6 Authentication Scheme

Types of Authentication Scheme / services
1. Message Authentication service
2. Digital signature service

1. Message Authentication Scheme
↳ It is the process to verify that received message are coming from alleged source and havenot been altered in transit. They may also verify sequencing (any modification to a sequence of message) and timeless (delay or replay of message).

2. Digital signature
↳ Also indicates the measures to counter non-repudiation by source or destination.
↳ There is an esenhal difference between digital signature and authentication:

Authentication:
↳ To protect two communicating parties from a masquerading third party (Eve as Alice or Bob) or modification of message in transit.
↳ But authentication can't help Bob or Alice if they donot trust each other.

A digital signature is the solution of the problem.

→ Authentication is concerned with
- Protecting integrity of message
- Validating identity of origin
- Non-repudiation of services

Alice's digital signature on message reassures Bob that it indeed came from Alice and Alice cannot deny sending this message on later time.

Any message authentication or digital signature mechanism can be viewed as having two fundamental levels.

At lower level
→ There must be some sort of function that produce authentication (a value to be used for authentication message)

At higher level
→ The lower level function is used as primitive that enables the receiver to verify authenticity of message.

The three classes of function that can produce authentication are:
1. Message Authentication Code (MAC)
2. Hash function
3. Message Encryption.

Message Authentication Code (MAC)

↳ Involves the use of a secret key to generate a small fixed size block of data aka cryptographic ek. checksum or MAC that is appended to a message.

↳ Assumes that the communication parties A & B share common key 'K' when A has to send a message to B, it calculates the MAC as a function of message and the key.

i.e.

$$MAC = C_K(M)$$

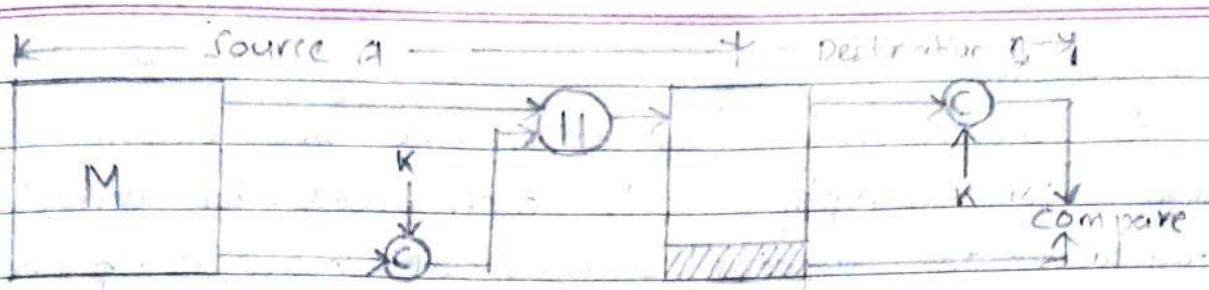
where, M = i/p message
K = shared key
C = MAC function

↳ The message plus MAC are transmitted to the intended receipient. The receipient performs some calculation on the received message using the shared key to generate new MAC. The received MAC is compared to the new MAC, if they are equal the message is considered to be authenticated / authentic.

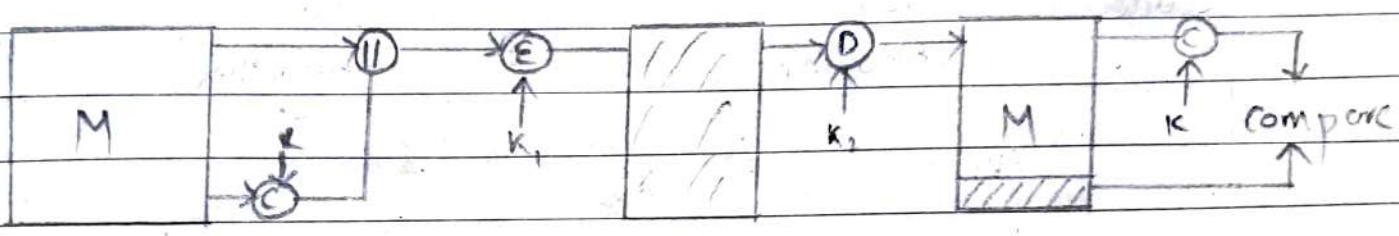
↳ A MAC function is similar to the encryption. One difference is that MAC algorithm need not to be reversible as it must for decryption.

↳ In general, the MAC is many to one function.

↳ The various MAC authentication are shown below:

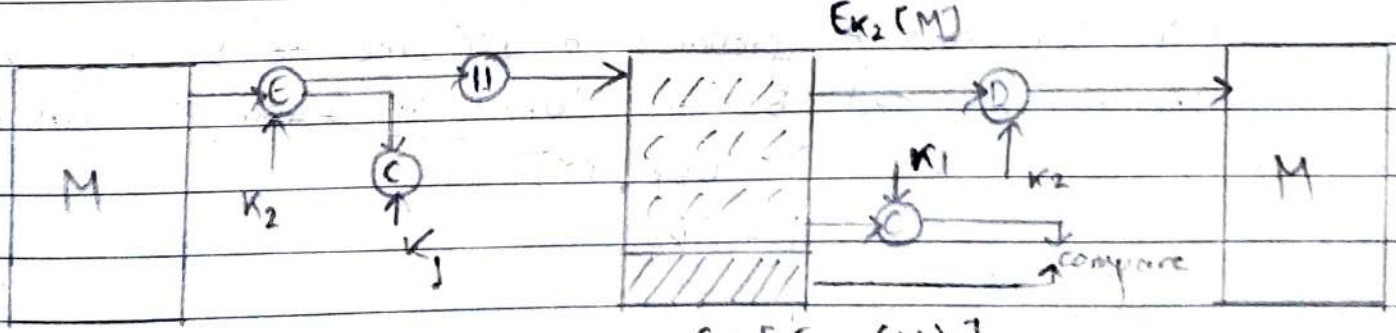


a) Message Authentication



$E_{K_1}(M \| C_K(M))$

b) Message Authentication and confidentiality Authentication tied to plain text



$C_{K_1}[E_{K_2}(M)]$

c) Message Authentication & Confidentiality Authentication tied to cipher text

Authentication:

As mentioned earlier, either the key element may be used for encryption with other used for decryption. This facilitates on different schemas as shown in figure below:

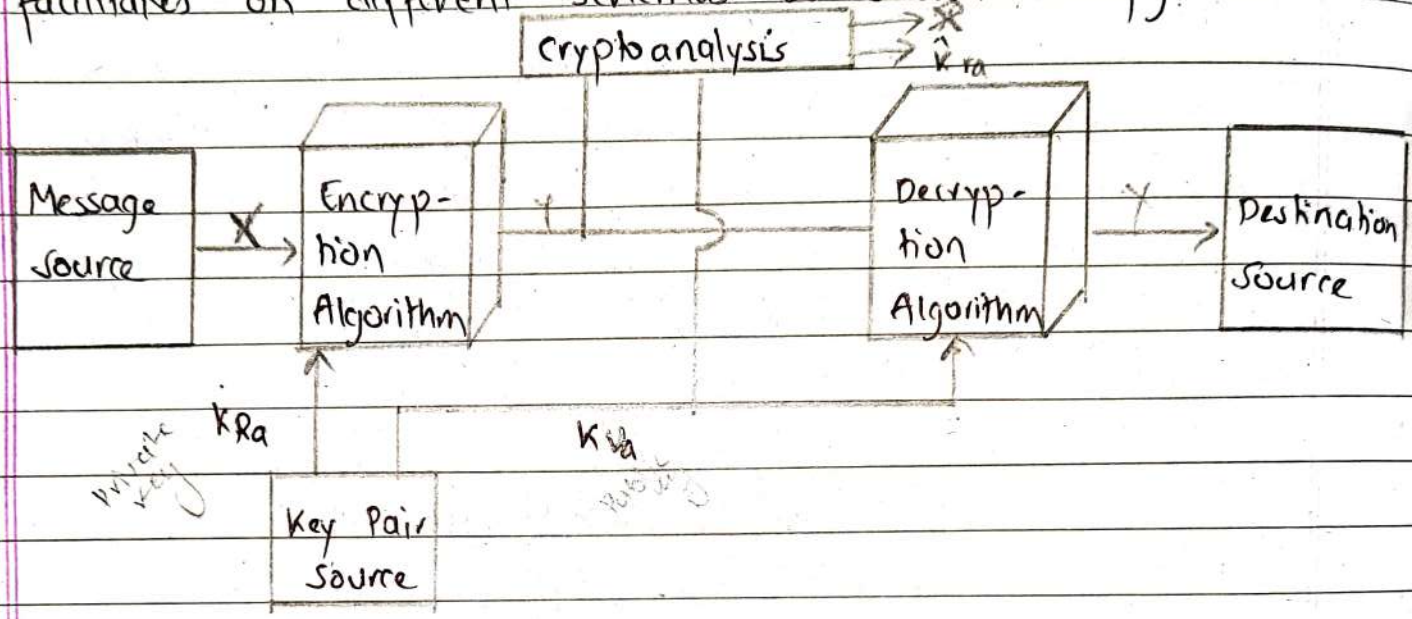


Fig: P-K Authentication

In this case, A prepares a message to B using his private key and B can decrypt it using A's public key

$$Y = E_{KR_a}(X)$$

$$X = D_{KU_a}(Y)$$

As the message was prepared using A's private key. It could have only come from A, therefore entire message serves as a digital signature.

It doesnot provide confidentiality as everyone knows A's public key. Also, the scheme is not as efficient

as B must maintain. Cipher text as proof of authenticity and decoded text for partial use of the document.

A more efficient way of achieving the same result is to encrypt small block of bits that are function of bits that come from the document. This block is called authentication and it must have the property that is infeasible to change the document without changing the authenticator.

If the authentication is encrypted using sender's private key then it serves as a signature that verifies origin, content and sequencing of document.

Authentication and Confidentiality

If both are required, the double use of public key scheme facilitates the process.

Hence, $Y = E_{KU_b}[E_{KR_a}(X)]$

$X = D_{KU_b}[D_{KU_a}(Y)]$

In this case, the message is first encrypted with private key of A which provides the digital signature and encryption is performed using receiver's public key ensuring confidentiality. The main disadvantage with this scheme is the public key algorithm which is relatively complex that must be 4 times.

$$Y = E_{KU_b}\left(E(KR_a(X))\right)$$

$$X = D_{KU_a}\left(D(KR_b(Y)\right)$$

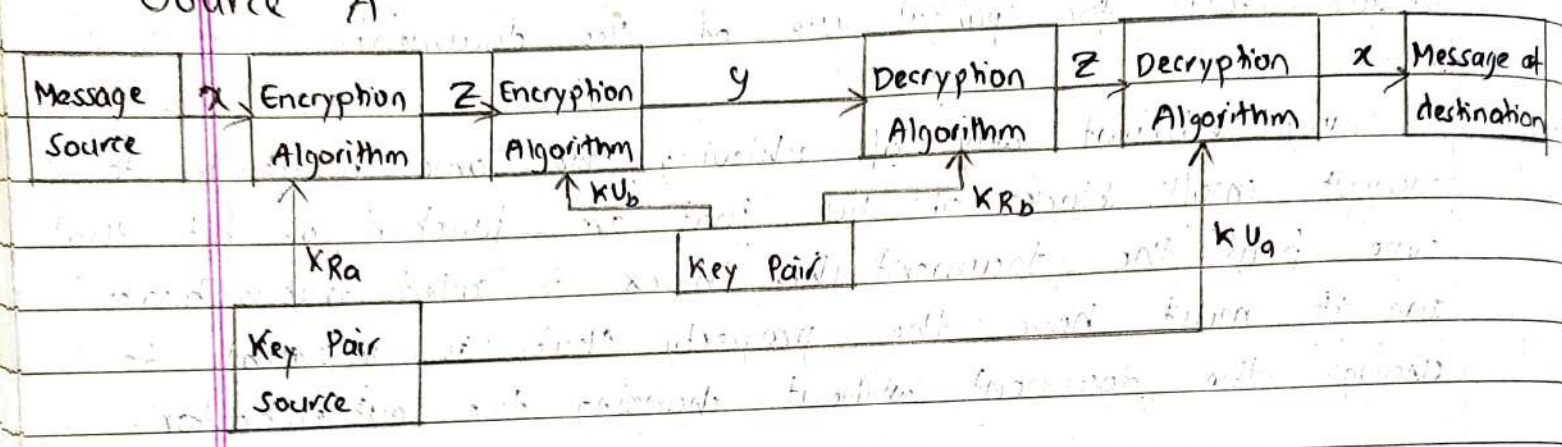


Fig: P-K Cryptography Confidentiality & Authentication

Application of Algorithm

In broad terms, we can classify the ~~public~~ use of P-K cryptosystem in three categories:

1. Encryption and Decryption; where the sender encrypts with receiver public key.
2. Digital signature; where the sender signs a message with his private key.
3. Key Exchange Algorithm like Diffie Hellman Algorithm and others.

However, all the algorithm may not be suitable for the above mentioned application. Some can only be used for digital signature. It may be noted that RSA algorithm can be used for all 3 applications.

Requirement of Algorithms:

The requirement of public key cryptosystem are laid out by Diffie Hellman. It is comp

1. It is computationally easy for the party B to generate a key pair (K_U, K_R).
2. It is computationally easy for the sender A knowing public key of B ie. K_{U_b} and encrypt the message to generate the ciphertext using $E_{K_{U_b}}(M)$

   ie.

   public key of B $\rightarrow K_{U_b}$

   ciphertext $= E_{K_{U_b}}(M)$

3. It is computationally easy for receiver B to decrypt the ciphertext using his private key (K_{R_b}) to recover the digital message.

   ie. Private key of B $\rightarrow K_{R_b}$

   $M = D_{K_{R_b}}(C.T)$

   $= D_{K_{R_b}}(E_{K_{U_b}}(M))$

4. It is computationally infeasible for an opponent (eaves dropper) knowing the public key, K_{U_b} to determine K_{R_b}.

5. It is computationally infeasible for an opponent knowing K_{U_b} and ciphertext (CT) to uncover M.

6. Although useful (not necessary for all P-K application), the encryption and decryption can be applied in any order.

For example,

$$M = D_{KR_b}\left[E_{KU_b}(M)\right]$$

OR

$$M = E_{KU_b}\left[D_{KR_b}(M)\right]$$

These requirements derive to need a trap door or one way function.

A one way function is a function that maps every functional value to a unique value with the condition that the calculation of the function is easy and the calculation of inverse is infeasible.

Hash Function

A variation of on the message authentication code is the one way hash function. A hash function with MAC accepts a variable size message M as input and produces fixed size output refer to as hash code H(M). Unlike MAC, hash code doesnot use the key but is a function only of the input message; the hash code is also called hash value or message digest.

There are varieties of way in which hash function code can be used to provide message authentication as disiussed below:

1. The message plus hash code is encrypted with symmetric encryption. This is identical to internal error control strategy because encryption is applied to the entire message and hash code confidentiality is also provided.

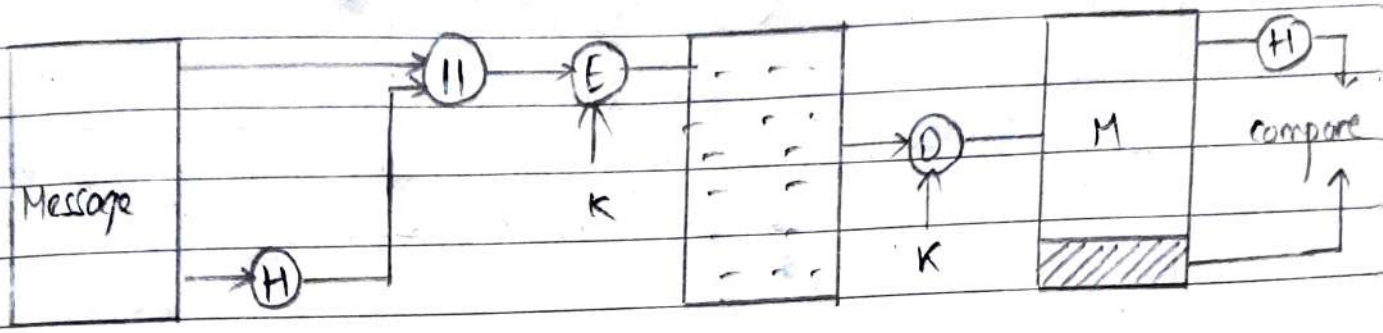


Fig (a).

2. Only the hash code is encrypted using symmetric key encryption. This reduces the processing burden for those application that do not require confidentiality.

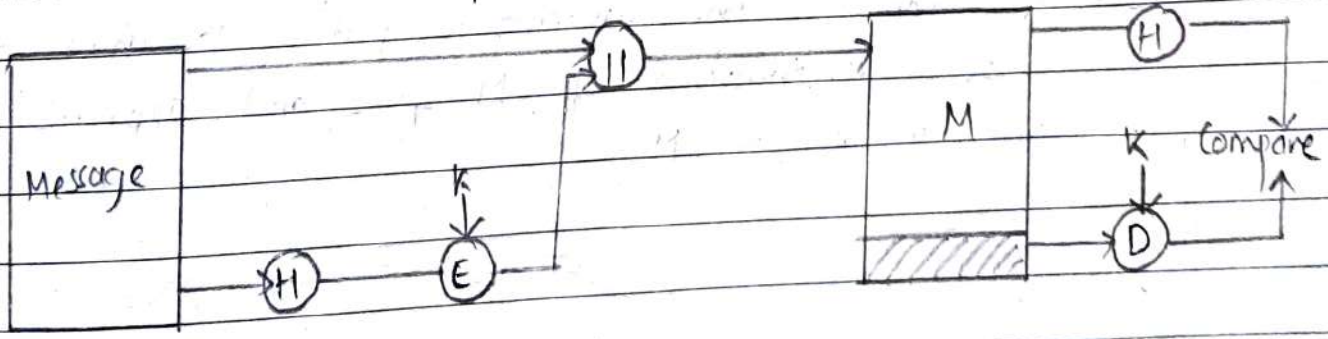
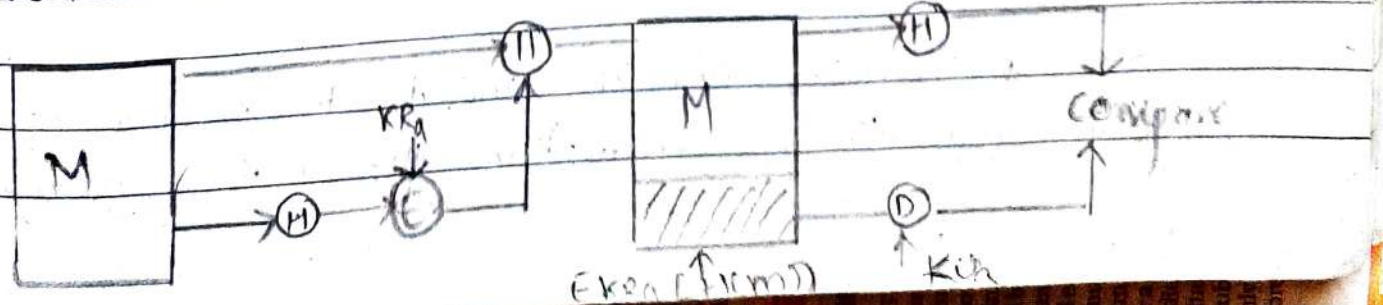


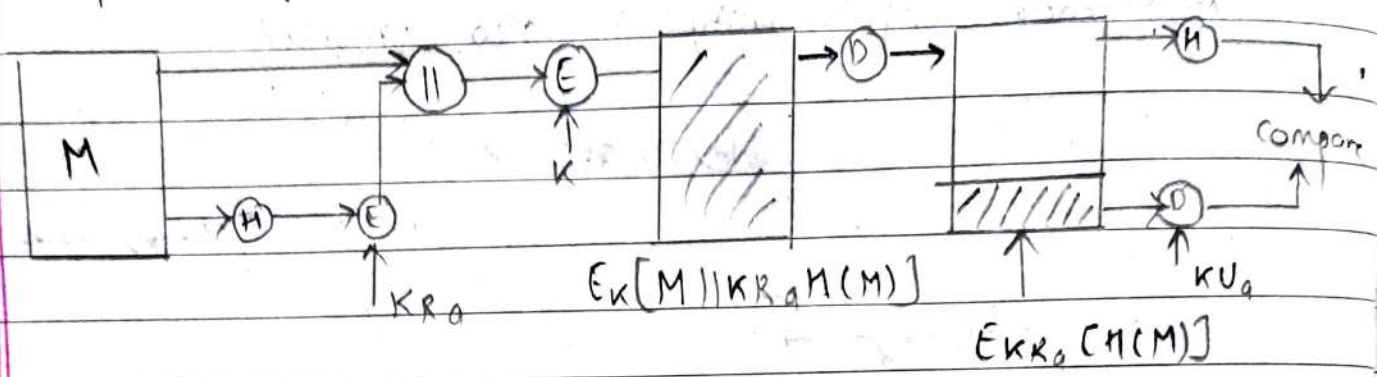
Fig (b) Only Authentication (A)

3. Hash code is encrypted using P-K encryption.

Fig (c) Only (A) with P-K

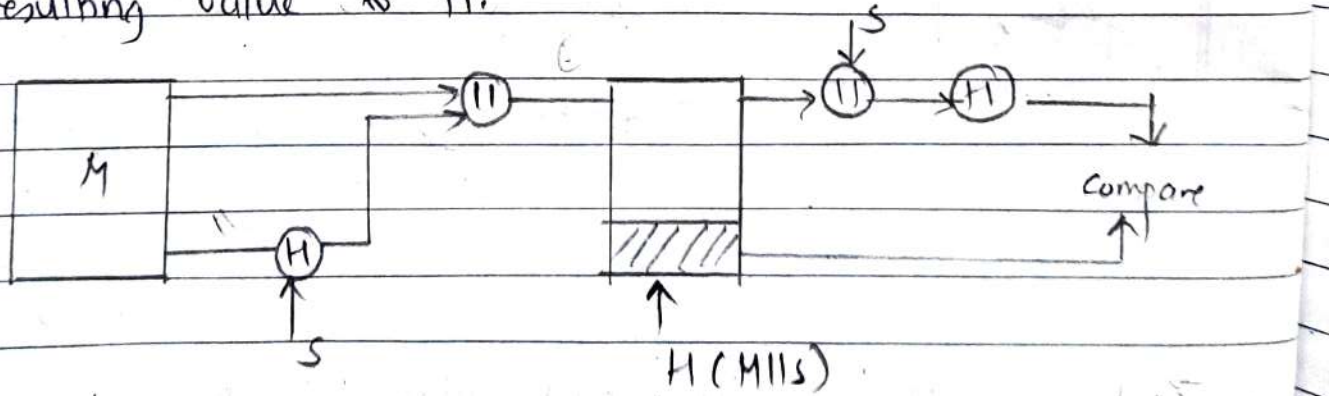


4. Message plus hash code encrypted with P-K are encrypted with S-K encryption which provides confidentiality.



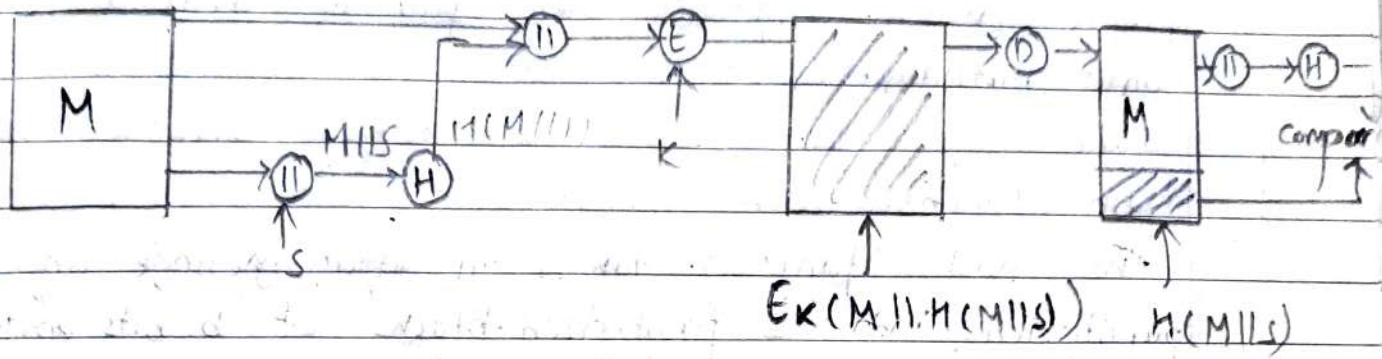
Fig(d): Message plus hash function P-K encryption again encrypted with symmetric key.

5. This technique uses a hash function but no encryption technique of message authentication. This technique assumes that two communicating parties share a common secret value 's'. The source computes the hash function value over continuation of M & s and appends the resulting value to M.



6. Confidentiality can be added to previous approach in step (5) by encrypting the entire message

plus the hash code.

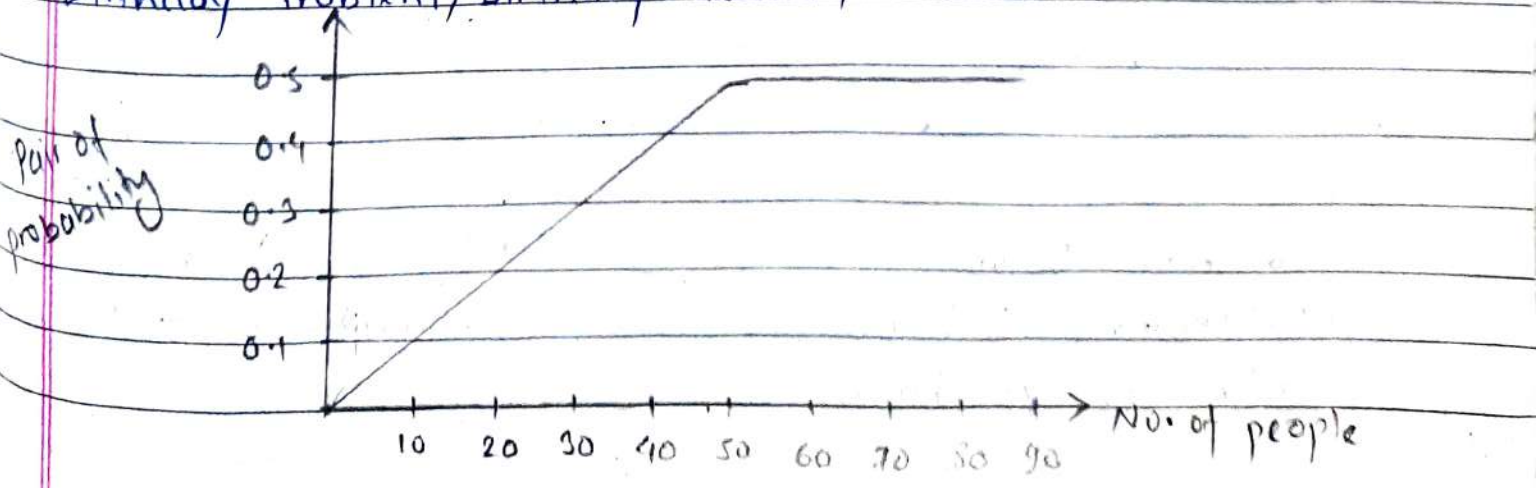


A hash function satisfying the five properties in the above preceeding list is referred to as a weak hash function and the 6th property satisfies, then it will be known as strong hash function which prevent against attacks birthday attacks.

Birthday Attack

It is the cryptographic attack exploiting the mathematics behind the birthday problem in probability theory.
Birthday attack can be used to abuse communication between two or more properties.

Birthday Problem / Birthday Paradox



Birthday paradox is a probability that in a set of n people chosen randomly, some pair of them have the same birthday.

Hash Function

The hash function takes an input message and partitioned into L fixed-sized blocks of 'b' bits each and if it is necessary, the final block is also padded with b-bits. The final block also includes total length of input to hash function. This inclusion makes the job of opponent difficult. Either the opponent must find two equal message of equal length that hash to same values or message of different length that together with their length value hash to the same values.

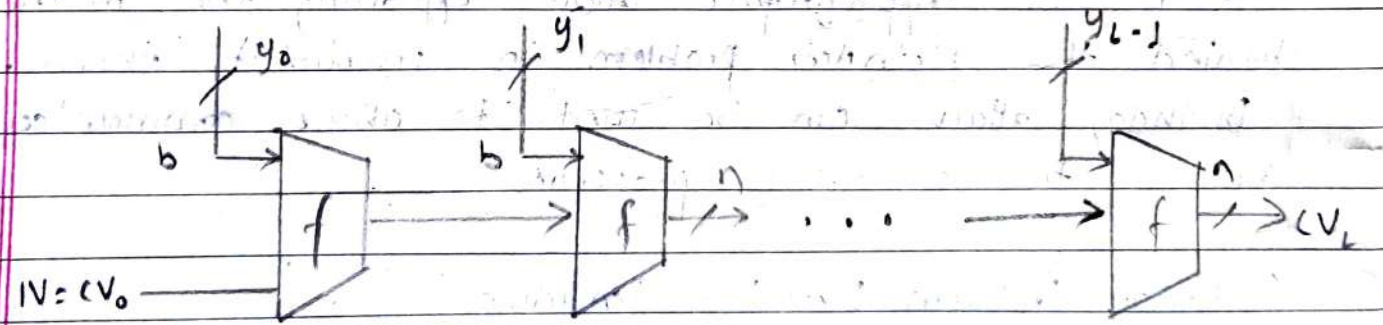


Fig: Block diagram of Hash code Generation

IV: Initial Value
CV: Chaining Variable
L: No. of i/p blocks
n: length of hash code
b: length of i/p block

The hash function algorithm involves repeated use of compression function (f) that takes 2 inputs, (an n-bit input from previous step called chaining variable and a b-bit block) and produces n-bit output. At the start of hashing function, the chaining variable has an initial value that is specified as part of the algorithm. The final value of chaining variable is the hash value, often $b > n$. Hence the term compression.

The hash function can be summarized as

$CV_0 = IV$ = Initial n-bit

$CV_i = f(CV_{i-1}, Y_{i-1}) \quad 1 \leq i \leq L$

$H(M) = CV_L$

where i/p to function is a message M consisting of $Y_0, Y_1, \ldots Y_L$. The structure can be used to produce secured Hash function to operate on a message of any length.

Secured Hash Algorithm (SHA)

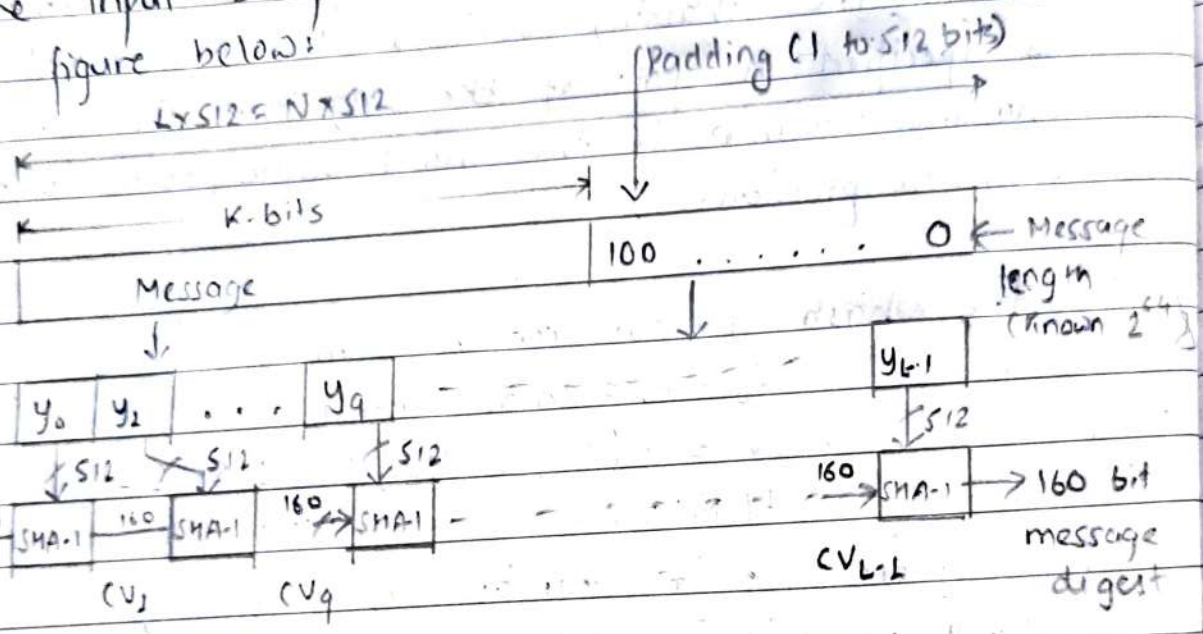
→ SHA was developed by National Institute of Science and Technology (NIST) in 1993 as a Federal Information Processing standard (FIPS 180)

→ SHA was later revised in 1995 as (FIPS-180-1) as is generally refferred to as SHA-1.

→ SHA is based on md4 which was later on replaced by md5.

→ Both md4 and md5 produce a 128 bit message digest (md) whereas SHA produces 160 bits.

→ SHA takes as input message with a maximum length of less than 2^{64} bits and produces 160 bit message digest.
→ The input is produced in 512-bits blocks as shown in figure below:



The processing consists of following 5 steps:

1. Appending Bits
→ Message is padded such that the length is congruent to 448 modulo 512 (448 mod 512). ie. the length of the padded message is 64 bit less than integer multiple of 512 bits.
→ Padding is always done even if message is of desired length.
→ Ranges from 1 to 512 bits.
→ Padding consists of a 1 followed by necessary number of 0's.

2. Append length

→ A block of 64 bits length is appended to the message which is treated as unsigned 64 bit integer (MSB first) and contain the length of the message before padding.

3. Initialize MD Buffer:

→ A 160 bit buffer is used to hold intermediate values and final result of Hash function is represented as five 32 bits register (ABCDE) initialized as follows:

A	=	67	45	23	01
B	=	ef	cd	ab	89
C	=	98	ba	dc	fe
D	=	10	32	54	76
E	=	c3	d2	e1	f0

4. Process the 512 bit (16 word) block

→ The heart of the algorithm is a module which consists of '4 rounds' of processing of 20 steps each. Each round has similar structure but uses different primitive logical function (f_1, f_2, f_3, f_4). Each round takes as input the current 512 bit block being processed (Y_q) and the 160 bit buffer value A, B, C, D, E and update the current buffer.

→ Each round also makes use of an additive constant K_E, where $0 \leq E \leq 79$, indicates one of the 80 steps occuring in 4 rounds.

In fact only 4 distinct constants are used - (one for each for $0 \le t \le 19$; $20 \le t \le 39$, $40 \le t \le 59$; $60 \le t \le 79$)

→ The o/p of q^{th} round is added (mod 2^{32}) to i/p to the first round (CV_q) to produce CV_{q+1}.

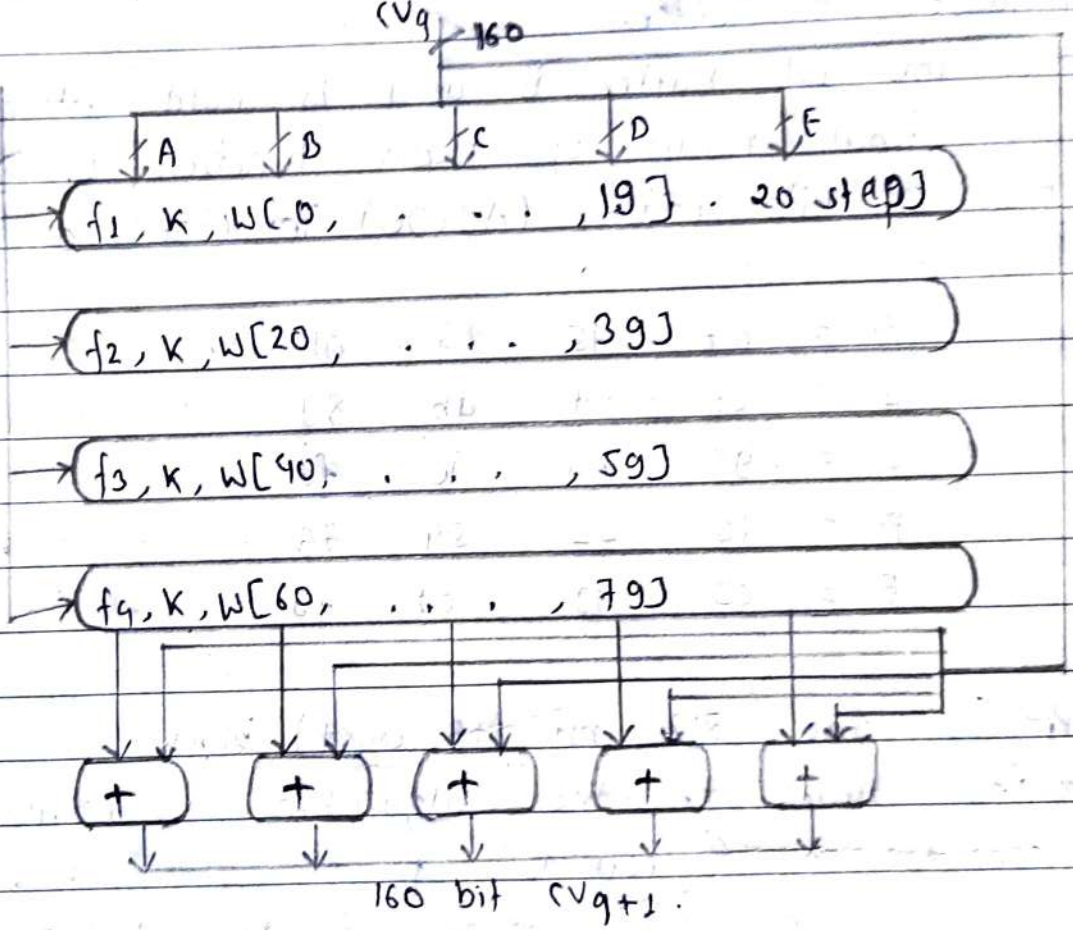


Fig: SHA-1 processing for single 512 bit block

5. The output after all L-512 bit blocks have been processed and the output from L^{th} stage is the 160-bit digest.

We can summarize the behaviour of SHA-1 as:

$CV_0 = IV$

$CV_q = SUM_{32}(CV_q, ABCDE_q)$

$MD = CV_L$

where,

IV = Initial value of ABCDE defined in above steps.

$ABCDE_q$ = o/p of last round processing.

SUM_{32} = Addition of modulo 2^{32} performed separately on each word of the pair of input.

MD = final message digest value.

MD_5

↳ It has same diagram as SHA-1.

↳ As the hash function and the structure is same as SHA-1. Here is an exception that the message length is not limited for MD_5 and the hash value and intermediate values of CV_i are 128 bits. It is designed by Ron Rivest and used very widely.

↳ Its collision resistance was broken in 2004 and it keeps getting worse.

Digital Signature:

↳ MAC only protects two parties during message exchange from third party but it doesnot protect the two parties from eachother. In this situation where they lacks trust between the sender and receiver, something more than just authentication is required. Hence Digital Signature becomes most attractive solution to the problem.

Digital signature is analogous to handwritten signature which must have the following properties.

1. It must verify the author, date and time of signature.
2. It must authenticate the contents at the time of signature.
3. It must be verifiable by third party to result any disputes.

Thus, digital function of signature includes the authentication function. On the basis of these function, we can formulate following function.

1. signature must be a bit pattern depending upon the message being signed.
2. The signature must use some unique standard information to the sender to prevent both forgery and denial.
3. It must be relatively easy to produce, recognize and verify the digital signature.
4. It must be computationally infeasible to forge a digital signature either by constructing a new message for existing digital signature or by constructing a fradulent digital signature for a given message.
5. It must be practical to retain a copy of a digital signature in storage.

The most popular algorithm for implementing digital signature is discussed below:

Digital Signature Standard (DSS)

NIST published a variant of FIPS or FIPS 186, also known as digital signature standard. It makes use of SHA and presents a new signature technique which is called Digital Signature Standard Algorithm (DSS). DSS was first proposed in 1991 and revised in 1993 in response to public feedback concerning security scheme.

In 2000, an extended version of the standard was issued as FIPS-2. The latest version incorporate digital signatures based on RSA and elliptic curve cryptography.

DSS uses an algorithm designed to provide only digital signature function. Unlike RSA, it cannot be used for encryption or key exchange. However, it is a P-K technique and is based on difficulties of computing discrete logarithms as in the case of Diffie-Hellman.

(Global) Public Key Component

$p \rightarrow$ Prime number where $2^{L-1} < p < 2^L$ for $512 \leq L \leq 1024$ and L is a multiple of 64 ie. bit length of in between 512 & 1024 bits in increment of 64 bits.

$q \rightarrow$ Prime divisor of $\phi(p) = (p-1)$: $2^{159} < q < 2^{160}$

$g = h^{(p-1)/q} \bmod p$

where,

$h \rightarrow$ any integer with $1 < h < p-1$ such that

$h^{(p-1)/q} \bmod p > 1$

User's Private Key
$x \rightarrow$ random or pseudorandom integer with $0 < x < q$

User's Public Key
$y = g^x \bmod p$

User's per message secret no.
$k \rightarrow$ random or pseudorandom number integer with $0 < k < q$.

Signing:
$r = (g^k \bmod p) \bmod q$
$s = [k^{-1}(H(m) + x \cdot r)] \bmod q$
Signature = (r, s).

Verifying:
$w = (s')^{-1} \bmod q$
$u_1 = [H(m') w] \bmod q$
$u_2 = (r') w \bmod q$
$= [g^{u_1} g^{u_2} \bmod p] \bmod q$

Test $v = r'$

where,
M = message to be signed.
H(M) = hash message using SHA-1.
M', s', r' → received ver version of M, S, R.

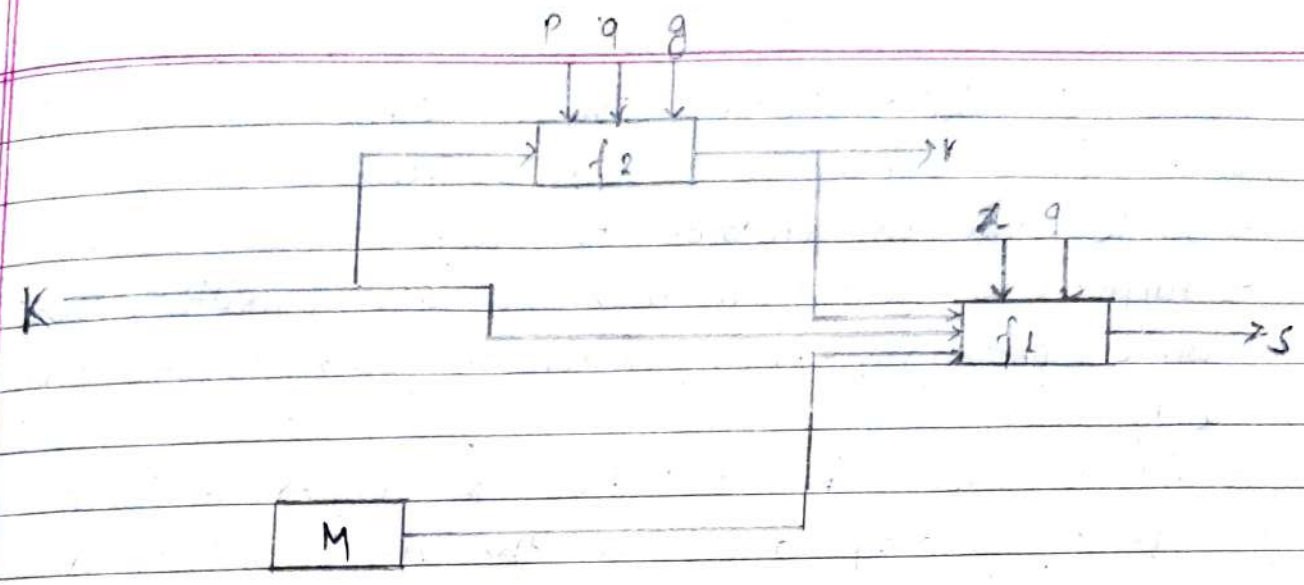


Fig: Signing

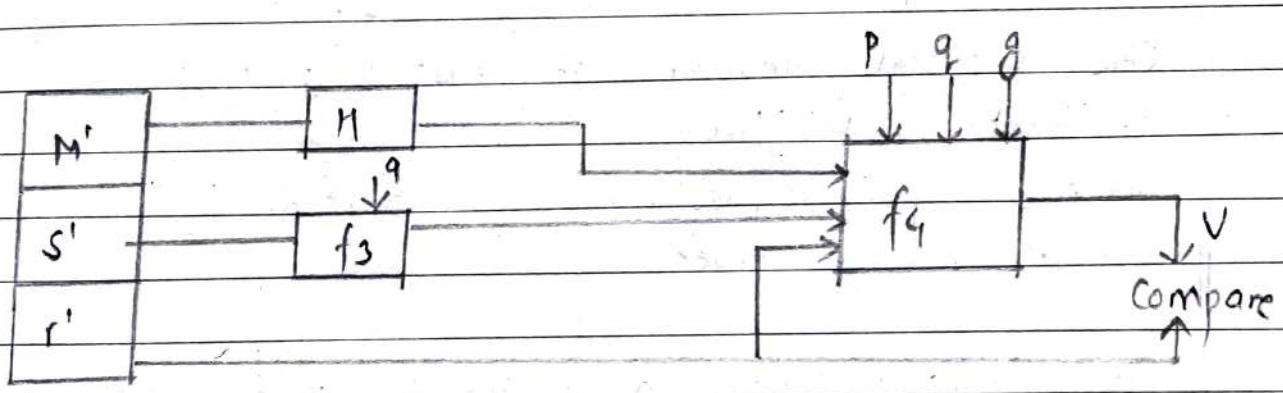


Fig: Verifying

Fig: Block diagram of DSS

Centralized Authentication Scheme (CAS)

↳ It was developed shawn Bayren.
↳ Single sign on protocol for web.
↳ Purpose: To permit a user to access multiple application while providing their credentials (such as uid and pwd) only once.
↳ Allows the web application to authenticate users without gaining access to the user's security credentials.
↳ CAS is also a software package that implement the protocols.

CAS protocols involves at least 3 parties:
1. Client Web Browser
2. Web app requesting authentication
3. The CAS Server.

It may includes a backend service such as db server that doesnot have its own HTTP interface but communicates with a web application.

When a client visits an application desiring to authenticate, the application redirects it to centralized authentcation scheme. CAS then validates the client authenticity usually by checking username and password against a database. (Kerberos, LDAP or active directories.

If the authentication succeeds, CAS returns the client to the application passing along the service ticket. The application then validates the ticket by containing cache over a secure connection and providing its own service identifier and the ticket. CAS then deletes the application trusted information about the client. (whether or not it has successfully authenticated.)

- CAS allows multi-tier authentication by using proxy addresses. Backend services like database server or main server can participate in CAS validating the authenticity of users. After it receives the information from the web application.

Kerberos

→ It is a trusted key server system developed by MIT.
→ Kerberos provides centralized third party authentication in a distributed network.
→ Access control may require on every computer resources in either local or remote network.
→ It consists of Key Distribution Center (KDC) containing database for customer services and encryption key. KDC provides non-corruptible authentication credential ticket or tokens.
→ There are two version of Kerberos.
1. Version 4 restricted to a single network or realm
2. Version 5 that allows inter-realm authentication (which is in beta-testing)

Steps:

1. User log on to a workstation and sends request to the host.
2. Authentication server verifies users access right from the database, creates a ticket and session key. The results are encrypted using the encrypted derived from user's password.
3. The workstation prompts the user for the password and the user provides password for decryption. The incoming message then send tickets and authenticator that contains username, network address and time to ticket generating server (TGS).
4. TGS decrypts the ticket and the authenticator verifies the requested service.
5. The workstation sends an authenticator and ticket to the server.
6. The server verifies that ticket and authenticator match only then it grants access to the service. If mutual authentication is required, the server returns an authentication.

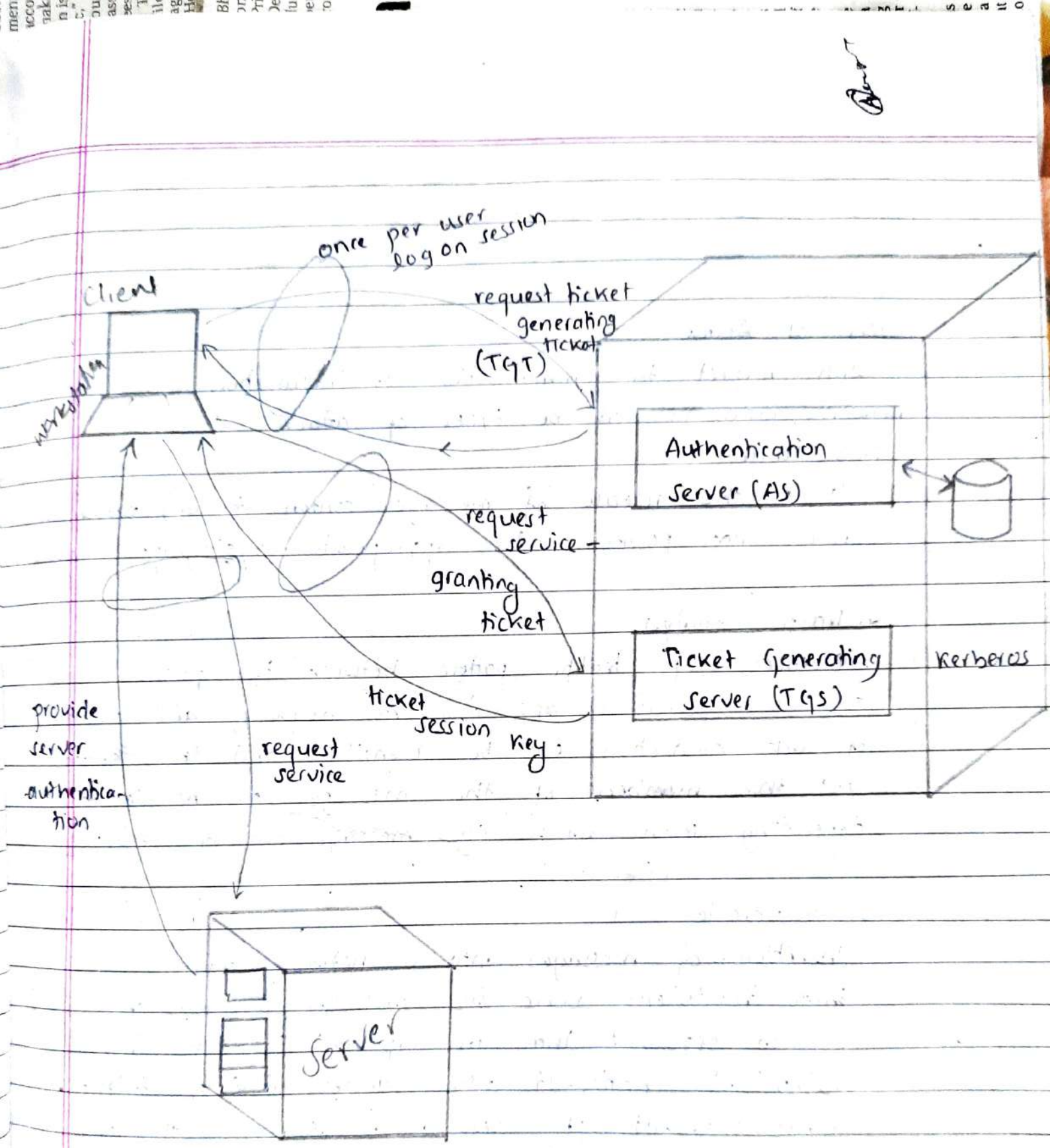


Fig: Kerberos