YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DOĞRUSAL GERİ BESLEMELİ ÖTELEMELİ YAZICI (LFSR) TABANLI ŞİFRELEME SİSTEMLERİNE KORELASYON SALDIRILARI

Matematikçi/Matematik Öğretmeni Uğur Kaşif BOYACI

FBE Matematik Mühendisliği Anabilim Dalında Hazırlanan

YÜKSEK LİSANS TEZİ

Prof. Dr. A. Beneş

Tez Danışmanı : Yrd. Doç. Dr. Ayla ŞAYLI(YTÜ)

İncelenmiştir
Y. Doç. Dr. Ayla SAYLI

Y. Doç. Dr. İsmet

İSTANBUL,2003

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ

c	(Dizi şifrelemede) çıktı dizisi
c_i	Sonlu cisim elemanı katsayı
C	(Kriptolojide) şifreli metin
C	Kod
D	Edit uzaklığı
D	(LFSR polinomu) değişkeni
d	Uzaklık
E	(Kriptoloji) Şifreleme fonksiyonu
E	(Kodlama teorisi) kodlama
f	Booleen fonksiyon
F_q	Sonlu cisim
G	Üreteç matrisi
H	Parite kontrol matrisi
K	Anahtar
L	LFSR dizisi boyu
L	Doğrusal karmaşıklık
N	Alınan dizi uzunluğu
P	Açık metin
p	Olasılık değeri
$P(x\|y)$	Koşullu olasılık fonksiyonu
u	Bilgi dizisi
v	Kodlama çıktısı
z	(Dizi şifrelemede) kayar anahtar
$\oplus$	İkili toplama

KISALTMA LİSTESİ

BSC	İkili Simetrik Kanal
LFSR	Doğrusal Geribeslemeli Ötelemeli Yazıcı

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖZET

Bilgi güvenliğinde gizliliğin hızlı bir şekilde sağlanması gerektiği bazı durumlarda, dizi şifreleme sistemleri kaçınılmazdır.Dizi şifreleme sistemleri arasında ise doğrusal geribeslemeli ötelemeli yazıcı (LFSR) tabanlı sistemler oldukça yaygın kullanılmaktadır.

Bir LFSR çıktısı bazı güzel istatistiksel özelliklere sahip olsa da, tek başına kullanılamaz. Birkaç LFSR çıktısının bir arada kullanılması da kriptografik saldırılara karşı tamamen güvenli olduğu anlamına gelmez. Bu tezde, LFSR tabanlı şifreleme sistemlerine düzenlenen en yaygın saldırı şekillerinden biri olan korelasyon saldırılarına göz atılmıştır. Saldırılar anlatılmadan önce saldırılarla ilgili temel bazı kavramlar sunulmuştur.

Burada anlatılan saldırılar, LFSR tabanlı sistemlere yapılan korelasyon saldırılarının tümünü kapsamamaktadır. Bunun sebebi hem bu saldırıların adedinin çokluğu, hem de sürekli yeni saldırıların keşfedilmesidir. Diğer yandan, seçilen saldırılar hem bu tür saldırıların temelini kavratacak, hem de farklı bakış açılarını gösterecek şekilde seçilmeye çalışılmıştır. Sadece birleştirici fonksiyonlarla yapılan LFSR tabanlı sistemler değil, zaman kontrollü sistemlere de korelasyon saldırısı düzenlenebileceği bazı örneklerle gösterilmiştir.

Anahtar kelimeler: Kriptoloji, dizi şifreleme, doğrusal geribeslemeli ötelemeli yazıcı, korelasyon saldırıları, zaman kontrollü üreteçler, kodlama teorisi.

ABSTRACT

Stream cipher is a must today in many applications where security has to be obtained in a fast way. Among the stream ciphers, linear feedback shift register based cryptosystems are commonly used.

Despite some good statistical properties, an LFSR output cannot be used as a keystream generator alone. A few LFSRs combined together does not mean that the system is totally safe against cryptographic attacks. In this thesis, several examples of correlation attacks, one of most common types of attacks to LFSR based cryptosystems are examined. Before the explanation of the attacks, some fundamental concepts are given.

The attacks presented here do not include all of the correlation attacks on LFSR based systems. This is because there are many of them and new ones appear frequently. However, the examples chosen try to reflect both the fundamentals of these attacks and different points of view. It is shown that not only LFSR based systems with a combining function but clock-controlled systems may also be prone to this kind of attacks.

Keywords: Cryptology, stream ciphers, linear feedback shift registers, correlation attacks, clock controlled generators, coding theory.

1. GİRİŞ

Kriptoloji biliminin en büyük uğraşlarından biri gizliliği sağlamak olmuştur. Günümüzde gizliliği sağlamanın yanı sıra bunu sağlayacak sistemin hızı, kolayca donanım ve yazılıma aktarılabilmesi gibi konular da sistemin tasarımında etkili olmaktadır. Dizi şifreleme sistemlerinde en önemli hususlardan olan şifreleme hızı, üretilen dizinin rassal görünümlülüğü ve donanıma kolayca geçirilebilirliği açısından doğrusal geri beslemeli ötelemeli yazıcılar (LFSR) yaygın olarak kullanılmaktadır. Üstelik bu sistemlerin temeli olan yinelemeli dizilerin matematikçiler tarafından (başka dallarda da karşımıza çıktığından) oldukça iyi incelendiği söylenebilir. Bu dizilerin oluşturacağı dizilerin doğrusal karmaşıklık gibi birçok özelliği iyi incelenebildiğinden teorik olarak analizi de görece olarak birçok sisteme göre daha iyi yapılabilmektedir. Bu özelliğin kriptolojide sıklıkla karşılaşıldığı gibi çift yönlü olduğu, yani birçok saldırıya da kapı açabileceği de bir gerçektir.

Özellikle booleen fonksiyonların tabii yapısı ve yazıcının doğrusallığından dolayı LFSR tabanlı bir sistemin hem lineer tutarlılık testine dayanıklı olması hem de ilişkiye bağışık olması (korelasyona dayanıklı olması) zordur. Özellikle düşük anahtar boylarında tasarlanan birçok eski sistem, hatta yakın zamanda usta kriptanalistler tarafından hazırlanmış bazı sistemler korelasyon saldırıları ile kırılabilmiştir.

Siegenthaler'in yaptığı ilk korelasyon saldırısından cesaretle birçok saldırı düzenlenmiş fakat bunlar, Johannson ve Jönnson'un turbo kodlarla yapılan saldırısına kadar karmaşıklığı düşürebilme açısından temel bir değişiklik getirememişlerdir. Son bir kaç senede ise özellikle Golic'in katkılarıyla korelasyon saldırılarında hem nitelik, hem de nicelik açısından epey yol katedilmiştir. Son yıllarda, kriptoloji konferanslarında korelasyon saldırıları ile ilgili yeni bir sonucun ortaya çıkması şaşırtıcı olmamaktadır. Dolayısıyla bir tezde bütün saldırıları bütün yönleriyle sığdırmak hem zaman açısından hem hacim açısından oldukça güçtür. Tek bir saldırının bütün detaylarıyla uzun uzun anlatılması ise daha kuvvetli bir algoritmanın anlatılanı "tedavülden kaldırması" durumunda beklenen faydayı getirmeyeceğinden tercih edilmedi.

Bu tezde, uygulanan saldırının matematiksel altyapısı ve uygulanan sistemin çeşidi açısından farklı bakış açılarını yansıtmaya gayret edildi. Algoritmalara ait teorik argümanların verilmesinin yanı sıra saldırıların pratiğe dökülebilmesine yardımcı olacak şekilde tanımı tarif edilmeye çalışıldı, makale yazarlarına ait bazı deney sonuçları sunuldu.

Bu tez sırasında emeği geçen herkese teşekkür etmem gerekirse tezin hatırı sayılır bir bölümünü onlara ayırmak gerekecekti. Diğer yandan, hiç isim anmamak büyük bir haksızlık ve saygısızlık olacaktı. Bu yüzden, tez hocam Yrd. Doç. Ayla Şaylı'ya, tez sırasında görüşlerinden faydalandığım ve beni kriptoloji alanında çalışmaya teşvik eden Prof Dr. İsmail Güloğlu'na, istatistik ve olasılık konusunda epey yararlandığım Prof Dr. Ali Doğan Aksoy'a ve Prof. Dr. Erkan Türe'ye, iş yerindeki elverişli koşulların sağlanması açısından Enstitü yöneticilerine ve mesai arkadaşlarımdan özellikle Fatih Birinci, Zernişan A. Emirleroğlu ve Orhun Kara'ya, evinde sağladığı huzurlu çalışma ortamı için teyzeme teşekkürü bir borç bilirim.

2. KRİPTOLOJİYE GİRİŞ

Kriptoloji, Yunanca *krifos* (saklı, gizli) veya *kripti* (gizli geçit, sığınak) kelimeleri ile *logos* söz, bilgi) kelimelerinden türemiştir. Başlangıcı Sezar'a kadar dayandırılan kriptoloji bilimi tarih boyunca genelde (yazılı bir mesaj için) gizlilik bilimi olarak anlaşılmışsa da günümüzdeki gelişmeler sonucunda mesaj bütünlüğü, kimlik doğrulama teknikleri sağlaması nedeniyle güvenilirlilik bilimi olarak karşımıza çıkmaktadır. Kriptoloji bilimi bilginin güvenli şekilde saklanması, güvenli erişim, bilgiye sadece yetkili kişilerin ulaşmasını sağlayan bilgi güvenliği(information security)nin bir alt kolu olarak bir mühendislik dalı şeklinde görülebileceği gibi bu bilimde kullanılan birçok tekniğin matematiğin özellikle cebir dalıyla ilintili olmasından dolayı matematiğin bir alt dalı olarak da görülebilir. Kriptoloji iki ana dala ayrılmaktadır: Kriptografi ve kriptoanaliz.

Kriptografi: (Yunanca *graf*: Yunanca yazı oradan da İng. Çizgi,grafik,yazılı çalışma) Bilgi güvenliğinin mesaj gizliliği, kimlik doğrulama, veri bütünlüğü, sayısal imza gibi konularına ilişkin (matematiksel) tekniklerin incelenmesi.

Kriptoanaliz: Kriptografik tekniklerin (matematiksel) tekniklerle alt edilmeye çalışılması.

2.1 Kriptoloji ile ilgili kullanılan bazı terimler:

Kriptosistem: Bilgi güvenliğini sağlamak amacıyla kullanılan kriptografik yapıtaşları ile oluşturulmuş sistemlerin genel adı. (Genelde bu terim şifreleme gibi gizlilik sağlayan yapıtaşları kümesi için kullanılır.)

Kriptosistemlerde temelde mesaj alışverişi yapan iki taraf bulunur. Kriptolojide bunlardan birinin (genelde mesajı yollayanın) Alice(A) diğerinin Bob(B) olarak adlandırılması adet olmuştur.

Şifreleme(Encryption): Yalnız mesajı alacak kişinin anlayabileceği şekilde belli, (matematiksel) iyi tanımlı işlemlerden geçirmek.

Deşifre etme: Şifreli mesajdan açık mesajı elde etme işlemi.

Algoritma: Değişken girdisini alıp, iyi tanımlanmış işlemlerden geçirerek bir çıktı veren yöntem. Kriptologlar genelde şifreleme algoritmalarına kısaca algoritma demektedir.

Açık Mesaj: Herhangi bir işlemden geçirilmemiş, hususi olarak şifrelenmemiş metin. P(Plaintext) ile gösterilir.

Şifreli Mesaj: Şifreleme algoritmasından geçirilmiş metin. C(Cipher) ile gösterilir.

Anahtar: Açık mesaja bağlı olmayan şifreleme işleminin yapılabilmesi için kullanılan veri.

Exor (exclusive or): Mod 2'de toplama. $\oplus$ ile gösterilir.

2.2 Kerkhoff Kuralları

Kriptolojinin en büyük uğraşı şifreleme sistemleridir. Hollandalı bir dilbilimci olan Auguste Kerkhoff'un 1883 yılında yazdığı bir makalede geçen kurallarına günümüzde şifreleme sistemlerinin bilinen ilk mühendislik kriterleri olarak bakılmaktadır. Kerkhoff'a göre:

1) Sistem, teorik olarak kırılabiliyor olsa bile en azından pratikte kırılamamalıdır.
2) Karşı taraf kullanılan tekniği bilse bile sistem tehlike altında olmamalıdır. Diğer bir deyişle, (yukarıda belirtildiği gibi) güvenlik, anahtar seçimine dayanmalıdır.
3) Anahtar kolayca hatırlanmalı ve değiştirilebilmelidir.
4) Kriptogramlar telgrafla iletilebilmelidir.
5) Cihaz veya dokümanlar taşınabilir ve tek kişi tarafından kullanılabilir olmalıdır.
6) Sistem, uzun bir kurallar listesi veya beyin zorlayıcı bir kullanım içermemelidir.

Bu kuralların hepsinin günümüzde aynen bu şekilde geçerli olduğunu söyleyemeyiz ama ilk iki madde, özellikle "Şifreleme sisteminin güvenliği, sistemin nasıl şifreleme yaptığının karşı kriptanalist tarafından bilinmemesine değil sadece anahtara bağlı olmalıdır." prensibi hala geçerlidir. Bununla beraber, kullanılan tekniğin bilinmemesinin düşman kriptanaliste ek bir zorluk çıkaracağı gözardı edilmemelidir.

Özellikle parola tabanlı şifreleme sistemlerinde "kolayca hatırlanabilme" pek istenmeyen bir şey olsa da, üçüncü madde günümüzde anahtarların dağıtımı, erişimi, saklanması gibi işlemlerle uğraşan anahtar yönetimi(key management) konusunda ilk derli toplu ilke olarak görülebilir.

Dördüncü maddeyi arzu edilen iletişim vasıtasını güvenli görüşebilir hale getirme, beşinci maddeyi kriptografik cihazın ergonomikliği, altıncı maddeyi de kullanıcı kolaylığı adına yakın zamana kadar yol gösteren ilkeler olarak görebiliriz.

2.3 Simetrik Anahtarlı ve Açık Anahtarlı Sistemler

Şifreleme sistemleri, *simetrik anahtarlı(symmetric key)* ve *açık anahtarlı (public key)* olmak üzere ikiye ayrılır.

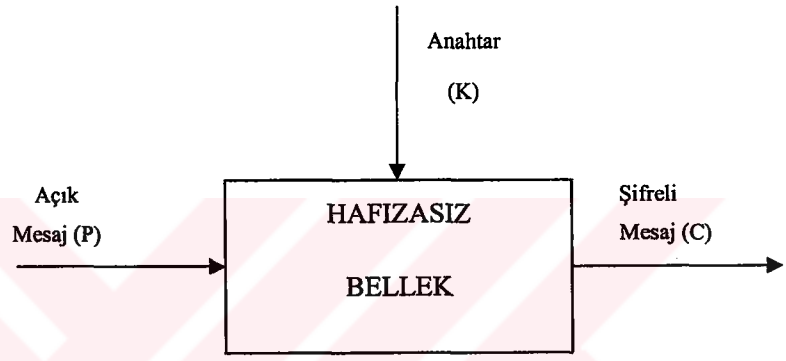
Simetrik sistemlerde Alice ile Bob aynı anahtara sahiptir. Yüzyıllardır şifreleme sistemi denilince akla simetrik anahtarlı sistemler geldiğinden bu sistemlere İngilizce'de *conventional* ("alışılagelmiş"), anahtarın gizli tutulması gerektiğinden *secret key ("gizli anahtarlı")* sistem denildiği de olmaktadır. Şifreleme işlemi ile şifre çözme işlemi benzer şekilde yapılır.

Açık anahtarlı sistemlerde biri *özel(private)* diğeri *açık(public)* olmak üzere iki ayrı anahtar vardır ve açık anahtar herkes tarafından bilinebilir. Alice, Bob'a mesaj atmak istediğinde Bob'un açık anahtarını temin eder ve göndereceği mesajı Bob'un açık anahtarı vasıtasıyla şifreler. Mesajı alan Bob, sadece kendisinin bildiği özel anahtarını kullanarak şifreyi açar. Açık anahtarlı sistemler tamsayıları çarpanlarına ayırma, sonlu cisimlerde ayrık logaritma gibi NP(non-polynomial) yani genel çözümü polinomsal zamanda olan bir algoritma bulunamayan matematiksel problemlere dayanmaktadır. Şifreleme işlemi açık anahtar sayesinde herkes tarafından yapılabilmekte iken şifreli mesajı açabilmek ancak ek bir özel bilgi (özel anahtar) sayesinde yapılabilmektedir.

Birbiriyle haberleşme isteyen n kişi için, açık anahtarlı sistemlerde her bir kişi için bir açık bir özel anahtar gerekli ve yeter olduğundan $2n$ anahtar gerekmekte iken, simetrik anahtarlı sistemlerde her bir çift için bir gizli anahtar gerektiğinden $n(n-1)/2$ anahtar gerekmektedir. 1970'lerde ortaya çıkan açık anahtarlı sistemler sadece haberleşme için gerekli anahtar sayısını $O(n^2)$ mertebesinden $O(n)$ mertebesine düşürmekle kalmamış, simetrik anahtarla yapılamayan mesaj imzalama gibi bazı kavramlara kapı açmasıyla kriptolojide yeni bir çığır açılmasına yol açmıştır. Fakat açık anahtarlı sistemlerin çoğu güvenlik sağlayabilmek için yüksek anahtar boyuna gerek duymaktadır ve gizli anahtarlı sistemlerle kıyaslandıklarında çok yavaş kalmaktadırlar. Bu yüzden açık anahtarlı sistemler uzun mesajların şifrelenmesinde neredeyse hiç kullanılmamaktadır.

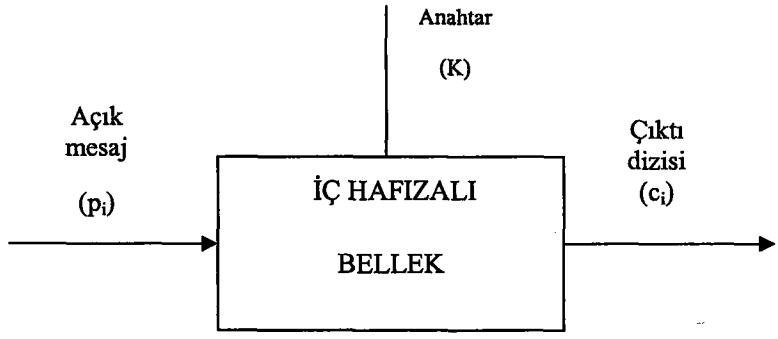
2.4 Blok ve Dizi Şifreleme

Simetrik anahtarlı sistemler ise *blok şifreleme (block cipher)* ve *dizi şifreleme(stream cipher)* olmak üzere ikiye ayrılır. Blok şifrelemede, açık mesaj bloklara ayrılır ve aynı anahtar K ile kapatılır. Kapatılan her bir bloktaki bit sayısına blok uzunluğu adı verilir. Günümüzde 64 bit blok uzunluklu, (56 bit anahtarlı) DES gibi algoritmalar tarihe karışmak üzeredir. 128 bit blok için 128, 192 ve 256 bit anahtar yaygın olarak blok şifreleme parametreleridir. Bir blok şifreleme $C_i = E_K(P_i)$ şeklinde ifade edilebilir. P_i ve C_i sırasıyla açık mesajın ve şifreli mesajın i nci bloklarını temsil etmektedir. E_K ise şifreleme işleminin K anahtarı ile yapıldığını göstermektedir.



Şekil 2.1 Blok şifreleme

Dizi şifrelemede ise başlangıç durumu anahtara bağlı olan bir *iç bellek (internal memory)* vardır. Çıkan şifreli mesaj belleğin o andaki durumuna (state) bağlıdır. Dizi şifreleme işlemi $z_i = f(K, c_{i-1}, p_{i-1})$ şeklinde ifade edilebilir. Dizi şifreleme gösteriminde küçük harf kullanılmasının nedeni daha küçük birimlerin(bit ya da byte) şifrelenmesidir.

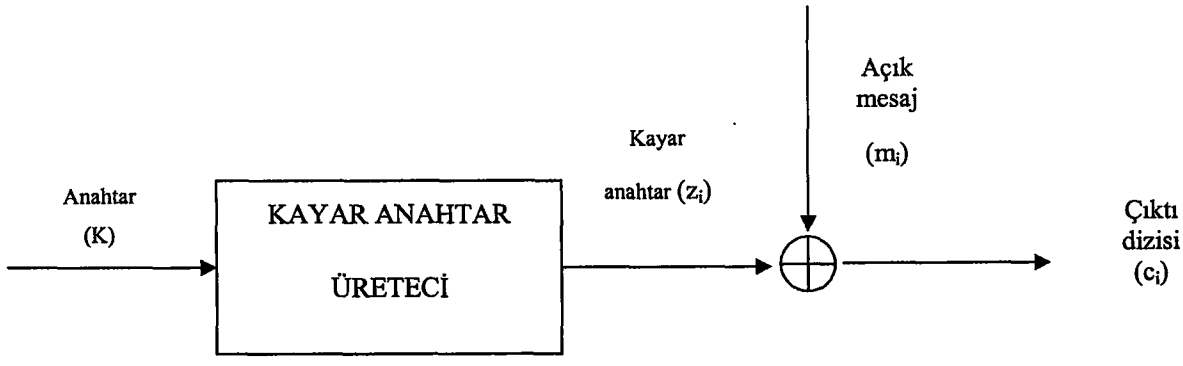


$$z_i = f(K, c_{i-1}, p_{i-1}) \; , \; E_{zi}(p_i) = c_i$$

Şekil 2.2 Dizi şifreleme

Dizi şifreleme sistemleri kendi içinde ikiye ayrılır. Senkron ve asenkron. Senkron sistemlerde kayaranahtar açık mesaj ve şifreli mesajdan bağımsız olarak üretilir. Asenkron sistemde ise kayaranahtar dizisi anahtarın ve sabit sayıda eski şifreli metin karakterlerinin bir fonksiyonu tarafından üretilir.

İkili toplamalı dizi şifreleme sisteminde bir *K ilk anahtar*ından *(initial key)* uzun bir anahtar dizisi üretir. Üretilen bu anahtar dizisine *kayar anahtar(running key)* denir. Verilen gizli anahtardan kayar anahtar üreten cihaza *anahtar dizisi üreteci (key stream generator)* veya *kayar anahtar üreteci (running key generator)* denir. Mesajla bu anahtar dizisi toplanarak şifreli mesaj elde edilir.



Şekil 2.3 İkili (toplamalı) dizi şifreleme

Dizi şifreleme sistemleri bir seferde az sayıda bit şifrelenmesine karşılık yüksek hızları ve kolay eşzamanlı hale getirilebilmelerinden (senkronizasyon) dolayı bazı durumlarda (örneğin telsiz haberleşmesinde veya bazı internet protokollerinde) blok şifreleme sistemlerine tercih edilmektedir. Ayrıca dizi şifreleme sistemlerinin donanım kolaylığı, "buffer" sınırının önemli olduğu durumlarda veya alınan karakterlerin ayrı ayrı işlenmesi gerektiği durumlarda blok şifrelemeye tercih edilmelerine yol açar.

Bilinen ilk meşhur şifreleme sistemi Roma İmparatoru Julius Ceasar tarafından kullanılmıştır. Bu yüzden bu sisteme Ceasar şifresi denilmektedir. Bu sistemi dizi şifrelemeye örnek olarak gösterebiliriz. Şöyle ki:

Ceasar mesajın her harfini 3 karakter kaydırıyordu. Türkçe için bu şifrelemeyi formüle dökecek olursak edecek olursak

$$z_i = x_i + 3 \pmod{29} \tag{1.1}$$

olarak gösterebiliriz. z_i burada her zamanki gibi çıktı karakterini, x_i ise girdi karakterini göstermektedir. Sistemin kırılması çok uzun sürmedi. Julius'un yeğeni Augustus mesajı 4 karakter kaydırmayı denedi ise de bu sistem de görüldüğü gibi güvensizdi. 1917'de Almanlar'ın kıramayacağı bir şifre tasarlaması istenen (Gilbert) Vernam'ın adıyla anılan "benzer" bir şifreleme sistemi ise tam aksine "mükemmel gizlilik" sağlamaktadır.

Tanım: Bir kriptosistemde, gözlenen her çıktı karakteri z, açıkmesaj karakteri x için $P(x|z) = P(x)$ ise sisteme *mükemmel gizlilik(perfect secrecy)* sağlıyor denir.

Vernam şifresini şu şekilde formüle dökebiliriz:

$$z_i = x_i + k_i \pmod{29} \tag{1.2}$$

Burada k_i birbirinden bağımsız, her harfin gelme olasılığının düzgün dağılımlı olduğu rassal bir dizidir ve "tek kullanımlık" tır. Bu yüzden bu şifreleme sistemi daha çok *One Time Pad* ("Tek Seferlik Ekleme") olarak bilinir. Shannon(1949) kriptoloji biliminin en temel sonuçlarından biri olan makalesinde Vernam şifresinin "kırılamaz" olduğunu ispatlamıştır. Bunun nedeni anahtarın istatistiğinin mesaja bağlı olmamasıdır. Fakat yine aynı ispatta, "mükemmel gizli" bir sistemin anahtar entropisinin mesaj entropisinden daha yüksek olması gerektiğini göstermiştir. Entropi kavramı fizikte olduğu gibi bilgi teorisinde de (sistem)

belirsizliğini belirtmektedir. Bu da anahtar boyunun mesaj boyundan daha uzun olmasını gerektirmektedir. Anahtarın gizli iletilmesi gerektiğinden bu şifreleme pratik değildir. Geçmişte, "tek kullanımlık" kuralı ihlal edilerek iki tarafın da "kod kitapçığı" (code book) denilen ve sonuna gelindiğinde tekrar başına dönülen uzun anahtar dizi dosyaları kullanılması neticesinde önemli bilgi sızdırmaları olmuştur. (Kahn 1967) Ayrıca bu dizilerin saklanması da zor olmaktadır. Bu yüzden tek seferlik ekleme pek başvurulan bir yöntem değildir. Pratikte bu sorunu aşmak için yukarıda da bahsi geçen *ikili toplama dizi şifreleme (binary additive stream cipher)* sistemlerinde açık metine (kısa) bir anahtardan deterministik şekilde kayar anahtar dizisi çıktısı ekleyen sistemler kullanılmaktadır. Bu diziyi üreten mekanizmaya anahtar dizisi üreteci denir. Bu durumda sistemin güvenliği en fazla anahtarın entropisi kadar olacaktır. Diğer yandan düşman kriptanalist üretilen bu diziyi tahmin edememelidir.

3. DOĞRUSAL GERİBESLEMELİ ÖTELEMELİ YAZICILAR TEORİSİNE GİRİŞ

Tanım: L bir pozitif tamsayı ve $c_0, c_1, \ldots, c_{L-1}$ içlerinden en az biri sıfırdan farklı olmak üzere F_q'dan katsayılar olsun. Her $n \geq L$ için, $a_j = c_1 a_{j-1} + c_2 a_{j-2} + \ldots + c_L a_{j-L}$ koşulunu sağlayan $\{a_n\}$ dizilerine L ninci derece *(homojen) doğrusal yinelemeli dizi* denir. Dizinin ardışık herhangi L elemanına *durum(state),* diziyi tamamen belirleyen ilk L terime ise *ilk durum(initial state)* adı verilir.

Tanım: *Doğrusal Geribeslemeli Ötelemeli Yazıcı (Linear Feedback Shift Register)* , bir homojen lineer yinelemeli dizi üreten bir sonlu durum makinesidir. Kısaca LFSR ile gösterilir.

LFSR saat yönünde çalışır. Dizinin o andaki durumu bir yazıcıda tutulur. L ardışık terimin yer aldığı bir iç bellek olan yazıcı L göze ayrılmıştır ve her bir göz $\{a_n\}$ dizisinin bir terimini içerir. L'ye LFSR'ın boyu denir. LFSR'ın bir *zaman*ında*(clock)* aşağıdaki işlemler yapılır:

1) $a_{n-1}, a_{n-2}, \ldots, a_{n-L+1}, a_{n-L}$ durumundan $a_L = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_L a_{n-L}$ hesaplanır. *(doğrusal yineleme)*

2) a_{n-i} , i nci gözden $i+1$ nci göze *ötelenir. (shift)*

3) a_n ilk göze *yazılır. (register)*

4) a_{n-L} çıktı olarak dışarıya verilir.

Tanım: L uzunluklu bir LFSR, F_q cisminde $a_n = \sum_{i=1}^{L} c_i a_{n-i}$ bağıntısıyla $\{a_n\}$ dizisini üretiyor olsun. $g(D) = \sum_{i=1}^{L} c_i D^i \in F_q(D)$ (2.1)

polinomuna LFSR'ın *geribesleme polinomu* ve

$f(x) = 1 - g(D)$ (2.2)

polinomuna LFSR'ın *karakteristik polinomu* denir. Karakteristik polinomun derecesi L'dir.

Önerme: F_q cismi üzerinde boyu L olan bir LFSR'ın ürettiği sıfır olmayan $\{a_n\}$ dizisi periyodiktir ve periyodun boyu $p \leq q^L - 1$ olur.

İspat: F_q üzerinde uzunluğu L olan sıfır olmayan vektörlerin sayısı $q^L - 1$ adettir. (0 vektörü 0 dizisini üretir.) Birbirinin aynı olan iki vektör aynı diziyi üretir. Öyleyse bir LFSR dizisi en fazla bütün sıfır olmayan vektörleri dolaştıktan sonra yani $q^L - 1$ adım sonra başa dönmüş olur.

Tanım: $C(D) = 1 + c_1 D + c_2 D^2 + \ldots + c_L D^L$ şeklinde verilen bir polinomun (F_q üzerindeki) **periyodu**, $C(D)$ polinomunun $1 - \mathrm{D}^{\mathrm{N}}$ polinomunu böldüğü en küçük N pozitif tamsayısıdır.

Tanım: $C(D) = 1 + c_1 D + c_2 D^2 + \ldots + c_L D^L$ şeklinde verilen bir polinom olsun. Eğer F_q cismi üzerinde $B(D) = c_0^{-1} C(D)$ polinomunun periyodu $q^L - 1$ ise, $C(D)$ polinomuna (F_q üzerinde) *ilkel polinom* denir.

Teorem: F_q cismi üzerinde boyu L olan bir LFSR, ancak ve ancak karakteristik polinomu ilkelse, *maksimum periyod*a sahiptir.

İspat: LFSR'lar için bu çok bilinen teoremim ispatı çok zor olmamakla birlikte birkaç yeni tanım ve lemma gerektirdiğinden burada verilmeyecektir. Ana fikir olarak tamamen sıfır olmayan bir LFSR dizisinin periyodunun, karakteristik polinomun periyodunu böldüğünün gösterilmesi gerekir.

Dizi şifrelemede LFSR kullanımının yaygın olmasının nedenlerinden biri de F_2 üzerinde derecesi L olan (L çok büyük olmamak şartıyla) ilkel polinomların kolayca bulunabilmesidir. Daha genel olarak, herhangi bir sonlu F_q cisminde başkatsayısı 1 olan $\phi(q^L - 1)/L$ adet ilkel polinom vardır. $\phi(\cdot)$ fonksiyonu 1'den büyük bir pozitif tamsayı için, o tamsayıdan küçük ve o tamsayı ile aralarında asal tamsayıların adedini veren Euler-ϕ fonksiyonudur ve $\phi(1) = 1$ olarak tanımlanır.

İndirgenemez fakat ilkel olmayan bir polinoma örnek olarak F_2 üzerinde $x^6 + x^4 + x^2 + x1$ polinomunu örnek verebiliriz.

3.1 Golomb Kuralları

Golomb(1967), periyodu p olan bir dizinin neredeyse rassal olabilmesi için bazı şartlar ileri sürdü. Buna göre periyot boyunca :

1) 0 ve 1 sayısı dengeli olmalıdır.

2) Bir elemeanlı öbeklerin (0 veya 1) herbirinin oranı yaklaşık ½, iki elemanlı öbeklerin (00,01,10 veya 11) herbirinin oranı yaklaşık ¼, üç elemanları öbeklerin (000,001,...,111) herbirinin oranı 1/8 olmalıdır.

3) Dizinin sıfır olmayan dizileri için otokorelasyon değeri sadece iki değer almalıdır. Diğer bir deyişle, otookrelasyon fonksiyonu $c(\tau) = \begin{cases} 1 & \tau = 0 \\ a & 0 < \tau < p \end{cases}$

Her terimi $A = \{a_1, \ldots, a_q\}$ alfabesinden seçilen $s^{\infty} = s_0, s_1, \ldots$ dizisi alalım. s^a ,dizinin a ncı terimine kadar olan altdizisini, s_i ise dizinin i ninci terimini ifade etsin.

TANIM: Periyodu N olan ve alfabesi q sembolden oluşan bir s^{∞} dizisi , eğer s^{N+m-1} alt dizisinde olası her m li aynı sayıda(neredeyse aynı sayıda) gözüküyorsa m *inci dereceden ideal(neredeyse ideal) yerel istatistik* sahibidir denir. Neredeyse aynı bir eksik ya da bir fazla anlamındadır.

Teorem: (İdeal ve neredeyse ideal yerel istatistiklerin iç içe geçme özelliği) Eğer bir s^{∞} dizisi m inci dereceden ideal(neredeyse ideal) yerel istatistiğe sahipse $l = 1,2,\ldots,m-1$ için l ninci dereceden ideal(neredeyse ideal) yerel istatistiğe sahiptir.

İspat: Tümevarım ile gösterilebilir.

Teorem: Uzunluğu L olan bir FSR çıktısı s^{∞} ancak ve ancak bütün FSR çıktıları tek bir devir üzerinde yer alıyorsa L inci dereceden ideal yerel istatistiğe sahiptir.

İspat: Uzunluğu L olan bir FSR çıktısının periyodu N değeri, olası bütün L lilerin adedi olan q^L den küçük bir sayı olduğundan önermenin iki yönü de sayma prensibi ile kolayca gösterilebilir.

Sonuç: L uzunluklu bir tekil LFSR'ın sıfır olmayan herhangi bir ilk durumdan başlayan (o ilk duruma ait) her çıktı dizisi ancak ve ancak bahis konusu olan LFSR bir maksimum uzunluklu bir LFSR ise L inci dereceden neredeyse ideal yerel istatistiklere sahiptir.

İspat: LFSR maksimum uzunluklu ise 0 olamayan her L li sadece bir kere gözükeceğinden ve 0_L hiç gözükmeyeceğinden çıktı dizisi neredeyse ideal yerel istatistiğe sahip olacaktır. Eğer LFSR maksimum uzunluklu değil ise çıktı dizisinin periyodunun düşüklüğü nedeniyle neredeyse ideal istatistiğe sahip olamaz.

Yukarıdaki sonuçtan maksimum uzunluktaki bir LFSR için 0,1 dengeliliği ve L uzunluklu öbeklerin periyot boyunca sadece birer kez gözükmelerinden dolayı Golomb kurallarının ikisine uymaktadır. Üçüncü özellik olan otokorelasyon için Kara(2001)'e bakınız.

3.2 Doğrusal Karmaşıklık

Tanım: Eğer bir s dizisi bir ilk durum için bir LFSR'ın çıktısına eşitse o *LFSR s dizisini üretiyor* denir. Benzer şekilde, bir ilk durum için bir LFSR çıktısının ilk n terimi sonlu bir s^n dizisi ile aynıysa, o LFSR s^n dizisini üretiyor denir.

Tanım: (İkili) s dizisinin doğrusal karmaşıklığı $L(s)$ aşağıdaki gibi tanımlanır:

Eğer $s = 0,0,0,\ldots$ sıfır dizisi ise $L(s) = 0$

Eğer hiçbir LFSR çıktısı s dizisine eşit değil ise $L(s) = \infty$

$L(s)$, s dizisini üreten en küçük LFSR'ın boyu.

Sonlu bir s^n dizisi için de benzer şekilde $L(s)$ tanımlanabilir. Yalnız sonlu durumda doğrusal karmaşıklığın sonsuz olmayacağı açıktır.

Örnek: $L(0,1,0,1,0,1,\ldots) = 2$

$L(1,0,1,0,0,1,0,0,0,1,\ldots,1) = \infty$

3.2.1 Doğrusal Karmaşıklığın Özellikleri:

1) s ve t ikili diziler olsun. Her $n \geq 1$ için s^n dizisinin doğrusal karmaşıklığı $0 \leq L(s) \leq n$ koşulunu sağlar.

2) $L(s^n) = 0$ ancak ve ancak $s^n = 0$.

3) $L(s^n) = n$ ancak ve ancak $s^n = 0,0,\ldots,0,1$.

4) $L(s \oplus t) \leq L(s) + L(t)$. $\oplus$ sembolü mod2 toplamını temsil etmektedir.

İspat: Son madde hariç diğer özelliklerin ispatı doğrudan gözükmektedir. Son madde için LFSR teoremini (Fatih 1998) kullanmak gerekir. İspatı sonradan kullanmayacağımız için burada yer vermeyeceğiz.

Teorem(Massey,1969): Eğer L uzunluklu bir LFSR <L,C(D)> s^n dizisini üretiyor fakat $s^{n+1} = s^n s_n$ dizisini üretmiyorsa $L(s^{n+1}) \geq n+1-L$ olur.

İspat(U. Maurer ve R. Viscardi, 1984): Hipoteze göre <L,C(D)> LFSR'ı $\tilde{s}_n \neq s_n$ olmak üzere $t^{n+1} = s^n \tilde{s}_n$ dizisini üretir. $\Delta = s_n - \tilde{s}_n \neq 0$ olsun. $s^{n+1} - t^{n+1} = 0^n \Delta$ olduğundan ve doğrusal karmaşıklığın dördüncü özelliğinden dolayı $L(s^{n+1} - t^{n+1}) = n+1 \leq L(s^{n+1}) + L(t^{n+1}) \leq L(s^{n+1}) + L$ bulunur. Eşitsizlikten $L(s^{n+1}) \geq n+1$-L sonucu elde edilmiş olur.

Teorem(Doğrusal Karmaşıklık Değişim teoremi): Eğer L uzunluklu bir LFSR <L,C(D)> s^n dizisini üretiyor fakat $s^{n+1} = s^n s_n$ dizisini üretmiyorsa o zaman s^{n+1} dizisinin doğrusal karmaşıklığı $L(s^{n+1}) = \max[L(s^n), n+1-L(s^n)]$ olur.

İspat: Teoremin ispatı çok uzun olmamakla birlikte öncesinde yardımcı teoremlerin ispatını ve ayrım teriminin tanımını gerektirdiğinden burada verilmeyecektir.

Berlekamp Massey algoritması sayesinde L uzunluklu bir LFSR'ın geribesleme polinomu bilinmese dahi LFSR çıktısından $2L$ uzunlukta bir dizi ile LFSR'ın geribesleme noktaları bulunabilir ve dolayısıyla tek LFSR "kırılmış" olur. Bu durum, tek bir LFSR çıktısının tek başına kullanılamayacağı anlamına gelir. Güvenliği artırmak için genelde birkaç LFSR çıktısı doğrusal olmayan(nonlineer) bir fonksiyon sayesinde *birleştirilir*. Ne var ki "birleşik çıktının" doğrusal karmaşıklığı yüksek olsa da çıktı ile LFSR'lardan biri ya da birkaçı arasında korelasyon bulunur. Bu ilişkiden faydalanmak temelindeki saldırılara *korelasyon saldırıları* adı verilir. Berlekamp Massey algoritmasının karmaşıklığının düşük olmasından dolayı genelde korelasyon saldırıları genelde *"böl-fethet"(divide and conquer)* saldırıları ile birleştirilerek LFSR'lardan biri "yalnız bırakılmaya" çalışılır.

Tanım: n değişkenli bir f booleen fonksiyonunu $GF(2)^n$'den $GF(2)$'ye bir fonksiyon olarak ele alalım. $f(x_1, x_2, \ldots, x_n)$ *fonksiyonun cebirsel normal formu*

$$f(x_1, x_2, \ldots, x_n) = a_0 \oplus \underbrace{a_1 x_1 \oplus \ldots \oplus a_n x_n}_{\substack{1nci\ dereceden \\ (dogrusal)\ terimler}} \oplus \underbrace{a_{1,2} x_1 x_2 \oplus \ldots \oplus a_{n-1,n} x_{n-1} x_n}_{2nci\ dereceden\ terimler} \oplus \ldots \oplus \underbrace{a_{1,2,\ldots,n} x_1 x_2 \ldots x_n}_{ninci\ dereceden\ terimler}$$

şeklinde yani $GF(2)$'de bir sabit ile değişkenlerin birbirinden farklı çarpımlarının toplamı

şeklinde ifade edilmesidir. Katsayısı sıfır olmayan terimlerin dereceleri içinde en büyüğü f fonksiyonunun *nonlineerlik derecesi* olur. Her booleen fonksiyonun cebirsel formu vardır ve bu cebirsel normal form tektir.

Teorem (İkili LFSR dizilerinin nonlineer kombinasyonu ile oluşturulan dizilerin doğrusal karmaşıklık değeri üzerine sınır)(Selmers ve Herlestam): n ayrı LFSR çıktısının $f(x_1,x_2,\ldots,x_n)$ booleen fonksiyonu ile birleştirildiğini varsayalım. Diğer bir deyişle, $s_i^{(j)}$ j ninci LFSR'ın i nci çıktı bitini, z_i sistemin i nci çıktı bitini temsil etmek üzere $z_i = f(s_i^{(1)},s_i^{(2)},\ldots,s_i^{(n)})$ olsun. LFSR'ların her birinin periyodunun tek olduğunu varsayalım. L_j , $j$nci LFSR'ın uzunluğunu belirtmek üzere çıktı dizisinin doğrusal karmaşıklığı $\mathrm{L}(z^\infty) \le \tilde{f}(L_1,L_2,\ldots,L_n)$ olur. $\tilde{f}$ fonksiyonu birleştirici f fonksiyonunun cebirsel normal formundaki modulo2 toplam ve çarpımının tamsayı toplam ve çarpımı ile değiştirilmesiyle elde edilir.

İspat: İspat ayrık Fourier dönüşümü ve Hadarmard çarpım özelliği gerektirdiğinden ve korelasyon saldırıları için gerekmeyeceğinden verilmemiştir.

Örnek : Geffe sistemi üç LFSR çıktısını $f(x_1,x_2,x_3) = x_3 \oplus x_1x_2 \oplus x_2x_3$ fonksiyonu ile birleştirmektedir.LFSR uzuluklarını $L_1 = 17, L_2 = 19, L_3 = 21$ alalım. Bu durumda $\mathrm{L}(z^\infty) \le L_3 + L_1L_2 + L_2L_3 = 743$ olur. Geffe üretecinde L_1, L_2 ve L_3 aralarında asal ise üretecin doğrusal karmaşıklığı $\mathrm{L}(z^\infty) = L_3 + L_1L_2 + L_2L_3$ olmaktadır. Dolayısıyla doğrusal karmaşıklık yüksektir. Üreteç çıktısını $z = f(x_1,x_2,x_3) = (1+x_2)x_3 + x_1x_3$ şeklinde ifade edebiliriz. Geffe üretecinin 0,1 dengeliliği gibi istatistiksel değerleri incelenirse istatiksel açıdan iyi çıkmaktadır çünkü iki maksimum uzunluktaki LFSR'ın çıktısından birini seçmektedir. Fakat LFSR_1 çıktısı üreteç çıktısına *sızmak*tadır çünkü çıktının LFSR_1 çıktısı olma olasılığı $\frac{1}{2}$'den farklıdır. Aynı durum LFSR_3 çıktısı için de geçerlidir.

$$\left.\begin{aligned} P(z = x_1 \mid s_2 = 1) &= 1 \\ P(z = x_1 \mid s_2 = 0) &= \frac{1}{2} \end{aligned}\right\} \Rightarrow P(z = x_1) = \frac{3}{4}.$$

Geffe üreteci kriptolojik açıdan çok zayıf bir sistemdir. Bu kriptosistemin kriptanalizi için Yang'ın makalesine bakılabilir. Simetrik anahtarlı dizi şifreleme sistemlerinin yaygın türlerinden biri alt üreteç olarak n LFSR kullanıp , bu LFSR'ları (LFSR_i , $i = 1,2,\ldots,n$) "iyi"

bir birleştirici bir f fonksiyonu kullanılır. f fonksiyonu ancak nonlineer fonksiyonlardan seçilebilir fakat bu koşul tek başına yeterli değildir. $LFSR_i$'nin ilk durumunu K_i ile gösterelim. Genelde LFSR tabanlı kriptosistemlerin anahtarları $K_1, K_2, \ldots, K_n$ olarak kabul edebiliriz. Kriptanaliz sırasında (genelde) anahtar dışında üreteç hakkındaki bilgilerin düşman kriptoanalist tarafından biliniyor kabul edildiğini hatırlayalım.(Kerkhoff prensibi) Sayma prensibi nedeniyle, üretecin mümkün olan anahtar sayısı M_i, $LFSR_i$ alt üretecinin mümkün olan anahtar sayısını temsil etmek üzere $M = \prod_{i=1}^{n} M_i$ olur. İdeal durumda, düşman kriptoanalistin anahtar uzayının ortalama olarak yarısını taraması sonucu sistemi kırması beklenir. Fakat eğer f fonksiyonu zayıf ise üreteç çıktıları ile LFSR çıktıları arasındaki korelasyondan dolayı $M' << M$ sayıda anahtar denenmesi yeterli olabilir. Örneğin eğer üreteç çıktısı $LFSR_1$ ile yüksek korelasyona sahipse ve diğer LFSR çıktıları ile (neredeyse) tamamen bağımsız ise M_1 denemede $LFSR_1$ için K_1 bulunur. Geriye diğer LFSR'ların ilk durumunu bulmak kalır. Diğer bir deyişle $M' \approx M_1 + \prod_{i=2}^{n} M_i = O(\prod_{i=2}^{n} M_i)$ iş yeterli olur. Fakat bazen f fonksiyonu birden fazla LFSR ile ilişkili olduğundan, örneğin Geffe üreteci LFSR'dan ikisi ile yüksek korelasyona sahiptir, yapılacak iş çok daha düşük olur.

LFSR alt üreteçleri ile üreteç arsında her zaman iyi bir korelasyon bulunmasa da alt üreteçlerin birkaçının kombinasyonu arasında istatiksel bir bağımlılık bulunabilir.(mesela x_{a_i}, $LFSR_{a_i}$ çıktısını ifade etmek üzere $\bigoplus_{a_i=1}^{j} x_{a_i}$, $j << n$ ile üreteç çıktısı arasında korelasyon olabilir.) Bu durumda, yukarıda M_1 yerine $O(\prod_{a_i=1}^{j} M_{a_i})$ ön iş yapılması gerekir. Korelasyon saldırılarına karşı sitemi güvenli yapmak için üreteç çıktısının bu tür istatistiksel bağıntılardan yoksun olmalıdır.

4. KLASİK HIZLI KORELASYON SALDIRILAR

4.1 Siegenthaler Saldırısı

n LFSR'lı bir kriptosistemimiz olduğunu düşünelim. LFSR'ların ilk durumlarının yanında bağlantı polinomlarının da bilinmediğini varsayalım. $i = 0,1,\ldots,n$ için $LFSR_i$'nin bağlantı polinomunun ilkel olduğunu (diğer bir deyişle LFSR'ların maksimum uzunluklu olduğunu) ve LFSR boylarının r_i olduğunu varsayalım. r_i uzunluktaki LFSR için R_i farklı ilkel polinom olsun. Sıfır ilk durumu kabul edilmeyeceğinden üreteç için mümkün anahtar sayısı $\prod_{i=1}^{n} R_i(2^{r_i}-1)$ olur. Fakat LFSR'lar ile üreteç çıktısı arasında yüksek korelasyon varsa taranması gereken anahtar sayısı yaklaşık $\sum_{i=1}^{n} R_i 2^{r_i}$ mertebesine düşürülebilir. Üreteçleri birleştiren fonksiyonun bilindiğini kabul edelim.

f fonksiyonuna giren $LFSR_i$ çıktısı X_t^i'lerin bağımsız özdeş dağılımlı tesadüfü değişken olduğunu varsayalım. t anındaki üreteç çıktısını $Z_t = f(X_t^1, X_t^2, \ldots, X_t^n)$ ile ifade edelim. Z_t dağılımının düzgün olduğunu varsayalım. (Aksi takdirde birleştiren fonksiyon girdilerinden bağımsız olarak istatiksel bir saldırı düzenlenebilir.) $P_z(Z_t = 0) = P_z(Z_t = 1)$ olduğunu varsayalım. LFSR çıktıları ile üreteç çıktıları arasındaki korelasyon da q_i olsun, yani $P(Z_t = X_t^i) = q_i$ olsun. Açıkmesajın da $P(Y_t = 0) = p_0$ istatistiğine sahip olduğunu varsayalım. Hayali bir X_t^0 tesadüfi değişkeni tanımlayalım, öyle ki $P(X_t^0 = 0) = P(X_t^0 = 1) = \frac{1}{2}$ olsun. Şifreli mesaj C_t ile X_t^j arasındaki korelasyon ölçüsü olarak $\alpha_j = \sum_{t=1}^{N} (-1)^{(C_n \oplus X_t^j)}$, $j \in \{0,1,\ldots,n\}$ alalım. $P(C_t \oplus X_t^j = 0)$ olasılığı p_e aşağıdaki gibi hesaplanabilir:

$$p_e = P(C_t \oplus X_t^j = 0) = P(Z_t = X_t^j)P(Y_t = 0) + P(Z_t \neq X_t^j)P(Y_t = 1)$$

$$= \quad q_j \quad \cdot \quad p_0 \quad + \quad (1-q_j)\cdot(1-p_0)$$

$$= \quad 1-(p_0+q_j)+2p_0q_j \qquad (4.1)$$

Şifreli mesaj ile LFSR çıktısının farklı olma olasılığı $P(C_t \oplus X_t^j = 1) = 1 - p_e$ olur. α_j değerini veren toplamda sabit bir $j \in \{0,1,\ldots,n\}$ için $C_t \oplus X_t^j$ terimlerinin her biri benzer dağılımlı bağımsız değişkenler olduğundan ve $\beta = \sum_{t=1}^{N} (C_t \oplus X_t^j)$ tesadüfi değişkeni normal dağılıma sahip olduğundan ortalama değeri m_β ve varyansı σ_β^2 değerleri sırasıyla

$$m_\beta = N(1 - p_e) \tag{4.2}$$

$$\sigma_\beta^2 = Np_e(1 - p_e) \tag{4.3}$$

olur. α_j değişkenine ait ortalama değer ve varyans ise

$$m_{\alpha_j} = N - 2N(1 - p_e) = N(2p_e - 1) \tag{4.4}$$

$$\sigma_{\alpha_j}^2 = \text{var}(N - 2\beta) = 2^2 \sigma_\beta^2 \tag{4.5}$$

olur. β değişkeninin varyansı son eşitlikte yerine konulursa $\sigma_{\alpha_j}^2 = 4Np_e(1 - p_e)$ bulunur. Z_t ve X_t^0 değişkenlerinin bağımsız olmaları nedeniyle $j = 0$ için $q_0 = 1/2$ ve $p_e = 1/2$ buluruz. α_0 için ortalama değer ve varyans ise sırasıyla 0 ve N olur.

Yeterince büyük N için, α_j tesadüfi değişkeni, merkezi limit teoremi nedeniyle m_{α_j} ve $\sigma_{\alpha_j}^2$ parametreleri olan normal dağılıma sahip olarak düşünülebilir. Bir atak sırasında α_j için α_j^0 gerçek değeri N şifreli mesaj biti ve r_i uzunluklu R_i tesadüfi bağlantı polinom kümesinden alınan tesadüfi bir ilk durumlu LFSR tarafından üretilen N bit tarafından belirlenebilir. Bir karara varabilmek için aşağıdaki istatiksel hipotezler kullanılabilir.

H_1 : r_i uzunluklu LFSR'ın $N > r_i$ biti $LFSR_i$ tarafından üretilen N bit ile çakışır. Bu durumda α_i, C_t ile $i \in \{0,1,\ldots,n\}$ için X_t^i arasındaki korelasyondur.

H_0 : r_i uzunluklu LFSR'ın $N > r_i$ biti $LFSR_i$ tarafından üretilen N bit ile çakışmaz. Bu durumda α_i, C_t ile X_t^0 arasındaki korelasyondur.

İstatistiksel hipotez testlerinden alışık olunduğu gibi, iki hipotez arasında bir *karar eşik değeri* T seçmek gerekir, öyle ki $\alpha_i^0 < T$ durumunda H_0 hipotezi, $\alpha_i^0 \geq T$ durumunda H_1 hipotezi kabul edilsin. $p_e = 1/2$, daha doğrusu $q_i = 1/2$ ve(veya) $p_0 = 1/2$, durumunda $p_{\alpha_i|H_0(x)}$ ve $p_{\alpha_i|H_1(x)}$ olasılık yoğunluk fonksiyonları aynı olacaktır, dolayısıyla herhangi bir karar verilemez. T eşik değeri öyle seçilmelidir ki P_f ve P_m değerleri mümkün olduğunca küçük olsun. İşlemsel karmaşıklık daha çok yanlış karar sayısına yani T eşik değerini aşan α_i^0 değeri sayısına dayanacağından ilk önce $P(\alpha_i \geq T | H_0)$ olasılığına bakmak gerekir. Fakat eşik değerinin hesaplanması için $P(\alpha_i < T | H_1)$ olayı kaçırma olasılığı P_f de göz önüne alınmalıdır. İki olasılık değeri $P_f = \int_T^{\infty} p_{\alpha_i|H_0(x)} dx$ ve $P_m = \int_{-\infty}^{t} p_{\alpha_i|H_1(x)} dx$ formülleriyle hesaplanır. $m_0 = 0$ $m_1 = N(2p_e - 1)$ ve $\sigma_0 = \sqrt{N}$ $\sigma_1 = 2\sqrt{N}\sqrt{p_e(1-p_e)}$ olduğundan dolayı $P_{\alpha|H_k(x)} = \frac{1}{\sqrt{2\pi}} e^{\frac{(x-m_k)^2}{2\sigma_k^2}}$ olur. Normal dağılımda $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-\frac{y^2}{2}} dy$ integral değeri hesaplandığından $P_f = Q\left(\left|\frac{T}{\sqrt{N}}\right|\right)$ ve $P_m = Q\left(\left|\frac{N(2p_e-1)-T}{2\sqrt{N}\sqrt{p_e(1-p_e)}}\right|\right)$ olur. T eşik değeri yerine kolaylık için $\gamma_0 = \frac{N(2p_e-1)-T}{2\sqrt{N}\sqrt{p_e(1-p_e)}}$ ve $\frac{T}{\sqrt{N}} = \sqrt{N}(2p_e-1) - 2\gamma_0\sqrt{p_e(1-p_e)}$ olduğundan P_m ve P_f değerleri $P_m = Q(|\gamma_0|)$ ve $P_f = Q\left(\left|\sqrt{N}(2p_e-1) - 2\gamma_0\sqrt{p_e(1-p_e)}\right|\right)$ değerlerini alır.

LFSR tabanlı bir üretece saldırmak için önce çıktı fonksiyonu ile en yüksek korelasyona sahip değişkenden başlayabiliriz. p_0 olasılığı açık metinin dil ve kodlama istatistiğinden bulunabilir. p_e olasılığı , $p_e = 1 - (p_0 + q_j) + 2p_0q_j$ eşitliğinden hesaplanabilir. Seçilen bir P_m değeri için γ_0 değeri sabittir ve yanlış alarm olasılığı kullanılan N şifreli metin bitinin fonksiyonu olarak hesaplanabilir.

Anahtarın $LFSR_i$ kısmını bulabilmek için (başta bağlantı polinomunu bilmediğimizi varsaydığımızdan) mümkün olan bütün R_i adet ilkel bağlantı polinomunu kullanarak $2^{r_i} - 1$ periyodunda maksimum uzunluklu diziyi oluştururuz. Diğer LFSR dizilerinin bilinmesi zorunluluğu yoktur. Çıktı dizisi ile LFSR dizisi arasındaki korelasyondan faydalanarak LFSR

ilk durumunu tekrar oluşturabiliriz. $2^{r_i}-1$ ilk durumun diğer bütün olası N bit uzunluklu dizileri için α_i korelasyonu hesaplanır. Daha sonra bütün x_i dizileri için aynı yöntem tekrarlanır. Her $\alpha_i \geq T$ için doğru bağlantı polinomu ve doğru pozisyon kullanıldığı kabul edildiğinden anahtarın $LFSR_i$ kısmının doğru olarak bilindiği kabul edilecektir. $\alpha_i \geq T | H_0$ olayı P_f olasılığı ile ortaya çıktığından verilen karar doğru olmayabilir. Bu yüzden, $\alpha_i \geq T$ durumlarda yeni şifreli metin parçaları ile ilave denemeler yapılmalıdır.

Eğer H_1 hipotezi bütün bağlantı polinomları için reddedilirse yeni bir bağlantı polinomu seçilerek aynı işlemler yapılır. En kötü durumda R_i bağlantı polinomu için $2^{r_i}-1$ olası ilk durumun hepsi deneneceğinden anahtarın $LFSR_i$ kısmını bulabilmek için yaklaşık $R_i 2^{r_i}$ deney yapılması gerekir. Yanlış alarmların sayısı, dolayısıyla yapılacak deneme sayısı seçilen N şifreli metin dizisi bitine dayanmaktadır. Eğer N_1 sayısını yanlış alarm sayısının beklenen değeri 1 olacak şekilde seçersek yaklaşık $R_i 2^{r_i}$ toplam anahtar deneme sayısında $LFSR_i$ ilk durumu oluşturulmuş olur. P_f için "sadeleştirilmiş" normal dağılım değeri kullanılır ise $\frac{1}{R_i 2^{r_i}} = Q\left(\left|\sqrt{N_1}(2p_e-1)-2\gamma_0\sqrt{p_e(1-p_e)}\right|\right)$ eşitliği bulunur. Hassas normal dağılım tabloları kullanılarak N_1 değeri $N_1 < \left[\frac{\frac{1}{\sqrt{2}}\sqrt{\ln\left(R_i 2^{r_i-1}\right)}+\gamma_0\sqrt{p_e(1-p_e)}}{p_e-\frac{1}{2}}\right]$ şeklinde sınırlanabilir. (Birinci 1998). Hesaplanan bu N_1 değeri saldırı için gerekli olan şifreli metin dizisini tahmin etmek için kullanılabilir.

Siegenthaler'in saldırısı ile ilgili birçok deneyler yapılmıştır.

$$f(x_1,x_2,\cdots,x_n)=\begin{cases}1 & \text{Eğer } (x_1,x_2,\ldots,x_n) > n/2 \text{ ise} \\ 0 & \text{Diğer durumlarda}\end{cases}$$

şeklinde tanımlanan Bruer üretecine yapılan saldırı sırasında yanlış ilk durum veya yanlış bağlantı polinomu seçildiğinde ve şifreli metin dizisi yeterince uzun ise, $i=0,1,\ldots,n$ için hesaplanan α_i^0 değerleri sıfıra yakın çıkmış, bazı ilk durum veya bazı bağlantı polinomları için ise sıfırdan büyük olmasına rağmen beklenen değerlerden çok düşük çıktığı gözlemlenmiştir. Diğer bir deyişle, yeterince uzun bir şifreli metin dizisi alındığında doğru ilk durum veya doğru bağlantı polinomu ile yanlış ilk durum veya yanlış bağlantı polinomları

arasındaki fark belli olmaktadır. Şifreli metin yeterince uzun seçildiğinde beklenen değere yakın sonuçlar veren bir ilk durum ve bağlantı polinomunda saldırıyı durdurabileceğimiz gözlenmiştir. Şifreli metin dizisinin yeterince uzun olmadığı durumlarda α^0 ların dağılımı tam olarak normal dağılıma uymamakta olduğundan yanlış karar verme ihtimali doğmaktadır. Siegenthaler'in atağı bazı LFSR'lar için LFSR'ın tamamının(ilk durumların hepsi ve taranmasını gerektirdiğinden yüksek LFSR boyu için etkili olamamaktadır. Siegenthaler'in saldırısı, pratikte çok uygulanabilir olmasa da LFSR tabanlı üreteçlerde düşük LFSR boyunun yol açabileceği zayıflıkları göstermesi açısından LFSR tabanlı şifreleme sistemlerinin tasarımı ve daha sonra bulunan saldırı türlerine esin kaynağı olması açısından bir dönüm noktası olmuştur. Siegenthaler saldırısı, boyu 60'dan daha yüksek LFSR'lara pek uygulanamamaktadır.

4.2 Staffelbach –Meier Saldırısı

Staffelbach-Meier ise çok daha uzun LFSR boylarında çalışabilen fakat geribesleme noktalarının sayısı az olan sitemler için saldırılar düzenlemiştir. Geribeslemenin az olması yazılım veya donanım kolaylığı nedeniyle tercih edilebilir. Uygun koşullarda saldırılabilecek LFSR'ın boyu 100'ü bile aşabilmektedir.

Saldırıya geçmeden önce çıktı dizisi z 'den N bit verildiğini ve t geribeslemeli noktalı LFSR dizisi a ile $p > 0.5$ 'den daha yüksek olasılıkla korelasyon olduğunu varsayalım. Bağlantı polinomunun bilindiği varsayılmaktadır fakat düşük geribesleme sayısı olan ilkel polinomların sayısı çok olmadığından bu kabul gerekli değildir. z çıktı dizisi LFSR dizisi a 'nın ikili asimetrik hafızasız rassal kaynak tarafından $P(0) = p$ olasılıkla bozulmuş hali olarak bakılabilir. LFSR dizisinin tekrar inşa edilebilmesi için algoritmada şu koşul gereklidir: a dizisinin her biti a_n temel doğrusal bağıntıdan türetilen ve her biri a nın t farklı bitini içeren birkaç doğrusal bağıntıyı sağlamalıdır. Bu bağıntılara karşılık gelen z bitlerini yerine koyarak her bir z_n biti için , kimisi tutmayan, denklemler elde edilir. z_n dizisi ile a_n dizisinin eşitliğini test etmek için, z_n için geçerli olan denklemlerin sayısı kaydedilir. Doğru olan denklemlerin sayısı ne kadar çoksa, z_n ile a_n 'in eşit olma olasılığı o kadar artar. Bu önerme istatistiksel bir model ile doğrulanmaktadır. BU düşünce ile iki farklı saldırı uygulanmaktadır.

Saldırılardan ilkinde $z_n = a_n$ olan bitler aranmaktadır. Bu bitler, en fazla sayıda denklem sağlayanları seçerek bulunur. Bu yolla LFSR dizisi a nın karşılık gelen bit dizileri tahmin edilmiş olur. "Uygun koşullar altında" bu elde edilen bitlerin doğru olma olasılığı epey yüksektir ve yapılan tahmin için sadece ufak bir değişiklik yeterli olacaktır. Bu da bize LFSR dizisini doğrusal denklemlerle bulmak amacıyla yapacağımız bütün LFSR olası anahtar uzayını taramadan önce oldukça küçültülmüş bir anahtar uzayı verecektir.

İkinci saldırıda ise en güvenilir bitler yerine doğru olma olasılıkları ile birlikte z dizisinin bütün bitleri göz önüne alınacaktır. Önce kabul edelim ki, a dizisi ile z dizisinin karşılık gelen bitlerinin uyuşma olasılığı p olsun. Sonra, z 'in her biti için $z_n = a_n$ olasılığına eşit olan ve sağlanan denklem sayılarıyla koşullandırmak şartıyla yeni bir p^* olasılığı tanımlayalım. Algoritmanın her yinelenmesinde girdi olarak yenilenen p^* olasılıkları ile işlemleri devam ettirelim. (Saldırı) algoritmasının birkaç yinelenmesinden sonra p^* olasılıkları belli bir eşik değerinin altına inen z bitleri tersine çevrilebilir. "Uygun koşullar

altında" yanlış bitlerin sayısının azalması beklenir. z dizisi yerine elde edilen yeni dizi kullanılarak bütün süreç tekrarlanırsa LFSR dizisi a elde edilmiş olur.

Birinci ve ikinci saldırılar boyu 100'den daha büyük LFSR'lara uygulanabilir. İki algoritma arasında yapılan kıyaslamalar birinci saldırının $c << 1$ ve $p \approx 0.75$ civarında iken ikinci saldırıya oranla daha iyi sonuçlar verdiğini gösterirken, ikinci saldırı çok daha düşük korelasyonlarda ($p \approx 0.5$ iken) etkili olduğunu göstermektedir.

Bu iki saldırının en önemli sonuçlarından birisi LFSR tabanlı şifreleme sistemi tasarımında, çıktı dizisinin geribesleme noktası 10'dan az olan herhangi bir LFSR'la korelasyonu olmaması ve uzunluğu 100'den daha kısa (genel) bir LFSR ile korelasyonu olmaması gibi ölçütlerin konulmasına yol açmıştır.

Üreteç çıktısı z ile doğrusal bağlantısı $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_L a_{n-L}$ şeklinde verilen LFSR dizisi a ile $p = P(z_n = a_n) > 0.5$ korelasyonu olsun. Bağıntının bağlantı polinomu $C(D) = 1 + c_1 D + c_2 D^2 + \ldots + c_L D^L$ olacağı açıktır. Geribesleme noktalarının sayısı t, $\{c_1, c_2, \ldots, c_L\}$ arasındaki sıfır olmayan terimlerin sayısına eşittir. Yukarıdaki denklem $t+1$ terimli $\sum_{\substack{0 \le i \le L \\ c_i \neq 0}} a_{j-i} = 0$ şeklinde yazılabilir. a dizisi kaydırılarak sabit bir a_n biti $t+1$ denklem sağlar. $C(D)$ polinomunun her (polinomsal) katı $f(D)$ de a dizisi için bir doğrusal bağıntı tanımlar. Özel olarak, $j = 2^i$ için $C(D^j) = C(D)^j$ olur. (F_2 cisminde çalışıldığından) Bu yöntemle, geribesleme bağıntısının kaydırılması dışında her birinde t geribesleme noktası olan farklı denklemler elde ederiz. Verilen bir n için $a_n = z_n$ olup olmadığına çıktı dizisi z 'in bu doğrusal denklemlerin sağlayıp sağlamadığına bakarak karar verilir.

a_n sabit olsun. b_i lerin her biri LFSR dizisi a dan tam olarak t bitin toplamı olmak üzere, (diğer bir deyişle $b_i = \sum_{i=1}^{t} a_{i_j}$), $a_n + b_i = 0$ şeklinde denklemleri sağlar. Bu denklemlerin saldırılarda kullanılışı ve uygun koşullar için Birinci(1998)'e bakınız.

5. KORELASYON SALDIRILARI ve KODLAMA TEORİSİ

5.1 Kodlama Teorisine Giriş

Veri iletimi esnasında veri, *gürültü(noise)* adını verdiğimiz bazı hatalar yüzünden bozulur. Orijinal veriyi elde etmek için uygulanan yöntemlerden biri de veriye birtakım bilgiler ekleyerek veri ile muhtemel hatalı veri arasındaki uzaklığı açmaktır. Veriyi temsil eden sözcüklere *kod sözcüğü(codeword)* denir. Kodlama teorisi ile ilgili bazı tanımlar aşağıda verilmektedir:

Tanım: A bir *alfabe* olsun. Harfleri A'dan olan n uzunluklu kelimelerin kümesi A^n ile gösterilir.

Tanım: q harften oluşan A alfabesi üzerinde (n,k) parametreli C *blok kodu* A^n'nin tam olarak q^k kod sözcüğünden oluşan bir alt kümesidir. n parametresine *kodun uzunluğu* denir.

Tanım: C blok kodu verilmiş olsun. A^k'dan C'ye giden ve k uzunluklu x sözcüklerini C'deki u kod sözcüklerine taşıyan $E(x)=u$ şeklinde tanımlanan fonksiyona C kodunun *kodlama* fonksiyonu denir.

Kodlama fonksiyonu her sözcüğü farklı bir kod sözcüğüne götürmelidir, diğer bir deyişle kodlama fonksiyonu birebirdir.

Tanım: Yukarıdaki parametrelerle tanımlanmış C kodu için $R(C)=\frac{k}{n}$ oranına C kodunun *bilgi oranı* denir.

Örnek: $A=\{0,1,\ldots,k-1\}$ olsun. Her sembolü n defa tekrar ettiren koda tekrar kodu denir.

Örnek: Üreteç matrisi ileride verilecek olan (n,k) = (7,4) Hamming kodu. Bu kod Hamming kodları adı verilen bir kod ailesinin en küçük elemanıdır.

Tanım: $a=(a_1\ldots a_n)$ ve $b=(b_1\ldots b_n)$ iki sözcük olsun. $d_H(a,b)=\#\{i|i=1,\ldots,n \quad a_i\neq b_i\}$ şeklinde tanımlanan ve a ile b sözcüklerinin farklı olan sembollerini sayan fonksiyona *Hamming uzaklığı* denir. $w_H(a)=\#\{i|i=1,\ldots,n \quad a_i\neq 0\}=d_H(a,0)$ şeklinde tanımlanan ve verilen a sözcüğündeki sıfır olmayan sembolleri sayan fonksiyona *Hamming ağırlığı* denir. Eğer başka bir fonksiyonla karıştırılmayacaksa sadece d ile gösterilir.

Önerme: n uzunluklu sözcükler kümesi üzerinde Hamming uzaklığı bir metriktir.

İspat: Metrik tanımındaki özellikler doğrudan kontrol edilir.

Tanım: A alfabesi q elemanlı cisim olsun. A üzerinde (n,k) parametreli C *lineer blok* **kodu** A^n vektör uzayının (A üzerinde k boyutlu) bir altuzayıdır.

Blok kodun tanımında alfabenin cisim olmak zorunda olmadığı gözden kaçırılmamalıdır. Lineer blok kodlarda iki kod sözcüğünün toplamı, cisimden bir skalerle çarpımı da yine bir kod sözcüğü oluşturur. Yine bütün bileşenleri 0 olan 0_n sözcüğü de her zaman n uzunluklu kodlara ait bir kod sözcüğüdür.

Tanım: Verilen bir (n,k) parametreli C kodu için $d_C=\min\{d(a,b)|a,b\in C, \quad a\neq b\}$ şeklinde tanımlanan tamsayıya *kodun minimum uzaklığı* adı verilir.

Önerme: Lineer kod için $d_C=\min\{w(a)|a\in C, \quad a\neq 0\}$.

İspat: $a,b \quad (a\neq b)$ kod sözcükleri için $a-b$ ve 0 kod sözcükleri de lineer koda ait olduğundan $d(a,b)=d(a-b,0)$ eşitliği geçerli olur çünkü a ile b nin farklı olduğu indislerde $a-b$ ile 0 da farklıdır. Uzaklığın tanımından ve $a-b$ lerin bütün kod sözcüklerini taramasından dolayı eşitlik gösterilmiş olur.

C kodunun d minimum uzaklığı biliniyorsa (n,k,d) parametreleri ile belirtilir. Verilen herhangi bir kodun minimum uzaklığını tespit etmek kolay değildir. Hatta verilen n ve k değerleri için olabilecek en büyük d değerini hesaplamak da kolay değildir. Kodlama teorisinin en büyük uğraş alanlarından biri de d değeri için sınırlar bulmaktır. Bunlardan en bilineni

$$k+d \le n+1 \tag{5.1}$$

temel eşitsizliğidir.

Tanım: $\{0,1\}$ sembollerinin $\{0,1\}$ sembollerine gönderildiği bir kanal düşünelim. Sembolden bağımsız olarak karşı tarafa hatalı gitme olasılığı aynı olsun. (Diğer bir deyişle, gönderilen sembol u ve alınan sembol z olsun. $P(z=0|u=1)=P(z=1|u=0)=p$ olsun.) Böyle bir kanala *ikili simetrik kanal* adı verilir. $1-p$ olasılığına kanalın *korelasyon* olasılığı adı verilir. İngilizce'si *binary ssymmetric channel* olduğundan BSC ile gösterilir.

5.1.1 Hata Düzeltilmesi

BSC üzerinden u mesajına karşılık gelen v kod sözcüğünün yollandığını varsayalım.($E(u)=v$). Kanal çıktısı $r=r_0,r_1,\ldots,r_{n-1}$ sembolleri dizisine *çıktı dizisi* adı verilir. Gürültü ile bozulmuş olması muhtemel r çıktısını *dekoder* $\hat{u}=\hat{u}_0,\hat{u}_1,\ldots,\hat{u}_k$ *tahmini mesaj*a dönüştürür. İdeal olanı $\hat{u}=u$ olmasıdır fakat gürültü yüzünden birtakım dekodlama hatası yapılmış olabilir. u mesajları ile v kod sözcükleri arasında birebir eşleme olduğundan dolayı dekoder çıktısını $\hat{u}$ 'ye karşılık gelen $\hat{v}=E(\hat{u})$ kod sözcüğü olarak düşünebiliriz çünkü dekodlama sırasında hata yapılması ancak ve ancak $\hat{v}\neq v$ ise gerçekleşir. Bu durumun geçekleşmesine yani $\hat{v}\neq v$ olması olasılığına *blok hata olasılığı* denir ve P_E ile gösterilir. Kanal çıktısının r olma olasılığı dekodlama kuralından bağımsız olduğundan şu eşitlik bulunur: $P_E=\sum_r P(\hat{v}\neq v|r)P(r)$.

P_E olasılığını en aza indirmek için bütün r ler için dekoder $P(\hat{v} \neq v \mid r)$ olasılığını en aza indirecek, diğer bir deyişle $P(\hat{v} = v \mid r)$ olasılığını en çok yapacak şekilde tanımlanmalıdır. Bu yüzden r alındığında olması en muhtemel kod sözcüğü $\hat{v}$ alınır. Kanala v kod sözcüğü girdiğinde kanaldan r çıkma $P(r \mid v)$ olasılığını en büyük yapacak şekilde $\hat{v}$ seçecek şekle dekodlamaya *maksimum olabilirlik* dekodlaması denilir.

BSC kanalı için $P(r \mid v)$ olasılığını en büyük yapmak, kanal çıktısı r ile kod sözcüğü arasındaki $d(r,v)$ uzaklığını en az yapmak ile eşdeğerdir. Bunu görmek için bir BSC alalım. Hata olasılığını $p < 1/2$ kabul edelim. Alınan kanal çıktısı r ile olası bir kod sözcüğü v_0 arasındaki Hamming uzaklığı d_{v_0} olsun. Her sembol için hata ihtimali aynı olacağından

$$P(r \mid v) = p^{d_{v_0}} (1-p)^{n-d_{v_0}} = (1-p)^n \left(\frac{p}{1-p} \right)^{d_{v_0}} \tag{5.2}$$

olur. $0 < \frac{p}{1-p} < 1$ olduğundan dolayı $P(r \mid v)$ olasılığını en büyük yapmak $d(r,v)$ uzaklığını en küçük yapmakla eşdeğerdir. Bu sonuçtan da görüldüğü gibi maksimum olabilirlik dekodlaması için kanal çıktısı r ye en yakın kod sözcüğü seçilmelidir.

5.1.2 Üreteç Matrisi

Bir lineer blok kod C, F_q^k'dan F_q^n'ye bir lineer dönüşüm olduğundan F_q^k'nın herhangi bir bazı için dönüşümü $k \times n$ boyutlu bir G matrisiyle ifade edebiliriz. Bu matrise C kodunun *üreteç matrisi* denir. Kodlama işlemi c kod sözcüğünü, i bilgi sözcüğünü (iletilecek veriyi) temsil etmek üzere

$$c = iG \tag{5.3}$$

şeklinde ifade edilebilir. Bu ifade biçimiyle, bilgi sözcükleri ile kod sözcükleri arasındaki eşlendirme baz vektörleri seçimine bağlı olur fakat toplam kod sözcükleri kümesi etkilenmez.

Örnek: (7,4) Hamming kodu için üreteç matrisi $G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$

C kodu bir vektör uzayının bir altkümesi olduğundan C'deki bütün vektörlere dik olan vektörlerin oluşturduğu $C^{\perp}$ dik tamlayan vektör uzayı vardır. $C^{\perp}$ de kendi başına boyutu $n-k$ olan bir kod olarak görülebilir. Bu koda C kodunun *dual kodu* adı verilir. H aynı baz altında $C^{\perp}$ kodunun üreteç matrisi olsun. F_q^n'nin bir c vektörü, ancak ve ancak H matrisinin her satırına dik ise C kodunun bir kod sözcüğüdür. Diğer bir deyişle, c vektörü ancak ve ancak $cH^T = 0$ ise C kodunun bir sözcüğüdür. Bu metot kod sözcüklerini tanıyabilmek için bir yol daha sağlamaktadır. H matrisine *parite kontrol matrisi* adı verilir. H matrisinin boyutlarının $(n-k) \times n$ olduğu açıktır.

Örnek: (7,4) Hamming kodu için parite kontrol matrisi $H = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$

5.2 Konvolüsyonel(Katlamalı) Kodlar

İletilecek veriyi k sembol uzunluğunda bloklara ayırıp her bloğu n sembol uzunluğunda kod sözcüklerine dönüştürerek gürültüden koruma yerine, çok daha küçük uzunlukta bloklar fakat ek bellek kullanarak oluşacak kod sözcüğünün kendisinden önce gelen m bloğa bağlı olmasını sağlama metodu da uygulanabilir. Bu şekilde inşa edilen kodlara *ağaç kodları* adı verilir. En önemli ağaç kodları türü *katlamalı kod* ailesidir.

5.2.1 Geribeslemesiz Katlamalı Kodlama

$k \le c$ olmak üzere $u = u_0 u_1 \cdots = \underbrace{u_0^1 u_0^2 \cdots u_0^k}_{u_0} \underbrace{u_1^1 u_1^2 \cdots u_1^k}_{u_1} \cdots$ iletilecek bilgi dizisini ve

$v = v_0 v_1 \cdots = \underbrace{v_0^1 v_0^2 \cdots v_0^c}_{v_0} \underbrace{v_1^1 v_1^2 \cdots v_1^c}_{v_1} \cdots$ geribeslemesiz katlamalı kod sözcüğü dizisini temsil etsin. t zamanında oluşacak katlamalı kod sözcüğü, f fonksiyonu $F_2^{(m+1)k}$ dan F_2^c ye bir lineer dönüşüm olmak üzere

$$v_t = f(u_t, u_{t-1}, \ldots, u_{t-m}) \tag{5.4}$$

şeklinde hesaplanır. Lineer blok kodlar için yaptığımız gibi katlamalı kodlar için matris formunda ifade etmek istersek

$$v_t = u_t G_0 + u_{t-1} G_1 + \cdots + u_{t-m} G_m \tag{5.5}$$

ifadesini elde ederiz. $0 \le i \le m$ için G_i matrisleri $k \times c$ ikili matrisleridir.

$$G = \begin{bmatrix} G_0 & G_1 & G_2 & \cdots & G_m & & \\ & G_0 & G_1 & \cdots & G_{m-1} & G_m & \\ & & G_0 & \cdots & G_{m-2} & G_{m-1} & G_m \\ & & & \ddots & \cdots & \cdots & \cdots & \ddots \end{bmatrix}, \quad v = v_0 v_1 \cdots \text{ ve } u = u_0 u_1 \cdots \text{ olmak üzere}$$

katlamalı kod $v = uG$ şeklinde ifade edilebilir.

Tanım: G matrisine *üreteç matrisi*, G_i matrislerine *üreteç alt matrisi* adı verilir.

$R = k/c$ oranına katlamalı kodun *bilgi oranı* adı verilir.

Örnekler ve grafiksel gösterimler için Arslan(2001)'e bakınız.

5.2.2 Ağaç gösterimi, trellis gösterimi, durum gösterimi

Bir katlamalı kod, bir ağaç diyagramı aracılığıyla gösterilebilir. Her düğümden çıkan iki dal bulunmaktadır. Üst dal bir 0 girdi bitine, alt dal ise 1 girdi bitine karşılık gelmektedir. Daha çok girdi biti alındıkça ağaç soldan sağa büyür. Her bir zaman biriminde kodlayıcının durumu her dalın altında gözükmektedir.

İlk üç daldan sonra yapı kendini tekrar etmeye başlamaktadır. Bunun sebebi kodlayıcı incelendiğinde açıkça ortaya çıkmaktadır. Kodlayıcıya dördüncü girdi biti girdiğinde, ilk girdi biti verisi kaybolur ve çıktı kod sembollerine daha fazla etkisi olmaz. Dolayısıyla 100xy... ve 000xy... üçüncü daldan sonra aynı kod sembollerini üretmektedir. Bu durum ağaç diyagramının aşağıdaki gibi çizilmesine yol açar.

Bu şekil gösterime trellis gösterimi denir. Bir trellis bir kodu zaman indeksli olarak gösterimine yarar.

Tanım: Derinliği n olan bir T=(V,E) *trellis*i, sonlu, yönlü, kenarları etiketli ve aşağıdaki özellikleri sağlayan bir çizgedir.

1. Her v köşesine karşılık bir $d(v) \in \{0,1,\ldots,n\}$ derinliği karşılık gelir.
2. Her e kenarı derinliği i olan bir köşeyi derinliği $i+1$ olan bir köşeye bağlar.
3. 0 derinliğinde *kök* denilen bir köşe ve n derinliğinde *dip* denilen bir köşe vardır. (Bu terimlerin İngilizce karşılıkları *root* ve bu kelimenin tersten yazılışı olan *toor* dur.)

Bir trellis diyagramı, bir kod için kod sözcükleri ile kökten dibe kadar olan yollar arasında birebir eşleştirme yapmak için kullanılabilir. Bir trellis, bir başlangıç durumu, bir bitiş durumu olan, döngü içermeyen ve trellis tarafından temsil edilen kod tam olarak trellisin

dili(kelimeler bütünü) olan bir sonlu otomata olarak görülebilir.

Hata ile bozulmuş olması mümkün herhangi bir sözcük alındığını varsayalım. Kodun hata olasılıkları ile trellisdeki her bir kenar üzerindeki ağırlıkları eşleştirerek maksimum olabilirlik dekodlamasını trellis içinde kökten dibe en kısa yolu bulma problemine dönüştürebiliriz. Bunu Viterbi algoritması olarak bilinen ve i derinliğine kadar olan en olabilir önekleri adım adım bulan bir dinamik programlama algoritması kullanarak yapabiliriz. Viterbi algoritmasına ileride değineceğiz.

Trellisler bir çok yaygın dekodlama metodunda kullanılmalarına rağmen, Lafourcade asimptotik iyi kodların boyunda dekodlama karmaşıklığının üssel olarak büyüdüğünü göstermiştir. Bu yüzden dekodlama için Taner çizgesi gibi değişik çizge yapıları önerilmektedir.

5.2.3 Durum Diagramı

Olası bütün kodlayıcı durumları ve aralarında geçişleri sağlayan girdiler durum diagramı vasıtasıyla gösterilebilir. Her kare ötelemeli yazıcının içeriğini gösteren bir durumdur. Aradaki eğriler bir girdinin sebep olduğu durm değiştirmeyi göstermektedir. Durum diagramındaki dallar girdi/çıktı bitleri ile etiketlenmiştir. Kodlama işlemi, bilinen bir durumdan başlayan bir durum değiştirmeler dizisine denk gelmektedir.

5.3 Viterbi Algoritması

Viterbi dekodlama algoritması, 1967'de ilk olarak Viterbi tarafından keşfedilmiş ve analiz edilmiştir. Viterbi dekodlamasının kısa kısıtlanmış uznluktaki kodlar için etkili ve pratik bir dekodlama tekniği olduğunu ilk olarak Heller gösterdi. Forney algoritmanın aslında maksimum olabilirlikli olduğunu ispat etti.

Viterbi algoritması, verilen bir vektörün maksimum olabilirlik dekodlamasını yapabilmek için $|E|$ çarpma ve $|E|-|V|+1$ toplama gerektirir. L bit kod çözmek için sadece L cinsinden doğrusal olan $L2^{k(K-1)}$ kod çözme işlemi yapılması gerçekleştirilmesi yeterlidir. Ne var ki, K değerine üssel olarak bağlı olduğu için görece kısa kod kısıt uzunluğu için kullanılabilir. Trellislerin derinlikli yapısını kullanarak bilinen bir çok algoritmadan daha hızlıdır. Ayrıca, Viterbi algoritması çok geneldir ve herhangi bir yarı-halka alfabesine uygulanabilir. . (http://people.bu.edu/trachten).

LFSR'ların GF(2) dışında da çalıştırılabileceği düşünüldüğünde, Viterbi dekodlamasının genelliği korelasyon saldırısının yeni halkaya uyarlanmasında önemli bir rol üstlenebilir.

Şimdi Viterbi algoritmasının temel tanımını yapalım:

Koda ait trellis yapısının 3 derinliğinde(daha genel olarak K derinliğinde) sabit bir periyodik yapısının olduğunu varsayalım. Bu noktadan sonra, bütün 4 duruma önceki iki durumun herhangi birisinden girilebilir. 3 derinliğinde örneğin her duruma giden ikişerden 8 kod yolu vardır. Örneğin, 000 ve 100 durumlarına karşılık gelen bilgi dizilerinin ikisi de 3 seviyesindeki 00 durumuna girebilir. Bu yollara 0 derinliğinde 00 durumunda ıraklaştı denilir. Yollar 2(daha genel olarak $k(K-1)$) ardışık özdeş bilgi bitinden sonra tekrar birlişir. Bir Viterbi kod çözücüsü verilen bir duruma giren 2^k durumun her birinin olabilirliğini hesaplar ve daha sonraki bilgilerin ışığı altında bu duruma getiren en olabilirlikli dışındaki yolları eler. Verilen bir trellis derinliğindeki $2^{k(K-1)}$ durumun her biri için bu işlem yapılır. Her kod çözme işleminden sonra her duruma götüren sadece bir yol kalır. (Bu yola *"hayatta kalan"(survivor)* denir. Kod çözücü daha sonra trellisde bir seviye daha derine inerek bu süreci tekrarlar. $K=3$ kod trellisi için 3 derinliğinde 8 yol vardır. 3 derinliğinde kod çözme her duruma giden 1 yolu eler. Geriye 4 yol kalır. 4 derinliğine inildiğinde tekrar 8 yol ile karşılaşılır. Her duruma giden en düşük olabilirlikli yolu eleyerek, Viterbi dekodlaması kaba kuvvet maksimum olabilirlikli dekodlama metodunun seçtiği hiçbir yolu reddetmez. Tarif edilen kod

çözme aslında en yüksek olabilirlikli yola karar vermektedir. Her kod çözme adımından sonra daima $2^{k(K-1)}$ yol kalmaktadır. Kalan her yol, verilen bir kodlanmış duruma giren en yüksek olabilirlikli yoldur. Tek bir en yüksek olabilirlikli yol seçmek için kodlayıcıya L bit bilgiden sonra sabit bir $K-k$ bit bilgi dizisi vererek, kodlayıcının daha önceden ayarlanmış bir duruma girmesini sağlamak olabilir. Kod çözücü daha sonra bu bilinen kod durumuna ulaşan yolu seçebilir.

5.3.1 Yol Belleği

Viterbi algoritmasını pratik bir kod çözme tekniği yapmak için temel algoritmada bazı iyileştirmeler gerekebilir. Her şeyden önce,kodlayıcıyı periyodik olarak bilinen bir duruma zorlamak ne istenen ne de gerekli bir durumdur. Kod çözücünün seçtiği $2^{k(K-1)}$ yol şu anda bulunan kod çözme derinliğinden çok daha geride birbirinden ayrı olmayacaktır. Bu gözlem, eğer kod çözücü her bir yolun geçmişindeki bilgi bitlerinden yeterince depolarsa, o zaman bütün yollardaki en eski bitlerin aynı olacağını öngörmektedir. Eğer sabit bir miktar yol geçmiş bilgisi saklanması sağlanabilirse kod çözücü trellisde bir seviye daha derine inildiği her seferde herhangi bir yoldaki en eski bitleri dışarı verebilir. Gereken yol belleği u durum sayısı $2^{k(K-1)}$ ile durum başına yol geçmiş bilgisi bit uzunluğu h çarpımına eşittir. $u = h2^{k(K-1)}$. Bir Viterbi dekodlayıcısının maliyetinin önemli bir kısmını yol geçmiş hafızası teşkil edeceğinden gerekli yol geçmiş bilgi uzunluğu h mümkün olduğunca küçük tutulmalıdır. Bu bağlamda yapılabilecek iyileştirmelerden biri herhangi bir yol üzerinde en eski bitleri tutmaktansa en yüksek olabilirlikli $2^{k(K-1)}$ yol üzerindeki en eski bitleri tutmaktır. Nasa Ames araştırma merkezinde yayımlanan ayrı iki araştırma sonucuna göre teorik ve pratik olarak kod kısıt uzunluğu K'nın 4 ya da 5 katı bir h değerinin yeterli olduğu gösterilmiştir. (Arslan(2001))

5.3.2 Durum ve Dal Metriği

Her bir duruma giden yolların karşılaştırılması, söz konusu alınan bilgi bit dizisi için söz konusu her bir yolun olabilirliğinin hesaplanmasını gerektirmektedir. Kanal belleksiz olduğundan, yolun olabilirlik fonksiyonu teker teker kod sembollerinin olabilirlikleri çarpımına eşittir. $P(r|v) = \prod_i P(r_i|v_i)$ Çarpmadan kurtulmak amacıyla, logaritma toplamını alabiliriz. Böylece $\log P(r|v) = \sum_i \log P(r_i|v_i)$'yi hesaplamaya dönüşür. Bu sayede toplamsal

bir uzaklıkla uğraşmış oluruz. Yollardan biri bir dal ile genişletildiği zaman yeni yolun metriği eski yol metriği ile yeni dal sembollerinin metriğinin toplamı olur. Bu hesabı gerçekleştirebilmek için, her bir duruma giden en iyi yolun metriği kod çözücü tarafından bir durum metriği olarak saklanmalıdır. Bu da saklanacak ek bilgi demektir.

5.3.3 Bilinmeyen Başlangıç Durumu

Buraya kadar Viterbi kod çözücüsünün kod çözme işlemi başlamadan önce kodlayıcı başlangıç durumunun bilindiği varsayıldı. Bilinen bir başlangıç durumu işlevsel olarak çok istenen bir şey değildir. Pratik uygulamalar bütün durum metrikleri başlangıçta sıfırlandığında Viterbi kod çözücüsünün yayın sırasında herhangi bir noktadan başlayabileceğini göstermiştir. Başlangıç durumunun bilinmemesinden dolayı kod çözücünün çıktısı olan ilk 3-4 kısıt uzunluğu bilgi pek güvenilir olmayabilir fakat yaklaşık 4 kısıt uzunluğundan sonra yüksek olasılıklı durum metrikleri başlangıç değerlerinden bağımsız olmaya başlar.

5.3.4 Algoritmanın Tanımı

l uzunluklu LFSR tarafından üretilen C kodu bir katlamalı koda çevrilir. Daha sonra bilinen etkili bir katlamalı kod dekodlaması kullanılarak LFSR'ın ilk durumu oluşturulmaya çalışılır. Bu katlamalı kodun kodlayıcısı uygun parite kontrol denklemleri bulunarak oluşturulacaktır. Bunun için önce bir u_n sembolü, kendisinden önce gelen B sembol $u_{n-1},\ldots,u_{n-B}$ ve gerekirse en fazla t adet başka sembolden oluşan herhangi bir lineer kombinasyonu olan parite kontrol denklemleri bulunur. t sayısının çok büyük olmaması gerektiği açıktır. Algoritmanın anlatımında $t=2$ alınacaktır.

Bu denklemleri bulmak için $n=B+1$ indeks pozisyonunu ele alalım. Üreteç matrisi için aşağıdaki gösterimi kullanalım:

$$G_{LFSR}=\begin{pmatrix} I_{B+1} & Z_{B+1} \\ 0_{l-B-1} & Z_{l-B-1} \end{pmatrix} \tag{5.6}$$

Bu durumda, u_{B+1} için ilk $B+1$ pozisyon dışında kalan ağırlığı t olan parite kontrol denklemleri Z_{l-B-1}'in toplamları sıfır sütununa denk gelen t sütununun lineer kombinasyonları olarak bulunabilir.

$t=2$ için durum oldukça basittir: Eğer son $l-b-1$ girdi göz önüne alındığında G_{LFSR}'ın

iki sütunu aynı değere sahip ise $t=2$ olacak şekilde bir parite kontrol denklemi bulunmuş olur.

Diyelim ki, u_{B+1} için elimizde m adet bu şekilde denklem olsun.

$$\begin{aligned} u_{B+1} + \sum_{i=1}^{B} c_{i1} u_{B+1-i} + u_{h1} + u_{k1} &= 0 \\ u_{B+1} + \sum_{i=1}^{B} c_{i2} u_{B+1-i} + u_{h2} + u_{k2} &= 0 \\ &\vdots \\ u_{B+1} + \sum_{i=1}^{B} c_{im} u_{B+1-i} + u_{hm} + u_{km} &= 0 \end{aligned} \tag{5.7}$$

LFSR'ın devirsel yapısı nedeniyle semboller kaydırılarak herhangi bir n pozisyonu için eçerli aynı parite kontrol kümesini bulabiliriz. Örneğin ilk denklem

$$u_n + \sum_{i=1}^{B} c_{i1} u_{n-i} + u_{h1} + u_{k1} = 0 \tag{5.8}$$

haline dönüşür.

Yukarıdaki denklemler kullanılarak üreteç matrisi, boşluklar 0 olmak üzere,

$$G = \begin{pmatrix} \ddots & \ddots & \cdots & \ddots & & \\ & G_0 & G_1 & \cdots & G_B & \\ & & G_0 & G_1 & \cdots & G_B \\ & & & \ddots & \ddots & \cdots & \ddots \end{pmatrix} \tag{5.9}$$

şeklinde yazılabilecek bir $R = 1/(m+1)$ oranlı bir katlamalı kod kodlayıcısı oluşturulur. Alt üreteç matrisleri denklem eşitlenerek bulunabilir:

$$\begin{pmatrix} G_0 \\ G_1 \\ \vdots \\ \vdots \\ G_B \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 0 & c_{11} & c_{12} & \cdots & c_{1m} \\ 0 & c_{21} & c_{22} & \cdots & c_{2m} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & c_{B1} & c_{B2} & \cdots & c_{Bm} \end{pmatrix} \tag{5.10}$$

Katlamalı kodda tanımlanmış her $v_n^{(i)}$ kod sözcüğü için sistem çıktısı z dizisinden bir tahmin bulabiliriz. $t=2$ kabul edelim. Eğer $v_n^{(i)} = u_n$ ise $P\left(v_n^{(i)} = z_n\right) = 1-p$ olur. Aksi takdirde, eğer $v_n^{(i)} = u_{hi} + u_{ki}$ ise $P\left(v_n^{(i)} = z_{hi} + z_{ki}\right) = (1-p)^2 + p^2$ olur. Bu tahminler kullanılarak, $r_n^{(0)} = z_n$

ve $r_n^{(i)} = z_{hi} + z_{ki}$, $1 \le i \le m$ olmak üzere katlamalı kod için alınan dizi rolünü üstlenecek $r = \ldots r_n^{(0)} r_n^{(1)} \ldots r_n^{(m)} r_{n+1}^{(0)} r_{n+1}^{(1)} \ldots r_{n+1}^{m} \ldots$ dizisi oluşturulabilir. Yukarıdaki kabullerden, $P(v_n^{(0)} = r_n^{(0)}) = 1 - p$ ve $1 \le i \le m$ için $P\left(v_n^{(i)} = r_n^{(i)}\right) = (1-p)^2 + p^2$ bulunur.

5.3.5 Dekodlama

LFSR'ın ilk durumunu bulmak için ardışık l bitin doğru dekodlanması yeterlidir. Bunun için Viterbi dekodlaması kullanılacaktır.

Hakiki Viterbi dekodlamasında katlamalı kod kodlayıcısının 0 durumundan başladığı kabul edilir. Halbuki bu uygulamalarda ne belli bir başlangıç, ne de bir bitiş durumu vardır. Bu durumun üstesinden gelebilmek için trellisdeki her s durumuna $\log P(s = z_1, z_2, \ldots, z_B)$ metriği karşılık getirilecektir. Sonrasında $n = B$ den başlanılarak alışılageldik dekodlama işlemleri yapılacaktır. Son noktalar arasındaki fark nedeniyle, ortadaki l sembol istenilen l bitlik dizi olarak kabul edilip $J = l + 10B$ adet bilgi sembolü üzerinden Viterbi algoritması uygulanabilir. Buradan çıkan sonuç LFSR'ın ilk durumu olarak tahmin edilir.

5.4 Saldırı Algoritması

LFSR üreteç polinomu $g(D) = 1 + g_1 D + g_2 D^2 + \ldots + g_l D^l$ olsun. $(u_1, u_2, \ldots, u_N)$ kod sözcükleri ve $(u_1, u_2, \ldots, u_l)$ bilgi sözcüğü olmak üzere, $u_{l+i} = g_1 u_{l+i-1} + g_2 u_{l+i-2} + \ldots + g_l u_i, 1 \le i \le N - l$ denklemleri yazılabilir. Buradan da, $u_{l+i} = h_{l+i}^1 u_{l+i-1} + h_{l+i}^2 u_{l+i-2} + \ldots + h_i^l u_i, 1 \le i \le N - l$ bulunur. Eğer $h(D) = h_i^1 + h_i^2 D + \ldots + h_i D^{l-1}$ gösterimini kabul edersek, $i = 1,2,\ldots,N$ için $h_{i+1}(D) = D h_i(D) \bmod g'(D)$ olduğu kolayca gösterilebilir. Bu durumda, C kodunun $l \times N$ üreteç matrisi $G = \begin{pmatrix} h_1^1 & h_2^1 & \ldots & h_N^1 \\ h_1^2 & h_2^2 & \ldots & h_N^2 \\ \vdots & & \ldots & \\ h_1^l & h_2^l & \ldots & h_N^l \end{pmatrix}$ şeklinde yazılabilir.

Sabit bir hafıza büyüklüğü B için, parite kontrol denklemleri ararken $h$ların aşağıdaki denklemleri sağlayacak şekilde ararız: ($B < i \neq j < N$ olmak şartıyla)

$$\begin{aligned} h_i^{B+1} + h_j^{B+1} &= 1 \\ h_i^{B+2} &= h_j^{B+2} \\ h_i^{B+3} &= h_j^{B+3} \\ &\vdots \\ h_i^{l} &= h_j^{l} \end{aligned} \tag{5.11}$$

Yukarıdaki denklemleri sağlayan i ve j ler için $u_i + u_j$ toplamı $u_1, u_2, \ldots, u_{B+1}$ bilgi sembollerinin bir lineer kombinasyonudur. Böylece ilk $B+1$ pozisyon dışında ağırlığı 2 olan parite kontrol denklemleri bulabiliriz.

Bu tür bütün denklemleri bulmak için yukarıdaki denklemi sağlayıp sağlamadığını kontrol etmek için bütün sütunlar kıyaslanır. Doğrudan bütün sütunların kıyaslanması $O(N^2)$ mertebesinde olacaktır. Bu yöntemin verimsizliğinden dolayı önce matrisin sütunlarını sort etmek gerekmektedir. Arslan(2001)'da veri yapısının ve sort edilmesinin üreteç matris oluşturma ve parite kontrol denklemleri bulma konusunda ne kadar etkili olduğu görülmektedir. Lineer kodlarla yapılan saldırı uygulamasında bu dikkate alınmıştır.

(i, j) çiftlerinin beklenen değeri $E(m) = \dfrac{(N-B)(N-B-1)}{2} 2^{-l+B}$ kadardır. Beklenen değer analizi lineer kodlarla yapılan saldırıda anlatılmıştır.

5.4.1 Algoritmanın Performansı

Lineer kodlarla yapılan saldırıda olduğu gibi, BSC kanalının hata olasılığı p_0 'ın kritik değeri önemlidir. Algoritmada inşa edilen katlamalı kodun oranı ile kanal kapasitesi eşitlenerek bulunur. Böylece, p_0 değeri hata yapma olasılığı sıfıra yeterince yakın olacak kodların varlığını garantileyecek maksimum olasılık olur. p_0 değeri hesaplanırken BSC kanal hata olasılığı $p = 1/2 - \varepsilon$ için $C(p) \approx \varepsilon^2 .2/\ln 2$ yaklaştırması kullanılmıştır. Oluşturulan yeni kodda hata olasılığı $p' = 1/2 - 2\varepsilon^2$ olacağından $C(2p(1-p)) \approx \dfrac{8\varepsilon^4}{\ln 2}$ yaklaştırmasını elde ederiz. Algoritma için maksimum p_0 değerleri aşağıda gösterilmiştir:

5.5 Lineer Kodlarla Yapılan bir Korelasyon Saldırısı

Katlamalı kodlarla yapılan saldırının bir benzeri, daha az bellek ile ve daha karmaşık olmmamak üzere lineer kodlarla yapıldı. Yine burada anlatılmayan ama benzer bir şekilde lineer kodlarla yapılan ve aynı konferansta anlatılan yüksek performanslı bir algoritma için Mihaljević, Foosorier ve Imai'nin(2000) yazdıkları makaleye bakılabilir. Aşağıda Chepyzhov, Johannson ve Smeets(2000) saldırısı anlatılmaktadır. Johannson'un katlamalı kodlarla yapılan saldırının mucitlerinden biri olduğunu ve saldırılarının benzer bir hat çizdiğini belirtelim.

p, hata olasılığını $1-p$ korelasyon olasılığını belirtsin. Problemimiz şu şekilde ifade edilebilir:

$p=\frac{1}{2}-\varepsilon<0.5$ olsun. Derecesi l olan LFSR geribesleme polinomumuz $g(D)$ ile gösterilsin. Problemimiz, gözlenen $z=z_1,z_2,\ldots,z_N$ çıktı dizisinden $x_1,x_2,\ldots,x_l$ LFSR ilk durumunu bulabilmek.

Teorik bazı sonuçlara varabilmek için aşağıdaki savı kullanacağız. Oluşacak lineer kodun esas kodlama teoremindeki şartları sağlayacak kadar "rassal" olduğunu farz edelim:

Teorem (Shannon1949): Eğer $R=k/n$ kodun bilgi oranı, $C(p)=1-H(p)$ ikili simetrik kanalın kapasitesi ise, doğrusal (n,k) kodları ailesinde, kod çözülürken yapılacak hata olasılığı sıfıra yaklaşır. Burada $H(x)$ fonksiyonu, $H(x)=-x\log_2 x-(1-x)\log_2(1-x)$ şeklinde tanımlanan *ikili entropi fonksiyonu*dur.

Siegenthaler, yukarıda bahsi geçen $[N,l]$ kod için dekodlama yöntemi olarak bütün kod sözcüklerini taramayı düşündü. Bu algoritma (hata açısından) optimaldir çünkü bir en yüksek olabilirlik yöntemidir. Siegenthaler, eğer $N>n_0$ ise başarı olasılığının 1/2 den yüksek olacağını gösterdi. Burada, n_0 dan kastımız $n_0=\lceil l/C(p)\rceil$ kapasite ve uzunluğa bağlı kritik uzunluktur. Algoritmanın karmaşıklığı yaklaşık $O\left(2^l\cdot l/C(p)\right)$ kadardır. Hızlı korelasyon saldırılarının ana fikri bu 2^l çarpanından kurtulabilmek ve $\alpha<1$ olacak şekilde $O\left(2^{\alpha l}\right)$ karmaşıklığında algoritmalar geliştirebilmektir.(Chepyzhov vd.)

Şimdi kodlama teorisinden bazı sonuçları hatırlayalım. Her x_i sembolü başlangıçtaki l ilk değerin bir doğrusal bileşeni olduğundan $(x_1,x_2,\ldots,x_N)$ sözcükler kümesi bir C lineer

kodunu oluşturur. $(x_1, x_2, \ldots, x_N)$ lere kod sözcükleri denir. $(x_1, x_2, \ldots, x_l)$ sembolleri kod sözcüğünü oluşturmak için gerekli bütün bilgiyi içermektedir. Bu yüzden $(x_1, x_2, \ldots, x_l)$ e (bir) bilgi sembolü kümesi denir. Eğer geribesleme polinomu $c_0 = c_l = 1$ olmak üzere $g(D) = c_0 + c_1 D + \ldots + c_l D^l$ ise aşağıdaki bağıntıları buluruz.

$$\begin{cases} c_l x_1 + & c_{l-1} x_2 + & \ldots & + c_1 x_l + & c_0 x_{l+1} & = & 0 \\ c_l x_2 + & c_{l-1} x_3 + & \ldots & c_1 x_{l+1} + & c_0 x_{l+2} & = & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & = & 0 \\ c_l x_{N-l} + & c_{l-1} x_{N-l+1} + & \ldots & + c_1 x_{N-1} + & c_0 x_N & = & 0 \end{cases} \tag{5.12}$$

Burada c_i ler $g(D)$ polinomunun katsayılarıdır. H matrisini aşağıdaki gibi tanımlarsak,

$$H = \begin{pmatrix} c_l & c_{l-1} & c_{l-2} & \ldots & c_0 & 0 & \ldots & 0 \\ 0 & c_l & c_{l-1} & \ldots & c_1 & c_0 & \ldots & \vdots \\ 0 & 0 & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \ldots & c_l & c_{l-1} & \ldots & c_1 & c_0 \end{pmatrix}_{(N-l)\times l}$$

matrisi LFSR kodunun parite kontrol matrisi olur. Daha önce de bahsettiğimiz gibi, lineer kodumuz $H(x_1, x_2, \ldots, x_n)^T = 0$ olan bütün kod sözcüklerinden oluşur. H matrisine LFSR kodunun parite kontrol matrisi denir. Yukarıdaki denklemde $c_0 = c_l = 0$ olduğunu anımsarsak LFSR çıktılarını H matrisinin girdileri cinsinden ifade edebiliriz. Bu da bize $h_i(D) = D^{i-1} \bmod g(D) \quad i = 1,2,\ldots,N$ olmak üzere LFSR'ın üreteç matrisi G yi oluşturma imkanı verir.

$$G = \begin{pmatrix} h_1^1 & h_2^1 & \ldots & h_N^1 \\ h_1^2 & h_2^2 & \ldots & h_N^2 \\ \vdots & & \ldots & \\ h_1^l & h_2^l & \ldots & h_N^l \end{pmatrix}_{l\times N} \tag{5.13}$$

G nin ilk $l \times l$ lik kısmının birim matrisiçıkması şaşırtıcı değildir. Sonraki sütunlar ise kolaylıkla geribesleme polinomundan hesaplanabilir.

Algoritmanın gözlenen çıktı dizisi $z = (z_1, z_2, \ldots, z_N)$ ve LFSR çıktısı $x = (x_1, x_2, \ldots, x_N)$ ise biraz önceki LFSR ilk hali bulma problemi aşağıdaki probleme dönüşür:

$p < 0.5$ ($p > 0.5$ durumunda bütün bitleri tersine çevirerek aynı sonuca ulaşabiliriz) ve $g(D)$ üreteç polinomunun derecesi l olsun. Buna karşılık gelen $[N,l]$ LFSR kodu C'yi gözönüne alalım. Problemimiz bize ulaşan z sözcüğünden x kod sözcüğünü bulmaktır.

Uyarı : LFSR teorisi ve dizi şifreleme teorisinde, p nin tipik değerleri $p = 0.25, 0.30, 0.4, 0.45$ gibi 0.5'e yakın değerlerdir. Fakat genelde hata düzeltme kodları tasarlanırken tahammül edilen hata oranları çok çok daha düşüktür. Bu yüzden, kodlama teorisinde alıştığımızdan çok daha yüksek gürültü ile mücadele etmek durumundayız. Diğer yandan, kodlama teorisinin aksine, algoritamamızın ilk durumu bulma olasılığının çok düşük olmadığını sözgelimi $1/2$ olduğunu göstememiz yeterlidir. Bu yüzden, dekodlama sırasında çok daha yüksek hata payı kabul edilebilir.

5.5.1 Yeni bir Hızlı Korelasyon Saldırısı

$k < l$ den küçük sabit bir sayı olsun. $z_1,\ldots,z_N$ 'i gözlemlerken $x_1,\ldots,x_k$ sembollerini geri oluşturacak bir algoritma tanımlayacağız. LFSR temel bağıntısından elde edilen kod sisteminde $h_i^1 = h_j^1, h_i^2 = h_j^2,\ldots,h_i^l = h_j^l$ olacak şekilde denklem çiftlerini arayacağız. G üreteç matrisini düşündüğümüzde bu G matrisinde k ıncı satırdan sonrası aynı sütunları bulmak demektir. Bu şekildeki farklı çiftlerin sayısı n_2 olsun. Bunları $\{i_1,j_1\},\{i_2,j_2\},\ldots,\{i_{n_2},j_{n_2}\}$ olarak gösterelim. Eğer i ve j için yukarıdaki eşitlikler sağlanıyorsa, o zaman $x_i + x_j$ toplamı sadece $x_1,x_2,\ldots,x_k$ bilgi sembollerinin bir lineer kombinasyonu olur ve geri kalan bilgi sembolleri $x_{k+1},x_{k+2},\ldots,x_l$ 'den bağımsız olur.

$$x_i + x_j = \left(h_i^1 + h_j^1\right)x_1 + \left(h_i^2 + h_j^2\right)x_2 + \ldots + \left(h_i^k + h_j^k\right)x_k \tag{5.14}$$

Bu ise, $\left(X_1,\ldots,X_{n_2}\right) = \left(x_{i_1} + x_{j_1},\ldots,x_{i_{n_2}} + x_{j_{n_2}}\right)$ dizisinin C_2 ile göstereceğimiz bir (n_2,k) kodu oluşturduğunu ve oluşan kodun bilgi sembollerinin ilk k sembol olduğunu (aslında oluşan kod daha az sayıda sembole de bağlı olabilir fakat önemli olan bütün sembollerin bu k sembolden üretilebilmesidir) , diğer bir deyişle oluşun kodun boyutunun $\leq k$ olduğunu gösterir. C_2 kodunun üreteç matrisi

$$G_2 = \begin{pmatrix} h_{i_1}^1 + h_{j_1}^1 & h_{i_2}^1 + h_{j_2}^1 & \ldots & h_{i_{n_2}}^1 + h_{j_{n_2}}^1 \\ h_{i_1}^2 + h_{j_1}^2 & h_{i_2}^2 + h_{j_2}^2 & \ldots & h_{i_{n_2}}^2 + h_{j_{n_2}}^2 \\ \vdots & \vdots & \ldots & \vdots \\ h_{i_1}^k + h_{j_1}^k & h_{i_2}^k + h_{j_2}^k & \ldots & h_{i_{n_2}}^k + h_{j_{n_2}}^k \end{pmatrix} \text{ olur.}$$

$Z_1 = z_{i_1} + z_{j_1}, Z_2 = z_{i_2} + z_{j_2},\ldots,Z_{n_2} = z_{i_{n_2}} + z_{j_{n_2}}$ ile gösterelim. Zaten çıktı dizisinde z sembollerini gözlemlediğimiz için $Z_1,Z_2,\ldots,Z_{n_2}$ sembollerini hesaplayabiliriz. Z sembol dizisi bizim için C_2 kodunda gürültülü bir kanal üzerinden bize ulaşan ve aslını bulmaya çalışacağımız kod sözcüğü görevi olacaktır. C_2 kodunun hata olasılığı C kodu ile aynı değildir. Eğer $(e_1,e_2,\ldots,e_N)$,yi C kodu için gürültü dizisi olarak alırsak, ($e_i = x_i + z_i$ olmak üzere) C_2 kodu için gürültü dizisi $E_1,E_2,\ldots,E_{n_2}$ olacaktır. Modelimiz ikili simetrik kanal olduğundan, bütün e_i lerin p hata olasılıklı, bağımsız, tesadüfi ikili değişkenler olarak kabul edebiliriz.

$1 \le m \le n_2$ olmak üzere $i_m \neq j_m$ için $E_m = e_{i_m} + e_{j_m}$ değerleri bağımsız tesadüfi ikili değişken olacaktır. Z_m nin hatalı gelmesi z_{i_m} nin hatalı, z_{j_m} nin hatasız veya tam tersi hataların yer değiştiği durumda olur. Bu yüzden C_2 kodunda bir sembol için hata olasılığı

$$P_2 = \Pr(e_{i_m} + e_{j_m} = 1) = p(1-p) + (1-p)p = 2p(1-p) = 1/2 - \varepsilon^2 \tag{5.15}$$

olacaktır. Böylece, daha küçük boyutlu bir C_2 lineer kodu oluşturmuş olduk fakat hata düzeltebilme olasılığı yönünden kaybımız oldu. Ne var ki, yeni kodumuz C_2 nin uzunluğu tek bir şekilde dekodlamaya yettiği sürece bu kodun dekod edilmesi C kodundan daha hızlı olacaktır. (İleride de göstereceğimiz gibi daha küçük bir alt uzayla uğraştığımızdan işlemler daha hızlı olacaktır.) Biraz önceki kritik uzunluk tanımından yola çıkacak olursak bilgi sözcüğünü güvenilir bir şekilde elde edebilmek için $n_2 > k/C(p_2)$ olmalıdır. Şimdi algoritmayı tarif edebiliriz:

Veri : hedef LFSR uzunluğu l, üreteç polinomu $g(D)$.

Ön Hesaplama : $k < l$ (örneğin $k = l/2$) olacak şekilde işlemsel karmaşıklık seviyesi tespit edelim. G üreteç matrisini oluşturalım. Bir sıralama(sort) algoritması kullanarak, G nin sütunlarını $\left(h_i^{k+1}, h_i^{k+2}, \ldots, h_i^{l}\right), i = 1,2,\ldots,N$ ye göre sıralayalım. (Bu sıralama yapılmadığı zaman algoritmanın çok çok yavaş olduğu gözlendi). Bir önceki bölümde anlatılan koşulları sağlayan bütün $\{i, j\}$leri bulalım. Her ikili için, (i, j) indisleri ile birlikte $\left(h_i^1 + h_j^1, h_i^2 + h_j^2, \ldots, h_i^h + h_j^k\right)$ değerlerini saklayalım. Böylece G_2 üreteç matrisi için C_2 kodunu inşa etmiş olduk.

NOT : $(x_1, x_2, \ldots, x_k)$ sembollerini geri aldıktan sonra ilk durumun diğer sembollerini de geri elde etmeliyiz. Akla ilk gelen metod, aynı işlemleri sonraki k sembol için yani $(x_1, x_2, \ldots, x_k)$ için de tekrarlamaktır. Kod devirsel olduğu için aynı parite denklemlerini kullanabiliriz.

Dikkat edilmesi gereken bir nokta ilk k sembol bulunduktan sonra problemimiz ilk baştaki probleme oranla daha kolaydır.Bu yüzden karmaşıklık ve hata olasılığı hesabını bu adımda ihmal edebiliriz.(örneğin 80 uzunluğundaki bir LFSR'ın ilk 20 bitini elde edersek, bu 20 biti kullanarak yeni bir dekodlama problemi elde ederiz fakat bu sefer uğraşacağımız LFSR boyu 60 olacaktır.)

Sadece ikili denklemleri ele almak zorunda değiliz. Sadece $(x_1, x_2, \ldots, x_k)$ sembollerini içeren t parite kontrol denklemlerini kullanabiliriz. Bu durumda ön işlem kısmı şöyle olacaktır.

Ön Hesaplama : $k < l$ (örneğin $k = l/2$) olacak şekilde işlemsel karmaşıklık seviyesi ve $t \geq 2$ tespit edelim. G üreteç matrisini oluşturalım. Bir sıralama(sort) algoritması kullanarak, G nin sütunlarını $\left(h_i^{k+1}, h_i^{k+2}, \ldots, h_i^{l}\right)$, $i = 1,2,\ldots,N$ ye göre sıralayalım. (Bu sıralama yapılmadığı zaman algoritmanın çok çok yavaş olduğu gözlendi) için

$$\sum_{j=1}^{t} h_{i(j)}^{m} = 0 \tag{5.16}$$

denklemini sağlayan bütün $\{i(1), i(2), \ldots, i(t)\}$ indis kümelerini bulalım. Bu şekildeki kümelerin sayısı n_t olsun. Her bir küme için, $\{i(1), i(2), \ldots, i(t)\}$ ile birlikte

$$\sum_{j=1}^{t} h_{i(j)}^{1}, \sum_{j=1}^{t} h_{i(j)}^{2}, \ldots, \sum_{j=1}^{t} h_{i(j)}^{k} \tag{5.17}$$

değerlerini saklayalım.Böylece (n_t, k) parametreli bir C_t kodu inşa etmiş olduk.

Dekodlama

Veri: Gözlenen vektör $(z_1, z_2, \ldots, z_N)$.

Adım 1: Aşağıdaki eşitliklerle verilen sembolleri hesapla.

$$Z_1 = \sum_{j=1}^{t} z_{i_1(j)}, Z_2 = \sum_{j=1}^{t} z_{i_2(j)}, \ldots, \sum_{j=1}^{t} z_{i_n(j)} \tag{5.18}$$

Adım 2: C_t kodunu bütün olası 2^k kod sözcüğü $(x_1, x_2, \ldots, x_k)$ları tarayarak dekodla.

Simetrik İkili Kanal modelimizi kullanarak bütün e_i'lerin $p = 1/2 - \varepsilon$ olasılıklı tesadüfi ikili değişken olduklarını kabul ederek, $E_m = \sum_{j=1}^{t} e_{i_m(j)}$ sembollerinin

$$p_t = \Pr\left(\sum_{j=1}^{t} e_{i_m(j)} = 1\right) = 1/2 - 2^{t-1}\varepsilon^t \tag{5.19}$$

hata olasılıklı bağımsız rassal ikili değişken oldukları gösterilebilir.

5.5.2 Algoritmanın Teorik Analizi

Meşhur "doğumgünü paradoksu" sayesinde kaç $(i,j)\,1 \le i,j \le N$ ikilisinin yeni kodu oluşturmak için kullanılan denklemleri sağlamasını beklediğimizi hesaplayıp, bu beklenen n_2 değerinin N ve k'ya nasıl bağlı olduğunu bulabiliriz.

Lemma (Doğumgünü Yanıltmacı) $\xi_1,\xi_2,\ldots,\xi_N$ eleman sayısı L olan kümeden değerler alan düzgün dağılımlı rassal(tesadüfi) değişkenler olsun. $i \neq j$ için ξ_i ile ξ_j lerin birbirinden bağımsız olduklarını varsayalım. Eğer $\xi_i = \xi_j$ olan $\{i,j\}$ ikililerinin sayısını ψ ile gösterirsek, ψ'nin beklenen değeri $E(\psi) = \dfrac{N(N-1)}{2L}$ olur.

NOT : Bu önermeye doğumgünü yanıltmacı denilmesinin sebebi şu matematiksel gerçeğin genellemesi olmasındandır: Rasgele seçilmiş (sadece) 23 kişi arasında iki kişinin doğumgünlerinin yılın aynı günü olma ihtimali ½'den büyüktür. Genelde insanların ½ olasılığının çok daha büyük bir sayıda yakalanacağını beklediklerinden bu gerçeğe doğumgünü yanıltmacı denmiştir. Bu önerme kriptolojide özellikle hash(öğüt) fonksiyonlarının analizinde çok kullanılan bir argümandır.

Doğumgünlerinde **mutlaka** çakışma olması gereken topluluğun (artık yıllar da düşünüldüğünde) en az 367 kişi olması gerektiği gerçeği ise "güvercin yuvası prensibi" olarak bilinir ve sayma prensibi gibi çok kullanılan bir argümandır. Bütün anahtar uzayını tarama, bu prensibin bir uygulamasıdır.

İspat : (İspata geçmeden önce bu lemmanın değişik şekillerde ifadesine rastlamak mümkündür. Burada biz ispat ve ifadeyi analizimize uygun olacak şekilde ifade edeceğiz.)

Her i<j için $\pi_{i,j} = \begin{cases} 1 & \xi_i = \xi_j \ ise \\ 0 & aksi\ halde \end{cases}$ rassal değişkenini tanımlayalım. Bu değerlerin sayısı $N(N-1)/2$ dir. ξ_i ve ξ_j değerleri birbirinden bağımsız olduklarından $P(\pi_{i,j}=1) = L^{-1}$ olduğu kolayca görülür. Bu yüzdenbeklenen değer de bu olur. $\psi = \sum_{\{i,j\}} \pi_{i,j}$ olduğundan

$$E(\psi) = \sum_{\{i,j\}} E(\pi_{i,j}) = \frac{N(N-1)}{2} L^{-1} \tag{5.20}$$

olur. İspat böylece tamamlanmış olur.

Bu lemmayı uygulayabilmek için $\xi_i = \left(h_i^{k+1}, h_i^{k+2}, \ldots, h_i^{l}\right),\ i = l+1, \ldots, l+N$ alalım. $L = 2^{l-k}$ olur. LFSR teorisinden ξ_i lerin düzgün dağılımlı, ikişer ikişer bağımsız neredeyse rassal değerler olduklarını biliyoruz.

Tabii Sonuç 1: İstediğimiz koşulları sağlayan $\{i, j\}$ çiftlerinin sayısı n_2 ise $E(n_2) = \frac{N(N-1)}{2} 2^{-(l-k)}$ olur.

$(x_1, x_2, \ldots, x_k)$ sembollerini 0.5'e yakın olasılıkla tutturabilmek için gözlemlenmesi gereken çıktı uzunluğu N 'in beklenen değeri aşağıdaki sonuçla hesaplanabilir.

Teorem : k, l, ε verilmiş olsun. N, algoritmanın başarılı olabilmesi için gerekli gözlemlenmesi gerekli dizi uzunluğu ise, $N \approx 1/2\sqrt{k(\ln 2)}\varepsilon^{-2} 2^{\frac{l-k}{2}}$ olmalıdır.

İspat : Biraz önceki sonuçtan C_2 kodunun beklenen değeri n_2 uzunluğunun yaklaşık $E(n_2) = \frac{N(N-1)}{2} 2^{-(l-k)}$ olduğunu biliyoruz. Daha önceki bir sonuçtan yüksek olasılıklı ve başarılı bir dekodlama için uzunluğunun $k/C(p_2)$ olduğunu biliyoruz. Kapasite için $C(p_2) \approx (2\varepsilon^2)^2 . \frac{2}{\ln 2}$ ve $N(N-1) \approx N^2$ yaklaşık değerlerini kullanırsak istenilen sonuç elde edilir.

Biraz önceki teorem sayesinde gözlenen değer sayısı, korelasyon olasılığı ve işlemsel karmaşıklığın birbiriyle olan ilişkilerini yaklaşık olarak ifade etmiş olduk. k'yı işlem karmaşıklığı açısından temel alacak olursak verilen k değeri için tam karmaşıklığın ne olacağına yakından bakalım.

İşlem karmaşıklığının önişlem ve dekodlama safhası olarak iki ayrı kısımda inceleyeceğiz. Önişlem kısmında C_2 kodunun bütün parite kontrol hesaplarının hesaplanması $O(N \log N)$ mertebesinde olacaktır. Ayrıca G_2 matrisinin her bir sütunu oluşturan iki indis ile birlikte üreteç matrisin depolanma işleminde gereken yer en çok $n_2(k + 2\log N)$ kadardır.

Dekodlama işleminde karmaşıklık $O(2^k \cdot n_2)$ kadardır. Demek ki daha küçük bir k alırsak dekodlama işlemindeki 2^k çarpanını azaltırız fakat bu sefer de n_2 ve dolayısıyla gözlenmesi gereken dizi uzunluğu N artar.

Dikkat edilirse dekodlama işleminde hafıza kullanımı ihmal edilebilir çünkü hafızada sadece en uygun kelimeyi tutuyoruz. (Hafızada birden fazla kelime tutularak dekodlama işlemi hızlandırılabilir.) Bahsettiğimiz diğer algoritmalarda dekodlama işleminde yoğun bir hafıza kullanımı olduğuna dikkat ediniz.

İzlenmesi gereken dizi uzunluğu $t \geq 2$ için aşağıdaki gibi genelleştirilebilir:

Teorem : k,l,ε,t verilmiş olsun.Algoritmanın başarılı olabilmesi için gözlenmesi gereken dizi uzunluğu $N >> n_t$ kabulü altında $N \approx 1/4 \cdot (2kt!\ln 2)^{1/t} \cdot \varepsilon^{-2} \cdot 2^{\frac{l-k}{t}}$ olmalıdır.

İspat : C_t kodunun beklenen uzunluğu yaklaşık $\frac{N^t}{t!} 2^{-(l-k)}$ olacaktır. Başarılı bir dekodlama olasılığının yüksek olması için bu uzunluk en az $k/C(p_t)$ olmalıdır.Kapasite için aşağıdaki yaklaştırma kullanılırsa istenen sonuç bulunur.

$$C(p_t) \approx \left(2^{t-1}\varepsilon^t\right)^2 \cdot \frac{2}{\ln 2}$$

Sadece iki sütunun eşit olduğu değil de üç sütunun toplamının veya dört sütunun toplamının 0'a eşit olması durumunda algoritma daha yüksek olasılıkla doğru ilk durumu vermekte fakat karmaşıklık ve kullanılan hafıza miktarı çok büyümektedir. C_t kodundaki bütün parite kontrol denklemlerinin hesaplanmasıen az $O(N^2)$ mertebesinde olmaktadır.

Ayrıca, C_t kodunun n_t ile göstereceğimiz uzunluğu (gürültü daha da artacağından) daha da artacaktır. Kullanılması gereken hafıza miktarı $n_t(k + t\log_2 N)$ mertebesinde olacaktır.

Bir diğer dikkat edilmesi gereken nokta da, $N >> n_t$ şartını her zaman sağlanmamasının yanında bazı sütunlar birkaç kez kullanılacağı için C_t kodunun pozisyonlarındaki bağımsız şartı geçerli olamayacaktır. Bu nedenle yapılan gözlemler $t > 2$ için algoritmanın $t = 2$ den daha iyi sonuç verdiğini gösterse de teoremde belirtilen performansın altına düşmektedir.

Bir sonraki bölümde bu saldırının NESSIE projesi adaylarından LILI algoritmasına uygulanması anlatılmaktadır.

5.6 Lili Algoritmasına Korelasyon Atağı

LILI-128 kayaranahtar üreteci iki ikili LFSR ve neredeyse rassal kayaranahtar dizisi üreten basit ve hızlı bir kriptosistemdir. LILI-128, LILI kriptosistem ailesinin anahtar boyu 128 bit olan bir üyesidir. (NESSIE projesi .....) Hızlı korelasyon saldırısı ile LILI-128 algoritmasının güvenliğinin çok daha düşük olduğu gösterilebilir. Aşağıda bu algoritmaya biraz önce tarif edilen lineer kodlarla saldırı uygulanacaktır. Algoritmanın yapısından dolayı temel algoritmada bazı değişiklikler yapılacaktır.

İlk olarak LILI-128 algoritmasını kısa bir tanıtımını yapalım. Sistemde $LFSR_c$ ve $LFSR_d$ ile göstereceğimiz iki LFSR'dan oluşmaktadır. İki LFSR'ın ilk durumlarında toplam 128 bit bulunmaktadır. $LFSR_c$ uzunluğu 39 ve LFSr teorisinden alıştığımız üzere geribesleme polinomu $g_c(x)=x^{39}+x^{35}+x^{33}+x^{31}+x^{17}+x^{15}+x^{14}+x^2+1$ ilkel olan ve dolayısıyla maksimum uzunluklu bir dizi üreten bir LFSR'dır. $LFSR_c$ 'nin 12 ve 20. pozisyonları iki değerli $f_c(y_1,y_2)=2y_1+y_2+1$ fonksiyonuna gönderilir. Bu fonksiyon çıktılarının oluşturduğu diziyi $c=\{c_k,\ k\geq 1\}$ ile gösterelim. c_k dizisinin aldığı değere göre geribesleme polinomu $g_d(x)=x^{89}+x^{83}+x^{80}+x^{55}+x^{53}+x^{42}+x^{39}+x+1$ olan 89 uzunluklu $LFSR_d$ 'nin çıktı vermeden önce kaç kere çalıştırılacağını gösterir. Görüleceği gibi c_k değeri 1 ile 4 arasında değişmektedir. En son çalışmadaki $LFSR_d$ durumunda 10 değişik pozisyondaki bit aşağıda daha yakından inceleyeceğimiz $f_d : F_2^{10} \to F_2$ Booleen fonksiyona yollanır. f_d fonksiyonuna. Fonksiyon çıktısı kayar anahtar dizisini oluşturur.

Kriptoanaliz yaparken önce kayar anahtar dizisinin $LFSR_d$ düzenli çalıştırılıyor gibi farzedip f_d fonksiyonun çıktısını $d=(d_1,d_2,\ldots,d_M)$ ile gösterelim.Ayrıca $s_k=\sum_{i=1}^{k} c_i,\quad k=1,\ldots,N$ olmak üzere $s=(s_1,s_2,\ldots,s_N)$ dizisi tanımlayalım. d ve s dizilerini kullanarak z dizisini $z_k=d_{s_k},\quad k=1,\ldots,N$ olarak ifade edebiliriz.

Saldırının genel prensipi şöyledir: Bir $LFSR_c$ ilk durumu tahmin edelim ve c denetleme dizisini ve buna karşılık gelen $s_k=\sum_{i=1}^{k} c_i,\quad k=1,\ldots,N$ olmak üzere bu diziye karşılık gelen s dizisini hesaplayalım. Eğer tahmin edilen $LFSR_c$ ilk durumu gerçekse, z ile birlikte s dizisi bize N adet $d_{s_1},d_{s_2},\ldots,d_{s_N}$ sembolünü verecektir.

Sonra d dizisinin elde edilen N sembolünü kullanarak $LFSR_d$ nin ilk halini bulmaya

çalışacağız. $LFSR_d$ nin düzgün çalıştırıldığında elde edilen diziyi $u=(u_1,u_2,\ldots,u_N)$ ile gösterlim. f_d nin i nci sembolü $d_i=f_d(u_{i-89},u_{i-88},u_{i-86},u_{i-79},u_{i-77},u_{i-69},u_{i-59},u_{i-45},u_{i-24},u_{i-9})$ şeklinde hesaplanr.(Bkz LILI kaynağı) Bir sonraki adım ise f_d fonksiyonunn bir lineer yaklaştırmasını bulmak olacaktır. Bunun için Walsh transformunu kullanabiliriz.

Tanım : $f:F_2^n\to F$, $f(x)$ fonksiyonunun **Walsh transformu** $F(w)$ reel değerli fonksiyonu $F(w)=\sum_x(-1)^{f(x)\oplus w\cdot x}$ şeklinde verilir. "." işlemi alıştığımız $w\cdot x=w_1x_1+\ldots+w_nx_n$ skaler çarpımdır.

Tanım : Bir Booleen fonksiyonu $f(x)$'e ait **nonlineerlik** N_f ile gösterilir ve en yakın afin fonksiyona olan Hamming uzaklığıdır. Diğer bir deyişle, eğer n değişkenli afin fonksiyonların kümesini A_n ile gösterirsek ve f,g sırasıyla $f(x),g(x)$ fonksiyonlarının doğruluk tablosu ise $N_f=\min_{g\in A_n}d_H(f,g)$'e eşit olur. d_H, iki vektör arasında aynı sıradaki farklı sembollerin farkını sayan Hamming uzaklığıdır.

Walsh transformu kullanılarak $f(x)$ fonksiyonunun nonlineerliği Walsh dönüşümü sayesinde $N_f=2^{n-1}-\frac{1}{2}\max_w|F(w)|$ ile hesaplanabilir.

LILI-128'de kullanılan f_d fonksiyonunun nonlineerliği $N_{f_d}=480$'dir. Nonlineerliğin tanımından anlamı $d_H(f_d,f_l)=480$ olacak şekilde bir lineer fonksiyon bulunabilir. Bu da bize f_d fonksiyonu ile korelasyonlu bir fonksiyon verir.

$$P(f_d(x)=f_l(x))=\frac{1024-480}{1024}=0.53125$$

(Teorik olarak f_d nonlinnerlik sınırına çok yakın olduğundan bu fonksiyonun afin fonksiyonlara yeterince uzak olduğunu düşünebiliriz.)

f_l, f_d ye en yakın tek fonksiyon değildir. Aslında F_2^{10} da $F(w)=0$ olan 720, $F(w)=\pm32$ olan 64, $F(w)=\pm64$ olan 240 adet w vektörü vardır. Bu da $P(f_d(x)=f_{l_i}(x))=0.53125$, $1\le i\le 240$ olan $f_{l_1},f_{l_2},\ldots,f_{l_{240}}$ birbirinden farklı afin fonksiyonları vardır.

LILI_128'e bir önceki bölümde anlattığımız algoritmayı uygulayabilmek için bazı değişiklikler yapacağız. Farkedileceği gibi daha önce algoritma çıktısı ile LFSR çıktıları arasındaki korelasyonu direkt olarak hesaplıyorduk. $t = 3$ alalım. Mümkün olan bütün LFSR_d dizilerini Λ ile gösterelim. LFSR_d nin boyu 89 olduğundan $|\Lambda| = 2^{89}$ olduğundan ve sabit bir N için, bu uzunlukta Λ den alınan diziler $[N,89]$ bir lineer kod oluşturur.Bu kodu C ile gösterelim. Bu koda karşılık $89 \times N$ lik G_{LFSR} üreteç matrisi karşılık gelir. u_o ilk durum ise oluşan LFSR dizisi $u = u_o G_{LFSR}$ yardımıyla bulunabilir. f_l fonksiyonunun oluşturduğu dizi, doğrusallık nedeniyle ilk durumdaki bitlerden elde edilebilir. Bu yüzden f_l fonksiyonunu G_l üreteç matrisi ile oluşturuyorsak G_l nin boyutu da $89 \times N$ olacaktır. f_l fonksiyonlarının hepsinin f_d fonksiyonu ile korelasyonu aynı olduğuna göre bu fonksiyonların hepsini doğrusal yaklaştırma amacıyla kullanabiliriz. Eğer üreteç matrisleri yanyana eklersek $G_1|G_2|\ldots|G_{240}$ $89 \times 240N$ lik G' matrisini buluruz. Burada $|$ matrislerin yanyana eklendiğini gösteren semboldür. Diğer bir deyişle, g_i $i$ninci sütunu göstermek üzere $G' = (g_1, g_2, \ldots, g_N, g_{N+1}, \ldots, g_{2N}, \ldots\ldots, g_{240N})$ şeklinde ifade edilebilir. $k < 89$ sabit bir sayı olsun. Önce * herhangi bir değeri temsil etmek üzere $(g_{i_1} + g_{i_2} + g_{i_3})^T = (\underbrace{*,*,\ldots,*}_{k}, \underbrace{0,0,\ldots,0}_{89-k})$ koşulunu sağlayan $(g_{i_1}, g_{i_2}, g_{i_3})$ üçlülerini bulalım.

Bu üçlüleri şu şekilde bulabiliriz. G' matrisinin her sütununu son $89-k$ girdi değerine göre sort edelim. Her g_{i_1}, g_{i_2} sütun ikilileri için son $89-k$ pozisyondaki $g_{i_1} + g_{i_2}$ değerlerini hesaplayalım. $(g_{i_1} + g_{i_2} + g_{i_3})^T = (\underbrace{*,*,\ldots,*}_{k}, \underbrace{0,0,\ldots,0}_{89-k})$ koşulunu sağlayan bir g_{i_3} sütunu olup olmadığını kontrol edelim. Sütunlar sıralanmış olduklarından bu işlem çok daha çabuk yapılabilir. Önişlem safhasının karmaşıklığı yaklaşık $O(N^2)$ olacaktır. Koşulu sağlayan üçlülerin sayısı m olsun. Bütun bu üçlülerin indislerini $\{i_1(k), i_2(k), i_3(k)\}$, $1 \le k \le m$ ile gösterelim.

Daha önce de görüldüğü gibi, $(v_{i_1(1)} + v_{i_2(1)} + v_{i_3(1)}, v_{i_1(2)} + v_{i_2(2)} + v_{i_3(2)}, \ldots, v_{i_1(m)} + v_{i_2(m)} + v_{i_3(m)})$ sembolleri bir (m,k) kodu C_3 oluşturur. z yi kullanarak $Z_k = z_{i_1(k)} + z_{i_2(k)} + z_{i_3(k)}$, $1 \le k \le m$ hesaplanabilir. $(Z_1, Z_2, \ldots, Z_m)$ sembolü C_3 kodu için ele geçen bir sözcük gibi olur. z_i ile v_i arasındaki korelasyondan dolayı $z_i = v_i + e_i$ şeklinde yazabiliriz. Burada e_i ikili rassal

değişken hatayı temsil etmektedir. $P(e_i = 1) = p = P(z_i \neq v_i) = 1 - 0.53125$ olasılığında değer alacaktır. C_3 için hata vektörünü $E_k = e_{i_1(k)} + e_{i_2(k)} + e_{i_3(k)}$, $1 \leq k \leq m$ olmak üzere $(E_1, E_2, \ldots, E_m)$ olarak alalım. e_i ler birbirinden bağımsız değişkenler olduğundan, $k = 1,2,\ldots,m$ için E_k değişkenleri de hata olasılığı $p_3 = P(e_{i_1(k)} + e_{i_2(k)} + e_{i_3(k)} = 1) = 1/2 - 4\varepsilon^3$ olan bağımsız ikili rassal değişkenler olacaktır. LILI-128 için $\varepsilon = 0.03125$ alacağımızdan $p_3 \approx 0.49988$ olur.

6. EDİT UZAKLIĞI KORELASYON SALDIRISI

LFSR'lara uygulanabilecek korelasyon saldırılarından biri de edit uzaklığı korelasyon saldırısıdır. Bu bölümde bu saldırı türü anlatılmaktadır. Yine zaman kontrollü bir üreteç olan kısaltılmış üretece uygulanan korelasyon saldırısı da edit uzaklığı temelli olsa da tanımlanan edit uzaklığının farklı oluşu nedeniyle burada anlatılmamıştır.

6.1 Değişmeli Adım Üreteci

Değişmeli adım üreteci(alternating step generator) [Günther 87] birisi($LFSR_3$) düzenli çalıştırılan diğer ikisi($LFSR_1$ Ve $LFSR_2$) bu düzenli çalışan LFSR'a bağlı olarak dur/ilerle yapılan bir üreteçtir. Üçüncü LFSR yerine herhangi ikili kayar anahtar üreteci kullanılabilir. Daha açık ifade etmek gerekirse eğer ($LFSR_3$den gelen) kontrol biti 0 ise $LFSR_1$ çalıştırılır $LFSR_2$ çalıştırılmaz, 0 ise tam tersi olur. Çıktı dizisi bu iki LFSR dizisinin bitleri toplamı olur. Yine aynı kaynakta, kontrol dizisinin($LFSR_3$ rolündeki dizinin) 2^k periyotlu bir de Brujin dizisi olma varsayımı altında değişmeli adım üretecinin uzun bir P periyodu, yüksek bir L doğusal karmaşıklığı olduğu ve periyot boyunca kısa çıktı desenlerinin göreceli frekanslarının düzgün dağılıma yaklaştığı gösterilmiştir. Burada elde edilen sonuçlara göre, periyot $P = P_1 P_2 2^k$ ve doğrusal karmaşıklık $(r_1 + r_2)2^{k-1} < L \le (r_1 + r_2)2^k$ olur. Çıktı desenleri için $LFSR_1$ ve $LFSR_2$ geribesleme polinomlarının ilkle olması ve çıktı desen uzunluğunun $\min(r_1, r_2)$ 2'den küçük olması gerekmektedir. Benzer sonuçların periyodu $P_1 P_2$'yle aralarında asal periyotlu, ilkel polinoma sahip bir $LFSR_3$ ile de geçerli olacağı beklenmektedir. Günther(1987)'de $LFSR_3$'ün ilk durumunun bir böl-fethet saldırısı yoluyla elde edilebileceği gösterilmiştir. Diğer bir deyişle, $LFSR_3$ ilk durumu doğrudur ancak ve ancak çıktı dizisinin ilk türevi $LFSR_1$ ve $LFSR_2$ dizilerinin ikisinin de ilk türevlerine eşitse. Bu durumda, $LFSR_1$ ve $LFSR_2$ dizileri Berlekamp-Massey algoritması ile doğrusal karmaşıklığı denenir.

LFSR1 ve LFSR2 için bunun mümkün olup olmadığını araştırmak amacıyla hafizasız birleştiriciler için geliştirilmiş Golić ve Mihaljević (1991)'de anlatılan edit uzaklığı ya da Golic ve Petrovic(1993)'de anlatılan edit olasılığı yaklaşımlarını dur/ilerle üreteçlerine uyarlamaya çalışalım. Önce değişmeli adım üreteci ile ilgili varsayımlarımızı ortaya koyalım. $LFSR_1$ ve $LFSR_2$'nin dereceleri sırayla r_1 ve r_2 olan farklı ilkel geribesleme polinomları olduğunu ve aralarında asal periyotlarının sırasıyla P_1 ve P_2 olduğunu varsayalım. Her bir adımda, sadece bir LFSR adımlanır ve çıktı bitinin ilerle-sonra-topla şeklinde elde edilir.

6.2 Edit Uzaklığı

Edit uzaklığı tanımını yapmadan önce edit tanımını verelim.

Tanım: Herhangi bir alfabe üzerinde bir karakter dizisi verilsin. Aşağıdaki işlemlere *edit (düzenleme)* adı verilir:

Ekleme (insertion): Diziyi bir (veya birkaç) karakter eklenmesi.

Silme (deletion): Diziden bir (veya birkaç) karakter silinmesi.

Değiştirme (substitution): Dizideki bir karakterin başka bir karakterle değiştirilmesi.

Tanım: A ve B karakter dizileri arasındaki edit uzaklığı birinden diğerine geçmek için kullanılabilecek en az edit sayısıdır.

İddia: Edit uzaklığı bir uzaklık fonksiyonudur.

İspat: (Negatif olmama) Bir A karakter dizisi, herhangi bir düzenleme yapılmadan kendisini verir. Bu yüzden $d(A,A)=0$ olur. Farklı iki karakter dizisi arasındaki uzaklık sıfır olamayacağı için pozitiftir. Burada şu soru akla gelebilir. Acaba yukarıdaki düzenlemelerle bir karakter dizisinden diğerine geçilebilir mi? Ayrıca bu geçiş sayısı minimal yapılabilir mi? Dönüşümün mümkün olduğunu göstermek için $m \leq n$ olmak üzere m uzunluğunda bir A karakter dizisi ve n uzunluğunda bir B karakter dizisi alalım. A dizisinin her karakteri B dizisinin aynı sıradaki karakteriyle değiştirilsin, geri kalan B dizisi karakterleri aynı sırayla eklensin. B dizisi elde edilmiş olur. $m > n$ durumunda ise sondan karakter silerek ve aynı şekilde değiştirme yaparak A'dan B'yi elde edebiliriz. Böylece değişiklik sayısı için bir üst sınır bulmuş olduk. Simetri özelliğinin ispatı ile birlikte edit uzaklığı tanımındaki "birinden diğerini" ifadesi de sonlu diziler için anlamlı olur. Üst sınır bulunduktan sonra minimum bulunabilir. Sonsuz diziler için de ispat benzer bir şekilde yapılabilir. Dikkat edilmesi gereken bir nokta ise sonsuz dizilerde uzaklığın sonsuz olabileceğidir.

(Simetri) Aşağıdaki gözlemler sonucu iki karakter dizisi arasındaki bir dönüşüm dizisinden aynı uzunlukta bir dönüşüm dizisi ile geri dönülebilir.

Gözlem: Ekleme ile silme birbirinin tersidir. Bir karakter eklenerek elde edilen farklı B dizisinden geriye o karakteri silinerek dönülebilir. Tersi de doğrudur.

Gözlem: Değiştirme kendisinin tersidir. Bir karakter değiştirilerek elde edilen farklı dizisinden geriye o karakteri değiştirilerek dönülebilir. Tersi de doğrudur.

A dizisinden B dizisine geçerken izlenen herhangi bir yoldan aynı sayıda düzeltme yapılarak geri dönülebileceği için $d(A,B)=d(B,A)$ olur. Tanımdaki birinden diğerine ifadesi de anlamlı olur.

(Üçgen Eşitsizliği) A, B ve C karakter dizilerini ele alalım. A'dan C'ye giderken (uzaklık tanımı açısından) "en kötü ihtimalle" B'ye uğrayabiliriz. Bu nedenle, üçgen eşitsizliği $d(A,C) \le d(A,B)+d(B,C)$ sağlanır.

Edit uzaklığı bilgisayar mühendisliğinde (yanlış yazılmış olabilecek bir kelimenin doğrusunu bulma gibi problemlerde) ve genetik biliminde (mutasyon problemleri) çok kullanılan bir fonksiyondur. (Yukarıdaki şekliyle tanımlanan edit uzaklığına Levensthein uzaklığı da denilmektedir.)

Korelasyon saldırısı için farklı bir edit uzaklığı tanımlayacağız. Hatta klasik anlamda bir uzaklı fonksiyonu olarak dahi kabul edilebilirliği tartışılabilirse de (örneğin 3 değişkenli olması) bu farklı tanımın tamamen tutarsız olmadığı anlaşılacaktır. Farklı tanıma geçmeden önce ikili dizilerle uğraşacağımızdan ve edit uzaklığı tanımını değişmeli adım üreteci için uyarlayacağından bazı hatırlatmalar ve tanımlar yapalım.

$X^{n+1}=x_1,x_2,\ldots,x_{n+1}$ ve $Y^{n+1}=y_1,y_2,\ldots,y_{n+1}$ ikili iki girdi dizisi ve $Z^n=z_1,z_2,\ldots,z_n$ ikili bir çıktı dizisi olsun. $C^n=c_1,c_2,\ldots,c_n$ ikili bir kontrol dizisi ve $\hat{Z}^n=\hat{z}_1,\hat{z}_2,\ldots,\hat{z}_n$ ikili dizisi ise C^n'ye uyularak ilerle-sonra-topla değişmeli adım üreteci ile X^{n+} ve Y^{n+1} dizilerinden üretilen dizi olsun. (X^{n+1} ve Y^{n+1} dizilerinin $n+1$ uzunluklu LFSR_1 ve LFSR_2 dizilerine tekabül ettiğine dikkat ediniz.) Buna göre $c_1=0$ ise $\hat{z}_1=x_1\oplus y_2$ ve $c_1=1$ ise $\hat{z}_1=x_2\oplus y_1$ olur. Daha genel olarak, l sayısı C^s dizisindeki 1 lerin sayısı olmak üzere, $1\le s\le n-1$ ve $0\le l\le s$ için $\hat{z}_s=x_{l+1}\oplus y_{s+1-l}$ olur. Eğer $c_{s+1}=0$ ise $\hat{z}_{s+1}=x_{l+1}\oplus y_{s+2-l}$ ve $c_{s+1}=1$ ise $\hat{z}_{s+1}=x_{l+2}\oplus y_{s+1-l}$ olur.

Tanım: Verilen $\left(X^{n+1},Y^{n+1}\right)$ dizi ikilisi ve Z^n dizisi için $D\left(X^{n+1},Y^{n+1};Z^n\right)$ şeklinde gösterilen edit uzaklığı

$$D\left(X^{n+1},Y^{n+1};Z^n\right)=\min_{C^n\in\{0,1\}^n} d_H\left(Z^n,\hat{Z}^n\right) \tag{6.1}$$

şeklinde tanımlanır. $d_H(\,)$ fonksiyonu Hamming uzaklığını temsil etmektedir. Diğer bir deyişle, C^n bütün ikili kontrol dizilerini gezmek üzere, edit uzaklığı Z^n dizisinden $\hat{Z}^n$

dizisini elde edebilmek için gerekli "etkili" değiştirme minimum sayısıdır. "Etkili" kelimesinin sebebi (X^{n+1}, Y^{n+1}) girdi dizilerinden Z^n dizisi elde edilirken, C^n kontrol dizisinden bağımsız olarak, en baştaki girdi bitlerinden biri silineceğinden ve girdi bitlerinin tam $n-1$ adet tekrarını işlemlerini içermektedir. Dolayısıyla, düzenleme işlemleri için esas bilgi taşıyan kısmı "etkili" değiştirmelerdir.

C^n üzerinde bütün uzayı tarayarak 2^n adımda edit uzaklığını hesaplayabiliriz. Fakat bu karmaşıklık oldukça büyüktür. Doğal olarak bu yeni edit uzaklığının işlemsel karmaşıklık açıdan yeterince küçük bir algoritma ile hesaplanıp hesaplanmadığını kontrol edelim. Bu amaçla bir tanım yapalım.

Tanım: Herhangi bir $1 \le s \le n$ ve $0 \le l \le s$ için $W(l,s)$ *kısmi edit uzaklığı*, ikili kontrol dizisi C^s dizisinin tam olarak l adet 1 içerdiği varsayımıyla, $D(X^{s+1}, Y^{s+1}; Z^s)$ olarak tanımlanır. Diğer bir deyişle, $w_H(C^s)$, C^s dizisinin Hamming ağırlığını temsil etmek üzere, $W(l,s) = \min_{C^s : w_H(C^s)=l} d_H(Z^s, \hat{Z}^s)$ olur. $\hat{z}_s = x_{l+1} \oplus y_{s+1-l}$ olduğundan son bit her zaman x_{l+1} ve y_{s+1-l} girdi bitlerinden elde edilir. Bu yüzden edit dönüşümlerini sadece X^{l+1} ve Y^{s+1-l} önekleri ilgilendirir. Kısmi edit uzaklığını Hamming uzaklıklarını ikili değişkenlerin tamsayı toplamı olarak göstererek

$$W(l,s) = \min_{C^s : w_H(C^s)=l} \sum_{k=1}^{s} (z_k \oplus \hat{z}_k) \tag{6.2}$$

şeklinde de tanımlayabiliriz.

Kısmi edit uzaklıklarının yinelemeli bir özelliğini kullanarak edit uzaklığını hızlı bir şekilde hesaplayabiliriz. Bunun için aşağıdaki teorem kullanılacaktır.

Teorem: Herhangi bir X^{n+1}, Y^{n+1} ve Z^n için $W(l,n)$ kısmi edit uzaklığını göstermek üzere.

$$D\left(X^{n+1},Y^{n+1};Z^n\right)=\min_{0\le l\le n}W(l,n) \tag{6.3}$$

olarak hesaplanabilir. Kısmi edit uzaklığı $W(l,n)$ ise yinelemeli olarak başlangıç koşulları $W(-1,s)=W(s+1,s)=\infty$ ve $W(0,0)=0$ olmak üzere $1\le s\le n$ ve $0\le l\le s$ için

$$W(l,s)=\left(x_{l+1}\oplus y_{s+1-l}\oplus z_s\right)+\min\left(W\left(l-1,s-1\right),W\left(l,s-1\right)\right) \tag{6.4}$$

formülüyle hesaplanabilir.

İspat: Kısmi edit uzaklığı tanımından $W(l,n)=\min_{C^n:w_H(C^n)=l} d_H(Z^n,\hat{Z}^n)$ olur. $\min_{0\le l\le n}$ ile $\min_{C^s:w_H(C^s)=l}$ ardı ardına geldiklerinde $\min_{C^s\in\{0,1\}^n}$ sağlanacağından (6.3) eşitliği gösterilmiş olur.

İkinci eşitliği göstermek için önce $s=1$ alalım. $W(0,1)$ durumunda $C^s=C^1$'de $w_H(\ )=0$ olan kontrol dizisi $\{0\}$ olacağından (kontrol biti 0 olacağından) $W(0,1)=x_1\oplus y_2\oplus z_1$ olur. Benzer bir şekilde $W(1,1)=x_2\oplus y_1\oplus z_1$ olur.

Şimdi $s>1$ alalım. Tanım gereği $\hat{z}_s=x_{l+1}\oplus y_{s+1-l}$ olduğundan toplamdaki son terim olan $x_{l+1}\oplus y_{s+1-l}\oplus z_s$ toplamını dışarı alalım. (6.2) denkleminde uzunluğu $s-1$ olan kontrol dizileri üzerinde c_s in 0 olması durumunda ağırlığı l , c_s in 1 olması durumunda ağırlığı $l-1$ olan kontrol dizilerinde minimum almak gerekecektir. Bu durumda kısmi edit uzaklığını

$$W(l,s)=(x_{l+1}\oplus y_{s+1-l}\oplus z_s)+\min\left(\min_{C^{s-1}:w_H(C^{s-1})=l-1}\sum_{k=1}^{s-1}(z_k\oplus\hat{z}_k),\min_{C^{s-1}:w_H(C^{s-1})=l}\sum_{k=1}^{s-1}(z_k\oplus\hat{z}_k)\right) \tag{6.5}$$

olacaktır. min parantezi içindeki kısmi edit uzaklığı tanımından $W(l-1,s-1)$ ve $W(l,s-1)$ olduğundan (6.5) eşitliği kanıtlanmış olur.

Teorem 1'de kullanılacak yinelemeli algoritmanın zaman karmaşıklığı $O(n^2)$ mertebesinde olacaktır. (Paskal üçgeninde bir terimi bulma gibi düşünülürse) Hafıza karmaşıklığı ise s'in şimdiki ve bir önceki değeri için kısmi edit uzaklığı değerlerini saklayacağından $O(n)$ mertebesindedir. Görüldüğü gibi n çok büyük olduğu takdirde bile, algoritma çalıştırılabilir.

Algoritmada $O(n^2)$ lik bir hafıza artışı ile s ve l ile indislenen kısmi edit uzaklıklarının tamamının matrisi ile birlikte (5) eşitliğini elde edilirken kullanılan c_s kontrol bitlerini depolanabilir. Matristen geriye giderek, edit uzaklığını temsil eden minimum sayıda etkili dönüşüme yol açan olası bütün kontrol dizilerini elde edebiliriz.

Aşağıda edit uzaklığı ile ilgili bazı temel simetri özellikleri verilmektedir:

Herhangi bir $A = a_1, a_2, a_3, a_4, \ldots$ karakter dizisi alındığında $\overline{a} = a \oplus 1$ (a'nın tümleyeni) olmak şartıyla $\overline{A} = \overline{a}_1, \overline{a}_2, \overline{a}_3, \overline{a}_4, \ldots$, $\widetilde{A} = a_1, \overline{a}_2, a_3, \overline{a}_4, \ldots$ ve $\widetilde{\overline{A}} = \overline{a}_1, a_2, \overline{a}_3, \overline{a}_4, \ldots$ olsun.

Önerme: $D(X^{n+1}, Y^{n+1}; Z^n) = D(\overline{X}^{n+1}, \overline{Y}^{n+1}; Z^n) = D\left(\overline{X}^{n+1}, Y^{n+1}; \overline{Z}^n\right)$.

İspat: Bir kontrol dizisi C^n verildiğinde, şayet $\hat{Z}^n$ dizisi (X^{n+1}, Y^{n+1}) çiftinden üretilmişse tümleyenlerin toplamı yine aynı olacağından $\hat{Z}^n$ aynı zamanda $(\overline{X}^{n+1}, \overline{Y}^{n+1})$ çifti tarafından da üretilir. X^{n+1}'in tümleyeni alınırsa toplam da $\hat{Z}^n$ dizisinin tümleyeni olacağından önerme ispatlanmış olur.

Önerme: $D(X^{n+1}, Y^{n+1}; Z^n) = D(\widetilde{X}^{n+1}, \widetilde{\overline{Y}}^{n+1}; \widetilde{Z}^n)$.

İspat: Bir kontrol dizisi C^n verildiğinde, şayet $\hat{Z}^n$ dizisi (X^{n+1}, Y^{n+1}) çiftinden üretilmiş olsun. Önce z_1'in aynı kaldığını gösterelim. Kontrol bitinin durumuna göre $\hat{z}_1 = x_1 \oplus y_2$ ya da $\hat{z}_1 = x_2 \oplus y_1$ oluyordu. $(\overline{X}^{n+1}, \widetilde{\overline{Y}}^{n+1})$ dizisinin üreteceği çıktı dizisini Z' ile gösterelim. Z' dizisinin ilk biti z'_1 ya $x_1 \oplus y_2$ olacak ya da $\overline{x}_2 \oplus \overline{y}_1$ olur. Böylece ilk bit değişmemiş olur. Yeni çıktı dizisinde $\hat{z}_s = x_{l+1} \oplus y_s$ korunuyor olsun($z'_s = \widetilde{x}_{l+1} \oplus \widetilde{\overline{y}}_{s+1-l} = \hat{z}_s$ olsun). Aynı kontrol dizisi altında:

- Ya $\hat{z}_{s+1} = x_{l+1} \oplus y_{s+2-l}$ ve $z'_{s+1} = \widetilde{x}_{l+1} \oplus \widetilde{\overline{y}}_{s+2-l}$ olur. $(\hat{z}_s \oplus z'_s) \oplus (\hat{z}_{s+1} \oplus z'_{s+1})$ toplamını hesaplarsak x li terimler her parantez içinde birer, toplamda ikişer kez geçtiğinden sadeleşecek ve elimizde $(y_{s+1-l} \oplus \widetilde{\overline{y}}_{s+1-l}) \oplus (y_{s+2-l} \oplus \widetilde{\overline{y}}_{s+2-l})$ toplamı kalacaktır. Y dizisi alterne değiştiğinden parantez içindeki toplamlardan biri 0, diğeri 1 olacaktır. $z'_s = \hat{z}_s$ eşitliği kabul edildiğinden , $\hat{z}_{s+1} \oplus z'_{s+1} = 1$ sonucunu elde ederiz. Diğer bir deyişle, z'_{s+1}'in $\hat{z}_s$ den farklı olduğunu görürüz.

- Ya da $\hat{z}_{s+1} = x_{l+2} \oplus y_{s+1-l}$ ve $z'_{s+1} = \tilde{x}_{l+2} \oplus \tilde{\tilde{y}}_{s+1-l}$ olur. $(\hat{z}_s \oplus z'_s) \oplus (\hat{z}_{s+1} \oplus z'_{s+1})$ toplamını hesaplarsak y li terimler her parantez içinde birer, toplamda ikişer kez geçtiğinden sadeleşir. Yukarıdakine benzer bir şekilde X dizisi alterne değiştiğinden z'_{s+1}'in $\hat{z}_s$ den farklı olduğunu görürüz.

Benzer bir şekilde $s \geq 2$ olmak üzere $z'_s = \hat{z}_s \oplus 1$ için $z'_{s+1} = \hat{z}_{s+1}$ olduğu gösterilebilir. Bu şekilde Z' çıktı dizisinin ilk terimi Z dizisinin ilk terimi ile aynı, sonraki terimleri bir aynı,bir farklı olarak giden bir dizi, diğer bir deyişle, $\tilde{Z}_n$ elde edilmiş olur.

Çizelge 6.1 Tesadüfi seçilmiş 1000 (X^{n+1}, Y^{n+1}, Z^n) üçlüsü için D istatistikleri

Dizi uzunluğu	10	20	30	40	50	60	70	80	90	100
Mimimum	0	0	0	0	1	2	2	3	4	5
Maksimum	5	6	9	10	11	12	14	15	16	18
Ortalama	1.611	2.783	3.973	5.125	6.147	7.243	8.293	9.336	10.393	11.542
Medyan	2	3	4	5	6	7	8	9	10	12
St. Sapma	.996	1.194	1.373	1.508	1.570	1.715	1.822	1.859	2.002	1.917

Golic ve Menocci(1997)

Çizelge 6.2 Tesadüfi seçilmiş 1000 (X^{n+1}, Y^{n+1}, Z^n) üçlüsü için D istatistikleri (devam)

Dizi uzunluğu	200	300	400	500	600	700	800	900	1000
Mimimum	13	24	32	41	51	59	70	77	81
Maksimum	29	40	56	63	72	86	96	104	117
Ortalama	21.593	31.918	42.016	52.071	61.932	72.518	81.925	92.035	102.15
Medyan	22	32	42	52	62	72	82	92	102
St. Sapma	2.626	2.823	3.137	3.550	3.707	3.767	3.942	4.304	4.562

Golic ve Menocci(1997)

6.3 Gömme Olasılıkları

$P_n(D|Z^n)$ olasılığı, Z^n sabit ve (X^{n+1},Y^{n+1}) çifti düzgün olasılık dağılımına sahip iken $D(X^{n+1},Y^{n+1};Z^n)$ edit uzaklığının D'ye eşit olama olasılığını göstersin. $P_n(D)$ ise düzgün dağılımlı bir Z^n üzerinde $P_n(D|Z^n)$'in beklenen değeri, yani $(X^{n+1},Y^{n+1};Z^n)$ düzgün olasılık dağılımlı diziler arasından rasgele seçildiğinde edit uzaklığının D'ye eşit olması olasılığını belirtsin.

Tanım : Özel olarak edit uzaklığının 0 olma olasılığı $P_n(0|Z^n)$'yi $P_n(Z^n)$ ile gösterelim. n uzunluğunda Z^n çıktı dizisi verildiğinde sıfır edit uzaklığı, Z^n dizisi (X^{n+1},Y^{n+1}) ikilisinden ilerle-sonra-topla değişmeli adım üretecinden oluşturulabilecek şekilde bir kontrol dizisi bulunduğu anlamına geldiğinden $P_n(Z^n)$ olasılığına *gömme olasılığı(embedding probability)* denir.

J. Dj. Golić, L. O'Connor ile birlikte yazdığı makalelerinde zaman kontrollü ötelemeli yazıcılar için yeni bir korelasyon saldırısı ortaya atarken bu kavramdan yararlanmıştır. Daha sonra kendisinin tek başına yazdığı makalede(1996) ise bu kavramı matematiksel açıdan daha detaylı incelemiştir.

Tanım : Düzgün dağılımlı bir Z^n üzerinde $P_n(Z^n)$ olasılığının ortalama, maksimum ve minimum değerlerini sırasıyla $\overline{P}_n$, $P_n^{\max}$ ve $P_n^{\min}$ ile gösterelim.

Edit uzaklığı korelasyon saldırısının başarısı yukarıdaki olasılıkların hepsine bağlıdır. Ortalama değer $\overline{P}_n$'nin n ile azalması hayatidir ve bu azalmanın üssel hızda olması arzu edilir. Bunun yanında $P_n^{\max}$ olasılığının da benzer bir davranış göstermesi iyi olur.

Yukarıdaki gömme olasılıklarının hesaplanması zor bir kombinatorik problem olarak görünmektedir. Çıktı dizisi uzunluğu n ile birlikte istenilen üssel azalma hiç de alani olarak görünmemektedir. Bu yüzden küçük n değerleri için tüm uzay sayımı yapılmış ve daha büyük n ler için uygun girdi ve çıktı dizileri örneklemesi ile değerler tahmin edilmeye çalışılmıştır.

Bu sonuçlara göre $\overline{P_n}$, P_n^{max} ve P_n^{min} olasılıkları büyük n için $b<1$ olmak üzere ab^n şeklinde gibi görünmektedir. Yapılan deneyler sonucu $\overline{P_n} \approx 0,830\ ,83^{\ n}$, $P_n^{max} \approx 0,72 \cdot 0,915^n$ ve $P_n^{min} \approx 2,7 \cdot 0,562^n$ yaklaştırmaları bulunmuştur.

6.4 Korelasyon Saldırısı

$LFSR_1$ ve $LFSR_2$ geribesleme polinomlarının bilindiğini varsayalım. Edit uzaklığı korelasyon saldırısının amacı, bilinen açık-mesaj saldırısı ile yeterince uzun çıktı dizisi kesitinden faydalanarak bu LFSR'ların ilk halini oluşturabilmektir. Çıktı dizisi ile düzenli ilerletilmiş $LFSR_1$ ve LFSR2 dizileri arasındaki istatistiksel bağımlılık veya korelasyon olarak edit uzaklığı kullanılmaktadır.

Eğer bilinmeyen kontrol dizisi tamamen rassal kabul edilirse, diğer bir deyişle, düzgün dağılımlı ikili tesadüfi değişken dizisi ise, girdi LFSR'larından ortalama $n/2+1$, maksimum $n+1$ uzunlukta kesit kullanılacaktır.

Tahmini bir LFSR1 ve LFSR2 ilk durumundan, geribesleme polinomu bilindiğinden dolayı, $n+1$ uzunluklu X^{n+1} ve Y^{n+1} dizileri oluşturulabilir. Daha sonra $D\left(X^{n+1}, Y^{n+1}; Z^n\right)$ edit uzaklığı yinelemeli kısmi edit uzaklığı algoritması sayesinde kolayca hesaplanabilir. Bu işlem olası bütün (toplam $2^{n_1+n_2}$ adet) LFSR1 ve LFSR2 ilk durumları için tekrarlanır. Gerçek LFSR ilk durumu için edit uzaklığı sıfır olacağından, edit uzaklığı sıfır olan ilk durum ikilileri gerçek ilk durum adayları olurlar.

Kontrollü tek ötelemeli yazıcı durumunda Levenshtein edit uzaklığı (Golić Mihaljević) gömme saldırısına dönüştüğü gibi bu edit uzaklığı saldırısını da özel bir gömme saldırısı olarak görebiliriz.

6.5 Teorik Analiz

Saldırıdan ideal olarak beklenen yeterince uzun n için, LFSR! Ve LFSR2 ilk durumları için tek aday kalmasıdır. Bu ise ancak gömme olasılığı $P_n(Z^n)$ yeterince küçük ise olur. $P_n(Z^n)$ olasılığı $D(X^{n+1},Y^{n+1};Z^n)=0$ olan girdi dizisi çiftlerinin bütün (X^{n+1},Y^{n+1}) dizileri içindeki oranı olduğundan, aday sayısının beklenen değeri $2^{r_1+r_2}P_n(Z^n)$ olur. Saldırının başarılı sayılabilmesi için $2^{r_1+r_2}P_n(Z^n)\le 1$ olmalıdır. $\overline{P}_n$, $P_n^{\max}$ ve $P_n^{\min}$ bu eşitsizlikte $P_n(Z^n)$ yerine konulursa algoritmanın başarılı sayılabilmesi için gerekli n uzunluğunun sırasıyla ortalama, en kötümser ve en iyimser tahminlerini buluruz. Eğer bu olasılıklar gerçekten de ab^n şeklindeyse $n\ge\frac{r_1+r_2+\log a}{-\log b}$ olmalıdır. Bu ise gerekli çıktı dizisi uzunluğunun $LFSR_1$ ve $LFSR_2$ uzunlukları toplamı cinsinden doğrusal olduğunu gösterir. Ortalama, en kötümser ve en iyimser gerekli uzunlukları $LFSR_1$ ve $LFSR_2$ toplamı $r=r_1+r_r$ cinsinden $n\ge 3.72\,r$, $n\ge 7.8r$ ve $n\ge 1.2r$ bulunur.

Çıktı dizisi uzunluğu n arttıkça ilk durum çiftleri aday sayısının 1'e inmesini beklerken dizilerin rassal olduğu varsayımını yapmıştık. Ne var ki, dizilerin arasındaki lineer ilişkiler nedeniyle aday sayısı n uzunluğundan bağımsız olarak en iyi ihtimalle 2'ye düşürülebilir. Aşağıdaki önerme bunu ifade etmektedir.

Önerme: $n+1>\max(r_1,r_2)$ şartı sağlanmak suretiyle sıfır edit uzaklığı ölçütüne göre seçilen $LFSR_1$ ve $LFSR_2$ ilk durum adaylarının sayısı en an ikiye eşittir.

İspat: $c_1=1$ olmak üzere $C^n=c_1,c_2,\ldots,c_n$ kontrol dizisinin (X^{n+1},Y^{n+1}) çiftini Z^n'ye dönüştürdüğünü varsayalım. Ayrıca, $n+1>\max(r_1,r_2)$ olmak üzere, X^{n+1} ve Y^{n+1} dizilerinin ilk durumu $X^{r_1}=x_1,x_2,\ldots,x_{r_1}$ olan $LFSR_1$ ve ilk durumu $Y^{r_2}=y_1,y_2,\ldots,y_{r_2}$ olan $LFSR_2$ tarafından üretilmiş olsun. y_0 $LFSR_2$'nin bir geriye çalıştırılması ile elde edilen bit olmak üzere , $\hat{y}_i=y_{i-1}\ i=1,2,\ldots,r_2$ olan $\hat{Y}^{n+1}$ dizisi ve $\hat{x}_i=x_{i+1}, i=1,2,\ldots,r_1$ olan $\hat{Y}^{n+1}$ dizisi $\hat{c}_1=0$ ve $\hat{c}_i=c_i, i=1,2,\ldots,n$ olan $\hat{C}^n$ dizisi ile kontrollendiğinde $D(\hat{X}^{n+1},\hat{Y}^{n+1};Z^n)=0$ buluruz. Kısacası, ilk adımı hangi LFSR'ın attığına göre dizilerin terimlerini hafif bir ayarlamayla aynı çıktı dizisini veren bir ilk durum çifti bulunmuş olur. İlk başta kabulümüz $c_1=0$ idi fakat sonuçta bulduğumuz kontrol dizisinde $\hat{c}_1=0$ olduğu gözükmektedir. Bu da

$c_1 = 0$ olması durumunda kontrol dizilerinin rolleri değiştirilerek ispatın tamamlanabileceğini göstermektedir.

Elde edilen her bir ilk durum adayı çifti için, kısmi edit uzaklıkları matrisi depolanarak ve bu matriste geriye doğru gidilerek sıfır edit uzaklığı veren olası bütün kontrol dizilerini ele geçirebiliriz.

Elde edilen her ilk durum adayı için, kısmi edit uzaklıkları matrisi depolanabilir. Matriste geriye giderek sıfır edit uzaklığını veren olası bütün kontrol dizileri C^n bulunabilir. Aday ilk durum çifti için ortalama çift sayısı $m_n = \frac{2^n}{2^n \overline{P}_n} \approx 1{,}2 \cdot 2^{0{,}269^n}$ ile tahmini olarak hesaplanabilir.Eğer n, $2^{r_1+r_2}\overline{P}_n \approx 1$ olacak şekilde seçilirse, o zaman $m_n \approx 2^{r_1+r_2}$ mertebesinde olacaktır. Eğer kontrol dizisi uzunluğu $r_3 < n$ olan bir $LFSR_3$ tarafından üretiliyorsa, C^n dizilerinin sayısı kısmi edit uzaklıkları matrisinde geriye gidilerek her seferinde, LFSR denkleminde bir bitin LFSR geribesleme denklemini sağlayıp sağlamadığı kontrol edilerek, azaltılabilir. İncelenen her bitin "hayatta kalan" kontrol dizisi sayısını yarıya indirmesi beklenir. Bu yüzden eğer $n - r_3 \geq \log m_n$ ise ,ya da diğer bir deyişle, yaklaşık olarak $n \geq 1.37 r_3$ ise, hayatta kalan kontrol dizisi sayısının tek olması beklenir.

Bu iki sonucu birleştirerek ($LFSR_1$ ve $LFSR_2$ için $n \approx 3{,}72(r_1 + r_2)$ için "tek" bir aday durum çifti kalması ve $n \geq 1{,}37 r_3$ için tek bir aday kontrol dizisi kalması), $O\left(2^{r_1+r_2}\right)$ mertebede bir saldırı ile doğru ilk durum aday üçlüsü $(X^{r_1}, Y^{r_2}, C^{r_3})$ sayısını sadece birkaç(muhtemelen sadece bir) adede indirebiliriz. Doğru olanı bulmak için değişmeli adım üretecinde elde edilen diziler denenerek kontrol edilir. Değişmeli adım üretecinin yapısından dolayı, yeterince uzun bir çıktı dizisi alındığında farklı LFSR ilk durumlarının farklı çıktıları vereceği varsayımıyla, elde edilen LFSR ilk durumlarının doğru olduklarını varsayabiliriz.

6.6 Deneysel Saldırılar

Anlatılan edit uzaklığı saldırısının bilgisayarda denemeleri yapıldı. Kontrol dizisinin de boyu r_3 olan bir $LFSR_3$ olduğu varsayıldı. Bütün LFSR geribesleme polinomlarının bilindiği ve ilkel olduğu varsayıldı. Saldırıda, yeterince uzun değişmeli adım üreteci çıktısından yararlanılarak $LFSR_1$, $LFSR_2$ ve $LFSR_3$'ün ilk durumları oluşturulmaya çalışıldı. Aşağıdaki çizelgede n, çıktı uzunluğunu, $N_{1,2}$, sıfır edit uzaklığı kriterini sağlayan $LFSR_1$ ve $LFSR_2$ aday sayısını, N_3 ise $N_{1,2}$ adaylarına karşılık gelen kontrol dizileridir. 1 ile 1', $N_{1,2}$ olabilecek minimum değerdeyken çıktı dizisi daha uzun alınarak N_3'ün teke indirilebileceğini göstermek için yapılmıştır. Detaylar için Golic'in makalelerine bakınız.

7. KISALTILMIŞ ÜRETEÇ

Kısaltan üreteç, düzenli olarak çalıştırılan iki LFSR'dan oluşur. Bunları $LFSR_A$ ve $LFSR_S$ ile gösterelim. Üretecin çıktısı, $LFSR_A$ çıktısının "kısaltılmış" bir hali olduğundan bu ad verilmiştir. $LFSR_S$, $LFSR_A$'nın çıktısının alınıp alınmayacağını kontrol eder. Diğer bir deyişle, iki LFSR çalıştırılır. Eğer . $LFSR_S$ çıktısı 1 ise, $LFSR_A$ çıktısı üreteç çıktısı olarak alınır, 0 ise alınmaz. Dolayısıyla, üreteç çıktısını, $\{a_i\}$ dizisinin $\{s_i\}$ dizisinin 0 olduğu yerleri silinmiş ya da kısaltılmış hali olarak düşünebiliriz. Daha matematiksel bir anlatımla : Üreteç çıktısını $z=\{z_k\}_{k=1}^{\infty}$, $LFSR_S$ çıktısını $s=\{s_i\}_{i=1}^{\infty}$ ve $LFSR_A$ çıktısını $a=\{a_i\}_{i=1}^{\infty}$ ile gösterirsek; i_k, s dizisinde k ıncı 1 olmak üzere, $z_k=a_{i_k}$ olur.

Eğer LFSR geribesleme polinomları ilkel ise, o zaman a ve s dizileri sırasıyla periyodları $2^{r_a}-1$ ve $2^{r_s}-1$ olan maksimum uzunluklu diziler olur. Üstelik, r_a ve r_s aralarında asal ise, z dizisinin periyodunun $(2^{r_a}-1)2^{r_s-1}$ ve doğrusal karmaşıklığın $r_A 2^{r_s-2} < DK \le r_A 2^{r_s-1}$ olacağı gösterilebilir. Üretecin periyod analizi sırasında bazı farklı anahtarların aynı çıktı dizisini üreteceği ortaya çıkar.Bu anahtar uzayını tarama miktarını $(2^{r_a}-1)2^{r_s-1}$ mertebesine düşürse de, bunun yeterli olmadığı açıktır. Saldırının ayrıntılarına geçmeden önce kullanacağımız bazı gösterimleri verelim. $A=a_1,a_2,\ldots$ dizisinde k ıncı terimden itibaren başlayan altdiziyi $A_k=a_k,a_{k+1},\ldots$ ile , dizinin ilk n terimini $A^n=\{a_i\}_{i=1}^{n} a_1,a_2,\ldots,a_n$ ile gösterelim.

Aynı uzunluktaki iki dizi arasındaki korelasyonu ölçmek için Hamming uzaklığı kullanılabilir. Kısaltılmış üreteçte ise, üreteç çıktısı $LFSR_A$ çıktısının bir alt dizisi olduğundan yani farklı uzunluklarda olduklarından Hamming uzaklığını kullanamayız. Uzunluğu m olan X^m dizisi ile uzunluğu $n \le m$ olan Y^n dizisi arasındaki "birleşik" olasılığı korelasyon ölçütü olarak kullanacağız. Bu olasılığın geçerliliğini yapılan deney sonuçlarıyla göstereceğiz. Bu olasılığı $P(X^m,Y^n)$ ile göstereceğiz. Bu olasılık bize X^m dizisinden Y^n dizisinin çeşitli değişikliklerle(ekleme, çıkarma ve karakter değiştirme) elde edilmesi olasılığını temsil etmektedir. Bu olasılık, birazdan tanımlayacağımız kısmi birleşik olasılık yoluyla adım adım hesaplanabilir. $X^{e+s}=\{x_i\}_{i=1}^{e+s}$, X^m dizisinin $e+s$ uzunluğundaki öneki, $Y^s=\{y_i\}_{i=1}^{s}$ ise Y^n dizisinin s uzunluklu öneki olsun. $1 \le s \le n$ ve $0 \le e \le m-n$ olmak üzere $P(X^{e+s},Y^s)$ olasılığını $P(e,s)$ ile gösterelim. $\delta(x,y)$ değiştirme olasılığını belirtsin, öyle ki eğer $x=y$

ise $\delta = 0.5$ ve diğer durumlarda $\delta = 0$ olsun. Kısmi olasılık $0 \le e \le m-n$ için $P(e,0) = p^e$ ve $1 \le s \le n$ için $P(-1,s) = 0$ ilk değerleri ile $P(e,s) = P(e-1,s)p + P(e,s-1)\delta(x_{e+s}, y_s)$ bağıntısını sağlar.

Dikkat edilirse, $P(X^m, Y^n) = P(m-n,n)$ olduğu görülür. Bu hesaplamanın işlemsel karmaşıklığı $O(n(m-n))$ mertebesindedir.

Ele alınan iki dizi arasındaki ilişki hakkında karar verebilmek için, gözönüne almamız gereken iki hipotez vardır:

- H_0 : *(sıfır hipotezi)* X^m ile Y^n birbirinden bağımsızdır.
- H_1 : X^m ile Y^n birbiriyle bağlantılıdır, diğer bir deyişle, Y^n dizisi X^m'den istatistiksel modelle elde edilmiştir.

Düzensiz çalıştırılan ötelemeli yazıcı sitemlerine korelasyon saldırısında, H_0 LFSR'ın ilk durumunun yanlış tahminine, H_1 ise doğru tahmine tekabül eder. Hipotez testinin dayandığı istatistik daha önce tanımlanan birleşik olasılıktır.

7.1 Birleşik Olsılığın Korelasyon Ölçüsü Olarak Geçerliliği

Kısaltılmış üreteçte $p = 0.5$ değeri için birleşik olasılığın, tamamen rassal bir dizi çifti ile birbiriyle bağlantılı dizi çifti arasındaki farkı anlayabilmede geçerli bir istatistk olup olmadığını anlamak için yapılan bilgisayare denemeleri yapıldı.[Golic,9] İki durum incelendi: İlk olarak, X^m dizisi ile Y^n nin birbirinden bağımsız olarak üretildikleri rassal durum (RAND), ikinci olarak Y^n dizisinin X^m den elde edildiği ilişkili durum (CORR) gözden gözlendi. $n = 150, 225$ ve 300 bit için $m(n) = n/(1-p) + 3\sqrt{n}$ olmak üzere çeşitli dizilere simülasyon uygulandı. Burada $m(n)$ sayısı, rasgele bir m^* dizisinin n uzunlukta bir çıktı dizisi vermesi olayının beklenen değeri olmak üzere, $m^* > m(n)$ olasılığı çok küçük olacak şekilde seçildi. Belirtilen her n uzunluğu için, her durumda 10000 dizi üretildi ve her çift için normalize edilmiş birleşik olasılık hesaplandı. Bu işlem, tekrarlı bağıntı ile üretilen olasılığın $4/(3(1-p))$ ile çarpılmış halidir. Bu yapılmadığı takdirde, büyük n sayıları için birleşik olasılık değerleri hızla sıfıra yaklaşır.

Sonuçlar, normalize edilmiş birleşik olasılıkların rassal ve ilişkili durumlarda birbirinden açıkça farklı olduklarını göstremektedir. Örneğin, $n = 150$ için, rassal durumda üretilen 10000 olasılık değerinin %75'i, ilişkili durumda elde edilen en küçük değer olan 1.9×10^{-27} değerinin altında çıktı. n arttıkça dağılımlar arsındaki fark da gittikçe daha belirgin olmaktadır. Yine ilişkili durumda elde edilen en küçük değer gözönüne alınırsa, $n = 225$ için rasgele durumda hesaplanan olasılık değerlerinin %90'ı bu değerin altında çıkarken $n = 300$ için bu oran %97.5'a çıkmaktadır. Bu sonuçlar, normalize edilmiş birleşik olasılığın farklı uzunluklardaki iki karakter dizisinin korelasyonu için geçerli bir ölçü olduğunu göstermektedir.

Hesaplanan birleşik olasılık değerini sıfır hipotezini test ederken yapılabilecek iki hata vardır: Birincisi, H_0'ı doğru olduğu halde reddetmek, yani diziler bağıntılı olmadığı halde öyle olduklarını iddia etmektir. Buna *"yanlış alarm"* diyelim. İkinci hata türü ise, H_1 doğru olduğu halde H_0'ı kabul etmek, yani diziler aslında biribiri ile ilişkili iken birbirinden bağımsız olduklarını iddia etmektir. Bu hataya da *"olayı kaçırmak"* diyelim. Sabit bir n için, P_y ile göstereceğimiz "yanlış alarm" tahmini değeri için kritik değerleri ve P_o ile göstereceğimiz "olayı kaçırmak" olasılığı için eşik değerler hesaplanabilir. Benzer bir şekilde,

hesaplanan bir P_o olasılığı için, farklı kayaranahtar uzunluğu için kritik değerler bulunup bu eşik değerleri için P_y hesaplanabilir.

Tablo sabit bir n uzunluğu için, P_o olasılığını düşürmek için, P_y'nin artırılması gerektiğini gösterir. Ne var ki, eğer dizi uzunluğu artırılırsa, hem P_o hem de P_y düşmüş olur.

7.2 Üretece Saldırının Uygulanması

LFSR'lara yapılan birçok saldırı gibi, bu saldırı da bir "böl-fethet" saldırısı türüdür. Diğer bir deyişle, LFSR'lardan biri düşük karmaşıklıkta elde edilip sonra diğierine saldırılacaktır.

7.2.1 $LFSR_a$ nın İlk Halinin Bulunması

Silme senkronizasyon hatalı iletişim kanalları için kapasite argümanını kullanarak, Golić ve Connor Eurocrypt94'de şunu iddia etmiştir : "Eğer gözlenene kayar anahtar dizisi, LFSR uzunluğu cinsinden doğrusal bir değerden daha büyük ise istatiksel optimal olasılıksal korelasyon saldırısı her zaman geçerlidir." Bu tezi test etmek için, değişik kayar anahtar uzunlukları için, saldırının ne kadar sıklıkla $LFSR_A$ ilk durumunu doğru olarak bulabildiği veya ilk halin maksimum olasılığa ne kadar yakın olduğunu ölçmek için deneyler yapıldı. Bu sav, kayaranahtar uzunluğunun artması halinde "yanlış alarm" olasılığının üssel olarak azaldığı gözlemiyle uyuşmaktadır. $LFSR_A$ ilk durumu için bulunan hatalı adayların beklenen değeri 1'den küçük ise saldırı başarılı diyebiliriz. Bu sayı yaklaşık olarak $2^{r_A} P_y(n)$ olduğundan, $P_y(n)$'nin ab^n üssel yazımı(a sabiti ihmal edilecek olursa) saldırının başarılı olması için $n \geq \frac{1}{-\log_2 b} r_A \approx 20.r_A$ olması gerektiğini gösterir. $LFSR_A$'ya kısıtlanmamış olasılıksal korelasyon saldırısını uygulayabilmek için, $\{z_i\}_{i=1}^{n}$ kayaranahtar dizisi parçası ve $LFSR_A$ geribesleme polinomunun bilinmesi gereklidir. $LFSR_s$'in detayları $LFSR_A$'nın her olası durumu için, geribesleme polinomu ile düzenli çalıştırılarak $m(n) \geq n$ olacak şekilde $m(n)$ uzunluğunda bir dizi parçası üretilir. Deneylerde, $c = 3$ olmak üzere $m(n) = n/(1-p) + c\sqrt{n}$ alınmıştır. Bu bit dizisi ve bilinen kayaranahtar parçası arasında normalize edilmiş birleşik olasılık hesaplanmır. Eğer hesaplanan normalleştirilmiş birleşik olasılık, daha önce kabul edilen "olayı kaçırma" eşik olasılığından, sözgelimi $P_o = 0.1$, büyük ise bu $LFSR_A$ ilk durumu gerçek ilk durum için bire aday olur. Ya da başka bir yöntemle,

aday $LFSR_A$ ilk durumları en yüksek birleşik olasılıklı olanlar olarak alınabilir. Saldırı bu yüzden $LFSR_A$'nın bütün ilk durumları üzerinden taramayı gerektirdiğinden, işlem karmaşıklığı $O\left(2^{r_A} r_A^2\right)$ kadar olacaktır.

Verilmiş bir z kayaranahtar dizisi uzunluğu n için, önce korelasyon saldırısı için gerekli LFSR çıktı uzunluğu $m(n)$ hesaplanır. Sonra iki LFSR'ın ilk durumları rasgele üretilir ve anahtar dizisinin n biti üretilir. $LFSR_A$'nın bütün ilk durumları için, bu anahtar dizisi ile $LFSR_A$ dizisinin $m(n)$ uzunlukta parçası arasında birleşik olasılık hesaplanır. Birleşik olasılıklar kıyaslanır ve o andaki ilk durumdan daha yüksek birleşik olasılık veren ilk durum sayıları kaydedilir. Algoritmanın ana hatları aşağıdadır:

Veri : $LFSR_A$ ve $LFSR_S$ geribesleme polinomları, gözlenen anahtar parçası uzunluğu n, gerekli LFSr dizileri uzunluğu $m(n)$ ve bit silme olasılığı p (Kısaltan üreteç için 0.5'e ayarlıdır)

İlkleme : $LFSR_S$ ve $LFSR_A$ ilk durum tohum indisleri $i, j = 1$. Ayrıca İlk durum tohum maksimum adeti

Durma Koşulu : $LFSR_S$ ilk durum tohum sayısının maksimum adete ulaşması.

Adım 1. $j = 1$ olsun. Rasgele bir $LFSR_S$ ilk durumu üret.

Adım 2. Geribesleme polinomunu kullanarak ilk durumdan, n adet 1 içeren $LFSR_S$ dizisi $\{s_k\}_{k=1}^{m^*}$ üret.

Adım 3. Rasgele bir $LFSR_A$ ilk durumu üret.

Adım 4. Geribesleme polinomunu kullanarak bu ilk durumdan $LFSR_A$ dizisi $\{a_k\}_{k=1}^{m^*}$ 'yı üret.

Adım 5. $\{s_k\}_{k=1}^{m^*}$ dizisini $\{a_k\}_{k=1}^{m^*}$ dizisine uygulayarak $LFSR_A$'dan $\{z_k\}_{k=1}^{m^*}$ anahtar dizisini üret.

Adım 6. z dizisine olasılıksal korelasyon saldırısını uygula.

Adım 7. Gerçek ilk durumdan daha yüksek birleşik olasılık veren bütün $LFSR_A$ ilk durumlarını say.

Adım 8. Eğer $j \le j_{max}$ ise j'yi artır ve Adım 3'e git.

Adım 9. Eğer $i \le i_{max}$ ise i'yi artır ve Adım 1'e git.

Adım 10. Programdan çık.

Çıktı : Gerçek ilk durumlar için birleşik olasılıklar, en yüksek birleşik olasılık değerleri ve bu değerleri üreten $LFSR_A$ ilk durumlarının indisleri, gerçek ilk durumdan daha yüksek birleşik olasılığı olan $LFSR_A$ ilk durumlarının sayısı.

7.2.2 $LFSR_s$ nin İlk Halinin Bulunması

Saldırının tamamlanması için, $LFSR_s$ nin de ilk halinin bulunması gerekir. Kriptoanalistin kayaranahtar parçası, aday bir $LFSR_A$dizisi ile $LFSR_s$ geribesleme polinomunu bildiğini varsayalım. Yaklaşımlardan birisi her $LFSR_A$ anahtar adayı için, $LFSR_A$'nin bütün olası ilk durumlarını taramaktır ki, bütün anahtar uzayını taramaya nispetle gerçekten önemli bir azalma olacaktır. Fakat eğer (Levenstein) edit uzaklığını kullanırsak işlem karmaşıklığı daha da düşecektir. $LFSR_A$ dizi adaylarını, $LFSR_A$ dizisi adayı $\hat{a}$, $\hat{A}^{m(n^*)} = \{\hat{a}_i\}_{i=1}^{m(n)^*}$ ile bilinen kayaranahtar dizisinin bir parçası $Z^{n^*} = \{z_i\}_{i=1}^{n^*}$ arasındaki edit uzaklığını hesaplayarak test edelim. Eğer edit uzaklığı iki dizi uzunluğu arasındaki farktan büyükse, o zaman $LFSR_A$ adayı hatalı kabul edilecek ve gözardı edilir. Edit uzaklığını, karşılık gelen kısmi edit uzaklıklarını ard arda hesaplayarak elde edebiliriz. Bütün kısmi edit uzaklıkları bir matriste depo edilir. Eğer $LFSR_A$ ilk durumu minimum edit uzaklığındanı veriyorsa, o zaman bütün (tam tamına n^* 1 ihtiva eden) $\hat{A}^{m(n^*)}$ ve Z^{n^*} ile tutarlı bütün saat-kontrollü dizileri bu matristen geriye gitme yoluyla tekrar elde edebiliriz. Her bir "saat-kontrollü" dizinin sonondaki 0 lar atılacaktır. $LFSR_s$ dizisi adayı $\hat{s}$, her bir saat kontrollenmiş dizinin son r_s bitinden tekrar oluşturulabilir. Bu yüzden, $n^* = \min\left(r_s, r_s/2 + 3\sqrt{r_s}\right)$ ve $m(n^*)$'yü bir önceki olduğu gibi seçmeliyiz. Her aday ilk durumu $LFSR_s$ geribesleme polinomunu kullanarak $\hat{s}$ aday dizisi

üretmek ve $\hat{a}$ dizisine $\hat{s}$ yi uygulayarak bir çıktı dizisi üreterek test ederiz. Eğer üretilen çıktı bilinen kayaranahtar dizisi ile aynı değil ise, $LFSR_s$ aday ilk durumunu reddet.

Bu işlemi ya bütün olası $LFSR_s$ ilk durum adaylarını reddeden kadar devam ederiz ki, bu durumda $LFSR_A$ ilk durumu yanlıştır veya aynı jayar anahtar dizisi veren bir $LFSR_s$ ilk durumunu bulana kadar devam ederiz. r_s uzunluğunda saatlenmiş ve test edilmesi gerekecek tutarlı dizi sayısının 2^{r_s}'den küçük olması beklenir.

7.3 Sonuçlar

Simpson, Golic vd.(1998) tarafından yapılan deneylerde, $LFSR_A$ için $1+x+x^{15}$ geribesleme polinomu ve $LFSR_s$ için $1+x^3+x^{17}$ geribesleme polinomu ile deneyler yapılıyor. Toplam anahtar uzunluğu 32 bit ve n bilinen kayar anahtar dizisi uzunluğunu belirtsin. Her bir n için, beş değişik rasgele üretilmiş $LFSR_S$ ilk durumu ve on farklı rasgele üretilmiş $LFSR_A$ ilk durumu için deneyler yapıldı. Yani, her bir n değeri için, kısıtlanmamış korelasyon saldırısı için 50 deneme yapılmış. Her deneme anahtar uzayının tamamen taranmasını gerektirdi.

8. SONUÇLAR ve YORUMLAR

Doğrusal geri beslemeli ötelemeli yazıcıların (LFSR) ürettikleri çıktı dizisinin doğrusallığı nedeniyle tek başına kullanılamayacağı bilinmektedir. Berlekamp_Massey algoritması sayesinde geribesleme polinomu düşman kriptanalist tarafından bilinmiyor olsa bile LFSR uzunluğunun iki katı kadar uzunlukta bir dizi sayesinde anahtar ele geçirilebilir. LFSR tabanlı bir sistemi daha güvenli bir hale getirmenin yollarından birisi birkaç LFSR çıktısını bir booleen fonksiyonla birleştirmektir. Ne var ki, booleen fonksiyonların doğal yapıları nedeniyle LFSR çıktısı ile birleştirici fonksiyon çıktısı arasında korelasyon olması neredeyse kaçınılmazdır.

Siegenthaler'in "hızlı korelasyon" saldırılarının öncüsü kabul edilebilecek makalesi, *yüksek korelasyon ve yeterince uzun olmayan LFSR boylarında sistemin yeterli güvenlikte olmadığını* göstermiştir. Saldırının dikkate değer bir özelliği, dil istatistikleri bilindiğinde yalnızca şifreli metin saldırısı olmasıdır.

Staffelbach ve Meier, temel çıkış noktası aynı olan iki farklı saldırı ile pratik oldukları için kullanılan *düşük bağlantı noktası sayılı LFSR'ların sistem için bir zayıflık olabileceğini* göstermişlerdir. Ayrıca, bu saldırılar daha yüksek anahtar boyları ve daha düşük korelasyon için etkili olmaktadır. Bu saldırıların dezavantajı ise saldırılan LFSR'ın bağlantı nokta sayısının az olmasını gerektirmesidir.

Johannson ve Jönnson, sistem çıktısından LFSR'ları oluşturma problemini bir kodlama teorisi problemine dökmüşlerdir. Hızlı dekodlama teknikleri bilinen katlamalı kodlarla yapılan saldırıları ile hem *karmaşıklık açısından mesafe katedilmiş*, hem de başka saldırıların çıkış noktası olmuştur. Bunlar arasında yine kendilerine ait (bu tezde incelenmeyen) "öğrenme teorisi yoluyla doğrusal polinomları geri oluşturma" tekniğine dayalı saldırıyı ya da lineer kodlarla yapılan birkaç saldırı sayılabilir. Bunlardan Chepyzov vd. yaptıkları saldırı hem *daha az bellek gerektirmesi,* hem de bu saldırının NESSIE projesi LILI-128'e karşı bir saldırıya imkan vermesi nedeniyle bu tezde incelenmiştir. Lineer kodlarla yapılan saldırılar *düşük korelasyona sebep veren iyi bir birleştirici fonksiyonun tek başına yeterli olmayacağını* göstermiştir.

LFSR çıktılarını güvenli bir sisteme dönüştürmek için kullanılan tekniklerden biri de zaman kontrollü üreteç kullanılmasıdır. Golić ve diğerlerinin yaptığı çalışmalarda kullanılan korelasyon tanımı biraz farklı olsa da kısaltılmış üreteç ve değişmeli adım üreteçlerine düzenlenen saldırılar, *zaman kontrollü üreteçlerin de tek başlarına yeterli olmayacaklarını düşündürmektedir.*

Bu saldırılarda kullanılan uzaklık fonksiyonları saldırılar için adapte edilmiş "edit" uzaklıklarıdır. Edit uzaklığı korelasyon saldırılarını klasik LFSR çıktısı ile üreteç çıktısı arasında "birebir" korelasyon ele alınarak yapılan klasik hızlı korelasyon saldırılarının "zaman kontrollü versiyonu" olarak düşünebiliriz. Bu saldırılarda kullanılan kısaltılmış üreteç için kullanılan "birleşik olasılık" ve değişmeli adım üreteci için kullanılan "gömme olasılığı" kavramlarının daha detaylı teorik ve pratik analizleri bu saldırıların daha etkili kılınmasın ayol açabilir.

Korelasyon saldırılarının daha geniş bir dizi şifreleme sistemi kümesine (örneğin FCSR tabanlı şifreleme sistemleri de dahil edilerek bir FSR alt kümesi) uygulanıp uygulanamayacağı konusunda yeterli araştırma bulunmamaktadır.

Yapılan korelasyon saldırıları ve diğer saldırı türleri göz önüne alındığında (lineer tutarlılık testi vs.) LFSR tabanlı sistemlerin birleştirici fonksiyon, süzme fonksiyonu, zaman kontrollü üreteç vb. yapılardan en az ikisini içeren *bir "melez" yapı olması* gerektiğini düşünmekteyiz. Ayrıca pratik oldukları için kullanılan düşük bağlantı noktası sayısından vazgeçilemiyorsa bunu maskeleyecek bir önlem alınmalıdır.

Korelasyon saldırılarının dizi şifreleme saldırılarına karşı tek etkili saldırı türü olmadığı, son yıllarda LFSR tabanlı sistemler ile ilgili bu tezde incelenmeyen bir çok makale sonucunun saldırıların verimini önemli göz ardı edilmemelidir. Anahtar (LFSR) boyları seçilirken özellikle gelecekte işlem gücü, bellek ve paralel işlem konularında büyük ilerlemeler olabileceği unutulmamalıdır.

KAYNAKLAR

Aksoy, Ali Doğan(2000) Statistics and probability, Lecture Notes in UEKAE

Aslan, Zernişan Emirleroğlu(2001) Correlation Attavks on Stream Ciphers, Marmara Üniversitesi Bilgisayar Mühendisliği Bölümü yüksek Lisans Tezi.

Birinci, Fatih(1998) Correlation Attack on Stream Ciphers, Ortadoğu Teknik Üniversitesi Fen Edebiyat Fakültesi Matematik Bölümü Yüksek Lisans Tezi.

Chepyzhov, V., Johannson T. ve Smeets,B. "A Simple Algorithm for Fast Correlation Attacks on Stream Ciphers", Fast Software Encryption, FSE 2000, Lecture Notes in Computer Science, Springer Verlag.

Coppersmith, D., Krawczyk, H. ve Mansour,Y.(1993) "The Shrinking Generator". Advances in Cryptology-CRYPTO'93, Lecture Notes in Computer Science, 773:22-39.

Goldreich, O., Rubinfield,R. ve Sudan,M.(1995) "Learning Polynomials with Queries: The Highly Noise Case", 36th Yıllık Foundation of Computer Science Sempozyumu, Milwaukee, Wisconsin.

Golić, J. Dj. ve Mihaljević, M.(1991), "A generalized correlation attack on a class of stream ciphers based on Levenshtein distance", *Journal of Cryptology*, 3(3):201-212.

Golić, J. Dj. ve Petrović, S. (1993), "A generalized correlation attack with a probabilistic constrained edit distance", Advances in Cryptology-EUROCRYPT '92, Lecture Notes in Computer Science, R. A. Rueppel ed., Springer-Verlag, 658:472-476.

Golić, J. Dj. ve O'Connor, L. (1995), "Embedding and probabilistic correlation attacks on clock-controlled shift registers", Advances in Cryptology-EUROCRYPT '94, Lecture Notes in Computer Science, A. De Santis ed., Springer-Verlag, 950:230-243.

Golić, J. Dj. (1996), "Constrained embedding probability for two binary strings", SIAM Journal on Discrete Mathematics, 9(3): 360-364.

Golic, J. ve Menicocci, Dj. Renato (1997): "Edit Distance Correlation Attack on the Alternating Step Generator", Lecture Notes in Computer Science, Springer-Verlag, CRYPTO'97 : 499-512

Golomb, S. W. (1967), Shift Register Sequences, Holden-Day, San Francisco.

Gollmann, D. ve Chambers, W.G. (1989), "Clock-controlled shift registers: a review" IEEE Journal on Selected Areas in Communications,7:525-533.

Guy, R. K. (1988) "The strong law of small numbers," Amer. Math. Monthly, **95**(8): 697-712.

Günther, C.G. (1988) "Alternating step generators controlled by de Brujin sequences", Advances in Cryptology-EUROCRYPT '87, Lecture Notes in Computer Science, D. Chaum and W.L. Price eds., Springer-Verlag, 304:5-14.

Johannson, T. ve Jönnson, F.(1999), "Improved Fast Correlation Attacks on Stream Ciphers via Convolutional Codes", Advances in Cryptology-EUROCRYPT'99, Lecture Notes in Computer Science, 1592:347-362

Johannson, T. ve Jönnson, F.(1999), "Fast Correlation Attacks based on Turbo Code Techniques", Advances in Cryptology-CRYPTO'99, Lecture Notes in Computer Science, 1666:181-197

Kahn, D. (1967), The Codebreakers, McMillan, NewYork.

Kara, Orhun (2001), LFSR Based Stream Ciphers, Lecture Notes in UEKAE.

Klapper, Andrew ve Xu, Jinzhong(1999) "Algebraic feedback shift registers", Theoritical Computer Science, 226(1-2):61-92

Massey, J.L. (1969), "Shift-register synthesis and BCH decoding", IEEE Trans. Inform. Theory, IT-15:122-127.

Meier W. ve Staffelbach O.(1989), "Fast Correlation Attacks on Certain Stream Ciphers" Journal of Cryptology, 1:159-176.

Menezes, A., van Oorschot, P. ve Vanstone, S.(1997), Handbook of Applied Cryptography, CRC Press.

Mihaljević, Miodrag J., Fossorier, Marc P.C. ve Imai, Hideki(2000) "A low Complexity and High-Performance Algorithm for the Fast Correlation Attack" Fast Software Encryption, FSE 2000, Lecture Notes in Computer Science, Springer Verlag.

Rueppel, R.(1986), Analysis and design of stream ciphers, Springer-Verlag, Berlin.

Siegenthaler T.(1985), "Decrypting a class of Stream Ciphers Using Ciphertext Only", IEEE Transactions on Computers, C-34:81-85.

Simpson, L., Golić, J. Dj. Ve Dawson, E. (1998) "A Probabilistic Correlation Attack on the Shrinking Generator",Information Security and Privacy, Lecture Notes in Computer Science, 1438:147-158.

Yang, Chung-Huang, "On the Design of High-Speed Pseudorandom bit Generators for Secure Broadcasting Systems" http://crypto.nknu.edu.tw/publications/jw_isc2000_chyang.PDF.

Zeng, K., Yang, C.H. ve Rao, T.R.N.(1990), "On the Linear Consistency test(LCT) in Cryptanalysis and its Applications", Advances in Cryptology- CRYPTO'89, Lecture Notes in Computer Science, 434:164-174.

Zeng, K., Yang, C.H. ve Rao, T.R.N.(1991), "An ımproved Linear Syndrome Method in Cryptanalysis with Applications", Advances in Cryptology- CRYPTO'90, Lecture Notes in Computer Science, 537:34-47.

Živković, M.V. (1991), "An Algorithm for the Initial State Reconstruction of the Clock-controlled Shift Register", IEEE Trans. Inform. Theory, IT-37:1488-1490.

ÖZGEÇMİŞ

Doğum Tarihi	17.10.1972	
Doğum Yeri	Antalya	
Lise	1987-1990	Ankara Fen Lisesi
Üniversite	1990-1994	Orta Doğu Teknik Üniversitesi Eğitim Fakültesi Matematik Öğretmenliği
	1990-1994	Orta Doğu Teknik Üniversitesi Fen-Edebiyat Fakültesi Matematik Lisans(çift anadal)
	1996-2000	Fransa(MEB Bursu)

Çalıştığı Kurumlar

OrtaDoğu Teknik Üniversitesi	1995 içi	Araştırma Görevlisi
TÜBİTAK UEKAE	2000-Devam ediyor	Araştırmacı