Blockchain Technology

Assignment – 01

Aim: Program to generate hash value of given data, ask user for which algorithum they want use and then ask then either they want to continue otherwise they leave the program.

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. About hashing algorithums:
   1. Sha224:

SHA-224 is a one-way hash function that **provides 112 bits of security**, which is the generally accepted strength of Triple-DES [3DES].

* 1. Sha256:

SHA-256 stands for Secure Hash Algorithm 256-bit and it's used for cryptographic security. Cryptographic hash algorithms produce irreversible and unique hashes. The larger the number of possible hashes, the smaller the chance that two values will create the same hash.

* 1. Sha512:

SHA-512, or Secure Hash Algorithm 512, is a hashing algorithm used to convert text of any length into a fixed-size string. Each output produces a SHA-512 length of 512 bits (64 bytes). This algorithm is commonly used for email addresses hashing, password hashing, and digital record verification.

* 1. Md5:

The MD5 message-digest algorithm is a cryptographically broken but still widely used hash function producing a 128-bit hash value. Although MD5 was initially designed to be used as a cryptographic hash function, it has been found to suffer from extensive vulnerabilities.

Code:

import hashlib

while True:

    # Input the data from user

    data = input('Enter your data: ')

    #The encode() method encodes the string, using the specified encoding. If no encoding is specified, UTF-8 will be used.

    encoded\_data = data.encode()

    # select the hash algorithm to generate hash value

    hash\_algo = input('Enter hash algorithum [sha224, sha256, sha512, md5] : ')

    if hash\_algo == 'sha224':

        hash\_obj = hashlib.sha224(encoded\_data)

    elif hash\_algo == 'sha256':

        hash\_obj = hashlib.sha256(encoded\_data)

    elif hash\_algo == 'sha512':

        hash\_obj = hashlib.sha512(encoded\_data)

    elif hash\_algo == 'md5':

        hash\_obj = hashlib.md5(encoded\_data)

    else:

        print('please enter only listed options!')

        continue

    #the digest is returned as a string object of double length, containing only hexadecimal digits.

    hash\_value = hash\_obj.hexdigest()

    print(hash\_value, type(hash\_value))

    want\_continue = input('Do you want to continue [y/n] : ').lower()

    print()

    if want\_continue == 'n':

        break

Output:

Enter your data: vishal

Enter hash algorithum [sha224, sha256, sha512, md5] : sha224

fef472c95d1aa47bca196cc6e31e08e4eb95cbbe70d7dad6db829f03

Do you want to continue [y/n] : y

Enter your data: vishal

Enter hash algorithum [sha224, sha256, sha512, md5] : sha256

2b98fb6954368ff3f28f3367b697c1ae3b3d1e69b82a5de92b58a5e40993bbe0

Do you want to continue [y/n] : y

Enter your data: vishal

Enter hash algorithum [sha224, sha256, sha512, md5] : sha512

63c9142792d571d0f7c28eb30626d6f38792a2e7679b76d784231676d62447fb80af8953745f709c6622dda2cb4d754c262d0d31b3030a08f7b524079a6b336b

Do you want to continue [y/n] : y

Enter your data: vishal

Enter hash algorithum [sha224, sha256, sha512, md5] : md5

8b64d2451b7a8f3fd17390f88ea35917

Do you want to continue [y/n] : n