Blockchain Technology

Assignment – 05

Aim: Program to generate single block.

Description:

1. Hashlib module:

This module implements a common interface to many different secure hash and message digest algorithms. Included are the FIPS secure hash algorithms SHA1, SHA224, SHA256, SHA384, and SHA512 (defined in FIPS 180-2) as well as RSA’s MD5 algorithm (defined in internet RFC 1321). The terms “secure hash” and “message digest” are interchangeable. Older algorithms were called message digests. The modern term is secure hash.

1. Datetime module:

Python Datetime module supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times and time intervals. Date and datetime are an object in Python, so when you manipulate them, you are actually manipulating objects and not string or timestamps.

1. Proof of Work (PoW) algorithm:

Proof of work is a form of cryptographic proof in which one party proves to others that a certain amount of a specific computational effort has been expended.

Code:

import hashlib

from datetime import datetime

*#defining global variables*

DIFFICULTY\_LEVEL = 0

BLOCKCHAIN = []

class Block:

    def \_\_init\_\_(self, data, prev\_hash="0"\*64):

        self.blockId = len(BLOCKCHAIN)

        self.timestamp = datetime.now().strftime("%d-%h-%y  %H:%M:%S.%f")

        self.data = data

        self.prev\_hash = prev\_hash

        self.nonce = 0

        self.hash = self.hashBlock()

    def hashBlock(self):

        self.hash = hashlib.sha256((str(self.blockId) + str(self.data) + str(self.prev\_hash) + str(self.nonce)).encode()).hexdigest()

        while self.hash[:DIFFICULTY\_LEVEL] != "".zfill(DIFFICULTY\_LEVEL):

            self.nonce += 1

            self.hash = hashlib.sha256((str(self.blockId) + str(self.data) + str(self.prev\_hash) + str(self.nonce)).encode()).hexdigest()

        return self.hash

data = input("\nEnter the data: ")

*#choose the difficulty leve for the Proof of Work*

DIFFICULTY\_LEVEL = int(input("\nEnter the difficulty level: "))

*# blockchain is initialized and genesis block is created*

block = Block(data)

BLOCKCHAIN.append(block)

print("\n", BLOCKCHAIN[-1].\_\_dict\_\_)

Output:

Enter the data: my name is Vishal

Enter the difficulty level: 5

{'blockId': 0, 'timestamp': '20-Aug-22 19:30:30.799876', 'data': 'my name is Vishal', 'prev\_hash': '0000000000000000000000000000000000000000000000000000000000000000', 'nonce': 2593292, 'hash': '00000ec32a35dd923c46ea2534b7cc9ad1b25752f2902359bf789ae4143a6787'}