CRYPTOGRAPHY HASHING

A cryptographic hash function is a mathematical algorithm that maps data of arbitrary size to a bit array of fixed size. All the operations are public - such as the h(x) hash-function. No private keys and it is deterministic and random**.**

H : { 0, 1 }*	**–>**	**{ 0, 1 }^{d}**
Data of arbitrary size		Fixed Length string of d bits

Properties of Hashing

1.**Preimage resistance**

In 1976, Diffie and Hellman introduced the use of a one-way hash function in cryptography. A one-way function is a computation function wherein its direction is straight forward, but if the computation reverses in direction, it becomes more difficult. Preimage resistance is in line with a one-way function, which makes it relatively easy to protect a file.
It is exponentially hard (computationally infeasible) to determine the m message from the h(m) message digest.

2. **Second preimage resistance**

If m1 is given then it is infeasible to find m2 such that h(m1) = h(m2).
Many times, second preimage resistance is mistaken as first preimage resistance because of the similarities they share. For instance, imagine a hash function is given and it is second preimage resistance, but the resistance is not preimage. The outcome of such a result might be contradictory which implies that you will have to get preimage resistance before you can get second preimage resistance.

3. **Collision Resistance**

It is infeasible to find m1 and m2 such that h(m1) = h(m2) .
Since a hash function is a compressing function with fixed hash length, it is impossible for a hash function not to have collisions. This property of collision free only confirms that these collisions should be hard to find. This property makes it very difficult for an attacker to find two input values with the same hash. Also, if a hash function is collision-resistant then it is second preimage resistant.

SHA

It is a deterministic hashing function. SHA works in such a way even if a single character of the message changed, then it will generate a different hash. Secure Hashing Algorithms are required in all digital signatures and certificates relating to SSL/TLS connections. In blockchain technology, particularly in cryptocurrencies like Bitcoin, SHA-256 plays a crucial

role. It is integral to the mining process, where it helps solve complex mathematical puzzles to validate and add new transactions to the blockchain.

The SHA hash function can be implemented using the python library hashlib which contains a lot of famous and important cryptographic hash functions.

```
import hashlib
str = "My name is Kamyak"
result = hashlib.sha256(str.encode())
print("The hexadecimal equivalent of SHA256 is : ")

print(result.hexdigest())
```

Diffie Hellman Key Exchange

Diffie-Hellman Key Exchange (DHKE) is a cryptographic protocol that allows two parties to securely share a secret key over an insecure communication channel. Until the 1970s, the development of secure communications entailed ever more sophisticated symmetric ciphers. A key is used to scramble messages, and the same key is used to unscramble them.

They described how, by using one-way functions, two parties can arrive at a secret number that they both know, but that any eavesdropping party cannot determine. This secret is known as a shared secret key. Once the shared secret key is in place, we can switch to traditional symmetric encryption for speed.

The way it works is mentioned below:
First the two parties choose prime p and g such that g is a primitive root modulo p. These numbers p and g need not be kept secret from other users.
A number g is said to be a primitive root modulo p if g^1 ,g^2, .. g ^ φ(n) modulo n is a permutation of the set {a∈N : 1≤a<n, gcd(a,n)=1}.

Now both the parties choose large numbers m and n as their private keys. The first person then computes g^m(mod p) and the other computes g^n(mod p) and both of them send these values to each other. The shared key "**K**" is g^(mn) (mod p). Now they can use this key to share information with each other.

In order for a potential eavesdropper to get access to K, they would need to obtain its value using g, p , M = g^m (mod p) and N = g^n (mod p).
This can be done by computing m from M = g^m (mod p) or n from g^n (mod p). This is the discrete logarithm problem, which is computationally infeasible for large p.

Computing the discrete logarithm of a number modulo p takes roughly the same amount of time as factoring the product of two primes the same size as p, which is what the security of the RSA cryptosystem relies on. Thus, the Diffie-Hellman protocol is roughly as secure as RSA.