Unit 3

Q) Types of Authentication

1) Password-based authentication
2) Address-based authentication
3) cryptographic authentication
4) smart cards
5) biometrics
6) mutual authentication

To secure a network the first step is to avoid unauthorized access to the network.
This can be achieved with any type of authentication mechanisms.
When one node wants to communicate with the other node in a network in a secure manner they use an authentication mechanism.
The node that wants to communicate has to prove its identity in the network so that its right to access the network resources can be determined.

- Knowledge based Authentication is based on "What you know" i.e what the users know eg:, usernames, passwords, keys etc
- Artifact based Authentication is based on "What you possess" i.e what the user possess such as Certificates, tokens, smart cards etc.
- Biometric based Authentication is based on "What you are" i.e what the user inherits for eg: biometric techniques

1) Password-based authentication

Setup :-

- User chooses password
- Hash of password stored in password file

Authentication

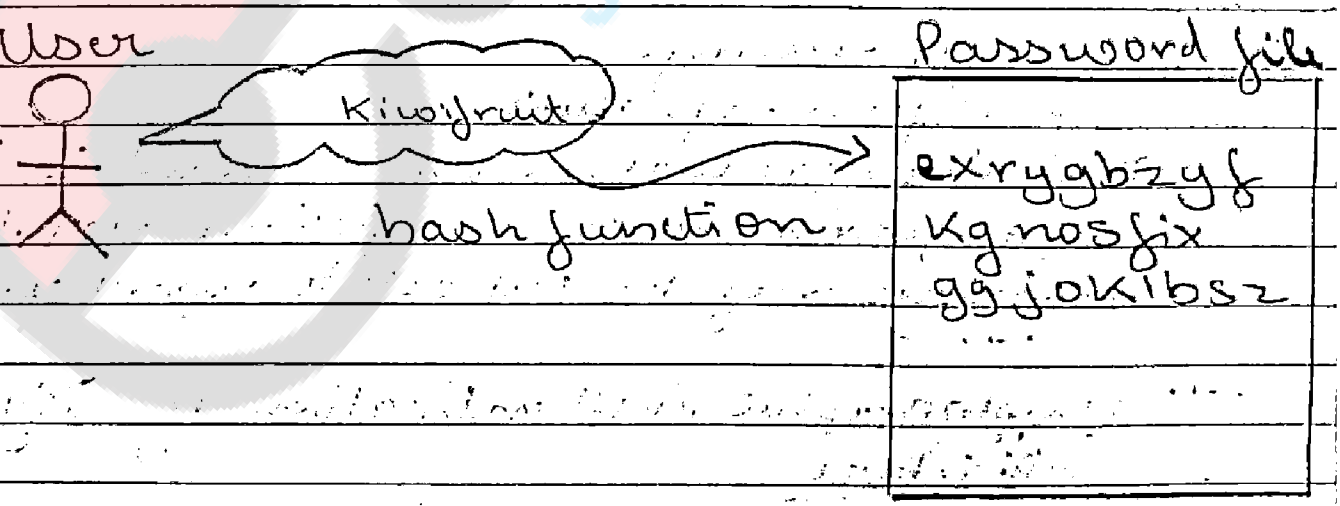
- User logs into system, supplies password
- System computes hash, compares to file

Basic password scheme

- Hash function h: Strings → Strings
- User password stored as h(password) in password file
- When user enters password
  System computes h(password)
  Compares with entry in password file

If password entered by user is equal to password from password file, then user is authenticated.

Else it gives an error saying that username or password is incorrect.



2) Address-based Authentication

- Use the address of the node in the network.
- MAC address or IP address
- Allows only a preconfigured set of MAC or IP address to access the network
- Usually implemented in the switch or router
- Loop holes:
  - Simple one-to-one mapping between a node and a user.
  - So does not really authenticate the user
  - Weak to MAC spoofing and IP spoofing attack.

3) Cryptographic Authentication
A prover can authenticate to a verifier by proving knowledge of a private key. The private key may pertain to any kind of public key cryptosystem:
- Encryption
- Key exchange
- Digital signature

Digital signature is not objected to by any governments, encryption and key exchange may be subject to export controls.

Cryptographic Authentication by Digital Signature
Verifier generates random nonce
A challenge is constructed from material

including the verifiers name.
The provider signs the challenge with the private key.

Digital Signature Cryptosystems
• RSA
- Dual purpose : encryption, signature
• DSA
- Designed by the NSA to provide signature but no encryption
• ECDSA
- Elliptic curve version of DSA

4) Smart Cards
Smart cards are hardware devices that provide a much secure authentication for storing and transferring the important information.
They are of the size of a credit card containing a small chip which stores the private key and a copy of the certificate. A PIN (Personal Identification Number) is used in association with smart card, to provide more secure authentication.

Working of smart cards
The authentication method used in smart card is Challenge Response type of authentication.
When the user inserts his smart card in

a card reader, the program that is stored in the client system asks the user for his unique PIN.
The user enters the PIN and if the PIN is correct the communication between the client application and the smart card starts.
A challenge response procedure takes place between the client and the server. Private key on the card is used to encrypt the data and encrypted data is then transferred to the server.
The public key stored on the server is used to decrypt the data.
If the data gets successfully decrypted, the user is authenticated.

Advantages of Smart Cards

This is a very secure authentication mechanism because:
- the process works on a two-factor authentication - what you know (PIN) and what you have (smart card or private key).
- Brute force attacks and dictionary attacks don't work here as only a limited number of PIN entries are allowed for the smart card holder.

Applications of Smart Cards

Although smart cards are new in India, in many other countries

they are used extensively for applications like:

- Electronic toll collection
- Financial services
- Healthcare services
- Cellular phones
- Set-top boxes
- Secure network access

5) Biometrics

A biometric authentication system attempts to authenticate an individual based on his or her unique physical characteristics. Individuals characteristics can be static characteristics (fingerprints, hand geometry, facial characteristics, and retinal and iris patterns) and dynamic characteristics (voiceprint and signature).

Advantages:

Cannot be disclosed, lost, forgotten

Disadvantages:

Cost, installation, maintenance

Physiological

- Iris
- Fingerprint (including nail)
- Hand (including knuckle, palm, vascular)
- Face
- Voice
- Retina
- DNA
- Even Odor, Earlobe, Sweat pore, lips

Behavioral Patterns
- Signature
- Keystroke
- Voice

Physiological Biometrics
- Fingerprints: A fingerprint is the pattern of ridges on the surface of the fingertip.
Two premises for fingerprint identification
- fingerprint details are permanent
- Fingerprints are unique

Face: Facial characteristics are the most common means of human-to-human identification.
We can define characteristics based on relative location and shape of key facial features, such as eyes, eyebrows, nose, lips, and chin shape.

- Retinal Pattern: The pattern formed by veins beneath the retinal surface is unique and therefore suitable for identifications.

- Hand: Hand geometry systems identify features of the hand, including shape and lengths and widths of fingers.

- Iris: The detailed structure of the iris also a unique physical characteris-

tic in authentication.

Behavioral Biometrics

- Signature: Each individual has a unique style of handwriting and this is reflected especially in the signature, which is typically a frequently written sequence. However, multiple signature samples from a single individual will not be identical. This complicates the task of developing a computer representation of the signature that can be matched to future samples.
- Voice: Voice patterns are more closely tied to the physical and anatomical characteristics of the speaker. There is still a variation from sample to sample over time from the same speaker, complicating the biometric recognition task.

6) Mutual Authentication

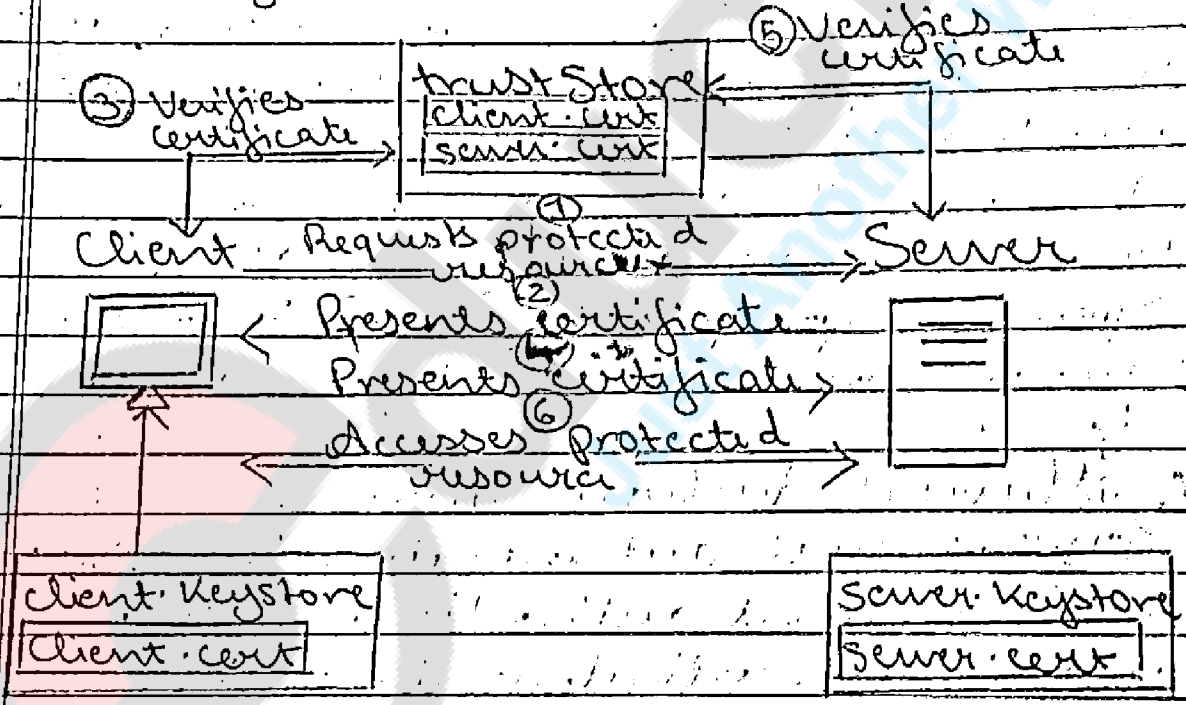
With mutual authentication, the server and the client authenticate each other. Mutual authentication is of two types:

1) Certificate-based
2) Username/password-based

When using certificate-based mutual authentication, the following actions occur:

1) A client requests access to a protected resource.
2) The web server presents its certificate to the client.
3) The client verifies the server's certificate
4) If successful, the client sends its certificate to the server
5) The server verifies the client's credentials.
6) If successful, the server grants access to the protected resource requested by the client.

Certificate-based Mutual Authentication



In user name / password-based mutual authentication, the following actions occur.
1) A client requests access to a protected

resource.

2) The web server presents its certificate to the client.

3) The client verifies the server's certificate

4) If successful, the client sends its username and password to the server, which verifies the client's credentials.

5) If the verification is successful, the server grants access to the protected resource requested by the client.

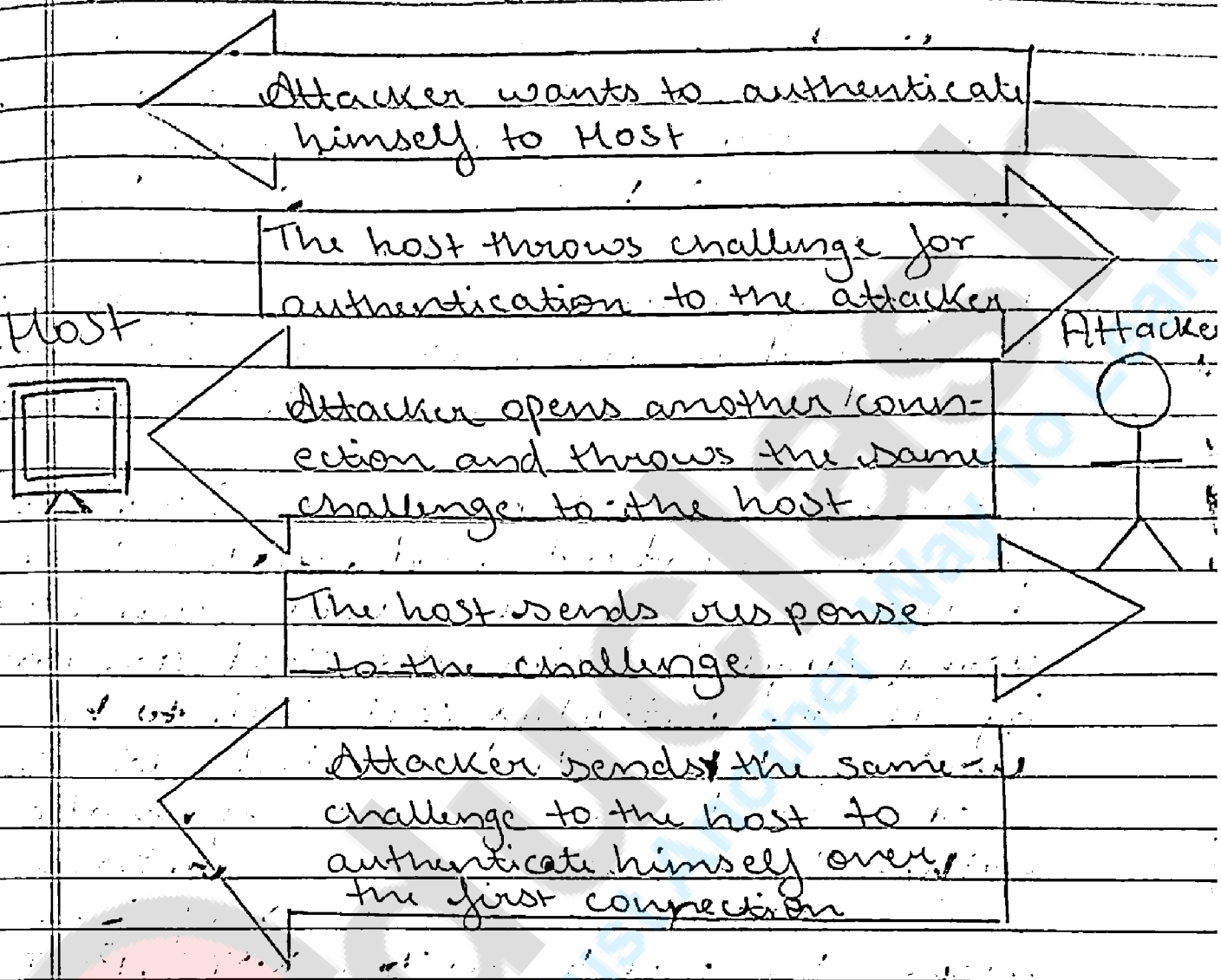
User Name / Password Based Mutual Authentication

(7) Reflection Attack

Two hosts over the network use Challenge - Response - Authentication system to authenticate each other.

In this system, one host transmits a challenge to another host and the other host sends the response back to the first host. If the response matches, the other host is authenticated.
But, sometimes same protocol is used to authenticate hosts in either direction. That is, the same challenge-response protocol is used to authenticate either of the hosts. To authenticate any host, the first host encrypts the challenge C with encryption key K and sends $E(K, C)$ to the other host.
But, as the same challenge-response authentication protocol is used in either direction, the other host can open another connection to the first host and throw the same challenge $E(K, C)$ to the first host. At this point, if the first host sends a response to the challenge to the second host, the second host can use the same response in the first connection and send it back to the first host. As a result, the second host will be able to fraudulently authenticate it to the first host ~~through~~ though it is not authorized to do so. And the attackers use this vulnerability to attack a system and steal data. This is called Authentication Reflection Attack.

Authentication Reflection Attack



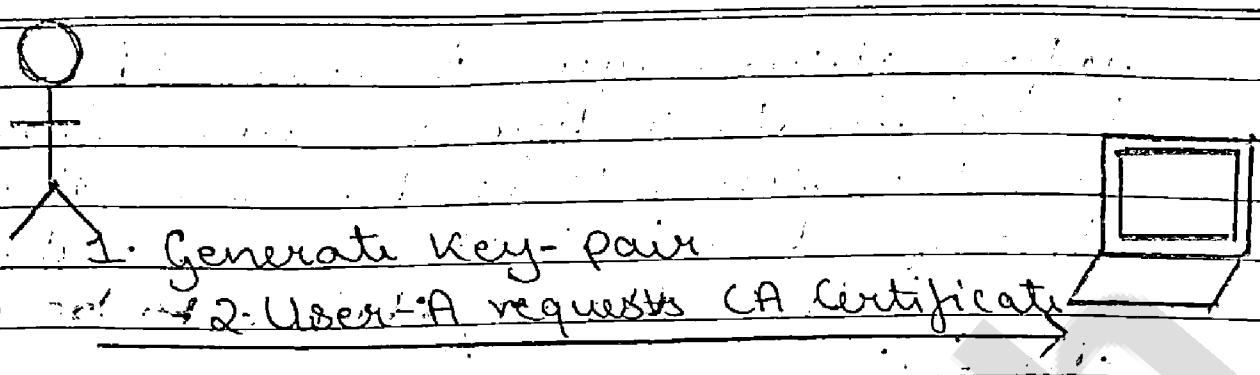
Q) Digital certificate process

A digital certificate is a digital form of identification, like a passport. A digital certificate provides information about the identity of an entity. A digital certificate is issued by a Certification Authority (CA). Examples of trusted CA across the world are Verisign, Entrust, etc. The CA guarantees the validity of the information in the certificate.

In Understanding Digital Signatures article, it was assumed that the receiver knows the Public key of the sender. In fact, the issue of distributing Public key is massive, because the Public key should be distributed in a scalable way as well as be trusted as the true Public key of the sender. These problems are solved when a user obtains another user's Public key from the digital certificate.

Public key Infrastructure (PKI) consists of protocols, standards and services, that allows users to authenticate each other using digital certificates that are issued by CA. For a digital certificate to be useful, it has to be structured in a standard way so that information within the certificate can be retrieved and understood regardless of who issued the certificate. The X.509, PKI X.509 and Public key Cryptography Standards (PKCS) are the building blocks of PKI system that defines the standard formats for certificates and their use.

The process of obtaining a Digital Certificate



1. Generate Key-pair: User-A generates a Public and Private key-pair or is assigned a key-pair by some authority in their organization.

2. Request CA Certificate: User-A first requests the certificate of the CA Server.

3. CA Certificate Issued: The CA responds with its Certificate. This includes its Public Key and its Digital Signature signed using its Private Key.

4. Gather Information: User-A gathers all

information required by the CA server to obtain its certificate. This information could include User-A email address, fingerprints, etc that the CA needs to be certain that User-A claims to be who she is.

5. Send Certificate Request : User-A sends a certificate request to the CA consisting of her Public Key and additional information. The certificate request is signed by CA's Public Key.

6. CA verifies User-A : The CA gets the certificate request, verifies User-A's identity and generates a certificate for User-A, binding her identity and her Public Key. The signature of CA verifies the authenticity of the Certificate.

7. CA issues the certificate : The CA issues the certificate to User-A.

Q) Kerberos

Kerberos is an authentication protocol and a software suite implementing this protocol. Kerberos uses symmetric cryptography to authenticate clients to services and vice versa. For example, Windows servers use Kerberos as the primary authentication mechanism,

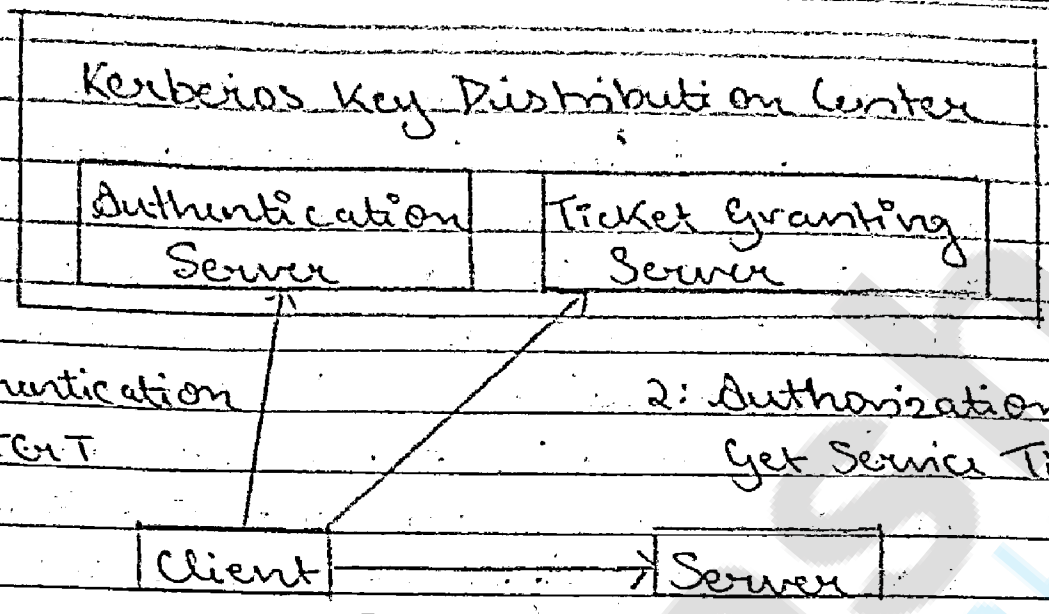
working in conjunction with Active Directory to maintain centralized user information. Other possible uses of Kerberos include allowing users to log into other machines in a local-area network, authentication for web services, authenticating email client and servers, and authenticating the use of devices such as printers.
Kerberos is a protocol for authenticating service requests between trusted hosts across an untrusted network.
The Kerberos protocol uses strong cryptography so that a client can prove its identity to a server (and vice versa) across an insecure network connection. After a client and server has used Kerberos to prove their identity, they can also encrypt all of their communications to assure privacy and data integrity as they go about their business.
Kerberos uses the concept of a ticket as a token that proves the identity of a user. Tickets are digital documents that store session keys. They are typically ~~used~~ issued during a login session and then can be used instead of passwords for any Kerberized services. During the course of authentication, a client receives two tickets :- A ticket-granting ticket (TGT), which acts as a global identifier for a user and a session key - A service ticket, which authenticates a user to

a particular service. These tickets include time stamps that indicate an expiration time after which they become invalid. This expiration time can be set by Kerberos administrators depending on the service.

To accomplish secure authentication, Kerberos uses a trusted third party known as a key distribution center (KDC), which is composed of two components, typically integrated into a single server:

- An authentication server (AS), which performs user authentication
- A ticket-granting server (TGS), which grants tickets to users

The authentication server keeps a database storing the secret keys of the users and services. The secret key of a user is typically generated by performing a one-way hash of the user-provided password. Kerberos is designed to be modular, so that it can be used with a number of encryption protocols, with AES being the default cryptosystem. Kerberos aims to centralize authentication for an entire network - rather than storing sensitive authentication information at each user's machine, this data is only maintained in one presumably secure location.



To start the Kerberos authentication process, the initiating client sends a request to an authentication server for access to a service. The initial request is sent as plaintext because no sensitive information is included in the request. The authentication server retrieves the initiating client's private key, assuming the initiating client's username is in the KDC database. If the initiating client's username cannot be found in the KDC database, the client cannot be authenticated and the authentication process stops. If the clients username can be found in the KDC database, the authentication server generates a session key and a ticket granting ticket. The ticket granting ticket is timestamped and encrypted by the authentication server with the initiating client's password. The initiating client is then prompted for a password; if what is

entered matches the password in the KDC database, the encrypted ticket granting ticket sent from the authentication server is decrypted and used to request a credential from the ticket granting server for the desired service. The client sends the ticket granting ticket to the ticket granting server, which may be physically running on the same hardware as the authentication server, but performing a different role. The ticket granting service carries out an authentication check similar to that performed by the authentication server, but this time sends credentials and a ticket to access the requested service. This transmission is encrypted with a session key specific to the user and service being accessed. This proof of identity can be used to access the requested "Kerberized" service, which, once having validated the original request, will confirm its identity to the requesting system. The timestamped ticket sent by the ticket granting service allows the requesting system to access the service using a single ticket for a specific time period without having to be re-authenticated. Making the ticket valid for a limited time period makes it less likely that someone else will be able to use it later; it is also possible to set the maximum lifetime to 0 , in which case service tickets

will not expire.

Kerberos Advantages:

1) The Kerberos protocol is designed to be secure even when performed over an insecure network.
2) Since each transmission is encrypted using an appropriate secret key, an attacker cannot forge a valid ticket to gain unauthorized access to a service without compromising an encryption key or breaking the underlying encryption algorithm, which is assumed to be secure.
3) Kerberos is also designed to protect against replay attacks, where an attacker eavesdrops legitimate Kerberos communications and retransmits messages from an authenticated party to perform unauthorized actions.
- The inclusion of timestamps in Kerberos message restricts the window in which an attacker can retransmit messages.
- Tickets may contain the IP addresses associated with the authenticated party to prevent replaying messages from a different IP address.
- Kerberized services make use of "replay cache", which stores previous authentication tokens and detects their reuse.
4) Kerberos makes use of symmetric encryption instead of public-key encryption, which makes Kerberos computa-

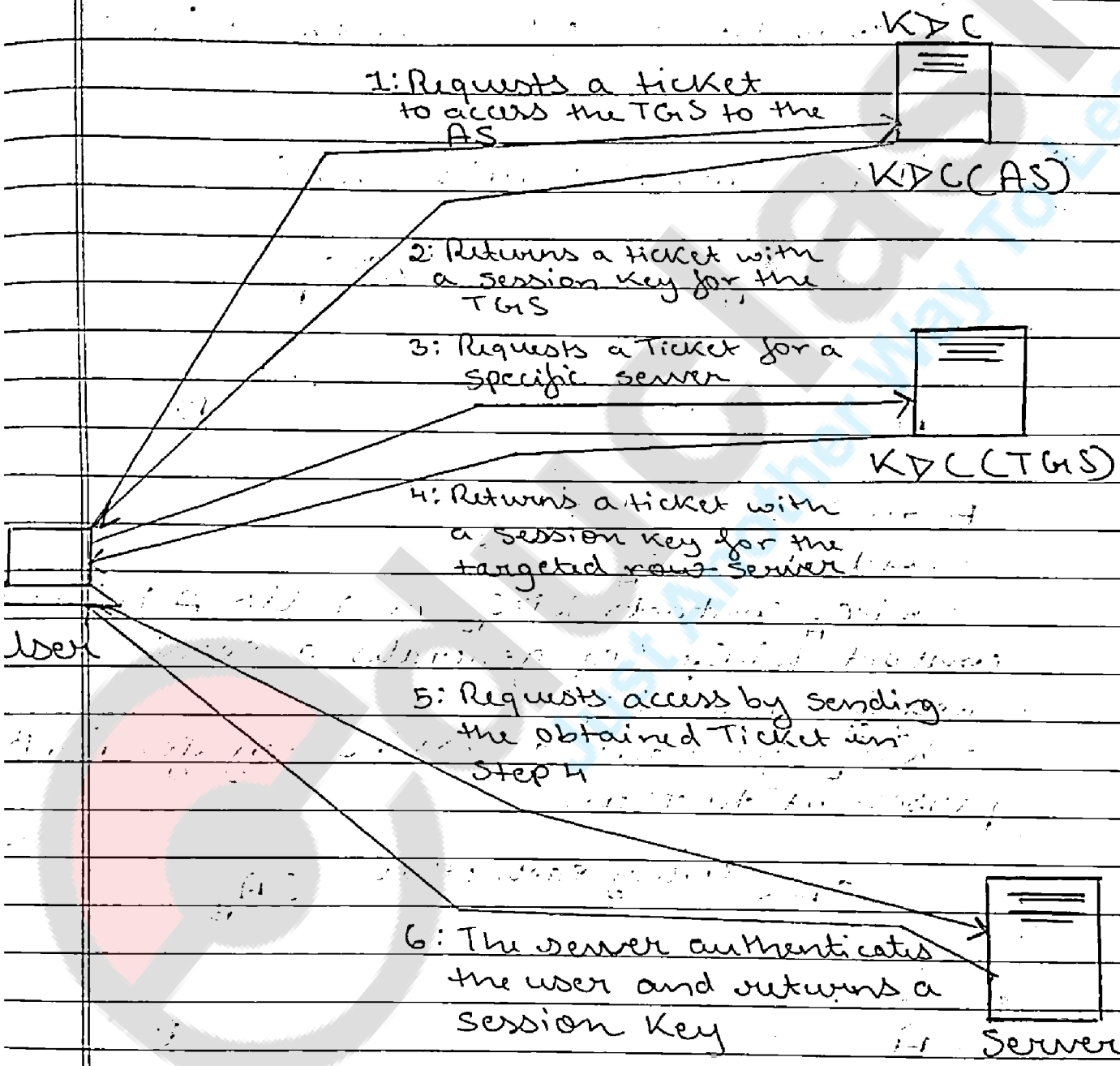
tionally efficient.
5) The availability of an open-source implementation has facilitated the adoption of Kerberos.

Kerberos Disadvantages
1) Kerberos has a single point of failure: if the key Distribution Center becomes unavailable, the authentication scheme for an entire network may cease to function. Larger networks sometimes prevent such a scenario by having multiple KDCs or having KDCs available in case of emergency.
2) If an attacker compromises the KDC, the authentication information of every client and server on the network would be revealed.
3) Kerberos requires that all participating parties have synchronized clocks, since time stamps are used.

Q) KDC - working
The KDC contains three components:
1) an Authentication Service
2) a ticket Granting Service
3) a database (Apache DS)
The KDC role is to authenticate users and distribute tickets based on the information stored in its database.
The Apache Kerberos Server contains

all these three components and hence is a KDC.

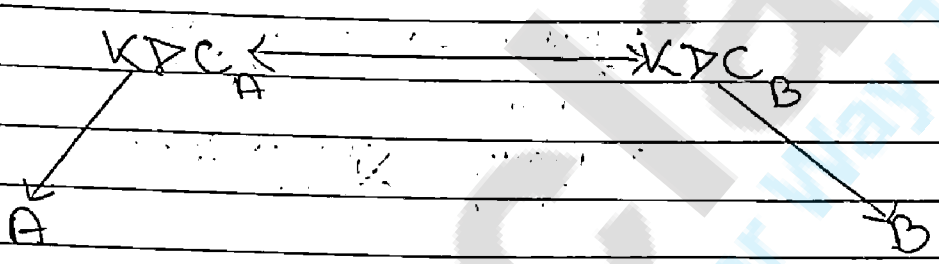
The following diagram explains how the KDC works:



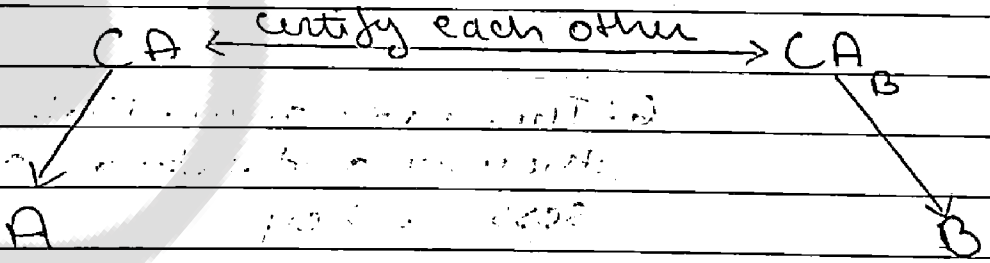
In order to use a service, the client needs to get a ticket for this service from the KDC. This requires a two step process,

where the client first authenticates himself and then get back a ticket to use with the targeted server.
Though the Authentication and Ticket Granting Service look like running in separate servers, a single Kerberos server implementation often contains both.

Multiple Domains with KDC



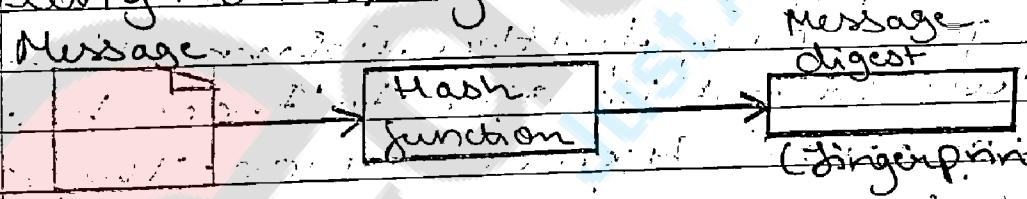
A to talk to B:
- contacts KDC_A
- KDC_A contacts KDC_B, or tells A how to contact KDC_B (eg generates a session key for A & KDC_B)
- KDC_B generates a session key for A & B, passes it to them



A, to authenticate the public key of B,
- verifies B's cert. issued by CA_B,
- verifies CA_B's cert issued by CA_A,

B does vice versa to authenticate A's key

Q] Message Digest :-
Public-key encryption is efficient if the message is short.
Using a public key to sign the entire message is very inefficient if the message is very long.
The solution is to let the sender sign a digest of the document instead of the whole document. The sender creates a ~~miniature~~ miniature version or digest of the document and signs it; the receiver then checks the signature on the miniature.
To create a digest of the message, we use a hash function. The hash function creates a fixed-size digest from a variable-length message.

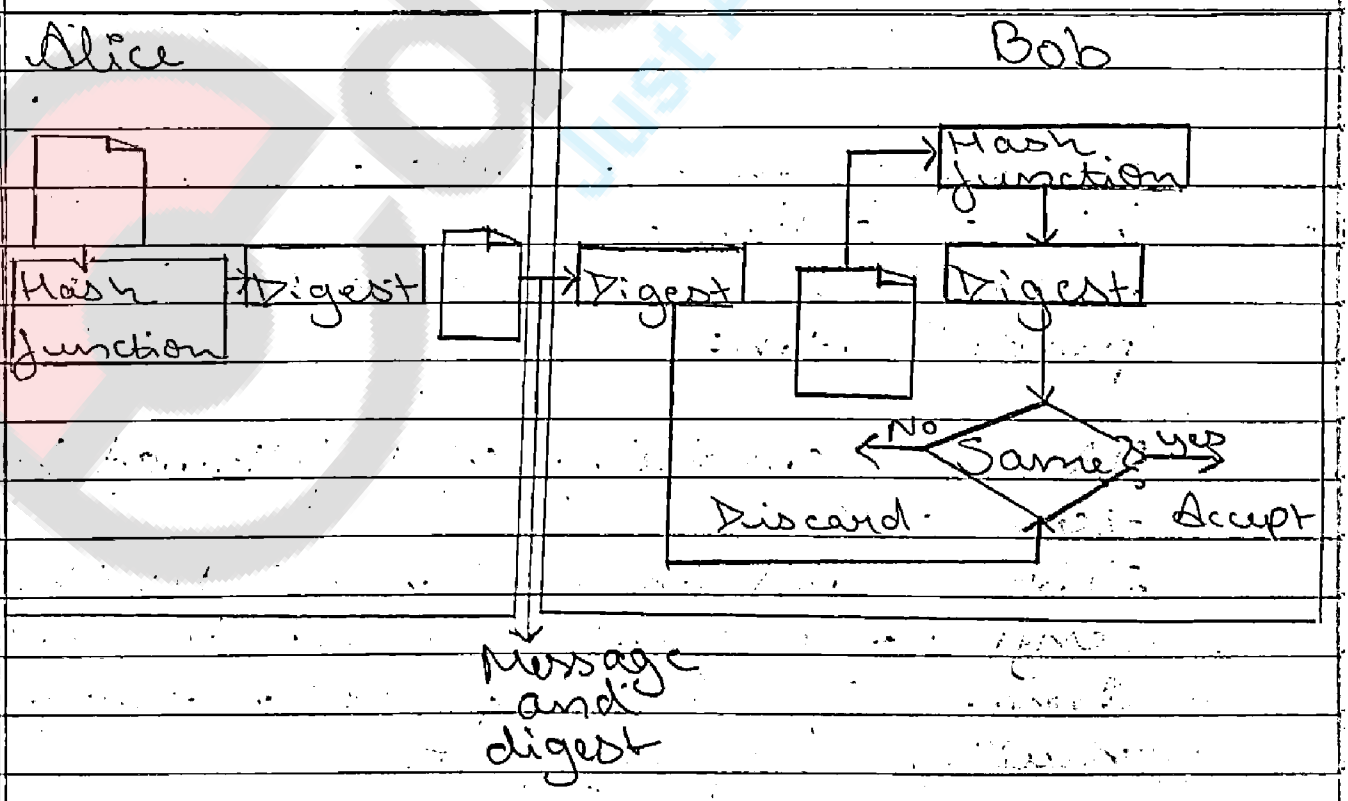


The two most common hash functions are called MD5 (Message Digest 5) and SHA-1 (Secure Hash Algorithm 1). The first one produces a 120-bit digest. The second produces a 160-bit digest.
Note that a hash function must have two properties to guarantee its success. First, hashing is one-way; the digest

can only be created from the message not vice versa.
Second, hashing is a one-to-one function; there is little probability that two messages will create the same digest. After the digest has been created, it is encrypted (signed) using the sender's private key. The encrypted digest is attached to the original message and sent to the receiver.
Suppose that we have a number 4000 and we divide it by 4 to get 1000. Thus, 4 can become a fingerprint of the number 4000. Dividing 4000 by 4 will always yield 1000. If we change either 4000 or 4, the result will not be 100
Another important point is, if we are simply given the number 4, but are not given any further information, we would not be able to trace back the equation $4 \times 1000 = 4000$. Thus, we have one more important concept here. The fingerprint of a message (in this case, the number 4) does not tell anything about the original message (in this case, the number 4000). This is because there are infinite other possible equation, which can produce the result 4.

We can summarize the requirements

of the message digest concept, as follows:
1) Given a message, it should be very easy to find its corresponding message digest. Also for a given message, the message digest must always be the same.
2) Given a message digest, it should be very difficult to find the original message for which the digest was created.
3) Given any two messages, if we calculate their message digests, the two message digests must be different.
Another basis of message digest is that it should not give any clue or indication of the original message i.e it should not be possible to revert back to original message from the digest. Also, for a given message its digest should be the same always.

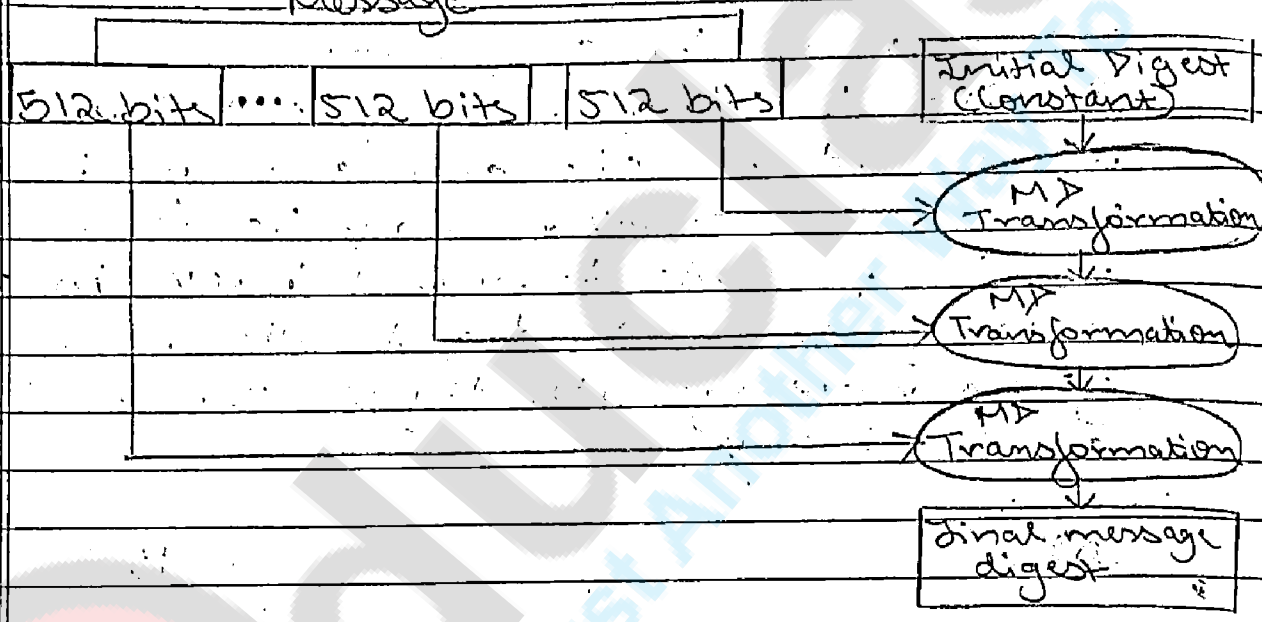


9] MD5:-

There are a number of popular message digest algorithms known as MDn for various values of n.

MD5 is the most popular and is fifth in a series of message digests designed by Ronald Rivest.

The basic operation of MD5 is shown in the diagram.



This algorithm operates on message 512 bits at a time.

Messages not multiple of 512 bits are padded with:

1) A string consisting of 1 followed by zeros, and

2) 64-bit integer that indicates the length of original message, to make the length of the composite message multiples of 512 bits.

The message digest calculation begins with a digest value initialized to a constant. This value is combined with the first 512 bits of the message to produce a new value for the digest.
This new value is then combined with the next 512 bits of the message using the same transformation.
This process is repeated on each 512-bit block till the final value of digest is obtained from the last block of the message.
The digest is 128-bit long for any message length.

g) SHA
SHA is a hash algorithm developed and published by the collaboration of NIST and NSA in 1993 as a Federal Information Processing Standard. SHA1 was the revised version of SHA published in 1995. However SHA1 is relatable to MD5 as it is based on MD5.
The SHA-1 can take any arbitrary message as an input which is 2^{64} bits in length and produce 160-bit long message digest. SHA stands for Secure Hash Algorithm where secure signifies the one-way property and inability to produce a similar message from two messages. Here, one-way means

that the one can not obtain the original message with the help of the message digest of that message.

SHA 1 Working

The step by step process is as follows:-

1) Padding :- This step adds the padding to the end of the genuine message in such a manner that the length of the message is 64 bits less of the multiple of 512.

2) Append :- The length of the message is computed devoiding the padding bits and appended to the end of the padding as a 64-bit block.

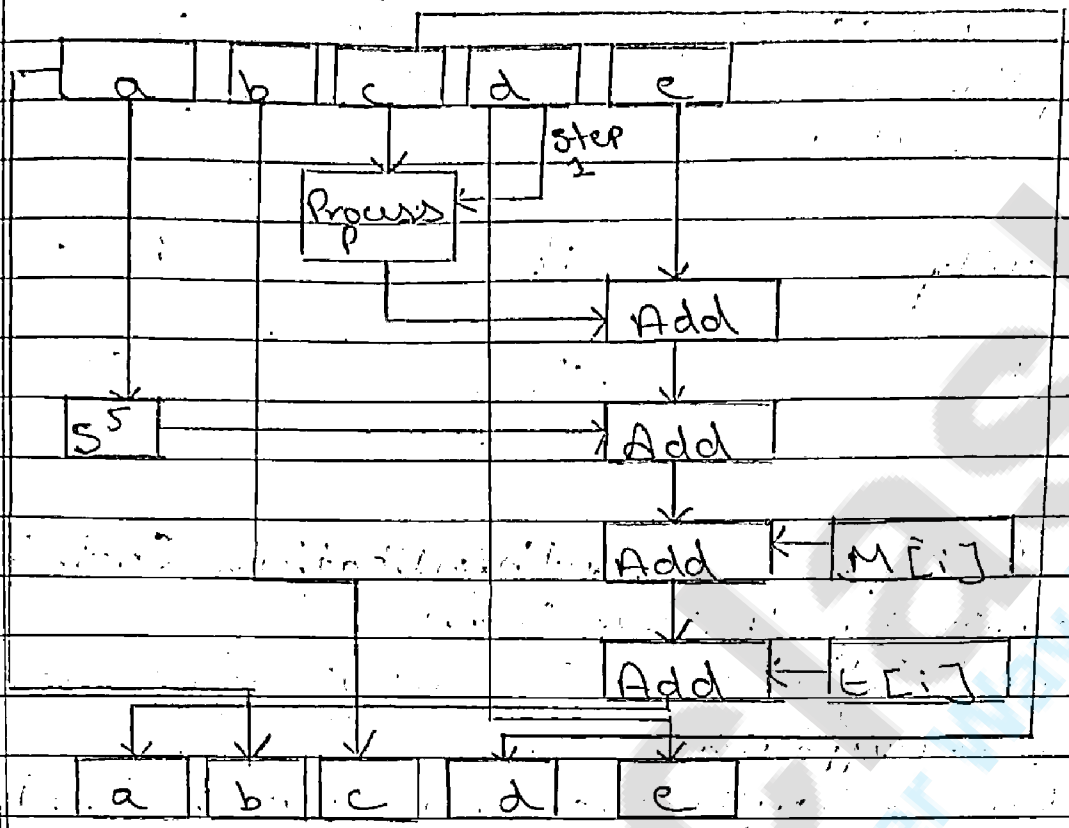
3) Division of input - Divide the input into 512-bit blocks.

4) Initialize chaining variable - Here 5 chaining variables are initialized A, B, C, D and E each of 32-bits total of 160-bits.

5) Process blocks - This step includes copying of chaining variables, division of 512 bit-block into 16 sub-blocks, and processing 4 rounds of 20 step each.

However, in SHA 1 there are four rounds and each round consist of 20 steps in this each round takes the current 512-bit block, the register abcde and constant K[t] (where t=0 to t=9) as

the three inputs.



Single SHA-1 Iteration

97 makes 80 iterations.

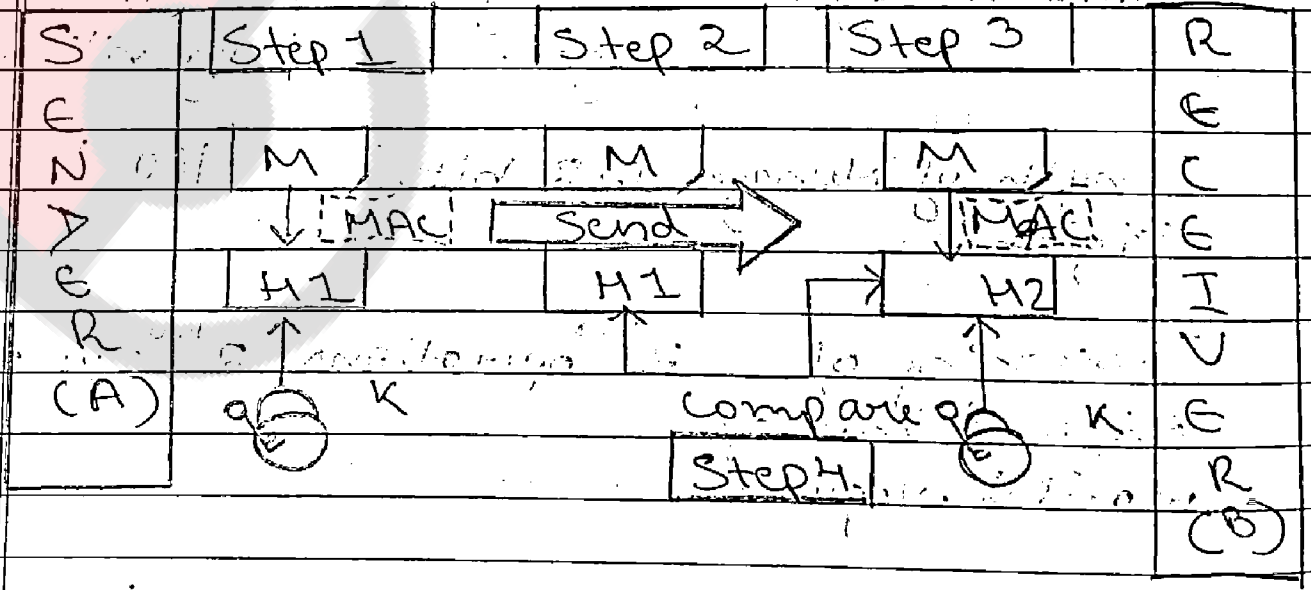
Difference between MD5 and SHA 1

Basis for comparison	MD5	SHA 1
Stands for	Message Digest	Secure Hash Algorithm
Length of Message Digest	128 bits	160 bits
Discerning of original message would require	2^{128} operations	2^{160} operations

for finding two messages generating the same message digest	2^{64} operations would be needed	2^{80} operations are required
Security	Poor	Moderate
Speed	Fast	Slow

(5) MAC (Message Authentication Code)
- Similar to Message Digest
- Shared Symmetric (Secret) key is used for encryption
- Message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)

Message Encryption



- MAC generation of message using shared symmetric (secret) key
- sends original message and MAC (H1)
- At receiver end, it receives original message and MAC
- Receiver calculate MAC (H2) using key and original message
- Compare H1 & H2
  - If H1 != H2 then, Message altered
  - If H1 == H2 then, Message not changed

MAC is generated by an algorithm that creates a small fixed-sized block
- depending on both message and some key
- like encryption though need not be reversible
- appended to message as a signature
- receiver peforms same computation on message and checks it matches the MAC
- provides assurance that message is unaltered and comes from sender

- MAC provides confidentiality
- can also use encryption for secrecy
  - generally use seperate keys for each
  - can compute MAC either before or after encryption
  - is generally regarded as better done before

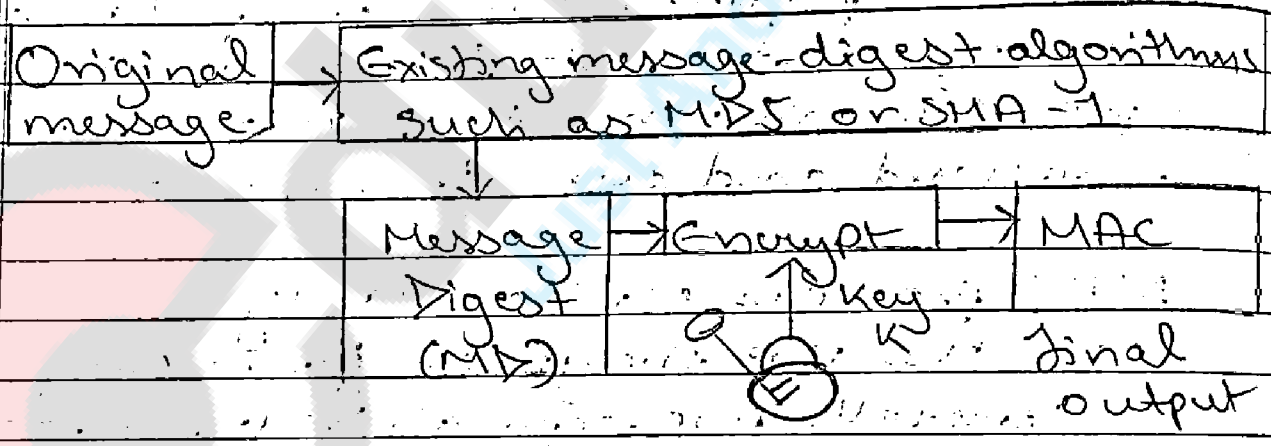
Why use a MAC?
- Sometimes only authentication is needed
- Sometimes need authentication to persist

longer than the encryption (eg: archival use)
- Note that a MAC is not a digital signature

Q] HMAC

HMAC stands for Hash Message Authentication Code
- Mandatory for security implementation for Internet Protocol security.
- Idea of HMAC is to reuse existing Message-Digest algorithms (such as MD5, SHA-1 ...)
- Uses shared symmetric key to encrypt message digest.

HMAC Concept



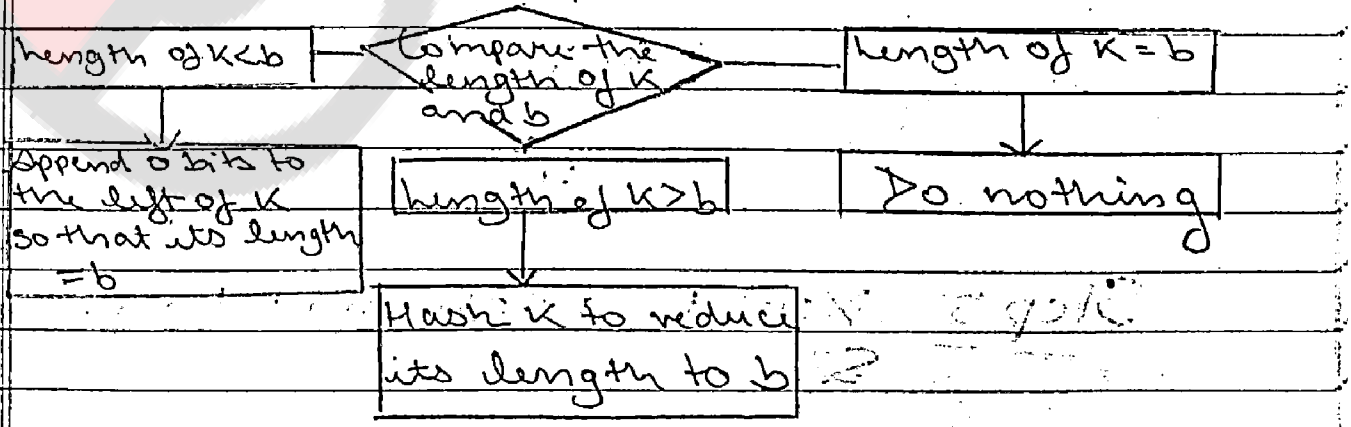
Working of HMAC:
- Variables used in HMAC
  MD = the message digest/ hash function used (eg MD5, SHA-1, etc)
  M = the input message whose MAC is to be calculated
  L = the number of blocks in the

message M.
b = the numbers of bits in each blocks.
K = the shared symmetric key to be used in HMAC.
ipad = A string 00110110 repeated b/8 times
opad = A string 01011010 repeated b/8 times.

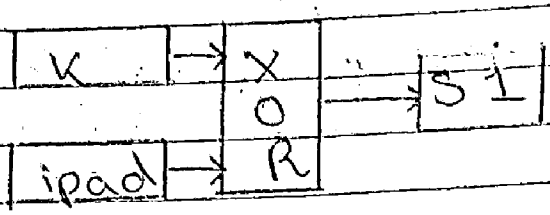
Working of HMAC

Step 1 : Make the length of K equal to b.
Step 2 : XOR K with Ipad to produce S1.
Step 3 : Append M to S1
Step 4 : Message-digest algorithm.
Step 5 : XOR K with opad to produce S2.
Step 6 : Append H to S2.
Step 7 : Message-digest algorithm

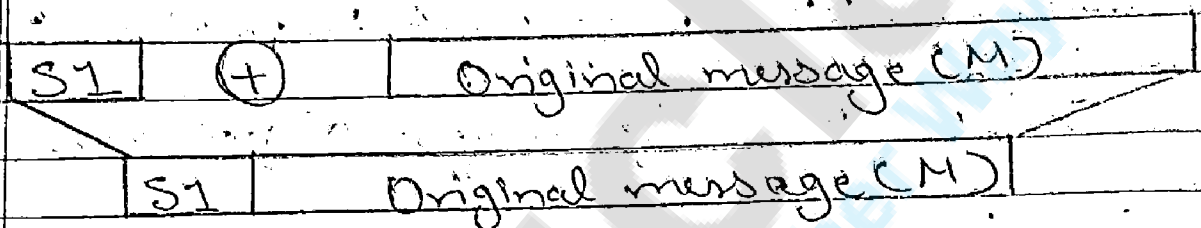
Step-1 Make the length of K equal to b
If length of K < b : add 0 bit as required to the left of K
If length of K = b : In this case, we do not take any action, and proceed to step 2.
If length of K > b : we need to trim K, for this, we pass K through the message-digest algorithm (H) selected for this particular instance of HMAC.



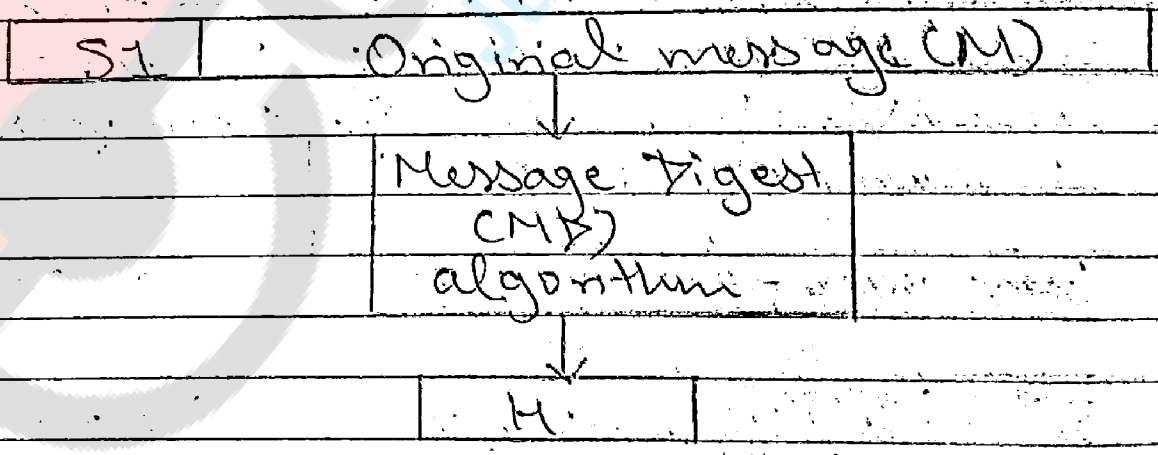
Step-2 XOR K with ipad to produce S1
· XOR K (the output of Step-1) and ipad to produce a variable called S1



Step-3 Append M to S1
Take the original message (M) and simply append it to the end of S1.

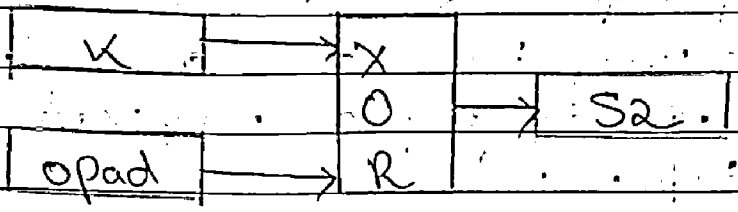


Step 4 Message-digest algorithm
The selected message-digest algorithm (eg. MD5, SHA-1, etc) is applied to the output of step 3.



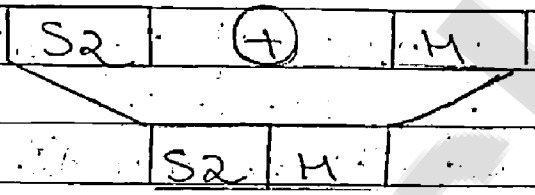
Step 5 XOR K with opad to produce S2

XOR K (the output of step 1) with opad to produce a variable called as S2.



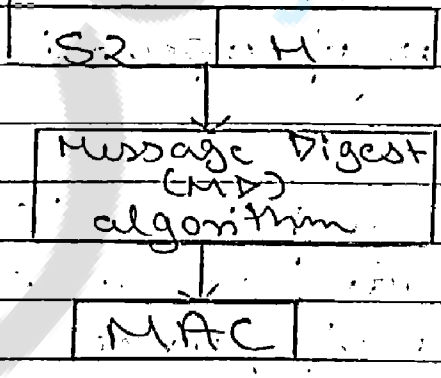
Step 6 Append H to S2

Append the message digest calculated in step 4 to the end of S2.



Step 7 Message digest algorithm

The selected message-digest algorithm (eg MD5, SHA-1, etc) is applied to the output of step 6 (i.e to the concatenation of S2 and H). This is the final MAC that we want.



Its advantages of HMAC

1) Key exchange is main issue
2) Somehow the key-exchange problem

is resolved, HMAC cannot be used if the number of receivers is greater than one.

3) If multiple parties share the same symmetric key. How does a receiver know that the message was prepared and sent by the sender

4) Replay of Message