Blockchain Technology

19BCE248

Practical 2

AIM: To create a blockchain and implement replay attacks on blockchain.

Code:

```
import hashlib
import time
import json
import threading
import random

class Block:
    def __init__(self):
        self.chain=[]

    def addNewBlock(self,msg):
        data={}
        data["time"]=time.time()
        data["id"]=len(self.chain)
        data["message"]=msg
        if len(self.chain)==0:
            data["prevHash"]=0
        else:
            data["prevHash"]=self.chain[-1]["currHash"]
        str=json.dumps(data).encode()
        data["currHash"]=hashlib.sha256(str).hexdigest()
        self.chain.append(data)

    def validate_blocks(self):
        while True:
            ind=-1
            no_blocks=len(self.chain)
            for i in range(1,no_blocks):
                if(self.chain[i]["prevHash"]!=self.chain[i-1]["currHash"]):
                    print("Tampering Found!!")
                    ind=i
                    break
            if ind !=-1:
                for i in range(ind,no_blocks):
                    self.chain[i]["prevHash"]=self.chain[i-1]["currHash"]
```

```
def validate():
    validate_thread = threading.Thread(target=block.validate_blocks,
name="Validate Blocks")
    validate_thread.start()


if __name__=="__main__":
    block=Block()
    for i in range(5):
        block.addNewBlock("My current count is"+str(i))
    # block.chain[2]["currHash"]='1'
    validate()

# print(block.chain[2])
```

Output:

```
{'time': 1665487626.7521374, 'id': 0, 'message': 'My current count i
s0', 'prevHash': 0, 'currHash': '7342494cbdbeed15fd844d3e49c161a4bb9
b5701f824236e58b0be30fdf81f32'}
{'time': 1665487626.7521374, 'id': 1, 'message': 'My current count i
s1', 'prevHash': '7342494cbdbeed15fd844d3e49c161a4bb9b5701f824236e58
b0be30fdf81f32', 'currHash': '3206997df22a8e4e7a70783a96919b9aaf9b10
4fa958fb5c91318910979b48df'}
{'time': 1665487626.7521374, 'id': 2, 'message': 'My current count i
s2', 'prevHash': '3206997df22a8e4e7a70783a96919b9aaf9b104fa958fb5c91
318910979b48df', 'currHash': '8c0c633a0fe2d07a19616f6f472ed150869999
7c0999191de554c3d25b4d5786'}
{'time': 1665487626.7521374, 'id': 3, 'message': 'My current count i
s3', 'prevHash': '8c0c633a0fe2d07a19616f6f472ed1508699997c0999191de5
54c3d25b4d5786', 'currHash': '791ea31488787b3b332a824945f65a7db854b4
60eabdb604fae3c6d77b802e86'}
{'time': 1665487626.7521374, 'id': 4, 'message': 'My current count i
s4', 'prevHash': '791ea31488787b3b332a824945f65a7db854b460eabdb604fa
e3c6d77b802e86', 'currHash': '1fed1e641dbb413f2a3bc25c10c246cb527a9c
25dd613b9b038b5a17ea689e35'}
```

When a particular hash is tampered the value would be

```
Tampering Found!!{'time': 1665487699.5234108, 'id': 0, 'message': 'My current count is0', 'prevHash': 0,
'currHash': '0e7d5ff071d00d133f5e54e65bb99e4cbf3ff7a9832756334644aedbbfb9098c'}
```

Explanation of Implementation:

- Created a class named Block which will acts as a blueprint for each blocks in blockchain.
- It will contain fields such as:

1. Time: time at which transaction was performed
2. Id: Unique key for each block
3. Message: data of each node
4. Prevhash: hash of previous node
5. Currhash: hash of current node

- For validating hash, we will create a thread which will check whether the previous hash of current block and current hash of previous block are same or not.
- For the same I have created a separate function for the same.

Learning:

These practical taught me how to build a blockchain from the ground up. Also, how can we validate the blocks if they have been intentionally altered.