Fundamentals of Cryptography

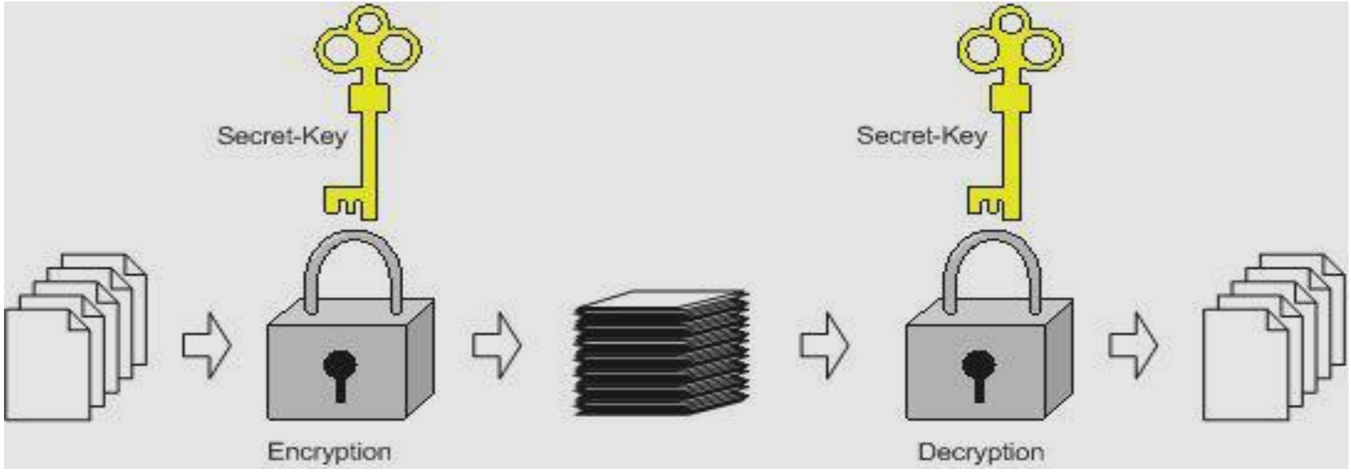
Ijaz Ahmad Umarzai

ijazumarzai@mcs.edu.pk

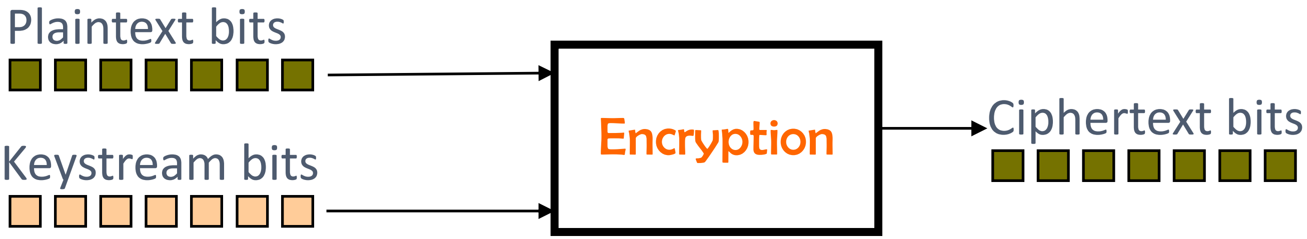
Secret Key Cryptography

Secret Key Cryptography

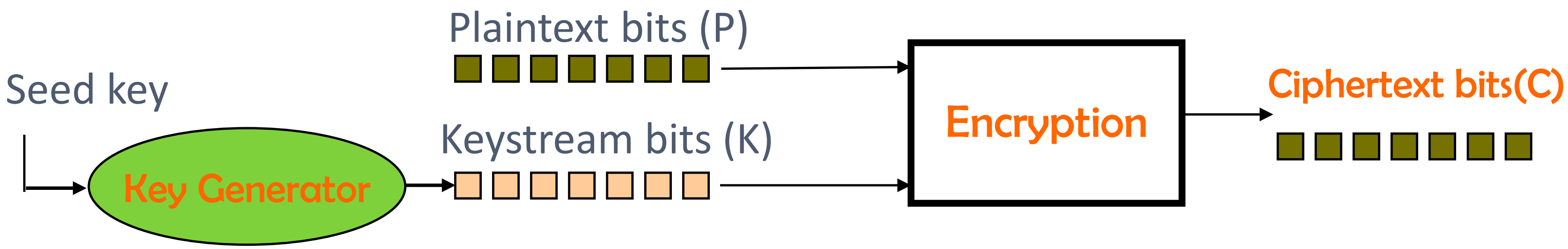
- Both encryption and decryption keys are the same and are kept secret
- The secret key must be known at both ends to perform encryption or decryption (Fig)
- Secret Key algorithms are fast and they are used for encrypting/decrypting high volume data
- Secret key cryptography is classified into two types
  - Block Ciphers
  - Stream Ciphers



Stream Ciphers

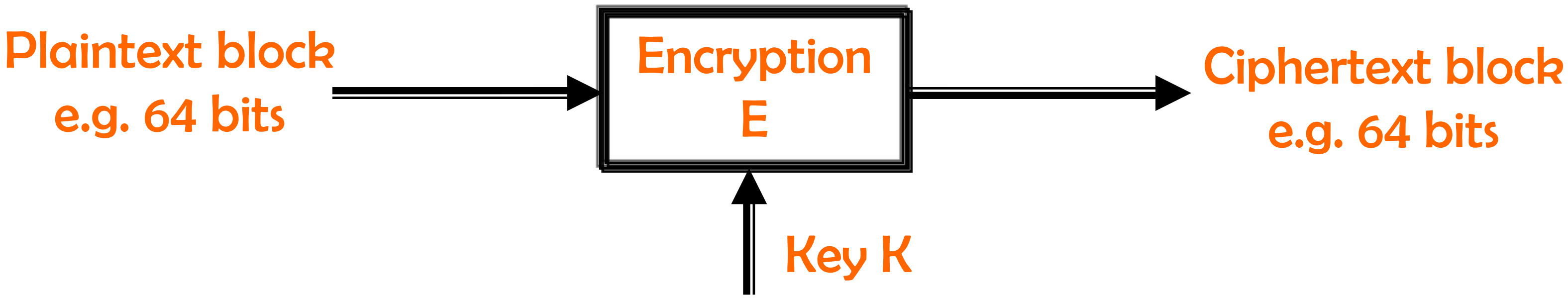


- A stream cipher is a type of symmetric encryption in which input data is encrypted one bit (sometime one byte) at a time
- Examples of stream ciphers include SEAL, TWOPRIME, RC4, A5



- To encrypt plaintext stream
  - A random set of bits is generated from a seed key, called keystream which is as long as the message
  - Keystream bits are added modulo 2 to plaintext to form the ciphertext stream
- To decrypt ciphertext stream
  - use the same seed key to generate the same keystream used in encryption
  - Add the keystream modulo 2 to the ciphertext to retrieve the plaintext
  - i.e. $C = P \oplus K \Rightarrow C \oplus K = (P \oplus K) \oplus K = P$

Block Cipher



- A block cipher is a type of symmetric encryption which operates on blocks of data. Modern block ciphers typically use a block length of 128 bits or more
- Examples of block ciphers include DES, AES, RC6, and IDEA
- A block cipher breaks message into fixed sized blocks
- Takes one block (plaintext) at a time and transform it into another block of the same length using a user provided secret key
- Decryption is performed by applying the reverse transformation to the ciphertext block using the same secret key
- Most symmetric block ciphers are based on a Feistel Cipher Structure (Explained in next slides)

Properties of Good Ciphers

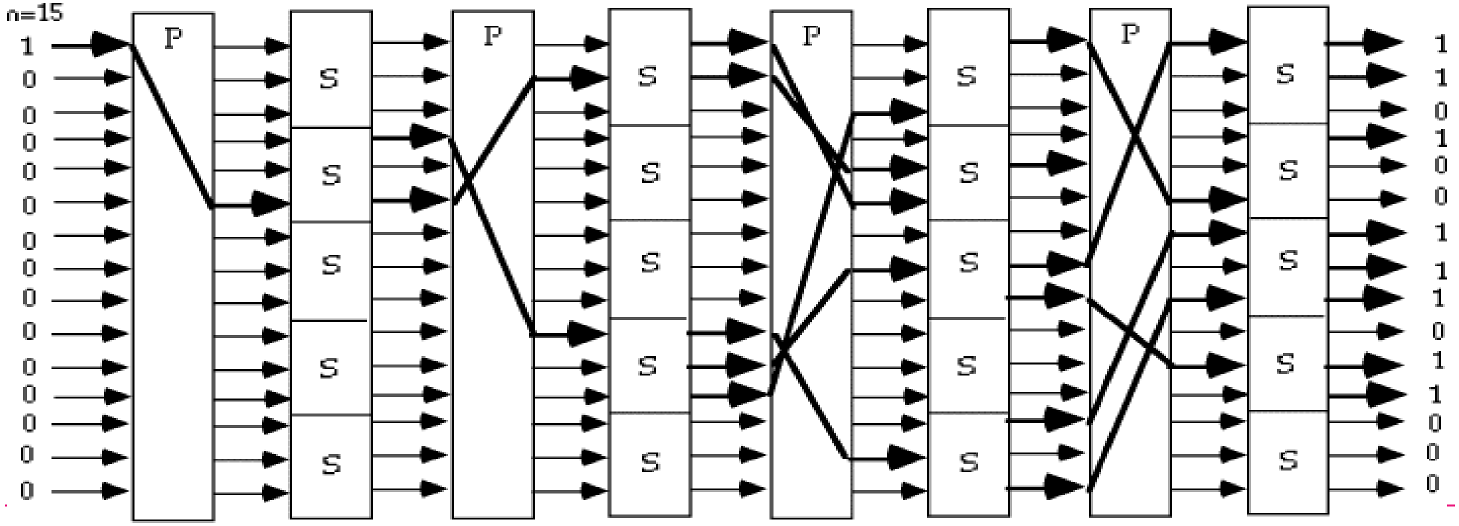
Confusion and Diffusion

Confusion and Diffusion

- In cryptography, confusion and diffusion are two properties of the operation of a secure cipher which were identified by Shannon in his paper, "Communication Theory of Secrecy Systems" published in 1949
- Confusion refers to making the relationship between the key and the ciphertext as complex and involved as possible
  - Substitution is one of the mechanism for primarily confusion
- Diffusion refers to the property that redundancy in the statistics of the plaintext is "dissipated" in the statistics of the ciphertext
  - Transposition (Permutation) is a technique for diffusion
  - Associate dependency of bits of the output to the bits of input
  - In a cipher with good diffusion, flipping an input bit should change each output bit with a probability of one half

Motivation for Feistel Cipher Structure

- In 1949, Claude Shannon also introduced the idea of substitution-permutation (S-P) networks which form the basis of modern block ciphers
- S-P networks are based on the two primitive cryptographic operations: substitution (S-box) & permutation (P-box)
- provide confusion and diffusion of message



Motivation for Feistel Cipher Structure

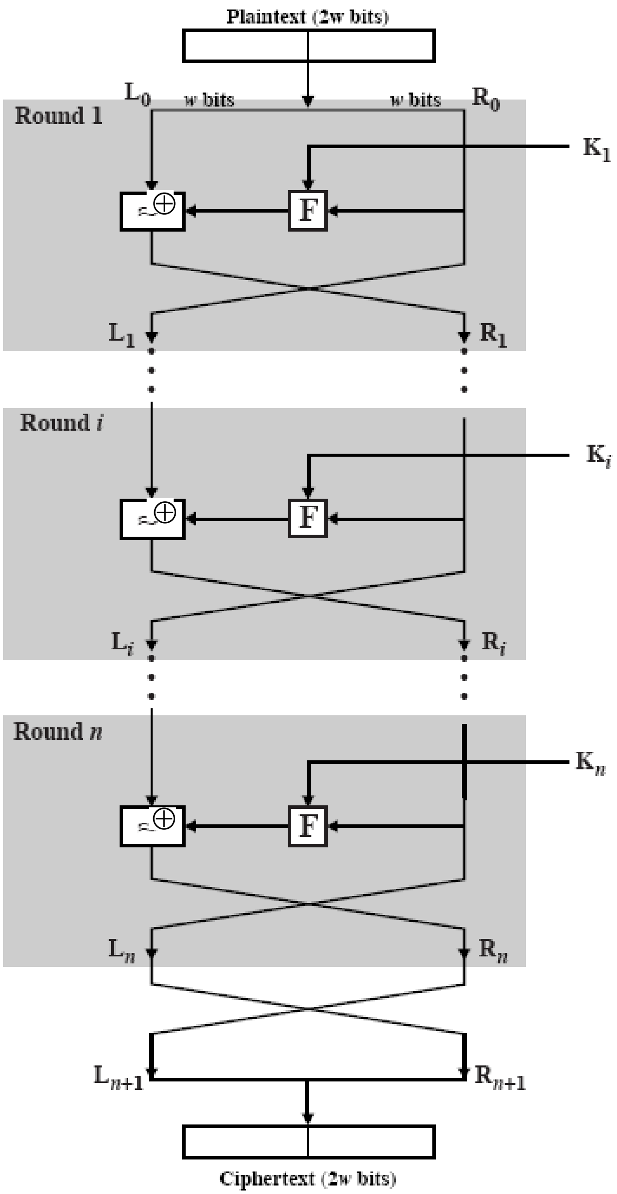
- S-P network: a special form of substitution-transposition product cipher
- Product cipher

  Two or more simple ciphers are performed in sequence in such a way that the final result or product is cryptographically stronger than any of the component ciphers

- Feistel cipher

  In 1970's, Horst Feistel (IBM T.J. Watson Research Labs) invented a suitable (practical) structure which adapted Shannon's S-P network

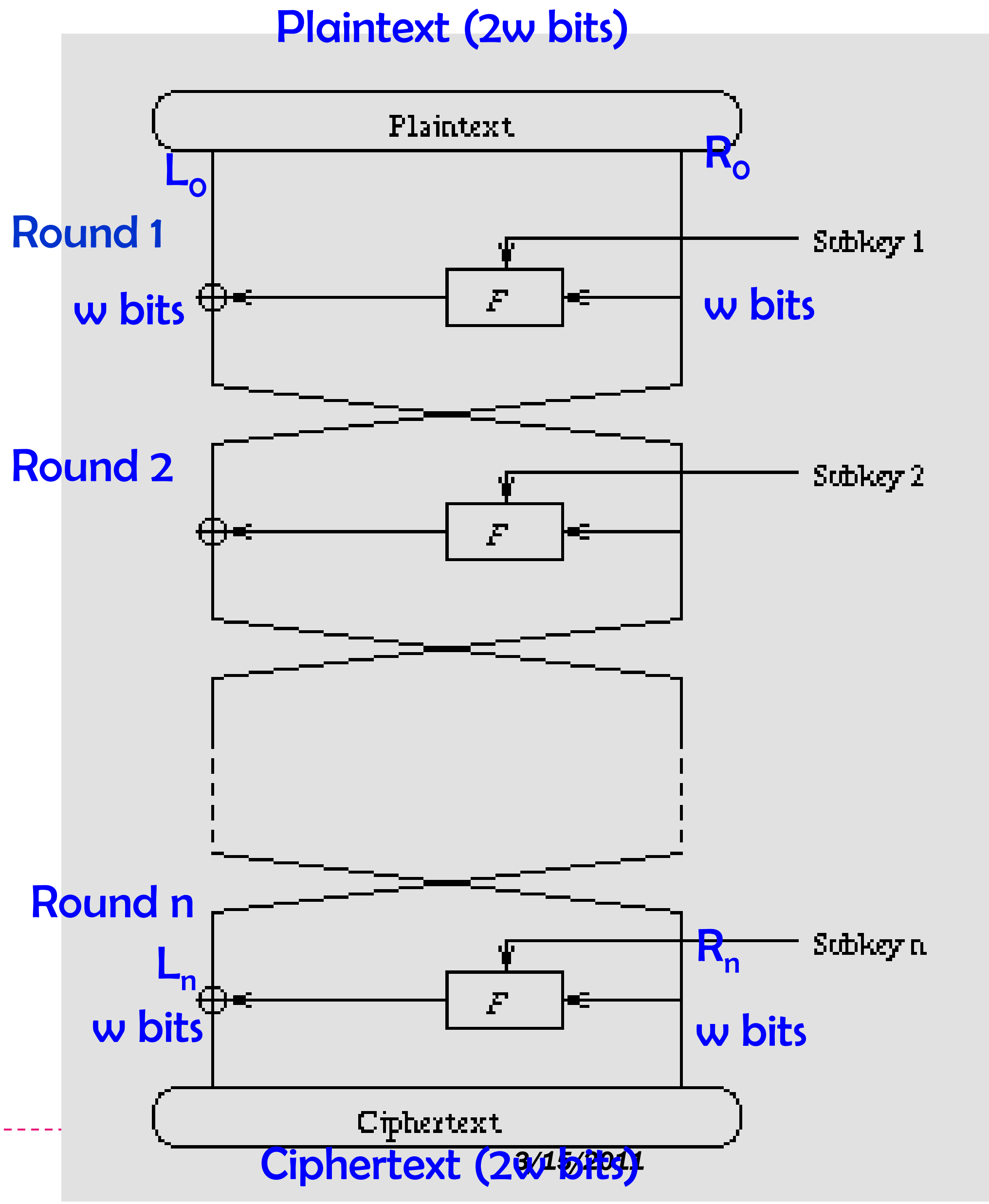
  Encryption and decryption use the same structure



3/15/2011

Feistel Cipher Structure

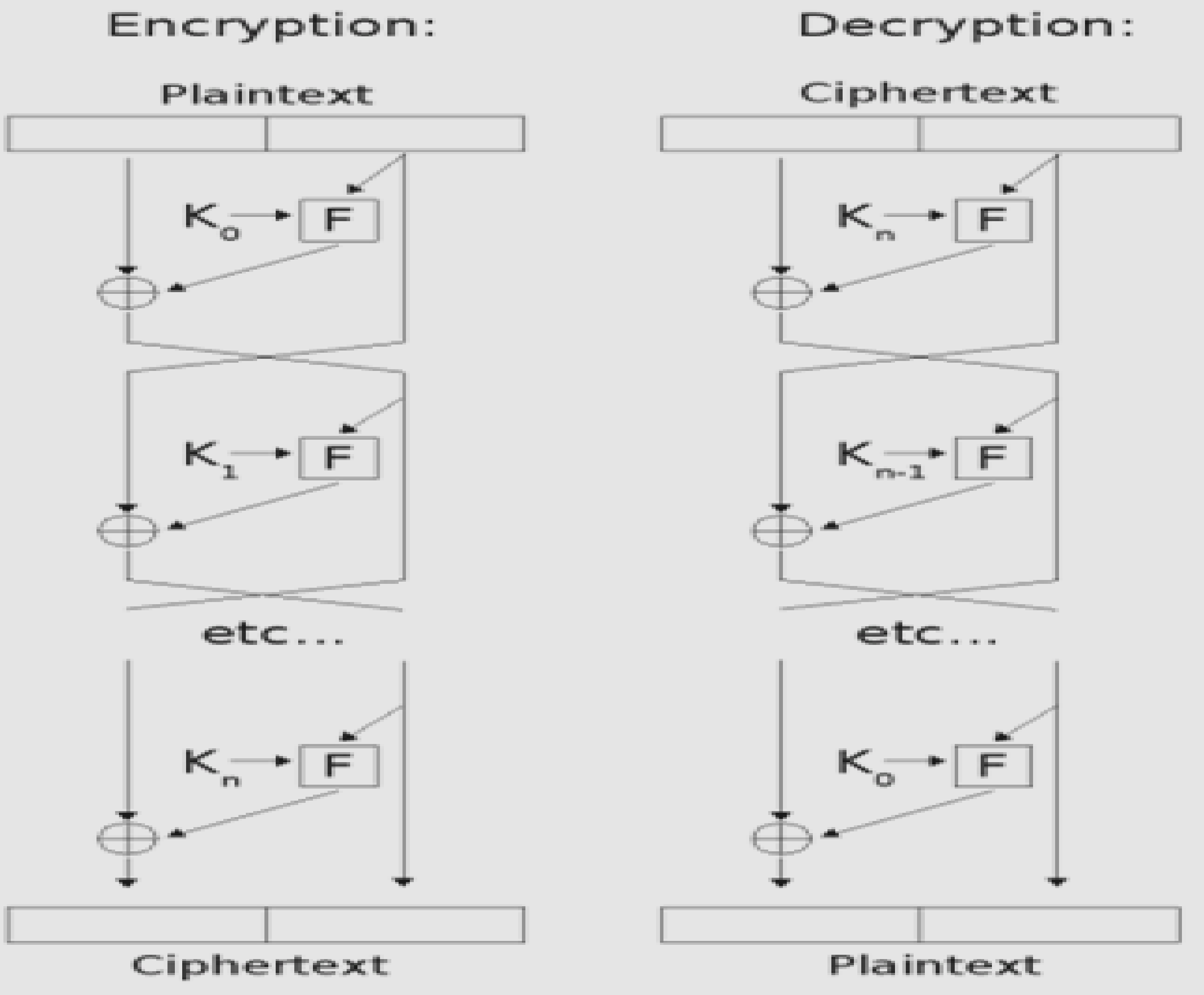
- Ideas for each round:
  - Partition input block into two halves
  - Process through multiple rounds
  - In each round:
    - Perform a substitution on left data half based on a round function of right half & subkey
    - Then have permutation swapping halves



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

- Overview
- Encryption
- Decryption
- Security

DES – Data Encryption Standard

- A Block cipher

- Data encrypted in 64-bit blocks using a 56-bit key (effective key); Ciphertext is of 64-bit long

- Encrypts by series of substitution and transpositions (or permutations)

DES History

- The first commercially available Feistel Cipher was developed by IBM in the 1960's; called Lucifer (by Feistel and Coppersmith)
- US National Bureau of Standards (NBS) issued a call for proposals in 1972
- Lucifer was refined, renamed the Data Encryption Algorithm (DEA) in 1974
- Adopted as the standard by NBS in 1976
- DES is the first official U.S. government cipher intended for commercial use
- Replacement standard (AES) is in effect May 26, 2002
  - http://csrc.nist.gov/CryptoToolkit/aes/frn-fips197.pdf

DES Design Controversy

- There has been 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, especially in financial applications
- Best known and widely used symmetric algorithm in the world
- But, no longer is considered secure for highly sensitive applications.

Input of DES

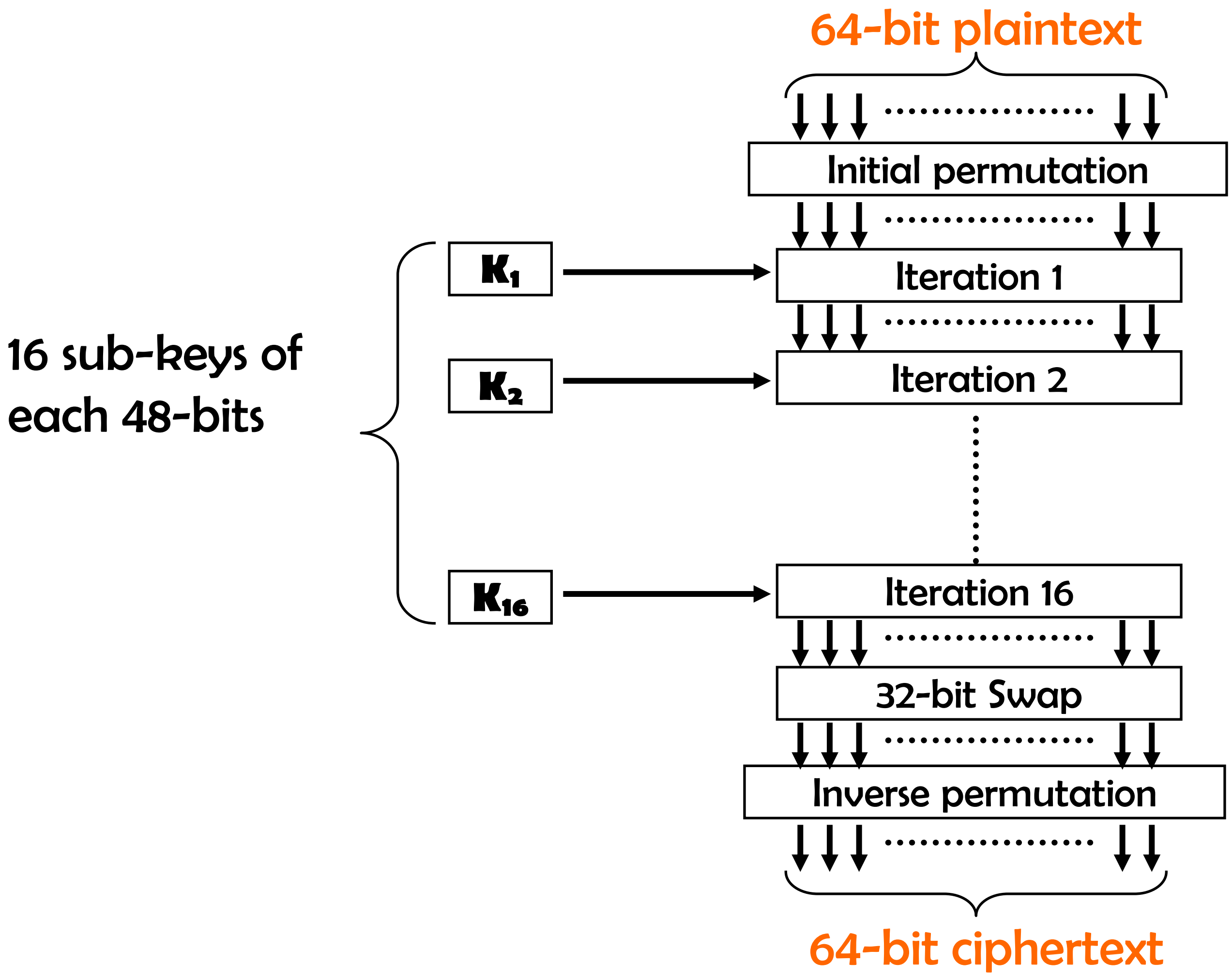
- Data: need to be broken into 64-bit blocks; add pad at the last message if necessary.
  - e.g. X =(3 5 0 7 7 F 1 0 A B 1 2 F C 6 5)HEX

- Secret key:
  - Any string of 64 bits long including 8 parity bits.
  - 1 parity bit in each 8-bit byte of the key may be utilized for error detection in key generation, distribution, and storage
  - K=(k1…k7k8… k15k16 k17…k24…k32… k40… k48… k56… k64)
  - The bits k8, k16, k24, k32, k40, k48, k56, k64 can be used for parity check

DATA ENCRYPTION STANDARD

- Overview
- Encryption
- Decryption
- Security

DES Encryption Diagram



Description

- DES operates on 64-bit blocks of plaintext. After an initial permutation the block is broken into right half and left half, each being 32 bits long
- There are 16 rounds of identical operations, call function f, in which data are combined with 16 keys of 48 bits, one for each round
- After the 16th round the right and left halves are joined, and a final permutation (the inverse of the initial permutation) finishes the algorithm
- Because DES's operation is very repetitive, it is readily implementable in hardware, as well as software

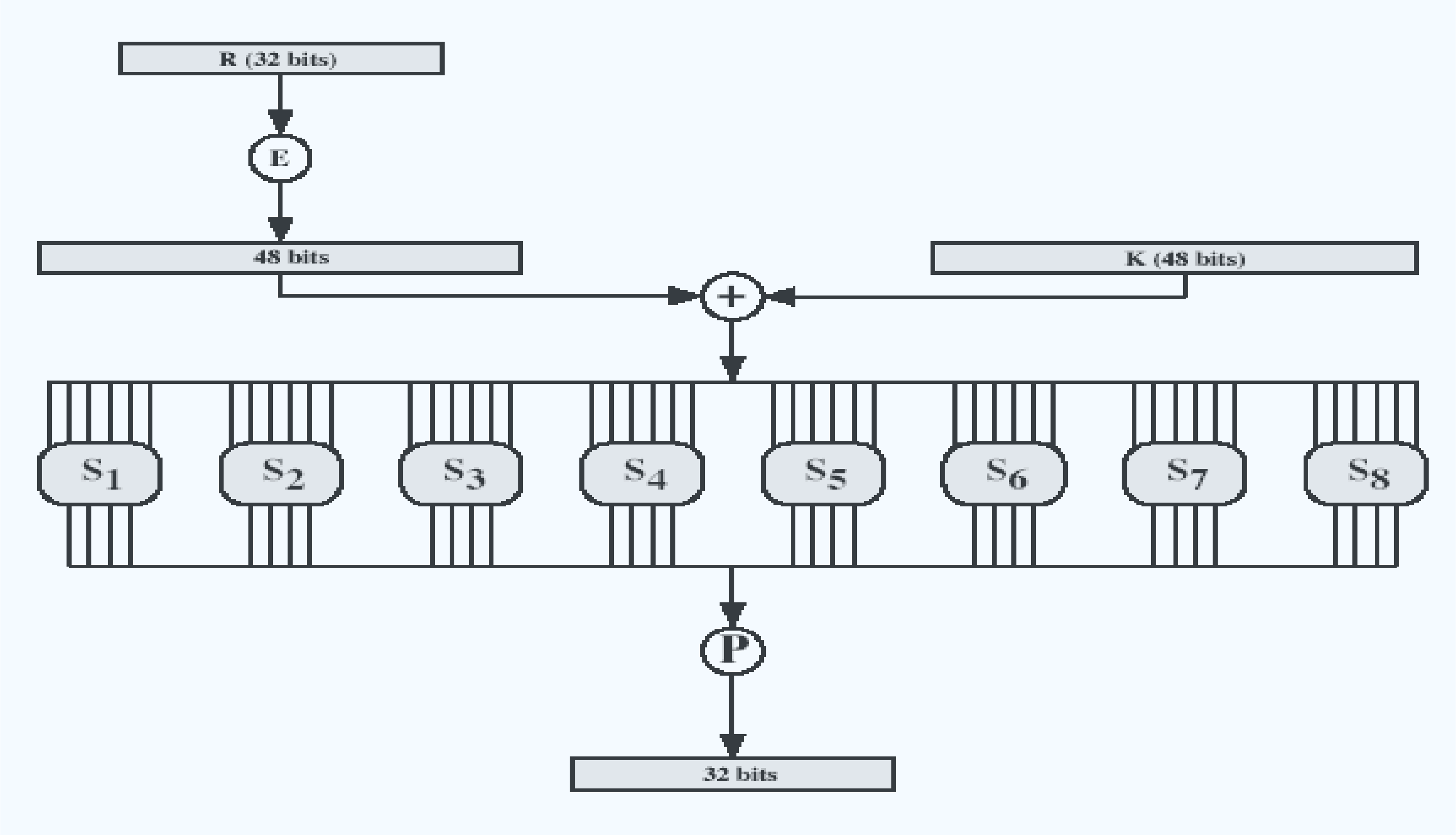
DES Round Structure

- Uses two 32-bit L & R halves

- As for any Feistel cipher can describe as:
  - Li = (Ri–1)
  - Ri = (Li–1) xor F(Ri–1, Ki)

- Takes 32-bit R half and 48-bit sub-key and:
  - expands R to 48-bits using perm E (Transposition)
  - adds to subkey (Substitution)
  - passes through 8 S-boxes to get 32-bit result (S&T)
  - finally permutes this using 32-bit perm P (transposition)

DES Round Structure

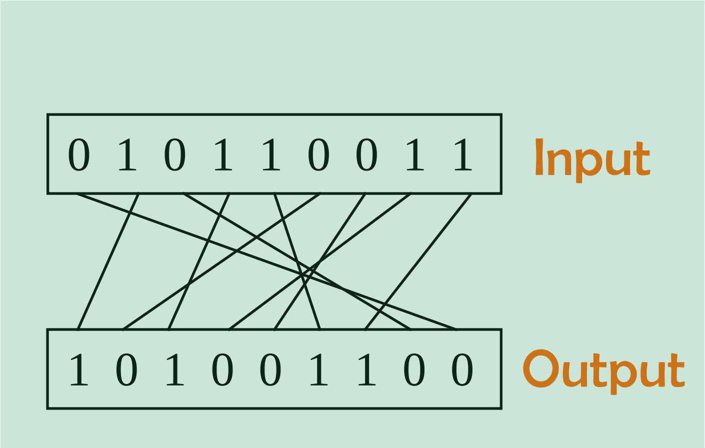


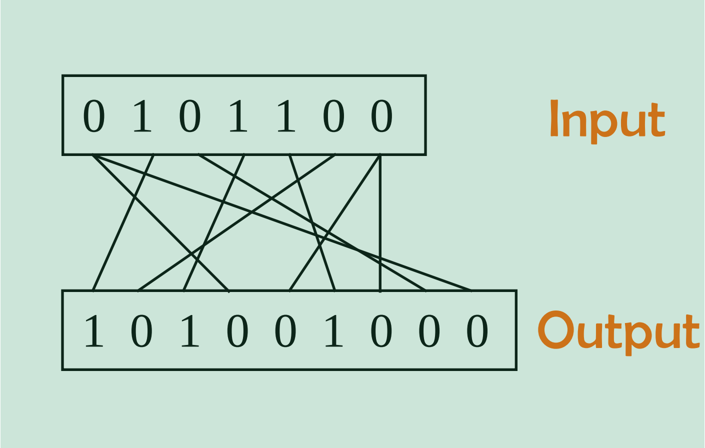
DES Module Operations

- Permutation boxes
  - Specific boxes used in DES includes: PC1 and PC2 for sub-key generation; IP, IP-1, E-box and P-box
- Substitution boxes
  - 8 specific S-boxes are used in DES; This is the core of DES; This step is non-linear
- Modulo 2 addition
  - Addition in binary form; used in function f
- 32 bits registers
  - Use only to store data. In the key generator two shift registers are used to cyclically shift the data used in key generation

Permutation

- Re-order the bit stream; e.g. 1st bit of input stream is moved to 9th bit of output stream
- Permutation: size of input and output are the same; used in DES' Initial permutation, Inverse permutation, etc
- Expansion: size of output is greater than input stream, some input bits appear at two places in output
- Compression box: size of output is smaller than input stream, then some input stream will not appear in the output





Substitution

- Substitution boxes provide a substitution code, i.e. there is a code output stored for each input

- Each S box stores a different set of 48 hexadecimal numbers in a matrix of 16×4 (total are 64 some of them are duplicated)

- There are 8 S-boxes in DES, each accepts a 6-bit input and returns a 4-bit output

- Consider a 48-bit input stream, first 6 bits input will be input to the first S box, next 6 bits will be for the second S box, and so on.

DES Key Schedule

Generating subkeys used in each round

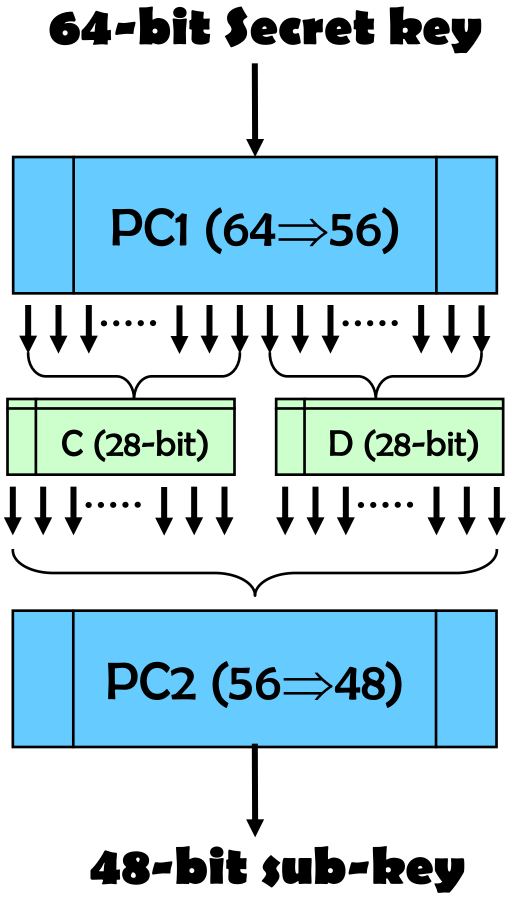
- consists of:
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - selecting 24-bits from each half
    - permuting them by PC2 for use in function f,
    - rotating each half separately either 1 or 2 places depending on the key rotation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

Sub-Key generations

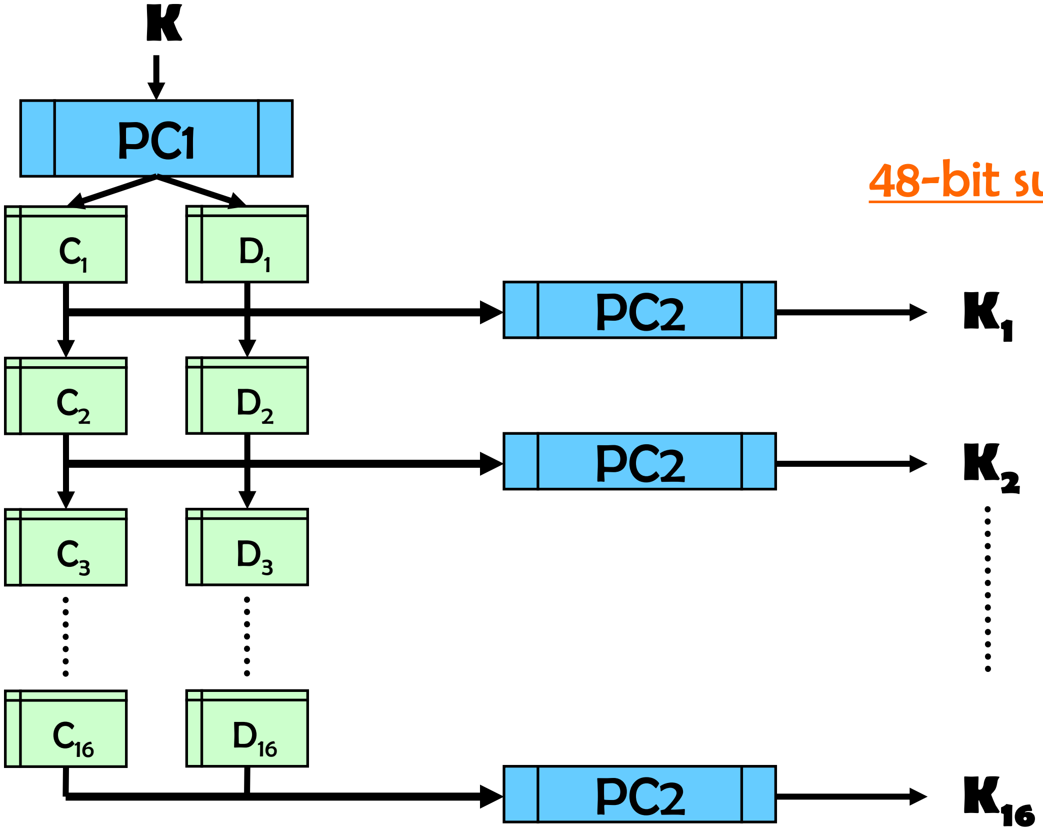
- Now, let's first learn how to generate 16 sub-keys for each round of DES, given a secret key K of 64 bits long (includes 8 parity bits) by the sender
  - K= [0101 1000 0001 1111 1011 1100 1001 0100 1101 0011 1010 0100 0101 0010 1110 1010]
  - For each byte, the 8th bit is 1 if the number of 1s in the first 7 bits is even, 0 otherwise.

One sub-key

- 64 bits of secret key are input to the key generator, 8 parity bits are removed; So, DES key has only 56 bits
- Objective: use these 56 bits to generate a different 48 bit sub-key for each round of DES
    - PC1 is a P box where 8 parity bits are removed with input of 64 bits key
    - 56-bit output of PC1 is split into two 28-bit keys which is input into shift registers C and D
    - PC2 is also a P box which ignores certain input bits and permutes to a 48-bit sub-key



Generation of Many Sub-Keys



48-bit sub-keys

Permuted Choice 1 (PC1)

- The table below specifies how the key is loaded to memory in PC1.
- If 64-bit Secret Key K = [0101 1000 0001 1111 1011 1100 1001 0100 1101 0011 1010 0100 0101 0010 1110 1010], then PC1(K) = [L R] where both L and R are 28 bits long and

L = [1011110011010001101001000101]

R = [1101001000101110100001111111]

Bit	57	49	41	33	25	17	9	1	58	50	42	34	26	18
Goes to bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Bit	10	2	59	51	43	35	27	19	11	3	60	52	44	36
Goes to bit	15	16	17	18	19	20	21	22	23	24	25	26	27	28

Bit	63	55	47	39	31	23	15	7	62	54	46	38	30	22
Goes to bit	29	30	31	32	33	34	35	36	37	38	39	40	41	42

Bit	14	6	61	53	45	37	29	21	13	5	28	20	12	4
Goes to bit	43	44	45	46	47	48	49	50	51	52	53	54	55	56

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	1	0	1	1	0	0	0	0	0	0	1	1	1	1	1
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	0	1	1	1	1	0	0	1	0	0	1	0	1	0	0
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
1	1	0	1	0	0	1	1	1	0	1	0	0	1	0	0
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
0	1	0	1	0	0	1	0	1	1	1	0	1	0	1	0

Bit	**57**	**49**	**41**	**33**	**25**	**17**	**9**	**1**	**58**	**50**	**42**	**34**	**26**	**18**
Goes to bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Bit	**10**	**2**	**59**	**51**	**43**	**35**	**27**	**19**	**11**	**3**	**60**	**52**	**44**	**36**
Goes to bit	15	16	17	18	19	20	21	22	23	24	25	26	27	28

Bit	**63**	**55**	**47**	**39**	**31**	**23**	**15**	**7**	**62**	**54**	**46**	**38**	**30**	**22**
Goes to bit	29	30	31	32	33	34	35	36	37	38	39	40	41	42

Bit	**14**	**6**	**61**	**53**	**45**	**37**	**29**	**21**	**13**	**5**	**28**	**20**	**12**	**4**
Goes to bit	43	44	45	46	47	48	49	50	51	52	53	54	55	56

Shift Registers C and D

- The contents of C = {C1, C2, … C16} and D = {D1, D2, … D16} are circularly shifted to left by 1 or 2 bits (according to a shift table) prior to each iteration
- Total of 28 bit shifts will be done after 16 rounds
- Shift tables is determined as below.
- Assume we are at the first round. According to the table, the number of shift to left =1.
- C1(L) = 0111100110100011010010001011 and
  D1(R) = 1010010001011101000011111111
- And C2(C1(L)) 1111001101000110100100010110 and
  D2(D1(R)) = 0100100010111010000111111111

Round	No. of Shift to left	Round	No. of Shift to left
1	1	9	1
2	1	10	2
3	2	11	2
4	2	12	2
5	2	13	2
6	2	14	2
7	2	15	2
8	2	16	1

Permuted Choice 2 (PC2)

- PC2 is determined by the table below
- Consider input X= [C1(L) D1(R)] and Y=[C2(L) D2(R)]
- K1 = PC2(X) = 27 A1 69 E5 8D DA HEX =
  (00100111 10100001 01101001 11100101 10001101 11011010)
- K2 = PC2(Y) = DA91 DDD7 B748HEX =
  (110110 101001 000111 011101 110101 111011 011101 001000)

Bit	**14**	**17**	**11**	**24**	**1**	**5**	**3**	**28**	**15**	**6**	**21**	**10**
Goes to bit	1	2	3	4	5	6	7	8	9	10	11	12

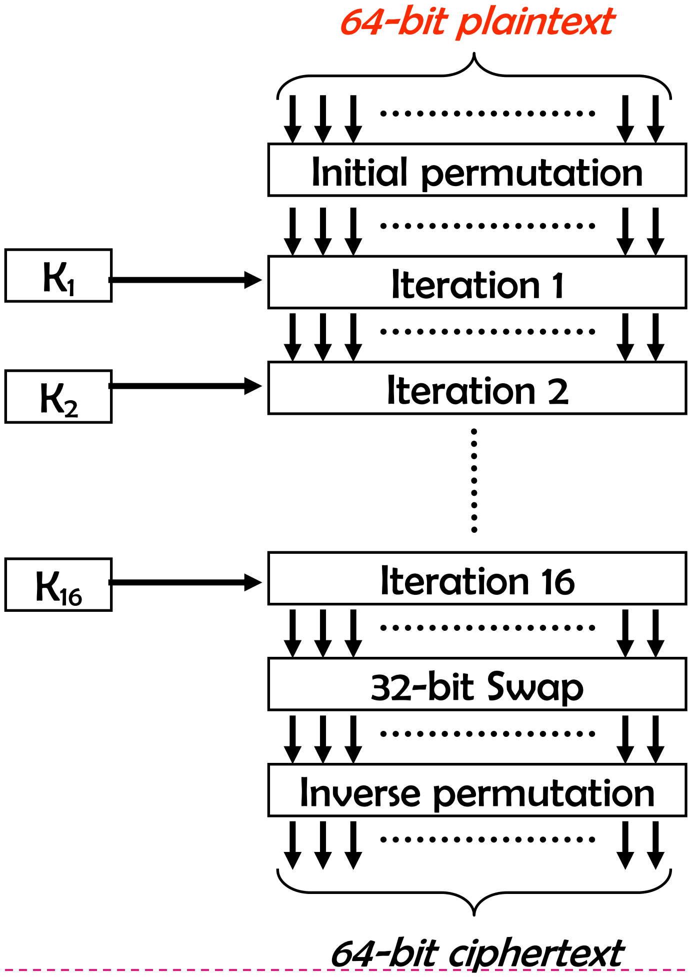
Bit	**23**	**19**	**12**	**4**	**26**	**8**	**16**	**17**	**27**	**20**	**13**	**2**
Goes to bit	13	14	15	16	17	18	19	20	21	22	23	24
Bit	**41**	**52**	**31**	**37**	**47**	**55**	**30**	**40**	**51**	**45**	**33**	**48**
Goes to bit	25	26	27	28	29	30	31	32	33	34	35	36

Bit	**44**	**49**	**39**	**56**	**34**	**53**	**46**	**42**	**50**	**36**	**29**	**32**
Goes to bit	37	38	39	40	41	42	43	44	45	46	47	48

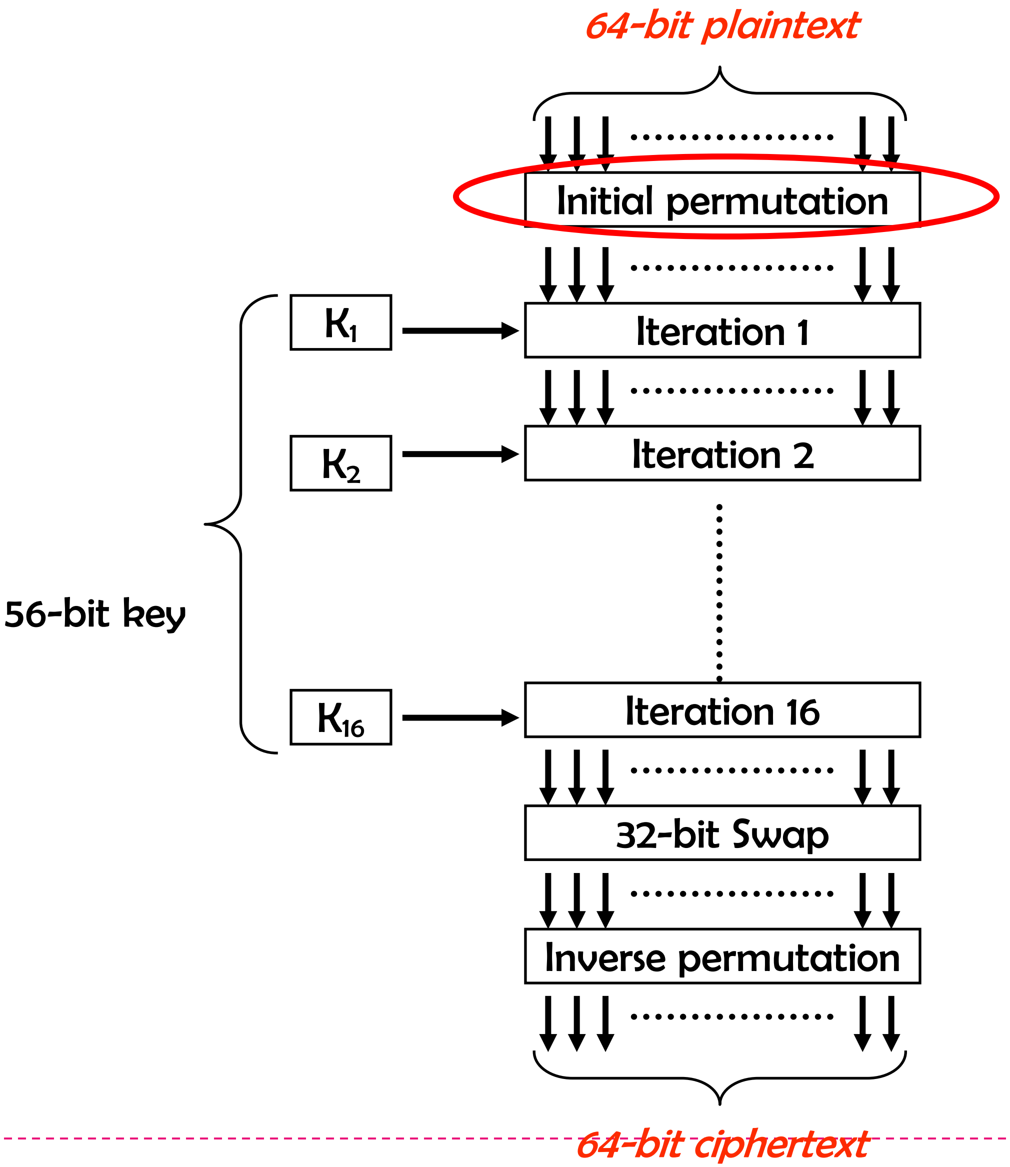
Use Sub-keys to encrypt

- Now we have K1 and K2;
- Repeat the previous process 14 more times, we will get altogether 16 sub-keys
- Assume M is the 64-bit plaintext

$M = 3570E2F1BA4682C7_{HEX}$



Initial Permutation



- 64 bits output of Initial permutation is split:
  - Left hand 32 bits sent to L
  - Right hand 32 bits sent to R

Initial Permutation (IP)

- IP is determined as the following table
- It occurs before round one
- Bits in the plaintext are moved to next location, e.g. bit 58 to bit 1, bit 50 to bit 2 and bit 42 to bit 3, etc

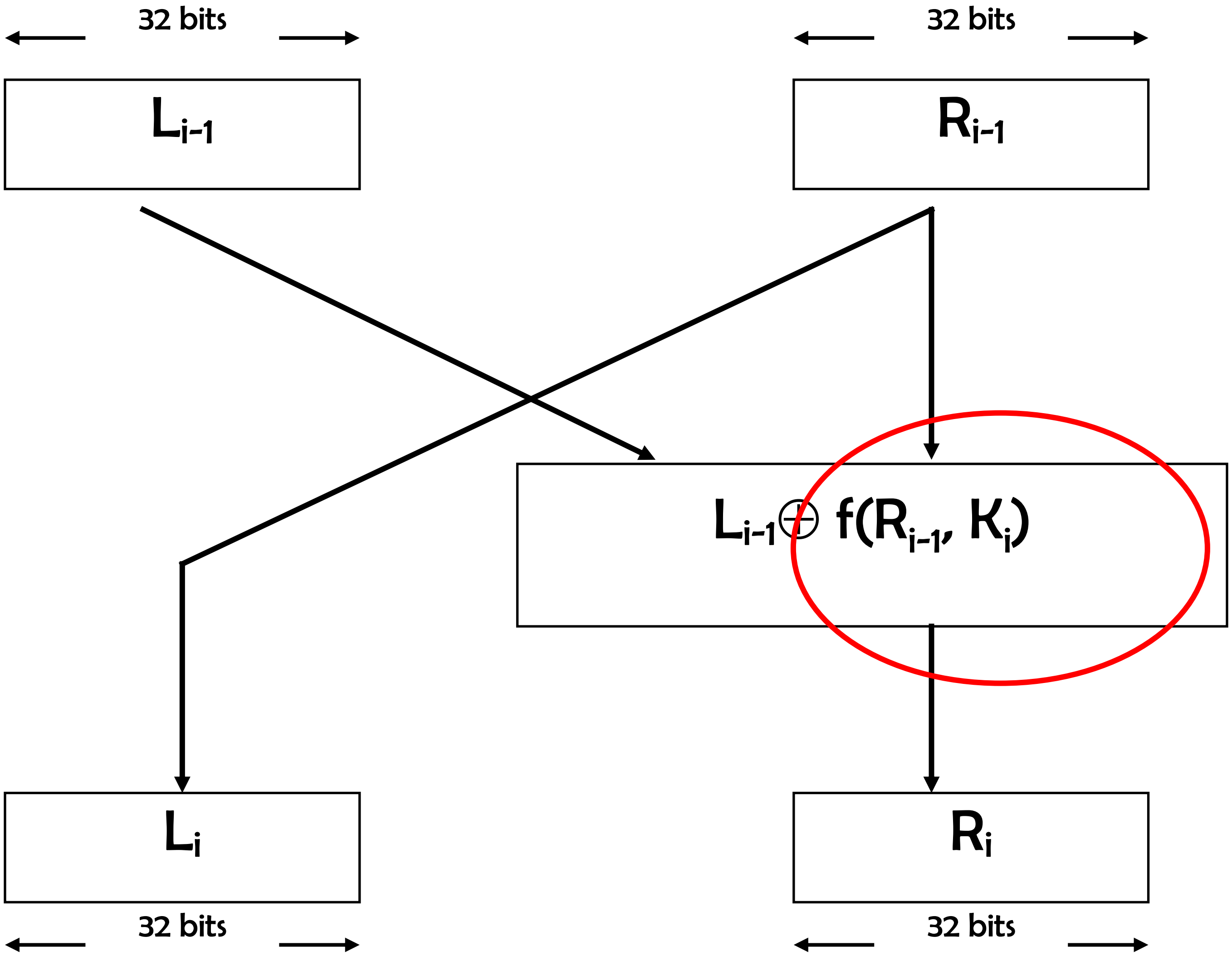
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Initial Permutation (IP)

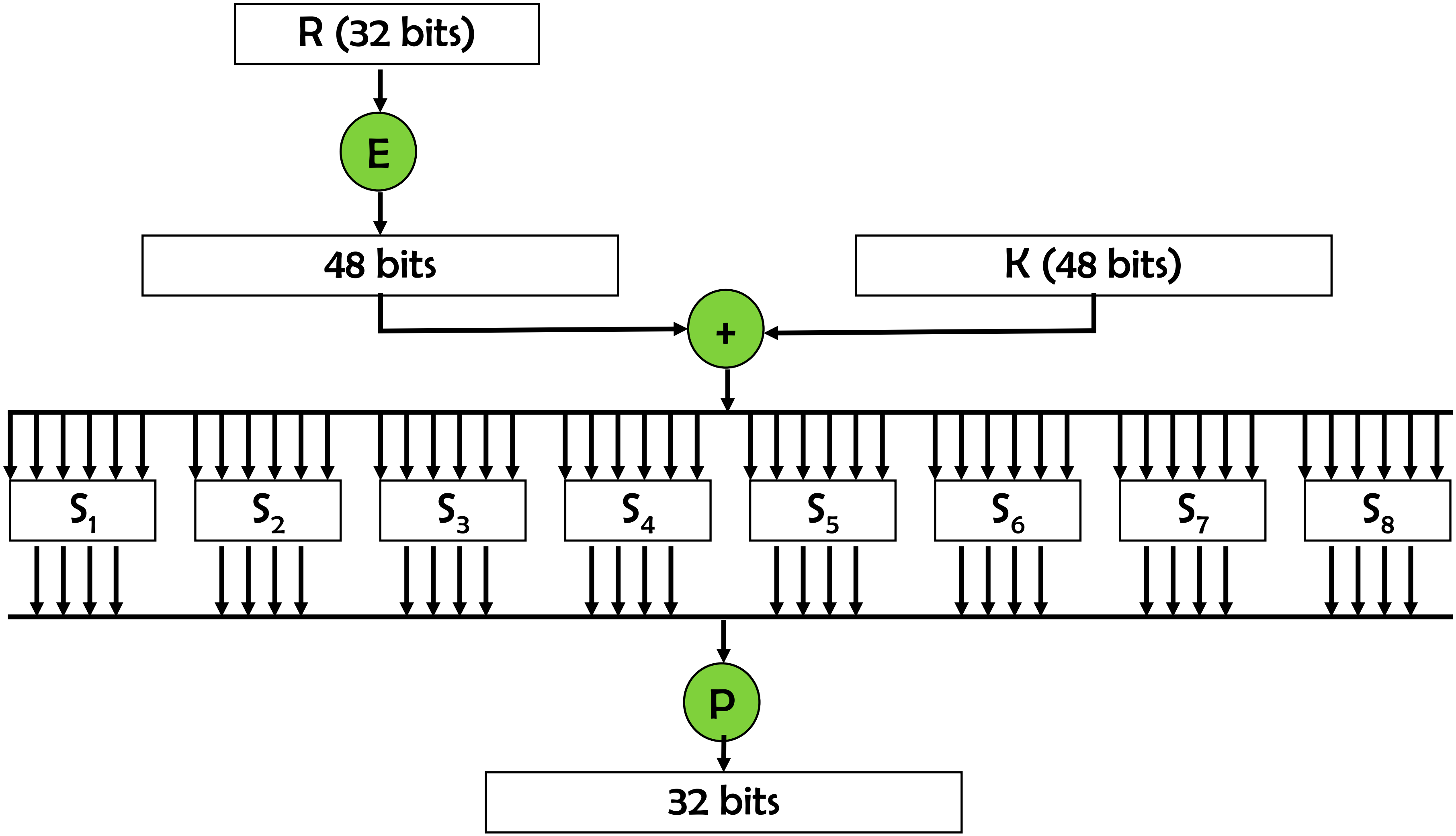
- Since M = 3570 E2F1 BA46 82C7HEX = (0011 0101 0111 0000 1110 0010 1111 0001 1011 1010 0100 0110 1000 0010 1100 0111), then IP(M) = [L0 R0] where
- L0 = 1010 1110 0001 1011 1010 0001 1000 1001 = AE1BA189HEX
- R0 = 1101 1100 0001 1111 0001 0000 1111 0100 = DC1F10F4HEX
- Now we have L0 and R0 ready for iteration!

Operations in Each Round

Structure



f(Ri-1, Ki)



Computation of f(Ri-1, Ki)

- Three types of boxes: E, S, P
- R (32 bits) is passed to expansion and permutation box E-box
- 48 bits output of E-box is added modulo 2 to 48 bits sub-key and result sent to S boxes
- S boxes (S1, S2…S8) store a set of numbers; input 48 (=6×8) bits used to look up numbers like a code book and 32 bits output is sent to permutation box P
- Permutation box P permutes 32 bit input producing a 32-bit output

E-box used in DES

- The E-box expands 32 bits to 48 bits; it changes the order of the bits as well as repeating certain bits.

Bit	**32**	**1**	**2**	**3**	**4**	**5**	**4**	**5**	**6**	**7**	**8**	**9**
Goes to bit	1	2	3	4	5	6	7	8	9	10	11	12

Bit	**8**	**9**	**10**	**11**	**12**	**13**	**12**	**13**	**14**	**15**	**16**	**17**
Goes to bit	13	14	15	16	17	18	19	20	21	22	23	24

Bit	**16**	**17**	**18**	**19**	**20**	**21**	**20**	**21**	**22**	**23**	**24**	**25**
Goes to bit	25	26	27	28	29	30	31	32	33	34	35	36

Bit	**24**	**25**	**26**	**27**	**28**	**29**	**28**	**29**	**30**	**31**	**32**	**1**
Goes to bit	37	38	39	40	41	42	43	44	45	46	47	48

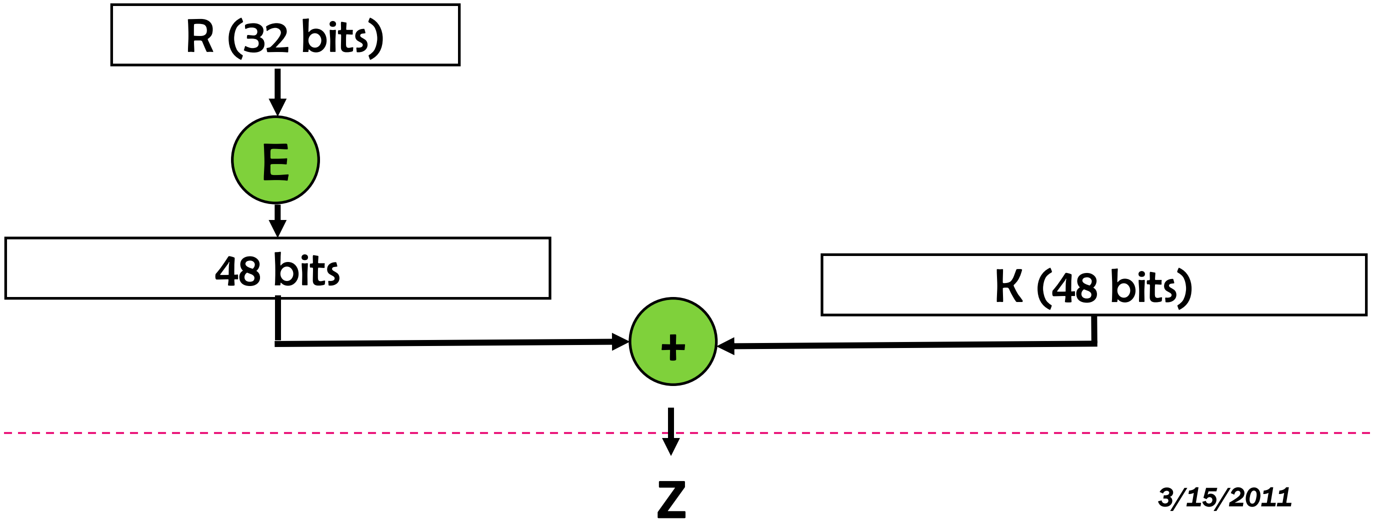
Substitution Boxes S

- 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(1809123d11173839) = 5fd25e03

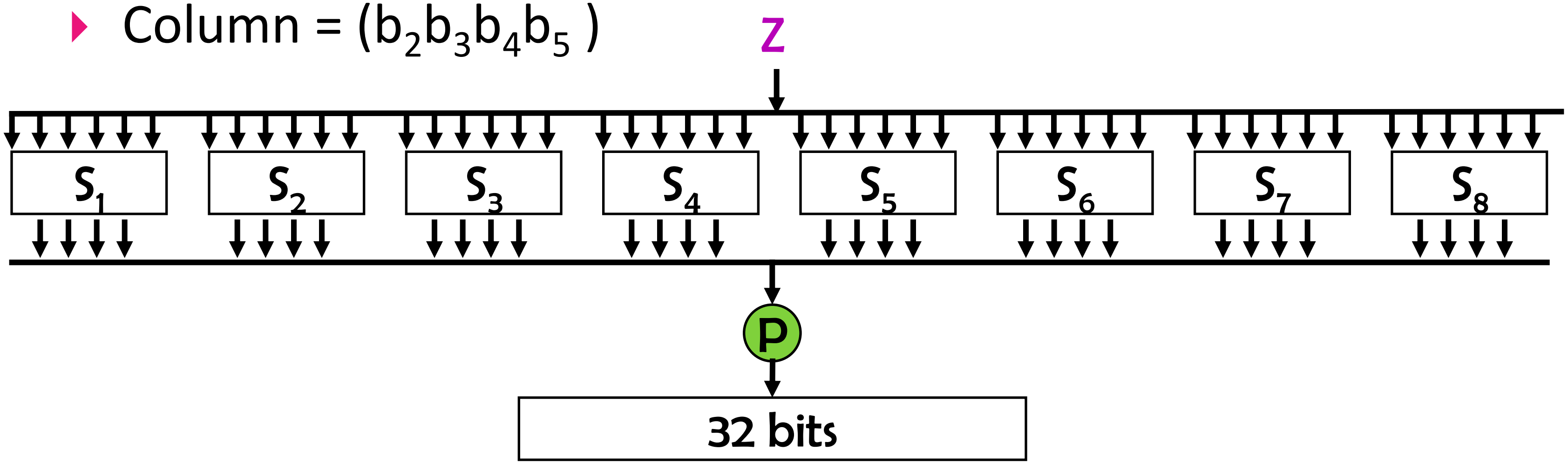
Input of S-boxes

- R0 = DC1F10F4$_{HEX}$ and
- K = K0 = 27A169E58DDA$_{HEX}$; (here K is not the secret key but a symbol for all sub-keys)
- $\Rightarrow$ E(R0) = 0110 1111 1000 0000 1111 1110 1000 1010 0001 0111 1010 1001 = 6F80FE8A 17A9$_{HEX}$
- $\Rightarrow$ E(R0) $\oplus$ K0 = 0100 1000 0010 0001 1001 0111 0110 1111 1001 1010 0111 0011 = 4821976F9A73$_{HEX}$
- $\Rightarrow$ Input Z = 4821976F9A73$_{HEX}$ into S-boxes



S-box

- After the sub-key is XORed with the expanded right blocked, 48-bit result moves to the substitution operation, S-boxes
- The S-boxes in DES swap bits around in the 48-bit block in a reversible manner
- Each S-box are differently defined.
- Each input “$b_1b_2b_3b_4b_5b_6$”, S box will output a hexadecimal number at
  - Row = (b_1b_6)
  - Column = $(b_2b_3b_4b_5\ )$



S-box used in DES

- The 48-bit input (from Z) is separated into eight 6-bit blocks (B_{1-8}).
- Each block is subjected to a unique substitution function (S_{1-8}) yielding a 4-bit block as output.
- This is done by taking the first and last bits of the block to represent a 2-digit binary number (i) in the range of 0 to 3.
- The middle 4 bits of the block represent a 4-digit binary number (j) in the range of 0 to 15.
- The unique substitution number to use is the one in the i^{th} row and j^{th} column, which is in the range of 0 to 15 and is represented by a 4-bit block.

S1 and S2

- Since Z= $4821976F9A73_{HEX}$ = 010010 000010 000110 010111 011011 111001 101001 110011
- $\Rightarrow S_1(010010)$ is the value 10 (at row 0 and column $1001_2 = 9_{10}$)
- $\Rightarrow S_2(000010) = 1_{10} = 0001_2$ (at row 0 and column $0001_2 = 1_{10}$)

S_1

Row 0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
Row 1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
Row 2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
Row 3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2

Row 0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
Row 1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
Row 2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
Row 3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3 and S4

- Since Z= $4821976F9A73_{HEX}$ = 010010 000010 000110 010111 011011 111001 101001 110011
- $\Rightarrow S_3(000110) = 14_{10} = 1110_2$
- $\Rightarrow S_4(010111) = 12_{10} = 1100_2$

S_3

Row 0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
Row 1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
Row 2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
Row 3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4

Row 0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
Row 1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
Row 2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
Row 3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S5 and S6

- Since Z= 4821 976F 9A73$_{HEX}$ = 010010 000010 000110 010111 011011 111001 101001 110011
- $\Rightarrow S_5(011011) = 9_{10} = 1001_2$
- $\Rightarrow S_6(111001) = 6_{10} = 0110_2$

S_5

Row 0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
Row 1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
Row 2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
Row 3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6

Row 0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
Row 1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
Row 2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
Row 3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S7 and S8

- Since Z= $4821976F9A73_{HEX}$ = 010010 000010 000110 010111 011011 111001 101001 110011
- $\Rightarrow S_7(101001) = 1_{10} = 0001_2$
- $\Rightarrow S_8(010011) = 9_{10} = 1100_2$

S_7

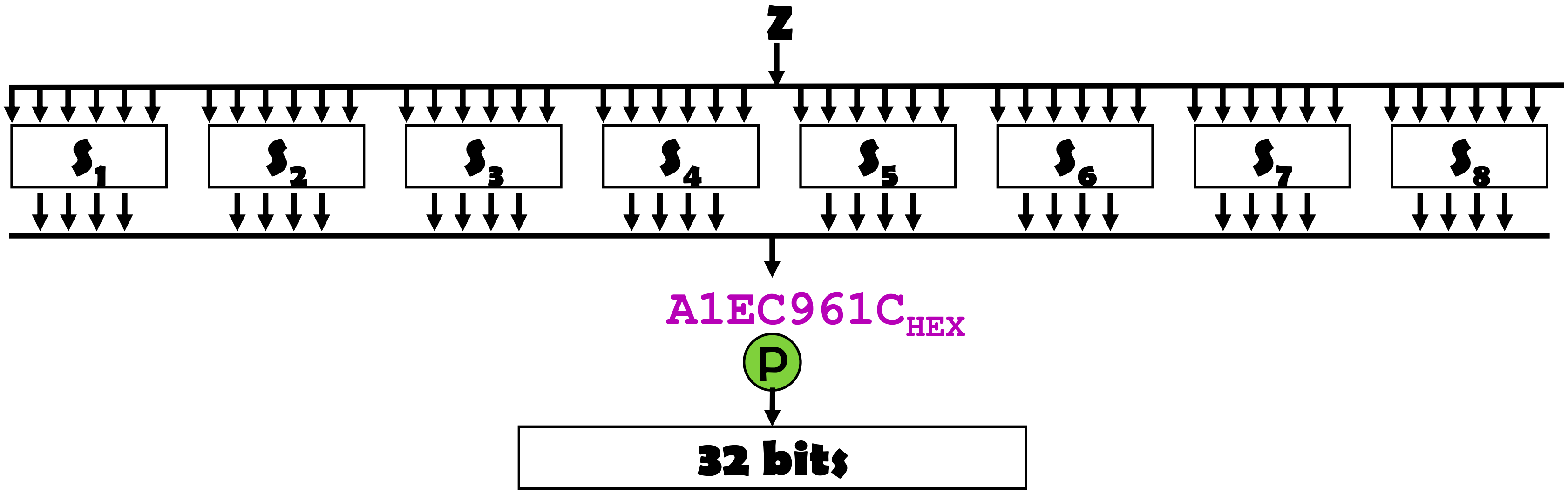
Row 0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
Row 1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
Row 2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
Row 3	6	11	13	8	1	4	0	7	9	5	0	15	14	2	3	12

S_8

Row 0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
Row 1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
Row 2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
Row 3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Combine all 8 S-boxes

- Now we have all outputs from 8 S-boxes
- S(Z) = 1010 0001 1110 1100 1001 0110 0001 1100 = A1EC961C$_{HEX}$
- Input the result into P-box!



3/15/2011

P-box used in DES

- The P-box permutation is determined as below which is a straight permutation; no bits are used twice, and no bits are ignored.
- ⇒ P(**A1EC961C**$_{\text{HEX}}$) = 0010 1011 1010 0001 0101 0011 0110 1100 = **2BA1536C**$_{\text{HEX}}$

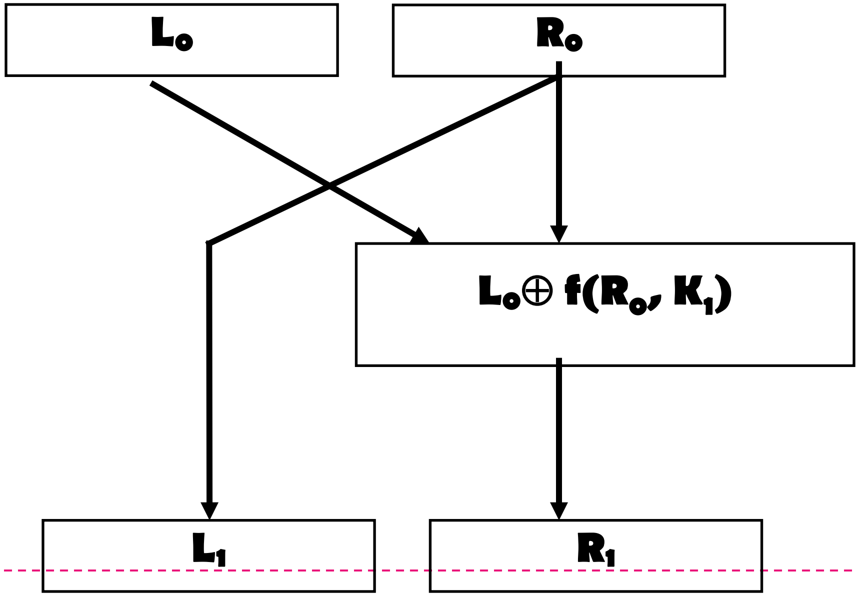
Bit	16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
Goes to bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Bit	2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25
Goes to bit	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

An example for first two rounds

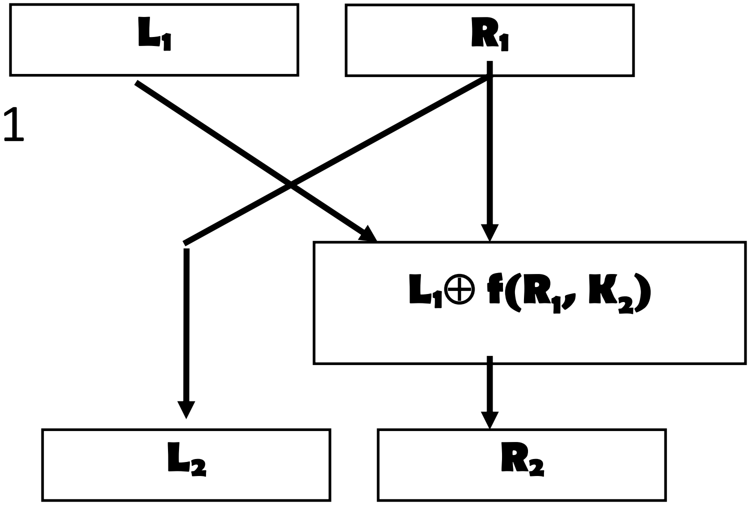
First Round

- L_0 = **AE1BA189$_{HEX}$** and R_0 = **DC1F10F4$_{HEX}$**
- Sub-key K_1 = **27A169E58DDA$_{HEX}$**
- $f(R_0,K_1)$ = **2BA1536C$_{HEX}$**
- $\Rightarrow L_0 \oplus f(R_0,K_1)$=1000 0101 1011 1010 1111 0010 1110 0101=**85BAF2E5$_{HEX}$**
- $\Rightarrow$ L_1 = **DC1F10F4$_{HEX}$** and R_1 = **85BAF2E5$_{HEX}$**



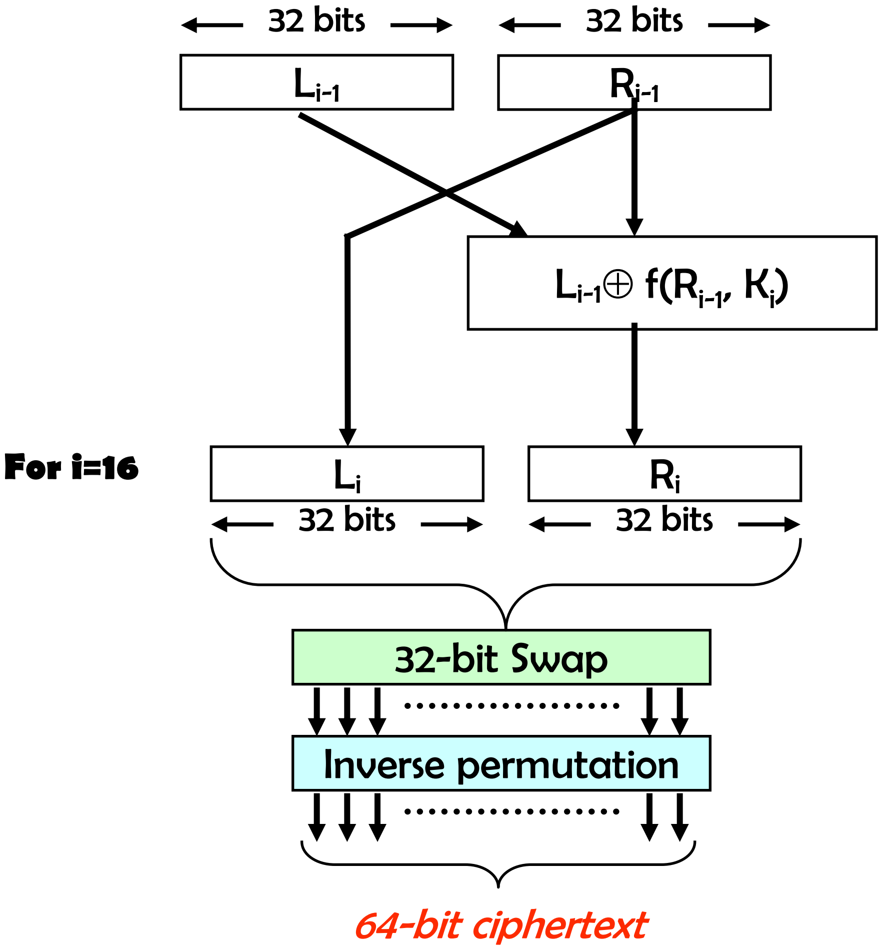
Second Round

- L_1 = **DC1F10F4$_{HEX}$** and R_1 = **85BAF2E5$_{HEX}$**
- Sub-key K_2 = **DA91DDD7B748$_{HEX}$**
- $E(R_1)$ =110000001011110111111010101111010010101110000101 1
  = **C0BDF57A570B$_{HEX}$**
- $E(R_1)\oplus K_2$=$\underline{000110}100010\underline{110000}101000\underline{101011}011110\ \underline{000001}000011$
- S_1(000110)=0001; S_2(100010)=1110;S_3(110000)=1011; S_4(101000)=1100;
  S_5(101011)=1110; S_6(011110)=1011; S_7(000001)=1101; S_8(000011)=1111
- P(8 outputs of S-boxes) = 0101 1111 0011 1110 0011 1001 1111 0111
  = **5F3E39F7$_{HEX}$** = $f(R_1,K_2)$
- $\Rightarrow L_1\oplus f(R_1,K_1)$
  = 1000 0011 0010 0001 0010 1001 0000 0011
  = 8321 2903$_{HEX}$
- $\Rightarrow$ L_2= R_1 = **85BAF2E5$_{HEX}$** ; R_2= **83212903$_{HEX}$**



The last step to get Ciphertext

DES – Ciphertext



- Express DES encryption algebraically (in binary number)
  - ⇨ $R_j = L_{j-1} \oplus f(R_{j-1}, K_j)$
  - ⇨ $L_j = R_{j-1}$
- After 16 rounds of iterations, the contents of L and R are swapped and input to Inverse permutation
- Finally, a 64-bit ciphertext is done!

Inverse Initial Permutation (IP^{-1})

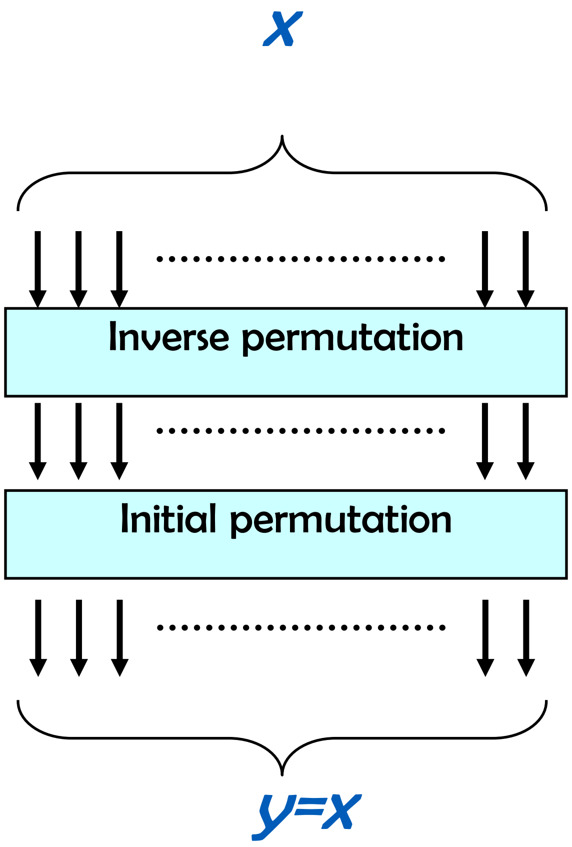
- IP^{-1} is determined as the following table;
- Since DES consists of 16 rounds, too many for our lecture!
- Consider DES algorithm of two rounds.
- Ciphertext = $IP^{-1}(R_1L_1)$ = 1101 0111 0110 1001 1000 0010 0010 0100 0010 1000 0011 1110 0000 1010 1110 1010 = **$\mathrm{D7698224283E0AEA}_{HEX}$**

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

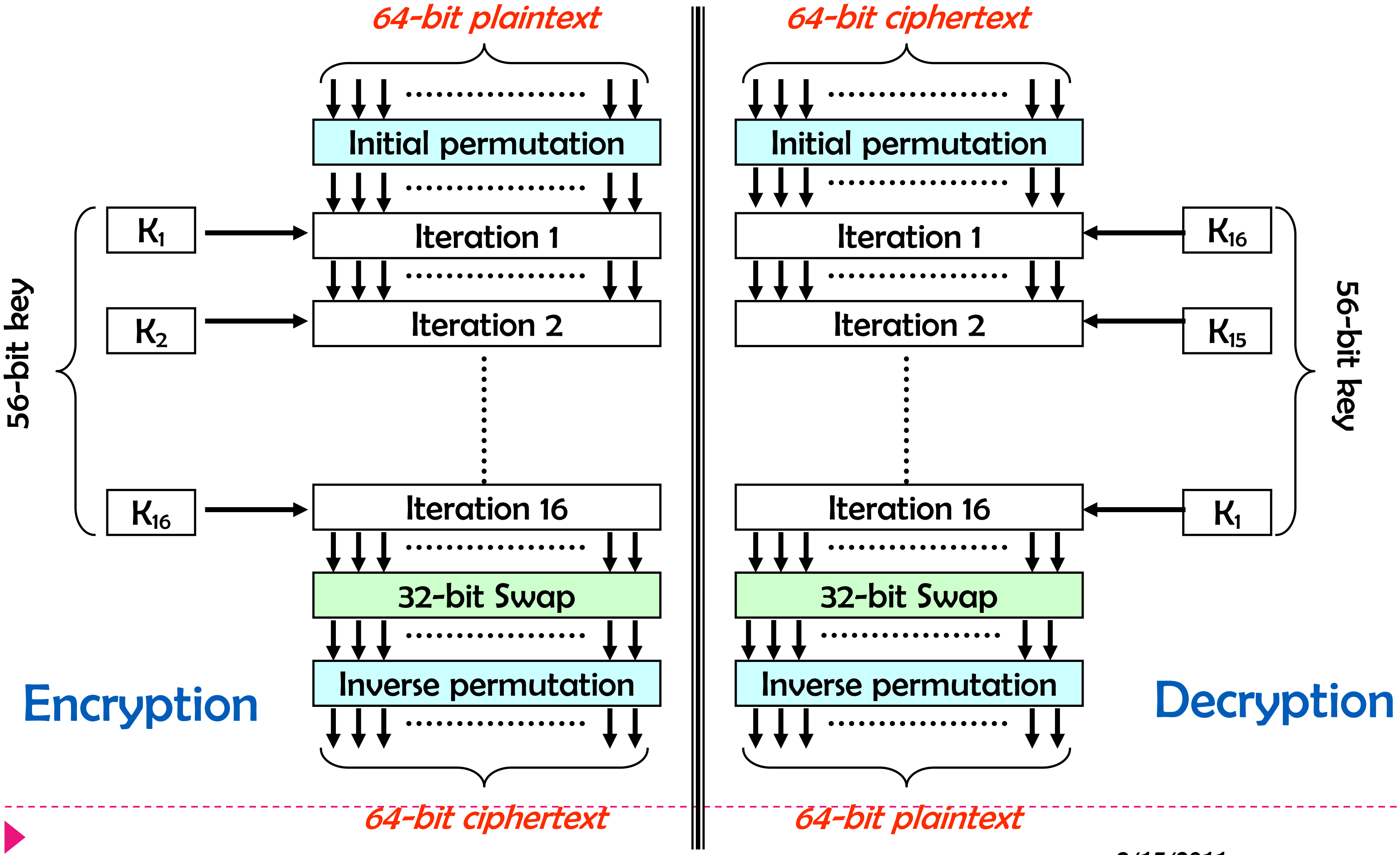
Data Encryption Standard

- Overview
- Encryption
- Decryption
- Security

- DES decryption is straightforward;
  - ⇨ e.g. to permute *n* bits with inverse permutation and then initial permutation will do nothing on the *n* bits
- Decryption processes are almost the same except that
  - ⇨ 16 sub-keys are entered in reverse order
  - ⇨ Decryption sub-keys are formed using a different shift table with C and D shifts to the right in stead of the left



DES Encryption & Decryption



3/15/2011

Detailed Description

- Decrypt must “undo” 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

Algebraic Expressions

Encryption (M)

- Input plaintext to Initial permutation box to get L_0 and R_0
- Repeat 15 times with

  $R_j=L_{j-1}\oplus f(R_{j-1},K_j)$

  $L_j=R_{j-1}$

  to get L_{16} and R_{16}
- Swap them to get $R_{16}L_{16}$
- Put $R_{16}L_{16}$ to Inverse permutation box to get ciphertext

Decryption (C)

- Input ciphertext to Initial permutation box to get A_{16} and B_{16}
- Repeat 15 times with

  $B_{j-1}=A_j\oplus f(B_j,K_j)$

  $A_{j-1}=B_j$

  to get A_0 and B_0
- Swap them to get B_0A_0
- Put B_0A_0 to Inverse permutation box to get back the plaintext

A Simple Example

- Consider there are only 2 rounds in DES
- Given ciphertext = C = $D7698224283E0AEA_{HEX}$
- Let's decipher it to get back our plaintext M.
- Normally, in deciphering operation, sub-key must be used in reversed order; i.e. K_{16}, K_{15},…
- In our case, we will use K_2 and then K_1 only
- Also, those shift registers C = $\{C_1, C_2\}$ and D = $\{D_1,D_2\}$ will be altered to right shift
- $\Rightarrow$IP(C) = **$8321290385BAF2E5_{HEX}$**
- $\Rightarrow$Let A_2= **83212903_{HEX}** and B_2= **$85BAF2E5_{HEX}$**

Decryption

- A_2= **83212903**$_{HEX}$ and B_2= **85BAF2E5**$_{HEX}$
- First Round
  - ⇨ $E(B_2)$=110000 001011 110111 110101 011110 100101 011100 001011
  - ⇨ $E(B_2)\oplus K_2$=000110 100010 110000 101000 101011 011110 000001 000011
  - ⇨ S_1(000110)=0001; S_2(100010)=1110; S_3(110000)=1011; S_4(101000)=1100; S_5(101011)=1110; S_6(011110)=1011; S_7(000001)=1101; S_8(000011)=1111;
  - ⇨ Let S=0001 1110 1011 1100 1110 1011 1101 1111
  - ⇨ P(S)=0101 1111 0011 1110 0011 1001 1111 0111
  - ⇨ $P(S)\oplus A_2$= 1101 1100 0001 1111 0001 0000 1111 0100 = **DC1F10F4**$_{HEX}$

Decryption

- B_1= **DC1F10F4$_{HEX}$** and A_1= B_2 = **85BAF2E5$_{HEX}$**
- Second Round
  - ⇨ $E(B_1)$=011011 111000 000011 111110 100010 100001 011110 101001
  - ⇨ $E(B_1)\oplus K_1$=010010 000010 000110 010111 011011 111001 101001 110011
  - ⇨ S_1(010010)=1010; S_2(000010)=0001; S_3(000110)=1110; S_4(010111)=1100; S_5(011011)=1001; S_6(111001)=0110; S_7(101001)=0001; S_8(110011)=1100;
  - ⇨ Let S = 1010 0001 1110 1100 1001 0110 0001 1100
  - ⇨ P(S) = 0010 1011 1010 0001 0101 0011 0110 1100
  - ⇨ $P(S)\oplus A_1$= 1010 1110 0001 1011 1010 0001 1000 1001 = AE1BA189$_{HEX}$
- B_0 = AE1B A189$_{HEX}$ and A_0 = B_1= DC1F 10F4$_{HEX}$
- $IP^{-1}(B_0A_0)$ = 0011 0101 0111 0000 1110 0010 1111 0001 1011 1010 0100 0110 1000 0010 1100 0111 = 3570 E2F1 BA46 82C7$_{HEX}$ = **M**

Data Encryption Standard

- Overview
- Encryption
- Decryption
- Security

Strength of DES

- 56-bit key (2^{56} = 7.2 x 10^{16} values) is susceptible to exhaustive key search due to rapid advances in computing speed

- Have demonstrated breaks
  - 1997 on a large network of computers in a few months
  - 1998 on dedicated H/W in a few days (www.eff.org/descracker)
    - EFF (Electronic Frontier Foundation) DES Cracker
    - 1536 chips and search 88 billion keys/second
    - $250,000 cost, won the RSA DES Challenge II Contest in less than 3 days (56 hours)
  - 1999 above combined in 22 hours !! (DES Cracker + 100,000 computers)
  - DES also theoretically broken using Differential or Linear Cryptanalysis

- DES Controversy
  - Although the standard is public, the design criteria used are classified

Key property

- **Avalanche**
  - ⇨ small change in plaintext or in key produces significant change in ciphertext
- **test for avalanche**
  - ⇨ encrypt two plaintext blocks that differ only in one bit
  - ⇨ about half the (ciphertext) bits will differ
- **Strict Avalanche Effect**
  - ⇨ any output bit j of an S-box should change with probability 1/2 when any single input bit i is inverted for all i, j.
- **Bit Independence Criteria**
  - ⇨ output bits j and k should change independently when any single input bit i is inverted, for all i, j, and k.

DES Design Criteria

- as reported by Coppersmith in [COPP94]

- 7 criteria for S-boxes provide for
  - ⇨non-linearity
  - ⇨resistance to differential cryptanalysis
  - ⇨good confusion

- 3 criteria for permutation P provide for
  - ⇨increased diffusion

DES S-boxes Design Criteria

- No output bit of any S-box should be too close a linear function of the input bits. Specifically, if we select any output bit and any subset of the six input bits, the fraction of inputs for which this output bit equals the XOR of these input bits should not be close to 0 or 1, but rather should be near 1/2.
- Each row of an S-box (determined by a fixed value of the leftmost and rightmost input bits) should include all 16 possible output bit combinations.
- If two inputs to an S-box differ in exactly one bit, the outputs must differ in at least two bits.
- If two inputs to an S-box differ in the two middle bits exactly, the outputs must differ in at least two bits.
- If two inputs to an S-box differ in their first two bits and are identical in their last two bits, the two outputs must not be the same.
- For any nonzero 6-bit difference between inputs, no more than 8 of the 32 pairs of inputs exhibiting that difference may result in the same output difference.

Block Cipher Design

- basic principles still like Feistel’s in 1970’s
- number of rounds
  - ⇨ more is better, exhaustive search best attack
- function f:
  - ⇨ provides “confusion”, is nonlinear, avalanche
  - ⇨ have issues of how S-boxes are selected
- key schedule
  - ⇨ complex subkey creation, key avalanche

Thank You