İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

KRİPTO ANALİZDE MELEZ BİR YÖNTEM: ÇAKIŞMA SALDIRISI

YÜKSEK LİSANS TEZİ

Müh. Ferhat KARAKOÇ

Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Programı : BİLGİSAYAR MÜHENDİSLİĞİ

OCAK 2008

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

KRİPTO ANALİZDE MELEZ BİR YÖNTEM: ÇAKIŞMA SALDIRISI

YÜKSEK LİSANS TEZİ

Müh. Ferhat KARAKOÇ
(504051509)

Tezin Enstitüye Verildiği Tarih : 24 Aralık 2007
Tezin Savunulduğu Tarih : 29 Ocak 2008

Tez Danışmanı :	**Prof. Dr. A. Emre HARMANCI**
Tez Eş Danışmanı :	**Y.Doç.Dr. S. Berna Örs YALÇIN (İ.T.Ü.)**
Diğer Jüri Üyeleri	**Prof.Dr. Bülent ÖRENCİK (İ.T.Ü.)**
	Y.Doç.Dr. D. Turgay ALTILAR (İ.T.Ü.)
	Prof.Dr. Ece Olcay GÜNEŞ (İ.T.Ü.)

OCAK 2008

ÖNSÖZ

Tezimde yapmış oldukları yardımlardan ve desteklerinden dolayı değerleri hocalarım Prof. Dr. A. Emre Harmancı ve Yrd. Doç. Dr. Berna Örs Yalçın'a teşekkürü bir borç bilirim.

Ocak 2008

Ferhat Karakoç

Bilgisayar Mühendisi

İÇİNDEKİLER

KISALTMALAR

3DES : Üçlü DES (Triple DES)
EKM : Elektronik Kod Modu
DES : Veri Şifreleme Standardı (Data Encryption Standard)
SPN : Yer Alma- Yer Değiştirme Ağı (Substution-Permutation Network)

TABLO LİSTESİ

ŞEKİL LİSTESİ

SEMBOL LİSTESİ

C	**:** Kapalı Metin
D	**:** Şifre Çözme Fonksiyonu
E	**:** Şifreleme Fonksiyonu
K	**:** Anahtar
P	**:** Açık Metin

KRİPTO ANALİZDE MELEZ BİR YÖNTEM: ÇAKIŞMA SALDIRISI

ÖZET

Şifre bilimi olan kriptoloji kendi içerisinde iki dala ayrılır. Bu dallardan biri olan kriptografi, şifreleme sistemlerinin oluşturulmasını içerirken kripto analiz, şifreleme sistemlerinin analizi ile uğraşır. Kripto analizin zamanla gelişmesi ile şifreleme sistemleri kırılabildiği için bu gelişme kriptografinin de gelişmesine neden olur. Sistemin kırılması, sistemde kullanılan gizli anahtarın ele geçirilmesi ve/veya şifrelenmiş (kapalı) metinden şifrelenmemiş (açık) metnin bulunabilmesi demektir. Kriptografinin gelişmesi ile de kripto analiz, gelişmesini devam ettirmek zorunda kalmaktadır. Dolayısıyla bu iki dal birbirini tetikleyerek ilerlemektedir.

Sistemi analiz ederek sistemin kırılması için gerekli süre, kaynak ve veri ihtiyacını ortaya koyan kripto analiz içinde temelde iki farklı yaklaşım bulunmaktadır. Klasik kripto analiz bu yaklaşımlardan bir tanesi iken yan kanal saldırıları da diğeridir. Klasik kripto analize matematiksel veya teorik kripto analiz de denmektedir.

Klasik kripto analizde, saldırı direk olarak şifreleme algoritmasına yapılmaktadır. Bu analizde, şifreleme algoritmasında bulunan zayıflıklar ortaya çıkarılarak bu zayıflıklar ile sistemde kullanılan anahtar elde edilmeye çalışılır. Bu zayıflıklara, sistemin bir olasılık ile doğrusal modelinin çıkarılması, açık metin faklarının algoritma çıkışında bir olasılık ile başka bir farka yaklaşması, açık metin toplamlarının algoritma içerisinde bazı özellikleri sağlayarak ilerlemesi örnek olarak verilebilir. Klasik kripto analizin tipik özelliği, elde sadece açık ve/veya kapalı metinlerin olmasıdır.

Yan kanal saldırılarında ise saldırı, şifreleme sisteminin uygulanışına yapılmaktadır. Yan kanal saldırılarında açık ve/veya kapalı metin çiftlerinin yanı sıra fiziksel bilgiler de kullanılır. Bu fiziksel bilgiler çekilen akım, ortama yayılan manyetik alan, oluşan ısı gibi büyüklükler olabilir. Bu değerleri elde edebilmek için de şifreleme yapan sisteme sahip olmak gerekir. Mesela elektronik bir kart üzerindeki anahtarı bulabilmek için şifreleme yapan elektronik karta sahip olmak gerekir.

Son zamanlarda, klasik ve yan kanal saldırılarının bir arada kullanılması üzerine çalışmalar yapılmaktadır. Bu yöntemlerin herhangi birinden elde edilen bilgi diğer yöntemde kullanılabilir. Böylece, ortaya daha güçlü bir saldırı çıkmaktadır. İki yöntemin beraber kullanıldığı saldırılardan bir tanesi çakışma saldırısıdır. Bu saldırıda, farksal kripto analiz ile yan kanal saldırısı beraber kullanılmaktadır. Şifreleme algoritmasına giren iki farklı girişin aynı çıkışa neden olması sağlanarak saldırı uygulanmaktadır. Bu saldırı, algoritma içerisinde olduğundan ve algoritma çıkışına yansımadığından yan kanal bilgileri kullanılır. Yan kanal bilgileri ile çakışma olduğu anlaşılır ve bu çakışmaya neden olan girişler kullanılarak anahtar hesaplanır.

Yapılan çalışmada, DES algoritmasına uygulanan çakışma saldırısının gerçeklenmesi yapılıp saldırının genel bir tanımı oluşturuldu ve saldırı, 3DES algoritmasına uygulandı. 3DES'e yapılan diğer saldırılarla çakışma saldırısı karşılaştırıldığında çakışma saldırısının daha etkili olduğu görüldü.

A NEW COMBINED CRYPTANALYSIS METHOD: COLLISION ATTACK

SUMMARY

Cryptology is the practice and study of hiding information. It has two main areas. One of them is cryptography that is the use and practice of cryptographic techniques. The other one is cryptanalysis. The goal of cryptanalysis is to find some weakness or insecurity in a cryptographic scheme to get the secret. Cryptography is getting stronger, because cryptanalysis is being more powerful. They challenge each other.

There are two main approaches in cryptanalysis. Classical cryptanalysis is one of them. The other one is side channel attacks. Classical cryptanalysis is called as mathematical or theoretical cryptanalysis also.

In classical cryptanalysis, the weakness is searched in the algorithm itself while in side channel attack the weakness is searched in the usage of the algorithm.

In recent years, there are some studies on combined cryptanalysis. Combined cryptanalysis is the cryptanalysis in which classical cryptanalysis and side channel attacks are used.

In this thesis, we searched a new method applied on DES algorithm. We give a general definition for the method whose name is collision attack and apply the method to the 3DES algorithm. We show that, collision attack on 3DES gives better results than other attacks on 3DES.

1. GİRİŞ

Bu bölümde, tezin amacı ve tezde yapılan çalışmalar belirtilmektedir.

1.1 Tezin Amacı

Bu çalışmada, farksal kripto analiz [1] ile güç tüketimi analizinin [2] kullanıldığı çakışma saldırı [3] yönteminin DES algoritması [1] üzerindeki uygulaması incelendi ve saldırı simülasyon kullanılarak gerçeklendi. Saldırının genel bir tanımı yapıldı ve saldırı 3DES [4] algoritmasına uygulandı.

1.2 Çakışma Saldırısının DES Algoritması Üzerinde Uygulanması

[3]'te anlatılan çakışma saldırısı incelendi ve saldırının uygulaması gerçeklendi. Güç tüketimi analizi için akım ölçme fiziksel olarak yapılmadı. Bunun yerine simülasyon kullanıldı.

1.3 Saldırının Genel Tanımı

Çakışma saldırısı ile ilgili yapılan çalışmalar genelde algoritmaya özel anlatılmaktadır [3,5]. Çakışma saldırısının genel tanımı bu tez kapsamında yapıldı. Saldırı genel bir ifade ile Bölüm 5'de anlatıldı.

1.4 Saldırının 3DES Algoritması Üzerinde Uygulaması

3 adet DES algoritmasının art arda kullanıldığı 3DES algoritmasına çakışma saldırısının nasıl uygulanabileceği gösterildi ve uygulaması yapıldı. Saldırının 3DES'e uygulanması Bölüm 6'da anlatıldı.

2. BLOK ŞİFRELEME ALGORİTMALARI

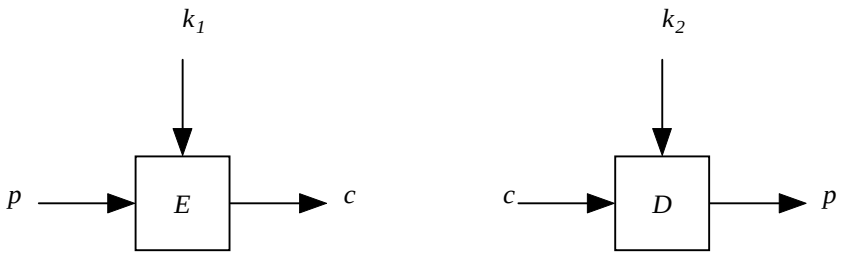
Bu bölümde, şifreleme algoritma tiplerinden bir tanesi olan blok şifreleme algoritmaları hakkında bilgi verilmektedir. Ayrıca, blok şifreleme algoritmalarından olan DES ve 3DES algoritmaları da anlatılmaktadır.

2.1 Giriş

Grafiksel olarak Şekil 2.1'de görülen bir şifreleme sistemi (*P,C,K,E,D*) beşlisinden oluşur [6]. *P* açık metin (şifrelenecek metin) kümesi, *C* kapalı metin (şifreli metin) kümesi, *K* anahtar kümesi, *E* şifreleme kuralı (algoritması) ve *D* şifre çözme kuralıdır. Açık metinden kapalı metin elde etme işlemine şifreleme, kapalı metinden açık metin elde etme işlemine de şifre çözme işlemi denir. $p \in P, c \in C, k_1 \in K, k_2 \in K$ olmak üzere şifreleme ve şifre çözme işlemleri sırayla Eşitlik (2.1) ve (2.2)'deki gibidir.

$$c = E(k_1, p) \tag{2.1}$$

$$p = D(k_2, c) \tag{2.2}$$



Şekil 2.1 : Şifreleme sistemi [7]

$k_1 = k_2$ ise veya birinden diğeri kolayca elde edilebiliyor ise şifreleme sistemine simetrik anahtarlı şifreleme sistemi denir [7]. k_1 anahtarı açık (herkes tarafından

bilinir) ve k_2 anahtarı gizli ise ve k_1'den k_2'yi bulmak pratik olarak imkansız ise bu tip şifreleme sistemlerine de açık anahtarlı şifreleme sistemleri denir.

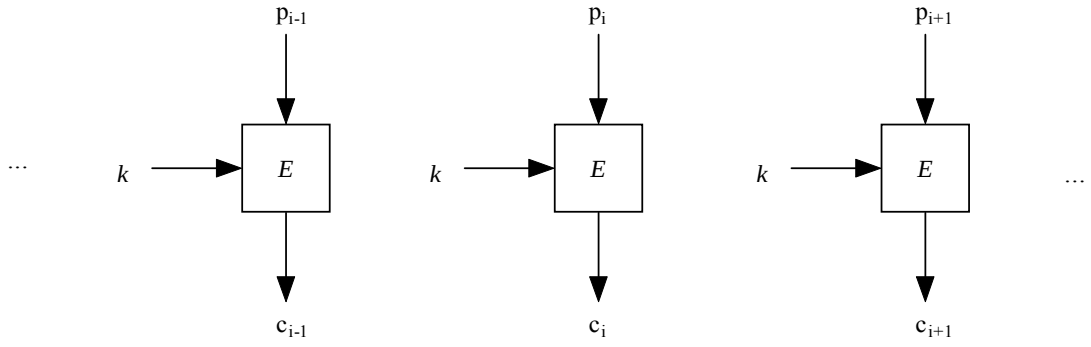
Simetrik anahtarlı şifreleme olan blok şifreleme sistemlerinde, açık metin bloklara ayrılır ve her blok bir bütün olarak şifrelenir [7]. EKM ile şifrelemede bloklar birbirlerinden bağımsız şifrelendiği için şifrelenmiş verinin iletimi sırasında meydana gelen hata, sadece hatanın olduğu bloğun bozulmasına neden olur, diğer bloklar bundan etkilenmez. Metin içinde aynı bloklara karşılık gelen şifrelenmiş bloklar aynı olur. Bu durum sistemin güvenilirliği için dezavantaj oluşturur. Blok şifrelemedeki bu dezavantajı ortadan kaldırmak için bazı farklı modlar geliştirilmiştir. Bu modlar Bölüm 2.2'de yer almaktadır.

Blok şifrelemede kullanılan algoritmalara blok şifreleyiciler denir [7]. Modern blok şifreleyiciler, birim şifreleyicilerin çarpımından oluşur [6]. Birim şifreleyiciler, yerini alma ve yer değiştirme işlemlerine dayanan şifreleyicilerdir [8]. Çarpım şifreleyiciler birim şifreleme algoritmalarının bir arada kullanılması ile oluşmuş şifreleme yapılarıdır. Bu şifreleyiciler [9]'da Shannon tarafından yayımlanan makaleye dayanmaktadır.

2.2 Şifreleme Modları

2.2.1 Elektronik Kod Modu

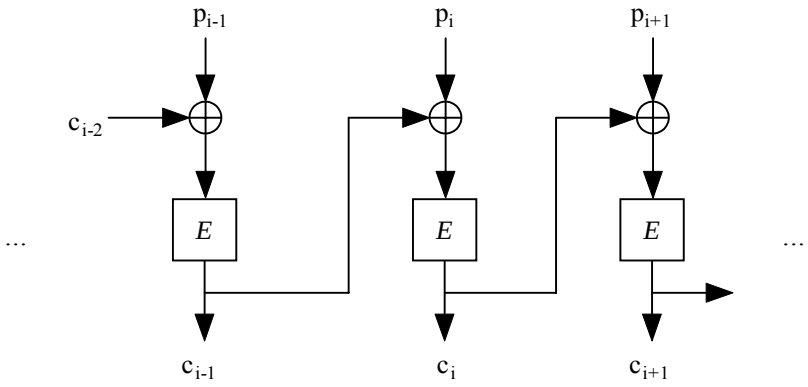
Şekil 2.2'de gösterilen Elektronik Kod Modunda şifrelenecek her bir blok birbirinden bağımsız olarak şifrelenir [7].



Şekil 2.2 : Elektronik Kod Modu [7]

2.2.2 Kapalı Metin Zincirleme Modu

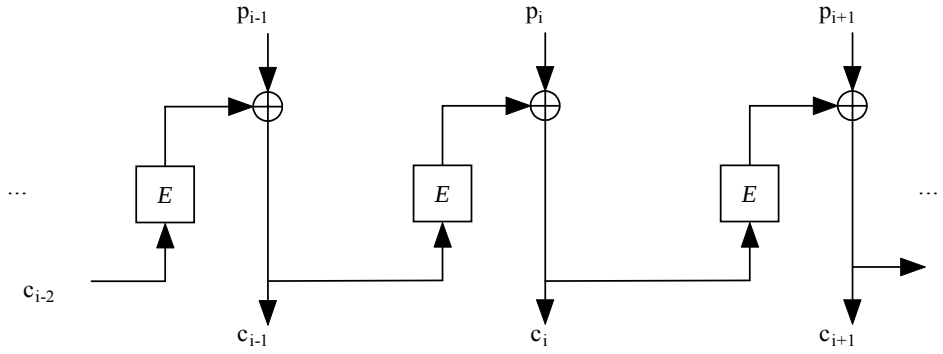
Grafiksel olarak Şekil 2.3'te görülen Kapalı Metin Zincirleme modunda, bir bloğun şifrelenmiş hali bir sonraki bloğun şifrelenmesi sırasında kullanılır [7].



Şekil 2.3 : Kapalı Metin Zincirleme Modu [7]

2.2.3 Çıktıyı Geri Besleme Modu

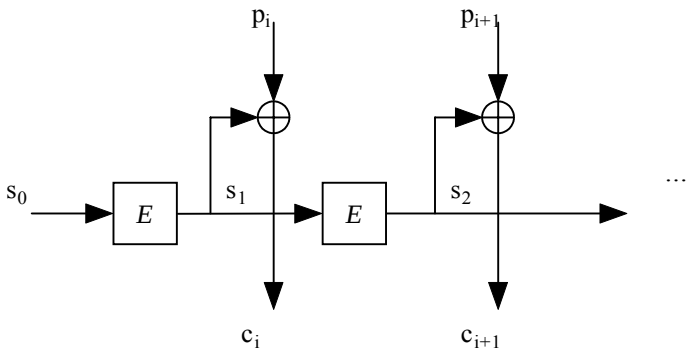
Çıktıyı Geri Besleme Modunda, şifrelenmiş veri tekrar şifrelenmektedir [7]. Yapısı Şekil 2.4'te bulunmaktadır.



Şekil 2.4 : Çıktıyı Geri Besleme Modu [7]

2.2.4 Girdiyi Geri Besleme Modeli

Yapısı Şekil 2.5'te görülen Girdiyi Geri Besleme Modunda, başlangıçta seçilen bir değer sürekli şifrelenir ve açık metin ile şifreleme sonucu elde edilen değer *dar veya* (*xor*) işlemine sokularak kapalı metin elde edilir [7].



Şekil 2.5 : Girdiyi Geri Besleme Modu [7]

2.3 Blok Şifreleyici Tasarım Kriterleri

Şifreleme ve şifre çözme işlemlerinin kolay yapılabilmesinin yanında $c = E(k, p)$ ve $p = D(k, c)$ eşitliklerinden k anahtarı kolaylıkla bulunamamalıdır. İlk defa Shannon tarafından önerilen tasarım ölçütleri yayılma ve nüfuz etmedir [9].

2.3.1 Yayılma

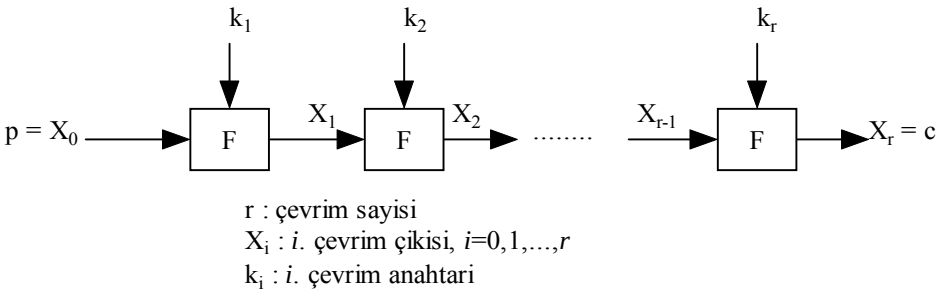
Yayılma, şifreli metin ile anahtar arasındaki ilişkinin olabildiğince karışık olması demektir [9]. Bu ilişki doğrusal olmamakla beraber anahtarın tüm bitlerinin çıkışta eşit şekilde etkili olması sağlanır.

2.3.2 Nüfuz Etme

Nüfuz etme ile anahtarın ve açık metnin her bitinin kapalı metni etkilemesi sağlanır [9]. Böylece şifreli metin ile açık metin arasında istatistiksel bağlılık kurulamaz hale getirilir.

2.4 İteratif Blok Şifreleyiciler

Blok şifrelerin büyük bir kısmı iteratif yapıdadır [7]. Çevrim adı verilen fonksiyon çevrim sayısı kadar çalıştırılır. Çevrim sayısı arttıkça yayılma ve nüfuz etme artmaktadır. Bir çevrimin çıkışı bir sonraki çevrimin girişi olur. İlk çevrim girişi açık metin, son çevrim çıkışı da kapalı metindir. Çevrim fonksiyonu, çevrim girişini ve çevrim anahtarını alır ve bir çevrim çıkışı oluşturur. Çevrimlerde, genelde farklı çevrim anahtarı kullanılır. Çevrim anahtarı bir anahtar üretme algoritması kullanılarak ana bir anahtardan üretilir.



Şekil 2.6 : İteratif blok şifreleme

2.4.1 Yerine Koyma- Yer Değiştirme Ağı

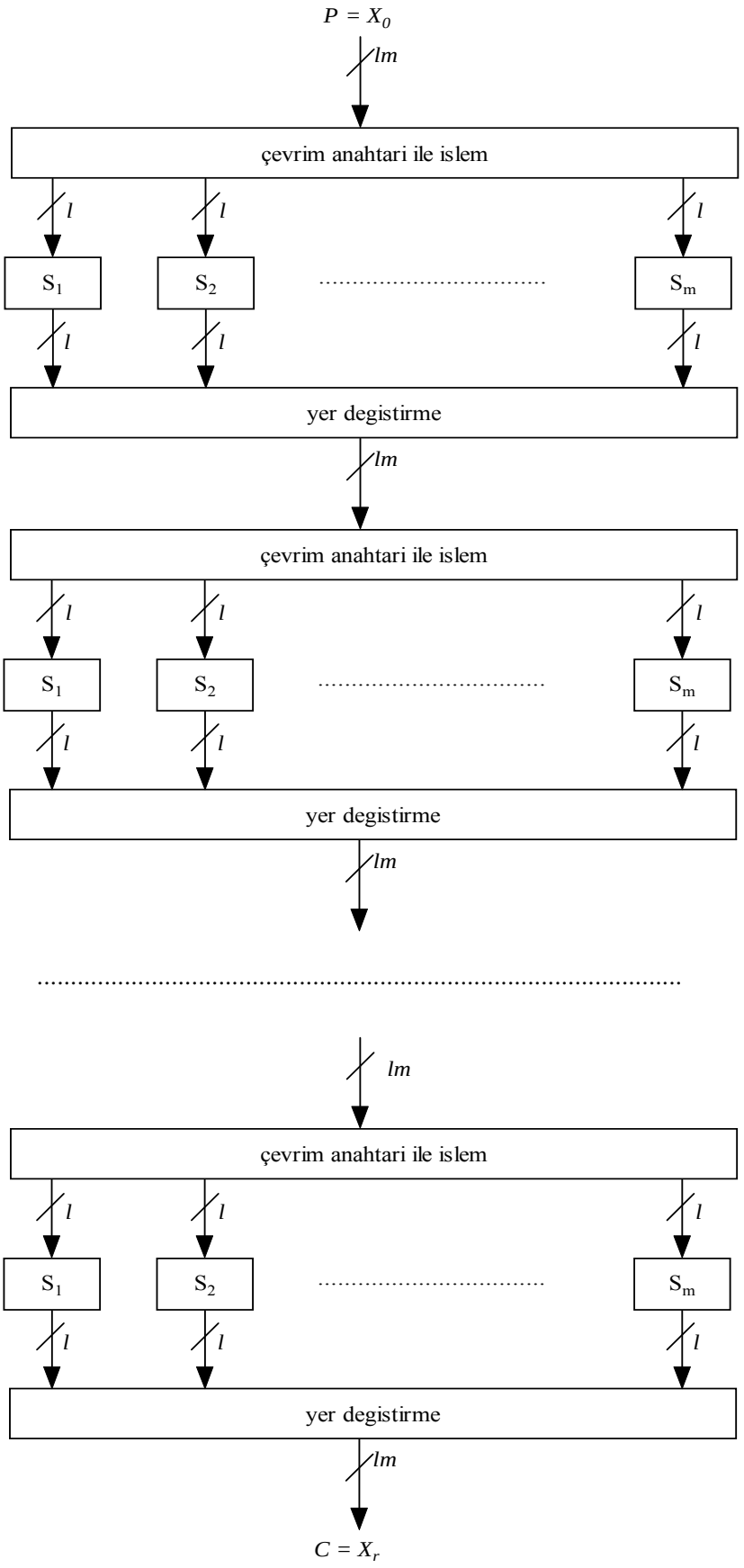
Yapısı Şekil 2.8'de bulunan yerine koyma- yer değiştirme ağı döngülü blok şifreleme yapısındadır [6]. İçerisinde temel iki işlem vardır. Blok uzunluğu $n = l.m$ olmak üzere işlemler Eşitlik (2.3) ve Eşitlik (2.4)'te bulunmaktadır.

$$\pi_s : \{0,1\}^l \rightarrow \{0,1\}^l \tag{2.3}$$

$$\pi_p = \{1,...,lm\} \rightarrow \{1,...,lm\} \tag{2.4}$$

Eşitlik (2.3) yerine koyma işlemi olup buradaki fonksiyon, doğrusal olmayan bir fonksiyondur ve s-kutusu olarak adlandırılır. Bu fonksiyon ile yayılma yani anahtar bitleri ile açık metnin bitlerinin birbirine karıştırılması sağlanır. Ağ içindeki her bir çevrimde m adet s-kutusu bulunur.

Eşitlik (2.4)'te bulunan fonksiyon ise yer değiştirme fonksiyonudur. Bu işlem de anahtar bitlerinin kapalı metin içerisinde dağılmasını sağlar. Her çevrimde bir adet yer değiştirme işlemi bulunur.



Şekil 2.7 : Yerine koyma – yer değiştirme ağı [6]

2.4.2 Feistel Yapıları

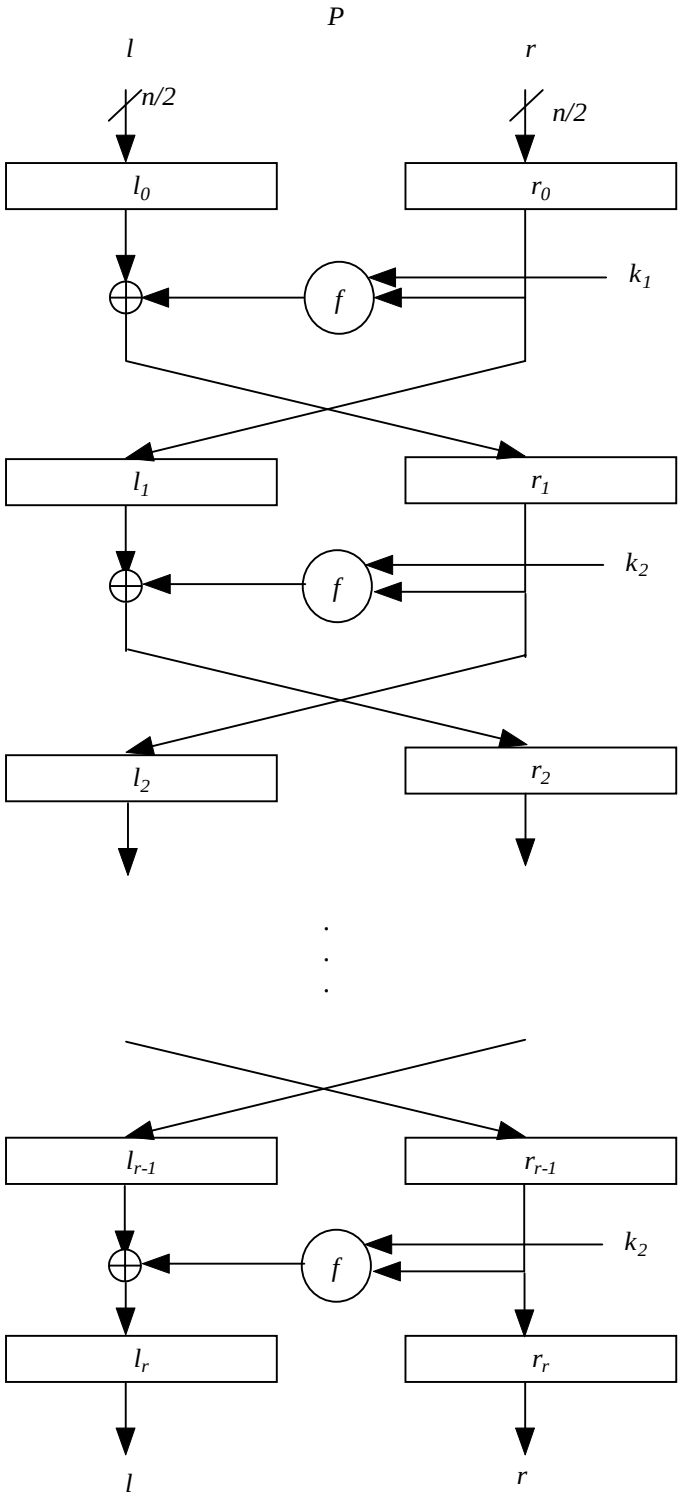
Yapısı Şekil 2.8'de görülen Feistel yapılarında, çevrim bloğu iki eşit parçaya ayrılır. Her çevrimde, bu parçalardan sadece biri değiştirilir [7]. Değiştirilmeyen parça çevrim çıkışına bir değişikliğe uğramadan yansır. Feistel yapılarının temel özelliği şifreleme işlemi ile şifre çözme işleminin aynı olmasıdır. Şifreleme ile şifre çözme arasındaki tek fark, çevrim anahtarlarının ters sırada olmasıdır. Şifreleme yapılırken çevrim anahtarları sırasıyla $\{k_0, k_1, \ldots, k_r\}$ iken şifre çözme işleminde çevrim anahtarları sırası ile $\{k_r, k_{r-1}, \ldots, k_0\}$'dır.

F çevrim fonksiyonu olmak üzere bir çevrim fonksiyonu Eşitlik (2.5) ile ifade edilir.

$$F(l,r) = (r, l \oplus f(r,k)) \tag{2.5}$$

F fonksiyonu iki parça giriş alır (çevrim girişinin sağ ve sol yarımları) ve iki parça çıkış üretir. *f* çevrim içerisinde kullanılan bir fonksiyondur ve çevrim anahtarını giriş olarak alarak giriş bloğu üzerinde içlem yapar. Yayılma ve nüfuz etme özelliklerinin sağlanması için *f* fonksiyonu içerisinde s-kutuları ve yer değiştirme işlemleri vardır.

Festiel tipi şifreleme algoritmasının tersinin şifreleme algoritmasının kendisine eşit olması için son çevrimde sağ ve sol yarımların yer değiştirme işlemi bulunmaz.



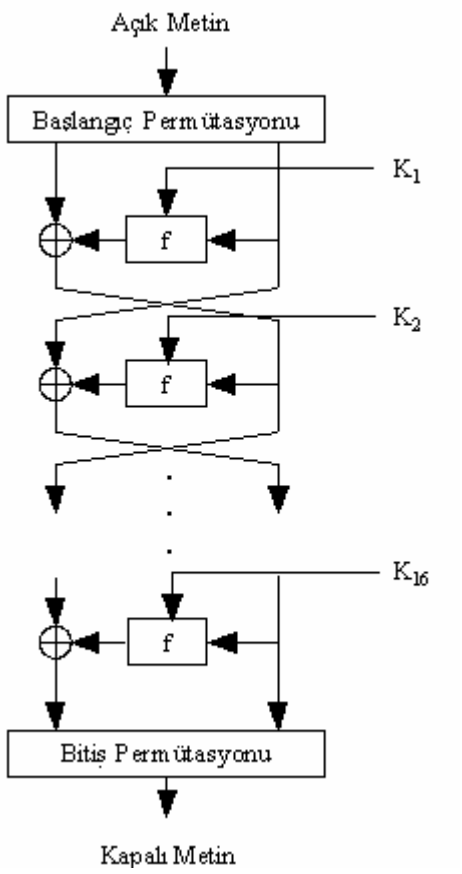
Şekil 2.8 : Feistel yapısı [7]

2.5 DES

Şekil 2.9'da yapısı bulunan DES algoritması, 1970 yılında IBM tarafından geliştirilen Lucifer algoritmasının biraz daha geliştirilmiş halidir [7]. DES, 1974 yılında IBM ve NSA'nın birlikte çalışması ile geliştirilmiştir. Algoritmanın anahtar uzayının günün teknolojisi karşısında küçük olması nedeniyle algoritma artık saf hali ile kullanılmamaktadır. Değiştirilmiş halleri kullanılmaya devam etmektedir.

2.5.1 Algoritma

DES algoritması Feistel yapısında olup 16 döngüden oluşur. İlk döngü girişinden önce başlangıç permütasyonu, son döngüden sonra da başlangıç permütasyonunun tersi vardır. Başlangıç permütasyonunun güvenlik açısından bir etkisi bulunmaz. Blok uzunluğu 64 bittir. 64 bitlik anahtarın 56 biti kullanılır, diğer 8 bit eşlik biti olarak kullanılabilmektedir.



Şekil 2.9 : DES algoritması [7]

Başlangıç permütasyonu olan IP Tablo 2.1'de bulunmaktadır. *i*. satırın *j*. sütununda bulunan değer (*i* ve *j* 0'dan başlamaktadır) permütasyon çıkışının (8*j*+*i*). bitinin girişin hangi bitine karşılık geldiğini ifade eder. Giriş ve çıkış bitlerinin indeksleri de 0'dan başlamaktadır. Bu bölümde anlatılan tabloların okunuşları Tablo 2.1'in okunuşu gibidir.

Tablo 2.1 : DES algoritması başlangıç permütasyonu

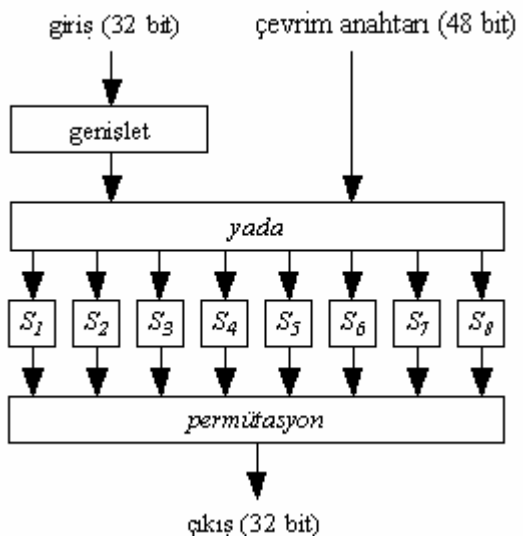
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Başlangıç permütasyonunun tersi (IP^{-1}) Tablo 2.2'de bulunmaktadır.

Tablo 2.2 : DES algoritması çıkış permütasyonu

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

f fonksiyonu içerisinde 32 bitlik giriş 48 bite genişletilir ve çevrim anahtarı ile bit bazında *dar veya* işlemine sokulur. Bit bazında işlem demek her iki girişteki *i*. bitlerin kendi aralarında *dar veya* işlemine girmesi demektir. *Dar veya* işlemi sonucunda oluşan 48 bitlik veri 6'şar bitler bazında 8 S-kutusuna girer. Her bir S-kutusu 6 bite karşılık 4 bitlik çıkış üretir. S-kutuları çıkışı oluşan 4x8=32 bitlik veri çevrim permütasyonundan geçirilir ve *f* fonksiyonunun çıkışı elde edilir. *f* fonksiyonu grafiksel olarak Şekil 2.10'da bulunmaktadır.



Şekil 2.10 : DES algoritması *f* fonksiyonu

f fonksiyonu içerisinde kullanılan genişletme fonksiyonu Tablo 2.3'te bulunmaktadır.

Tablo 2.3 : DES çevrim fonksiyonu içerisindeki genişletme fonksiyonu

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

f fonksiyonu içerisinde kullanılan S-kutuları Tablo 2.4'te bulunmaktadır. S-kutuları 4x16 boyutunda matristen oluşur. Satırları, girişin ilk ve son bitleri indeksler, sütunları da girişlerin ortada kalan 4 biti indeksler. Örneğin S1'e 001011 girişi geldiğinde çıkış 01 = 1. satırın 0101 = 5. sütunudur. Her bir gözdeki değerler 10'luk tabandadır.

Tablo 2.4 : DES algoritması s-kutuları

S_1															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

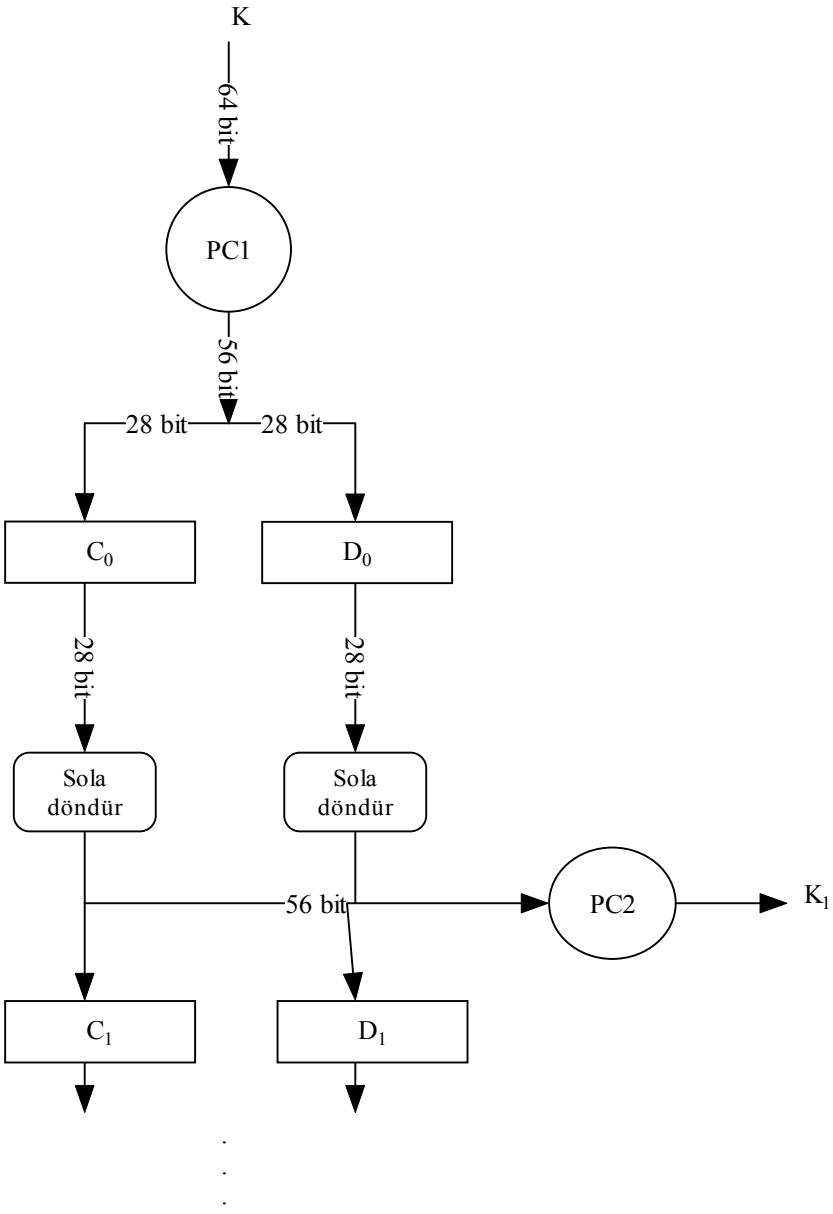
.Çevrim sonundaki permütasyon Tablo 2.5'te bulunmaktadır.

Tablo 2.5 : DES algoritması çevrim permütasyonu

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

2.5.2 Anahtar Üreteci

64 bitlik anahtardan 16 adet 48 bitlik anahtar üretilir. Grafiksel olarak Şekil 2.11'deki gibidir.



Şekil 2.11 : DES çevrim anahtarı üreteci

PC1 ile 64 bitlik anahtardan 56 bit seçilir. Yapısı Tablo 2.6'daki gibidir.

Tablo 2.6 : DES anahtar üreteci PC1 permütasyonu

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

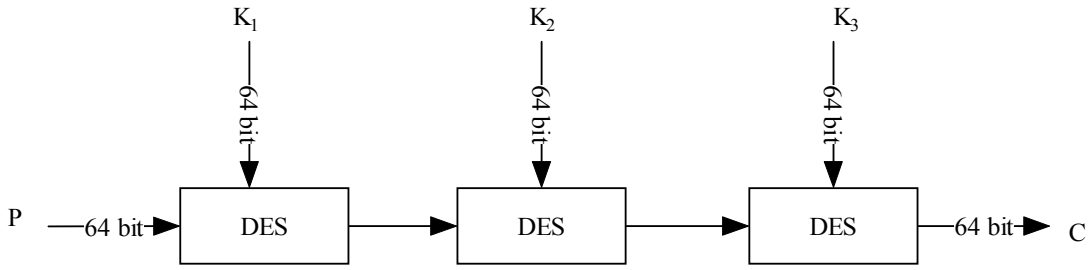
Çıkan 56 bit iki eşit parçaya ayrılır ve sola döndürme işlemi yapılır. Sola döndürme miktarları sırayla 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1 şeklindedir. Sola döndürme işleminden sonra PC2 ile 56 bitten 48 bit seçilir. PC2 Tablo 2.7'de bulunmaktadır.

Tablo 2.7: DES anahtar üreteci PC2 fonksiyonu

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

2.6 3DES

DES algoritmasında 56 bit anahtar kullanılması bilgisayar teknolojisinin gelişmesiyle birlikte DES'e karşı deneme-yanılma saldırısını güçlendirildi [7]. Bu nedende art arda 3 DES şifrelemesini içeren 3DES [4] kullanılmaktadır. Anahtar uzunluğu 56 x 3 = 168 bit olduğu için de deneme yanılma ile anahtar bulma günün teknolojisi ile imkansız hale gelmiştir. Yapısı Şekil 2.12'te görülmektedir.



Şekil 2.12 : 3DES

3. FARKSAL KRİPTOANALİZ

Bu bölümde güçlü klasik kripto analiz yöntemlerinden bir tanesi olan farksal (diferansiyel) kripto analiz anlatılmaktadır.

3.1 Giriş

Farksal kripto analiz DES, GDES, Lucifer, FEAL dahil olmak üzere bir çok sayıda blok şifre sistemine uygulanmış bir seçilmiş açık metin saldırısıdır [7,10]. Seçilmiş açık metin saldırısı, istenilen açık-kapalı metin çiftlerinin elde edilip kullanılabildiği saldırı demektir [7]. İlk olarak Biham ve Shamir tarafından geliştirilmiş olup DES'in indirgenmiş döngü çeşitlerine uygulandıktan sonra 16 döngülü DES algoritmasına uygulandı [10]. Farksal kripto analiz, açık metin farklarının algoritma içerisinde ilerlemesine dayanmaktadır. Farkın algoritmada ilerlemesi istatistiksel olarak hesaplanır ve çıkışta bu istatistiksel özellik gözlenir. İstatistiksel olmasının nedeni algoritma içerisinde yer alan ve doğrusal olmayan S-kutularıdır. Fark, iki farklı girişin *dar veya* işlemi sonucu olarak tanımlanır.

3.2 Yerine Koyma- Yer Değiştirme Ağı Üzerinde Farksal Kripto Analiz

Yerine koyma- yer değiştirme ağı üzerinde farksal kripto analiz basit bir ağ üzerinde anlatılmaktadır [6]. Ağ, Şekil 3.1'de bulunmaktadır. Blok uzunluğu 16 bit olup her döngüde 4x4 boyutunda birbirinin aynı 4 S-kutusu vardır ve kullanılan permütasyon ise Şekil 3.1'de görüldüğü gibidir. Örnekte, ağ 5 çevrimden oluşmakta ve son çevrimde permütasyon işlemi yapılmaktadır. Ayrıca çıkışta anahtar ile bit bazında *dar veya* işlemi yapılmaktadır. Anlatılacak yöntem ile K5, yani çıkışta kullanılan anahtar bulunacaktır. Farksal analiz için öncelikle S-kutularının farksal özellikleri çıkartılmalıdır. S-kutularının fark tablosu Tablo 3.1'de bulunmaktadır. Fark tablosu bir giriş farkına karşı bir çıkış farkının kaç farklı giriş ile sağlandığını gösterir. 16 farklı giriş olduğundan bu 16 farklı girişi bir farka götüren tek bir giriş bulunur. Bu, bir fark için 16 farklı giriş çifti demektir. Bu nedenle s-kutusunun fark tablosunda bir satırda bulunan sayıların toplamı 16'dır.

Tablo 3.1 : S-kutusu fark tablosu

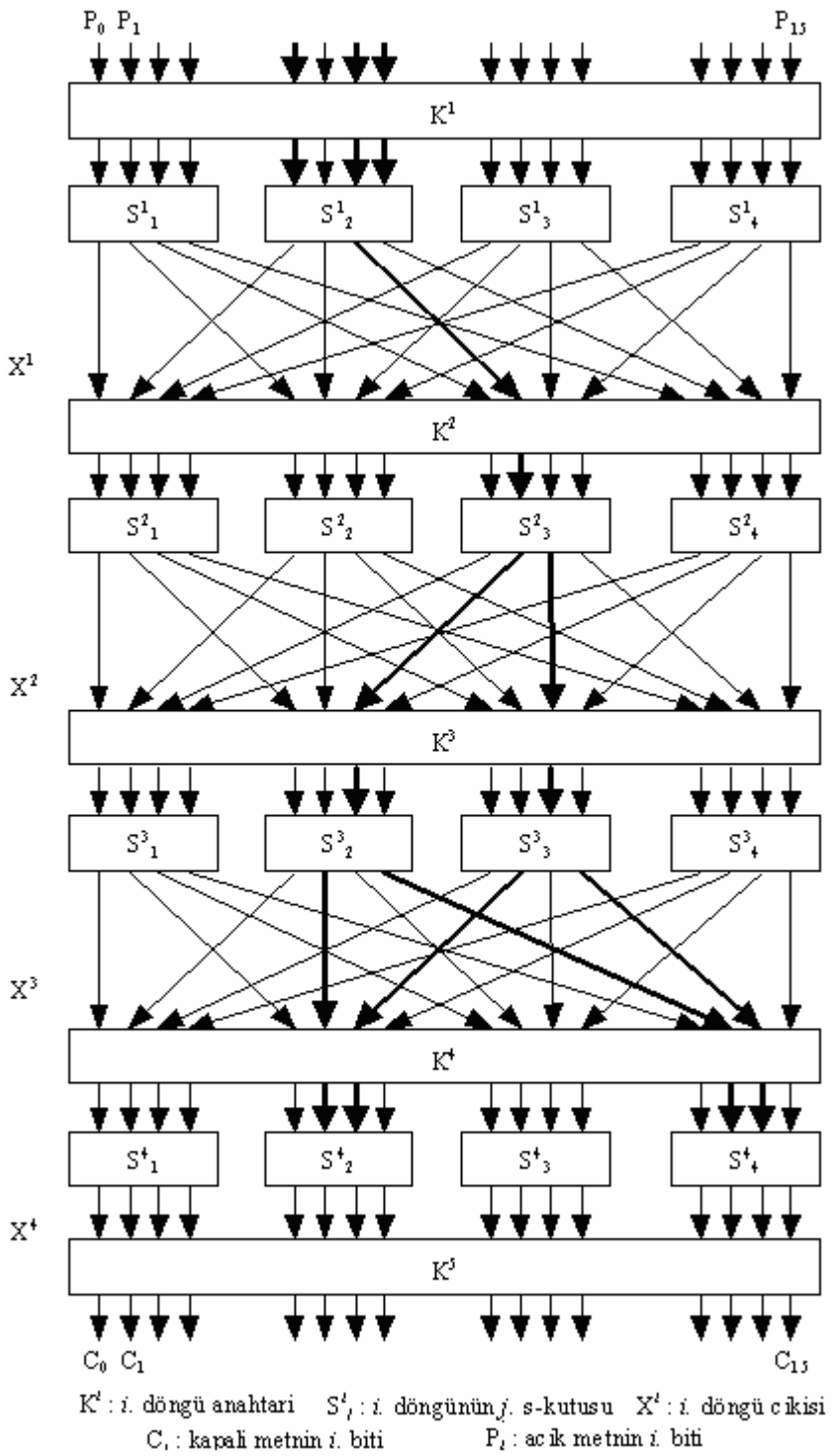
giriş farkı	çıkış farkı															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	2	0	0	0	2	0	2	4	0	4	2	0	0
2	0	0	0	2	0	6	2	2	0	2	0	0	0	0	2	0
3	0	0	2	0	2	0	0	0	0	4	2	0	2	0	0	4
4	0	0	0	2	0	0	6	0	0	2	0	4	2	0	0	0
5	0	4	0	0	0	2	2	0	0	0	4	0	2	0	0	2
6	0	0	0	4	0	4	0	0	0	0	0	0	2	2	2	2
7	0	0	2	2	2	0	2	0	0	2	2	0	0	0	0	4
8	0	0	0	0	0	0	2	2	0	0	0	4	0	4	2	2
9	0	2	0	0	2	0	0	4	2	0	2	2	2	0	0	0
A	0	2	2	0	0	0	0	0	6	0	0	2	0	0	4	0
B	0	0	8	0	0	2	0	2	0	0	0	0	0	2	0	2
C	0	2	0	0	2	2	2	0	0	0	0	2	0	6	0	0
D	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
E	0	0	2	4	2	0	0	0	6	0	0	0	0	0	2	0
F	0	2	0	0	6	0	0	0	0	4	0	2	0	0	2	0

Tabloyu yorumlamada kolaylık açısından bir örnek vermek gerekirse, giriş farkı 5 (0101) ve çıkış farkı 1 (0001) olan 4 farklı giriş çifti bulunduğu tablodan görülmektedir.

Şekil 3.1 iki farklı girişin paralel olarak aktığı düşünülerek yorumlanmalıdır. İki verinin algoritma içerisinde akışında koyu oklara karşılık gelen bitler farklı, diğer bitler aynıdır.

Farkı (0000 1011 0000 0000) olan iki farklı açık metin seçilsin. Bu fark sadece S^1_2 kutusuna girer. Kutu giriş farkı 1011 = B olan giriş çiftlerinden 8 tanesinde çıkış farkını 0010 = 2 vermektedir. Girişe uygulanan fark s-kutusundan 8/16 = 1/2 olasılıkla şekil 3.1'de görüldüğü gibi çıkışa yansımaktadır. İlk döngü sonucunda fark X^1_{10} bitine yansır. Fark ikinci döngüde S^2_3 kutusuna (0100) olarak yansır. S-kutusu giriş farkı 4 olan 6 farklı giriş çifti için çıkış farkını 6 (0110) vermektedir. İkinci çevrim sonucunda iki açık metin farkı 1/2 x 6/16 = 3 / 16 olasılıkla (0000 0010 0010 0000) olmaktadır. Bu fark 3. çevrimde S^3_2 ve S^3_3 kutularına yansır. S-kutusu giriş farkı 2 olan 6 giriş çifti için çıkış farkını 5 vermektedir. 3. çevrimde iki s-kutusu da birlikte çıkıştaki farkı sağladığı için çıkışın Şekil 3.1'de görüldüğü gibi olması olasılığı 6/16 x 6/16 = 9/64'tür. Açık metin farkının 3 döngü sonunda (0000 0110 0110 0000) olmasının olasılığı 3/16 x 9/64 = 27/1024'tür. Bir giriş farkı için 3. çevrim çıkışının farkının Şekil 3.1'de gösterildiği gibi olma olasılığı 27/1024

olduğundan, üçüncü döngü sonucu Şekil 3.1'de gösterildiği gibi fark oluştuğunun görülmesi için yaklaşık 1024/27 giriş çifti seçilmelidir.



Şekil 3.1 : Farksal kripto analizi

Giriş farkları (0000 1011 0000 0000) olan açık-kapalı metin çiftleri seçilir. 16 farklı $K^5_4K^5_5K^5_6K^5_7$ değeri için S^4_2 nin girişine tersten gidilir ve 0110 farklını sağlayan çiftler sayılır. Farkı en fazla sağlayan aday anahtar doğru kabul edilir. Benzeri işlem

16 farklı $K^5_8 K^5_9 K^5_{10} K^5_{11}$ değeri içinde S^4_2 girişine tersten gidilerek yapılır ve doğru anahtar parçası bulunur. Çıkışta kullanılan anahtarın 8 biti bulunmuş olunur. Başka fark akışı seçilerek diğer 8 bit de bulunur. Eğer anahtar üretme algoritması tersi alınabilir ise şifreleme algoritması çıkışında bulunan anahtar ile ana anahtar bulunur. Tersi alınabilir değilse şifreleme algoritmasının döngü sayısı bir azaltılmış olduğundan aynı işlemler K^4 ü bulmak için yapılır.

4. YAN KANAL SALDIRILARI

Bu bölümde, yan kanal saldırıları ile ilgili kısa bilgiler verilmektedir.

4.1 Yan Kanal Saldırısı

Yan kanal saldırısı ile anahtar, algoritmanın zayıflığından yararlanma yerine algoritmanın elektroniksel uygulanması sonucu çalıştığı elektriksel ortamın sağladığı bilgilerden yararlanarak bulunur. Bu bilgiler zamanlama bilgisi, güç tüketimi olabilmektedir [11,12].

Yan kanal saldırısı klasik kripto analizden daha güçlü olabilirken şifreleme yapılan sisteme fiziksel erişim gereklidir. Yan kanal saldırılarının uygulandığı cihazlar genelde ağ üzerindeki bir bilgisayar yerine basit elektronik cihazlardır.

4.2 Saldırı Çeşitleri

4.2.1 Zamanlama Saldırısı

Algoritmanın uygulanmasından doğan zayıflıktan faydalanılır. Algoritma içerisindeki parçaların çalışma süresinin analizine ve sürenin veriye bağlılığının incelenmesine dayanır [11]. Veriye bağımlılık genelde koşullu dallanmalardan kaynaklanmaktadır. Elektronik ortamda bulunan algoritmanın bilinmesi saldırıyı güçlendirmektedir.

Zamanlama bilgileri elde edilirken oluşan gürültüleri yok etmek için yeterli sayıda ölçüm yapmak ve istatistiksel filtreler kullanmak gerekir.

Zamanlama saldırısına örnek olarak *mod* alma işlemi verilebilir. x mod n algoritma içerisinde bir işlem olsun ve n gizli bilgi olsun. x n'den küçükken bir işlem yapılmazken x n'den büyükken çıkarma işlemi yapılır. x'e 0'dan başlayıp büyüterek değerler verilir, zaman açısından ani bir büyüme görüldüğünde büyümeye neden olan x değeri n'dir denir.

4.2.2 Güç Analizi Saldırısı

Şifreleme algoritması elektronik sistem üzerinde çalışır ve elektronik sistem transistörlerden oluşur. Transistörler anahtar gibi davranmaktadır. Eğer anahtar 0 ise akım bloke olur, 1 ise akım oluşur. 0'dan 1'e geçişler akım çekilmesine neden olur. Güç analizi ile akım çekme farklarından yararlanılarak bilgi elde edilmeye çalışılır. Genelde bilinen bir algoritmanın hangi safhasının işlediğini anlamak için yardımcı bir yöntem olarak kullanılır.

Güç analizi 2'ye ayrılır:

Basit Güç Analizi (*SPA, Simple Power Analysis*)

Sadece işlemcinin harcadığı enerji analiz edilir.

Farksal Güç Analizi (*DPA, Differential Power Analysis*)

Basit güç analizine ek olarak istatistiksel analizlerden yararlanarak ve hata düzeltme algoritmaları çalıştırarak gerekli örnek sayısını azaltılır.

4.2.3 Hata Analizi Saldırısı

Algoritma işletilirken sıcaklık, elektrik gerilimi gibi çevresel bazı parametreler değiştirilerek işlemcinin hata yapması sağlanır. Yanlış sonuçlar elde edilip doğru sonuçlarla karşılaştırılarak bilgi elde edilmeye çalışılır.

4.2.4 Akustik Analiz

Elektronik devreden çıkan ve duyulabilen veya duyulamayan seslerin hassas cihazlarla kaydedilip incelenmesi sonucu bilgi elde etmeye çalışılır. Shamir tarafından sadece işlemciden çıkan ses ile zamanlama saldırısı yapılabileceği ispatlandı.

4.2.5 Elektromanyetik Analiz

Elektronik devrede oluşan elektromanyetik alanın izlenip değerlendirilmesi ile yapılan yan kanal saldırısı yöntemidir.

4.3 Saldırılara Karşı Önlemler

Elektronik ve fiziksel verilerden bilgi edilebilmesini imkansız hale getirmek için bazı önlemler geliştirildi.

4.3.1 Zamanlama Saldırısına Karşı Önlemler

Gecikme eklemek zamanlama saldırısına karşı bir önlem olabilir. Tüm işlemlerin aynı sürede bitmesi sağlanarak saldırganların işlem süresini hesaplayarak bilgi elde etmesi engellenebilmektir. Bu yöntem ile tüm işlemler süresi en uzun işlem süresine eşitleneceği için algoritma üzerinde süre iyileştirmesi yapılamaz duruma gelir.

Çarpma ve üs alma sürelerinin eşitlemek de zamanlama saldırılarına karşı etkili olmaktadır.

4.3.2 Güç Analizini Saldırısına Karşı Önlemler

Güç analizi saldırısına karşı geliştirilen önlemlerden bir tanesi harcanan gücün dengelenmesidir. Dengeleme için gereksiz saklayıcılar ve kapılar kullanılır. Dengelemenin sağlanması için diğer bir yöntem de başka bir donanımın enerji harcamayı bir seviyede tutma çalışmasıdır.

Güç analizini imkansız hale getirmek için geliştirilen diğer bir yöntem de uygulama içerisine gereksiz işlemler konulmasıdır. Gereksiz işlemlerden dolayı güç tüketiminde meydana gelen gürültüler analizi zorlaştırmaktadır.

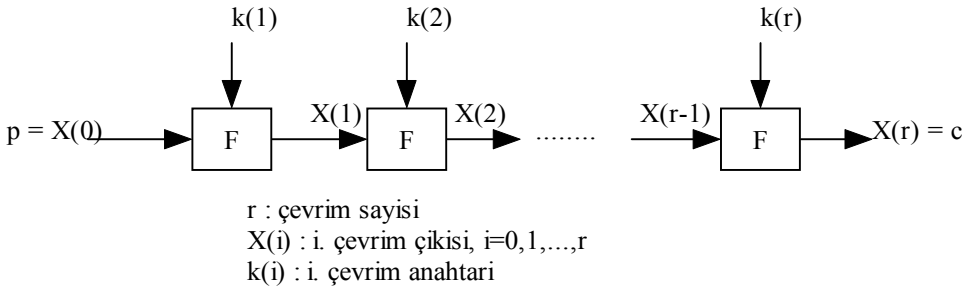
5. ÇAKIŞMA SALDIRISI

Bu bölümde, tezde genel tanımı yapılan çakışma saldırısı anlatılmaktadır.

5.1 Saldırı Tanımı

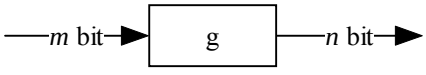
Çakışma saldırısı şifreleme algoritması içerisinde meydana gelen çakışmalara dayanır. Çakışma bir fonksiyonun farklı iki giriş için aynı çıkışı üretmesidir.

Şekil 5.1'de yapısı gösterilen iteratif blok şifreleyicilerde, çevrim içerisinde çakışma meydana gelebilir. *F,* çevrim fonksiyonu olmak üzere birkaç parçadan (fonksiyondan) oluşur. Parçalardan bir veya birkaç tanesi doğrusal olmayan fonksiyonlardır.



Şekil 5.1 : İteratif blok şifreleme

Şekil 5.2'de gösterilen *g* fonksiyonu, *F* fonksiyonu içerisinde olsun ve *m* bitlik giriş, *n* bitlik çıkış üretsin ve ayrıca *m*>*n* olsun.



Şekil 5.2 : Çakışma oluşturan fonksiyon

g fonksiyonunun giriş uzayı kümesi *A* ve çıkış uzayı kümesi *B* olsun. $|A| = 2^m$ ve $|B| \leq 2^n$ olur. *m*>*n* olduğundan |A|>|B|'dir. Giriş uzayının eleman sayısı çıkış uzayının eleman sayısından büyük olması bazı farklı giriş çiftleri için aynı çıkışın oluşması sonucunu doğurur. Δ giriş farkları kümesi olmak üzere $|\Delta| = 2^m$'dir. α çakışma bağıntısı $\alpha = \{(x, y) \mid x \in A, y \in \Delta, g(x) = g(x \oplus y)\} \subset A x \Delta$ şeklinde

tanımlansın. Bağıntılar kümesi olan γ şu şekilde tanımlansın;

$$\gamma = \{\alpha_i \mid \alpha_i \subset \alpha \wedge i \in \{0,1,...,2^m - 1\} \wedge [(x,y) \in \alpha_i \wedge (u,v) \in \alpha_i \Rightarrow y = v] \wedge$$
$$[(x,y) \in \alpha_i \wedge (u,v) \in \alpha_j \wedge i \neq j \Rightarrow y \neq v] \wedge [\alpha_0 \cup \alpha_1 \cup ... \cup \alpha_{i-1} = \alpha]\}$$

Eşitlik 5.1'de tanımlanan *h,* girişi α_i kümesi olan ve çıkışı *m* bitlik vektör olan bir fonksiyon olsun.

$$h(\alpha_i)[j] = \begin{cases} x[j] & ,\forall (x,y),(u,v) \in \alpha_i, x[j] = u[j] \\ -1 & ,\text{diğer} \end{cases} \quad \textbf{(5.1)}$$

[*i*], vektörün *i.* biti anlamında kullanılmıştır. *h* fonksiyonu, bir giriş farkı ile girişteki hangi bitlerin bulunabileceğini verir. *h* fonksiyonu ile bir çevrim içerisinde çakışma durumunda *g* fonksiyonunun girişinde bulunan bazı bitler öğrenilir. *g* fonksiyonu girişine gelen bitlerden bazıları bilinerek de gizli olan anahtar bitlerinden bazıları bulunabilir.

Çevrim fonksiyonu içerisinde çakışma olması ise yan kanal bilgileri ile elde edilir. *g* fonksiyonu, farklı iki giriş için aynı çıkış vereceğinden *g* fonksiyonu çıkışında belli bir komut dizisi boyunca aynı veri işlenir. Bir komut dizisi boyunca aynı verinin işlenmesi kaynaktan çekilen akımın o komut dizisi boyunca aynı olmasına neden olur. Giriş farkları belli olan çiftler için şifreleme işlemi sırasında çekilen akım incelenerek çakışma durumu sezilir. Fark ve çakışma belli olduğundan dolayı da döngü içerisinde bulunan bazı bitler bulunur ve bu bitler ile de döngü anahtarının bazı bitleri bulunur.

Çoğu blok şifreleme işleminde giriş kümesinin eleman sayısı çıkış kümesinin eleman sayısından fazla olan fonksiyonlar bulunmaz. Çakışma saldırısı bu tip algoritmalara parçalı çakışma kavramı ile uygulanabilir. Parçalı çakışma, iki farklı giriş için fonksiyonun çıkışının bir kısmının aynı olması demektir.

6. ÇAKIŞMA SALDIRISI UYGULAMALARI

Bu bölümde, tezde tanımlanan çakışma saldırısının önceden yapılmış uygulamaları ile tezde yapılan saldırı uygulamaları yer almaktadır. Tezde yapılan uygulamalardan bir tanesi DES algoritmasının 1. ve 2. çevrimlerine yapılan saldırılardır. Diğer bir uygulama da 3DES algoritmasına saldırının nasıl uygulanabileceği ile ilgilidir.

6.1 Literatürde Varolan Çalışmalar

Son zamanlarda çakışma saldırısı ile ilgili yapılmış çalışmalar bulunmaktadır.

2003 yılında Schramm ve diğerleri tarafından DES algoritması üzerinde saldırı uygulanmıştır [3]. Saldırı, DES algoritmasındaki s-kutularının girişlerinin 6 bit çıkışlarının ise 4 bit olmasına dayanmaktadır. Tezde DES üzerindeki saldırı detaylı incelendi ve s-kutuları analiz edildi.

Saldırı, Schramm ve diğerleri tarafından 2004'te AES [13] algoritmasına uygulanmıştır [5]. AES algoritmasında DES algoritmasında olduğu gibi çakışmaya neden olan fonksiyonlar bulunmamaktadır, yani fonksiyonların giriş uzay kümesi ile çıkış uzay kümesi aynıdır. AES için kısmi çakışma uygulanmıştır. Kısmi çakışma çıkışın bir bölümünde meydana gelen çakışmadır.

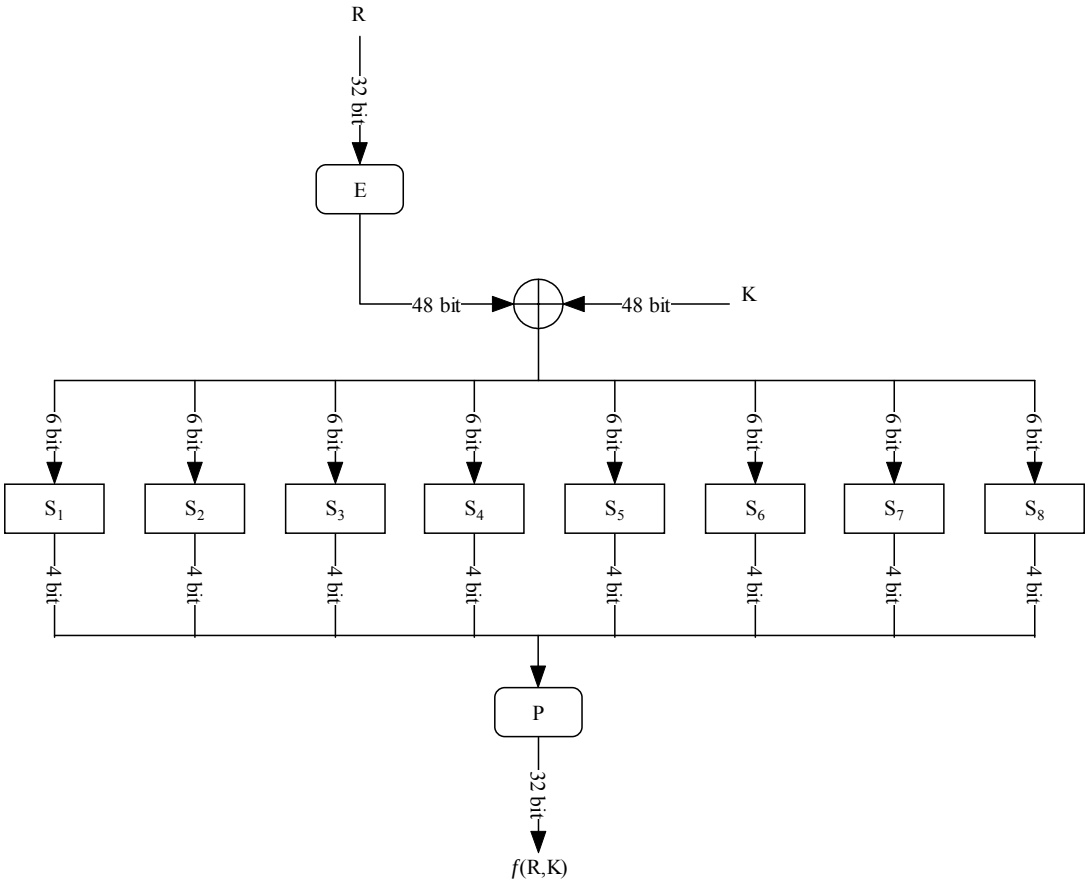
6.2 DES Algoritmasına Saldırı

DES algoritmasının başında ve sonunda bulunan permütasyon işlemleri (IP, IP^{-1}) çakışma saldırısına etki etmemektedir. Saldırının kolay anlaşılabilir olması için IP ve IP^{-1} işlemleri algoritmada yokmuş gibi davranılacaktır.

6.2.1 Çevrim Fonksiyonu

Fesitel yapısında olan DES algoritmasının çevrim fonksiyonu $F(L,R) = (R, L \oplus f(R,K))$ dir. L ve R çevrim girişinin 32'şer bitlik sol ve sağ yarımlarını, K ise 48 bitlik çevrim anahtarını ifade etmektedir. f ise çevrim içerisinde kullanılan fonksiyondur. f fonksiyonu, 32 bitlik girdiyi 48 bite genişletir ve çevrim

anahtarı ile *dar veya* işlemine tabi tutar. Çıkan sonuç s-kutularından geçirilir ve sonuca permütasyon uygulanır. Permütasyon çıkışı *f* fonksiyonunun çıkışıdır. Grafiksel olarak Şekil 6.1'de görülmektedir.

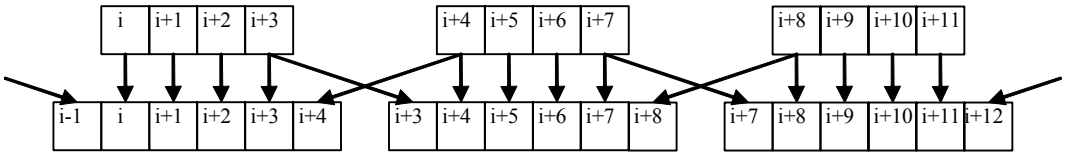


Şekil 6.1: DES *f* fonksiyonu

f fonksiyonu içerisindeki s-kutularının giriş uzayı çıkış uzayından büyük olduğundan bazı girişler için çakışma meydana gelmektedir. Giriş uzayı 26 elemandan oluşurken çıkışı uzayı 24 elemandan oluşmaktadır. Burada dengeli bir dağılım oluşur, yani çıkıştaki her bir değere girişten 4 değer karşılık gelir. Böylece bir s-kutusu için C(4,2) . 24 = 96 farklı giriş çiftleri aynı çıkışa neden olmaktadır.

Çakışma oluşturmak için olabildiğince az s-kutusu aktif olarak kullanılmalıdır. Aktif olarak kullanılmak, farklı iki girişin değişen bitlerinin s-kutusuna yansıması demektir. Az sayıda aktif s-kutusu kullanılması, çakışma oluşturmak için denenmesi gereken açık metin çiftinin az olmasını sağlar. Çakışma oluşturmak için tek bir s-kutusu kullanılmış olsa, bir s-kutusuna açık metnin 6 biti yansıdığından denenmesi gereken açık metin sayısı en fazla $2^6/2=32$ olur.

DES algoritmasında s-kutularına gelen giriş önce genişletme fonksiyonundan geçtiği için girişte meydana gelen bir değişiklik s-kutu girişlerine aynen yansımayabilir. 32 bitlik $X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}, X_{11}, X_{12}, X_{13}, X_{14}, X_{15}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{21}, X_{22}, X_{23}, X_{24}, X_{25}, X_{26}, X_{27}, X_{28}, X_{29}, X_{30}, X_{31}, X_{32}$ girişi, genişletme fonksiyonu sonucunda s-kutularına $X_{32}, X_1, X_2, X_3, X_4, X_5, X_4, X_5, X_6, X_7, X_8, X_9, X_8, X_9, X_{10}, X_{11}, X_{12}, X_{13}, X_{12}, X_{13}, X_{14}, X_{15}, X_{16}, X_{17}, X_{16}, X_{17}, X_{18}, X_{19}, X_{20}, X_{21}, X_{20}, X_{21}, X_{22}, X_{23}, X_{24}, X_{25}, X_{24}, X_{25}, X_{26}, X_{27}, X_{28}, X_{29}, X_{28}, X_{29}, X_{30}, X_{31}, X_{32}, X_1$ olarak yansır. *f* fonksiyonu girişinin s-kutuları girişlerine yansıması Şekil 6.2 ile görülmektedir.



Şekil 6.2 : *f* girişinin s-kutularına dağılımı

Bir adet s-kutusu üzerinde çakışma oluşturmak için s-kutusu girişindeki ilk ve son bitlerin değişmemesi gerekir, aksi halde değişiklik diğer s-kutularına yansır. Fakat ilk ve son bitler sabit iken s-kutusu bire-bir fonksiyon gibi davranmaktadır. Bu durum saldırılara karşı s-kutusunun sağlaması gereken ölçütlerden bir tanesidir [1]. Bir s-kutusu Şekil 6.3'te bulunmaktadır.

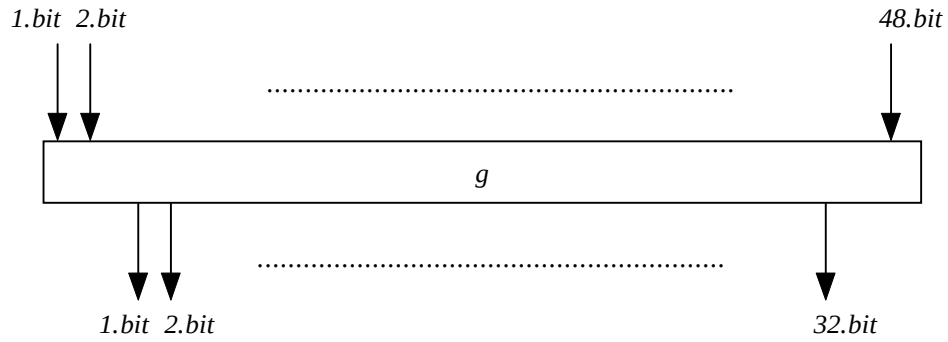
giriş = $g_0g_1g_2g_3g_4g_5$

g_0g_5 \ $g_1g_2g_3g_4$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	0010	1100	0100	0001	0111	1100	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1100	0011	1001	1000	0110
10	0100	0010	0001	1011	1100	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1100	0100	0101	0011

Şekil 6.3 : DES S_5-kutusu

En az peş peşe 3 aktif s-kutusu kullanılarak çakışma oluşturulabilmektedir [14].

g fonksiyonu, s-kutuları olsun. 48 bit giriş almakta ve 32 bit çıkış üretmektedir. *g* giriş bitlerinin indeksleri soldan 1 ile başlamaktadır. *g* fonksiyonu Şekil 6.4 de bulunmaktadır.



Şekil 6.4 : DES içerisinde tanımlanan g fonksiyonu

Çevrimdeki f fonksiyonu $f(R,K) = P(g(E(R) \oplus K))$ şeklinde yazılabilir. g fonksiyonu, giriş farklarının 3 s-kutusuna yansıdığı 258 fark için çakışma üretmektedir. 258 farktan 167 tanesi çakışma saldırısında kullanılabilirdir. Kullanılabilir olması, fark kullanılarak g fonksiyonu girişindeki bazı bitlerin bulunabilmesi demektir. Daha genel açıklama Bölüm 5'te yer almaktadır. 167 fark, bu farklar ile g girişinin hangi bitlerinin bulunabileceği ve bulunabilen bitlerin değerleri Tablo A.1'de bulunmaktadır. Farklar 3 s-kutusuna uygulanacağı için, bu s-kutularına giren bitlerin açık metindeki karşılıkları olan 14 bit değiştirilir.

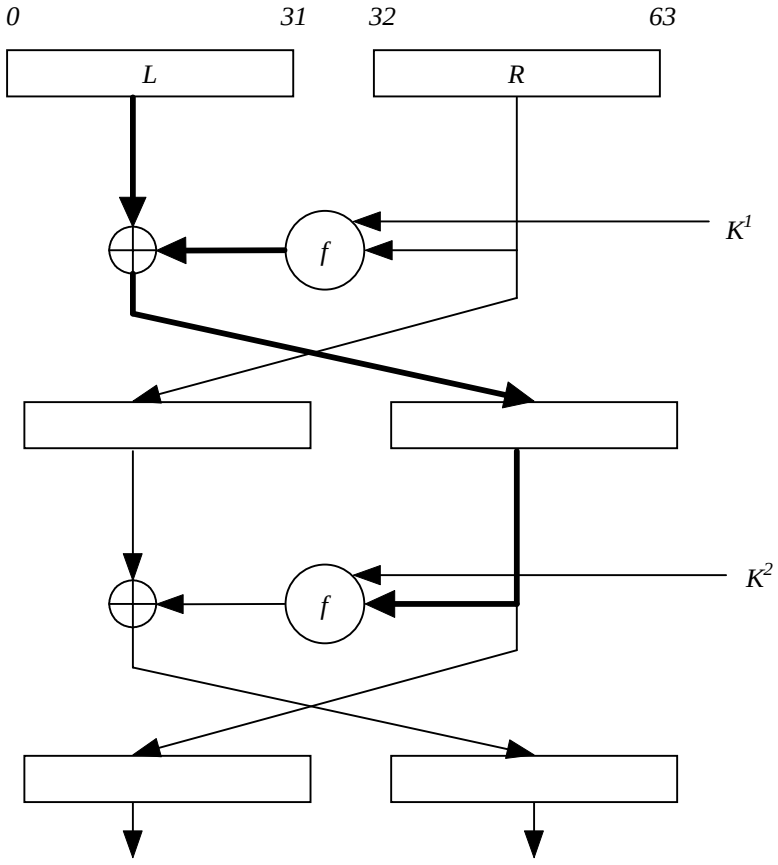
Tablo A.1'deki farklar kullanılarak çakışma ile g girişindeki bitlerden en fazla 37 bit bulunabilmektedir. Bulunabilen bitler 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 18, 20, 25, 26, 27, 28, 29, 30, 31, 32, 34, 36, 37, 38, 39, 40, 42, 43, 44, 46, 47 ve 48. bitlerdir. Bu bitler Tablo 6.1'deki 10 fark kullanılarak $10x2^{14}/2$=81920 açık metin çiftlerinin şifrelenmesi ile bulunabilmektedir.

Tablo 6.1 : Çevrim İçerisindeki 37 Biti Veren Farklar

Aktif s-kutuları	S-kutularına giren farklar	Bulunan anahtar bitleri	Bit değerleri
345	001011110010101000	13,14,16,26,28,29,30	1,0,0,0,1,0,1
812	001011110010101000	3,6,11,43,44,46	1,1,0,0,1,0
123	001011110010101100	1,2,4,10,12	1,0,1,1,1
234	000111110010101000	7,9,15,16,18	1,1,0,0,0
678	001111111010101100	44,47,48,31,32	0,1,0,0,1
456	000111110010101000	20,27,28,34	1,1,0,0
678	000111110010101000	42,32,39,40	0,1,1,0
781	001111110110101000	37,38,48	0,0,0
567	001011111010101000	28,36,25	1,0,1
234	000011110110101000	8,10,15,18	1,1,1,0

6.2.2 1. Çevrime Saldırı

1. çevrimdeki s-kutularının yani g fonksiyonunun girişleri, açık metnin sağ yarımı ile çevrim anahtarından oluşmaktadır. Açık metnini girişleri ve çakışma ile de g girişlerinin bazı bitleri bilindiğinden dolayı çevrim anahtarının bazı bitleri bulunabilmektedir. Çakışma oluşması ise 2. çevrimde çekilen akımın aynı olmasından anlaşılır. Buradaki yaklaşım görsel olarak Şekil 6.5'de bulunmaktadır. Şekil, farklı iki girişin paralel olarak algoritma içerisinde akması şeklinde düşünülmelidir. Koyu oklarla gösterilen kısımlardaki veriler iki farklı giriş için eşit anlamına gelmektedir. Şifrelenecek olan metin çiftleri (L, R) ve $(L, R \oplus \Delta)$ olsun. K^1, 1. döngü anahtarı olmak üzere $f(R, K^1) = f(R \oplus \Delta, K^1)$ olursa ikinci çevrimdeki f fonksiyonuna giren girişler $L \oplus f(R, K^1)$ ve $L \oplus f(R \oplus \Delta, K^1)$ eşit olmaktadır. Bu durumda ikinci çevrimdeki f fonksiyonu aynı akımı çekmektedir. Böylece çekilen akımın aynı olmasından çakışma olduğu anlaşılmaktadır.



Şekil 6.5 : DES üzerinde çakışma atağı

1. çevrim anahtarı oluşturan ana anahtar bitleri Tablo 6.2'de gösterilmiştir. İlk satır çevrim anahtar bitlerini, ikinci satır ise ana anahtar bitlerini göstermektedir. İlk satırda koyu olarak yazılan bitler, çakışma ile bulunabilen bitleridir.

Tablo 6.2 : 1. Çevrimde Çakışma ile Bulunabilen Anahtar Bitleri

1	**2**	**3**	**4**	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
10	51	34	60	49	17	33	57	2	9	19	42	3	35	26	25

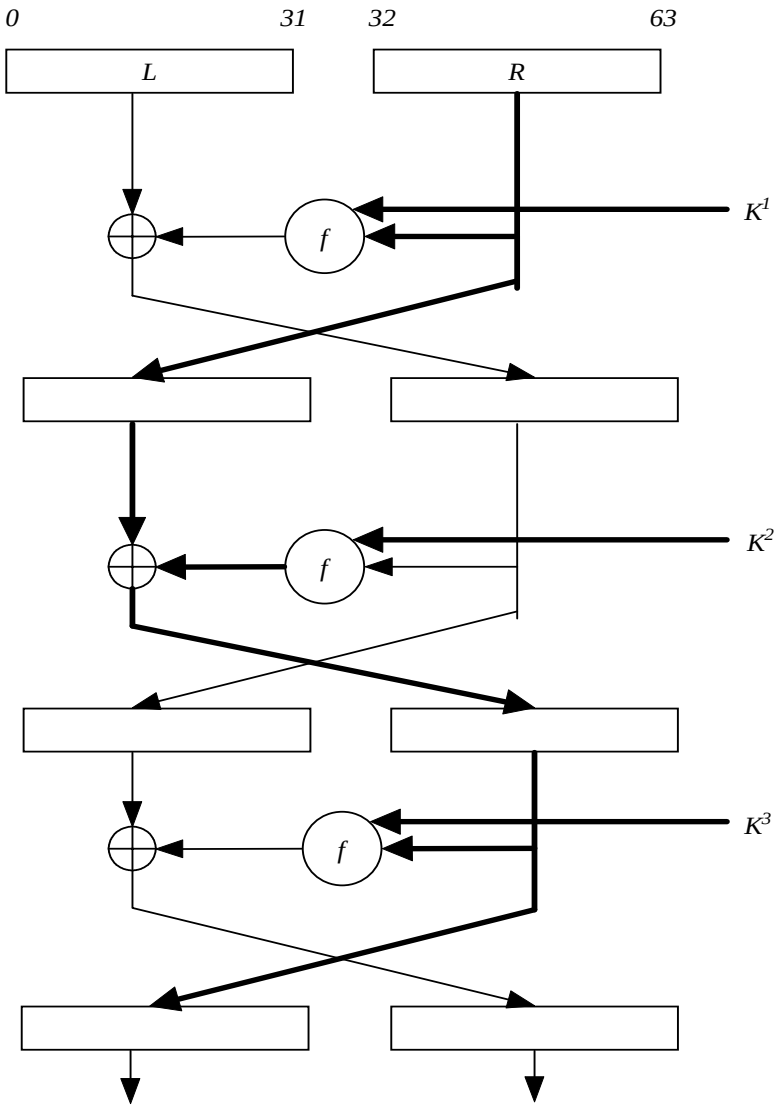
17	**18**	19	**20**	21	22	23	24	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**
44	58	59	1	36	27	18	41	22	28	39	54	37	4	47	30

33	**34**	35	**36**	**37**	**38**	**39**	**40**	41	**42**	**43**	**44**	45	**46**	**47**	**48**
5	53	23	29	61	21	38	63	15	20	45	14	13	62	55	31

1. çevrimde oluşturulan çakışma ile anahtarın 37 biti bulunmaktadır. 56 bitlik anahtarın diğer bitleri ise çok kısa bir sürede deneme yanılma ile elde edilebilmektedir. Tüm anahtar, 81920 metin çiftinin algoritmanın bulunduğu kart üzerinde şifrelenmesi ve herhangi bir bilgisayar üzerinde 2^{19} şifreleme işlemi ile bulunabilmektedir. Gerekli olan veri ise sadece bir adet bilinen açık-kapalı metin çiftidir.

6.2.3 2. Çevrime Saldırı

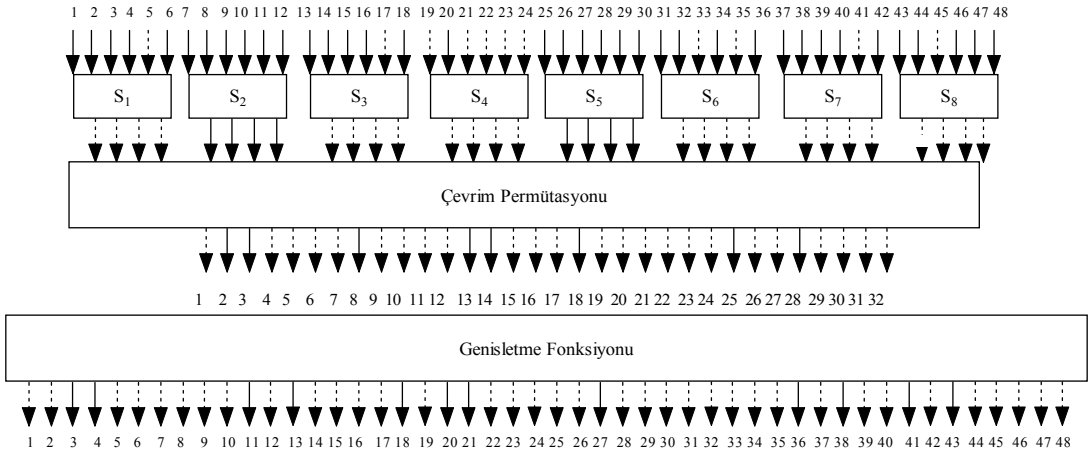
Saldırı 1.çevrimdekine benzer olarak ikinci çevrime genişletilebilir. 3. çevrimdeki *f* fonksiyonu nedeniyle çekilen akımların eşit olması ile çakışma olduğu anlaşılır. Saldırıdaki farkların akışı Şekil 6.6'da bulunmaktadır.



Şekil 6.6 : 2. Çevrime Saldırı

İkinci çevrimde *g* fonksiyonunun girişinde 2. çevrim anahtarı ve açık metin bitleri ile beraber 1. çevrim anahtar bitleri de bulunmaktadır. R_1'in (ikinci çevrim girişinin sağ yarımı) bilinen bitleri, 2. çevrimde bulunabilecek anahtar bitlerini etkilemektedir.

1. çevrime yapılan saldırı sonucu elde edilen anahtar bitleri ve bu bitlerin s-kutularına girişi ile 2. çevrime yansıyışı Şekil 6.7'de bulunmaktadır. Kesikli oklar bilinmeyen bitleri ifade etmektedir.



Şekil 6.7 : 1. çevrim sonucu bilenen bitler

2. çevrimdeki g girişini R_1 ve K^2 (2. çevrim anahtarı) oluşturduğu için 2. çevrim anahtarından R_1 ve g girişinin bilinen bitlerine karşılık gelen bitleri bulunabilir. Şekil 6.8 de görüldüğü gibi 2. çevrim anahtarının 3,4,11,13,18,20,27,36,38,43. bitleri bulunabilir.

R_1	1	2	**3**	**4**	5	6	7	8	9	10	**11**	12	**13**	14	15	16	17	**18**	19	**20**	**21**	22	23	24
K^2	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
⊕																								
g girisi	**1**	**2**	**3**	**4**	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	17	**18**	19	**20**	21	22	23	24

R_1	25	26	**27**	28	29	30	31	32	33	34	35	**36**	37	**38**	39	40	**41**	42	**43**	44	45	46	47	48
K^2	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
⊕																								
g girisi	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	33	**34**	35	**36**	**37**	**38**	**39**	**40**	41	**42**	**43**	**44**	45	**46**	**47**	**48**

Şekil 6.8 : 2. Çevrim Anahtarının Bulunabilen Bitleri

Tablo 6.3'de 2. çevrim anahtarını oluşturan ana anahtar bitleri bulunmaktadır. İlk satır çevrim anahtarı, ikinci satır ise ana anahtar bitleridir. Koyu olanlar bulunabilen çevrim anahtarlarıdır.

Tablo 6.3 : 2. Çevrim Anahtarını oluşturan Anahtar Bitleri

1	2	**3**	**4**	5	6	7	8	9	10	**11**	12	**13**	14	15	16	17	**18**	19	**20**	21	22	23	24
2	43	26	52	41	9	25	49	59	1	11	34	60	27	18	17	36	50	51	58	57	19	10	33

25	26	**27**	28	29	30	31	32	33	34	35	**36**	37	**38**	39	40	41	42	**43**	44	45	46	47	48
14	20	31	46	29	63	39	22	28	45	15	21	53	13	30	55	7	12	37	6	5	54	47	23

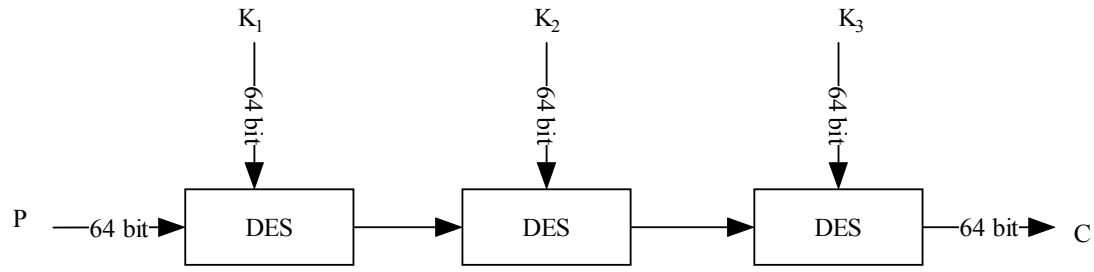
2. çevrime saldırı ile Tablo 6.4'te bulunan 7 fark kullanılarak ana anahtarın 11, 13, 21, 26, 31, 37, 50, 52, 58 ve 60. bitleri bulunabilir. Böylece 1. çevrim ile bulunamayan 11, 13, 50 ve 52. anahtar bitleri de bulunmuş olunur. Çakışma saldırısı ile kart üzerinde yaklaşık $17x2^{14}/2 = 2^{17}$ adet şifreleme işlemi yapılarak 56 bitlik anahtarın 41 biti bulunmuş olunur.

Tablo 6.4 : 2. çevrim için kullanılan farklar

aktif s-kutuları	s-kutularına giren farkla	bulunabilen anahtar bitleri	bit değerleri
812	001011110010101000	3,6,11,43,44,46	1,1,0,0,1,0
123	001011110010101100	1,2,4,10,12	1,0,1,1,1
345	001011110010101000	13,14,16,26,28,29,30	1,0,0,0,1,0,1
234	000111110010101000	7,9,15,16,18	1,1,0,0,0
456	000111110010101000	20,27,28,34	1,1,0,0
567	001011111010101000	28,36,25	1,0,1
781	001111110110101000	37,38,48	0,0,0

6.3 3DES Algoritmasına Saldırı

Şekil 6.9'da yapısı görülen 3DES algoritmasında 3 DES şifreleme işlemi peş peşe uygulanır.



Şekil 6.9 : 3DES

DES algoritmasında çakışma ile bulunamayan anahtar bitleri deneme yanılma ile bulunabilmektedir. 3DES algoritmasında ise K_1 anahtarının tamamı çakışma sağlanarak bulunamamaktadır. Bunun nedeni, ilk DES işlemi çıkışının bilinememesidir. Sadece çakışma ile K_1 anahtarının en fazla 41 biti bulunabilmektedir. Şifreleme ve şifre çözme işlemi DES algoritmasında aynı olduğundan çakışma saldırısı ile K_3 anahtarının da 41 biti bulunabilir. K_2 anahtarını bulabilmek için K_1 anahtarının 15 biti tahmin edilir ve her tahmin için K_2 anahtarının 41 biti bulunur. K_2 anahtarının da 15 biti tahmin edilir. K_3 anahtarı için de 15 bit tahmin edilir ve 2^{45} tahmin yapılarak deneme yanılma ile tüm anahtarlar bulunmuş

olunur. Saldırının toplam karmaşıklığı, kart üzerinde $2^{17} + 2^{15}x2^{17} + 2^{17} = 2^{32}$ ve herhangi bir bilgisayarda 2^{45} adet şifreleme işlemidir. Çakışma saldırısı, 3DES'e uygulanan saldırılardan daha iyi bir karmaşıklık sunmaktadır.

7. ÇAKIŞMA SALDIRISININ DES ÜZERİNDE GERÇEKLENMESİ

Bu bölümde Çakışma saldırısının gerçeklenmesi ile ilgili bilgiler verilmektedir.

7.1 Güç Harcama Simülasyonu

Çakışma saldırısının gerçeklenmesi için güç harcama ölçümü yerine simülasyon kullanılması tercih edildi. Bu nedenle güç harcama simülasyonu hazırlandı ve saldırı simülasyon kullanılarak gerçeklendi.

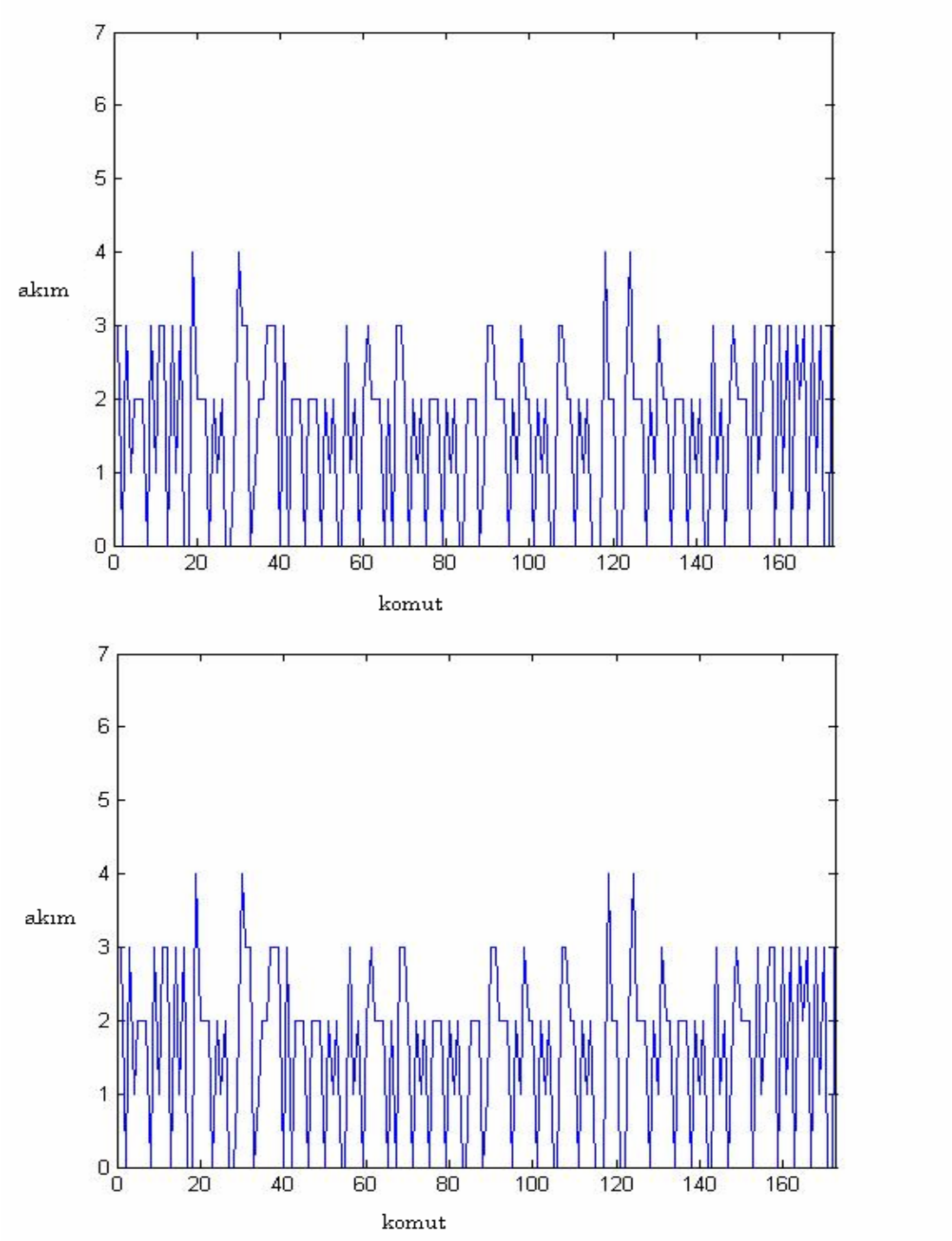
Algoritma içerisinde çakışma olduğu, seçilen farklı iki açık metin için şifreleme yapılırken aynı gücün harcanması ile anlaşılmaktadır. DES algoritmasında 1. çevrimde çakışma olması 2. çevrimde genişletme fonksiyonuna aynı girişlerin gelmesine neden olur. 2. çevrim genişletme fonksiyonu sırasında harcanan güç karşılaştırılarak çakışma olup olmadığı anlaşılabilmektedir. Simülasyonda genişletme fonksiyonunun makine kodu yazıldı ve yazılan makine kodunu yürütecek program hazırlandı. Program ayrıca her makine komutu sırasında çekilen akımı dosyaya kaydetmektedir. Komutun akım çekme miktarı için aşağıdaki model kullanıldı.

7.1.1 Güç Harcama Modeli

CMOS transistörlerde 0'dan 1'de geçişler güç harcanmasına neden olmaktadır. Her bir komutta bir saklayıcı değeri değişmektedir. Bir komut sırasında harcanan güç, değeri değişen saklayıcıdaki 0'dan 1'e geçişlerin sayısı olarak alındı. Saklayıcı 8 bitlik olduğundan harcanan güç 0-8 arası bir ayrık bir değer almaktadır. Harcanan güce 1 birimlik gürültü eklendi. 1 birimlik gürültü, çekilen akıma -1, 0 veya 1 eklenmesidir.

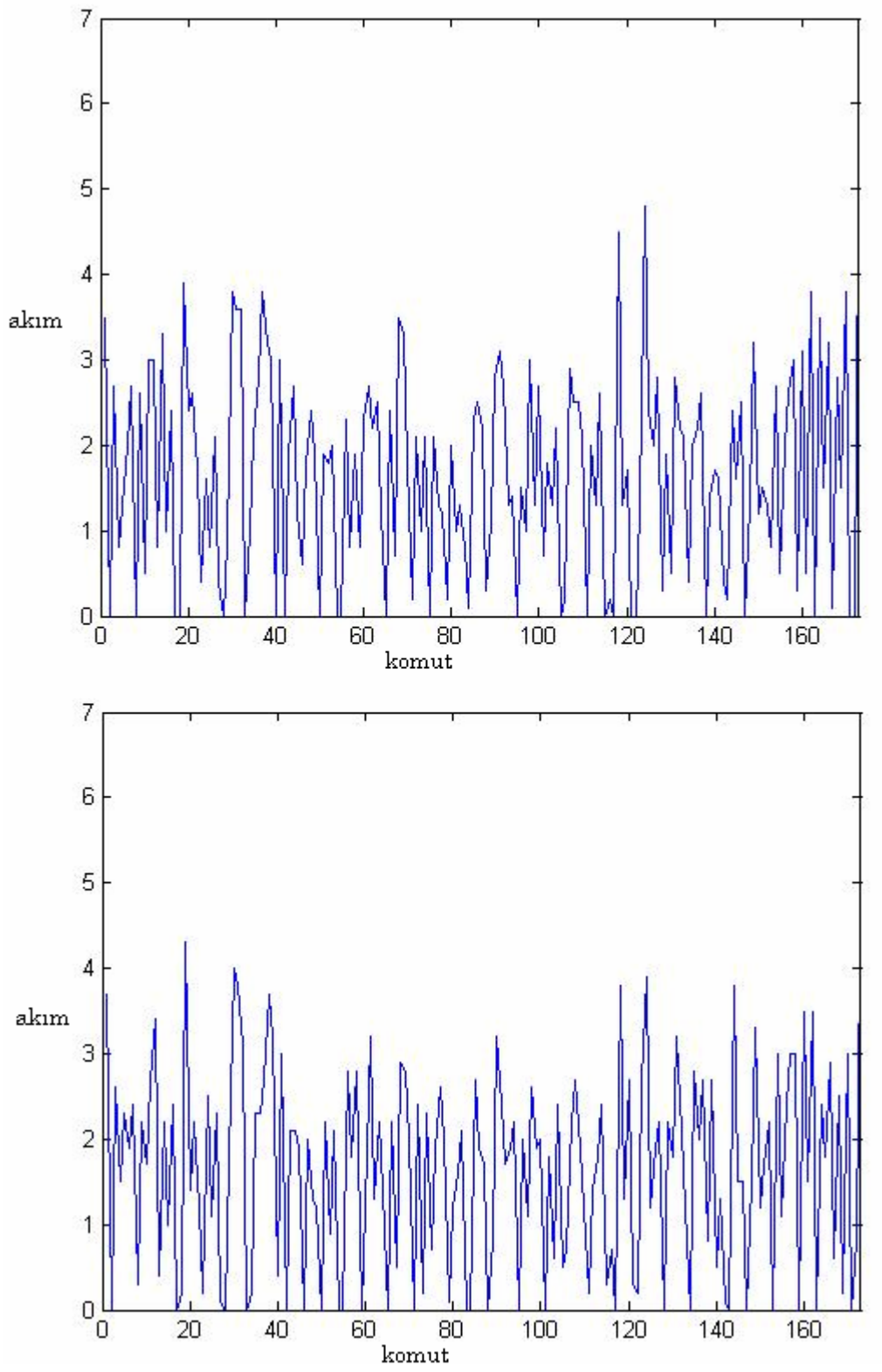
7.1.2 Modelin Testi

Genişletme fonksiyonuna aynı girişler verildiğinde gürültüsüz olarak harcanan gücün zamana göre değişimi Şekil 7.1'de bulunmaktadır.



Şekil 7.1: Aynı iki giriş için genişletme fonksiyonu güç harcaması

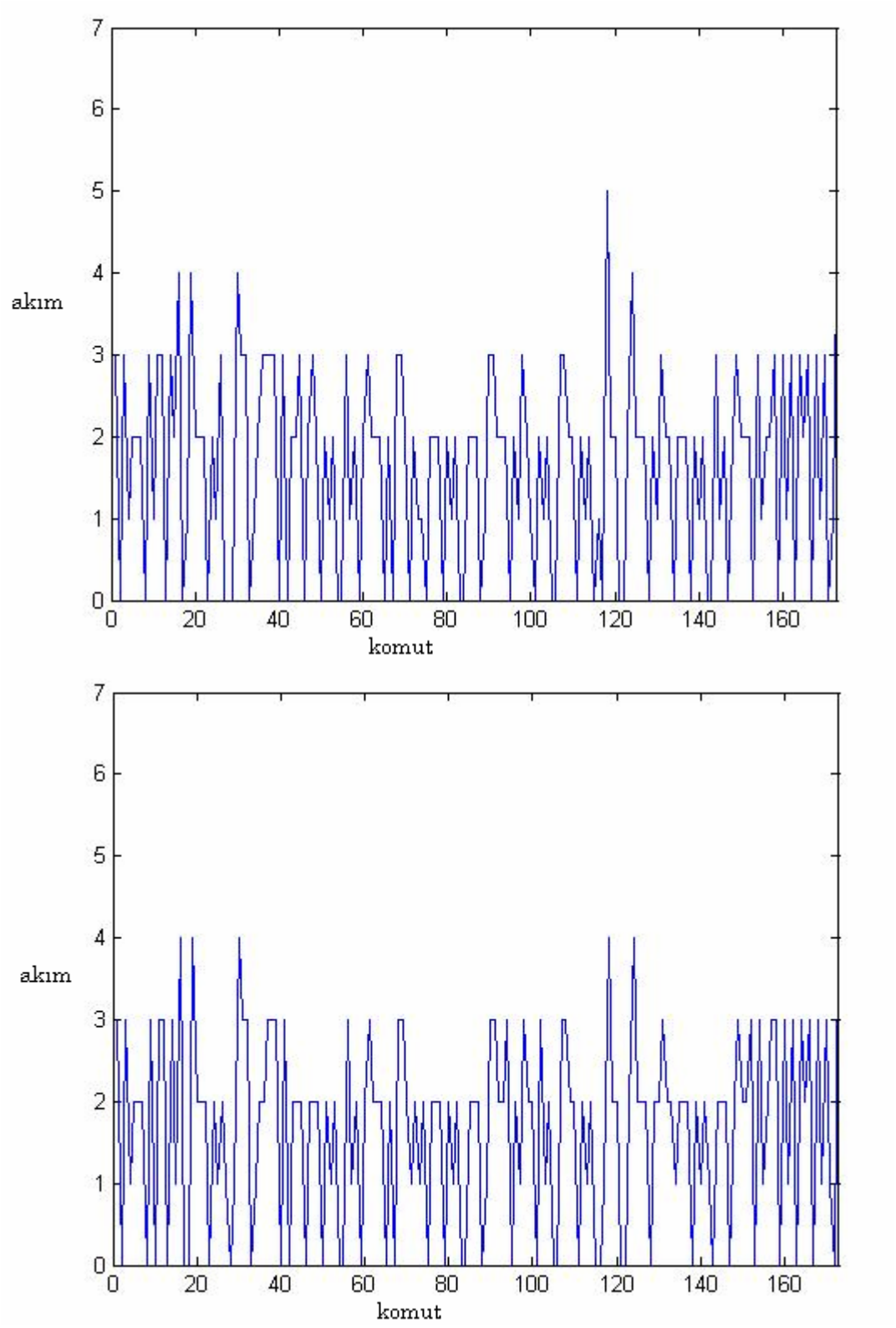
Gürültüsüz durumda ilinti katsayısı 1 olarak hesaplanmıştır. Aynı girişler için 0.8 birimlik gürültü binmesi durumunda harcanan gücün grafiği Şekil 7.2'de bulunmaktadır. Bu durumda ilinti katsayısı 0.870 olmuştur.



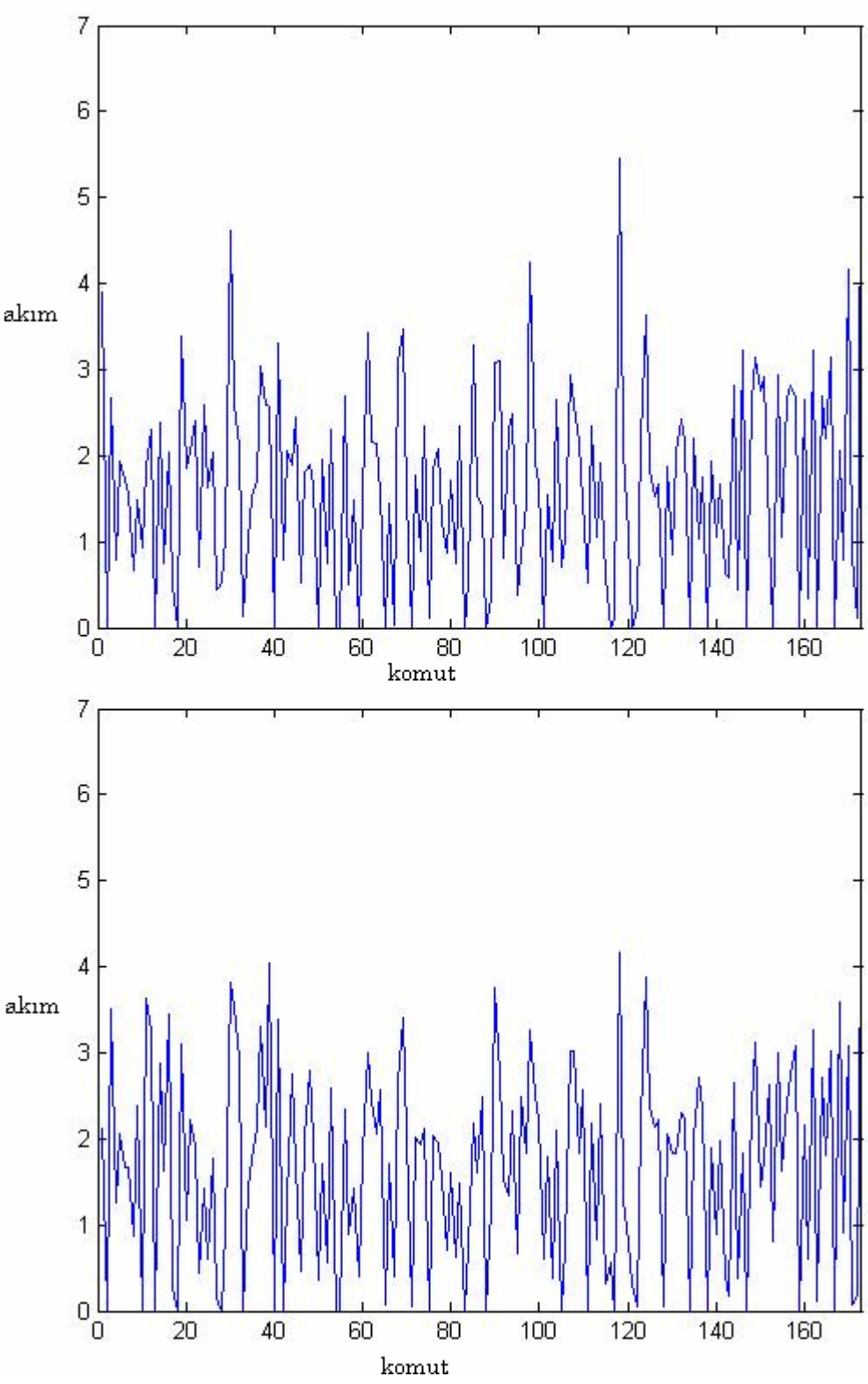
Şekil 7.2 : Aynı iki giriş için 0.8 birimlik üniform dağılımlı gürültülü ortamda genişletme fonksiyonu güç harcaması

Aynı iki giriş için gürültünün rasgele noktalara dağılması durumumdaki güç harcama grafiği Şekil 7.3'te, normal dağılımlı gürültülü durumdaki güç harcama grafiği ise

Şekil 7.4'te bulunmaktadır. Bu gürültülü durumlarda ise ilinti katsayısı sırayla 0.95 ve 0.859 olarak hesaplandı.



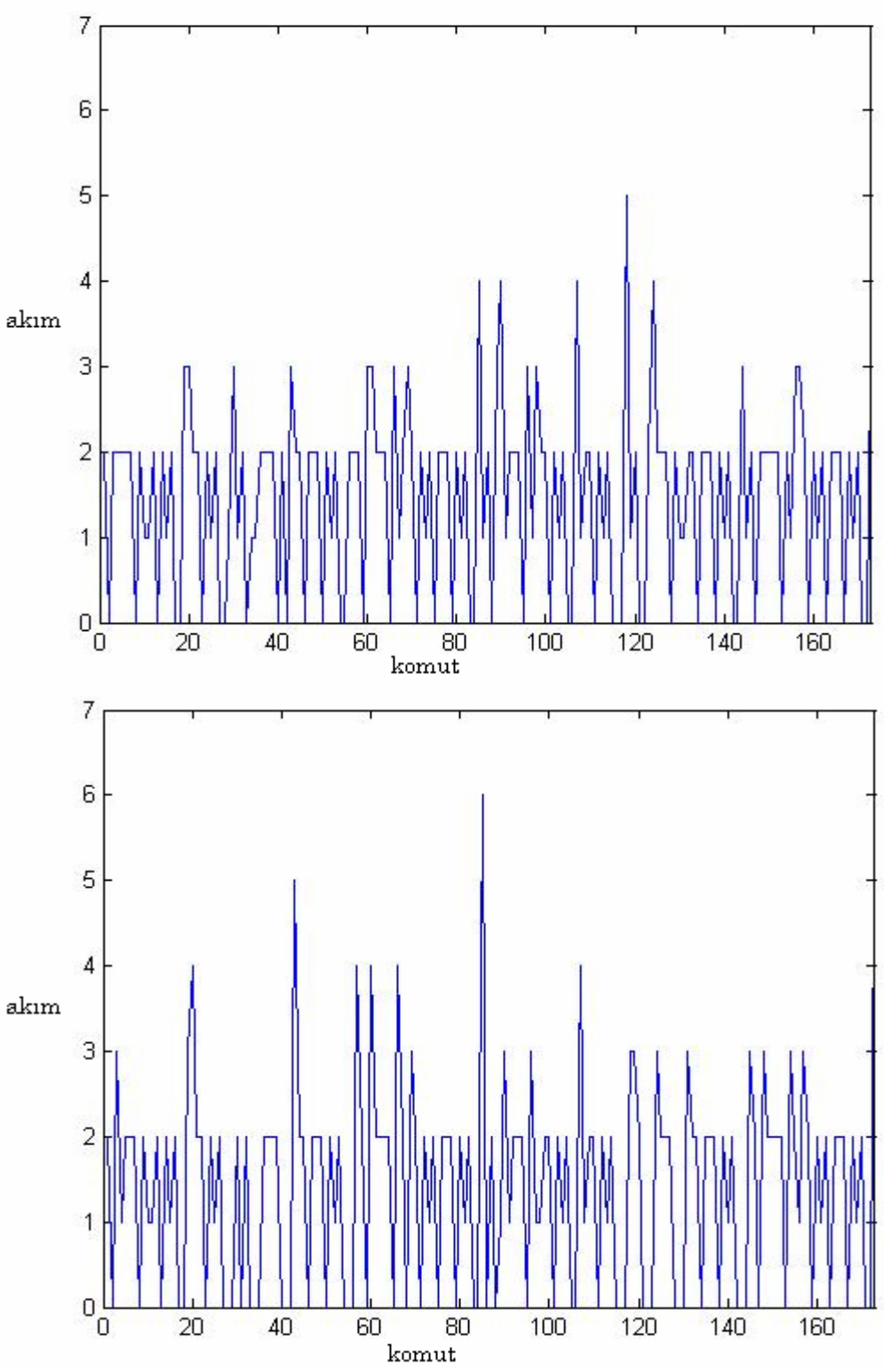
Şekil 7.3 : Aynı iki giriş için gürültünün rasgele noktalara dağılması durumunda genişletme fonksiyonu güç harcaması



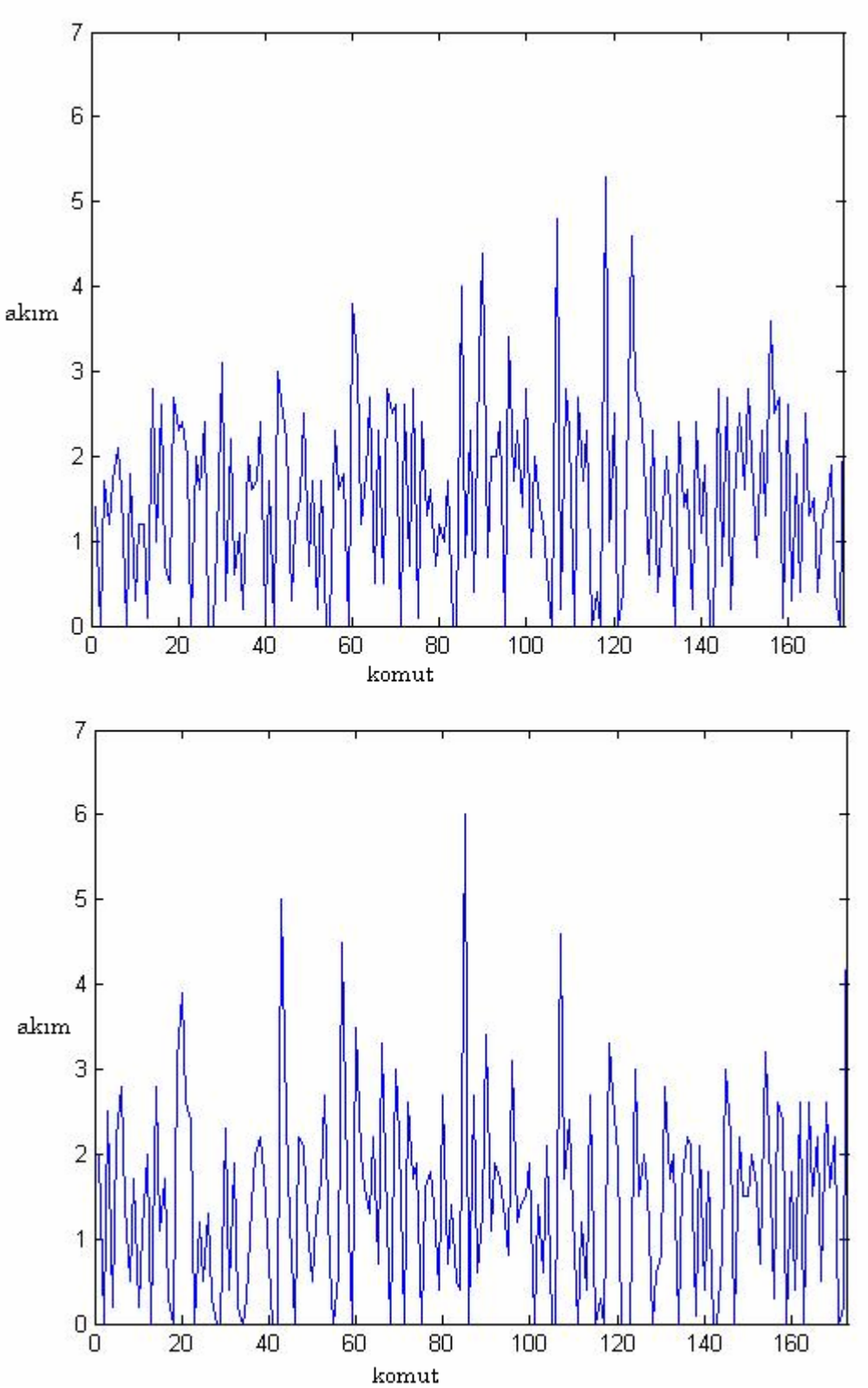
Şekil 7.4 : Aynı iki giriş için normal dağılımlı gürültülü ortamda genişletme fonksiyonu güç harcaması

Farklı giriş çiftleri için simülasyon çalıştırıldı ve farklı güç harcamanın oluştuğu gözlendi. Bir farklı giriş çifti için gürültüsüz ortamda harcanan güç Şekil 7.5'te, 0.8 birimlik üniform dağılımlı gürültülü ortamda harcana güç ise Şekil 7.6'da

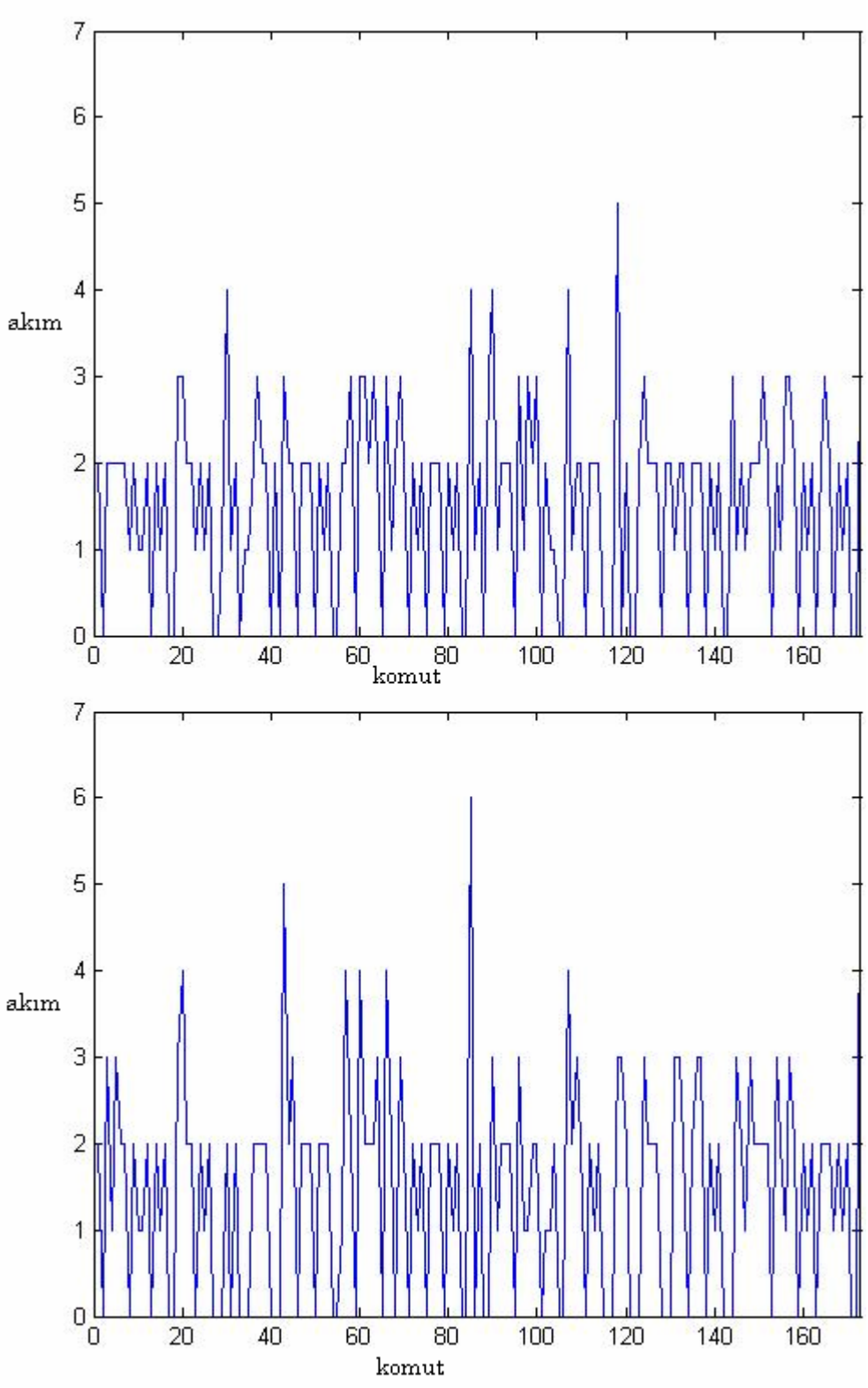
bulunmaktadır. Bu ortamlarda hesaplanan ilinti katsayıları ise sırayla 0.806 ve 0.698'dir. Gürültünün rasgele noktalara dağılması durumunda harcanan güç grafiği Şekil 7.7'de ve normal dağılımlı gürültülü ortamda harcanan güç grafiği de Şekil 7.8'de görülmektedir. Bu durumlarda ilinti katsayıları ise sırayla 0.743 ve 0.685'dir.



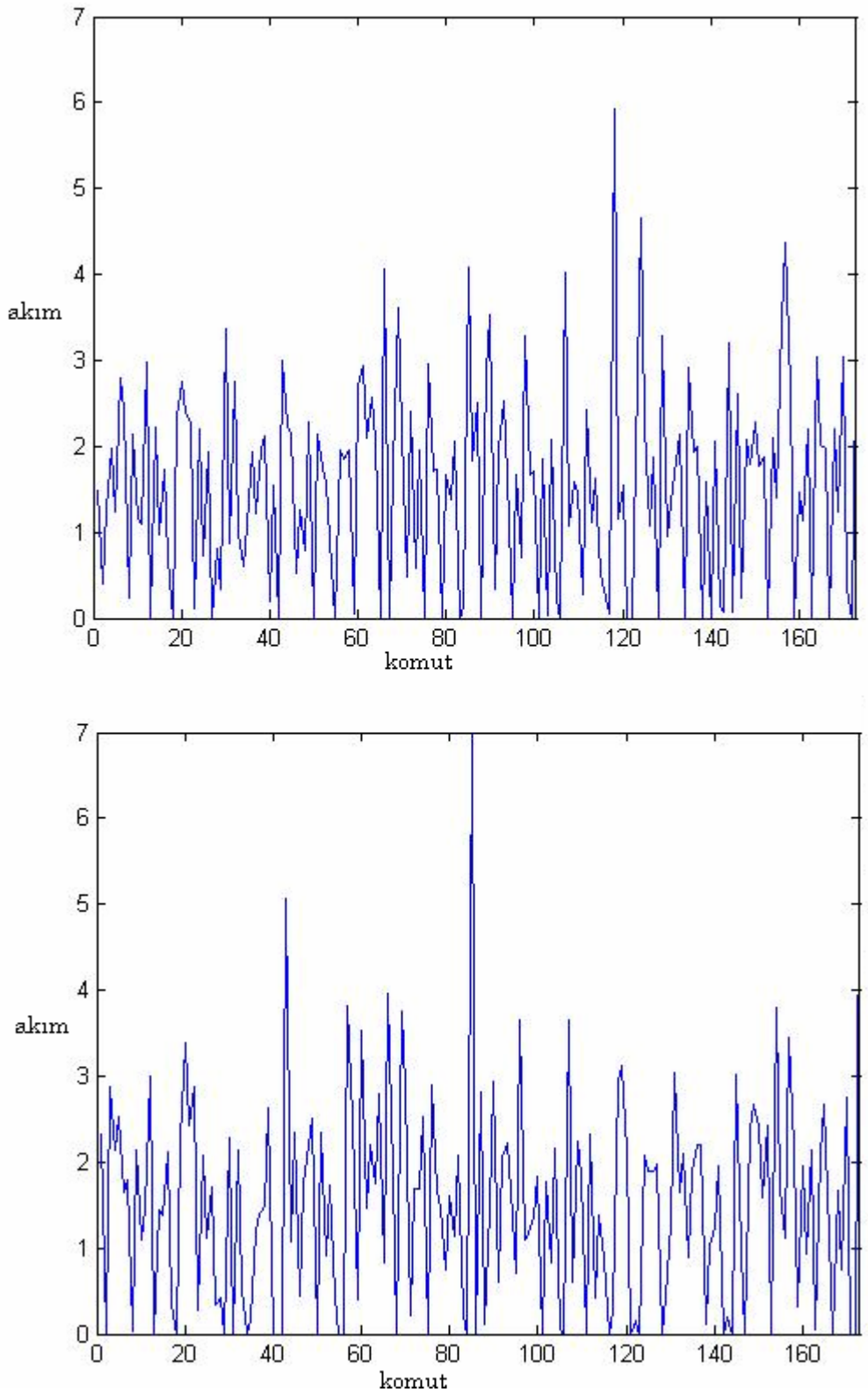
Şekil 7.5: Farkı iki giriş için genişletme fonksiyonu güç harcaması



Şekil 7.6 : Farklı iki giriş için 0.8 birimlik üniform dağılımlı gürültülü ortamda genişletme fonksiyonu güç harcaması



Şekil 7.7 : Farklı iki giriş için gürültünün rasgele noktalara dağılması durumunda genişletme fonksiyonu güç harcaması



Şekil 7.8 : Aynı iki giriş için normal dağılımlı gürültülü ortamda genişletme fonksiyonu güç harcaması

7.2 Saldırının Gerçeklenmesi

Tablo 6.1'deki bir fark alındı ve 1. çevrimdeki s-kutularına bu fark gelecek şekilde iki 2^{13} giriş çifti belirlendi ve şifreleme işlemi yapıldı, çekilen akımlar kaydedildi. Çekilen akımlar karşılaştırılarak çakışmaya neden olan çift belirlendi. Çakışma olduğunda s-kutularının bazı girişlerine gelen bit değerleri belli olduğundan ve açık metinden gelen bitler de belli olduğundan ilgili anahtar bitleri bulundu.

8. SONUÇ

Bu çalışmada, klasik kripto analizin ve yan kanal saldırısının bir arada kullanıldığı çakışma saldırısının literatürdeki bazı algoritmalara uygulaması incelendi. Bu kapsamda, [3]'te anlatılan saldırı detaylı incelendikten sonra aşağıda anlatılan çalışmalar yapıldı.

Literatürde var olan çalışmalar, algoritma bazında çakışma saldırısından bahsetmektedir. Bu çalışmada ise çakışma saldırısının geniş bir tanımı yapıldı ve blok şifreleme algoritmalarına nasıl uygulanabileceği anlatıldı.

Bunun yanı sıra, bu çalışmada, DES algoritmasındaki çevrim fonksiyonu içerisinde bulunan s-kutuların farksal analizi yapıldı ve farklı girişlere karşı aynı çıkışı verme durumları incelendi. Analiz sonucunda, çakışmaya neden olan giriş farkları, bu farklar ile çakışmaya neden olan giriş çiftleri, bu farklar kullanılarak s-kutularının girişindeki hangi bitlerin bulunabileceği ve bu bitlerin değerleri çıkartıldı.

DES algoritmasında bulunan s-kutularının analizinden sonra, DES algoritmasına saldırının nasıl uygulanabileceği açıklandı ve 1. çevrim anahtarını elde etmek için saldırı yapıldı. 1. çevrim anahtarının 37 bitinin, 10 adet fark kullanılarak 81920 açık metin çiftinin algoritmanın bulunduğu kart üzerinde şifrelenmesi ve şifreleme sırasında harcanan gücün analizi ile bulunabileceği gösterildi.

Çakışma saldırısının DES algoritmasının 1. çevrimine uygulanmasından sonra, saldırının 2. çevrime uygulanabilirliği analiz edildi. 1. çevrime saldırı sonucu anahtarın 37 bitinin bulunmasının ardından saldırının 2. çevrime uygulanması ile anahtarın, 7 adet fark kullanılarak 4 bitinin daha bulunabileceği gösterildi.

Çakışma saldırısının DES algoritmasının 1. ve 2. çevrimlerine uygulanmasının ardından, 3DES algoritmasına saldırının nasıl uygulanabileceği analiz edildi. Saldırı ile 3DES algoritmasının, 2^{32} adet metin çiftinin kart üzerinde şifrelenmesi ve 2^{45} adet deneme yanılma ile kırılabildiği gösterildi.

Bu tezde, ayrıca, çakışma saldırısı DES algoritması üzerinde gerçeklendi. Gerçekleme için fiziksel olarak güç harcama ölçümleri yapma yerine simülasyon

kullanıldı. Saldırı gerçeklemesinde kullanılan simülasyon, bu çalışma kapsamında modellendi ve gerçeklendi. Simülasyon ile saldırı gerçekleme için ihtiyacı karşılayacak bir makine kodu yürütücüsü yazıldı. Yürütücü, makine kodunu çalıştırmakla beraber, her kod çalıştırmasında kodda kullanılan mikroişlemci saklayıcılarının ve bellek gözlerinin kod çalıştırılmadan önceki ve kod çalıştırıldıktan sonraki değerlerini kaydetmektedir. Bu değerler kullanılarak da güç harcama değerleri elde edildi ve bu değerler çakışma sezilmesi için kullanıldı.

8.1 Devamında Yapılabilecek Çalışmalar

Tezde çakışma saldırısı deneyinde gerçek güç harcama ölçümü yapılmadı, bunun yerine güç harcama simülasyonu kullanıldı. Saldırı deneyi güç harcama ölçümleri yapılarak gerçeklenebilir.

Güç harcama ölçümlerini yapmak ve analiz etmek kolay gerçekleştirilemediği için çakışma saldırısının yan kanal bilgileri kullanılmadan uygulanabilirliği araştırılabilir.

Çakışma saldırısının başka şifreleme algoritmalarına uygulanması şeklinde de çalışmalar yapılabilir.

KAYNAKLAR

[1] **Coppersmith, D.,** 1994. The Data Encryption Standard (DES) and its strength against attacks, *IBM Journal of Research and Development*., vol. 38, no. 3, pp. 243–250.

[2] **Kocher, P., Jaffe, J., Jun, B.,** 1999. Differential Power Analysis. *In Advanges in Cryptology* – CRYPTO '99. Springer-Verlag.

[3] **Schramm, K., Wollinger, T., and Paar, C.,** 2003. A New Class of Collision Attacks and Its Application to DES. In: Johansson, T. (ed.) FSE 2003. LNCS, vol 2887, pp. 206-222. Springer, Heidelberg.

[4] **NIST,** 2004. Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher. National Institute of Standards and Technology.

[5] **Schramm, K., Leander, G., Fekle, P., and Paar, C.,** 2004. A Collision-Attact on AES: Combining Side Channel- and Differential-Attack. In: Joye, M., Quisquarter, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 163-175. Springer, Heidelberg.

[6] **Stinson, D., R.,** 2002. Cryptography Theory and Practice. CRC Press. New York.

[7] **Menezes, A., J., Oorschot, P., C., Vanstone, S., A.,** 1996. Handbook of Applied Cryptography. CRC Pres, http://www.cacr.math.uwaterloo.ca/hac/

[8] **Sinkov, A.,** 1966. Elementary Cryptanalysis: A Mathematical Approach, Random House, Inc., New York.

[9] **Shannon, C., E.,** 1949. Communication theory and secrecy systems, *Bell Systems Technical Journal*, 28, 656-715.

[10] **Biham, E., Shamir, A.,** 1990. Differential Cryptanalysis of the Data Encryption Standard. Advances in Cryptology – CRYPTO'90, pp. 2-21. Springer-Verlag.

[11] **Acıiçmez, O., Schindler, W., Koç, Ç., K.,** 2007. Cache based remote timing attack on the AES. *Topics in Cryptology, The Cryptographers' Track at the RSA Conference, CT-RSA 2007,* M. Abe, editor, pp. 271-286, Springer, LNCS Nr. 4377, San Francisco, California, February 5-9.

[12] **Kocher, P., Jaffe, J., Jun, B.,** 1998. Introduction to Differential Power Analysis and Related Attacks. Technical Report, Cryptography Research Inc., http://www.cryptography.com/dpa/technical/index.html

[13] **Daemen, J., Rijmen, V., 1998**. AES proposal: Rijndael. In AES Round 1 Technical Evaluation CD-1: Documentation. NIST.

[14] **Davio, M., Desmedt, Y., and Quisquarter, J.,** 1984. Propagation Characteristics of the DES. In Advances in Cryptology – CRYPTO '84, pp. 62-74. Springer-Verlag.

EKLER

EK A.1 : DES 3lü s-kutusu fark tablosu

EK A.1

Tablo A.1 : DES 3lü s-kutusu fark tablosu

	Aktif s-kutuları	S-kutularına giren farklar	Bulunan anahtar bitleri	Bit değerleri
1	345	001011110010101000	13,14,16,26,28,29,30	1,0,0,0,1,0,1
2	345	000111110010101000	13,14,15,26,28,29,30	0,0,0,0,1,0,1
3	812	001011110010101000	3,6,11,43,44,46	1,1,0,0,1,0
4	812	001011110010100100	3,6,11,43,44,46	1,1,1,0,1,0
5	812	001011110010101100	3,6,43,44,46	1,1,0,1,0
6	123	000111110010101100	1,2,3,10,12	0,1,0,1,1
7	123	001011110010101100	1,2,4,10,12	1,0,1,1,1
8	678	001111111010100100	44,47,48,31,32	1,0,0,0,1
9	345	001111110010101000	13,26,28,29,30	0,0,1,0,1
10	678	001111111010101100	44,47,48,31,32	0,1,0,0,1
11	234	000111110010101000	7,9,15,16,18	1,1,0,0,0
12	781	000111110010101100	38,39,46,2	0,0,0,0
13	456	000111110010101000	20,27,28,34	1,1,0,0
14	812	001011111010100100	11,43,44,46	1,0,1,0
15	812	001011111110100100	11,43,44,46	1,0,1,0
16	812	001011111010101000	11,43,44,46	0,0,1,0
17	812	000111110010101000	3,6,11,43	1,1,0,1
18	812	001011111110101000	11,43,44,46	0,0,1,0
19	123	001111110010101100	1,2,10,12	1,0,1,1
20	123	000111110110101100	1,2,3,12	0,1,0,0
21	123	001011110110101100	1,2,4,12	1,0,1,0
22	234	000111110110101000	7,9,15,18	1,1,1,0
23	234	000011110110101000	8,10,15,18	1,1,1,0
24	345	000011110010101000	26,28,29,30	0,1,0,1
25	345	000111110010100100	13,14,15,30	0,0,0,0
26	345	001011110010100100	13,14,16,30	1,0,0,0
27	567	000111111110101000	27,36,25,26	0,1,1,0
28	567	000111111010101000	36,25,26,27	0,1,0,0
29	567	001011110010101000	28,34,36,25	1,1,0,1
30	678	000111111010101100	44,47,48,32	0,1,0,1
31	678	000111111110101100	44,47,48,32	0,1,0,1
32	678	001011111010101100	44,47,48,32	0,1,0,0
33	678	000111111010100100	44,47,48,32	1,0,0,1
34	678	001011111010100100	44,47,48,32	1,0,0,0
35	678	000111111110100100	44,47,48,32	1,0,0,1
36	678	000111110010101000	42,32,39,40	0,1,1,0
37	456	000011110010101000	20,27,28,34	0,1,0,0
38	812	000111110010100100	6,11,43,3	1,1,1,1
39	781	000111110010100100	38,39,46	0,0,0
40	678	000111110110101000	32,39,42	1,0,1
41	781	000111110010101000	38,39,46	0,0,0
42	781	000111111010101000	38,39,48	0,0,0
43	781	000111110110101000	38,39,48	0,0,0
44	781	001111110110101000	37,38,48	0,0,0
45	781	001111111010101000	37,38,48	0,0,0
46	781	000111111010100100	38,39,48	0,0,0
47	781	000111110110100100	38,39,48	0,0,0

48	781	0011111110110100100	37,38,48	0,0,0
49	781	0011111111010100100	37,38,48	0,0,0
50	812	0001111110010101100	3,6,43	1,1,1
51	812	0010111111010101100	43,44,46	0,1,0
52	812	0010111111110101100	43,44,46	0,1,0
53	812	0000111110010100100	6,11,3	1,1,1
54	812	0011111110010100100	6,11,3	1,1,1
55	812	0000111110110100100	3,6,11	0,1,1
56	812	0000111110010101000	3,6,11	1,1,0
57	812	0011111110010101000	3,6,11	1,1,0
58	812	0000111110110101000	3,6,11	0,1,0
59	123	0011111110110101100	1,2,12	1,0,0
60	234	0010111110010101000	15,16,18	0,0,0
61	234	0001111111010101100	7,9,20	1,1,0
62	234	0001111111110101100	7,9,20	1,1,0
63	345	0001111110010101100	13,14,15	0,0,0
64	345	0010111110010101100	13,14,16	1,0,0
65	456	0000111110010100100	20,27,28	0,1,0
66	456	0001111110010100100	20,27,28	1,1,0
67	567	0010111111110101000	28,36,25	1,1,1
68	567	0010111111010101000	28,36,25	1,0,1
69	567	0011111110010101000	34,36,26	1,0,0
70	781	0001111111110101100	38,39,2	0,0,0
71	781	0001111111110100100	38,39	0,0
72	781	0000111110010101100	46,2	0,0
73	781	0001111111110101000	38,39	0,0
74	812	0000111110010101100	6,3	1,1
75	812	0011111110010101100	3,6	1,1
76	812	0000111110110101100	3,6	0,1
77	812	0001111111010100100	11,43	1,1
78	812	0001111111110100100	11,43	1,1
79	812	0001111111010101000	11,43	0,1
80	812	0001111111110101000	11,43	0,1
81	123	0000111110010101100	10,12	1,1
82	234	0001111111110101000	7,9	1,1
83	234	0000111111110101000	8,10	1,1
84	234	0001111111010101000	7,9	1,1
85	234	0010111110110101000	15,18	1,0
86	345	0011111110010100100	13,30	0,0
87	567	0011111111110101000	36,26	1,0
88	567	0011111111010101000	36,26	0,0
89	567	0000111110010101000	34,36	1,0
90	678	0011111111010101000	31,32	0,1
91	781	0000111110010100100	46	0
92	678	0010111111010101000	32	0
93	781	0000111110010101000	46	0
94	781	0000111111010101000	48	0
95	781	0000111110110101000	48	0
96	781	0000111111010100100	48	0
97	781	0000111110110100100	48	0
98	812	0001111111010101100	43	1

99	812	0001111111110101100	43	1
100	812	0000111111010100100	11	1
101	812	0000111111110100100	11	1
102	812	0011111111010100100	11	1
103	812	0011111111110101000	11	0
104	812	0000111111110101000	11	0
105	812	0000111111010101000	11	0
106	812	0011111111010101000	11	0
107	812	0011111111110100100	11	1
108	123	0000111110110101100	12	0
109	123	0000111111010101100	12	0
110	123	0000111111110101100	12	0
111	234	0010111111010101100	20	0
112	234	0010111111110101100	20	0
113	345	0000111110010100100	30	0
114	345	0011111110010101100	13	0
115	567	0000111111110101000	36	1
116	567	0000111111010101000	36	0
117	678	0001111111010101000	32	1
118	678	0001111111110101000	32	1
119	781	0000111111110101100	2	0

ÖZGEÇMİŞ

1983 tarihinde Sinop ilinin Ayancık ilçesinde doğmuştur. Samsun Kocatepe İlkokulunda okuduktan sonra orta okulu ve liseyi Samsun Karşıyaka Lisesinde sürdürmüştür. 2000 senesinde ÖSS sayısal sıralamada 875. olmuş ve İstanbul Teknik Üniversitesi Bilgisayar Mühendisliği Bölümüne yerleştirilmiştir. 2005 Haziran ayında Bilgisayar Mühendisliği Bölümünü ikincilikle bitirmiştir. 2005 Eylül ayında İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsünde Bilgisayar Mühendisliği Bölümünde Yüksek Lisansa başlamıştır. 2005 Temmuz ayında Türkcell İletişim Hizmetleri A.Ş.'de veritabanı üzerinde uygulama geliştirmeci olarak işe başlamış ve 2007 Ekim ayına kadar çalışmaya devam etmiştir. 2007 Ekim ayından beri TÜBİTAK Ulusal Elektronik ve Kriptoloji Araştırma Enstitüsünde araştırmacı olarak çalışmaktadır.