

# Tek Anahtarlı Yeni Bir Şifreleme Algoritması Daha

**Gökhan DALKILIÇ, Gülşah YILDIZOĞLU**

Dokuz Eylül Üniversitesi, Bilgisayar Mühendisliği, İzmir  
dalkilic@cs.deu.edu.tr, gulsah\_yildizoglu@hotmail.com

**Özet:** Simetrik algoritmalar karakter tabanlı ve bit tabanlı sistemler olmak üzere iki bölümde incelenir. Açıklanan şifreleme algoritması bit tabanlı şifreleme algoritmalarını baz alarak gerçekleştirilmiştir. Bu makalede, öncelikle bit tabanlı algoritmalar hakkında bilgi verilmiştir ve bu kategoriye giren algoritmalarından örnekler sunulmuştur. Bu anlatımın takibinde projede kullanılan algoritma hakkında detaylı açıklamaya yer verilmiştir. Simetrik anahtar kullanılan bu algoritmada aynı anahtar ile şifreleme ve şifre çözümü yapılabilmektedir. Algoritma Feistel yapısı üzerine geliştirilmiştir, kullanılan teknikler arasında XOR operatörü ile şifreleme, S-box, E-box ve bitlerle ilgili işlemlerde kullanılan çeşitli operatörler bulunmaktadır. Algoritmanın başarılı çalıştığı örneklerle ispatlanmıştır ve bu örneklerden bir tanesi makalede sunulmuştur.

**Anahtar Kelimeler:** Simetrik şifreleme algoritması, şifreleme algoritması, bit tabanlı şifreleme.

## Yet Another One-Key Encyrption Algorithm

**Abstract:** Symmetric algorithms is examined in two groups character wise systems and bit wise systems. Clarified encryption algorithm is implemented base on bit wise encryption algorithms. In this article, firstly some explanations are given about bit wise algorithms, and some examples are given about the algorithms in this category. Following the algorithm used in project is expounded in details. Via the same key encryption and decryption can be done in this algorithm that uses symmetric key. Algorithm is improved on Feistel structure, some of used techniques are encryption with XOR operator, substitution box, expansion box, and some operators used for bit operations. It is proved that, algorithm works successfully with examples and one of these algorithms is presented in this article.

**Keywords:** Symmetric encryption algorithm, encryption algorithm, bit wise encryption.

## 1. Giriş

Şifreleme bir mesajın gizliliğini sağlamak için kullanılan bir yöntemdir. Şifreleme çeşitlerinden biri olan simetrik şifrelemede ise amaç gönderici ile alıcının ortak bir anahtar üzerinde ve ortak bir şifreleme ile deşifreleme algoritması üzerinde anlaşılıp, mesajı diğer kişilerden korumaktır.

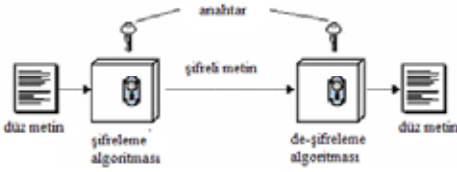
Simetrik şifrelemede beş bileşen bulunmaktadır. Bu bileşenler ve şifreleme işleminin nasıl

yapıldığı Şekil 1’de gösterilmektedir.

Simetrik şifrelemede güvenliği sağlayan anahtardır. Çünkü gizli olan tek şey anahtardır, şifreleme ve deşifreleme algoritmaları herkese açıktır. Farklı anahtarlar sayesinde aynı mesaj ve aynı algoritma ile birbirinden bağımsız şifreli metinler üretilebilir.

Simetrik şifreleme yöntemleri metin üzerindeki işlemlerine göre iki grup altında sınıflandırılabilir. Bunlardan biri karakter tabanlı yani

geleneksel şifreleme sistemleri (Monoalfabetik ve polialfabetik) ve diğeri ise bit tabanlı şifreleme yani modern şifreleme sistemleridir.



Şekil 1. Bir Şifreleme Algoritmasındaki Beş Bileşen[1]

Bit tabanlı şifreleme sistemlerine örnek olarak DES, TWOFISH, IRON ve AES verilebilir.

**a.) DES (Data Encryption Standard) :** Dünyada en yaygın kullanılan şifreleme algoritmalarından birisidir. DES, IBM tarafından geliştirilmiştir. 1975 yılında “Federal Register” tarafından yayınlanmıştır. DES 64 bitlik veriyi 56 bitlik anahtar kullanarak şifreler [2]. Ayrıca klasik Feistel Ağı kullanılarak [3] temelde şifreleme işleminin deşifreleme işlemiyle aynı olması sağlanmıştır. Kullanılan teknikler yayılma ve karıştırmadır. DES’in en büyük dezavantajı anahtar uzunluğunun 56 bit olmasıdır. 1975 yılında yayınlanan bu algoritma günümüzde geliştirilen modern bilgisayarlar tarafından yapılan saldırılar karşısında yetersiz kalmaktadır. Daha güvenli şifreleme ihtiyacından dolayı DES, Triple-DES olarak geliştirilmiştir. Triple-DES algoritması geriye uyumluluğu da desteklemek amacıyla 2 adet 56 bitlik anahtar kullanır. Triple-DES algoritması, DES algoritmasının şifreleme, deşifreleme, şifreleme şeklinde uygulanmasıdır.

**b.) TWOFISH :** 1998 yılında yayınlanan bu algoritma Bruce Schneier - John Kelsey - Doug Whiting - David Wagner - Chris Hall - Niels Ferguson tarafından yaratılmış ve analiz edilmiştir [4]. AES finalistlerinden biridir

ve AES kadar hızlıdır. Aynı DES gibi Feistel yapısını kullanır. DES’den farklarından biri anahtar kullanılarak yaratılan değişken S-box (Substitution box – Değiştirme kutuları)’lara sahip olmasıdır. Ayrıca 128 bitlik düz metni 32 bitlik parçalara ayırarak işlemlerin çoğunu 32 bitlik değerler üzerinde gerçekleştirir. AES’den farklı olarak eklenen 2 adet 1 bitlik rotasyon, şifreleme ve deşifreleme algoritmalarını birbirinden farklı yapmış, bu ise uygulama maliyetini arttırmış, aynı zamanda yazılım uygulamalarını %5 yavaşlatmıştır [5].

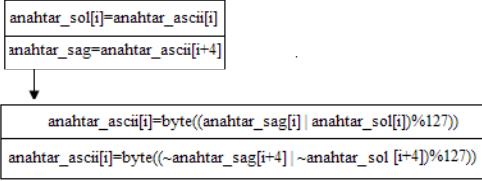
**c.) IRON :** Diğer iki algoritma gibi Feistel yapısını kullanır. IRON, 64 bitlik veri bloklarını 128 bitlik anahtarla şifrelemede kullanılır. Döngü (round) sayısı 16 ile 32 arasındadır. Alt anahtarlar döngü sayısına bağlıdır. Alt anahtarların sayısı döngü sayısına eşittir. Bu nedenden dolayı algoritma anahtar bağımlıdır. IRON algoritmasının var olan algoritmalarından farkı da budur. Bu algoritmanın avantajı bitler yerine 16-tabanındaki sayılar kullanmasıdır, dezavantajı ise yazılım için tasarlanmış olmasıdır [6].

**d.) AES (The Advanced Encryption Standard) :** AES, John Daemen ve Vincent Rijmen tarafından Rijndael adıyla geliştirilmiş ve 2002 yılında standart haline gelmiştir. AES uzunluğu 128 bitte sabit olan blok ile uzunluğu 128, 192 ya da 256 bit olan anahtar kullanır. Kullanılan tekniklerden bazıları baytların yer değiştirmesi,  $4 \times 4$  lük matrisler üzerine yayılmış metin parçalarının satırlarına uygulanan kaydırma işlemleridir. 2006 yılı itibariyle en popüler simetrik algoritmalarından biridir [7].

Makalenin devamında projede kullanılan algoritmadan, bu algoritmanın avantajlarından ve kısıtlamalarından söz edilmektedir. Ek olarak algoritmanın geliştirilmesi için neler yapılabilir konusu ele alınmıştır.

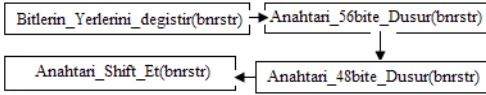


*cii* dizisindeki her sayı tek tek bitlerine ayrılır. 1 sayıdan 8 tane bit elde edilir ve bu bitler bir dizide toplanır. Toplam 64 tane bit elde edilir. Artık 8 elemanlı dizi, 64 elemanlı başka bir bayt dizisine dönüştürülmüş oldu. Bu işlem yer kaybına neden olur ama bitler üzerinde işlem kolaylığı sağlaması açısından yararlıdır.



Şekil 4. Anahtarlar\_uretiliyor fonksiyonu

Bu fonksiyonda Şekil 5’te belirtilen 4 tane alt fonksiyon çağrılmaktadır:



Şekil 5. Alt Fonksiyonlar

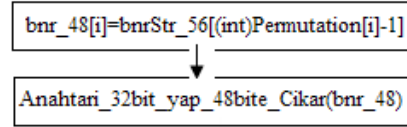
Bu fonksiyonlardan bir tanesi anahtara ait olan 64 bitin *bnrstr* dizisindeki yerini karıştırmak için kullanılmaktadır. Fonksiyonun çalışması modüler aritmetik üzerine kuruludur. Eğer indekxin modu 2’ye göre 0’a eşitse bu indeksteki değer *cift\_indekteki\_bitler\_sol\_tarafa* dizisine, 1’se *tek\_indekteki\_bitler\_sag\_tarafa* dizisine atanır. Daha sonra *cift\_indekteki\_bitler\_sol\_tarafa* dizisindeki elemanlar ile *tek\_indekteki\_bitler\_sag\_tarafa* dizisindeki elemanlar sırayla OR operatörü yardımıyla bit tabanlı bir işleme tabi tutulur ve her işlemin sonucunda oluşan yeni değer *bnrstr* dizisine atanır. 32 defa devam eden bu atamadan sonra, *tek\_indekteki\_bitler\_sag\_tarafa* dizisindeki elemanların değili ile *cift\_indekteki\_bitler\_sol\_tarafa* dizisindeki elemanların değilleri sırasıyla OR operatörü ile işlenir ve bu 32 değer *bnrstr* dizisinin kalan 32 elemanlık yerini doldurur.

*Anahtari\_56ya\_Dusur* fonksiyonunda 64 bitlik anahtar uzunluğu *permutation choice* tekniği ile 56 bite düşürülür. Şekil 6’daki kod 56 kez bir döngü içinde tekrarlandığında hedefe ulaşılır.

```
bnr_56[i] = gelen_bnrStr[(int)Permutation[i] - 1]
```

Şekil 6. Anahtari\_56ya\_Dusur

Bu fonksiyondan sonra Şekil 7’de belirtilen *Anahtari\_48e\_Dusur* fonksiyonu 56 bite düşen anahtarı 48 bite azaltmak için kullanılır. Fakat bu fonksiyon içinde başka bir fonksiyon daha çağrılır. Çağrılan fonksiyonun amacı anahtardaki karmaşıklığı arttırıp, olabilecek ataklara karşı güvenliği sağlamaktır.



Şekil 7. Anahtari\_48e\_Dusur

*Anahtari\_32\_yap\_48e\_Cikar* fonksiyonu anahtarı 48 bitten 32 bite azaltmak için S-box adı verilen yapıyı kullanır. Bu yapı sayesinde rastgele üretilmiş sayılardan oluşan bir tablodan geçen her 6 bit 4 bit olarak çıkar (Bu algorithmada ise 6 dizi elemanı geliyor ve bu 6 taneden herhangi 4’ü olarak çıkıyor.). Toplam 8 adet S-box kullanılmıştır. Bu fonksiyonun sonunda ise “expansion box” tekniği ile 32 bitlik anahtar 48 bite arttırılır.

“Expansion Box” Tekniği : Bazı bitler rastgele olarak tekrarlanır.

*Anahtari\_48e\_Dusur* fonksiyonunu *Anahtari\_Shift\_et* fonksiyonu takip etmektedir. Bu fonksiyonda 16 adet alt anahtar üretilmektedir. Bir sonraki anahtar bir önceki anahtarın bitlerinin 1 kez sola kaydırılmasıyla üretilmiştir. Bu nedenle her anahtar bir öncekine bağlıdır. Bir bit değiştirildiğinde bir çok anahtar üretilir.

mektedir. Terminolojide bu olaya avalans etkisi denilmektedir.

### SIFRELEME\_FONKSİYONU Fonksiyonu :

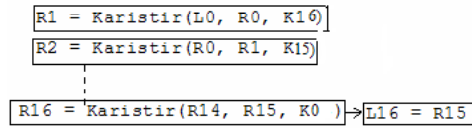
Bu fonksiyonda (Şekil 7) öncelikle kullanıcıdan alınan düz metnin ilk 8 karakteri alınır ve *Karakterleri\_karistir\_text* fonksiyonunda her karakter ASCII değerine çevrilir ve bir dizide saklanır. Daha sonra bu dizinin elemanları anahtar dizisinin ASCII değerlerini toplayarak elemanlarını kaydırma algoritmasının nerdeyse aynısı olan bir algoritma ile yer değiştirir.

Daha sonra dizideki her eleman bitlerine ayrılır ve toplam 64 bit elde edilip bir dizi içinde saklanır. Bu 64 elemanlı dizi sol ve sağ blok olmak üzere iki parçaya ayrılır, ve artık sahnenin sahibi Feistel yapısıdır.

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \text{ xor } F(R_{i-1}, K_i)$$

16 kez döngü uygulanır ve bu 16 döngü sonunda elde edilen iki dizinin Feistel yapısındaki çaprazlaması sonucu şifreli metnin ilk bloğu oluşur. Şekil 8'de bu işlem akış diyagramı ile anlatılmıştır:



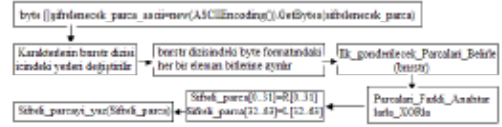
Şekil 8. Feistel yapısı

R16 ve L16 şifreli metni oluşturan parçalardır. Feistel yapısı kullanıldığı için şifrenin çözümü anahtarı bilen kişi olan alıcı için çok kolaydır, anahtarı bilmeyen kişi olan saldırgan (opponent) için ise oldukça zordur.

### 2.2 ) Deşifreleme Algoritması

Şifreyi çözmek için öncelikle şifreli metnin ilk 64 bitlik bloğundan başlanır. Bu bloğun şifresi

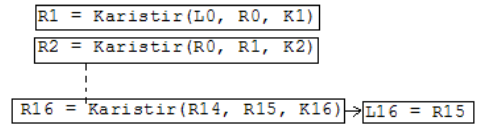
çözüldükten sonra eğer varsa diğer 64 bitlik blok deşifre edilir. Bu işlem şifreli metnin sonuna kadar devam eder.



Şekil 7. Şifreleme fonksiyonu

Şifreyi çözmeye kullanılan fonksiyonda şifreli metin bloğu bitlerine ayrılır. Bu bitlerden ilk 32 si L0 parçasını, geri kalan 32 bit ise R0 parçasını oluşturur. Şifreleme algoritmasında anahtarları üretmek için kullanılan fonksiyonun aynısı deşifrelemede anahtarları tekrar üretmek için kullanılır. Aynı anahtardan 16 tane alt anahtar üretilir. Feistel yapısı kullanıldığından şifreyi çözmek için anahtarları fonksiyona tersten veririz (16. anahtardan başlanıp 1. anahtara doğru).

Şekil 9'daki akış diyagramı şifrelemedekine çok benzer, tek farklılık anahtarların tersten verilmesidir:



Şekil 9. Deşifreleme

En son aşamaya gelindiğinde yapılması gereken, şifrelerken karıştırılan dizideki karakterleri eski yerlerine geri koymaktır. Çünkü karakterleri karıştırılan dizi alt anahtarlarla şifreleme işlemine tabi tutulmuştur. Deşifrelemede tersten gidileceği için elimizdeki dizinin karakterlerindeki karışıklığı düzeltmemiz gerekmektedir.

Bu işlemde yapılması gereken toplama işleminin tersi olan çıkarma işlemini kullanmaktır (1). Detaylı açıklamak gerekirse; şifrelerken örneğin 3. indekste bulunan elemanın indek-

sine 2 ekleyerek 5. indekste yer alması sağlanır, deşifre ederken de 5. indeksteki elemanın indeksinden 2 çıkartıldığında 3. indeks yani olması gereken indeks bulunur. Bunun için kullanılan kod satırı aşağıda verilmiştir:

```
degisen_karistirilan_karakter_dizisi[i]=karistirilan_karakter_dizisi[(i-shift_etme_miktari+64) % 64] (1)
```

Elimizde artık düz metne ait olan bitler bulunmaktadır. Bu bitleri bir araya getirip topladığımızda düz metne ait karakterlerin ASCII karşılıklarını buluruz. Artık bu aşamadan sonra blokların birleştirilmesiyle düz metne ulaşılmış olunur.

### 3. Algoritmanın Avantajları

Avalans etkisi ve anahtar uzunluğu bu algoritmanın güçlü parçalarıdır. Çünkü anahtarı bulmak için yapılan saldırılarda anahtarın bir bitindeki değişiklik bir değil birden fazla farklı anahtarların oluşmasına neden olur. Anahtar uzunluğundan dolayı *Kaba Kuvvet (Brute Force)* saldırısıyla anahtarın bulunması çok zordur.

### 4. Algoritmadaki Kısıtlamalar

Algoritmadaki ilk kısıtlama anahtar uzunluğunun sabit olmasıdır. Değişen uzunlukta anahtar kullanılamamaktadır. Diğer kısıtlama ise döngü sayısının sabit olmasıdır. Eğer döngü sayısı anahtar uzunluğuna bağlı olsaydı ve anahtar uzunluğu da değişebiliyor olsaydı, algoritma saldırılara karşı daha dirençli olurdu. Ayrıca donanım için tasarlanmadığından, donanıma uyum göstermeyebilir. Çalışma hızı, modüler aritmetik ve bit kaydırma işlemlerinden dolayı yavaşlamaktadır.

## 5. Örnek Uygulama

Düz Metin: Kriptoloji şifreleme ve deşifreleme işlemlerini kapsayan bir bilimdir ve diğer bilimlerde olduğu gibi bu biliminde bir tarihi vardır.

Anahtar: Simetrik şifreleme bir mesajın gizliliğini sağlamak için kullanılan bir şifreleme türüdür.

Şifrelenmiş-Metin: *25F221670F1-FEC9C5DD9DD1B0E16EF0389DB391A524-AF1C9D019568830E98A6C7B1C069A62BE11-AD810A01AEC139168AC82AA81F468A1108D-D22099E8650E7DB5C5CDADACF96FC46F-9131DC7D518268ED21F008536E5CB76D91A-A31AE6BE3728F1472C82DD582682DB484A5-E09C4BF85DD0213B014506FD52CDA8CBF33-CAB039940955DA74EDD57B*

De-şifre Edilmiş Metin: *Kriptoloji şifreleme ve de-sifreleme işlemlerini kapsayan bir bilimdir ve diğer bilimlerde olduğu gibi bu biliminde bir tarihi vardır.*

(ASCII kodlamadaki kısıtlamalardan dolayı düz metin *hexadecimal* formda gösterilmiştir).

## 6. Sonuç

Algoritma daha önceden var olan şifreleme algoritmalarındaki tekniklerin üzerine yeni teknikler ekleyerek ve bu tekniklerin uygulama yerlerinin değiştirilmesiyle oluşturulmuştur. Algoritmanın başarılı bir şekilde çalıştığı örneklerle de test edilmiş, performans üzerindeki kısıtlamalarının ne olduğu açıkça dile getirilmiştir. Algoritma donanım için geliştirilmeye açıktır ve daha güvenilir bir sistem için yenilikleri içine alabilecek bir yapı üzerine kurulmuştur.

Algoritmayı geliştirmek için algoritmasındaki gibi anahtar uzunluğuyla döngü sayısının değişken olması sağlanabilir. Döngü sayısı anahtar uzunluğuna bağlı olduğu sürece anahtar uzunluğunu bulmak güç olduğundan döngü sayısının da bulunması güçleşecektir.

## **7. Kaynaklar**

[1]. Stallings W., *Cryptography and Network Security: Principles and Practice Fourth Edition*, Prentice Hall, New Jersey, 2006, ss.30.

[2]. Stinson D. R., *Cryptography, Theory and Practice*, CRC Press, 1995, ss. 70.

[3]. Stallings W., *Network Security Essentials: Applications and Standards, Second Edition*, Prentice Hall, New Jersey, 2003, ss. 33

[4]. Twofish, Bruce Schneier kişisel websitesi, <http://www.schneier.com/twofish.html>.

[5]. Ferguson N., Schneier B., “Practical Cryptography”, Wiley Publishing, 2003, ss. 59-61

[6]. Demir N., Dalkılıç G., “Anahtar Bağımlı Bir Şifreleme Algoritması (IRON)”, *Akademik Bilişim 2007*, 31 Ocak – 2 Şubat 2007, Kastamonu.

[7]. Wikipedia, The Free Encyclopedia, [http://en.wikipedia.org/wiki/Advanced\\_Encryption\\_Standard](http://en.wikipedia.org/wiki/Advanced_Encryption_Standard).