T.C
TRAKYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AKIŞ ŞİFRELERİN TASARIM TEKNİKLERİ VE GÜÇ ANALİZİ

Fatma BÜYÜKSARAÇOĞLU SAKALLI

Doktora Tezi

Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Yrd. Doç. Dr. Ercan BULUŞ

EDİRNE-2011

T.C.
TRAKYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AKIŞ ŞİFRELERİN TASARIM TEKNİKLERİ VE GÜÇ ANALİZİ

Fatma BÜYÜKSARAÇOĞLU SAKALLI

Doktora Tezi

Bilgisayar Mühendisliği Anabilim Dalı

Bu tez 12 / 09 / 2011 tarihinde aşağıdaki jüri tarafından kabul edilmiştir.

Jüri

Yrd. Doç. Dr. Ercan BULUŞ
Danışman
Jüri Başkanı

Doç. Dr. Mümin ŞAHİN
Üye

Yrd. Doç. Dr. Tarık YERLİKAYA
Üye

Yrd. Doç. Dr. Altan MESUT
Üye

Yrd. Doç. Dr. Erdinç UZUN
Üye

T.C
TRAKYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENTSTİTÜSÜ

AKIŞ ŞİFRELERİN TASARIM TEKNİKLERİ VE GÜÇ ANALİZİ

Fatma BÜYÜKSARAÇOĞLU SAKALLI

Doktora Tezi
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Yrd. Doç. Dr. Ercan BULUŞ

EDİRNE-2011

Doktora Tezi
Trakya Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Bölümü

ÖZET

Bu tez, simetrik şifreleme algoritmalarından akış şifreler ve blok şifreler ile ilgilidir. Akış şifrelerin tasarım yapıları, rassal sayı üretimi, rassallığın test edilmesi ve test kriterleri incelenmiş, akış şifreleme algoritmalarından bazıları için test kriterlerinin sonuçlarına yer verilmiştir. Diğer yandan simetrik şifreleme algoritmalarından olan blok şifreler için anahtar genişletme algoritmalarına yönelik yeni bir mimari sunulmuştur. Bu mimari Katı Çığ Kriteri Testi (SAC – Strict Avalanche Criterion) ve bit sızdırma özellikleri dikkate alınarak geliştirilmiştir.

Tezin giriş bölümünde temel şifreleme yapıları olan blok ve akış şifreler ve bu şifrelere karşı yapılan kriptanaliz saldırıları ile istatistiksel testlerin tanımı yapılmıştır.

Tezin 2. bölümünde akış şifrelerin tasarım mimarileri ve matematiksel alt yapıları incelenmiştir

3. Bölümde akış şifreleme algoritmalarında kullanılan rassal sayı üreteçleri incelenmiştir.

4. bölümde kriptografik uygulamalarda kullanılan rassal sayıların ve bu sayılar kullanılarak geliştirilen anahtarların güvenilirliğini sağlamada önemli kriter oluşturan istatistiksel testler incelenmiştir. Rassallık için günümüzde pek çok uygulamada kullanılan NIST (National Institute of Standards and Technology - Ulusal Standartlar ve Teknoloji Enstitüsü) test paketinden faydalanılmış ve bu bölümde NIST test paketinde bulunan testlerin matematiksel alt yapıları incelenmiştir.

5. bölümde blok şifreler ve bu şifrelerde kullanılan önemli yapılar tartışılmıştır. Buna ek olarak AES (Advanced Encryption Standard) blok şifresinin içyapısı ve anahtar genişletme algoritması incelenmiştir.

6. bölümünde, seçilen akış şifreleme algoritmalarının çalıştırılmasıyla elde edilen anahtar değerlerinin NIST test paketi programına uygulanmasına ve sonuçlarının değerlendirilmesine yer verilmiştir

7. bölümde AES'de kullanılan anahtar genişletme algoritmasının iki önemli eksiklikten (yavaş yayılım, bit sızdırma) yavaş yayılım özelliğindeki eksiklliğe dikkat çekilmiş, bu sorun geliştirilen programlar ile ölçülerek değerlendirmeler yapılmış ve gözlenen sorun için yeni bir yaklaşım önerilmiştir. Ayrıca bu çalışma iki eksikliği giderecek şekilde kolaylıkla geliştirilebilir.

8. bölümde bir önceki bölümde açıklanan anahtar planlama stratejilerinden yola çıkarak herhangi bir blok şifresinde kullanılabilecek, istatistiksel özellikleri iyi yeni bir anahtar genişletme mimarisi sunularak bu mimariye ilişkin deneysel sonuçlara yer verilmiştir.

9. bölümde tezde elde edilen sonuçların değerlendirilmesi yapılmıştır.

Anahtar Sözcükler: Akış Şifreler, Rassal Sayı Üreteçleri, NIST (National Institute of Standards) Testleri, Anahtar Planlama, AES (Advanced Encryption Standard)

Yıl: 2011

Sayfa: 189

Doctorate Thesis
Trakya University Graduate School of
Natural and Applied Sciences
Department of Computer Engineering

ABSTRACT

This thesis is related with symmetric encryption algorithms which cover stream ciphers and block ciphers. The design principles of stream ciphers, (pseudo) random number generation, testing of randomness and testing criteria of randomness are investigated. Also, the results of testing criteria of randomness for some chosen stream ciphers are shown. On the other hand, a new key expansion algorithm structure to be used in block cipher design is proposed by taking into account the two important properties for key expansion algorithm design for block ciphers which are Strict Avalanche Criterion (SAC) and bit leakage.

In the introduction section of the thesis, an overview of block ciphers and stream ciphers, cryptanalytic attacks against these ciphers and an introduction to statistical tests for these ciphers are given.

In the second chapter of the thesis, the design architecture of the stream ciphers and mathematical background of them are investigated.

In the third chapter of the thesis, pseudo random number generators used in stream ciphers are investigated.

In the fourth chapter of the thesis, statistical tests found in NIST (National Institute of Standards and Technology) package for evaluating the statistical properties of cryptographic primitives are investigated. Also, the mathematical background of these statistical tests is given.

In the fifth chapter of the thesis, block ciphers and important components of block ciphers are discussed. In addition, an overview of AES (Advanced Encryption Standard) block cipher and its key expansion algorithm are given.

In the sixth chapter of the thesis, the statistical tests found in NIST is applied to key streams obtained from chosen stream ciphers and an evaluation of the results are given.

In the seventh chapter of the thesis, two important deficiencies related with AES key expansion algorithm are identified and a new approach for AES key expansion algorithm solving the problem of slow diffusion property is given. This study can also easily be developed to solve the two important deficiencies which are slow diffusion and bit leakage.

In the eighth chapter of the thesis, a new structure for key expansion algorithms, which can be implemented independently of a block cipher, is given. In addition, an implementation of this structure together with results is given. This new structure gives very good results from the point of statistical view and solves two important problems (slow diffusion and bit leakage) for key expansion algorithms.

In the nineth chapter of the thesis, an evaluation of the given results of the thesis is discussed.

Keywords: Stream ciphers, (Pseudo) Random Generators, NIST (National Institute of Standards) Test Suicide, Key Expansion, AES (Advanced Encryption Standard)

Year: 2011

Page: 189

TEŞEKKÜR

Bu tez için gerçekleştirdiğim araştırmalar sırasında birçok kişinin bana katkısı olmuştur. Bu kişilere burada destekleri için teşekkür etmek isterim.

İlk olarak tez çalışmam sırasında bana yardımlarından ve katkılarından dolayı değerli hocam ve danışmanım Sayın Yrd.Doç.Dr. Ercan BULUŞ'a sonsuz teşekkürlerimi sunmak isterim.

Bu tezin izleme komitesinde yer alan ve yine bana verdikleri destek ve değerli katkılarından dolayı Doç.Dr. Mümin ŞAHİN, Yrd.Doç.Dr. Tarık YERLİKAYA ve Yrd.Doç.Dr. Altan MESUT'a sonsuz teşekkürlerimi sunarım.

Trakya Üniversitesi Bilgisayar Mühendisliği Bölümü öğretim üyeleri Yrd.Doç.Dr. Andaç MESUT ve Yrd.Doç.Dr. Özlem AYDIN'a sıcak dostlukları, paylaşımları ve destekleri için, Arş.Gör. Edip Serdar GÜNER ve Arş.Gör. Nazan DEMİRCİ'ye verdikleri pozitif enerji için, Arş.Gör. Emir ÖZTÜRK'e bilgisi ve desteğiyle çalışmalarımın yapı taşlarını oluşturmamda gösterdiği çaba ve harcadığı zaman için teşekkürlerimi sunuyorum.

Yrd.Doç.Dr. H.Nusret BULUŞ olmasa sanırım tez süreci ve çalışma ortamı çekilir olmazdı. Güzel yüreği, bilgisi ve desteğiyle ofis arkadaşım olarak paylaştığı tüm zamanları için kendisine teşekkür ederim.

Beraber doktora sürecini paylaştığım ve bu süreç boyunca daima yanımda olan Arş.Gör. Derya ARDA, Öğr.Gör. Deniz Mertkan GEZGİN ve Öğr.Gör. Fatma AKGÜN'e paylaşımları, destekleri ve dostlukları için teşekkürlerimi sunarım.

Öğrencilik ve akademik hayatım süresince bilgi ve birikimini benimle paylaşarak çalışmalarımda desteğini esirgemeyen, dostum ve meslektaşım Sayın Mehmet Ali Aksoy TÜYSÜZ'e, beraber büyüdüğüm gerçek bir dost olarak tüm

kalbiyle her zaman yanımda olup beni asla yalnız bırakmayan Sayın Pelin AKYÜZLÜ'ye sonsuz teşekkürlerimi sunarım.

Çalışma ortamında paylaştıkları dostluk ve eşsiz bilgileri ile çalışmalarıma destek oldukları için Trakya Üniversitesi Makine Mühendisliği Bölümü Araştırma Görevlileri'ne teşekkür ederim.

Tez çalışmamın ilerlemesinde büyük emeği olan, bilgi ve paylaşımlarının yanı sıra sıcak dostluklarını da esirgemeyerek bana destek veren Öğr.Gör. Bora ASLAN ve değerli eşi Füsun YAVUZER ASLAN'a sonsuz teşekkür ederim.

Dünyada sahip olunacak nadir annelerden biri olan, hayatımın her adımında yanımda olduğunu hissettiğim ve kızı olmaktan sonsuz onur ve gurur duyduğum biricik annem Sayın Mesude BÜYÜKSARAÇOĞLU'na hem bu tez süreci hem de hayatımın tüm anları için minnettarlığımı sunarım.

Yapılacak ne kadar fedakarlık varsa yapabilecek kadar büyük bir ruha ve kalbe sahip olan, her anımda beni en çok anlayan tek insan olan kardeşim olduğu için onur ve gurur duyduğum Sayın Funda BÜYÜKSARAÇOĞLU'na teşekkürü bir borç bilirim.

Bu tezin ortaya çıkmasını sağlayan büyük emeği, hayatımdaki varlığı ve desteği için sevgili eşim Yrd.Doç.Dr. M. Tolga SAKALLI'ya teşekkür ederim.

Bu mesleği seçmemi sağlayan ve akademik yönde çalışmam konusunda beni teşvik eden, karakteri ve hayata duruşuyla örnek alınacak nadir insanlardan biri olan, başarılı olacağıma inandığı halde beni en zor koşullara göre tekrar tekrar sınayıp ayaklarımın üstünde durma becerisini kazandıran, 4 sene önce aramızdan ayrılıp beni sonsuz bir yalnızlığa ve hasretine muhtaç bıraksa da öğrettikleriyle hala yanımda olduğunu hissettiğim babam demekten şeref duyduğum Sayın **Muammer BÜYÜKSARAÇOĞLU**'na sonsuz sevgi, saygı ve teşekkürlerimi sunuyor bu tezi onun mukaddes ruhuna itaf ediyorum.

Eylül 2011 **Fatma BÜYÜKSARAÇOĞLU SAKALLI**

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

1. GİRİŞ

Telgrafın icadıyla başlayan ve internetin gelişimi ile beraber devam eden teknolojik gelişim uzak mesafelere bilginin gönderilmesinde şifrelemenin önemini arttırmıştır (Wu., 2008). Buna ek olarak şifreleme, askeri ve diplomatik mekanizmalarda bilginin gizliliğinin korunmasında geniş bir uygulama alanı bulmuştur. Günümüzde ise daha fazla insan çok büyük miktarlarda verinin karşı tarafa iletilmesi için internete bağlanmaktadır. Dolayısıyla kriptografi bilginin gizliliğinin, bütünlüğünün korunmasında ve kimlik denetiminin sağlanmasında önemli bir rol üstlenmiştir. Özellikle simetrik anahtarlı şifreleme yöntemleri gizli bilginin iletiminde ve depolanmasında önemli rol oynar.

Julius Caesar'ın (MÖ 100-44) kullandığı Sezar şifresi ile başlayan şifreleme yöntemleri 1970 yılının ortalarında IBM ve NSA tarafından ortaya atılan DES (Data Encryption Standard) (Federal Information Processing Standards Publications, 1999) algoritması ile yeni bir boyuta ulaşmıştır. Bu algoritmanın ortaya atılması ile günümüz modern şifreleme algoritmalarının gelişimi başlamıştır.

Kriptoloji, bilgi güvenliği servislerinin tasarımını, analizini ve bu servislere olan saldırıları inceleyen geniş bir bilim dalıdır. Dolayısıyla kriptoloji, kriptografi ve kriptanaliz olmak üzere iki alt alanı içerir. Bu alt alanlardan kriptografi, bilgi servislerinin tasarımı ve analizi ile ilgili iken kriptanaliz bu servislere olan saldırılar ile ilgilidir (Keliher, 2003). Aşağıda kriptografik servislerinin amaçları verilmiştir:

- *Gizlilik* (Confidentiality): Bilginin gizliliği sağlanmasını amaç edinir,
- *Veri Bütünlüğü* (Data integrity): Bilginin sadece yetkili kullanıcılar tarafından değiştirilebilmesini temin etmeyi amaç edinir,
- *Kimlik Denetimi* (Authentication): Bilginin gönderildiği yerin ve gittiği hedefin kimlik denetiminin yapılmasını amaç edinir,
- *İnkar Edememe* (Non-repudiation): Bir kişinin bir önceki taahhüdünü veya eylemini inkar edememesini temin etmeyi amaç edinir.

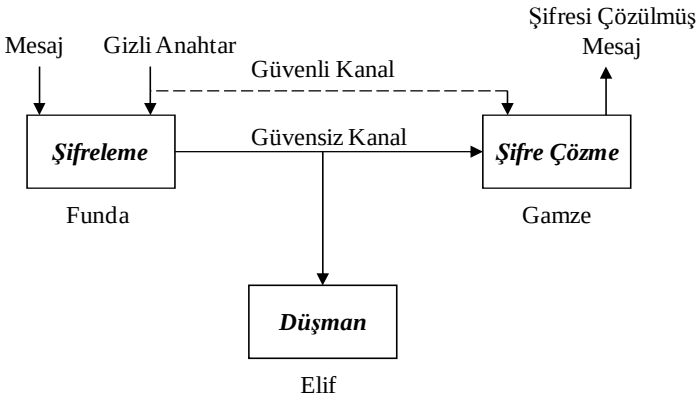
Bu bahsedilen konuların çözümü kriptografik yapıların kullanımı ile sağlanır. Örneğin gizliliğin sağlanması için şifreleme algoritmaları kullanılır.

Kriptografik yapılar üç temel alana bölünebilir: Simetrik anahtarlı yapılar, asimetrik anahtarlı yapılar (açık anahtarlı yapılar) ve anahtarsız yapılar. Simetrik anahtarlı yapılar, şifreleme ve şifre çözme (deşifreleme) işlemlerinde aynı anahtarı kullanır. Dolayısıyla anahtarın daha önceden güvenli bir kanal üzerinden karşı tarafa iletilmesi gerekir. Bu tür bir şifre için klasik bir örnek Vigenere şifresi (Ekdahl, 2003; Stinson, 2002) verilebilir. Simetrik anahtarlı bir şifre için orijinal mesaja açık metin (P) ve bu mesajın anlaşılmaz formuna da şifreli metin (C) adı verilir. Gizli anahtar K ile tanımlanır. P den C'ye dönüşüme şifreleme, bu işlemin tersine ise şifre çözme (deşifreleme) adı verilir. E_K, K gizli anahtarı ile şifreleme algoritmasını temsil ederse şifreleme ve deşifreleme işlemi aşağıdaki gibi verilebilir:

$$E_K(P) = C,\ E_K{}^{-1}(C) = P$$

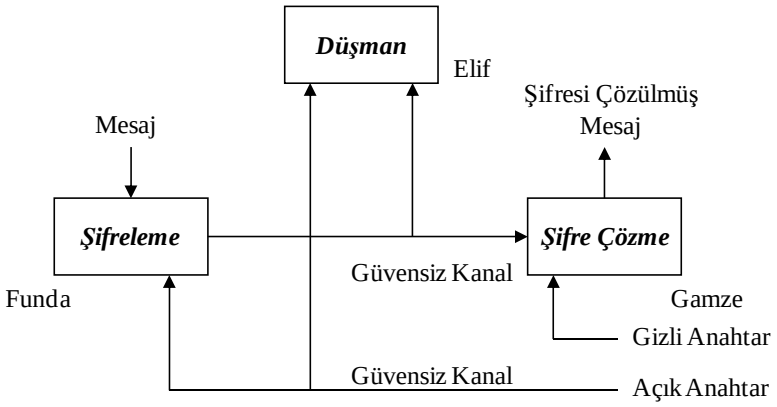
Şekil 1.1'de güvensiz bir kanalda bulunan düşman ile birlikte simetrik şifre kullanılan bir haberleşme örneği gösterilmiştir.

Diğer yandan simetrik anahtarlı şifreler akış şifreler ve blok şifreler olmak üzere ikiye ayrılır. Blok şifreler bazı uygulamalar için daha uygundur. Buna ek olarak bir blok şifre hash fonksiyonları ve mesaj kimlik denetimi kodları (Message Authentication Codes-MACs) gibi diğer kriptografik yapıların elde edilmesinde kullanışlıdır. Bununla beraber akış şifrelerin blok şifrelere göre iki önemli avantajı vardır. İlki, blok şifreler ile aynı güvenlik seviyesine bir blok şifreden daha az hesaplama ile erişilebilir olmasıdır. Diğeri ise anahtar dizisinin daha önceden hesaplanması ile akış şifrenin şifrelenmesi ve deşifrelenmesinin oldukça hızlı olarak gerçekleştirilebilmesidir (Wu, 2008).



Şekil 1.1. Güvensiz Kanalda Dinleme Yapan Bir Düşman İle Birlikte Simetrik Şifre Kullanan Haberleşme (Ekdahl, 2003)

Asimetrik anahtarlı yapılar ise Diffie ve Hellman (Diffie, Hellman, 1976) tarafından yapılan bir çalışma ile gelişmeye başlamıştır ve ana fikir anahtarın simetrik olmamasına dayanır. Bu çalışmadan iki yıl sonra geliştirilen RSA şifreleme algoritması (Rivest, Shamir, Adleman, 1978) ile şifreleme ve deşifreleme için farklı anahtarlar kullanılmıştır. Buradaki önemli nokta deşifreleme anahtarının şifreleme anahtarından yola çıkarak hesaplanabilir olamamasıdır. Şekil 1.2'de düşmanın hem açık anahtarı hem de şifreli metni gördüğü asimetrik anahtarlı şifre kullanarak haberleşme örneği gösterilmiştir.



Şekil 1.2. Asimetrik Anahtarlı Şifre Kullanan Haberleşme (Ekdahl, 2003)

Anahtarsız kriptografik yapılar genellikle anahtarlı yapıların elde edilmesinde kullanılır. Bu tür yapılara örnek olarak Hash fonksiyonları verilebilir. Hash fonksiyonları bilginin kısa bir özetinin oluşturulmasını sağlayan fonksiyonlardır. Kriptografide en yaygın uygulama alanı sayısal imzanın yaratılması ve doğrulanmasındadır. Verinin bütünlüğünün sağlanmasında, veritabanlarında ve arama algoritmalarında da kullanım alanları vardır (Grah, 2008).

Tezin Önemi ve Gerekçesi: Bu tez simetrik şifreleme tekniklerinden akış şifreler ile bu şifreleme tekniklerine uygulanan istatistiksel testler üzerinedir. NIST test paketinde bulunan çeşitli test teknikleri incelenerek seçilen bazı akış şifrelere bu testler uygulanmış ve sonuçlar değerlendirilmiştir. Buna ek olarak son yıllarda blok şifrelere olan bazı saldırılar blok şifrelerin anahtar genişletme algoritmalarını ve bunların kötü tasarımlarını hedef almaktadır. Dolayısıyla bu tezde AES blok şifresinin anahtar genişletme algoritmasının iki zayıf noktası (yavaş yayılım ve bit sızdırma) tespit edilerek bu zayıf noktalardan yavaş yayılım özelliği düzeltilmeye çalışılmıştır. Buna ek olarak günümüzdeki blok şifrelerde ön plana çıkan iki temel problemin çözümü için yeni bir anahtar genişletme mimarisi öne sürülerek bu mimariye ilişkin uygulamalar geliştirilmiş ve sonuçlar değerlendirilmiştir. Önemli diğer bir noktada geliştirilen mimarinin herhangi bir blok şifreden bağımsız ve uygulamasının sadece XOR işlemli tabanlı olmasıdır.

1.1. Sonlu Cisimler Teorisine Giriş

Sonlu cisimler teorisi, hata düzeltme kodları, sayısal sinyal işleme ve kriptografi gibi alanlarda kullanılan bir teoridir. Bu bölümde bu teori kriptografide kullanımı açısından incelenecektir. Bu teori ile ilgili daha detaylı bilgi (McEliece, 1987; Lidl, Niederreiter, 1983) referanslarından elde edilebilir.

Tanım 1.1. Bir cisim aşağıda verilen aksiyomları toplama ve çarpma işlemine göre sağlayan boş olmayan bir F kümesidir.

$\forall a, b, c \in F$ olmak üzere

a-) F toplama işlemi altında kapalıdır.

b-) F toplama işlemi altında değişme özelliğini sağlar

c-) F toplama işlemi altında geçişme özelliğini sağlar

d-) F toplama işlemi altında dağılma özelliğini sağlar.

Bunun ötesinde iki farklı birim eleman 0 ve 1 (toplamaya ve çarpmaya göre birim elemanlar) F 'de aşağıdaki aksiyomları sağlayacak şekilde bulunmalıdır;

e-) $a + 0 = a, \forall a \in F$.

f-) $a.1 = a \; ve \; a.0 = 0, \forall a \in F$

g-) F içindeki herhangi bir a için, toplama işlemine göre ters eleman $(-a)$, $a + (-a) = 0$ olacak şekilde F içinde vardır.

h-) F içindeki herhangi bir eleman için $a \neq 0$ için, $a.a^{-1} = 1$ olacak şekilde a^{-1} çarpma işlemine göre ters eleman olmalıdır.

Genellikle $a.b$ basitçe ab olarak yazılabilir ve F^* set $F \setminus \{0\}$ olarak tanımlanabilir.

Teorem 1.1. Z_m ($\mathrm{mod}\, m$ işlemine göre kalanların oluşturduğu küme) eğer m asal ise bir sonlu cisim oluşturur.

Örnek 1.1: Z_4 ($\mathrm{mod}\, 4$ işlemine göre kalanların oluşturduğu küme) bir cisim oluşturmaz. Bunun nedeni olarak aşağıda görüldüğü gibi bir cismin oluşabilmesi için gereken tüm özellikler sağlanmaz. Çünkü 2 değerinin çarpma işlemine göre tersi yoktur.

+	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

X	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Örnek 1.2: Z_5 ($\mathrm{mod}\,5$ işlemine göre kalanların oluşturduğu küme) Teorem 1.1 gereği bir cisim oluşturur. Aşağıda Z_5'de toplama ve çarpma işlemleri için oluşturulan iki tablo bunu doğrulamaktadır.

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

X	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Tanım 1.2. Sonlu bir cisim F için, $\alpha \in F$ elemanının derecesi $\alpha^m = 1$ olacak şekilde en küçük pozitif m sayısıdır.

Teorem 1.2. Her tamsayı $n > 1$ asal sayıların bir ürünü olarak yazılabilir. Diğer bir deyişle her tamsayı $n = p_1^{m_1} . p_2^{m_2} \ldots p_r^{m_r}$ şeklinde ifade edilebilir ve buna n'nin cononical faktorizasyonu adı verilir.
Not: İfadedeki m_i'ler pozitif ve $p_1 < p_2 < \ldots < p_r$ şeklindedir.

Teorem 1.3. $m = \prod_{i=1}^{n} p_i^{m_i}$ p_i'ler farklı tamsayılar ve $m_i > 0, 1 \le i \le n$ olmak üzere

$\phi(m) = \prod_{i=1}^{n} (p_i^{e_i} - p_i^{e_i - 1})$ şeklindedir. $\phi(m)$'e Euler Phi fonksiyonu denir ve bu fonksiyon 1 ile m arasında m ile aralarında asal olanların sayısını verir.

Teorem 1.4. Bir Z_p kümesinin, eğer p asal sayı ise, bir elemanının en yüksek derecesi $\phi(p) = p - 1$ dir ve bu dereceye sahip elemana ilkel (primitive) eleman adı verilir. Buna ek olarak herhangi bir elemanın derecesi verilen $p - 1$ değerini böler.

Teorem 1.5. p asal sayı ve $\alpha \in Z_p^*$ olsun. O zaman sadece ve sadece $\frac{p-1}{q}$ olacak şekilde tüm asal q'lar için $\alpha^{\frac{p-1}{q}} \neq 1 \bmod p$ şeklinde ise $\alpha \bmod p$'ye göre ilkel elemandır. (Z_p^*, 0 elemanı hariç diğer elemanların oluşturduğu kümeyi temsil eder.)

Teorem 1.6. Eğer p bir asal sayı ve $\alpha \bmod p$'ye göre ilkel eleman ise $\beta \in Z_p^*$ ve $\beta = \alpha^i, 0 \leq i \leq p-2$ şeklinde yazılabilir. $\beta = \alpha^i$'nin derecesi $\frac{p-1}{\gcd(p-1, i)}$ şeklinde elde edilebilir.

Verilen teorileri kullanarak Örnek 1.3'de Z_{11} sonlu cismi için ilkel elemanlar gösterilmektedir.

Örnek 1.3: Z_{11}'de kaç tane ilkel eleman vardır ve bu ilkel elemanları bulunuz.

İlk önce bir ilkel eleman bulmalıyız:

$$2^0 \bmod 11 = 1 \qquad 2^8 \bmod 11 = 3$$
$$2^1 \bmod 11 = 2 \qquad 2^9 \bmod 11 = 6$$
$$2^2 \bmod 11 = 4 \qquad 2^{10} \bmod 11 = 1$$
$$2^3 \bmod 11 = 8$$
$$2^4 \bmod 11 = 5$$
$$2^5 \bmod 11 = 10$$
$$2^6 \bmod 11 = 9$$
$$2^7 \bmod 11 = 7$$

Yukarıda gösterildiği gibi 2 elemanı Z_{11}'de ilkel bir elemandır ve $\gcd(10, i) = 1$ olacak şekilde i değerlerine sahip 2^i değerleri ilkel eleman olacaktır. $\gcd(10, i)$ değerleri 1, 3, 7, 9 olacağından diğer ilkel elemanlar; $2^1 \bmod 11 = 2, 2^3 \bmod 11 = 8, 2^7 \bmod 11 = 7, 2^9 \bmod 11 = 6$ şeklinde hesaplanabilir. Dolayısıyla Z_{11}'de ilkel elemanlar 2, 6, 7 ve 8 dir.

Tanım 1.3. F bir cisim olsun. Küme $F[x] := \left\{ \sum_{i=0}^{n} a_i x^i : a_i \in F, n \geq 0 \right\}$ F üstüne polinom halka olarak isimlendirilir. $F[x]$'in bir elemanına F üstüne bir polinom adı verilir. Bir $f(x) = \sum_{i=0}^{n} a_i x^i$ polinomu için tamsayı n $f(x)$'in derecesi olarak adlandırılır ve $\deg(f(x))$ ile tanımlanır. Bunun ötesinde $n.$dereceden $f(x) = \sum_{i=0}^{n} a_i x^i$ sıfır olmayan polinomu için $a_n = 1$ ise monic olarak isimlendirilir. Bir $f(x)$ polinomu $\deg(g(x)) < \deg(f(x))$ ve $\deg(h(x)) < \deg(f(x))$ olacak şekilde $f(x) = g(x).h(x)$ şeklinde yazılabiliyorsa indirgenebilir, aksi halde pozitif dereceli $f(x)$ polinomu indirgenemez polinom olarak adlandırılır.

Teorem 1.7: $f(x)$ derecesi 1'den büyük olmak üzere bir F cisminin üzerine bir polinom olsun. Yani $f(x)$ polinomunun elemanları F cisminin elemanlarından oluşsun. O zaman $F[x]/f(x)$ toplama ve çarpma işlemi ile birlikte bir halka oluşturur. Bunun ötesinde $f(x)$ indirgenemez bir polinom ise $F[x]/f(x)$ bir cisim oluşturur.
Yukarıdaki teoreme göre $F[x]/f(x)$ mod $f(x)$'e göre indirgeme sonucu oluşan sistemi ifade etmektedir. Sonuç olarak bir indirgenemez polinoma göre mod alma işlemi sonucunda cisim elde edilebilir ve bu cisimler F_{p^n} ya da $GF(p^n)$ cisimleri olarak isimlendirilir ve p^n eleman içerir.

Örnek 1.4: $Z_2[x]/(1+x+x^3)$ cisminin elemanlarını yani F_{2^3} cismini oluşturmak için x'in üslerini $\text{mod}(1+x+x^3)$ işlemi ile elde edelim.

$$x^0 \equiv 1$$
$$x^1 \equiv x$$
$$x^2 \equiv x^2$$
$$x^3 \equiv x+1$$
$$x^4 \equiv x^2 + x$$
$$x^5 \equiv x^2 + x + 1$$
$$x^6 \equiv x^2 + 1$$
$$x^7 \equiv 1$$

Aynı tabloyu üç boyutlu vektör $\alpha = [0,1,0]$ kullanılarak da elde edilebilir.

$$\alpha^0 = 1$$
$$\alpha^1 = \alpha$$
$$\alpha^2 = \alpha^2$$
$$\alpha^3 = \alpha + 1$$
$$\alpha^4 = \alpha^2 + \alpha$$
$$\alpha^5 = \alpha^2 + \alpha + 1$$
$$\alpha^6 = \alpha^2 + 1$$
$$\alpha^7 = 1$$

Böylece α'nın ilk 7 kuvveti F_{2^3}'de birbirinden farklıdır ve sadece F_{2^3}'de birbirinden ve sıfırdan farklı 7 eleman vardır.

Örnek 1.5: $Z_2[x]/(1+x+x^3)$ cisminde $a.b = (110).(111)$ sonucunu elde edelim.

$a = \alpha^4, b = \alpha^5$ şeklinde yazılabileceğinden $a.b = \alpha^4.\alpha^5 = \alpha^9 = \alpha^2.\alpha^7 = \alpha^2.1 = (100)$ olarak elde edidilebilir (α'nın üsleri mod 7'ye göre indirgeme yapılarak elde edilir çünkü $\alpha^7 = 1$ dir).

Örnek 1.5'de bir cisimde iki polinomun çarpımı için bir tablo oluşturma yolu ile elde edilmesi gösterilmiştir. Diğer yandan iki polinomun bilgisayar uygulamalarında

çarpımının sonucunun elde edilebilmesi için iyi bir algoritma aşağıda örnekte olduğu gibi verilebilir. Bu algoritma kısaca şöyle verilebilir.

1-) x ile çarpma çarpılacak polinomun 1 bit sola ötelemesi demektir.

2-) Ancak bir önceki sonucun en anlamlı biti bir ise 1 bit sola öteleme işleminden sonra elde edilen değerin modulo ile XOR işlemine tabi tutulmalıdır.

Örneğin bir değerin x^2 ile çarpımı söz konusu ise o zaman değerin iki defa arka arkaya x ile çarpımı ile istenilen sonuç elde edilir. Diğer yandan eğer $x^2 + x$ ile çarpım söz konusu ise x^2 ve x ile çarpım sonuçları elde edilir. Daha sonra bu elde edilen değerler XOR işlemine tabi tutulur.

Örnek 1.6: $P_1 = (x^5 + x^2 + x)$ ile $P_2 = (x^7 + x^4 + x^3 + x^2 + x)$ polinomlarının $GF(2^8)$ de modulo $(x^8 + x^4 + x^3 + x + 1)$ kullanılarak çarpımını elde edelim.

Üsler	**İşlem**	**Yeni Sonuç**	**İndirgeme**
$x^0 \otimes P_1$		$\left(x^7 + x^4 + x^3 + x^2 + x\right)$	Yok
$x^1 \otimes P_1$	$x \otimes \left(x^7 + x^4 + x^3 + x^2 + x\right)$	$\left(x^5 + x^2 + x + 1\right)$	Var
$x^2 \otimes P_1$	$x \otimes \left(x^5 + x^2 + x + 1\right)$	$\left(x^6 + x^3 + x^2 + x\right)$	Yok
$x^3 \otimes P_1$	$x \otimes \left(x^6 + x^3 + x^2 + x\right)$	$\left(x^7 + x^4 + x^3 + x^2\right)$	Yok
$x^4 \otimes P_1$	$x \otimes \left(x^7 + x^4 + x^3 + x^2\right)$	$\left(x^5 + x + 1\right)$	Var
$x^5 \otimes P_1$	$x \otimes \left(x^5 + x + 1\right)$	$\left(x^6 + x^2 + x\right)$	Yok
$P_1 \times P_2 = \left(x^6 + x^2 + x\right) + \left(x^6 + x^3 + x^2 + x\right) + \left(x^5 + x^2 + x + 1\right) = x^5 + x^3 + x^2 + x + 1$			

Örnek 1.6'da verilen indirgenemez polinom $(x^8 + x^4 + x^3 + x + 1)$ AES blok şifresinde kullanılmaktadır. Bu indirgenemez polinom ilkel bir polinom olmadığı için tablo oluşturarak çarpma işlemi yapmak istendiğinde Örnek 1.4'de verildiği gibi önce cismin ilkel bir eleman kullanılarak üretilmesi gerekir. Ancak $\alpha = [0, 0, 0, 0, 0, 0, 1, 0]$ elemanı bu indirgenemez polinom ile tanımlı F_{2^8} ya da $GF(2^8)$ cisminde ilkel eleman

olmadığı için tüm cismin elemanları bu eleman kullanılarak elde edilemez. Örneğin $\alpha+1=[0,0,0,0,0,0,1,1]$ elemanı ilkel bir elemandır ve bu eleman yolu ile tüm cismin elemanları üretilebilir ve daha sonra bu cisim üzerinde çarpma işlemi, ters alma işlemi gibi işlemler kolaylıkla yapılabilir.

Bir $GF(2^n)$ cismi üzerinde toplama işlemi bu cismin karakteristiği 2 olduğu için XOR işlemi ile kolaylıkla gerçekleştirilebilir. Buna ek olarak dördüncü dereceden $GF(2^4)$ cismini tanımlayabilecek 3 tane indirgenemez polinom varken; $(x^4+x+1, x^4+x^3+1, x^4+x^3+x^2+x+1)$ $GF(2^8)$ cismini tanımlayabilecek 30 tane indirgenemez polinom bulunmaktadır. Örnek 1.7'de $(x^8+x^4+x^3+x+1)$ indirgenemez polinomu ile tanımlı $GF(2^8)$ cisminde çarpma işlemi verilen algoritma ile elde edilmektedir.

Örnek 1.7: "AD" Hexadecimal (onaltılık) byte değeri ile "02" Hexadecimal byte değerinin çarpımını hem tablo yöntemi hem de verilen algoritma ile elde edelim.
$\alpha+1=[0,0,0,0,0,0,1,1]$ ilkel elemanı kullanılarak;

$$'AD' = 10101101 = (\alpha+1)^{118}$$

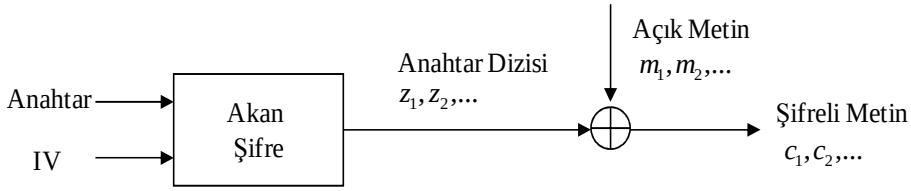
$$'02' = 00000010 = (\alpha+1)^{25}$$

olarak elde edilir. Bu iki byte değerinin çarpımı $(\alpha+1)^{118}.(\alpha+1)^{25} = (\alpha+1)^{143}$ şeklinde elde edilebileceğinden çarpım sonucu Hexedecimal "41" ya da "01000001"şeklindedir. Eğer çarpım sonucunun üs değeri 255 değerinden yüksek bir değer elde edilirse, mod 255'e göre indirgeme yapılarak sonuç elde edilir. Çünkü $(\alpha+1)^{255}=1$ dir. Verilen algoritmayı kullanarak $'AD' \otimes '02'$ sonucu 'AD' Hexadecimal değerinin 1 sola ötelenmesi ve bu byte değerinin en anlamlı biti 1 olduğu için modulo değeri olan Hexadecimal değer '1B' ile XOR işlemine tabi tutulması gerekir.

$$'AD' = 10101101 \xrightarrow{1\ sola\ öteleme} 01011010 \rightarrow 01011010 \oplus 00011011 \rightarrow 01000001 = "41"$$

1.2. Akış Şifreler

Akış şifreler girdi olarak alınan bir anahtar (K) ve başlangıç vektörü (IV-Initialization Vector) ile mümkün olduğu kadar uzun periyotlu ve rassal gözüken anahtar dizilerini üretir ve elde ettiği anahtarı bir fonksiyona (genellikle XOR işlemi) sokarak şifreli metni elde eder (Turan, 2008). Şekil 1.3'de bir akış şifrenin şifreli metin üretme safhası ile beraber örnek gösterimi verilmiştir.



Şekil 1.3. XOR Fonksiyonu ile Akış Şifre Gösterimi (Turan, 2008)

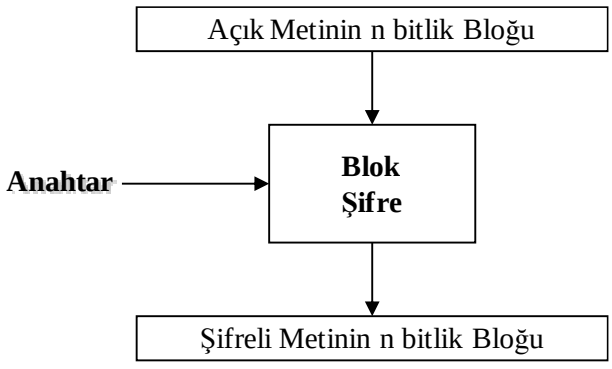
Önemli akış şifreleme algoritmalarına örnek olarak GSM de kullanılan A5/1 ve E_0 (SIG Bluetooth, 2003) verilebilir. Son 7 yıl içerisinde akış şifreleme algoritmalarında büyük gelişim meydana gelmiştir. ECRYPT (European Network of Excellence for Cryptology-Kriptoloji için Avrupa Mükemmellik Ağı)'in 2004'te eStream yarışmasında 34 aday algoritmadan Eylül 2008 itibariyle 7 tanesi kullanılabilir olarak seçilmiştir ancak bunların standart olmaları için erken olduğu belirtilmektedir. Bu şifrelerden yazılım profiline uygun olanlar HC-128 (Wu, 2007), Rabbit (Boesgaard, Vesterager, vd. 2005), Salsa20/12 (Bernstein, 2005), SOSEMANUK (Berbain, Billet vd., 2005) iken donanım profiline uygun olanlar Grain (Hell, Johansson vd., 2005), MICKEY (Babbage, Dodd, 2005) ve Trivium (De Canniere, Preneel, 2005)'dur. Yukarıdaki şifrelerin tasarımları ile beraber akış şifrelerde 5 farklı tasarım stratejisinin olduğu söylenebilir. Bunlar;

- LFSR (Linear Feedback Shift Registers- Doğrusal Geribeslemeli Ötelemeli Saklayıcılar) Tabanlı Akış Şifreler (Örnek Snow (Ekdahl, Johansson, 2003), E_0),

- NFSR (Nonlinear Feedback Shift Registers- Doğrusal Olmayan Geribeslemeli Ötelemeli Saklayıcılar) Tabanlı Akış Şifreler (Örnek Trivium),
- Blok Şifre Tabanlı Akış Şifreler (Örnek LEX (Biryukov, 2005)),
- Karıştırma Tabanlı Akış Şifreler (Örnek RC4 (Forouzan, 2008)),
- Hash Fonksiyon Tabanlı Akış Şifreler (Örnek Salsa20) şeklindedir.

1.3. Blok Şifreler

Bir blok şifre n-bit uzunluğunda açık metin bloğunu ve m-bit uzunluğunda anahtarı alarak n-bit uzunluğunda şifreli metni çıkış olarak üreten simetrik anahtarlı bir kriptografik yapıdır. Dolayısıyla bir blok şifre n-bit blokları ayrı ayrı şifreler (Sakallı, 2006; Aslan, 2008). Örneğin daha önce belirtilen Vigenere şifresi anahtar ve blok uzunluğu eşit olan bir blok şifredir. Şekil 1.4, bir blok şifrenin genel yapısını göstermektedir.



Şekil 1.4. Bir Blok Şifrenin Genel Gösterimi (Ekdahl, 2003)

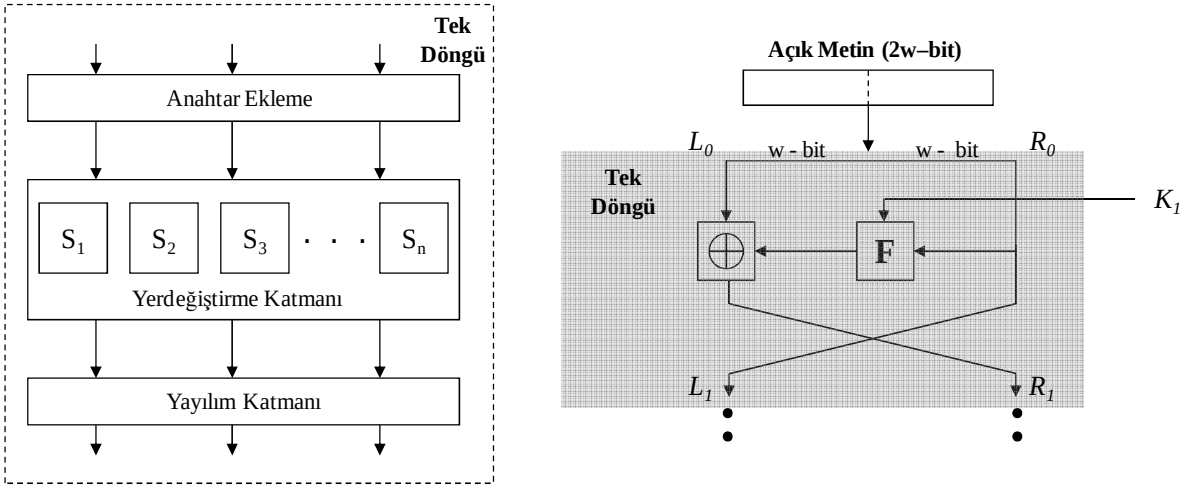
Önemli blok şifreleme algoritmalarına örnek olarak DES, AES (Advanced Encryption Standard) (US National Institute of Standards and Technology, 2001),

ARIA (Kwon, Kim, vd., 2004), Camellia (Aoki, Ichikawa, vd., 2001), Serpent (Biham, Anderson, Knudsen, 1998) verilebilir. 1997 yılında NIST (National Institute of Standards and Technology) blok uzunluğu 64-bit ve anahtar uzunluğu 56-bit olan DES algoritması yerine anahtar uzunluğu için 128, 192 ve 256-bit seçenekleri ile blok uzunluğu 128-bit olan AES olarak isimlendirilecek yeni bir şifreleme algoritması için yarışma düzenlemiş ve 2001 yılında Rijndael şifresi 21 şifreleme algoritması arasından AES olarak seçilmiştir.

1.3.1. Blok Şifre Mimarileri ve Blok Şifrelerde Anahtar Genişletme Algoritmaları

Blok şifreleme algoritmalarının kökleri Claude Shannon'un (Shannon, 1949) ortaya koyduğu karıştırma ve yayılım ilkelerine dayanmaktadır. Karıştırma açık metin ile şifreli metin arasındaki ilişkiyi gizlemeyi amaçlarken, yayılım açık metindeki izlerin şifreli metinde sezilmemesini sağlama amacındadır. Blok şifrelerde karıştırma yer değiştirme ile sağlanırken yayılım doğrusal dönüşümler ile gerçekleştirilir.

Blok şifrelerin tasarımında temel olarak iki mimari kullanılmaktadır. Bu mimariler Feistel ve SPN (Substitution Permutation Networks-Yerdeğiştirme Permütasyon Ağları) mimarisidir. Bu iki mimari arasındaki en temel fark şifreleme işleminde o anki bloğun işlenmesinde ortaya çıkar. Feistel mimarisi kullanan şifreler o anki bloğun yarısını işlerken, SPN mimarisini kullanan şifrelerde o anki bloğun tümü işlenir. Feistel mimarisinin bir avantajı Feistel ağı döngü fonksiyonunun tersi alınabilir ihtiyacının olmamasıdır. Bu da şifrenin tasarımında büyük bir esneklik sağlar. DES şifreleme algoritması Feistel mimarisine, AES şifreleme algoritması ise SPN mimarisine örnek olarak verilebilir. Her iki mimari de yerdeğiştirme ve doğrusal dönüşüm işlemlerini kullanır. Şekil 1.5 tek döngülük SPN ve Feistel mimarisini göstermektedir.

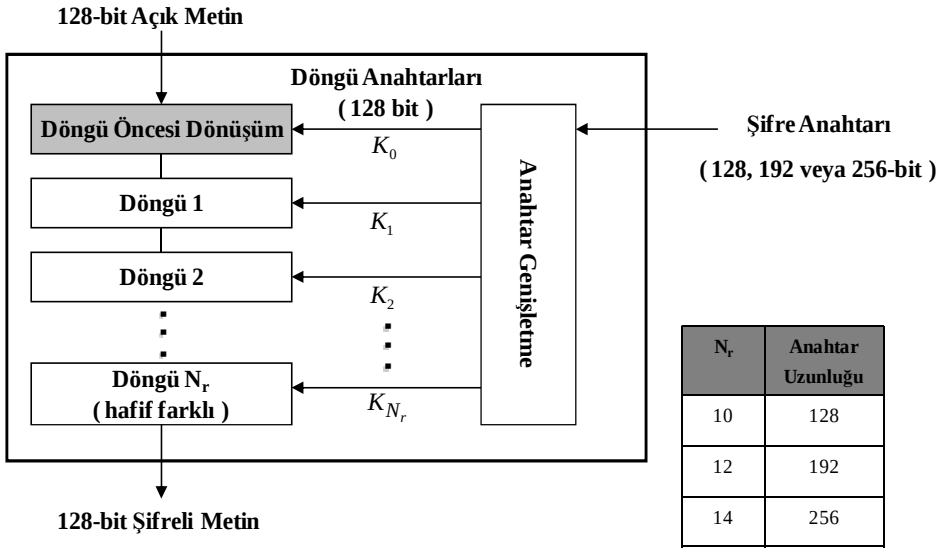


a-) SPN Mimarisi

b-) Feistel Mimarisi

Şekil 1.5. Tek Döngülük SPN ve Feistel Mimarisinin Gösterimi

Blok şifreler döngü adı verilen adımlardan oluşur ve her adım aynı yer değiştirme ve doğrusal dönüşüm mekanizmalarından oluşur. Bu adımlarda ya da döngülerde meydana gelecek simetriyi bozmak için ise bir anahtar genişletme algoritması ile gizli anahtardan elde edilen farklı anahtarlar her döngüde bloğa eklenir. Örneğin AES blok şifresi 128-bit anahtar seçeneği ile 10 döngüde, 192-bit anahtar seçeneği ile 12 döngüde ve 256-bit anahtar seçeneği ile 14 döngüde şifreleme yapmaktadır. Şekil 1.6, kabaca AES blok şifresinin genel yapısını göstermektedir.



N_r	Anahtar Uzunluğu
10	128
12	192
14	256

Şifre Anahtarı ve Döngü Sayıları Arasındaki İlişki

Şekil 1.6. AES Blok Şifresinin Genel Yapısı (Stinson, 2002)

Anahtar genişletme algoritmaları gizli bir anahtardan N_r döngüye sahip bir blok şifre için N_r adet farklı alt anahtar üretmek amacındadır. Genellikle şifre tasarımcıları şifrede kullanılan elemanların özellikleri ile anahtar genişletme algoritmalarının tasarımını gerçekleştirirler. Kötü tasarıma sahip bir anahtar genişletme algoritması şifreye karşı ilişkili anahtar saldırısı (Biham, 1994) (related-key attacks) gibi saldırıların yolunu açabilir. Bu tür saldırılar gerçek hayatta her ne kadar pratik olmasalar da AES-192 (192-bit anahtar kullanan AES blok şifresi) ve AES-256 (256-bit anahtar kullanan AES blok şifresi) için ilişkili anahtar saldırıların ne kadar faydalı olduğu (Biryukov, Khovratovich, 2009; Biryukov, Khovratovich, Nikolic, 2009) çalışmalarında gösterilmiştir. Bunun temel nedeni olarak AES-192 ve AES-256 versiyonlarındaki anahtar planlama algoritmasının AES-128 (128-bit anahtar kullanan AES blok şifresi) versiyonuna göre daha düşük yayılım özelliği sağlaması olarak verilebilir. Ayrıca zaman karmaşıklığı açısından Biryukov A. vd. (Biryukov, Dunkelman, Keller, 2009) 10 döngüye kadar bir AES algoritmasına pratik bir saldırıyı göstermeye çalışmışlardır.

1.4. Kriptanaliz

Kriptanaliz (Biryukov, 1999) kısaca şifre kırma bilimi olarak ifade edilebilir. Diğer bir deyişle açık metni ya da anahtarı elde etme bilimidir. Bir şifreleme sisteminin zayıf ve güçlü yanlarını ortaya çıkarmak için kullanılabileceği gibi bir şifrenin kırılarak açık metni ya da anahtarı elde etme şeklinde kötü niyetli olarak ta kullanılabilir.

Bir kriptografik yapının tasarımı için ilk ve en önemli kural Kerckhoffs prensibidir. Bu prensibe göre: *"Bir kriptografik yapının güvenliği sadece anahtarın gizliliğine bağımlı olmalıdır. Algoritmanın gizliliğine bağımlı olmamalıdır"*. Diğer bir deyişle kriptografik yapının bileşenleri herkese açık olmalıdır. Buna ek olarak kriptanaliz saldırısı için saldırganın ya da kriptanalistin sahip olabileceği veriler olabilir. Bu sahip olabileceği verilere göre saldırı modellerinden birini seçebilir. Bu saldırı modellerinden en yaygın olanları aşağıda verilmiştir:

- *Sadece şifreli metin saldırısı* (Ciphertext only attack): Sadece şifreli metin saldırısında, saldırgan sadece bazı şifreli metinlere erişime sahiptir. O şifreli metne ilişkili anahtarı ve açık metni elde etmeye çalışır. Bu saldırı modelindeki varsayım saldırganın şifreleme algoritmasını bildiği ve şifreli metinleri elde edebildiği üzerine dayalıdır. Sadece şifreli metin saldırısı en olası saldırıdır. Çünkü saldırgan bu saldırı için sadece şifreli metinlere ihtiyacı vardır. Bir şifreli mesajın düşman tarafından çözülmesini etkisiz hale getirmek için bir şifrenin bu saldırı tipine karşı oldukça dirençli olması gerekir.
- *Bilinen açık metin saldırısı* (Known plaintext attack): Bilinen açık metin saldırısında saldırgan şifreli metinlere ve bazı açık metin ve şifreli metin çiftlerine erişime sahiptir.
- *Seçilmiş açık metin saldırısı* (Chosen plaintext attack): Seçilmiş açık metin saldırısı bilinen açık metin saldırısına benzerdir. Fakat açık metin şifreli metin çiftleri saldırgan tarafından seçilmiştir.
- *Seçilmiş şifreli metin saldırısı* (Chosen ciphertext attack): Seçilmiş şifreli metin saldırısı seçilmiş açık metin saldırısına benzerdir. Bu saldırıdan farklı olarak saldırgan bazı şifreli metinleri seçebilir. Buna ek olarak bir açık metin ve şifreli metin çifti elde edebilmek için deşifreleme yapabilir.

Kriptanalitik saldırılar yukarıda verilen saldırı modellerinden birini kullanabilir. Önemli bazı kriptanalitik saldırı tipleri doğrusal kriptanaliz, diferansiyel kriptanaliz, kesik diferansiyel kriptanaliz, imkânsız diferansiyel kriptanaliz, çoklu set saldırıları, interpolasyon saldırısı gibi cebirsel saldırılar, boomerang saldırısı, kare (rectangular) saldırısı, yan kanal saldırısı şeklinde verilebilir (Z'aba, 2010). Bir şifrenin gücü değerlendirilirken verilen saldırılar geniş anahtar arama saldırısı (exhaustive key search) ile karşılaştırılır.

Geniş anahtar arama saldırısı tüm olası anahtarları deneyerek doğru olanı bulmaya dayanır. Dolayısıyla şifrenin erişilebilecek maksimum gücü tüm anahtarların denenmesi için gerekli süre ile ilişkilidir. Bununla beraber bir istisnai durum tek kullanımlık şerit (one time pad) için geçerlidir. Tek kullanımlık şerit kesinlikle güvenlidir ve anahtar uzunluğunun bilginin uzunluğuna eşit olduğu Vigenere şifresi olarak tanımlanabilir. Bilgi bitleri $m = m_1 m_2 ... m_l$ ve anahtar bitleri $k = k_1 k_2 ... k_l$ şeklinde tanımlanırsa tek kullanımlık şerit ile elde edilecek şifreli metin bitleri açık metin bitleri ile anahtar bitlerinin XOR sonucu elde edilir:

$$c_t = m_t \oplus k_t \qquad t = 1....l$$

Tek kullanımlık şeridin güvenliği için önemli şart anahtar bitlerinin tümüyle rastlantısal olması ve sadece bir kere kullanılmasıdır. Buradan yola çıkarak akış şifreleri sözde rastlantısal sayı üreteci ya da anahtar dizisi üreteci olarak görülebilir. Dolayısıyla akış şifre kullanılmasının önemli nedeni tek kullanımlık şerit için bir mesaj uzunluğunda büyük uzunlukta anahtar üretme ve bunu güvenli bir kanal üzerinden iletme yerine küçük boyutlu bir anahtarı şifre için besleme olarak kullanarak rastlantısal görünümlü bir anahtar dizisi üretmektir.

1.5. İstatistiksel Testler

Şifreleme algoritmalarının güvenilirliği, rassal sayı üreteçlerinin ürettiği sayılara bağlıdır. Bu sayılar istatistiksel olarak rassal ise, yani önceki çıkışlara bakarak daha sonrakiler tahmin edilemezse üretecimizin istatistiksel özelliği iyi demektir. Başka bir deyişle iyi bir şifreleme iyi bir Rassal Sayı Üreteci (RSÜ) gerektirir. RSÜ'leri kendi aralarında gerçek ve sözde RSÜ'ler şeklinde ikiye ayırmak mümkündür. Uygulamanın amacına göre bu iki yapıdan biri tercih edilmektedir.

Rassal sayı üreteçleri, çıkışı rassal sayılar olan sistemlerdir. Rassal sayı dizileri, aralarında korelasyon bulunmayan, istatistiksel olarak birbirinden bağımsız sayılardan oluşurlar. Rassal sayılara, bu özelliklerden dolayı, rassal süreçlerin modellerinin oluşturulması, sayısal analiz yöntemleri, Monte Carlo metodu uygulamaları gibi bilgisayar uygulamalarında ihtiyaç duyulmaktadır. Bununla beraber, sayısal haberleşmenin çok hızlı bir şekilde yaygınlaşması ve bu nedenle gizlilik içeren verilerin internet gibi herkese açık sistemler kullanılarak iletilmesi şifrelemenin ve bunun bir sonucu olarak rassal sayılara olan ihtiyacın artmasına neden olmuştur. Böylece rassal sayılar ve onları üreten rassal sayı üreteçleri büyük önem kazanmıştır (Yalçın, Suykens, Vandewalle, 2004).

Sayı üreteçlerinin çıkışının rassal olduğu, matematiksel olarak kanıtlanamamasına rağmen, geçerli istatistiksel testleri uygulayarak, sayı dizilerinin rassal olmadığını veya bunun tersini söyleyebiliriz. Bu testler, üretecin çıkışının gerçek bir rassal diziden beklenenleri karşılayıp karşılamadığını söyler. Ayrıca testlerin sonuçlarına bakılarak rassal sayı üretecinin kalitesi hakkında yorum yapılabilir. Bir sayı dizisinin rassal olduğunu söylemek için, tüm testlerden geçmesi gerekir. Sadece bir tane test başarısız olsa bile dizi rassal kabul edilemez.

Bilinen testlerden başlıca iki tanesi, Ulusal Standartlar ve Teknoloji Enstitüsü (National Institute of Standards and Tecnology-NIST) tarafından yayınlanan FIPS (Federal Information Processing Standards-Federal Bilgi İşleme Standartları) 140-1 ve NIST 800-22'dir (National Institute of Standard and Technology, 2001). FIPS 140-1 testi blok uzunluğu küçük olan verilerin testinde kullanılır. NIST 800-22 testi ise uzun

bloklardan oluşan verileri test etmek için kullanılır ve FIPS 140-1 testine göre kriterleri daha zorlayıcıdır.

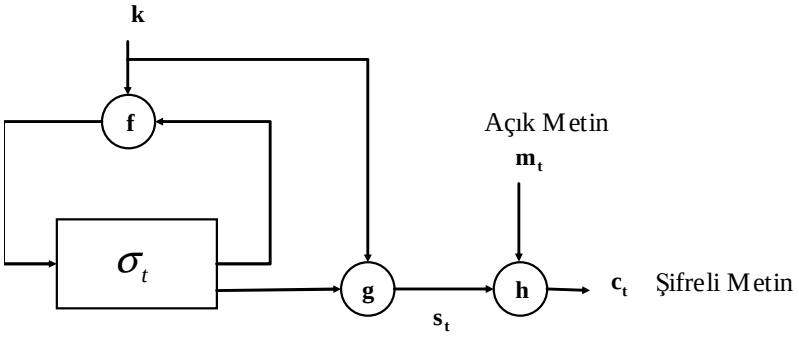
Bu nedenle kullanım amacına uygun olarak bu testlerden biri tercih edilebilir (Demirkol, 2007).

2. AKIŞ ŞİFRELERİN GENEL YAPISI

Akış şifreler (Stream Cipher) (Menezes, Oorschot, Vanstone, 1997), (Canteaut, 2005; Günther, 1988) açık metnin bir seferde bir karakterine (genellikle ikili sayı) zamanla değişen bir şifreleme fonksiyonu uygulayarak açık metin karakterlerini ayrı ayrı şifreler. Blok şifreler ise açık metnin bir bloğunu sabit bir şifreleme fonksiyonu kullanarak şifreleme işlemini gerçekleştirir.

Akış şifreler, *eşzamanlı* (*senkron*) ve *asenkron* akış şifreler olmak üzere ikiye ayrılabilir. Eşzamanlı akış şifreler *sonlu durum makinesi*'dir (*finite state machine*). Anahtar dizisi (keystream), açık metin ve şifreli metinden bağımsız olarak gizli anahtardan üretilir ve anahtar dizisi ve şifreli metnin üretimi 2.1'deki eşitliklerle $t \geq 0$ olmak üzere tanımlanabilir.

$$\begin{aligned} \sigma_{t+1} &= f(\alpha_t, k), \\ s_t &= g(\sigma_t, k), \\ c_t &= h(s_t, m_k) \end{aligned} \qquad (2.1)$$



Şekil 2.1. Senkron Şifre Yapısı (Ekdahl, 2003)

Şekil 2.1, eşzamanlı bir şifrenin yapısını göstermektedir. Burada σ_0 başlangıç durumunu (initial state) (anahtar k'ya bağlı olabilir), k anahtarı, f diğer durum fonksiyonunu (next state function), g anahtar dizisi s_t'yi üreten fonksiyonu, h ise açık

metin ile anahtar dizisini (keystream) birleştirerek şifreli metin c_t'yi üreten çıkış fonksiyonunu temsil etmektedir. h çıkış fonksiyonu yerine XOR fonksiyonu kullanılırsa bu tür şifrelere *toplamsal akış şifreler (additive stream cipher)* adı verilir. Bu tür şifrelere *sözde rassal sayı üreteçleri* (*pseudo random number generator*) ya da *anahtar dizisi üreteçleri* (*keystream generator*) de denmektedir. Bilindiği gibi XOR işleminin tersi kendisidir ($h = h^{-1}$). Bundan dolayı bu tür şifrelerde şifreleme ile deşifreleme aynıdır ve bu işlemin kullanılması karşımıza kullanışlı bir özellik olarak çıkar.

Bu tür şifreler iletim hatalarına karşı zayıf değillerdir çünkü her karakter bağımsız olarak şifrelenmektedir. Ancak bir saldırgan şifreli bir metni silebilir ya da değiştirebilir. Dolayısıyla gönderilen mesajın kimlik denetiminin (authentication) yapılmasını sağlayan mekanizmalara gereksinim vardır. Aynı nedenden dolayı senkronizasyon bozulabilir. Gönderen ve alıcı arasında iyi bir senkronizasyon sağlanmalı ve senkronizasyon bozulmasını sezecek mekanizmalar kullanılmalıdır.

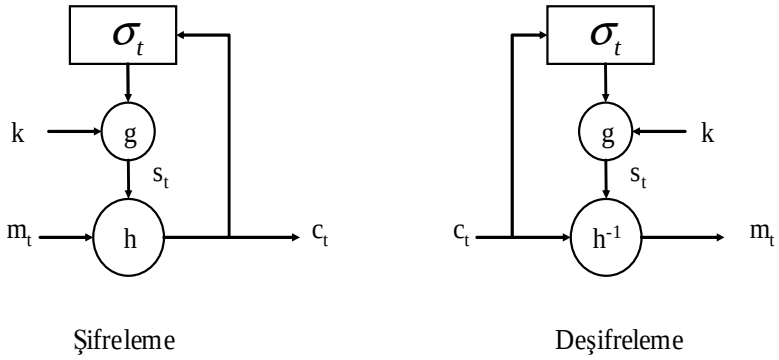
Toplamsal akış şifreler (additive stream cipher), One Time Pad (Tek Kullanımlık Şifreler) şifrelere anlayış olarak çok benzemektedir. Bu yöntemden farklı olarak gizli anahtar başlangıç durumu ya da üreteci beslemek için kullanılır ve sözde rassal (pseudo-random) bitler üretilir.

Diğer akış şifre tipi olan asenkron akış şifreler de sonlu durum makinesidir. Ancak anahtar dizisi, sabit uzunluktaki bir önceki şifreli metinlerin ve anahtarın bir fonksiyonu ile elde edilir. Asenkron şifreler de şifreli metin üretme işlemi 2.2'deki eşitliklerle $t \geq 0$ olmak üzere tanımlanabilir.

$$\begin{aligned} \sigma_t &= (c_{t-v}, c_{t-v+1}, ..., c_{t-1}), \\ s_t &= g(\sigma_t, k), \\ c_t &= h(s_t, m_t) \end{aligned} \qquad (2.2)$$

Burada σ_0 başlangıç durumunu (initial state), k anahtarı, g anahtar dizisi s_t'yi üreten fonksiyonu, h ise açık metin ile anahtar dizisini (keystream) birleştirerek şifreli metin c_t'yi üreten çıkış fonksiyonunu temsil etmektedir. Başlangıç durumu

$\sigma_0 = (c_{-v}, c_{-v+1}, ...c_{-1})$ herkes tarafından bilinebilir. Asenkron akış şifrenin yapısı Şekil 2.2'de görülmektedir.



Şekil 2.2. Asenkron Şifrenin Genel Yapısı (Ekdahl, 2003)

Şifreleme ve deşifreleme senkron şifrelere göre şekilde görüldüğü üzere farklılık göstermektedir. Asenkron şifrelerde şifreleme v şifreli metin sembolüne bağlı olduğu için bir iletim hatası durumunda v sembol sonra şifrenin tekrar senkronizasyonu mümkün olacaktır. Böyle bir durum söz konusu olduğunda öteki v sembol hatalı olacaktır. Yani hata yayılması (error propagation) senkron olan şifrelere göre kötüdür. Ancak senkronizasyon düşünüldüğünde asenkron şifreler senkron olanlara göre daha iyidir. Çünkü bu tür şifrelerde v doğru şifreli metin sembolü elde edildikten sonra senkronizasyon kendiliğinden sağlanacaktır. Senkron şifreler ise senkronizasyonu tekrar sağlayamazlar.

Tarihsel olarak akış şifre türleri ve genel özellikleri aşağıdaki tabloda özetlenmiştir (Akış Şifreler, 2011).

Akış Şifreleri	Tarih	Hız (cycles/byte)	Anahtar Uzunluğu	(Bits) Başlangıç Vektörü	İç Durum	Saldırılar En Bilinen	Saldırılar Hesaplanabilir Karmaşıklık
A5/1	1989	Voice (W_{pleine})	54	114	64	Aktif KPA ya da KPA Zaman-Bellek Tradeoff	~2 saniye ya da $2^{39.91}$
A5/2	1989	Voice (W_{pleine})	54	114	64?	Aktif	4.6 milisaniye
FISH	1993	Oldukça Hızlı ($W_{çok}$)	Büyük	?	?	Bilinen Açık Metin Saldırısı	2^{11}
Grain	2004	Hızlı	80	64	160	Anahtar Türetme	2^{43}
HC-256	2004	4 (W_{PIII})	256	256	65536	?	?
ISAAC	1996	2.375 (W_{64-bit}) 4.6875 (W_{32-bit})	8-8288 genellikle 40-256	N/A	8288	(2006) İlk Çevrim Zayıf Başlangıç Durumu Türetme	4.67×10^{1240} (2001)
MUGI	1998-2002	?	128	128	1216	N/A (2002)	$\sim 2^{82}$
PANAMA	1998	2	256	128?	1216?	Hash Çarpışmaları (2001)	2^{82}
Phelix	2004	8'den fazla (W_{x86})	256 + 128-bit (Durumsal)	128?	?	N/A (2005)	N/A (2005)
Pike	1994	0.9 x FISH ($W_{çok}$)	Huge	?	?	N/A (2004)	N/A (2004)
Py	2004	2.6	8-2048? genellikle 40-256?	64	8320	Kriptanalitik Teori(2006)	2^{75}
Rabbit	2003	3.7(W_{P3}) 9.7(W_{ARM7})	128	64	512	N/A (2006)	N/A (2006)
RC4	1987	Etkileyici	8-2048 genellikle 40-256	8	2064	Shamir İlk-Byte'lar Anahtar Türetme ya da KPA	2^{13} ya da 2^{33}
Salsa20	2004	8.91 (W_{G4}) 16.44 (W_{P4})	256 + 64-bit	512	512 + 384 anahtar+IV+index	Diferansiyel (2005)	N/A (2005)
Scream	2002	4 - 5 ($W_{çok}$)	128 + 128-bit	32?	64-bit çevrim fonksiyonu	?	?
SEAL	1997	Çok Hızlı (W_{32-bit})	?	32?	?	?	?
SNOW	2003	Çok İyi (W_{32-bit})	128 ya da 256	32	?	?	?
SOBER-128	2003	?	128'den fazla	?	?	Mesaj Taklidi	2^{-6}
SOSEMANUK	2004	Çok İyi (W_{32-bit})	128	128	?	?	?
Trivium	2004	4 (W_{x64}) 8 (W_{IG})	80	80	288	Brute Force Saldırısı (2006)	2^{135}
Turing	2000-2003	5.5 (W_{x86})	?	160	?	?	?
VEST	2005	42 (W_{ASIC}) 64 (W_{FPGA})	genellikle 80-256	80-256	256 - 800	N/A (2006)	N/A (2006)
WAKE	1993	Hızlı	?	?	8192	CPA & CCA	Zayıf

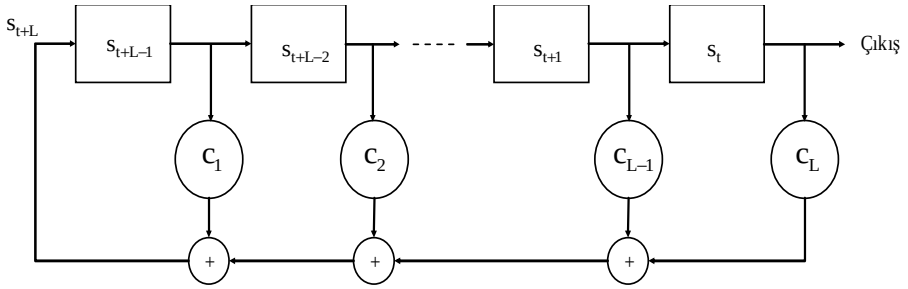
2.1. Doğrusal Geri Beslemeli Öteleyici Saklayıcılar (Linear Feedback Shift Registers)

Doğrusal Geri Beslemeli Öteleyici Saklayıcılar (Canteaut, 2005), birçok anahtar dizisi üretecinde kullanılmaktadır. Bunun nedeni olarak donanımsal uygulamalarda uygunlukları, geniş periyoda sahip olmaları, üretilen serinin iyi istatistiksel özellikler göstermesi ve cebirsel tekniklerle kolayca analiz edilebilmeleri gösterilebilir. L uzunluğunda bir LFSR (Linear Feedback Shift Register- Doğrusal Geribeslemeli Öteleyici Saklayıcılar) F_q üzerine bir sonlu durum otomatıdır (finite state automation) ve F_q elemanlarının yarı-sonsuz bir serisini üretir. Yine L uzunluğunda bir LFSR, 0'dan $L-1$'e kadar numaralanmış her biri bir bit depolayabilme yeteneği olan L tane gecikme ünitesi içerir. Ayrıca her gecikme hücresi bir giriş ve bir çıkışa ve verinin hareketini kontrol eden bir saate sahiptir.

Bir LFSR, $s=(s_t)=s_0,s_1,.....$ olmak üzere F_q üzerine derece L'ye sahip doğrusal tekrarlayan bir ilişkiye sahiptir ve bu ilişki 2.3 eşitliğinde gösterilmiştir.

$$s_{t+L}=\sum_{i=1}^{L} c_i s_{t+L-i}, \quad \forall t \geq 0 \tag{2.3}$$

L'nin katsayıları olan $c_1,c_2,....,c_L$ F_q'nun elemanlarıdır ve LFSR'ın geri besleme katsayıları olarak isimlendirilir. F_q üzerine L uzunluğunda bir LFSR Şekil 2.3 formundadır.



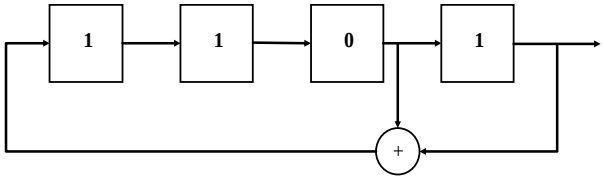
Şekil 2.3. Doğrusal Geri Beslemeli Saklayıcının Genel Yapısı (Ekdahl, 2003)

Şekil 2.3'de de görüldüğü gibi $1 \le i \le L-1$ olmak üzere durum i'nin içeriği her i için $i-1$ durumuna kaydırılır. Daha sonra $L-1$ durumunun yeni içeriği geri besleme s_{t+L} (s_{t+L} bazı sabit sayıdaki durumların modulo 2 de toplanması ile elde edilir) ile elde edilir.

Daha önce de bahsedildiği gibi LFSR'ı oluşturan her biri F_q'nun bir elemanını içeren L gecikme hücresine evreler (stages) denir. L evrenin içeriği $s_t,.....s_{t+L-1}$ ise LFSR'ın durumunu (stage) oluşturur. L durum başlangıçta keyfi olarak seçilmiş keyfi F_q'da L eleman ile yüklenir ve başlangıç durumu olan $s_0,.....s_{L-1}$'i oluşturur. Örnek 2.1'de 4 uzunluğunda bir LFSR'ın çıkış değerlerinin elde edilişini göstermektedir.

Örnek 2.1:

Şekil 2.4'de 4. dereceden ya da 4 uzunluğunda bir LFSR'ın geri besleme katsayıları $c_1 = c_2 = 0, c_3 = c_4 = 1$ ve başlangıç durumu $(s_0, s_1, s_2, s_3) = (1011)$ olmak üzere diğer durumları gösterilmektedir. Bu LFSR tarafından üretilen çıkış serisi $s_0, s_1,.....$ ise 10111100 şeklindedir. Bu LFSR için doğrusal tekrarlayan ilişki $s_{t+4} = s_{t+1} + s_t$ şeklinde verilebilir.



t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
s_t	1	0	1	1	1	1	0	0	0	1	0	0	1	1	0	1
s_{t+1}	0	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0
s_{t+2}	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1
s_{t+3}	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1

Şekil 2.4. LFSR ve Ardışık Durumları (Ekdahl, 2003)

Bir LFSR'ın çıkış dizisi onun geri besleme katsayıları ve başlangıç durumundan elde edilir. L uzunluğundaki LFSR'ın $c_1,....,c_L$ katsayıları 2.4 eşitliğinde gösterilen genellikle LFSR *geri besleme polinomu* (*feedback polynomial, connection polynomial*) ile temsil edilir.

$$P(X) = 1 - \sum_{i=1}^{L} c_i X^i \tag{2.4}$$

Geri Besleme polinomunun *karakteristik polinomu* da 2.5 denklemindeki gibi verilebilir.

$$P^*(X) = X^L P(\frac{1}{X}) = X^L - \sum_{i=1}^{L} c_i X^{L-i} \tag{2.5}$$

Örnek 2.2:

İkili bir LFSR'ın geri besleme polinomu $P(X) = X^4 + X^3 + 1$ ise onun karakteristik polinomu $P^*(X) = X^4 + X + 1$ olur.

Bir LFSR'ın geri besleme polinomunun derecesi LFSR uzunluğuna eşitse, non-singular olarak isimlendirilir (Geri besleme katsayısı c_L, 0'dan farklı ise). Bu tür bir LFSR ile üretilen dizinin periyodu $q^L - 1$'i (ikili LFSR'lar için $2^L - 1$'i) geçemez. LFSR en fazla q^L duruma sahiptir. Bunun ötesinde eğer LFSR tekilse (singular) tüm diziler eninde sonunda periyodiktir.

L: LFSR'ın boyu

Başlangıç fazı $\sigma_0 = a_{-L}, a_{-L+1},..., a_{-2}, a_{-1}$

$c_1, c_2,..., c_{L-1}, c_L \in Z_2 = \{0,1\}$

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

Buna göre; $\Rightarrow \sigma_1 : a_{-L+1}, a_{-L+2},..., a_{-1}, a_0$

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

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

Bu yinelemeli bağıntının polinomu aynı zamanda LFSR'ın karakteristik polinomudur.

Buna göre karakteristik polinom: $m(x) = x^L + c_1 x^{L-1} + c_2 x^{L-2} ... + c_{L\_1} x + c_L$

LFSR'ın bağlayıcı polinomu: $C(D) = 1 + c_1 D + c_2 D^2 + ... + c_{L-1} D^{L-1} + c_L D^L$

Bağlayıcı polinom ile karakteristik polinom arasındaki bağıntı şu şekildedir: $m(x) = x^L C\left(\frac{1}{x}\right)$

Bir LFSR, boyu *L* ve bağlayıcı polinomu *C(D)* ile belirlenir: $LFSR = \langle L, C(D) \rangle$

Örnek 2.3:

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

$$c_1 = 1, c_2 = c_3 = 0, c_4 = 1$$

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

Bu LFSR'ı çalıştırmak için başlangıç fazı olarak $\sigma_0 = (0011)$ alınırsa, periyodu 5 olan $z = (001111010110010)^{\infty}$ dizisi üretilir.

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

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

Boyu L olan bir LFSR'ın ürettiği dizinin periyodu en fazla 2^L-1 olabilir. Çünkü; LFSR'da faz olarak L uzunluğundaki vektörler alınır. L uzunluğunda 2^L-1 adet vektör vardır. LFSR'da 0 vektörü görülmezse en fazla 2^L-1 adet değişik vektör görülebilir. Yani; LFSR en fazla 2^L-1 adım sonra başlangıç noktasına geri döner.

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

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

Örnek 2.4:

$\langle L, C(D)=1+D^2+D^4\rangle$

$c_1=0, c_2=1, c_3=0, c_4=1$

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

$C(D)=1+D^2+D^4=(1+D+D^2)^2$

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

Periyod = 6

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

Periyod = 6

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

Periyod = 3

- Maksimum periyotta dizi üretebilmek için $C(D)$ polinomunun çarpanlarına ayrılamaz olması gerekir. Eğer $C(D)$ çarpanlarına ayrılamıyorsa dizinin periyodu başlangıç fazına bağlı değildir ve $C(D)$ polinomunun böldüğü $1+D^p$ polinomlarından en küçük dereceli olanın derecesi üretilen dizinin periyoduna eşittir. Buradaki *p* sayısı 2^L-1 sayısının bir bölenidir.
- Dizinin periyodunun maksimum yani 2^L-1 olması için $C(D)$ polinomunun böldüğü en küçük dereceli polinom $1+D^{2^L-1}$ olmalıdır. Bunu sağlayan $C(D)$ polinomuna *ilkel (primitive) polinom* denir.

Örnek 2.5:

$\langle 3,1+D+D^2 \rangle$ $\qquad f(x_3,x_2,x_1)=x_1 \oplus x_1x_2 \oplus x_2x_3$

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

$$z=(1001011)^{\infty}$$

Bir LFSR q^L farklı seri (sequence) üretir ve bunlar F_q üzerinde bir vektör uzayı oluştururlar. Tüm serilerin seti için geri besleme polinomu P, bir seri $(s_t)_{t\geq 0}$ olmak üzere aşağıdaki şekilde tanımlanır (Kriptolojiye Giriş Ders Notları, 2004).

Tanım 2.1: P geri besleme polinomu ile F_q üzerine L uzunluğunda bir LFSR tarafından ancak ve ancak $\deg(V) < L$ olmak üzere $V \in F_q[X]$ şeklinde 6 ifadesinde gösterildiği gibi bir polinom varsa bu LFSR tarafından $(s_t)_{t\geq 0}$ (aynı şekilde $(s_t)_{t\geq 0}$'nin üreteç fonksiyonu 2.6'daki denklemi sağlar) üretilebilir.

$$\sum_{t\geq 0} s_t X^t = \frac{V(X)}{P(X)} \tag{2.6}$$

Buna ek olarak LFSR'ın başlangıç durumu ve P'nin katsayıları kullanılarak 2.7'de gösterildiği gibi $V(X)$ elde edilebilir.

$$V(X) = -\sum_{i=0}^{L-1} X^i \left(\sum_{j=0}^{i} c_{i-j} s_j\right),\ P(X) = \sum_{i=0}^{L} c_i x_i \tag{2.7}$$

Yukarıdaki ifade göstermektedir ki L uzunluğunda P geri besleme polinomuna sahip bir LFSR tarafından üretilen serilerle $\deg(V) < L$ olmak üzere $\frac{V(X)}{P(X)}$ kesirleri arasında birebir bir ilişki vardır. Bunun sonucu olarak;

- P geri besleme polinomuna sahip bir LFSR tarafında üretilen herhangi bir seri P'nin katı olan herhangi bir geri besleme polinomuna sahip bir LFSR tarafından üretilebilir. Bu özellik hızlı ilinti (correlation) saldırılarında (fast correlation attacks) kullanılabilir.
- Diğer yandan P geri beslemeli bir LFSR tarafından üretilen herhangi bir dizi geri besleme polinomu P' olan, eğer $\frac{V(X)}{P(X)}$ kesri için $OBEB(V, P) \neq 1$ gerçekleşiyorsa, daha küçük bir LFSR tarafından üretilebilir. Böylece P, F_q üzerine indirgenemez polinom değilse P geri besleme polinomuna sahip bir LFSR'ın ürettiği tüm serilerin içerisinden biri daha kısa bir LFSR tarafından üretilebilir (Canteaut, 2005).

Bunun ötesinde, doğrusal tekrarlayan seri (linear recurring sequence) $(s_t)_{t\geq 0}$ için P_0 ve V_0 birbirlerine göre asal olmak üzere seri $(s_t)_{t\geq 0}$'in üreteç fonksiyonu $\frac{V_0(X)}{P_0(X)}$'tir ve bu seri için sabit terimi 1 olan tek bir P_0 polinomu mevcuttur. Dolayısıyla $(s_t)_{t\geq 0}$'ı üreten en kısa LFSR $L = \max(\deg(P_0), \deg(V_0)+1)$ uzunluğuna sahiptir. P_0'ın *reciprocal* polinomu, $(s_t)_{t\geq 0}$'ı üreten en kısa LFSR'ın karakteristik polinomudur ve serinin *minimal polinomu* olarak isimlendirilir. Dolayısıyla o da seri tarafından sağlanan en düşük dereceli tekrarlayan ilişkiyi elde eder. Bir doğrusal tekrarlayan dizinin minimal polinomunun derecesi serinin doğrusal karmaşıklığıdır. Doğrusal karmaşıklık seriyi üreten en kısa uzunluktaki LFSR'a karşılık gelmektedir.

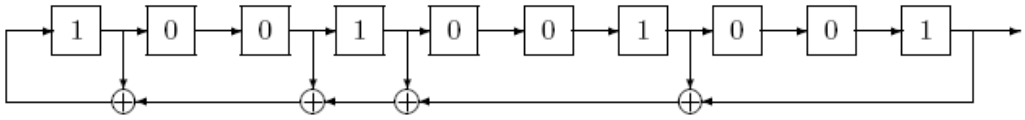
Bir $s = (s_t)_{t\geq 0}$ serisinin minimal polinomunun doğrusal karmaşıklığı $\lambda(s)$ s'in en az iki $2\lambda(s)$ sıralı bit bilgisi ile *Berlekamp –Massey algoritması* (Menezes, Oorschot, Vanstone, 1997; Canteaut, 2005) kullanılarak elde edilebilir.

Örnek 2.6:

Geri besleme polinomu $P(X) = X^{10} + X^7 + X^4 + X^3 + X + 1$ ve başlangıç durumu $s_{0,}.....s_9 = 1001001001$ olan Şekil 2.5'de gösterilen LFSR'ı düşünelim. Bu LFSR tarafından üretilen üreteç fonksiyonu aşağıda verilmiştir.

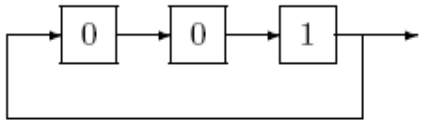
$$\sum_{t\geq 0} s_t X^t = \frac{V(X)}{P(X)}$$

V, P'nin katsayıları ve başlangıç durumundan faydalanılarak bulunabilir. Sonuç olarak $V(X) = X^7 + X + 1$ olarak bulunur.



Şekil 2.5. 10 Uzunluğuna Sahip Bir LFSR (Ekdahl, 2003)

Bundan dolayı $\sum_{t\geq 0} s_t X^t = \frac{X^7 + X + 1}{X^{10} + X^7 + X^4 + X^3 + X + 1} = \frac{1}{X^3 + 1}$ olarak bulunur. Bu da göstermektedir ki $(s_t)_{t\geq 0}$, Şekil 2.6'da gösterilen $P_0(X) = X^3 + 1$ geri beslemeli bir polinom tarafından da üretilecektir. Serinin minimal polinomu $X^3 + 1$ ve karmaşıklığı da 3'e eşittir. Örnek 2.7, $V(X)$ polinomunun X^7 teriminin katsayısının bulunuşunu göstermektedir.



Şekil 2.6. Bir Önceki Şekilde Görülen LFSR ile Aynı Seriyi Üreten Uzunluğu 3 olan LFSR (Ekdahl, 2003)

Örnek 2.7:

$V(X) = X^7 + X + 1$ polinomunda X^7 teriminin katsayısının 1 olduğunu gösterelim.

$$V(X) = -\sum_{i=0}^{L-1} X^i \left(\sum_{j=0}^{i} c_{i-j} s_j\right)$$

O zaman X^7 teriminin katsayısı;

$$s_0 = 1, s_1 = 0, s_2 = 0, s_3 = 1, s_4 = 0, s_5 = 0, s_6 = 1, s_7 = 0$$
$$c_0 = 1, c_1 = 1, c_2 = 0, c_3 = 1, c_4 = 1, c_5 = 0, c_6 = 0, c_7 = 1$$

olmak üzere;

$$X^7 \sum_{j=0}^{i} c_{i-j} s_j = X^7 (c_7 s_0 + c_6 s_1 + c_5 s_2 + c_4 s_3 + c_3 s_4 + c_2 s_5 + c_1 s_6 + c_0 s_7)$$
$$= X^7 (1 + 0 + 0 + 1 + 0 + 0 + 1 + 0) = X^7$$

Doğrusal tekrarlayan dizinin minimal polinomu o dizinin doğrusal karmaşıklığını ve en düşük periyodunu bulmada çok önemli rol oynar. Gerçekte doğrusal tekrarlayan dizinin en düşük periyodu onun minimal polinomunun periyoduna eşittir. $F_q[x]$'te bir P polinomunun periyodu, $P(0) \neq 0$ olmak üzere, $X^e - 1$'i bölen $P(X)$ polinomu için en küçük e değeridir. Dolayısıyla s serisinin minimal polinomu asal bir polinom ise s serisinin maksimum periyodu $q^{\lambda(s)} - 1$ (ikili seriler için $2^{\lambda(s)} - 1$) dir. Bir önceki örnekte LFSR tarafından üretilen serinin periyodu 3'tür. Çünkü onun minimal polinomu 3 periyoduna sahiptir. Daha önceki örnekte LFSR tarafından üretilen serinin periyodu $2^4 - 1 = 15$'tir. Bunu sebebi bu serinin minimal polinomu $P^*(X) = X^4 + X + 1$ karakteristik polinomuna denk düşer ve $P^*(X)$ polinomu indirgenemez hatta asal bir polinomdur.

L uzunluğunda bir LFSR tarafından üretilen herhangi bir $s = (s_t)_{t \geq 0}$ serisi asal bir geri besleme polinomuna sahipse serinin olası en yüksek doğrusal karmaşıklığı $\lambda(s) = L$ ve $q^L - 1$ olası en büyük periyot değeridir. Bu tür serilere *en uzun (maximum-length) seriler* denir. Bir LFSR'ın geri besleme polinomu daima asal polinom olmalıdır.

En uzun LFSR'lar ile üretilen seriler anahtar dizisi üreteçleri tasarlama da iyi istatistiksel özellikler ortaya koyar. En azından Golomb'un önerilerini yerine getirirler.

Tanım 2.2: N periyotlu periyoduk bir s dizisini düşünelim. Golomb'un rastlantısallık (randomness) ile ilgili öneri aşağıdaki gibidir (Menezes, Oorschot, Vanstone, 1997).

G1: Her periyot boyunca 0'ların sayısı ve 1'lerin sayısı olabildiğince eşit olmalıdır.

G2: Run değerlerinin sayısının yarısı 1 uzunluğunda, $\frac{1}{4}$'ü 2 uzunluğunda, $\frac{1}{8}$'i 3 uzunluğunda vb. olmalıdır. Bunun ötesinde bu uzunlukların her biri için eşit sayıda boşluklar (gap) ve bloklar içermelidir.

G3: Otokorelasyon fonksiyonu $r_f(d)$ K bir tam sayı olmak üzere iki değerlidir. $N.r_f$ değeri 2.8 ifadesinde gösterilmiştir.

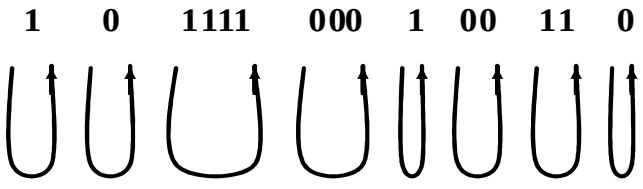
$$N.r_f(d) = \sum_{i=0}^{N-1}(2s_i - 1)(2s_{i+d} - 1) = \begin{cases} N, & d = 0 \\ K, & 1 \le d \le N-1 \end{cases} \qquad (2.8)$$

Örnek 2.8, bir asal geri besleme polinomuna sahip bir LFSR tarafından üretilen serinin Golomb'un önerilerini yerine getirdiğini göstermektedir.

Örnek 2.8:

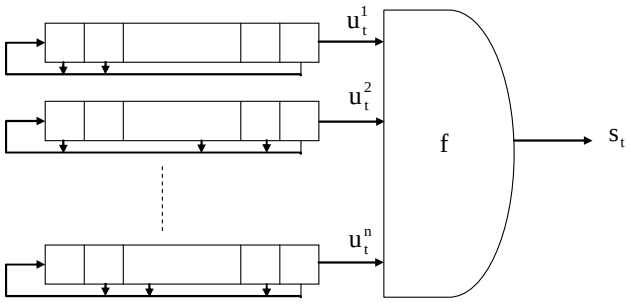
$P(X) = X^4 + X^3 + 1$ geri besleme polinomuna sahip bir LFSR tarafından üretilen bir seri $s^{15} = 1,0,1,1,1,1,0,0,0,1,0,0,1,1,0$ için Golomb'un önerilerinin sağlandığını gösterelim.

- G1: Seride 7 tane 0 ve 8 tane 1 vardır.
- G2: Seride 8 tane run vardır. Bunların 4 tanesi 1 uzunluğunda (2 tanesi boşluk ve 2 tanesi blok), 2 tanesi 2 uzunluğunda (1 tanesi boşluk ve bir tanesi blok), 1 tanesi 3 uzunluğunda (boşluk) ve 1 tanesi de 4 uzunluğundadır (1 blok).



- G3: Otokorelasyon fonksiyonu iki değer alır. Bunlar $r_f(0) = 1$ ve $r_f(d) = \frac{-1}{15}$, $1 \le d \le 14$ tir.

Seri kullanılmadan önce LFSR tarafından ortaya konan doğrusallık özelliğini yok etmemiz ve doğrusal karmaşıklık değerini arttırmamız gerekmektedir. Bunun için klasik yaklaşımlardan biri olan birden fazla LFSR kullanarak Şekil 2.7'de görüldüğü gibi bunları bir Boolean fonksiyonu ile birleştirme yolu örnek olarak verilebilir.



Şekil 2.7. LFSR Serilerinde Doğrusallığı Yok Etmek Amacıyla Kullanılan Örnek bir Doğrusal Olmayan Birleştirici (Ekdahl, 2003)

LFSR tabanlı akış şifreleri *doğrusal olmayan bileşim üreteçleri* (*nonlinear combination generators*), *doğrusal olmayan filtre üreteçleri* (*nonlinear filter generators*) ve *saat kontrollü üreteçler* (*clock-controlled generators*) olarak üç genel kategori altında toplanabilir. Bununla beraber çeşitli tasarım teknikleri bir arada kullanılabileceği için kesin bir sınıflandırma yapmakta mümkün değildir.

2.2. Doğrusal Olmayan Bileşim Üreteçleri (Nonlinear Combination Generators)

Şekil 2.7'de görülen doğrusal olmayan bileşim üreteci birden fazla LFSR ile bu LFSR'ları bir Boolean fonksiyonu ile birleştirme yoluyla elde edilir. f , n değişkenli bir fonksiyondur ve n LFSR'ın F_q üzerine oluşturduğu bir bileşim (combination) üreteç için $f : F_q^n \to F_q$ şeklindedir.

Buna ek olarak f Boolean fonksiyonu uniform çıkış dağılımı elde edilebilmesi için dengeli olmak zorundadır. Çıkış serilerinin iyi istatistiksel özellikler gösterebilmesi için arka arkaya gelen LFSR'ların asal (primitive) geri besleme polinomuna sahip olması gerekmektedir. Genellikle birleştirici fonksiyon ve LFSR'ların özellikleri herkes tarafından bilinir. Gizli paremetreler LFSR'ların başlangıç durumlarıdır ve bir anahtar yükleme algoritması yardımıyla şifrenin gizli anahtarından elde edilir. Dolayısıyla saldırıların çoğu bir üreteç tarafından üretilen serinin bazı bitlerinin bilgisinden yola çıkılarak LFSR'ların başlangıç durumlarını elde etmeyi amaç edinir (Bilinen Açık Metin Saldırısı). Eğer LFSR'ların geri besleme polinomları ve birleştirici fonksiyonu bilinmiyorsa tekrar inşa saldırısı (reconstruction attack) geniş bir şifreli metin segment bilgisinden yola çıkarak üretecin toplam tanımının elde edilmesini sağlar (Canteaut, 2005).

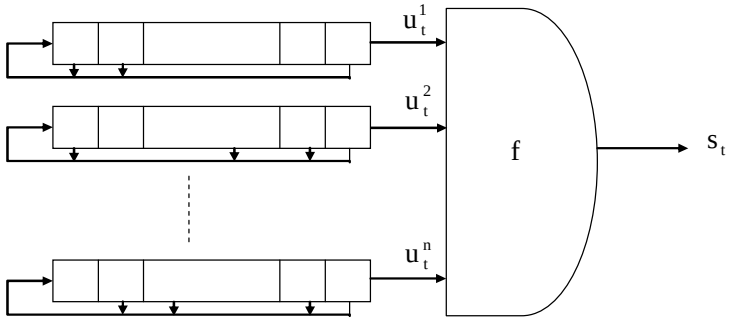
Birleştirici fonksiyon tarafından üretilen seri doğrusal tekrarlayan seridir. Periyodu ve karmaşıklığı seriyi üreten sıralı LFSR'lar ile birleştirici fonksiyonun cebirsel gösterim biçiminden (ANF) yararlanılarak elde edilebilir. F_q^n üzerine iki doğrusal tekrarlayan seri u ve v olsun. Yine bu serilerin karmaşıklıkları $\lambda(u)$ ve $\lambda(v)$ ise o zaman $u+v=(u_t+v_t)_{t\geq 0}$ dizisinin karmaşıklığı $\lambda(u+v)\leq\lambda(u)+\lambda(v)$'dir.

Eğer u ve v'nin minimal polinomları birbirlerine göre asal ise $\lambda(u+v)=\lambda(u)+\lambda(v)$'dir. Aynı şekilde uv serisi $uv=(u_t v_t)_{t\geq 0}$ olmak üzere uv serisinin karmaşıklığı $\lambda(uv)\leq\lambda(u)\lambda(v)$'dir. Eğer u ve v'nin minimal polinomları birbirlerine göre asal ise $\lambda(uv)=\lambda(u)\lambda(v)$'dir (Menezes, Oorschot, Vanstone, 1997; Canteaut, 2005; Günther, 1988). Böylece f Boolean fonksiyonu ile birleştirilmiş asal geri beslemeli n tane ikili LFSR'dan oluşan bir bileşim üreteci tarafından üretilen bir anahtar dizisi (keystream) serisi takip eden özelliği sağlar.

- Eğer tüm LFSR uzunlukları $L_1, L_2, ...L_n$ birbirlerinden farklı ve 2 den büyükse çıkış serisinin doğrusal karmaşıklığı $f(L_1, L_2, ..., L_n)$'e eşittir. Burada cebirsel gösterim biçimi (ANF) kullanılır ve bu gösterim biçimi tamsayılar ile değerlendirilir.

Örnek 2.9:

Uzunlukları L_1, L_2, L_3 ve L_4 olan 4 LFSR için birleştirici fonksiyon $x_1x_2 + x_2x_3x_4 + x_4$ olsun. Sonuç serisinin doğrusal karmaşıklığı $L_1L_2 + L_2L_3L_4 + L_4$ olarak elde edilir.



Şekil 2.8. Doğrusal Olmayan Bileşim Üretecinin Yapısı (Ekdahl, 2003)

2.3. Doğrusal Olmayan Filtre Üreteçleri (Nonlinear Filter Generators)

Şekil 2.8'de görüldüğü gibi doğrusal olmayan filtre üreteçleri tek bir LFSR kullanırlar ve bu LFSR Boolean fonksiyonuna girişler sağlar. Çıkış dizisi $s_t = f(u_{t+\gamma_1}, u_{t+\gamma_2}, ..., u_{t+\gamma_n}), \forall t \geq 0$ olmak üzere n değeri LFSR uzunluğuna eşit ya da küçük ve f ise n değişkenli bir fonksiyondur. Buna ek olarak γ_i, $1 \leq i \leq n$ olmak üzere tapping serisi olarak isimlendirilir ve negatif olmayan tamsayıların azalan bir sırasıdır. İyi istatistiksel özelliklere sahip bir anahtar dizi serisi elde edebilmek için filtre fonksiyonu f dengeli olmalı ve LFSR geri besleme polinomu asal olarak seçilmelidir.

Bir filtre üretecinde, LFSR geri besleme polinomu, tapping sırası herkes tarafından bilinir. Gizli parametre bir anahtar yükleme algoritması yoluyla şifrenin gizli

anahtarı kullanılarak elde edilen LFSR'ın başlangıç durumudur. Buna ek olarak herhangi bir filtre üreteci, özel bir birleştirici üretece denktir.

Doğrusal olmayan filtre yaklaşımı, F_{2^w} genişletilmiş cismini kullanan ve yazılım yoluyla tasarlanan akış şifrelerinde tasarım için etkin bir yoldur. Bunun nedeni olarak F_{2^w} üzerine tanımlanan maksimum uzunluklu LFSR'ların ötelenmesinin yazılımda oldukça maliyetli olması gösterilebilir (Ekdahl, 2003).

Bir filtre üretecin s çıkış sırası doğrusal tekrarlayan bir seridir ve karmaşıklığı $\lambda(s)$ LFSR uzunluğuna ve filtre fonksiyonu f'in cebirsel derecesine bağlıdır. Geri besleme fonksiyonu ile ikili bir LFSR'ın karmaşıklığı, LFSR'ın uzunluğu L ve f'in cebirsel derecesi (Key, 1976; Massey, 2001; Canteaut, 2005) d olmak üzere 2.9 ifadesindeki gibi verilebilir.

$$\lambda(s) \leq \sum_{i=0}^{d} \binom{L}{i} \tag{2.9}$$

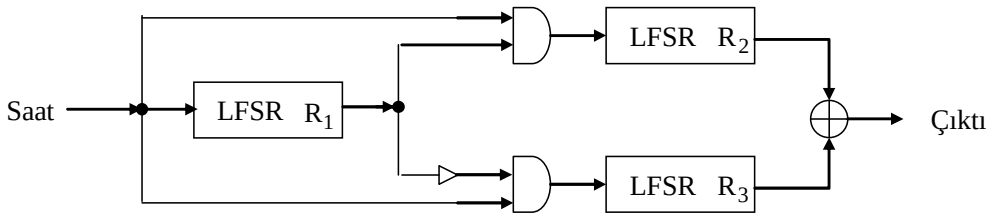
Üretilen seri s'in periyodu $2^L - 1$'i böler ve L büyük bir asalsa $\lambda(s)$ çoğu filtreleme fonksiyonları için en azından $\binom{L}{d}$ değerine eşittir. Yüksek doğrusal karmaşıklığı elde edebilmek için LFSR uzunluğu L ve filtreleme fonksiyonunun derecesi yeterince büyük olmalıdır. Daha kesin bir ifadeyle saldırganın elindeki anahtar dizisi uzunluğu $\binom{L}{\deg(f)}$ değerinden küçük olmalıdır.

2.4. Saat Kontrollü Üreteçler (Clock-Controlled Generators)

Doğrusal olmayan bileşim üreteçleri ve filtre üreteçlerinde kullanılan saat tetiklemesi, o anki durumun 1 adım ileri gitmesini ve anahtar dizisinin bitinin ya da bitlerinin her tetikleme de üretilmesini sağlamaktadır. Diğer bir deyişle saat düzenli bir

şekilde tetiklenir. Saat kontrollü üreteçlerde ise ana fikir düzensiz sinyaller kullanarak LFSR'ların saat tetiklemelerinin sayısını kontrol etmektir (Menezes, Oorschot, Vanstone, 1997; Ekdahl, 2003).

Düzensiz sinyaller, başka bir LFSR'ın çıkışı ya da şifrenin içsel bir değişkeni olabilir. Düzensiz olarak kontrol edilen LFSR'ın çıkışındaki doğrusallık yok edilerek düzenli olarak tetiklenen üreteçlere bu özelliğinden dolayı yapılan saldırıların önüne geçilebilir. Bu tür üreteçlere örnek olarak Günther (Rueppel, 1986) tarafından öne sürülen *alternatifli adım üreteci* (*alternating step generator*) verilebilir ve Şekil 2.9'da gösterilmektedir. Diğer saat kontrollü üreteçlere örnek olarak *büzülen üreteç* (*shrinking generator*) verilebilir.



Şekil 2.9. Alternatifli Adım Üreteci (Ekdahl, 2003)

Alternatifli adım üretecinin çalışma prensibi aşağıdaki gibi açıklanabilir:

- R_1 saklayıcısı sistem saati tarafından tetiklenir.
- R_1 saklayıcısının çıkışı 1 ise, R_2 tetiklenir ve R_3 tetiklenmeden bir önceki bit değeri tekrar edilir. (İlk çevrimde (cycle) R_3 saklayıcısının ilk çıkış biti 0 alınır.)
- R_1 saklayıcısının çıkışı 0 ise, R_3 tetiklenir ve R_2 tetiklenmeden bir önceki bit değeri tekrar edilir. (İlk çevrimde (cycle) R_2 saklayıcısının ilk çıkış biti 0 alınır.)
- R_2 ve R_3 saklayıcılarının toplamı anahtar dizisi çıkışı olarak elde edilir.

R_1 saklayıcısının 2^{L_1} periyotlu bir Bruijn serisi ürettiğini düşünürsek ve R_2 ile R_3 saklayıcıları maksimum uzunluklu LFSR olmak üzere, $OBEB(L_2, L_3) = 1$, aralarında asal ise

- serinin periyodu $2^{L_1}(2^{L_2}-1)(2^{L_3}-1)$ 'dir

- *s* serisinin doğrusal karmaşıklığı $\lambda(s)$ aşağıdaki ifadeyi sağlar.

$$(L_2 + L_3)2^{L_1-1} < \lambda(s) < (L_2 + L_3)2^{L_1}$$

- serideki örüntülerin (patern) dağılımı hemen hemen düzgündür.

Alternatifli adım üretecinde kullanılan LFSR'lar R_1, R_2, R_3 maksimum uzunluklu LFSR'lar olmalı ve LFSR'ların uzunlukları L_1, L_2, L_3 birbirlerine göre asal olmalıdır $(OBEB(L_1, L_2) = 1, OBEB(L_2, L_3) = 1, OBEB(L_1, L_3) = 1)$. Bunun ötesinde uzunlukları birbirlerine yakın olmalıdır. Yani, eğer $L_1 = l$ ise $L_2 \approx l$ ve $L_3 \approx l$ olmalıdır. Alternatifli adım üretecine en iyi saldırı böl ve fethet (divide and conquer) saldırısıdır. Bu saldırı R_1 saklayıcısına 2^l adım ile mümkündür. Eğer $l = 128$ ise üreteç bütün bilinen saldırılara karşı güvenlidir (Menezes, Oorschot, Vanstone, 1997).

2.5. Tezde İncelenen Akış Şifre Algoritmaları

2.5.1. Grain-128 Akış Şifresi

Grain akış şifresi Martin Hell, Thomas Johansson, Willi Meier ve Alexander Maximov tarafından 2004 yılında sunulan bir akış şifreleme algoritmasıdır. GRAIN, öncelikle sınırlı donanım ortamları için tasarlanmıştır. İlk tasarımı olan Grain 1.0, 80-bit anahtar ve 64-bit IV kullanırken Grain-128 olarak adlandırılan geliştirilmiş versiyonu 128-bit anahtar ve 96-bit IV kullanmaktadır.

Tezde Grain-128 versiyonu incelenerek istatistiksel testlere tabi tutulmuştur.

Şifreleme LFSR (Linear Feedback Shift Register- Doğrusal Geri Beslemeli Saklayıcı), NFSR (Nonlinear Feedback Shift Register- Doğrusal Olmayan Geri

Beslemeli Saklayıcı) ve bir çıkış fonksiyonu olmak üzere üç ana yapı taşından oluşur. LFSR'ın içeriği $s_i, s_{i+1},..., s_{i+127}$ biçiminde, NFSR'ın ise benzer şekilde $b_i, b_{i+1},..., b_{i+127}$ biçiminde ifade edilir. LFSR'ın geri besleme polinomu $f(x)$ olarak ifade edilir ve şu şekildedir:

$$f(x) = 1 + x^{32} + x^{47} + x^{58} + x^{90} + x^{121} + x^{128}$$

Herhangi bir belirsizlik olasılığını ortadan kaldırmak için LFSR'ın güncelleme fonksiyonu şu şekilde ifade edilmiştir:

$$s_{i+128} = s_i + s_{i+7} + s_{i+38} + s_{i+70} + s_{i+81} + s_{i+96}$$

NFSR'ın geri besleme polinomu bir lineer ve bir bükük fonksiyonun toplamından oluşur. $g(x)$ olarak ifade edilir ve şu şekildedir:

$$g(x) = 1 + x^{32} + x^{37} + x^{72} + x^{102} + x^{128} + x^{44}x^{60} + x^{61}x^{125} + x^{63}x^{67} + x^{69}x^{101} + x^{80}x^{88} + x^{110}x^{111} + x^{115}x^{117}$$

Benzer şekilde herhangi bir belirsizlik olasılığını ortadan kaldırmak için NFSR'ın güncelleme fonksiyonu ise şu şekilde ifade edilmiştir:

$$b_{i+128} = s_i + b_i + b_{i+26} + b_{i+56} + b_{i+91} + b_{i+96} + b_{i+3}b_{i+67} + b_{i+11}b_{i+13} + b_{i+17}b_{i+18} + b_{i+27}b_{i+59} + b_{i+40}b_{i+48} + b_{i+61}b_{i+65} + b_{i+68}b_{i+84}$$

İki kaydırma saklayıcısındaki 256 bellek elemanı şifrenin durumunu gösterir. Bu durum bir Boolean fonksiyonu olan $h(x)$'e girdi olarak 9 değişken alır. $h(x)$'in iki girdisi NFSR'dan yedi girdisi ise LFSR'dan alınır. Bu fonksiyonun derecesi 3'tür ve şu şekilde ifade edilir:

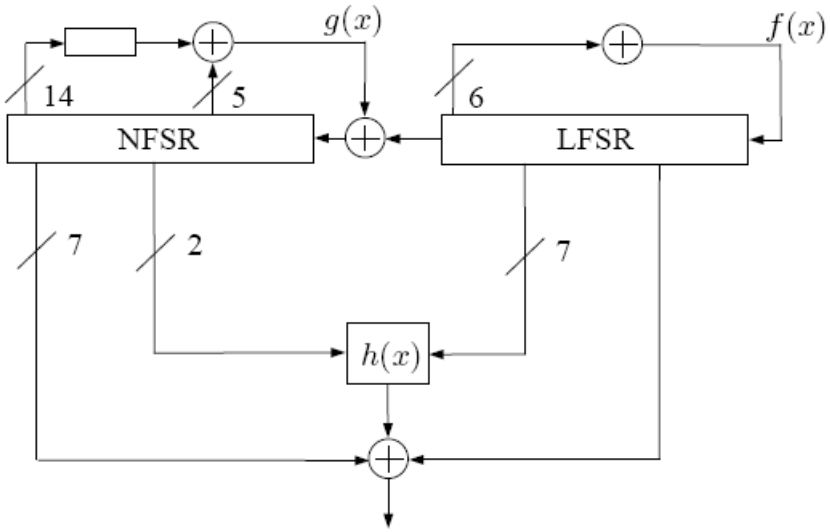
$$h(x) = x_0x_1 + x_2x_3 + x_4x_5 + x_6x_7 + x_0x_4x_8$$

Bu dokuz değişken sırasıyla $b_{i+12}, s_{i+8}, s_{i+13}, s_{i+20}, b_{i+95}, s_{i+42}, s_{i+60}, s_{i+79}, s_{i+95}$ bağlantı pozisyonlarından alınır.

Çıktı fonksiyonu şu şekilde ifade edilir:

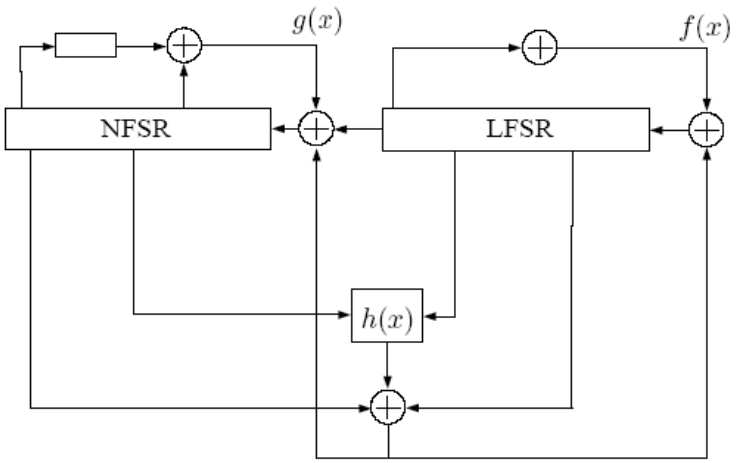
$$z_i = \sum_{j \in A} b_{i+j} + h(x) + s_{i+93} \qquad A = \{2,15,36,45,64,73,89\}$$

Şifrenin genel yapısı Şekil 2.10'da gösterilmektedir.



Şekil 2.10. Grain-128 Akış Şifresi Tasarım Şeması (Hell, Johansson, vd., 2006)

Anahtar akışı üretilmeden önce şifre anahtar ve IV ile başlatılmak zorundadır. Anahtar bitleri k_i ile ($0 \le i \le 127$), IV bitleri ise IV_i ($0 \le i \le 95$) ile ifade edilir. Daha sonra anahtarın ve IV'nin başlatılması şu şekilde gerçekleştirilir: NFSR'ın 128 elemanı anahtar bitleriyle birlikte yüklenir $b_i = k_i, 0 \le i \le 127$. Daha sonra; LFSR'ın ilk 96 elemanı IV bitleriyle birlikte yüklenir $s_i = IV_i, 0 \le i \le 95$. LFSR'ın son 32 biti birlerle doldurulur $s_i = 1, 96 \le i \le 127$. Anahtar ve IV bitlerinin yüklenmesinden sonra şifre 256 kez herhangi bir anahtar akışı üretmeden çalıştırılır. Çıkış fonksiyonu yerine geri besleme hem LFSR'ın hem de NFSR'ın girişleriyle XOR'lanır.



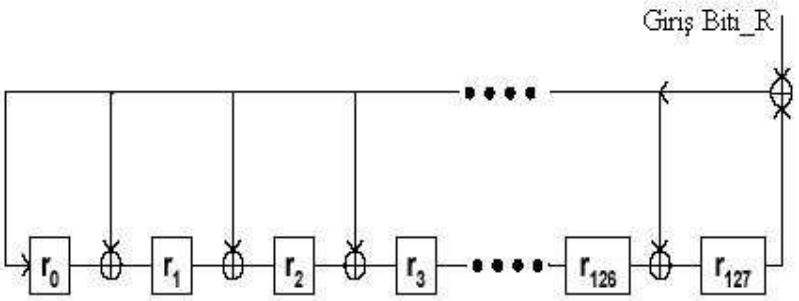
Şekil 2.11. Grain-128 Anahtar Başlatma (Hell, Johansson, vd., 2006)

Grain-128 şifresinde tasarım parametreleri doğrusal yaklaşımlar ve diğer olası saldırılar için teorik kanıtlara göre seçilmiştir. Şifre az kapı sayısı, düşük güç tüketimi ve küçük çip kullanımı gerektiren donanım ortamları için uygundur. Aynı donanım ve güvenlik kriterleri ile 128-bitlik güvenlik sunan nadir akış şifrelerden biridir (Hell, Johansson, vd., 2006).

2.5.2. Mickey-128 Akış Şifresi

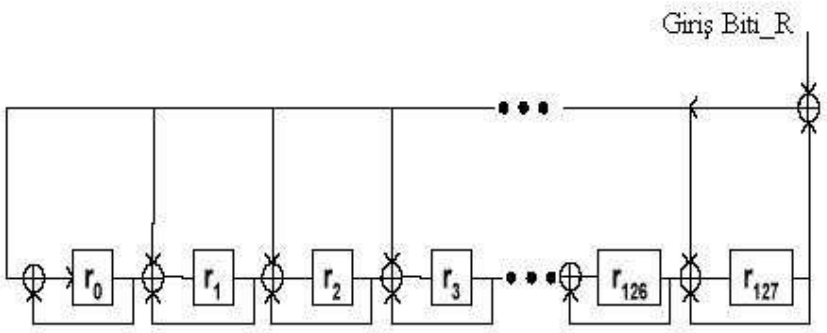
Mickey-128, Steve Babbage ve Matthew Dodd tarafından 2005 yılında sunulan bir akış şifreleme algoritmasıdır. Mutual Irregular Clocking KEYstream- Karşılıklı Düzensiz Saat Kontrollü Anahtar Akışı kelimelerinden türetilmiştir. Mickey-128 akış şifresi, yüksek seviyede güvenlik sağlarken, düşük donanım karmaşıklığı için tasarlanmıştır. Kaydırma saklayıcıları bazı kriptanalitik saldırıları önlemek, bazı yeni tekniklerin periyod garantisini sağlamak vb. durumlar için düzensiz saat kontrolü kullanır.

Mickey-128 akış şifresi, 128-bitlik gizli anahtar *K* ve 0 ile 128-bit arası değer alabilen başlangıç vektörü *IV*'den oluşan iki giriş parametresi alır. 128-bitlik R ve S saklayıcıları tasarımı oluşturmak için kullanılmıştır. R saklayıcısı LFSR, S saklayıcısı ise NFSR formundadır. *Kontrol_biti_R* ve *Kontrol_biti_S* şeklinde iki değer tanımlanır. *Kontrol_biti_R* $s_{43} \oplus r_{85}$, *Kontrol_biti_S* ise $s_{85} \oplus r_{42}$ şeklinde ifade edilir. *Kontrol_biti_R=0* olduğunda *R* saklayıcısı Şekil 2.12'de gösterildiği gibi standart bir LFSR gibi davranır.



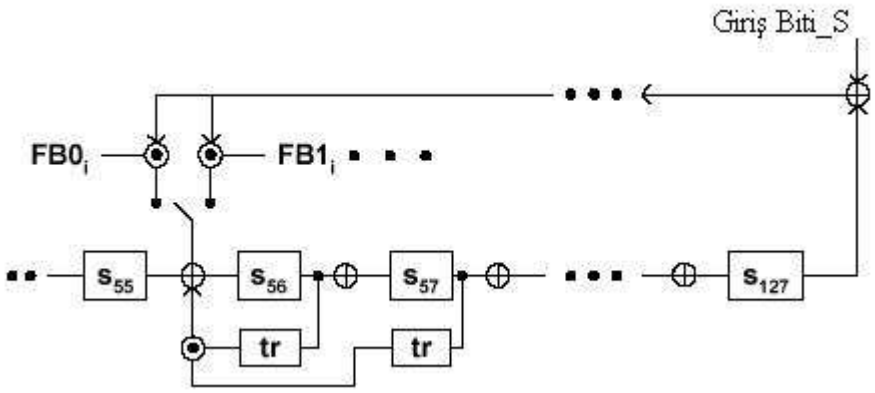
Şekil 2.12. *Kontrol_biti_R=0* için R Saklayıcısı (Kitsos, 2006)

Kontrol_biti_R=1 olduğunda Şekil 2.13'de gösterildiği gibi saklayıcıdaki her bit sağa kaydırılarak XOR'lanır.



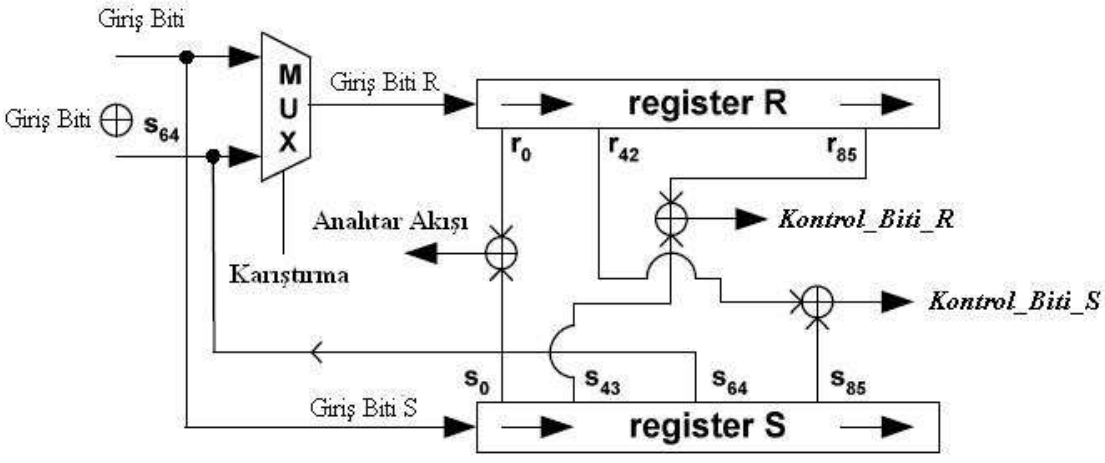
Şekil 2.13. *Kontrol_biti_R=1* için R Saklayıcısı (Kitsos, 2006)

Şekil 2.14, NFSR tipinde *S* saklayıcısının tasarımını göstermektedir. $FB0_i$, $FB1_i$ ve *tar* dönüşümleri ve matematiksel eşitlikleri (Babbage, Dodd, 2005)'den elde edilebilir.



Şekil 2.14. S Saklayıcısı (Kitsos, 2006)

Mickey-128 akış şifresi açıklanan bu yapılar doğrultusunda Şekil 2.15'de gösterildiği gibi tasarlanmaktadır.



Şekil 2.15. Mickey-128 Akış Şifresi Mimarisi (Kitsos, 2006)

Mickey-128 akış şifresinin iki önemli avantajı düşük donanım karmaşıklığı ve yüksek seviyede güvenlik sağlamasıdır. Şifrenin tasarımı taşınabilir teknolojiler, akıllı kartlar gibi uygulamalar için uygundur (Kitsos, 2006).

2.5.3. Sosemanuk Akış Şifresi

Sosemanuk, Come Berbain, Olivier Billet, Anne Canteaut, Nicolas Courtois, Henri Gilbert, Louis Goubin, Aline Gouget, Louis Granboulan, Cédric Lauradoux, Marine Minier, Thomas Pornin ve Hervé Sibert tarafından geliştirlmiş yazılım tabanlı bir akış şifresidir. 256-128 bit anahtar ve 128-bit başlangıç vektörü (IV) kullanır. Bu şifreleme SNOW 2.0 akış şifresine göre tasarlanmıştır. 320-bit LFSR ve iki adet 32-bitlik saklayıcı içeren FSM (Finite State Machine-Sonlu Durum Makinesi)'den oluşur. Sosemanuk akış şifresi FSM'nin bazı yapısal özelliklerinde görülen potansiyel zayıflıklara ve dahili durum boyutunu azaltmaya yönelik etkili yöntemler içermektedir.

Sosemanuk, 12-word dahili durum içeren word tabanlı bir akış şifredir. Her word 32-bitten oluşur. LFSR GF(2^{32}) için 10 eleman içerir. Geri besleme polinomu şu şekildedir:

$$\pi(X) = \alpha X^{10} + \alpha^{-1} X^{7} + X + 1$$

α, GF(2^{32})'de asal bir elemandır. LFSR durumları arasındaki yinelemeli ilişki şu şekilde ifade edilir:

$$S_{t+10} = S_{t+9} + \alpha^{-1} S_{t+3} + \alpha S_t$$

FSM giriş olarak LFSR'dan üç tane 32-bit word, çıkış olarak bir tane 32-bit word alır ve R1, R2 olarak adlandırılan iki tane 32-bit word'lük hafızaya sahiptir. FSM aşağıdaki gibi gösterilen üç fonksiyona sahiptir:

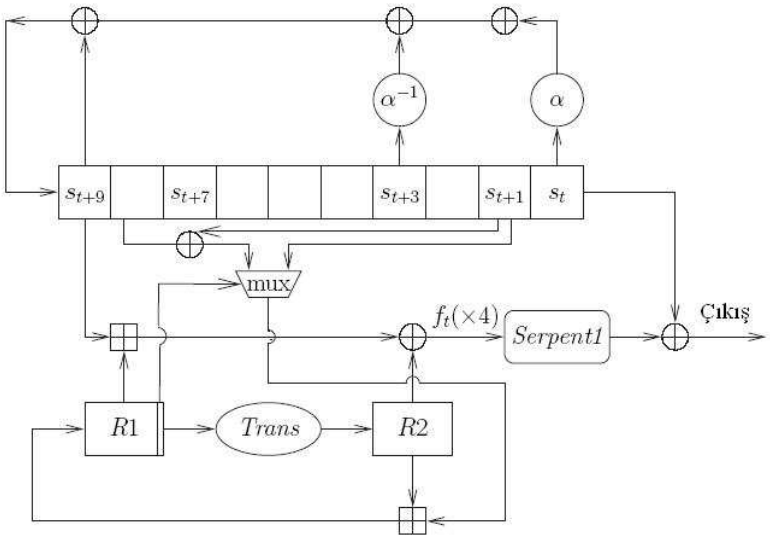
$$R1_t = \left(R2_{t-1} + mux(lsb(R1_{t-1}), S_{t+1}, S_{t+1} \oplus S_{t+8})\right) \bmod 2^{32}$$
$$R2_t = Trans(R1_{t-1})$$
$$f_t = (S_{t+9} + R1_t \bmod 2^{32}) \oplus R2_t$$

lsb(x), x'in en anlamsız bitidir. *mux(c,x,y)*; c=0 ise x'e, c=1 ise y'ye eşittir. Geçiş fonksiyonu olan *Trans(z)* GF(2^{32})'de çalışır. FSM'nin çıktıları dört grupta depolanır ve *Serpent1* her dört word grubuna uygulanır. *Serpent1*'in çıktısı LFSR'ın karşılık gelen dört çıktısıyla XOR'lanır. Böylelikle anahtar akışının dört word grubu oluşturulmuş olur.

$$(Z_{t+3}, Z_{t+2}, Z_{t+1}, Z_t) = Serpent1(f_{t+3}, f_{t+2}, f_{t+1}, f_t) \oplus (S_{t+3}, S_{t+2}, S_{t+1}, S_t)$$

Serpent1 fonksiyonu SERPENT blok şifresinin bir S-kutusunu, S_2, 128-bitlik bir girişin yer değiştirilerek paralel olarak uygulanmasıyla elde edilen 32 kopyasını kullanır. *Serpent1*'in uygulanması bit-bölme modunda çıkışın her bir bitinin girişteki dört word'ün hepsine ve tam tersine ilgili fonksiyonun uygulanmasıyla gerçekleştirilir.

Sosemanuk akış şifresinin temel yapısı Şekil 2.16'da gösterilmektedir (Ahmadi, Eghlidos, Khazaei, 2005)



Şekil 2.16. Sosemanuk Akış Şifresinin Genel Yapısı (Ahmadi, Eghlidos, Khazaei, 2005)

3. AKIŞ ŞİFRE TASARIMLARINDA KULLANILAN RASSAL SAYI ÜRETEÇLERİ

Rassallığın tanımı kısaca tahmin edilemeyen, belirli bir kalıba sahip olmayan olarak verilebilir. Rassal olarak üretilen sayılar, şans oyunlarında, istatistiksel örneklemelerde ve simulasyon uygulamalarında sıkça kullanılır. Rassallık kriptografide kullanılan en temel özelliklerden biridir. Atak yapan kişiye, bir kriptosistem çıktısının olabildiğince tahmin edilemez olması gerekir. Rassal sayılar birçok kriptografik uygulamanın temelini oluşturur. Oluşturulması en gerekli ve aynı zamanda en zor olan kısımdır. Neredeyse bütün kriptografik protokollerde gizli ve tahmin edilmesi zor değerlere ihtiyaç duyulur, örneğin asimetrik şifreleme yöntemlerinde (RSA, Diffie Hellman) anahtar oluşturulurken, rassal sayılar kullanılır. Anahtar gizliliği kriptosistemlerde çok önemli olduğu için, programlama dillerinde standart olarak kullanılan rassal sayı üreteçlerinin kriptografik amaçlar için kullanılması sakıncalıdır. Genelde, bu algoritmalar istatistiksel rassallık için tasarlanmıştır, kriptanalize karşı dayanıklı değillerdir. Temel olarak rassal sayılar iki farklı yöntemle oluşturulur. Bunlar Gerçek Rassal Sayı Üreteçleri (GRSÜ) ve Sözde-Rassal (Pseudo-Random) Sayı Üreteçleri (SRSÜ)'dir.

Gerçek Rassal Sayı Üreteçleri: İçinde rassal bir yapı bulunduran fiziksel sinyal kaynakları kullanarak dizi üreten üreteçlerdir. Bu üreteçlerin en önemli avantajları:

- Dizinin bir kısmına sahipken, farklı bir kısmını elde etmenin mümkün olmaması;
- Üretilen diziler kendi içinde herhangi bir gizli bağıntının bulunmaması;
- Periyodik olmamalarıdır.

Bu avantajların yanı sıra, gerçek rassal sayı üreteçlerinin önemli dezavantajları da bulunur. Bu üreteçler çoğunlukla verimsizdir, uzun sayı dizileri elde etmenin maliyeti yüksektir. Deneyi tekrarlayıp bir sayı dizisini yeniden elde etmek mümkün değildir.

Sözde-Rassal (Pseudo-Random) Sayı Üreteçleri: Matematiksel algoritmalar kullanarak diziler üreten üreteçlerdir. Bu algoritmalar kendi içlerinde herhangi bir rassallık barındırmazlar, algoritmalar da genelde açıktır. Buradaki rassallık algoritmaların girdileri (seed) ile sağlanır. Bu yüzden algoritmaların girdileri gizli tutulmalıdır ve kolay tahmin edilemez olmalıdır. Algoritma ve girdi bilinirse, dizinin tümü elde edilebilir. Bu üreteçler

verimlidir ve uzun diziler üretmenin maliyeti düşüktür. Kriptografik olarak kullanılabilecek sözde-rassal sayı üreteçleri ile üretilen bir dizinin bir kısmı biliniyorsa, bu dizinin diğer kısımları ile ilgili bir bilgi vermemelidir. Aynı üreteçle üretilen farklı diziler birbirleri ile ilişkileri olmamalıdır (correlation). Dizilerin periyotları mümkün olduğunca uzun olmalıdır (Kriptolojiye Giriş Ders Notları, 2004).

Uygulamada, pek çok sözde-rassal sayı üreteci istatistiksel olarak önemli testleri geçmelerini engelleyen bazı durumlar sergiler. Bunlardan sadece birkaçını söylemek gerekirse:

- Bazı başlangıç durumları için beklenenden daha kısa periyodlar
- Kötü boyutsal dağılım
- Birbirini takip eden değerlerin bağımsız olmaması
- Bazı bitlerin diğerlerinden 'daha rassal' olabilmesi
- Tekbiçimlilik eksikliği Hatalı sözderassal sayı üreteçlerinin problemleri kolay kolay tespit edilemeyecek türde olabileceği gibi saçma denecek kadar açık da olabilir.

Şifre bilimsel olarak uygun olan bir sözderassal sayı üreteci rassallık testlerini geçmeye ek olarak bazı ek şifre bilimsel koşulları da sağlamak zorundadır. Bazı şifre bilimsel olarak güvenli sözde-rasssal sayı üretici algoritmalar şunlardır:

- Counter modda veya çıktı besleme modunda çalışan akış veya blok şifreleri.
- Güvenlik kanıtı olan özel tasarımlar. Örneğin; Blum Blum Shub algoritmasının güçlü bir koşullu güvenlik kanıtı vardır ancak yavaş çalışmaktadır.
- Şifre bilimsel olarak güvenliğe dikkat ederek tasarlanmış özel sözderassal sayı üreteçleri. Örn. ISAAC algoritması. Bu algoritma epey hızlıdır ve periyodu büyüktür.

Akış şifrelerde kullanılmış rassal sayı üreteci algoritmalarını şu şekilde sıralamak mümkündür.

- Doğrusal Geri Beslemeli Öteleyici Saklayıcılar (Linear Feedback Shift Register-LFSR)
- Küçülen Üreteç (Shrinking Generator)
- Kendinden Küçülen Üreteç (Self-Shrinking Generator)
- Blum Blum Shub
- Ters Benzer Üreteç (Inversive Congruential Generator)

- ISAAC (Indirection, Shift, Accumulate, Add, and Count Operation- Dolaylı, Kaydırma, Toplama, Sayma İşlemi)
- Gecikmeli Fibonacci Üreteci (Lagged Fibonacci Generator)
- Ters Doğrusal Üreteç (Linear Congruential Generator)
- Elde ile Çarpma (Multiply With Carry)
- Mersenne Twister
- Maksimum Periyodik Karşıtlık (Maximal Periodic Reciprocals)
- XOR Kaydırma (XOR-Shift)

4. İSTATİSTİKSEL TESTLER

İstatistiksel çıkarım yapmak için istatistiksel hipotez testleri kullanılır. Bu testlerde bir hipotez (null hypothesis, H_0) öne sürülür, bu hipotezin tersi de alternatif hipotez, H_a olarak kabul edilir. İstatistiksel test sonucunda varılabilecek iki farklı temel karar vardır:

- H_0'ı reddet ya da H_0'ı reddetme.

Birinci karar, H_0 aleyhine güçlü bir kanıt elde edildiğinde verilir. Bu güçlü kanıt bulunamadığında ise ikinci karar verilir.

Bütün istatistiksel testlerde kaçınılmaz hata yapma payı vardır. Test sonucunda iki farklı hata, birinci tip (alfa) hata ve ikinci tip (beta) hata yapılabilir. Birinci tip hata hipotezimiz doğruyken, kararımız H_0'ı reddet olduğunda gerçekleşir. İkinci tip hata ise hipotezimiz yanlışken, kararımız H_0'ı reddetme olduğunda gerçekleşir. Hipotez testinde birinci tip hata yapma olasılığını sınırlamak gerekir. Test sonucunda birinci tip hata yapma olasılığımız, testimizin güvenilirlik seviyesini verir. Bu değer genel olarak 0.01-0.05 olarak seçilir. İstatistiksel bir testin gücü, ikinci tip hatayı yapmama olasılığına eşittir. Testin gücünü arttırmak için daha fazla örnekleme yapılır. İstatistiksel bir test yapılacağı zaman ilk olarak, H_0 ve H_a belirlenir. Daha sonra testin güvenilirlik seviyesine karar verilir. Bir örnekleme yapılır ve test istatistiği ve buna bağlı olarak p-değeri hesaplanır. P-değeri, birinci tip hata yapma olasılığını kontrol etmek yerine, H_0'ın doğru olduğu varsayımı ile test istatistiğinin gözlemleme değeri veya daha uç bir değer olması olasılığına karşılık gelir. Bu tanıma uygun olarak hesaplanan olasılık p-değerini verir. Eğer bu değer seçilen güvenilirlik değerinden küçükse H_0 hipotezi reddedilir. İstatistiksel testlerde en çok kullanılan dağılımlar Normal ve Ki-kare dağılımlarıdır.

Normal Dağılım: Gauss Dağılımı adı ile de bilinen normal dağılım ilk kez De Moivre tarafından bulunmuştur. Genelde, hipotez testleri dağılımın normal olduğu varsayımına göre düzenlenir. Dağılımın ortalama ve standart sapma olmak üzere iki parametresi vardır. Çan eğrisi olarak da bilinir. Eğrinin tepe noktası ortalamasına denk gelir. Ayrıca

bu dağılımda ortalama, medyan ve mod aynı değerdir. Ortalamaya göre simetrik bir grafiği vardır. Dağılımın standart sapması eğrinin genişliğini belirler. Ortalaması sıfır ve standart sapması 1 olan normal dağılıma sahip bir değişkenin dağılımına standart normal dağılım denir. Standart normal dağılıma sahip değişkenler Z ile gösterilir. Ortalamadan iki yöne 1,2 ve 3 standart sapma kadar uzaklaşıldığında, toplam alanın sırasıyla %68:26 , %95:44 ve %99:74'ü kapsanır.

Ki-Kare Dağılımı: Standart normal bir dağılımdan seçilen bir birimin x değerinin karesi bir ki-kare değeri olur. Bu şekilde tek bir birimden elde edilen ki-karelerin dağılımı bir serbestlik derecelidir. Standart normal bir dağılımdan seçilen n değerin karelerinin toplamı n serbestlik dereceli bir ki-kare dağılımı olur. Dağılımın şekli serbestlik derecesine göre değişir ve asimetriktir. Sürekli bir dağılıma sahiptir (Kriptolojiye Giriş Ders Notları,2004; National Institute of Standard and Technology, 2001).

Sayı üreteçlerinin çıkışının rassal olduğu, matematiksel olarak kanıtlanamamasına rağmen, geçerli istatistiksel testleri uygulayarak, sayı dizilerinin rassal olmadığını veya bunun tersini söyleyebiliriz. Bu testler, üretecin çıkışının gerçek bir rassal diziden beklenenleri karşılayıp karşılamadığını söyler. Ayrıca testlerin sonuçlarına bakılarak rassal sayı üretecinin kalitesi hakkında yorum yapılabilir. Bir sayı dizisinin rassal olduğunu söylemek için, tüm testlerden geçmesi gerekir. Sadece bir tane test başarısız olsa bile dizi rassal kabul edilemez.

Bilinen testlerden başlıca iki tanesi, Ulusal Standartlar ve Teknoloji Enstitüsü tarafından yayınlanan (NIST) FIPS 140-1 ve NIST 800-22’dir. FIPS 140-1 testi blok uzunluğu küçük olan verilerin testinde kullanılır. NIST 800-22 testi ise görece uzun bloklardan oluşan verileri test etmek için kullanılır ve FIPS 140-1 testine göre kriterleri daha zorlayıcıdır. Bu nedenle kullanım amacına uygun olarak bu testlerden biri tercih edilebilir (Demirkol, 2007).

4.1. FIPS 140-1

FIPS 140-1 testi dört tane ayrı testten oluşur. RSÜ'nün çıkışından alınan ve 20000 tane bit içeren bir bit dizisi dört teste birden tabi tutulur ve dizinin rassal olabilmesi için tüm testlerden geçmesi gerekir. Bu dört test aşağıda açıklanmıştır (Demirkol, 2007; National Institute of Standard and Technology, 2001).

4.1.1. Monobit Testi

Bu testin amacı, bit dizisindeki 0 ve 1 sayısının rassal bir diziden beklendiği gibi olup olmadığını tespit etmektir. Testin başarılı olabilmesi için 20000 bitlik bir dizideki '1' sayısının 9654 < n < 10346 aralığında olması gerekir.

4.1.2. Poker Testi

Bu testte, 'k' bit içeren dizi, m $k \geq 5.2^m$ olacak şekilde, üst üste çakışmayan m-bitlik parçalara ayrılır ve i. parça n_i diye adlandırılır. Rassal bir diziden beklenen, tüm mbitlik blokların k uzunluklu bir dizide aynı sayıda birbirini tekrar etmesidir. Testin başarılı olabilmesi için, $X = \frac{2^m}{k}\left(\sum_{i=0}^{2^m} n_i^2\right) - k$ formülüyle hesaplanan X değerinin, k=20000 ve m=4 için, 1.03 < X < 57.4 aralığında olması gerekir.

4.1.3. Blok (Runs) Testi

Elimizdeki bit dizisinin bu testten başarıyla geçmesi için, dizide ardarda gelen '1' ve '0'lardan oluşan çeşitli uzunluktaki blokların (runs'ların) sayısının tabloda

belirtildiği gibi olması beklenir. 6 bitten daha uzun bloklar 6 bitlik olarak kabul edilmektedir (Demirkol, 2007; National Institute of Standard and Technology, 2001).

Runs Testi Koşulları

Blok uzunluğu	Blok sayısı aralığı
1	2267-2733
2	1079-1421
3	502-748
4	223-402
5	90-223
6+	90-223

4.2. NIST Testleri

NIST testi, FIPS 140-1 testine göre çok daha güçlü bir testtir. FIPS 140-1 testini geçen bir bit dizisi NIST 800-22 testinden kalabilir. Bu nedenle ciddi uygulamalarda NIST 800-22 tercih edilmektedir. NIST 800-22 kendi içinde 15 tane ayrı testten oluşur. Teste tabi tutulan bit dizisinin başarılı olabilmesi için tüm testleri başarıyla geçmesi gerekmektedir. Aşağıda bu testlerin hepsi kısa açıklamasıyla birlikte verilmiştir:

1. *Frekans (Frequency) Testi:* Bit dizisindeki 1 ve 0 dengesini inceler.
2. *Blok Frekans (Block Frequency) Testi:* m bitlik bit bloklarının 0 ve 1 dengesini inceler.
3. *Akış (Runs) Testi:* Dizideki 0 ve 1 bloklarının (runs) sayısını inceler.
4. *Bloktaki En Uzun Birler (Longest run of Ones in a Block) Testi:* Dizideki 0 ve 1 bloklarının (runs) uzunluklarını inceler.
5. *Rank Testi:* Sabit uzunluklu bit blokları kullanılarak, her biri bir satırı belirtecek şekilde, bir matris oluşturulur ve matrisin rankı hesaplanarak bloklar arasındaki lineer bağımlılık incelenir.

6. *Ayrık Fourier Dönüşümü (Discrete Fourier Transform) Testi:* Mevcut bit dizisinin ayrık Fourier dönüşümünü alır ve periyodikliği inceler.
7. *Çakışmayan Şablon Eşleşme (Non-Overlapping Template Matching) Testi:* m bitlik bir bloğun dizi içinde tekrarını inceler. Tekrar edilmesi halinde, tekrar edilen bloktan itibaren yeni bir m bitlik blok oluşturulur.
8. *Çakışan Şablon Eşleşme (Overlapping Template Matching) Testi:* m bitlik bir bloğun dizi içinde tekrarını inceler. Tekrar edilmesi halinde, blok 1 bit ötelenerek yenisi oluşturulur.
9. *Evrensel (Universal) Testi:* Dizinin veri kaybı olmadan ne kadar sıkıştırılabileceğini inceler.
10. *Doğrusal Karmaşıklık (Linear Complexity) Testi:* Bit dizisinin LFRS (linear feedback shift register) uzunluğuna bakarak kompleksliğini inceler.
11. *Seri (Serial) Testi:* Tekrar eden m bitlik 2m tane bloğun tekrar sayısının dağılımını inceler. m=1 için, birinci teste denktir.
12. *Yaklaşık Entropi (Approximate Entropy) Testi:* Tekrar eden m ve (m+1) bitlik blokların entropisini inceler.
13. *Birikimli Toplamlar (Cumulative Sums) Testi:* Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirler ve bloklar arasındaki dengesizlik farkına bakar.
14. *Rassal Farklılık (Random Excursion) Testi:* Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirler ve daha sonra blokların dengesinin dağılımını inceler.
15. *Rassal Farklılık (Random Excursion) Varyans Testi:* Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirleyip ortalama değerden sapma miktarını belirler (Demirkol, 2007; National Institute of Standard and Technology, 2001).

4.2.1. Frekans (Frequency) Testi

Verilen bir dizide bulunan 0 ve 1'lerin oranını kontrol eder. Testin herhangi bir parametresi yoktur. Testte kullanılan referans dağılım yarım normal dağılımdır. Testin sonunda elde edilen p-değerinin çok küçük çıkması dizideki 1'lerin ya da 0'ların sayısının beklenenden fazla olduğunu gösterir. Testin geçerli olabilmesi için dizi uzunluğunun en az 100 olması gerekir. Test denklemleri kullanılarak üretilen değer olan $p > 0.01$ ise dizi rassal olarak kabul edilir (National Institute of Standard and Technology, 2001; Büyüksaraçoğlu, Buluş, 2009).

n: Bit dizisinin boyutu

ε: RNG veya PRNG ile üretilen bit dizisi

S_n: Bit dizisinin toplam değeri (Bit dizisindeki 0'lar (-1), 1'ler ise kendi değeri kabul edilerek toplama işlemi gerçekleştirilerek elde edilen sonuçtur).

$$S_{obs} = \frac{|S_n|}{\sqrt{n}}$$

*erfc:*Hata fonksiyonu $erfc(z) = \frac{2}{\sqrt{\pi}} \int_z^{\infty} e^{-u^2} du$

P-değeri= $erfc\left(\frac{S_{obs}}{\sqrt{2}}\right)$

ε=1011011010110110001010101010101110011100011011000101000110010011010101001101010010101100110011100110

n = 100 $S_{100} = 2$ $S_{obs} = 0.2$

P-değeri = 0.843053 > 0.01 olduğundan dizi rassal kabul edilir (Büyüksaraçoğlu, Buluş, 2009).

4.2.2. Blok Frekans (Block Frequency) Testi

Verilen bir dizide bulunan 0 ve 1'lerin oranını M bitlik bloklar içinde kontrol eder. Testin tek parametresi blok uzunluğudur (M). Blok uzunluğu 1 olarak alındığında blok frekans testi, frekans testine dönüşür. Her bir bloktaki 1'lerin beklenen oranı $M/2$ 'dir. Testte kullanılan referans dağılımı ki-kare dağılımıdır. Testin sonunda elde edilen p-değeri çok küçük çıkması, dizideki bloklarda 1'lerin ve 0'ların oranının $1/1$'den fazlasıyla saptığını gösterir. Testin geçerli olabilmesi için blok uzunluğunun en az 20, dizi uzunluğunun da en az 100 olması gerekir (National Institute of Standard and Technology, 2001; Büyüksaraçoğlu, Buluş, 2009).

$$\pi_i = \frac{\sum_{j=1}^{M} \varepsilon_{(i-1)M+j}}{M} \qquad \chi^2(obs) = 4M \sum_{i=1}^{N} \left(\pi_i - \frac{1}{2} \right)^2 \qquad N = \left\lfloor \frac{n}{M} \right\rfloor$$

Gama Fonksiyonu: $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, dt$

Ters Gama Fonksiyonu: $\frac{1}{\Gamma(z)} \int_0^x e^{-t}\, t^{z-1}\, dt$

P-değeri = igamc (ters gama fonksiyonu) $\left(\frac{N}{2}, \frac{\chi^2(obs)}{2} \right)$

ε=10110110101101100010101010101011100111000110110001010001100100110101
01001101010010101100110011100110

$\pi_1 = \frac{3}{5},\ \pi_2 = \frac{1}{2},\ \pi_3 = \frac{1}{2},\ \pi_4 = \frac{3}{5},\ \pi_5 = \frac{1}{2},\ \pi_6 = \frac{2}{5},\ \pi_7 = \frac{1}{2},\ \pi_8 = \frac{2}{5},\ \pi_9 = \frac{3}{5},\ \pi_{10} = \frac{1}{2}$

$\chi^2(obs) = 2$

$P - value = igamc\left(\frac{N}{2}, \frac{\chi^2}{2} \right)$

$P - value = igamc(5,1) = 0.99634015$

P-değeri $\geq$ *0.01* olduğundan dizi rassal kabul edilir (Büyüksaraçoğlu, Buluş, 2009).

4.2.3. Akış (Runs) Testi

Bu test bit dizisindeki akışların toplam sayısıyla ilgilidir. Akış ardışık aynı bit sıralamasını ifade eder. Böylece 0'lar ve 1'ler arasındaki dalgalanmaların kontrolü sağlanarak üretilen bit dizisinin yavaş ya da hızlı olabileceği konusunda fikir verir (National Institute of Standard and Technology, 2001; Büyüksaraçoğlu, Buluş, 2009).

$V_n(obs)$ *:* Akışların sayısı

π *:*Bit dizisindeki 1'lerin sayısı

$$\pi = \frac{\sum j\varepsilon_j}{n} \qquad \left|\pi - \frac{1}{2}\right| \geq \tau \qquad \tau = 2/\sqrt{n}$$

$V_n(obs) = \sum_{k=1}^{n-1} r(k) + 1$ $\varepsilon_k = \varepsilon_{k+1}$ ise $r(k) = 0$ diğer durumlarda $r(k) = 1$ olarak alınır.

$$P - değ. = erfc\left(\frac{|V_n(obs) - 2n\pi(1-\pi)|}{2\sqrt{2n}\pi(1-\pi)}\right)$$

ε=10110110101101100010101010101011100111000110110001010001100100110101
01001101010010101100110011100110

$\pi = 0.51$

$\tau = 0.63246$

$r(1)=1$ $r(2)=1$ $r(3)=0$ $r(4)=1$ $r(5)=1$
$r(6)=0$ $r(7)=1$ $r(8)=1$ $r(9)=1$ $r(10)=1$
$r(11)=0$ $r(12)=1$ $r(13)=1$ $r(14)=0$ $r(15)=1$
$r(16)=0$ $r(17)=0$ $r(18)=1$ $r(19)=1$ $r(20)=1$
$r(21)=1$ $r(22)=1$ $r(23)=1$ $r(24)=1$ $r(25)=1$
$r(26)=1$ $r(27)=1$ $r(28)=1$ $r(29)=1$ $r(30)=1$
$r(31)=0$ $r(32)=0$ $r(33)=1$ $r(34)=0$ $r(35)=1$
$r(36)=0$ $r(37)=0$ $r(38)=1$ $r(39)=0$ $r(40)=0$
$r(41)=1$ $r(42)=0$ $r(43)=1$ $r(44)=1$ $r(45)=0$
$r(46)=1$ $r(47)=0$ $r(48)=0$ $r(49)=1$ $r(50)=1$
$r(51)=1$ $r(52)=1$ $r(53)=0$ $r(54)=0$ $r(55)=1$
$r(56)=0$ $r(57)=1$ $r(58)=0$ $r(59)=1$ $r(60)=1$
$r(61)=0$ $r(62)=1$ $r(63)=0$ $r(64)=1$ $r(65)=1$
$r(66)=1$ $r(67)=1$ $r(68)=1$ $r(69)=1$ $r(70)=1$
$r(71)=0$ $r(72)=1$ $r(73)=0$ $r(74)=1$ $r(75)=1$
$r(76)=1$ $r(77)=1$ $r(78)=1$ $r(79)=0$ $r(80)=1$
$r(81)=1$ $r(82)=1$ $r(83)=1$ $r(84)=1$ $r(85)=0$
$r(86)=1$ $r(87)=0$ $r(88)=1$ $r(89)=0$ $r(90)=1$
$r(91)=0$ $r(92)=1$ $r(93)=0$ $r(94)=0$ $r(95)=1$
$r(96)=0$ $r(97)=1$ $r(98)=0$ $r(99)=1$

$V_n(obs)=0.420184$

P-değeri $\geq$ *0.01*, olduğundan dizi rassal kabul edilir (Büyüksaraçoğlu, Buluş, 2009).

4.2.4. Bloktaki En Uzun Birler (Longest Run of Ones in a Block) Testi

Test, M-bitlik bloklarda bulunan en uzun birler grubu üzerinde odaklaşır. Testin tek parametresi blok uzunluğudur (M). Dizi M-bitlik n tane bloğa bölünür ve her blok içerisindeki en uzun birler öbeğinin uzunluğuna bakılır. Bu değerlerin frekansları beklenen değerlerle kıyaslanır ve ciddi bir sapma olup olmadığı kontrol edilir. Testte kullanılan referans dağılım ki-kare dağılımıdır. Dizi uzunluğuna göre blok uzunluğu ve blok sayısına karar verilir (National Institute of Standard and Technology, 2001).

Minimum *n*	***M***
128	8
6272	128
750000	10^4

v_i	***M=8***	***M=128***	***M=10^4***
v_0	≤ 1	≤ 4	≤ 10
v_1	2	5	11
v_2	3	6	12
v_3	≥ 4	7	13
v_4		8	14
v_5		≥ 9	15
v_6			≥ 16

$$P(v \leq m) = \sum_{r=0}^{M} \binom{M}{r} P(v \leq m|r) \frac{1}{2^M}$$

$$\chi^2(obs) = \sum_{i=0}^{K} \frac{(v_i - N\pi_i)^2}{N\pi_i}$$

K=5 , M=128

Sınıflar	$v \le 1$	$v = 2$	$v = 3$	$v \ge 4$
Olasılıklar	$\pi_0 = 0{,}2148$	$\pi_1 = 0{,}3672$	$\pi_2 = 0{,}2305$	$\pi_3 = 0{,}1875$

K=5 , M=128

Sınıflar	$v \le 4$	$v = 5$	$v = 6$	$v = 7$	$v = 8$	$v \ge 9$
Olasılıklar	$\pi_0 = 0{,}1174$	$\pi_1 = 0{,}2430$	$\pi_2 = 0{,}2493$	$\pi_3 = 0{,}1752$	$\pi_4 = 0{,}1027$	$\pi_5 = 0{,}1124$

K=5 , M=512

Sınıflar	$v \le 6$	$v = 7$	$v = 8$	$v = 9$	$v = 10$	$v \ge 11$
Olasılıklar	$\pi_0 = 0{,}1174$	$\pi_1 = 0{,}2460$	$\pi_2 = 0{,}2523$	$\pi_3 = 0{,}1755$	$\pi_4 = 0{,}1015$	$\pi_5 = 0{,}1077$

K=5 , M=1000

Sınıflar	$v \le 7$	$v = 8$	$v = 9$	$v = 10$	$v = 11$	$v \ge 12$
Olasılıklar	$\pi_0 = 0{,}1307$	$\pi_1 = 0{,}2437$	$\pi_2 = 0{,}2452$	$\pi_3 = 0{,}1714$	$\pi_4 = 0{,}1002$	$\pi_5 = 0{,}1088$

K=6 , M=10000

Sınıflar	$v \le 10$	$v = 11$	$v = 12$	$v = 13$	$v = 14$	$v = 15$	$v \ge 16$
Olasılıklar	$\pi_0 = 0{,}0882$	$\pi_1 = 0{,}2092$	$\pi_2 = 0{,}2483$	$\pi_3 = 0{,}1933$	$\pi_4 = 0{,}1208$	$\pi_5 = 0{,}0675$	$\pi_5 = 0{,}0727$

M	**K**	**N**
8	3	16
128	5	49
10^4	6	75

$$\textit{P-değeri} = igamc\left(\frac{K}{2}, \frac{\chi^2(obs)}{2}\right)$$

K = 3 ve M = 8

ε =1100110000010101011011000100110011100000000000010010011010101000100010011110101101000000011010111110011001110011011011000101100 10

n = 128

Alt Blok	Max-Run	Alt Blok	Max-Run
11001100	(2)	00010101	(1)
01101100	(2)	01001100	(2)
11100000	(3)	00000010	(1)
01001101	(2)	01010001	(1)
00010011	(2)	11010110	(2)
10000000	(1)	11010111	(3)
11001100	(2)	11100110	(3)
11011000	(2)	10110010	(2)

$v_0 = 4, v_1 = 9, v_2 = 3, v_3 = 0;\ \chi^2 = 4.882457$

P-değeri = 0.180609 $\geq$. 0.01 olduğundan dizi rassal kabul edilir (Büyüksaraçoğlu, Buluş, 2009).

4.2.5. Rank Testi

Bu testte sabit uzunluklu bit blokları kullanılarak, her biri bir satırı belirtecek şekilde, bir matris oluşturulur ve matrisin rankı hesaplanarak bloklar arasındaki lineer bağımlılık incelenir (National Institute of Standard and Technology, 2001).

n: Bit dizisinin boyutu

M: Her bir matristeki satır sayısı. Test için bu sayı 32 kabul edilir.

Q: Her bir matristeki sütun sayısı. Test için bu sayı 32 kabul edilir.

N: Matris sayısı $N = \left[\frac{n}{MQ} \right]$

R_l *:* Her bir matrisin rankı

F_M *:* $R_l = M$ olan matris sayısı

F_{M-1} *:* $R_l = M - 1$ olan matris sayısı

$N - F_M - F_{M-1}$ *:* Arta kalan matrislerin sayısı

$$\chi^2(obs) = \frac{(F_M - 0.2888N)^2}{0.2888N} + \frac{(F_{M-1} - 0.5776N)^2}{0.5776N} + \frac{(N - F_M - F_{M-1} - 0.1336N)^2}{0.1336N}$$

P – değeri: $e^{-\chi^2(obs)/2}$

P-değeri ≥ *0.01* olduğunda dizi rassal kabul edilir.

4.2.6. Ayrık Fourier Dönüşümü (Discrete Fourier Transform) Testi

Mevcut bit dizisinin ayrık Fourier dönüşümünü alır ve periyodikliği inceler (National Institute of Standard and Technology, 2001).

n: Bit dizisinin boyutu

d: %95 eşik değerinden fazla olan elemanların beklenen ve gerçekleşen sayısı arasındaki farkın normalize değeri.

T: %95 zayıflıktaki eşik değeri $T = \sqrt{\left(\log \frac{1}{0.05}\right)^n}$

M: Modül*(S´)* ≡ |*S*'|, *S´*bit dizisinin ilk n/2 elemanı içindeki alt dizileri temsil eder ve modül fonksiyonu zayıf eşik değerlerinin serisini üretir.

N_0*:* Teorik olarak T değerinden daha az sayıda beklenen eşik değeri sayısı

$N_0 = 0.95n/2$

N_1*:* Gözlenen ve *M*'deki T değerinden daha az sayıda beklenen eşik değeri sayısı

$$d = \frac{(N_1 - N_0)}{\sqrt{n(0.95)(0.05)/4}}$$

P-değeri: $erfc\left(\frac{|d|}{\sqrt{2}}\right)$

P-değeri ≥ *0.01* olduğunda dizi rassal kabul edilir.

4.2.7. Çakışmayan Şablon Eşleşme (Non-Overlapping Template Matching) Testi

m bitlik bir bloğun dizi içinde tekrarını inceler. Eşleşme bulunamazsa blok 1 bit kaydırılır. Tekrar edilmesi halinde, tekrar edilen bloktan itibaren m bit öteleme yapılarak yeni bir m bitlik blok oluşturulur (National Institute of Standard and Technology, 2001).

m : Her şablonun bit uzunluğu. Şablon amaçlanan koşuldur.
n: Testteki tüm bit dizisinin uzunluğudur.
ε : RNG veya test edilen PRNG tarafından üretilen bit dizisi
B: Eşleşecek m-bitlik şablon
M: Test edilecek ε'nun alt dizilerinin bit uzunluğu.
N: Bağımsız blokların sayısı. Test kodunda 8 sabit değerini alır.
Wj (*j* = 1, …, *N*): j. blok içinde oluşan *B*'lerin sayısı.

ε = 10100100101110010110 için m=3 ve B=001 alınırsa W_1=2 ve W_2=1 olur. İlgili adımlar aşağıdaki tabloda gösterilmektedir.

	Blok 1		**Blok 2**	
Bit Pozisyonları	***Bitler***	***W_1***	***Bitler***	***W_2***
1-3	101	0	111	0
2-4	010	0	110	0
3-5	100	0	100	0
4-6	001 (eşleşti)	1 arttırma	001 (eşleşti)	1 arttırma
5-7	İşlem Yok		İşlem Yok	
6-8	İşlem Yok		İşlem Yok	
7-9	001	2 arttırma	011	1
8-10	010 (eşleşti)	2	110	1

$$\mu = (M - m + 1) / 2^m \qquad \sigma^2 = M\left(\frac{1}{2^m} - \frac{2m-1}{2^{2m}}\right)$$

$$\chi^2(obs) = \sum_{j=1}^{N} \frac{(W_j - \mu)^2}{\sigma^2}$$

$$\text{P-değeri} = \text{igamc}\left(\frac{N}{2}, \frac{\chi^2(obs)}{2}\right)$$

P-değeri $\geq$ *0.01* olduğunda dizi rassal kabul edilir.

4.2.8. Çakışan Şablon Eşleşme (Overlapping Template Matching) Testi

m bitlik bir bloğun dizi içinde tekrarını inceler. Tekrar edilmesi halinde, blok 1 bit ötelenerek yenisi oluşturulur. Non-overlapping Template Matching testten tek farkı eşleşme bulunduğunda bloğun sadece 1 bit ötelenmesidir (National Institute of Standard and Technology, 2001).

m: Her şablonun bit uzunluğu. Şablon amaçlanan koşuldur.
n: Testteki tüm bit dizisinin uzunluğudur.
ε : RNG veya test edilen PRNG tarafından üretilen bit dizisi
K: Serbestlik derecesi sayısı. Test kodunda 5 sabit değerini alır.
M: Test edilecek ε'nun alt dizilerinin bit uzunluğu. Test kodunda 1032 değerini alır.
N: Bağımsız blokların sayısı. Test kodunda 968 değerini alır.
π_i : Teorik olasılıklar
v_i (i = 0, …, 5): i. blok içinde oluşan *B*'lerin sayısı.

ε = *1011101111**00101101000**1110010111**01111110000**101101001* için m=2 ve B=11 alınırsa ilk blok (*1011101111)* için ilgili adımlar aşağıdaki tabloda gösterilmektedir.

Bit Pozisyonları	***Bitler***	***B=11***
1-2	10	0
2-3	01	0
3-4	11(eşleşti)	1 arttırma
4-5	11(eşleşti)	2 arttırma
5-6	10	2
6-7	01	2
7-8	11(eşleşti)	3 arttırma
8-9	11(eşleşti)	4 arttırma
9-10	11(eşleşti)	5 arttırma

$$\lambda = (M - m + 1) / 2^m \qquad \eta = \frac{\lambda}{2}$$

$$\chi^2(obs) = \sum_{i=0}^{5} \frac{(v_i - N\pi_i)^2}{N\pi_i}$$

$$P\text{-}değeri = \text{igamc}\left(\frac{5}{2}, \frac{\chi^2(obs)}{2}\right)$$

P-değeri $\geq$ *0.01* olduğunda dizi rassal kabul edilir.

4.2.9. Evrensel (Universal) Test

Dizinin veri kaybı olmadan ne kadar sıkıştırılabileceğini inceler (National Institute of Standard and Technology, 2001).

*L :*Her bir bloğun büyüklüğü.

Q : Başlangıç dizisindeki blokların sayısı.

n : Bit dizisinin uzunluğu.

fn : Eşleşen L-bitlik şablonlar arasındaki mesafelerin *log2* tabanında toplamı

ε : Teste tabi tutulan bit dizisi

$\varepsilon = \varepsilon_1\varepsilon_2\ldots\varepsilon_n$

K : Test edilen blokların sayısı

T_j : Her bir L-bitlik bloğun blok sayısını tutan j'ye bağlı tablo değeri

sum : K bloklarında tespit edilen farklılıkların *log2* tabanında toplamı

σ : Standart sapma.

c : Sezgisel yaklaşım.

$n \geq (Q + K) \times L$

$6 \leq L \leq 16$

$Q = 10 \bullet 2^L$

$$K = \left\lceil \frac{n}{L} \right\rceil - Q \approx 1000 \times 2^L$$

L, Q ve n değerleri aşağıdaki tabloya göre seçilir.

n	**L**	**$Q=10\bullet 2^L$**
≥387.840	6	640
≥904.960	7	1.280
≥2.068.480	8	2.560
≥4.654.080	9	5.120
≥1.342.400	10	10.240
≥22.753.280	11	20.480
≥49.643.520	12	40.960
≥107.560.960	13	81.920
≥231.669.760	14	163.840
≥496.435.200	15	327.680
≥1.059.061.760	16	655.360

$$f_n = \frac{1}{K} \sum_{i=Q+1}^{Q+K} \log_2(i - T_j) = \frac{sum}{K}$$

$$P\text{-}değeri = erfc\left(\left| \frac{f_n - BeklenenDeğer(L)}{\sqrt{2}\sigma} \right| \right)$$

BeklenenDeğer (L) aşağıdaki tablodan elde edilir.

L	Beklenen Değer	Varyans
6	5.2177052	2.954
7	601962507	3.125
8	7.1836656	3.238
9	8.1764248	3.311
10	9.1723243	3.356
11	10.170032	3.384
12	11.168765	3.401
13	12.168070	3.410
14	13.167693	3.416
15	14.167488	3.419
16	15.167379	3.421

$$\sigma = c\sqrt{\frac{Varyans(L)}{K}}$$

$$c = 0.7 - \frac{0.8}{L} + \left(4 + \frac{32}{L}\right) \times \frac{K^{-\frac{3}{L}}}{15}$$

P-değeri $\geq$ *0.01* olduğunda dizi rassal kabul edilir.

4.2.10. Doğrusal Karmaşıklık (Linear Complexity) Testi

Bit dizisinin LFRS (linear feedback shift register) uzunluğuna bakarak kompleksliğini inceler (National Institute of Standard and Technology, 2001).

M: Bloktaki bit uzunluğu.

N: M bitlik bağımsız blok sayısı

n: Bit dizisinin uzunluğu. n=M.N

K: Serbestlik derecesi. Test kodunda *K = 6* olarak alınmıştır.

T_i : Alt dizi sayısı

$$\mu = \frac{M}{2} + \frac{(9+(-1)^{M+1})}{36} - \frac{\left(M/3 + 2/9\right)}{2^M}$$

$$T_i = (-1)^M \bullet (L_i - \mu) + 2/9$$

T_i değerleri için v_0 ,…,v_6 değerleri şu şekilde hesaplanır:

$Ti \leq -2.5$	v_0 1 arttırılır.
$-2.5 < Ti \leq -1.5$	v_1 1 arttırılır.
$-1.5 < Ti \leq -0.5$	v_2 1 arttırılır.
$-0.5 < Ti \leq 0.5$	v_3 1 arttırılır.
$0.5 < Ti \leq 1.5$	v_4 1 arttırılır.
$1.5 < Ti \leq 2.5$	v_5 1 arttırılır.
$Ti > 2.5$	v_6 1 arttırılır.

$$\chi^2(obs) = \sum_{i=0}^{K} \frac{(v_i - N\pi_i)^2}{N\pi_i}$$

$$P - değeri = \text{igamc}\left(\frac{K}{2}, \frac{\chi^2(obs)}{2}\right)$$

P-değeri $\geq$ *0.01* olduğunda dizi rassal kabul edilir.

4.2.11. Seri Testi

Tekrar eden m bitlik 2m tane bloğun tekrar sayısının dağılımını inceler. m=1 için, birinci teste denktir. $\nabla\psi_m^2(obs)$ ve $\nabla^2\psi_m^2(obs)$ değerleri m-bitlik öbeklerin frekansını hesaplamak için kullanılır (National Institute of Standard and Technology, 2001).

$$\psi_m^2 = \frac{2^m}{n} \sum_{i_1 \ldots i_m} v_{i_i \ldots i_m}^2 - n$$

$$\psi_{m-1}^2 = \frac{2^{m-1}}{n} \sum_{i_1 \ldots i_{m-1}} v_{i_i \ldots i_{m-1}}^2 - n$$

$$\psi_{m-2}^2 = \frac{2^{m-2}}{n} \sum_{i_1 \ldots i_{m-2}} v_{i_i \ldots i_{m-2}}^2 - n$$

$\nabla \psi_m^2 = \psi_m^2 - \psi_{m-1}^2$ ve $\nabla^2 \psi_m^2 = \psi_m^2 - 2\psi_{m-1}^2 + \psi_{m-2}^2$.

$$P - değ.1 = igamc\left(2^{m-2}, \nabla \psi_m^2\right)$$
$$P - değ.2 = igamc\left(2^{m-3}, \nabla^2 \psi_m^2\right)$$

ε=10110110101101100010101010101011100111000110110001010001100100110101
01001101010010101100110011100110

$v_{000} = 4 \quad v_{001} = 12 \quad v_{010} = 18 \quad v_{011} = 15$
$v_{100} = 12 \quad v_{101} = 20 \quad v_{110} = 15 \quad v_{111} = 2$

$v_{00} = 16 \quad v_{01} = 32 \quad v_{10} = 33 \quad v_{11} = 18$

$v_0 = 49 \quad v_1 = 51$

$$\psi_3^2 = \frac{2^3}{100}(16 + 144 + 324 + 225 + 144 + 400 + 225 + 4) - 100 = 18.56$$
$$\psi_2^2 = \frac{2^2}{100}(256 + 1024 + 1089 + 324) - 100 = 7.72$$
$$\psi_1^2 = \frac{2^1}{100}(2401 + 2601) - 100 = 0.04$$

$\nabla \psi_3^2 = 10.84$
$\nabla^2 \psi_3^2 = 3.16$

$P-değ.1 = igamc(2, 10.84) = 0.0002319$

$P-değ.2 = igamc(1, 3.16) = 0.04242574$

$P-değ.1 < 0.01$ olduğundan dizi rassal kabul edilmez.

$P-değ.2 > 0.01$ olduğundan dizi rassal kabul edilir (Büyüksaraçoğlu, Buluş, 2009).

4.2.12. Yaklaşık Entropi (Approximate Entropy) Testi

Tekrar eden m ve (m+1) bitlik blokların entropisini inceler (National Institute of Standard and Technology, 2001).

m: Her bir blok uzunluğu.

n: Tüm bit dizisinin uzunluğu.

C_i^m: Her bir i değeri için hesaplanan m-bitlik blok sayısı

ApEn(m) = $\varphi^{(m)} - \varphi^{(m+1)}$

$$C_i^m = \frac{\#i}{n}$$

$$\varphi^{(m)} = \sum_{i=0}^{2^m-1} \pi_i \log \pi_i \qquad \pi_i = C_j^3 \qquad j = \log_2^i$$

(m+1) bitlik bloklar için de bir önceki denklem değerleri hesaplanır.

$$\chi^2 = 2n[\log_2 - ApEn(m)]$$

$$P-değeri = igamc\left(2^{m-1}, \frac{\chi^2}{2}\right)$$

P-değeri $\geq$ *0.01* olduğunda dizi rassal kabul edilir.

4.2.13. Birikimli Toplamlar (Cumulative Sums) Testi

Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirler ve bloklar arasındaki dengesizlik farkına bakar (National Institute of Standard and Technology, 2001).

n: Bit dizisinin uzunluğu

mod: Test uygulaması dizinin başından sonuna doğru yapılırsa mod=0, sondan başa doğru yapılırsa mod=1'dir.

S_i = Artarak büyüyen alt dizilerin toplamı

$Mod = 0$	$Mod = 1$
$S_1 = X_1$	$S_1 = X_n$
$S_2 = X_1 + X_2$	$S_2 = X_n + X_{n-1}$
$S_3 = X_1 + X_2 + X_3$	$S_3 = X_n + X_{n-1} + X_{n-2}$
.	.
.	.
$S_k = X_1 + X_2 + ... + X_k$	$S_k = X_n + X_{n-1} + ... + X_{n-k+1}$
.	.
.	.
$S_n = X_1 + X_2 + ... + X_n$	$S_n = X_n + X_{n-1} + ... + X_{k-1} + ... + X_1$

$z = \max_{1 \le k \le n} |S_k|$

Φ = Normal Birikimli Olasılık Dağılım Fonksiyonu = $\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-u^2/2} du$

$$P\text{-değeri} = 1 - \sum_{k=\left(\frac{-n}{z}+1\right)/4}^{\left(\frac{n}{z}-1\right)/4} \left[\Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right) - \Phi\left(\frac{(4k-1)z}{\sqrt{n}}\right)\right] + \sum_{k=\left(\frac{-n}{z}-3\right)/4}^{\left(\frac{n}{z}-1\right)/4} \left[\Phi\left(\frac{(4k+3)z}{\sqrt{n}}\right) - \Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right)\right]$$

P-değeri $\geq$ *0.01* olduğunda dizi rassal kabul edilir.

4.2.14. Rassal Farklılık (Random Excursion) Testi

Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirler ve daha sonra blokların dengesinin dağılımını inceler (National Institute of Standard and Technology, 2001).

n: Bit dizisinin uzunluğu

X: Bit dizisindeki 0 ve 1'lerin toplamı (0= -1, 1=1 alınarak yapılan aritmetik toplama sonucu).

S_i : Her defasında X_1'den başlanarak adım adım arttırılan kısmi toplamlar

$$S_1 = X_1$$
$$S_2 = X_1 + X_2$$
.
.
$$S_k = X_1 + X_2 + \ldots + X_k$$
.
$$S_n = X_1 + X_2 + \ldots + X_k + \ldots + X_n$$

S': Oluşturulan S dizisinin başına ve sonuna 0 eklenerek oluşturulan yeni alt dizi

J: S' dizisinde geçen sıfırların sayısıdır. Aynı zamanda S''deki döngü sayısıdır. Döngü sayısı sıfır ile başlayan ve biten dizilerin sayısına bağlıdır.

x: Her bir döngü ve sıfır içermeyen durum sayısı. $-4 \le x \le -1$ ve $1 \le x \le 4$

$v_k(x)$: *k*'ya bağlı *x* koşulunun tüm döngülerdeki toplam sayısı. *k* = 0,1,..,5 için

$$\sum_{k=0}^{5} v_k(x) = J$$

$$\chi^2(obs) = \sum_{k=0}^{5} \frac{\left(v_k(x) - J\pi_k(x)\right)^2}{J\pi_k(x)}$$

$\pi_k(x)$: *x* koşulunun rassal bir dağılımda k kez oluşma durumudur

$$P - değeri = igamc\left(\frac{5}{2}, \frac{\chi^2}{2}\right)$$

P-değeri $\ge$ *0.01* olduğunda dizi rassal kabul edilir.

ε =0110110101

X= -1,1,1 -1,1,1 -1,1 -1,1

S= {-1,0,1,0,1,2,1,2,1,2}

S' = 0, -1,0,1,0,1,2,1,2,1,2,0

J = 3 {0,-1,0}, {0,1,0} , {0,1,2,1,2,1,2,0}

x	**Döngü Sayıları**					
	0	**1**	**2**	**3**	**4**	**5**
-4	3	0	0	0	0	0
-3	3	0	0	0	0	0
-2	3	0	0	0	0	0
-1	2	1	0	0	0	0
1	1	1	0	1	0	0
2	2	0	0	1	0	0
3	3	0	0	0	0	0
4	3	0	0	0	0	0

$v_0(-1) = 2$ (-1 durumu 0. döngüde 2 kez gerçekleştiği için)

$v_1(-1) = 1$ (-1 durumu 0. döngüde 1 kez gerçekleştiği için)

$v_2(-1) = v_3(-1) = v_4(-1) = v_5(-1) = 0$

4.2.15. Rassal Farklılık (Random Excursion) Varyans Testi

Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirleyip ortalama değerden sapma miktarını belirler. Bir önceki Rassal Farklılık Testi ile aynı stratejiyi kullanır (National Institute of Standard and Technology, 2001).

$\xi(x)$: Tüm J döngülerindeki x durumlarının meydana gelme sayısı

$$P - değeri = erfc\left(\frac{|\xi(x) - J|}{\sqrt{2J(4|x| - 2}}\right)$$

P-değeri $\geq$ *0.01* olduğunda dizi rassal kabul edilir.

TEZDE YAPILAN ÇALIŞMALAR

5. BLOK ŞİFRELER VE ANAHTAR GENİŞLETME ALGORİTMALARI

Modern blok şifreler köklerini Shannon'ın dönüm noktası olan makalesinde (Shannon, 1949) sunulan karıştırma (confusion) ve yayılım (diffusion) prensiplerinden almaktadır (Keliher, 2003). Karıştırma şifreli metin ve açık metin arasındaki ilişkiyi gizlemeyi amaçlarken, yayılım açık metindeki izlerin şifreli metinde sezilmemesini sağlamak için kullanılır. Karıştırma ve yayılım, sırasıyla yer değiştirme kutuları (S-kutuları) ve doğrusal dönüşüm işlemleri ile gerçekleştirilir.

Bir blok şifrenin tasarımında genel olarak Feistel ve SPN (Substitution Permutation Networks) mimarisi olmak üzere iki mimari kullanılır. DES (Data Encryption Standard) ve AES (Advanced Encryption Standard) sırasıyla Feistel ve SPN mimarisine örnek olarak verilebilir. Bu iki mimaride şifreleme ve deşifreleme işlemleri döngü adı verilen şifreleme adımlarının birleşmesi ile gerçekleştirilir. Diğer yandan bu mimarilerin arasındaki en temel fark döngü içerisinde bir bloğun işlenmesinde ortaya çıkmaktadır. Örneğin Feistel mimarisinde bir döngüde o anki bloğun yarısı işlenirken SPN mimarisinde o anki bloğun tümü işlenir. Buna ek olarak bir blok şifrenin genel tasarımında bir döngü içinde yer değiştirme S-kutuları ile yayılım ise doğrusal dönüşüm veya dönüşümler ile sağlanır ve her döngüde döngünün sonunda anahtar planlamadan gelen farklı bir anahtar değeri ile XOR'lama işlemi gerçekleştirilir.

Bir blok şifrenin tasarımında kullanılan 3 genel yapı aşağıdaki gibi verilebilir:

- S-kutuları (Substitution Boxes),
- Doğrusal Dönüşümler,
- Anahtar Genişletme algoritmaları.

5.1. S-kutuları (Yer değiştirme Kutuları-Substitution Boxes)

Bir $n \times n$ S-kutusu $f : \{0,1\}^n \rightarrow \{0,1\}^n$ şeklinde *n*-bit girişi farklı *n*-bit çıkışa haritalayan bir fonksiyondur. S-kutuları bir blok şifrenin içerisinde bulunan tek doğrusal olmayan yapıdır. Dolayısıyla S-kutuları için kriptografik özelliklerden biri olan doğrusal olmama özelliği önemli bir özelliktir. Bununla beraber doğrusal saldırılar için önemli olan LAT (Linear Approximation Table-Doğrusal Yaklaşım Tablosu), diferansiyel saldırılar için önemli olan DDT (Difference Distribution Table- Fark Dağılım Tablosu-XOR Tablosu), bütünlük (completeness), çığ (avalanche), katı çığ (strict avalanche) gibi kriptografik özellikler S-kutularının doyurulması gerektiren özellikler olarak karşımıza çıkmaktadır (Aslan, Sakallı, Buluş, 2008).

S-kutularının tasarım tekniklerine örnek olarak pseudo-random üretim, sonlu cisimde ters haritalama, sonlu cisimde üs haritalama ve heuristik teknikler verilebilir (Aslan, Sakallı, Buluş, 2008). Sonlu cisimde ters alma işlemi (Nyberg, 1994), üs haritalama işleminin özel bir durumu olarak görülebilir ve bu iki teknik ile doğrusal olmama ölçüsü yüksek ve diğer kriptografik özellikleri iyi S-kutuları elde edilebilir. Bunun yanında bu tasarım teknikleri kullanılarak tasarlanan S-kutuları monomial tabanlı polinomlara dayalı üs haritalama ve ters alma gibi cebirsel işlemler olduğu için doğrusal denklik (Fuller, Millan, 2003; Youssef, Tavares, 2005) ve S-kutularının cebirsel ifadesinde bazı basit cebirsel yaklaşımlar (Youssef, Tavares, Gong, 2006) gibi istenmeyen özellikleri de beraberinde getirmektedir.

Günümüzde blok şifrelerin içyapısında kullanılan S-kutuları genellikle 4×4 (4-bit giriş 4-bit çıkış) ya da 8×8 (8-bit giriş 8-bit çıkış) büyüklüğündedir. Örneğin AES blok şifresi sonlu cisimde ters haritalama yöntemiyle elde edilen 8×8 büyüklüğünde bir S-kutusu kullanmaktadır.

5.2. Doğrusal Dönüşümler

Doğrusal dönüşümler bir blok şifreye yayılım eklemek için kullanılan elemanlardır. Blok şifre mimarilerinin (Feistel ve SPN) her ikisinde de kullanılabilirler. Doğrusal dönüşümler sabit uzunluktaki bir giriş bloğunu doğrusal olarak karıştırarak aynı uzunlukta bir çıkış bloğu elde etmeyi sağlar (Z'aba, 2010). Doğrusal dönüşümlerin sağladığı yayılımın ölçülmesi için var olan teknikler aşağıdaki gibi sıralanabilir:

- Çığ Etkisi (Avalanche ciriterion) (Feistel, 1973),
- Katı çığ etkisi (Strict avalance criterion) (Webster, Tavares, 1986),
- Bütünlük (Completeness) (Kam, Davida, 1979),
- Dallanma sayısı (Branch number) (Daemen, Rijmen, 2002),
- Sabit noktalar (Fixed points) (Z'aba, 2010).

Dallanma sayısı ve sabit noktalar bu tezde üzerinde durulacak olan yayılımın ölçülmesi için verilen tekniklerdir.

Dallanma sayısı bir blok şifrede iki ardışık döngüde aktif S-kutularının minimum sayısını temsil eder. Bu sayı, bir blok şifreye karşı uygulanacak doğrusal ve diferansiyel saldırıların başarımını ölçmek için kullanılır. Çoğu yayılım elemanı doğrusal dönüşümlerdir ve matrisler ile temsil edilirler. Dolayısıyla bir yayılım elemanını $A:\left(\{0,1\}^m\right)^n \rightarrow \left(\{0,1\}^m\right)^n$ şeklinde tanımlanabilir ve (5.1) ifadesinde gösterildiği gibi bir doğrusal dönüşümdür:

$$A(x) = A.x^T = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ \dots \\ x_n \end{bmatrix} \tag{5.1}$$

Yukarıda verilen (5.1) ifadesinde $x = (x_1, x_2, \dots x_n)^T$, $x_i \in \{0,1\}^m$, $i = 1, \dots, n$ şeklindedir. Buna ek olarak *n* bir yayılım elemanındaki S-kutularının sayısını temsil

eder ve her S-kutusunun giriş ve çıkış genişliği m-bittir. *A* matrisinin elemanları $GF(2^m)$'den elemanlardır (Özellikle sonlu cisim $GF(2^8)$ ya da $GF(2)$'nin elemanları olabilir). Örneğin AES'in doğrusal dönüşümü 4×4 boyutundaki MixColumns (Sütunları karıştırma) dönüşümünün elemanları sonlu cisim $GF(2^8)$'in elemanları iken ARIA (Kwon, Kim., vd., 2004.) şifresinde kullanılan 16×16 boyutundaki doğrusal dönüşümün elemanları $GF(2)$'nin elemanlarıdır. Bir $n \times n$ boyutundaki *A* matrisinin dallanma sayısı (5.2) ifadesinde gösterildiği gibi tanımlanabilir:

$$\beta(A) = \min\left\{ wt(x) + wt(A.x^T) \mid x \in \left(\{0,1\}^m\right)^n, \mathrm{x} \neq 0 \right\} \tag{5.2}$$

Blok şifrelerin tasarımında en yüksek dallanma sayısına sahip yayılım elemanları kullanılmaktadır. Optimal dallanma sayısı da MDS (Maximum Distance Seperable) matrisler ile sağlanmaktadır. Örneğin; blok şifrelerden AES blok şifresi elemanları GF(2^8)'den olan 4×4 MDS matris kullanarak 32-bitten 32-bite dönüşüm yapan bir yayılım elemanı kullanırken, Khazad (Barreto, Rijmen, 2000) blok şifresi yine elemanları GF(2^8)'den olan 8×8 MDS matris kullanarak 64-bitten 64-bite dönüşüm yapan bir yayılım elemanı kullanmaktadır. Bu şifreler için dallanma sayısı AES'in yayılım elemanı için 5 iken Khazad blok şifresi için 9'dur (Daemen, Rijmen, 2002; Barreto, Rijmen, 2000).

Yardımcı Önerme 5.1: Bir doğrusal $[n,k,d]$-kod $d \leq n-k+1$ Singleton sınırını karşılıyorsa bu koda MDS kod adı verilir. Alternatif olarak, bir matrisin MDS matris olabilmesi için satır ve sütunlarından oluşturulan tüm alt matrislerinin determinantının 0'dan farklı olması gerekir (Nakahara, Abrahao, 2009; Youssef, Mister, Tavares, 1997).

Yardımcı Önerme 5.1 den yola çıkarak elemanları GF(2^m)'den olan bir $n \times n$ matrisin tüm hesaplanması gereken alt determinantlarının sayısı

$$\sum_{k=1}^{n-2} \left[C \binom{n}{n-k} \right]^2 \tag{5.3}$$

şeklinde verilebilir. Örneğin 4×4 bir matrisin hesaplanması gereken alt determinantlarının sayısı 2×2 alt determinantlarının sayısı ile 3×3 alt determinantlarının sayısının toplamı şeklinde elde edilebilir. Bu da yukarıdaki ifadeden $\left(C\binom{4}{2}\right)^2+\left(C\binom{4}{3}\right)^2=36+16=52$ şeklinde elde edilebilir.

Diğer yandan blok şifrelerin tasarımında kullanılan doğrusal dönüşümlerde iki önemli matris tipi bulunmaktadır. Bunlar dairesel (circulant) ve Hadamard matrislerdir. Örneğin AES blok şifresinde kullanılan sütunları karıştırma dönüşümü dairesel matris türüne girerken Khazad şifresinde kullanılan matris, Hadamard matris türüne girmektedir. Her iki matris tipi ile MDS matrisler Yardımcı Önerme 1 ile elde edilebilirken bu tür matrisler arasındaki temel fark matrislerin involutif matris (şifreleme ile deşifreleme de aynı elemanın kullanılması) olup olmamasında ortaya çıkmaktadır. Örneğin dairesel matrisler ile involutif yapılar elde edilemez. Ancak Hadamard matrisler ile bu mümkün olmaktadır. Hadamard matrislerin kullanılmasının bir avantajı şifreleme ve deşifreleme performansının aynı olduğu şifreleme algoritmalarının tasarımının yapılabilmesini sağlamasıdır.

Örnek 5.1: Elemanları x^4+x+1 ile tanımlı GF(2^4) sonlu cisminden olan 4×4 genişliğinde biri dairesel formda ve diğeri Hadamard formda 16-bitten 16-bite dönüşüm yapan iki doğrusal dönüşümü örnek olarak gösterelim.

Dairesel formdaki matrisler ilk satırdaki elemanların birer sağa ötelenerek diğer satırların oluşturulduğu matrislerdir. Örneğin;

$$D(3_h,4_h,B_h,D_h)=\begin{bmatrix} 3_h & 4_h & B_h & D_h \\ D_h & 3_h & 4_h & B_h \\ B_h & D_h & 3_h & 4_h \\ 4_h & B_h & D_h & 3_h \end{bmatrix}_{4\times 4}$$

elemanları GF(2^4) sonlu cisminden dairesel formda bir matristir. Yine daha önce verilen Yardımcı Önerme 5.1 kullanılarak tüm alt determinantları tanımlanan indirgenemez

polinomuna göre incelendiğinde 52 alt determinanttan $\begin{vmatrix} D_h & 3_h \\ B_h & D_h \end{vmatrix} = 0$, $\begin{vmatrix} B_h & D_h \\ D_h & 3_h \end{vmatrix} = 0$, $\begin{vmatrix} 3_h & D_h \\ D_h & B_h \end{vmatrix} = 0$ olduğu için verilen örnek dairesel matris MDS matris değildir.

Hadamard formdaki matrisler 4×4 bir matris için aşağıdaki formdadır:

$$Had(a_1, a_2, a_3, a_4) = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ a_2 & a_1 & a_4 & a_3 \\ a_3 & a_4 & a_1 & a_2 \\ a_4 & a_3 & a_2 & a_1 \end{bmatrix}_{4\times4}$$

Aynı elemanlar ve indirgenemez polinom ile tanımlı Hadamard formdaki matrisi düşünelim:

$$Had(3_h, 4_h, B_h, D_h) = \begin{bmatrix} 3_h & 4_h & B_h & D_h \\ 4_h & 3_h & D_h & B_h \\ B_h & D_h & 3_h & 4_h \\ D_h & B_h & 4_h & 3_h \end{bmatrix}_{4\times4}$$

Yine daha önce verilen Yardımcı Lemma kullanılarak tüm alt determinantları tanımlanan indirgenemez polinomuna göre incelendiğinde 52 alt determinantın hepsi 0'dan farklı olduğu için bu yayılım elemanı MDS bir matristir. Dolayısıyla dallanma sayısı optimal değer olan 5'tir.

Diğer yandan Camellia (Aoki, Ichikawa, vd., 2001) ve ARIA şifresinde olduğu gibi yüksek dallanma sayısına sahip MDBL (Maximum Distance Binary Linear Codes) kodlarda yayılım elemanı olarak kullanılmaktadır. Bu ikili doğrusal dönüşümler verilen şifrelerde sırasıyla 8×8 ve 16×16 boyutunda matrislerdir. Bunun yanında bu boyutlardaki matrisler için optimal dallanma sayısı değerleri sırasıyla 5 ve 8 olduğu (Kang, Hong, vd., 2001; Kwon, Sung, vd., 2005) çalışmalarında verilmiştir.

Doğrusal dönüşümlerde *sabit noktaların* önemi (Z'aba, 2010)'da verilmiştir. Bu çalışmada doğrusal dönüşümdeki sabit nokta sayısı bir rastlantısal doğrusal dönüşümde olması gerekenden çok daha fazla sayıda olursa bunun yayılım elemanının kötü bir yayılım sağladığının göstergesidir denmektedir. Rastlantısal bir doğrusal dönüşümde beklenen sabit nokta sayısı 1'dir.

Bir doğrusal dönüşüme bir giriş bloğunun GF(2^m) den m-bit değerlerden oluştuğunu varsayalım. Buna ek olarak doğrusal dönüşüm matrisinin $n \times n$ boyutunda ve I matrisinin $n \times n$ boyutunda birim matris olduğunu varsayalım. O zaman doğrusal dönüşüm matrisi A (determinantı 0'dan farklı) için tüm sabit noktaların sayısı aşağıdaki denklemin çözülmesi ile elde edilebilir:

$$(A-I)x^T = 0 \tag{5.4}$$

(5.4) ifadesinde 0, n uzunluğunda tüm elemanları 0 olan vektörü temsil etmektedir. Buradan yola çıkarak sabit noktaların sayısı aşağıdaki gibi verilebilir:

$$F_A = 2^{m(rank(A)-rank(A-I)} = 2^{m(n-rank(A-I))} \tag{5.5}$$

(5.5) ifadesinden anlaşılacağı gibi *(A-1)* matrisinin daha büyük bir rank değerine sahip olması A doğrusal dönüşümünün daha az sayıda sabit noktaya sahip olacağının göstergesidir.

Örnek 5.2: Aşağıda verilen 8×8 büyüklüğünde elemanları GF(2) den olan bir ikili doğrusal dönüşümü düşünelim.

$$A=\begin{bmatrix}1&0&0&1&0&1&0&1\\1&1&0&1&1&0&1&0\\0&1&1&1&1&1&0&1\\0&0&1&0&1&0&1&1\\1&0&1&1&1&0&0&1\\1&1&0&0&1&1&0&1\\1&1&1&1&0&1&1&1\\0&1&0&1&0&0&1&1\end{bmatrix}$$

Verilen doğrusal dönüşüm (5.2) ifadesinde verilen denklem tüm giriş ve çıkışlar için incelendiğinde dallanma sayısı optimal değer olan 5 olarak elde edilebilir (MDBL kod). Buna ek olarak bu matrisin aşağıda verilen $(A-I)$ matrisinin rankı da 8 olarak elde edilebileceğinden bu yayılım elemanı giriş vektörü 8-bit (byte) elemanlardan oluşacak şekilde 64-bitten 64-bite bir dönüşüm olarak kullanılırsa sabit nokta sayısı aşağıda gösterildiği gibi 1 olarak elde edilir.

$$F_A = 2^{m(rank(A)-rank(A-I))} = 2^{8(8-8)} = 2^0 = 1$$

Diğer yandan $(A-I)$ matrisinin rankını 7 olarak elde etmiş olsaydık aynı dönüşümün $2^{8(8-7)} = 2^8$ adet sabit nokta içereceğini söyleyebilirdik.

$$A-I=\begin{bmatrix}0&0&0&1&0&1&0&1\\1&0&0&1&1&0&1&0\\0&1&0&1&1&1&0&1\\0&0&1&1&1&0&1&1\\1&0&1&1&0&0&0&1\\1&1&0&0&1&0&0&1\\1&1&1&1&0&1&0&1\\0&1&0&1&0&0&1&0\end{bmatrix}$$

5.3. Anahtar Genişletme Algoritmaları

Bir blok şifre daha önce de belirtildiği gibi döngülerden ve döngülerdeki aynı adımlardan oluşmaktadır. Dolayısıyla döngülerdeki simetriyi bozmak için her döngüde farklı bir anahtar materyalinin kullanılması gereklidir. Anahtar genişletme algoritmaları gizli anahtardan her döngüde kullanılacak farklı anahtarların (alt anahtarların) elde edilmesini sağlayan algoritmalardır. Her blok şifrede farklı algoritmalar kullanılabilmektedir ve şifreleme algoritmasında kullanılan yapılar tercih edilerek bu algoritmalar geliştirilebilir. Lars Knudsen (Knudsen, 1993) güçlü bir anahtar planlamanın özelliklerini aşağıdaki gibi vermektedir:

1- Çarpışmaya dayanıklı tek yönlü fonksiyon (one-way function) olma,
2- Tüm alt anahtarlar ve gizli anahtar arasında minimum karşılıklı ilişki bulunma,
3- Uygulama etkinliği.

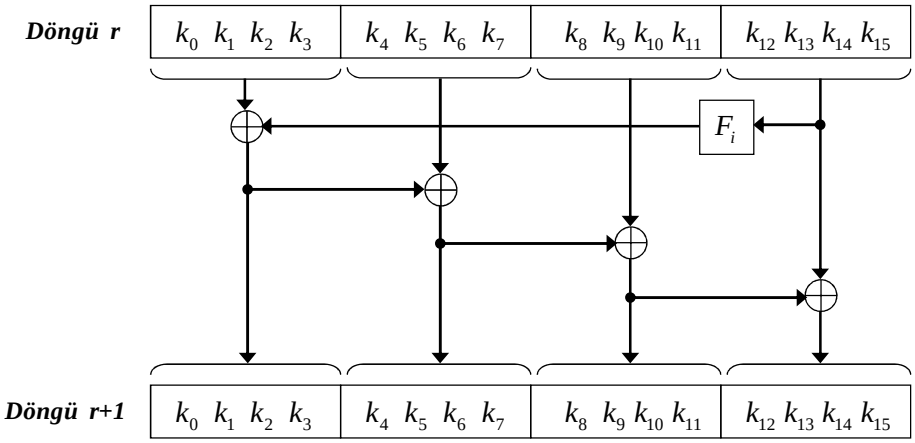
Bir blok şifre, gizli anahtar bilinmediği zaman tek yönlü bir fonksiyon olarak düşünülebilir. Dolayısıyla alt anahtarların üretilmesinde şifreleme algoritmasının kullanımı alt anahtarlarının tersinirlik (invertibility) özelliğinin olamaması için yeterli bir teknik olarak düşünülebilir (May, Henricksen, vd., 2002).

Tüm alt anahtarlar ve gizli anahtar arasında minimum karşılıklı ilişki özelliği blok şifreler üzerine saldırı senaryolarının karmaşıklığını azaltarak saldırgana yardımcı olacak ilişkileri yok edecektir (May, Henricksen, vd., 2002). Bu tür ilişkilerin kullanıldığı saldırılara örnekler DES blok şifresine karşı doğrusal kriptanaliz (Matsui, 1994), diferansiyel kriptanaliz (Biham, Shamir, 1990) gibi saldırılar ile AES blok şifresine karşı olan çeşitli saldırılar verilebilir. Yine (Ferguson, Kelsey, vd., 2000) çalışmasının yazarları “Bazı saldırıların genişletilen anahtar bitleri arasındaki ilişkileri kullandıklarını ve bu ilişkilerin olamaması durumunda saldırıların daha yüksek karmaşıklık gerektireceğini” belirtmişlerdir.

Şifreleme algoritması ve anahtar genişletme algoritması güvenlik açısından olduğu kadar uygulama yönüyle de birbirlerini tamamlamalıdırlar. Bu açıdan bakıldığında anahtar planlama algoritmasında, şifreleme algoritmasında kullanılan optimize edilen elemanların tekrar kullanılması bir avantaj olarak kabul edilebilir (May, Henricksen, vd., 2002).

Anahtar genişletme algoritmaları ile elde edilen alt anahtarların üzerinde yürütülen iki önemli test, frekans testi ve katı çığ kriteri testidir. Frekans testi, bit karıştırma özelliğinin ölçülmesinde (Shannon'nın karıştırma özelliğinin ölçülmesinde temel teşkil eder) kullanılırken katı çığ kriteri testi, bit yayılım özelliğinin ölçülmesinde kullanılır. Bu test, giriş bloğunda bir bit değişimin çıkış bloğundaki bitlerin yarısının değişimini kontrol eder (Shannon'nın yayılım özelliğinin ölçümünü sağlar).

AES-128 blok şifresinin anahtar genişletme algoritması düşünüldüğünde, Şekil 5.1'de genel formu verilmiştir, yukarıda verilen özelliklerden sadece üçüncü özelliği sağladığı (May, Henricksen, vd., 2002) belirtilmiştir. Bunun yanında AES'in anahtar genişletme algoritmasının kötü yayılım özelliği ilişkili anahtar saldırıları gibi bazı saldırılarda etkin olarak kullanılmaktadır. Bu tür saldırılar gerçek hayatta her ne kadar pratik olmasalar da AES-192 (192-bit anahtar kullanan AES blok şifresi) ve AES-256 (256-bit anahtar kullanan AES blok şifresi) için ilişkili anahtar saldırıların ne kadar faydalı olduğu (Biryukov, Khovratovich, 2009; Biryukov, Khovratovich, Nikolic, 2009) çalışmalarında gösterilmiştir. Bunun temel nedeni olarak AES-192 ve AES-256 versiyonlarındaki anahtar planlama algoritmasının AES-128 (128-bit anahtar kullanan AES blok şifresi) versiyonuna göre daha yavaş yayılım özelliği sağlaması olarak verilebilir. Ayrıca zaman karmaşıklığı açısından Biryukov vd. (Biryukov, Dunkelman, vd., 2009) 10 döngüye kadar bir AES algoritmasına pratik bir saldırıyı göstermişlerdir.

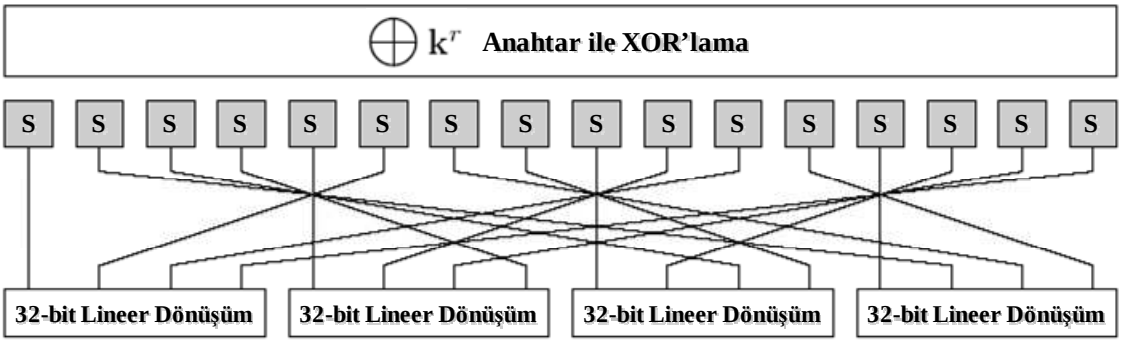


Şekil 5.1. AES-128 Blok Şifresinin Anahtar Genişletme Algoritmasının Genel Formu (Rimoldi, 2009)

Diğer yandan AES'in anahtar genişletme algoritmasında bit sızıntısı (bit leakage) problemi bulunmaktadır. Bu problem kullanılarak çeşitli saldırılarda bir alt anahtardan faydalanarak diğer alt anahtardan parçalar elde edilebilmektedir. Örneğin (Phan, 2004) çalışmasında bu sızıntı problemi kullanılmıştır. Bu problemin önüne geçmek için alt anahtarların birbirinden bağımsız olarak üretilmesi bir yöntem olarak kullanılabilir.

5.4. AES (Advanced Encryption Algorithm-İleri Şifreleme Standardı)

2001 yılında DES şifreleme algoritmasının yerini alan ve standart haline getirilen AES blok şifresi 128-bit veri bloklarını 128-bit, 192-bit ve 256-bit anahtar seçenekleri ile şifreleyen bir blok şifreleme algoritmasıdır (Behrouz, 2008; National Institute of Standards and Technology, 2001). Döngü sayısı 128-bit, 192-bit ve 256-bit anahtar seçenekleri için sırasıyla 10, 12 ve 14 döngüdür. Her döngü dört adım içerir: SubBytes (Byte Yerdeğiştirme), ShiftRows (Satırları Öteleme), MixColumns (Sütunları Karıştırma), AddRoundKey (Döngü Anahtarı Ekleme). Her döngüde sırasıyla gerçekleştirilen bu adımlardan byte yerdeğiştirme adımında 8-bit (byte) değerleri farklı 8-bit (byte) değerleri ile yer değiştirilir. Bu dönüşüm doğrusal olmayan bir dönüşümdür ve GF(2^8) sonlu cisminde ters haritalama tabanlıdır (Aslan, Sakallı, Buluş, 2008; Aslan, Sakallı, Aslan, 2010). Satırları öteleme adımında byte değerlerinin permütasyonu ile byte değerlerinin sırası değiştirilirken, MixColumns doğrusal dönüşümde 32-bit giriş değerlerinden sabit bir matris çarpımı yardımıyla 32-bit çıkış değerleri elde edilmektedir. Diğer yandan son adım olan döngü anahtarı ekleme evresinde 128 bit anahtar seçeneği ile şifreleme yapan AES şifresi için anahtar planlama evresinden gelen 128-bit anahtar değer ile o anki blok XOR'lama işlemine tabi tutulur. Şekil 5.2, tek döngülük SPN mimarisine uygun AES algoritmasını göstermektedir (Teknik olarak alt anahtar doğrusal dönüşümden sonra XOR işlemine sokulur. Şekil 5.2'de anahtar ile XOR'lama döngünün başında gösterilmektedir. Çünkü ek alt anahtar ilk döngüden önce XOR'lama işlemine girmektedir).



Şekil 5.2. Tek döngülük AES algoritması (Keliher, 2003)

AES şifresinde döngü yapısına ait özellikler aşağıda verilmiştir:

1. Her döngü tersi alınabilir dönüşümler kullanır,
2. Her döngü, son döngü hariç, 4 dönüşüm kullanır: SubBytes, ShiftRows, MixColumns ve AddRoundKey,
3. Son döngü de MixColumns dönüşümü göz ardı edilir,
4. Her döngüde farklı anahtar materyali kullanılır,
5. Farklı anahtar materyalleri anahtar planlama evresinde gelen anahtarlardır. Gizli anahtardan farklı anahtarlar elde edilerek şifrede kullanılır,
6. Deşifreleme kısmında ters dönüşümler kullanılır: InvSubByte, InvShiftRows, InvMixColumns ve AddRounKey (tersi kendisidir- XOR işlemi).

5.4.1. SubBytes (Byte Yerdeğiştirme) Dönüşümü

AES şifresi her byte (8-bit) değere karşılık farklı bir byte değerine dönüşümü yapan ve şifreye doğrusal olmama özelliğini katan bir S-kutusu kullanır. S-kutusunun tasarımı iyi kriptografik özellikler veren sonlu cisimde ters haritalama işlemi kullanılarak yapılmıştır. AES şifresinde kullanılan S-kutusu değerleri

$GF(2^8) = Z_2(x)/(x^8 + x^4 + x^3 + x + 1)$ sonlu cisminde $x \to x^{-1}$ ters haritalama işleminin çıkışına aşağıdaki verilen bitsel doğrusal dönüşüm uygulanarak elde edilmiştir.

$$L_A(x) = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Bu doğrusal dönüşümün eklenmesi S-kutusunun kriptografik özellikleri açısından herhangi bir iyileştirme yapmamakla beraber cebirsel ifadesini daha karmaşık hale getirmektedir. GF(2^8) de cebirsel ters alma işleminin cebirsel ifade x^{-1} veya x^{254} olarak ifade edilirken doğrusal dönüşüm eklenerek bu ifade aşağıdaki şekli almıştır:

$$S(x) = '05'x^{254} + '09'x^{253} + 'f9'x^{251} + '25'x^{247} + 'f4'x^{239} + '01'x^{223} + 'b5'x^{191} + '8f'x^{127} + '63'.$$

Diğer bir ifadeyle S-kutusunun cebirsel ifadesi geliştirilmiştir. Yine S-kutusunun sonlu bir cisimde ters alma işlemi ile tasarlanmasındaki amaç doğrusal kriptanaliz ve diferansiyel kriptanalize karşı dayanıklı bir S-kutusu seçme amaçlıdır. Örneğin AES S-kutusu doğrusal olmama değeri 112 ve fark dağılım tablosundaki en büyük değer 4'tür. AES S-kutusu Tablo 5.1 de Hexadecimal formda verilmiştir.

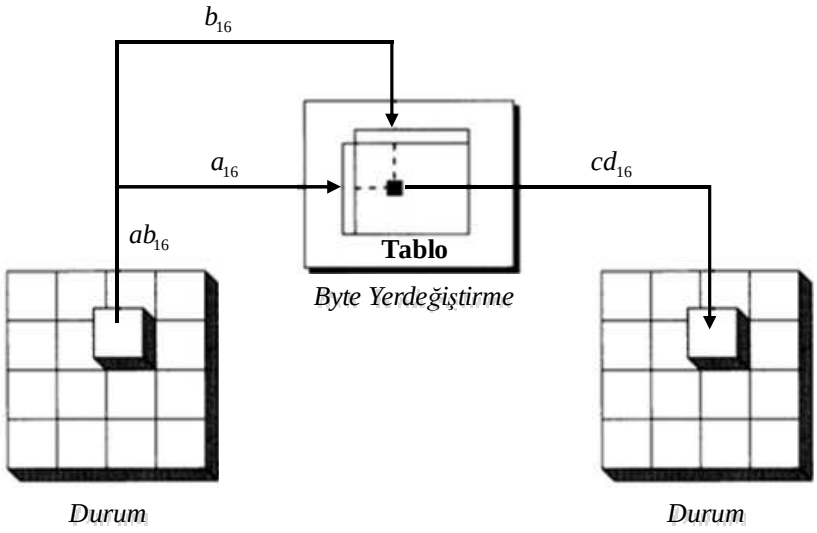
Tablo 5.1. AES S-kutusu

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Örnek 5.3: $GF(2^8) = Z_2(x)/(x^8 + x^4 + x^3 + x + 1)$ cismi tanımlansın. Buna ek olarak $f : x \mapsto x^{-1}, x \in \mathrm{GF}(2^8)$ haritalaması için bu fonksiyon 8-bit girişli ve 8-bit çıkışlı AES S-kutusu tanımlar. Yukarıdaki doğrusal dönüşümü de kullanarak Hexadecimal "D3" değerine karşı Hexadecimal "66" değerini elde edelim. Cismi, verilen indirgenemez polinom primitive (asal) olmadığı için $\alpha + 1$ elemanını kullanarak elde edebiliriz. Örneğin (11010011) bit temsili ile gösterilen hexadecimal "D3" değeri tanımlanan cisimde $(\alpha+1)^{60} = \alpha^7 + \alpha^6 + \alpha^4 + \alpha + 1$ şeklindedir. $f : x \mapsto x^{-1}$ veya $f : x \mapsto x^{254}$ haritalaması ile $(\alpha+1)^{60} \to ((\alpha+1)^{60})^{-1} \to (\alpha+1)^{-60} \to (\alpha+1)^{195}$ şeklinde haritalanır. $a \in GF(2^n)$ için $z^a = z^{a \bmod 2^n - 1}$ bilgisinden de faydalanarak $(-60) \bmod 255 = 195$ olacağından $(\alpha+1)^{60} \to (\alpha+1)^{195} \to \alpha^6 + \alpha^5 + \alpha + 1$ veya Hexadecimal gösterimle $"D3" \to "63"$ şeklinde haritalanır. Daha sonra Hexadecimal "63" değeri doğrusal dönüşüme sokularak aşağıda gösterildiği gibi Hexadecimal "66" değeri elde edilir.

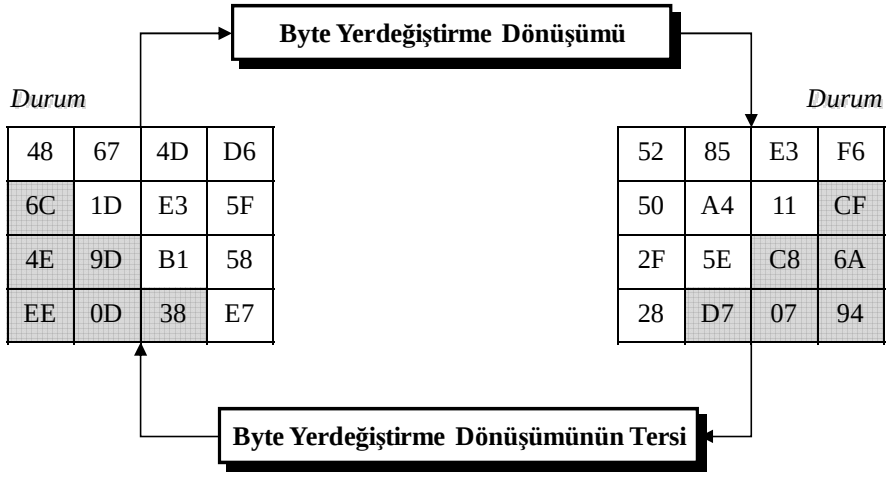
$$S(x) = "66" = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Şekil 5.3, bu dönüşümü 4×4 boyutunda bir durum matrisi üzerinde göstermektedir.



Şekil 5.3. AES Şifresinde Byte Yerdeğiştirme Dönüşümü (Stinson, 2002)

Örnek 5.4: Byte yerdeğiştirme dönüşümünü 4×4 durum matrisi üzerinde aşağıdaki gibi bir örnekle verebiliriz (Durum matrisinde veri Hexadecimal formda verilmiştir).



S-kutusunun tersi algoritmanın deşifreleme aşamasında kullanılır ve Tablo 5.2 de Hexadecimal formda verilmiştir.

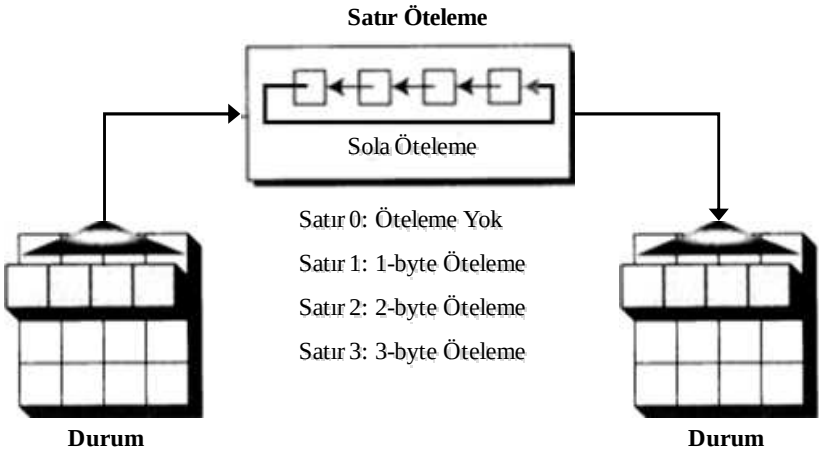
Tablo 5.2. AES S-kutusunun Tersi

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

5.4.2. ShiftRows (Satırları Öteleme) Dönüşümü

AES şifresi 4×4 boyutunda bir durum matrisi şeklinde değerlendirilirse satırları öteleme dönüşümü byte değerlerinin sola öteleme işlemidir. İlk satırda sola öteleme

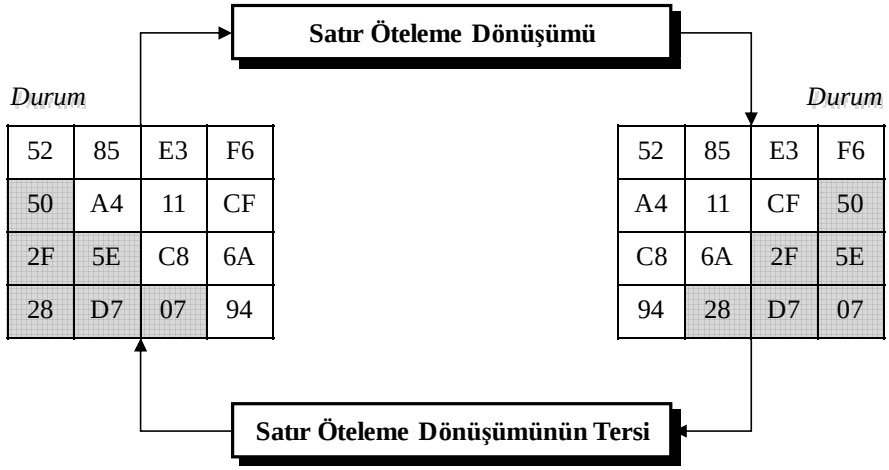
yapılmaz iken ikinci, üçüncü ve dördüncü satırlar sırasıyla 1 defa, 2 defa ve 3 defa sola ötelenir. Diğer yandan Şekil 5.2'deki gibi bir SPN mimarisine uyan şekil düşünüldüğünde 1. byte 1. byte'a, 2. byte 14. byte'a, 3. byte 11. byte'a ve 4. byte 8. byte'a ve bu şekilde byte değerlerinin permütasyonudur. Şekil 5.4 bu dönüşümü 4×4 boyutunda bir durum matrisi üzerinde göstermektedir.



Şekil 5.4. AES Şifresinde Satırları Öteleme Dönüşümü (Stinson, 2002)

Satırları öteleme dönüşümü bir sonraki bölümde anlatılacak MixColumns (Sütunları Karıştırma) dönüşümüne döngü içerisinde destek verme amacıyla ikincil olarak eklenmiş bir dönüşümdür. Bu dönüşümün eklenmesinin temel nedeni sütunları karıştırma dönüşümü 32-bitten 32-bite haritalama yaptığı için bu dönüşümü destekleme gereğidir. Örneğin sütunları karıştırma dönüşümü 128-bitten 128 bite bir doğrusal dönüşüm olmuş olsaydı bu dönüşüme gerek kalmayacaktı. Ancak böyle bir dönüşümün kullanılması performans sıkıntılarını da beraberinde getirecekti.

Örnek 5.5: Satırları öteleme dönüşümünü 4×4 durum matrisi üzerinde aşağıdaki gibi bir örnekle verebiliriz (Durum matrisinde veri Hexadecimal formda gösterilmiştir).



5.4.3. MixColumns (Sütunları Karıştırma) Dönüşümü

AES şifrenin içyapısında 32-bitten 32 bite dönüşüm yapan bir doğrusal dönüşüm içerir ve bu dönüşüm **MixColumns** (sütunları karıştırma) olarak isimlendirilir. Bu dönüşüm doğrusal ve diferansiyel kriptanalizi zorlaştırıcı etki yapma amacındadır ve sonlu cisimde çarpma tabanlıdır. Aşağıda görüldüğü gibi bu matris dairesel matris formundadır ve elemanları olabildiğince küçük seçilmiştir. Buna ek olarak bu dönüşüm bir MDS matristir.

$$\begin{bmatrix} a & b & 1 & 1 \\ 1 & a & b & 1 \\ 1 & 1 & a & b \\ b & 1 & 1 & a \end{bmatrix}$$

Yukarıdaki dönüşümde $a = x$ ve $b = x + 1$ olarak seçilmiştir. Buna ek olarak matris elemanları 8-bit ya da GF(2^8) cisminden elemanlardır.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}$$

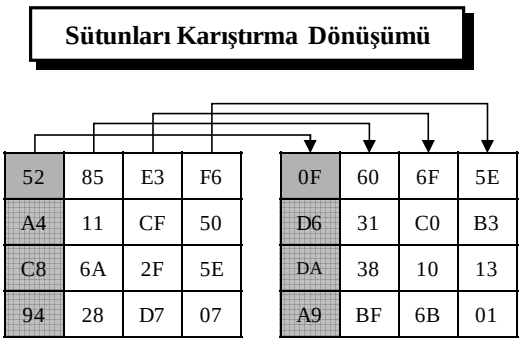
Sonuç olarak 32 bitlik dönüşüm $y_{0,}....y_3, a_0,...a_3$ 8-bit değerleri yani 1 byte değerleri temsil etmek üzere sütunları karıştırma dönüşümü

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

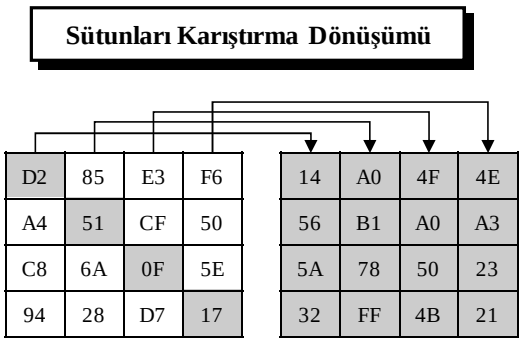
şeklinde gösterilebilir. Sonlu cisimde çarpma ve XOR işlemleri ile temsil edilebilecek bu dönüşüm indirgeme işlemleri için $x^8 + x^4 + x^3 + x + 1$ polinomunu kullanır. Daha önce belirtildiği gibi dairesel matrisler involutif olamayacağından dolayı bu dönüşümün tersi aşağıdaki gibi verilebilir:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix}$$

Örnek 5.6: Aşağıda sütunları karıştırma dönüşümüne 4 byte girişe karşılık 4 byte çıkış ve 1-bit değişim sonucunda (1 byte değişimi) çıkıştaki 4 byte değişim gösterilmiştir. Ayrıca bu örnek dallanma sayısının 5 olduğunu doğrulamaya yönelik örnek bir gösterimdir (Durum matrisinde veri Hexadecimal formda gösterilmiştir).



a-) *Sütunları Karıştırma Dönüşümü*



b-) *Sütunları Karıştırma Dönüşümünde 1-bit ya da 1-byte Değişiminin Etkisi*

5.4.4. AddRoundKey (Döngü Anahtarı Ekleme)

Döngü içerisindeki bu son aşamada anahtar planlama evresinden o döngü için elde edilen anahtar ile sütunları karıştırma dönüşüm çıkışı XOR işlemine tabi tutulur. Bu işleme beyazlatma (whitening) adı da verilir. Bir blok şifrede güvenliğin artırılması amacıyla uygulanır.

5.5. AES Şifresinde Anahtar Genişletme Algoritması

AES şifresi 128 bit veri bloklarını 128, 192, 256 bit anahtar seçenekleri ile şifreleyen bir algoritma olduğu için döngü sayısı anahtar genişliğine göre değişmektedir. 128 bit anahtar için 10 döngüde şifreleme yaparken 192 ve 256 bit anahtarlar için sırasıyla 12 ve 14 döngüde şifreleme yapmaktadır. Dolayısıyla AES-128 için her döngüde kullanılmak üzere 10 farklı anahtar üretmek gerekirken AES-192 ve AES-256 için sırasıyla 12 ve 14 adet anahtar gereklidir.

Her döngüde kullanılmak üzere döngü anahtarı yaratmak için AES bir anahtar genişletme işlemi kullanır. Eğer döngü sayısı N_r ise anahtar genişletme işlemi $N_r + 1$ adet 128-bit döngü anahtarını tek bir 128 bit gizli anahtardan elde eder. Gizli anahtar döngü başlamadan önce kullanılırken geri kalan döngü anahtarları her döngünün sonundaki son dönüşüm olarak kullanılır.

Anahtar genişletme rutini döngü anahtarlarını kelime kelime (word-32bit 32 bit) yaratır. Rutin aşağıdaki gibi tanımlanan $4 \times (N_r + 1)$ kelime yaratır.

$$w_0, w_1, w_2, ..., w_{4(N_r+1)-1}$$

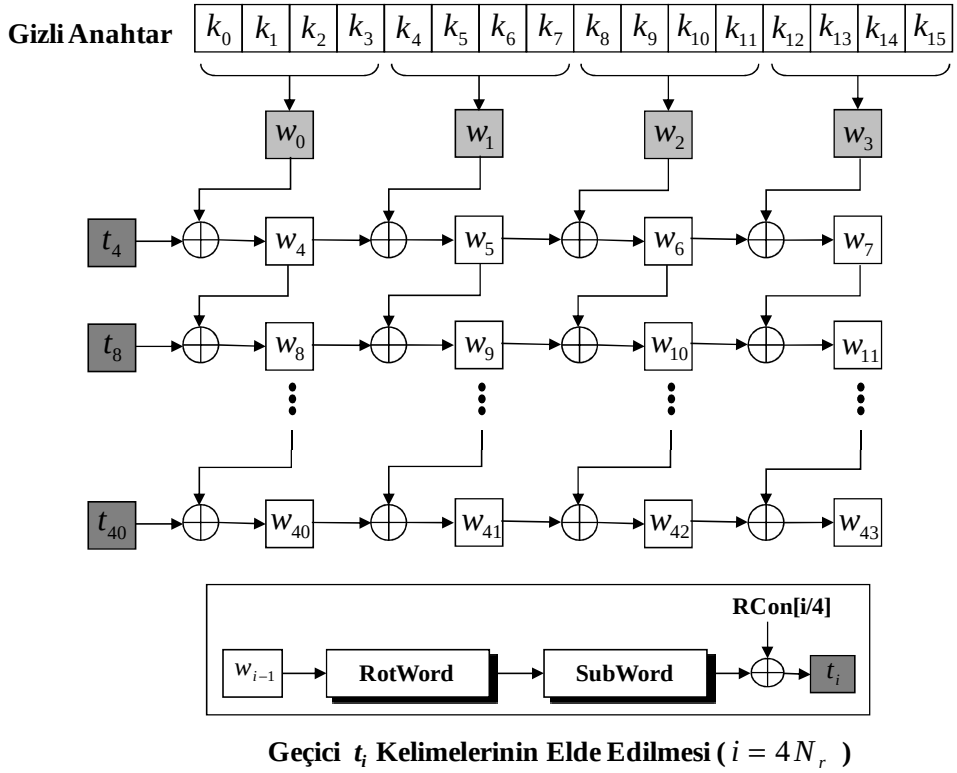
Diğer bir deyişle AES-128 şifresinde 44 kelime, AES-192 şifresinde 52 kelime ve AES-256 şifresinde 60 kelime büyüklüğünde anahtar bulunur. Şekil 5.5 döngüler ve kelimeler arasındaki ilişkiyi göstermektedir:

Döngüler	**Kelimeler**
Döngü Öncesi	w_0 w_1 w_2 w_3
1	w_4 w_5 w_6 w_7
2	w_8 w_9 w_{10} w_{11}
...	...
N_r	w_{4N} w_{4N_r+1} w_{4N_r+2} w_{4N_r+3}

Şekil 5.5. AES Şifresinin Anahtar Genişletme Algoritmasında Döngüler ve Kelimeler Arasındaki İlişki

5.5.1. AES-128'de Anahtar Genişletme

Bu bölümde AES-128 için anahtar genişletme algoritması incelenecektir. Blok şifrenin AES-192 ve AES-256 iki versiyonu için de bazı küçük değişiklikler ile birlikte aynıdır. Şekil 5.6'da orijinal anahtardan 44 kelimenin nasıl yaratıldığı gösterilmektedir.



Şekil 5.6. AES -128 için Anahtar Genişletme Algoritması (Stinson, 2002)

Şekil 5.6'da genel şekli verilen AES-128 anahtar genişletme algoritması aşağıdaki gibi ifade edilebilir:

1. İlk 4 kelime (w_0, w_1, w_2, w_3) gizli anahtardan elde edilir. Gizli anahtar k_0 dan k_{15}'e kadar 16 byte bir dizi olarak düşünülür. İlk 4 byte (k_0 dan k_3'e) w_0, ikinci 4 byte (k_4'ten k_7'ye) w_1 ve benzer şekilde diğer kelimeler w_2 ve w_3'te gizli anahtarın kelimeler şeklinde yan yana konması ile elde edilir,
2. Diğer kelimeler w_i (i = 4 den 43'e kadar) aşağıdaki şekilde elde edilir:
    a. Eğer i (mod 4) $\neq$ 0 ise $w_i = w_{i-1} \oplus w_{i-4}$ şeklinde tablodan da görüldüğü gibi soldan ve üstten bir değerden elde edilir.
    b. Eğer i (mod 4) = 0 ise $w_i = t \oplus w_{i-4}$ şeklinde elde edilir. Burada t geçici bir bellek ve iki rutinin w_{i-1} üzerindeki uygulama sonucudur: SubWord ve RotWord. t'nin elde edilme süreci bir döngü sabiti RCon

ile XOR lama işlemi ile sonlanır. Diğer bir deyişle; $t = SubWord\ (RotWord\ (w_{i-1})) \oplus RCon_{i/4}$.

Kelime Döndürme (RotWord):

Bu rutin AES şifresinde kullanılan satırları öteleme (ShiftRows) dönüşümüne benzemektedir. Ancak sadece 1 satıra uygulanır. Bu rutin bir kelimeyi 4 bytelık bir dizisi olarak alır ve her byte'ı sola dairesel olarak öteler.

Kelime Yer değiştirme (SubWord):

Bu rutin AES şifresinde kullanılan Byte yerdeğiştirme (SubBytes) dönüşümüne benzemektedir. Ancak sadece 4 byte'a uygulanır. Bu döngü kelimedeki her byte değerini alır ve diğer bir byte ile yerdeğiştirir.

Döngü Sabitleri (Round Constants):

Anahtar genişletme algoritması her döngüde farklı sabit değer kullanır. Bu sabit, RCon, 4 byte değerinde ve en sağdaki 3 byte'ı 0 olan bir değerdir. AES-128 için (10 döngü) için farklı döngü sabitlerini Tablo 5.3'te gösterilmektedir.

Tablo 5.3 AES-128 Anahtar Genişletme Algoritmasında Kullanılan Döngü Sabitleri (Stinson, 2002)

Döngü	***Döngü Sabiti (RCon)***	***Döngü***	***Döngü Sabiti (RCon)***
1	(**01** 00 00 00)$_{16}$	**6**	(**20** 00 00 00)$_{16}$
2	(**02** 00 00 00)$_{16}$	**7**	(**40** 00 00 00)$_{16}$
3	(**04** 00 00 00)$_{16}$	**8**	(**80** 00 00 00)$_{16}$
4	(**08** 00 00 00)$_{16}$	**9**	(**1B** 00 00 00)$_{16}$
5	(**10** 00 00 00)$_{16}$	**10**	(**36** 00 00 00)$_{16}$

Anahtar genişletme algoritması anahtar kelime değerlerini hesaplarken döngü sabitlerini için ya yukarıdaki tabloyu kullanır ya da en soldaki byte değerini dinamik olarak hesaplar ve bu hesaplama işleminde GF(2^8) cismini kullanır. AES şifresinde daha önce de verildiği gibi indirgenemez polinom $x^8 + x^4 + x^3 + x + 1$ kullanılır. En soldaki byte, RC_i, x^{i-1} dir ve *i* değeri döngü sayısını temsil eder.

RC_1	$\rightarrow x^{1-1}$	$=x^0$	mod	$x^8+x^4+x^3+x+1$	$=1$	$\rightarrow 00000001$	$\rightarrow 01_{16}$
RC_2	$\rightarrow x^{2-1}$	$=x^1$	mod	$x^8+x^4+x^3+x+1$	$=x$	$\rightarrow 00000010$	$\rightarrow 02_{16}$
RC_3	$\rightarrow x^{3-1}$	$=x^2$	mod	$x^8+x^4+x^3+x+1$	$=x^2$	$\rightarrow 00000100$	$\rightarrow 04_{16}$
RC_4	$\rightarrow x^{4-1}$	$=x^3$	mod	$x^8+x^4+x^3+x+1$	$=x^3$	$\rightarrow 00001000$	$\rightarrow 08_{16}$
RC_5	$\rightarrow x^{5-1}$	$=x^4$	mod	$x^8+x^4+x^3+x+1$	$=x^4$	$\rightarrow 00010000$	$\rightarrow 10_{16}$
RC_6	$\rightarrow x^{6-1}$	$=x^5$	mod	$x^8+x^4+x^3+x+1$	$=x^5$	$\rightarrow 00100000$	$\rightarrow 20_{16}$
RC_7	$\rightarrow x^{7-1}$	$=x^6$	mod	$x^8+x^4+x^3+x+1$	$=x^6$	$\rightarrow 01000000$	$\rightarrow 40_{16}$
RC_8	$\rightarrow x^{8-1}$	$=x^7$	mod	$x^8+x^4+x^3+x+1$	$=x^7$	$\rightarrow 10000000$	$\rightarrow 80_{16}$
RC_9	$\rightarrow x^{9-1}$	$=x^8$	mod	$x^8+x^4+x^3+x+1$	$=x^4+x^3+x+1$	$\rightarrow 00011011$	$\rightarrow 1B_{16}$
RC_{10}	$\rightarrow x^{10-1}$	$=x^9$	mod	$x^8+x^4+x^3+x+1$	$=x^5+x^4+x^2+x$	$\rightarrow 00110110$	$\rightarrow 36_{16}$

Örnek 5.7: AES-128 blok şifresi için gizli anahtar;
Hexadecimal (01 23 45 67 89 AB CD EF ED CB A9 98 76 54 32 10) olsun. Diğer döngüler için 10 alt anahtarın elde edilmesi aşağıdaki gibi gösterilebilir:

Döngü	t Değerleri	Döngüdeki Birinci Kelime	Döngüdeki İkinci Kelime	Döngüdeki Üçüncü Kelime	Döngüdeki Dördüncü Kelime
-		w_{00} = 01234567	w_{01} = 89ABCDEF	w_{02} = EDCBA998	w_{03} = 76543210
1	2123CA38	w_{04} = 20008F5F	w_{05} = A9AB42B0	w_{06} = 4460EB28	w_{07} = 3234D938
2	1A350723	w_{08} = 3A35887C	w_{09} = 939ECACC	w_{10} = D7FE21E4	w_{11} = E5CAF8DC
3	704186D9	w_{12} = 4A740EA5	w_{13} = D9EAC469	w_{14} = 0E14E58D	w_{15} = EBDE1D51
4	4A740EA5	w_{16} = 5FD0DF4C	w_{17} = 863A1B25	w_{18} = 882EFEA8	w_{19} = 63F0E3F9
5	9C1199FB	w_{20} = C3C146B7	w_{21} = 45FB5D92	w_{22} = CDD5A33A	w_{23} = AE2540C3
6	1F092EE4	w_{24} = DCC86853	w_{25} = 993335C1	w_{26} = 54E696FB	w_{27} = FAC3D638
7	6EF6072D	w_{28} = B23E6F7E	w_{29} = 2B0D5ABF	w_{30} = 7FEBCC44	w_{31} = 85281A7C
8	B4A21097	w_{32} = 069C7FE9	w_{33} = 2D912556	w_{34} = 527AE912	w_{35} = D752F36E
9	1B0D9F0E	w_{36} = 1D91E0E7	w_{37} = 3000C5B1	w_{38} = 627A2CA3	w_{39} = B528DFCD
10	029EBDD5	w_{40} = 1F0F5D32	w_{41} = 2F0F9883	w_{42} = 4D75B420	w_{43} = F85D6BED

Yukarıda görüldüğü gibi her döngüde son üç kelimenin hesabı oldukça basittir. İlk kelimenin hesabı için ilk önce geçici *t* değişkeninin hesaplanması gerekmektedir. Örneğin ilk geçici *t* (döngü 1 için) aşağıdaki gibi hesaplanabilir;

RotWord (76543210) = 54321076
SubWord (54321076) = 2023CA38
t = 2023CA38 $\oplus$ $RCon_1$ = 2023CA38 $\oplus$ 01000000 = 2123CA38

AES şifresinde her döngü anahtarı bir önceki döngü anahtarına bağlıdır. Bununla beraber bu bağımlılık kelime yer değiştirme dönüşümünden dolayı doğrusal değildir. Döngü sabitlerinin eklenmesi de her döngü anahtarının bir öncekinden farklı olmasını sağlar.

6. AKIŞ ŞİFRELER İÇİN NIST TEST PROGRAMI UYGULAMASI

Uygulamada rassal sayıların istatistiksel analizi için pek çok farklı strateji kullanılır. NIST test paketi sayıların rassallığının istatistiksel analizi için beş aşama belirlemiştir. Bunlar:

1. Sayı üretecinin seçilmesi
2. İkilik (binary) dizinin üretilmesi
3. Test paketinin uygulanması
4. P-değerlerinin incelenmesi
5. Değerlendirme. Başarılı / Başarısız kararının verilmesi

şeklindedir.

Her istatistik testi için testin uygulandığı diziye karşılık gelen p-değerleri üretilir. Sabit bir anlamlılık düzeyi için p-değerlerinin belli bir yüzde değeri başarısızlığı gösterir. Örneğin anlamlılık düzeyi $\alpha = 0.01$ olarak seçilirse teste tabi tutulan dizinin %1 başarısız olması beklenir. Teste tabi tutulan dizinin istatistik testinden geçmesi P-değeri $\geq \alpha$ şartına bağlıdır. Aksi durumda testi geçemez.

P-değerleri yorumlanırken üç farklı değerlendirme yapılabilir. Bunlar:

1. P-değerlerinin analizi rassallıktan sapma göstermemiştir.
2. P-değerlerinin analizi rassallıktan sapma göstermiştir.
3. Analizler yetersizdir.

şeklindedir (National Institute of Standard and Technology, 2001).

Tezimizde bazı akış şifreleme algoritmaları kullanılarak üretilen değerlerin rassallığı için NIST test paketi kullanılmıştır. Seçilen akış şifrelerinden Grain ve Mickey donanım tabanlı, Sosemanuk ise yazılım tabanlı akış şifrelerindendir. Grain ve Mickey akış şifrelerinin 128 bit olan versiyonları tercih edilmiştir. NIST'in belirlediği beş aşama dikkate alınarak işlemler gerçekleştirilmiştir. Seçilen akış şifresinin çıktıları ikilik forma dönüştürülerek test programının veri dosyası oluşturulmuştur. Başarı değeri $\alpha = 0.01$ olarak alınmıştır. Test paketi uygulandıktan sonra elde edilen p-değerleri değerlendirilerek grafiksel olarak ifade edilmiştir. NIST test paketine uygulanan program çıktıları testlerden 8 tanesi için yeterli kriterleri sağlayamadığından p-değerleri elde edilememiştir.

6.1. GRAIN-128 Akış Şifresi

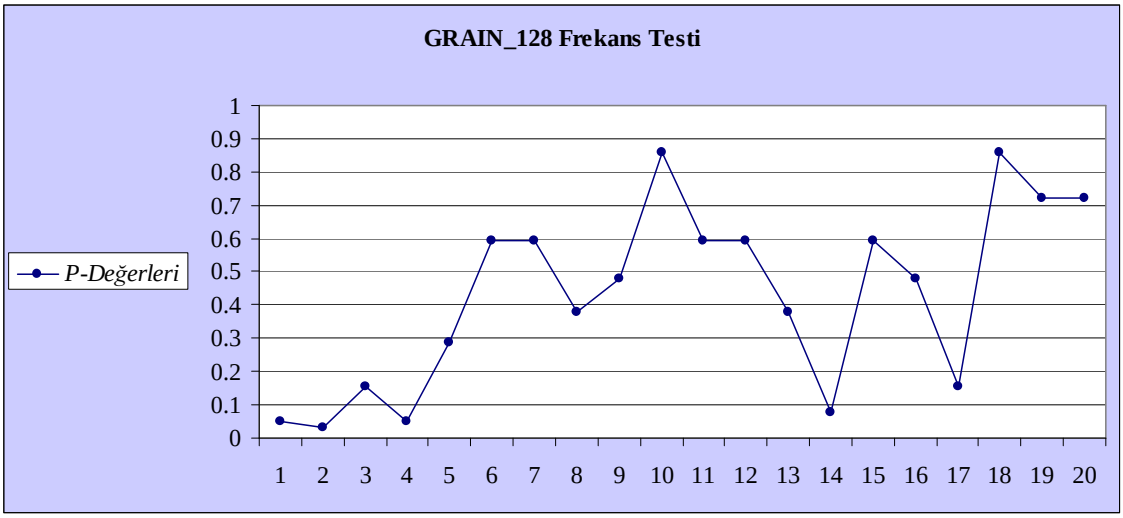
Grain-128 akış şifresi için 20 anahtar değeri girilerek algoritmaya bağlı olarak çıktı üretmesi sağlanmıştır. Başlangıç vektörü (Initialization Vector-IV) her anahtar için sabit değer alınmıştır. Elde edilen program çıktısı Ek E'de verilmiştir. İlgili anahtar değerleri Tablo 6.1'de gösterilmektedir.

Tablo 6.1. Grain-128 Akış Şifresi İçin Girilen Anahtar Değerleri

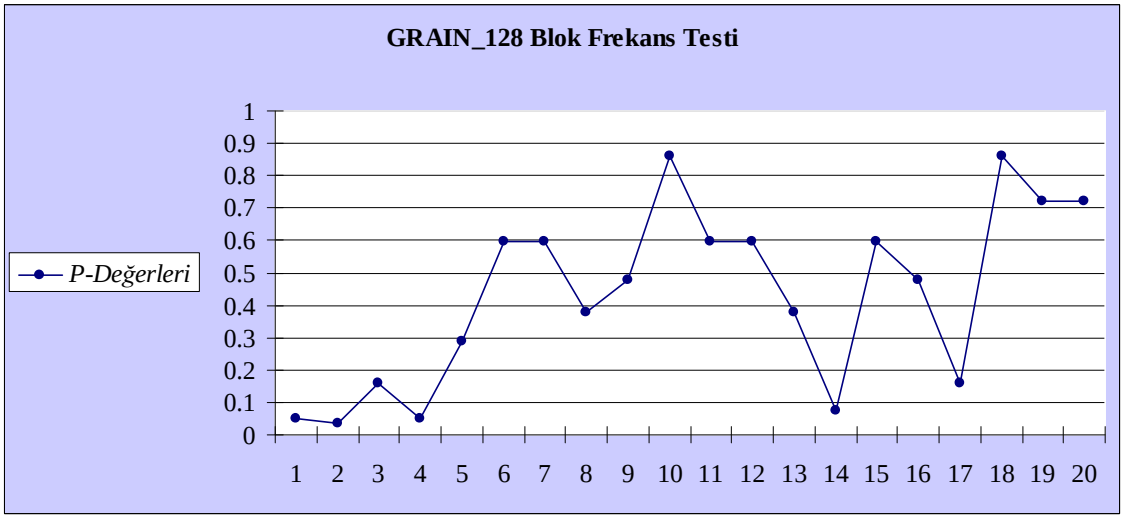
IV	01 23 45 67 89 AB CD EF 12 34 56 78
Anahtar 1	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Anahtar 2	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01
Anahtar 3	0D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 11
Anahtar 4	A0 00 00 B0 00 10 00 00 00 00 00 00 00 00 00 00
Anahtar 5	12 10 CD 0F 70 00 00 00 00 00 00 00 00 00 00 11
Anahtar 6	67 00 A0 F0 00 00 40 00 00 00 00 00 00 00 00 00
Anahtar 7	12 34 56 78 9A 00 00 00 00 00 00 00 00 00 00 00
Anahtar 8	11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11
Anahtar 9	00 00 00 00 00 00 00 00 01 23 45 67 89 AB CD EF
Anahtar 10	1A 2B 3C 4D 5E 6F 11 11 11 11 11 11 11 11 11 11
Anahtar 11	F0 E0 D0 C0 B0 A0 10 20 30 40 50 60 70 80 90 00
Anahtar 12	0F 0E 0D 0C 0B 0A 01 02 03 04 05 05 07 08 09 00
Anahtar 13	CD E0 D0 C0 B0 AA 10 2F 30 40 50 6A 70 80 90 8F
Anahtar 14	1F 0E 0D 7C 0B 3A 01 A2 D3 04 05 05 07 08 09 1B
Anahtar 15	38 E0 D0 C2 B0 4A 10 2F 00 40 50 6A 70 80 90 8F
Anahtar 16	18 0E FD 7C 0B FF 01 D5 D3 33 05 05 07 08 09 1B
Anahtar 17	89 00 12 00 B1 AA 10 2F 00 00 50 6A 70 80 90 8F
Anahtar 18	00 00 00 00 00 FF 00 D0 D0 00 00 00 00 00 00 00
Anahtar 19	AB CD EF 11 23 AA 99 2F 00 00 50 6A 70 80 90 00
Anahtar 20	A4 F7 00 22 00 FF 77 D0 D0 00 00 00 00 00 BB F9

Program çıktısının NIST test programına tabi tutulması sonucu elde edilen P-değerleri ve grafikleri aşağıda gösterilmektedir.

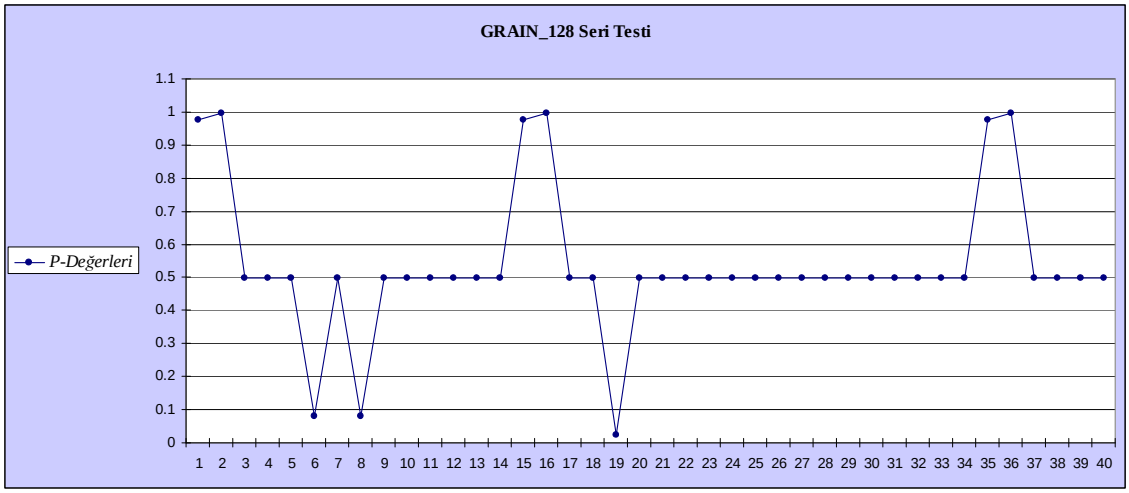
P-Değerleri	0.05183	0.288844	0.4795	0.376759	0.157299
	0.033895	0.595883	0.859684	0.0771	0.859684
	0.157299	0.595883	0.595883	0.595883	0.723674
	0.05183	0.376759	0.595883	0.4795	0.723674



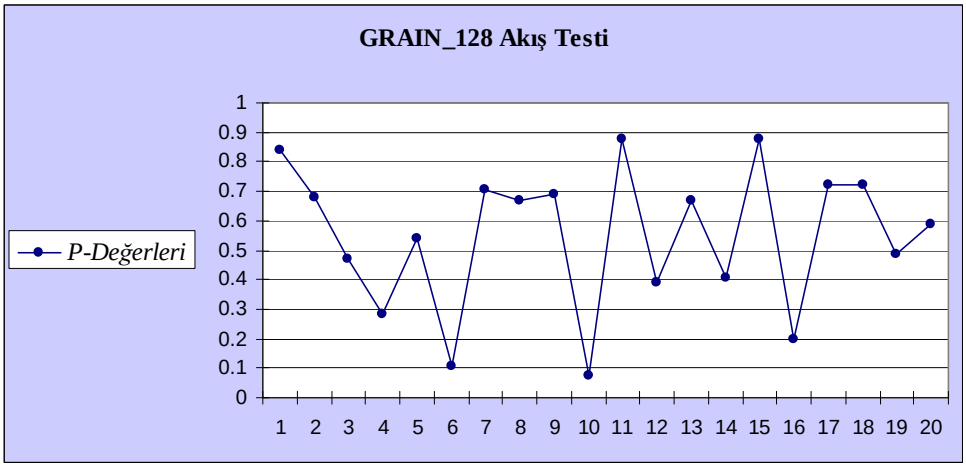
P-Değerleri	0.05183	0.288844	0.4795	0.376759	0.157299
	0.033895	0.595883	0.859684	0.0771	0.859684
	0.157299	0.595883	0.595883	0.595883	0.723674
	0.05183	0.376759	0.595883	0.4795	0.723674



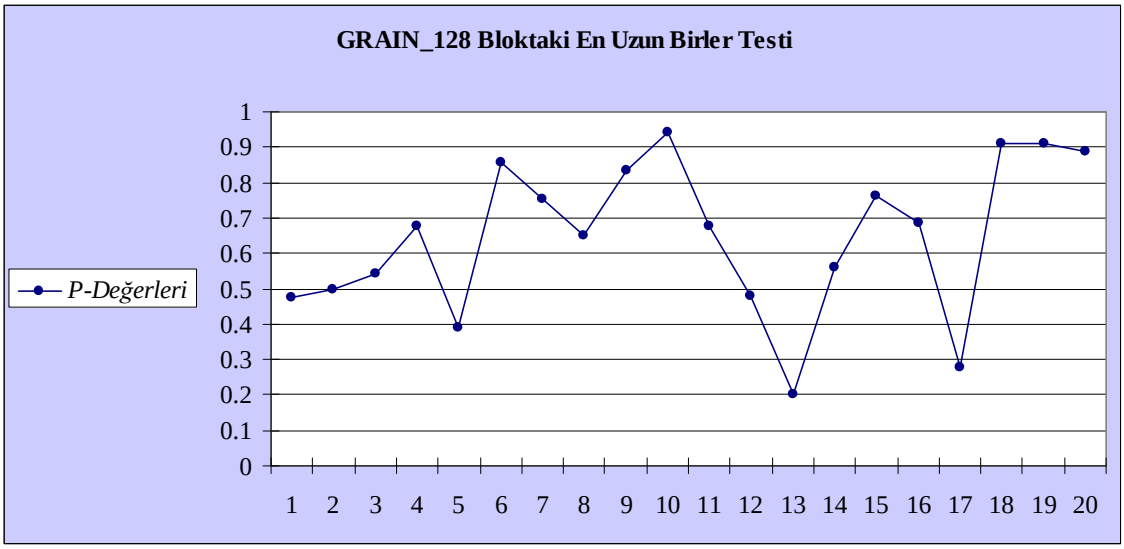
P-Değerleri	0.977673	0.498961	0.498961	0.498961	0.498961
	0.997846	0.498531	0.498531	0.498531	0.498531
	0.498961	0.498961	0.02317	0.498961	0.977673
	0.498531	0.498531	0.498531	0.498531	0.997846
	0.498961	0.498961	0.498961	0.498961	0.498961
	0.079182	0.498531	0.498531	0.498531	0.498531
	0.498961	0.977673	0.498961	0.498961	0.498961
	0.079182	0.997846	0.498531	0.498531	0.498531



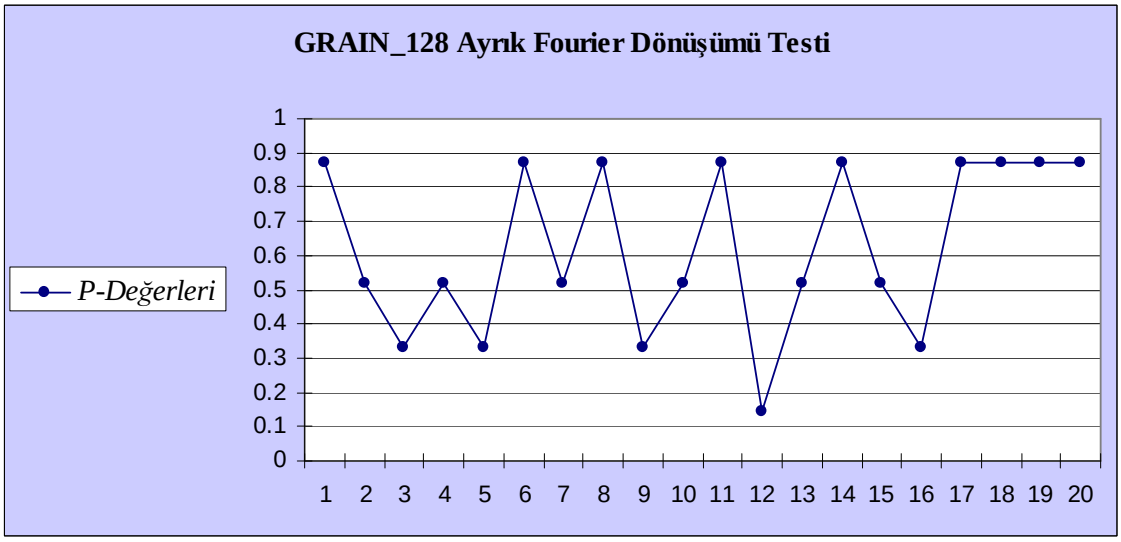
P-Değerleri	0.839853	0.539838	0.689667	0.670689	0.719471
	0.680163	0.10536	0.076567	0.408392	0.721539
	0.472553	0.704505	0.878988	0.878988	0.485966
	0.283261	0.670689	0.389284	0.198214	0.587882



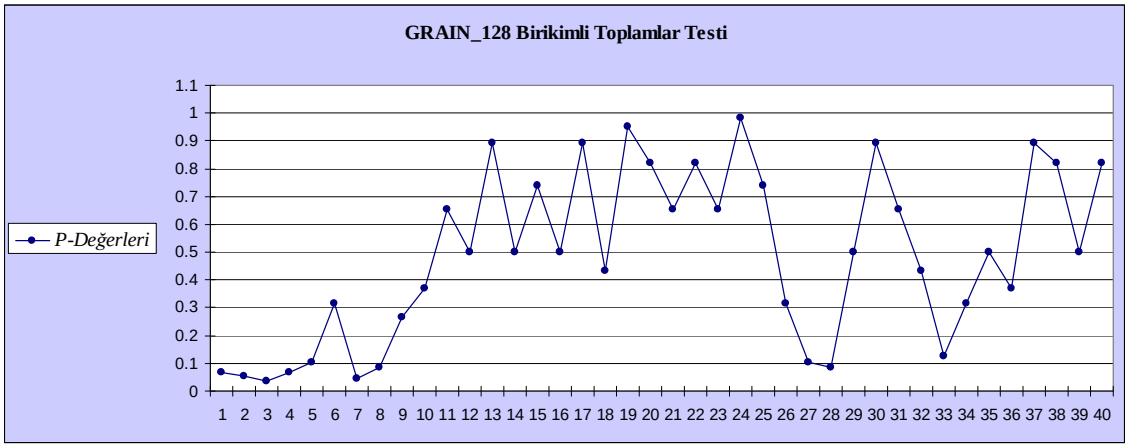
P-Değerleri	0.474919	0.39162	0.835349	0.201332	0.277026
	0.496028	0.856871	0.94039	0.558963	0.908716
	0.541472	0.752917	0.674916	0.763188	0.908716
	0.674916	0.648665	0.480009	0.685706	0.889318



P-Değerleri	0.871131	0.33039	0.33039	0.516412	0.871131
	0.516412	0.871131	0.516412	0.871131	0.871131
	0.33039	0.516412	0.871131	0.516412	0.871131
	0.516412	0.871131	0.144292	0.33039	0.871131



P-Değerleri	0.06779	0.26587	0.892023	0.737518	0.126863
	0.054251	0.369656	0.431439	0.314554	0.314554
	0.034021	0.654761	0.949266	0.10366	0.499939
	0.06779	0.499939	0.81877	0.084119	0.369656
	0.10366	0.892023	0.654761	0.499939	0.892023
	0.314554	0.499939	0.81877	0.892023	0.81877
	0.043113	0.737518	0.654761	0.654761	0.499939
	0.084119	0.499939	0.984155	0.431439	0.81877



6.2. MICKEY-128 Akış Şifresi

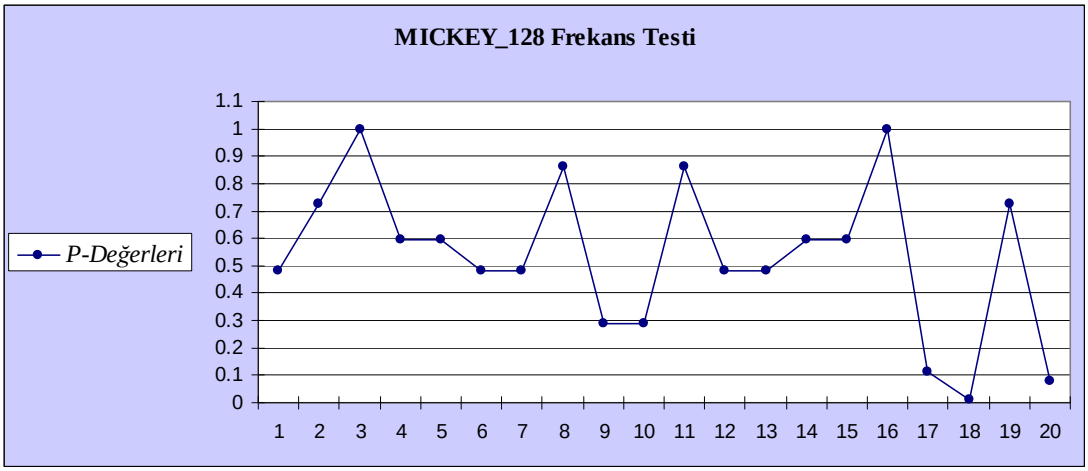
Mickey-128 akış şifresi için Grain-128 şifresi için girilen 20 anahtar değeri kullanılmış ve algoritmaya bağlı olarak çıktı üretmesi sağlanmıştır. Başlangıç vektörü (Initialization Vector-IV) her anahtar için sabit değerde alınmıştır. Şifrenin algoritmik yapısı gereği IV Grain-128 şifresinden farklı değer almaktadır. Elde edilen program çıktısı Ek F'de verilmiştir. İlgili anahtar değerleri Tablo 6.2'de gösterilmektedir.

Tablo 6.2. Mickey-128 Akış Şifresi İçin Girilen Anahtar Değerleri

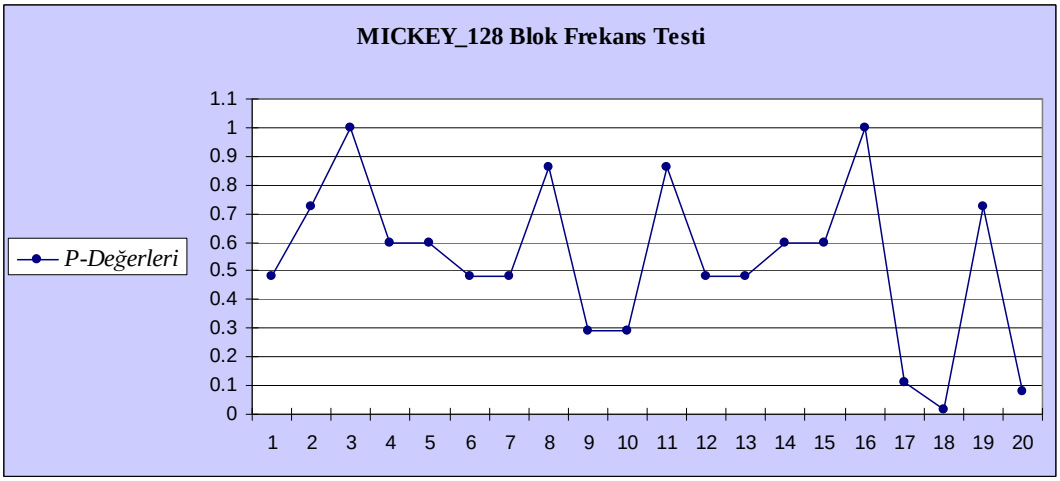
IV	01 23 45 67 89 ab cd ef 12 34 56 78 9a bc de f0
Anahtar 1	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Anahtar 2	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01
Anahtar 3	0D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 11
Anahtar 4	A0 00 00 B0 00 10 00 00 00 00 00 00 00 00 00 00
Anahtar 5	12 10 CD 0F 70 00 00 00 00 00 00 00 00 00 00 11
Anahtar 6	67 00 A0 F0 00 00 40 00 00 00 00 00 00 00 00 00
Anahtar 7	12 34 56 78 9A 00 00 00 00 00 00 00 00 00 00 00
Anahtar 8	11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11
Anahtar 9	00 00 00 00 00 00 00 00 01 23 45 67 89 AB CD EF
Anahtar 10	1A 2B 3C 4D 5E 6F 11 11 11 11 11 11 11 11 11 11
Anahtar 11	F0 E0 D0 C0 B0 A0 10 20 30 40 50 60 70 80 90 00
Anahtar 12	0F 0E 0D 0C 0B 0A 01 02 03 04 05 05 07 08 09 00
Anahtar 13	CD E0 D0 C0 B0 AA 10 2F 30 40 50 6A 70 80 90 8F
Anahtar 14	1F 0E 0D 7C 0B 3A 01 A2 D3 04 05 05 07 08 09 1B
Anahtar 15	38 E0 D0 C2 B0 4A 10 2F 00 40 50 6A 70 80 90 8F
Anahtar 16	18 0E FD 7C 0B FF 01 D5 D3 33 05 05 07 08 09 1B
Anahtar 17	89 00 12 00 B1 AA 10 2F 00 00 50 6A 70 80 90 8F
Anahtar 18	00 00 00 00 00 FF 00 D0 D0 00 00 00 00 00 00 00
Anahtar 19	AB CD EF 11 23 AA 99 2F 00 00 50 6A 70 80 90 00
Anahtar 20	A4 F7 00 22 00 FF 77 D0 D0 00 00 00 00 00 BB F9

Program çıktısının NIST test programına tabi tutulması sonucu elde edilen P-değerleri ve grafikleri aşağıda gösterilmektedir.

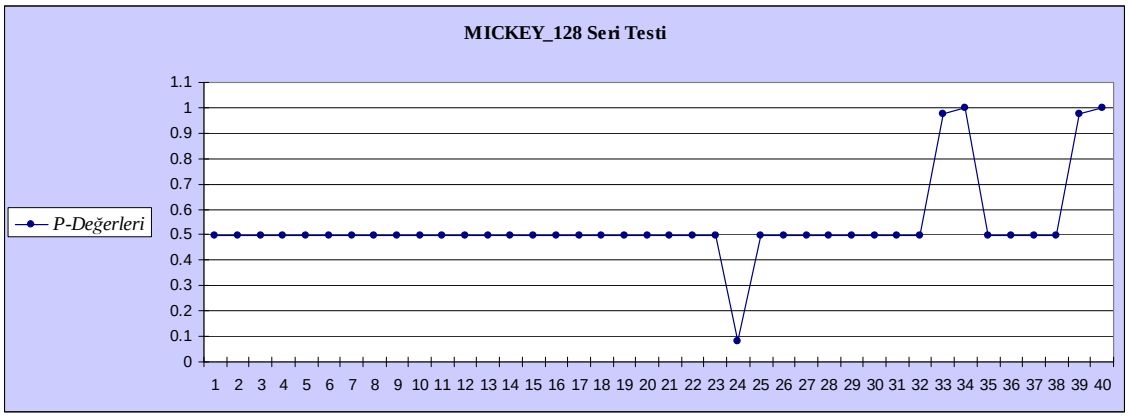
P-Değerleri	0.4795	0.595883	0.288844	0.4795	0.111612
	0.723674	0.4795	0.288844	0.595883	0.013328
	1	0.4795	0.859684	0.595883	0.723674
	0.595883	0.859684	0.4795	1	0.0771



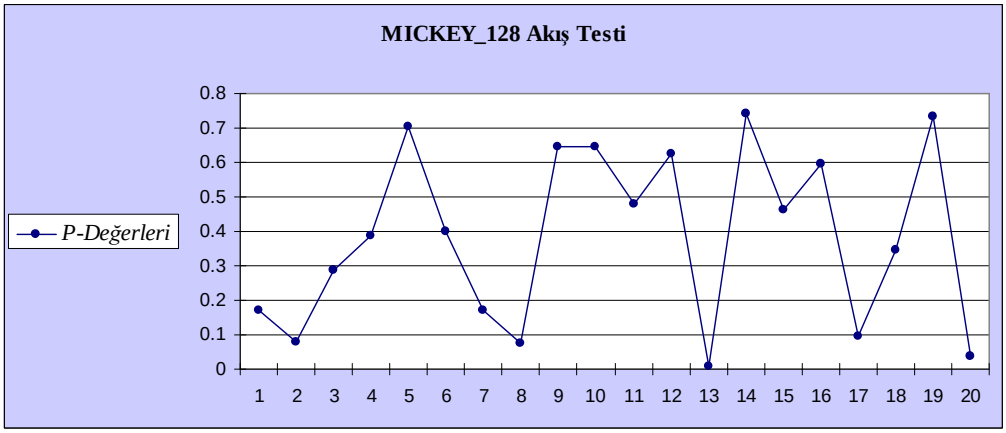
P-Değerleri	0.4795	0.595883	0.288844	0.4795	0.111612
	0.723674	0.4795	0.288844	0.595883	0.013328
	1	0.4795	0.859684	0.595883	0.723674
	0.595883	0.859684	0.4795	1	0.0771



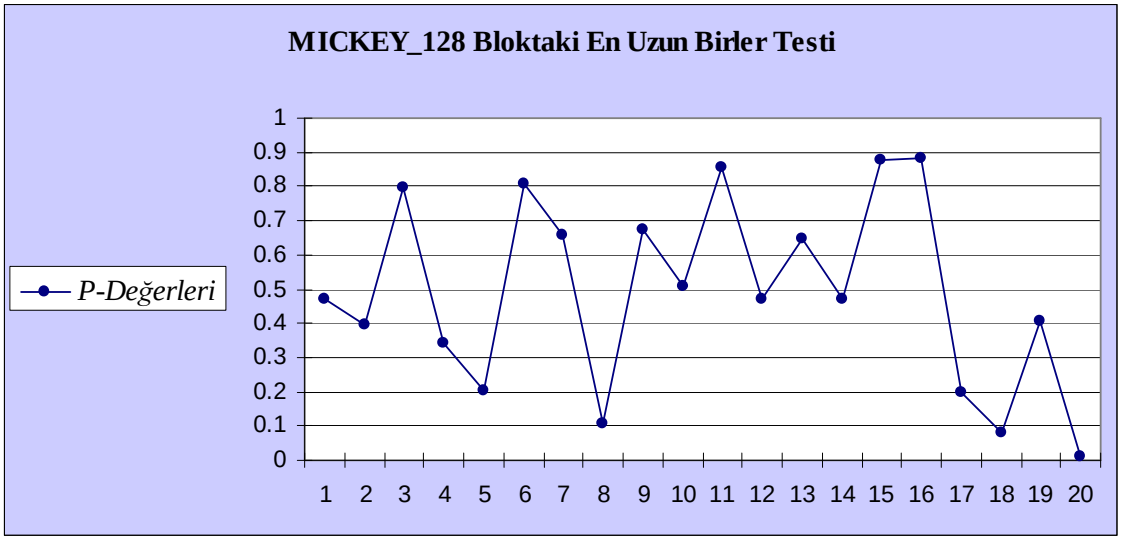
P-Değerleri	0.498961	0.498961	0.498961	0.498961	0.977673
	0.498531	0.498531	0.498531	0.498531	0.997846
	0.498961	0.498961	0.498961	0.498961	0.498961
	0.498531	0.498531	0.498531	0.498531	0.498531
	0.498961	0.498961	0.498961	0.498961	0.498961
	0.498531	0.498531	0.498531	0.498531	0.498531
	0.498961	0.498961	0.498961	0.498961	0.977673
	0.498531	0.498531	0.079182	0.498531	0.997846



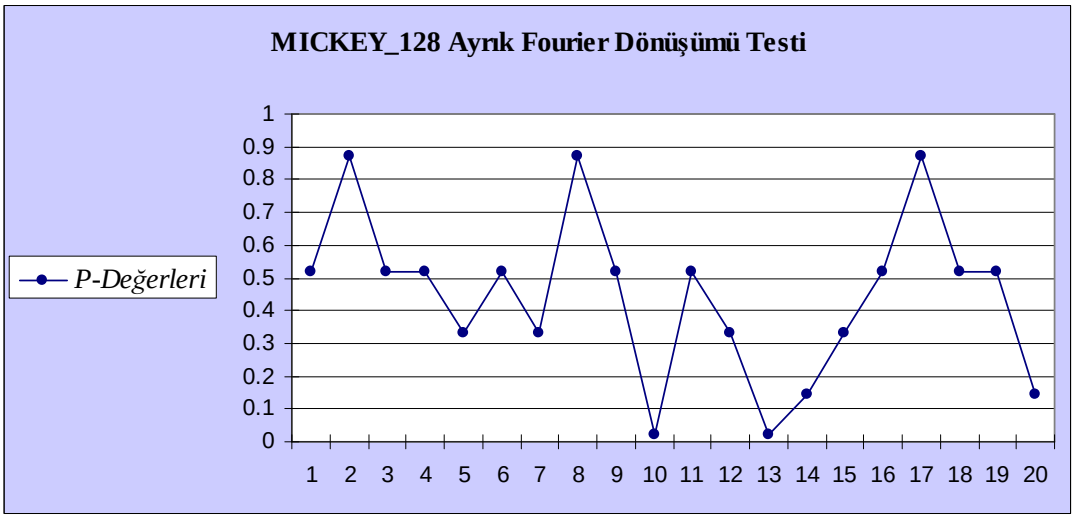
P-Değerleri	0.16901	0.704505	0.647666	0.008853	0.094723
	0.078673	0.399238	0.647666	0.74184	0.347265
	0.288844	0.16901	0.477678	0.463206	0.731719
	0.389284	0.076567	0.62552	0.595883	0.038221



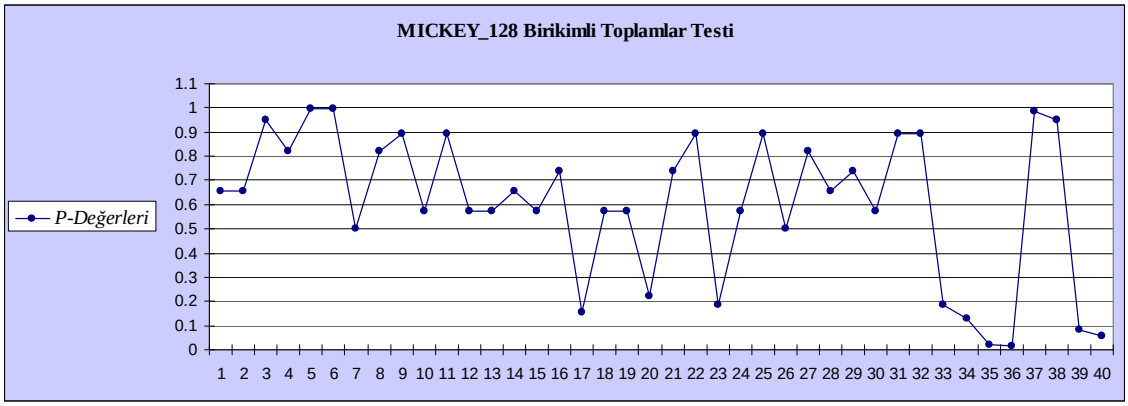
P-Değerleri	0.469614	0.203286	0.674916	0.648665	0.198306
	0.398062	0.810056	0.508286	0.471057	0.079934
	0.795721	0.655529	0.856871	0.87699	0.403786
	0.341617	0.106813	0.471057	0.884161	0.008135



P-Değerleri	0.516412	0.33039	0.516412	0.023141	0.871131
	0.871131	0.516412	0.023141	0.144292	0.516412
	0.516412	0.33039	0.516412	0.33039	0.516412
	0.516412	0.871131	0.33039	0.516412	0.144292



P-Değerleri	0.654761	0.892023	0.1542	0.892023	0.186156
	0.654761	0.574764	0.574764	0.499939	0.126863
	0.949266	0.892023	0.574764	0.81877	0.020739
	0.81877	0.574764	0.22322	0.654761	0.01602
	0.9977	0.574764	0.737518	0.737518	0.984155
	0.9977	0.654761	0.892023	0.574764	0.949266
	0.499939	0.574764	0.186156	0.892023	0.084119
	0.81877	0.737518	0.574764	0.892023	0.054251



6.3. SOSEMANUK Akış Şifresi

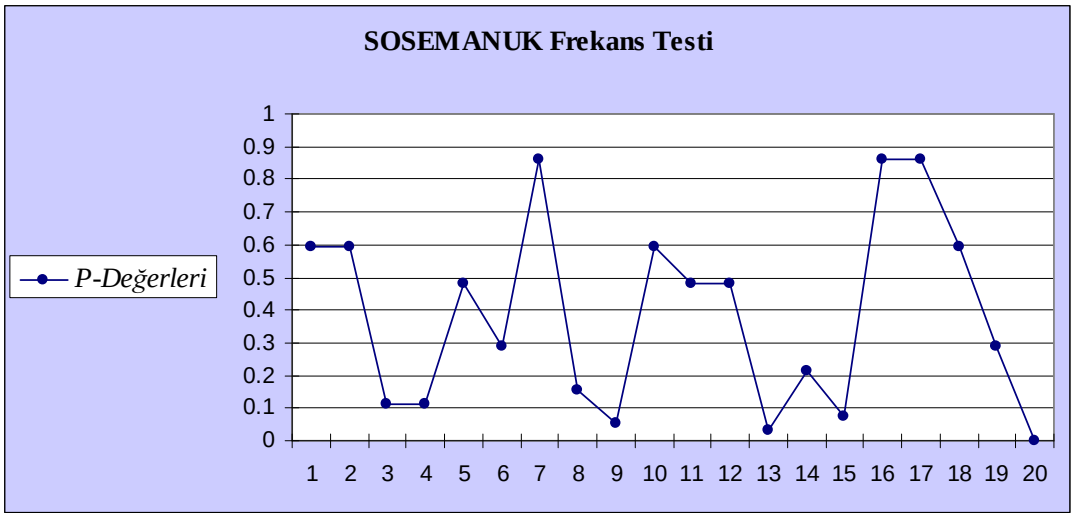
Sosemanuk akış şifresi için 20 anahtar değeri girilerek algoritmaya bağlı olarak çıktı üretmesi sağlanmıştır. Başlangıç vektörü (Initialization Vector-IV) her anahtar için sabit değer alınmıştır. Elde edilen program çıktısı Ek G'de verilmiştir. İlgili anahtar değerleri Tablo 6.3'de gösterilmektedir.

Tablo 6.3. Sosemanuk Akış Şifresi İçin Girilen Anahtar Değerleri

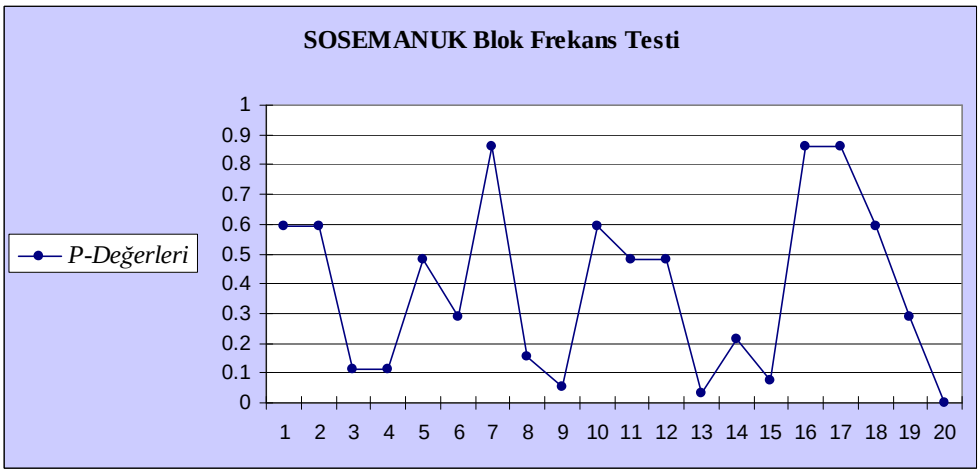
IV	00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF
Anahtar 1	A7 FD D3 33 DC
Anahtar 2	62 A9 A0 C2 BA
Anahtar 3	FE 21 73 40 31
Anahtar 4	D4 A1 32 FA 74
Anahtar 5	EF 33 2F 1B 7F
Anahtar 6	E4 33 2F 1B 8E
Anahtar 7	11 E7 88 3E C6
Anahtar 8	BA 02 4C 8E 9E
Anahtar 9	0D B3 63 EC AA
Anahtar 10	22 68 11 A3 B0
Anahtar 11	CA E1 27 05 CE
Anahtar 12	21 56 6A 0E E7
Anahtar 13	B3 62 B2 EC DD
Anahtar 14	C6 8D 79 D7 F4
Anahtar 15	01 F0 BD 80 CE
Anahtar 16	E2 E4 AD 48 E6
Anahtar 17	56 F2 92 F3 11
Anahtar 18	A1 A8 E4 A0 EC
Anahtar 19	68 F1 38 C0 C2
Anahtar 20	F8 C2 B2 AC FE

Program çıktısının NIST test programına tabi tutulması sonucu elde edilen P-değerleri ve grafikleri aşağıda gösterilmektedir.

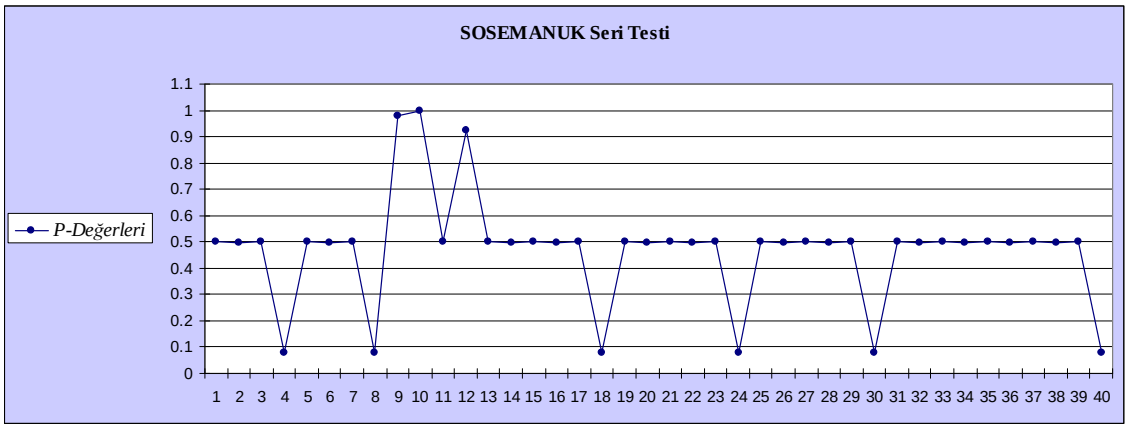
P-Değerleri	0.595883	0.4795	0.05183	0.033895	0.859684
	0.595883	0.288844	0.595883	0.215925	0.595883
	0.111612	0.859684	0.4795	0.0771	0.288844
	0.111612	0.157299	0.4795	0.859684	0.001463



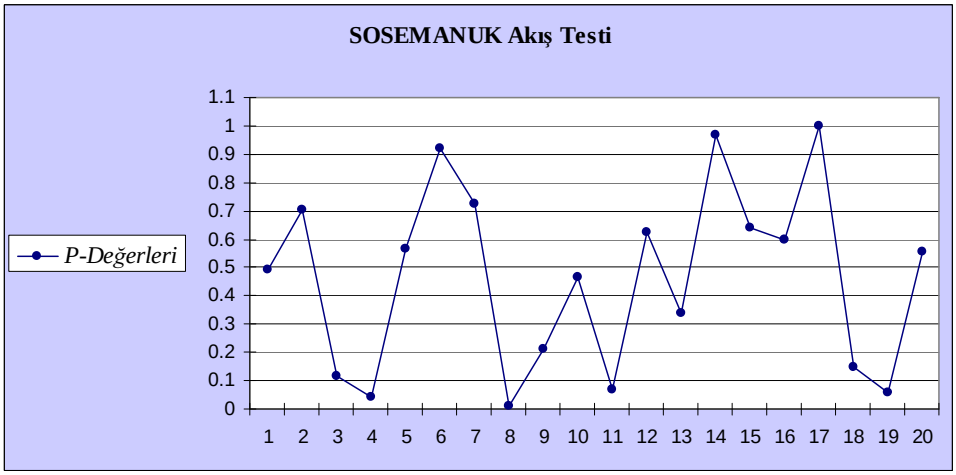
P-Değerleri	0.595883	0.4795	0.05183	0.033895	0.859684
	0.595883	0.288844	0.595883	0.215925	0.595883
	0.111612	0.859684	0.4795	0.0771	0.288844
	0.111612	0.157299	0.4795	0.859684	0.001463



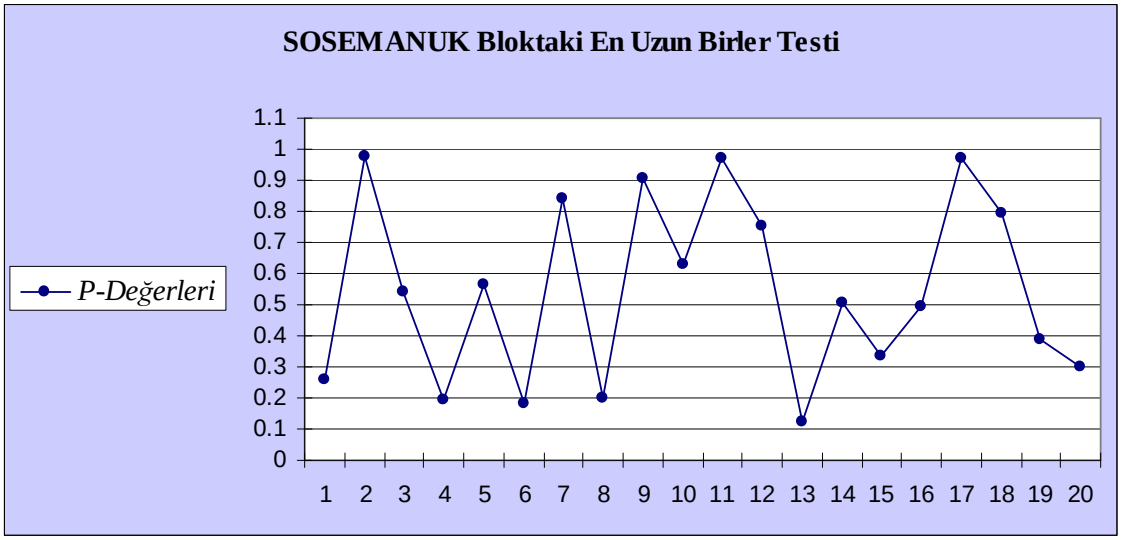
P-Değerleri	0.498961	0.977673	0.498961	0.498961	0.498961
	0.498531	0.997846	0.079182	0.498531	0.498531
	0.498961	0.498961	0.498961	0.498961	0.498961
	0.079182	0.921899	0.498531	0.498531	0.498531
	0.498961	0.498961	0.498961	0.498961	0.498961
	0.498531	0.498531	0.498531	0.079182	0.498531
	0.498961	0.498961	0.498961	0.498961	0.498961
	0.079182	0.498531	0.079182	0.498531	0.079182



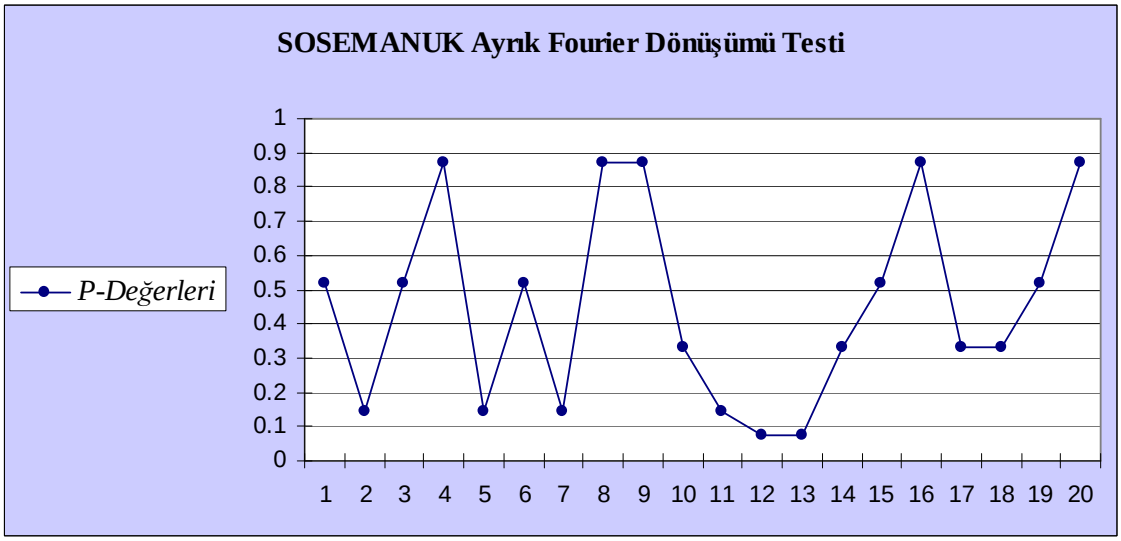
P-Değerleri	0.494133	0.56409	0.209413	0.336103	0.997796
	0.704505	0.920091	0.463206	0.966552	0.149234
	0.115214	0.725681	0.068901	0.642414	0.059597
	0.042186	0.011932	0.62552	0.59771	0.556612



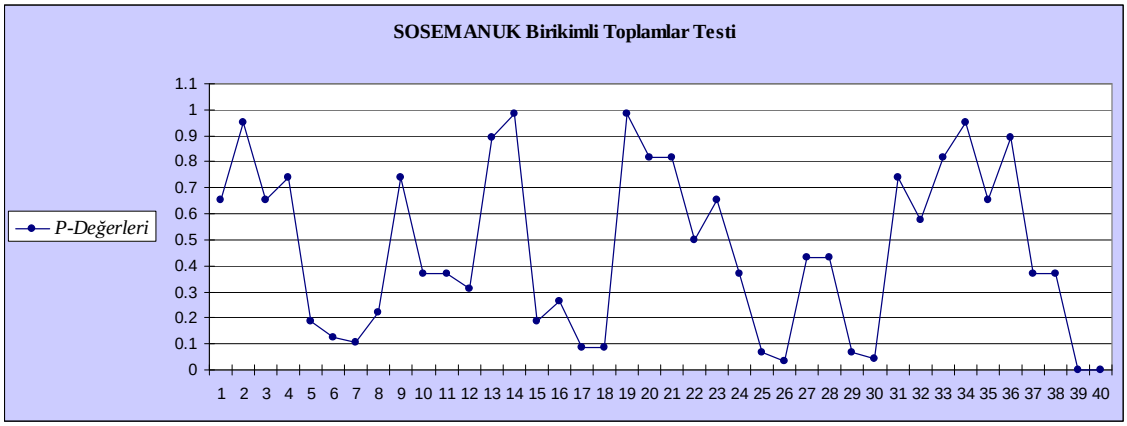
P-Değerleri	0.257132	0.56595	0.908716	0.121742	0.969343
	0.973813	0.180609	0.63038	0.508286	0.795721
	0.541472	0.839146	0.969343	0.337421	0.388787
	0.19379	0.202941	0.752936	0.495995	0.300691



P-Değerleri	0.516412	0.144292	0.871131	0.074353	0.33039
	0.144292	0.516412	0.33039	0.33039	0.33039
	0.516412	0.144292	0.144292	0.516412	0.516412
	0.871131	0.871131	0.074353	0.871131	0.871131

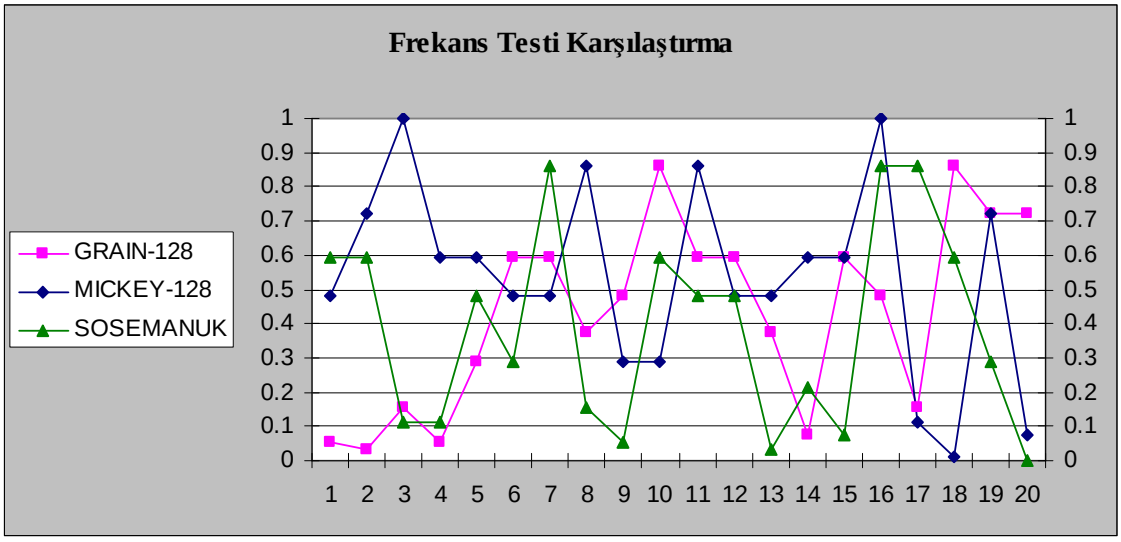


P-Değerleri	0.654761	0.737518	0.084119	0.06779	0.81877
	0.949266	0.369656	0.084119	0.034021	0.949266
	0.654761	0.369656	0.984155	0.431439	0.654761
	0.737518	0.314554	0.81877	0.431439	0.892023
	0.186156	0.892023	0.81877	0.06779	0.369656
	0.126863	0.984155	0.499939	0.043113	0.369656
	0.10366	0.186156	0.654761	0.737518	0.001133
	0.22322	0.26587	0.369656	0.574764	0.002148

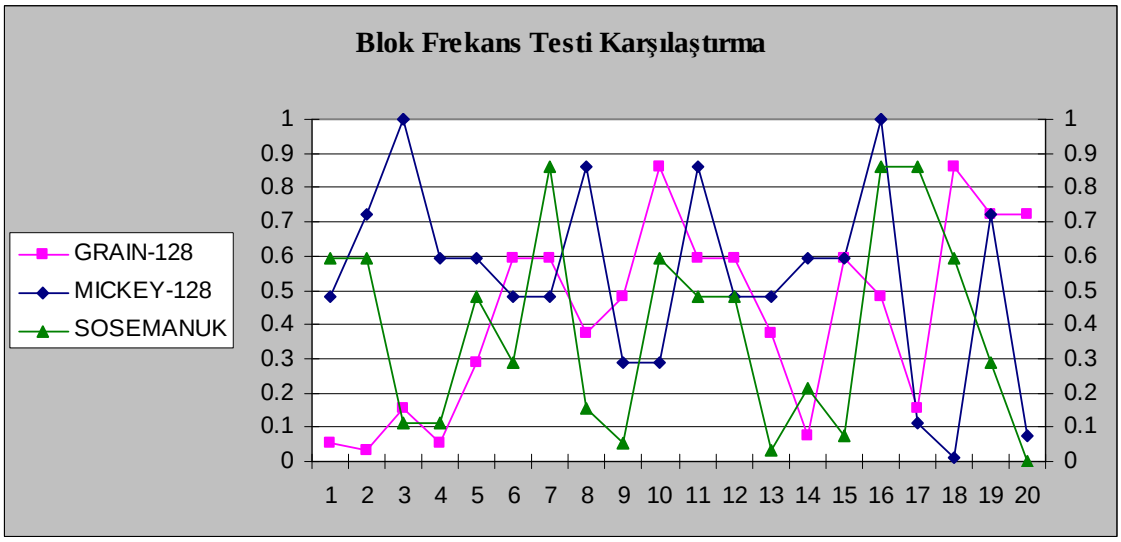


6.4. Test Sonuçlarının Karşılaştırılması

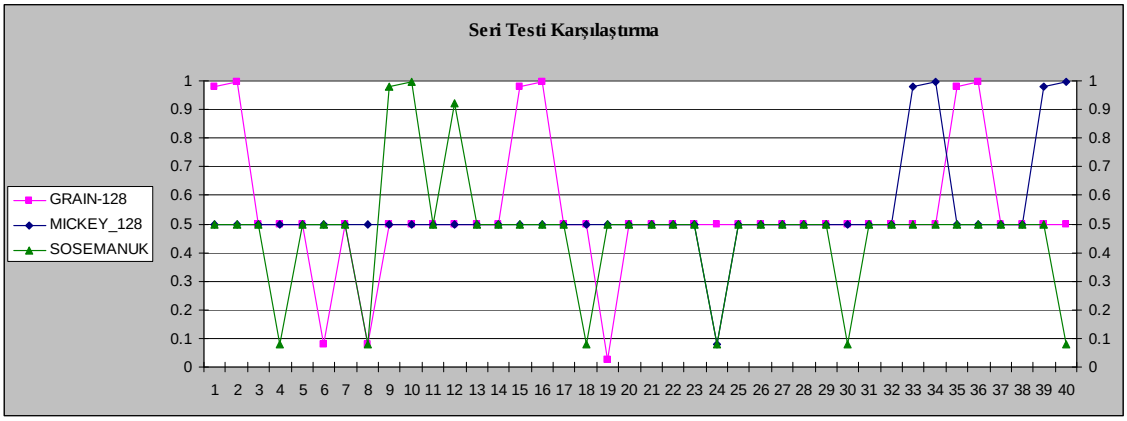
Test sonuçları değerlendirilirken her bir test için uygulanmış olan akış şifreleri çıktılarının teste tabi tutulması sonucu elde edilen p-değerleri dikkate alınmıştır. P-değeri testin başarı oranını değerlendiren kriter olduğundan her bir akış şifresinin ilgili teste göre aldığı p-değeri kıyaslanan üç şifre için ilgili grafiklerde gösterilmiştir.



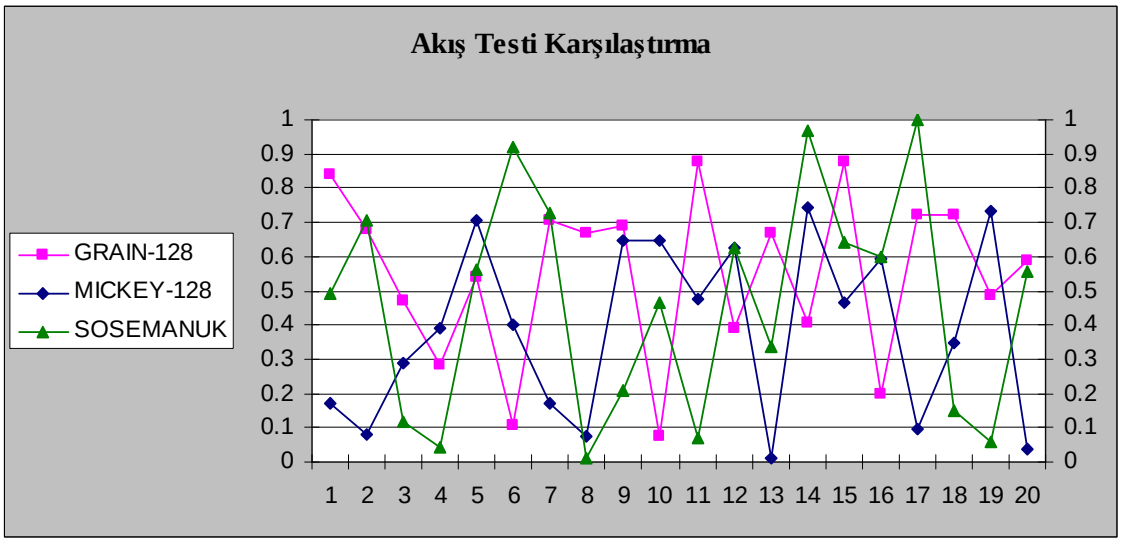
Frekans testi için her üç akış şifresi incelendiğinde MICKEY-128 şifresinin diğer şifrelere göre daha dengeli sonuçlar verdiği gözlenmektedir. Her ne kadar başarısızlık kriteri içerse de çıktı değerlerinin içerdiği bitsel denge diğer şifrelere göre daha kuvvetlidir.



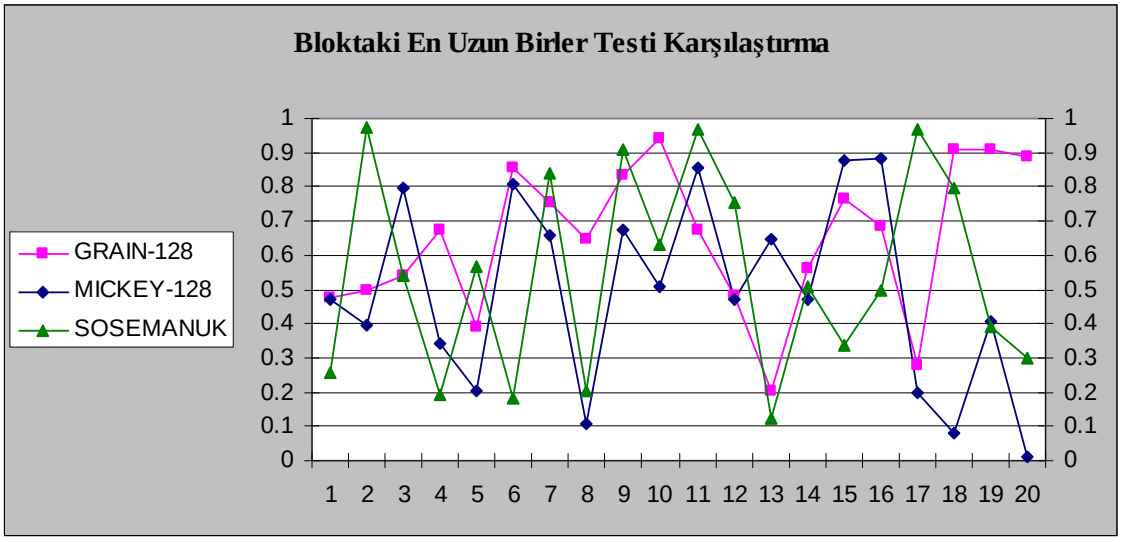
Blok frekans testi frekans testi temelli olduğundan her üç akış şifresi incelendiğinde bu test için de MICKEY-128 şifresinin diğer şifrelere göre daha dengeli sonuçlar verdiği gözlenmektedir. MICKEY-128 şifresinin bit bloklarının dengesi diğer şifrelere göre daha kuvvetli olduğu gözlenmiştir.



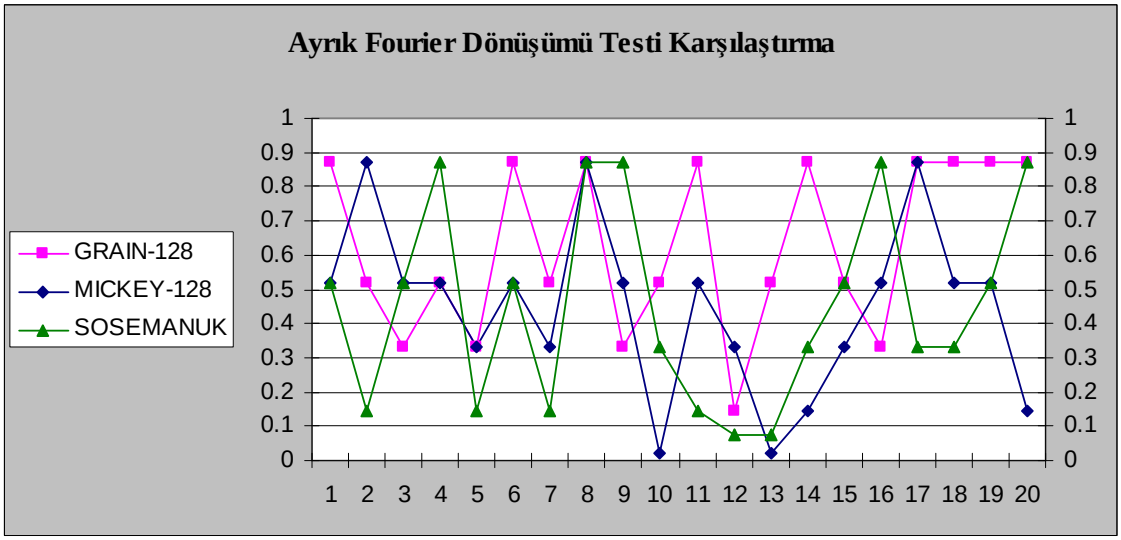
Seri testi tekrar eden bit bloklarının sayı dağılımını incelediğinden her üç akış şifresinin farklı döngü zamanlarında aynı sonuçları verdiği gözlenmiştir. Bu nedenle seri testi için üç şifrenin de aynı başarı oranında olduğu söylenebilir.



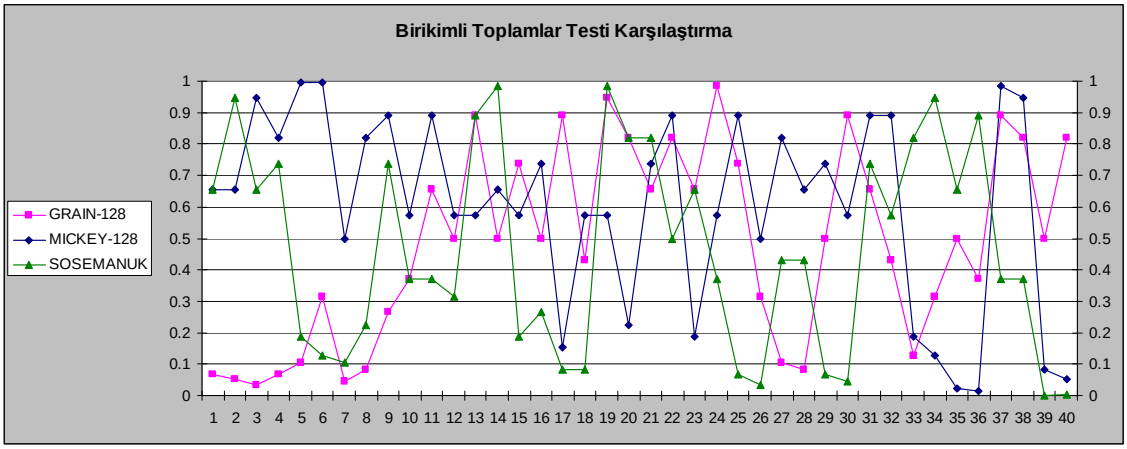
Akış testi için her üç akış şifresi incelendiğinde SOSEMANUK akış şifresinin diğer şifrelere göre daha iyi sonuçlar verdiği gözlenmiştir. Bu testte 0 ve 1'lerden oluşan bit blokları değerlendirildiğinden elde edilen sonuçlara bakıldığında SOSEMANUK akış şifresinin bu dengeyi sağlama oranının diğerlerine göre daha iyi olduğu gözlenmiştir.



Bloktaki en uzun birler testi için her üç akış şifresi incelendiğinde SOSEMANUK akış şifresinin diğer şifrelere göre daha iyi sonuçlar verdiği gözlenmiştir. Bu test dizideki 0 ve 1 bloklarının uzunluklarını incelediğinden SOSEMANUK akış şifresinin testteki başarı oranının diğer akış şifrelere göre daha iyi olduğu gözlenmiştir.



Ayrık Fourier dönüşümü testi için her üç akış şifresi incelendiğinde GRAIN-128 akış şifresinin diğer şifrelere göre daha iyi sonuçlar verdiği gözlenmiştir. Bu test bit dizisinin ayrık Fourier dönüşümünü alıp periyodikliğini incelediğinden GRAIN akış şifresinin testteki başarı oranının diğer akış şifrelere göre daha iyi olduğu gözlenmiştir.



Birikimli toplamlar testi için her üç akış şifresi incelendiğinde MICKEY-128 akış şifresinin diğer şifrelere göre daha iyi sonuçlar verdiği gözlenmiştir. Bu test bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirleyerek bloklar arasındaki dengesizlik farkına baktığından MICKEY akış şifresinin testteki başarı oranının diğer akış şifrelere göre daha iyi olduğu gözlenmiştir.

Tüm değerlendirmeler sonucunda karşılaştırma sonucu başarılı olarak görülen şifreler aşağıdaki tabloda özetlenmektedir.

	GRAIN-128	**MICKEY-128**	**SOSEMANUK**
Frekans Testi		•	
Blok Frekans Testi		•	
Seri Testi	•	•	•
Akış Testi			•
Bloktaki En Uzun Birler Testi			•
Ayrık Fourier Dönüşümü Testi	•		
Birikimli Toplamlar Testi		•	

7. AES ANAHTAR GENİŞLETME ALGORİTMASININ YAYILIM ÖZELLİĞİ İÇİN DENEYSEL SONUÇLAR

5. bölümde belirtildiği gibi AES anahtar genişletme algoritmasının yayılımının kötü özellikler göstermektedir. Bu duruma dair bir örnek Örnek 7.1'de verilmektedir. Örnek 7.1'de AES genişletme algoritması ile üretilen 10 farklı anahtar ile bu gizli anahtardaki 1 bitin değişimi ile üretilen anahtarlar arasındaki yayılım özelliği üretilen anahtarlardaki bit pozisyonlarında meydana gelen farklar ile gösterilmiştir. Bu örnekte hiçbir alt anahtar değerinin katı çığ kriterini verilen bit pozisyonundaki farklılaştırma ile sağlamadığı gösterilmektedir.

Örnek 7.1: Örnek bir gizli anahtarın sadece 1-bit farklılaştığı durum için anahtar genişletme algoritması ile elde edilen döngü anahtarları ve bu iki gizli anahtardan elde edilen döngü anahtarları arasındaki karşılıklı ilişki aşağıdaki gibi verilebilir.

Gizli Anahtarı 1: (**0**1 23 45 67 89 AB CD EF ED CB A9 98 76 54 32 10)
Gizli Anahtarı 2: (**4**1 23 45 67 89 AF CD EF ED CB A9 98 76 54 32 10)

Döngü	Gizli Anahtar 1 için Döngü Anahtarları	Gizli Anahtar 2 için Döngü Anahtarları	Bit Farkı	Bit Farkı % Değişim
-	**0**1234567 89ABCDEF EDCBA998 76543210	**4**1234567 89ABCDEF EDCBA998 76543210	1	0.78
1	**2**0008F5F **A**9AB42B0 **4**460EB28 **3**234D938	**6**0008F5F **E**9AB42B0 **0**460EB28 **7**234D938	4	3.13
2	3A35887C 939ECACC D7FE21E4 E5CAF8DC	7A35881F 939ECAAF 97FE2187 E5CAF8BF	18	14.06
3	4A740EA5 D9EAC469 0E14E58D EBDE1D51	0A7480C6 99EA4A69 0E146BEE EBDE9351	26	20.31
4	5FD0DF4C 863A1B25 882EFEA8 63F0E3F9	1FA8512F 86421B46 885670A8 6388E3F9	33	25.78
5	C3C146B7 45FB5D92 CDD5A33A AE2540C3	CBB9C8D4 4DFBD392 C5ADA33A A62540C3	24	18.75
6	DCC86853 993335C1 54E696FB FAC3D638	D4B0E6F0 994B3562 5CE69658 FAC3D69B	30	23.44
7	B23E6F7E 2B0D5ABF 7FEBCC44 85281A7C	BA46F2DD 230DC7BF 7FEB51E7 8528877C	34	26.56
8	069C7FE9 2D912556 527AE912 D752F36E	0E51E24A 2D5C25F5 52B77412 D79FF36E	39	30.47
9	1D91E0E7 3000C5B1 627A2CA3 B528DFCD	CE5C7D44 E30058B1 B1B72CA3 6628DFCD	44	34.38
10	1F0F5D32 2F0F9883 4D75B420 F85D6BED	CCC2C077 2FC298C6 9E75B465 F85D6BA8	37	28.91

7.1. AES Anahtar Genişletme Algoritması Test Sonuçları

Tezin bu bölümünde AES anahtar genişletme algoritmasına ait katı çığ kriteri testi ile ilgili elde edilen deneysel sonuçlar verilmektedir. Örnek 7.1'de verildiği gibi katı çığ kriteri testi olası bir gizli anahtarın tüm bit pozisyonlarının 1-bit değişimi sonucu elde edilecek alt anahtarların gizli anahtarın alt anahtarları ile bit farklarının ölçülmesi ile gerçekleştirilebilir. Bir gizli anahtar için olası 128 farklı bit pozisyonu için 1 bit değişimi sonucu elde edilecek 1280 alt anahtarın gizli anahtar ile elde edilecek alt anahtarlar ile karşılaştırmaları sonucu bir ortalama değer elde edilebilir. Daha sonra rassal olarak seçilen bir anahtar kümesi üzerinde bu test ile anahtar genişletme algoritmasının karakteristik özelliği çıkarılabilir. Örneğin aşağıda verilen tablodaki değerler EK A'da verilen 20 rassal 128-bit anahtar için (dolayısıyla 20 × 1280 = 25600 alt anahtar) elde edilen sonuçları göstermektedir. Örneğin Tablo 7.1'de Anahtar 1 ile Alt anahtar 1'in kesişim ile gösterilen 6.09 *(bit)* değeri 1-bit değişimlerle elde edilen 128 alt anahtarın, gizli anahtardan elde edilen alt anahtar ile olan bit farkının ortalamasını göstermektedir. Diğer değerler benzer şekilde diğer alt anahtar ile gizli anahtardan elde edilen alt anahtarların bit farklarının ortalamasını göstermektedir. Sonuçlar, AES anahtar genişletme algoritmasının yavaş yayılım gösterdiğini ve katı çığ kriteri testine göre 1-bit değişim ile anahtarlar arasındaki % 50 lik değişime yaklaşılamadığını göstermektedir.

Tablo 7.1. Rassal Olarak Seçilen 20 Anahtarın 128 Farklı Bit Pozisyonlarındaki Değişimlerle Elde Edilen Alt Anahtarların Yayılım Özelliği

	Alt Anahtar 1	**Alt Anahtar 2**	**Alt Anahtar 3**	**Alt Anahtar 4**	**Alt Anahtar 5**	**Alt Anahtar 6**	**Alt Anahtar 7**	**Alt Anahtar 8**	**Alt Anahtar 9**	**Alt Anahtar 10**
Anahtar 1	6.09	21.52	34.44	45.29	50.09	50.13	51.41	50.45	51.34	49.82
Anahtar 2	6.66	21.64	33.33	44.65	49.45	49.51	48.70	51.09	50.28	52.24
Anahtar 3	6.63	21.41	35.36	47.23	49.56	49.96	49.87	49.72	49.07	51.95
Anahtar 4	6.25	21.78	32.70	46.53	51.26	49.72	49.79	50.23	51.22	49.39
Anahtar 5	6.56	21.09	33.22	47.63	50.21	49.48	50.20	50.68	50.15	49.16
Anahtar 6	6.75	22.56	33.34	47.41	49.08	50.65	50.41	49.84	50.23	51.12
Anahtar 7	6.50	21.38	34.19	46.84	49.99	50.57	51.48	51.38	50.67	50.59
Anahtar 8	6.19	21.50	32.52	47.39	49.52	49.21	48.52	49.52	50.45	50.48
Anahtar 9	6.97	21.61	34.42	45.35	49.66	49.22	51.51	51.48	49.47	50.66
Anahtar 10	6.31	22.19	34.02	45.50	50.24	50.52	49.80	50.33	49.44	50.23
Anahtar 11	6.38	21.34	33.08	44.48	50.43	50.77	50.31	50.56	50.76	50.20
Anahtar 12	6.56	21.06	32.89	46.92	51.02	51.17	48.77	50.69	49.74	52.05
Anahtar 13	6.34	22.80	33.86	46.82	50.43	48.68	49.78	51.03	49.65	50.26
Anahtar 14	6.44	21.66	34.03	46.58	51.90	50.84	51.70	49.88	50.49	49.15
Anahtar 15	6.59	21.05	32.48	45.80	50.87	50.80	50.52	50.91	50.31	49.41
Anahtar 16	6.50	21.44	34.56	44.80	49.05	50.41	50.63	49.16	50.23	49.95
Anahtar 17	6.41	22.17	35.22	45.85	51.56	50.10	49.86	49.64	49.71	50.02
Anahtar 18	6.41	21.70	33.92	46.73	50.55	49.55	50.59	50.77	49.52	51.19
Anahtar 19	5.97	22.30	32.05	47.63	49.84	51.03	50.23	49.63	49.94	50.51
Anahtar 20	7.00	22.06	33.47	45.86	49.48	50.06	49.93	50.28	49.45	50.21
Ortalama	**6.48 (% 5.06)**	**21.49 (% 16.96)**	**33.25 (% 26.29)**	**45.81 (% 36.14)**	**49.67 (% 39.23)**	**49.57 (% 39.16)**	**49.59 (% 39.22)**	**49.81 (% 39.35)**	**49.50 (% 39.15)**	**49.91 (% 39.40)**

Diğer yandan Tablo 7.1'de görüldüğü gibi ilk alt anahtar için yayılım oldukça düşük gözükmektedir. Bununla beraber bu ortalama değer, 128 farklı bit pozisyonunun ilk 96 bit pozisyonu için daha düşük bir değerdedir. Çünkü anahtar genişletme algoritmasının son 32-bit değerinin yayılımı ile tüm 128 bit değere yayılım sağlanmaya çalışılmaktadır.

AES blok şifresinin 192-bit anahtarı için elde edilen deneysel sonuçlar Tablo 7.2'de verilmiştir. Katı çığ kriteri testi 192-bit bir gizli anahtar için olası 192 farklı bit pozisyonu için 1 bit değişimi sonucu elde edilecek alt anahtarlar ile gizli anahtarın alt anahtarları arasındaki bit farklarının ölçülmesi ile gerçekleştirilebilir. Ek olarak bir 192-bit anahtar için elde edilmesi gereken alt anahtar sayısı 8'dir. Bunun nedeni AES-192 blok şifresi 12 döngüde şifreleme yapmasıdır. Dolayısıyla AES-192 blok şifresi temel alındığında 1536 (8×192) alt anahtarın gizli anahtar ile elde edilecek alt anahtarlar ile karşılaştırmaları sonucu bir ortalama değer elde edilebilir. Tablo 7.2 EK B'de 20 rassal 192-bit anahtar için (dolayısıyla 20 × 1536 = 30720 alt anahtar) elde edilen sonuçları göstermektedir.

Tablo 7.2. Rassal Olarak Seçilen 20 Anahtarın 192 Farklı Bit Pozisyonlarındaki Değişimlerle Elde Edilen Alt Anahtarların Yayılım Özelliği

	Alt Anahtar 1	**Alt Anahtar 2**	**Alt Anahtar 3**	**Alt Anahtar 4**	**Alt Anahtar 5**	**Alt Anahtar 6**	**Alt Anahtar 7**	**Alt Anahtar 8**
Anahtar 1	7.38	28.16	47.98	68.11	81.34	83.99	82.97	84.88
Anahtar 2	7.53	29.98	50.21	68.49	79.32	84.24	84.83	83.89
Anahtar 3	7.22	30.02	44.62	65.86	83.78	84.04	82.52	82.90
Anahtar 4	7.41	31.95	50.02	69.76	80.27	84.04	82.65	83.55
Anahtar 5	7.56	28.72	46.30	67.74	81.43	83.90	84.69	84.61
Anahtar 6	7.44	29.47	48.70	68.68	81.57	84.63	83.42	83.95
Anahtar 7	7.59	29.77	49.83	72.42	84.07	84.43	83.88	83.17
Anahtar 8	7.56	28.56	48.02	69.10	82.60	84.81	84.89	84.14
Anahtar 9	7.53	29.89	49.56	71.73	82.46	85.25	83.66	83.73
Anahtar 10	7.63	29.44	47.72	68.60	83.69	86.07	83.82	82.89
Anahtar 11	7.78	28.98	50.59	74.36	82.36	83.68	84.92	83.86
Anahtar 12	7.56	30.31	47.54	71.61	82.81	84.92	84.78	84.73
Anahtar 13	7.59	32.52	47.46	70.01	81.78	83.43	84.95	84.17
Anahtar 14	8.19	31.44	48.68	69.05	79.97	82.13	83.16	82.85
Anahtar 15	7.44	32.28	49.20	68.20	84.60	84.92	84.85	83.28
Anahtar 16	7.50	28.41	51.71	67.61	81.97	84.41	84.15	84.34
Anahtar 17	7.13	29.63	48.43	67.51	81.15	83.59	82.94	83.61
Anahtar 18	7.69	29.28	48.60	68.95	81.83	84.03	84.19	84.65
Anahtar 19	7.47	29.89	49.31	69.21	81.26	83.94	83.61	85.30
Anahtar 20	7.38	27.75	47.21	69.83	82.85	83.69	84.81	85.03
Ortalama	**7.53 (% 3.92)**	**29.82 (% 15.53)**	**48.58 (% 25.30)**	**69.34 (% 36.12)**	**82.06 (% 42.74)**	**84.21 (43.86)**	**83.98 (% 43.74)**	**83.98 (% 43.74)**

AES blok şifresinin 256-bit anahtarı için elde edilen deneysel sonuçlar Tablo 7.3'de verilmiştir. Katı çığ kriteri testi 256-bit bir gizli anahtar için olası 256 farklı bit pozisyonu için 1 bit değişimi sonucu elde edilecek alt anahtarlar ile gizli anahtarın alt anahtarları arasındaki bit farklarının ölçülmesi ile gerçekleştirilebilir. Ek olarak bir 256-bit anahtar için elde edilmesi gereken alt anahtar sayısı 7'dir. Bunun nedeni AES-256 blok şifresi 14 döngüde şifreleme yapmasıdır. AES blok şifresi temel alındığında 1792 (7×256) alt anahtarın gizli anahtar ile elde edilecek alt anahtarlar ile karşılaştırmaları sonucu bir ortalama değer elde edilebilir. Tablo 7.3 EK C'de 20 rassal 256-bit anahtar için (dolayısıyla 20 × 1792 = 35840 alt anahtar için) elde edilen sonuçları göstermektedir.

Tablo 7.3. Rassal Olarak Seçilen 20 Anahtarın 256 Farklı Bit Pozisyonlarındaki Değişimlerle Elde Edilen Alt Anahtarların Yayılım Özelliği

	Alt Anahtar 1	**Alt Anahtar 2**	**Alt Anahtar 3**	**Alt Anahtar 4**	**Alt Anahtar 5**	**Alt Anahtar 6**	**Alt Anahtar 7**
Anahtar 1	15.39	44.10	67.85	92.72	101.33	100.79	99.65
Anahtar 2	14.80	44.77	69.09	93.07	100.53	99.52	99.63
Anahtar 3	14.73	42.70	69.02	90.89	101.74	101.07	102.49
Anahtar 4	14.11	43.71	70.54	92.48	99.25	99.92	98.46
Anahtar 5	15.36	43.49	67.84	92.60	101.05	99.15	99.84
Anahtar 6	14.72	44.14	68.94	91.29	99.48	99.34	99.77
Anahtar 7	14.89	43.70	68.96	91.33	98.97	101.01	101.02
Anahtar 8	15.34	42.88	68.07	89.57	99.64	99.42	99.02
Anahtar 9	14.97	44.97	67.54	93.01	99.94	99.86	100.01
Anahtar 10	14.30	42.30	68.52	92.99	100.68	101.14	99.81
Anahtar 11	14.81	43.52	67.23	93.55	100.36	101.77	100.45
Anahtar 12	14.23	42.51	67.00	91.23	99.64	99.02	100.77
Anahtar 13	15.58	43.93	69.61	91.85	99.68	99.96	100.98
Anahtar 14	14.48	44.29	68.11	90.72	99.81	99.19	99.99
Anahtar 15	14.84	44.59	69.84	91.92	99.66	98.24	100.27
Anahtar 16	14.59	43.56	67.10	91.87	101.00	100.08	101.04
Anahtar 17	14.45	43.60	67.27	89.96	99.10	99.88	100.93
Anahtar 18	13.98	45.27	68.57	89.39	98.61	100.55	101.21
Anahtar 19	13.95	44.88	68.38	92.56	101.23	101.80	100.48
Anahtar 20	15.44	44.03	68.61	92.26	101.13	99.90	100.90
Ortalama	**14.75 (% 5.76)**	**43.85 (% 17.13)**	**68.40 (% 26.72)**	**91.76 (% 35.84)**	**100.14 (% 39.12)**	**100.08 (39.09)**	**83.98 (% 43.74)**

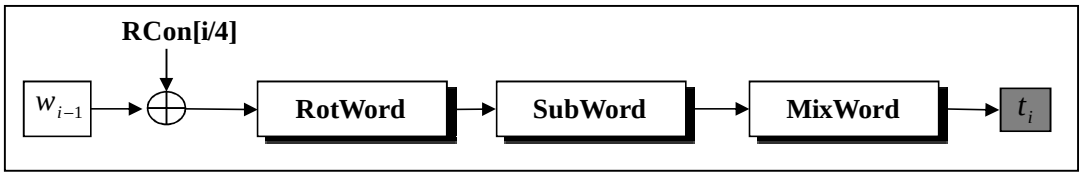
AES blok şifresinin 192-bit ve 256-bit anahtar için genişletme algoritmasının yayılım özellikleri iyi olmamakla beraber yayılımın yavaş bir şekilde arttığı Tablo 7.2 ve Tablo 7.3'den gözlenmektedir.

7.2. AES-128 Anahtar Genişletme Algoritmasının Yayılım Özelliğinin Geliştirilmesi için Önerilen Anahtar Genişletme Algoritması

AES-128 anahtar genişletme algoritmasının tasarımında kullanılan herhangi bir yapı değiştirilmeden ve sadece AES şifresinde kullanılan sütunları karıştırma

(MixColumns) dönüşümü AES-128 anahtar genişletme algoritmasına eklenerek yeni bir anahtar genişletme algoritmasının tasarımı bu bölümde verilecektir. Gerçekte bu tasarım stratejisi bir sonraki Bölüm 8'de anlatılacak olan blok şifreler için anahtar genişletme algoritması tasarımında yeni bir mimari önerisi konusundan esinlenerek geliştirilmiştir.

Geliştirilen anahtar genişletme algoritmasında yeni eklenen kelime karıştırma (MixWord) rutini bir (a_0, a_1, a_2, a_3) şeklinde 32-bit girişi alarak AES şifresindeki MixColumns dönüşümüne sokarak (a_0', a_1', a_2', a_3') şeklinde 32-bit çıkış üretir. Bununla beraber yeni anahtar genişletme algoritmasında geçici t_i değerleri elde edilirken Şekil 5.6'da verilen sıralama yerine Şekil 7.1'de verilen sıralama kullanılmaktadır. Bu sıralama w_i $(\mathrm{i} = 4\mathrm{N_r})$ 32-bit değerine sırasıyla RCon (Döngü Sabiti), RotWord (Kelime Döndürme), SubWord (Kelime Yerdeğiştirme) ve son olarak MixWord (Kelime Karıştırma) uygulanması şeklindedir. AES-128 anahtar genişletme algoritmasının diğer yapıları aynı şekilde kullanılmaktadır.



Şekil 7.1. Geliştirilen Anahtar Genişletme Algoritmasında Geçici t_i değerinin Elde Edilmesinde Kullanılan Elemanlar ve Sırası

Örnek 7.2: Örnek bir gizli anahtarın sadece 1-bit farklılaştığı durum için önerilen anahtar genişletme algoritması ile elde edilen döngü anahtarları ve bu iki gizli anahtardan elde edilen döngü anahtarları arasındaki karşılıklı ilişki aşağıdaki gibi verilebilir.

Gizli Anahtarı 1: (**0**1 23 45 67 89 AB CD EF ED CB A9 98 76 54 32 10)
Gizli Anahtarı 2: (**4**1 23 45 67 89 AF CD EF ED CB A9 98 76 54 32 10)

Döngü	Gizli Anahtar 1 için Döngü Anahtarları	Gizli Anahtar 2 için Döngü Anahtarları	Bit Farkı	Bit Farkı % Değişim
-	**0**1234567 89ABCDEF EDCBA998 76543210	**4**1234567 89ABCDEF EDCBA998 76543210	1	0.78
1	**1**BF5CD1F **9**25E00F0 **7**F95A968 **0**9C19B78	**5**BF5CD1F **D**25E00F0 **3**F95A968 **4**9C19B78	4	3.13
2	4051BF69 D20FBF99 AD9A16F1 A45B8D89	98C90C42 4A970CB2 7502A5DA 3CC33EA2	62	48.44
3	92C0B503 40CF0A9A ED551C6B 490E91E2	34D1EDB6 7E46E104 0B4444DE 37877A7C	68	53.13
4	5C422AE1 1C8D207B F1D83C10 B8D6ADF2	0A2C9FAF 746A7EAB 7F2E3A75 48A94009	75	58.59
5	44C71D63 584A3D18 A9920108 1144ACFA	C784F91D B3EE87B6 CCC0BDC3 8469FDCA	65	50.78
6	30559F67 681FA27F C18DA377 D0C90F8D	EF0067BF 5CEEE009 902E5DCA 1447A000	72	56.25
7	36E32EF0 5EFC8C8F 9F712FF8 4FB82075	CCFE8187 9010618E 003E3C44 14799C44	74	57.81
8	3BCC5161 6530DDEE FA41F216 B5F9D263	FBE0B9C7 6BF0D849 6BCEE40D 7FB77849	55	42.97
9	C9D6A74C ACE67AA2 56A788B4 E35E5AD7	15241F26 7ED4C76F 151A2362 6AAD5B2B	71	55.47
10	ADF85812 011E22B0 57B9AA04 B4E7F0D3	D48194DE AA5553B1 BF4F70D3 D5E22BF8	68	53.13

7.2.1. Önerilen AES Anahtar Genişletme Algoritması için Deneysel Sonuçlar

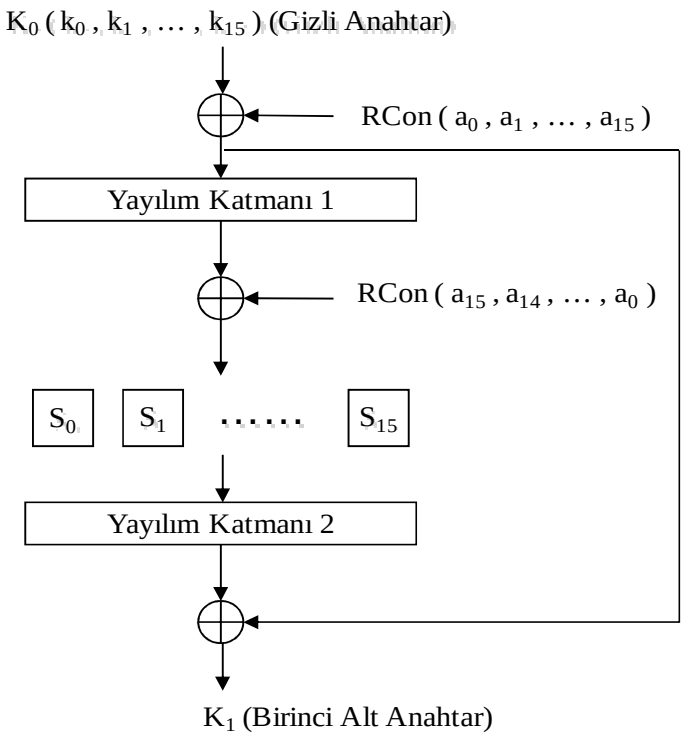
Bölüm 7.1'de gösterildiği gibi rassal olarak seçilen ve EK A'da verilen 20 anahtar üzerindeki deneysel sonuçlar Tablo 7.4'de verilmektedir. Deneysel sonuçlar göstermektedir ki önerilen AES genişletme algoritması yayılım özelliği (katı çığ kriteri testi) açısından çok iyi özellikler sunmaktadır. Ancak sadece ilk alt anahtar için katı çığ kriteri testinin sonuçları iyi özellikler göstermemektedir. Bunun nedeni olarak yayılımın ve karıştırmanın anahtarın son 32-bit veri üzerinde yapılması olarak gösterilebilir. Ancak diğer alt anahtarlar için yeni önerilen algoritmanın özgün olana göre çok daha iyi sonuçlar verdiği gözlenmektedir.

Tablo 7.4. Önerilen AES Anahtar Genişletme Algoritması İçin Rassal Olarak Seçilen 20 Anahtarın 128 Farklı Bit Pozisyonlarındaki Değişimlerle Elde Edilen Alt Anahtarların Yayılım Özelliği

	Alt Anahtar 1	**Alt Anahtar 2**	**Alt Anahtar 3**	**Alt Anahtar 4**	**Alt Anahtar 5**	**Alt Anahtar 6**	**Alt Anahtar 7**	**Alt Anahtar 8**	**Alt Anahtar 9**	**Alt Anahtar 10**
Anahtar 1	18.44	65.25	65.05	63.66	64.50	63.91	64.44	64.13	64.29	63.76
Anahtar 2	19.09	63.75	65.02	63.66	63.81	64.24	63.79	64.11	64.47	64.47
Anahtar 3	19.23	64.31	63.22	64.79	65.09	63.43	62.83	64.08	62.95	64.17
Anahtar 4	17.44	62.89	63.86	63.70	63.61	63.91	63.57	63.68	63.63	63.98
Anahtar 5	18.19	66.41	64.48	63.71	64.07	63.16	63.88	64.49	63.98	63.46
Anahtar 6	18.50	63.91	63.61	64.46	63.87	63.26	63.70	63.83	63.31	63.46
Anahtar 7	18.11	67.69	64.56	64.98	62.98	64.20	63.52	64.17	64.37	64.55
Anahtar 8	18.13	62.28	63.58	64.40	65.22	64.02	63.59	64.08	64.51	63.11
Anahtar 9	18.25	65.53	63.98	64.16	64.00	64.96	64.30	63.56	64.49	63.84
Anahtar 10	18.17	65.83	64.09	64.05	64.49	63.78	63.75	63.60	63.42	64.56
Anahtar 11	17.72	65.83	64.58	64.54	64.41	63.71	63.57	64.40	64.05	64.02
Anahtar 12	17.73	62.88	64.48	63.93	64.09	64.37	64.22	63.91	64.00	63.96
Anahtar 13	18.52	65.44	64.03	63.62	63.71	64.62	63.20	64.34	64.52	64.56
Anahtar 14	18.33	62.06	62.95	64.77	64.10	63.86	64.59	64.33	64.13	63.60
Anahtar 15	18.94	64.30	64.06	65.10	64.16	63.26	64.84	64.34	64.29	63.59
Anahtar 16	18.19	65.19	63.44	64.20	63.93	63.85	64.45	64.38	64.34	63.77
Anahtar 17	18.20	64.95	64.33	64.78	63.43	62.79	63.81	63.87	63.83	63.56
Anahtar 18	17.34	64.47	64.44	63.99	64.61	63.91	64.23	64.53	64.04	64.02
Anahtar 19	17.28	67.19	63.27	64.74	63.98	63.63	64.27	62.86	64.26	64.13
Anahtar 20	18.38	62.06	64.67	63.60	63.84	64.53	63.27	64.21	63.25	63.02
Ortalama	**18.21 (% 14.23)**	**64.61 (% 50.48)**	**64.09 (% 50.07)**	**64.24 (% 50.19)**	**64.10 (% 50.07)**	**63.87 (% 49.90)**	**63.89 (% 49.91)**	**64.05 (% 50.04)**	**64.01 (% 50.01)**	**63.88 (% 49.91)**

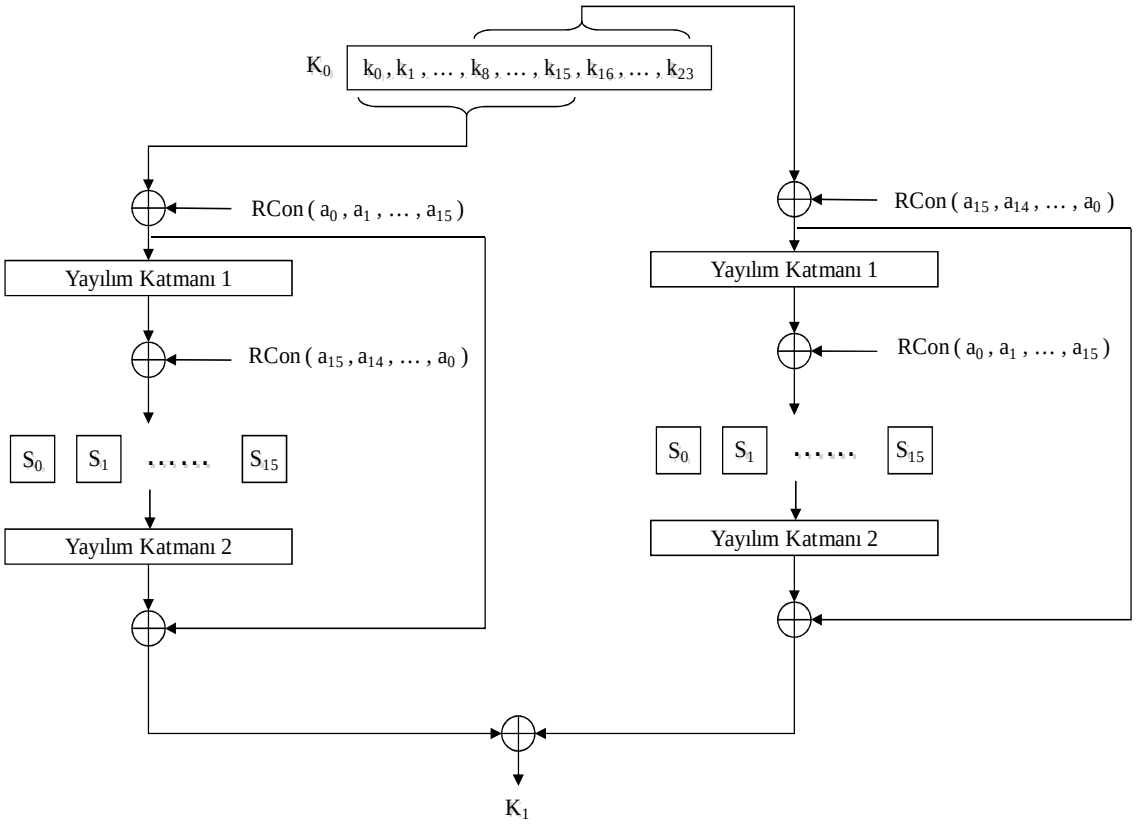
8. BLOK ŞİFRELER İÇİN YENİ BİR ANAHTAR GENİŞLETME MİMARİ ÖNERİSİ

Bir anahtar genişletme algoritması daha önce de belirtildiği gibi bazı önemli özelliklere sahip olması gerekir. Özellikle AES blok şifresinin anahtar genişletme algoritması katı çığ kriteri testleri kötü özellikler göstermektedir. Buna ek olarak bu algoritmanın herhangi bir saldırıda kullanılabilecek bit sızdırma özelliği de bulunmaktadır. Bu gibi problemleri yok etme amacıyla yeni geliştirilen mimaride alt anahtarlar birbirlerinden bağımsız şekilde üretilmektedir. Ayrıca yüksek yayılım özelliği gösterecek ve herhangi bir blok şifreye adapte edilebilecek şekilde tasarımı yapılmaktadır. Bu mimari kısaca yayılım-yer değiştirme-yayılım (*Diffusion-Substitution-Diffusion*) şeklinde isimlendirilebilir. Yeni önerilen anahtar genişletme mimarisinin en önemli özelliği alt anahtarların elde edilmesi birbirinden bağımsız hale getirilirken yazılım ve donanım uygulamasının tamamen XOR işlemleri tabanlı hale getirilmesidir. Şekil 8.1'de bu mimari 128-bit gizli anahtar için bir alt anahtarın elde edilmesi için gösterilmektedir. Aynı mimari ve farklı döngü sabitleri kullanılarak farklı alt anahtarlar üretmek mümkündür. Yine Şekil 8.1'de gösterilen yayılım katmanı 1 ve 2, yüksek yayılım özelliği gösteren doğrusal dönüşümlerdir. Yer değiştirme katmanı ise S-kutularından oluşmaktadır ve birçok blok şifrede kullanıldığı gibi 8-bit giriş ve 8-bit çıkışa sahip doğrusal olmayan yapılardır. Bir alt anahtarı elde ederken döngü sabiti (RCon) bu mimaride $RCon(a_0, a_1, ..., a_{15})$ ve $RCon(a_{15}, a_{14}, ..., a_0)$ şeklinde 2 defa kullanılmaktadır. Bunun nedeni yayılım katmanı 1'in sahip olduğu az sayıdaki sabit nokta sayısını yok etme amacıdır. Bununla beraber bu çalışmada doğrusal dönüşümler sabit nokta sayısı oldukça düşük 128-bit girişi 128-bit çıkışa haritalayan ikili doğrusal dönüşümler olarak seçilmektedir.



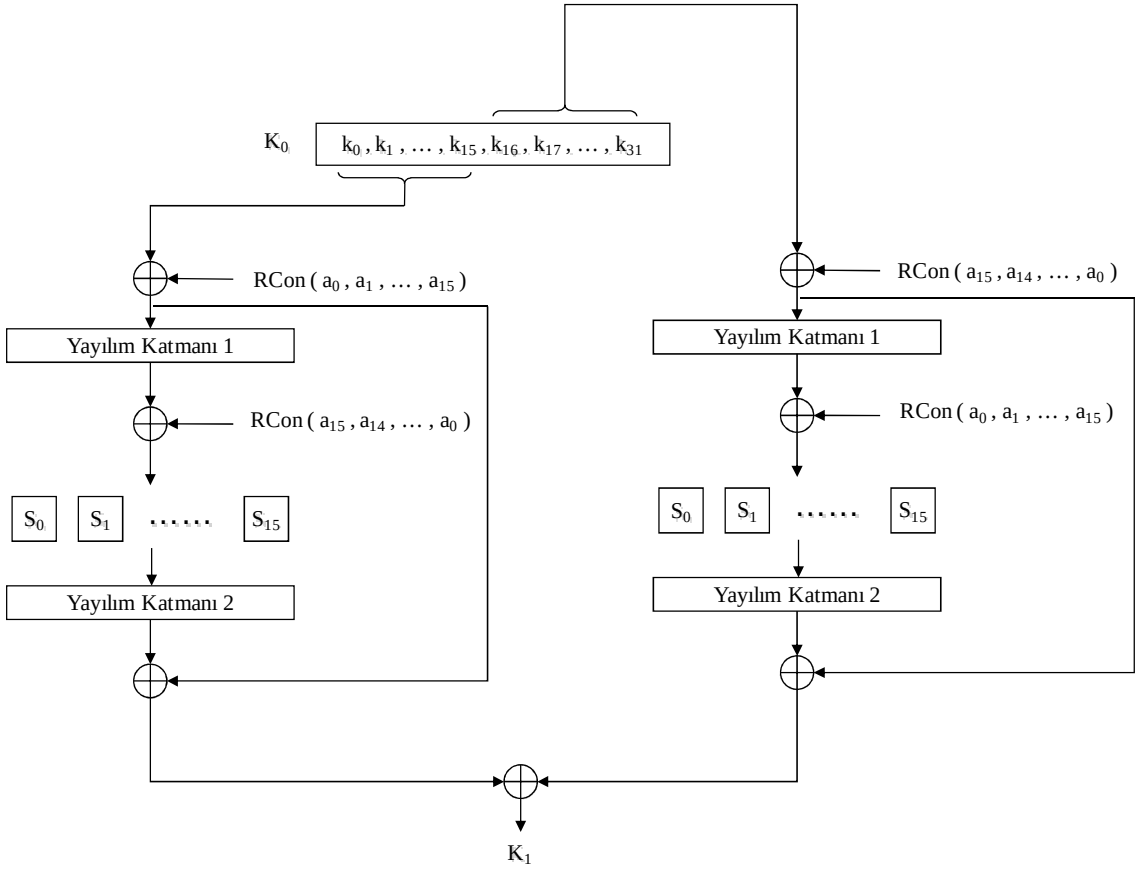
Şekil 8.1. 128-bit Gizli Anahtar için Önerilen Anahtar Genişletme Mimarisi

AES blok şifresinde olduğu gibi 192-bit ve 256-bit anahtarlar seçenekleri içinde anahtar genişletme algoritmasının verilen mimariye uygun hale getirilmesi ve 128-bit anahtar seçeneği için geliştirilecek uygulama ile yakın istatistiksel özellikler göstermesi gerekmektedir. 192-bit anahtar seçeneği için mimarinin uygun hale getirilmesi için Şekil 8.2'de gösterildiği gibi gizli anahtar iki parçaya ayrılarak 192-bit anahtar genişletme algoritması geliştirilmiştir. Geliştirilen anahtar genişletme algoritması simetrik çalışan 128-bit iki yapı sonucu 128-bit anahtar üretmektedir. Yine AES blok şifresi düşünüldüğünde 192-bit anahtar seçeneği ile şifreleme yapan AES algoritması 12 döngüde şifreleme yaptığı için elde edilecek alt anahtarların sayısı 12 tane 128-bit alt anahtar olacak şekildedir.



Şekil 8.2. 192-bit Gizli Anahtar için Önerilen Anahtar Genişletme Mimarisi

256-bit anahtar seçeneği için geliştirilen mimari 192-bit anahtar için olanla benzerdir. 256-bit anahtar önce iki parçaya ayrılarak Şekil 8.3'de gösterildiği gibi simetrik çalışan 128-bit iki yapı sonucu 128-bit anahtar üretmektedir. Yine AES blok şifresi temel alındığında AES-256 blok şifresi 14 döngüde şifreleme yaptığı için elde edilecek alt anahtarların sayısı 14 tane 128-bit alt anahtar olacak şekildedir.



Şekil 8.3. 256-bit Gizli Anahtar için Önerilen Anahtar Genişletme Mimarisi

8.1. Geliştirilen Yeni Anahtar Genişletme Mimarisi İçin Örnek Uygulamalar

Yeni anahtar genişletme mimarisi için 128-bit, 192-bit ve 256-bit anahtar seçenekleri için örnek bir uygulama bu bölümde tanıtılacaktır. Bölüm 8'de verilen 128-bit anahtar genişletme algoritması için gerekli olan elemanlar aşağıda verilmiştir:

- 10 farklı döngü sabiti,
- 2 farklı yüksek yayılım özelliğine sahip doğrusal dönüşüm,
- S-kutusu,

Uygulamada kullanılan döngü sabitleri EK D'de verilmiştir. Buna ek olarak yüksek dallanma sayısına sahip ikili doğrusal dönüşümler uygulamada kullanılacaktır. Yayılım katmanı 1 için dallanma sayısı 8 olan ve sabit nokta sayısı 2^8 olan bir ikili doğrusal dönüşüm ($A-I$ matrisinin rankı 15 olan) ve yayılım katmanı 2 için dallanma sayısı 7 olan ve sabit nokta sayısı 1 olan ikili bir doğrusal dönüşüm ($A-I$ matrisinin rankı 16 olan) kullanılmıştır. Anahtar genişletme algoritması için S-kutusu olarak AES S-kutusu seçilmiştir. Doğrusal dönüşümler (Aslan, 2011) çalışmasında verilen cebirsel yöntem ile geliştirilmiştir. Doğrusal dönüşüm 1,

$$Had(3_h, B_h, 7_h, E_h) = \begin{bmatrix} 3_h & B_h & 7_h & E_h \\ B_h & 3_h & E_h & 7_h \\ 7_h & E_h & 3_h & B_h \\ E_h & 7_h & B_h & 3_h \end{bmatrix}_{4\times 4}$$

GF(2^4) elemanlarından oluşan 4×4 matrisin ikili forma dönüştürülmesinden (16×16 matrise) sonra tüm satırlarının 1-bit sağa ötelenmesi ile elde edilmiştir (Not edilmelidir ki GF(2^4) indirgenemez polinom $x^4 + x + 1$ ile tanımlıdır). Elde edilen ikili doğrusal dönüşüm (dallanma sayısı 8 ve $A-I$ matrisinin rankı 15) aşağıda verilmiştir:

$$DD_1 = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \\ y_{11} \\ y_{12} \\ y_{13} \\ y_{14} \\ y_{15} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \end{bmatrix}$$

Verilen ikili doğrusal dönüşümde $(x_0, x_1, ..., x_{15})$ giriş byte değerlerini temsil ederken $(y_0, y_1, ..., y_{15})$ çıkış byte değerlerini temsil etmektedir. Dolayısıyla 16 byte (128-bit) giriş değerleri 16 byte (128-bit) çıkış değerlerini haritalanmaktadır.

Diğer yandan ikinci yayılım elemanı dallanma sayısı 7 ve $A - I$ matrisinin rankı 16 olan ikili bir doğrusal dönüşümdür. Yine yukarıda bahsedildiği gibi cebirsel olarak elde edilmiştir ve

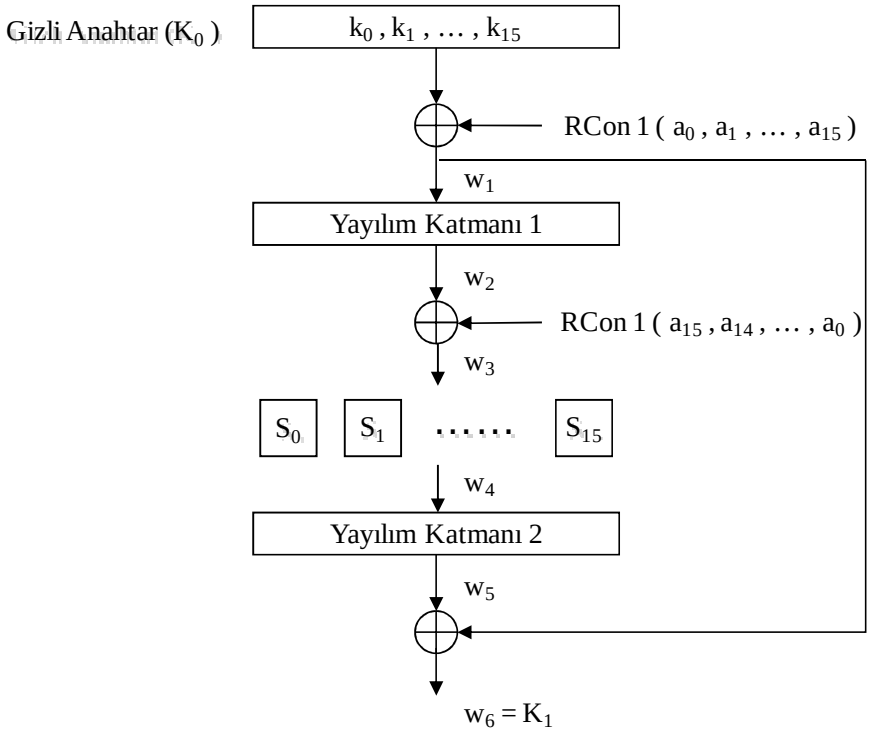
$$Had(7_h, 8_h, B_h, D_h) = \begin{bmatrix} 7_h & 8_h & B_h & D_h \\ 8_h & 7_h & D_h & B_h \\ B_h & D_h & 7_h & 8_h \\ D_h & B_h & 8_h & 7_h \end{bmatrix}_{4\times 4}$$

GF(2^4) elemanlarından oluşan 4×4 matrisin ikili forma dönüştürülmesi (16×16 matrise) ile elde edilen ikili doğrusal dönüşüm aşağıdaki gibi verilebilir:

$$DD_2 = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \\ y_{11} \\ y_{12} \\ y_{13} \\ y_{14} \\ y_{15} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \end{bmatrix}$$

Örnek 8.1: Önerilen anahtar genişletme algoritmasının 128-bit versiyonu için adım adım gizli anahtardan, EK D'de verilen döngü sabiti 1'i kullanarak, ilk alt anahtarın elde edilmesini gösterelim. Şekil 8.4'de verilen $(w_1, w_2, w_3, w_4, w_5, w_6)$ konumlarına göre Hexadecimal gösterimde çıkışlar aşağıdaki gibi verilebilir:

Gizli anahtar:	`0123456789ABCDEFEDCBA99876543210`
w_1 ($RCon(a_0, a_1, ..., a_{15})$ çıkışı):	`9DE01A63E3D338961189458C59998314`
w_2 (Doğrusal Dönüşüm 1 çıkışı):	`AF1747C65B58841788F7D0AE5F5C7E49`
w_3 ($RCon(a_{15}, a_{14}, ..., a_0)$ çıkışı):	`ABA68AE94FB4C6EBF102A8C45B03BDD5`
w_4 (S-kutusu çıkışı):	`62247E1E848DB4E9A177C21C397B7A03`
w_5 (Doğrusal Dönüşüm 2 çıkışı):	`7D7D2908A8BBED39E1FECF86EF4AE3C9`
w_6 (İlk anahtar $(w_1 \oplus w_5)$):	`E09D336B4B68D5AFF0778A0AB6D360DD`



Şekil 8.4. 128-bit Gizli Anahtar için Bir Alt Anahtarın Adım Adım Elde Edilmesi

Örnek 8.2: Örnek bir 128-bit gizli anahtarın sadece 1-bit farklılaştığı durum için önerilen anahtar genişletme algoritması (EK D'de verilen ilk 10 döngü sabiti kullanılarak) ile elde edilen döngü anahtarları ve bu iki gizli anahtardan elde edilen döngü anahtarları arasındaki karşılıklı ilişki aşağıdaki gibi verilebilir.

Gizli Anahtarı 1: (**0**1 23 45 67 89 AB CD EF ED CB A9 98 76 54 32 10)
Gizli Anahtarı 2: (**4**1 23 45 67 89 AF CD EF ED CB A9 98 76 54 32 10)

Döngü	Gizli Anahtar 1 için Döngü Anahtarları	Gizli Anahtar 2 için Döngü Anahtarları	Bit Farkı	Bit Farkı % Değişim
-	**0**123456789ABCDEFEDCBA99876543210	**4**123456789ABCDEFEDCBA99876543210	1	0.78
1	E09D336B4B68D5AFF0778A0AB6D360DD	23FFC31877D5B036E5B0F7CEA062E775	65	50.78
2	BB79AC96DCB1A552CBB4A79B8804BB35	B024D66BE1A698415D6AC2AAE5531E78	72	56.25
3	471DADDDF0591CC08298407C9828E918	A3224782D5E69A9B58FA352662B8FE6C	72	56.25
4	1D7681B79037AAB2B31DF0EFF2EFE8CA	C3B50399416B0457A6E709C57F32F4B1	71	55.47
5	EB8DCC9136584BDC3D1EC527EAAA97D7	C87F040A604B80117499581D543DAEBD	68	53.13
6	F72FD12E0614268FBF14B969863D7318	819AF5CF1F2FB1AED04F08D326BAA3BF	65	50.78
7	280FA5DF58B3374637FB13CEE1FF8000	1C5223FBBA8505668B7512EBE64DF75C	55	42.97
8	A860298D6F7379F97B325D2531C75D49	EE1663E52064F9D5A2D973F127C648A3	58	45.31
9	6516BC9F8DE9E90CEBC25497337232B8	E2342C84D738845B28D195F079F4BEE6	59	46.09
10	EE194C17EA7855A7C21DDE04EE86A149	D1B07FB705DE356E8A63235DDFE42D4B	62	48.44

Bölüm 7.1'de gösterildiği gibi rassal olarak seçilen ve EK A'da verilen 20 anahtar üzerindeki deneysel sonuçlar Tablo 8.1'de verilmektedir. Deneysel sonuçlar göstermektedir ki önerilen anahtar genişletme mimarisi uygulamada kullanılan elemanlar ile birlikte yayılım özelliği (katı çığ kriteri testi) açısından çok iyi özellikler sunmaktadır.

Tablo 8.1. Önerilen Anahtar Genişletme Algoritması İçin Rassal Olarak Seçilen 20 Anahtarın 128 Farklı Bit Pozisyonlarındaki Değişimlerle Elde Edilen Alt Anahtarların Yayılım Özelliği

	Alt Anahtar 1	**Alt Anahtar 2**	**Alt Anahtar 3**	**Alt Anahtar 4**	**Alt Anahtar 5**	**Alt Anahtar 6**	**Alt Anahtar 7**	**Alt Anahtar 8**	**Alt Anahtar 9**	**Alt Anahtar 10**
Anahtar 1	63.81	64.00	63.24	63.46	63.80	65.15	65.09	63.13	64.15	63.79
Anahtar 2	63.70	63.68	64.38	64.00	63.21	64.51	64.79	63.30	63.93	64.89
Anahtar 3	63.87	64.79	63.75	64.58	64.57	63.48	63.30	64.73	64.45	64.02
Anahtar 4	64.55	64.12	63.82	63.62	65.05	63.54	64.12	64.73	63.62	64.59
Anahtar 5	63.64	65.02	64.20	64.47	63.91	63.66	64.93	64.50	64.18	64.34
Anahtar 6	64.31	63.24	63.62	63.73	64.45	64.30	63.77	63.71	63.92	64.39
Anahtar 7	64.03	64.01	64.43	63.84	64.33	64.44	65.00	64.66	64.20	63.61
Anahtar 8	63.18	63.76	63.50	64.41	63.70	64.02	63.98	63.91	63.14	64.43
Anahtar 9	63.61	63.45	63.63	64.02	63.64	63.01	63.97	64.57	64.56	63.91
Anahtar 10	64.83	63.16	64.64	63.70	63.20	64.53	64.05	64.13	63.09	64.38
Anahtar 11	62.92	64.71	63.40	63.62	63.79	64.71	63.75	64.83	64.81	63.81
Anahtar 12	62.84	64.58	64.49	63.18	65.37	64.73	64.55	63.95	64.32	63.45
Anahtar 13	65.02	63.65	63.97	63.64	63.62	63.25	64.02	63.55	65.16	64.11
Anahtar 14	64.23	63.70	64.87	64.43	65.08	64.68	63.86	64.78	64.41	64.21
Anahtar 15	64.77	63.20	64.09	64.05	64.38	64.38	63.81	63.74	64.21	64.39
Anahtar 16	63.44	63.60	64.02	63.87	64.69	63.36	63.88	63.67	64.39	63.95
Anahtar 17	64.39	64.23	63.70	64.45	64.63	64.17	64.47	63.95	63.98	64.30
Anahtar 18	62.66	63.18	63.63	63.91	64.19	63.58	64.83	64.22	63.70	64.15
Anahtar 19	63.46	64.42	64.11	64.28	64.41	64.07	63.66	63.55	64.16	64.09
Anahtar 20	63.86	65.03	63.86	64.56	65.14	64.05	63.07	62.62	64.28	63.58
Ortalama	**63.86 (% 49.89)**	**63.98 (% 49.98)**	**63.97 (% 49.97)**	**63.99 (% 49.99)**	**64.26 (% 50.20)**	**64.08 (% 50.06)**	**64.15 (% 50.11)**	**64.01 (% 50.01)**	**64.13 (% 50.10)**	**64.12 (% 50.09)**

Örnek 8.3: Örnek 192-bit bir gizli anahtarın sadece 1-bit farklılaştığı durum için önerilen anahtar genişletme algoritması (EK D'de verilen ilk 12 döngü sabiti kullanılarak) ile elde edilen döngü anahtarları ve bu iki gizli anahtardan elde edilen döngü anahtarları arasındaki karşılıklı ilişki aşağıdaki gibi verilebilir.

Gizli Anahtarı 1: (**0**123456789ABCDEFEDCBA99876543210**0**123456789ABCDEF)
Gizli Anahtarı 2: (**4**123456789ABCDEFEDCBA99876543210**0**123456789ABCDEF)

Döngü	Gizli Anahtar 1 için Döngü Anahtarları
-	0123456789ABCDEFEDCBA9987654321001234567 89ABCDEF
1	687728B7F64D5F1D1F1B74AB2E5265D501F281251A2935B2
2	50BACCAB9CED41F8FF9BB365658B8E3EC59A511799298A7E
3	5E5F55C92640D5C20D8983D549CA1E6C582E991002E8DE64
4	8C2AEBDA80D6AD3CA19A49862A872FCF0F0394C206A56F88
5	D7AA0C91FFC4903978C66C8A76F2C5B2157D8D08E4082208
6	124E1339548AB72FD6DAEC68D0C3058383F8A2E46B3BA6E1
7	E82CDA474B68FED49C4AB4B4D21909A98733AB72A625D719
8	BE784D6703F76580241248A9C7DD93AD9BA7DB227EB8BA46

Döngü	Gizli Anahtar 2 için Döngü Anahtarları	Bit Farkı	Bit Farkı % Değişim
-	4123456789ABCDEFEDCBA9987654321001234567 89ABCDEF	1	0.52
1	AB15D8C4CAF03A840ADC096F38E3E27D0AAFFBD8273E08A1	102	53.13
2	C664A99AF1BAE4B51BA4593A403408651FF8244D63B99D0A	107	55.73
3	809CD7E7F71C7B2718737AFFC41702177BDC518B54FB15A9	104	54.17
4	C5AD76E03E419456D72F6D6733BCB8EE60582578A622BF2F	100	52.08
5	E3F78AB51DF2A219C4486DAF7140B2EE530BC760AB1FA224	82	42.71
6	CBA53DED428BA2C551F87C738A1268D440EB638321BD2ABF	90	46.88
7	D785E9E7A4CE9E1DD43449EDE37B85AB373E4D86C33A9D95	93	48.44
8	107FF0EF5EB45CD36B67555490A5C9EBB41DD94807477BFE	104	54.17

192-bit anahtar genişletme için önerilen mimari ile ilgili deneysel sonuçlar Tablo 8.2’de verilmektedir. Tablo 8.2’de elde edilen veriler rassal olarak seçilen ve EK B’de verilen 20 anahtardan elde edilmiştir. Buna ek olarak anahtarların elde edilmesinde kullanılan döngü sabitleri EK D’de verilmiştir. 192-bit gizli anahtardan farklı alt anahtarlar elde edildiği için EK D’deki döngü sabitlerinin ilk 12’si kullanılmıştır. Tablo 8.2’deki sonuçlara göre katı çığ kriteri testi için önerilen mimari anahtar genişletme algoritmasında kullanılan elemanlar ile birlikte çok iyi sonuçlar göstermektedir.

Tablo 8.2. Önerilen Anahtar Genişletme Algoritması İçin Rassal Olarak Seçilen 20 Anahtarın 192 Farklı Bit Pozisyonlarındaki Değişimlerle Elde Edilen Alt Anahtarların Yayılım Özelliği

	Alt Anahtar 1	**Alt Anahtar 2**	**Alt Anahtar 3**	**Alt Anahtar 4**	**Alt Anahtar 5**	**Alt Anahtar 6**	**Alt Anahtar 7**	**Alt Anahtar 8**
Anahtar 1	96.11	96.51	94.49	96.14	95.41	95.42	95.31	96.28
Anahtar 2	95.85	95.88	95.96	96.21	97.29	96.33	95.36	95.60
Anahtar 3	97.06	95.80	95.93	95.91	96.59	95.81	95.86	95.67
Anahtar 4	96.40	95.91	95.79	95.64	95.97	95.54	95.79	96.59
Anahtar 5	95.72	95.20	96.15	95.90	95.78	95.78	96.15	95.52
Anahtar 6	96.48	96.04	96.17	96.08	96.38	95.92	96.13	96.05
Anahtar 7	97.33	96.69	96.01	96.52	95.44	95.72	96.52	94.99
Anahtar 8	95.81	96.42	95.33	95.67	96.89	96.28	96.27	95.01
Anahtar 9	96.16	95.81	96.34	95.33	95.71	95.14	95.04	96.48
Anahtar 10	96.16	96.02	95.91	95.72	96.54	95.41	96.48	95.94
Anahtar 11	96.88	95.43	96.19	95.55	95.74	96.26	96.07	96.07
Anahtar 12	95.86	96.06	96.24	96.27	95.17	96.50	95.79	96.17
Anahtar 13	96.48	95.54	97.01	95.53	97.31	96.04	96.34	96.07
Anahtar 14	96.21	94.64	95.40	96.02	96.19	96.60	95.14	96.40
Anahtar 15	96.32	95.51	95.90	95.88	96.42	96.54	95.85	96.68
Anahtar 16	96.80	96.09	96.65	95.22	95.05	96.63	96.03	95.55
Anahtar 17	96.02	95.26	96.62	96.29	96.57	95.74	96.55	95.28
Anahtar 18	96.47	96.20	95.08	96.98	96.77	95.81	96.56	96.22
Anahtar 19	94.92	95.67	95.95	96.70	97.05	95.88	96.08	96.65
Anahtar 20	96.31	95.45	95.30	96.19	96.47	96.34	96.10	96.91
Ortalama	**96.27 (% 50.14)**	**95.81 (% 49.90)**	**95.92 (% 49.96)**	**95.99 (% 49.99)**	**96.24 (% 50.12)**	**95.98 (49.99)**	**95.97 (% 49.98)**	**96.01 (% 50.00)**

Örnek 8.4: Örnek 256-bit bir gizli anahtar için önerilen anahtar genişletme algoritması (EK D'de verilen ilk 14 döngü sabiti kullanılarak) ile elde edilen 256-bit döngü anahtarları aşağıdaki tabloda verilmektedir. AES algoritması ile 256-bit anahtar için 14 döngüde şifreleme yapıldığından aşağıda verilen alt anahtarların sayısı 7'dir. Buna ek olarak diğer tabloda örnek 256-bit anahtarın 1-bit farklılaşması ile elde edilen alt anahtarlar ve gizli anahtardan elde edilen alt anahtarlar arasındaki ilişki de verilmektedir.

Döngü	Gizli Anahtar 1 için Döngü Anahtarları
-	0123456789ABCDEFEDCBA99876543210 0123456789ABCDEFEDCBA99876543210
1	4C010E2498BEEEDE9A9A61C9E4560DCF1AD467D9352550141876EEA7DDE28828
2	5B84D745FA65B1A6AF3F84E332ADF42E1E92415A0214A152912D4C02A5C8B09A
3	25966A1F2BBE497121FAFAD2226426B8F72A23F35F1D1AE1A7F48D07869E3036
4	EC31248B2FF78430A6A2C67A0C032B8BAF32F70CA5CC83A7FB4A71B0196DA825
5	58A0D2027D790248ABB8F4597E338B60763C0EF7CE7DC25C6B70A13800590981
6	38E948AC046B31B4075A086CE0EEFC72656FDEDC02E9A9CE3826F16DF35DEB41
7	6CFA008C6F7FE6FA65A48F59FAB98A40FF558D451EA8889BF7F76A0803BA09AC

Döngü	Gizli Anahtar 2 için Döngü Anahtarları	Bit Farkı	Bit Farkı % Değişim
-	4123456789ABCDEFEDCBA99876543210 0123456789ABCDEFEDCBA99876543210	1	0.52
1	8F63FE57A4038B478F5D1C0DF2E78A6711891D2408326D078EA88B96B0B52D65	137	53.52
2	BFBB3D1ADFDA37FD755DF1B9C83DE35AC051C374D3480FB784D7B5282815ACE1	143	55.86
3	0664A2847DAD82BC687D67E89CF31FD2819F071246268DC0C8AF3CBD2619E091	133	51.95
4	D86CA2AFCDC1B6101A2CC75F0BB15CD7E944BD64EADB038B22A15F640F6CBDCF	113	44.14
5	DF82421927A86F1F68AB353E34B5073E49953D5721DBA295230E5C61313B8583	121	47.27
6	88E4AE5861747B38A95DB5E4BDADC5212A1AC3215591F388179CF3078AA22AF9	135	52.73
7	7380B22DC5F3F0660B76F198D35978EE1F4649BF8390BC380E19058B0E48BEF3	136	53.13

256-bit anahtar genişletme için önerilen mimari ile ilgili deneysel sonuçlar Tablo 8.3'de verilmektedir. Tablo 8.3'de elde edilen veriler rassal olarak seçilen ve EK C'de verilen 20 anahtardan elde edilmiştir. Buna ek olarak anahtarların elde edilmesinde kullanılan döngü sabitleri EK D'de verilmiştir. 256-bit gizli anahtardan farklı alt anahtarlar elde edildiği için EK D'deki döngü sabitlerinin ilk 14'ü kullanılmıştır. Tablo 8.3'deki sonuçlara göre katı çığ kriteri testi için önerilen mimari anahtar genişletme algoritmasında kullanılan elemanlar ile birlikte çok iyi sonuçlar göstermektedir.

Tablo 8.3. Önerilen Anahtar Genişletme Algoritması İçin Rassal Olarak Seçilen 20 Anahtarın 256 Farklı Bit Pozisyonlarındaki Değişimlerle Elde Edilen Alt Anahtarların Yayılım Özelliği

	Alt Anahtar 1	**Alt Anahtar 2**	**Alt Anahtar 3**	**Alt Anahtar 4**	**Alt Anahtar 5**	**Alt Anahtar 6**	**Alt Anahtar 7**
Anahtar 1	127.72	127.97	127.22	126.74	128.04	128.79	128.34
Anahtar 2	128.51	127.57	127.28	128.27	128.29	128.33	128.66
Anahtar 3	127.51	128.61	126.55	128.21	129.03	128.08	128.22
Anahtar 4	127.19	127.40	127.58	128.47	128.10	127.84	127.94
Anahtar 5	128.48	128.13	128.60	127.93	127.96	128.45	128.25
Anahtar 6	127.30	127.60	128.40	127.57	127.75	126.76	127.64
Anahtar 7	128.87	127.93	127.29	127.76	128.45	128.30	127.50
Anahtar 8	128.42	127.73	128.63	128.44	128.32	127.75	128.18
Anahtar 9	128.76	126.97	128.80	128.12	128.19	128.22	128.10
Anahtar 10	128.59	128.07	127.45	127.55	128.05	128.36	128.43
Anahtar 11	128.07	127.80	128.37	127.41	127.79	128.99	128.11
Anahtar 12	128.50	128.76	127.97	127.70	128.82	128.48	126.54
Anahtar 13	127.39	127.56	127.89	127.64	127.98	128.02	128.46
Anahtar 14	127.79	127.46	127.27	127.90	128.12	127.60	128.01
Anahtar 15	128.39	128.58	127.80	127.76	128.03	128.17	128.78
Anahtar 16	128.12	127.11	128.54	128.35	128.00	128.17	128.45
Anahtar 17	127.29	127.32	127.21	127.11	128.99	127.64	126.95
Anahtar 18	128.29	127.77	127.10	128.21	127.78	127.06	129.07
Anahtar 19	128.21	128.43	127.04	127.98	129.06	127.70	127.78
Anahtar 20	128.21	128.34	127.29	128.70	128.58	127.54	128.20
Ortalama	**128.08 (% 50.03)**	**127.86 (% 49.94)**	**127.71 (% 49.89)**	**127.89 (% 49.96)**	**128.27 (% 50.10)**	**128.01 (% 50.00)**	**128.08 (% 50.03)**

9. SONUÇLAR VE DEĞERLENDİRME

Bu çalışma akış şifreleme algoritmalarının incelenmesi sonucu göze çarpan anahtar akışı ve bu anahtarların güç analiziyle ilgilidir. Şifreleme algoritmalarının gücünün anahtarın gücüne bağımlı olması tez çalışmasının anahtar üretimine yönelik incelemelere yoğunlaşmasını sağlamıştır. Anahtarın gücü matematiksel olarak tahmin edilememe felsefesine dayanmaktadır. Tahmin edilememe rassallıkla ilişkilidir. Kullanılan anahtar ne kadar rassalsa anahtar dolayısıyla da şifreleme algoritması o kadar güçlüdür. Bu bilgiler çerçevesinde akış şifre algoritmalarından son dönemde öne çıkan bazı algoritmaların ürettiği anahtar akışları rassallık testine tabi tutularak güçleri analiz edilmiştir.

Uygulanan yöntem için NIST test paketi tercih edilmiş, böylelikle uluslararası standartlara bağlı kriterlerle algoritmaların gücünün incelenmesi sağlanmıştır.

Akış şifrelerin algoritmalarında son dönemde dikkat çeken özellik simetrik şifreleme tekniklerinin diğer bir dalı olan blok şifrelerin akış şifrelerle birleştirilerek akış şifrelerdeki dayanıklılığı arttırma çabalarıdır. Bununla beraber bir blok şifre olan AES şifresinin anahtar genişletme algoritması bu tezde incelenerek çeşitli saldırılara karşı kullanılabilir olan bu rutinin güçlendirilebileceği gözlenmiştir. Bu gözlem için SAC (Strict Avalanche Criterion) testi çeşitli rassal anahtarlara uygulanmıştır. Bu inceleme sonucu AES blok şifresinin anahtar genişletme rutinin yapısının iyi ama eksik bir yapıya sahip olduğu kanaatine varılmıştır.

Diğer yandan bir anahtar genişletme algoritması için iki önemli özellik olan yavaş yayılım ve bit sızdırma özellikleri temel alınarak bir blok şifreden bağımsız yeni bir anahtar genişletme mimarisi önerilmiş ve deneysel sonuçlar sunulmuştur. Bununla beraber bu önerilen yeni tekniğin yazılım uygulaması XOR tabanlı olduğu için performansı da iyi sonuçlar vermektedir. Öne sürülen yaklaşımın blok şifrelerin gücünü arttırması açısından önemli bir katkı sağladığı düşünülmektedir.

TEZ SIRASINDA YAPILAN ÇALIŞMALAR

Uluslararası Bilimsel Toplantılarda Sunulan Ve Bildiri Kitaplarında Basılan Bildiriler

Sakallı M.T., Aslan B., Buluş E., Şahin Mesut A., Büyüksaraçoğlu F., Karaahmetoğlu O., Second International Conference, NDT 2010 konferansı dahilinde "Networked Digital Teknologies" bildiri kitapçığındaki "On the Algebraic Expression of the AES S-Box like S-Boxes", 213-227 pp., Prague, Czech Republic, Temmuz 2010

Büyüksaraçoğlu F., Buluş E., Sakallı M.T., Unitech'08 konferansı dahilinde "International Scientific Conference Unitech'08 Proceedings" bildiri kitapçığındaki "Examining Five Cryptographic Test Methods For Cryptographic Primitives", III-371 - III-376 pp., Gabrovo, Bulgaristan, Kasım 2008

Sakalli M. T., Bulus E., Sahin A., Buyuksaracoglu F., SIU 2007 konferansı dahilinde "2007 IEEE 15th Signal Processing and Communications Applications, Vols 1-3" bildiri kitapçığındaki "Obtaining algebraic expression of an S-box based on inversion mapping using finite field theory", 426-429 pp., Eskisehir, TURKEY, Haziran 2007

Sakalli M. T.; Bulus E.; Sahin A.; Buyuksaracoglu F., SIU 2006 konferansı dahilinde "Signal Processing and Communications Applications, 2006 IEEE 14th" bildiri kitapçığındaki "Affine Equivalence in S-boxes", 45-48 pp., Antalya, Türkiye, Nisan 2006

Sakallı M.T., Buluş E., Şahin A., Büyüksaraçoğlu F., ELECO'2005 konferansı dahilinde "4th Internternational Conference on Electrical and Electronics Engineering" bildiri kitapçığındaki "Differential Cryptanalysis for a 3-Round SPN", 297-301 pp., Bursa, Türkiye, Aralık 2005

Ulusal Bilimsel Toplantılarda Sunulan Ve Bildiri Kitaplarında Basılan Bildiriler

Büyüksaraçoğlu F., Buluş E., İTUSEM 2009 konferansı dahilinde "IV.İletişim Teknolojileri Ulusal Sempozyumu" bildiri kitapçığındaki "Sözde rastsal sayı üretiminin kriptografik açıdan incelenmesi", 125-130 pp., Adana, TÜRKİYE, Ekim-2009

Sakallı M.T., Aslan B., Buluş E., Şahin A., Büyüksaraçoğlu F., ABG'08 konferansı dahilinde "Ağ ve Bilgi Güvenliği Ulusal Sempozyumu-ABG'08" bildiri kitapçığındaki "Aes S-Kutusuna Alternatif Cebirsel Olarak Kuvvetlendirilmiş Bir S-Kutusu Önerisi", 45-50 pp., Girne-Kuzey Kıbrıs Türk Cumhuriyeti, Mayıs-2008

Sakallı M.T., Buluş E., Şahin A., Büyüksaraçoğlu F., ISC 2007 konferansı dahilinde "Uluslararası Katılımlı Bilgi Güvenliği ve Kriptoloji Konferansı" bildiri kitapçığındaki "Ters Haritalama Tabanlı S-kutularının Cebirsel Açıdan İyileştirilmesi", 79-84 pp., Ankara, TÜRKİYE, Aralık 2007

Sakallı M.T., Buluş E., Şahin A., Büyüksaraçoğlu F., Buluş H.N., AB2006 konferansı dahilinde "Bilgi teknolojileri kongresi IV, Akademik bilişim konferansı 2006" bildiri kitapçığındaki "AES S KUTUSUNA BENZER S KUTULARI ÜRETEN SİMULATÖR", 271-278 pp., Denizli, Türkiye, Şubat 2006

Sakallı M.T., Buluş E., Şahin A., Büyüksaraçoğlu F., MBGAK'2005 konferansı dahilinde "II. Mühendislik Bilimleri Genç Araştırmacılar Kongresi-MBGAK'2005" bildiri kitapçığındaki "Aes S Kutusuna Benzer 4-Bit Girişe Ve 4-Bit Çıkışa Sahip S Kutularının Tasarımı", 16-23 pp., İstanbul, Türkiye, Kasım 2005

Sakallı M.T., Buluş E., Şahin A., Büyüksaraçoğlu F., ABG2005 konferansı dahilinde "Ağ ve Bilgi Güvenliği Ulusal Sempozyumu-ABG2005" bildiri kitapçığındaki "Bir Blok Şifreleme Algoritmasına Karşı Square Saldırısı", 92-99 pp., İstanbul, Türkiye, Haziran 2005

EK A: Deneysel Sonuçlar için Kullanılan ve Rassal Olarak Seçilen 20 Adet 128-bit Anahtar

1. 4578EAFC4D5902750B3471F5EDC3801A
2. 12790FCDEFB4569936655472FDEBCA22
3. 334987F1C5A9845800245CD2FB466394
4. A52139FCB4EFA0387692D37742048891
5. 22813498FECBA54012765DC143765BB0
6. 0F4CBA590431738ADE5234671952466C
7. CC1325BADCFFEDAB249712966444821D
8. 6E3543FCD36AB20DEF72891774AA3242
9. 444126FECDBA2396187500652DB14309
10. 2B8CEF7591233BDDFC613F45627139AC
11. FF268100432965978CCDEF1234256D3B
12. 72823000F54213346FC9BA5C73310180
13. 665934FCE24BA873540ABDCF101327A9
14. CC23BD669F015462FEBA682737B37421
15. D8297C4A7BFDE824180917362483B6A8
16. 5F2609ABCD780EDA9872563910013472
17. BF00092CDE8167449ADE527017638BAF
18. EE8A0C4892F1134666653BCD41711085
19. 8120FFCDE45AD4510453156CDFF23BB7
20. 9960FCD316903785C9A34B7259013CA3

EK B: Deneysel Sonuçlar için Kullanılan ve Rassal Olarak Seçilen 20 Adet 192-bit Anahtar

1. 01C45DEF789261F992731002928384BA75A910364435BADD
2. FF45BA6C8D90012267913318347 46B5A7D89A0123ACDEE71
3. C10DF561BA7D9E42919BADEF14367829F4CDE618AB692102
4. B36A891BD65EFF527182945633BADCF18235382B7A812010
5. A510923C4D8E562A8F15729018233BADF34528BA5D6C1E6F
6. 9380017C12A638BD92163455BACC628310274C63ABCDC273
7. D52178934BC35ADEF89F27102839821274BA82EF16829364
8. E9FBACA6229138461233193674BDF267DCC182303872619B
9. 4CD732019648A8B238002BA729CC254391FF81328B7A819E
10. 3372801CE63DFA1269344BAD2582100AFCD2DE7290125519
11. 2F284930A89C6D5B1A5380182037462B5A92735192737211
12. 50178F4739AB6E23BA730193836DA7817ADC37814652BAFF
13. 6A9B4C1D3E7F028371C289F6391BA83D19337471BA63DC38
14. 102F37458BA924C4529A520BD27E8162F374938D921BDA82
15. 882B091ACD364901834BD744F2800355C82AD93BDC28A81C
16. 7B5A8D01C63F382725364AB63CD620027ADDBACDF3720137
17. 63019A7B26D8263FACE72829AAF28101CC72924510F31635
18. CE6301727453B46AD823049B5A72C346D29BB936A7319283
19. F29CC2200183645B24ACD6382BE28927A172548366BAD281
20. 5EF2378102BA634021ADEF17237992BACD256D92F16349A4

EK C: Deneysel Sonuçlar için Kullanılan ve Rassal Olarak Seçilen 20 Adet 256-bit Anahtar

1. 3DCE9237001245296AD273540173FBACD24ED23918624287BAD290AFF01BAD25
2. F183476BACD35923BA739825300ABEDF26A83BC93D253919183614270A93238A
3. 051BA37ADE2583729634BA6293D28CA726359100BA73624D428A9C0D3EF269AF
4. 152CA8F3549882A746AF73BD37D0E37CCDA82F4538ACD369836FD73CB28AE939
5. DE34098DBA5C2D17B5844321CA4D7A09D4B3CA48754C312A69806F43E68A9C7E
6. 225FC5D1076D3C6B28735344AC7B31EF0873A5F7D2AC2B708468215699643CBD
7. 4D5C8B0A2F3AD69C03B84E1DA3A896C20954ABDEF7909743257BA246109F5685
8. C369B46A8C9D3E6FF1901378249DA9BA92C384AA471001BAD37CD037A123759D
9. 5F2CC18DA72845B710D74E83017A3834620CA8927356A63BCD73EFF183387204
10. 62390ADF35CBB3923785619387462B72A63DC4056D834BA34F9734C629E28126
11. BA45D901C3D6E83298B237AA63944519F73F10992CA63D0128364C238DA73301
12. A4679123B9C36DE82B237B38120DD2363425ABDEF39A90347BA7311A02BD381F
13. 77201F369B28ADE8F7351732BC2610386ADD3428B1027810210FCAB125239447
14. 662387B29AD20980AF236CB27D63301259F3452BAC28AF38E29228E83762E261
15. 85B19A7D63C94DF92EA83BC28092745230183474 53C3B26A782D349EFF238923
16. 3BA7309DE73F127ACD7230AE2836445192BC126A83630017A732CAFE5D29122B
17. FAE34602973BCAD2375926100126ACC22375429DEF36391ACB1EADF369BB278C
18. 74BBB338A74DCEF6329001ADEFBAC3913764529D7439378299283746BAD3EF39
19. A73FBCDA6244920026BA5231DAE6FBA9230836512BAD36EFCC3923843BA64DC3
20. 49823600FFE53BCAD36498CA7EBB52918A63B6453AD39C1927012BAE7DCA8F82

EK D: Deneysel Sonuçlar için Kullanılan 128-bit Döngü Sabitleri

1. 9CC35F046A78F579FC42EC142FCDB104
2. 57EF4269CABDDEFC18046066798A03F2
3. 68C9F104BDEFCC692045679836435675
4. F54830446978C9B5E679646209AB9CED
5. DF570249AEFCD51269334589ACDE0F14
6. 9AF5667ED340E69FAB2C4D05328B8791
7. 0F882DCE6524778A29F57E9A5326C5E0
8. 3AB0E25DEFD5B128FF20C8219ECC4AB1
9. DBE0908C4E507749329237 1D2126E970
10. 187A8F4936E38867B5CB27009B152FCD
11. 05679C3EAB8422194FDEF533490ADF13
12. C069778AFC114E637B8C0A29ABC3D4B9
13. 406A7FCDACBE2319DDE3B04951EFCDA2
14. 2A3CFDAB489AE0FFC3776B832FA4D22D

EK E: GRAIN-128 Akış Şifresi'nin Seçilen Anahtarlar İçin Ürettiği Çıktı Değerleri

0001 0111 0011 1101 1000 0000 1011 0011 1000 0000 1011 0000 0101 0001 1010
0000 1001 0010 0011 1101 1100 1011 0000 1100 0011 1001 0001 0100 1001 0110
0010 1110 0000 1000 0010 1011 0111 0010 0010 1000 0010 1000 0100 0010 0011
0111 1010 0001 1011 0001 0111 1010 1101 0100 1000 1100 0000 0110 0011 0000
1001 1100 1101 0011 1110 0101 1011 1101 0000 1111 1001 0000 1011 0011 1001
0000 1011 1111 1111 1010 1011 1001 1101 0101 0111 1111 0110 0110 1100 1000
1011 1010 1001 0100 0010 0101 0110 1001 0101 1101 1001 1100 1011 0010 0010
1111 1010 1001 1011 1111 0110 0101 1111 1111 1111 1101 0111 0100 0111 0010
1000 1011 1000 1001 0001 1010 0111 0101 0101 1101 0100 1110 1101 1001 0110
0000 0001 1111 1111 1100 0111 1100 0101 1110 1101 1110 0011 0000 1011 0100
0011 1011 1110 1101 0100 1001 0111 0101 1000 0001 0101 0110 1101 1011 1010
0000 1001 0000 1010 1101 0001 0010 0110 0111 0110 1011 1111 0110 0111 1101
0110 1011 1001 1010 1011 1010 0010 0000 1110 1110 0011 0010 1001 0001 1001
0111 1001 0000 0010 0110 0110 0100 1010 1011 1110 1110 1000 1101 0111 1110
0010 1111 1011 1001 1000 0100 1100 1010 0010 0100 0111 0010 0100 0010 0110
0101 0101 0001 1111 0001 1001 0111 1101 1001 0101 0011 0001 0000 1111 1110
0101 0110 1001 1010 0111 1001 0001 1111 0101 0100 1000 0111 1110 0011 0100
1111 0101 1111 0001 0010 1000 0001 0100 1110 1001 1101 1011 1011 1101 1001
0110 0101 1100 0100 1111 0001 1101 0010 1100 0001 1001 0101 1100 0111 1100
0110 1011 1011 0111 0101 0000 0111 0101 0011 0101 1010 1010 0001 0100 0100
1101 1001 1010 0110 0010 0110 1011 0101 0111 1011 0011 0000 1100 1111 0101
1111 1000 0010 0111 0101 0110 1000 0101 1010 0110 1110 1101 0110 1101 1111
0001 0111 1100 1000 0000 1111 1110 0001 1011 1100 0111 0010 1110 1001 0100
0011 1010 0010 1100 1111 0100 0100 0000 0001 0000 1110 0001 0111 0011 1111
1011 0100 0110 0010 1010 0000 1101 1011 1101 1001 0111 1010 0001 1101 0100
0101 0111 0001 0100 0000 0111 0011 1101 1100 0100 0001 0110 1001 1010 1110
1110 0000 1101 1010 0101 0111 0011 1101 1010 0100 1000 0000 1011 1110 1111
1110 1011 1000 1010 0100 1000 0101 1111 1010 1110 1111 0011 0011 1000 1001
0111 1010 1001 0110 0011 1110 0111 1001 0101 1110 0010 1010 1110 0011 1111
1111 1110 1011 0011 0101 0011 1000 1101 1011 1101 0010 1010 0110 1101 0010

0001 0100 1010 0001 1010 1111 0001 0111 0010 1100 0100 0001 1011 1000 0000
0111 1000 0111 0100 1110 1010 0110 1111 1100 1111 0100 0110 1101 0000 1001
0101 1111 0001 1101 0100 1100 1001 0101 1111 0000 1001 0101 0101 1100 1101
0110 1100 1000 0000 0101 0000 1111 0000 0000 0101 1001 1101 0100 1001 1011
1010 1010 1101 1110 1110 1011 1011 0010 0001 0001 1011 1110 1010 0110 1011
1110 1111 0011 0111 1001 0011 1000 1001 1011 1110 0100 0110 0010 0010 0100
0001 1001 1011 1111 0111 1010 0001 1010 0010 1000 1100 0101 1110 0011 1010
0010 1000 1110 1011 0000 0011 0001 0000 0110 0101 1111 0110 1111 1011 1010
0111 1101 0111 0010 1010 0101 0100 0111 1111 0011 0110 1100 0001 0111 1001
1100 0000 1000 1111 1011 0000 0100 1000 0111 1110 1111 0001 0010 1010 1011
1011 0101 1010 1111 0100 0011 0000 1101 1001 0100 0001 1011 0100 0110 0111
1010 0001 0101 1101 0111 1100 0011 0101 1001 0110 0101 1100 1111 1110 1110
1110 1010 1010 1101 1000 1000 0001 1100 0000 1011

EK F: MICKEY-128 Akış Şifresi'nin Seçilen Anahtarlar İçin Ürettiği Çıktı Değerleri

0001 0111 0011 1101 1000 0000 1011 0011 1000 0000 1011 0000 0101 0001 1010
0000 1001 0010 0011 1101 1100 1011 0000 1100 0011 1001 0001 0100 1001 0110
0010 1110 0000 1000 0010 1011 0111 0010 0010 1000 0010 1000 0100 0010 0011
0111 1010 0001 1011 0001 0111 1010 1101 0100 1000 1100 0000 0110 0011 0000
1001 1100 1101 0011 1110 0101 1011 1101 0000 1111 1001 0000 1011 0011 1001
0000 1011 1111 1111 1010 1011 1001 1101 0101 0111 1111 0110 0110 1100 1000
1011 1010 1001 0100 0010 0101 0110 1001 0101 1101 1001 1100 1011 0010 0010
1111 1010 1001 1011 1111 0110 0101 1111 1111 1111 1101 0111 0100 0111 0010
1000 1011 1000 1001 0001 1010 0111 0101 0101 1101 0100 1110 1101 1001 0110
0000 0001 1111 1111 1100 0111 1100 0101 1110 1101 1110 0011 0000 1011 0100
0011 1011 1110 1101 0100 1001 0111 0101 1000 0001 0101 0110 1101 1011 1010
0000 1001 0000 1010 1101 0001 0010 0110 0111 0110 1011 1111 0110 0111 1101
0110 1011 1001 1010 1011 1010 0010 0000 1110 1110 0011 0010 1001 0001 1001
0111 1001 0000 0010 0110 0110 0100 1010 1011 1110 1110 1000 1101 0111 1110
0010 1111 1011 1001 1000 0100 1100 1010 0010 0100 0111 0010 0100 0010 0110
0101 0101 0001 1111 0001 1001 0111 1101 1001 0101 0011 0001 0000 1111 1110
0101 0110 1001 1010 0111 1001 0001 1111 0101 0100 1000 0111 1110 0011 0100
1111 0101 1111 0001 0010 1000 0001 0100 1110 1001 1101 1011 1011 1101 1001
0110 0101 1100 0100 1111 0001 1101 0010 1100 0001 1001 0101 1100 0111 1100
0110 1011 1011 0111 0101 0000 0111 0101 0011 0101 1010 1010 0001 0100 0100
1101 1001 1010 0110 0010 0110 1011 0101 0111 1011 0011 0000 1100 1111 0101
1111 1000 0010 0111 0101 0110 1000 0101 1010 0110 1110 1101 0110 1101 1111
0001 0111 1100 1000 0000 1111 1110 0001 1011 1100 0111 0010 1110 1001 0100
0011 1010 0010 1100 1111 0100 0100 0000 0001 0000 1110 0001 0111 0011 1111
1011 0100 0110 0010 1010 0000 1101 1011 1101 1001 0111 1010 0001 1101 0100
0101 0111 0001 0100 0000 0111 0011 1101 1100 0100 0001 0110 1001 1010 1110
1110 0000 1101 1010 0101 0111 0011 1101 1010 0100 1000 0000 1011 1110 1111
1110 1011 1000 1010 0100 1000 0101 1111 1010 1110 1111 0011 0011 1000 1001
0111 1010 1001 0110 0011 1110 0111 1001 0101 1110 0010 1010 1110 0011 1111
1111 1110 1011 0011 0101 0011 1000 1101 1011 1101 0010 1010 0110 1101 0010

0001 0100 1010 0001 1010 1111 0001 0111 0010 1100 0100 0001 1011 1000 0000
0111 1000 0111 0100 1110 1010 0110 1111 1100 1111 0100 0110 1101 0000 1001
0101 1111 0001 1101 0100 1100 1001 0101 1111 0000 1001 0101 0101 1100 1101
0110 1100 1000 0000 0101 0000 1111 0000 0000 0101 1001 1101 0100 1001 1011
1010 1010 1101 1110 1110 1011 1011 0010 0001 0001 1011 1110 1010 0110 1011
1110 1111 0011 0111 1001 0011 1000 1001 1011 1110 0100 0110 0010 0010 0100
0001 1001 1011 1111 0111 1010 0001 1010 0010 1000 1100 0101 1110 0011 1010
0010 1000 1110 1011 0000 0011 0001 0000 0110 0101 1111 0110 1111 1011 1010
0111 1101 0111 0010 1010 0101 0100 0111 1111 0011 0110 1100 0001 0111 1001
1100 0000 1000 1111 1011 0000 0100 1000 0111 1110 1111 0001 0010 1010 1011
1011 0101 1010 1111 0100 0011 0000 1101 1001 0100 0001 1011 0100 0110 0111
1010 0001 0101 1101 0111 1100 0011 0101 1001 0110 0101 1100 1111 1110 1110
1110 1010 1010 1101 1000 1000 0001 1100 0000 1011

EK G: SOSEMONUK Akış Şifresi'nin Seçilen Anahtarlar İçin Ürettiği Çıktı Değerleri

01011101 11101100 10011111 00011011 10011010 10111000 00001110 01100001
11111101 10000101 00011100 10010010 10011101 11011010 00000110 00110111
01000011 11111111 00011010 10001011 11101010 10111101 10000001 11010100
01101110 01010101 10001000 10000111 11111010 01100000 11010110 101001010
1111000 01101000 00111100 00001001 01000110 00001110 01011111 11001000
11100001 00110000 00100000 01111001 10101110 00101111 00001000 01010101
11101101 00101111 10010110 00010100 11101010 11101011 11111100 11011111
10101110 10110001 10000010 11010111 00100101 10101001 01110110 10010100
10111000 01111111 10001001 10011010 10011001 00001000 11110010 10100101
01001001 01011000 11101101 11001100 00011010 00010011 10010000 10001101
01000110 01011011 11011001 11010011 00000001 00110011 00110100 01100011
00000001 01110110 01010100 11100010 00111000 00110011 00101001 10110011
00011001 10110101 01100101 11101000 00100110 00010000 11111100 10110111
00110001 01110100 01111000 00100011 11100100 10101100 10001000 11111011
10011000 11000110 01010011 10110000 01011000 01100101 01000001 00100101
01010001 00010110 10101000 01110101 01101100 01010111 00101000 10010111
10011101 01111111 10110011 10111111 10111101 01100100 11111001 10010010
11100101 01010000 10100011 10011011 01010010 10101100 10011101 11000111
10010110 10110000 00001110 01001110 01101001 11001000 11101010 10001111
11010010 11000011 01111100 10101110 00100011 01101011 01001000 01000010
10001111 11010100 00000100 01110100 11010101 11010101 10010001 01010101
11010011 10111011 01011110 10110100 10101010 00000111 00101110 01111110
11010010 10100000 00011101 11111111 10101011 10011010 10000000 01011111
01100011 01001001 10100001 01111100 11111101 11011000 00111001 00101101
01100101 01001000 10111111 01011001 00100010 01001001 10100001 00100000
11000011 01000000 00111010 00001011 00111001 10110101 10110001 00001000
01000101 00010100 11010010 00110011 11000011 01100100 01111111 10110001
00010000 00110111 01110000 00010111 00101011 01100110 11000001 01000100
10101111 11001100 00001100 00001101 00010101 01100111 00000100 00011011
00010000 01011101 10111000 10010000 10010100 00000010 10110110 10110101

01111000 01100110 00110011 11010000 01001010 00001100 00111100 11011011
10100111 01110010 11000110 11101101 10010111 00000001 00111011 10101110
11010010 00111011 00010100 00001110 10000001 10011111 11110110 00110001
01101111 01110011 01000001 01101000 10100111 00001010 11011011 01011000
01001110 01101110 11100111 00101010 11000001 11100100 11010101 10001111
10001100 00111110 01110111 01011010 01010001 00010001 10101101 01010101
00001001 01010110 00110111 01110011 01010101 00011001 11110101 00001111
10010010 01100010 00011010 10010000 00001010 01010011 01010001 10101101
01111110 11111110 01100111 01010111 11011011 11111000 01011010 11011100
10011001 11011110 10101011 11010111 10110011 11111000 01001000 11101101
11010011 00101111 11100011 01000000 00010011 01101111 10111111 11010111
10111000 00111011 11011111 01000010 00110001 01000101 10001101 00000100
01100001 11011001 01100001 11100110 10000011 10110110 10011110 01000111
11110101 01011000 00101101 10100001 11010100 10100000 01011111 01111100
00010110 00000010 11101110 10100110 11011100 01010011 00101011 10010110
10000001 11000000 10010110 00011001 01001010 01000101 01100110 01110001
00100011 00001111 01000001 01011101 11101010 00111010 01001000 11100000
00001010 10101110 00110011 01111111 10010010 10010000 00010011 00101110
00100010 01001111 00001101 00110000 10001110 11000100 10110100 10000100
00010100 11010001 01110111 01101100 10000011 11011000 10100101 10001000
00010000 10010001 10001010 00111100 00101100 10001000 10101100 00110011
10110100 10110101 00000100 11101100 00001000 10110111 00101111 11111101
11100010 01110100 11011111 01011101 00110011 10011000 11110000 11011110
10111000 00100000 00011101 00011101 11010100 00011011 01101111 00010100
01011000 01111110 01000111 00011001 10011100 10001111 10000010 01101111
01111001 01011100 01101001 10111010 00000111 11000010 01101001 00001111
10000101 01000100 10010101 11011100 01011110 01010100 10011010 01110110
11000011 01110010 10010001 11011111 01100110 01111001 11101100 10100111
00000010 10010000 10011000 11110011 10110101 11000001 11111110 01111111
10100000 00011101 10111100 01010100 00011001 10101001 01010111 01111011
10100001 11110100 11100100 11010101 01101001 11101010 00111110 00110001
10100101 11001011 00001100 01110111 00100100 01110000 01110100 11110111

11110001 01000000 11100000 11000110 10011011 11101111 11101100 10100111
01000101 11010111 11100101 01000001 00110100 10111010 10111100 10101011
11110011 01101001 11100111 10110001 10110110 01010111 10101100 11100101
00000000 01001000 01000001 00101011 01100110 10000111 01010001 01001100
10010001 00001001 11100001 11111001 10100111 01000011 01101010 10100000
01001010 01011011 11001010 00101110 10101101 10000111 01010001 10101011
11011000 00100011 10111000 01010011 10011001 01000101 11011000 10110101
11001111 11000011 11101000 10010010 11100100 01111001 10010111 01101111
00001111 00001100 01000011 00100111 01110111 10001110 01100011 10101010
11000000 11000111 11010100 01000011 11110110 01111011 01100001 01111000
11110111 10100110 11111000 11010110 10101101 00101000 01010011 11110011
11101100 00010100 11110110 11110010 01101011 01010110 10001100 10001100
10100110 11011001 10001110 11011110 10000101 10010110 01001011 01100101
10000010 11001101 10001101 11000001 00111101 01010001 11100101 10001000
01001101 11010010 10101000 01000101 10000100 00010010 10110001 10111000
10000100 00110101 01000011 01110001 01100100 11100001 11101011 10001011
00101001 11001111 10000011 01101011 01010101 00001010 01010000 01101011
00011100 00001101 00100111 01101010 01000010 00101110 01110100 01010101
10000101 10001010 11111000 01100000 11001100 00101010 00111000 10111101
11111111 10110011 01110110 00100000 00111100 01110101 00001010 10100010
10100000 01010011 10101001 10110010 00100011 11000010 00101111 00110111
10100110 11001000 00001100 00100111 11001011 10011000 10000101 11101110
10001001 11110110 10011100 00101011 10000110 11110000 00100011 10100110
10001100 00111101 01001010 00001100 10101011 11011111 11101001 00110000
11010100 10100100 01111101 11001011 10100101 10101110 11010100 01111110
00010000 00110101 00100110 00111011 11110111 01011110 11000000 01110100
00111000 10100010 00001001 10100100 10100001 11011011 10001000 01101001
11010000 01010000 10010101 11100110 11111001 11010010 01100100 00111110
01111001 00110001 01111100 11101111 11111111 00100000 10110110 00001010
11101001 10110111 11100010 11101000 01011101 01000001 00101110 01111111
10010101 00100110 10001111 10011110 00000001 10010001 10001110 01011011
01100101 00111101 01001011 10100101 11001001 01111101 00111011 11010000

00110100 10111000 00000100 01111001 01110001 11101010 11100110 00101110
00011111 00101111 01001111 01100011 00110000 00100110 01110010 11000101
10011000 10100101 10000101 01011011 01101100 11110100 01101110 10000100
01100000 10010011 11011101 10101110 11010100 10011111 01011100 01001110
00101111 00110000 11011100 01000001 11110000 01011100 11011100 10000111
10101111 01010011 10111111 10100000 01001011 10000101 11000101 01111011
01100110 10011110 00101001 11101111 01000000 10001110 11001101 00110110
10110110 00010001 00010110 00000111 11101101 00010101 00001101 01001101
11000111 11000011 11110100 11010001 01001101 11000011 00011001 01000111
10001100 10000000 01000101 00000001 00100111 10010100 00011111 10100100
01110110 00001111 00011110 00010001 00011011 00000101 00011110 01110111
10100100 10100111 00101101 10000101 00011010 01001110 11010101 01100100
11000010 11100001 00100001 10011111 00111100 00111001 11001000 11111110
11011001 01001101 10101001 11010001 01010101 11001110 10110000 10100001
00010101 10111110 11111100 00101001 11001100 00000100 10101100 11100011
01111101 01110110 01010100 01001011 01101100 11111100 00101101 10001001
01101001 10001110 10000010 01010111 01110100 01011111 01111110 01101110
01100101 11000111 11010000 00010000 01101000 00100000 10100110 01000011
10010100 11000001 01001010 10010101 00101110 11010010 01010101 11101001
00100101 00110100 10110001 01011110 00001110 11000000 10010111 11101101
11011001 10001001 00000111 01001010 01001111 01101001 10001111 11100111
00011110 10111001 10011000 00101100 11000011 01101000 01011111 11011110
11001011 10010010 00001000 10101001 00001101 10111000 11001011 00001111
01100011 01110011 10100110 00011000 00110110 00001001 00110100 10011010
01011000 11010010 10100000 11100110 00100111 11011101 01100010 01011000
01110000 10000010 11000001 10010010 10101000 00000001 00111111 10111010
00010000 00111001 10011000 11000110 01100101 11011011 00111101 00010000
11000000 00010110 01101100 10011101 11100101 00000100 00011000 10010011
10011101 00010000 00011010 00100011 00000000 01010001 11101000 01010000
11111101 10111100 10101011 11000001 01110000 00111010 00100000 01101100
11110110 01010100 11011001 11011000 01111000 10010011 00111011 10011010
01000010 01101100 11110001 10101001 01110011 11111011 01010110 00111010

11011100 11111000 00001111 11111101 01001111 10101001 00100000 01010100
01001001 10110101 01011001 10001110 00100101 10100111 01011001 00100001
01111101 01011000 11101110 11111001 10101110 01001111 11100010 01010000
00100111 11101100 01000011 11100010 10100010 00110000 01011110 11111000
11010000 01111101 00011101 10000011 01110001 01100011 01011000 01011001
01001100 00110101 00000111 11100110 11101110 01110101 01001100 01110001
00001100 01110110 10000000 01000001 01011001 11001111 10111001 10000010
10111010 11011000 00101111 01010001 00101001 10011100 10111110 01010000
01011011 10101101 10101010 11001111 10110110 00000000 10011100 10110100
00010001 01010001 00100111 10010011 00110010 00100001 00111001 10000101
00010111 10010010 00000000 11010010 11111110 11101110 10100001 10101110
10001010 11110000 10010011 00110111 01110001 01100011 11000101 00100010
01001110 10010010 11010000 10000001 10011001 11010100 10101011 10101000
10011010 11001110 10000110 10101111 00111101 00111111 11100100 10011010
11110111 00111110 01000001 11001101 10001110 00110000 11011001 01101010
10110000 00100111 10010101 01010000 01110011 11000110 00001010 01101110
01110110 01110101 01101010 10010110 11101010 11000010 00110110 01100101
11001000 11000111 00001001 01111011 10100110 10010111 00001111 01110010
01011000 11101111 01100101 10010100 00000110 11100100 10001100 00001110
00001100 10001100 11100000 00011111 00100010 11010100 00100000 10001010
11110011 10010100 01110101 10000010 00010110 01010101 10101001 00001100
11101010 01110111 11001100 10001110 11100111 00101100 01001000 00100001
00001101 11010000 00000011 01100110 10110000 11010100 11010111 00010100
01110111 10100010 10011111 10010110 11111100 01011010 01001011 01010001
10110100 01110001 01000110 01001110 00101111 10010011 00001001 10001000
00111001 00010101 01101011 10111110 01000101 11000001 10011101 11101001
00110111 00100100 00101101 10010001 10101010 01000110 11101011 11011111
10110010 10001100 00000001 01101011 11111100 11010010 11101000 00000101
01001010 11011101 00010101 11101000 11100111 10111110 01111010 11111110
01000001 00111001 00010100 01000000 11000001 10111011 01110011 10011100
01010100 10000101 01010111 10000010 00010111 11111110 01100101 10011111
01101101 11110111 00010010 10110010 00011010 11010110 10011011 10111010

10110111 01001011 01101100 10100101 01110101 00010110 11010010 01110010
11101000 01010001 00101111 01001000 10111101 11100100 11011101 00100110
00001000 01001011 11111111 01011100 11110011 10001001 01011110 10001011
01101101 10101110 00110111 11011111 10001011 00010110 11000001 01101010
11110101 01001010 00010100 01010011 01001011 11001111 01100101 00101011
00011101 10010000 01000100 10000000 10000010 01010110 11010110 00111100
00011010 11111001 01111001 10010101 10000110 00100010 00110000 01010101
00110111 10000001 11111000 00001110 10001000 10010000 01101000 10111110
11000111 10101111 00101110 10010101 10001111 00001101 10100011 11110110
01000101 00011100 10010000 01100101 11101001 11110000 00110101 10001001
01011000 10110011 10010110 01110010 10101110 01110000 11100000 11010001
10110010 00000010 10001100 00100000 10000111 01101101 01100010 10011000
10111110 01001110 00100110 11111101 01001000 01011110 01111111 11101110
11101100 11011110 11001000 00011000 10111001 11101111 11110100 10011101
01011100 11011110 10111010 11000001 11011001 01011000 00000001 01000011
11110011 00011101 10110011 00101101 00111010 00110010 11011000 01000000
11110111 00001100 00101101 10110000 01100111 01000001 11101111 11001001
01010111 00101101 00110100 00110011 11110110 11111101 01110111 00001011
11100001 11010001 01011010 01000110 10110010 11001010 01101101 00110000
10100011 10010010 01110100 01010100 11010001 11000010 10001100 11001101
01101011 01011011 01010101 10101110 10000111 00011110 01011111 11000110
01110101 11001101 11010111 10000000 10011101 11100111 01100101 00111010
01010100 10110100 01001001 10100001 01011111 10100100 11010011 00010000
00110001 11010010 10101101 11110101 00111101 10011010 10100100 11111000
10001010 11010101 11100111 01100111 11110110 11111100 10011101 10010101
01101101 10110100 01011011 01110011 00010001 00111000 01011100 01100011
00001111 01010111 01011100 10111000 00110110 00011001 11111101 01101010
11100000 00010101 11010111 01000000 01111010 11100101 01100000 11111101
11000001 00100110 01000010 10011000 00000011 00011000 10001000 11000001
10111101 10101010 00110011 01111111 00110100 10000100 11011100 11101110
10011111 00010110 01111011 11101100 11010001 10000110 11100111 11011101
00110001 10111101 11010010 00010110 11001010 01110101 11001111 10010101

11111110 10011010 00000000 11100001 00000011 11111111 10101000 11101010
01111100 00111010 01110100 00011100 11101110 00000010 00001110 00110101
01101110 11110110 01001001 00110010 11010001 01110010 11100101 11110010
01011010 00111110 10011011 11101000 00011001 00110001 01000011 01000101
00100100 11010101 00010001 11001010 00000110 11111100 11101001 00010011
01101011 11010010 01110010 00111000 11011110 11010100 01011110 11101010
00001100 10001101 11001011 01001011 01110110 00000011 11111011 00000111
00011000 00010100 11000110 00011101 01010000 10001011 10010011 00000100
00001111 11101000 00001101 11001100 01000000 10011101 01000010 01101011
11100010 10100000 11111111 11100110 11111100 00011111 01110100 10110100
01110101 10110011 10110100 11110101 11101010 00110101 10010010 10000001
00101010 01110100 01110011 11111101 00011101 10001001 01001011 11001110
01000111 11011100 00100000 00010111 10110010 10111010 10001000 00010110
00111111 10111011 01011010 11000110 01000111 11011101 00001111 01111101
00100100 10001011 01111101 10101100 01110000 01100111 00011101 10011110
10110011 01101000 11010001 10001111 10001101 00010000 00111000 11110010
10110111 11101110 00000011 01101100 00100110 10001011 00100100 11100000
00000000 11001010 11101001 11010010 10011111 10010010 01101011 11000000
10111111 11001110 01101010 00001110 11001110 01111100 01001101 10000001
10110011 11100000 00101101 10111001 00001011 11110110 00101110 11001011
11101011 11011001 11100001 10101010 10000000 01111001 10100110 11010101
11100111 10100110 10010111 01101101 01011110 01110000 01000100 00101011
11101011 10100110 00010110 11010011 11010111 00001100 01110010 00110111
01000001 11011010 11011101 00011111 01011100 01110100 11011110 01011100
01001010 01101111 01110110 11011010 10000100 00101001 11100000 10101001
10101100 00000010 10110001 00010001 00000010 11011111 10111110 11000010
10111011 01001110 00011001 10001011 01101110 01110111 00100011 01000001
11001110 11001100 00011010 11000010 11110010 11000111 11001000 11101001
11010011 00101100 11010100 11000010 10011100 11010111 11101001 00000000
11111100 00001001 11100100 11011011 01100010 11000000 10100011 01011110
11000101 01000001 11000100 11101011 01100110 11110000 00000100 00001100
10111101 11111000 11011001 00010011 01000111 00011111 00100101 10110110

11001111 01111000 11111110 01101010 00011010 01100101 01110000 10011111
00001000 10111000 01110111 10000011 11011010 00010010 00110101 01011000
00111111 01010100 10001100 11100110 01000001 00100000 11110111 11001100
11001100 10111011 10111111 10110111 10011100 00000001 01000100 00001001
01100101 00111100 11000000 01101001 01010100 11000101 01011100 00111010
01010001 10100000 11010110 00011111 10101011 10000101 10111011 11000001
00010011 01101110 11101001 11000101 01110111 00000011 00100100 01011110
00100110 11000011 00011000 01100110 00011011 10111100 10101001 11101100
01001110 11111010 00011110 10000111 11111011 11010001 11000110 00000101
01100010 00010011 11100100 11110111 01101100 01010110 10101100 11111111
11110010 01111001 11001100 00001010 01010011 11010000 01100000 11111000
10010001 00100100 01000101 00100100 10101001 10011000 11110000 11001110
01111000 00010001 11100011 10011001 00000101 11000010 10011010 01111101
10111100 10011101 11001111 01100001 00101100 01011011 10010100 00101101
10001010 01000100 00010100 00011100 00010011 00110011 01010101 00101001
00000101 10110100 00011001 00100101 10011001 01111010 01011100 01101001
01101011 10101001 00101001 10000111 00001110 00100111 01101111 00011101
10111000 11100110 00110111 00101001 01110000 01010100 10000101 11110011
00010000 10101101 11010110 00011111 00110101 11010101 11110011 10101101
11100101 11011000 10001111 00011100 00010010 00001010 00010011 00000010
01100100 00011100 01000010 00100111 10111000 10000111 00001001 00011010
10001011 11010011 11110101 10000010 11000000 00100000 01101001 10000101
11110000 01010101 00000101 11001101 00001111 00001011 00000101 11001110
01000001 10001001 01100010 01010101 01011111 10100001 10101011 10011011
11001110 00111111 10101010 00111010 10101000 00011111 11101100 00010011
11100101 00110101 00100100 10000001 11001000 11111001 01110101 10010111
00011101 01011111 11101101 01111001 10100010 11110110 11111101 11110101
00000001 11000100 00111011 00010111 01110010 00000101 11011111 11010111
00111100 00001100 11000111 11110001 11010010 10100101 01111100 00000100
00011111 11000001 11010001 11001000 11111010 00110010 00101101 10000001
10100110 01010001 10101011 01001101 00011110 00001110 10000100 00011011
00010110 10111001 00011110 10101000 10110111 11110100 00110110 00100011

01011011 00000101 01011010 01000101 10101101 01001101 00001011 00010110
10111110 00011001 01110010 01010111 10001000 11111100 10110011 01110001
00100110 10001000 00100111 00010111 11011111 01110100 11000111 10001110
11101100 00111011 00010101 00010110 01000010 11001111 01011000 00111010
00101101 11101001 00010110 01111010 00111111 11001111 10011110 00111001
00010001 11111000 10100100 10010010 10001000 10110011 01100111 11000111
00001101 10100010 01101011 00001110 01100011 11010011 11100001 11011000
10100101 10100011 11001010 01100011 11111100 11011001 00000011 11100111
10011001 01010011 01000010 10111101 10111111 11110110 10010101 11010001
11000001 11011000 11100111 00100011 01100111 10010001 00110101 00011001
00110100 00000001 11010111 00001101 00110110 00010011 01101010 11010111
10100000 11011101 10111100 11100101 01000010 01110101 10101101 01110110
10100000 10111011 11001111 10101011 00010010 01101011 11011010 11011110
10000100 11010101 11010100 10010110 01101000 01101000 00101000 00010001
11101111 11010111 00000000 01001010 11100100 00010000 11111100 11011111
11001110 01001001 10000011 00111101 00000110 11010001 10111000 01001101
01000000 00101111 00100000 11101101 00000110 10101100 01011010 01110000
01100011 11110111 00111100 01000101 01011100 10001001 01011111 10100110
01100000 11110101 10100011 10100000 10010001 10010011 11100010 10111101
11010001 10011100 00001100 00100000 01100000 10111111 00000011 01100011
01101011 00110100 11010000 00010100 00011111 10001111 00101100 00110110
00101000 11101011 10000010 00101110 10011100 10111011 10011110 01101000
00000001 11011100 00101010 11011110 00001010 11100111 10000000 00001001
01111101 11100100 01010101 01111001 11001111 11000011 10110110 10010011
11000001 01011110 01011011 00001000 10000000 01100000 00000111 01000000
11111000 11000111 10101011 00000001 10011101 01000111 00010110 10010111
11000100 01111100 01110011 01011010 10011100 00001110 10000111 01001111
11000100 10011001 11010011 10111110 11011111 10100110 00010010 10001010
00101110 01111100 00010011 00001010 11010111 11101111 00000010 01111001
11010011 00011001 11000001 11011100 01011110 11011100 01101110 11011001
01111001 11011100 01100111 01100010 10011000 10011111 11011011 10001100
10010011 10101111 01000111 00111110 01011110 10100110 01000000 10011101

11010000 00000101 00100100 01010101 01001000 00101110 11001010 11000100
00110001 00110011 01101101 00101101 10110101 10100100 10111110 11110110
10101110 10001100 01111001 00010001 00111010 00101000 11000110 10010000
10011101 11101101 00111000 10100011 01010110 10100100 11000110 10110110
10101111 11101001 00110011 10110101 11101000 11100000 01111011 11011100
00000101 11001010 00111110 00001001 11101110 11000100 00001110 10100111
00111011 11111110 11111100 10000110 00100100 10110000 11110111 00101100
01111001 10110110 10010111 00000100 10110100 01100000 11011000 01111011
00100000 01001101 10111110 10011100 11010100 01100000 11110010 10100111
10111110 11110001 00011010 11000100 00100100 01100111 11100001 01011011
00101011 10010011 00110001 10000100 10111001 01100000 10010110 11111010
00111000 00001110 00110010 00111100 10110001 11000011 11011111 11100011
01011111 10111011 00011011 11001101 01001110 01111110 11110100 00011001
10001101 01111101 10100011 10101101 11001010 01101011 00010001 00000010
11101000 10010011 10101111 01111111 00100011 00011011 10010100 11111001
01100110 00110000 01100101 10011110 01010111 01101111 10110101 01110001
00100111 10101110 11110111 10110110 01011100 10011010 11011001 11110001
01000110 00101000 00000101 10010111 11110000 00001101 11001100 01001001
10110001 10111010 10000010 11010100 11001011 10100001 01001000 10100011
00110111 11101010 11011101 11110100 01010010 01100101 00001100 01101100
10101111 01010111 11110011 00111010 10010100 01111001 10100001 10001101
11010100 11110111 01010111 01111011 11110011 00110111 00100000 10000100
00010011 00111010 11101101 01000100 00110101 10110001 00100100 01101110
10110001 00110110 11010110 00111101 10001000 11101110 01011011 11111100
00100010 10001001 00101110 11001100 01100110 01010110 01101110 11111101
00011010 10101001 00011101 00010100 11110111 10011000 00110101 00110011
10011001 10011011 00010010 11101100 11010011 10001111 01101110 01110111
00001110 11110111 01100001 11001000 00011100 11011001 10100110 01101001
01010111 10100101 10110100 01111111 01000100 11011110 00110000 10010010
00010111 11110100 01100001 01001110 01000000 11100111 01101100 01100000
00010011 00100000 01100111 01011000 01001100 01011001 10110000 11111010
00101000 01000001 11111101 10110000 10010011 01111100 01100110 10111111

00101100 10101110 00111011 01101111 00011110 00101000 10001010 10111101
11011010 10110010 00100010 11000011 10110111 10001010 01011000 10101011
01111011 11000001 10001100 01111110 11111111 01000111 10001111 00110010
01010101 11001010 11010011 01000010 10000001 11111000 00000111 11110010
10001010 01100101 01101011 10010011 11111111 01010110 10111010 11000101
00100111 00000111 00000100 01110010 10111110 11000000 11100110 10001000
11111010 11101101 01111000 01110000 11001100 00100111 11110011 00111001
11100010 01100001 00001011 11010000 00000001 00110011 01010001 01111110
00010010 01100100 11010001 00000100 10100100 10110100 11101100 10110000
00010111 10010000 00111000 00110011 11110110 01110001 11000111 11110101
10000010 01010111 10110011 10111111 01001011 00010000 00010101 10101110
10110000 11010010 00101001 00110001 10000100 11110100 11111010 10011000
10100111 01010111 11000100 00000101 00100110 11011000 00110111 01001100
10001111 10100110 11110000 10000000 00110111 10001110 11101111 10010000
11011111 00110000 11111001 11101000 01100100 10001110 10100101 01111111
10100000 01011000 10110110 10111101 10010110 01101010 10101001 11010010
01111100 00110100 00101001 01010110 01101010 11001010 01011001 10010011
10100111 01001101 10111110 01001011 10111000 00111010 11001111 01101110
11000101 11011101 01001111 11000111 11010100 11001010 01100101 10100101
10101011 11010010 10110100 01010001 10001101 11010001 11011111 01100011
10100010 01111111 00000111 11011001 01111100 10001110 10000111 11001101
01000001 11101010 01001001 01001000 00110000 01111001 00101111 01110000
01010100 00100100 10010010 01101101 11110111 11000101 10011100 11011100
00000101 00100100 00110001 00001001 10111110 00101000 11011010 11001011
01010110 11011011 01000100 10001101 01011111 10001000 01001100 10001011
10100100 11001110 00000010 11000011 01011011 11001111 11110110 10010100
10010011 01001000 10111100 10110101 11111011 00010011 11001001 00011101
10111111 00101110 01010001 01010111 00001010 00110101 01010011 01101100
00011111 01100011 10000110 11011000 11010000 10111110 00101100 01111001
00000101 00011000 11001110 00101101 00100101 01010101 01011010 11100011
00000101 10111101 10000001 11110011 11100100 11111010 00011101 01011001
11110110 11101110 10110011 11001110 11101011 01011000 10000010 01011110

11001001 01011010 11101101 10010000 11000110 01001011 10111010 10001001
01010110 10110000 11000001 00010001 01000111 10000110 01011010 10100000
11010000 10101010 00000010 11100001 10011100 10110110 10000010 10101101
01001101 01111111 00001110 11110100 00110011 01000010 10011111 00010000
01110101 11010010 11011111 10000010 11011101 11010011 10010111 01100111
01111000 00011000 00000010 01000010 00101100 10111101 01011110 01011111
10100011 10010010 10110000 01110001 11011111 00001010 10101000 00110001
10111011 10101010 00110010 10111000 00010100 00111100 11001000 01011110
00010000 01100100 00001110 00001111 10100010 01000111 00101100 00011111
00000101 01010100 00001101 11100100 10100101 01110101 00011111 01100010
11100101 00010111 10011101 11101110 10010110 11010111 10001100 11010101
01010110 01000100 00000000 11001111 01111100 01001111 00101010 01011000
10011111 11000000 10110111 01000110 11001001 00000111 11100101 01101011
00000011 00011111 10000101 01000100 10000000 10010101 10010110 11010000
10101011 00110001 10110001 11010010 00101001 01100010 00000101 11110100
00001110 00111100 11110101 01000000 00000110 01000000 10001010 11111010
10001101 01111101 11011110 10100011 01000110 10101000 00010011 00111101
11100011 01011001 11111000 10000100 11001110 11000110 11100100 11100011
10110111 01101110 10010111 00011101 01110011 00011011 10101100 01001001
11011000 10101110 01100010 00111000 11000000 11101010 01001001 01000011
00110001 10101110 00011001 01011011 01101001 00010000 01111001 01100011
10111100 11000011 10001000 10000110 00010001 11101100 01110011 10101000
00000011 11100011 00110001 01010100 01010110 00100010 11110010 01101101
11001011 10010011 11111110 00000011 00001001 00010111 00011101 01110011
01001111 01110011 11000011 00111001 11011101 00011010 10110101 11100011
01001100 01111100 00101111 01111100 00101000 00011100 00010101 10001000
11001000 11110011 01001110 00111000 01111000 11011101 01011111 10011100
11101100 10010010 10001101 01101111 01010111 11101001 11011110 10110011
00110000 01000110 01111110 01011111 01001010 10000000 10010110 11001011
01111011 10000011 10100011 10010001 00101010 11001010 00001110 10110001
11011101 11100101 01010101 11001010 11000010 00101101 10011111 11010010
01011010 01001111 00000100 00111001 11000011 11011000 11011011 01101001

11100100 11010010 00001010 11101000 10010100 11111011 10111110 10001010
10100111 00100100 11011110 11110001 11000101 10001100 11000110 00110001
11100011 11001111 00010110 00010010 10101001 11000001 01111011 01010111
00010100 00111110 10110011 11111010 10001001 10100010 10110111 00111100
11101110 10000110 10011110 10110111 00110111 10010001 00011101 10100110
11011011 00101100 00111111 01001101 01010000 10101010 11011111 10000101
11001110 01101001 00010010 00101000 11111000 10100011 11000101 10010000
00101101 11100011 11000011 11111101 01000101 11110011 00101100 10110010
10010001 11011001 00001000 10000101 10010001 11101101 10000010 11001001
00110100 11010110 01011000 01101000 10110001 01101100 01111001 10100010
01110111 00100110 11100001 10010001 00001101 11000111 00001111 00000111
10100000 00011001 00010111 10100000 00001000 01111010 10111101 11110111
11010101 00101111 01000011 01100101 11000000 10000010 11100011 10110110
10001001 10111010 10001101 01110001 01011100 10011100 11001100 11100111
11011010 11010010 10011011 11010001 10000110 00100011 11001101 00001101
10111100 01101000 10111010 11011011 01000011 00100101 11101000 11100100
10010010 10100001 00100000 11111000 10000000 11010110 10100100 11110111
11101010 00001010 01011001 10110101 01011100 00110001 11100011 01011001

REFERANSLAR

Ahmadi H., Eghlidos T., Khazaei S., "Improved Guess and Determine Attack on SOSEMANUK", eStream, ECRYPT Stream Cipher Project, Report , 2005

Akış Şifreler (Stream Cipher), http://en.wikipedia.org/wiki/Stream_cipher, 2011.

Aoki K., Ichikawa T., Kanda M., Matsui M., Moriai S., Nakajima J., Tokita T., Camellia: A 128-bit block cipher suitable for multiple platforms-design and analysis, In Proceedings of Selected Areas in Cryptography (SAC 2000), Lecture Notes in Computer Science, Vol. 2012, pp. 39-56, Springer-Verlag, 2001.

Aslan B., Boole Fonksiyonları ve S-kutularının Kriptografik Özelliklerinin İncelenmesi ve Ters Haritalama Tabanlı Cebirsel Açıdan Güçlendirilmiş Bir S-kutusu Önerisi, Yüksek Lisans Tezi, Trakya Üniversitesi, Türkiye, 2008.

Aslan B., Blok Şifreler için 16×16 İkili Doğrusal Dönüşümlerin Cebirsel Tasarımı, Doktora Semineri, Trakya Üniversitesi Fen Bilimleri Enstitüsü, 2011.

Aslan B., Sakallı M. T., Buluş E., Üs Haritalama Tabanlı Cebirsel 8-bit giriş 8-bit çıkışlı S-kutularının Sınıflandırılması, Ağ ve Bilgi Ulusal Sempozyumu 2, Girne-Kıbrıs, 2008.

Aslan B., Sakallı M. T., Buluş E., Classifying 8-bit to 8-bit S-boxes based on Power Mappings from the point of DDT and LAT Distributions, Proceedings of International Workshop on the Arithmetic of Finite Fields, WAIFI 2008, Lecture Notes in Computer Science, Springer-Verlag, Vol. 5130, pp. 123-133, 2008.

Aslan F. Y., Sakallı M. T., Aslan B., AES Şifresinde Kullanılan MixColumns Dönüşümü Üzerine, Elektrik-Elektronik ve Bilgisayar Mühendisliği Sempozyumu, ELECO 2010, 2010.

Babbage S., Dodd M., The Stream Cipher MICKEY (version 1), eStream, ECRYPT Stream Cipher Project, Report 2005/015, 2005, http://www.ecrypt.eu.org/stream.

Babbage S., Dodd M., "The stream cipher MICKEY-128", (ECRYPT) Stream Cipher Project Report 2005/016.

Barreto P. S. L. M., RijmeN V., The Khazad Legacy-Level Block Cipher, Proceedings First open NESSIE Workshop, Leuven, 2000.

Berbain C., Billet O., Canteaut A., Courtois N., Gilbert H., Goubin L., Gouget A., Granboulan L., Lauradoux C., Minier M., Pornin T., Sibert H., Sosemanuk – a Fast

Software-oriented Stream Cipher, eStream, ECRYPT Stream Cipher Project, Report 2005/027, 2005, http://www.ecrypt.eu.org/stream.

Bernstein D. J., Salsa20 Design, eStream, ECRYPT Stream Cipher Project, Report 2005/025, 2005, http://www.ecrypt.eu.org/stream.

Biham E., Anderson R., Knudsen L. R., Serpent: A New Block Cipher Proposal, In Proceedings of 5th International Workshop of Fast Software Encryption-FSE'98, Lecture Notes in Computer Science, Vol. 1372, pp. 222-238, Springer-Verlag, 1998.

Biham E, Shamir A. Differential Cryptanalysis of DES-like Cryptosystems. In Proceedings of CRYPTO'90, Lecture Notes in Computer Science, Springer-Verlag, 1990; 537: 2-21.

Biham E., New types of cryptanalytic attacks using related keys, Advances in Cryptology-EUROCRYPT'93, Lecture Notes in Computer Science, Vol. 765, pp. 398-409, Springer-Verlag, 1994.

Biryukov A., *Methods of Cryptanalysis*, PHd Thesis, 1999.

Biryukov A., A New 128-bit Key Stream Cipher LEX, eStream, ECRYPT Stream Cipher Project, Report 2005/012, 2005, http://www.ecrypt.eu.org/stream.

Biryukov A., Dunkelman O., Keller N., Khovratovich D., Shamir A., Key Recovery Attacks of Practical Complexity on AES Variants With Up To 10 Rounds, Cryptology ePrint Archive, Report 2009/374, 2009. Available at http://eprint.iacr.org/2009/374/.

Biryukov A., Khovratovich D., Related-key Cryptanalysis of the Full AES-192 and AES-256, Cryptology ePrint Archive, Report 2009/317, 2009. Available at http://eprint.iacr.org/2009/317/.

Biryukov A., Khovratovich D., Related-key Cryptanalysis of the Full AES-192 and AES-256, Advances in Cryptology, ASIACRYPT'09, 15th International Conference on the Theory and Application of Cryptology and Information Security, Lecture Notes in Computer Science, Vol. 5912, pp. 1-18, Springer-Verlag, 2009.

Biryukov A., Khovratovich D., Nikolic I., Distinguisher and Related-Key Attack on the Full AES-256, Advances in Cryptology-CRYPTO'09, Lecture Notes in Computer Science, Vol. 5677, pp. 231-249, Springer-Verlag, 2009.

Boesgaard M., Vesterager M., Christensen T., Zenner E., The Stream Cipher Rabbit, eStream, ECRYPT Stream Cipher Project, Report 2005/024, 2005, http://www.ecrypt.eu.org/stream.

Büyüksaraçoğlu F., Buluş E., Sözde Rastsal Sayı Üretiminin Kriptografik Açıdan İncelenmesi, İletişim Teknolojileri Ulusal Sempozyumu (İTUSEM), Türkiye, 2009.

Canteaut A., Stream Ciphers, Encyclopedia of Cryptography and Security, 2005, available at: http://www-rocq.inria.fr/codes/Anne.Canteaut/encyclopedia.pdf.

Daemen J., Rijmen V., The Design of Rijndael, AES-The Advanced Encryption Standard, Springer-Verlag, 2002.

De Canniere C., Preneel B., Trivium – a stream cipher construction inspired by block cipher design principles, eStream, ECRYPT Stream Cipher Project, Report 2005/030, 2005, http://www.ecrypt.eu.org/stream.

Demirkol A. Ş., Kaotik Osilatör Girişli Adc Tabanlı Rastgele Sayı Üreteci, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, Türkiye, Haziran 2007.

Diffie W., Hellman M.E., New Directions in Crytography, IEEE Transactions on Information Theory, Vol. 22, pp. 644-654, 1976.

Ekdahl P., On LFSR Based Stream Ciphers: Analysis and Design, Ph.D. Thesis, Lund University, Sweden, 2003.

Ekdahl P., Johansson T., A new version of the stream cipher SNOW, In SAC'02: Revised Papers from the 9th Annual International Workshop on Selected Areas in Cryptography, pp. 47-61, London, UK, 2003, Springer-Verlag.

Feistel H., Cryptography and Computer privacy, Scientific American, Vol. 228, no. 5, pp. 15-23, 1973

Ferguson N., Kelsey J., Lucks S., Schneier B., Stay M., Wagner D. and Whiting D., Improved Cryptanalysis of Rijndael, Proceedings of the 7th Fast Software Encryption, Lecture Notes in Computer Science, Springer-Verlag, Vol. 1978, pp. 213-230, 2000.

Forouzan Behrouz A., Cryptography and Network Security, McGraw-Hill International Edition, 2008.

Fuller J., Millan W, Linear redundancy in S-boxes, Proceedings of the Fast Software Encryption (FSE 2003), Lecture Notes in Computer Science, Vol. 2887, pp. 74–86 Springer, Berlin, 2003.

Grah J. S., Hash Functions in Cryptography, MSc. Thesis, University of Bergen, Norway, 2008.

Günther C. G., *Alternating Step Generators Controlled by the Brujin Sequences*, EUROCRYPT'87, Lecturer Notes in Computer Science, pp. 91-103, 1988.

Hell M., Johansson T., Meier W. Maximov A., Meier W, A Stream Cipher Proposal: Grain-128, eStream, ECRYPT Stream Cipher Project, Report 2006, http://www.ecrypt.eu.org/stream.

Hell M., Johansson T., Meier W., Grain - A Stream Cipher for Constrained Environments, eStream, ECRYPT Stream Cipher Project, Report 2005/010, 2005, http://www.ecrypt.eu.org/stream.

Kam J. B., Davida G. I., Structured Design of Substitution-Permutation Encryption Networks. IEEE Transactions on Computers, Vol. 28, no. 10, pp. 747-753, 1979.

Kang J-S, Hong S., Lee S., Yi O., Park C., Lim J., Practical and Provable Security against Differential and Linear Cryptanalysis for Substitution-Permutation Networks, ETRI journal, Vol. 23, No. 4, pp. 158-167, 2001.

Keliher L., Linear Cryptanalysis of Substitution Permutation Networks, Ph.D. Thesis, Queen's University, Kingston, Ontario, Canada, 2003.

Key E.L., An analysis of the structure and complexity of nonlinear binary sequence generators, IEEE Transactions on Information Theory, vol. 22, pp. 732–736, 1976.

Kitsos P., On the Hardware Implementation of the MICKEY-128 Stream Cipher, eStream, ECRYPT Stream Cipher Project, Report 2006/059.

Knudsen L., Practically Secure Feistel Ciphers, Fast Software Encryption, First International Workshop Proceedings, Lecture Notes in Computer Science, Springer-Verlag, Vol. 809, pp 211-221, 1993.

Kriptolojiye Giriş Ders Notları, Uygulamalı Matematik Enstitüsü, Kriptografi Bölümü, ODTÜ,Türkiye,Şubat 2004.

Kwon D., Kim J., Park S., Sung S.H., Sohn Y., Song J. H., Yeom Y., Yoon E-J, Lee S., Lee J., Chee S., Han D., Hong J., New Block Cipher: ARIA, In Proceedings of International Conference on Information Security and Cryptology, Lecture Notes in Computer Science, Vol. 2971, pp. 432-445, Springer-Verlag, 2004.

Kwon D., Sung S. H., Song J. H., Park S., Design of Block Ciphers and Coding Theory, Trends in Mathematics, Vol. 3, No. 1, pp. 13-20, 2005.

Lidl R., Niederreiter H., Finite Fields, Encyclopedia of Mathematics and its Applications, Addison-Wesley, Reading, MA, 1983.

Massey J.L., *The ubiquity of Reed-Muller Codes*, In Applied Algebra, Algebraic Algorithms and Error-Correcting Codes - *AAECC-14*, Lecture Notes in Computer Science number 2227, pages 1–12. Springer-Verlag, 2001.

Matsui M. Linear Cryptanalysis Method for DES Cipher. In Proceedings of EUROCRYPT 93, Lecture Notes in Computer Science, Springer-Verlag, 1994;765: 386-397.

May L., Henricksen M., Millan W., Carter G., and Dawson E., Strengthening the Key Schedule of the AES, Proceedings of the 7th Australasian Conference on Information Seucurity and Privacy (ACISP 2002), Lecture Notes in Computer Science, Springer-Verlag, Vol. 2384, pp. 226–240, 2002.

McEliece R. J., Finite fields for Computer Scientists and Engineers, Kluwer Academic Publishers, Dordrecht, 1987.

Menezes A., P. v. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.

Nakahara Jr. J., Abrahao E., A New Involutory MDS Matrix for the AES, International Journal of Network Security, Vol. 9, No. 2, pp. 109-116, 2009.

National Institute of Standard and Technology, A Statistical Test Suite for Random and Pseudo Random Number Generators for Cryptographic Applications, NIST800-22, 2001. Available at: *http://csrc.nist.gov/rng/SP800-22b.pdf*

Nyberg K., Differentially uniform mappings for cryptography, Proceedings of Eurocrypt'93, Lecture Notes in Computer Science, Vol. 765, Springer, Berlin, pp. 55-64, 1994.

Phan R. C.-W., Impossible differential cryptanalysis of 7-round Advanced Encryption Standard (AES), Information Processing Letters Vol. 91, pp. 33-38, 2004.

Rimoldi Anna, On algebraic and statistical properties of AES-like ciphers, Ph.D. Thesis, University of Trento, 2009.

Rivest R. L., Shamir A., Adleman L., A method for obtaining digital signatures and public-key cryptosystems, Communications of ACM, Vol. 21, pp. 120-126, 1978.

Rueppel R.A., *Analysis and design of stream ciphers*, Springer-Verlag, 1986.

Sakallı M. T., Modern Şifreleme Yöntemlerinin Gücünün İncelenmesi, Doktora Tezi, Trakya Üniversitesi, Türkiye, 2006.

Shannon C. E., Communication theory of secrecy systems, Bell System Technical Journal, Vol. 28, No. 4, pp. 656-715, 1949.

SIG Bluetooth, Bluetooth specification, Available at http://www.bluetooth.com, 2003.

Stinson D. R., Cryptography: Theory and Practice, Second Edition, CRC Press, 2002.

Turan M. S., On Statistical Analysis of Synchronous Stream Ciphers, Ph.D. Thesis, Middle East Technical University, Turkey, 2008.

US National Institute of Standards and Technology, Data Encryption Standard, Federal Information Processing Standards Publications, No. 46-3, 1999.

US National Institute of Standards and Technology Advanced Encryption Standard, Federal Information Processing Standards Publications, No. 197, 2001.

Webster A. F., Tavares S. E., On the Design of S-boxes, Proceedings of CRYPTO'85, Lecture Notes in Computer Science, Springer-Verlag, Vol. 218, pp. 523-534, 1986.

Wu H., Cryptanalysis and Design of Stream Ciphers, Ph.D. Thesis, K.U. Leuven, Belgium, 2008.

Wu H., The Stream Cipher HC-128, http://www.ecrypt.eu.org/stream/hcp3.html.

Yalçın M., Suykens J. and Vandewalle J., 2004. True Random Bit Generation from a Double Scroll Attractor, *IEEE Trans. Circuits Syst. I*, 51, 1395-1404.

Youssef A. M., Mister S., Tavares S. E., On the Design of Linear Transformation for Substitution Permutation Encryption Networks, Proceedings of Selected Areas in Cryptography (SAC'97), pp. 40-48, 1997.

Youssef A. M., Tavares S.E., Affine equivalence in the AES round function, Discrete Applied Mathematics, Elsevier, 2005.

Youssef A. M., Tavares S.E., Gong G., On Some probabilistic approximations for AES-like s-boxes, Discrete Mathematics, Elsevier, 2006.

Z'aba M. R., Analysis of Linear Relationships in Block Ciphers, Ph.D. Thesis, Queensland University, Brisbane, Australia, 2010.

ÖZGEÇMİŞ

Fatma BÜYÜKSARAÇOĞLU SAKALLI 1977 yılında Sakarya'da doğdu. İlk ve orta öğrenimini Sakarya'da tamamladıktan sonra 1996 yılında girdiği Trakya Üniversitesi Mühendislik-Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü'nden 2000 yılında mezun oldu. 2000 yılında Trakya Üniversitesi Mühendislik Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü'nde Araştırma Görevlisi kadrosuna atandı ve aynı yıl Trakya Üniversitesi Fen Bilimleri Enstitüsü'ne bağlı olarak Yüksek Lisans eğitimine başladı. Yüksek lisansını 2003 yılında başarıyla bitirdi. 2005 yılında Trakya Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Ana Bilim Dalı'nda doktora çalışmalarına başladı. Fatma BÜYÜKSARAÇOĞLU SAKALLI halen Trakya Üniversitesi Mühendislik Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü'nde Araştırma Görevlisi olarak çalışmaktadır.