# Κρυπτογραφία

## Ψευδοτυχαιότητα - Κρυπτοσυστήματα ροής

Άρης Παγουρτζής - Πέτρος Ποτίκας

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

# Περιεχόμενα

# Εισαγωγή

- Τυχαίοι αριθμοί αποτελούν σημαντικό στοιχείο της κρυπτογραφίας

# Εισαγωγή

- Τυχαίοι αριθμοί αποτελούν σημαντικό στοιχείο της κρυπτογραφίας
- Αλγόριθμοι και πρωτόκολλα τους χρησιμοποιούν:
  - Κατανομή κλειδιών, σχήματα αυθεντικοποίησης
  - Παραγωγή κλειδιών συνεδρίας
  - Παραγωγή ροής από bit για συμμετρική κρυπτογράφηση (stream cipher)

# Ψευδοτυχαίες συμβολοσειρές

- Ιδέα: κάτι που "μοιάζει" με τυχαίο, αλλά δεν είναι πραγματικά
- Δε ξεχωρίζει ένα τυχαίο string από ένα που δημιουργείται από τη γεννήτρια ψευδοτυχαιότητας
- Εφαρμογή ψευδοτυχαιότητας και αλλού όπως π.χ. παίγνια, δειγματοληψία
- Θα την χρησιμοποιήσουμε για να αποδείξουμε την ασφάλεια σχημάτων κρυπτογράφησης ιδιωτικού κλειδιού

- Κατανομή πάνω σε strings: $D\colon \{0,1\}^n \to [0,1]$, ώστε $\Sigma_x D(x) = 1$
- Ορισμός ψευδοτυχαιότητας μέσω στατιστικών τεστ: Μια κατανομή $D$ πάνω σε $n$-bit strings είναι ψευδοτυχαία αν ικανοποιεί κάποια τεστ
  1. $\Pr_{x \leftarrow D}[\text{1ο bit του } x = 1] = 1/2$
  2. $\Pr_{x \leftarrow D}[\text{parity του } x = 1] = 1/2$
  3. …
  4. $\Pr_{x \leftarrow D}[\#1 = \#0 \text{ in } x] = 1/2$

- Κατανομή πάνω σε strings: $D\colon \{0,1\}^n \to [0,1]$, ώστε $\Sigma_x D(x) = 1$
- Ορισμός ψευδοτυχαιότητας μέσω στατιστικών τεστ: Μια κατανομή $D$ πάνω σε $n$-bit strings είναι ψευδοτυχαία αν ικανοποιεί κάποια τεστ
  1. $\Pr_{x \leftarrow D}[\text{1ο bit του } x = 1] = 1/2$
  2. $\Pr_{x \leftarrow D}[\text{parity του } x = 1] = 1/2$
  3. …
  4. $\Pr_{x \leftarrow D}[\#1 = \#0 \text{ in } x] = 1/2$
- Όμως με αντίπαλο, δε γνωρίζουμε τα τεστ που έχει

- Κατανομή πάνω σε strings: $D\colon \{0,1\}^n \to [0,1]$, ώστε $\Sigma_x D(x) = 1$
- Ορισμός ψευδοτυχαιότητας μέσω στατιστικών τεστ: Μια κατανομή $D$ πάνω σε $n$-bit strings είναι ψευδοτυχαία αν ικανοποιεί κάποια τεστ
  1. $\Pr_{x \leftarrow D}[\text{1ο bit του } x = 1] = 1/2$
  2. $\Pr_{x \leftarrow D}[\text{parity του } x = 1] = 1/2$
  3. …
  4. $\Pr_{x \leftarrow D}[\#1 = \#0 \text{ in } x] = 1/2$
- Όμως με αντίπαλο, δε γνωρίζουμε τα τεστ που έχει
- Κρυπτογραφικά, η κατανομή $D$ είνα ψευδοτυχαία, αν περνάει όλα τα *αποδοτικά* στατιστικά τεστ

# PRG

## Ορισμός

Μια γεννήτρια ψευδοτυχαιότητας (PRG) είναι ένας αποδοτικός, ντετερμινιστικός αλγόριθμος που επεκτείνει ένα *μικρό, ομοιόμορφο σπόρο* σε μια μεγαλύτερη, ψευδοτυχαία έξοδο.

- Από λίγα πραγματικά τυχαία bits, παράγονται πολλά περισσότερα bits που "φαίνονται" τυχαία
- Παραγωγή πραγματικά τυχαίων bits είναι χρονοβόρα
- Φροντίδα για τον σπόρο

- $n$ παράμετρος ασφαλείας, $p$ πολυώνυμο
- $D_n$: κατανομή σε $p(n)$-bit strings
- Η ψευδοτυχαιότητα είναι μια ιδιότητα μιας ακολουθίας κατανομών $\{D_n\} = \{D_1, D_2, ...\}$.

## Ορισμός

Η $\{D_n\}$ είναι ψευδοτυχαία αν για κάθε πιθανοτικό πολυωνυμικού χρόνου (PPT) αλγόριθμο $\mathcal{A}$, υπάρχει μια αμελητέα[1] συνάρτηση $\epsilon$ τ.ω.

$$|Pr_{x \leftarrow D_n}[\mathcal{A}(x) = 1] - Pr_{x \leftarrow U_{p(n)}}[\mathcal{A}(x) = 1]| \leq \epsilon(n)$$

[1] αμελητέα συνάρτηση: για κάθε σταθερά $c$, η τιμή της συνάρτησης είναι μικρότερη από $n^{-c}$, ασυμπτωτικά

# PseudoRandom Generator (PRG)

- Έστω $G$ ένας ντετερμινιστικός αλγόριθμος πολυωνυμικού χρόνου
- Ο $G$ επεκτείνει την είσοδο του αν: $|G(x)| = p(|x|) > |x|$
- Ο $G$ ορίζει μια ακολουθία κατανομών: $D_n^G$ = η κατανομή στα $p(n)$-bit strings των εξόδων $G(x)$ όταν $x \leftarrow U_n$

# PseudoRandom Generator (PRG)

## Ορισμός

$G$ είναι ψευδοτυχαίος (PRG) αν $\{D_n^G\}$ είναι ψευδοτυχαία, δηλ. για κάθε πιθανοτικό πολυωνυμικού χρόνου αντίπαλο (PPT) $\mathcal{A}$, υπάρχει μια αμελητέα συνάρτηση $\epsilon$, ώστε

$$|Pr_{x \leftarrow U_n}[\mathcal{A}(G(x)) = 1] - Pr_{y \leftarrow U_{p(n)}}[\mathcal{A}(y) = 1]| \leq \epsilon(n)$$

- Παρατηρήσεις: ντετερμινιστικός και αποδοτικός (πολυωνυμικός) αλγόριθμος
- Είναι τυχαία η κατανομή; Όχι τελείως!

- Παρατηρήσεις: ντετερμινιστικός και αποδοτικός (πολυωνυμικός) αλγόριθμος
- Είναι τυχαία η κατανομή; Όχι τελείως!
- αν $p(n) = n + 1$, τότε η ομοιόμορφη κατανομή στο $\{0, 1\}^{n+1}$, έχει χώρο $2^{n+1}$, άρα η πιθανότητα να επιλεγεί μια συμβολοσειρά είναι $1/2^{n+1}$,
- $|dom(G)| = 2^n, |range(G)| = 2^{n+1}$, άρα πιθανότητα μια συμβολοσειρά μήκους $n + 1$ να εμφανιστεί στην έξοδο της $G$ είναι $1/2^n$ για τις μισές και $0$ για τις υπόλοιπες

- Παρατηρήσεις: ντετερμινιστικός και αποδοτικός (πολυωνυμικός) αλγόριθμος
- Είναι τυχαία η κατανομή; Όχι τελείως!
- αν $p(n) = n + 1$, τότε η ομοιόμορφη κατανομή στο $\{0, 1\}^{n+1}$, έχει χώρο $2^{n+1}$, άρα η πιθανότητα να επιλεγεί μια συμβολοσειρά είναι $1/2^{n+1}$,
- $|dom(G)| = 2^n, |range(G)| = 2^{n+1}$, άρα πιθανότητα μια συμβολοσειρά μήκους $n + 1$ να εμφανιστεί στην έξοδο της $G$ είναι $1/2^n$ για τις μισές και $0$ για τις υπόλοιπες
- Αν ο διαχωριστής είναι εκθετικού χρόνου, τότε με εξαντλητική αναζήτηση μπορεί να ξεχωρίσει την κατανομή $D_n^G$ από την ομοιόμορφη
- Ο σπόρος πρέπει να μείνει μυστικός και αρκετά μεγάλος, ώστε να μη γίνεται brute force επίθεση

# PRG

- Υπάρχουν αποδεδειγμένα ασφαλείς γεννήτριες ψευδοτυχαιότητας; Άγνωστο. Υπάρχουν όμως υποψήφιες.
- Σχετίζεται με την υπόθεση ύπαρξης συναρτήσεων μονής κατεύθυνσης (one-way functions), θα αποδείκνυε ότι $\mathcal{P} \neq \mathcal{NP}$.

# PRG

- Υπάρχουν αποδεδειγμένα ασφαλείς γεννήτριες ψευδοτυχαιότητας; Άγνωστο. Υπάρχουν όμως υποψήφιες.
- Σχετίζεται με την υπόθεση ύπαρξης συναρτήσεων μονής κατεύθυνσης (one-way functions), θα αποδείκνυε ότι $\mathcal{P} \neq \mathcal{NP}$.
- Ισχύει: $G$ γεννήτρια ψευδοτυχαιότητας ανν $G$ μη προβλέψιμη

## Ορισμός

(Προβλέψιμη) Υπάρχει πολυωνυμικός αλγόριθμος A τέτοιος ώστε:

$$Pr[A(G(K)_{1..i}) = G(K)_{i+1}] > \frac{1}{2} + \epsilon$$

για μη αμελητέο $\epsilon$

# PRG

- Υπάρχουν αποδεδειγμένα ασφαλείς γεννήτριες ψευδοτυχαιότητας; Άγνωστο. Υπάρχουν όμως υποψήφιες.
- Σχετίζεται με την υπόθεση ύπαρξης συναρτήσεων μονής κατεύθυνσης (one-way functions), θα αποδείκνυε ότι $\mathcal{P} \neq \mathcal{NP}$.
- Ισχύει: $G$ γεννήτρια ψευδοτυχαιότητας ανν $G$ μη προβλέψιμη

## Ορισμός

(Προβλέψιμη) Υπάρχει πολυωνυμικός αλγόριθμος A τέτοιος ώστε:

$$Pr[A(G(K)_{1..i}) = G(K)_{i+1}] > \frac{1}{2} + \epsilon$$

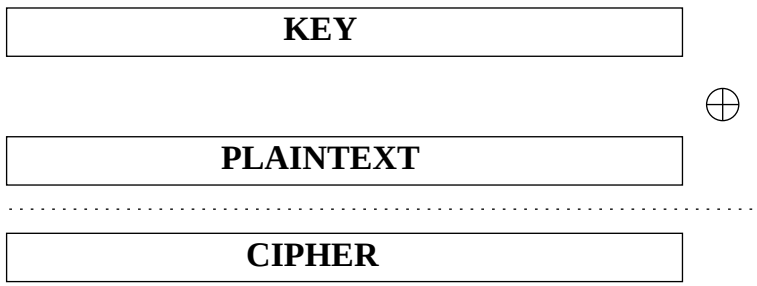
για μη αμελητέο $\epsilon$

- Στη συνέχεια, από την υπόθεση ότι υπάρχουν PRG φτιάχνουμε υπολογιστικά ασφαλή κρυπτοσυστήματα

# OTP

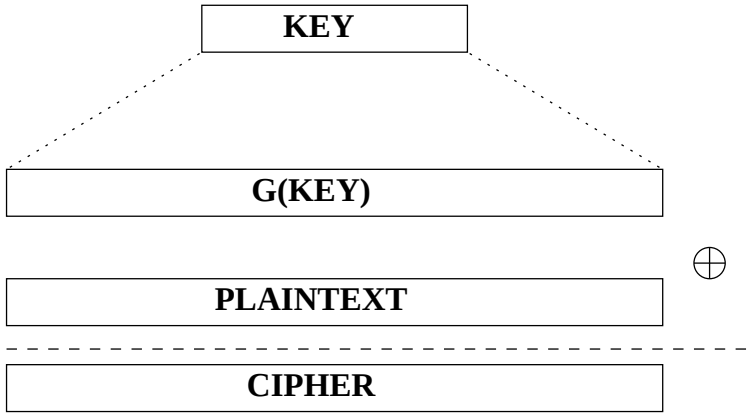
- Ένα καλό σύστημα κρυπτογράφησης: One Time Pad (OTP): $|M| = |K| = |C| = \{0,1\}^n$,

$$Enc_k(m) = k \oplus m, Dec_k(c) = k \oplus c$$




- Το OTP έχει τέλεια μυστικότητα, αλλά πρέπει $|key| \geq |plaintext|$
- Μη ρεαλιστικό!

- Ιδέα: από ένα μικρό, πραγματικά "τυχαίο" σπόρο (seed) φτιάχνω ένα μεγάλο, "ψευδοτυχαίο" κλειδί, έτσι κρυπτογραφώ μεγάλου μεγέθους δεδομένα:




$$c = Enc_k(m) = G(k) \oplus m$$

$$m = Dec_k(c) = G(k) \oplus c$$

- $G$ ντετερμινιστική συνάρτηση πολυωνυμικού χρόνου με $|G(k)| = p(|k|)$
- μοιράζομαι κλειδί $k \leftarrow \{0,1\}^n$
- Απόδειξη ασφάλειας: υπόθεση πως η $G$ είναι ψευδοτυχαία (αναγωγή)

Με βάση τη μη διακρισιμότητα του PRG και το IND-EAV έχουμε:

## Θεώρημα

*Αν $G$ είναι ένας γεννήτορας ψευδοτυχαιότητας, τότε το παραπάνω σχήμα κρυπτογράφησης έχει μη διακρίσιμες κρυπτογραφήσεις στο μοντέλο παθητικού αντιπάλου (IND-EAV).*

## Απόδειξη.

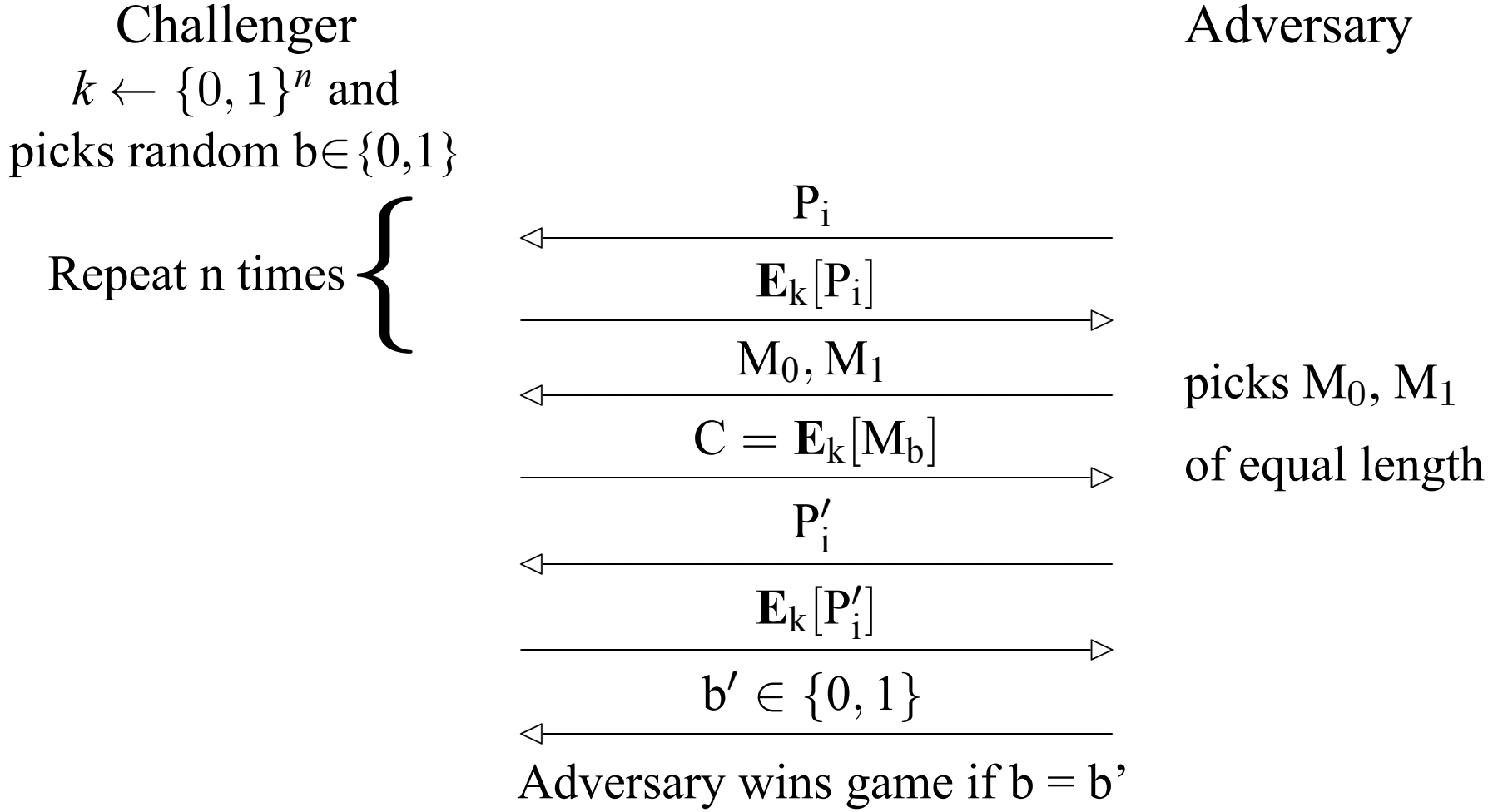
(Ιδέα) Με αναγωγή: Απόδειξη βασισμένη στην υπόθεση της ασφάλειας του γεννήτορα. Υποθέτουμε ότι έχουμε αντίπαλο $\mathcal{A}$ ο οποίος διακρίνει τις κρυπτογραφήσεις. Χρησιμοποιώντας τον $\mathcal{A}$ ως μαντείο μπορούμε να διακρίνουμε την έξοδο του $G$ από μία πραγματικά τυχαία. Άτοπο.

□

- Τι γίνεται όταν έχουμε πολλά μηνύματα;
- Δουλεύει το παραπάνω σχήμα;
- Θα ορίσουμε ένα άλλο είδος επίθεσης
- Ελάχιστο επίπεδο ασφάλειας

# CPA

- Για σταθερά $\Pi, \mathcal{A}$
- Ορίζουμε το τυχαίο πείραμα $PrivCPA_{\mathcal{A},\Pi}(n)$:



# CPA

## Ορισμός

Το Π είναι CPA-ασφαλές αν για όλους τους PPT αντιπάλους $\mathcal{A}$, υπάρχει μια αμελητέα συνάρτηση $\epsilon$ ώστε

$$Pr[PrivCPA_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \epsilon(n)$$

- Επίθεση: $c_0 = Enc_k(m_0), c_1 = Enc_k(m_1)$

- Επίθεση: $c_0 = Enc_k(m_0), c_1 = Enc_k(m_1)$
- Έλεγξε αν $c = c_0$ ή $c = c_1$

- Επίθεση: $c_0 = Enc_k(m_0), c_1 = Enc_k(m_1)$
- Έλεγξε αν $c = c_0$ ή $c = c_1$
- Πιθανότητα επιτυχίας 1

- Επίθεση: $c_0 = Enc_k(m_0), c_1 = Enc_k(m_1)$
- Έλεγξε αν $c = c_0$ ή $c = c_1$
- Πιθανότητα επιτυχίας 1
- Λύση: Πιθανοτική κρυπτογράφηση!

# Πιθανοτική κρυπτογράφηση

Μη ασφαλές για πολλαπλά μηνύματα: Two Time Pad

Ανάγκη για πιθανοτική κρυπτογράφηση

## Θεώρημα

*Έστω* $\Pi = (Gen, Enc, Dec)$ *ένα σχήμα κρυπτογράφησης όπου* $Enc$ *είναι ντετερμινιστικό. Τότε το* $\Pi$ *δεν έχει μη διακρίσιμες πολλαπλές κρυπτογραφήσεις στο μοντέλο παθητικού αντιπάλου.*

## Απόδειξη.

$\mathcal{A}$ στέλνει τα $\vec{M}_0 = (0^n, 0^n)$ και $\vec{M}_1 = (0^n, 1^n)$ και παίρνει $\vec{C} = (c^1, c^2)$ □

# PRF

- Συνάρτηση που φαίνεται ίδια με μια τυχαία συνάρτηση
- Τυχαία συνάρτηση: $Func_n$ = όλες οι συναρτήσεις από το $\{0,1\}^n$ στο $\{0,1\}^n$
- Πόσες; Μπορούμε να αναπαραστήσουμε μια συνάρτηση στο $Func_n$ με $n2^n$ bits
- Άρα, $|Func_n| = 2^{n2^n}$
- Τυχαία συνάρτηση: διάλεξε ομοιόμορφα μια $f \in Func_n$
- Ισοδύναμα: σε κάθε θέση του πίνακα τιμών διάλεξε ομοιόμορφα ένα string από το $\{0,1\}^n$

- Δεν έχει νόημα να μιλάμε για σταθερή συνάρτηση, αλλά θέλουμε κάποια κατανομή
- Αν έχουμε μια $F\colon \{0,1\}^* \times \{0,1\}^* \mapsto \{0,1\}^*$, τότε αν κρατήσουμε σταθερή την πρώτη παράμετρο έχουμε συναρτήσεις $F_k(x) = F(k,x)$, όπου $k$ κλειδί (επιλέγεται ομοιόμορφα)
- Επιλέγοντας το κλειδί $k \leftarrow \{0,1\}^n$ επιλέγεται μια $F_k\colon \{0,1\}^n \to \{0,1\}^n$
- Άρα η $F$ ορίζει μια κατανομή στις συναρτήσεις στην $Func_n$

# PRF

## Ορισμός

Η $F$ είναι ψευδοτυχαία συνάρτηση αν για κάθε πολυωνυμικού χρόνο αλγόριθμο $D$ έχουμε:

$$|Pr_{k \leftarrow \{0,1\}^n}[D^{F_k()} = 1] - Pr_{f \leftarrow Func_n}[D^{f()} = 1]| \leq \epsilon(n)$$

# CPA-secure scheme

- Έστω $F$ μια PRF
- $k \leftarrow \{0,1\}^n$
- $\text{Enc}_k(m)$: $r \leftarrow \{0,1\}^n$
  Cipher: $< r, F_k(r) \oplus m >$
- $\text{Dec}_k(< c_1, c_2 >)$: $c_2 \oplus F_k(c_1)$

# PRF vs PRG

PRF πιο ισχυρή από PRG

Μπορούμε από μια συνάρτηση να πάρουμε γεννήτρια: $G(k) = F_k(0)|F_k(1)$

pseudorandom permutation: bijection, inverse

# Δημιουργία πραγματικής τυχαιότητας

- υλικό, φυσικά φαινόμενα π.χ. θερμικός ή ηλεκτρικός θόρυβος
- λογισμικό π.χ. πάτημα πλήκτρων πληκτρολογίου, κίνηση του ποντικιού

Γενικού σκοπού γεννήτριες τυχαίων αριθμών, μη κατάλληλες για κρυπτογραφία π.χ. random() της C.

# ‘Αποδεδειγμένα’ ασφαλείς γεννήτριες ψευδοτυχαίων

- RSA-based, BBS.
- Βασίζονται σε (γενικά παραδεκτές) αριθμοθεωρητικές μονόδρομες συναρτήσεις: ύψωση σε δύναμη modulo $n$, τετραγωνισμός modulo $n$.
- Λειτουργία: διαδοχικές εφαρμογές της συνάρτησης, έξοδος κάθε φορά το λιγότερο σημαντικό bit του αριθμού (ή κάποια από τα λιγότερο σημαντικά bit).
- Είναι ασφαλείς κάτω από την υπόθεση δυσκολίας αντιστροφής της αντίστοιχης συνάρτησης.
- Απαιτούν μεγαλύτερη υπολογιστική προσπάθεια.

# Blum-Blum-Shub (1986)

**Αλγόριθμος**

- Πάρε δύο μεγάλους πρώτους $p, q$, με $p \equiv q \equiv 3 \pmod 4$, και θέσε $n = pq$.
- Επίλεξε τυχαία ένα $s_0$ σχετικά πρώτο με το $n$.
- Πάρε

$$z_0 = s_0^2 \bmod n$$

Για $1 \leq i \leq \infty$

$$z_i = (z_{i-1}^2 \bmod n) \bmod 2$$

Παρατήρηση: σχετικά αργό, αλλά ασφαλές με την υπόθεση ότι ο έλεγχος τετραγωνικών υπολοίπων mod$n$ είναι δύσκολος αν δεν είναι γνωστή η παραγοντοποίηση του $n$.

# Παράδειγμα BBS

Έστω $n = 192649 = 383 * 503$ και $s_0 = 101355^2 \mod n = 20749$.

Τα πρώτα 5 bits που παράγονται από τον BBS είναι

$$11001$$

και προκύπτουν:

| $i$ | $s_i$ | $z_i$ |
|---|---|---|
| 0 | 20749 | |
| 1 | 143135 | 1 |
| 2 | 177671 | 1 |
| 3 | 97048 | 0 |
| 4 | 89992 | 0 |
| 5 | 174051 | 1 |

# Κρυπτοσυστήματα ροής (stream ciphers)

- Η ιδέα της κρυπτογράφησης και αποκρυπτογράφησης ανά byte (ή bit) με χρήση της πράξης $\oplus$
- Πιο γρήγορη από block ciphers, αλλά το ίδιο ασφαλής
- Χρήση στο internet (επικοινωνία μεταξύ server και web browser)

# Κρυπτοσυστήματα ροής (stream ciphers)

Παραγωγή ακολουθίας κλειδιών με βάση κάποιο αρχικό κλειδί, και (πιθανά) το plaintext.

## Ορισμός

- Plaintext: $x_0, x_1, \ldots, x_{n-1}$
- Ciphertext: $y_0, y_1, \ldots, y_{n-1}$
- Αρχικό κλειδί: $k$
- Βοηθητικές συναρτήσεις: $f_i, 0 \leq i < m$
- Key stream: $z_i = f_{i \bmod m}(k, x_0, \ldots, x_{i-1}, z_0, \ldots, z_{i-1})$
- Κρυπτογράφηση: $y_i = enc_{z_i}(x_i)$
- Αποκρυπτογράφηση: $x_i = dec_{z_i}(y_i)$

Π.χ. για δυαδικές ακολουθίες:

$$enc_z(x) = x \oplus z = x + z \bmod 2$$
$$dec_z(y) = y \oplus z = y + z \bmod 2$$

# Κρυπτοσυστήματα ροής (stream ciphers)

Διακρίνονται σε synchronous (το κλειδί δεν εξαρτάται από το plaintext), και asynchronous (λέγονται και self-synchronizing).

Επίσης σε periodic ($\forall i : z_{i+d} = z_i$, όπου $d$ η περίοδος) και aperiodic.

Παράδειγμα: το Vigenère είναι synchronous και periodic.

# Η γεννήτρια ψευδοτυχαίων RC4

- Συστατικά: 2 arrays of bytes:
  - Μετάθεση $P[0..255]$. Αρχικοποίηση:
    **for all** $i \in \{0..255\}$ **do** : $P[i] = i$
  - Κλειδί $K[0..keylen-1]$, $keylen \leq 256$ – συνήθως $keylen \in [5..8]$.
    Επιλέγεται από χρήστη.
- Δημιουργία σειράς κλειδιών (key-scheduling algorithm – KSA). Η αρχική (ταυτοτική) μετάθεση $P$ μετατρέπεται μέσω μιας σειράς ανταλλαγών (swap) σε μια (φαινομενικά τυχαία) μετάθεση.
  Το "ανακάτεμα" επηρεάζεται από το αρχικό κλειδί $K$.
- Παραγωγή ψευδοτυχαίων bytes (pseudorandom generation algorithm – PRGA)
  Επαναληπτικός βρόχος. Σε κάθε επανάληψη επιλέγεται κάποιο byte της $P$ ως κλειδί εξόδου με τρόπο που καθορίζεται από τα τρέχοντα περιεχόμενα της $P$.
  Οι επαναλήψεις συνεχίζονται για όσο χρειάζεται (δηλ. μέχρι να τελειώσει το stream). Σε κάθε επανάληψη γίνεται και ένα νέο swap.

# Η γεννήτρια ψευδοτυχαίων RC4

## Περιγραφή KSA, PRGA

- Δημιουργία σειράς κλειδιών (KSA)

  $j = 0$

  **for** $i = 0$ **to** $255$ **do** :

  $j = (j + P[i] + K[i \bmod keylen]) \bmod 256$

  swap$(P[i], P[j])$

- Παραγωγή ψευδοτυχαίων bytes (PRGA)

  $i = 0; j = 0$

  **while** next key needed :

  $i = (i + 1) \bmod 256 \; ; j = (j + P[i]) \bmod 256$

  swap$(P[i], P[j])$

  $K_o = P[(P[i] + P[j]) \bmod 256]$

  output $K_o$

Κάθε κλειδί εξόδου $K_o$ χρησιμοποιείται για την κρυπτογράφηση ενός byte αρχικού κειμένου.

# Η γεννήτρια ψευδοτυχαίων RC4

## Παρατηρήσεις

- Με ίδιο αρχικό κλειδί $K$ προκύπτει η ίδια σειρά κλειδιών εξόδου.
- Απλή και γρήγορη στην υλοποίηση με software (σε αντίθεση με άλλα stream cipher, π.χ. αυτά που βασίζονται σε LFSRs).
- Χρήση σε πολύ διαδεδομένα πρωτόκολλα: TSL, WEP, WPA.
- Η ασφάλεια της γεννήτριας RC4 έχει αμφισβητηθεί έντονα. Κάποιοι τρόποι χρήσης ιδιαίτερα ανασφαλείς (π.χ. WEP) – επίθεση Fluhrer, Mantin, Shamir (2001).
- Άμυνα: απόρριψη αρχικού τμήματος κλειδοροής (RC4-drop[$n$]), ενδεικτικά: $n = 768$ bytes, συστήνεται ακόμη και $n = 3072$.

# Η γεννήτρια ψευδοτυχαίων RC4

## Παρατηρήσεις

- Με ίδιο αρχικό κλειδί $K$ προκύπτει η ίδια σειρά κλειδιών εξόδου.
- Απλή και γρήγορη στην υλοποίηση με software (σε αντίθεση με άλλα stream cipher, π.χ. αυτά που βασίζονται σε LFSRs).
- Χρήση σε πολύ διαδεδομένα πρωτόκολλα: TSL, WEP, WPA.
- Η ασφάλεια της γεννήτριας RC4 έχει αμφισβητηθεί έντονα. Κάποιοι τρόποι χρήσης ιδιαίτερα ανασφαλείς (π.χ. WEP) – επίθεση Fluhrer, Mantin, Shamir (2001).
- Άμυνα: απόρριψη αρχικού τμήματος κλειδοροής (RC4-drop[$n$]), ενδεικτικά: $n = 768$ bytes, συστήνεται ακόμη και $n = 3072$.
- Μη ασφαλές!

# Κρυπτοσυστήματα ροής: Linear Recurrence Keystream

Αρχικό διάνυσμα κλειδιών: $(z_0, z_1, \ldots, z_{m-1})$.
Τα υπόλοιπα κλειδιά υπολογίζονται ως εξής:

$$z_{i+m} = \sum_{j=0}^{m-1} c_j \cdot z_{i+j} \pmod 2, \quad \forall j, c_j \in \{0, 1\}$$

Εάν το πολυώνυμο $c_0 + c_1 x + c_2 x^2 + \ldots + c_{m-1} x^{m-1} + x^m$ είναι primitive, τότε το κρυπτοσύστημα έχει περίοδο $d = 2^m - 1$.
Π.χ. $c_0 = c_1 = 1, c_2 = c_3 = 0$ ορίζουν το πολυώνυμο $x^4 + x + 1$, και με δεδομένο αρχικό κλειδί $z_0, \ldots, z_3$ έχουμε $z_{4+i} = z_i + z_{i+1} \bmod 2$.

Το κρυπτοσύστημα αυτό έχει περίοδο 15.

Υλοποίηση με Linear Feedback Shift Register (LFSR).

# Καταχωρητές Ολίσθησης Γραμμικής Ανάδρασης - LFSRs

- Δημιουργούν περιοδικές ακολουθίες, με περίοδο το πολύ $2^L - 1$, $L$ το πλήθος των ψηφίων.
- Αν το αντίστοιχο πολυώνυμο είναι primitive έχουμε maximum-length LFSR. Πολλά γνωστά primitive πολυώνυμα.

# Καταχωρητές Ολίσθησης Γραμμικής Ανάδρασης - LFSRs

- Δημιουργούν περιοδικές ακολουθίες, με περίοδο το πολύ $2^L - 1$, $L$ το πλήθος των ψηφίων.
- Αν το αντίστοιχο πολυώνυμο είναι primitive έχουμε maximum-length LFSR. Πολλά γνωστά primitive πολυώνυμα.
- Σημαντικό μέγεθος για ακολουθίες: γραμμική πολυπλοκότητα (linear complexity). Είναι το ελάχιστο μέγεθος LFSR που παράγει την ίδια ακολουθία.
- Αλγόριθμος Berlekamp-Massey: υπολογίζει τη γραμμική πολυπλοκότητα και τον αντίστοιχο LFSR.

# Καταχωρητές Ολίσθησης Γραμμικής Ανάδρασης - LFSRs

- Δημιουργούν περιοδικές ακολουθίες, με περίοδο το πολύ $2^L - 1$, $L$ το πλήθος των ψηφίων.
- Αν το αντίστοιχο πολυώνυμο είναι primitive έχουμε maximum-length LFSR. Πολλά γνωστά primitive πολυώνυμα.
- Σημαντικό μέγεθος για ακολουθίες: γραμμική πολυπλοκότητα (linear complexity). Είναι το ελάχιστο μέγεθος LFSR που παράγει την ίδια ακολουθία.
- Αλγόριθμος Berlekamp-Massey: υπολογίζει τη γραμμική πολυπλοκότητα και τον αντίστοιχο LFSR.
- Αύξηση γραμμικής πολυπλοκότητας: χρήση περισσότερων LFSRs, συνδυασμός εξόδων με μη γραμμικό τρόπο.
  Π.χ. Geffe generator συνδυάζει 3 maximum-length LFSRs με μήκος $L_1, L_2, L_3$ και εξόδους $x_1, x_2, x_3$:
  $$f(x_1, x_2, x_3) = x_1x_2 \oplus (1 \oplus x_2)x_3$$
  έχει περίοδο $(2^{L_1} - 1) \cdot (2^{L_2} - 1) \cdot (2^{L_3} - 1)$ και γραμμική πολυπλοκότητα $L = L_1L_2 + L_2L_3 + L_3$

# Κρυπτοσυστήματα ροής με LFSRs

LFSR (linear feedback shift register): εύκολο στο hardware, κακό γιατί είναι γραμμικό

Χρήση:

1. DVD κρυπτογράφηση (CSS): 2 LFSRs
2. GSM (A5/1,2): 3 LFSRs
3. Bluetooth (E0): 4 LFSRs

# Σύγχρονα κρυπτοσυστήματα ροής

Σύγχρονα κρυπτοσυστήματα ροής:

$$G : \{0,1\}^s \times R \to \{0,1\}^n$$

όπου το $R$: nonce, δεν επαναλαμβάνεται για το ίδιο κλειδί

$$Enc_k(m, k; r) = m \oplus G(k, r)$$

Ιδέα: επαναχρησιμοποίηση ίδιου κλειδιού $k$ καθώς το $(k, r)$ αλλάζει

- eSTREAM project (2004-08): Salsa20, Sosemanuk, Trivium