Low-Weight Common Multiples of Binary Primitive Polynomials through Discrete Logarithms

Pietro Peterlongo, Claudia Tinnirello, Massimiliano Sala
(Department of Mathematics, University of Trento)

YACC 2014
Ile de Porquerolles
June, 12th

1. Introduction
   - basic definitions and notation
   - basic facts and problem setting
2. Motivation
   - a class of stream ciphers
   - an example of correlation attack
   - the case of [Lu Vaudenay 2004a] attack to E0
   - Birthday-based approaches
3. Algorithm
   - requirements
   - case of a single primitive polynomial (Zech logarithms)
   - case of r primitive polynomials
   - Full pseudocode and complexity
   - Example output for case of E0
4. Conclusions

Outline

All polynomials will be in $\mathbb{F}_2[x]$ (binary polynomials).

Definition

The *weight* of a polynomial is the number of nonzero coefficients.

Definition

The *order* of a polynomial p is the smallest M such that $x^M - 1$ is a multiple of p.

If p is irreducible, its order is the same as the order of all its roots.

Definition

A *primitive* polynomial of degree m is the minimal polynomial of a primitive element of the finite extension field $\mathbb{F}_{2^m}$. Equivalently, it is a polynomial with order $2^m - 1$.

A primitive polynomial is irreducible, but the converse is not true (example: $x^4 + x^3 + x^2 + x + 1$).

Left Shift Register Sequence (LFSR)

Definition

An LFSR of length L over $\mathbb{F}_2$ is a finite state automaton which, starting from an initial state $(x_0, \ldots, x_{L-1})$, produces a semi-infinite sequence of bits x_t satisfying a linear recurrence relation of degree L.

$$x_{t+L} = \bigoplus_{i=1}^{w-1} x_{t+\eta_i}$$

where η_i are integers in $[0, L-1]$.

Associated to an LFSR is its *feedback polynomial* (of weight w) given by

$$1 + x^{\eta_1} + \ldots + x^{\eta_{w-1}}$$

Left Shift Register Sequence (LFSR)

Definition

An LFSR of length L over $\mathbb{F}_2$ is a finite state automaton which, starting from an initial state $(x_0, \ldots, x_{L-1})$, produces a semi-infinite sequence of bits x_t satisfying a linear recurrence relation of degree L.

$$x_{t+L} = \bigoplus_{i=1}^{w-1} x_{t+\eta_i}$$

where η_i are integers in $[0, L-1]$.

Associated to an LFSR is its *feedback polynomial* (of weight w) given by

$$1 + x^{\eta_1} + \ldots + x^{\eta_{w-1}}$$

If this polynomial is irreducible, all other linear recurrence relations (*parity checks*) are associated to a multiple of the feedback polynomial. Primitive polynomials produce sequence with maximal period $(2^L - 1)$.

We sometimes denote a binary polynomial by its exponents.

Example

$$[5,2,0] := x^5 + x^2 + 1 \quad \text{and} \quad [3,2,0] := x^3 + x^2 + 1$$

are weight 3 primitive polynomials of order, respectively, 31 and 7. Their product is the weight 5 polynomial $[8,7,4,3,0]$.

With this notation the remainder of division of a polynomial by $x^N + 1$ is reduction modulo N.

Example (continued)

$[129,2,0]$ is a weight 3 multiple both of $[5,2,0]$ and $[3,2,0]$, since

$$[129,2,0] \bmod 31 = [5,2,0]$$

and

$$[129,2,0] \bmod 7 = [3,2,0].$$

Lemma

Let p be an irreducible polynomial and let α be a root of p in an extension field $\mathbb{F}_{2^m}$. Then for a polynomial q we have $q(\alpha) = 0$ if and only if p divides q.

this implies that

- all multiples of an irreducible polynomials p are defined up to division by $x^D + 1$, where D is the order of p.
- all multiples of a product of distinct irreducible polynomials are defined up to division by $x^N + 1$, where N is the least common multiples of the orders of the polynomials.

Lemma

Let p be an irreducible polynomial and let α be a root of p in an extension field $\mathbb{F}_{2^m}$. Then for a polynomial q we have $q(\alpha) = 0$ if and only if p divides q.

this implies that

- all multiples of an irreducible polynomials p are defined up to division by $x^D + 1$, where D is the order of p.
- all multiples of a product of distinct irreducible polynomials are defined up to division by $x^N + 1$, where N is the least common multiples of the orders of the polynomials.

Problem setting

Given a polynomial p of order N and a weight $w \geq 3$ find all w-uples $(e_1, \ldots, e_w)$ with $e_j \in \mathbb{Z}_N$ for $j = 1, \ldots, w$ and such that $[e_1, \ldots, e_w]$ is a multiple of p.

We will assume without losing generality that $e_w = 0$.

Outline

In order to motivate the problem, we discuss in the following slides an attack scheme taken from [Lu Vaudenay 2004a]. It is a type of fast correlation attack against the keystream generator of $E0$, the stream cipher used for encryption in Bluetooth Protocol.

Nonlinear combiner with memory

A nonlinear combiner with memory is a keystream generator composed by:

- r LFSRs with primitive feedback polynomials $p_1, \ldots, p_r$.
  Output at time t will be denoted by x_t^i.
- a memory (an additional register) with an update function that depends both on memory bits and register outputs.
- a nonlinear combiner, that is a nonlinear Boolean function that takes as input the memory bits and x_t^i's and outputs a single keystream bit z_t.
  We can always think that

$$z_t = x_t^1 \oplus \cdots \oplus x_t^r \oplus c_t$$

  putting all nonlinear dependance on c_t.

In our context, a correlation is a bias ε for a bit written as $\bigoplus_{k=1}^{B} c_{t+\gamma_k}$ for certain integers $\gamma_1, \ldots, \gamma_B$.
After one (or more) correlations are found, in order to perform a partial key-recovery attack on one of the registers the effects of the other registers should be cancelled.

In our context, a correlation is a bias ε for a bit written as $\bigoplus_{k=1}^{B} c_{t+\gamma_k}$ for certain integers $\gamma_1, \ldots, \gamma_B$.
After one (or more) correlations are found, in order to perform a partial key-recovery attack on one of the registers the effects of the other registers should be cancelled.

This is the moment where finding low-weight polynomial multiples becomes important!

Let us concentrate on attacking the first register and let $P(x)$ be a polynomial of weight w which is a common multiple of $p_2, \ldots, p_r$. We denoted by η_i with $i = 1, \ldots, w$ the exponents of the monomials appearing in $P(x)$ (we assume $\eta_1 = 0$ and $\eta_w = D$ with D the degree of P). By elementary properties of LFSRs we have that $\bigoplus_{i=1}^{w} x^j_{t+\eta_i} = 0$ for every $j = 2, \ldots, w$. Thus we have that

$$\bigoplus_{i=1}^{w} \left(\bigoplus_{k=1}^{B} z_{t+\eta_i+\gamma_k} \right) = \bigoplus_{i=1}^{w} \left(\bigoplus_{k=1}^{B} (x^1_{t+\eta_i+\gamma_k} \oplus c_{t+\eta_i+\gamma_k}) \right)$$

The bias is thus ε^w and the data complexity Q of the attack is bounded from below by $\frac{1}{\varepsilon^{2w}}$ and by D the degree of the polynomial multiple.
It is therefore important to have a target degree D and try to find polynomials of degree less than D.
Also, the weight w should be small. It usually is $w = 3, 4, 5$.

Polynomials of E0

The following are the 4 primitive polynomials (of weight 5) used in the LFSRs of E0

$$p_1 = x^{25} + x^{20} + x^{12} + x^8 + 1$$

$$p_2 = x^{31} + x^{24} + x^{16} + x^{12} + 1$$

$$p_3 = x^{33} + x^{28} + x^{24} + x^4 + 1$$

$$p_4 = x^{39} + x^{36} + x^{28} + x^4 + 1$$

The key-recovery attack described in [Lu Vaudenay 2004a] assumes that one can find:

- a weight 5 multiple of p_2, p_3, p_4 of degree $\leq 2^{34.3}$ with (precomputation) time complexity $2^{36.3}$;
- and a weight 3 multiple of p_3, p_4 of degree $\leq 2^{36}$ with (precomputation) time complexity 2^{37}.

Heuristic on low-weight polynomial multiples

Given a polynomial of degree n and a weight w, we want to know which is the degree at which we can find multiples with weight w. A first answer uses the following:

Statistical assumption

The multiples of degree at most D of a polynomial of degree n has weight w with probability $\simeq \binom{D}{w-1} 2^{-D}$

The expected number of the polynomials multiples of weight w can be estimated for large D as:

$$N_{n,w} \simeq \frac{\binom{D}{w-1}}{2^n} \simeq \frac{D^{w-1}}{(w-1)!2^n}$$

Birthday-based approaches

The critical degree where polynomials of degree w will start to appear is

$$D_0 \simeq (w-1)!^{\frac{1}{w-1}} \cdot 2^{\frac{n}{w-1}}$$

There is a basic method based on conventional birthday paradox which finds the multiple of weight w with minimal degree D_0 with time complexity

$$O(D_0^{\lceil \frac{w-1}{2} \rceil})$$

Another method is based on generalized birthday problem ([Wagner 2002]) and finds a multiple of same weight but higher degree ($D_1 \simeq 2^{\frac{n}{\lfloor \log(w-1) \rfloor}}$) in less time:

$$O((w-1) \cdot D_1)$$

Memory complexity for birthday based methods is high (order of D)

Outline

We want an algorithm that does the following:

Requirements

Input:

$p_1, \ldots, p_r$: r primitive polynomials over $\mathbb{F}_2$
w: weight ($w \geq 3$)
D: a target degree

Output:

P: a polynomial multiple of $p_1, \ldots, p_r$ with weight w and degree $\leq D$

We will break down the solution into steps:

1. A multiple of weight w for a primitive polynomial.
2. A common multiple between r primitive polynomials with coprime degrees.
3. How do we target a degree D.
4. What happens in the case of nonprime degrees.

case $r = 1$, introducing zech logarithms ($w = 3$)

Let p be a primitive polynomial of degree m and let α be the primitive root of order $M := 2^m - 1$.

Definition

The Zech Logarithm with base α of an integer i is the integer j such that $\alpha^j = 1 + \alpha^i$ (it is the discrete logarithm of $1 + \alpha^i$) and will be denoted by $Z_\alpha(i) = j$.
When $i \bmod M = 0$, we have that $1 + \alpha^i = 0$ and we will say that $Z_\alpha(i)$ is not defined.

The calculation of a single Zech Logarithm gives a trinomial multiple of the primitive polynomial p. In fact we have that

$$1 + \alpha^i = \alpha^j \leftrightarrow 1 + \alpha^i + \alpha^j = 0 \leftrightarrow p \mid 1 + x^i + x^j$$

In [Didier Laigle-Chapuy 2007] the case of a single primitive polynomial is treated with discrete logarithms and a time-memory tradeoff approach.

case $r = 1$, $w > 3$

Let $w > 3$ and let $0 < e_1 < \ldots < e_{w-2} < M$. We have that:

$$1 + \alpha^{e_1} + \alpha^{e_2} + \ldots + \alpha^{e_{w-2}} =$$
$$1 + \alpha^{e_1}\left(1 + \alpha^{e_2 - e_1}\left(\cdots\left(1 + \alpha^{e_{w-3} - e_{w-2}}(1 + \alpha^{e_{w-2}})\right)\cdots\right)\right)$$

Thus, if we can compute the following chain of Zech Logarithms:

$$e_{w-1} := Z(e_1 + Z(\alpha^{e_2 - e_1} + Z(\cdots + Z(e_{w-3} - e_{w-2} + Z(e_{w-2}))\cdots)))$$

we have that $[0, e_1, \ldots, e_{w-1}]$ is a multiple of p.

Remark

Since Z is a bijection when restricted from $(0, M)$ to $(0, M)$, when choosing the integers $e_1, \ldots, e_{w-2}$ randomly in $(0, M)$ the chance of being able to compute the chain of Zech Logarithms is very high (you fail in at most $w - 3$ cases over M^{w-2}).

We will denote the above chain of Zs as $\mathrm{ZLogs}_\alpha([e_1, \ldots, e_{w-2}])$.

r primitive polynomials with coprime orders

Let $p_1, \ldots, p_r$ be r primitive polynomials and let $\alpha_1, \ldots, \alpha_r$ be their primitive roots. We suppose for the moment that the orders $N_1, \ldots, N_r$ are mutually coprime, and that we know already for each $i = 1, \ldots, r$ a multiple of p_i of weight w, which will be denoted by:

$$[0, e_{i,1}, \ldots, e_{i,w-1}]$$

where all $e_{i,j} \in (0, N_i)$.
Setting N the least common multiple of $N_1, \ldots, N_r$, using Chinese Remainder Theorem (CRT), we obtain a polynomial of weight w with exponents $e_j \in (0, N)$ which is a common multiple of all p_i (for $j = 1, \ldots, w-1$; we put $e_w = 0$). We will denote this computation with:

$$e_j \leftarrow \mathrm{CRT}([e_{1,j}, \ldots, e_{r,j}], [N_1, \ldots, N_r])$$

Targeting a degree D

Instead of using CRT for $j = 1, \ldots, w-1$, if we want to target a degree D we proceed as follows:

1. Select $w-2$ random distinct integers $e_1 < e_2 < \ldots < e_{w-2} < D$.
2. Compute for each $i = 1, \ldots, r$ the remainders $e_{i,1}, \ldots, e_{i,w-2}$ mod N_i.
3. Compute (if possible, using ZLogs) for each i the 'missing' exponent $e_{i,w-1}$ to have that $[0, e_{i,1}, e_{i,2}, \ldots, e_{i,w-1}]$ is a multiple of p_i.
4. 'Lift' the missing exponents $e_{i,w-1}$ to an exponent $e_{w-1} \in (0, N)$ through CRT.
5. If $e_{w-1} \leq D$ we have that $[0, e_1, \ldots, e_{w-1}]$ is a polynomial of weight w which is a multiple of $p_1 \cdot p_2 \cdots p_r$ and has degree $\leq D$. Otherwise repeat from step 1.

The case of non coprime orders

What happens if $(N_i, N_j) \neq 1$ for some (i, j)?

The CRT is able to compute the exponent e_{w-1} if

$$e_{i,w-1} \equiv e_{j,w-1} \pmod{(N_i, N_j)}$$

Otherwise we repeat the cycle...

Pseudocode for the algorithm

1: **function** $(p_1, \ldots, p_r, w, D)$
2: $\quad \alpha_i \leftarrow \mathrm{PrimitiveRoot}(p_i)$
$\quad \triangleright$ All lines with i repeat for $i = 1, \ldots, r$
3: $\quad N_i \leftarrow 2^{\deg(p_i)} - 1$
4: $\quad$ **repeat**
5: $\quad\quad [e_1, \ldots, e_{w-2}] \leftarrow \mathrm{RandomDistinctLessThan}(D)$
6: $\quad\quad [e_{i,1}, \ldots, e_{i,w-2}] \leftarrow [e_1, \ldots, e_{w-2}] \mod N_i$
7: $\quad\quad e_{i,w-1} \leftarrow \mathrm{ZLogs}_{\alpha_i}([e_{i,1}, \ldots, e_{i,w-2}])$
$\quad\quad \triangleright$ If not possible, restart the cycle
8: $\quad\quad e_{w-1} \leftarrow \mathrm{CRT}([e_{1,w-1}, \ldots, e_{r,w-1}], [N_1, \ldots, N_r])$
$\quad\quad \triangleright$ If not possible, restart the cycle
9: $\quad$ **until** $e_{w-1} \leq D$
10: $\quad$ **return** $1 + x^{e_1} + \ldots + x^{e_{w-1}}$
11: **end function**

Note that memory complexity is $O(1)$

Time Complexity of the algorithm

Statistical Assumption

All Zech Logarithm computation Z_α with random input produce an output which is uniformly random over $(0, M)$, where M is the order of α.

Using this assumption we estimate the number of tentative e_{w-1} to compute before succeeding as $O(N/D)$.
If there are non coprime factors only a fraction of the cycles produce a valid e_{w-1} and we must multiply by the factor $P := \Pi_{(i,j),i\neq j}(N_i, N_j)$ (this might be a big constant).
During each cycle the most demanding computation is that of the Zech Logarithm, which we will estimate with a constant C (which depends on highest prime factor of the orders). This computation is done at most $(w-2)r$ times.
Thus the time complexity is

$$O((1+C)(w-2)rP \cdot \frac{N}{D})$$

Some experiments

Recall that p_1, p_2, p_3, p_4 of E0 have degrees $25, 31, 33, 39$.
Note that $(N_3, N_4) = 2^3 - 1 = 7$.
Most difficult logarithm is the one relating to p_2.

Experiment 1

Try to find a polynomial multiple of $p_1 \cdot p_3 \cdot p_4$ ($N \approx 2^{97}$) with weight $w = 5$ and targeting degree $D \leq 2^{35}$. Best result after $\approx 2^{30}$ successful cycles (over a total of 7 times more cycles)

$$[4378792629032416110 38,$$
$$10286802898, 13210333327, 28706973559, 0]$$

which has degree $\approx 2^{68.6}$.

Some experiments

Recall that p_1, p_2, p_3, p_4 of E0 have degrees $25, 31, 33, 39$.
Note that $(N_3, N_4) = 2^3 - 1 = 7$.
Most difficult logarithm is the one relating to p_2.

Experiment 2

Try to find a polynomial multiple of $p_2 \cdot p_3 \cdot p_4$ ($N \approx 2^{103}$) with weight $w = 5$ and targeting degree $D \leq 2^{35}$. Best result after $\approx 2^{26}$ successful cycles (over a total of 7 times more cycles)

$$[1299390329503626986044 4,$$
$$25576778776, 27393341749, 31182294315, 1004869052, 0]$$

which has degree $\approx 2^{73.5}$.

Some experiments

Recall that p_1, p_2, p_3, p_4 of E0 have degrees $25, 31, 33, 39$.
Note that $(N_3, N_4) = 2^3 - 1 = 7$.
Most difficult logarithm is the one relating to p_2.

Experiment 3

Try to find a polynomial multiple of $p_2 \cdot p_3 \cdot p_4$ ($N \approx 2^{103}$) with weight $w = 3$ and targeting degree $D \leq 2^{60}$. Best result after $\approx 2^{20.6}$ successful cycles (over a total of 7 times more cycles)

$$[1282348956133250774387 99,$$
$$1018121256595116545, 0]$$

which has degree $\approx 2^{76.8}$.

Some experiments

Recall that p_1, p_2, p_3, p_4 of E0 have degrees $25, 31, 33, 39$.
Note that $(N_3, N_4) = 2^3 - 1 = 7$.
Most difficult logarithm is the one relating to p_2.

Experiment 4

Try to find a polynomial multiple of $p_3 \cdot p_4$ ($N \approx 2^{72}$) with weight $w = 3$ and targeting degree $D \leq 2^{37}$. Best result after $\approx 2^{22.9}$ successful cycles (over a total of 7 times more cycles)

$$[170725371212982,$$
$$19352428054, 0]$$

which has degree $\approx 2^{47.3}$.

Outline

Conclusions

We have described a log-based approach to find low-weight multiples of a given polynomial.
With respect to existing birthday-based approaches, it has better time complexity for a range of parameters not usually useful for applications (it is better when D is nearer to N), but in some cases it is competitive with respect to time complexity (and in this case it is preferable since it has minimal memory requirements).
The analysis of the algorithm has also shown that it is able to capture different phenomena than those that are invisible to a birthday-based approach.

Future work

Directions of future work:

- extend the algorithm to irreducible not primitive polynomial and to power of primitive polynomials (already done).

Future work

Directions of future work:

- extend the algorithm to irreducible not primitive polynomial and to power of primitive polynomials (already done).
- try to find a time-memory tradeoff approach.

Future work

Directions of future work:

- extend the algorithm to irreducible not primitive polynomial and to power of primitive polynomials (already done).
- try to find a time-memory tradeoff approach.
- use properties of Zech logarithms to 'guide' the random search faster to the target.

Future work

Directions of future work:

- extend the algorithm to irreducible not primitive polynomial and to power of primitive polynomials (already done).
- try to find a time-memory tradeoff approach.
- use properties of Zech logarithms to 'guide' the random search faster to the target.
- **question**: can we find primitive polynomials with no low-weight multiples?

Future work

Directions of future work:

- extend the algorithm to irreducible not primitive polynomial and to power of primitive polynomials (already done).
- try to find a time-memory tradeoff approach.
- use properties of Zech logarithms to 'guide' the random search faster to the target.
- **question**: can we find primitive polynomials with no low-weight multiples?

Thank you for the attention!