جامعة المستقبل
AL MUSTAQBAL UNIVERSITY

كلية العلوم

قسم علوم الامن السيبراني

Cyber Security Department

Subject: Nonlinear Shift Register Nonlinear combination generators
Class: 2nd
Lecturer: Asst.Lect Mustafa Ameer Awadh

Lecture: (6)

Study Year: 2024-2025

1.1 Nonlinear Shfit Register:

LFSR are not always useful ,since they are not resistence to a given know- plaintext attack.as an alternative ,nonlinear feedback shfit register with nonlinear feed back funtion are often used.

Linear feedback shfit register are unsafe because they have relative small linear complxity ,and hence a relatively small fragment of the key stream(LFSR sequnce) can be used to obtain the entire sequnce by solving as set of linear equation .to increase the linear complxity of LFSR ,one or more output sequnce of LFSR's are combined with some nonlinear function to produce relative higher linear complxity ,for example shfit register SR1 genertates sequnce(S1) with sequnce length of (2^n-1),and shfit register SR2 generates sequnce (S2) with sequnce length (2^m-1),then output sequnce.

(S3) will be:

*S3=S1*S2 with period (sequnce length)= (2^n-1)* (2^m-1)*

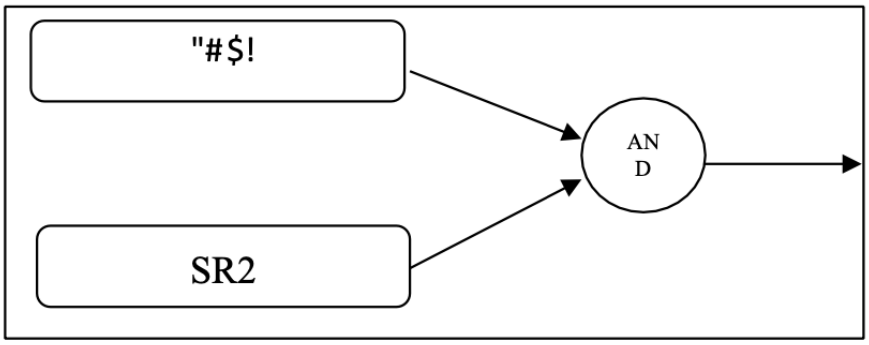

Fig.1 AND operation in NLFSR.

In which the key stream generator is a shift register with non-linear feedback function .as illustrated in following figure.
In this type one LFSR is used with n-stages and non-linear feedback function. The simplest nonlinear function is **"AND"** functions, for example:

$$F = 1 + X_1X_2 + X_2X_3 + X_2X_3X_4$$

Where X_1X_2 are (X_1 **AND** X_2)

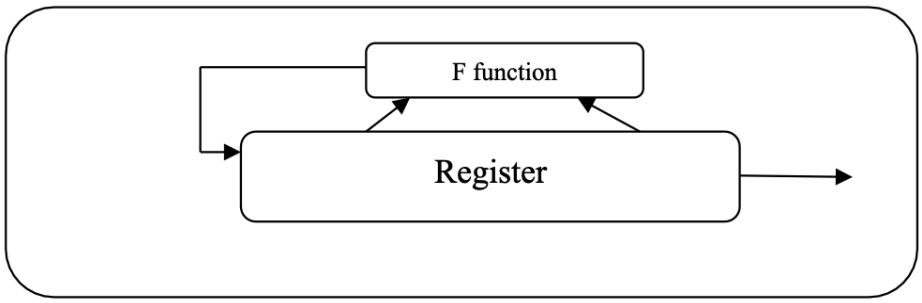

Fig.2 Non-Linearity Filtered LFSR System.

In which a nonlinear logical function is applied to the contented of the LFSR. **Gorth** generator is an example of this type, as illustrated in following figure **Gorth** sequence consists of:

- Linear feedback function given by:

$$F1 = S1 + S2 + S3$$

- Nonlinear filter given by:

$$F2 = S_0S_3 + S_1S_5 + S_2S_4$$

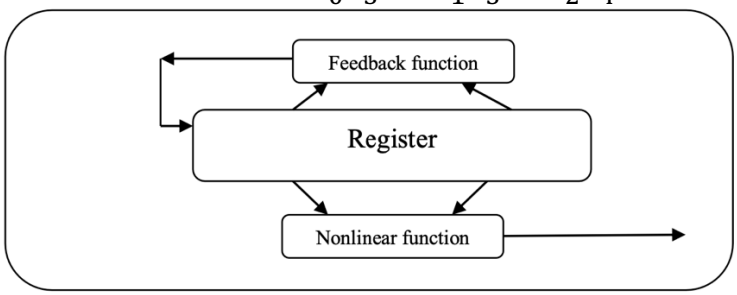

Fig3 Nonlinear Feedback Shift Register (NFSR).

This diagram represents a Nonlinear Feedback Shift Register (NFSR).

In this structure:

- A Register holds the current state.
- A Feedback Function processes the state and feeds it back to the register.
- A Nonlinear Function takes the output from the register to generate the final output of the sequence.

2.1Nonlinear combination generators

One general technique for destroying the linearity inherent in LFSRs is to use several LFSRs in parallel. The key stream is generated as a nonlinear function **f** of the outputs of the component LFSRs; this construction is illustrated in the following Figure. Such key stream generators are called ***nonlinear combination*** *generators*, and **f** is called the *combining function*. The remainder of this subsection demonstrates that the function **f** must satisfy several criteria to withstand certain particular cryptographic attacks.

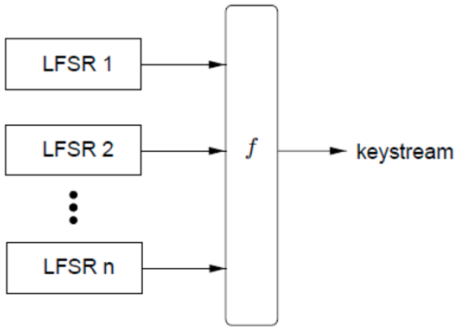

Fig.4 nonlinear combination.

In Boolean functions, which are functions involving variables like X1,X2,X3 and so on, we can express any function as a "sum" of products of these variables. For example, if you multiply three different variables (like X1×X2×X3), this is called a "3rd order product" because it involves three distinct variables.

The **Algebraic Normal Form (ANF)** of a Boolean function is a way of writing the function as a sum (using XOR) of various products like these.

For the Boolean function:

$$f(X1,X2,X3,X4,X5)=X1 \text{ XOR } X2 \text{ XOR } X3 \text{ XOR } (X4 \times X5) \text{ XOR } (X1 \times X3 \times X4 \times X5)$$

This function consists of several products, some of which are of degree 1 (like X1), some are of degree 2 (like X4×X5), and one is of degree 4 (like X1×X3×X4×X5). Since the highest degree of products here is 4, we say that the **nonlinear order** of the function is 4. A higher nonlinear order means the function is harder to predict or analyze using simple linear methods. In generating random sequences, especially for cryptographic purposes, using a Boolean function with a high nonlinear order makes the output sequence more complex and harder to analyze. This results in a higher linear complexity of the generated sequence, making it more secure because it's more difficult to break.

2.2 Geffe generator

The Geffe generator, as depicted in following Figure. is defined by three maximum-length LFSRs whose lengths ***L1, L2, L3*** are pairwise relatively prime, with nonlinear combining function

$$F(X_1, X_2, X_3) = (X_1 \text{ and } X_2) \text{ Xor } (\text{not } (X_2) \text{ and } X_3)$$

And the maximal length of Geffe is

$$\textbf{Max period} = (2^{L1}-1) * (2^{L2}-1) * (2^{L3}-1)$$

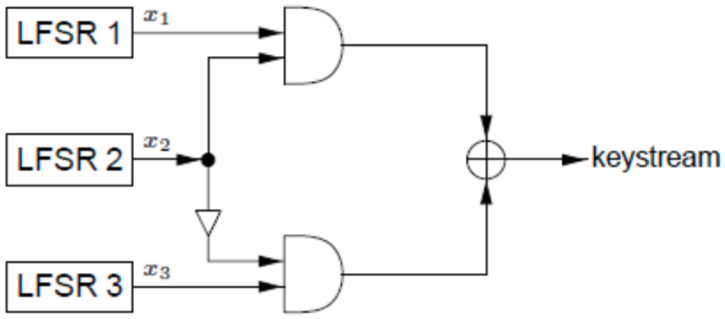

Fig.5 Geffe generator.

Example with Numbers:
Let's assume we have three LFSRs with the following lengths:

- *L1=3*
- *L2=5*
- *L3=7*

Since these lengths are pairwise relatively prime, they meet the conditions for the Geffe generator. Let's go through the calculations step-by-step.

1. Maximal Period Calculation

The maximal period is calculated using:

$$\textbf{Max period} = (2^{L1}\text{-}1) * (2^{L2}\text{-}1) * (2^{L3}\text{-}1)$$

Substitute L1=3, L2=5, and L3=7:

$$\begin{aligned}\text{Max period} &= (2^3 - 1) \times (2^5 - 1) \times (2^7 - 1) \\ &= (8 - 1) \times (32 - 1) \times (128 - 1) \\ &= 7 \times 31 \times 127 = 27517\end{aligned}$$

So, the maximal period of this Geffe generator would be **27517**.

2. Example Calculation of *F(X1,X2,X3)*

Let's take some example outputs for X1, X2, and X3 from the LFSRs. Assume we have:

- *X1 = 1*
- *X2 = 0*
- *X3 = 1*

Using the combining function:

$$F(X1,X2,X3) = (X1 \wedge X2) \oplus (\neg X2 \wedge X3)$$

Step-by-Step Calculation:

1. Calculate X1 ∧ X2:

$$X1 \wedge X2 = 1 \wedge 0 = 0$$

2. Calculate ¬X2:

$$\neg X2 = \neg 0 = 1$$

3. Calculate ¬X2∧X3:

$$\neg X2 \wedge X3 = 1 \wedge 1 = 1$$

Now, XOR the results:

$$F(X1,X2,X3)=0 \oplus 1=1$$

So, for this set of inputs (X1=1, X2=0, X3=1), the output of the Geffe generator, *F(X1,X2,X3)* is **1**.

Another Example with Different Inputs

Let's try another set of values:

- X1 = 0
- X2 = 1
- X3 = 1

Using the same combining function:

1. Calculate X1 ∧ X2:

$$X1 \wedge X2 = 0 \wedge 1 = 0$$

2. Calculate ¬X2 :

$$\neg X2 = \neg 1 = 0$$

3. Calculate ¬X2 ∧ X3 :

$$\neg X2 \wedge X3 = 0 \wedge 1 = 0$$

4. XOR the results:

$$F(X1,X2,X3) = 0 \oplus 0 = 0$$

For this set of inputs (X1 = 0 , X2 = 1), the output is **0**.

2.3 Hardmard Generator

Nonlinear generator consists of two LFSR of(**X1,X2**) with nonlinear function:

$$F(x) = (S1 \text{ and } S2)$$

And the maximal:

$$\textbf{Max period} = (2^{L1} - 1) * (2^{L2} - 1)$$

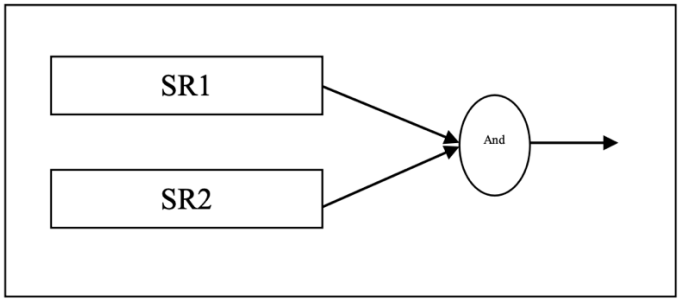

Fig.6 Hardmard Generator.

2.4 Threshold Generator

nonlinear generators consist of three LFSR with nonlinear function:

$$F(x) = (S1 \text{ and } S2) \text{ Xor } (S1 \text{ and } S3) \text{ Xor } (S2 \text{ and } S3)$$

And the maximal:

$$\textbf{Max period} = (2^{L1}-1) * (2^{L2}-1) * (2^{L3}-1)$$

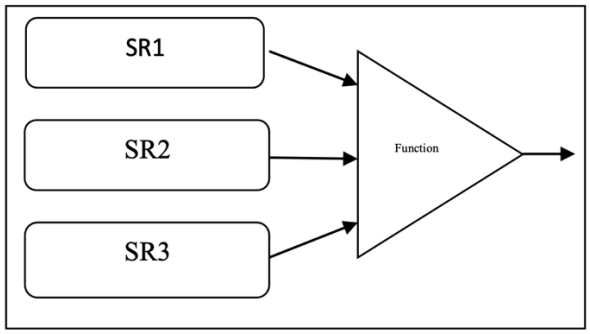

Fig.7 Threshold Generator.

3.1 LFSR/FCSR Summation/Parity Cascade

The **LFSR/FCSR Summation/Parity Cascade** is a cryptographic structure that combines Linear Feedback Shift Registers (LFSRs) and Feedback with Carry Shift Registers (FCSRs) in a cascade structure. This design leverages both the properties of LFSRs and FCSRs to generate sequences with desirable cryptographic properties, such as long periods, high linear complexity, and resistance to certain attacks. Let's break down these components and the cascade structure:

3.2Components of the Cascade

3.2.1 LFSR (Linear Feedback Shift Register):

- An LFSR is a shift register where each bit is shifted to the next position, and a feedback function (typically XOR) based on certain positions (taps) determines the new bit in the register.
- LFSRs are popular in stream ciphers because they can produce long pseudo-random sequences with minimal hardware.
- However, pure LFSR-based sequences may be vulnerable to attacks due to their linear nature.

3.2.2 FCSR (Feedback with Carry Shift Register):

- An FCSR is a type of shift register similar to an LFSR, but it includes a "carry" mechanism, which allows for more complex feedback patterns.
- This carry mechanism gives FCSRs a nonlinear characteristic, making them harder to break with linear attacks.
- FCSRs can also generate sequences with long periods and high complexity.

3.3 Summation/Parity Cascade

In a **summation/parity cascade**, the outputs of multiple LFSRs and/or FCSRs are combined in a layered or "cascaded" structure to produce the final output sequence. This combination typically involves:

- **Summation**: Adding the outputs of the LFSRs and FCSRs (usually using XOR for binary sequences).
- **Parity Check**: Using the parity (even or odd number of 1's in a sequence) as an additional combining function.

The goal of combining multiple LFSRs and FCSRs is to create a sequence that inherits the good qualities of both components while masking the weaknesses of each.

How the Cascade Works

Stage 1: Each LFSR/FCSR produces its own output sequence.

Stage 2: The outputs of these registers are combined through summation (XOR) or parity checks.

Final Output: The result of this summation/parity operation becomes the final keystream or pseudo-random output.

This cascade structure increases the linear complexity and makes it harder for an attacker to predict the sequence, as the nonlinear behavior from FCSRs adds an additional layer of security.

Example

To demonstrate the concept with a simplified example:

- Suppose you have two LFSRs and one FCSR in a cascade:
    - **LFSR1** with output sequence x1.
    - **LFSR2** with output sequence x2.
    - **FCSR** with output sequence x3.

The output of the cascade could be computed as:

$$\text{Output} = (x1 \oplus x2) \oplus x3$$

Each register will shift and generate new values, and the cascade will combine these values to produce a keystream.

Key Benefits of the Cascade

- **Higher Complexity**: Combining LFSRs and FCSRs increases the linear complexity of the sequence, making it harder to break with linear attacks.
- **Longer Period**: The period of the sequence is determined by the least common multiple of the periods of the individual registers, which can be very long.
- **Nonlinearity**: The FCSR introduces nonlinearity, adding to the security of the keystream.

3.4 Algorithm Alternating step generator

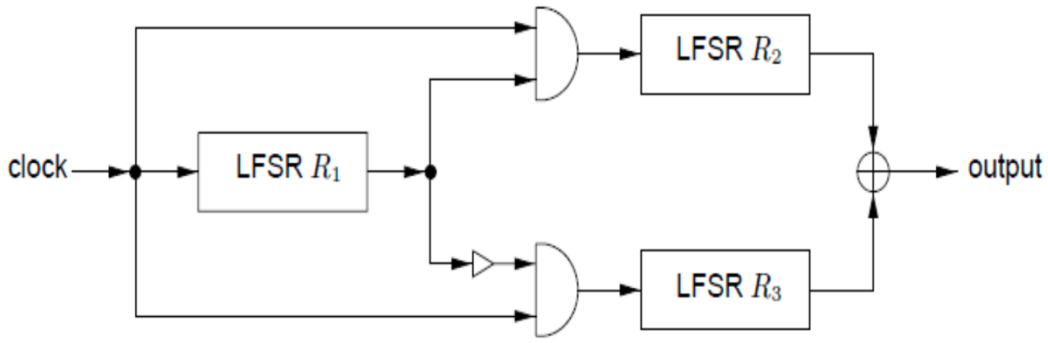

Fig.8 NLFSRs algorithm structure.

SUMMARY: a control LFSR R1 is used to selectively step two other LFSRs, R2 and R3.

OUTPUT: a sequence which is the bitwise XOR of the output sequences of R2 and R3.

The following steps are repeated until a keystream of desired length is produced.

1.Register R1 is clocked.

2. If the output of R1 is 1 then:

R2 is clocked; R3 is not clocked but its previous output bit is repeated.
(For the first clock cycle, the "previous output bit" of R3 is taken to be 0.)

3. If the output of R1 is 0 then:

R3 is clocked; R2 is not clocked but its previous output bit is repeated.
(For the first clock cycle, the "previous output bit" of R2 is taken to be 0.)
4. The output bits of R2 and R3 are XORed; the resulting bit is part of the key stream.