Task 1



Figure : Generated Identities for Parties A, B & C

The above image shows generated identities for 3 people, A, B and C. These include private keys and wallet addresses for making transactions and redeeming transactions. Once you have your private key, you can use the system to load your wallet and access your keys and address. Using your wallet, you can generate new transactions using the transaction IDs of the transactions you wish to use the outputs from, along with the public keys of the people you wish to send currency to. Transactions are stored in a compact form on the blockchain, making it space efficient, and every transaction output is marked with the relevant user’s public key.

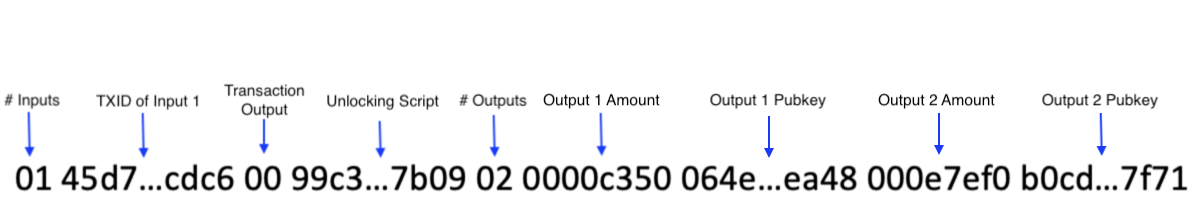


Figure : Structure of Transactions Stored on Blockchain

Transactions can only be redeemed with the correct private key using P2PK algorithms and ecdsa, meaning that only the person with access to the relevant private key can spend the coins. All transactions are verified before being added to the blockchain to ensure double-spends or other invalid transactions cannot happen, ensuring all transactions are safe and secure. All transactions on the blockchain can be viewed by anyone, meaning they are transparent. Merkle hash trees are used to verify the integrity of transactions on the blockchain, so you can check that none of the transactions have been tampered with. Since these are stored in the block header, any changes to the Merkle root which would indicate changes to the transaction list would alter the hash of the block, automatically removing it from the chain.

Task 2

For this task, I studied block 93036 on the BTC block chain. This block received 622,383 confirmations. The difficulty for mining this block was 6,866.90, whereas the difficulty on the date which I accessed the block (21/12/2021) was 24,195,286,980,613.62. This is 3,513,233,382.31 times more difficult, which shows that the hash rate of the BTC network has increased massively since November 2010. If we take the log of this number in base 2, we get approximately 31, meaning the difficulty has doubled 31 times in about 11 years. If we compare this to Moore’s Law, which posits that computing power should roughly double every year [1], this increase is clearly far larger than what should be attributed to increases in computing power alone. This shows how significant the increase in popularity of bitcoin has been since 2010: if we look at the changes in price between these two dates, we see the price has doubled roughly 17 times, which confirms that most of this increase in difficulty can be attributed to increases in popularity rather than advances in computing power. The block reward was 50 BTC, which would have been worth £8.88 at the time (20/11/2010), but would be worth £1,838,762.51 on the date accessed. Two transactions in this block by hash were 650168cf2c0111fa4fd6589f621cd1980b3c439983c72dfed9946853124935ae (<https://www.blockchain.com/btc/tx/650168cf2c0111fa4fd6589f621cd1980b3c439983c72dfed9946853124935ae>) and 3356edd82fdeea8c7a57e1b828dcef558a0d36e092486bc9eb261075f43bd28b (<https://www.blockchain.com/btc/tx/3356edd82fdeea8c7a57e1b828dcef558a0d36e092486bc9eb261075f43bd28b)>. The reward was sent to the address 1ENDYyEEXqdhKTM2pmhLghtq4NAkvgSNAU. The first transaction at this address was the coinbase transaction from the mining of the block, this transaction has the hash da2cc47a9bd590b3e3741f730045b82513c28d49463a47622430d0b32627a7f8 and it received the block reward of 50 BTC. The only other transaction involving this address was on 01/02/2021 with hash 621a5b844bc602388adff93237cc86ee0bccd334d72ee8c0012c439b03f2f698, where they received 0.00000547 BTC from the address bc1qw7wg6mk352uetqyelcs35szqa9389hzg57qxkt. There were a total of 314 recipients in this transaction, with 1.64234928 BTC being sent in total. The fee for this transaction was

0.00264275 BTC.

For John’s bitcoins, the ScriptSig script that can be used to redeem the transaction is <blockchain and cryptocurrencies>.

Password is not a secure way to store bitcoins, as someone could find the address of john’s transaction and then use brute force methods by hashing strings until one matches the address of the transaction, which may not take very long if they already have some idea of what the password could be, or if John had used the password elsewhere on either a past transaction or unsecure website. They could then use this to redeem the transaction.

P2SH would fix this security issue as it is not a password that is hashed but a script. This script usually contains a public key with a corresponding signature using ECDSA. This would be much harder to brute force as a script involving signatures would be typically a lot less “guessable” than passwords, and since the private key is not revealed in the script it can be reused, unlike the password which would be revealed in plaintext after every transaction.

Task 3

The first step is to install the Metamask browser extension. Links to versions for different browsers can be found at <https://metamask.io/download.html>. The purpose of Metamask is to provide us with a wallet with which to store Ethers at an address for the Ropsten test network which we can use to deploy contracts on the blockchain. Once you have installed the extension, it will walk you through the steps of creating a new wallet. After creating a wallet, we must switch from the Ethereum main network to the Ropsten test network, since we don’t want to have to spend actual Ethereum and waste space and resources on the blockchain when testing and developing contracts. To do this, navigate to Settings => Advanced and then toggle on ‘Show Test Networks’. Once you have done this, you should be able to switch to the Ropsten test network at the top, as shown in the series of images below.

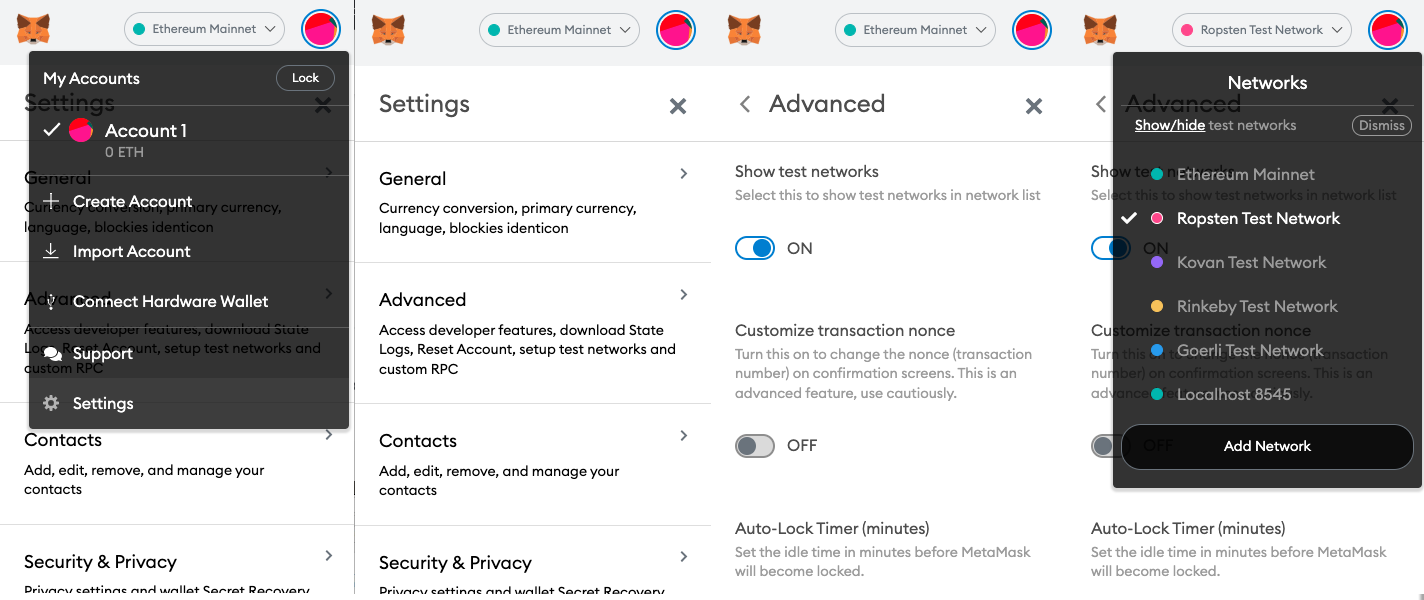


Figure : Steps to change from Ethereum Mainnet to Ropsten Test Network

Task 4

Bibliography

[1] Britannica, The Editors of Encyclopaedia. "Moore's law". Encyclopedia Britannica, 26 Dec. 2019, https://www.britannica.com/technology/Moores-law. Accessed 23 December 2021.