ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Οικονομικό Πανεπιστήμιο Αθηνών
Τμήμα Πληροφορικής
ΠΜΣ στα Πληροφοριακά Συστήματα

Κρυπτογραφία και Εφαρμογές

Διαλέξεις Ακ. Έτους 2015-2016

Μαρκάκης Ευάγγελος
markakis@aueb.gr

Ντούσκας Θεόδωρος
tntouskas@aueb.gr

Ας γνωριστούμε …

- Παρακαλώ παρουσιάστε ο καθένας σας τον διπλανό του αναφέροντας:
  - ✓ Ονοματεπώνυμο
  - ✓ Σπουδές
  - ✓ Επαγγελματική εμπειρία
  - ✓ Γνώσεις Ασφάλειας ΠΣ
  - ✓ Γνώσεις Κρυπτογραφίας
  - ✓ Τεχνικές γνώσεις (γλώσσες προγραμματισμού)
  - ✓ Πιστοποιήσεις

Θεωρία Αριθμών και Θεωρία Ομάδων: Μέρος 2

Μαθηματικό Υπόβαθρο για το AES

Επανάληψη

- Μια ομάδα (group) $(S, \oplus)$ είναι ένα σύνολο S μαζί με ένα τελεστή $\oplus: S \times S \rightarrow S$, έτσι ώστε να ισχύουν:
  - ✓ Κλειστότητα: Για κάθε $a, b \in S$, $(a \oplus b) \in S$
  - ✓ Προσεταιριστική ιδιότητα: Για κάθε $a, b, c \in S$, $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - ✓ Ύπαρξη μοναδιαίου (ή ουδέτερου) στοιχείου (Identity): υπάρχει στοιχείο $e \in S$, έτσι ώστε $e \oplus a = a \oplus e = a$ για κάθε $a \in S$. Το συμβολίζουμε και με $e_{\oplus}$
  - ✓ Ύπαρξη αντιστρόφου (Inverse): για κάθε $a \in S$, υπάρχει στοιχείο $a^{-1} \in S$, που καλείται αντίστροφος του a, τέτοιο ώστε $a \oplus a^{-1} = a^{-1} \oplus a = e$
- Αν σε μια ομάδα $(S, \oplus)$ ικανοποιείται και η αντιμεταθετική ιδιότητα ($a \oplus b = b \oplus a$ για όλα τα $a, b \in S$), τότε την αποκαλούμε αβελιανή (abelian) ομάδα

Δακτύλιος

- Μια δομή $(S, \oplus, \otimes)$ ονομάζεται δακτύλιος (ring) αν
  - ✓ Η $(S, \oplus)$ είναι αβελιανή ομάδα
  - ✓ Η $\otimes$ προσεταιριστική: Για κάθε a, b, c ϵ S, $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
  - ✓ Η $\otimes$ είναι επιμεριστική ως προς $\oplus$: $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$ και $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$, για κάθε a, b, c ϵ S
  - ✓ Αν η $\otimes$ ικανοποιεί την αντιμεταθετική ιδιότητα τότε η δομή αναφέρεται ως αντιμεταθετικός δακτύλιος
  - ✓ Αν η $\otimes$ έχει μοναδιαίο στοιχείο αναφέρεται ως δακτύλιος με μοναδιαίο στοιχείο. Τα μοναδιαία στοιχεία θα τα συμβολίζουμε με $e_{\oplus}$ και $e_{\otimes}$ για τις $\oplus$ και $\otimes$ αντίστοιχα.

Παράδειγμα 1: Ο Δακτύλιος $(Z_8, +_8, *_8)$

Είναι αντιμεταθετικός δακτύλιος

$+_8$	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

$*_8$	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

a	0	1	2	3	4	5	6	7
-a	0	7	6	5	4	3	2	1
a^{-1}	-	1	-	3	-	5	-	7

Παράδειγμα 2: Ο Δακτύλιος πινάκων nxn (Π_{nxn}, +, *)

- Το (Π_{nxn}, +) είναι αβελιανή ομάδα
- Δεν ισχύει η αντιμεταθετική ιδιότητα ως προς πολλαπλασιασμό πινάκων
- Δεν έχουν αντίστροφο όλοι οι nxn πίνακες (μόνο αυτοί που έχουν μη μηδενική ορίζουσα).

- Μία δομή $(S, \oplus, \otimes)$ λέγεται σώμα ή πεδίο (field) αν
  - ✓ Η $(S, \oplus)$ είναι αβελιανή ομάδα
  - ✓ Η $(S- e_{\oplus}, \otimes)$ είναι αβελιανή ομάδα
  - ✓ Ισχύει η επιμεριστική ιδιότητα της $\otimes$ ως προς $\oplus$
  - ✓ Δηλαδή είναι ένας δακτύλιος όπου η $\otimes$ είναι αντιμεταθετική και έχει αντίστροφο και μοναδιαίο

- Ιδιότητες πεδίου:
  - ✓ Για κάθε a, b στο S οι εξισώσεις
    - $a \oplus x = b$ και $a \otimes x = b$ $(a \neq 0)$ έχουν μοναδική λύση στο S
  - ✓ Για κάθε a, b, c στο S ισχύουν οι κανόνες απλοποίησης και διαγραφής
    - $a \oplus c = b \oplus c \rightarrow a = b$
    - $a \otimes c = b \otimes c \rightarrow a = b$ if $c \neq e_{\oplus}$
  - ✓ Για κάθε a, b στο S ισχύει η συνεπαγωγή
    - $a \otimes b = e_{\oplus} \rightarrow (a = e_{\oplus}$ ή $b = e_{\oplus})$

- Παραδείγματα: Οι κάτωθι δομές είναι πεδία
  - ✓ (Q, +, *) των ρητών
  - ✓ (R, +, *) των πραγματικών
  - ✓ (C, +, *) των μιγαδικών
  - ✓ ({0,1}, +2, *2) των δυαδικών (είναι και πεπερασμένο)

+	0	1
0	0	1
1	1	0

*	0	1
0	0	0
1	0	1

Ερώτηση: Υπάρχουν άλλα πεδία με πεπερασμένο αριθμό στοιχείων?

Πεδία Galois

- Evariste Galois (1811-1832)
- Θεωρία Galois
  - ✓ Αρχικό πρόβλημα: εύρεση αναλυτικού τύπου πολυωνύμων 5ου ή μεγαλύτερου βαθμού
- Για κάθε πρώτο αριθμό p, η δομή (Zp, +p , *p) είναι πεπερασμένο πεδίο
  - ✓ (Zp, +p) είναι αβελιανή ομάδα
  - ✓ (Zp*, *p) είναι αβελιανή ομάδα
  - ✓ Συμβολίζεται με GF(p) (πεδίο Galois τάξης p)
- Θεώρημα: Αν p πρώτος, το πεδίο Galois GF(p) είναι το μοναδικό πεπερασμένο πεδίο με p στοιχεία.
- Αν p δεν είναι πρώτος?

Πολυώνυμα στο πεδίο Galois GF(p)

- $Z_p[x]$ = Το σύνολο όλων των πολυωνύμων με συντελεστές από το GF(p). π.χ. $f(x)=a_0+a_1x+\ldots+a_kx_k$ με $a_i \in GF(p)$ (ο βαθμός συμβολίζεται ως deg(f))
- Πρόσθεση και πολλαπλασιασμός πολυωνύμων γίνεται ως συνήθως αλλά στο τέλος παίρνουμε τους συντελεστές mod p.

  Στο GF(3) $(2x^2 + x)2x = 4x^3+2x^2= x^3 + 2x^2$
- Το $Z_p[x]$ με πρόσθεση και πολλαπλασιασμό mod p είναι δακτύλιος
- Παρατήρηση: υπάρχουν p^{k+1} πολυώνυμα βαθμού έως k στο $Z_p[x]$

Πολυώνυμα στο πεδίο Galois GF(p)

- Για p=2 οι συντελεστές a_i είναι από το δυαδικό σύστημα
- Πρόσθεση και πολλαπλασιασμός στο GF(2): XOR και AND

+	0	1
0	0	1
1	1	0

*	0	1
0	0	0
1	0	1

- Αρκετές ιδιότητες των ακεραίων του Zp ισχύουν για τα πολυώνυμα του Zp[x]
- Θα λέμε ότι f(x) | g(x) αν υπάρχει q(x) ∈ Zp[x] έτσι ώστε g(x) = q(x)f(x)
- Για πολυώνυμα f(x), g(x), h(x) θα λέμε ότι g(x) ≡ h(x) (mod f(x)) αν f(x) | (g(x)-h(x))

Πολυώνυμα στο πεδίο Galois GF(p)

- Θεώρημα της διαίρεσης για πολυώνυμα: Έστω πολυώνυμα f(x), g(x) ∈ Z_p[x] με deg(f) = n. Υπάρχουν μοναδικά πολυώνυμα q(x), r(x) ∈ Z_p[x] έτσι ώστε:
  - ✓ g(x) = q(x)f(x) + r(x) και deg(r) ≤ n-1
  - ✓ r(x) είναι το υπόλοιπο της διαίρεσης, g(x) ≡ r(x) (mod f(x))
- Παράδειγμα: έστω g(x) = $x^6 + x^4 + x^3 + x + 1$, f(x) = $x^3 + x + 1$

$$
\begin{array}{llll|l}
x^6 & +x^4+x^3 & +x+1 & & x^3+x+1 \\
\cline{5-5}
x^6 & +x^4+x^3 & & & x^3 \quad = q(x) \\
\cline{1-3}
 & & x+1 & & \quad = r(x)
\end{array}
$$

- g(x) = f(x)·x^3 + (x+1)

Άσκηση

- **Κάντε τις ακόλουθες διαιρέσεις**
  - ✓ $(x^{14}+x^{11}+x^{10}+x^9+x^8+x^3+x+1) / (x^7+x^5+x^2+1)$
  - ✓ $(x^4+x^2+x+1) / (x^2+x+1)$

- **Είδος άσκησης; Ατομική**
- **Χρόνος προετοιμασίας: 10 λεπτά**

Πολυώνυμα στο πεδίο Galois GF(p)

- Η αντίστοιχη έννοια των πρώτων αριθμών στο $Z_p[x]$ είναι τα αμείωτα ή ανάγωγα (irreducible) πολυώνυμα.
- Ορισμός: Ένα πολυώνυμο f(x) με συντελεστές από ένα σώμα F ονομάζεται ανάγωγο (ή αμείωτο) στο F αν δεν είναι δυνατόν να βρεθούν δύο πολυώνυμα με συντελεστές από το F, με μικρότερο (αλλά θετικό) βαθμό, τέτοια ώστε το γινόμενό τους να είναι το f(x)
  - ✓ Το αν ένα πολυώνυμο είναι ανάγωγο εξαρτάται από το σώμα στο οποίο το θεωρούμε
  - ✓ Π.χ. $g(x) = 2x^2+x$ δεν είναι αμείωτο στο GF(3)
  - ✓ Το $g(x)=x^2+1$ αμείωτο στο GF(3)
  - ✓ Το $g(x) = x^4 + 1$ είναι ανάγωγο στο R αλλά όχι ανάγωγο στο GF(2) διότι: $g(x) = (x +1)(x^3 + x^2 + x +1)$ στο GF(2)

Πολυώνυμα στο πεδίο Galois GF(p)

- Δεδομένου ενός πολυωνύμου f(x) με deg(f) = n, Zp[x]/(f(x)) = όλα τα πολυώνυμα του $Z_p[x]$ βαθμού ≤ n-1 : $p(x)=a_0+a_1x+\ldots+a_{n-1}x^{n-1}$ όπου $a_i \in GF(p)$ (= όλα τα πιθανά υπόλοιπα όταν διαιρούμε με f(x))

- Το Zp[x]/(f(x))
  - ✓ περιέχει ακριβώς p^n πολυώνυμα
  - ✓ είναι δακτύλιος με πρόσθεση και πολ/μό πολυωνύμων mod f(x) (αν στον πολ/μό προκύψει πολυώνυμο βαθμού ≥ n, το ανάγουμε mod f(x))
  - ✓ Είναι πεδίο αν και μόνο αν το f(x) είναι ανάγωγο
  - ✓ Εύρεση αντιστρόφου γίνεται με εκτενή αλγόριθμο Ευκλείδη όπως και στο Z_p

- Θεώρημα:
  - (i) Κάθε πεπερασμένο πεδίο έχει τάξη της μορφής p^n, όπου p πρώτος και n θετικός ακέραιος
  - (ii) Για κάθε πρώτο p και θετικό ακέραιο n, υπάρχει ένα μοναδικό πεπερασμένο πεδίο τάξης p^n, το οποίο συμβολίζουμε με $GF(p^n)$ και ταυτίζεται με το $Z_p[x]/(f(x))$, για κάποιο αμείωτο πολυώνυμο f(x) βαθμού n.
  - Αν υπαρχουν πολλά αμείωτα πολυώνυμα βαθμού n, δεν έχει σημασία ποιο επιλέγουμε. Τα πεδία που προκύπτουν είναι όλα ισομορφικά μεταξύ τους

- $GF(p^n)$: πολυώνυμα βαθμού αυστηρά μικρότερου του n με συντελεστές στο GF(p) και πράξεις πρόσθεση και πολ/μο mod f(x):
  - Οι συντελεστές ανάγονται mod p
  - Τα πολυώνυμα μετά τον πολ/μό ανάγονται mod f(x)

Πεπερασμένα πεδία μορφής GF(p^n)

- Π.χ., για p=3 (άρα a_i =0,1,2), και n=2 έχουμε 9 πολυώνυμα:
  - 0 1 2
  - x x+1 x+2
  - 2x 2x+1 2x+2
  - ✓ Η αριθμητική επί των συντελεστών γίνεται modulo 3
- Πεπερασμένα πεδία μορφής GF(2^n)
  - ✓ Θα μας απασχολήσουν κυρίως τέτοια πεδία (π.χ. στο AES)
  - ✓ Υπάρχει πεπερασμένο πεδίο με 4 στοιχεία GF(4) = GF(2^2)
  - ✓ Υπάρχει πεπερασμένο πεδίο με 8 στοιχεία GF(8) = GF(2^3)
  - ✓ Δεν υπάρχει πεπερασμένο πεδίο με 6 στοιχεία
    - επειδή το 6 δεν είναι δύναμη κανενός πρώτου αριθμού.

Πεπερασμένο πεδίο GF(2^3) ή GF(8)

Integer Representation	Binary Representation	Element of GF(8)
0	000	0
1	001	1
2	010	A
3	011	A + 1
4	100	A^2
5	101	$A^2 + 1$
6	110	$A^2 + A$
7	111	$A^2 + A + 1$

τα πολυώνυμα είναι

0	1	x	x+1
x^2	x^2+1	x^2+x	x^2+x+1

Κάθε πολυώνυμο αντιστοιχεί σε ένα binary string με τους συντελεστές

Από πίνακα:
πολυώνυμο 6 + πολυώνυμο 3 =
πολυώνυμο 5
Ισοδύναμα: $(x^2+x) + (x+1) = x^2+1$

	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	0	3	2	5	4	7	6
2	2	3	0	1	6	7	4	5
3	3	2	1	0	7	6	5	4
4	4	5	6	7	0	1	2	3
5	5	4	7	6	1	0	3	2
6	6	7	4	5	2	3	0	1
7	7	6	5	4	3	2	1	0

Πεπερασμένο πεδίο GF(2^3) ή GF(8)

- Εύρεση αμείωτου πολυωνύμου:
  - ✓ Εμπειρικά
  - ✓ Ένα πολυώνυμο f(x) βαθμού k ονομάζεται μονοειδές (monic) στο GF(p) αν ο συντελεστής a_k=1
  - ✓ Γενικά ψάχνουμε για μονοειδή αμείωτα πολυώνυμα βαθμού n=3
  - ✓ Ο σταθερός όρος πρέπει να είναι 1

Παράδειγμα αμείωτου πολυωνύμου

- Εφόσον οι συντελεστές a_i μπορεί να είναι μόνο 0 και 1, υποψήφια:
  - ✓ $p_0(x) = x^3 + 1$
  - ✓ $p_1(x) = x^3 + x + 1$
  - ✓ $p_2(x) = x^3 + x^2 + 1$
  - ✓ $p_3(x) = x^3 + x^2 + x + 1$
- Αλλά:
  - ✓ $p_0(x) = (x+1)(x^2+x+1)$
  - ✓ $p_3(x) = (x+1)(x^2+1)$

- Αμείωτα είναι τα $p_1(x)= x^3+x+1$ και $p_2(x)= x^3 +x^2+1$
- Μπορούμε να επιλέξουμε π.χ. το $p_1(x)$

Πεπερασμένα πεδία GF(2^m)

m	Default Primitive Polynomial	Integer Representation
1	D + 1	3
2	D^2 + D + 1	7
3	D^3 + D + 1	11
4	D^4 + D + 1	19
5	D^5 + D^2 + 1	37
6	D^6 + D + 1	67
7	D^7 + D^3 + 1	137
8	D^8 + D^4 + D^3 + D^2 + 1	285
9	D^9 + D^4 + 1	529
10	D^10 + D^3 + 1	1033
11	D^11 + D^2 + 1	2053
12	D^12 + D^6 + D^4 + D + 1	4179
13	D^13 + D^4 + D^3 + D + 1	8219
14	D^14 + D^10 + D^6 + D + 1	17475
15	D^15 + D + 1	32771
16	D^16 + D^12 + D^3 + D + 1	69643

- Αμείωτα πολυώνυμα για διάφορα m
- Αμείωτο πολυώνυμο στο GF(2^8)
  - ✓ **$m(x) = x^8+x^4+x^3+x+1$**
- Χρησιμοποιείται στα AES S-boxes
  - ✓ Υπολογίζεται ο πολλαπλασιαστικός αντίστροφος για κάθε byte εισόδου A(x)
- Δηλαδή υπολογίζεται το πολυώνυμο *G*(x) τέτοιο ώστε
  - ✓ A(x)*G*(x)=1 (mod m(x))

Byte ως πολυώνυμο

- Ένα byte $b_7b_6b_5b_4b_3b_2b_1b_0$ θεωρείται ως πολυώνυμο βαθμού 7, deg(f) = 7 με $b_i \in \{0,1\}$

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = \sum_{i=0}^{7} b_i x^i$$

- Για παράδειγμα το 01100011 είναι το πολυώνυμο: $x^6 + x^5 + x + 1$

Άσκηση

- Να βρείτε σε ποια πολυώνυμα αντιστοιχούν τα ακόλουθα bytes

- 1011 0111 στο GF(2^8)
- 1010 0101 στο GF(2^8)

- **Είδος άσκησης: ατομική**
- **Χρόνος προετοιμασίας: 5 λεπτά**

Πράξεις με bytes

- Πρόσθεση (XOR: modulo2)
  - ✓ Πρώτο byte: $\{a_7a_6a_5a_4a_3a_2a_1a_0\}$
  - ✓ Δεύτερο byte: $\{b_7b_6b_5b_4b_3b_2b_1b_0\}$
- Το άθροισμά τους είναι $\{c_7c_6c_5c_4c_3c_2c_1c_0\}$ όπου: $c_i = a_i \oplus b_i$,
  - ✓ $c_7 = a_7 \oplus b_7$, $c_6 = a_6 \oplus b_6$, ...$c_0 = a_0 \oplus b_0$

- Παράδειγμα:
- Ως πολυώνυμα: $(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$
- Ως bytes: $\{01010111\} \oplus \{10000011\} = \{11010100\}$
- Ως HEX: $\{57\} \oplus \{83\} = \{d4\}$

Πράξεις με bytes

- Πολλαπλασιασμός
- Γίνεται πολλαπλασιασμός των πολυωνύμων modulo $\mathbf{x^8+x^4+x^3+x+1}$

- **Οπότε το αποτέλεσμα θα είναι πάντα ένα πολυώνυμο με βαθμό <8**

- Παράδειγμα:
- Ως bytes: {01010111} *{10000011}
- Ως HEX:{57} *{83} = {c1}

- Ως πολυώνυμα: $(x^6+ x^4+ x^2 + x +1) * (x^7+ x +1) = x^{13} + x^{11} + x^9 + x^8 + x^7 + x^7 + x^5 + x^3 + x^2 + x + x^6 + x^4 + x^2 + x +1 = x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 +1$

- $(x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 +1)$ modulo $(\mathbf{x^8+x^4+x^3+x+1}) = x^7+x^6+1=11000001=1100 0001=C1$

Πράξεις με bytes

- Πολλαπλασιασμός με X
- x*b(x) γίνεται με αριστερή ολίσθηση του byte (xtime)

- Παράδειγμα:
- {57}*{02}=xtime({57})

- 01010111 * 00000010=10101110

Άσκηση

- **Να κάνετε τις παρακάτω πράξεις:**
  - ✓ 0x*B*7 $\oplus$ 0xA5
  - ✓ 0x3A $\oplus$ 0x2C
  - ✓ 0x*B*7 * 0xA5
  - ✓ 0x3A * 0x2C
- **Είδος άσκησης: Ατομική**
- **Χρόνος προετοιμασίας: 10 λεπτά**

Advanced Encryption Standard (AES)

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

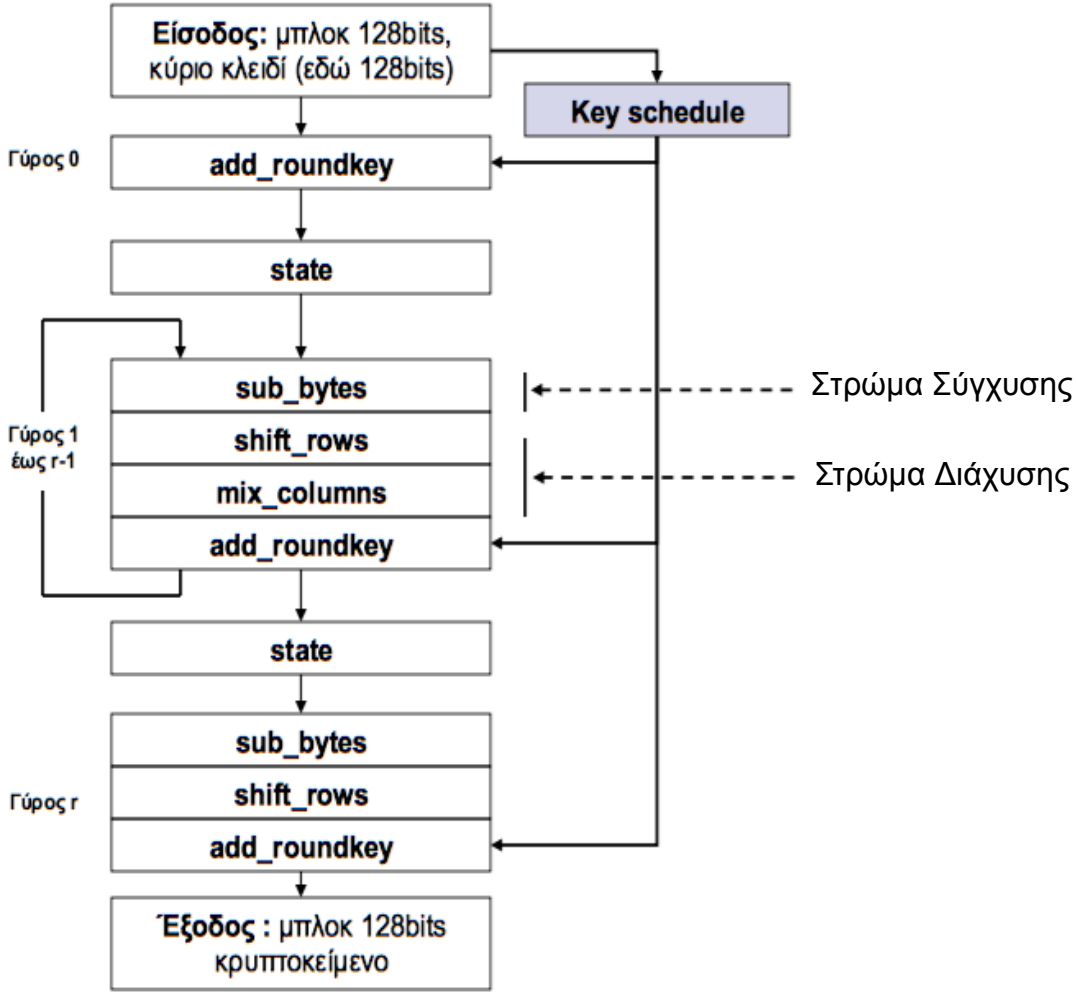
Χαρακτηριστικά

- 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

Κρύπτο

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}

:=

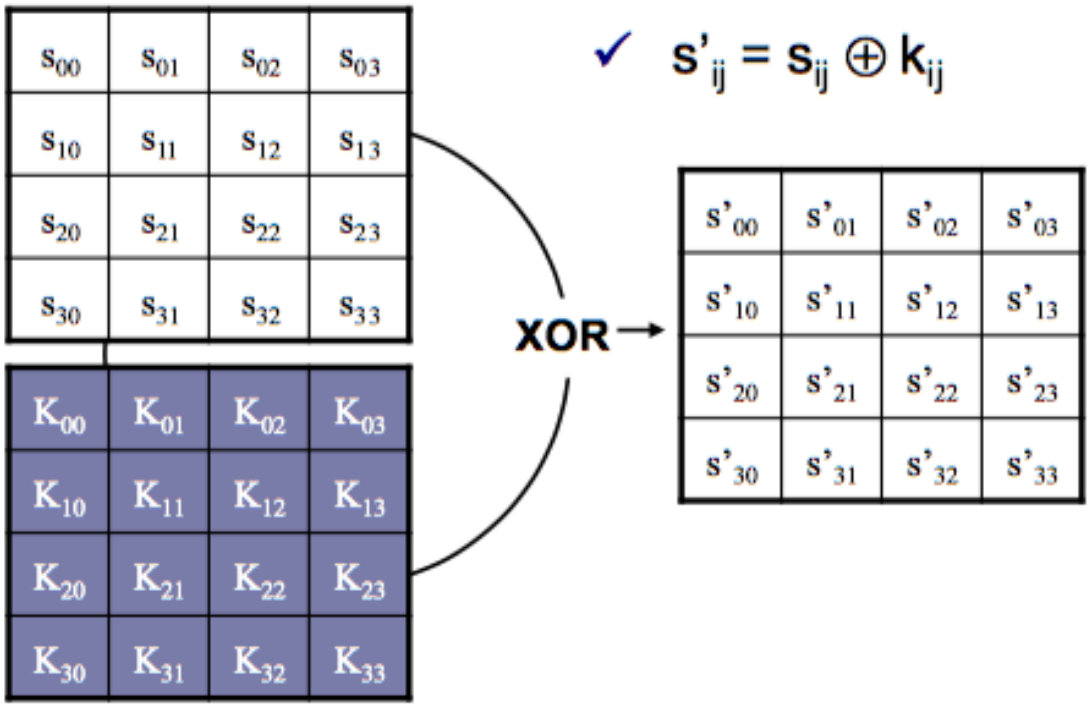
x_0	x_4	x_8	x_{12}
x_1	x_5	x_9	x_{13}
x_2	x_6	x_{10}	x_{14}
x_3	x_7	x_{11}	x_{15}

(= 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}



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 $x^7 + x^6 + x^3 + x = (11001010) = (a_7a_6a_5a_4a_3a_2a_1a_0)$

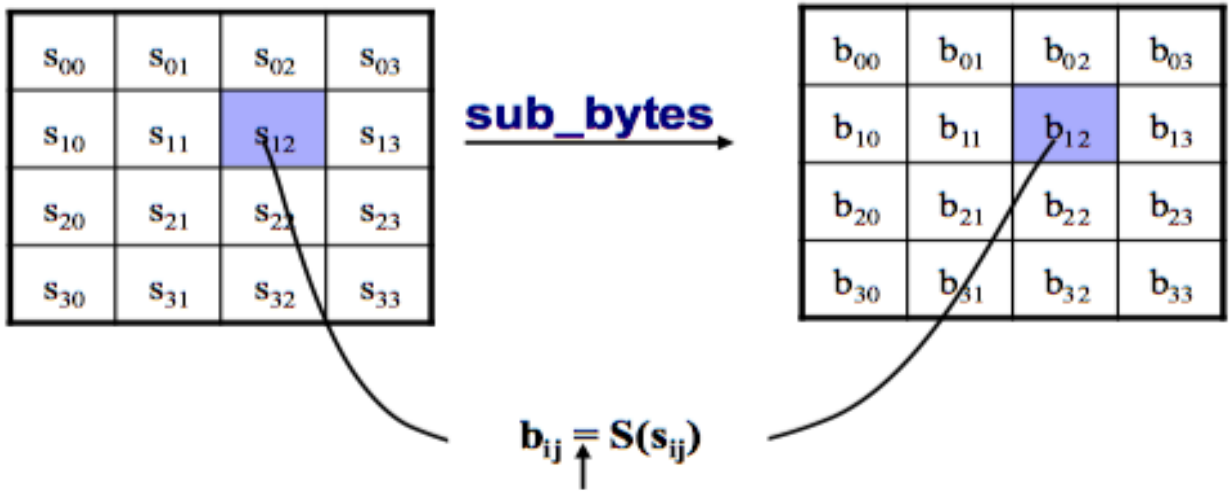
- Έστω $c_7c_6c_5c_4c_3c_2c_1c_0$=01100011 (σταθερός πίνακας του μετασχηματισμού)
- Ένα καλό κόλπο για το γραμμικό μετασχηματισμό:
  - ✓ for i=0 to 7
    $b_i \leftarrow (a_i+a_{i+4}+a_{i+5}+a_{i+6}+a_{i+7}+ c_i)$ mod2 /* δείκτες ανάγονται mod8
- Return $(b_7b_6b_5b_4b_3b_2b_1b_0)$

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 $b = b_7b_6b_5b_4b_3b_2b_1b_0 = (11101101) = \{ED\}_{16}$

- 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



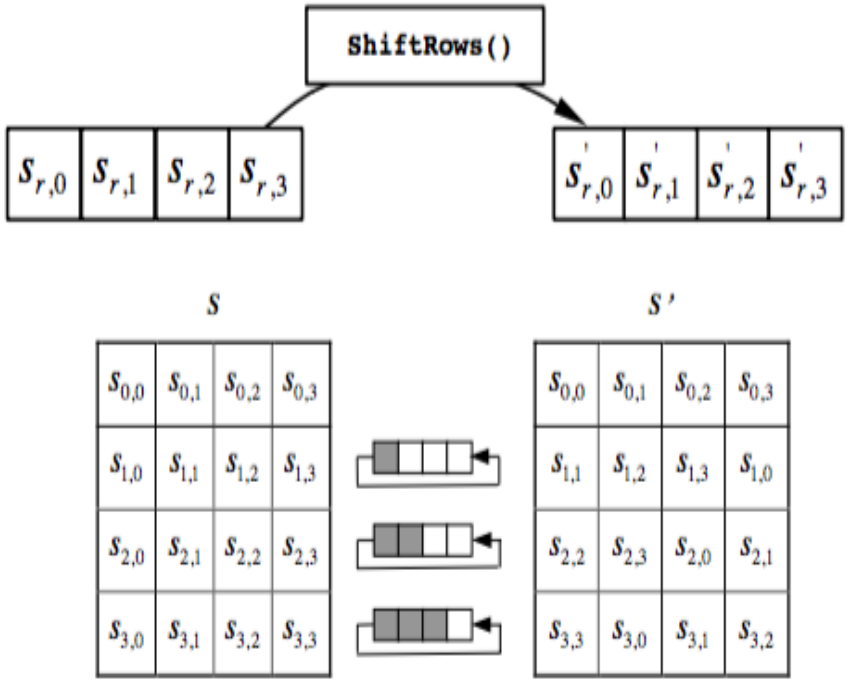
53 →ed

	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 3: αριστερή ολίσθηση κατά 3 θέσεις

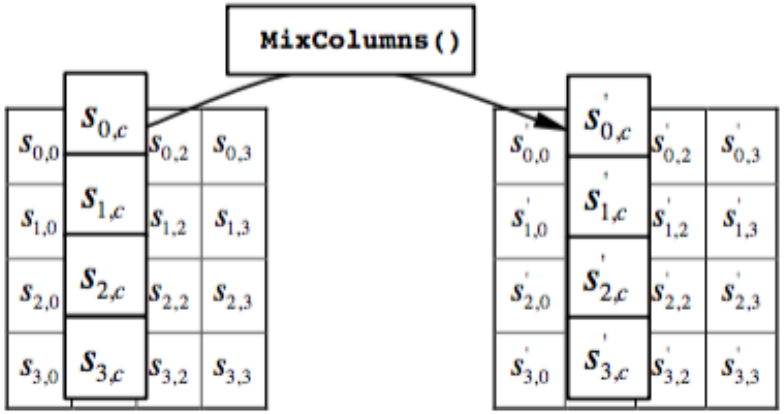


Στρώμα διάχυσης – 2η διαδικασία: mix_columns

- 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 S'(x) το πολυώνυμο που προκύπτει
- Τελικά η στήλη αντικαθίσταται από τα bytes που αντιστοιχούν στους συντελεστές του S'(x)

Στρώμα διάχυσης – 2η διαδικασία: mix_columns

- Τελικά, η διαδικασία μπορεί να κωδικοποιηθεί ως:
  - $S'_i = 2S_i + 3S_j + S_k + S_m$
  - $S'_j = S_i + 2S_j + 3S_k + S_m$
  - $S'_k = S_i + S_j + 2S_k + 3S_m$
  - $S'_m = 3S_i + S_j + S_k + 2S_m$



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

for $0 \le c < Nb$.

Key Schedule

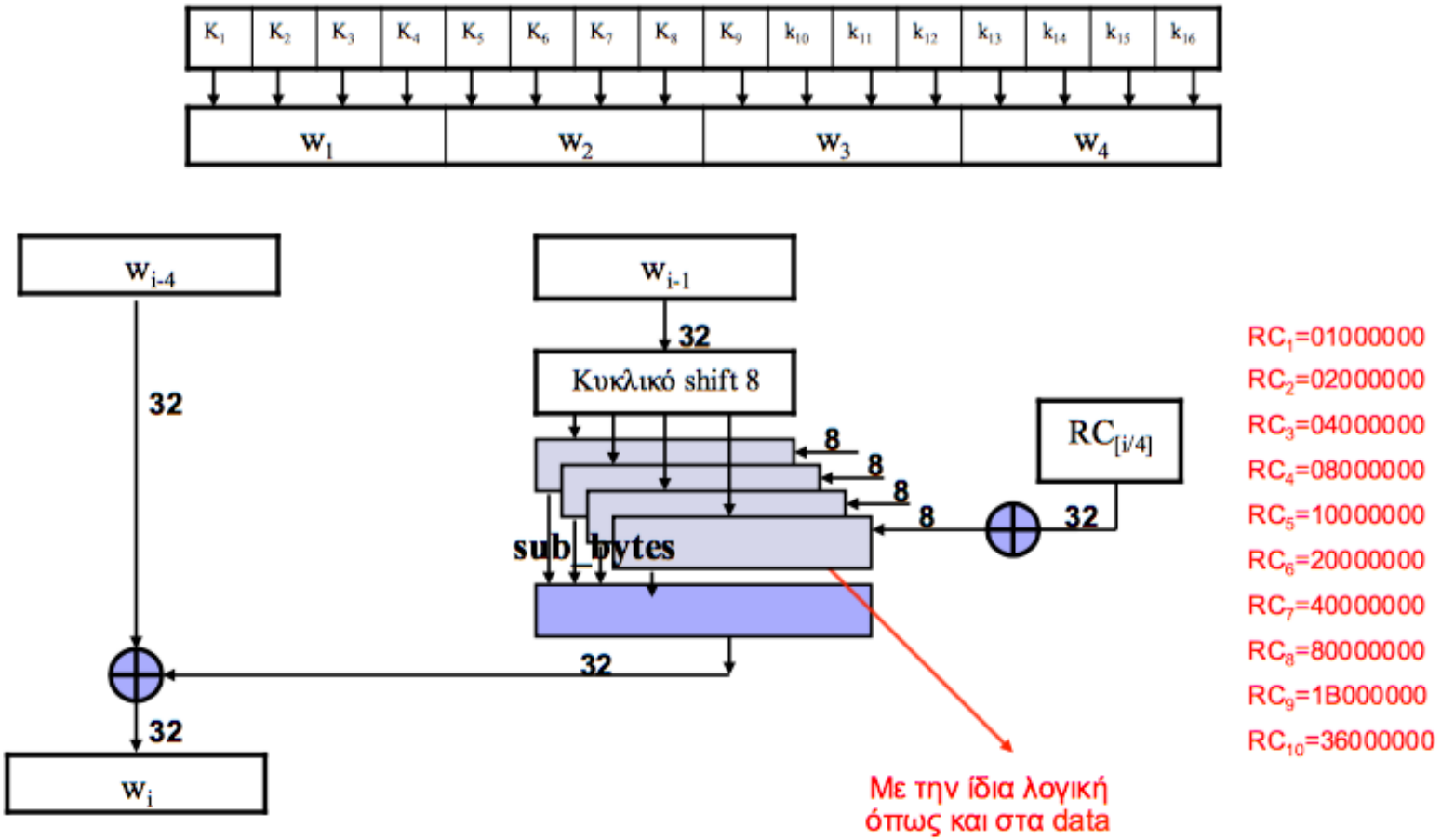
- Είσοδος: Κλειδί 128bits (master key)
- Έξοδος: Για 10 γύρους χρειαζόμαστε 11 subkeys των 128 bits
  - ✓ 4 λέξεις (words) για κάθε κλειδί (1 word = 4 bytes)
  - ✓ Συνολικά πρέπει να παράγουμε 44 λέξεις (44x32=1408 bits) σε ένα πίνακα $w_0,\ldots, w_{43}$
- Οι λέξεις w χρησιμοποιούνται ανά τέσσερις σε κάθε διαδικασία add_roundkey
- Αρχικά οι τέσσερις πρώτες λέξεις $_{w0}$ 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

- 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

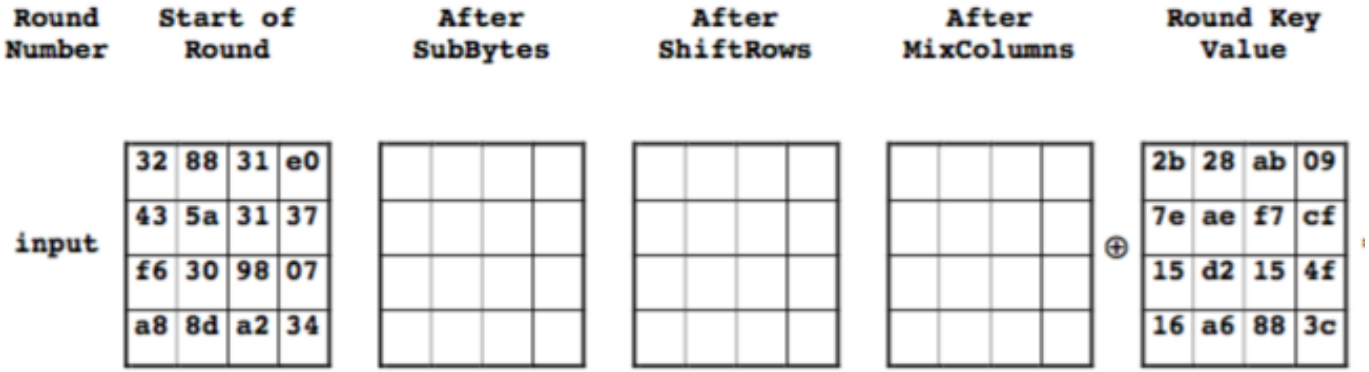
```
Key expansion(key)
    (w0,…, w3) = key
    for i=4 to 43 {
        temp = wi-1
        if (i≡0 mod 4)
            then temp = sub_word(rot_word(temp))⊕ RCi/4
        wi = wi-4 ⊕ temp
    }
```

Key Schedule



Παράδειγμα

- 128 bit Cipher key
- Nk=4
- Input: 32 43 F6 A8 88 5A 30 8D 31 31 98 A2 E0 37 07 34
- Cipher Key: 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c



Παράδειγμα

- ROUND 0
  - ✓ Συγκεκριμένα γίνονται οι εξής διεργασίες
    - Το αρχικό input μπαίνει σε έναν πίνακα 4x4
    - Το αρχικό κλειδί μπαίνει σε έναν πίνακα 4x4
    - Η Νέα κατάσταση (state) προκύπτει από XOR του input με το αρχικό κλειδί

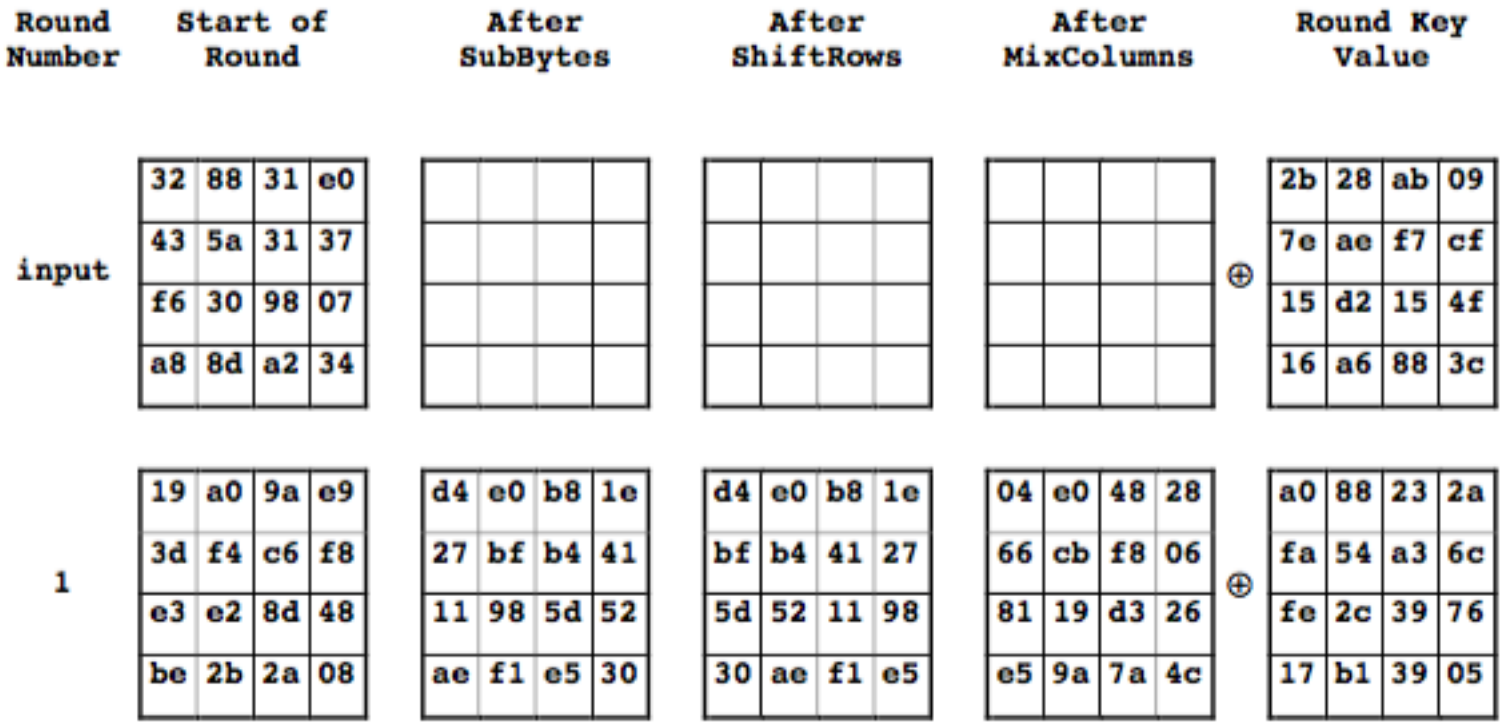
$$S=\begin{bmatrix} 32 & 88 & 31 & E0 \\ 43 & 5A & 31 & 37 \\ F6 & 30 & 98 & 07 \\ A8 & 8D & A2 & 34 \end{bmatrix} \oplus \begin{bmatrix} 2B & 28 & AB & 09 \\ 7E & AE & F7 & CF \\ 15 & D2 & 15 & 4F \\ 16 & A6 & 88 & 3C \end{bmatrix} = \begin{bmatrix} 19 & A0 & 9A & E9 \\ 3D & F4 & C6 & F8 \\ E3 & E2 & 8D & 48 \\ BE & 2B & 2A & 08 \end{bmatrix}$$

Παράδειγμα

- Υπολογισμός κλειδιού (ROUND 1)
  - ✓ add_roundkey για round 1
  - ✓ Cipher Key: 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c
  - ✓ Χωρίζουμε το key σε 4 words (32bits) ως ακολούθως:
    - W_0= 2b 7e 15 16
    - W_1= 28 ae d2 a6
    - W_2= ab f7 15 88
    - W_3= 09 cf 4f 3c
  - ✓ Για τον υπολογισμό του w_4 έχω:
    - Temp = w_{i-1}=w_3=09 cf 4f 3c
    - RotWord: cf 4f 3c 09
    - Subword: 8a 84 eb 01(κάνω την αντικατάσταση από τον lookup table)
    - Rcon [i/Nk]=Rcon [4/4]= 01000000 ,
    - W_4=w_0 ⊕ g(w[3])= 2b 7e 15 16 ⊕ 8a 84 eb 01= a0fafe17
    - W_5=w_{i-4} ⊕ temp=w_{i-4} ⊕ w_{i-1}=w_1 ⊕ w_4= 28 ae d2 a6 ⊕ a0fafe17 =88542cb1
    - W_6=23a33939
    - W_7=2a6c7605

Παράδειγμα

- Round 1



Παράδειγμα

- subbytes

$$\begin{bmatrix} 19 & A0 & 9A & E9 \\ 3D & F4 & C6 & F8 \\ E3 & E2 & 8D & 48 \\ BE & 2B & 2A & 08 \end{bmatrix} \rightarrow \begin{bmatrix} d4 & e0 & b8 & 1e \\ 27 & bf & b4 & 41 \\ 11 & 98 & 5d & 52 \\ ae & f1 & e5 & 30 \end{bmatrix}$$

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
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

Παράδειγμα

- ShiftRows

$$\begin{bmatrix} d4 & e0 & b8 & 1e \\ 27 & bf & b4 & 41 \\ 11 & 98 & 5d & 52 \\ ae & f1 & e5 & 30 \end{bmatrix} \rightarrow \begin{bmatrix} d4 & e0 & b8 & 1e \\ bf & b4 & 41 & 27 \\ 5d & 52 & 11 & 98 \\ 30 & ae & f1 & e5 \end{bmatrix}$$

Παράδειγμα

- MixColumns

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

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \oplus \begin{bmatrix} d4 & e0 & b8 & 1e \\ bf & b4 & 41 & 27 \\ 5d & 52 & 11 & 98 \\ 30 & ae & f1 & e5 \end{bmatrix} = \begin{bmatrix} 04 & e0 & 48 & 28 \\ 66 & cb & f8 & 06 \\ 81 & 19 & d3 & 26 \\ e5 & 9a & 7a & 4c \end{bmatrix}$$

- {02}*{d4}⊕ {03}*{bf} ⊕{01}*{5d} ⊕ {01}*{30}

- Το νέο State του Round 2 προκύπτει από την εξής πράξη:
  - ✓ Τελευταίο state **XOR** Κλειδί Γύρου 1

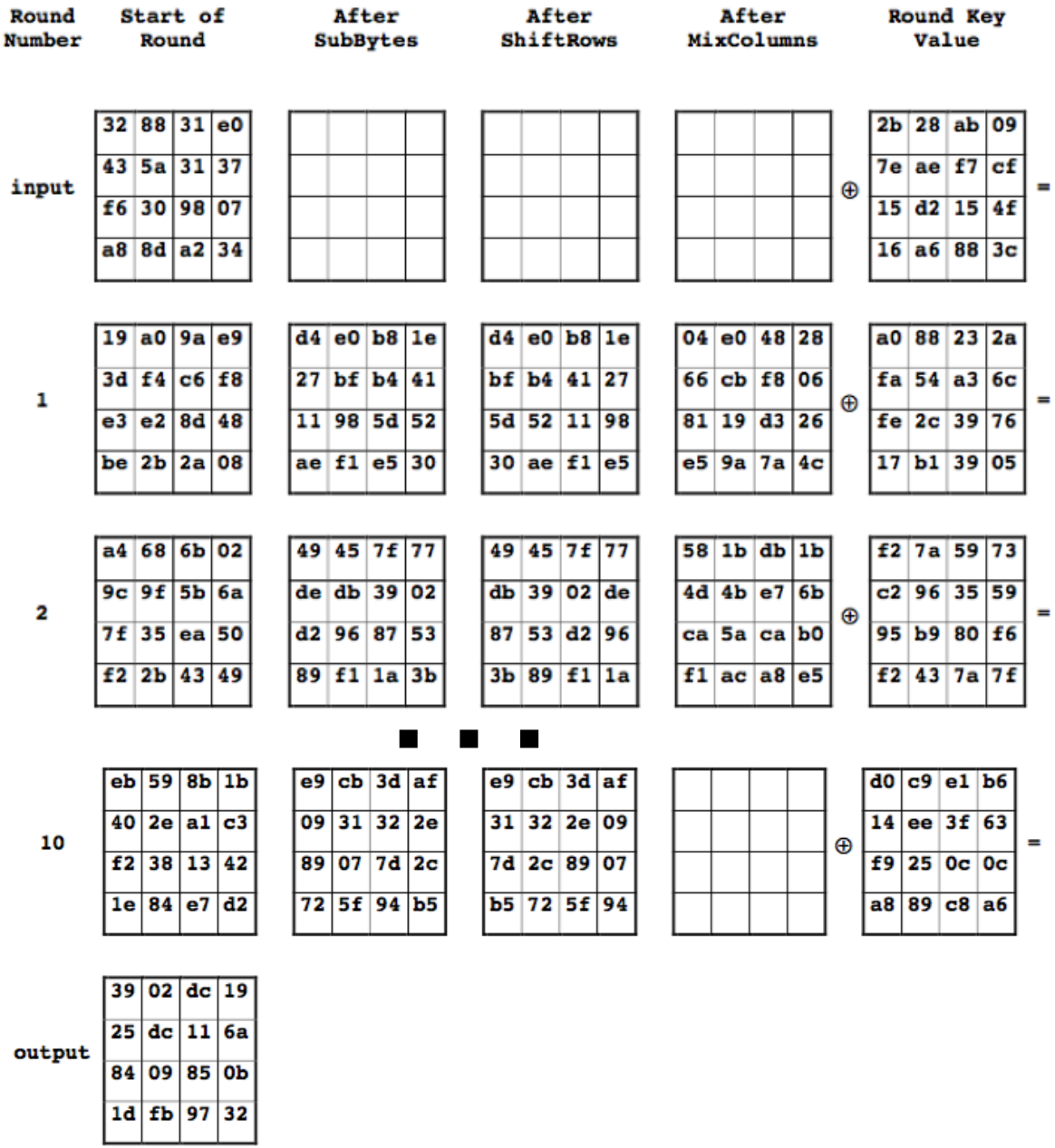
04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

$\oplus$

a0	88	23	2a
fa	54	a3	6c
fe	2c	39	76
17	b1	39	05

=

a4	68	6b	02
9c	9f	5b	6a
7f	35	ea	50
f2	2b	43	49


Round Number
Start of Round
After SubBytes
After ShiftRows
After MixColumns
Round Key Value
input
32 88 31 e0
43 5a 31 37
f6 30 98 07
a8 8d a2 34
2b 28 ab 09
7e ae f7 cf
15 d2 15 4f
16 a6 88 3c
1
19 a0 9a e9
3d f4 c6 f8
e3 e2 8d 48
be 2b 2a 08
d4 e0 b8 1e
27 bf b4 41
11 98 5d 52
ae f1 e5 30
d4 e0 b8 1e
bf b4 41 27
5d 52 11 98
30 ae f1 e5
04 e0 48 28
66 cb f8 06
81 19 d3 26
e5 9a 7a 4c
a0 88 23 2a
fa 54 a3 6c
fe 2c 39 76
17 b1 39 05
2
a4 68 6b 02
9c 9f 5b 6a
7f 35 ea 50
f2 2b 43 49
49 45 7f 77
de db 39 02
d2 96 87 53
89 f1 1a 3b
49 45 7f 77
db 39 02 de
87 53 d2 96
3b 89 f1 1a
58 1b db 1b
4d 4b e7 6b
ca 5a ca b0
f1 ac a8 e5
f2 7a 59 73
c2 96 35 59
95 b9 80 f6
f2 43 7a 7f
10
eb 59 8b 1b
40 2e a1 c3
f2 38 13 42
1e 84 e7 d2
e9 cb 3d af
09 31 32 2e
89 07 7d 2c
72 5f 94 b5
e9 cb 3d af
31 32 2e 09
7d 2c 89 07
b5 72 5f 94
d0 c9 e1 b6
14 ee 3f 63
f9 25 0c 0c
a8 89 c8 a6
output
39 02 dc 19
25 dc 11 6a
84 09 85 0b
1d fb 97 32

Παράδειγμα

Υπολογισμός των κλειδιών κάθε γύρου

i (dec)	temp	After RotWord()	After SubWord()	Rcon[i/Nk]	After XOR with Rcon	w[i–Nk]	w[i]= temp XOR w[i-Nk]
4	09cf4f3c	cf4f3c09	8a84eb01	01000000	8b84eb01	2b7e1516	a0fafe17
5	a0fafe17					28aed2a6	88542cb1
6	88542cb1					abf71588	23a33939
7	23a33939					09cf4f3c	2a6c7605
8	2a6c7605	6c76052a	50386be5	02000000	52386be5	a0fafe17	f2c295f2
9	f2c295f2					88542cb1	7a96b943
10	7a96b943					23a33939	5935807a
11	5935807a					2a6c7605	7359f67f
12	7359f67f	59f67f73	cb42d28f	04000000	cf42d28f	f2c295f2	3d80477d
13	3d80477d					7a96b943	4716fe3e
14	4716fe3e					5935807a	1e237e44
15	1e237e44					7359f67f	6d7a883b
16	6d7a883b	7a883b6d	dac4e23c	08000000	d2c4e23c	3d80477d	ef44a541

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

Άσκηση

- Key: This is the key
- In HEX: 54 68 69 73 20 69 73 20 74 68 65 20 6b 65 79
- Binary:
- 01010100 01101000 01101001 01110011 00100000 01101001 01110011 00100000 01110100 01101000 01100101 00100000 01101011 01100101 01111001

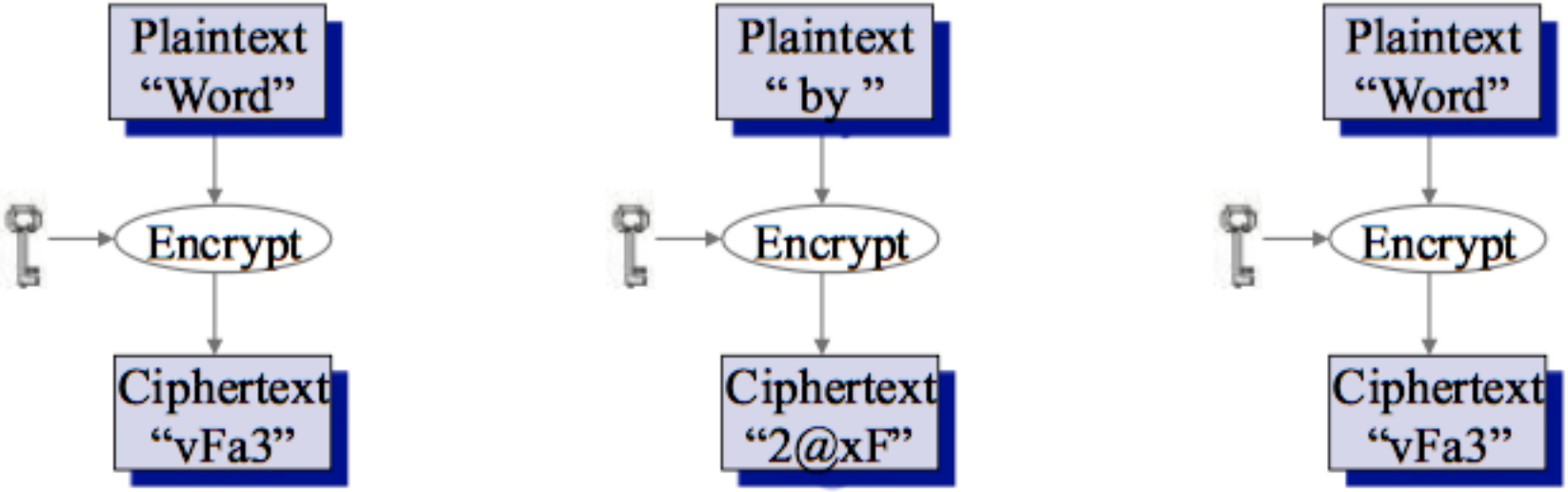
- Plaintext: This is the secret
- In HEX: 54 68 69 73 20 69 73 20 74 68 65 20 73 65 63 72 65 74
- Binary:
- 01010100 01101000 01101001 01110011 00100000 01101001 01110011 00100000 01110100 01101000 01100101 00100000 01110011 01100101 01100011 01110010 01100101 01110100

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

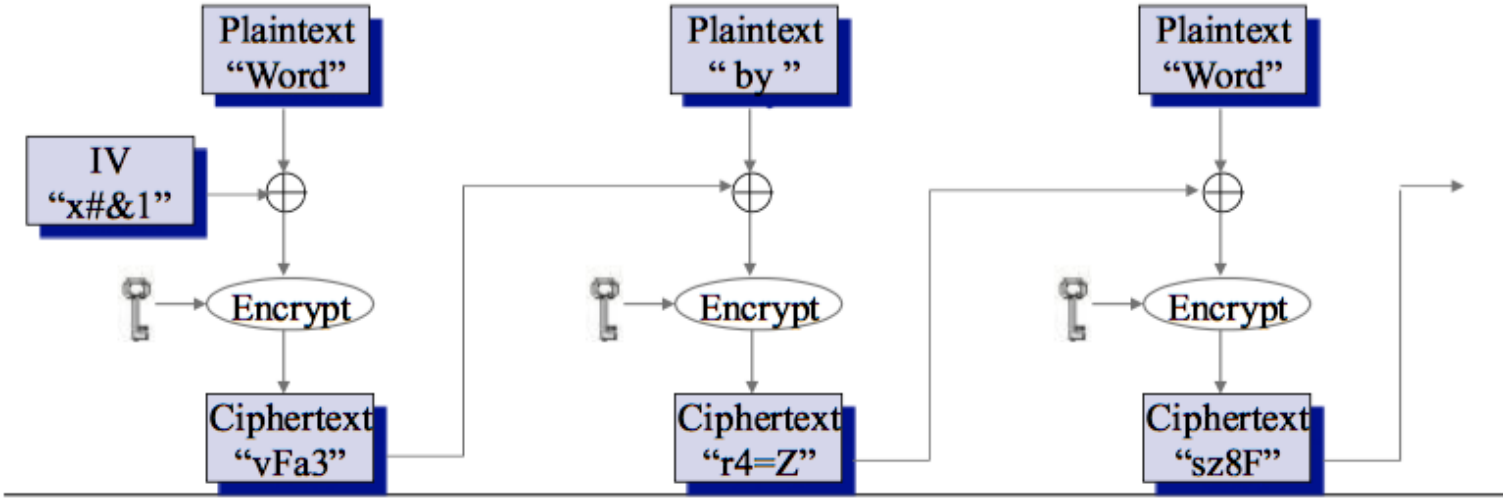
ECB – Electronic Codebook

- 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



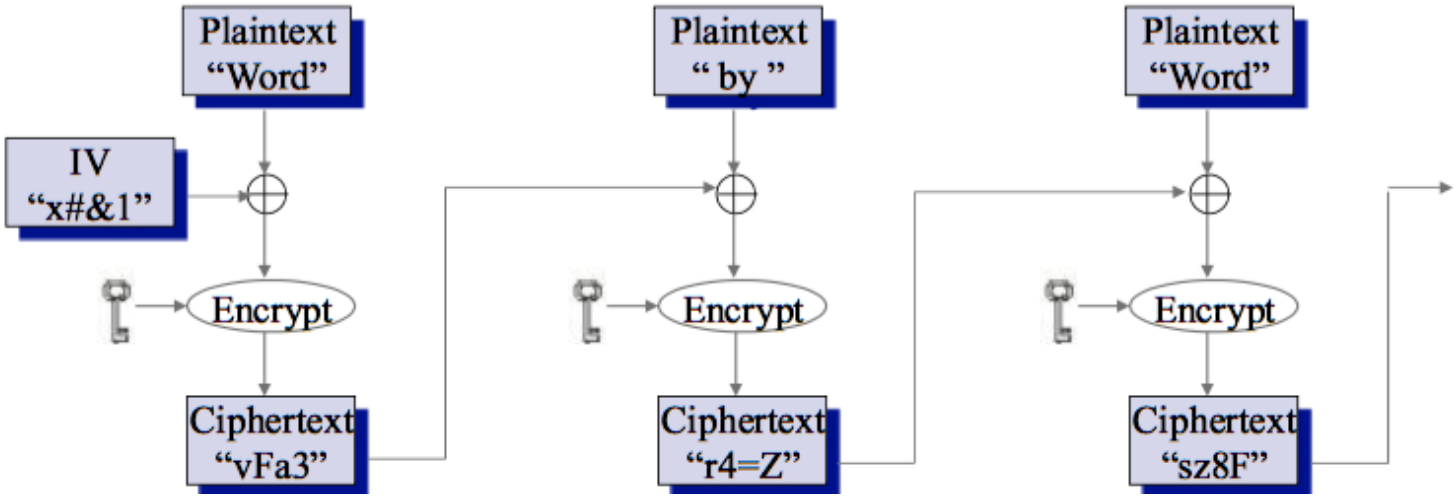
CBC – Cipher Block Chaining

- 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)



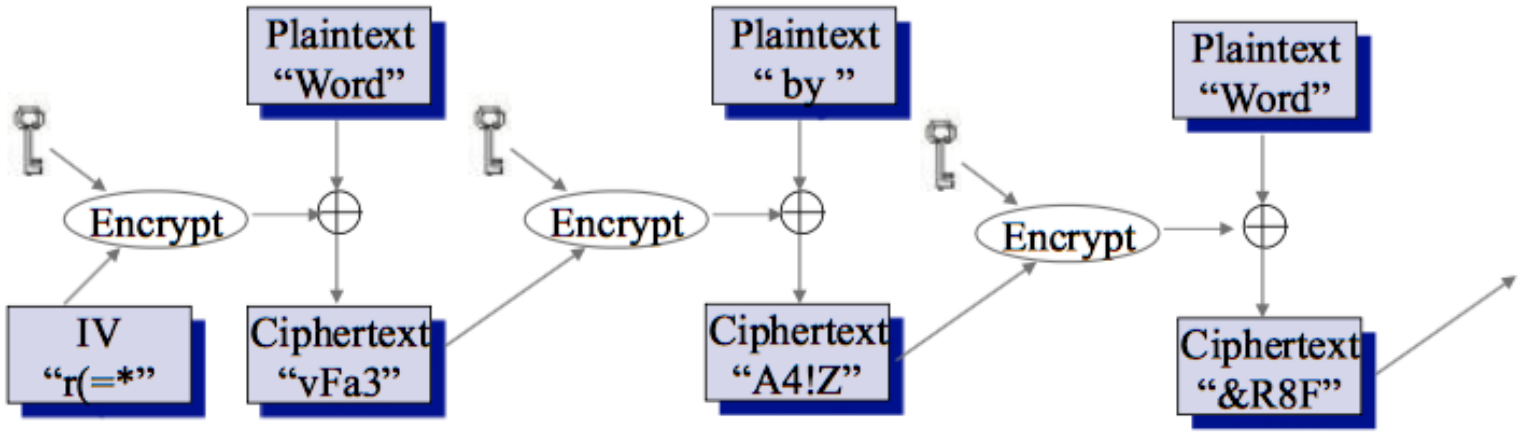
CBC – Cipher Block Chaining

- ***Chaining dependencies:*** Το ciphertext cj 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



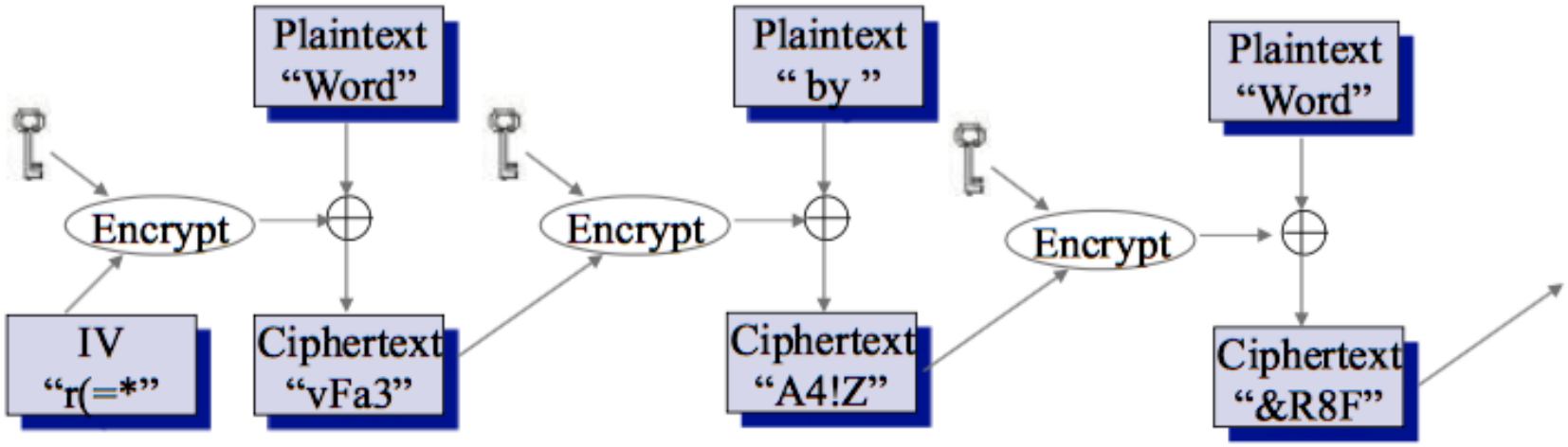
CFB – Cipher Feedback

- 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: $c_i = ek(c_{i-}1) \oplus x_i$
  - ✓ Decryption: $x_i = ek(c_{i-1}) \oplus c_i$
  - Ίδια plaintext blocks **δεν** παράγουν ίδιο ciphertext block (μόνο για ίδιο IV)



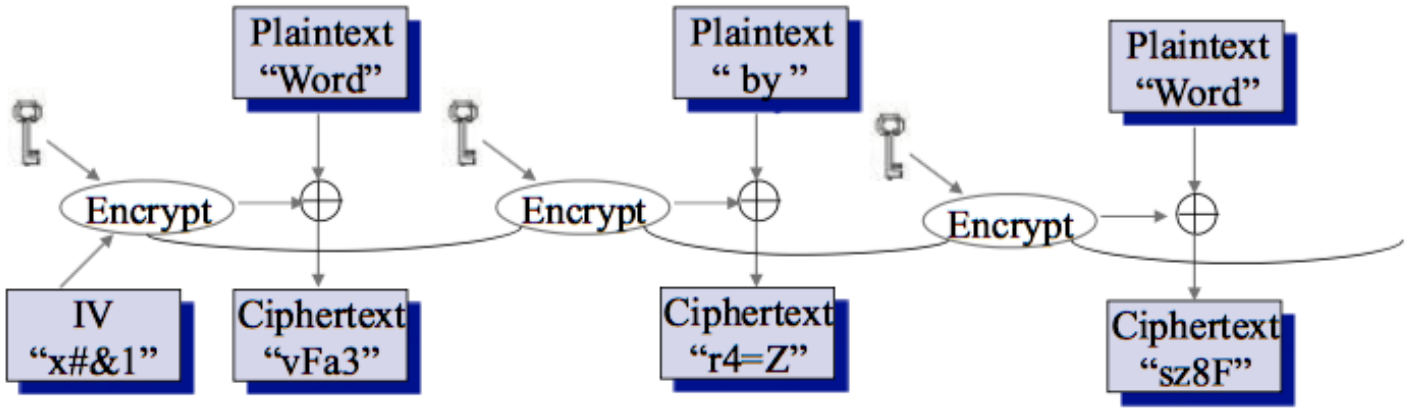
CFB – Cipher Feedback

- ***Chaining dependencies:*** Όπως και στο CBC
- ***Error propagation:*** ένα η περισσότερα bit errors σε ένα ciphertext block cj 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



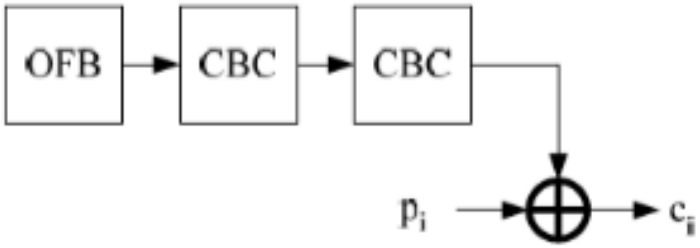
OFB – Output Feedback (ISO 10116)

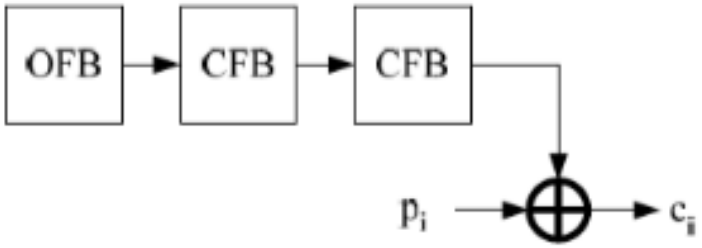
- Το keystream του προηγούμενου block επανα-κρυπτογραφείται παράγοντας ένα νέο keystream. Το νέο keystream συνδυάζεται με XOR με το plaintext του τρέχοντος block
  - ✓ Συνήθως με ένα μέρος αυτού, δηλαδή ένα r-bit plaintext < n
  - ✓ Απαιτείται Initialization Vector (IV)
- Ίδια plaintext blocks **δεν** 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
- **Errorrecovery**: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

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Hexadecimal (base 16)

Hex	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

- $OxA=(A)_{16} = (1010)_2$
- $OxE=(E)_{16}=(1110)_2$
- $OxE=(4)_{16}=(0100)_2$

- $Ox4E=(4E)_{16} = (0100\ 1110)_2$

References – Useful Material

- AES: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
- AES Implementation in PHP
- http://www.movable-type.co.uk/scripts/aes-php.html
- AES Implementation in .NET (C# and VB)
  - ✓ https://msdn.microsoft.com/en-us/library/system.security.cryptography.aes(v=vs.110).aspx

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Οικονομικό Πανεπιστήμιο Αθηνών
Τμήμα Πληροφορικής
ΠΜΣ στα Πληροφοριακά Συστήματα

Κρυπτογραφία και Εφαρμογές
Διαλέξεις Ακ. Έτους 2015-2016

Μαρκάκης Ευάγγελος
markakis@aueb.gr

Ντούσκας Θεόδωρος
tntouskas@aueb.gr