PLEASE CORRECT IF WRONG ANS 🙁 Couldn’t find another document for this pastpaper, will be nice if we can check ans

THANKS FOR CREATING THIS :)

1ai) The use of private and public key should be kept confidential to maintain blockchain privacy.

(looking at the following topics, I think the question is more looking for data that users might want to keep private when using the network, ie info that could identify them. So addresses, amounts, location, unlinkability and untraceability)

(account balances, transaction history)

ii) We will be able to capture the IP addresses of the peers as well as their client version by fingerprinting. By connecting to peers, we can also know their operating system during port scans.

iii) All the transactions will be stored on the blockchain, and it will not be able to remove it. Therefore, it is possible for people to trace back to the owner of the illegal transactions.

iv) We cannot be sure if one is the originator of a transaction by only observing the network layer. However, if we obtained enough connections, and if we broadcast the transaction to everyone and we receive a “get” request from all of them, it is more likely that the peer who forwarded the transaction is the originator.

bi) The first heuristic is that if there is a new address in a transaction, it is likely to be a change address, because it may infer that the sender is trying to send the fund back to himself. The second heuristic is that if there is a multi-input transaction, the inputs are likely to be signed by the same entity. However, the fundamental problem of heuristics is that they are just heuristic so it is not proven in any way to be true, so we cannot be certain about how people cluster addresses.

bii) The proposed heuristics cannot be applied on Ethereum because the heuristic is designed for UTXO blockchains such as bitcoin, which has multiple input and output. However, Ethereum is account-based, which only allows one input and output. Therefore, the heuristics are not applicable.

ci) ZCash uses transparent transactions and shielded transactions such that the source, destination, and amount will be private. Another blockchain is Monero that uses stealth addresses and ring signature to provide unlinkability and untraceability.

cii) It is weaker because we need a trusted third party to handle our transactions. And add-on mixer such as CoinJoin provides a small anonymity set, which means it is still fairly easy for people to identify addresses.

ciii) ?? Smaller transactions are people who paid those receiving larger transactions ? :0

:c

civ)

Don’t think so. Cuz as long as you spend cryptocurrency in the physical world, there’s always link to real-world financial services, cashflow etc.

2ai) A smart contract is an executed code for everyone on the blockchain. It is an agreement and will be executed when some predetermined conditions are met.

*Perhaps could mention more explicitly that it removes need for trust.*

aii) The fallback function are executed when a function identifier do not match any of the available function or when no data is supplied at all.

aiii) There three types are send, create and call. The send transaction is used when sending ETH to another address, and the user will have to specify the amount of ETH to send as well as the address of receiver. The create transaction is for creating smart contract, and creator will have to provide the EVM bytecode as well as starting balance. Lastly, the call transaction Is for calling smart contract function, address of contract, arguments for the function as well as amount of ETH will need to be specified.

aiv) A front-running attack for smart contract is when the attacker put at least p+1 transaction fee to front run a transaction with transaction fee p. (What example to put...)

Example (sandwich attack):

* Look at pending transactions in the mempool and filter for transactions that are about to buy an asset on a DEX (e.g. Uniswap)
* Replicate the buy using p + 1 gas so the miner that picks up the txs orders your buy ahead of the buy you’re trying to frontrun
* Your buy will cause unexpected slippage for the tx you’re frontrunning (I.e. you get the better execution price)
* Optionally: Add a tx with p gas, selling the asset as the unexpected slippage you caused on the attacked transaction will pump the price (back-run)

Probably an arbitrage bot will be a better example :)

av) <https://evannetwork.github.io/docs/developers/concepts/smart-contract-permissioning.html>