Next Generation of Block Ciphers Providing High-Level Security

May 7th, 2014

Roman Oliynykov

Associated Professor at
Information Technologies Security Department
Kharkov National University of Radioelectronics

Head of Scientific Research Department
JSC “Institute of Information Technologies”
Ukraine

Visiting professor at
Samsung Advanced Technology Training Institute
Korea
ROliynykov@gmail.com

Outline

- Block cipher basics, overview of their application
- Requirements to block ciphers and their construction principles
- Basics of cryptanalysis: differential, linear, etc.
- Advanced Encryption Standard: construction, advantages and disadvantages
- Directions of block ciphers further development: lightweight and high-level security
- Newly developed block ciphers providing high level security: solutions from the USA, Russia, Belorussia and Ukraine
- Construction and properties of perspective cipher for Ukraine, speed comparison
- Beyond block cipher security: can encryption be broken if we use high-level strength cipher?

About myself (I)

- I'm from Ukraine (Eastern part of Europe),
  host country of Euro2012 football championship

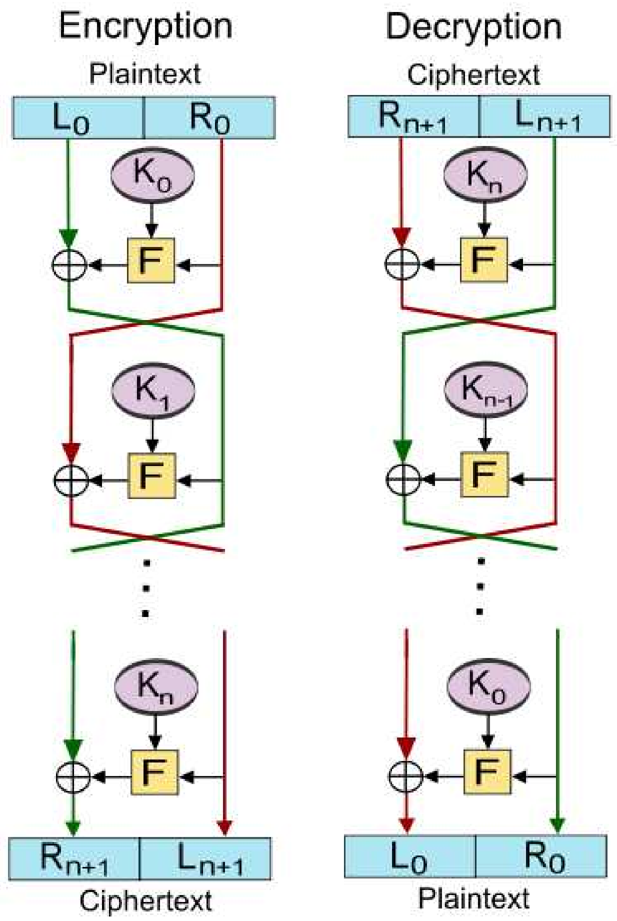
- I live in Kharkov (the second largest city in the country, population is 1.5 million people), Eastern Ukraine (near Russia),
  former capital of the Soviet Ukraine (1918-1934)
  three Nobel prize winners worked at Kharkov National University

About myself (II)

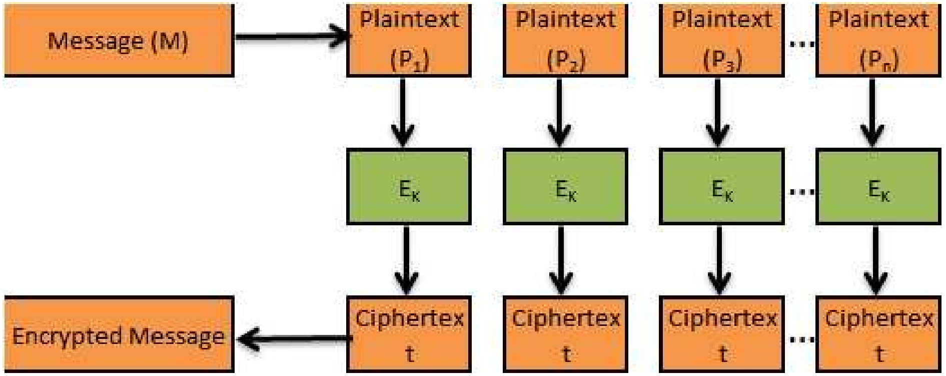
- Associated professor at Information Technologies Security Department at Kharkov National University of Radioelectronics
  - courses on computer networks and operation system security, special mathematics for cryptographic applications
- Head of Scientific Research Department at JSC “Institute of Information Technologies”
  - Scientific interests: symmetric cryptographic primitives synthesis and cryptanalysis
- Visiting professor at Samsung Advanced Technology Training Institute
  - courses on computer networks and operation system security, software security, effective application and implementation of symmetric cryptography

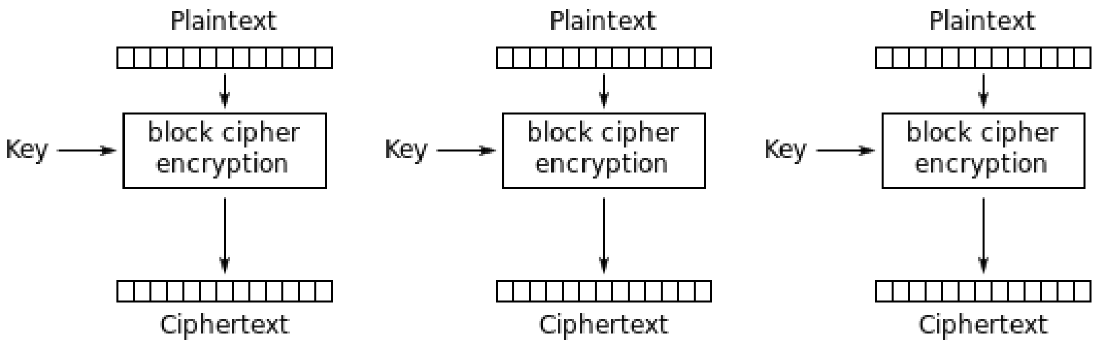
Block ciphers

- one of the most popular cryptographic transformations
- most widely used cryptographic algorithms for providing confidentiality in commercial systems
- symmetric key cryptographic transformation
- very often used as main construction element for hash functions, pseudo random number generators (PRNG), etc.



Block cipher encryption: electronic codebook mode (ECB)





NB: ECB mode is only used as basic block for more complex transformations

Block cipher

- Encryption

$$E_K(P) := E(K,P) : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$$

- Decryption

$$E_K^{-1}(C) := D_K(C) = D(K,C) : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$$

- Main property for practical implementation

$$\forall K : D_K(E_K(P)) = P$$

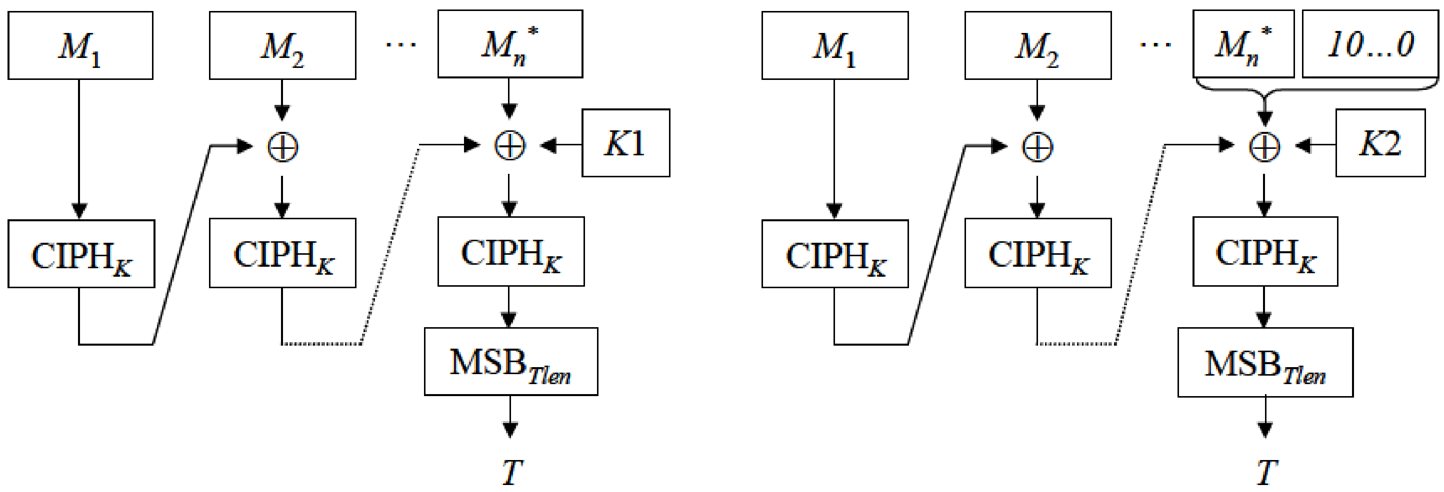
- The same key is used for encryption and decryption (opposite to RSA and other public key crypto)

Applications of block ciphers: encryption (confidentiality)

- network connections: SSL/TLS protocols (AES, Camellia, Triple DES, etc. in CBC or GCM modes)
- network traffic: IPsec protocol suite (AES, Camellia, Triple DES, GOST 28147-89 etc. in CBC, CTR or GCM modes)
- storage protection (AES in XTS mode)
- etc.

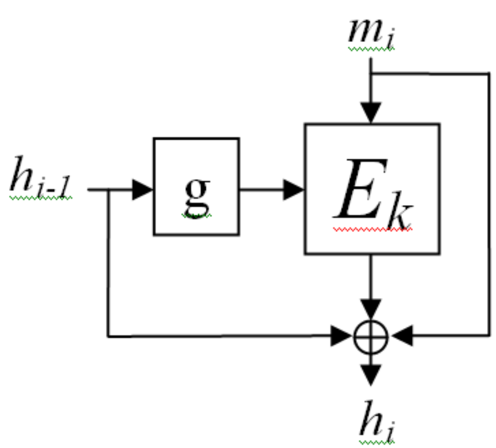
Applications of block ciphers: integrity

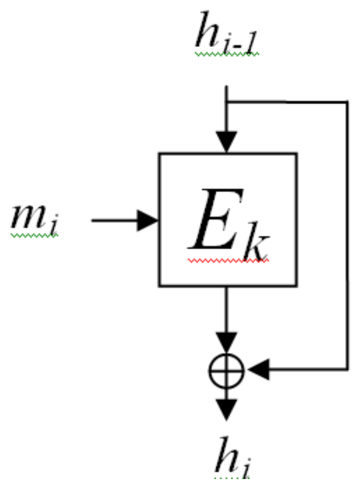
- verification, that the message was not modified/forged during transmission via untrusted channel (Internet, wireless networks, etc.):
  - **CMAC** (*C*ipher-based *M*essage *A*uthentication *C*ode)
  - **GMAC** (*G*alois *M*essage *A*uthentication *C*ode), **GCM** (*G*alois/*C*ounter *M*ode)

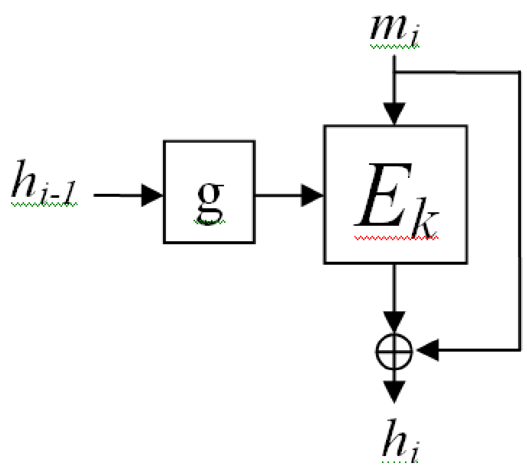


Applications of block ciphers: elements of other primitives

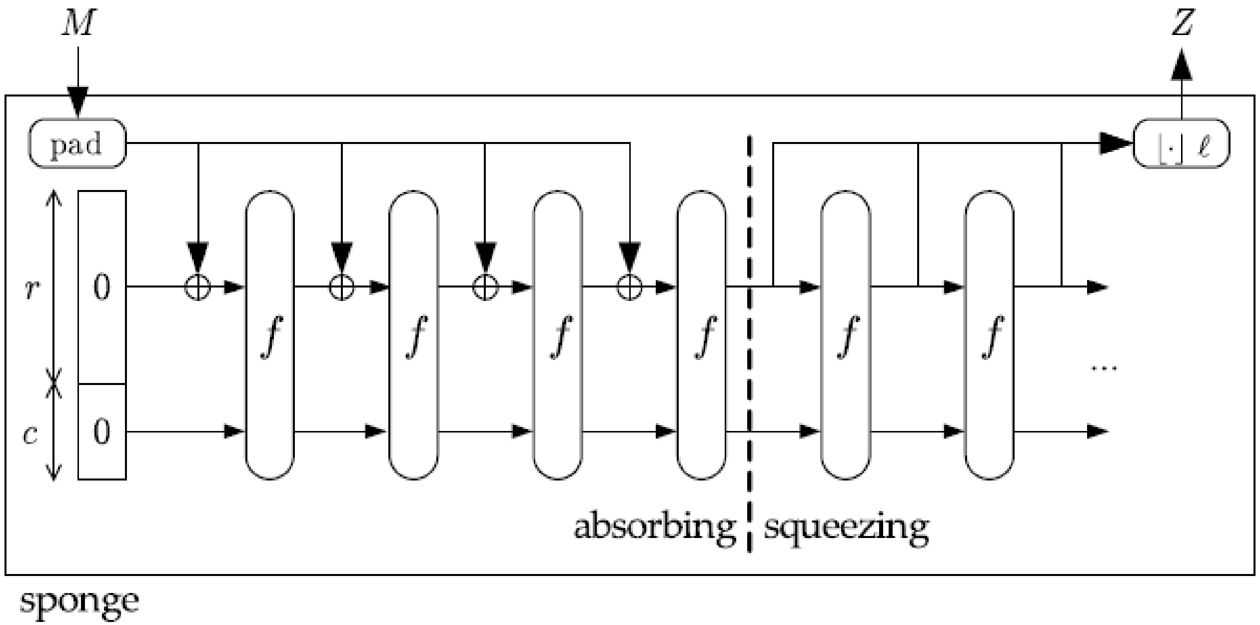
- Hash function constructions:
  - Miyaguchi–Preneel
  - Davies–Meyer
  - Matyas–Meyer–Oseas





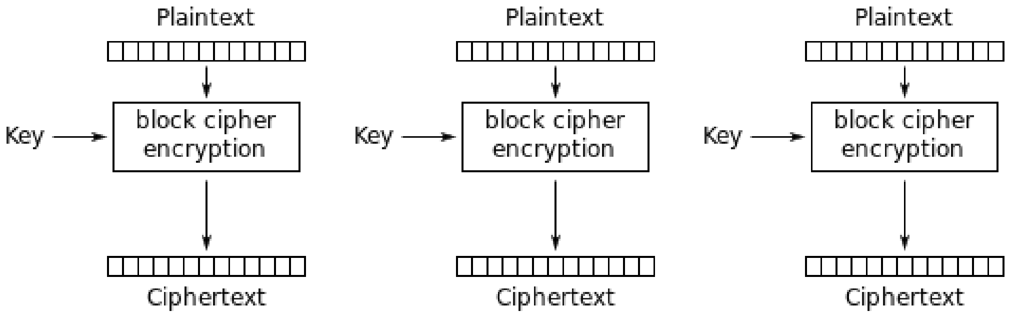


Applications of block ciphers: a permutation in sponge construction



- hash function (Keccak/SHA-3)
- message authentication code
- stream cipher
- authenticated encryption

Ideal block cipher model



$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 5 & 4 & 3 & 1 \end{pmatrix}$$

$$\begin{pmatrix} x_1 & x_2 & x_3 & \cdots & x_n \\ \sigma(x_1) & \sigma(x_2) & \sigma(x_3) & \cdots & \sigma(x_n) \end{pmatrix}$$

- block cipher as a random permutation (fixed key gives one permutation)
- number of random permutations: $(2^n)!$, where *n* is block size in bits
- practical implementation is impossible: requires $2^{64} \cdot 8 = 2^{67}$ bytes just for simple 64-block encryption using the single key
- real block cipher only takes 2^k from $(2^n)!$, where *k* is key length

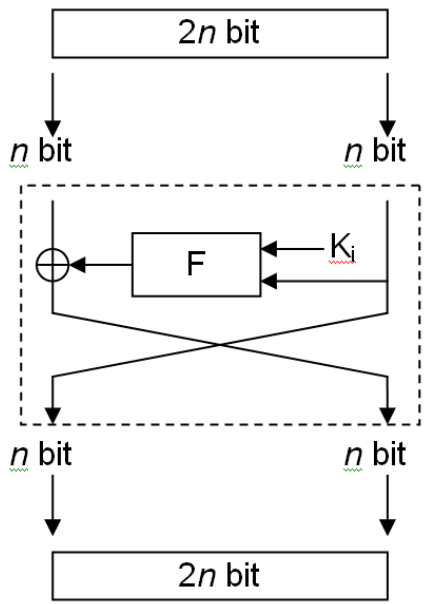
Block cipher: requirements and construction principles

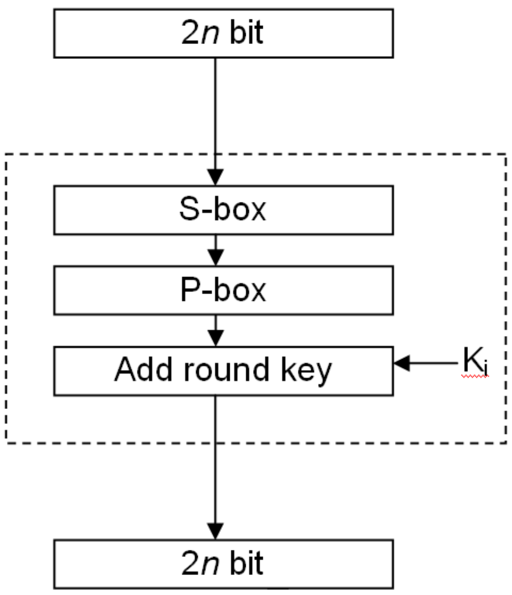
- must behave like a **random substitution** (hiding all redundancy of plaintext)
- truly random substitution of corresponding size is quite ineffective in implementation
- **iterative structure**: sequential application of *different* weak ciphers gives a strong one
- **each plaintext bit** and **each key bit** must have **influence on each ciphertext bit**
- linear and non-linear operations must be used (Shannon's *confusion* and *diffusion*)
- only **small tables and simple operations** (repeating many times) may be used to archive effective implementation

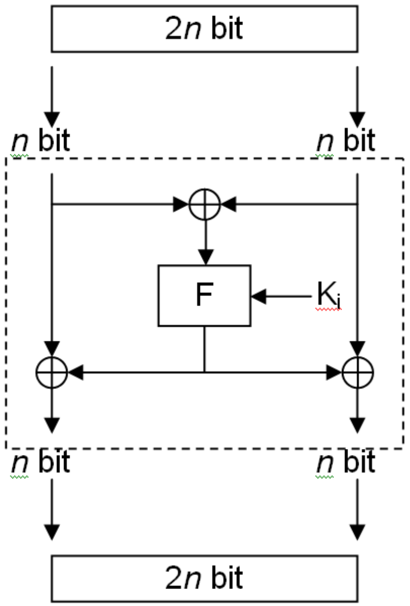
Practical implementation: iterated block ciphers

Repeating *weak* round function many times.
Main constructions of block ciphers:

- Feistel network
- SPN structure
- Lai-Massey scheme

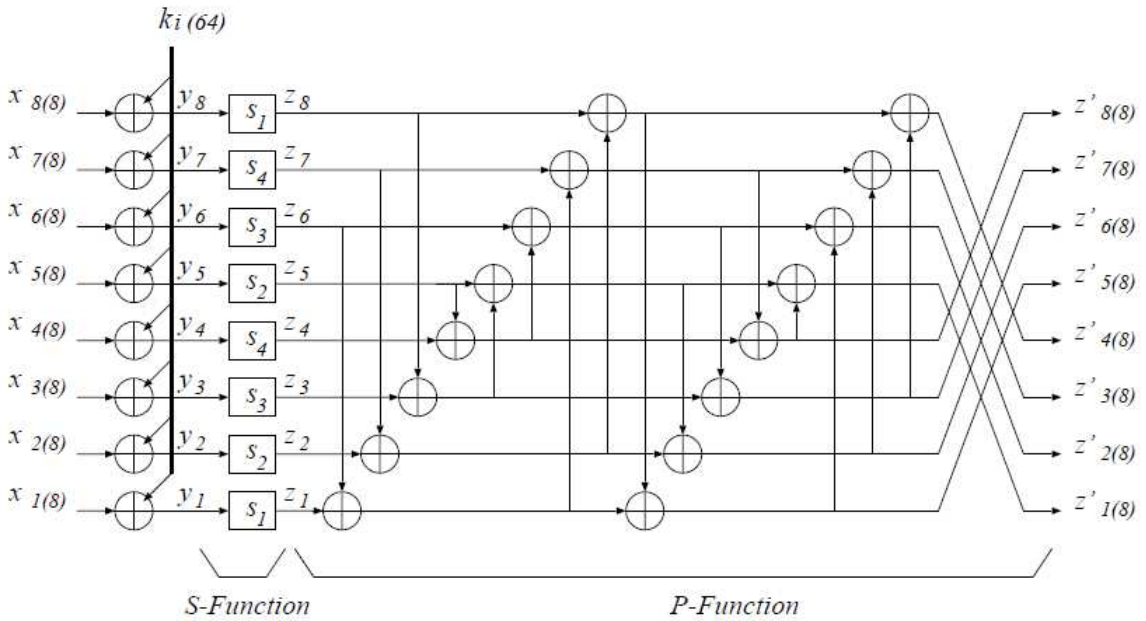






Block cipher round function

- Linear and non-linear layers for providing complex input/output dependency;
- One or few rounds can be easily broken, but enough will give a strong cipher
- Can be implemented:
  - S-boxes followed by linear transformations
  - sequence of addition, rotation and XOR (ARX-ciphers)
  - mix of above variants



Example: Camellia block cipher round function

Avalanche effect for block ciphers and hash functions

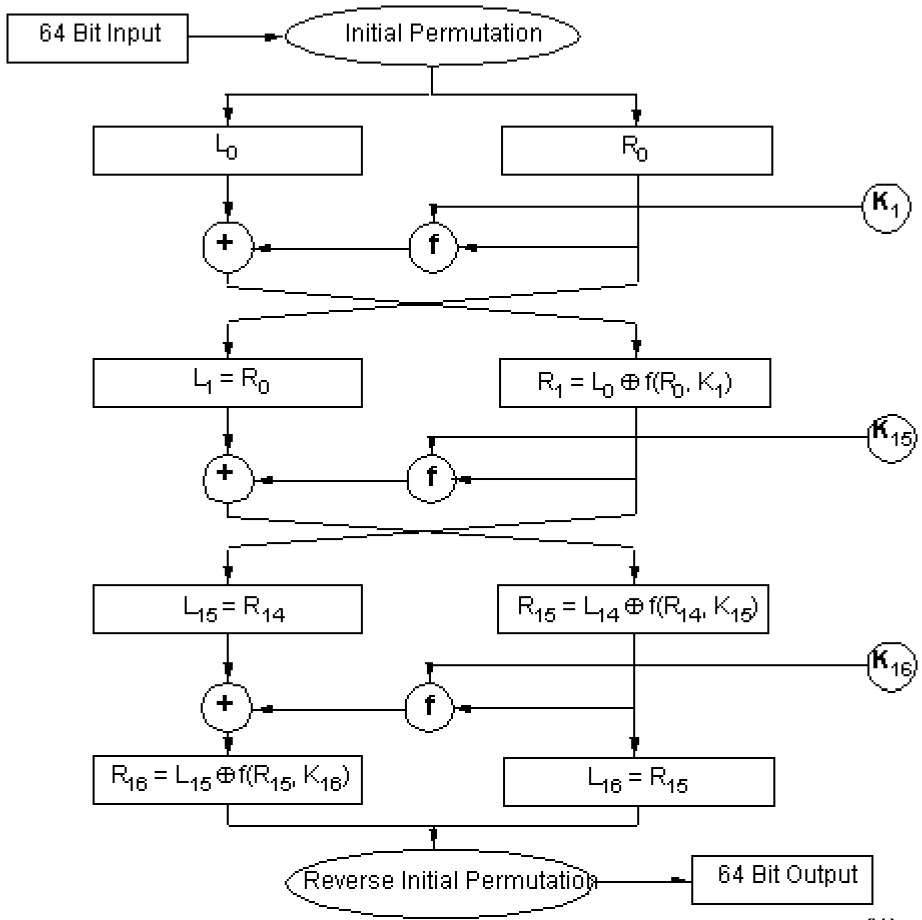
- changing one input bit (plaintext or key) leads to changing approximately half output bits (ciphertext) at random positions
- non-linear blocks (S-boxes) give complex dependency between S-box input bits (diffusion)
- linear blocks (bit permutation, linear bit transformations, including MDS matrix multiplication) gives “difference spreading” to the rest of S-boxes
- multiple rounds (product cipher) allow to get complex non-linear dependencies of all output bits on all plaintext and key bits
- avalanche property is very important for strength to different cryptanalysis methods

Practical (computational) security: wide spread block ciphers and their characteristics

Name	Block length, bits	Key length, bits	Theoretically broken	Practically broken	Effective in software	Effective in hardware
AES	128	128, 192, 256	Yes	No	Yes	Yes
DES	64	56	Yes	Yes	No	Relatively
Triple DES	64	168	Yes	No	No	No
Blowfish	64	32-448	Yes	No	Yes	No
Twofish	128	128, 192, 256	No	No	Yes	Relatively
IDEA-NXT	64, 128	128, 256	No	No	Yes	Relatively
GOST 28147	64	256	Yes	No	Yes	Yes
PRESENT	64	80, 128	No	No	Relatively	Yes

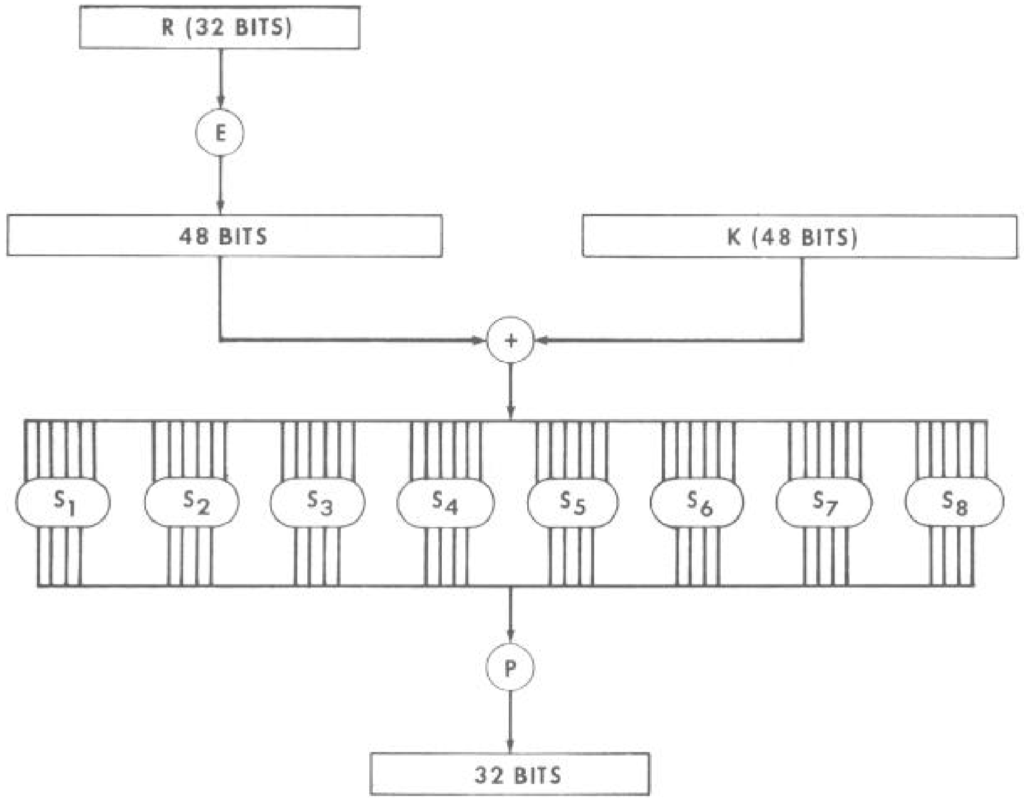
Legacy block cipher: Data Encryption Standard (DES)

- 64 bit block, 56 bit key
- 16-round Feistel network
- linear key schedule (master key bit permutation)
- based on IBM solution: Lucifer
- NSA improvement: decreased key length and improved strength to different cryptanalytic attacks, published later



Legacy block cipher: DES round function

- bit expansion *E* (32 bit -> 48 bit)
- round key addition (XOR)
- S-boxes (substitution tables, 6 bit -> 4 bit)
- bit permutation *P*



DES S-boxes

S_1

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

S_2

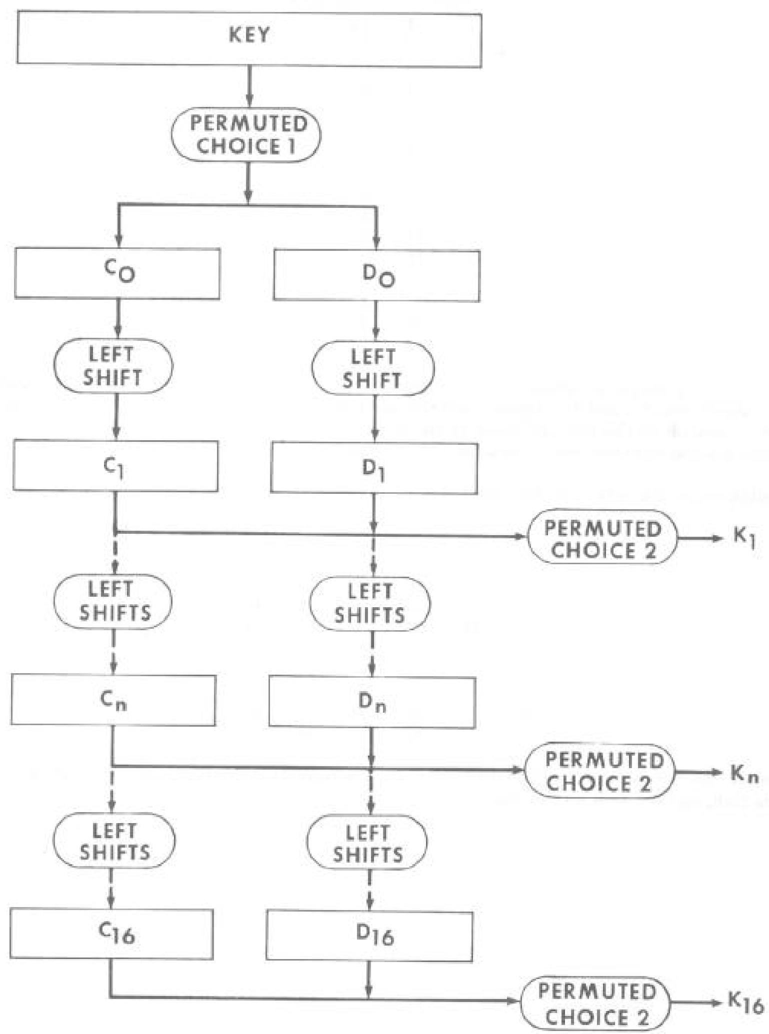
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3

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

DES
key schedule

- 56 bit of the encryption key are transformed into 16 round keys of 48 bit each
- cyclic shifts and bit permutation of encryption key are only used
- each round key is just a selected and permuted bits of encryption key



Data Encryption Standard: advantages and disadvantages

- the first publically available worldwide spread cipher with practically acceptable strength level
- no effective attacks exploiting internal properties completely breaking cipher strength were found (cf.: FEAL)
- improved version (TDEA or TripleDES with 168 bit key is allowed to be used by NIST together with AES)

- DES can be practically broken with brute-force attacks or using precomputed tables due to 56-bit key
- slow in software (comparing to AES, etc.)
- not effective as a lightweight solution

How ciphers are broken: examples of basic cryptanalysis methods

- brute force attacks
- precomputed tables (Hellman, rainbow, etc.)
- differential cryptanalysis and modifications
  - impossible differentials
  - truncated differentials
  - rectangle attack
  - boomerang attack
- linear cryptanalysis and modifications
- algebraic analysis
- etc.

Differential cryptanalysis

- very widely applied method of cryptanalysis for block ciphers, hash functions, etc.
- learns how the difference propagates via cryptographic transformations
- chosen plaintext attack (for most cases)
- the first method for successful analytical attack against DES (estimated complexity 2^{47})
- the first publication in open literature appeared in 1990 (IBM researches say they discovered it in 1974 and optimized DES against it, and NSA already knew about DC then)
- many other attacks are based on differential cryptanalysis
- some ciphers successfully had been practically broken (e.g., FEAL) with DC

Basics of differential cryptanalysis

- Difference $$\Delta X = X \oplus X'$$

- Linear function $$E(X) \oplus E(X') = E(\Delta X)$$

- Difference and round key addition $$(X \oplus K) \oplus (X' \oplus K) = \Delta X$$

- Non-linear function $$0 < P(S(\Delta X) = \Delta Y) < 1$$

Difference distribution table of DES S-box (S1)

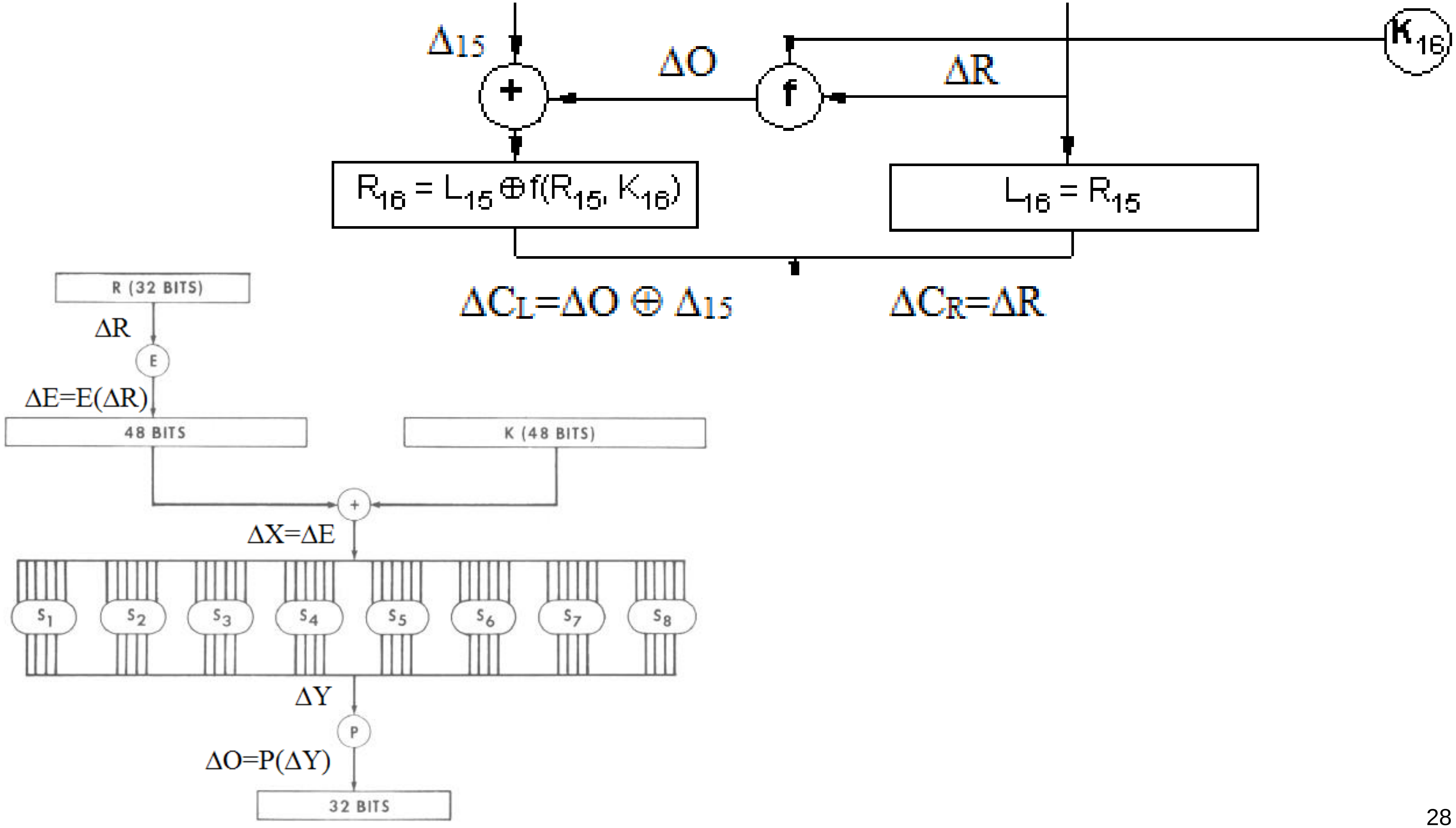
Input XOR	Output XOR															
	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	0	0	6	0	2	4	4	0	10	12	4	10	6	2	4
2_x	0	0	0	8	0	4	4	4	0	6	8	6	12	6	4	2
3_x	14	4	2	2	10	6	4	2	6	4	4	0	2	2	2	0
4_x	0	0	0	6	0	10	10	6	0	4	6	4	2	8	6	2
5_x	4	8	6	2	2	4	4	2	0	4	4	0	12	2	4	6
6_x	0	4	2	4	8	2	6	2	8	4	4	2	4	2	0	12
7_x	2	4	10	4	0	4	8	4	2	4	8	2	2	2	4	4
8_x	0	0	0	12	0	8	8	4	0	6	2	8	8	2	2	4
9_x	10	2	4	0	2	4	6	0	2	2	8	0	10	0	2	12
A_x	0	8	6	2	2	8	6	0	6	4	6	0	4	0	2	10
B_x	2	4	0	10	2	2	4	0	2	6	2	6	6	4	2	12
C_x	0	0	0	8	0	6	6	0	0	6	6	4	6	6	14	2
D_x	6	6	4	8	4	8	2	6	0	6	4	6	0	2	0	2
E_x	0	4	8	8	6	6	4	0	6	6	4	0	0	4	0	8
F_x	2	0	2	4	4	6	4	2	4	8	2	2	2	6	8	8
									$\vdots$							
30_x	0	4	6	0	12	6	2	2	8	2	4	4	6	2	2	4
31_x	4	8	2	10	2	2	2	2	6	0	0	2	2	4	10	8
32_x	4	2	6	4	4	2	2	4	6	6	4	8	2	2	8	0
33_x	4	4	6	2	10	8	4	2	4	0	2	2	4	6	2	4
34_x	0	8	16	6	2	0	0	12	6	0	0	0	0	8	0	6
35_x	2	2	4	0	8	0	0	0	14	4	6	8	0	2	14	0
36_x	2	6	2	2	8	0	2	2	4	2	6	8	6	4	10	0
37_x	2	2	12	4	2	4	4	10	4	4	2	6	0	2	2	4
38_x	0	6	2	2	2	0	2	2	4	6	4	4	4	6	10	10
39_x	6	2	2	4	12	6	4	8	4	0	2	4	2	4	4	0
$3A_x$	6	4	6	4	6	8	0	6	2	2	6	2	2	6	4	0
$3B_x$	2	6	4	0	0	2	4	6	4	6	8	6	4	4	6	2
$3C_x$	0	10	4	0	12	0	4	2	6	0	4	12	4	4	2	0
$3D_x$	0	8	6	2	2	6	0	8	4	4	0	4	0	12	4	4
$3E_x$	4	8	2	2	2	4	4	14	4	2	0	2	0	8	4	4
$3F_x$	4	8	4	2	4	0	2	4	4	2	4	8	8	6	2	2

Non-linear transformation (S-box)

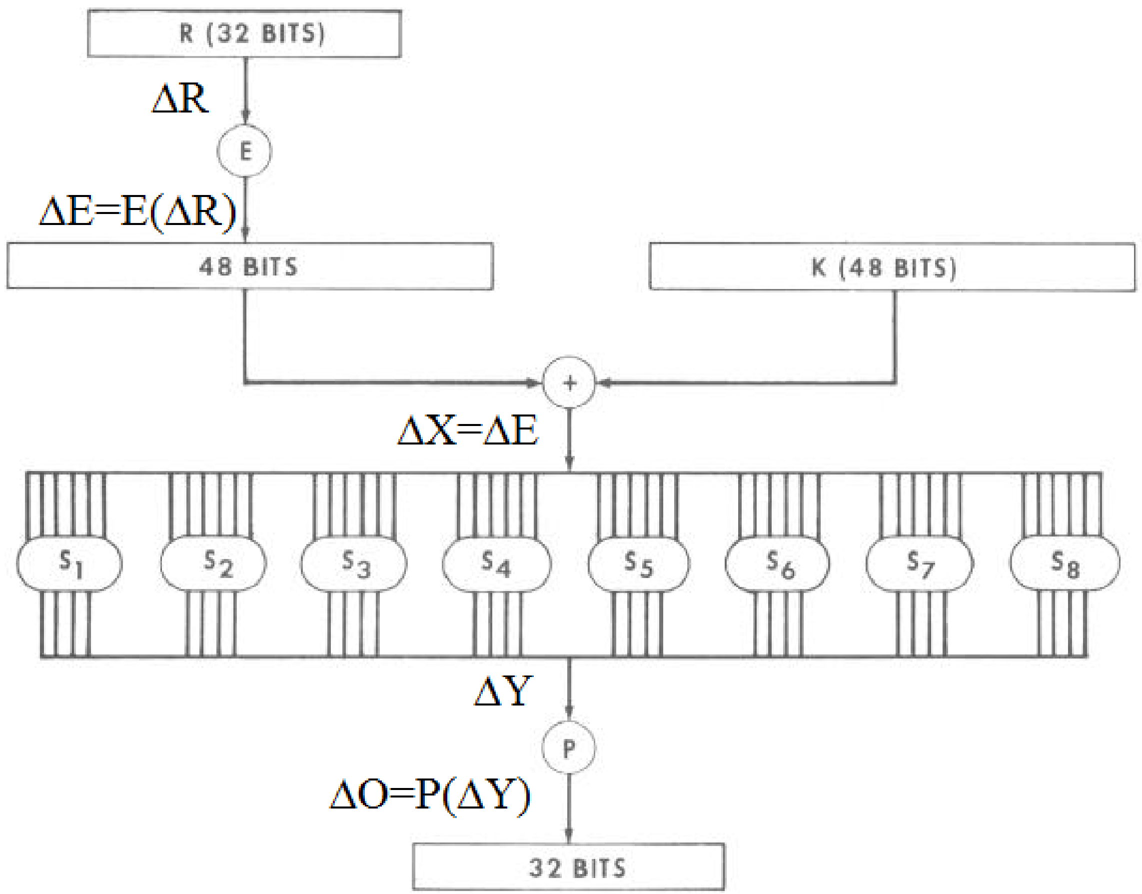
non-zero difference can be formed by limited (not all) input and output values:

Output XOR ($S1'_O$)	Possible Inputs ($S1_I$)
1	03, 0F, 1E, 1F, 2A, 2B, 37, 3B
2	04, 05, 0E, 11, 12, 14, 1A, 1B, 20, 25, 26, 2E, 2F, 30, 31, 3A
3	01, 02, 15, 21, 35, 36
4	13, 27
7	00, 08, 0D, 17, 18, 1D, 23, 29, 2C, 34, 39, 3C
8	09, 0C, 19, 2D, 38, 3D
D	06, 10, 16, 1C, 22, 24, 28, 32
F	07, 0A, 0B, 33, 3E, 3F

Differential cryptanalysis: the last round of encryption

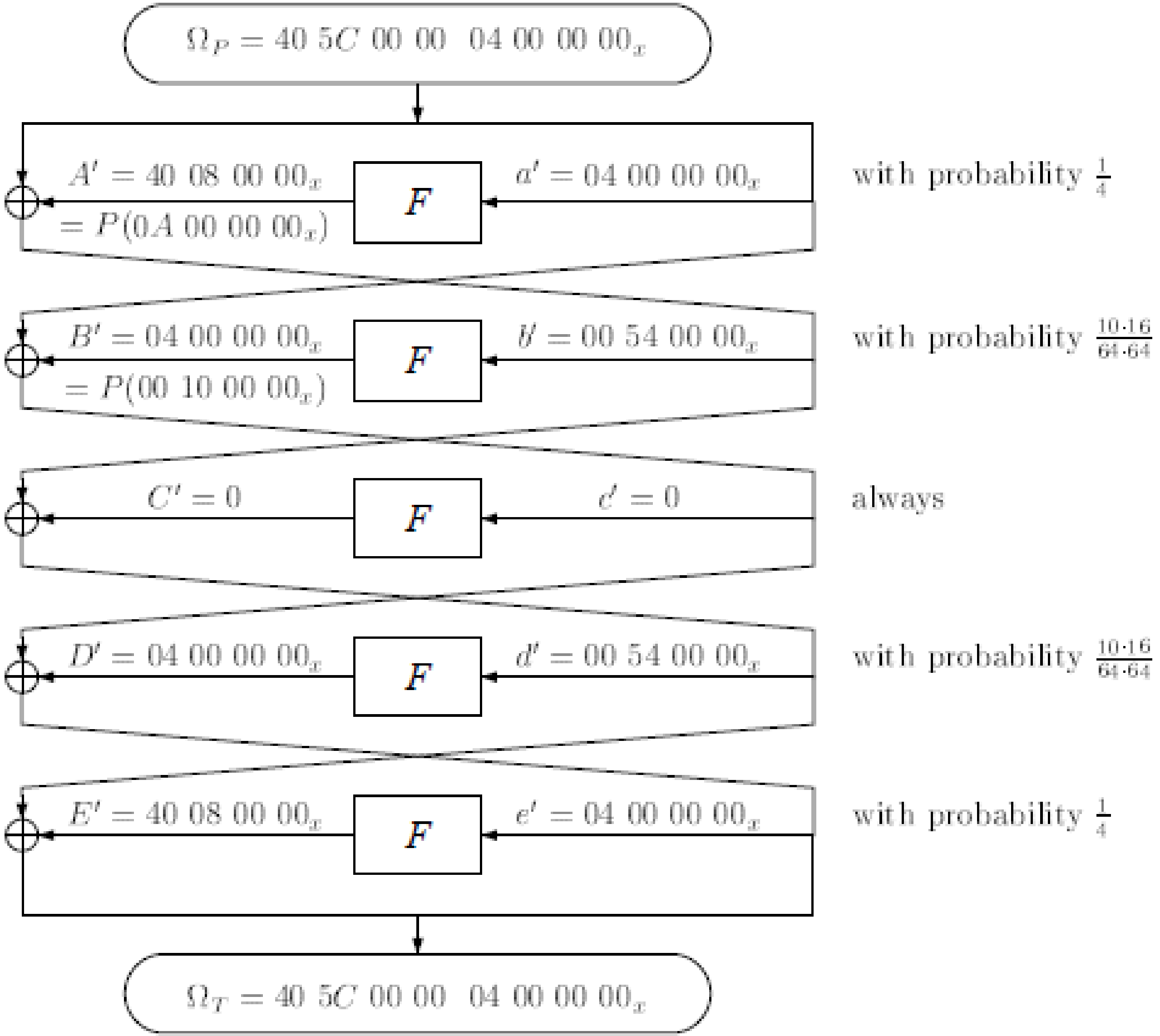


Differential cryptanalysis: transformations in the last round function



- ΔX, ΔY are known => only several variants of X (not all) are possible
- R, R' are known (equal to right halves of ciphertext)
- Possible key bits values: $K = R \oplus X$

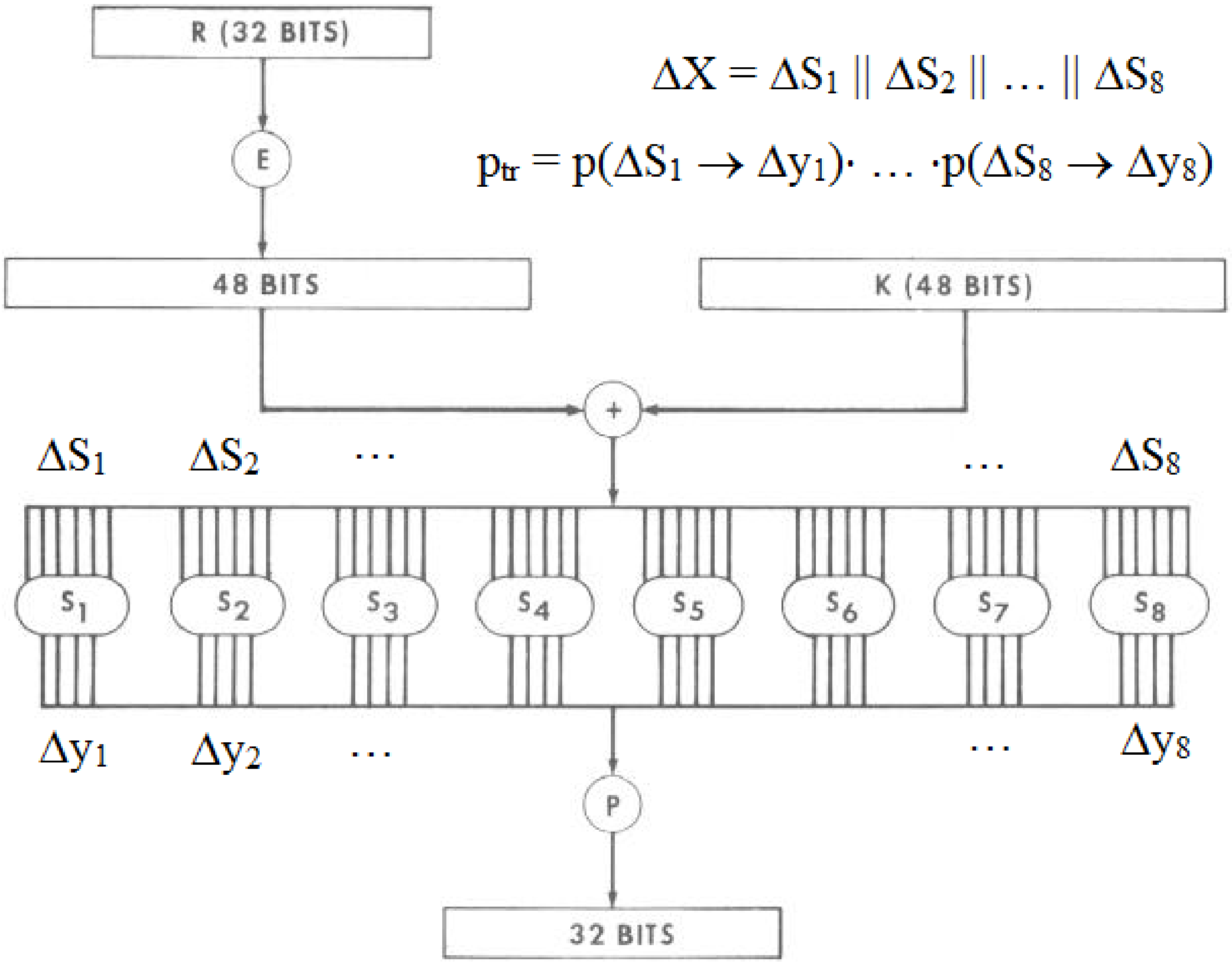
Differential characteristics: obtaining necessary differences on the last round



Differential characteristics probabilities: one round calculation

$$\Delta X = \Delta S_1 \,||\, \Delta S_2 \,||\, \ldots \,||\, \Delta S_8$$

$$p_{tr} = p(\Delta S_1 \rightarrow \Delta y_1) \cdot \ldots \cdot p(\Delta S_8 \rightarrow \Delta y_8)$$



NB: random independent round keys (hypothesis stochastic equivalence)

Attack complexity and strength to differential cryptanalysis

- Probability of differential characteristic determines the required number of chosen plaintext encryptions (mathematical expectation)
- Complexity of the attack (classic approach) depends on
  - maximal probability of difference transformation on S-box(es)
  - number of active S-boxes used in differential characteristic
- Cryptographic primitive is resistant do differential cryptanalysis, if the complexity of the attack is higher than the brute force search

Complexity of DES differential cryptanalysis

No. of Round	Chosen Plaintexts	Known Plaintexts
8	2^{14}	2^{38}
10	2^{24}	2^{43}
12	2^{31}	2^{47}
14	2^{39}	2^{51}
16	2^{47}	2^{55}

Linear cryptanalysis

- very widely applied method of cryptanalysis for block ciphers, hash functions, etc.
- learns how the non-linear cryptographic transformation can be approximated with linear/affine equations
- known (not chosen) plaintext attack (for most cases)
- the first practically implemented method for successful analytical attack against DES (with complexity 2^{43})
- first publication in open literature appeared in 1992 (against FEAL cipher, then applied to DES)

S-box: non-linear element and its approximation table

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

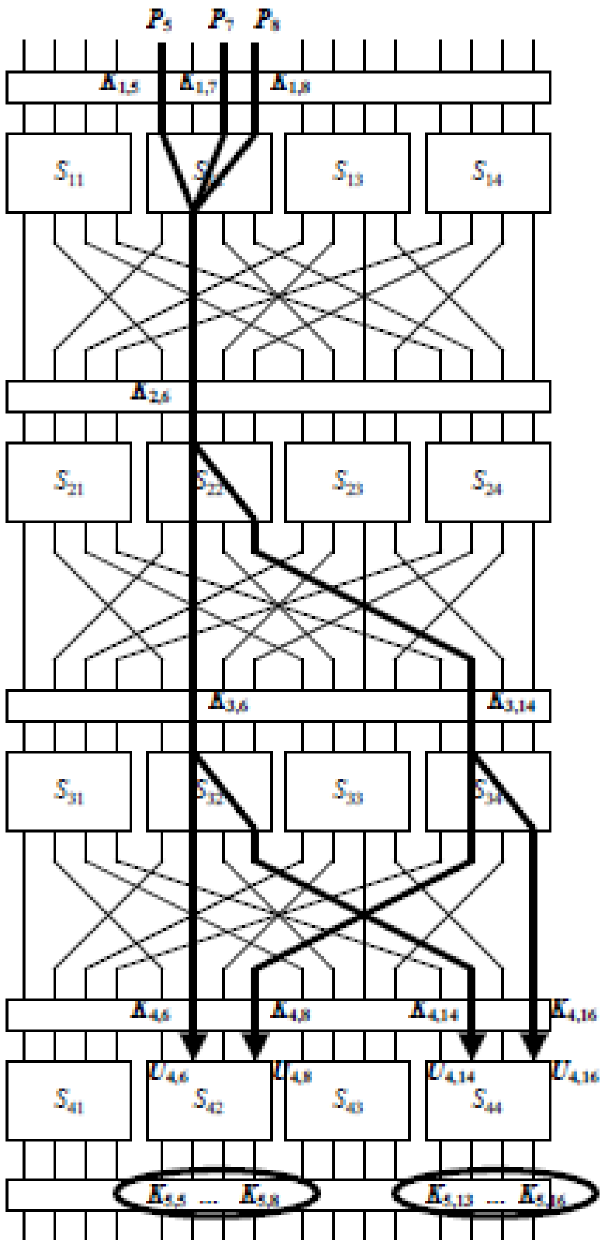
$$NS(\alpha,\beta)=\#\{x \mid w_t(x \,\&\, \alpha)=w_t(S(x) \,\&\, \beta)\}$$

		Output Sum															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Input Sum	0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	−2	−2	0	0	−2	+6	+2	+2	0	0	+2	+2	0	0
	2	0	0	−2	−2	0	0	−2	−2	0	0	+2	+2	0	0	−6	+2
	3	0	0	0	0	0	0	0	0	+2	−6	−2	−2	+2	+2	−2	−2
	4	0	+2	0	−2	−2	−4	−2	0	0	−2	0	+2	+2	−4	+2	0
	5	0	−2	−2	0	−2	0	+4	+2	−2	0	−4	+2	0	−2	−2	0
	6	0	+2	−2	+4	+2	0	0	+2	0	−2	+2	+4	−2	0	0	−2
	7	0	−2	0	+2	+2	−4	+2	0	−2	0	+2	0	+4	+2	0	+2
	8	0	0	0	0	0	0	0	0	−2	+2	+2	−2	+2	−2	−2	−6
	9	0	0	−2	−2	0	0	−2	−2	−4	0	−2	+2	0	+4	+2	−2
	A	0	+4	−2	+2	−4	0	+2	−2	+2	+2	0	0	+2	+2	0	0
	B	0	+4	0	−4	+4	0	+4	0	0	0	0	0	0	0	0	0
	C	0	−2	+4	−2	−2	0	+2	0	+2	0	+2	+4	0	+2	0	−2
	D	0	+2	+2	0	−2	+4	0	+2	−4	−2	+2	0	+2	0	0	+2
	E	0	+2	+2	0	−2	−4	0	+2	−2	0	0	−2	−4	+2	−2	0
	F	0	−2	−4	−2	−2	0	+2	0	0	−2	+4	−2	−2	0	+2	0

Linear approximation of several rounds

involving:

- linear approximations of S-boxes
- plaintext and ciphertext bits
- round keys bits



Attack complexity and strength to linear cryptanalysis

- The required number of plaintext encryptions is determined by the probability that linear approximation (linear hull) holds
- Complexity of the attack (classic approach) depends on
  - maximal bias of linear approximation on S-box(es)
  - number of active S-boxes used in linear approximation for the whole cipher
- Cryptographic primitive is resistant to linear cryptanalysis, if the complexity of the attack is higher than the brute force search

Algebraic cryptanalysis

- follows Claude Shannon idea (published 1949) “breaking a good cipher should require as much work as solving a system of simultaneous equations in a large number of unknowns of a complex type”
- known plaintext attack (usually)
- requires small amount of plaintext-ciphertext pairs (near to unicity distance)
- usually crypto transformation is described with overdefined system of a small (2-3) degree
- several ciphers were successfully broken with algebraic attacks
- methods of solving multivariate overdefined systems are being improved

Advanced Encryption Standard (AES)

- 128 bits block and 128, 192 or 256 bits key
- developed in Belgium, selected from 15 candidates (proposal from the US, Denmark, Germany, Israel, Japan, Switzerland, Armenia, etc.) during 4 year public cryptographic competition held by US National Institute of Standards (NIST)
- adopted as the US standard in 2001
- In 2002 allowed for protection of classified US government information
- the most researched cipher ever (in open publications)
- NSA cannot break even AES-128 and employs thousands of mathematician for this task (according to Ed.Snowden files)
- contemporary assumption: strong (practically unbreakable) encryption

Advanced Encryption Standard (AES)

- transparent design
- SPN construction (*S*ubstitution *P*ermutation *N*etwork)
- 10, 12 or 14 rounds for AES-128, AES-192 and AES-256 correspondingly
- quite effective in software (32-bit platforms), good for hardware implementation (not taking into account lightweight solutions)

AES: presentation of processing bytes as a “cipher state”

input bytes

in_0	in_4	in_8	in_{12}
in_1	in_5	in_9	in_{13}
in_2	in_6	in_{10}	in_{14}
in_3	in_7	in_{11}	in_{15}

→

State array

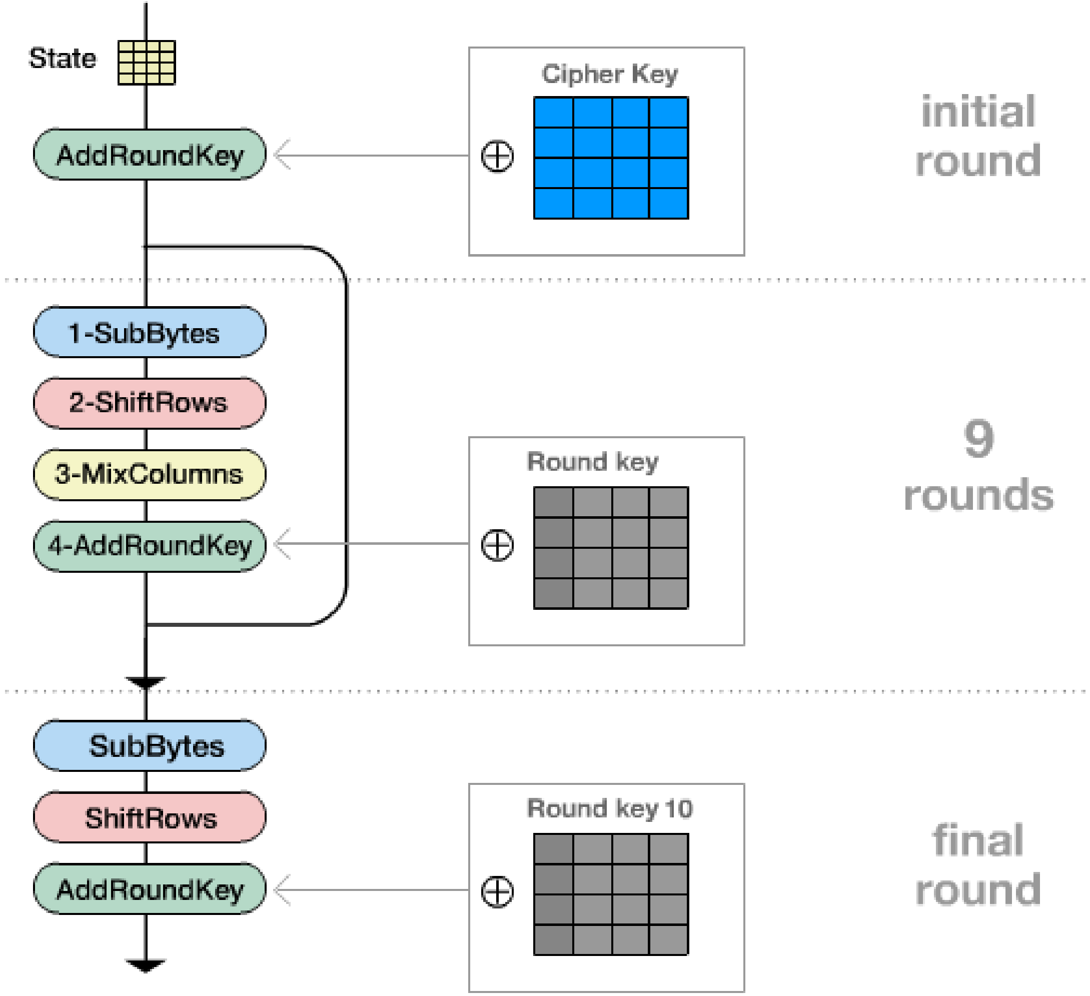
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

→

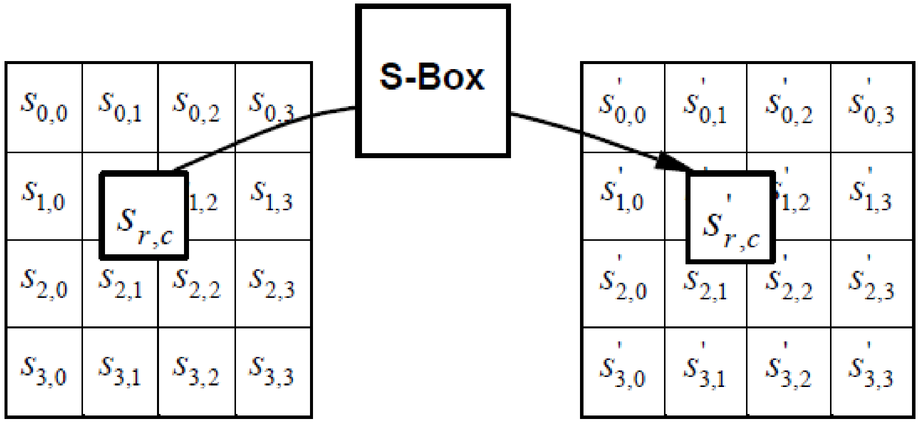
output bytes

out_0	out_4	out_8	out_{12}
out_1	out_5	out_9	out_{13}
out_2	out_6	out_{10}	out_{14}
out_3	out_7	out_{11}	out_{15}

AES: high-level structure (picture for 128 bit key)

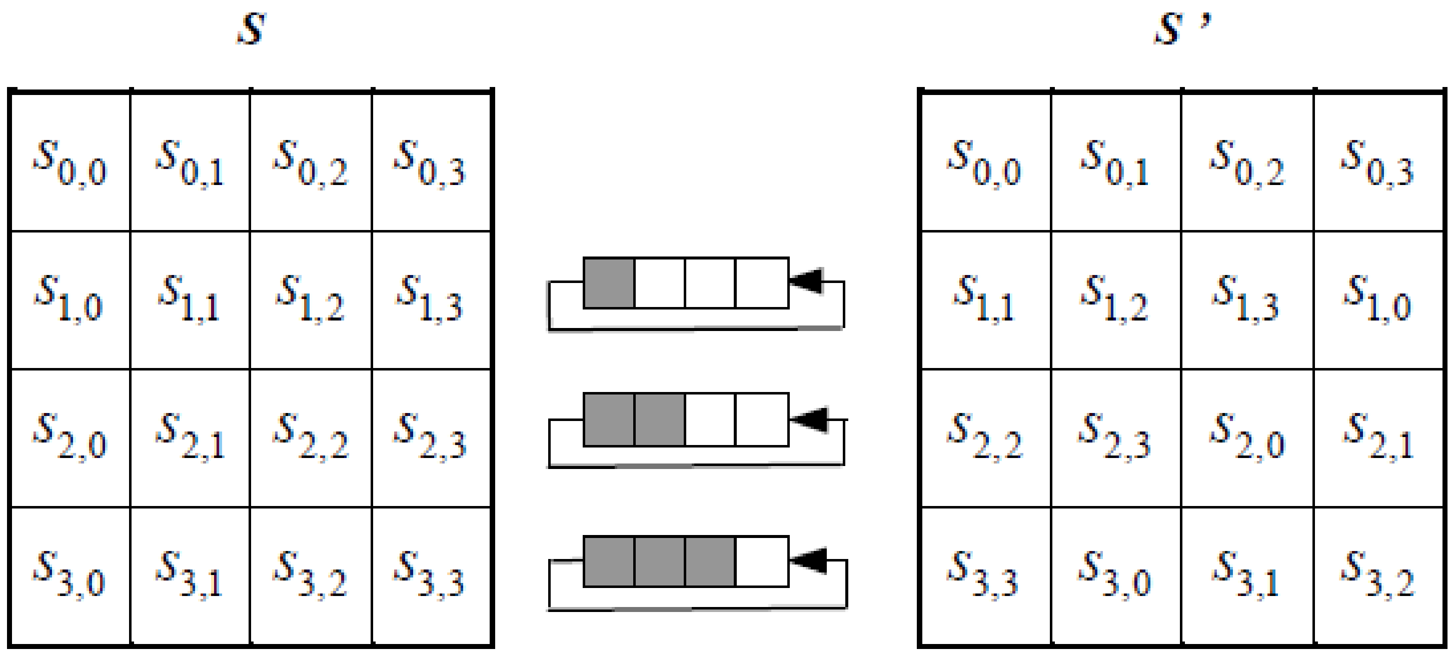


AES: SubBytes transformation



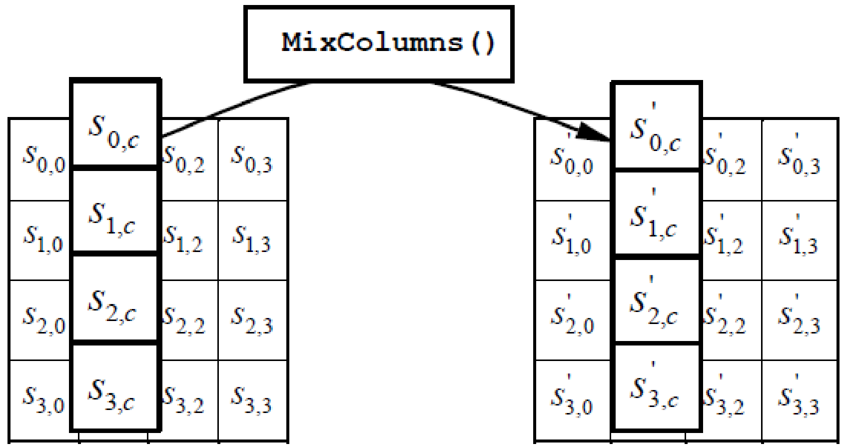
		y															
		0	1	2	3	4	5	6	7	8							
	0	63	7c	77	7b	f2	6b	6f	c5	30							
	1	ca	82	c9	7d	fa	59	47	f0	ad							
	2	b7	fd	93	26	36	3f	f7	cc	34							
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	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

AES: ShiftRows transformation



AES: MixColumns transformation

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



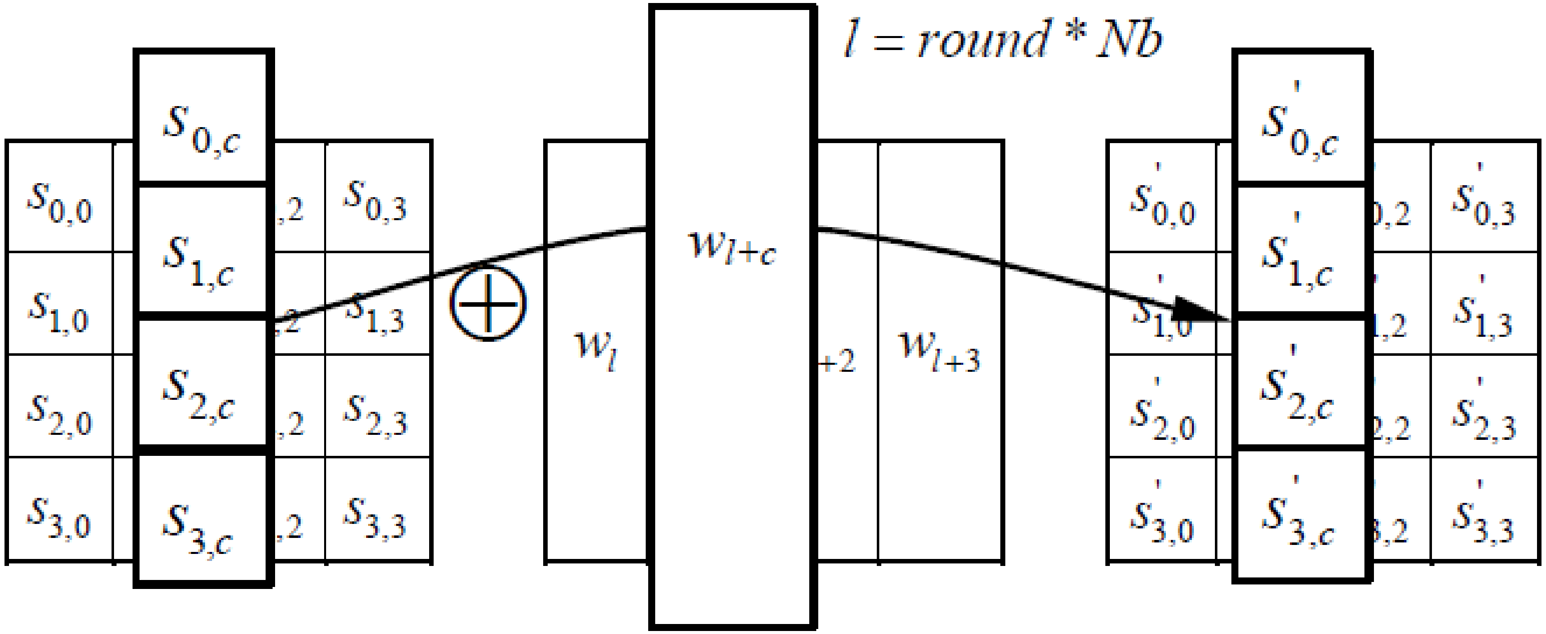
$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$

AES: AddRoundKey transformation

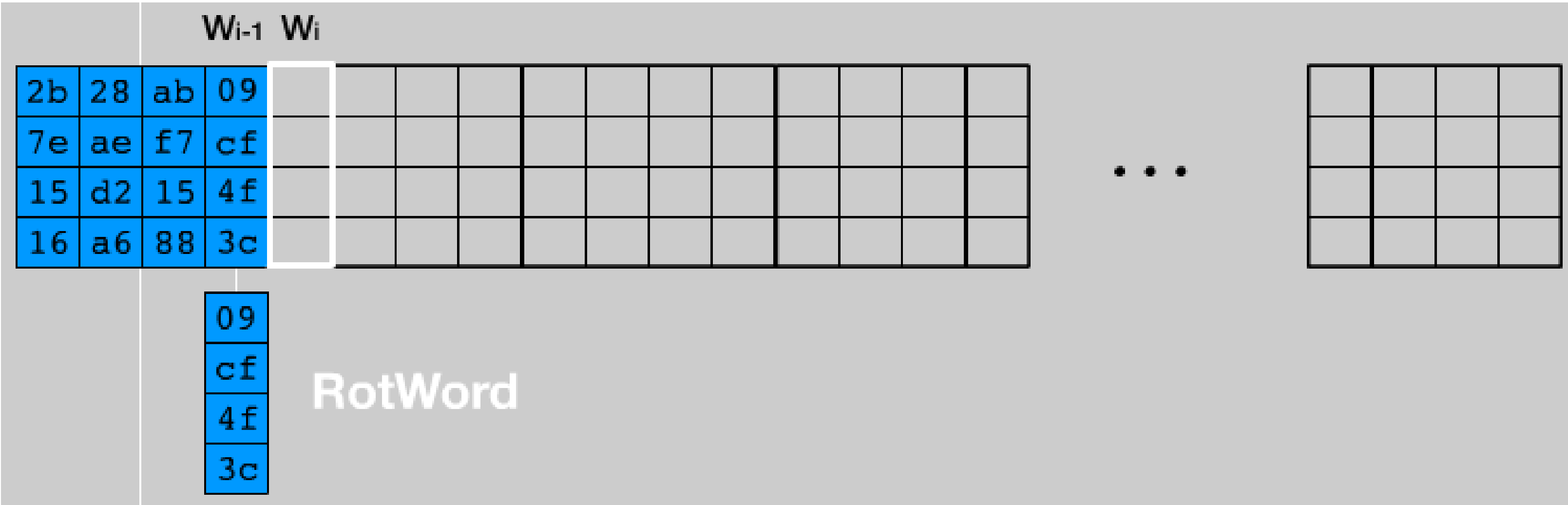


AES round key generation (key expansion)

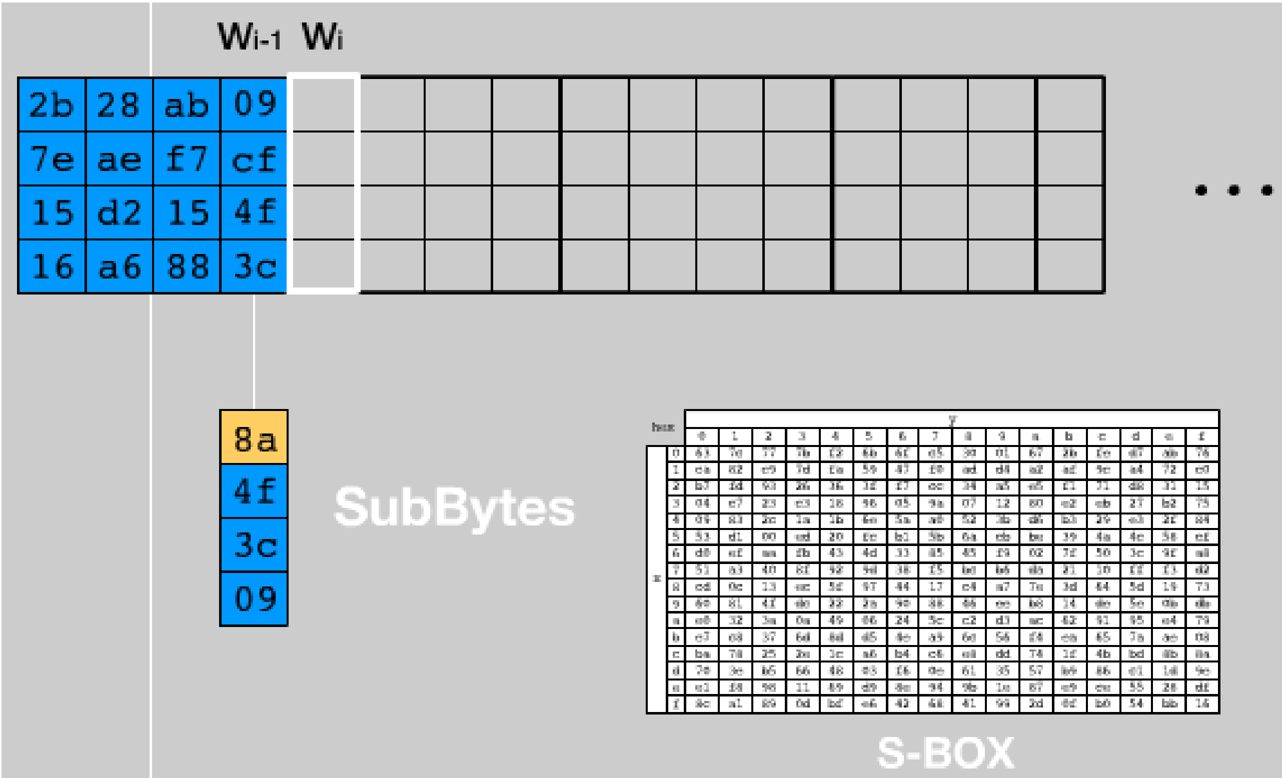
```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
   word  temp

   i = 0

   while (i < Nk)
      w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
      i = i+1
   end while

   i = Nk

   while (i < Nb * (Nr+1)]
      temp = w[i-1]
      if (i mod Nk = 0)
         temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
      else if (Nk > 6 and i mod Nk = 4)
         temp = SubWord(temp)
      end if
      w[i] = w[i-Nk] xor temp
      i = i + 1
   end while
end
```

NB: not all key length (128, 192, 256) must be supported; for many applications it's enough to have the single key length

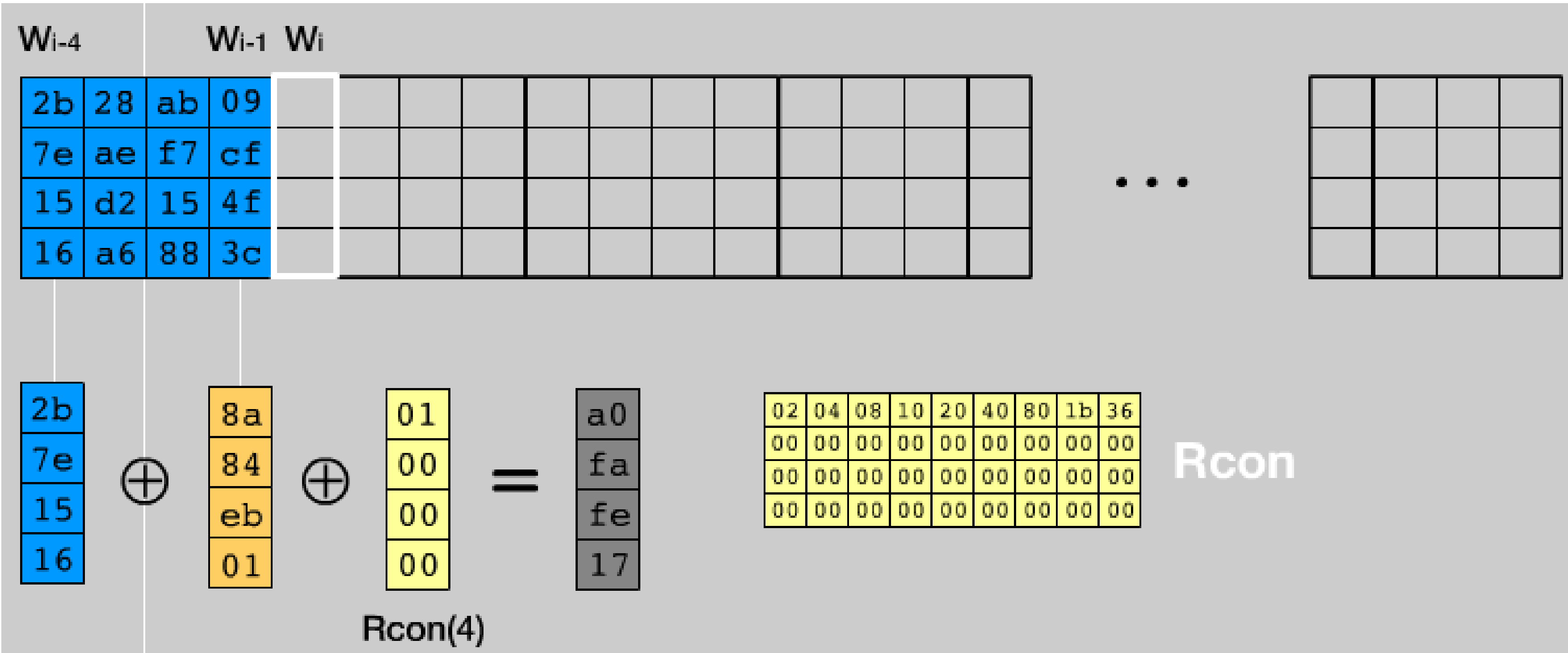
AES round key generation: RotWord



AES round key generation: SubBytes



AES round key generation: round constant application



NB: without Rcon there would be equal blocks in ciphertext if plaintext and keys have equal blocks (1, 2 or 4 bytes repeats in plaintext and key)

AES round key sequence

Cipher Key				Round key 1				Round key 2				Round key 3					Round key 10			
2b	28	ab	09	a0	88	23	2a	f2	7a	23	73	3d	47	1e	6d		d0	c9	e1	b6
7e	ae	f7	cf	fa	54	a3	6c	c2	96	a3	59	80	16	23	7a	. . .	14	ee	3f	63
15	d2	15	4f	fe	2c	39	76	95	b9	39	f6	47	fe	7e	88		f9	25	0c	0c
16	a6	88	3c	17	b1	39	05	f2	43	39	7f	7d	3e	44	3b		a8	89	c8	a6

AES effective software implementation: 32-bit platform

- **three different operations** can be united into the **single (!) look-up** table access:
  - SubBytes (**non-linear**)
  - ShiftRows (*linear*)
  - MixColumns (*linear*)

- cipher consists of look-up table accesses and round key additions

AES recommendation

- symmetric encryption on general purpose platform (32 bit, 64 bit) for commercial systems: **AES as the main cipher is a good solution**
- recommended mode for confidentiality is CTR (if you don’t use well researched authenticated encryption)
- the longer the key, the slower cipher is (20% slower for 192 bits and 40% slower for 256 bits comparing to 128 bit key speed)
- for very reliable systems implement AES-256 and an additional cipher (e.g., Camellia, Serpent, etc.)
- remember about implementation **integrity check** for plaintext or ciphertext together with encryption

Advanced Encryption Standard

Advantages

- one of the most spread commercial and open source solutions all over the world
- high level of practical security
- effective in software
- many hardware accelerators, including Intel processors AES instructions

Disadvantages

- theoretical attacks more effective than brute force are known
- 32-bit oriented (condition of the AES competition), does not take all advantages of the 64-bit platform

Further development of block ciphers: the first direction

Lightweight

- constrained devices: RFID chips, embedded medical devices, etc. (number of gates, available memory, power consumption and so on)
- acceptable strength level (cipher cannot be broken in the near future by small group of hackers)
- *not intended* to be strong against powerful adversary keeping theoretical strength for tens of years

Further development of block ciphers: the second direction

Governmental-level (high and ultrahigh) security

- must be cryptographically strong
- must have enough security margin to be protected (with high level of confidence) of newly discovered attacks
- not intended for highly constrained devices (used on servers, routers, PC, etc.)
- must provide fast encryption

Newly developed block ciphers providing high level security

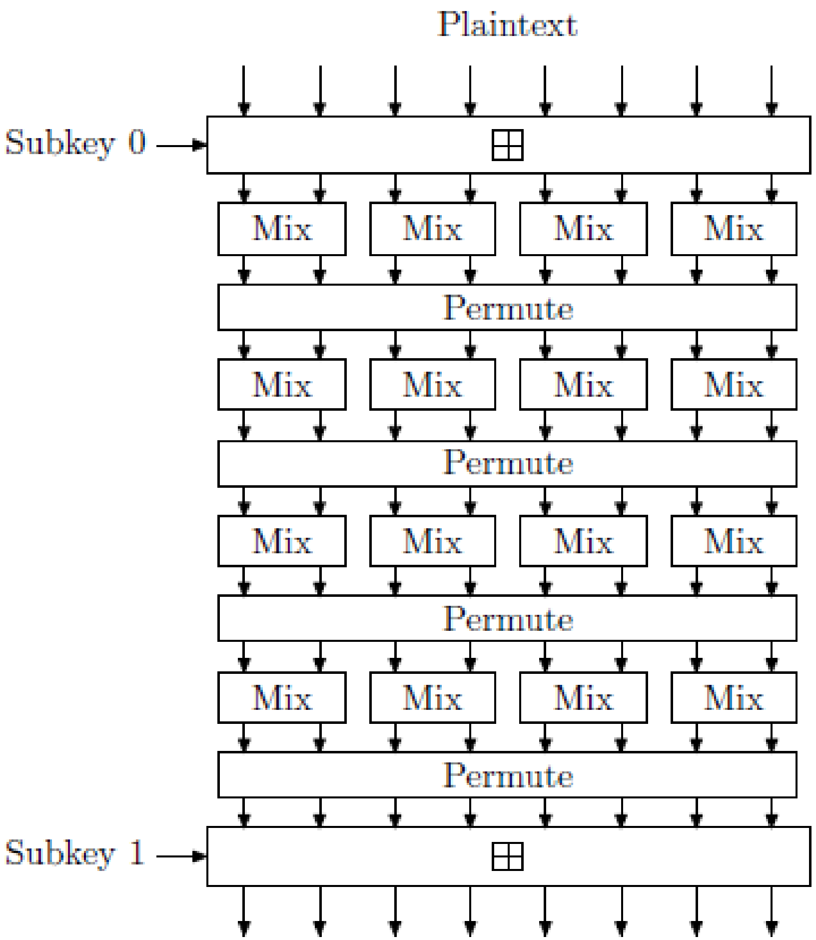
- Threefish (USA)
- STB 34.101.31-2011 (Belorussia)
- Kuznechik (Russia)
- Kalyna (Ukraine)

Threefish block cipher

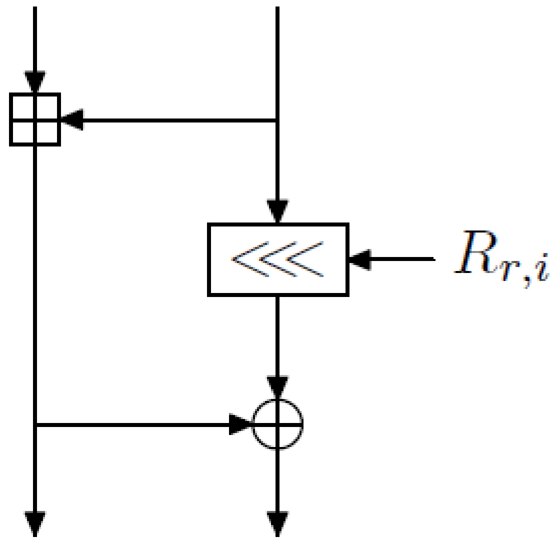
- a main part of Skein hash function (supports very big block sizes)
- ARX-cipher (addition, rotation, XOR)
- simple round function, many rounds

Block/Key Size	# Words N_w	# Rounds N_r
256	4	72
512	8	72
1024	16	80

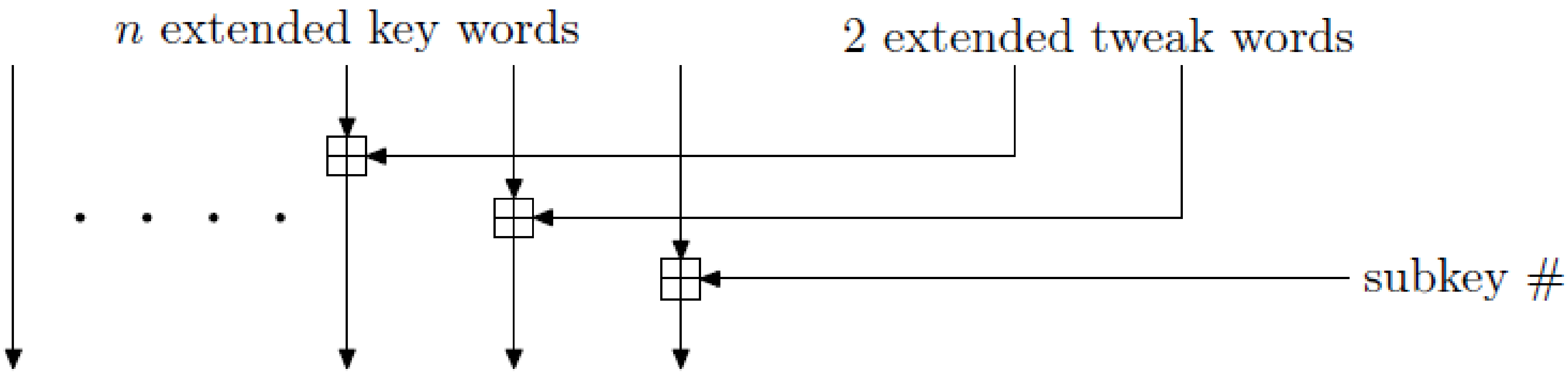
Threefish-512: 4 rounds of the 72



The MIX function



Threefish round key generation



$$k_{N_w} := C_{240} \oplus \bigoplus^{N_w - 1} k_i \qquad t_2 := t_0 \oplus t_1$$

$C_{240} = \texttt{0x1BD11BDAA9FC1A22}$

$$k_{s,i} := k_{(s+i) \bmod (N_w+1)} \qquad \text{for } i = 0, \ldots, N_w - 4$$

$$k_{s,i} := k_{(s+i) \bmod (N_w+1)} + t_{s \bmod 3} \qquad \text{for } i = N_w - 3$$

$$k_{s,i} := k_{(s+i) \bmod (N_w+1)} + t_{(s+1) \bmod 3} \qquad \text{for } i = N_w - 2$$

$$k_{s,i} := k_{(s+i) \bmod (N_w+1)} + s \qquad \text{for } i = N_w - 1$$

Threefish encryption performance

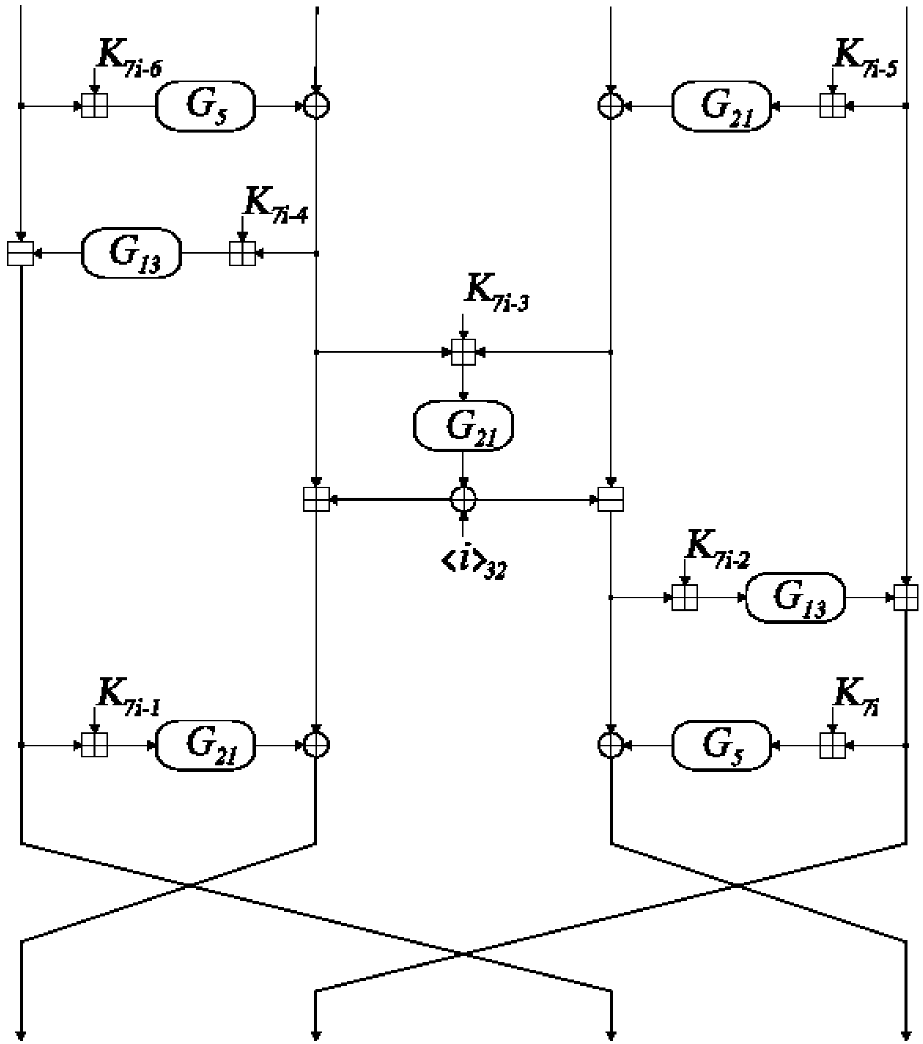
Very fast software implementation:

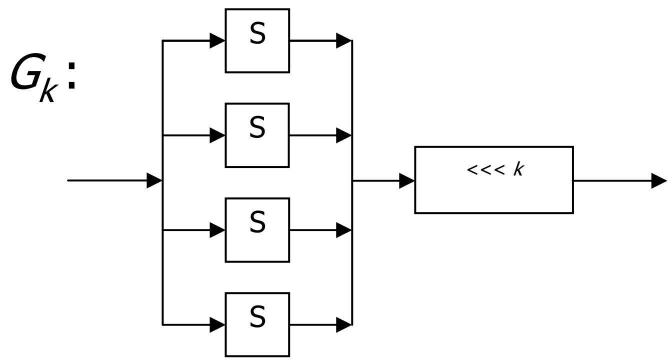
- 4 Gb/s on Intel Core 2 Duo x64
- or 6.1 clocks per byte for encryption

Belorussian standard STB 34.101.31-2011 (Bel-T)

- 128 bit block
- 128, 192 or 256 bit key
- 8-round combination of Feistel network and Lai-Massey scheme
- Single fixed S-box (8 bit-to-8 bit) with good properties
- no key schedule (parts of encryption key are used as round keys; key shorter than 256 bits is just padded)
- The latest version adopted in 2011

Belorussian standard STB 34.101.31-2011 (Bel-T)





The new Russian cipher: “Kuznechik” (“Grasshopper”)

- well-researched AES-like construction (S-boxes, ShiftRows, MixColumns)
- 10 rounds of encryption
- MixColumns: a big (16x16) MDS matrix over GF(2^8) generated by special method; cf.: AES has the 4x4 MDS matrix
- key schedule: Feistel network with constants as its round keys; each round gives a round key for the main cipher

- high level of security
- slower in software comparing to other modern block ciphers
- not adopted and officially published (only discussed on several conferences in Russia): final version of S-boxes and MDS matrix are not disclosed to public yet

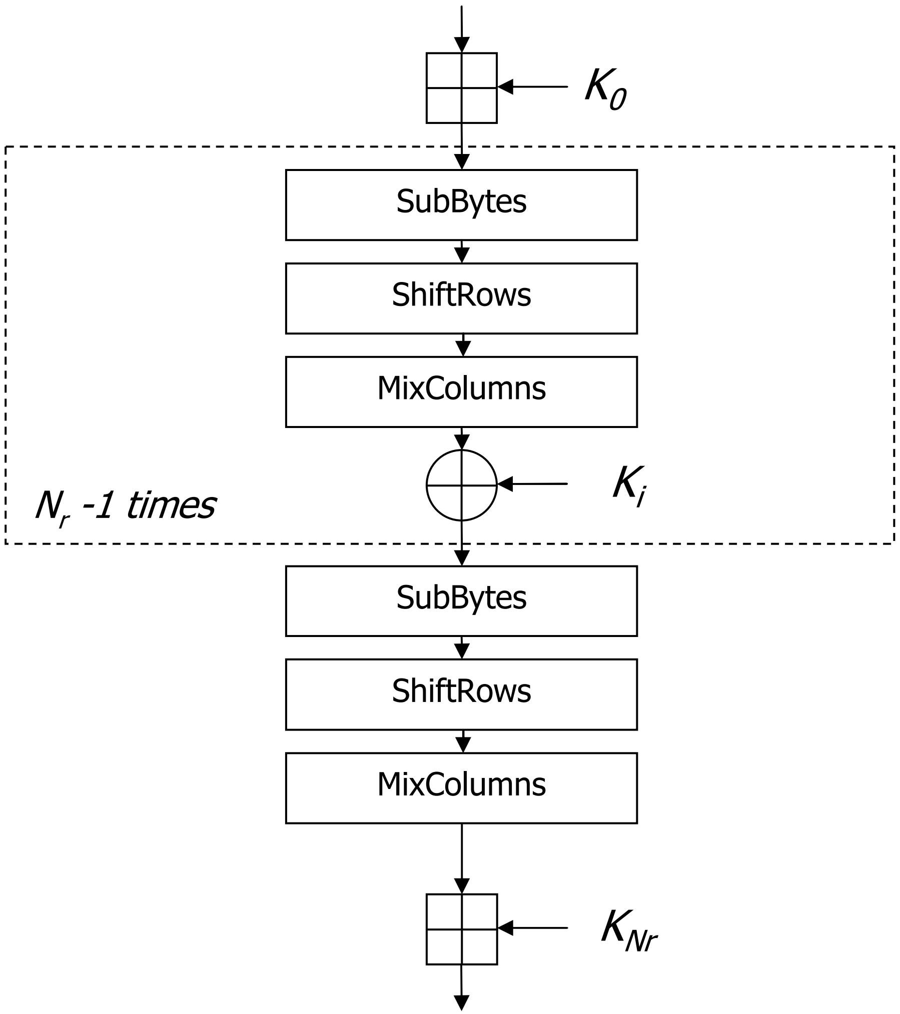
Requirements to the new perspective cipher for Ukraine

- block size and key length: 128, 256 and 512 bits (high and ultrahigh security level)
- strength against known methods of cryptanalysis
- security margin against future improved attacks
- transparent design
- effective high-speed software implementation on the 64-bit platform
- estimated time: at least 30 years (in condition of quantum cryptanalysis impossibility)

Perspective block cipher “Kalyna”

- SPN-construction (AES-like)
- increased size of linear transformation matrix
- several S-boxes generated with respect to differential, linear and algebraic properties
- quite new construction of key schedule, simple in implementation

“Kalyna” encryption function



$$Kalyna_K = \chi_{K_{N_r+1}} \circ \theta \circ \pi \circ \gamma \circ \prod_{i=1}^{N_r-1} \left(\sigma_{K_i} \circ \theta \circ \pi \circ \gamma \right) \circ \chi_{K_0}$$

“Kalyna”: number of rounds

Block size \ Key length	128 ($N_k = 2$)	256 ($N_k = 4$)	512 ($N_k = 8$)
128 ($N_b = 2$)	10	14	–
256 ($N_b = 4$)	–	14	18
512 ($N_b = 8$)	–	–	18

S-boxes for "Kalyna"

4 different S-boxes (which are not CCZ-equivalent) with the following characteristics:

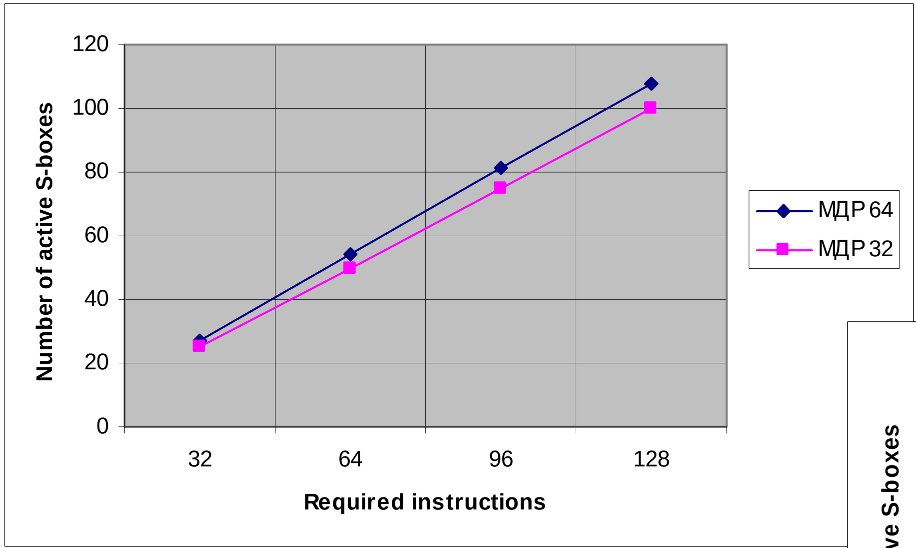
Nonlinearity	104
Minimal algebraic degree of component Boolean functions	7
Max. value of difference distribution table	8
Max. value of linear bias	24
Overdefined system degree	3 (441 equations)

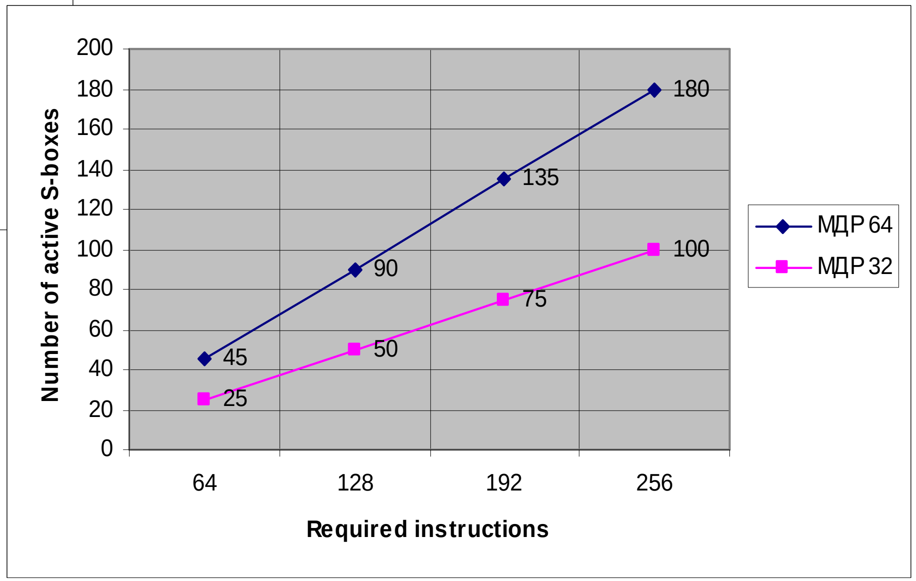
AES S-box: overdefined system degree: 2, nonlinearity: 112, dd: 4

The best known nonlinearity of S-boxes with 3rd degree:

Crypton, Safer+, Skipjack, SNOW, Twofish, Whirlpool, CS, Anubis, Stribog

Number of active S-boxes depending on required 64-bit processor instructions for 4x4 and 8x8 MDS matrix over GF(2^8) for 128 bit (left) and 256 bit (right) block





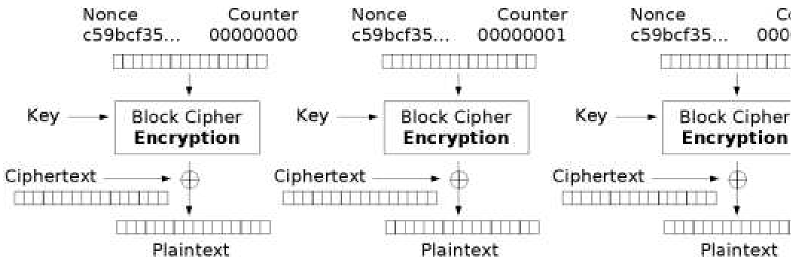
Increased size of MDS matrix gives essential advantages for required cryptographic properties, and has effective implementation on modern platforms

“Kalyna” encryption function design principles

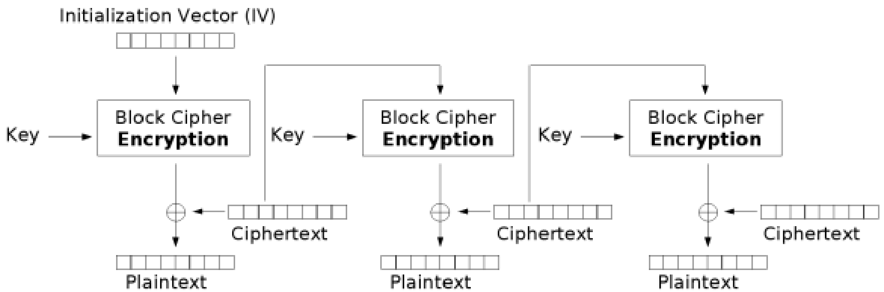
- well known wide trail design strategy (strength to differential, linear cryptanalysis, etc.) combined with modular pre- and post-whitening
- clear construction, no trapdoors
- new set of S-boxes (without essential algebraic structure)
- 64-bit platform operations ($\bmod 2^{64}$ addition, 8x8 MDS matrix)
- direct transformation (encryption) is more often used than reverse (decryption)
- effective software implementation
- developed for and most effective on 64-bit platforms

Optimization for direct transformation (encryption)

- most block cipher modes of operation (CTR, OFB, CFB, CCM, GCM, etc.) do not need block cipher decryption:



Counter (CTR) mode decryption



Cipher Feedback (CFB) mode decryption

- block cipher based hashing does not need decryption
- block cipher based pseudorandom number generation does not need decryption
- sponge construction does not need block cipher decryption

Number of precomputed tables:

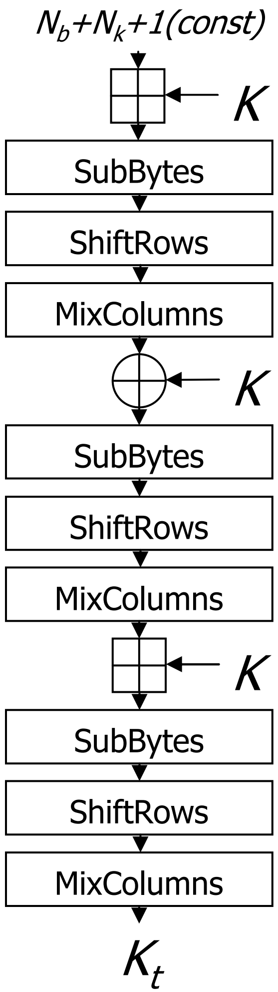
- AES (4 tables)
  - 2 tables for encryption
  - 2 tables for decryption
- Kalyna (4 tables)
  - 1 table for encryption
  - 3 tables for decryption

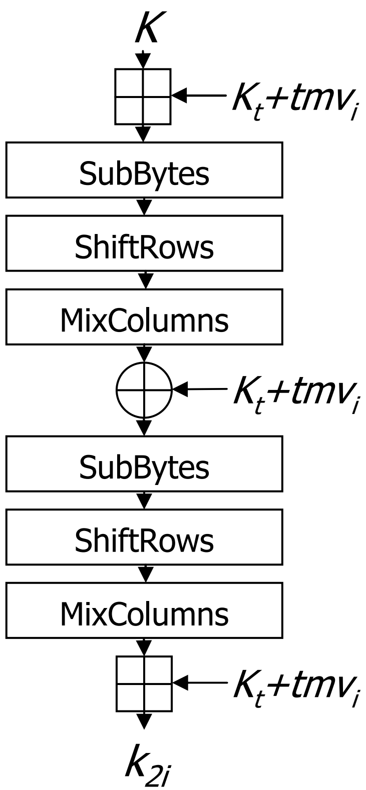
More effective implementation for CTR, OFB, CFB, CCM, GCM hashing, PRNG

Requirements to “Kalyna” key schedule

- non-linear dependence of every round key bit on every encryption key bit
- round key independence
- high computational complexity of encryption key recovery even having all round keys
- strength to all known cryptanalytic attacks on key schedule
- absence of weak key worsen cryptographic properties
- implementation simplicity (application of round transformation only)
- partial protection from side-channel attacks

"Kalyna" key schedule





tmv_0=0x01000100...0100

$tmv_{i+2}= tmv_i << 1$

All operations (excluding rotation and shifting which can be effectively implemented by memory access processor instructions) are taken from the encrypt function

$$k_{2i+1} = k_{2i} <<< (2 \cdot N_b + 3)$$

“Kalyna” key schedule properties

- correspondence to requirements
- all operations are taken from encryption function
- round keys can be generated in order to encryption and decryption with the same computational complexity
- effective countermeasure against round transformation symmetry
- minimal number of constants, their clearness
- key agility is less than 2.5
  (key schedule takes time less than 2.5 encryption of one block)
- non-bijective round keys dependency on encryption key

Non-bijective round key dependence

- implemented in
  - Twofish
    (AES competition finalist; key agility > 10)
  - Blowfish (widely used in public cryptographic libraries; key agility > 10)
  - Fox (block cipher developed in Switzerland; key agility > 5)
- key schedule works as PRNG with cryptographic properties
- no estimation was published in open literature

$$P\left(\#\{K\} \geq \#\{K_0, K_1, \dots, K_r\}\right)$$

Percentage of unique round keys for “Kalyna”

Block size	Key length	Part of unique round keys
128	128	0.997521
128	256	0.981684
256	256	0.999665
256	512	0.981684
512	512	0.999978

- Advantages:
  - good cryptographic properties
  - additional protection from different attacks, including side-channel
  - high computational complexity of encryption key recovery even having all round keys
- Disadvantage:
  - less than 2% of encryption keys might have equivalent keys (highly pseudorandom dependence of equivalent keys, if there are any)

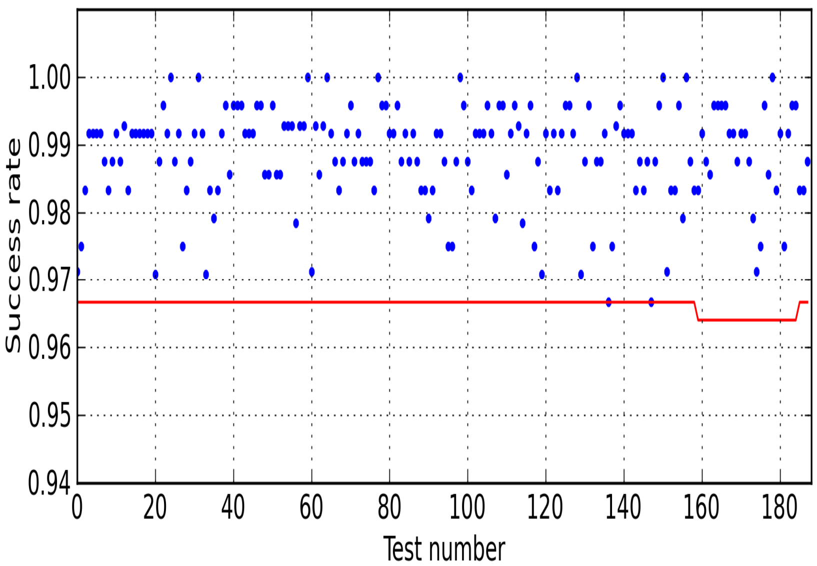
“Kalyna” (block size 128 bits, 10 rounds) strength to cryptanalytic attacks

Type of the attack	Min. rounds for prevention	Attack characteristics		
		Max. rnds	Encryptions	Memory
Differential	5	4	2^{55}	negligible
Linear	5	3	$2^{52,8}$	
Trunc. diff.	4			
Integral	6	5	2^{97}	2^{33+4}
Impos. diff.	6	5	2^{62}	2^{66}
Interpolation	3	2		
Boomerang	3	4	2^{120}	

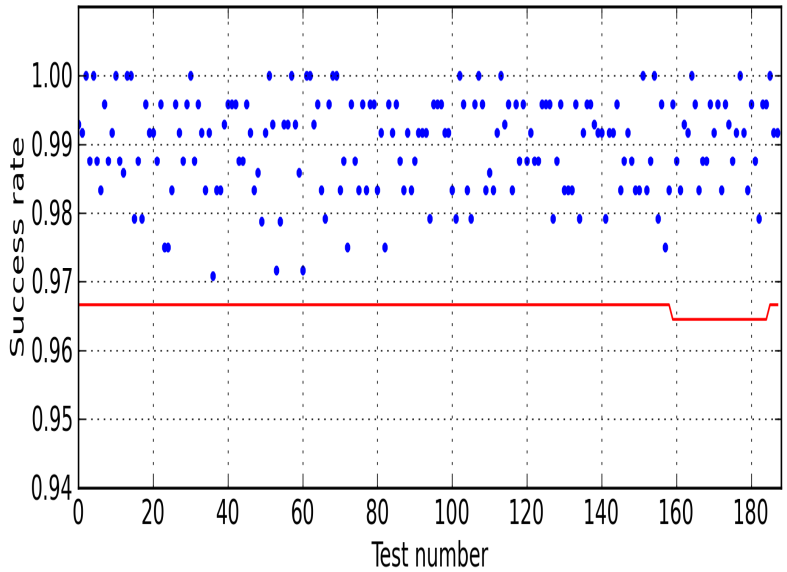
Similar results (enough security margin) are also obtained for 256 and 512-bit block

“Kalyna” output sequences NIST STS statistical testing

- Even round keys generation



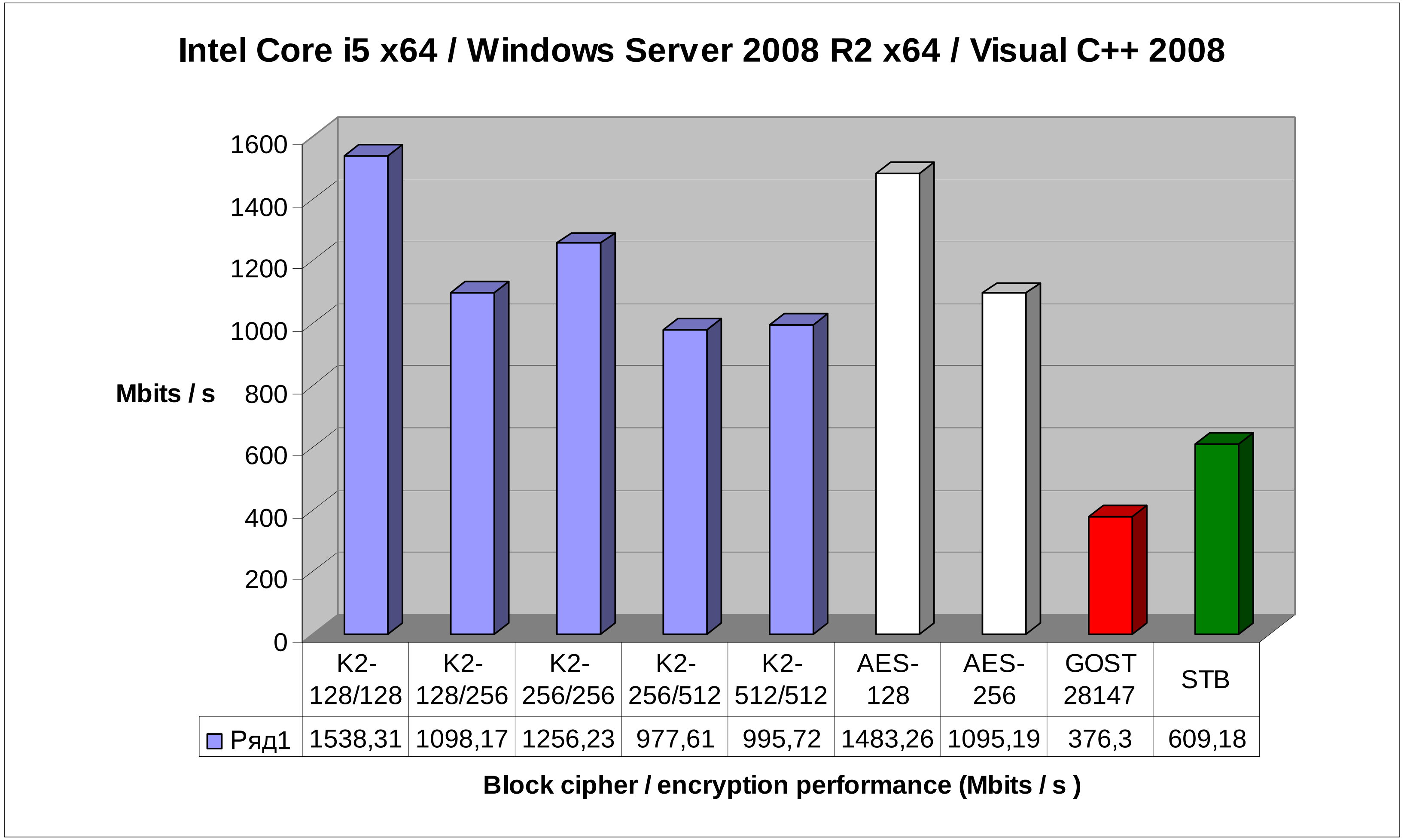
- CTR encryption



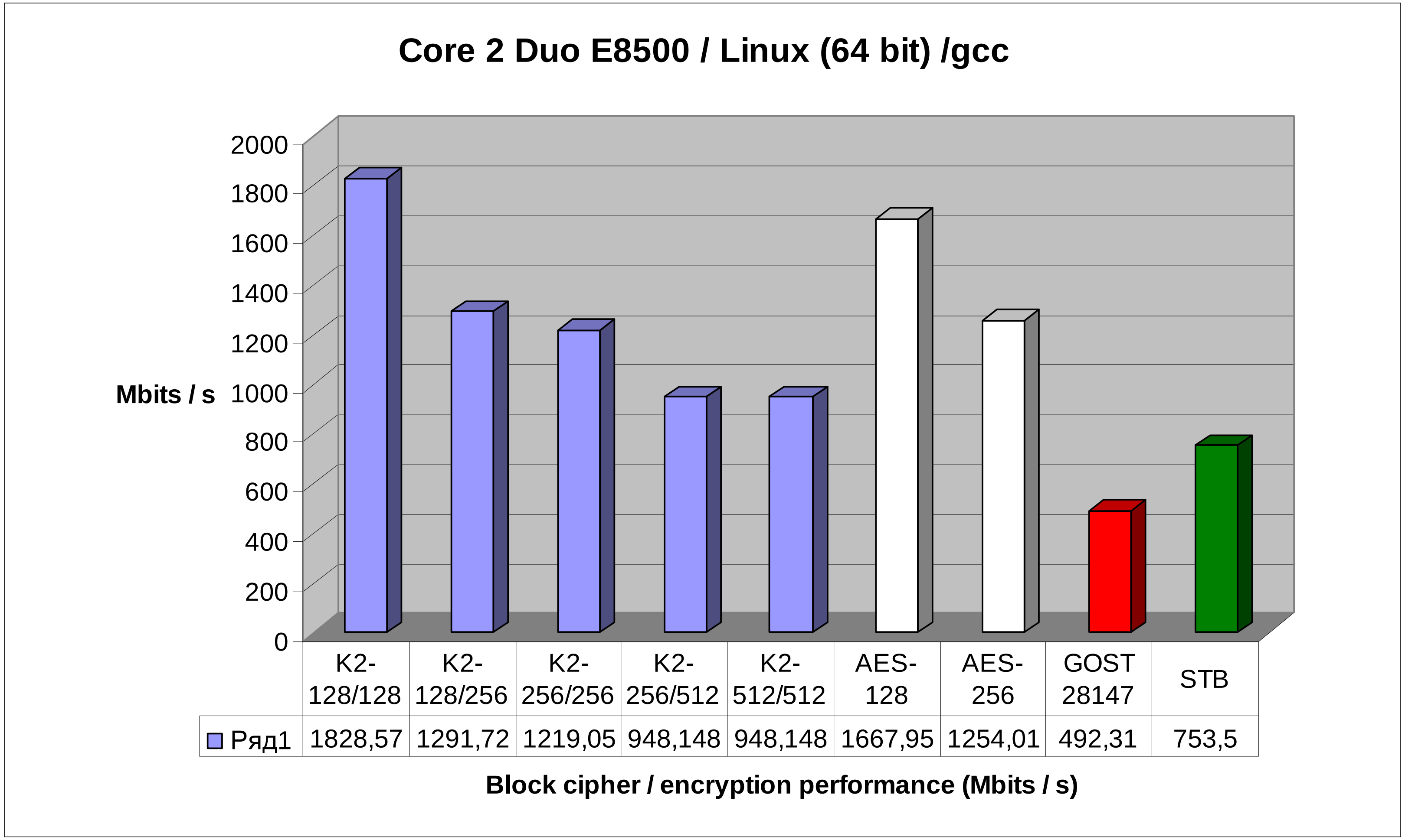
Comparison of required operations for one byte processing

- GOST 28147-89:
  - **72** operations/byte (32-bit memory accesses, modulo 2^{32} additions, XORs, ANDs, shifts)
- AES-128:
  - **45.375** operations/byte (32-bit memory accesses, XORs, ANDs, shifts)
- STB (Bel-T):
  - **40.5** operations/byte (32-bit memory accesses, modulo 2^{32} additions and subtractions, XORs, ANDs, shifts)
- Kalyna_128/128:
  - **40.375** operations/byte (64-bit memory accesses, modulo 2^{64} additions, XORs, ANDs, shifts)

Encryption performance (Windows/Visual C++)



Encryption performance (Linux/gcc)



Advantages of “Kalyna” block cipher

- has high and ultrahigh level of cryptographic security
- based on verified constructions and clear solutions
- fast on modern 64-bit processors
- compact software implementation
- perspective for application in common cryptographic systems, Internet and banking security, cloud computing security, etc.

Trends in block cipher development

- refusing of S-box algebraic structure (reverse element in the finite field, etc.)
- increasing size of MDS matrix
- families of ciphers: different block sizes and key lengths
- combinations of XOR and modular addition
- application of round function transformation for round key generation

Beyond block cipher security

- mode of operation security
  - BEAST attack (CBC mode)
- implementation security
  - CRIME/BREACH attacks
  - heartbleed bug in OpenSSL
  - timing attacks: cache misses
  - side-channel attacks
  - software vulnerabilities (buffer and heap overflows, etc.)
- high-level protocol security (e.g.: encryption key generation)

we need to use highly secure block ciphers, but should also pay a lot of attention to security of the whole system