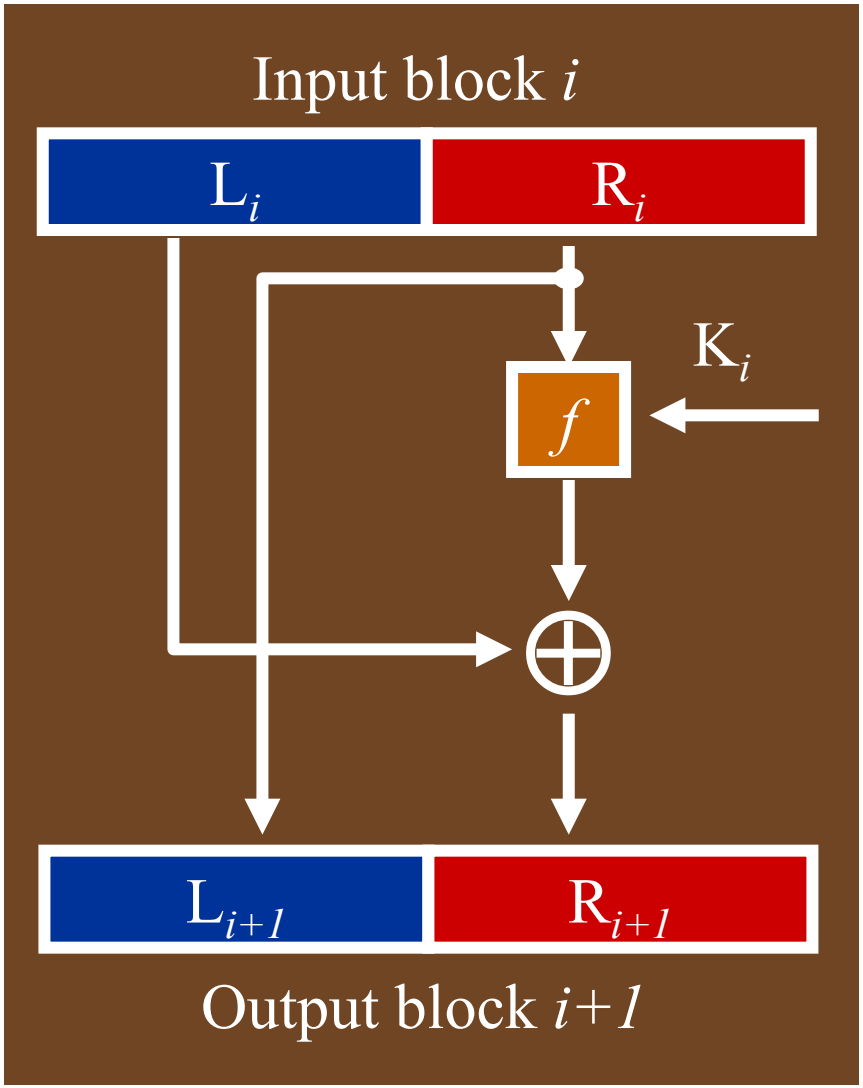
CS 4173/5173
COMPUTER SECURITY

Data Encryption Standard (DES)

OUTLINE LAST TIME

- Key size
  - 56 bits used by DES were adequate in the 80's
  - 128 bits are adequate for now
  - 5 more key bits per decade to stay "sufficiently" hard to break
- Feistel Cipher
  - Decryption is the same as encryption, only reversing the order in which round keys are applied.
  - Function f can be anything

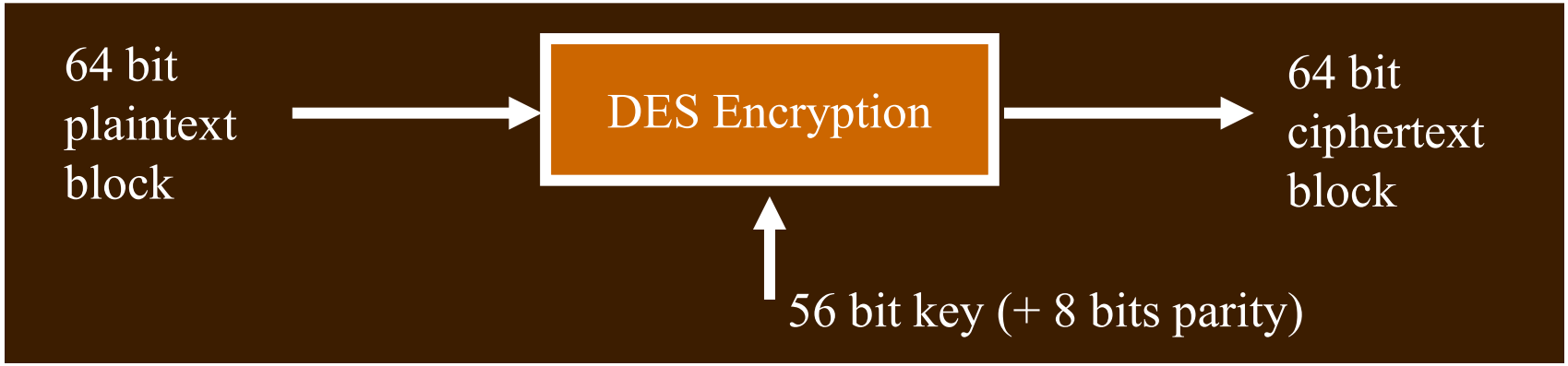


DES (DATA ENCRYPTION STANDARD)

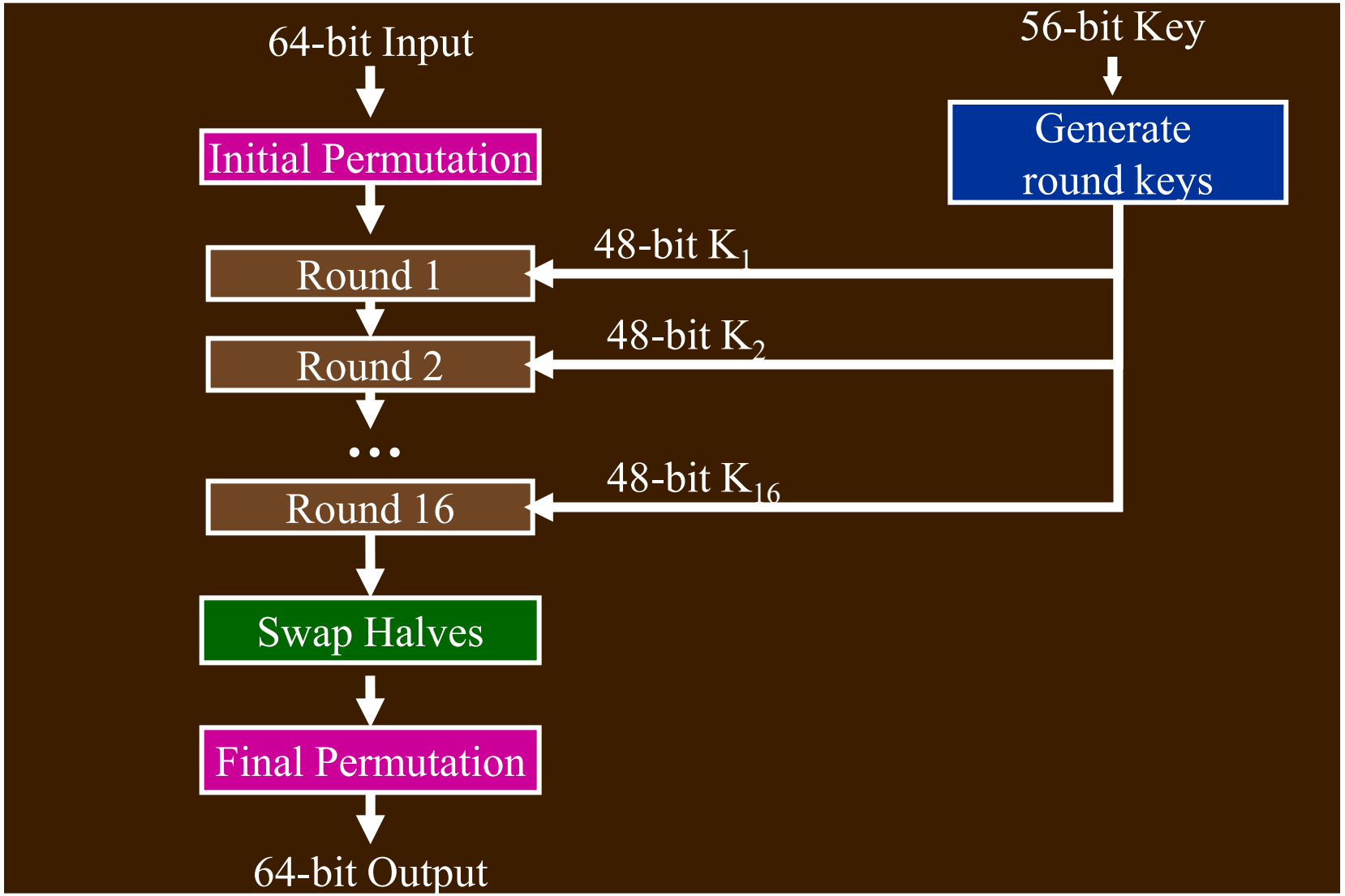
- Standardized in 1976 by NBS (National Bureau of Standards), now NIST (National Institute of Science and Technologies)
  - proposed by IBM,
  - Feistel cipher
- Criteria (official)
  - provide high level of security
  - security must reside in key, not algorithm
  - not patented
  - efficient to implement in hardware (make slow in software).

DES BASICS

- Blocks: 64 bit plaintext input,
  64 bit ciphertext output
- Rounds: 16
- Key: 64 bits
  - every 8th bit is a parity bit, so really 56 bits long



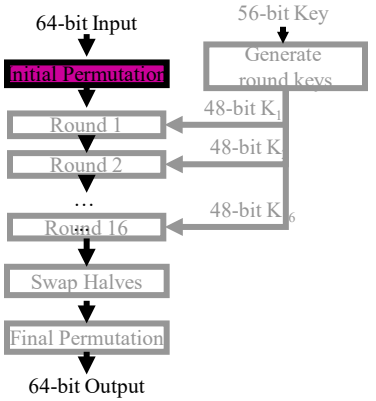
DES TOP LEVEL VIEW



INITIAL AND FINAL PERMUTATIONS

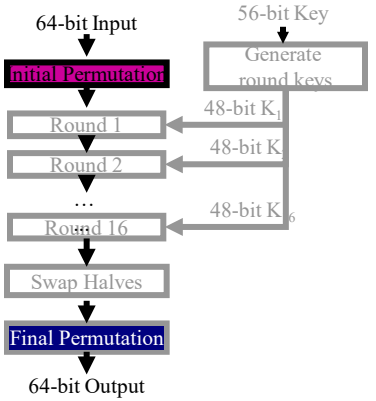
- Initial permutation given below
  - input bit #58→output bit #1, input bit #50→ output bit #2, …

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… (CONT'D)

- Final permutation is just inverse of initial permutation, i.e.,
  - input bit #1→ output bit #58
  - input bit #2→ output bit #50
  - …



INITIAL… (CONT'D)

- Note: Initial Permutation is fully specified (independent of key)
  - Does this improve security?
  - why needed?

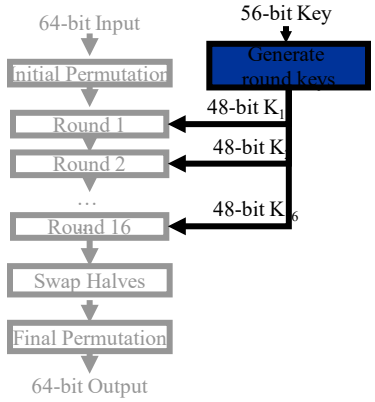
KEY GENERATION: FIRST PERMUTATION

- First step: throw out 8 parity bits, then permute resulting 56 bits

7 columns

8 rows

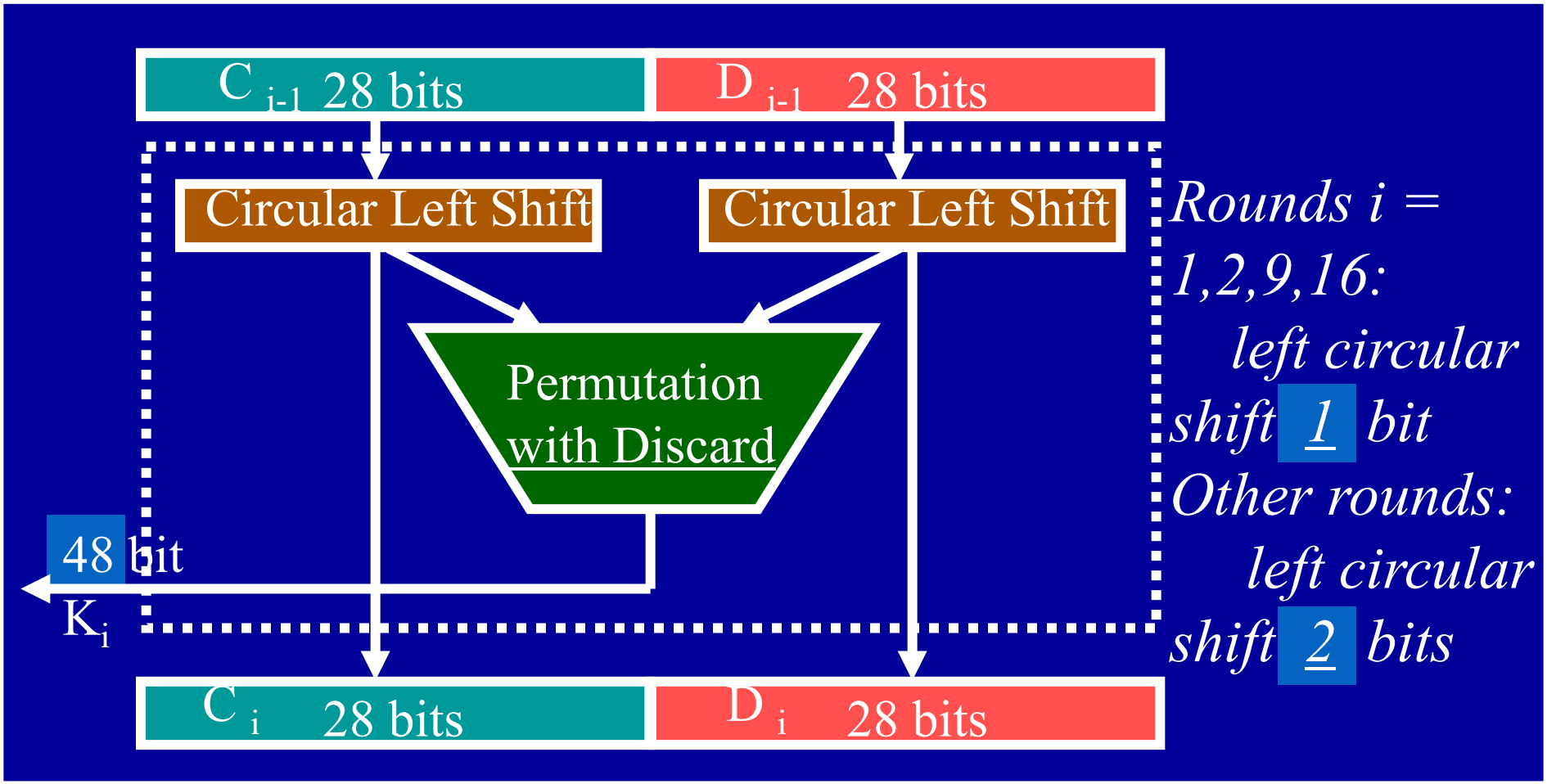
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4



Parity bits left out: 8,16,24,…

The output: all bits from left to right, top to bottom

KEYGEN: PROCESSING PER ROUND



KEYGEN: PERMUTATION WITH DISCARD

- 28 bits → 24 bits, each half of key

Left half of K_i = permutation of C_i

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2

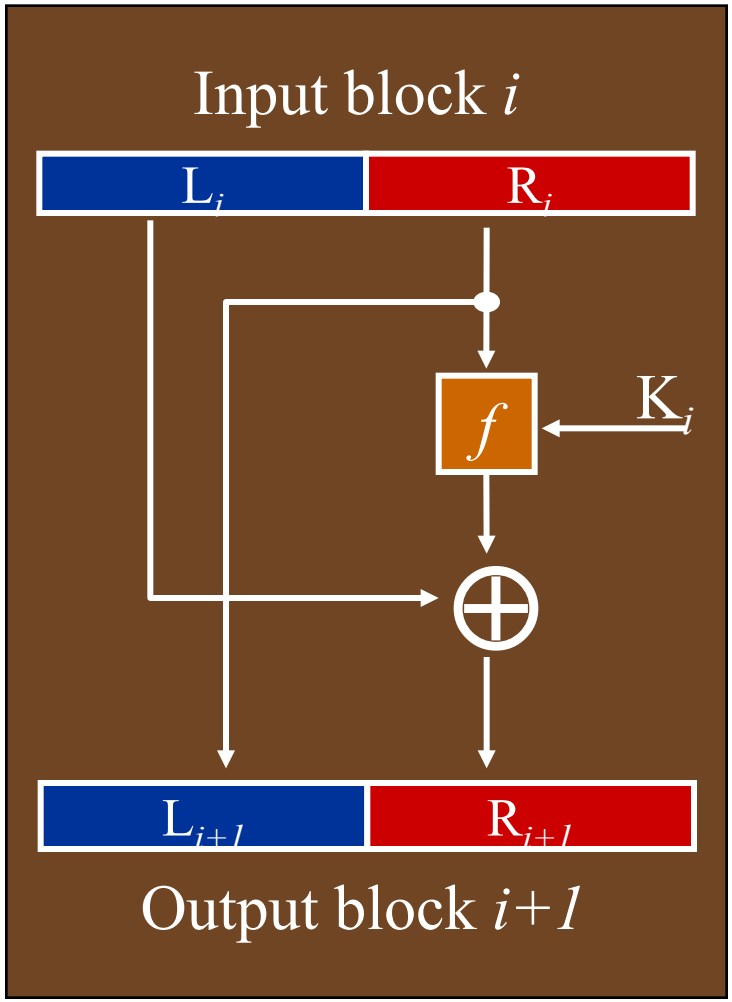
Bits discarded: 9,18,22,25

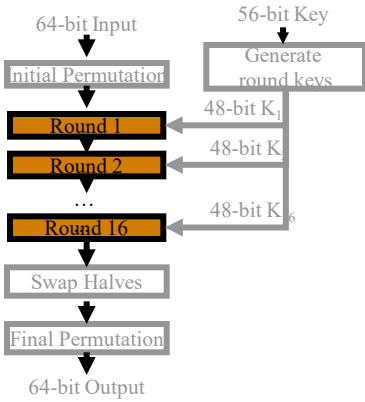
Right half of K_i = permutation of D_i

41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Bits discarded: 35,38,43,54

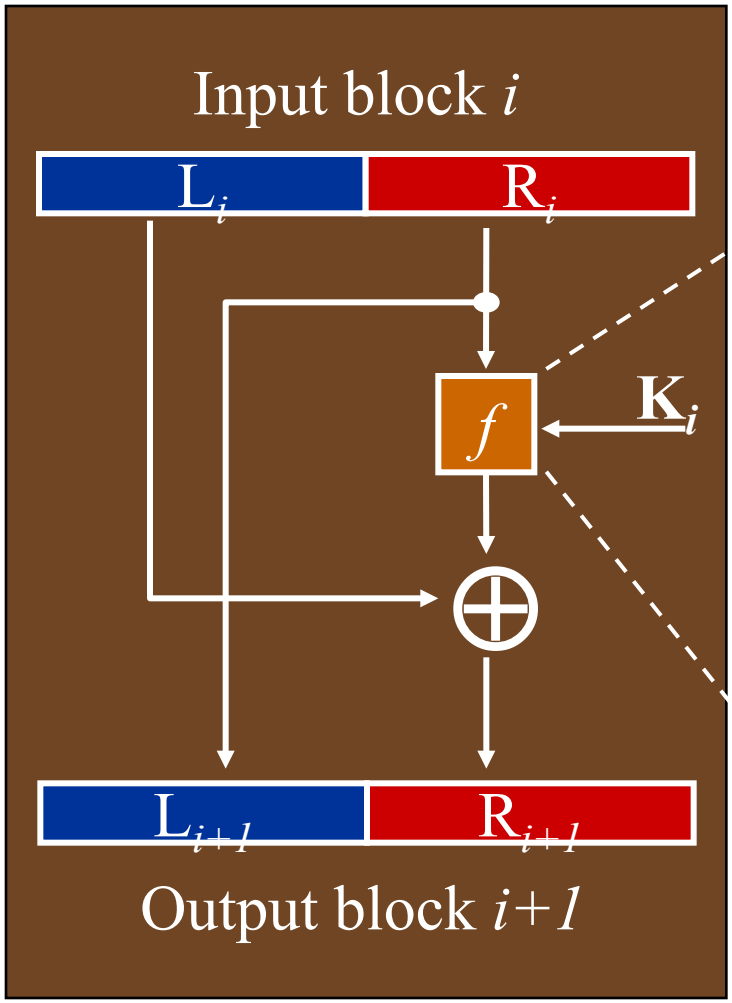
ONE DES (FEISTEL) ROUND

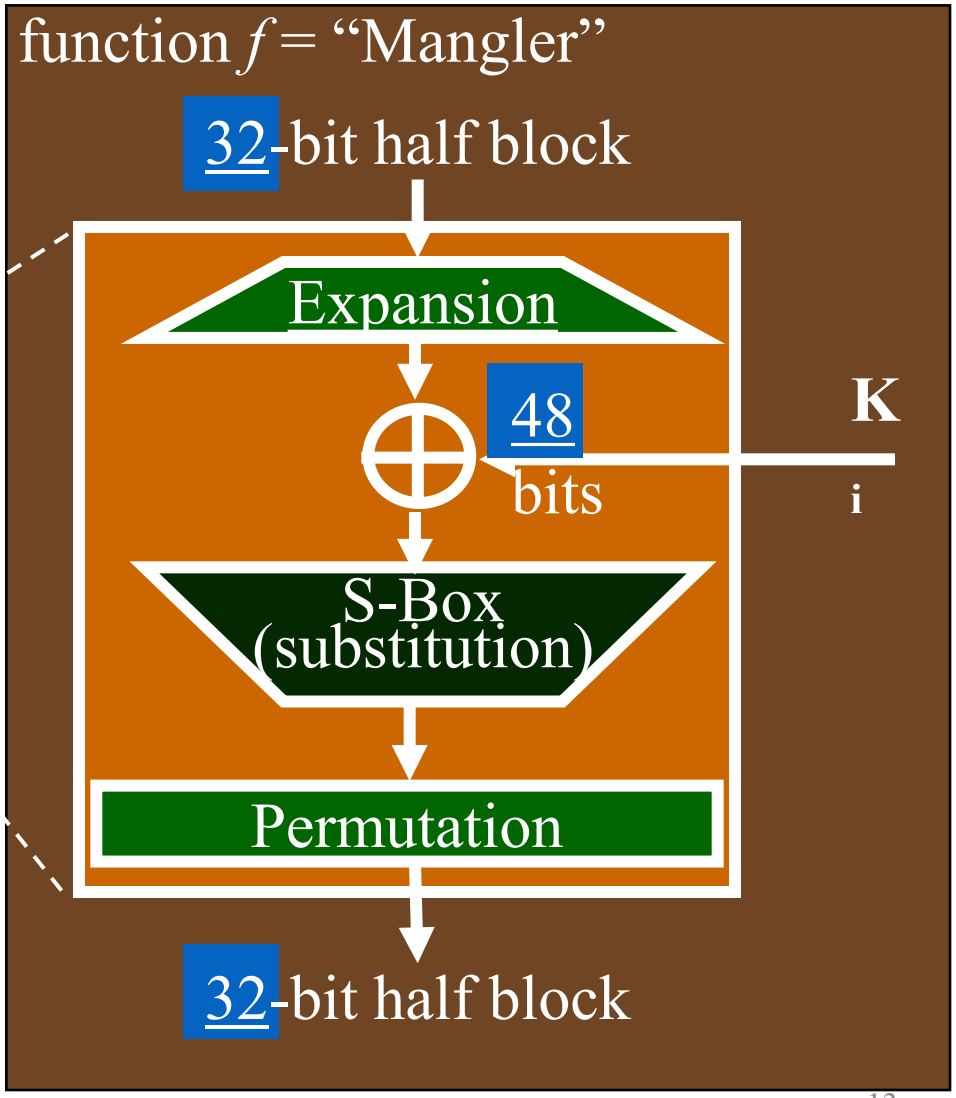




12

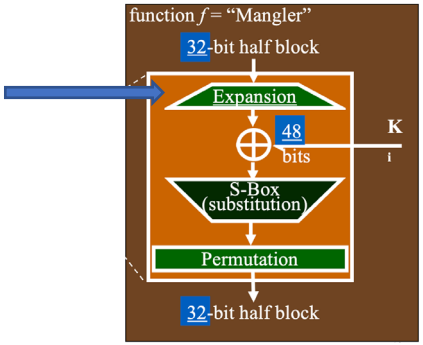
DES ROUND: *F* (MANGLER) FUNCTION





F: EXPANSION FUNCTION

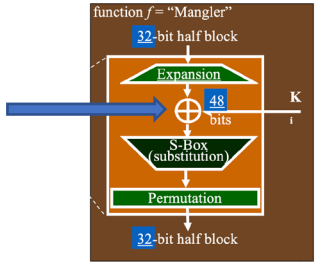
- 32 bits ➔ 48 bits



these bits are repeated

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

XOR *THE KEY*



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

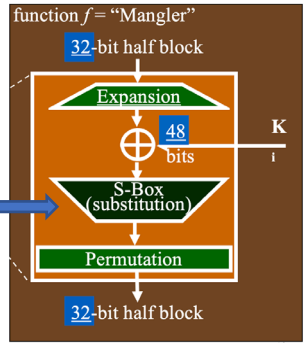
From Expansion function in Mangler

XOR

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

From key generation process

THE RESULT AFTER XOR



The result is a table of bits with 6 columns and 8 rows

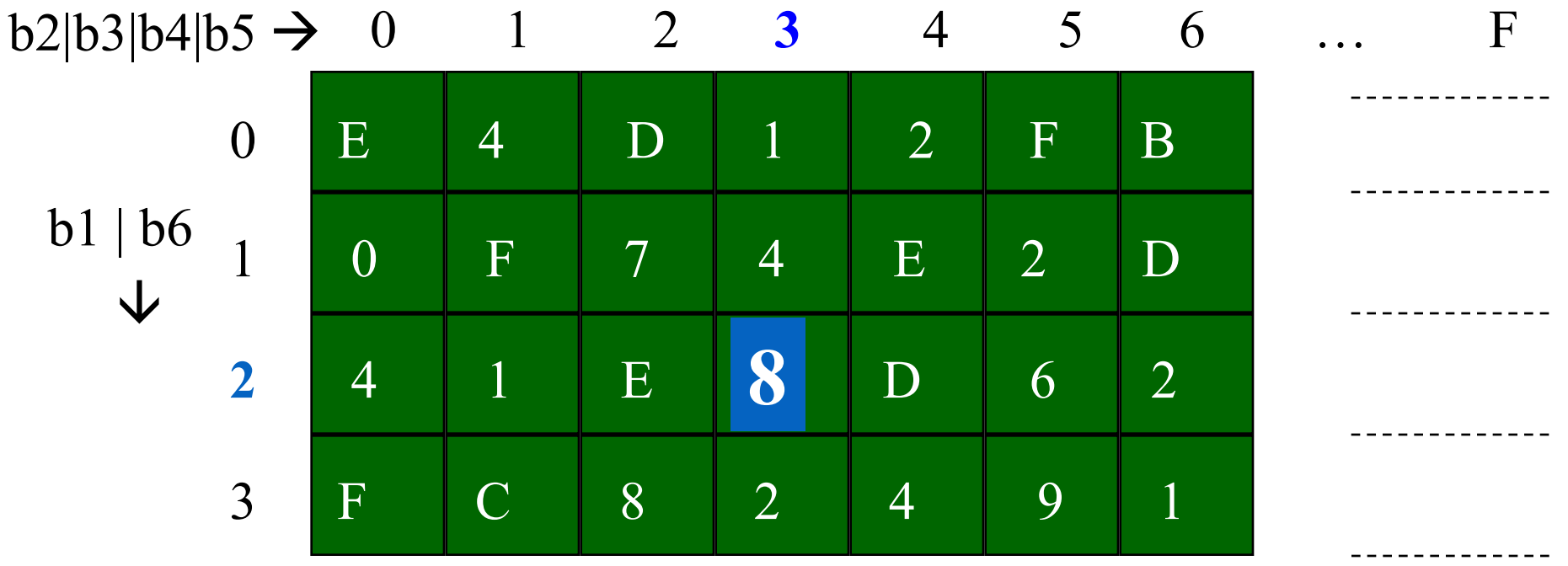
Each row (6 bits) will enter an S-Box S_i

1	0	0	1	1	1
0	0	1	1	0	0
1	1	0	1	0	1
1	1	1	1	1	0
1	1	0	0	0	1
0	0	0	1	1	1
0	1	0	1	0	1
0	1	0	0	1	0

F: S_1 (SUBSTITUTION)

A S-Box is a look-up table (**non-linear part for confusion**)

- each row and column contain different numbers



Example: input= 100110, output= 1000
input = 101101, output =?

4 BIT OUTPUT FOR EACH ROW

48 bits to 32 bits

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

→

1 1 0 1

0 1 1 1

0 0 0 1

1 1 1 0

1 1 0 0

1 0 0 1

0 0 0 1

1 0 0 0

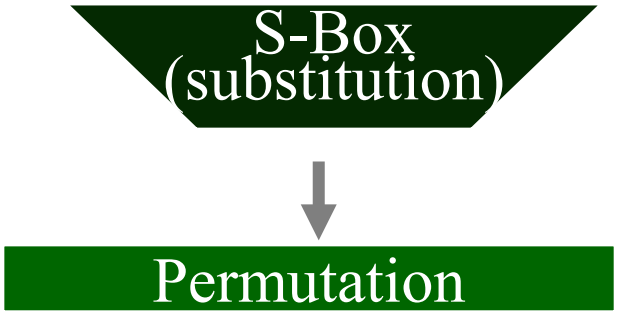
48 bits

32 bits

F: PERMUTATION

- All bits from the S-Box will be again permuted
- 32bits ➔ 32bits

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



DES IMPLEMENTATION

- That's it!
- Operations
  - Permutation
  - Swapping halves
  - Substitution (S-box, table lookup)
  - Bit discard
  - Bit replication
  - Circular shift
  - XOR
- Hard to implement? HW: No, SW: Yes

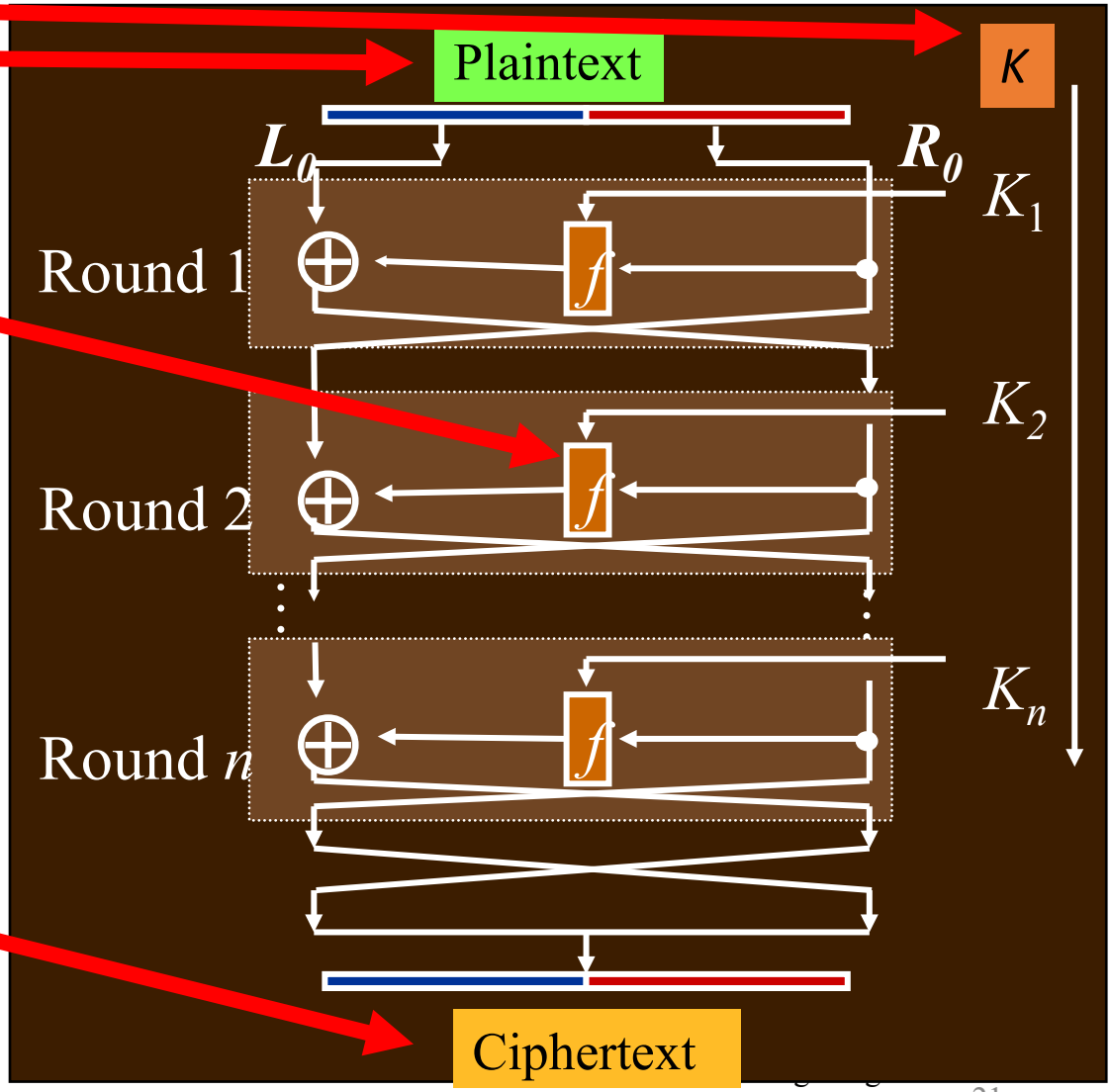
DESIRABLE PROPERTY: AVALANCHE EFFECT

- a small change

f function:
- should achieve avalanche effect
- output bits should be uncorrelated

Number of rounds should be large enough!

Big change!
Any output bit should be inverted (flipped) with probability .5



DES AVALANCHE EFFECT: EXAMPLE

- 2 plaintexts with 1 bit difference:
  0x0000000000000000 and
  0x8000000000000000 (hex format)
  encrypted using the same key:
  0x016B24621C181C32 (hex format)
- Resulting ciphertexts differ in 34 bits (out of 64)
- Similar results when keys differ by 1 bit

EXAMPLE (CONT'D)

- An experiment: number of rounds vs. number of bits difference

Round #	0	1	2	3	4	5	6	7	8
Bits changed	1	6	21	35	39	34	32	31	29

9	10	11	12	13	14	15	16
42	44	32	30	30	26	29	34

DES: KEYS TO AVOID USING

- "Weak keys": These are keys which, after the first key permutation, are:
  - 28 0's followed by 28 0's
  - 28 1's followed by 28 1's
  - 28 0's followed by 28 1's
  - 28 1's followed by 28 0's
- Property of weak keys
  - Easy clue for brute force attacks.
  - Sixteen identical subkeys.
  - Encrypting twice produces the original plaintext.

DES KEY SIZE

- 56 bits is currently too small to resist brute force attacks using readily-available hardware
- Ten years ago it took $250,000 to build a machine that could crack DES in a few hours
- Do it in software?
  - $2^{56} = 72057594037927936$
  - 1 billion every second
  - $\frac{72057594037927936}{10^9} = 2.3 \text{ years}$

TODAY'S CAPABILITY

- Massive Cracking Array
  - One hundred trillion guesses per second

  - $\frac{72057594037927936}{10^{14}} = 721$ seconds!!

WHAT ABOUT 128 BIT KEY

- $\frac{2^{128}}{10^{14}} =$ 100 quadrillion years!!
  - 1 quadriollion is 1,000 trillion, and 1,000,000 billion.

- Life of earth: 4.5 billion years
- Life of universe: 13.7 billion years

CRYPTANALYSIS OF DES

- Differential cryptanalysis exploits differences between encryptions of two different plaintext blocks

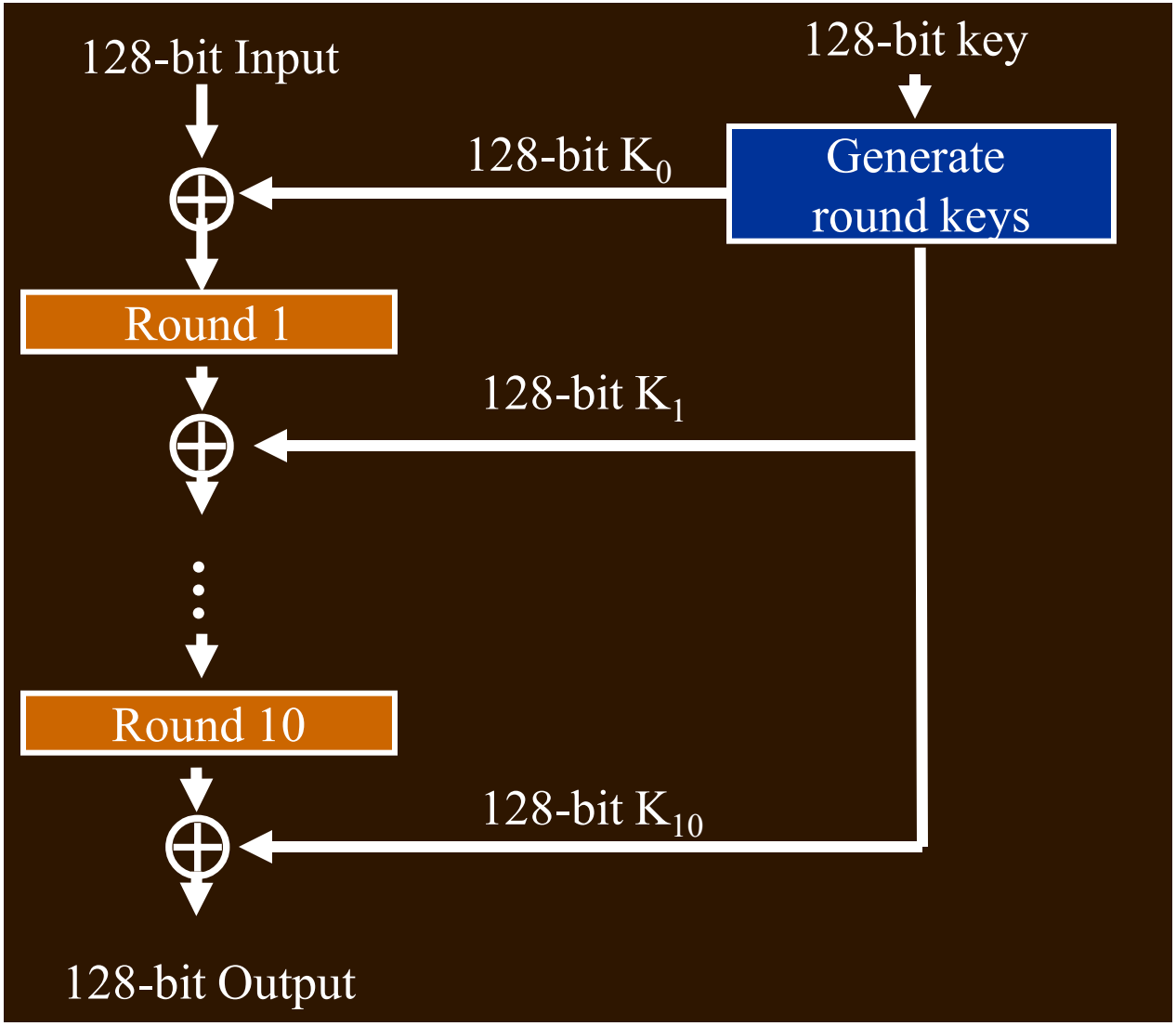
- Linear cryptanalysis requires known plaintext / ciphertext pairs, analyzes relationships to discover key value

- No attacks on DES so far are significantly better than brute force attacks, for comparable cost

ADVANCED ENCRYPTION STANDARD (AES)

- Selected from an open competition, organized by NSA
  - winner: Rijndael algorithm, standardized as AES
  - http://en.wikipedia.org/wiki/Advanced_Encryption_Standard
- Some similarities to DES (rounds, round keys, alternate permutation and substitution)
  - but not a Feistel cipher
- Block size = 128 bits
- Key sizes = 128, 192, or 256 (DES: 56 bits)

AES-128 OVERVIEW



AES ASSESSMENT

- No known successful attacks on full AES
  - Best attacks work on 7–9 rounds
- For brute force attacks, AES-128 will need much more effort than DES

ATTACKS ON AES

- Differential Cryptanalysis: based on how differences in inputs correlate with differences in outputs
- Linear Cryptanalysis: based on correlations between input and output
- Side Channel Attacks: Implementations of the cipher on systems inadvertently leak data
  - Timing Attacks: measure the time it takes to do operations