UNIT IV NETWORK SECURITY **9**

OSI Security Architecture – Attacks – Security Services and Mechanisms – Encryption –Advanced Encryption Standard – Public Key Cryptosystems – RSA Algorithm – Hash Functions – Secure Hash Algorithm – Digital Signature Algorithm

OSI Security Architecture

- International Telecommunication Union(ITU-T) Telecommunication Standardization Sector Recommended X.800, Security Architecture for OSI.
- It is a systematic approach. The OSI security architecture is useful to managers as a way of organizing the task of providing security.
- The OSI security architecture focuses on security attacks, mechanisms, and services. These can be defined briefly as

  Security attack: Any action that compromises the security of information owned by an organization.

  Security mechanism: A process that is designed to detect, prevent, or recover from a security attack.

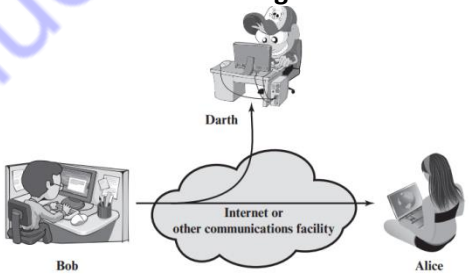
  Security service: A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization.

Security Attacks:

- Security attacks mainly classified into two types. **Passive attacks and Active attacks**

Passive Attacks:

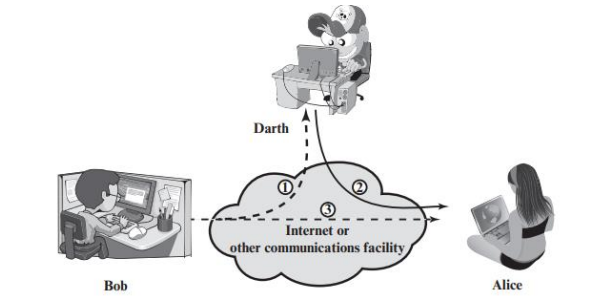
- Passive attacks are in the nature of listening and monitoring others transmissions. The goal of the opponent is to obtain information that is being transmitted.
- Two types of passive attacks are the release of **message contents and traffic analysis**.



- The release of message contents is easily understood. A telephone conversation, an electronic mail message, and a transferred file may contain sensitive or confidential information.
- A second type of passive attack, **traffic analysis**. Suppose that we had a way of masking the contents of messages or other information traffic so that opponents, even if they captured the message, could not extract the information from the message.
- Passive attacks are very difficult to detect, because they do not involve any alteration of the data.
- Typically, the message traffic is sent and received in an apparently normal fashion, and neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern.

Active attacks

- It involves some modification of the data stream or the creation of a false stream and can be subdivided into four categories: masquerade, replay, modification of messages, and denial of service.



Masquerade:

It takes place when one entity pretends to be a different entity.

A masquerade attack usually includes one of the other forms of active attack.

Replay:

It involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect (paths 1, 2, and 3 active).

Modification of messages: It simply means that some portion of a legitimate message is altered, or that messages are delayed or reordered, to produce an unauthorized effect (paths 1 and 2 active).

Denial of service: It prevents or inhibits the normal use or management of communications facilities. This attack may have a specific target. Another form of service denial is the disruption of an entire network, either by disabling the network or by overloading it with messages so as to degrade performance.

Security Services:

- X.800 defines a security service as a service that is provided by a protocol layer of communicating open systems and that ensures adequate security of the systems or of data transfers.
- A processing or communication service that is provided by a system to give a specific kind of protection to system resources; security services implement security policies and are implemented by security mechanisms

Authentication:

- The authentication service is concerned with assuring that a communication is authentic.
- In the case of a single message, such as a warning or alarm signal, the function of the authentication service is to assure the recipient that the message is from the source that it claims to be from.
- In the case of an ongoing interaction, such as the connection of a terminal to a host, two aspects are involved. First, at the time of connection initiation, the service assures that the two entities are authentic, that is, that each is the entity that it claims to be.
- Second, the service must assure that the connection is not interfered with in such a way that a third party can masquerade as one of the two legitimate parties for the purposes of unauthorized transmission or reception.

Two specific authentication services are defined in X.800:

■ **Peer entity authentication:** Provides for the corroboration of the identity of a peer entity in an association. Two entities are considered peers if they implement to same protocol in different systems; for example two TCP modules in two communicating systems. Peer entity

authentication is provided for use at the establishment of, or at times during the data transfer phase of, a connection. It attempts to provide confidence that an entity is not performing either a masquerade or an unauthorized replay of a previous connection.

■ **Data origin authentication:** Provides for the corroboration of the source of a data unit. It does not provide protection against the duplication or modification of data units. This type of service supports applications like electronic mail, where there are no prior interactions between the communicating entities

AUTHENTICATION

The assurance that the communicating entity is the one that it claims to be.

Peer Entity Authentication
Used in association with a logical connection to provide confidence in the identity of the entities connected.

Data-Origin Authentication
In a connectionless transfer, provides assurance that the source of received data is as claimed.

ACCESS CONTROL

The prevention of unauthorized use of a resource (i.e., this service controls who can have access to a resource, under what conditions access can occur, and what those accessing the resource are allowed to do).

DATA CONFIDENTIALITY

The protection of data from unauthorized disclosure.

Connection Confidentiality
The protection of all user data on a connection.

Connectionless Confidentiality
The protection of all user data in a single data block.

Selective-Field Confidentiality
The confidentiality of selected fields within the user data on a connection or in a single data block.

Traffic-Flow Confidentiality
The protection of the information that might be derived from observation of traffic flows.

DATA INTEGRITY

The assurance that data received are exactly as sent by an authorized entity (i.e., contain no modification, insertion, deletion, or replay).

Connection Integrity with Recovery
Provides for the integrity of all user data on a connection and detects any modification, insertion, deletion, or replay of any data within an entire data sequence, with recovery attempted.

Connection Integrity without Recovery
As above, but provides only detection without recovery.

Selective-Field Connection Integrity
Provides for the integrity of selected fields within the user data of a data block transferred over a connection and takes the form of determination of whether the selected fields have been modified, inserted, deleted, or replayed.

Connectionless Integrity
Provides for the integrity of a single connectionless data block and may take the form of detection of data modification. Additionally, a limited form of replay detection may be provided.

Selective-Field Connectionless Integrity
Provides for the integrity of selected fields within a single connectionless data block; takes the form of determination of whether the selected fields have been modified.

NONREPUDIATION

Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.

Nonrepudiation, Origin
Proof that the message was sent by the specified party.

Nonrepudiation, Destination
Proof that the message was received by the specified party.

Data Confidentiality:

- Data Confidentiality is the protection of transmitted data from passive attacks.
- With respect to the content of a data transmission, several levels of protection can be identified. The broadest service protects all user data transmitted between two users over a period of time.
- Narrower forms of this service can also be defined, including the protection of a single message or even specific fields within a message.
- The other aspect of confidentiality is the protection of traffic flow from analysis.
- This requires that an attacker not be able to observe the source and destination, frequency, length, or other characteristics of the traffic on a communications facility.

Data Integrity:

- Data integrity can apply to a stream of messages, a single message, or selected fields within a message. The most useful and straightforward approach is total stream protection.
- A connection-oriented integrity service, one that deals with a stream of messages, assures that messages are received as sent with no duplication, insertion, modification, reordering, or replays.
- The destruction of data is also covered under this service.
- Thus, the connection-oriented integrity service addresses both message stream modification and denial of service.
- A connectionless integrity service, one that deals with individual messages without regard to any larger context, generally provides protection against message modification only.
- Because the integrity service relates to active attacks, we are concerned with detection rather than prevention.
- If a violation of integrity is detected, then the service may simply report this violation, and some other portion of software or human intervention is required to recover from the violation. Alternatively, there are mechanisms available to recover from the loss of integrity of data.

Security Mechanisms:

- The mechanisms are divided into those that are implemented in a specific protocol layer, such as TCP or an application-layer protocol, and those that are not specific to any particular protocol layer or security service.

Encipherment or Encryption

The use of mathematical algorithms to transform data into a form that is not readily intelligible. The transformation and subsequent recovery of the data depend on an algorithm and zero or more encryption keys.

Digital Signature:

Data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g., by the recipient)

Access Control -A variety of mechanisms that enforce access rights to resources.

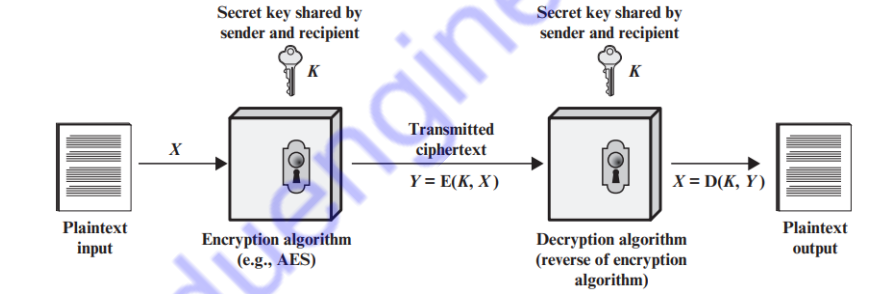
Data Integrity -A variety of mechanisms used to assure the integrity of a data unit or stream of data units

Security Label- The marking bound to a resource that names or designates the security attributes of that resource.

Event Detection -Detection of security-relevant events.

Security Audit Trail -Data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities.
Security Recovery Deals with requests from mechanisms, such as event handling and management functions, and takes recovery actions.

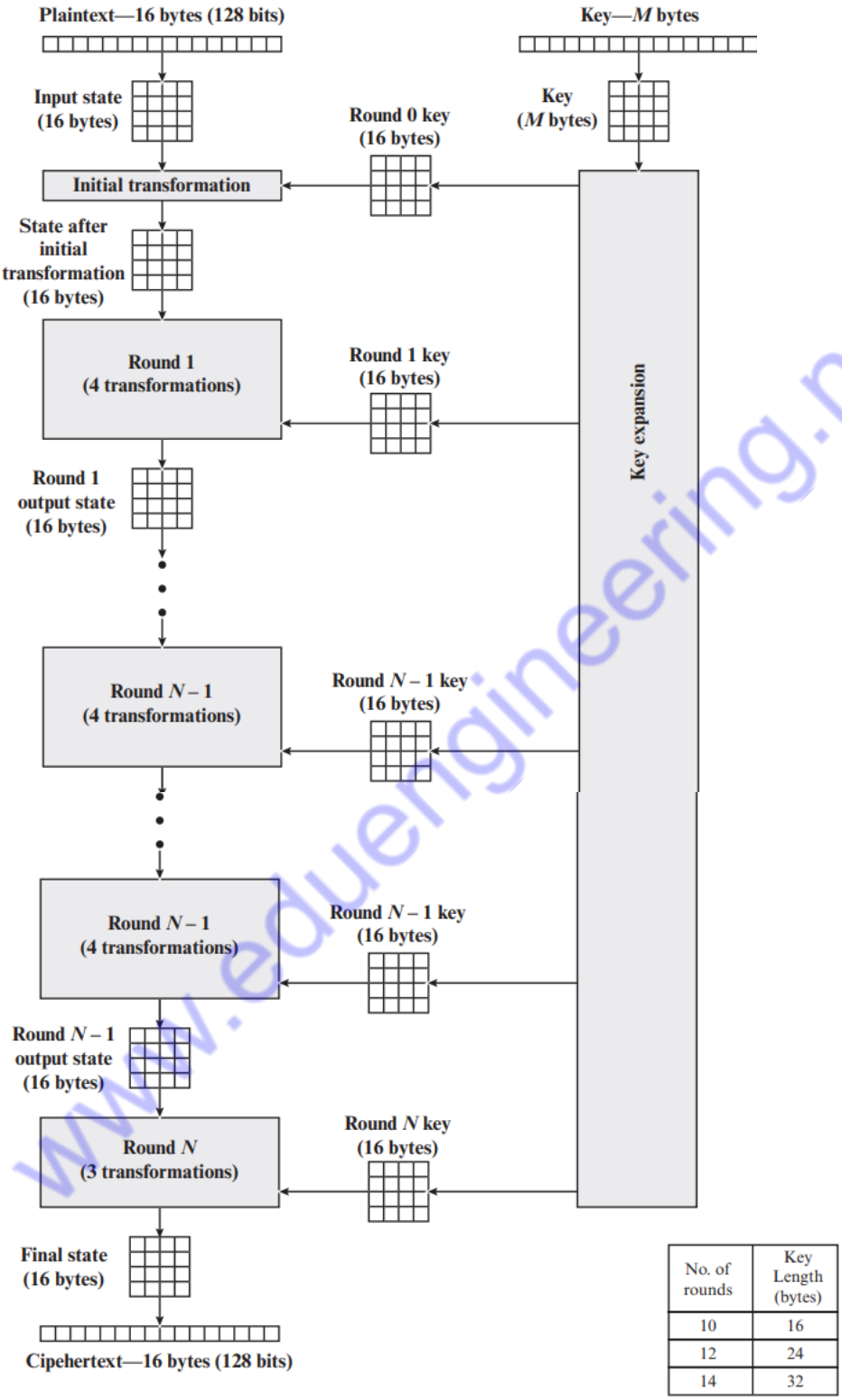
Encryption:
An original message is known as the **plaintext**, while the coded message is called the **ciphertext**. The process of converting from plaintext to ciphertext is known as **enciphering or encryption.** restoring the plaintext from the ciphertext is **deciphering or decryption**. The many schemes used for **encryption constitute the area of study known as cryptography**.
Symmetric Cipher Model:
Plaintext: This is the original intelligible message or data that is fed into the algorithm as input.
Encryption algorithm: The encryption algorithm performs various substitutions and transformations on the plaintext.
Secret key: The secret key is also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time.
Ciphertext: This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different ciphertexts.
Decryption algorithm: This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext



The Advanced Encryption Standard (AES)

- Advanced Encryption Standard was published by the National Institute of Standards and Technology (NIST) in 2001.
- AES is widely used today as it is a much stronger than Data Encryption standard despite being harder to implement
- In AES, all operations are performed on 8-bit bytes. In particular, the arithmetic operations of addition, multiplication, and division are performed over the finite field.
- The cipher takes a plaintext block size of 128 bits, or 16 bytes. The key length can be 16, 24, or 32 bytes (128, 192, or 256 bits).
- The algorithm is referred to as AES-128, AES-192, or AES-256, depending on the key length. The input to the encryption and decryption algorithms is a single 128-bit block.
- This block is depicted as a 4 * 4 square matrix of bytes. This block is copied into the State array, which is modified at each stage of encryption or decryption. After the final stage, State is copied to an output matrix.
- The key is depicted as a square matrix of bytes. This key is then expanded into an array of key schedule words

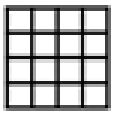
- Each word is four bytes, and the total key schedule is 44 words for the 128-bit key



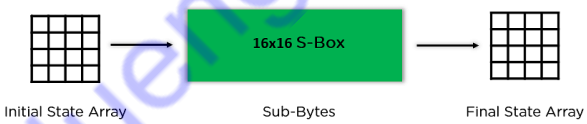
No. of rounds	Key Length (bytes)
10	16
12	24
14	32

- The cipher consists of N rounds, where the number of rounds depends on the key length: 10 rounds for a 16-byte key, 12 rounds for a 24-byte key, and 14 rounds for a 32-byte key
- The first N - 1 rounds consist of four distinct transformation functions: SubBytes, ShiftRows, MixColumns, and AddRoundKey, which are described subsequently. The final round contains only three transformations, and there is a initial single transformation

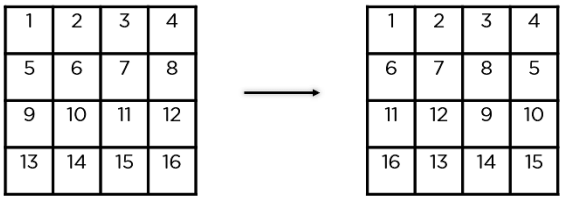
  Eg: AES considers each block as a 16 byte (4 byte x 4 byte = 128) grid in a column major arrangement.



- Each round comprises of 4 steps :SubBytes, Shift Rows, MixColumns, Add Round Key
- The last round doesn't have the MixColumns round.
- The SubBytes does the substitution and ShiftRows and MixColumns performs the permutation in the algorithm.
- **SubBytes :**
  This step implements the substitution.
- In this step each byte is substituted by another byte. Its performed using a lookup table also called the S-box.
- This substitution is done in a way that a byte is never substituted by itself and also not substituted by another byte which is a compliment of the current byte. The result of this step is a 16 byte (4 x 4) matrix like before.



- **ShiftRows :**
  This step is just as it sounds. Each row is shifted a particular number of times.
- The first row is not shifted
- The second row is shifted once to the left.
- The third row is shifted twice to the left.
  The fourth row is shifted thrice to the left. (A left circular shift is performed.)

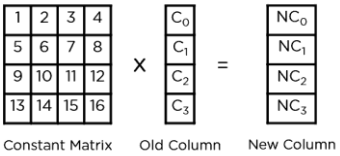


1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

→

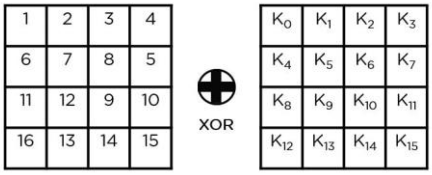
1	2	3	4
6	7	8	5
11	12	9	10
16	13	14	15

MixColumns :
This step is basically a matrix multiplication. Each column is multiplied with a specific matrix and thus the position of each byte in the column is changed as a result.
This step is skipped in the last round.

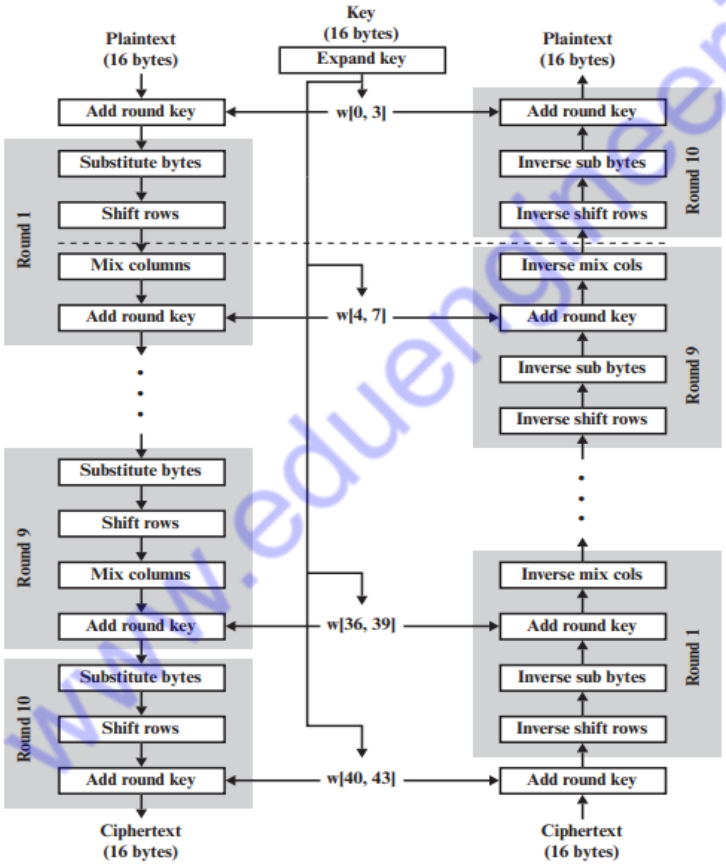


Add Round Keys :

The resultant output of the previous stage is XOR-ed with the corresponding round key. Here, the 16 bytes is not considered as a grid but just as 128 bits of data.



After all these rounds 128 bits of encrypted data is given back as output. This process is repeated until all the data to be encrypted undergoes this process.



Decryption :

The stages in the rounds can be easily undone as these stages have an opposite to it which when performed reverts the changes. Each 128 blocks goes through the 10,12 or 14 rounds depending on the key size.

The stages of each round in decryption is as follows:

- Add round key
- Inverse Mix Columns
- Shift Rows
- Inverse SubByte

The decryption process is the encryption process done in reverse .

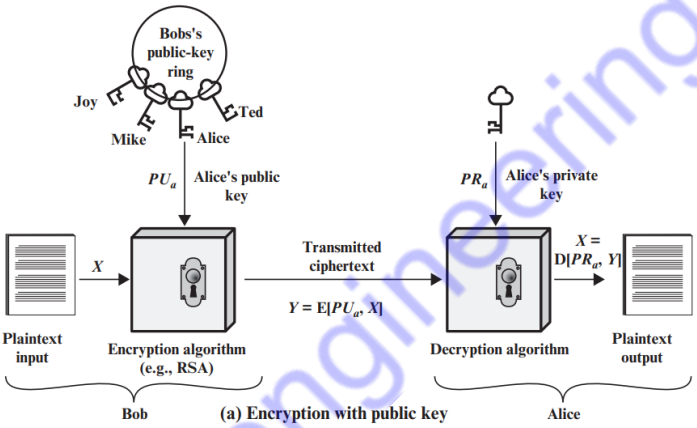
Public Key Cryptosystems:

Asymmetric algorithms rely on one key for encryption and a different but related key for decryption.

It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.

A public-key encryption scheme has six ingredients

- **Plaintext:** This is the readable message or data that is fed into the algorithm as input.
- **Encryption algorithm:** The encryption algorithm performs various transformations on the plaintext.



Public and private keys: This is a pair of keys that have been selected so that if one is used for encryption, the other is used for decryption. The exact transformations performed by the algorithm depend on the public or private key that is provided as input.

Ciphertext: This is the encrypted message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different cipher texts.

Decryption algorithm: This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

1. Each user generates a pair of keys to be used for the encryption and decryption of messages.
2. Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private.
3. If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.
4. When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice's private key.

Conventional Encryption	Public-Key Encryption
Needed to Work: 1. The same algorithm with the same key is used for encryption and decryption. 2. The sender and receiver must share the algorithm and the key. *Needed for Security:* 1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if the key is kept secret. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	*Needed to Work:* 1. One algorithm is used for encryption and a related algorithm for decryption with a pair of keys, one for encryption and one for decryption. 2. The sender and receiver must each have one of the matched pair of keys (not the same one). *Needed for Security:* 1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if one of the keys is kept secret. 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

Applications of Public Key cryptosystems:

Encryption/decryption: The sender encrypts a message with the recipient's public key, and the recipient decrypts the message with the recipient's private key.

Digital signature: The sender "signs" a message with its private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is a function of the message.

Key exchange: Two sides cooperate to exchange a session key, which is a secret key for symmetric encryption generated for use for a particular transaction and valid for a short period of time

RSA Algorithm:

- Rivest-Shamir-Adleman (RSA) scheme is a most widely accepted and implemented general-purpose approach to public-key encryption.
- The RSA scheme is a cipher in which the plaintext and ciphertext are integers between 0 and n - 1 for some n.
- A typical size for n is 1024 bits, or 309 decimal digits. That is, n is less than 2^{1024}.
- RSA makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number n. That is, the block size must be less than or equal to $\log_2(n) + 1$

Steps in RSA Algorithm:

Step-1: Select two prime numbers p and q where $p \neq q$.

Step-2: Calculate $n = p * q$.

Step-3: Calculate $\Phi(n) = (p-1) * (q-1)$.

Step-4: Select e such that, e is relatively prime to $\Phi(n)$, i.e. $(e, \Phi(n)) = 1$ and $1 < e < \Phi(n)$

Step-5: Calculate $d = e^{-1} \bmod \Phi(n)$ or $ed = 1 \bmod \Phi(n)$.

Step-6: Public key = {e, n}, private key = {d, n}.

Step-7: Find out cipher text using the formula,

$C = P^e \bmod n$ where, $P < n$ where C = Cipher text, P = Plain text, e = Encryption key and n=block size.

Step-8: $P = C^d \bmod n$. Plain text P can be obtain using the given formula. where, d = decryption key

Step – 1: Select two prime numbers p and q where $p \neq q$.

Example, Two prime numbers p = 13, q = 11.

Step – 2: Calculate n = p * q.

Example, n = p * q = 13 * 11 = 143.

Step – 3: Calculate $\Phi(n) = (p-1) * (q-1)$.

Example, $\Phi(n) = (13 - 1) * (11 - 1) = 12 * 10 = 120$.

Step – 4: Select e such that, e is relatively prime to $\Phi(n)$, i.e. $(e, \Phi(n)) = 1$ and $1 < e < \Phi(n)$.

Example, Select e = 13, gcd (13, 120) = 1.

Step – 4: Select e such that, e is relatively prime to $\Phi(n)$, i.e. $(e, \Phi(n)) = 1$ and $1 < e < \Phi(n)$.

Example, Select e = 13, gcd (13, 120) = 1.

Step – 5: Calculate $d = e^{-1} \bmod \Phi(n)$ or $e * d = 1 \bmod \Phi(n)$

Example, Finding d: $e * d \bmod \Phi(n) = 1$

$13 * d \bmod 120 = 1$

$d = ((\Phi(n) * i) + 1) / e$

$d = (120 + 1) / 13 = 9.30$ ($\because i = 1$)

$d = (240 + 1) / 13 = 18.53$ ($\because i = 2$)

$d = (360 + 1) / 13 = 27.76$ ($\because i = 3$)

$d = (480 + 1) / 13 = 37$ ($\because i = 4$))

Step – 6: Public key = {e, n}, private key = {d, n}.

Example, Public key = {13, 143} and private key = {37, 143}.

Step – 7: Find out *cipher text* using the formula, $C = P^e \bmod n$ where, P < n.

Example, Plain text P = 13. (Where, P < n)

$C = P^e \bmod n = 13^{13} \bmod 143 = 52$.

Step – 8: $P = C^d \bmod n$. Plain text P can be obtain using the given formula.

Example, Cipher text C = 52

$P = C^d \bmod n = 52^{37} \bmod 143 = 13$.

Question: P and Q are two prime numbers. P=7, and Q=17. Take public key E=5. If plain text value is 6, then what will be cipher text value according to RSA algorithm? Again calculate plain text value from cipher text.

Solution:

1. Two prime numbers P=7, Q=17
2. n = P * Q = 17 * 7 = 119 **n = 119**
3. Φ(n) = (P–1) * (Q –1) = (17 – 1) * (7 – 1) = 16 * 6 = 96 **Φ(n) = 96**
4. Public key E = 5. **E= 5**
5. Calculate d = 77. d = ((Φ(n) * i) + 1) / e **d= 77**
   d = ((96*1)+1) / 5 = 19.4
   d = ((96*2)+1) / 5 = 38.6
   d = ((96*3)+1) / 5 = 57.8
   d = ((96*4)+1) / 5 = 77 (Stop finding d because getting integer value)
6. Public key = {e, n} = {5, 119}, private key = {d, n} = {77, 119}.
7. Plain text PT = 6, CT = PT^E mod n = 6^5 mod 119 = 41. **Cipher Text = 41**
8. Cipher text CT = 41, PT = CT^d mod n = 41^{77} mod 119 = 6. **Plain Text = 6**

Question: In a public key cryptosystem using RSA algorithm, user uses two prime numbers 5 and 7. He chooses 11 as Encryption key, find out decryption key. What will be the cipher text, if the plaintext is 2? Decrypt the cipher text, what will be the value of plain text?

Solution:

1. Two prime numbers p = 5, q = 7
2. n = p * q = 5 * 7 = 35 **n = 35**
3. Φ(n) = (p–1) * (q –1) = (5 – 1) * (7 – 1) = 4 * 6 = 24 **Φ(n) = 24**
4. Public key e = 11. **e= 11**
5. Calculate d = 11. d = ((Φ(n) * i) + 1) / e **d= 11**
6. Public key = {e, n} = {11, 35}, private key = {d, n} = {11, 35}.
7. Plain text P = 2, C = P^e mod n = 2^{11} mod 35 = 18. **Cipher Text = 18**
8. Cipher text C = 18, P = C^d mod n = 18^{11} mod 35 = 2. **Plain Text = 2**

Question: In a public key cryptosystem using RSA algorithm, user uses two prime numbers 5 and 7. He chooses 11 as Encryption key, find out decryption key. What will be the cipher text, if the plaintext is 2? Decrypt the cipher text, what will be the value of plain text?

Solution:

1. Two prime numbers p = 5, q = 7
2. n = p * q = 5 * 7 = 35 **n = 35**
3. Φ(n) = (p–1) * (q –1) = (5 – 1) * (7 – 1) = 4 * 6 = 24 **Φ(n) = 24**
4. Public key e = 11. **e= 11**
5. Calculate d = 11. d = ((Φ(n) * i) + 1) / e **d= 11**
6. Public key = {e, n} = {11, 35}, private key = {d, n} = {11, 35}.
7. Plain text P = 2, C = P^e mod n = 2^{11} mod 35 = 18. **Cipher Text = 18**
8. Cipher text C = 18, P = C^d mod n = 18^{11} mod 35 = 2. **Plain Text = 2**

Hash Function:

- Cryptographic Hash is a Hash function that takes random size input and yields a fixed-size output. It is easy to calculate but challenging to retrieve the original data.
- It is strong and difficult to duplicate the same hash with unique inputs and is a one-way function so revert is not possible. Hashing is also known by different names such as Digest, Message Digest, Checksum, etc

Properties of Cryptography Hash Function:

The ideal cryptographic hash function has the following main properties:

1. **Deterministic:** This means that the same message always results in the same hash.
2. **Quick:** It is quick to compute the hash value for any given message.
3. Avalanche Effect: This means that every minor change in the message results in a major change in the hash value.
4. **One-Way Function:** It is not possible to reverse the cryptographic hash function to get to the data.
5. **Collision Resistance:** It is infeasible to find two different messages that produce the same hash value.
6. **Pre-Image Resistance:** The hash value shouldn't be predictable from the given string and vice versa.

Applications of Hash Functions:

Message authentication:

It is a mechanism or service used to verify the integrity of a message.

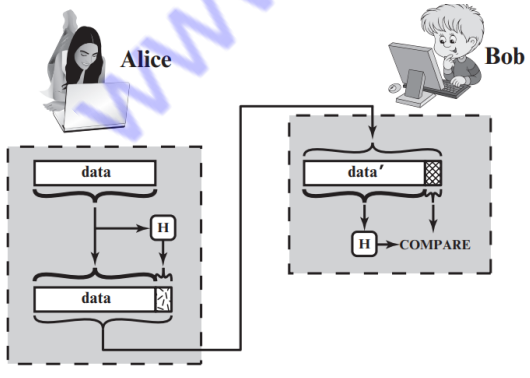
Message authentication assures that data received are exactly as sent (i.e., there is no modification, insertion, deletion, or replay).

When a hash function is used to provide message authentication, the hash function value is often referred to as a message digest.

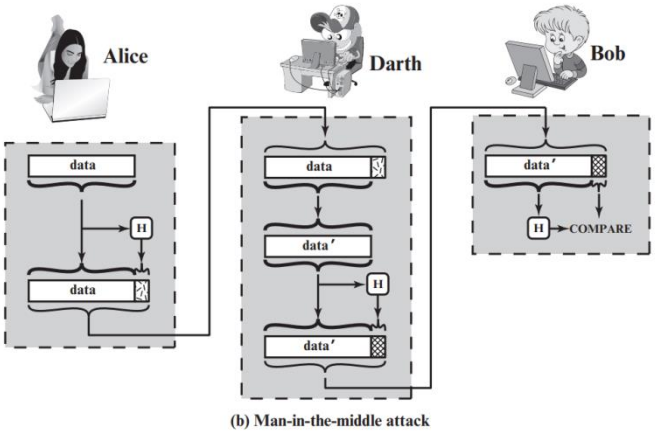
The sender computes a hash value as a function of the bits in the message and transmits both the hash value and the message. The receiver performs the same hash calculation on the message bits and compares this value with the incoming hash value.

If there is a mismatch, the receiver knows that the message has been altered. The hash value must be transmitted in a secure fashion.

That is, the hash value must be protected so that if an adversary alters or replaces the message, it is not feasible for adversary to also alter the hash value to fool the receiver. In this example, Alice transmits a data block and attaches a hash value. Darth intercepts the message, alters or replaces the data block, and calculates and attaches a new hash value. Bob receives the altered data with the new hash value and does not detect the change. To prevent this attack, the hash value generated by Alice must be protected.



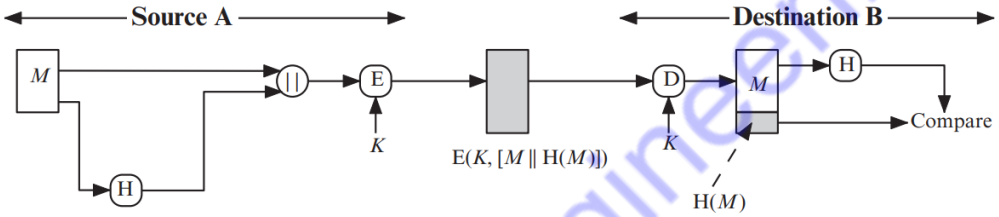
(a) Use of hash function to check data integrity



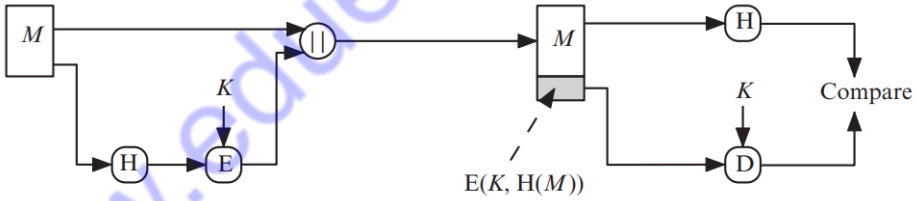
(b) Man-in-the-middle attack

A variety of ways in which a hash code can be used to provide message authentication, as follows.

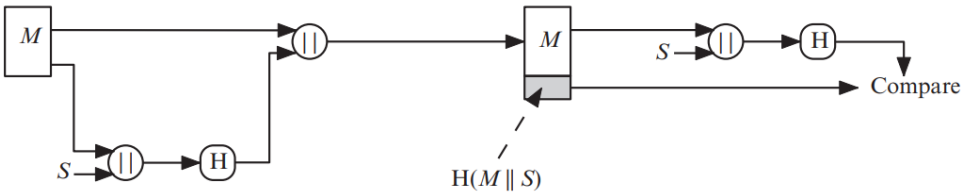
a. The message plus concatenated hash code is encrypted using symmetric encryption. Because only A and B share the secret key, the message must have come from A and has not been altered. The hash code provides the structure or redundancy required to achieve authentication. Because encryption is applied to the entire message plus hash code, confidentiality is also provided.



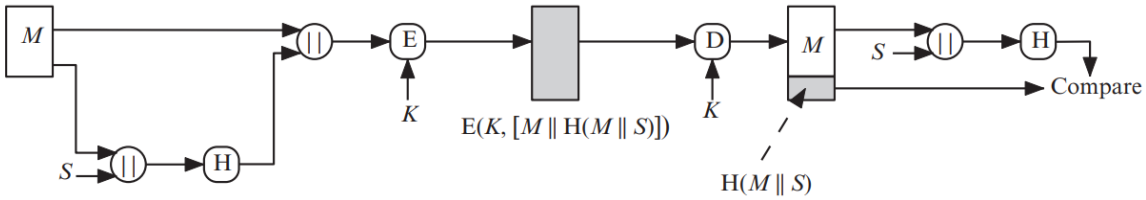
b. Only the hash code is encrypted, using symmetric encryption. This reduces the processing burden for those applications that do not require confidentiality



c. It is possible to use a hash function but no encryption for message authentication. The technique assumes that the two communicating parties share a common secret value S. A computes the hash value over the concatenation of M and S and appends the resulting hash value to M. Because B possesses S, it can recompute the hash value to verify. Because the secret value itself is not sent, an opponent cannot modify an intercepted message and cannot generate a false message.

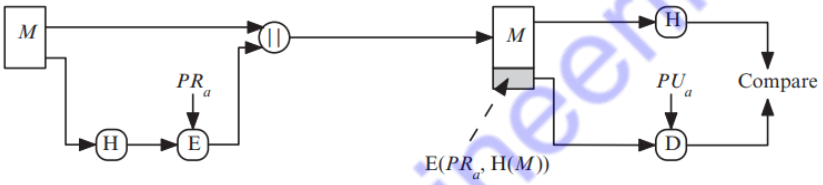


d. Confidentiality can be added to the approach of method (c) by encrypting the entire message plus the hash code.

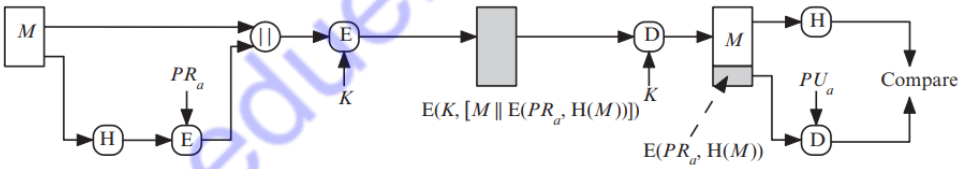


Digital Signature:

- An important application, which is similar to the message authentication application, is the digital signature. The operation of the digital signature is similar to that of the MAC. In the case of the digital signature, the hash value of a message is encrypted with a user's private key.
- Anyone who knows the user's public key can verify the integrity of the message that is associated with the digital signature. In this case, an attacker who wishes to alter the message would need to know the user's private key. The hash code is encrypted, using public-key encryption with the sender's private key.
- It also provides a digital signature, because only the sender could have produced the encrypted hash code. In fact, this is the essence of the digital signature technique.



- If confidentiality as well as a digital signature is desired, then the message plus the private-key-encrypted hash code can be encrypted using a symmetric secret key. This is a common technique.

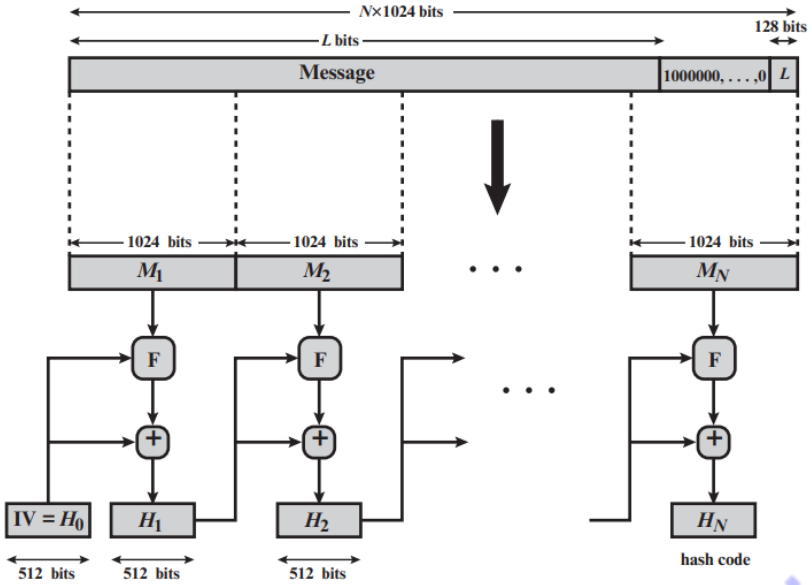


Secure Hash Algorithm:

- SHA developed by National Institute of Standards and Technology (NIST) and published as a federal information processing standard (FIPS 180) in 1993.
- SHA-1 produces a hash value of 160 bits. In 2002, NIST produced a revised version of the standard, FIPS 180-2, that defined three new versions of SHA, with hash value lengths of 256, 384, and 512 bits, known as SHA-256, SHA-384, and SHA-512, respectively.

SHA-512 Logic:

- The algorithm takes as input a message with a maximum length of less than 2128 bits and produces as output a 512-bit message digest. The input is processed in 1024-bit blocks.
  Step 1: Append padding bits. The message is padded so that its length is congruent to 896 modulo 1024 [length K 896(mod 1024)]. Padding is always added, even if the message is already of the desired length. Thus, the number of padding bits is in the range of 1 to 1024. The padding consists of a single 1 bit followed by the necessary number of 0 bits.



Step 2 Append length. A block of 128 bits is appended to the message. This block is treated as an unsigned 128-bit integer (most significant byte first) and contains the length of the original message in bits (before the padding). The outcome of the first two steps yields a message that is an integer multiple of 1024 bits in length. In the expanded message is represented as the sequence of 1024-bit blocks M_1, M_2….M_N, so that the total length of the expanded message is N * 1024 bits.

Step 3 Initialize hash buffer. A 512-bit buffer is used to hold intermediate and final results of the hash function. The buffer can be represented as eight 64-bit registers (a, b, c, d, e, f, g, h).

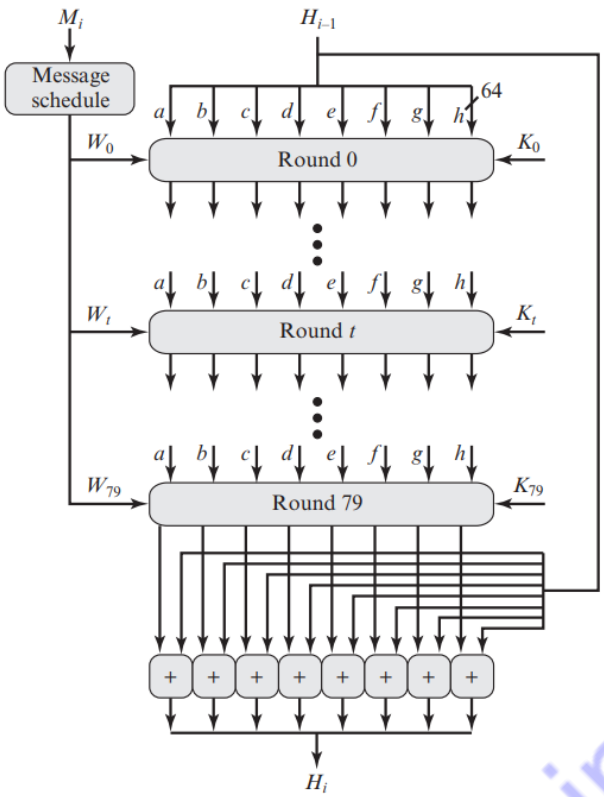
These registers are initialized to the following 64-bit integers (hexadecimal values):

a = 6A09E667F3BCC908 b = BB67AE8584CAA73B c = 3C6EF372FE94F82B
d = A54FF53A5F1D36F1 e= 510E527FADE682D1 f=9B05688C2B3E6C1F
g = 1F83D9ABFB41BD6B h = 5BE0CD19137E2179

These values are stored in big-endian format, which is the most significant byte of a word in the low-address (leftmost) byte position. These words were obtained by taking the first sixty-four bits of the fractional parts of the square roots of the first eight prime numbers

Step 4 Process message in 1024-bit (128-byte) blocks.

- The heart of the algorithm is a module that consists of 80 rounds.
- Each round takes as input the 512-bit buffer value, abcdefgh, and updates the contents of the buffer.
- At input to the first round, the buffer has the value of the intermediate hash value, Hi-1. Each round t makes use of a 64-bit value W_t, derived from the current 1024-bit block being processed (Mi). These values are derived using a message schedule described subsequently.
- Each round also makes use of an additive constant K_t , where 0 … t … 79 indicates one of the 80 rounds. These words represent the first 64 bits of the fractional parts of the cube roots of the first 80 prime numbers.
- The constants provide a “randomized” set of 64-bit patterns, which should eliminate any regularity in the input data.
- The output of the eightieth round is added to the input to the first round (Hi-1) to produce Hi.
- The addition is done independently for each of the eight words in the buffer with each of the corresponding words in Hi-1, using addition modulo 264.



Step 5 **Output.** After all N 1024-bit blocks have been processed, the output from the Nth stage is the 512-bit message digest.

We can summarize the behavior of SHA-512 as follows:

$$H_0 = \text{IV}$$
$$H_i = \text{SUM}_{64}(H_{i-1}, \text{abcdefgh}_i)$$
$$MD = H_N$$

where

IV = initial value of the abcdefgh buffer, defined in step 3

abcdefgh_i = the output of the last round of processing of the ith message block

N = the number of blocks in the message (including padding and length fields)

SUM_{64} = addition modulo 2^{64} performed separately on each word of the pair of inputs

MD = final message digest value

Advanced Encryption Standard Problems

Given the plaintext {000102030405060708090A0B0C0D0E0F} and the key {01010101010101010101010101010101}

a. Show the original contents of State, displayed as a 4×4 matrix.

b. Show the value of State after initial Add Round Key.

c. Show the value of State after Sub Bytes.

d. Show the value of State after Shift Rows

e. Show the value of State after Mix Columns

$$\text{state} = \begin{bmatrix} 00 & 04 & 08 & 0C \\ 01 & 05 & 09 & 0D \\ 02 & 06 & 0A & 0E \\ 03 & 07 & 0B & 0F \end{bmatrix}$$

Add 0^{th} round key

$$\text{Key} = \begin{bmatrix} 01 & 01 & 01 & 01 \\ 01 & 01 & 01 & 01 \\ 01 & 01 & 01 & 01 \\ 01 & 01 & 01 & 01 \end{bmatrix}$$

$$\begin{bmatrix} 00 & 04 & 08 & 0C \\ 01 & 05 & 09 & 0D \\ 02 & 06 & 0A & 0E \\ 03 & 07 & 0B & 0F \end{bmatrix} \oplus \begin{bmatrix} 01 & 01 & 01 & 01 \\ 01 & 01 & 01 & 01 \\ 01 & 01 & 01 & 01 \\ 01 & 01 & 01 & 01 \end{bmatrix} = \begin{bmatrix} 01 & 05 & 09 & 0D \\ 00 & 04 & 08 & 0C \\ 03 & 07 & 0B & 0F \\ 02 & 06 & 0A & 0E \end{bmatrix}$$

c. Byte Substitution

$$\begin{bmatrix} 01 & 05 & 09 & 0D \\ 00 & 04 & 08 & 0C \\ 03 & 07 & 0B & 0F \\ 02 & 06 & 0A & 0E \end{bmatrix} \rightarrow \begin{bmatrix} 7C & 6B & 01 & D7 \\ 63 & F2 & 30 & FE \\ 7B & C5 & 2B & 76 \\ 77 & 6F & 67 & AB \end{bmatrix}$$

d. Shifting Rows

$$\begin{bmatrix} 7C & 6B & 01 & D7 \\ 63 & F2 & 30 & FE \\ 7B & C5 & 2B & 76 \\ 77 & 6F & 67 & AB \end{bmatrix} \rightarrow \begin{bmatrix} 7C & 6B & 01 & D7 \\ F2 & 30 & FE & 63 \\ 2B & 76 & 7B & C5 \\ AB & 77 & 6F & 67 \end{bmatrix}$$

e. Mixing Column

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 7C & 6B & 01 & D7 \\ F2 & 30 & FE & 63 \\ 2B & 76 & 7B & C5 \\ AB & 77 & 6F & 67 \end{bmatrix} = \begin{bmatrix} 74 & E7 & 0F & A2 \\ 55 & E6 & 04 & 22 \\ 3E & 2E & B8 & 8C \\ F6 & 15 & 58 & 0B \end{bmatrix}$$

EDU ENGINEERING

PIONEER OF ENGINEERING NOTES

TAMIL NADU'S BEST EDTECH PLATFORM FOR ENGINEERING

CONNECT WITH US

WEBSITE: www.eduengineering.net

TELEGRAM: @eduengineering

INSTAGRAM: @eduengineering

- **Regular Updates for all Semesters**
- **All Department** Notes AVAILABLE
- **Handwritten** Notes AVAILABLE
- **Past Year** Question Papers AVAILABLE
- **Subject wise** Question Banks AVAILABLE
- **Important Questions** for Semesters AVAILABLE
- Various **Author Books** AVAILABLE