
Classical Cryptography

CS498SH

Fall 2006

Chapter 9: Classic Cryptography

- Historical Cryptography
- Symmetric Key Cryptography

Reading

- Chapter 9 from *Computer Science: Art and Science*
- *Applied Cryptography*, Bruce Schneier

Overview

- Classical Cryptography
 - Cæsar cipher
 - Vigènere cipher
- Symmetric Key Cryptography
 - DES
 - AES

Cryptosystem

- Is the basic component of cryptography
- 5-tuple $(\mathbf{E}, \mathbf{D}, \mathbf{M}, \mathbf{K}, \mathbf{C})$
 - \mathbf{M} set of plaintexts
 - \mathbf{K} set of keys
 - \mathbf{C} set of ciphertexts
 - \mathbf{E} set of encryption functions $e: \mathbf{M} \times \mathbf{K} \rightarrow \mathbf{C}$
 - \mathbf{D} set of decryption functions $d: \mathbf{C} \times \mathbf{K} \rightarrow \mathbf{M}$

Example

- Example: Cæsar cipher (The most basic cipher)
 - $\mathcal{M} = \{ \text{sequences of letters} \}$
 - $\mathcal{K} = \{ i \mid i \text{ is an integer and } 0 \leq i \leq 25 \}$
 - $\mathcal{E} = \{ E_k \mid k \in \mathcal{K} \text{ and for all letters } m, \\ E_k(m) = (m + k) \bmod 26 \}$
 - $\mathcal{D} = \{ D_k \mid k \in \mathcal{K} \text{ and for all letters } c, \\ D_k(c) = (26 + c - k) \bmod 26 \}$
 - $\mathcal{C} = \mathcal{M}$

Attacks

- Opponent whose goal is to break cryptosystem is the *adversary*
 - Standard cryptographic practice: Assume adversary knows algorithm used, but not the key
- Three types of attacks:
 - *ciphertext only*: adversary has only ciphertext; goal is to find plaintext, possibly key
 - *known plaintext*: adversary has ciphertext, corresponding plaintext; goal is to find key
 - *chosen plaintext*: adversary may supply plaintexts and obtain corresponding ciphertext; goal is to find key

Basis for Attacks

- Mathematical attacks
 - Based on analysis of underlying mathematics
- Statistical attacks
 - Make assumptions about the distribution of letters, pairs of letters (digrams), triplets of letters (trigrams), *etc.*
 - Called *models of the language*
 - E.g. Caesar Cipher, letter E
 - Examine ciphertext, correlate properties with the assumptions.

Classical Cryptography

- Sender, receiver share common key
 - Keys may be the same, or trivial to derive from one another
 - Sometimes called *symmetric cryptography*
- Two basic types
 - Transposition ciphers
 - Substitution ciphers
 - Combinations are called *product ciphers*

Transposition Cipher

- Rearrange letters in plaintext to produce ciphertext
- Example (Rail-Fence Cipher)
 - Plaintext is HELLO WORLD
 - Rearrange as
HLOOL
ELWRD
 - Ciphertext is HLOOL ELWRD

Attacking the Cipher

- Anagramming
 - If 1-gram frequencies match English frequencies, but other n -gram frequencies do not, probably transposition
 - Rearrange letters to form n -grams with highest frequencies

Example

- Ciphertext: HLOOLELWRD
- Frequencies of 2-grams beginning with H
 - HE 0.0305
 - HO 0.0043
 - HL, HW, HR, HD < 0.0010
- Frequencies of 2-grams ending in H
 - WH 0.0026
 - EH, LH, OH, RH, DH ≤ 0.0002
- Implies E follows H

Example

- Arrange so the H and E are adjacent

HE

LL

OW

OR

LD

- Read off across, then down, to get original plaintext

Substitution Ciphers

- Change characters in plaintext to produce ciphertext
- Example (Cæsar cipher)
 - Plaintext is HELLO WORLD
 - Change each letter to the third letter following it (X goes to A, Y to B, Z to C)
 - Key is 3, usually written as letter ‘D’
 - Ciphertext is KHOOR ZRUOG

Attacking the Cipher

- Exhaustive search
 - If the key space is small enough, try all possible keys until you find the right one
 - Cæsar cipher has 26 possible keys
- Statistical analysis
 - Compare to 1-gram model of English

Statistical Attack

- Compute frequency of each letter in ciphertext:

G 0.1 H 0.1 K 0.1 O 0.3

R 0.2 U 0.1 Z 0.1

- Apply 1-gram model of English
 - Frequency of characters (1-grams) in English is on next slide

Character Frequencies

a	0.080	h	0.060	n	0.070	t	0.090
b	0.015	i	0.065	o	0.080	u	0.030
c	0.030	j	0.005	p	0.020	v	0.010
d	0.040	k	0.005	q	0.002	w	0.015
e	0.130	l	0.035	r	0.065	x	0.005
f	0.020	m	0.030	s	0.060	y	0.020
g	0.015					z	0.002

Statistical Analysis

- $f(c)$ frequency of character c in ciphertext

$\forall \varphi(i)$ correlation of frequency of letters in ciphertext with corresponding letters in English, assuming key is i

$\varphi(i) = \sum_{0 \leq c \leq 25} f(c)p(c - i)$ so here,

$$\varphi(i) = 0.1p(6 - i) + 0.1p(7 - i) + 0.1p(10 - i) + 0.3p(14 - i) + 0.2p(17 - i) + 0.1p(20 - i) + 0.1p(25 - i)$$

- $p(x)$ is frequency of character x in English

Correlation: $\varphi(i)$ for $0 \leq i \leq 25$

i	$\varphi(i)$	i	$\varphi(i)$	i	$\varphi(i)$	i	$\varphi(i)$
0	0.0482	7	0.0442	13	0.0520	19	0.0315
1	0.0364	8	0.0202	14	0.0535	20	0.0302
2	0.0410	9	0.0267	15	0.0226	21	0.0517
3	0.0575	10	0.0635	16	0.0322	22	0.0380
4	0.0252	11	0.0262	17	0.0392	23	0.0370
5	0.0190	12	0.0325	18	0.0299	24	0.0316
6	0.0660					25	0.0430

The Result

- Most probable keys, based on φ :
 - $i = 6$, $\varphi(i) = 0.0660$
 - plaintext EB IIL TLOLA
 - $i = 10$, $\varphi(i) = 0.0635$
 - plaintext AXEEH PHKEW
 - $i = 3$, $\varphi(i) = 0.0575$
 - plaintext HELLO WORLD
 - $i = 14$, $\varphi(i) = 0.0535$
 - plaintext WTAAD LDGAS
- Only English phrase is for $i = 3$
 - That's the key (3 or 'D')

Cæsar's Problem

- Key is too short
 - Can be found by exhaustive search
 - Statistical frequencies not concealed well
 - They look too much like regular English letters
- So make it longer
 - Multiple letters in key
 - Idea is to smooth the statistical frequencies to make cryptanalysis harder

Vigènere Cipher

- Like Cæsar cipher, but use a phrase as key
- Example
 - Message THE BOY HAS THE BALL
 - Key VIG
 - Encipher using Cæsar cipher for each letter:

key	VIGVIGVIGVIGVIGV
plain	THEBOYHASTHEBALL
cipher	OPKWWECIYOPKWIRG

Relevant Parts of Tableau

	<i>G</i>	<i>I</i>	<i>V</i>
<i>A</i>	<i>G</i>	<i>I</i>	<i>V</i>
<i>B</i>	<i>H</i>	<i>J</i>	<i>W</i>
<i>E</i>	<i>L</i>	<i>M</i>	<i>Z</i>
<i>H</i>	<i>N</i>	<i>P</i>	<i>C</i>
<i>L</i>	<i>R</i>	<i>T</i>	<i>G</i>
<i>O</i>	<i>U</i>	<i>W</i>	<i>J</i>
<i>S</i>	<i>Y</i>	<i>A</i>	<i>N</i>
<i>T</i>	<i>Z</i>	<i>B</i>	<i>O</i>
<i>Y</i>	<i>E</i>	<i>H</i>	<i>T</i>

- Tableau shown has relevant rows, columns only
- Example encipherments(?):
 - key V, letter T: follow V column down to T row (giving “O”)
 - Key I, letter H: follow I column down to H row (giving “P”)

Useful Terms

- *period*: length of key
 - In earlier example, period is 3
- *tableau*: table used to encipher and decipher
 - Vigènere cipher has key letters on top, plaintext letters on the left
- *polyalphabetic*: the key has several different letters
 - Cæsar cipher is monoalphabetic

Attacking the Cipher

- Approach
 - Establish period; call it n
 - Break message into n parts, each part being enciphered using the same key letter
 - Solve each part
- We will show each step
- Automated in applet
 - <http://math.ucsd.edu/~crypto/java/EARLYCIPHERS/Vigenere.html>

The Target Cipher

- We want to break this cipher:

ADQYS MIUSB OXKKT MIBHK IZOOO
EQOOG IFBAG KAUMF VVTAA CIDTW
MOCIO EQOOG BMBFV ZGGWP CIEKQ
HSNEW VECNE DLAAV RWKXS VNSVP
HCEUT QOIOF MEGJS WTPCH AJMOC
HIUIX

Establish Period

- *Kasiski: repetitions in the ciphertext occur when characters of the key appear over the same characters in the plaintext*

- **Example:**

key VIGVIGVIGVIGVIGV

plain THEBOYHASTHEBALL

cipher OPKWWECIYOPKWIRG

Note the key and plaintext line up over the repetitions (underlined). As distance between repetitions is 9, the period is a factor of 9 (that is, 1, 3, or 9)

Repetitions in Example

<i>Letters</i>	<i>Start</i>	<i>End</i>	<i>Distance</i>	<i>Factors</i>
MI	5	15	10	2, 5
OO	22	27	5	5
OEQOOG	24	54	30	2, 3, 5
FV	39	63	24	2, 2, 2, 3
AA	43	87	44	2, 2, 11
MOC	50	122	72	2, 2, 2, 3, 3
QO	56	105	49	7, 7
PC	69	117	48	2, 2, 2, 2, 3
NE	77	83	6	2, 3
SV	94	97	3	3
CH	118	124	6	2, 3

Estimate of Period

- OEQOOG is probably not a coincidence
 - It's too long for that
 - Period may be 1, 2, 3, 5, 6, 10, 15, or 30
- Most others (7/10) have 2 in their factors
- Almost as many (6/10) have 3 in their factors
- Begin with period of $2 \times 3 = 6$

Check on Period

- Index of coincidence is probability that two randomly chosen letters from ciphertext will be the same
- Tabulated for different periods:

1	0.066	3	0.047	5	0.044
2	0.052	4	0.045	10	0.041
Large	0.038				

Compute IC

- $IC = [n (n - 1)]^{-1} \sum_{0 \leq i \leq 25} [F_i (F_i - 1)]$
 - where n is length of ciphertext and F_i the number of times character i occurs in ciphertext
- Here, $IC = 0.043$
 - Indicates a key of slightly more than 5
 - This is a statistical measure, so it can be an error, but it agrees with the previous estimate (which was 6)

Splitting Into Alphabets

alphabet 1: AIKHOIATTOBGEEERNEOSAI

alphabet 2: DUKKEFUAWEMGKWDWSUFWJU

alphabet 3: QSTIQBMAMQBWQVLKVTMTMI

alphabet 4: YBMZOAFCCOFPHEAXPQEPOX

alphabet 5: SOIOOGVICOVCSVASHOGCC

alphabet 6: MXBOGKVDIGZINNVVCIJHH

- ICs (#1, 0.069; #2, 0.078; #3, 0.078; #4, 0.056; #5, 0.124; #6, 0.043) indicate all alphabets have period 1, except #4 and #6; consider them as the error of statistics

Frequency Examination

ABCDEFGHIJKLMNOPQRSTUVWXYZ

1 31004011301001300112000000

2 10022210013010000010404000

3 12000000201140004013021000

4 21102201000010431000000211

5 10500021200000500030020000

7 01110022311012100000030101

Letter frequencies are (H high, M medium, L low):

HMMMHHMMHHMMMMHMLHHHMLLLLLL

Begin Decryption

- First matches characteristics of unshifted alphabet
- Third matches if I shifted to A
- Sixth matches if V shifted to A
- Substitute into ciphertext (bold are substitutions)

ADIYS RIUKB OCKKL MIGHK AZOTO
EIOOL IFTAG PAUEF VATAS CIITW
EOCNO EIOOL BMTFV EGGOP CNEKI
HSSEW NECSE DDAAA RWCXS ANSNP
HHEUL QONOF EEGOS WLPCM AJEOC
MIUAX

Look For Clues

- **AJE** in last line suggests “are”, meaning second alphabet maps A into S:

ALIYS RICKB OCKSL MIGHS AZOTO

MIOOL INTAG PACEF VATIS CIITE

EOCNO MIOOL BUTFV EGOOP CNESI

HSSEE NECSE LDAAA RECXS ANANP

HHECL QONON EEGOS ELPCM AREOC

MICAX

Next Alphabet

- **MICAX** in last line suggests “mical” (a common ending for an adjective), meaning fourth alphabet maps O into A:

ALIMS RICKP OCKSL AIGHS ANOTO
MICOL INTOG PACET VATIS QIITE
ECCNO MICOL BUTTV EGOOD CNESI
VSSEE NSCSE LDOAA RECLS ANAND
HHECL EONON ESGOS ELDCM ARECC
MICAL

Got It!

- QI means that U maps into I, as Q is always followed by U...So we get the key for the fifth alphabet:

**ALIME RICKP ACKSL AUGHS ANATO
MICAL INTOS PACET HATIS QUITE
ECONO MICAL BUTTH EGOOD ONESI
VESEE NSOSE LDOMA RECLE ANAND
THECL EANON ESSOS ELDOM ARECO
MICAL**

One-Time Pad

- A Vigenère cipher with a random key at least as long as the message
 - Provably unbreakable
 - Why? Look at ciphertext DXQR. Equally likely to correspond to plaintext DOIT (key AJIY) and to plaintext DONT (key AJDY) and any other 4 letters
 - Warning: keys *must* be random, or you can attack the cipher by trying to regenerate the key
 - Approximations, such as using pseudorandom number generators to generate keys, are *not* random

Enigma - Rotor Machines

- Substitution cipher
 - Each rotor is a substitution
 - Changes in rotor position change how substitutions are stacked
 - Key press passes through all rotors and back through a reflector rotor
 - Rotors advance after each key press changing the substitution.
- Key is initial position of the rotors
- More details
 - <http://www.codesandciphers.org.uk/enigma/>

Lessons from Enigma

- The importance of known plaintext (cribs)
- Mechanical assisted key breaking
 - Leading to modern computers
- Information in the pattern of traffic
 - Traffic analysis
- Humans in the loop are important
 - Information from spies
 - Poor user procedures
 - Birthday messages – many cribs
 - Repeated patterns
 - Reluctance to believe cipher has been broken

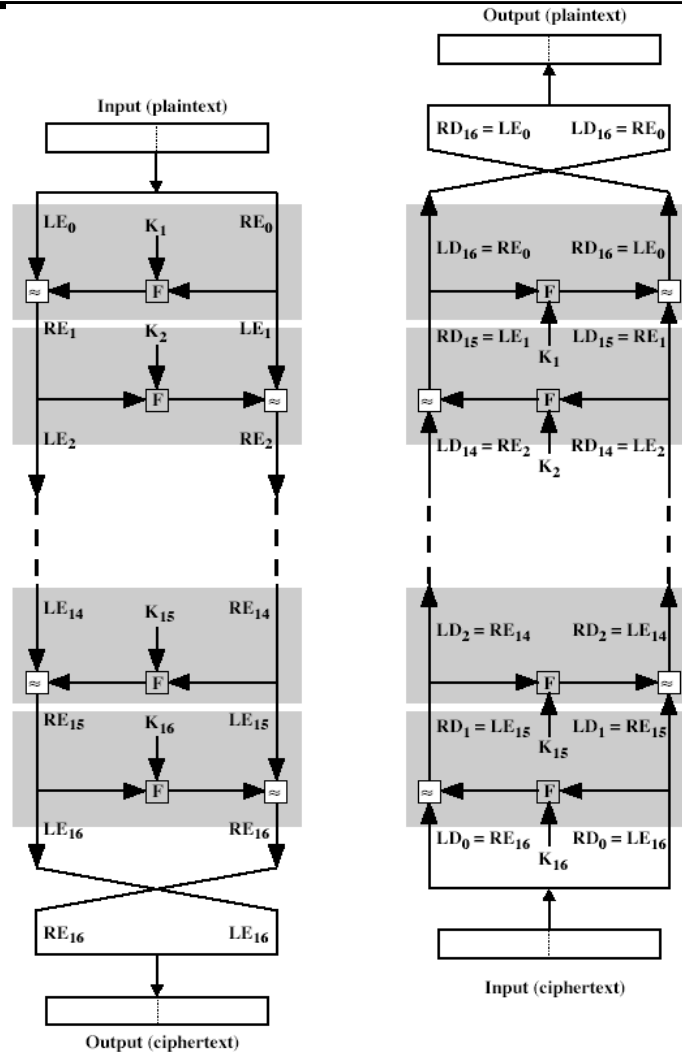
Overview of the DES

- A block cipher:
 - encrypts blocks of 64 bits using a 56 bit key
 - outputs 64 bits of ciphertext
- A product cipher
 - basic unit is the bit
 - performs both substitution (S-box) and transposition (permutation) (P-box) on the bits
- Cipher consists of 16 rounds (iterations) each with a round key generated from the user-supplied key

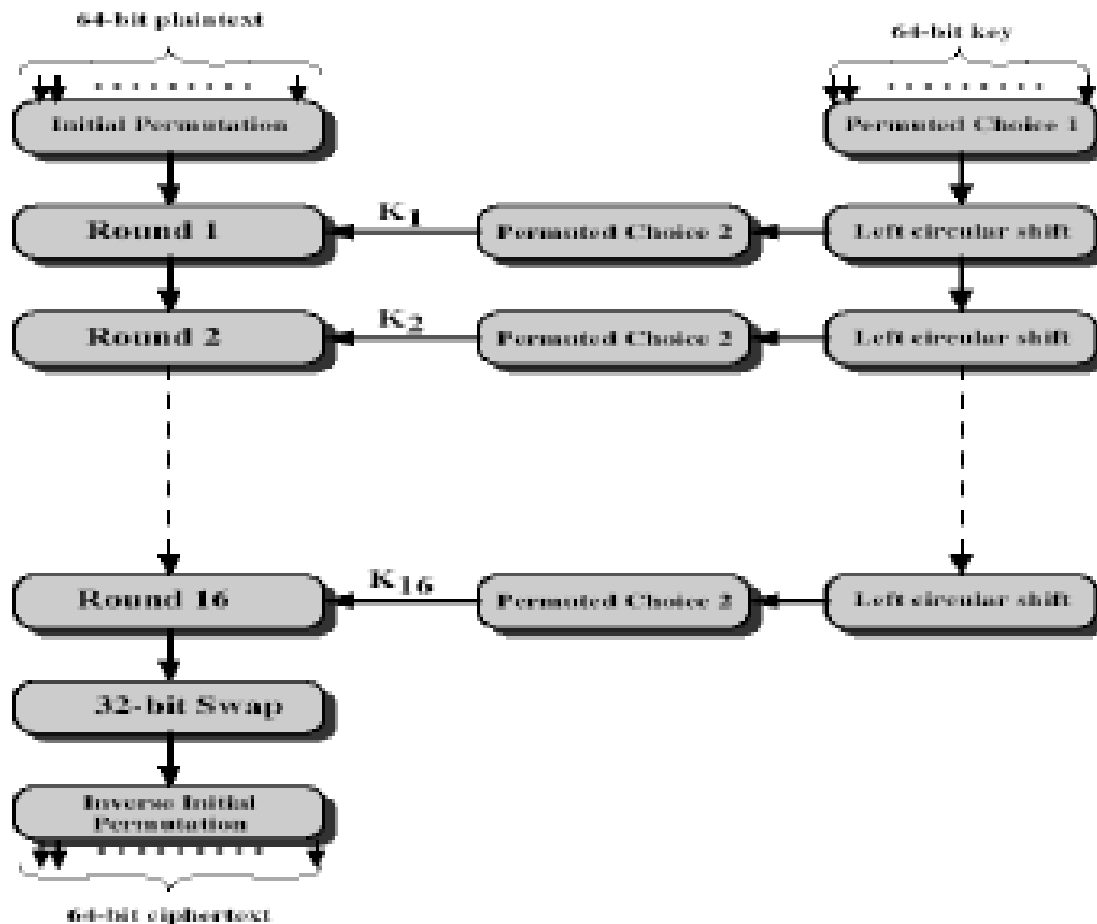
Feistel Network

- Structured to enable use of same S-box and P-box for encryption and decryption
 - Change only key schedule
- Major feature is key division and swapping
 - $L(i) = R(i-1)$
 - $R(i) = L(i-1) \text{ xor } f(K(i), R(i-1))$

Feistel Structure Decryption

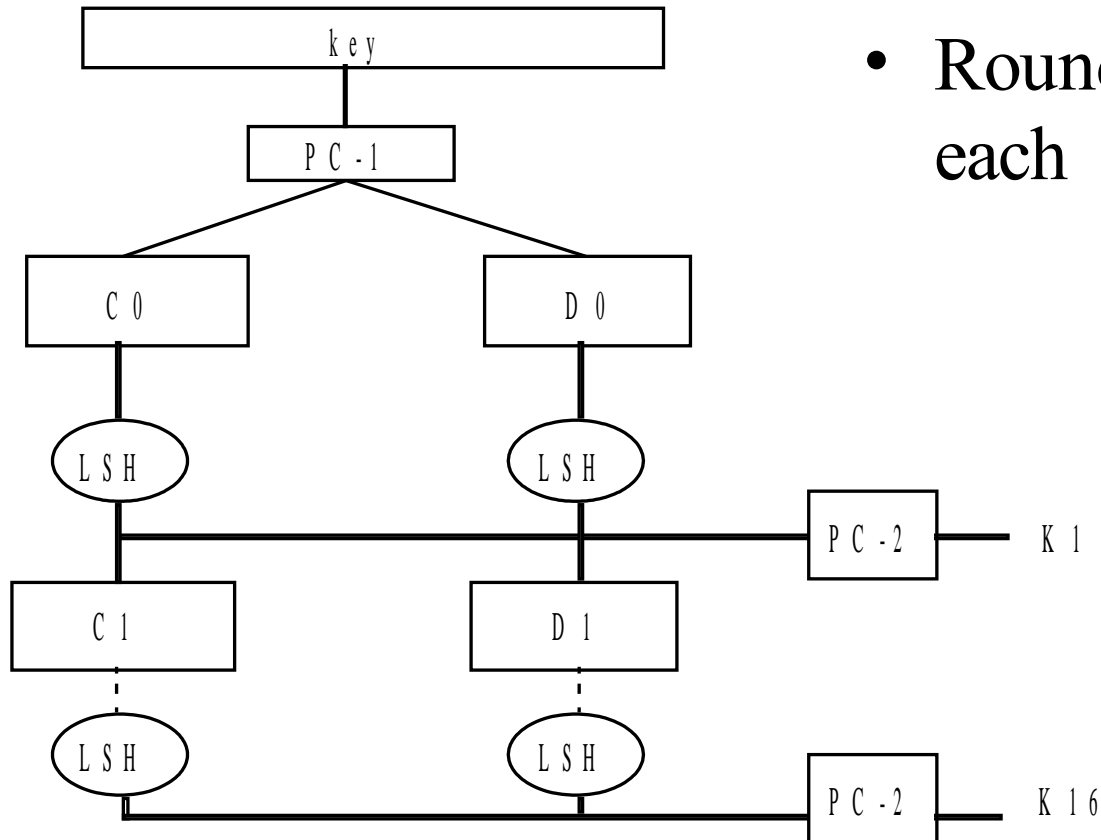


The Big Picture

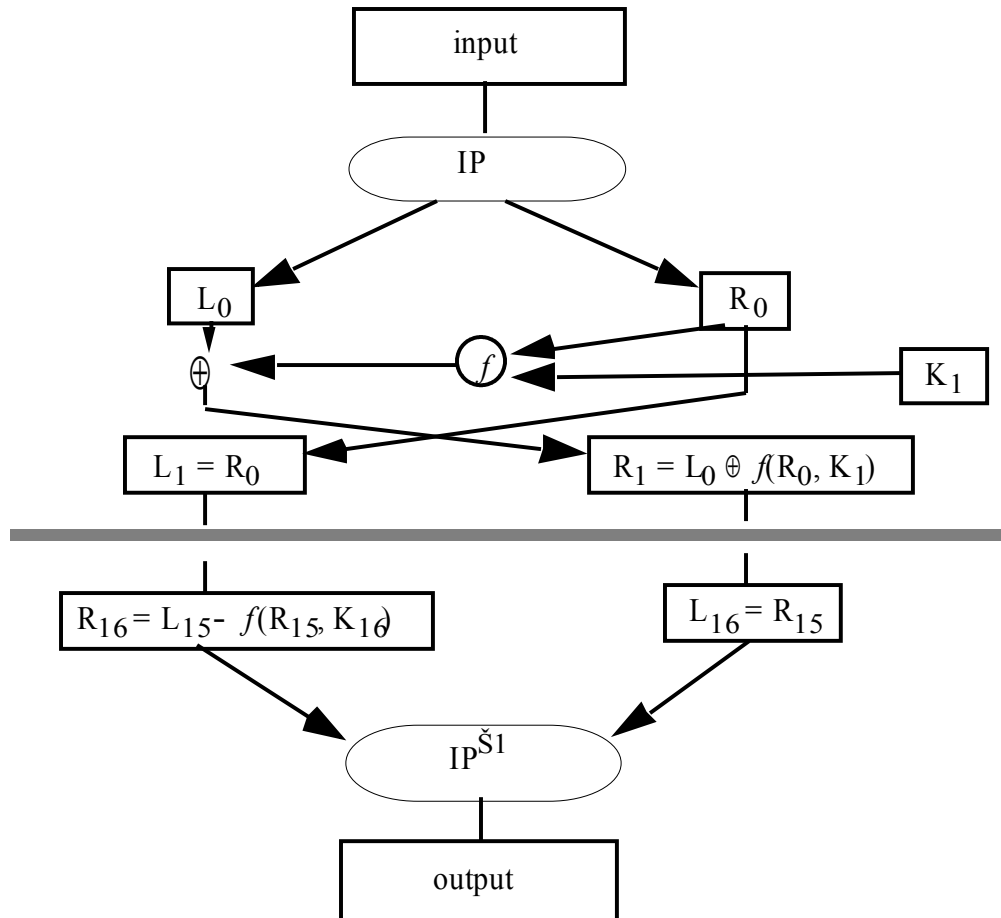


Generation of Round Keys

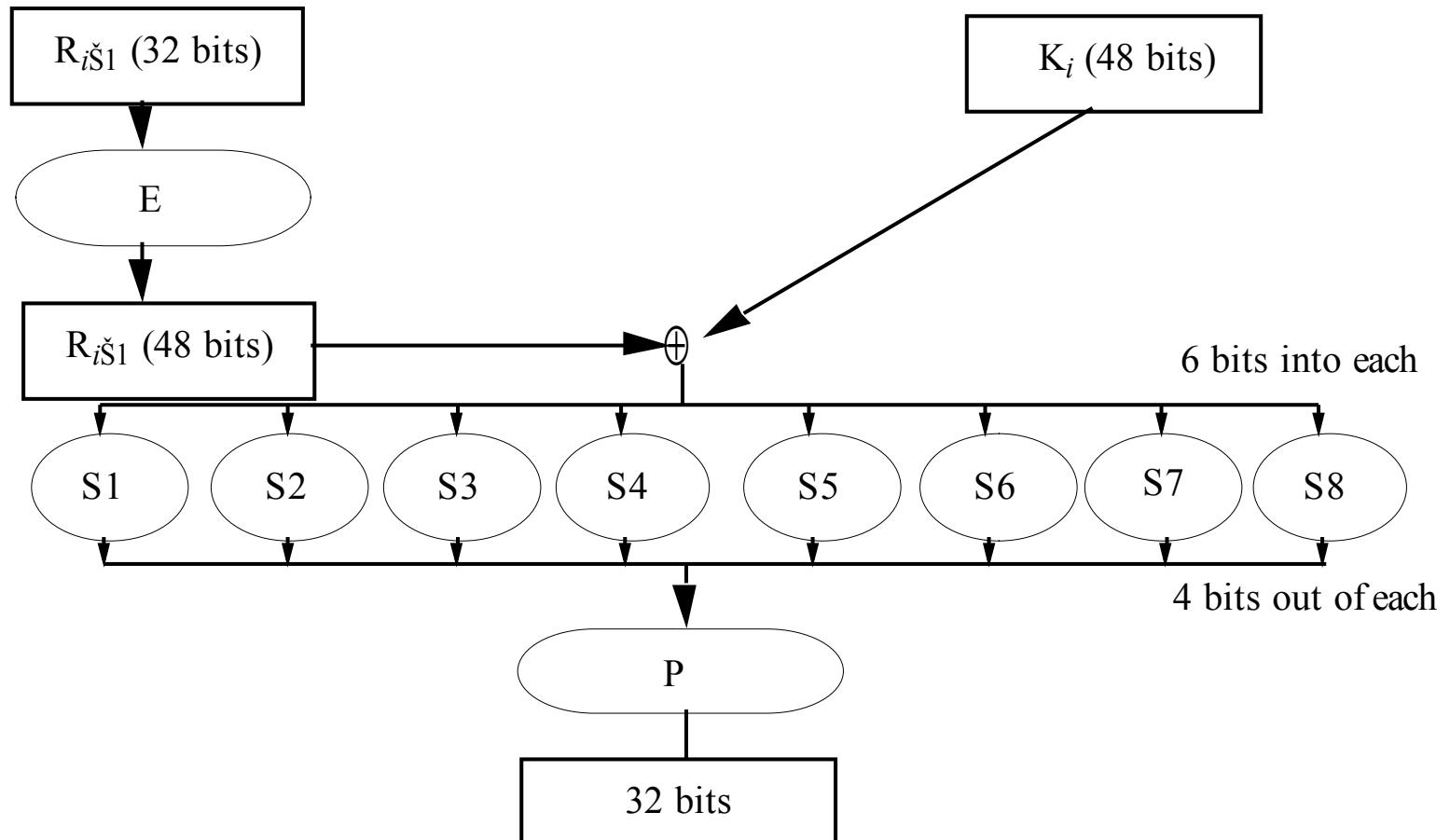
- Round keys are 48 bits each



Encryption



The f Function



Substitution boxes

- Key non-linear element to DES security
- 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 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 an encryption algorithm
- Where a change of **one** input or key bit results in changing approx **half of the** output bits
- If the change were small, this might provide a way to reduce the size of the key space to be searched
- DES exhibits strong avalanche

Controversy

- Considered too weak
 - Diffie, Hellman said in a few years technology would allow DES to be broken in days
 - Design using 1999 technology published
 - Design decisions not public
 - Some of the design decisions underlying the S-Boxes are unknown
 - S-boxes may have backdoors
 - Key size reduced from 112 bits in original Lucifer design to 56 bits

Undesirable Properties

- 4 weak keys
 - They are their own inverses
 - *i.e.* $\text{DES}_k(m) = c \Rightarrow \text{DES}_k(c) = m$
 - All 0's. All 1's. First half 1's second half 0's. Visa versa.
- 12 semi-weak keys
 - Each has another semi-weak key as inverse
 - *i.e.* $\text{DES}_{k1}(m) = c \Rightarrow \text{DES}_{k2}(c) = m$
- Possibly weak keys
 - Result in same subkeys being used in multiple rounds
- Complementation property
 - $\text{DES}_k(m) = c \Rightarrow \text{DES}_k(m \hat{=}) = c'$

Brute Force Attack

- What do you need?
- How many steps should it take?
- How can you do better?

Differential Cryptanalysis

- Was not reported in open literature until 1990
 - Tracks probabilities of differences inputs matching differences in outputs
- Chosen ciphertext attack
 - Requires 2^{47} plaintext, ciphertext pairs

Differential Cryptanalysis

- Build table of probabilities of inputs and outputs per round
 - $\Delta m_{i+1} = m_{i+1} \text{ xor } m'_{i+1}$
 - $\Delta m_{i+1} = [m_{i-1} \text{ xor } f(m_i, K_i)] \text{ xor } [m'_{i-1} \text{ xor } f(m'_i, K_i)]$
 - $\Delta m_{i+1} = \Delta m_{i-1} \text{ xor } [f(m_i, K_i) \text{ xor } f(m'_i, K_i)]$
- Compose probabilities per round

Differential Cryptanalysis

- Revealed several properties
 - Small changes in S-boxes reduces the number of pairs needed
 - The method was known to designer team as early as 1974
- Not so useful to break DES
 - But very useful to analyze the security of Feistel Network systems

Other Cryptoanalysis

- Linear Cryptoanalysis
 - Find linear approximations to describe transformations performed in DES
- Linear Key Cryptoanalysis
 - Use linear approximations to attack key system
 - Showed that the shift schedule actually increases security of DES
- More all the time

Current Status of DES

- A design for computer system and an associated software that could break any DES-enciphered message in a few days was published in 1998
- Several challenges to break DES messages solved using distributed computing
- National Institute of Standards and Technology (NIST) selected Rijndael as Advanced Encryption Standard (AES), successor to DES
 - Designed to withstand attacks that were successful on DES
 - It can use keys of varying length (128, 196, or 256)

Want to know more about DES?

- For a more detailed discussion on DES, see the slides for lecture 8 of a previous year's Information Assurance course at:
 - <http://www.cs.uiuc.edu/class/fa05/cs498sh/slides/lecture8>
- Bruce Schneier, *Applied Cryptography*.
- William Stallings, *Cryptography and Network Security*, Second Edition, Prentice Hall, 1998.

Advance Encryption Standard

- Go to AES slides.
 - <http://www.cs.uiuc.edu/class/fa05/cs498sh/slides/>
- NIST standard writeup
 - <http://csrc.nist.gov/publications/fips/fips197/fips-1>

Stream, Block Ciphers

- E encipherment function
 - $E_k(b)$ encipherment of message b with key k
 - In what follows, $m = b_1b_2 \dots$, each b_i of fixed length
- Block cipher
 - $E_k(m) = E_k(b_1)E_k(b_2) \dots$
- Stream cipher
 - $k = k_1k_2 \dots$
 - $E_k(m) = E_{k_1}(b_1)E_{k_2}(b_2) \dots$
 - If $k_1k_2 \dots$ repeats itself, cipher is *periodic* and the length of its period is one cycle of $k_1k_2 \dots$

Examples

- Vigenère cipher
 - $b_i = 1$ character, $k = k_1k_2 \dots$ where $k_i = 1$ character
 - Each b_i enciphered using $k_{i \bmod \text{length}(k)}$
 - Stream cipher
- DES
 - $b_i = 64$ bits, $k = 56$ bits
 - Each b_i enciphered separately using k
 - Block cipher

Stream Ciphers

- Often (try to) implement one-time pad by xor'ing each bit of key with one bit of message

– Example:

$$m = 00101$$

$$k = 10010$$

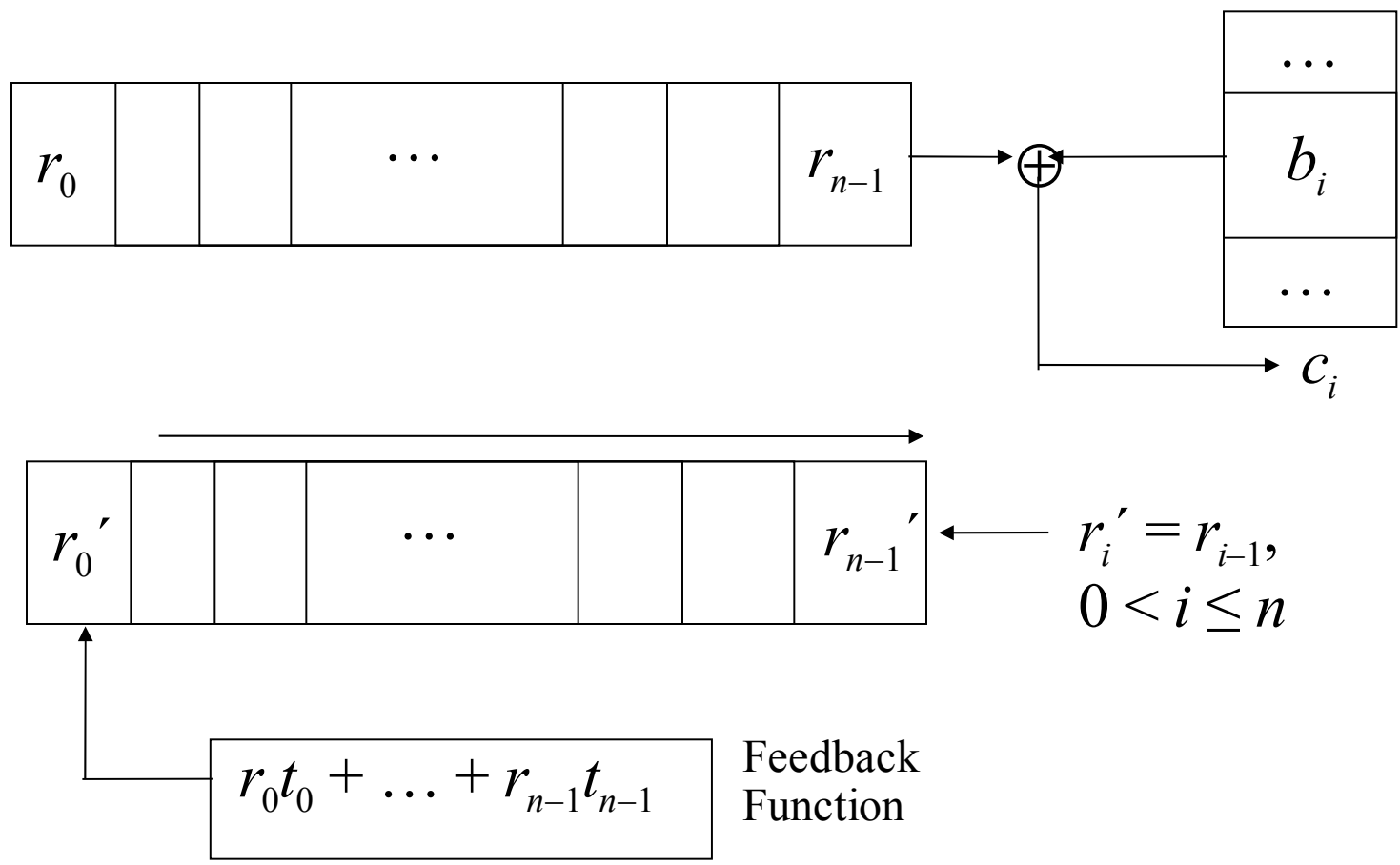
$$c = 10111$$

- But how to generate a good key?

Synchronous Stream Ciphers

- n -stage Linear Feedback Shift Register:
consists of
 - n bit register $r = r_0 \dots r_{n-1}$
 - n bit tap sequence $t = t_0 \dots t_{n-1}$
 - Use:
 - Use r_{n-1} as key bit
 - Compute $x = r_0 t_0 \oplus \dots \oplus r_{n-1} t_{n-1}$
 - Shift r one bit to right, dropping r_{n-1} , x becomes r_0

Operation



Example

- 4-stage LFSR; $t = 1001$

r	k_i	$new\ bit\ computation$	$new\ r$
0010	0	$01 \oplus 00 \oplus 10 \oplus 01 = 0$	0001
0001	1	$01 \oplus 00 \oplus 00 \oplus 11 = 1$	1000
1000	0	$11 \oplus 00 \oplus 00 \oplus 01 = 1$	1100
1100	0	$11 \oplus 10 \oplus 00 \oplus 01 = 1$	1110
1110	0	$11 \oplus 10 \oplus 10 \oplus 01 = 1$	1111
1111	1	$11 \oplus 10 \oplus 10 \oplus 11 = 0$	0111
8	00	$11 \oplus 10 \oplus 10 \oplus 11 = 1$	1011

- Key sequence has period of 15 (010001111010110)

LFSR Period

- For n bit register
 - Maximum possible period is $2^n - 1$
 - -1 because 0's will only yield 0's
- Not all tap sequences will yield this period
 - Large theory on computing maximal period feedback functions

NLFSR

- n-stage Non-Linear Feedback Shift Register:
consists of
 - n bit register $r = r_0 \dots r_{n-1}$
 - Use:
 - Use r_{n-1} as key bit
 - Compute $x = f(r_0, \dots, r_{n-1})$; f is any function
 - Shift r one bit to right, dropping r_{n-1} , x becomes r_0

Note same operation as LFSR but more general
bit replacement function

Example

- 4-stage NLFSR; $f(r_0, r_1, r_2, r_3) = (r_0 \& r_2) | r_3$

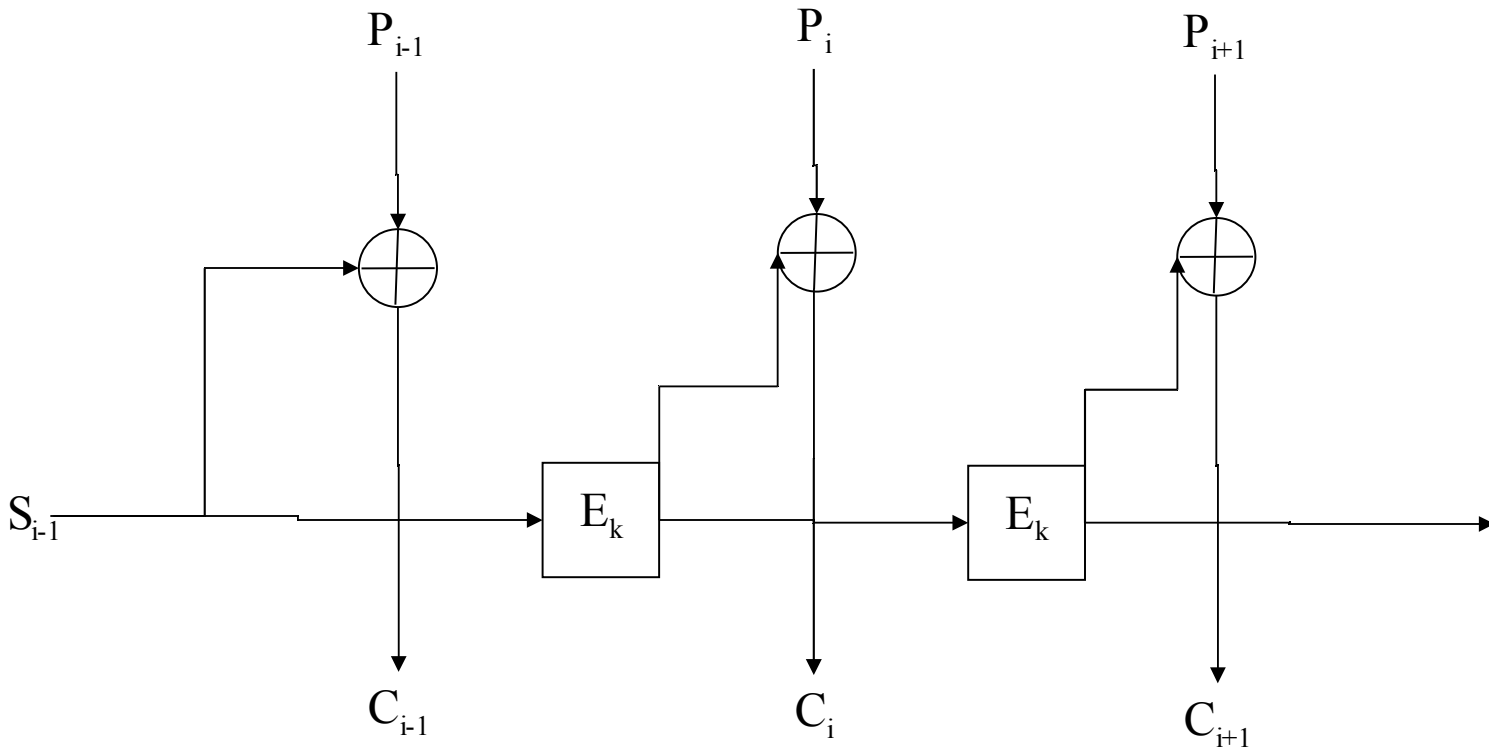
r	k_i	<i>new bit computation</i>	<i>new r</i>
1100	0	$(1 \& 0) 0 = 0$	0110
0110	0	$(0 \& 1) 0 = 0$	0011
0011	1	$(0 \& 1) 1 = 1$	1001
1001	1	$(1 \& 0) 1 = 1$	1100
1100	0	$(1 \& 0) 0 = 0$	0110
0110	0	$(0 \& 1) 0 = 0$	0011
0011	1	$(0 \& 1) 1 = 1$	1001

– Key sequence has period of 4 (0011)

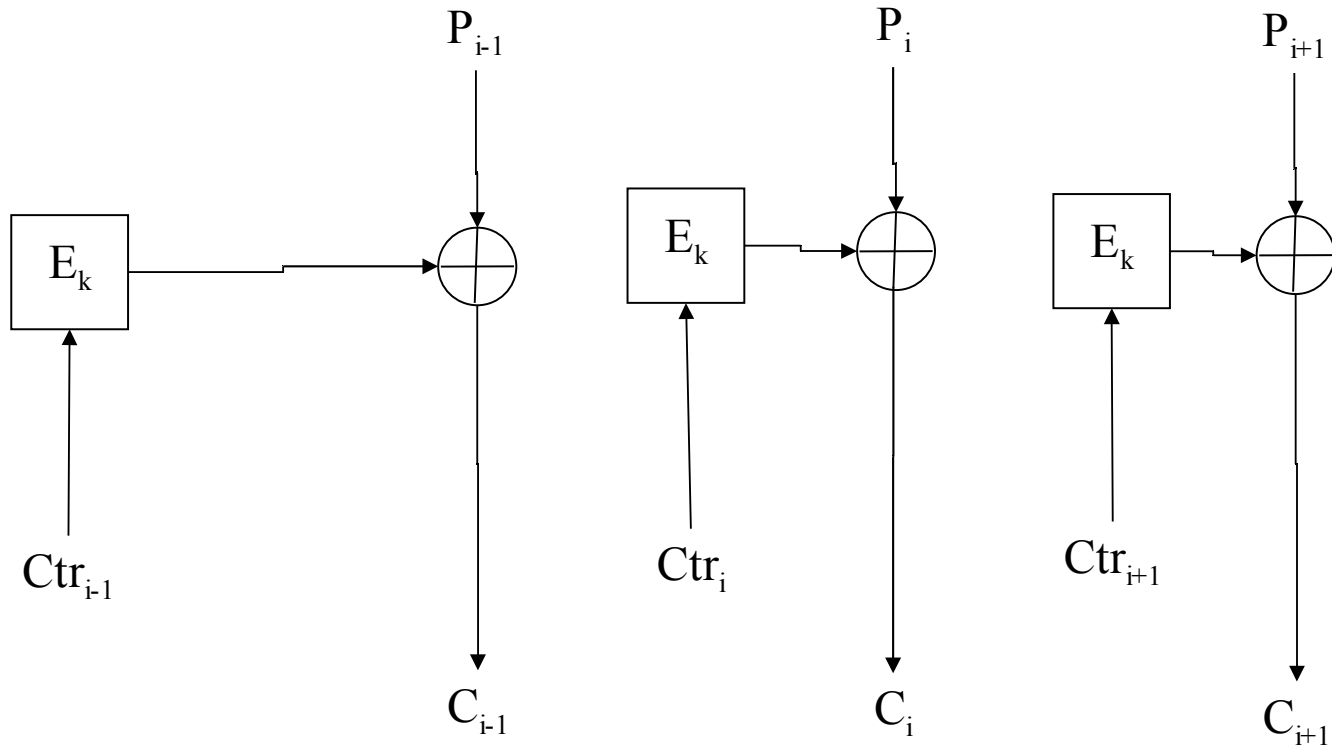
Eliminating Linearity

- NLFSRs not common
 - No body of theory about how to design them to have long period
- Alternate approach: *output feedback mode*
 - For E encipherment function, k key, r register:
 - Compute $r' = E_k(r)$; key bit is rightmost bit of r'
 - Set r to r' and iterate, repeatedly enciphering register and extracting key bits, until message enciphered
 - Variant: use a counter that is incremented for each encipherment rather than a register
 - Take rightmost bit of $E_k(i)$, where i is number of encipherment

OFB Mode



Counter Mode



Issues with OFB/Counter

- Additional standard modes for DES/AES
- Losing Synchronicity is fatal
 - All later decryptions will be garbled
- OFB needs an initialization vector
- Counter mode lets you generate a bit in the middle of the stream
- RC4 is a well-known stream cipher that uses OFB. Used in WEP

Self-Synchronous Stream Cipher

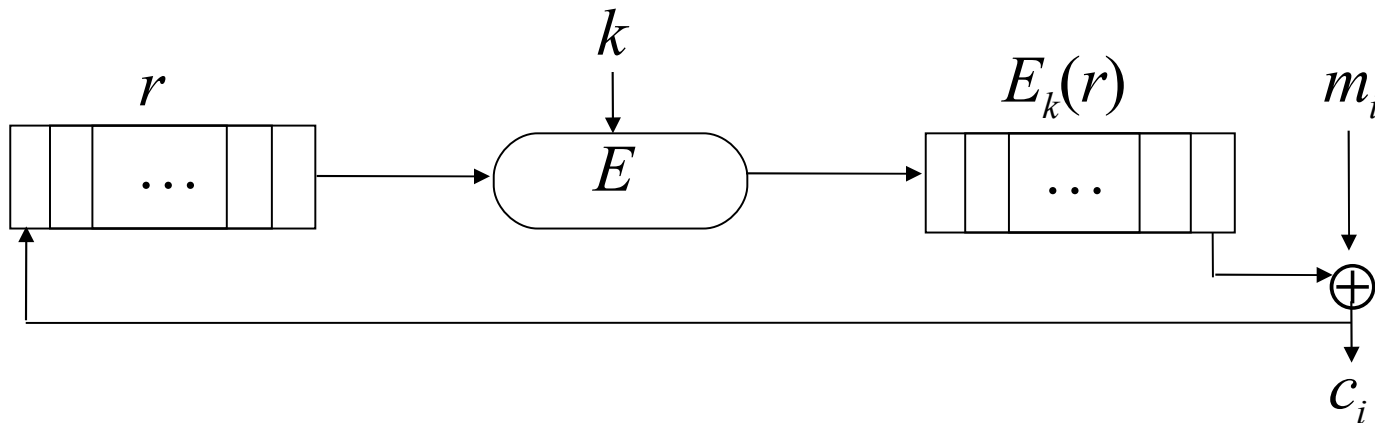
- Take key from message itself (*autokey*)
- Example: Vigenère, key drawn from plaintext
 - *key* XTHEBOYHASTHEBA
 - *plaintext* THEBOYHASTHEBAG
 - *ciphertext* QALFPNFHSLALFCT
- Problem:
 - Statistical regularities in plaintext show in key
 - Once you get any part of the message, you can decipher more

Another Example

- Take key from ciphertext (*autokey*)
- Example: Vigenère, key drawn from ciphertext
 - *key* XQXBCQOVVNGNRTT
 - *plaintext* THEBOYHASTHEBAG
 - *ciphertext* QXBCQOVVNGNRTTM
- Problem:
 - Attacker gets key along with ciphertext, so deciphering is trivial

Variant

- Cipher feedback mode: 1 bit of ciphertext fed into n bit register
 - Self-healing property: if ciphertext bit received incorrectly, it and next n bits decipher incorrectly; but after that, the ciphertext bits decipher correctly
 - Need to know k, E to decipher ciphertext



Block Ciphers

- Encipher, decipher multiple bits at once
- Each block enciphered independently
 - Electronic Code Book Mode (ECB)

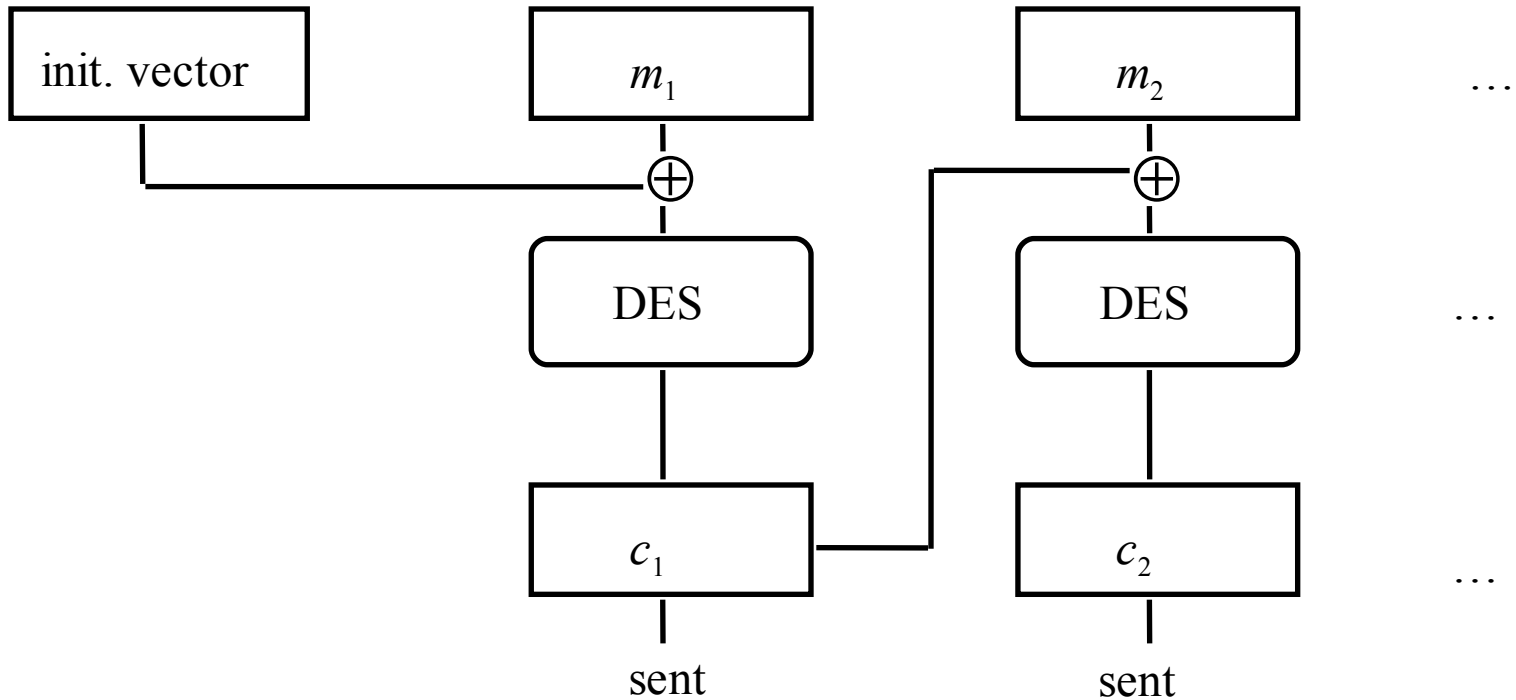
ECB Problem

- Problem: identical plaintext blocks produce identical ciphertext blocks
 - Example: two database records
 - MEMBER: HOLLY INCOME \$100,000
 - MEMBER: HEIDI INCOME \$100,000
 - Encipherment:
 - ABCQZRME GHQMRSIB CTXUVYSS RMGRPFQN
 - ABCQZRME ORMPABRZ CTXUVYSS RMGRPFQN

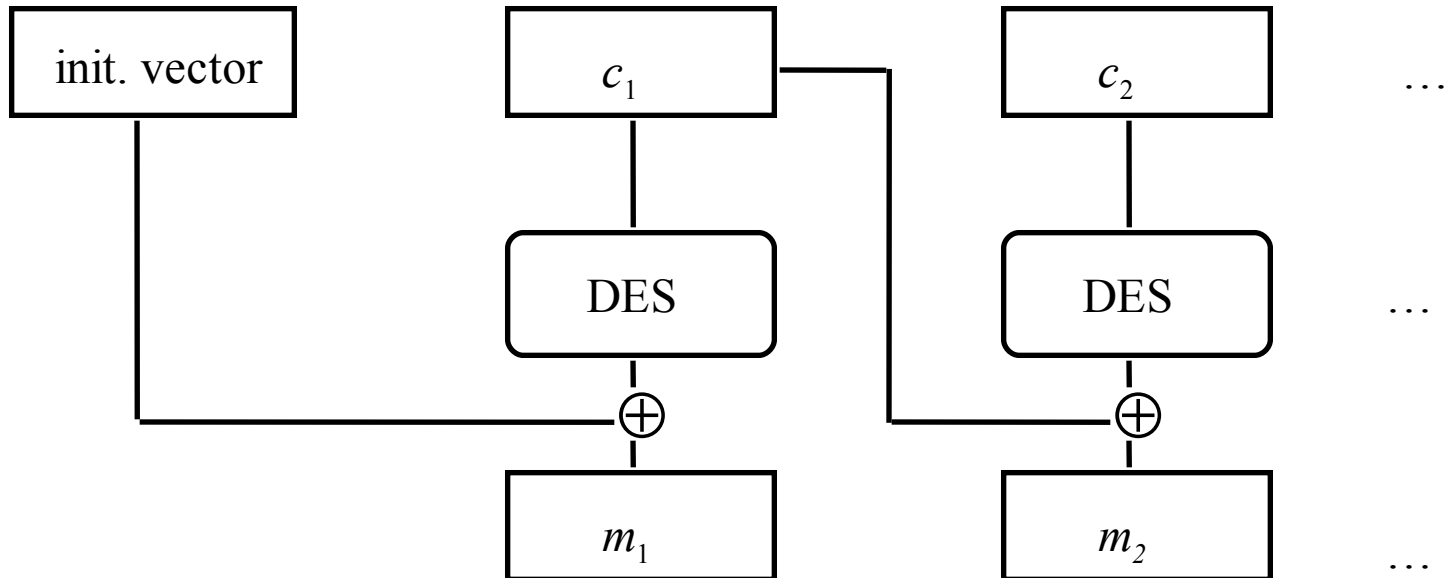
Solutions

- Insert information about block's position into the plaintext block, then encipher
 - *Cipher block chaining (CBC)*:
 - Exclusive-or current plaintext block with previous ciphertext block:
 - $c_0 = E_k(m_0 \oplus I)$
 - $c_i = E_k(m_i \oplus c_{i-1})$ for $i > 0$
- where I is the initialization vector

CBC Mode Encryption



CBC Mode Decryption



Self-Healing Property

- If one block of ciphertext is altered, the error propagates for at most two blocks
- Initial message
 - 3231343336353837 3231343336353837 3231343336353837
3231343336353837
- Received as (underlined 4c should be 4b)
 - ef7c4cb2b4ce6f3b f6266e3a97af0e2c 746ab9a6308f4256
33e60b451b09603d
- Which decrypts to
 - efca61e19f4836f1 3231333336353837 3231343336353837
3231343336353837
 - Incorrect bytes underlined
 - Plaintext “heals” after 2 blocks

Multiple Encryptions

- Double encryption not generally used
 - Meet-in-the-middle attack
 - $C = E_{k_2}(E_{k_1}(P))$
 - Modifies brute force to require only 2^{n+1} steps instead of 2^{2n}
- Encrypt-Decrypt-Encrypt Mode (2 or 3 keys: k, k')
 - $c = DES_k(DES_{k'}^{-1}(DES_{k'}(m)))$
 - Also called Triple DES or 3DES when used with 3 keys
 - 168 bits of key, but effective key length of 112 due to meet-in-the middle
 - Not yet practical to break but AES much faster
- Encrypt-Encrypt-Encrypt Mode (3 keys: k, k', k'')
 - $c = DES_k(DES_{k'}(DES_{k''}(m)))$

Key Points

- Historical Ciphers
 - Give examples of linguistic attacks
 - Substitution and transposition ciphers
- Symmetric key ciphers
 - AES and DES
 - Today's workhorse algorithms
 - Crypto analysis attacks on algorithms
 - Product ciphers