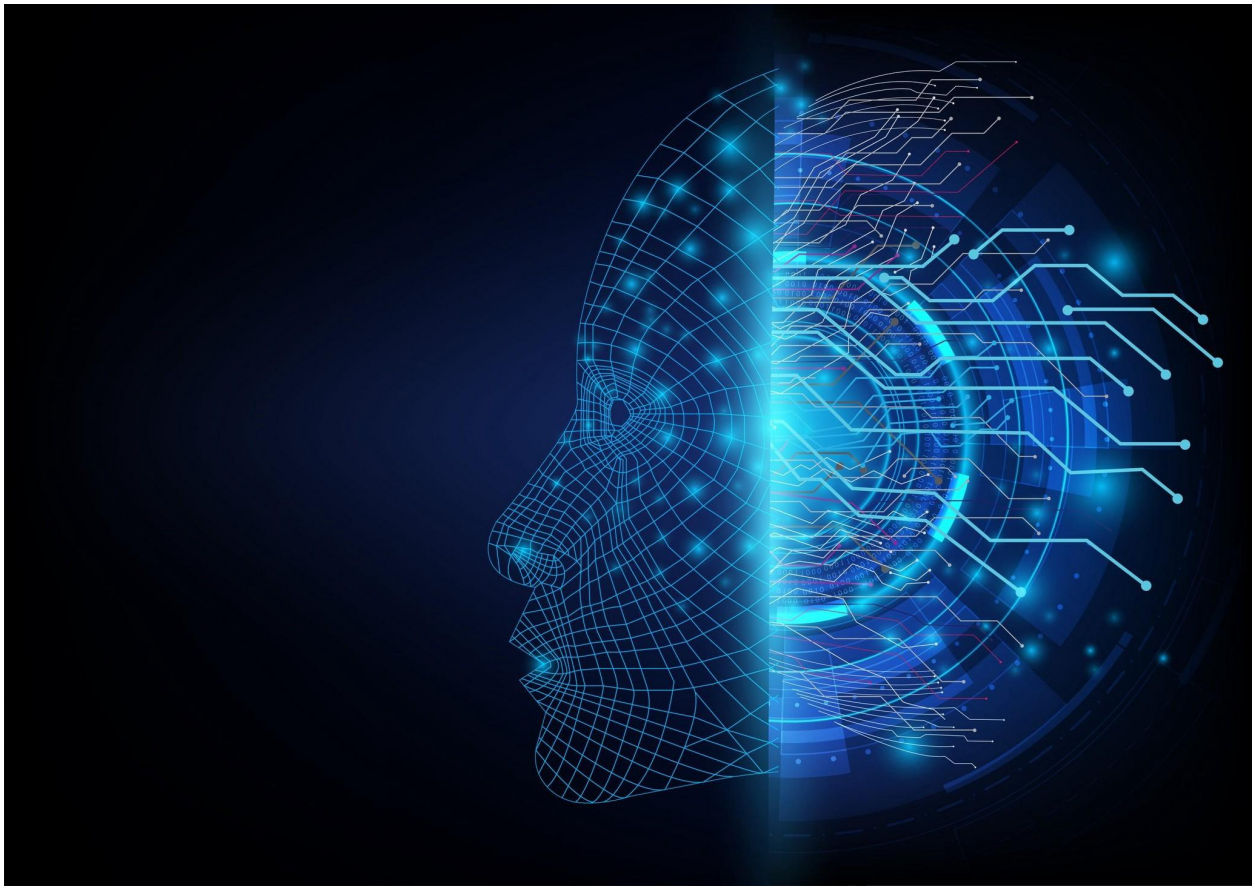
Question 2

Candidate Number: 59069

Blockchain Computing - CS3BC20



2. (a) Describe any three cyber-attack types to which a smart contract could be vulnerable.

(9 marks)

Default visibility - Default visibility relies on the fact that by default, functions in Solidity are public, and if not explicitly privatised, this can be abused by malicious actors to cause unwanted effects. Such as fund transfers being called by public users allowing funds to be stolen.

Entropy - Every node must be able to run a smart contract and return the exact same result, in a calculable way with no uncertainty. Hence no randomness. Pseudorandomness. Simulated through future block variables, hashes, timestamps, block number, gas limit.

These are not random. Miners have control over these. This means that it is possible to withhold blocks with undesirable features in an attempt to manipulate the blockchain.

Arithmetic Overflow / Underflow

Integers in EVM are fixed-size data types. Fixed-size integers wrap around when exceeding minimum or maximum limit. Attackers can bypass the balance check function, however maths libraries can prevent this.

Re-entrancy - Reentrancy works of the prerequisite that contracts such as ethereum contracts are able to call and use code from other smart contracts. The concept relies on an attacker abusing these external calls in order to force a smart contract to execute further code, achieved through a fallback function. The attacker forces the smart contract to execute calls back to itself, essentially allowing the execution to “re-enter” the contract.

Relies on code that transfers ether prior to logic checks.

(b) Describe one design feature which could support post-hoc vulnerability fixing of smart contracts and how this might be implemented despite the immutability of Blockchain.

(3 marks)

In the original smart contract code, there exists a module designed to make the smart contracts upgradable allowing bugs and vulnerabilities to be patched out, and upgrading the smart contracts.

(c) Describe an alternative consensus platform that in your view improves upon Proof-of-Work (PoW) with respect to the drawbacks of PoW that you can identify. Explain how such a consensus platform would work.

(3 marks)

An alternative consensus to a Proof-of-Work(PoW) platform, is a Proof-of-State(PoS) consensus algorithm. PoS accounts for many of the shortcomings enabled in PoW systems. One issue introduced by PoW is the ability that centralised organisations have the ability to purchase masses of powerful hardware devices in order to create a mining pool, this can result in an unfair system in which the average person stands little chance to ever win a reward, whereas just four mining pools in China control more than 50% of the Bitcoin mining power.

PoS prevents groups collaborating in order to dominate the network, as those contributing to the network are rewarded proportionately to their “staked” amount.

Another major downfall with PoW lies in the Electricity consumption it requires, which is incredibly high, especially when compared to the alternatives such as Proof-of-Stake which is much lower.

PoS is also logistically immune to a 51% attack, as it would not make financial sense to perform this on a PoS network, due to the fact it would cost the attacker more than they could ever gain, and once the remaining nodes in the network became aware of the attack, the attacker would lose all of their stake.

Proof of state functions by having the block creator chosen based on how much has been ‘staked’. PoS offers no block rewards, however the block contributing towards the PoS system earns the transaction fee. PoS algorithms work by having the user transfer their coins to a specific wallet which freezes the coins, using them as a stake. The winner is chosen randomly based on the proportion of the total amount of coins in circulation that has been staked.

(d) Specify the algorithm deployed to ensure the maintenance of blockchain integrity by verifying the authenticity of each block at the node, wallet, transaction and block levels and describe how this is achieved through a system of crypto-hash pointers and digital signatures.

(5 marks)

The main method in which to validate and verify the authenticity of a blockchain and each block at node, wallet, transactions and block levels revolves around the use of a merkle tree algorithm. A merkle tree can be evaluated from the produced merkle root value, which is a result of the hashes of each transaction being hashed together to create an individual hash.

At the most basic level, digital signatures can be used to verify authenticity, integrity and the non-repudiation of the sender. This prevents malicious actors from signing transactions and impersonating another actor.

Hashing can also be used as this creates a single value from all of the inputs, which is impractical to derive the inputs from, a slight change to the inputs would result in a drastic change in hash, and this means that the integrity of the block can be checked by re-hashing the inputs in the block to check it hashes to the same value as the stored hash.

Furthermore, the validity of the blockchain can be assessed by making sure that the elements of the previous block hash to create the same value as the stored previous hash value, iteratively throughout the entire chain.

The Merkle root can be re-calculated and validated against the stored result.

Wallet validation can be performed by checking the private key and public key match, and sufficient funds are in the wallet for any transaction.