Advanced Cryptology
Final Exam
Fast correlation attacks on filtered LFSRs

2020-01-22

Instructions. ***This subject is for students who followed the*** **Advanced Cryptology** ***part of this course only.*** *The duration of this test is* two hours*. All documents are authorised. Be careful to justify all your answers, possibly including any assumption that you deem to be necessary. Algorithms must be described at a "reasonably high" level; the use of formal pseudocode is not mandatory.*

The goal of this subject is to study some aspects of *fast correlation attacks* on filtered linear-feedback shift-register (LFSR) stream ciphers.

Filtered LFSRs & the noisy channel model

We define a (binary) maximum-period-LFSR in Galois configuration with an n-bit state in the following way: let P be a primitive (irreducible) polynomial of degree n over $\mathbb{F}_2[X]$ and $\mathbb{K} := \mathbb{F}_2[X]/\langle P\rangle$ be the corresponding field extension. The *state* of the LFSR is a non-zero element $S \in \mathbb{K}^\times$ (i.e. a non-zero polynomial of $\mathbb{F}_2[X]$ of degree at most $n-1$); one then defines a sequence $s(t)$ associated with S as $s(t) := S \times X^t$ (where $X \in \mathbb{K}$; alternatively as polynomials over $\mathbb{F}_2$, the sequence is $s(t) = S \times X^t \mod P$).

Let us denote by $s(t)[i]$, $0 \leq i < n$, the i^{th} coefficient of $s(t)$ (seen as a polynomial); a *filtered* LFSR is defined by the output sequence $v(t) := f(s(t)[I_1], \ldots, s(t)[I_l])$, where $f : \mathbb{F}_2^l \to \mathbb{F}_2$ is a (non-linear) Boolean function in l variables and $\mathcal{I} := \{I_1, \ldots, I_l\} \subseteq [\![0, n-1]\!]$ is a set of l indices. The *noisy channel model* consists in modelling this sequence v as $v(t) \approx \sum_{i\in\mathcal{I}} s(t)[i] + e_t$, where $e_t \sim \text{Ber}_\eta$ is a *noise* term that is zero with probability $\eta < 0.5$; we write c the associated correlation $|2\eta - 1|$. To simplify the notation and without loss of generality, we will actually model $v(t)$ as $v'(t) := s(t)[n-1] + e_t$.

In this exercise, we will consider an attacker who has access to v and tries to determine the corresponding initial state S. The entire analysis will be done in the noisy channel model, using v'.

A first approach would be for an adversary to recover $S[n-1] = s(0)[n-1] = v'(0) + e_0$, $S[n-2] = s(1)[n-1] = v'(1) + e_1$ etc. by using majority-logic decoding on many samples $v'(0)$s with different e_0s, many samples $v'(1)$s with different e_1s, etc. Unfortunately, v' only provides a single sample for every t. The idea is then to combine well-chosen terms of v to emulate the above approach by "creating" many samples for a fixed state coefficient $S[i]$. This can be done using *parity-check* equations for the LFSR, defined thusly: a parity-check equation of *weight* k for an LFSR defined by P is a subset $\mathcal{T} = \{t_1, \ldots, t_k\} \subset \mathbb{N}$ s.t. $\sum_{t\in\mathcal{T}} s(t) = 0$ for all possible initial values $S = s(0)$. Note that if $M := X^{t_1} + \cdots + X^{t_k}$ is a multiple of P, then $S \times M \mod P = 0$ and $\{t_1, \ldots, t_k\}$ is a parity-check equation.

Let us now assume that an adversary knows sufficiently many parity-check equations that all share a common index t_1; this means that the adversary knows many relations of the form $s(t_1) = s(t_2) + \cdots + s(t_k)$, $s(t_1) = s(t'_2) + \cdots + s(t'_k)$, etc.. Since those relations hold for the entire polynomials, they also hold in particular for their individual coefficients, i.e. $s(t_1)[n-1] = s(t_2)[n-1] + \cdots + s(t_k)[n-1]$ etc.. Now since $s(t_2)[n-1] + \cdots + s(t_k)[n-1] = v'(t_2) + \cdots + v'(t_k) + e_{t_2} + \cdots + e_{t_k}$ etc., one may try to determine the value $s(t_1)[n-1]$ by using majority-logic decoding on the many samples $v'(t_2) + \cdots + v'(t_k)$, $v'(t'_2) + \cdots + v'(t'_k)$, etc. which all involve independent error terms. This overall approach is known as a *fast correlation attack*.

Q.1 (majority-logic decoding): Let d_0, d_1 be two random sequences over $\{0,1\}$ where $d_0(t) \sim \mathrm{Ber}_{0.5}$ (i.e. is uniform) and $d_1(t) \sim \mathrm{Ber}_{\eta \neq 0.5}$.

1. Give a distinguisher for the two sequences, i.e. define an algorithm that has oracle access to $(d_b, d_{b\oplus 1})$, $b \xleftarrow{\$} \{0,1\}$ and that returns a guess for the value b (i.e., the adversary is given access to *both* d_0 and d_1, but doesn't know which is which).
2. How many samples of d_b and $d_{b\oplus 1}$ (up to a constant) does your distinguisher need in function of η to succeed with an advantage close to 1?

Q.2 (—— of a random code): Let $\boldsymbol{A} \xleftarrow{\$} \mathbb{F}_2^{k\times n}$; we make the simplifying hypothesis that $n \gg k$ so that we may assume that the rank of $\boldsymbol{A}$ is k. Further let $\boldsymbol{x} \in \mathbb{F}_2^k$, $\boldsymbol{e} \sim \mathrm{Ber}^n_{\eta\neq 0.5}$ (a vectorial Bernoulli distribution of parameter η), $\boldsymbol{c} := \boldsymbol{x}\boldsymbol{A}$, $\hat{\boldsymbol{c}} := \boldsymbol{c} + \boldsymbol{e}$.

1. Explain how one can recover $\boldsymbol{x}$ from $\hat{\boldsymbol{c}}$ using majority-logic decoding, and state the necessary assumptions for this decoding to be successful with probability close to 1.
2. What is the time complexity of this decoder? Is it (in general) possible to do better?

Q.3 (—— using parity-check equations): We now consider again the setting of a fast-correlation attack as described in the introduction.

1. Explain how to use parity-check equations with a common term t_1 which is *arbitrary* (i.e. not necessarily 0) to recover S *in its entirety* (i.e. $S[0], \ldots S[n-1]$).
2. How many distinct parity-check equations of weight k are necessary for this recovery to be successful with a probability close to 1?

We now turn to the problem of finding the parity-check equations necessary for a fast-correlation attack. For this several approaches are possible, and we focus on one based on a *generalised birthday problem*: given a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^{n'}$, an integer $k > 2$, and a constant $c \in \mathbb{F}_2^{n'}$, find $x_1, \ldots, x_k \in \mathbb{F}_2^n$ s.t. $\sum_{i=1}^k F(x_i) = c$. In all of the following, we take $k = 4$.

Q.4 (generalised join operator): Let L_1, L_2 be two lists of N elements of $\mathbb{F}_2^n$, and $\mathrm{trunc}_l : \mathbb{F}_2^n \to \mathbb{F}_2^l$ be the truncation map to the l lowest coordinates of its argument. We define a *generalised join operator* $\bowtie_l$ of parameter $l \leq n$ as $L_1 \bowtie_l L_2 = \{(x_1, x_2) \ : \ x_1 \in L_1,\, x_2 \in L_2,\, \mathrm{trunc}_l(x_1) = \mathrm{trunc}_l(x_2)\}$.

1. Give an efficient algorithm (with time complexity $\mathcal{O}(N \log N)$, under the assumption that the size of the output $L_1 \bowtie_l L_2 \leq N$) for $\bowtie_l$, and analyse its time and memory complexity.
2. Explain how to use $\bowtie_l$ to form the list L_{12} of pairs $(x_i, x_j) \in L_1 \times L_2$ s.t. $\mathrm{trunc}_l(x_i + x_j) = 0$.
3. Assuming that L_1 and L_2 contain uniformly random elements of $\mathbb{F}_2^n$, explain why the expected size of $L_{12} \approx N^2/2^l$.

Q.5 (generalised birthday): Let $L_1, L_2, L_3, L_4, L_{12}, L_{34}$ be lists defined as in the previous question, with parameter l.

1. Show that if $(x_1, x_2) \in L_{12}$, $(x_3, x_4) \in L_{34}$, then $\Pr[x_1 + x_2 + x_3 + x_4 = 0] = 2^{l-n}$, where the probability is taken over the sampling of $L_1, \ldots$.
2. Let $l = n/3$. Describe an algorithm that takes as input L_1, L_2, L_3, L_4 of size 2^l and that with "high" probability returns one quadruple $(x_1, x_2, x_3, x_4) \in L_1 \times L_2 \times L_3 \times L_4$ s.t. $x_1 + x_2 + x_3 + x_4 = 0$, and analyse its time and memory complexity.
3. Explain how to adapt the algorithm to find a quadruple of elements of $L_1, \ldots$ that sums to any constant c (or equivalently, explain how to find a quintuple $(x_1, x_2, x_3, x_4, x_5)$ that sums to 0, where $(x_1, x_2, x_3, x_4) \in L_1 \times L_2 \times L_3 \times L_4$ and x_5 is fixed *a priori*).
4. Explain how to adapt the algorithm and the initial size of $L_1, \ldots$ so that it returns an expected number of $2^q > 1$ quadruples of $L_1, \ldots$ that sum to 0?

Q.6 (application to parity-check equations): We now make the assumption that the values X^t mod P are pseudo-random, i.e. that they can be considered to be independent and uniformly random over $\mathbb{F}_2^n$.

1. Explain how to use the above algorithm with 4 lists to generate many parity check equations of weight 5 with a common term X^{t_1}.

References

[Wag02] David A. Wagner. A Generalized Birthday Problem. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 288–303. Springer, 2002.

This page intentionally left blank