Cryptography, winter 2006

PROF. DR. WERNER SCHINDLER, DIPL.-INF. DANIEL LOEBENBERGER

7. Exercise sheet (10.01.2007)
Hand in solutions to the homework exercises on Wednesday, January 24th, in the tutorial/the lecture.

Exercise 7.1 (Block ciphers in OFB mode).

Assume that the block cipher Enc is operated in OFB mode and generates random numbers $r_1, r_2, \ldots$. Prove: If an adversary knows the random numbers $r_i, \ldots, r_{i+j}$, finding r_{i+j+1} or r_{i-1} is at least as difficult as a chosen-plaintext, resp. a chosen-ciphertext attack, on the block cipher Enc.

Exercise 7.2 (Homework: Siegenthaler's Attack). (20 points)

Here is our first programming task: Implement Siegenthaler's Attack in a programming language of your choice. For this task it is allowed to work in groups of at most three students. Please give some information on the authors in the sources.

- Implement three LFSRs of size 15 bits, 16 bits and 19 bits (respectively) with connection polynomials $\boxed{7}$

$$\begin{aligned} f_1(x) &= x^{15} + x + 1 \\ f_2(x) &= x^{16} + x^{14} + x^{12} + x + 1 \\ f_3(x) &= x^{19} + x^{18} + x^{14} + x + 1 \end{aligned}$$

  To check the correctness of your implementation compare the first 100 output bits of the LFSRs for the seeds $[1, 0, \ldots, 0]$ 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

$\boxed{3}$ ◦ Implement the nonlinear combiner

$$f(x_1, x_2, x_3) = x_1x_2 \oplus x_1x_3 \oplus x_2x_3$$

and connect the LFSRs using f. To check your implementation, here are the first 100 bits of the output of the key stream generator (3 LFSRs with a nonlinear combiner, each of them with seed $[1, 0, \dots, 0]$):

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

$\boxed{10}$ ◦ 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 Siegenthaler's 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 $th = th_1 = th_2 = th_3 = 0.57$. What happens if you decrease/increase th?

Hand in the commented source code and seeds.

Solution. The desired seeds are

$$\begin{aligned} s_1 &= 110001000010100 \\ s_2 &= 0100000001100100 \\ s_3 &= 10000001101100000000 \end{aligned}$$

Those who are interested may ask me for my code. It was implemented in ANSI C. ○