KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed University Established Under Section 3 of UGC Act 1956)

Coimbatore - 641021.

(For the candidates admitted from 2017 onwards)

DEPARTMENT OF COMPUTER SCIENCE, CA & IT

SUBJECT	: **CRYPTOGRAPHY AND NETWORK SECURITY**		
SEMESTER	: **IV**		**L T P C**
SUBJECT CODE: **17CAP404N**	**CLASS** : **II MCA**		**4 0 0 4**

SYLLABUS

Scope: To make student understand the goals, issues, technologies, algorithms, protocols and design criteria used in cryptography and data security and solution.

Objective:

- To teach fundamental aspects of security in a modern networked environment with the focus on system design aspects and cryptography in the specific context of network.
- To build protection mechanisms in order to secure computer networks.
- Write coding to encrypt "Plain Text" into "Cipher Text" and vice versa, using different encryption algorithms.
- The ability to choose a suitable ciphering algorithm according to the required security level.
- Build cryptosystems by applying encryption algorithms,
- Build secure authentication systems by use of message authentication techniques.

UNIT I

Introduction to Cryptography – Security Attacks – Security Services – Security Algorithm – Stream cipher and Block cipher – Symmetric and Asymmetric – Key Cryptosystem; Symmetric Key Algorithms: Introduction – DES – Triple DES – AES – IDEA – Blowfish – RC5.

UNIT II

Public Key Cryptosystem: Introduction to Number Theory – RSA Algorithm – Key Management – Diffie-Hell man key exchange – Introduction to Elliptic Curve Cryptography; Message Authentication and Hash functions – Hash and Mac Algorithm – Digital Signatures and Authentication Protocol.

UNIT III

Network Security Practice: Authentication Applications – Kerberos – X.509 Authentication Services and Encryption Techniques:; E-mail security – PGP – s/MIME – IP Security.

UNIT IV

Web Security – Secure Socket Layer – Secure Electronic Transaction; System Security – Intruders and Viruses – Firewalls – Password Security.

UNIT V

Case Study: Implementation of Cryptographic Algorithms – RSA – DSA – ECC (C / JAVA Programming). Network Forensic – Security Audit; Other Security Mechanism: Introduction to Stenography – Quantum Cryptography – Water Marking – DNA Cryptography.

SUGGESTED READINGS

1. William Stallings (2013), "Cryptography and Network Security", 6th Edition. Pearson Education, New Delhi.
2. Bruce Schneir (2006), "Applied Crptography", 2nd Edition. CRC Press, New Delhi.
3. A.Menezes, P.Van Oorschot and S.Vanstone(2010),"Hand Book of Applied Cryptography", 2nd Edition. CRC Press, NewDelhi.
4. Ankit Fadia.(2010), Network Security, 2nd Edition. McMillan India Ltd, New Delhi.

Web Sites

1. williamstallings.com/Crypto3e.html
2. u.cs.biu.ac.il/~herzbea/book.html
3. www.flipkart.com/search-books/cryptography+and+network+security+William+stallings+ebook

KARPAGAM ACADEMY OF HIGHER EDUCATION

(Deemed University Established Under Section 3 of UGC Act 1956)

Coimbatore - 641021.

(For the candidates admitted from 2017 onwards)

DEPARTMENT OF COMPUTER SCIENCE, CA & IT

SUBJECT : **CRYPTOGRAPHY AND NETWORK SECURITY**
SEMESTER : **IV** **L T P C**
SUBJECT CODE : **17CAP404N** **CLASS** : **II MCA** **4 0 0 4**

LECTURE PLAN

STAFF NAME: Dr. G. ANITHA

UNIT-1

S.NO	LectureDuration (Hr)	Topics to be Covered	Support Materials
1.	1	Introduction to cryptography Security Attacks	T1 : 1-3 T1: 9-22
2.	1	Security Services Security Algorithm	T1: 16- 19 T4: 29- 33
3.	1	Stream cipher, Block cipher	T4: 34- 57
4.	1	Symmetric and Asymmetric Key Cryptosystem	T1:pg 59- 61
5.	1	Symmetric Key Algorithms Introduction DES, Triple DES	T4: 51-73 W2
6.	1	AES IDEA	T4: 98-109 T4:75-97
7.	1	Blowfish, RC5	T1:pg 98-109 W2
8.	1	Recapitulation and Important Questions Discussions	
Total no.of Hours planned for Unit – I : 8 Hrs			

UNIT-2

S.NO	Lecture Duration (Hr)	Topics to be Covered	Support Materials
1.	1	Public Key Cryptosystem Introduction to Number Theory.	T1: 234-236 T1: 259-268
2.	1	RSA Algorithm	T1: 268- 280
3.	1	Key Management Diffie-Hellman key exchange	T1: 506-516 T4: 46-53
4.	1	Introduction to Elliptic Curve Cryptography.	T1: 310-313
5.	1	Message Authentication, Hash functions	T1: 331-334
6.	1	Hash and MAC Algorithm.	T1: 352-374
7.	1	Digital Signatures and Authentication Protocol.	T1: 379-393
8.	1	Recapitulation and Important Questions Discussions	
Total no.of Hours planned for Unit – II : 8 Hrs			

UNIT-3

S.NO	Lecture Duration (Hr)	Topics to be Covered	Support Materials
1.	1	Network Security Practices Authentication Applications	T1: 399-400
2.	1	Kerberos	T1:401-419
3.	1	X.509 Authentication Services and Encryption Techniques	T1:420-434
4.	1	E-mail security	T1:436-438
5.	1	PGP	T1: 439-457
6.	1	S/MIME	T1: 458-474
7.	1	IP security	T1: 484-516 W5
8.	1	Recapitulation and Important Questions Discussions	
Total no. of Hours planned for Unit – III : 8 Hrs			

UNIT-4

S.NO	Lecture Duration (Hr)	Topics to be Covered	Support Materials
1.	**1**	Web Security	T1:522—528
2.	**1**	Secure Socket Layer	T1:531-549
3.	**1**	Secure Electronic Transaction	T1:549-560
4.	**1**	System Security	T1:563-564
5.	**1**	Intruders and Viruses	W6 T1:565-597
6.	**1**	Firewalls	T1: 623-639
7.	**1**	Password Security	T1:582-591
8.	**1**	Recapitulation and Important Questions Discussions	
Total no.of Hours planned for Unit – IV : 8 Hrs			

UNIT-5

S.NO	LectureDuration (Hr)	Topics to be Covered	Support Materials
1.	1	Case Study**:** Implementation of Cryptographic Algorithms. RSA Algorithm	T1:301-315 T1:26-27
2.	1	DSA, ECC(C/JAVA Programming)	T1:427-430
3.		Network Forensic	W7
4.	1	Security Audit	W7
5.	1	Other Security Mechanisms Introduction to Stenography	T1:468-508
6.	1	Quantum Cryptography.	W8
7.		Water Marking	W9
8.	1	DNA Cryptography.	W9
9.	1	Recapitulation and Important Questions Discussions	
10.	**1**	**Discussion of Previous ESE Question Papers**	
11.	**1**	**Discussion of Previous ESE Question Papers**	
12.	**1**	**Discussion of Previous ESE Question Papers**	
Total no.of Hours planned for Unit – V & Previous ESE QP Discussion : 12 Hrs			

SUGGESTED READINGS

1. William Stallings (2013), "Cryptography and Network Security", 6th Edition. Pearson Education, New Delhi.
2. Bruce Schneir (2006), "Applied Crptography", 2nd Edition. CRC Press, New Delhi.
3. A.Menezes, P.Van Oorschot and S.Vanstone(2010),"Hand Book of Applied Cryptography", 2nd Edition. CRC Press, NewDelhi.
4. Ankit Fadia.(2010), Network Security, 2nd Edition. McMillan India Ltd, New Delhi.

WEBSITES

1. **www.comptechdoc.org**
2. **www.webopedia.com/Team/Symmetric-key.cryptography**
3. **www.Courses.cs.vt.edu/rsa**
4. **www.cs.columbia.edu/lecturers/macs.pdf**
5. **www.slideshare.net/lecture of IP Security**
6. **www.researcher.watson.ibm.com**
7. **www.forensicscontest.com**
8. **www.cse.unr.edu/../watermarking.pptx**
9. **www.cs.duke.edu/../QuantCryptOverview**

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

COURSE CODE: 17CAP404N UNIT: I BATCH-2017-2020

SYLLABUS

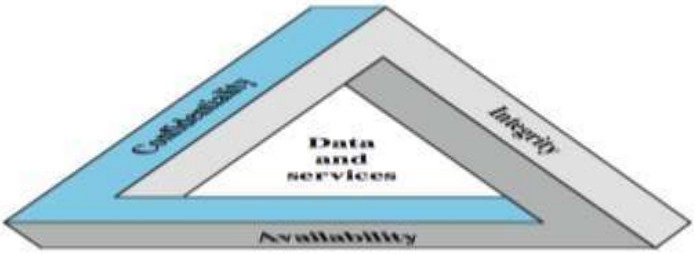
Introduction to Cryptography – Security Attacks – Security Services – Security Algorithm – Stream cipher and Block cipher – Symmetric and Asymmetric – Key Cryptosystem; Symmetric Key Algorithms: Introduction – DES – Triple DES – AES – IDEA – Blowfish – RC5.

INTRODUCTION TO CRYPTOGRAPHY:

Computer Security:

- The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability and confidentiality of information system resources (includes hardware, software, firmware, information/data, and telecommunications)

Key Security Concepts:



Three Key Objectives:

1. **Confidentiality**

Data confidentiality
Privacy

2. **Integrity**
   Data integrity
   System integrity
3. **Availability**
4. **Additional concepts**
   - **Authenticity**
   - **Accountability**

Computer Security Challenges

1. **not simple**
2. **must consider potential attacks**
3. **procedures used counter-intuitive**
4. **involve algorithms and secret info**
5. **must decide where to deploy mechanisms**
6. **battle of wits between attacker / admin**
7. **not perceived on benefit until fails**
8. **requires regular monitoring**
9. **too often an after-thought**
10. **regarded as impediment to using system**

OSI Security Architecture

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

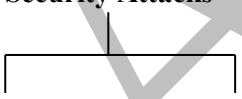
COURSE CODE: 17CAP404N UNIT: I BATCH-2017-2020

ITU-T X.800 "Security Architecture for OSI"defines a systematic way of defining and providing security requirements for us it provides a useful, if abstract, overview of concepts we will study

Aspects of Security

- 3 aspects of information security:
  - security attack
  - security mechanism: detect, prevent, recover
  - security service
- terms
  - threat – a potential for violation of security
  - attack – an assault on system security, a deliberate attempt to evade security services

Security Attacks:



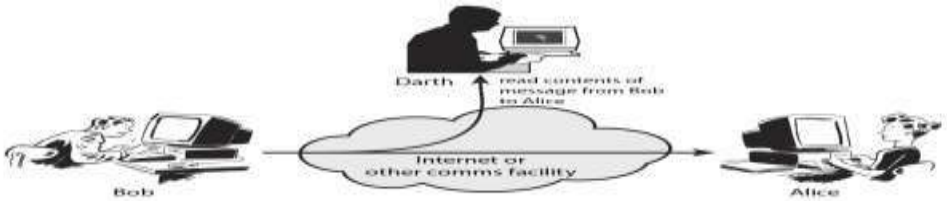
Passive Attacks **Active Attacks**

Passive Attacks

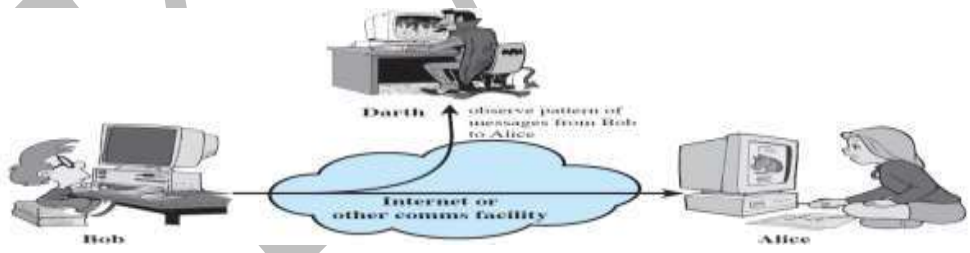
- Passive attacks do not affect system resources
  - Eavesdropping, monitoring
- Two types of passive attacks
  - Release of message contents
  - Traffic analysis

- Passive attacks are very difficult to detect
  - Message transmission apparently normal
    - No alteration of the data
  - Emphasis on prevention rather than detection
    - By means of encryption

- **Passive Attacks (1)**
  Release of Message Contents



- **Passive Attacks (2)**
  Traffic Analysis

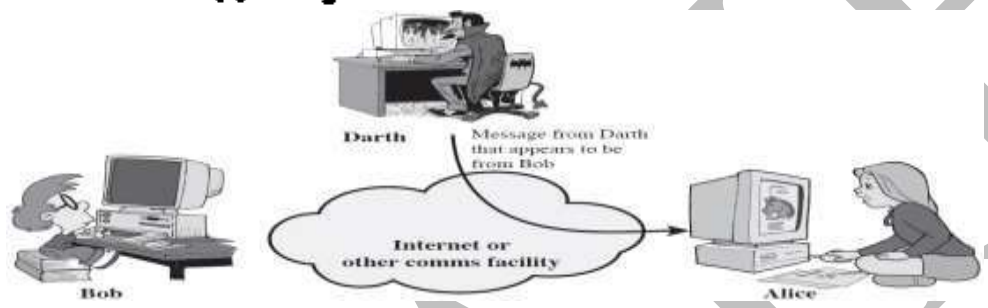


Active attacks

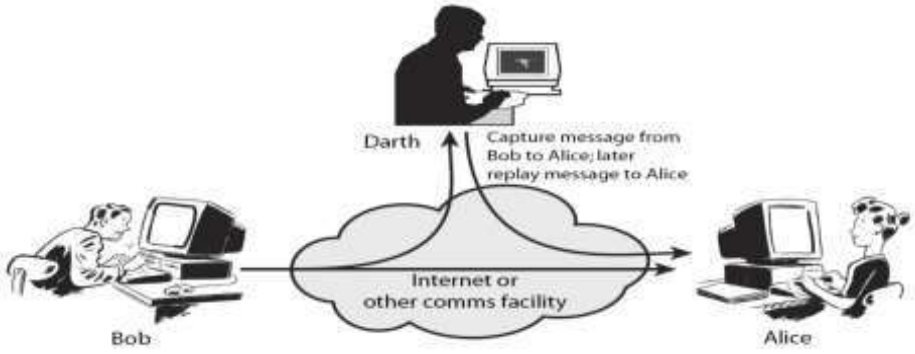
- Active attacks try to alter system resources or affect their operation
  - Modification of data, or creation of false data

- Four categories
  - Masquerade
  - Replay
  - Modification of messages
  - Denial of service: preventing normal use
    - A specific target or entire network
- Difficult to prevent
  - The goal is to detect and recover

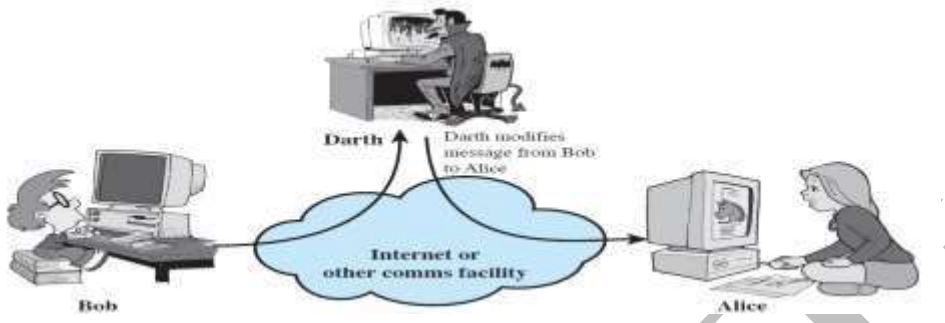
Active Attacks (1) Masquerade



Active Attacks (2) Replay



Active Attacks (3): Modification of Messages



Active Attacks (4): Denial of Service

Security Services:

- enhance security of data processing systems and information transfers of an organization intended to counter security attacks using one or more security mechanisms often replicates functions normally associated with physical documents **For example:** have signatures, dates; need protection from disclosure, tampering, or destruction; be notarized or witnessed; be recorded or licensed.

- X.800:

  "A service provided by a protocol layer of communicating open systems, which ensures adequate security of the systems or of data transfers"

- RFC 2828:

  "A processing or communication service provided by a system to give a specific kind of protection to system resources"

Security Services (X.800):

- **Authentication** - Assurance that communicating entity is the one claimed have both peer-entity & data origin authentication.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; e.g., 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.

- **Access Control:**

In the context of network security, access control is the ability to limit and control the access to host systems and applications via communications links. To achieve this, each entity trying to gain access must first be identified, or authenticated, so that access rights can be tailored to the individual.

- **Data Confidentiality:**

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. For example, when a TCP connection is set up between two systems, this broad protection prevents the release of any user data transmitted over the TCP connection. Narrower forms of this service can also be defined, including the protection of a single message or even specific fields within a message. These refinements are less useful than the broad approach and may even be more complex and expensive to implement. 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**: Guarding against improper information modification or destruction, including ensuring information Nonrepudiation and authenticity. A loss of integrity is the unauthorized modification or destruction of information.
- **Non-Repudiation**

Nonrepudiation prevents either sender or receiver from denying a transmitted message. Thus, when a message is sent, the receiver can prove that the

alleged sender in fact sent the message. Similarly, when a message is received, the sender can prove that the alleged receiver in fact received the message.

- **Availability**: Ensuring timely and reliable access to and use of information.A loss of availability is the disruption of access to or use of information or an information system.

Security Mechanism:

Feature designed to detect, prevent, or recover from a security attack no single mechanism that will support all services required however one particular element underlies many of the security mechanisms in use: cryptographic techniques

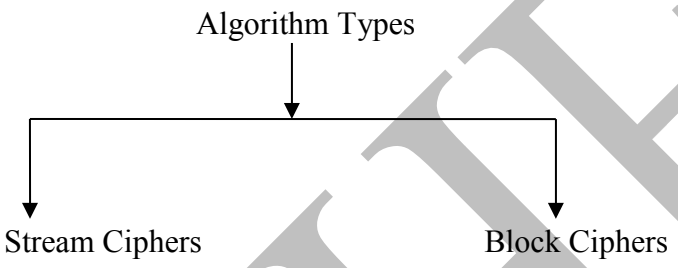
Security Mechanisms (X.800):

- specific security mechanisms:
  - Encipherment, digital signatures, access controls, data integrity, authentication exchange, traffic padding, routing control, notarization
- pervasive security mechanisms:
  - Trusted functionality, security labels, event detection, security audit trails, security recovery

Security algorithms:

Algorithm Types:

We have been talking about the transformation of plain text message into cipher text messages. Regardless of the techniques used, at a broad level, the generation of cipher text from plain text itself can be done in two basic ways, stream ciphers and block ciphers. This is shown in F



Stream Ciphers:

In Stream Ciphers, the plain text is encrypted one bit at a time. Suppose the original message (plain text) is pay 100 in ASCII (i.e. text format). When we convert these ASCII characters to their binary values, let us assume that it translate to 01011100(hypothetically, just for simplicity, in reality, the binary text would be much larger as each text character takes seven bits). Suppose the key to be applied is 10010101 in binary. Let us also assume that we apply the XOR logic as the encryption algorithm. XOR is quite simple to understand. As shown in fig. 2.2 in simple terms, XOR produces an output of 1 only if one input is 0 and the other is 1. The output is 0 if both the inputs are 0 or if both the inputs are 1 (hence the name exclusive).

Input 1	Input 2	Input 3
0	0	0
0	1	1

1	**0**	**1**
1	**1**	**0**

Functioning of XOR logic

As a result of applying one bit of key for every respective bit of the original message, the cipher text is generated as 11001001 in binary (ZTU91 ^% in text). Note that each bit of the plain text is encrypted one after the other. Thus, what is transmitted is 11001001 in binary, which even when translated back to ASCII would mean ZTU91 ^%. This makes no sense to an attacker, and thus protects the information.

Another interesting property of XOR is that when used twice, it produces the original data. For example, suppose we have two binary numbers A=101 and B=110. We now want to perform an XOR operation on A and B to produce a third number C, i.e.:

C = A XOR B

Thus, we will have: C = 101 XOR 110= 011

Now, if we perform C XOR A, we will get B.

That is: B = 011 XOR 101=110

Similarly, if we perform C XOR B, we will get A. That is:

A = 011 XOR 110 = 101

This reversibility of XOR operations has many implications in cryptographic algorithms, as we shall notice. XOR irreversible –when used twice, it produces the original values. This is useful

in cryptography.

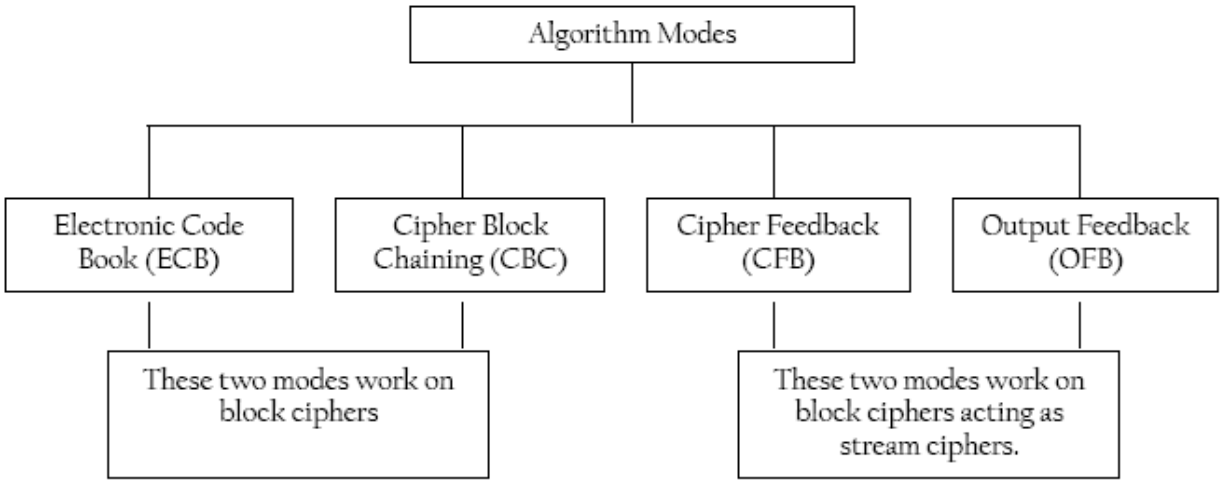
Block Ciphers:

In Block Ciphers, rather than encrypting one bit at a time, a block of bits is encrypted at one go. Suppose we have a plain text FOUR_AND_FOUR that needs to be encrypted. Using block cipher, FOUR could be encrypted first, followed by _AND_ and finally FOUR. Thus, one block of characters gets encrypted at a time. During decryption, each block would be translated back to the original form.

In actual practice, the communication takes place only in bits. Therefore, FOUR actually means binary equivalent of the ASCII characters FOUR. After any algorithm encrypts these, he resultant bits are converted back into their ASCII equivalents. Therefore, we get funny symbols such as Vfa%, etc.

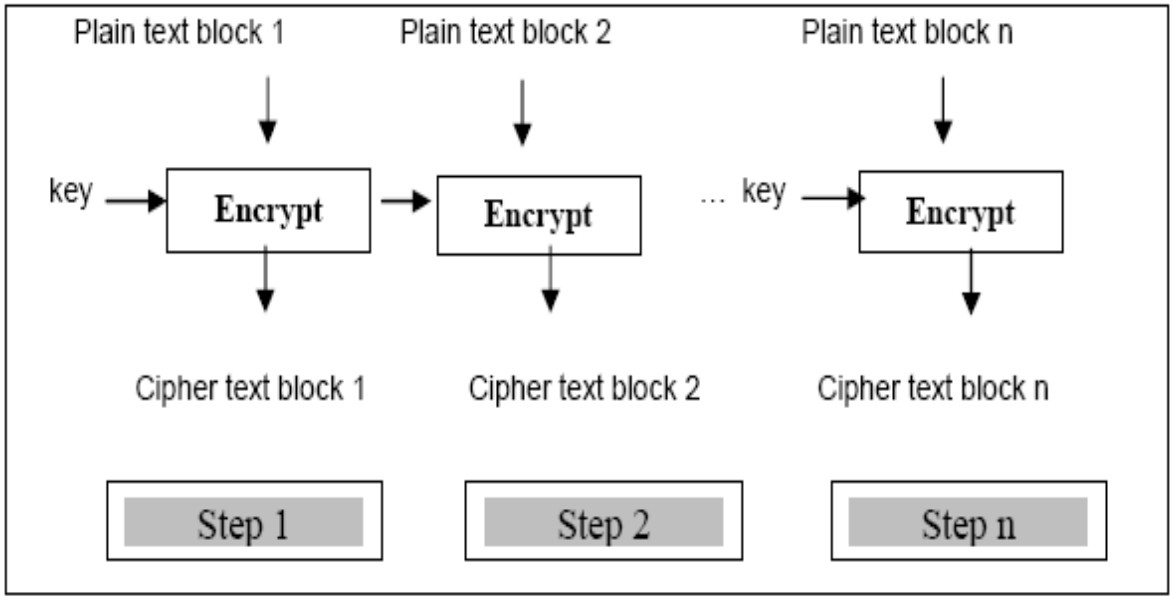
In actual practice, their binary equivalents are received, which are decrypted back into binary equivalent of ASCII FOUR.

Algorithm Modes:

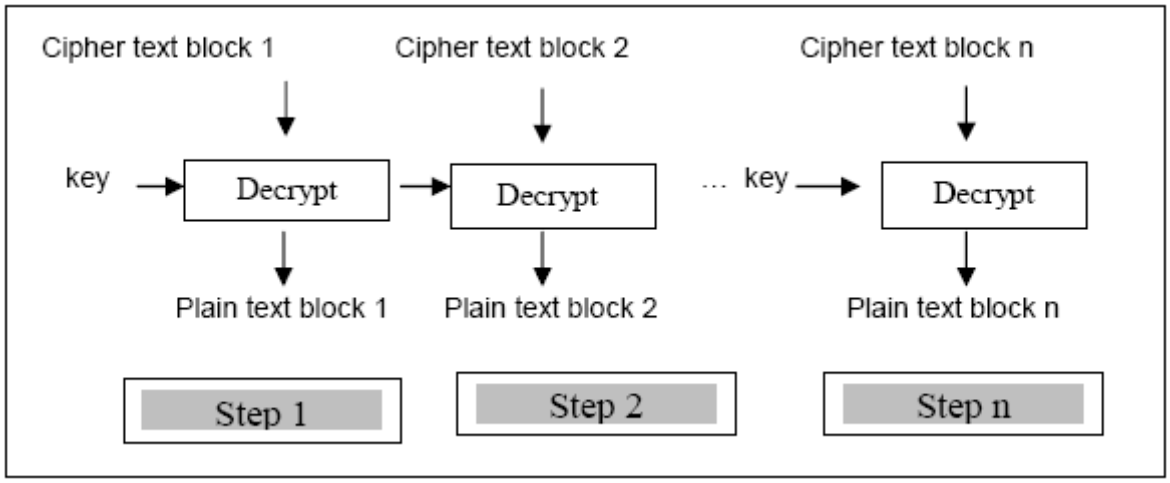


Electronic Code Book (ECB) mode:

Electronic Code Book (ECB) is the simplest mode of operation. Here the meaning plain text message is divided into blocks of 64 bits each. Each such block is encrypted independently of the other blocks. For all blocks in a message, the same key is used for encryption. This process is shown in fig



At the receiver's end, the incoming data is divided into 64-bit blocks, and by using the same key as was used for encryption, each block is decrypted to produce the corresponding plain text block. This process is shown in fig



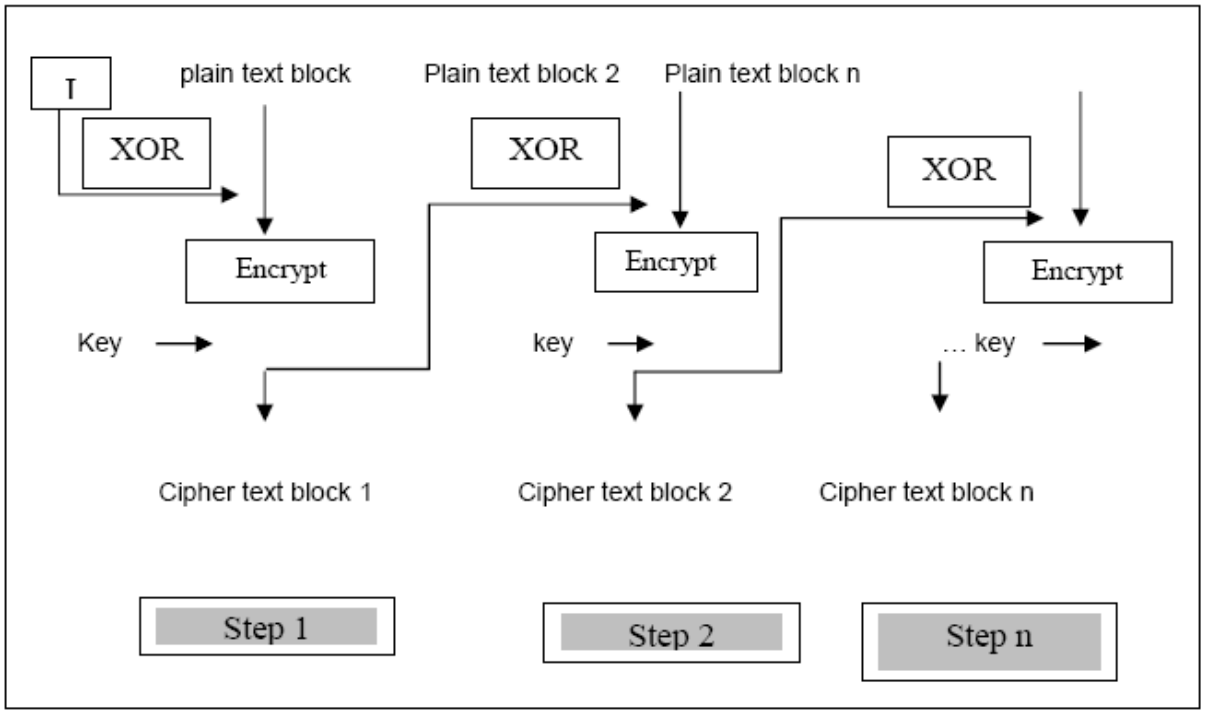
In ECB, since a single key is used for encrypting all the blocks of a message, if a plain text block repeats in the original message the corresponding cipher text block will also repeat in the encrypted message. Therefore, ECB is suitable only for encrypting small messages, where the scope for repeating the same plain text blocks is quite less.

2. Cipher Block Chaining (CBC) mode:

We saw that in the case of ECB, within a given message (i.e. for a given key), a plain text block always produces the same cipher mtext block. Thus, if a block of plain text occurs more than once in the input, the corresponding cipher text block will also occur more than once in the output, thus providing some clues to a cryptanalyst. To overcome this problem, Cipher Block Chaining (CBC) mode ensures that even if a block of plain text repeats in the input, these two (or more) identical plain text blocks yield totally different cipher text blocks in the output. For this, a feedback mechanism is used, as we shall learn now. Chaining adds a feedback mechanism to a block cipher. In Cipher Block Chaining (CBC), the results of the encryption of the previous block are fed back into the encryption of the current block. That is, each block is used to modify the encryption of the next block. Thus each

block of cipher text is dependent on the corresponding current input plain text block, as well as all the previous plain text blocks.

The encryption process of CBC is depicted in fig



1. As shown in the figure, the first step receives two inputs: the first block of plain text and a random block of text, called as Initialization Vector (IV).

a. The IV has no special meaning: it is simply used to make each message unique. Since the value of IV is randomly generated, the likelihood of it repeating in two different messages is quite rare. Consequently, IV helps in making the cipher text somewhat unique or at least quite different from all the other cipher texts in a different message. Interestingly, it is not mandatory to keep IV secret –it can beknown to everybody. This seems slightly concerning and confusing. However, if we

relook at the operation of CBC, we will realize that IV is simply one of the two inputs to the first encryption step. The output of step 1 is cipher text block 1, which is also one of the two inputs to the second encryptionstep.

In other words, Cipher text block 1 is also an IV for step 2! Similarly, Cipher text block 2 is also an IV for step 3, and so on. Since all these cipher text blocks will be sent to the receiver, we are actually anyway sending all IVs for step 2 onwards! Thus, there is no special reason why the IV for step 1 should be kept secret. The key used for encryption is what needs to be kept secret. However, in practice, for maximum security, both the key and the IV are kept secret.

b. The first block of cipher text and IV are combined using XOR and then encrypted using a key to produce the first cipher text block. The first cipher text block is then provided as a feedback to the next plain text block, as explained below.

2. In the second step, the second plain text block is XORed with the output of step 1, i.e. the first cipher text block. It is then encrypted with the same key, as used in step 1. This vproduces cipher text block 2.

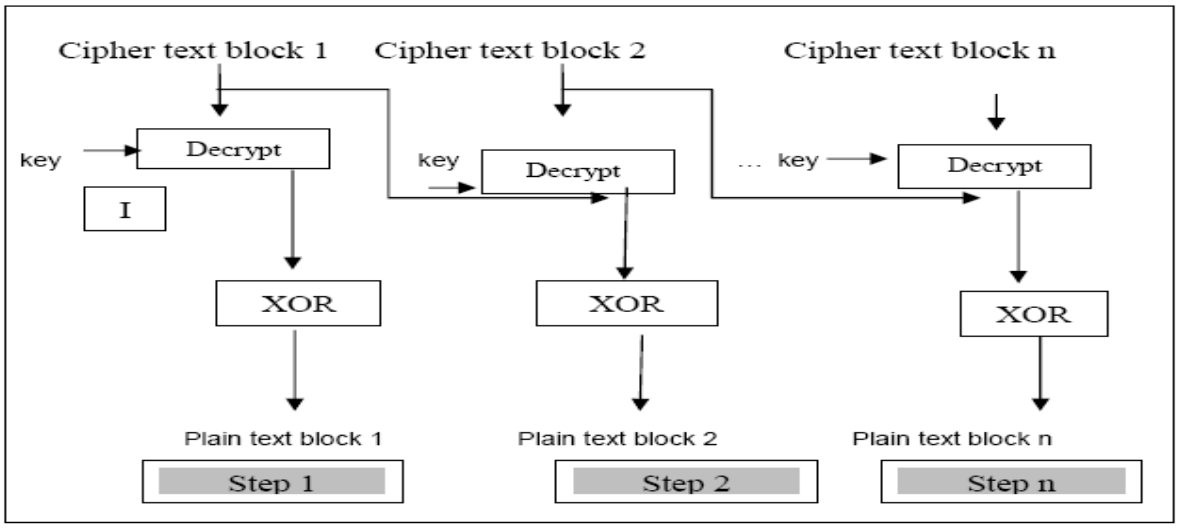
3. In the third step, the third plain text block is XORed with the output of step 2, i.e. the second cipher text block. It is then encrypted with the same key, as used in step 1.

4. This process continues for all the remaining plain text blocks of the original message. Remember that the IV is used only in the first plain text block. However, the same key is use for encryption of all plain text blocks.

1. The Cipher text block 1 is passed through the decryption algorithm using the same key, which was used during the encryption process for all the plain text blocks. The output of this step is then XORed with the IV. This process yields Plain text block 1.

2. In step 2, the Cipher text block 2 is decrypted, and its output is XORed with Cipher text block 1, which yields Plain text block 2.

3. This process continues for all the Cipher text blocks in the encrypted message.

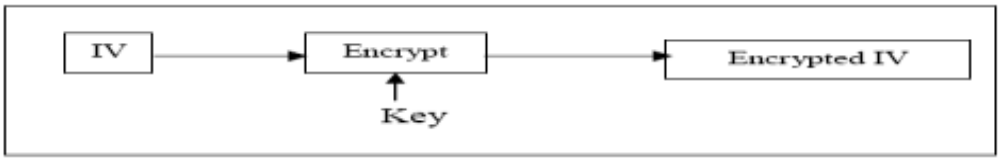


3. Cipher Feedback (CFB) mode:

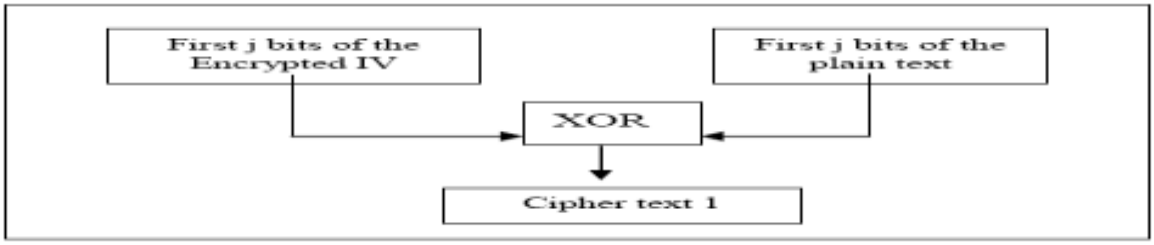
Not all applications can work with blocks of data. Security is also Required in applications that are character-oriented. For instance, an operator can be typing keystrokes at a terminal, which need to be immediately transmitted across the communications link in a secure manner, i.e. by using encryption. In such situations, stream cipher must be used. The Cipher Feedback (CFB) mode is useful in such cases. In this mode, data is encrypted in units that are smaller (e.g. they could be of size 8 bits, **i.e.** the size of character typed by an operator) than a defined block size (which is usually 64 bits).
Let us understand how CFB mode works, assuming that we are dealing with j bits at a time (as we have seen, usually, but not always, j=8). Since CFB is slightly more complicated as compared to the first two cryptography modes, we shall study CFB in a **stepby- step fashion.**

Step 1: Like CBC, a 64-bit Initialization vector (IV) is used in thecase of CFB mode. The IV is kept in a shift register. It is encrypted in the first step to produce a corresponding 64-bit IV cipher text.

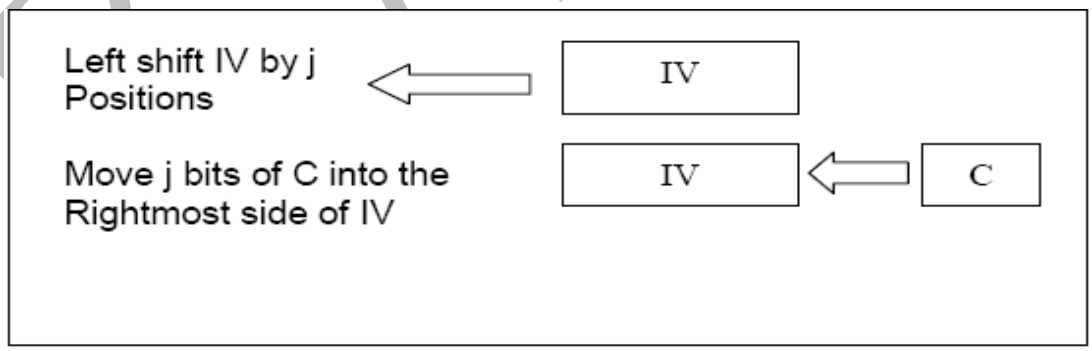
This is **shown fig.**



Step 2: Now, the leftmost (i.e. the most significant) j bits of the encrypted IV are XORed with the first j bits of the plain text. This produces the first portion of cipher text (say C) as shown in fig. transmitted to the receiver.



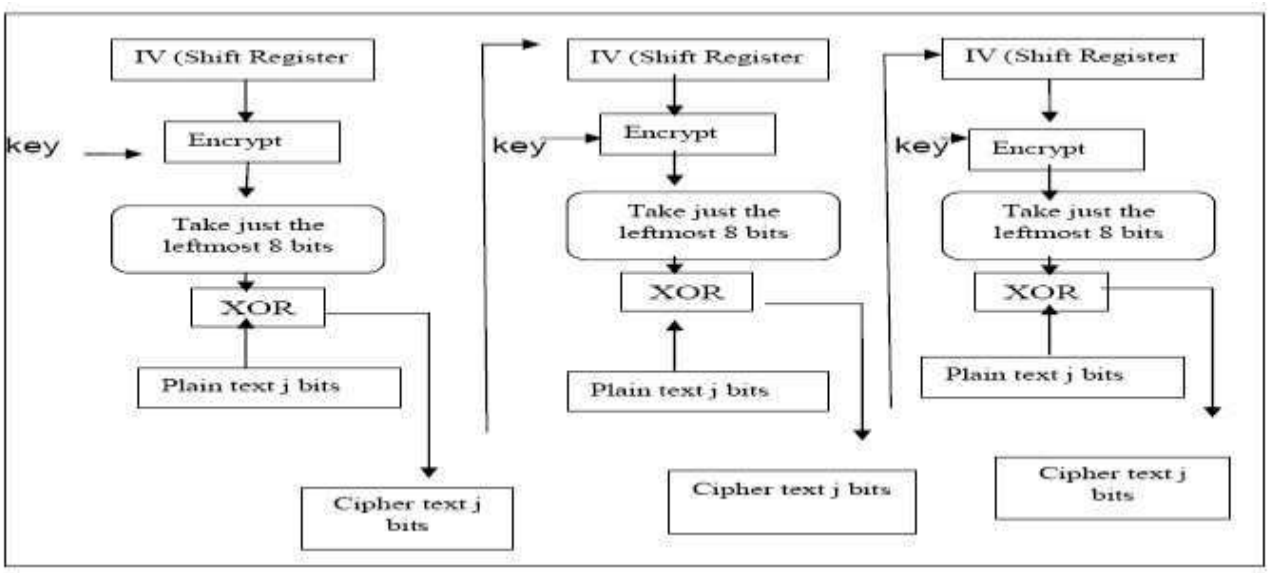
Step 3: Now, the bits of IV (i.e. the contents of the shift register containing IV) are shifted left by j positions. Thus the rightmost j positions of the shift register now contain unpredictable data. These rightmost j positions are now filled with C. This is shown in fig



Step 4: Now step 1 through 3 continue until all the plain text units are encrypted. That is, the following steps repeat:

- IV is encrypted.
- The leftmost j bits resulting from this encryption process are XORed with the next j bits of the plain text.
- The resulting cipher text portion (i.e. comprising of the next j bits of the cipher text) is sent to the receiver.
- The shift register containing the IV is left-shifted containing the IV.
- Fig shows the overall conceptual view of the CFB mode.
- At the receiver's end, the decryption process is pretty similar, with minor changes.

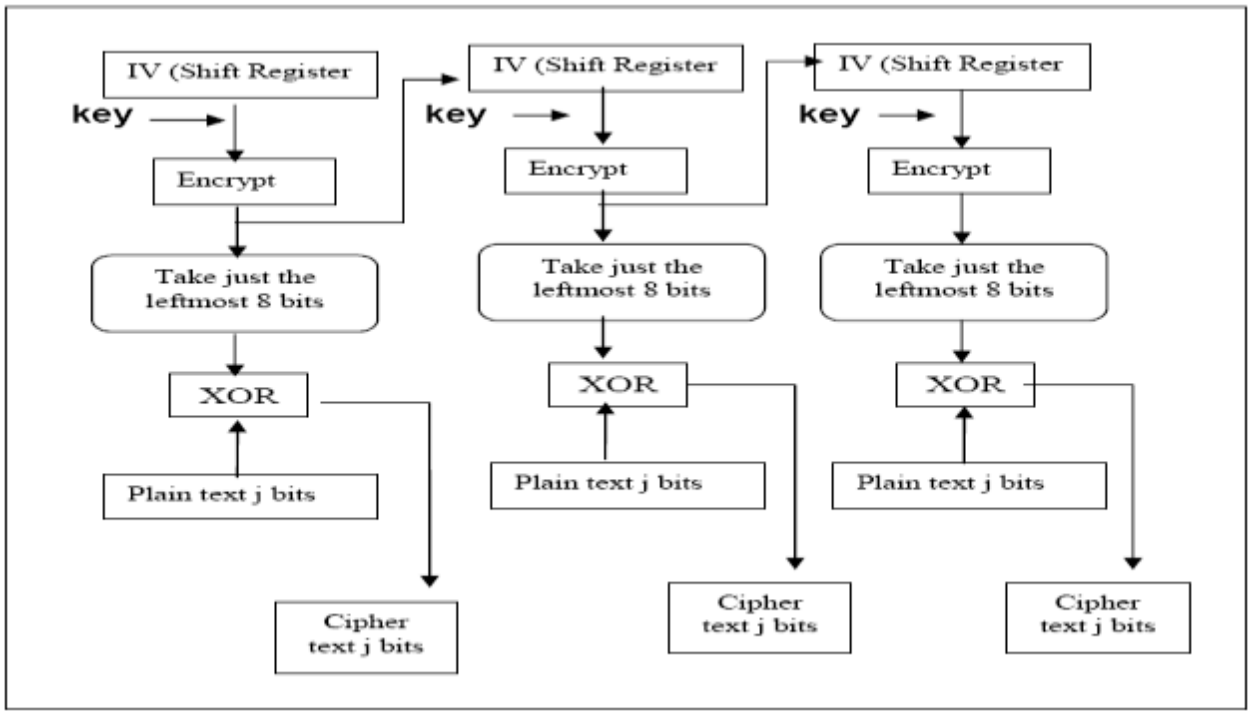
CONCEPTUAL VIEW OF THE CFB



4. Output Feedback (OFB) mode:

The Output Feedback (OFB) mode is extremely similar to the CFB. The only difference is that in case of CFB, the cipher text is fed into the next stage of encryption process. But in the case of OFB, the output of the IV encryption process is fed into the next stage of encryption process. Simply draw the block diagram of the OFB process, as shown in fig

BLOCK DIAGRAM OF THE OFB PROCESS:



CRYPTOSYSTEM

The combination of algorithm key and key management functions used to perform cryptographic operations. Encryption is a key concept in cryptography.

- **Symmetric Encryption**

Symmetric Encryption uses a single secret key that needs to be shared among the people who needs to receive the message while Asymmetric encryption uses a pair of public key, and a private key to encrypt and decrypt messages when communicating. Symmetric Encryption is an age old technique while Asymmetric Encryption is relatively new.

- **Asymmetric Encryption**

  Mathematically related key pairs for encryption and decryption.Public and private keys.Asymmetric Encryption was introduced to complement the inherent problem of the need to share the key in symmetric encryption model eliminating the need to share the key by using a pair of public-private keys.

KEY CRYPTOSYSTEM

SYMMETRIC KEY CRYPTOSYSTEM:

Symmetric encryption, also referred to as conventional encryption or single-key encryption, was the only type of encryption in use prior to the development of publickey encryption in the 1970s. It remains by far the most widely used of the two types of encryption. Part One examines a number of symmetric ciphers. In this chapter, we begin with a look at a general model for the symmetric encryption process; this will enable us to understand the context within which the algorithms are used. Next, we examine a variety of algorithms in use before the computer era. Finally, we look briefly at a different approach known as steganography.

SYMMETRIC CIPHER MODEL:

A symmetric encryption scheme has five ingredients

• **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. The exact substitutions and transformations performed by the algorithm depend on the key.

• **Cipher text:**

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 cipher texts. The cipher text is an apparently random Stream of data and, as it stands, is unintelligible.

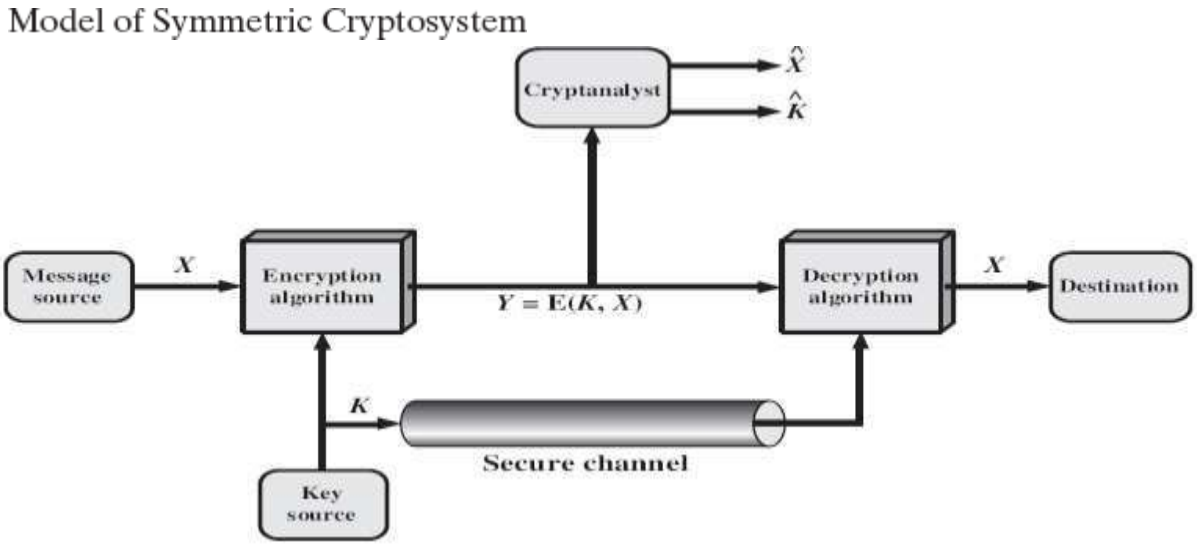
• **Decryption algorithm:**

This is essentially the encryption algorithm run in reverse. It takes the cipher text and the secret key and produces the original plaintext. There are two requirements for secure use of conventional encryption:

1. We need a strong encryption algorithm. At a minimum, we would like the algorithm to be such that an opponent who knows the algorithm and has access to one or more cipher texts would be unable to decipher the cipher text or figure out the key.

2. Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure. If someone can discover the key and +knows the algorithm, all communication using this key is readable.

We assume that it is impractical to decrypt a message on the basis of the ciphertext *plus* knowledge of the encryption/decryption algorithm. In other words, we do not need to keep the algorithm secret; we need to keep only the key secret. This feature of symmetric encryption is what makes it feasible for widespread use. The fact that the algorithm need not be kept secret means that manufacturers can and have developed low-cost chip implementations of data encryption algorithms. These chips are widely available and incorporated into a number of products. With the use of symmetric encryption, the principal security problem is maintaining the secrecy of the key.



BLOCK CIPHER PRINCIPLES

Many symmetric block encryption algorithms in current use are based on a structure referred to as a Feistel block cipher [FEIS73]. For that reason, it is important to examine the design principles of the Feistel cipher. We begin with a comparison of stream ciphers and block ciphers. Then we discuss the motivation for the Feistel block cipher structure. Finally, we discuss some of its implications.

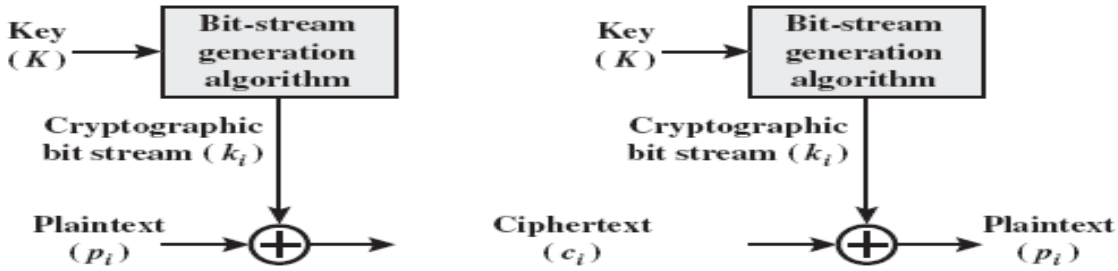
Stream Ciphers and Block Ciphers

A **stream cipher** is one that encrypts a digital data stream one bit or one byte at a time. Examples of classical stream ciphers are the auto keyed Vigenère cipher and theVernam cipher. In the ideal case, a one-time pad version of the Vernam cipher wouldbe used in which the key stream is as long as the plaintext bit stream. If the cryptographic key stream is random, then this cipher is

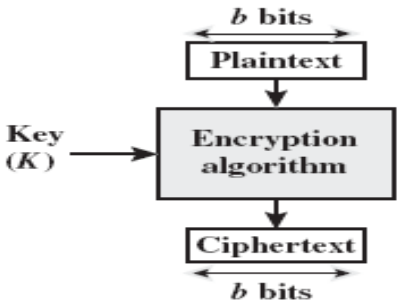
unbreakable by any means other than acquiring the key stream. However, the key stream must be provided to both users in advance via some independent and secure channel.

A block cipher is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. Typically, a block size of 64 or 128 bits is used. As with a stream cipher, the two users share a symmetric encryption key.

Stream Cipher and Block Cipher



(a) Stream cipher using algorithmic bit-stream generator



(b) Block cipher

Data Encryption Standard:

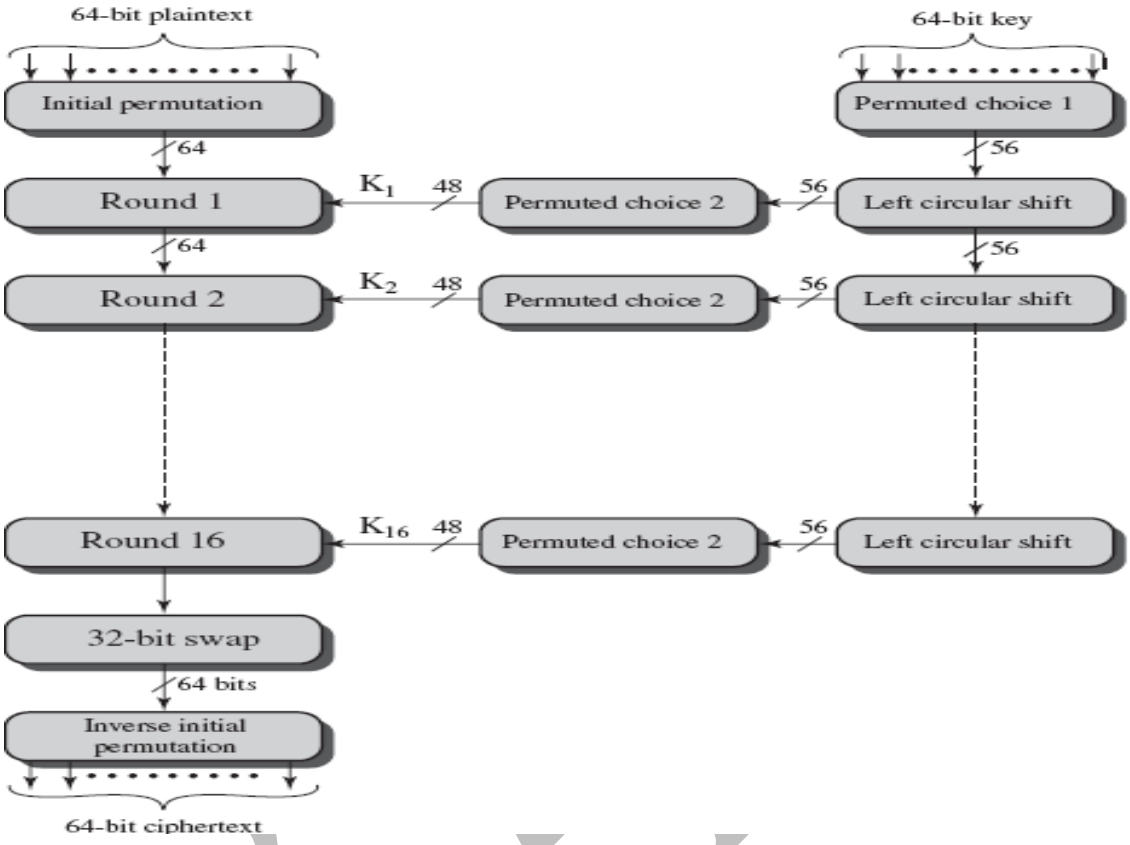
The most widely used encryption scheme is based on the Data Encryption Standard (DES) adopted in 1977 by the National Bureau of Standards, now the National Institute of Standards and Technology

(NIST), as Federal Information Processing Standard 46 (FIPS PUB 46). The algorithm itself is referred to as the Data Encryption Algorithm (DEA).7 For DES, data are encrypted in 64-bit blocks using a 56bit key. The algorithm transforms 64-bit input in a series of steps into a 64-bitoutput.The same steps, with the same key, are used to reverse the encryption. The DES enjoys widespread use. It has also been the subject of much controversy concerning how secure the DES is.

DES Encryption

The overall scheme for DES encryption is illustrated in Figure As with anyEncryption scheme, there are two inputs to the encryption function: the plaintext tobe encrypted and the key. In this case, the plaintext must be 64 bits in length and thekey is 56 bits in length.

General Depiction of DES Encryption Algorithm



A DES EXAMPLE:

We now work through an example and consider some of its implications. Although you are not expected to duplicate the example by hand, you will find it informative to study the hex patterns that occur from one step to the next. For this example, the plaintext is a hexadecimal palindrome. The plaintext, key, and resulting cipher text are as follows:

Plaintext: **02468aceeca86420**

Key: **0f1571c947d9e859**

Ciphertext: **da02ce3a89ecac3b**

THE STRENGTH OF DES:

With a key length of 56 bits, there are possible keys, which is an approximately key. Thus, on the face of it, a brute-force attack appears impractical. Assuming that, on average, half the key space has to be searched; a single machine performing one DES encryption per microsecond would take more than a thousand years to break the cipher.

DES finally and definitively proved insecure in July 1998, when the Electronic Frontier Foundation (EFF) announced that it had broken a DES encryption using a special-purpose "DES cracker" machine that was built for less than $250,000.The attack took less than three days.The EFF has published a detailed description of the machine, enabling others to build their own cracker [EFF98].And, of course, hardware prices will continue to drop as speeds increase, making DES virtually worthless.It is important to note that there is more to a key-search attack than simply running through all possible keys. Unless known plaintext is provided, the analyst must be able to recognize plaintext as plaintext. If the message is just plain text in English, then the result pops out easily, although the task of recognizing English would have to be automated. If the text message has been compressed beforeencryption, then recognition is more difficult. And if the message is some more generaltype ofdata, such as a numerical file, and this has been compressed, the problembecomes even more difficult to automate. Thus, to supplement the brute-forceapproach, some degree of knowledge about the expected plaintext is needed, andsome means of automatically distinguishing plaintext from garble is also needed.The EFF approach addresses this issue as well and introduces some automated techniques that would be effective in many contexts.

THE NATURE OF THE DES ALGORITHM:

Another concern is the possibility that cryptanalysis is possible by exploiting the characteristics of the DES algorithm. The focus of concern has been on the eight substitution tables, or S-boxes, that are used in each iteration. Because the design criteria for these boxes, and indeed for the entire algorithm, were not made public, there is a suspicion that the boxes were constructed in such a way that cryptanalysis is possible for an opponent who knows the weaknesses in the S-boxes.This assertion is tantalizing, and over the years a number of regularities and unexpected behaviors of the S-boxes have been discovered. Despite this, no one has so far succeeded indiscovering the supposed fatal weaknesses in the S-boxes.9

TIMING ATTACKS:

Timing attacks in more detail in Part Two, as they relate to public-key algorithms. However, the issue may also be relevant for symmetric ciphers. In essence, a timing attack is one in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts.

A timing attack exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs. [HEVI99] reports on an approach that yields the Hamming weight (number of bits equal to one) of the secret key. This is a long way from knowing the actual key, but it is an intriguing first step. The authors conclude that DES appears to be fairly resistant to a successful timing attack but suggest some avenues to explore. Although this is an interesting line of attack, it so far appears unlikely that this technique will ever be successful against DES or more powerful symmetric ciphers such as triple DES and AES.

TRIPLE DES

Triple-DES with two keys is a popular alternative to single-DES, but suffers from being 3 times slower to run.Although there are no practical attacks, have some indications of attack approaches.Hence some are now adopting Triple-DES with three keys for greater security.

- Clear a replacement for DES was needed
  - Theoretical attacks that can break it
  - Demonstrated exhaustive key search attacks
- AES is a new cipher alternative
- Prior to this alternative was to use multiple encryption with DES implementations
- Triple-DES is the chosen form

Why Triple-DES?

- Why not Double-DES?
  - NOT same as some other single-DES use, but have
- Meet-in-the-middle attack
  - Works whenever use a cipher twice
  - Since $X = E_{K1}[P] = D_{K2}[C]$
  - Attack by encrypting P with all keys and store
  - Then decrypt C with keys and match X value
  - Can show takes $O(2^{56})$ steps

Triple-DES with Two-Keys

- Hence must use 3 encryptions
  - Would seem to need 3 distinct keys
- But can use 2 keys with e-d-e sequence
  - $C = e_{k1}[d_{k2}[e_{k1}[p]]]$

- Nb encrypt & decrypt equivalent in security
- If k1=k2 then can work with single des

- Standardized in ansi x9.17 & iso8732
- No current known practical attacks.

Triple-DES with Three-Keys

- Although are no practical attacks on two-key Triple-DES have some indications
- Can use Triple-DES with Three-Keys to avoid even these
  - $C = E_{K3}[D_{K2}[E_{K1}[P]]]$
- Has been adopted by some Internet applications, eg PGP, S/MIME.

Triple DES Modes

Triple ECB (Electronic Code Book)

- This variant of Triple DES works exactly the same way as the ECB mode of DES.
- This is the most commonly used mode of operation.

Triple CBC (Cipher Block Chaining)

- This method is very similar to the standard DES CBC mode.
- As with Triple ECB, the effective key length is 168 bits and keys are used in the same manner, as described above, but the chaining features of CBC mode are also employed.
- The first 64-bit key acts as the Initialization Vector to DES.
- Triple ECB is then executed for a single 64-bit block of plaintext.

- The resulting ciphertext is then XORed with the next plaintext block to be encrypted, and the procedure is repeated.

- This method adds an extra layer of security to Triple DES and is therefore more secure than Triple ECB, although it is not used as widely as Triple ECB.

AES

AES Structure:

• data block of 4 columns of 4 bytes is state

• key is expanded to array of words

• has 9/11/13 rounds in which state undergoes: – byte substitution (1 S-box used on every byte) – shift rows (permute bytes between groups/columns) – mix columns (subs using matrix multiply of groups) – add round key (XOR state with key material) – view as alternating XOR key & scramble data bytes

• initial XOR key material & incomplete last round

• with fast XOR & table lookup implementation

Comments on AES

1. an iterative rather than Feistel cipher

2. key expanded into array of 32-bit words– four words form round key in each round

3. 4 different stages are used as shown

4. has a simple structure

5. only AddRoundKey uses key

6. AddRoundKey a form of Vernam cipher

7. each stage is easily reversible

8. decryption uses keys in reverse order

9. decryption does recover plaintext

10.final round has only 3 stages

IDEA

Operations:

XOR, Addition mod 216, multiplication mod 216

+1

Why these special mod for addition, multiplication

They do not satisfy the distributive law

They do not satisfy the associative law

BLOWFISH

- A symmetric block cipher designed by Bruce Schneier in 1993/94
- Characteristics
  - Fast implementation on 32-bit cpus
  - Compact in use of memory
  - Simple structure eases analysis/implementation

- Variable security by varying key sizehas been implemented in various products.

Blowfish Key Schedule

- Uses a 32 to 448 bit key
- Used to generate
  - 18 32-bit sub keys stored in k-array k_j
  - Four 8x32 s-boxes stored in $s_{i,j}$
- Key schedule consists of:
  - Initialize P-array and then 4 S-boxes using pi
  - XOR P-array with key bits (reuse as needed)
  - Loop repeatedly encrypting data using current P & S and replace successive pairs of P then S values
  - Requires 521 encryptions, hence slow in rekeying

Blowfish Encryption

- Uses two primitives: addition & XOR
- Data is divided into two 32-bit halves L_0&R_0

  for*i*= 1 to 16 do

  $R_i = L_{i-1}$ XOR P_i;

  L_i= F[R_i] XOR R_{i-1};

  $L_{17} = R_{16}$ XOR P_{18};

$R_{17} = L_{16}$ XOR i_{17};

- where

F[*a*,*b*,*c*,*d*] = (($S_{1,a}$+ $S_{2,b}$) XOR $S_{3,c}$) + $S_{4,a.}$

RC5

- a proprietary cipher owned by RSADSI
- designed by Ronald Rivest (of RSA fame)
- used in various RSADSI products
- can vary key size / data size / no rounds
- very clean and simple design
- easy implementation on various CPUs
- yet still regarded as secure

RC5 Ciphers

- RC5 is a family of ciphers RC5-w/r/b
  - w = word size in bits (16/32/64) nb data=2w
  - r = number of rounds (0..255)
  - b = number of bytes in key (0..255)
- Nominal version is RC5-32/12/16
  - 32-bit words so encrypts 64-bit data blocks
  - using 12 rounds
  - with 16 bytes (128-bit) secret key.

RC5 Key Expansion

- RC5 uses 2r+2 sub key words (w-bits)
- sub keys are stored in array S[i], i=0..t-1
- then the key schedule consists of
  - initializing S to a fixed pseudorandom value, based on constants e and phi
  - the byte key is copied (little-endian) into a c-word array L
  - a mixing operation then combines L and S to form the final S array.

RC5 Encryption

- Split input into two halves A & B

$L_0 = A + S[0];$

$R_0 = B + S[1];$

for $i = 1$ to r do

$L_i = ((L_{i-1} \text{ XOR } R_{i-1}) <<< R_{i-1}) + S[2 \times i];$

$R_i = ((R_{i-1} \text{ XOR } L_i) <<< L_i) + S[2 \times i + 1];$

- Each round is like 2 DES rounds
- Note rotation is main source of non-linearity
- Need reasonable number of rounds (eg 12-16).

RC5 Modes

- RFC2040 defines 4 modes used by RC5
  - RC5 Block Cipher, is ECB mode

- RC5-CBC, is CBC mode
- RC5-CBC-PAD, is CBC with padding by bytes with value being the number of padding bytes.
- RC5-CTS, a variant of CBC which is the same size as the original message, uses cipher text stealing to keep size same as original.

UNIT I

POSSIBLE QUESTIONS

1. Write short notes on
   (i) Security Attack (ii) Security Services (iii) Security Algorithm
2. Explain briefly about Key Cryptosystem
3. Explain the overview of algorithm modes with neat diagram
4. Explain briefly
   i)DES (ii)AES (iii) Blowfish
5. Explain in detail about stream cipher and block cipher.
6. Write short notes on
   (i) Triple DES (ii) IDEA (iii) RC5.
7. Explain Symmetric key cryptography and the problems of key distribution.

8. Explain in detail about Data Encryption Standard
9. Explain in detail about Passive Attacks and Active Attacks
10. Explainindetailabout Symmetric key cryptographyand Asymmetric key cryptography
11. Perform encryption and decryption using RSA algorithm with p=3, q=11, e=7 and M=5

KARPAGAM ACADEMY OF HIGHER EDUCATION
DEPARTMENT OF COMPUTER APPLICATIONS
II MCA
CRYPTOGRAPHY AND NETWORK SECURITY

UNIT 1

S.NO	Question	Option 1	Option 2	Option 3	Option 4	Answer
1	CMAC stands for	cipher based mask authendication code	cipher based message architecture code	common based message architecture code	cipher based message authendication code	cipher based message authendication code
2	_____ is the study of techniques for ensuring the secrecy and/or authenticity of information	cryptology	network security	computer security	internet	cryptology
3	____ which deals with the defeating such techniques to recover information	computer security	network security	cryptoanalysis	internet	cryptoanalysis
4	_____area covers the use of cryptographic algorithms in network protocol and network application	computer security	network security	internet	mobile security	network security
5	_____ term to refer to the security of computers against intruders.	internet	network security	computer security	mobile security	computer security
6	the generic name for the collection of tools designed to protect data and to thwart hackers is	computer security	network security	internet	mobile security	computer security
7	_____ measures are needed to protect data during their transmission	mobile security	computer security	internet	network security	network security
8	IAB stands for _____	Intel Aithmetic Board	Internet Aithmetic Board	Internet Architecture Board	Intel Architecture Board	Internet Architecture Board
9	CERT stands for_____	computer electric response team	computer emergency response team	computer electric reply team	computer electric reply task	computer emergency response team
10	the ____ security architecture is useful to managers as a way of organizing the task of providing security	OSI	SIO	OSII	OSA	OSI
11	the OSI architecture focuses on security and	internet,base,security	net,mech,none	attack,internet,services	attack,mechanism,service	attack,mechanism,service
12	Any action that compromises the security of information owned by an organization is known as_____	Security attack	Security Service	Security mechanism	None	Security attack
13	_____is a mechanism that is designed to detect,prevent,or recover from a security attack	Security service	Security attack	Security mechanism	None	Security mechanism
14	_____ is a service that enhances the security of the data processing systems and the information transfers of an organization	Security mechanism	Security service	Security attack	None	Security service
15	_____ are in the nature of eavesdropping on,or monitoring of,transmissions	mobile attack	active attacks	internet attacks	passive attacks	passive attacks
16	_____ are very difficult to detect because they do not involve any alteration of the data	active attacks	passive attacks	internet attacks	mobile attack	passive attacks
17	_____involves some modification of the data stream or the creation of a false stream.	active sttacks	passive attacks	internet attacks	mobile attack	active sttacks
18	A _____ takes place when one entity pretends to be a different entity.	Reply	masquerade	denial of service	X.800	masquerade
19	_____ involves the passive capture of a data unit and its subsequent retransmission to produce an unauthorized effect	Reply	masquerade	denial of service	X.801	Reply
20	the _____ prevents or inhibits the normal use or management of communication facilities.	masquerade	denial of service	Reply	X.802	denial of service
21	_____ defines a security service as a service provided by a protocol layer of communicating open systems	X.600	X.500	X.800	X.8000	X.800
22	X.800 divides services into_____categories and specific services	5,14	6,14	5,19	6,19	5,14
23	'-_____requires the access to information reduces may be controlled by or for the target system	authentication	Non repudiation	Access control	confidentiality	Access control
24	_____ requires that neither the sender nor the receiver of the message be able to deny the transmission	Non repudiation	Authentication	Access control	Access control	Non repudiation
25	_____requires that computer system assets be available to authorized parties when needed.	Non repudiation	Authentication	Availability	Access control	Availability
26	_____ is the protection of transmitted dsta from passive attacks	Authentication	confidentiality	Non repudiation	Access control	confidentiality
27	_____ provides for the corroboration of the source of a data unit	data origin authentication	peer entity authentication	transformation	Access control	data origin authentication
28	_____ exploit service flaws in computer to inhibit use by legitimate users	service threats	information threats	internet	network security	service threats
29	_____ threats intercepts or modify data on behalf of users who should not have access to that data	internet	service threats	information access	network security	information access
30	_____ and _____ are two examples of software attacks	viruses and worms	serivce	internet	network	viruses and worms
31	_____ encryption is a form of cryptosystem in which encryption and decryption are performed using the same key	symmetric	asymmetric	service	network	symmetric
32	_____encryption transforms plaintext into ciphertext using a secreat key and an encryption algorithms	service	asymmetric	symmetric	network	symmetric
33	_____ involves trying all possible keys	brute force	symmetric	asymmetric	network	brute force
34	_____ techniques map plaintext elements into ciphertext elements	transposition	substitution	service	network	substitution
35	_____ techniques systematically transpose the positions of plaintext elements	transposition	substitution	service	network	transposition
36	_____machines are sophisticated precomputer hardware devices that use substitution techniques	computer	rotor	network	embedded	rotor
37	_____ is a technique for hiding a secret message within a larger one	steganography	encryption	decryption	none	steganography
38	An original message is known as	ciphertext	input	plaintext	output	plaintext
39	the coded message is called the _____	plaintext	ciphertext	input	output	ciphertext
40	the process of converting from plain text into ciphertext is known as_____	enciphering	deciphering	substitution	transposition	enciphering
41	restoring the plaintext from the ciphertext is_____	deciphering	enciphering	substitution	transposition	deciphering
42	the _____ key is also input to the encryption algorithm	secret	plain	cipher	none	secret
43	_____is essientially the encryption algorithm run in reverse	substitution	encryption	decryption	transposition	decryption
44	the _____ algorithm performs various substition and transformation on the plaintext	transposition	encryption	substitution	decryption	encryption
45	_____is the scrambled message produced as output	plaintext	ciphertext	input	output	ciphertext
46	_____ types of attack explots the characteristics of the algorithm to attempt to deduce a specific plaintext	transposition	substitution	cryptalalysis	none	cryptalalysis
47	A ____ techniques is one in which the letter of plaintext are replaced by other letters or by numbers or symbols	substitution	transposition	cryptalalysis	none	substitution
48	the _____cipher involves replacing each letter of the alphabet with the letter standing three places futher down the alphabet	caesar	monoalphabetic cipher	playfair cipher	hill ciphers	caesar
49	_____,the relative frequency of the letters can be determined and compared to a standard frequency distribution for english	playfair cipher	hill ciphers	monoalphabetic cipher	caesar	monoalphabetic cipher
50	____ treats digrams in the plaintext as single units and translates these units into ciphertext digrams	playfair cipher	monoalphabetic cipher	hill ciphers	caesar	playfair cipher
51	_____ takes m successive palintext letters and substitutes for them m ciphertext letters	playfair cipher	monoalphabetic cipher	caesar	hill ciphers	hill ciphers
52	_____produces random output that bears no stastical relationship to the plaintext	one time pad	vigenere	rail fence	none	one time pad
53	_____ which the plaintext is written down as a sequence of diagnoals and then read off as a sequence of rows	one time pad	rail fence	vigenere	none	rail fence
54	_____selected letters of printed or typewritten text are overwritten in pencil	pin punctures	invisible ink	character marking	type writer correction ribbon	character marking
55	_____a number of substances can be used for writing but leave no visible trace until heat	character marking	invisible ink	pin punctures	type writer correction ribbon	invisible ink
56	small _____ on selected letters are ordinarly not visible unless the paper is held up in front of a light	pin punctures	type writer correction ribbon	character marking	pin punctures	pin punctures
57	_____used between lines typed with a black ribbon	character marking	pin punctures	invisible ink	type writer correction ribbon	type writer correction ribbon
58	the machines consists of a set of independently rotating cylinders through which _____ can flow	electrical pules	waves	rail fence	none	electrical pules
59	_____,in which a keyword is concatenated with the plaintext itself to provide a running key	autokey system	random key system	rail fence	none	autokey system
60	the best known and one of the simplest such algorithm is referred to as the _____ cipher	vigenere	rail fence	one time pad	none	vigenere

CLASS: II MCA **COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY**

COURSE CODE: 17CAP404N **UNIT: II** **BATCH-2017-2020**

SYLLABUS

Public Key Cryptosystem: Introduction to Number Theory – RSA Algorithm – Key Management – Diffie-Hell man key exchange – Introduction to Elliptic Curve Cryptography; Message Authentication and Hash functions – Hash and Mac Algorithm – Digital Signatures and Authentication Protocol.

Public-key cryptography

Public-key cryptography, also known as **asymmetric cryptography**, refers to a cryptographic algorithm which requires two separate keys one of which is *secret* (or *private*) and one of which is *public*. Although different, the two parts of this key pair are mathematically linked. The public key is used to encrypt plaintext or to verify a digital signature; whereas the private key is used to decrypt ciphertext or to create a digital signature. The term "asymmetric" stems from the use of different keys to perform these opposite functions, each the inverse of the other – as contrasted with conventional ("symmetric") cryptography which relies on the same key to perform both.

Public-key algorithms are based on mathematical problems which currently admit no efficient solution that are inherent in certain integer factorization, discrete logarithm, and elliptic curve relationships. It is computationally easy for a user to generate his or her public and private key-pair and to use them for encryption and decryption. The strength lies in the fact that it is "impossible" (computationally infeasible) for a properly generated private key to be determined from its corresponding public key. Thus the public key may be published without compromising security, whereas the private key must not be revealed to anyone not authorized to read messages or perform digital signatures. Public key algorithms, unlike symmetric key algorithms, do *not* require a secure initial exchange of one (or more) secret keys between the parties.

Message authentication involves processing a message with a private key to produce adigital signature. Thereafter anyone can verify this signature by processing the signature value with the signer's corresponding public key and comparing that result with the message. Success confirms the

message is unmodified since it was signed, and – presuming the signer's private key has remained secret to the signer – that the signer, and no one else, intentionally performed the signature operation. In practice, typically only a hash or digest of the message, and not the message itself, is encrypted as the signature.

Public-key algorithms are fundamental security ingredients in cryptosystems, applications and protocols. They underpin such Internet standards as Transport Layer Security (TLS), PGP, and GPG. Some public key algorithms provide key distribution and secrecy (e.g., Diffie–Hellman key exchange), some provide digital signatures (e.g.,Digital Signature Algorithm), and some provide both (e.g., RSA).

Introduction to Number Theory

1. prime numbers
2. Fermat's and Euler's Theorems &
3. Primality Testing
4. Chinese Remainder Theorem
5. Primitive Roots & Discrete Logarithms

Prime Numbers

• Prime numbers only have divisors of 1 and self they cannot be written as a product of other numbers note: 1 is prime, but is generally not of interest

• eg. 2,3,5,7 are prime, 4,6,8,9,10 are not

• prime numbers are central to number theory

• list of prime number less than 200 is:

2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59

61 67 71 73 79 83 89 97 101 103 107 109 113 127

131 137 139 149 151 157 163 167 173 179 181 191

193 197 199

Fermat's Theorem

• a p--1 = 1 (mod p) where p is prime and GCD(a,p) = 1

• also known as Fermat's Little Theorem

• also have: a p = a (mod p)

• useful in public key and primality testing

Euler's Theorem

• a generalisation of Fermat's Theorem

• a ø(n) = 1 (mod n) for any a,n where GCD(a,n) = 1

• eg.

A = 3 ; n = 10 ; ø (10) = 4; hence 3

4 = 81 = 1 mod 10 a = 2 ;

N = 11 ; ø (11) = 10; hence 2

10 = 1024 = 1 mod 11

• also have: a ø(n)+1 = a (mod n)

Primality Testing

• often need to find large prime numbers

• traditionally sieve using trial division

ie, divide by all numbers (primes) in turn less than the square root of the number only works for small numbers

• alternatively can use statistical primality tests based on properties of primes for which all primes numbers satisfy property but some composite numbers, called pseudo--primes, also satisfy the property

• can use a slower deterministic primality test

Chinese Remainder Theorem

• used to speed up modulo computations, if working modulo a product of numbers

eg.

Mod M = m1m2..mk

• Chinese Remainder theorem lets us work in each modulus mi separately

• Since computational cost is proportional to size, this is faster than working in the full modulus M

Primitive Roots

• from Euler's theorem have a ø(n) mod n = 1

• consider a m = 1 (mod n), GCD(a,n) = 1 must exist for m = ø (n) but may be smaller once powers reach m, cycle will repeat

• if smallest is m = ø (n) then a is called a primitive root

• if p is prime, then successive powers of a "generate" the group mod p

• these are useful but relatively hard to find

Discrete Logarithms

• the inverse problem to exponentiation is to find the discrete logarithmof a number b modulo p

• that is to find i such that b = a i (mod p)

• this is written as i = dloga b (mod p)

• if a is a primitive root mod p then dloga always exists, otherwise it may not,

eg.

X = log3 4 mod 13 has no answer x = log3 3 mod 13 = 4 by trying successive powers

• whilst exponentiation is relatively easy, finding discrete logarithms is generally a hard problem (which is good for cryptography, of course)

RSA Algorithm

RSA is a cryptosystem, which is known as one of the first practicable public-key cryptosystems and is yet widely used for secure data transmission. In such a cryptosystem, the encryption key is public and differs from the decryption key which is kept secret. In RSA, this asymmetry is based on the practical difficulty of factoring the product of two large prime numbers,

the factoring problem. RSA stands for Ron Rivest, Adi Shamir and Leonard Adleman, who first publicly described the algorithm in 1977. Clifford Cocks, an English mathematician, had developed an equivalent system in 1973, but it wasn't declassified until 1997.

A user of RSA creates and then publishes the product of two large prime numbers, along with an auxiliary value, as their public key. The prime factors must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, if the public key is large enough, only someone with knowledge of the prime factors can feasibly decode the message. Whether breaking RSA encryption is as hard as factoring is an open question known as the RSA problem.

Operation

The RSA algorithm involves three steps: key generation, encryption and decryption.

Key generation

RSA involves a **public key** and a **private key.** The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA algorithm are generated the following way:

1. Choose two distinct prime numbers p and q.
   - For security purposes, the integer's p and q should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test.
2. Compute $n = pq$.
   - n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
3. Compute $\varphi(n) = \varphi(p)\varphi(q) = (p - 1)(q - 1)$, where φ is Euler's totient function.
4. Choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$; **i.e.** e and $\varphi(n)$ are coprime.
   - e is released as the public key exponent.

- *e* having a short bit-length and small Hamming weight results in more efficient encryption – most commonly $2^{16} + 1 = 65,537$. However, much smaller values of *e* (such as 3) have been shown to be less secure in some settings.[5]

5. Determine *d* as $d^{-1} \equiv e \pmod{\varphi(n)}$, i.e., *d* is the multiplicative inverse of *e* (modulo φ(*n*)).

- This is more clearly stated as: solve for *d* given $d \cdot e \equiv 1 \pmod{\varphi(n)}$
- This is often computed using the extended Euclidean algorithm.
- *d* is kept as the private key exponent.

The **public key** consists of the modulus *n* and the public (or encryption) exponent *e*. The **private key** consists of the modulus *n* and the private (or decryption) exponent *d*, which must be kept secret. *p*, *q*, and φ(*n*) must also be kept secret because they can be used to calculate *d*.

- An alternative, used by PKCS#1, is to choose *d* matching $de \equiv 1 \pmod{\lambda}$ with $\lambda = \text{lcm}(p-1, q-1)$, where lcm is the least common multiple. Using λ instead of φ(*n*) allows more choices for *d*. λ can also be defined using the Carmichael function, λ(*n*).
- The ANSI X9.31 standard prescribes, IEEE 1363 describes, and PKCS#1 allows, that *p* and *q* match additional requirements: being strong primes, and being different enough that Fermat factorization fails.

Encryption

Alice transmits her public key (*n*, *e*) to Bob and keeps the private key secret. Bob then wishes to send message *M* to Alice.

He first turns *M* into an integer *m*, such that $0 \leq m < n$ by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext *c* corresponding to

$$c \equiv m^e \pmod{n}.$$

This can be done quickly using the method of exponentiation by squaring. Bob then transmits *c* to Alice.

Decryption

Alice can recover *m* from *c* by using her private key exponent *d* via computing

$$m \equiv c^d \pmod{n}.$$

Given *m*, she can recover the original message *M* by reversing the padding scheme.

(In practice, there are more efficient methods of calculating c^d using the precomputed values below.)

Key management

Key management is the management of cryptographic keys in a cryptosystem. This includes dealing with the generation, exchange, storage, use, and replacement of keys. It includes cryptographic protocol design, key servers, user procedures, and other relevant protocols. Key management concerns keys at the user level, either between users or systems. This is in contrast to key scheduling; key scheduling typically refers to the internal handling of key material within the operation of a cipher. Successful key management is critical to the security of a cryptosystem. In practice it is arguably the most difficult aspect of cryptography because it involves system policy, user training, organizational and departmental interactions, and coordination between all of these elements.

Types of Keys

Cryptographic systems may use different types of keys, with some systems using more than one. These may include symmetric keys or asymmetric keys. In a symmetric key algorithm the keys involved are identical for both encrypting and decrypting a message. Keys must be chosen carefully,

and distributed and stored securely. Asymmetric keys, in contrast, are two distinct keys that are mathematically linked. They are typically used in conjunction to communicate.

Diffie-Hellman key exchange (exponential key exchange)

Diffie-Hellman key exchange, also called exponential key exchange, is a method of digitalencryption that uses numbers raised to specific powers to produce decryption keys on the basis of components that are never directly transmitted, making the task of a would-be code breaker mathematically overwhelming.

To implement Diffie-Hellman, the two end users Alice and Bob, while communicating over a channel they know to be private, mutually agree on positive whole numbers p and q, such that p is a prime number and q is a generator of p. The generator q is a number that, when raised to positive whole-number powers less than p, never produces the same result for any two such whole numbers. The value of p may be large but the value of q is usually small.

Once Alice and Bob have agreed on p and q in private, they choose positive whole-number personal keys a and b, both less than the prime-number modulus p. Neither user divulges their personal key to anyone; ideally they memorize these numbers and do not write them down or store them anywhere. Next, Alice and Bob compute public keys $a*$ and $b*$ based on their personal keys according to the formulas

$a* = q^a \bmod p$ and $b* = q^b \bmod p$

The two users can share their public keys $a*$ and $b*$ over a communications medium assumed to be insecure, such as the Internet or a corporate wide area network (WAN). From these public keys, a number x can be generated by either user on the basis of their own personal keys. Alice computes x using the formula

$x = (b^*)^a \bmod p$ Bob computes x using the formula $x = (a^*)^b \bmod p$

The value of x turns out to be the same according to either of the above two formulas. However, the personal keys a and b, which are critical in the calculation of x, have not been transmitted over a public medium. Because it is a large and apparently random number, a potential hacker has almost no chance of correctly guessing x, even with the help of a powerful computer to conduct millions of trials. The two users can therefore, in theory, communicate privately over a public medium with an encryption method of their choice using the decryption key x.

The most serious limitation of Diffie-Hellman in its basic or "pure" form is the lack ofauthentication. Communications using Diffie-Hellman all by itself are vulnerable to man in the middle attacks. Ideally, Diffie-Hellman should be used in conjunction with a recognized authentication method such as digital signatures to verify the identities of the users over the public communications medium. Diffie-Hellman is well suited for use in data communication but is less often used for data stored or archived over long periods of time.

Introduction to Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is a public key cryptography method, which evolved form Diffie Hellman. To understanding how ECC works, lets start by understanding how Diffie Hellmanworks.

The Diffie Hellman key exchange protocol, and the Digital Signature Algorithm (DSA) which is based on it, is an asymmetric cryptographic systems in general use today. It was discovered by Whitfield Diffie and Martin Hellman in 1976, and uses a problem known as the Discrete Logarithm Problem (DLP) as its asymmetric operation. The DLP concerns finding a logarithm of a number within a finite field arithmetic system.

Prime fields are fields whose sets are prime. In other words, they have a prime number of members. Prime fields turn out to be of great use in asymmetric cryptography since exponentiation over a prime field is relatively easy, while its inverse, computing the logarithm, is difficult. The "Diffie-Hellman Method for Key Agreement" allows two hosts to create and share a secret key. This is done by the following method

1. First the hosts must get the "Diffie-Hellman parameters": a prime number p(larger than 2) and "base", g, an integer that is smaller than p. They can be hard coded or fetched from a server, depending on the implementation.
2. The hosts each generate a secret private number called x, which is less than $p-1$.
3. Next, the hosts generate the public keys, y. They are created with the function:

   $y = g^x \bmod p$
4. The two host then exchange the public keys ( ) and these exchanged numbers are then converted into a secret key, z as follows:

   $z = y^x \bmod p$
5. The secret key z can at this point be used as the key for a standard encryption method, used to transfer the information between the hosts. Mathematically, the two hosts have generated the same value for the secret key z since:

$$z = (g^x \bmod p)^{x'} \bmod p = (g^{x'} \bmod p)^x \bmod p$$

Using the values in the equation above, finding the discrete logarithm problem is finding x when only , g and p are known. As an example, take the situation in which someone has multiplied g by itself x times, and reduced the result into the field (by performing the modulo operation) as often as needed to keep the result smaller than p. In this case, when knowing y , g and p, the problem is trying to find what what value of x was used. This turns out to be extraordinarily difficult to do for large enough values of p, where p is prime. It is in fact so much

more difficult to do than just finding y from g, x and p that, even using the world's fastest supercomputer, it would be unfeasible to attempt within a reasonable amount of time. Mathematically, a proof to this effect is neither known nor thought to be forthcoming. Before wide-scale implementation, it is thus of the utmost importance that an extensive investigation of the true complexity of the problem is done in order to obtain the highest degree of confidence in the security of discrete logarithm based cryptographic systems. Such an investigation is in progress by various researchers around the world.

Ellipticcurves

Since the discovery of RSA (and El-Gamal) their ability to withstand attacks has meant that these two cryptographic systems have become widespread in use. They are being used every day both for authentication purposes as well as encryption/decryption. Both systems cover the current security standards--so why invent a new system? Even though ECC is relatively new, the use of elliptic curves as a base for a cryptographic system was independently proposed by By Victor Miller and Neil Koblitz. What makes it stand apart from RSA and El-Gamal is its ability to be more efficient that those two. The reason why this is important are the developments in information technology--most importantly hand held, mobile devices, sensor networks, etc. Somehow, there must be a way to secure communications generated by these devices, however their computing power and memory are not nearly as abundant as on their desktop and laptop counterparts. A contemporary desktop or laptop system has no problems working with 2048 bit keys and higher, but these small embedded devices do since we do not want to spend a lot of their resources and bandwidth securing traffic.

The operations on which RSA are founded are modular exponentiation in integer rings. The security of RSA depends on the difficulty of factoring large integers which can be done in sub-exponential times. For the ECDLP however, only exponential algorithms are known which means

we can use shorter keys for security levels where RSA and El-Gamal would need much bigger keys. For example, a 160 bit ECC key and a 1024 bit RSA key offer a similar level of security. To reach the same level of security than a 15360 bit RSA key, one only needs 512 bit ECC key.

Operations on elliptic curves

The security of ECC depends on the difficulty of the Elliptic Curve Discrete Logarithm Problem. This problem is defined as follows: let P and Q be two points on an elliptic curve such that $kP = Q$, where k is a scalar. Given P and Q, it is computationally unfeasible to obtain k, if k is sufficiently large. Hence, k is the discrete logarithm of Q to P. We can see that the main operation involved in ECC is point multiplication, namely, multiplication of a scalar k with any point P on the curve to obtain another point Q on the curve.

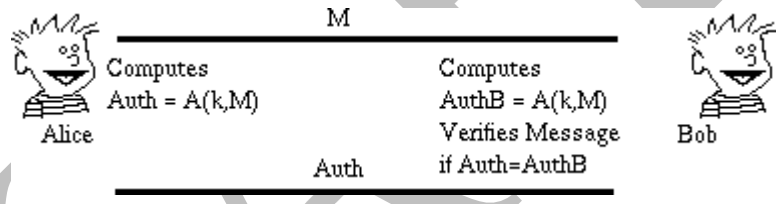
This is also the reason a ECC key of 160 bits provides the equivalent protection of a symmetric key of 80 bits, namely because of the methods used to crack $kP = Q$. If one knows P and Q, one must guess at least the square root of the number of points on average to find k. So if the field size is 2^m, one must guess $2^{(m/2)}$ points. With a 80 bit symmetric key, it takes 2^{79} guesses to crack it on average. The table below gives a comparison of equivalent key sizes.

Bits of Security	Symmetric Algorithm	RSA	ECC
80	2TDEA	$k = 1024$	$f = 160 - 223$
112	3TDEA	$k = 2048$	$f = 224 - 255$
128	AES-128	$k = 3072$	$f = 256 - 383$
192	AES-192	$k = 7680$	$f = 384 - 511$
256	AES-256	$k = 15360$	$f = 512+$

Message Authentication

- message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)

- electronic equivalent of a signature on a message
- an **authenticator**, **signature**, or **message authentication code (MAC)** is sent along with the message
- the MAC is generated via some algorithm which depends on both the message and some (public or private) key known only to the sender and receiver
- the message may be of any length
- the MAC may be of any length, but more often is some fixed size, requiring the use of some **hash function** to condense the message to the required size if this is not acheived by the authentication scheme
- need to consider replay problems with message and MAC
    - require a message sequence number, timestamp or negotiated random values



Authentication using Private-key Ciphers

- if a message is being encrypted using a session key known only to the sender and receiver, then the message may also be authenticated
    - since only sender or receiver could have created it
    - any interference will corrupt the message (provided it includes sufficient redundancy to detect change)
    - but this does not provide non-repudiation since it is impossible to prove who created the message
- message authentication may also be done using the standard modes of use of a block cipher

- sometimes do not want to send encrypted messages
- can use either CBC or CFB modes and send final block, since this will depend on all previous bits of the message
- no hash function is required, since this method accepts arbitrary length input and produces a fixed output
- usually use a fixed known IV
- this is the approached used in Australian EFT standards AS8205
- major disadvantage is small size of resulting MAC since 64-bits is probably too small

Hashing Functions

- hashing functions are used to condense an arbitrary length message to a fixed size, usually for subsequent signature by a digital signature algorithm
- good cryptographic hash function h should have the following properties:
  - h should destroy all homomorphic structures in the underlying public key cryptosystem (be unable to compute hash value of 2 messages combined given their individual hash values)
  - h should be computed on the entire message
  - h should be a one-way function so that messages are not disclosed by their signatures
  - it should be computationally infeasible given a message and its hash value to compute another message with the same hash value
- should resist **birthday attacks** (finding any 2 messages with the same hash value, perhaps by iterating through minor permutations of 2 messages
- it is usually assumed that the hash function is public and not keyed
- traditional CRCs do not satisfy the above requirements
- length should be large enough to resist birthday attacks (64-bits is now regarded as too small, 128-512 proposed)

Message Authentication

In cryptography, a **message authentication code** (often **MAC**) is a short piece of information used to authenticate a message and to provide integrity and authenticity assurances on the message. Integrity assurances detect accidental and intentional message changes, while authenticity assurances affirm the message's origin.

A MAC algorithm, sometimes called a **keyed** (**cryptographic**) **hash function** (however, cryptographic hash function is only one of the possible ways to generate MACs), accepts as input a secret key and an arbitrary-length message to be authenticated, and outputs a MAC (sometimes known as a *tag*). The MAC value protects both a message's data integrity as well as its authenticity, by allowing verifiers (who also possess the secret key) to detect any changes to the message content.

Security

While MAC functions are similar to cryptographic hash functions, they possess different security requirements. To be considered secure, a MAC function must resist existential forgery under chosen-plaintext attacks. This means that even if an attacker has access to an oracle which possesses the secret key and generates MACs for messages of the attacker's choosing, the attacker cannot guess the MAC for other messages (which were not used to query the oracle) without performing infeasible amounts of computation.

MACs differ from digital signatures as MAC values are both generated and verified using the same secret key. This implies that the sender and receiver of a message must agree on the same key before initiating communications, as is the case with symmetric encryption. For the same reason, MACs do not provide the property of non-repudiation offered by signatures specifically in the case of a network-wide shared secret key: any user who can verify a MAC is also capable of generating MACs for other messages. In contrast, a digital signature is generated using the private key of a key pair, which is asymmetric encryption. Since this private key is only accessible to its holder, a digital

signature proves that a document was signed by none other than that holder. Thus, digital signatures do offer non-repudiation. However, non-repudiation can be provided by systems that securely bind key usage information to the MAC key; the same key is in possession of two people, but one has a copy of the key that can be used for MAC generation while the other has a copy of the key in a hardware security module that only permits MAC verification. This is commonly done in the finance industry

Hash Function

A **hash function** is any algorithm that maps data of a variable length to data of a fixed length. The values returned by a hash function are called **hash values**, **hash codes**,**hash sums**, checksums or simply **hashes**.

Hash functions are primarily used to generate fixed-length output data that acts as a shortened reference to the original data. This is useful when the output data is too cumbersome to use in its entirety.

One practical use is a data structure called a hash table where the data is stored associatively. Searching for a person's name in a list is slow, but the hashed value can be used to store a reference to the original data and retrieve constant time (barring collisions). Another use is in cryptography, the science of encoding and safeguarding data. It is easy to generate hash values from input data and easy to verify that the data matches the hash, but hard to 'fake' a hash value to hide malicious data. This is the principle behind thePretty Good Privacy algorithm for data validation.

Hash functions are also used to accelerate table lookup or data comparison tasks such as finding items in a database, detecting duplicated or similar records in a large file, finding similar stretches in DNA sequences, and so on. A hash function should be referentially transparent (stable), i.e., if called twice on input that is "equal" (for example, strings that consist of the same sequence of characters), it should give the same result. There is a construct in many programming languages that allows the user to override equality and hash functions for an object: if two objects are equal, their hash codes must be the same. This is crucial to finding an element in a hash table quickly, because two of the same element would both hash to the same slot. Hash functions are destructive, akin to lossy compression, as the original data is lost when hashed. Unlike compression algorithms, where something resembling the original data can be decompressed from compressed data, the goal of a hash value is to uniquely identify a reference to the object so that it can be retrieved in its entirety. Unfortunately, all hash functions that map a larger set of data to a smaller set of data cause collisions. Such hash functions try to map the keys to the hash values as evenly as possible because collisions become more frequent as hash tables fill up. Thus, single-digit hash values are frequently restricted to 80% of the size of the table. Depending on the algorithm used, other properties may be required as well, such as double

hashing and linear. Although the idea was conceived in the 1950s, the design of good hash functions is still a topic of active research. Hash functions are related to (and often confused with) checksums, check digits, fingerprints, randomization functions, error-correcting codes, and cryptographic. Although these concepts overlap to some extent, each has its own uses and requirements and is designed and optimized differently. The Hash Keeper database maintained by the American National Drug Intelligence Center, for instance, is more aptly described as a catalog of file fingerprints than of hash values.

Message Authentication and Hash Functions

Message Authentication

- Message authentication is concerned with:
  - Protecting the integrity of a message
  - Validating identity of originator
  - Non-repudiation of origin (dispute resolution)
- Will consider the security requirements
- Then three alternative functions used:
  - Message encryption
  - Message authentication code (mac)
  - Hash function

Message Authentication Code (MAC)

- Generated by an algorithm that creates a small fixed-sized block
  - Depending on both message and some key
  - Like encryption though need not be reversible
- Appended to message as a signature
- Receiver performs same computation on message and checks it matches the mac
- Provides assurance that message is unaltered and comes from sender.

MAC Properties

- A MAC is a cryptographic checksum

  $MAC = C_K(M)$
  - Condenses a variable-length message M

- Using a secret key K
- To a fixed-sized authenticator

- Is a many-to-one function
  - Potentially many messages have same MAC
  - But finding these needs to be very difficult

Requirements for MACs

- Taking into account the types of attacks
- Need the mac to satisfy the following:
  - Knowing a message and mac, is infeasible to find another message with same mac
  - Macs should be uniformly distributed
  - Mac should depend equally on all bits of the message

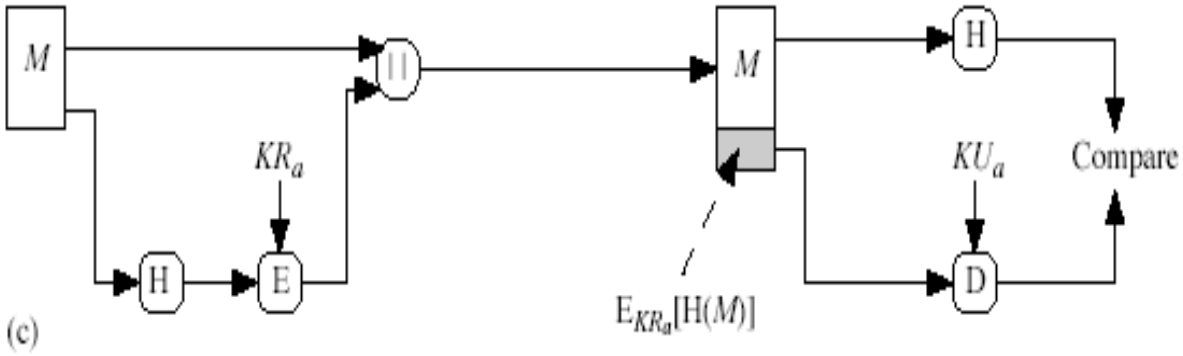
Using Symmetric Ciphers for macs

- Can use any block cipher chaining mode and use final block as a MAC
- Data Authentication Algorithm (DAA) is a widely used MAC based on DES-CBC
  - Using IV=0 and zero-pad of final block
  - Encrypt message using DES in CBC mode
  - And send just the final block as the MACOr the leftmost M bits ($16 \leq M \leq 64$) of final block
- But final MAC is now too small for security.

Hash Functions

- Condenses arbitrary message to fixed size
- Usually assume that the hash function is public and not keyed
  - Cf. Mac which is keyed
- Hash used to detect changes to message
- Can use in various ways with message
- Most often to create a digital signature

Hash Functions & Digital Signatures



Hash Function Properties

- A Hash Function produces a fingerprint of some file/message/data

  H = H(M)
  - Condenses a variable-length message M
  - To a fixed-sized fingerprint
- Assumed to be public

Requirements for Hash Functions

- can be applied to any sized message M
- produces fixed-length output h
- is easy to compute h=H(M) for any message M
- given h is infeasible to find x s.t. H(x)=h
    - one-way property
- given x is infeasible to find y s.t. H(y)=H(x)
    - weak collision resistance
- is infeasible to find any x,ys.t. H(y)=H(x)
    - strong collision resistance

Digital Signatures and Authentication Protocol

Digital signatures are often used to implement electronic signatures, a broader term that refers to any electronic data that carries the intent of a signature, but not all electronic signatures use digital signatures. In some countries, including the United States, India, and members of the European Union, electronic signatures have legal significance.Digital signatures employ a type of asymmetric cryptography. For messages sent through a nonsecure channel, a properly implemented digital signature gives the receiver reason to believe the message was sent by the claimed sender. Digital signatures are equivalent to traditional handwritten signatures in many respects, but properly implemented digital signatures are more difficult to forge than the handwritten type. Digital signature schemes, in the sense used here, are cryptographically based, and must be implemented properly to be effective. Digital signatures can also provide non-repudiation, meaning

that the signer cannot successfully claim they did not sign a message, while also claiming their private key remains secret; further, some non-repudiation schemes offer a time stamp for the digital signature, so that even if the private key is exposed, the signature is valid. Digitally signed messages may be anything represent able as a bit string: examples include electronic mail, contracts, or a message sent via some other cryptographic protocol.

A digital signature scheme typically consists of three algorithms:

- A key generation algorithm that selects a private key uniformly at random from a set of possible private keys. The algorithm outputs the private key and a corresponding public key.
- A signing algorithm that, given a message and a private key, produces a signature.
- A signature verifying algorithm that, given the message, public key and a signature, either accepts or rejects the message's claim to authenticity.

Two main properties are required. First, a signature generated from a fixed message and fixed private key should verify the authenticity of that message by using the corresponding public key. Secondly, it should be computationally impossible to generate a valid signature for a party without knowing that party's private key.

A digital signature is an authentication mechanism that allows the sender to attach an electronic code with the message in order to ensure its authenticity and integrity. This electronic code acts as the signature of the sender and, hence, is named digital signature. Digital signatures use the public-key cryptography technique. The sender uses his or her private key and a signing algorithm to create a digital signature, and the signed document can be made public. The receiver, on the other hand, uses the public key of the sender and a verifying algorithm to verify the digital signature. Digital signatures are analogous to physical handwritten signatures and provide the following security services.

CLASS: II MCA **COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY**

COURSE CODE: 17CAP404N **UNIT: II** **BATCH-2017-2020**

UNIT II

POSSIBLE QUESTIONS

1. Describe about Message Authentication and Hash Functions?
2. Explain in detail about Digital signature
3. Explain in detail about Key management
4. Explain in detail about Digital signature and Authentication protocol.
5. Explain in detail about RSA algorithm.
6. Describe about Fermat's and Euler's theorems.
7. Explain Diffie-Hellman key Exchange Algorithm.
8. Explain in detail about Message Authentication Code
9. Explain in detail about principles of public key cryptosystem.
10. Describe about Elliptic Curve cryptography?
11. Describe about encryption and decryption using Column transposition method.

KARPAGAM ACADEMY OF HIGHER EDUCATION
DEPARTMENT OF COMPUTER APPLICATIONS
II MCA
CRYPTOGRAPHY AND NETWORK SECURITY

UNIT II

S.NO	Question	Option 1	Option 2	Option 3	Option 4	Answer
1	A _____ is one which a block of plain text is treated as a whole and used to produce a ciphertext block of equal length.	stream cipher	symmetric cipher	block cipher	none	block cipher
2	The _____ has been the most widely used encryption algorithm until recently	DES	AES	AFS	DEM	DES
3	A _____ is one that encrypts a digital data streem one bit or one byte at a time.	block cipher	symmetric cipher	stream cipher	plain	stream cipher
4	A block cipher operates on a plain text block of n bits to produce a ciphertext block of _____ bits	n*n*	n*n	n power n	n	n
5	most symmetric block encryption algorithms in current use are based on a structure referred to as a _____ block cipher.	Feistel	block	stream	none	Feistel
6	Feistel refers to this as the _____ cipher because it allows for the maximum number of possible encryption mappings from the plaintext block.	assymmetric	product	symmetric	ideal block	ideal block
7	the terms _____ and _____ were introduced by claude shannon to capture the two basic building blocks for any cryptographic system	diffusion confusion	diffusion	permutation	none	diffusion confusion
8	In _____ the statical structure of the plain text is dissipated into long range statistics of the ciphertext.	diffusion	substitution	permutation	confussion	diffusion
9	_____ seeks to make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible again	permutation	substitution	confussion	diffusion	confussion
10	A_____ is performed on the left half of the data	substitution	permutation	confussion	trasformation	substitution
11	A_____ is performed that consists of the interchange of the two halves of the data	substitution	substitution	confussion	trasformation	substitution
12	A block size of _____ bits has been considered a resonable tradeoff and was nearly universal in block cipher design	16	8	32	64	64
13	_____ key size means greated security but may decrease encryption/decryption speed	larger	smaller	medium	double	larger
14	The DES adopted in _____ by the national bureaue of standards	1977	1978	1979	1980	1977
15	NIST stands for	national institute of standards and technology.	national institute of state and technology	national industiral of standards and technology.	national institute of state and telephone.	national institute of standards and technology.
16	For DES data are encrypted in ___ blocks using a _____bit key.	8 bit,25	32bit,45	64 bit,56	16 bit,55	64 bit,56
17	The left and right halves of the output are swapped to produce the _____	preoutput	preinput	postinput	postoutput	preoutput
18	The _____ function is the same for each round but a different subkey is produced because of the repeated shifts of the key bits.	substitution	permutation	confussion	trasformation	permutation
19	with a key length of 56 bits there are 2power 56 possible keys which is approximately _____ keys	7.2*10 power 16	7.5*10 power 16	7.7*10 power 16	795*10 power 16	7.2*10 power 16
20	EFF stands for	Electronic Frontier Foundation	Electronic Form Foundation	Electric Form Foundation	Email Frontier Foundation	Electronic Frontier Foundation
21	AES stands for	Aavent Encryption Standard	Advanced Enable Standard	Advanced Encryption Standard	Advanced Encryption Stante	Advanced Encryption Standard
22	_____ refers to the effort required to cryptanalyze an algorithm.	security	cost	randomness	product	security
23	_____ of the mathematical basis for the algorithms security	soundness	randomness	cost	product	soundness
24	A candidate algorithm shall be judged according to relative _____ of design.	permutation	complexcity	random	simplicity	simplicity
25	_____ algorithms with greater flexibility will meet the needs of more user than less flexible ones.	candidate	primary	super	foreign	candidate
26	_____ refers to the ability to change keys quickly and with a minimum of resources.	key agility	key enable	key distribution	key change	key agility
27	The forward substitute byte transformation called _____ bytes	max	super	min	sub	sub
28	The _____ key expansion algorithm takes as input a 4 word key and produce a linear of 44 words	AES	DES	CIS	AEM	AES
29	AES can be implemented very efficiently on an _-bit processor	8	16	32	64	8
30	_____ uses an s-box to perform a byte-by-byte substitution of the block	substitution butes	permutation butes	diffusion butes	confussion butes	substitution butes
31	ECB stands for	electronic coin book	electronic code book	electronic code bill	email code book	electronic code book
32	CBC	Cipher Block Chating	Cylinder Block Chaining	Cipher Base Chaining	Cipher Block Chaining	Cipher Block Chaining
33	CFB	Cipher Formback	Cylinder Feedback	Cipher Front	Cipher Feedback	Cipher Feedback
34	The ____mode is similar in structure to that of CFB	OFB	CTR	CTL	CTTL	OFB
35	the _____ mode has increased recently with applications to ATM network security	CTL	OFB	CTR	CTTL	CTR
36	ATM stands for		Asynchronous teller mode	All time teller mode	Asynchronous target mode	Asynchronous transfer mode
37	_____ the execution of the underlying encryption algorithm does not depend on input of the plaintext or ciphertext.	preprocessing	software	hardware	none	preprocessing
38	_____ can be shown that CRT is at least as secure as the other modes	random access	Provable Security	simplicity	software	Provable Security
39	_____CRT modes requires only the implementation of the encryption algorithm and not decryption algorithm.	simplicity	random access	preprocessing	software	simplicity
40	_____ is a stream cipher designed in 1987 by Ron Rivest for RSA security	RC4	RC5	RC6	RC7	RC4

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

COURSE CODE: 17CAP404N UNIT: III BATCH-2017-2020

SYLLABUS

Network Security Applications - Authentication Applications – KERBEROS – X.509 Authentication Service – Public Key Infrastructure – Electronic Mail Security – Pretty Good Privacy – S/MIME – IP Security.

NETWORK SECURITY

Kerberos is an authentication service designed for use in a distributed environment.

● Kerberos makes use of a trusted third-part authentication service that enables clients and servers to establish authenticated communication.

AUNTENTICATION APPLICATIONS : KERBEROS

Kerberos provides a centralized authentication server whose function is to authenticate users to servers and servers to users. Kerberos relies exclusively on conventional encryption, making no use of public-key encryption.

The following are the requirements for Kerberos:

- **Secure:** A network eavesdropper should not be able to obtain the necessary information to impersonate a user. More generally, Kerberos should be strong enough that a potential opponent does not find it to be the weak link.
- **Reliable:** For all services that rely on Kerberos for access control, lack of availability of the Kerberos service means lack of availability of the supported services. Hence, Kerberos should be highly reliable and should employ a distributed server architecture, with one system able to back up another.
- **Transparent:** Ideally, the user should not be aware that authentication is taking place, beyond the requirement to enter a password.
- **Scalable:** The system should be capable of supporting large numbers of clients and servers. This suggests a modular, distributed architecture.

To support these requirements, the overall scheme of Kerberos is that of a trusted third- party

authentication service that uses a protocol based on that proposed by Needham and Schroeder. It is trusted in the sense that clients and servers trust Kerberos to mediate their mutual authentication. Assuming the Kerberos protocol is well designed, then the authentication service is secure if the Kerberos server itself is secure.

A simple authentication dialogue

In an unprotected network environment, any client can apply to any server for service. The obvious security risk is that of impersonation. To counter this threat, servers must be able to confirm the identities of clients who request service. But in an open environment, this places a substantial burden on each server.

An alternative is to use an authentication server (AS) that knows the passwords of all users and stores these in a centralized database. In addition, the AS shares a unique secret key with each server. The simple authentication dialogue is as follows:

1. C >> AS: IDc||Pc||IDv
2. AS >> C: Ticket
3. C >> V: IDc||Ticket

Ticket= EKv (IDc||ADc||IDv)

AS	: Authentication Server	V	: Server
IDc	: ID of the client	IDv	: ID of the server
Kv	: secret key shared by AS and V	Pc	:Password of the client
ADc	: Address of client	||	: concatenation

A more secure authentication dialogue

There are two major problems associated with the previous approac[redacted] Plaintext transmission of the password.

[redacted] Each time a user has to enter the password.

To solve these problems, we introduce a scheme for avoiding plaintext passwords, and anew server, known as ticket granting server (TGS). The hypothetical scenario is as follows:

Once per user logon session:

1. C >> AS: IDc||IDtgs
2. AS >> C: Ek_c ($Ticket_{tgs}$)

Once per type of service:

3. C >> TGS: IDc||IDv||$Ticket_{tgs}$
4. TGS >> C: $ticket_v$

Once per service session:

5. C >> V: IDc||$ticket_v$

$Ticket_{tgs}$= Ek_{tgs}(IDc||ADc||IDtgs||TS1||Lifetime1)

$Ticket_v$= Ek_v(IDc||ADc||IDv||TS2||Lifetime2)

C	: Client	V	: Server
IDc	: ID of the client	AS	: Authentication Server
Pc	:Password of the client	ADc	: Address of client
Kv	: secret key shared by AS and V	IDv	: ID of the server
||	: concatenation	IDtgs	: ID of the TGS server
TS1, TS2	: time stamps	lifetime	: lifetime of the ticket

The new service, TGS, issues tickets to users who have been authenticated to AS. Thus, the user first requests a ticket-granting ticket ($Ticket_{tgs}$) from the AS. The client module in the user workstation saves this ticket. Each time the user requires access to a new service, the client applies to the TGS, using the ticket to authenticate itself. The TGS then grants a ticket for the particular service. The client saves each service-granting ticket and uses it to authenticate its user to a server each time a particular service is requested. Let us look at the details of this scheme:

1. The client requests a ticket-granting ticket on behalf of the user by sending its user's ID and password to the AS, together with the TGS ID, indicating a request to use the TGS service.

2. The AS responds with a ticket that is encrypted with a key that is derived from the user's password.

When this response arrives at the client, the client prompts the user for his or her password, generates the key, and attempts to decrypt the incoming message. If the correct password is supplied, the ticket is successfully recovered.

Because only the correct user should know the password, only the correct user can recover the ticket. Thus, we have used the password to obtain credentials from Kerberos without having to transmit the password in plaintext. Now that the client has a ticket-granting ticket, access to any server can be obtained with steps 3 and 4:

3. The client requests a service-granting ticket on behalf of the user. For this purpose, the client transmits a message to the TGS containing the user's ID, the ID of the desired service, and the ticket-granting ticket.
4. The TGS decrypts the incoming ticket and verifies the success of the decryption by the presence of its ID. It checks to make sure that the lifetime has not expired. Then it compares the user ID and network address with the incoming information to authenticate the user. If the user is permitted access to the server V, the TGS issues a ticket to grant access to the requested service.

The service-granting ticket has the same structure as the ticket-granting ticket. Indeed, because the TGS is a server, we would expect that the same elements are needed to authenticate a client to the TGS and to authenticate a client to an application server. Again, the ticket contains a timestamp and lifetime. If the user wants access to the same service at a later time, the client can simply use the previously acquired service-granting ticket and need not bother the user for a password. Note that the ticket is encrypted with a secret key (K_V) known only to the TGS and the server, preventing alteration.

Finally, with a particular service-granting ticket, the client can gain access to the corresponding service with step 5:

5. The client requests access to a service on behalf of the user. For this purpose, the client transmits a message to the server containing the user's ID and the service-granting ticket. The server authenticates by using the contents of the ticket.

This new scenario satisfies the two requirements of only one password query per user session and protection of the user password.

Kerbero V4 Authentication Dialogue Message Exchange

Two additional problems remain in the more secure authentication dialogue:

- Lifetime associated with the ticket granting ticket. If the lifetime is very short, then the user will be repeatedly asked for a password. If the lifetime is long, then the opponent has the greater opportunity for replay.
- Requirement for the servers to authenticate themselves to users. The actual Kerberos protocol version 4 is as follows :
  - a basic third-party authentication scheme
  - have an Authentication Server (AS)
    - users initially negotiate with AS to identify self
      - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)

- have a Ticket Granting server (TGS)

  – users subsequently request access to other services from TGS on basis of users TGT

(a) Authentication Service Exchange: to obtain ticket-granting ticket

(1) C → AS: $ID_c \| ID_{tgs} \| TS_1$

(2) AS → C: $E_{K_c}[K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}]$

$$Ticket_{tgs} = E_{K_{tgs}}[K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2]$$

(b) Ticket-Granting Service Exchange: to obtain service-granting ticket

(3) C → TGS: $ID_v \| Ticket_{tgs} \| Authenticator_c$

(4) TGS → C: $E_{K_{c,tgs}}[K_{c,v} \| ID_v \| TS_4 \| Ticket_v]$

$$Ticket_{tgs} = E_{K_{tgs}}[K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2]$$

$$Ticket_v = E_{K_v}[K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4]$$

$$Authenticator_c = E_{K_{tgs}}[ID_C \| AD_C \| TS_3]$$

(c) Client/Server Authentication Exchange: to obtain service

(5) C → V: $Ticket_v \| Authenticator_c$

(6) V → C: $E_{K_{c,v}}[TS_5 + 1]$ (for mutual authentication)

$$Ticket_v = E_{K_v}[K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4]$$

$$Authenticator_c = E_{K_{c,v}}[ID_C \| AD_C \| TS_5]$$

Table 4.1 illustrates the mode of dialogue in V4

Message (1)	**Client requests ticket-granting ticket**
IDC	Tells AS identity of user from this client
ID_{tgs}	Tells AS that user requests access to TGS
TS1	Allows AS to verify that client's clock is synchronized with that of A
Message (2)	**AS returns ticket-granting ticket**
Kc	Encryption is based on user's password, enabling AS and client to verify password, and protecting contents of message (2)

$K_{c,tgs}$	Copy of session key accessible to client created by AS to permit sec exchange between client and TGS without requiring them to share a permanent key
ID_{tgs}	Confirms that this ticket is for the TGS
$K_{c,tgs}$	
ID_{tgs}	Confirms that this ticket is for the TGS
TS2	Informs client of time this ticket was issued
Lifetime2	Informs client of the lifetime of this ticket
$Ticket_{tgs}$	Ticket to be used by client to access TGS
	(a) Authentication Service Exchange
Message (3)	**Client requests service-granting ticket**
IDV	Tells TGS that user requests access to server V
$Ticket_{tgs}$	Assures TGS that this user has been authenticated by AS
$Authenticator_c$	Generated by client to validate ticket
Message (4)	**TGS returns service-granting ticket**
$K_{c,tgs}$	Key shared only by C and TGS protects contents of message (4)
$K_{c,v}$	Copy of session key accessible to client created by TGS to permit secure exchange between client and server without requiring them to share a permanent key
ID_v	Confirms that this ticket is for server V
TS4	Informs client of time this ticket was issued

$Ticket_v$	Ticket to be used by client to access server V
$Ticket_{tgs}$	Reusable so that user does not have to reenter password
K_{tgs}	Ticket is encrypted with key known only to AS and TGS, to prevent tampering
$K_{c,tgs}$	Copy of session key accessible to TGS used to decrypt authenticator thereby authenticating ticket
IDC	Indicates the rightful owner of this ticket
ADC	Prevents use of ticket from workstation other than one that initially requested the ticket
ID_{tgs}	Assures server that it has decrypted ticket properly
TS2	Informs TGS of time this ticket was issued
Lifetime2	Prevents replay after ticket has expired
$Authenticator_c$	Assures TGS that the ticket presenter is the same as the client for whom the ticket was issued has very short lifetime to prevent replay
$K_{c,tgs}$	Authenticator is encrypted with key known only to client and TGS, t prevent tamperig
ID_c	Must match ID in ticket to authenticate ticket
AD_c	Must match address in ticket to authenticate ticket
TS3	Informs TGS of time this authenticator was generated
	(b) Ticket-Granting Service Exchange
Message (5)	**Client requests service**
$Ticket_v$	Assures server that this user has been authenticated by AS

$Authenticator_C$	Generated by client to validate ticket
Message (6)	Optional authentication of server to client
$K_{c,v}$	Assures C that this message is from V
$TS_5 + 1$	Assures C that this is not a replay of an old reply
$Ticket_V$	Reusable so that client does not need to request a new ticket from T for each access to the same server
K_V	Ticket is encrypted with key known only to TGS and server, to prev tampering
$K_{c,v}$	Copy of session key accessible to client; used to decrypt authenticat thereby authenticating ticket
IDC	Indicates the rightful owner of this ticket
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket
ID_V	Assures server that it has decrypted ticket properly
TS_4	Informs server of time this ticket was issued
$Lifetime_4$	Prevents replay after ticket has expired
$Authenticator_C$	Assures server that the ticket presenter is the same as the client for whom the ticket was issued; has very short lifetime to prevent replay
$K_{c,v}$	Authenticator is encrypted with key known only to client and server, prevent tampering
IDC	Must match ID in ticket to authenticate ticket
AD_C	Must match address in ticket to authenticate ticket
TS_5	Informs server of time this authenticator was generated

	(c) Client/Server Authentication

Kerberos 4 Overview

Kerberos Realms and Multiple Kerberi

A full-service Kerberos environment consisting of a Kerberos server, a number of clients, and a number of application servers requires the following:

1. The Kerberos server must have the user ID and hashed passwords of all participating users in its database. All users are registered with the Kerberos server.
2. The Kerberos server must share a secret key with each server. All servers are registered with the Kerberos server.

Such an environment is referred to as a **Kerberos realm**.

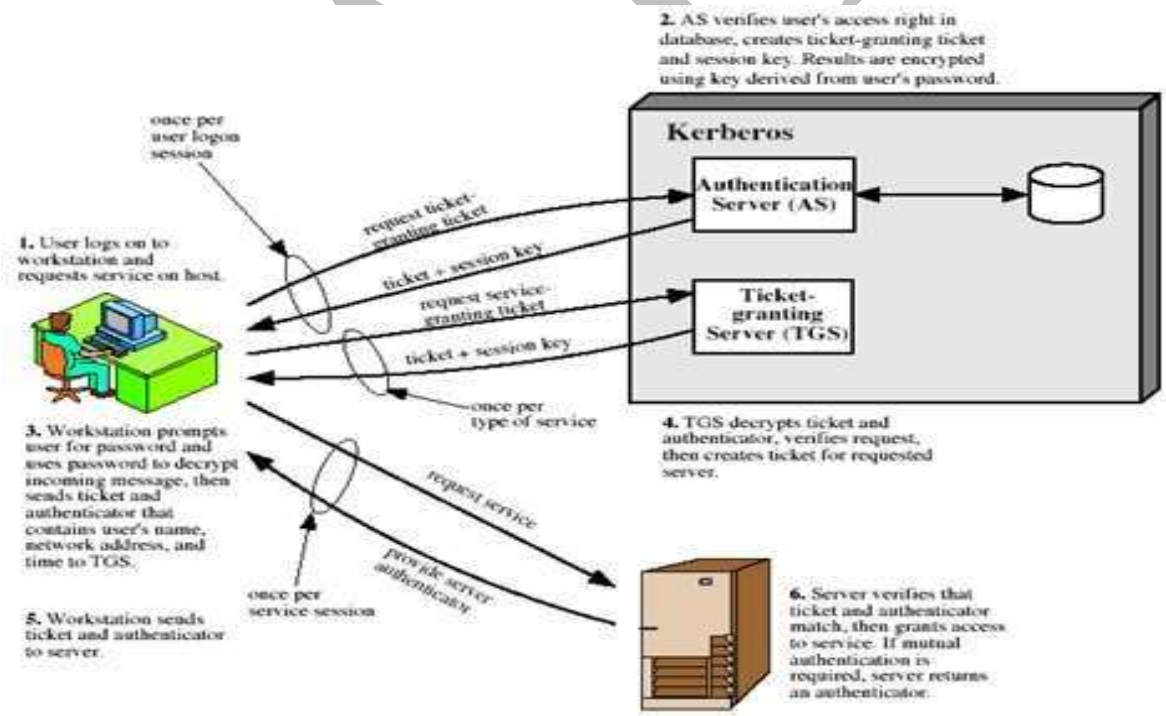


Figure 4.1 Kerberos

The concept of *realm* can be explained as follows.

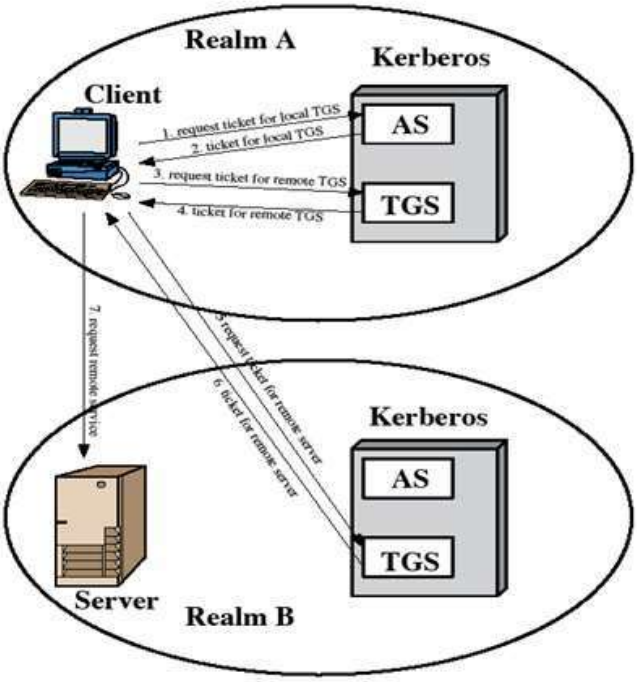


Figure 4.2 Request for Service in Another Realm

A Kerberos realm is a set of managed nodes that share the same Kerberos database. The Kerberos database resides on the Kerberos master computer system, which should be kept in a physically secure room. A read-only copy of the Kerberos database might also reside on other Kerberos computer systems. However, all changes to the database must be made on the master computer system. Changing or accessing the contents of a Kerberos database requires the Kerberos master password.

A related concept is that of a Kerberos principal, which is a service or user that is known to the Kerberos system. Each Kerberos principal is identified by its principal name. Principal names consist of three parts: a service or user name, an instance name, and a realm name.

Networks of clients and servers under different administrative organizations typically constitute different realms. That is, it generally is not practical, or does not conform to administrative policy, to have users and servers in one administrative domain registered w ith a Kerberos server elsewhere. However, users in one realm may need access to servers in other realms, and some servers may be willing to provide service to users from other realms, provided that those users are authenticated. Kerberos provides a mechanism for supporting such interrealm authentication. For two

realms to support interrealm authentication, a third requirement is added:

3. The Kerberos server in each interoperating realm shares a secret key with the server in the other realm. The two Kerberos servers are registered with each other.

The scheme requires that the Kerberos server in one realm trust the Kerberos server in the other realm to authenticate its users. Furthermore, the participating servers in the second realm must also be willing to trust the Kerberos server in the first realm.

Kerberos version 5

Version 5 of Kerberos provides a number of improvements over version 4. • developed in mid 1990's

- provides improvements over v4
  - –addresses environmental shortcomings
  - –and technical deficiencies
- specified as Internet standard RFC 1510

Differences between version 4 and 5

Version 5 is intended to address the limitations of version 4 in two areas:

Environmental shortcomings

- o encryption system dependence
- o internet protocol dependence
- o message byte ordering
- o ticket lifetime
- o authentication forwarding
- o inter-realm authentication

Technical deficiencies

- o double encryption
- o PCBC encryption
- o Session keys
- o Password attacks

The version 5 authentication dialogue

(a) Authentication Service Exchange: to obtain ticket-granting ticket
(1) C → AS: Options ‖ ID_c ‖ $Realm_c$ ‖ ID_{tgs} ‖ $Times$ ‖ $Nonce_1$
(2) AS → C: $Realm_c \| ID_C \| Ticket_{tgs} \| E_{K_c}[K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}]$
$Ticket_{tgs} = E_{K_{tgs}}[Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times]$
(b) Ticket-Granting Service Exchange: to obtain service-granting ticket
(3) C → TGS: Options ‖ ID_v ‖ $Times$ ‖ ‖ $Nonce_2$ ‖ $Ticket_{tgs}$ ‖ $Authenticator_c$
(4) TGS → C: $Realm_c \| ID_C \| Ticket_v \| E_{K_{c,tgs}}[K_{c,v} \| Times \| Nonce_2 \| Realm_v \| ID_V]$
$Ticket_{tgs} = E_{K_{tgs}}[Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times]$
$Ticket_v = E_{K_v}[Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times]$
$Authenticator_c = E_{K_{c,tgs}}[ID_C \| Realm_c \| TS_1]$
(c) Client/Server Authentication Exchange: to obtain service
(5) C → V: Options ‖ $Ticket_v$ ‖ $Authenticator_c$
(6) V → C: $E_{K_{C,V}}$ [TS_2 ‖ Subkey ‖ Seq#]
$Ticket_v = E_{K_v}[Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times]$
$Authenticator_c = E_{K_{c,V}}[ID_C \| Realm_c \| TS_2 \| Subkey \| Seq\#]$

First, consider the authentication service exchange. Message (1) is a client request for a ticket-granting ticket. As before, it includes the ID of the user and the TGS. The following new elements are added:

[redacted]ealm: Indicates realm of user

[redacted]ptions: Used to request that certain flags be set in the returned ticket

[redacted]imes: Used by the client to request the following time settings in the ticket:

from: the desired start time for the requested ticket

till: the requested expiration time for the requested ticket

rtime: requested renew-till time

[redacted]**once**: A random value to be repeated in message (2) to assure that the response is fresh and has not been replayed by an opponent

Message (2) returns a ticket-granting ticket, identifying information for the client, and a block encrypted using the encryption key based on the user's password.

This block includes the session key to be used between the client and the TGS, times specified in message (1), the nonce from message (1), and TGS identifying information.

The ticket itself includes the session key, identifying information for the client, the requested time values, and flags that reflect the status of this ticket and the requested options.

These flags introduce significant new functionality to version 5. For now, we defer a

discussion of these flags and concentrate on the overall structure of the version 5 protocol.

Let us now compare the ticket-granting service exchange for versions 4 and 5. We see that message (3) for both versions includes an authenticator, a ticket, and the name of the requested service.

In addition, version 5 includes requested times and options for the ticket and a nonce, all with functions similar to those of message (1).

The authenticator itself is essentially the same as the one used in version 4.

Message (4) has the same structure as message (2), returning a ticket plus information needed by the client, the latter encrypted with the session key now shared by the client and the TGS.

Finally, for the client/server authentication exchange, several new features appear in version 5. In message (5), the client may request as an option that mutual authentication is required. The authenticator includes several new fields as follows:

- **Subkey**: The client's choice for an encryption key to be used to protect this specific application session. If this field is omitted, the session key from the ticket ($K_{c,v}$) is used.
- **Sequence number**: An optional field that specifies the starting sequence number to be used by the server for messages sent to the client during this session. Messages may be sequence numbered to detect replays.

If mutual authentication is required, the server responds with message (6). This message includes the timestamp from the authenticator. Note that in version 4, the timestamp was incremented by one. This is not necessary in version 5 because the nature of the format of messages is such that it is not possible for an opponent to create message (6) without knowledge of the appropriate encryption keys.

Ticket Flags

The flags field included in tickets in version 5 supports expanded functionality compared to that available in version 4.

X.509 CERTIFICATES OVERVIEW:

- X.509 defines the format for public-key certificates. This format is widely used in a variety of applications.
- A public key infrastructure (PKI) is defined as the set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates based on asymmetric cryptography.
- Typically, PKI implementations make use of X.509 certificates.

- issued by a Certification Authority (CA), containing:
  - version (1, 2, or 3)
  - serial number (unique within CA) identifying certificate

–signature algorithm identifier

– issuer X.500 name (CA)

– period of validity (from - to dates)

– subject X.500 name (name of owner)

– subject public-key info (algorithm, parameters, key) – issuer unique identifier (v2+)

– subject unique identifier (v2+) – extension fields (v3)

– signature (of hash of all fields in certificate)

- **notation CA<<A>> denotes certificate for A signed by CA**

X.509 defines a framework for the provision of authentication services by the X.500 directory to its users. The directory may serve as a repository of public-key certificates. Each certificate contains the public key of a user and is signed with the private key of a trusted certification authority. In addition, X.509 defines alternative authentication protocols based on the use of public-key certificates.

X.509 is an important standard because the certificate structure and authentication protocols defined in X.509 are used in a variety of contexts. For example, the X.509 certificate format is used in S/MIME), IP Security and SSL/TLS and SET

X.509 is based on the use of public-key cryptography and digital signatures. The standard does not dictate the use of a specific algorithm but recommends RSA. The digital signature scheme is assumed to require the use of a hash function.

Certificates

The heart of the X.509 scheme is the public-key certificate associated with each user. These user certificates are assumed to be created by some trusted certification authority (CA) and placed in the directory by the CA or by the user.

ersion:

Differentiates among successive versions of the certificate format; the default is version 1. If the Issuer Unique Identifier or Subject Unique Identifier are present, the value must be version 2. If one or more extensions are present, the version must be version 3.

Serial number:
An integer value, unique within the issuing CA, that is unambiguously associated with this certificate.

Signature algorithm identifier:
The algorithm used to sign the certificate, together with any associated parameters. Because this information is repeated in the Signature field at the end of the certificate, this field has little, if any, utility.

Issuer name:
X.500 name of the CA that created and signed this certificate.

Period of validity:
Consists of two dates: the first and last on which the certificate is valid.

Subject name:
The name of the user to whom this certificate refers. That is, this certificate certifies the public key of the subject who holds the corresponding private key.

Subject's public-key information:
The public key of the subject, plus an identifier of the algorithm for which this key is to be used, together with any associated parameters.

Issuer unique identifier:
An optional bit string field used to identify uniquely the issuing CA in the event the X.500 name has been reused for different entities.

Subject unique identifier:
An optional bit string field used to identify uniquely the subject in the event the X.500 name has been reused for different entities.

Extensions:
A set of one or more extension fields. Extensions were added in version 3 and are discussed later in this section.

Signature:
Covers all of the other fields of the certificate; it contains the hash code of the other fields, encrypted with the CA's private key. This field includes the signature algorithm identifie

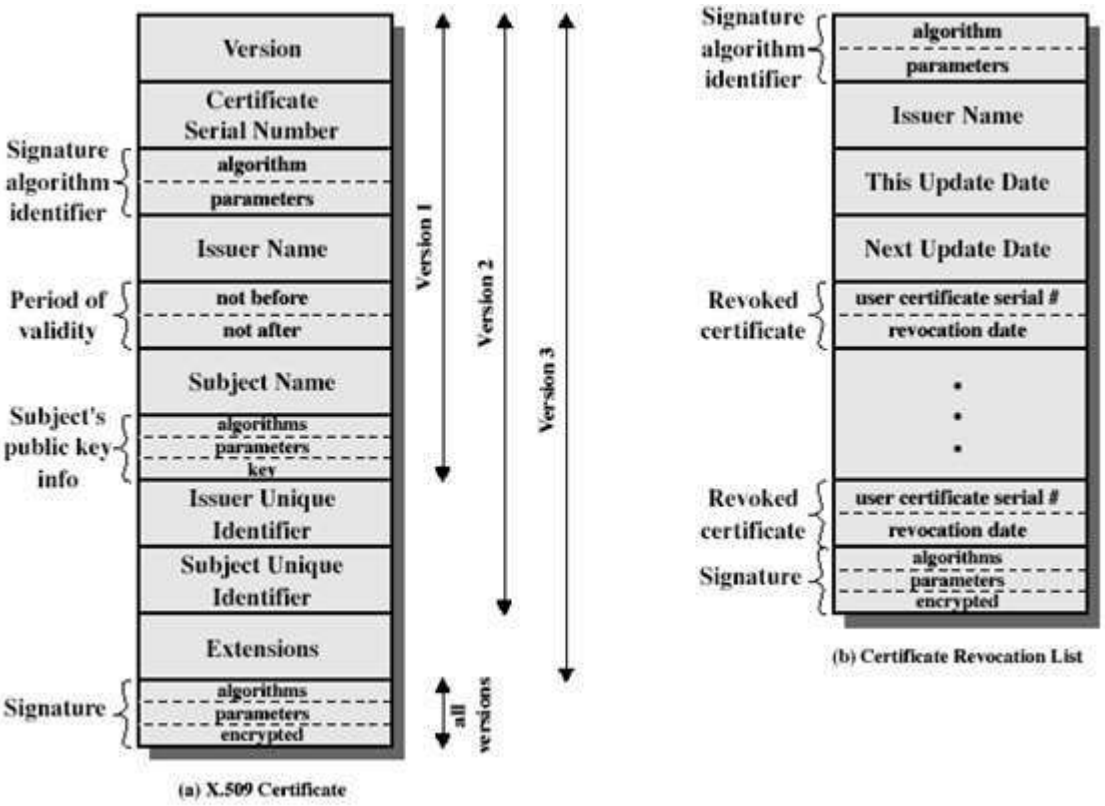


Figure 4.3 : X.509 Formats

The standard uses the following notation to define a certificate:

CA<<A>> = CA {V, SN, AI, CA, TA, A, Ap}

where

Y <<X>> = the certificate of user X issued by certification authority Y

Y {I} = the signing of I by Y. It consists of I with an encrypted hash code appended

The CA signs the certificate with its private key. If the corresponding public key is known to a user, then that user can verify that a certificate signed by the CA is valid.

Obtaining a User's Certificate

User certificates generated by a CA have the following characteristics:

- Any user with access to the public key of the CA can verify the user public key that was certified.
- No party other than the certification authority can modify the certificate without this being detected.

Because certificates are unforgeable, they can be placed in a directory without the need for the directory

to make special efforts to protect them.

If all users subscribe to the same CA, then there is a common trust of that CA. All user certificates can be placed in the directory for access by all users.

If there is a large community of users, it may not be practical for all users to subscribe to the same CA. Because it is the CA that signs certificates, each participating user must have a copy of the CA's own public key to verify signatures. This public key must be provided to each user in an absolutely secure (with respect to integrity and authenticity) way so that the user has confidence in the associated certificates. Thus, with many users, it may be more practical for there to be a number of CAs, each of which securely provides its public key to some fraction of the users.

Now suppose that A has obtained a certificate from certification authority X_1 and B has obtained a certificate from CA X_2. If A does not securely know the public key of X_2, then B's certificate, issued by X_2, is useless to A.

A can read B's certificate, but A cannot verify the signature. However, if the two CAs have securely exchanged their own public keys, the following procedure will enable A to obtain B's public key:

A has used a chain of certificates to obtain B's public key. In the notation of X.509, this chain is expressed as

$X_1\langle\langle X_2\rangle\rangle$ $X_2\langle\langle B\rangle\rangle$

In the same fashion, B can obtain A's public key with the reverse chain:

$X_2\langle\langle X_1\rangle\rangle$ $X_1\langle\langle A\rangle\rangle$

This scheme need not be limited to a chain of two certificates. An arbitrarily long path of CAs can be followed to produce a chain. A chain with N elements would be expressed as

$X_1\langle\langle X_2\rangle\rangle$ $X_2\langle\langle X_3\rangle\rangle \ldots X_N\langle\langle B\rangle\rangle$

In this case, each pair of CAs in the chain (X_i, X_{i+1}) must have created certificates for each other.

All these certificates of CAs by CAs need to appear in the directory, and the user needs to know how they are linked to follow a path to another user's public-key certificate. X.509 suggests that CAs be arranged in a hierarchy so that navigation is straightforward.

Figure 4.5, taken from X.509, is an example of such a hierarchy. The connected circles indicate the hierarchical relationship among the CAs; the associated boxes indicate certificates maintained in the directory for each CA entry. The directory entry for each CA includes two types of certificates:

- Forward certificates: Certificates of X generated by other CAs
- Reverse certificates: Certificates generated by X that are the certificates of other CAs

CA HIERARCHY USE

In the example given below , user A can acquire the following certificates from the directory to establish a certification path to B:

X<<W>> W <<V>> V <<Y>> <<Z>> Z <<B>>

When A has obtained these certificates, it can unwrap the certification path in sequence to recover a trusted copy of B's public key. Using this public key, A can send encrypted messages to B. If A wishes to receive encrypted messages back from B, or to sign messages sent to B, then B will require A's public key, which can be obtained from the following certification path:

Z<<Y>> Y <<V>> V <<W>> W <<X>>X <<A>>

B can obtain this set of certificates from the directory, or A can provide them as part of its initial message to B.

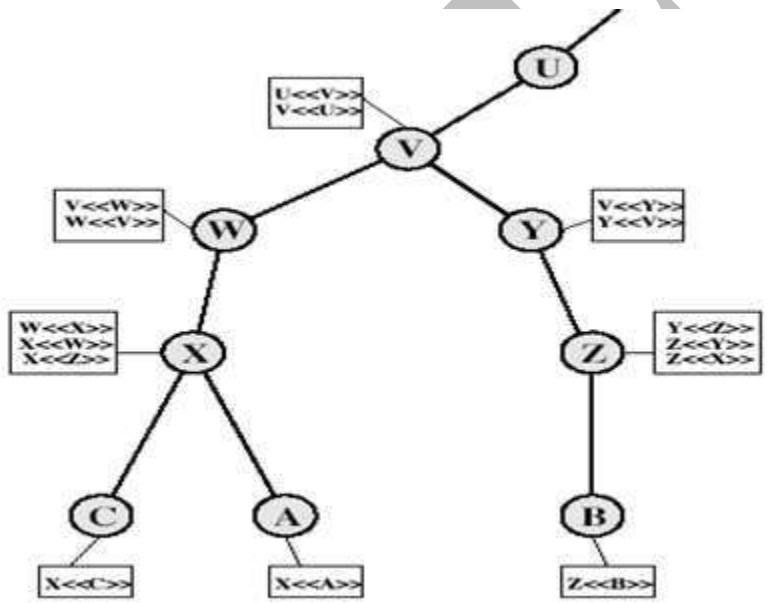


Figure 4.4 X.509 Hierarchy: A Hypothetical Example

CERTIFICATE REVOCATION

- certificates have a period of validity
- may need to revoke before expiry, for the following reasons

eg: 1. user's private key is compromised

2. user is no longer certified by this CA
3. CA's certificate is compromised

- CA's maintain list of revoked certificates

1. the Certificate Revocation List (CRL)

- users should check certs with CA's CRL

AUTHENTICATION PROCEDURES

X.509 includes three alternative authentication procedures:

- **One-Way Authentication**
- **Two-Way Authentication**
- **Three-Way Authentication**

All use public-key signatures

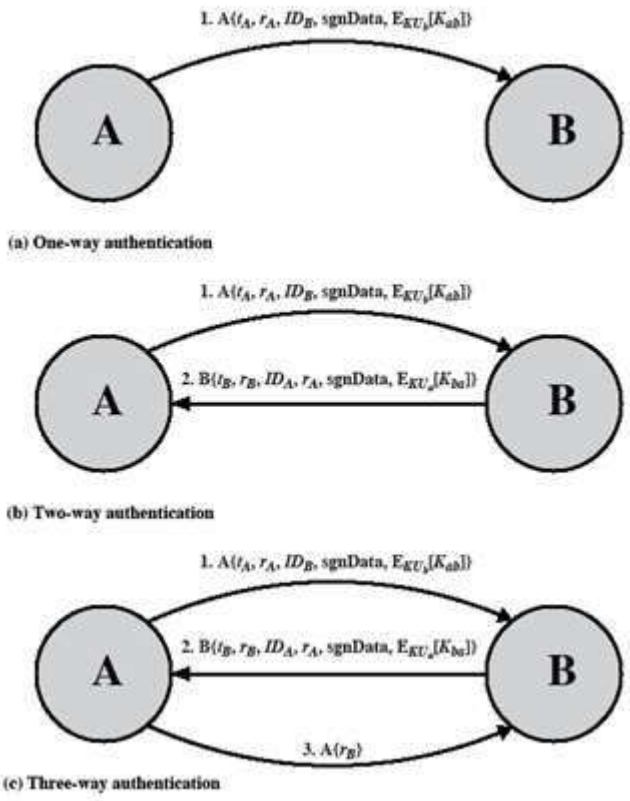


Figure 4.5: X.509 Strong Authentication Procedures

One-Way Authentication

•1 message (A->B) used to establish

– the identity of A and that message is from A

– message was intended for B

– integrity & originality of message

•message must include timestamp, nonce, B's identity and is signed by A

Two-Way Authentication

• 2 messages (A->B, B->A) which also establishes in addition:

– the identity of B and that reply is from B

– that reply is intended for A

– integrity & originality of reply

• reply includes original nonce from A, also timestamp and nonce from B

Three-Way Authentication

• 3 messages (A->B, B->A, A->B) which enables above authentication without synchronized clocks

• has reply from A back to B containing signed copy of nonce from B

• means that timestamps need not be checked or relied upon

X.509 VERSION 3

The X.509 version 2 format does not convey all of the information that recent design and implementation experience has shown to be needed. The following requirements not satisfied by version 2:

1. The Subject field is inadequate to convey the identity of a key owner to a public-key user.
2. The Subject field is also inadequate for many applications, which typically recognize entities by an Internet e-mail address, a URL, or some other Internet-related identification.
3. There is a need to indicate security policy information. There is a need to limit the damage that can result from a faulty or malicious CA by setting constraints on the applicability of a particular certificate.
4. It is important to be able to identify different keys used by the same owner at different times.

The certificate extensions fall into three main categories: key and policy information, subject and issuer attributes, and certification path constraints.

Key and Policy Information

These extensions convey additional information about the subject and issuer keys, plus indicators of certificate policy.. For example, a policy might be applicable to the authentication of electronic data interchange (EDI) transactions for the trading of goods within a given price range.

This area includes the following:

Authority key identifier: Identifies the public key to be used to verify the signature on this certificate or CRL.

Subject key identifier: Identifies the public key being certified. Useful for subject key pair updating.

Key usage: Indicates a restriction imposed as to the purposes for which, and the policies under which, the certified public key may be used.

Private-key usage period: Indicates the period of use of the private key corresponding to the public key.. For example, with digital signature keys, the usage period for the signing private key is typically shorter than that for the verifying public key.

Certificate policies: Certificates may be used in environments where multiple policies apply.

Policy mappings: Used only in certificates for CAs issued by other CAs.

Certificate Subject and Issuer Attributes

These extensions support alternative names, in alternative formats, for a certificate subject or certificate issuer and can convey additional information about the certificate subject, to increase a certificate user's confidence that the certificate subject is a particular person or entity. For example, information such as postal address, position within a corporation, or picture image may be required.

The extension fields in this area include the following:

Subject alternative name: Contains one or more alternative names, using any of a variety of forms

Subject directory attributes: Conveys any desired X.500 directory attribute values for the subject of this certificate.

Certification Path Constraints

These extensions allow constraint specifications to be included in certificates issued for CAs by other CAs.

The extension fields in this area include the following:

Basic constraints: Indicates if the subject may act as a CA. If so, a certification path length constraint may be specified.

Name constraints: Indicates a name space within which all subject names in subsequent certificates in a certification path must be located.

Policy constraints: Specifies constraints that may require explicit certificate policy identification or inhibit policy mapping for the remainder of the certification path.

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

COURSE CODE: 17CAP404N UNIT: III BATCH-2017-2020

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

COURSE CODE: 17CAP404N UNIT: III BATCH-2017-2020

ELECTRONIC MAIL SECURITY

PGP is an open-source freely available software package for e-mail security. It provides authentication through the use of digital signature; confidentiality through the use of symmetric block encryption; compression using the ZIP algorithm; e-mail compatibility using the radix-64 encoding scheme; and segmentation and reassembly to accommodate long e-mails.

- PGP incorporates tools for developing a public-key trust model and public-key certificate management.
- S/MIME is an Internet standard approach to e-mail security that incorporates the same functionality as PGP.

PRETTY GOOD PRIVACY (PGP)

PGP provides the confidentiality and authentication service that can be used for electronic mail and file storage applications. The steps involved in PGP are

elect the best available cryptographic algorithms as building blocks.

ntegrate these algorithms into a general purpose application that is independent of operating system and processor and that is based on a small set of easy-to-use commands.

Make the package and its documentation, including the source code, freely available via the internet, bulletin boards and commercial networks.

nter into an agreement with a company to provide a fully compatible, low cost commercial version of PGP.

PGP has grown explosively and is now widely used. A number of reasons can be cited for this growth.

is available free worldwide in versions that run on a variety of platform.

t is based on algorithms that have survived extensive public review and are considered extremely secure.

e.g., RSA, DSS and Diffie Hellman for public key encryption CAST-128, IDEA and 3DES for conventional encryption SHA-1 for hash coding.

has a wide range of applicability.

[redacted] was not developed by, nor it is controlled by, any governmental or standards organization.

Operational description

The actual operation of PGP consists of five services: authentication, confidentiality, compression, e-mail compatibility and segmentation.

1. Authentication

The sequence for authentication is as follows[redacted] The sender creates the message

[redacted]HA-1 is used to generate a 160-bit hash code of the message

[redacted]he hash code is encrypted with RSA using the sender's private key and the result is prepended to the message

[redacted]he receiver uses RSA with the sender's public key to decrypt and recover the hash code.

[redacted]he receiver generates a new hash code for the message and compares it with the decrypted hash code. If the two match, the message is accepted as authentic.

2. Confidentiality

Confidentiality is provided by encrypting messages to be transmitted or to be stored locally as files. In both cases, the conventional encryption algorithm CAST-128 may be used. The 64-bit cipher feedback (CFB) mode is used. In PGP, each conventional key is used only once. That is, a new key is generated as a random 128-bit number for each message. Thus although this is referred to as **a session key**, it is in reality a **one time key**. To protect the key, it is encrypted with the receiver's public key.

The sequence for confidentiality is as follows:

[redacted]he sender generates a message and a random 128-bit number to be used as a session key for this message only.

[redacted]he message is encrypted using CAST-128 with the session key.

[redacted]he session key is encrypted with RSA, using the receiver's public key and is prepended to the message.

[redacted]he receiver uses RSA with its private key to decrypt and recover the session key.

The session key is used to decrypt the message.

Confidentiality and authentication

Here both services may be used for the same message. First, a signature is generated for the plaintext message and prepended to the message. Then the plaintext plus the signature is encrypted using CAST-128 and the session key is encrypted using RSA.

3. Compression

As a default, PGP compresses the message after applying the signature but before encryption. This has the benefit of saving space for both e-mail transmission and for file storage.

The signature is generated before compression for two reasons:

It is preferable to sign an uncompressed message so that one can store only the uncompressed message together with the signature for future verification. If one signed a compressed document, then it would be

necessary either to store a compressed version of the message for later verification or to recompress the message when verification is required.

Even if one were willing to generate dynamically a recompressed message fro verification, PGP's compression algorithm presents a difficulty. The algorithm is not deterministic; various implementations of the algorithm achieve different tradeoffs in running speed versus compression ratio and as a result, produce different compression forms.

Message encryption is applied after compression to strengthen cryptographic security. Because the compressed message has less redundancy than the original plaintext, cryptanalysis is more difficult. The compression algorithm used is ZIP.

4. e-mail compatibility

Many electronic mail systems only permit the use of blocks consisting of ASCII texts. To accommodate this restriction, PGP provides the service of converting the raw 8-bit binary stream to a stream of printable ASCII characters. The scheme used for this purpose is radix-64 conversion. Each

group of three octets of binary data is mapped into four ASCII characters.

e.g., consider the 24-bit (3 octets) raw text sequence 00100011 01011100 10010001, we can express this input in block of 6-bits to produce 4 ASCII characters.

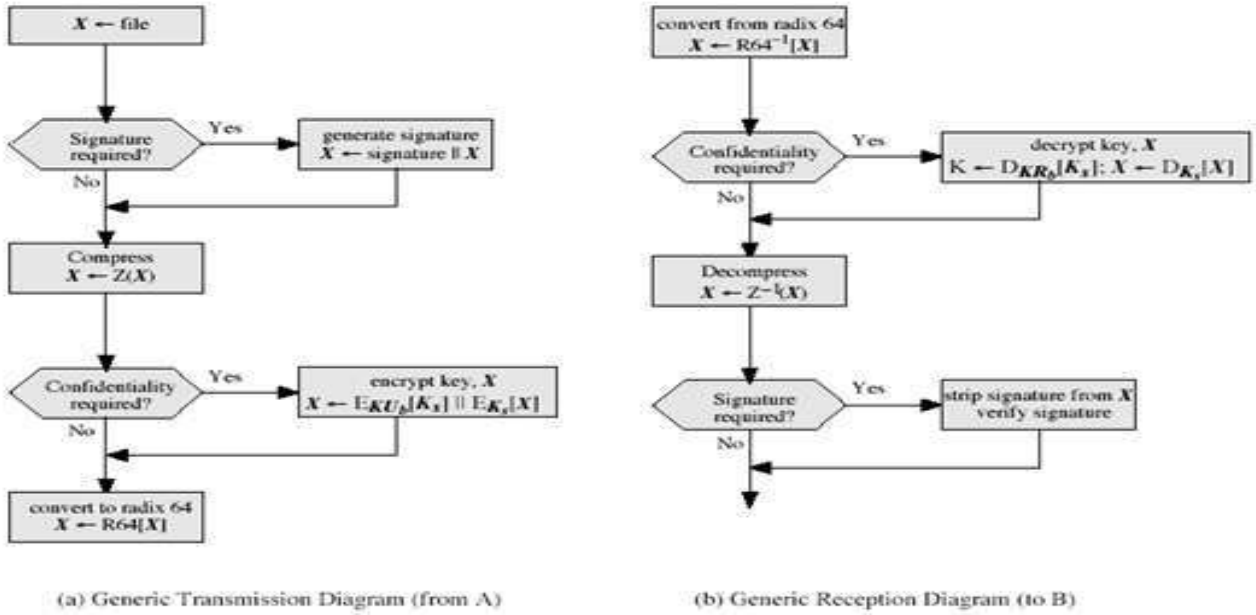
001000	110101	110010	010001	
I	L	Y	R	=> corresponding ASCII characters

5. Segmentation and reassembly

E-mail facilities often are restricted to a maximum length. E.g., many of the facilities accessible through the internet impose a maximum length of 50,000 octets. Any message longer than that must be broken up into smaller segments, each of which is mailed separately.

To accommodate this restriction, PGP automatically subdivides a message that is too large into segments that are small enough to send via e-mail. The segmentation is done after all the other processing, including the radix-64 conversion. At the receiving end, PGP must strip off all e-mail headers and reassemble the entire original block before performing the other steps.

PGP Operation Summary:



(a) Generic Transmission Diagram (from A) (b) Generic Reception Diagram (to B)

Figure 4. 7 Transmission and Reception of PGP Messages

Cryptographic keys and key rings

Three separate requirements can be identified with respect to these keys:

- A means of generating unpredictable session keys is needed.
- It must allow a user to have multiple public key/private key pairs.
- Each PGP entity must maintain a file of its own public/private key pairs as well as a file of public keys of correspondents.

We now examine each of the requirements in turn.

1. Session key generation

Each session key is associated with a single message and is used only for the purpose of encryption and decryption of that message. Random 128-bit numbers are generated using CAST-128 itself. The input to the random number generator consists of a 128-bit key and two 64-bit blocks that are treated as plaintext to be encrypted. Using cipher feedback mode, the CAST-128 produces two 64-bit cipher text blocks, which are concatenated to form the 128-bit session key. The

plaintext input to CAST-128 is itself derived from a stream of 128-bit randomized numbers. These numbers are based on the keystroke input from the user.

2. Key identifiers

If multiple public/private key pair are used, then how does the recipient know which of the public keys was used to encrypt the session key? One simple solution would be to transmit the public key with the message but, it is unnecessary wasteful of space. Another solution would be to associate an identifier with each public key that is unique at least within each user.

The solution adopted by PGP is to assign a key ID to each public key that is, with very high probability, unique within a user ID. The key ID associated with each public key consists of its least significant 64 bits. i.e., the key ID of public key KU_a is ($KU_a \bmod 2^{64}$).

A message consists of three components.

Message component – includes actual data to be transmitted, as well as the filename and a timestamp that specifies the time of creation.

ignature component – includes the following

o Timestamp – time at which the signature was made. o Message digest – hash code.

o Two octets of message digest – to enable the recipient to determine if the correct public key was used to decrypt the message.

o Key ID of sender's public key – identifies the public key

ession key component – includes session key and the identifier of the recipient public key.

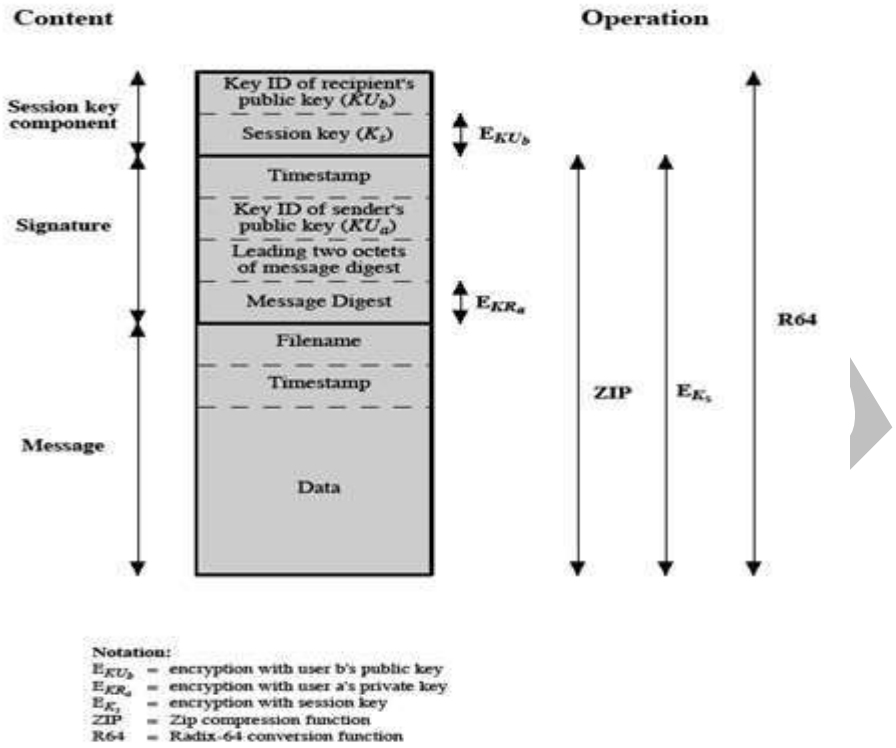


Figure 4.8: General Format of PGP Message (from A to B)

3. Key rings

PGP provides a pair of data structures at each node, one to store the public/private key pair owned by that node and one to store the public keys of the other users known at that node. These data structures are referred to as private key ring and public key ring.

The general structures of the private and public key rings are shown below:

Timestamp – the date/time when this entry was made.

Key ID – the least significant bits of the public key.

Public key – public key portion of the pair.

Private key – private key portion of the pair.

User ID – the owner of the key.

Key legitimacy field – indicates the extent to which PGP will trust that this is a valid public key for

this user.

Private Key Ring

Timestamp	Key ID*	Public Key	Encrypted Private Key	User ID*
• • •	• • •	• • •	• • •	• • •
T_i	PU_i mod 2^{64}	PU_i	E(H(P_i), PR_i)	User i
• • •	• • •	• • •	• • •	• • •

Public Key Ring

Timestamp	Key ID*	Public Key	Owner Trust	User ID*	Key Legitimacy	Signature(s)	Signature Trust(s)
• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •
T_i	PU_i mod 2^{64}	PU_i	trust_flag$_i$	User i	trust_flag$_i$		
• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •

* = field used to index table

Figure 4.9: General Structure of Private and Public Key Rings

Signature trust field – indicates the degree to which this PGP user trusts the signer to certify public key.

Owner trust field – indicates the degree to which this public key is trusted to sign other public key certificates.

PGP message generation

First consider message transmission and assume that the message is to be both signed and encrypted. The sending PGP entity performs the following steps:

1. signing the message

PGP retrieves the sender's private key from the private key ring using user ID as an index. If user ID was not provided, the first private key from the ring is retrieved.

PGP prompts the user for the passpharse (password) to recover the unencrypted private key.

- The signature component of the message is constructed.

2. encrypting the message

- GP generates a session key and encrypts the message.
- GP retrieves the recipient's public key from the public key ring using user ID as index.
- The session key component of the message is constructed.

The receiving PGP entity performs the following steps:

1. decrypting the message

- PGP retrieves the receiver's private key from the private key ring, using the key ID field in the session key component of the message as an index.
- PGP prompts the user for the passpharse (password) to recover the unencrypted private key.
- PGP then recovers the session key and decrypts the message.

2. Authenticating the message

- PGP retrieves the sender's public key from the public key ring, using the key ID field in the signature key component of the message as an index.
- PGP recovers the transmitted message digest.
- PGP computes the message digest for the received message and compares it to the transmitted message digest to authenticate.

Public-Key Management

This whole business of protecting public keys from tampering is the single most difficult problem in practical public key applications. PGP provides a structure for solving this problem, with several suggested options that may be used.

Approaches to Public-Key Management

The essence of the problem is this: User A must build up a public-key ring containing the public keys of other users to interoperate with them using PGP. Suppose that A's key ring contains a public key attributed to B but that the key is, in fact, owned by C. This could happen if, for example, A got the key from a bulletin board system (BBS) that was used by B to post the public key but that has been

compromised by C. The result is that two threats now exist. First, C can send messages to A and forge B's signature, so that A will accept the message as coming from B. Second, any encrypted message from A to B can be read by C.

A number of approaches are possible for minimizing the risk that a user's public-key ring contains false public keys. Suppose that A wishes to obtain a reliable public key for B. The following are some approaches that could be used:

1. Physically get the key from B. B could store her public key (PU_b) on a floppy disk and hand it to A..
2. Verify a key by telephone. If A can recognize B on the phone, A could call B and ask her to dictate the key, in radix-64 format, over the phone.
3. Obtain B's public key from a mutual trusted individual D. For this purpose, the introducer, D, creates a signed certificate. The certificate includes B's public key, the time of creation of the key, and a validity period for the key.
4. Obtain B's public key from a trusted certifying authority. Again, a public key certificate is created and signed by the authority. A could then access the authority, providing a user name and receiving a signed certificate.

For cases 3 and 4, A would already have to have a copy of the introducer's public key and trust that this key is valid. Ultimately, it is up to A to assign a level of trust to anyone who is to act as an introducer.

The Use of Trust

Although PGP does not include any specification for establishing certifying authorities or for establishing trust, it does provide a convenient means of using trust, associating trust with public keys, and exploiting trust information. The basic structure is as follows. Each entry in the public-key ring is a public-key certificate.

We can describe the operation of the trust processing as follows:

1. When A inserts a new public key on the public-key ring, PGP must assign a value to the trust flag that is associated with the owner of this public key. If the owner is A, and therefore this public key also appears in the private-key ring, then a value of ultimate trust is automatically assigned to the trust field. Otherwise, PGP asks A for his assessment of the trust to be assigned to the owner of this key, and A must enter the desired level. The user can specify that this owner is unknown, untrusted, marginally trusted, or completely trusted.
2. When the new public key is entered, one or more signatures may be attached to it. More signatures may be added later. When a signature is inserted into the entry, PGP searches the public-key ring to see if the author of this signature is among the known public-key owners. If so, the OWNERTRUST

value for this owner is assigned to the SIGTRUST field for this signature. If not, an unknown user value is assigned.

3. The value of the key legitimacy field is calculated on the basis of the signature trust fields present in this entry. If at least one signature has a signature trust value of ultimate, then the key legitimacy value is set to complete.

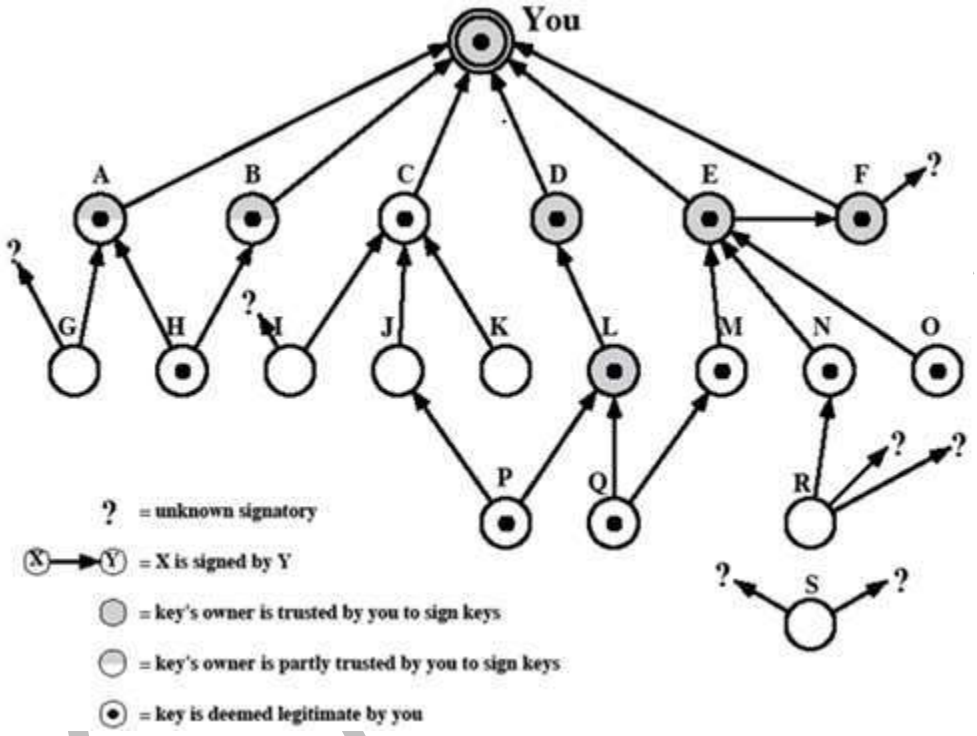


Figure 4.12: PGP Trust Model Example

The node labeled "You" refers to the entry in the public-key ring corresponding to this user. This key is legitimate and the OWNERTRUST value is ultimate trust. Each other node in the key ring has an OWNERTRUST value of undefined unless some other value is assigned by the user. In this example, this user has specified that it always trusts the following users to sign other keys: D, E, F, L. This user partially trusts users A and B to sign other keys.

So the shading, or lack thereof, of the nodes in Figure 4 .12 indicates the level of trust assigned by this user. The tree structure indicates which keys have been signed by which other users. If a key is signed by a user whose key is also in this key ring, the arrow joins the signed key to the signatory. If the key is signed by a user whose key is not present in this key ring, the arrow joins the signed key to a question mark, indicating that the signatory is unknown to this user.

S/MIME

S/MIME (Secure/Multipurpose Internet Mail Extension) is a security enhancement to the MIME Internet e-mail format standard, based on technology from RSA Data Security. S/MIME is defined in a number of documents, most importantly RFCs 3369, 3370, 3850 and 3851.

Multipurpose Internet Mail Extensions

MIME is an extension to the RFC 822 framework that is intended to address some of the problems and limitations of the use of SMTP (Simple Mail Transfer Protocol) or some other mail transfer protocol and RFC 822 for electronic mail. Following are the limitations of SMTP/822 scheme:

1. SMTP cannot transmit executable files or other binary objects.
2. SMTP cannot transmit text data that includes national language characters because these are represented by 8-bit codes with values of 128 decimal or higher, and SMTP is limited to 7-bit ASCII.
3. SMTP servers may reject mail message over a certain size.
4. SMTP gateways that translate between ASCII and the character code EBCDIC do not use a consistent set of mappings, resulting in translation problems.
5. SMTP gateways to X.400 electronic mail networks cannot handle nontextual data included in X.400 messages.
6. Some SMTP implementations do not adhere completely to the SMTP standards defined in RFC 821. Common problems include:

o Deletion, addition, or reordering of carriage return and linefeed o Truncating or wrapping lines longer than 76 characters

o Removal of trailing white space (tab and space characters) o Padding of lines in a message to the same length

o Conversion of tab characters into multiple space characters

MIME is intended to resolve these problems in a manner that is compatible with existing RFC 822 implementations. The specification is provided in RFCs 2045 through 2049.

Overview

The MIME specification includes the following elements:

1. **Five new message header** fields are defined, which may be included in an RFC 822 header. These fields provide information about the body of the message.
2. **A number of content formats** are defined, thus standardizing representations that support multimedia electronic mail.
3. **Transfer encodings** are defined that enable the conversion of any content format into a form that is protected from alteration by the mail system.

The five header fields defined in MIME are as follows:

MIME-Version: Must have the parameter value 1.0. This field indicates that the message conforms to RFCs 2045 and 2046.

Content-Type: Describes the data contained in the body with sufficient detail

Content-Transfer-Encoding: Indicates the type of transformation that has been used to represent the body of the message in a way that is acceptable for mail transport.

Content-ID: Used to identify MIME entities uniquely in multiple contexts.

Content-Description: A text description of the object with the body; this is useful when the object is not readable (e.g., audio data).

MIME Content Types

Table 4.2 lists the content types specified in RFC 2046. There are seven different major types of content and a total of 15 subtypes

Table 4.2. MIME Content Types

Type	Subtype	Description
Text	Plain	Unformatted text; may be ASCII or ISO 8859.
	Enriched	Provides greater format flexibility.

Multipart	Mixed	The different parts are independent but are to be transmitted together. They should be presented to the receiver in the order th they appear in the mail message.
	Parallel	Differs from Mixed only in that no order is defined for deliverin the parts to the receiver.
	Alternative	The different parts are alternative versions of the same information. They are ordered in increasing faithfulness to the original, and the recipient's mail system should display the "best version to the user.
	Digest	Similar to Mixed, but the default type/subtype of each part is message/rfc822.
Message	rfc822	The body is itself an encapsulated message that conforms to RF 822.
	Partial	Used to allow fragmentation of large mail items, in a way that is transparent to the recipient.
	External-bod	Contains a pointer to an object that exists elsewhere.
Image	jpeg	The image is in JPEG format, JFIF encoding.
	gif	The image is in GIF format.
Video	mpeg	MPEG format.
Audio	Basic	Single-channel 8-bit ISDN mu-law encoding at a sample rate of kHz.
Application	PostScript	Adobe Postscript.
	octet-stream	General binary data consisting of 8-bit bytes.

MIME Transfer Encodings

The other major component of the MIME specification, in addition to content type specification, is a definition of transfer encodings for message bodies. The objective is to provide reliable delivery across the largest range of environments.

The MIME standard defines two methods of encoding data. The Content-Transfer-Encoding field can actually take on six values, as listed in Table 4.3. For SMTP transfer, it is safe to use the 7bit

form. The 8bit and binary forms may be usable in other mail
transport contexts. Another Content-Transfer-Encoding value is x-token, which indicates that some other encoding scheme is used, for which a name is to be supplied. The two actual encoding schemes defined are quoted-printable and base64.

Table 4.3 MIME Transfer Encodings

7bit	The data are all represented by short lines of ASCII characters.
8bit	The lines are short, but there may be non-ASCII characters (octets with the high-order bit set).
binary	Not only may non-ASCII characters be present but the lines are not necessaril short enough for SMTP transport.
quoted-printable	Encodes the data in such a way that if the data being encoded are mostly ASC text, the encoded form of the data remains largely recognizable by humans.
base64	Encodes data by mapping 6-bit blocks of input to 8-bit blocks of output, all o which are printable ASCII characters.
x-token	A named nonstandard encoding.

Canonical Form

An important concept in MIME and S/MIME is that of canonical form. Canonical form is a format, appropriate to the content type, that is standardized for use between systems. This is in contrast to native form, which is a format that may be peculiar to a particular system.

S/MIME Functionality

In terms of general functionality, S/MIME is very similar to PGP. Both offer the ability to sign and/or encrypt messages. In this subsection, we briefly summarize S/MIME capability.

Functions

S/MIME provides the following functions:

nveloped data: This consists of encrypted content of any type and encrypted-content encryption keys for one or more recipients.

Signed data: A digital signature is formed by taking the message digest of the content to be signed and then encrypting that with the private key of the signer. The content plus signature are then encoded using base64 encoding. A signed data message can only be viewed by a recipient with S/MIME capability.

Clear-signed data: As with signed data, a digital signature of the content is formed. However, in this case, only the digital signature is encoded using base64. As a result, recipients without S/MIME capability can view the message content, although they cannot verify the signature.

Signed and enveloped data: Signed-only and encrypted-only entities may be nested, so that encrypted data may be signed and signed data or clear-signed data may be encrypted.

Cryptographic Algorithms

• hash functions: SHA-1 & MD5 • digital signatures: DSS & RSA
• session key encryption: ElGamal & RSA • message encryption: Triple-DES, RC2/40 and others • have a procedure to decide which algorithms to use.

Table 4.4 summarizes the cryptographic algorithms used in S/MIME. S/MIME uses the following terminology, taken from RFC 2119 to specify the requirement level:

Must: The definition is an absolute requirement of the specification. An implementation must include this feature or function to be in conformance with the specification.

Should: There may exist valid reasons in particular circumstances to ignore this feature or function, but it is recommended that an implementation include the feature or function.

Table 4.4. Cryptographic Algorithms Used in S/MIME

Function	Requirement

Create a message digest to be used in forming a digital signature. Encrypt message digest to form digit signature.	MUST support SHA-1. Receiver SHOULD support MD5 for backward compatibility. Sending and receiving agents MUST support DSS. Sending agents SHOULD support RSA encryption. Receiving agents SHOULD support verification of R$ signatures with key sizes 512 bits to 1024 bits.
Encrypt session key for transmission with message.	Sending and receiving agents SHOULD support Diffi Hellman. Sending and receiving agents MUST support RSA encryption with key sizes 512 bits to 1024 bits.
Encrypt message for transmission wit one-time session key.	Sending and receiving agents MUST support encrypti with triple DES Sending agents SHOULD support encryption with AE Sending agents SHOULD support encryption with RC2/40.
Create a message authentication code	Receiving agents MUST support HMAC with SHA-1 Receiving agents SHOULD support HMAC with SHA 1.

S/MIME Messages

S/MIME makes use of a number of new MIME content types, which are shown in Table 4.5. All of the new application types use the designation PKCS. This refers to a set of public-key cryptography specifications issued by RSA Laboratories and made available for the S/MIME effort.

Table 4.5. S/MIME Content Types

Type	Subtype	smime Parameter	Description
Multipart	Signed		A clear-signed message in two parts: one is the message and the other is the signature.
Application	pkcs 7-mime	signedData	A signed S/MIME entity.
	pkcs 7-mime	envelopedData	An encrypted S/MIME entity.
	pkcs 7-mime	degenerate signedDa	An entity containing only public- key certificates.
	pkcs 7-mime	CompressedData	A compressed S/MIME entity
	pkcs 7-signature	signedData	The content type of the signature subpart of multipart/signed message.

We examine each of these in turn after first looking at the general procedures for S/MIME message preparation.

SECURING A MIME ENTITY

S/MIME secures a MIME entity with a signature, encryption, or both. A MIME entity may be an entire message (except for the RFC 822 headers), or if the MIME content type is multipart, then a MIME entity is one or more of the subparts of the message. In all cases, the message to be sent is converted to canonical form. In particular, for a given type and subtype, the appropriate canonical form is used for the message content. For a multipart message, the appropriate canonical form is used for each subpart.

The use of transfer encoding requires special attention.

i) EnvelopedData

An application/pkcs7-mime subtype is used for one of four categories of S/MIME processing, each with a unique smime-type parameter. In all cases, the resulting entity, referred to as an object, is represented in a form known as Basic Encoding Rules (BER), which is defined in ITU-T Recommendation X.209. The steps for preparing an envelopedData MIME entity are as follows:

1. Generate a pseudorandom session key for a particular symmetric encryption algorithm (RC2/40 or tripleDES).

2. For each recipient, encrypt the session key with the recipient's public RSA key.

3. For each recipient, prepare a block known as RecipientInfo that contains an identifier of the recipient's public-key certificate, an identifier of the algorithm used to encrypt the session key, and the encrypted session key.

This is an X.509 certificate, discussed later in this section.

4. Encrypt the message content with the session key.

The RecipientInfo blocks followed by the encrypted content constitute the envelopedData. This information is then encoded into base64. To recover the encrypted message, the recipient first strips off the base64 encoding. Then the recipient's private key is used to recover the session key. Finally, the message content is decrypted with the session key.

ii) SignedData

The signedData smime-type can actually be used with one or more signers. For clarity, we confine our description to the case of a single digital signature. The steps for preparing a signedData MIME entity are as follows:

1. Select a message digest algorithm (SHA or MD5).

2. Compute the message digest, or hash function, of the content to be signed.

3. Encrypt the message digest with the signer's private key.

4. Prepare a block known as SignerInfo that contains the signer's public-key certificate, an identifier of the message digest algorithm, an identifier of the algorithm used to encrypt the message digest, and the encrypted message digest.

The signedData entity consists of a series of blocks, including a message digest algorithm identifier, the message being signed, and SignerInfo. The signedData entity may also include a set of public-key certificates sufficient to constitute a chain from a recognized root or top-level certification authority to the signer. This information is then encoded into base64.

To recover the signed message and verify the signature, the recipient first strips off the base64 encoding. Then the signer's public key is used to decrypt the message digest. The recipient independently computes the message digest and compares it to the decrypted message digest to verify the signature.

iii) Clear Signing

[redacted]ar signing is achieved using the multipart content type with a signed subtype.

[redacted] was mentioned, this signing process does not involve transforming the message to be signed, so that the message is sent "in the clear."

[redacted]s, recipients with MIME capability but not S/MIME capability are able to read the incoming message.

A multipart/signed message has two parts. The first part can be any MIME type but must be prepared so that it will not be altered during transfer from source to destination. This means that if the first part is not 7bit, then it needs to be encoded using base64 or quoted-printable. Then this part is processed in the same manner as signedData, but in this case an object with signedData format is created that has an empty message content field. This object is a detached signature. It is then transfer encoded using base64 to become the second part of the multipart/signed message. This second part has a MIME content type of application and a subtype of pkcs7-signature

The protocol parameter indicates that this is a two-part clear-signed entity. The receiver can verify the signature by taking the message digest of the first part and comparing this to the message digest recovered from the signature in the second part.

Registration Request

[redacted]ically, an application or user will apply to a certification authority for a public-key certificate.

[redacted] application/pkcs10 S/MIME entity is used to transfer a certification request. The certification request includes certificationRequestInfo block, followed by an identifier of the public-key encryption algorithm, followed by the signature of the certificationRequestInfo block, made using the sender's private key.

[redacted] certificationRequestInfo block includes a name of the certificate subject (the entity whose public key is to be certified) and a bit-string representation of the user's public key.

Certificates-Only Message

A message containing only certificates or a certificate revocation list (CRL) can be sent in response to a registration request. The message is an application/pkcs7-mime type/subtype with an smime-type parameter of degenerate. The steps involved are the same as those for creating a signedData message, except that there is no message content and the signerInfo field is empty.

S/MIME Certificate Processing

S/MIME uses public-key certificates that conform to version 3 of X.509 The key-management scheme used by S/MIME is in some ways a hybrid between a strict X.509 certification hierarchy and PGP's web of trust. As with the PGP model, S/MIME managers and/or users must configure each client with a list of trusted keys and with certificate revocation lists.

****User Agent Role***

An S/MIME user has **several key-management functions** to perform:

Key generation: The user of some related administrative utility (e.g., one associated with LAN management) MUST be capable of generating a key pair from a good source of nondeterministic random input and be protected in a secure fashion. A user agent SHOULD generate RSA key pairs with a length in the range of 768 to 1024 bits and MUST NOT generate a length of less than 512 bits.

Registration: A user's public key must be registered with a certification authority in order to receive an X.509 public-key certificate.

Certificate storage and retrieval: A user requires access to a local list of certificates in order to verify incoming signatures and to encrypt outgoing messages. Such a list could be maintained by the user or by some local administrative entity on behalf of a number of users.

****VeriSign Certificates***

There are several companies that provide certification authority (CA) services. For example, Nortel has designed an enterprise CA solution and can provide S/MIME support within an organization. There are a number of Internet-based CAs, including VeriSign, GTE, and the U.S. Postal Service. Of these, the most widely used is the VeriSign CA service, a brief description of which we now provide.

The information contained in a Digital ID depends on the type of Digital ID and its use. At a minimum, each Digital ID contains

- Owner's public key
- Owner's name or alias
- Expiration date of the Digital ID
- Serial number of the Digital ID

- Name of the certification authority that issued the Digital ID
- Digital signature of the certification authority that issued the Digital ID

Digital IDs can also contain other user-supplied information, including

- Address
- E-mail address
- Basic registration information (country, zip code, age, and gender)

VeriSign provides three levels, or classes, of security for public-key certificates. A user requests a certificate online at VeriSign's Web site or other participating Web sites. Class 1 and Class 2 requests are processed on line, and in most cases take only a few seconds to approve. Briefly, the following procedures are used:

- For Class 1 Digital IDs, VeriSign confirms the user's e-mail address by sending a PIN and Digital ID pick-up information to the e-mail address provided in the application.
- For Class 2 Digital IDs, VeriSign verifies the information in the application through an automated comparison with a consumer database in addition to performing all of the checking associated with a Class 1 Digital ID. Finally, confirmation is sent to the specified postal address alerting the user that a Digital ID has been issued in his or her name.
- For Class 3 Digital IDs, VeriSign requires a higher level of identity assurance. An individual must prove his or her identity by providing notarized credentials or applying in person.

Class	Identity Checks	Usage
1	name/email check	web browsing/email
2	+ enroll/addr check	email, subs, s/w validate
3	+ ID documents	e-banking/service access

Enhanced Security Services

As of this writing, three enhanced security services have been proposed in an Internet draft.:

- **Signed receipts:** A signed receipt may be requested in a SignedData object. Returning a signed receipt provides proof of delivery to the originator of a message and allows the originator to demonstrate to a third party that the recipient received the message.

Security labels: A security label may be included in the authenticated attributes of a SignedData object. A security label is a set of security information regarding the sensitivity of the content that is protected by S/MIME encapsulation. The labels may be used for access control, by indicating which users are permitte access to an object.

Secure mailing lists: When a user sends a message to multiple recipients, a certain amount of per-recipient processing is required, including the use of each recipient's public key. The user can be relieved of this work by employing the services of an S/MIME Mail List Agent (MLA). An MLA can take a single incoming message, perform the recipient-specific encryption for each recipient, and forward the message. The originator of a message need only send the message to the MLA, with encryption performed using the MLA's public key.

– IP Security

IP security (IPSec) is a capability that can be added to either current version of the Internet Protocol (IPv4 or IPv6), by means of additional headers.

- IPSec encompasses three functional areas: authentication, confidentiality, and key management.
- Authentication makes use of the HMAC message authentication code. Authentication can be applied to the entire original IP packet (tunnel mode) or to all of the packet except for the IP header (transport mode).
- Confidentiality is provided by an encryption format known as encapsulating security payload. Both tunnel and transport modes can be accommodated.
- IPSec defines a number of techniques for key management.

To provide security, the IAB (Internet Architecture Board) included authentication and encryption as necessary security features in the next-generation IP, which has been issued as IPv6. Fortunately, these security capabilities were designed to be usable both with the current IPv4 and the future IPv6. This means that vendors can begin offering these features now, and many vendors now do have some IPsec capability in their products. The IPsec specification now exists as a set of Internet standards.

Applications of IP security

IPsec provides the capability to secure communications across a LAN, across private and public WANs, and across the Internet.

Examples of its use include the following:

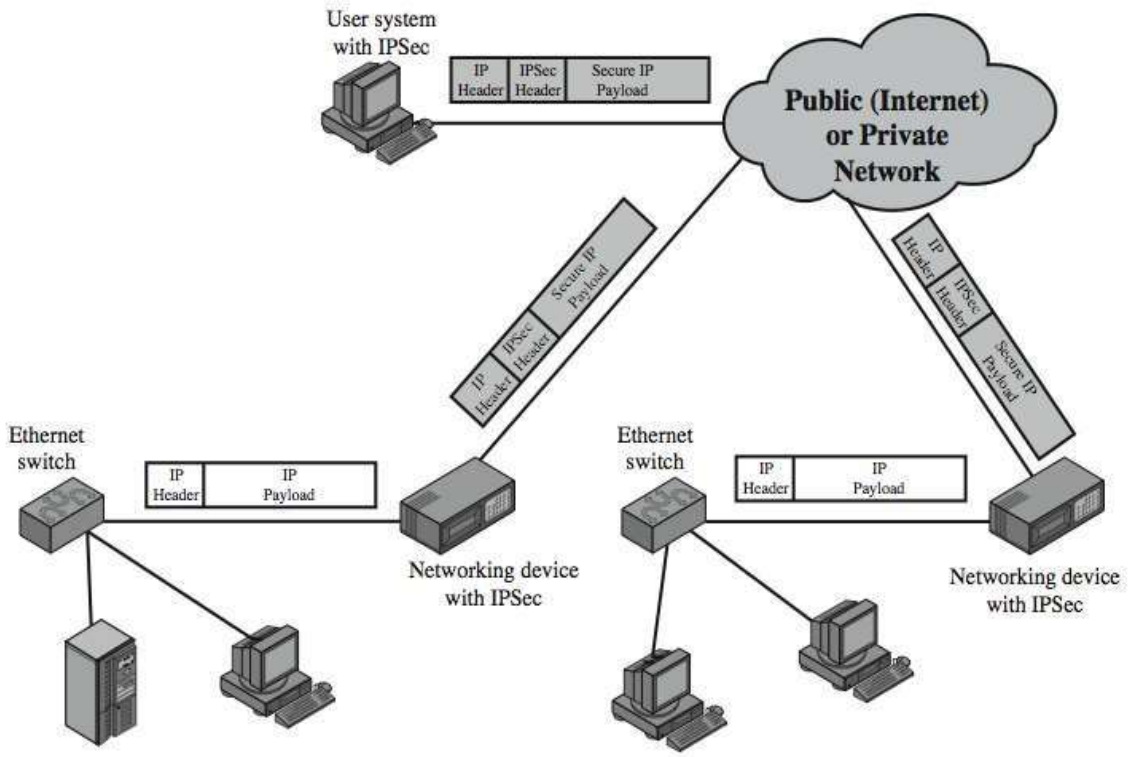
- **Secure branch office connectivity over the Internet:**
- **Secure remote access over the Internet:**
- **Establishing extranet and intranet connectivity with partners:**
- **Enhancing electronic commerce security**

The principal feature of IPsec that enables it to support these varied applications is that it can encrypt and/or authenticate *all* traffic at the IP level. Thus, all distributed applications (including remote logon, client/server, e-mail, file transfer, Web access, and so on) can be secured.

IP Security

- have a range of application specific security mechanisms
  - eg. S/MIME, PGP, Kerberos, SSL/HTTPS
- however there are security concerns that cut across protocol layers
- would like security implemented by the network for all applications
-
- general IP Security mechanisms
- provides
  - authentication
  - confidentiality
  - key management
- applicable to use over LANs, across public & private WANs, & for the Internet
- need identified in 1994 report
  - need authentication, encryption in IPv4 & IPv6
-

IP Security Scenario



➢

IP security services

IPsec provides security services at the IP layer by enabling a system to select required security Protocols.

Two protocols are used to provide security: an authentication protocol designated by the header of the protocol, Authentication Header (AH);
and a combined encryption/ authentication protocol designated by the format of the packet for that protocol, Encapsulating Security Payload (ESP). RFC 4301 lists the following services:

1. Access control
2. Connectionless integrity
3. Data origin authentication
4. Rejection of replayed packets (a form of partial sequence integrity)
5. Confidentiality (encryption)
6. Limited traffic flow confidentiality

Benefits of IPSec

- in a firewall/router provides strong security to all traffic crossing the perimeter

- in a firewall/router is resistant to bypass
- is below transport layer, hence transparent to applications
- can be transparent to end users
- can provide security for individual users
- secures routing architecture

IPSec Services

- Access control
- Connectionless integrity
- Data origin authentication
- Rejection of replayed packets
  - a form of partial sequence integrity
- Confidentiality (encryption)
- Limited traffic flow confidentiality

IP Security Architecture

- specification is quite complex, with groups:
  - Architecture
    - RFC4301 *Security Architecture for Internet Protocol*
  - Authentication Header (AH)
    - RFC4302 *IP Authentication Header*
  - Encapsulating Security Payload (ESP)
    - RFC4303 *IP Encapsulating Security Payload (ESP)*
  - Internet Key Exchange (IKE)
    - RFC4306 *Internet Key Exchange (IKEv2) Protocol*
  - Cryptographic algorithms
  - Other

Security Associations

A key concept that appears in both the authentication and confidentiality mechanisms for IP is the security association (SA).

An association is a one-way relationship between a sender and a receiver that affords security services to the traffic carried on it.

A security association is uniquely identified by three parameters:

Security Parameters Index (SPI) - The SPI is carried in AH and ESP headers to enable the receiving system to select the SA under which a received packet will be processed

IP Destination Address - this is the address of the destination endpoint of the SA,

Security Protocol Identifier - This indicates whether the association is an AH or ESP security association.

- SA parameters
  - seq no counter, AH info (Authentication algorithm, keys, key lifetimes) & EH (Encryption and authentication algorithm, keys,), lifetime etc.
-
-
-

- **Benefits of IPSec**

- in a firewall/router provides strong security to all traffic crossing the perimeter
- in a firewall/router is resistant to bypass
- is below transport layer, hence transparent to applications
- can be transparent to end users
- can provide security for individual users
- secures routing architecture
-

- **IP Security Architecture**

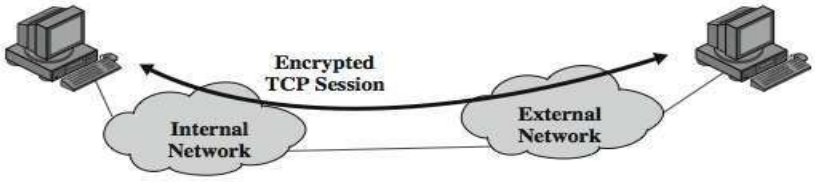
- specification is quite complex, with groups:
  - Architecture
    - RFC4301 *Security Architecture for Internet Protocol*
  - Authentication Header (AH)
    - RFC4302 *IP Authentication Header*
  - Encapsulating Security Payload (ESP)
    - RFC4303 *IP Encapsulating Security Payload (ESP)*
  - Internet Key Exchange (IKE)
    - RFC4306 *Internet Key Exchange (IKEv2) Protocol*
  - Cryptographic algorithms

- Other

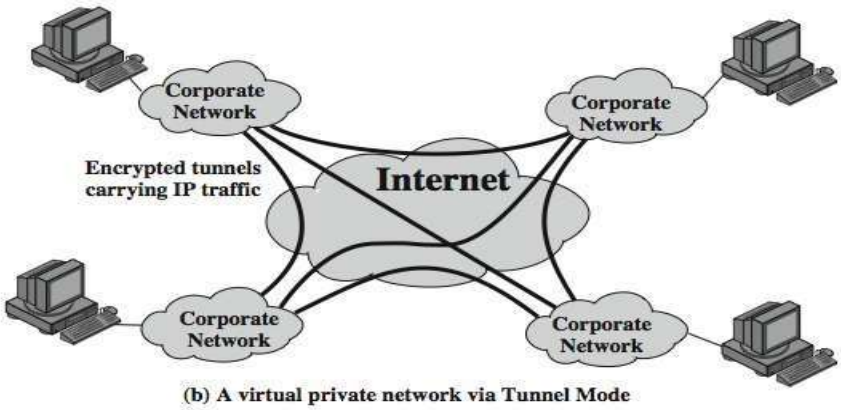
➢ Transport and Tunnel Modes

- Transport Mode
  - to encrypt & optionally authenticate IP data
  - can do traffic analysis but is efficient
  - good for ESP host to host traffic
- Tunnel Mode
  - encrypts entire IP packet
  - add new header for next hop
  - no routers on way can examine inner IP header
  - good for VPNs, gateway to gateway security
-

- **Transport and Tunnel Modes**

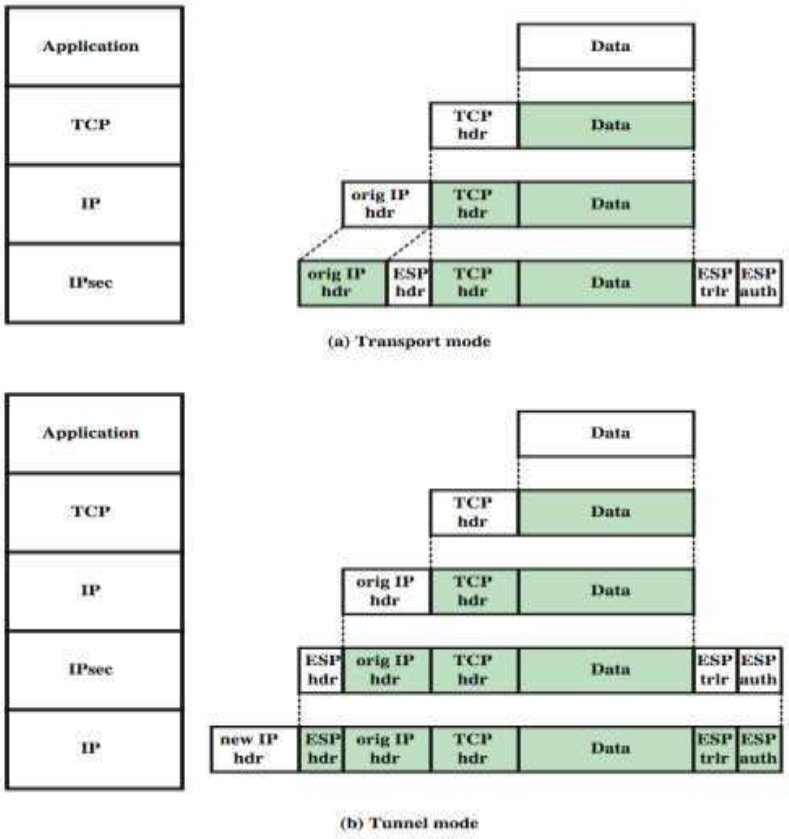


(a) Transport-level security



(b) A virtual private network via Tunnel Mode

- **Transport and Tunnel Mode Protocols**



➢

Transport Mode

Transport mode provides protection primarily for upper-layer protocols.

That is, transport mode protection extends to the payload of an IP packet.

Tunnel Mode

Tunnel mode provides protection to the entire IP packet.

To achieve this, after the AH or ESP fields are added to the IP packet, the entire packet plus security fields is treated as the payload of new "outer" IP packet with a new outer IP header.

The entire original, or inner, packet travels through a "tunnel" from one point of an IP network to another; no routers along the way are able to examine the inner IP header.

Because the original packet is encapsulated, the new, larger packet may have totally different source and destination addresses, adding to the security.

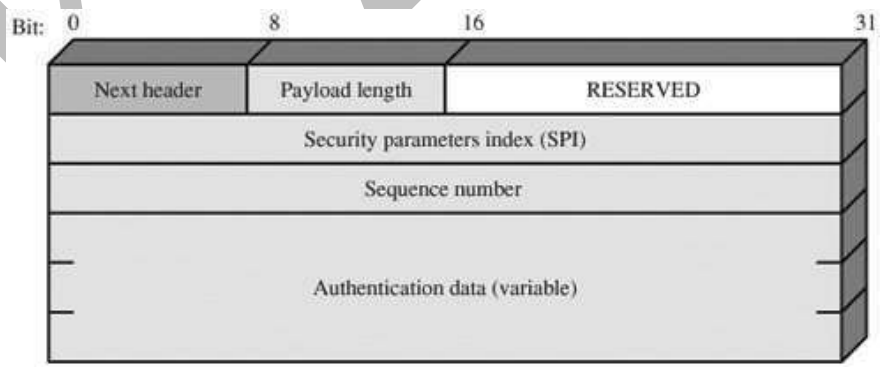
Tunnel mode is used when one or both ends of an SA are a security gateway, such as a firewall or router that implements IPSec.

Authentication Header

The Authentication Header provides support for data integrity and authentication of IP packets. The data integrity feature ensures that undetected modification to a packet's content in transit is not possible. The authentication feature enables an end system or network device to authenticate the user or application and filter traffic accordingly; it also prevents the address spoofing attacks observed in today's Internet. The AH also guards against the replay attack described later in this section. Authentication is based on the use of a message authentication code (MAC) hence the two parties must share a secret key.

The Authentication Header consists of the following fields :

- **Next Header (8 bits):** Identifies the type of header immediately following this header.
- **Payload Length (8 bits):** Length of Authentication Header in 32-bit words, minus 2. For example, the default length of the authentication data field is 96 bits, or three 32-bit words. With a three-word fixed header, there are a total of six words in the header, and the Payload Length field has a value of 4.
- **Reserved (16 bits):** For future use.
- **Security Parameters Index (32 bits):** Identifies a security association.
- **Sequence Number (32 bits):** A monotonically increasing counter value, discussed later.
- **Authentication Data (variable):** A variable-length field (must be an integral number of 32-bit words) that contains the Integrity Check Value (ICV), or MAC, for this packet, discussed later.



IPSec Authentication Header

Encapsulating Security Payload

The Encapsulating Security Payload provides confidentiality services, including confidentiality of message contents and limited traffic flow confidentiality. As an optional feature, ESP can also provide an authentication service.

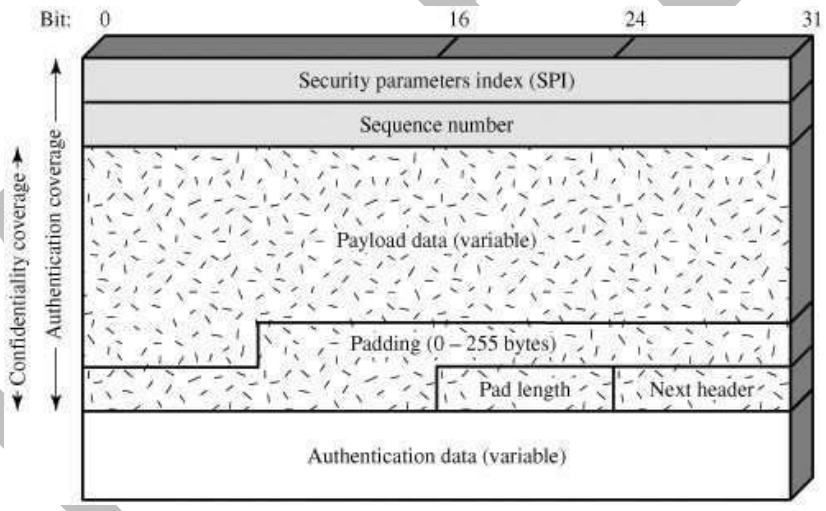
ESP Format

Figure shows the format of an ESP packet. It contains the following fields:

● **Security Parameters Index (32 bits):** Identifies a security association.

● **Sequence Number (32 bits):** A monotonically increasing counter value; this provides an antireplay function, as discussed for AH.

● **Payload Data (variable):** This is a transport-level segment (transport mode) or IP packet (tunnel mode) that is protected by encryption.



IPSec ESP format

ESP uses padding

- to expand plaintext to required length
- to align pad length and next header fields
- to provide partial traffic flow confidentiality

- Anti-Replay Service

- replay is when attacker resends a copy of an authenticated packet
- use sequence number to thwart this attack
-

Key Management

- The key management portion of IPSec involves the determination and distribution of secret keys. A typical requirement is four keys for communication between two applications: transmit and receive pairs
- for both AH and ESP. The IPSec Architecture document mandates support for two types of key management:
- ● **Manual:** A system administrator manually configures each system with its own keys and with the keys of other communicating systems. This is practical for small, relatively static environments.
- ● **Automated:** An automated system enables the on-demand creation of keys for SAs and facilitates the use of keys in a large distributed system with an evolving configuration.
- The default automated key management protocol for IPSec is referred to as ISAKMP/Oakley and consists of the following elements:
- ● **Oakley Key Determination Protocol**: Oakley is a key exchange protocol based on the Diffie-Hellman algorithm but providing added security. Oakley is generic in that it does not dictate specific formats.
- ● **Internet Security Association and Key Management Protocol (ISAKMP):** ISAKMP provides a framework for Internet key management and provides the specific protocol support, including formats, for negotiation of security attributes.

Oakley

- a key exchange protocol
- based on Diffie-Hellman key exchange
- adds features to address weaknesses
  - no info on parties, man-in-middle attack, cost
  - so adds cookies, groups (global params), nonces, DH key exchange with authentication
- can use arithmetic in prime fields or elliptic curve fields

ISAKMP

- Internet Security Association and Key Management Protocol
- provides framework for key management
- defines procedures and packet formats to establish, negotiate, modify, & delete SAs

- independent of key exchange protocol, encryption alg, & authentication method
- IKEv2 no longer uses Oakley & ISAKMP terms, but basic functionality is same.

UNIT III

POSSIBLE QUESTIONS

1. Explain in detail about Kerberos.
2. Explain in detail about PGP with diagram.
3. Explain in detail about X.509 Authentication Services
4. Explain in detail about S/MIME.
5. Explain in detail about Encryption Techniques
6. Explain the overview of IP security Architecture with a neat diagram.
7. Explain in detail about Pretty Good Privacy
8. Explain in detail about E-mail security
9. Explain in detail about One-Way Authentication and Two-Way Authentication
10. Explain the overview of Secure/Multipurpose Internet Mail Extension
11. Convertthe cipher text back to plaintext "WE ARE INDIAN" using
    (i) Ceasar cipher with the key: 3
    (ii) Polyalphabetic cipher with the key : deceptive

Transposition techniques with the key : 4312567

KARPAGAM ACADEMY OF HIGHER EDUCATION
DEPARTMENT OF COMPUTER APPLICATIONS
II MCA
CRYPTOGRAPHY AND NETWORK SECURITY

UNIT III

S.NO	Question	Option 1	Option 2	Option 3	Option 4	Answer
1	Two major approaches to encryption placement _____ and _____	link and end to end	data	link	information	link and end to end
2	_____ are transmitted in the form of frames.	data	information	link	none	data
3	The _____ does not generate electromagnetic emanations and hence is not vulnerable to inductive taps.	fiber	feast	data	link	fiber
4	The most powerful and most common approach to secuting the points of vulnerability highlighted in the preceding section is _____	data	decryption	encryption	link	encryption
5	_____ ,the encryption function is performed at a low level of the communications hierarchy.	link encryption	end to end	decryption	pre	link encryption
6	_____ model ,link encryption occurs at either the physical or link layers.	OSA	OSI	OSII	OSAI	OSI
7	_____,the encryption process is carried out at the two end systems.	data	link	end to end	information	end to end
8	the _____ accepts packets	FEA	FEP	FEK	FEPP	FEP
9	FEP stands for	front email processor	front end processor	form end processor	front end performing	front end processor
10	another concern related to traffic is the user of traffic patterns to create a _____	covert channel	distribution	link chennal	data	covert channel
11	_____ a term refers to the means of delevering a key to two parties who wish to exchance data,without allowing others to see the key.	form feed	key exchange	data processing	key distribution techniques	key distribution techniques
12	_____ used in electronic funds transfer and point of sale application.	PAN	file encryption	data	PIN	PIN
13	_____ key ,for encrypting files stored in publicly accessible locations.	file encryption	data encryption	PIN	file decryption	file encryption
14	_____ key,for general communication across a network.	Pin	file encryption	data encryption	file decryption	data encryption
15	A number of network security algorithms based on cryptography make use of .	random numbers	prime numbers	even numbers	odd numbers	random numbers
16	PRNGs stands for	Pesudorandom Number Generators	Prime Number Generators	Pesudorandom Null Generators	PersonalNumber Generators	Pesudorandom Number Generators
17	BBS	Blue Blue Shub Generator	Blum Blum Sub Generator	Blum Blum Shub Generator	Blue Blum Sub Generator	Blum Blum Shub Generator
18	a popular approach to generating secure pesudorandom number is known as the	BBS	BBA	BBC	BBD	BBS
19	CSPRBG	cylinder Secure Pesudorandom bit Generator	cryptographically Secure Pesudorandom bit Generator	cylinder Secure Primebit Generator	crypto Secure Prime bit Generator	cryptographically Secure Pesudorandom bit Generator
20	A _____ number generator uses a nondeterministic source to produce randomness.	true random	false random	odd number	even number	true random
21	Asymmetric encryption transforms _____ into ciphertext using a one of two keys and an encryption algorithms.	cipher text	plaintext	covert	images	plaintext
22	the most widely used public key cryptosystem is ____	RSD	RAB	RSC	RSA	RSA
23	_____ is the readable message or data that is fed into the algorithmas input	plaintext	cipher text	images	chennal	plaintext
24	the _____ algorithm performs various transformation on the plaintext.	images	decryption	encryption	chennal	encryption
25	_____ and _____ keys that have been selected so that if one is used for encryption , the other is used for decryption	public&protective	protected&private	encryption and decryption	public&private	public&private
26	_____ is the scrambled message produced as output	images	plaintext	ciphertext	chennal	ciphertext
27	_____algorithm accepts the ciphertext and the matching key and produces the original plaintext	decryption	encryption	plaintext	images	decryption
28	The sender_____ a message with the secipients public key	encrypt	decrypt	plaintext	images	encrypt
29	the sender signs a message with its private key is known as	covert writing	analog signature	digital signature	type writter	digital signature
30	A_____ function is one that maps a domain into a range such that every function value has a unique inverse	one-way	multiway	twoway	three way	one-way
31	plaintext is encrypted in blocks,with each block having a binary value less than some number n.the block size must be less than or equal to	log2(n)	logn	log n*n	logn2	log2(n)
32	_____ involves trying all possible kays	poly force	hill force	brust force	none	brust force
33	_____ depends on the running time of the decryption algorithm	one pad	timing attack	signature	force	timing attack
34	_____types of attack exploits properties of the RSA algorithm	chosen plain text	encryption	chosen ciphertext attack	decrytion	chosen ciphertext attack
35	_____ are several approaches,all equivalent in effort to factoring the products of two primes	timing attack	force attack	burst force	mathematical attacks	mathematical attacks
36	SNFS stands for	spell number fiels sieve	special number fiels sieve	special null fiels sieve	special number fiels store	special number fiels sieve
37	GNFS stands for	generalized number field store	generalized null field sieve	generalized number field sieve	great number field sieve	generalized number field sieve
38	_____multiply the ciphertext by a random number before performing exponentiation	blinding	random delay	constant	none	blinding
39	Better performance could be achieved by adding a _____ to the exponentiation algorithm to confuse the timing attack.	random delay	blinding	constant	none	random delay
40	a simple public key algorithm is _____ key exchange	brust force	hellman	diffie-hellman	none	diffie-hellman
41	ECC stands for	Elliptic Curve Cryptography	elliptic corner cryptography	End Curve Cryptography	Elliptic Cursor Cryptography	Elliptic Curve Cryptography
42	The _____ algorithm depends fos its effectiveness on the difficulty of computing discrete logarithmss	hellman	diffie-hellman	brust force	none	diffie-hellman
43	The protocol depicted insecure against an adversary who can intercept messages and then either relay the intercepted message is known as	man in the middle	man in the front	man in the end	none	man in the middle
44	_____,the hybrid schema is easily overlaid on an existing KDC schema,with minimal disruption or software changes	fordward compatibility	middle compatiblity	backward compatibility	front compatibility	backward compatibility

CLASS: II MCA **COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY**

COURSE CODE: 17CAP404N **UNIT: IV** **BATCH-2017-2020**

SYLLABUS

Web Security – Secure Socket Layer – Secure Electronic Transaction; System Security – Intruders and Viruses – Firewalls – Password Security.

WEB SECURITY

SECURE SOCKETS LAYER

SSL (Secure Sockets Layer) is a standard security technology for establishing an encrypted link between a server and a client—typically a web server (website) and a browser; or a mail server and a mail client (e.g., Outlook).SSL allows sensitive information such as credit card numbers, social security numbers, and login credentials to be transmitted securely. Normally, data sent between browsers and web servers is sent in plain text—leaving you vulnerable to eavesdropping. If an attacker is able to intercept all data being sent between a browser and a web server they can see and use that information.More specifically, SSL is a security protocol. Protocols describe how algorithms should be used; in this case, the SSL protocol determines variables of the encryption for both the link and the data being transmitted.

Overview

The Secure Sockets Layer (SSL) is a method for providing security for web based applications. It is designed to make use of TCP to provide a reliable end-to-end secure service.SSL is not a single protocol but rather two layers of protocols. It can be seen that one layer makes use of TCP directly. This layer is known as the SSL Record Protocol and it provides basic security services to various higher layer protocols.
An independent protocol that makes use of the record protocol is the Hypertext Markup Language (HTTP) protocol. Another three higher level protocols that also make use of this layer are part of the SSL stack.

SSL Architecture

SSL is designed to make use of TCP to provide a reliable end-to-end secure service. SSL is not a single protocol but rather two layers of protocols. The SSL Record Protocol provides basic security services to various higher layer protocols. In particular, the Hypertext Transfer Protocol (HTTP), which providesthe transfer service for Web client/server interaction, can operate on top of ssl.

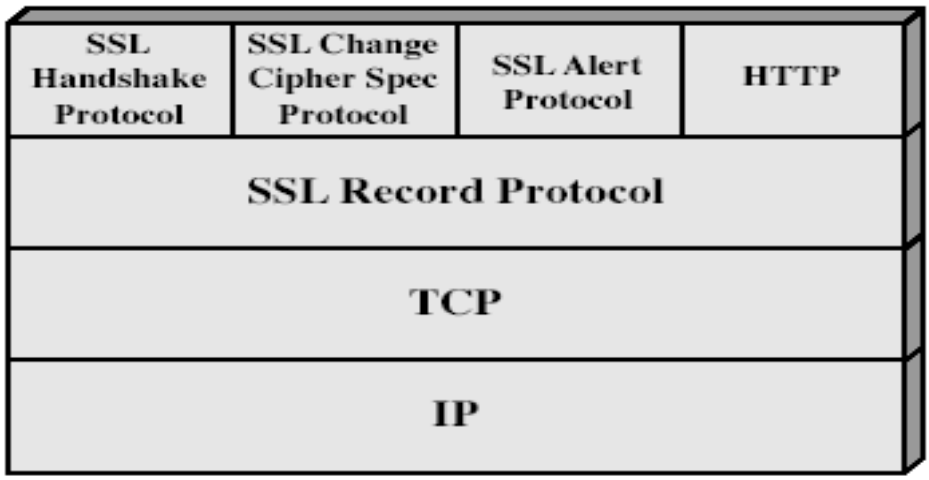
SSL. Three higher-layer protocols are defined as part of SSL:

1. Handshake Protocol.
2. Change Cipher Spec Protocol.
3. Alert Protocol

Two important SSL concepts are the SSL session and the SSL connection,
Which are defined in the specification as follows

Connection: A connection is a transport (in the OSI layering model definition)That provides a suitable type of service. For SSL, such connections are Peer-to-peer relationships. The connections are transient. Every connection is associated with one session.

Session: An SSL session is an association between a client and a server. Sessions are created by the Handshake Protocol. Sessions define a set of cryptographic security parameters which can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.



SSL Architecture

A session state is defined by the following parameters

Session identifier: An arbitrary byte sequence chosen by the server to identify an active or resumable session state.

Peer certificate: An X509.v3 certificate of the peer. This element of the state may be null.

Compression method: The algorithm used to compress data prior to encryption.

Cipher spec: Specifies the bulk data encryption algorithm (such as null, AES, etc.) and a hash algorithm (such as MD5 or SHA-1) used for MAC calculation. It also defines cryptographic attributes such as the hash_size.

Master secret: 48-byte secret shared between the client and server.

Resumable: A flag indicating whether the session can be used to initiate new Connections.

A connection state is defined by the following parameters

Server and client random: Byte sequences that are chosen by the server and client for each connection.

Server write MAC secret: The secret key used in MAC operations on data sent by the server.

Client write MAC secret: The secret key used in MAC operations on data sent by the client.

Server write key: The secret encryption key for data encrypted by the server and decrypted by the client.

Client write key: The symmetric encryption key for data encrypted by the client and decrypted by the server.

Initialization vectors: When a block cipher in CBC mode is used, an initialization vector (IV) is maintained for each key. This field is first initialized by the SSL Handshake Protocol. Thereafter, the final cipher text block from each record is preserved for use as the IV with the following record.

Sequence numbers: Each party maintains separate sequence numbers for transmitted and received messages for each connection. When a party sends or receives a change cipher spec message, the appropriate sequence number is set-to zero. Sequence numbers may not exceed 264 – 1.

SSL session

An association between client & server .It created by the Handshake Protocol and define a set of cryptographic parameters .it may be shared by multiple SSL connections

SSL connection

A transient, peer-to-peer, communications link associated with 1 SSL session

SSL Record Protocol

Confidentiality: The Handshake Protocol defines a shared secret key that is used for conventional encryption of SSL payloads.

Message Integrity: The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC).

overall operation of the SSL Record Protocol. The Record Protocol takes an application message to be transmitted, fragments the data into manageable blocks, optionally compresses the data, applies a MAC,

encrypts, adds a header, and transmits the resulting unit in a TCP segment. Received data are decrypted, verified, decompressed, and reassembled before being delivered to higher-level users.

The first step is **fragmentation**. Each upper-layer message is fragmented into blocks of 214 bytes (16384 bytes) or less. Next, **compression** is optionally applied. Compression must be lossless and may not increase the content length by more than 1024 bytes.1In SSLv3 (as well as the current version of TLS),

no compression algorithm is specified, so the default compression algorithm is null. The next step in processing is to compute a **message authentication code** over the compressed data. For this purpose, a shared secret key is used.

SSL Change Cipher Spec Protocol

One of 3 SSL specific protocols which use the SSL Record protocol a single message causes pending state to become current hence updating the cipher suite in use

SSL Alert Protocol

specific alert are unexpected message, bad record mac, decompression failure, handshake failure, illegal parameter, close notify, no certificate, bad certificate, unsupported certificate, certificate revoked, certificate expired, certificate unknown It is Compressed & encrypted like all SSL data

Handshake Protocol

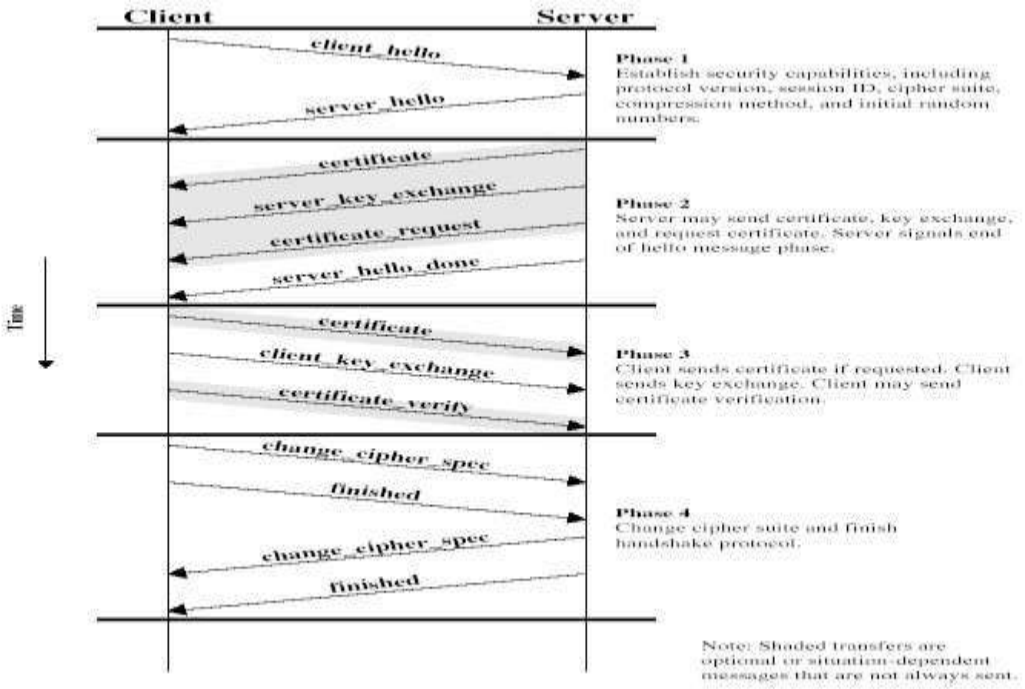
The most complex part of SSL is the Handshake Protocol. This protocol allows the server and client to authenticate each other and to negotiate an encryption and MAC algorithm and cryptographic keys to be

used to protect data sent in an SSL record. The Handshake Protocol is used before any application data is transmitted. The Handshake Protocol consists of a series of messages exchanged by client and server.

Type (1 byte): Indicates one of 10 messages.

Length (3 bytes): The length of the message in bytes.

Content (>=0 bytes): The parameters associated with this message



PHASE 1. ESTABLISH SECURITY CAPABILITIES This phase is used to initiate a logical connection and to establish the security capabilities that will be associated with it. The exchange is initiated by the client, which sends a **client hello message** with the following parameters:

Version: The highest SSL version understood by the client.

Random: A client-generated random structure consisting of a 32-bit timestamp and 28 bytes generated by a secure random number generator. These values serve as nonce's and are used during key exchange to prevent replay attacks.

Session ID: A variable-length session identifier. A nonzero value indicates that the client wishes to update the parameters of an existing connection or to create a new connection on this session. A zero value indicates that the client wishes to establish a new connection on a new session.

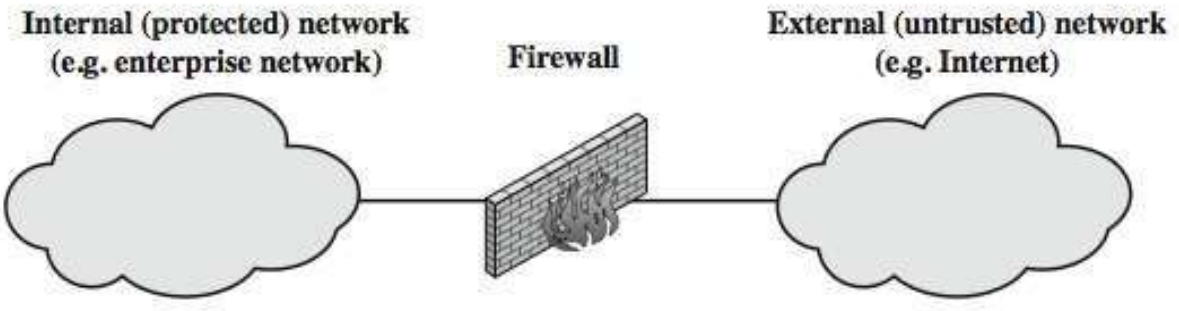
Cipher Suite: This is a list that contains the combinations of cryptographic algorithms supported by the client, in decreasing order of preference. Each element of the list (each cipher suite) defines both a key exchange algorithm and a Cipher Spec; these are discussed subsequently.

Compression Method: This is a list of the compression methods the client supports.

SECURE ELECTRONIC TRANSACTION

FIREWALLS

A firewall forms a barrier through which the traffic going in each direction must pass. A firewall security policy dictates which traffic is authorized to pass in each direction. A firewall may be designed to operate as a filter at the level of IP packets, or may operate at a higher protocol layer.



FIREWALL TECHNIQUES

Service control: Determines the types of Internet services that can be accessed, inbound or outbound. The firewall may filter traffic on the basis of IP address, protocol, or port number; may provide proxy software that receives and interprets each service request before passing it on; or may host the server software itself, such as a Web or mail service.

Direction control: Determines the direction in which particular service requests may be initiated and allowed to flow through the firewall.

User control: Controls access to a service according to which user is attempting to access it. This feature is typically applied to users inside the firewall perimeter (local users). It may also be applied to incoming traffic from external user. the latter requires some form of secure authentication technology, such as is provided in IPSec.

Behavior control: Controls how particular services are used. For example, the firewall may filter e-mail to eliminate spam, or it may enable external access to only a portion of the information on a local Web server.

TYPES OF FIREWALLS

A firewall may act as a packet filter. It can operate as a positive filter, allowing passing only packets that meet specific criteria, or as a negative filter, rejecting any packet that meets certain criteria. Depending on the type of firewall, it may examine one or more protocol headers in each packet, the payload of each packet

Source IP address: The IP address of the system that originated the IP packe**t**

(eg., 192.178.1.1)

Destination IP address: The IP address of the system the IP packet is trying to reach (e.g., 192.168.1.2)

Source and destination transport-level address: The transport-level (e.g., TCP or UDP) port number, which defines applications such as SNMP or TELNET

IP protocol field: Defines the transport protocol

Interface: For a firewall with three or more ports

INTRUDER One of the two most publicized threats to security is the intruder (the other is viruses), often referred to as a hacker or cracker. In an important early study of intrusion, Anderson [ANDE80] identified three classes of intruders:

Masquerader: An individual who is not authorized to use the computer and who penetrates a system's access controls to exploit a legitimate user's account

Misfeasor: A legitimate user who accesses data, programs, or resources for which such access is not authorized, or who is authorized for such access but misuses his or her privileges

Clandestine user: An individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection The masquerader is likely to be an outsider; the misfeasor generally is an insider; and the clandestine user can be either an outsider or an insider. Intruder attacks range from the benign to the serious. At the benign end of the scale, there are many people who simply wish to explore internets and see what is out there. At the serious end are individuals who are attempting to read privileged data, perform unauthorized modifications to data, or disrupt the system.[GRAN04] lists the following examples of intrusion:

Performing a remote root compromise of an e-mail server, defacing a Web server, Guessing and cracking passwords, copying a database containing credit card numbers, Viewing sensitive data, including payroll records and medical information, Without authorization, Running a packet sniffer on a workstation to capture usernames and passwords ,Using a permission error on an anonymous FTP server to distribute pirated software and music files, Dialing into an unsecured modem and gaining internal network access,

Posing as an executive, calling the help desk, resetting the executive's e-mail password and learning the new password Using an unattended, logged-in workstation without permission.

Intruder Behavior Patterns The techniques and behavior patterns of intruders are constantly shifting, to exploit newly discovered weaknesses and to evade detection and countermeasures. Even so, intruders typically follow one of a number of recognizable behavior patterns, and these patterns typically differ from those of ordinary users.

Some Examples of Intruder Patterns of Behavior

(a) Hacker

1. Select the target using IP lookup tools such as NSLookup, Dig, and others.
2. Map network for accessible services using tools such as NMAP.
3. Identify potentially vulnerable services (in this case, pcAnywhere).
4. Brute force (guess) pcAnywhere password.
5. Install remote administration tool called Dame Ware.
6. Wait for administrator to log on and capture his password.
7. Use that password to access remainder of network.

(b) Internal Threat

1. Create network accounts for themselves and their friends.
2. Access accounts and applications they wouldn't normally use for their daily jobs.
3. E-mail former and prospective employers.
4. Conduct furtive instant-messaging chats.
5. Visit Web sites that cater to disgruntled employees, such

(c) Criminal Enterprise

1. Act quickly and precisely to make their activities harder to detect.
2. Exploit perimeter through vulnerable ports.
3. Use Trojan horses (hidden software) to leave back doors for reentry.
4. Use sniffers to capture passwords.
5. Do not stick around until noticed.
6. Make few or no mistakes.

INTRUSION TECHNIQUES

The objective of the intruder is to gain access to a system or to increase the range of privileges accessible on a system. Most initial attacks use system or software vulnerabilities that allow a user to execute code that opens a back door into the system. Alternatively, the intruder attempts to acquire information that should have been protected. In some cases, this information is in the form of a user password. With knowledge of some other user's password, an intruder can log in to a system and exercise all the privileges accorded to the legitimate user. Typically, a system must maintain a file that associates a password with each authorized

user. If such a file is stored with no protection, then it is an easy matter to gain access to it and learn passwords. The password file can be protected in one of two ways:

One-way function: The system stores only the value of a function based on the user's password. When the user presents a password, the system transforms that password and compares it with the stored value. In practice, the system usually performs a one-way transformation (not reversible) in which the
Password is used to generate a key for the one-way function and in which a fixed-length output is produced.
Access control: Access to the password file is limited to one or a very few accounts.

Viruses

A computer virus is a piece of software that can "infect" other programs by modifying them; the modification includes injecting the original program with a routine to make copies of the virus program, which can then go on to infect other programs. Biological viruses are tiny scraps of genetic code—DNA or RNA—that can take over the machinery of a living cell and trick it into making thousands of flawless replicas of the original virus. Like its biological counterpart, a computer virus carries in its instructional code the recipe for making perfect copies of itself. The typical virus becomes embedded in a program on a computer. Then, whenever the infected computer comes into contact with an uninfected piece of software, a fresh copy of the virus passes into the new program. Thus, the infection can be spread from computer to computer by unsuspecting users who either swap disks or send programs to one another over a network. In a network environment, the ability to access applications and system services on other computers provides a perfect culture for the spread of a virus. A virus can do anything that other programs do. The difference is that a virus attaches itself to another program and executes secretly when the host program is run. Once a virus is executing, it can perform any function, such as erasing files and
programs that is allowed by the privileges of the current user.

A computer virus has three parts [AYCO06]:

Infection mechanism: The means by which a virus spreads, enabling it to replicate. The mechanism is also referred to as the **infection vector**.

Trigger: The event or condition that determines when the payload is activated or delivered.

Payload: What the virus does, besides spreading. The payload may involve damage or may involve benign but noticeable activity. During its lifetime, a typical virus goes through the following four phases:

Dormant phase: The virus is idle. The virus will eventually be activated by some event, such as a date, the presence of another program or file, or the capacity of the disk exceeding some limit. Not all viruses have this stage.

Propagation phase: The virus places a copy of itself into other programs or into certain system areas on the disk. The copy may not be identical to the propagating version; viruses often morph to evade detection. Each infected program will now contain a clone of the virus, which will itself enter a propagation phase.

Triggering phase: The virus is activated to perform the function for which it was intended. As with the dormant phase, the triggering phase can be caused by a variety of system events, including a count of the number of times that this copy of the virus has made copies of itself.

Execution phase: The function is performed. The function may be harmless, such as a message on the screen, or damaging, such as the destruction of programs and data files.

VIRUS STRUCTURE

A virus can be pretended or post pended to an executable program, or it can be embedded in some other fashion. The key to its operation is that the infected program, when invoked, will first execute the virus code and then execute the original code of the program. In this case, the virus code,V, is pretended to infected programs, and it is assumed that the entry point to the program, when invoked, is the first line of the

program. The infected program begins with the virus code and works as follows. The first line of code is a jump to the main virus program.

The second line is a special marker that is used by the virus to determine whether or not a potential victim program has already been infected with this virus. When the program is invoked, control is immediately transferred to the main virus program. The virus program may first seek out uninfected executable files and infect them. Next, the virus may perform some action, usually detrimental to the system. This action could be performed every time the program is invoked, or it could be a logic bomb that triggers only under certain conditions. Finally, the virus transfers control to the original program. If the infection

Logic for a Compression Virus

```
Program V: _
1234567;
subroutine infect-executable :_
{loop:
file :_ get-random-executable-file;
if (first-line-of-file _ 1234567)
then goto loop
else prepend V to file; }
subroutine do-damage :_
{whatever damage is to be done}
subroutine trigger-pulled :_
{return true if some condition holds}
main: main-program :_
{infect-executable;
if trigger-pulled then do-damage;
goto next;}
next:
```

}

{goto main;

A Simple Virus

21.2 / VIRUSES 21-9

program CV :_

{goto main;

01234567;

subroutine infect-executable :_

{loop:

file :_ get-random-executable-file;

if (first-line-of-file _ 01234567) then goto loop;

(1) compress file;

(2) prepend CV to file;

}

main: main-program :_

{if ask-permission then infect-executable;

(3) uncompress rest-of-file;

(4) run uncompressed file;}

}

Viruses Classification

Boot sector infector: Infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus.

File infector: Infects files that the operating system or shell consider to be executable.

Macro virus: Infects files with macro code that is interpreted by an application. A virus classification by concealment strategy includes the following categories:

Encrypted virus: A typical approach is as follows. A portion of the virus creates a random encryption key and encrypts the remainder of the virus. The key is stored with the virus. When an infected program is invoked, the virus uses the stored random key to decrypt the virus. When the virus replicates, a different random key is selected. Because the bulk of the virus is encrypted with a different key for each instance, there is no constant bit pattern to observe.

Stealth virus: A form of virus explicitly designed to hide itself from detection by antivirus software. Thus, the entire virus, not just a payload is hidden.

Polymorphic virus: A virus that mutates with every infection, making detection by the "signature" of the virus impossible.

Metamorphic virus: As with a polymorphic virus, a metamorphic virus mutates with every infection. The difference is that a metamorphic virus rewrites itself completely at each iteration, increasing the difficulty of detection.

Virus Kits

Another weapon in the virus writers' armory is the virus-creation toolkit. Such a toolkit enables a relative novice to quickly create a number of different viruses. Although viruses created with toolkits tend to be less sophisticated than viruses designed from scratch, the sheer number of new viruses that can be generated using a toolkit creates a problem for antivirus schemes.

Macro Viruses

In the mid-1990s, macro viruses became by far the most prevalent type of virus. Macro viruses are particularly threatening for a number of reasons. A macro virus is platform independent. Many macro viruses infect Microsoft Word documents or other Microsoft Office documents. Any hardware platform and operating system that supports these applications can be infected. Macro viruses infect documents, not executable portions of code. Most of the information introduced onto a computer system is in the form of a document rather than a program. Macro viruses are easily spread. A very common method is by electronic mail. Because macro viruses infect user documents rather than system programs, traditional file system access controls are of limited use in preventing their spread.

E-Mail Viruses

A more recent development in malicious software is the e-mail virus. The first rapidly spreading e-mail viruses, such as Melissa, made use of a Microsoft Word macro embedded in an attachment. If the recipient opens the e-mail attachment, the Word macro is activated.

1. The e-mail virus sends itself to everyone on the mailing list in the user's e-mail Package.

2. The virus does local damage on the user's system. Thus we see a new generation of malware that arrives via e-mail and uses e-mail Software features to replicate itself across the Internet. The virus propagates itself as soon as it is activated (either by opening an e-mail attachment or by opening the e-mail) to all of the e-mail addresses known to the infected host.

Intruders

- clearly a growing publicized problem
  - from "Wily Hacker" in 1986/87
  - to clearly escalating CERT stats
- may seem benign, but still cost resources
- may use compromised system to launch other attacks

Intrusion Techniques:

- aim to increase privileges on system
- basic attack methodology
  - target acquisition and information gathering
  - initial access
  - privilege escalation

- covering tracks

- key goal often is to acquire passwords
- so then exercise access rights of owner

Password Guessing:

- one of the most common attacks
- attacker knows a login (from email/web page etc)
- then attempts to guess password for it
  - try default passwords shipped with systems
  - try all short passwords
  - then try by searching dictionaries of common words
  - intelligent searches try passwords associated with the user (variations on names, birthday, phone, common words/interests)
  - before exhaustively searching all possible passwords
- check by login attempt or against stolen password file
- success depends on password chosen by user
- surveys show many users choose poorly

Password Capture:

- another attack involves password capture
  - watching over shoulder as password is entered
  - using a trojan horse program to collect
  - monitoring an insecure network login (eg. telnet, FTP, web, email)

- extracting recorded info after successful login (web history/cache, last number dialed etc)

- using valid login/password can impersonate user
- users need to be educated to use suitable precautions/countermeasures

Intrusion Detection:

- inevitably will have security failures
- so need also to detect intrusions so can
  - block if detected quickly
  - act as deterrent
  - collect info to improve security

Approaches to Intrusion Detection:

- statistical anomaly detection
  - threshold
  - profile based
- rule-based detection
  - anomaly
  - penetration identification

Honeypots:

- decoy systems to lure attackers
  - away from accessing critical systems
  - to collect information of their activities
  - to encourage attacker to stay on system so administrator can respond

- are filled with fabricated information
- instrumented to collect detailed information on attackers activities
- may be single or multiple networked systems

Password security

Password Management:

- front-line defense against intruders
- users supply both:
  - login – determines privileges of that user
  - password – to identify them
- passwords often stored encrypted
  - Unix uses multiple DES (variant with salt)
  - more recent systems use crypto hash function

Managing Passwords:

- need policies and good user education
- ensure every account has a default password
- ensure users change the default passwords to something they can remember
- protect password file from general access
- set technical policies to enforce good passwords
  - minimum length (>6)
  - require a mix of upper & lower case letters, numbers, punctuation
  - block know dictionary words

- may reactively run password guessing tools
- note that good dictionaries exist for almost any language/interest group
- may enforce periodic changing of passwords
- have system monitor failed login attempts, & lockout account if see too many in a short period
- do need to educate users and get support
- balance requirements with user acceptance
- be aware of **social engineering** attacks

Proactive Password Checking:

- most promising approach to improving password security
- allow users to select own password
- but have system verify it is acceptable
  - simple rule enforcement (see previous slide)
  - compare against dictionary of bad passwords
  - use algorithmic (markov model or bloom filter) to detect poor choices

Summary:

- have considered:
  - problem of intrusion
  - intrusion detection (statistical & rule-based)
- password management

CLASS: II MCA **COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY**

COURSE CODE: 17CAP404N **UNIT: IV** **BATCH-2017-2020**

UNIT IV

POSSIBLE QUESTIONS

1. Explain in detail about Handshake Protocol
2. Write short notes on
   (i) Types of Firewalls (ii) Intruder Behaviour Patterns
3. Explain in detail about SSL Architecture.
4. Write short notes on
   (i) FIREWALLS (ii) E-Mail Viruses
5. Describe about secure electronic transaction?
6. Explain about intrusion detection.
7. Explain in detail about Secure Socket Layer
8. Describe about intruders and its techniques?
9. Explain in detail about Web Security
10. Write short notes on
    (i) Types of viruses (ii)Virus countermeasures
11. Users A and B use the Diffie Hellman key exchange technique, a common prime q=11 and a primitive root alpha=7
    (i) If user A has private key XA=3.What is A's public key YA?
    (ii) If user B has private key XB=6 what is B's public key YB?
    (iii) What is the shared secret key? Also write the algorithm.

KARPAGAM ACADEMY OF HIGHER EDUCATION
DEPARTMENT OF COMPUTER APPLICATIONS
II MCA
CRYPTOGRAPHY AND NETWORK SECURITY

UNIT IV

S.NO	Question	Option 1	Option 2	Option 3	Option 4	Answer
1	_____ release of message contents to any person	disend	traffic	disclosure	none	disclosure
2	_____ discovery of the pattern of traffic between parties	traffic analysis	disclosure	timing	none	traffic analysis
3	_____ insertion of messages into the network from a fraudulent source	masquerade	timing	traffic	sequence	masquerade
4	_____modification changes to the contents of a message	traffic analysis	sequence	timing	content	content
5	_____ modification delay or replay of messages	timing	sequence	content	traffic	timing
6	_____ modification to a sequence of messages	timing	content	sequence	none	sequence
7	_____ is a mechanism or service used to verify the integrity of a message	message transformation	message authentication	sequence	traffic	message authentication
8	_____ encryption provides authentication among those who share the secret key.	symmetric	assymmetric	traffic	none	symmetric
9	A_____ is an algorithm that requires the use of a secret key	MAC	MCA	MAB	MAD	MAC
10	A____ function maps a variable-length message into a fixed length value	symmetric	assymmetric	hash	none	hash
11	_____ the ciphertext of the entire message and a serves as its authenticator	message authentication	message transformation	sequence	none	message authentication
12	the straight forward use of _____ key encryption provides confidentiality but not authentication	private	public	protected	none	public
13	An alternative authentication technique involves the use of a secret key to generate a small fixed size block of data known as a_____	encryption	Cryptographic checksum	decryption	traffic	Cryptographic checksum
14	the hash code is also referred to as a _____	message code	authentication	message digest	sequence	message digest
15	the _____is a function of all the bits of the message and provides an error_____ capability	process	creation	detection	insertion	detection
16	A MAC is generated by a function C of the form_____	MAC=C(K,M)	MAC=C(K,N)	MAC=E(K,M)	MAC=D(K,N)	MAC=C(K,M)
17	the data authentication algorithm , based on _____,has been one of the most widley used MACs	AES	DES	DFS	AEM	DES
18	A hash value h is generated by a function H of the form _____	h=H(M)	h=H(N)	h=H(NM)	h=H(MM)	h=H(M)
19	the purpose of a hash function is to produce a_____ of a file,message,or other block of data.	HASH	fingerprint	message digest	none	fingerprint
20	For any given value h.it is computionally infeasible to find x such that H(x)=h.this is sometimes referred to the literature as the _____	one-way property	two way property	multi way property	none	one-way property
21	For any given block x,it is computationally infeasible to find 'y not equal x' with H(y)=H(x) referredto as_____	one-way property	weak collision resistance	strong collision resistance	none	weak collision resistance
22	It is computationally infeasible to find any pair(x,y)such thatH(x)=H(y) is referred to as_____	strong collision resistance	weak collision resistance	one-way property	none	strong collision resistance
23	_____protects two parties who exchange messages from any third party	message transformation	message authentication	sequence	none	message authentication
24	The ____ must be a bit pattern that depends on the message being signed	digital signature	message authentication	sequence	none	digital signature
25	_____ may be formed by encrypting the entire message with the sender private ket	message authentication	digital signature	sequence	none	digital signature
26	_____ is an NIST standard that uses the secure hash algorithms	DSS	AES	DES	RSA	DSS
27	In _____ authentication the recipient wants some assurence that a message is from the alleged sender	Two way	multi way	one way	three way	one way
28	_____,the opponent simply copies a message and replys it later	repetition that can be logged	simple reply	timing reply	none	simple reply
29	_____,the opponent can replay a timestamped message within the valid time window	repetition that can be logged	simple reply	timing reply	none	repetition that can be logged

30	_____ cannot be used for encryption or key exchange	RSAA	AES	DES	RSA	RSA

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

COURSE CODE: 17CAP404N UNIT: V BATCH-2017-2020

SYLLABUS

Case Study: Implementation of Cryptographic Algorithms – RSA – DSA – ECC (C / JAVA Programming). Network Forensic – Security Audit; Other Security Mechanism: Introduction to Stenography – Quantum Cryptography – Water Marking – DNA Cryptography.

Implementation of Cryptographic Algorithms

RSA Algorithm

Private-Key Cryptography:

- traditional **private/secret/single key** cryptography uses **one** key
- Key is shared by both sender and receiver
- if the key is disclosed communications are compromised
- also known as **symmetric**, both parties are equal
- hence does not protect sender from receiver forging a message & claiming is sent by sender

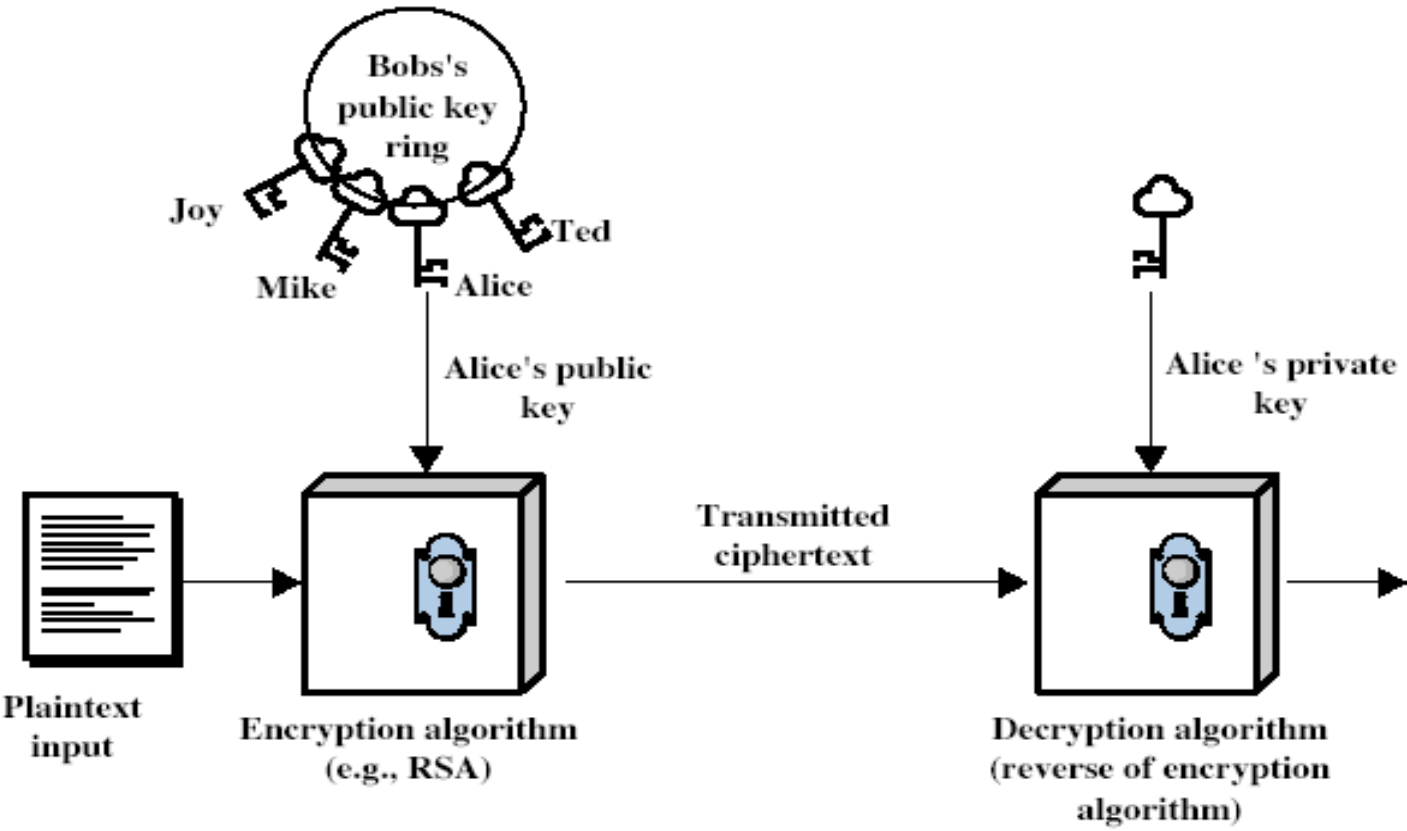
Public-Key Cryptography:

- probably most significant advance in the 3000 year history of cryptography
- uses **two** keys – a public key and a private key
- **asymmetric** since parties are **not** equal
- uses clever application of number theory concepts to function
- complements **rather than** replaces private key cryptography
- public-key/two-key/asymmetric cryptography involves the use of two keys
- a public-key, which may be known by anybody, and can be used to encrypt messages, and verify signatures

- a private-key, known only to the recipient, used to decrypt messages, and sign (create) signatures
- those who encrypt messages or verify signatures cannot decrypt messages or create signatures

figure:

Public-Key Cryptography



Public-Key Characteristics:

- Public-Key algorithms rely on two keys with the characteristics that it is:
  - computationally infeasible to find decryption key knowing only algorithm & encryption key

- computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
- either of the two related keys can be used for encryption, with the other used for decryption (in some schemes)

Security of Public Key Schemes:

- like private key schemes brute force **exhaustive search** attack is always theoretically possible
- but keys used are too large (>512bits)
- security relies on a **large enough** difference in difficulty between **easy** (en/decrypt) and **hard** (cryptanalyse) problems
- more generally the **hard** problem is known, its just made too hard to do in practise
- requires the use of **very large numbers**
- hence is **slow** compared to private key schemes

RSA:

- by Rivest, Shamir & Adleman of MIT in 1977
- best known & widely used public-key scheme
- based on exponentiation in a finite (Galois) field over integers modulo a prime
  - nb. exponentiation takes $O((\log n)^3)$ operations (easy)
- uses large integers (eg. 1024 bits)
- security due to cost of factoring large numbers
  - nb. factorization takes $O(e^{\log n \log \log n})$ operations (hard)

RSA Key Setup:

- each user generates a public/private key pair by:
- selecting two large primes at random - p, q
- computing their system modulus N=p.q
    - note ø(N)=(p-1)(q-1)
- selecting at random the encryption key e
        - where 1<e<ø(N), gcd(e,ø(N))=1
- solve following equation to find decryption key d
    - e.d=1 mod ø(N) and 0≤d≤N
- publish their public encryption key: KU={e,N}
- keep secret private decryption key: KR={d,p,q}

RSA Use:

- to encrypt a message M the sender:
    - obtains public key of recipient KU={e,N}
    - computes: $C=M^e \bmod N$, where $0\leq M<N$
- to decrypt the ciphertext C the owner:
    - uses their private key KR={d,p,q}
    - computes: $M=C^d \bmod N$
- note that the message M must be smaller than the modulus N (block if needed)

Why RSA Works:

- because of Euler's Theorem:

- $a^{ø(n)} \bmod N = 1$
  - where gcd(a,N)=1
- in RSA have:
  - N=p.q
  - ø(N)=(p-1)(q-1)
  - carefully chosen e & d to be inverses mod ø(N)
  - hence e.d=1+k.ø(N) for some k
- hence :
  $C^d = (M^e)^d = M^{1+k.ø(N)} = M^1.(M^{ø(N)})^q = M^1.(1)^q = M^1 = M \bmod N$

RSA Example:

1. Select primes: p=17 & q=11
2. Compute $n = pq$ =17×11=187
3. Compute ø(n)=(p–1)(q-1)=16×10=160
4. Select e *:* gcd(e,160)=1; choose e=7
5. Determine d*:* de=1 mod 160 and d < 160 Value is d=23 since 23×7=161= 10×160+1
6. Publish public key KU={7,187}
7. Keep secret private key KR={23,17,11}

Exponentiation:

- can use the Square and Multiply Algorithm
- a fast, efficient algorithm for exponentiation

- concept is based on repeatedly squaring base
- and multiplying in the ones that are needed to compute the result
- look at binary representation of exponent
- only takes $O(\log_2 n)$ multiples for number n
    - eg. $7^5 = 7^4.7^1 = 3.7 = 10 \bmod 11$
    - eg. $3^{129} = 3^{128}.3^1 = 5.3 = 4 \bmod 11$

RSA Key Generation:

- users of RSA must:
    - determine two primes at random - p, q
    - select either e or d and compute the other
- primes p,q must not be easily derived from modulus N=p.q
    - means must be sufficiently large
    - typically guess and use probabilistic test
- exponents e, d are inverses, so use Inverse algorithm to compute the other

Factoring Problem:

- mathematical approach takes 3 forms:
    - factor N=p.q, hence find ø(N) and then d
    - determine ø(N) directly and find d
    - find d directly
- currently believe all equivalent to factoring
    - have seen slow improvements over the years

- as of Aug-99 best is 130 decimal digits (512) bit with GNFS
- biggest improvement comes from improved algorithm
  - cf "Quadratic Sieve" to "Generalized Number Field Sieve"
- barring dramatic breakthrough 1024+ bit RSA secure
  - ensure p, q of similar size and matching other constraints

Timing Attacks:

- developed in mid-1990's
- exploit timing variations in operations
  - eg. multiplying by small vs large number
  - or IF's varying which instructions executed
- infer operand size based on time taken
- RSA exploits time taken in exponentiation
- countermeasures
  - use constant exponentiation time
  - add random delays
  - blind values used in calculations

Summary:

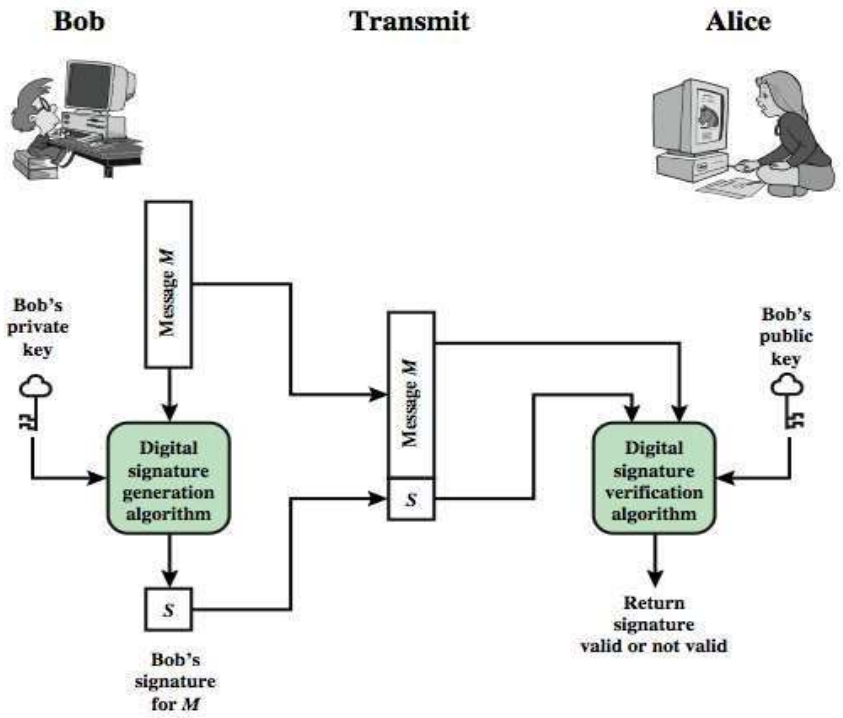
- **have considered:**
  - principles of public-key cryptography
  - RSA algorithm, implementation, security

DSA

Digital Signatures algorithm:

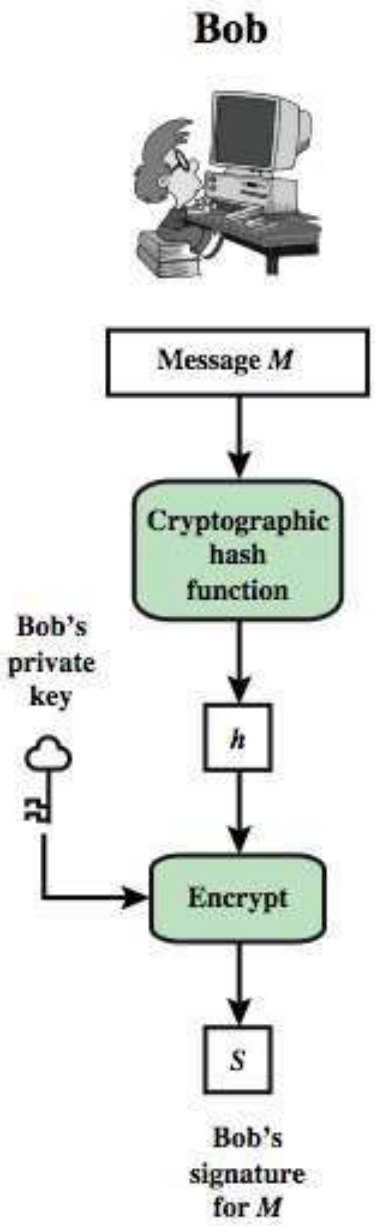
- have looked at message authentication
  - but does not address issues of lack of trust
- digital signatures provide the ability to:
  - verify author, date & time of signature
  - authenticate message contents
  - be verified by third parties to resolve disputes
- hence include authentication function with additional capabilities

Digital Signature Model:



Digital Signature Model:

Digital Signature Model



Attacks and Forgeries:

- attacks
  - key-only attack
  - known message attack
  - generic chosen message attack
  - directed chosen message attack
  - adaptive chosen message attack
- break success levels
  - total break
  - selective forgery
  - existential forgery

Digital Signature Requirements:

- must depend on the message signed
- must use information unique to sender
  - to prevent both forgery and denial
- must be relatively easy to produce
- must be relatively easy to recognize & verify
- be computationally infeasible to forge
  - with new message for existing digital signature
  - with fraudulent digital signature for given message
- be practical save digital signature in storage

Direct Digital Signatures:

- involve only sender & receiver
- assumed receiver has sender's public-key
- digital signature made by sender signing entire message or hash with private-key
- can encrypt using receivers public-key
- important that sign first then encrypt message & signature
- security depends on sender's private-key

ElGamal Digital Signatures:

- signature variant of ElGamal, related to D-H
  - so uses exponentiation in a finite (Galois)
  - with security based difficulty of computing discrete logarithms, as in D-H
- use private key for encryption (signing)
- uses public key for decryption (verification)
- each user (eg. A) generates their key
  - chooses a secret key (number): $1 < x_A < q-1$
  - compute their **public key**: $y_A = a^{xA} \bmod q$

ElGamal Signature Example:

- use field GF(19) q=19 and a=10
- Alice computes her key:
  - A chooses x_A=16 & computes y_A=$10^{16} \bmod 19 = 4$

- Alice signs message with hash m=14 as (3,4):
  - choosing random K=5 which has gcd(18,5)=1
  - computing $S_1 = 10^5 \bmod 19 = 3$
  - finding $K^{-1} \bmod (q-1) = 5^{-1} \bmod 18 = 11$
  - computing $S_2 = 11(14-16.3) \bmod 18 = 4$
- any user B can verify the signature by computing
  - $V_1 = 10^{14} \bmod 19 = 16$
  - $V_2 = 4^3.3^4 = 5184 = 16 \bmod 19$
  - since 16 = 16 signature is valid

Schnorr Digital Signatures:

- also uses exponentiation in a finite (Galois)
  - security based on discrete logarithms, as in D-H
- minimizes message dependent computation
  - multiplying a 2*n-bit* integer with an *n-bit* integer
- main work can be done in idle time
- have using a prime modulus *p*
  - *p–1* has a prime factor *q* of appropriate size
  - typically *p* 1024-bit and *q* 160-bit numbers

Schnorr Key Setup:

- choose suitable primes *p , q*
- choose *a* such that $a^q = 1 \bmod p$

- (a,p,q) are global parameters for all
- each user (eg. A) generates a key
  - chooses a secret key (number): $0 < s_A < q$
  - compute their **public key**: $v_A = a^{-sA} \bmod q$

Schnorr Signature:

- user signs message by
  - choosing random r with $0<r<q$ and computing $x = a^r \bmod p$
  - concatenate message with x and hash result to computing: $e = H(M \| x)$
  - computing: $y = (r + se) \bmod q$
  - signature is pair (e, y)
- any other user can verify the signature as follows:
  - computing: $x' = a^y v^e \bmod p$
  - verifying that: $e = H(M \| x')$

Digital Signature Standard (DSS):

- US Govt approved signature scheme
- designed by NIST & NSA in early 90's
- published as FIPS-186 in 1991
- revised in 1993, 1996 & then 2000
- uses the SHA hash algorithm
- DSS is the standard, DSA is the algorithm
- FIPS 186-2 (2000) includes alternative RSA & elliptic curve signature variants

- DSA is digital signature only unlike RSA
- is a public-key technique

Network Security Forensics

As security threats grow in complexity and organizations face stringent regulatory requirements to audit access to private data on the network, organizations require an increased level of network visibility and surveillance. Network security forensics, a new method of capturing and storing every packet traversing the network, has emerged to address this need.

Where network security forensics is needed:

- Monitoringinternal threats
- Documentingevidencefor investigations
- Solvingthe"Whodunnit"mystery
- Regulatoryaudit compliancefor HIPAA,SOX, GLB,and the EU Data Protection Directive
- Complying withcorporate HRandacceptable-usepolicies

Network security forensic tools have emerged to address this need. The appliances are unique in their ability to capture and save terabytes of packet-level data to local disk or SAN. dministrators then select a specific time period of captured network activity, and the appliance sifts through the indexed traffic to identify any anomalous traffic or potential security breach. In analyzing and

identifying security events such as an attack, these devices support intrusion detection rules such as Snort, or other signature-based rule systems. In addition to identifying security breaches, forensic tools should have the ability to reconstruct captured packet data into its original format, whether itbeane-mail, IM, webpage, VoIPcall,orother form of communication.

Network Security Auditing

Network Security Auditing Tool:

Comprehensive Security Auditing

With years of experience providing security audits for leading global organizations we understand that it is important for a security audit to cover all potential threats and not just be a review of firewall rules. So although Nipper Studio provides a thorough security audit of firewall rules it also provides an audit of many other configuration options that could also pose a security risk and provide you with the information to make an informed decision.

Prioritize Your Security Issues

Nipper Studio includes support for both the industry standard CVSSv2 rating system and the established Nipper v1 rating systems. When using the CVSSv2 rating system you can define what is important for your environment, such as confidentiality or integrity, and any identified security issue ratings will be calculated to reflect your own priorities. The Nipper v1 rating system is an excellent all round choice when you also want to include best practice priorities.

Intelligent Analysis

Some configuration options can be as simple as either on or off, however many modern network devices can often be complex to configure with configuration options that influence each other. Nipper Studio understands the relationships between those complex settings and is able to determine if a security risk is exposed. Nipper Studio also excels in areas which are traditionally weak for automated tools.

Advanced Dynamic Report Writing Technology

Nipper Studio includes advanced dynamic report writing technology that enables the creation of a report that reads like a human had written it. Each sentence that is written takes into consideration account names, plurality and references amongst other details. Many of the audit companies that make use of Nipper Studio for their own work will often use the report text directly in their own reports.

Customize Checks For Your Organization

At Titania we understand that different organizations have their own policies, such as password requirements. Although we have configured Nipper Studio with what we believe to be sensible default values, there are a huge number of options to enable you to fine tune your security audits. Once configured Nipper Studio will identify compliance with your own organizations policy, report on any discrepancies and make recommendations that reflect your policy.

Security Auditing Overview

Software Versions:

The network devices software may be out of date and contain known vulnerabilities. An attacker may be able to exploit a vulnerability to gain access to your devices.

Authentication Passwords:

Ensure that access cannot be gained using default, dictionary- based or weak passwords. A huge number of real-world attacks are based on exploiting authentication credentials.

Authentication Services:

Ensure that remote and local authentication services securely configured. A weakness with these could enable an attacker to gain access to your device.

Administration Services:

Administrative services are key to the security of any network device. Weak configurations could enable an attacker to re-configure your network.

VPN Configuration:

VPNs are widely used to connect remote networks or enable mobile workers to connect to a corporate network. Weaknesses with their configuration could enable a remote attacker to gain access to your network.

Web Services: It is common for modern network devices to provide web-based services to perform a variety of tasks.

Time Synchronisation: Accurate time synchronisation can be critical for authentication services, is important for logging and forensic investigations.

Name Resolution Services: Name lookups are used by a number of different network services. Nipper identifies security issues with those settings.

Logon Messages: Logon messages are easily overlooked when configuring a device. These should be used to provide a legal warning against unauthorised access without leaking sensitive information.

Firewall Rules: Network filtering is important for restricting access to networks and services. Weak rules could enable an attacker to gain unauthorised access to systems.

Intrusion Detection Systems: Many modern network devices now include IDS functionality. These facilities are useful for detecting potential attacks against your systems and alerting system administrators.

Intrusion Prevention Systems: Similar to IDS, IPS can proactively prevent attacks. These settings can provide an extra layer of defence that can stop an attack dead.

Routing Protocols: Routing protocols are useful for complex networks, adapting to topology changes and dynamically routing network communications. An attacker could use weaknesses with these protocols to modify the path that network communications take.

Cryptographic Settings: Encryption is important to prevent eavesdropping of sensitive information. Vulnerabilities and weaknesses can exist with encryption that could lead an attacker to de-crypt captured communications.

Logging: The logging of system events is not only important for troubleshooting problems but is useful for detecting and tracing an attack.

Wireless: Many devices offer wireless connectivity. Weaknesses with the wireless configuration could enable an attacker within wireless range to gain access to your network.

Printing Services: Print servers are often overlooked, but could be used by an attacker for storage or used to attack other network systems.

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

COURSE CODE: 17CAP404N UNIT: V BATCH-2017-2020

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

COURSE CODE: 17CAP404N UNIT: V BATCH-2017-2020

DNA Cryptography

CLASS: II MCA **COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY**

COURSE CODE: 17CAP404N **UNIT: V** **BATCH-2017-2020**

Various Device Specific Settings: Devices will often include configuration settings that are unique to those devices that have a security impact.

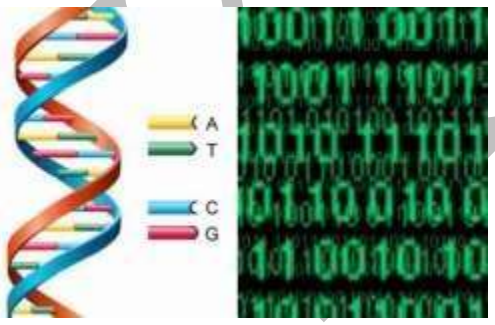
Bio-CryptographyBio-cryptography or Bioencryption is a next generation security mechanism, storing almost a million gigabytes of data inside bacteria. Research from two prominent universities indicates that it is not only possible but also practical to store digital data in the genome of a living organism and retrieve that data hundreds or even thousands of years later, after the organism has reproduced its genetic material through hundreds of generations.

Note: A milliliter of liquid can contain up to 1 billion bacteria, and you can see that the potential capacity of bacteria-based memory is enormous. The idea of storing data inside bacteria has been around for about a decade. Even very simple bacteria have long strands of DNA with tons of

bases available for data encryption, and bacteria are by their nature far more resilient to damage than more traditional electronic storage. Bacteria are nature's hardiest survivors, capable of surviving just about any disaster that would finish off a regular hard drive. Besides, bacteria's natural reproduction would create lots of redundant copies of the data, which would help preserve the integrity of the information and make retrieval easier.

Preparing traditional data for storage inside bacteria is simple enough. There are four DNA bases that can be used to make up the DNA strings: adenine, cytosine, guanine, and thymine. That basically means we're working with a four number system, also known as quaternary numbers.

In a presentation on their breakthrough, the Hong Kong researchers showed how to change the word "iGEM" into DNA-ready code. They used the ASCII table to convert each of the individual letters into a numerical value (i=105, G=71, etc.), which can then be changed from base-10 to base-4 (105=1221, 71=0113, etc.). Finally, those numbers can be changed into their DNA base equivalents, with 0, 1, 2, and 3 replaced with A, T, C, and G. And so iGEM becomes atctattgatttatgt.



Once the raw data is ready, the researchers say a few algorithms can be used to weed out redundant and repetitive information. That doesn't just save a ton of space - lots of repetition in the DNA sequence can actually be biologically harmful to the wellbeing of the DNA and bacteria, so this step rather neatly solves two problems at once.

DNA strands aren't long enough to store complicated information like a photograph or a book, so the best available solution is to fragment the data into lots of little pieces and spread it among the different cells. To make that work, the researchers have to create a system that allows the fragments to identified and ultimately put back in the right order. So they created a three-part structure for all the DNA: header, message, and checksum.

The header is an 8-base-long sequence that is divided into four levels of identifying information - zone, region, area and district - which allows each fragment to be put back in the right order. After the message carries the actual usable data, the checksum provides a repetition of the original header, which is useful in controlling for minor mutations to the bacteria.

So, let's say the information has been encrypted and placed in lots of different cells of bacteria. How then does someone retrieve the data on the other end? The decrypter would take the DNA and run it through what's known as next-generation high-throughput sequencing, or NGS. This particular type of sequencing analyzes and compares multiple copies of the same sequence and

then uses majority-voting to figure out which bases are correct if parts of the data have decayed. Then the compression algorithms could be reversed to restore the raw data to its original form.

The last step would be snapping the fragments back together in the correct order so that the DNA strands could be translated back into useful data. This is where we go from just data storage to data *encryption*. The person trying to read the data needs a formula that will reveal the right order of the headers and checksums - without that formula, the data remains meaningless.

Now, there does seem like one potential concern with using *E. coli* to store data: isn't*E. coli* dangerous? It appears there's not too much to worry about there - the researchers used non-virulent strains of the bacteria, and the bacteria can't do much more than store the data and reproduce. The DNA sequences that represent the data are total gibberish when it comes to encoding potentially dangerous proteins.

Cryptography - Watermarking

Copyright Crusaders

What if you took a really great photograph and posted it to your website, and the next thing you know its on the cover of a magazine and the credit and the money belong to someone else?

Unless you can prove that you took the photo, and youre ready to sue, youre out of luck. But theres hope for you in a technique called content watermarking being researched by the Cryptography Group at Microsoft Research. It imprints the image with a watermark that is not visible to the naked eye, but can be detected by another piece of software. The detection software can be embedded within a spider that crawls all over the World Wide Web and finds all the watermarked images, making sure that no one is displaying them in violation of copyright laws. Similar techniques developed by Microsoft Research Fellow Darko Kirovski and Research Director Henrique Malvar can be used to watermark audio files.

International cryptography researchers, always testing their own work, have developed techniques to remove watermarks. Actually, you dont need a special program to remove a watermark, a combination of effects in most photo editing programs will suffice.

Cryptography researchers Mariusz Jakubowski and Ramarathnam Venkatesan have developed a program that produced watermarks that have survived photo editing techniques and cryptographic attacks designed to remove watermarks. They plan to release it soon, but they also expect to have to improve it as soon as it is released because they have no doubt that the watermark erasers will find a way to defeat it.

Its a never ending duel, sighs Jakubowski. Perhaps someday he will find a way to make a permanent watermark, and artists and publishers everywhere will erect statues in his honor. Then again, maybe hell just have to stay one step ahead of the competition.

CLASS: II MCA COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY

COURSE CODE: 17CAP404N UNIT: V BATCH-2017-2020

Quantum Cryptography

Quantum cryptography uses our current knowledge of physics to develop a cryptosystem that is not able to be defeated - that is, one that is completely secure against being compromised without knowledge of the sender or the receiver of the messages. The word*quantum* itself refers to the most fundamental behavior of the smallest particles of matter and energy: quantum theory explains everything that exists and nothing can be in violation of it.Quantum cryptography is different from traditional cryptographic systems in that it relies more on physics, rather than mathematics, as a key aspect of its security model.

Essentially, quantum cryptography is based on the usage of individual particles/waves of light (photon) and their intrinsic quantum properties to develop an unbreakable cryptosystem - essentially because it is impossible to measure the quantum state of any system without disturbing that system. It is theoretically possible that other particles could be used, but photons offer all the necessary qualities needed, their behavior is comparatively well-understood, and they are the information carriers in optical fiber cables, the most promising medium for extremely high-bandwidth communications.

How It Works in Theory

In theory, quantum cryptography works in the following manner (this view is the "classical" model developed by Bennett and Brassard in 1984 - some other models do exist):

Assume that two people wish to exchange a message securely, traditionally named Alice and Bob. Alice initiates the message by sending Bob a key, which will be the mode for encrypting the message data. This is a random sequence of bits, sent using a certain type of scheme, which can see two different initial values represent one particular binary value (0 or 1).

Let us assume that this key is a stream of photons travelling in one direction, with each of these photon particles representing a single bit of data (either a 0 or 1). However, in addition to their linear travel, all of these photons are oscillating (vibrating) in a certain manner. These oscillations can occur in any 360-degree range across any conceivable axis, but for the purpose of simplicity (at least as far as it is possible to simplify things in quantum cryptography), let us assume that their oscillations can be grouped into 4 particular states: we'll define these as UP/DOWN, LEFT/RIGHT, UPLEFT/RIGHTDOWN and UPRIGHT/LEFTDOWN. The angle of this vibration is known as the polarization of the photon. Now, let us introduce a polarizer into the equation. A polarizer is simply a filter that permits certain photons to pass through it with the same oscillation as before and lets others pass through in a changed state of oscillation (it can also block some photons completely, but let's ignore that property for this exercise). Alice has a polarizer that can transmit the photons in any one of the four states mentioned - in effect, she can choose either rectilinear (UP/DOWN and LEFT/RIGHT) or diagonal (UPLEFT/RIGHTDOWN and UPRIGHT/LEFTDOWN) polarization filters.

Alice swaps her polarization scheme between rectilinear and diagonal filters for the transmission of each single photon bit in a random manner. In doing so, the transmission can have one of two polarizations represent a single bit, either 1 or 0, in either scheme she uses.

When receiving the photon key, Bob must choose to measure each photon bit using either his rectilinear or diagonal polarizer: sometimes he will choose the correct polarizer and at other times he will choose the wrong one. Like Alice, he selects each polarizer in a random manner. So what happens with the photons when the wrong polarizer is chosen?

The Heisenberg Uncertainty Principle states that we do not know exactly what will happen to each individual photon, for in the act of measuring its behavior, we alter its properties (in addition to the fact that if there are two properties of a system that we wish to measure, measuring one precludes us from quantifying the other). However, we can make a guess as to what happens with them as a group. Suppose Bob uses a rectilinear polarizer to measure UPLEFT/RIGHTDOWN and UPRIGHT/LEFTDOWN (diagonal) photons. If he does this, then the photons will pass through in a changed state - that is, half will be transformed to UP/DOWN and the other half to LEFT/RIGHT. But we cannot know which individual photons will be transformed into which state (it is also a reality that some photons may be blocked from passing altogether in a real world application, but this is not relevant to the theory).

Bob measures some photons correctly and others incorrectly. At this point, Alice and Bob establish a channel of communication that can be insecure - that is, other people can listen in. Alice then proceeds to advise Bob as to which polarizer she used to send each photon bit - but not how she polarized each photon. So she could say that photon number 8597 (theoretically) was sent using the rectilinear scheme, but she will not say whether she sent an UP/DOWN or LEFT/RIGHT. Bob then confirms if he used the correct polarizer to receive each particular photon. Alice and Bob then discard all the photon measurements that he used the wrong polarizer to check. What they have, is, on average, a sequence of 0s and 1s that is half the length of the original transmission...but it will form the basis for a one-time pad, the only cryptosystem that, if properly implemented, is proven to be completely random and secure.

Now, suppose we have an eavesdropper, Eve, who attempts to listen in, has the same polarizers that Bob does and must also randomly choose whether to use the rectilinear or diagonal one for each photon. However, she also faces the same problem that Bob does, in that half the time she

will choose the wrong polarizer. But Bob has the advantage of speaking to Alice to confirm which polarizer type was used for each photon. This is useless to Eve, as half the time she used the wrong detector and will misinterpret some of the photons that will form that final key, rendering it useless.

Furthermore, there is another level of security inherent in quantum cryptography - that of intrusion detection. Alice and Bob would know if Eve was eavesdropping on them. The fact that Eve is on the "photon highway" can become obvious because of the following.

Let's say that Alice transmits photon number 349 as an UPRIGHT/LEFTDOWN to Bob, but for that one, Eve uses the rectilinear polarizer, which can only measure UP/DOWN or LEFT/RIGHT photons accurately. What Eve will do is transform that photon into either UP/DOWN or LEFT/RIGHT, as that is the only way the photon can pass. If Bob uses his rectilinear polarizer, then it will not matter what he measures as the polarizer check Alice and Bob go through above will discard that photon from the final key. But if he uses the diagonal polarizer, a problem arises when he measures its polarization; he may measure it correctly as UPRIGHT/LEFTDOWN, but he stands an equal chance, according to the Heisenberg Uncertainty Principle, of measuring it incorrectly as UPLEFT/RIGHTDOWN. Eve's use of the wrong polarizer will warp that photon and will cause Bob to make errors even when he is using the correct polarizer.

To discover Eve's nefarious doings, they must perform the above procedures, with which they will arrive at an identical key sequence of 0s and 1s - unless someone has been eavesdropping, whereupon there will be some discrepancies. They must then undertake further measures to check the validity of their key. It would be foolish to compare all the binary digits of the final key over the unsecured channel discussed above, and also unnecessary.

Let us assume that the final key comprises 4,000 binary digits. What needs to be done is that a subset of these digits be selected randomly by Alice and Bob, say 200 digits, in terms of both position (that is, digit sequence number 2, 34, 65, 911 etc) and digit state (0 or 1). Alice and Bob compare these - if they match, then there is virtually no chance that Eve was listening. However, if she was listening in, then her chances of being undiscovered are one in countless trillions, that is, no chance in the real world. Alice and Bob would know someone was listening in and then would not use the key - they would need to start the key exchange again over a secure channel inaccessible to Eve, even though the comparisons between Alice and Bob discussed above can still be done over an insecure channel. However, even if Alice and Bob have concluded that the their key is secure, since they have communicated 200 digits over an un-secure channel, these 200 digits should be discarded from the final key, turning it from a 4,000 into a 3,800 bit key).

Thus, quantum cryptography is a way to combine the relative ease and convenience of key exchange in public key cryptography with the ultimate security of a onetime pad.

How It Works in Practice

In practice, quantum cryptography has been demonstrated in the laboratory by IBM and others, but over relatively short distances. Recently, over longer distances, fiber optic cables with incredibly pure optic properties have successfully transmitted photon bits up to 60 kilometers. Beyond that, BERs (bit error rates) caused by a combination of the Heisenberg Uncertainty Principle and microscopic impurities in the fiber make the system unworkable. Some research has seen successful transmission through the air, but this has been over short distances in ideal weather conditions. It remains to be seen how much further technology can push forward the distances at which quantum cryptography is practical.

Practical applications in the US are suspected to include a dedicated line between the White House and Pentagon in Washington, and some links between key military sites and major defense contractors and research laboratories in close proximity.

Network Forensics

Network forensics is the capture, recording, and analysis of network events in order to discover the source of security attacks or other problem incidents. (The term, attributed to firewall expert Marcus Ranum, is borrowed from the legal and criminology fields where*forensics* pertains to the investigation of crimes.) According to Simson Garfinkel, author of several books on security, network forensics systems can be one of two kinds:

- *"Catch-it-as-you-can" systems*, in which all packets passing through a certain traffic point are captured and written to storage with analysis being done subsequently in batch mode. This approach requires large amounts of storage, usually involving a RAID system.
- *"Stop, look and listen" systems*, in which each packet is analyzed in a rudimentary way in memory and only certain information saved for future analysis. This approach requires less storage but may require a faster processor to keep up with incoming traffic.

Both approaches require significant storage and the need for occasional erasing of old data to make room for new. The open source programs *tcpdump* and *windump* as well as a number of commercial programs can be used for data capture and analysis.

One concern with the "catch-it-as-you-can" approach is one of privacy since all packet information (including user data) is captured. Internet service providers (ISPs) are expressly forbidden by the Electronic Communications Privacy Act (ECPA) from eavesdropping or disclosing intercepted contents except with user permission, for limited operations monitoring, or

under a court order. The U.S. FBI's Carnivore is a controversial example of a network forensics tool.Network forensics products are sometimes known as Network Forensic Analysis Tools (NFATs).

STEGANOGRAPHY-INTRODUCTION

- Steganography is the art and science of hiding communication; a
- steganographic system thus embeds hidden content in unremarkable cover media so as not to arouse an eavesdropper's suspicion. In the past, people used hidden tattoos or invisible ink to convey steganographic content. Today, computer and network technologies provide easy-to-use communication channels for steganography. Essentially,the information-hiding process in a steganographic system starts by identifying a cover medium's redundant bits (those that can be modified without destroying that medium's integrity).
- The embedding process creates a stego medium by replacing these redundant bits with data from the hidden message. Modern steganography's goal is to keep its mere presence undetectable, but steganographic systems—because of their invasive nature—leave behind detectable traces in the cover medium. Even if secret content is not revealed, the existence of it is: modifying the cover medium changes its statistical properties, so eavesdroppers can detect the distortions in the resulting stego medium's statistical properties. The process of finding these distortions is calledstatistical steganalysis.
- This article discusses existing steganographic systems and presents recent research in detecting them via statistical steganalysis. Other surveys focus on the general usage of information hiding and watermarking or else provide an overview of detection algorithms. Here, we present recent research and discuss the practical application of detection algorithms and the mechanisms for getting around them.

- Steganography is a technique used to hide information within images. Using stenography, watermarks and copyrights can be placed on an image to protect the rights of its owner without altering the appearance of the image. Almost like magic, images, executable programs, and text messages can hide in images. The cover image does not appear to be altered. People look at the cover image and never suspect something is hidden. Your information is hidden in plain sight.
- Steganography in History
- Steganography comes from Greek and means “covered writing”. The ancient Greeks wrote text on wax-covered tablets. To pass a hidden message, a person would scrape off the wax and write the message on the underlying wood. He/She would then once again cover the wood with wax so it appeared unused. Many developments in steganography occurred during world war II. This included the development of invisible inks, microdots, and encoded messages.
- While cryptography is preoccupied with the protection of the contents of a message or information, steganography concentrates on concealing the very existence of such messages from detection. The term steganography is adapted from the Greek word steganographia,meaning “covered writing”, and is taken in its modern form to mean the hiding of information inside other information [1].
- Naturally these techniques date back throughout history, the main applications being in couriering information during times of war.
- The Greek writer Herodotus gave a famous anecdotal account of this around 440 B.C. His tale was of a Demeratus, a Greek in the Persian court who warned Sparta of an invasion by Xerxes, the King of Persia. He did this by removing the wax from a writing tablet, scoring his message in the wood underneath, and then covering it with wax again before sending it

toSparta [1]. With the invention of digital audio and images files this has taken on a whole new meaning; creating new methods for performing "reversible data hiding" as it is often dubbed.

UNIT V

POSSIBLE QUESTIONS

1. Explain in detail about RSA Cryptographic Algorithm
2. Write a short note on
   (i) Quantum Cryptography (ii) Water marking
3. Describe about Private-Key Cryptography and Public-Key Cryptography?
4. Discuss about Steganography.
5. Explain in detail about DSA Cryptographic Algorithm
6. Discuss about Network Security Auditing and Watermarking
7. Explain in detail about RSA Algorithm
8. Discuss about Network Forensics and Network Security Auditing Tool.
9. Explain in detail about ECC
10. Describe in detail about DNA Cryptography?
11. Illustrate encryption and decryption using One-Time Pad method.

CLASS: II MCA **COURSE NAME: CRYPTOGRAPHY AND NETWORK SECURITY**

COURSE CODE: 17CAP404N **UNIT: V** **BATCH-2017-2020**

KARPAGAM ACADEMY OF HIGHER EDUCATION

DEPARTMENT OF COMPUTER APPLICATIONS

II MCA

CRYPTOGRAPHY AND NETWORK SECURITY

UNIT V

S.NO	Question	Option 1	Option 2	Option 3	Option 4	Answer
1	_____ is an authentication service designed for use in a distributed environment	kerberos	X.509	secure	none	kerberos
2	_____ defines the format for public key certificates	X.508	X.507	X.505	X.509	X.509
3	_____ is an authentication service developed as part of project athena at MIT	secure	X.509	kerberos	none	kerberos
4	A network eavesdropper should not be able to obtain the necessary information to impersonate a user is called_____	reliable	secure	scalable	transparent	secure
5	_____,for all services that rely on kerberos for access control lack of availability of the service	reliable	secure	scalable	transparent	reliable
6	_____,the user should not be aware that suthentication is taking place	reliable	transparent	secure	scalable	transparent
7	TGS stands for	ticket granting server	ticket gain server	ticket grant sell	track grant server	ticket granting server
8	_____ is a set of managed nodes that share the same kerberos databaswe	secure	X.509	kerberos relam	none	kerberos relam
9	_____,which is a service or user that is known to the kerberos systems	kerberos principal	server	X.505	ticket granting server	kerberos principal
10	_____which provide unambiguous byte ordering	BER	AER	CER	DER	BER
11	BER stands for	basic enabling rules	basic emmiting rules	basic editing rules	basic encoding rules	basic encoding rules
12	PCBC stands for	proper cipher block chaining	propagating cipher block chaining	propagating cipher base chaining	propagating chemical base chaining	propagating cipher block chaining
13	X.509 defines a frame work for the provision of authentication services by the _____directory to its users	X.500	X.600	X.700	X.800	X.500
14	_____covers all of the other fields of the certificate	transparent	server	signature	secure	signature
15	_____ way authentication involves a single transfer of information from one user to another user	one	two	three	four	one
16	_____as the set of hardware,software,people policies and procedures	digital signature	private key infrastructure	public key infrastructure	none	public key infrastructure
17	_____used to denote any method for storing certificates and CRLs.	repository	private key infrastructure	public key infrastructure	none	repository
18	_____ is an open source freely available aoftware package for e mail security	PGP	PGA	PGB	PGC	PGP
19	_____incorporates tools for developing a public key trust model and public key certificate management	PGA	PGB	PGP	PGC	PGP
20	_____ ia an internet standard approach to e mail security that incorporates the same functionality as PGP	S/MIME	PGP	digital signature	none	S/MIME
21	PGP stands for	prime good privacy	pretty good privacy	pretty good prime	pretty gain privacy	pretty good privacy
22	_____,which is provided by encrypting messages to be transmitted or to be stored locally as files	PGP confidentiality	PGP compression	PGP segment	PGP email	PGP confidentiality
23	_____ the message sfter applying the signature but before encryption	PGP email	PGP segment	PGP compression	PGP confidentiality	PGP compression
24	_____ that indicates the extent to which PGP will trust that this is a valid public key for this user	signature trust key	owner trust key	key legitimacy field	none	key legitimacy field
25	_____ that indicates the degree to which this PGP users trust the signer to certify public keys	signature trust key	owner trust key	key legitimacy field	none	signature trust key
26	S/MIME stands for	secure/multipurpose internet mail extension	small/multipurpose internet mail extension	small/multiple internet mail expansion	secure/multipurpose internet message extension	secure/multipurpose internet mail extension
27	_____ defines a format for text message that are sent using electronic mail	RFC 822	RFC 833	RFC 922	RFC 722	RFC 822
28	_____ cannot transmit executable files or other binary objects	MMTP	MIME	SMTP	SMME	SMTP
29	_____ cannot transmit text data that indicates national language charactures	SMTP	MMTP	MIME	SMME	SMTP
30	_____ server may reject mail message over a certain size	MMTP	SMTP	MIME	SMME	SMTP
31	_____ used to identify MIME entities uniquely in multiple contexts	course id	server id	client id	content id	content id
32	_____ transfer encoding is useful when the data consists largely of octets that correspond to printable ASCII characters	server labels	security labels	quoted printable	client lables	quoted printable

33	_____ may be included in the authenticated attributes of a signed data object	security labels	quoted printable	server labels	client lables	security labels
34	_____ is a capability that can be added to either current version of the intenet protocol	web security	IP security	internet security	network security	IP security
35	_____ defines a number of techniques for key management	IP security	internet security	network security	web security	IP security
36	_____ provides security services between TCP and application that use TCP	secure socket layer	handshake	transport layer service	cipher spec	secure socket layer
37	the internet standard version is called_____	handshake	transport layer service	secure socket layer	cipher spec	transport layer service
38	_____ is an open encryption and security specification designed to protect credit card transactions on the internet		handshake	cipher spec	none	secure electronic transaction
39	the most complex part of SSL is the _____ protocol	handshake	cipher spec	alter	none	handshake
40	_____ provides trust by the use of X.509v3 digital signature	handshake	cipher spec	secure electronic transaction	alter	secure electronic transaction

Reg. No.....

[15CAP404N]

KARPAGAM UNIVERSITY

Karpagam Academy of Higher Education
(Established Under Section 3 of UGC Act 1956)
COIMBATORE – 641 021
(For the candidates admitted from 2015 onwards)

MCA DEGREE EXAMINATION, APRIL 2017

Fourth Semester

COMPUTER APPLICATIONS

CRYPTOGRAPHY AND NETWORK SECURITY

Time: 3 hours Maximum : 60 marks

PART – A (20 x 1 = 20 Marks) (30 Minutes)
(Question Nos. 1 to 20 Online Examinations)

PART B (5 x 6 = 30 Marks)
Answer ALL the Questions

21. (a) Write short notes on
(i) Security Attack (ii) Security Services (iii) Security Algorithm
(or)
(b) Explain briefly about Key Cryptosystem

22. (a) Explain in detail about principles of public key cryptosystem.
(or)
(b) Describe about Elliptic Curve cryptography?

23. (a) Explain in detail about Kerberos.
(or)
(b) Explain in detail about PGP with diagram.

24. (a) Explain in detail about Web Security
(or)
(b) Write short notes on : (i) Types of viruses (ii) Virus countermeasures

25. (a) Explain in detail about RSA Cryptographic Algorithm
(or)
(b) Write a short note on: (i) Quantum Cryptography (ii) Water marking

PART C (1 x 10 = 10 Marks)
(Compulsory)

26. Perform encryption and decryption using RSA algorithm with p=3, q=11, e=7 and M=5

Reg. No.....

[13CAP505N]

KARPAGAM UNIVERSITY

Karpagam Academy of Higher Education
(Established Under Section 3 of UGC Act 1956)
COIMBATORE – 641 021
(For the candidates admitted from 2013 onwards)

MCA DEGREE EXAMINATION, NOVEMBER 2015

Fifth Semester

COMPUTER APPLICATIONS

CRYPTOGRAPHY AND NETWORK SECURITY

Time: 3 hours Maximum : 60 marks

PART – A (20 x 1 = 20 Marks) (30 Minutes)
(Question Nos. 1 to 20 Online Examinations)

PART B (5 x 8 = 40 Marks) (2 ½ Hours)
Answer ALL the Questions

21. a. Explain Simplified DES with example.

Or

b. List and explain the characteristics of advanced symmetric ciphers.

22. a. Write RSA algorithm with example.

Or

b. Explain the attacks on Hash-function and MACs.

23. a. Explain the steps involved in Kerberos authentication service.

Or

b. What do you mean by Security Association (SA)? Explain the parameters associated with SA.

24. a. Define Intrusion detection. Explain any two mechanisms.

Or

b. Explain any two firewalls.

25. a. Implement DSA algorithm using C.

Or

b. What is Watermarking? Explain its types.

Reg. No.....

[12CAP505N]

KARPAGAM UNIVERSITY

(Under Section 3 of UGC Act 1956)
COIMBATORE – 641 021
(For the candidates admitted from 2012 onwards)

MCA DEGREE EXAMINATION, NOVEMBER 2014

Fifth Semester

COMPUTER APPLICATIONS

CRYPTOGRAPHY AND NETWORK SECURITY

Time: 3 hours Maximum : 100 marks

PART – A (15 x 2 = 30 Marks)
Answer ALL the Questions

1. What is a stream cipher?
2. Write a brief note on ideal block cipher.
3. List down the symmetric key operations.
4. What are the four possible approaches for attacking the RSA algorithm?
5. What is man-in-middle attack?
6. Define Traffic algorithm.
7. Write a brief note on S/MIME.
8. Define MIME content types.
9. What is the application of IPSec?
10. What are the parameters used to define a connection state?
11. Write a brief note on SSL connection and session.
12. What are the elements of authorization response message?
13. What is Network forensic?
14. What is DNA Cryptography?
15. How to implement RSA in C?

PART B (5 X 14= 70 Marks)
Answer ALL the Questions

16. a. i. Elaborate Symmetric key cryptography and the problems of key distribution.
ii. Explain Diffi-Hellman key Exchange Algorithm.

Or

b. Give an overview of algorithm modes with a neat diagram.

17. a. Explain in detail about RSA algorithm.

Or

b. Describe about Fermat's and Euler's theorems.

18. a. Describe the authentication procedures.

Or

b. Give an overview of IP security Architecture with a neat diagram.

19. a. Explain in detail about SSL Architecture.

Or

b. Write a short note on : i. Quantum Cryptography ii. Water marking.

20. **Compulsory : -**

Design an authentication system by using Kerberos working group.

Reg. No.....

[14CAP85N]

KARPAGAM UNIVERSITY
Karpagam Academy of Higher Education
(Established Under Section 3 of UGC Act 1956)
COIMBATORE – 641 021
(For the candidates admitted from 2014 onwards)

MCA DEGREE EXAMINATION, NOVEMBER 2016
Fifth Semester

COMPUTER APPLICATIONS

CRYPTOGRAPHY AND NETWORK SECURITY

Time: 3 hours Maximum : 60 marks

PART – A (20 x 1 = 20 Marks) (30 Minutes)
(Question Nos. 1 to 20 Online Examinations)

PART B (5 x 8 = 40 Marks) (2 ½ Hours)
Answer ALL the Questions

21. a. Explain the overview of algorithm modes with neat diagram

Or

b. Explain briefly : (i) DES (ii) AES (iii) Blowfish

22. a. Explain Diffie-Hellman key Exchange Algorithm.

Or

b. Explain in detail about Message Authentication Code

23. a. Explain in detail about X.509 Authentication Services

Or

b. Explain in detail about S/MIME.

24. a. Explain in detail about Secure Socket Layer

Or

b. Describe about intruders and its techniques?

25. a. Describe about Private-Key Cryptography and Public-Key Cryptography?

Or

b. Discuss about Steganography

Reg. No.....

[15CAP404N]

KARPAGAM UNIVERSITY

Karpagam Academy of Higher Education
(Established Under Section 3 of UGC Act 1956)
COIMBATORE – 641 021
(For the candidates admitted from 2015 onwards)

MCA DEGREE EXAMINATION, APRIL 2017

Fourth Semester

COMPUTER APPLICATIONS

CRYPTOGRAPHY AND NETWORK SECURITY

Time: 3 hours Maximum : 60 marks

PART – A (20 x 1 = 20 Marks) (30 Minutes)
(Question Nos. 1 to 20 Online Examinations)

PART B (5 x 6 = 30 Marks)
Answer ALL the Questions

21. (a) Write short notes on
(i) Security Attack (ii) Security Services (iii) Security Algorithm
(or)
(b) Explain briefly about Key Cryptosystem

22. (a) Explain in detail about principles of public key cryptosystem.
(or)
(b) Describe about Elliptic Curve cryptography?

23. (a) Explain in detail about Kerberos.
(or)
(b) Explain in detail about PGP with diagram.

24. (a) Explain in detail about Web Security
(or)
(b) Write short notes on : (i) Types of viruses (ii) Virus countermeasures

25. (a) Explain in detail about RSA Cryptographic Algorithm
(or)
(b) Write a short note on: (i) Quantum Cryptography (ii) Water marking

PART C (1 x 10 = 10 Marks)
(Compulsory)

26. Perform encryption and decryption using RSA algorithm with p=3, q=11, e=7 and M=5

Reg. No.....

[13CAP505N]

KARPAGAM UNIVERSITY

Karpagam Academy of Higher Education
(Established Under Section 3 of UGC Act 1956)
COIMBATORE – 641 021
(For the candidates admitted from 2013 onwards)

MCA DEGREE EXAMINATION, NOVEMBER 2015

Fifth Semester

COMPUTER APPLICATIONS

CRYPTOGRAPHY AND NETWORK SECURITY

Time: 3 hours Maximum : 60 marks

PART – A (20 x 1 = 20 Marks) (30 Minutes)
(Question Nos. 1 to 20 Online Examinations)

PART B (5 x 8 = 40 Marks) (2 ½ Hours)
Answer ALL the Questions

21. a. Explain Simplified DES with example.

Or

b. List and explain the characteristics of advanced symmetric ciphers.

22. a. Write RSA algorithm with example.

Or

b. Explain the attacks on Hash-function and MACs.

23. a. Explain the steps involved in Kerberos authentication service.

Or

b. What do you mean by Security Association (SA)? Explain the parameters associated with SA.

24. a. Define Intrusion detection. Explain any two mechanisms.

Or

b. Explain any two firewalls.

25. a. Implement DSA algorithm using C.

Or

b. What is Watermarking? Explain its types.

Reg. No.....

[12CAP505N]

KARPAGAM UNIVERSITY

(Under Section 3 of UGC Act 1956)
COIMBATORE – 641 021
(For the candidates admitted from 2012 onwards)

MCA DEGREE EXAMINATION, NOVEMBER 2014

Fifth Semester

COMPUTER APPLICATIONS

CRYPTOGRAPHY AND NETWORK SECURITY

Time: 3 hours Maximum : 100 marks

PART – A (15 x 2 = 30 Marks)
Answer ALL the Questions

1. What is a stream cipher?
2. Write a brief note on ideal block cipher.
3. List down the symmetric key operations.
4. What are the four possible approaches for attacking the RSA algorithm?
5. What is man-in-middle attack?
6. Define Traffic algorithm.
7. Write a brief note on S/MIME.
8. Define MIME content types.
9. What is the application of IPSec?
10. What are the parameters used to define a connection state?
11. Write a brief note on SSL connection and session.
12. What are the elements of authorization response message?
13. What is Network forensic?
14. What is DNA Cryptography?
15. How to implement RSA in C?

PART B (5 X 14= 70 Marks)
Answer ALL the Questions

16. a. i. Elaborate Symmetric key cryptography and the problems of key distribution.
ii. Explain Diffi-Hellman key Exchange Algorithm.

Or

b. Give an overview of algorithm modes with a neat diagram.

17. a. Explain in detail about RSA algorithm.

Or

b. Describe about Fermat's and Euler's theorems.

18. a. Describe the authentication procedures.

Or

b. Give an overview of IP security Architecture with a neat diagram.

19. a. Explain in detail about SSL Architecture.

Or

b. Write a short note on : i. Quantum Cryptography ii. Water marking.

20. **Compulsory : -**

Design an authentication system by using Kerberos working group.

Reg. No.....

[14CAP85N]

KARPAGAM UNIVERSITY
Karpagam Academy of Higher Education
(Established Under Section 3 of UGC Act 1956)
COIMBATORE – 641 021
(For the candidates admitted from 2014 onwards)

MCA DEGREE EXAMINATION, NOVEMBER 2016
Fifth Semester

COMPUTER APPLICATIONS

CRYPTOGRAPHY AND NETWORK SECURITY

Time: 3 hours Maximum : 60 marks

PART – A (20 x 1 = 20 Marks) (30 Minutes)
(Question Nos. 1 to 20 Online Examinations)

PART B (5 x 8 = 40 Marks) (2 ½ Hours)
Answer ALL the Questions

21. a. Explain the overview of algorithm modes with neat diagram

Or

b. Explain briefly : (i) DES (ii) AES (iii) Blowfish

22. a. Explain Diffie-Hellman key Exchange Algorithm.

Or

b. Explain in detail about Message Authentication Code

23. a. Explain in detail about X.509 Authentication Services

Or

b. Explain in detail about S/MIME.

24. a. Explain in detail about Secure Socket Layer

Or

b. Describe about intruders and its techniques?

25. a. Describe about Private-Key Cryptography and Public-Key Cryptography?

Or

b. Discuss about Steganography