Q.2 a. Discuss the security mechanisms recommended by ITU-T (X.800) to provide the security services. (8)

Answer:

ITU-T (X.800) recommends some **security mechanisms** to provide the security services defined. The taxonomy of these mechanisms is shown in figure below:

- **Encipherment:** Encipherment, hiding or covering data, can provide confidentiality. It can also be used to complement other mechanisms to provide other services. Today two techniques – cryptography and steganography – are used for enciphering.
- **Data Integrity:** The data integrity mechanism appends to the data a short checkvalue that has been created by a specific process from the data itself. The receiver receives the data and the checkvalue. He creates a new checkvalue from the received data and compares the newly created checkvalue with the one received. If the two checkvalues are the same, the integrity of data has been preserved.
- **Digital Signature:** A digital signature is a means by which the sender can electronically sign the data and the receiver can electronically verify the signature. The sender uses a process that involves showing that she owns a private key related to the public key that she has announced publicly. The receiver uses the sender's public key to prove that the message is indeed signed by the sender who claims to have sent the message.
- **Authentication Exchange:** In authentication exchange, two entities exchange some messages to prove their identity to each other. For example, one entity can prove that she knows a secret that only she is supposed to know.
- **Traffic Padding:** Traffic padding means inserting some bogus data into the data traffic to thwart the adversary's attempt to use the traffic analysis.
- **Routing Control:** Routing control means selecting and continuously changing different available routes between the sender and the receiver to prevent the opponent from eavesdropping on a particular route.
- **Notarization:** Notarization means selecting a third trusted party to control communication between two entities. This can be done, for example, to prevent repudiation. The receiver can involve a trusted party to store the sender request in order to prevent the sender from later denying that she has made such a request.

Access Control: Access control uses methods to prove that a user has access right to the data or resources owned by a system.

b. Define Euclidean algorithm. Write the pseudo code of the algorithm. Use this algorithm to find the greatest common divisor of 2740 and 1760. (8)

Answer:

The **Euclidean algorithm** is based on the following two facts:

- gcd(a, 0) = a
- gcd(a, b) = gcd(b, r), where r is the remainder of dividing a by b.

The first fact tells that if the second integer is 0, the greatest common divisor is the first one. The second fact allows us to change the value of a, b until b becomes 0.

```
$r_1 \leftarrow a$; $r_2 \leftarrow b$;
while ($r_2 > 0$) {
        $q \leftarrow r_1 / r_2$;
        $r \leftarrow r_1 - q \text{ x } r_2$;
        $r_1 \leftarrow r_2$; $r_2 \leftarrow r$;
    }
    gcd(a, b) $\leftarrow r_1$;
```

Apply the above algorithm to find the gcd(2740, 1760). We initialize r_1 to 2740 and r_2 to 1760. The value of q in each step is shown using a table below:

q	r_1	r_2	r
1	2740	1760	980
1	1760	980	780
1	980	780	200
3	780	200	180
1	200	180	20
9	180	20	0
	20	0	

Hence, gcd(2740, 1760) = 20.

Q.3 a. What is the pattern in the cipher text of a one-time pad cipher in each of the following cases?
(i) The plaintext is made of n 0's.
(ii) The plaintext is made of n 1's.
(iii) The plaintext is made of alternating 0's and 1's.
(iv) The plaintext is a random string of bits. (2×4)

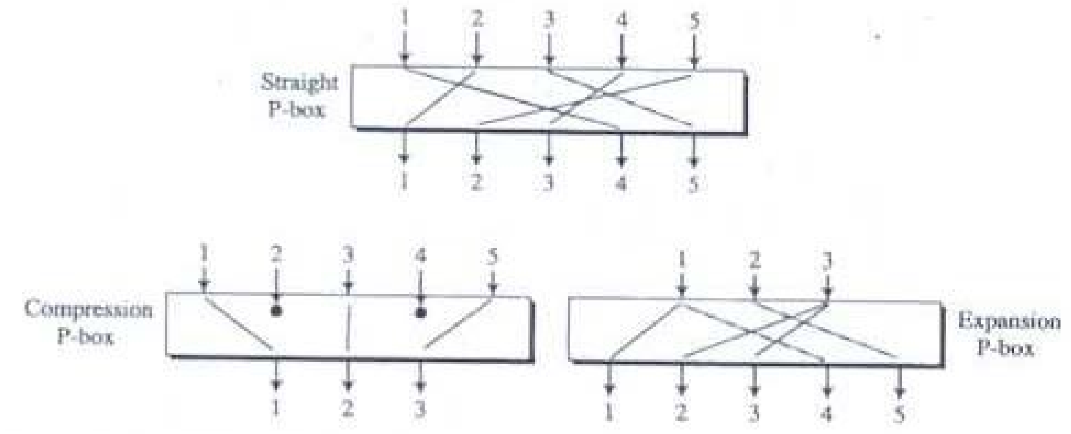
Answer:

i. Because $0 \oplus k_i = k_i$, the ciphertext stream is the same as the key stream. If the key stream is random, the ciphertext is also random. The patterns in the plaintext are not preserved in the ciphertext.
ii. Because $1 \oplus k_i = \overline{k_i}$ where $\overline{k_i}$ is the complement of k_i, the ciphertext stream is the complement of the key stream. If the key stream is random, the ciphertext is also random. Again the patterns in the plaintext are not preserved in the ciphertext.
iii. In this case, each bit in the ciphertext stream is either the same as the corresponding bit in the key stream or the complement of it. Therefore, the result is also a random string if the key stream is random.
iv. In this case, the ciphertext is definitely random because the exclusive-or of two random bits also result in a random bit.

b. Define D-box and briefly describe its three variations. Which variation is invertible? Also define S-box. (8)

Answer:

A **D-box** (called as diffusion box) parallels the traditional transposition cipher for characters. It transposes bits. There are three types of D-boxes in modern ciphers: straight D-boxes, expansion D-boxes, and compression D-boxes as shown in figure below. It helps in the spreading or diffusion of the input disturbances.



Three types of D-boxes

Straight D-boxes: A straight D-box with n-inputs and n outputs is a permutation. There are n! possible mappings.
Although a d-box can use a key to define one of the n! mappings, D-boxes are normally keyless, which means that the mapping is predetermined. If the D-box is implemented in hardware, it is prewired; if it is implemented in software, a permutation table shows the rule of mapping. In the second case, the entries in the table are the inputs and the positions of the entries are the outputs. Following table shows an example of a straight permutation table when n is 64.the table has 64 entries, corresponding to the 64 inputs. The position (index) of the entry corresponds to the output. Because the first entry contains the number 58, we know that the first output comes from the 58th input. Because the last entry is 7, we know that the 64th output comes from the 7th input, and so on.

58	50	42	34	26	18	10	02	60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06	64	56	48	40	32	24	16	08
57	49	41	33	25	17	09	01	59	51	43	35	27	19	11	03
61	53	45	37	29	21	13	05	63	55	47	39	31	23	15	07

Example of a permutation table for a straight D-box

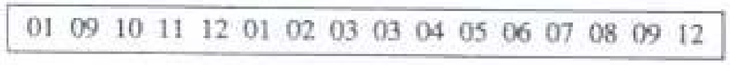
Compression D-boxes: A compression D-box is a D-box with n inputs and m outputs where m < n. Some of the inputs are blocked and do not reach the output (as shown in figure above). The compression D-boxes used in modern block ciphers normally are keyless with a table showing the rule for transposing bits. We need to know that a table for a compression D-box has m entries, but the content of each entry is from 1 to n with some missing values (those inputs that are blocked). The following table shows an example of a table for a 32 x 24 compression D-box. The inputs 7, 8, 9, 15, 16, 23, 24, and 25 are blocked.

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

Example of a 32 x 24 D-box

Compression D-boxes are used when we need to permute bits and the same time decrease the number of bits for the next stage.
Expansion D-boxes: An expansion D-box is a D-box with n inputs and m outputs where m > n. Some of the inputs are connected to more than one output (as shown in figure above). The

expansion D-boxes used in modern block ciphers normally are keyless, where a table shows the rule for transposing bits. We need to know that a table for expansion D-box has m entries, but m – n of the entries are repeated (those inputs mapped to more than one output). The following table shows an example of a table for a 12 x 16 expansion D-box. Each of the inputs 1, 3, 9, and 12 is mapped to two outputs.

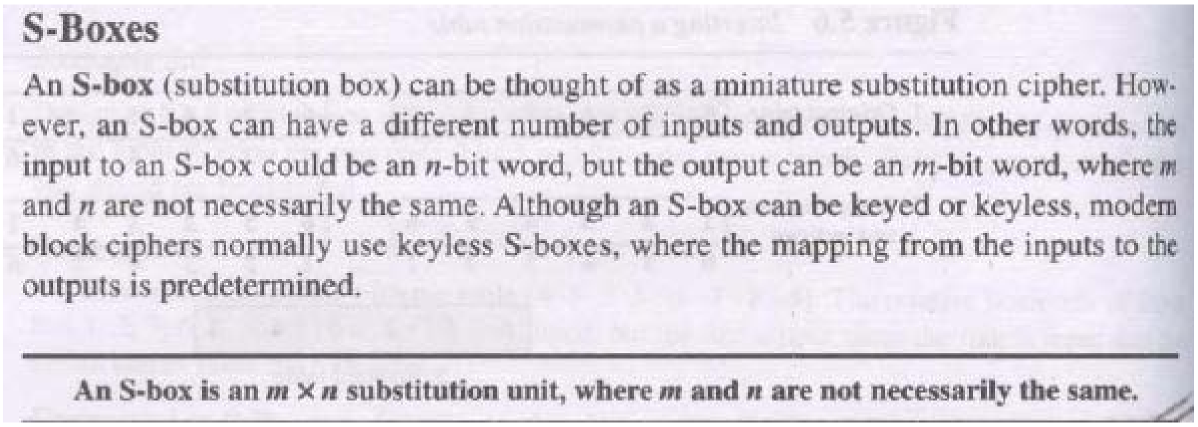


01	09	10	11	12	01	02	03	03	04	05	06	07	08	09	12

Example of a 12 x 16 D-box

Expansion D-boxes are used when we need to transpose bits and the same increase the number of bits for the next stage.

A straight D-box is invertible. This means that we can use a straight D-box in the encryption cipher and its inverse in the decryption cipher. The mapping defined by a straight D-box is a permutation. And thus may be referred to as P-box. The permutation tables, however, need to be the inverses of each other.



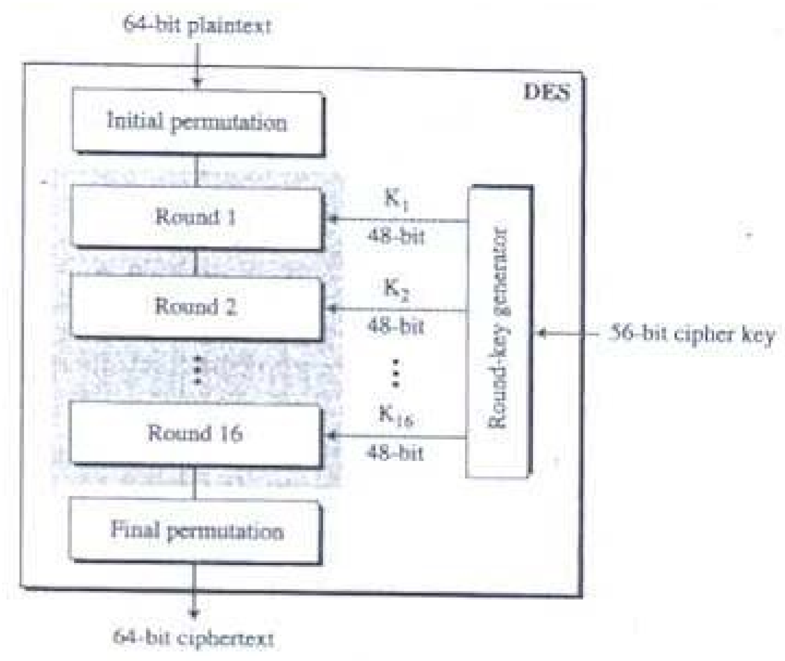
S-Boxes

An **S-box** (substitution box) can be thought of as a miniature substitution cipher. However, an S-box can have a different number of inputs and outputs. In other words, the input to an S-box could be an *n*-bit word, but the output can be an *m*-bit word, where *m* and *n* are not necessarily the same. Although an S-box can be keyed or keyless, modern block ciphers normally use keyless S-boxes, where the mapping from the inputs to the outputs is predetermined.

An S-box is an $m \times n$ substitution unit, where m and n are not necessarily the same.

Q.4 a. Draw the figure of general structure of DES. (4)

Answer:



General structure of DES

b. **What is difference between a weak key, a semi-weak key and a possible weak key? What is the disadvantage of using a weak key? (8)**

Answer:

A **Weak Key** is the one that, after parity drop operation, consists either of all 0s, all 1s or half 0s and half 1s. The round keys created from any of these weak keys are the same and have the same pattern as the cipher key. For example, the sixteen round keys created from the first key is all made of 0s; the one from the second is made of half 0s and half 1s. The reason is that the key-generation algorithm first divides the cipher key into two halves. Shifting or permutation of a block does not change the block if it is made of all 0s or all 1s. Four out of 2^{56} possible keys are called weak keys. These keys are shown are below:

Keys before parities drop (64 bits)	*Actual key (56 bits)*
0101 0101 0101 0101	0000000 0000000
1F1F 1F1F 0E0E 0E0E	0000000 FFFFFFF
E0E0 E0E0 F1F1 F1F1	FFFFFFF 0000000
FEFE FEFE FEFE FEFE	FFFFFFF FFFFFFF

Weak keys

A **Semi-Weak Key** creates only two different round keys and each of them is repeated eight times. In addition, the round keys created from each pair are the same with different orders. There are six key pairs that are called semi-weak keys. These six pairs are shown below:

First key in the pair	*Second key in the pair*
01FE 01FE 01FE 01FE	FE01 FE01 FE01 FE01
1FE0 1FE0 0EF1 0EF1	E01F E01F F10E F10E
01E0 01E1 01F1 01F1	E001 E001 F101 F101
1FFE 1FFE 0EFE-0EFE	FE1F FE1F FE0E FE0E
011F 011F 010E 010E	1F01 1F01 0E01 0E01
E0FE E0FE F1FE F1FE	FEE0 FEE0 FEF1 FEF1

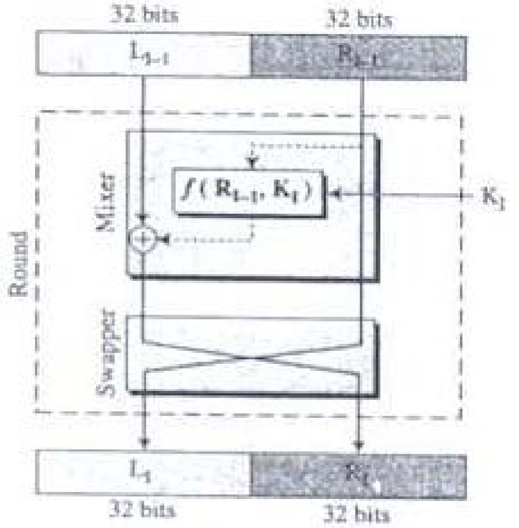
Semi-weak keys

A **possible weak key** is a key that creates only four distinct round keys; in other words, the sixteen round keys are divided into four groups and each group is made of four equal round keys. There are 48 keys that are called possible weak keys.

c. What is the number of rounds in DES? Explain. (4)

Answer:

DES uses 16 rounds. Each rounds of DES is a Feistel cipher as shown in figure below:



A round in DES (encryption site)

The round takes L_{I-1} and R_{I-1} from previous round (or the initial permutation box) and creates L_I and R_I which go to the next round (or final permutation box). We can assume that each round has two cipher elements (mixer, and swapper). Each of these elements is invertible. The swapper swaps the left half of the text with the right text. The mixer is invertible because of the XOR operation. All noninvertible elements are collected inside the function $f(R_{I-1}, K_I)$.

Q.5 a. Explain Electronic Codebook (ECB) mode? What are the security issues in ECB mode? (6)

Answer:

The simplest mode of operation is called the **electronic codebook (ECB) mode.** The plaintext is divided into N blocks. The block size is n bits. If the plaintext size is not a multiple of the block size, the text is padded to make the last block the same size as the other blocks. The same key is used to encrypt and decrypt each block. Following figure shows the encryption and decryption in this mode.

The relation between plaintext and ciphertext block is shown below:

Encryption: $C_i = E_K (P_i)$ **Decryption: $P_i = D_K (C_i)$**

Security issues: Following are the security issues in ECB mode.

1. Patterns at the block level are preserved. For example, equal blocks in the plaintext become equal blocks in the ciphertext. If eve finds out that ciphertext blocks 1, 5, and 10 are the same, she knows that the plaintext blocks 1, 5, and 10 are the same. This is a leak in security. For example, Eve can do an exhaustive search to decrypt only one of these blocks to find the contents of all of them.
2. The block independency creates opportunities for Eve to exchange some ciphertext blocks without knowing the key. For example, if she knows that block 8 always conveys some specific information, she can replace this block with the corresponding block in the previously intercepted message.

b. How initialization is done for each frame of encryption in A5/1? (4)

Answer:

Initialization is done for each frame of encryption by using a 64-bit secret key and 22 bits of the corresponding frame number. **Following are the steps:**

1. First, set all bits in three LFSRs to 0.
2. Second, mix the 64-bit key with the value of register according to the following code. Clocking means that each LFSR goes through one shifting process.

```
for (i = 0 to 63) {
        Exclusive-or K[i] with the leftmost bit in all three registers.
        Clock all three LFSRs
}
```

3. Then repeat the previous process but use the 22-bit frame number.

```
for (i = 0 to 21) {
        Exclusive-or FrameNumber[i] with the leftmost bit in all three registers.
        Clock all three LFSRs
}
```

4. For 100 cycles, clock the whole generator, but use the Majority function to see which LFSR should be clocked. Here clocking means that sometimes two and sometimes all three LFSRs go through the shifting process.

```
for (i = 0 to 63) {
        Clock the whole generator based on the majority function.
}
```

c. Write the algorithm inv_knapsackSum for a superincreasing k-tuple. Assume that a = [17, 25, 46, 94, 201, 400] and s = 272 are given. Use the algorithm to show how tuple x is found. (6)

Answer:

```
inv_knapsackSum(s, a[1 ... k]) {
        for (i = k down to 1) {
                if ($s \geq a_i$) {
                        $x_i \leftarrow 1$
                        $s \leftarrow s - a_i$
            }
            else
                $x_i \leftarrow 0$
        }
        return x[1 ... k]
}
```

Values of i, a_i, s, and x_i are listed in following table.

i	a_i	s	$s \geq a_i$	x_i	$s \leftarrow s - a_i \times x_i$
6	400	272	false	$x_6 = 0$	272
5	201	272	true	$x_5 = 1$	71
4	94	71	false	$x_4 = 0$	71
3	46	71	true	$x_3 = 1$	25
2	25	25	true	$x_2 = 1$	0
1	17	0	false	$x_1 = 0$	0

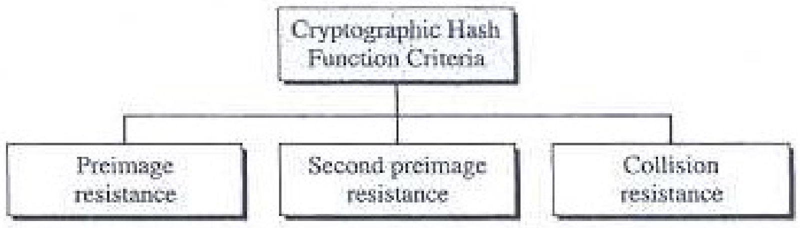
Hence, x = [0, 1, 1, 0, 1, 0] which means that 25, 46, 201 are in the knapsack.

Q.6 a. What are the different criterion for a cryptographic hash function? (8)

Answer:

Cryptographic hash functions need to satisfy certain properties, which are essential for the hash functions to be useful for various applications. The properties are three-fold, namely:

1. Preimage resistance
2. Second preimage resistance
3. Collision resistance


Cryptographic Hash Function Criteria
Preimage resistance
Second preimage resistance
Collision resistance

Criteria of a cryptographics hash function

Preimage resistance: This property of the hash function implies given a hashed value it should be difficult for an adversary to compute the preimage of the hashed value. Thus, mathematically, a hash function h is said to be preimage resistance, if given the value of y = h(x), for some x it is difficult to compute the value of any x′ such that h(x′) = y.

It may be noted that the definition of preimage resistance does not exclude the condition that x = x′.

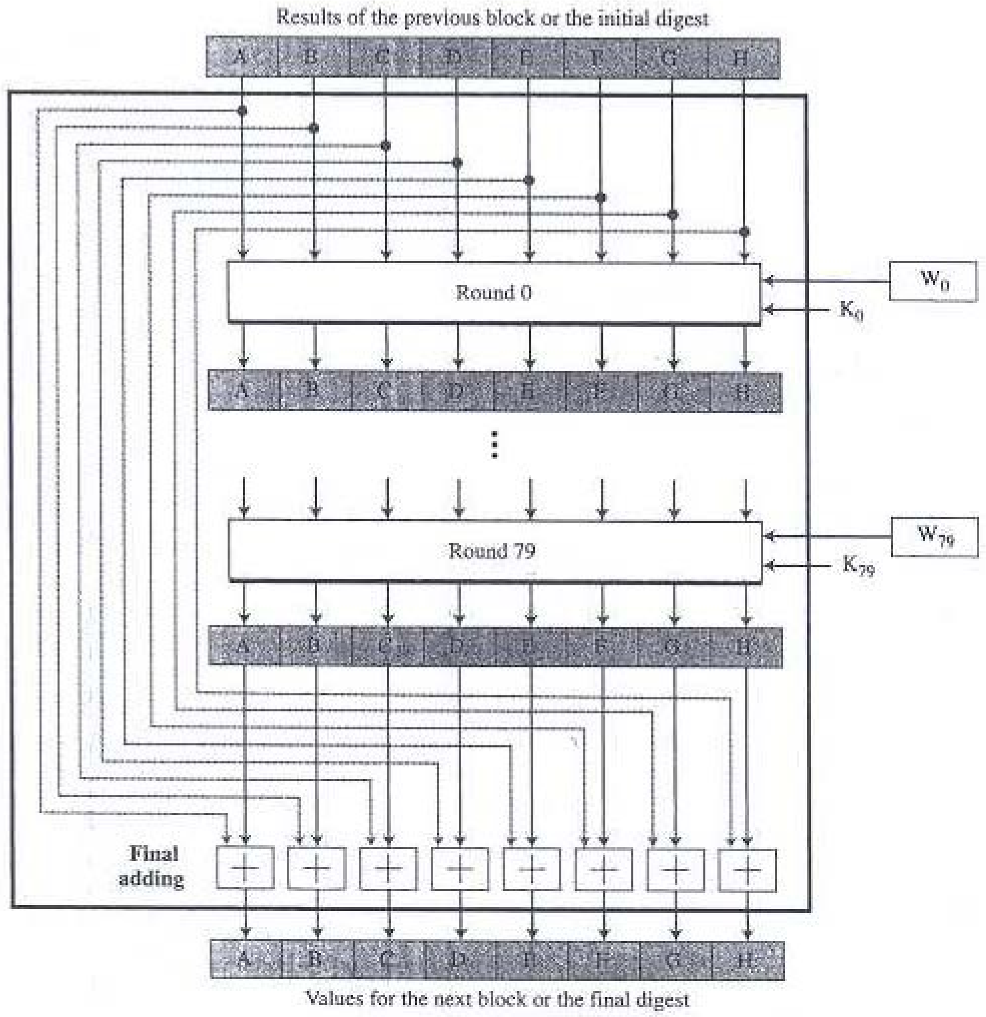
Second Preimage Resistance: In this criterion, an adversary is provided with the value of x and is asked to compute the value of $x' \neq x$, such that h(x) = h(x′). If it is difficult for the adversary to perform this computation we claim that the hash function is second preimage resistant. We call the pair (x′, h(x′)) a valid pair.

Collision Resistance: Collision of a hash function is the event when two values x and x′, such that $x \neq x'$ hash to the same value, i.e., h(x) = h(x′). A given hash function is said to have the property of collision resistance when it is difficult for the adversary to find the collisions. It may be noted that since the domain of a hash function is much larger compared to the range, collisions are bound to occur for any hash function.

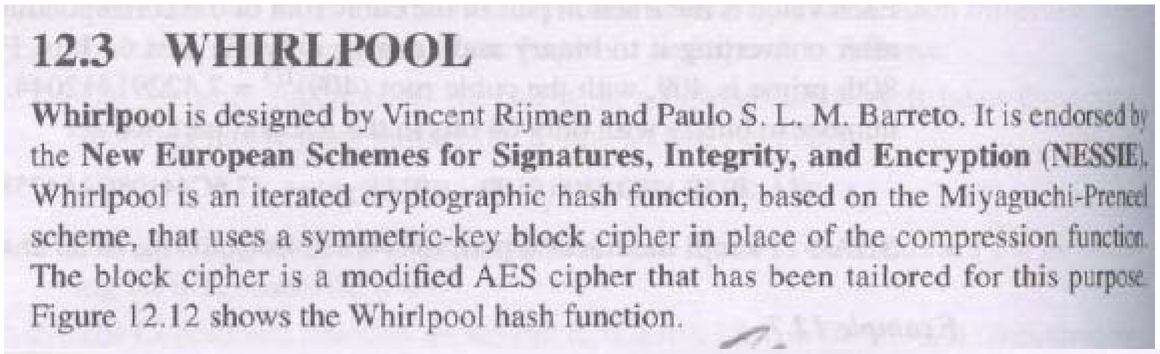
b. What kind of compression function is used in SHA-512? Differentiate it with WHIRLPOOL Cipher Method. (8)

Answer:

SHA-512 creates a 512-bits (eight 64-bit words) message digest from a multiple-block message where each block is 1024 bits. The processing of each block of data in SHA-512 involves 80 rounds. Figure below shows the general outline for the compression function. In each round, the contents of eight previous buffers, one word from the expanded block (W_i), and one 64-bit constant (K_i) are mixed together and then operated on to create a new set of eight buffers. At the beginning of processing, the values of eight buffers are saved into eight temporary variables. At the end of the processing (after step 79), these values are added to the values created from step 79. This last operation is called as *final adding*.

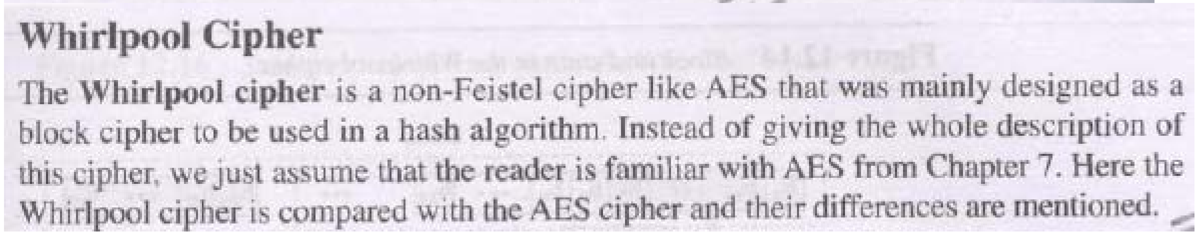

Results of the previous block or the initial digest
A
B
C
D
E
F
G
H
Round 0
W0
K0
Round 79
W79
K79
Final adding
Values for the next block or the final digest

Compression function in SHA 512



12.3 WHIRLPOOL

Whirlpool is designed by Vincent Rijmen and Paulo S. L. M. Barreto. It is endorsed by the **New European Schemes for Signatures, Integrity, and Encryption** (NESSIE). Whirlpool is an iterated cryptographic hash function, based on the Miyaguchi-Preneel scheme, that uses a symmetric-key **block cipher** in place of the compression function. The block cipher is a modified AES cipher that has been tailored for this purpose. Figure 12.12 shows the Whirlpool hash function.



Whirlpool Cipher

The **Whirlpool cipher** is a non-Feistel cipher like AES that was mainly designed as a block cipher to be used in a hash algorithm. Instead of giving the whole description of this cipher, we just assume that the reader is familiar with AES from Chapter 7. Here the Whirlpool cipher is compared with the AES cipher and their differences are mentioned.

Q.7 a. Compare and contrast a conventional signature and a digital signature.(6)

Answer:

Inclusion: A conventional signature is included in the document; it is part of the document. When we write a check, the signature is on the check; it is not a separate document. But when we sign a document digitally, we send the signature as a separate document. The sender sends two documents: the message and the signature. The recipient receives both documents and verifies that the signature belo0ngs to the supposed sender. If this is proven, the message is kept; otherwise, it is rejected.

Verification Method: The second difference between the two types of signatures is the method of verifying the signature. For a conventional signature, when the recipient receives a document, she compares the signature on the document with the signature on file. If they are the same, the document is authentic. The recipient needs to have a copy of this signature on file for comparison. For a digital signature, the recipient receives the message and the signature. A copy of the signature is not stored anywhere. The recipient needs to apply a verification technique to the combination of the message and the signature to verify the authenticity.

Relationship: For a conventional signature, there is normally a one-to-many relationship between a signature and documents. A person uses the sane signature to sign many documents. For a digital signature, there is a one-to-one relationship between a signature and a message. Each message has its own signature. The signature of one message cannot be used in other message. If Bob receives two messages, one after another, he cannot use the signature of the first message to verify the second. Each message needs a new signature.

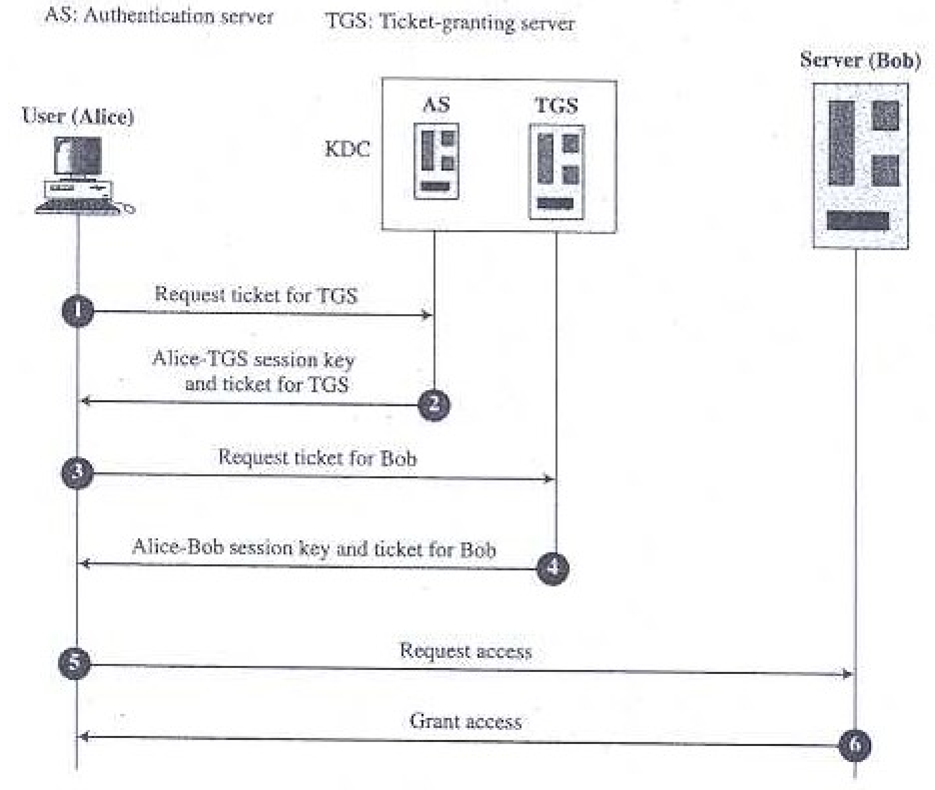
Duplicity: Another difference between the two types of signatures is a quality called duplicity. In conventional signature, a copy of the signed document can be distinguished from the original one on file. In digital signature, there is no such distinction unless there is a factor of time (such as a timestamp) on the document.

b. Define Kerberos and name its server. Briefly explain the duties of each server. (6)

Answer:

Kerberos is an authentication protocol, and at the same time a KDC, that has become very popular. Several systems, including Windows 2000, use Kerberos. It is named after the threeheaded dog in Greek mythology that guards the gates of Hades. Originally designed at MIT, it has gone through several versions.

Three servers are involved in the Kerberos protocol: an authentication server (AS), a ticket-granting server (TGS), and a real (data) server that provides services to others. Following figure shows the relationship between these three servers. In this figure, Bob is the real server and Alice is the user requesting server.


AS: Authentication server
TGS: Ticket-granting server
Server (Bob)
User (Alice)
KDC
AS
TGS
1 Request ticket for TGS
2 Alice-TGS session key and ticket for TGS
3 Request ticket for Bob
4 Alice-Bob session key and ticket for Bob
5 Request access
6 Grant access

Kerberos servers

Authentication Server (AS): The authentication server is the KDC in the Kerberos protocol. Each user registers with the AS and is granted a user identity and a password. The AS has a database with these identities and the corresponding passwords. The AS verifies the user, issues a session key to be used between Alice and the TGS, and sends a ticket for the TGS.
Ticket-Granting Server (TGS): The ticket-granting server (TGS) issues a ticket for the real server (Bob). It also provides the session key (K_{AB}) between Alice and Bob. Kerberos has separated user verification from the issuing of tickets. In this way, though Alice verifies her ID just once with the AS, she can contact the TGS multiple times to obtain tickets for different real servers.
Real Server: The real server (Bob) provides services for the user (Alice). Kerberos is designed for a client-server program, such as FTP, in which a user uses the client process to access the server process. Kerberos is not used for person-to-person authentication.

c. List the different ways using the public keys can be distributed. (4)

Answer:
Public keys, like secret keys, need to be distributed to be useful. The different ways in which public keys can be distributed are as follows:

- Public Announcement
- Trusted Center
- Controlled Trusted Center
- Certification Authority

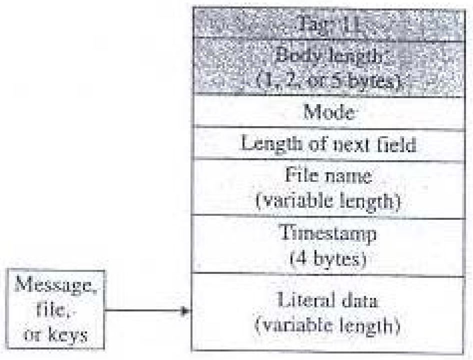
Q.8 a. Name the seven types of packets used in PGP and explain the purpose of any four. (8)

Answer:

Seven different types of packets used in PGP are following:

1. **Literal Data Packet**
2. **Compressed Data Packet**
3. **Data Packet Encrypted with Secret Key**
4. **Signature Packet**
5. **Session-Key Packet Encrypted with Public Key**
6. **Public-key Packet**
7. **User ID Packet**

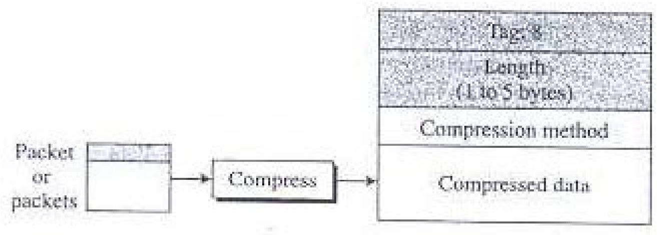
Literal Data Packet: The literal data packet is the packet that carries or holds the actual data that is being transmitted or stored. This packet is the most elementary type of message; that is, it cannot carry any other packet. The format of the packet is shown below:


Tag: 11
Body length: (1, 2, or 5 bytes)
Mode
Length of next field
File name (variable length)
Timestamp (4 bytes)
Literal data (variable length)
Message, file, or keys

Literal data packet

- **Mode:** This one-byte field defines how data is written to the packet. The value of this field can be "b" for binary, "t" for text, or any other lo0cally defined value.
- **Length of next field:** This one-byte field defines the length of the next field (file name field).
- **File name:** This variable-length field defines the name of the file or message as an ASCII string.
- **Timestamp:** This four-byte field defines the time of creation or last modification of the message. The value can be 0, which means that the user chooses not to specify a time.
- **Literal data:** This variable-length field carries the actual data (file or message) in text or binary (depending on the value of the mode field).

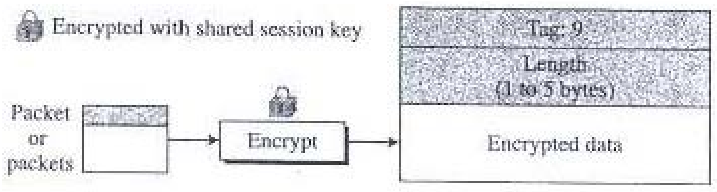
Compressed Data Packet: This packet carries compressed data packets. Following figure shows the format of a compressed data packet.



Compressed data packet

- **Compression method:** This one-byte field defines the compression method used to compress the data (next field). The values defined for this field so far are 1 (ZIP) and 2(ZLIP). Also, an implementation can use other experimental compression methods.
- **Compressed data:** This variable-length field carries the data after compression. The data in field can be one packet or the concatenation of two or more packets. The common situation is a single literal data packet or a combination of a signature packet followed by a literal data packet.

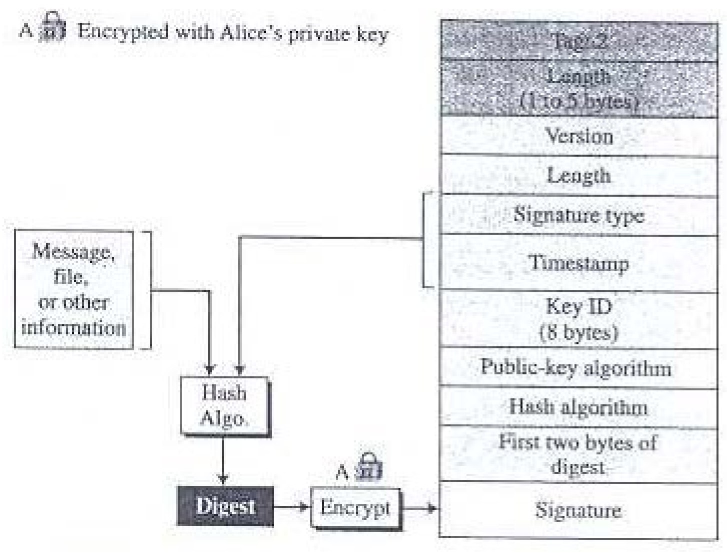
Data Packet Encrypted with Secret Key: This packet carries data from one packet or a combination of packets that have been encrypted using a conventional symmetric-key algorithm. A packet carrying the one-time session key must be sent before this packet. Following figure shows the format of the encrypted data packet.



Encrypted data packet

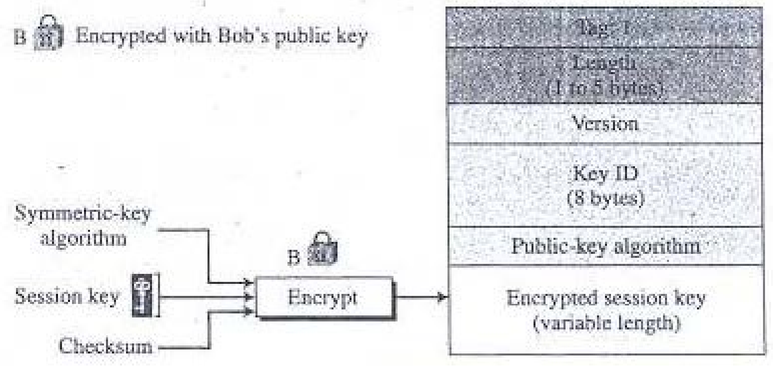
Signature Packet: A signature packet protects the integrity of the data. Following figure shows the format of the signature packet.

- **Version:** This one-byte field defines the PGP version that is being used.
- **Length:** This field was originally designed to show the length of the next two fields, but because the size of these fields is now fixed, the value of this field is 5.
- **Signature type:** This one-byte field defines the purpose of the signature, the document it signs.
- **Timestamp:** This four-byte field defines the time the signature was calculated.
- **Key ID:** This eight-byte field defines the public-key ID of the signer. It indicates to the verifier which signer public key should be used to decrypt the digest.
- **Public-key algorithm:** This one-byte field gives the code for the public-key algorithm used to encrypt the digest. The verifier uses the same algorithm to decrypt the digest.
- **Hash algorithm:** This one-byte field gives the code for the hash algorithm used to create the digest.
- **First two bytes of message digest:** These two bytes are used as a kind of checksum. They ensure that the receiver is using the right key ID to decrypt the digest.
- **Signature:** This variable-length field is the signature. It is the encrypted digest signed by the sender.



Signature packet

Session-Key Packet Encrypted with Public Key: This packet is used to send the session key encrypted with the receiver public key. The format of the packet is shown below:



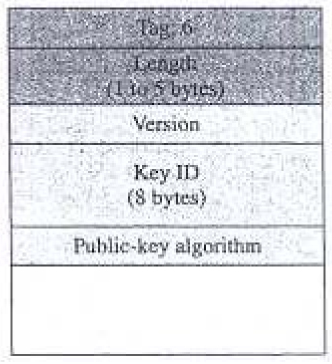
Session-key packet

- **Version:** This one-byte field defines the PGP version being used.
- **Key ID:** This eight-byte field defines the public-key ID of the sender. It indicates to the receiver which sender public key should be used to decrypt the session key.
- **Public-key algorithm:** This one-byte field gives the code for the public-key algorithm used to encrypt the session key. The receiver uses the same algorithm to decrypt the session key.
- **Encrypted session:** This variable-length field is encrypted value of the session key created by the sender and sent to the receiver. The encryption is done on the following:
    a) One-octet symmetric encryption algorithm.
    b) The session key
    c) A two-octet checksum equal to the sum of the preceding session-key octets

Public-Key Packet: This packet contains the public key of the sender. The format of the packet is shown below:

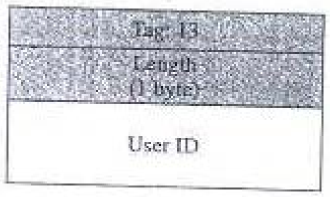
- **Version:** This one-byte field defines the PGP version of the PGP being used.
- **Timestamp:** This four-byte field defines the time the key was created.
- **Validity:** This two-byte field shows the number of days the key is valid. If the value is 0, it means the key does not expire.

- **Public-key algorithm:** This one-byte field gives the code for the public-key algorithm.
- **Public key:** This variable-length field holds the public key itself. Its contents depend on the public-key algorithm used.



Public-key packet

User ID Packet: This packet identifies a user and can normally associate the user ID contents with a public key of the sender. Following figure shows the format of the user ID packet. The length field of the general header is only one byte.



User ID packet

- **User ID:** This variable-length string defines the user ID of the sender. It is normally the name of the user followed by an e-mail address.

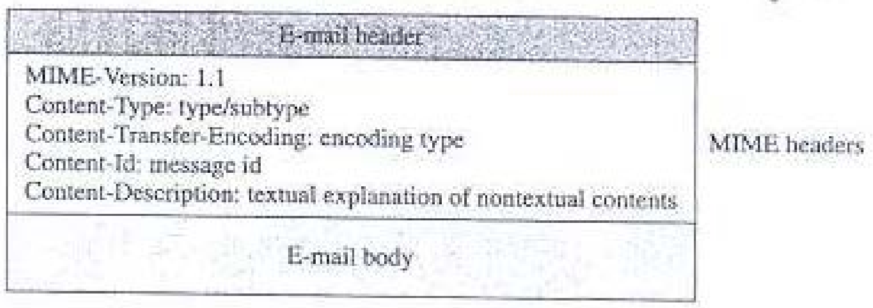
b. Describe each of the five headers of MIME, with the help of figure that can be added to the original e-mail header section to define the transformation parameters. (8)

Answer:

MIME defines five headers that can be added to the original e-mail header section to define the transformation parameters. They are:

- MIME-Version
- Content-Type
- Content-Transfer-encoding
- Content-ID
- Content-Description

Following figure shows the MIME header.


E-mail header
MIME-Version: 1.1
Content-Type: type/subtype
Content-Transfer-Encoding: encoding type
Content-Id: message id
Content-Description: textual explanation of nontextual contents
MIME headers
E-mail body

MIME header

MIME-Version: This header defines the version of MIME used. The current version is 1.1.

MIME-Version: 1.1

Content-Type: This header defines the type of data used in the body of the message. The content type and the content subtype are separated by a slash. Depending on the subtype, the header may contain other parameters.

Content-Type: <type / subtype; parameters>

MIME allows seven different types of data. They are as follows:

- **Text:** The original message is in 7-bit ASCII format and no transformation by MIME is needed. There are two subtypes currently used, *plain* and *HTML*.
- **Multipart:** The body contains multiple, independent parts. The multipart header needs to define the boundary between each part. A parameter is used for this purpose. The parameter is a string token that comes before each part; it is on a separate line by itself and is preceded by two hyphens. The body is terminated using the boundary token, again preceded by two hyphens, and then terminated with two hyphens. Four subtypes are defined for this type: *mixed, parallel, digest,* and *alternative.* The following is an example of a multipart message using a mixed subtype:
  Content-Type: multipart/mixed; boundary=xxxx
  --xxxx
  Content-Type:text/plain;
  ..
  --xxxx
  Content-Type:image/gif;
  ..
  --xxxx--
- **Message:** In the message type, the body is itself an entire mail message, a part of a mail message, or a pointer to a message. Three subtypes are currently used: *RFC822, partial,* and *external-body*. The following is an example of a message with three fragments:
  Content-Type: message/partial;
  id= "forouzan@challenger.atc.fhda.edu";
  number=1;
  total=3;

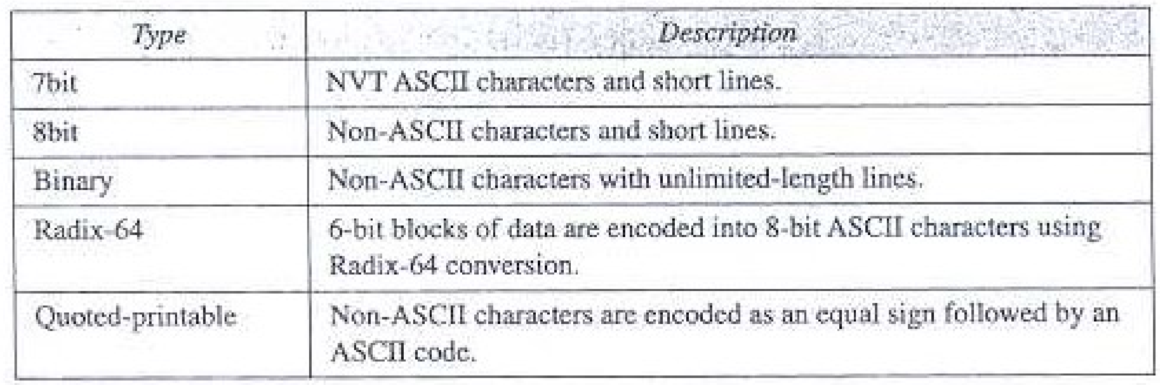
  ....
  ....

- **Image:** The original message is a stationary image, indicating that there is no animation. The two currently used subtypes are *Joint Photographic experts Group (JPEG),* which uses images compression, and *Graphics Interchange-Format (GIF)*.
- **Video:** The original message is a time-varying image (animation). The only subtype is *Moving Picture Experts group (MPEG)*. If the animated image contains sound, it must be sent separately using audio content type.
- **Audio:** The original message is sound. The only subtype is *basic*, which uses 8 kHz standard audio data.
- **Application:** The original message is a type of data not previously defined. There are only two subtypes used currently: *PostScript* and *octet-stream*.

Content-Transfer-Encoding: This header defines the method used to encode the messages into 0s and 1s for transport.

Content-Transfer-Encoding: <type>

The five types of encoding methods are listed below:



Type	*Description*
7bit	NVT ASCII characters and short lines.
8bit	Non-ASCII characters and short lines.
Binary	Non-ASCII characters with unlimited-length lines.
Radix-64	6-bit blocks of data are encoded into 8-bit ASCII characters using Radix-64 conversion.
Quoted-printable	Non-ASCII characters are encoded as an equal sign followed by an ASCII code.

Content-Transfer-Encoding

Content-ID: This header uniquely identifies the whole message in a multiple message environment.

Content-Id: id=<content-id>

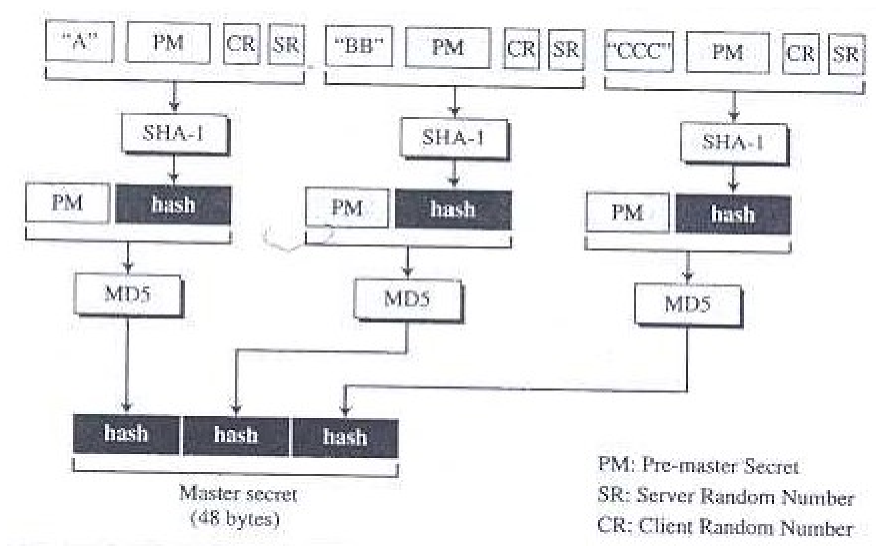
Content-Description: This header defines whether the body is image, audio, or video.

Content-Description: <description>

Q.9 a. Explain the procedure using which cryptographic parameters are generated. (8)

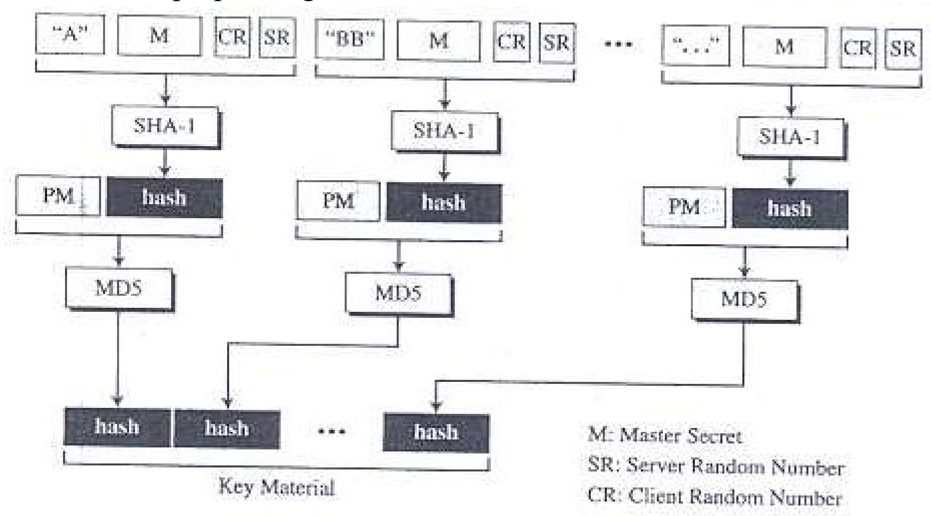
Answer:

1. The client and server exchange two random numbers; one is created by client and the other by the server.
2. The client and server exchange one pre-master secret using one of the key-exchange algorithms.
3. A 48-byte master secret is created from the pre-master secret by applying two hash functions (SHA-1 and MD5), as shown in figure below.



Calculation of master secret from pre-master secret

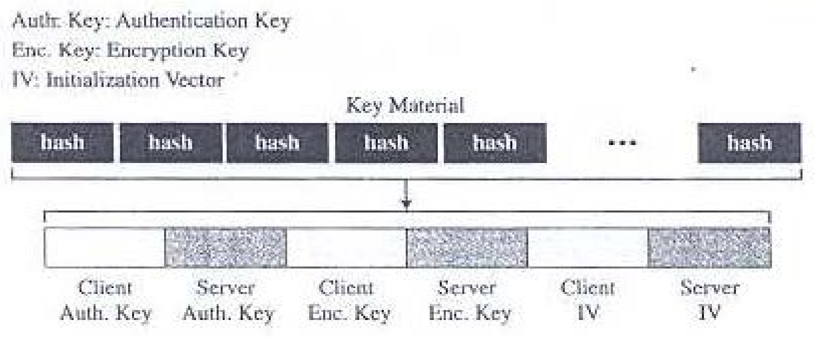
4. The master secret is used to create variable-length key material by applying the same set of hash functions and prepending with different constants as shown below.



Calculation of key material fro0m master secret

The module is repeated until key material of adequate size is created. The length of the key material block depends on the cipher suite selected and the size of keys needed for the suite.

5. Six different keys are extracted from the key material as shown in figure below.



Extractions of cryptographic secrets from key material

b. Draw the format of Record protocol general header that is added to each message coming from the sources and discuss each fields of the header. (8)

Answer:

The Record Protocol has a general header that is added to each message coming from the sources as shown in the following figure:

0 8 16 24 31

Protocol	**Version**	**Length...**
...Length		

Record Protocol General Header

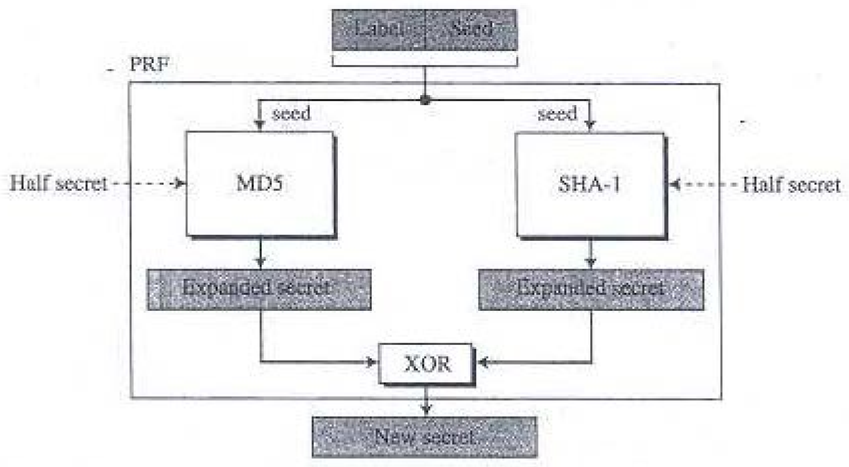
The fields in the header are discussed below:

- **Protocol:** This 1-byte field defines the source or destination of the encapsulated message. It is used for multiplexing and demultiplexing. The values are 20 (ChangeCipherSpec Protocol), 21 (Alert Protocol), 22 (Handshake Protocol) and 23 (data from the application layer).
- **Version:** This 2-byte field defines the version of the SSL; one byte is the major version and the other is the minor. The current version of SSL is 3.0 (major 3 and minor 0).
- **Length:** This 2-byte field defines the size of the message (without the header) in bytes.

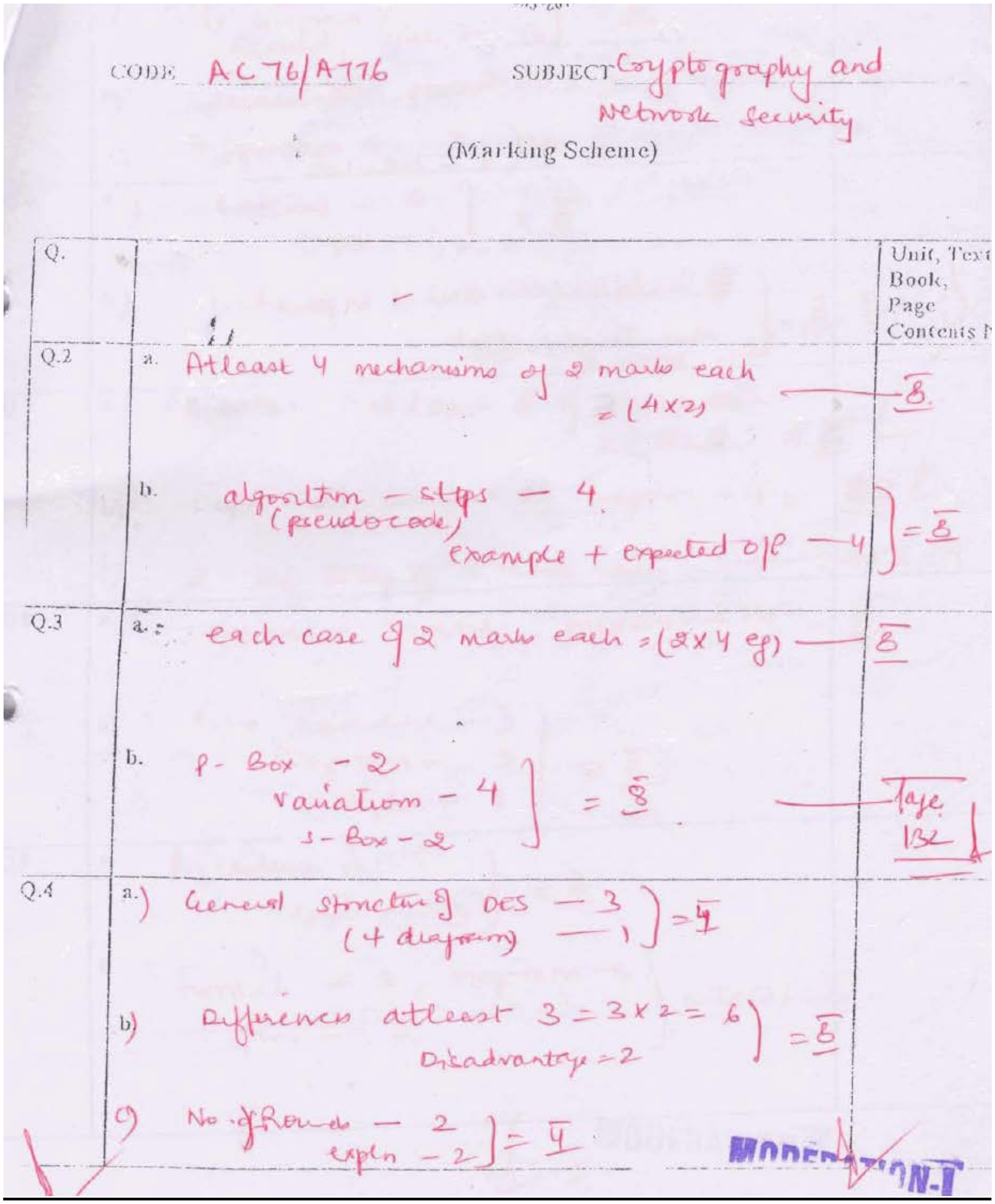
(c) With the help of a diagram discuss Pseudorandom function (PRF)?

Answer

TLS defines a **pseudorandom function (PRF)** to be the combination of two-expansion functions, one using MD5 and the other SHA-1. PRF takes three inputs, a secret, a label, and a seed. The label and seed are concatenated and serve as the seed for data-expansion function. The secret is dived into two halves; each half is used as the secret for each data-expansion function. The output of two data-expansion functions is exclusive-ored together to create the final expanded secret. Since, the hashes created from MD5 and SHA-1 are 0of different sizes, extra sections of MD5-based functions must be created to make the two outputs the same size. Following figure shows the idea of PRF.



Pseudorandom Function

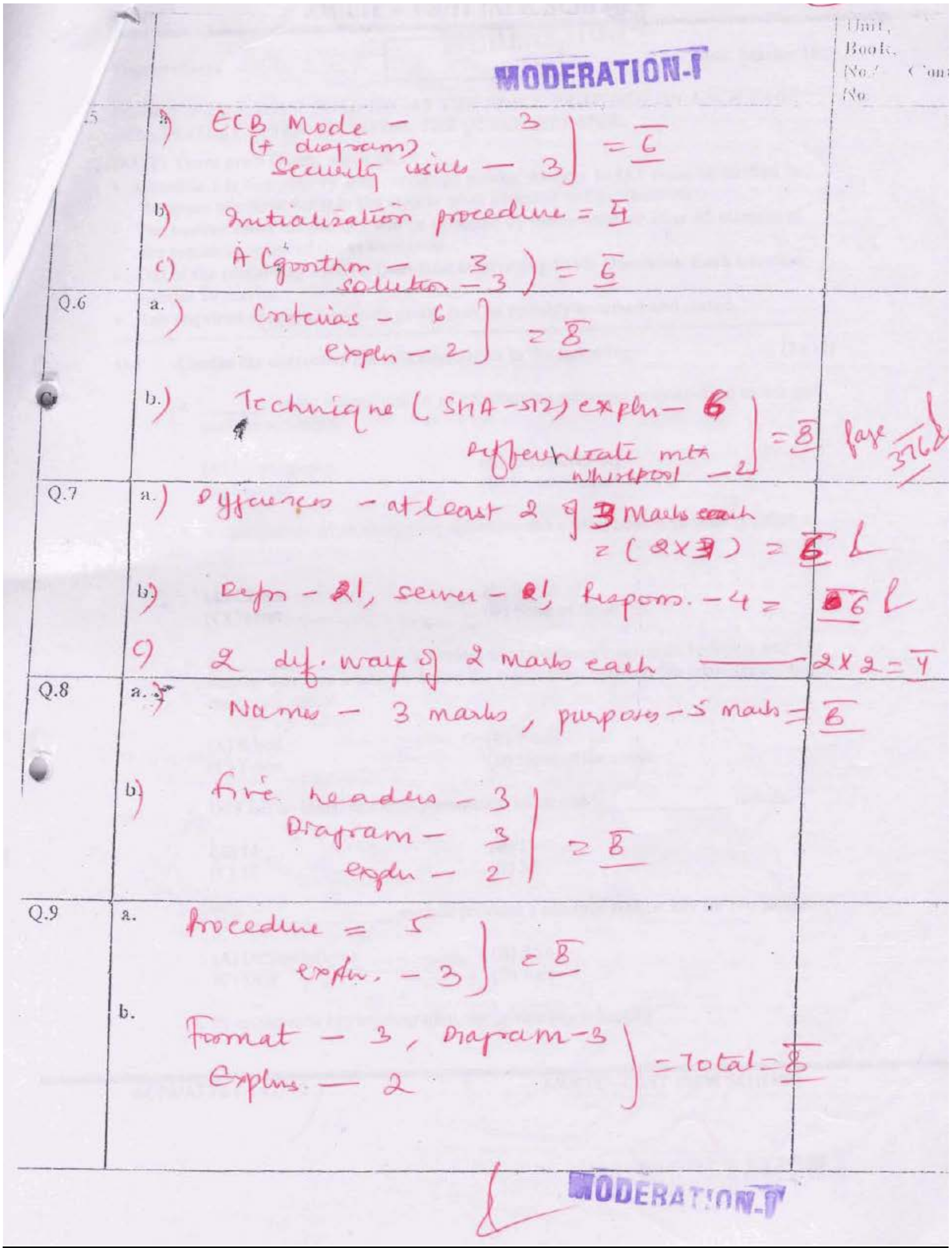


CODE AC76/AT76 SUBJECT Cryptography and Network Security

(Marking Scheme)

Q.			Unit, Text Book, Page Contents
Q.2	a.	Atleast 4 mechanisms of 2 marks each = (4x2)	8
	b.	algorithm – steps – 4 (pseudocode) example + expected o/p – 4] = 8	
Q.3	a.	each case of 2 marks each = (2x4 eg)	8
	b.	P-Box – 2 variation – 4 S-Box – 2] = 8	Page 132
Q.4	a.)	General structure of DES – 3 (+ diagram) – 1] = 4	
	b)	Differences atleast 3 = 3x2 = 6 Disadvantage = 2) = 8	
	c)	No. of Rounds – 2 expln – 2] = 4	

MODERATION-I



MODERATION-I

			Unit, Book, No./ Page No.
Q.5	a)	ECB Mode – 3 (+ diagram) Security issues – 3 = 6	
	b)	Initialization procedure = 4	
	c)	Algorithm – 3, solution – 3 = 6	
Q.6	a.)	Criterias – 6, expln – 2 = 8	
	b.)	Technique (SHA-512) expln – 6, Differentiate with Whirlpool – 2 = 8	Page 376
Q.7	a.)	Differences – atleast 2 of 3 Marks each = (2×3) = 6	
	b)	Reqn – 1, server – 1, Respons. – 4 = 6	
	c)	2 dif. ways of 2 marks each —— 2×2 = 4	
Q.8	a.	Names – 3 marks, purpose – 5 marks = 8	
	b)	five headers – 3, Diagram – 3, expln – 2 = 8	
Q.9	a.	Procedure = 5, expln. – 3 = 8	
	b.	Format – 3, Diagram – 3, Explns – 2 = Total = 8	

[illegible]

MODERATION-I

Text Book

Behrouz A. Forouzan Cryptography & Network Security Special Indian Edition.