ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Οικονομικό Πανεπιστήμιο Αθηνών
Τμήμα Πληροφορικής
ΠΜΣ στα Πληροφοριακά Συστήματα

Κρυπτογραφία και Εφαρμογές

Διαλέξεις Ακ. Έτους 2018-2019

Περίληψη

- ✓ Substitution-Permutation networks
- ✓ Feistel networks
- ✓ The Data Encryption Standard (DES)
  - Ιστορική αναδρομή
  - Περιγραφή
  - Τεχνικά χαρακτηριστικά
- ✓ Κρυπτανάλυση του DES
  - Γραμμική κρυπτανάλυση
  - Διαφορική κρυπτανάλυση
- ✓ Triple DES και άλλα παρεμφερή κρυπτοσυστήματα
- ✓ The Advanced Encryption Standard (AES)

Επιθυμητές ιδιότητες κρυπτοσυστημάτων

1.Σύγχυση (confusion): Η ικανότητα του αλγορίθμου ώστε ο αντίπαλος να μην μπορεί να προβλέψει ποιες μεταβολές θα συμβούν στο ciphertext αν αλλάξουμε κάτι στο plaintext

2.Διάχυση (diffusion) Το να μπορεί μία μικρή αλλαγή σε ένα τμήμα του plaintext να αλλάζει όσο το δυνατόν περισσότερο το ciphertext

Μοντέλα αξιολόγησης ασφάλειας:

- Unconditionally secure
- Computationally secure
- Secure under complexity-theoretic assumptions

Συμμετρική Κρυπτογραφία Τμημάτων

- Computationally secure Block Cipher
  - ✓ Στην προηγούμενη ενότητα είδαμε *unconditionally secure* κρυπτοσυστήματα (one-time pad)
  - ✓ Πρακτικά είναι ανεφάρμοστα
  - ✓ *computationally secure cipher:* καμία γνωστή επίθεση δεν έχει μικρότερη πολυπλοκότητα από (σχεδόν) brute force
  - ✓ Brute force:
    - data complexity: 2^n για n-bit plaintexts, exhaustive data analysis
    - processing complexity: 2^k για k-bit keys, exhaustive key search
  - ✓ Ως ελάχιστο, μέγεθος κλειδιού και μέγεθος block πρέπει να είναι ικανοποιητικά μεγάλα

Συμμετρική Κρυπτογραφία Τμημάτων

- Αξιολόγηση
  - ✓ *Μέγεθος τμήματος (block size).* Επηρεάζει
    - ασφάλεια (μεγάλα είναι επιθυμητά)
    - πολυπλοκότητα (μεγάλα είναι μη επιθυμητά)
    - απόδοση (π.χ., λόγω συχνών padding)
  - ✓ *Μέγεθος κλειδιού (key size).*
    - Παρόμοια trade-offs με το block size
  - ✓ *Πολυπλοκότητα συνάρτησης.* Επηρεάζει
    - κόστη υλοποίησης και πόρων (hardware, software),
    - απόδοση σε πραγματικό χρόνο (throughput).
  - ✓ *Error propagation*
    - Αποκρυπτογράφηση μπορεί να παράγει bit errors
    - Δυσάρεστα αποτελέσματα στο plaintext,
    - Αναπαραγωγή λαθών σε επόμενα plaintext blocks.

Δίκτυα Αντικατάστασης-Μετάθεσης

- Τα περισσότερα σύγχρονα συμμετρικά συστήματα είναι product ciphers
- Βασισμένα στις αρχές του Shannon περί μετασχηματισμών ανάμειξης (product transformations)
- Είσοδος: plaintext x, key K
- Σταθερός αριθμός από γύρους
- Round function: συνάρτηση που επιδρά πάνω στην έξοδο του προηγούμενου γύρου και παράγει την είσοδο για τον επόμενο
- Key schedule: αλγόριθμος που παράγει το κλειδί του κάθε γύρου, με βάση το αρχικό κλειδί K

- Συνήθως σε κάθε γύρο εφαρμόζεται η ίδια λειτουργία αλλά πάνω στο αποτέλεσμα του προηγούμενου γύρου

Δίκτυα Αντικατάστασης-Μετάθεσης

- Encryption με round function $g(\cdot, \cdot)$:
  - ✓ $w^0 = x$
  - ✓ $w^1 = g(w^0, K^1)$
  - ✓ $w^2 = g(w^1, K^2)$
  - ✓ ….
  - ✓ $w^r = g(w^{r-1}, K^r)$
  - ✓ $y = w^r$
- Η g πρέπει να είναι αντιστρέψιμη

- Decryption:
  - ✓ $w^r = y$
  - ✓ $w^{r-1} = g^{-1}(w^r, K^r)$
  - ✓ …
  - ✓ $w^0 = g^{-1}(w^1, K^1)$
  - ✓ $x = w^0$

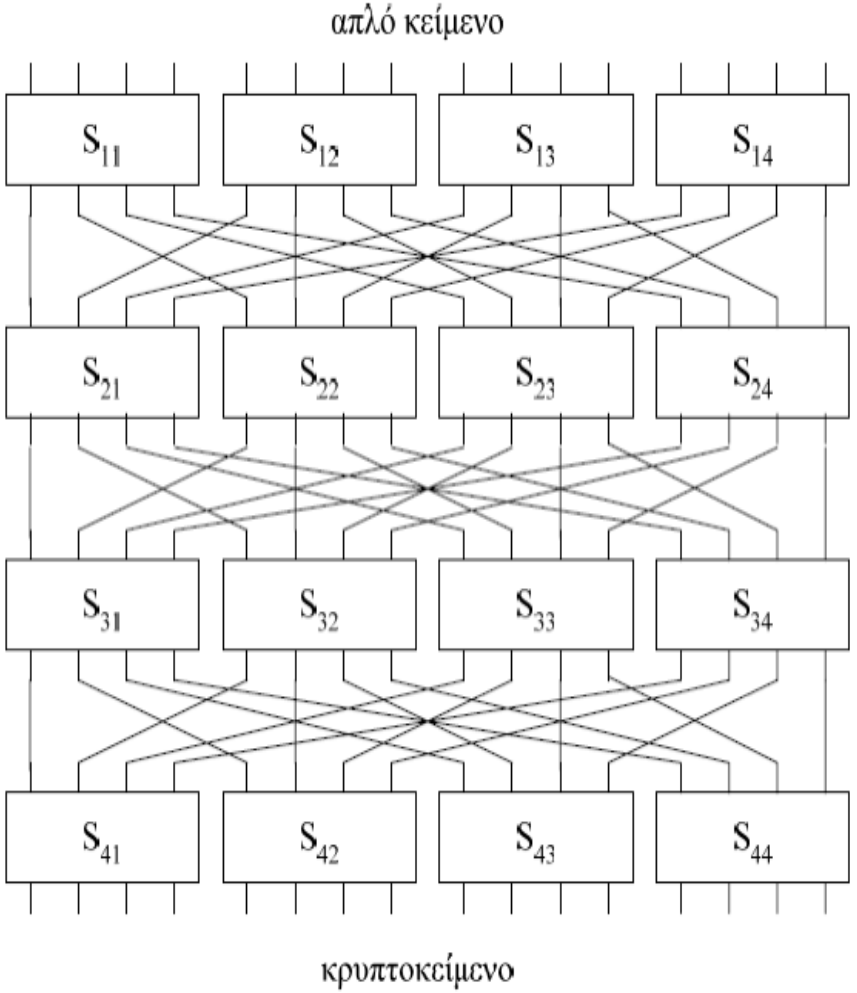
Δίκτυα Αντικατάστασης-Μετάθεσης

- Δίκτυα Αντικατάστασης-Μετάθεσης, (Substitution Permutation Networks, SPNs)

  - ✓ Feistel, Notz, Smith, Some cryptographic techniques for machine-to-machine data communications, 1975
  - ✓ Σε κάθε γύρο: 1 πράξη αντικατάστασης (τοπικά) και 1 πράξη μετάθεσης (ολικά)
  - ✓ Στόχος: υψηλή διάχυση και σύγχυση
    - διάχυση από διαδοχικά στάδια της μετάθεσης,
      - αλγόριθμος υψηλής διάχυσης όταν ένα στοιχειώδες τμήμα απλού κειμένου έχει τη δυνατότητα να επηρεάσει όλα τα τμήματα του κρυπτοκειμένου
    - σύγχυση από διαδοχικά στάδια της αντικατάστασης
      - αλγόριθμος υψηλής σύγχυσης όταν οι σχέσεις μεταξύ του απλού κειμένου και του κρυπτοκειμένου είναι αρκετά πολύπλοκες,

Κουτιά αντικατάστασης (S-boxes)

- ✓Αντιστοιχίζουν m bits σε t bits
- ✓$f : \{0, 1\}^m \rightarrow \{0, 1\}^t$
- ✓$(m \times t)$ *S-box*
- Ένας γύρος ορίζεται από μια σειρά από S-boxes, συνοδευόμενα από μια συνάρτηση μετάθεσης.
- Οι παράμετροι που ορίζουν ένα SPN είναι το μήκος της εισόδου *n*, ο αριθμός των γύρων *r*, και το μέγεθος των κουτιών αντικατάστασης (εδώ *m x m)*.
- Παράδειγμα: n = 16, r = 4, m = 4



Κουτιά αντικατάστασης (S-boxes)

Είσοδος i	$S(i)$	Είσοδος i	$S(i)$
000	1010	100	1111
001	1000	101	0010
010	0011	110	1110
011	0110	111	0010

Κουτί αντικατάστασης $\{0,1\}^3 \rightarrow \{0,1\}^4$

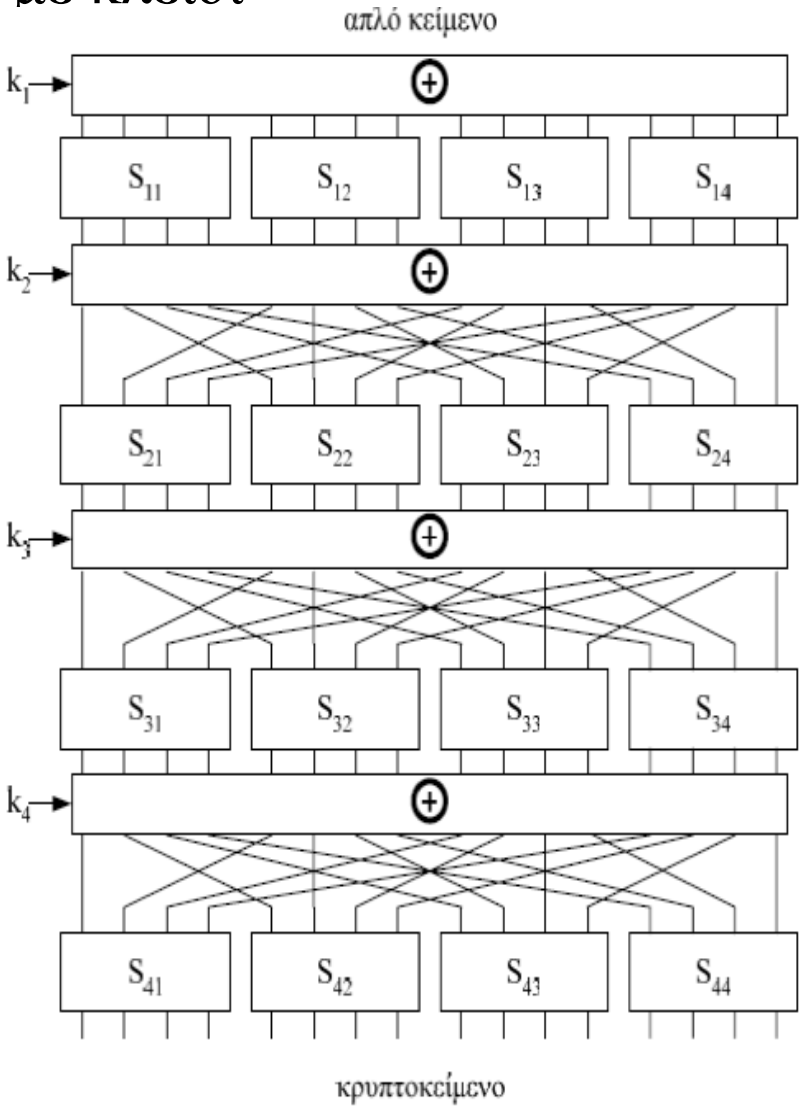
- ✓ Κριτήρια
  - **Μη γραμμικότητα**. Ένα S-box το οποίο είναι γραμμικό, μπορεί να κρυπταναλυθεί με μεγάλη ευκολία.
  - **Injective**. Απαραίτητο για να ορίζεται μονοσήμαντα η αποκρυπτογράφηση
  - **Αυστηρής χιονοστιβάδας** (strict avalanche criterion): για οποιοδήποτε bit εισόδου η αντιστροφή του έχει τη δυνατότητα να προκαλέσει αντιστροφή οποιουδήποτε bit της εξόδου, με πιθανότητα 0,5
  - **Ανεξαρτησία των bits της εξόδου.** Να μην εμφανίζονται σχέσεις μεταξύ των bits της εξόδου, π.χ. «το bit i είναι το AND του bit j και του bit k, με πιθανότητα 0,95»

Κουτιά αντικατάστασης (S-boxes)

- ✓ Παράμετροι ασφάλειας
  - Μέγεθος του κουτιού (σχέση m και t)
    - Πιο καίριο κριτήριο η εξασφάλιση της μη γραμμικότητας
    - αν $t \geq 2^m$-m, υπάρχει γραμμική σχέση μεταξύ bits εξόδου με την είσοδο
    - αν $m \geq 2^t$, υπάρχουν γραμμικές σχέσεις μεταξύ των bits της εξόδου
    - Πολλές φορές m = t (π.χ. AES)
    - Στο DES, m = 6, t = 4
  - Ο τρόπος συμπλήρωσης των τιμών της συνάρτησης του S-box
    - Για μικρά κουτιά η συμπλήρωση με τυχαίες τιμές είναι κρυπτογραφικά αδύναμη
    - Για πιο μεγάλα S-boxes, οι τυχαίες τιμές μπορούν να συντηρήσουν μη γραμμικότητα
    - Στους περισσότερους block ciphers τα κουτιά που χρησιμοποιούνται είναι σχετικά μικρά
  - Διαφοροποίηση μεταξύ των S-boxes
    - Συνήθως σε κάθε επανάληψη χρησιμοποιούνται τα ίδια S-boxes
    - Είναι διαφορετικά όμως για κάθε τμήμα του plaintext
    - Στο παράδειγμα, $S_{1i} = S_{2i} = S_{3i} = S_{4i}$, i = 1,…4

- Δίκτυα Αντικατάστασης-Μετάθεσης με κλειδί

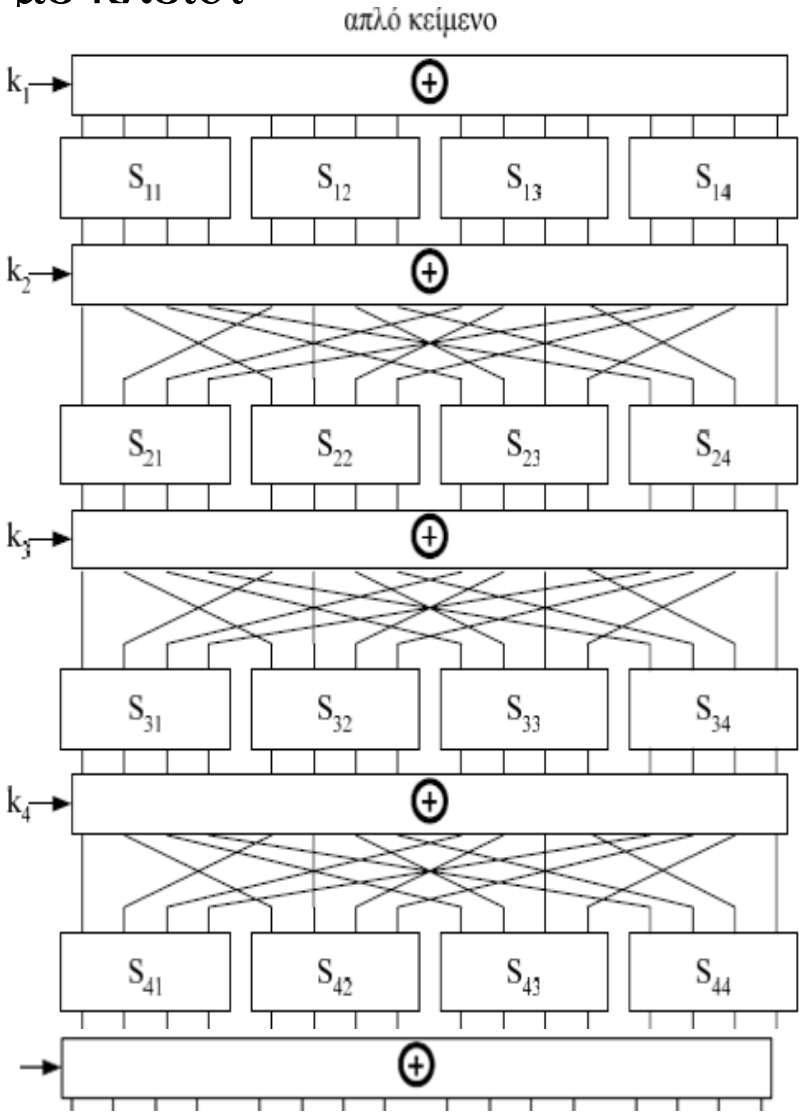
  - *m* × *m S-boxes*
  - Η αντικατάσταση είναι και αυτή μία μετάθεση (τοπική)
  - σε κάθε γύρο χρησιμοποιούνται *n*/*m* κουτιά.
  - Το κρυπτοσύστημα αποτελείται από *r* γύρους,
  - στο κρυπτοσύστημα εκτελούνται 3*r*-2 κρυπτογραφικά γινόμενα



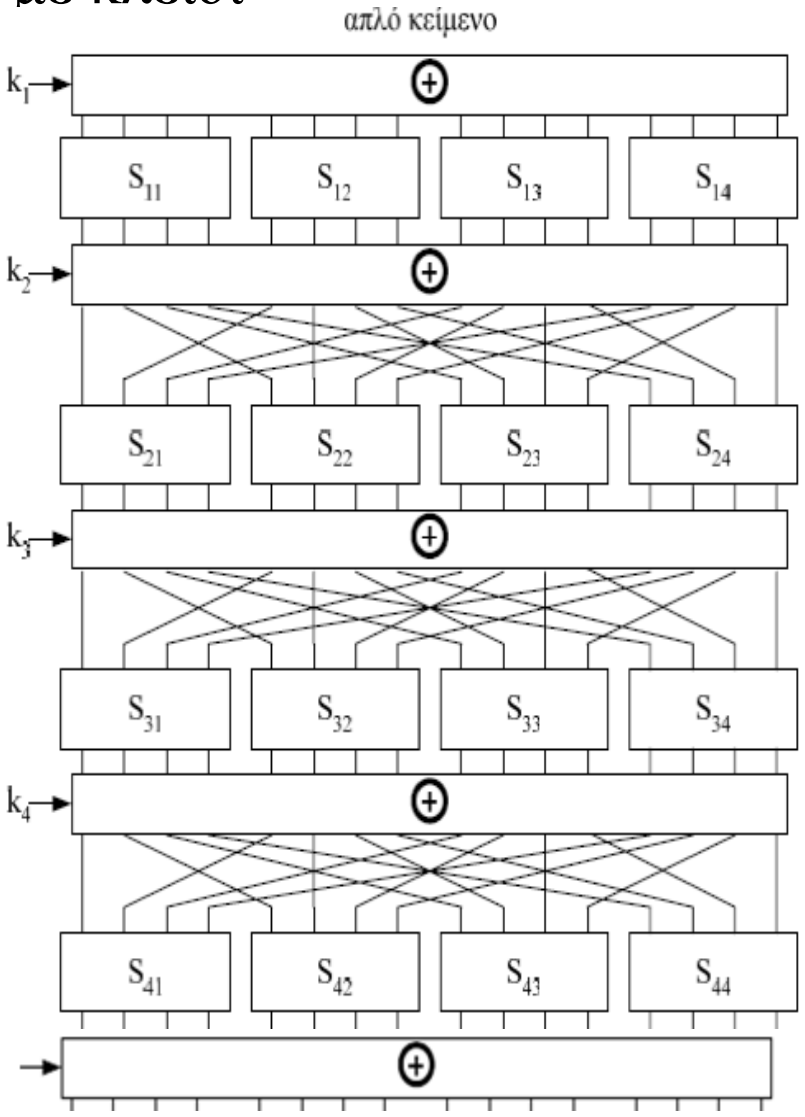
- Δίκτυα Αντικατάστασης-Μετάθεσης με κλειδί
  - Formally:
  - $\pi_S : \{0, 1\}^m \rightarrow \{0, 1\}^m$ *(S-box)*
  - $\pi_p : \{0, 1\}^n \rightarrow \{0, 1\}^n$
  - Plaintext x χωρίζεται σε n/m κομμάτια
  - $x = x_{<1>} \| x_{<2>} \| \ldots \| x_{<n/m>}$
  - Έστω w^j το αποτέλεσμα στο τέλος του γύρου j

```
w^0 = x
for j = 1 to r {
    u^j = w^(j-1) ⊕ K_j
    v^j = π_p(u^j)
    for i = 1 to m
        w^j_<i> = π_S(v^j_<i>)
}
y = w^r ⊕ K_(r+1)
```

  - ✓ Στο τέλος προαιρετικά κάνουμε άλλο ένα XOR (λεύκανση-whitening)



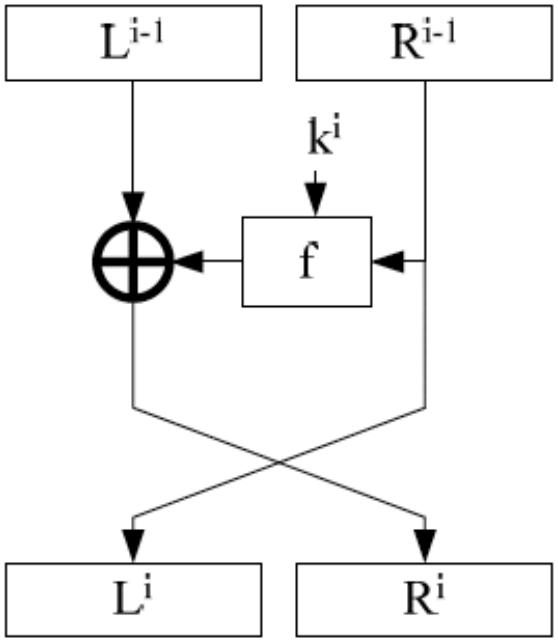
- Δίκτυα Αντικατάστασης-Μετάθεσης με κλειδί
  - Πλεονεκτήματα:
    - Απλότητα
    - Εύκολη υλοποίηση σε hardware και software
    - Σε software: look-up table
    - Σε hardware: αποθήκευση 2^m τιμών
    - Σε hardware implementations, τα κουτιά πρέπει να είναι σχετικά μικρά
    - Πρακτικά παρέχουν καλό επίπεδο ασφάλειας αν οι παράμετροι ρυθμιστούν κατάλληλα



Δίκτυα Feistel

- Feistel, H. 1973. Cryptography and Computer Privacy. Scientific American. 228(5): 15-23
- Δομές Feistel
  - ✓ Στηρίζονται και αυτές σε substitutions, permutations και λειτουργίες XOR
  - ✓ Δομή χιονοστιβάδας (avalanche)
    - "As the input moves through successive layers the pattern of 1's generated is amplified and results in an unpredictable avalanche. In the end the output will have, on average, half 0's and half 1's"
  - ✓ Ανεπαίσθητη αλλαγή στο input: Παράγει πολλαπλές αλλαγές στον 1ο κύκλο, περισσότερες στο 2ο, κοκ
  - ✓ Τελικά το μισό block αλλάζει κατά μέσο όρο

Δίκτυο **r** επιπέδων



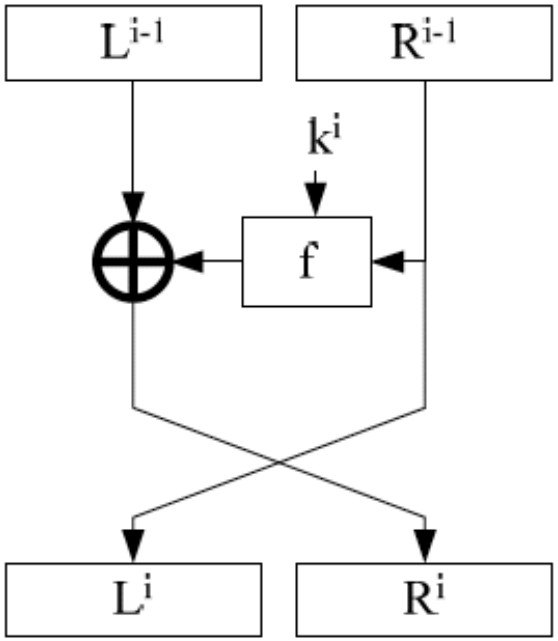
- Είσοδος:
  - ✓ Το plaintext χωρίζεται σε δύο τμήματα, $x = L \,||\, R$ (Left, Right)
  - ✓ Κλειδί K
- *f:* ίδια για κάθε κύκλο

- Είσοδος σε κάθε κύκλο i: (L^{i-1}, R^{i-1}) (έξοδοι προηγούμενου)
- Υπόκλειδί K_i παράγεται από το K
- $K_i \neq K_j$, *για* $i \neq j$

- Έξοδος κύκλου i: (L^i, R^i) όπου:
  - ✓ $L^i = R^{i-1}$
  - ✓ $R^i = L^{i-1} \oplus f(R^{i-1}, K_i)$

Δίκτυα Feistel

Παρατηρήσεις:



1. Συνήθως το αριστερό και το δεξιό τμήμα έχουν το ίδιο μέγεθος (balanced network)

2. Η συνάρτηση f δεν χρειάζεται να είναι αντιστρέψιμη
   - ✓ Ο συνολικός μετασχηματισμός είναι αντιστρέψιμος ανεξάρτητα από την f
   - ✓ Από το (L^i, R^i), μπορούμε να ανακτήσουμε το (L^{i-1}, R^{i-1}) αν έχουμε το κλειδί:
   - ✓ $L^{i-1} = R^i \oplus f(L^i, K_i)$
   - ✓ $R^{i-1} = L^i$

Data Encryption Standard (DES)

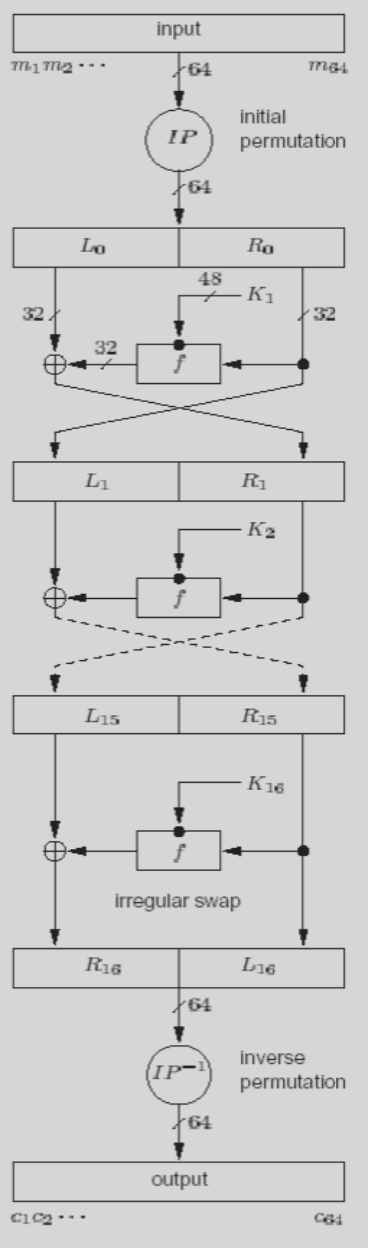
History of DES:

- May 15, 1973: National Bureau of Standards (now National Institute of Standards and Technology, NIST) δημοσιεύει call for proposals for cryptosystems
- 1973-1975: Το DES σχεδιάστηκε από την IBM ως τροποποίηση παλιότερου συστήματος του Feistel (Lucifer)
  - ✓ Το Lucifer είχε 128-bit keys
  - ✓ Το DES 56-bit keys μετά και από υποδείξεις της NSA
- 1977: Το DES υιοθετείται ως FIPS 46 (Federal Information Procesing Standard)
  - ✓ Πολλές κριτικές για τις προδιαγραφές ασφαλείας
  - ✓ Εικάζεται ότι δεν θα αντέξει πάνω από 10-15 χρόνια
- 1983: DES reaffirmed από τη NIST
- 1992: Biham και Shamir, 1η θεωρητικά επιτυχημένη επίθεση με λιγότερη πολυπλοκότητα από brute force:
  - ✓ differential cryptanalysis. Απαιτεί 2^{47} chosen plaintexts. Ανεδαφικό

History of DES:

- 1994: Πρώτη πειραματική κρυπτανάλυση του DES
  - ✓ linear cryptanalysis (Matsui, 1994), απαιτεί 2^{43} known plaintexts
- 1997: Το DESCHALL Project σπάει ένα DES encrypted μήνυμα για 1η φορά παρουσία κοινού
- 1998: «έσπασε» σε 56 ώρες με μηχανή αξίας 250K$
  - ✓ EFF's DES cracker (Deep Crack)
- 1999: «έσπασε» σε 1 μέρα
  - ✓ EFF's DES cracker (Deep Crack) + distributed.net
- 1999: Τελευταία ανανέωση του DES ως FIPS 46-3 με προτιμώμενη χρήση του triple-DES

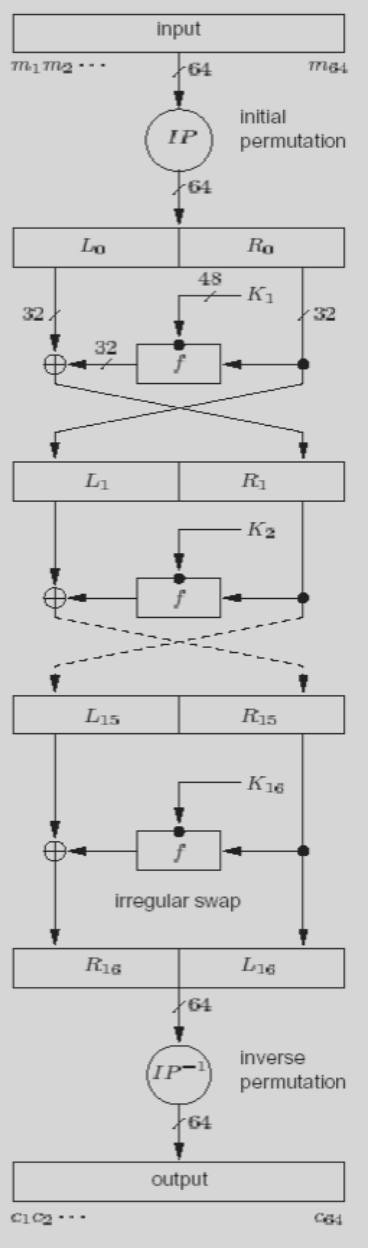
Data Encryption Standard (DES)



- ✓ Δίκτυο Feistel **r = 16** επιπέδων
- ✓ Κρυπτογραφεί blocks των 64 bits
- ✓ Κλειδί **K** = 56 bits, Key Space = 2^{56}

Βήματα:

- ✓ Αρχική Μετάθεση του plaintext (**IP**, initial permutation)
  - ✓ Σχετίζεται με την αρχική υλοποίηση σε hardware
  - ✓ Χωρίς κρυπτογραφική σημασία
- ✓ Διαχωρισμός σε δύο τμήματα, **L** και **R** (Left, Right)
- ✓ 16 επαναλήψεις της **g (round function),** ίδια για κάθε κύκλο
  - ✓ Σε κάθε κύκλο, subkey $\mathbf{K_i}$ των 48 bits που παράγεται από το **K**
- ✓ Με το πέρας του 16ου κύκλου αντιμετατίθενται τα R_{16} και L_{16}
- ✓ Εφαρμόζεται η αντίστροφη μετάθεση $\mathbf{IP^{-1}}$
- ✓ Τελικό ciphertext: $IP^{-1}(R_{16}, L_{16})$, μήκους 64 bits



Round function g

- ✓ Είσοδος σε κάθε κύκλο **i**
  - ✓ **L_{i-1}, R_{i-1}** (έξοδοι προηγούμενου κύκλου), 32 bits το καθένα
  - ✓ Υπο-κλειδί **K_i** των **48 bits** που παράγεται από το **K**
- ✓ Έξοδος του κύκλου i: $(L_i, R_i) = g(L_{i-1}, R_{i-1}, K_i)$

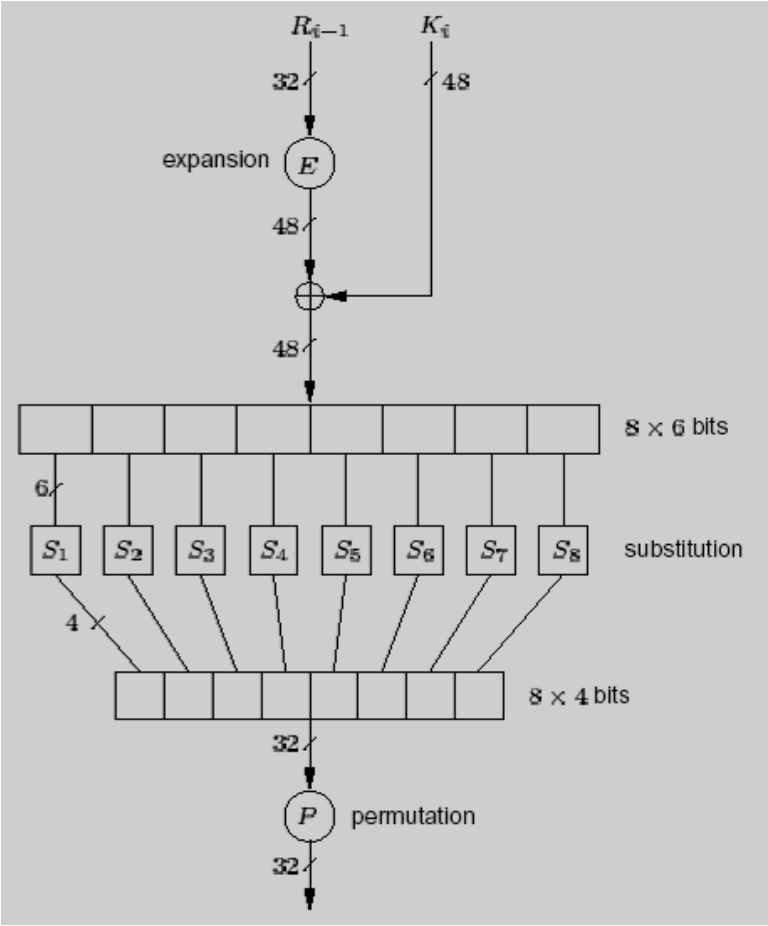
$$L_i = R_{i-1} \text{ και } R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

$$f(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i))$$

- ✓ f: $\{0,1\}^{32} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32}$
- ✓ P: μετάθεση των 32 bits P: $\{0,1\}^{32} \rightarrow \{0,1\}^{32}$
- ✓ S: Substitution, λειτουργία 8 S-boxes σε 8 6-bit strings
- ✓ E: expansion function E: $\{0,1\}^{32} \rightarrow \{0,1\}^{48}$

Data Encryption Standard (DES)

Η συνάρτηση $f(R_{i-1},K_i) = P(S(E(R_{i-1}) \oplus K_i))$



DES f function at step i

- ✓ **E:** Το R_{i-1} (στον κύκλο i) επεκτείνεται από τα 32 στα 48 bits.
  - ✓ Εμπεριέχει μετάθεση του R_{i-1} και επανάληψη κάποιων bits
- ✓ $E(R_{i-1}) \oplus K_i$ (48 bits)
- ✓ Τα 48 bits χωρίζονται σε οκτώ 6-bit strings
- ✓ Λειτουργίες αντικατάστασης:
  - ✓ 8 *S-boxes* S_i: $\{0, 1\}^6 \rightarrow \{0, 1\}^4$
  - ✓ Μετά τα S-boxes: 32 bits
- ✓ **P** είναι μετάθεση των 32 bits
- ✓ Οι **E**, **S**, **P** πραγματοποιούνται με χρήση καθορισμένων πινάκων επέκτασης, αντικατάστασης και αντιμετάθεσης.

Data Encryption Standard (DES)

DES step by step

INPUT: plaintext $m_1 \ldots m_{64}$

Βήμα 1. Initial Permutation (IP)

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

Left part → 32 symbols, $L_0 = m_{58}m_{50}m_{42} \ldots m_{16}m_8$

Right part → 32 symbols, $R_0 = m_{57}m_{49}m_{41} \ldots m_{15}m_7$

$(L_0,R_0) \leftarrow IP(m_1m_2...m_{64})$.

Βήμα 2. Expansion Table

Στον κύκλο i επεκτείνεται το R_{i-1} από τα 32 στα 48 bits

Προκύπτουν οκτώ εξάδες χαρακτήρων

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

$E(R_0) = [m_7m_{57}m_{49}m_{41}m_{33}m_{25}][m_{33}m_{25}\ldots]\ldots[\ldots m_7m_{57}]$

Για κάθε i, αν $R_{i-1} = r_1r_2...r_{32}$
$T_i \leftarrow E(R_{i-1})$, άρα $T = r_{32}r_1r_2r_3 \ldots r_{32}r_1$

Βήμα 3. XOR

$T'_i \leftarrow T \oplus \mathbf{K_i}$. Το T' μπορεί να γραφεί ως οκτώ 6-bit strings T'= B1… B8

row	column number															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
S_1																
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																
[0]	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
[1]	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
[2]	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
[3]	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3																
[0]	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
[1]	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
[2]	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
[3]	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4																
[0]	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
[1]	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
[2]	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
[3]	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5																
[0]	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
[1]	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
[2]	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
[3]	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6																
[0]	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
[1]	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
[2]	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
[3]	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7																
[0]	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
[1]	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
[2]	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
[3]	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8																
[0]	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
[1]	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
[2]	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
[3]	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

DES step by step

Βήμα 4. S-box substitutions

$U_i = (S_1(B_1), S_2(B_2), \ldots S_8(B_8))$

Κάθε S-box χρειάζεται να έχει αποθηκευμένες 64 τιμές (= 2^6)

Αποθήκευση σε 4x16 πίνακα

Ένα block $\mathbf{B_i=b_1b_2\ldots b_6}$ αντιστοιχίζεται στο 4-bit string της γραμμής r και της στήλης c, όπου:

- Η γραμμή προσδιορίζεται από τα $\mathbf{b_1}$ και $\mathbf{b_6}$
  - $\mathbf{r = 2*b_1+b_6}$,
- Η στήλη από το $\mathbf{c = b_2b_3b_4b_5}$, $\mathbf{0 \le c \le 15}$

Παράδειγμα: Έστω το κουτί S_1 και το string B = 011011.

- r = 01, 1η γραμμή
- c = 1101 = 13,

Άρα $S_1(B) = 5$, σε binary 0101

Data Encryption Standard (DES)

row	column number															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
S_1																
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																
[0]	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
[1]	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
[2]	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
[3]	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3																
[0]	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
[1]	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
[2]	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
[3]	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4																
[0]	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
[1]	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
[2]	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
[3]	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5																
[0]	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
[1]	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
[2]	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
[3]	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6																
[0]	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
[1]	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
[2]	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
[3]	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7																
[0]	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
[1]	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
[2]	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
[3]	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8																
[0]	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
[1]	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
[2]	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
[3]	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Τα κριτήρια σχεδιασμού των S-boxes

- Μυστικά για περίπου 20 χρόνια
- Coppersmith. IBM *Journal of Research and Development,* 1994.
- Κανένα από τα bits της εξόδου δεν θα πρέπει να βρίσκεται σε γραμμική σχέση με οποιοδήποτε από τα bits της εισόδου.
- Αν τα δύο πρώτα bits και τα δύο τελευταία bits της εισόδου είναι σταθερά ενώ τα ενδιάμεσα bits αλλάζουν, οι έξοδοι που προκύπτουν θα πρέπει να είναι μοναδικές.
- Αν η Hamming distance δύο εισόδων είναι ίση με 1, τότε η απόσταση Hamming των αντίστοιχων εξόδων θα πρέπει να είναι το λιγότερο ίση με 2

Data Encryption Standard (DES)

row	column number															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
S_1																
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																
[0]	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
[1]	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
[2]	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
[3]	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3																
[0]	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
[1]	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
[2]	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
[3]	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4																
[0]	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
[1]	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
[2]	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
[3]	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5																
[0]	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
[1]	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
[2]	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
[3]	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6																
[0]	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
[1]	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
[2]	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
[3]	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7																
[0]	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
[1]	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
[2]	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
[3]	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8																
[0]	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
[1]	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
[2]	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
[3]	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Τα κριτήρια σχεδιασμού των S-boxes

- Αν δύο είσοδοι διαφέρουν στα δύο μεσαία bits, τότε οι αντίστοιχες έξοδοι θα πρέπει να διαφέρουν το λιγότερο σε 2 bits.
- Αν δύο είσοδοι έχουν τα δύο πρώτα bits διαφορετικά ενώ τα δύο τελευταία bits είναι ίδια, τότε οι αντίστοιχες έξοδοι θα πρέπει να είναι διαφορετικές.
- Για οποιαδήποτε μη μηδενική διαφορά των 6 bits της εισόδου, θα πρέπει το πολύ 8 από τα 32 ζευγάρια να προκαλούν την ίδια διαφορά εξόδου.
- Όμοια με το παραπάνω κριτήριο, αλλά θα πρέπει να εφαρμόζεται συγχρόνως σε οποιαδήποτε 3 από τα 8 κουτιά αντικατάστασης.
- Για κάθε S-box, κάθε γραμμή του πίνακα είναι permutation

DES step by step

Βήμα 5. Permutation P

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

$V_i = P(U_i)$ Αντιμετάθεση με βάση τον πίνακα

Αν $U_i = u_1u_2...u_{32}$, τότε $V_i = u_{16}u_7...u_{25}$

Βήμα 6. Αντιμετάθεση των τελικών blocks των 32 bits R_{16}, L_{16} και συνένωση

$b_1b_2\ldots b_{64} \leftarrow (R_{16}, L_{16})$

Βήμα 7. Ανάστροφο Permutation IP^{-1}

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

$C \leftarrow IP^{-1}(b_1b_2\ldots b_{64})$

Ciphertext : $(b_{40}b_8\ldots b_{25})$

DES step by step
Key schedule

Key Schedule

Βήμα 1. **Επιλέγεται Κ 56 bits + 8 parity bits**

Βήμα 2. **Διαχωρισμός Κ**

Χρήση Permutation Choice Table 1

PC1: $\{0, 1\}^{64} \rightarrow \{0, 1\}^{56}$

T = PC1(K) και χωρίζεται σε δύο 28-bit τμήματα (C_0, D_0)

T: $C0=k_{57}k_{49}...k_{36}$, $D0=k_{63}k_{55}... k_4$

8 bits $(k_8, k_{16}, \ldots k_{64})$ έχουν απορριφθεί

PC1						
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
above for C_i; below for D_i						
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

Βήμα 3. **Σε κάθε κύκλο i:**

1. Γίνεται αριστερή ολίσθηση κατά s_i bits στα C_{i-1}, και D_{i-1} ($s_i = 1$ για $i \in \{1, 2, 9, 16\}$, και $s_i = 2$ διαφορετικά)
Συνολικός αριθμός ολισθήσεων: 28 (για να επανέλθουμε στο (C_0, D_0))
2. Χρήση Permutation Choice Table 2

PC2: $\{0, 1\}^{56} \rightarrow \{0, 1\}^{48}$

Μετάθεση όπου ορισμένα bits αγνοούνται

PC2					
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

DES step by step
Key schedule

Συνοπτικά:

Βήμα 3.

for i=1 to 16 {

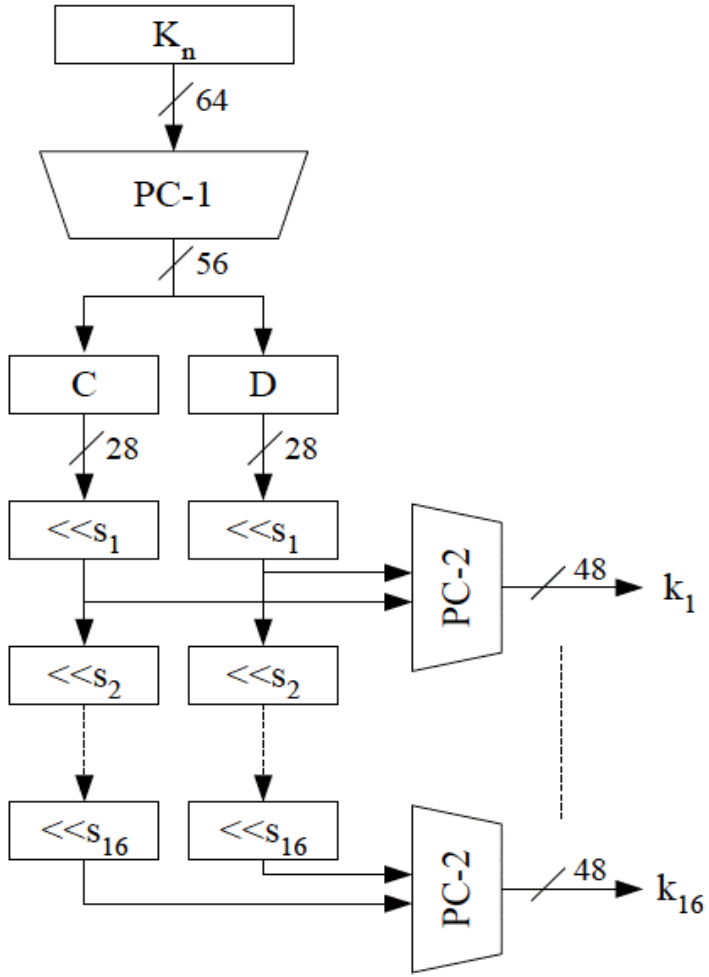
$C_i \leftarrow (C_{i-1} << s_i)$

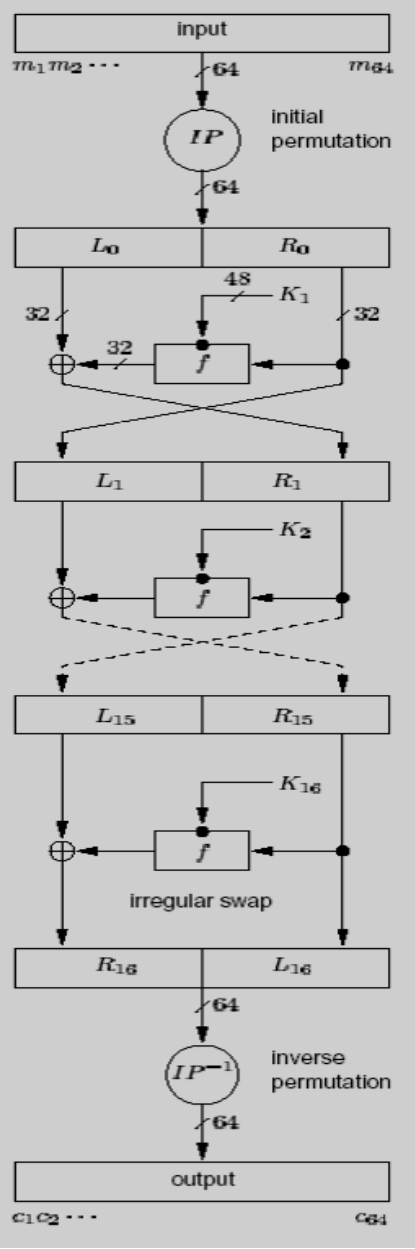
$D_i \leftarrow (D_{i-1} << s_i)$

$K_i \leftarrow PC2(C_i, D_i)$

}

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





Decryption

✓ Χρησιμοποιείται το ίδιο κλειδί K και τα ίδια υποκλειδιά αλλά με την ανάστροφη σειρά, $K_{16}, K_{15}, \ldots, K_1$

- ✓ Δεξιές αντί για αριστερές ολισθήσεις για το key schedule

✓ Η επίδραση της IP^{-1} ακυρώνεται από την IP κατά την πρώτη φάση της αποκρυπτογράφησης. Έτσι γίνονται γνωστά τα (R_{16}, L_{16})

✓ Recall $L_{16} = R^{15}$, $R^{16} = L^{15} \oplus f(R^{15}, K_{16})$

✓ Άρα έχουμε το R^{15} και επίσης $L^{15} = R^{16} \oplus f(L^{16}, K_{16})$

✓ Έτσι ο 1ος κύκλος αντιστρέφει τον κύκλο16 της διαδικασίας κρυπτογράφησης

✓ Ο 2ος κύκλος αποκαλύπτει τα (R_{14}, L_{14})

✓ …

✓ Στο τέλος χρειαζόμαστε ξανά την IP^{-1}

Κρυπτανάλυση του DES

- Ιδιότητες και Αδυναμίες DES
  - ✓ Αδύναμα κλειδιά
  - ✓ 56 bits δίνουν πολύ μικρό key space (2^{56}) για σημερινά δεδομένα
  - ✓ Αρχικά υπέστη μεγάλη κριτική και για το μικρό μέγεθος των S-boxes
    - Τελικά όμως η επιλογή έγινε με ιδιαίτερη προσοχή
  - ✓ Από οποιοδήποτε round key μπορεί να βρεθεί το αρχικό κλειδί
  - ✓ Υπάρχουν κλειδιά με τα οποία το key schedule για encryption και decryption είναι ίδια: $K_1 = K_{16}$, $K_2 = K_{15}$, κοκ.
    - Τότε encryption και decryption συμπίπτουν
  - ✓ Ορισμός: Ένα DES weak key είναι ένα κλειδί K τέτοιο ώστε $E_K(E_K(x))=x$ για κάθε x.
  - ✓ Ορισμός: Ένα ζεύγος κλειδιών (K_1,K_2) αποτελεί ζεύγος από semi-weak keys αν έχει την ιδιότητα ότι $E_{K1}(E_{K2}(x)) = x$.
    - Κρυπτογράφηση με ένα κλειδί του ζεύγους αποκρυπτογραφεί τη λειτουργία του άλλου.
  - ✓ Το DES έχει 4 weak keys και 6 ζεύγη semi-weak keys.

- Ιδιότητες και Αδυναμίες DES
  - ✓ Weak και semi-weak keys

weak key (hexadecimal)	C_0	D_0
0101 0101 0101 0101	$\{0\}^{28}$	$\{0\}^{28}$
FEFE FEFE FEFE FEFE	$\{1\}^{28}$	$\{1\}^{28}$
1F1F 1F1F 0E0E 0E0E	$\{0\}^{28}$	$\{1\}^{28}$
E0E0 E0E0 F1F1 F1F1	$\{1\}^{28}$	$\{0\}^{28}$

4 DES weak keys

✓Κλειδιά με όλα τα bits 0 ή 1

✓Κανένα αποτέλεσμα η κυκλική ολίσθηση

C_0	D_0	semi-weak key pair (hexadecimal)	C_0	D_0
$\{01\}^{14}$	$\{01\}^{14}$	01FE 01FE 01FE 01FE, FE01 FE01 FE01 FE01	$\{10\}^{14}$	$\{10\}^{14}$
$\{01\}^{14}$	$\{10\}^{14}$	1FE0 1FE0 0EF1 0EF1, E01F E01F F10E F10E	$\{10\}^{14}$	$\{01\}^{14}$
$\{01\}^{14}$	$\{0\}^{28}$	01E0 01E0 01F1 01F1, E001 E001 F101 F101	$\{10\}^{14}$	$\{0\}^{28}$
$\{01\}^{14}$	$\{1\}^{28}$	1FFE 1FFE 0EFE 0EFE, FE1F FE1F FE0E FE0E	$\{10\}^{14}$	$\{1\}^{28}$
$\{0\}^{28}$	$\{01\}^{14}$	011F 011F 010E 010E, 1F01 1F01 0E01 0E01	$\{0\}^{28}$	$\{10\}^{14}$
$\{1\}^{28}$	$\{01\}^{14}$	E0FE E0FE F1FE F1FE, FEE0 FEE0 FEF1 FEF1	$\{1\}^{28}$	$\{10\}^{14}$

6 DES semi-weak keys pairs

Κρυπτανάλυση του DES

- Ιδιότητες και Αδυναμίες DES
  - ✓ Εξειδικευμένες επιθέσεις
  - ✓ Δύο επιθέσεις είναι γνωστό ότι παραβιάζουν τον DES με λιγότερη πολυπλοκότητα από brute-force
  - ✓ differential cryptanalysis (DC)
    - Τέλη δεκαετίας 1980s (Biham και Shamir).
    - Για παραβίαση των 16 κύκλων απαιτεί 2^{57} chosen plaintexts
    - Για παραβίαση 15 κύκλων απαιτεί 2^{51} chosen plaintexts
  - ✓ linear cryptanalysis (LC)
    - 1993 από M. Matsui,
    - Χρειάζεται 2^{47} chosen plaintexts
  - ✓ Οι επιθέσεις αυτές ήταν τότε μη πρακτικές για να αναπτυχθούν ρεαλιστικά
    - Στοιχεία από το http://www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf

Κρυπτανάλυση του DES

- Γραμμική κρυπτανάλυση
  - ✓ Είναι known plaintext attack
    - Υποθέτουμε ότι ο Oscar έχει μεγάλο πλήθος από ζεύγη plaintext-ciphertext
  - ✓ Βασική ιδέα: έκφραση ορισμένων bits εισόδου, ορισμένων bits εξόδου και ορισμένων bits του κλειδιού με γραμμική προσέγγιση
    - Παράδειγμα γραμμικής σχέσης: το XOR των bits 1 και 3 του plaintext ισούται με το XOR των bits 2, 5 και 7 του ciphertext
    - Αν το σύστημα είχε τέλεια μυστικότητα, η πιθανότητα να ισχύει μία τέτοια σχέση θα ήταν 1/2
    - Κρυπτοσύστημα ευάλωτο σε γραμμική κρυπτανάλυση όταν βρεθεί γραμμική σχέση που ισχύει με πιθανότητα διαφορετική του 1/2.
    - Στόχος: ανίχνευση πολλών γραμμικών σχέσεων που ισχύουν με πιθ/τες διαφορετικές από1/2
    - Ποιο είναι το μη γραμμικό κομμάτι του DES? Τα S-boxes
    - Η μέθοδος κάνει ένα linear approximation των S-boxes

- Γραμμική Κρυπτανάλυση
  - ✓ Έστω δύο τυχαίες, binary, μεταβλητές, X_1 και X_2.
  - ✓ $X_1 \oplus X_2 = 0$ είναι γραμμική σχέση και ισοδύναμη με $X_1 = X_2$.
  - ✓ $X_1 \oplus X_2 = 1$ είναι συγγενική και ισοδύναμη με $X_1 \neq X_2$.
  - ✓ Έστω πυκνότητες πιθανότητας

$$\Pr(X_1 = i) = \begin{cases} p_1 & ,i = 0 \\ 1 - p_1 & ,i = 1 \end{cases} \qquad \Pr(X_2 = i) = \begin{cases} p_2 & ,i = 0 \\ 1 - p_2 & ,i = 1. \end{cases}$$

  - ✓ Τότε για ανεξάρτητες X_1, X_2

$$\Pr(X_1 = i, X_2 = j) = \begin{cases} p_1 p_2 & ,i = 0, j = 0 \\ p_1(1 - p_2) & ,i = 0, j = 1 \\ (1 - p_1)p_2 & ,i = 1, j = 0 \\ (1 - p_1)(1 - p_2) & ,i = 1, j = 1 \end{cases}$$

  - ✓ $\Pr(X_1 \oplus X_2 = 0) = \Pr(X_1 = X_2) = \Pr(X_1 = 0, X_2 = 0) + \Pr(X_1 = 1, X_2 = 1) = p_1 p_2 + (1-p_1)(1-p_2)$

Γραμμική Κρυπτανάλυση

- Ορισμός: Η πόλωση (bias) μίας δυαδικής τυχαίας μεταβλητής X_i είναι
$$\varepsilon_i = p_i - 1/2 = \Pr[X_i = 0] - 1/2$$
  - Είναι η απόκλιση από το να έχουμε ισοπίθανα ενδεχόμενα, $-\frac{1}{2} \le \varepsilon_i \le +\frac{1}{2}$
- Έστω 2 ανεξάρτητες μεταβλητές με: $p_1 = \frac{1}{2} + \varepsilon_1$ και $p_2 = \frac{1}{2} + \varepsilon_2$
- Έτσι $\Pr(X_1 \oplus X_2 = 0) = \frac{1}{2} + 2\varepsilon_1\varepsilon_2$
- Επομένως η πόλωση $\varepsilon_{1,2}$ για τη μεταβλητή $X_1 \oplus X_2$ είναι $\varepsilon_{1,2} = 2\varepsilon_1\varepsilon_2$
- Επέκταση σε n τυχαίες μεταβλητές:
- Piling-up Lemma: Έστω οι ανεξάρτητες τυχαίες μεταβλητές $X_1, X_2, \ldots, X_n$ με πολώσεις $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$ αντίστοιχα. Τότε η πόλωση της μεταβλητής $Y = X_1 \oplus X_2 \oplus \ldots \oplus X_n$ θα είναι:
$$\varepsilon_{1,2,\ldots,n} = 2^{n-1} \prod_{i=1}^{n} \varepsilon_i$$
  - Πόρισμα: Αν $\exists$ i: $\varepsilon_i = 1/2$, τότε $\varepsilon_{1,2,\ldots,n} = 0$ (αν μία μεταβλητή δεν έχει πόλωση, τότε και το XOR δεν έχει πόλωση)
  - Προσοχή: το piling-up lemma είναι μόνο για ανεξάρτητες μεταβλητές

Γραμμική Κρυπτανάλυση

✓ Παράδειγμα

- Έστω 3 ανεξάρτητες δυαδικές τυχαίες μεταβλητές X_1, X_2, X_3
- Έστω $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = 1/4$
- Piling up lemma $\Rightarrow \varepsilon_{1,2} = \varepsilon_{1,3} = \varepsilon_{2,3} = 1/8$
- Έστω η μεταβλητή $X_1 \oplus X_3$
- $X_1 \oplus X_3 = (X_1 \oplus X_2) \oplus (X_2 \oplus X_3)$
- Οι $X_1 \oplus X_2$ και $X_2 \oplus X_3$ δεν είναι ανεξάρτητες
- Αν ήταν, το Piling-Up Lemma θα έδινε $\varepsilon_{1,3} = 2\,(1/8)^2 = 1/32$ ενώ είδαμε ότι είναι 1/8

Κρυπτανάλυση του DES

- Γραμμική κρυπτανάλυση
  - ✓ Ο Matsui έδωσε μέθοδο εύρεσης γραμμικών προσεγγίσεων με μέγιστη πόλωση στα S-boxes του DES
  - ✓ Η μέθοδος μπορεί να εφαρμοσθεί σε οποιοδήποτε S-box
  - ✓ Έστω S-box S: $\{0, 1\}^m \rightarrow \{0, 1\}^n$
  - ✓ Θεωρούμε κάθε bit ως τυχαία μεταβλητή $\Rightarrow$ έχουμε συνολικά $m + n$ τυχαίες μεταβλητές
  - ✓ Θεωρούμε ότι όλες οι είσοδοι από το $\{0, 1\}^m$ είναι ισοπίθανες
  - ✓ Όσες μεταβλητές αντιστοιχούν στα bits της εισόδου είναι μεταξύ τους ανεξάρτητες και με μηδενική πόλωση
  - ✓ Στην έξοδο, οι μεταβλητές που αντιστοιχούν στα bits της εξόδου εξαρτώνται από τα bits εισόδου

Κρυπτανάλυση του DES

- Γραμμική κρυπτανάλυση

  - ✓ bits εισόδου: τυχαίες μεταβλητές $P_1, P_2, \ldots, P_m$
  - ✓ bits εξόδου: τυχαίες μεταβλητές $C_1, C_2, \ldots, C_n$.
  - ✓ Αριθμός πιθανών εισόδων: 2^m

- Γραμμική κρυπτανάλυση
  - ✓ Μπορούμε να κοιτάξουμε οποιονδήποτε γραμμικό συνδυασμό των bits εισόδου με τα bits της εξόδου.
  - ✓ Κάθε γραμμική σχέση αντιστοιχεί σε μία τυχαία μεταβλητή της μορφής

$$Z(a,\ b) = \left(\bigoplus_{i=1}^{m} a_i P_i \right) \oplus \left(\bigoplus_{i=1}^{n} b_i C_i \right),$$

  - ✓ όπου $a = (a_1, \ldots, a_m)$, $b = (b_1, \ldots, b_n)$, $a_i, b_i \in \{0, 1\}$ καθορίζουν το συνδυασμό των bits εισόδου και εξόδου που συμμετέχουν στη γραμμική σχέση
  - ✓ Ο συνολικός αριθμός γραμμικών σχέσεων είναι 2^{m+n}.

- Γραμμική κρυπτανάλυση
  - ✓ Μας ενδιαφέρει η πόλωση της μεταβλητής $Z(a, b)$
  - ✓ Δηλαδή: *η συχνότητα με την οποία το XOR επιλεγμένων bits εισόδου είναι ίσο με το XOR επιλεγμένων bits εξόδου*
  - ✓ Ορισμός: Για δεδομένο S-box $S:\{0,1\}^m \rightarrow \{0,1\}^n$, και $a = (a_1, \ldots, a_m)$, $b = (b_1, \ldots, b_n)$, με $a_i, b_i \in \{0,1\}$, η ποσότητα ***NS a,b*** ισούται με:

$$\left|\{(P_1,\ldots,P_m, C_1,\ldots,C_n): (C_1,\ldots,C_n) = S(P_1,\ldots,P_m), \text{ and } \oplus a_i P_i = \oplus b_i C_i\}\right|$$

  - ✓ Η πόλωση της μεταβλητής $Z(a, b)$ θα είναι:

$$\varepsilon(a,b) = \frac{N S(a,b) - 2^{m-1}}{2^m}.$$

Χρησιμότερες γραμμικές σχέσεις: εκείνες όπου η ε(a,b) είναι μακριά από το 0

Γραμμική κρυπτανάλυση

- Παράδειγμα εύρεσης βέλτιστων γραμμικών προσεγγίσεων
- Έστω S-box. S_4: $\{0, 1\}^4 \rightarrow \{0, 1\}^3$
- Δείκτης a : ορίζει δυαδικές μεταβλητές της εισόδου, παίρνει τιμές από 0 έως 15
- Δείκτης b : ορίζει δυαδικές μεταβλητές της εξόδου, τιμές από 0 έως 7
- Η δυαδική απεικόνιση φανερώνει τα επιλεγμένα bits στη γραμμική σχέση.
  - Πχ , αν $a = (5)_{10} = (0101)_2$ ο a αντιστοιχεί στο άθροισμα $P1 \oplus P3$.

Πίνακας του S-box

P_1	P_2	P_3	P_4	C_1	C_2	C_3
0	0	0	0	1	0	1
0	0	0	1	1	1	0
0	0	1	0	0	0	0
0	0	1	1	1	1	0
0	1	0	0	0	1	1
0	1	0	1	1	1	0
0	1	1	0	1	1	0
0	1	1	1	1	1	1
1	0	0	0	0	0	0
1	0	0	1	0	1	1
1	0	1	0	1	0	1
1	0	1	1	0	1	0
1	1	0	0	1	1	0
1	1	0	1	1	0	1
1	1	1	0	0	1	1
1	1	1	1	1	0	1

Γραμμική κρυπτανάλυση

- Παράδειγμα εύρεσης βέλτιστων γραμμικών προσεγγίσεων
- Μπορούμε να υπολογίσουμε την ποσότητα $NS(a, b)$ για όλες τις τιμές των a, b.
- Υπάρχουν γραμμικές προσεγγίσεις οι οποίες απέχουν από τη μέση τιμή
  - Εδώ μέση τιμη = 8
- Μέγιστη απόσταση = 6, συμβαίνει για τις τιμές (10,2), (10,5) και (13,4).
- Βέλτιστες γραμμικές σχέσεις που προκύπτουν είναι
  - $P2 \oplus P4 = C2$,
  - $P2 \oplus P4 = C1 \oplus C3$ και
  - $P1 \oplus P3 \oplus P4 = C3$,
  - Με πόλωση 3/8 ή -3/8

Τιμές της $NS(a, b)$

a \ b	0	1	2	3	4	5	6	7
0	16	8	8	8	8	8	8	8
1	8	6	6	12	10	6	8	8
2	8	10	10	8	10	10	8	8
3	8	6	10	8	8	8	6	10
4	8	8	8	8	8	6	8	10
5	8	8	8	4	6	8	6	8
6	8	8	8	8	6	8	10	8
7	8	4	8	8	8	10	4	6
8	8	10	10	8	8	8	10	10
9	8	10	10	4	6	10	8	8
10	8	6	14	8	6	14	8	8
11	8	10	6	8	8	8	10	6
12	8	8	8	8	8	10	8	6
13	8	8	8	4	2	8	10	8
14	8	8	8	8	10	8	6	8
15	8	4	8	8	8	6	4	10

Κρυπτανάλυση του DES

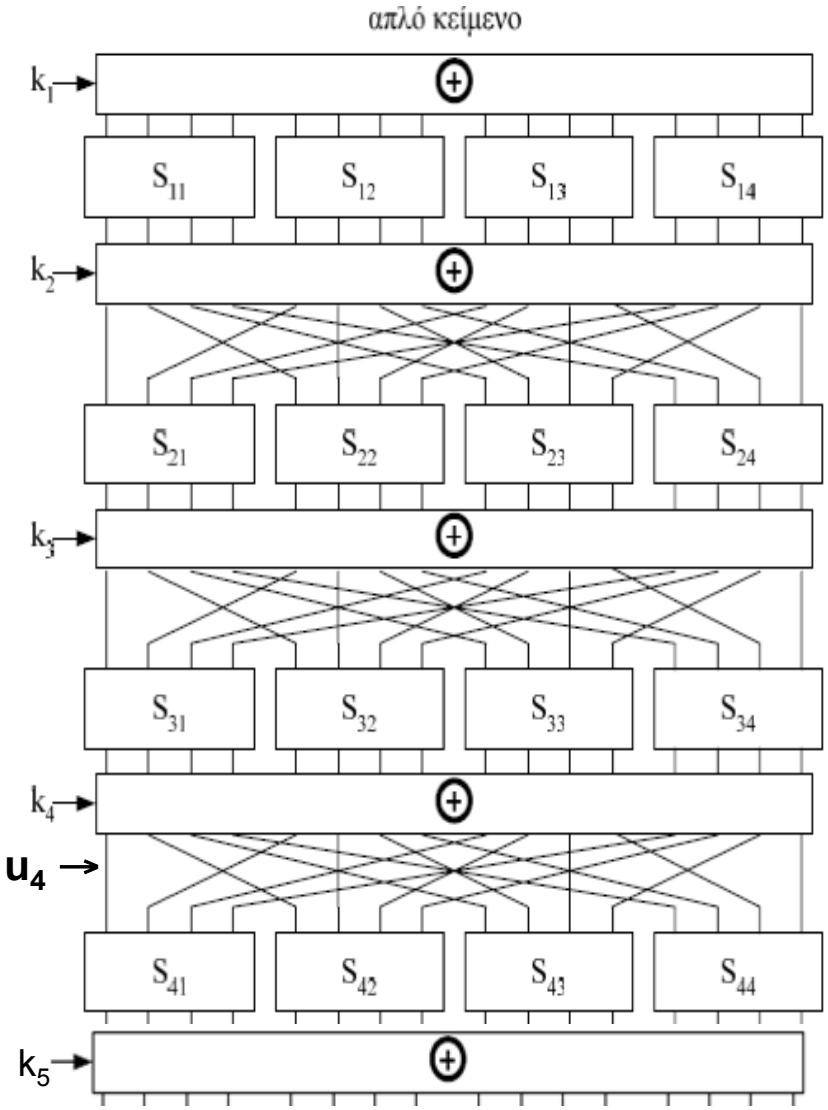
- Γραμμική κρυπτανάλυση
  - ✓ Στο DES, αρκεί να μάθουμε το κλειδί του τελευταίου γύρου
  - ✓ **Ιδέα:** έστω ότι έχουμε βρει γραμμικές σχέσεις με μέγιστη πόλωση σε S-boxes που καλύπτουν όλους τους γύρους εκτός τον τελευταίο
  - ✓ Αν αυτές οι σχέσεις μπορούν να συνδεθούν προς τα πάνω σε όλους τους γύρους, τότε:
    - Από piling up lemma θα έχουμε (προσεγγιστικά καθώς δεν έχουμε ανεξαρτησία) μία γραμμική σχέση με πόλωση μακριά από το 0 (έστω **ε**), που θα περιλαμβάνει
      - κάποια plaintext bits (μάσκα στο input)
      - καποια bits απο το ξεκίνημα του τελευταίου γύρου (μάσκα «κοντά» στο output) πριν εφαρμοστεί το XOR με το τελευταίο κλειδί.
  - ✓ Έστω ότι έχουμε πολλά ζεύγη plaintext-ciphertext, ας πούμε T το πλήθος.

Κρυπτανάλυση του DES

- Γραμμική κρυπτανάλυση
  - ✓ Μπορούμε να δοκιμάσουμε όλους τους συνδυασμούς για τα bits του κλειδιού που αντιστοιχούν στη μάσκα εξόδου. Για κάθε συνδυασμό μετράμε πόσες φορές (στα T ζεύγη plaintext-ciphertext) το XOR της γραμμικής σχέσης είναι 0.
  - ✓ Για όλους τους συνδυασμούς που δεν αντιστοιχούν στο σωστό κλειδί, περιμένουμε η εξίσωση να ισχύει με πιθανότητα ½
  - ✓ Αν για κάποιον συνδυασμό βρούμε ότι η σχέση ισχύει με πιθανότητα **½±ε,** εικάζουμε ότι έχουμε μαντέψει τα σωστά bits.
  - ✓ Δοκιμάζοντας και άλλες μάσκες μπορούμε να βρούμε και τα υπόλοιπα bits
    - Με πόλωση **ε** χρειαζόμαστε περίπου c/ε^2 ζεύγη plaintext-ciphertext κρυπτογραφημένα με το ίδιο κλειδί, για κάποια μικρή σταθερά c, $c \approx 8$
  - ✓ Έτσι υπολογίζονται round key bits των εξώτερων κύκλων και μετά των εσώτερων

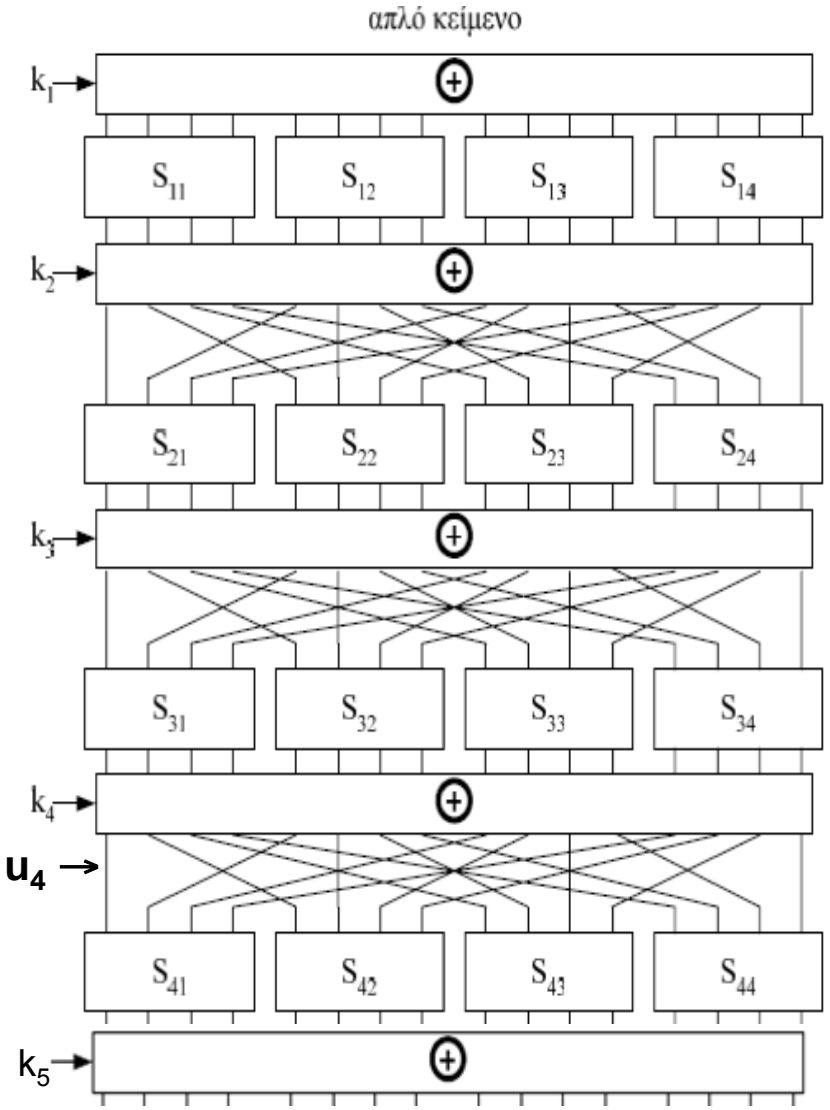
Γραμμική κρυπτανάλυση

- ✓ Παράδειγμα με SPN
- ✓ u^4 : string αμέσως μετά την εφαρμογή του K_4
- ✓ Έστω ότι έχουμε βρει σχέσεις για τα S_{12}, S_{22}, S_{32}, S_{34}, με καλή πόλωση
  - Γενικά ψάχνουμε για κάποιο **«μονοπάτι»** σχέσεων από πάνω προς τα κάτω
- ✓ Από Piling up lemma, το XOR όλων των μεταβλητών που συμμετέχουν στις σχέσεις θα έχει σχετικά καλή πόλωση
  - Όχι ακριβώς, γιατί δεν είναι ανεξάρτητες οι μεταβλητές
  - Αλλά μας δίνει μία προσέγγιση
- ✓ Αν επιλέξουμε προσεκτικά τις γραμμικές σχέσεις, κάνοντας αντικατάσταση ενδιάμεσων μεταβλητών, η σχέση που τελικά προκύπτει θα περιέχει
  - Plaintext bits
  - Bits κλειδιών από όλους τους γύρους
  - Κάποια bits του u^4



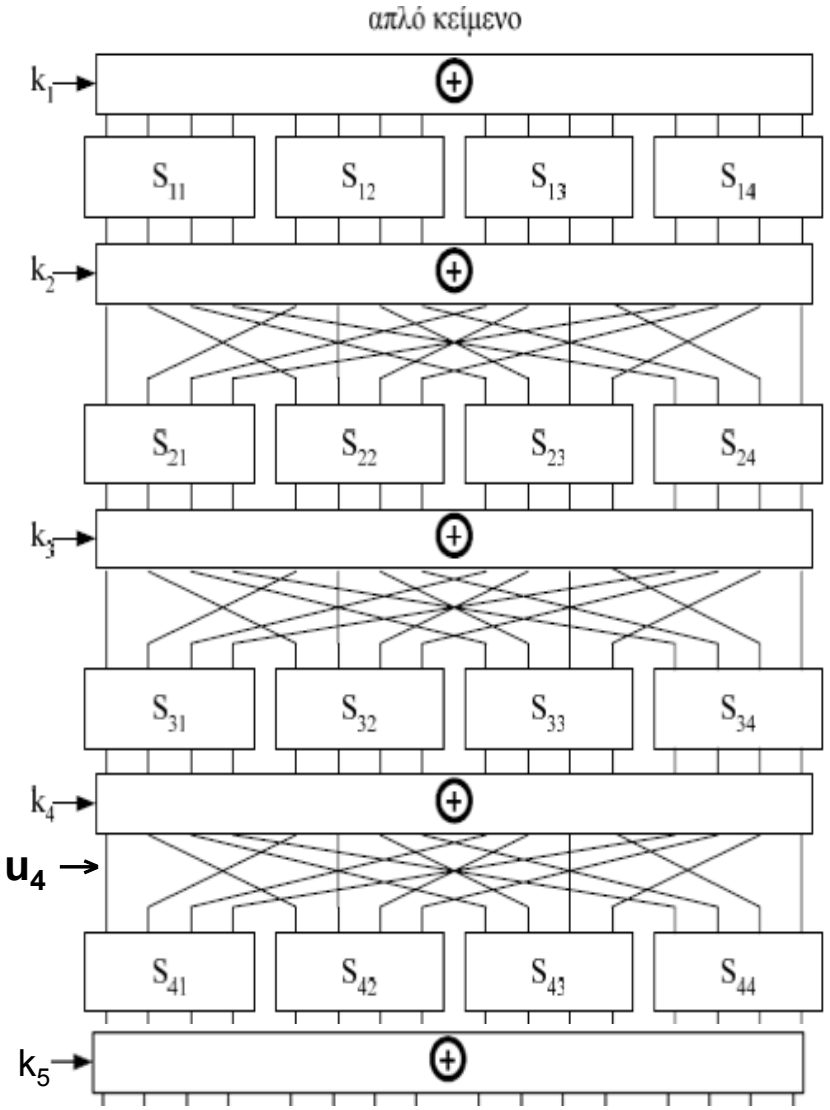
Γραμμική κρυπτανάλυση

- ✓ Τα bits που συμμετέχουν από κλειδιά των προηγούμενων γύρων έχουν κάποια fixed τιμή (ανεξάρτητη από τα T ζεύγη plaintext-ciphertext)
- ✓ Έστω Z το XOR των υπολοίπων bits
- ✓ Αν το piling up lemma έδωσε ε, τότε η Z θα έχει προσεγγιστικά πόλωση +ε ή -ε
  - Ανάλογα με την τιμή που έχει το XOR των bits των κλειδιών από τους προηγούμενους γύρους
- ✓ Για τα bits του u_4 που συμμετέχουν στη Z:
  - Αν έχουμε ένα ciphertext, ξέρουμε ποια bits του κλειδιού k_5 πρέπει να προσδιορίσουμε για να φτάσουμε σε αυτά
  - Π.χ. εδώ έστω ότι χρειαζόμαστε το 2ο και 4ο block του k_5 (8 bits)



Γραμμική κρυπτανάλυση

- ✓ Δοκιμάζουμε όλους τους συνδυασμούς για τα σχετικά bits του k_5 (π.χ. 2^8)
- ✓ Για καθένα από τα T ζεύγη plaintext-ciphertext μετράμε πόσες φορές η Z είναι 0
- ✓ Για τα σωστά bits του k_5 περιμένουμε ο μετρητής να δείξει περίπου $T/2 + \varepsilon T$ ή $T/2 - \varepsilon T$
- ✓ Έχοντας προσδιορίσει έτσι μερικά από τα bits του k_5 συνεχίζουμε με άλλα μονοπάτια γραμμικών σχέσεων για να βρούμε και τα υπόλοιπα



Κρυπτανάλυση του DES

- Διαφορική κρυπτανάλυση
  - ✓ Biham και Shamir (1991)
  - ✓ Είναι chosen plaintext attack
  - ✓ Έχει κάποια κοινά χαρακτηριστικά με τη γραμμική κρυπτανάλυση
  - ✓ Εξετάζει κατά πόσο διαφορές (differences) στο input επηρεάζουν τις επακόλουθες διαφορές στο output
  - ✓ Διαφορά δύο κειμένων : XOR
  - ✓ Βασίζεται στο γεγονός ότι η κατανομή των διαφορών μεταξύ κρυπτοκειμένων δεν είναι ομοιόμορφη
    - ορισμένες διαφορές εμφανίζονται περισσότερο από άλλες διαφορές
    - η κατανομή εξαρτάται από το κλειδί.
  - ✓ Η μέθοδος ήταν μάλλον γνωστή στους σχεδιαστές του DES

- Διαφορική κρυπτανάλυση
  - ✓ Έστω σύστημα με κρυπτογραφική πράξη e_k
    - Θα το δούμε αρχικά για ένα γύρο
    - π.χ. ο τελευταίος γύρος του DES
  - ✓ Η διαφορά 2 input m-bit strings, x και x* είναι: x' = x ⊕ x*
  - ✓ Η διαφορά x' μπορεί να επιτευχθεί με 2^m ζεύγη
    - Παίρνοντας για κάθε x, το x* = x' ⊕ x
  - ✓ Όταν η διαφορά εισόδου (input XOR) είναι x', η διαφορά εξόδου (output XOR) δεν ακολουθεί την ίδια κατανομή
  - ✓ Ορισμός: Για διαφορά εισόδου x', η σχετική συχνότητα εμφάνισης της διαφοράς εξόδου y', $N_D e_k(x', y')$, είναι:

$$N_D e_k(x', y') = \left| \{x, x^* : 0 \le x \le 2^m - 1, x^* = x \oplus x', e_k(x) \oplus e_k(x^*) = y'\} \right|$$

- Διαφορική κρυπτανάλυση

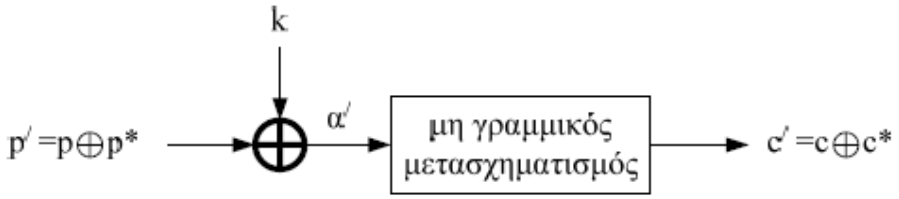
  - ✓ Από την $N_D e_k$ μπορούμε να υπολογίσουμε την πιθανότητα εμφάνισης κάποιας διαφοράς εξόδου y' δεδομένης διαφοράς εισόδου x':

$$\Pr\{y' \mid x'\} = \frac{N_D e_k(x', y')}{2^m},$$

  - ✓ Το ζευγάρι (x', y') ονομάζεται ***διαφορικό χαρακτηριστικό (differential)***

  - ✓ Στην διαφορική κρυπτανάλυση ψάχνουμε για κατάλληλα μονοπάτια διαφορικών χαρακτηριστικών (***differential trails***)

- Διαφορική κρυπτανάλυση
  - ✓ Στην περίπτωση που το κλειδί του γύρου συνδυάζεται με την είσοδο με XOR, η πιθανότητα των διαφορικών χαρακτηριστικών είναι ανεξάρτητη από τα bits του κλειδιού
    - Αν p, p' 2 plaintexts (ή οι έξοδοι από προηγούμενο γύρο), τότε το input XOR σε ένα S-box θα είναι $a' = (p \oplus k) \oplus (p^* \oplus k) = p \oplus p^*$
  - ✓ Άρα πλεονέκτημα στον αντίπαλο
    - Με plaintext attacks, ο αντίπαλος γνωρίζει τα p, p^*, p', a', c, c^* και c' (output XOR)
    - μπορεί να βρει bits του k από τα δυνατά a, a^* και γνωστό c'



γνωστά	άγνωστα
p′	α
p	α*
p*	k
c′	
α′=p′	

row	column number															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
S_1																
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																
[0]	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
[1]	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
[2]	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
[3]	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3																
[0]	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
[1]	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
[2]	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
[3]	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4																
[0]	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
[1]	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
[2]	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
[3]	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5																
[0]	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
[1]	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
[2]	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
[3]	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6																
[0]	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
[1]	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
[2]	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
[3]	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7																
[0]	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
[1]	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
[2]	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
[3]	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8																
[0]	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
[1]	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
[2]	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
[3]	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

ΠΑΡΑΔΕΙΓΜΑ

Ας θεωρήσουμε το 1ο S-box του DES

Υπενθύμιση:
το $\mathbf{B_i = b_1 b_2 \ldots b_6}$ απεικονίζεται στο $\mathbf{S_1(B_i)}$ και αντιστοιχεί στη γραμμή $\mathbf{r = 2*b_1 + b_6}$, και στη στήλη $\mathbf{b_2 b_3 b_4 b_5}$

- *Π.χ.* $S_1(110101)$ αντιστοιχεί στο στοιχείο της γραμμής 3 και της στήλης 10, δηλαδή $S_1(110101) = 3 = 0011$
- Τα bits που πάνε ως είσοδο στο S-box περνάνε πρώτα από XOR με κλειδί

Κρυπτανάλυση του DES

row	column number															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
S_1																
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																

→ **ΠΑΡΑΔΕΙΓΜΑ**

- Έστω η διαφορά εισόδου $110100 = 34_{16}$ και η διαφορά εξόδου $0100 = 4_{16}$.
- Η σχετική συχνότητα εμφάνισης για το ζευγάρι αυτό προκύπτει $N_D S_1(34_{16}, 4_{16}) = 2$
- Οι δύο τιμές για τις οποίες η διαφορά εισόδου 34_{16} μπορεί να προκαλέσει τη διαφορά εξόδου 4_{16}, είναι η 13_{16} και η 27_{16}
  - ✓ $(13_{16} \oplus 27_{16} = 34_{16})$,
- με αντίστοιχες εξόδους 6_{16} και 2_{16}
  - ✓ Δηλαδή $S_1(13_{16}) = 6_{16}$
  - ✓ και $S_1(27_{16}) = 2_{16}$

Πιθανές διαφορές εξόδου για διαφορά εισόδου 34_{16}

διαφορά εξόδου	πιθανές είσοδοι (σε δεκαεξαδική βάση) S_{11}
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

Κρυπτανάλυση του DES

row	column number															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
S_1																
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																

→ **ΠΑΡΑΔΕΙΓΜΑ**

- Έστω τώρα διαφορά εισόδου 34_{16}, και έστω ότι δημιουργήθηκε από τις εισόδους 1_{16} και 35_{16} ($34_{16} = 1_{16} \oplus 35_{16}$)
- Έστω ότι παράγεται διαφορά εξόδου D_{16}
- Στην είσοδο του S_1 (μετά την εφαρμογή του XOR με το κλειδί), θα υπάρχει μια από τις 8 τιμές της διαφοράς εξόδου D_{16}
- Αυτό θα μας δώσει 8 πιθανές τιμές για το κλειδί

διαφορά εξόδου	πιθανές είσοδοι (σε δεκαεξαδική βάση) S_1
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

Κρυπτανάλυση του DES

row	column number															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
S_1																
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																

→ **ΠΑΡΑΔΕΙΓΜΑ**

- Έτσι προκύπτει ένας πίνακας και μπορούμε να δοκιμάσουμε όλα τα πιθανά κλειδιά
- Αν η διαφορά εισόδου 34_{16} δημιουργήθηκε από άλλες εισόδους (π.χ. 33_{16} και 7_{16}) τότε θα προκύψει άλλος πίνακας
- Παίρνοντας την τομή μειώνουμε τον αριθμό των πιθανών κλειδιών

διαφορά εξόδου	πιθανές είσοδοι (σε δεκαεξαδική βάση) S_{1I}
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

πιθανές είσοδοι του S_1		Πιθανά κλειδιά	
06,	32	07,	33
10,	24	11,	25
16,	22	17,	23
1C,	28	1D,	29

Κρυπτανάλυση του DES

- Διαφορική κρυπτανάλυση
  - ✓ Έτσι μπορούμε εύκολα να «σπάσουμε» 1 γύρο
  - ✓ Για συστήματα με πολλαπλούς γύρους, χρειάζεται να βρούμε ένα μονοπάτι με διαφορικά
  - ✓ Οι πιθανότητες των διαφορικών κάθε κύκλου πολλαπλασιάζονται
  - ✓ Αν κάποιος διαδίδει διαφορές διαμέσου των κύκλων με ικανοποιητική πιθανότητα, τότε μπορεί να δημιουργήσει μια non-random διαφορά μεταξύ bits σε μια μάσκα κοντά στο input, και στα bits μιας μάσκας, κοντά στο output.
  - ✓ Αυτό επιτρέπει την εικασία των round key bits του 1ου και 16ου κύκλου
  - ✓ Έτσι υπολογίζονται round key bits των εξώτερων κύκλων και μετά των εσώτερων.
  - ✓ Χρειάζονται όμως πολλά ζεύγη από chosen plaintext-ciphertext
    - Για 4 γύρους, «σπάει» σε ≤ 0.3 sec
    - Για 6 γύρους, απαιτούνται 2^7 ζεύγη
    - Για 8 γύρους, απαιτούνται 2^{20} ζεύγη
    - Για 15 γύρους, απαιτούνται 2^{41} ζεύγη
    - Για 16 γύρους, απαιτούνται 2^{47} ζεύγη

- Ιδιότητες και Αδυναμίες DES
  - ✓ Ευρωστία DES

attack method	data complexity		storage complexity	processing complexity
	known	chosen		
exhaustive precomputation	—	1	2^{56}	1 (table lookup)
exhaustive search	1	—	negligible	2^{55}
linear cryptanalysis	2^{43} (85%)	—	for texts	2^{43}
	2^{38} (10%)	—	for texts	2^{50}
differential cryptanalysis	—	2^{47}	for texts	2^{47}
	2^{55}	—	for texts	2^{55}

DES

- Από το 1998 άρχισε να θεωρείται ευάλωτο σε κρυπταναλύσεις

3DES

- Το Triple DES, TDES, ή TDEA ή απλά 3DES προτάθηκε στις αρχές του 1980
- Προτυποποιήθηκε ως ANSI X9.17 το 1985
- 1999: NIST υιοθετεί 3DES ως FIPS 46-3
- Χρησιμοποιεί τρία κλειδιά και τρεις εκτελέσεις του DES
- 3 φορές πιο αργό αλλά αρκετές φορές πιο ασφαλές

3DES

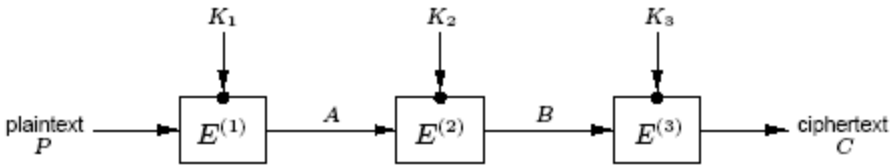
EDE (Encryption – Decryption – Encryption)

Οι 16 κύκλοι του DES γίνονται 48:

16 για Encryption

16 για Decryption

16 για Encryption



Encryption $C = E_{K3} [D_{K2} [E_{K1} [P]]]$

Decryption $P = D_{K1} [E_{K2} [D_{K3} [C]]]$

Είναι αποδεκτό K1 = K3 → two-key triple-encryption

Άλλοι Συμμετρικοί και Block Ciphers (Βασισμένοι σε παρόμοιες ιδέες)

Algorithm	Πότε	Μέγεθος κλειδιού	Αριθμός Κύκλων	Μετασχηματισμοί	Σχόλια
DES	1977	56	16	XOR, fixed S-boxes	Kerberos
3DES	1985	112 ή 168	48	XOR, fixed S-boxes	PGP, S/MIME
IDEA	1991	128	8	XOR, πρόσθεση, πολλαπλασιασμός	PGP
Blowfish	1993	Έως 448	16	XOR, dynamic S-boxes, δυαδική πρόσθεση,	Subkeys and S-boxes παράγονται με επαναλήψεις του Blowfish. Όχι καλό για εφαρμογές συχνής αλλαγής κλειδιού
RC5	1994	Έως 2048	Έως 255	Πρόσθεση, αφαίρεση, XOR, ολίσθηση	*RFC2040*, H/W και S/W, low memory requirements
CAST-128	1997	40 έως 128	16	Πρόσθεση, αφαίρεση, XOR, ολίσθηση, fixed S-boxes	*RFC2144*, Round Function (*F*) differs per round. PGP

Εξαντλητική Αναζήτηση

Μήκος Κλειδιού (bits)	Αριθμός Πιθανών Κλειδιών	Χρόνος για δοκιμές μίας αποκρυπτογράφησης (μs)	Χρόνος για δοκιμές αν οι λειτουργίες επιταχυνθούν κατά 10^6
32	$2^{32} = 4{,}3 \times 10^9$	2^{31} μs = 35,8 minutes	2,15 msecs
56 (DES)	$2^{56} = 7{,}2 \times 10^{16}$	2^{55} μs = 1142 χρόνια	10 ώρες
128	$2^{128} = 3{,}4 \times 10^{38}$	2^{127} μs = $5{,}4 \times 10^{24}$ χρόνια	$5{,}4 \times 10^{18}$ χρόνια
168 (3DES)	$2^{168} = 3{,}7 \times 10^{50}$	2^{167} μs = $3{,}9 \times 10^{36}$ χρόνια	$3{,}9 \times 10^{30}$ χρόνια

- ***Διαφαίνεται ότι τα 168 bits του 3DES αποτελούν εγγύηση έναντι εξαντλητικής αναζήτησης για τα επόμενα χρόνια***

Advanced Encryption Standard (AES)

NIST Initiative (1997)

Γιατί	DES/3DES αργοί σε υλοποιήσεις S/W Blocks των 64bit (ανάγκη για αύξηση αποδοτικότητας και ασφάλειας) Keys up to 256
NIST Specs	Symmetric block cipher Blocks των 128bit Keys: 128, 192 και 256 bits
Κριτήρια αξιολόγησης	Ασφάλεια Αλγορίθμων (τουλάχιστον ίση με 3DES, αλλά απλούστερος) Κόστος (μνήμη, processing power) Απλότητα υλοποίησης (S/W, H/W)

History of AES

- ✓ September 12, 1997: Προκύρηξη για υποβολή προτάσεων για ένα νέο encryption standard
- ✓ June 15, 1998: Λήξη προθεσμίας υποβολής
  - 21 προτεινόμενα κρυπτοσυστήματα
  - 15 πληρούσαν όλες τις προϋποθέσεις
- ✓ August, 1998: 1st AES candidate conference
- ✓ March 1999: 2nd AES candidate conference
- ✓ August 1999: Ανακοινώνονται πέντε finalists MARS, RC6, Rijndael, Serpent, Twofish
- ✓ April, 2000: 3rd AES candidate conference
- ✓ October 2000: Επιλέγεται το Rijndael (από τους Βέλγους Daemen και Rijmen)
  - Δεν ακολουθεί δομή Feistel. Χρησιμοποιεί S-Boxes
- ✓ February 2001: Βγαίνει σε δημόσια διαβούλευση
- ✓ November 2001: Προτυποποιείται ως FIPS 197

History of AES

- ✓ Σε αντίθεση με το DES, η διαδικασία επιλογής χαρακτηρίστηκε από πολύ περισσότερη διαφάνεια και «international flavor»
  - 3 συνέδρια
  - Επίσημη πρόσκληση για public comments, feedback, etc
- ✓ Χώρες που εκπροσωπήθηκαν από τα 15 υποψήφια συστήματα:
  - Australia, Belgium, Canada, Costa Rica, France, Germany, Israel, Japan, South Korea, Norway, UK, USA
- ✓ Στο τέλος και τα 5 finalists θεωρήθηκαν ότι είναι ασφαλή κρυπτοσυστήματα. Το Rijndael επιλέχθηκε ως το σύστημα που έδινε τον καλύτερο συνδυασμό από security, performance, implementability, και flexibility

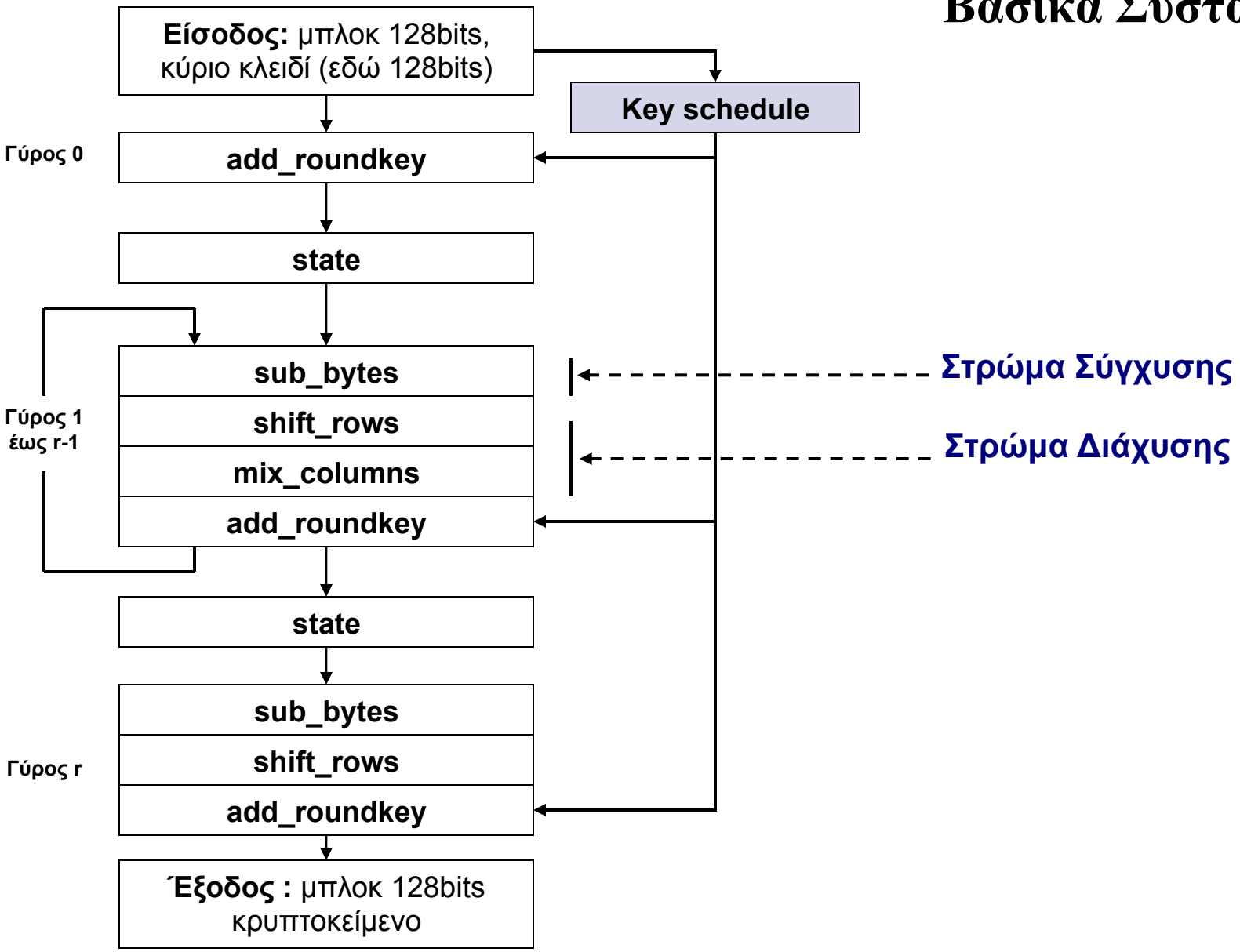
Χαρακτηριστικά

- ✓ Block cipher
  - Μέγεθος plaintext/ciphertext: $P = C = \{0,1\}^{128}$
- ✓ Κλειδιά : μεταβλητό μέγεθος, 128, 192 ή 256 bits
  - Μεγάλος χώρος κλειδιών : αδύνατη η εξαντλητική αναζήτηση (για τα επόμενα χρόνια)
- ✓ Δεν ακολουθεί δίκτυο Feistel για σύγχυση και διάχυση
  - Ένα δίκτυο Feistel κρυπτογραφεί σε κάθε γύρο τα μισά bits του μπλοκ εισόδου
- ✓ Στον AES κρυπτογραφείται όλο το μπλοκ εισόδου σε κάθε γύρο
  - Λιγότεροι γύροι για ισοδύναμη ευρωστία με δίκτυα Feistel
- ✓ Αριθμός γύρων εξαρτάται από μέγεθος κλειδιού
  - $r_{128}=10$, $r_{192}=12$, $r_{256}=14$

Χαρακτηριστικά

- ✓ Υπάρχει ένας αποθηκευτικός χώρος ενδιάμεσων αποτελεσμάτων που καλείται ***state*** (κατάσταση)
- ✓ Κάθε κύκλος αποτελείται από τρία στρώματα:
- ✓ Σύγχυσης
  - Με τη διαδικασία sub_bytes (substitute) μέσω μη γραμμικών S-boxes
- ✓ Διάχυσης
  - Διαδικασίες shift_rows (ή rotate_rows) και mix_columns
- ✓ Κρυπτογράφησης
  - Διαδικασία XOR με κλειδί γύρου (add_roundkey ή xor_roundkey)
  - Όπως και στο DES υπάρχει key schedule από ένα master key

Βασικά Συστατικά



- Σε κώδικα…

```
#define LENGTH 16                          /* # bytes in data block or key */
#define NROWS 4                            /* number of rows in state */
#define NCOLS 4                            /* number of columns in state */
#define ROUNDS 10                          /* number of iterations */
typedef unsigned char byte;                /* unsigned 8-bit integer */

rijndael(byte plaintext[LENGTH], byte ciphertext[LENGTH], byte key[LENGTH])
{
  int r;                                   /* loop index */
  byte state[NROWS][NCOLS];                /* current state */
  struct {byte k[NROWS][NCOLS];} rk[ROUNDS + 1];     /* round keys */

  expand_key(key, rk);                     /* construct the round keys */
  copy_plaintext_to_state(state, plaintext);   /* init current state */
  xor_roundkey_into_state(state, rk[0]);   /* XOR key into state */

  for (r = 1; r <= ROUNDS; r++) {
      substitute(state);                   /* apply S-box to each byte */
      rotate_rows(state);                  /* rotate row i by i bytes */
      if (r < ROUNDS) mix_columns(state);  /* mix function */
      xor_roundkey_into_state(state, rk[r]);   /* XOR key into state */
  }
  copy_state_to_ciphertext(ciphertext, state);   /* return result */
}
```

Σύγχυση

Διάχυση

Κρυπτο.

- State
  - ✓ Μια δομή των 128 bits, πάνω στην οποία γίνονται όλοι οι μετασχηματισμοί
  - ✓ Δομημένη σε 4x4=16 bytes
  - ✓ Κάθε byte της μορφής $(\alpha_7\alpha_6\alpha_5\alpha_4\alpha_3\alpha_2\alpha_1\alpha_0)$ αντιστοιχεί στο πολυώνυμο $\Sigma_{0 \text{ έως } 7}\alpha_i x^i$ του $GF(2^8)$
  - ✓ Αρχικά:

s_{00}	s_{01}	s_{02}	s_{03}
s_{10}	s_{11}	s_{12}	s_{13}
s_{20}	s_{21}	s_{22}	s_{23}
s_{30}	s_{31}	s_{32}	s_{33}

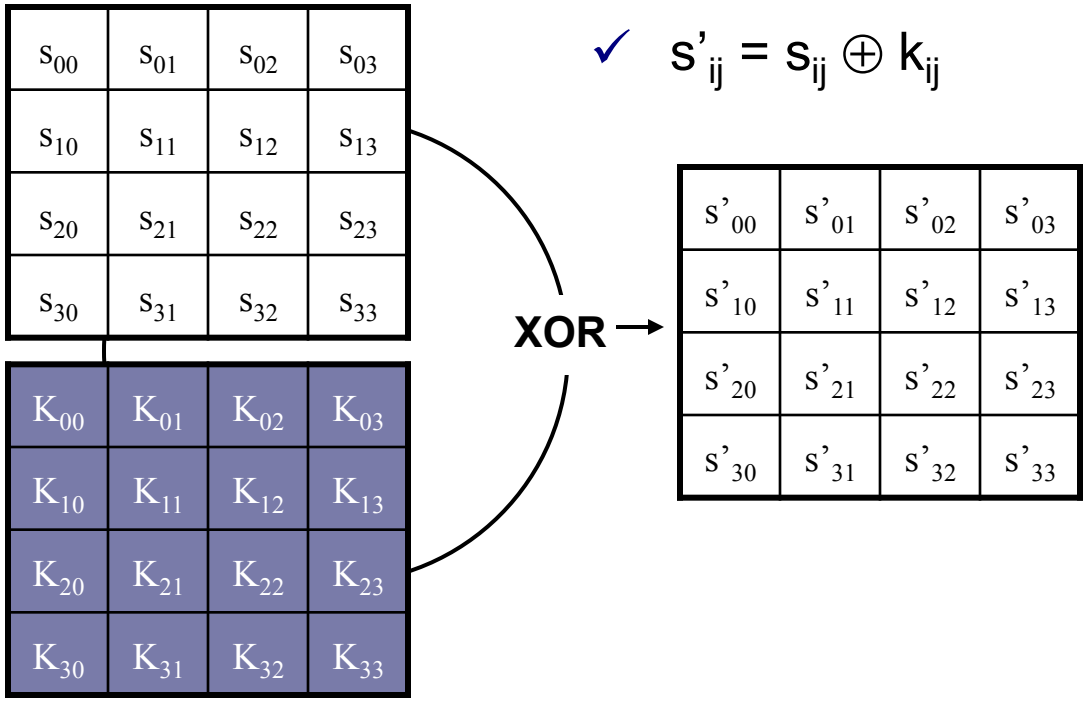
:=

x_0	x_4	x_8	x_{12}
x_1	x_5	x_9	x_{13}
x_2	x_6	x_{10}	x_{14}
x_3	x_7	x_{11}	x_{15}

(= plaintext)

- add_roundkey: XOR με το κλειδί κάθε γύρου
  - ✓ Η μόνη διαδικασία στο γύρο r=0
  - ✓ Η τελευταία διαδικασία (στρώμα) κάθε κύκλου r>1

- add_roundkey
  - ✓ Κάθε subkey παράγεται από μια διαδικασία key_schedule
  - ✓ Κάθε subkey είναι του ιδίου μεγέθους με state
  - ✓ Κλειδί: Δομημένο σε 4x4=16 bytes k_{ij}



✓ $s'_{ij} = s_{ij} \oplus k_{ij}$

- Στρώμα Σύγχυσης – sub_bytes
  - ✓ Σε κάθε γύρο είναι η πρώτη διαδικασία που πραγματοποιείται
  - ✓ Κάθε byte του state αντικαθίσταται από ένα άλλο byte μέσω ενός S-box S: $\{0, 1\}^8 \rightarrow \{0, 1\}^8$
  - ✓ Ίδιο S-box και για τα 16 bytes
  - ✓ Σε αντίθεση με το DES, η συνάρτηση του S-box ορίζεται αλγεβρικά

Recall:

- ✓ Το πεδίο $GF(2^8)$ παράγεται από ανάγωγο πολυώνυμο βαθμού 8
  - Για το AES χρησιμοποιείται το ανάγωγο πολυώνυμο $m(x)=x^8+x^4+x^3+x+1$
- ✓ $GF(2^8) = Z_2[x]/(m(x))$ = πολυώνυμα μέγιστου βαθμού 7, με συντελεστές 0 ή 1
- ✓ Υπάρχουν ακριβώς $2^8 = 256$ τέτοια πολυώνυμα
- ✓ Κάθε byte από το state θα το βλέπουμε ως ένα πολυώνυμο

- Στρώμα Σύγχυσης – sub_bytes
  - ✓ Κατά τη διαδικασία sub_bytes εκτελούνται τα ακόλουθα βήματα για κάθε byte s_{ij} του state:
    - Βήμα 1 Για το byte s_{ij}, έστω $s_{ij}(x)$ το αντίστοιχο πολυώνυμο. Υπολογίζεται ο πολλαπλασιαστικός αντίστροφος του $s_{ij}(x)$, δηλαδή το πολυώνυμο r(x) τέτοιο ώστε $s_{ij}(x)r(x) = 1 \pmod{m(x)}$
      - Εκτεταμένος αλγόριθμος Ευκλείδη
    - Βήμα 2. Έστω $(x_0,x_1,x_2,\ldots,x_7)$ ο αντίστροφος του $s_{ij}(x)$. Πραγματοποιείται ο ακόλουθος μετασχηματισμός:

$$(b_0,\ldots, b_7) = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

    - Το S-box αντικαθιστά το s_{ij} με το $(b_0,\ldots, b_7)$

- Στρώμα Σύγχυσης – sub_bytes
  - ✓ Παράδειγμα: έστω ότι $s_{00} = 53_{16} = 01010011$
  - ✓ Τότε $s_{00}(x) = x^6 + x^4 + x + 1$
  - ✓ Ο πολλαπλασιαστικός αντίστροφος στο $GF(2^8)$, με χρήση του αλγορίθμου του Ευκλείδη, είναι $x^7 + x^6 + x^3 + x = (11001010) = (a_7a_6a_5a_4a_3a_2a_1a_0)$
  - ✓ Έστω $c_7c_6c_5c_4c_3c_2c_1c_0$=01100011 (σταθερός πίνακας του μετασχηματισμού)
  - ✓ Ένα καλό κόλπο για το γραμμικό μετασχηματισμό:

    for i=0 to 7

    $b_i \leftarrow (a_i+a_{i+4}+a_{i+5}+a_{i+6}+a_{i+7}+ c_i)$ mod2 /* δείκτες ανάγονται mod8

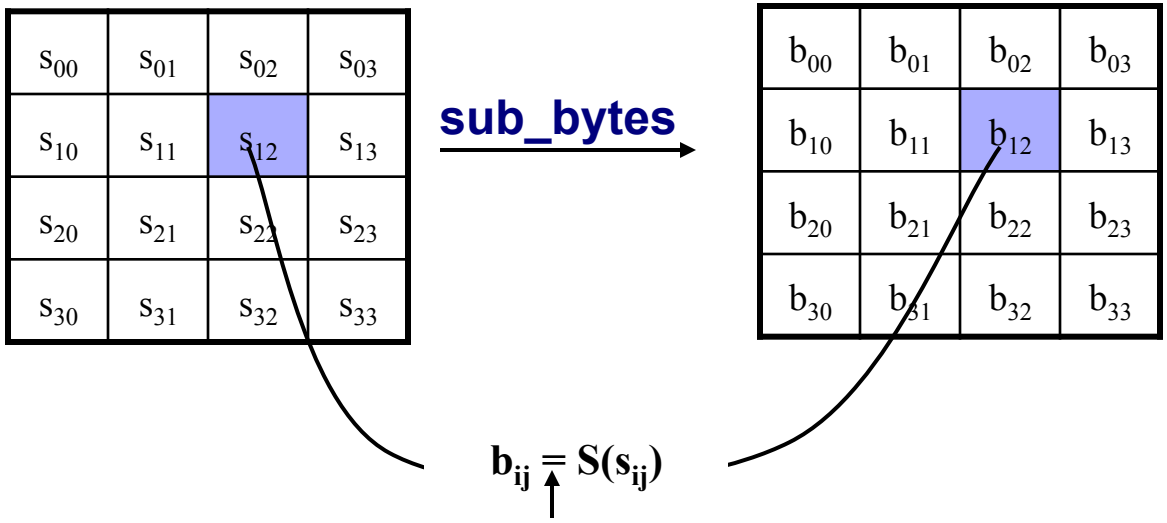
    Return $(b_7b_6b_5b_4b_3b_2b_1b_0)$

- Στρώμα Σύγχυσης – sub_bytes
  - ✓ Επομένως
    - $b_0 = (a_0+a_4+a_5+a_6+a_7+c_0) \bmod 2 = (0 + 0 + 0 +1 +1 +1) \bmod 2 = 1$
    - $b_1 = (a_1+a_5+a_6+a_7+a_0+c_1) \bmod 2 = (1 + 0 + 1 +1 +0 +1) \bmod 2 = 0$
    - …
  - ✓ Τελικά παίρνουμε $b = (11101101) = \{ED\}_{16}$

Παρατηρήσεις:

- ✓ Μπορούμε να αναπαραστήσουμε κάθε byte με 2 δεκαεξαδικά νούμερα
- ✓ Χρήση ενός πίνακα για την αποθήκευση της συνάρτησης του S-box
- ✓ Π.χ. για το byte 1c, η τιμή της S είναι στην γραμμή 1 και στήλη c
- ✓ Συνολικά η sub_bytes διαθέτει καλή μη-γραμμική συμπεριφορά

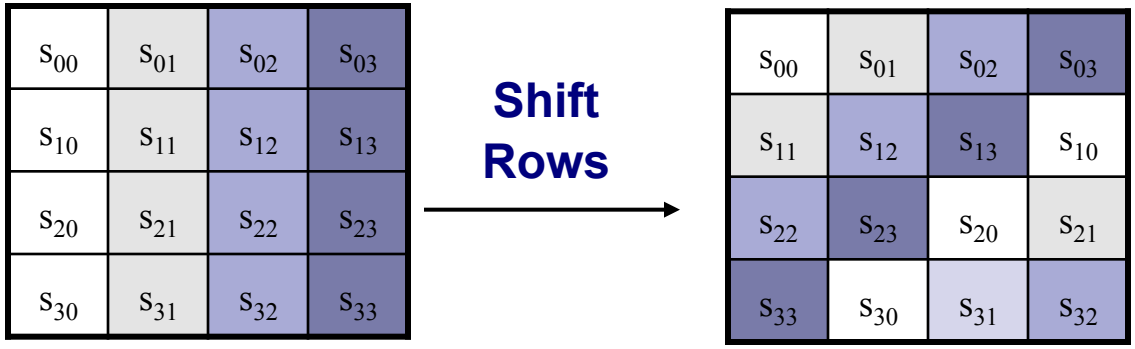
- Στρώμα Σύγχυσης – sub_bytes



	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Look-up table για την αντικατάσταση των bytes

- Στρώμα διάχυσης – 1η διαδικασία: shift_rows
  - ✓ Η shift_rows εφαρμόζεται στις γραμμές του state και τις ολισθαίνει κατά 0, 1, 2 και 3 θέσεις αριστερά αντίστοιχα:
    - Γραμμή 0: καμία ολίσθηση
    - Γραμμή 1: αριστερή ολίσθηση κατά 1 θέση
    - Γραμμή 2: αριστερή ολίσθηση κατά 2 θέσεις
    - Γραμμή 2: αριστερή ολίσθηση κατά 3 θέσεις



Στρώμα διάχυσης – 2η διαδικασία: mix_columns

- Θεωρούμε κάθε στήλη του state ως ένα πολυώνυμο βαθμού 3 με συντελεστές πολυώνυμα του GF(2^8)
  - Αν (S_i, S_j, S_k, S_m) μία στήλη του state, μπορούμε να τη δούμε ως το πολυώνυμο $a_3x^3 + a_2x^2 + a_1x + a_0$
  - Όπου a_3 το πολυώνυμο που αντιστοιχεί στο S_i, a_2 το πολυώνυμο που αντιστοιχεί στο S_j, κ.ο.κ.
- Κάθε στήλη πολλαπλασιάζεται με ένα προκαθορισμένο πολυώνυμο $c(x) = 3x^3+x^2+x+2$ και το αποτέλεσμα ανάγεται mod (x^4+1). Έστω $r(x)$ το πολυώνυμο που προκύπτει
- Τελικά η στήλη αντικαθίσταται από τα bytes που αντιστοιχούν στους συντελεστές του $r(x)$

- Στρώμα διάχυσης – 2η διαδικασία: mix_columns
  - ✓ Τελικά, η διαδικασία μπορεί να κωδικοποιηθεί ως:
    - $r_i = 2S_i + 3S_j + S_k + S_m$
    - $r_j = S_i + 2S_j + 3S_k + S_m$
    - $r_k = S_i + S_j + 2S_k + 3S_m$
    - $r_m = 3S_i + S_j + S_k + 2S_m$

$$\begin{bmatrix} r_i \\ r_i \\ r_k \\ r_m \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} S_i \\ S_j \\ S_k \\ S_m \end{bmatrix}$$

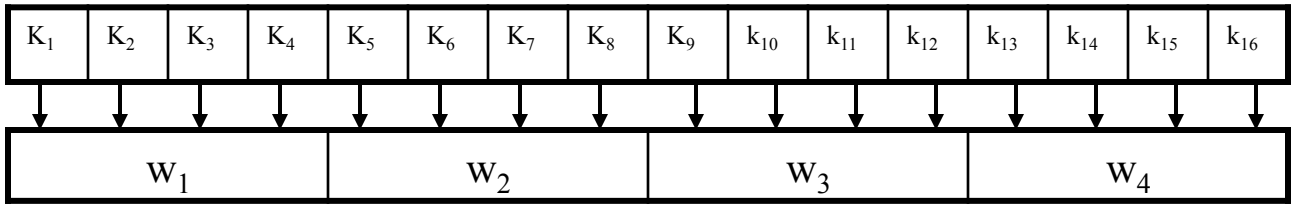
- Key Schedule
  - ✓ Είσοδος: Κλειδί 128bits (master key)
  - ✓ Έξοδος: Για 10 γύρους χρειαζόμαστε 11 subkeys των 128 bits
    - 4 λέξεις (words) για κάθε κλειδί (1 word = 4 bytes)
    - Συνολικά πρέπει να παράγουμε 44 λέξεις (44x32=1408 bits) σε ένα πίνακα $w_0,\ldots, w_{43}$
  - ✓ Οι λέξεις w χρησιμοποιούνται ανά τέσσερις σε κάθε διαδικασία add_roundkey
  - ✓ Αρχικά οι τέσσερις πρώτες λέξεις w_0 έως w_3 φορτώνονται με τα 128 bits του master key
  - ✓ Στη συνέχεια ακολουθεί διαδικασία που επαναλαμβάνεται 40 φορές για να καθοριστούν οι λέξεις w_4 έως w_{43}

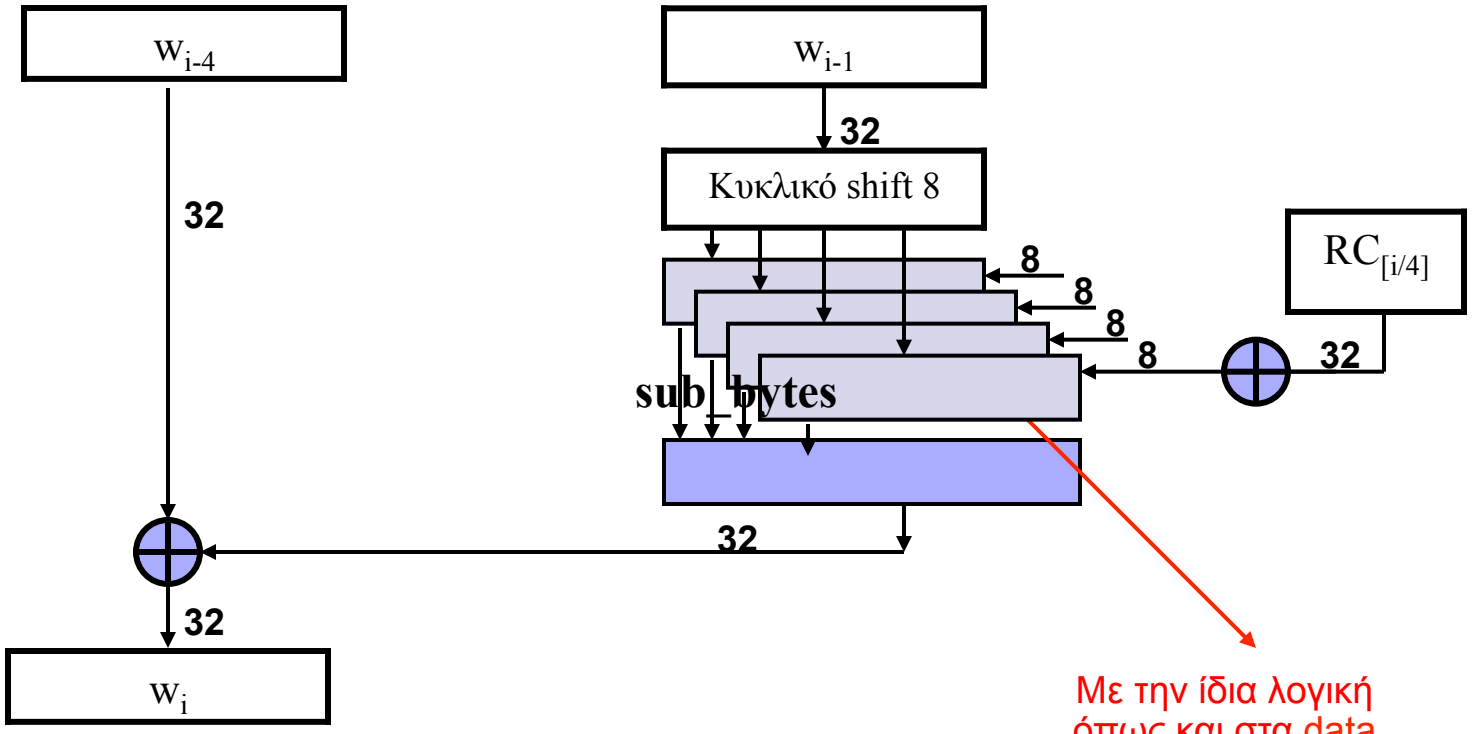
- Key Schedule

```
Key expansion(key)
    (w0,…, w3) = key
    for i=4 to 43 {
        temp = wi-1
        if (i≡0 mod 4)
            then temp = sub_word(rot_word(temp))⊕ RCi/4
        wi = wi-4 ⊕ temp
    }
```

- ✓ rot_word: αριστερή ολίσθηση κατά 1 byte
- ✓ sub_word: εφαρμογή του S-box σε κάθε byte της λέξης
- ✓ Όταν το i είναι πολ/σιο του 4:
  - Εφαρμόζεται κυκλική ολίσθηση προς τα αριστερά και αντικατάσταση πριν το XOR
  - Πίνακας RC: πίνακας με 10 λέξεις που έχουν fixed τιμή από την αρχή

- Key Schedule





RC_1=01000000
RC_2=02000000
RC_3=04000000
RC_4=08000000
RC_5=10000000
RC_6=20000000
RC_7=40000000
RC_8=80000000
RC_9=1B000000
RC_{10}=36000000

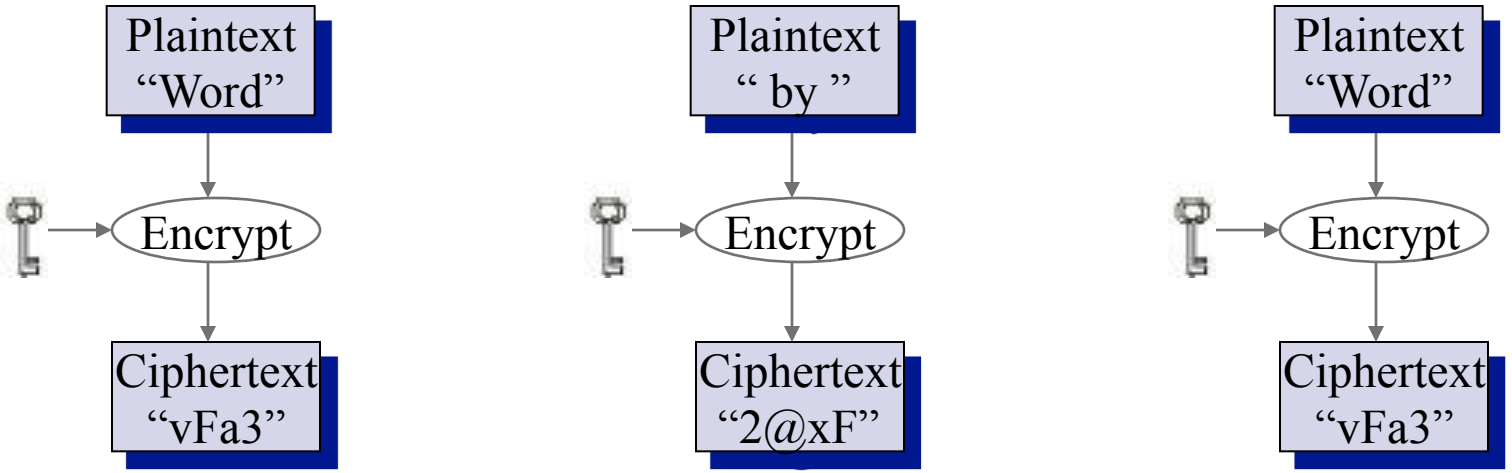
- Decryption
  - ✓ Ο Bob μπορεί να υπολογίσει το key schedule με τον ίδιο τρόπο
  - ✓ Όλες οι λειτουργίες αντιστρέφονται εύκολα
  - ✓ Π.χ. Για το S-box, χρησιμοποιείται η Inv_sub_bytes, που υλοποιείται με look-up table
  - ✓ Παρόμοια μπορούν να υλοποιηθούν και όλοι οι άλλοι μετασχηματισμοί

- Κρυπτανάλυση
  - ✓ Δεν υπάρχουν weak ή semi-weak κλειδιά στο AES
  - ✓ Δεν υπάρχει κανένας περιορισμός ως προς την επιλογή του κλειδιού
  - ✓ Αρκετά ασφαλές σύστημα σε γραμμική και διαφορική κρυπτανάλυση
  - ✓ Η αλγεβρική κατασκευή των S-boxes εξασφαλίζει ότι
    - η κατανομή των διαφορών εξόδου είναι ομοιόμορφη
    - Δεν υπάρχουν γραμμικές σχέσεις με αυξημένη πόλωση
  - ✓ Αυτή τη στιγμή δεν υπάρχει κάποια γνωστή επιτυχημένη επίθεση στο AES
  - ✓ Υπάρχουν κάποιες επιθέσεις αλλά για μικρότερο αριθμό γύρων
    - Δεν είναι αποτελεσματικές για 10 γύρους
  - ✓ Αρχικά υπήρξε κριτική για την αλγεβρική κατασκευή
  - ✓ Πιθανότητα εύρεσης αλγεβρικής επίθεσης

- Μέθοδοι Λειτουργίας
  - ✓ Υπάρχουν διάφορες προτάσεις ως προς τον μετασχηματισμό του επόμενου block (τμήματος) και την ανάδραση προηγούμενων μετασχηματισμών στο τρέχον τμήμα.
  - ✓ Τέσσερις μέθοδοι λειτουργίας για block ciphers:
    - ✓ **ECB electronic-codebook**
    - ✓ **CBC cipher-block chaining**
    - ✓ **CFB cipher feedback**
    - ✓ **OFB output feedback**
  - ✓ Έχουν προταθεί και μερικές καινούριες μέθοδοι για το AES

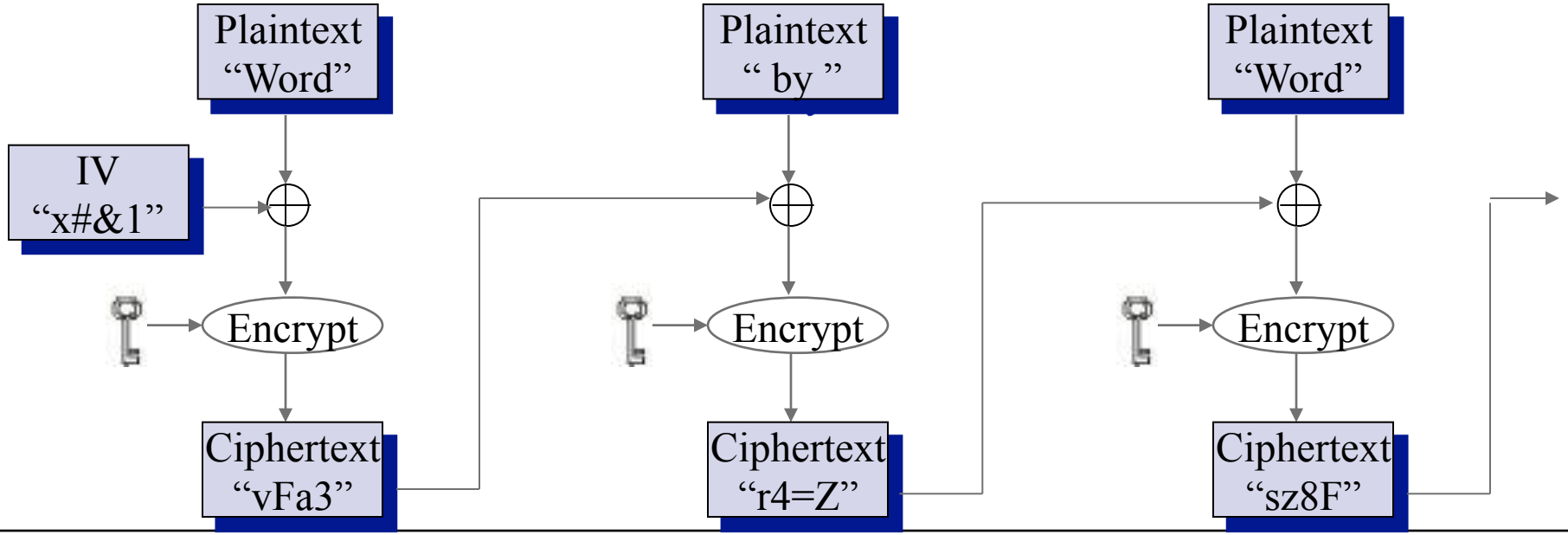
ECB – Electronic Codebook

- Κάθε block στο plaintext κρυπτογραφείται με το ίδιο κλειδί
- ***Chaining dependencies:*** Τα plaintext blocks κρυπτογραφούνται ανεξάρτητα
- ***Error propagation:*** ένα η περισσότερα bit errors σε ένα ciphertext block επηρεάζουν το decipherment μόνο αυτού του block
- Απλός και γρήγορος



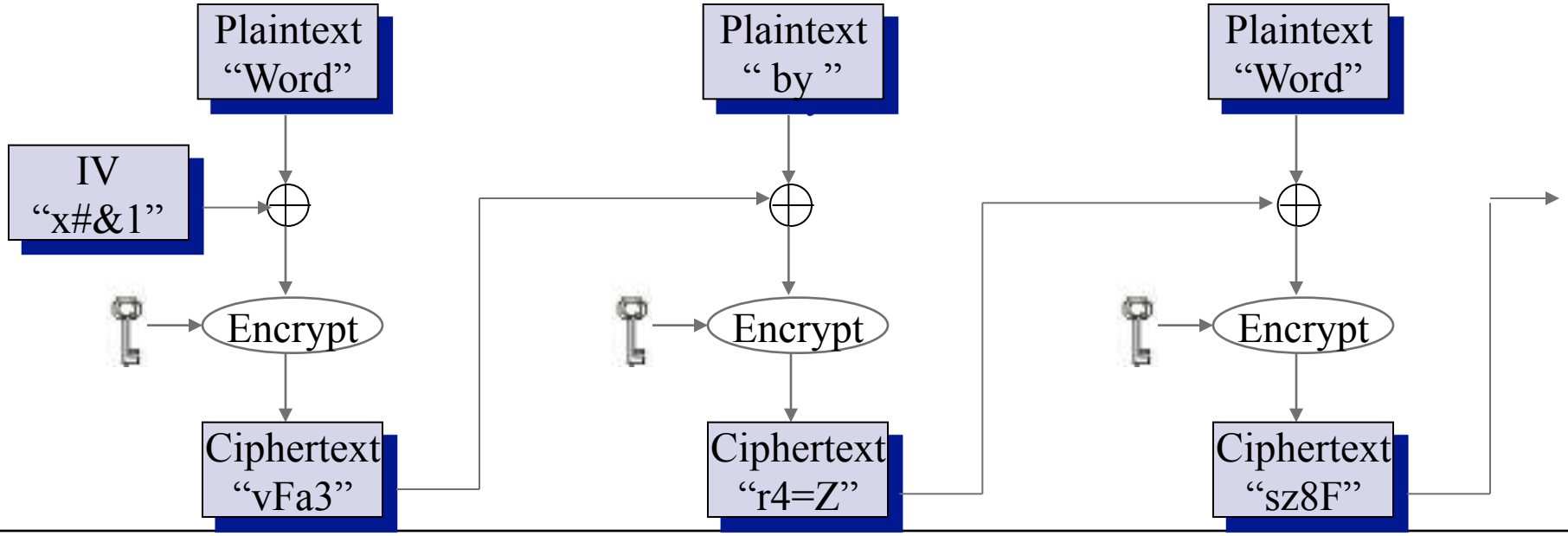
CBC – Cipher Block Chaining

- Κάθε block κρυπτογραφείται αφού πρώτα περάσει από XOR με το ciphertext του προηγούμενου block
- Απαιτείται Initialization Vector (IV) γνωστό σε πομπό και δέκτη
- Έστω $c_0 = IV$
- Για το block i: $c_i = e_k(c_{i-1} \oplus x_i)$
- Decryption: $x_i = d_k(c_i) \oplus c_{i-1}$
- Ίδια plaintext blocks **δεν** παράγουν ίδιο ciphertext block (μόνο για ίδιο IV)



CBC – Cipher Block Chaining

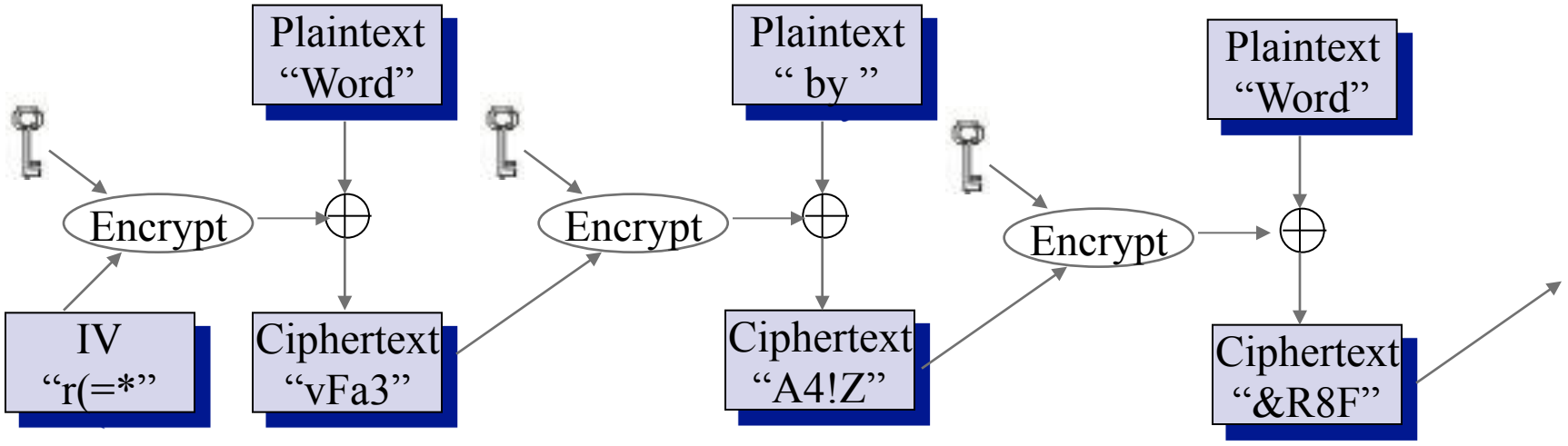
- ***Chaining dependencies:*** Το ciphertext c_j εξαρτάται από το plaintext μπλοκ x_{j-1} και *όλα* τα προηγούμενα. Προσοχή στο rearranging των ciphertext blocks.
- ***Error propagation:*** ένα bit error στο ciphertext block c_j επηρεάζει το decipherment στα blocks c_j και c_{j+1}
- **Error recovery:** Είναι self-synchronizing. Αν ένα error παρουσιαστεί στο block c_j αλλά όχι στα c_{j+1} και c_{j+2}, το x_{j+2} αποκρυπτογραφείται σωστά



CFB – Cipher Feedback

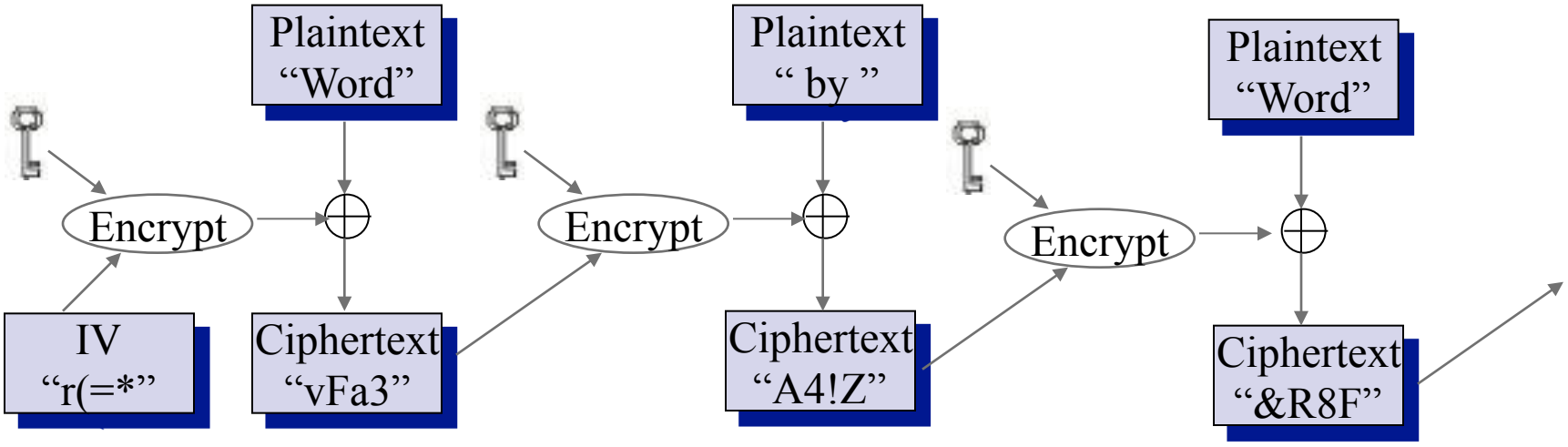
- Το ciphertext του προηγούμενου block κρυπτογραφείται και γίνεται XOR με το τρέχον plaintext block
  - Παραλλαγή r-bit CFB: Συνήθως όλα τα c_i μπαίνουν με τη σειρά σε ένα καταχωρητή ολίσθησης κατά r bits, και μετά την κρυπτογράφηση επιλέγονται τα πρώτα r bits και γίνονται XOR με r-bits του plaintext (r < n)
  - Απαιτείται Initialization Vector (IV, n bits)
  - Για το block i: $c_i = e_k(c_{i-1}) \oplus x_i$
  - Decryption: $x_i = e_k(c_{i-1}) \oplus c_i$

- Ίδια plaintext blocks **δεν** παράγουν ίδιο ciphertext block (μόνο για ίδιο IV)



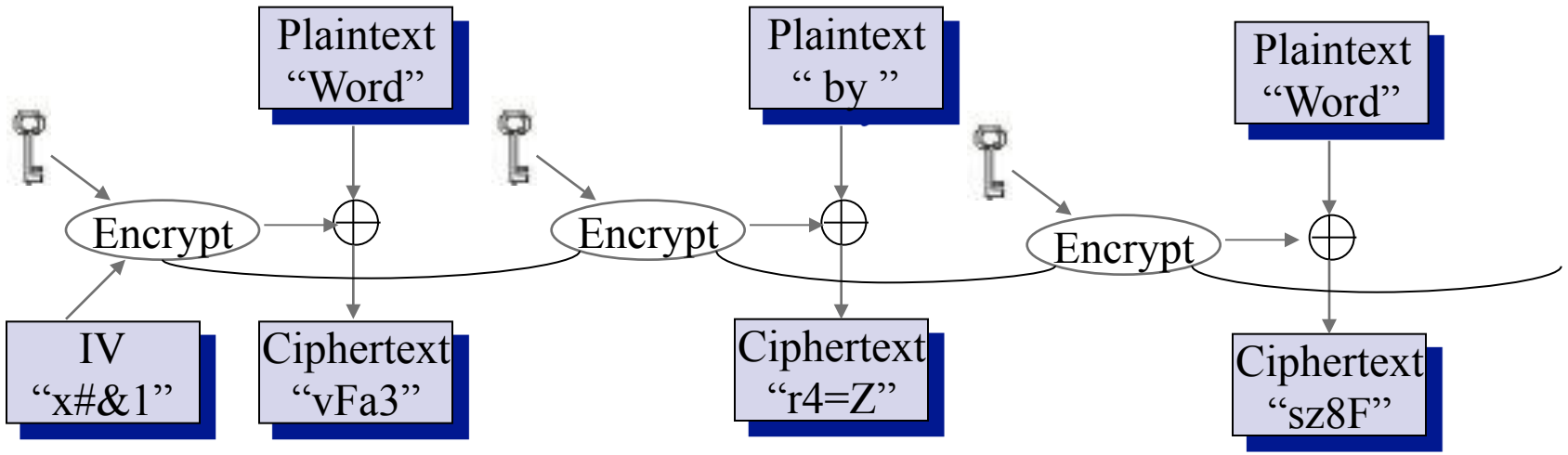
CFB – Cipher Feedback

- ***Chaining dependencies:*** Όπως και στο CBC
- ***Error propagation:*** ένα η περισσότερα bit errors σε ένα ciphertext block c_j επηρεάζουν το decipherment αυτού και των υπόλοιπων |n/r| ciphertext blocks (στην παραλλαγή r-bit CFB)
- **Error recovery:** η CFB είναι self-synchronizing
  - απαιτεί |n/r| ciphertext blocks για ανάκαμψη



OFB – Output Feedback (ISO 10116)

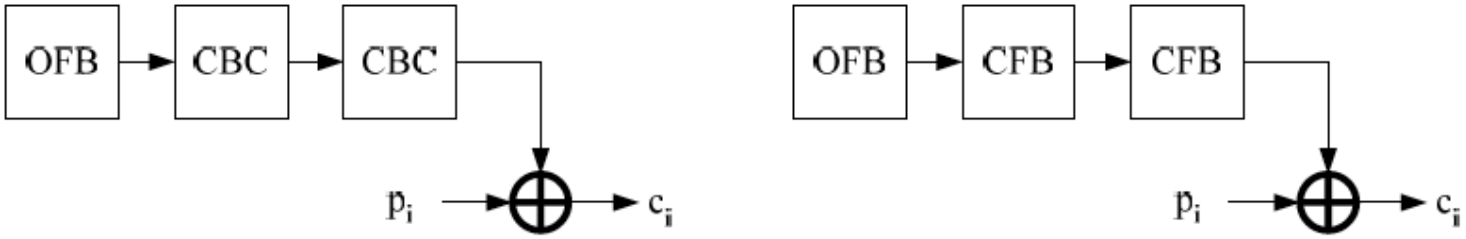
- Το keystream του προηγούμενου block επανα-κρυπτογραφείται παράγοντας ένα νέο keystream. Το νέο keystream συνδυάζεται με XOR με το plaintext του τρέχοντος block
  - Συνήθως με ένα μέρος αυτού, δηλαδή ένα r-bit plaintext < n
  - Απαιτείται Initialization Vector (IV)
- Ίδια plaintext blocks **δεν** παράγουν ίδιο ciphertext block (μόνο για ίδιο IV)
- ***Chaining dependencies:*** το keystream είναι ανεξάρτητο από plaintext
- **Error propagation:** ένα ή περισσότερα bit errors σε οποιοδήποτε χαρακτήρα ciphertext c_j επηρεάζει την αποκρυπτογράφηση μόνο αυτού του χαρακτήρα
- **Error recovery**: Ανακάμπτει καλύτερα από όποιον άλλον σε ciphertext bit errors



Μη τυποποιημένοι τρόποι λειτουργίας

- Έχουν προταθεί αρκετές άλλες μέθοδοι
- Ελάχιστες έχουν μελετηθεί σε βάθος
  - Δεν συνίστανται σε περιπτώσεις που δεν θέλουμε να ρισκάρουμε

- Πιθανές ασφαλείς κατασκευές: συνδυασμοί των τυποποιημένων μεθόδων



- Τοποθέτηση της ανάδρασης
  - Σε product cipher, είναι επιθυμητό να μην υπάρχει ανάδραση ενδιάμεσα
  - Αλλιώς δημιουργούνται shortcuts
  - Ο Oscar μπορεί να επιτεθεί χωριστά σε καθένα από τους αλγορίθμους του γινομένου