A. Hash function can be used to provide different applications. Hash function offers different application, such as:

1. Integrity, authentication.
2. Integrity, authentication and confidentiality.
3. Integrity, authentication and Non-repudiation
4. Integrity, authentication, confidentiality & DS

Choose **one** of these application and:

- Illustrate any assumption if needed
- Display how the hash function provided these application by figure.
- Display how the hash function provide these application by equation
- Indicate exactly, which part has provided each application.
- Remember you have to explain your choice in choosing any part, such as type

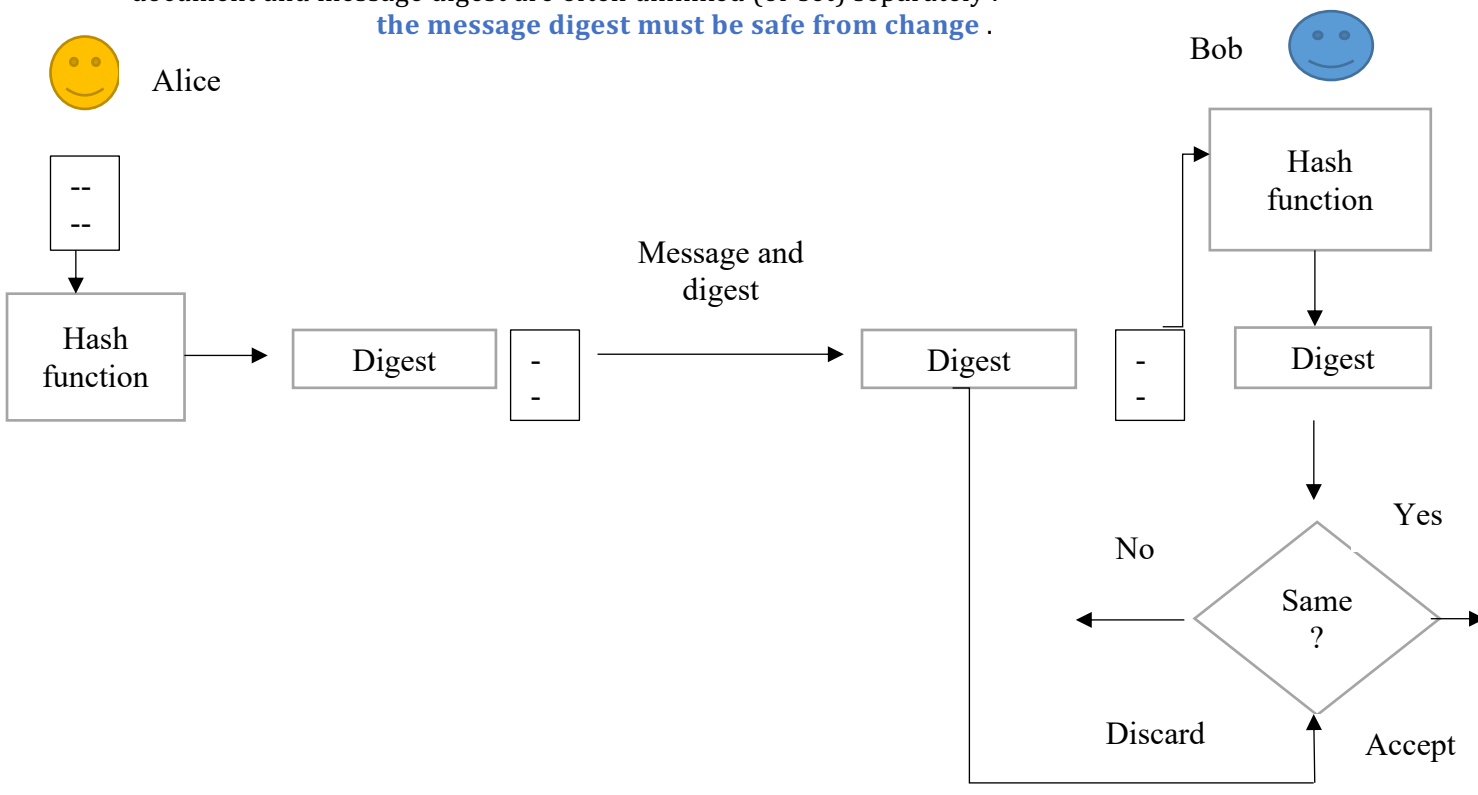
  of key, sender and receiver, ..etc.

The choice was made from the list **Integrity, authentication**.

Message document ⟶ hash function ⟶ message digest

the various the document and message digest are similar and same differences . the document and message digest are often unlinked (or set) separately .

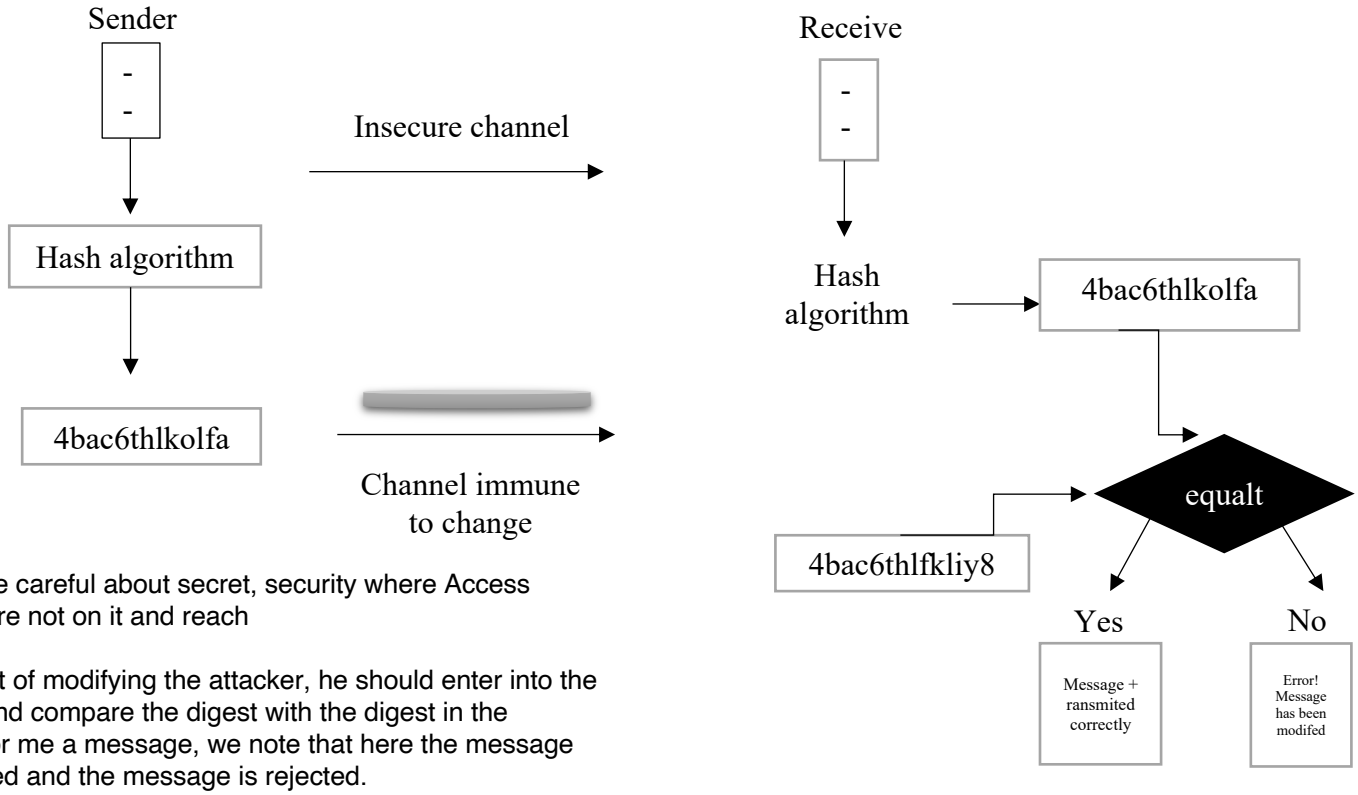
the message digest must be safe from change .



Outputs :

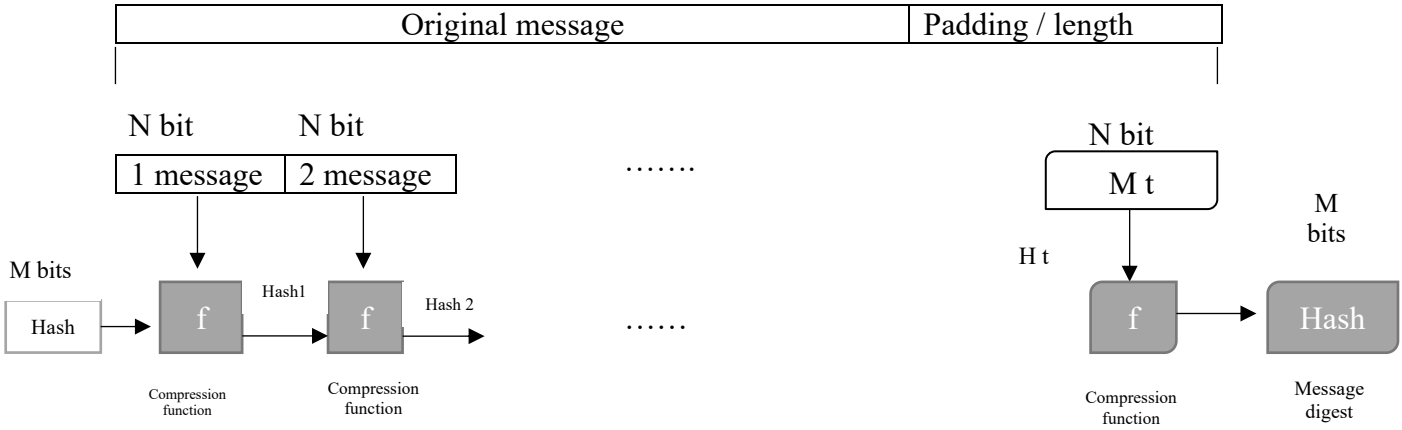
Alice wants to send a message to Bob takes the message to the hash to produce for us digest after message and suplatled digest. After the receiver, he enters the hash after a new digest after comparing the value or content of yes or no with the digest that I received from the base canter if the content is identical and the sending time has not been modified And the message is accepted, but if it is changed and does not match, if it is rejected, it has been modified at the time of sending. This method is used to verify the integrity of the sent data, regardless of whether it was changed by attacker until the encryption happened to the data during the sending process. Destroyed a message and changed its content and I am keen on the integrity of the message. Using this method, this method is in order to keep it from corruption or malfunction of the attacker.

We explain in the drawing a scenario if the message was correct and it was accepted, and here we differ from the drawing before and explain in which the addition is that the original message after entering it in the hash algorithm and according to the digest value is sent an insecure channel is not necessary if the attacker has modified or copied the message and Receiver forwarded through Digest. Here I can tell if the message has been modified



We must be careful about secret, security where Access attackers are not on it and reach

In the event of modifying the attacker, he should enter into the Receiver and compare the digest with the digest in the Receiver for me a message, we note that here the message has changed and the message is rejected.



To make the input fixed half padding the length. The length of each block is m / n. After we enter a hash function after an initial value, hash 1 is deduced for us, and after message 2 a hash function 2 appears, and so on until the last block of the message arrives m. ht .

Alice -> Bob : (m , hash)
C -> (msg , key) = fixed length code (hash code)
A -> B (m* hash m) kab .

MESSAGE AUTHENTICATION :

A message digest
It achieved integrity but authentication did not achieve us during the transmission process , but we were not entitled to sender .
We need for message authentication is code MAC 'message authentication code ' .

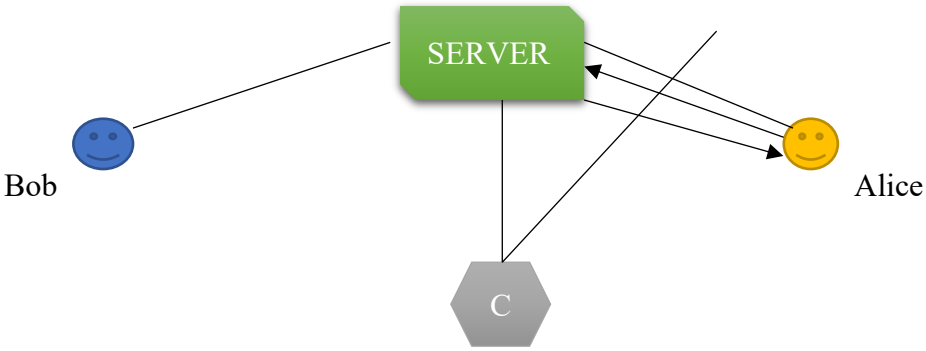
B - The following is a key exchange protocol used by two clients, A and B, to obtain a symmetric key K , using a trusted server, S. Assume that A and B had previously

obtained the symmetric keys K_{AS} and K_{BS} securely with the server. Also assume that

anyone can securely obtain a secret symmetric key with the server.

I. $A \rightarrow S : A, B$
II. $S \rightarrow A : (K) K_{AS}, (K) K_{AB} K_{BS}$
III. $A \rightarrow B : (K) K_{AB}, A K_{BS}$

• Identify the attack that can be executed against this protocol, assuming that an attacker C can eavesdrop, block, or modify all messages. The attacker also has access to old (expired) keys.



In order to communicate with some but not known , some requested a shared key
A ⟶ S : A,B application received for A,B
C ⟶ S : A , C enter c disconnects before connecting s says " give me an ID follow Alice and Bob
S ⟶ C : (kac) kas, (kac) kcs .. s replies c saying take this key to you

Mutual Encrypted key A — Mutual Encrypted key C

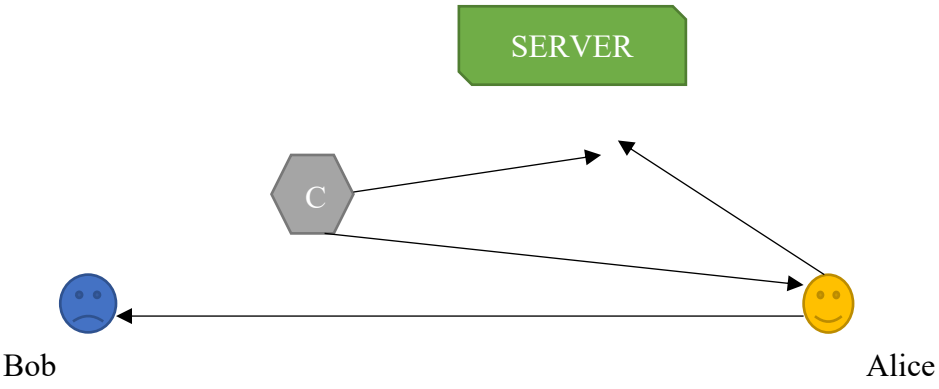
C ⟶ A : (kac) kas, (kac) kcs .. C sends an A say S this is the key
A ⟶ C : (kac) kcs , A second time he sends C while he is B he sends his private key in it and identifies himself.
A now thinks he's communicating with B, but he's actually communicating with C who is able to read all the messages sent by A .

It is possible to break the protocol without breaking the code. The opponent may be a legitimate participant in the protocol from within or from outside. This assumption leads to an attack against the other.
-main the middle attack message modification .

C - The above protocol is now modified to become

I. $A \rightarrow S : A, B$
II. $S \rightarrow A : (K, B)\ K_{AS}, (K, A)\ K_{BS}$
III. $A \rightarrow B : (K, A)\ K_{BS}$

• Why does the new modification improve the security over the old version? Identify the new vulnerability that exists in the new version



Previous attacks show that we must add participants messages in a secure manner. We assume that the competitor is able to obtain the value of the KAB session key used in any operation of the old enough protocol.

A ⟶ S : A, B
S ⟶ A : (KAB, B) KAS, (KAB ,A) KBS
A ⟶ B : (KAB,A) KBS

A message from A to S has cleared the key AB is outdated. The session key used by A, B and in a previous session, the key AB is outdated C.

It may have succeeded in breaking it even if it is a key, c, AB is not broken. It can replay old messages in this new session and it can cause a lot of problems .

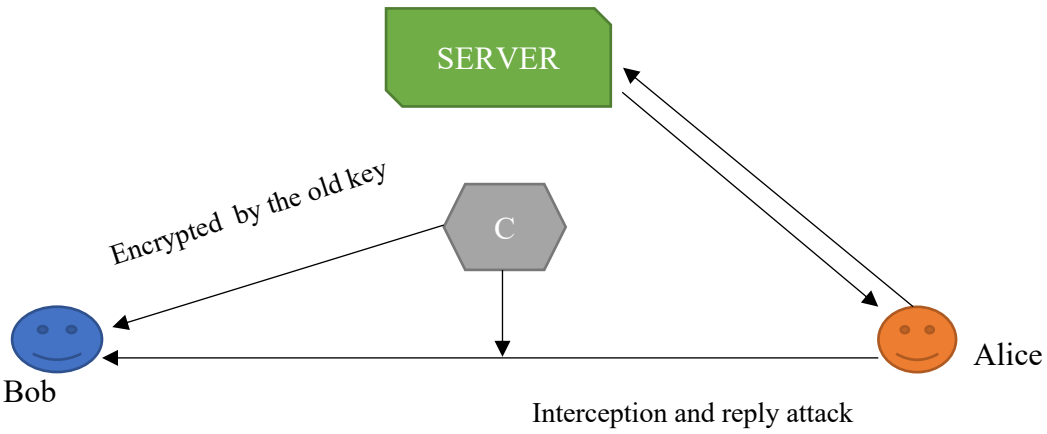
D - The above protocol is now modified to become

I. $A \rightarrow S : A, B, NA$
II. $S \rightarrow A : (K_{AB}, B, NA, (K_{AB}, A) K_{BS}) K_{AS}$
III. $A \rightarrow B : (K, A) K_{AB}$
IV. IV. $B \rightarrow A : (NB) K_{AB}$
V. V. $A \rightarrow B : (NB - 1) K_{AB}$

• Identify the vulnerability in this new version.

In order to prevent restart operations we must set timestamps or unexpected codes in The nonce protocol file is a random value created by one party and returned to that party to show that the message was newly created using nonces.

This nodem protocol is a popular protocol, which Needham and Schroeder, published in 1978, still has flaws if the attacker knows the old session. Keys that can be used in the last three messages and then B thinks he is too. Communicate with the attacker using an old session key C .. By adding nonces, we finally achieve a valid protocol.



A ⟶ S : A,B ,NA, NB
S ⟶ A : (KAB ,B , NA) KAS, (KAB,A,NB) KBS
A ⟶ B : (KAB,A,NB) KBS
B ⟶ A: B, NB

in the scenario c can intercept the message from A to B and C sends to B new session key encrypted by old (kab) than c , encryptions the new session key between C and B (kac) by
C ⟶ B (KAB,C) KAB
B ⟶ C (NB) KBC
C ⟶ B (NB-1) KBC

so as we see that this Protocol are often modified by the attacker to his purpose even with the modification , the sender done it wasn't enough to assure the serve of we Protocol.

E. Modify the protocol in part (D) to have a secure protocol.

modified the protocol partially (D) to possess a secure protocol to boost the protocol and make immune and has no vulnerability in it we'd like to repair it in away that the A send nonce value of NA to S and also nonce value for b NB to the server to make sure that the message integrity . the subsequent the instance illustrate
A ⟶ S : A,B , NA , NB
S ⟶ A : (KAB , B , NA) KAS , (KAB , A, NB) KBS
A ⟶ B : (KAB , A , NB) KBS
B ⟶ A : B , NB

by doing this we will have secured Protocol which will assure the integrity for A,B

C&B the attacker intercepted and sent old keys . then A&B accepted the keys shared by the attacker this is replay attack .