Advanced Cryptography: Algorithmic Cryptanalysis

DANIEL LOEBENBERGER, KONSTANTIN ZIEGLER

4. Exercise sheet
Hand in solutions until Saturday, 07 May 2011, 23:59h.

Exercise 4.1 (Primitivity). (4 points)

In the lecture we have seen that the period of the output sequence of an LFSR over $\mathbb{F}_q$ with minimal polynomial g is maximal if and only if the polynomial g is primitive. One can show the following 4

Theorem. *An irreducible polynomial g of degree k over $\mathbb{F}_q$ is primitive if the smallest exponent n for which g divides $x^n - 1$ is $n = q^k - 1$.*

Specify an algorithmic method using the above fact that decides efficiently whether a given polynomial is primitive over $\mathbb{F}_q$ given the prime factorization of $q^k - 1$. Note: $x^e - 1$ divides $x^n - 1$ if and only if e divides n.

Exercise 4.2 (Linear recurring sequences). (10 points)

Consider the following two linear recurrent sequences over $\mathbb{F}_2$ defined for integers $n \geq 0$ by

$$s_{n+15} = s_{n+14} + s_n$$

and

$$t_{n+17} = t_{n+14} + t_{n+2} + t_{n+1} + t_n.$$

(i) Draw the two corresponding linear feedback shift registers (LFSRs) that implement the sequences $(s_n)_{n\geq 0}$ and $(t_n)_{n\geq 0}$. 2

(ii) Now initialize the LFSR corresponding to $(s_n)_{n\geq 0}$ with $(0, \ldots, 0, 1)$ and the other one with $(0, \ldots, 0, 1, 1, 1)$. Compute the next 15 sequence elements. What do you observe? 3

(iii) Compute for both registers the characteristic polynomials $g_s(x)$ and $g_t(x)$. 2

(iv) Show that g_s divides g_t. 1

(v) Show that g_s is primitive over $\mathbb{F}_2$. 2

Exercise 4.3 (The correlation attack running). (14+5 points)

We are going to see the correlation attack running on the following generator: It consists of three LFSRs of size 15 bits, 16 bits and 19 bits (respectively) over $\mathbb{F}_2$ with minimal polynomials

$$\begin{aligned} g_1(x) &= x^{15} + x^{14} + 1 \\ g_2(x) &= x^{16} + x^{15} + x^4 + x^2 + 1 \\ g_3(x) &= x^{19} + x^{18} + x^5 + x + 1 \end{aligned}$$

and nonlinear combiner

$$g(x_1, x_2, x_3) = x_1 x_2 \oplus x_1 x_3 \oplus x_2 x_3.$$

$\boxed{2}$ (i) Determine the period of the sequence generated by the key-stream generator. Give a proper argument to justify your claim.

$\boxed{1}$ (ii) Compute the correlation probabilities for all three LFSRs and the output key-stream.

$\boxed{4}$ (iii) Write a routine that implements the three LFSRs. To check the correctness of your implementation compare the first 100 output bits of the LFSRs for the seeds $[0, \ldots, 0, 1]$ with the correct sequences given below:

Output LFSR 1:

0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 0 0 1 0 1 1 0 1 0 0 1 0
1 1 0 0 0 1 1 0 1 1 0 0 0 1

Output LFSR 2:

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 0 1 0 1 0 1 1 0 0 1
0 1 0 1 1 0 0 1 0 0 1 1 0 0 0 1 0 0 1 1 0 1 0 0 1 0 1 1 1 0 1 1 0 0 1 1 0 1 1 0 1 0 1
1 1 0 0 0 1 1 1 1 1 0 1 0 1

Output LFSR 3:

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 1 0 1 0 1 0 1
0 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 0 1 1 1 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 1 1 0 1 0 0 1 0
1 1 1 1 0 1 1 0 1 1 0 0 0 1

$\boxed{2}$ (iv) Implement the full generator, i.e. connect the outputs of the LFSRs using the nonlinear combiner g. To check your implementation, here are the first 100 bits of the output of the key stream generator (3 LFSRs with nonlinear combiner g, each of them with seed $[0, \ldots, 0, 1]$):

Output LFSRs with nonlinear combiner f:

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 0 1 0 1 0 1 0 1
0 1 0 1 1 0 0 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 0 1 1 0 1 1 1 0 1 0 0 1 0 1 1 0 1 0 0 1 0
1 1 0 0 0 1 1 0 1 1 0 0 0 1

(v) On the webpage you will find the first 1000 bits of the output of the LFSRs with nonlinear combiner where the seeds are unknown. Perform the correlation attack to find the seeds. Note that the program will run several minutes to perform the attack. In order to filter the seed candidates use the threshold $\theta = \theta_1 = \theta_2 = \theta_3 = 0.57$, i.e. add a seed candidate to a list of correct candidates if the guessed keystream agrees in a fraction of at least θ bits. 5

(vi) What happens if you decrease/increase θ? +5

Hand in your written solutions, the commented source code and the correct seeds.