T-79.514 Special Course on Cryptology

November 25th, 2004

Algebraic Attacks and Stream Ciphers

Mikko Kiviharju

Helsinki University of Technology

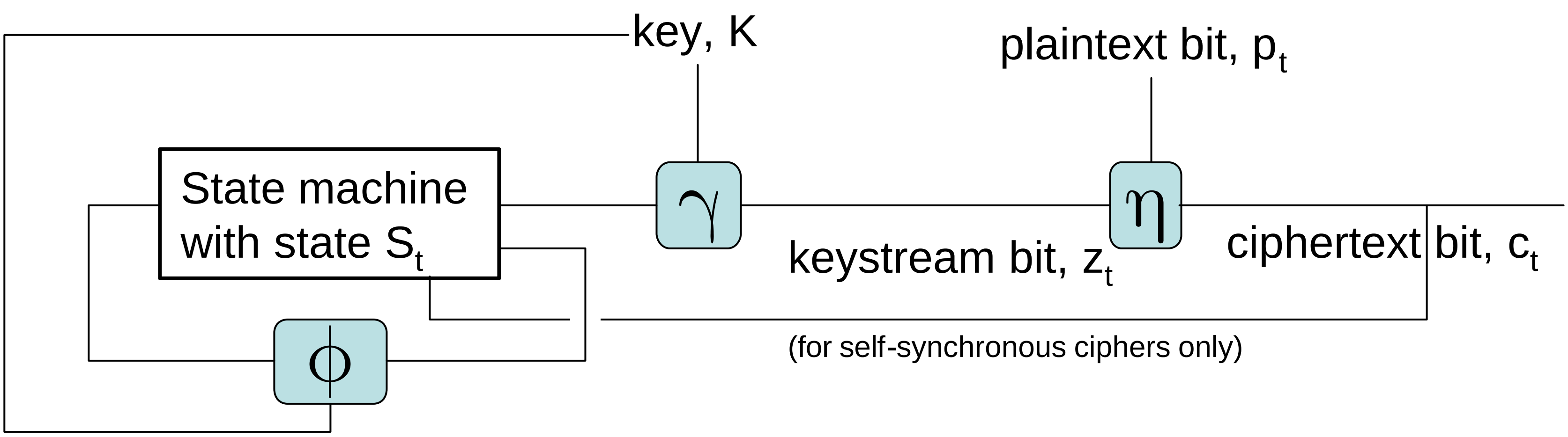
mkivihar@cc.hut.fi

Overview

- Stream ciphers and the most common attacks
- Algebraic attacks (on LSFR-based ciphers)
- Fast(er) algebraic attacks
- Case: E0
- Conclusion

Stream ciphers

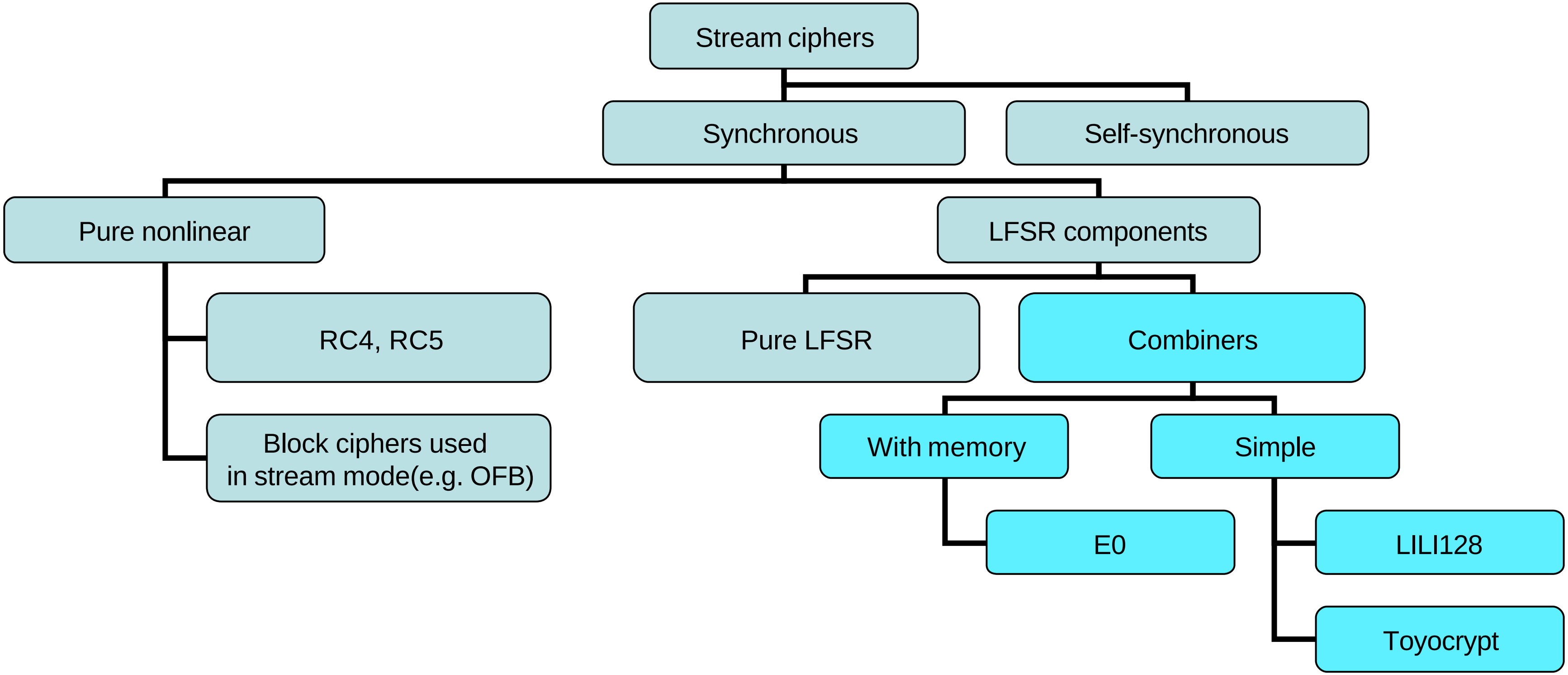
- *Stream cipher*: output stream of symbols, usually bits, is a function of plaintext and key stream symbols.
- Key stream could be anything (i.e a genuine OTP), but is usually a state machine.



Stream ciphers: attacks

- Key reuse (medieval)
- Time-memory tradeoffs (Babbage, 1995)
- Guess-and-determine (Günther, 1988)
- Correlation (Siegenthaler, 1984)
- Algebraic (Shamir et al., 1999)
- Backtracking (Golic, 1997)
- Binary Decision Diagrams (Krause, 2002)
- Side channel (Kocher et al., 1999)
- Resynchronization (Daemen et al. 1993)
- etc.

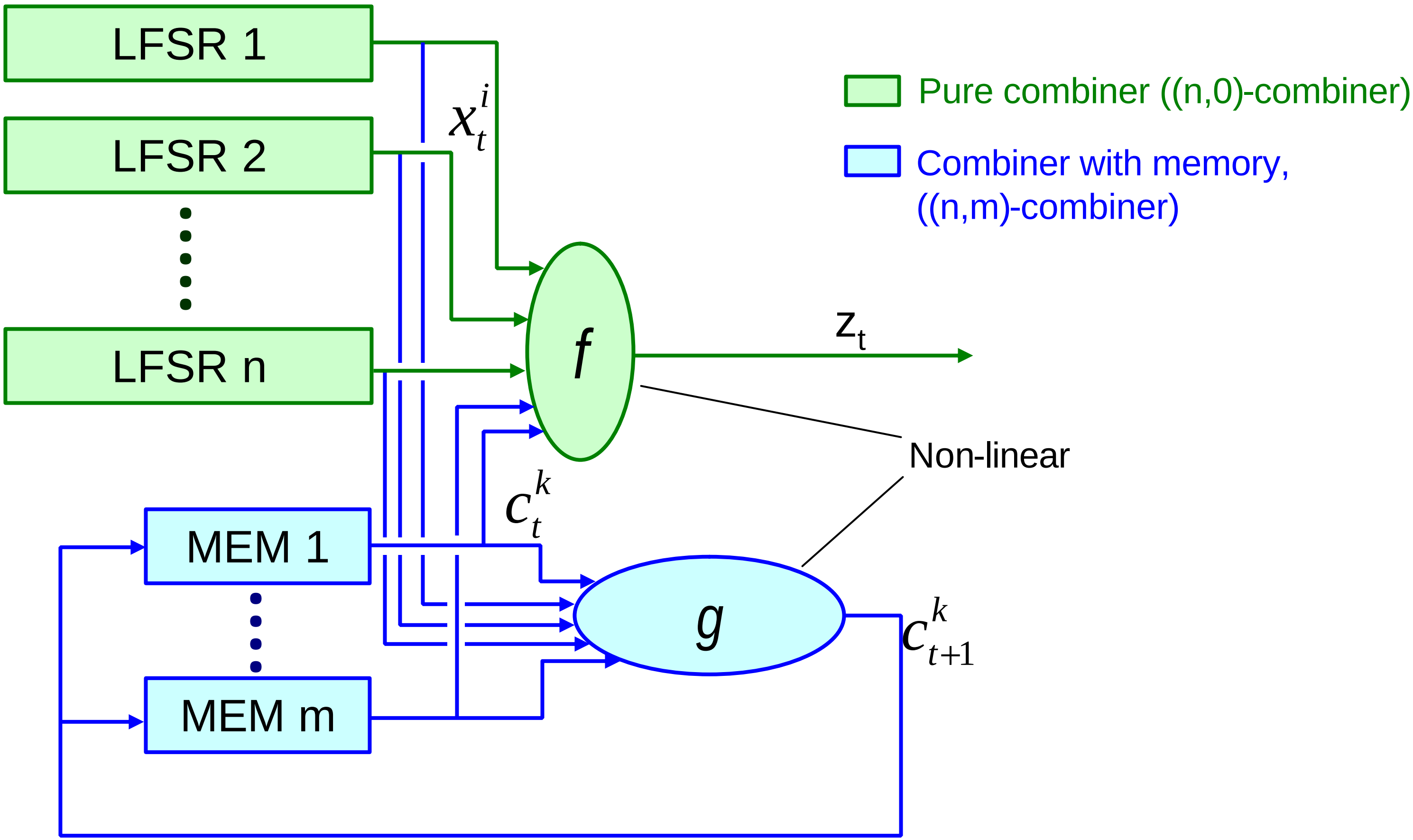
Stream ciphers: categories



Stream ciphers: combiners

- Pure LFSR-ciphers trivial to break
  - complexity $O(n^3)$, from 2n linear equations
- Add non-linearity (in $GF(2^k)$-arithmetic)
  - a non-linear function combining some LFSRs => (pure) *combiner.* Example: LILI-128
- In pure combiners, high correlation immunity implies vulnerability to algebraic attacks
- Make keystream dependent on a (non-linear) state-machine as well
  - *Combiner with memory.* Example: Bluetooth E0

Stream ciphers: combiners



Algebraic attacks

- Principle:
  - Find equations (on any cipher) with the key bits as unknowns
  - Fill in the known variables and constants
  - Solve the equation
- Problems:
  - Non-linear equations (of high degree)
  - Finite field algebras (fast methods from analysis generally not applicable, general Diophantine equations at least as hard as NP-hard)
  - Finding the equations highly dependent on the cipher
  - Inserting the keystream bits turns out to be non-trivial

Algebraic attacks: combiners

- Promising target:
  - Components mainly linear
  - Algebraic degree in real-life combiner ciphers usually of reasonable order (due to recent trends to make them correlation-immune)
- By Kerckhoff's principle the keystream z_t is known
- General idea: form equations consisting of known constants, z_t (for all t), and secret key bits of the LFSRs as unknowns.
- Combiners with memory: more unknown variables => can be cancelled, but require more known keystream

Algebraic attacks: pure combiners

Why have that $\forall t : z_t = f\left(x_t^1,\ldots,x_t^n\right)$ But each x_t^i is a linear function of the secret key bits (applied t times), so we have, for all t:

$z_t = f\left(L^t\left(k_1,\ldots,k_n\right)\right) = f\left(L^t\left(K\right)\right)$, where K represents the whole secret key and L^t is the linear function in matrix-form applied t times (raised to the t^th^ power).

Now we have $f\left(L^t\left(K\right)\right) \oplus z_t = 0$ for every clock. By Kerckhoff's principle the attacker knows all z_t, and can collect as many keystream bits as he/she likes without increasing the number of unknown variables.

Solution?

Algebraic attacks: equation solving (1)

- Task: solve non-linear diophantine system of equations
- Assume: equations are consist of polynomials (not e.g. infinite series). This is valid, since every Boolean function can be representeda as a polynomial over GF(2)
- Methods:
  - Gröbner Bases
  - Linearization (system needs to be grossly overdefined)
  - XL
  - XLS
  - …

Algebraic attacks: equation solving (2)

- Gröbner bases: ”Gaussian for non-linear systems”
  - Definition: an subset of an ideal in given polynomials is a Gröbner basis, if the ideals generated by the leading term of the whole ideal and the leading terms of the individual polynomials (in the subset) are identical
  - Usage:
    - Transform the polynomial equations to other types of polynomials (Gröbner basis) using e.g. Buchberger’s algorithm
    - A Gröbner basis has the property of Gaussian elimination, i.e. it is possible to solve one variable at a time (although still polynomial)
    - Solution to the Gröbner basis is the same as for the original equation

Algebraic attacks: equation solving (3)

- Linearization algorithms (basic, XL, XSL and variations), principle:
  - Use an overdefined equation
  - Replace each monomial with a new variable
  - Solve as a linear system

$$
\begin{aligned}
x \oplus y \oplus z &= 0 \\
x^2 \oplus xy \oplus z^2 &= 0 \\
y \oplus x^2 &= 0 \\
z^2 \oplus x^2 \oplus y &= 0 \\
xy \oplus x &= 0 \\
z^2 \oplus xy \oplus 1 &= 0
\end{aligned}
\quad \rightarrow \quad
\begin{aligned}
t &= xy \\
u &= x^2 \\
v &= z^2
\end{aligned}
\quad \rightarrow \quad
\begin{aligned}
x \oplus y \oplus z &= 0 \\
u \oplus t \oplus v &= 0 \\
y \oplus u &= 0 \\
v \oplus u \oplus y &= 0 \\
t \oplus x &= 0 \\
v \oplus t \oplus 1 &= 0
\end{aligned}
\quad \rightarrow \quad
\begin{pmatrix} x \\ y \\ z \\ t \\ u \\ v \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}
$$

Verification:

$$
\begin{aligned}
t &= 1 = 1 \cdot 1 = xy \\
u &= 1 = 1^2 = x^2 \\
v &= 0 = 0^2 = z^2
\end{aligned}
$$

Algebraic attacks: linearization

- How ”over”defined does the system need to be? (i.e: how many keystream bits are needed?)
- Upper bound for monomials of at most degree d in the equations, with n secret key bits (=unknowns):

$$M(n,d) = \sum_{i=0}^{d} \binom{n}{i} \approx O\left(n^d\right)$$

- (how many different solutions are there for exponents of a certain monomial adding up to i in GF(2))
- Exponential on the degree => lower the degree

Algebraic attacks: (n,m)-combiners (1)

In this case $\forall t : z_t = f\left(x_t^1,\ldots,x_t^n,c_t^1,\ldots,c_t^m\right)$

Each x_t^i is still a linear function of the key (applied t times), and the memory bits: $\forall t : z_t = f\left(L^t\left(k_1,\ldots,k_n\right),c_t^1,\ldots,c_t^m\right) = f\left(L^t\left(K\right),\overline{c_t}\right)$
where K and L^t are as before.

Now we have $f\left(L^t\left(K\right),\overline{c_t}\right) \oplus z_t = 0$, but collecting key bits does not help.

We could substitute all the c_t with a function of c_0, after all $\overline{c_{t+1}} = g\left(\overline{c_t}\right)$ for all t.
(c0 can be assumed to be known to the attacker) **But**: equation degree would increase exponentially with t.

Solution?

Algebraic attacks: (n,m)-combiners (2)

- Task: cancelling out the memory-bits from (n,m)-combiners
- Result by Armknecht and Krause in Crypto 2003:
  - there is a boolean function $H(\neq 0)$ of a degree at most $\left\lceil \frac{n(m+1)}{2} \right\rceil$ and an integer *r* strictly larger than the number of memory bits, such that $\forall t : H\left(L^t(K), z_t, \ldots, z_{t+r-1}\right) = 0$. Here *K* and *L* are as before.
  - Also: algorithm for finding *H,* to be *ad hoc equations*

Algebraic attacks: ad hoc equations

- Outline of proof for the upper bound
  - Define a set $Crit_c(z)$ as the set of those secret key values that do not map to given r consecutive keystream bits for any state of the memory bits. Accordingly, let $NCrit_c(z)$ be the complement of $Crit_c(z)$.
  - Show that the number of degree d polynomials that define the combiner solely based on the secret key bits equals the null space of all monomials of degree d w.r.t $NCrit_c(z)$
  - Note that the null space has a nontrivial solution iff the number of all monomials (of degree d) is greater than $NCrit_c(z)$.
  - Size of $NCrit_c(z)$ is estimated and this result is assigned to the number of all monomials, which is a function of d.
- Algorithm for finding the polynomial consists of computing the afore-mentioned null-space.

Fast algebraic attacks: reducing the degree (1)

Assume an system of equations of the form

$H\left(L^t(K), z_t, \ldots, z_{t+r-1}\right) = 0$ can be split into two halves:

$$H_1\left(L^t(K)\right) \oplus H_2\left(L^t(K), z_t, \ldots, z_{t+r-1}\right) = 0$$

such that d_1=deg(H)=deg(H_1), and d_2=deg(H_2) and $d_1 > d_2$.

H_1 only dependent on linear function of the secret key bits $\Rightarrow$ after "several" clocks the system of H_1:s will be linearly dependent. $\Rightarrow \exists a_0, \ldots, a_{h-1} : \sum_{i=0}^{h} a_i \cdot H_1\left(L^{t+i}(K)\right) = 0$. Here h is about $\binom{|K|}{d}$. (Theory of linear recurring sequences)

Fast algebraic attacks: reducing the degree (2)

Now consider $$\sum_{i=0}^{h} a_i \cdot H\left(L^{t+i}(K), z_{t+i}, \ldots, z_{t+i+r-1}\right) = 0 \Leftrightarrow$$

$$\sum_{i=0}^{h} a_i \cdot H_2\left(L^{t+i}(K), z_{t+i}, \ldots, z_{t+i+r-1}\right) = 0$$

Degree reduced, but number of needed consecutive keystream bits increased (dramatically). Operation known as *precomputation step.*

- Assumption and efficient retrieval of coefficients a_i was proven correct for most stream ciphers by Armknecht in Oct 2004 at SASC, Belgium, by associating the low-degree solutions to low-degree annihilators of Boolean functions.
- Note that H or H_2 could consist *only* of monomials containing z_i, in which case the splitting would not be possible.

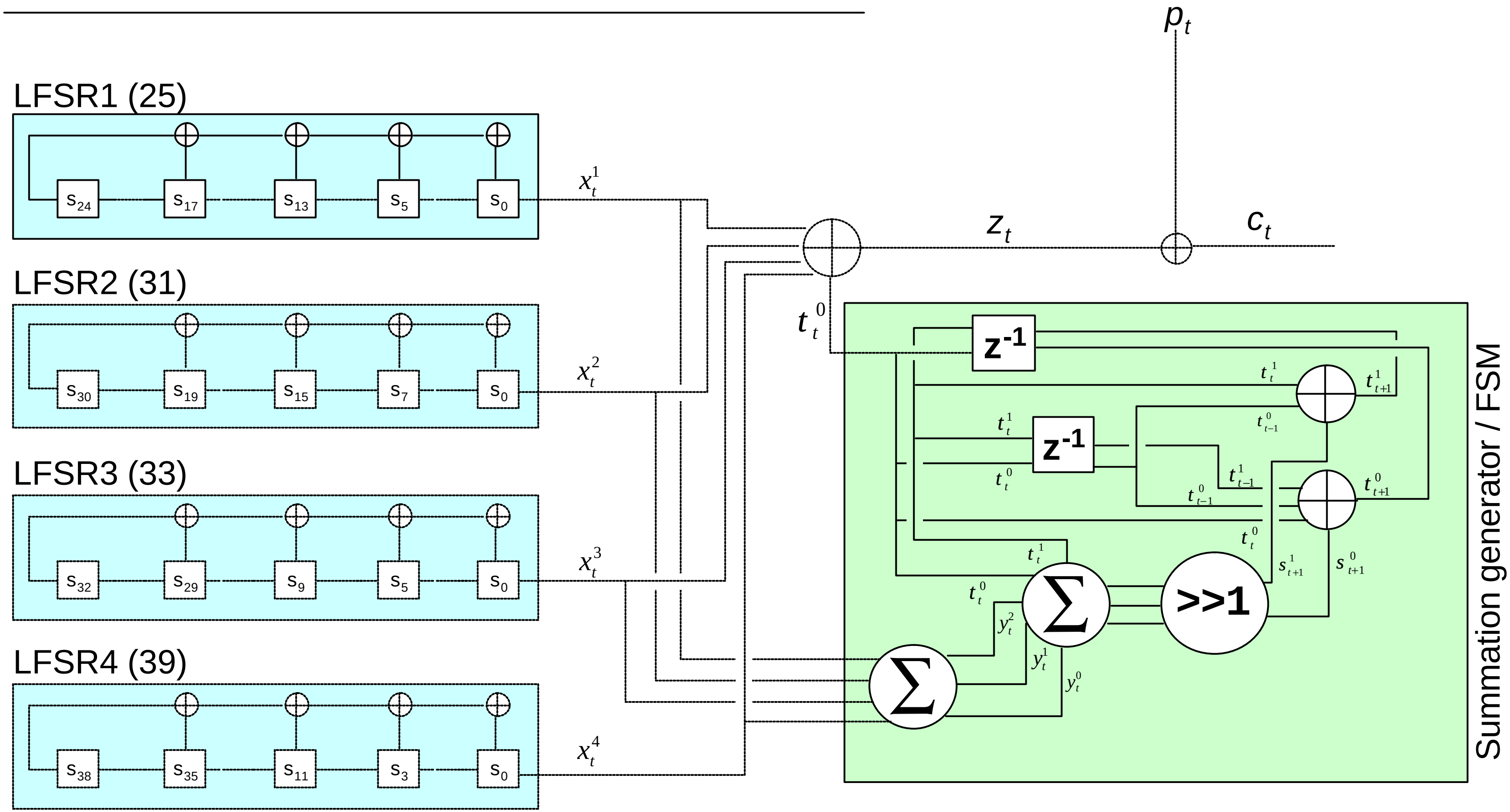
Fast algebraic attacks: precomputation step

- Coefficients computed from the minimal polynomial of the sequence $H(K)$, $H(L(K))$, $H(L^2(K))$,… (Berlekamp-Massey algorithm)
- *Problem*: polynomials generally not unique, especially not with Bluetooth E0
- *Refinement* (Armknecht, June 2004): form minimal polynomials from pairwise coprime components => parallelizable, produces unique minimal polynomials.
- *Problem:* finding pairwise coprime polynomials that are components of H
- Refinement (Armknecht, October 2004): coefficients computed with the help of Boolean annihilators

Bluetooth

- *Bluetooth:* An industry standard for small appliances connectivity on close range (PAN)
- Bluetooth security has four named algorithms:
  - *E0*: symmetric and synchronous stream cipher
  - *E1*: authentication algorithm on SAFER+
  - *E2*: authentication key generation based on SAFER+
  - *E3*: E0 key generation, SAFER+
- Bluetooth security has a number of flaws, most severe of which are not in E0. (i.e key replay attacks, encryption key length negotiation, PIN enumeration)
- This paper focuses on the encryption algorithm E0 only

Bluetooth: E0 structure

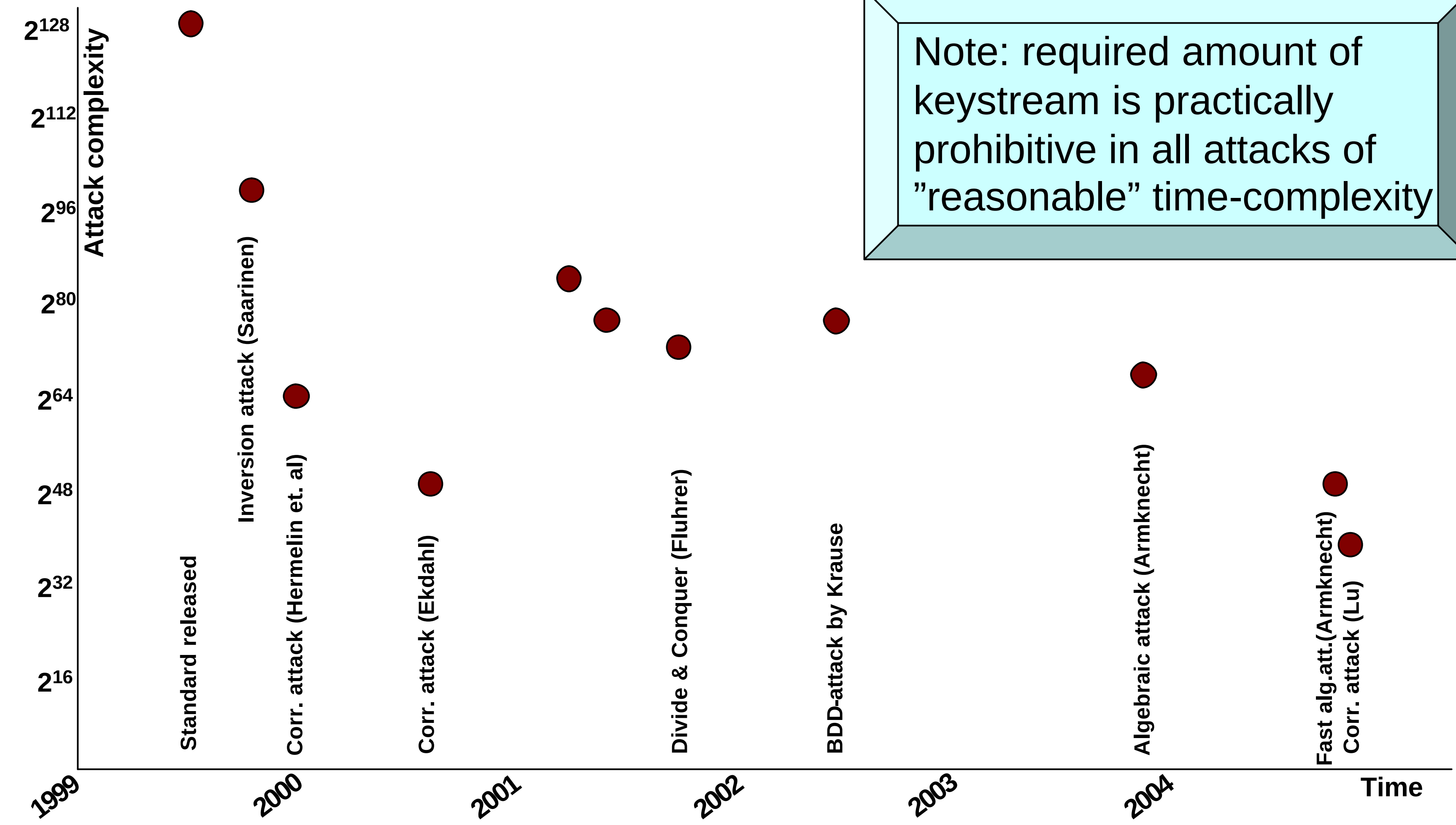


(4,4)-combiner: four LFSRs and memory bits $\left(t_{t-1}^{0}, t_{t-1}^{1}, t_{t}^{0}, t_{t}^{1}\right)$

Bluetooth: E0 initialization

- Two level operation
  - Level 1: Initialisation of the summation generator and the LSFRs for Level 2
  - Level 2: Actual keystream generation
- Level 1 initialises its LFSR block with the key XORed with nonce and FSM block is reset
- The level 1 – blocks clocked 200 times
- The last 128 output (keystream) bits are fed into a permutation function
- The output of the permutation forms the initial state of the level 2 LFSR blocks. Level 2 FSM block is initialised to the final state of the level 1 FSM block

Bluetooth: attacks on E0



Note: required amount of keystream is practically prohibitive in all attacks of ”reasonable” time-complexity

Bluetooth: ad hoc equation for E0

- Prediction: degree at most 10, dependency of at most 5 consecutive keystream bits
- Practice: degree 4, dependency of 4 consecutive bits

$$
\begin{aligned}
G\left(L^t(K), z_t, z_{t+1}, z_{t+2}, z_{t+3}\right) = {} & z_t \oplus z_{t+1} \oplus z_{t+2} \oplus z_{t+3} \oplus p_{t+1}^{2} \cdot \left(z_t \oplus z_{t+1} \oplus z_{t+2} \oplus z_{t+3}\right) \oplus p_{t+1}^{4} \\
& \oplus p_{t+1}^{1} \cdot \left(z_t \oplus z_{t+2} \oplus z_{t+3} \oplus z_{t+1} \cdot z_t \oplus z_{t+1} \cdot z_{t+2} \oplus z_{t+1} \cdot z_{t+3}\right) \\
& \oplus p_{t}^{1} \oplus p_{t}^{1} \cdot p_{t+1}^{1} \cdot \left(1 \oplus z_{t+1}\right) \oplus p_{t}^{1} \cdot p_{t+1}^{2} \\
& \oplus p_{t+2}^{1} \cdot z_{t+2} \oplus p_{t+2}^{1} \cdot p_{t+1}^{1} \cdot z_{t+2} \cdot \left(z_{t+1} \oplus 1\right) \oplus p_{t+2}^{1} \cdot p_{t+1}^{2} \cdot z_{t+2} \\
& \oplus p_{t+2}^{2} \oplus p_{t+2}^{2} \cdot p_{t+1}^{1} \cdot \left(1 \oplus z_{t+1}\right) \oplus p_{t+2}^{2} \cdot p_{t+1}^{2} \\
& \oplus p_{t+3}^{1} \oplus p_{t+3}^{1} \cdot p_{t+1}^{1} \cdot \left(1 \oplus z_{t+1}\right) \oplus p_{t+3}^{1} \cdot p_{t+1}^{2} \\
= {} & 0
\end{aligned}
$$

(where p_t^i is the i^{th} elementary symmetric polynomial in the unknown outputs of the four LFSRs)

Bluetooth: analysis of E0

- Fast algebraic attack: Decomposition into G_1 and G_2, where $G = G_1 \oplus G_2$ and $G_1\left(L^{t+i}(K)\right) = p_{t+i+1}^{4} \oplus p_{t+i+2}^{2} \cdot p_{t+i+1}^{2}$ and deg(G_2)=3.
- Armknecht's results on Boolean annihilators: the size of E0's characteristic function's ”one-set” (the set of arguments which makes the function-value = 1) is too big to allow annihilators of degree < 3. => Described attack is of optimal order of complexity.
- Attack complexity: Number of monomials and solved with Strassen (e.g) $7 \cdot \left[\binom{128}{0} + \binom{128}{1} + \binom{128}{2} + \binom{128}{3}\right]^{\log_2 7} \approx 2^{54,51}$
- Number of successive keystream bits: $\binom{128}{4} \approx 2^{23}$. Infeasible, given at most 2744 bits per frame and same key.

Bt: combined algebraic and resync?

- *What if:* algebraic attack over several frames? *Resync?*
- Armknecht's results on combining resynchronisation attacks with algebraic attacks (SAC '04), **but:**
  - only for pure combiners
- Extendable to combiners with memory, **but**:
  - workload is increased exponentially on the number of memory bits
- Ad hoc equations ok, **but:**
  - known construction methods do not extend over permutation (=non-linear) function (the one between E0 levels 1 and 2)
- Room for future ideas…

Conclusion and open questions

- Algebraic attacks one of the newest and most efficient forms of cryptanalytic attacks, especially with stream ciphers
- Correlation attacks less time-consuming, but alg. attack need less data
- Tools and criteria for providing security against algebraic attacks evolving (e.g. Meier et al, Eurocrypt 2004)
- Bluetooth E0 is ”broken”, but only in academic sense.

- Can ad hoc equations be formed for systems with non-linearity in the input? (Two levels of E0)
- When is it possible to use the idea of fast algebraic attacks (i.e. reduction of the degree of polynomials) iteratively?