CS458: Introduction to Information Security

Notes 4: Symmetric Cryptography - Block Cipher

Yousef M. Elmehdwi

Department of Computer Science

Illinois Institute of Technology

yelmehdwi@iit.edu

September 13, 2018

Slides: Modified from Christof Paar and Jan Pelzl

Outline

- Crypto Continued
- Modern Crypto
- Data Encryption Standard (DES)
- Advanced Encryption Standard (AES)

Symmetric Encryption

- Also referred to as:
  - Conventional encryption
  - Secret-key or single-key encryption
- Only alternative before public-key encryption in 1970's
  - Still most widely used alternative
- Has five ingredients:
  - Plaintext
  - Encryption algorithm
  - Secret key
  - Ciphertext
  - Decryption algorithm

Cryptography

- Classified along three independent dimensions:

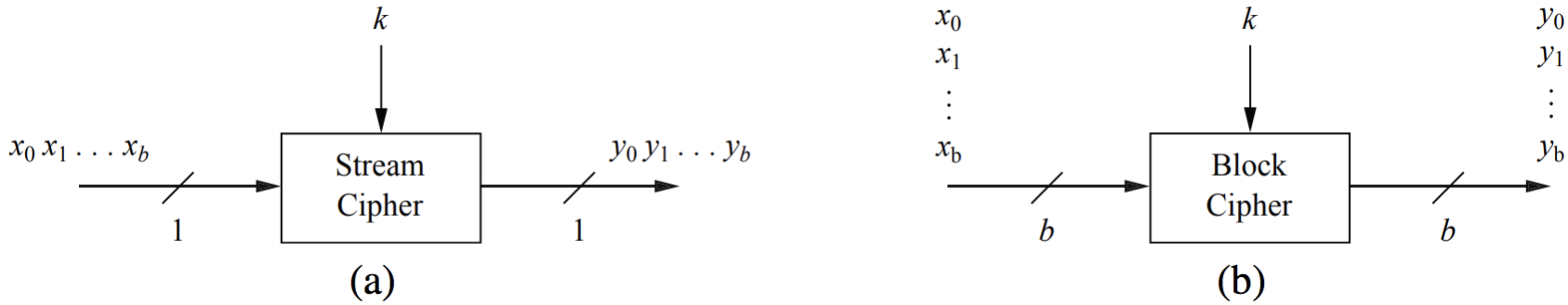
1. The type of operations used for transforming plaintext to ciphertext
   - Substitution: each element in the plaintext is mapped into another element
   - Transposition: elements in plaintext are rearranged
2. The number of keys used
   - Sender and receiver use same key - symmetric
   - Sender and receiver each use a different key - asymmetric
3. The way in which the plaintext is processed
   - Block cipher: processes input one block of elements at a time
   - Stream cipher: processes the input elements continuously

Modern Cryptography

- Symmetric cryptography
  - Block ciphers
    - Operates on blocks of plaintext.
    - DES, AES
  - Stream ciphers
    - Operates on smallest units of the plaintext (bits, letters, etc).
    - One bit of key is XORed with one bit of plaintext to produce ciphertext.
    - Inspired by one-time pad.
- Public key (asymmetric) cryptography.
  - We will cover it next

Stream Cipher vs. Block Cipher

- Operational differences between stream (a) and block (b) ciphers when we want to encrypt b bits at a time, where b is the width of the block cipher.



a) Stream Ciphers
   - Encrypt bits individually
   - Usually small and fast $\rightarrow$ common in embedded devices (e.g., A5/1 for GSM phones)

b) Block Ciphers
   - Always encrypt a full block (several bits)
   - Are common for Internet applications

Symmetric Cryptosystem Components: Building blocks

- Symmetric (one-key) ciphers combine simple ideas, some of which we've already seen:
  - Substitution
  - Transposition
  - Composition
  - Subkey generation
  - Chaining

Substitution: Replacing one letter by another

- The methods discussed so far are based on letter substitution.
- The Caesar cipher shifts the alphabet cyclically.
- This yields 26 possible permutations of the alphabet.
- In general, one can use any permutation of the alphabet, as long as we have a way of computing the permutation and its inverse. This gives us 26! possible permutations.
- Often, permutations are specified by a table called an S-box.

Transposition: Rearranging letters

- Another technique is to rearrange the letters of the plaintext.

  `this message is encoded with a transposition cipher`

  1. Pick a number: 9.
  2. Write the message in a 9-column matrix (ignoring spaces):

     thi sme ssa
     gei sen cod
     edw ith atr
     ans pos iti
     onc iph er

3. Read it out by columns[1].

   `tgeao hednn iiwsc ssipi metop enhsh scaie sottr adri`

[1] Spaces are not part of ciphertext

Composition: Building new ciphers from old

- Let (E', D') and (E'', D'') be ciphers.
- Their composition is the cipher (E, D) with keys of the form $k = (k'', k')$, where

$$E_{(k'',k')}(m) = E''_{k''}(E'_{k'}(m))$$
$$D_{(k'',k')}(c) = D'_{k'}(D''_{k''}(c))$$

Subkey generation

- When ciphers are composed, each component cipher needs a key called a subkey. Together, those subkeys can get rather large and unwieldy.
- For practical reasons, the subkeys are themselves often generated by a deterministic process dependent on a master key, which is the user key of the resulting cryptosystem.

Chaining modes

- A chaining mode describes how to employ the cipher on a sequence of blocks.
- One obvious way is to repeatedly use the cipher with the same key on each successive block. This is called Electronic Code Book (ECB) mode.
  - i.e., the message is divided into blocks, and each block is encrypted separately.
- We can improve on this by generating a different subkey for each block.
- For example, successive subkeys might depend on the block number, as in simple stream ciphers, or also on previous plaintext and/or ciphertext

- Data Encryption Standard (DES)

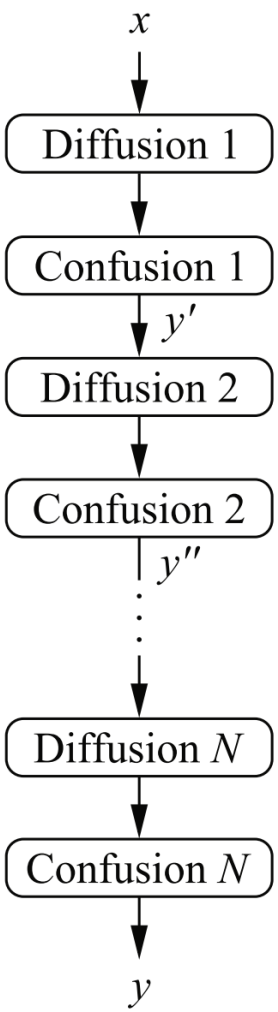
Data encryption standard (DES)

- The Data Encryption Standard is a block cipher that operates on 64-bit blocks and uses a 56-bit key.
- Developed by IBM based on the cipher Lucifer under influence of the National Security Agency (NSA), the design criteria for DES have not been published
- It became an official Federal Information Processing Standard (FIPS) in 1976. It was officially withdrawn as a standard in 2005 after it became widely acknowledged that the key length was too short and it was subject to brute force attack.
- Nevertheless, triple DES (with a 112-bit key) is approved through the year 2030 for sensitive government information.
- The Advanced Encryption Standard (AES), based on the Rijndael algorithm, became an official standard in 2001. AES supports key sizes of 128, 192, and 256 bits and works on 128-bit blocks.

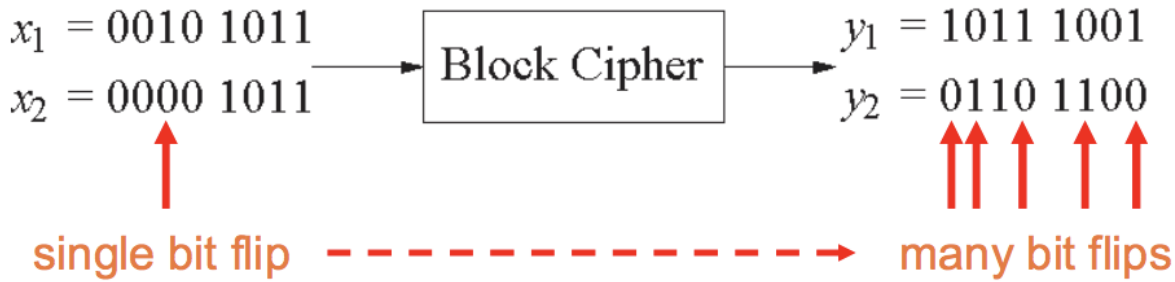
Block Cipher Primitives: Confusion and Diffusion

- Claude Shannon: There are two primitive operations with which strong encryption algorithms can be built:
  1. Confusion:
     - An encryption operation where the relationship between key and ciphertext is obscured.
     - Today, a common element for achieving confusion is substitution, which is found in both AES and DES.
  2. Diffusion
     - An encryption operation where the influence of one plaintext symbol is spread over many ciphertext symbols with the goal of hiding statistical properties of the plaintext.
     - A simple diffusion element is the bit permutation, which is frequently used within DES.
- Both operations by themselves cannot provide security. The idea is to concatenate confusion and diffusion elements to build so called product ciphers.

Product Ciphers



- Most of today's block ciphers are product ciphers as they consist of rounds which are applied repeatedly to the data.
- Can reach excellent diffusion: changing of one bit of plaintext results on average in the change of half the output bits.
- Example:



Feistel networks

- Many symmetric block encryption algorithms, including DES, have a structure first described by Horst Feistel of IBM in 1973
- DES is based on a Feistel network.
- It consists of some number of stages.
  - Each stage i maps a pair of $w\text{-}bit$ words (L_i, R_i) to a new pair (L_{i+1}, R_{i+1}). ($w = 32$ in case of DES.)
  - By applying the stages in sequence, a $t\text{-}stage$ network maps (L_0, R_0) to (L_t, R_t).
  - (L_0, R_0) is the plaintext, and (L_t, R_t) is the corresponding ciphertext.

Feistel networks

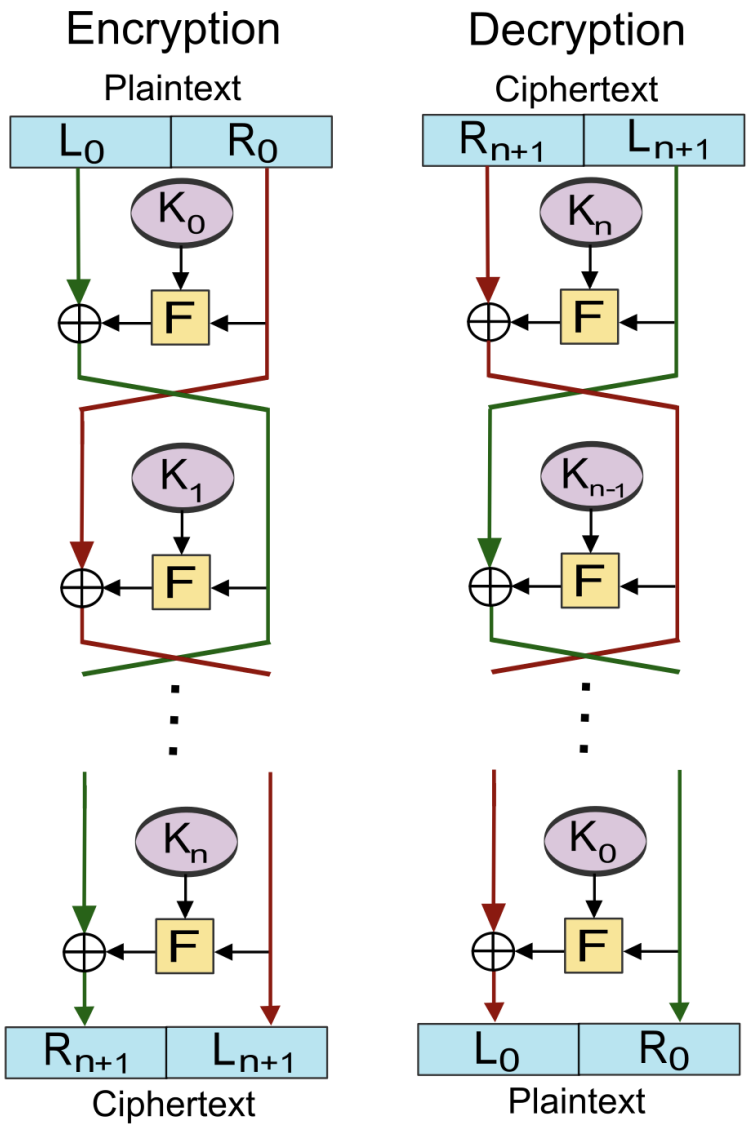


Image credit: Wikipedia: Feistel cipher

One stage

- Each stage works as follows:

$$L_{i+1} = R_i \tag{1}$$

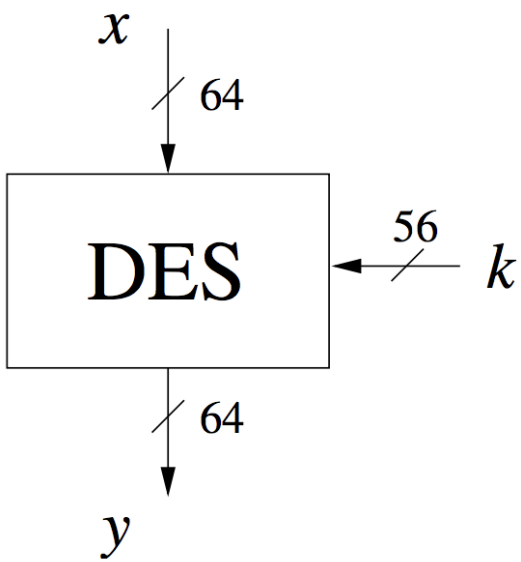
$$R_{i+1} = L_i \oplus f(R_i, K_i) \tag{2}$$

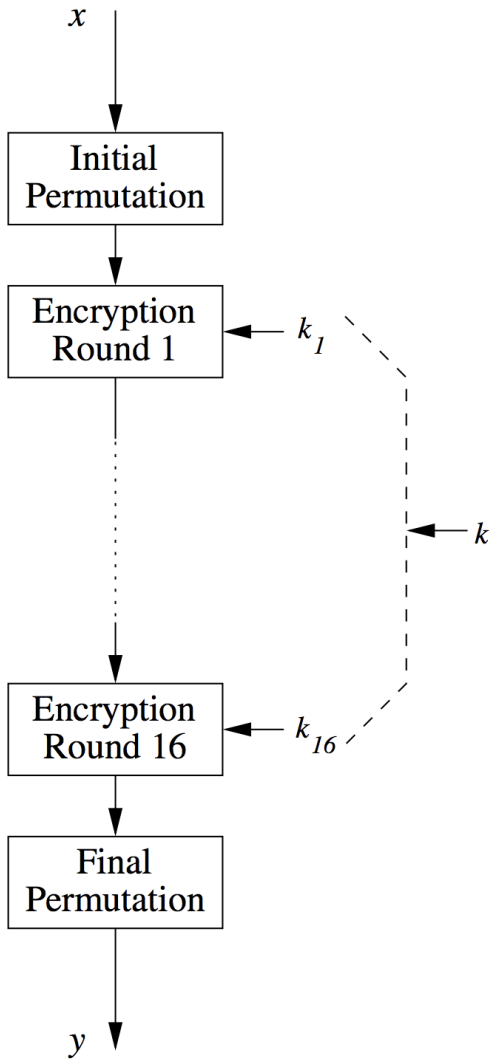
- Here, K_i is a subkey, which is generally derived in some systematic way from the master key K, and f is the scrambling function (shown as F in the diagram).
- The inversion problem is to find (L_i, R_i) given (L_{i+1}, R_{i+1}).
  - Equation 1 gives us R_i.
  - Knowing R_i and K_i, we can compute $f(R_i, K_i)$.
  - We can then solve equation 2 to get $L_i = R_{i+1} \oplus f(R_i, K_i)$

Properties of Feistel networks

- The security of a Feistel-based code lies in the construction of the scrambling function f and in the method for producing the subkeys K_i.
- The invertibility follows just from properties of $\oplus$ (exclusive-or).

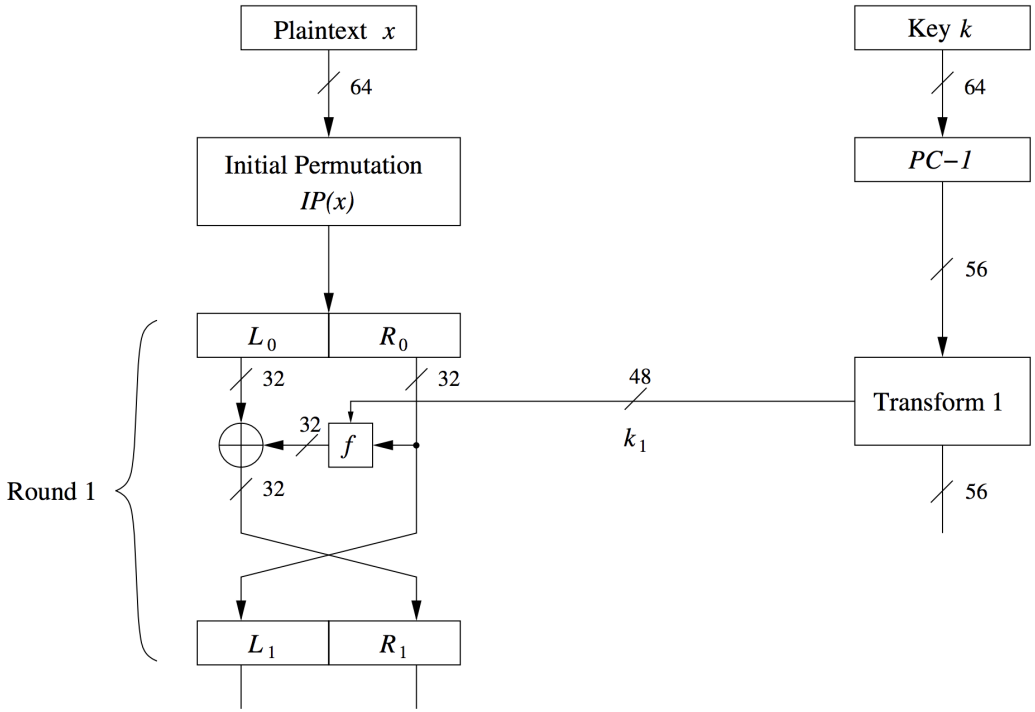
Overview of the DES Algorithm





- Encrypts blocks of size 64 bits.
- Uses a key of size 56 bits.
- Symmetric cipher: uses same key for encryption and decryption
- Uses 16 rounds which all perform the identical operation

DES Feistel network



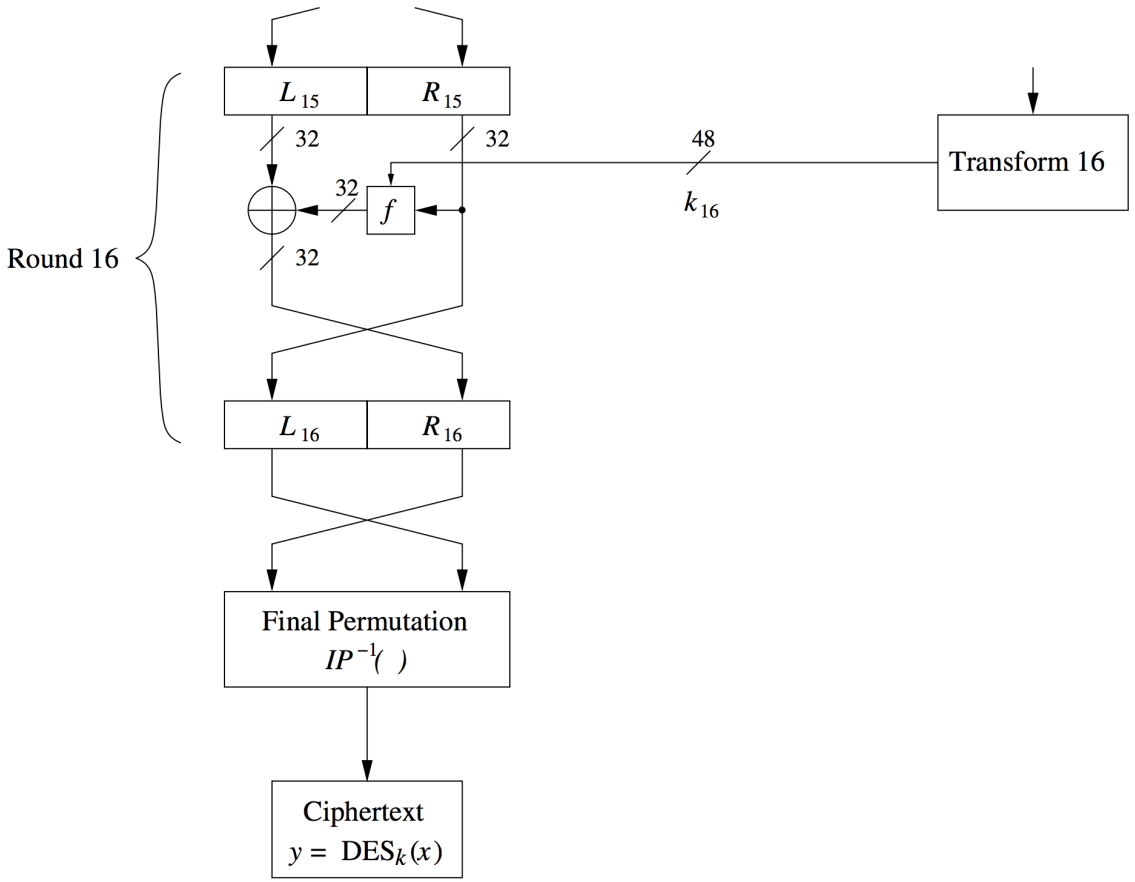
- Bitwise initial permutation, then 16 rounds

1. Plaintext is split into 32-bit halves L_i and R_i
2. R_i is fed into function f, the output of which is then XORed with L_i
3. Left and right half are swapped

- Rounds can be expressed as:
  - $L_i = R_{i-1}$
  - $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$

DES Feistel network

- L and R swapped again at the end of the cipher, i.e., after round 16 followed by a final permutation

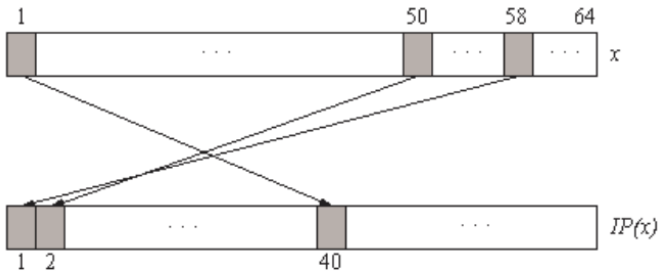


Internal Structure of DES: Initial and Final Permutation

- Bitwise Permutations.
- Inverse operations.
- Described by tables IP and IP^{-1}.

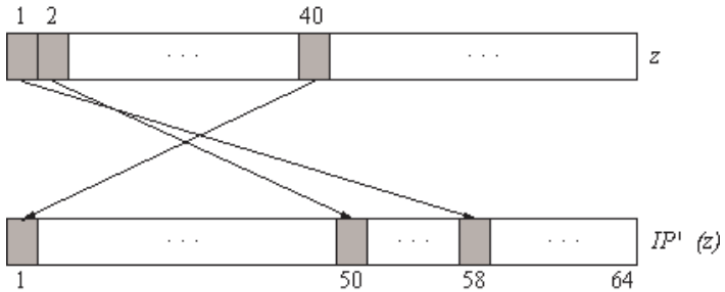
Initial Permutation

IP							
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

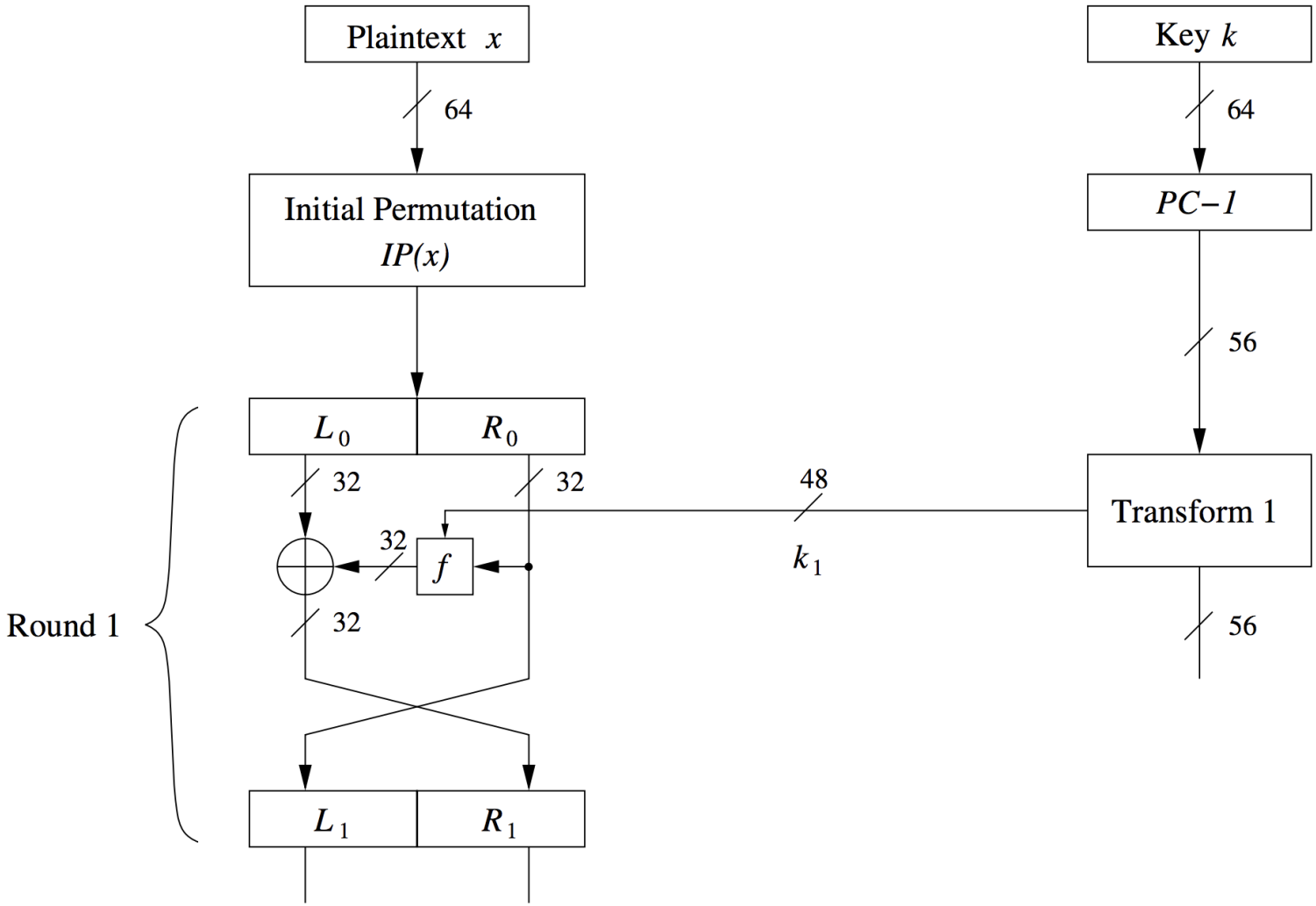


Final Permutation

IP^{-1}							
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

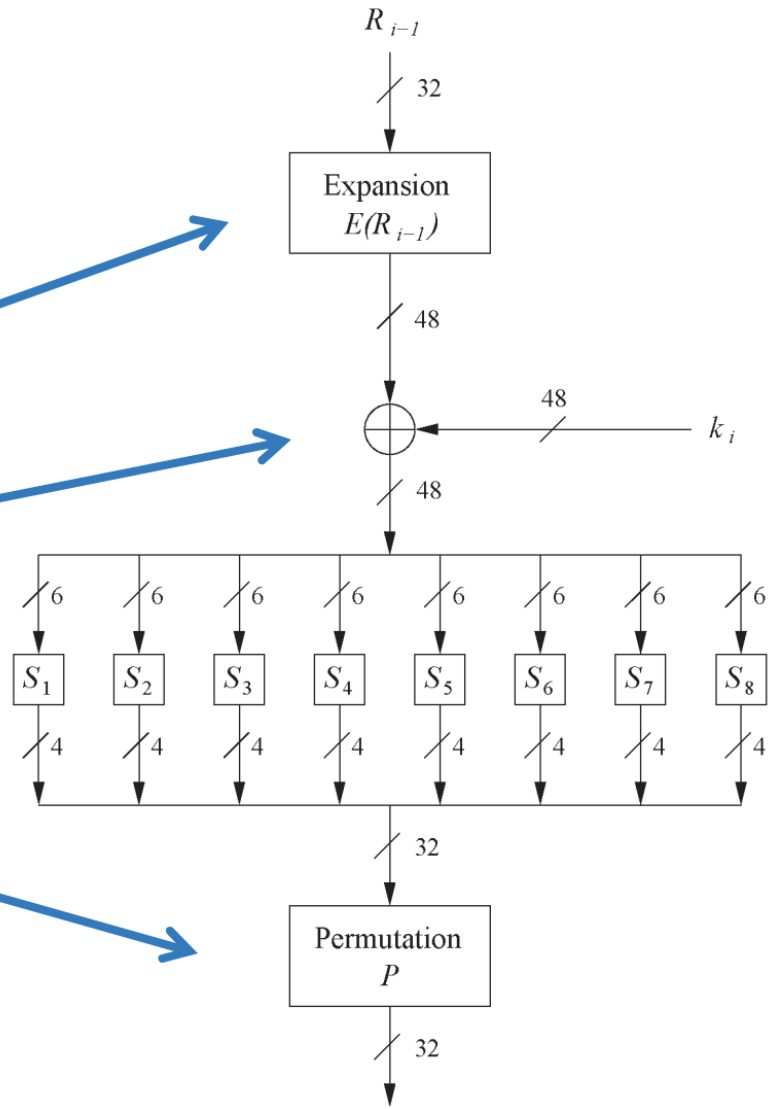


DES Feistel network



Internal Structure of DES: The f-Function

- **main operation of DES**
- f-Function inputs:
  R_{i-1} and round key k_i
- **4 Steps**:
  1. Expansion E
  2. XOR with round key
  3. S-box substitution
  4. Permutation

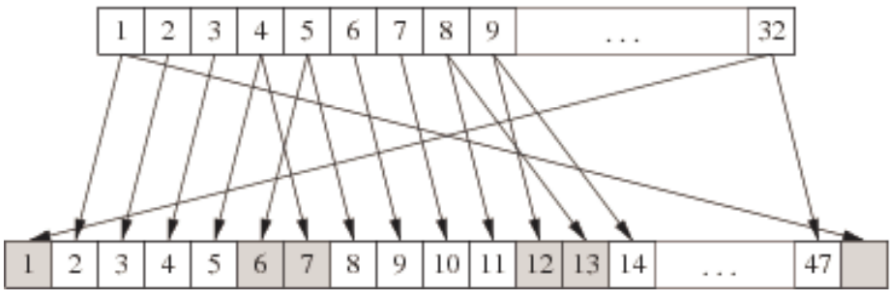


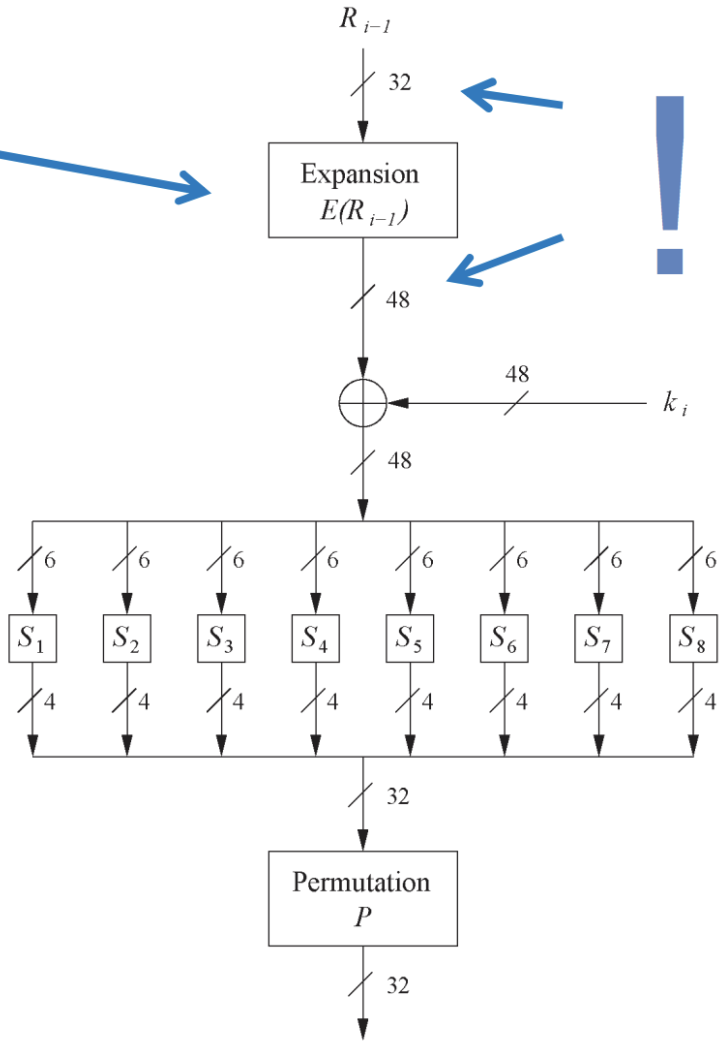
The f-Function: The Expansion Function E

1. **Expansion *E***

- **main purpose:**
  increases diffusion

E					
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

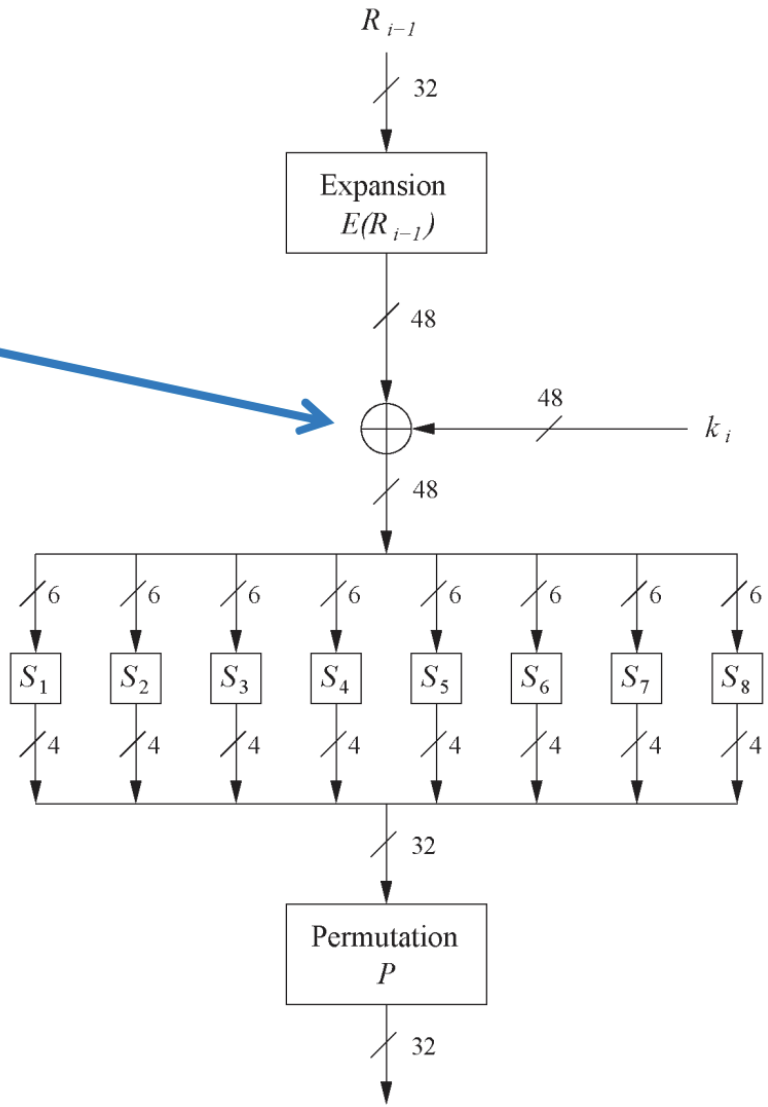




The f-Function: Add Round Key

2. XOR Round Key

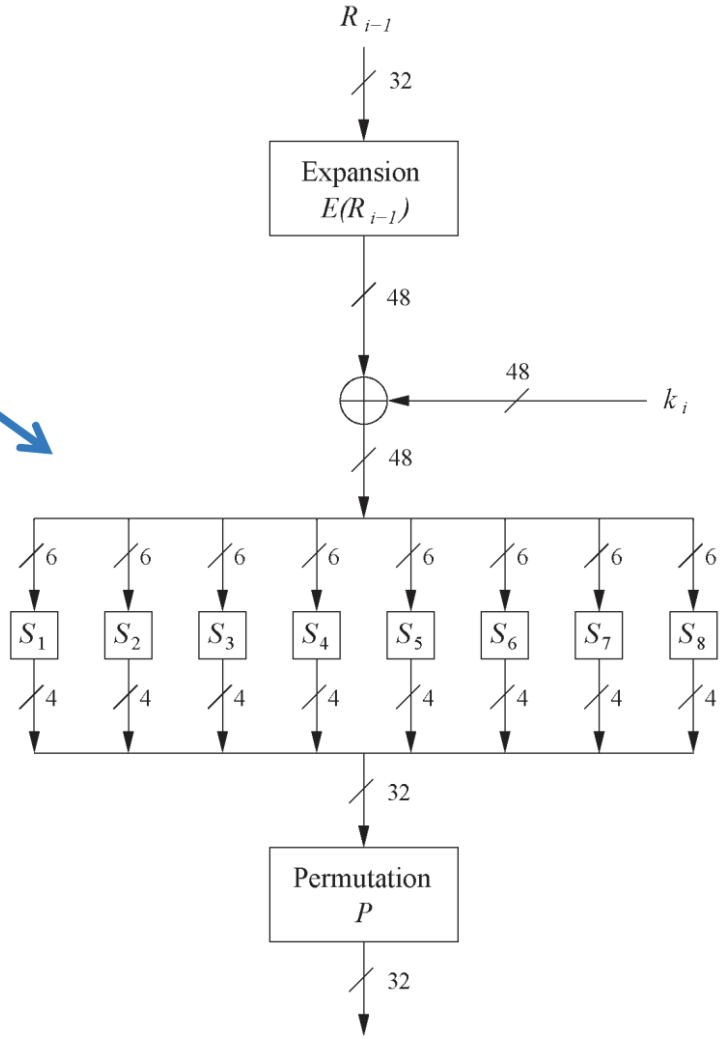
- **Bitwise XOR of the round key and the output of the expansion function *E***
- **Round keys are derived from the main key in the DES keyschedule (in a few slides)**

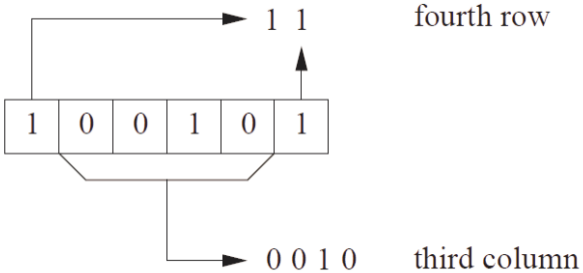


The f-Function: The DES S-Boxes

3. S-Box substitution

- Eight substitution tables.
- 6 bits of input, 4 bits of output.
- Non-linear and resistant to differential cryptanalysis.
- Crucial element for DES security!





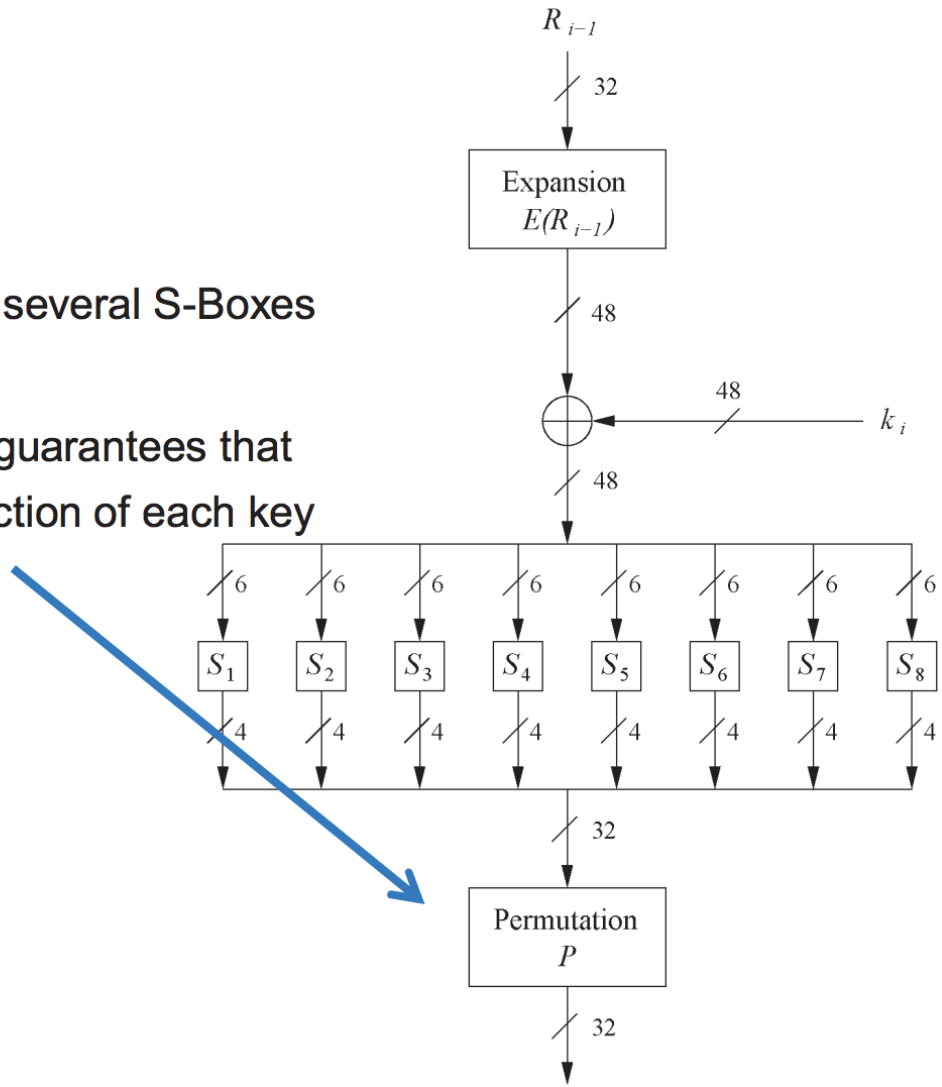
S_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	01	10	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

The f-Function: The Permutation P

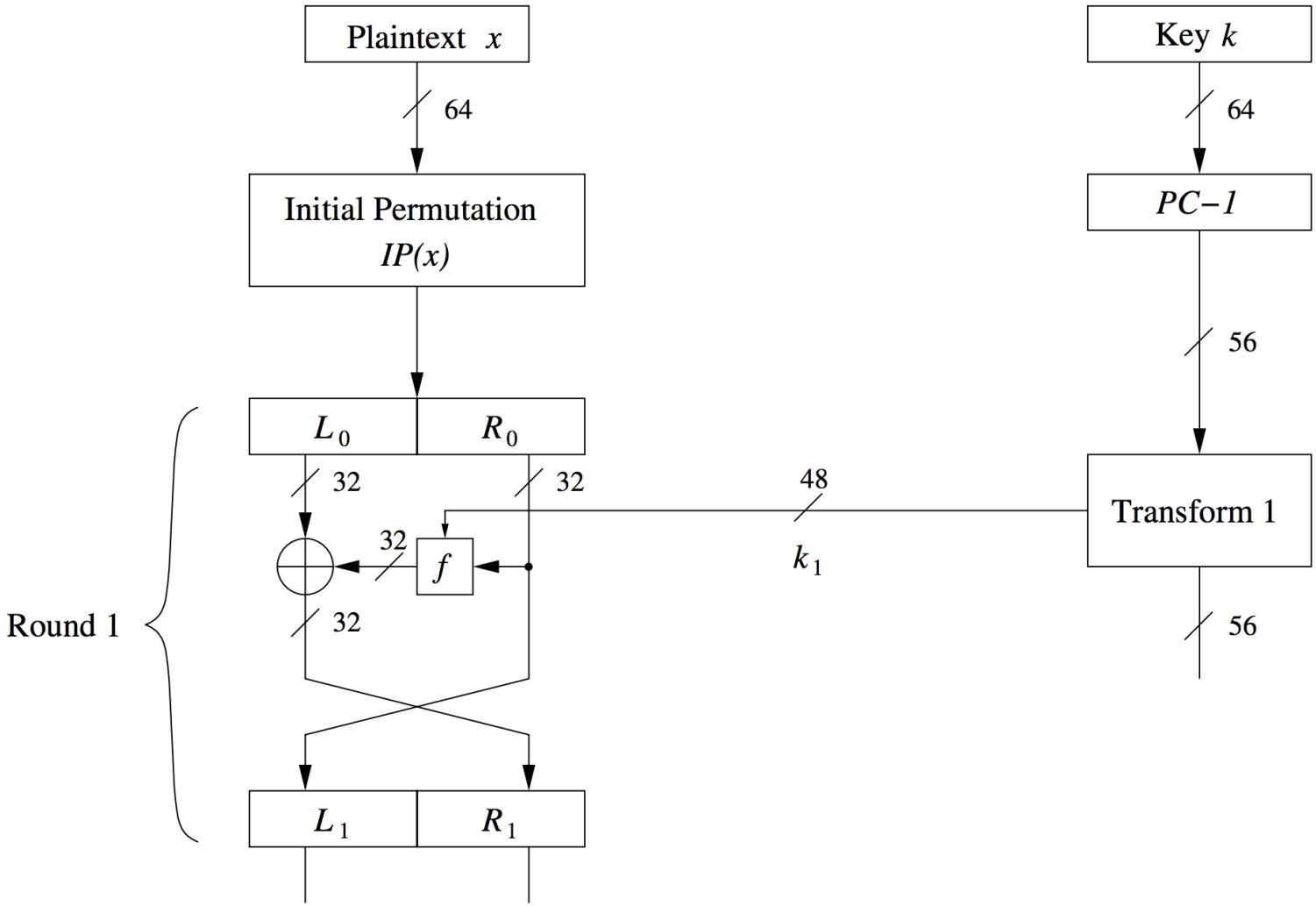
4. Permutation P

- Bitwise permutation.
- Introduces diffusion.
- Output bits of one S-Box effect several S-Boxes in next round
- Diffusion by E, S-Boxes and P guarantees that after Round 5 every bit is a function of each key bit and each plaintext bit.

P							
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

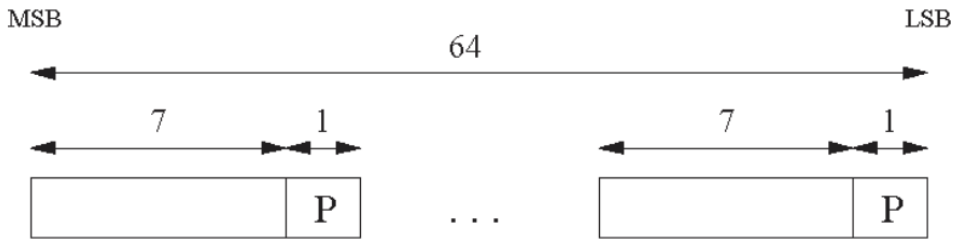


DES Feistel network



Internal Structure of DES: Key Schedule (1)

- Derives 16 round keys (or *subkeys*) k_i of 48 bits each from the original 56 bit key.
- The input key size of the DES is 64 bit: **56 bit key** and 8 bit parity:



P = parity bit

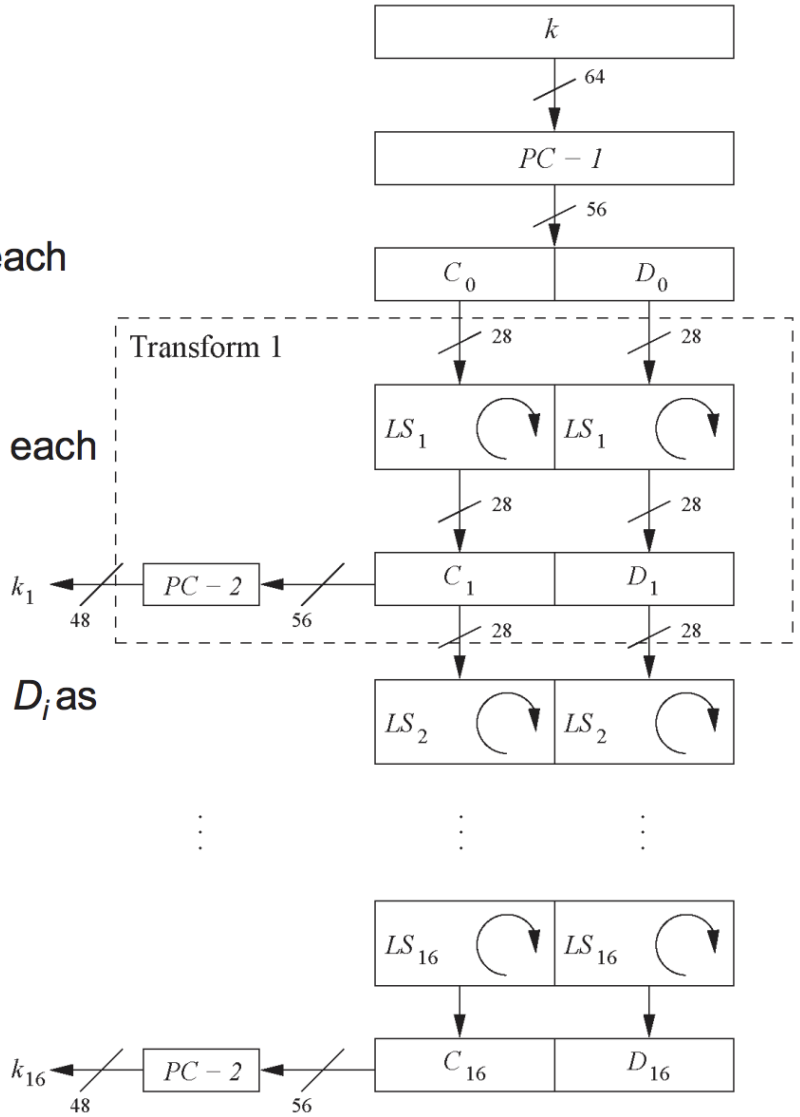
- **Parity bits are removed** in a first **permuted choice** *PC-1*: (note that the bits 8, 16, 24, 32, 40, 48, 56 and 64 are not used at all)

$PC-1$							
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

Internal Structure of DES: Key Schedule (2)

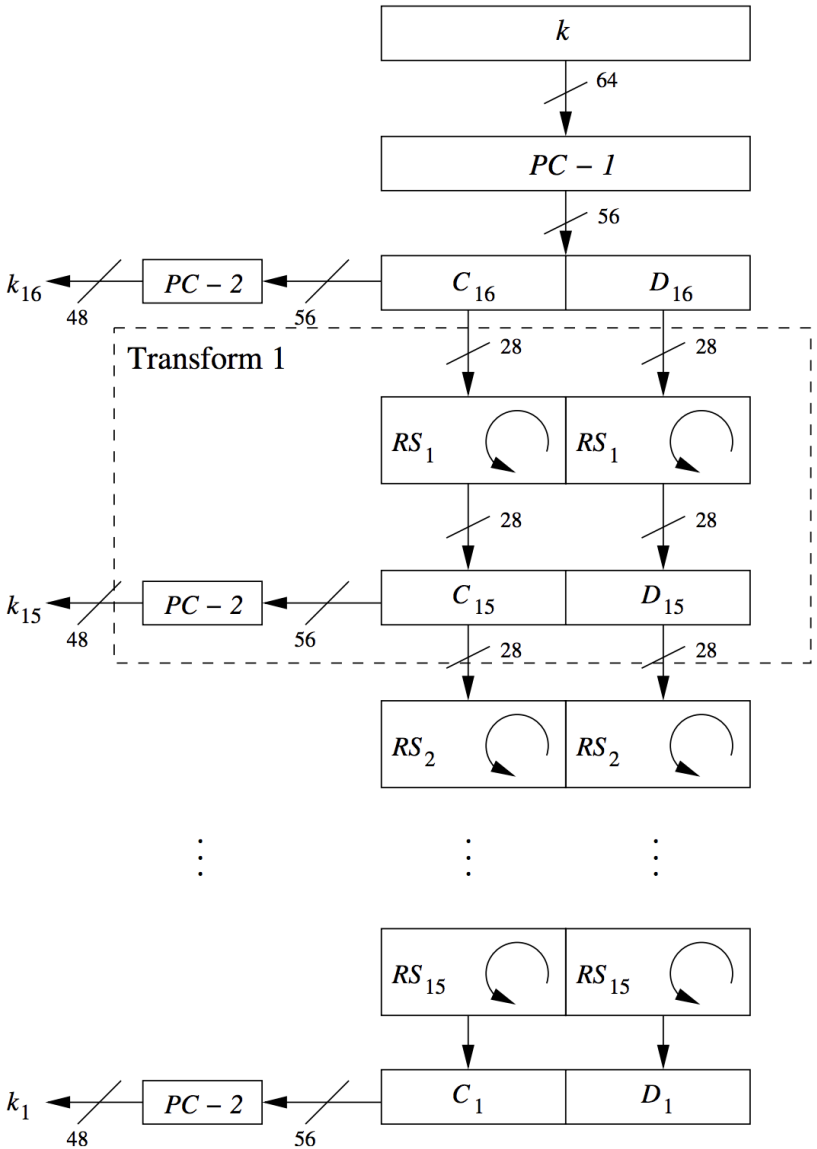
- Split key into 28-bit halves C_0 and D_0.
- In **rounds i = 1, 2, 9 ,16,** the two halves are each rotated left by **one bit**.
- In **all other rounds** where the two halves are each rotated left by **two bits**.
- *In each round i permuted choice* ***PC-2*** selects a permuted subset of 48 bits of C_i and D_i as round key k_i,

PC – 2							
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



- **Note:** The total number of rotations:
  4 x 1 + 12 x 2 = 28 $\Rightarrow D_0 = D_{16}$ and $C_0 = C_{16}$!

Decryption


k
64
PC – 1
56
k_{16}
48
PC – 2
56
C_{16}
D_{16}
Transform 1
28
28
RS_1
RS_1
28
28
k_{15}
48
PC – 2
56
C_{15}
D_{15}
28
28
RS_2
RS_2
RS_{15}
RS_{15}
k_1
48
PC – 2
56
C_1
D_1

Decryption

- **Decryption**
- In Feistel ciphers only the key schedule has to be modified for decryption.
- Generate the same 16 round keys in reverse order.
- **Reversed key schedule:**
- As $D_0 = D_{16}$ and $C_0 = C_{16}$ the first round key can be generated by applying PC-2 right after PC-1 (no rotation here!).
- All other rotations of C and D can be reversed to reproduce the other round keys resulting in:
  - No rotation in round 1.
  - One bit rotation to the right in rounds 2, 9 and 16.
  - Two bit rotations to the right in all other rounds

DES Last Word (Almost)

- An initial permutation before round 1
- Halves are swapped after last round
- A final permutation (inverse of initial perm) applied to (R_{16}, L_{16})
- None of this serves security purpose

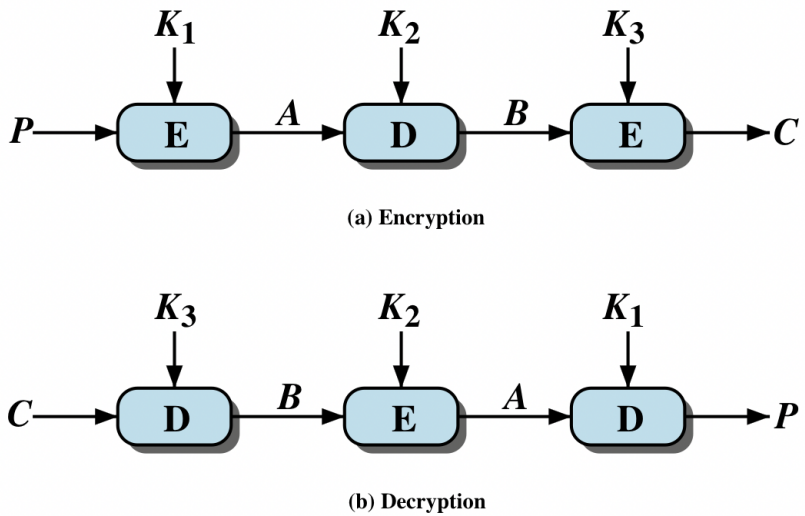
Security of DES

- After proposal of DES two major criticisms arose:
  1. Key space is too small (2^{56} keys)
  2. S-box design criteria have been kept secret: Are there any hidden analytical attacks (backdoors), only known to the NSA?
- **Analytical Attacks**: DES is highly resistant to both differential and linear cryptanalysis, which have been published years later than the DES. This means IBM and NSA had been aware of these attacks for 15 years! So far there is no known analytical attack which breaks DES in realistic scenarios
- A final permutation (inverse of initial perm) applied to (R_{16}, L_{16})
- **Exhaustive key search**: For a given pair of plaintext-ciphertext (x, y) test all 2^{56} keys until the condition $DES_k^{-1}(x)=y$ is fulfilled.
  - $\Rightarrow$ Relatively easy given today's computer technology!

Triple DES - 3DES

- 3DES uses three keys and three executions of the DES algorithm.
- The function follows an encrypt-decrypt-encrypt (EDE) sequence:

$$c = E_{K_3}(D_{K_2}(E_{K_1}(p)))$$



- Decryption is simply the same operation with the keys reversed:

$$p = D_{K_1}(E_{K_2}(D_{K_3}(c)))$$

Triple DES - 3DES

- There is no cryptographic significance to the use of decryption for the second stage of 3DES encryption. Its only advantage is that it allows users of 3DES to decrypt data encrypted by users of the older single DES:
- With three distinct keys, 3DES has an effective key length of $168\ bits$. FIPS 46-3 also allows for the use of two keys, with $K_1 = K_3$; this provides for a key length of $112\ bits$.

Lessons Learned

- DES was the dominant symmetric encryption algorithm from the mid-1970s to the mid-1990s. Since 56-bit keys are no longer secure, the Advanced Encryption Standard (AES) was created.
- Standard DES with 56-bit key length can be broken relatively easily nowadays through an exhaustive key search.
- DES is quite robust against known analytical attacks: In practice it is very difficult to break the cipher with differential or linear cryptanalysis.
- By encrypting with DES three times in a row, triple DES (3DES) is created, against which no practical attack is currently known.
- The "default" symmetric cipher is nowadays often AES. In addition, the other four AES finalist ciphers all seem very secure and efficient.

- Advanced Encryption Standard (AES)

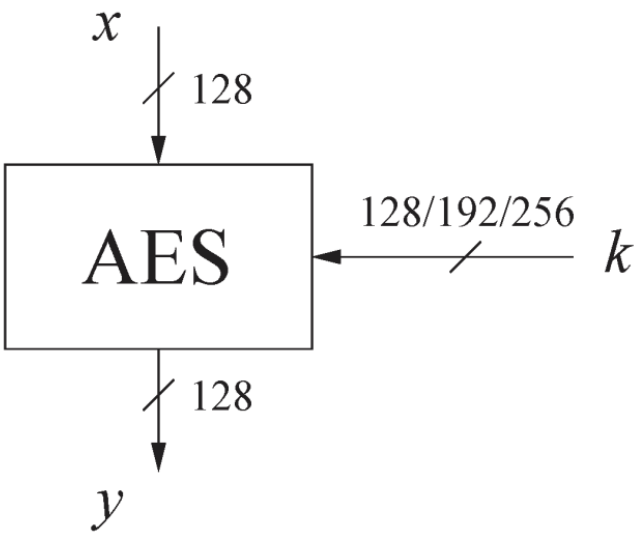
Some Basic Facts

- AES is the most widely used symmetric cipher today
- The algorithm for AES was chosen by the US National Institute of Standards and Technology (NIST) in a multi-year selection process
- The requirements for all AES candidate submissions were:
  - Block cipher with 128-bit block size
  - Three supported key lengths: 128, 192 and 256 bit
  - Security relative to other submitted algorithms
  - Efficiency in software and hardware

Chronology of the AES Selection

- The need for a new block cipher announced by NIST in January, 1997
- 15 candidates algorithms accepted in August, 1998
- 5 finalists announced in August, 1999:
  - Mars - IBM Corporation
  - RC6 - RSA Laboratories
  - Rijndael - J. Daemen & V. Rijmen
  - Serpent - Eli Biham et al.
  - Twofish - B. Schneier et al.
- In October 2000, Rijndael was chosen as the AES
- AES was formally approved as a US federal standard in November 2001

AES: Overview

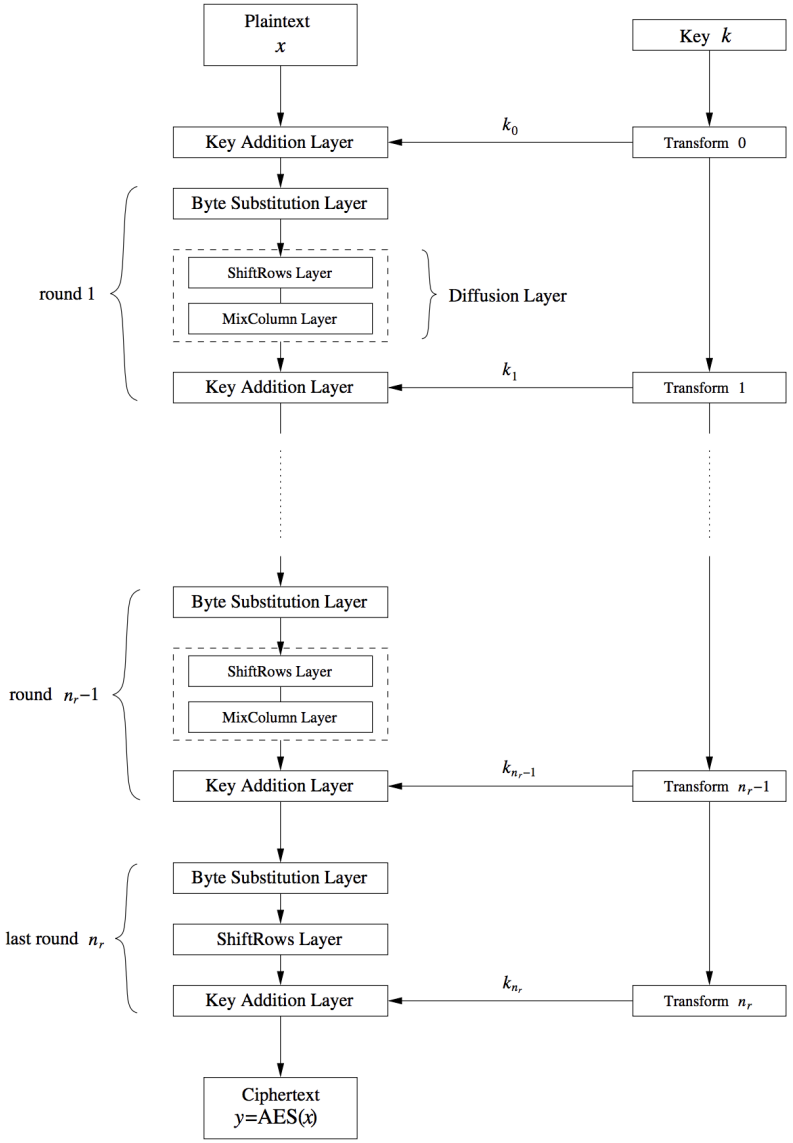


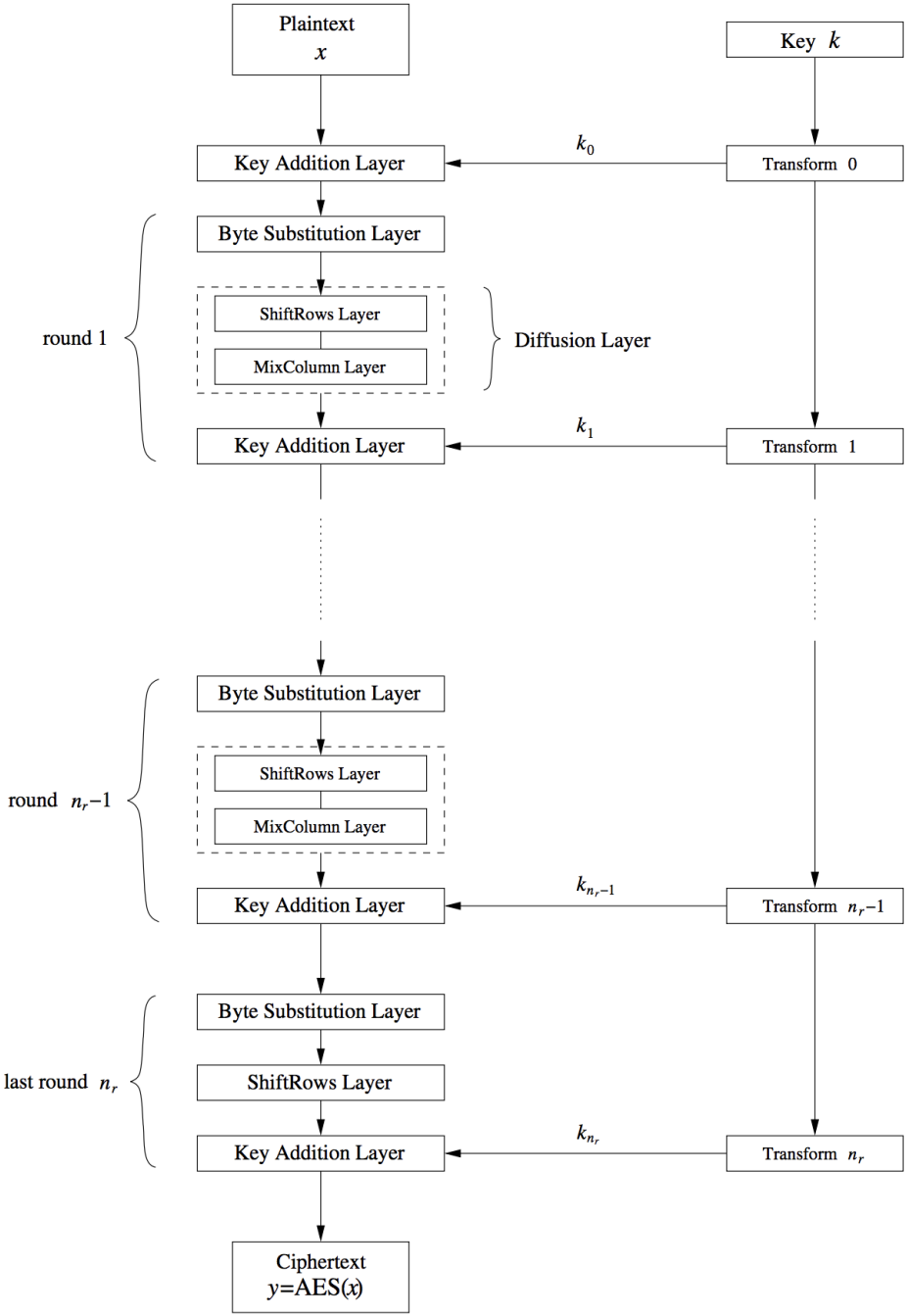
- The number of rounds depends on the chosen key length:

Key length (bits)	Number of rounds
128	10
192	12
256	14

Overview of the DES Algorithm

- Iterated cipher with 10/12/14 rounds
- Each round consists of "Layers"




Plaintext
x
Key k
k_0
Key Addition Layer
Transform 0
Byte Substitution Layer
round 1
ShiftRows Layer
MixColumn Layer
Diffusion Layer
k_1
Key Addition Layer
Transform 1
Byte Substitution Layer
round n_r-1
ShiftRows Layer
MixColumn Layer
k_{n_r-1}
Key Addition Layer
Transform n_r-1
Byte Substitution Layer
last round n_r
ShiftRows Layer
k_{n_r}
Key Addition Layer
Transform n_r
Ciphertext
y=AES(x)

Internal structure of AES

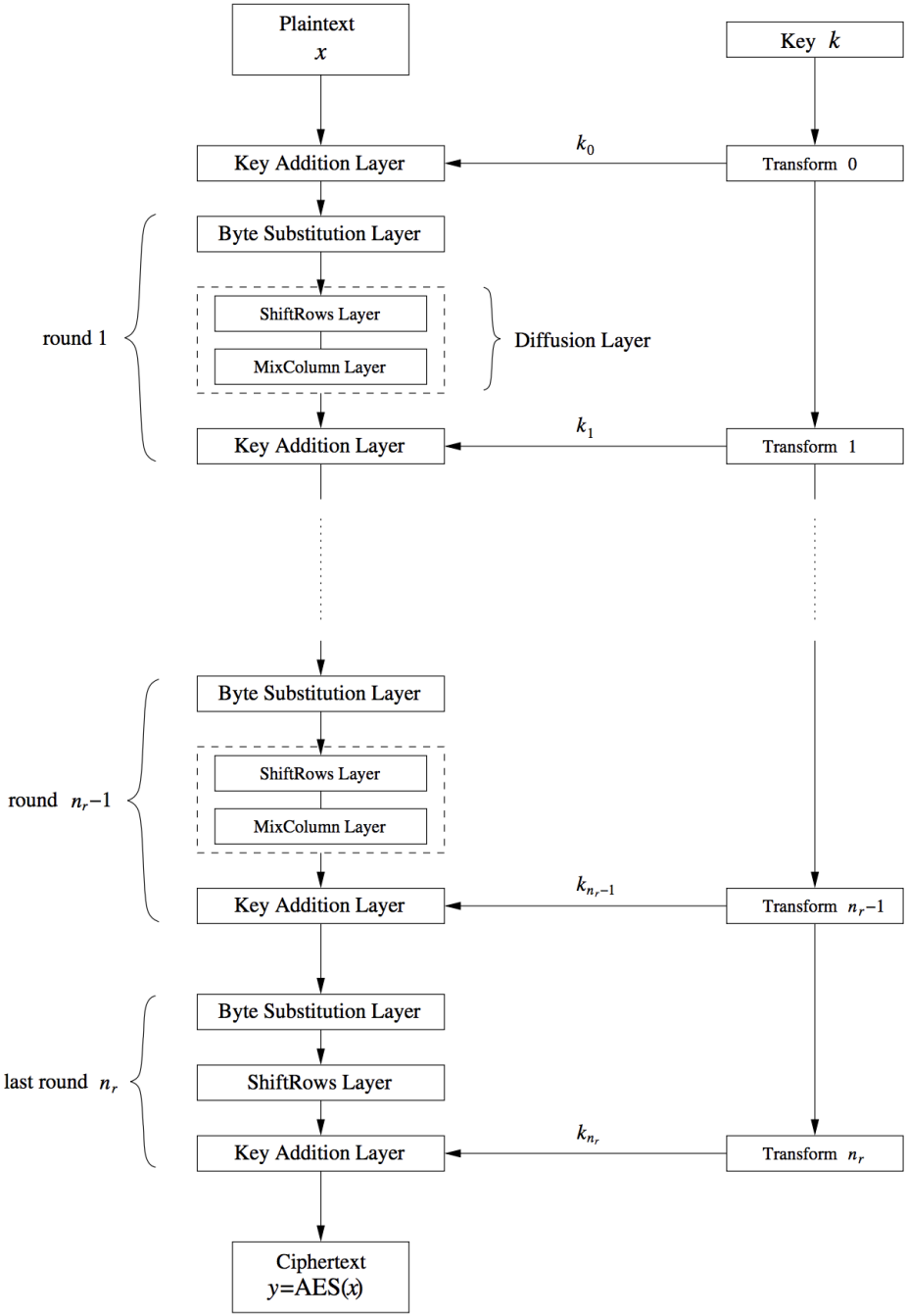
- Byte Substitution layer
- Diffusion layer
- Key Addition layer
- Key schedule

Internal structure of AES

- AES is a byte-oriented cipher
- The state A (i.e., the 128-bit data path) can be arranged in a 4×4 matrix:

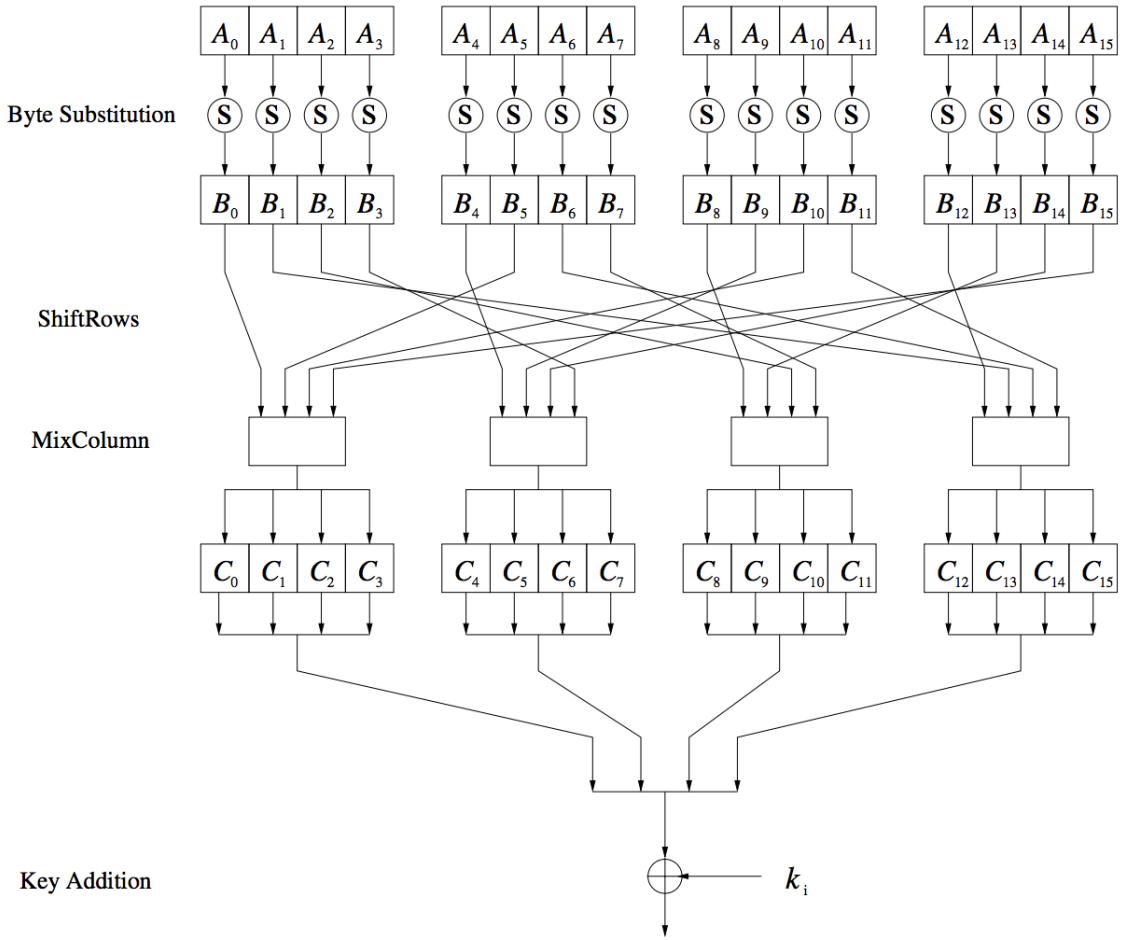
A_0	A_4	A_8	A_{12}
A_1	A_5	A_9	A_{13}
A_2	A_6	A_{10}	A_{14}
A_3	A_7	A_{11}	A_{15}

- with $A_0, \ldots, A_{15}$ denoting the *16-byte* input of AES


Plaintext
x
Key k
k_0
Key Addition Layer
Transform 0
Byte Substitution Layer
round 1
ShiftRows Layer
MixColumn Layer
Diffusion Layer
k_1
Key Addition Layer
Transform 1
Byte Substitution Layer
round n_r-1
ShiftRows Layer
MixColumn Layer
k_{n_r-1}
Key Addition Layer
Transform n_r-1
Byte Substitution Layer
last round n_r
ShiftRows Layer
k_{n_r}
Key Addition Layer
Transform n_r
Ciphertext
y=AES(x)

Internal structure of AES

- Round function for rounds $1, 2, \ldots, n_{r-1}$:



- Note: In the last round, the MixColumn transformation is omitted

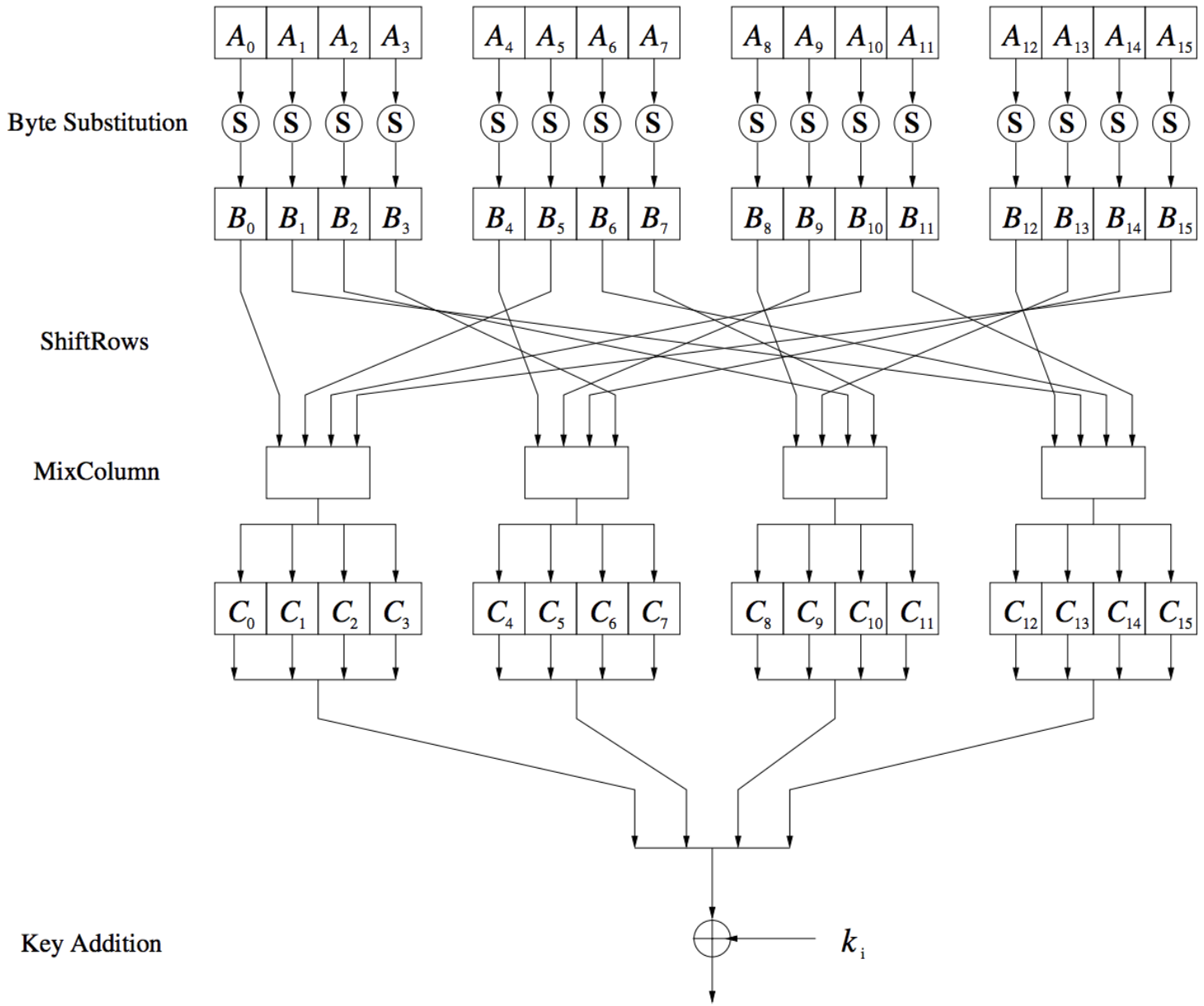
Byte Substitution Layer

- The Byte Substitution layer consists of 16 S-Boxes with the following properties:
  - The S-Boxes are
    - **identical**
    - the only **nonlinear** elements of AES,
      i.e., $ByteSub(A_i)+ByteSub(A_j) \neq ByteSub(A_i+A_j)$, for $i,j=0,..,15$
    - **bijective**, i.e., there exists a one-to-one mapping of input and output bytes $\Rightarrow$ S-Box can be uniquely reversed
- In software implementations, the S-Box is usually realized as a lookup table

AES S-Box: Substitution values in hexadecimal notation for input byte (xy)

		y														
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
x	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Internal structure of AES



Diffusion Layer

- The Diffusion layer
  - provides diffusion over all input state bits
  - consists of two sublayers:
    - ShiftRows Sublayer: Permutation of the data on a byte level
    - MixColumn Sublayer: Matrix operation which combines ("mixes") blocks of four bytes
  - performs a linear operation on state matrices A, B, i.e., $DIFF(A)+DIFF(B)=DIFF(A+B)$

ShiftRows Sublayer

- Rows of the state matrix are shifted cyclically:

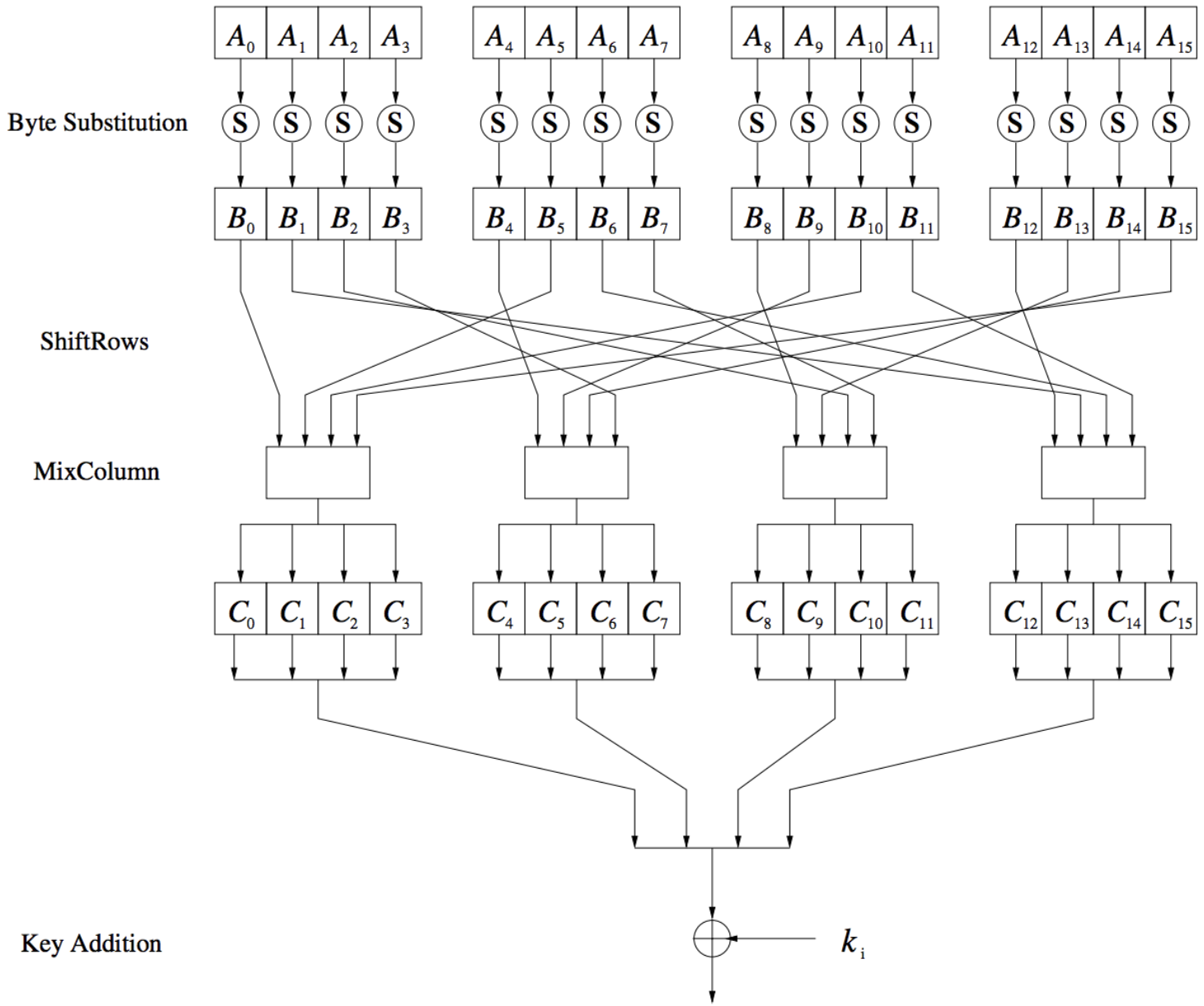
Input matrix

B_0	B_4	B_8	B_{12}
B_1	B_5	B_9	B_{13}
B_2	B_6	B_{10}	B_{14}
B_3	B_7	B_{11}	B_{15}

Output matrix

B_0	B_4	B_8	B_{12}	no shift
B_5	B_9	B_{13}	B_1	← one position left shift
B_{10}	B_{14}	B_2	B_6	← two positions left shift
B_{15}	B_3	B_7	B_{11}	← three positions left shift

Internal structure of AES



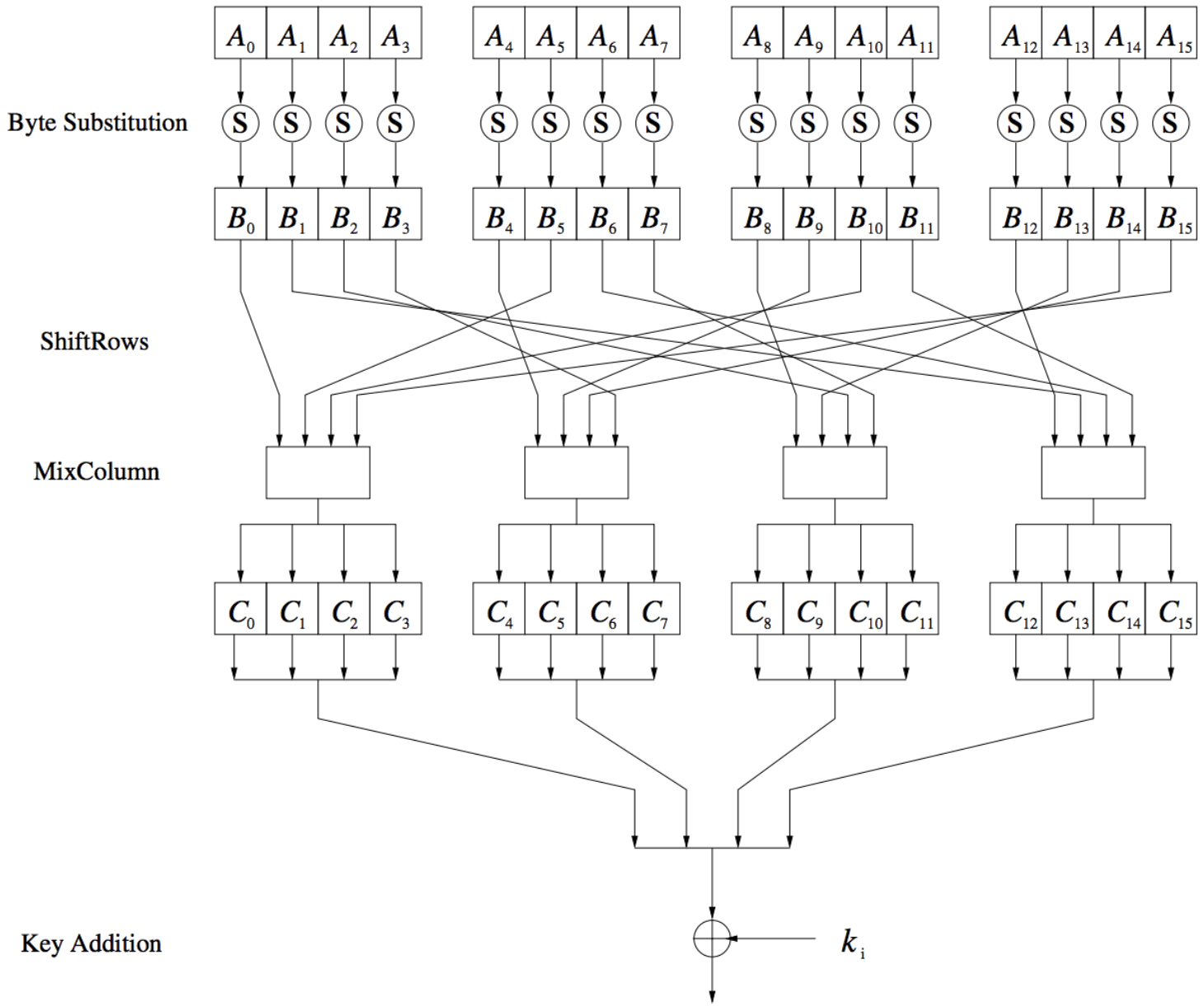
MixColumn Sublayer

- Linear transformation which mixes each column of the state matrix
- Since every input byte influences four output bytes, the MixColumn operation is the major diffusion element in AES
- Each $4\text{-}byte$ column is considered as a vector and multiplied by a fixed 4×4 matrix, e.g.,

$$\begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} B_0 \\ B_5 \\ B_{10} \\ B_{15} \end{pmatrix}$$

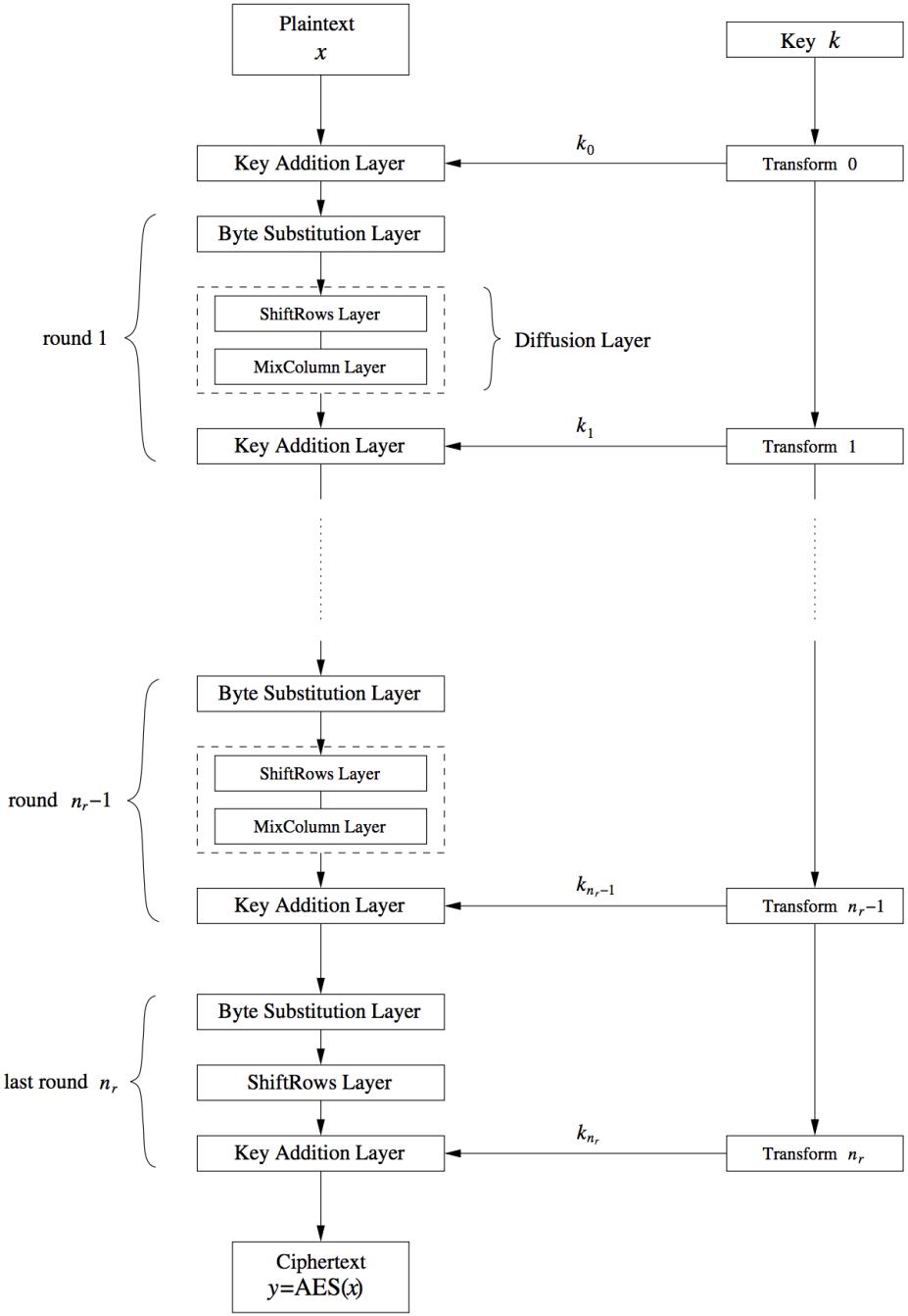
- where 01, 02 and 03 are given in hexadecimal notation
- All arithmetic is done in the Galois field $GF(2^8)$
  - In AES the finite field contains 256 elements and is denoted as $GF(2^8)$. This field was chosen because each of the field elements can be represented by one byte
- The combination of the ShiftRows and MixColumn layer makes it possible that after only three rounds every byte of the state matrix depends on all 16 plaintext bytes

Internal structure of AES



Key Addition Layer

- Inputs:
  - *16-byte* state matrix C
  - *16-byte* subkey k_i
- Output: $C \oplus k_i$
- The subkeys are generated in the key schedule


Plaintext
x
Key k
k_0
Key Addition Layer
Transform 0
Byte Substitution Layer
round 1
ShiftRows Layer
MixColumn Layer
Diffusion Layer
k_1
Key Addition Layer
Transform 1
Byte Substitution Layer
round n_r-1
ShiftRows Layer
MixColumn Layer
k_{n_r-1}
Key Addition Layer
Transform n_r-1
Byte Substitution Layer
last round n_r
ShiftRows Layer
k_{n_r}
Key Addition Layer
Transform n_r
Ciphertext
y=AES(x)

Key Schedule

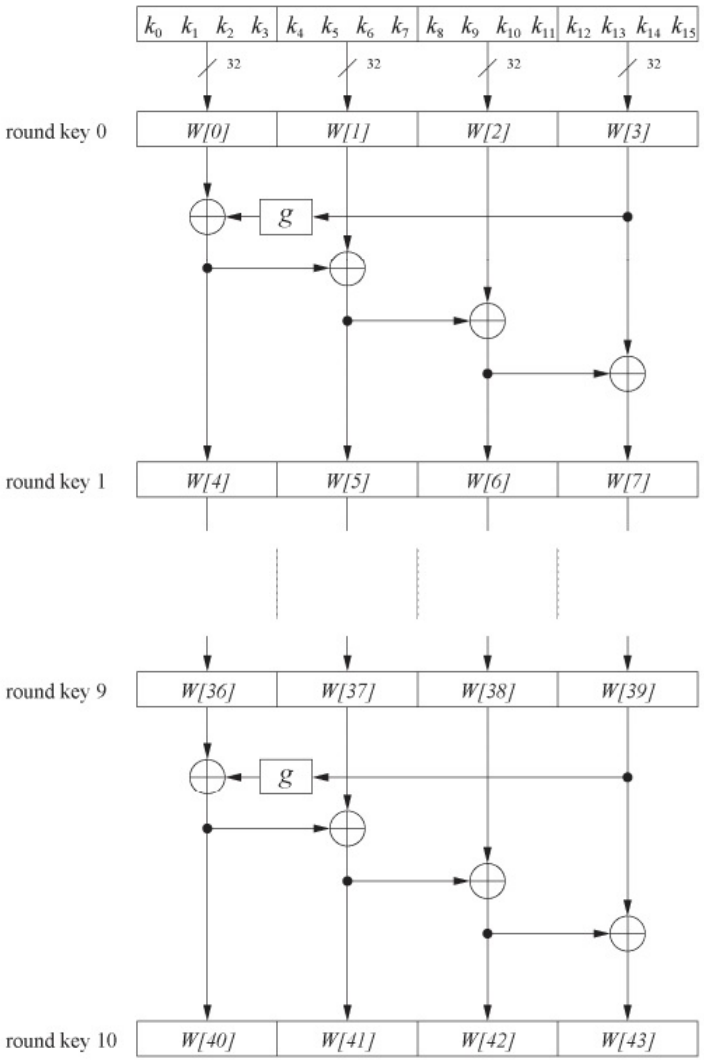
- Subkeys are derived recursively from the original *128/192/256-bit* input key
- Each round has 1 subkey, plus 1 subkey at the beginning of AES

Key length (bits)	Number of subkeys
128	11
192	13
256	15

- Note that an XOR addition of a subkey is used both at the input and output of AES. This process is sometimes referred to as Key whitening.
- i.e., key whitening is a technique intended to increase the security of an iterated block cipher. It consists of steps that combine the data with portions of the key.
- The number of subkeys: $\Rightarrow$ *# subkeys = # rounds + 1*
- There are different key schedules for the different key sizes

Key Schedule

Example: Key schedule for 128-bit key AES



- Word-oriented: 1 word = 32 bits
- 11 subkeys are stored in *W[0]…W[3], W[4]…W[7], … , W[40]…W[43]*
- First subkey *W[0]…W[3]* is the original AES key

AES Decryption

- To decrypt, process must be invertible
- Inverse of AddRoundKey is easy, since $\oplus$ is its own inverse
- MixColumn is invertible (inverse is also implemented as a lookup table)
- Inverse of ShiftRow is easy (cyclic shift the other direction)
- ByteSub is invertible (inverse is also implemented as a lookup table)

Security

- Brute-force attack: Due to the key length of 128, 192 or 256 bits, a brute-force attack is not possible
- Analytical attacks: There is no analytical attack known that is better than brute-force
- Side-channel attacks:
  - Any attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm itself (e.g. cryptanalysis and software bugs). Timing information, power consumption, electromagnetic leaks or even sound can provide an extra source of information, which can be exploited.[2]
  - Several side-channel attacks have been published
  - Note that side-channel attacks do not attack the underlying algorithm but the implementation of it

[2] Credit:https://en.wikipedia.org/wiki/Side-channel_attack

Lessons Learned

- AES is a modern block cipher which supports three key lengths of 128, 192 and 256 bit. It provides excellent long-term security against brute-force attacks.
- AES has been studied intensively since the late 1990s and no attacks have been found that are better than brute-force.
- AES is not based on Feistel networks. It's basic operations use Galois field arithmetic and provide strong diffusion and confusion.
- AES is part of numerous open standards such as IPsec or TLS, in addition to being the mandatory encryption algorithm for US government applications. It seems likely that the cipher will be the dominant encryption algorithm for many years to come
- AES is efficient in software and hardware.