2 Symmetric and Asymmetric Key Cryptography and Key Management

Syllabus

At the end of this unit, you should be able to understand and comprehend the following syllabus topics :

- Block cipher principles
- Data Encryption Standard (DES)
  - Double DES
  - Triple DES
  - Block cipher modes of operation
- Advanced Encryption Standard (AES)
- Stream Cipher: RC4 algorithm
- Public key cryptography
  - Principles of public key cryptosystems
  - RSA Algorithm
  - Knapsack Algorithm
  - Key management techniques
  - Using symmetric and asymmetric algorithms and trusted third party
    - Symmetric key agreement: Diffie Hellman
    - Symmetric Key Distribution
      - Needham-Schroeder protocol
      - Kerberos: Kerberos Authentication protocol
  - KDC
    - Public key Distribution
    - Digital Certificate: X.509
    - PKI

2.1 Concept Building - Types of Symmetric Algorithms (Ciphers)

Symmetric key based algorithms (ciphers) can work either on blocks of bits (characters) or one bit at a time.

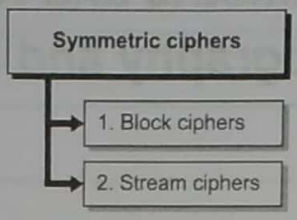


Fig. 2.1.1 : Symmetric Ciphers

2.1.1 Block Ciphers

✍ ***Definition :*** *The algorithms that work on blocks are called block ciphers.*

In block ciphers, the information that needs to be encrypted is broken into smaller and equal block sizes. Then, the encryption operation (substitution and transposition) is applied to each block. The resultant ciphertext from each block is then combined to produce the encrypted information.

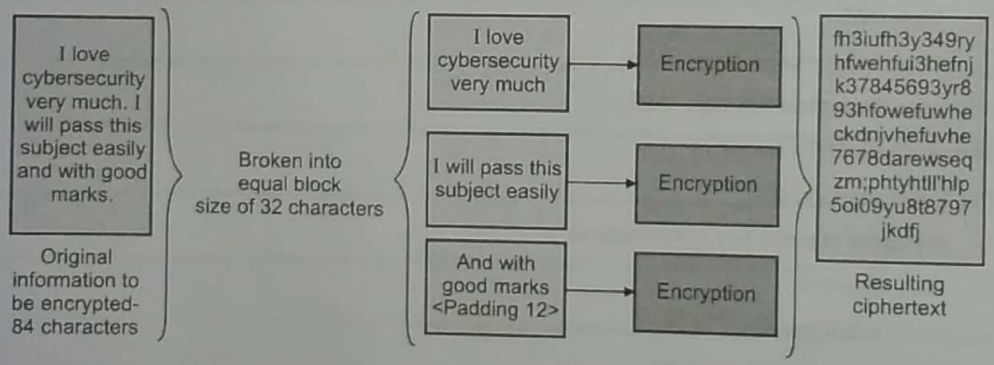


Fig. 2.1.2 : Block Ciphers

This is a highly simplified block diagram of how a block cipher works. The information is broken into equal size blocks and then the encryption operation is carried out on each block. If the block size has lesser number of characters than required to form a block, then padding is done to fill the block. Padding is just filling some temporary information to form a block. Finally, the resulting encrypted information from each block is combined to get the overall encrypted message.

DES and AES are two of the examples of Symmetric Block Ciphers.

2.1.2 Stream Ciphers

✍ ***Definition :*** *The algorithms that work on one-bit at a time are called stream ciphers.*

Unlike block ciphers, stream ciphers work on one bit of plaintext at a time. Each bit of plaintext is combined with the bits of security key and then XORed to get ciphertext.

Note : If you recall from your logical design classes, the Table 2.1.1 is truth table of XOR. For result to be 1, both the inputs should be different.

Table 2.1.1 : XOR Truth Table

Sr. No.	Input X	Input Y	Output Z (XOR X, Y)
1.	0	0	0
2.	0	1	1
3.	1	0	1
4.	1	1	0

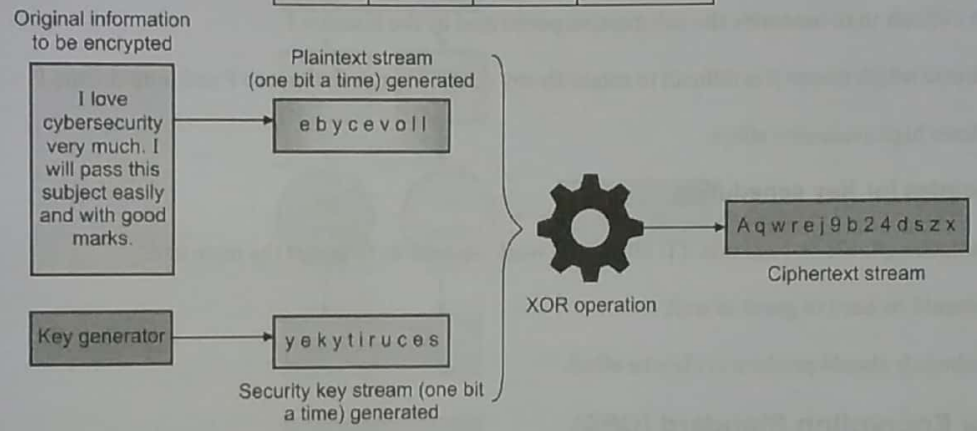


Fig. 2.1.3 : Stream Ciphers

RC4 is an example of stream cipher.

2.1.3 Comparison between Block and Stream Cipher

Q. Compare and contrast - Block and stream ciphers. **MU - Dec. 16, 3 Marks**

Sr. No.	Comparison Attribute	Block Cipher	Stream Cipher
1.	Security	High	Low
2.	Speed	Low	High
3.	Application	Non-real time such as documents	Real time data such as Voice
4.	Commonly used	Yes	No

2.1.4 Block Cipher Principles

There are three critical components in designing a block cipher.

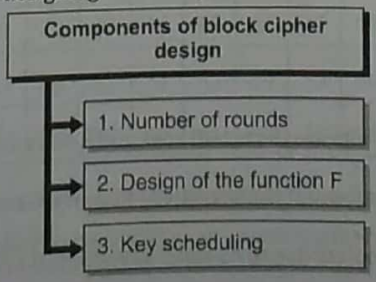


Fig. 2.1.4 : Components of Block Cipher Design

Following are the principles around each of these.

Design Principles for Number of Rounds in the Block Cipher Algorithm

1. The greater the number of rounds, the more difficult it is to perform cryptanalysis.
2. The number of rounds is chosen such that a known cryptanalysis takes a greater effort compared to brute-force attack.

Design Principles for function F (Feistel network) in the Block Cipher Algorithm

1. It must be difficult to re-assemble the substitution performed by the function F.
2. F is non-linear which means it is difficult to establish any relation between input to F and output from F.
3. F should have high avalanche effect.

Design Principles for Key scheduling

1. Subkey selection should be such that it is difficult to work backwards to derive the main key.
2. Subkeys should be hard to guess as well.
3. The key schedule should produce avalanche effect.

2.2 Data Encryption Standard (DES)

Q.	Explain working of DES detailing the Fiestel structure.	MU - Dec. 15, 10 Marks
Q.	Explain working of DES.	MU - May 16, 10 Marks
Q.	Explain DES, detailing the Fiestel structure and S-block design.	MU - May 17, 10 Marks
Q.	What is the purpose of S-boxes in DES? Explain the avalanche effect?	MU - May 18, 5 Marks

✍ ***Definition :*** *Data Encryption Standard (DES) is a symmetric key based block cipher standard used for encryption and decryption.*

It came into existence and usage around Nov 1976 and was predominantly used in the industry until 2002.

Major attributes of DES

- It is a symmetric key based algorithm.
- It works as a block cipher.
- It uses 64-bit blocks.
- It uses a key size of 64-bits in which 56-bits are the actual keys and 8-keys are used for error detection.
- It uses 16 rounds of operation (substitution and transposition) to convert a block of plaintext into ciphertext.
- DES is now considered insecure and obsolete due to its short key-size (56-bits only).

2.2.1 Block Diagram and Internals of DES

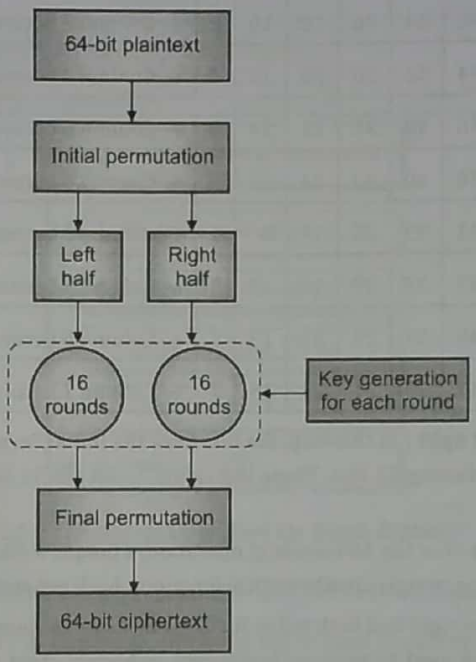


Fig. 2.2.1 : Block diagram of DES

This is very simplistic view of DES. Let us understand what happens at each stage.

Step 1 - Creation of 64-bit blocks : In this step, the plaintext information to be encrypted is broken into 64-bit blocks. DES is a block cipher and block creation is similar to as explained in the earlier section.

Table 2.2.1 : 64 bits of Plaintext

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

Step 2 - Initial Permutation : In this step, the 64 bits in the plaintext blocks are re-arranged (transposed). This is done as per the diffusion property of the cipher to ensure that any small variance in plaintext produces a large variance in the ciphertext.

Table 2.2.2 : Initial Permutation (Re-arrange bits of Plaintext)

58	50	42	34	26	18	10	2	← Column 2 becomes 1st row
60	52	44	36	28	20	12	4	← Column 4 becomes 2nd row
62	54	46	38	30	22	14	6	← Column 6 becomes 3rd row
64	56	48	40	32	24	16	8	← Column 8 becomes 4th row
57	49	41	33	25	17	9	1	← Column 1 becomes 5th row
59	51	43	35	27	19	11	3	← Column 3 becomes 6th row
61	52	45	37	29	21	13	5	← Column 5 becomes 7th row
63	55	47	39	31	23	15	7	← Column 7 becomes 8th row

- **Step 3 - Left Half and Right Half Split :** In this step, the bits from the Initial Permutation stage are split into two parts – left half and right half each containing 32-bits. These individual 32-bit blocks are then continuously worked through the 16 rounds of operation.
- **Step 4 - Subkey Key Generation :** For the 16 rounds of operation, a unique subkey is derived for each round from the 56-bit key. The key is derived using complex mathematical functions. Each generated subkey is 48-bit long.
- **Step 5 - Rounds :** Left half and the right half both individually go through 16 rounds of encryption operation. In each of the rounds, the derived subkey is used to produce temporary ciphertext. This temporary ciphertext produced after each round is used in the next round until the final round is complete. Each round consists of substitutions and successive permutations.
- **Step 6 - Final Permutation :** In the last stage, we need to bring the bits back to their respective positions. The bit positions were changed at the initial permutation stage.

Table 2.2.3 : Final Permutation (Re-arrange bits of Ciphertext)

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

- **Step 7 - Final Ciphertext :** Once all the steps are done, you get the final ciphertext for the plaintext given the security key of your choice via DES.

✍ ***Exam Tip :*** *IF you hear that an algorithm is broken or is insecure, it means that it is computationally feasible to find out the key used for encryption. Note that cryptography heavily depends upon our understanding of mathematics and the computation power available today. What is secure and infeasible today, could be insecure and feasible to crack in future.*

2.2.2 Block Cipher Modes of Operation (for DES and other Block Ciphers in General)

Q.	Explain with examples the CBC and ECB modes of block ciphers.	MU - Dec. 16, 3 Marks
Q.	Discuss in detail block cipher modes of operation.	MU - May 19, 10 Marks

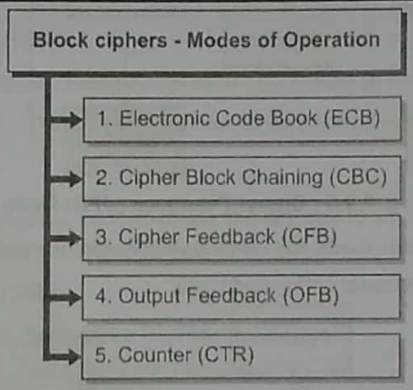


Fig. 2.2.2 : Modes of Operation for Block Ciphers

DES and other block ciphers can potentially work in several modes. Let's review them carefully.

1. **Electronic Code Book (ECB) Mode :** In this mode, the same key is used to encrypt all the blocks. Key derivatives or subkeys are not used. Additionally, each block is treated separately and the ciphertext of previous block does not influence successive blocks.

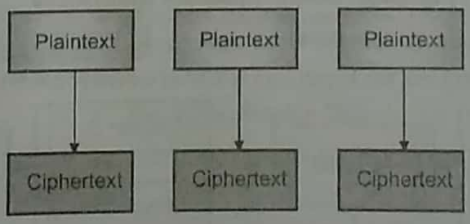


Fig. 2.2.3 : Electronic Code Book (ECB) Mode

2. **Cipher Block Chaining (CBC) Mode :** In this mode, the ciphertext of previous block is used with the next plaintext block. The two blocks (ciphertext of previous block and plaintext of next block) are XORed and then passed through the encryption operation. This generates a lot more randomness in the final ciphertext.

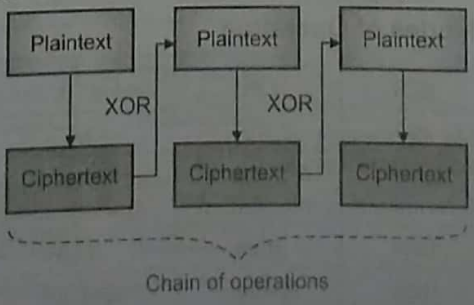


Fig. 2.2.4 : Cipher Block Chaining (CBC) Mode

3. **Cipher Feedback (CFB) Mode :** In this mode, the block cipher works like a stream cipher. The ciphertext from the previous block is XORed with the key (keystream) for the next block. This way the key increasingly becomes random and brings more randomness in the overall encryption process.

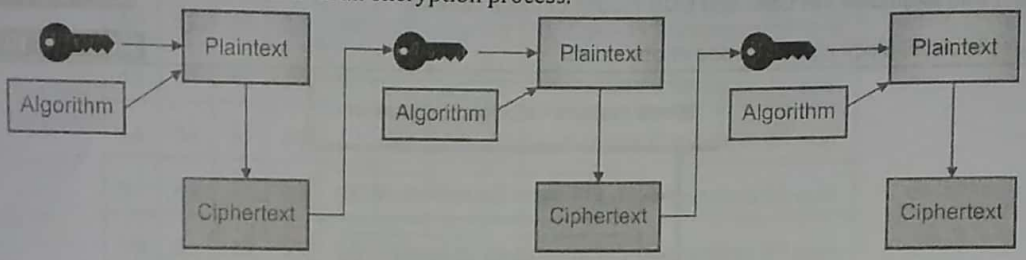


Fig. 2.2.5 : Cipher Feedback (CFB) Mode

4. **Output Feedback (OFB) Mode :** In this mode, the block cipher works like a stream cipher. The keystream used in the previous block is XORed with the keystream of the next block.

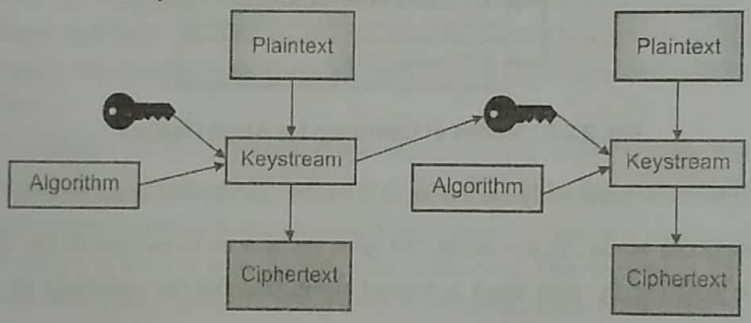


Fig. 2.2.6 : Output Feedback (OFB) Mode

5. **Counter (CTR) Mode :** In this mode as well, the block cipher works like a stream cipher. The key is converted into keystream (as used in stream cipher) and the keystream is XORed with a counter that increases for every block.

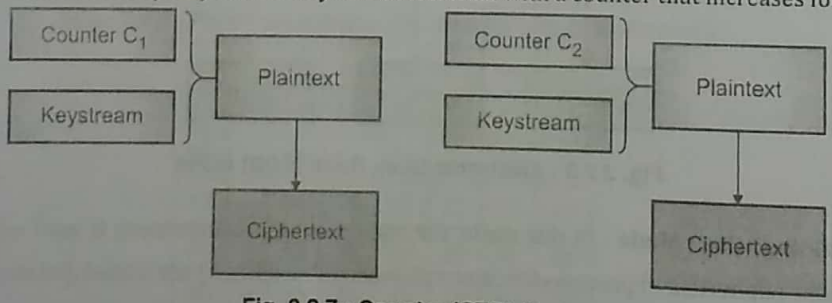


Fig. 2.2.7 : Counter (CTR) Mode

2.2.3 Comparison between Modes of Operation

Sr. No.	Mode	ECB	CBC	CFB	OFB	CTR
1.	In-Parallel block encryption	Yes	No	No	No	Yes
2.	Suited for	Small Information	Any size of information	Small Information	Small Information	Any size of information
3.	Security and randomness	Low	High	High	High	High
4.	Speed	High	Medium	Medium	Medium	High
5.	Complexity	Low	High	High	High	Low
6.	Works like stream cipher?	No	No	Yes	Yes	Yes

Weakness in DES

1. **Small key size** : 56-bits of keys have a keyspace (possible values) of 2^{56}. While that might seem a lot, it is actually not given the compute power we have today. In 1990s, the compute power we had was significantly lower and hence was considered secure at that time.

✍ ***Definition*** : *The type of attack where each combination is tried in an attempt to find the right combination is also called as brute force attack.*

2. **Prone to linear cryptanalysis** : DES has been proven to be susceptible to linear cryptanalysis.
3. **Prone to differential cryptanalysis** : DES has been proven to be susceptive to differential cryptanalysis.

2.2.4 Double DES

In order to strengthen DES, it was considered to increase the key size to 112-bits effectively. The way it was chosen to do so was to use 2 keys of 56-bits each. Let's call them K_1 and K_2.

Mathematically, it can be denoted as below :

Ciphertext C = Encryption (K_2, Encryption (K_1,Plaintext P))

Plaintext P = Decryption (K_1, Decryption (K_2,Ciphertext C))

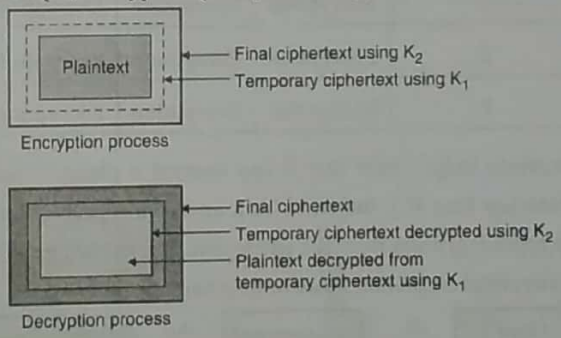


Fig. 2.2.8 : Double DES

- During the encryption process, first the plaintext is encrypted with Key K_1 and then the result is again encrypted with Key K_2 to get the final ciphertext for plaintext.
- During the decryption process, first the Key K_2 is used to decrypt to get the ciphertext that Key K_1 can decrypt to get the plaintext.
- However, Double DES was proven to be ineffective. Meet in the middle attack was shown to reduce the complexity to just 2^{57}(2^{56} attempts made twice, hence $2x2^{56} = 2^{57}$) instead of 2^{112} as originally thought.

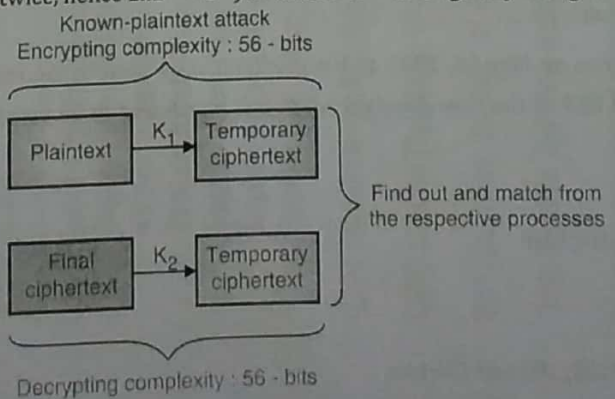


Fig. 2.2.9 : Complexity in Double DES

- So, using K_1 if you could derive temporary ciphertext using encryption process and using K_2 if you could also derive the same temporary ciphertext using decryption process, you have found a match and the keys you chose (K_1 and K_2) are now known to you. Hence, you could effectively find both the keys and break Double DES without original thought of complexity of 112 bits.
- Hence, Double DES was not adopted in the industry and is not used.

2.2.5 3DES or Triple DES

Q. Write short notes on 3DES. **MU - May 19, 4 Marks**

Finding that Double DES was ineffective, Triple DES or 3DES was conceived. 3DES uses 48 rounds of operation and can work in the following modes using two or three keys.

Table 2.2.4 : 3DES or Triple DES

Mode	Number of keys	Key 1	Key 2	Key 3
DES-EEE3	3	Encryption	Encryption	Encryption
DES-EDE3	3	Encryption	Decryption	Encryption
DES-EEE2	2	Encryption	Encryption	Encryption Using Key 1
DES-EDE2	2	Encryption	Decryption	Encryption Using Key 1

You might wonder how decryption helps? Note that if you encrypt a plaintext using a key (say K_1) and run the decryption process using a different key (say K_2), the text (from encryption process using K_1) becomes more random. The use of a different key in the decryption process from the encryption process brings added randomness and hence helps to make attacks such as linear or differential cryptanalysis extremely hard.

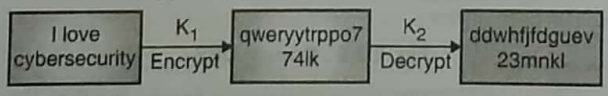


Fig. 2.2.10 : Decryption Process

2.3 Advanced Encryption Standard (AES)

Definition : *Advanced Encryption Standard (AES) is a symmetric key based block cipher standard used for encryption and decryption.*

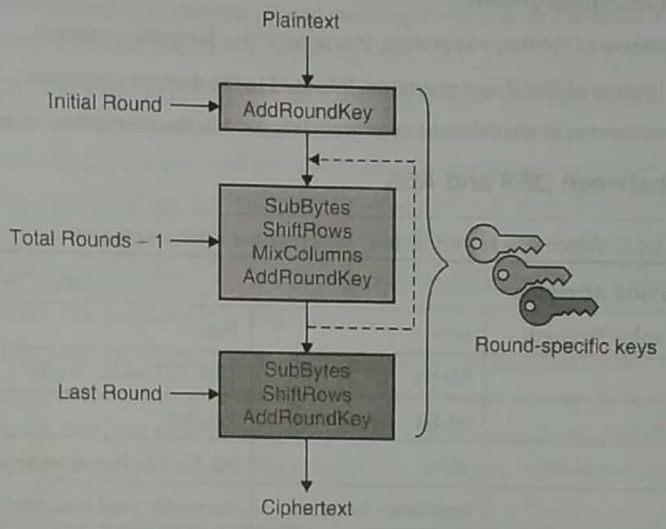
The standard became effective on May 26, 2002 and is predominantly used in the industry today due to its strong cipher properties. AES replaced DES as the new standard when DES was found to be insecure and vulnerable to various attacks.

Major attributes of AES

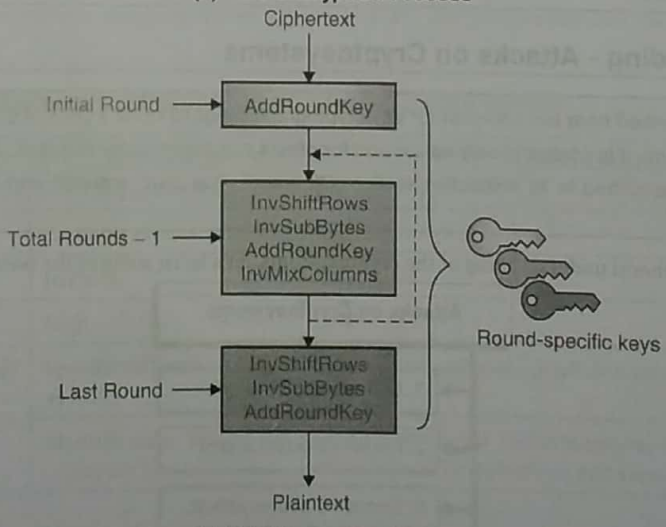
- It is a symmetric key based algorithm.
- It works as a block cipher.
- It uses 128-bit blocks.
- It can work with key sizes of 128, 192 and 256 bits.
- Number of rounds of operation depends upon the key size.

- 128-bit key undergoes 10 rounds
- 192-bit key undergoes 12 rounds
- 256-bit key undergoes 14 rounds
- AES is considered highly secure due to its long key sizes and is used in the industry today.

2.3.1 Block Diagram and Internals of AES



(a) AES Encryption Process



(b) AES Decryption Process

Fig. 2.3.1 : Block diagram of AES

Let's understand the blocks.

1. **AddRoundKey** : In this transformation step, a round key is generated and XORed with the intermediate (temporary) ciphertext. This block is used in both encryption as well as decryption process.

2. **SubBytes :** In this transformation step, the intermediate ciphertext undergoes various substitution operations. It is used for encryption process.
3. **ShiftRows :** In this transformation step, the intermediate ciphertext undergoes various row-wise transposition operations. It is used for encryption process.
4. **MixColumns :** In this transformation step, the intermediate ciphertext undergoes various column-wise transposition operations.It is used for encryption process.
5. **InvSubBytes :** This is inverse of SubBytes operation. It is used in the decryption process.
6. **InvShiftRows :** This is inverse of ShiftRows operation. It is used in the decryption process.
7. **InvMixColumns :** This is inverse of MixColumns operation. It is used in the decryption process.

2.3.2 Comparison between DES and AES

Q. Compare AES and DES. Which one is bit oriented? Which one is byte oriented? **MU - May 19, 5 Marks**

Comparison Attribute	DES	AES
Cryptographic Strength	Low	High
Key Size	56-bit	128, 192 and 256 bits
Block Size	64-bit	128-bit
Rounds	16	10, 12, 14 - based on key size
Usage	Obsolete - Not used	Currently used industry standard
Orientation	DES is bit oriented	AES is byte oriented

2.4 Concept Building - Attacks on Cryptosystems

Note : The attacks described here are common for any cryptographic algorithm be it DES, AES, RSA or any other. While for some algorithms it is comparatively easier and for others it is theoretically possible. Any specific attack against an algorithm is described in its respective section. Otherwise, you could mention and elaborate on the following attacks.

Now that you have a general understanding of the cryptosystems, let's learn some of the possible attacks on them.

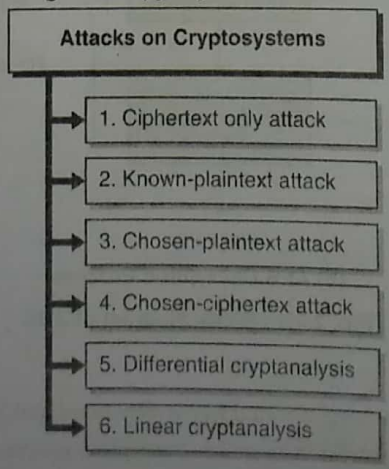


Fig. 2.4.1 : Attacks on Cryptosystems

1. **Ciphertext-only attack :** In this type of attack, the attacker has ciphertext of several messages. The algorithm is known, and the goal of the attack is to find out the key used in encryption. Once the key is found out, it is possible to decrypt messages that were encrypted using the key.
2. **Known-Plaintext attack :** The attacker knows the plaintext partially and the corresponding ciphertext. The goal is to find out the key. Once the key is known, the attacker can then use the key to decrypt ciphertext for which she does not know the plaintext. For example, you might be using a fixed greeting in your messages (For example, "Dear Friend") or you might be sending same message (for example, "Good morning") everyday to someone. The attacker could know this and also the corresponding ciphertext and try to find out the key.
3. **Chosen-Plaintext attack :** The attacker knows the exact plaintext and the corresponding ciphertext. The goal is to find out the key. Once the key is known, the attacker can then use the key to decrypt ciphertext for which she does not know the plaintext. For example, the attacker can send you a message that she knows that you will definitely forward to your friends. While forwarding, you might encrypt the message using your key. Now, the attacker can grab the ciphertext that you sent, and she already knows the plaintext message she sent you earlier.
4. **Chosen-Ciphertext attack :** In this attack, the attacker choses the ciphertext that she wants to be decrypted and know the corresponding plaintext. The goal again is to find out the key.
5. **Differential Cryptanalysis :** In this attack, the attacker choses a pair of plaintexts and follows through each stage in their respective encryption process and compare the difference between the results at each stage. The key used in encrypting the pair is same. The goal again is to figure out the key by carefully studying the differences in results at various stages between the pair. Since, the attacker chooses the plaintexts, differential analysis is considered to be a type of chosen-plaintext attack.
6. **Linear Cryptanalysis :** The attacker carries out a known-plaintext attack and tries to figure out the key. She evaluates the input and output at various stages of the encryption process and tries to find out the probability of specific key values.

2.4.1 Comparison between Differential and Linear Cryptanalysis

Comparison Attribute	Differential Cryptanalysis	Linear Cryptanalysis
Plaintext selection	Carefully chosen	Any random plaintext
Plaintext used	In Pairs	One by one
Complexity of attack	High	Low
Mathematical relation between plaintexts used	Specific differences (such as XOR)	Linear approximation (such as a series of XOR operations)
Goal of the attack	Identify some bits of the unknown key	Identify the linear relation between some bits of the plaintext, some bits of the ciphertext and some bits of the unknown key

2.5 Public Key Cryptography

Q. Elaborate the steps of key generation using RSA algorithm. **MU - May 16, 5 Marks**

Q. Briefly define idea behind RSA and also explain :
(i) Give public key and private key.
(ii) Describe security in this system. **MU - May 17, 5 Marks**

You learnt about the use of asymmetric keys earlier. Recall and revise that section before you proceed.

Public key cryptography relies on the use of such asymmetric keys for providing various cryptographic services such as encryption and digital signature.

✍ ***Definition :*** *Public key cryptography is a cryptographic scheme that uses two mathematically related keys, a public key and a private key, for providing various cryptographic services.*

2.5.1 Principles of Public Key Cryptosystems

Following are some basic principles of public key-based cryptosystems.

1. Public key cryptosystems require the use of two keys – a public key and a private key.
2. Public keys are widely known.
3. Private key is kept secret with its owner.
4. The two keys are mathematically related and form a key pair.
5. One key in the key pair cannot be used to derive the other key in the key pair.
6. Any key in the key pair can be used for encryption. The other key then must be used for decryption.
7. Sender and the receiver both must have their own key pairs.

In this section, you are going to learn about various asymmetric key based algorithms.

2.5.2 RSA Algorithm

RSA, named after its inventors Ron Rivest, Adi Shamir and Leonard Adleman, is an asymmetric key based algorithm. As you understand, RSA, or any other asymmetric key based algorithms can be used for confidentiality [encryption, decryption], authentication and non-repudiation.

RSA is based on finding prime factors for very large numbers. The length of numbers that we are referring to here is around 500 digits!

RSA Key Length	Number of digits
1024-bit	309
2048-bit	617
4096-bit	1233

Let's understand how RSA derives public and private keys and how does encryption and decryption process work based on the derived keys.

1. Choose two random large prime numbers, p and q.
2. Multiply the numbers. $n = p*q$.
3. Choose a random integer to be encryption key e such that e and $(p - 1)(q - 1)$ are relatively prime.
4. Decryption key is computed as $d = e^{-1} \bmod ([p - 1]*[q - 1])$.
5. The public key = (n, e).
6. The private key = (n, d).
7. For encrypting message M with public key (n, e), you get ciphertext $C = M^e \bmod n$.
8. For decrypting ciphertext with private key (n, d), you get plaintext $M = C^d \bmod n$.

Ex. 2.5.1 : Perform encryption and decryption using RSA algorithm with p = 7, q = 11, e = 17 and M = 8.

Soln. :

$$n = p * q$$

$$n = 7 * 11 = 77$$

$$r = (p - 1) * (q - 1)$$

$$r = 6*10 = 60$$

$$d = e^{-1} \bmod r$$

$$ed \equiv 1 \bmod 60$$

$$17d \equiv 1 \bmod 60$$

Let's calculate modulo inverse using extended Euclidean algorithm (swapping 17 and 60)

Index i	quotient q for i	Remainder r for i	s for i	t for i
0		60	1	0
1		17	0	1
2	60 / 17 = 3	60 – 3*17 = 9	1 – 3*0 = 1	0 – 3*1 = –3
3	17 / 9 = 1	17 – 1*9 = 8	0 – 1*1 = –1	1 – 1*–3 = 4
4	9 / 8 = 1	9 – 1*8 = **1**	1 – 1*–1 = **2**	–3 – 1*4 = **–7**
5	8 / 1 = 8	8 – 1*8 = 0	Do not calculate	Do not calculate

Let's re-swap the values.

$$x = -7$$

$$y = 2$$

Putting the values in the extended Euclidean algorithm, you get

$$ax + by = 1$$

$$17(-7) + 60(2) = 1$$

Since, you have to find multiplicative inverse in mod 60, divide both sides by mod 60.

$$17(-7) \bmod 60 + 60(2) \bmod 60 = 1 \bmod 60$$

$$17(-7) \bmod 60 + 0 = 1$$

Recall our discussion on calculating mod for negative numbers. You need to keep adding mod until the number turns positive and then calculate mod on the positive number you got.

$$-7 + 60 = 53$$

$$53 \bmod 60 = 53$$

Hence, $17(53) \bmod 60 = 1$

Hence, value of decrypting key, $d = 53$

Now, you have all the values needed for encryption and decryption.

As per RSA,

$$C = M^e \bmod n$$

$C = 8^{17} \bmod 77$

$\mathbf{C = 57}$

$M = C^d \bmod n$

$M = 57^{53} \bmod 77$

$\mathbf{M = 8}$

Ex. 2.5.2 : In an RSA system the public key (e, n) of user A is defined as (7,119). Calculate $\phi(n)$ and private key d. What is the ciphertext when you encrypt message m =10, using the public key? **MU - Dec. 15, 10 Marks**

Soln. :

Since, n is 119, assuming p and q were 7 and 17 respectively (by factorizing 119).

$r = (p-1)*(q-1)$

$r = 6*16 = 96$

phi(n) can be calculated using the formula

$$\text{phi}(n) = n * \left(1 - \frac{1}{7}\right) * \left(1 - \frac{1}{17}\right) \text{ [where 7 and 17 are prime factors of 119)}$$

Hence, $\text{phi}(n) = 96$

$d = e^{-1} \bmod r$

$ed \equiv 1 \bmod 96$

$7d \equiv 1 \bmod 96$

Let's calculate modulo inverse using extended Euclidean algorithm (swapping 7 and 96)

Index i	quotient q for i	Remainder r for i	s for i	t for i
0		96	1	0
1		7	0	1
2	96 / 7 = 13	96 – 7* 13 = 5	1 – 0*13 = 1	0 – 1*13 = – 13
3	7 / 5 = 1	7 – 5 * 1 = 2	0 – 1*1 = –1	1 – (– 13)*1 = 14
4	5 / 2 = 2	5 – 2* 2 = 1	1 – (– 1)*2 = 3	– 13 – 14*2 = **– 41**
5	2 /1 = 2	2 – 1*2 = 0	Do not calculate	Do not calculate

Let's re-swap the values.

$x = -41$

$y = 3$

Putting the values in the extended Euclidean algorithm, you get

$ax + by = 1$

$7*(-41) + 96*(3) = 1$

Since, you have to find multiplicative inverse in mod 96, divide both sides by mod 96.

$7*(-41) \bmod 96 + 96*(3) \bmod 96 = 1 \bmod 96$

$7*(-41) \bmod 96 + 0 = 1 \bmod 96$

$7*(-41) \bmod 96 = 1 \bmod 96$

You need to keep adding mod until the number turns positive and then calculate mod on the positive number you got.

$-41 + 96 = 55$

$55 \bmod 96 = 55$

Hence, multiplicative inverse is 55.

Therefore, the value of decrypting key, $d = 55$.

Now, you have all the values needed for encryption and decryption.

As per RSA,

$C = M^e \bmod n$

$C = 10^7 \bmod 119$

$C = 73$ [encrypted message]

$M = C^d \bmod n$

$M = 73^{55} \bmod 119$

$M = 10$ [decrypted message]

Attacks on RSA

1. **Brute-force attack :** Here the attacker tries to find factors of n by trying out various possibilities.
2. **Common Modulus :** To avoid generating a different modulus n = p*q for each user one may wish to fix n for all the users. It might seem like deriving the decrypting key d is not possible for every encrypting key e by any other user since encrypting key e is a randomly chosen value. But, the problem with this approach is that a particular user who knows her pair of e and d can successfully use her own pair to find the factors for common modulus. Once the factors are known, since the encrypting key e is known to everyone (because it is public key), the decrypting key d could be found out. Hence, you should not be using common modulus to generate keys for multiple users.
3. **Choosing smaller numbers :** The security of the algorithm comes from the fact that factoring large numbers is computationally intensive. Sometimes, to improve system performance, smaller numbers can be chosen which can significantly enhance the performance but at the cost of making the algorithm weaker. Hence, you should always choose large numbers to maintain the strength of the algorithm.
4. **Man in the middle attack :** The attacker could collect all the ciphertext coming out from the user's system (that is encrypted with her private key) and try to find the private key. The information known to the attacker is the ciphertext and public key.

2.5.3 Knapsack Algorithm

✍ ***Definition :*** *The Merkle-Hellman Knapsack algorithm is an asymmetric key based algorithm used for encryption and decryption.*

It was invented by Ralph Merkle and Martin Hellman in 1978.

Major attributes of the Knapsack Algorithm

- It is based on public key cryptography meaning that it requires two keys – public and private.
- Unlike RSA, it is one way. Public key is used for encryption and private key is used for decryption.
- Hence, it cannot be used for digital signature and non-repudiation.
- It is outdated and is not used anymore.

Algorithm

- Knapsack means a bag. The basic Knapsack problem aims at maximizing the weight of the knapsack taking values from a set of available weights.

> ***Definition :*** *Given a Knapsack of a maximum capacity of W and N items each with its own value and weight, choose items to place inside the Knapsack such that the final contents in the knapsack has the maximum value.*

In Merkle-Hellman Knapsack Algorithm, the keys are two knapsacks.

- The public key is a 'hard' knapsack A, and
- The private key is an 'easy', or super-increasing, knapsack B.

Two numbers, a multiplier and a modulus, can be used to convert the super-increasing knapsack B into the hard knapsack A.

So,

- Given a set of numbers *A* and a number *b*.
- Find a subset of *A* which sums to *b*.

So, if you have a set of weights of 1, 4, 6, 8 and 15, and you want to get a weight of 29, you could thus use 6, 8 and 15 (6 + 8 + 15 = 29). So, the code would become 00111 (represented by not picking '1', not picking '4', picking '6', picking '8' and picking '15').

Plaintext	10001	11001	00001	00101
Knapsack	1,4,6,8,15	1,4,6,8,15	1,4,6,8,15	1,4,6,8,15
Ciphertext	1+15 = 16	1+4+15 = 20	15	6+15 = 21

2.6 Key Management Techniques

Now that you understand various cryptographic methods and procedures, let us focus on the keys. As you know, keys play the critical role in delivering several cryptographic services. Without adequate and protected methods of key generation, distribution, and disposal, the entire cryptosystem could collapse, and you might not be able to use any cryptographic services.

2.6.1 What is a Key?

What is a Key? As per National Institute of Standards and Technology (NIST) :

> ***Definition :*** *A key is a parameter used in conjunction with a cryptographic algorithm that determines its operation in such a way that an entity with knowledge of the key can reproduce, reverse or verify the operation, while an entity without knowledge of the key cannot.*

Let us recall the keys used in various cryptographic services assuming that there are two parties involved in the communication and they both wish to use cryptographic services.

Cryptographic Service	Keys used
Symmetric Key Encryption / Decryption	One symmetric key
Asymmetric Key Encryption / Decryption	Two Public Private Key Pairs
Hashing	No Keys
MACs	One symmetric Key
Digital Signature	Two Public Private Key Pairs

2.6.2 Key States

It is important for you to understand that the keys could be in various states and can transition from one state to another state based on circumstances and scenarios.

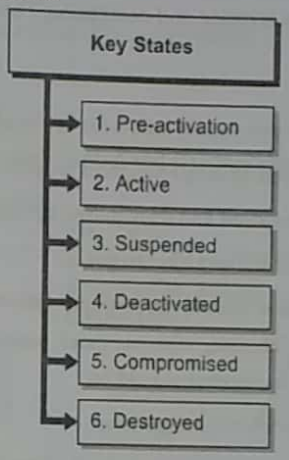


Fig. 2.6.1 : Key States

1. **Pre-activation state :** In this state, the key is generated but is not yet in use. This state is reached as soon as the key is generated.
2. **Active state :** In this state, the key can be used to provide cryptographic services.
3. **Suspended state :** The key in this state is restricted temporarily for use. The key can be transitioned to the active, de-active or destroyed state based on various criteria.
4. **Deactivated state :** In this state, the key is not used to carry out new cryptographic services. However, it may be used for previously consumed cryptographic services. For example, a deactivated key may not be used for encrypting new information but can still be used for decrypting previous information that it was used to encrypt when it was in the active state.
5. **Compromised state :** In this state, the key is exposed, and its secrecy can no more be guaranteed. Once a key is determined to be in this state, it is strongly advisable to suspend its use, carry out investigation to find the root cause of exposure and then destroy it.
6. **Destroyed state :** In this state, the key is safely disposed. The key is no more available to carry out any form of cryptographic services.

2.6.3 Cryptoperiod (Key lifetime)

All cryptographic keys come with a validity period. Some keys are short-lived, and some are long-lived.

✍ ***Definition :*** *A cryptoperiod is the time span during which a specific key is authorised for use by legitimate entities, or the keys for a given system will remain in effect.*

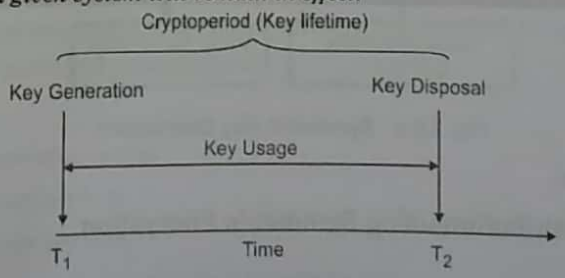


Fig. 2.6.2 : Cryptoperiod (Key Lifetime)

- Master Keys are usually long-lived. They are not used for providing cryptographic services themselves. Instead, they are used to periodically generate session keys.
- Session Keys are usually short-lived (up to a session). These are generated from the master keys and are used for providing cryptographic services such as encryption and decryption. These are frequently generated and destroyed.

2.6.4 Key Management Principles

Following are some general principles that you should follow for key management.

1. **Adequate key length :** The key length should be long enough to provide the necessary level of protection.
2. **Secure transmission :** The keys should be stored and transmitted by secure means.
3. **Difficult to guess :** The keys should be random, and hard to guess.
4. **Adequate Cryptoperiod :** The key's lifetime should correspond with the sensitivity of the data it is protecting. Low sensitive data may allow for a longer key lifetime, whereas more sensitive data might require a shorter key lifetime. Also, the more the key is used, the shorter its lifetime should be.
5. **Backup :** The keys should be backed up in case of emergencies.
6. **Adequate key disposal :** The keys should be properly destroyed when they expire.

2.6.5 Symmetric Key Distribution

Symmetric key distribution requires that the communication parties have the exact same key. The same key is used for cryptographic services between the parties.

Ways to distribute symmetric keys

Assume that there are two communicating parties – *A* and *B*. The symmetric key can be distributed in the following ways :

1. A and B can physically generate and distribute the key amongst each other.
2. Any third party C can physically generate and distribute the key to both A and B.
3. A and B can get connected to a key distribution center and get the key.

In reality, distributing the keys physically is infeasible. Hence, a third party, generally known as the Key Distribution Center, is tasked to distribute the keys amongst the communicating parties.

The symmetric key distribution can be achieved in two ways.

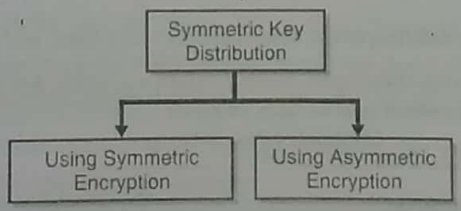


Fig. 2.6.3 : Symmetric Key Distribution

Let's learn about both of them.

2.6.5(A) Symmetric Key Distribution using Symmetric Encryption

Let's assume the following.

- *A* and *B* are the two communicating parties.

- There is a **trusted third party** known as Key Distribution Center (KDC) denoted by *K*.
- *A* and *K* have a shared master key *Ka*.
- Similarly, *B* and *K* have a shared master key *Kb*.
- *Na* is a random number called nonce generated by *A* to avoid replays.
- *Ida* is the unique identifier for the party *A*.
- *IDb* is the unique identifier for the party *B*.
- *Ks* is the desired session key that needs to reach both *A* and *B* so that they can securely communicate.

The steps for key distribution are as following.

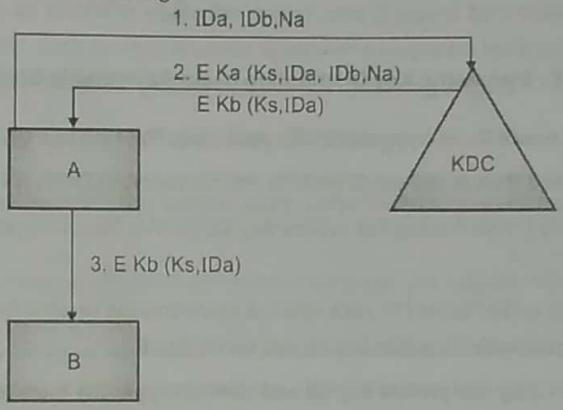


Fig. 2.6.4 : Symmetric Key Distribution using Symmetric Encryption

1. The communicating party A initiates a request with KDC by identifying itself as IDa and identifying the target party as IDb. It sends a nonce Na to ensure that the request is protected against the replay attacks.
2. The KDC generates the desired session key Ks and encrypts it with the master secret shared between it and A, Ka. It also sends back the original request from A to ensure that the response matches with the initial request. It also sends another copy of session key Ks and IDa encrypted using the master secret shared between it and B, Kb. This copy can only be decrypted by B.
3. A verifies the information received from KDC, stores the session key Ks and forwards the copy intended for B.
   B decrypts the copy, stores the session key Ks and gets to know that the communicating party is A with whom the received session key is to be used.

   This way, both *A* and *B* receive the session key Ks and can then begin secure communication.

2.6.5(B) Symmetric Key Distribution using Asymmetric Encryption

Using symmetric key encryption to distribute symmetric keys requires KDC. Using asymmetric encryption for key distribution requires the use of public and private key pairs.

Let's assume the following.

- *A* and *B* are the two communicating parties.
- A's public key is Pa and private key is Sa.
- B's public key is Pb and private key is Sb.
- *Na* is a random number called nonce generated by *A* to avoid replays.
- *Nb* is a random number called nonce generated by *B* to avoid replays.

- *IDa* is the unique identifier for the party *A*.
- *IDb* is the unique identifier for the party *B*.
- The session key to be distributed is Ks.

The steps for key distribution are as following.

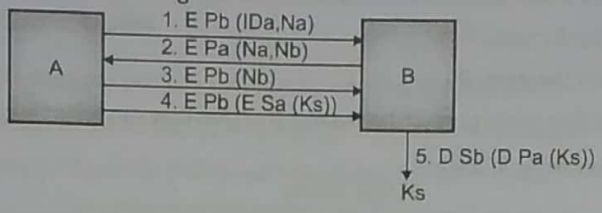


Fig. 2.6.5 : Symmetric Key Distribution using Asymmetric Encryption

1. A sends its identity IDa and a nonce Na encrypted with B's public key Pb.
2. B decrypts the message received from A and sends back Na and Nb encrypted with A's public key.
3. A decrypts the message received from B using her private key Sa, verifies Na, encrypts Nb with Pb and sends it back to B.
4. At this point, both A and B are authenticated to each other. A generates the session key Ks encrypts it with her private key Sa and then encrypts it again with B's public key Pb and sends it to B.
5. B first decrypts the message using her private key Sb and then decrypts the message again using A's public key Pa. B now gets the session key Ks and knows that only A could have used her private key to send the session key to her.

2.6.6 Asymmetric Key Distribution (Distribution of Public Keys)

Asymmetric keys involve public keys and private keys. Public keys are known in general. Private keys are kept secure and are in constant possession of their respective owners. The goal of asymmetric key distribution is to distribute public keys only. Private keys are not distributed.

Ways to distribute asymmetric keys

Asymmetric keys can be distributed using the following ways.

Fig. 2.6.6 : Asymmetric Key Distribution

1. **Broadcast :** Broadcast or public announcement of public keys is the simplest of all the distribution techniques. In this, the public keys are constantly broadcasted (shared with the larger user community). As new users get into the community, they hear such key broadcast messages and store the public keys to use if and when required.

2. **Public directory :** A centralized and publicly open directory (very much like an old telephone directory) could be established to register and distribute public keys. Users can register new keys, change previous keys or delete the keys if required. The company maintaining the directory could exercise control to ensure that the directory is highly available, and the key information contained there-in is not destroyed or maligned.
3. **Public-key authority :** In Public-key authority, the central authority itself has a public and private key pair. When a user asks for the public key of the desired target user, the authority encrypts the public key information of the desired user with its own private key to securely distribute the public key.
4. **Certificates :** Certificate is the most widely used technique to distribute public keys in the industry today. A certificate typically consists of a public key, an identifier of the key owner, and is signed by a trusted third party. Typically, the third party is a certificate authority, such as a government agency or a corporate institution, that is trusted by the user community. You will learn about certificates in greater detail in the upcoming section.

2.6.7 Key Management using Diffie Hellman Key Exchange

Q. Explain how a key is shared between two parties using Diffie Hellman key exchange algorithm. What is the drawback of this algorithm? **MU - Dec. 15, 10 Marks**

Q. Explain "Diffie-Hellmen" key exchange algorithm with suitable example. Also explain the problem of MIM attack in it. **MU - May 17, 10 Marks**

If you recall, one of the challenges with using symmetric key algorithms was key distribution. How could sender and the receiver agree upon the key that would be used for encryption and decryption (recall that the symmetric key based algorithms use the SAME key for both encryption as well as decryption)? I asked you this question in the symmetric key section and here I am again to help you with the answer. Diffie-Hellman algorithm is one of the answers to the question.

✍ ***Definition :*** *The Diffie-Hellman algorithm provides a way of generating a shared secret between the sender and the receiver in such a way that the secret need not be exchanged or transferred over the communication medium.*

Basically, the sender and the receiver create the key together at their respective ends at the same time. The key, that they create at their respective ends, is mathematically computed to be the same. Hence, the key distribution need not happen, and the sender and the receiver can confidentially communicate using the key they created. You should note here that the Diffie-Hellman algorithm is NOT used for actual encryption and decryption process. It is used only for key generation.

Let's understand the steps involved in generating the shared key. Assume that there are two users Alex and Bobby who need to generate a shared key for securely communicating with each other.

1. Alex chooses two prime numbers g and p and also a secret number a. He calculates value of A such that $A = g^a \bmod p$. He then sends g, p and A to Bobby. Note here that Alex does not share the secret number a with Bobby.
2. Similarly, Bobby chooses a secret number b and computes the value of B such that $B = g^b \bmod p$. She then sends B to Alex.
3. Alex computes the shared key at his end as Shared Key, $S = B^a \bmod p$
4. Bobby computes the shared key at her end as Shared Key, $S = A^b \bmod p$

The values of the shared key, *S*, derived in the step 3 and 4 are equal due to mod operation.

- $(g^a \bmod p)^b \bmod p = g^{ab} \bmod p$
- $(g^b \bmod p)^a \bmod p = g^{ba} \bmod p$

It does not matter which step you do earlier. Both the keys created would be equal and can be used with any of the algorithms such as DES and AES to encrypt the information and communicate confidentially.

Let's understand the algorithm by solving a question.

Ex. 2.6.1 : Calculate shared key between two users if the initial chosen prime numbers are 5 and 7.

Soln. :

$$g = 5$$

$$p = 7$$

Assume that the user Alex chooses a secret number $a = 2$

$$A = g^a \bmod p$$

$$A = 5^2 \bmod 7$$

$$A = 25 \bmod 7$$

$$A = 4$$

Alex sends g, p and A to Bobby.

Assume that the user Bobby chooses a secret number $b = 3$

$$B = g^b \bmod p$$

$$B = 5^3 \bmod 7$$

$$B = 125 \bmod 7$$

$$B = 6$$

Bobby sends B to Alex

Now, both the users compute the shared key, S, at their respective ends.

Alex calculates it as,

$$S = B^a \bmod p$$

$$S = 6^2 \bmod 7$$

$$S = 36 \bmod 7$$

$$S = 1$$

Bobby calculates it as,

$$S = A^b \bmod p$$

$$S = 4^3 \bmod 7$$

$$S = 64 \bmod 7$$

$$S = 1$$

So, the shared key, that can be used between Alex and Bobby, is $S = 1$.

Note : The example we took here involves very small numbers to make it easy for you to understand the steps involved in the key generation. Practically, the numbers used in the shared key generation are extremely large, possibly containing around 500 digits!

Ex. 2.6.2 : Users A and B use the Diffie-Hellman key exchange technique with a common prime 71 and primitive root 7. Show that 7 is primitive root of 71. If user A has private key x = 5, what is A's Public Key R_1? If user B has private key y =12, what is B's public key R_2? What is the shared secret key? **MU - May 19, 10 Marks**

Soln. :

- If the multiplicative order of a number r modulo n is equal to Euler Totient Function $\Phi(n)$, then it is a primitive root.
- $\Phi(71) = 70$ (all the 70 numbers before 71 are coprime with 71).
- Let's assume that 7 is a primitive root of 71. Then, it should be true that

  $\Phi(71)^7 \bmod 71 = \Phi(71)$.
- Calculate $70^7 \bmod 71$. This gives 70 which is equal to $\Phi(71)$. Hence, 7 is indeed a primitive root of 71.

  $g = 7$

  $p = 71$

For user A

- Private key x = 5 which means

  $A = g^x \bmod p$

  $A = 7^5 \bmod 71$

  $A = 51$
- Hence, the public key of user A is 51.

For user B

- Private key Y = 12 which means

  $B = g^y \bmod p$

  $B = 7^{12} \bmod 71$

  $B = 4$
- Hence, the public key of user B is 4.
- Now, both the users compute the shared key, S, at their respective ends.
- User A calculates it as,

  $S = B^x \bmod p$

  $S = 4^5 \bmod 71$

  $S = 30$
- User B calculates it as,

  $S = A^y \bmod p$

  $S = 51^{12} \bmod 71$

  $S = 30$
- Hence, the shared key between User A and User B is 30.

2.7 Digital Certificate - X.509

Q. Give the format of X.509 digital certificate and explain the use of a digital signature in it. **MU - Dec. 15, 5 Marks**

Q. What is a digital certificate? How does it help to validate the authenticity of a user? Explain the X.509 certificate format. **MU - Dec. 17, 10 Marks**

Q. Write short notes on X.509. **MU - May 19, 4 Marks**

X.509 is a standard that defines the requirements for public key certificates.

Definition : *A certificate is a signed data structure that binds a public key to a person, computer, or organization.*

The certificate uniquely identifies the owner of a public key. Certificates are issued by Certification Authorities (CAs) such as Verisign, Global Sign etc. In a secure communication, certificates are used to identify the systems and establish trust amongst the communicating parties. Since its inception in 1998, three versions of the X.509 public key certificate standard have evolved. The most commonly used version of X.509 is version 3. X.509 certificates are used for secure communication such as establishing a https connection.

Table 2.7.1 : Contents of a X.509 version 3 certificate

Fields	Purpose	Example
Version	Identifies the version of the certificate	V3
Serial Number	Unique number for the certificate	79ad16a14aa0a5ad4c7358f407132e65
Signature algorithm	Algorithm used to create digital signature for certificate	sha1RSA
Signature hash algorithm	Algorithm used to calculate hash of certificate	sha1
Issuer	Name of certificate issuer	CN = Microsoft Root Certificate Authority DC = Microsoft DC = com
Valid from	Date from which certificate is valid	Thursday, May 10, 2001 4:49:22 AM
Valid to	Date until which certificate is valid	Monday, May 10, 2021 4:58:13 AM
Subject	Name of certificate owner	CN = Microsoft Root Certificate Authority DC = Microsoft DC = com
Public key	Public key	a5 94 ef 15 14 89 fd 4b 73 ... (4096 bits)
Issuer unique ID	ID of issuing Certificate Authority (CA)	03475573948593hgfyuerwe327e7e52513fc2ae3
Subject unique ID	ID of subject	0eac826040562797e52513fc2ae10a539559e4a4
Extensions	Optional Information	Thumbprint, Friendly Name, Key Usage, etc.

Note : If you use Microsoft® Windows® OS, you can go to (Windows Key + R) Run and type certmgr.msc. It would open up certificate manager where you can browse various certificates stored on your system. Double-click on any certificate and examine the various fields it has.

2.8 Public Key Infrastructure (PKI)

Asymmetric cryptography has been widely adopted to provide various cryptographic services such as encryption, digital signature, authentication and non-repudiation. Asymmetric cryptography is critically dependent on the availability of an ecosystem of public private key-pairs. In this ecosystem, the key-pairs need to be securely generated, transported, distributed, verified, used and disposed. Public Key Infrastructure (PKI) provides such an ecosystem of operation.

✍ ***Definition*** *: Public Key Infrastructure (PKI) consists of programs, data formats, procedures, communication protocols, security policies, and public key cryptographic mechanisms working in a comprehensive manner to operationalize asymmetric cryptography based cryptographic services.*

It supports the distribution, revocation and verification of public keys used for public key encryption and enables linking of identities with public key certificates.

Note : Do not confuse between Public Key Cryptography and Public Key Infrastructure. Public Key Cryptography refers to the use of asymmetric keys for cryptographic services whereas Public Key Infrastructure is about managing the entire lifecycle of those keys. It is more about infrastructure management around those keys.

2.8.1 Components of PKI

PKI has several components that work together in a comprehensive manner to operationalize the ecosystem of public-private keys. The major components of PKI are as following.

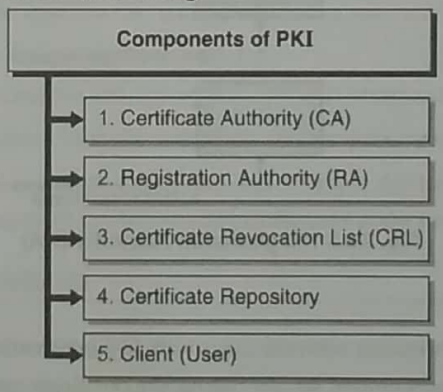


Fig. 2.8.1 : Components of PKI

1. Certificate Authority (CA)

✍ ***Definition :*** *The Certificate Authority (CA) is a trusted third party that provides the root of trust (highest level of trust) for all PKI certificates and provides services that can be used to authenticate the issued certificates.*

When the CA signs the certificate, it binds the certificate's identity to the public key, and the CA takes liability for the authenticity of that certificate. It is this trusted third party (the CA) that allows people who have never met to authenticate to each other and to communicate securely. A CA is a trusted organization (or server) that maintains and issues digital certificates. Some examples of CAs are Verisign, Comodo and Geotrust.

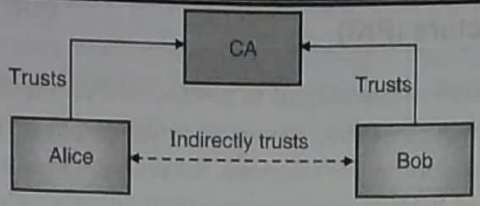


Fig. 2.8.2 : Certificate Authority (CA)

2. Registration Authority (RA)

✍ ***Definition :*** *The Registration Authority (RA) performs the certification registration duties. The RA establishes and confirms the identity of a certificate requester, initiates the certificate generation process with the CA, and manages the certificate life-cycle.*

The RA cannot issue certificates itself. It acts as a middle-man between the certificate requester and the associated CA. RA verifies all the necessary information before approving the request to get a certificate from the CA.

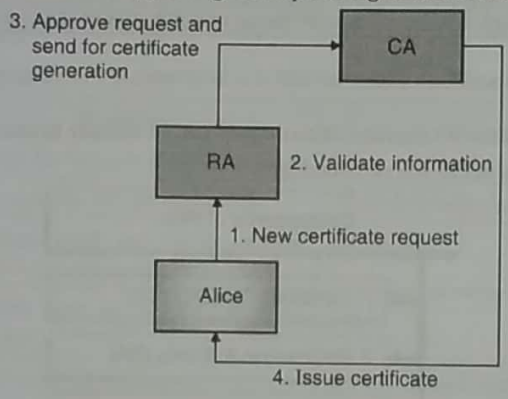


Fig. 2.8.3 : Registration Authority (RA)

3. Certificate Revocation List (CRL)

Certificates hold the public key information whereas the private key information should be kept secure with the user. In scenarios where the private key is compromised for any reason, the certificate cannot be trusted any further and hence it is crucial to let the world know that the certificate should not be trusted. Such a mechanism to notify the voidance of the certificate is called Certificate Revocation List (CRL).

✍ ***Definition :*** *A Certificate Revocation List (CRL) is a list of digital certificates that have been revoked by the issuing Certificate Authority (CA) before their scheduled expiration date and should no longer be trusted.*

CRLs provide the blacklist of certificates and are used by PKI clients to verify a certificate's validity and trustworthiness.

4. Certificate Repository

This is a generic database that stores both the valid as well as the invalid certificates. This database stores information about all the issued certificates. In addition to the certificate itself, the database includes validity period and status of each certificate. Certificate revocation is done by updating this database and marking the certificates as unfit for use.

5. Client

These are users, machines or other end point entities that use PKI infrastructure. They could request for new certificates, request certificates for other users (to get their public key) or carry out other certificate operations such as checking the CRL to confirm a certificate's validity.

Steps Involved in a Digital Certificate Generation

Following diagram provides the high-level steps involved in certificate generation.

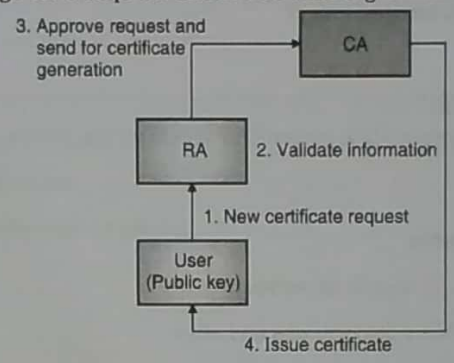


Fig. 2.8.4 : Steps involved in a Digital Certificate Generation

1. Any user, desiring to get a certificate from the CA, sends her information such as name, address, etc. along with her public key to the RA.
2. The RA verifies all the submitted information and approves the certificate request.
3. The RA forwards the information received from the user along with her public key to the CA.
4. The CA computes the hash value of the user's public key and encrypts the hash value with its own private key to generate a digital signature. It then sends all this information to the user in the form of a digital certificate.

2.9 Needham-Schroeder protocol

This is covered in Unit 4. Please refer Section 4.3.1.

2.10 Kerberos : Kerberos Authentication protocol

KDC and this topic is covered in Unit 4. Please refer Section 4.3.2.

Review Questions

Here are a few review questions to help you gauge your understanding of this chapter. Try to attempt these questions and ensure that you can recall the points mentioned in the chapter.

[A] Block Ciphers

Q. 1 Write a short note on block ciphers. **(6 Marks)**

Q. 2 List the design principles of block ciphers. **(8 Marks)**

Q. 3 What is Data Encryption Standard (DES)? List its major attributes. **(6 Marks)**

Q. 4 Draw a block diagram of DES and explain how it works. **(8 Marks)**

Q. 5	Write a short note on Electronic Code Book (ECB) Mode of Block Cipher operation.	**(4 Marks)**
Q. 6	Write a short note on Cipher Block Chaining (CBC) Mode of Block Cipher operation.	**(4 Marks)**
Q. 7	Write a short note on Cipher Feedback (CFB) Mode of Block Cipher operation.	**(4 Marks)**
Q. 8	Write a short note on Output Feedback (OFB) Mode of Block Cipher operation.	**(4 Marks)**
Q. 9	Write a short note on Counter (CTR) Mode of Block Cipher operation.	**(4 Marks)**
Q. 10	Compare various modes of Block Cipher operations.	**(8 Marks)**
Q. 11	Why is DES not recommended to be used today?	**(4 Marks)**
Q. 12	List the weaknesses in DES.	**(4 Marks)**
Q. 13	Write a short note on Double DES.	**(6 Marks)**
Q. 14	Write a short note on Triple DES.	**(6 Marks)**
Q. 15	Write a short note on 3DES.	**(6 Marks)**
Q. 16	What is AES? List its major attributes.	**(4 Marks)**
Q. 17	Draw the block diagram of AES and explain its working.	**(8 Marks)**
Q. 18	Compare AES and DES. Which one would you use and why?	**(8 Marks)**
Q. 19	Describe RC5 Algorithm.	**(8 Marks)**
Q. 20	Describe the various attacks possible on cryptosystems.	**(8 Marks)**
Q. 21	Compare differential and linear cryptanalysis.	**(8 Marks)**

[B] Public Key Cryptography

Q. 22	List the principles of public key cryptosystems.	**(6 Marks)**
Q. 23	Describe the various key states.	**(8 Marks)**
Q. 24	Write a short note on Cryptoperiod (Key lifetime).	**(4 Marks)**
Q. 25	List the Key Management Principles.	**(6 Marks)**
Q. 26	Describe Symmetric Key Distribution using Symmetric Encryption.	**(8 Marks)**
Q. 27	Describe Symmetric Key Distribution using Asymmetric Encryption.	**(8 Marks)**
Q. 28	Describe the ways to distribute asymmetric keys.	**(6 Marks)**
Q. 29	Describe the ways to distribute public keys.	**(6 Marks)**

[C] Digital Certificate - X.509

Q. 30	What is a X.509 Certificate? Why is it used?	**(4 Marks)**
Q. 31	What is a X.509 Certificate? Describe its contents.	**(8 Marks)**
Q. 32	What is a X.509 Certificate? Describe the steps involved in its generation.	**(8 Marks)**

[D] Public Key Infrastructure (PKI)

Q. 33	What is PKI? List its components.	**(8 Marks)**

❑❑❑