Sequences and Cryptography

Workshop on Shift Register Sequences
Honoring Dr. Solomon W. Golomb
Recipient of the 2016 Benjamin Franklin Medal in Electrical Engineering

Guang Gong

Department of Electrical and Computer Engineering

University of Waterloo

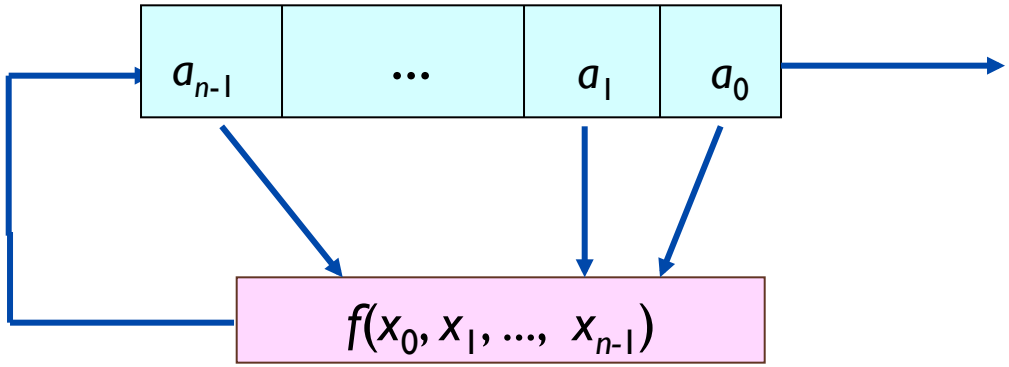
CANADA

<http://comsec.uwaterloo.ca/~ggong>

Outline

- Linear feedback shift register (LFSR) sequences
- Invariants and nonlinearity of boolean functions
- WG sequences and WG stream ciphers
- Some remarks

Feedback Shift Registers (FSR)



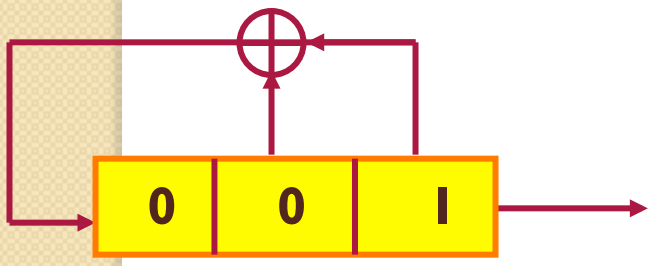
A Block Diagram of an FSR: *f* is a boolean function in *n* variables.

How does it work?

At each clock pulse: the state of each memory stage is shifted to the next stage in line, *i.e.*, there is a transition from one state to next.

Example. A 3-stage LFSR with a feedback function

$$f(x_0, x_1, x_2) = x_0 + x_1$$



Initial state: $(a_2, a_1, a_0) = (0, 0, 1)$

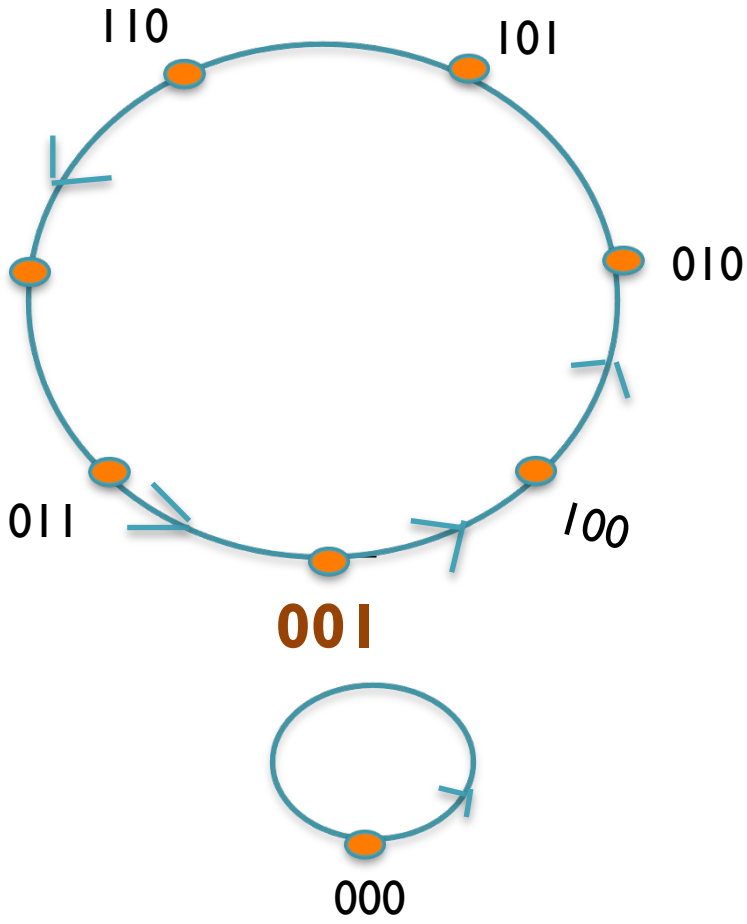
Recursive relation:

$$a_{3+k} = a_{1+k} + a_k, k = 0, 1, \cdots$$

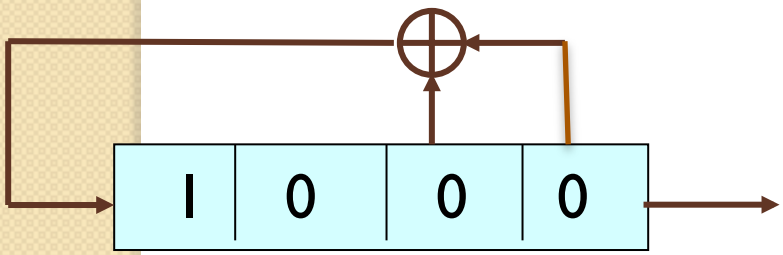
Output sequence:

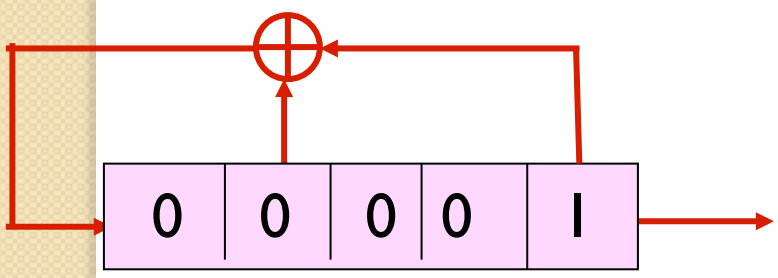
1 0 0 1 0 1 1 1 0 0 1 0 1 1 ….

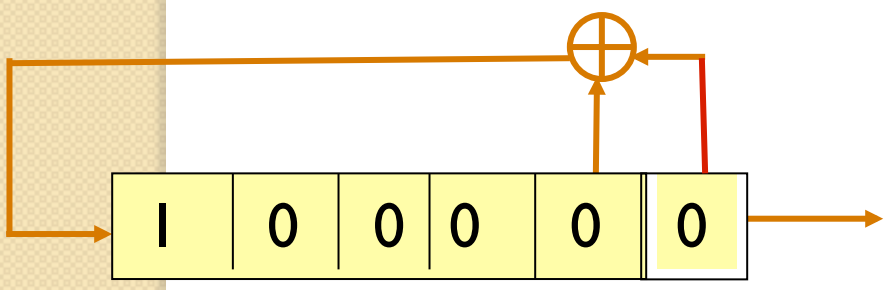
State Diagram



More examples of LFSRs







M-sequences: generated by an LFSR with the maximum period.

Output an *m*-sequence with period 15

000**1**0 0**1**1**0**1 0**1**1**1**1

Output an m-sequence of period 31

1 0 0 0 0 **1** 0 **1** 0 **1**

1 **1** 0 **1** **1** 0 0 0 **1** **1**

1 **1** **1** 0 0 **1** **1** 0 **1** 0 0

Output an m-sequence of period 63

0 0 0 0 0 **1** 0 0 0 0 **1** **1** 0 0 0 **1** 0 **1** 0 0 **1**

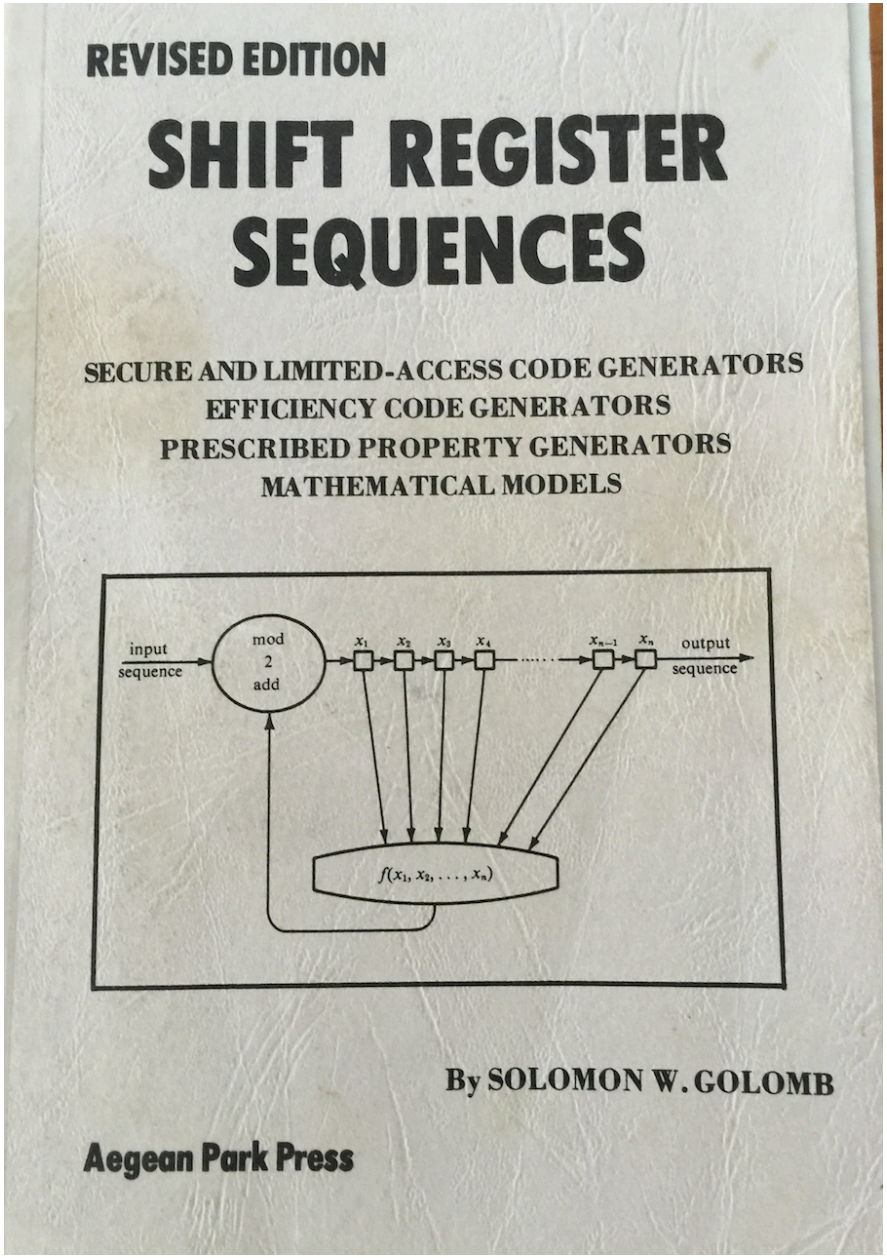
1 **1** **1** 0 **1** 0 0 0 **1** **1** **1** 0 0 **1** 0 0 **1** 0 **1** **1** 0

1 **1** **1** 0 **1** **1** 0 0 **1** **1** 0 **1** 0 **1** 0 **1** **1** **1** **1** **1**

How to generate *m*-sequences?

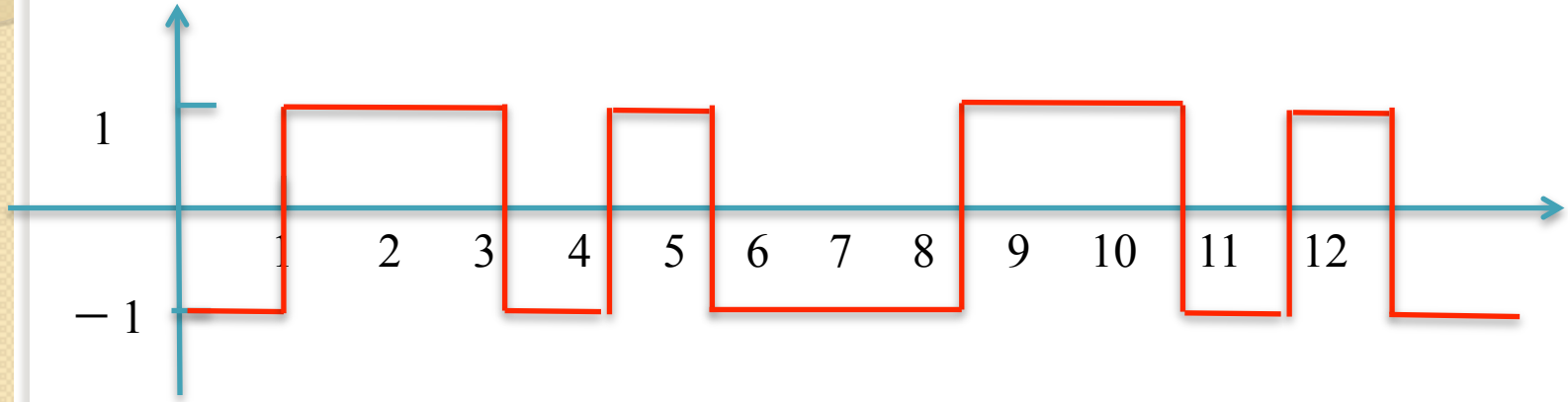
- **Result (Golomb, 1954)**

If a feedback corresponds to a primitive polynomial, then it generates an m-sequence. It collected in Dr. Golomb's book, published in 1967.

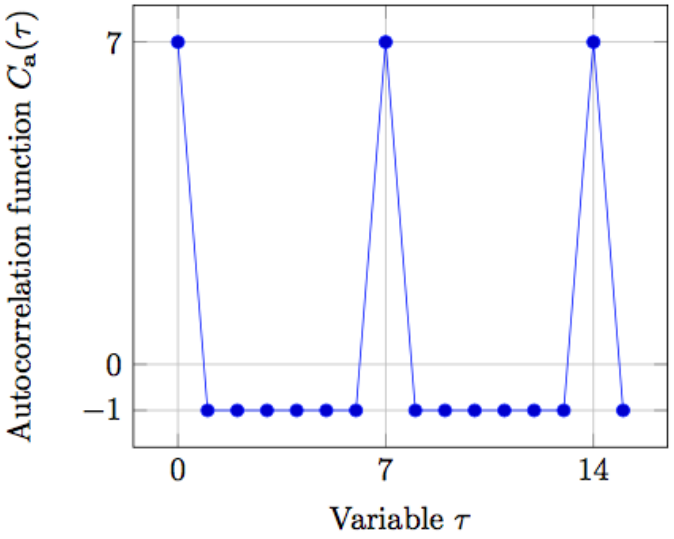


Autocorrelation

- M-sequence of period 7: 1001011
- Signal pulse: $0 \leftrightarrow 1, \quad 1 \leftrightarrow -1$



- Out-of-phase autocorrelation is equal to – 1.



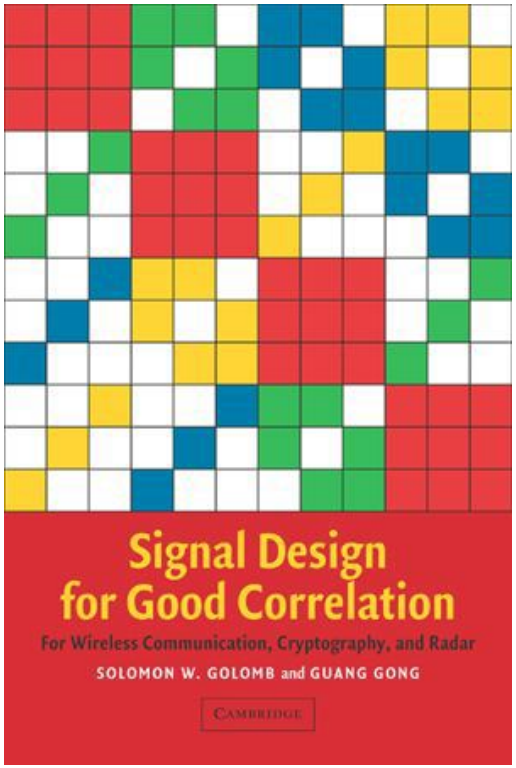
What are the pseudorandomness properties of binary m-sequences?

Period	$2^n - 1$
Golomb R1. Balance	Difference between number of 1's and 0's is 1
Golomb R2. Runs	Each consecutive of 1's or 0's occurs equally likely except for the length $n-1$ and n.
Golomb R3. Correlation	All out-of-phase autocorrelation is equal to -1
Span n property	Each nonzero n-tuple occurs once
Linear span	LFSR with shortest length: n

Pseudorandomness as good as it could be!

Too small to be secure for crypto app!

- Are there other sequences with the same 2-level autocorrelation as m-sequences?
- The answer is a YES!
  For more about those 2-level autocorrelation sequences, see Golomb and Gong's book.
- There is a trade-off between autocorrelation and span n property.



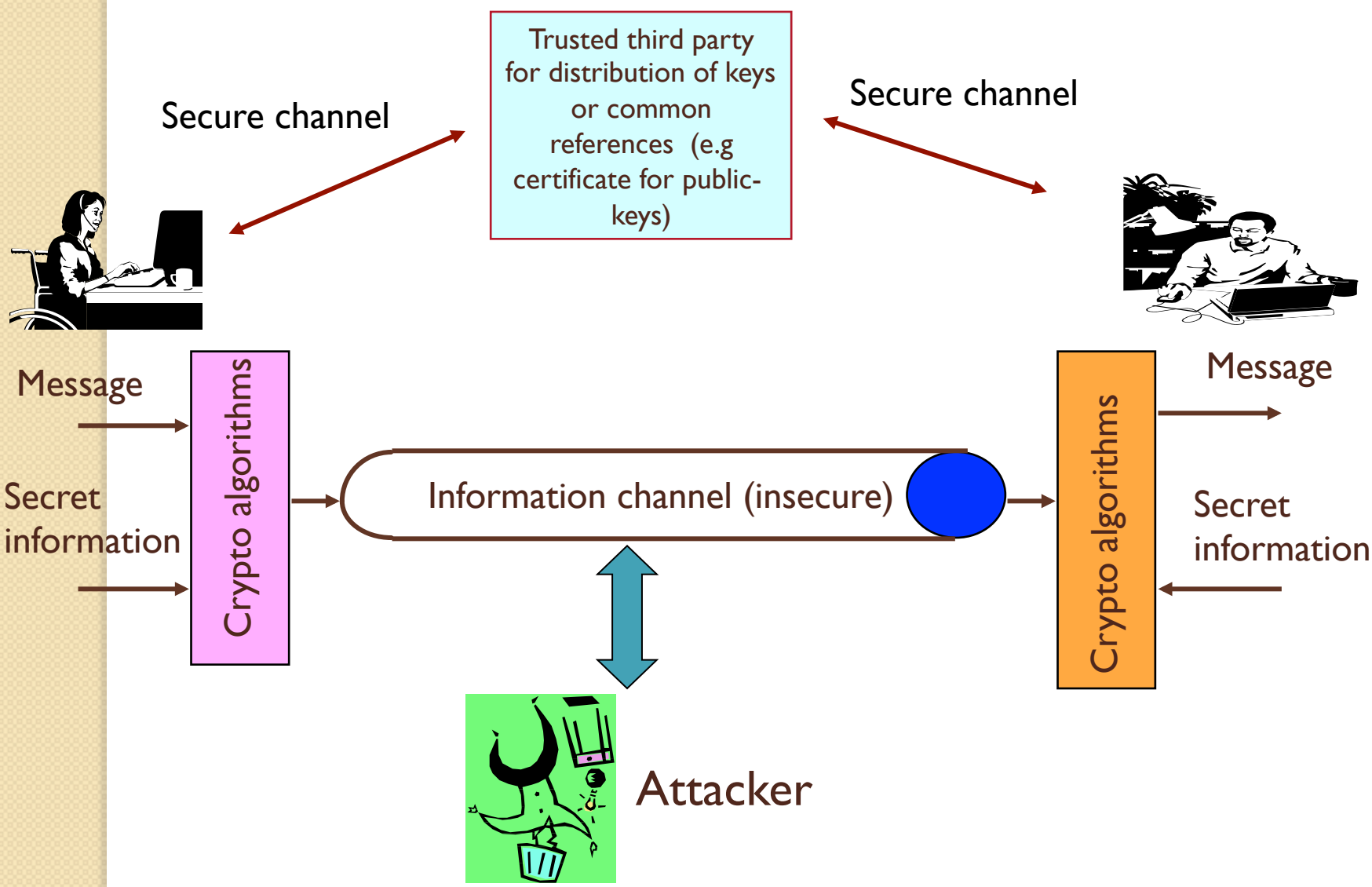
Golomb Conjecture (1980, open): Any sequence with those two properties must be an m-sequence.

Significance in crypto: A sequence with large linear span has to compromise one of those two properties!

Applications of pseudo-random Sequences in crypto

- Key stream generators in stream cipher
- Pseudorandom functions in block ciphers
- Session key generators and key deviation functions (KDF)
- Pseudo-random number generators (PRNG) in Digital Signature Standard (DSS), etc.
- Digital water-mark
- Hardware tests for crypto processors
- Masking sequences for anti side-channel attacks
- ….

Model of Security Communication

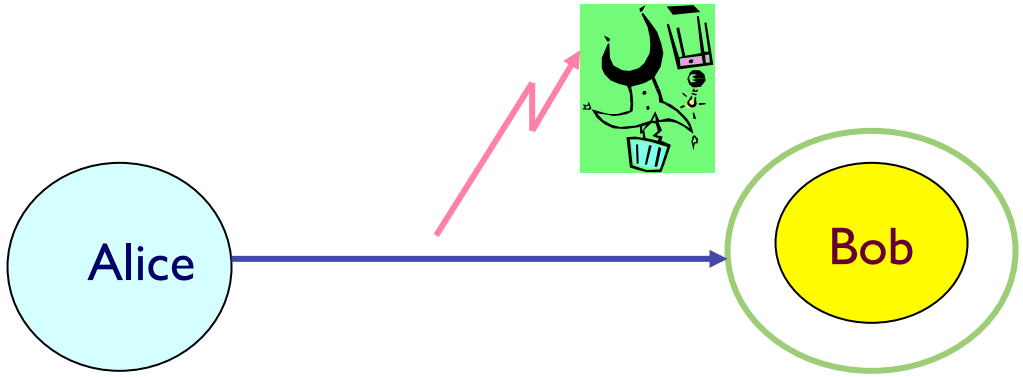


Historical Remarks

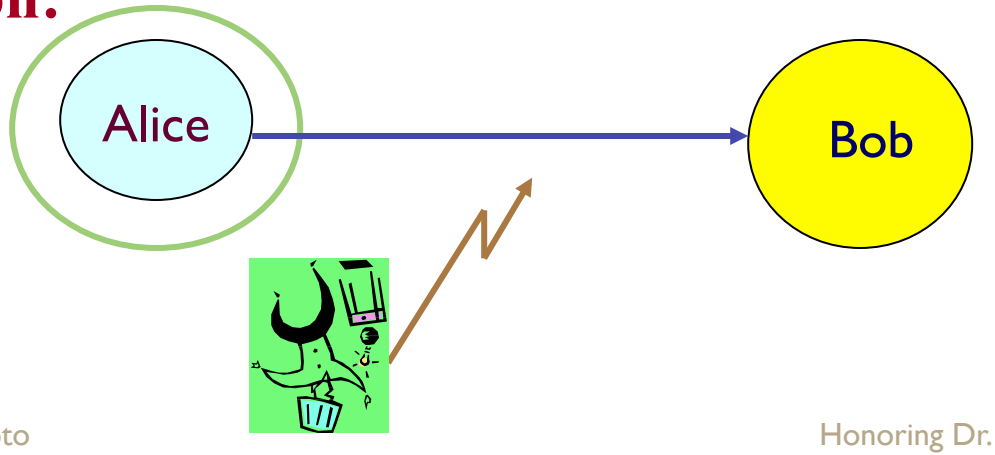
Cryptography, defined as the study of "mathematical" systems for solving two kinds of security problems:

privacy and authentication

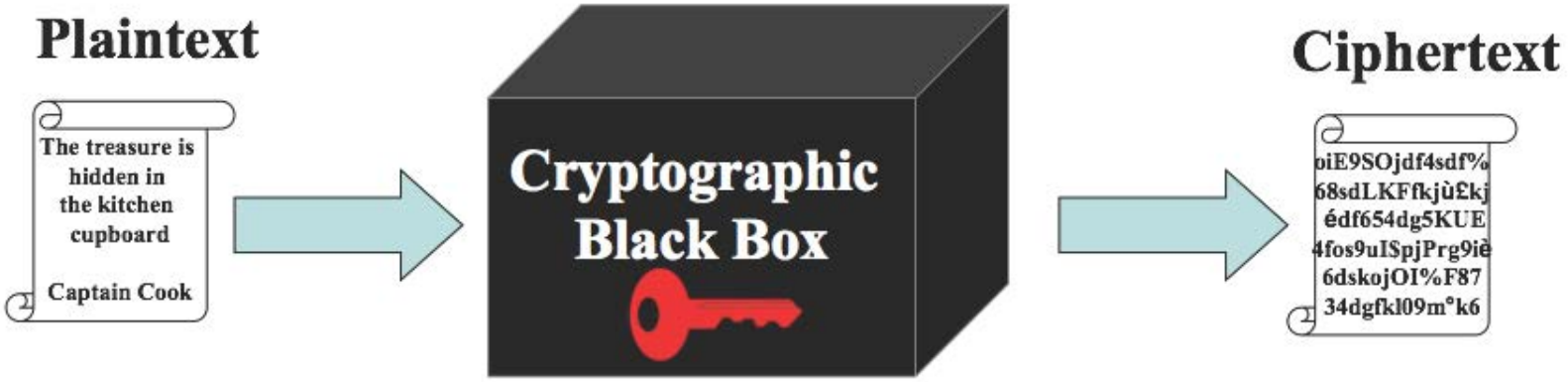
Privacy:

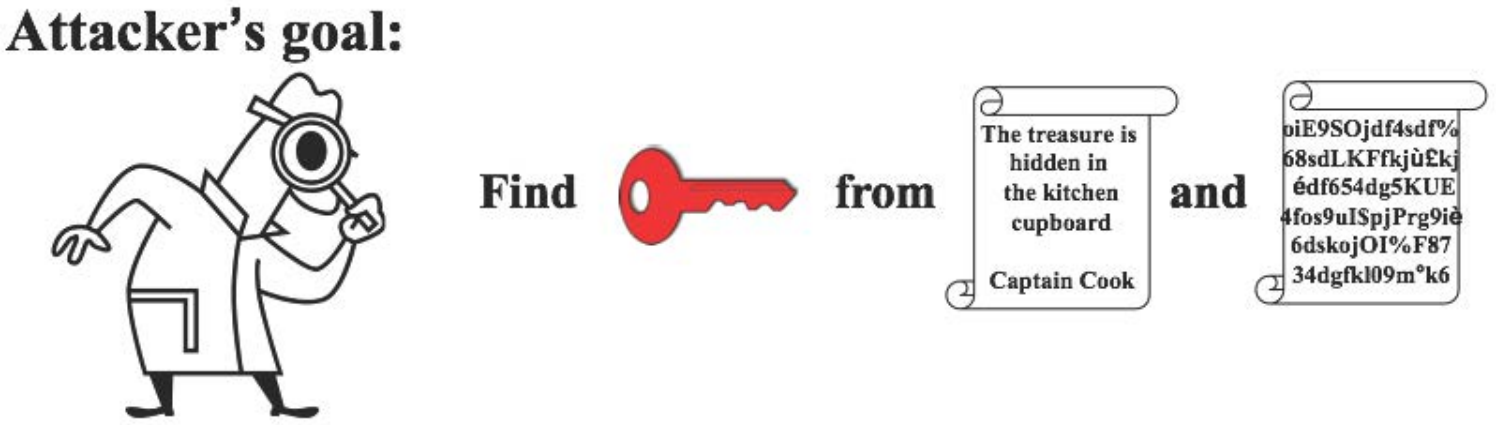


Authentication:

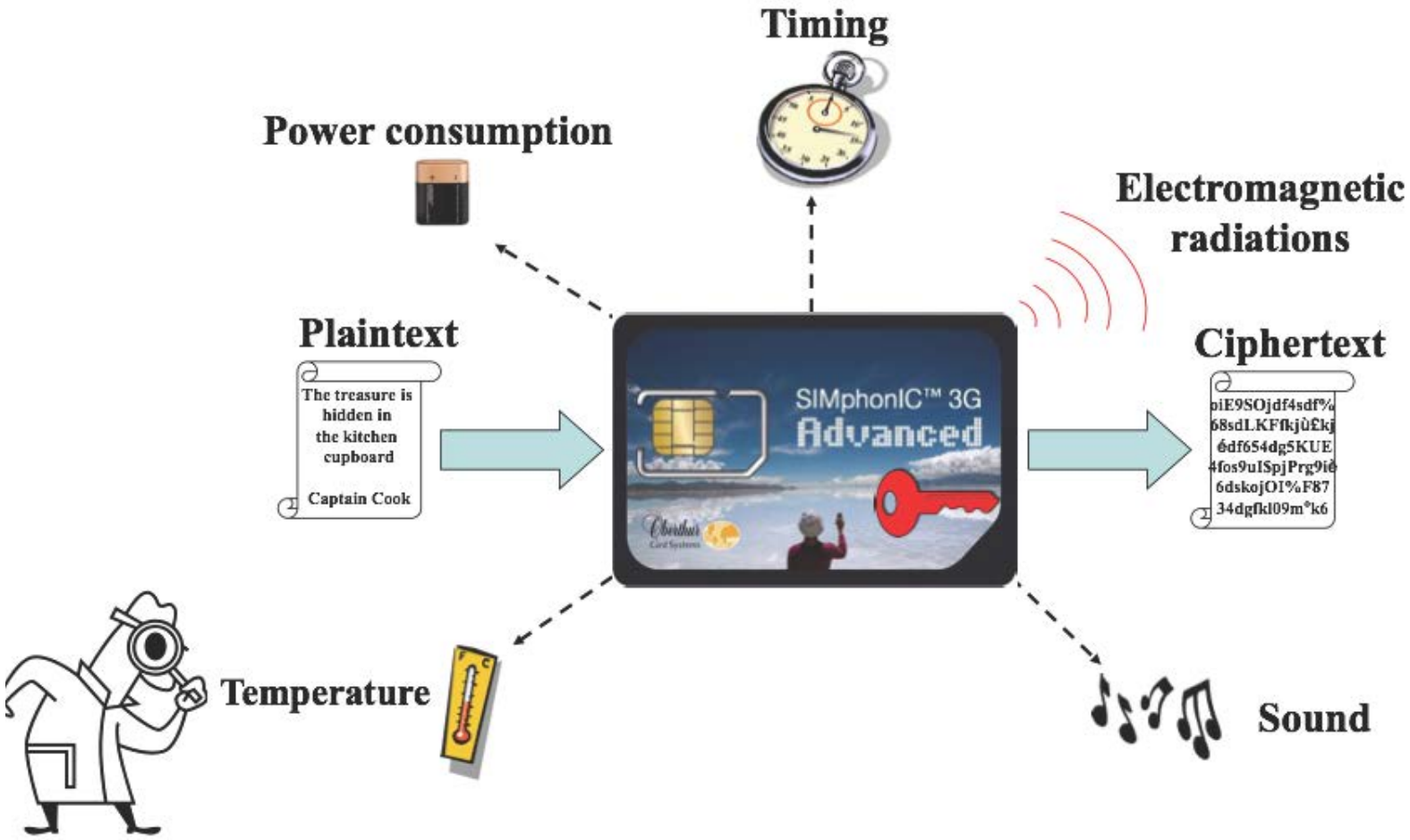


What are threats? - Cryptanalysis



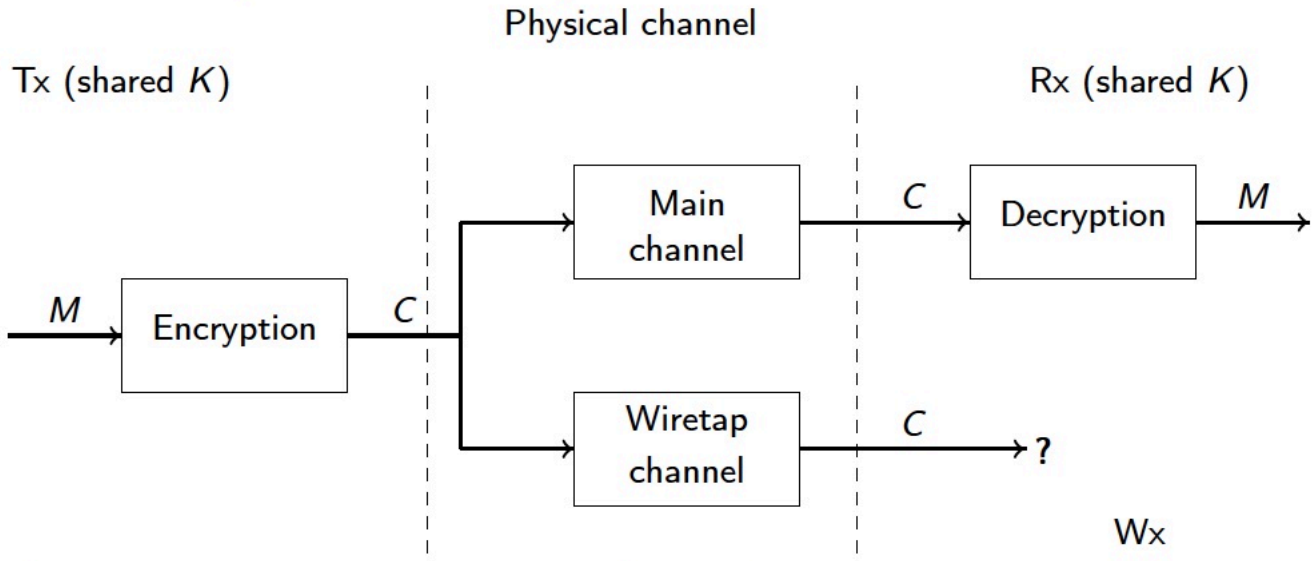


Crypto algorithms may also be attacked by side-channel information



How to measure the strength of crypto algorithms?

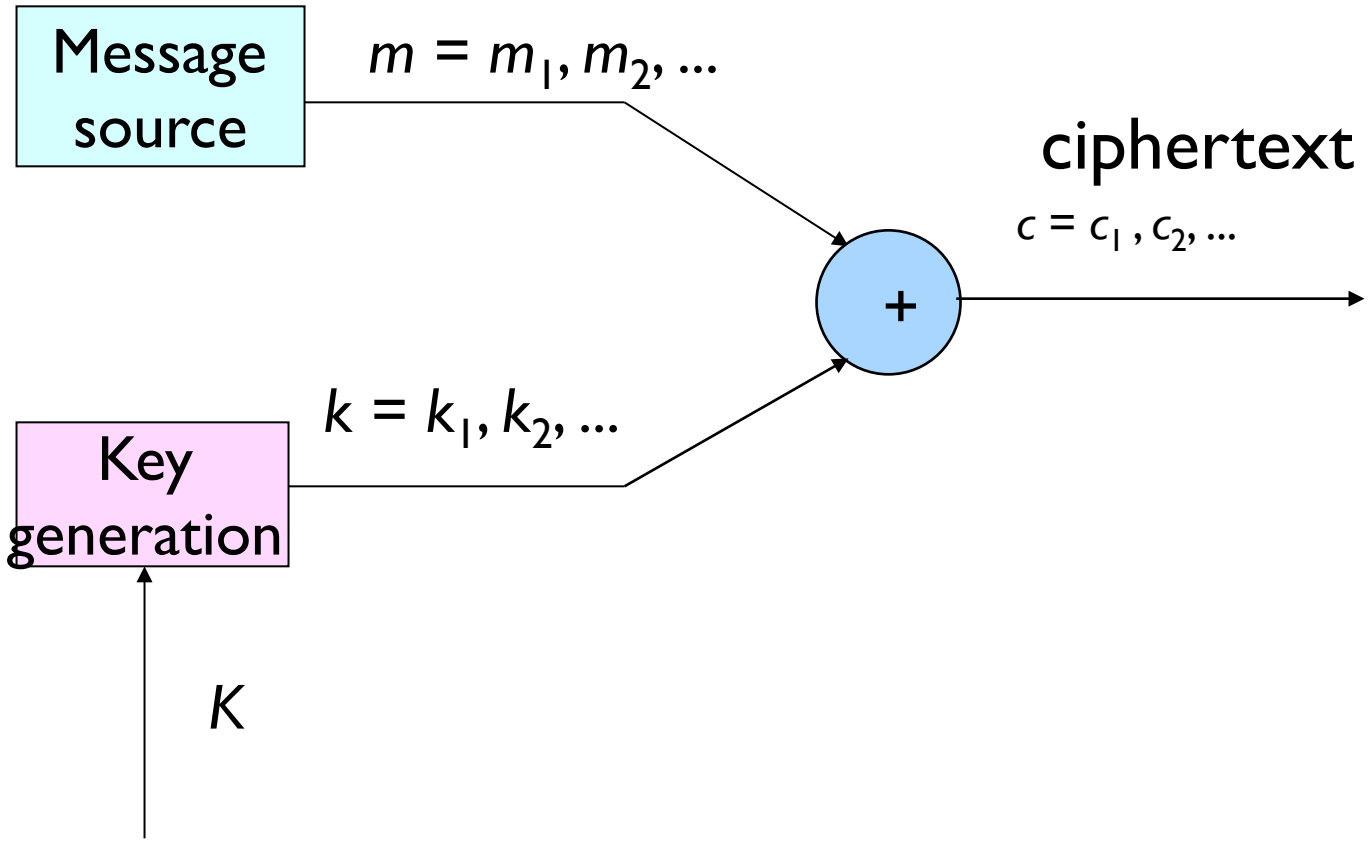
Shannon's perfect secrecy



where Tx=transmitter, Rx=receiver, and Wx=adversary or wiretapper.

A cipher system has perfect secrecy if plaintext M, treated as a random variable, is independent of ciphertext C for any key K, i.e., Pr(M) = Pr (M|C) (Shannon 1948).

Model of Stream Cipher



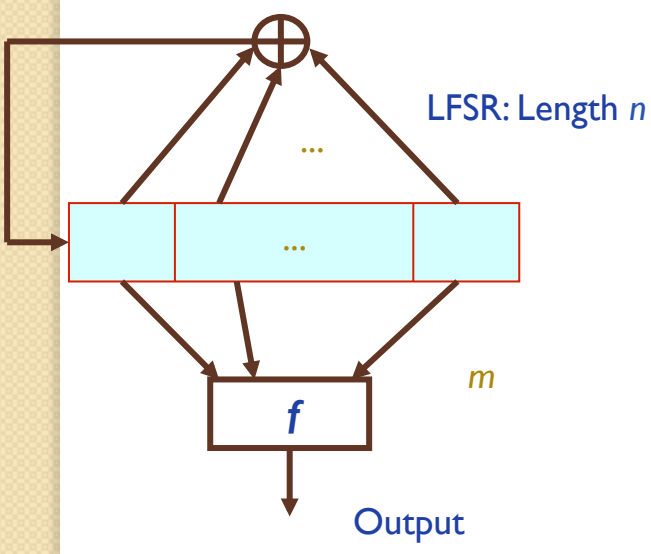
Stream cipher and one-time-pad

- One-time-pad means that different messages are encrypted by different key streams in stream cipher model.
  - Shannon (1948): One-time-pad has perfect secrecy. This requests a key stream has a large period.
  - → Use of m-sequences generated by n-stage LFSRs, since they have period $2^n - 1$.
  - E.g., $n = 100$, the period is $2^{100} - 1$!

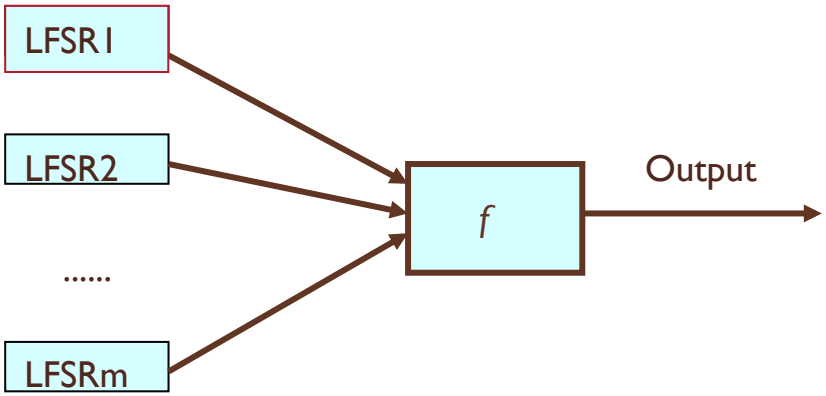
How about a random bit stream?

- **Question:** Given a random bit stream: 0010000101110 …, can one find an LFSR to generate the sequence?
  - Berlekamp's result in coding context (1968, and Massey used it in LFSR 1969): knowing 2*n* consecutive bits of a sequence with linear span *n*, the rest of bit of the sequence can be reconstructed.
  - Applying this result to an m-sequence with period 2^{100}-1, the attacker only needs to know 200 consecutive bits, then the rest of bits can be reconstructed!
  - This ends the monopoly life of LFSRs used as key stream generators (1954-1969).

What should be used to generate pseudorandom sequences? → Change the config of LFSRs!

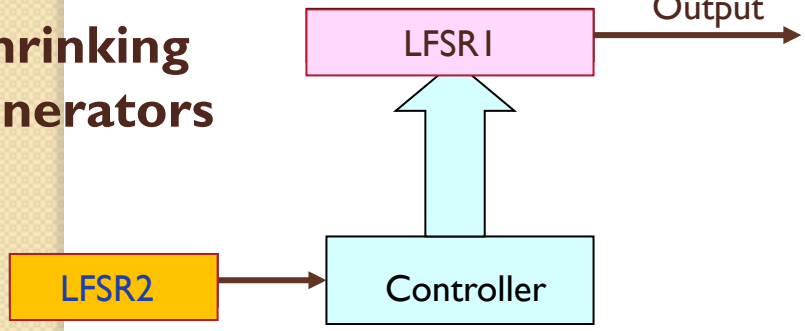


A Filtering Generator



A Combinatorial Function Generator

Clock-control / Shrinking Generators



- Those configurations are used as key stream generators since 1970. Their crypto strength is dominated by function *f*.

How to measure the crypto strength of those filtering functions?

- Golomb (1959): Invariants of a boolean function, which measure the distances between the boolean function and linear combinations of its inputs.

Golomb, IRE Transactions on Information Theory, May 1959.

ON THE CLASSIFICATION
OF BOOLEAN FUNCTIONS

Solomon W. Golomb
Jet Propulsion Laboratory
Pasadena, California

1. Introduction

The Boolean functions of k variables, $f(x_1, x_2, \ldots, x_k)$, fall into equivalence classes (or families) when two functions differing only by permutation or complementation of their variables are considered equivalent. The number of such families is easily computed, as illustrated by Slepian [1]. The next step is to discover the invariants of the logic families, and determine to what extent they characterize the individual families. Given the class decomposition, one also wishes to select a "representative assembly", with one delegate from each family. That is, canonical forms for the logics are sought, with every family having its characteristic canonical form. This mathematical program will be carried out in Sections 2 through 6.

Given certain of the invariants, it is possible to say something about the size of the corresponding family. Applications of this principle are explored in Section 7.

The practical significance of the symmetry classes is that a circuit which mechanizes a given function f will also mechanize any other function of the same class, provided that complements of all the inputs are available, simply by permuting and complementing the inputs. In particular, the extensive investigations on the minimization of logical circuitry can be confined, without loss of generality, to one representative function in each symmetry class.

Example. The two Karnaugh charts of Figure 0 represent Boolean functions belonging to the same symmetry class. In particular, if the function at the left is designated f(v,x,y,z), then the function on the right is f(x,v',z',y).

Note: The group G of allowable symmetries contains k! permutations of the variables, 2^n complementations of the variables, and a two-fold choice of the constant b. Thus G consists of $2^{k+1} \cdot k!$ operators. In particular, the size of any family of equivalent Boolean functions is some factor of $2^{k+1} \cdot k!$ Since there are 2^{2^k} Boolean functions of k variables, the number of families is approximately $(2^{2^k})/(2^{k+1} \cdot k!)$.

Theorem 1: The quantity $T_o = \max(c_o, 2^k - c_o)$, where $c_o = \sum_{x_1,x_2,\ldots,x_k} f(x_1,x_2,\ldots,x_k)$, is an invariant for the family containing $f(x_1,x_2,\ldots,x_k)$. (The capital sigma denotes ordinary summation.)

Proof: The quantity c_o is the sum of all the entries in the truth table for $f(x_1,x_2,\ldots,x_k)$. Permutation and complementation of the variables rearranges the entries in the truth table, but leaves their sum c_o unaltered. Complementing $f(x_1,x_2,\ldots,x_k)$ replaces c_o by $2^k - c_o$, which means that half the operations of G leave c_o fixed, and the other half replace c_o by $2^k - c_o$. Thus $T_o = \max(c_o, 2^k - c_o)$ is invariant under all operations of G.

Corollary: Relative to the subgroup H of G which allows permutation and complementation of the variables but forbids complementation of the function (so that $[G/H] = 2$), the quantity c_o is itself an invariant of the equivalence classes.

Def. T_o is the zero order invariant of the logic family for the group G. (Likewise c_o is the zero order invariant for the group H.)

Theorem 2: For every i, $1 \leq i \leq k$, let $R_1^i = \max(c_1^i, 2^k - c_1^i)$, where $c_1^i = \sum_{x_1,x_2,\ldots,x_k} (f(x_1,x_2,\ldots,x_k) \oplus x_i)$. Then the set of numbers $R_1^1, R_1^2, \ldots, R_1^k$, when rearranged in descending order, forms a collection of k invariants $T_1^1, T_1^2, \ldots, T_1^k$ for the family to which f belongs. (The symbol $\oplus$ denotes modulo 2 addition.)

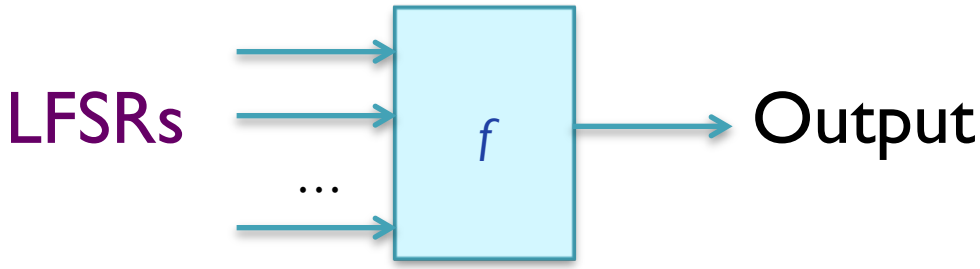
3. The Higher Order Invariants

Def. For every pair (i,j) with $1 \leqq i < j \leqq k$, define $c_2^{ij} = \sum_{x_1,x_2,\ldots,x_k} (f(x_1,x_2,\ldots,x_k) \oplus x_i \oplus x_j)$.

The complement of c_2^{ij} is $2^k - c_2^{ij}$. That ordering and complementing of variables and complementation of the function f, consistent with producing the $\{R_1^i\}$ in descending order, which makes the sequence of $\{c_2^{ij}\}$ numerically greatest, gives the second-order invariants $T_2^{1,2}, T_2^{1,3}, \ldots, T_2^{2,3}, \ldots, T_2^{k-1,k}$.

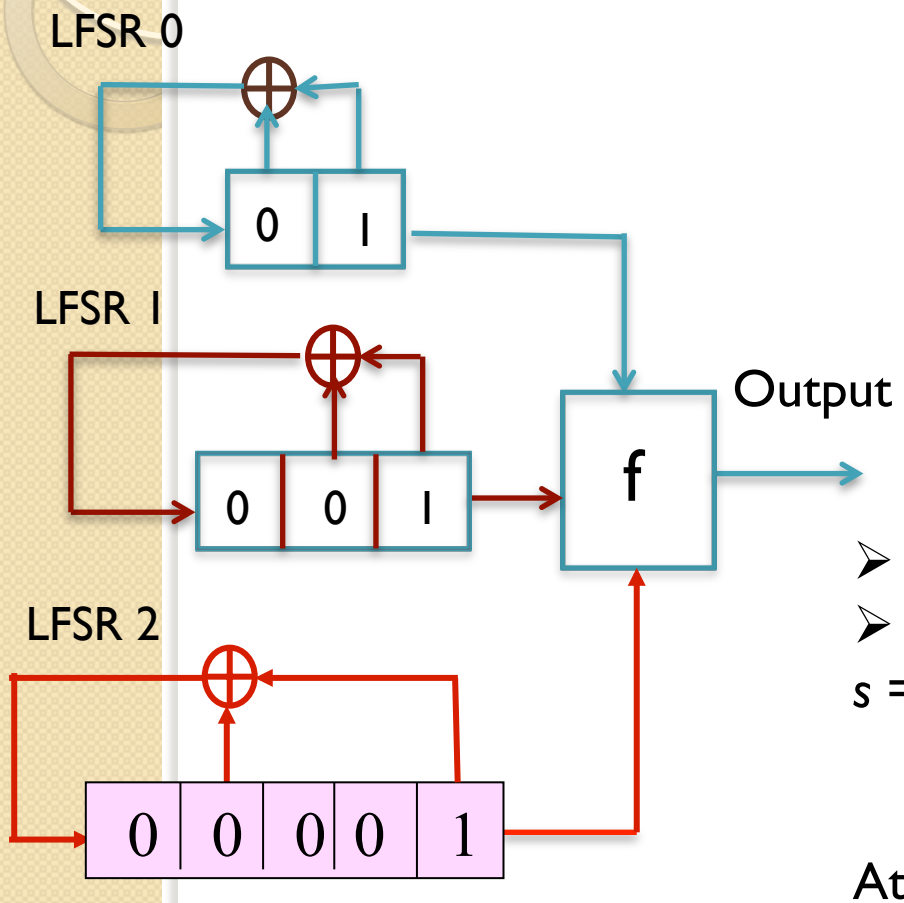
What follows?

- Inputs from LFSRs (single or multiple)



- The boolean function *f* should be far or independent from input variables or linear combination of input variables!
- This can be measured by **invariants** under Golomb's term and termed as **nonlinearity** in modern cryptography!
- The results presented by Golomb in 1959 is rediscovered by Xiao and Massy in 1988.

E.g. Correlation attack: application of Golomb's invariances



$$f(x_0, x_1, x_2) = x_0 + x_0x_1 + x_1x_2$$

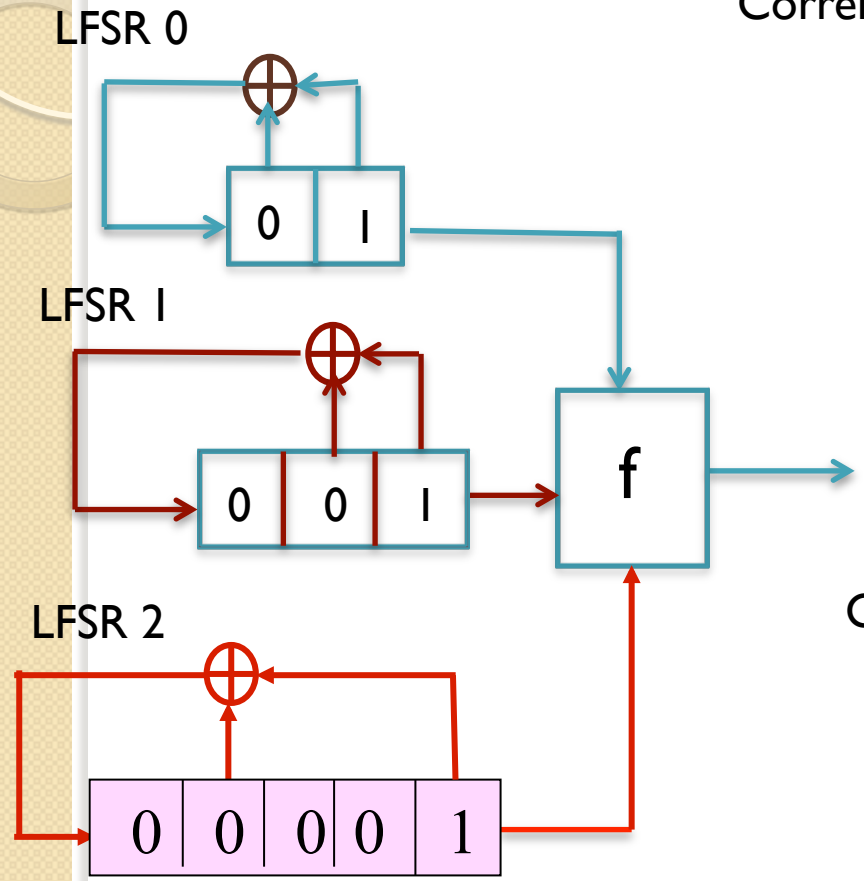
- This boolean function is correlated with input variables x_0 and x_2.
- Suppose it is used as a key stream generator and a 10-bit key is loaded as initial states of three LFSRs.

- Attacker recovered 3
- 40 bits of the output bits:

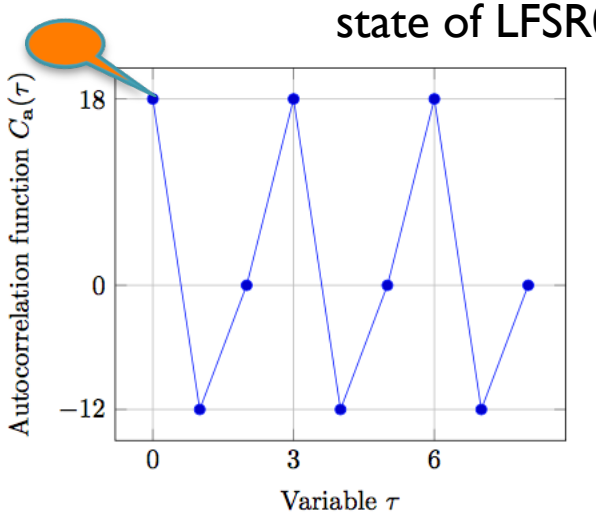
s = 0 1 0 0 0 0 1 1 1 0 1 1 1 0 1 0 0 1 0 1
1 0 1 1 0 0 1 1 1 1 0 1 1 0 0 1 0 1 1 1

Attacker's goal: recover the 10-bit key, i.e., the initial state of each LFSR.

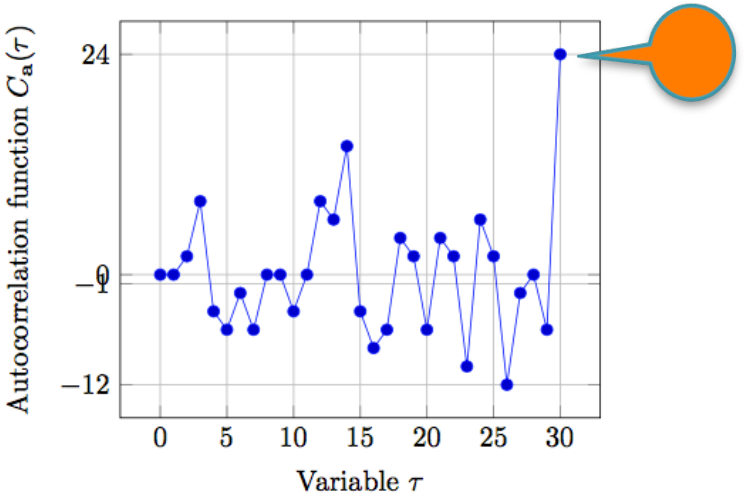
Since f is correlated with LFSR0 and LFSR2, one can compute the correlation between the known 40-bits with LFSR0 and LSFR2 respectively.



Correlation with LFSR0 → decode the initial state of LFSR0 as 10

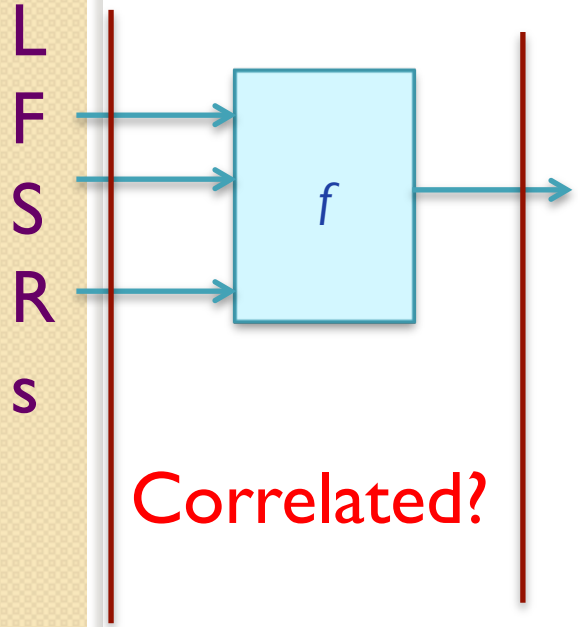


Correlation with LFSR0 → decode the initial state of LFSR2 as 01000



How to get the initial state of LFSR 1?
Exhaustive search! But this complexity is much smaller than exhaustive search for all.

Some remarks on invariants



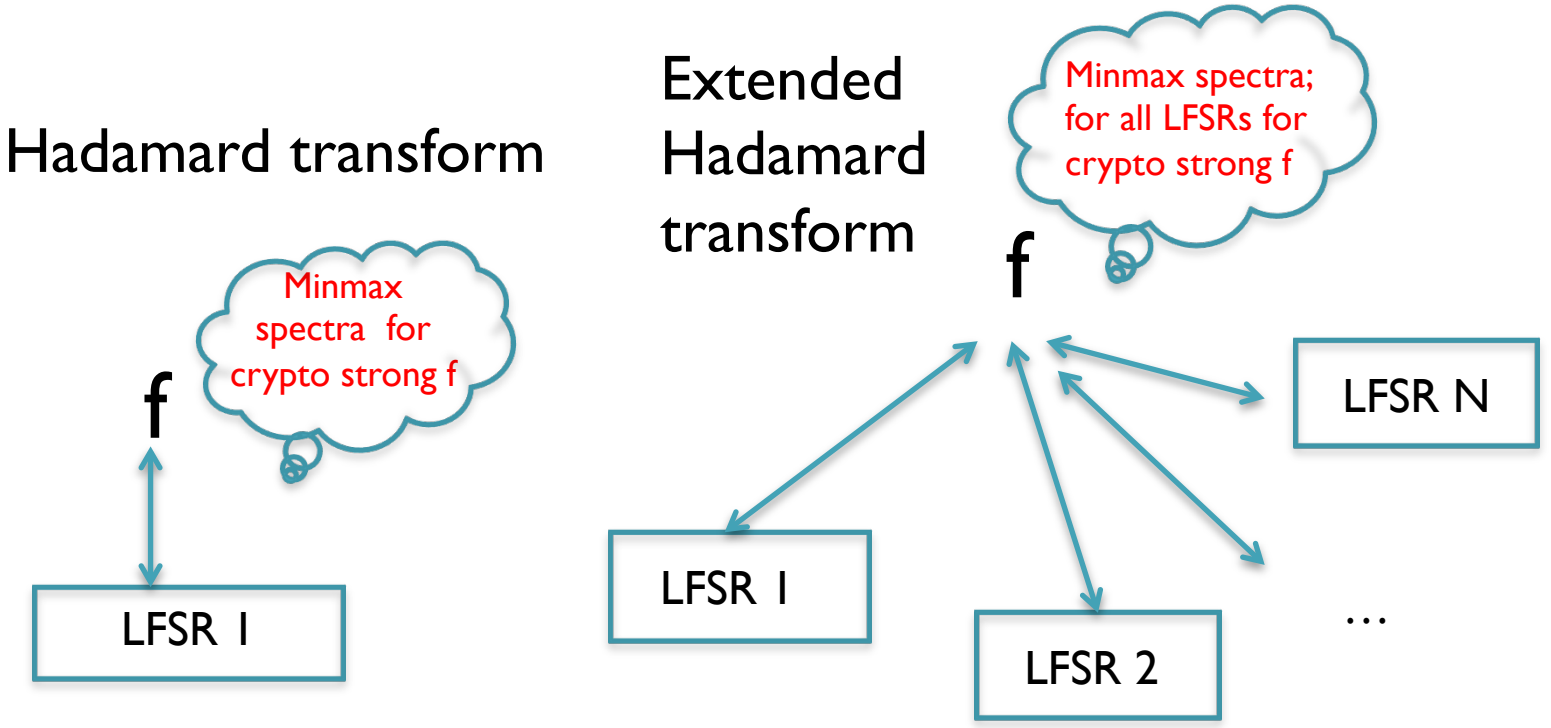
- The applications of Golomb's work in 1959 is to design *f* for the distance ranging problem using LFSRs with short periods to get a sequence with large period.
- So, *f* should be correlated with input variables!
- But in crypto, it should be uncorrelated.

Invariants, nonlinearity, and Hadamard transform

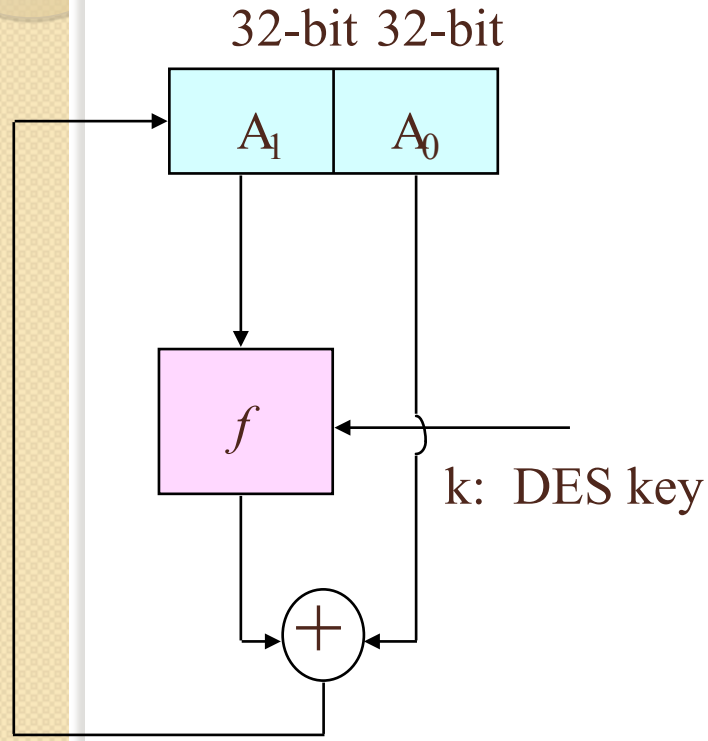
- Golomb's invariants or nonlinearity of boolean functions can be computed through Hadamard transform.
- Those three metrics measure the correlation between a sequence and an m-sequence.
- However, there are a number of distinct LFSRs, corresponding to the number of primitive polynomials, which generates distinct *m*-sequences with the same period.

Extended Hadamard transform (Gong-Golomb, 1999)

- It makes the sense that the crypto strength should be measured from all distinct LFSRs instead of a single LFSR!

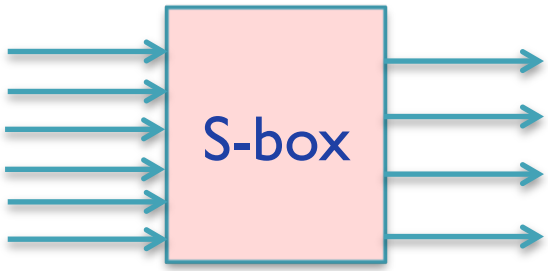


Spectral analysis of DES (G and Golomb, 1999)



- DES (Data Encryption Standard, NIST 1976)
- It can be viewed as an NLFSR with input k
- The feedback function f consists of 8 S-boxes each with 6-bit input and 4-bit output.
- The 32-bit input to f is first extended to 48 bit.
- The 64-bit plaintext is loaded as an initial state, then it clocks 16 times without output.
- The ciphertext is the 17th state of the NLFSR.

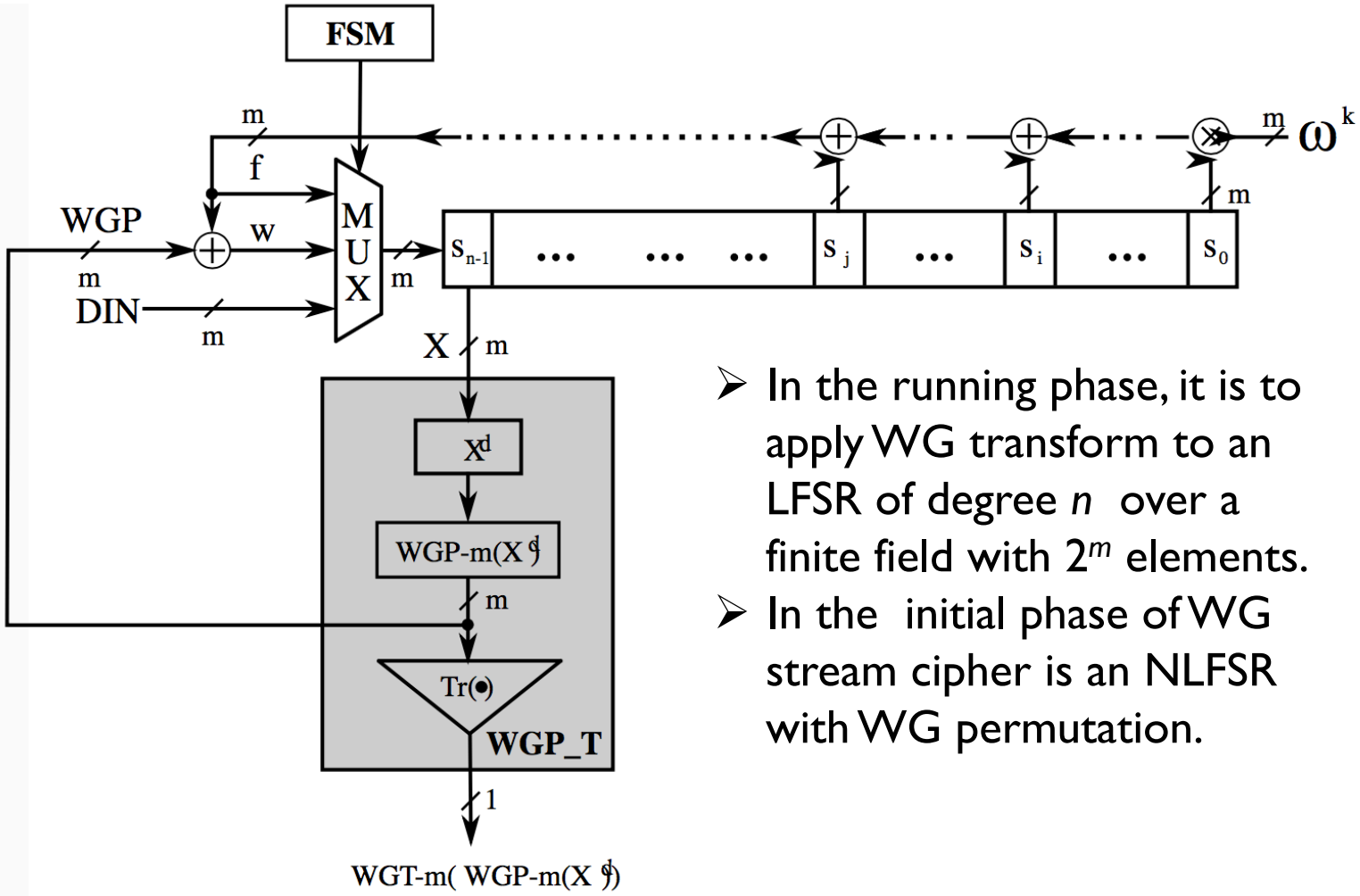
Co-spectral Property of S-boxes in DES (G. and Golomb 1999)



- Each output of an S-box can be considered as a boolean function in 6 variables.
- There are 6 distinct LFSRs with degree 6.

- Each of 32 outputs of 8 S-boxes has the same spectra under the extended Hadamard transform.
- So, S-boxes in DES have good crypto properties, which are the currently only known class with this property except for hyper bent functions, discovered in 2001.

WG stream cipher



- In the running phase, it is to apply WG transform to an LFSR of degree n over a finite field with 2^m elements.
- In the initial phase of WG stream cipher is an NLFSR with WG permutation.

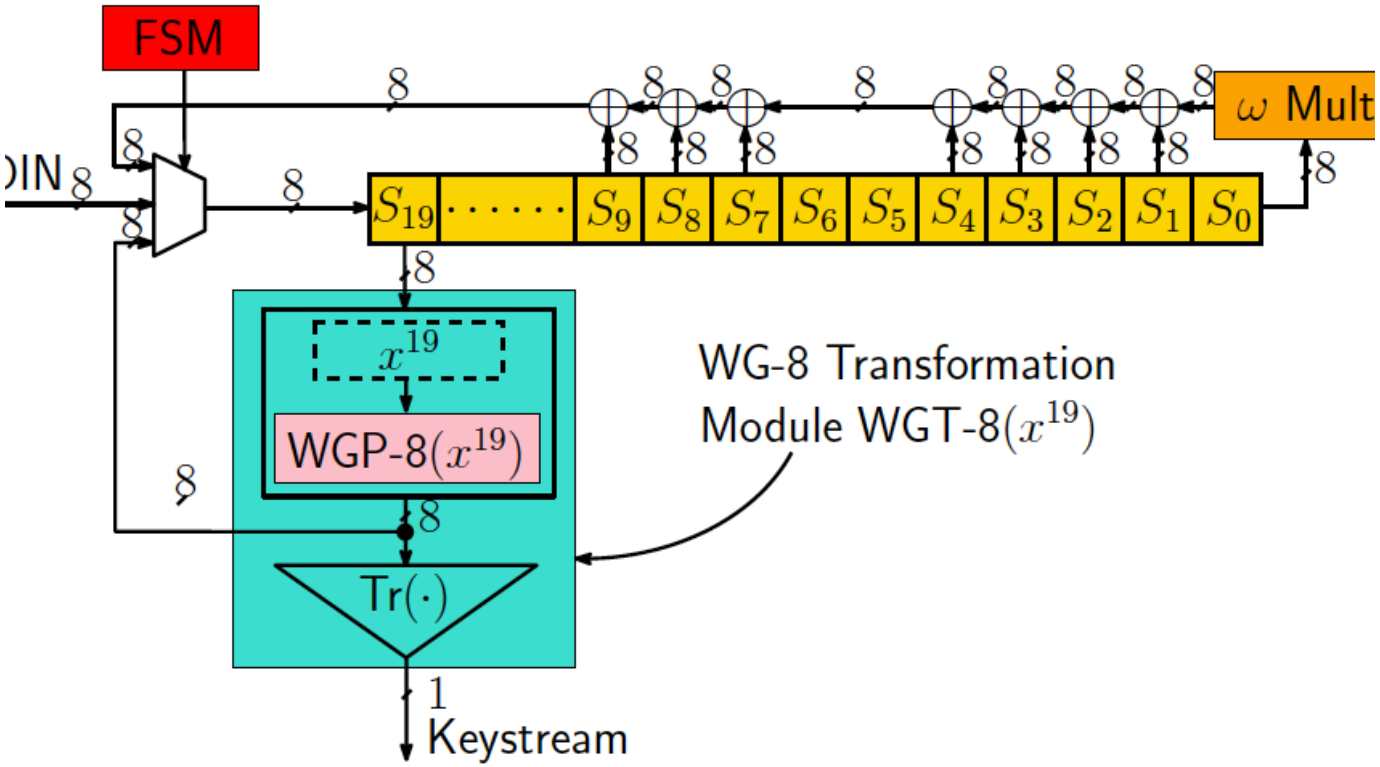
WG transformation sequences

- WG transformation sequences are discovered in 1997 by Gong, Golomb and Gaal, which are conjectured that there are infinite many such sequences with 2-level autocorrelation, and verified their result up to period 2^{23}-1.
- Dr. Golomb named it as Welch-Gong (WG) transformation sequences!

Some remarks on WG sequences and WG cipher

- A few month later after the discovery of WG sequences, No etc., found another representation of WG and also verified their result for the same period.
- In 1999, Dillon proved the result for odd case.
- In 2005, Dobbertin and Dillon proved the result for even cases in their milestone work. They also proved the validity of all the conjectured 2-level autocorrelation sequences.
- In 2003, Gong and Youssef showed cryptographic properties of WG transformations.
- In 2005, Yassir and G submitted WG cipher to ESTREAM competition.
- WG cipher is the only cipher currently known whose randomness properties are mathematically proved.
- Since then, WG cipher family has been investigated (e.g., Communication Security Lab at Univ. of Waterloo) for many different applications, such as Internet-of-Things (IoT).

Example. WG-8 (patent in 2014, Aagaard, G, Fan) for embedded system security



RFIDGUARD: Secure and Privacy-Preserving RFID Communication Systems

Description

As passive and active RFID systems are increasingly deployed for **mission-critical applications** and **business processes**, their vulnerability to a wide range of attacks has raised a lot of attentions. The tight cost constraints inherent in mass deployments of RFID tags bring forward impending requirements for novel and **lightweight** security solutions that can perform **strong authentication and encryption**, and protect **end-user privacy** for ultra-low power RFID applications. The RFIDGUARD developed by the research team at University of Waterloo is based on the lightweight stream cipher **WG-8**, which **fits even on low-cost passive RFID tags** and gives system integrators and end-users a peaceful mind about the security and privacy of their sensitive business data.

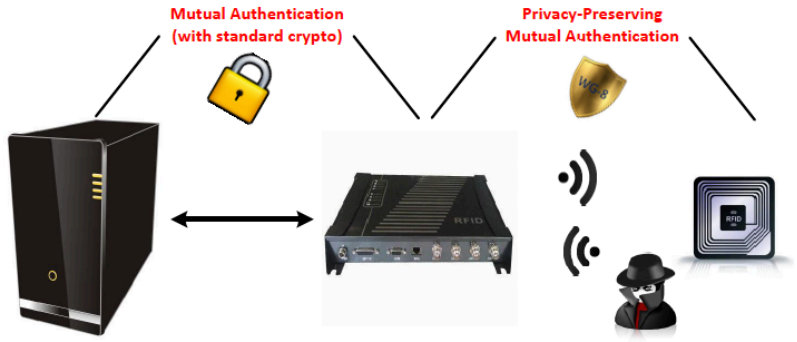


Figure 1: A Privacy-Preserving RFID Authentication System with WG-8 Stream Cipher

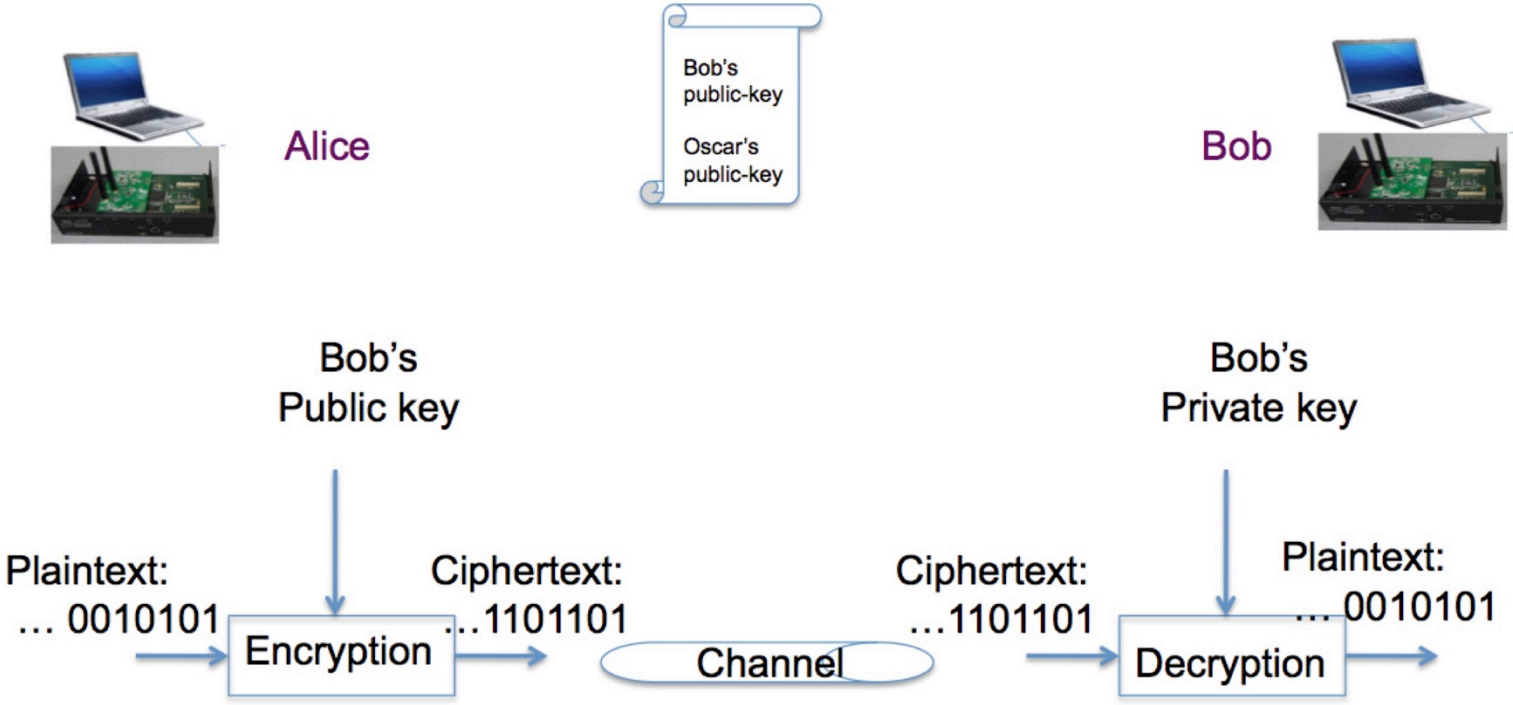
Key Features

- mathematically proven randomness properties
- 80-bit key for lightweight data protection
- privacy-preserving mutual authentication
- high-speed stream cipher encryption
- low communication bandwidth
- low code space requirement
- ultra-low power consumption
- well-suited for software and hardware implementations
- scalable key management and distribution system

Table 1: **Software Performance of Lightweight Stream Cipher WG-8 on Low-Power Microcontrollers @ 8 MHz**

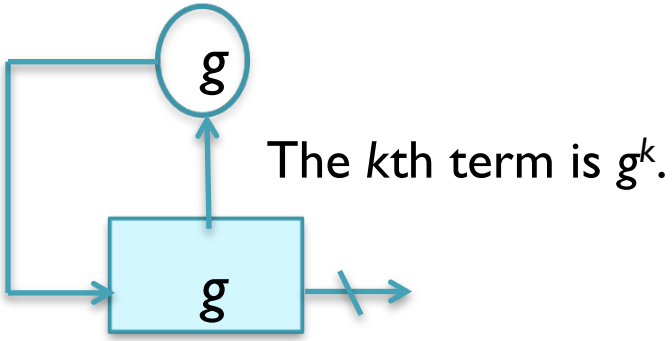
Low-Power Micro-controller	Memory Usage [byte]		Setup [cycle]	Throughput [Kbits/sec]
	Flash	SRAM		
ATmega	**1,984**	**20**	**1,379**	**185.5**
MSP430	**1,558**	**20**	**3,604**	**95.9**

Can we do public-key crypto using sequences?



Simplified Model of Public-Key Encryption

LFSR based Diffie-Hellman (DH) key agreement



The kth term is g^k.

- Diffie-Hellman key exchange protocol (1976) can be considered as using 1st order LFSR over GF(p) = {0, 1, …, p -1} where p is prime.

Following this observation:

- LUC (Smith and Skinner, 1994), using 2nd order LFSR over GF(p)
- GH public-key (G, Harn, 1999), using 3rd order LFSR over GF(p)
- XTR (Lenstra and Verheul, 2000), using 3rd order LFSR over GF(p^2)
- Analogues to GH and XTR (Giuliani and G, 2003)
- …

What are happened in the real world for deployed bad pseudo-random sequence/number generators (PRG)?

- Sony Playstation 3's master key was exposed, because it used a PRSG with poor randomness properties.
- NIST standardizes Random Number Generation Using Deterministic Random Bit Generators (DRBG) in 2012 in the following two ways:
  - use of block cipher, hash function to a random seed
  - use of elliptic public-key algorithm, Dual_EC_DRBG to a random seed
- The Dual_EC_DRBG is adopted from NSA's cipher suite, which has been found a backdoor (i.e., nonrandomness).
- NIST has removed it from their standard in 2015.
- Break PRG, then most of the time you break the entire security system!

Concluding Remarks

- The field on shift register sequences (or equivalently, pseudorandom sequences), created by Dr. Golomb, are widely used in numerous cryptographic algorithms:
  - stream cipher, block cipher, PRNG, KDF, pseudorandom functions, challenge number generations for authentication protocols
  - public-key schemes
  - hardware test vector
  - countermeasure for side-channel attacks

  …

  for protecting our daily digital world including
  - on-line banking, shopping, health record transfer, social security number for on-line job applications, …

References

All works introduced here can be found at

http://comsec.uwaterloo.ca