Prague, December 3 - 7, 2007

Design and Cryptanalysis of Stream Ciphers

University of Applied Sciences Northwestern Switzerland
School of Engineering

willi.meier@fhnw.ch

Overview

- Stream Ciphers: A short Introduction
- Stream Ciphers based on Linear Feedback Shift Registers
- Cryptanalysis principles
- Correlation attacks
- Linear attacks
- Distinguishing attacks
- Algebraic attacks
- Algebraic Immunity of S-Boxes and Augmented functions
- Multivariate Hash Functions
- Other attacks (Historical Overview)

Why stream ciphers?

Potential applications:

Embedded systems

RFID‘s

Components in lightweight cryptography

Introduction

Stream cipher:

Encrypts sequence of plaintext characters, e.g., from binary alphabet {0,1}.

Synchronous stream cipher:

The output of a pseudorandom generator, the ***key-stream***, is used together with plaintext to produce ciphertext.

Additive stream cipher:

Ciphertext symbols c_i are obtained from plaintext symbols m_i and keystream symbols b_i by addition.

Addition is often bitwise XOR (i.e., addition mod 2):

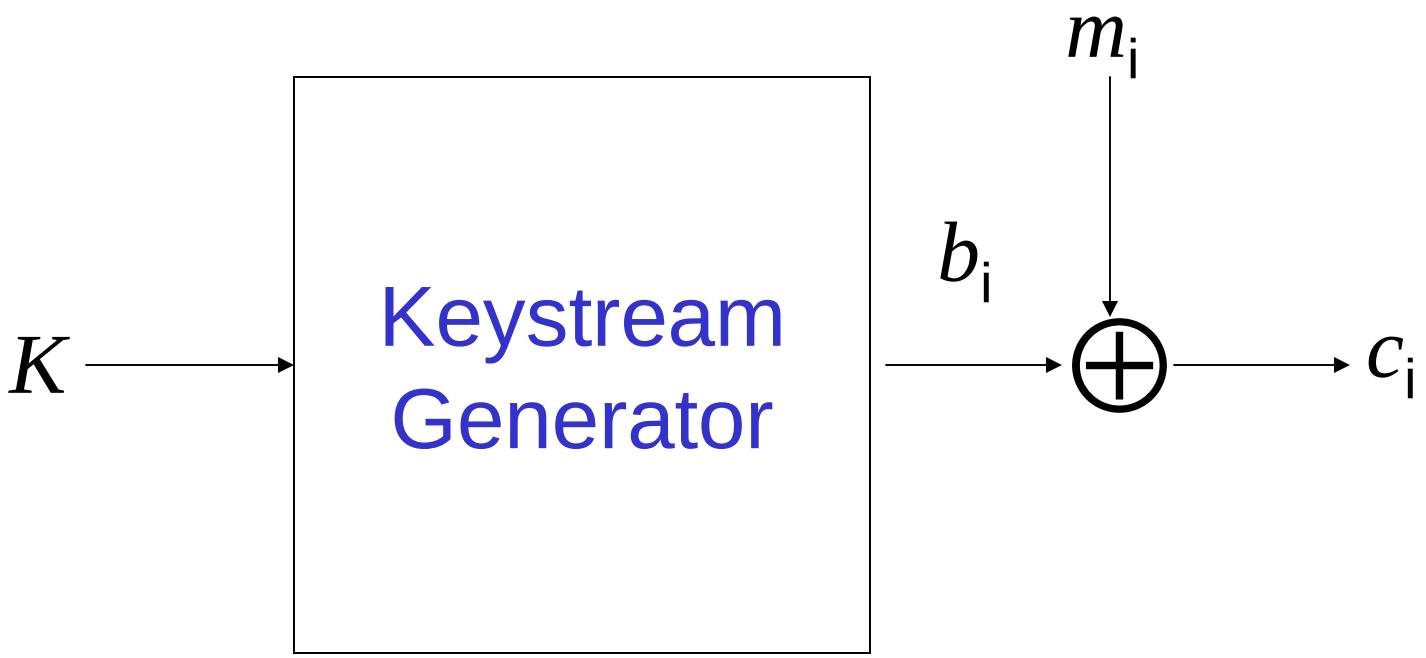
$$c_i = m_i + b_i \pmod 2.$$

Before transmission, secret key K has to be transmitted in *secure way* to receiver.

Decryption:

Achieved simply by subtracting keystream symbols from ciphertext symbols: $m_i = c_i + b_i \pmod 2$

Model of a binary additive stream cipher:



Some popular stream ciphers:

- RC4, used in Netscape‘s Secure Socket Layer (SSL) protocol
- A5, in the Global System for Mobile Communication (GSM)
- Bluetooth stream cipher, standard for wireless short-range connectivity, specified by the Bluetooth Special Interest Group

Stream ciphers

- Are **generally faster** than other symmetric encryption systems like block ciphers, and are much faster than any public key cryptosystem.

- Are more appropriate, when **buffering is limited**, or when characters must be individually processed.

- Have **no error propagation**.

Prototype stream cipher: **One-time-pad**

Keystream is randomly chosen binary string of same length as plaintext, and is never used again.

One-time-pad is „unconditionally secure“.

If keystream is reused, the one-time-pad (like every stream cipher) is insecure.

Drawback of one-time-pad: Key as long as plaintext; makes distribution of secret key difficult in practice.

In practical applications:
Random keystream is replaced by output of an efficient deterministic pseudorandom generator.

Initial state is short random string K of binary digits (e.g. of 128 bits).
Only secret key K needs to be securely transmitted.

Thereby provable security is lost.

For cryptographic applications, generated key-stream should pass a whole battery of statistical tests, e.g. NIST statistical test suite.

Computational effort to predict the keystream for unknown initial state:
Should be far beyond the capabilities of an adversary.

Many pseudorandom generators used, e.g., for computer simulations, wouldn‘t satisfy this requirement.

Focus here on stream ciphers based on Linear Feedback Shift Registers (LFSR‘s).

Are easy to implement, at least in hardware, and run efficiently.

Stream Ciphers based on LFSR‘s

A LFSR of length *n:*

Consists of a bit vector $(x_n,...,x_1)$. In one step, each bit is shifted one position to the right, except the right-most bit x_1 which is output.

On the left, a new bit is shifted in, by a *linear recursion*

$$x_j = (c_1 x_{j-1} + c_2 x_{j-2} + ... + c_L x_{j-L}) \bmod 2,$$

for $j \geq n$

Depending on the chosen linear recursion, LFSR‘s have desirable properties:

- Produce output sequences of large period (e.g. maximum period 2^n-1)
- Produce sequences with good statistical properties
- Can be readily analyzed using algebraic techniques

Linear recursion of LFSR can also be described by feedback polynomial:

For an n-stage LFSR with feedback coefficients $c_0, c_1,...,c_n$, the characteristic polynomial is defined by

$$f(x)=c_0 + c_1x + ... + c_{n-1}x^{n-1} + x^n$$

f is primitive, if f divides $x^{2^n-1} + 1$, but not

$x^e + 1$ with $e < 2^n - 1$. (Polynomial arithmetic

over GF(2)). If f is primitive, output sequence of LFSR has maximum period $2^n - 1$.

Serious drawback of LFSR‘s for cryptography:
Output is easily predictable, even for unknown initial state of bit vector $(x_n,...,x_1)$, and unknown recursion:

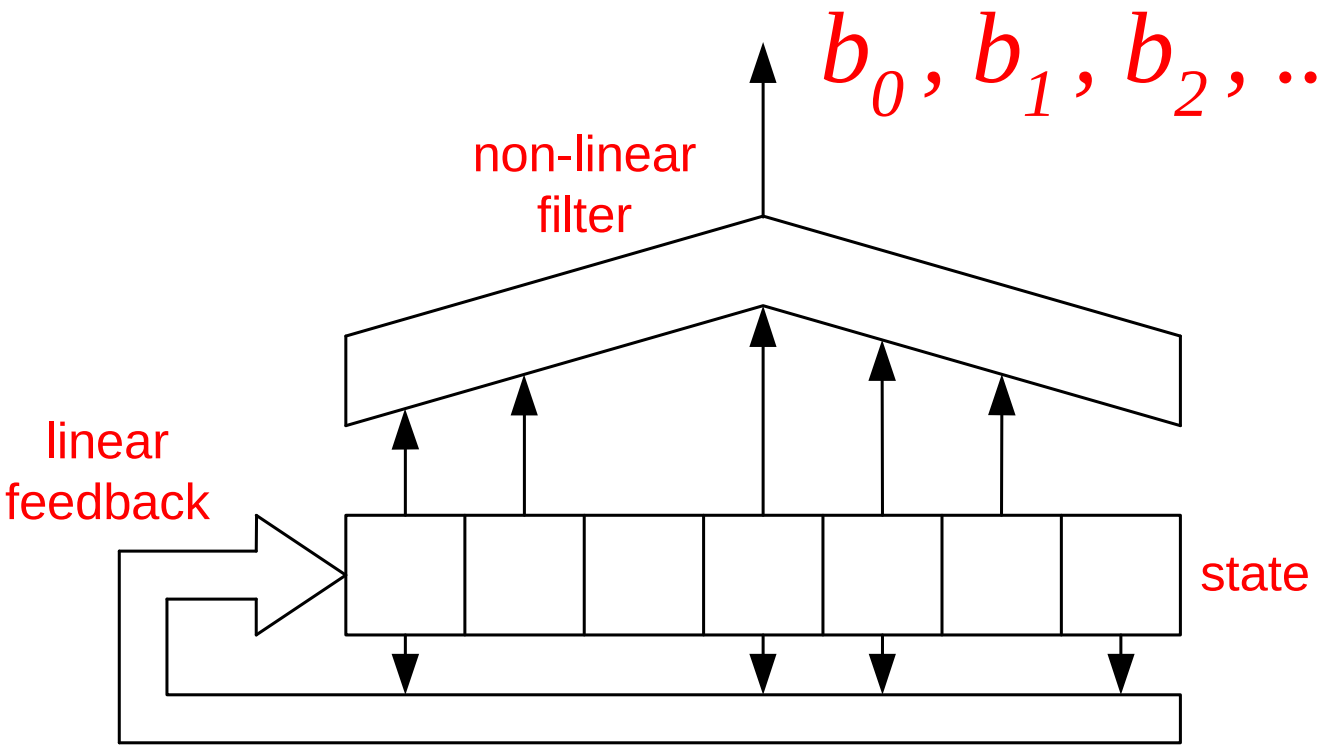
Solve a system of linear equations in unknown state bits (and coefficients for the recursion).

Common methods for destroying linearity properties of LFSR‘s:

- Use nonlinear filter/combining function on outputs of one/several LFSR‘s
- Use output of one/more LFSR‘s to control the clock of one/more other LFSR‘s.

Nonlinear filter generator:

Generate key-stream bits b_0, b_1, b_2 ,..., as some nonlinear function f of the stages of a single LFSR.



Vice versa:

Given any binary vector $\underline{b}$ of length $2^n - 1$, and given any maximum length LFSR of length n with a nonzero initial state.

Then there exists a unique filter function f which can produce this vector $\underline{b}$ as the first period of the output sequence of the filter generator with this LFSR and initial state, and with this filter function f.

Example of combiner generator: Geffe generator (historical)

3 LFSR's X, Y , Z, with outputs x_t, y_t, z_t.

Output b_t determined as:

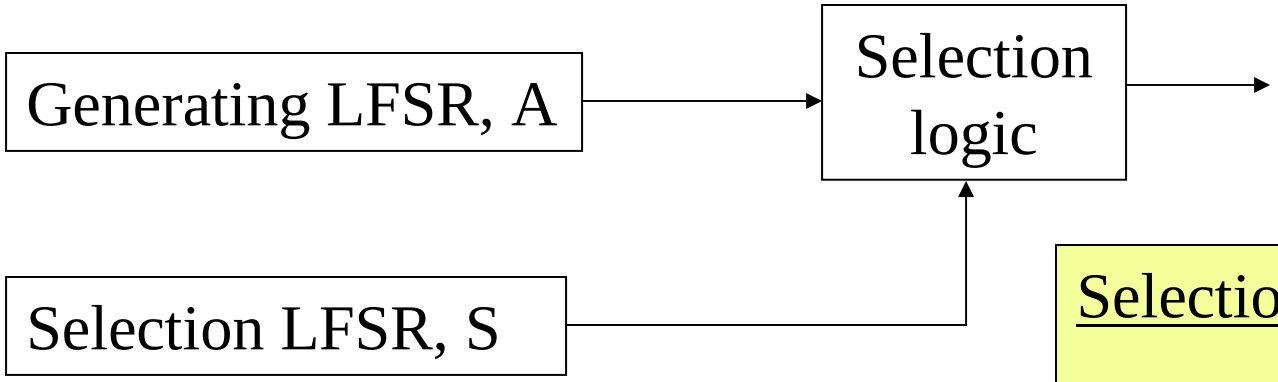
If $y_t = 1$
$b_t = x_t$
else
$b_t = z_t$

Combining function:

$$b = f(x,y,z) = x_t * y_t + (y_t + 1) * z_t.$$

The Shrinking generator

Proposed in 1993 by Coppersmith, Krawczyk and Mansour



Selection logic

The output of A is taken as a keystream output if the current output bit of S is 1, otherwise it is discarded.

Shrinking generator efficient in hardware, but not so in software.

Recommended that linear recursions of LFSR's A and S be secret.

Shrinking generator has withstood all cryptanalytic attempts, even if recursions are known.

Self-shrinking generator:
Requires only one LFSR. Selection rule same as in shrinking generator.

Uses even bits a_0, a_2,.. as S-bits, and odd bits as A-bits. A tuple (a_{2i}, a_{2i+1}) in output of LFSR outputs a_{2i+1} if $a_{2i} = 1$, else it outputs nothing.

Other types of stream ciphers:

- Word-oriented stream ciphers, suitable for software implementation (e.g. SNOW, SOBER, SCREAM)

- Stream cipher modes of operation of block ciphers (e.g., cipher feedback, output feedback mode of Triple DES or AES)

Stream cipher with provable security:

QUAD (Berbain-Gilbert-Patarin, 2006)

Based on difficulty of solving systems of multivariate quadratic equations mod 2.

Cryptanalysis principles

In cryptanalysis of stream ciphers: Common to assume either that

- some part of plaintext is known, (known-plaintext attack), or
- plaintext has redundancy (e.g., has ASCII format).

For additive stream cipher, a known part of plaintext is equivalent to a known part of keystream.

- Key recovery attack: Attempt to recover secret key K out of observed keystream

- Distinguishing attack: Try to distinguish observed keystream from being a purely random sequence

Distinguishing attacks often weaker than key recovery attacks.
May still be threat, if they allow to deduce information on unknown plaintext out of known part of plaintext, e.g. if period of keystream sequence is small.

Conseqence: Period needs to be large.

Linear complexity of a binary sequence:

Length of shortest LFSR that can produce the given sequence.

Berlekamp-Massey algorithm:
Efficient procedure to deliver shortest LFSR, together with initial state that can generate given sequence.

Consequence: Linear complexity of key-stream needs to be large.

For LFSR-based stream ciphers, the initial states of LFSR‘s involved are either

- derived by a key schedule mechanism out of secret key K and an initial vector IV, or
- directly coincide with K

Divide-and-conquer:

Attempt to determine first initial states of subset of LFSR‘s, in order to reduce complexity of search for right key.

Correlation Attacks

Example: Combination generator

The outputs a_m of s LFSR's are used as input of a Boolean function f to produce keystream,

$$f(a_{1m},\ldots,a_{sm}) = b_m$$

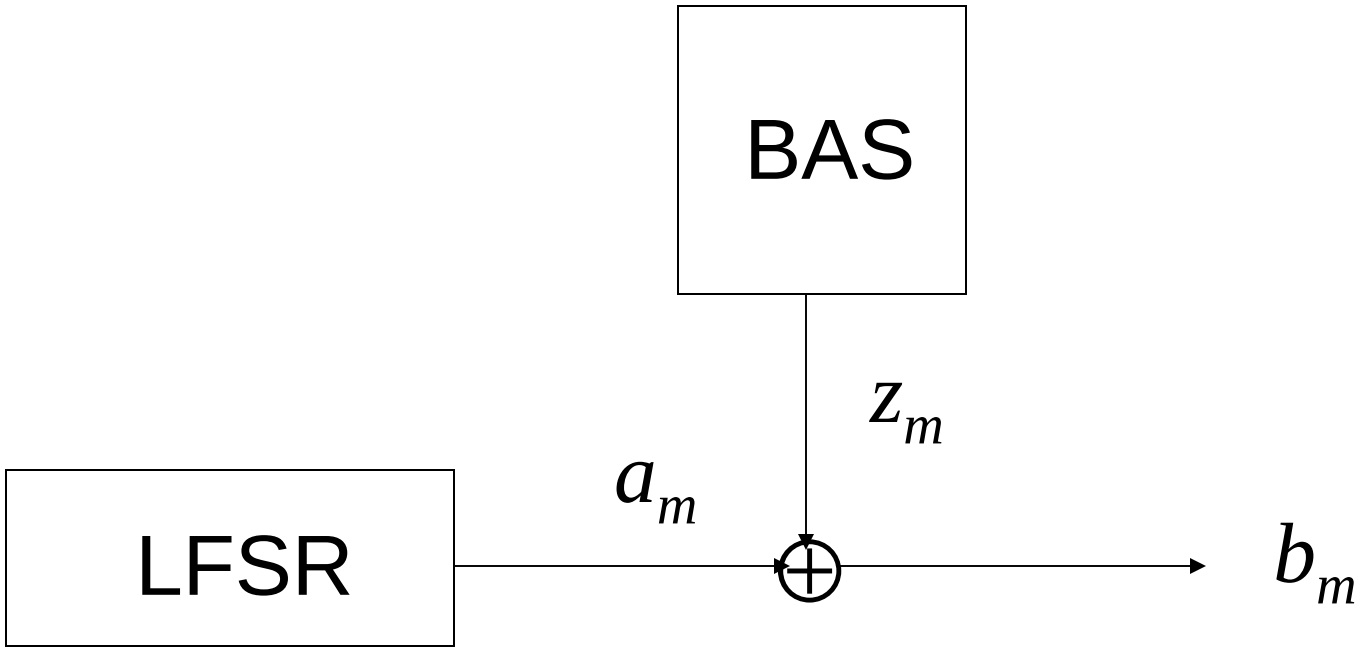
Correlation: $Prob(b_m = a_{im}) = p, p \neq 0.5$

Example: $s = 3$

$$f(x_1, x_2, x_3) = x_1x_2 + x_1x_3 + x_2x_3$$

$$p = 0.75$$

Statistical Model:



BAS: Binary asymmetric source,

$Prob(z_m = 0) = p > 0.5$

Problem: Given N digits of $\underline{b}$ (and the structure of the LFSR, of length n)

Find correct output sequence $\underline{a}$ of LFSR

Known solution: By exhaustive search over all initial states of LFSR find $\underline{a}$ such that

$$T = \#\{ j \mid b_j = a_j, 1 \leq j \leq N \}$$

is maximum. Complexity: $O(2^n)$
Feasible for n up to about 50.

Fast correlation attacks

Fast correlation attack: Significantly faster than exhaustive search over all initial states of target LFSR.

Based on using certain parity check equations created from feedback polynomial of LFSR.

Two phases

- Search for suitable parity check equations
- Equations are used in fast decoding algorithm to recover initial state of LFSR.

Algorithms most efficient if feedback connection has only few taps.

Closely related: Linear syndrome decoding, has been applied for fast correlation attacks (Zheng-Yang, 1988)

Algorithm description:

Example: $n = 3$. Recursion: $x_j = x_{j-1} + x_{j-3} \bmod 2$

Squaring: Recursion $x_j = x_{j-2} + x_{j-6} \bmod 2$ does also hold.

$$a_{j-3} + a_{j-1} + a_j = 0$$
$$a_{j-2} + a_j + a_{j+1} = 0$$
$$a_j + a_{j+2} + a_{j+3} = 0$$

A fixed digit a_j of the LFSR sequence $\underline{a}$ satisfies a certain number m of linear relations (involving a fixed number t of other digit of $\underline{a}$), obtained by shifting and iterated squaring of LFSR-relation.

Substitute the digits of the known sequence $\underline{b}$ in these relations (some relations may hold; some others not)

Observation:

The more relations are satisfied for a digit b_j, the higher is the (conditional) probability that $b_j = a_j$

Compute probability p^* for $b_j = a_j$, conditioned on the number of relations satisfied.

Digit contained in one relation:

Assume a fixed digit $a^{(0)} = a_j$ satisfies a linear relation involving t other digits of the LFSR-sequence $\underline{a}$,

$$a^{(0)} + a^{(1)} + a^{(2)} + ... + a^{(t)} = 0$$

Denote by $b^{(0)}, b^{(1)}, b^{(2)}, ..., b^{(t)}$ the digits in same positions of the perturbed sequence

$$b^{(0)} = a^{(0)} + z^{(0)}$$
$$b^{(1)} = a^{(1)} + z^{(1)}$$
$$.....$$
$$b^{(t)} = a^{(t)} + z^{(t)}$$

$$Prob(z^{(0)} = 0) = \ldots = Prob(z^{(t)} = 0) = p$$

$$s = Prob(\ z^{(1)} + \ldots + z^{(t)} = 0) : s = s(p,t)$$

$$s(p,t) = p * s(p,t\text{-}1) + (1\text{-}\ p)(1\text{-}\ s(p,t\text{-}1))$$

$$s(p,1) = p$$

Digit contained in several relations:

Assume that a fixed digit $a = a_j$ is contained in m relations each involving t other digits.

For a subset S of relations denote by $E(S)$ the event that exactly the relations in S are satisfied.

$$Prob((b=a) \text{ and } E(S)) = ps^h(1-s)^{m-h}$$

$$Prob((b \mathrel{!}= a) \text{ and } E(S)) = (1-p)s^{m-h}(1-s)^h$$

where $h = |S|$ denotes the number of relations in S.

New probability $p^* = Prob(b = a \mid E(S))$:

$$p^* = \frac{ps^h(1-s)^{m-h}}{ps^h(1-s)^{m-h} + (1-p)s^{m-h}(1-s)^h}$$

Probability distributions for number of relations satisfied: **Binomial distributions**

Correct digits: $b = a$

$$p_1(h) = \binom{m}{h} s^h (1-s)^{m-h}$$

Incorrect digits: $b \mathrel{!}= a$

$$p_0(h) = \binom{m}{h} s^{m-h} (1-s)^h$$

Average number m of relations available:

$$m = \log_2\left(\frac{N}{2n}\right)(t+1)$$

Example: $p = 0.75$, $t = 2$, LFSR-length $n = 100$, $N = 5000$ output bits of $\underline{b}$.

Then $m = 12$ (in the average), and $s = 0.75^2 + 0.25^2 = 0.625$.

Example (cont.) Value of p^*, if h relations are satisfied:

h	p^*
12	0.9993
11	0.9980
10	0.9944

Two algorithms, Algorithms A and B, for „fast correlation attacks“ (Meier-Staffelbach, J. Cryptology, (1989)). Much faster than exhaustive search, even for long LFSR‘s (n=1000 or longer). Only efficient for low weight recursions ($t < 10$).

Algorithm A

Take the digits of $\underline{b}$ with highest (conditional) probability p^* as a guess of the sequence $\underline{a}$ at the corresponding positions.

Approximately n digits are required to find $\underline{a}$ by solving linear equations.

Computational complexity: $O(2^{cn})$, $0 < c < 1$, i.e., complexity is exponential. c is a function of p, t and N/n.

Example: $c = 0.012$ if $p = 0.75$, $t = 2$, and $N/n = 100$.

Algorithm B

1. Assign the correlation probability p to every digit of $\underline{b}$
2. To every digit of $\underline{b}$ assign the new probability p^* . Iterate this step a number of times.
3. Complement those digits of $\underline{b}$ with $p^* < p_{\text{thr}}$ (suitable threshold).
4. Stop, if $\underline{b}$ satisfies the basic relation of the LFSR, else go to 1.

The number of iterations in 2. and the probability shreshold in 3. have to be adequately chosen to obtain maximum correction effect.

Algorithm B is essentially linear in the LFSR-length n

Successful only if $t < 10$.

Problem: Fast correlation attacks for arbitrary linear relations, i.e., for arbitrary t ?

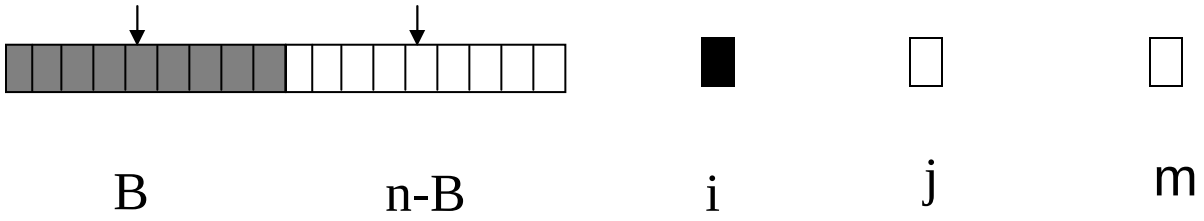
Problem can be viewed as a decoding problem.

Result: (Johansson-Jönsson, Mihaljević-Fossorier-Imai, Joux, et. al.) Fast correlation attacks feasible for arbitrary linear relations and LFSR-length n up to 100.

Fast correlation attack for arbitrary weight t:

- Call target bit a LFSR output bit to be predicted.
- Construct set of parity checks, involving k output bits.
- Evaluate estimators and conduct majority poll among them to recover initial state of LFSR.

Procedure is combined with partial exhaustive search for efficiency:
For length n LFSR, B bits are guessed through exhaustive search, and n-B bits found using parity checks.



Parity check combines two bits j and m together with linear combination of guessed bits B in order to predict target bit i.

Let $D > n$. For each of D target bits, evaluate large number of parity checks using noisy values b_t, and count number of parity checks that are satisified.

Number of parity checks satisfied: N_s
Number of parity checks not satisfied: N_u.

If difference $N_s - N_u$ is larger than threshold, predict $x_i = b_i$ if $N_s > N_u$, else $x_i = b_i + 1$.

If majority polls successful for at least n-B of the D target bits, can easily recover initial state of LFSR.

Preprocessing: Parity checks found by collision search using Birthday paradox.

Extensions

If recursion not of low weight, can consider multiples of feedback polynomial that have low weight. Apply correlation attack to linear recursion of sparse polynomial multiple.

Correlation attacks applicable in simple cases, even if feedback connections not known

Correlation attacks are successful if cipher allows for good approximations of the output function by linear functions in state bits of LFSR's involved.

Impact of correlation attacks to design of stream ciphers:

Boolean functions f used

- should be correlation immune, and
- should have large distance to affine functions

f is correlation immune if output is uncorrelated to single inputs.
Distance of Boolean functions: Measured by Hamming weight of truth tables.

Tradeoff between correlation immunity and algebraic degree of function (Siegenthaler): The higher the order of correlation immunity, the lower the degree.

In order to resist Berlekamp-Massey, degree should be large

Nonlinearity bounds and constructions of interesting functions.

Criteria found for Boolean functions also relevant for design of Block ciphers

Problem: Conflicting **tradeoffs** amongst various design criteria

Can be avoided by introducing memory in combining functions.

Example: Integer addition. Is nonlinear when considered over GF(2). Carry bit serves as memory.

Bluetooth stream cipher is combiner with memory. Nevertheless conditional correlation attack found (LMV, Crypto 2005).

Application of correlation attacks: TCHo Hardware-oriented public-key cryptosystem

Security based on hardness of finding a low-weight multiple of a given polynomial (FV, 2006, AFMV, 2007).

Publicly known: A LFSR with feedback polynomial P

Trapdoor: Low weight polynomial multiple K

Encryption:

Plaintext x is encoded, $C(x)$

Ciphertext y :
Is addition (mod 2) of $C(x) + S_{\mathrm{LP}} + N_{\mathrm{b}}$

S_{LP} : Output of LFSR

N_{b} : Random noise with bias b

Decryption:

K is used to delete S_{LP} by summing up bits of ciphertext y according to recursion defined by K.

Obtain noised version of a code word $C(x')$. Noise has bias b^{w} , (w equals weight of K).

Decode $C(x')$ by majority decoding, if C is repetition code.

Get x out of x' by linear transformation.

Only possessor of low weight multiple can decript, as otherwise noise too large.

Linear Attacks

Recall: Correlation attack successful, if linear relations hold with nonnegligible probabilities, between single output bits and a subset of state bits of driving LFSR‘s.

Linear attack: Successful if there are correlations between linear functions of several output bits and linear functions of a subset of the LFSR-bits.

If there are such correlations, get a linear system of equations, each of which does hold with some probability.

Linear system can be solved by methods reminiscent to fast correlation attacks (Golić).

Methods efficient if known keystream is long enough, i.e., if many more equations are available than number of unknowns.

Distinction between correlation attacks and linear attacks relevant, if combiner contains memory:

Consider block of m consecutive inputs

$Z_t = (z_t, z_{t-1},..., z_{t-m+1})$ as a function of the corresponding block of m consecutive inputs $X_t=(X_t, X_{t-1},...,X_{t-m+1})$ and the preceeding memory bits C_{t-m+1}.

X_t denotes bit vector at time t of state bits of driving LFSR's, and C_{t-m+1} bit vector of M memory bits at time t-m+1.

Assume that X_t and C_{t-m+1} are balanced and mutually independent.

Then, if $m >= M$, there must exist linear correlations between the output and input bits (Golić), but they may also exist if $m<M$.

Linear Cryptanalysis of Bluetooth Stream Cipher (Golić-Bagini-Morgari)

Generator consists of 4 LFSR‘s of total length 128 that are combined by a nonlinear function with 4 bits of memory.

Initialization vector (IV) consists of 74 bits.

Combining function is modified combining function of summation generator:

Reduces some weakness of summation generator identified earlier.

Secret key 128 bits.

Internal state is 132 bits.

Keystream sequences produced very short, i.e., at most 2745 bits for each initialization vector.

Large class of linear correlations identified in Blue-tooth combiner which, in spite of short keystream sequences available:

Enable to reconstruct the LFSR initial states, but even the secret key from some number of IV 's.

Different types of linear correlations:

- Unconditioned correlations,
- correlations conditioned on output, and
- correlations conditioned on both, output and assumed input.

Distinguishing attacks

Goal of key-recovery attacks: Find secret key faster than by exhaustive search.

Distinguishing attacks:

Allow for distinguishing observed keystream from random, or

make prediction about future portions of keystream out of known keystream segment.

General statistical framework:

Hypothesis testing

Need to distinguish probability distribution generated by output of a stream cipher from truly random distribution.

Specific distinguishing attacks are linear attacks and low diffusion attacks (Golić, Coppersmith-Halevi-Jutla).

Linear attack: Concentrate on non-linear output function to look for characteristic that can be distinguished from random, e.g. linear approximation that has noticeable bias.

Linear attacks have been applied e.g. to RC4, to SNOW and to SOBER.

Low diffusion on Scream-0, a simplified variant of Scream.

Debate about applicability of distinguishing attacks.

It has been argued that some distinguishing attacks against stream ciphers are unrelated to their security in practical use:
Amount of data required to perform distinguishing attack is huge compared to actual lifetime of secret key used.

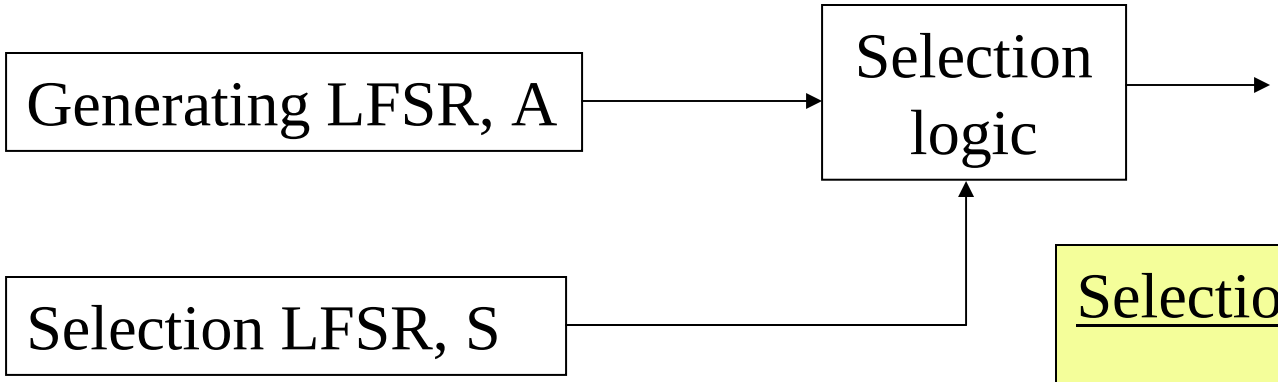
Distinguishing attack on shrinking generator

Previous work:

Divide-and-conquer attack (by designers)
requires exhaustive search through all possible initial states and feedback polynomials of S.
Thereafter linear consistency test using known recursion of A.

Correlation attacks (Johansson, Simpson-Golić-Dawson), are exponential in length of A.

Shrinking generator



Selection logic

The output of A is taken as a keystream output if the current output bit of S is 1, otherwise it is discarded.

Distinguishing attack (Golić): Detectable linear statistical weakness if feedback polynomial of A of very low weight and moderate degree.

Attack considers shrunken linear recursions in single bits.

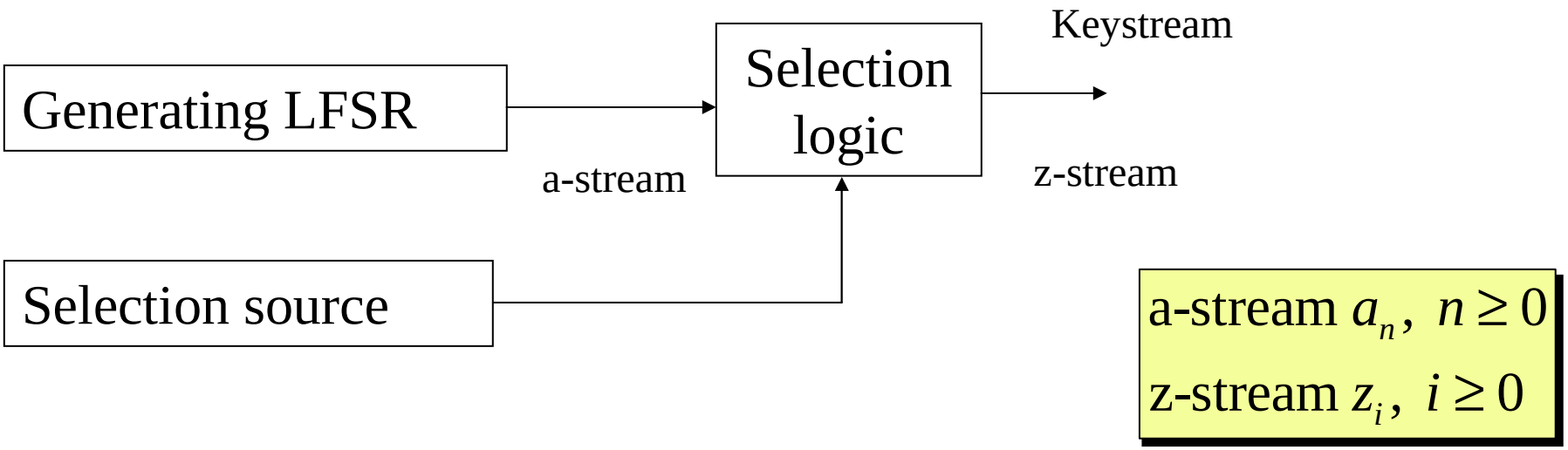
Another distinguishing attack on shrinking generator (SG) (Ekdahl-Meier-Johansson, 2003)

Rather than single bits, consider bit strings (blocks) in the a-stream, and compare with suitable blocks in z-stream.

Attack model

Linear recursion:

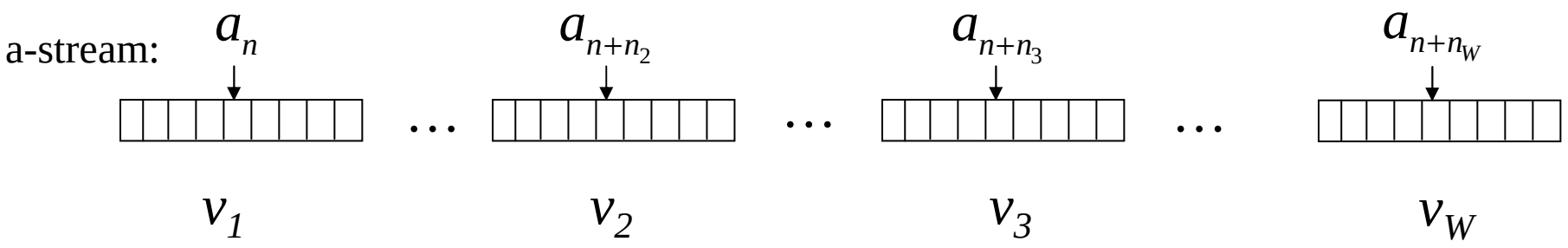
$$a_n + a_{n+n_2} + a_{n+n_3} + \ldots + a_{n+n_W} = 0, \qquad n \geq 0$$



A weight W feedback polynomial which is known.

The selection sequence can be any random sequence with independent and equally distributed bit probabilities.

Main observation



The xor sum of these vectors (blocks) equals zero. $\sum_j v_j = \overline{0}$

Take the majority bit of each block:

$m_j = Maj(v_j)$, then we have for the xor sum

$$P(\sum_j m_j = 0) > \frac{1}{2}.$$

General idea of the attack

Definitions: *The imbalance of a block B, is defined as:*
Imb(B) = #1 - #0.

The positions

z_{i_1}, $z_{i_1+n_2/2}$, $z_{i_1+n_3/2}$, …, $z_{i_1+n_W/2}$

are called the shrunken tap positions.

Find blocks in the z-stream with high imbalance around the shrunken tap positions

↓ *with high probability*

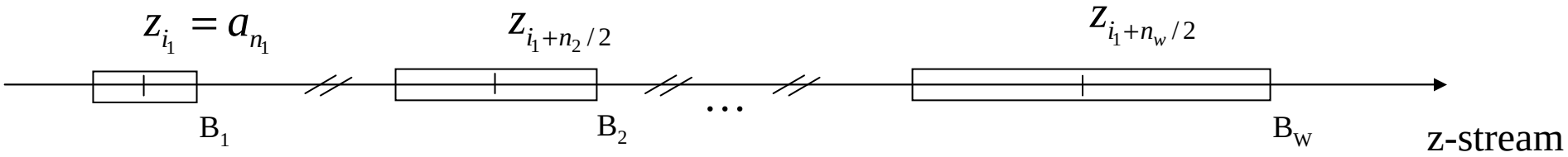
corresponding blocks in the a-stream are imbalanced

↓

estimate the majority bits of the blocks surrounding the tap positions.

First phase

Goal: Find suitable positions in the z-stream which have imbalanced blocks.



Block lengths:

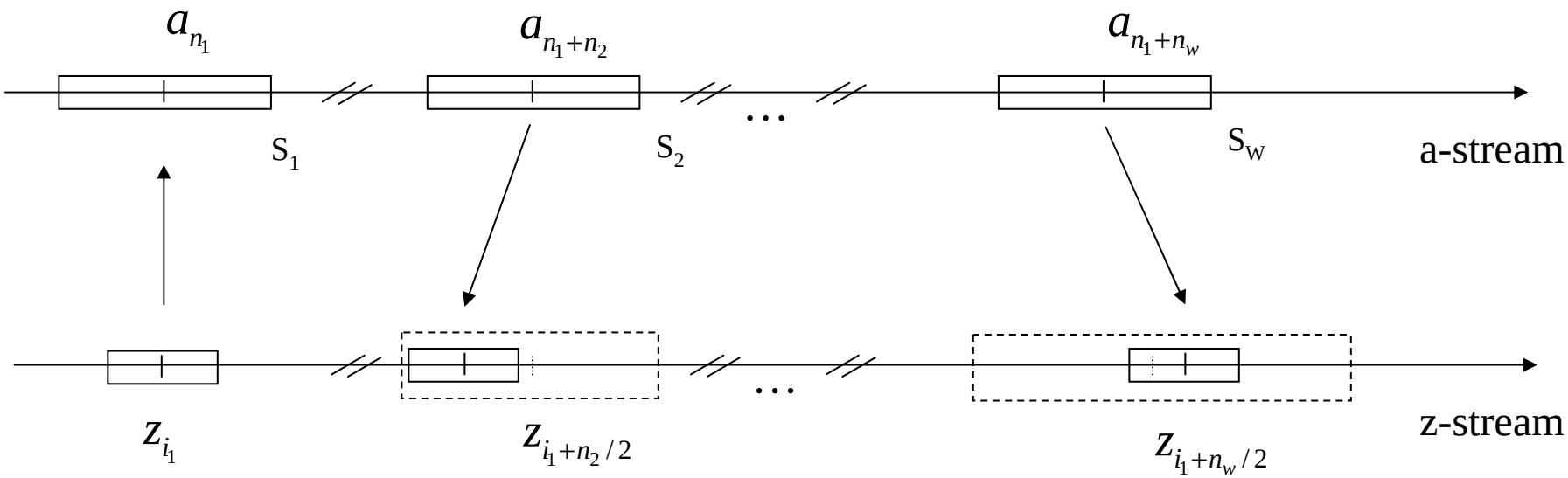
$BL_1 = E+1$, where E is an even parameter to the attack.

$$BL_j \approx BL_1 + \frac{\sqrt{n_j}}{2}, \quad j = 2...W$$ take closest odd integer.

Whenever $|Imb(B_j)| > T$, $j=1 \ldots W$ we have a "**hit**" and invoke the second phase.

Second phase

Goal: Estimate the bit probability of the corresponding a-blocks, and thus the majority bits.



p_j : estimated bit probability of S_j, but also an estimate of the majority bit of S_j.

$$\hat{m}_j = \begin{cases} 1 \text{ if } p_j \geq 1/2 \\ 0 \text{ otherwise} \end{cases}, \quad j = 1...W$$

Distinguishing

Denote by **good** the number of times $\sum_j \hat{m}_j = 0.$

Derive final probability $P = \frac{\text{good}}{\text{hits}}.$

If shrinking, we expect $P = 0.5 + \varepsilon_H,$

where ε_H is a positive value depending on the number of hits in first phase.

Use a Maximum Likelihood (ML) test to distinguish from the random case where $P = 0.5$.

Theoretical analysis

- Probability P_{M} that true majority bit sum zero for uniformly distributed bits
- P_{M} in case of skewed distribution due to condition |imb(B_{j})|>T.
- Expected ε_H in P=0.5 + ε_H
- Upper bound on required number of hits, H.
- Lower bound on expected number of hits.
- Required number of observed keystream symbols, N.
- Computational complexity $O(N\sqrt{n_W})$

Simulation results (1)

Weight 4 polynomials, T=3, E=14.

Tap positions (exluding 0)	Theoretical parameters				N used in attack	Successes out of 50 runs
	P_H	ε_H	H	N		
302,733,1000	0.02648	$2^{-10.1}$	$2^{20.2}$	$2^{25.4}$	2^{23} 2^{24} 2^{25}	43 46 50
812,1433,2500	0.03586	$2^{-11.5}$	$2^{23.0}$	$2^{27.8}$	2^{25} 2^{26} 2^{27}	39 46 50
2333,5847,8000	0.05542	$2^{-13.5}$	2^{27}	$2^{31.2}$	2^{28} 2^{29} 2^{30}	42 48 50
3097,6711,10000	0.05989	$2^{-13.9}$	$2^{27.7}$	$2^{31.8}$	2^{28} 2^{29} 2^{30}	45 45 46

Simulation results (2)

Weight 3 and 5 polynomials, T=3, E=14.

Tap positions (exluding 0)	Theoretical parameters				N used in attack	Successes out of 50 runs
	P_H	ε_H	H	N		
17983,40000	0.1414	$2^{-10.2}$	$2^{20.3}$	$2^{23.1}$	2^{21} 2^{22} 2^{23}	36 46 50
73,131,219,300	0.0068	$2^{-11.56}$	$2^{23.1}$	$2^{30.3}$	2^{29} 2^{30}	48 50

Practical distinguishing attack for Shrinking Generator with known feedback polynomial for the generating LFSR.

Use a tap in the middle as reference and thus increase the probability of the estimates.

Prediction of bit distribution in the last block.

Algebraic Attacks

Belief: Ciphers using LFSR's can be made secure against attacks by using output functions that are correlation immune and have large distance to affine functions.

What are algebraic attacks?

Attacks by solving a system of algebraic equations (CM, Eurocrypt 2003).

Type of equations:

System of multivariate polynomial equations over a finite field, e.g., GF(2).

$$x_1 + x_0x_1 + x_0x_2 + \ldots = 1$$
$$x_1x_2 + x_0x_3 + x_7 + \ldots = 0$$
$$\ldots\ldots\ldots\ldots$$

Why algebraic attacks?

Breaking a „good“ cipher should require:

„ ... as much work as solving a system of simultaneous equations in a large number of unknowns of a complex type“ [Shannon, 1949, Communication theory of secrecy systems]

Common experience: Large systems of equations become intractable soon with increasing number of unknowns (is NP hard problem) .

However

Systems that are

- Overdefined, i. e. have more equations than unknowns, or
- Sparse

are easier to solve than expected:

- Linearization
- The XL method (Shamir-Patarin-Courtois-Klimov, Eurocrypt 2000), Gröbner bases

Direct algebraic approach:

Derive equations in key bits $k_0,\ldots,k_{n-1}$

$$\begin{cases} f(k_0,\ldots,k_{n-1}) = b_0 \\ f(L(k_0,\ldots,k_{n-1})) = b_1 \\ f(L^2(k_0,\ldots,k_{n-1})) = b_2 \\ \ldots\ldots\ldots\ \ldots\ldots\ldots\ \ldots\ldots\ldots\ \ldots\ldots \end{cases}$$

$L(\,)$: Linear recursion.

Solve this system of equations. Very overdefined, even for moderate quantity of keystream, e.g., 20 Kbytes.

An obvious linearization attack:

Assumption: f is of low degree d. Then the key is found given $K = \binom{n}{d}$ keystream bits and within K^{ω} computations, where ω is the exponent of Gaussian reduction ($\omega < 3$).

Linearization: One new variable for each monomial; Solve a linear system.

Improvement

What if the degree d is too large?

Example: Toyocrypt stream cipher (submission to the Japanese government Cryptrec call for crypto-graphic primitives).

Filter generator with one LFSR of length $n=128$, output function of degree $d = 63$.
This output function satisfies all previously known design criteria.

Reduce the degree of the equations?

Weakness of Toyocrypt

Output function $f(s_0,\ldots, s_{127})$ is of degree $d = 63$, but:
Is sum of only linear and quadratic terms, plus single monomials, each of degree 4, 17, and 63, respectively.

Parts of degree 4, 17 and 63 are all divisible by a common factor $s_{23}s_{42}$.

Assume $f(s) = 1$. Then $f(s)(s_{23} + 1) = s_{23} + 1$.

Higher terms cancel out, i.e., get an equation of degree $d = 3$!

New Type of Attack

By multiplying the equations by a well chosen polynomial, their degree can be reduced from $d = 63$ to $d = 3$.

Toyocrypt is

- broken in 2^{49} CPU clocks (few days on a PC)
- with 2^{18} keystream bits
- attack was verified experimentally.

Scenarios

Degree of output function f large, $f=g*h$

- $f*g=0$, degree of g low
- $f*g=h$, degrees of g and h low

If output bit $b_i=1$, take $g(s)=0$, else take equation $h(s)=0$

Overview of attack

Instead of $f(s) = b_t$ with $s = L^t(K)$, $K =$ key:

Solve the equations

$$f(s) * g(s) = b_t * g(s)$$

with well chosen function g.

Question: Do „good“ functions $g(s)$ exist ?

In some cases, such $g(s)$ ALWAYS do exist.

Theorem (Low degree relations)
Let f be any Boolean function in k variables.
Then there is a nonzero Boolean function g of degree at most $k/2$ such that $f(x) * g(x)$ is of degree at most $k/2$.

(Take ceilings of $k/2$ if k is odd)

Theorem has been motivated by cryptanalysis of multivariate digital signature schemes as well as by cryptanalysis of AES block cipher.

Proof of Theorem (sketch):

Look for Boolean function g, such that $f*g = h$ with $h = 0$ (the case h non zero is similar). Assume f is balanced (i.e., same number of 0 and 1 in truth table)

$g(x)$ is yet unknown Boolean function, i.e., in algebraic normal form is sum of monomials in the x_i's with unknown coefficients.

Substitute each argument $x = (x_1,...,x_k)$ with $f(x) = 1$ in $g(x)$. Get linear equation in the unknown coefficients of g. If degree of g at least $k/2$, number of coefficients to determine g larger than number of arguments x with $f(x) = 1$. Thus we have more unknowns than equations, and hence always a solution.

Consequences

- Can break any stream cipher with linear feedback and Boolean output function with small number k of state bits as input, in $\binom{n}{k/2}^{\omega}$
- Polynomial complexity, if k considered as small constant.
- Already known: By linearization, such cipher can be broken in $\binom{n}{k}^{\omega}$
- Complexity of new generic attack only approx. square root of known attack.

Extensions

Attack is very general and can be adapted to some stream ciphers that are not regularly clocked.

Example: LILI-128 (NESSIE proposal). Complexity of algebraic attack is $O(2^{57})$.

Attack can be generalized to stream ciphers that use combiner with memory (instead of memoryless output function):

Work by Armknecht and Krause: Algebraic attacks on combiners with memory, CRYPTO 2003, e.g., on Bluetooth generator *E0* . Complexity: $O(2^{68})$

Extension to combiners with memory and several outputs (Courtois).

Improved attacks: Fast algebraic attacks on stream ciphers with linear feedback, CRYPTO 2003.
Much lower complexities:

Toyocrypt	$O(2^{23})$
LILI-128	$O(2^{39})$
E0	$O(2^{49})$

Algebraic attacks on stream ciphers with memory:

Idea is to look at algebraic relations that are of low degree in input variables and involve output bits (of any degree), so that memory bits cancel out.

Theorem (Armknecht-Krause, Courtois) Let F be an arbitrary fixed circuit/component with k binary inputs, l bits of memory, and one output. Then, considering $M = l+1$ consecutive steps/states $(t,...,t+M-1)$, there is a multivariate relation, involving only the input bits and the output bits for these states, and with degree at most $k(l+1)/2$ in the input variables.

If combiner is memoryless, then $l=0$: Get previous Theorem as special case.

If number of memory bits increases, so does increase upper bound of degree of multivariate relations. Does not preclude existence of lower degree relations.

Example: Bluetooth stream cipher has combining function with $k=4$ inputs and $l=4$ bit memory. Theorem assures multivariate relation of degree at most 10, but actually, multivariate relation of degree 4 was found!

Recall: Linear complexity of keystream generator with filter function f of degree k is often about $\binom{n}{k}$ where n is length of LFSR.

Data complexity of new algebraic attack on such generator is only about square root of that of Berlekamp-Massey algorithm.

Cases known, where successful algebraic attack only needs slightly more known keystream than length of driving LFSR.

Contrast: For many designs provable resistance against Berlekamp-Massey synthesis available, but not yet against algebraic attacks.

Consequences for the design of stream ciphers

- Output function f should use large subset of state bits (unlike LILI-128).
- Function f should not be sparse.
- No multivariate equations of low degree should exist that relate key bits and one or more output bits.
- Immunity against (fast) algebraic attacks: Symmetric functions f are very vulnerable to fast algebraic attacks (ACFGMR, Eurocrypt 2006).

Algebraic Immunity of S-Boxes and Augmented Functions

Several methods known how to compute algebraic immunity of Boolean functions.
Aspect less well studied:

Problem: Algebraic attacks on stream ciphers with lower data complexity?
(e.g., why can Gröbner bases work with very small known output data for a few filter generators?)

Algebraic properties of S-boxes

S-box can be described by implicit equations, $G(x,y) = 0$, with $S(x) = y$.

In algebraic attacks: Focus on low degree equations or sparse equations (Courtois-Pieprzyk, Armknecht).

Algebraic immunity of S-box: Maximum total degree of a monomial in x_i and y_i in $G(x,y) = 0$.

In our applications: Cryptographic meaning of x and y different: Consider degree of x_i's and y_i's in monomials in $G(x,y) = 0$ separately.

Augmented function of stream cipher is viewed as S-box.

Notation: Let F denote $GF(2)$.

S-box S: $F^n \rightarrow F^m$

with $S(x) = y$, $x = (x_1,..x_n)$, $y = (y_1,..,y_m)$.

Component equations $S_i(x) = y_i$.

Each monomial in $G(x,y) = 0$ written as $x^a y^b$, for multi-indices a,b in F^n, where $x^a = (x_1^{a_1} \cdot \ldots \cdot x_n^{a_n})$

d_x: maximum degree of x-part in monomials $x^a y^b$

d_y : maximum degree of y-part in monomials $x^a y^b$

Is there an equation $G(x,y) = 0$ of degrees d_x, d_y?

Number of possible monomials in equation: $D_x D_y$, where

$$D_x = \sum_{i=0}^{d_x} \binom{n}{i} \quad \text{and} \quad D_y = \sum_{i=0}^{dy} \binom{m}{i}$$

Consider matrix M in $GF(2)$ with $N_r = 2^n$ rows and $N_c = D_x D_y$ columns (Courtois):

Each row corresponds to input x, each column to evaluated monomial.

Number of implicit equations of degrees at most d_x , d_y is: $R = D_x D_y - \mathrm{rank}(M)$.

If number of columns in M is larger than number of rows, nontrivial linear combinations of columns (i.e., monomials) exist.

Hence equations $G(x,y) = 0$ can be expected if

$$D_x D_y > 2^n \quad (1)$$

If $n > 20$, solving for M is impractical due to memory requirements.

Simple improvement: Consider set of parameters where inequality (1) is not satisfied.

Instead of full matrix M, consider smaller matrix M' with N'_r random inputs, where N'_r is small multiple of N_c.

Nonexistence of solution:

Solve for M'. If no solution exists, no solution exists for M either.

Can determine non-existence of solution: Deduce algebraic immunity for parameters as set.

If solutions exist for M', they needn't hold for original setup, but do hold for a fraction p of all 2^n inputs to S-box.

Conditional equations

For large S-boxes, Matrix M to solve for $G(x,y) = 0$ still very large.

Suppose m is smaller than n.
Fix output y and determine conditional equations of degree d_x for this y.

Number of columns in M reduces to $N_c = D_x$.

Number of rows equals number of preimages of y, and is about 2^{n-m} (if S nearly balanced).

For fixed output vectors y, determine minimum degree d_x of a conditional equation $G_y(x) = 0$ with $S(x) = y$.

Bottleneck: Finding preimages of y for random S-box needs $D_x\, 2^m$ trials.

Algebraic attacks on augmented functions

Integrate general approach of implicit equations in the context of stream ciphers.

New scenarios of algebraic attacks on stream ciphers.

Based on algebraic properties of augmented function.

Stream cipher with update function L, output function f.

Augmented function S_m: $F^n \rightarrow F^m$

defined by

$$S_m: \quad x \rightarrow (f(x), f(L(x)),..., f(L^{m-1}(x))$$

Update function L linear (e.g. in LFSR) or nonlinear (e.g. in eSTREAM candidate TRIVIUM)

Recover initial state x by algebraic attack:

Deal with multiple outputs of augmented function.

+ : Allows to set up equations of smaller degree than for ordinary alg. attacks.

- : All n state variables of LFSR are involved instead of k variables of filter function.

+ : Can adapt output size m.

Implicit and conditional equations

According to inequality (1):

For an augmented function S_m: $F^n \rightarrow F^m$

with $m > m_0$, one can set up equations $G(x,y) = 0$ of degrees at most d_x and d_y, $d_y = m$, where $m_0 := n - \log_2 D_x$

m_0 is upper bound for m so that equation of degree d_x can be expected.

Tradeoff between degree d_x and m:

Would like to choose both as small as possible.

For efficiency, consider conditional equations $G_y(x) = 0$ for given m-bit outputs y.

Finding preimages:

As opposed to random S-boxes, augmented function of some stream ciphers has special (simple) structure.

Sampling methods in time-memory-tradeoff attacks on stream ciphers (Biryukov-Shamir).

New scenario:

1. There are many low-degree conditional equations for output size m of the augmented function, where m is smaller than m_0.
2. Finding preimages is feasible for output size m.

Application: Filter generator

Data complexity of algebraic attack: $\binom{n}{d}$

with LFSR length n and algebraic immunity d.

Experimental result: Algebraic attacks with Gröbner bases only need slightly more than n output bits in a few cases (Faugère-Ars).

Open issue to understand such behavior from Boolean filter function and tapping sequence.

Aim: Algebraic attack on augmented function with (very) low data complexity.

1st Step: Find preimages x for augmented function S_m for fixed output $y = S_m(x)$.

Filter inversion: Use table of filter function f (with k inputs) to choose input with correct observed output bit. Repeat for about n/k successive output bits, until state is unique.

Time complexity to find preimage of $m > n/k$ bits about $2^{m-n/k}$, i.e., efficient if k is small.

Linear Sampling: In each time step, a number of linear conditions is imposed in input variables of f, so that f becomes linear. The linearized filter gives one additional linear equation for each keystream bit until there arises a contradiction.

2nd Step: Existence of equations

Experimental results for instances of the CanFil family (as considered by Faugère-Ars) as well as a Majority function.

All functions have $k = 5$ inputs, and algebraic immunity 2 or 3.

LFSR's correspond to random primitive feedback polynomial.

Example (CanFil5): f is given by $x_2x_3x_4x_5 + x_2x_3 + x_1$

For function f and parameters d_x and m, consider augmented function, $y = S_m(x)$.

Determine the number R_y of conditional equations

$$G_y(x) = 0$$

of degree d_x for each m-bit output y.

For $n = 20$, and for varying m, the overall number of equations, $R := \sum_y R_y$ is recorded:

Filter	m	R for 5 setups				
CanFil1	14	0	0	0	0	0
	15	3139	4211	3071	4601	3844
CanFil2	14	0	0	0	0	0
	15	2136	2901	2717	2702	2456
CanFil5	6	0	0	0	2	0
	7	0	0	0	8	0
	8	0	0	0	24	0
	9	0	0	0	64	0
	10	6	0	0	163	0
	11	113	0	2	476	0
	12	960	16	215	1678	29
Majority5	9	0	0	0	2	0
	10	1	10	1	18	1
	11	22	437	40	148	56

CanFil1 and CanFil2:
Linear equations only for $m >= m_0$, independent of setup.

Contrast: CanFil5 (and also Majority5)
Linear equations exist already for m about $n/2$.

Example: CanFil5, $n = 20$, setup 4, $y = 000000$ of $m = 6$ bits.
There are exactly 2^{14} preimages, i.e., 2^{14} rows, and $D_x = 21$ columns in matrix M. M has rank 20.
One linear equation.

Observation 1: Number of equations only weakly depends on setup, but mainly depends on properties of filter function.

Observation 2: Experimental results are scalable, i.e., are likely to generalize to larger LFSR-lengths n.

Probabilistic equations

In practical situations: n is much larger than 20. Number of preimages available only small multiple of D_x .

May introduce probabilistic solutions.

Example: CanFil5, $n = 20$, setup 4, $y = 000000$
Pick instead of all 2^{14} preimages only $N' = 80$ random preimages.

Determine all solutions for much smaller matrix: Obtained always $2 – 4$ solutions, with correlations
$p = 0.98,..1$.

Majority of outputs y different from $y = 000000$ give similar picture.

p is impressively large, so that probabilistic equations are useful in attacks.

Heuristic estimate for p:

$(1\text{-}p)\ N'_r < 1$, i.e., $p = 1\text{-}\ 1/N'$

Example: CanFil5, $n = 40$, $m = 20$, y = 0,..,0, setup 4. Determine 200 preimages. Gives 11 linear relations. With 2000 preimages, only 3 out of 11 relations detected to be probabilistic.

Can find (probabilistic) equations for quite large n; e.g., CanFil5, $n = 80$, $m = 40$.

Time complexity to find a linear equation using filter inversion: 2^{32}

Discussion of attacks

Ordinary algebraic attack on filter generator with output function f :

Let f have algebraic immunity d.

Using known output, accumulate about $\binom{n}{d}$ equations of degree d with initial state bits of LFSR as unknowns, so as to do linearization.

Each new low degree equation (found by investigating augmented function of f) can serve to reduce data complexity.

Have identified functions f which show resistance to this approach:

No additional equations exist, and/or effort of finding preimages is too large.

Several other functions f shown to be weak:

Many low degree equations for augmented function of f exist and can be determined efficiently.

Particular case: n linear equations can be found (e.g. with CanFil5 or Majority functions as filters).

Data complexity of order n

Computational complexity: depends on effort for finding preimages for augmented function, but can be low (e.g. for CanFil5 and other functions).

Probabilistic equations:

linear: powerful variant of conditional correlation attack (with correlations close to 1).

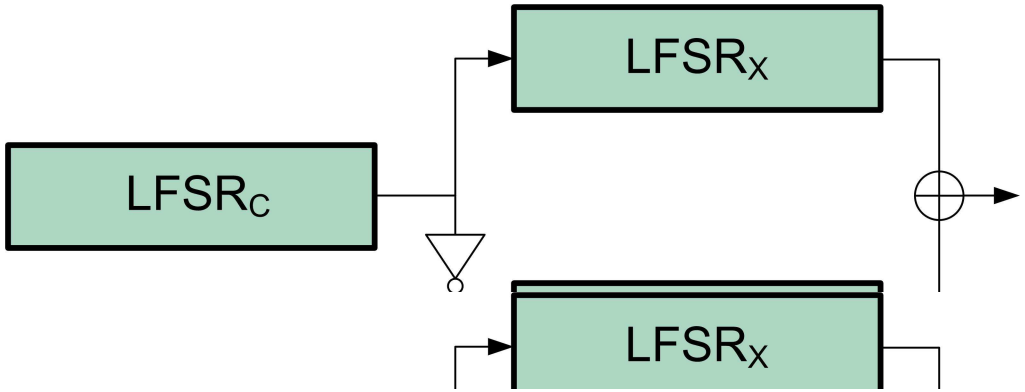
nonlinear (but low degree): kind of higher order correlation attack (Courtois)

Application of framework: Attack on ASG

Alternating step generator (ASG) is a classical construction, based on 3 LFSR's.

Output bit of regularly clocked $LFSR_C$ controls which of $LFSR_X$ and $LFSR_Y$ is clocked (stop/go LFSR's).

ASG has been target of much analysis.


LFSR$_X$
LFSR$_C$
LFSR$_X$

For ASG sampling is easy if output length m is about twice the length of stop/go LFSR's.

Another weakness: Different initial states of any of the stop/go LFSR's have far different probabilities to be accepted as a candidate which can produce a given output segment of length m.

Allows (probabilistic) attack which is about 7000 times faster than all previous attacks for many parameters (KFM, 2007).

Application of algebraic attacks to multivariate hash functions

Merkle-Damgård construction with message blocks of m field elements and chaining value of n field elements.

Compression function

$$h : K^{m+n} \rightarrow K^{n}$$

defined as explicit algebraic expressions of n component functions h_i.

Informally, a hash function h should fulfill

- Collision resistance: Finding two messages m and $m*$ with $m^* \neq m$ such that $h(m) = h(m*)$ is not easier than about $2^{n/2}$.
- Second preimage resistance: For a given message m, finding a second message $m*$ with $m^* \neq m$ such that $h(m) = h(m*)$ is not easier than about 2^n evaluations of h.
- Preimage resistance: For a given hash value v, finding a message m such that $h(m) = v$ is not easier than about 2^n evaluations of h.

NIST SHA-3 Project

Call for new proposals of hash functions.

Search for alternative structures to MD5, SHA-1, which have been broken by Wang et. al.

In multivariate hash functions, preimage resistance comes to the difficulty of solving a (nonlinear) system of multivariate equations.

For a random system with approx. the same number of equations as unknowns, this is known to be a hard problem, even for quadratic systems (degree 2).

However, for quadratic component functions h_i, can find collisions efficiently, by solving a linear system

$$h(x) - h(x-D) = 0 \quad D \neq 0$$

for an arbitrary difference

Finding collisions in a hash function of degree d reduces to solving a system of degree $d - 1$.

Sparse cubic hash (degree 3), (Ding-Yang, 2007)

Cubic components h_i, with

$$h : K^{2n} \to K^n$$

Fixed density of coefficients of 0.1%.

Sparse components: More efficient and use less memory.

No longer reduction to hard problem!

Consider security of cubic hash over GF(2), low density for cubic monomials only.

Algorithm for collision search faster than by birthday paradox (AM, 2007):

1. Compute quadratic system $h(x)$-$h(x$-$D)$
2. Remove quadratic terms to get a linear system $h'(x)=0$.
3. Compute the generating matrix of the corresponding linear code.
4. Search for a low weight code word of this code (i.e., a sparse solution of $h(x)=0$).
5. Plug solution into $h(x)$-$h(x$-$D)$: sums of quadratic terms vanish with non-negligible probability: a collision may be found.

Difficulty: find low-weight words in a random code

Fastest known algorithm in (Canteaut-Chabaud, 1998).

For a cubic system over GF(2) with 160 equations and 320 unknowns, density 0.1% for cubic monomials, but random for lower degrees:

Ratio time/success approx. 2^{52}

Hence faster than by birthday paradox, with time and memory complexity about 2^{80}.

Attacks on Stream Ciphers (Historical Overview)

Design and analyis of stream ciphers: For long time a proprietary and confidential matter.

Early research papers date from 1970‘s.

State of the art in mid 1980‘s: Book by R. Rueppel, Analysis and Design of Stream Ciphers.

Main emphasis in those days on criteria like large linear complexity and correlation immunity.

Tradeoff between correlation immunity and algebraic degree (Th. Siegenthaler, 1984)

Study of Boolean functions with good cryptographic properties has been ongoing topic since.

Impact of cryptanalysis of DES block cipher (Chaum-Evertse, 1986) to design of stream ciphers:

Alternative solution of correlation problem (MS, 1989):

Bent functions

Are however not exactly balanced.

Bent functions have:

Maximum nonlinearity, i.e., largest possible distance to all affine functions, and

Good correlation immunity properties.

Important role of functions with these criteria in design of AES block cipher, to counter differential and linear cryptanalysis

Tradeoff between correlation immunity and algebraic degree can be avoided if combining function is allowed to have memory (Rueppel, 1985)

Combiners with memory: Summation generator, Bluetooth, both based on integer addition

Different development, leading to cryptanalysis of summation generator:

Feedback with carry shift registers (FCSR‘s) (Klapper-Goresky, 1997)

Are equipped with auxiliary memory for storing an integer carry.

An FCSR is similar to an LFSR, except that the contents of the tapped stages of the shift register are added as integers to current content of the memory, to form S. The lsb of S is then fed back to be the value of the first cell of the register, and the new value of the memory is a right shift of S by 1.

Interest for cryptanalysis: Synthesis algorithm for FCSR‘s similar to Berlekamp-Massey for LFSR‘s.

Keystream may have high linear complexity but may be efficiently synthesized by a relatively short FCSR.

This happens for many parameters of summation generator.

Summation generator not as secure as believed.

However, integer addition remains essential operation in design of stream ciphers and block ciphers.

Different line (unrelated to combiners with memory):

Inversion attack (Golić, 1996), on filter generators:

May work even when filter function is correlation immune or close to Bent function,

e. g., if driving LFSR is short and tap positions are not well chosen (i.e., don‘t constitute full positive difference set).

In same paper by Golić: Updated list of design criteria for filter generators, as far as known 1996.

Another general type of attack on stream ciphers:

Free binary decision diagram attack (Krause, 2002):

Exploits that many LFSR-based stream ciphers produce keystream according to rule $b = C(L(x))$, where $L(x)$ denotes internal bitstream generated by small number of parallel LFSR‘s, and C denotes some nonlinear compression function.

Attack needs known keystream segment of length which is only small multiple of bitlength of initial state.

Attack yields security bounds lower than suggested by bitlength of initial state, e.g. for

Self-shrinking generator, A5, or Bluetooth.

According to needs for applications, there has been shift from hardware oriented stream ciphers to software oriented stream ciphers.

As LFSR‘s are not particularly efficient in software, this made different design and analysis necessary

(e.g., SNOW, or Scream: New type of analysis)

Resynchronization attacks

Some practical stream ciphers (e.g., Bluetooth, A5) use reinitialization mechanism.

This uses secret key and publicly known initialization vector. Result is a new secret initial state of keystream generator.

Reinitialization: Enables reuse of same secret key with different initialization vector. Important if synchronization is lost.

Frequent reinitialization can increase security of stream cipher (only short keystream with same initial state is exposed).

Disadvantage: Stream cipher can become completely insecure when

reinitialization mechanism is weak (e.g., linear), or

when there exist statistical dependencies of output streams for different initializations.

Cryptanalysis of stream ciphers with unknown combining function and/or unknown feedback connections

Initial steps in this direction by Palit-Roy (1999), and by Canteaut-Filiol (2000)

Algebraic attacks thus far need both, known combining function as well as known feedback connections for being applicable.

Stream cipher with key-dependent output function: Turing (Rose-Hawkes, 2003).

Open Problems

- Algebraic attacks on irregularly clocked stream ciphers?
- Design and analysis of secure software and hardware oriented stream ciphers? In particular:
- Are there efficient stream ciphers as secure as block ciphers?