CRIPTOGRAFIE ȘI SECURITATE

Conf.univ.dr. Radu Boriga

GENERATOARE DE NUMERE PSEUDO-ALEATOARE (PRNG)

- Un **generator de numere pseudo-aleatoare** este un sistem $P = (S, s_0, T, U, G)$, în care:
  - S este o mulțime nevidă numită *mulțimea stărilor*;
  - s_0 este o stare numită *stare inițială*;
  - $T: S \to S$ este o funcție numită *funcție de tranziție*;
  - U este o mulțime nevidă numită *mulțimea ieșirilor*;
  - $G: S \to U$ este o funcție numită *funcție de ieșire*.

- **Perioada unui generator de numere pseudo-aleatoare** este cel mai mic număr natural k cu proprietatea că starea S_n este identică cu starea S_{n+k}.

- Valorile generate de către un PRNG trebuie să satisfacă următoarele cerințe:
  - să fie cât mai puțin corelate între ele
  - să fie uniform distribuite
  - să aibă o perioadă cât mai mare
  - să aibă o rezoluție cât mai mare
  - să fie rapid

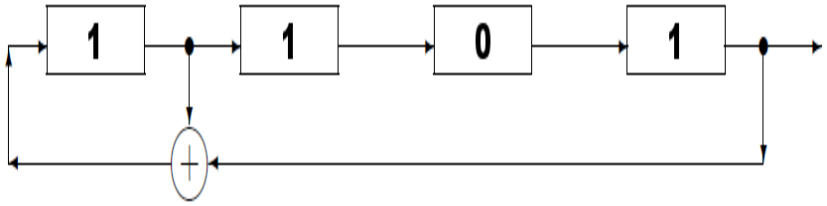
➢ Generatorul congruențial liniar (LCG): $x_{n+1} = (ax_n + b) \bmod m$

- Perioada maximă a unui LCG este m și se atinge dacă și numai dacă sunt îndeplinite simultan următoarele condiții:
  1. $\text{cmmdc}(b, m) = 1$
  2. $a - 1$ este multiplu de p pentru orice număr prim p care-l divide pe m
  3. dacă m este multiplu de 4, atunci și $a - 1$ este multiplu de 4

PRNG	m	a	b	Biții de ieșire
Borland C/C++	2^{32}	22695477	1	30…16
glibc (GCC)	2^{31}	1103515245	12345	30…0
ANSI C	2^{31}	1103515245	12345	30…16
C99, C11	2^{32}	1103515245	12345	30…16
Microsoft Visual C++	2^{32}	214013	2531011	30…16
Microsoft Visual Basic	2^{24}	1140671485	12820163	
Apple CarbonLib	$2^{31} - 1$	16807	0	
C++11	$2^{31} - 1$	48271	0	
MMIX (Donald Knuth)	2^{64}	6364136223846793005	1442695040888963407	
java.util.Random	2^{48}	25214903917	11	47...16
RANDU	2^{31}	65539	0	

➢ Registru de deplasare cu feedback liniar (LFSR)

- Este un circuit liniar format dintr-un registru serial şi o funcţie de feedback. Funcţia de feedback constă într-o adunare $modulo\ 2$ a anumitor biţi din registru.



$$f(x) = 1 + X + X^4$$

- Perioada maximă a unui LFSR cu n biţi este $2^n - 1$ şi se obţine dacă polinomul $f \in \mathbb{Z}_2[X]$ ataşat LFSR-ului respectiv este primitiv.

➢ Generatorul Blum-Blum-Shub (BBS): $\boldsymbol{x_{n+1} = x_n^2\ mod\ m}$

- Modulul m trebuie să fie de forma $m = p * q$, unde p şi q sunt numere prime cu proprietatea că $p, q \equiv 3\ (mod\ 4)$.
- Valoarea iniţială x_0 trebuie să îndeplinească următoarele condiţii: $x_0 > 1$, $p \nmid x_0$ şi $q \nmid x_0$.
- Funcţia de ieşire este, de obicei, bitul de paritate al valorii curente.

➢ Generatoare de tip Lagged Fibonacci (LFG):

- Sunt definite printr-o relaţie recurentă de forma:

$$x_n = (x_{n-j} \star x_{n-k}) \, mod \, m$$

în care $0 < j < k$, $m = 2^L$ (L este lungimea cuvântului de memorie), iar $\star$ este unul dintre operatorii binari de adunare, scădere, înmulţire sau XOR.

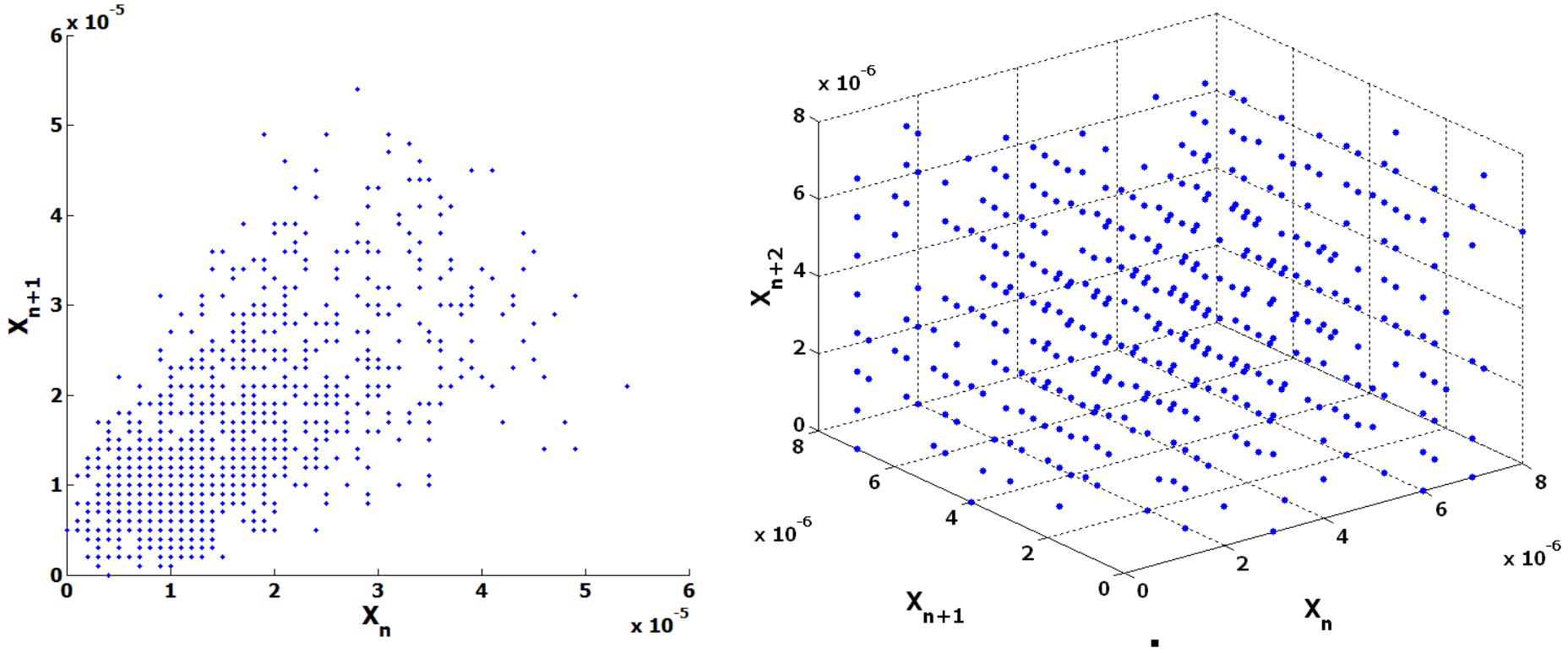
- Generatoarele de acest tip au o perioadă maximă egală cu $(2^k - 1) * 2^{L-1}$ dacă se foloseşte operaţia de adunare sau cea de scădere, $(2^k - 1) * k$ dacă se foloseşte operaţia XOR şi, respectiv, $(2^k - 1) * 2^{L-3}$ dacă se foloseşte operaţia de înmulţire și parametrii de deplasare j şi k sunt puterile unui trinom primitiv $x^k + x^j + 1$.

- Parametrii buni:

j	k
7	170
24	55
65	71
128	159

j	k
31	63
97	127
353	521
168	521

j	k
334	607
273	607
418	1279
1029	2281



Distribuţia a 1000 de valori generate folosind un LFG aditiv ($j = 237$ şi $k = 607$)

➢ Clasa de generatoare RANROT

- **Tipul A:** $x_n = \left(\left(x_{n-j} + x_{n-k} \right) mod\ 2^b \right) \gg r$
- **Tipul B:** $x_n = \left(\left(x_{n-j} \gg r_1 \right) + \left(x_{n-k} \gg r_2 \right) \right)\ mod\ 2^b$
- **Tipul B3:** $x_n = \left(\left(x_{n-i} \gg r_1 \right) + \left(x_{n-j} \gg r_2 \right) + \left(x_{n-k} \gg r_3 \right) \right)\ mod\ 2^b$
- **Tipul W:** $y_n = \left(\left(z_{n-j} \gg r_4 \right) + \left(z_{n-k} \gg r_2 \right) \right)\ mod\ 2^{\frac{b}{2}}$

  $z_n = \left(\left(y_{n-j} \gg r_3 \right) + \left(y_{n-k} \gg r_1 \right) \right) mod\ 2^{\frac{b}{2}}$

  $x_n = y_n + z_n 2^{\frac{b}{2}}$

- Este necesar ca $0 < i < j < k$, iar x_n, y_n și z_n trebuie să fie întregi fără semn
- Perioadele maximale ale acestor generatoare sunt 2^{kb}.

➢ Generatorul Mother-of-All:

$$S_n = 2111111111 * x_{n-4} + 1492 * x_{n-3} + 1776 * x_{n-2} + 5115 * x_{n-1} + \left\lfloor \frac{S_{n-1}}{2^{32}} \right\rfloor$$

$$x_n = S_n\ mod\ 2^{32}$$

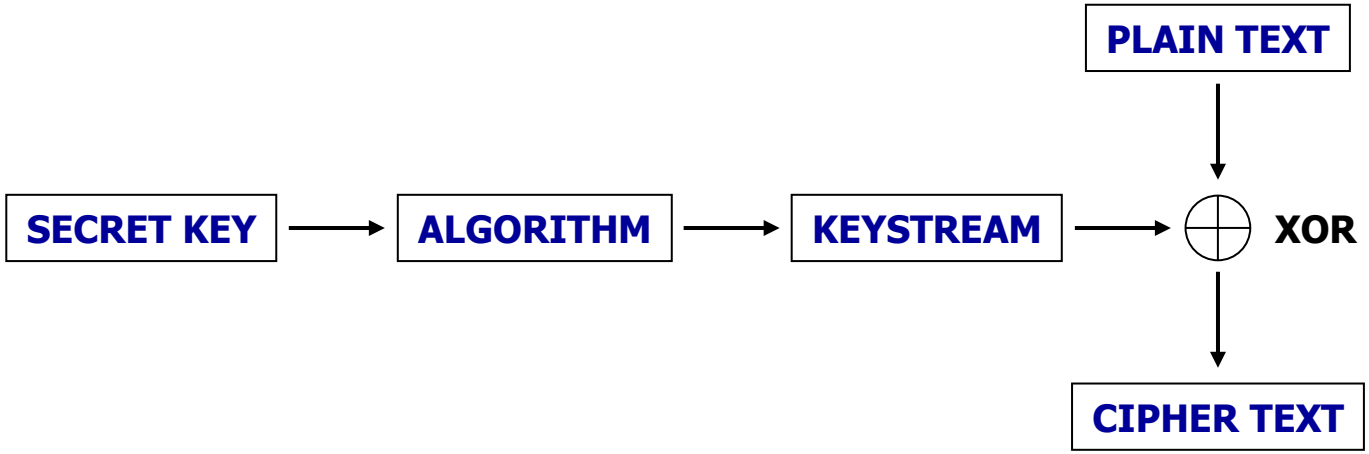
- Perioada acestui generator este aproximativ 2^{250}.

➢ Generatorul RANMAR

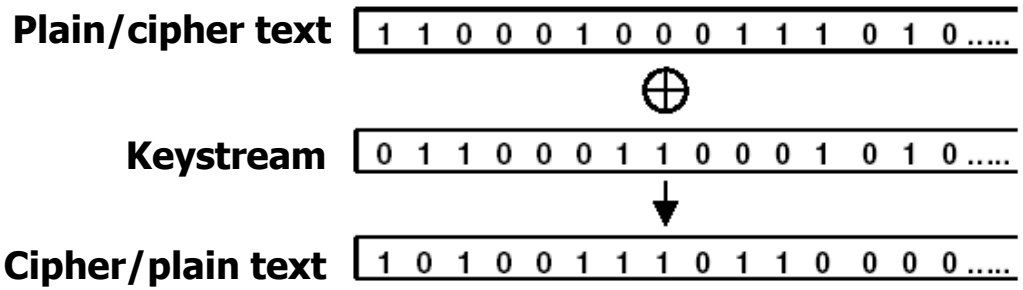
$$x_n = (y_n - c_n + 2^{24}) \; mod \; 2^{24}$$
$$y_n = (y_{n-97} - y_{n-33} + 2^{24}) \; mod \; 2^{24}$$
$$c_n = (c_{n-1} - 7654321 + 2^{24} - 3) \; mod \; (2^{24} - 3)$$

- Perioada acestui generator este aproximativ 2^{144}.

- Criptarea/decriptarea se realizează la nivel de octet, prin XOR-are cu octeții unei chei fluide.
- Cheia fluidă se obține dintr-o cheie secretă (de dimensiune mică) folosind un algoritm determinist.



- Cifrul Vernam este considerat singurul sistem de criptare complet sigur!



- **Criptare:**

$$C_i = P_i \oplus K_i$$

- **Decriptare:**

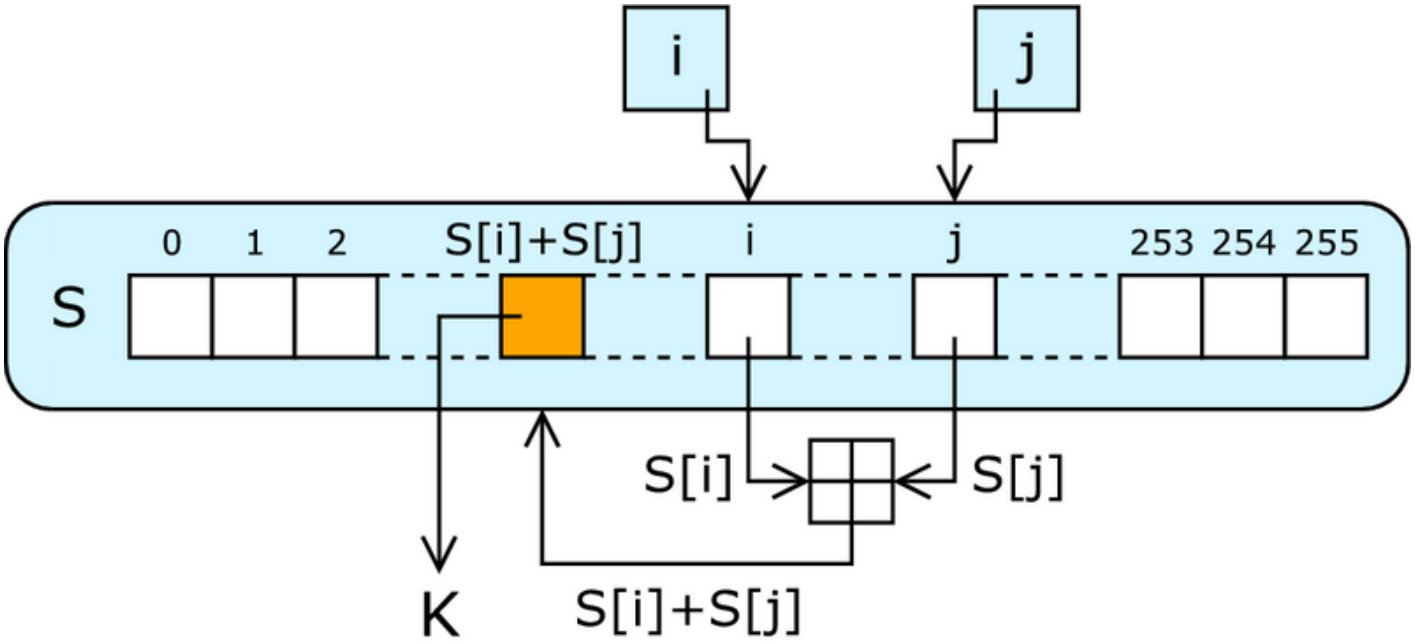
$$P_i = C_i \oplus K_i$$

- **RC4** este un cifru fluid creat de **Ron Rivest** (RSA Security) în 1987.
- Atuurile sale sunt simplitatea și viteza foarte bună (200-300 MB/s).
- Cheia secretă K poate să aibă o lungime L cuprinsă între 1 și 32 de octeți, dar, de obicei, lungimea sa este cuprinsă între 5 și 16 octeți.
- Utilizează o stare internă formată din:
  - o permutare S de lungime 256 (formată din valorile $0,1,\dots,255$)
  - doi indici $i, j \in \{0,1,\dots,255\}$
- Algoritmul constă din două etape:
  - inițializarea cheii fluide (*Key-scheduling algorithm* - KSA)
  - generarea cheii fluide (*Pseudo-random generation algorithm* - PRGA)
- Criptarea/decriptarea se realizează prin XOR-area octetului curent din mesajul clar M format din N octeți cu octetul aleator curent R.

Key-scheduling algorithm (KSA)	Pseudo-random generation algorithm (PRGA)

for i **from** 0 **to** 255
  S[i] := i
endfor
 	
j := 0
for i **from** 0 **to** 255
  j := (j + S[i] + K[i%L])%256
  swap(S[i],S[j])
endfor
	i := 0
	j := 0
	for k **from** 0 **to** N-1
	  i := (i + 1) % 256
	  j := (j + S[i]) % 256
	  swap(S[i],S[j])
	  R := S[(S[i] + S[j]) % 256]
	  E[i] := M[i] xor R
	endfor

Exemplu de utilizare pas cu pas: https://www.youtube.com/watch?v=KM-xZYZXElk



- RC4 a fost utilizat în mai multe protocoale: WEP (1997), WPA (2003/2004), SSL (1995) și TLS (1999).
- A. Roos (1995), S. Fluhrer, I. Mantin și A. Shamir (2001), A. Klein (2005) au demonstrat teoretic faptul că RC4 are un nivel de securitate scăzut, iar atacul NOMORE (2015) a demonstrat și practic acest fapt.
- RC4 a fost eliminat din TLS în anul 2015 (RFC 7465).