A New Generation of Fast Flexible Stream Ciphers

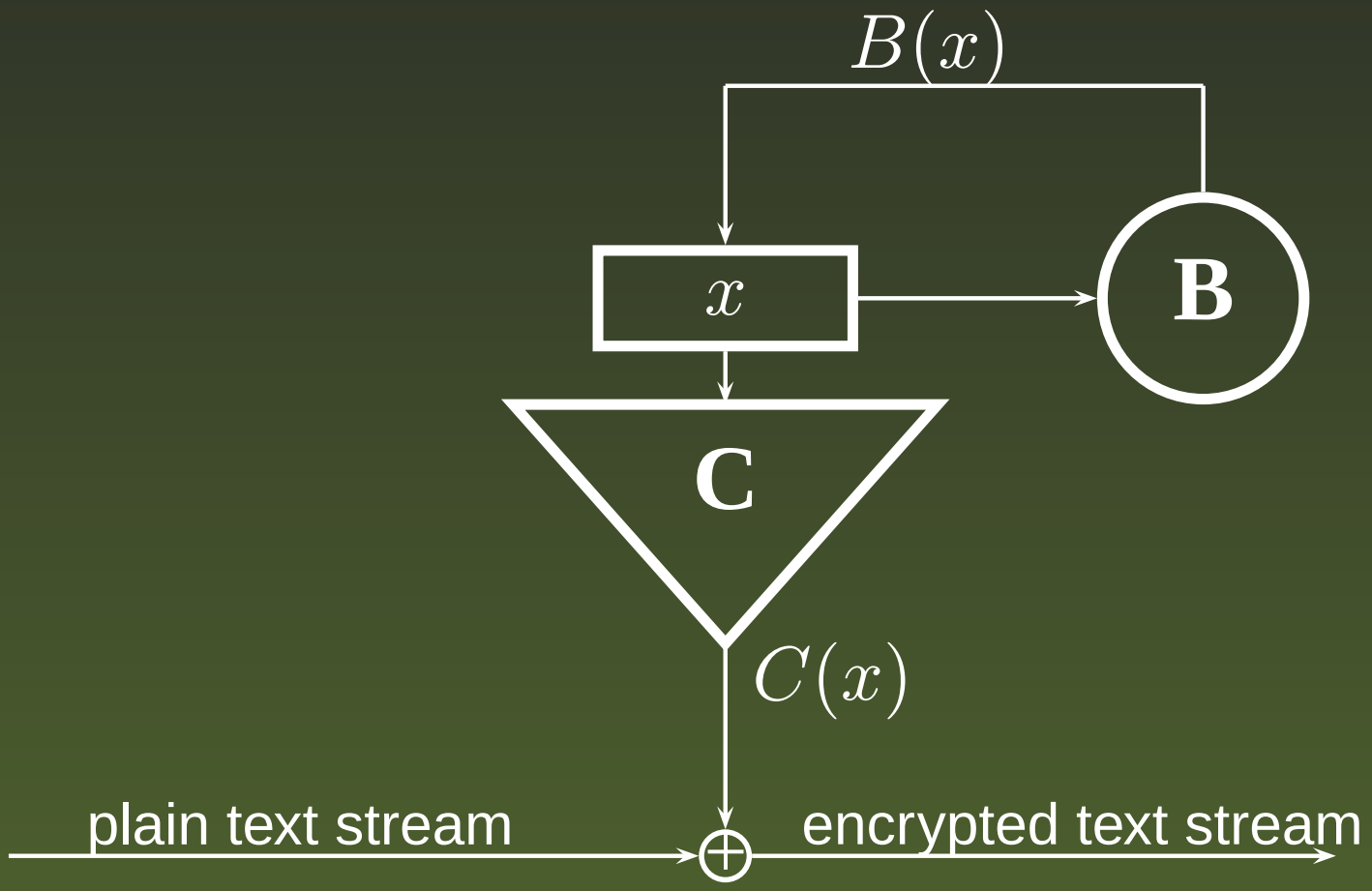
ANDREY BOGDANOV

University of Duisburg-Essen
Institute for Experimental Mathematics

Motivation

- Usage of new cryptographical primitives as building blocks of ciphers: T-functions and skew shifts.
- Illustration of some efficient techniques resting upon p-adic analysis and automata theory.
- A highly flexible framework for manufacturing fast and secure ciphers.

Traditional design of stream ciphers



B	state transition function,	period and distribution
C	non-linear filter function,	other crypto properties

Traditional design of stream ciphers

Traditionally the following primitives have been used:

- State transition B = LFSR over $GF(2)$ or $GF(2^k)$;
- Filter C = complicated boolean functions or some sort of movement irregularity.

There are also some specific design approaches:

- R-boxes (RC4, Py, Scream, SNOW),
- S-boxes (TSC),
- Block cipher iteration procedures (SUSEMANUK, LEX, Salsa20),
- Field or ring multiplications (Rabbit),
- Feedback with carry shift registers (FCSR) and s.o.

New primitives for stream ciphers

We suggest working in rings $\mathbb{Z}/2^k$ for some suitable k and using:

- T-functions:
  - More simple structures to analyse;
  - First of all transitive and bijective ones;
- Skew products of finite automata:
  - To build complex objects from simple ones.

An efficient theory to analyse very complex encrypting mappings!

New primitives for stream ciphers

Properties:

- Long period of output keystream,
- Uniform distribution of output keystream,
- High linear complexity over $GF(2)$,
- High l-error complexity (no efficient linear analogues of keystream).

Idea:

- Make all the used maps
  - key dependent and
  - clock-dependent.

Other crypto applications

As a matter of fact the techniques are very efficient for

- block ciphers,
- MACs,
- column substitution ciphers.

All these ciphers will inherit security properties mentioned above from the underlying building blocks!

T-functions: Definition, Part 1

- Every map G in $\mathbb{Z}/2^n\mathbb{Z}$

$$z \in \mathbb{Z}/2^n\mathbb{Z}, z = (z_0, \ldots, z_{n-1}) = \sum_{i=0}^{n-1} z_i 2^i,$$
$$G : \mathbb{Z}/2^n\mathbb{Z} \to \mathbb{Z}/2^n\mathbb{Z},$$
$$G : z \mapsto (\varphi_0(z_0, \ldots, z_{n-1}), \ldots, \varphi_{n-1}(z_0, \ldots, z_{n-1}))$$

can be given using n (boolean) coordinate functions

$$\varphi_i : (\mathbb{Z}/2\mathbb{Z})^n \to (\mathbb{Z}/2\mathbb{Z})^n, \ i = 0, \ldots, n.$$

T-functions: Definition, Part 2

- $G : \mathbb{Z}/2^n\mathbb{Z} \to \mathbb{Z}/2^n\mathbb{Z}, G = (\varphi_0, \ldots, \varphi_{n-1})$ is a **T-function** iff:

$$\begin{aligned}
\varphi_0 &= \varphi(z_0), \\
\varphi_1 &= \varphi(z_0, z_1), \\
&\ldots \\
\varphi_{n-2} &= \varphi(z_0, \ldots, z_{n-2}), \\
\varphi_{n-1} &= \varphi(z_0, \ldots, z_{n-2}, z_{n-1}).
\end{aligned}$$

- T-function = triangle function.
- A number of equivalent well-known definitions in mathematics.

T-functions: Definition, Part 3

What are examples of T-functions?

- Almost all computer operations: $+$ and $\cdot$ in $\mathbb{Z}/2^k$, all bitwise logical operations (AND, OR, XOR, NOT, ...), zero fill left shifts. Exceptions: Cyclic shifts, right shifts.
- All the T-functional compositions of T-functions are T-functions: Polynomials and s.o.

T-functions: Invertibility

- The T-function $G = (\varphi_0, \ldots, \varphi_{n-1})$ in $\mathbb{Z}/2^n\mathbb{Z}$ is **invertible** **(bijective)** iff:
  - $\forall i = 0, \ldots, n-1$ the ANF of $\varphi_i(z_0, \ldots, z_i)$ is linear with respect to z_i.
- It is often advisable to use invertible functions as output functions in stream ciphers.

T-functions: Singe Cycle Property

- The T-function $G = (\varphi_0, \ldots, \varphi_{n-1})$ in $\mathbb{Z}/2^n\mathbb{Z}$ is **transitive (defines a singe-cycle permutation)** iff:
  - $\forall i = 0, \ldots, n-1$ the ANF of $\varphi_i(z_0, \ldots, z_i)$ is linear with respect to z_i,
  - the weight of φ_0 is 1,
  - for $i = 1, \ldots, n-1$ the weight of φ_i is odd,
- The linear complexity of the i-th coordinate sequence of a transitive function is $2^i + 1$. Single cycle functions guarantee the longest possible period and can be maps of choice for state update functions of stream ciphers.

T-functions: Examples of Transitive Functions, Part 1

- (Larin, early 80th; published 2002) A polynomial with integer coefficients is transitive $\pmod{2^n}$ iff it is transitive modulo 8.
- (Anashin, 1993) The function $F(x) = a_0 + b_1 \cdot (x \oplus a_1) + b_2 \cdot (x \oplus a_2) + \cdots$ is transitive $\pmod{2^n}$ iff it is transitive modulo 4.
- (Anashin, 1993) For arbitrary polynomials $u(x), v(x) \in \mathbb{Z}/2^n\mathbb{Z}[x]$ the integer-valued function

$$F(x) = \frac{v(x)}{2 \cdot u(x) + 1}$$

is transitive $\pmod{2^n}$ iff it is transitive modulo 8.

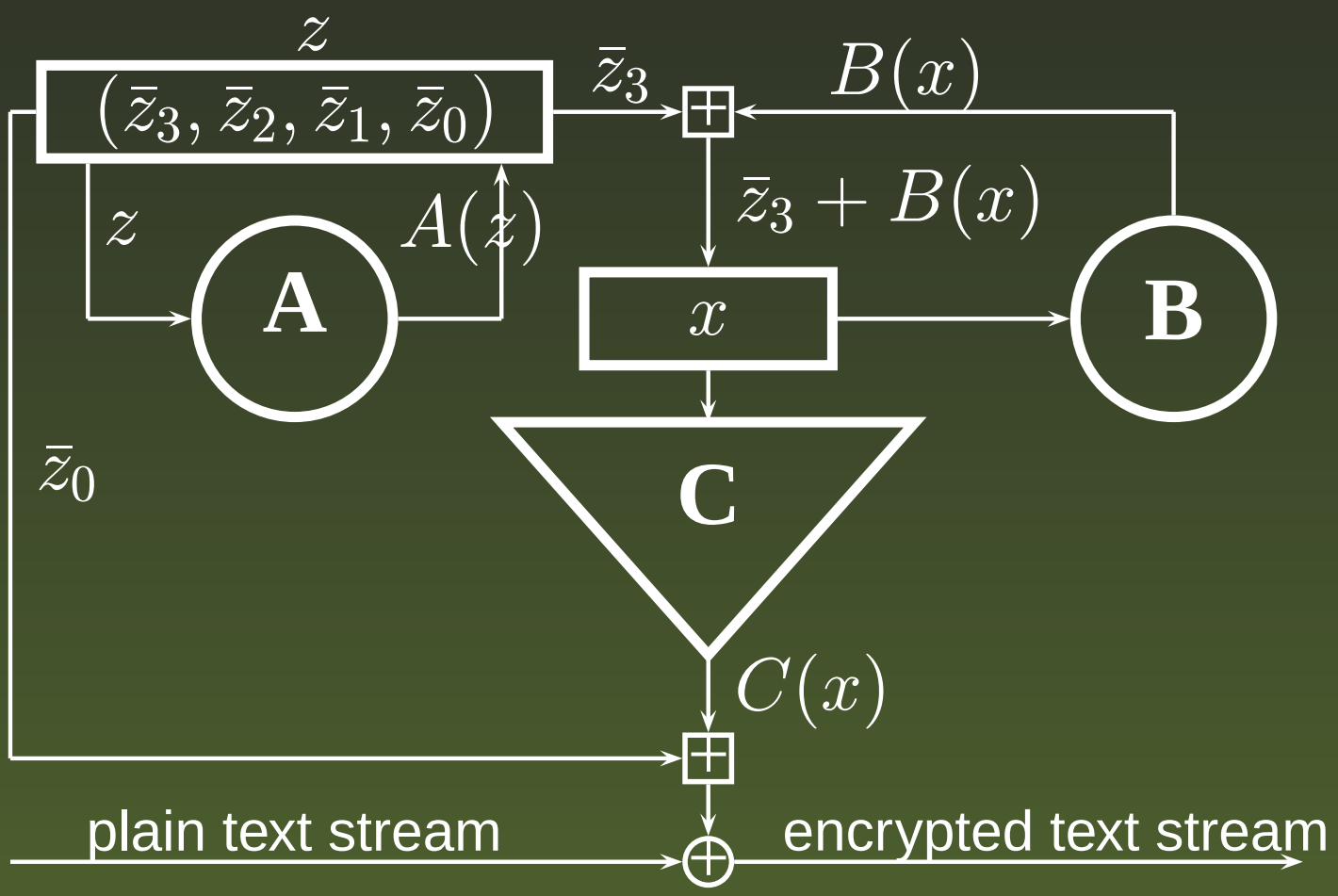
T-functions: Examples of Transitive Functions, Part 2

- (Kotomina, 1998) The function

  $$f(x) = (\ldots((((x+c_0)\oplus d_0)+c_1)\oplus d_1)+\cdots+c_m)\oplus d_m,$$

  is transitive $\pmod{2^n}$ iff f is transitive modulo 4
- (Anashin, 2002) The function $F(x) = a \cdot x + a^x$ is transitive $\pmod{2^n}$ iff a is odd.

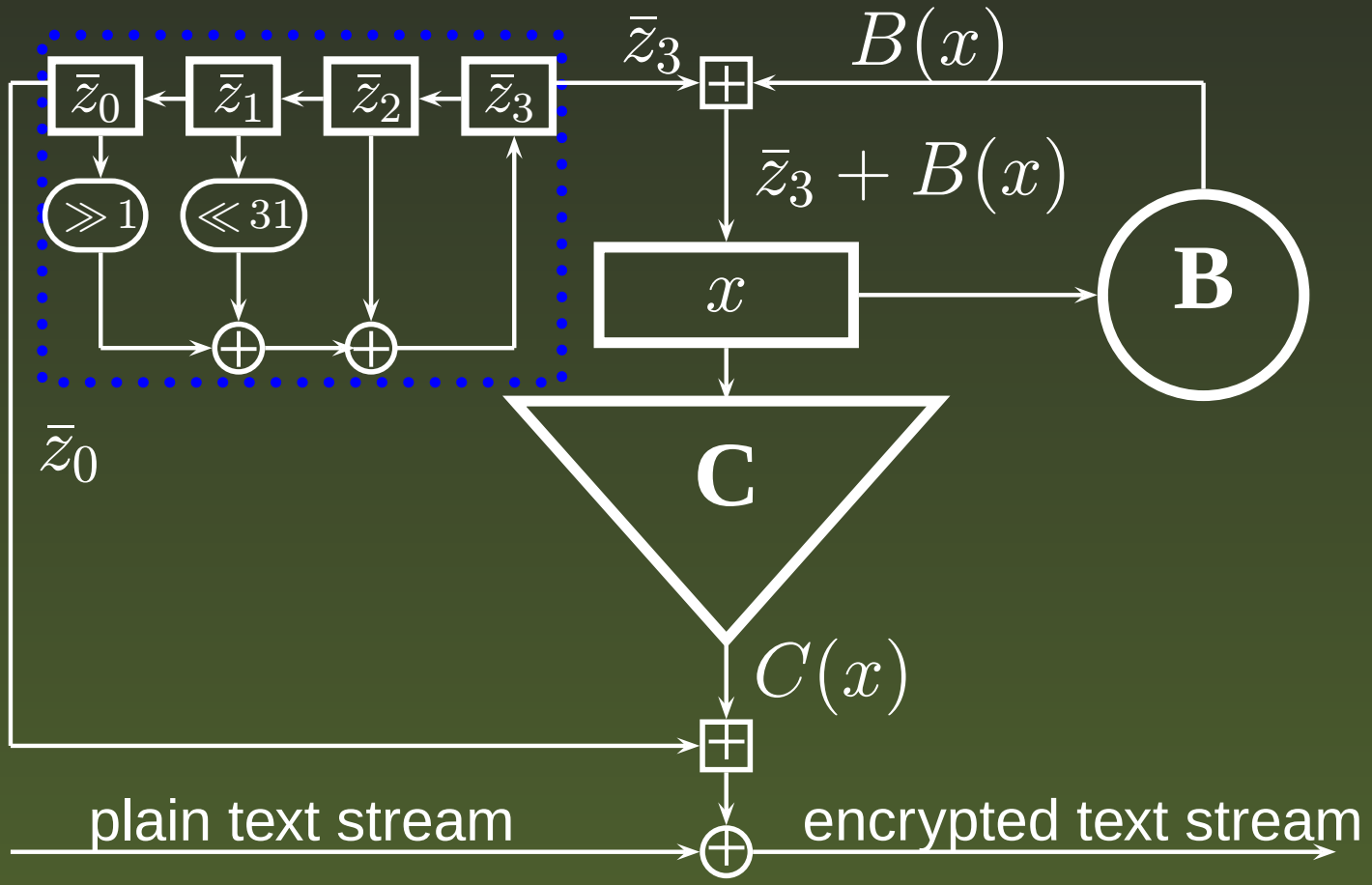
Example of new design pattern: ABC



$\boxplus$ = $+ \pmod{2^{32}}$

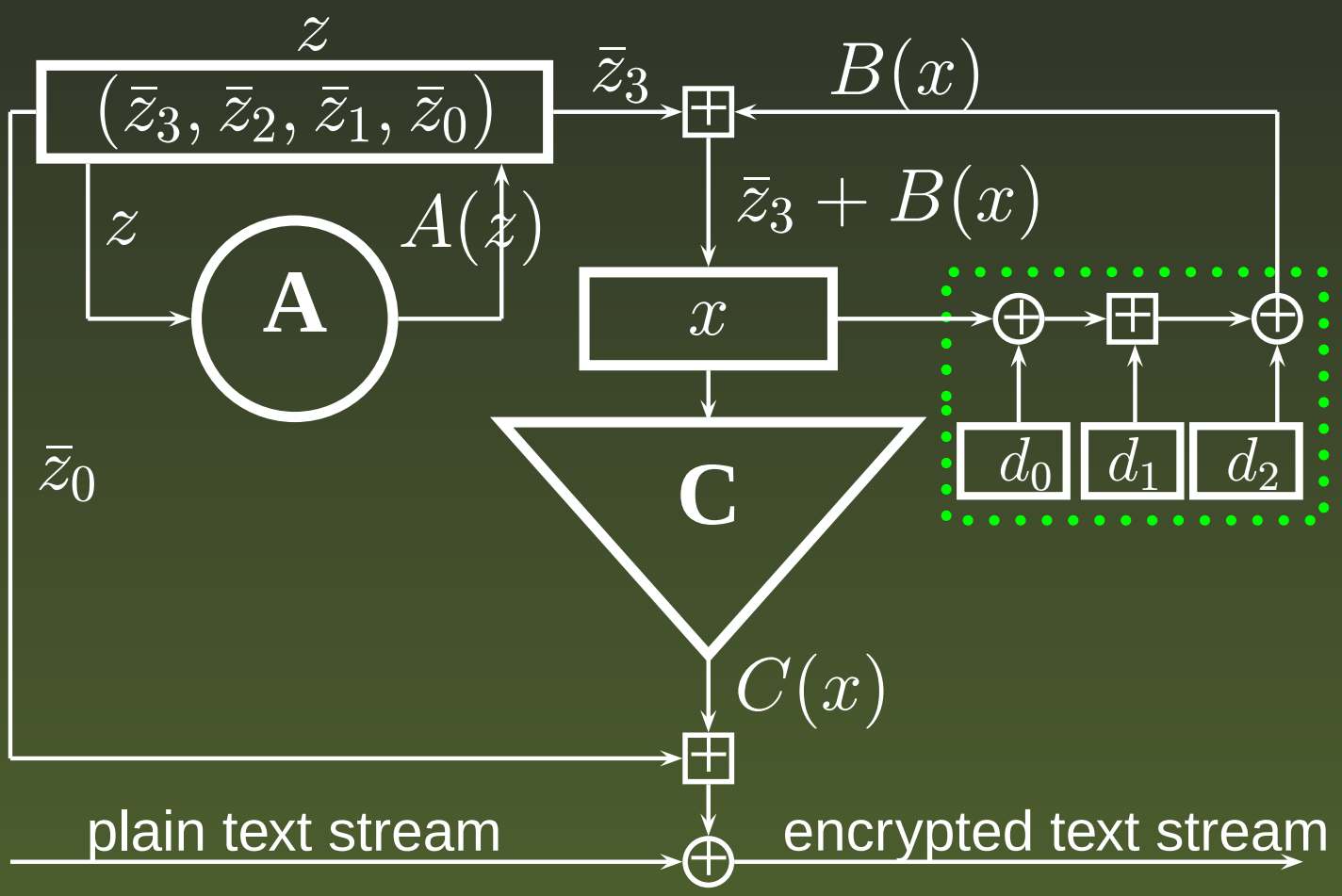
$\oplus$ = XOR

ABC: Function A



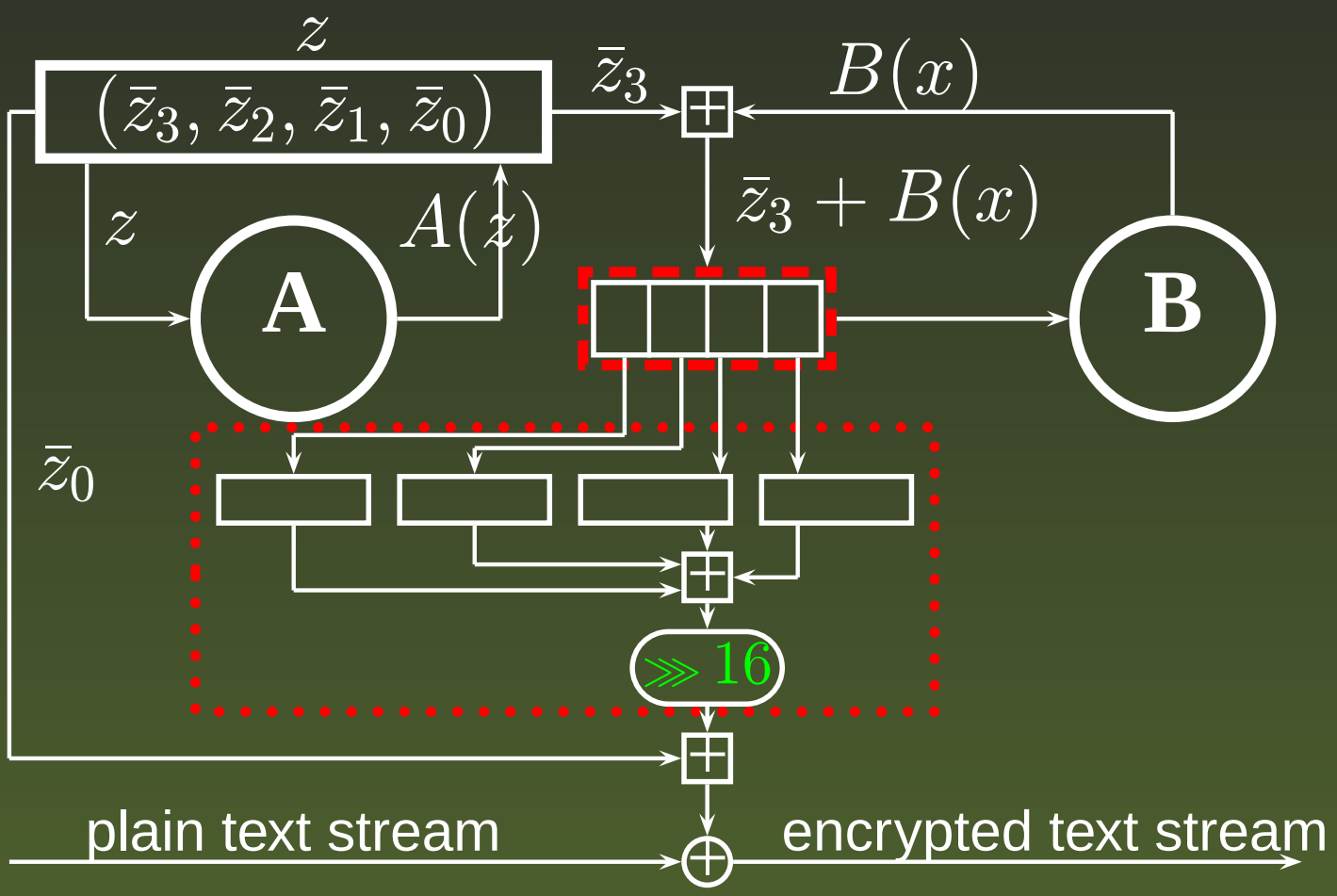
A : LFSR of period $2^{127} - 1$ for each 32-bit subword

ABC: Function B



B : Defines a single cycle permutation over $\mathbb{Z}/2^{32}\mathbb{Z}$

ABC: Function C



Properties of the ABC design pattern

Provable properties of the ABC key stream:

- The period of $(2^{127}-1)\cdot 2^{32}$ words;
- Uniformly distributed key stream: $\forall$ 32-bit word a the number $\mu(a)$ of occurrences of a on the period satisfies:

$$\left|\frac{\mu(a)}{(2^{127}-1)\cdot 2^{32}} - \frac{1}{2^{32}}\right| < \frac{1}{\sqrt{(2^{127}-1)\cdot 2^{32}}};$$

- High linear complexity λ of the key stream: $2^{31}\cdot(2^{127}-1)+1 \geq \lambda \geq 2^{31}+1$.

Properties of ABC circuit: Notes

- As a matter of fact we have proved the group of statements for a larger class of A, B, C. Thus, the designer can choose the maps suitable for the specific requirements.
- Note that the fact that these crucial security properties are proven does not exclude the necessity to analyse the concrete representations of A, B and C with respect to the whole set of cryptographical attacks.

ABC: Speed & Memory consumption

- A ***generic* reference C** implementation on a standard 3.2 GHz Intel Pentium 4 processor under Linux.
- Minimum **132** byte memory used.

w	Speed, Gbps	Cycles per byte	Table memory, bytes
2	2.25	11.38	256
4	4.24	6.04	512
8	**6.86**	3.73	4096

Conclusion

- Freedom to choose mappings;
- Important security properties are ***proven***;
- Novel approach to ***counter-dependence***;
- High degree of ***key-dependence***;
- ***Key*** material usage ***flexibility***;
- Extremely high throughput rate can be achieved.