and Response Scripts**

##### Challenge Script (ScriptPubKey):

The locking script requires the recipient to provide a script that hashes to the value in the script.

##### Response Script (ScriptSig):

The unlocking script provides the witness data, which includes the signature and public key.

## **Bitcoin Debugger Execution**

The Bitcoin Debugger (btcdeb) was used to validate the scripts. The debugger confirmed that the witness data successfully satisfied the locking script, allowing the transaction to be validated.

# Part 3: Analysis and Explanation

## **Comparison of P2PKH and P2SH-P2WPKH Transactions**

## Transaction Size:

##### P2PKH (Legacy):

The transaction size was 225 bytes.

##### P2SH-P2WPKH (SegWit):

The transaction size was 166 bytes.

## Script Structure:

##### P2PKH (Legacy):

The locking script requires a signature and public key directly in the ScriptSig.

##### P2SH-P2WPKH (SegWit):

The locking script requires a script hash, and the signature and public key are moved to the witness data.

### 

## Benefits of SegWit:

##### Smaller Transaction Size:

SegWit transactions are smaller because the signature data is moved to the witness, which is not counted in the block size limit.

##### Transaction Malleability Fix:

SegWit fixes transaction malleability by separating the witness data from the transaction ID calculation.

# Conclusion

This assignment demonstrated the process of creating and validating Bitcoin transactions using both Legacy and SegWit address formats. The analysis showed that SegWit transactions are more efficient in terms of size and provide additional benefits such as fixing transaction malleability. The use of the Bitcoin Debugger helped in understanding the script validation process and ensuring the correctness of the transactions.

# Link to Folder Containing Screenshots and Output Files

[Link to Folder containing Output (.txt) files](https://github.com/Anmoljain2005/DeCentrix_btc_tx_analyser/tree/main/Outputs%20(.txt))

[Link to Folder containing Screenshots](https://github.com/Anmoljain2005/DeCentrix_btc_tx_analyser/tree/main/Screenshots)