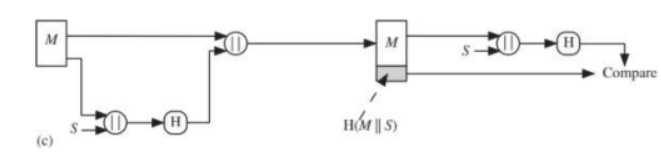
**PRACTICAL 7**

**Aim:** Refer to the attached figure here. Bob is preparing to send message to Alice. Bob and Alice both secretly computes the code(s) without sharing on any communication channel. Suggest key exchange algorithm to Bob and Alice for securely exchange information without sharing actual key. Once they form secret code, Bob applies SHA256 hash algorithm on original message (M) plus code (s) and send hash of original message and code (M||s) to Alice. Alice will receive bundle of H(M||s) and first append code (s) with received message (M) and produce hash of the message (H) that compare with H(M||s) to make sure that message is not altered by any attackers.

****

Task to perform:

1. Use some key exchange algorithm to calculate value of s (secret code) which must be unique at sender and receiver side.

2. Implementation can be done using any programming language such as c, c++, java, python, c#, javascript, php etc.

3. Apply SHA256 on message and secret code and display it on output screen. Verify the hash value at receiver end.

**Hardware Requirement:** Computer/Laptop.

**Software Requirement:** Any Python IDE.

**Theory:**

The SHA-256 algorithm is a variant of SHA-2 (Secure Hash Algorithm 2), a successor to SHA-1 developed by the National Security Agency in 2001. The SHA-256 hash function is a patented cryptographic hash function that generates a 256-bit result.

What exactly is hashing? Encryption converts data into a secure format that cannot be read unless the recipient possesses a key. The data can be any size in encrypted form, and it's typically just as long as it is unencrypted. In hashing, on the other hand, data of any size is mapped to data of a specific size. A 512-bit string of data, for example, would be turned into a 256-bit string using SHA-256 hashing.

**CODE:**

import hashlib

# prime no

q = 23

# p is primitive root of q, p < q

p = 9

private\_key\_alice = 4

public\_key\_alice = int(pow(p,private\_key\_alice,q))

private\_key\_bob = 3

public\_key\_bob = int(pow(p,private\_key\_bob,q))

print("--------------------Deffie Hellman Data--------------------")

print("Prime No q            | ", q)

print("Premitive Root Of Q, p| ", p)

print("Private Key Of Alice  | ", private\_key\_alice)

print("Public Key Of Alice   | ", public\_key\_alice)

print("Private Key Of Bob    | ", private\_key\_bob)

print("Public Key Of Bob     | ", public\_key\_bob, "\n")

print("------------------Deffie Hellman Data End------------------\n\n")

secret\_key\_alice = int(pow(public\_key\_bob, private\_key\_alice,q))

original\_msg = "hello world"

prpared\_msg\_to\_hash = str(original\_msg) + "," + str(secret\_key\_alice)

hashed\_msg = hashlib.sha256(prpared\_msg\_to\_hash.encode("utf-8")).hexdigest()

msg\_to\_send = str(original\_msg) +","+ hashed\_msg

print("----------------Alice Is Preparing To Send------------------")

print("Secret Key For Alice | ", secret\_key\_alice)

print("Original Msg         | ", original\_msg)

print("Prepared Msg To Hash | ", prpared\_msg\_to\_hash)

print("Hashed Msg           | ", hashed\_msg)

print("Msg To Send          | ", msg\_to\_send)

print("-------------------Alice Sended The Msg---------------------\n\n")

msg\_recieved = msg\_to\_send

msg\_recieved = msg\_recieved.split(",")

secret\_key\_bob = int(pow(public\_key\_alice, private\_key\_bob,q))

msg\_recieved\_with\_s = msg\_recieved[0] + "," +str(secret\_key\_bob)

hashed\_recieved\_msg = hashlib.sha256(msg\_recieved\_with\_s.encode("utf-8")).hexdigest()

print("---------------------Bob Recieved Msg-----------------------")

print("Secret Key For Bob   | ", secret\_key\_bob)

print("Msg Recieved         | ", msg\_to\_send)

print("Original Msg         | ", msg\_recieved[0])

print("Msg Bob's Secret Key | ", msg\_recieved\_with\_s)

print("Hash Of Recieved Msg | ", hashed\_recieved\_msg)

print("------------------------------------------------------------\n\n")

print("-----------Validating Hashed Msg With Recieved Hash-----------")

if(hashed\_recieved\_msg == msg\_recieved[1]):

    print("Msg Is Non Compromized")

else:

    print("Msg Is Compromized")

**OUTPUT:**

