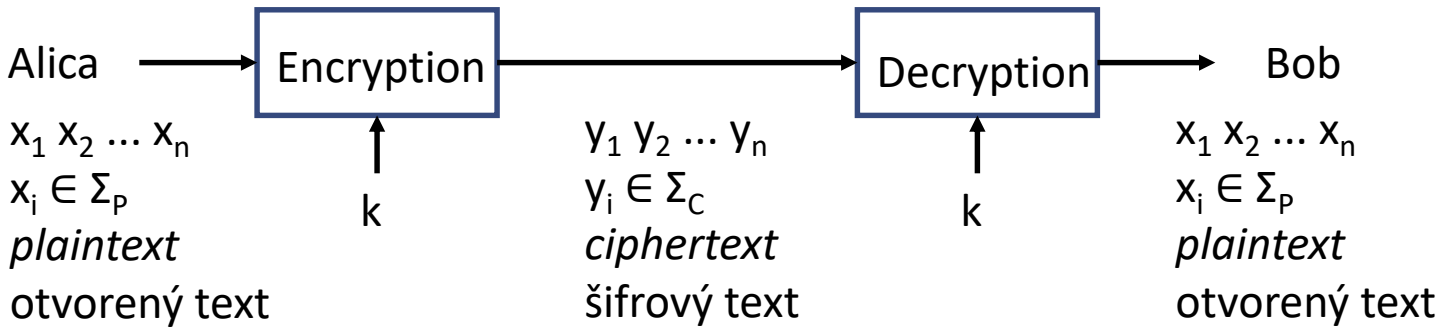
Kryptografické systémy
Prúdové šifry

doc. RNDr. Jozef Jirásek, PhD.

ZS 2023

Klasické symetrické šifrovacie systémy



- posuvná šifra
- monoalfabetická (jednoduchá) substitúcia (*simple substitution*)
- bigramová šifra (Playfair, Hill)
- polyalfabetické šifry

Polyalfabetické šifry – možnosti COA

posuvná šifra

PT	K	R	Y	P	T	O	G	R	A	F	I	A	N	A	U	P	J	S	J	E	S	U	P	E	R
K	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U
CT	E	L	S	J	N	I	A	L	U	Z	C	U	H	U	O	J	D	M	D	Y	M	O	J	Y	L

vzdialenosť jednoznačnosti (koľko znakov CT potrebujem, aby existoval štatisticky jediný kľúč, ktorý po dešifrovaní dá rozumný PT v použitom jazyku)

$$n_0 \geq H(K)/(H(L) - H_{EN}(L)) =$$
$$= \log_2 26/(\log_2 26 - 1{,}5) = 4{,}7/3{,}2 = 1{,}47$$

na kryptoanalýzu COA potrebujem (a štatisticky stačí) pre PT v angličtine aspoň dva znaky CT

ak budem mať len jeden znak ?

Polyalfabetické šifry – možnosti COA

Vigenèr

PT	K	R	Y	P	T	O	G	R	A	F	I	A	N	A	U	P	J	S	J	E	S	U	P	E	R
K	U	P	J	S	U	P	J	S	U	P	J	S	U	P	J	S	U	P	J	S	U	P	J	S	U
CT	E	G	H	H	N	D	P	J	U	U	R	S	H	P	D	H	D	H	S	W	M	J	Y	W	L

pre kľúč dĺžky m (náhodný)

vzdialenosť jednoznačnosti $n_0 \geq H(K)/(H(L) - H_{EN}(L)) =$
$= \log_2(26)^m/(\log_2 26 - 1{,}5) = m \cdot 4{,}7/3{,}2 = 1{,}47m$

na kryptoanlýzu COA pre náhodný kľúč dĺžky 4 potrebujem teoreticky aspoň 6 znakov CT

ak budem mať 5, dostanem viac relevantných riešení

ak budem mať len 4 znaky CT ...

Vernamova šifra

Vernam (1917)

PT	K	R	Y	P	T	O	G	R	A	F	I	A	N	A	U	P	J	S	J	E	S	U	P	E	R
K	U	P	J	S	R	W	Y	B	D	A	G	X	D	U	N	Y	I	V	Z	P	B	S	C	J	L
CT	E	G	H	H	K	K	E	S	D	F	O	X	Q	U	H	N	R	N	I	T	T	M	R	N	C

pre náhodný kľúč rovnakej dĺžky ako správa

k CT existuje pre každý PT taký K, že $D_K(CT) = PT$ **COA nie je možný**

CT	E	G	H	H	K	K	E	S	D	F	O	X	Q	U	H	N	R	N	I	T	T	M	R	N	C
K	U	P	J	S	R	W	Y	B	D	A	G	X	D	U	N	Y	I	V	Z	P	B	S	C	J	L
PT	K	R	Y	P	T	O	G	R	A	F	I	A	N	A	U	P	J	S	J	E	S	U	P	E	R

CT	E	G	H	H	K	K	E	S	D	F	O	X	Q	U	H	N	R	N	I	T	T	M	R	N	C
K	B	S	Y	E	C	P	K	Z	P	O	A	N	C	C	O	S	A	U	V	T	B	T	T	W	U
PT	D	O	J	D	I	V	U	T	O	R	O	K	O	S	T	V	R	T	N	A	S	T	Y	R	I

ak použijem ako kľúč EN text s $H(P) < 3{,}2$ tak by to teoreticky bolo možné

Miera bezpečnosti

bezpodmienečná bezpečnosť (*unconditional security*)

perfect secrecy

$\forall x \in P \;\; \forall y \in C : \;\; p(x|y) = p(x)$
pravdepodobnosť uhádnutia otvoreného textu ak poznáme šifrovaný text je rovnaká, ako keby sme ho nepoznali

nemôže byť prelomený ani nekonečne veľkou výpočtovou silou

Shannon: šifrovanie je bezpodmienečne bezpečné práve ak $\forall y \in C : \;\; p(y|x) = p(y) > 0$ ($\forall y \; \exists k$, že $E_k(x)=y$), každý kľúč je použitý s rovnakou pravdepodobnosťou a $|K|=|C|=|P|$

Miera bezpečnosti

dokázateľná bezpečnosť (*provable security*)

- problém nájdenia kľúča je možné previesť na problém s dostatočnou zložitosťou, ktorý zatiaľ nevieme vyriešiť

výpočtová bezpečnosť (*computational security*)

- známe postupy na hľadanie kľúča nie sú v rozumnom čase známymi prostriedkami (konštrukčnými, energetickými, finančnými) uskutočniteľné

OTP – One-time pad

jediná známa nerozlúštiteľná (*perfect secrecy*) šifra

pre binárne reťazce $x = x_1\, x_2 \ldots x_n \in \{0,1\}^*$ $s = s_1\, s_2 \ldots s_n \in \{0,1\}^*$

$E(x) = y = y_1\, y_2 \ldots y_n = (x_1 \text{ xor } s_1)\,(x_2 \text{ xor } s_2) \ldots (x_n \text{ xor } s_n)$

$D(y) = (y_1 \text{ xor } s_1)\,(y_2 \text{ xor } s_2) \ldots (y_n \text{ xor } s_n) = (x_1 \text{ xor } s_1 \text{ xor } s_1)$
$(x_2 \text{ xor } s_2 \text{ xor } s_2) \ldots (x_n \text{ xor } s_n \text{ xor } s_n) = x_1\, x_2 \ldots x_n = x$

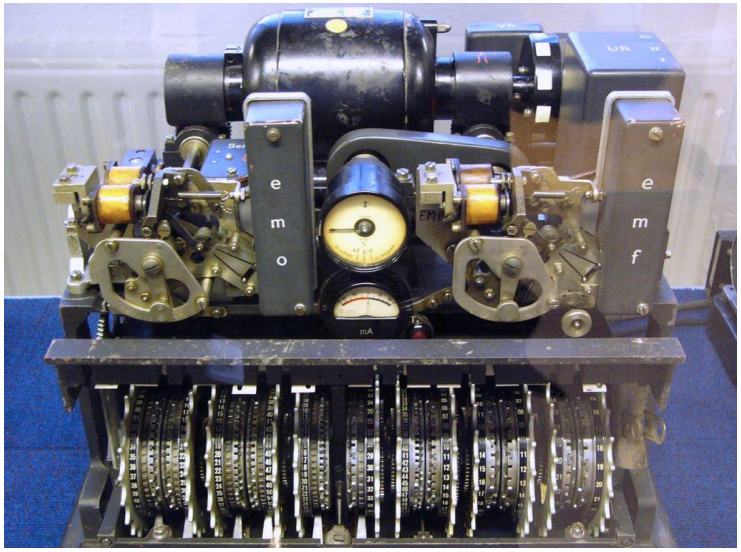
$$a \text{ xor } b = a \oplus b = (a + b) \bmod 2$$

reťazec s (*stream*) musí byť

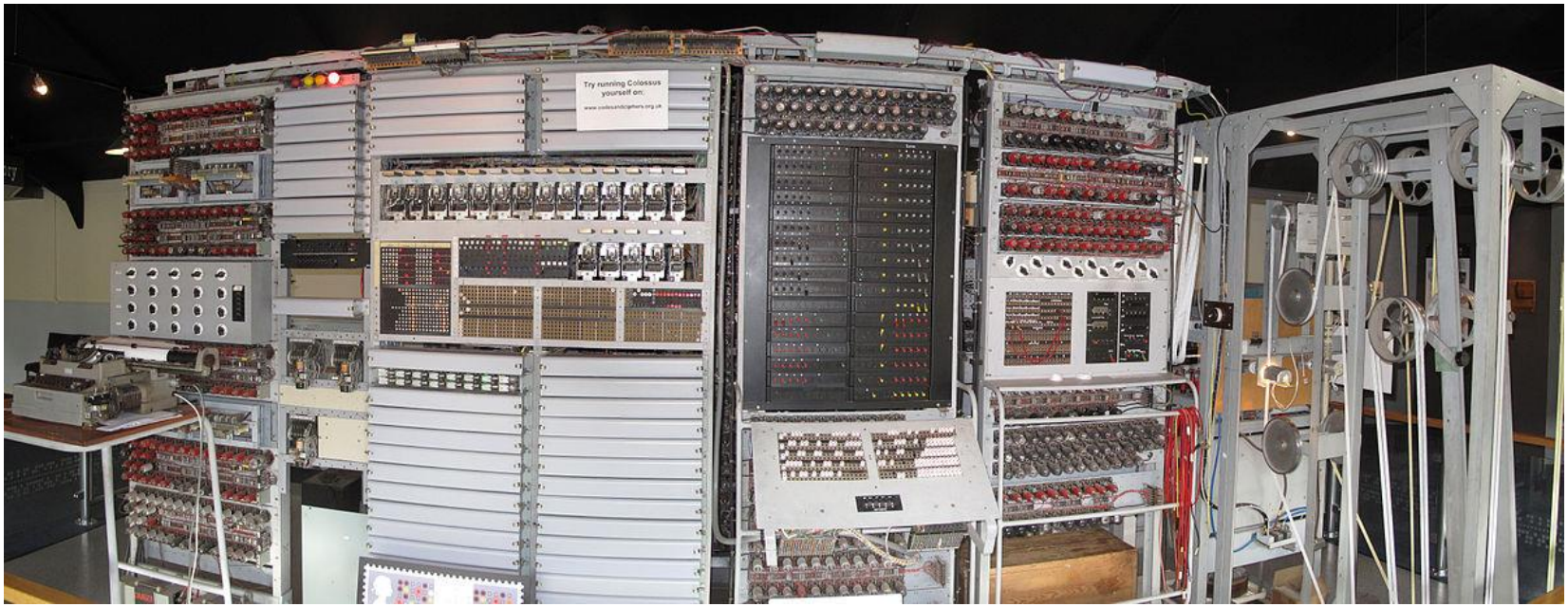
- náhodne vygenerovaný
- utajený (známy len odosielateľovi a príjemcovi)
- nesmie sa použiť viackrát

Šifrátor Lorenz (Tunny)

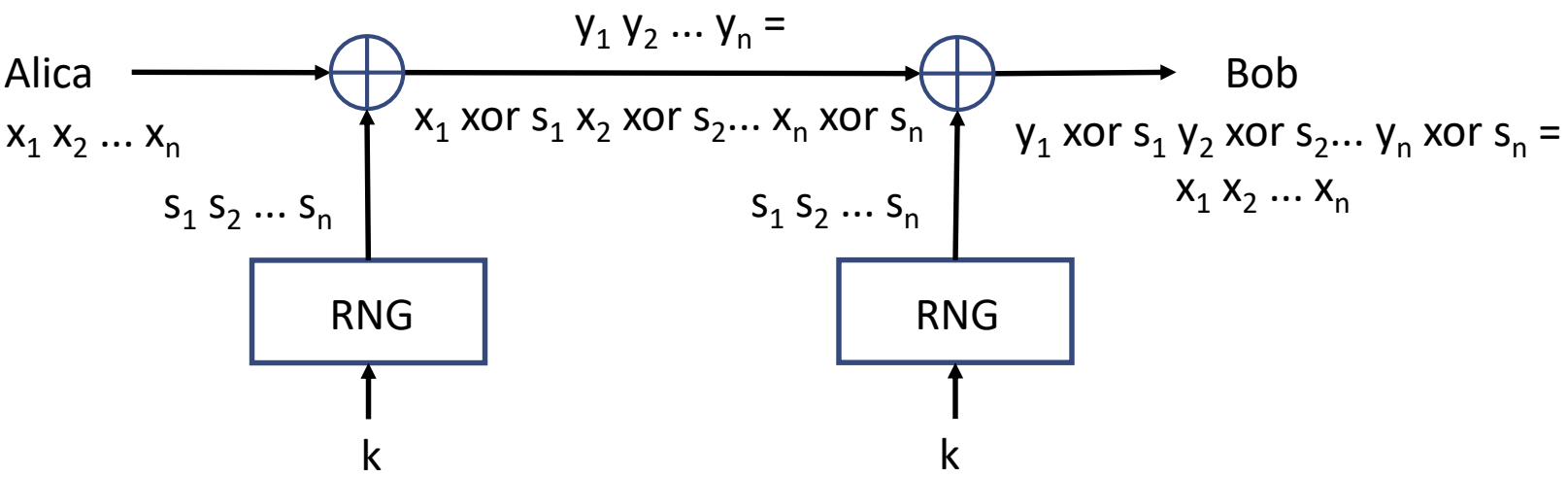
- zariadenie na generovanie dvoch reťazcov bitov pre telegrafický prenos
- XOR s telegrafickou správou (v 5 bitovom kóde - Baudot) vysielané telegraficky – Vernamova šifra
- 2x 5 ozubených kolies s rôznymi počtami „pinov“ v stave 0 resp. 1, druhých 5 kolies sa otáča len ak je prvých 5 kolies v stave 1 (spolu cca 6 x 10^{17} pozícií + ovplyvňovanie plaintextom)
- Colossus – elektronické zariadenie na kryptoanalýzu (1944)



Colossus - rekonštrukcia



Prúdové šifrovacie systémy



- stream (pseudo)náhodne generovaný
- utajený kľúč (počiatočné nastavenie generátora – *seed*)
- nesmie sa použiť viackrát
- problémy s integritou ! (negáciou bitov šifrového textu je možné cielene zmeniť bity v dešifrovanom texte)

Generátory náhodných (pseudonáhodných) čísel

- TRNG (*True random number generator*) hw riešenie (tepelný šum, kvantové javy, udalosti v PC) na dešifrovanie preniesť ďalším bezpečným kanálom !
- PRNG – generátory pseudonáhodných čísel – generovanie závisí (deterministicky) len od počiatočného nastavenia - kľúča (*seed*) konečný počet stavov – po čase za začnú opakovať

lineárne kongruenčné generátory

$$s_0 = seed;\ s_{i+1} = (a\, s_i + b)\ mod\ m$$

napr. $s_{i+1} = (84589\, s_i + 45989)\ mod\ 217728$ s periódou 2^{35}

ANSI C random $s_{i+1} = (1103515245\, s_i + 12345)\ mod\ 2^{31}$

Generátory náhodných (pseudonáhodných) čísel

- CSPRNG – kryptograficky bezpečné generátory pseudonáhodných čísel – výstup nie je možné predikovať (dokázateľne)

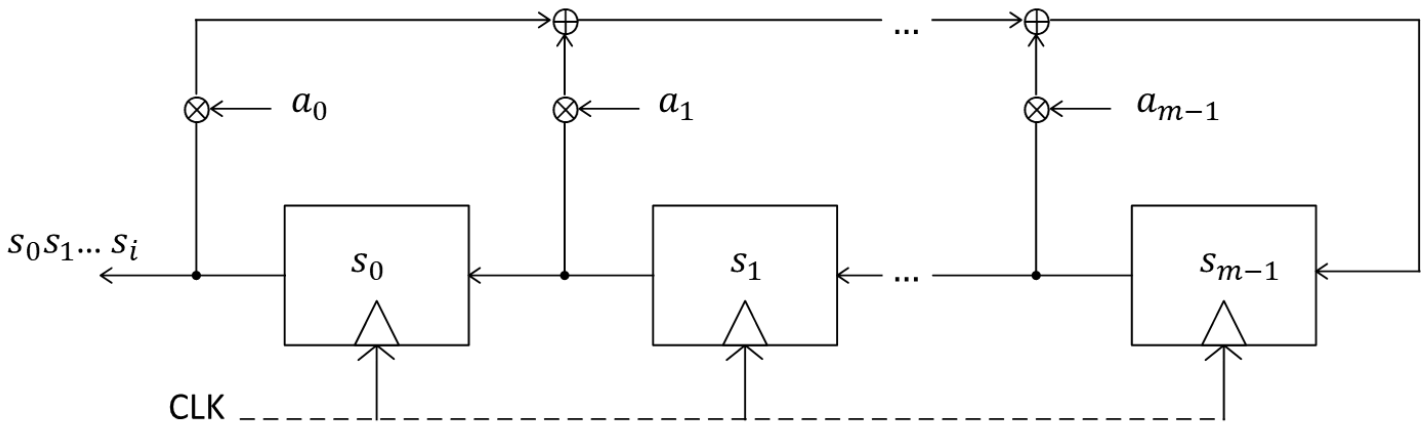
Blum-Blum-Shub generátor pre prvočísla p, q

$$b_{i+1} = b_i^2 \; mod \; pq \qquad p \equiv q \equiv 3 \; (mod \; 4)$$

$$s_{i+1} = b_{i+1} \; mod \; 2$$

predikcia nasledujúceho bitu streamu je ekvivalentná problému riešenia kvadratických rezíduí

Lineárne posuvné registre so spätnou väzbou



- pre binárny vstup $s_i, a_i \in \{0,1\}$; i ∈ {0, 1, … m-1}
  $s_m = (s_{m-1}a_{m-1} + \ \ldots + s_1 a_1 + s_0 a_0) \bmod 2$
- pre $n > m$ rekurentne
  $s_n = (s_{n-1}a_{m-1} + \ \ldots + s_{n-m+1}a_1 + s_{n-m}a_0) \bmod 2$
- implementácia logickými obvodmi $\oplus$ XOR $\otimes$ AND
- posuvné registre – posun synchronizovaný signálmi CLK

príklad

$s_4 = (s_2 + s_0) \bmod 2 \quad (a_0 = 1, a_1 = 0, a_2 = 1, a_3 = 0)$

a potom $\quad s_{i+4} = (s_{i+2} + s_i) \bmod 2$

kľúč 1010 00101000101000 … perióda 101000

kľúč 1100 111100111100 … perióda 110011

kľúč 1011 0110110 … perióda 101

$s_4 = (s_3 + s_0) \bmod 2 \quad (a_0 = 1, a_1 = 0, a_2 = 0, a_3 = 1)$

1011 00100011110 1011… perióda 101100100011110

- maximálna dĺžka periódy ?

LFSR (linear-feedback shift register)

- kľúčom (resp. IV - inicializačným vektorom) je počiatočná postupnosť $s_0, s_1, \dots, s_{m-1}$
- $\forall i \geq m: \quad s_i = \sum_{j=1}^{m} a_{m-j} s_{i-j} \mod 2$

  resp. tiež $\forall i \geq 0: \quad s_{i+m} = \sum_{j=0}^{m-1} a_j s_{i+j} \mod 2$
- maximálna dĺžka periódy $2^m - 1$ (PRNG s približne rovnakým výskytom 0 a 1)

- výpočty je možné reprezentovať v okruhu binárnych polynómov GF(2) s operáciami XOR a AND
- $\mathbb{F}_2[X]$ charakteristický polynóm

  $X^m + a_{m-1}X^{m-1} + a_{m-2}X^{m-2} \dots + a_1 X + a_0$

LFSR – kryptoanalýza (KPA)

- zo známej časti otvoreného a šifrovaného textu E(P) = P ⊕ S vyjadriť časť streamu S = P ⊕ E(P)
- pre postupnosť $s_0, s_1, \dots, s_{2m-1}$ vyriešiť sústavu rovníc v $\mathbb{Z}_2$

$$\begin{pmatrix} s_0 & \cdots & s_{m-1} \\ \vdots & \ddots & \vdots \\ s_{m-1} & \cdots & s_{2m-2} \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_{m-1} \end{pmatrix} = \begin{pmatrix} s_m \\ \vdots \\ s_{2m-1} \end{pmatrix}$$

- ak nepoznáme n, možno skúšať postupne hodnoty m, pokiaľ nedostaneme požadovaný tvar streamu
- overenie – do dĺžky periódy (2^m -1)

LFSR - príklad

011010111100010011010111 ...

pre m = 2 :

$0.a_0 + 1.a_1 = 1$

$1.a_0 + 1.a_1 = 0$

z prvej rovnice $a_1 = 1$
dosadením do druhej rovnice $a_0 = 1$ (v $\mathbb{Z}_2$)

teda $s_n = s_{n-1}a_1 + s_{n-2}a_0 = s_{n-1} + s_{n-2}$

dostaneme 011011011011011011 ...

nesprávne riešenie !

LFSR - príklad

011010111100010011010111 ...

pre m = 3 :

$$0.a_0 + 1.a_1 + 1.a_2 = 0$$

$$1.a_0 + 1.a_1 + 0.a_2 = 1$$

$$1.a_0 + 0.a_1 + 1.a_2 = 0$$

súčtom prvých dvoch rovníc je

$1.a_0 + 0.a_1 + 1.a_2 = 1$ čo je spor s treťou rovnicou – riešenie neexistuje

$$\det \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} = 0$$

LFSR - príklad

011010111100010011010111 ...

pre m = 4 :

$0.a_0 + 1.a_1 + 1.a_2 + 0.a_3 = 1$

$1.a_0 + 1.a_1 + 0.a_2 + 1.a_3 = 0$

$1.a_0 + 0.a_1 + 1.a_2 + 0.a_3 = 1$

$0.a_0 + 1.a_1 + 0.a_2 + 1.a_3 = 1$

- zo súčtu druhej a štvrtej rovnice je $a_0 = 1$,
- dosadením do tretej rovnice dostaneme $a_2 = 0$,
- potom z prvej rovnice $a_1 = 1$ a zo štvrtej $a_3 = 0$

teda $s_n = s_{n-1}a_3 + s_{n-2}a_2 + s_{n-3}a_1 + s_{n-4}a_0 = s_{n-3} + s_{n-4}$

skúška : 011010111100010011010111 o.k. ?

LFSR - príklad

0110101111000100110101 11 ...

pre m = 5 :

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

súčet prvých dvoch riadkov je piaty riadok – existuje viac riešení

jedno je $s_n = s_{n-3} + s_{n-4}$ ($a_0 = 0$)

druhé $s_n = s_{n-1} + s_{n-3} + s_{n-5}$ ($a_0 = 1$)

det = 0 dĺžka najkratšieho LFSR zodpovedá poslednému nenulovému determinantu matíc pre m=1, 2, ...

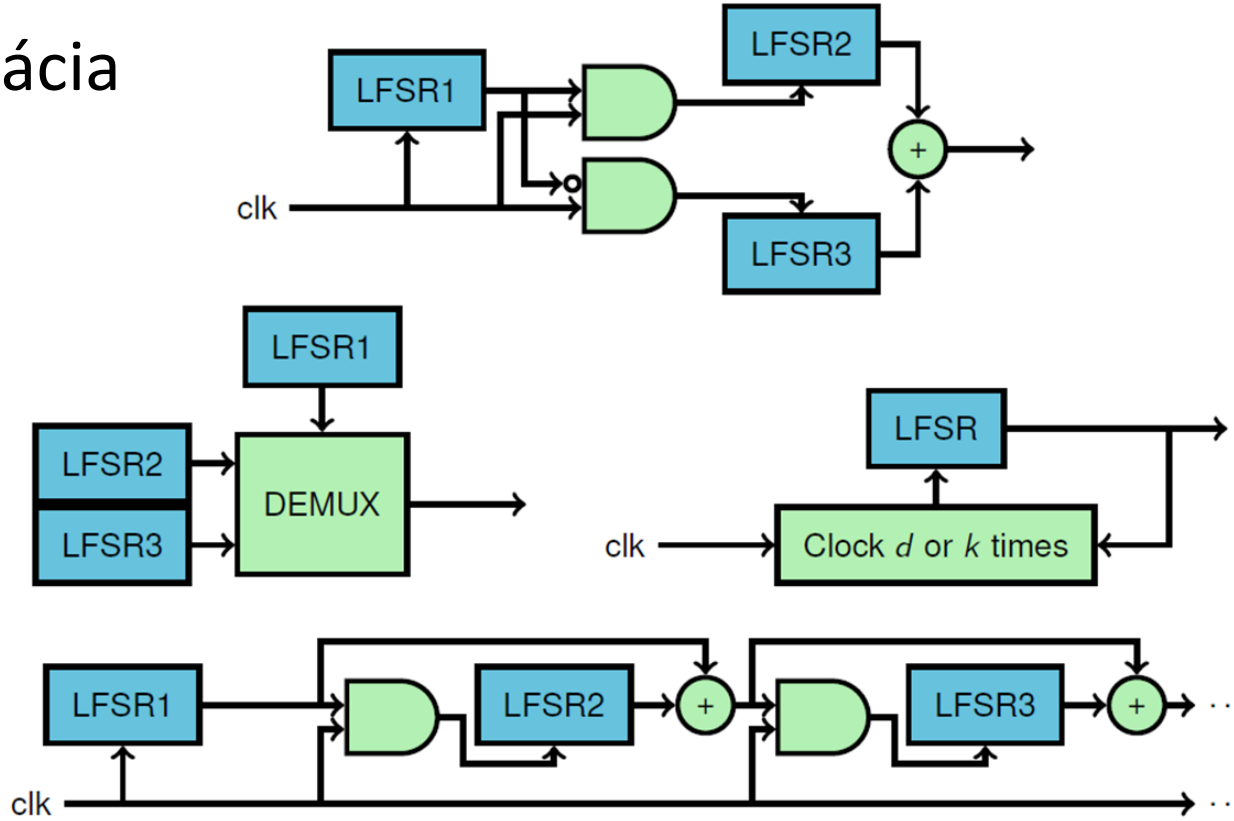
analýza LFSR

- stačí poznať sekvenciu 2m hodnôt kdekoľvek (KPA)
- pokiaľ je v známej sekvencii k núl za sebou -> m > k
- ak je k jednotiek za sebou -> m ≥ k
- skúšať postupne dĺžku LFSR m, pokiaľ nebude správna skúška do dĺžky $2^m - 1$

- Berlekamp-Massey algoritmus na určenie m – polynomiálna zložitosť

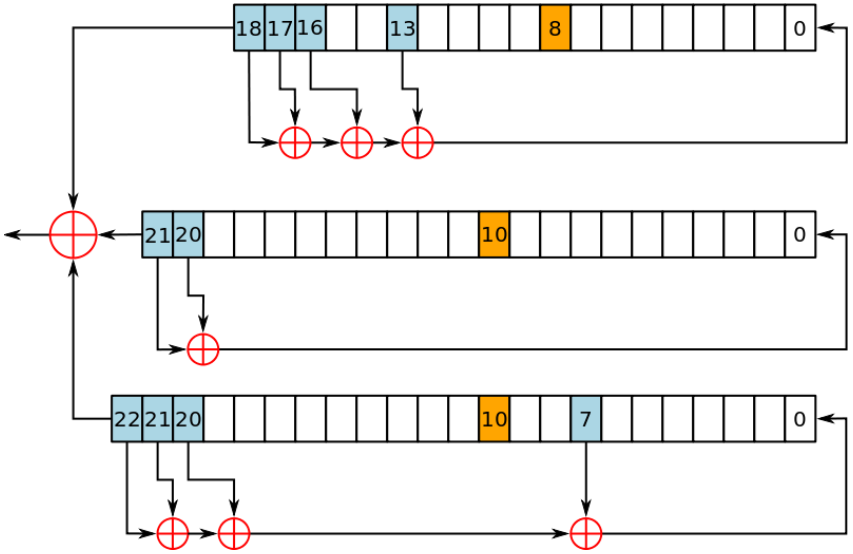
nelineárne generátory

- kombinácia LFSR registrov
- synchronizácia



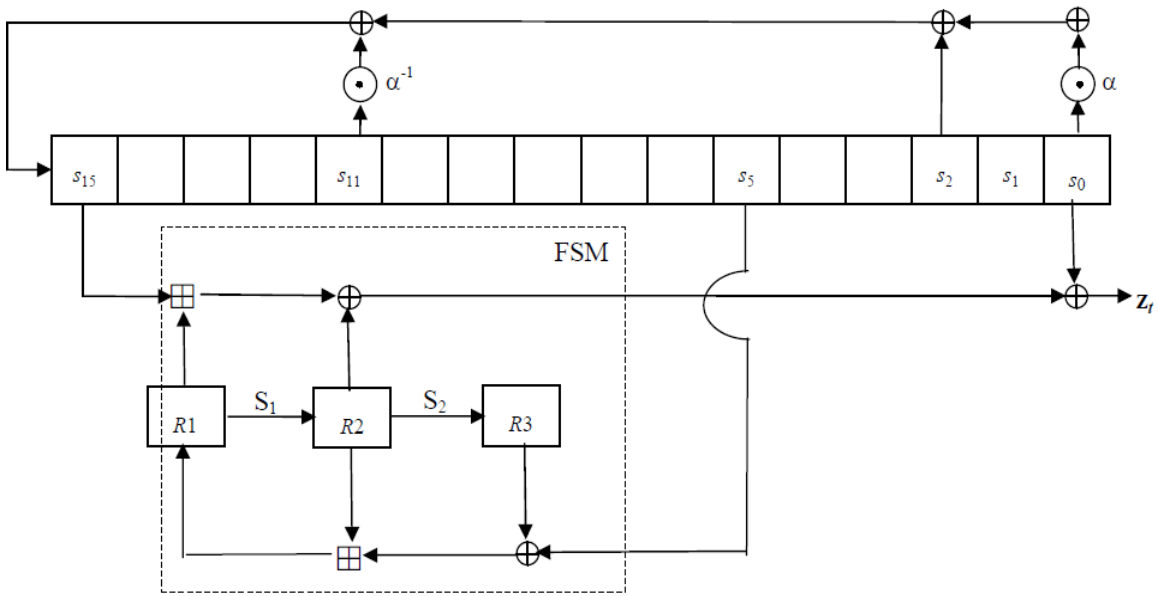
A5/1

- prúdová šifra pre GSM
- 3 LFSR registre 19, 22, 23
- posúvajú sa len tie, ktoré sa zhodnú na vyznačených pozíciách (oranžové)
- 64 bitový kľúč – pri inicializácii sa pridá do každého registra
- 22 bitové číslo rámca (frame number) podobne
- 100 prázdnych krokov
- 2x 114 bitov výstup (každých 4,615 ms)



prelomené Dec 2009
A5/1 cracking project

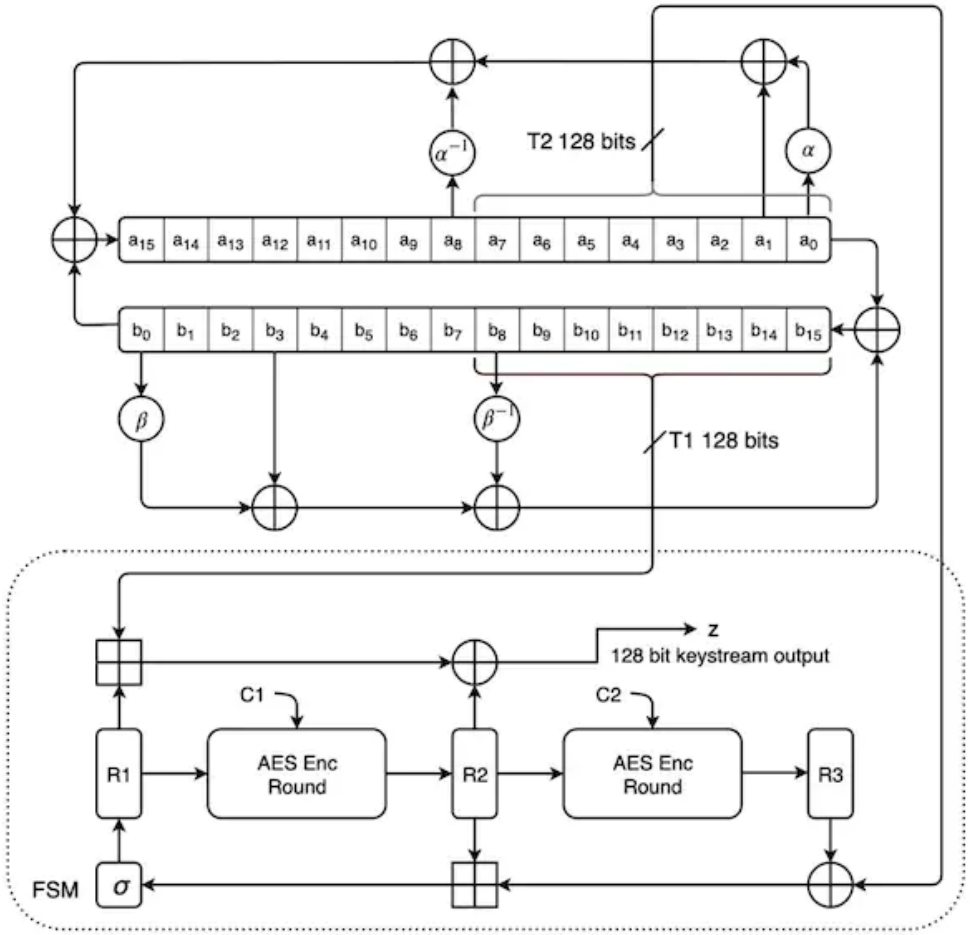
SNOW 3G



- 16 x 32 bit LFSR, 3 registre (32 bit) FSM
- operácie mod 2 a mod 2^{32} (3GPP)
- 128 bit key + 128 bit IV - nastavenie s_0 … s_{15}

SNOW-V pre virtualizované 5G

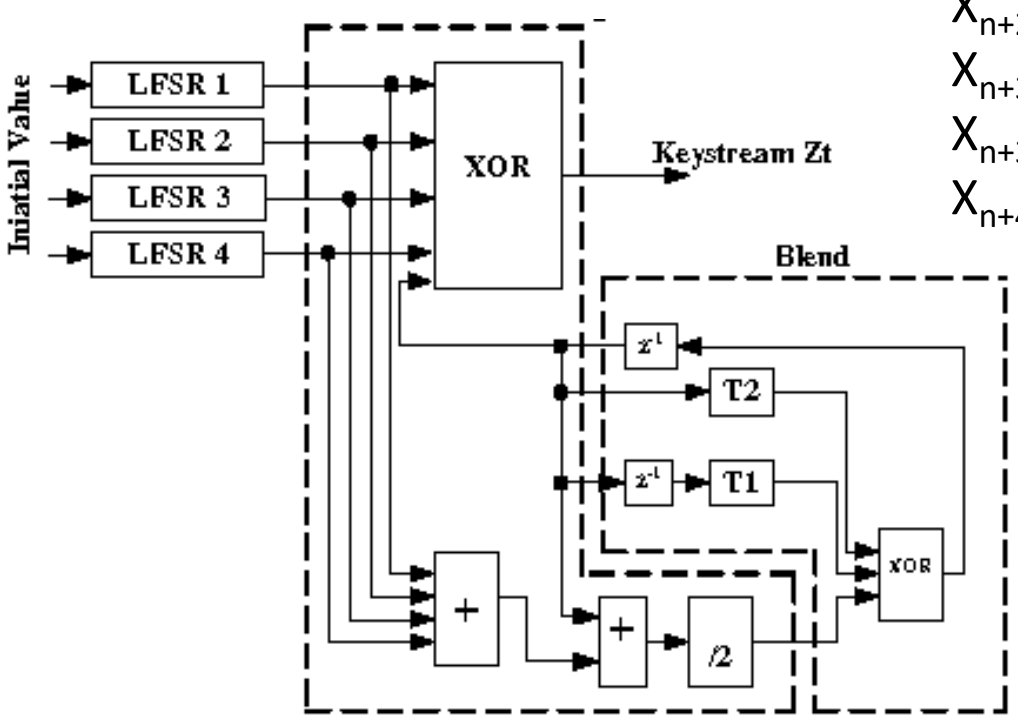
- virtualizovaný komponent - rôzne hw platformy
- aspoň 20 Gb/s
- 2x 16 x 16 bit LFSR (per. 2^{512} -1)
- 3x FSM reg (128b) AES 128b encrypt
- 256 bit key 128 bit IV



E0 (Bluetooth)

- 4 LFSR a dva dvojbitové vnútorné stavové registre

$$X_{n+26} = X_n + X_{n+8} + X_{n+12} + X_{n+20} + X_{n+25}$$
$$X_{n+32} = X_n + X_{n+12} + X_{n+16} + X_{n+24} + X_{n+31}$$
$$X_{n+34} = X_n + X_{n+4} + X_{n+24} + X_{n+28} + X_{n+33}$$
$$X_{n+40} = X_n + X_{n+4} + X_{n+28} + X_{n+36} + X_{n+39}$$



- známy útok zložitosti 2^{38}

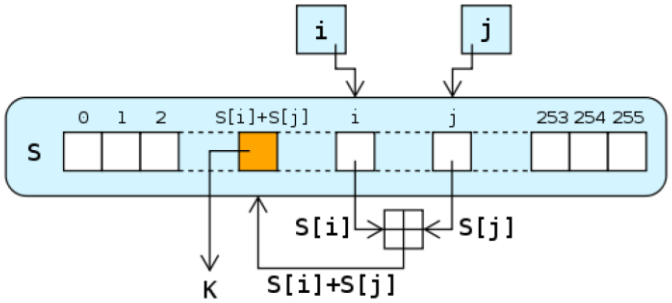
RC4 (ARCFOUR)

- prúdová šifra (Rivest cipher 1987) implementovateľná na jednoduchých procesoroch (čipových kartách)
- obchodné tajomstvo RSA (zverejnené 1994)
- neúspešne použitá pre zabezpečenie bezdrôtových sietí WEP (nevhodné použitie inicializačného vektora)

permutačné pole 256 bajtov, kľúč (IV) do 256 bajtov (prakticky 5 – 16 B, teda 40 – 128 bitov kľúča)

podľa kľúča sa inicializuje permutačné pole (číslami 0 – 255)

generovanie bajtov na základe dvoch indexov a výmeny obsahov bajtov v permutačnom poli



RC4

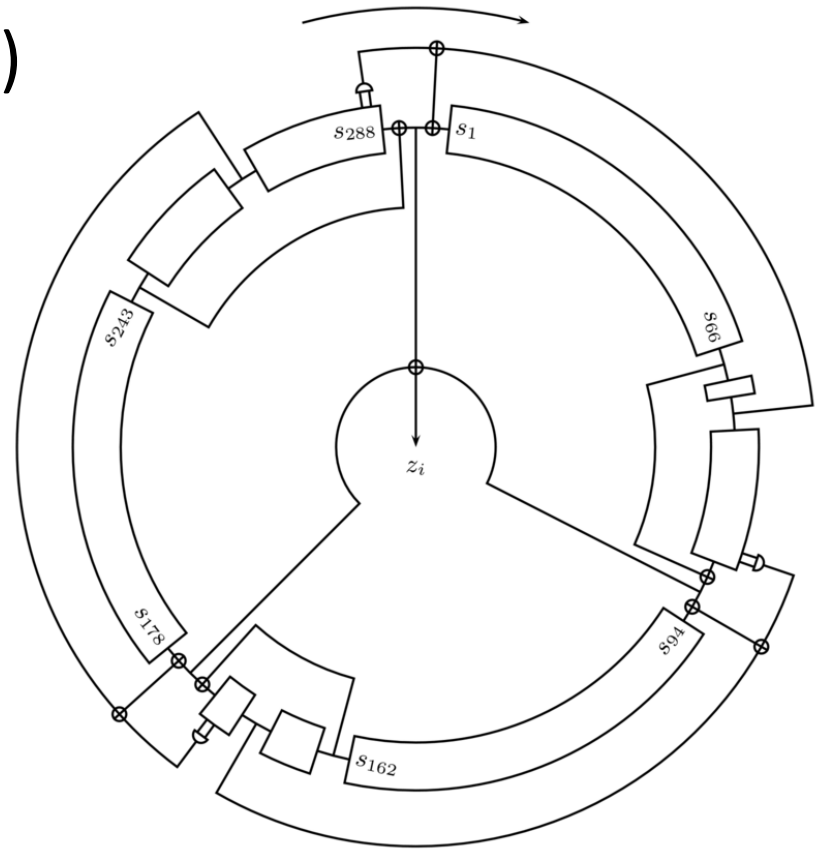
```
for i from 0 to 255 do S[i] := i endfor;
j := 0; for i from 0 to 255 do
   j := (j + S[i] + key[i mod keylength]) mod 256;
   swap values of S[i] and S[j]  endfor;

i := 0; j := 0;
while GeneratingOutput do
   i := (i + 1) mod 256;    j := (j + S[i]) mod 256;
   swap values of S[i] and S[j];
   K := S[(S[i] + S[j]) mod 256];
   output K
endwhile;
```

eSTREAM

- projekt EU ECRYPT – nové prúdové šifry (2004)
- vyhodnotenie 2012 – eSTREAM portfolio
- sw profil (kľúč 128 b)
  HC-128, Rabbit, Salsa20, SOSEMANUK
- hw profil (kľúč 80 b)
  Grain, MICKEY, Trivium
- bez licenčných obmedzení

Trivium

- 80 b kľúč + 80 b IV
- 3 LFSR registre (93, 84, 111) s nelineárnymi komponentami
- výstup jedného registra je vstupom do ďalšieho (288 b kruhový register)
- nelineárne prvky na vstupe i výstupe z registra
- výsledok – XOR výstupov

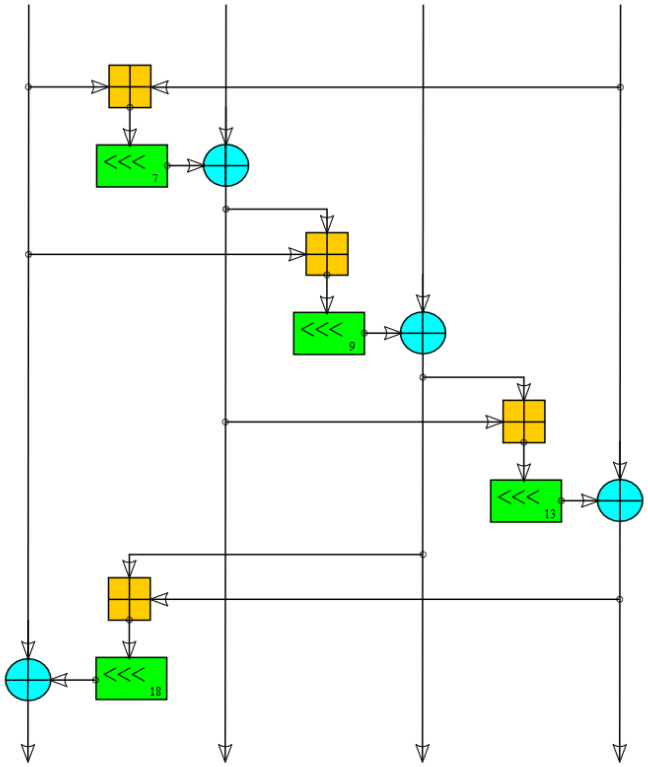


Salsa20

- Bernstein 2007 – public domain
- add-rotate-xor (ARX)
- kľúč 256 b, IV 64 b, cnt 64 b do 512 b bloku (key stream) 16 x 32 bitov

„expa“	K	K	K
K	„nd 3“	IV	IV
cnt	cnt	„2-by“	K
K	K	K	„te k“

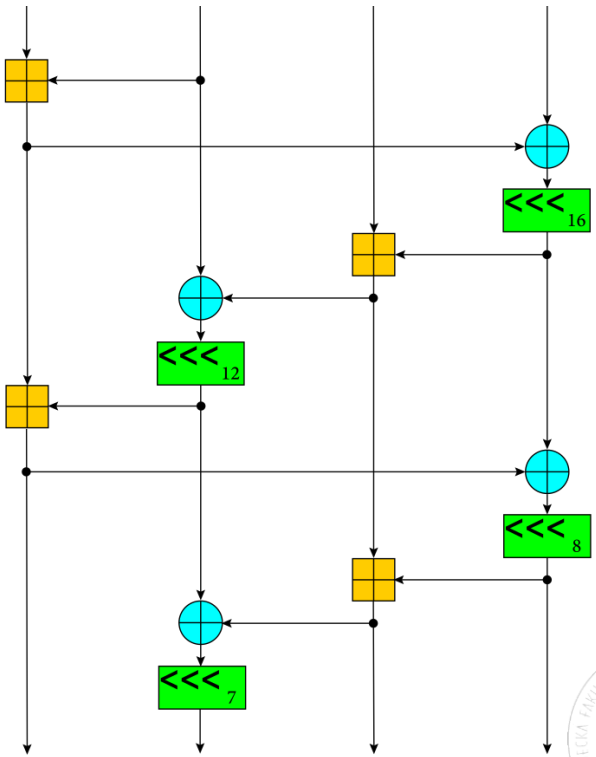
- 20 rúnd (stĺpce + riadky)



ChaCha20

- Bernstein 2008 – namiesto RC4 v TLS (RFC 7539) ARM
- kľúč 256 b, IV 64 b, cnt 64 b
  (resp. IV 96 b, cnt 32 b)
- do 512 b bloku

„expa“	„nd 3“	„2-by“	„te k“
K	K	K	K
K	K	K	K
cnt	cnt	IV	IV



- 10x 8 transf. (4 stĺpce + 4 diagonály)
  spolu 20x prepis celého bloku
  možno paralelizovať
- kvôli cnt len 2^{32} blokov (256 GB)

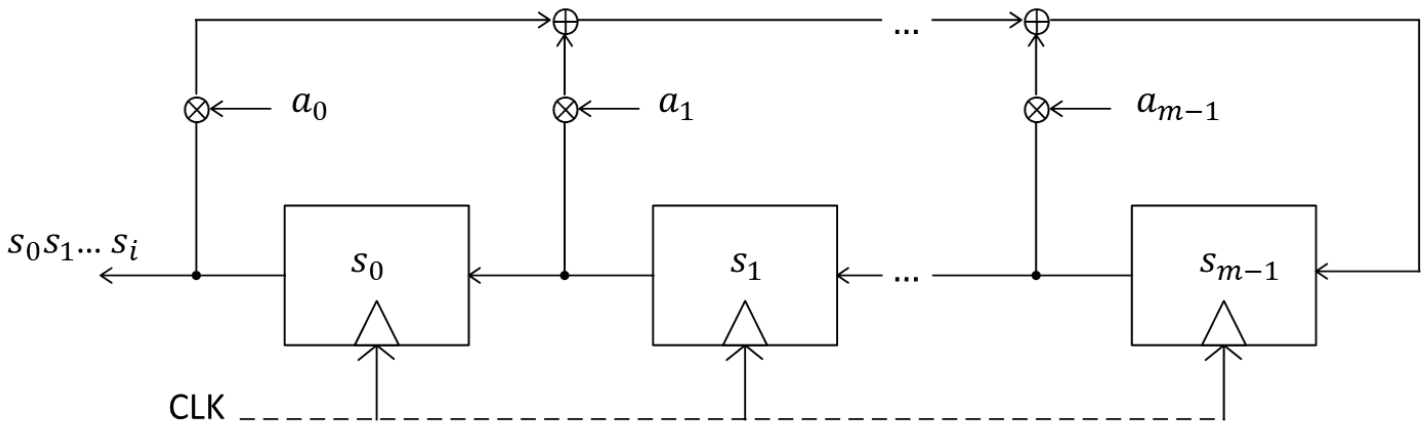
Chacha20

```
#define ROTL(a,b) (((a) << (b)) | ((a) >> (32 - (b))))
#define QR(a, b, c, d) (	a += b,  d ^= a,  d = ROTL(d,16),		\
                                c += d,  b ^= c,  b = ROTL(b,12),		\
                                a += b,  d ^= a,  d = ROTL(d, 8),		\
                                c += d,  b ^= c,  b = ROTL(b, 7)	)
void chacha_block(uint32_t out[16], uint32_t const in[16])
{		int i;		uint32_t x[16];
		for (i = 0; i < 16; ++i)	x[i] = in[i];
		for (i = 0; i < 20; i += 2) {
                                QR(x[0], x[4], x[ 8], x[12]); // column 0
                                QR(x[1], x[5], x[ 9], x[13]); // column 1
                                QR(x[2], x[6], x[10], x[14]); // column 2
                                QR(x[3], x[7], x[11], x[15]); // column 3
                                QR(x[0], x[5], x[10], x[15]); // diagonal 1 (main diagonal)
                                QR(x[1], x[6], x[11], x[12]); // diagonal 2
                                QR(x[2], x[7], x[ 8], x[13]); // diagonal 3
                                QR(x[3], x[4], x[ 9], x[14]); // diagonal 4
		}
		for (i = 0; i < 16; ++i)	out[i] = x[i] + in[i];
}
```

Ďakujem za pozornosť.

jozef.jirasek@upjs.sk

Linear-feedback shift register



- $s_i, a_i \in \{0,1\}$; i ∈ {0, 1, … m-1}
  $s_m = (s_{m-1}a_{m-1} + \ \ldots + s_1 a_1 + s_0 a_0) \bmod 2$
- for $n > m$
  $s_n = (s_{n-1}a_{m-1} + \ \ldots + s_{n-m+1}a_1 + s_{n-m}a_0) \bmod 2$
- implementation by logical circuits $\oplus$ XOR $\otimes$ AND
- shift register – shift synchronized by CLK signals

Example

$s_4 = (s_2 + s_0) \bmod 2 \quad (a_0 = 1, a_1 = 0, a_2 = 1, a_3 = 0)$

and $\quad s_{i+4} = (s_{i+2} + s_i) \bmod 2$

keystream 1010 00101000101000 … period 101000

keystream 1100 111100111100 … period 110011

keystream 1011 0110110 … period 101

$s_4 = (s_3 + s_0) \bmod 2 \quad (a_0 = 1, a_1 = 0, a_2 = 0, a_3 = 1)$

1011 00100011110 1011… period 101100100011110

- maximum period length ?

LFSR (linear-feedback shift register)

- key (or IV – initialization vector)
  is the initial sequence $s_0, s_1, \dots, s_{m-1}$
- $\forall i \geq m: \quad s_i = \sum_{j=1}^{m} a_{m-j} s_{i-j} \mod 2$

  or $\quad \forall i \geq 0: \quad s_{i+m} = \sum_{j=0}^{m-1} a_j s_{i+j} \mod 2$
- maximum period length $2^m - 1$
  (PRNG with approximately equal occurrence of 0 and 1)

- calculations can be represented using binary polynomial rings over GF(2) with operations XOR and AND
- $\mathbb{F}_2[X]$ characteristic polynomial
  $X^m + a_{m-1}X^{m-1} + a_{m-2}X^{m-2} \dots + a_1 X + a_0$

LFSR – cryptoanalysis with known plaintext (KPA)

- from the known part of the plaintext and ciphertext E(P) = P ⊕ S express the part of the stream S = P ⊕ E(P)
- for S = $s_0, s_1, \dots, s_{2m-1}$ solve linear equations over $\mathbb{Z}_2$

$$\begin{pmatrix} s_0 & \cdots & s_{m-1} \\ \vdots & \ddots & \vdots \\ s_{m-1} & \cdots & s_{2m-2} \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_{m-1} \end{pmatrix} = \begin{pmatrix} s_m \\ \vdots \\ s_{2m-1} \end{pmatrix}$$

- if we don't know n, we can try successive values of m until we get the desired stream shape (unique solution)
- verification – up to period length (2^m -1)

LFSR - example

011010111100010011010111 ...

for m = 2 :

$0.a_0 + 1.a_1 = 1$

$1.a_0 + 1.a_1 = 0$

from the first equation $a_1 = 1$
substituting to the second $a_0 = 1$ (in $\mathbb{Z}_2$)

so $s_n = s_{n-1}a_1 + s_{n-2}a_0 = s_{n-1} + s_{n-2}$

and we get 011011011011011011 ...

wrong solution !

LFSR - example

011010111100010011010111 ...

for m = 3 :

$$0.a_0 + 1.a_1 + 1.a_2 = 0$$

$$1.a_0 + 1.a_1 + 0.a_2 = 1$$

$$1.a_0 + 0.a_1 + 1.a_2 = 0$$

the sum of the first two equations is

$1.a_0 + 0.a_1 + 1.a_2 = 1$ contradiction to third equation – there is no solution

$$\det \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} = 0$$

LFSR - example

011010111100010011010111 ...

for m = 4 :

$0.a_0 + 1.a_1 + 1.a_2 + 0.a_3 = 1$

$1.a_0 + 1.a_1 + 0.a_2 + 1.a_3 = 0$

$1.a_0 + 0.a_1 + 1.a_2 + 0.a_3 = 1$

$0.a_0 + 1.a_1 + 0.a_2 + 1.a_3 = 1$

- from the sum of the second and fourth equations, $a_0 = 1$,
- substituting it into the third equation, we get $a_2 = 0$,
- then from the first equation $a_1 = 1$ and from the fourth $a_3 = 0$

so $s_n = s_{n-1}a_3 + s_{n-2}a_2 + s_{n-3}a_1 + s_{n-4}a_0 = s_{n-3} + s_{n-4}$

verify : 011010111100010011010111 o.k. ?

LFSR - example

011010111100010011010111 ...

for m = 5 :

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

the sum of the first two rows is the fifth row – there are many solutions

first $s_n = s_{n-3} + s_{n-4}$ ($a_0 = 0$)

second $s_n = s_{n-1} + s_{n-3} + s_{n-5}$ ($a_0 = 1$)

det = 0 the length of the shortest LFSR corresponds to the last non-zero determinant of the matrices for m = 1, 2, ...

LFSR analysis

- it is enough to know the sequence of 2m values of keystream anywhere (KPA)
- if there are k consecutive zeros -> $m > k$
- if there are k consecutive ones -> $m \geq k$
- tes successively the length m of the LFSR until the correct test within length $2^m - 1$

- Berlekamp-Massey algorithm to determine m with polynomial complexity