BÖLÜM 5

AKAN ŞİFRELER

Akan şifre sistemleri, mesajın her karakterini (bitini) ayrı ayrı şifreler. Şifreleme işlemi mesaj uzunluğunda bir anahtar kullanılarak yapılır. Anahtarın her biti mesajın her bitiyle mod 2'de karşılıklı toplanır. Bu işleme XOR işlemi denir ve $\oplus$ ile gösterilir.

Şifreleme :

$$\begin{aligned} \text{Açık metin } m &= m_1\, m_2 \ldots m_n \\ \text{Anahtar } k &= k_1\, k_2 \ldots k_n \\ \text{Kapalı metin } c &= c_1\, c_2 \ldots c_n \\ \text{Burada her } i \text{ için } c_i &= m_i \oplus k_i \text{ dir.} \end{aligned}$$

Şifreleme işleminde kullanılan iki tip anahtar dizisi vardır.

1. Tam Rastgele (true random) Dizi : Dizideki her bit birbirinden bağımsız olarak üretilir. Buna bir örnek olarak yazı tura atışı verilebilir.
2. Pseudo Rastgele (pseudo random) Dizi : Dizinin her biti kendinden önce gelen bitlere bağlıdır. Aynı zamanda her bit kendinden sonra gelen biti etkiler.

5.1 One Time Pad Sistemi

Şifrelenecek mesajın uzunluğunda tam rastgele bir anahtar dizisi seçilir. Mesaj ve anahtara XOR işlemi uygulanır.

Örnek 5.1.1

$$\begin{aligned} m &= 11010001011010110 \\ k &= 01110100101101000 \\ m \oplus k &= 10100101110111110 \end{aligned}$$

Mesajı açmak için aynı anahtara ve kapalı metine tekrar XOR işlemi uygulanır.

5.1.1 Sistemin Avantajları

Uzunluğu n bit olan bir mesaj için n bitlik bir anahtar dizisi seçilir. Mesaj şifrelenir ve gönderilir. Mesajı ele geçiren birisi olası bütün anahtarları (2^n tane) denese bile mesajı bulamaz. Çünkü bu işlemin sonunda n bitlik bütün kelimeleri bulur. Elinde birden fazla anlamlı mesaj olacağıiçin bu mesajların içinden gerçek mesajı tahmin etmek imkansızdır. Bu açıdan *koşulsuz güvenli* bir sistemdir.

5.1.2 Sistemin Dezavantajları

Uzun bir mesaj şifrelemek için uzun bir anahtar üretmek gerekir. Bu sistem tam rastgele bir anahtar dizisi kullandığından, uzun bir anahtar üretmek, bu anahtarı güvenli bir

şekilde karşı tarafa iletmek ve saklamak zor olur. Ayrıca kullanılan anahtar tekrar kullanılamayacağı için, her seferinde başka bir anahtar üretilmesi gerekir. Bu nedenlerden dolayı sistemin kullanımı zordur.

5.2 Dizi Üreticiler

Gerçekten rastgele dizilerin üretilmesi ve iletilmesi gibi zordur. Bu nedeniyle pseudo rastgele diziler kullanılr. Bu diziler kısa bir anahtar kullanılarak bir dizi üretici tarafından üretilir.

Dizi üreticiler açısından akan şifreleri ikiye ayırabiliriz.

1. Senkronize (Synchronous) Akan Şifre :

   f_s: Faz fonksiyonu, bir sonraki fazı belirleyen fonksiyon.

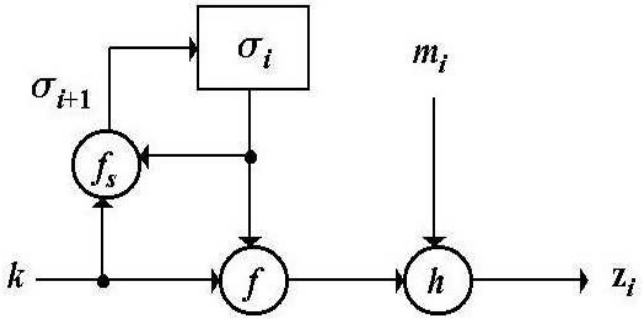
   f : Üreticinin fazına göre bit üreten fonksiyon.

   h : Şifreleme algoritması (XOR işlemi)

   σ_i: Üreticinin i zamandaki fazı

   σ_0: Üreticinin çalışmaya başlaması için belirlenen ilk faz. Gizlidir, genellikle anahtar ilk fazın ne olacağını belirler.

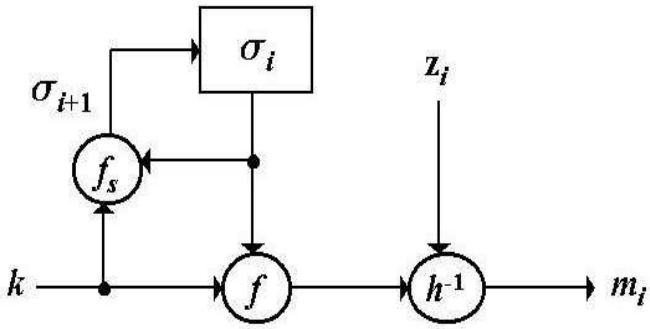
   Şifreleme :



Resim 1: Senkronize sistemlerde şifreleme

$$\begin{aligned} f(k,\sigma_i) &= z_i \\ f_s(k,\sigma_i) &= \sigma_{i+1} \\ h(m_i,z_i) &= c_i \end{aligned}$$

Deşifreleme :



Resim 2: Senkronize sistemlerde deşifreleme

$$\begin{aligned} f(k,\sigma_i) &= z_i \\ f_s(k,\sigma_i) &= \sigma_{i+1} \\ h^{-1}(c_i,z_i) &= m_i \end{aligned}$$

2. Oto-Senkronize (Self Synchronous) Akan Şifre :

   f_s: Faz fonksiyonu, bir sonraki fazı belirleyen fonksiyon.

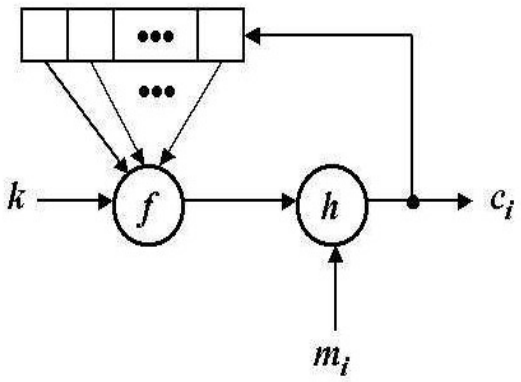
   f : Üreticinin fazına göre bit üreten fonksiyon.

   h : Şifreleme algoritması

   σ_i: $(c_{i-t}, c_{i-t+1}, \ldots, c_{i-1})$

   σ_0: $(c_{-t}, c_{-t+1}, \ldots, c_{-1})$

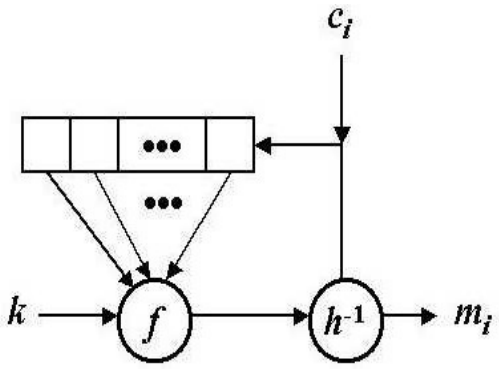
   Şifreleme :



Resim 3: Oto-Senkronize sistemlerde şifreleme

$$\begin{aligned} f(k,\sigma_i) &= z_i \\ h(m_i,z_i) &= c_i \end{aligned}$$

Deşifreleme :

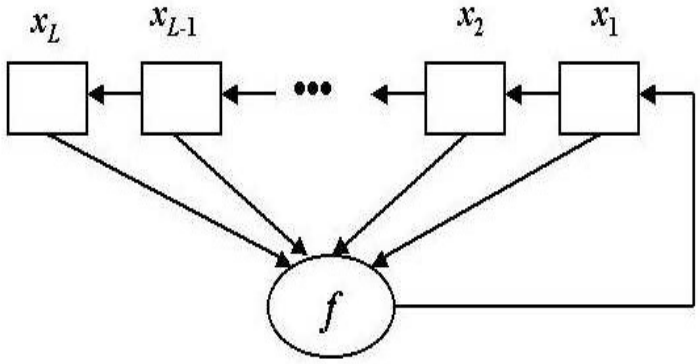


Resim 4: Oto-Senkronize sistemlerde deşifreleme

$$f(k, \sigma_i) = z_i$$
$$h^{-1}(c_i, z_i) = m_i$$

5.3 Geri Beslemeli Kaydırmalı Yazdırgaç (Feedback Shift Register)

Kısaca FSR olarak adlandırılan bir geri beslemeli kaydırmalı yazdırgaçın nasıl çalıŞtığı aŞağıdaki Şekilde gösterilmektedir.



Resim 5: Geri Beslemeli Kaydırmalı Yazdırgaç (FSR)

$f : \mathbb{Z}_2^n \to \mathbb{Z}_2$

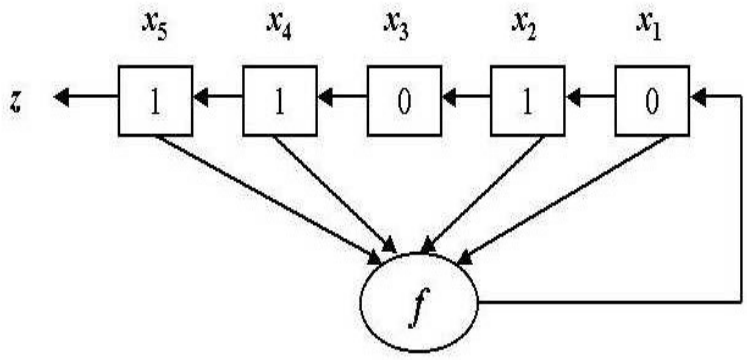
$\mathbb{Z}_2 = \{0, 1\}$ $\mathbb{Z}_2^n = \{(x_1, x_2, \ldots, x_n) \mid x_i \in \mathbb{Z}_2\}$

L: Yazdırgacın boyu

Bir sonraki fazda her bit sağa kayar. f fonksiyonu yeni bir bit üretir, üretilen bit en sağdaki göze yazılır.

$z = z_0 z_1 z_2 z_3 \ldots$ dizisinde her $i \in \mathbb{N}$ için $z_i = z_{i+p}$ eşitliğini sağlayan en küçük p sayısı dizinin periyodudur. Yani bu diziyi üreten FSR p adım sonra başlangıç fazına geri döner.

Örnek 5.3.1 $f(x_1, x_2, x_3, x_4, x_5) = x_1 \oplus x_2 x_3 \oplus x_4 x_5$



Resim 6: Fonksiyonu $f(x_1, x_2, x_3, x_4, x_5) = x_1 \oplus x_2x_3 \oplus x_4x_5$ olan FSR

$$\begin{aligned}
\sigma_0 &= 0\ 1\ 0\ 1\ 1 \qquad f(\sigma_0) = 1 \oplus 1.0 \oplus 1.0 = 1 \\
\sigma_1 &= 1\ 0\ 1\ 1\ 1 \qquad f(\sigma_1) = 1 \oplus 1.1 \oplus 0.1 = 0 \\
\sigma_2 &= 0\ 1\ 1\ 1\ 0 \qquad f(\sigma_2) = 0 \oplus 1.1 \oplus 1.0 = 1 \\
\sigma_3 &= 1\ 1\ 1\ 0\ 1 \qquad f(\sigma_3) = 1 \oplus 0.1 \oplus 1.1 = 0 \\
\sigma_4 &= 1\ 1\ 0\ 1\ 0 \qquad f(\sigma_4) = 0 \oplus 1.0 \oplus 1.1 = 1 \\
\sigma_5 &= 1\ 0\ 1\ 0\ 1 \qquad f(\sigma_5) = 1 \oplus 0.1 \oplus 0.1 = 1 \\
\sigma_6 = \sigma_0 &= 0\ 1\ 0\ 1\ 1
\end{aligned}$$

Böylece periyodu 6 olan $z = (010111)^\infty$ dizisi üretilir.

Örnek 5.3.2 Aynı fonksiyon kullanılarak periyodik olmayan bir dizi üretebiliriz. Eğer başlangıç fazını $\sigma_0 = (01010)$ seçersek:

	x_5	x_4	x_3	x_2	x_1
σ_0	0	1	0	1	0
σ_1	1	0	1	0	0
σ_2	0	1	0	0	0
σ_3	1	0	0	0	0
σ_4	0	0	0	0	0
σ_5	0	0	0	0	0

üretilen dizi $z = 0101000\ldots$, periyodik değildir.

Örnek 5.3.3 $f(x_1, x_2, x_3) = x_1 \oplus x_3$

	x_3	x_2	x_1
σ_0	1	0	1
σ_1	0	1	0
σ_2	1	0	0
σ_3	0	0	1
σ_4	0	1	1
σ_5	1	1	1
σ_6	1	1	0
$\sigma_0 = \sigma_7$	1	0	1

Periyodu 7 olan $z = (1010011)^\infty$ dizisi üretildi.

Bir f fonksiyonu, $a_i \in \{0, 1\}$ olmak üzere, $f(x_1, x_2, \ldots, x_L) = a_1x_1 \oplus a_2x_2 \oplus \ldots \oplus a_Lx_L$ şeklinde yazılabiliyorsa, bu fonksiyona doğrusal denir. FSR'nin kullandığı fonksiyon doğrusalsa bu üretece *doğrusal geri beslemeli kaydırmalı yazdırgaç* veya LFSR (linear feedback shift register) denir.

5.4 Üretecinin Sahip Olması Gereken Özellikler

1. Ürettiği dizi iyi istatistikler özellikler göstermelidir.
2. Periyodu büyük olan bir dizi üretmelidir.
3. Ürettiği dizinin doğrusal karmaşıklığı(linear complexity) büyük olmalıdır.

5.4.1 İstatistiksel Özellikler

Anahtar olarak kullanılacak dizinin tesadüfi ve kuralsız olması tercih edilir. Belirgin özellikleri olan bir dizi genellikle anahtar olarak kullanılmaz. Örneğin, $z = (10100100010000100000\ldots)$ dizisi kuralsız gözükmemektedir.

Dizinin kuralsız olmasını veya kuralsız bir diziye yakın olup olmadığını ölçen bir çok test vardır. AŞağıdaki üç testi LFSR'lar sağlar.

1. Dizinin bir tam periyodunda 1 ve 0 ların sayısı eşit olmalı, ya da aralarındaki fark 1 olmalıdır. Yani

$$0 \leq |(1\text{ 'lerin sayısı}) - (0\text{ 'ların sayısı})| \leq 1.$$

Örnek 5.4.1 $n = 21$ bit uzunluğunda $z = 000110101110010001101$ dizisinde 10 tane 1 ve 11 tane 0 vardır. Dolayısıyla aranan koşul sağlanır.

Örnek 5.4.2 $n = 14$ bit uzunluğunda ki $z = 10101010101010$ dizisinde
0 ve 1 yedi kere gözükürler ama bu dizi tesadüfi bir dizi değildir.

2. Tek çeşit karakterden oluşan bloklara *run* denir.

n bitlik bir dizide toplam $\frac{n+1}{2}$ tane run olması beklenir. Bunlardan

uzunluğu 1 olanların sayısının $\frac{n+1}{2^2}$,

uzunluğu 2 olanların sayısının $\frac{n+1}{2^3}$,

uzunluğu 3 olanların sayısının $\frac{n+1}{2^4}$,

$\vdots$

uzunluğu k olanların sayısının $\frac{n+1}{2^{k+1}}$ olması beklenir.

Örnek 5.4.3 $z = 00110001101$, n=11

$$\text{Beklenen run sayısı} \quad \frac{11+1}{2} = 6$$
$$\text{Uzunluğu 1 olan run sayısı} \quad \frac{11+1}{2^2} = 3$$

Dizinin run sayısı beklendiği gibi 6 dır, ancak uzunluğu 1 olan runların sayısı beklediği gibi 3 değil, 2 dir. Dolayısıyla, bu dizi run sayısı testini geçemez.

Örnek 5.4.4 $z = 10000101110110001111100110100 10$, n=31

Beklenen run sayısı	$\frac{31+1}{2} = 16$	✓
Uzunluğu 1 olan run sayısı	$\frac{31+1}{2^2} = 8$	✓
Uzunluğu 2 olan run sayısı	$\frac{31+1}{2^3} = 4$	✓
Uzunluğu 3 olan run sayısı	$\frac{31+1}{2^4} = 2$	✓
Uzunluğu 4 olan run sayısı	$\frac{31+1}{2^5} = 1$	✓
Uzunluğu 5 olan run sayısı	$\frac{31+1}{2^6} = 0.5 \cong 1$	✓

Bu dizi beklenen bütün değerleri sağlar ve run sayısı testinden geçer.

3. Periyodu p olan bir dizinin otokorelasyon fonksiyonu $C(\tau)$, dizinin kendisinin τ kadar kaydırılmasıyla oluŞan diziye ne kadar uyduğunu gösterir. Periyodu p olan $\{a_i\}$ dizisinin otokorelasyon fonksiyonunu

$$C(\tau) = \frac{1}{p}\sum_{i=1}^{p}(-1)^{a_i}(-1)^{a_{i+\tau}}$$

şeklinde ifade edilir. Her dizi için

$$C(0) = \frac{1}{p}\sum_{i=1}^{p}(-1)^{a_i}(-1)^{a_i} = \frac{1}{p}\sum_{i=1}^{p}(-1)^{a_i \oplus a_i} = \frac{1}{p}\sum_{i=1}^{p}1 = 1 \text{ dir.}$$

Bir dizinin rastgele olup olmadığına karar vermekte aranılan bir başka koşul da $C(\tau)$ fonksiyonunun

$$C(1) = C(2) = \ldots = C(p-1) \text{ eşitliğini sağlamasıdır.}$$

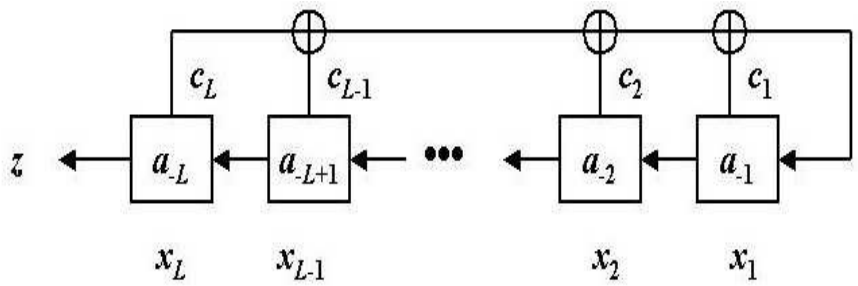
Örnek 5.4.5

$$\begin{aligned} a_i &= 0\ 0\ 0\ 1\ 0\ 1\ 1 \qquad C(0) = 1,\ p = 7 \\ (-1)^{a_i} &= 1\ 1\ 1\ -1\ 1\ -1\ -1 \end{aligned}$$

$$\begin{array}{rcccccccl}
(-1)^{a_{i+1}} & = & 1 & 1 & 1 & 1 & -1 & -1 & 1 & C(1) = \frac{1}{7}(1+1-1-1-1+1-1) = \frac{-1}{7} \\
(-1)^{a_{i+2}} & = & 1 & -1 & 1 & -1 & -1 & 1 & 1 & C(2) = \frac{1}{7}(1-1-1-1+1-1+1) = \frac{-1}{7} \\
(-1)^{a_{i+3}} & = & -1 & 1 & -1 & -1 & 1 & 1 & 1 & C(3) = \frac{1}{7}(-1+1-1+1+1-1-1) = \frac{-1}{7} \\
(-1)^{a_{i+4}} & = & 1 & -1 & -1 & 1 & 1 & 1 & -1 & C(4) = \frac{1}{7}(1-1-1-1+1-1+1) = \frac{-1}{7} \\
(-1)^{a_{i+5}} & = & -1 & 1 & 1 & 1 & 1 & -1 & 1 & C(5) = \frac{1}{7}(1+1-1-1-1+1-1) = \frac{-1}{7} \\
(-1)^{a_{i+6}} & = & -1 & 1 & 1 & 1 & -1 & 1 & -1 & C(6) = \frac{1}{7}(1+1-1-1-1+1-1) = \frac{-1}{7}
\end{array}$$

$C(1) = C(2) = C(3) = C(4) = C(5) = C(6) = \frac{-1}{7}$ olduğu için bu dizi testten geçer.

5.5 Doğrusal Geri Beslemeli Kaydırmalı Yazdırgaç (LFSR)



Resim 7: LFSR

L: LFSR'ın boyu

Başlangıç fazı $\sigma_0 : \ a_{-L}, a_{-L+1}, \ldots, a_{-2}, a_{-1}$

Geri besleme katsayıları (feedback coefficients): $c_1, c_2, \ldots, c_{L-1}, c_L \in \mathbb{Z}_2 = \{0, 1\}$

LFSR'ın doğrusal fonksiyonu:

$$f(x_1, x_2 \ldots x_L) = c_1 x_1 \oplus c_2 x_2 \oplus \ldots \oplus c_L x_L$$

Buna göre:

$$a_0 = c_L a_{-L} \oplus c_{L-1} a_{-L+1} \oplus \ldots c_2 a_{-2} \oplus c_1 a_{-1}$$

$$\Rightarrow \sigma_1 : \ a_{-L+1}, a_{-L+2}, \ldots, a_{-1}, a_0$$

$$a_1 = c_L a_{-L+1} \oplus c_{L-1} a_{-L+2} \oplus \ldots c_2 a_{-1} \oplus c_1 a_0$$

Genel olarak:

$$a_n = c_L a_{n-L} \oplus c_{L-1} a_{n-L+1} \oplus \ldots c_2 a_{n-2} \oplus c_1 a_{n-1}.$$

Bu recursive bağıntının karakteristik polinomu aynı zamanda LFSR'ın karakteristik polinomudur. Buna göre karakteristik polinom:

$$m(x) = x^L + c_1 x^{L-1} + c_2 x^{L-2} \ldots + c_{L-1} x + c_L.$$

LFSR'ın bağlayıcı polinomu (connection polynomial):

$$C(D) = 1 + c_1 D + c_2 D^2 + \ldots + c_{L-1} D^{L-1} + c_L D^L.$$

Bağlayıcı polinom ile karakteristik polinom arasındaki bağıntı aşağıdaki gibidir:

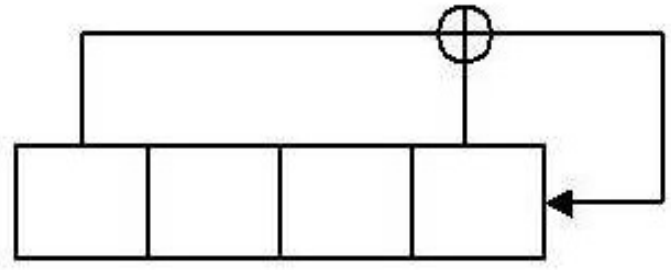
$$m(x) = x^L C\left(\frac{1}{x}\right).$$

Bir LFSR, boyu L ve bağlayıcı polinomu $C(D)$ ile belirlenir:

$$LFSR = \langle L, C(D) \rangle .$$

Örnek 5.5.1 LFSR=$\langle 4, C(D) = 1 + D + D^4 \rangle$

$c_1 = 1, \quad c_2 = c_3 = 0, \quad c_4 = 1 \;\Rightarrow f(x_1, x_2, x_3, x_4) = x_1 \oplus x_4$



Resim 8: $\langle 4, C(D) = 1 + D + D^4 \rangle$

Bu LFSR'ı çalıştırmak için başlangıç fazı olarak $\sigma_0 = (0\ 0\ 1\ 1)$ alınırsa, periyodu 5 olan $z = (001111010110010)^\infty$ dizisi üretilir.

x_1	x_2	x_3	x_4	$x_1 \oplus x_4$
0	0	1	1	1
0	1	1	1	1
1	1	1	1	0
1	1	1	0	1
1	1	0	1	0
1	0	1	0	1
0	1	0	1	1
1	0	1	1	0
0	1	1	0	0
1	1	0	0	1
1	0	0	1	0
0	0	1	0	0
0	1	0	0	0
1	0	0	0	1
0	0	0	1	1
0	0	1	1	$= \sigma_0$

LFSR'ın fonksiyonu doğrusal olduğundan $f(0, 0, \ldots, 0) = 0$ dır. Dolayısıyla başlangıç fazını 0 vektörü alınırsa 0 geri beslenir. 0 vektöründen başka bir faz görülmez. 0'dan farklı bir vektörle başlanırsa 0 vektörü faz olarak hiç görülmez.

5.5.1 Dizinin Periyodu

Boyu L olan bir LFRS'ın ürettiği dizinin periyodu en fazla $2^L - 1$ olabilir çünkü LFSR'da faz olarak L uzunluğunda vektörler gözükür. L uzunluğunda $2^L - 1$ tane vektör vardır. LFSR da 0 vektörünü görülmezse en fazla $2^L - 1$ tane değişik vektör görülebilir. Yani LFSR en fazla $2^L - 1$ adım sonra başlangıç noktasına geri döner.

$\langle L, C(D)\rangle$ LFSR'ının ürettiği dizinin periyodu $C(D)$ polinomunun çarpanlarına ayrılabilir olup olmamasıyla ve başlangıç fazıyla iliŞkilidir.

- Eğer $C(D)$ çarpanlarına ayrılıyorsa üretilen dizinin periyodu başlangıç fazına göre değişir.

  Örnek 5.5.2 $\langle L, C(D) = 1 + D^2 + D^4\rangle$
  $\Rightarrow \quad c_1 = 0, c_2 = 1, c_3 = 0, c_4 = 1 \quad \Rightarrow \quad f(x_1, x_2, x_3, x_4) = x_2 \oplus x_4$ ve $C(D) = 1 + D^2 + D^4 = (1 + D + D^2)^2$

	x_1	x_2	x_3	x_4
σ_0	1	0	0	0
	0	0	0	1
	0	0	1	0
	0	1	0	1
	1	0	1	0
	0	1	0	0
σ_0	1	0	0	0

Periyod = 6

	x_1	x_2	x_3	x_4
σ_0	1	1	1	1
	1	1	1	0
	1	1	0	0
	1	0	0	1
	0	0	1	1
	0	1	1	1
σ_0	1	1	1	1

Periyod = 6

	x_1	x_2	x_3	x_4
σ_0	1	0	1	1
	0	1	1	0
	1	1	0	1
σ_0	1	0	1	1

Periyod = 3

- Maksimum periyodda dizi üretmek için $C(D)$ polinomunun çarpanlarına ayrılamaz olması gerekir. Eğer $C(D)$ çarpanlarına ayrılamayan bir polinomsa dizinin periyodu başlangıç fazına bağlı değildir ve $C(D)$ polinomunun böldüğü $1+D^p$ polinomlarından en küçük dereceli olanın derecesi, üretilen dizinin periyoduna eşittir. Buradaki p sayısı $2^L - 1$ sayısının bir bölenidir. Örneğin $C(D) \mid 1 + D^5$ ve $C(D)$, $1 + D^k$, $k = 1, 2, 3, 4$, polinomlarını bölmüyorsa üretilen dizinin periyodu 5 tir.

- Dizinin periyodunun maksimum yani $2^L - 1$ olması için $C(D)$ polinomunun böldüğü

en küçük dereceli polinom $1 + D^{2^L-1}$ olmalıdır. Bunu sağlayan $C(D)$ polinomuna *ilkel polinom*(primitive polynomial) denir.

Maksimum periyodda dizi üreten bir LFSR'a *maksimum uzunlukta LFSR* denir.

5.6 Doğrusal Karmaşıklık (Linear Complexity)

Bir dizinin *doğrusal karmaşıklığı* (L.C.) onu üretebilecek LFSR'lardan en kısa olanın boyuna eşittir.

$z = 0000\ldots0\ldots$ dizisinin doğrusal karmaşıklığı 0 'dır.

$z = 1111\ldots1\ldots$ dizisinin doğrusal karmaşıklığı 1 'dir.

$z = \underbrace{0000\ldots1}_{n}$ dizisinin doğrusal karmaşıklığı n 'dir.

z	L.C	$C(D)$
0	0	0
1	1	$1+D$
01	2	$1+D^2$
001	3	$1+D^3$
011	2	$1+D+D^2$
100	1	1
101	2	$1+D^2$
110	2	$1+D+D^2$
111	1	$1+D$

Doğrusal karmaşıklık ile ilgili bazı özellikler:

- $z = z_0z_1z_2z_3\ldots$ dizisi için, bu diziden alınan her n bitlik z^n dizisinin doğrusal karmaşıklığı en fazla n olabilir. Yani $0 \leq \text{L.C.}(z^n) \leq n$ dir. Bu nedenle bir dizinin

doğrusal karmaşıklığı en fazla dizinin boyu kadardır.

- Eğer dizinin periyodu N ise L.C.$(z) \leq N$ dir.
- s ve t birer dizi olmak üzere L.C.$(s \oplus t) \leq$ L.C.(s) + L.C.(t) dir.

5.6.1 Doğrusal Karmaşıklık Profili (Linear Complexity Profile)

$s = s_0 s_1 s_2 \ldots$ bir dizi olsun. $N \geq 1$ için s^N sonlu dizisi $s^N = s_0 s_1 s_2 \ldots s_{N-1}$ şeklinde tanımlanır. O zaman her $N \geq 1$ için $L_N =$ L.C.(s^N) şeklinde tanımlanan $L_1, L_2, \ldots, L_N$ dizisine s^N dizisinin *doğrusal karmaşıklık profili* denir. Bu dizi aşağıdaki özellikleri gösterir:

- Eğer $j \geq i$ ise $L_j \geq L_i$ dir.
- $L_{N+1} > L_N$ olması için $L_N \leq N/2$ olması gerekir.
- Eğer $L_{N+1} > L_N$ ise $L_{N+1} + L_N = N + 1$ dir.

Profilde önemli noktalar $L_N \leq N/2$ olduğu yerlerdir. Bu noktalarda L_{N+1} artabilir. Boyu L_N olan hiç bir LFSR s^{N+1} dizisini üretmezse $L_{N+1} > L_N$ dir, dolayısıyla $L_{N+1} = L_N - N - 1$ dir.

Kriptografik anlamda iyi bir LFSR'ın ürettiği bir dizinin, doğrusal karmaşıklık profilinin grafiği $y = N/2$ doğrusuna yakın olmalıdır. Bu doğrudan fazla sapmamalıdır.

5.6.2 Berleckamp Massey Algoritması

Berleckamp Massey Algoritması sonlu bir dizinin doğrusal karmaşıklığını ve onu üretebilecek en kısa LFSR'ın bağlayıcı polinomu $C(D)$ yi bulmak için kullanılır.
$s^{N+1} = s_0 s_1 \dots s_{N-1} s_N$ dizisi verilsin. $\langle K, C(D) \rangle$ LFSR'ı $s^N = s_0 s_1 \dots s_{N-1}$ dizisini üretsin. s^{N+1} dizisinin ve $\langle K, C(D) \rangle$ LFSR'ının ürettiği dizinin $(N+1)$. terimi arasındaki farka *uymazlık sayısı* (next discrepancy) denir. Bu sayı

$$d_N = s_N + \sum_{i=1}^{L} c_i s_{N-i} \pmod 2$$

şeklinde hesaplanır. $d_N = 0$ olması için bu LFSR'ın s^{N+1} dizisinin $(N+1)$. terimini üretmesi gerekir.

Berleckamp Massey Algoritmasının işleyişi aşağıdaki gibidir:
Girdi olarak $s^n = s_0 s_1 s_2 \dots s_{n-1}$ dizisini alır.

1. BaŞlangıç konumu: $C(D) = 1,\ L = 0,\ m = -1,\ B(D) = 1,\ N = 0$
2. $N < n$ durumunda
   (a) $d = s_N + \sum_{i=1}^{L} c_i s_{N-i} \pmod 2$.
   (b) $d = 1$ ise
   $T(D) \longleftarrow C(D)$
   $C(D) \longleftarrow C(D) + B(D) D^{N-m}$
   Eğer $L \leq N/2$ ise $L \longleftarrow N + 1 - L$

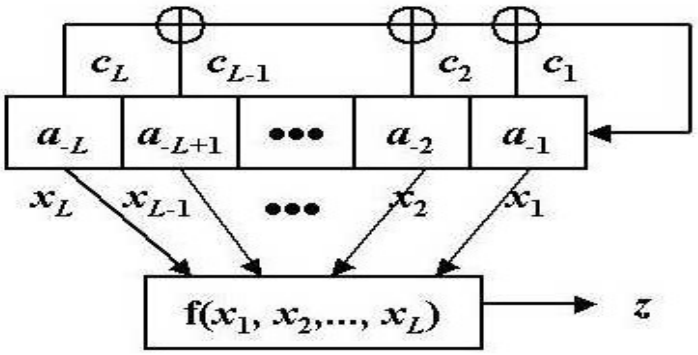
$$m \longleftarrow N$$
$$B(D) \longleftarrow T(D)$$

(c) $N \longleftarrow N + 1$

5.7 LFSR Kullanılarak Yapılan Akan Şifre Sistemleri

Bir dizinin doğrusal karmaşıklığı en fazla onu üreten LFSR'ın boyu kadar olabilir. Maksimum uzunlukta bir LFSR, doğrusal karmaşıklığı en fazla kendi boyuna eşit bir dizi üretebilir. Bu da doğrusal karmaşıklık için küçük bir sayıdır. Dizinin doğrusal karmaşıklığını arttırmak için çeşitli yollar vardır:

- LFSR'a doğrusal olmayan bir filtre bağlanır. Bu filtre doğrusal olmayan yani derecesi en az iki olan bir fonksiyondur.



Resim 9: Doğrusal olmayan filtre

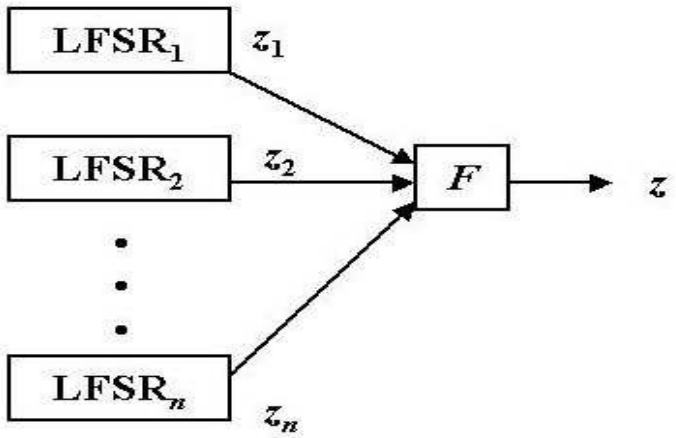
Doğrusal olmayan bir filtre: $f(x_L, x_{L-1}, \ldots, x_2, x_1) : \mathbb{Z}_2^L \to \mathbb{Z}_2$

Örnek 5.7.1 $\langle 3, 1 + D + D^2 \rangle$ ve doğrusal olmayan filtre $f(x_3, x_2, x_1) = x_1 \oplus x_1x_2 \oplus x_2x_3$

	x_3	x_2	x_1	$f(x_3, x_2, x_1)$
σ_0	1	0	1	1
	0	1	0	0
	1	0	0	0
	0	0	1	1
	0	1	1	0
	1	1	1	1
	1	1	0	1
σ_0	1	0	1	

$z = (1001011)^\infty$

- Birden fazla LFSR doğrusal olmayan bir fonksiyonla bağlanabilir.



Resim 10: Birden fazla LFSR'ın doğrusal olmayan bir fonksiyonla bağlanması

$F : (z_1, z_2, \ldots, z_n) : \mathbb{Z}_2^n \to \mathbb{Z}_2$ doğrusal olmayan bir fonksiyon.

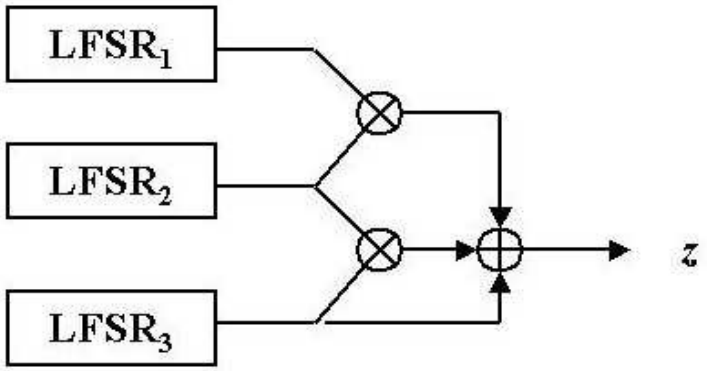
z dizisinin periyodu: $T(z) = \text{okek}(T_1, T_2, \ldots, T_n)$ dir.

Eğer LFSR'lar maksimum uzunlukta ise ve ürettikleri dizilerin periyodu ikiden büyük ve birbirlerinden farklı ise z dizisinin doğrusal karmaşıklığı

$F(L_1, L_2, \ldots, L_n)$ dir.

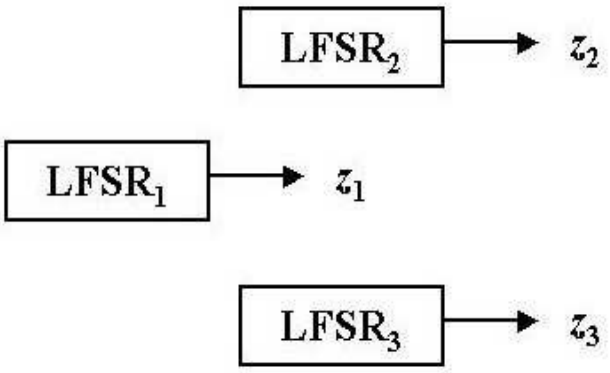
Örnek 5.7.2 *Geffe Üreteci:* Üç tane LFSR kullanır.

$$F(x_1, x_2, x_3) = x_1x_2 \oplus (1 \oplus x_2)x_3.$$



Resim 11: Geffe Üreteci

- Saat Kontrollü Üreteçler (Clock Controlled Generators):
  - Değişen Adımlı Üreteç (*Alternating Step Generator*): Üç tane LFSR kullanılır.



Resim 12: Değişen Adımlı Üreteç

$\text{LFSR}_1 = \langle L_1, C_1(D) \rangle$

$\text{LFSR}_2 = \langle L_2, C_2(D) \rangle$

$\text{LFSR}_3 = \langle L_3, C_3(D) \rangle$

LFSR_1 çalıştırılır;

$x_1 = 1$ ise LFSR_2 çalıştırılır. LFSR_3'ün bir önce ürettiği bit tekrar eder, LFSR_3 daha önce çalışmamışsa 0 alınır.

$x_1 = 0$ ise LFSR_3 çalıştırılır. LFSR_2'nin bir önce ürettiği bit tekrar eder, LFSR_2 daha önce çalışmamışsa 0 alınır.

LFSR_2 ve LFSR_3 XOR işlemine tabi tutulur. Eğer LFSR_1 periyodu 2^{L_1} olan bir dizi üretiyorsa LFSR_2 ve LFSR_3 maksimum periyodda diziler üretiyorsa ve o.b.e.b.(L_1, L_2) ise üretilecek z dizisinin

1. periyodu $2^{L_1}(2^{L_2} - 1)(2^{L_3} - 1)$,
2. doğrusal karmaşıklığı $(L_2 + L_3)2^{L_1 - 1} < \text{L.C.}(z) \leq (L_2 + L_3)2^{L_1}$ dir.

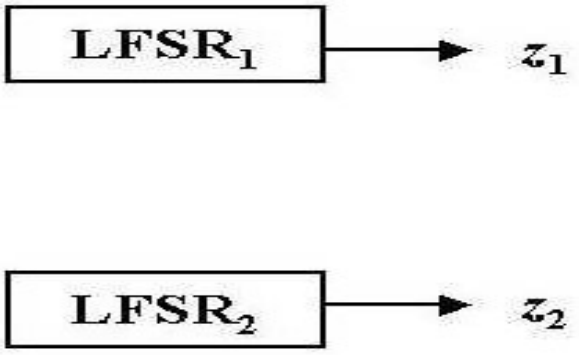
Örnek 5.7.3 $\text{LFSR}_2 \rightarrow 1,1,0,0,1,1,0,1,\ldots$

$\text{LFSR}_1 \rightarrow 0,1,1,1,0,0,1,1,\ldots$

$\text{LFSR}_3 \rightarrow 0,0,1,1,0,1,1,0,\ldots$

x_2	0	1	1	0	0	0	0
x_1	$0\downarrow$	$1\uparrow$	$1\uparrow$	$1\uparrow$	$0\downarrow$	$0\downarrow$	$1\uparrow$
x_3	0	0	0	0	0	1	1
z	0	1	1	0	0	1	1

– Küçülen Üreteç (*Shrinking Generator*): İki tane LFSR kullanılır. İkisi de aynı anda çalışır.



Resim 13: Küçülen Üreteç

$x = 1$ ise y 'den al.

$x = 0$ ise y 'den alma.

$\text{LFSR}_1 = \langle L_1, C_1(D) \rangle \Rightarrow T(x) = 2^{L_1} - 1$

$\text{LFSR}_2 = \langle L_2, C_2(D) \rangle \Rightarrow T(y) = 2^{L_2} - 1$

1. L_1 ve L_2 aralarında asal ise oluşan dizinin periyodu
   $(2^{L_1-1})2^{L_2}-1$,
2. doğrusal karmaşıklığı ise $L_2(2^{L_1-2}) < \text{L.C.}(z) \leq L_2(2^{L_1-1})$ dir.

Örnek 5.7.4 $\text{LFSR}_1 \rightarrow (101)^\infty$

$\text{LFSR}_2 \rightarrow (0101101)^\infty$

x	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1
y	0	1	0	1	1	0	1	0	1	0	1	1	0	1	0	1	0	1	1	0	1
z	0		0	1		0	1		1	0		1	0		0	1		1	1		1

$z = (00101101101111)^\infty$