CRYPTOGRAPHY & NETWORK SECURITY

CRYPTOGRAPHY & NETWORK SECURITY

UNIT -I
SYMMETRIC CIPHERS

Overview – Classical encryption techniques – Block ciphers and data encryption standard –Finite fields – Advanced encryption standard – Contemporary symmetric ciphers – Confidentiality using symmetric encryption.

UNIT- II
PUBLIC-KEY ENCRYPTION AND HASH FUNCTIONS

Number theory – Public-key cryptography and RSA – Key management – Diffie-hellman key exchange – Elliptic curve cryptography – Message authentication and hash functions – Hash algorithms – Digital signatures and authentication protocols.

UNIT -III
NETWORK SECURITY PRACTICE

Authentication applications – Kerberos – X.509 authentication service – Electronic mail security – Pretty good privacy – S/MIME – IP security – IP security architecture – Authentication header – Encapsulating security payload – Key management.

UNIT- IV
SYSTEM SECURITY

Intruders – Intrusion detection – Password management – Malicious software – Firewalls – Firewall design principles – Trusted systems.

UNIT V
WIRELESS SECURITY

Wireless LAN security standards – Wireless LAN security factors and issues.

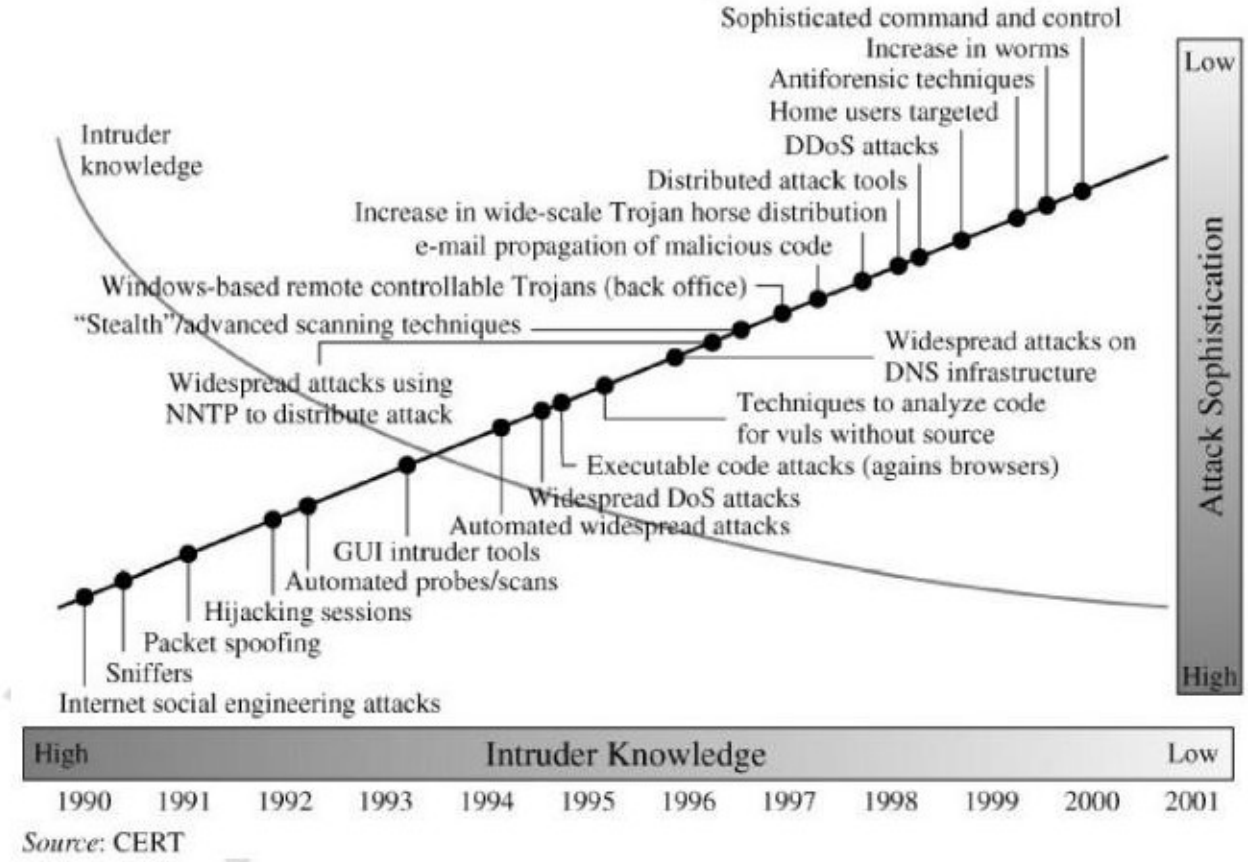
UNIT I FUNDAMENTALS

OSI security architecture – Classical encryption techniques – Cipher principles –Data encryption standard – Block cipher design principles and modes of operation –Evaluation criteria for AES – AES cipher – Triple DES – Placement of encryption function – Traffic confidentiality.

Security Trends:

In 1994, the Internet Architecture Board (IAB) issued a report entitled "Security in the Internet Architecture" (RFC 1636). The report stated the general consensus that the Internet needs more and better security, and it identified key areas for security mechanisms. Among these were the need to secure the network infrastructure from unauthorized monitoring and control of network traffic and the need to secure end-user-to-end-user traffic using authentication and encryption mechanisms.

Figure 1.1. Trends in Attack Sophistication and Intruder Knowledge



The OSI Security Architecture

To assess effectively the security needs of an organization and to evaluate and choose various security products and policies, the manager responsible for security needs some systematic way of defining the requirements for security and characterizing the approaches to satisfying those requirements.

ITU-T Recommendation X.800, *Security Architecture for OSI,* defines such a systematic approach. The OSI security architecture is useful to managers as a way of organizing the task of providing security

The OSI security architecture was developed in the context of the OSI protocol architecture.

The OSI security architecture focuses on security attacks, mechanisms, and services:

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

Security mechanism: A process (or a device incorporating such a process) that is designed to detect, prevent, or recover from a security attack.

Security service: A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.

Table 1. Threats and Attacks

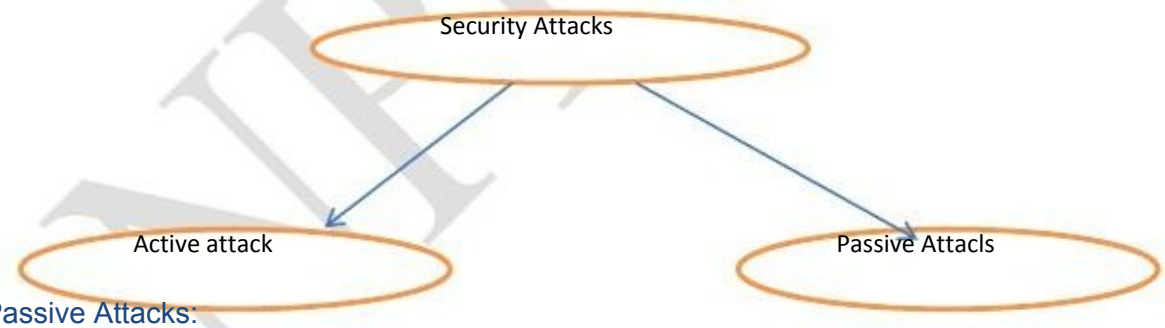
Threat

A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability.

Attack

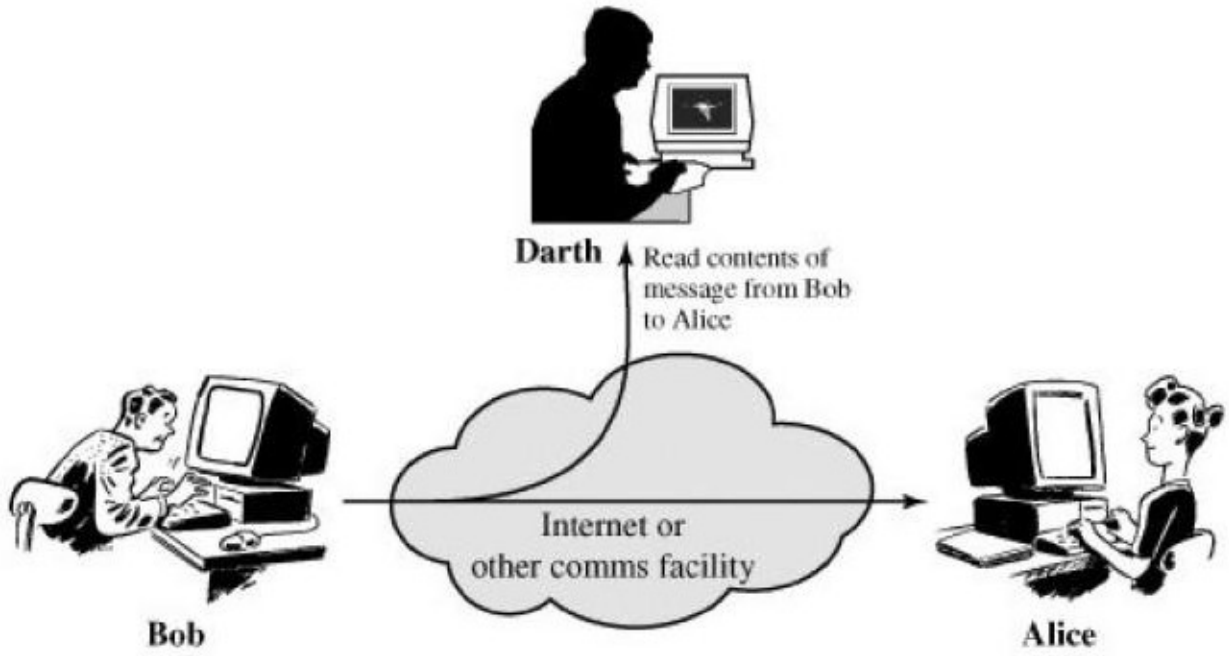
An assault on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt (especially in the sense of a method or technique) to evade security services and violate the security policy of a system.

Security Attacks:

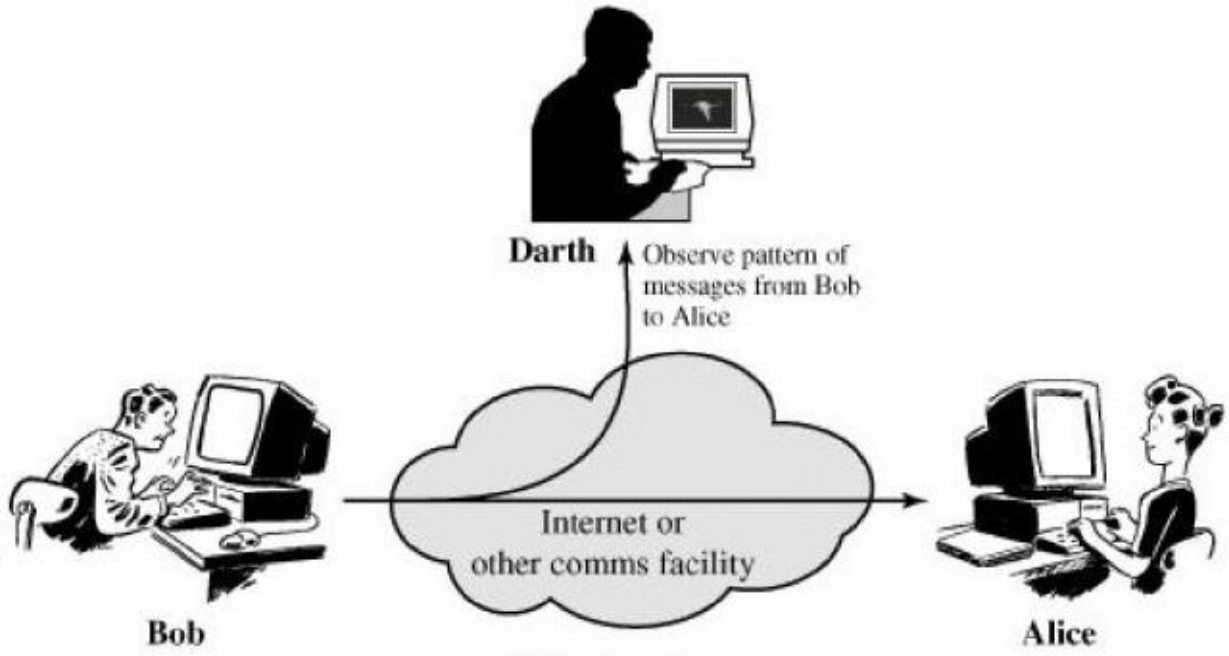


Passive Attacks:

Passive attacks are in the nature of eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted. Two types of passive attacks are release of message contents and traffic analysis.



(a) Release of message contents

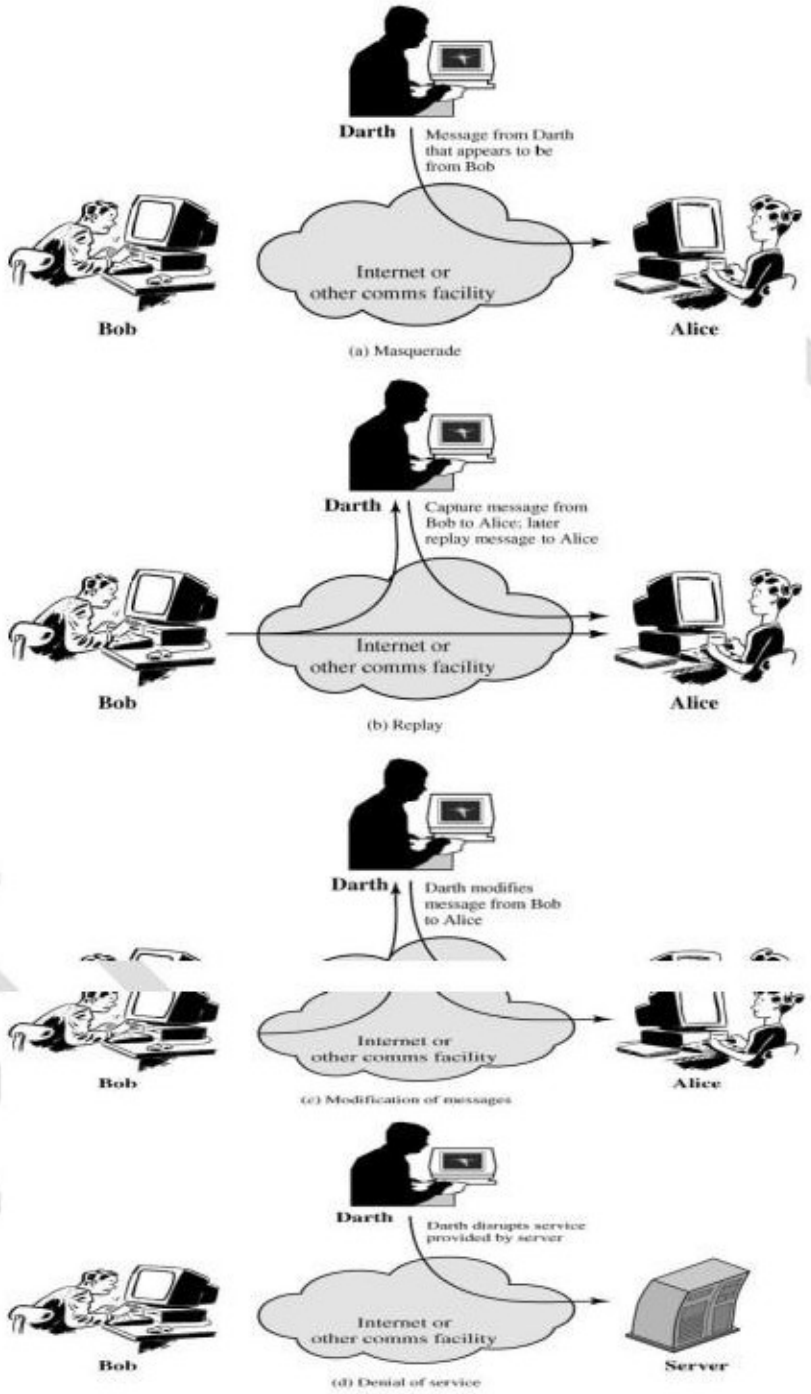


(b) Traffic analysis

Passive attacks are very difficult to detect because they do not involve any alteration of the data. Typically, the message traffic is sent and received in an apparently normal fashion and neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern. However, it is feasible to prevent the success of these attacks, usually by means of encryption. Thus, the emphasis in dealing with passive attacks is on prevention rather than detection.

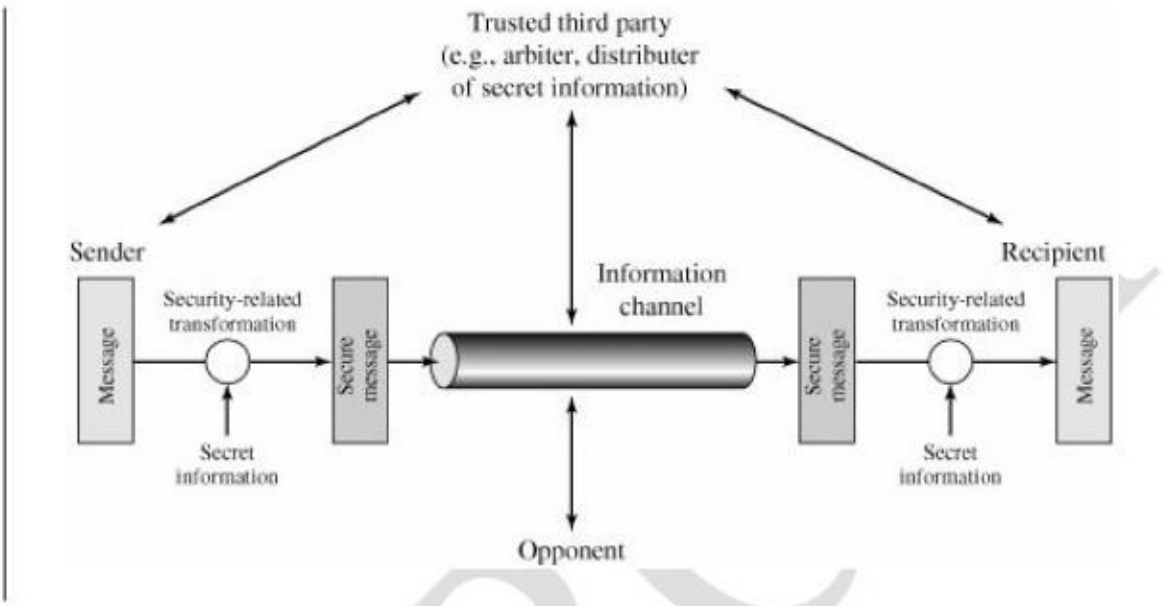
Active Attacks:

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



A Model for Network Security

The two parties, who are the *principals* in this transaction, must cooperate for the exchange to take place. A logical information channel is established by defining a route through the internet from source to destination and by the cooperative use of communication protocols (e.g., TCP/IP) by the two principals.

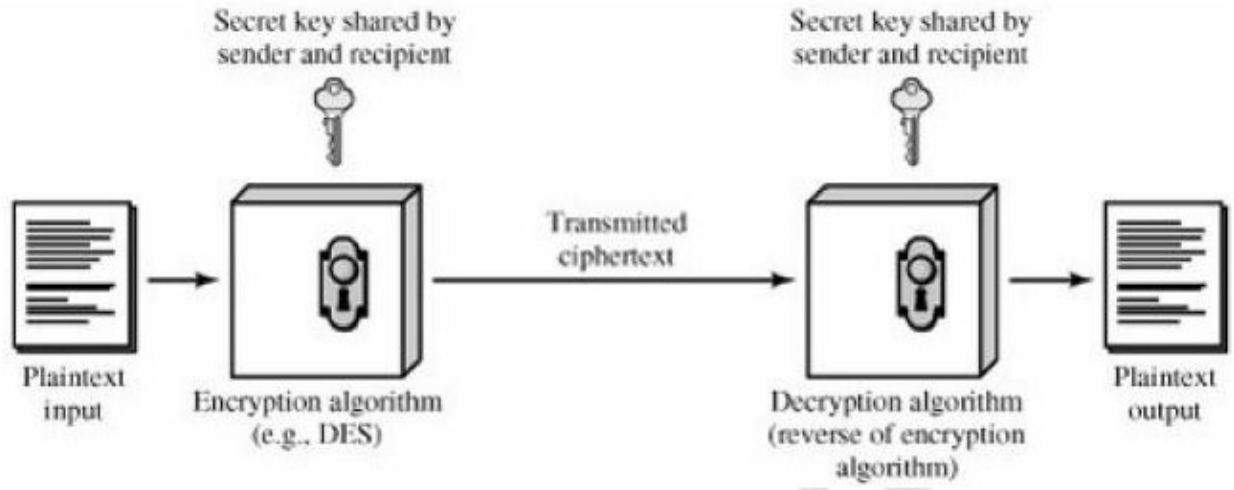


- A security-related transformation on the information to be sent.
- Some secret information shared by the two principals and, it is hoped, unknown to the opponent

2. Classical Encryption Techniques

plaintext, while the coded message is called the ciphertext. The process of converting from plaintext to ciphertext is known as enciphering or encryption; restoring the plaintext from the ciphertext is deciphering or decryption. The many schemes used for encryption constitute the area of study known as cryptography. Such a scheme is known as a cryptographic system or a cipher. Techniques used for deciphering a message without any knowledge of the enciphering details fall into the area of cryptanalysis. Cryptanalysis is what the layperson calls "breaking the code." The areas of cryptography and cryptanalysis together are called cryptology.

Symmetric Cipher Model:



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

Sender and receiver must have obtained copies of the secret key in a secure fashion and must keep the key secure.

Caesar Cipher:

The Caesar cipher involves replacing each letter of the alphabet with the letter standing three places further down the alphabet.

EXAMPLE:

plain: meet me after the toga party

cipher: PHHW PH DIWHU WKH WRJD SDUWB

Three important characteristics of this problem enabled us to use a brute-force cryptanalysis:

1. The encryption and decryption algorithms are known.

2. There are only 25 keys to try.

3. The language of the plaintext is known and easily recognizable

Monoalphabetic Ciphers:

instead, the "cipher" line can be any permutation of the 26 alphabetic characters, then there are 26! or greater than 4 x 10^{26}possible keys. This is 10 orders of magnitude greater than the key space for DES and would seem to eliminate brute-force techniques for cryptanalysis. Such an approach is referred to as a monoalphabetic substitution cipher.

PLAYFAIR CIPHER:

The Playfair algorithm is based on the use of a 5 x 5 matrix of letters constructed using a keyword..

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

1. Repeating plaintext letters that are in the same pair are separated with a filler letter, such as x, so that balloon would be treated as ba lx lo on.

2. Two plaintext letters that fall in the same row of the matrix are each replaced by the letter to the right, with the first element of the row circularly following the last. For example, ar is encrypted as RM.

3. Two plaintext letters that fall in the same column are each replaced by the letter beneath, with the top element of the column circularly following the last. For example, mu is encrypted as CM.

4. Otherwise, each plaintext letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter. Thus, hsbecomes BP and ea becomes IM (or JM, as the encipherer wishes).

POLYALPHABETIC CIPHERS:

Key	Plaintext																								
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
b	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
c	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
d	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
e	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
f	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
g	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
h	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
i	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
j	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
k	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
l	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
m	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
n	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
o	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
p	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
r	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
s	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
t	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
u	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
v	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
w	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
x	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

To encrypt a message, a key is needed that is as long as the message. Usually, the key is a repeating keyword. For example, if the keyword is *deceptive,* the message "we are discovered save yourself" is encrypted as follows:

key: *deceptivedeceptivedeceptive*

plaintext: wearediscoveredsaveyourself

ciphertext: ZICVTWQNGRZGVTWAVZHCQYGLMGJ

TRANSPOSITION TECHNIQUES:

The simplest such cipher is the rail fence technique, in which the plaintext is written down as a sequence of diagonals and then read off as a sequence of rows. For example, to encipher the message "meet me after the toga party" with a rail fence of depth 2, we write the following:

mematrhtgpry

etefeteoaat

The encrypted message is

MEMATRHTGPRYETEFETEOAAT

STEGANOGRAPHY:

Character marking: Selected letters of printed or typewritten text are overwritten in pencil. The marks are ordinarily not visible unless the paper is held at an angle to bright light.

Invisible ink: A number of substances can be used for writing but leave no visible trace until heat or some chemical is applied to the paper.

Pin punctures: Small pin punctures on selected letters are ordinarily not visible unless the paper is held up in front of a light.

Typewriter correction ribbon: Used between lines typed with a black ribbon, the results of typing with the correction tape are visible only under a strong light.

Modern Block Ciphers:

- one of the most widely used types of cryptographic algorithms
- provide secrecy /authentication services
- focus on DES (Data Encryption Standard)
- to illustrate block cipher design principles

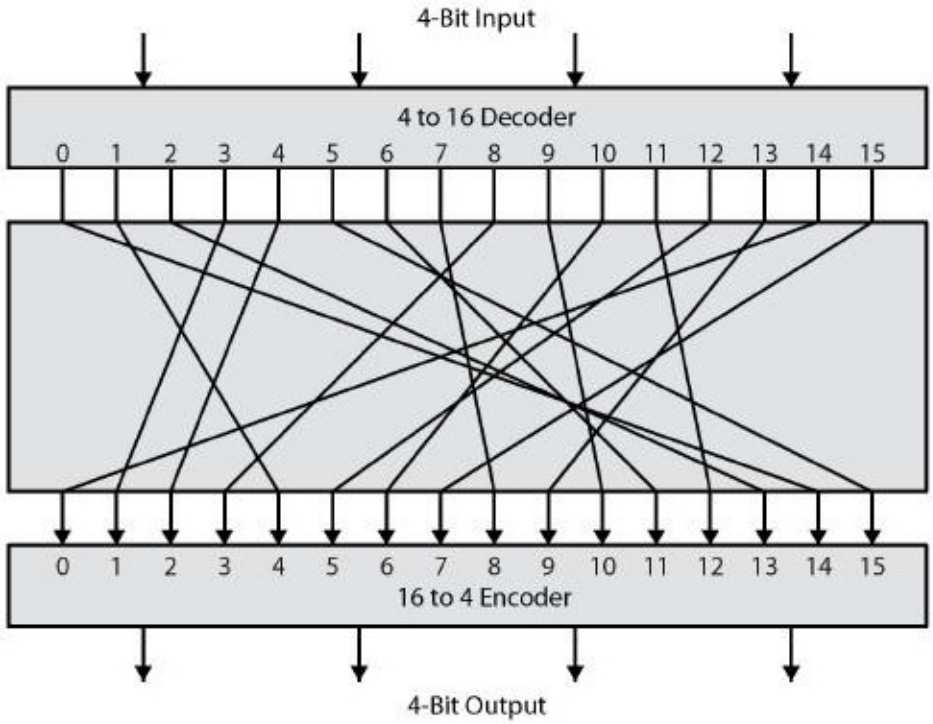
Block vs Stream Ciphers:

- block ciphers process messages in blocks, each of which is then en/decrypted
- like a substitution on very big characters
    - 64-bits or more
- stream ciphers process messages a bit or byte at a time when en/decrypting
- many current ciphers are block ciphers
- broader range of applications

Block Cipher Principles:

- most symmetric block ciphers are based on a Feistel Cipher Structure
- needed since must be able to decrypt ciphertext to recover messages efficiently
- block ciphers look like an extremely large substitution
- would need table of 2_{64} entries for a 64-bit block
- instead create from smaller building blocks
- using idea of a product cipher

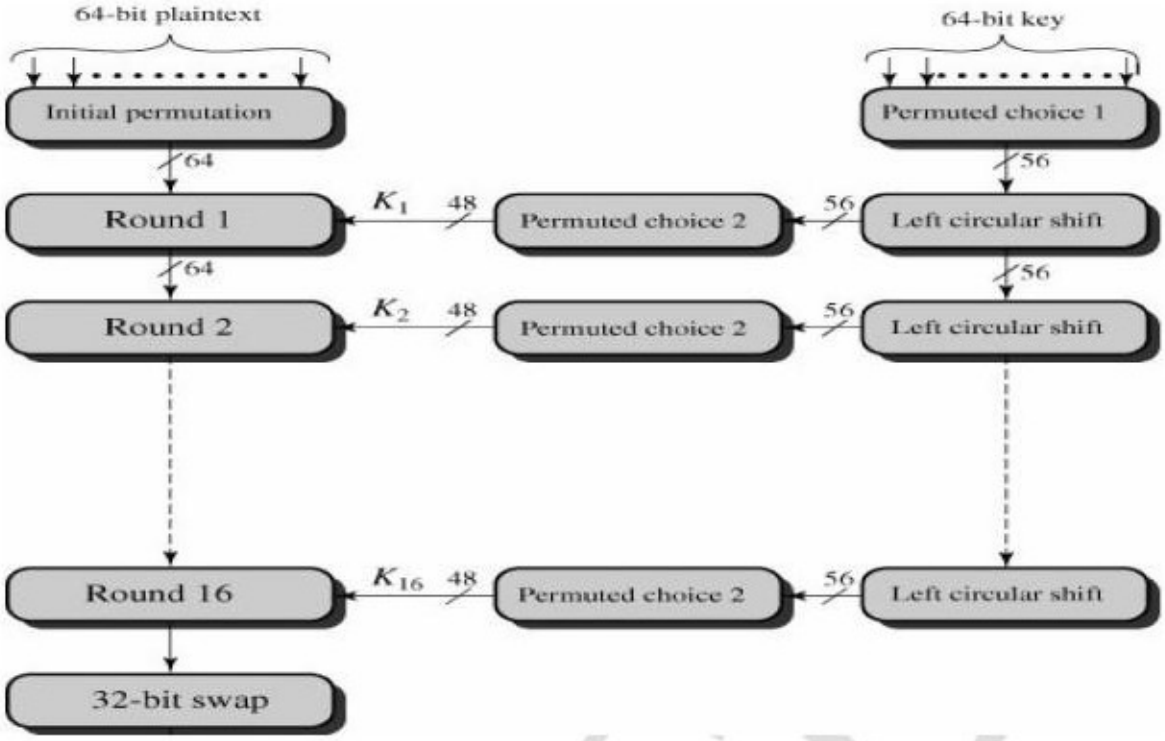
Ideal Block Cipher:



THE 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). For DES, data are encrypted in 64-bit blocks using a 56-bit key. The algorithm transforms 64-bit input in a series of steps into a 64-bit output. The same steps, with the same key, are used to reverse the encryption.

DES Encryption:



INITIAL PERMUTATION:

It is the first step of the data computation . IP reorders the input data bits. It changeeven bits to LH half, odd bits to RH half.

Example:

IP(675a6967 5e5a6b5a) = (ffb2194d 004df6fb)

DES Round Structure:

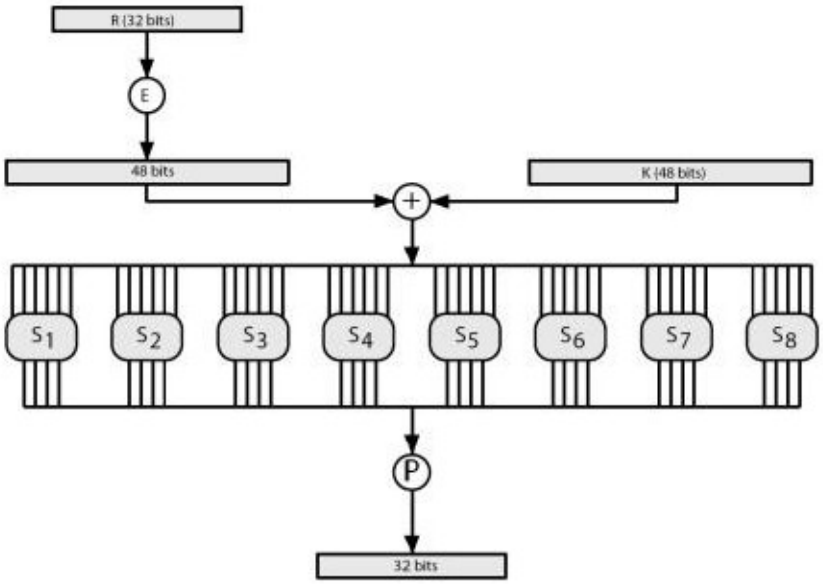
It uses two 32-bit L & R halves as for any Feistel cipher can describe as:

Li = Ri–1

Ri= Li–1 Å F(Ri–1 , *Ki)*

- F takes 32-bit R half and 48-bit subkey:
- expands R to 48-bits using perm E
- adds to subkey using XOR
- passes through 8 S-boxes to get 32-bit result
- finally permutes using 32-bit perm P

DES Round Structure:



Substitution Boxes S:

DES have eight S-boxes which map 6 to 4 bits. Each S-box is actually 4 little 4 bit boxes. Outer bits 1 & 6 (row bits) select one row of 4. Inner bits 2-5 (col bits) are substituted. Result is 8 lots of 4 bits, or 32 bits. Row selection depends on both data & key. Feature known as autoclaving (autokeying).

example:

S(18 09 12 3d 11 17 38 39) = 5fd25e03

DES Key Schedule:

- forms subkeys used in each round
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - rotating each half separately either 1 or 2 places depending on the key rotation schedule K
    - selecting 24-bits from each half & permuting them by PC2 for use in round function F
- note practical use issues in hardwarevs software

DES Decryption:

- decrypt must unwind steps of data computation
- with Feistel design, do encryption steps again using subkeys in reverse order (SK16 … SK1)

- IP undoes final FP step of encryption
- 1st round with SK16 undoes 16th encrypt round
- ….
- 16th round with SK1 undoes 1st encrypt round
- then final FP undoes initial encryption IP
- thus recovering original data value

Strength of DES – Analytic Attacks:

- now have several analytic attacks on DES
- these utilise some deep structure of the cipher
  - by gathering information about encryptions
  - can eventually recover some/all of the sub-key bits
  - if necessary then exhaustively search for the rest
- generally these are statistical attacks
- include
  - differential cryptanalysis
  - linear cryptanalysis
  - related key attacks

ADVANCED ENCRYPTION STANDARD (AES):

Origins:

- clear a replacement for DES was needed
  - have theoretical attacks that can break it
  - have demonstrated exhaustive key search attacks
- can use Triple-DES – but slow, has small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug-99

- Rijndael was selected as the AES in Oct-2000
- issued as FIPS PUB 197 standard in Nov-2001

AES Requirements:

- private key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- stronger & faster than Triple-DES
- active life of 20-30 years (+ archival use)
- provide full specification & design details
- both C & Java implementations
- NIST have released all submissions & unclassified analyses

AES Evaluation Criteria:

- initial criteria:
    - security – effort for practical cryptanalysis
    - cost – in terms of computational efficiency
    - algorithm & implementation characteristics
- final criteria
    - general security
    - ease of software & hardware implementation
    - implementation attacks
    - flexibility (in en/decrypt, keying, other factors)

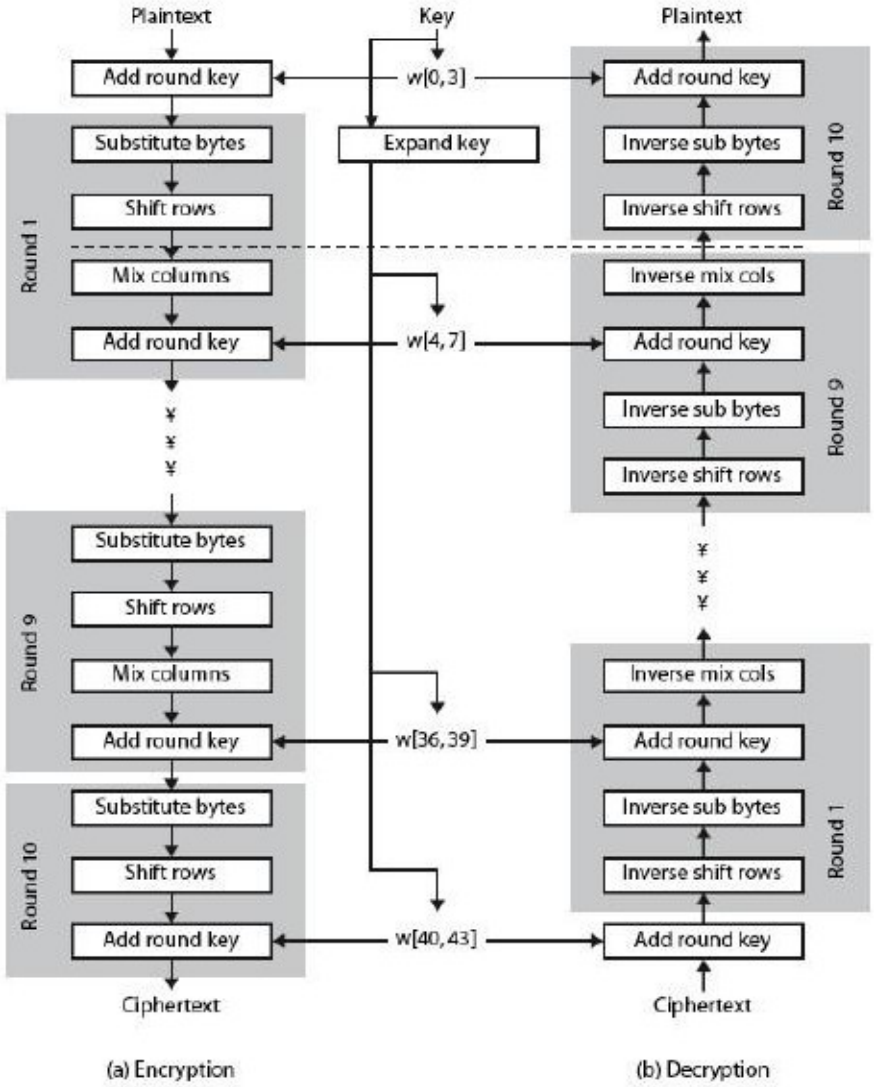
The AES Cipher – Rijndael

- designed by Rijmen-Daemen in Belgium
- has 128/192/256 bit keys, 128 bit data
- an iterative rather than feistel cipher
    - processes data as block of 4 columns of 4 bytes
    - operates on entire data block in every round
- designed to be:

- resistant against known attacks
- speed and code compactness on many CPUs
- design simplicity

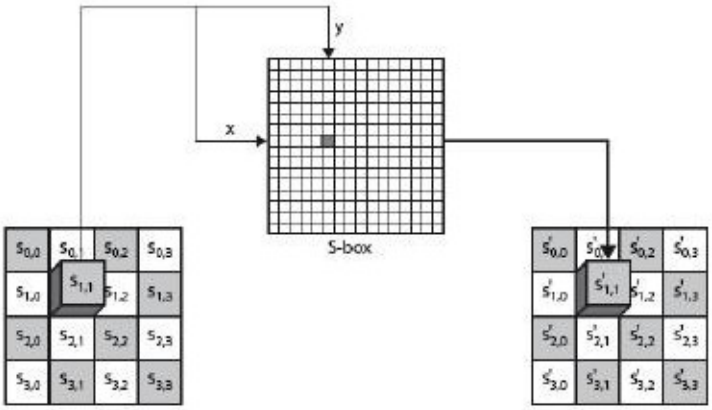
Rijndael:

- 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 multipy 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



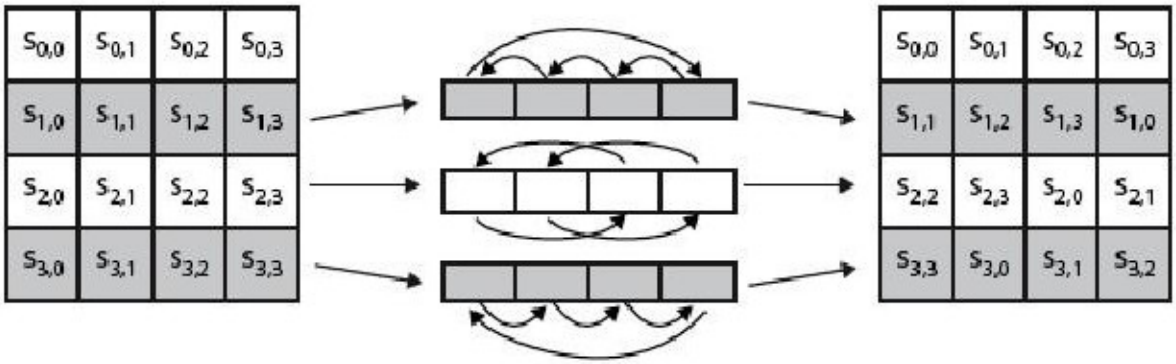
Byte Substitution:

- a simple substitution of each byte
- uses one table of 16x16 bytes containing a permutation of all 256 8-bit values
- each byte of state is replaced by byte indexed by row (left 4-bits) & column (right 4-bits)
  - eg. byte {95} is replaced by byte in row 9 column 5
  - which has value {2A}
- S-box constructed using defined transformation of values in GF(2^8)
- designed to be resistant to all known attacks



Shift Rows:

- a circular byte shift in each each
  - 1st row is unchanged
  - 2nd row does 1 byte circular shift to left
  - 3rd row does 2 byte circular shift to left
  - 4th row does 3 byte circular shift to left
- decrypt inverts using shifts to right
- since state is processed by columns, this step permutes bytes between the columns



Mix Columns:

- each column is processed separately
- each byte is replaced by a value dependent on all 4 bytes in the column
- effectively a matrix multiplication in GF(2^8) using prime poly $m(x) = x^8+x^4+x^3+x+1$

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

- can express each col as 4 equations
  - to derive each new byte in col
- decryption requires use of inverse matrix
  - with larger coefficients, hence a little harder
- have an alternate characterisation
  - each column a 4-term polynomial
  - with coefficients in GF(2^8)
  - and polynomials multiplied modulo (x^4+1)

Add Round Key:

Lastly is the Add Round Key stage which is a simple bitwise XOR of the current block with a portion of the expanded key. Note this is the only step which makes use of the key and obscures the result, hence MUST be used at start and end of each round, since otherwise could undo effect of other steps. But the other steps provide confusion/diffusion/non-linearity. That us you can look at the cipher as a series of XOR with key then scramble/permute block repeated. This is efficient and highly secure it is believed.

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

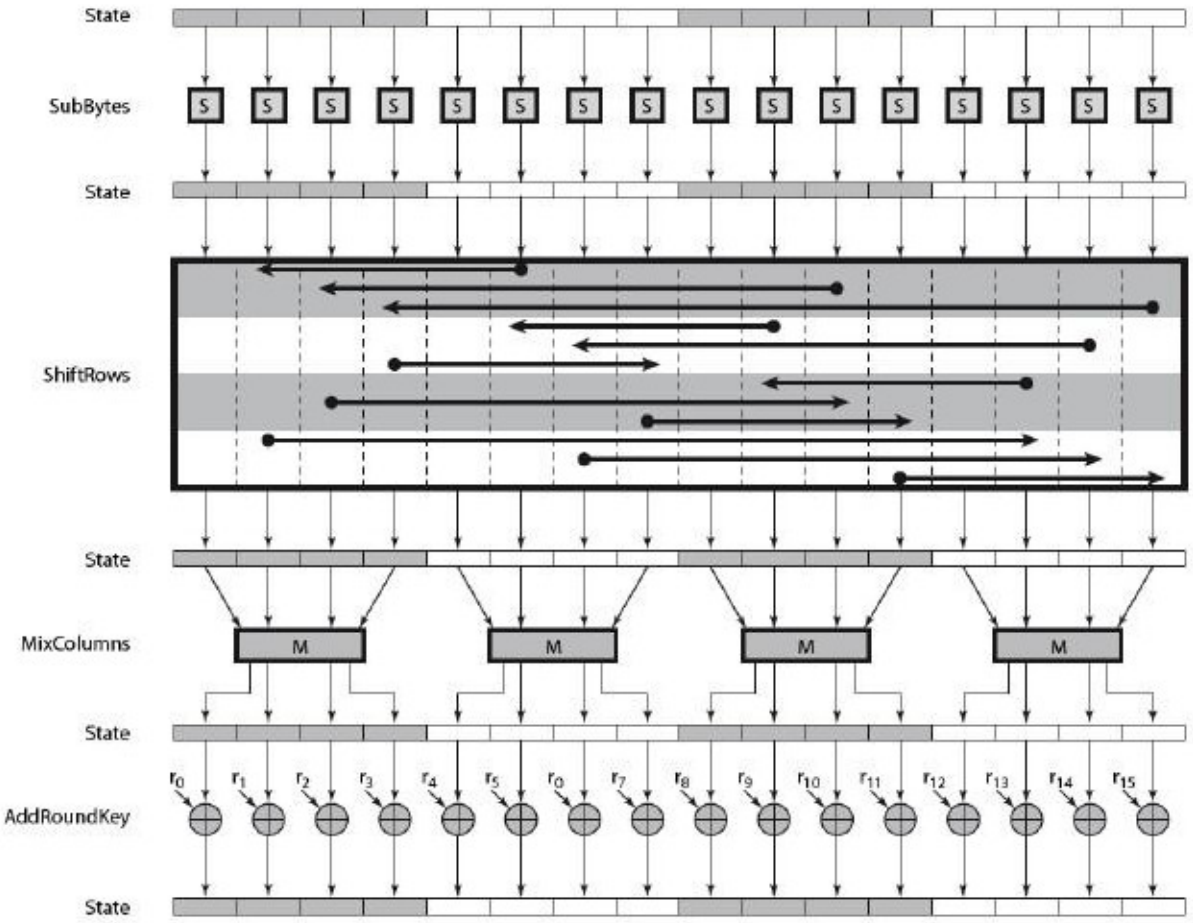
$\oplus$

w_i	w_{i+1}	w_{i+2}	w_{i+3}

$=$

$s'_{0,0}$	$s'_{0,1}$	$s'_{0,2}$	$s'_{0,3}$
$s'_{1,0}$	$s'_{1,1}$	$s'_{1,2}$	$s'_{1,3}$
$s'_{2,0}$	$s'_{2,1}$	$s'_{2,2}$	$s'_{2,3}$
$s'_{3,0}$	$s'_{3,1}$	$s'_{3,2}$	$s'_{3,3}$

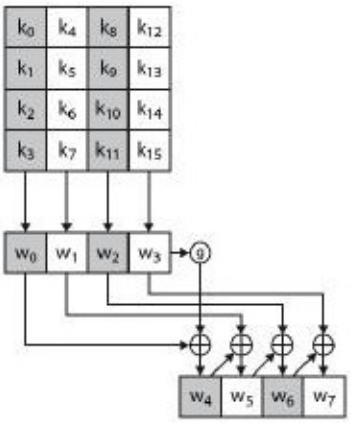
AES Round:



AES Key Expansion:

- takes 128-bit (16-byte) key and expands into array of 44/52/60 32-bit words
- start by copying key into first 4 words
- then loop creating words that depend on values in previous & 4 places back
  - in 3 of 4 cases just XOR these together
  - 1st word in 4 has rotate + S-box + XOR round constant on previous, before XOR 4th back

AES Key Expansion:



The first block of the AES Key Expansion is shown here in Figure. It shows each group of 4 bytes in the key being assigned to the first 4 words, then the calculation of the next 4 words based on the values of the previous 4 words, which is repeated enough times to create all the necessary subkey information.

Key Expansion Rationale:

- **designed to resist known attacks**
- **design criteria included**
  - **knowing part key insufficient to find many more**
  - **invertible transformation**
  - **fast on wide range of CPU's**
  - **use round constants to break symmetry**
  - **diffuse key bits into round keys**
  - **enough non-linearity to hinder analysis**
  - **simplicity of description**

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

Modes of Operation:

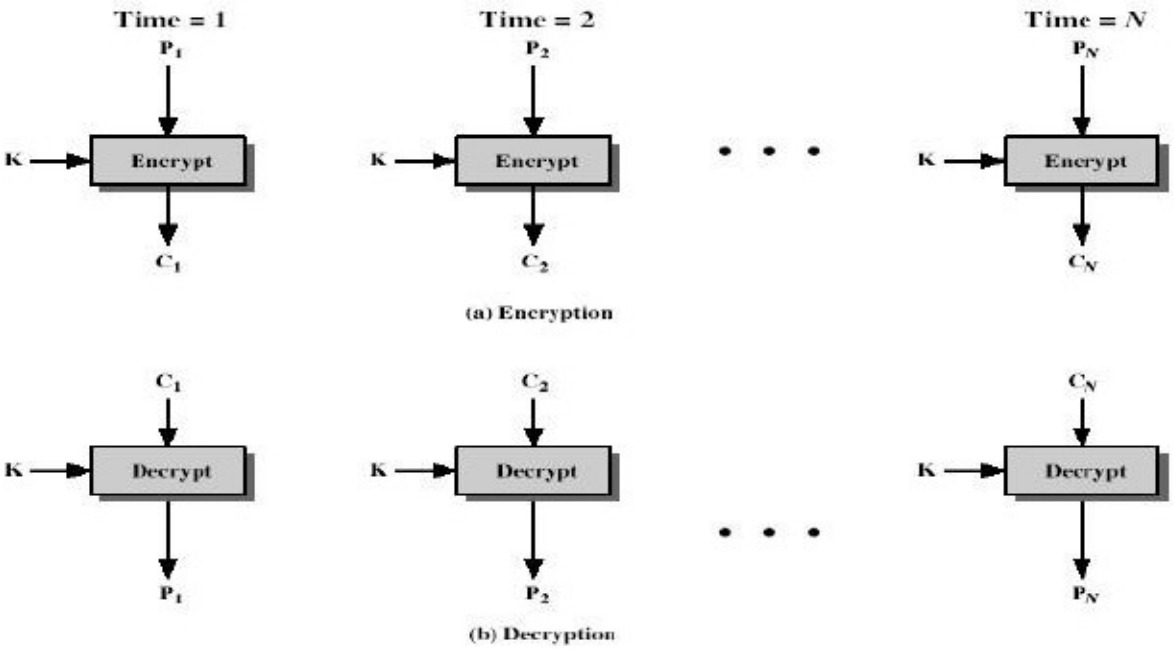
- block ciphers encrypt fixed size blocks
  - eg. DES encrypts 64-bit blocks with 56-bit key
- need some way to en/decrypt arbitrary amounts of data in practise
- **ANSI X3.106-1983 Modes of Use** (now FIPS 81)defines 4 possible modes
- subsequently 5 defined for AES & DES
- have **block** and **stream** modes

Electronic Codebook Book (ECB):

- message is broken into independent blocks which are encrypted
- each block is a value which is substituted, like a codebook, hence name
- each block is encoded independently of the other blocks

$C_i = DES_{K1}(P_i)$

- uses: secure transmission of single values



Advantages and Limitations of ECB:

- message repetitions may show in ciphertext
  - if aligned with message block
  - particularly with data such graphics
  - or with messages that change very little, which become a code-book analysis problem
- weakness is due to the encrypted message blocks being independent
- main use is sending a few blocks of data

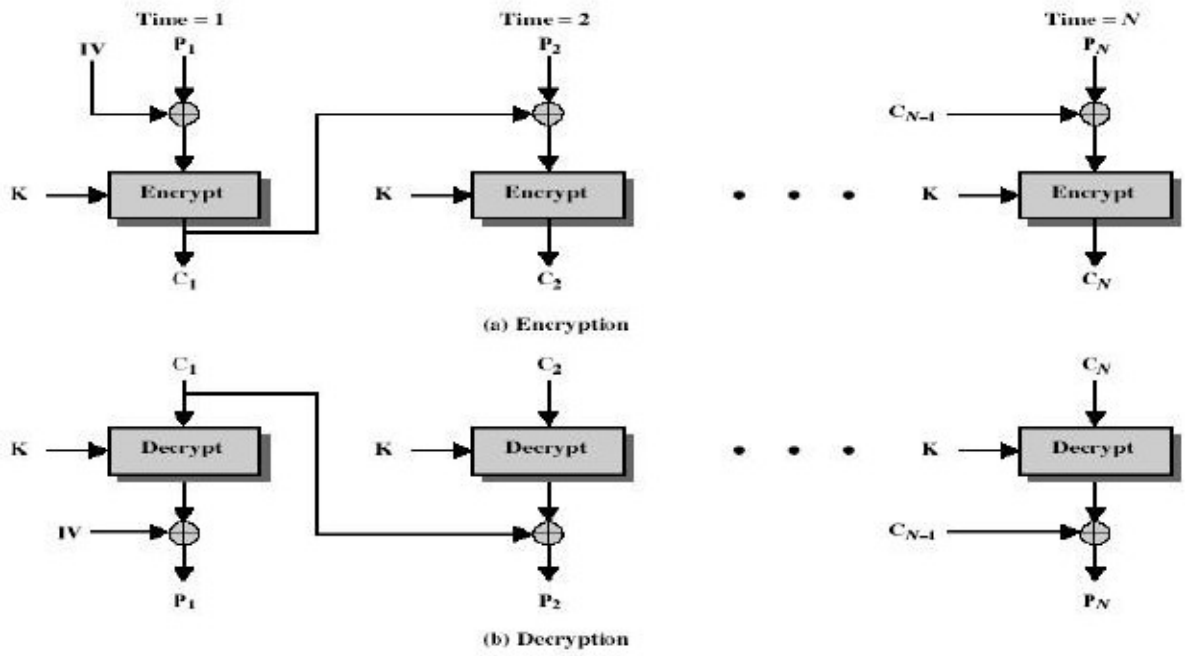
Cipher Block Chaining (CBC):

- message is broken into blocks
- linked together in encryption operation
- each previous cipher blocks is chained with current plaintext block, hence name
- use Initial Vector (IV) to start process

$C_i = DES_{K1}(P_i \text{ XOR } C_{i-1})$

$C_{-1} = IV$

- uses: bulk data encryption, authentication



Message Padding:

- at end of message must handle a possible last short block
  - which is not as large as blocksize of cipher
  - pad either with known non-data value (eg nulls)
  - or pad last block along with count of pad size
    - eg. [b1 b2 b3 0 0 0 0 5]
    - means have 3 data bytes, then 5 bytes pad+count
  - this may require an extra entire block over those in message
- there are other, more esoteric modes, which avoid the need for an extra block

Advantages and Limitations of CBC:

- **a ciphertext block depends on all blocks before it**
- **any change to a block affects all following ciphertext blocks**
- **need Initialization Vector (IV)**
  - **which must be known to sender & receiver**
  - **if sent in clear, attacker can change bits of first block, and change IV to compensate**
  - **hence IV must either be a fixed value (as in EFTPOS)**

- **or must be sent encrypted in ECB mode before rest of message**

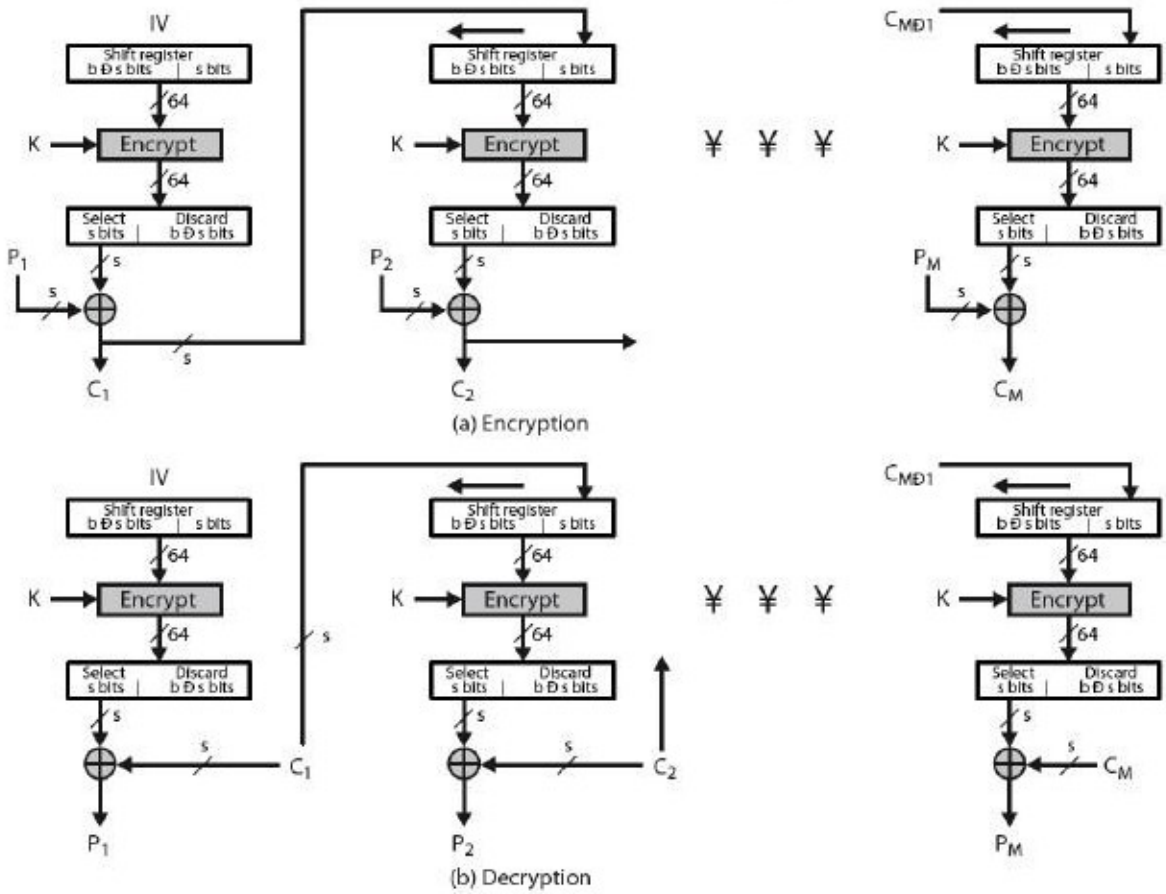
Cipher FeedBack (CFB):

- message is treated as a stream of bits
- added to the output of the block cipher
- result is feed back for next stage (hence name)
- standard allows any number of bit (1,8, 64 or 128 etc) to be feed back
  - denoted CFB-1, CFB-8, CFB-64, CFB-128 etc
- most efficient to use all bits in block (64 or 128)

$C_i = P_i \text{ XOR } DES_{K1}(C_{i-1})$

$C_{-1} = IV$

- uses: stream data encryption, authentication



Advantages and Limitations of CFB:

- appropriate when data arrives in bits/bytes
- most common stream mode

- limitation is need to stall while do block encryption after every n-bits
- note that the block cipher is used in **encryption** mode at **both** ends
- errors propogate for several blocks after the error

Output FeedBack (OFB):

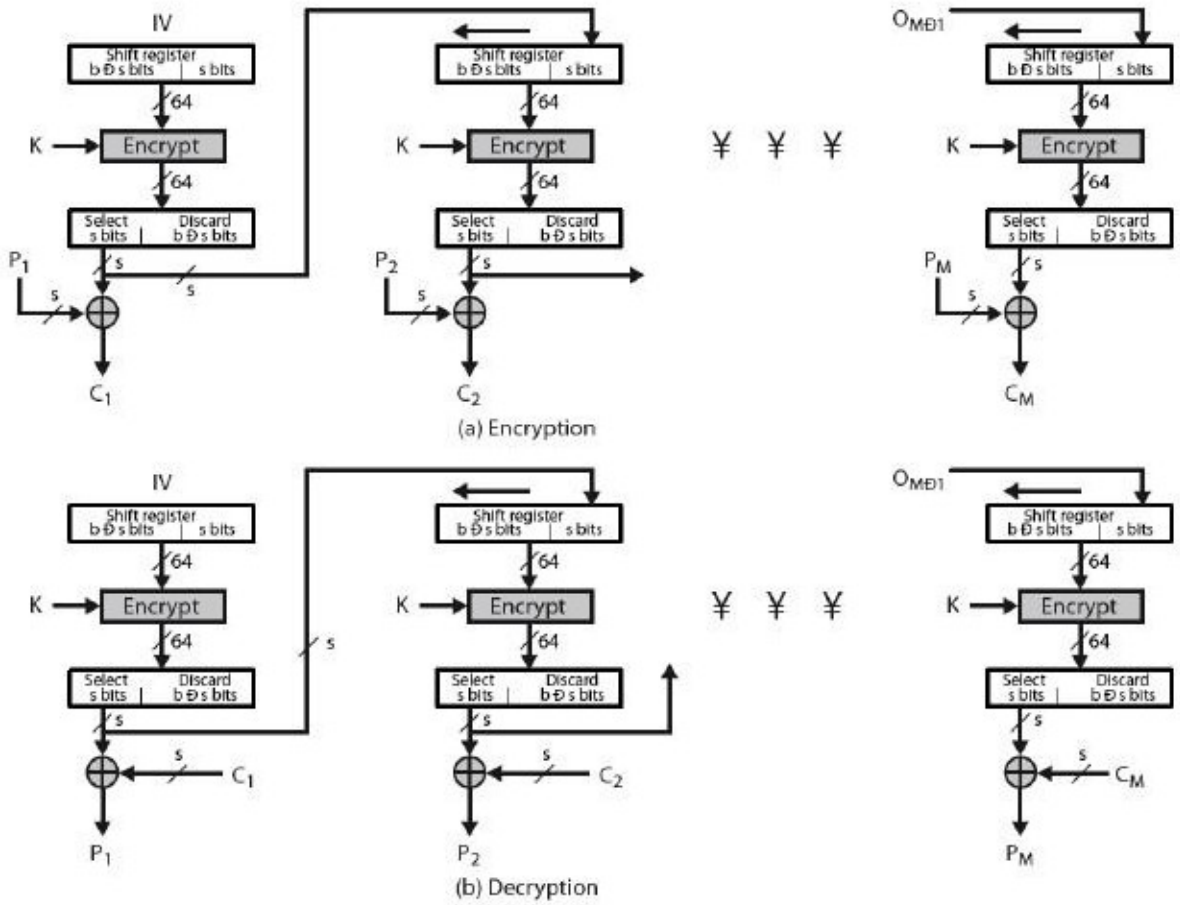
- message is treated as a stream of bits
- output of cipher is added to message
- output is then feed back (hence name)
- feedback is independent of message
- can be computed in advance

$C_i = P_i$ XOR O_i

$O_i = DES_{K1}(O_{i-1})$

$O_{-1} = IV$

- uses: stream encryption on noisy channels



Advantages and Limitations of OFB:

- bit errors do not propagate
- more vulnerable to message stream modification
- a variation of a Vernam cipher
  - hence must **never** reuse the same sequence (key+IV)
- sender & receiver must remain in sync
- originally specified with m-bit feedback
- subsequent research has shown that only **full block feedback** (ie CFB-64 or CFB-128) should ever be used

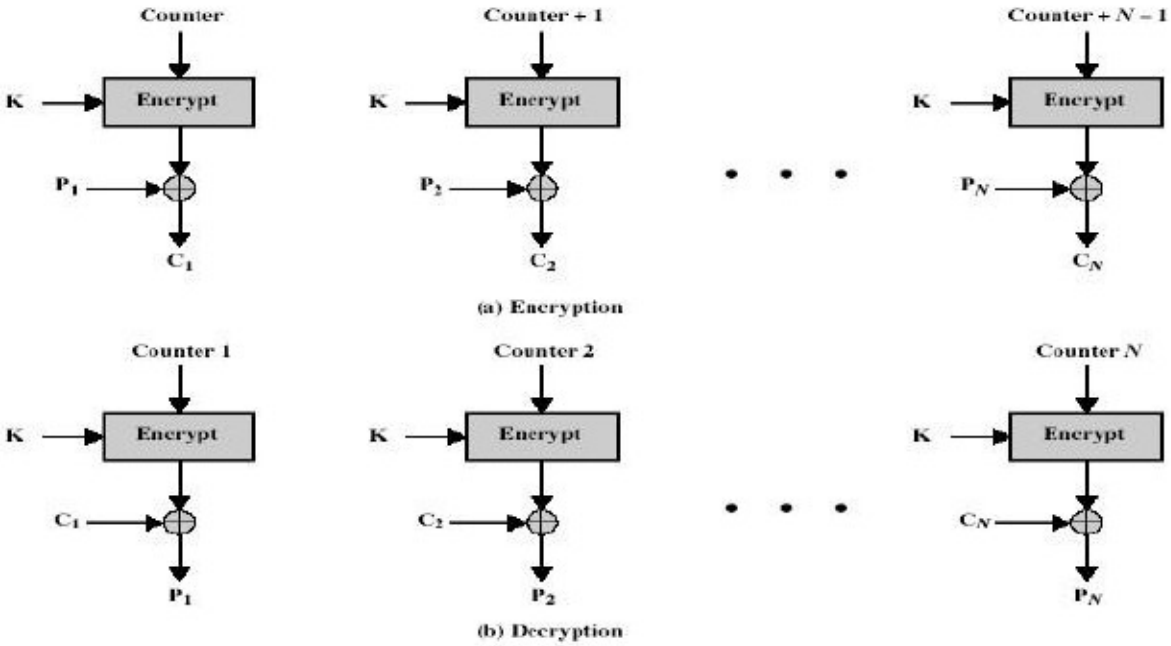
Counter (CTR):

- a "new" mode, though proposed early on
- similar to OFB but encrypts counter value rather than any feedback value
- must have a different key & counter value for every plaintext block (never reused)

$C_i = P_i$ XOR O_i

$O_i = DES_{K1}(i)$

- uses: high-speed network encryptions



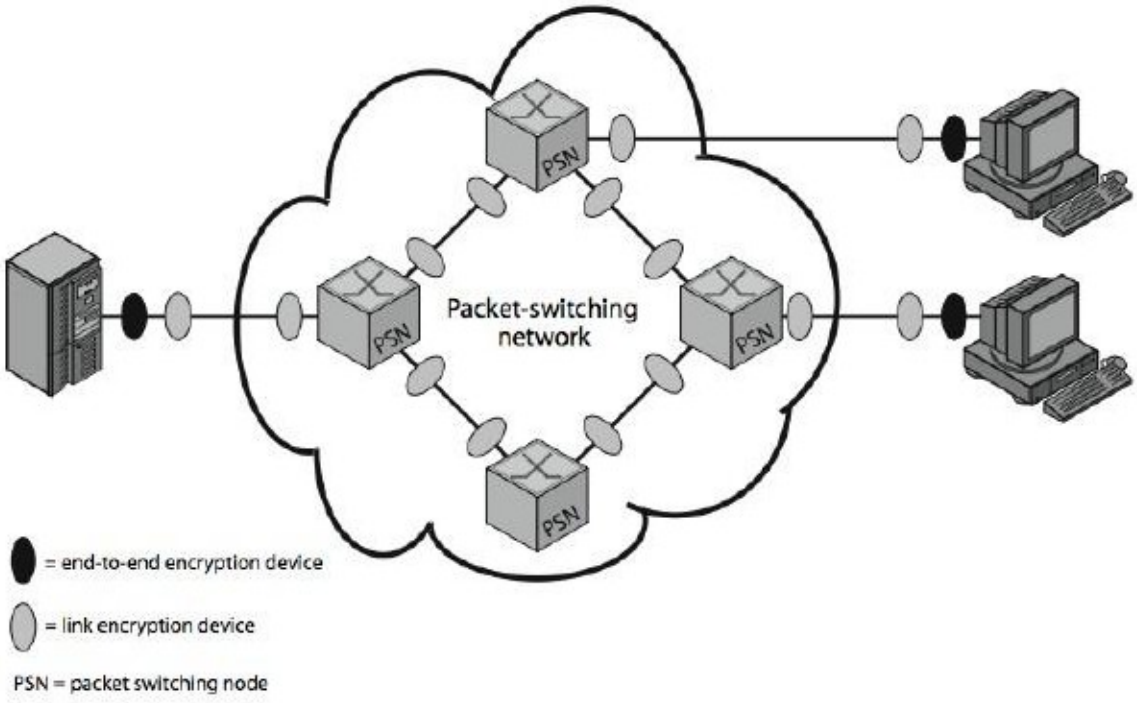
Advantages and Limitations of CTR:

- efficiency
  - can do parallel encryptions in h/w or s/w
  - can preprocess in advance of need
  - good for bursty high speed links
- random access to encrypted data blocks
- provable security (good as other modes)
- but must ensure never reuse key/counter values, otherwise could break (cf OFB)

PLACEMENT OF ENCRYPTION:

- have two major placement alternatives
- **link encryption**
  - encryption occurs independently on every link
  - implies must decrypt traffic between links
  - requires many devices, but paired keys
- **end-to-end encryption**

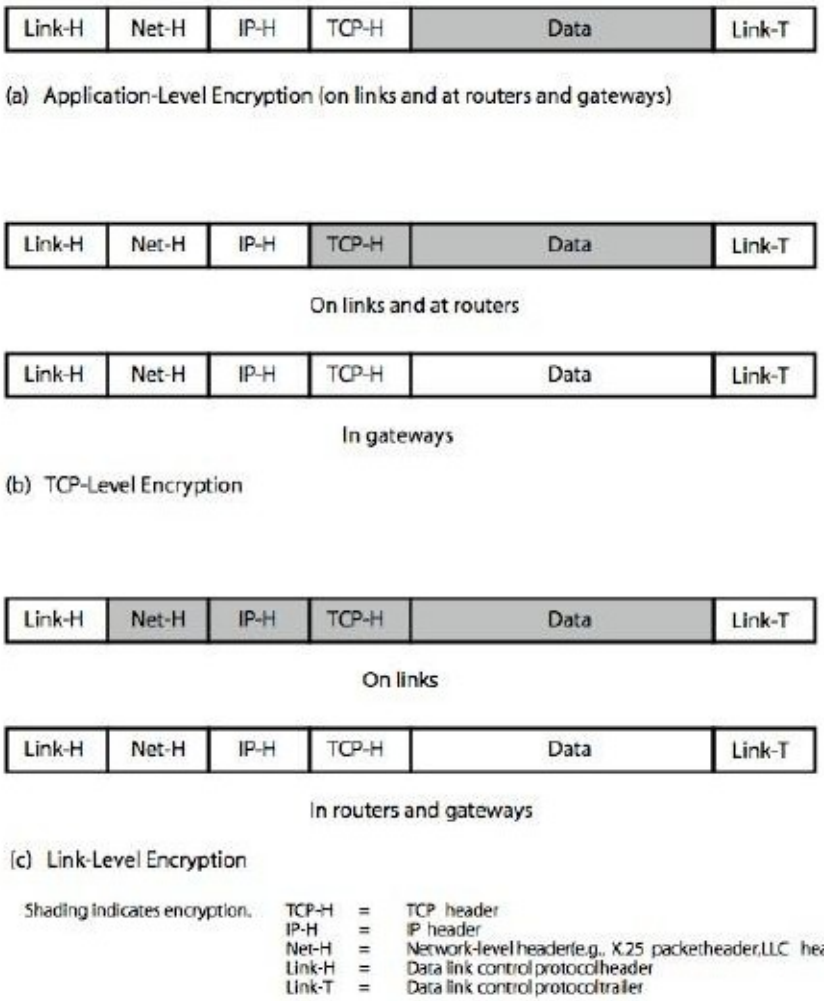
- encryption occurs between original source and final destination
- need devices at each end with shared keys



With end-to-end encryption, user data are secure, but the traffic pattern is not because packet headers are transmitted in the clear. However end-to-end encryption does provide a degree of authentication, since a recipient is assured that any message that it receives comes from the alleged sender, because only that sender shares the relevant key. Such authentication is not inherent in a link encryption scheme. To achieve greater security, both link and end-to-end encryption are needed, as is shown in Figure.

You can place encryption at any of a number of layers in the OSI Reference Model. Link encryption can occur at either the physical or link layers. End-to-end encryption could be performed at the network layer (for all processes on a system, perhaps in a Front End Processor), at the Transport layer (now possibly per process), or at the Presentation/Application layer (especially if need security to cross application gateways, but at cost of many more entities to manage). You can view alternatives noting that as you move up the communications hierarchy, less information is encrypted but it is more secure.

Encryption vs Protocol Level:



Traffic Analysis:

Some users are concerned with Traffic Analysis, which concerns knowledge about the number and length of messages between nodes which may enable an opponent to determine who is talking to whom, and hence suggest when important information is being exchanged, or to correlate with observed events.

With the use of link encryption, network-layer headers are encrypted, reducing the opportunity for traffic analysis. An effective countermeasure to this attack is traffic padding.

If only end-to-end encryption is employed, then the measures available to the defender are more limited since various protocol headers are visible. Padding of application data & null messages can be used.

Key Distribution:

For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others. This is one of the most critical areas in security

systems - on many occasions systems have been broken, not because of a poor encryption algorithm, but because of poor key selection or management. It is **absolutely critical** to get this right!

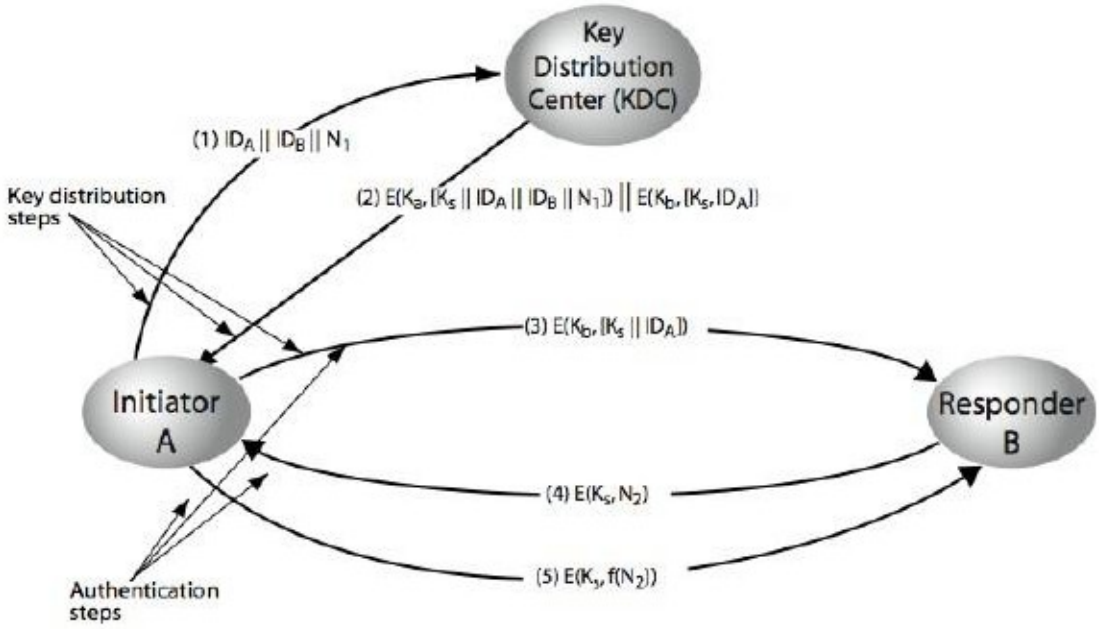
The strength of any cryptographic system thus depends on the key distribution technique. For two parties A and B, key distribution can be achieved in a number of ways:

Physical delivery (1 & 2) is simplest - but only applicable when there is personal contact between recipient and key issuer. This is fine for link encryption where devices & keys occur in pairs, but does not scale as number of parties who wish to communicate grows. 3 is mostly based on 1 or 2 occurring first.

A third party, whom all parties trust, can be used as a **trusted intermediary** to mediate the establishment of secure communications between them (4). Must trust intermediary not to abuse the knowledge of all session keys.As number of parties grow, some variant of 4 is only practical solution to the huge growth in number of keys potentially needed.

Key Hierarchy:

- typically have a hierarchy of keys
- session key
    - temporary key
    - used for encryption of data between users
    - for one logical session then discarded
- master key
    - used to encrypt session keys
    - shared by user & key distribution center



- hierarchies of KDC's required for large networks, but must trust each other
- session key lifetimes should be limited for greater security
- use of automatic key distribution on behalf of users, but must trust system
- use of decentralized key distribution
- controlling key usage

Random Numbers:

- many uses of **random numbers** in cryptography
  - nonces in authentication protocols to prevent replay
  - session keys
  - public key generation
  - keystream for a one-time pad
- in all cases its critical that these values be
  - statistically random, uniform distribution, independent
  - unpredictability of future values from previous values

Pseudorandom Number Generators (PRNGs):

- often use deterministic algorithmic techniques to create “random numbers”
    - although are not truly random
    - can pass many tests of “randomness”
- known as “pseudorandom numbers”
- created by “Pseudorandom Number Generators (PRNGs)”

Linear Congruential Generator:

- common iterative technique using:

$X_{n+1} = (aX_n + c) \bmod m$

- given suitable values of parameters can produce a long random-like sequence
- suitable criteria to have are:
    - function generates a full-period
    - generated sequence should appear random
    - efficient implementation with 32-bit arithmetic
- note that an attacker can reconstruct sequence given a small number of values
- have possibilities for making this harder

Using Block Ciphers as PRNGs:

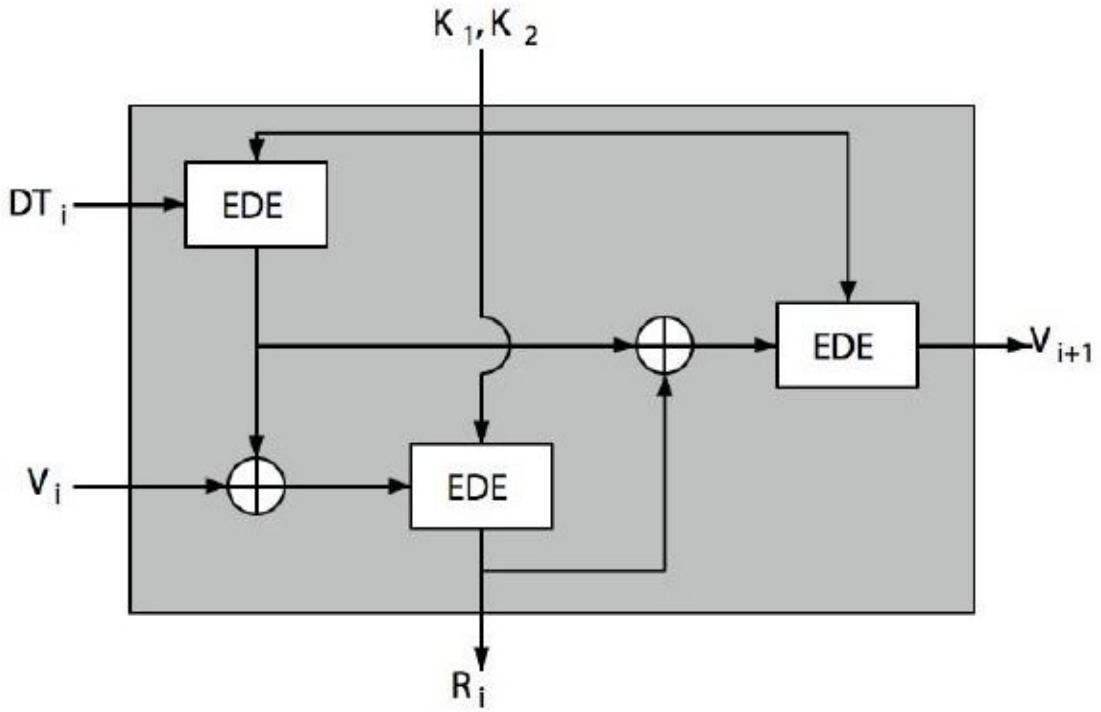
- for cryptographic applications, can use a block cipher to generate random numbers
- often for creating session keys from master key
- Counter Mode

$X_i = E_{Km}[i]$

- Output Feedback Mode

$X_i = E_{Km}[X_{i-1}]$

ANSI X9.17 PRG:



One of the strongest (cryptographically speaking) PRNGs is specified in ANSI X9.17. It uses date/time & seed inputs and 3 triple-DES encryptions to generate a new seed & random value. See discussion & illustration in Stallings section 7.4 & Figure 7.14 where:

DTi - Date/time value at the beginning of ith generation stage

Vi - Seed value at the beginning of ith generation stage

Ri - Pseudorandom number produced by the ith generation stage

K1, K2 - DES keys used for each stage

Then compute successive values as:

Ri = EDE([K1, K2], [Vi XOR EDE([K1, K2], DTi)])

Vi+1 = EDE([K1, K2], [Ri XOR EDE([K1, K2], DTi)])

Several factors contribute to the cryptographic strength of this method. The technique involves a 112-bit key and three EDE encryptions for a total of nine DES encryptions. The scheme is driven by two pseudorandom inputs, the date and time value, and a seed produced by the generator that is distinct from the pseudo-random number produced by the generator. Thus the amount of material that must be compromised by an opponent is overwhelming.

Blum BlumShub Generator:

- based on public key algorithms
- use least significant bit from iterative equation:

- $x_i = x_{i-12} \bmod n$
- where n=p.q, and primes p,q=3 mod 4

- unpredictable, passes **next-bit** test
- security rests on difficulty of factoring N
- is unpredictable given any run of bits
- slow, since very large numbers must be used
- too slow for cipher use, good for key generation

Question Bank

PART-A (2 MARKS)

1. Specify the four categories of security threads?
2. Explain active and passive attack with example?
3. Define integrity and nonrepudiation?
4. Differentiate symmetric and asymmetric encryption?
5. Define cryptanalysis?
6. Compare stream cipher with block cipher with example.
7. Define security mechanism
8. Differentiate unconditionally secured and computationally secured
9. Define steganography
10. Why network need security?
11. Define Encryption
12. Specify the components of encryption algorithm.
13. Define confidentiality and authentication
14. Define cryptography.
15. Compare Substitution and Transposition techniques.
16. Define Diffusion & confusion.
17. What are the design parameters of Feistel cipher network?
18. Define Product cipher.
19. Explain Avalanche effect.
20. Give the five modes of operation of Block cipher.

PART –B (16 MARKS)

1. a) Explain Playfair cipher &Vernam cipher in detail. (08)
b) Convert “MEET ME” using Hill cipher with the key matrix Convert the cipher text back to plaintext. (08)
2. Explain simplified DES with example. (16)
3. Write short notes on
4. a) Steganography (08)
b) Block cipher modes of operation (08)
5. Explain classical Encryption techniques in detail. (16)
6. Write short notes on
a) Security services (08)

b) Feistel cipher structure (08)
7. Explain Data Encryption Standard (DES) in detail. (16)
8. How AES is used for encryption/decryption? Discuss with example. (16)
9. List the evaluation criteria defined by NIST for AES. (16)

UNIT II PUBLIC KEY CRYPTOGRAPHY

Key management – Diffie Hellman key exchange – Elliptic curve architecture and cryptography – Introduction to number theory – Confidentiality using symmetric encryption – Public key cryptography and RSA.

KEY MANAGEMENT:

- public-key encryption helps address key distribution problems
- have two aspects of this:
  - distribution of public keys
  - use of public-key encryption to distribute secret keys

Distribution of Public Keys:

- can be considered as using one of:
  - public announcement
  - publicly available directory
  - public-key authority
  - public-key certificates

Public Announcement:

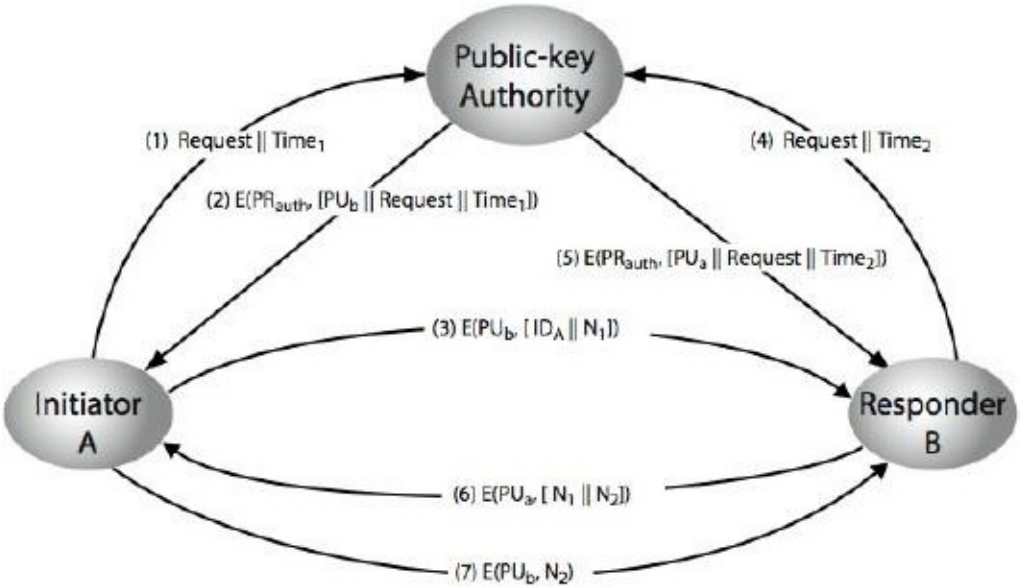
- users distribute public keys to recipients or broadcast to community at large
  - eg. append PGP keys to email messages or post to news groups or email list
- major weakness is forgery
  - anyone can create a key claiming to be someone else and broadcast it
  - until forgery is discovered can masquerade as claimed user

Publicly Available Directory:

- can obtain greater security by registering keys with a public directory
- directory must be trusted with properties:
  - contains {name,public-key} entries
  - participants register securely with directory
  - participants can replace key at any time
  - directory is periodically published
  - directory can be accessed electronically
- still vulnerable to tampering or forgery

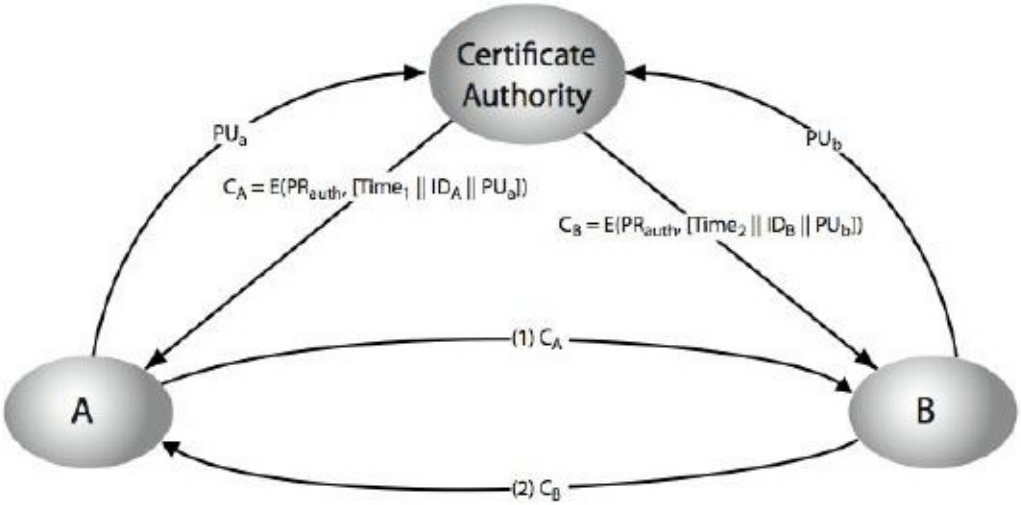
Public-Key Authority:

- improve security by tightening control over distribution of keys from directory
- has properties of directory
- and requires users to know public key for the directory
- then users interact with directory to obtain any desired public key securely
    - does require real-time access to directory when keys are needed



Public-Key Certificates:

- certificates allow key exchange without real-time access to public-key authority
- a certificate binds **identity** to **public key**
    - usually with other info such as period of validity, rights of use etc
- with all contents **signed** by a trusted Public-Key or Certificate Authority (CA)
- can be verified by anyone who knows the public-key authorities public-key



Public-Key Distribution of Secret Keys:

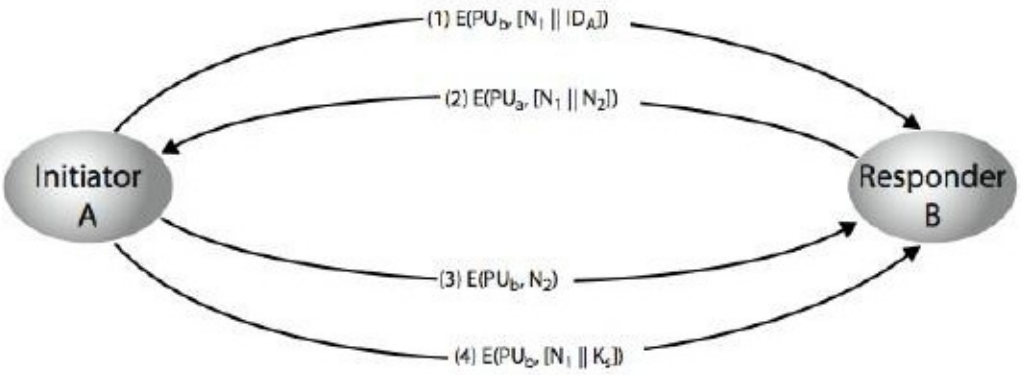
- use previous methods to obtain public-key
- can use for secrecy or authentication
- but public-key algorithms are slow
- so usually want to use private-key encryption to protect message contents
- hence need a session key
- have several alternatives for negotiating a suitable session

Simple Secret Key Distribution:

- proposed by Merkle in 1979
    - A generates a new temporary public key pair
    - A sends B the public key and their identity
    - B generates a session key K sends it to A encrypted using the supplied public key
    - A decrypts the session key and both use
- problem is that an opponent can intercept and impersonate both halves of protocol

Public-Key Distribution of Secret Keys:

- if have securely exchanged public-keys:



DIFFIE-HELLMAN KEY EXCHANGE:

- first public-key type scheme proposed
- by Diffie& Hellman in 1976 along with the exposition of public key concepts
    - note: now know that Williamson (UK CESG) secretly proposed the concept in 1970
- is a practical method for public exchange of a secret key
- used in a number of commercial products
- a public-key distribution scheme
    - cannot be used to exchange an arbitrary message
    - rather it can establish a common key
    - known only to the two participants
- value of key depends on the participants (and their private and public key information)
- based on exponentiation in a finite (Galois) field (modulo a prime or a polynomial) - easy
- security relies on the difficulty of computing discrete logarithms (similar to factoring) – hard

Diffie-Hellman Setup:

- all users agree on global parameters:
    - large prime integer or polynomial q

- a being a primitive root mod q

➢ each user (eg. A) generates their key

- chooses a secret key (number): $x_A < q$
- compute their **public key:** $y_A = a^{x_A} \bmod q$

➢ each user makes public that key y_A

➢ shared session key for users A & B is K_{AB}:

$K_{AB} = a^{x_A.x_B} \bmod q$

$= y_A^{x_B} \bmod q$ (which **B** can compute)

$= y_B^{x_A} \bmod q$ (which **A** can compute)

➢ K_{AB} is used as session key in private-key encryption scheme between Alice and Bob

➢ if Alice and Bob subsequently communicate, they will have the **same** key as before, unless they choose new public-keys

➢ attacker needs an x, must solve discrete log

Diffie-Hellman Example:

➢ users Alice & Bob who wish to swap keys:

➢ agree on prime q=353 and a=3

➢ select random secret keys:

- A chooses x_A=97, B chooses x_B=233

➢ compute respective public keys:

- $y_A = 3^{97} \bmod 353 = 40$ (Alice)
- $y_B = 3^{233} \bmod 353 = 248$ (Bob)

➢ compute shared session key as:

- $K_{AB} = y_B^{x_A} \bmod 353 = 248^{97} = 160$ (Alice)
- $K_{AB} = y_A^{x_B} \bmod 353 = 40^{233} = 160$ (Bob)

ELLIPTIC CURVE CRYPTOGRAPHY:

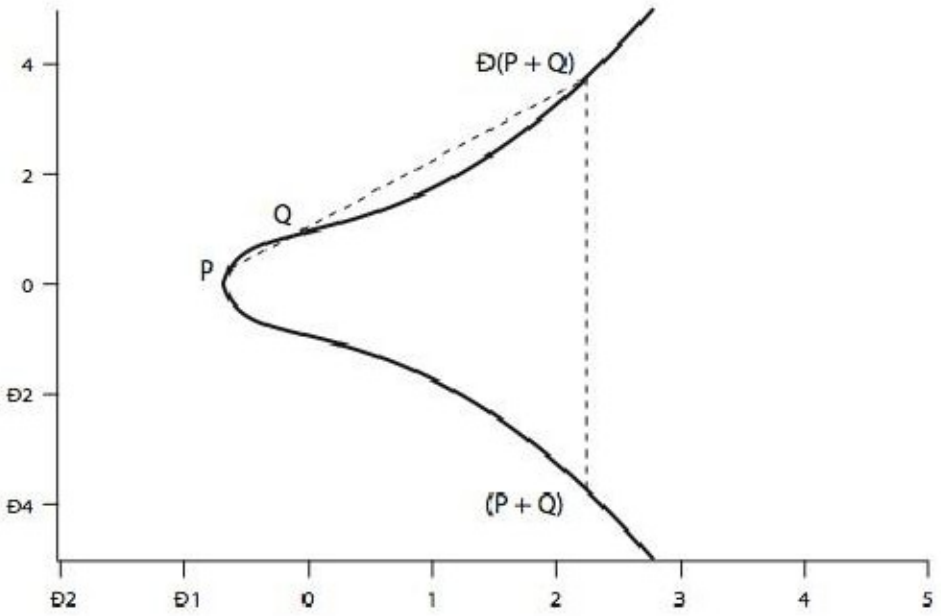
➢ majority of public-key crypto (RSA, D-H) use either integer or polynomial arithmetic with very large numbers/polynomials

➢ imposes a significant load in storing and processing keys and messages

- an alternative is to use elliptic curves
- offers same security with smaller bit sizes
- newer, but not as well analysed

Real Elliptic Curves:

- an elliptic curve is defined by an equation in two variables x & y, with coefficients
- consider a cubic elliptic curve of form
  - $y^2 = x^3 + ax + b$
  - where x,y,a,b are all real numbers
  - also define zero point O
- have addition operation for elliptic curve
  - geometrically sum of Q+R is reflection of intersection R



(b) $y^2 = x^3 + x + 1$

Finite Elliptic Curves:

- Elliptic curve cryptography uses curves whose variables & coefficients are finite
- have two families commonly used:
  - prime curves $E_p(a,b)$ defined over Z_p
    - use integers modulo a prime
    - best in software

- binary curves $E_{2m}(a,b)$ defined over $GF(2_n)$
    - use polynomials with binary coefficients
    - best in hardware

Elliptic Curve Cryptography:

- ECC addition is analog of modulo multiply
- ECC repeated addition is analog of modulo exponentiation
- need "hard" problem equiv to discrete log
    - Q=kP, where Q,P belong to a prime curve
    - is "easy" to compute Q given k,P
    - but "hard" to find k given Q,P
    - known as the elliptic curve logarithm problem
- Certicom example: $E_{23}(9,17)$

ECC Diffie-Hellman:

- can do key exchange analogous to D-H
- users select a suitable curve $E_p(a,b)$
- select base point $G=(x_1,y_1)$
    - with large order n s.t.nG=O
- A & B select private keys $n_A<n$, $n_B<n$
- compute public keys: $P_A=n_AG$, $P_B=n_BG$
- compute shared key: $K=n_AP_B$,$K=n_BP_A$
    - same since $K=n_An_BG$

ECC Encryption/Decryption:

- several alternatives, will consider simplest
- must first encode any message M as a point on the elliptic curve P_m
- select suitable curve & point G as in D-H
- each user chooses private key $n_A<n$
- and computes public key $P_A=n_AG$

- to encrypt P_m : $C_m=\{kG, P_m+kP_b\}$, k random
- decrypt C_m compute: P_m+kP_b-

$n_B(kG) = P_m+k(n_BG)-n_B(kG) = P_m$

ECC Security:

- relies on elliptic curve logarithm problem
- fastest method is "Pollard rho method"
- compared to factoring, can use much smaller key sizes than with RSA etc
- for equivalent key lengths computations are roughly equivalent
- hence for similar security ECC offers significant computational advantages

Comparable Key Sizes for Equivalent Security:

Symmetric scheme (key size in bits)	ECC-based scheme (size of n in bits)	RSA/DSA (modulus size in bits)
56	112	512
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	512	15360

NUMBER THEORY:

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

Prime Factorisation:

- to **factor** a number n is to write it as a product of other numbers: n=a x b x c
- note that factoring a number is relatively hard compared to multiplying the factors together to generate the number
- the **prime factorisation** of a number n is when its written as a product of primes
  - eg. $91=7\times13$; $3600=2^4\times3^2\times5^2$

$$a = \prod_{p \in \mathbb{P}} p^{a_p}$$

Relatively Prime Numbers & GCD:

- two numbers a, b are **relatively prime** if have **no common divisors** apart from 1
  - eg. 8 & 15 are relatively prime since factors of 8 are 1,2,4,8 and of 15 are 1,3,5,15 and 1 is the only common factor
- conversely can determine the greatest common divisor by comparing their prime factorizations and using least powers
  - eg. $300=2^1\times3^1\times5^2$ $18=2^1\times3^2$ hence $GCD(18,300)=2^1\times3^1\times5^0=6$

Fermat's Theorem:

- $a_{p-1} = 1 \pmod p$
  - where p is prime and gcd(a,p)=1
- also known as Fermat's Little Theorem
- also $a_p = p \pmod p$

- useful in public key and primality testing

Euler Totient Function ø(n):

- when doing arithmetic modulo n
- **complete set of residues** is: 0..n-1
- **reduced set of residues** is those numbers (residues) which are relatively prime to n
    - eg for n=10,
    - complete set of residues is {0,1,2,3,4,5,6,7,8,9}
    - reduced set of residues is {1,3,7,9}
- number of elements in reduced set of residues is called the **Euler Totient Function ø(n)**
- to compute ø(n) need to count number of residues to be excluded
- in general need prime factorization, but
    - for p (p prime) ø(p) = p-1
    - for p.q (p,q prime) ø(pq) =(p-1)x(q-1)
- eg.

ø(37) = 36

ø(21) = (3–1)x(7–1) = 2x6 = 12

Euler's Theorem:

- a generalisation of Fermat's Theorem
- $a^{ø(n)} = 1 \pmod n$
    - for any a,n where gcd(a,n)=1
- eg.

a=3;n=10; ø(10)=4;

hence $3^4 = 81 = 1 \bmod 10$

a=2;n=11; ø(11)=10;

hence $2^{10} = 1024 = 1 \bmod 11$

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

Miller Rabin Algorithm:

- a test based on Fermat's Theorem
- algorithm is:

TEST (n) is:

1. Find integers *k, q, k* > 0, *q* od'd, so that (n–1)=$2^k q$

2. Select a random integer *a,* 1<a<n–1

3. **if**$a_q$mod *n* = 1 **then** return ("maybe prime");

4. forj = 0 **to** *k* – 1 **do**

5. **if** ($a_{2^j q}$mod *n* = *n-1)*

thenreturn(" maybe prime ")

5. return ("composite")

Chinese Remainder Theorem:

- used to speed up modulo computations
- if working modulo a product of numbers
    - eg. mod M = $m_1 m_2 .. m_k$
- Chinese Remainder theorem lets us work in each moduli m_i separately
- since computational cost is proportional to size, this is faster than working in the full modulus M
- can implement CRT in several ways
- to compute A(mod M)

- first compute all $a_i = A \bmod m_i$ separately
- determine constants c_i below, where $M_i = M/m_i$
- then combine results to get answer using:

$$A \equiv \left(\sum_{i=1}^{k} a_i c_i \right) (\bmod M)$$

$$c_i = M_i \times (M_i^{-1} \bmod m_i) \quad \text{for } 1 \leq i \leq k$$

Private-Key Cryptography:

- traditional **private/secret/single key** cryptography uses **one** key
- shared by both sender and receiver
- if this key is disclosed communications are compromised
- also is **symmetric,** parties are equal
- hence does not protect sender from receiver forging a message & claiming is sent by sender
- probably most significant advance in the 3000 year history of cryptography
- uses **two** keys – a public & a private key
- **asymmetric** since parties are **not** equal
- uses clever application of number theoretic concepts to function
- complements **rather than** replaces private key crypto

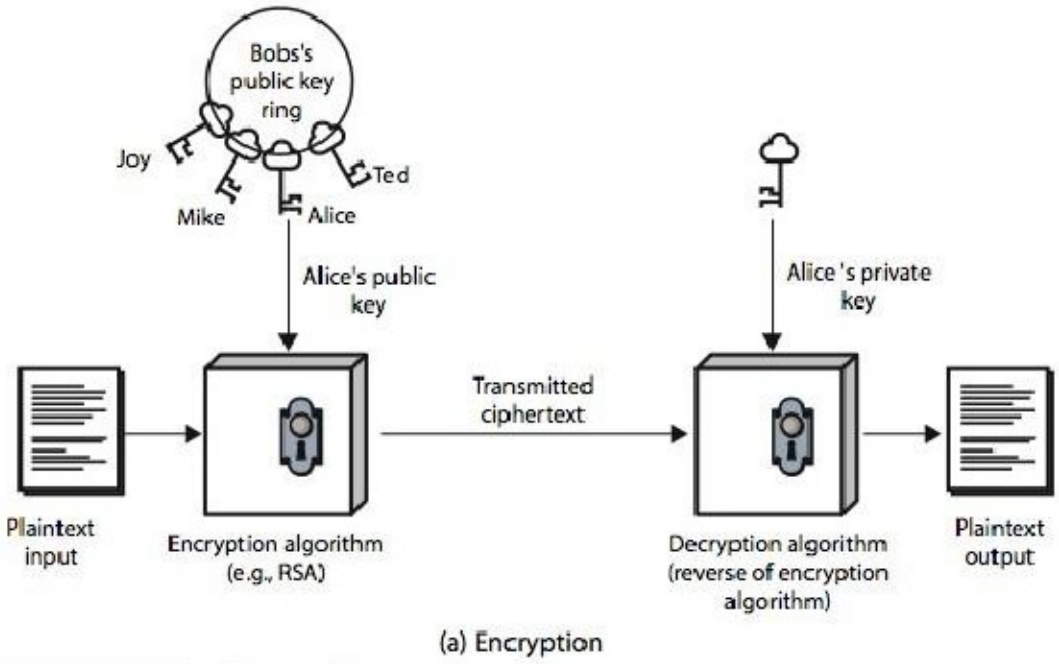
Why Public-Key Cryptography?:

- developed to address two key issues:
  - **key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **digital signatures** – how to verify a message comes intact from the claimed sender
- public invention due to Whitfield Diffie& Martin Hellman at Stanford Uni in 1976

- known earlier in classified community

Public-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**
- is **asymmetric** because
  - those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures

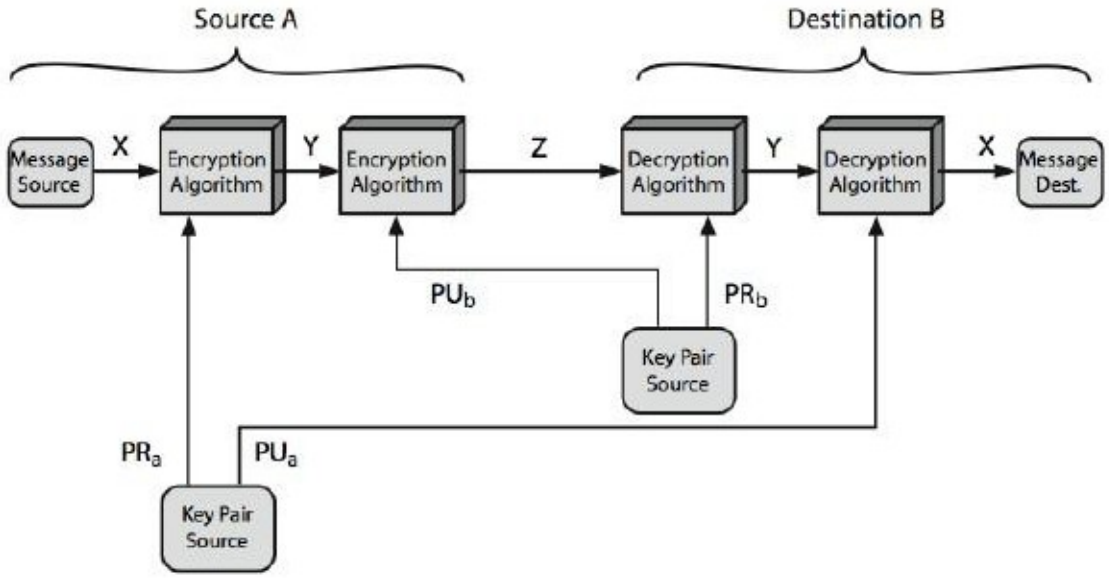


(a) Encryption

- **Public-Key Characteristics:**

Public-Key algorithms rely on two keys where:

- it is computationally infeasible to find decryption key knowing only algorithm & encryption key
- it is 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 (for some algorithms)



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, but is made hard enough to be impractical to break
- 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: PU={e,n}
- keep secret private decryption key: PR={d,n}

RSA Use:

- to encrypt a message M the sender:
  - obtains **public key** of recipient PU={e,n}
  - computes: $C = M^e \bmod n$, where 0≤M<n
- to decrypt the ciphertext C the owner:
  - uses their private key PR={d,n}
  - 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 chose e & d to be inverses mod ø(n)
  - hence e.d=1+k.ø(n) for some k

- hence :

$C_d = M_{e.d} = M_{1+k.\phi(n)} = M_1.(M_{\phi(n)})_k$

$= M_1.(1)_k = M_1 = M \bmod n$

RSA Example - Key Setup:

1. Select primes: *p=17* &q=11
2. Compute *n = pq=17* x 11=187
3. Compute ø(n)=(p–1)(q-1)=16 x 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 23x7=161= 10x160+1
6. Publish public key PU={7,187}
7. Keep secret private key PR={23,187}

RSA Example - En/Decryption:

- sample RSA encryption/decryption is:
- given message M = 88 (nb. 88<187)
- encryption:

$C = 88^7 \bmod 187 = 11$

- decryption:

$M = 11^{23} \bmod 187 = 88$

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. $3^{129} = 3^{128}.3^1 = 5.3 = 4 \bmod 11$

- c = 0; f = 1
- for i = k downto 0
- do c = 2 x c
- f = (f x f) mod n
- if b_i == 1 then
- c=c+1
- f = (f x a) mod n
- return f

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,qmust 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

RSA Security:

- possible approaches to attacking RSA are:
  - brute force key search (infeasible given size of numbers)
  - mathematical attacks (based on difficulty of computing ø(n), by factoring modulus n)
  - timing attacks (on running of decryption)
  - chosen ciphertext attacks (given properties of RSA)

QUESTION BANK

PART-A(2 MARKS)

1. Differentiate public key and conventional encryption?
2. What are the principle elements of a public key cryptosystem?
3. What are roles of public and private key?
4. Specify the applications of the public key cryptosystem?
5. What requirements must a public key cryptosystem to fulfill to a secured algorithm?
6. What is a one way function?
7. What is a trapdoor one way function?
8. Define Euler's theorem and it's application?
9. Define Euler's totient function or phi function and their applications?
10. Describe in general terms an efficient procedure for picking a prime number?
11. Define Fermat Theorem?
12. List four general characteristics of schema for the distribution of the public key?
13. What is a public key certificate?
14. What are essential ingredients of the public key directory?
15. Find gcd (1970, 1066) using Euclid's algorithm?
16. User A and B exchange the key using Diffie-Hellman algorithm. q=11 X_A=2 X_B=3. Find the value of Y_A, Y_B and k?
17. What is the primitive root of a number?
18. Determine the gcd (24140, 16762) using Euclid's algorithm.
19. Perform encryption and decryption using RSA Alg. for the following. P=7; q=11; e=17; M=8.
20. What is an elliptic curve?

PART –B (16 MARKS)

1. State and explain the principles of public key cryptography. (16)
2. Explain Diffie Hellman key Exchange in detail with an example (16)
3. Explain the key management of public key encryption in detail (16)
4. Explain RSA algorithm in detail with an example (16)
5. Briefly explain the idea behind Elliptic Curve Cryptosystem. (16)

UNIT III AUTHENTICATION AND HASH FUNCTION

Authentication requirements – Authentication functions – Message authentication codes – Hash functions – Security of hash functions and MACS – MD5 Message Digest algorithm – Secure hash algorithm – Ripend – HMAC digital signatures – Authentication protocols – Digital signature standard

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

Security Requirements:

- disclosure
- traffic analysis
- masquerade
- content modification
- sequence modification
- timing modification

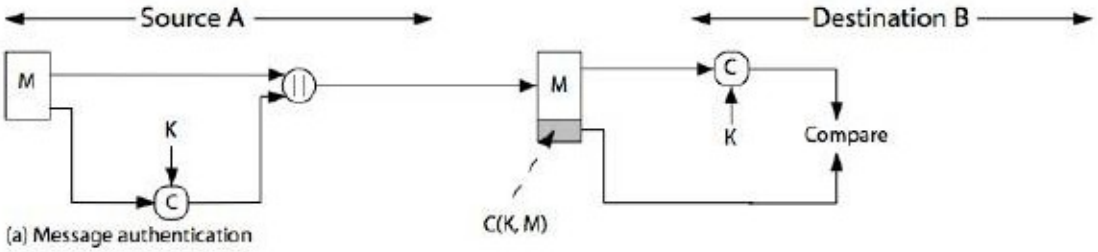
- source repudiation
- destination repudiation

Message Encryption:

- message encryption by itself also provides a measure of authentication
- if symmetric encryption is used then:
    - receiver know sender must have created it
    - since only sender and receiver now key used
    - know content cannot of been altered
    - if message has suitable structure, redundancy or a checksum to detect any changes
- if public-key encryption is used:
    - encryption provides no confidence of sender
    - since anyone potentially knows public-key
    - however if
        - sender **signs** message using their private-key
        - then encrypts with recipients public key
        - have both secrecy and authentication
    - again need to recognize corrupted messages
    - but at cost of two public-key uses on message

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



➢

as shown the MAC provides authentication

➢ can also use encryption for secrecy

- generally use separate keys for each
- can compute MAC either before or after encryption
- is generally regarded as better done before

➢ why use a MAC?

- sometimes only authentication is needed
- sometimes need authentication to persist longer than the encryption (eg. archival use)

➢ note that a MAC is not a digital signature

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:

1. knowing a message and MAC, is infeasible to find another message with same MAC

2. MACs should be uniformly distributed

3. 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 MAC
    - or the leftmost M bits (16≤M≤64) of final block
- but final MAC is now too small for security

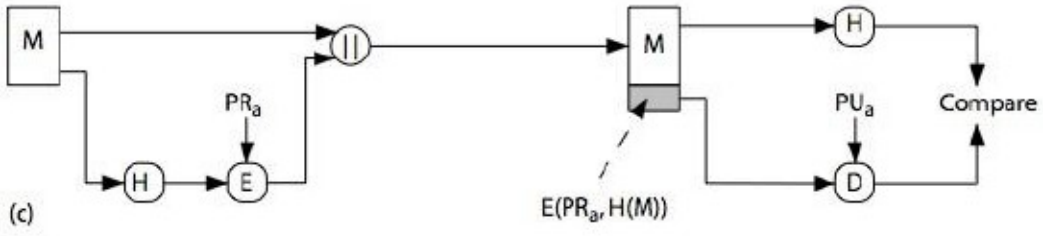
- can use any block cipher chaining mode and use final block as a MAC
- **Data Authentication Algorithm (DAA)** is a widely used MAC based on DES-CBC
  - using IV=0 and zero-pad of final block
  - encrypt message using DES in CBC mode
  - and send just the final block as the MAC
    - or the leftmost M bits (16≤M≤64) of final block
- but final MAC is now too small for security

Hash Functions:

- condenses arbitrary message to fixed size

h = H(M)

- 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



Requirements for Hash Functions:

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

Simple Hash Functions:

- are several proposals for simple functions
- based on XOR of message blocks
- not secure since can manipulate any message and either not change hash or change hash also
- need a stronger cryptographic function (next chapter)

Birthday Attacks:

- might think a 64-bit hash is secure
- but by **Birthday Paradox** is not
- **birthday attack** works thus:
    - opponent generates $2^{m/2}$ variations of a valid message all with essentially the same meaning
    - opponent also generates $2^{m/2}$ variations of a desired fraudulent message

- two sets of messages are compared to find pair with same hash (probability > 0.5 by birthday paradox)
- have user sign the valid message, then substitute the forgery which will have a valid signature

➢ conclusion is that need to use larger MAC/hash

Block Ciphers as Hash Functions:

➢ can use block ciphers as hash functions

- using $H_0=0$ and zero-pad of final block
- compute: $H_i = E_{M_i}[H_{i-1}]$
- and use final block as the hash value
- similar to CBC but without a key

➢ resulting hash is too small (64-bit)

- both due to direct birthday attack
- and to "meet-in-the-middle" attack

➢ other variants also susceptible to attack

Hash Functions & MAC Security:

➢ like block ciphers have:

➢ **brute-force** attacks exploiting

- strong collision resistance hash have cost $2^{m/2}$
  - have proposal for h/w MD5 cracker
  - 128-bit hash looks vulnerable, 160-bits better
- MACs with known message-MAC pairs
  - can either attack keyspace (cf key search) or MAC
  - at least 128-bit MAC is needed for security

➢ **cryptanalytic attacks** exploit structure

- like block ciphers want brute-force attacks to be the best alternative

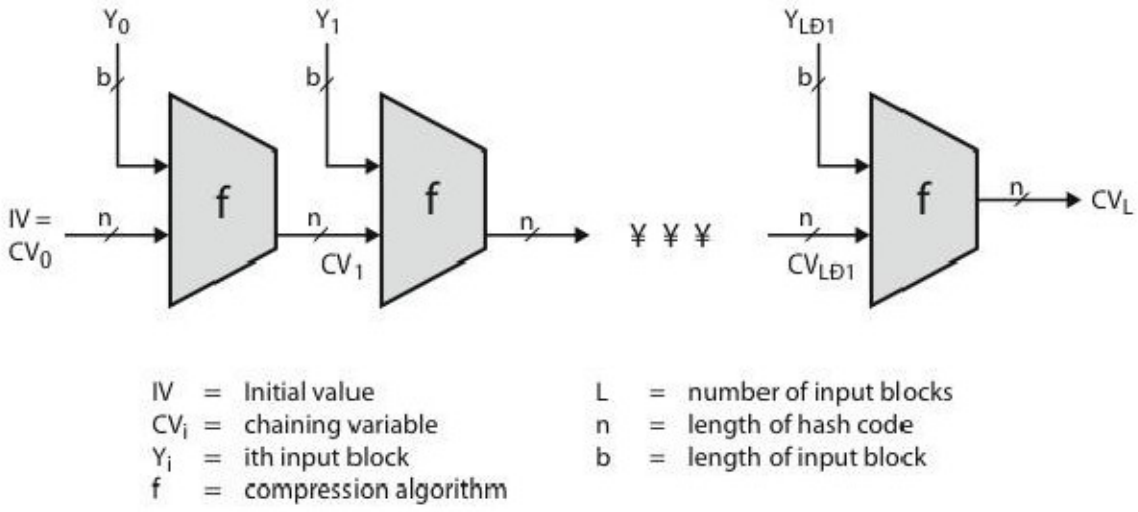
➢ have a number of analytic attacks on iterated hash functions

- $CV_i = f[CV_{i-1}, M_i]$; $H(M)=CV_N$

- typically focus on collisions in function f
- like block ciphers is often composed of rounds
- attacks exploit properties of round functions

Hash and MAC Algorithms:

- Hash Functions
  - condense arbitrary size message to fixed size
  - by processing message in blocks
  - through some compression function
  - either custom or block cipher based
- Message Authentication Code (MAC)
  - fixed sized authenticator for some message
  - to provide authentication for message
  - by using block cipher mode or hash function



Most important modern hash functions follow the basic structure shown in this figure. This has proved to be a fundamentally sound structure, and newer designs simply refine the structure and add to the hash code length. Within this basic structure, two approaches have been followed in the design of the compression function, as mentioned previously, which is the basic building block of the hash function.

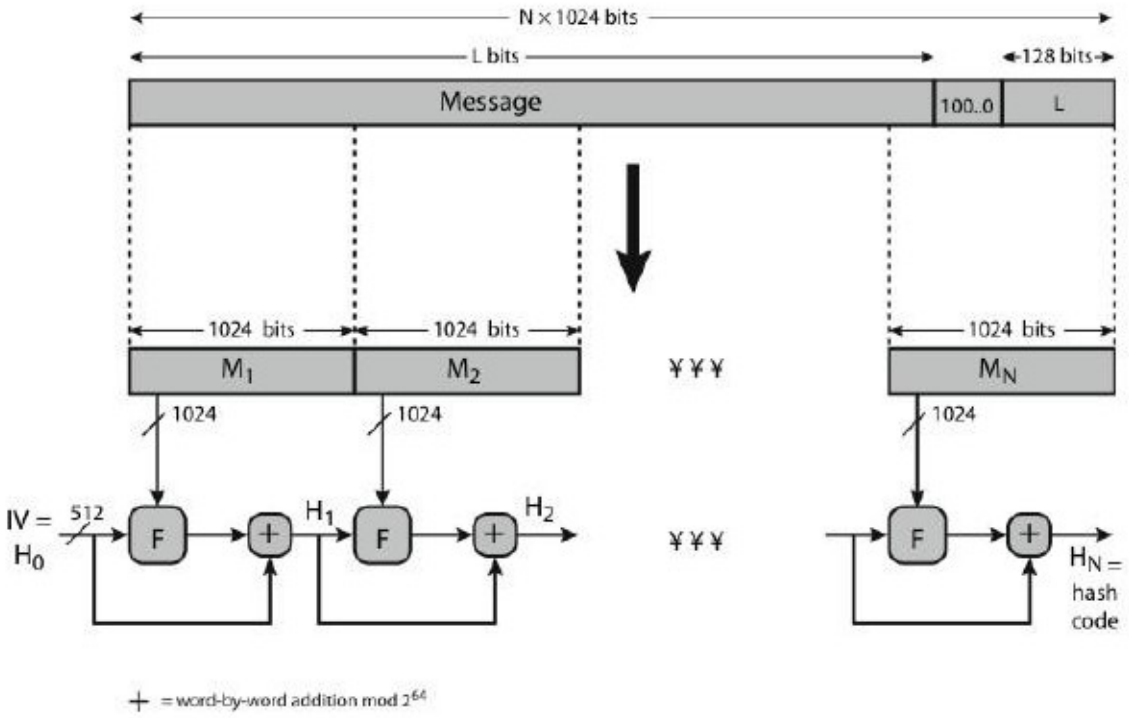
Secure Hash Algorithm:

- SHA originally designed by NIST & NSA in 1993
- was revised in 1995 as SHA-1
- US standard for use with DSA signature scheme
    - standard is FIPS 180-1 1995, also Internet RFC3174
    - nb. the algorithm is SHA, the standard is SHS
- based on design of MD4 with key differences
- produces 160-bit hash values
- recent 2005 results on security of SHA-1 have raised concerns on its use in future applications

Revised Secure Hash Standard:

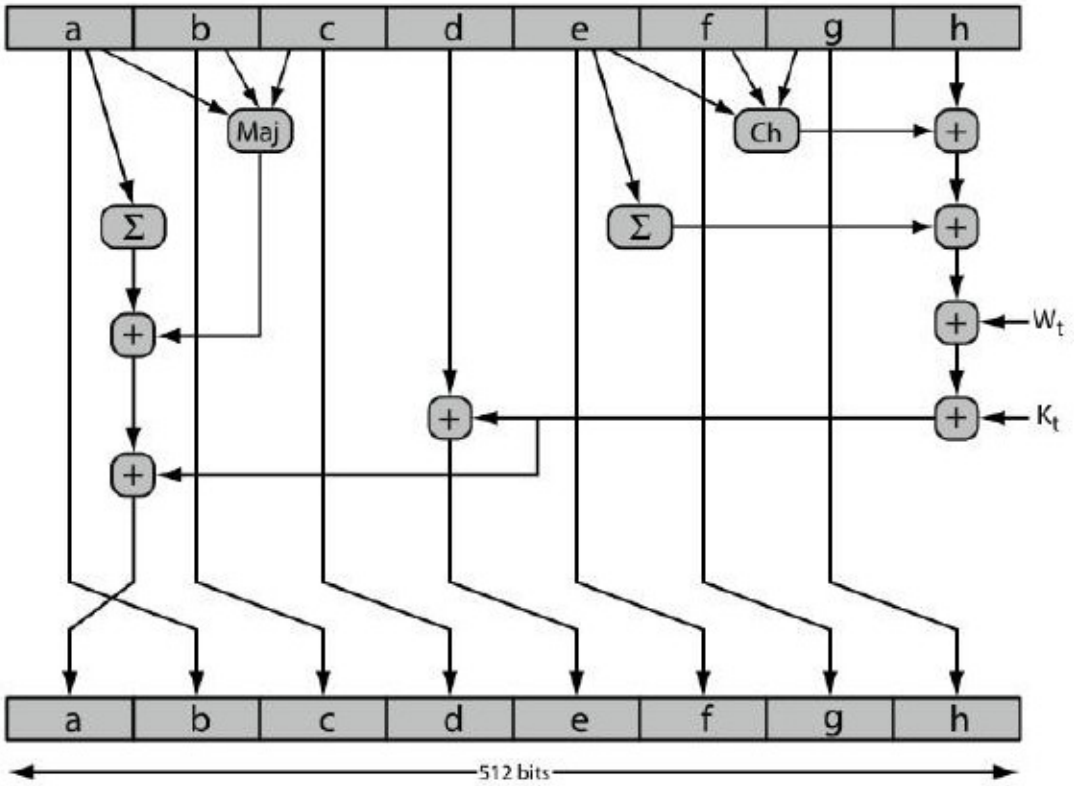
- NIST issued revision FIPS 180-2 in 2002
- adds 3 additional versions of SHA
    - SHA-256, SHA-384, SHA-512
- designed for compatibility with increased security provided by the AES cipher
- structure & detail is similar to SHA-1
- hence analysis should be similar
- but security levels are rather higher

SHA-512 Overview:

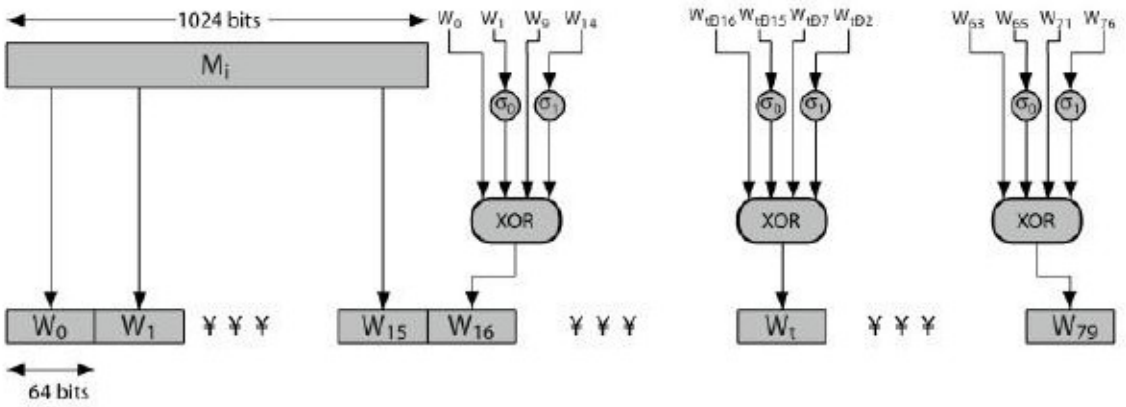


SHA-512 Compression Function:

- heart of the algorithm
- processing message in 1024-bit blocks
- consists of 80 rounds
  - updating a 512-bit buffer
  - using a 64-bit value Wt derived from the current message block
  - and a round constant based on cube root of first 80 prime numbers



SHA-512 Round Function:



Keyed Hash Functions as MACs:

- want a MAC based on a hash function
  - because hash functions are generally faster
  - code for crypto hash functions widely available
- hash includes a key along with message

- original proposal:

KeyedHash = Hash(Key|Message)

  - some weaknesses were found with this

- eventually led to development of HMAC

HMAC:

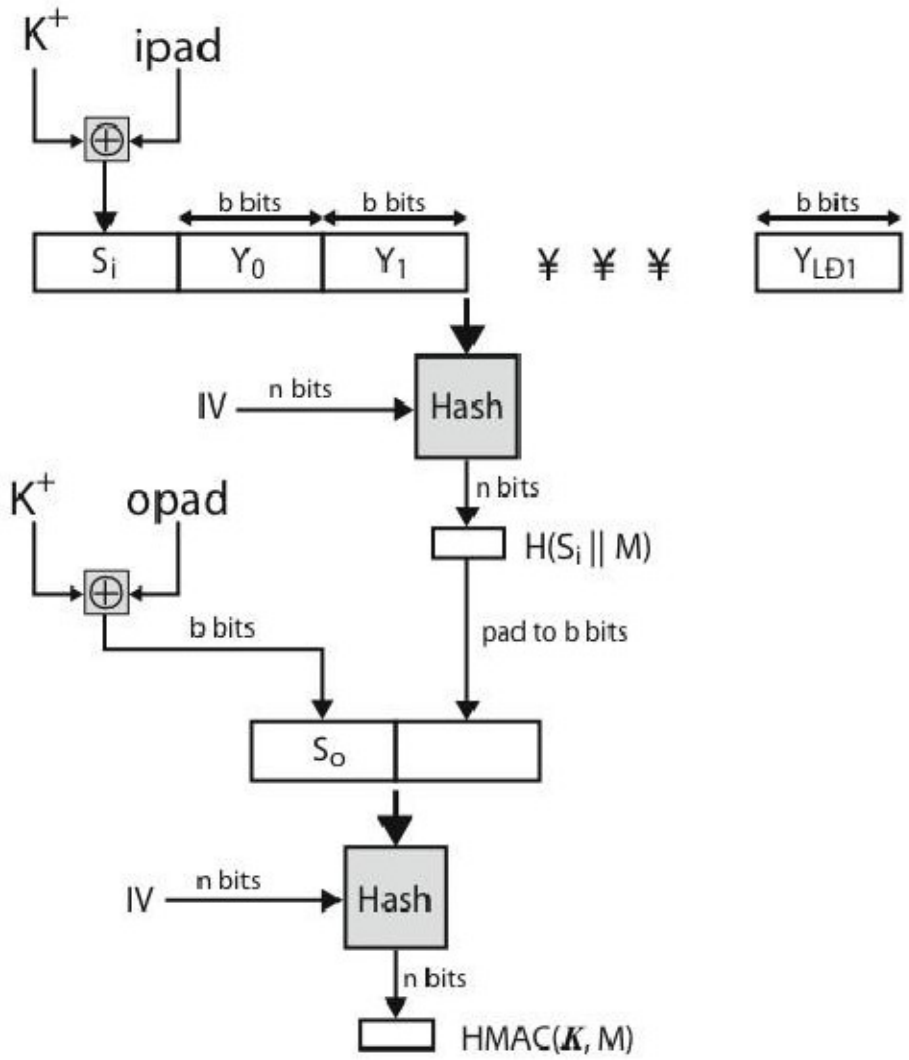
- specified as Internet standard RFC2104
- uses hash function on the message:

$HMAC_K$ = Hash[(K_+ XOR opad) ||

Hash[(K_+ XOR ipad)||M)]]

- where K_+ is the key padded out to size
- and opad, ipad are specified padding constants
- overhead is just 3 more hash calculations than the message needs alone
- any hash function can be used

  - eg. MD5, SHA-1, RIPEMD-160, Whirlpool



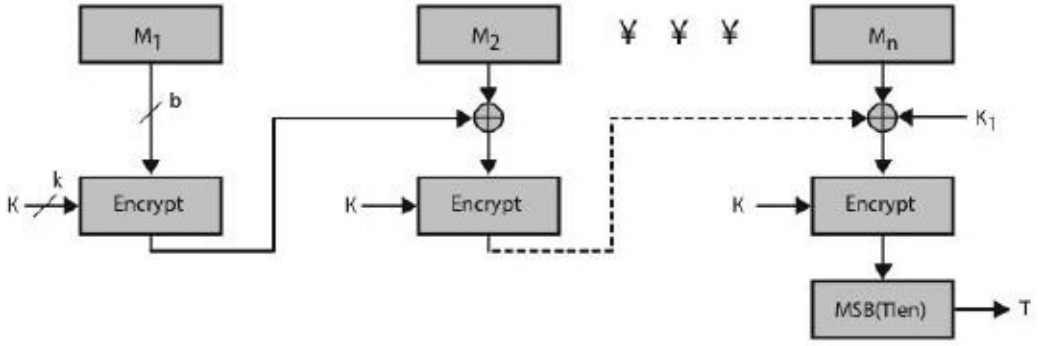
HMAC Security:

- proved security of HMAC relates to that of the underlying hash algorithm
- attacking HMAC requires either:
  - brute force attack on key used
  - birthday attack (but since keyed would need to observe a very large number of messages)
- choose hash function used based on speed verses security constraints

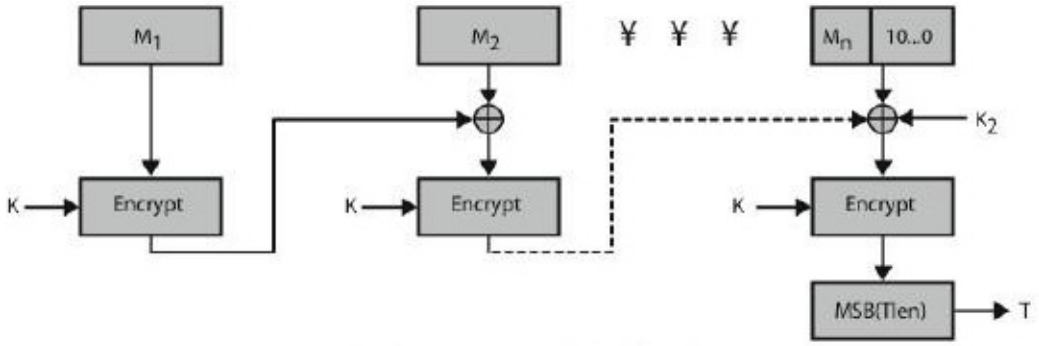
CMAC:

- **previously saw the DAA (CBC-MAC)**
- **widely used in govt& industry**
- **but has message size limitation**
- **can overcome using 2 keys & padding**

- **thus forming the Cipher-based Message Authentication Code (CMAC)**
- **adopted by NIST SP800-38B**



(a) Message length is integer multiple of block size



(b) Message length is not integer multiple of block size

Figure 12.12 Cipher-Based Message Authentication Code (CMAC)

Digital Signatures:

- 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 Properties:

- 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

Arbitrated Digital Signatures:

- involves use of arbiter A
    - validates any signed message
    - then dated and sent to recipient
- requires suitable level of trust in arbiter
- can be implemented with either private or public-key algorithms
- arbiter may or may not see message

Authentication Protocols:

- used to convince parties of each others identity and to exchange session keys
- may be one-way or mutual
- key issues are
    - confidentiality – to protect session keys

- timeliness – to prevent replay attacks

- published protocols are often found to have flaws and need to be modified
- **where a valid signed message is copied and later resent**
  - **simple replay**
  - **repetition that can be logged**
  - **repetition that cannot be detected**
  - **backward replay without modification**
- **countermeasures include**
  - **use of sequence numbers (generally impractical)**
  - **timestamps (needs synchronized clocks)**
  - **challenge/response (using unique nonce)**

Replay Attacks:

- where a valid signed message is copied and later resent
  - simple replay
  - repetition that can be logged
  - repetition that cannot be detected
  - backward replay without modification
- countermeasures include
  - use of sequence numbers (generally impractical)
  - timestamps (needs synchronized clocks)
  - challenge/response (using unique nonce)

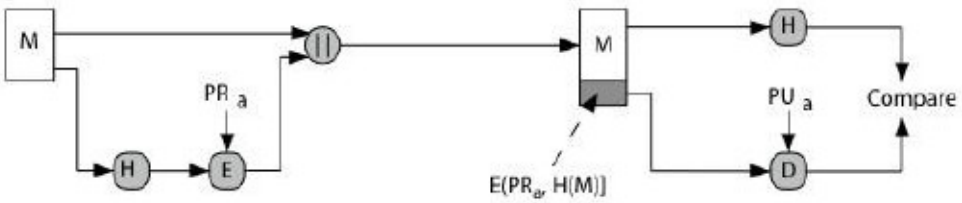
Digital Signature Standard (DSS):

- 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

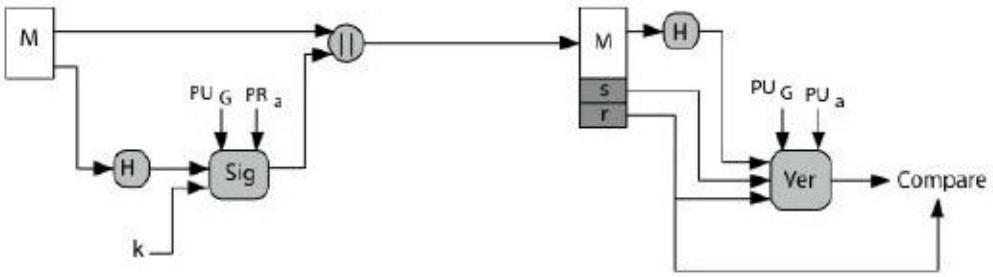
- DSS is the standard, DSA is the algorithm
- FIPS 186-2 (2000) includes alternative RSA & elliptic curve signature variants

Digital Signature Algorithm (DSA):

- creates a 320 bit signature
- with 512-1024 bit security
- smaller and faster than RSA
- a digital signature scheme only
- security depends on difficulty of computing discrete logarithms
- variant of ElGamal&Schnorr schemes



(a) RSA Approach



(b) DSS Approach

DSA Key Generation:

- have shared global public key values (p,q,g):
  - choose q, a 160 bit
  - choose a large prime $p = 2^L$
    - where L= 512 to 1024 bits and is a multiple of 64
    - and q is a prime factor of (p-1)
  - choose $g = h^{(p-1)/q}$
    - where h<p-1, $h^{(p-1)/q} \pmod p > 1$

- users choose private & compute public key:
  - choose x<q
  - compute y = g^x(mod p)

DSA Signature Creation:

- to **sign** a message M the sender:
  - generates a random signature key k, k<q
  - nb.k must be random, be destroyed after use, and never be reused
- then computes signature pair:

r = (g^k(mod p))(mod q)

s = (k^{-1}.H(M)+ x.r)(mod q)

- sends signature (r,s) with message M

DSA Signature Verification:

- having received M &signature (r,s)
- to **verify** a signature, recipient computes:

w = s^{-1}(mod q)

u1= (H(M).w)(mod q)

u2= (r.w)(mod q)

v = ($g^{u1}.y^{u2}$(mod p)) (mod q)

- if v=r then signature is verified
- see book web site for details of proof why

QUESTION BANK

PART-A(2 MARKS)

1. What is message authentication?
2. Define the classes of message authentication function.
3. What are the requirements for message authentication?
4. What you meant by hash function?
5. Differentiate MAC and Hash function?
6. Any three hash algorithm.
7. What are the requirements of the hash function?
8. What you meant by MAC?

9. Differentiate internal and external error control.

10. What is the meet in the middle attack?

11. What is the role of compression function in hash function?
12. What is the difference between weak and strong collision resistance?
13. Compare MD5, SHA1 and RIPEMD-160 algorithm.
14. Distinguish between direct and arbitrated digital signature?
15. What are the properties a digital signature should have?
16. What requirements should a digital signature scheme should satisfy?
17. Define Kerberos.
18. What 4 requirements were defined by Kerberos?
19. In the content of Kerberos, what is realm?
20. Assume the client C wants to communicate server S using Kerberos procedure. How can it be achieved?
21. What is the purpose of X.509 standard?

PART –B (16 MARKS)

1. Explain the classification of authentication function in detail (16)
2. Describe MD5 algorithm in detail. Compare its performance with SHA-1. (16)
3. Describe SHA-1 algorithm in detail. Compare its performance with MD5 and RIPEMD-160 and discuss its advantages. (16)
4. Describe RIPEMD-160 algorithm in detail. Compare its performance with MD5 and SHA-1. (16)
5. Describe HMAC algorithm in detail. (16)
6. Write and explain the Digital Signature Algorithm. (16)
7. Assume a client C wants to communicate with a server S using Kerberos protocol. How can it be achieved? (16)

UNIT IV NETWORK SECURITY

Authentication applications – Kerberos – X.509 authentication service – Electronic mail security – PGP – S/MIME – IP security – Web security.

Authentication Applications:

- will consider authentication functions
- developed to support application-level authentication & digital signatures
- will consider Kerberos – a private-key authentication service
- then X.509 - a public-key directory authentication service

Kerberos:

- trusted key server system from MIT
- provides centralised private-key third-party authentication in a distributed network
    - allows users access to services distributed through network
    - without needing to trust all workstations
    - rather all trust a central authentication server
- two versions in use: 4 & 5

Kerberos Requirements:

- its first report identified requirements as:

- secure
- reliable
- transparent
- scalable

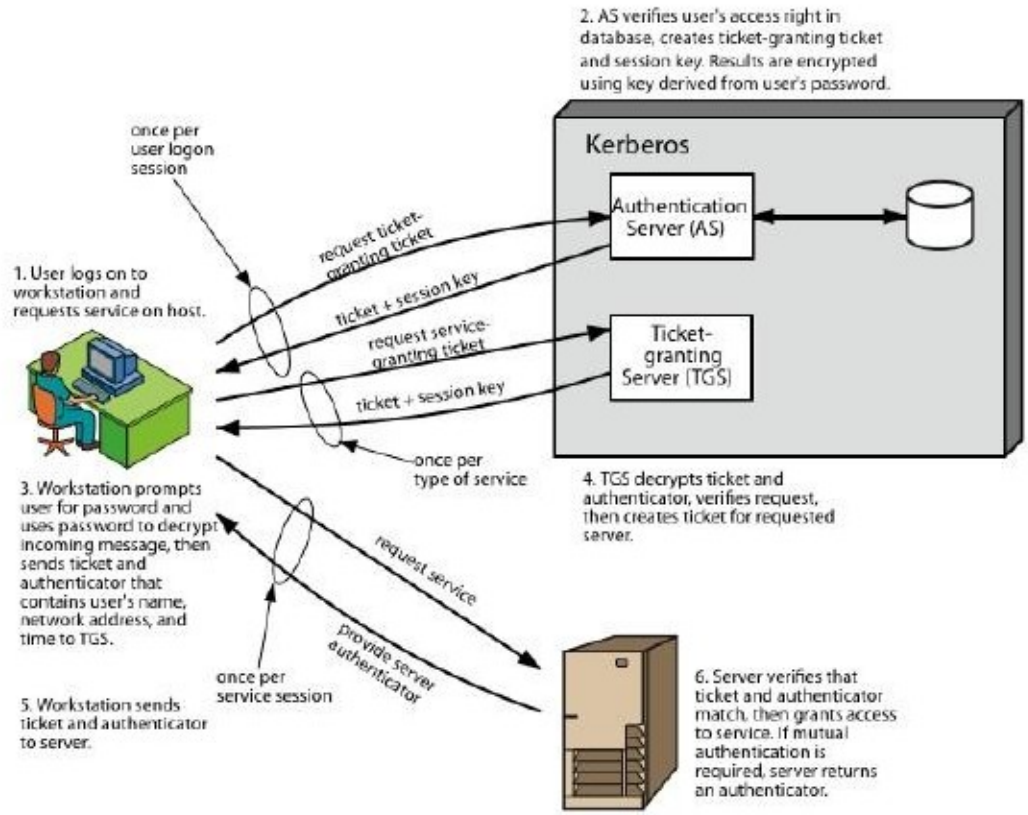
- implemented using an authentication protocol based on Needham-Schroeder

Kerberos v4 Overview:

- 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

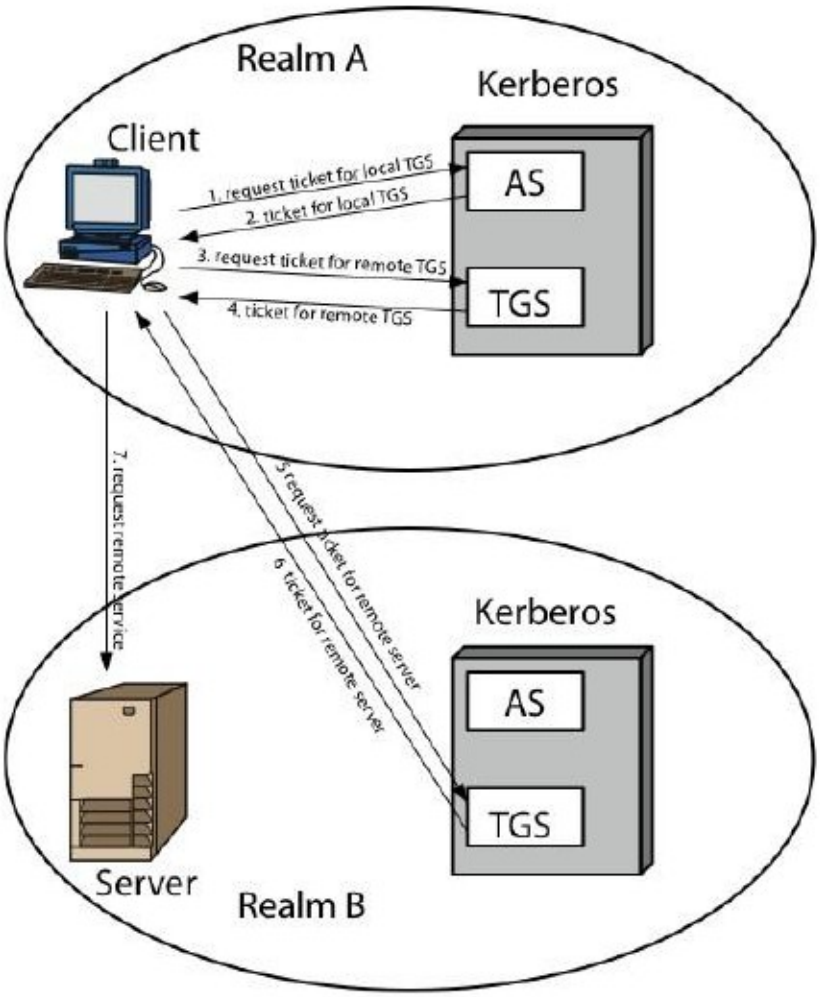
Kerberos v4 Dialogue:

1. obtain ticket granting ticket from AS
   - once per session
2. obtain service granting ticket from TGT
   - for each distinct service required
3. client/server exchange to obtain service
   - on every service request



Kerberos Realms:

- a Kerberos environment consists of:
  - a Kerberos server
  - a number of clients, all registered with server
  - application servers, sharing keys with server
- this is termed a realm
  - typically a single administrative domain
- if have multiple realms, their Kerberos servers must share keys and trust



Kerberos Version 5:

- developed in mid 1990's
- specified as Internet standard RFC 1510
- provides improvements over v4
    - addresses environmental shortcomings
        - encryption alg, network protocol, byte order, ticket lifetime, authentication forwarding, interrealmauth
    - and technical deficiencies
        - double encryption, non-std mode of use, session keys, password attacks

X.509 Authentication Service:

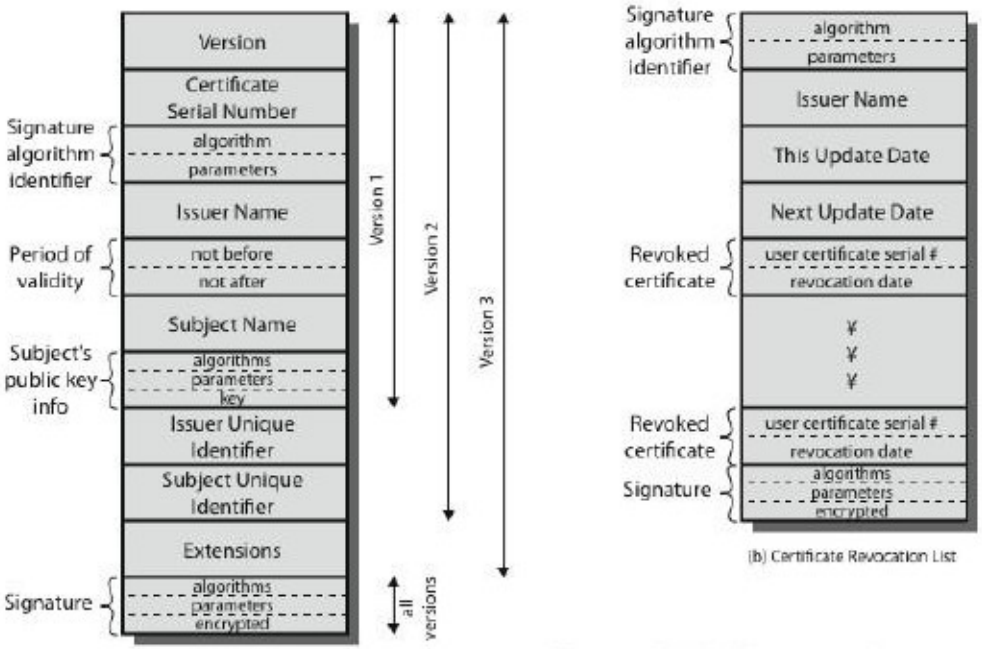
- part of CCITT X.500 directory service standards

- distributed servers maintaining user info database

- defines framework for authentication services
  - directory may store public-key certificates
  - with public key of user signed by certification authority
- also defines authentication protocols
- uses public-key crypto & digital signatures
  - algorithms not standardised, but RSA recommended
- X.509 certificates are widely used

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



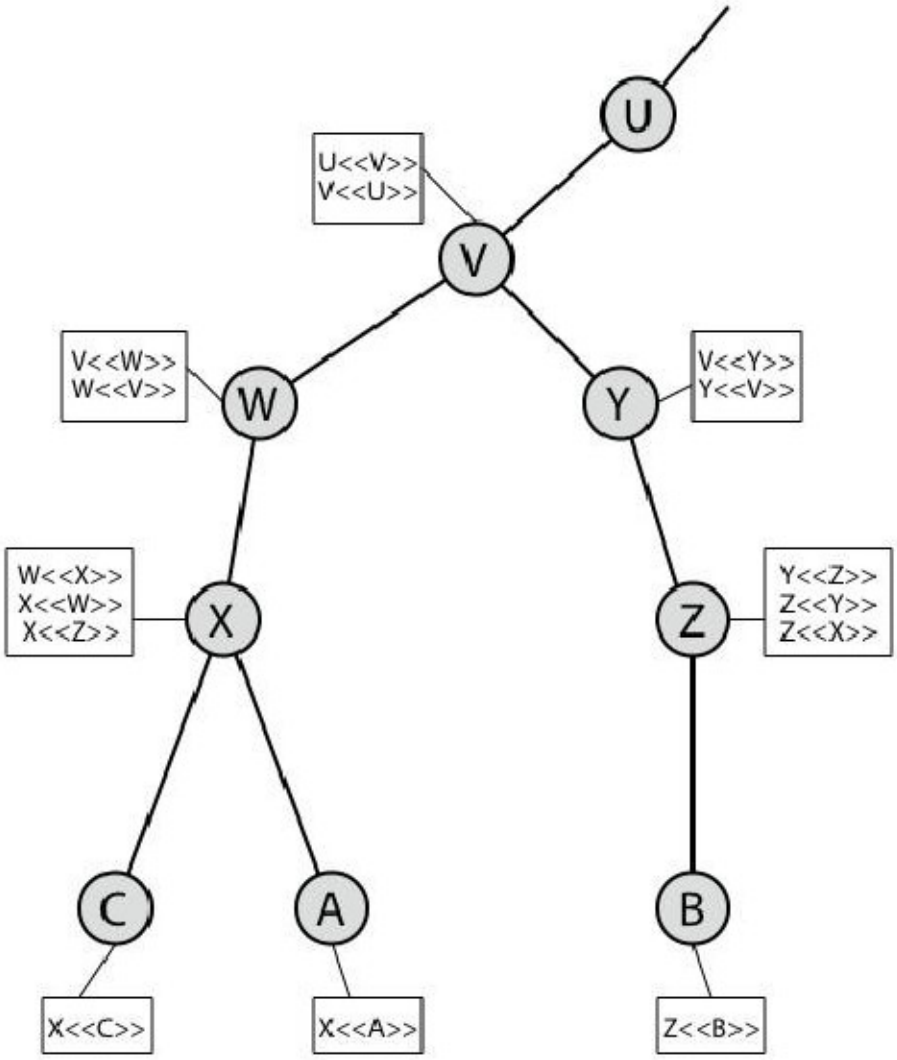
(b) Certificate Revocation List

Obtaining a Certificate:

- any user with access to CA can get any certificate from it
- only the CA can modify a certificate
- because cannot be forged, certificates can be placed in a public directory

CA Hierarchy:

- if both users share a common CA then they are assumed to know its public key
- otherwise CA's must form a hierarchy
- use certificates linking members of hierarchy to validate other CA's
    - each CA has certificates for clients (forward) and parent (backward)
- each client trusts parents certificates
- enable verification of any certificate from one CA by users of all other CAs in hierarchy



Certificate Revocation:

- certificates have a period of validity
- may need to revoke before expiry, 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 certificates 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

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
- may include additional info for B
  - eg session key

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
- may include additional info for A

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:

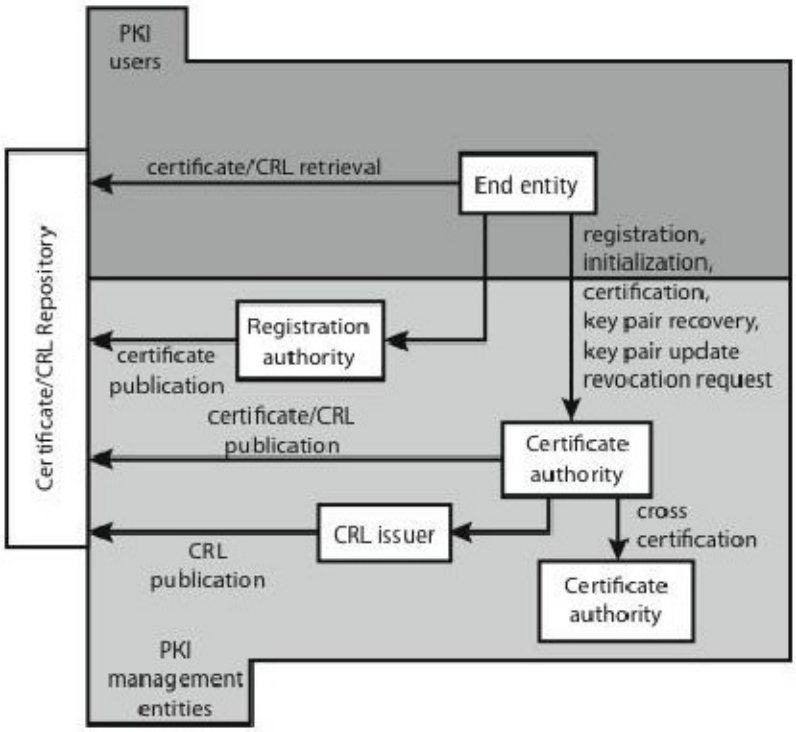
- has been recognised that additional information is needed in a certificate
  - email/URL, policy details, usage constraints

- rather than explicitly naming new fields defined a general extension method
- extensions consist of:
    - extension identifier
    - criticality indicator
    - extension value

Certificate Extensions:

- key and policy information
    - convey info about subject & issuer keys, plus indicators of certificate policy
- certificate subject and issuer attributes
    - support alternative names, in alternative formats for certificate subject and/or issuer
- certificate path constraints
    - allow constraints on use of certificates by other CA's

Public Key Infrastructure:



Email Security:

- email is one of the most widely used and regarded network services
- currently message contents are not secure
  - may be inspected either in transit
  - or by suitably privileged users on destination system

Email Security Enhancements:

- confidentiality
  - protection from disclosure
- authentication
  - of sender of message
- message integrity
  - protection from modification
- non-repudiation of origin
  - protection from denial by sender

Pretty Good Privacy (PGP):

- widely used de facto secure email
- developed by Phil Zimmermann
- selected best available crypto algs to use
- integrated into a single program
- on Unix, PC, Macintosh and other systems
- originally free, now also have commercial versions available

PGP Operation – Authentication:

1. sender creates message
2. use SHA-1 to generate 160-bit hash of message
3. signed hash with RSA using sender's private key, and is attached to message
4. receiver uses RSA with sender's public key to decrypt and recover hash code

5. receiver verifies received message using hash of it and compares with decrypted hash code

PGP Operation – Confidentiality:

1. sender generates message and 128-bit random number as session key for it
2. encrypt message using CAST-128 / IDEA / 3DES in CBC mode with session key
3. session key encrypted using RSA with recipient's public key, & attached to msg
4. receiver uses RSA with private key to decrypt and recover session key
5. session key is used to decrypt message

PGP Operation – Confidentiality & Authentication:

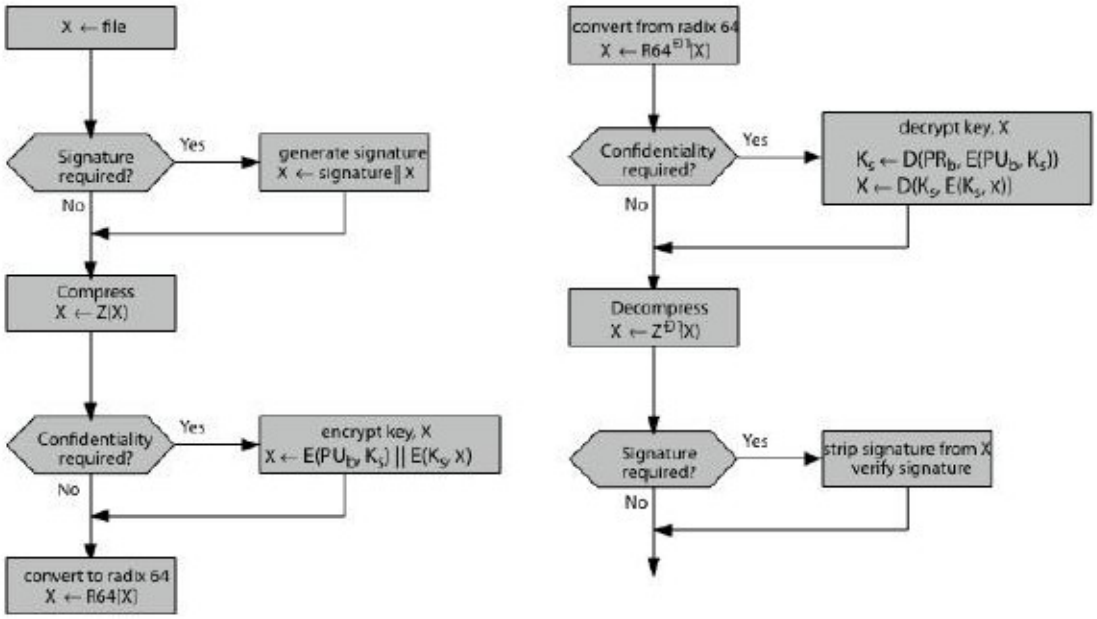
- can use both services on same message
  - create signature & attach to message
  - encrypt both message & signature
  - attach RSA/ElGamal encrypted session key

PGP Operation – Compression:

- by default PGP compresses message after signing but before encrypting
  - so can store uncompressed message & signature for later verification
  - & because compression is non deterministic
- uses ZIP compression algorithm

PGP Operation – Email Compatibility:

- when using PGP will have binary data to send (encrypted message etc)
- however email was designed only for text
- hence PGP must encode raw binary data into printable ASCII characters
- uses radix-64 algorithm
  - maps 3 bytes to 4 printable chars
  - also appends a CRC
- PGP also segments messages if too big



(a) Generic Transmission Diagram (from A)

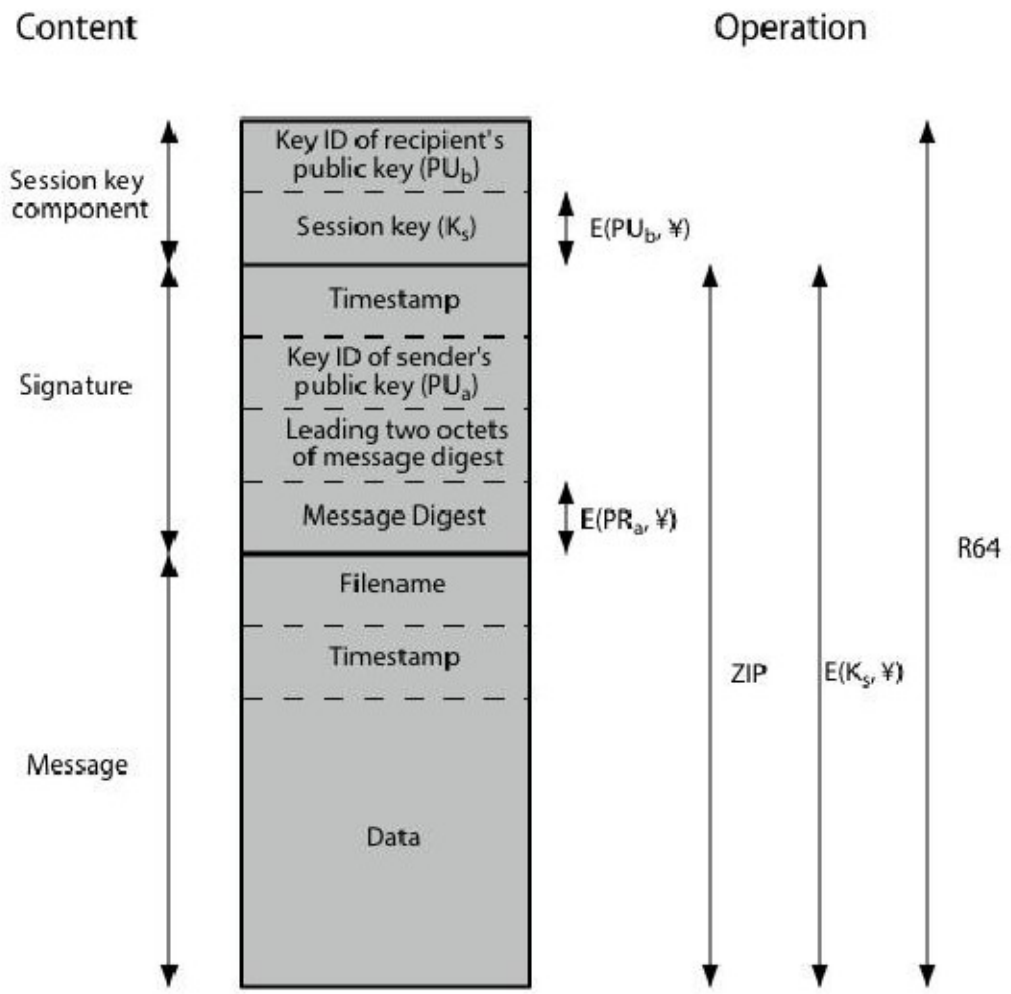
(b) Generic Reception Diagram (to B)

PGP Session Keys:

- need a session key for each message
  - of varying sizes: 56-bit DES, 128-bit CAST or IDEA, 168-bit Triple-DES
- generated using ANSI X12.17 mode
- uses random inputs taken from previous uses and from keystroke timing of user

PGP Public & Private Keys:

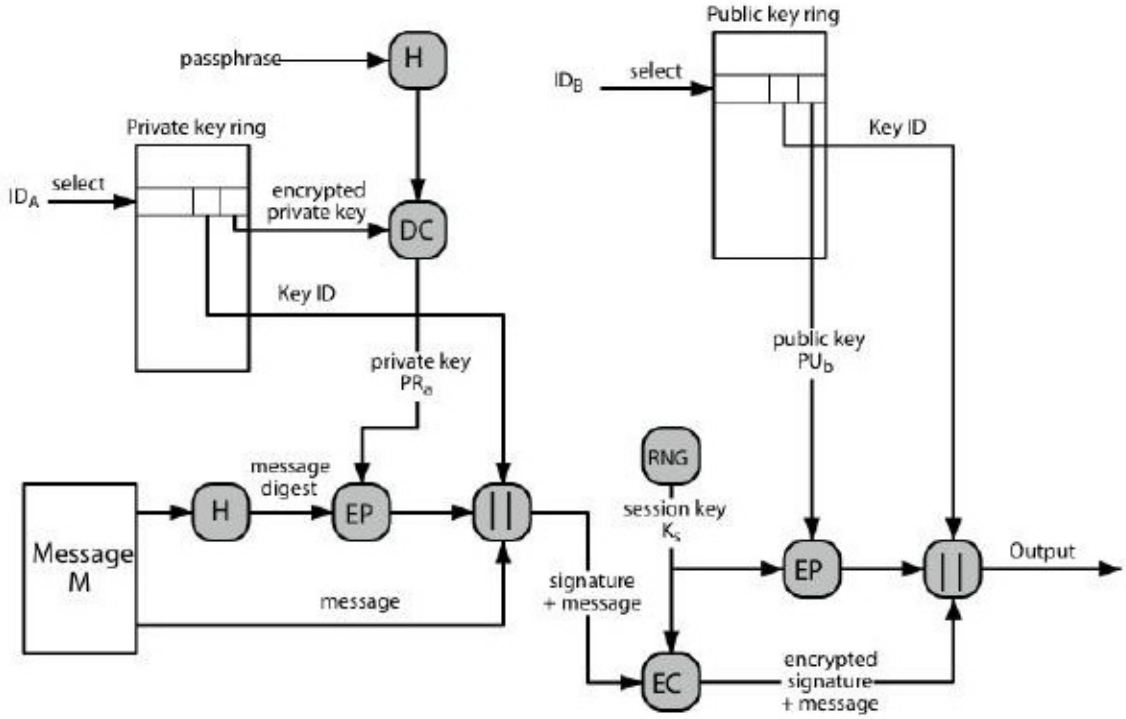
- since many public/private keys may be in use, need to identify which is actually used to encrypt session key in a message
  - could send full public-key with every message
  - but this is inefficient
- rather use a key identifier based on key
  - is least significant 64-bits of the key
  - will very likely be unique
- also use key ID in signatures



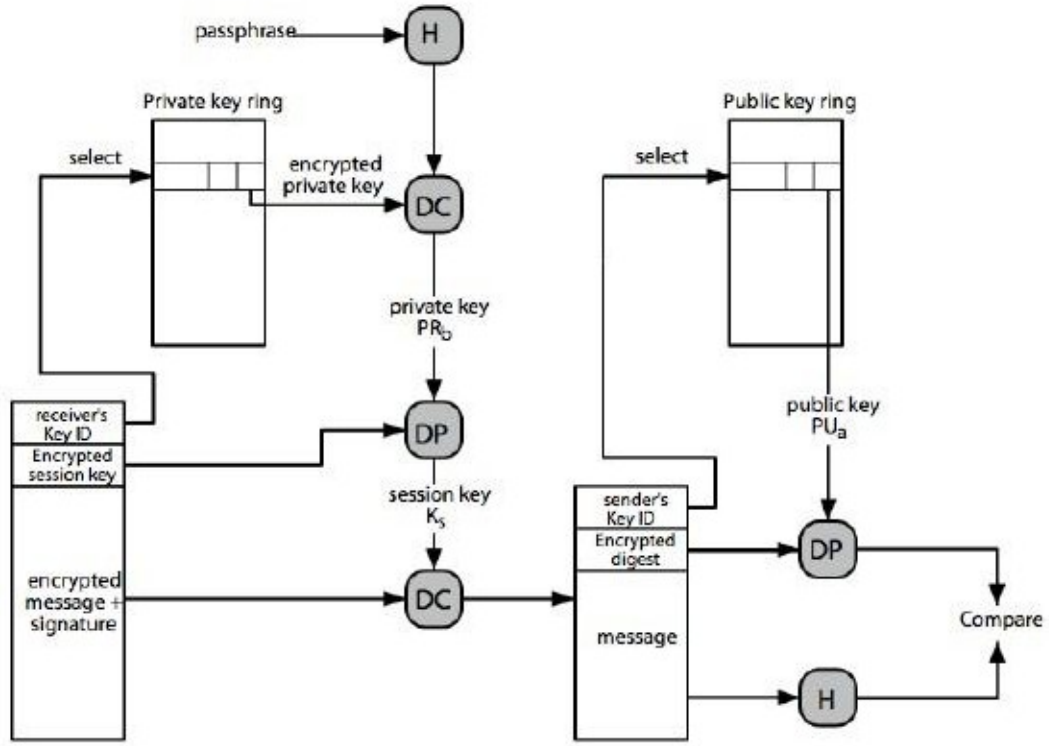
PGP Key Rings:

- each PGP user has a pair of keyrings:
  - public-key ring contains all the public-keys of other PGP users known to this user, indexed by key ID
  - private-key ring contains the public/private key pair(s) for this user, indexed by key ID & encrypted keyed from a hashed passphrase
- security of private keys thus depends on the pass-phrase security

PGP Message Generation:



PGP Message Reception:



S/MIME (Secure/Multipurpose Internet Mail Extensions):

- security enhancement to MIME email
  - original Internet RFC822 email was text only

- MIME provided support for varying content types and multi-part messages
- with encoding of binary data to textual form
- S/MIME added security enhancements

- have S/MIME support in many mail agents
  - eg MS Outlook, Mozilla, Mac Mail etc

S/MIME Functions:

- enveloped data
  - encrypted content and associated keys
- signed data
  - encoded message + signed digest
- clear-signed data
  - cleartext message + encoded signed digest
- signed & enveloped data
  - nesting of signed & encrypted entities

S/MIME Cryptographic Algorithms:

- digital signatures: DSS & RSA
- hash functions: SHA-1 & MD5
- session key encryption: ElGamal& RSA
- message encryption: AES, Triple-DES, RC2/40 and others
- MAC: HMAC with SHA-1
- have process to decide which algs to use

S/MIME Messages:

- S/MIME secures a MIME entity with a signature, encryption, or both
- forming a MIME wrapped PKCS object
- have a range of content-types:
  - enveloped data
  - signed data

- clear-signed data
- registration request
- certificate only message

S/MIME Certificate Processing:

- S/MIME uses X.509 v3 certificates
- managed using a hybrid of a strict X.509 CA hierarchy & PGP's web of trust
- each client has a list of trusted CA's certs
- and own public/private key pairs & certs
- certificates must be signed by trusted CA's

Certificate Authorities:

- have several well-known CA's
- Verisign one of most widely used
- Verisign issues several types of Digital IDs
- increasing levels of checks & hence trust

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

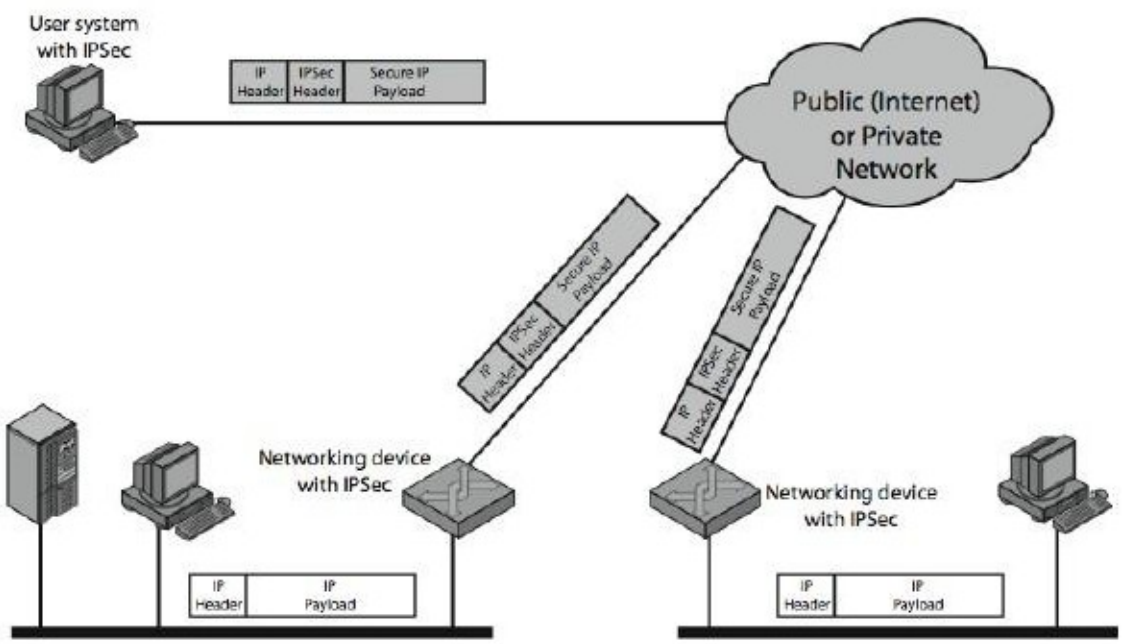
IP Security:

- 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

IPSec Uses:



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
- defined in numerous RFC's
    - incl. RFC 2401/2402/2406/2408
    - many others, grouped by category
- mandatory in IPv6, optional in IPv4

- have two security header extensions:
    - Authentication Header (AH)
    - Encapsulating Security Payload (ESP)

IPSec Services:

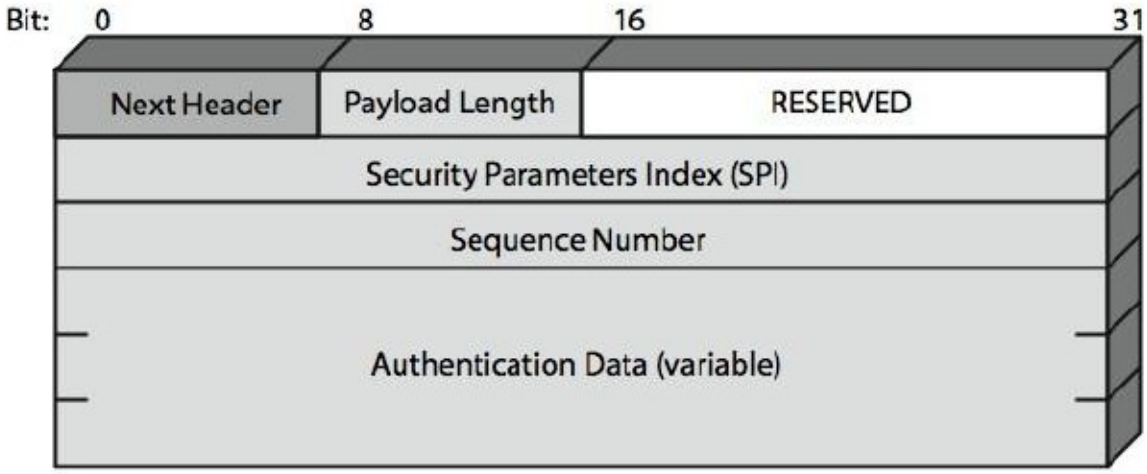
- Access control
- Connectionless integrity
- Data origin authentication
- Rejection of replayed packets
    - a form of partial sequence integrity
- Confidentiality (encryption)
- Limited traffic flow confidentiality

Security Associations :

- a one-way relationship between sender & receiver that affords security for traffic flow
- defined by 3 parameters:
    - Security Parameters Index (SPI)
    - IP Destination Address
    - Security Protocol Identifier
- has a number of other parameters
    - seq no, AH & EH info, lifetime etc
- have a database of Security Associations

Authentication Header (AH):

- provides support for data integrity & authentication of IP packets
    - end system/router can authenticate user/app
    - prevents address spoofing attacks by tracking sequence numbers
- based on use of a MAC
    - HMAC-MD5-96 or HMAC-SHA-1-96
- parties must share a secret key

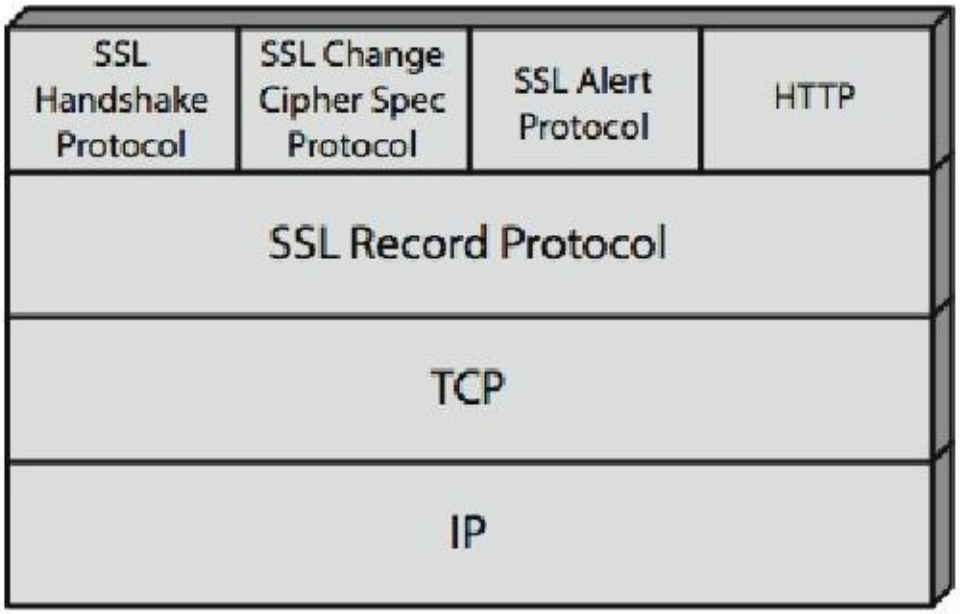


Web Security:

- Web now widely used by business, government, individuals
- but Internet & Web are vulnerable
- have a variety of threats
    - integrity
    - confidentiality
    - denial of service
    - authentication
- need added security mechanisms

SSL (Secure Socket Layer):

- transport layer security service
- originally developed by Netscape
- version 3 designed with public input
- subsequently became Internet standard known as TLS (Transport Layer Security)
- uses TCP to provide a reliable end-to-end service
- SSL has two layers of protocols

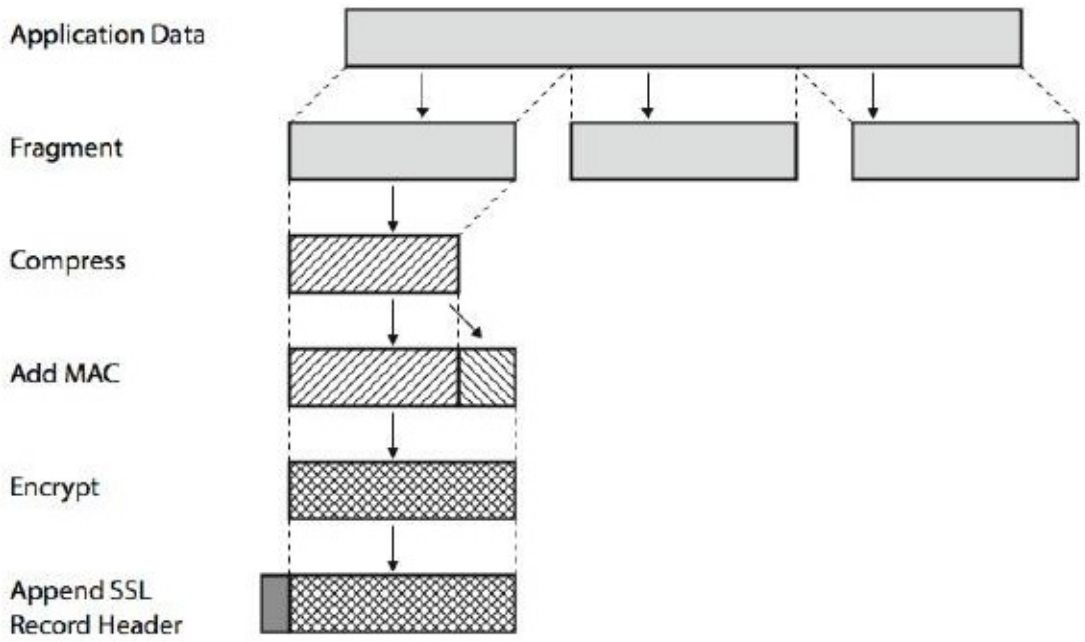


- **SSL connection**
  - a transient, peer-to-peer, communications link
  - associated with 1 SSL session
- **SSL session**
  - an association between client & server
  - created by the Handshake Protocol
  - define a set of cryptographic parameters
  - may be shared by multiple SSL connections

SSL Record Protocol Services:

- **message integrity**
  - using a MAC with shared secret key
  - similar to HMAC but with different padding
- **confidentiality**
  - using symmetric encryption with a shared secret key defined by Handshake Protocol
  - AES, IDEA, RC2-40, DES-40, DES, 3DES, Fortezza, RC4-40, RC4-128
  - message is compressed before encryption

SSL Record Protocol Operation:



SSL Change Cipher Spec Protocol:

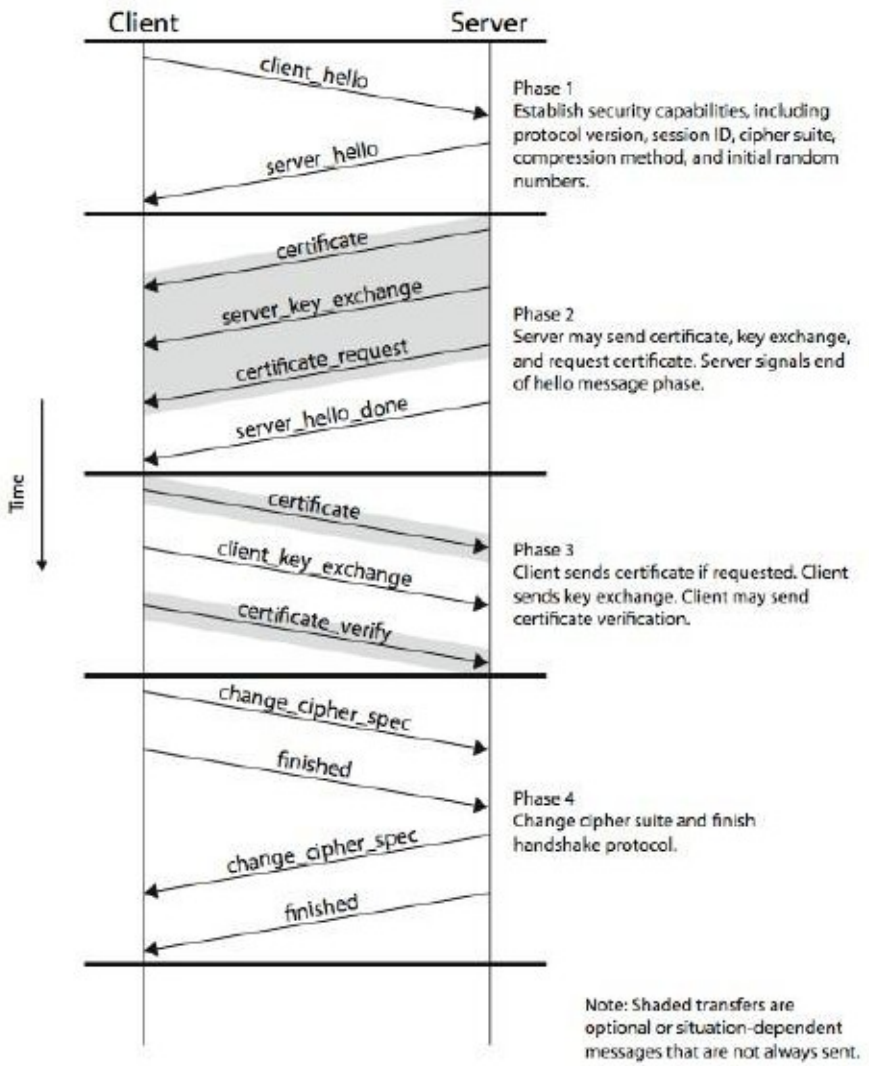
- one of 3 SSL specific protocols which use the SSL Record protocol
- a single message
- causes pending state to become current
- hence updating the cipher suite in use

SSL Alert Protocol:

- conveys SSL-related alerts to peer entity
- severity
    - warning or fatal
- specific alert
    - fatal: unexpected message, bad record mac, decompression failure, handshake failure, illegal parameter
    - warning: close notify, no certificate, bad certificate, unsupported certificate, certificate revoked, certificate expired, certificate unknown
- compressed & encrypted like all SSL data

SSL Handshake Protocol:

- allows server & client to:
  - authenticate each other
  - to negotiate encryption & MAC algorithms
  - to negotiate cryptographic keys to be used
- comprises a series of messages in phases
  - Establish Security Capabilities
  - Server Authentication and Key Exchange
  - Client Authentication and Key Exchange
  - Finish



QUESTION BANK

PART-A(2 MARKS)

1. What are the services provided by PGP services
2. Explain the reasons for using PGP?
3. Why E-mail compatibility function in PGP needed?
4. Name any cryptographic keys used in PGP?
5. Define key Identifier?
6. List the limitations of SMTP/RFC 822?
7. Draw the diagram for PGP message transmission reception?
8. What is the general format for PGP message?
9. Define S/MIME?
10. What are the elements of MIME?
11. What are the headers fields define in MIME?
12. What is MIME content type and explain?
13. What are the key algorithms used in S/MIME?
14. Give the steps for preparing envelope data MIME?
15. What you mean by Verisign certificate?
16. What are the function areas of IP security?
17. Give the application of IP security?
18. Give the benefits of IP security?
19. What are the protocols used to provide IP security?
20. Specify the IP security services?

PART –B (16 MARKS)

1. Explain the operational description of PGP. (16)
2. Write Short notes on S/MIME. (16)
3. Explain the architecture of IP Security. (16)
4. Write short notes on authentication header and ESP. (16)
5. Explain in detail the operation of Secure Socket Layer in detail. (16)
6. Explain Secure Electronic transaction with neat diagram. (16)

UNIT V SYSTEM LEVEL SECURITY

Intrusion detection – Password management – Viruses and related threats – Virus counter measures – Firewall design principles – Trusted systems.

Intruders:

- significant issue for networked systems is hostile or unwanted access
- either via network or local
- can identify classes of intruders:
  - masquerader
  - misfeasor
  - clandestine user
- varying levels of competence
- 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
- awareness of intruders has led to the development of CERTs

Intrusion Techniques:

- aim to gain access and/or increase privileges on a 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
    - defaults, short passwords, common word searches
    - user info (variations on names, birthday, phone, common words/interests)
    - exhaustively searching all possible passwords
- check by login 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 dialedetc)
- 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

- assume intruder will behave differently to a legitimate user
  - but will have imperfect distinction between

Approaches to Intrusion Detection:

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

Audit Records:

- fundamental tool for intrusion detection
- native audit records
  - part of all common multi-user O/S
  - already present for use
  - may not have info wanted in desired form
- detection-specific audit records
  - created specifically to collect wanted info
  - at cost of additional overhead on system

Statistical Anomaly Detection:

- threshold detection
  - count occurrences of specific event over time
  - if exceed reasonable value assume intrusion
  - alone is a crude & ineffective detector
- profile based
  - characterize past behavior of users
  - detect significant deviations from this

- profile usually multi-parameter

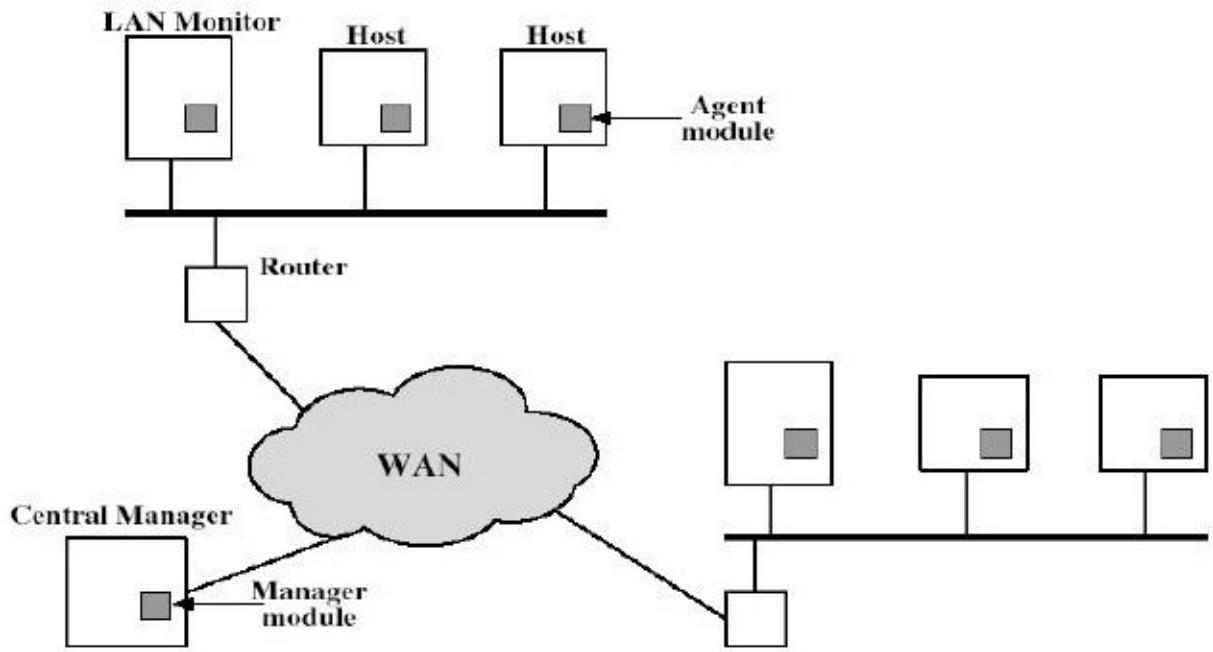
Audit Record Analysis:

- foundation of statistical approaches
- analyze records to get metrics over time
  - counter, gauge, interval timer, resource use
- use various tests on these to determine if current behavior is acceptable
  - mean & standard deviation, multivariate, markov process, time series, operational
- key advantage is no prior knowledge used

Rule-Based Intrusion Detection:

- observe events on system & apply rules to decide if activity is suspicious or not
- rule-based anomaly detection
  - analyze historical audit records to identify usage patterns & auto-generate rules for them
  - then observe current behavior & match against rules to see if conforms
  - like statistical anomaly detection does not require prior knowledge of security flaws
- rule-based penetration identification
  - uses expert systems technology
  - with rules identifying known penetration, weakness patterns, or suspicious behavior
  - compare audit records or states against rules
  - rules usually machine & O/S specific
  - rules are generated by experts who interview & codify knowledge of security admins
  - quality depends on how well this is done

Distributed Intrusion Detection – Architecture:



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
- single or multiple networked systems
- cf IETF Intrusion Detection WG standards

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

- should protect password file on system

Password Studies:

- Purdue 1992 - many short passwords
- Klein 1990 - many guessable passwords
- conclusion is that users choose poor passwords too often
- need some approach to counter this

Managing Passwords – Education:

- can use policies and good user education
- educate on importance of good passwords
- give guidelines for good passwords
    - minimum length (>6)
    - require a mix of upper & lower case letters, numbers, punctuation
    - not dictionary words
- but likely to be ignored by many users
- let computer create passwords
- if random likely not memorisable, so will be written down (sticky label syndrome)
- even pronounceable not remembered
- have history of poor user acceptance
- FIPS PUB 181 one of best generators
    - has both description & sample code
    - generates words from concatenating random pronounceable syllables

Managing Passwords - Reactive Checking:

- reactively run password guessing tools
    - note that good dictionaries exist for almost any language/interest group
- cracked passwords are disabled
- but is resource intensive
- bad passwords are vulnerable till found

Viruses:

- a piece of self-replicating code attached to some other code
  - cf biological virus
- both propagates itself & carries a payload
  - carries code to make copies of itself
  - as well as code to perform some covert task

Virus Operation:

- virus phases:
  - dormant – waiting on trigger event
  - propagation – replicating to programs/disks
  - triggering – by event to execute payload
  - execution – of payload
- details usually machine/OS specific
  - exploiting features/weaknesses

Virus Structure:

```
program V :=
    {goto main;
    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 condition holds}
    main: main-program := {infect-executable;
                    if trigger-pulled then do-damage;
                    goto next;}
```

next:

}

Types of Viruses:

- can classify on basis of how they attack
- parasitic virus
- memory-resident virus
- boot sector virus
- stealth
- polymorphic virus
- metamorphic virus

Macro Virus:

- **macro code** attached to some **data file**
- interpreted by program using file
    - eg Word/Excel macros
    - esp. using auto command & command macros
- code is now platform independent
- is a major source of new viral infections
- blur distinction between data and program files
- classic trade-off: "ease of use" vs "security"
- have improving security in Word etc
- are no longer dominant virus threat

Email Virus:

- spread using email with attachment containing a macro virus
    - cf Melissa
- triggered when user opens attachment
- or worse even when mail viewed by using scripting features in mail agent
- hence propagate very quickly

- usually targeted at Microsoft Outlook mail agent & Word/Excel documents
- need better O/S & application security

Worms:

- replicating but not infecting program
- typically spreads over a network
    - cf Morris Internet Worm in 1988
    - led to creation of CERTs
- using users distributed privileges or by exploiting system vulnerabilities
- widely used by hackers to create zombie PC's, subsequently used for further attacks, espDoS
- major issue is lack of security of permanently connected systems, esp PC's

Worm Operation:

- worm phases like those of viruses:
    - dormant
    - propagation
        - search for other systems to infect
        - establish connection to target remote system
        - replicate self onto remote system
    - triggering
    - execution

Virus Countermeasures:

- best countermeasure is prevention
- but in general not possible
- hence need to do one or more of:
    - **detection** - of viruses in infected system
    - **identification** - of specific infecting virus
    - **removeal** - restoring system to clean state

Anti-Virus Software:

- **first-generation**
  - scanner uses virus signature to identify virus
  - or change in length of programs
- **second-generation**
  - uses heuristic rules to spot viral infection
  - or uses crypto hash of program to spot changes
- **third-generation**
  - memory-resident programs identify virus by actions
- **fourth-generation**
  - packages with a variety of antivirus techniques
  - eg scanning & activity traps, access-controls
- arms race continues

Advanced Anti-Virus Techniques

- **first-generation**
  - scanner uses virus signature to identify virus
  - or change in length of programs
- **second-generation**
  - uses heuristic rules to spot viral infection
  - or uses crypto hash of program to spot changes
- **third-generation**
  - memory-resident programs identify virus by actions
- **fourth-generation**
  - packages with a variety of antivirus techniques
  - eg scanning & activity traps, access-controls
- arms race continues

What is a Firewall?

- a **choke point** of control and monitoring
- interconnects networks with differing trust
- imposes restrictions on network services
    - only authorized traffic is allowed
- auditing and controlling access
    - can implement alarms for abnormal behavior
- provide NAT & usage monitoring
- implement VPNs using IPSec
- must be immune to penetration

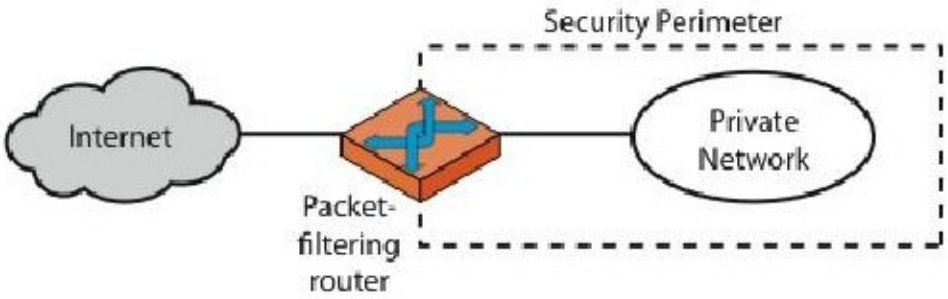
Firewall Limitations:

- cannot protect from attacks bypassing it
    - eg sneaker net, utility modems, trusted organisations, trusted services (eg SSL/SSH)
- cannot protect against internal threats
    - eg disgruntled or colluding employees
- cannot protect against transfer of all virus infected programs or files
    - because of huge range of O/S & file types

Firewalls – Packet Filters:

- simplest, fastest firewall component
- foundation of any firewall system
- examine each IP packet (no context) and permit or deny according to rules
- hence restrict access to services (ports)
- possible default policies
    - that not expressly permitted is prohibited
    - that not expressly prohibited is permitted

Firewalls – Packet Filters:



(a) Packet-filtering router

Firewalls – Packet Filters:

Pac et-Filtering Examples

tion	ourhost	port	theirhost	port	comment
ock	*	*	SPIGOT	*	we don't trust these people
low	OUR-GW	25	*	*	connection to our SMTP port

tion	ourhost	port	theirhost	port	comment
ock	*	*	*	*	default

tion	ourhost	port	theirhost	port	comment
low	*	*	*	25	connection to their SMTP port

tion	src	port	dest	port	flags	comment
low	{our hosts}	*	*	25		our packets to their SMTP port
low	*	25	*	*	ACK	their replies

tion	src	port	dest	port	flags	comment
low	{our hosts}	*	*	*		our outgoing calls
low	*	*	*	*	ACK	replies to our calls
low	*	*	*	>1024		traffic to nonservers

Attacks on Packet

Filters:

- IP address spoofing

- fake source address to be trusted
- add filters on router to block

- source routing attacks
  - attacker sets a route other than default
  - block source routed packets
- tiny fragment attacks
  - split header info over several tiny packets
  - either discard or reassemble before check

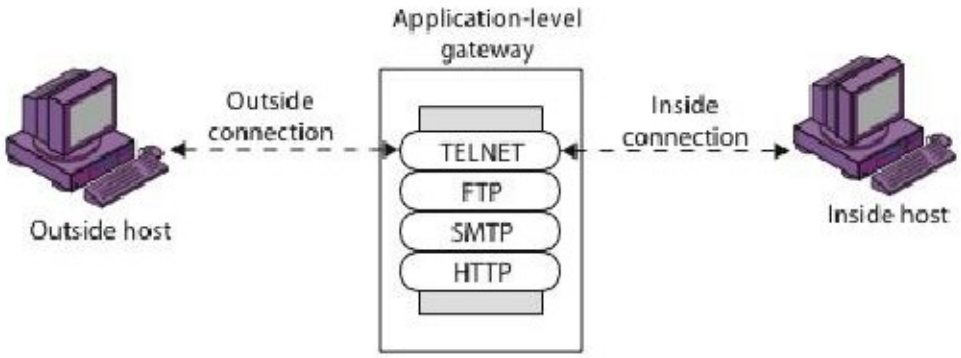
Firewalls – Stateful Packet Filters:

- traditional packet filters do not examine higher layer context
  - ie matching return packets with outgoing flow
- stateful packet filters address this need
- they examine each IP packet in context
  - keep track of client-server sessions
  - check each packet validly belongs to one
- hence are better able to detect bogus packets out of context

Firewalls - Application Level Gateway (or Proxy):

- have application specific gateway / proxy
- has full access to protocol
  - user requests service from proxy
  - proxy validates request as legal
  - then actions request and returns result to user
  - can log / audit traffic at application level
- need separate proxies for each service
  - some services naturally support proxying
  - others are more problematic

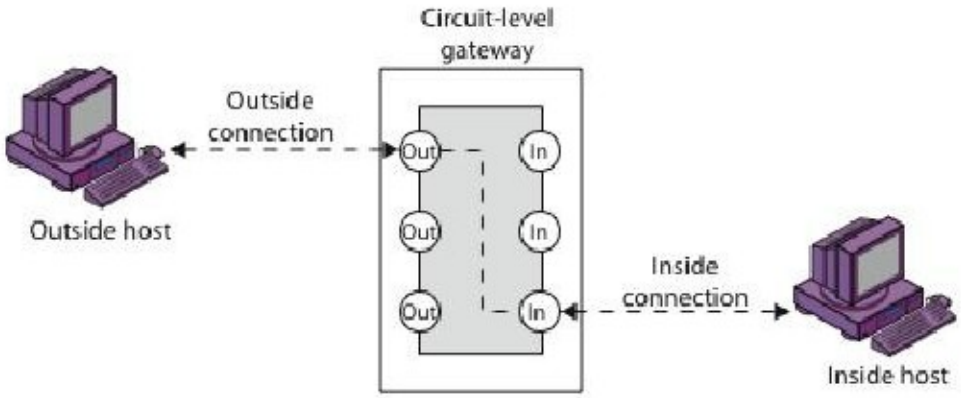
Firewalls - Application Level Gateway (or Proxy):



(b) Application-level gateway

Firewalls - Circuit Level Gateway:

- relays two TCP connections
- imposes security by limiting which such connections are allowed
- once created usually relays traffic without examining contents
- typically used when trust internal users by allowing general outbound connections
- SOCKS is commonly used



(c) Circuit-level gateway

Bastion Host:

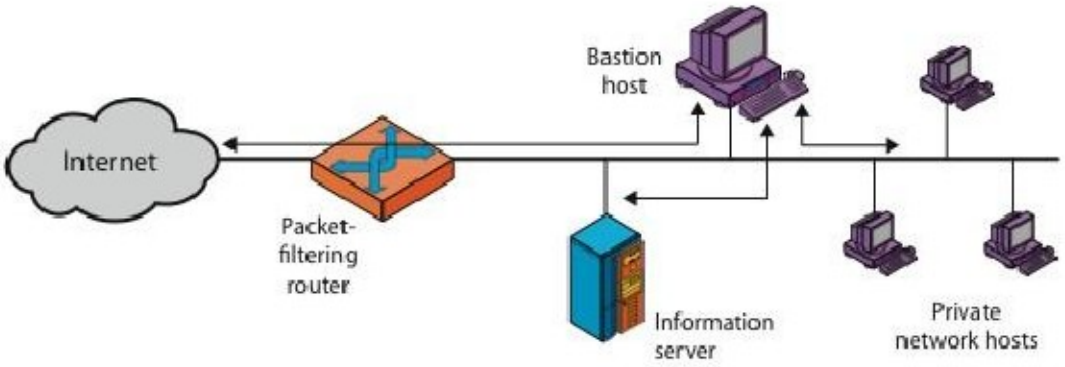
- highly secure host system
- runs circuit / application level gateways
- or provides externally accessible services
- potentially exposed to "hostile" elements
- hence is secured to withstand this

- hardened O/S, essential services, extra auth
- proxies small, secure, independent, non-privileged

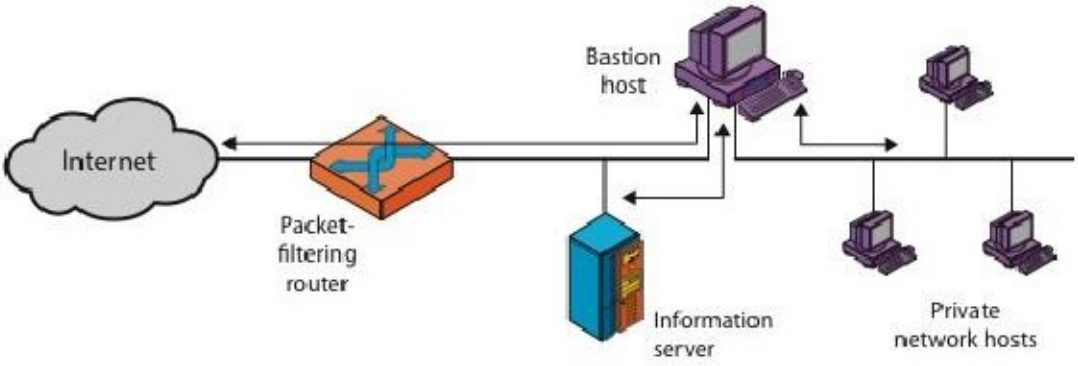
➢ may support 2 or more net connections

➢ may be trusted to enforce policy of trusted separation between these net connections

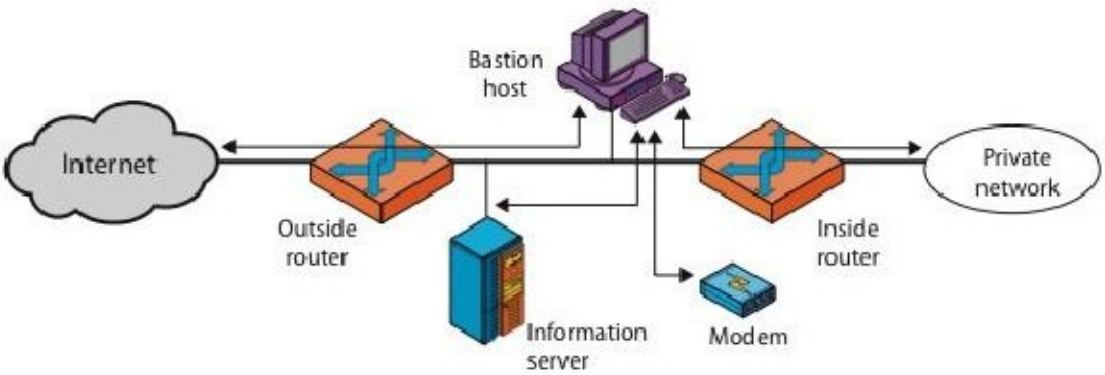
Firewall Configurations:



(a) Screened host firewall system (single-homed bastion host)



(b) Screened host firewall system (dual-homed bastion host)



(c) Screened-subnet firewall system

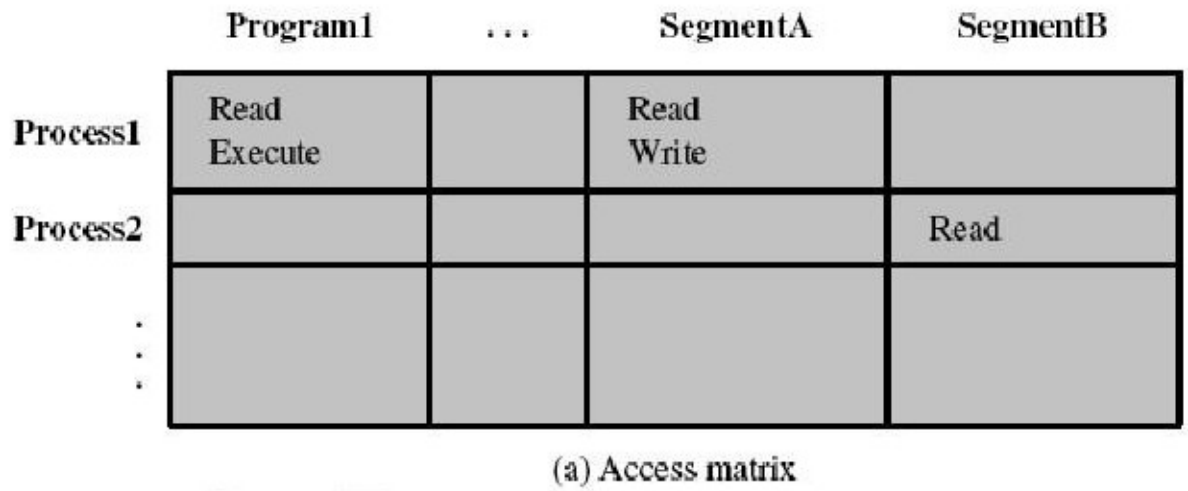
Access Control:

➢ given system has identified a user

- determine what resources they can access
- general model is that of access matrix with
  - **subject** - active entity (user, process)
  - **object** - passive entity (file or resource)
  - **access right** – way object can be accessed
- can decompose by
  - columns as access control lists
  - rows as capability tickets

:

Access Control Matrix:



(a) Access matrix

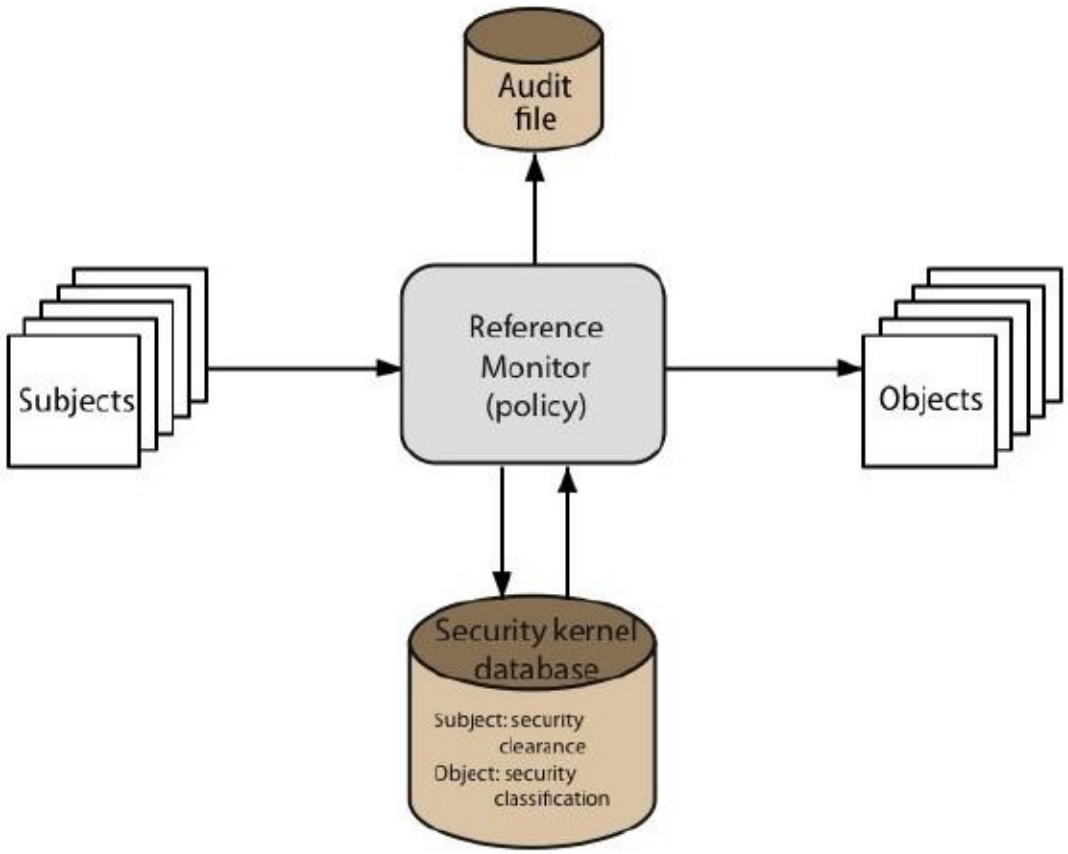
Trusted Computer Systems:

- information security is increasingly important
- have varying degrees of sensitivity of information
  - cf military info classifications: confidential, secret etc
- subjects (people or programs) have varying rights of access to objects (information)
- known as multilevel security
  - subjects have **maximum&current** security level
  - objects have a fixed security level **classification**
- want to consider ways of increasing confidence in systems to enforce these rights

Bell LaPadula (BLP) Model:

- one of the most famous security models
- implemented as mandatory policies on system
- has two key policies:
- **no read up** (simple security property)
    - a subject can only read/write an object if the current security level of the subject dominates (>=) the classification of the object
- **no write down** (*-property)
    - a subject can only append/write to an object if the current security level of the subject is dominated by (<=) the classification of the object

Reference Monitor:



Evaluated Computer Systems:

- governments can evaluate IT systems
- against a range of standards:
    - TCSEC, IPSEC and now Common Criteria
- define a number of "levels" of evaluation with increasingly stringent checking

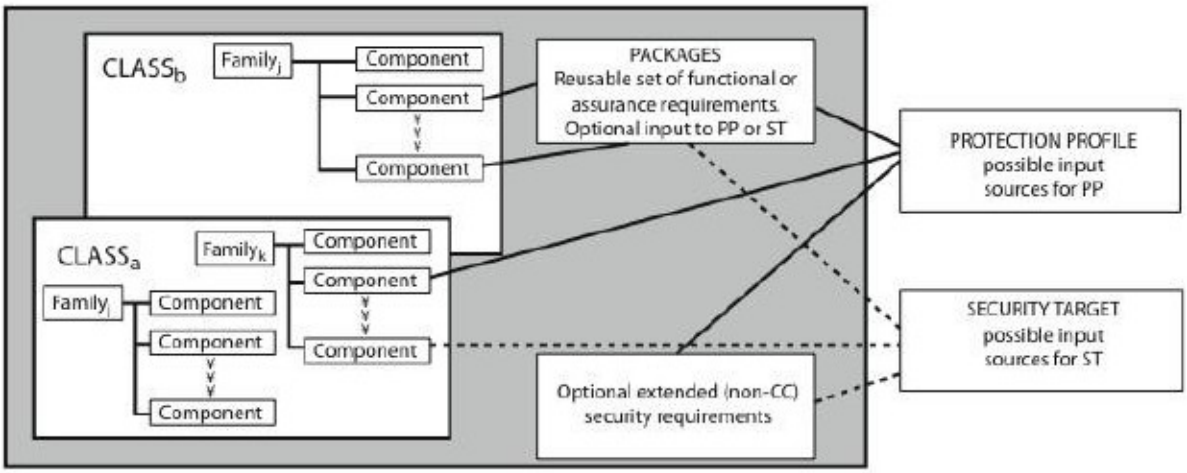
- have published lists of evaluated products
    - though aimed at government/defense use
    - can be useful in industry also

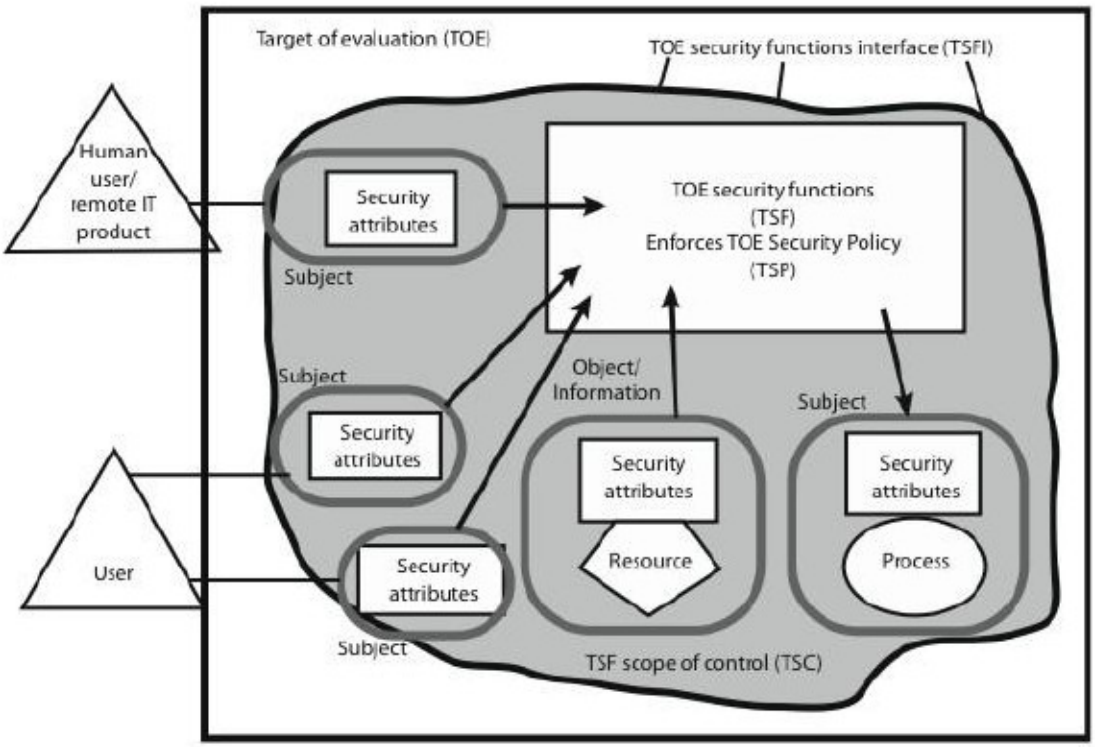
Common Criteria:

- international initiative specifying security requirements & defining evaluation criteria
- incorporates earlier standards
    - eg CSEC, ITSEC, CTCPEC (Canadian), Federal (US)
- specifies standards for
    - evaluation criteria
    - methodology for application of criteria
    - administrative procedures for evaluation, certification and accreditation schemes
- defines set of security requirements
- have a Target Of Evaluation (TOE)
- requirements fall in two categories
    - functional
    - assurance
- both organised in classes of families & components

Common Criteria Requirements:

- Functional Requirements
    - security audit, crypto support, communications, user data protection, identification & authentication, security management, privacy, protection of trusted security functions, resource utilization, TOE access, trusted path
- Assurance Requirements
    - configuration management, delivery & operation, development, guidance documents, life cycle support, tests, vulnerability assessment, assurance maintenance





QUESTION BANK

PART-A (2 MARKS)

1. General format of IPsec ESP Format?
2. What is Authentication Header? Give the format of the IPsec Authentication Header?
3. Define Transport Adjacency and Iterated Tunnel?
4. Give features and weakness of Diffie Hellman?
5. Explain man in the middle attack?

6. List the steps involved in SSL record protocol?
7. Give SSL record format?
8. What are the different between SSL version 3 and TLS?
9. What is mean by SET? What are the features of SET?
10. What are the steps involved in SET Transaction?
11. What is dual signature? What it is purpose?
12. List the 3 classes of intruder?
13. Define virus. Specify the types of viruses?
14. What is application level gateway?
15. List the design goals of firewalls?
16. Differentiate Transport and Tunnel mode in IPsec?
17. Explain the format of ESP Transport Mode?

PART –B (16 MARKS)

1. Explain the technical details of firewall and describe any three types of firewall with neat diagram. (16)
2. Write short notes on Intrusion Detection. (16)
3. Define virus. Explain in detail. (16)
4. Describe trusted system in detail. (16)
5. Explain in detail about password management. (16)

UNIVERSITY QUESTIONS

B.E/B.Tech DEGREE EXAMINATION, NOVEMBER/DECEMBER 2009

IT 1352-CRYPTOGRAPHY AND NETWORK SECURITY Question Papers

PART A

1. what is cryptanalysis and cryptography?
2. Define threat and attack
3. What is the role of session key in public key schemes?
4. what is a zero point of an elliptic curve?
5. what are the functions used to produce an authendicator?
6. list the properties a digital signature should possess?
7. mention the scenario where kerberos scheme is prefered
8. what are the technical deficiencies in the kerberos version 4 protocol?
9. list the classes of intruders
10. give the type of viruses

PART B

11. a) Explain the OSI security architecture alon with the services available(16)

OR

b)(i) given 10bit key k=1010000010. determine K1,K2 where
P10= 3 5 2 7 4 10 1 9 8 6
p8 = 6 3 7 4 8 5 10 9
by using SDES key generation method.(10)
(ii) using the positional value of alphabets, represent them in 5 bit binary. aplly the transformation Ci=Ki EX-OR Pi,Pi=Ci EX-OR Ki where Pi="scheme" Ki="cipher". find the cipher text(6)

12. a)(i)perform encryption/decryption using RSA algorithm for the following:
p=3, q=11, e=7, m=5 (12)
(ii) what attacks are possible on RSA algorithm?(4)

OR

b)(i)given the key "MONARCHY" apply play fair to pain text "FACTIONALISM" to ensure confidentially at the destination, decrypt the ciphertext and establish authenticity(8)
(ii) apply public key encryption to establish confidentiality in the message from A to B. you are given m=67. KU={7,187}, KR={23,187}.(8)

13. a)(i)Apply the MAC on the cryptographic checksum method to authendicate build

confidentiality of the message where the authentication is tied to message
M=8376, K1=4892, K2=53624071. (10)
(ii)what are the properties a hash function must satisfy? (6)

OR

b) explain MD5 message digest algorithm, with its logic and compression function.(16)

14. a)explain X.509 authentication servise and its certificates(16)

OR

b)(i)explain the services of PGP(12)
(ii)write down the functions provided by S/MIME(4)

15.a)(i)list the approaches for the intrusion detection(4)
(ii)explain firewall design principles, characteristics, and types of firewalls(12)

OR

b)(i)give the basic techniques which are in use for the password selection strategies(8)
(ii)write down the four generations of antivirus software(8)

B.E/B.Tech DEGREE EXAMINATION, NOVEMBER/DECEMBER 2008

IT 1352-CRYPTOGRAPHY AND NETWORK SECURITY Question Papers

PART-A **2x10=20**

1. What is cryptography?
2. Give any four names of substitution techniques
3. What are the services defined by x.800?
4. What is the purpose of Diffie-Hellman algorithm?
5. Define man in the middle attack
6. List design objectives for HMAC
7. What is MAC?
8. What are the requirements for digital signature?
9. Give the Kerberos simple dialogue
10. What is firewall?

PART-B **5x16=80**

11 a(i) Briefly explain about OSI security architecture (8)
(ii) Explain briefly about data encryption standard (8)

(OR)

b(i) Explain briefly about block cipher principles and modes of operation (12)
(ii) Explain about traffic confidentiality (4)

12 a(i) Explain briefly about Diffie-Hellman key exchange (16)

(OR)

b(i) Explain briefly about public key cryptography (8)
(ii) What is the use of RSA algorithm? (8)

13 a(i) Expalin briefly about MD5 message digest algorithm (12)
(ii) What is the use of authentication protocols? (4)

(OR)

b(i) Explain briefly about RIPEMD (16)

14 a(i) Explain about Kerberos (16)

(OR)

b(i) Explain briefly about web security (16)

15 a(i) Discuss the design principles of firewall (8)
(ii) What is meant by password management? (8)

(OR)

b(i) What is meant by virus and explain briefly about threats? (16)