

Stream Ciphers

BMEVITMAV52

Information and Network Security

feher.gabor@tmit.bme.hu

Stream ciphers

- Comparison to block ciphers
 - Block ciphers process blocks (64 bits or more)
 - Stream ciphers work on smaller “blocks” usually 1 bit
 - Block ciphers have no memory
 - Stream ciphers need state information (memory) in addition to the key
 - block ciphers + memory (e.g. CFB)
-> stream ciphers

One-time pad

- Vernam cipher
 - $c_i = p_i \text{ XOR } k_i$
- There exists a perfect cipher: one time pad
 - Vernam cipher
 - Key should be as long as the plaintext
 - Unconditionally secure (Shannon)
 - But, one-time pad is not feasible due to the long key
- Keystream generation
 - **The goal of the stream ciphers is to provide a long pseudorandom keystream based on a smaller key**
 - Computationally secure

Binary additive stream ciphers

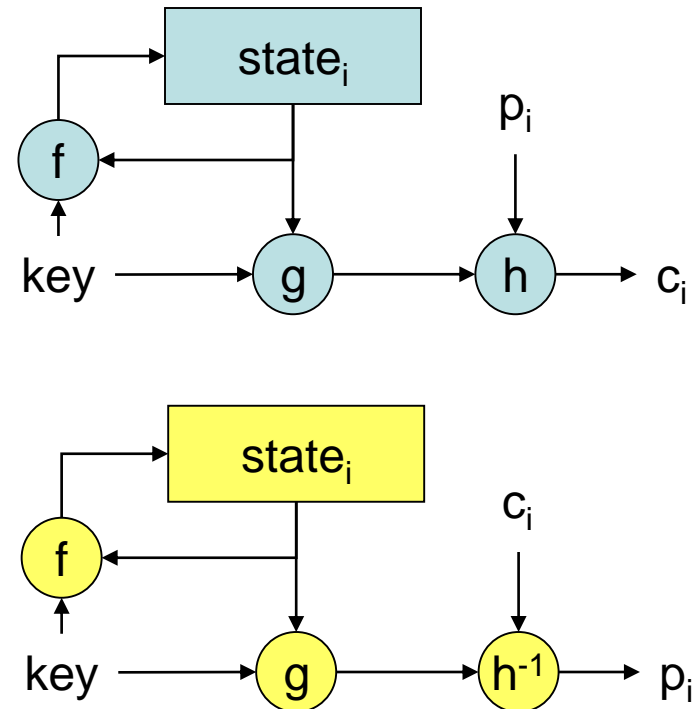
- Synchronous stream ciphers, where the output function is the XOR operation
 - Just like Vernam cipher
- Most of the stream ciphers are additive stream ciphers

Synchronous stream ciphers

- Definition: keystream is generated independently of the plaintext and ciphertext
- Procedure:
 - $\text{state}_{i+1} = f(\text{state}_i, k)$ next state function
 - $z_i = g(\text{state}_i, k)$ keystream
 - $c_i = h(p_i, z_i)$ output function

Synchronous stream ciphers 2.

- Properties
 - Sender and receiver must be synchronized
 - Same key and same position
 - Lost synchronization fails decryption
 - Bit insertion/removal or replay attack are recognized
 - Bit errors produce bit errors
 - No error propagation
 - Ideal for lossy communication
 - Attackers know the result of the modifications
- -> OFB is a synchronous stream cipher

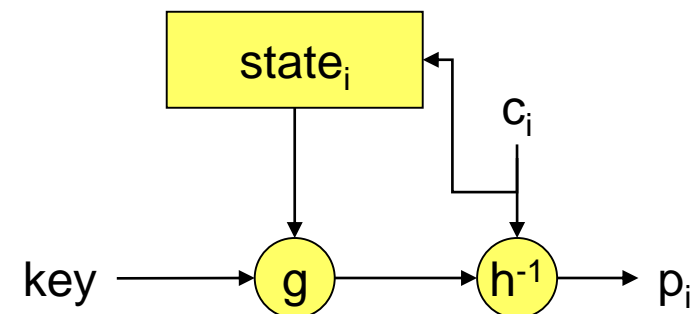
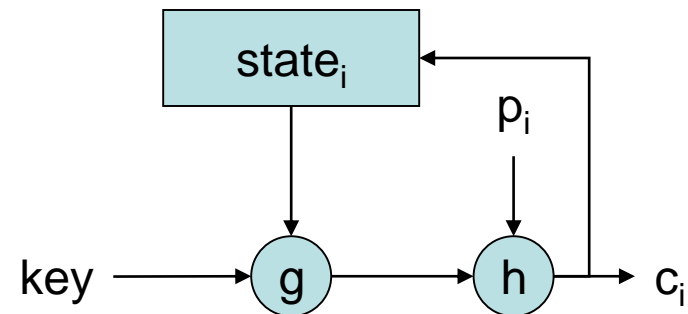


Asynchronous stream ciphers

- Self synchronizing stream ciphers
- Definition: keystream is generated from the key and a number of previous ciphertext bits
- Procedure
 - $\text{state}_i = (c_{i-1}, c_{i-2}, \dots, c_{i-t})$
 - $z_i = g(\text{state}_i, k)$ keystream
 - $c_i = h(p_i, z_i)$ output function

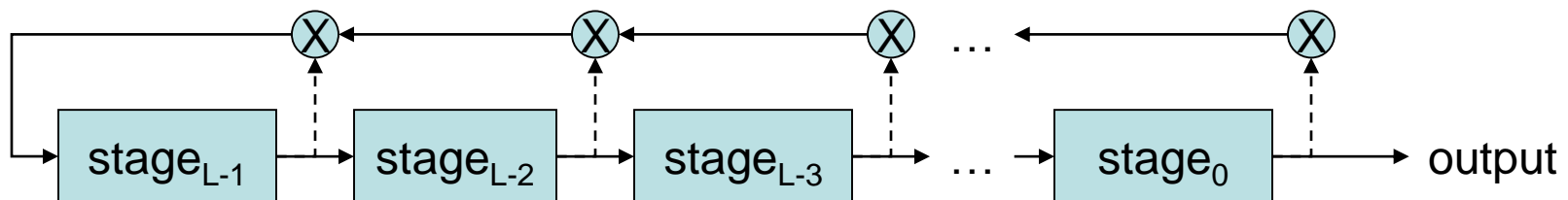
Asynchronous stream ciphers 2.

- Properties
 - Receiver is capable of recover after lost synchronization
 - Self synchronization after bit insertion/removal
 - Plaintext is unrecoverable during the synchronization
 - Bit errors produce more erroneous bits
 - Limited error propagation: up to t bits
 - Attackers can not be sure about the result of the modifications
 - Plaintext bits influence the rest of the ciphering
- -> CFB is an asynchronous stream cipher



Linear Feedback Shift Registers

- LFSR
 - Consists of L stages storing 1 bit information
 - A clock moves data through the stages \rightarrow shift
 - The last stage serves the output
 - The input of the first stage is a feedback, calculated as a modulo sum of a subset of the stages
- LFSR properties
 - Easy hardware implementation
 - Large period
 - Good statistical properties
 - Analysis through algebraic techniques

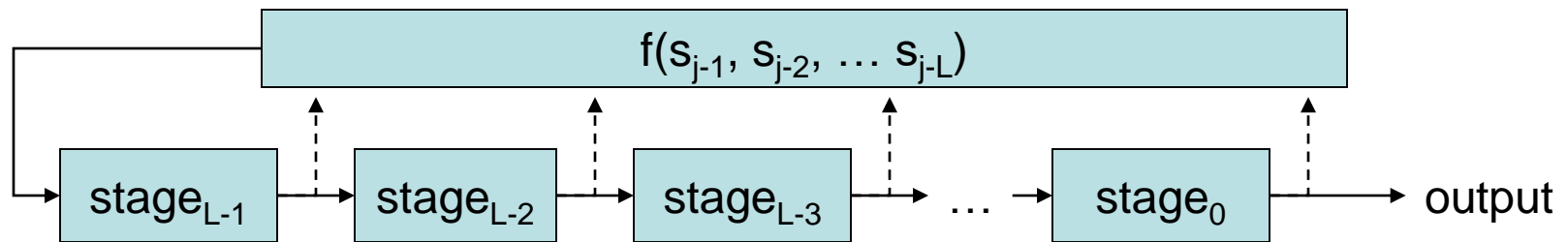


Maximum length LFSR

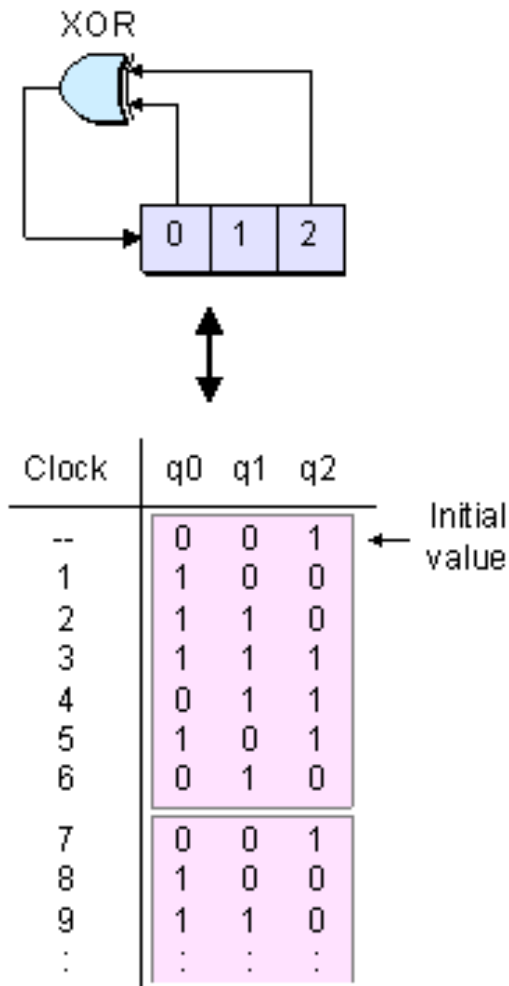
- LFSRs are periodic, since states are finite
 - Maximum length LFSR
 - Maximum length LFSR if it produces an output with period $2^L - 1$
 - Linear complexity
 - Linear complexity of s_n – denoted as $L(s_n)$ – is the shortest LFSR that produces a sequence having s_n as its first outputs

Nonlinear Feedback Shift Registers

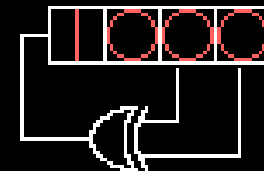
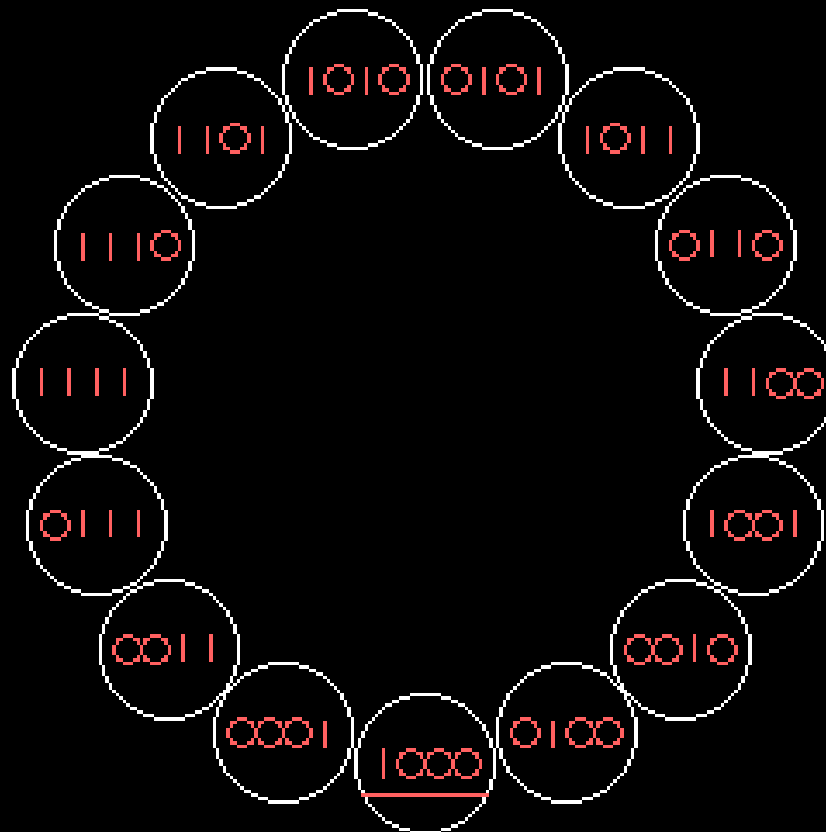
- Feedback is any function of the stages
- No theory, how to create long periods!



LFSR example



# of Bits	Length of Loop	Taps
2	3 *	[0,1]
3	7 *	[0,2]
4	15	[0,3]
5	31 *	[1,4]
6	63	[0,5]
7	127 *	[0,6]
8	255	[1,2,3,7]
9	511	[3,8]
10	1,023	[2,9]
11	2,047	[1,10]
12	4,095	[0,3,5,11]
13	8,191 *	[0,2,3,12]
14	16,383	[0,2,4,13]
15	32,767	[0,14]
16	65,535	[1,2,4,15]
17	131,071 *	[2,16]
18	262,143	[6,17]
19	524,287 *	[0,1,4,18]
20	1,048,575	[2,19]
21	2,097,151	[1,20]
22	4,194,303	[0,21]
23	8,388,607	[4,22]
24	16,777,215	[0,2,3,23]
25	33,554,431	[2,24]
26	67,108,863	[0,1,5,25]
27	134,217,727	[0,1,4,26]
28	268,435,455	[2,27]
29	536,870,911	[1,28]
30	1,073,741,823	[0,3,5,29]
31	2,147,483,647 *	[2,30]
32	4,294,967,295	[1,5,6,31]

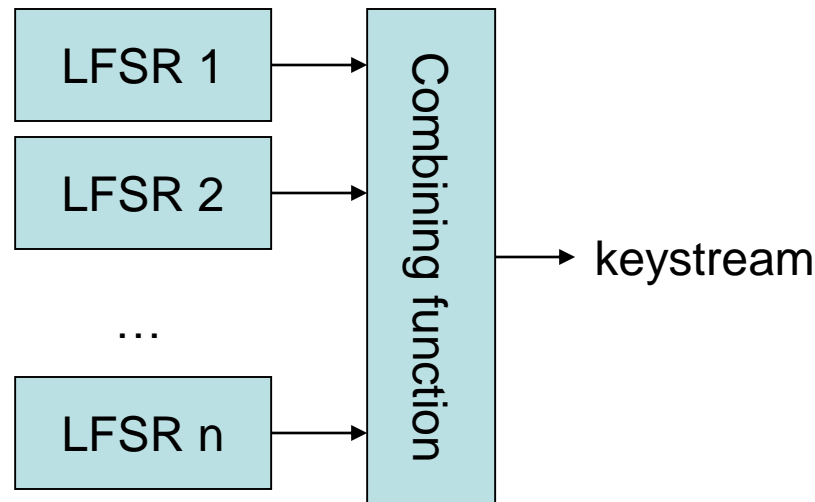


FSR based stream ciphers

- LFSR output is easily predictable
- To make the keystream secure:
 - Use nonlinear combination of LFSRs outputs
 - Use nonlinear filtering function on the LFSR output
 - Use LFSR output to clock more LFSRs
- Role of the key
 - Known LFSR connection: the key is the initial state of the LFSR(s)
 - Secret LFSR connection: the key is both the initial state of the LFSR(s) and the connection of the LFSRs
 - More security
 - More complex hardware

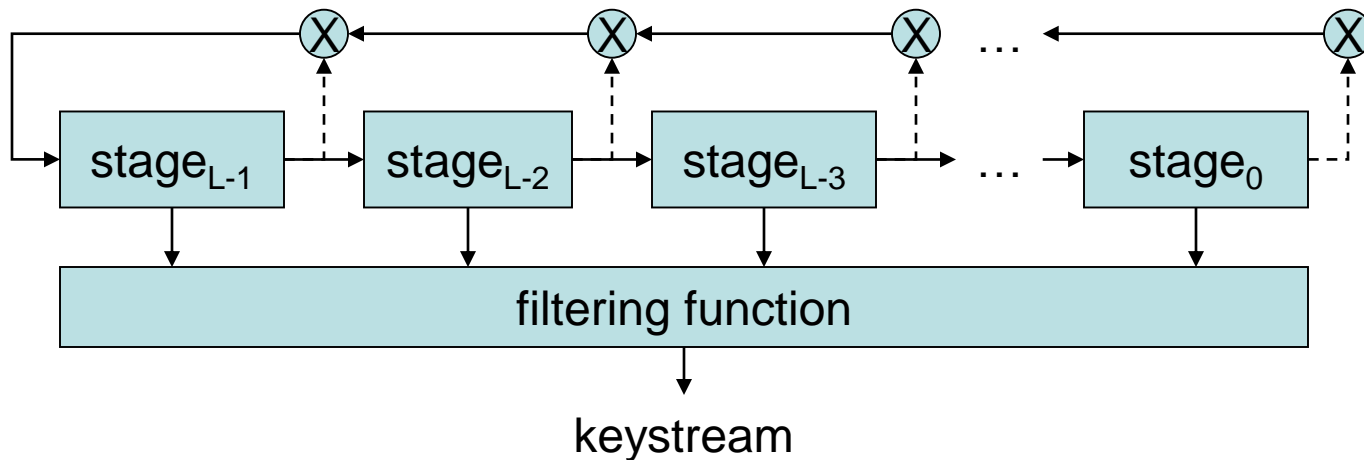
Nonlinear LFSR combinations

- Keystream is the nonlinear function of the LFSR outputs
 - Combining function



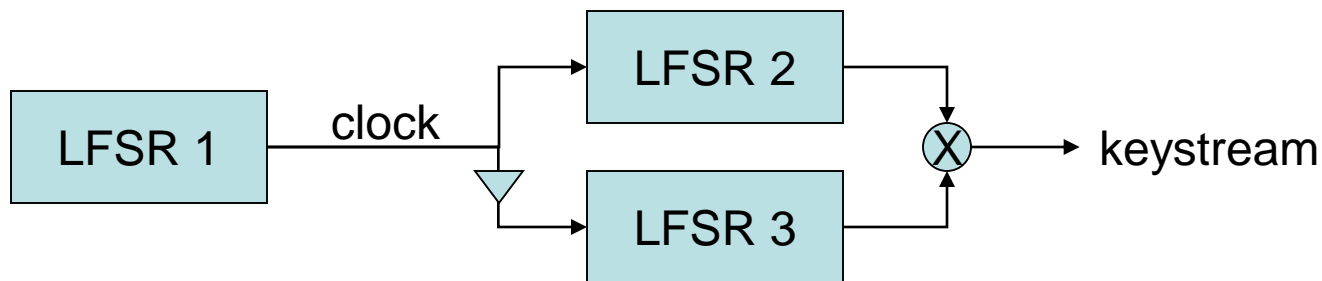
Nonlinear filtering function

- Keystream is a nonlinear combination of the states
 - Filtering function



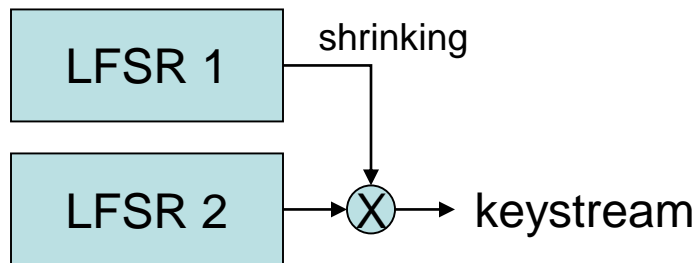
Clock-controlled generators

- Alternating step generator
- Security
 - Maximum length LFSRs L_1, L_2, L_3
 - L_1, L_2 and L_3 pairwise relatively prime
 - $L_1 \approx L_2 \approx L_3 \approx l$
 - Best known attack: 2^l steps



Clock-controlled generators 2.

- Shrinking generator
- Security
 - Maximum length LFSRs: L_1, L_2
 - L_1 and L_2 are relatively prime
 - Output period: $(2^{L_2-1}) \cdot 2^{L_1-1}$
 - $L_1 \approx L_2 \approx l$
 - Secret connections
 - Best known attack: 2^{2l} steps



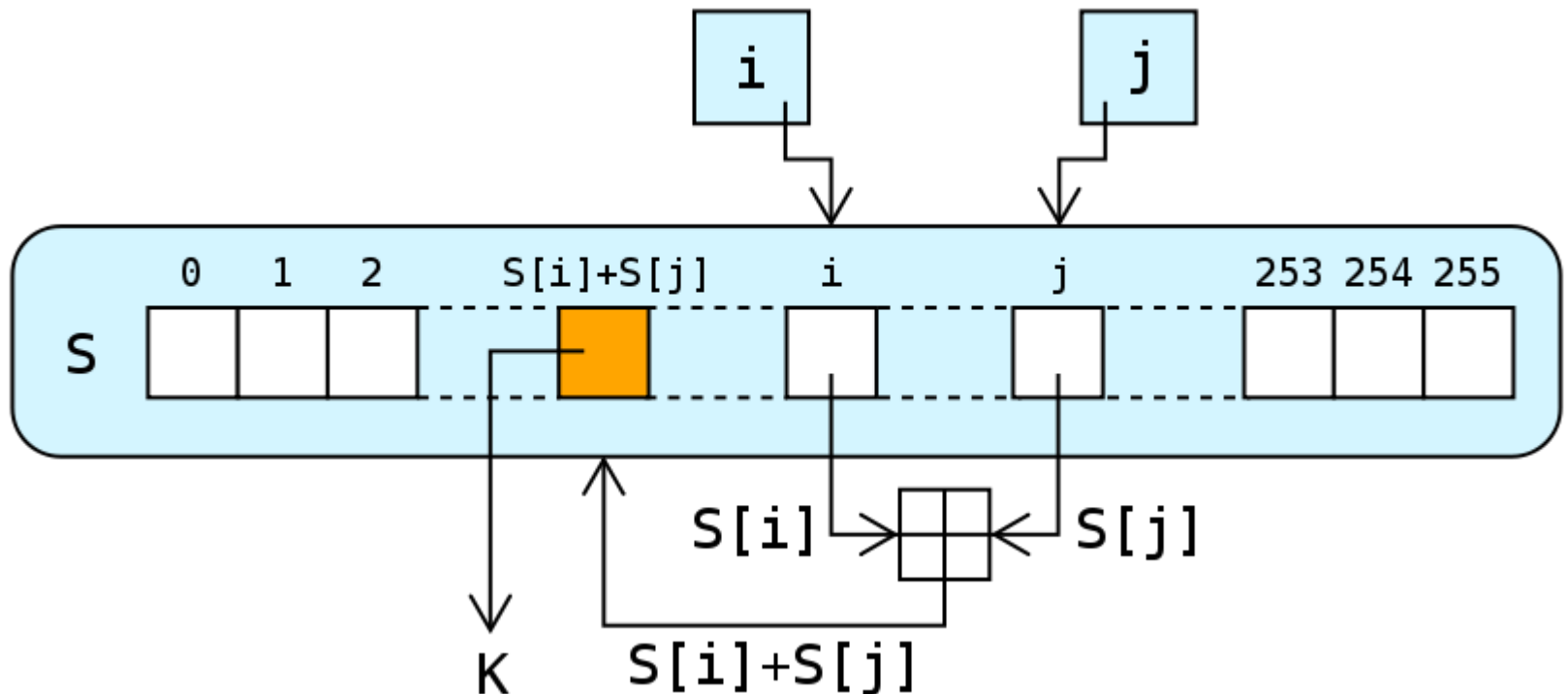
If shrinking = 0 then
discard LFSR 2 output

Software stream ciphers

- FSR based ciphers are for hardware ciphers, but not good for software ciphers
- There is a need for a fast software cipher
 - SEAL – Software Optimized Encryption Algorithm
 - RC4 – Ron's Code 4
 - SSL, TLS, WEP, TKIP
 - Block ciphers with OFB, CFB modes of operation

RC4 cipher

- RC4 – Ron's Code 4



RC4 code

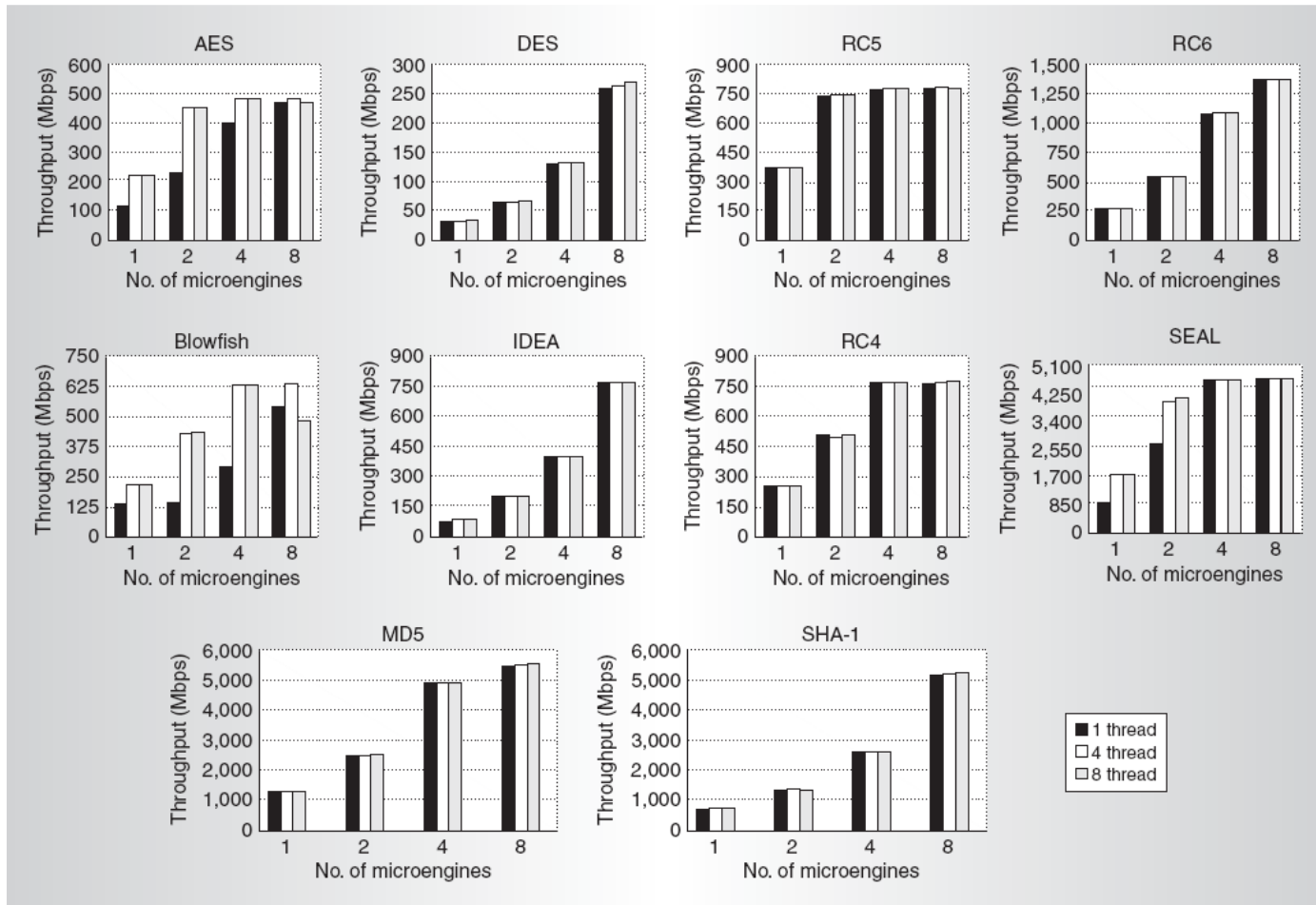
- Initialization

```
for i = 0 to 255:  
    Si = i  
j = 0; for i = 0 to 255:  
    j = (j + Si + Ki mod 1) mod 256  
    swap Si, Sj
```

- Keystream generation

```
i = (i + 1) mod 256  
j = (j + Si) mod 256  
swap Si, Sj  
Out = S[(Si + Sj) mod 256]
```

Speed of ciphers



Zhangxi Tan, Chuang Lin, Hao Yin and Bo Li: OPTIMIZATION AND BENCHMARK OF CRYPTOGRAPHIC ALGORITHMS ON NETWORK PROCESSORS

References

- Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, “Handbook of Applied Cryptography”, CRC Press, ISBN: 0-8493-8523-7
 - <http://www.cacr.math.uwaterloo.ca/hac/>
- Wikipedia - The free encyclopedia
 - <http://www.wikipedia.org/>