Designing Block Ciphers
AES

B Tech III CSE (March-April 2024)

Dhiren Patel

(Week#12/13)

SKC – Symmetric Key Cryptography

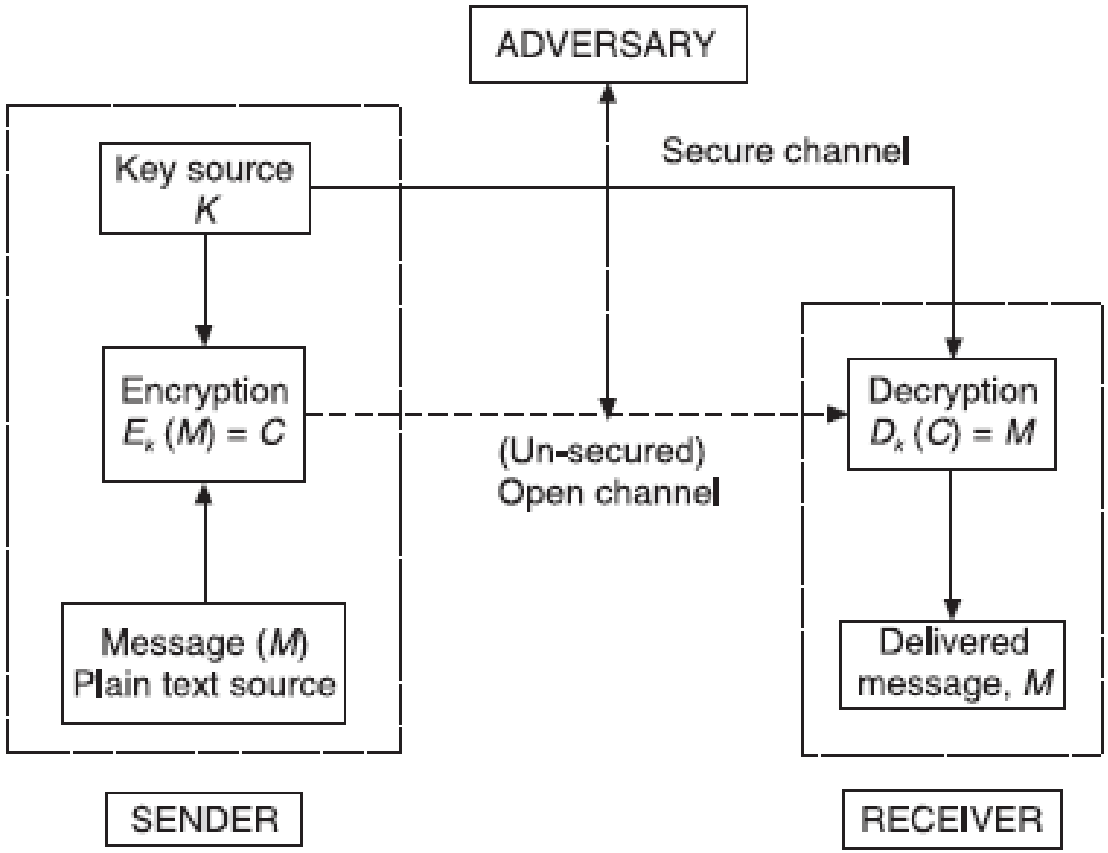
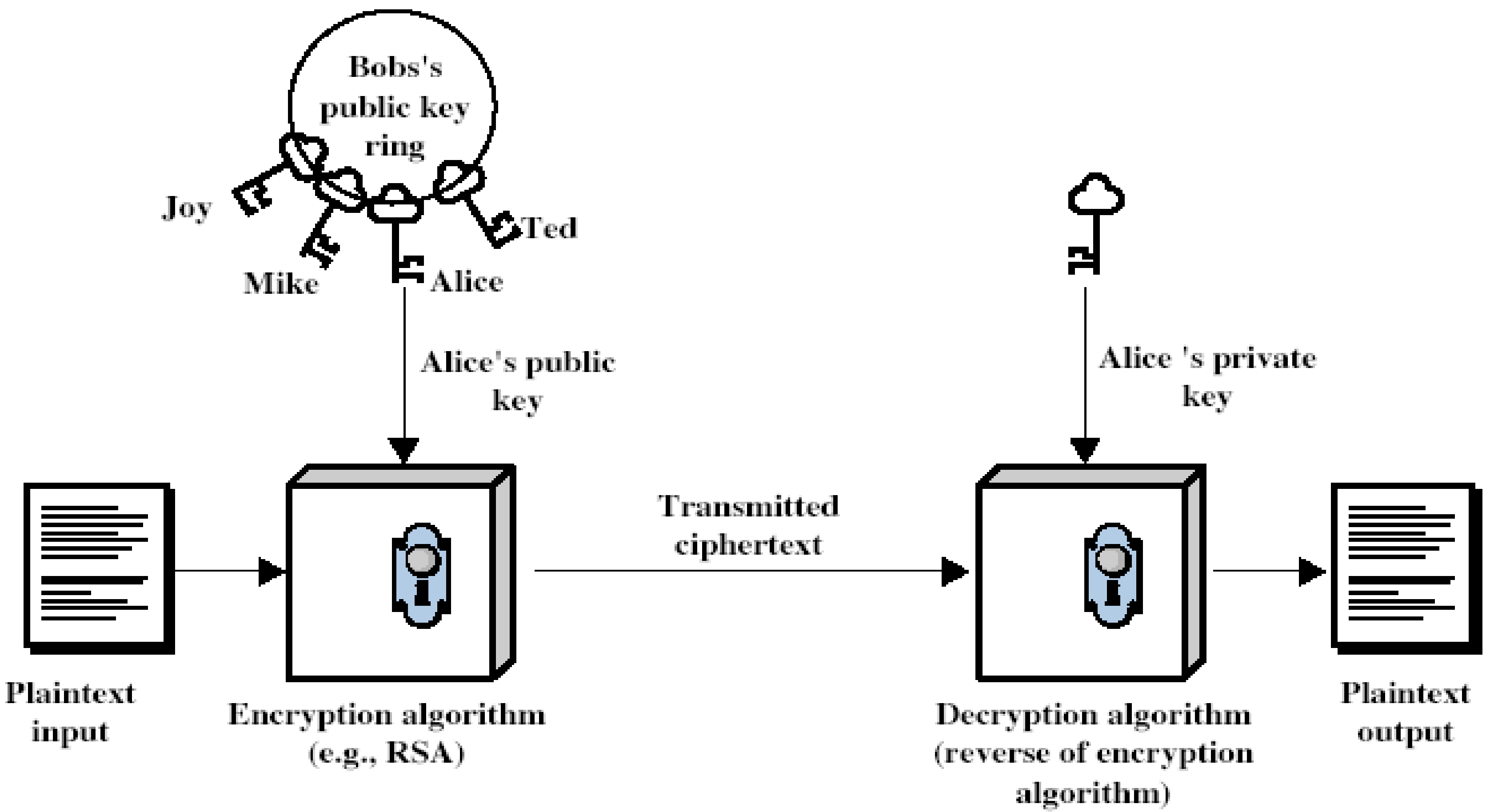


Figure 3.1 Communication using symmetric key cryptography ($k = k_1 = k_2$).

Public-Key Cryptography



S-box and P-box

- S - Substitution, P - Permutation
- Injects non-linearity into the design of cipher
- absence of a linear relationship between any subset of bits in the plain text, cipher text, and the key.

DES S Box 1

- takes 6 bits as input and gives 4 bits as output
- What is the output if input is 101000?
- 101000
- Row = 10 = 2 ,Column = 0100 = 4

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

Permutation box - P box

- Performs permutation or rearrangement of the bits in the input
- E.g. IP (initial permutation in DES – 64 bit), there is a 32bit P also!
- The meaning is as follows: the first bit of the output is taken from the 58th bit of the input; the second bit from the 50th bit, and so on, with the last bit of the output taken from the 7th bit of the input. This information is presented as a table for ease of presentation; it is a vector.

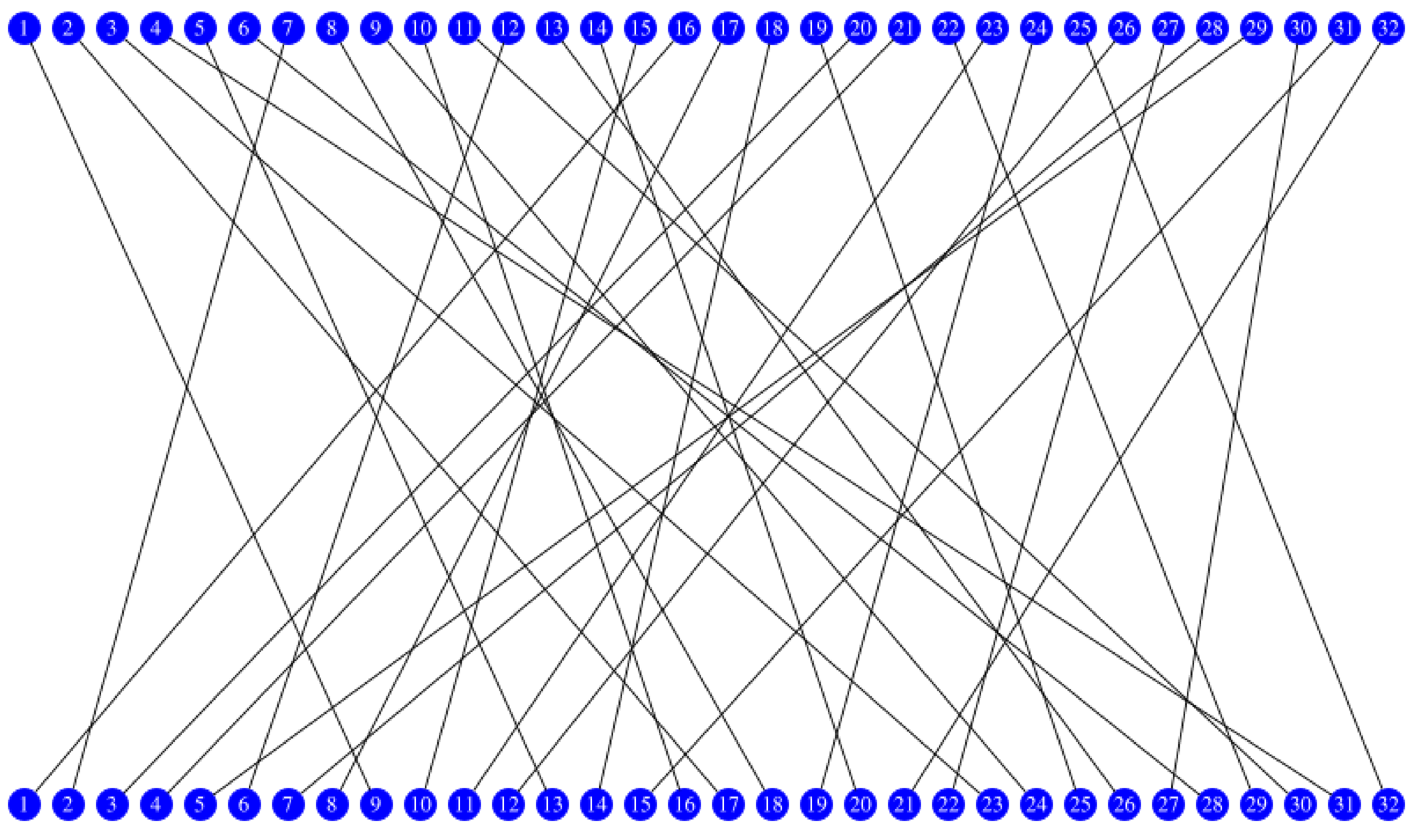
IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

P-Box

- Diffuses or spreads contiguous bits of the input across the block
- Removing local effect i.e. certain bits of the output would not be a certain bits of input

P (different than IP)

- The P permutation shuffles the bits of a 32-bit half-block.



DES subkeys

- DES consists of 16 “rounds”.
- Each round uses a roundkey, also called a subkey, derived from the main key.
- Subkey length is 48 bits for each subkey K1,...,K16.
- Subkeys are derived from the 56 bit key via the “key schedule” - a different 48-bit Sub Key is generated during each round **using a process called key transformation**

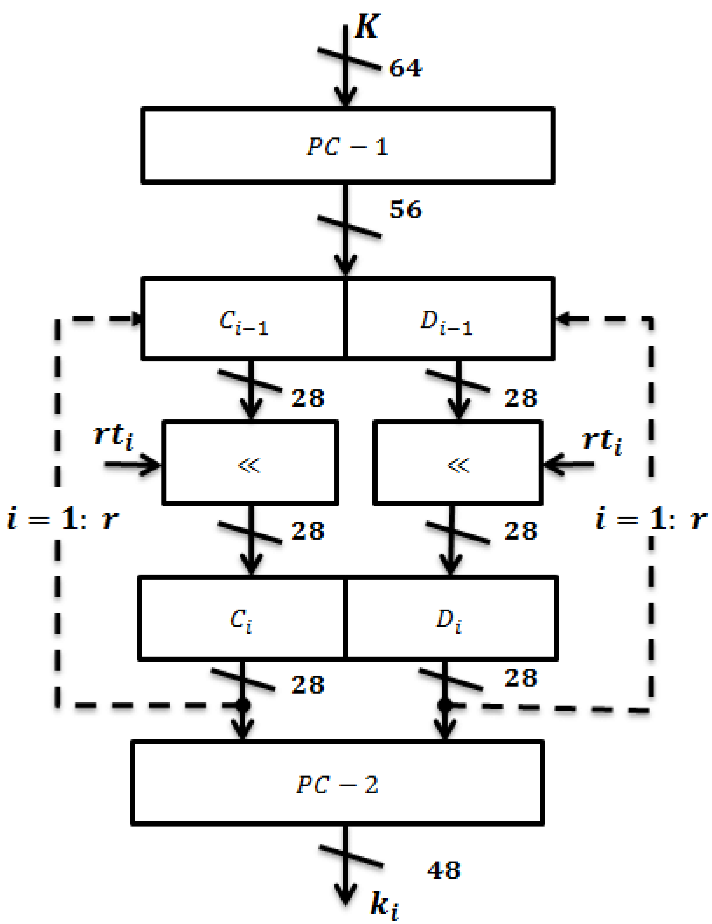
Key permutation (64 → 56) – PC1

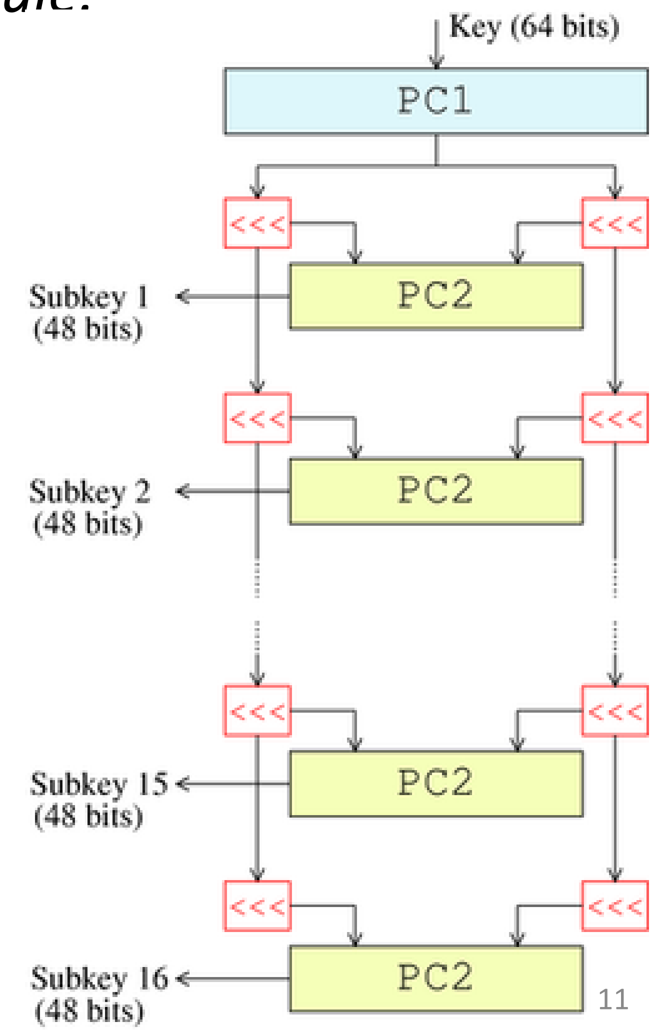
The 64-bit key is permuted to generate 56-bit key. (It is to note that bit 08, 16, 24, 32, 40, 48, 56, 64 are ignored/discarded)

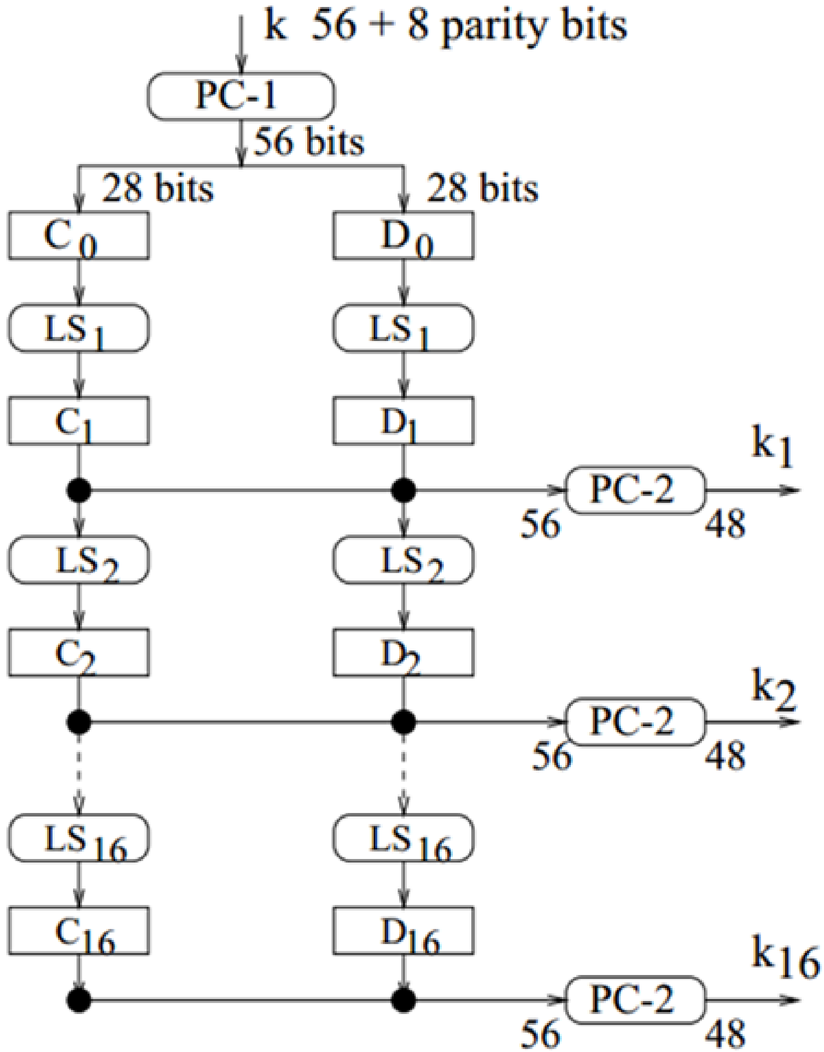
57	49	41	33	25	17	9	F
1	58	50	42	34	26	18	F
10	2	59	51	43	35	27	F
19	11	3	60	52	44	36	F
63	55	47	39	31	23	15	F
7	62	54	46	38	30	22	F
14	6	61	53	45	37	29	F
21	13	5	28	20	12	4	F

Key schedule of DES

- 16 nos. of 48-bit subkeys — one for each round — are derived from the main key using the *key schedule*.







1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Key schedule

- Next, split this key into left and right halves, C_0 and D_0, where each half has 28 bits.
- With C_0 and D_0 defined, create sixteen blocks C_n and D_n, 1<=n<=16.
- Each pair of blocks C_n and D_n is formed from the previous pair C_{n-1} and D_{n-1}, respectively using the following left shift schedule.
- C_3 and D_3 are obtained from C_2 and D_2, respectively, by two left shifts, and C_{16} and D_{16} are obtained from C_{15} and D_{15}, respectively, by one left shift

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Key schedule - **PC-2**

- Form the keys (48-bit) $\boldsymbol{K_n}$, for $1<=\boldsymbol{n}<=16$, by applying the following permutation table to each of the concatenated pairs $\boldsymbol{C_nD_n}$. Each pair has 56 bits, but **PC-2** only uses 48 of these.

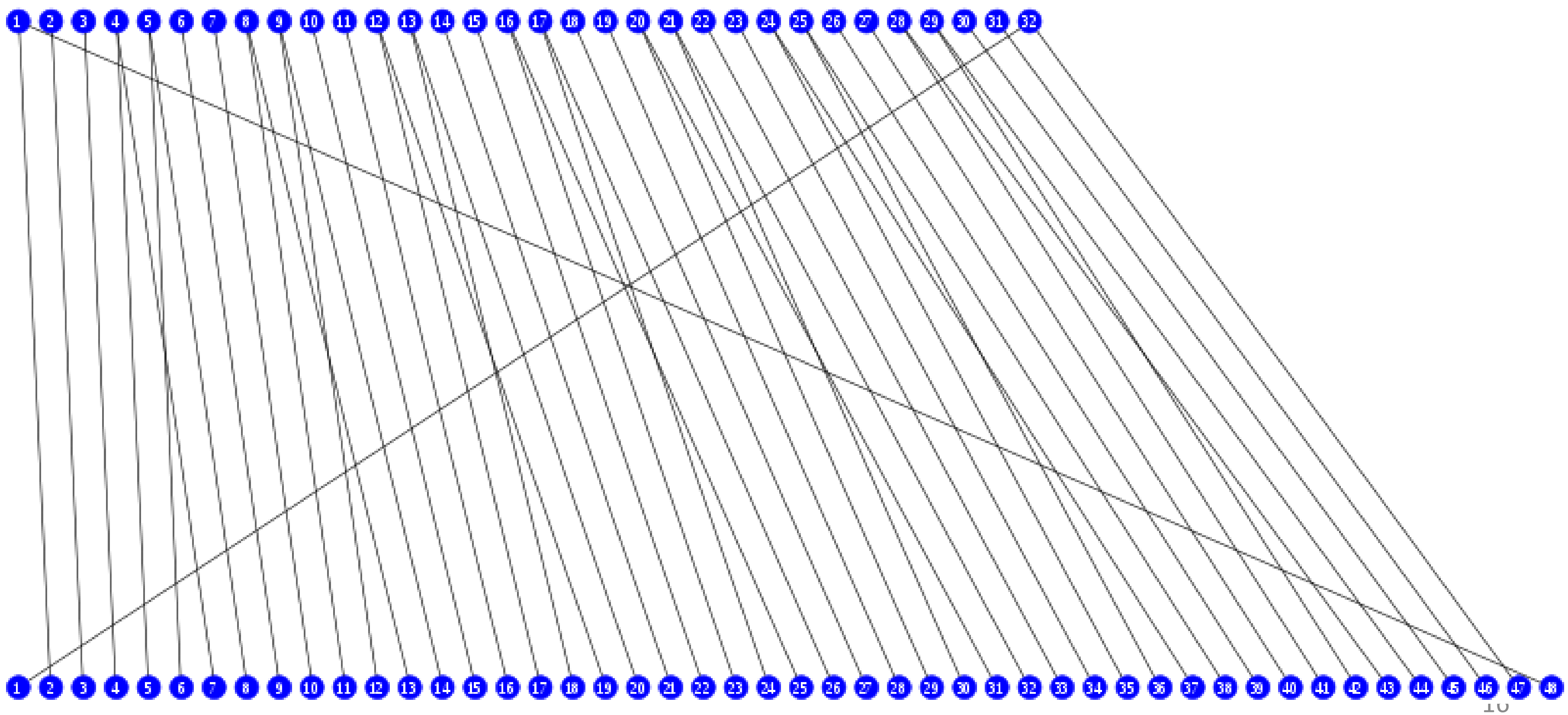
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

F -function

- operates on half a block (32 bits) at a time and consists of four stages:
- —The Feistel function (F-function) of DES
- *Expansion* — the 32-bit half-block is expanded to 48 bits using the *expansion permutation*, denoted *E* in the diagram, by duplicating half of the bits. The output consists of eight 6-bit(8*6=48bits) pieces, each containing a copy of 4 corresponding input bits, plus a copy of the immediately adjacent bit from each of the input pieces to either side.

E - function

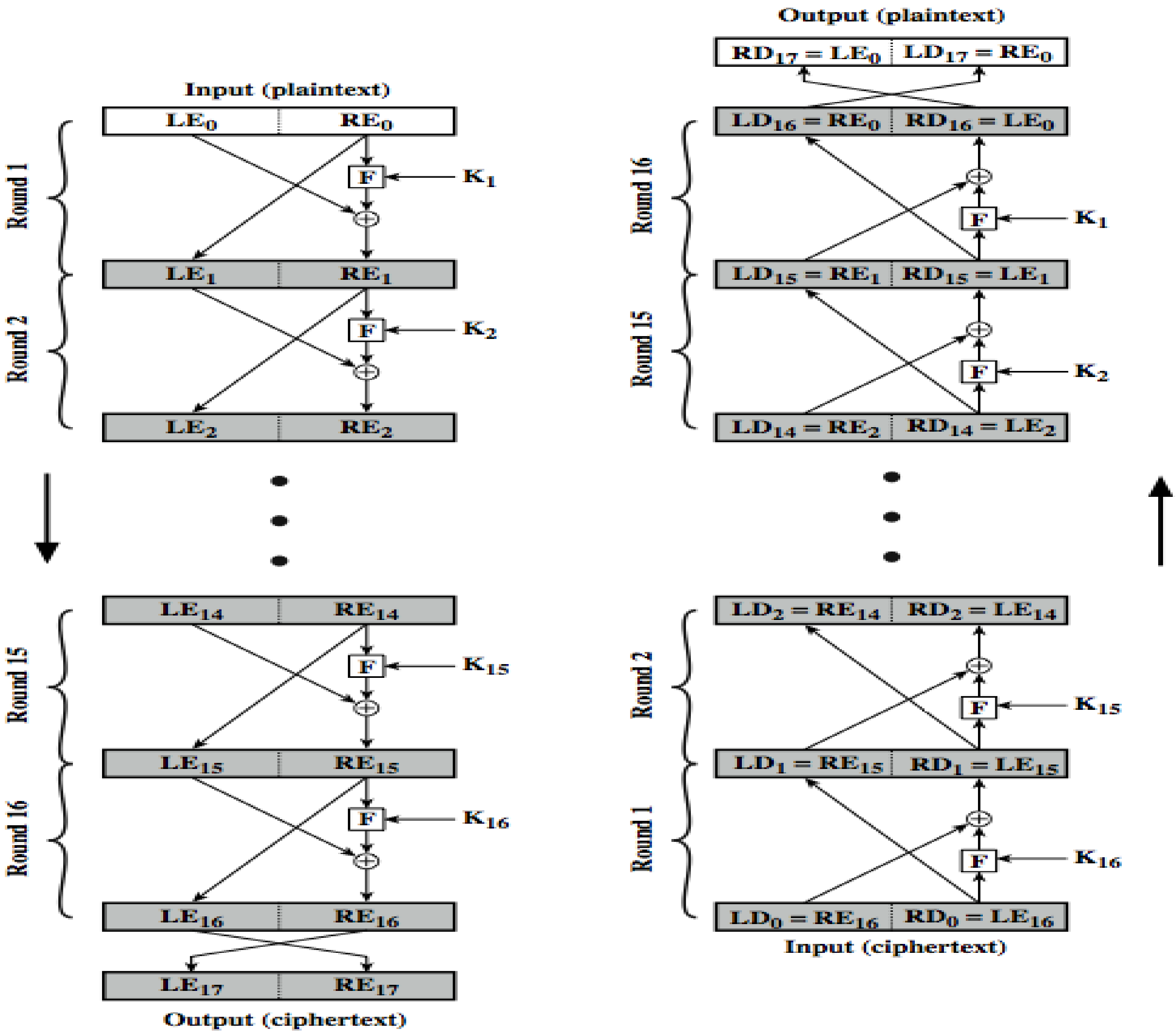
- Some bits from the input are duplicated at the output; e.g. the fifth bit of the input is duplicated in both the sixth and eighth bit of the output. Thus, the 32-bit half-block is expanded to 48 bits.



F-function

- *Substitution* — after mixing in the subkey, the block is divided into eight 6-bit pieces before processing by the *S-boxes*.
- Each of the eight S-boxes replaces its six input bits with four output bits according to a non-linear transformation, provided in the form of a lookup table.
- The S-boxes provide the core of the security of DES — without them, the cipher would be linear, and trivially breakable.
- *Permutation* — finally, the 32 outputs from the S-boxes are rearranged according to a fixed permutation, the *P-box*.
- This is designed so that, after expansion, each S-box's output bits are spread across 6 different S boxes in the next round.

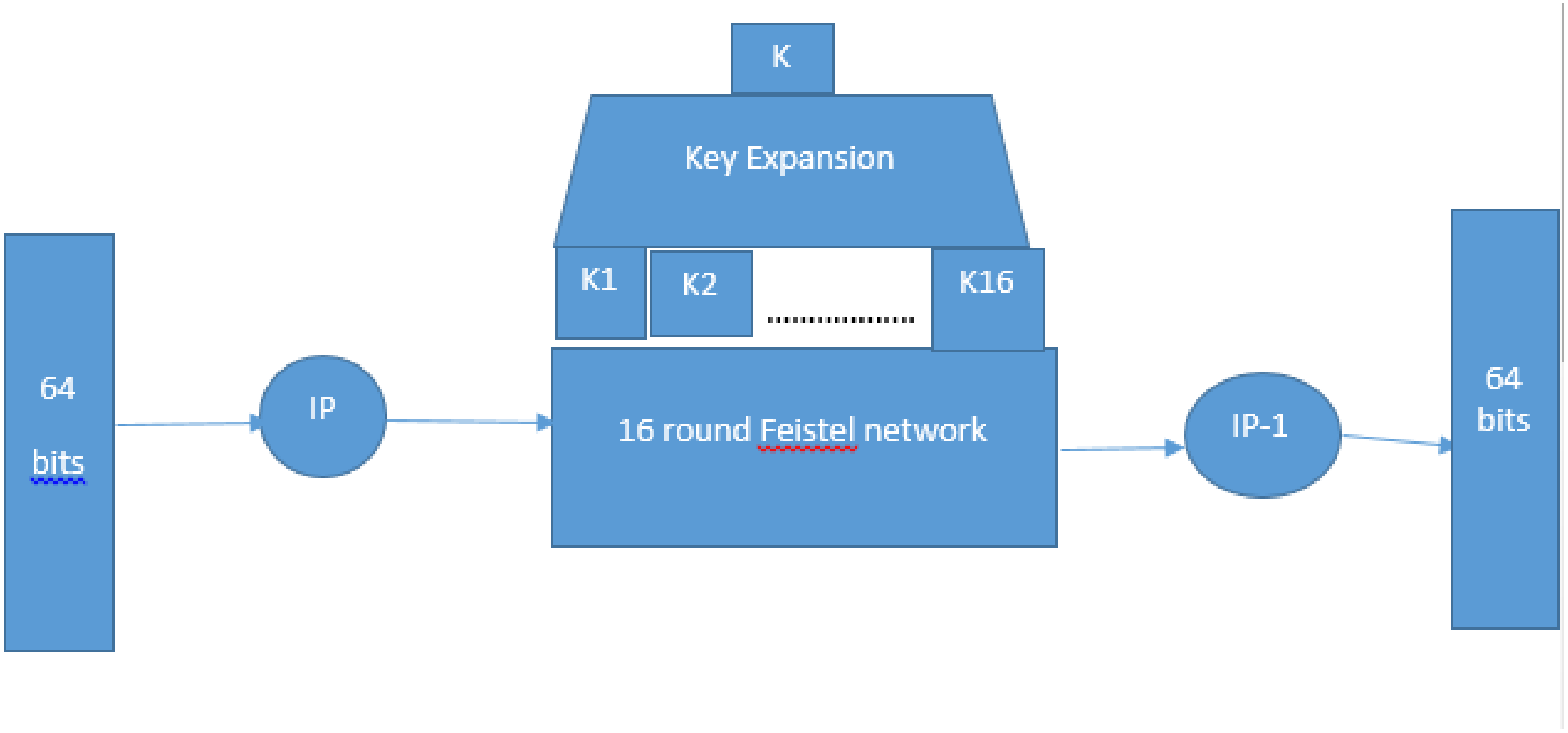
Feistel Cipher Structure (DES – encryption and decryption)



Decryption

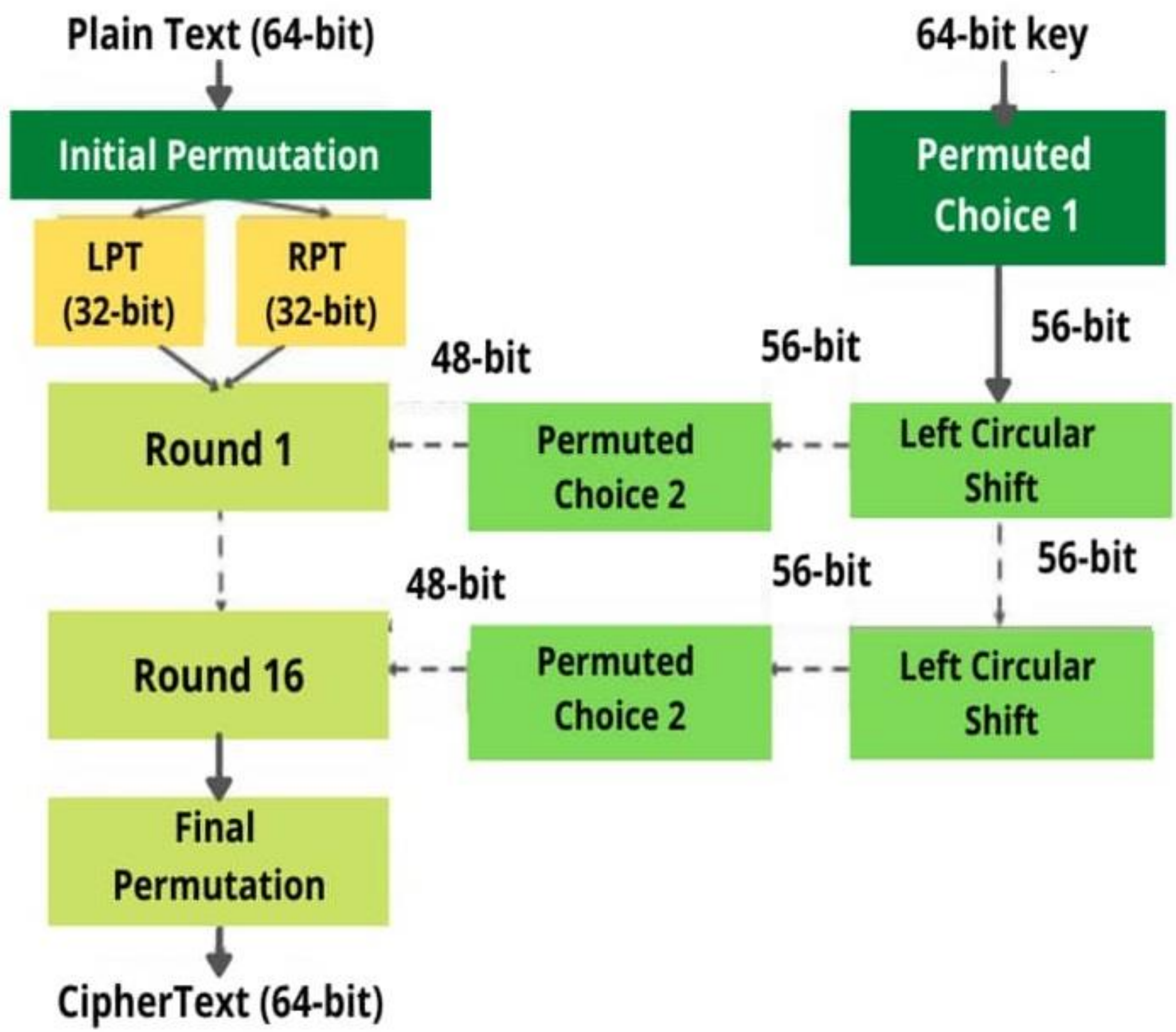
- Use cipher text as input to algorithm
- Use the subkeys in reverse order, i.e. k_n is for first round, k_{n-1} for the second round and so on…
- ***Avalanche effect*** – small change in either the plaintext or the key should produce a significant change in the ciphertext.

DES



64
bits
IP
K
Key Expansion
K1
K2
K16
16 round Feistel network
IP-1
64
bits

DES



Plain Text (64-bit)
Initial Permutation
LPT
(32-bit)
RPT
(32-bit)
Round 1
Round 16
Final
Permutation
CipherText (64-bit)
64-bit key
Permuted
Choice 1
56-bit
48-bit
56-bit
Permuted
Choice 2
Left Circular
Shift
56-bit
48-bit
56-bit
Permuted
Choice 2
Left Circular
Shift

Security of block ciphers

- The objective of a block cipher is to provide confidentiality.
- The corresponding objective of an adversary is to recover plaintext from ciphertext.
- The best measure of security for practical ciphers is the complexity of the best known attack. Various aspects of such complexity may be distinguished as follows:
  - Data Complexity
  - Storage Complexity
  - Processing Complexity
- Cost of attack v/s value of information!!!

Security of Block Ciphers

- A block cipher is *totally broken* if a key can be found, and *partially broken* if an adversary is able to recover part of the plaintext (but not the key) from ciphertext.
- To evaluate block cipher security, it is customary to always assume that an adversary
  (i) has access to all data transmitted over the ciphertext channel;
  (ii) knows all details of the encryption function except the secret key

Weak keys in DES

- few specific keys termed "weak keys" and "semi-weak keys".
- These are keys that cause the encryption mode of DES to act identically to the decryption mode of DES (albeit potentially that of a different key).
- four keys are weak and twelve keys are semi-weak
- DES *weak keys* produce sixteen identical subkeys.
- DES semi-weak keys produce 2 different subkeys (instead of 16).

Weak/Semi-weak keys

- Using weak keys, the outcome of the Permuted Choice 1 (PC1) in the DES key schedule leads to round keys being either all zeros, all ones or alternating zero-one patterns.
- Since all the subkeys are identical, and DES is a Feistel network, the encryption function is self-inverting; that is, encrypting twice produces the original plaintext.

https://www.youtube.com/watch?v=YVT4fcW7sI8

AES

https://www.youtube.com/watch?v=YVT4fcW7sI8

- **AES** stands for Advanced Encryption Standard, (developed? Declared as standard) in 2001. Key length can be 128-bits, 192-bits, and 256-bits. (Vincent Rijmen and Joan Daemen - KUL)
- AES is 128-bit block cipher
- USA NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug-99
- Rijndael was selected as the AES in Oct-2000
- issued as FIPS PUB 197 standard in Nov-2001

DES and AES

- DES involves 16 rounds of identical operations
- The rounds in DES are: Expansion, XOR operation with round key, Substitution and Permutation
- DES block size is 64 bit
- Number of rounds in AES depends on key length: 10(128-bits), 12(192-bits), or 14(256-bits)
- The rounds in AES are: Byte Substitution, Shift Row, Mix Column and Key Addition
- AES block size is 128 bit

AES Requirements (Request for Proposals (RFP) 1997)

- private key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- stronger & faster than Triple-DES
- active life of 20-30 years (+ archival use)
- provide full specification & design details
- both C & Java implementations
- NIST have released all submissions & unclassified analyses

AES Evaluation Criteria

- initial criteria:
  - security – effort to practically cryptanalyse
  - cost – computational
  - algorithm & implementation characteristics
- final criteria
  - general security
  - software & hardware implementation ease
  - implementation attacks
  - flexibility (in en/decrypt, keying, other factors)

AES Shortlist

- after testing and evaluation, shortlist in Aug-99:
  - MARS (IBM) - complex, fast, high security margin
  - RC6 (USA) - v. simple, v. fast, low security margin
  - Rijndael (Belgium) - clean, fast, good security margin
  - Serpent (Euro) - slow, clean, v. high security margin
  - Twofish (USA) - complex, v. fast, high security margin
- then subject to further analysis & comment
- saw contrast between algorithms with
  - few complex rounds verses many simple rounds
  - which refined existing ciphers verses new proposals

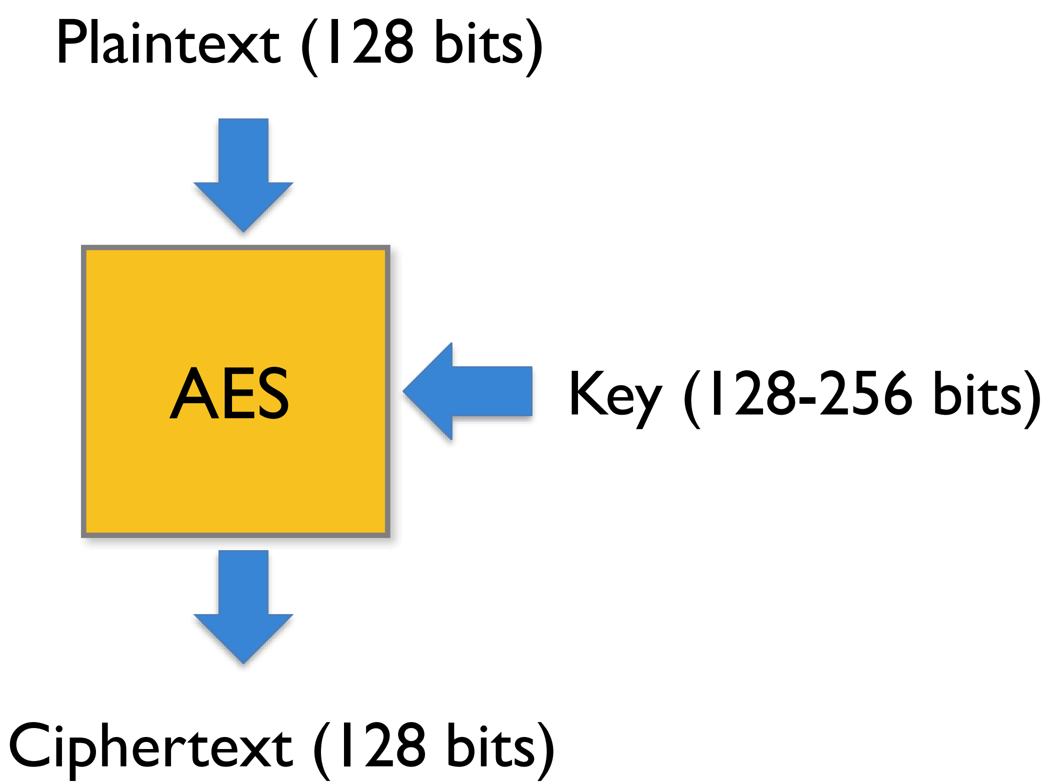
The AES Cipher - Rijndael

- Rijndael was selected as the AES in Oct-2000
  - Designed by Vincent Rijmen and Joan Daemen in Belgium
  - Issued as FIPS PUB 197 standard in Nov-2001
- An **iterative** rather than **Feistel** cipher
  - processes data as block of 4 columns of 4 bytes (128 bits)
  - operates on entire data block in every round
- Rijndael design:
  - simplicity
  - has 128/192/256 bit keys, 128 bits data
  - resistant against known attacks
  - speed and code compactness on many CPUs

V. Rijmen

J. Daemen

AES Conceptual Scheme



AES

- Stronger Key Lengths: AES comes in three variants with key lengths of 128 bits, 192 bits, and 256 bits. These longer key lengths make brute-force attacks significantly more difficult compared to DES's 56-bit key.
- Modern Design: AES benefits from advancements in cryptography. Its design incorporates substitutions, permutations, and other operations in a way that's more resistant to cryptanalysis techniques compared to DES.
- Wider Adoption: Endorsed by the National Institute of Standards and Technology (NIST) in the US, AES has become the de facto standard for symmetric-key encryption. Governments, businesses, and individuals worldwide rely on AES to protect sensitive data.

AES Mathematical Primitives

- Finite Field Arithmetic: AES operates in a finite field, specifically GF(2^8).
- Bitwise Operations: Basic bitwise operations like XOR (exclusive OR) and bit shifting are heavily used throughout the AES rounds.
- Linear Transformations: Specific linear transformations are applied to the data during the MixColumns step of each round. These transformations help to diffuse the data and increase the overall avalanche effect, meaning a small change in the plaintext should lead to a significant change in the ciphertext.

AES Cryptographic Primitives

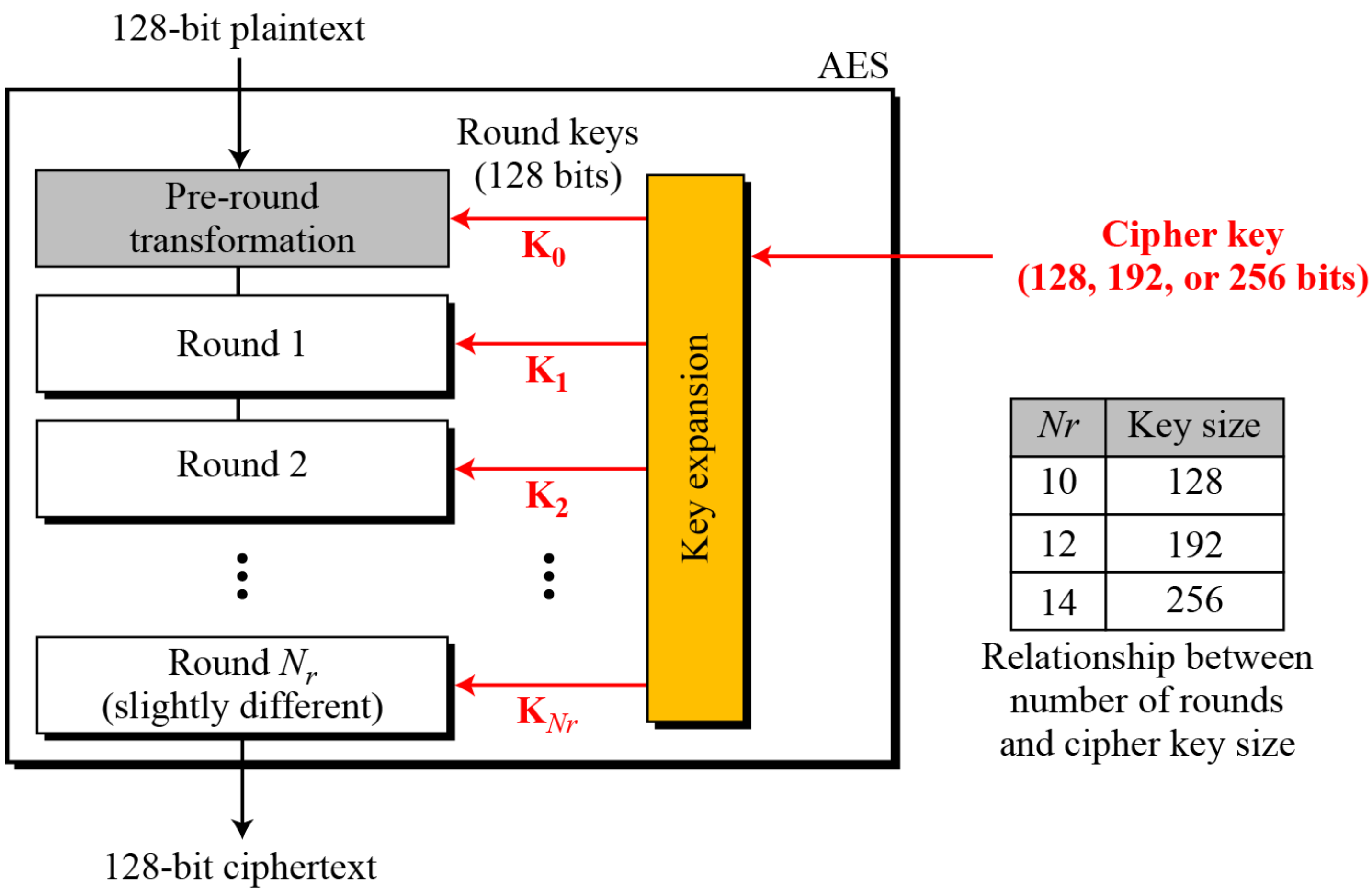
- Substitution: The core substitution operation in AES, called the S-box, replaces each byte of data with a different byte based on a predefined lookup table. This non-linear operation helps to confuse the relationship between the plaintext and ciphertext.
- Permutation: The ShiftRows step in each round permutes the bytes of the data matrix in a specific pattern. This disrupts the spatial relationship between data elements, further enhancing the confusion.
- Key Schedule: A key schedule algorithm takes the main secret key and generates a set of round keys used in each round of the encryption process. This ensures that different parts of the key are used in different rounds, strengthening the overall security.

The AES Cipher - Rijndael

- an **iterative** rather than **feistel** cipher
  - treats data in 4 groups of 4 bytes
  - operates an entire block in every round
- AES uses a specific Galois field, also known as *Rijndael's finite field*, to perform many essential operations.
- In particular, it uses $GF(2^8)$ with irreducible polynomial $x^8 + x^4 + x^3 + x + 1$.

Multiple rounds

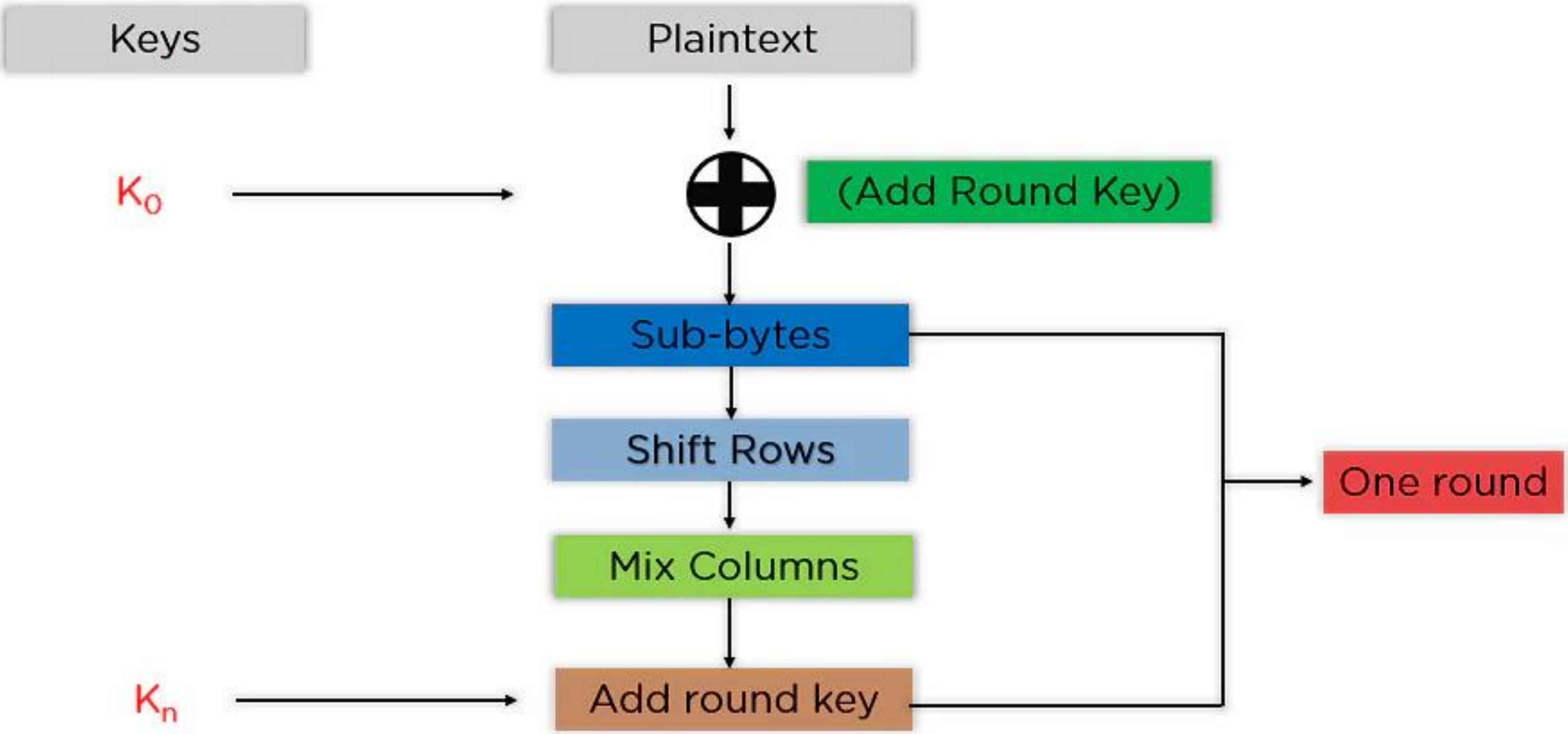
- Rounds are (almost) identical
  - First and last round are a little different

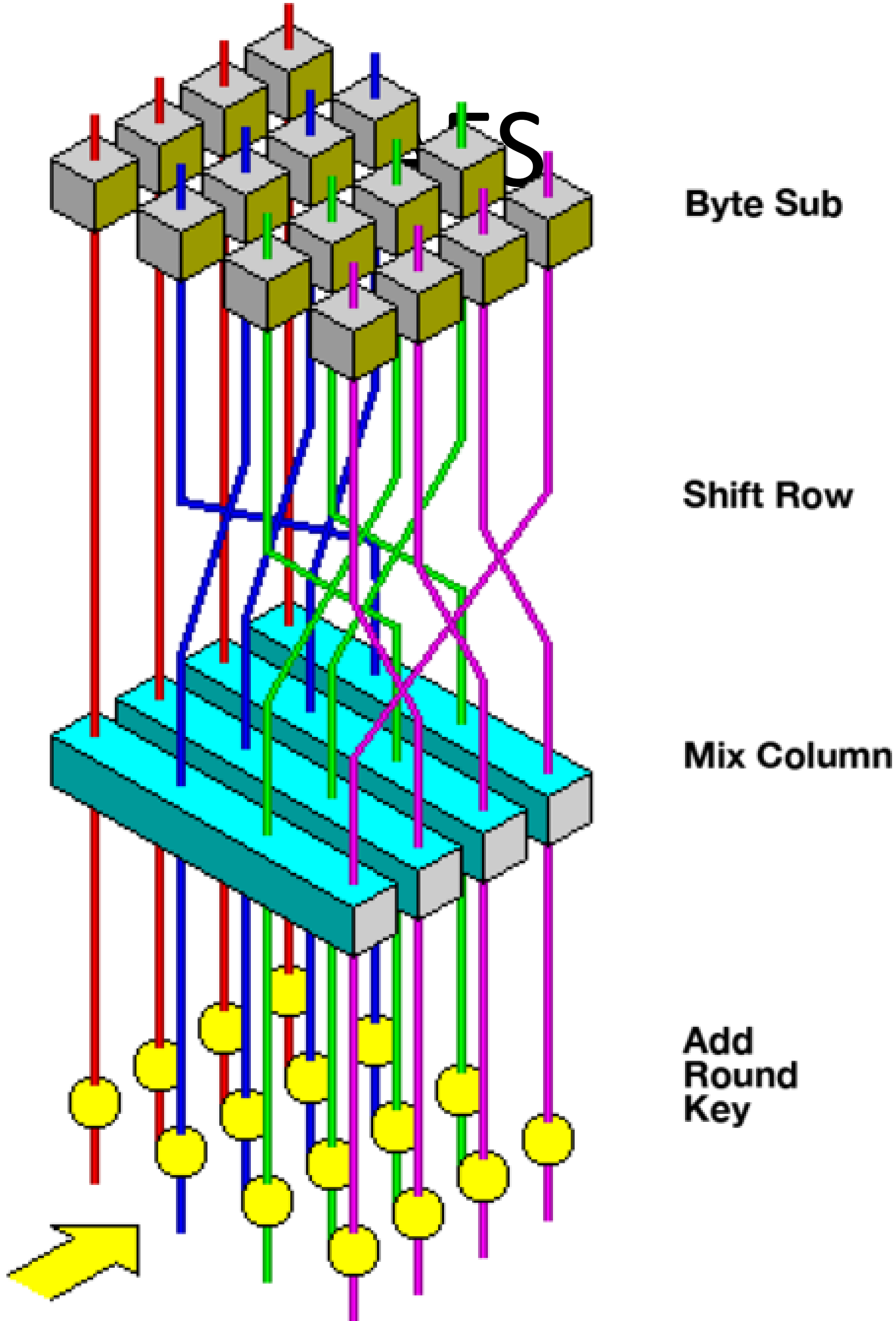


Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

AES one Round




Byte Sub
Shift Row
Mix Column
Add Round Key

High Level Description

Key Expansion

- Round keys are derived from the cipher key using Rijndael's key schedule

Initial Round

- AddRoundKey : Each byte of the state is combined with the round key using bitwise xor

Rounds

- SubBytes : non-linear substitution step
- ShiftRows : transposition step
- MixColumns : mixing operation of each column.
- AddRoundKey

Final Round

- SubBytes
- ShiftRows
- AddRoundKey

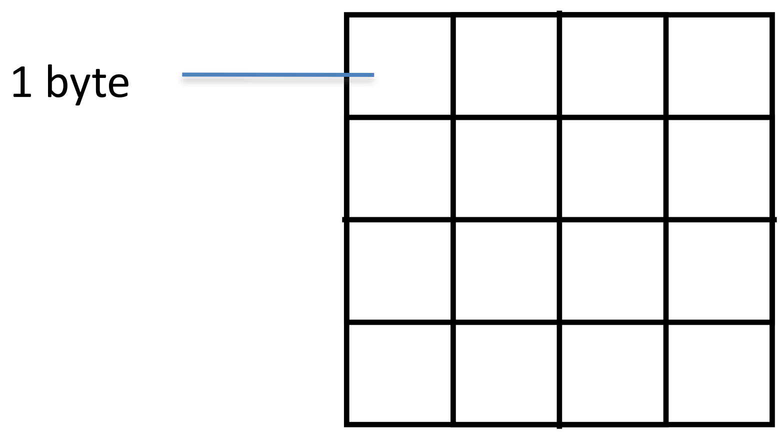
No MixColumns

Rijndael

- processes data as 4 groups of 4 bytes (state)
- has 9/11/13 rounds in which state undergoes:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multipy of groups)
  - add round key (XOR state with key material)
- initial XOR key material & incomplete last round
- all operations can be combined into XOR and table lookups - hence very fast & efficient

128-bit values

- Data block viewed as 4-by-4 table of bytes
- Represented as 4 by 4 matrix of 8-bit bytes.
- Key is expanded to array of 32 bits words

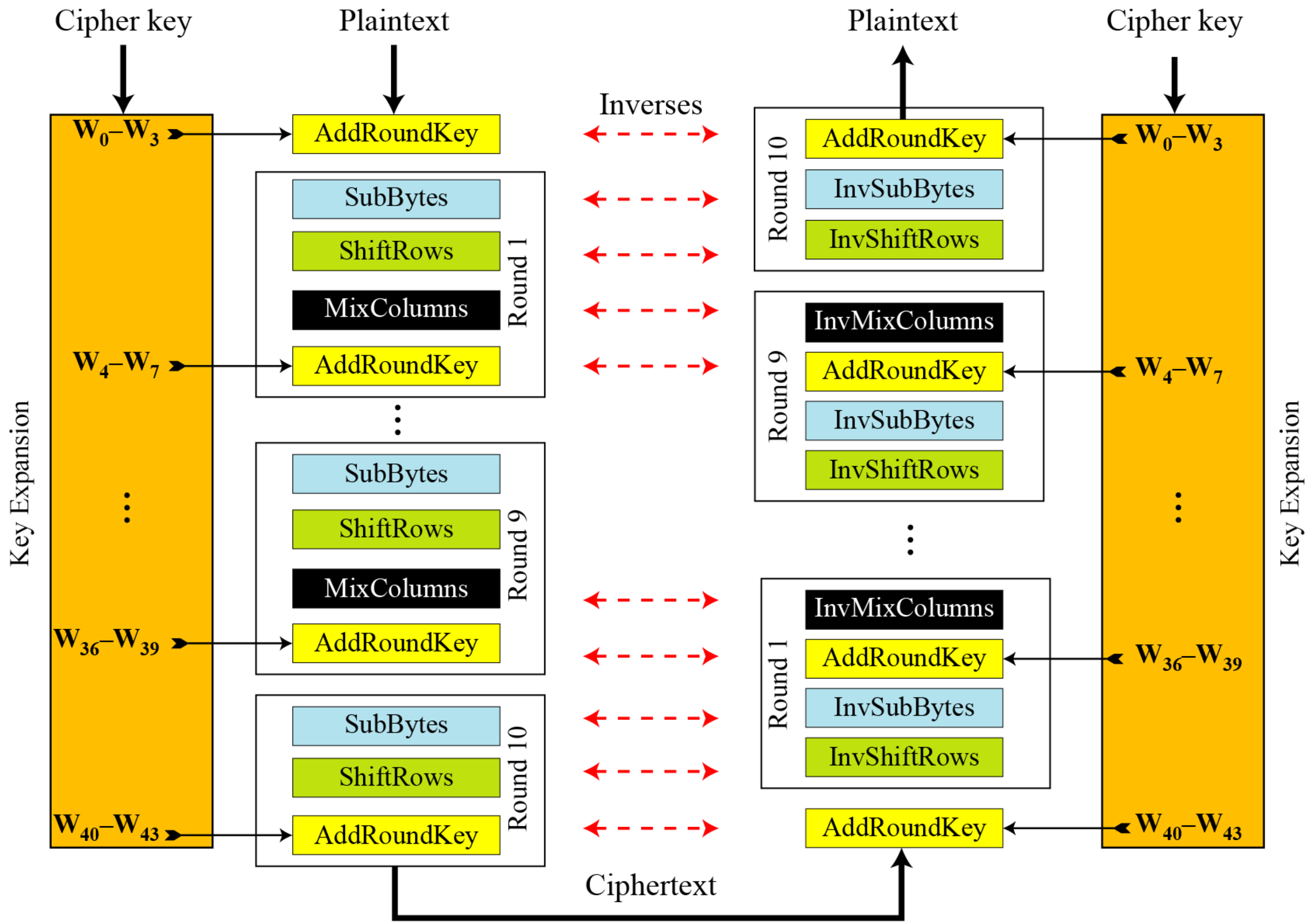


Changing Plaintext to State

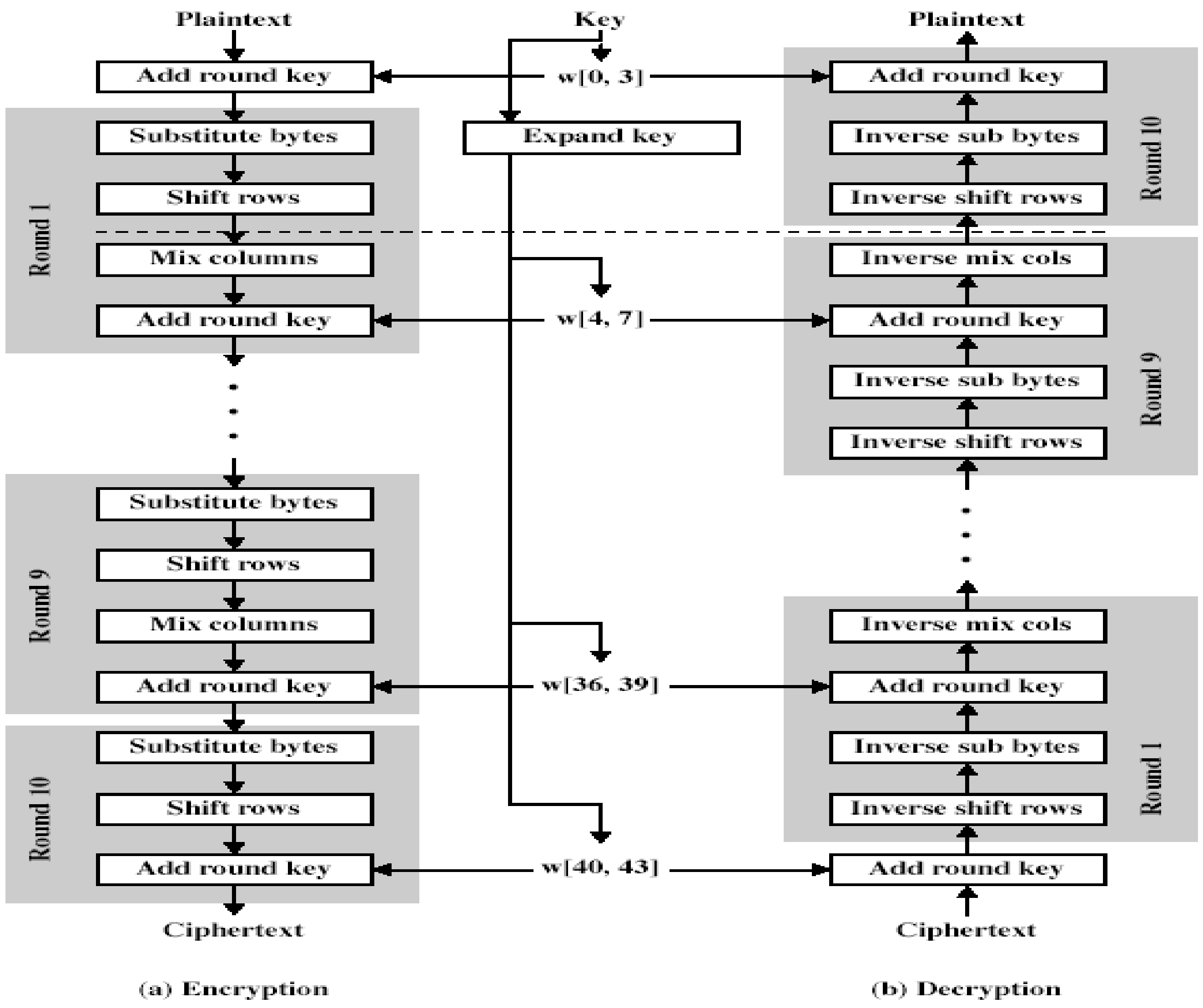
Text	A	E	S	U	S	E	S	A	M	A	T	R	I	X	Z	Z
Hexadecimal	00	04	12	14	12	04	12	00	0C	00	13	11	08	23	19	19

$$\begin{bmatrix} 00 & 12 & 0C & 08 \\ 04 & 04 & 00 & 23 \\ 12 & 12 & 13 & 19 \\ 14 & 00 & 11 & 19 \end{bmatrix} \text{State}$$

Overall Structure

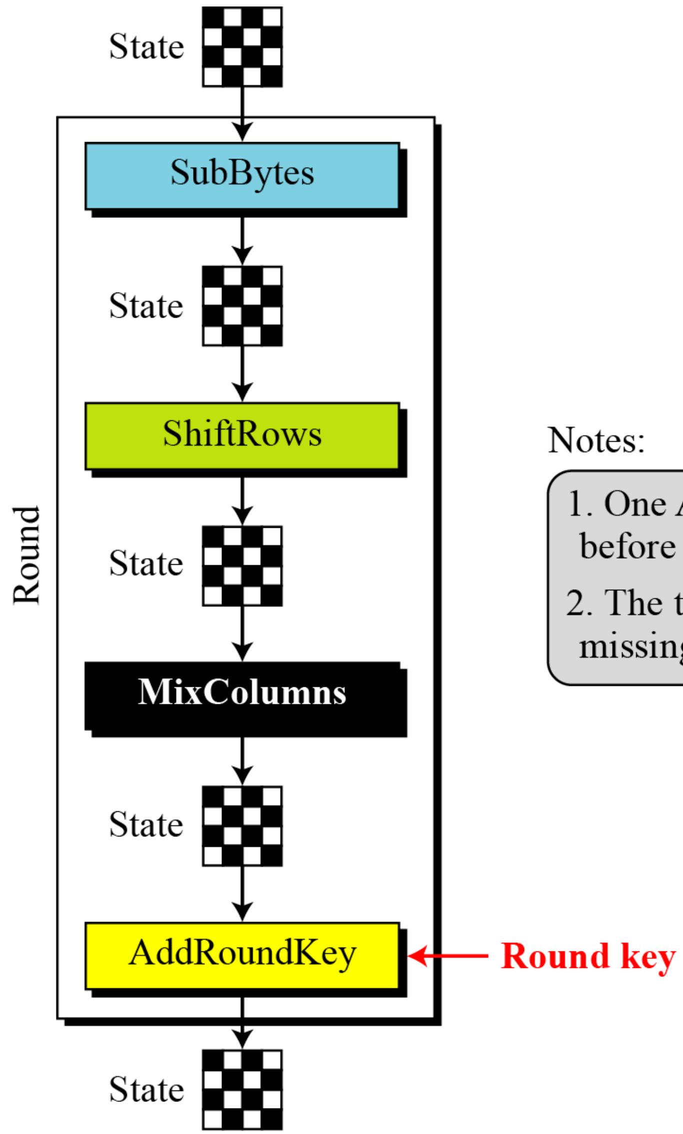


Rijndael



Plaintext
Key
Plaintext
Add round key
w[0, 3]
Add round key
Substitute bytes
Expand key
Inverse sub bytes
Round 10
Shift rows
Inverse shift rows
Round 1
Mix columns
Inverse mix cols
Add round key
w[4, 7]
Add round key
Inverse sub bytes
Round 9
Inverse shift rows
Substitute bytes
Round 9
Shift rows
Mix columns
Inverse mix cols
Add round key
w[36, 39]
Add round key
Substitute bytes
Inverse sub bytes
Round 1
Round 10
Shift rows
Inverse shift rows
Add round key
w[40, 43]
Add round key
Ciphertext
Ciphertext
(a) Encryption
(b) Decryption

Details of Each Round



Notes:

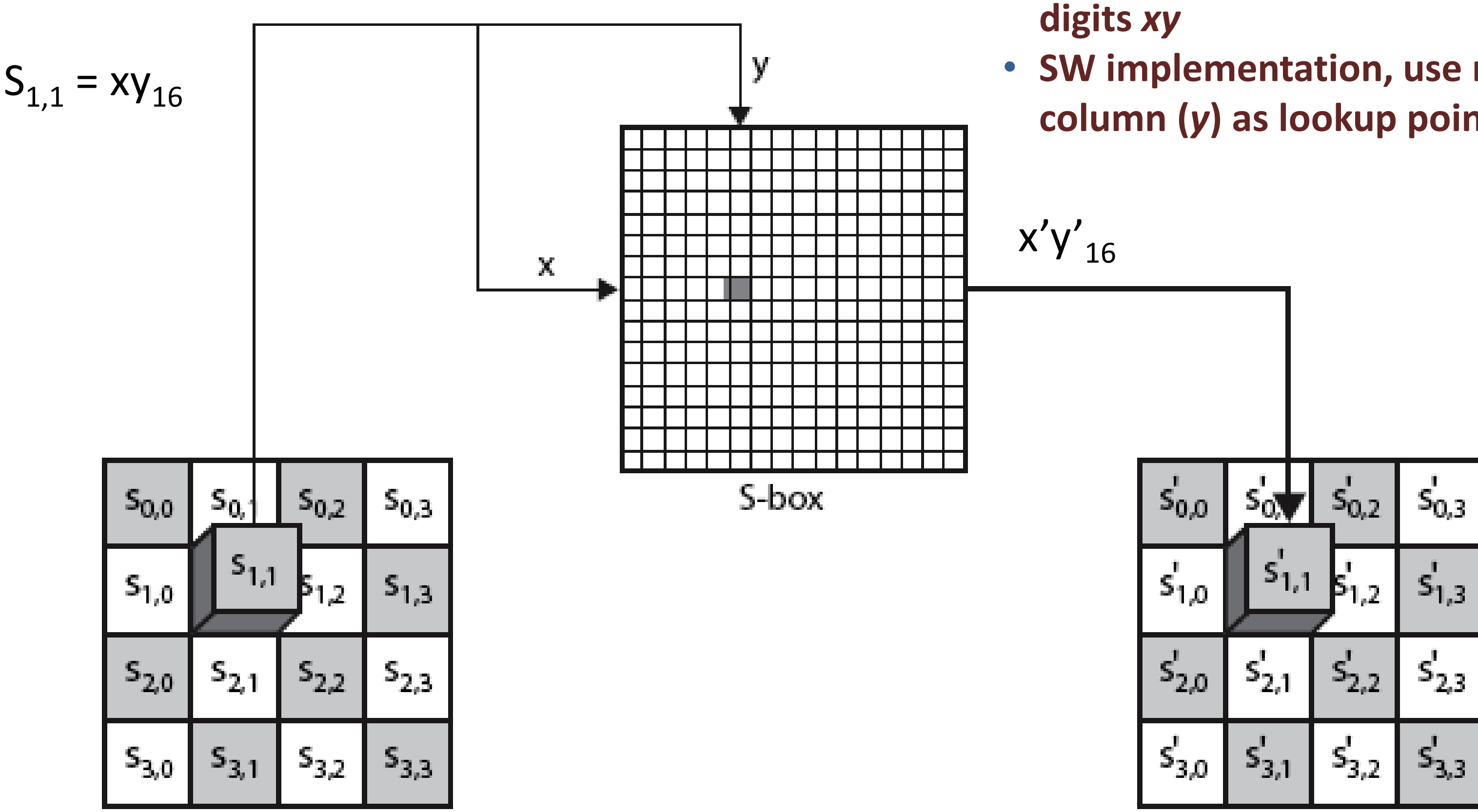
1. One AddRoundKey is applied before the first round.
2. The third transformation is missing in the last round.

Byte Substitution

- a simple substitution of each byte
- uses one table of 16x16 bytes containing a permutation of all 256 8-bit values
- each byte of state is replaced by byte in row (left 4-bits) & column (right 4-bits)
  - eg. byte {95} is replaced by row 9 col 5 byte
  - which is the value {2A}
- S-box is constructed using a defined transformation of the values in GF(2^8)
- designed to be resistant to all known attacks

SubBytes Operation

- The SubBytes operation involves 16 independent byte-to-byte transformations.



- **Interpret the byte as two hexadecimal digits *xy***
- **SW implementation, use row (*x*) and column (*y*) as lookup pointer**

SubBytes Table

- Implement by Table Lookup

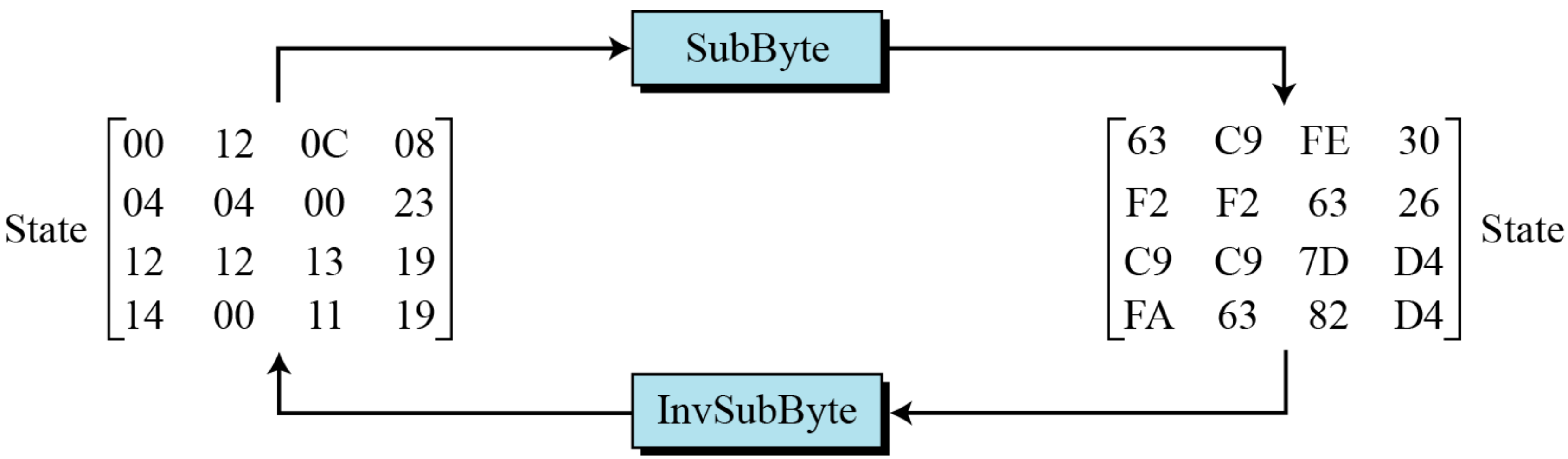
		y															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	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
x	**7**	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	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

InvSubBytes Table

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

Sample SubByte Transformation

- The SubBytes and InvSubBytes transformations are inverses of each other.

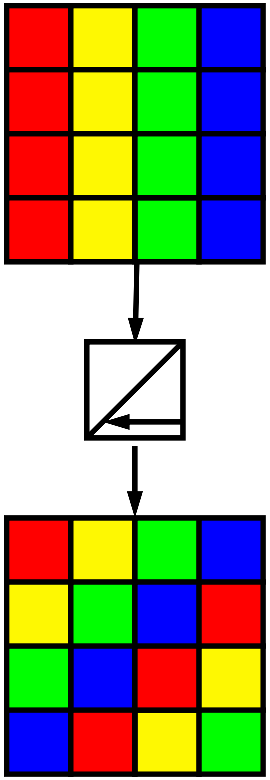


Shift Rows

- a circular byte shift in each each
  - 1st row is unchanged
  - 2nd row does 1 byte circular shift to left
  - 3rd row does 2 byte circular shift to left
  - 4th row does 3 byte circular shift to left
- decrypt does shifts to right
- since state is processed by columns, this step permutes bytes between the columns

ShiftRows

- Shifting, which permutes the bytes.
- A circular byte shift in each each
  - 1^{st} row is unchanged
  - 2^{nd} row does 1 byte circular shift to left
  - 3rd row does 2 byte circular shift to left
  - 4th row does 3 byte circular shift to left
- In the encryption, the transformation is called ShiftRows
- In the decryption, the transformation is called InvShiftRows and the shifting is to the right



Mix Columns

- each column is processed separately
- each byte is replaced by a value dependent on all 4 bytes in the column
- effectively a matrix multiplication in GF(2^8) using prime poly $m(x) = x^8+x^4+x^3+x+1$

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

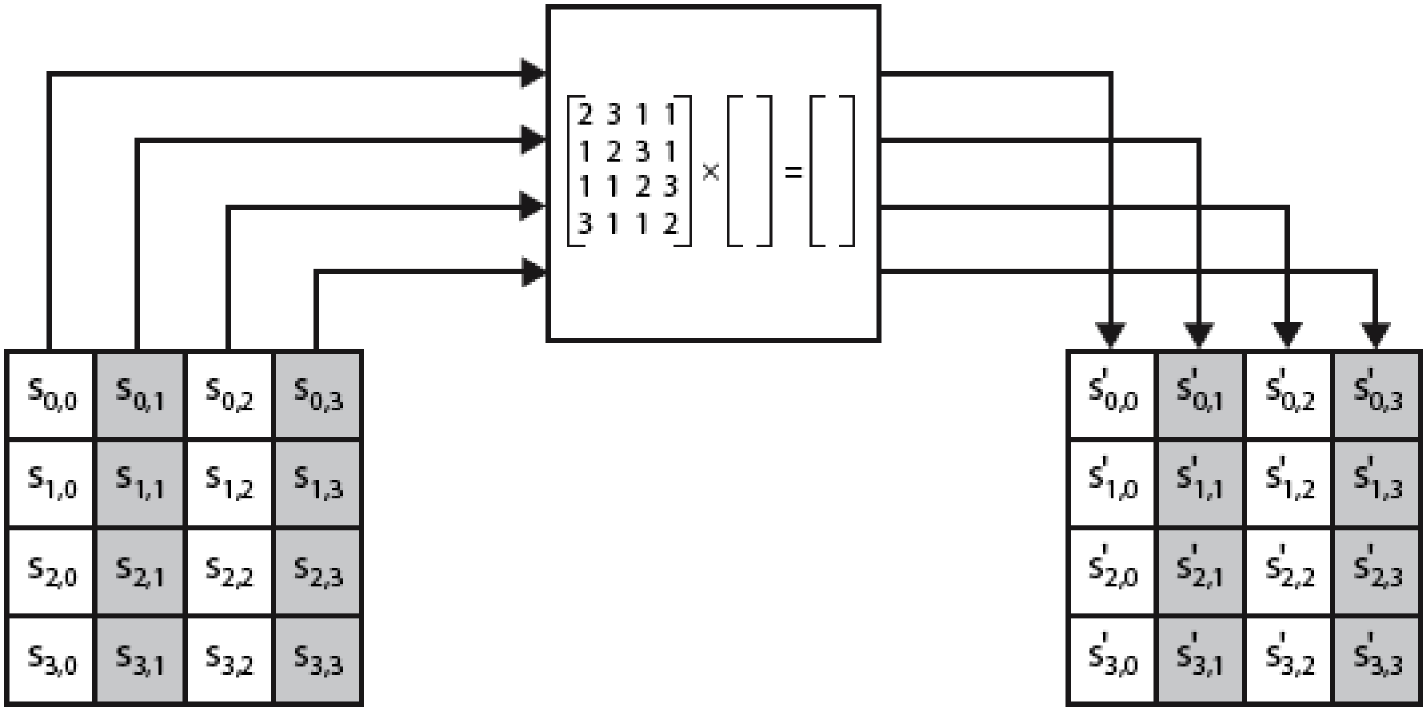
MixColumns

- ShiftRows and MixColumns provide diffusion to the cipher
- Each column is processed separately
- Each byte is replaced by a value dependent on all 4 bytes in the column
- Effectively a matrix multiplication in GF(2^8) using prime poly $m(x) = x^8+x^4+x^3+x+1$

$$\begin{array}{c} a\mathbf{x} + b\mathbf{y} + c\mathbf{z} + d\mathbf{t} \\ e\mathbf{x} + f\mathbf{y} + g\mathbf{z} + h\mathbf{t} \\ i\mathbf{x} + j\mathbf{y} + k\mathbf{z} + l\mathbf{t} \\ m\mathbf{x} + n\mathbf{y} + o\mathbf{z} + p\mathbf{t} \end{array} \begin{array}{c} \rightarrow \\ \rightarrow \\ \rightarrow \\ \rightarrow \end{array} \begin{bmatrix} \square \\ \square \\ \square \\ \square \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \times \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{z} \\ \mathbf{t} \end{bmatrix}$$

New matrix　　Constant matrix　　Old matrix

MixClumns Scheme

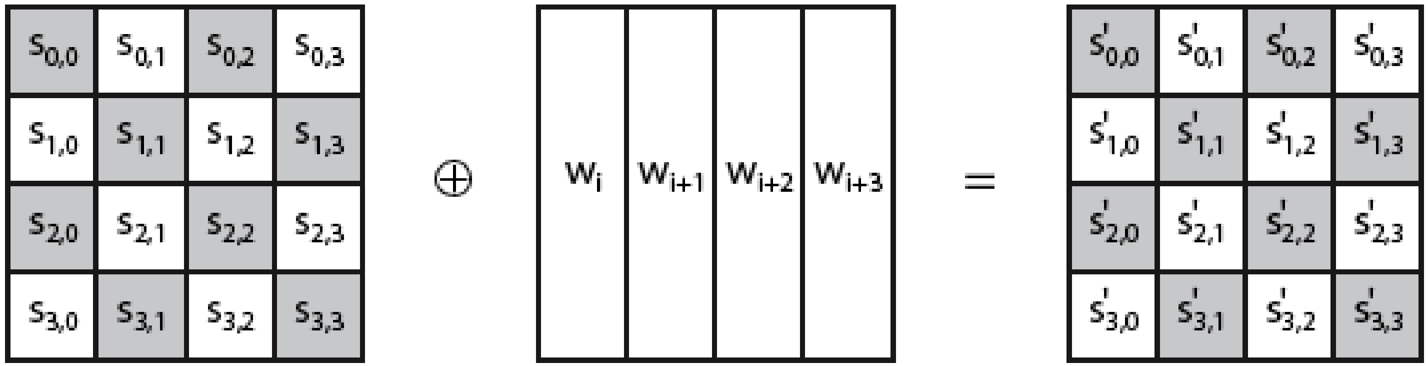


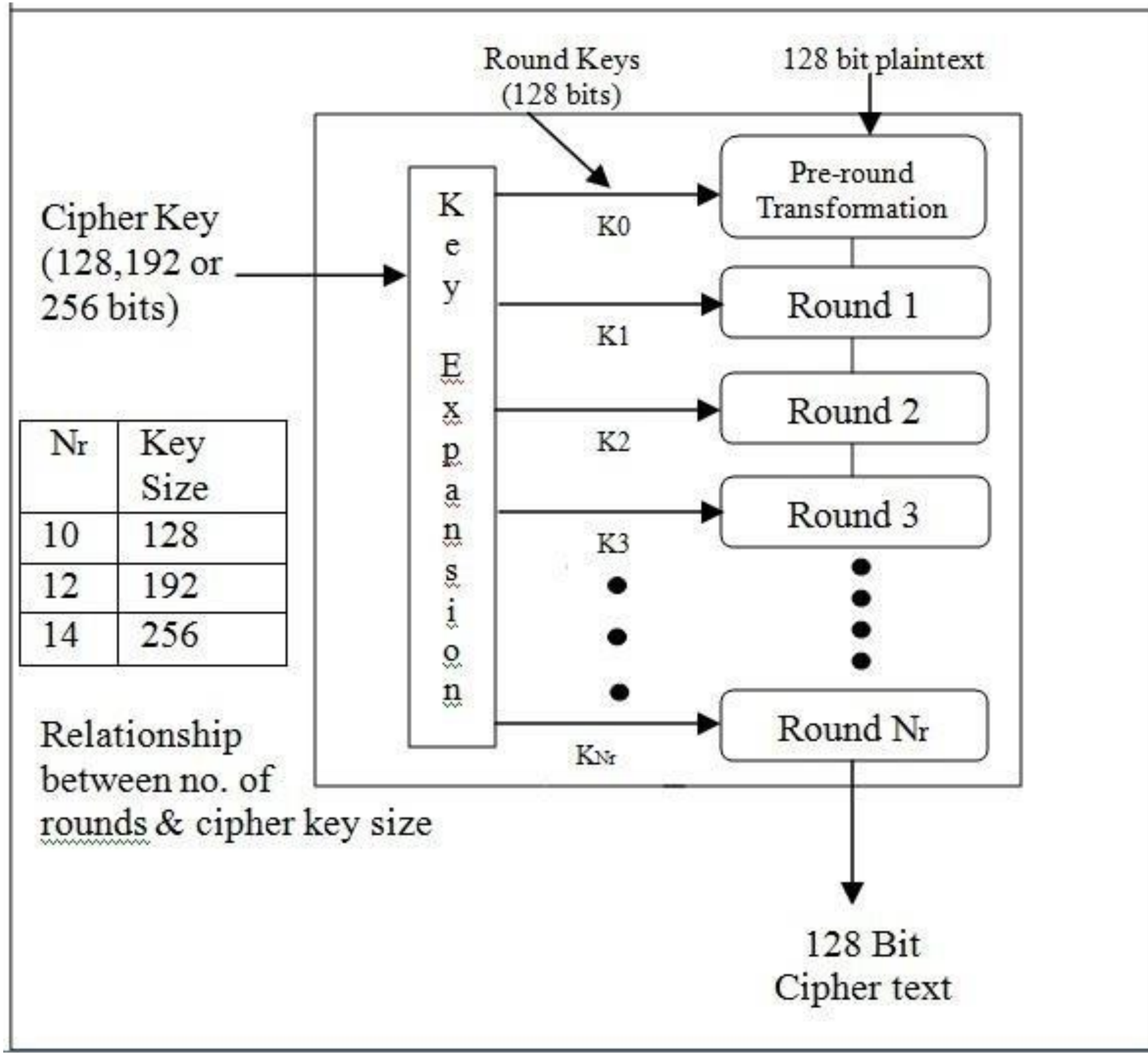
The MixColumns transformation operates at the column level; it transforms each column of the state to a new column.

Add Round Key (come from key schedule/expansion) – later slides

- (Before any encryption takes place, separate 128-bit keys must be generated for each round).
- XOR state with 128-bits of the round key
- again processed by column (though effectively a series of byte operations)
- inverse for decryption is identical since XOR is own inverse, just with correct round key
- designed to be as simple as possible

AddRoundKey Scheme




Round Keys
(128 bits)
128 bit plaintext
Cipher Key
(128,192 or
256 bits)
Key Expansion
Pre-round
Transformation
K0
Round 1
K1
Round 2
K2
Round 3
K3
Round Nr
KNr
Nr
Key Size
10
128
12
192
14
256
Relationship
between no. of
rounds & cipher key size
128 Bit
Cipher text

AES key expansion

- The AES key expansion takes as input a four word (16byte) key and produces a linear array of 44 words (176bytes).
- This is sufficient to provide a four-word round key for the initial AddRoundKey stage and each of the 10 rounds of the cipher.

AES Key Expansion

- takes 128-bit (16-byte) key and expands into array of 44/52/60 32-bit words (10,12,14 rounds)
- start by copying key into first 4 words
- then loop creating words that depend on values in previous & 4 places back
  - in 3 of 4 cases just XOR these together
  - every 4th has S-box + rotate + XOR constant of previous before XOR together
- designed to resist known attacks

AES Key Scheduling/expansion

- takes 128-bits (16-bytes) key and expands into array of 44 32-bit words

Round	*Words*			
Pre-round	$\mathbf{w}_0$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_3$
1	$\mathbf{w}_4$	$\mathbf{w}_5$	$\mathbf{w}_6$	$\mathbf{w}_7$
2	$\mathbf{w}_8$	$\mathbf{w}_9$	$\mathbf{w}_{10}$	$\mathbf{w}_{11}$
. . .	. . .			
N_r	$\mathbf{w}_{4N_r}$	$\mathbf{w}_{4N_r+1}$	$\mathbf{w}_{4N_r+2}$	$\mathbf{w}_{4N_r+3}$

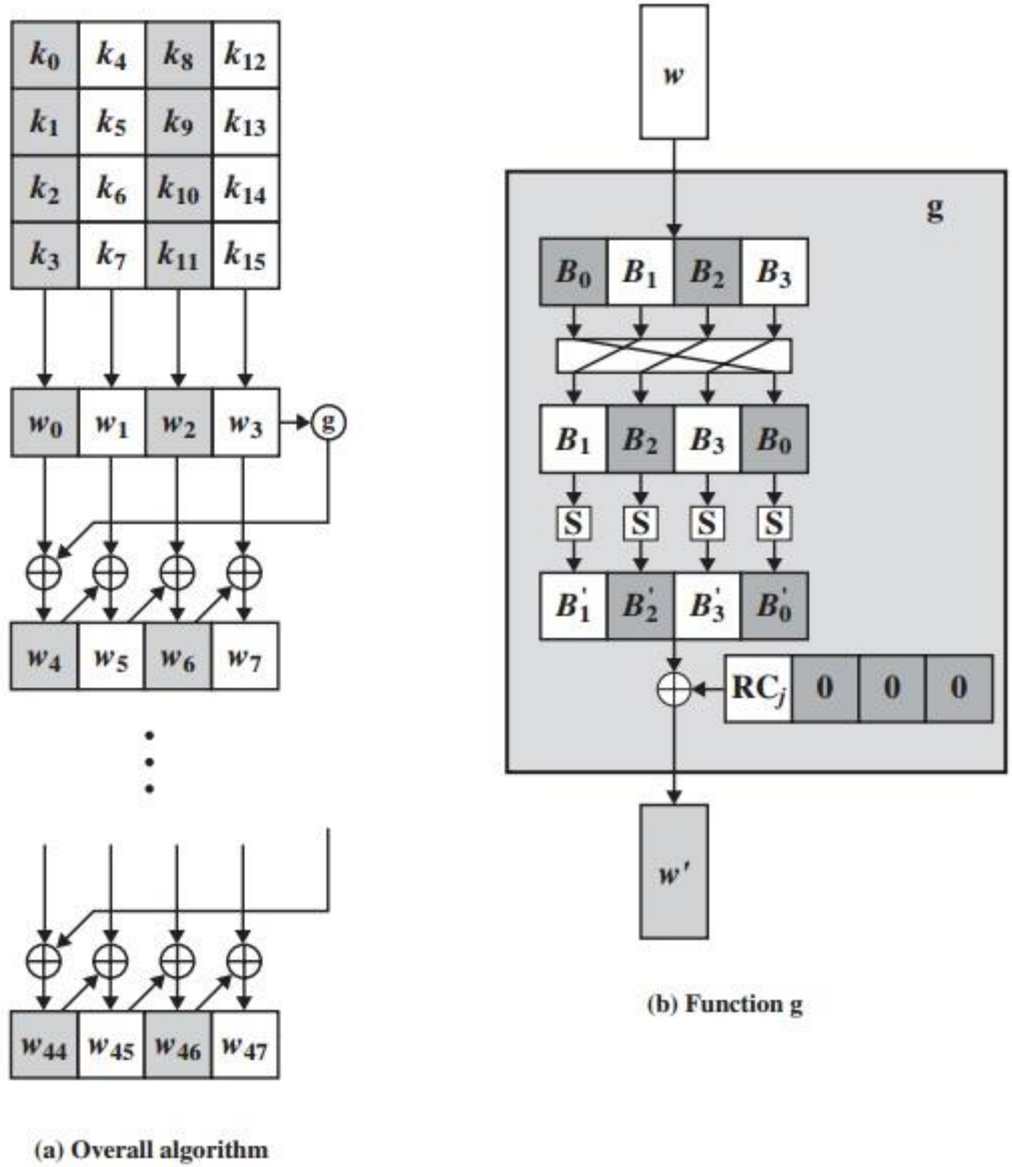
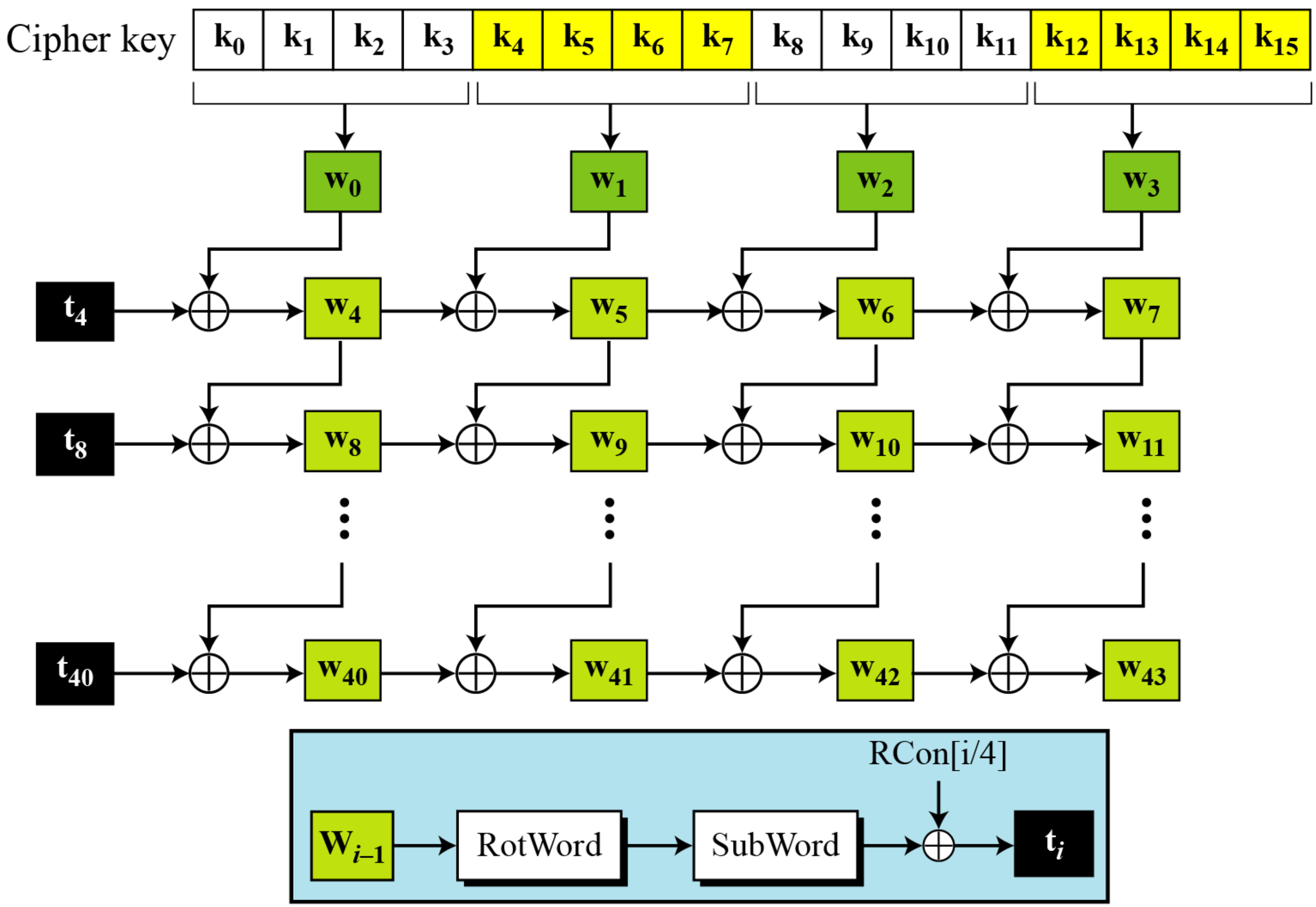


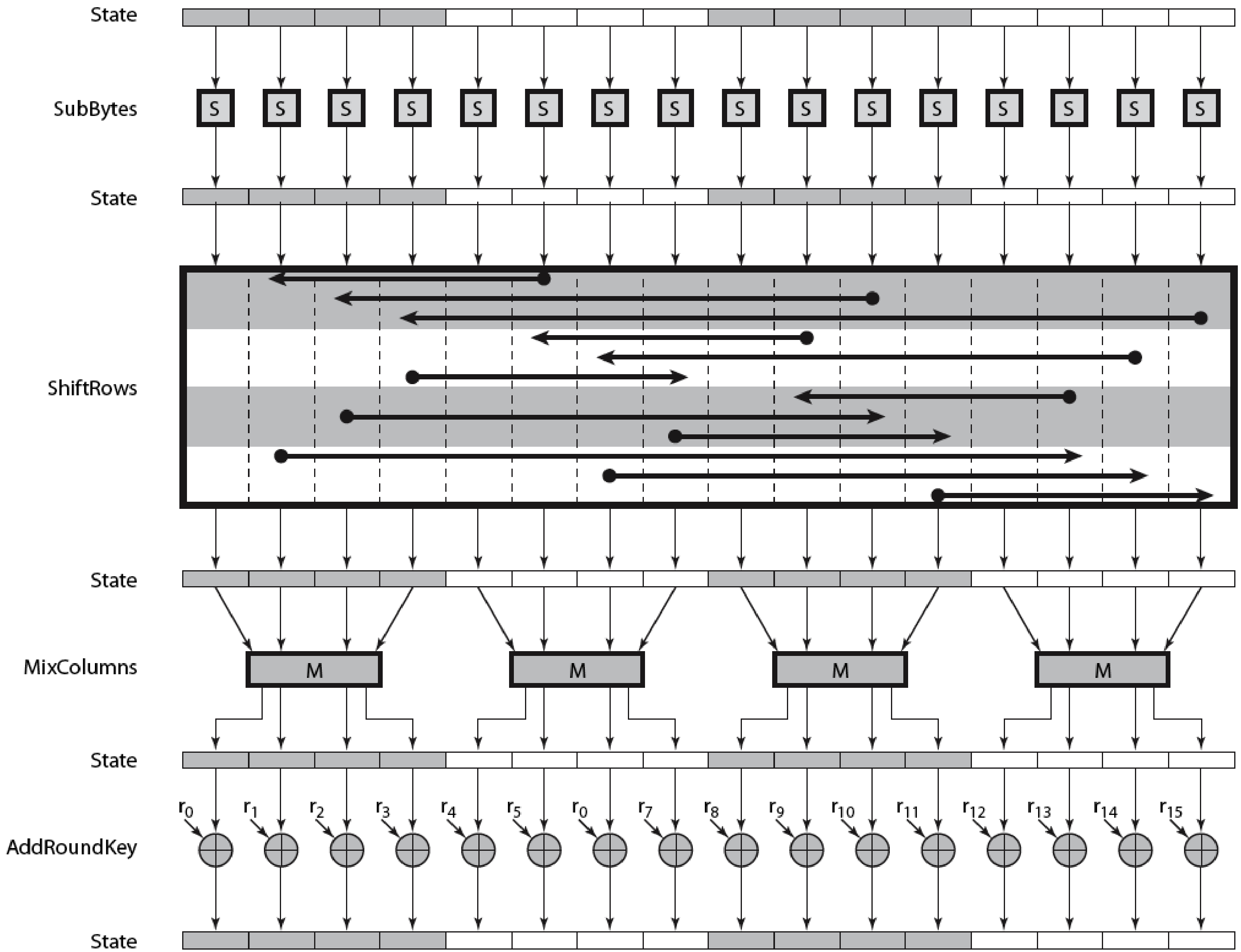
Figure 5.9 AES Key Expansion

Key Expansion Scheme



Making of t_i (temporary) words $i = 4\ N_r$

AES Round


State
SubBytes
S S S S S S S S S S S S S S S S
State
ShiftRows
State
MixColumns
M M M M
State
r_0 r_1 r_2 r_3 r_4 r_5 r_0 r_7 r_8 r_9 r_{10} r_{11} r_{12} r_{13} r_{14} r_{15}
AddRoundKey
State

- *The quick brown fox jumped over the lazy dog*
- 16 byte first block is like this

T	q	k	o
h	u		w
e	i	b	n
	c	r	

54	71	6b	6f
68	75	20	77
65	69	62	6e
20	63	72	20

AES Decryption

- AES decryption is not identical to encryption since steps done in reverse
- but can define an equivalent inverse cipher with steps as for encryption
  - but using inverses of each step
  - with a different key schedule
- works since result is unchanged when
  - swap byte substitution & shift rows
  - swap mix columns & add (tweaked) round key

Implementation Aspects

- can efficiently implemented on 8-bit CPU
  - byte substitution works on bytes using a table of 256 entries
  - shift rows is simple byte shifting
  - add round key works on byte XORs
  - mix columns requires matrix multiply in $GF(2^8)$ which works on byte values, can be simplified to use a table lookup

Implementation Aspects

- can efficiently implemented on 32-bit CPU
  - redefine steps to use 32-bit words
  - can precompute 4 tables of 256-words
  - then each column in each round can be computed using 4 table lookups + 4 XORs
  - at a cost of 16Kb to store tables
- designers believe this very efficient implementation was a key factor in its selection as the AES cipher

AES pseudo code

- function AESencrypt(plaintext, key)
- {

```
blocks := divideIntoBlocks(plaintext);
roundKeys = getRoundKeys(key)
for (block in blocks) { addRoundKey(roundKeys[0], block);
//intermediate rounds
for (8, 10 or 12 rounds) {
subBytes(block);
shiftRows(block);
mixColumns(block);
addRoundKey(roundKeys[..], block);
}
//last round
subBytes(block);
shiftRows(block);
addRoundKey(roundKeys[numRounds - 1], block); }
ciphertext := reassemble(blocks);
return ciphertext;}
```

DES and AES

- NIST FIPS 46 (DES – Jan 1977)
- FIPS 46-3 Triple DES (3DES) - applies the DES cipher algorithm three times to each data block
- NIST FIPS 197 (AES - November 2001) – request for update - **Classical security, Key size and Post quantum security and Implementation Security (Side Channel)**

Attacks on DES and AES

- Known attacks against DES include Brute-force, Linear crypt-analysis, and Differential crypt-analysis
- No known crypt-analytical attacks against AES but side channel attacks against AES implementations possible.
- Overall, AES is a well-vetted and secure encryption standard that offers a significant leap forward in protecting information

Moving forward

- **Cryptanalytic attacks.** The attacks rely on nature of the algorithm and also knowledge of the general characteristics of the plaintext
- Post-quantum cryptography: There's ongoing research in the field of post-quantum cryptography, which aims to develop algorithms resistant to attacks from powerful quantum computers

Post-quantum cryptography

- Quantum computers – machines that exploit quantum mechanical phenomena to solve mathematical problems that are difficult or intractable for conventional computers. If large-scale quantum computers are ever built, they will be able to break many of the public-key cryptosystems currently in use.
- This would seriously compromise the confidentiality and integrity of digital communications on the Internet and elsewhere.

Post-quantum cryptography

- The goal of post-quantum cryptography (also called quantum-resistant cryptography) is to develop cryptographic systems that are secure against both quantum and classical computers, and can interoperate with existing communications protocols and networks.
- Draft FIPS 203, Module-Lattice-Based Key-Encapsulation Mechanism Standard
- Draft FIPS 204, Module-Lattice-Based Digital Signature Standard
- Draft FIPS 205, Stateless Hash-Based Digital Signature Standard