Using Wiedemann’s algorithm to compute the algebraic immunity

Frédéric Didier
INRIA, projet CODES
Outline

1. Introduction
2. Linear algebra and algebraic immunity
3. Wiedemann’s algorithm
4. Results
Part 1

Introduction
A stream cipher: the filtered LFSR

Encoding/Decoding:
XOR the keystream bits ($z_i$) with the message bits

Secret Key: the initial state of the pseudo random generator
Algebraic attacks principle

Algebraic Normal Form (ANF) of a Boolean function $f$:

$$f(x_1, \ldots, x_n) = \sum_{m \in F_2^n} f_m x_1^{m_1} \cdots x_n^{m_n} \quad f_m \in F_2$$

$$f(x) = \sum_{m \in F_2^n} f_m x^m$$

Basic idea: solve the algebraic system given by

$$f(L^i(s_1, \ldots, s_n)) = z_i \quad \forall i \geq 0$$

Problem: The degree of $f$ is usually too high
Algebraic attacks [Courtois Meier 03]

If we find a low degree Boolean function $g$ such that

$$\forall x \in \mathbb{F}_2^n \quad f(x)g(x) = 0$$

It is called an annihilator of $f$ and we have

$$\forall x \in \mathbb{F}_2^n \quad f(x) = 1 \quad \implies \quad g(x) = 0$$

So we get a new system in the degree of $g$

$$g(L^i(s_1, \ldots, s_n)) = 0 \quad \text{for } i \geq 0 \text{ and } z_i = 1$$

Remark: when $z_i = 0$ we use annihilators of $1 + f$
Our issue: computing annihilators of $f$

We can use Gröbner basis:

Find low degree polynomial in the ideal

$$\langle 1 + f(X_1, \ldots, X_n), X_1^2 - X_1, \ldots, X_n^2 - X_n \rangle$$

But the linear algebra approach is more efficient in practice

Remark: We place ourselves in the case of a general $f$
Part 2

Linear algebra and algebraic immunity
Goal: existence/computation of relations

Let
• \( f \) be an \( n \)-variable Boolean function \((F_2^n \rightarrow F_2)\)
• \( d \) and \( e \) be given degrees \((e \geq d)\)

For normal algebraic attacks

\( g \) \hspace{1cm} \text{deg}(g) \leq d \text{ and } fg = 0

For fast algebraic attacks

\( g \) and \( h \) \hspace{1cm} \text{deg}(g) \leq d, \text{deg}(h) \leq e \text{ and } fg = h
Degree at most $d$ Boolean function space

Using the Algebraic Normal Form, a $g$ in this space can be written in a unique way as

$$g(x) = \sum_{|m| \leq d} g_m x^m$$

$|m| \overset{\text{def}}{=} \sum_i m_i$

Basis : $(x^m)_{|m| \leq d}$

Dimension : $D \overset{\text{def}}{=} \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{d}$

So we will represent $g$ by $D$ coefficients $(g_m)_{|m| \leq d}$ in $\mathbb{F}_2$
A Boolean functions $g$ can be represented by its images list

$$g(0), g(1), \ldots, g(2^n - 1)$$

$\text{RM}(d,n)$ is by definition the space of all Boolean functions of degree at most $d$ with this representation.

Previous slide: usual encoding for Reed-Muller codes
Degree $d$ evaluation matrix

Let
- $\{m_1, \ldots, m_D\}$ an order on $\{m \in F^n_2, |m| \leq d\}$
- $\{x_1, \ldots, x_N\}$ a set of $N$ points in $F^n_2$

$$V^d_{\{x_1, \ldots, x_N\}} \overset{\text{def}}{=} \begin{pmatrix} x_{i m_j} \\ \vdots \\ x_{i m_D} \end{pmatrix}_{i \in \{1, \ldots, N\}, j \in \{1, \ldots, D\}}$$

For $g$ such that $g(x) = \sum_{i=1}^{D} g_{m_i} x^{m_i}$ \hspace{1cm} ($\deg(g) \leq d$)

$$V^d_{\{x_1, \ldots, x_N\}} \begin{pmatrix} g_{m_1} \\ \vdots \\ g_{m_D} \end{pmatrix} = \begin{pmatrix} g(x_1) \\ \vdots \\ g(x_N) \end{pmatrix}$$
Basic Algebraic attacks

Find $g$ of degree $\leq d$ such that $fg = 0$

$$\forall x \in \{x, f(x) = 1\} \quad g(x) = 0$$

Finding $g$ is the same as solving the $|f| \times D$ system

$$V^d_{\{x, f(x) = 1\}} \bar{g} = 0$$

where the $D$ unknown coeffs of $g$ are in the vector $\bar{g}$
**Equivalent problems**

$V_{F_2}^d$ is actually the usual generator matrix of $\text{RM}(d,n)$

Solving the system $V_{\{x_1,\ldots,x_N\}}^d \bar{g} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$

- codewords/functions for which only the values at positions $x_i$ are known (and equal to $y_i$)

- Decoding over the erasure channel

- Multivariate interpolation problem
Fast Algebraic attacks

$g$ of degree $\leq d$ and $h$ of degree $\leq e$ such that $f g = h$

$$\forall x \in \mathbb{F}_2^n \quad f(x)g(x) + h(x) = 0$$

Finding $g$ and $h$ $\iff$ solving the $2^n \times (E + D)$ system

$$\left( \text{Diag}(f(x)_{x \in \mathbb{F}_2^n}) \mathbf{V}^d_{\mathbb{F}_2^n} \mid \mathbf{V}^e_{\mathbb{F}_2^n} \right) \left( \begin{array}{c} \overline{g} \\ \overline{h} \end{array} \right) = 0$$
Fast Algebraic attacks - improved system

\[ V^e_{\{x, |x| \leq e\}} \text{ is involutive} \]

So, given the equation \( h(x) = f(x)g(x) \) we have

\[ V^e_{\{x, |x| \leq e\}} \bar{h} = \text{Diag}(f(x) |x| \leq e) V^d_{\{x, |x| \leq e\}} \bar{g} \]

\[ \bar{h} = M \bar{g} \overset{\text{def}}{=} V^e_{\{x, |x| \leq e\}} \text{Diag}(f(x) |x| \leq e) V^d_{\{x, |x| \leq e\}} \bar{g} \]

and we get a new \( 2^n \times D \) system (actually \( (2^n - E') \times D \))

\[ \left( \text{Diag}(f(x)_{x \in F_2^n}) V^d_{F_2^n} + V^e_{F_2^n} M \right) \bar{g} = 0 \]
Fast matrix vector product for $V$

Over $\mathbb{F}_2^4$, $V_{\mathbb{F}_2^4}^{\mathbb{F}_2^4} =$

Moebius Transform: Computable in $O(2^n \log 2^n)$ over $\mathbb{F}_2^n$
The issue in term of linear algebra

For both attacks, we get a matrix $A$

- Is the matrix $A$ singular?
  - to assert the immunity

- Find elements in the kernel of $A$
  - to find relations and build an attack
Existing algorithms

Basic algorithm

▶ Gaussian elimination on $A$

[Armknecht Carlet Gaborit Künzli Meier Ruatta] EuroC 06
[Didier Tillich] FSE 06
[Braken Lano Preneel] ACISP 06

▶ Use structure to improve elimination

[Didier] Indocrypt 06

▶ Idea based on fast matrix vector product with $A$
Part 3
Wiedemann’s Algorithm
General Facts for an $N \times N$ matrix

- One of the algorithms designed to solve large sparse linear system
- Huge literature because of important applications (used in factorisation/discrete logarithm algorithms)
- Complexity in $O(N)$ matrix vector products
- Faster than Gaussian elimination as long as matrix vector product is faster than $O(N^2)$
Wiedemann’s Algorithm for a square matrix

Let \( A \) an \( N \times N \) matrix and \( b \) a vector in \( \mathbb{F}_2^N \)

The following Krylov sequence is linearly generated

\[
b, Ab, A^2b, \ldots, A^Nb, \ldots
\]

Let \( P_b \in \mathbb{F}_2[X] \) be its minimal polynomial, that is

the minimal degree polynomial such that \( P_b(A)b = 0 \)

\( P_b \) divide the minimal polynomial of \( A \), so \( \deg(P_b) \leq N \)
From $P_b$ to our problem solution

Assume $P_b$ is known and that

$$P_b(X) = c_0 + XQ(X) \quad Q(X) \in \mathbb{F}_2[X]$$

If $c_0 \neq 0$ (and therefore $c_0 = 1$) then $AQ(A)b = b$

► $Q(A)b$ is a solution $x$ to the system $Ax = b$

If $c_0 = 0$ then $AQ(A)b = 0$ (in particular $A$ is singular)

► $Q(A)b$ is a non-trivial kernel element of $A$
Computing \( P_b \) with Berlekamp-Massey

Choose a random vector \( u^t \) in \( \mathbb{F}_2^N \) then compute

\[
u.b, u.Ab, u.A^2b, \ldots, u.A^{2N}b\]

and its minimal polynomial \( P_{u,b} \) with Berlekamp-Massey

\[P_b \quad P_{u,b}/P_b \quad \text{and equality with probability} \quad \geq 1/(6 \log N)\]

Complexity

- \( 2N \) matrix vector product of \( A \)
- \( O(N^2) \) for the Berlekamp-Massey part
Version we used to assert the immunity

Over $\mathbb{F}_2$ and for a random choice of $b$ and $u$, if $A$ is singular then $X/P_{u,b}$ with probability $\geq 1/4$

$A$ singular?
- Try $i$ different values of $u$ and $b$
- If $X/P_{u,b}$ then yes
- If $\deg(P_{u,b}) = N$ and $X \nmid P_{u,b}$ then no
- Otherwise, no with $pb \geq 1 - (3/4)^i$
If $A$ is a $N \times k$ matrix, an algorithm exist to construct a “random” sparse $k \times N$ matrix $Q$ such that

- If $A$ is of full rank, then $QA$ is a $k \times k$ non-singular matrix with a probablity bounded away from 0
- The number of 1 in $Q$ is in $O(N \log N)$

Now:

- We can just run Wiedemann’s algorithm on $QA$
- However, more pass are needed
Part 4

Results
Degree $d$ immunity of a $n$-variable Boolean function

Square case:
- $O(D)$ matrix vector products in $O(n2^n)$
- $O(2^n)$ memory
- Moebius transform is vectorizable (SSE2)

Non square case:
- $O(D)$ evaluations in $O(n2^n)$
- $O(n2^n)$ memory for storing the matrix $Q$
- product by $Q$ is not vectorizable
## Complexity summary for normal AA

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>complexity</th>
<th>memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>$O(</td>
<td>f</td>
</tr>
<tr>
<td>Eurocrypt 2006(^{(1)})</td>
<td>$O(D^2)$</td>
<td>$O(D^2)$</td>
</tr>
<tr>
<td>FSE 2006(^{(2)})</td>
<td>$O(D)$, fixed $d$, $n \to \infty$</td>
<td>$O(D)$</td>
</tr>
<tr>
<td>Wiedemann’s</td>
<td>$O(n2^n D)$</td>
<td>$O(n2^n)$</td>
</tr>
</tbody>
</table>

\(^{(1)}\): Average complexity, worst should be around $O(2^n D)$

\(^{(2)}\): Average complexity and memory to assert the immunity only, in the general case no better results than the Gaussian elimination, but a lot faster in practice
Complexity summary for Fast AA

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>complexity</th>
<th>memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>$O(2^n(E + D)^2)$</td>
<td>$O((E + D)^2)$</td>
</tr>
<tr>
<td>Eurocrypt 2006$^{(1)}$</td>
<td>$O(ED^2)$</td>
<td>$O(D^2)$</td>
</tr>
<tr>
<td>ACISP 2006$^{(1)}$</td>
<td>$O(ED^2 + E^2)$</td>
<td>$O(ED)$</td>
</tr>
<tr>
<td>FSE 2006$^{(2)}$</td>
<td>$-$</td>
<td>$-$</td>
</tr>
<tr>
<td>Wiedemann’s$^{(3)}$</td>
<td>$O(n2^nD)$</td>
<td>$O(n2^n)$</td>
</tr>
</tbody>
</table>

$^{(1)}$: Average complexity, worst should be around $O(2^nDE)$

$^{(2)}$: Adaptable to this case, should give the same kind of result as for normal AA, but no theoretical proof

$^{(3)}$: Best algorithm for most values of the degree constraint
Degree $d$ immunity for a $n$-variable balanced Boolean function ($n = 2d + 1$)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$d, n$</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian elimination</td>
<td>$d = 8 \ n = 17$</td>
<td>a few hours</td>
</tr>
<tr>
<td>Eurocrypt 2006</td>
<td>$d = 9 \ n = 19$</td>
<td>-</td>
</tr>
<tr>
<td>FSE 2006</td>
<td>$d = 9 \ n = 19$</td>
<td>6h</td>
</tr>
<tr>
<td>Wiedemann’s algorithm</td>
<td>$d = 9 \ n = 19$</td>
<td>102s</td>
</tr>
<tr>
<td>(one pass)</td>
<td>$d = 11 \ n = 23$</td>
<td>11h</td>
</tr>
<tr>
<td></td>
<td>$d = 12 \ n = 25$</td>
<td>20d</td>
</tr>
</tbody>
</table>
Wiedemann’s algo for non-square case

We loose more than a factor $32$ (no vectorization)

- for AA, same limit as with previous algo ($n = 19$)
- Same limit for FAA! almost no dependance on $e$

Improvement: by using block Wiedemann’s algorithm we can expect the same kind of performance
Advantages of this approach

- Uses well-known algorithms and has good complexity
- Memory efficient → can deal with many variables
- Almost the same algorithm for AA and FAA
- Leads to an efficient decoding over the erasure channel for all codes that can be generated efficiently