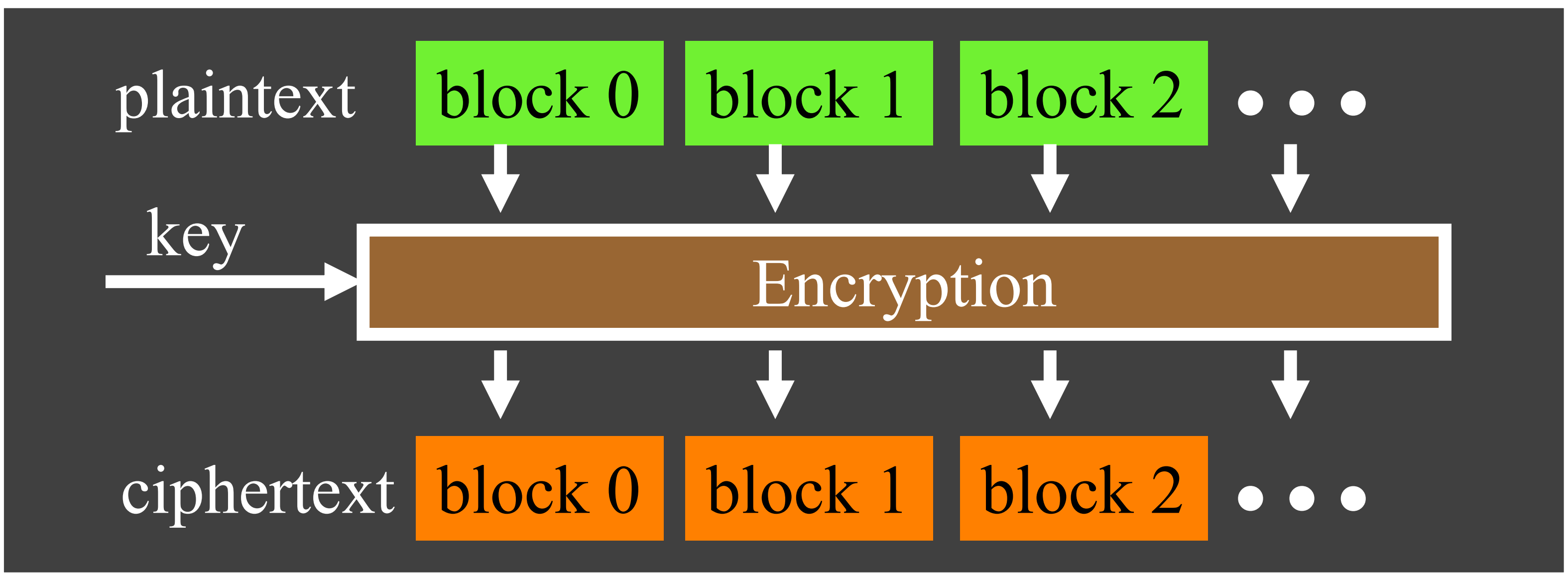
Generic Block Encryption

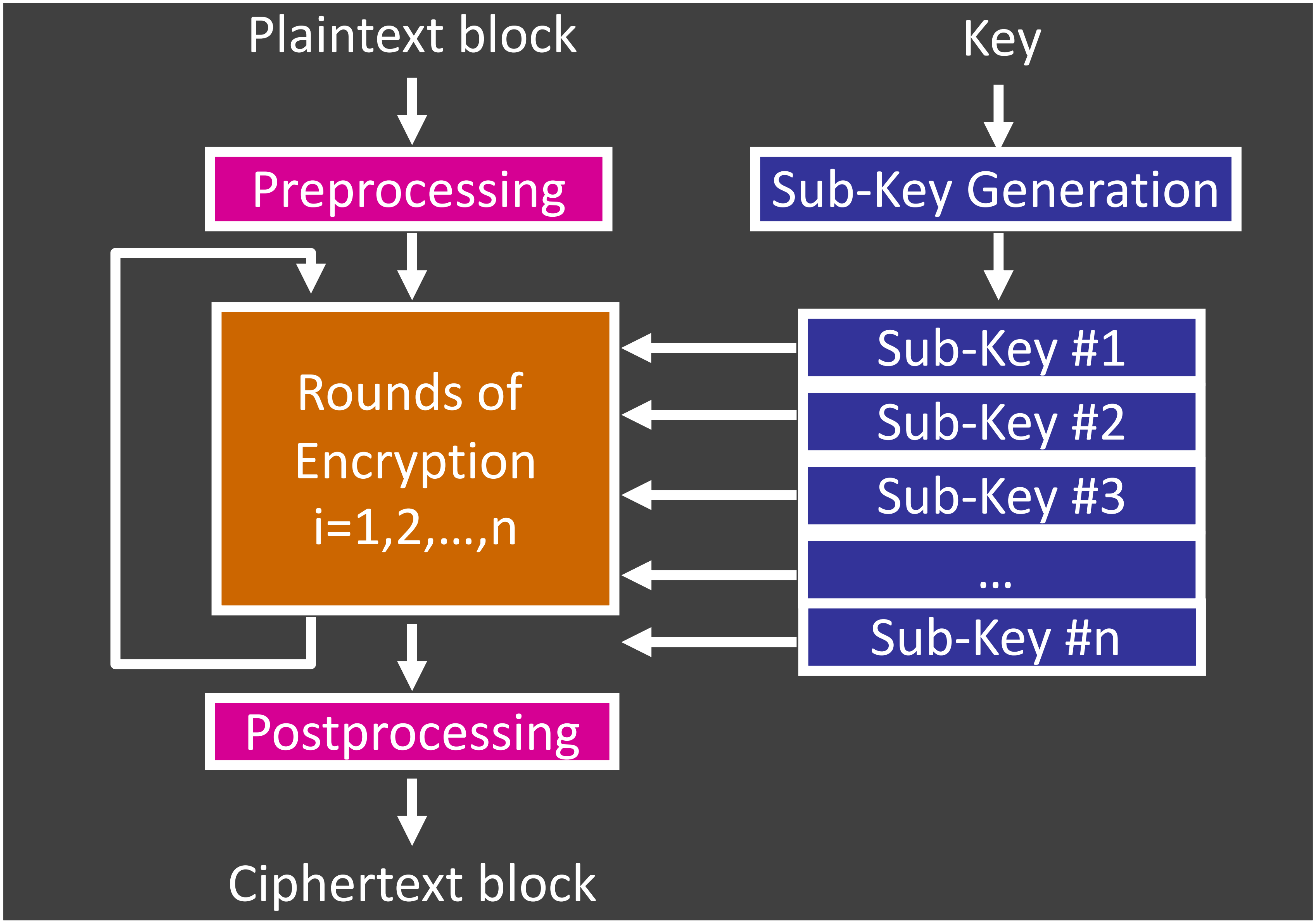
- Converts one input plaintext block of fixed size b bits to an output ciphertext block also of *b* bits
- Benefits of large *b*? of short *b*?



Two Principles for Cipher Design

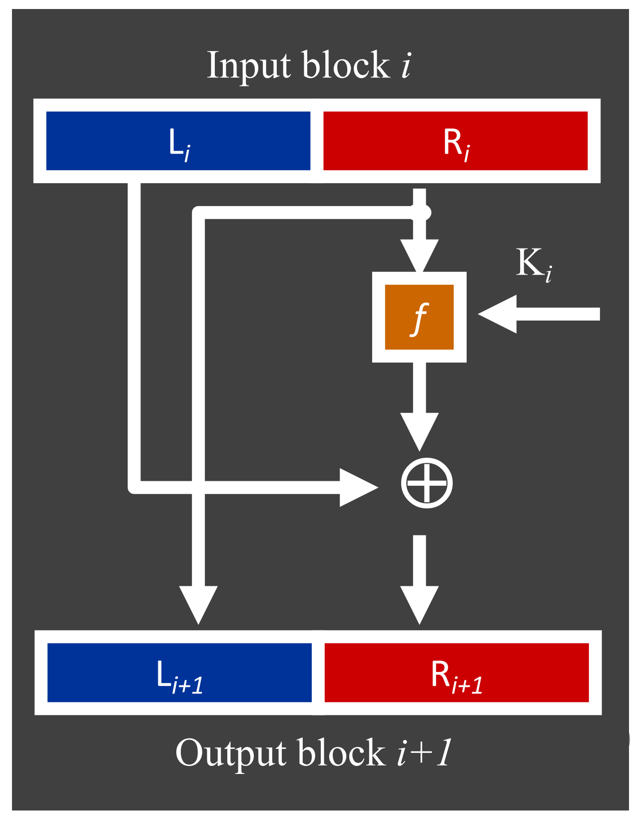
- *Confusion*: Make the relationship between the <plaintext, key> input and the <ciphertext> output as complex (non-linear) as possible
  - Mainly accomplished by *substitution*
- *Diffusion*: Spread the influence of each input bit across many output bits
  - Mainly accomplished by *permutation*
- Idea: use *multiple*, *alternating* permutations and subsitutions
  - *S→P→S→P→S→… or P→S→P→S→P→…*
  - Does it have to alternate?, e.g., S→S→S→P→P→P→S→S→…

Basic Form of Modern Block Ciphers



Feistel Cipher

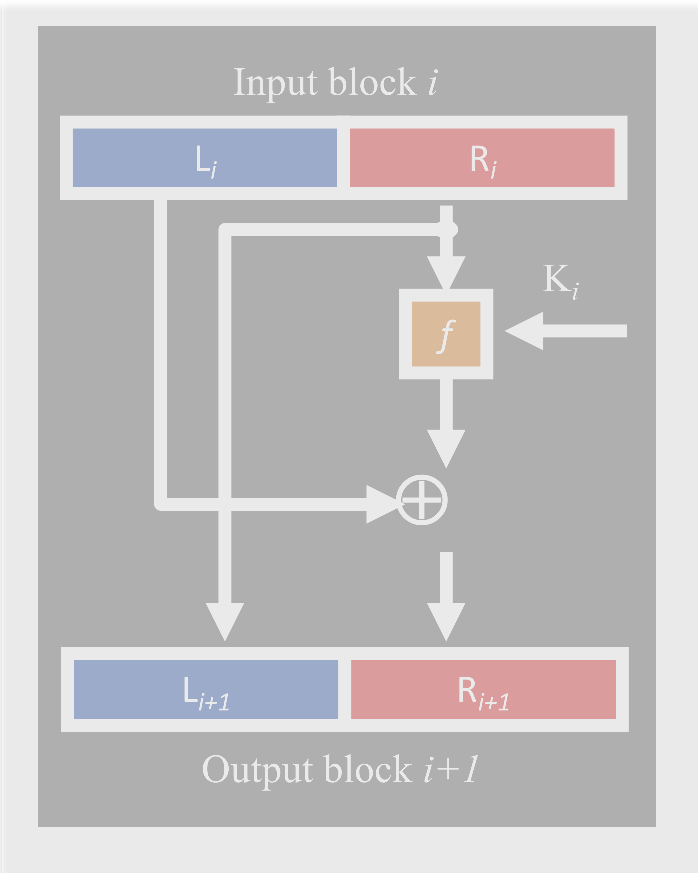
- Very influential "*template*" for designing block ciphers
- Major benefit: do encryption and decryption w/ same hardware
- One "round" of Feistel Encryption
    1. Break input block *i* into left and right halves L_i and R_i
    2. Copy R_i to create output half block L_{i+1}
    3. Half block R_i and key K_i are "scrambled" by function *f*
    4. XOR result with input half-block L_i to create output half-block R_{i+1}

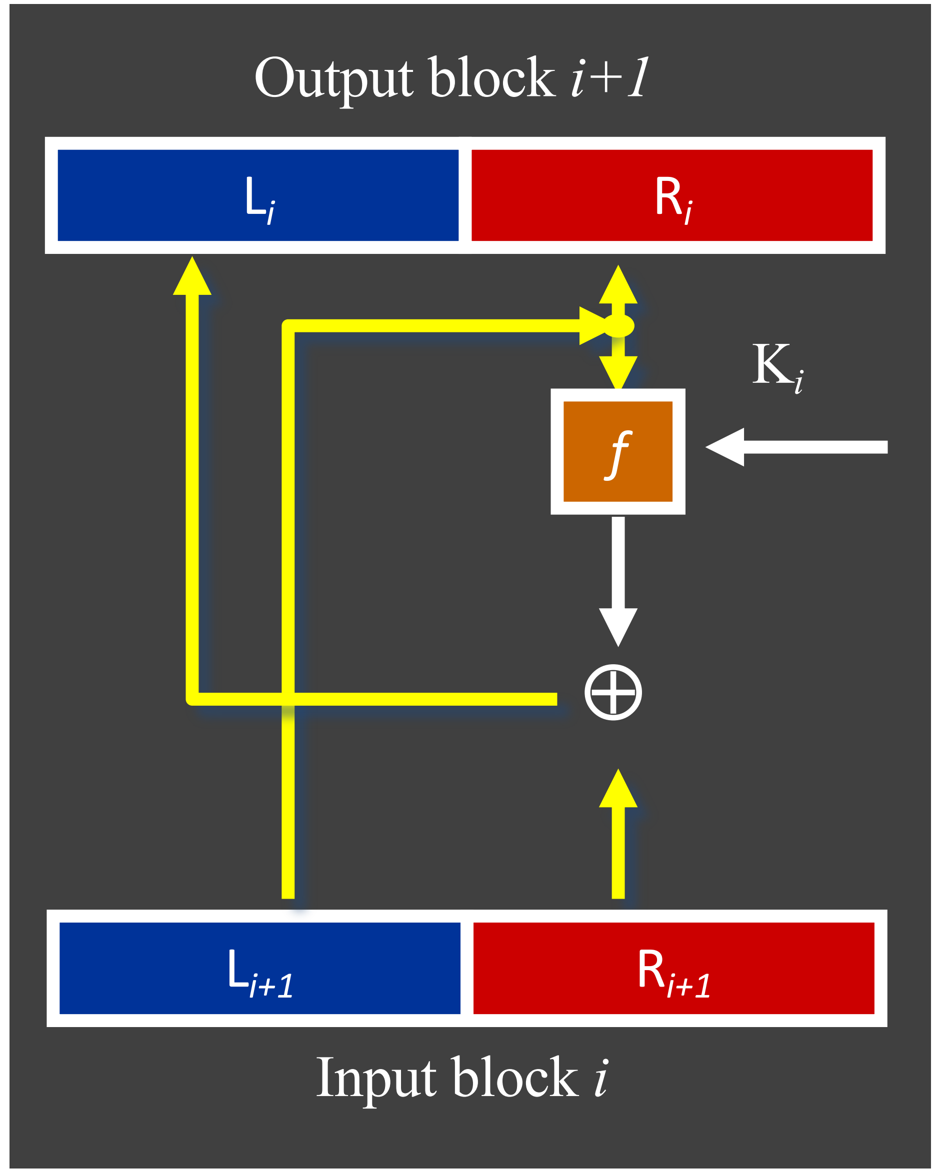


- f is generic
- Substitution (f) and permutation

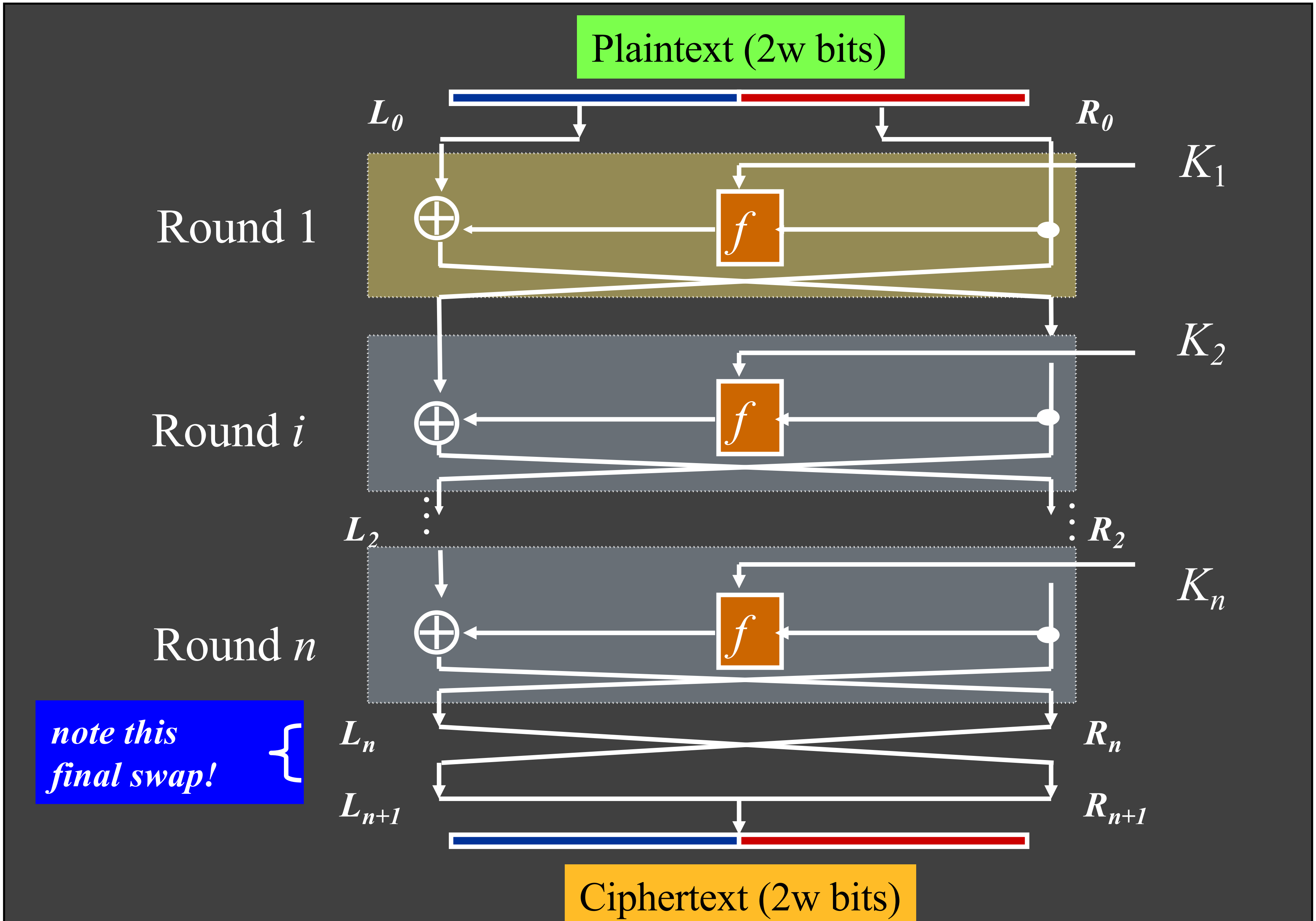
One “Round” of Feistel Decryption

- Just reverse the arrows!

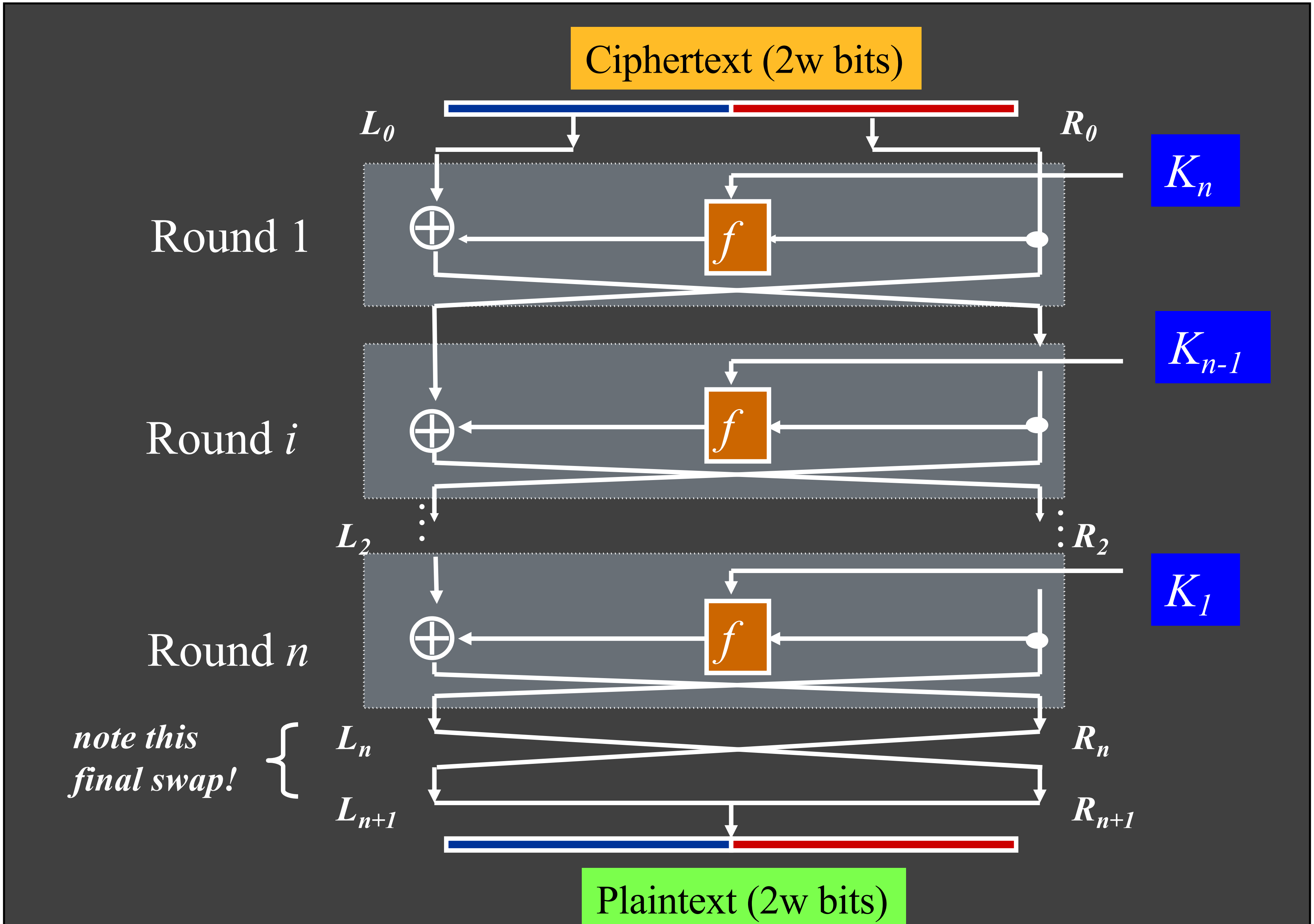




Complete Feistel Cipher: Encryption

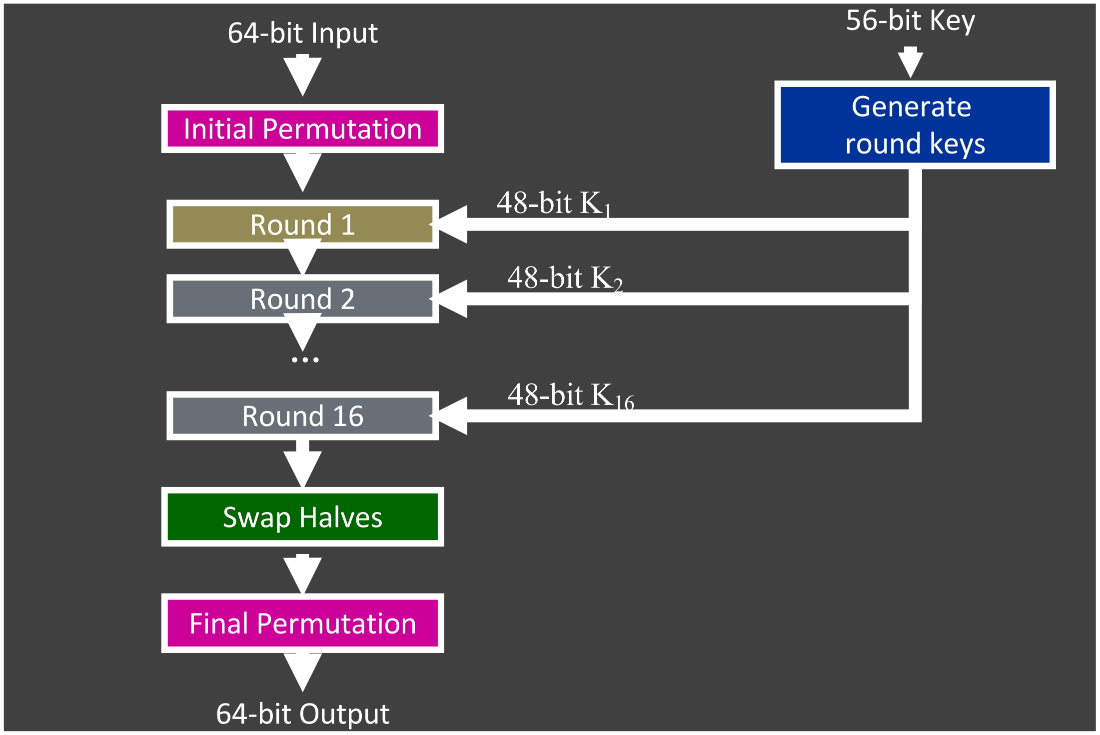


Feistel Cipher: Decryption

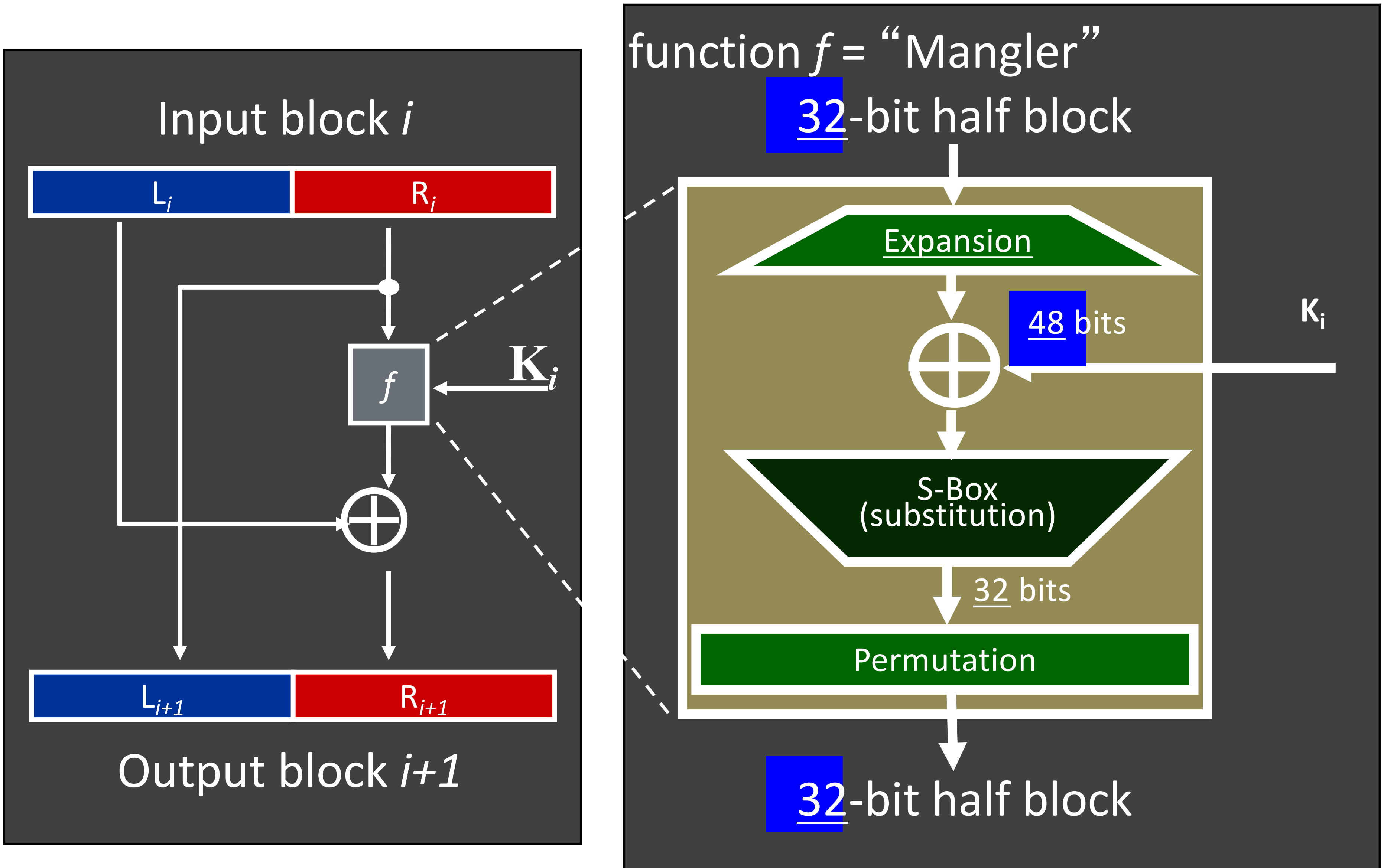


Data Encryption Standard (DES)

- Introduced by the US NBS (now NIST) in 1972
- Signaled the beginning of the modern area of cryptography
- Basics
  - Feistel Cipher
  - 8-byte (64 bit) input
  - 8-byte (64 bit) output
  - 8-byte key (56 bits + 8 parity bits)
  - 16 rounds



DES Round: *f* (Mangler) Function



Substitution Box (S-Box)

- A substitution box (or S-box) is used to obscure the relationship between the plaintext and the ciphertext
  - Shannon's property of *confusion*: the relationship between key and ciphertext is complex as possible
  - In DES, S-boxes are carefully chosen to resist cryptanalysis
  - Thus, that is where the security comes from

S_5		Middle 4 bits of input															
		0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
Outer bits	**00**	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

Example: Given a 6-bit input, the 4-bit output is found by selecting the row using the outer two bits, and the column using the inner four bits. For example, an input "011011" has outer bits "01" and inner bits "1101"; the corresponding output would be "1001".

Avalanche Effect in DES: Change in Plaintext

Round		δ
	02468aceeca86420 12468aceeca86420	1
1	3cf03c0fbad22845 3cf03c0fbad32845	1
2	bad2284599e9b723 bad3284539a9b7a3	5
3	99e9b7230bae3b9e 39a9b7a3171cb8b3	18
4	0bae3b9e42415649 171cb8b3ccaca55e	34
5	4241564918b3fa41 ccaca55ed16c3653	37
6	18b3fa419616fe23 d16c3653cf402c68	33
7	9616fe2367117cf2 cf402c682b2cefbc	32
8	67117cf2c11bfc09 2b2cefbc99f91153	33

Round		δ
9	c11bfc09887fbc6c 99f911532eed7d94	32
10	887fbc6c600f7e8b 2eed7d94d0f23094	34
11	600f7e8bf596506e d0f23094455da9c4	37
12	f596506e738538b8 455da9c47f6e3cf3	31
13	738538b8c6a62c4e 7f6e3cf34bc1a8d9	29
14	c6a62c4e56b0bd75 4bc1a8d91e07d409	33
15	56b0bd7575e8fd8f 1e07d4091ce2e6dc	31
16	75e8fd8f25896490 1ce2e6dc365e5f59	32
IP–1	da02ce3a89ecac3b 057cde97d7683f2a	32

(from Stallings, Crypto and Net Security)

Avalanche Effect in DES: Change in Key

Round		δ
	02468aceeca86420 02468aceeca86420	0
1	3cf03c0fbad22845 3cf03c0f9ad628c5	3
2	bad2284599e9b723 9ad628c59939136b	11
3	99e9b7230bae3b9e 9939136b768067b7	25
4	0bae3b9e42415649 768067b75a8807c5	29
5	4241564918b3fa41 5a8807c5488dbe94	26
6	18b3fa419616fe23 488dbe94aba7fe53	26
7	9616fe2367117cf2 aba7fe53177d21e4	27
8	67117cf2c11bfc09 177d21e4548f1de4	32

Round		δ
9	c11bfc09887fbc6c 548f1de471f64dfd	34
10	887fbc6c600f7e8b 71f64dfd4279876c	36
11	600f7e8bf596506e 4279876c399fdc0d	32
12	f596506e738538b8 399fdc0d6d208dbb	28
13	738538b8c6a62c4e 6d208dbbb9bdeeaa	33
14	c6a62c4e56b0bd75 b9bdeeaad2c3a56f	30
15	56b0bd7575e8fd8f d2c3a56f2765c1fb	33
16	75e8fd8f25896490 2765c1fb01263dc4	30
IP–1	da02ce3a89ecac3b ee92b50606b62b0b	30

(from Stallings, Crypto and Net Security)

Cryptanalysis of DES

- DES has an effective 56-bit key length
- Wiener: $1,000,000 - 3.5 hours (never built)
- July 17, 1998, the EFF DES Cracker, which was built for less than $250,000 < 3 days
- January 19, 1999, Distributed.Net (w/EFF), 22 hours and 15 minutes (over many machines)
- We all assume that NSA and agencies like it around the world can crack (recover key) DES in milliseconds

- *What now? Give up on DES?*

Variants of DES

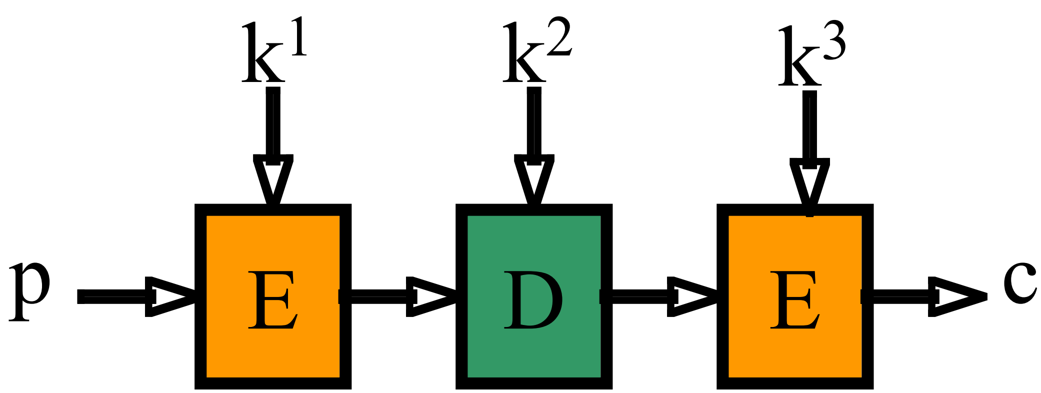
- DESX (XOR with separate keys ~= 60-bits)

$$DESX(m) = K_2 \oplus DES_K(m \oplus K_1)$$

  - Linear cryptanalysis
- Triple DES (three keys ~= 112 bits)
  - Keys k_1, k_2, k_3, but in practice $k_1 = k_3$

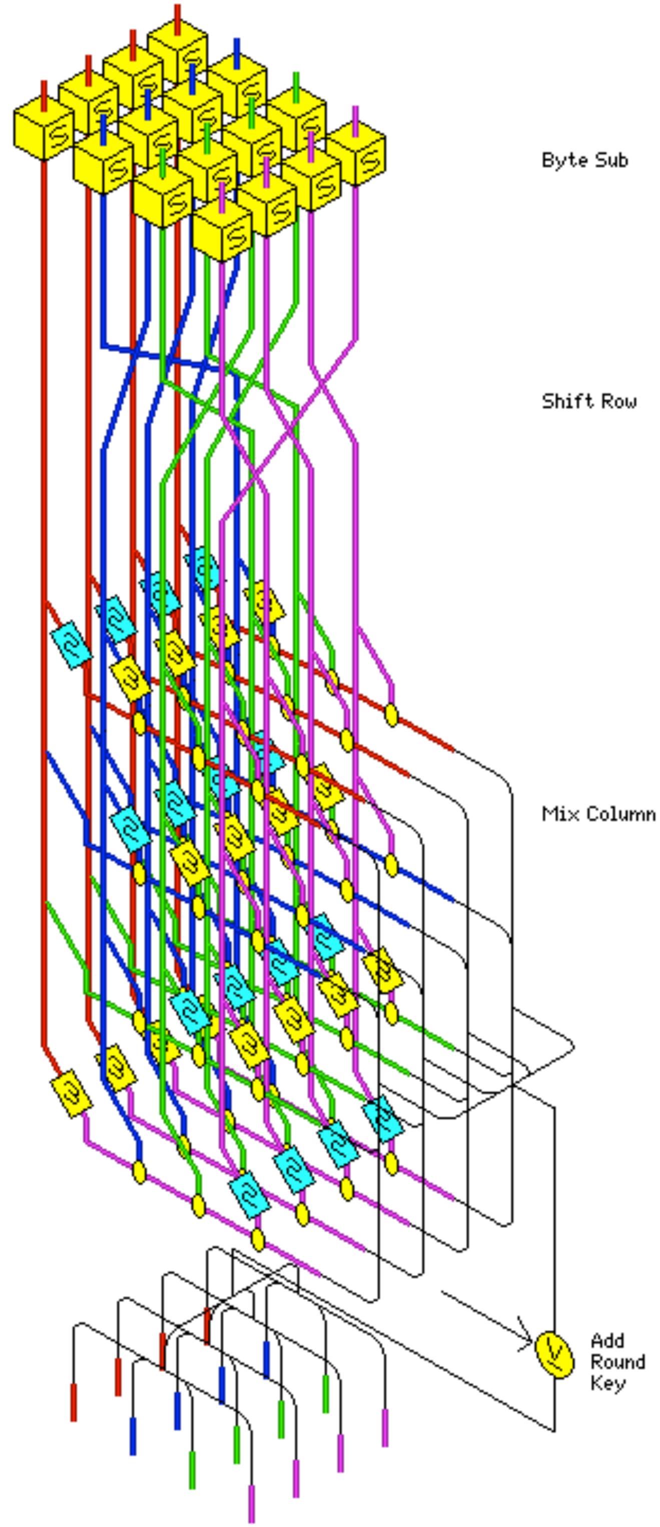
$$C \;=\; E(D(E(p, k1), k2), k3)$$

  - Compatible with normal DES if $k_1 = k_2 = k_3$

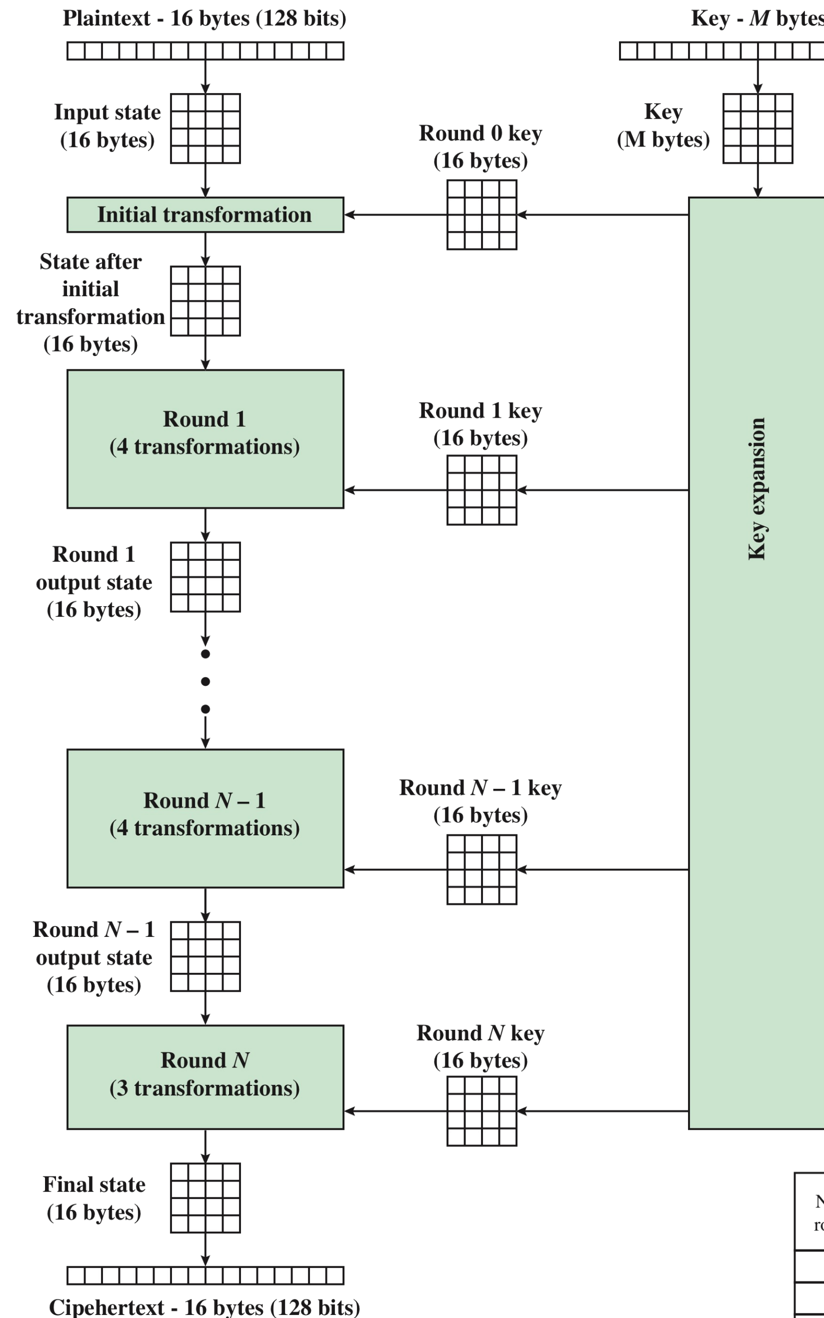


Advanced Encryption Standard (AES)

- International NIST bakeoff between cryptographers
  - Rijndael (pronounced "Rhine-dall")
- Replaced DES as the "accepted" symmetric key cipher
  - Substitution-permutation network, not a Feistel network
  - Variable key lengths (128, 192, or 256 bits)
  - Block size: 128 bits
  - Fast implementation in both hardware and software
  - Small code and memory footprint



AES Encryption Process

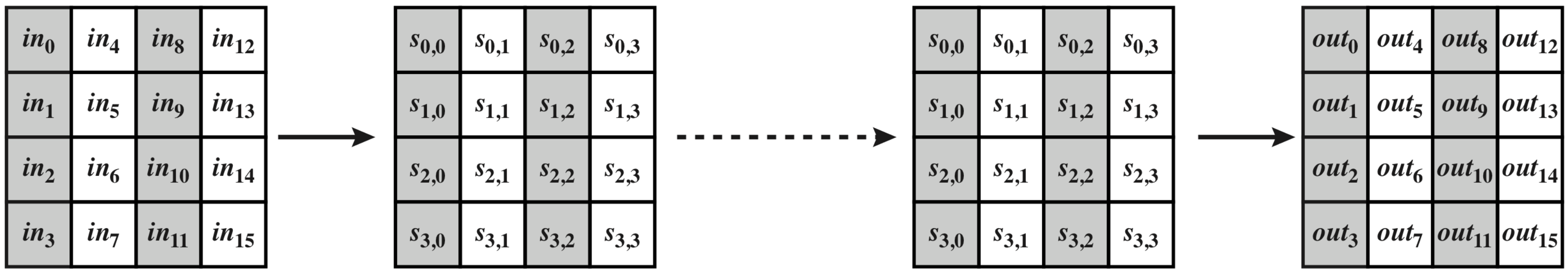


No. of rounds	Key Length (bytes)
10	16
12	24
14	32

Figure 5.1 AES Encryption Process

(from Stallings, Crypto and Net Security)

AES Data Structures



(a) Input, state array, and output

(b) Key and expanded key

Figure 5.2 AES Data Structures

(from Stallings, Crypto and Net Security)

AES Encryption and Decryption

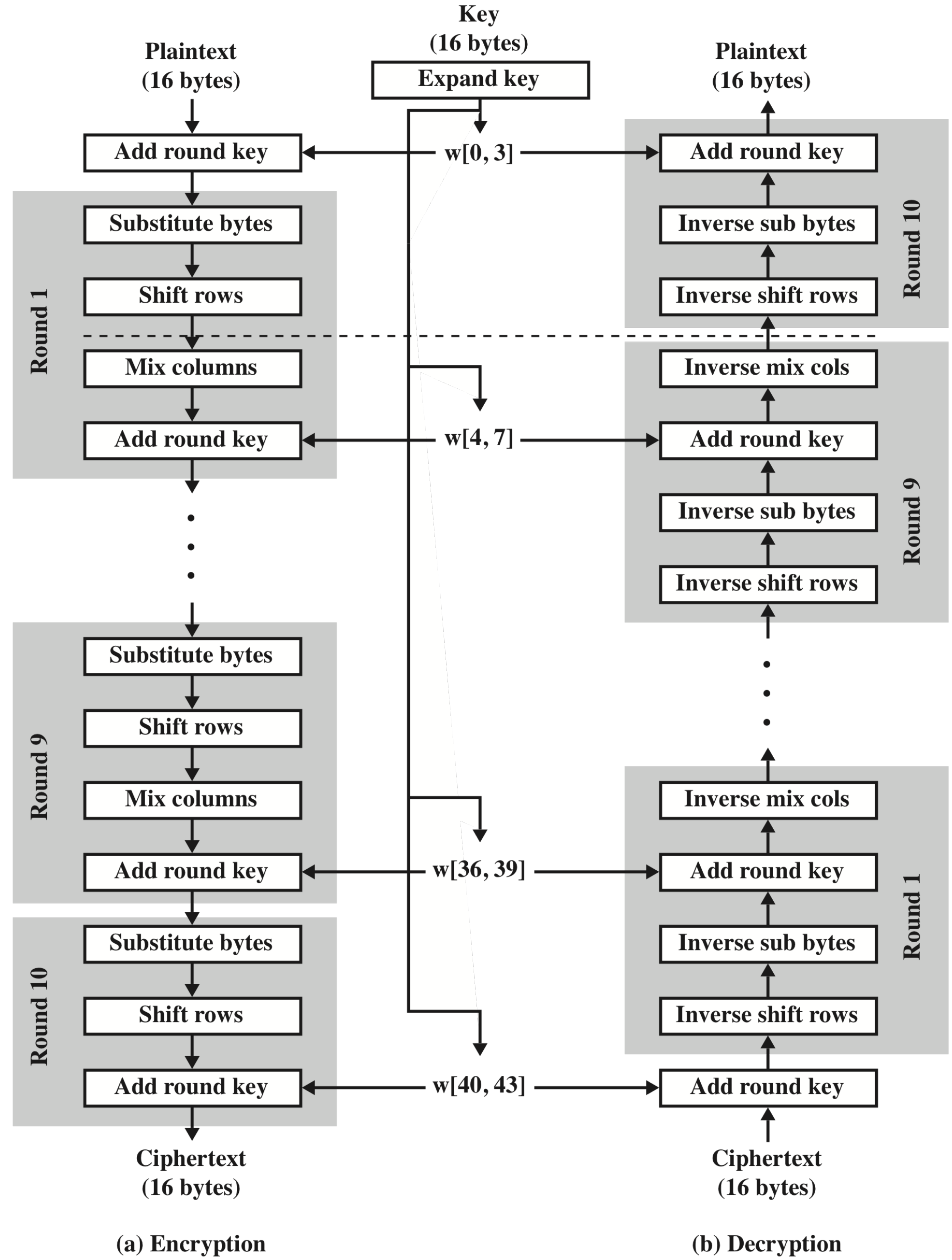


Figure 5.3 AES Encryption and Decryption

(from Stallings, Crypto and Net Security)

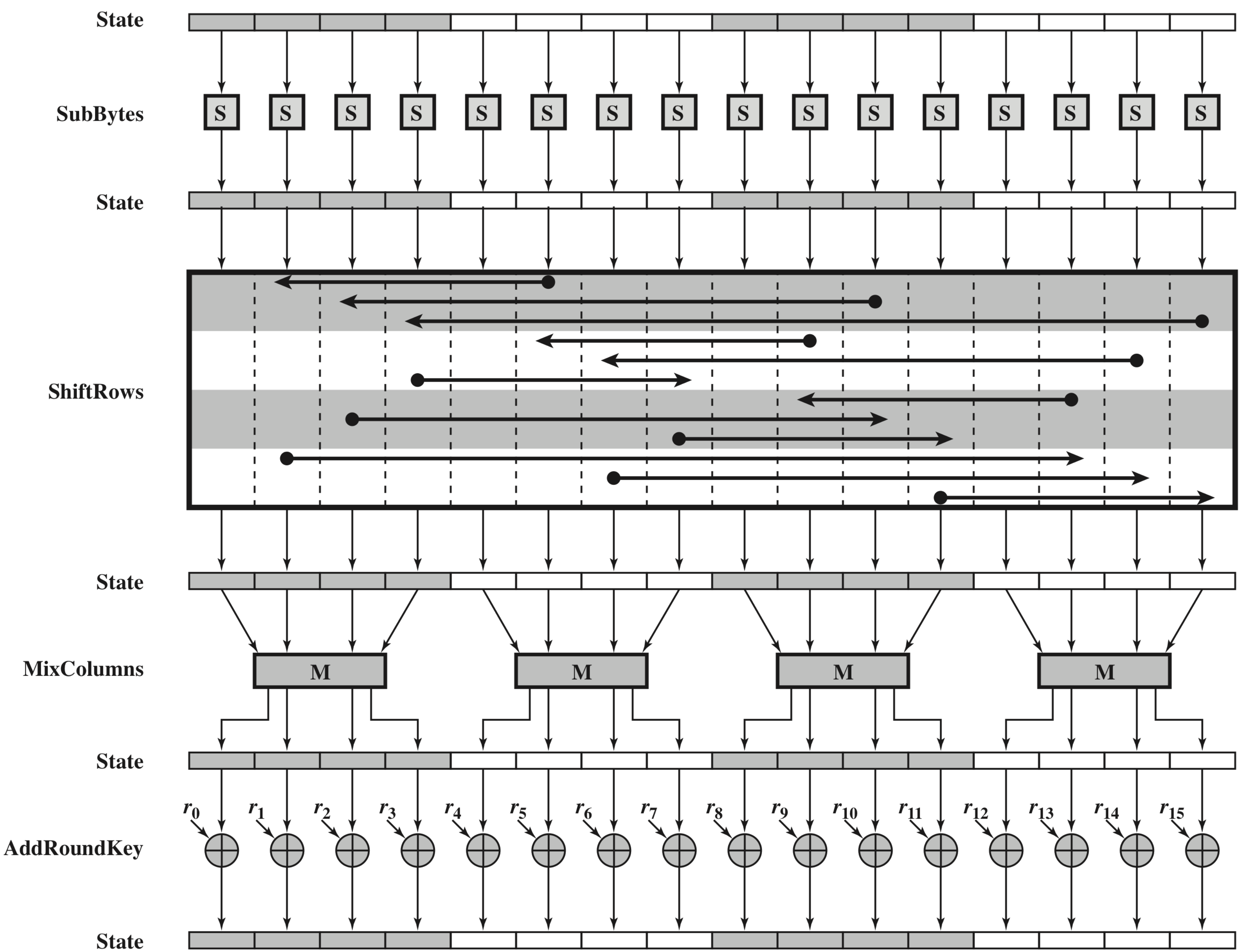


Figure 5.4 AES Encryption Round

(from Stallings, Crypto and Net Security)

S-box and Inverse S-box

S-box

		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	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
	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

Inverse S-box

		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	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
	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

(from Stallings, Crypto and Net Security)

Implementation Aspects

- AES can be implemented very efficiently on an 8-bit processor
- SubBytes operates at the byte level and only requires a table of 256 bytes
- ShiftRows is a simple byte-shifting operation
- AddRoundKey is a bytewise XOR operation
- MixColumns requires matrix multiplication in the field $GF(2^8)$, which means all operations are carried out on bytes
- Designers believe this very efficient implementation was a key factor in its selection as the AES cipher

(from Stallings, Crypto and Net Security)

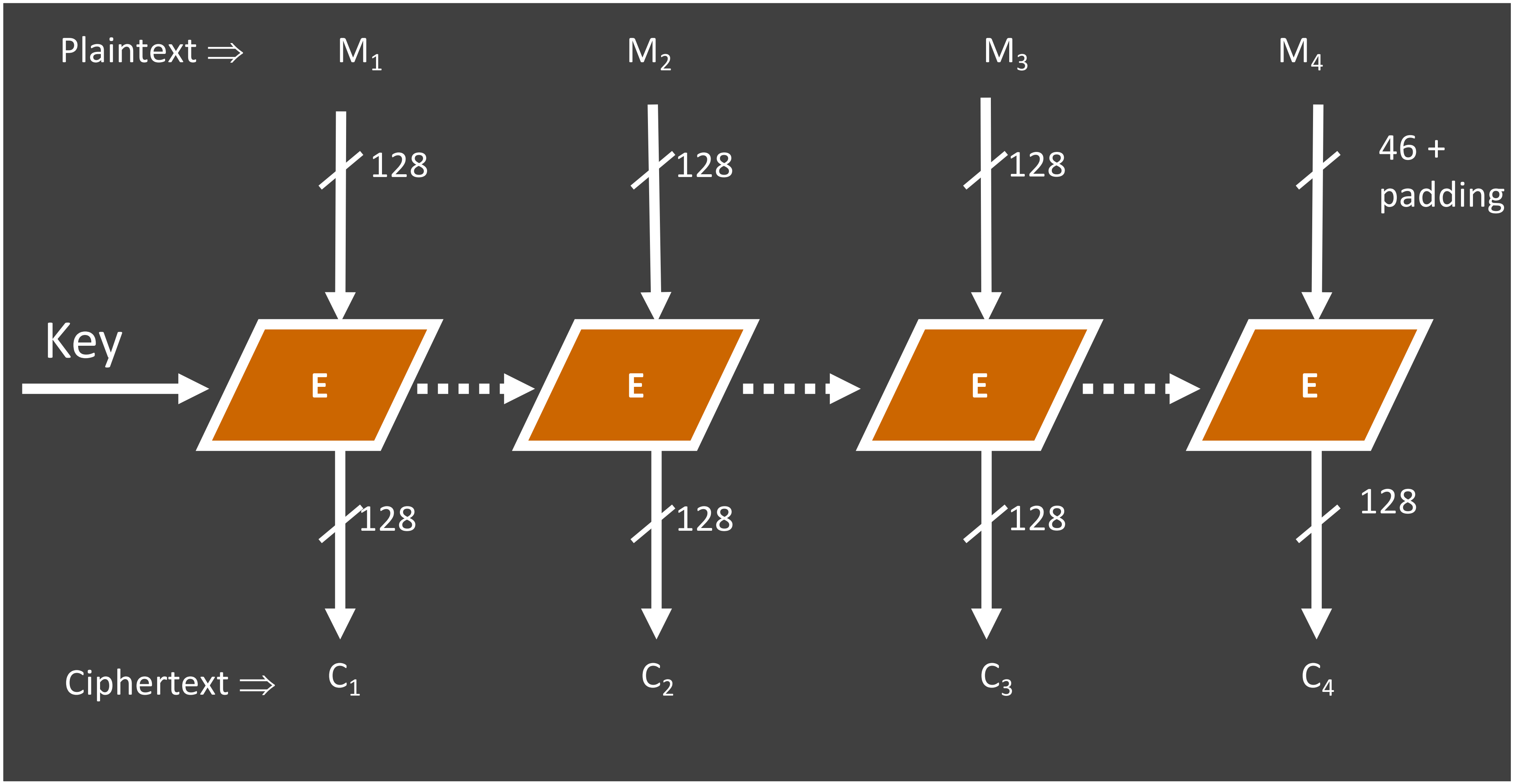
Modes of Operation

- Most ciphers work on blocks of fixed (small) size
- How to encrypt long messages?
- Modes of operation
  - ECB (Electronic Code Book)
  - CBC (Cipher Block Chaining)
  - OFB (Output Feedback)
  - CFB (Cipher Feedback)
  - CTR (Counter)

Issues for Block Chaining Modes

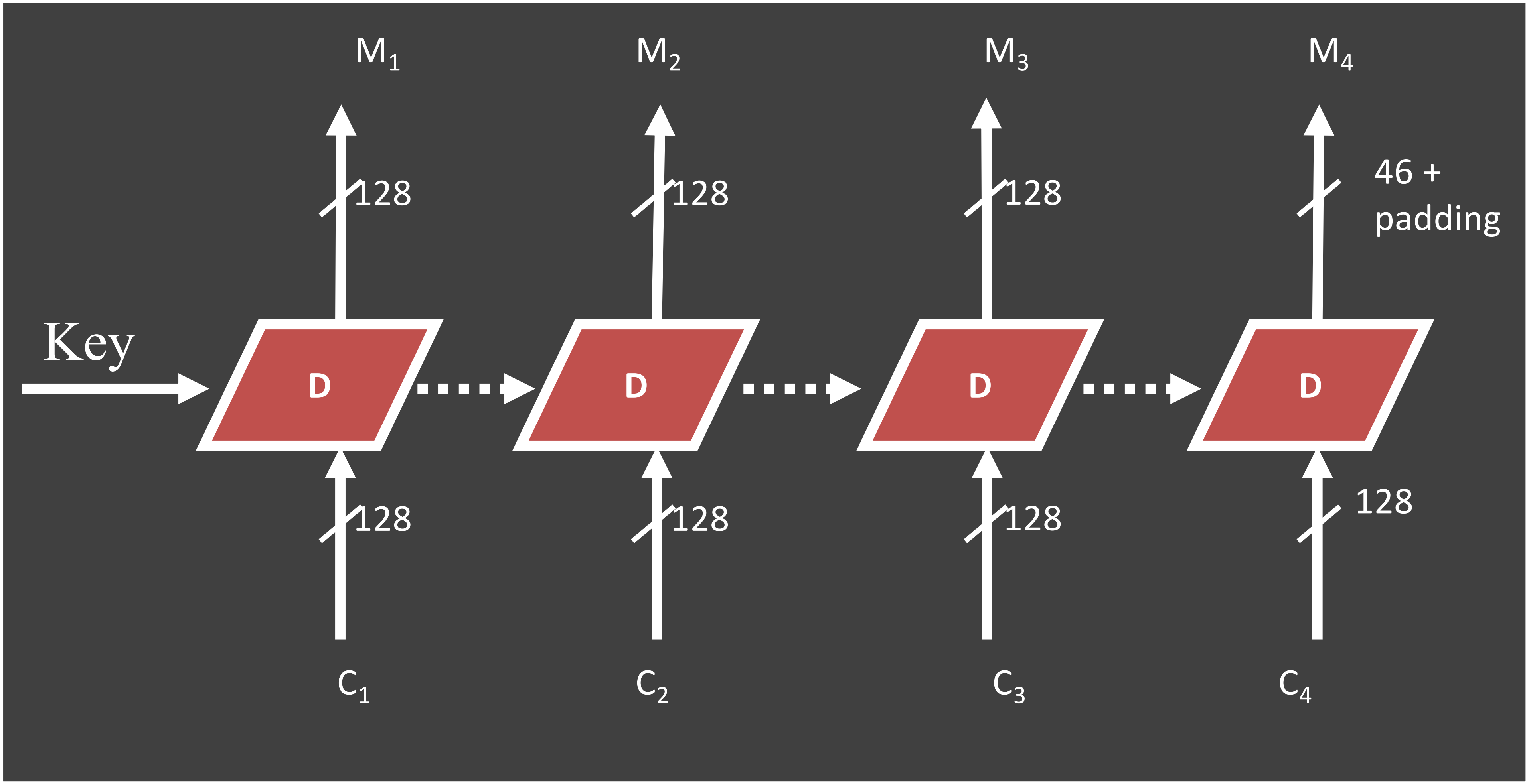
- *Information leakage*: Does it reveal info about the plaintext blocks?
- *Ciphertext manipulation*: Can an attacker modify ciphertext block(s) in a way that will produce a predictable/desired change in the decrypted plaintext block(s)?
  - Note: assume the structure of the plaintext is known, e.g., first block is employee #1 salary, second block is employee #2 salary, etc.
- *Parallel/Sequential*: Can blocks of plaintext (ciphertext) be encrypted (decrypted) in parallel?
- *Error Propagation*: If there is an error in a plaintext (ciphertext) block, will there be an encryption (decryption) error in more than one ciphertext (plaintext) block?

Electronic Code Book (ECB)



- The easiest mode of operation; each block is independently encrypted

ECB Decryption

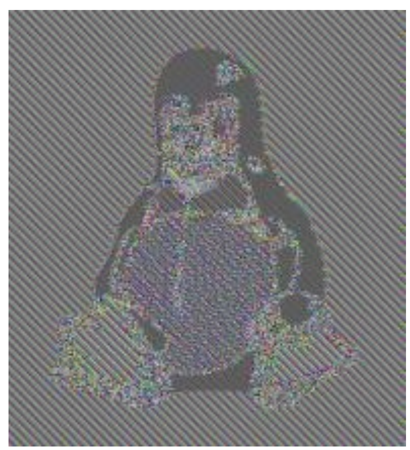


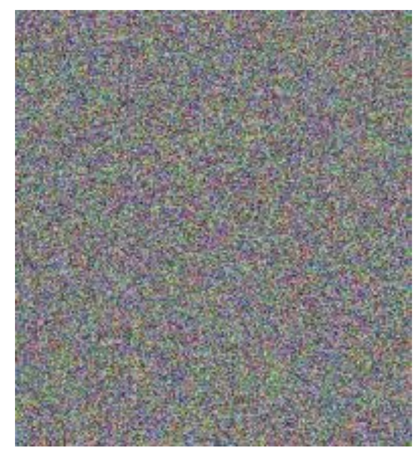
- Each block is independently decrypted

ECB Issues

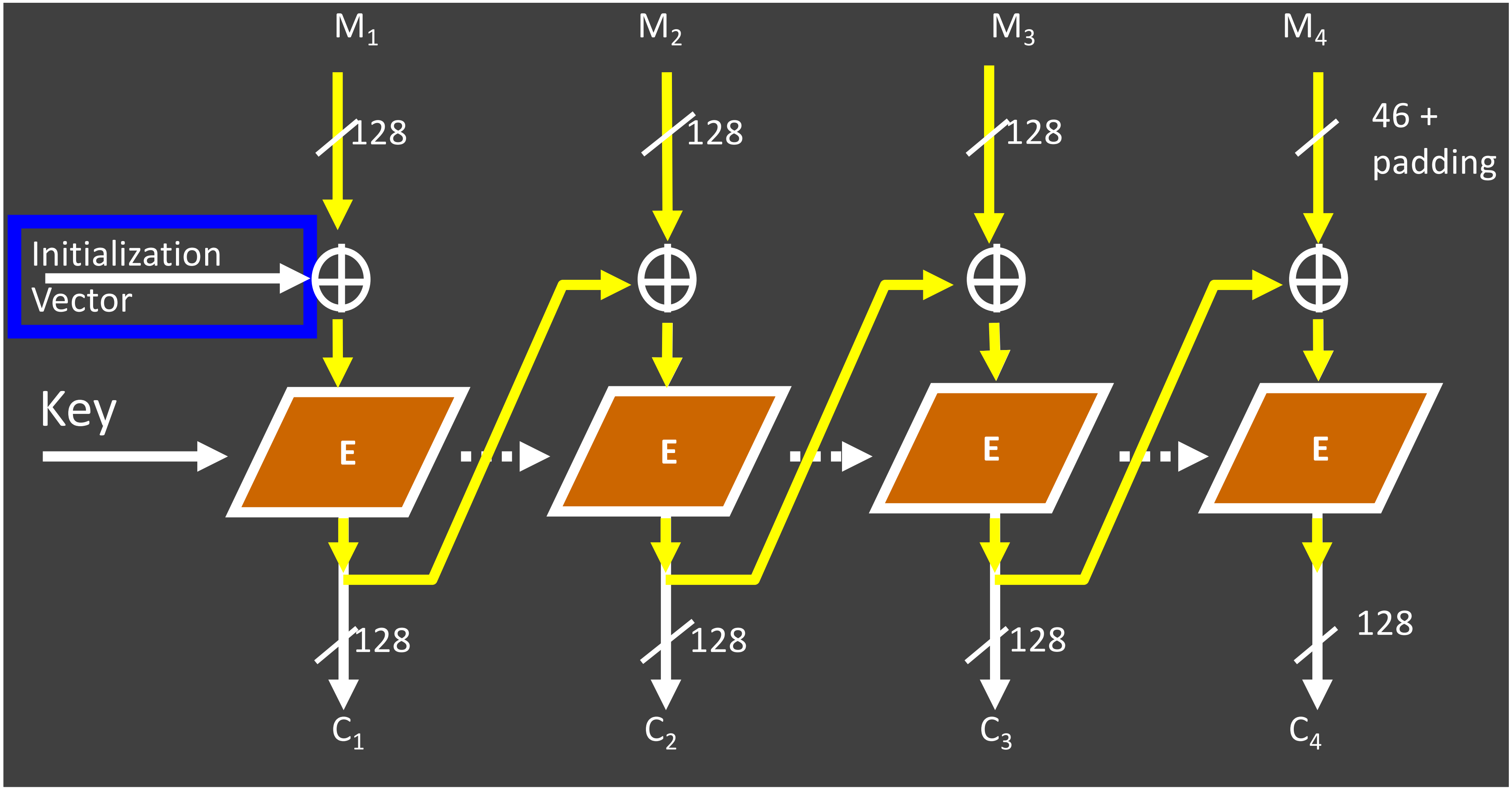
- *Information leaks*: two ciphertext blocks that are the same
- *Manipulation*: switch ciphertext with predictable results on plaintext (e.g., shuffle).
- *Parallel*: yes
- *Propagate*: no


Plaintext


ECB


Other modes

Cipher Block Chaining (CBC)

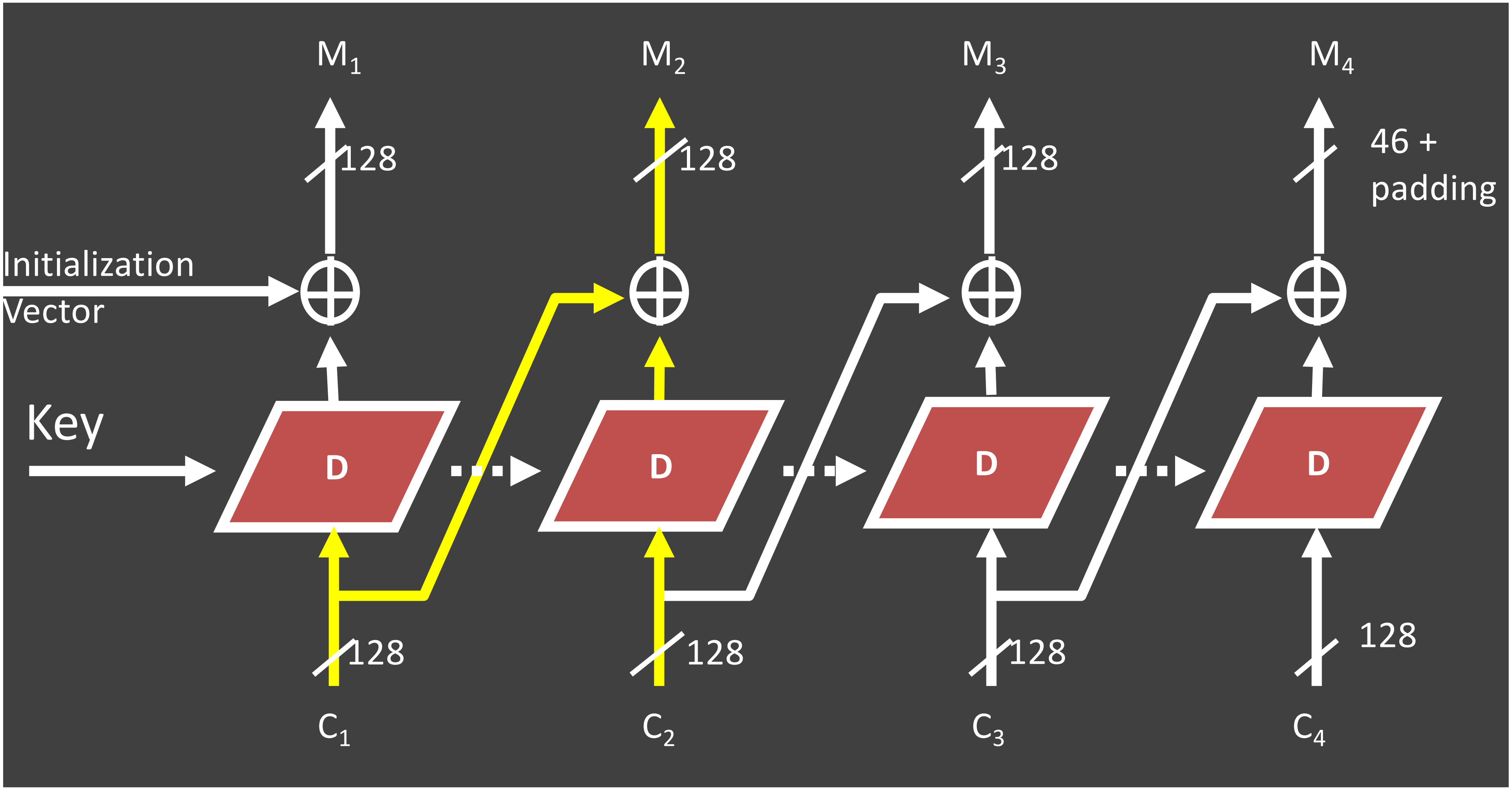


- Chaining dependency: each ciphertext block depends on all preceding plaintext blocks

Initialization Vectors

- Initialization Vector (IV)
  - Used along with the key; not secret
  - For a given plaintext, changing either the key, or the IV, will produce a different ciphertext
  - Why is that useful?
- IV generation and sharing
  - Random; may transmit with the ciphertext
  - Incremental; predictable by receivers

CBC Decryption

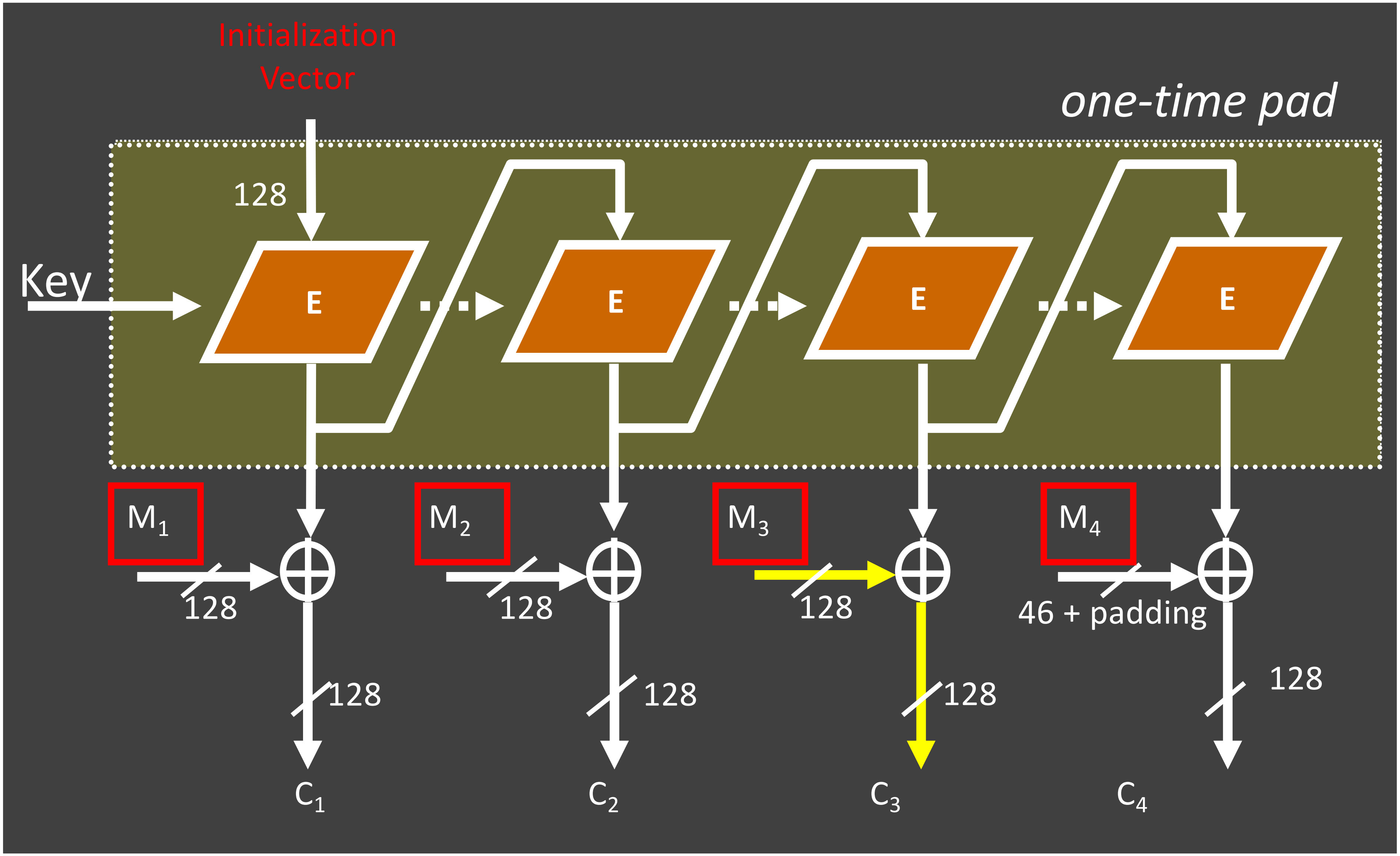


- How many ciphertext blocks does each plaintext block depend on?

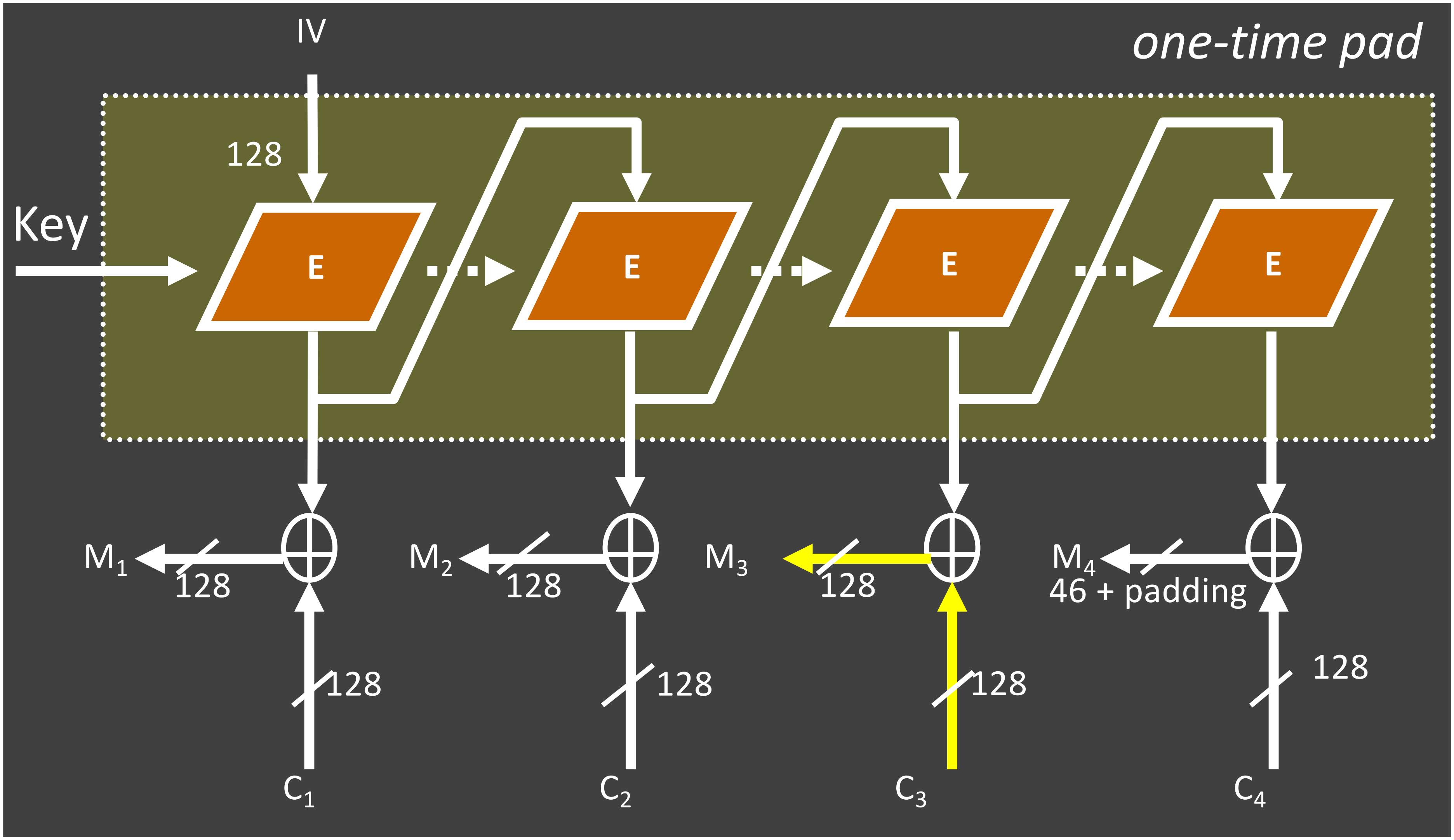
CBC Properties

- Does information leak?
  - Identical plaintext blocks will produce different ciphertext blocks
- Can ciphertext be manipulated profitably?
  - ???
- Parallel processing possible?
  - no (encryption), yes (decryption)
- Do ciphertext errors propagate?
  - yes (encryption), a little (decryption)

Output Feedback Mode (OFB)



OFB Decryption



No block decryption required!

OFB Properties

- Does information leak?
  - identical plaintext blocks produce different ciphertext blocks
- Can ciphertext be manipulated profitably?
  - ???
- Parallel processing possible?
  - no (generating pad), yes (XORing with blocks)
- Do ciphertext errors propagate?
  - ???

OFB … (Cont’d)

- If you know one plaintext/ciphertext pair, can easily derive the one-time pad that was used
  - i.e., should not reuse a one-time pad!
- Conclusion: IV must be different every time