INFO7500 15975 Cryptocurrency/Smart Contract

Homework-1

Krapali Rai
NUID: 001813750
(Boston campus)

Ques.1

ANSWER: Collision of hash occurs when H(x1) = H(x2), where (x1 != x2).

a) H(x) = x % 7^12

X1 = 7^12   H(X1) = 0

X2= 7^24   H(X2) = 0

H(X1) = H(X2),   where X1 != X2

Hence, it is a collision.

b) H(x) = no. of 1 bits in x, (where x can be any bit string).

X1 = 10101   H(X1) = 3

X2= 110001000   H(X2) = 3

H(X1) = H(X2),   where X1 != X2

Hence, it is a collision.

c) H(x) = 3 least significant bits of x, where x can be any bit string

X1 = 111000   H(X1) = 000

X2= 100000   H(X2) = 000

H(X1) = H(X2), where X1 != X2

Hence, it is a collision.

Ques.2

Prove the statement: In a class of 500 students, there must be two students with the same birthday.

ANSWER: Total number of days in a year = Total number of birthdays= 366

According to the definition of hash, **H(x) => Y,**

No of outputs in set Y = 366 , Number of inputs in X = 500,

Number of inputs > output, so there is 100% probability that even if all 366 students have different birthday the 367th student will definitely going to have same birthday as one of the 366 students with unique birthdays all around the year (366 days).

It is similar to the pigeon hole principle that if no of pigeons (500 students) is greater than number of holes (366 days) then there must holes having more than 2 pigeons (students with same birthday)

Ques.4

ANSWER

For the game to be fair for both Alice and Bob, we have following requirement:

1) Alice should not be able to change the number one the Bob reveals the answer
2) Bob should not be able to crack the number by using any mean or resources other than blindly guessing the number

For the above requirement we can build following mechanism:

Step No.	Alice Action	Bobs Action
Step 1	Alice will guess a number from 1-10 ***e.g.---- 2***	N/A

Step 2	Alice will concatenate the number with a random number RN ***e.g. ---- 2 || 154***	N/A
Step 3	Alice will find the hash of the concatenated number H(x || RN) using a hash function or algorithm e.g. ***H(2||154) = 0x1D1237…***	N/A
Step 4	Alice will also find the hash of random number k with the same hash function e.g. ***H(154) = 0x5CV143..***	N/A
Step 5	Alice will send both ***H(2||154) and H(154) to Bob, as proof of integrity and won't be able to change the number***	Bob receives: ***Hash of the number selected by Alice : 0x1D1237…*** ***Hash of Random no. (RN) : 0x5CV143…***
Step 6	NA	Bob guesses the number and send it over text to Alice e.g---- 7
Step 7	Alice receives the number, tells bob that the correct number is 2 and not 7 and send the Random number =154 and algorithm to calculate and verify the hash to Bob.	NA
Step 8	NA	Bob will first calculate the hash for random number RN= 154 using the function, to check if Alice has sent him the correct hashes. ***Hash of Random no. (RN) : 0x5CV143…***
Step 9	NA	Bob will now concatenate (2 || 154) and calculate the hash. ***Hash of the number selected by Alice :*** ***H(2 | | 154) = 0x1D1237…***

Hence proved, as the hash sent by Alice match the hash calculated by Bob, it is verified that Alice hasn't changed the number.

4.

ANSWER:

```
import javax.xml.bind.DatatypeConverter;
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Random;

public class CryptoReference1 {

    public static void main(String[] args) throws NoSuchAlgorithmException, IOException
{
        String hexString =
"ED00AF5F774E4135E7746419FEB65DE8AE17D6950C95CEC3891070FBB5B03C77";
        byte[] b = DatatypeConverter.parseHexBinary(hexString);
        boolean val=true;
        while(val){
            byte[] x = new byte[32]; //256 bit array
            new Random().nextBytes(x); //pseudo-random
            String nonceHex = DatatypeConverter.printHexBinary(x);
            System.out.println("x is: "+nonceHex);
            System.out.println("ID is: "+hexString);
            ByteArrayOutputStream outputStream = new ByteArrayOutputStream( );
            outputStream.write( x );
            outputStream.write( b );
            byte concat[] = outputStream.toByteArray( );
            String concatHex = DatatypeConverter.printHexBinary(concat);
            System.out.println("x||ID " + concatHex);
            MessageDigest digest = MessageDigest.getInstance("SHA-256");//puzzle-
friendly hash
            byte[] hash = digest.digest(concat); //SHA256 hash
            String hashHex = DatatypeConverter.printHexBinary(hash);
            System.out.println("H(x||ID)" +hashHex);
            for(byte b1 : hash)
            {
                if(b1 == 29)//29 is the byte value for 0x1D
                {
                    System.out.println("found");
                    String puzzle= DatatypeConverter.printHexBinary(x);
                    System.out.println("The x is "+puzzle);
                    val=false;
                }
            }
            System.out.println();
        }
    }
    }
}
```