

Cryptography

Ratan Guha

School of Computer Science

University of Central Florida

Orlando, FL 32816

guha@cs.ucf.edu

August 5, 2004

Acknowledgement

These slides have been developed from the slides provided by Prentice Hall Inc. and William Stallings to support Instructors in teaching the class with the book by William Stallings

Text and Recommended Books

- W. Stallings, *Network Security Essentials*, Prentice Hall Inc., 2003.
- **Recommended books**
 - W. Stallings, *Cryptography and Network Security*, Third Edition, Prentice Hall Inc., 2003.
 - D. Stinson, *Cryptography Theory and Practice*, Prentice Hall Inc., 1999.
 - S. Singh, *The Code Book*

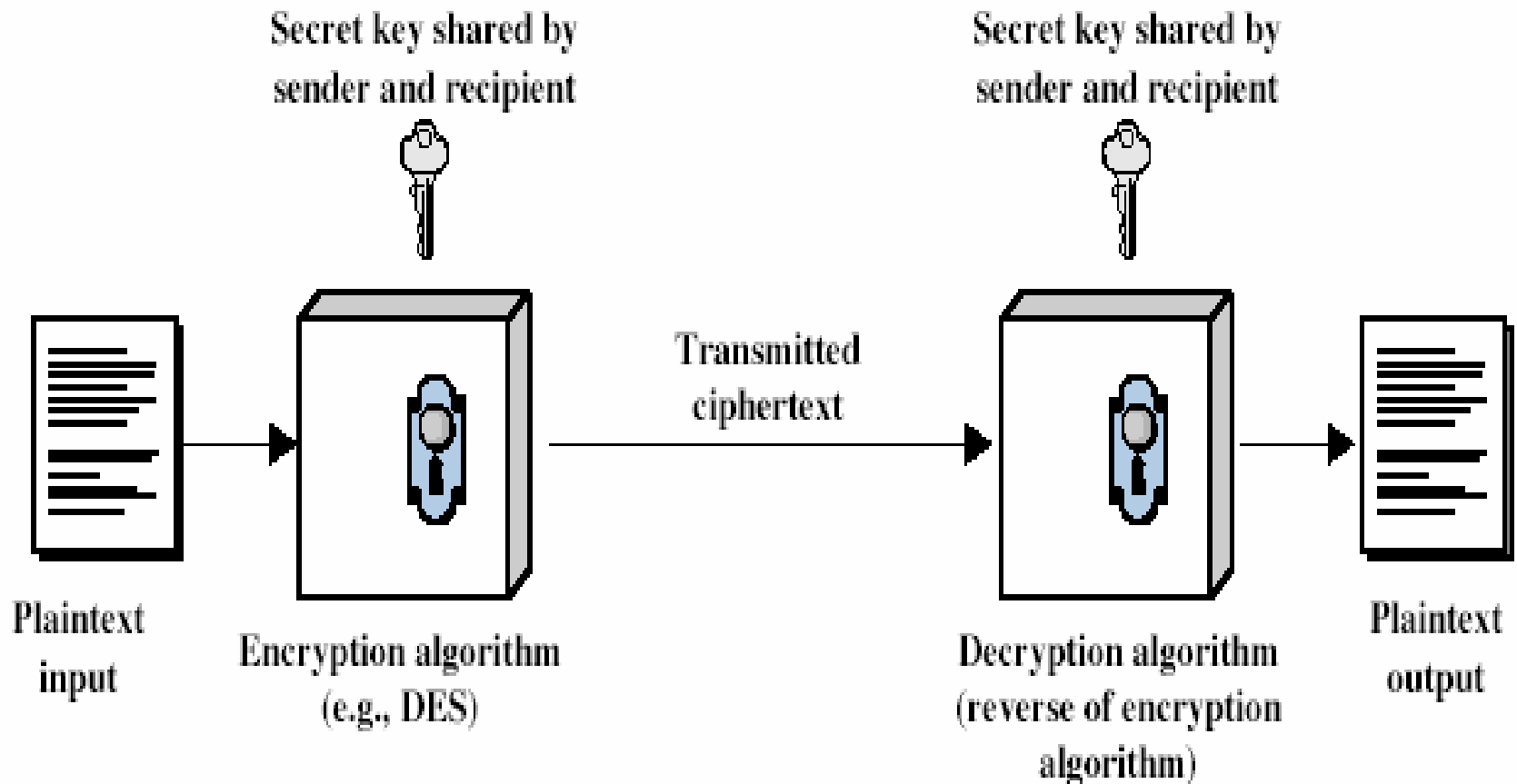
Course Objectives

- Provide a broad understanding of computer networks, and network security
- Discuss contemporary cryptography and its application to confidentiality and authentication, security attacks, detection and defense

Basic Terminology

- **plaintext** - the original message
- **ciphertext** - the coded message
- **cipher** - algorithm for transforming plaintext to ciphertext
- **key** - info used in cipher known only to sender/receiver
- **encipher (encrypt)** - converting plaintext to ciphertext
- **decipher (decrypt)** - recovering ciphertext from plaintext
- **cryptography** - study of encryption principles/methods
- **cryptanalysis (codebreaking)** - the study of principles/methods of deciphering ciphertext *without* knowing key
- **cryptology** - the field of both cryptography and cryptanalysis

Symmetric Cipher Model



Requirements

- two requirements for secure use of symmetric encryption:
 - a strong encryption algorithm
 - a secret key known only to sender / receiver
- $$Y = E_K(X)$$
- $$X = D_K(Y)$$
- assume encryption algorithm is known
 - implies a secure channel to distribute key

Cryptography

- can be characterized by:
 - type of encryption operations used
 - substitution / transposition / product
 - number of keys used
 - single-key or private / two-key or public
 - way in which plaintext is processed
 - block / stream

Types of Cryptanalytic Attacks

- **ciphertext only**
 - only know algorithm / ciphertext, statistical, can identify plaintext
- **known plaintext**
 - know/suspect plaintext & ciphertext to attack cipher
- **chosen plaintext**
 - select plaintext and obtain ciphertext to attack cipher
- **chosen ciphertext**
 - select ciphertext and obtain plaintext to attack cipher
- **chosen text**
 - select either plaintext or ciphertext to en/decrypt to attack cipher

Brute Force Search

- always possible to simply try every key
- most basic attack, proportional to key size
- assume either know / recognise plaintext

Key Size (bits)	Number of Alternative Keys	Time required at 1 encryption/ μ s	Time required at 10^6 encryptions/ μ s
32	$2^{32} = 4.3 \times 10^9$	$2^{31} \mu\text{s} = 35.8$ minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55} \mu\text{s} = 1142$ years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127} \mu\text{s} = 5.4 \times 10^{24}$ years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	$2^{167} \mu\text{s} = 5.9 \times 10^{36}$ years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	$2 \times 10^{26} \mu\text{s} = 6.4 \times 10^{12}$ years	6.4×10^6 years

More Definitions

- **unconditional security**
 - no matter how much computer power is available, the cipher cannot be broken since the ciphertext provides insufficient information to uniquely determine the corresponding plaintext
- **computational security**
 - given limited computing resources (eg time needed for calculations is greater than age of universe), the cipher cannot be broken

Classical Substitution Ciphers

- where letters of plaintext are replaced by other letters or by numbers or symbols
- or if plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with ciphertext bit patterns

Caesar Cipher

- earliest known substitution cipher
- by Julius Caesar
- first attested use in military affairs
- replaces each letter by 3rd letter on
- example:

meet me after the toga party

PHHW PH DIWHU WKH WRJD SDUWB

Caesar Cipher

- can define transformation as:

```
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
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
```

- mathematically give each letter a number

```
a b c d e f g h i j k l m
0 1 2 3 4 5 6 7 8 9 10 11 12
n o p q r s t u v w x y z
13 14 15 16 17 18 19 20 21 22 23 24 25
```

- then have Caesar cipher as:

$$C = E(p) = (p + k) \bmod (26)$$

$$p = D(C) = (C - k) \bmod (26)$$

Cryptanalysis of Caesar Cipher

- only have 26 possible ciphers
 - A maps to A,B,..Z
- could simply try each in turn
- a **brute force search**
- given ciphertext, just try all shifts of letters
- do need to recognize when have plaintext
- eg. break ciphertext "GCUA VQ DTGCM"

Monoalphabetic Cipher

- rather than just shifting the alphabet
- could shuffle (jumble) the letters arbitrarily
- each plaintext letter maps to a different random ciphertext letter
- hence key is 26 letters long

Plain: abcdefghijklmnopqrstuvwxyz

Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN

Plaintext: ifwewishtoreplaceletters

Ciphertext: WIRFRWAJUHYFTSDVFSFUUFYA

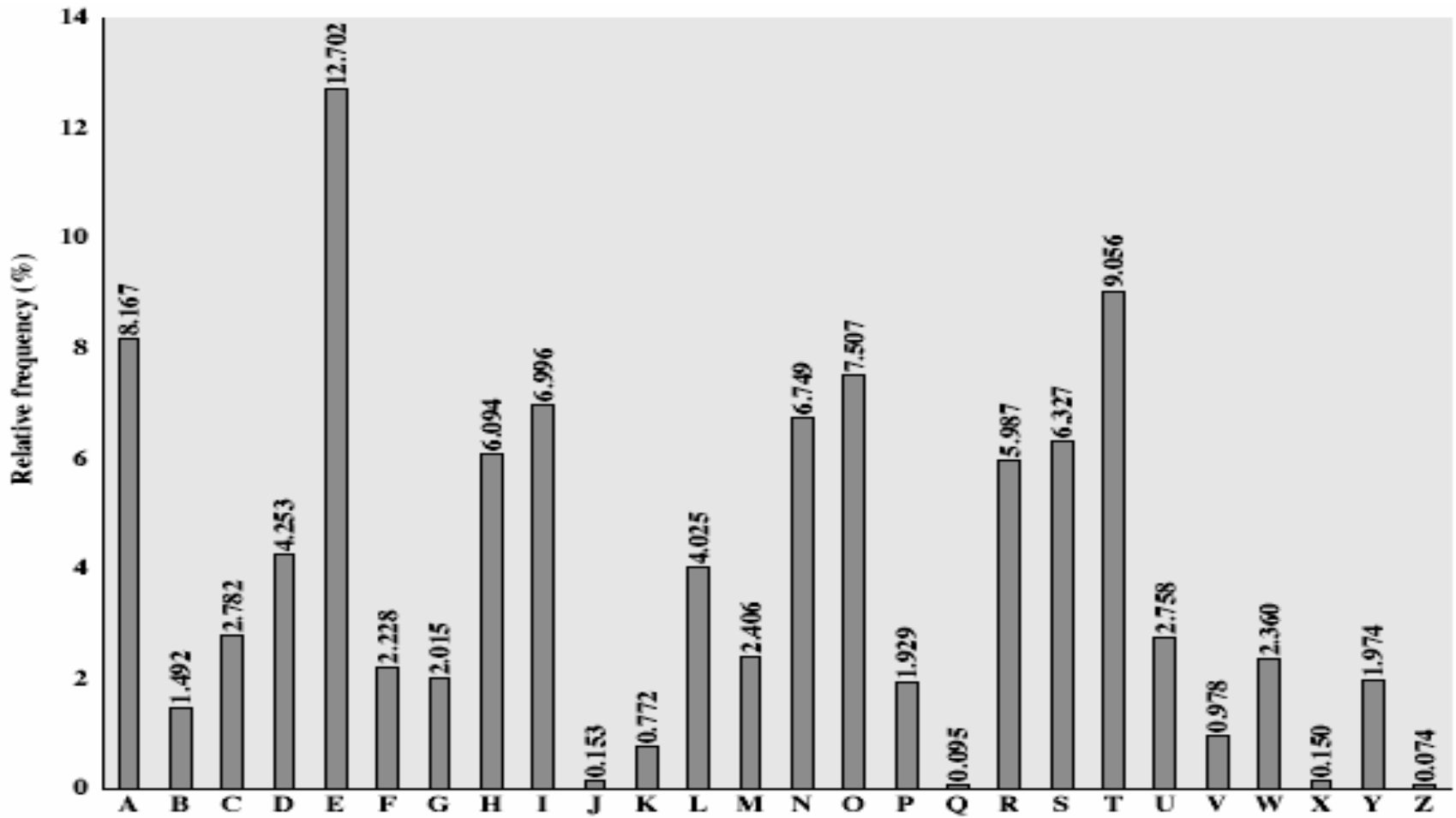
Monoalphabetic Cipher Security

- now have a total of $26! = 4 \times 10^{26}$ keys
- with so many keys, might think is secure
- but would be **!!!WRONG!!!**
- problem is language characteristics

Language Redundancy and Cryptanalysis

- human languages are **redundant**
- eg "th lrd s m shphrd shll nt wnt"
- letters are not equally commonly used
- in English **e** is by far the most common letter
- then T,R,N,I,O,A,S
- other letters are fairly rare
- cf. Z,J,K,Q,X
- have tables of single, double & triple letter frequencies

English Letter Frequencies



Use in Cryptanalysis

- key concept - monoalphabetic substitution ciphers do not change relative letter frequencies
- discovered by Arabian scientists in 9th century
- calculate letter frequencies for ciphertext
- compare counts/plots against known values
- if Caesar cipher look for common peaks/troughs
 - peaks at: A-E-I triple, NO pair, RST triple
 - troughs at: JK, X-Z
- for monoalphabetic must identify each letter
 - tables of common double/triple letters help

Example Cryptanalysis

- given ciphertext:

UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAI Z
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX
EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ

- count relative letter frequencies (see text)
- guess P & Z are e and t
- guess ZW is th and hence ZWP is the
- proceeding with trial and error finally get:

it was disclosed yesterday that several informal but
direct contacts have been made with political
representatives of the viet cong in moscow

Other Ciphers

- Playfair Cipher
- Vigenère Cipher
- Autokey Cipher
- Transposition Ciphers
- Rail Fence cipher
- Row Transposition Ciphers

Playfair Cipher

- not even the large number of keys in a monoalphabetic cipher provides security
- one approach to improving security was to encrypt multiple letters
- the **Playfair Cipher** is an example
- invented by Charles Wheatstone in 1854, but named after his friend Baron Playfair

Playfair Key Matrix

- a 5X5 matrix of letters based on a keyword
- fill in letters of keyword (sans duplicates)
- fill rest of matrix with other letters
- eg. using the keyword MONARCHY

MONAR

CHYBD

EFGIK

LPQST

UVWXZ

Encrypting and Decrypting

- plaintext encrypted two letters at a time:
 1. if a pair is a repeated letter, insert a filler like 'X',
eg. "balloon" encrypts as "ba lx lo on"
 2. if both letters fall in the same row, replace each with letter to right (wrapping back to start from end),
eg. "ar" encrypts as "RM"
 3. if both letters fall in the same column, replace each with the letter below it (again wrapping to top from bottom), eg. "mu" encrypts to "CM"
 4. otherwise each letter is replaced by the one in its row in the column of the other letter of the pair, eg. "hs" encrypts to "BP", and "ea" to "IM" or "JM" (as desired)

Security of the Playfair Cipher

- security much improved over monoalphabetic
- since have $26 \times 26 = 676$ digrams
- would need a 676 entry frequency table to analyse (verses 26 for a monoalphabetic)
- and correspondingly more ciphertext
- was widely used for many years (eg. US & British military in WW1)
- it **can** be broken, given a few hundred letters
- since still has much of plaintext structure

Polyalphabetic Ciphers

- another approach to improving security is to use multiple cipher alphabets
- called **polyalphabetic substitution ciphers**
- makes cryptanalysis harder with more alphabets to guess and flatter frequency distribution
- use a key to select which alphabet is used for each letter of the message
- use each alphabet in turn
- repeat from start after end of key is reached

Vigenère Cipher

- simplest polyalphabetic substitution cipher is the **Vigenère Cipher**
- effectively multiple caesar ciphers
- key is multiple letters long $K = k_1 k_2 \dots k_d$
- i^{th} letter specifies i^{th} alphabet to use
- use each alphabet in turn
- repeat from start after d letters in message
- decryption simply works in reverse

Example

- write the plaintext out
- write the keyword repeated above it
- use each key letter as a caesar cipher key
- encrypt the corresponding plaintext letter
- eg using keyword *deceptive*

key: deceptivedeceptivedeceptive

plaintext: wearediscoveredsaveyourself

ciphertext: ZICVTWQNGRZGVTWAVZHCQYGLMGJ

Aids

- simple aids can assist with en/decryption
- a **Saint-Cyr Slide** is a simple manual aid
 - a slide with repeated alphabet
 - line up plaintext 'A' with key letter, eg 'C'
 - then read off any mapping for key letter
- can bend round into a **cipher disk**
- or expand into a **Vigenère Tableau** (see text Table 2.3)

Security of Vigenère Ciphers

- have multiple ciphertext letters for each plaintext letter
- hence letter frequencies are obscured
- but not totally lost
- start with letter frequencies
 - see if look monoalphabetic or not
- if not, then need to determine number of alphabets, since then can attach each

Kasiski Method

- method developed by Babbage / Kasiski
- repetitions in ciphertext give clues to period
- so find same plaintext an exact period apart
- which results in the same ciphertext
- of course, could also be random fluke
- eg repeated “VTW” in previous example
- suggests size of 3 or 9
- then attack each monoalphabetic cipher individually using same techniques as before

Autokey Cipher

- ideally want a key as long as the message
- Vigenère proposed the **autokey** cipher
- with keyword is prefixed to message as key
- knowing keyword can recover the first few letters
- use these in turn on the rest of the message
- but still have frequency characteristics to attack
- eg. given key *deceptive*

key: deceptivewearediscoveredsave

plaintext: wearediscoveredsaveyourself

ciphertext: ZICVTWQNGKZEIIGASXSTSLVWLA

One-Time Pad

- if a truly random key as long as the message is used, the cipher will be secure
- called a One-Time pad
- is unbreakable since ciphertext bears no statistical relationship to the plaintext
- since for **any plaintext & any ciphertext** there exists a key mapping one to other
- can only use the key **once** though
- have problem of safe distribution of key

Transposition Ciphers

- now consider classical **transposition** or **permutation** ciphers
- these hide the message by rearranging the letter order
- without altering the actual letters used
- can recognise these since have the same frequency distribution as the original text

Rail Fence cipher

- write message letters out diagonally over a number of rows
- then read off cipher row by row
- eg. write message out as:

```
m e m a t r h t g p r y  
e t e f e t e o a a t
```

- giving ciphertext

```
MEMATRHTGPRYETEFETEOAAT
```

Row Transposition Ciphers

- a more complex scheme
- write letters of message out in rows over a specified number of columns
- then reorder the columns according to some key before reading off the rows

Key: 3 4 2 1 5 6 7

Plaintext: a t t a c k p

o s t p o n e

d u n t i l t

w o a m x y z

Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

Product Ciphers

- ciphers using substitutions or transpositions are not secure because of language characteristics
- hence consider using several ciphers in succession to make harder, but:
 - two substitutions make a more complex substitution
 - two transpositions make more complex transposition
 - but a substitution followed by a transposition makes a new much harder cipher
- this is bridge from classical to modern ciphers

Rotor Machines

- before modern ciphers, rotor machines were most common product cipher
- were widely used in WW2
 - German Enigma, Allied Hagelin, Japanese Purple
- implemented a very complex, varying substitution cipher
- used a series of cylinders, each giving one substitution, which rotated and changed after each letter was encrypted
- with 3 cylinders have $26^3=17576$ alphabets

Steganography

- an alternative to encryption
- hides existence of message
 - using only a subset of letters/words in a longer message marked in some way
 - using invisible ink
 - hiding in LSB in graphic image or sound file
- has drawbacks
 - high overhead to hide relatively few info bits

Summary

- have considered:
 - classical cipher techniques and terminology
 - monoalphabetic substitution ciphers
 - cryptanalysis using letter frequencies
 - Playfair ciphers
 - polyalphabetic ciphers
 - transposition ciphers
 - product ciphers and rotor machines
 - stenography

Modern Block Ciphers

- DES (Data Encryption Standard)
- One of the most widely used types of cryptographic algorithms
- provide secrecy and/or authentication services

Block vs Stream Ciphers

- block ciphers process messages in into 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
- hence are focus of course

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

Claude Shannon and Substitution-Permutation Ciphers

- in 1949 Claude Shannon introduced idea of substitution-permutation (S-P) networks
 - modern substitution-transposition product cipher
- these form the basis of modern block ciphers
- S-P networks are based on the two primitive cryptographic operations we have seen before:
 - *substitution* (S-box)
 - *permutation* (P-box)
- provide *confusion* and *diffusion* of message

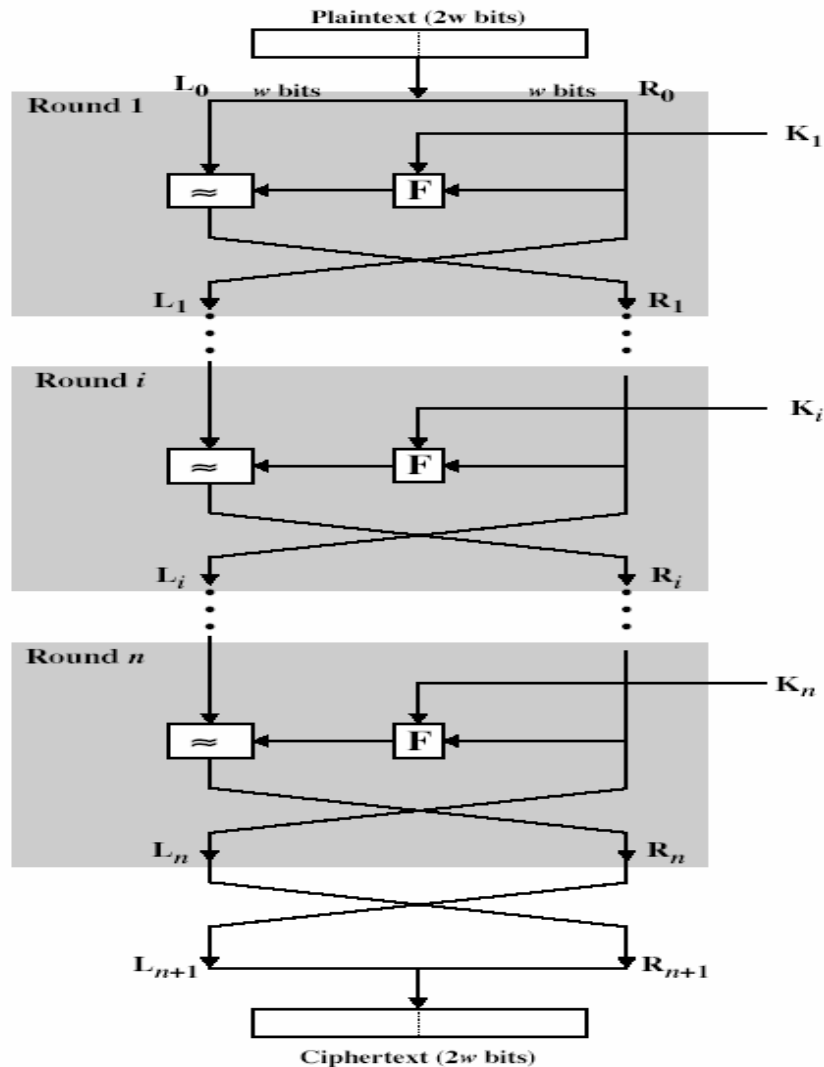
Confusion and Diffusion

- cipher needs to completely obscure statistical properties of original message
- a one-time pad does this
- more practically Shannon suggested combining elements to obtain:
- **diffusion** – dissipates statistical structure of plaintext over bulk of ciphertext
- **confusion** – makes relationship between ciphertext and key as complex as possible

Feistel Cipher Structure

- Horst Feistel devised the **feistel cipher**
 - based on concept of invertible product cipher
- partitions input block into two halves
 - process through multiple rounds which
 - perform a substitution on left data half
 - based on round function of right half & subkey
 - then have permutation swapping halves
- implements Shannon's substitution-permutation network concept

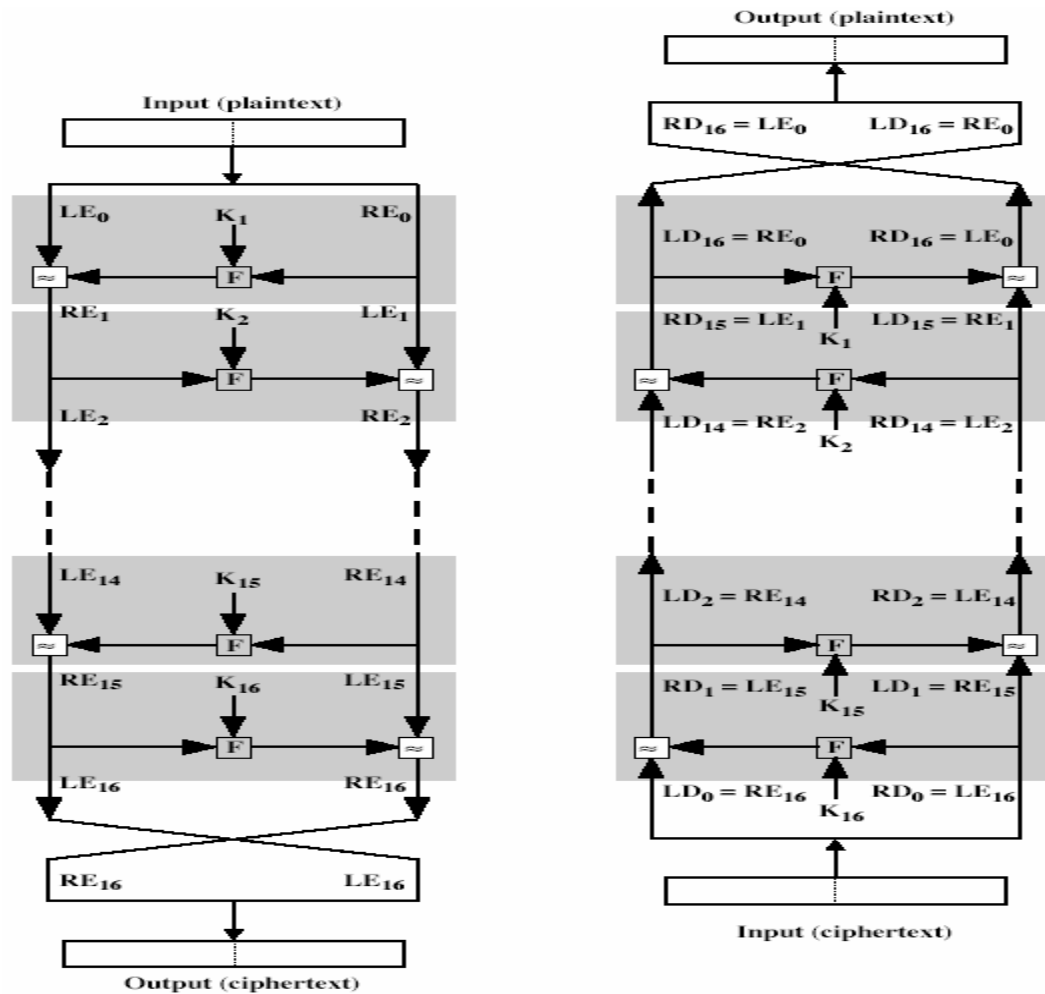
Feistel Cipher Structure



Feistel Cipher Design Principles

- **block size**
 - increasing size improves security, but slows cipher
- **key size**
 - increasing size improves security, makes exhaustive key searching harder, but may slow cipher
- **number of rounds**
 - increasing number improves security, but slows cipher
- **subkey generation**
 - greater complexity can make analysis harder, but slows cipher
- **round function**
 - greater complexity can make analysis harder, but slows cipher
- **fast software en/decryption & ease of analysis**
 - are more recent concerns for practical use and testing

Feistel Cipher Decryption



Data Encryption Standard (DES)

- most widely used block cipher in world
- adopted in 1977 by NBS (now NIST)
 - as FIPS PUB 46
- encrypts 64-bit data using 56-bit key
- has widespread use
- has been considerable controversy over its security

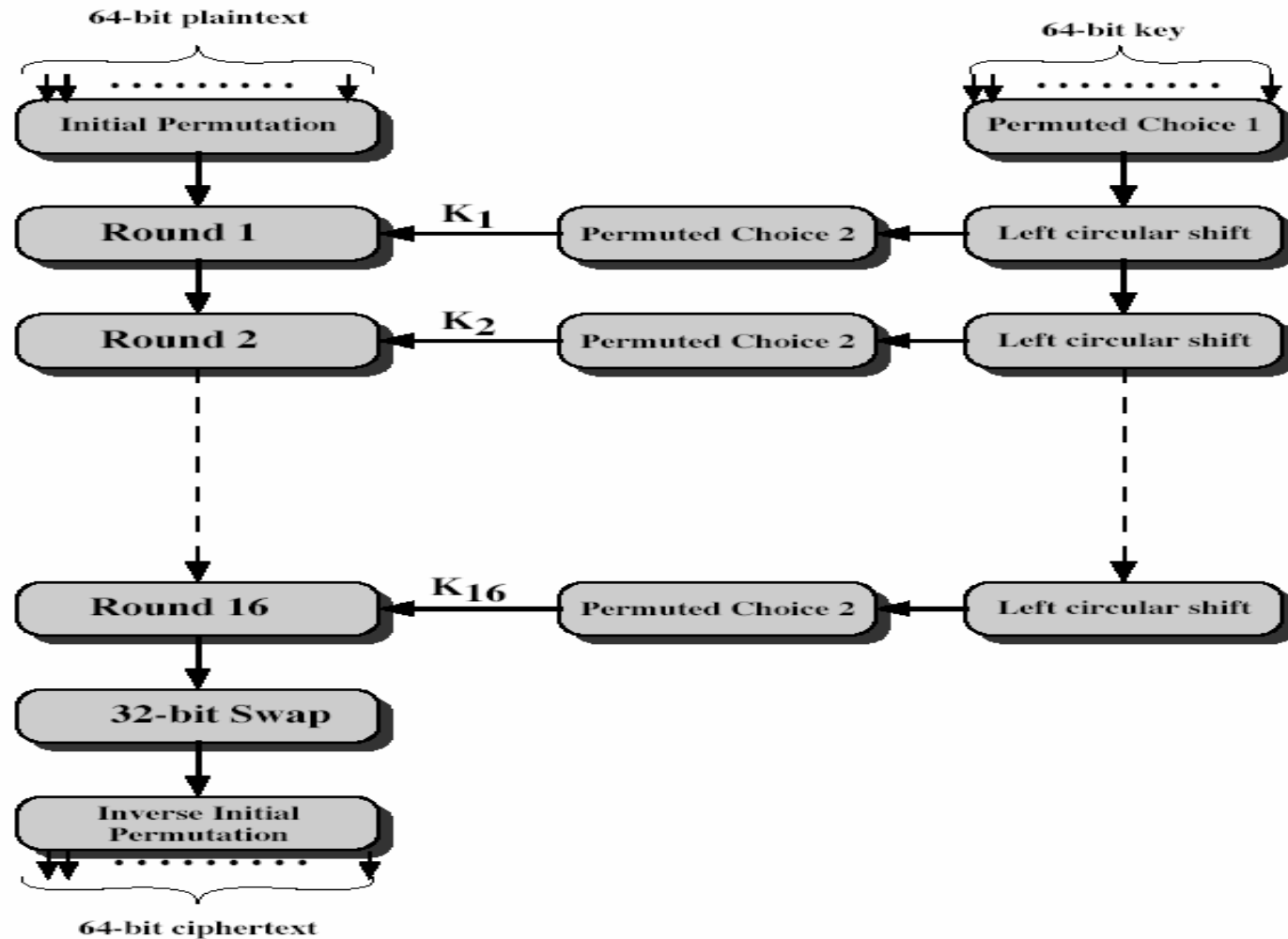
DES History

- IBM developed Lucifer cipher
 - by team led by Feistel
 - used 64-bit data blocks with 128-bit key
- then redeveloped as a commercial cipher with input from NSA and others
- in 1973 NBS issued request for proposals for a national cipher standard
- IBM submitted their revised Lucifer which was eventually accepted as the DES

DES Design Controversy

- although DES standard is public
- was considerable controversy over design
 - in choice of 56-bit key (vs Lucifer 128-bit)
 - and because design criteria were classified
- subsequent events and public analysis show in fact design was appropriate
- DES has become widely used (in financial applications)

DES Encryption



Initial Permutation IP

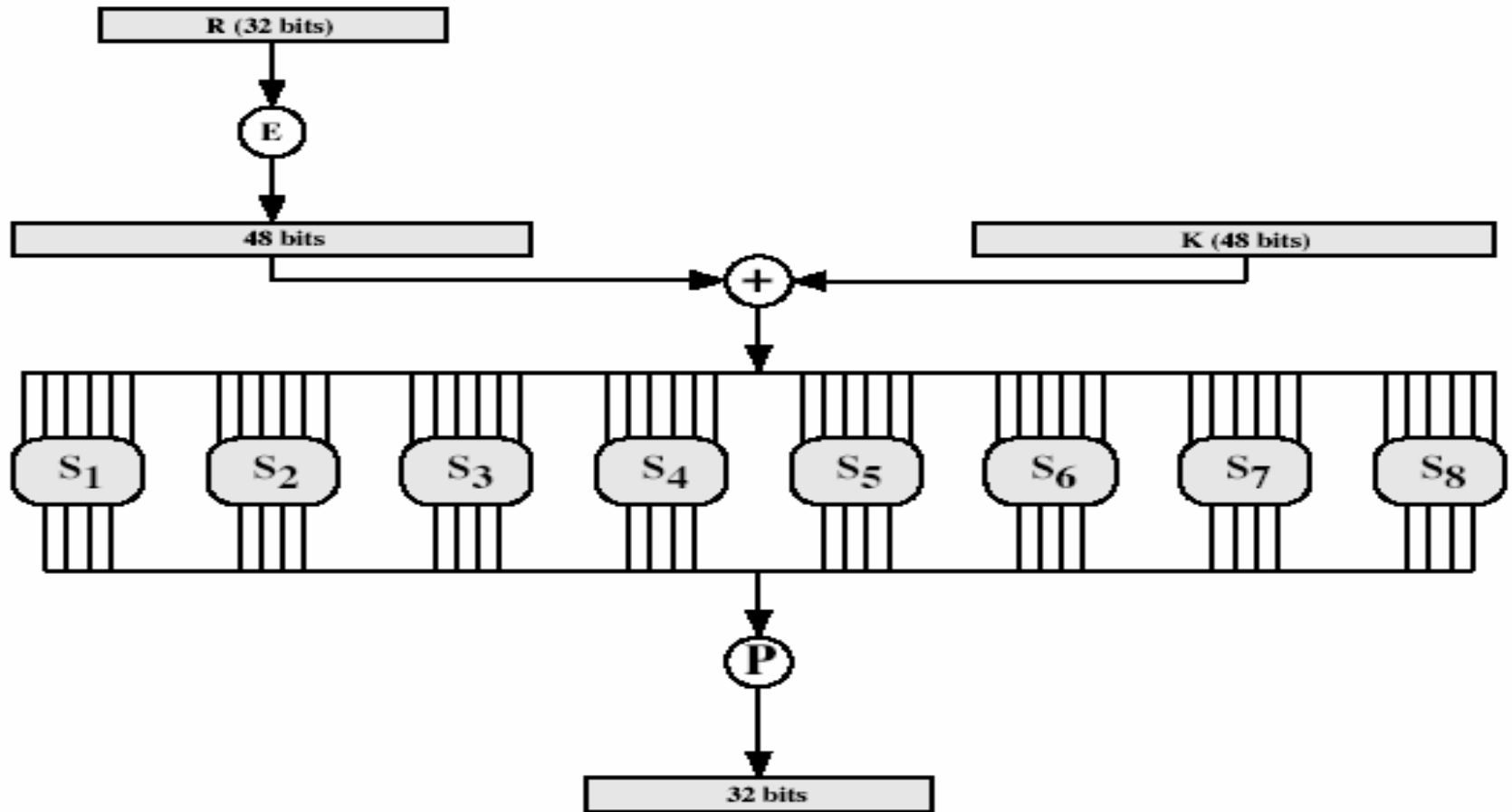
- first step of the data computation
- IP reorders the input data bits
- even bits to LH half, odd bits to RH half
- quite regular in structure (easy in h/w)
- see text Table 3.2
- example:

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

DES Round Structure

- uses two 32-bit L & R halves
- as for any Feistel cipher can describe as:
$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \text{ xor } F(R_{i-1}, K_i)$$
- takes 32-bit R half and 48-bit subkey and:
 - expands R to 48-bits using perm E
 - adds to subkey
 - passes through 8 S-boxes to get 32-bit result
 - finally permutes this using 32-bit perm P

DES Round Structure



Substitution Boxes S

- 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 rows
 - 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
- consists of:
 - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
 - 16 stages consisting of:
 - selecting 24-bits from each half
 - permuting them by PC2 for use in function f ,
 - rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule K**

DES Decryption

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

Avalanche Effect

- key desirable property of encryption alg
- where a change of **one** input or key bit results in changing approx **half** output bits
- making attempts to “home-in” by guessing keys impossible
- DES exhibits strong avalanche

Strength of DES – Key Size

- 56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values
- brute force search looks hard
- recent advances have shown is possible
 - in 1997 on Internet in a few months
 - in 1998 on dedicated h/w (EFF) in a few days
 - in 1999 above combined in 22hrs!
- still must be able to recognize plaintext
- now considering alternatives to DES

Strength of DES – Timing Attacks

- attacks actual implementation of cipher
- use knowledge of consequences of implementation to derive knowledge of some/all subkey bits
- specifically use fact that calculations can take varying times depending on the value of the inputs to it
- particularly problematic on smartcards

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

Differential Cryptanalysis

- one of the most significant recent (public) advances in cryptanalysis
- known by NSA in 70's cf DES design
- Murphy, Biham & Shamir published 1990
- powerful method to analyse block ciphers
- used to analyse most current block ciphers with varying degrees of success
- DES reasonably resistant to it, cf Lucifer

Differential Cryptanalysis

- a statistical attack against Feistel ciphers
- uses cipher structure not previously used
- design of S-P networks has output of function f influenced by both input & key
- hence cannot trace values back through cipher without knowing values of the key
- Differential Cryptanalysis compares two related pairs of encryptions

Differential Cryptanalysis

Compares Pairs of Encryptions

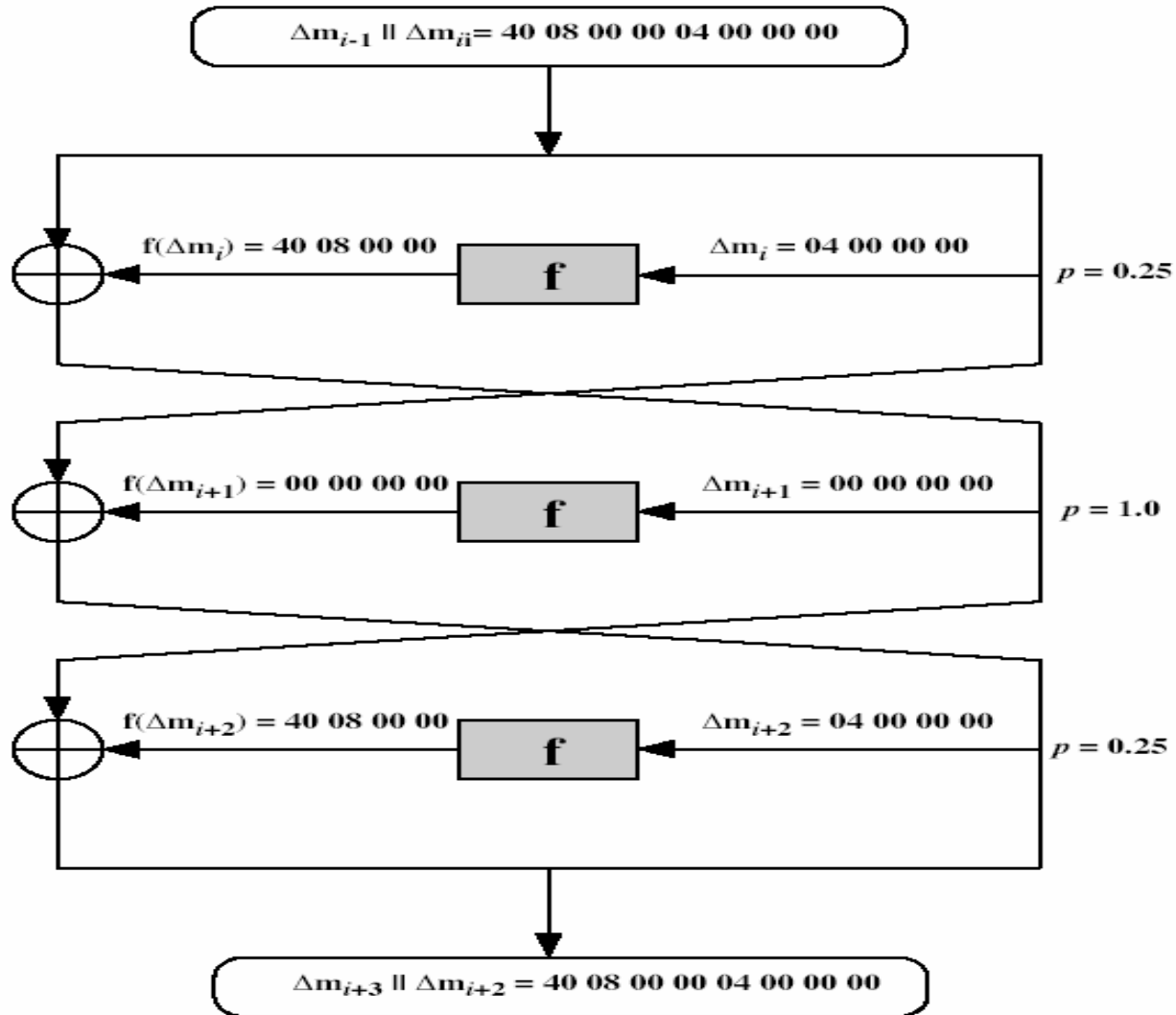
- with a known difference in the input
- searching for a known difference in output
- when same subkeys are used

$$\begin{aligned}\Delta m_{i+1} &= m_{i+1} \oplus m'_{i+1} \\ &= [m_{i-1} \oplus f(m_i, K_i)] \oplus [m'_{i-1} \oplus f(m'_i, K_i)] \\ &= \Delta m_{i-1} \oplus [f(m_i, K_i) \oplus f(m'_i, K_i)]\end{aligned}$$

Differential Cryptanalysis

- have some input difference giving some output difference with probability p
- if find instances of some higher probability input / output difference pairs occurring
- can infer subkey that was used in round
- then must iterate process over many rounds (with decreasing probabilities)

Differential Cryptanalysis



Differential Cryptanalysis

- perform attack by repeatedly encrypting plaintext pairs with known input XOR until obtain desired output XOR
- when found
 - if intermediate rounds match required XOR have a **right pair**
 - if not then have a **wrong pair**, relative ratio is S/N for attack
- can then deduce keys values for the rounds
 - right pairs suggest same key bits
 - wrong pairs give random values
- for large numbers of rounds, probability is so low that more pairs are required than exist with 64-bit inputs
- Biham and Shamir have shown how a 13-round iterated characteristic can break the full 16-round DES

Linear Cryptanalysis

- another recent development
- also a statistical method
- must be iterated over rounds, with decreasing probabilities
- developed by Matsui et al in early 90's
- based on finding linear approximations
- can attack DES with 2^{47} known plaintexts, still in practise infeasible

Linear Cryptanalysis

- find linear approximations with prob $p \neq \frac{1}{2}$

$$P[i_1, i_2, \dots, i_a] (+) C[j_1, j_2, \dots, j_b] = K[k_1, k_2, \dots, k_c]$$

where i_a, j_b, k_c are bit locations in P, C, K

- gives linear equation for key bits
- get one key bit using max likelihood alg
- using a large number of trial encryptions
- effectiveness given by: $|p - \frac{1}{2}|$

Block Cipher Design Principles

- basic principles still like Feistel in 1970's
- number of rounds
 - more is better, exhaustive search best attack
- function f :
 - provides “confusion”, is nonlinear, avalanche
- key schedule
 - complex subkey creation, key avalanche

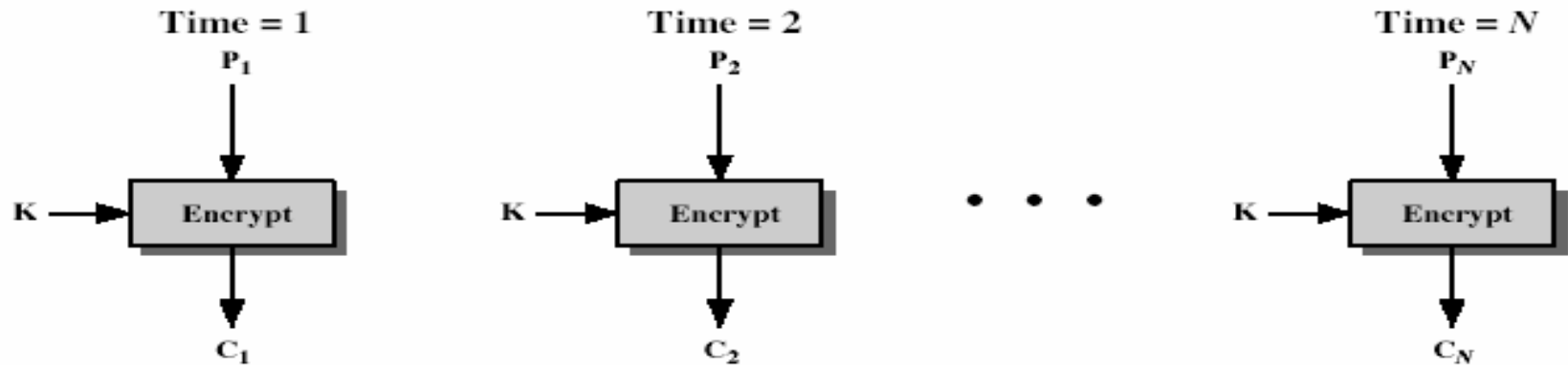
Modes of Operation

- block ciphers encrypt fixed size blocks
- eg. DES encrypts 64-bit blocks, with 56-bit key
- need way to use in practise, given usually have arbitrary amount of information to encrypt
- four were defined for DES in ANSI standard **ANSI X3.106-1983 Modes of Use**
- subsequently now have 5 for DES and AES
- have **block** and **stream** modes

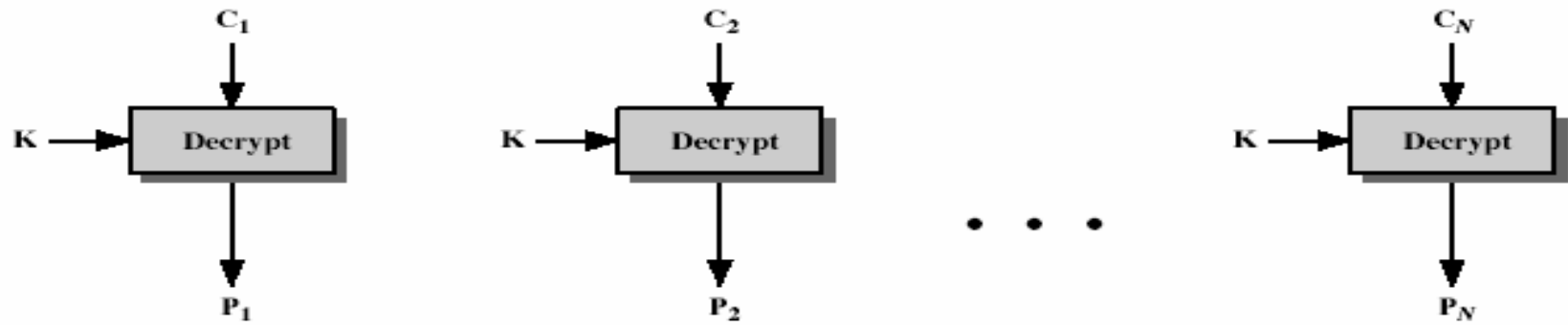
Electronic Codebook Book (ECB)

- 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
- $$C_i = \text{DES}_{K_1}(P_i)$$
- uses: secure transmission of single values

Electronic Codebook Book (ECB)



(a) Encryption



(b) Decryption

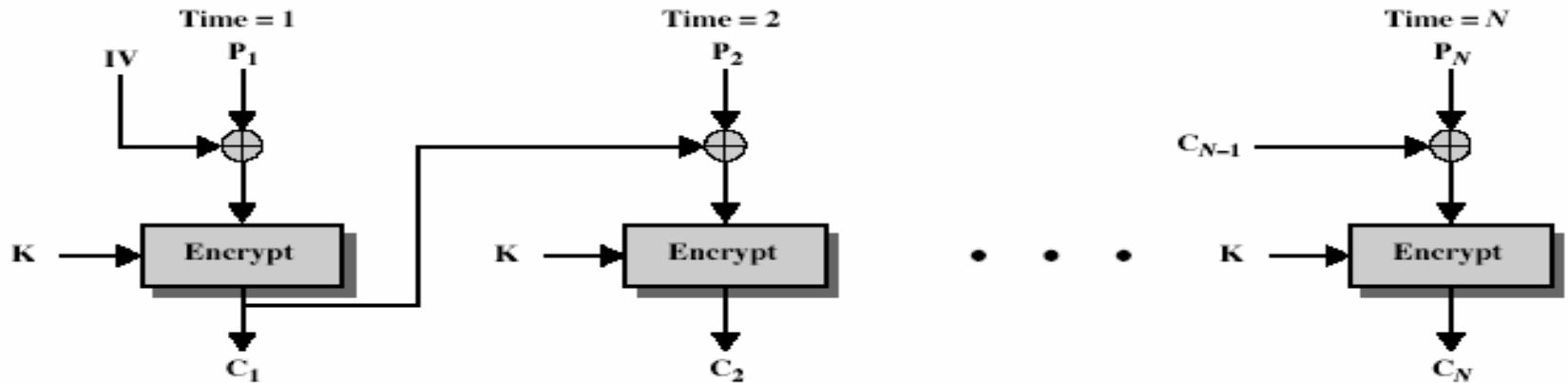
Advantages and Limitations of ECB

- repetitions in message 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 due to encrypted message blocks being independent
- main use is sending a few blocks of data

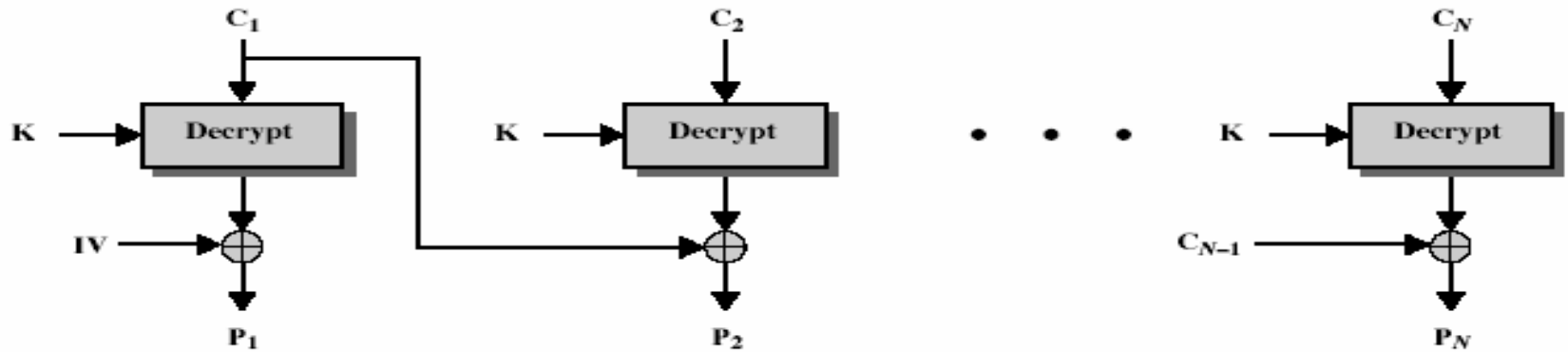
Cipher Block Chaining (CBC)

- message is broken into blocks
 - but these are linked together in the encryption operation
 - each previous cipher blocks is chained with current plaintext block, hence name
 - use Initial Vector (IV) to start process
- $$C_i = \text{DES}_{K1}(P_i \text{ XOR } C_{i-1})$$
- $$C_{-1} = \text{IV}$$
- uses: bulk data encryption, authentication

Cipher Block Chaining (CBC)



(a) Encryption



(b) Decryption

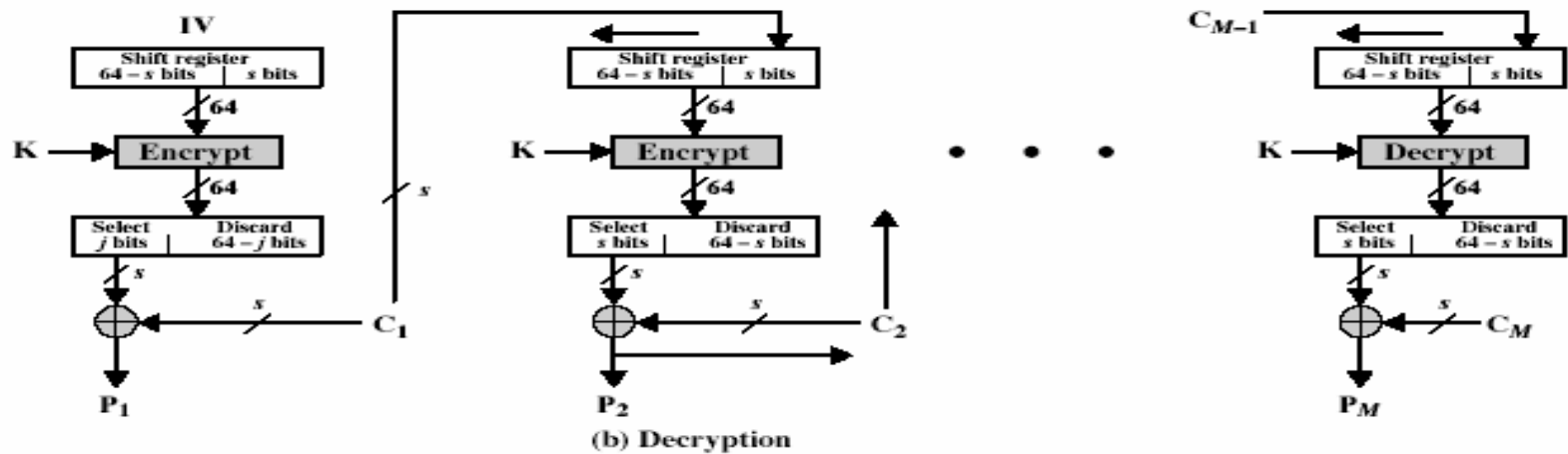
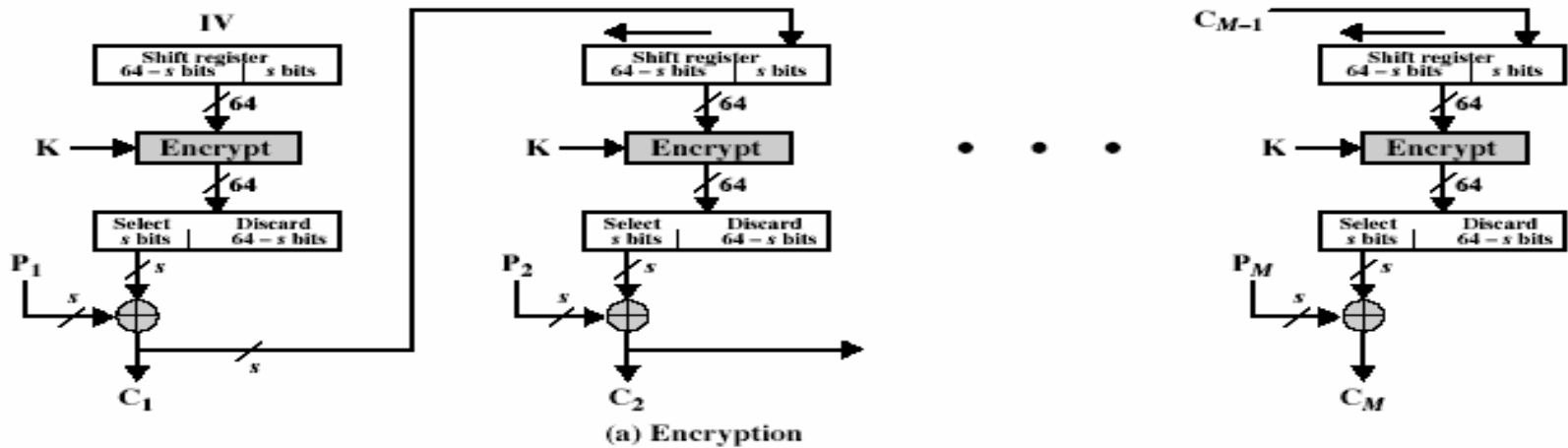
Advantages and Limitations of CBC

- each ciphertext block depends on **all** message blocks
- thus a change in the message affects all ciphertext blocks after the change as well as the original block
- need **Initial Value** (IV) known to sender & receiver
 - however if IV is sent in the clear, an attacker can change bits of the first block, and change IV to compensate
 - hence either IV must be a fixed value (as in EFTPOS) or it must be sent encrypted in ECB mode before rest of message
- at end of message, handle possible last short block
 - by padding either with known non-data value (eg nulls)
 - or pad last block with count of pad size
 - eg. [b1 b2 b3 0 0 0 0 5] <- 3 data bytes, then 5 bytes pad+count

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 or 64 or whatever) to be feed back
 - denoted CFB-1, CFB-8, CFB-64 etc
- is most efficient to use all 64 bits (CFB-64)
 - $C_i = P_i \text{ XOR } \text{DES}_{K1}(C_{i-1})$
 - $C_{-1} = \text{IV}$
- uses: stream data encryption, authentication

Cipher FeedBack (CFB)



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

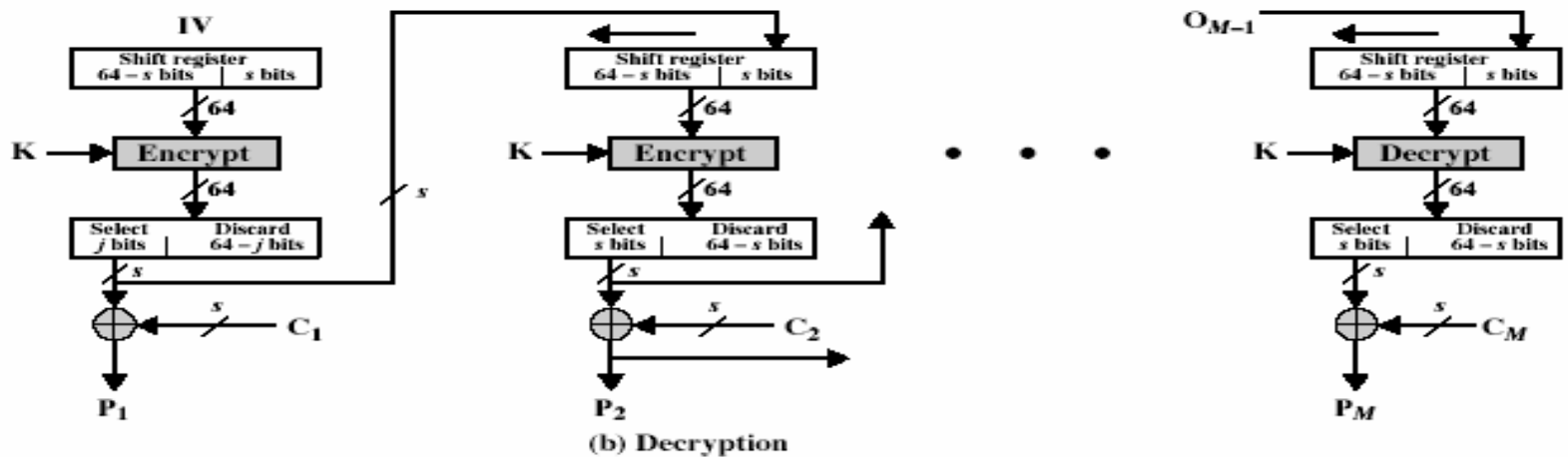
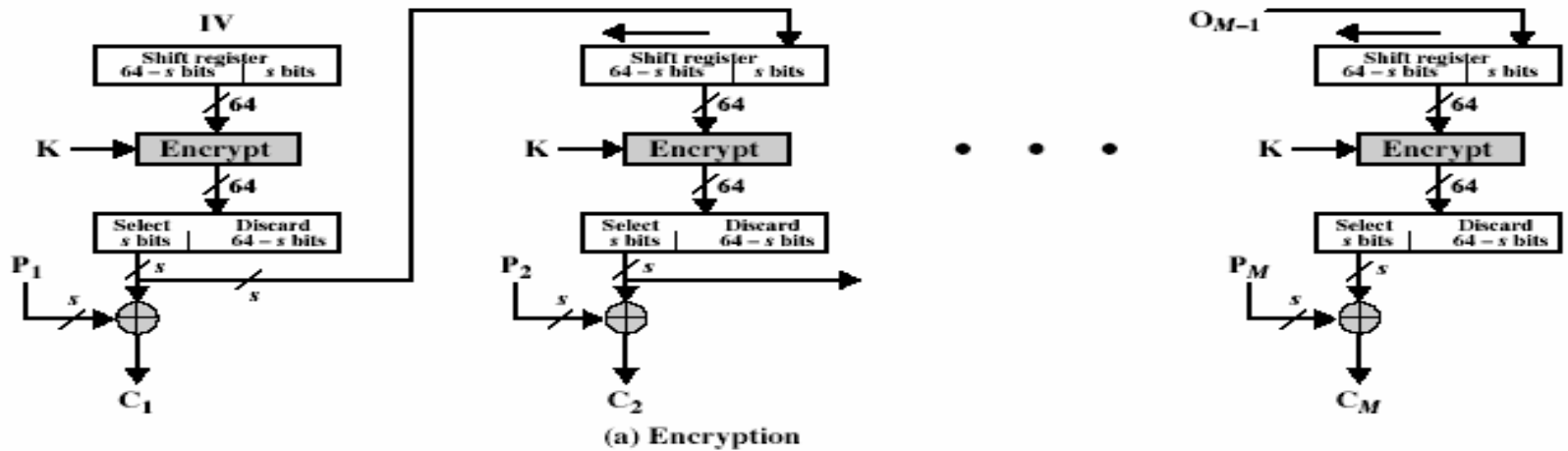
$$C_i = P_i \text{ XOR } O_i$$

$$O_i = \text{DES}_{K1}(O_{i-1})$$

$$O_{-1} = \text{IV}$$

- uses: stream encryption over noisy channels

Output FeedBack (OFB)



Advantages and Limitations of OFB

- used when error feedback a problem or where need to encryptions before message is available
- superficially similar to CFB
- but feedback is from the output of cipher and is independent of message
- a variation of a Vernam cipher
 - hence must **never** reuse the same sequence (key+IV)
- sender and receiver must remain in sync, and some recovery method is needed to ensure this occurs
- originally specified with m-bit feedback in the standards
- subsequent research has shown that only **OFB-64** should ever be used

Counter (CTR)

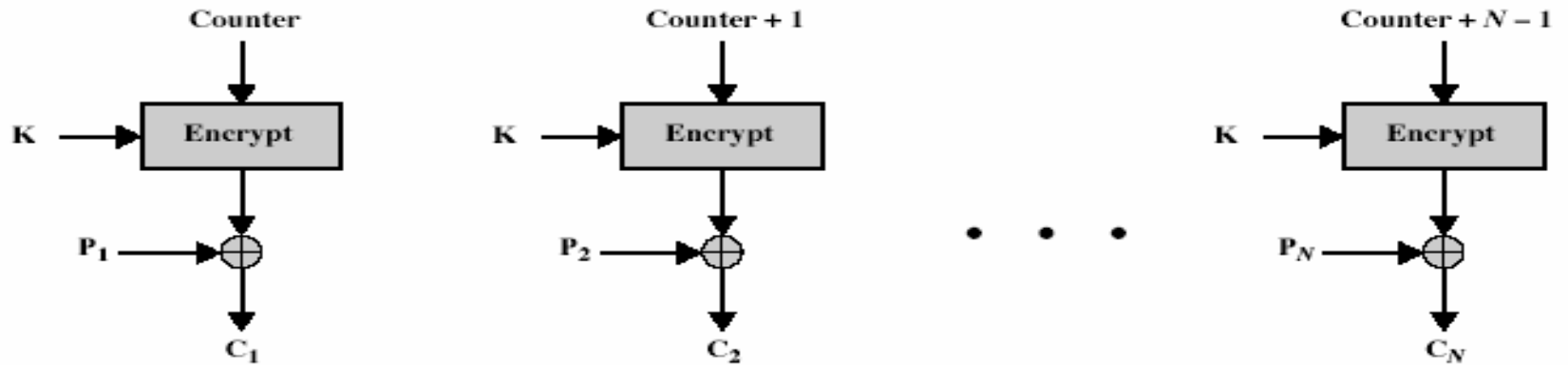
- a “new” mode, though proposed early on
- similar to OFB but encrypts counter value rather than any feedback value
- must have a different key & counter value for every plaintext block (never reused)

$$C_i = P_i \text{ XOR } O_i$$

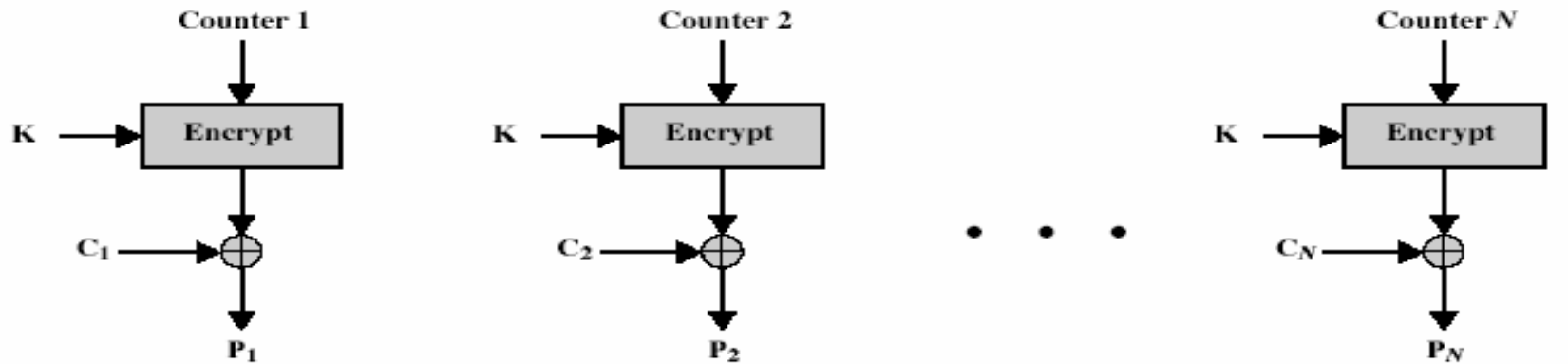
$$O_i = \text{DES}_{K_1}(i)$$

- uses: high-speed network encryptions

Counter (CTR)



(a) Encryption



(b) Decryption

Advantages and Limitations of CTR

- efficiency
 - can do parallel encryptions
 - 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)

Summary

- block cipher design principles
- DES
- Differential & Linear Cryptanalysis
- Modes of Operation
 - ECB, CBC, CFB, OFB, CTR

Finite Fields

- Important in cryptography
 - AES, Elliptic Curve, IDEA, Public Key
- Groups, rings, fields from abstract algebra

Group

- a set of elements or “numbers”
- with some operation whose result is also in the set (closure)
- obeys:
 - associative law: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
 - has identity e : $e \cdot a = a \cdot e = a$
 - has inverses a^{-1} : $a \cdot a^{-1} = e$
- if commutative $a \cdot b = b \cdot a$
 - then forms an **abelian group**

Cyclic Group

- define **exponentiation** as repeated application of operator
 - example: $a^{-3} = a \cdot a \cdot a$
- and let identity be: $e = a^0$
- a group is cyclic if every element is a power of some fixed element
 - ie $b = a^k$ for some a and every b in group
- a is said to be a generator of the group

Ring

- a set of “numbers” with two operations (addition and multiplication) which are:
- an abelian group with addition operation
- multiplication:
 - has closure
 - is associative
 - distributive over addition: $a(b+c) = ab + ac$
- if multiplication operation is commutative, it forms a **commutative ring**
- if multiplication operation has inverses and no zero divisors, it forms an **integral domain**

Field

- a set of numbers with two operations:
 - abelian group for addition
 - abelian group for multiplication (ignoring 0)
 - ring

Modular Arithmetic

- define **modulo operator** $a \bmod n$ to be remainder when a is divided by n
- use the term **congruence** for: $a \equiv b \pmod n$
 - when divided by n , a & b have same remainder
 - eg. $100 \equiv 34 \pmod{11}$
- b is called the **residue** of $a \bmod n$
 - since with integers can always write: $a = qn + b$
- usually have $0 \leq b < n$
 - $-12 \pmod 7 \equiv -5 \pmod 7 \equiv 2 \pmod 7 \equiv 9 \pmod 7$

Modulo 7 Example

...

-21	-20	-19	-18	-17	-16	-15
-14	-13	-12	-11	-10	-9	-8
-7	-6	-5	-4	-3	-2	-1
0	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	32	33	34

...

Divisors

- say a non-zero number b **divides** a if for some m have $a=mb$ (a, b, m all integers)
- that is b divides into a with no remainder
- denote this $b \mid a$
- and say that b is a **divisor** of a
- eg. all of 1,2,3,4,6,8,12,24 divide 24

Modular Arithmetic Operations

- is 'clock arithmetic'
- uses a finite number of values, and loops back from either end
- modular arithmetic is when do addition & multiplication and modulo reduce answer
- can do reduction at any point, ie
 - $a+b \text{ mod } n = [a \text{ mod } n + b \text{ mod } n] \text{ mod } n$

Modular Arithmetic

- can do modular arithmetic with any group of integers: $\mathbb{Z}_n = \{0, 1, \dots, n-1\}$
- form a commutative ring for addition
- with a multiplicative identity
- note some peculiarities
 - if $(a+b) \equiv (a+c) \pmod{n}$ then $b \equiv c \pmod{n}$
 - but $(ab) \equiv (ac) \pmod{n}$ then $b \equiv c \pmod{n}$ only if a is relatively prime to n

Modulo 8 Example

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

(a) Addition modulo 8

Greatest Common Divisor (GCD)

- a common problem in number theory
- GCD (a,b) of a and b is the largest number that divides evenly into both a and b
 - eg $\text{GCD}(60,24) = 12$
- often want **no common factors** (except 1) and hence numbers are **relatively prime**
 - eg $\text{GCD}(8,15) = 1$
 - hence 8 & 15 are relatively prime

Euclid's GCD Algorithm

- an efficient way to find the $\text{GCD}(a,b)$
- uses theorem that:
 - $\text{GCD}(a, b) = \text{GCD}(b, a \bmod b)$
- **Euclid's Algorithm** to compute $\text{GCD}(a,b)$:
 - $A=a, B=b$
 - while $B>0$
 - $R = A \bmod B$
 - $A = B, B = R$
 - return A

Example GCD(1970,1066)

$$1970 = 1 \times 1066 + 904$$

$$1066 = 1 \times 904 + 162$$

$$904 = 5 \times 162 + 94$$

$$162 = 1 \times 94 + 68$$

$$94 = 1 \times 68 + 26$$

$$68 = 2 \times 26 + 16$$

$$26 = 1 \times 16 + 10$$

$$16 = 1 \times 10 + 6$$

$$10 = 1 \times 6 + 4$$

$$6 = 1 \times 4 + 2$$

$$4 = 2 \times 2 + 0$$

$$\text{gcd}(1066, 904)$$

$$\text{gcd}(904, 162)$$

$$\text{gcd}(162, 94)$$

$$\text{gcd}(94, 68)$$

$$\text{gcd}(68, 26)$$

$$\text{gcd}(26, 16)$$

$$\text{gcd}(16, 10)$$

$$\text{gcd}(10, 6)$$

$$\text{gcd}(6, 4)$$

$$\text{gcd}(4, 2)$$

$$\text{gcd}(2, 0)$$

Galois Fields

- finite fields play a key role in cryptography
- can show number of elements in a finite field **must** be a power of a prime p^n
- known as Galois fields
- denoted $GF(p^n)$
- in particular often use the fields:
 - $GF(p)$
 - $GF(2^n)$

Galois Fields $GF(p)$

- $GF(p)$ is the set of integers $\{0, 1, \dots, p-1\}$ with arithmetic operations modulo prime p
- these form a finite field
 - since have multiplicative inverses
- hence arithmetic is “well-behaved” and can do addition, subtraction, multiplication, and division without leaving the field $GF(p)$

Example GF(7)

\times	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

(b) Multiplication modulo 7

Finding Inverses

- can extend Euclid's algorithm:

EXTENDED EUCLID(m, b)

1. $(A1, A2, A3) = (1, 0, m);$

$(B1, B2, B3) = (0, 1, b)$

2. **if** $B3 = 0$

return $A3 = \text{gcd}(m, b);$ no inverse

3. **if** $B3 = 1$

return $B3 = \text{gcd}(m, b); B2 = b^{-1} \text{ mod } m$

4. $Q = A3 \text{ div } B3$

5. $(T1, T2, T3) = (A1 - Q B1, A2 - Q B2, A3 - Q B3)$

6. $(A1, A2, A3) = (B1, B2, B3)$

7. $(B1, B2, B3) = (T1, T2, T3)$

8. **goto** 2

Inverse of 550 in GF(1759)

Q	A1	A2	A3	B1	B2	B3
—	1	0	1759	0	1	550
3	0	1	550	1	-3	109
5	1	-3	109	-5	16	5
21	-5	16	5	106	-339	4
1	106	-339	4	-111	355	1

Polynomial Arithmetic

- can compute using polynomials

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 = \sum_{i=0}^n a_i x^i$$

- several alternatives available
 - ordinary polynomial arithmetic
 - poly arithmetic with coords mod p
 - poly arithmetic with coords mod p and polynomials mod $M(x)$

Ordinary Polynomial Arithmetic

- add or subtract corresponding coefficients
- multiply all terms by each other
- eg

– let $f(x) = x^3 + x^2 + 2$ and $g(x) = x^2 - x + 1$

$$f(x) + g(x) = x^3 + 2x^2 - x + 3$$

$$f(x) - g(x) = x^3 + x + 1$$

$$f(x) \times g(x) = x^5 + 3x^2 - 2x + 2$$

Polynomial Arithmetic with Modulo Coefficients

- when computing value of each coefficient do calculation modulo some value
- could be modulo any prime
- but we are most interested in mod 2
 - ie all coefficients are 0 or 1
 - eg. let $f(x) = x^3 + x^2$ and $g(x) = x^2 + x + 1$
 - $f(x) + g(x) = x^3 + x + 1$
 - $f(x) \times g(x) = x^5 + x^2$

Modular Polynomial Arithmetic

- can write any polynomial in the form:
 - $f(x) = q(x) g(x) + r(x)$
 - can interpret $r(x)$ as being a remainder
 - $r(x) = f(x) \bmod g(x)$
- if have no remainder say $g(x)$ divides $f(x)$
- if $g(x)$ has no divisors other than itself & 1 say it is **irreducible** (or prime) polynomial
- arithmetic modulo an irreducible polynomial forms a field

Polynomial GCD

- can find greatest common divisor for polys
 - $c(x) = \text{GCD}(a(x), b(x))$ if $c(x)$ is the poly of greatest degree which divides both $a(x), b(x)$
 - can adapt Euclid's Algorithm to find it:
 - EUCLID[$a(x), b(x)$]
 1. $A(x) = a(x); B(x) = b(x)$
 2. **if** $B(x) = 0$ **return** $A(x) = \text{gcd}[a(x), b(x)]$
 3. $R(x) = A(x) \bmod B(x)$
 4. $A(x) \leftarrow B(x)$
 5. $B(x) \leftarrow R(x)$
 6. **goto** 2

Modular Polynomial Arithmetic

- can compute in field $GF(2^n)$
 - polynomials with coefficients modulo 2
 - whose degree is less than n
 - hence must reduce modulo an irreducible poly of degree n (for multiplication only)
- form a finite field
- can always find an inverse
 - can extend Euclid's Inverse algorithm to find

Example GF(2³)

Table 4.6 Polynomial Arithmetic Modulo ($x^3 + x + 1$)

		000	001	010	011	100	101	110	111
	+	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
000	0	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
001	1	1	0	$x+1$	x	x^2+1	x^2	x^2+x+1	x^2+x
010	x	x	$x+1$	0	1	x^2+x	x^2+x+1	x^2	x^2+1
011	$x+1$	$x+1$	x	1	0	x^2+x+1	x^2+x	x^2+1	x^2
100	x^2	x^2	x^2+1	x^2+x	x^2+x+1	0	1	x	$x+1$
101	x^2+1	x^2+1	x^2	x^2+x+1	x^2+x	1	0	$x+1$	x
110	x^2+x	x^2+x	x^2+x+1	x^2	x^2+1	x	$x+1$	0	1
111	x^2+x+1	x^2+x+1	x^2+x	x^2+1	x^2	$x+1$	x	1	0

(a) Addition

		000	001	010	011	100	101	110	111
	×	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
000	0	0	0	0	0	0	0	0	0
001	1	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
010	x	0	x	x^2	x^2+x	$x+1$	1	x^2+x+1	x^2+1
011	$x+1$	0	$x+1$	x^2+x	x^2+1	x^2+x+1	x^2	1	x
100	x^2	0	x^2	$x+1$	x^2+x+1	x^2+x	x	x^2+1	1
101	x^2+1	0	x^2+1	1	x^2	x	x^2+x+1	$x+1$	x^2+x
110	x^2+x	0	x^2+x	x^2+x+1	1	x^2+1	$x+1$	x	x^2
111	x^2+x+1	0	x^2+x+1	x^2+1	x	1	x^2+x	x^2	$x+1$

(b) Multiplication

Computational Considerations

- since coefficients are 0 or 1, can represent any such polynomial as a bit string
- addition becomes XOR of these bit strings
- multiplication is shift & XOR
 - cf long-hand multiplication
- modulo reduction done by repeatedly substituting highest power with remainder of irreducible poly (also shift & XOR)