**ANKARA ÜNİVERSİTESİ**
**FEN BİLİMLERİ ENSTİTÜSÜ**

**YÜKSEK LİSANS TEZİ**

# VERİ İLETİŞİMİNDE AĞ DİNLEME SALDIRILARINA KARŞI GELİŞTİRİLMİŞ MELEZ KRİPTOSİSTEM

**Ertan ERSAN**

**BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI**

**ANKARA**
**2017**

## ÖZET

Yüksek Lisans Tezi

VERİ İLETİŞİMİNDE AĞ DİNLEME SALDIRILARINA KARŞI GELİŞTİRİLMİŞ MELEZ KRİPTOSİSTEM

Ertan ERSAN

Ankara Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Yrd. Doç. Dr. Bülent TUĞRUL

Bu çalışmada; kriptoloji ile ilgili genel bilgilere, kriptolojinin tarihsel gelişimine ve kullanım alanlarına, bilgi güvenliği servislerine, elektronik tehditlere, gizli anahtarlı ve açık anahtarlı kriptosistemlere, Diffie-Hellman ve Eliptik Eğri Diffie-Hellman anahtar değişim sistemlerine, özetleme fonksiyonlarına ve elektronik imzaya değinilmiş, ilgili kavramlar detaylıca incelenmiştir.

Mobil cihazlar gibi işlem kapasitesi daha düşük olan ortamlarda uygulamaların performansı önemli bir konudur. Güvenlik önlemlerinin getirdiği yükten dolayı yaşanabilecek performans düşüşüne karşı etkili çözüm üretmek çok önemlidir. Önerilen kriptosistemde; şifreleme ve deşifreleme işlemlerindeki yüksek performansı sebebiyle AES tercih edilmektedir. Ancak AES'te gizli anahtarın taraflar arasında dağıtımı karşımıza sorun olarak çıkmaktadır. Yapılan çalışmada; bu sorun, gizli anahtarı tarafların ortak bir şekilde oluşturabileceği Diffie-Hellman anahtar değişim sistemi ile aşılmıştır. Mobil ortamlarda da kullanılabilecek bu kriptosistemde, Diffie-Hellman anahtar değişim sistemi kullanılırken aynı güvenlik seviyesini daha küçük anahtar boyutlarıyla sağlayan Eliptik Eğri Diffie-Hellman yönteminden faydalanılmıştır. Bununla birlikte, önerilen kriptosisteme elektronik imza entegrasyonu yapılarak bilgi güvenliği servislerinden biri olan gizliliğin yanı sıra; bütünlük, inkâr edememe ve kimlik doğrulama güvenlik servislerini de kapsaması sağlanmıştır.

Sonuç olarak; veri iletişiminde kullanılan ağ, saldırganlar tarafından dinlense bile; verinin şifreli olarak iletilmesini sağlayacak, kötü niyetli girişimleri boşa çıkaracak, yetkisi olmayan kişilerin bilgiye erişmesini engelleyecek ve taraflar arasında güvenli haberleşme ortamı sağlayacak bir melez kriptosistem önerilmiş ve mobil uygulama geliştirilmiştir.



**Mart 2017, 64 sayfa**

**Anahtar Kelimeler:** Kriptoloji, Bilgi Güvenliği Servisleri, Elektronik Tehditler, Ağ Dinleme Saldırıları, Gizli Anahtarlı Kriptografi, Açık Anahtarlı Kriptografi, Diffie–Hellman Anahtar Değişimi, Eliptik Eğri Diffie-Hellman Anahtar Değişimi, Özetleme Fonksiyonları, Elektronik İmza.

## ABSTRACT

Master Thesis

DEVELOPMENT OF A HYBRID CRYPTOSYSTEM AGAINST NETWORK SNIFFING ATTACKS IN DATA COMMUNICATIONS

Ertan ERSAN

Ankara University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Asst. Prof. Dr. Bülent TUĞRUL

In this thesis, general information on cryptology, its historical development and usage areas, information security services, electronic threats, private key and public key cryptosystems, Diffie-Hellman Key Exchange, Elliptic Curve Diffie-Hellman Key Exchange, hash functions and electronic signature are studied, all related concepts are analysed thoroughly.

Application performance is an important issue for environments with lower processing capacities such as mobile devices. It is very crucial to create an effective solution against the performance weakness that may be experienced due to the burden caused by security measures. In the proposed cryptosystem, AES is preferred due to its high performance for encryption and decryption processes. However, distribution of the secret key between parties in AES arises as a problem. In this study, this problem is overcome by using Diffie-Hellman Key Exchange system which enables both parties to create a secret key together. In this cryptosystem which may also be used in mobile environments, Elliptic Curve Diffie-Hellman Key Exchange technique, which provides the same security level with smaller key sizes, is used to make the proposed cryptosystem more efficient for those mobile environments. Besides, with the integration of electronic signature into the proposed cryptosystem; integrity, non-repudiation, and authentication security services together with confidentiality which is one of the information security services are achieved.

Thus, in this work, a hybrid cryptosystem is proposed which makes sure encrypted transmission of data, fails malicious attempts, prevents unauthorized access to information, and provides an efficient way of secure communication between parties even if the data communications network is sniffed by attackers. Moreover, a mobile application is developed which incorporates this method.



**March 2017, 64 pages**

**Key Words:** Cryptology, Information Security Services, Electronic Threats, Network Sniffing Attacks, Private Key Cryptography, Public Key Cryptography, Diffie–Hellman Key Exchange, Elliptic Curve Diffie-Hellman Key Exchange, Hash Functions, Electronic Signature.

## TEŞEKKÜR

Bu tez çalışmasının planlanmasında, araştırılmasında, yürütülmesinde ve oluşturulmasında ilgi ve desteğini esirgemeyen, engin bilgi ve tecrübelerinden yararlandığım, yönlendirme ve bilgilendirmeleriyle çalışmamı bilimsel temeller ışığında şekillendiren ve akademik ortamda olduğu kadar insani ilişkilerde de fikirleriyle yetişmeme ve gelişmeme katkıda bulunan danışman hocam Sayın Yrd. Doç. Dr. Bülent TUĞRUL'a, (Ankara Üniversitesi Bilgisayar Mühendisliği Anabilim Dalı)

Yüksek Lisans eğitimim boyunca çok değerli bilgilerini ve tecrübelerini aktaran, yardımlarını esirgemeyen Ankara Üniversitesi Bilgisayar Mühendisliği Anabilim Dalı hocalarına,

Her daim manevi desteklerini benden esirgemeyen çalışma arkadaşlarıma,

Bugüne ulaşmamı sağlayan, hayatımın her anında yanımda olan, maddi ve manevi desteklerini benden hiçbir zaman esirgemeyen aileme,

Tüm kalbimle teşekkür ederim.

Ertan ERSAN
Ankara, Mart 2017

# İÇİNDEKİLER

## SİMGELER DİZİNİ

| | |
|---|---|
| C | Şifreli Metin |
| D() | Deşifreleme Fonksiyonu |
| E() | Şifreleme Fonksiyonu |
| K | Anahtar |
| P | Açık Metin |
| $\oplus$ | XOR |
| \|\| | Birleştirme |

**Kısaltmalar**

| | |
|---|---|
| 3DES | Üçlü Veri Şifreleme Standardı (Triple Data Encryption Standard) |
| AES | Gelişmiş Şifreleme Standardı (Advanced Encryption Standard) |
| CBC | Şifre Blok Zincirleme (Cipher Block Chaining) |
| CFB | Şifre Geri Besleme (Cipher Feed Back) |
| DES | Veri Şifreleme Standardı (Data Encryption Standard) |
| DH | Diffie–Hellman Anahtar Değişimi |
| DSA | Dijital İmza Algoritması (Digital Signature Algorithm) |
| ECB | Elektronik Kod Kitabı (Electronic Code Book) |
| EEDH | Eliptik Eğri Diffie-Hellman Anahtar Değişimi |
| FIPS | Federal Bilgi İşlem Standardı (Federal Information Processing Standard) |
| IDEA | Uluslararası Veri Şifreleme Algoritması (International Data Encryption Algorithm) |
| IV | Başlangıç Vektörü (Initialization Vector) |
| MD4 | Mesaj Özetleme Algoritması 4 (Message Digest Algorithm 4) |
| MD5 | Mesaj Özetleme Algoritması 5 (Message Digest Algorithm 5) |

| | |
|---|---|
| NIST | ABD Ulusal Teknoloji ve Standartları Enstitüsü (National Institute of Standards and Technology) |
| NSA | ABD Ulusal Güvenlik Kurumu (National Security Agency) |
| RSA | Rivest – Shamir – Addleman Algoritması (Rivest – Shamir – Addleman Algorithm) |
| OFB | Çıktı Geri Besleme (Output Feed Back) |
| SHA | Güvenli Hash Algoritması (Secure Hash Algorithm) |

# ŞEKİLLER DİZİNİ

# ÇİZELGELER DİZİNİ

# 1. GİRİŞ

Kriptoloji, Yunanca ‘Cryptos’ (gizli) ve ‘Logos’ (kelime) kelimelerinin birleştirilmesinden oluşturulmuştur ve iletişimde gizlilik bilimi olarak değerlendirilmektedir.

Devlet işlerinde, ticari ilişkilerde, askeri işlerde güvenli ve hızlı iletişim önemli bir konudur. Sistemler arası bağlantılarda ya da herhangi iki nokta arasındaki haberleşmede verinin güvenli bir şekilde gittiğinden emin olmak gerekir. İletişimde, açık bir haberleşme kanalı kullanılıyorsa gizli tutulmak istenen bilginin yetkisiz bir kişi tarafından dinlenebileceği veya haberleşme kanalına girip veriyi bozabileceği ya da değiştirebileceği düşüncesi her zaman için önemli bir problem oluşturur. Bu problemin çözümü ise gönderilen verinin şifrelenmesi ile olur. Böylece açık haberleşme kanalları kullanılarak verinin güvenli bir şekilde ulaştırılması sağlanır.

Şifreleme, askeri ve diplomatik haberleşmelerde veri güvenliğini sağlamak için uzun yıllardır kullanılmaktadır ancak günümüzde internet teknolojisinin gelişmesi ile hayatımızın her alanına girmiştir. Sağlık hizmetleri, bankacılık işlemleri, mobil uygulamalar gibi alanlarda yapılan işlemlerin güvenli ve hızlı bir şekilde yapılabilmesi için şifrelemeye olan ihtiyaç artmıştır.

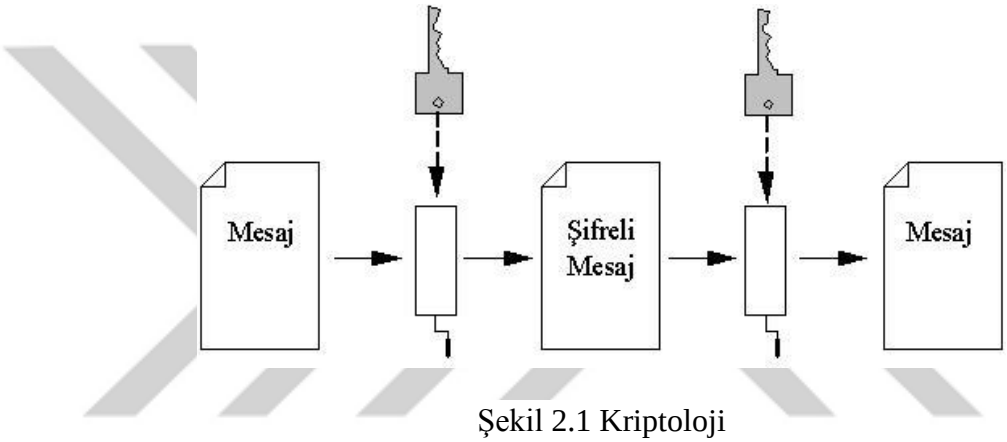
Bu çalışmada, Bölüm 2’de Kriptoloji Bilimi detaylıca incelenmiş, geçmişten günümüze kullanılan yöntemlerden bahsedilmiştir. Bolüm 3’te konu ile ilgili yapılan çalışmalar incelenip önerilen yöntemlerden bahsedilmiştir. Bunun yanı sıra yapılan çalışmaların zayıf yönleri tespit edilerek bu zayıflıklara karşı taraflar arasındaki haberleşmeyi en hızlı ve en güvenilir şekilde gerçekleştirmeye yönelik bir yöntem önerilmiştir. Bölüm 4’te önerilen yönteme yönelik bir mobil uygulama geliştirilmiş ve elde edilen bulgular değerlendirilmiştir. Bölüm 5’te ise sonuç ve önerilere yer verilmiştir.

# 2. KURAMSAL TEMELLER

## 2.1 Kriptoloji Genel Bilgi

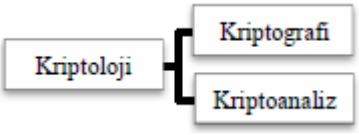
### 2.1.1 Kriptoloji

Çeşitli iletilerin, yazıların belli bir sisteme göre şifrelenmesi, bu verilerin alıcıya iletilmesi ve iletilmiş verilerin deşifresiyle uğraşan şifre bilimidir.




Şekil 2.1 Kriptoloji

Kriptoloji iki alt dala ayrılır.




Şekil 2.2 Kriptoloji bilimi alt dalları

#### 2.1.1.1 Kriptografi

Kriptografi, Yunan dilinde ‘Cryptos’ (gizli) ve ‘Graphos’ (yazı) kelimelerinden türetilmiş ve dilimizde şifreleme olarak adlandırılmıştır.

Kriptografi, okunabilir durumdaki bir bilginin istenmeyen taraflarca okunamayacak bir hale dönüştürülmesinde kullanılan tekniklerin tümü olarak da ifade edilebilir.

#### 2.1.1.2 Kriptoanaliz

Kriptoanaliz, Yunan dilinde ‘Cryptos’ (gizli) ve ‘Analys’ (çözme) kelimelerinden türetilmiştir.

Bir şifreleme sistemini veya sadece şifreli veriyi inceleyerek, şifreli verinin açık halini şifrelemede kullanılan anahtarı bilmeden elde etmeye çalışan kriptoloji dalıdır.

### 2.1.2 Kriptoloji tarihi

Kriptoloji insanlığın yaratılışından itibaren çeşitli evreler geçirerek günümüze ulaşmıştır. Özellikle internet teknolojisinin gelişmesiyle kriptolojiye olan ihtiyaç artmıştır.

Kronolojik olarak kriptolojideki gelişmeler şu şekildedir:

- MÖ.60-50 Julius Caesar, normal alfabedeki harflerin yerini belirli bir sisteme göre değiştirerek oluşturduğu şifreleme yöntemini diplomatik işlemlerde kullanmıştı. Bu yöntem açık metindeki her harfin alfabede kendisinden 3 harf sonraki harfle değiştirilmesine dayanıyordu.
- 1586’da Vigenere şifreleme ile ilgili bir kitap yazdı.
- 1623’de Sir Francis Bacon Stenografiyi buldu.
- 2. Dünya savaşında Almanlar, Arthur Scherbius tarafından icat edilmiş olan Enigma makinasını kullandılar. Bu makine Alan Turing ve ekibi tarafından çözüldü.
- 1970’de IBM DES’in temelini oluşturan Lucifer algoritmasını geliştirdi.
- 1976’da DES ABD tarafından standart olarak açıklandı.

- 1976'da Withfield Diffie ve Martin Hellman, Açık Anahtar sistemini anlattıkları makaleyi yayınladılar.
- 1978'de Ronald L. Rivest, Adi Shamir ve Leonard M. Adleman RSA algoritmasını buldular.
- 1985'de Neal Koblitz ve Victor S.Miller Eliptik Eğri Kriptografi (ECC) ile ilgili çalışmalar yaptılar.
- 1990'da Xuejia Lai ve James Massey, IDEA algoritmasını buldular.
- 1995'te SHA-1 özetleme algoritması NIST tarafından standart olarak yayınlandı.
- 1997'de ABD'nin NIST Kurumu DES'in yerini alacak bir simetrik algoritma için yarışma açtı.
- 2001'de NIST'in yarışmasını kazanan Belçikalı Joan Daemen ve Vincent Rijmen'e ait Rijndael algoritması, AES adıyla standart haline getirildi.

### 2.1.3 Kriptoloji bilimi nerelerde kullanılır

Kriptolojinin geçmişte daha çok askeri ve diplomatik alanda kullanıldığı görülmüştür. Ancak internet teknolojisinin gelişmesi sonucunda bilgi güvenliğini tehdit eden unsurlar daha zararlı hale gelmiştir ve kriptolojiye olan ihtiyaç artmıştır.

Günümüzde ulaşım araçları, bankacılık uygulamaları, e-ticaret ve e-devlet uygulamaları, cep telefonları vb. gibi pek çok alanda gizliliği sağlamak için kullanılmaktadır.

### 2.1.4 Haberleşmede bilgi güvenliği servisleri

Haberleşmede bilgi güvenliğinin sağlanması için 5 temel özellik ön plana çıkmaktadır:

- Gizlilik
- Bütünlük
- Kimlik Doğrulama
- İnkâr Edememezlik

- Haberleşmenin Sürekliliği

#### 2.1.4.1 Gizlilik

Gizlilik, iletilen bilginin gizli kalmasıdır. Böylece bilgi sadece ona erişme yetkisi olan kişiler tarafından görüntülenebilir.

#### 2.1.4.2 Bütünlük

Taşınan bilginin içeriğinin alıcıya ulaşana kadar ona erişme yetkisi olmayan kişiler tarafından değiştirilememesidir.

#### 2.1.4.3 Kimlik doğrulama

Bilgi saldırgan tarafından alıcıya gönderilebilir. Kimlik doğrulama, bilgiyi gönderen kişinin kimliğinin doğruluğundan emin olmaktır.

#### 2.1.4.4 İnkâr edememezlik

Kişinin yaptığı işleri veya işlediği suçları daha sonradan inkâr etmesinin engellenmesidir. Haberleşmede tarafların birbirinden gelen verileri aldığını veya gönderdiğini inkâr edememesi gerekir.

#### 2.1.4.5 Haberleşmenin sürekliliği

Haberleşmenin kesintiye uğramadan yapılmasıdır. Süreklilik, bilgiye erişme yetkisi olan kişiler tarafından bilgiye kesintisiz ulaşılmasıdır.

### 2.1.5 Haberleşmede elektronik tehditler

Haberleşen iki taraf, bilgisayar ağları, kablolu veya kablosuz iletişim kanalları kullanarak bir bilgiyi bir taraftan diğerine iletirler. Bu esnada haberleşen taraflar çeşitli tehditlerle karşı karşıya kalırlar.

Haberleşmede bilgi güvenliği servislerinin ihlalleri aşağıda belirtilen elektronik tehditlere sebep olmaktadır.

#### 2.1.5.1 Gizlilik ihlali

Haberleşme kanalını dinleyen saldırgan, gönderici ile alıcı arasındaki veri trafiğini dinleyebilir ve yetkisi olmadığı halde verileri ele geçirerek haberleşmenin gizliliğini bozar.

Gizliliği sağlamak için haberleşme sırasında gönderilen bilgiler şifreleme algoritmasıyla şifrelenir. Böylece üçüncü şahıslar şifreli verileri çözemeyecekleri için güvenli bir haberleşme ortamı oluşturulur.

#### 2.1.5.2 Bütünlük ihlali

Haberleşmeye müdahale edip göndericinin verilerini değiştiren saldırgan, alıcıya giden veriyi istediği şekle sokabilir.

Haberleşmede veri bütünlüğünü sağlamak için özetleme algoritmaları kullanılır. Gönderilen verinin özeti alınır, alınan özet ve veri karşı tarafa gönderilir. Karşı taraf alınan verinin özetini çıkartır. Kendisine gönderilmiş özet ile gelen veriden çıkardığı özeti karşılaştırır. Bu işlemde eşitlik sağlanırsa verinin içeriğinin değiştirilmediği anlaşılır. Eşitlik sağlanamamış ise verinin değiştirildiği anlamı çıkar ve bu veriye güvenilmez.

#### 2.1.5.3 Kimlik doğrulama ihlali

Saldırgan, göndericinin kimliğini taklit ederek karşı tarafa farklı bir veri gönderebilir. Eğer alıcı güvenilir bir kimlik doğrulaması yapmıyorsa yanlış verilerle kandırılabilir.

Bu ihlalin önüne geçmek için elektronik imzalar ve sertifikalar kullanılır.

#### 2.1.5.4 İnkâr edememezlik ihlali

Veriyi gönderen veya alan tarafın yaptığı işlemleri inkâr etmesidir. Bu durumu ortadan kaldırmak için elektronik imzalar kullanılmalıdır. Bu sayede veriyi gönderen ve alan kişilerin kayıtları güvenilir bir makamda tutulur.

#### 2.1.5.5 Süreklilik ihlali

Saldırganın, haberleşen iki taraf arasındaki hattı veya haberleşme araçlarını kullanılmaz hale getirerek haberleşmenin sürekliliğini engellemesidir.

Haberleşmenin sürekliliğini sağlamak için yedek sistemler ve alternatif haberleşme kanalları kullanılmalıdır.

### 2.1.6 Kriptoloji terminolojisi

Kriptolojide sıklıkla karşılaşılabilecek terimler vardır. Bu terimlerden en yoğun kullanılanları aşağıda belirtilmiştir.

#### 2.1.6.1 Şifreleme

Verinin kriptolanmasıdır. Şifreleme süreci veriyi gönderen tarafından gerçekleştirilir. Burada amaç içeriğin gizli kalması ve erişim yetkisi olmayan kişilerin veriyi okuyamamasıdır.

#### 2.1.6.2 Deşifreleme

Şifrelenmiş veri üzerindeki şifreyi çözerek ham veriye ulaşmaktır. Alıcı tarafından gerçekleştirilir.

#### 2.1.6.3 Şifreli metin

Verinin özel algoritmalar ile kriptolanması sonucu oluşan metindir.

#### 2.1.6.4 Düz metin

Verinin kriptolanmamış ilk halidir.

#### 2.1.6.5 Açık anahtar

Herkesin görebildiği, şifrelemenin yapıldığı alıcı anahtarıdır.

#### 2.1.6.6 Gizli anahtar

Gizli anahtarlı kriptosistemlerde hem şifreleme hem de deşifreleme için kullanılırken, açık anahtarlı kriptosistemlerde şifrelenmiş verinin deşifrelenmesi için kullanılır.

#### 2.1.6.7 Kaba kuvvet saldırısı

Şifreleme ve deşifrelemede kullanılacak olan anahtara bütün ihtimalleri deneyerek ulaşma çabasıdır.

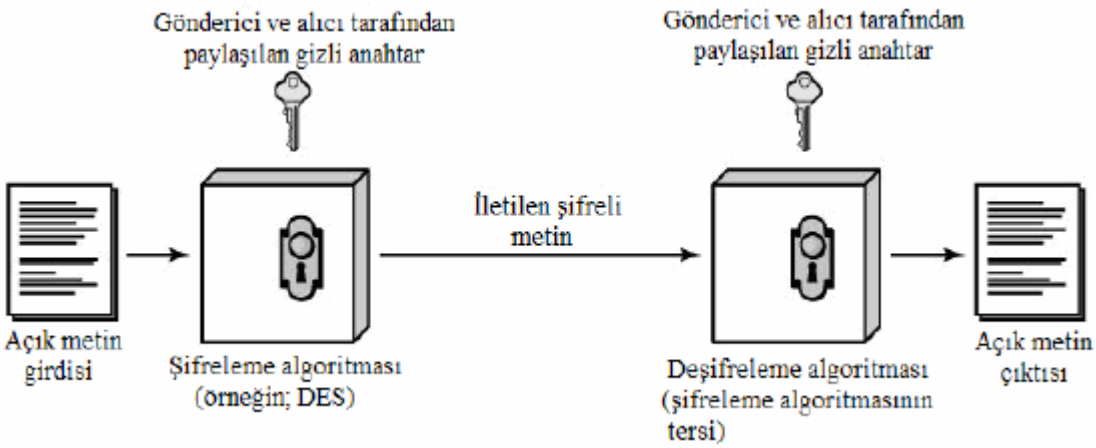
## 2.2 Gizli Anahtarlı Kriptografi

Gizli anahtarlı sistemler, haberleşmek isteyen taraflar arasında, önceden paylaşılan ve saklı tutulması gereken tek bir anahtarın, şifreleme ve deşifreleme işlemlerinde kullanılmasıyla, gizli iletişimin sağlandığı sistemlerdir.

Gizli anahtarlı kriptosistemlerde şifreleme işlemi yerine koyma ve dönüşüm işlemleriyle gerçekleştirilir.

Gizli anahtarlı kriptografi adımları aşağıdaki gibidir:

- Anahtar tektir, gönderici ve alıcı tarafından paylaşılır.
- Açık metin gönderici tarafından şifreleme algoritması ve gizli anahtar ile şifrelenir.
- Şifrelenen metin alıcıya gönderilir.
- Alıcı kullanılan şifreleme algoritması ve elindeki gizli anahtar ile şifreli metini açık metine dönüştürür.




Şekil 2.3 Gizli anahtarlı kriptografi (Başkök 2007)

### 2.2.1 Gizli anahtar algoritmaları

Haberleşmek isteyen taraflar arasında, önceden paylaşılan ve saklı tutulması gereken tek bir anahtarın, şifreleme ve deşifreleme işlemlerinde kullanılmasıyla, gizli iletişimin sağlandığı sistemlerdir.

Algoritmalar yer değiştirme ve dönüşüm işlemlerine göre çalışır.

Sezar, Vigenere, DES, 3DES, AES, RC4, RC5, IDEA gibi algoritmalar bu guruba girer.

#### 2.2.1.1 DES Algoritması

NIST 1973'te ulusal standart olabilecek kriptografik bir algoritma talebinde bulunmuştu. Bu talebe yönelik IBM'nin 1974'te sunduğu LUCIFER algoritması, Ulusal Güvenlik Ajansı'nın uyguladığı değişikliklerin ardından 1997'de DES adıyla resmi bilgi şifreleme standardı olarak kabul edildi. DES dünyanın en çok kullanılan şifreleme algoritmalardan biri oldu.

DES, 64 bit veri bloklarını 56 bit anahtar kullanarak şifreler ve yine 64 bitlik şifrelenmiş metin bloklarına dönüştürür. Deşifrelinirken de 64 bitlik şifrelenmiş veri blokları, 56 bit anahtarla deşifrelenerek yine 64 bitlik düz metine dönüştürülür.

Ancak 56 bit uzunluğundaki bir anahtar günümüzde bilgisayarların işlem hızının gelişmesiyle kaba kuvvet saldırılarına karşı zayıf kalmıştır.

Çizelge 2.1 Kaba kuvvet saldırısı ile anahtarı bulma süresi (Stallings 2011)

| **Anahtar Boyutu (bit)** | **Alternatif Anahtar Sayısı** | **Gerekli Zaman 1 Deneme/µs** | **Gerekli Zaman $10^6$ Deneme/µs** |
|---|---|---|---|
| 32 | $2^{32} = 4.3 \times 10^9$ | 35.8 dk | 2.15 ms |
| 56 | $2^{56} = 7.2 \times 10^{16}$ | 1142 yıl | 10.01 saat |
| 128 | $2^{128} = 3.4 \times 10^{38}$ | $5.4 \times 10^{24}$ yıl | $5.4 \times 10^{18}$ yıl |
| 168 | $2^{168} = 3.7 \times 10^{50}$ | $5.9 \times 10^{36}$ yıl | $5.9 \times 10^{30}$ yıl |

### 2.2.1.2 3DES algoritması

3DES (Üçlü DES), 1978 yılında IBM tarafından geliştirilmiş olan bir şifreleme algoritmasıdır. DES algoritmasının kaba kuvvet saldırılarına karşı koymakta zorlanması üzerine geliştirilmiştir.

3DES algoritmasının özellikleri aşağıdaki gibidir:

- DES algoritmasının 3 kere art arda yapılması şeklinde çalışır.
- DES algoritmasına göre 3 kat daha yavaş çalışır.
- Şifreleme yapmak için uzunluğu 192 bit olan bir anahtar kullanılır.
- Daha gelişmiş bir algoritmaya sahip olan AES algoritmasına göre 6 kat daha yavaş çalışır.

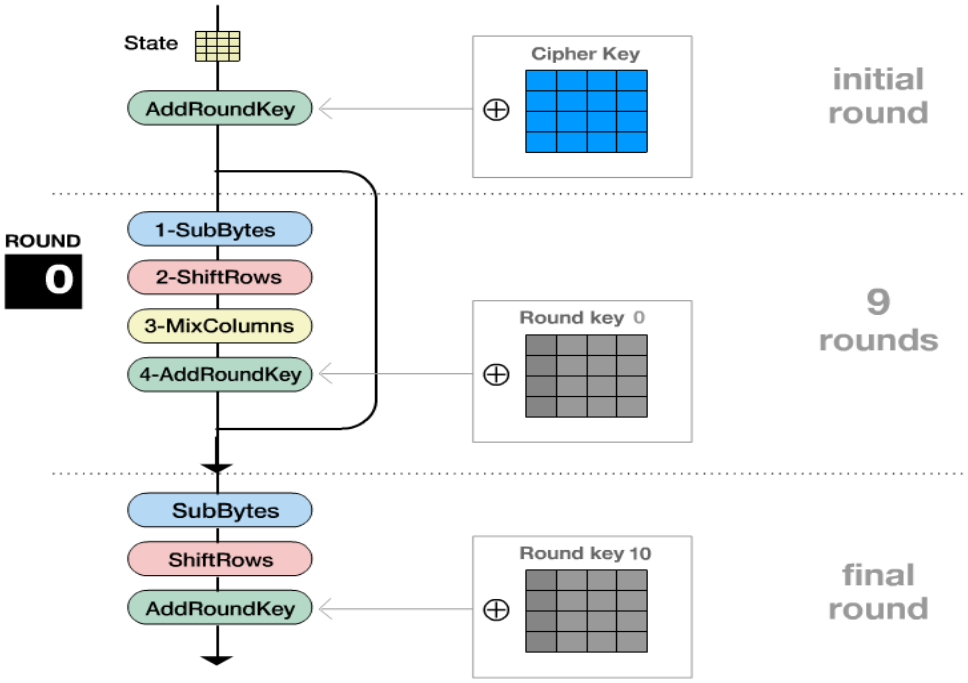
### 2.2.1.3 AES algoritması

AES, uluslararası olarak kullanılan bir şifreleme algoritmasıdır. Vincent Rijmen ve Joan Daemen tarafından geliştirilmiş olup DES algoritmasının zayıf noktalarını karşı oluşturulmuştur.

AES algoritması, 128 bit veri bloklarını 128, 192, 256 bit anahtarlarla şifreleyen bir şifreleme algoritmasıdır. AES'in döngü sayısı anahtar genişliğine göre değişkenlik gösterir.

AES'te 128 bit anahtarla 10 döngüde şifreleme yapılır. 192 ve 256 bit anahtarlarla sırasıyla 12 ve 14 döngüde şifreleme işlemi gerçekleşir. AES, kullandığı anahtar boyutuna göre tanımlanır. Bunlar; "AES-128", "AES-192", "AES-256" gibi adlandırılır. AES algoritmasının her döngüsü dört bileşenden oluşur.

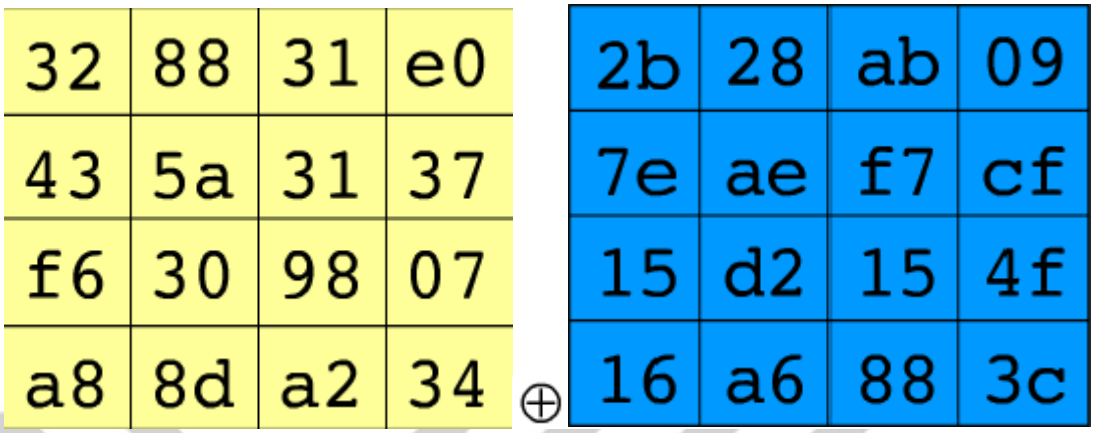
Bu algoritmada 128 bit veri bloğu 4×4 baytlık matrise dönüştürülür. Bu işlemden sonra ilk bölümde, matrise dönüştürülmüş veri ile anahtar XOR işlemine sokulur ve oluşan matris ikinci bölümde 9 kere 4 adımlık döngüye girer. Son bölümde ise sütunların karıştırılması dışındaki adımlar birer kere daha yapılarak şifreli veri oluşturulur.




Şekil 2.4 AES algoritması (Wikipedia 2014)

**AES algoritması işlem adımları:**

**a. Döngüye anahtar eklenmesi**




Şekil 2.5 AES veri ve anahtar

| | | | |
|---|---|---|---|
| 19 | a0 | 9a | e9 |
| 3d | f4 | c6 | f8 |
| e3 | e2 | 8d | 48 |
| be | 2b | 2a | 08 |

Şekil 2.6 Döngüye anahtar eklenmesi

**Tur anahtarı üreteci**

AES anahtar üreteci, farklı uzunluklarda anahtar bloklarıyla (128, 192 veya 256 bit) çalışabilmektedir. Ancak veri bloklarının uzunluğu sabit (128 bit) olduğu için, ürettiği tur anahtarlarının uzunluğu da sabit olup, 128 bittir.

Anahtar üreteci, her turun son adımında, üretilen ara değerle toplanacak olan tur anahtarlarını üretmektedir. Bu nedenle anahtar üretecinin, şifreleme anahtarından, tur sayısı kadar, 128 bitlik tur anahtarı üretmesi gerekir.

### b. Baytların yer değiştirmesi

Bayt yer değiştirme, algoritma içerisindeki tek doğrusal olmayan dönüşümdür. Bayt yer değiştirme katmanı, girişindeki durumun her bir baytını, 'S kutusu' adı verilen bir değiştirme çizelgesi kullanarak, başka bir bayta dönüştürür.

| hex | y | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
| 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 |

x

| | | | |
|---|---|---|---|
| d4 | e0 | b8 | 1e |
| 27 | bf | b4 | 41 |
| 11 | 98 | 5d | 52 |
| ae | f1 | e5 | 30 |

Şekil 2.7 S-kutusu ve baytların yer değiştirmesi

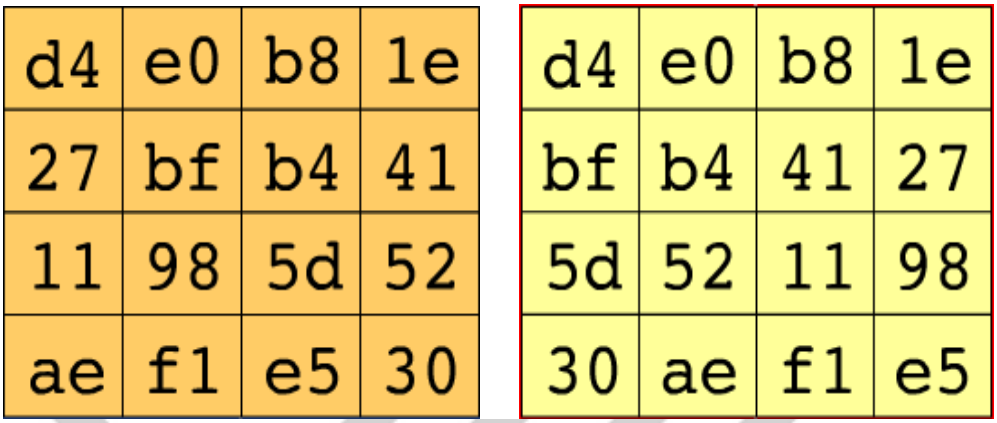
**S- kutuları**

Blok şifreleme algoritmalarının en önemli elemanı S-kutularıdır ve karıştırma işlevini üstlenirler. Bu yüzden iyi bir S-kutusu seçimi şifrenin karmaşıklığını doğrudan etkiler. S-kutuları şifre içerisinde bit bloklarının yer değiştirmesinde kullanılır. Bit blokları S-kutusundan geçirilerek farklı bit bloklarına dönüştürülür.

### c. Satırların ötelenmesi

16 baytlık durum verisi, elemanları baytlar olan 4*4 boyutunda bir matris gibi ele alınır. Satırları öteleme katmanı, adından da anlaşıldığı üzere, bu matrisin satırları (son üç satır) üzerinde işlem yapar. Şifreleme için, ilk satır aynen bırakılır. İkinci satır sağdan sola doğru bir pozisyon değiştirecek şekilde, dairesel olarak, ötelenir. Dairesel öteleme nedeniyle, 1. sütunda bulunan eleman ötelendiğinde 4. sütuna geçer. Üçüncü satır

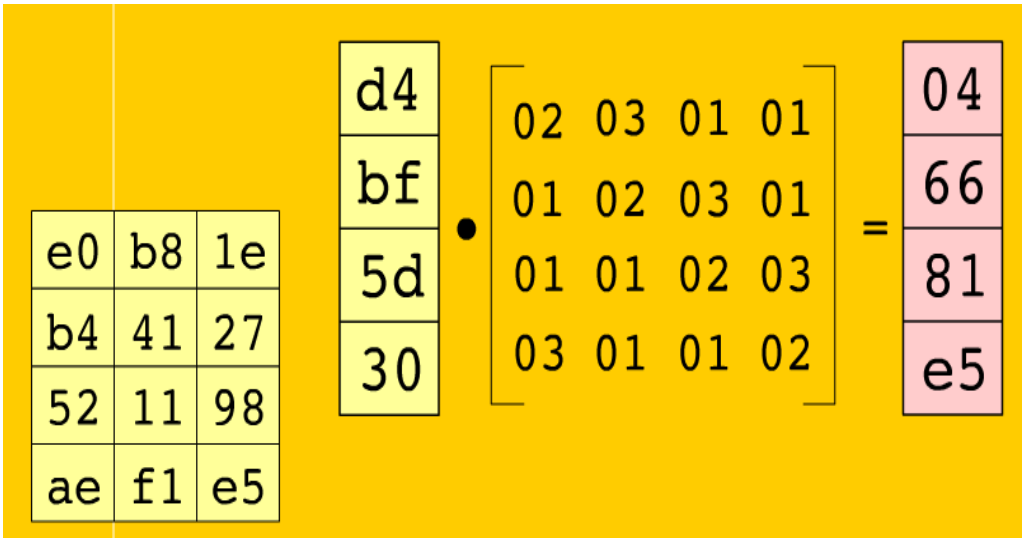
benzer şekilde iki pozisyon, dördüncü satır da üç pozisyon sola doğru, dairesel olarak ötelenir.




Şekil 2.8 Satırların ötelenmesi

**d. Sütunların karıştırılması**

Sütunları karıştırma katmanı, 4*4'lük durum matrisinin sütunları üzerinde işlem yapar. Giriş durum matrisinin her sütunu, sütunları karıştırma dönüşümünden geçirilerek, çıkış durum matrisi elde edilir. Doğrusal bir işlem olan sütunları karıştırma, algoritmaya, difizyon özelliği kazandırır.




Şekil 2.9 Sütunların karıştırılması

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad \begin{aligned} b_0 &= 2a_0 + 3a_1 + 1a_2 + 1a_3 \\ b_1 &= 1a_0 + 2a_1 + 3a_2 + 1a_3 \\ b_2 &= 1a_0 + 1a_1 + 2a_2 + 3a_3 \\ b_3 &= 3a_0 + 1a_1 + 1a_2 + 2a_3 \end{aligned}$$

Şekil 2.10 Sütunların karıştırılmasında kullanılan matris

**b0** = {d4.02} **XOR** {bf.03} **XOR** {5d.01} **XOR** {30.01}

**{d4.02}**

**d4** => 1101 0100

**02** => 10

**d4.02** = **1011 0011**

**{bf.03}**

**bf** => 1011 1111

**03** = > 11 => 10 **XOR** 01

**bf.03** = **1101 1010**

**{5d.01}**

**5d =>** 0101 1101

**5d.01** = **0101 1101**

**{30.01}**

**30** => 0011 0000

**30.01** = **0011 0000**

**b0** = (1011 0011) **XOR** (1101 1010) **XOR** (0101 1101) **XOR** (0011 0000)

**b0** = **0000 0100** (Binary)

**b0** = **04** (Hexadecimal)

Matrisin bütün elemanları bu dönüşümü yaşadıktan sonra ortaya çıkan sonuç aşağıdaki gibidir.

| | | | |
|---|---|---|---|
| 04 | e0 | 48 | 28 |
| 66 | cb | f8 | 06 |
| 81 | 19 | d3 | 26 |
| e5 | 9a | 7a | 4c |

Şekil 2.11 Sütunların karıştırılması

Simetrik şifrelemenin güvenli kullanımı için:

- Güçlü bir şifreleme algoritmasına ve anahtara ihtiyaç vardır. En azından, bir ya da daha fazla şifreli metine erişimi olan bir kişi, şifreli metini deşifre edememeli veya anahtarı çözememelidir.

- Anahtarın güvenliğine son derece önem verilmeli ve anahtar gizli tutulmalıdır. Eğer herhangi birisi anahtarı ele geçirmişse ve algoritmayı da biliyorsa bu anahtar ile yapılan tüm iletişim dinlenebilir.

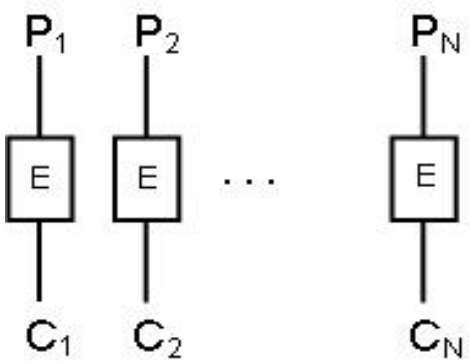
Gizli anahtarlı sistemlerde temel problem, göndericinin ve alıcının, anahtarın üçüncü bir kişinin eline geçmesini engelleyerek ortak bir anahtar üzerinde anlaşmalarıdır.

Gizli anahtarlı kriptosistemler, blok şifreleme sistemleri ve dizi şifreleme sistemleri olarak ikiye ayrılmaktadır.

### 2.2.2 Blok şifreleme

Blok şifreleme sistemlerinde şifreli metin veya orijinal metin belirli boyutlardaki bloklara ayrılır ve ayrılan blok bir bütün halinde işlenir.

Şekil 2.12'de gösterildiği gibi her biri aynı boyuttaki $P_1$, $P_2$, …, $P_N$ açık metin blokları şifreleme işleminden geçirilerek $C_1$, $C_2$, …, $C_N$ şifreli metinlerine dönüştürülür.




Şekil 2.12 Blok Şifreleme (Stallings 2011)

Blok şifreleme algoritmalarına AES ve DES örnek gösterilebilir.

Blok şifreleme algoritmalarının gücü, anahtar uzunluğuna, blok uzunluğuna, S-kutularına ve döngü sayısına bağlıdır.

#### 2.2.2.1 Blok şifreleme modları

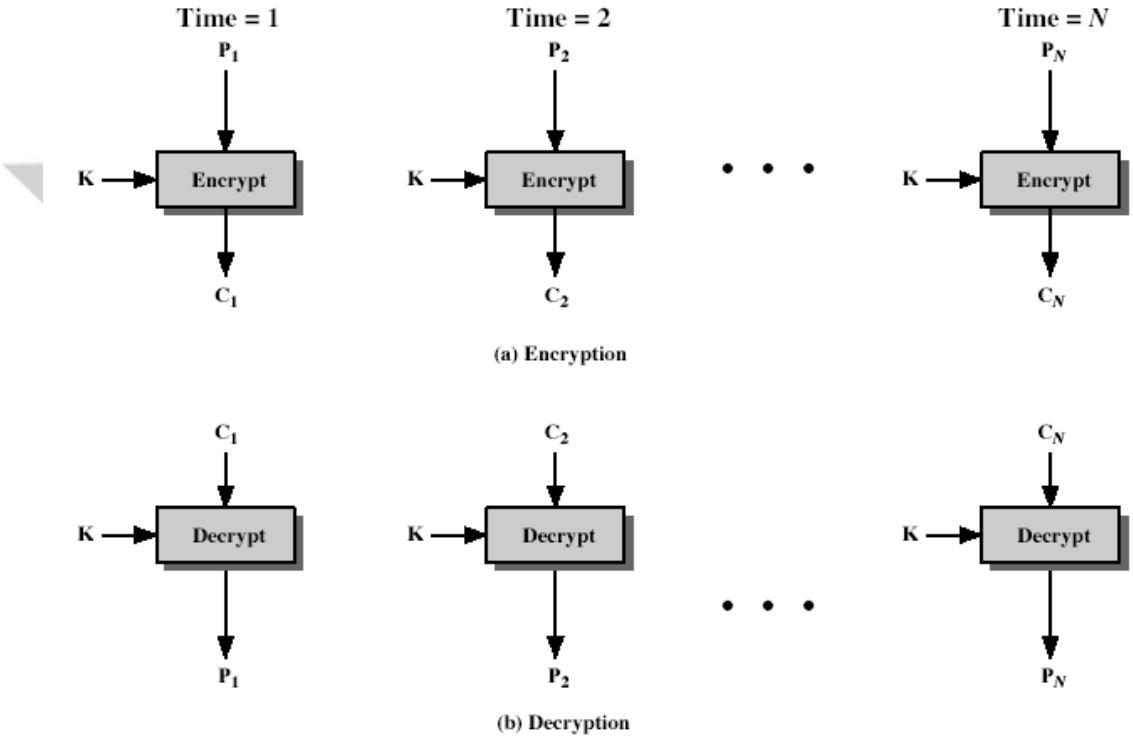
Blok şifreyi farklı uygulamalarda kullanabilmek için, NIST tarafından çalışma modelleri belirlenmiştir.

Farklı çalışma modelleri kullanılarak, kriptografik algoritmaların etkisi güçlendirilebilir veya algoritmanın herhangi bir uygulama için uygun hale getirilmesi sağlanmış olur.

**a. Elektronik kod kitabı modeli**

Açık metinin her seferinde bir bloğu işleme alınır ve her bir açık metin bloğu aynı anahtarla şifrelenir. Her bloğun x bitten oluştuğunu düşündüğümüzde, veri eğer x bitten fazlaysa, veri x bitlik bloklara bölünür ve gerekirse son blok x bite tamamlanır. Deşifreleme işleminde de şifrelemede olduğu gibi her seferinde bir blok aynı anahtarla işleme girer.

ECB modeli, şifreleme anahtarı gibi küçük boyuttaki veriler için idealdir. ECB modelinin en önemli özelliği ise, bir veri içerisinde birden fazla aynı b-bitlik blok bulunuyorsa, karşılığında üretilen şifreli metin blokları da hep aynı olacaktır. Bu yüzden uzun veriler için ECB modelinin kullanılması güvenli olmayabilir. Eğer veri oldukça düzenli bir yapıya sahipse, kriptoanalist için bu düzenlilikleri açığa çıkarmak mümkündür ve bu da istenmeyen sonuçlara yol açabilir.




Şekil 2.13 Elektronik kod kitabı modeli (Stallings 2011)

ECB modelinin çalışma prensibi şu şekilde ifade edilebilir:

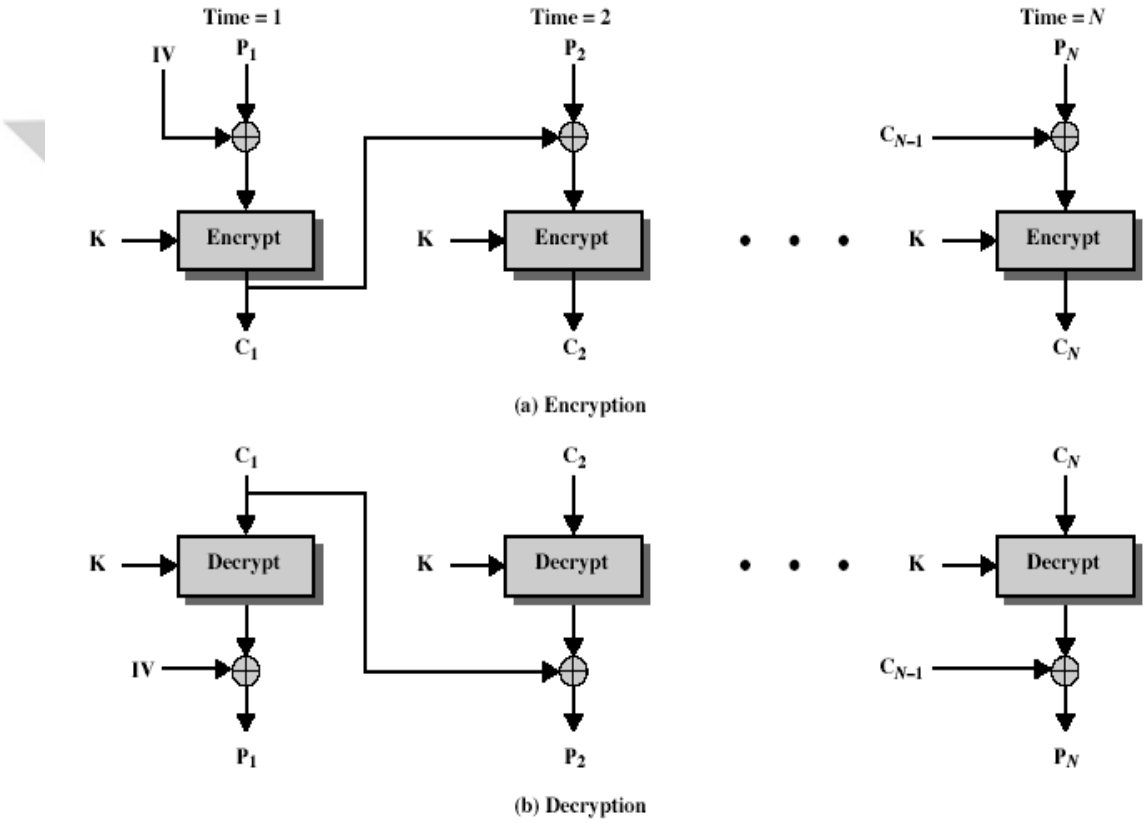
Şifreleme: Cx = E(K, Px)

Deşifreleme: Px = D(K, Cx)

### b. Şifre blok zincirleme modeli

ECB modelinin güvenlik açıklarını kapatmak için geliştirilmiş olan, aynı açık metin bloğunun, tekrarlı olsa dahi, farklı şifreli metin bloklarını ürettiği bir modeldir.

CBC modelinde, şifreleme algoritmasının girişi, o anki açık metin bloğu ve bir önceki şifreli metin bloğunun XOR'lanmasından meydana gelir ve her blok için aynı anahtar kullanılır. Şekil 2.14'te CBC modeline yer verilmiştir. Bu modelde, her bir açık metin bloğu için, şifreleme fonksiyonunun girişinin, açık metin bloğu ile doğrudan bir bağlantısı bulunmamaktadır, dolayısıyla tekrarlayan kalıpların gözlenmesi mümkün değildir. Deşifrelemede, her bir şifre bloğu deşifreleme algoritmasına sokulur ve sonuç, açık metin bloğunu üretmek için, bir önceki şifreli metin bloğuyla XOR'lanır.




Şekil 2.14 Şifre blok zincirleme modeli (Stallings 2011)

CBC modelinin çalışma prensibi şu şekilde ifade edilebilir:

Şifreleme: $C_j = E(K, [C_{j-1} \oplus P_j])$

Deşifreleme: $P_j = C_{j-1} \oplus D(K, C_j)$

Şifreli metnin ilk bloğunu oluşturmak için, bir başlangıç vektörü (IV) ile açık metnin ilk bloğu XOR'lanır. Deşifrelemede ise, açık metnin ilk bloğunu elde etmek için, IV, deşifreleme algoritmasının çıktısıyla XOR'lanır.

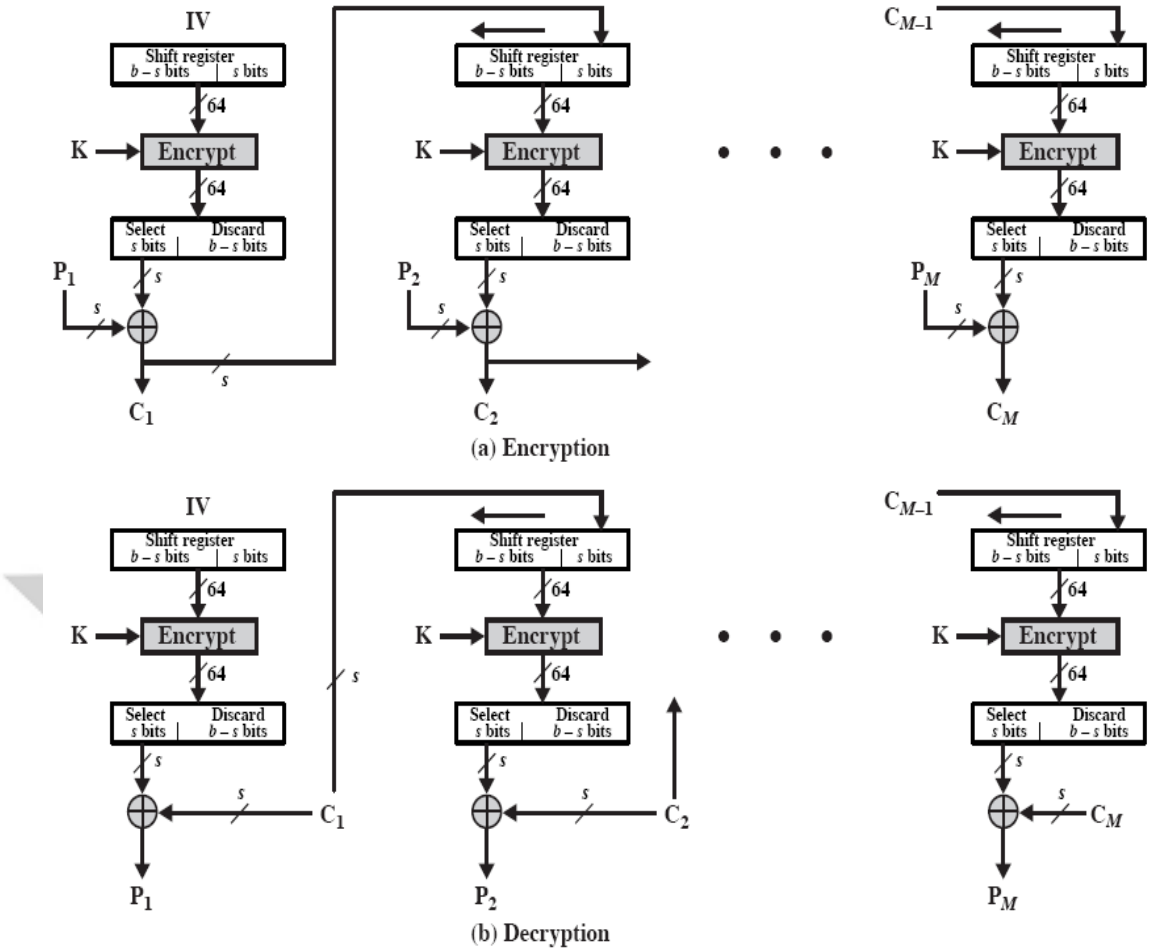
IV, şifre bloğuyla aynı uzunluktaki bir veri bloğudur ve yalnız gönderici ve alıcı tarafından bilinmesi gereken, üçüncü şahıslar tarafından tespit edilememesi gereken bir bilgidir.

**c. Şifre geri besleme modeli**

CFB ya da OFB modelini kullanarak, blok şifreyi akan şifreye dönüştürmek mümkündür. Akan şifre, veriyi blok uzunluğunun tam katı olacak şekilde doldurma işlemine ihtiyaç duymaz ve gerçek zamanlı çalışabilir. Böylece, bir karakter dizisi iletiliyorsa, karaktere dayalı bir akan şifre kullanılarak, her bir karakter anında şifrelenebilir ve taşınabilir.

Akan şifrenin güzel bir özelliği de, şifreli metnin, açık metinle aynı uzunlukta olmasıdır. Eğer 8-bit karakterler iletiliyorsa, her karakter, 8-bit şifreli metin çıkışı üretecek şekilde şifrelenmelidir. 8-bitten fazlası üretilirse, iletim kapasitesi boşa harcanmış olur.

Şekil 2.15'te CFB modeli gösterilmiştir. İletim biriminin s-bit olduğu varsayılmaktadır ve s için yaygın kullanılan değer s = 8'dir. CFB modelinde, her bir açık metin biriminin şifreli metni, önceki tüm açık metinlerin fonksiyonu olacak şekilde, açık metin birimleri birbirine zincirlenir. Bu durumda, açık metin, b-bitlik birimlerden ziyade, s-bit bölümlere ayrılmış olur.




Şekil 2.15 Şifre geri besleme modeli (Stallings 2011)

CFB modelinin çalışma prensibi, Ss(X), X'in en soldaki s biti olarak varsayılırsa şu şekilde ifade edilebilir:

Şifreleme: $C1 = P1 \oplus Ss[E(K, IV)]$

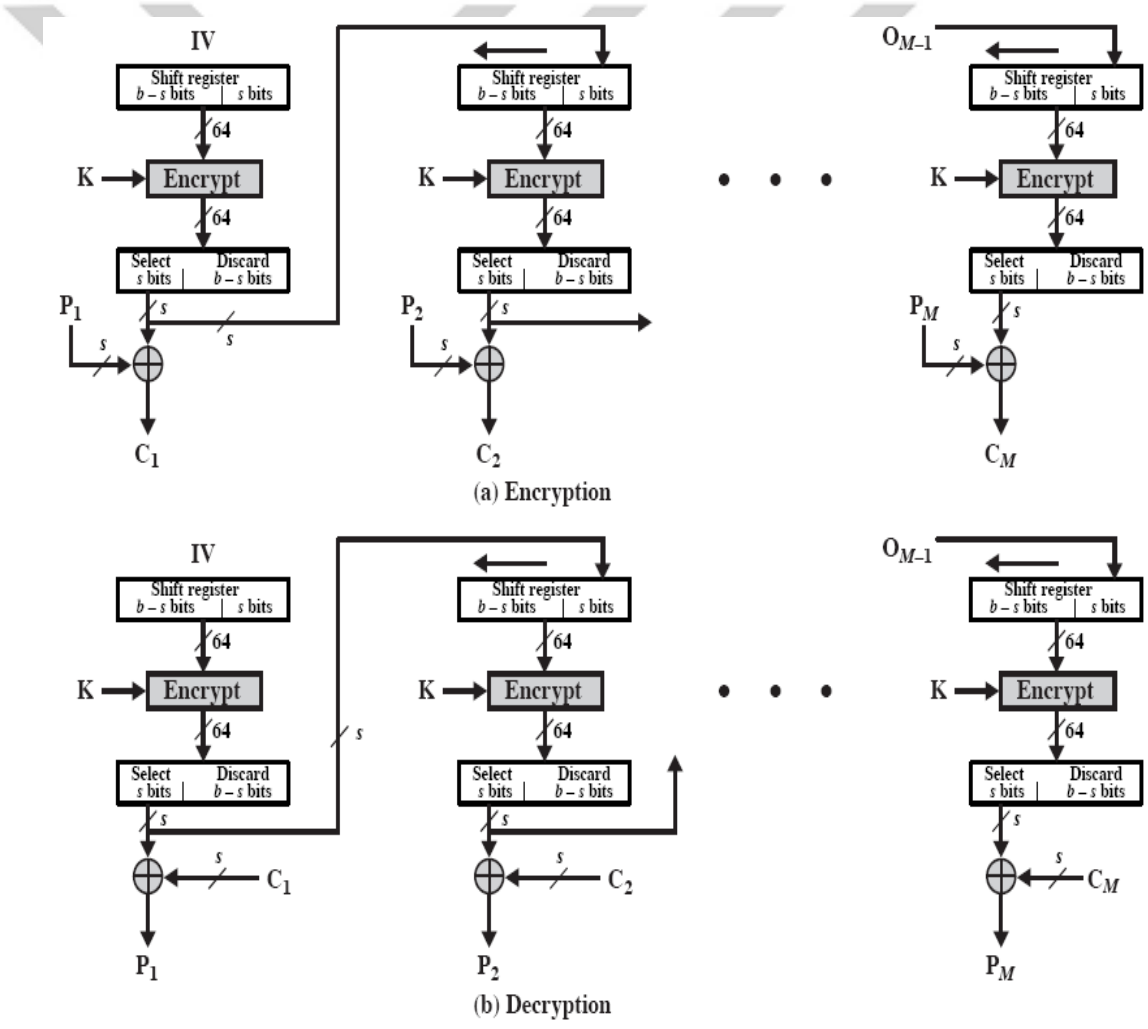
Deşifreleme: $P1 = C1 \oplus Ss[E(K, IV)]$

Şifreleme işleminde, şifreleme fonksiyonunun girişi, başlangıç vektörüne ayarlı, b-bit kaydırmalı yazmaçtır. İletilecek olan şifreli metnin ilk birimini (C1) üretmek için, şifreleme fonksiyonunun çıkışının en soldaki s biti, açık metnin ilk bölümü olan P1 ile XOR'lanır. Bununla birlikte, kaydırmalı yazmacın içeriği s-bit sola kaydırılır ve C1, kaydırmalı yazmacın en sağındaki s-bite yerleştirilir. Bu işlem tüm açık metin birimleri şifrelenene kadar devam eder.

Deşifrelemede, açık metin birimini üretmek için, alınan şifreli metin biriminin, şifreleme fonksiyonunun çıkışıyla XOR'lanması haricinde, aynı düzen kullanılır. Dikkat edilmesi gereken, kullanılanın, deşifreleme fonksiyonu değil, şifreleme fonksiyonu olduğudur.

### d. Çıktı geri besleme modeli

OFB modeli, şekil 2.16'da görüldüğü gibi, yapısal olarak CFB modeline benzemektedir. CFB modelinde, kaydırmalı yazmaca geri beslenen, şifreli metin birimi iken; OFB modelinde, şifreleme fonksiyonunun çıkışıdır.



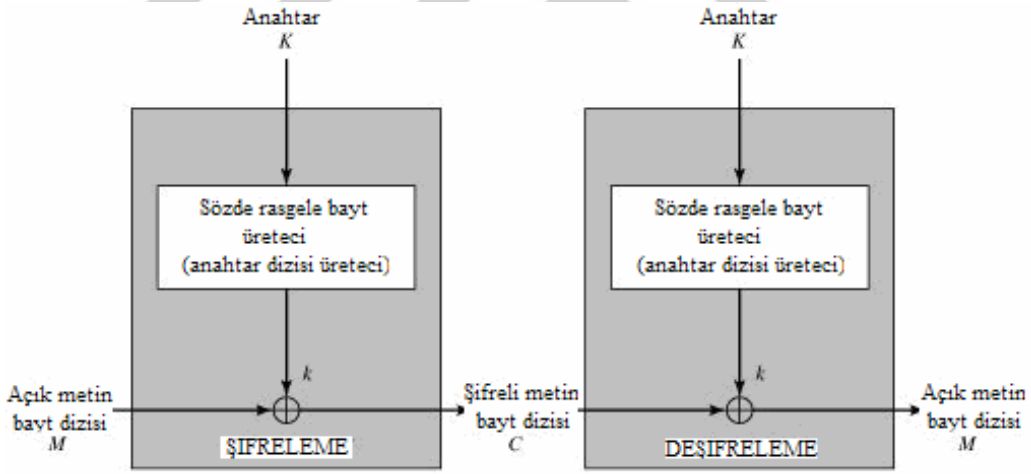

Şekil 2.16 Çıktı geri besleme modeli (Stallings 2011)

OFB modelinin bir avantajı, iletimde, bit hatalarının yayılmamasıdır. Örneğin, C1'de bit hatası meydana gelirse, sadece elde edilen P1 bundan etkilenir; sonraki açık metin birimleri kesintiye uğramaz. Fakat CFB modelinde, C1 aynı zamanda kaydırmalı yazmacın girişi olduğundan, ilave kesintiye sebep olur.

### 2.2.3.2 Dizi şifreleme

Üretilen anahtar ile açık metin XOR işlemine sokularak şifreli metin elde edilir. Alıcı şifreyi çözmek için üretilen anahtarı elde etmelidir. Böylece şifreli metin ile anahtar XOR işlemine girer ve sonuçta açık metin bulunur.

Daha çok ses ve görüntü gibi veri transferlerinde tercih edilir. RC-4 algoritması akan şifreleme sistemine bir örnektir.




Şekil 2.17 Dizi şifreleme (Başkök 2007)

Dizi şifrelemenin çalışma prensibi şu şekilde ifade edilebilir:

Açık metin m = m1 m2…mn

Anahtar k = k1 k2…kn

Kapalı metin c = c1 c2...cn

Burada her i için ci = mi ⊕ ki

Blok şifreleme algoritmaları OFB ve CFB modlarında dizi şifreleme algoritmaları gibi kullanılabilir.

### 2.2.3 Gizli anahtar algoritmalarının avantajları ve dezavantajları

**a) Avantajlar**

- Algoritmalar hızlıdır.
- "Gizlilik" güvenlik servisini sağlar.
- Algoritmaların donanımla uygulanması kolaydır.

**b) Dezavantajlar**

- Güvenli anahtar dağıtımı zordur.
- "Bütünlük", "Kimlik Doğrulama" ve "İnkâr Edememezlik" güvenlik servisleri sağlanamaz.
- Elektronik imza desteği yoktur.

## 2.3 Açık Anahtarlı Kriptografi

Başlangıcından günümüze kadar, neredeyse bütün kriptografik sistemler, yerine koyma ve permütasyon işlemlerinin temel alınmasıyla oluşturulmuşlardır. Sadece elle hesaplanabilen algoritmalarla çalışabilme döneminden sonra, şifreleme/deşifreleme yapan rotor makinelerinin ortaya çıkması sonucunda, geleneksel kriptografide büyük bir gelişme kaydedilmiştir. Mevcut bilgisayarlarla daha karmaşık sistemler tasarlanmış ve en tanınanlarından olan IBM'in Lucifer girişimi gelişerek DES'i oluşturmuş ve DES'i dünyadaki kriptografi teknikleri arasında en yüksek seviyeye getirmiştir. Rotor makineleri ve DES, önemli avantajlar sunmalarına rağmen, halen yerine koyma ve permütasyon işlemlerine bağımlıdırlar.

Açık anahtarlı kriptografi ile ilgili ilk makalelerin ortaya atılmasına kadar olan süreçte kullanılan simetrik sistemler göz önünde bulundurulduğunda, açık anahtarlı kriptografinin gelişmesi, bütün kriptografi tarihindeki en büyük devrimlerden biridir.

İnsanlığa duyurulan ilk açık anahtar algoritması, Withfield Diffe ve Martin Hellman'ın 1976 yılında yayımladıkları "New Directions of Cryptography" isimli makalelerinde yer aldı ve bu algoritma Diffie-Hellman anahtar değişimi idi.

Açık anahtarlı kriptosistemler:

- Şifreleme/Deşifreleme,
- Elektronik imza,
- Anahtar değişimi

gibi işlemlerde kullanılabilir.

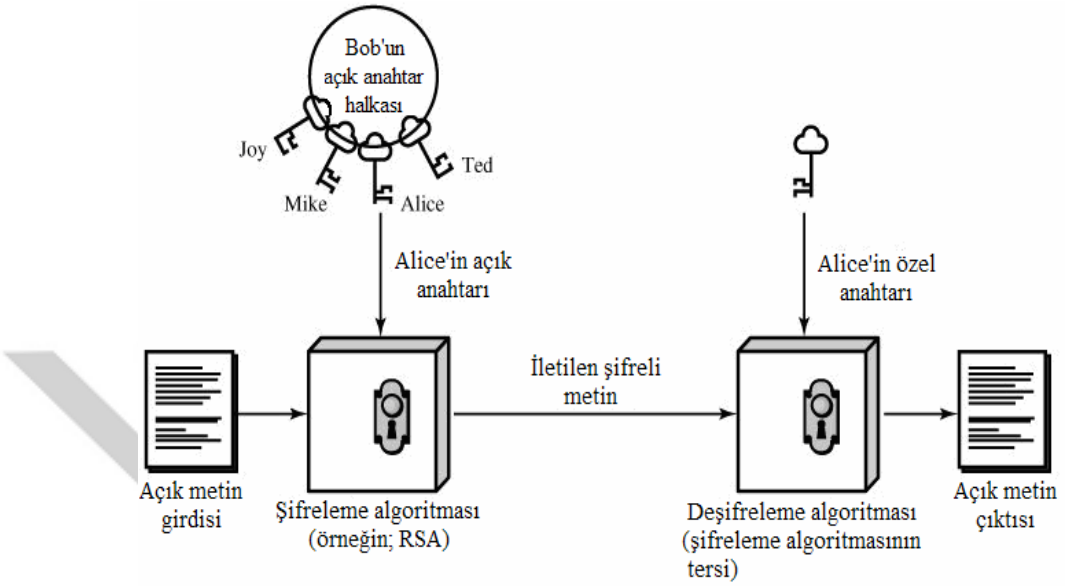
Çizelge 2.2 Açık anahtarlı kriptosistemlerin kullanım alanları

| **Algoritma** | **Şifreleme Deşifreleme** | **Elektronik İmza** | **Anahtar Değişimi** |
|---|---|---|---|
| RSA | Evet | Evet | Evet |
| Eliptik Eğri | Evet | Evet | Evet |
| Diffie-Hellman | Hayır | Hayır | Evet |
| DSS | Hayır | Evet | Hayır |

Açık anahtarlı kriptografi adımları aşağıdaki gibidir:

- Her bir kullanıcı, verinin şifreleme ve deşifreleme işlemlerinde kullanılacak bir anahtar çifti üretir.
- Her kullanıcı, şifreleme anahtarını herkesçe erişilebilecek bir dosya kaydederek paylaştırır. Bu anahtar açık olandır, özel anahtar saklı tutulur.
- Eğer, A, B'ye gizli bir veri yollamak isterse, veriyi B'nin açık anahtarını kullanarak şifreler.

- B, veriyi aldığında, bu veriyi kendi özel anahtarını kullanarak deşifre eder. Diğer hiçbir alıcı veriyi deşifreleyemez, çünkü veriyi deşifre edecek olan özel anahtarı sadece B bilir.




Şekil 2.18 Açık anahtarlı kriptografi (Başkök 2007)

Şekil 2.18'den de anlaşıldığı üzere her kullanıcı, diğerlerinin açık anahtarlarına erişim hakkına sahiptir ve kullanıcılar özel anahtarlarını yerel olarak yaratırlar. Bu nedenle, özel anahtarların paylaşılmasına gerek yoktur. Kullanıcının özel anahtarı korunduğu ve gizli tutulduğu sürece haberleşme güvendedir.

### 2.3.1 Açık anahtar algoritmaları

Şifreleme için bir anahtar ve şifre çözme içinse bu anahtarla matematiksel ilişkisi olan ancak farklı bir anahtarın kullanıldığı algoritmalardır. Asimetrik algoritmalar simetrik algoritmalarda kullanılan yerine koyma ve permütasyon işlemlerinden ziyade temelini karmaşık matematik işlemlerinden almaktadır. Asimetrik algoritmaları kullanan taraflar; aynı şifreleme algoritmasını kullanmaktadırlar.

Açık anahtarlı algoritmalar, gizli anahtarlı algoritmaların yerini almak için değil aksine onları tamamlamak için geliştirilmiştir. Gizli anahtarlı algoritmalar gibi açık anahtarlı algoritmalarda da güvenlik anahtarın boyutuna bağlıdır. Ancak açık anahtarlı algoritmalarda açık anahtar ve gizli anahtarın birbiri ile matematiksel ilişkisi bulunması ve açık anahtara herkesin ulaşabilmesi sebebiyle anahtar boyutları simetrik sistemlere göre daha büyük olmak zorundadır.

Yaygın olarak kullanılan açık anahtarlı algoritmalardan bazıları RSA, Diffie-Hellman ve DSA'dır.

#### 2.3.1.1 RSA algoritması

RSA algoritması 1978 yılında R.Rivest, A.Shamir ve L.Adleman tarafından geliştirilmiştir. En çok kullanılan açık anahtar şemasıdır. Güvenliği büyük tam sayıların çarpanlarına kolayca ayrılamaması problemine dayanır.

RSA, anahtar dağıtımının yanında şifreleme, deşifreleme sistemlerinde ve sayısal imza işlemlerinde güvenli bir şekilde kullanılır.

İki ayrı anahtar kullanılır. Anahtarlardan birisi herkesin görebileceği açık anahtar, diğeri sadece kişinin kendinde bulunan gizli anahtardır.

Veri alıcının paylaştığı açık anahtarla şifrelenerek gönderilir ve yalnızca alıcıdaki gizli anahtarla açılabilir.

Anahtarlar arasında matematiksel bağlantı bulunmaktadır.

| | |
|---|---|
| $p$, $q$ asal sayıları seçilir | $p \neq q$ |
| $n = p \times q$ hesaplanır | |
| $\varphi(n) = (p - 1)(q - 1)$ | |
| $e$ tam sayısı seçilir | $1 < e < \varphi(n)$; $\gcd(\varphi(n), e) = 1$ |
| $d$ hesaplanır | $d.e \equiv 1 \pmod{\varphi(n)}$ |
| Açık anahtar | $PU = \{e, n\}$ |
| Özel anahtar | $PR = \{d, n\}$ |

Şekil 2.19 RSA anahtar oluşturma (Stallings 2011)

**Şifreleme adımları**

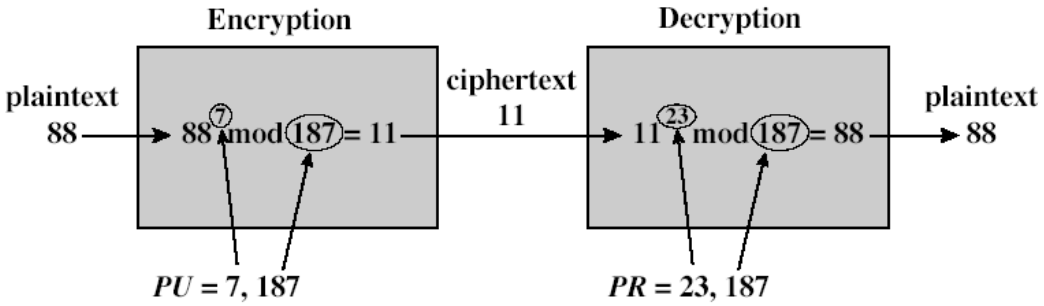
- Çok büyük iki asal sayı üretilir. ($p$ ve $q$)
- $n = p$ x $q$ işlemi ile Mod değeri üretilir.
- $\varphi(n) = (p - 1)$ x $(q - 1)$
- 1 ile $\varphi(n)$ arasında ve $\varphi(n)$ ile aralarında asal olacak şekilde bir açık anahtar bulunur.

**Örnek**

$p = 17$ ve $q = 11$ olsun

$n = p$ x $q = 187$ olur

$\varphi(n) = 16$ x $10 = 160$ ve $e = 7$ olsun, $d$ x $e = 1 \pmod{\varphi(n)}$, $d = 23$ olur




Şekil 2.20 RSA şifreleme / deşifreleme işlemi

### 2.3.2 Açık anahtar algoritmalarının avantajları ve dezavantajları

**a) Avantajlar**

- Kriptoanalize karşı dirençli.
- "Bütünlük", "Kimlik Doğrulama" ve "İnkâr Edememezlik" güvenlik hizmetleri sağlanabilir.
- Elektronik imza desteği var.

**b) Dezavantajlar**

- Algoritmalar simetrik algoritmalara göre anahtar uzunluklarının fazla olması sebebiyle çok daha yavaştır.

## 2.4 Kriptosistemlerin Karşılaştırılması

Çizelge 2.3'te simetrik ve asimetrik kriptosistemlerin bilgi güvenliği servislerine karşı yeteneklerinden bahsedilmiştir.

Çizelge 2.3 Simetrik ve asimetrik kriptosistemlerin karşılaştırılması

| Servisler | Simetrik Sistemler | Asimetrik Sistemler |
|---|---|---|
| Gizlilik | Sağlar | Sağlar |
| Bütünlük | - | Sağlar |
| Kimlik Doğrulama | - | Sağlar |
| İnkâr Edemezlik | - | Sağlar |
| Hesaplama Hızı | Yüksek | Düşük |

Çizelge 2.4'te en çok kullanılan sistemler olan DES, AES ve RSA'nın geliştirilme tarihi, anahtar boyutu, blok boyutu, şifreleme/deşifreleme anahtarı ve performansı, güç tüketimi, donanım ve yazılım ile uygulamaları gibi özelliklerinden bahsedilmiştir.

Çizelge 2.4 DES, AES ve RSA karşılaştırması

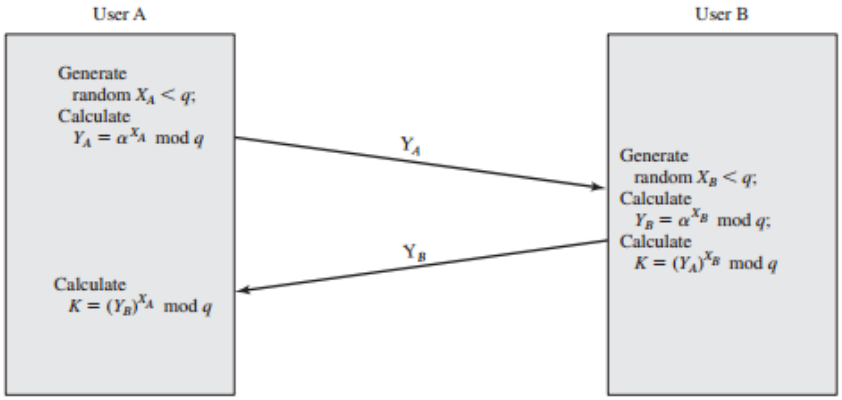
| **Özellikler** | **DES** | **AES** | **RSA** |
|---|---|---|---|
| Geliştirilme Tarihi | 1977 | 2000 | 1978 |
| Anahtar Boyutu | 56 | 128-192-256 | 1024 ve üstü |
| Blok boyutu | 64 bit | 128 bit | 512 bit ve üstü |
| Şifreleme/Deşifreleme Anahtarı | Aynı | Aynı | Farklı |
| Şifreleme | Orta | Hızlı | Yavaş |
| Deşifreleme | Orta | Hızlı | Yavaş |
| Güç Tüketimi | Düşük | Düşük | Yüksek |
| Donanım ve Yazılım Uygulamaları | Donanımda daha iyi | Hızlı | Yavaş |

Yapılan çalışmalar gösteriyor ki, şifreleme ve deşifreleme işlemlerinde simetrik sistemler asimetrik sistemlere göre çok daha iyi performans göstermiştir.

## 2.5 Diffie-Hellman Anahtar Değişimi

Withfield Diffie ve Martin Hellman 1976 de yayınladıkları "New Directions In Cryptography" adlı makale ile anahtar paylaşımı problemine ilk pratik çözümü getirmişlerdir. Ayrık logaritma probleminin çözümünün zorluğuna dayanan bu sistem ile tarafların daha önce bir araya gelmesine gerek duyulmadan açık bir kanal üzerinden gerekli parametreleri birbirlerine göndererek ortak anahtar oluşturmaları sağlanır.

Diffie–Hellman yalnızca anahtar değişimi için tasarlanmıştır, şifreleme/deşifreleme ve elektronik imza işlemlerinde kullanılamaz.




Şekil 2.21 Diffie–Hellman anahtar değişimi (Stallings 2011)

$q$ asal sayı ve $\alpha$ primitive kök olsun. $\alpha$ ve $q$ herkes tarafından bilinsin. $A$ ve $B$ kullanıcıları anahtar oluştururken:

i. Kullanıcı $A$, $X_A < q$ şartını sağlayan rastgele bir $X_A$ sayısı seçer ve $Y_A = \alpha^{X_A} \bmod q$ hesaplar ve $B$'ye gönderir.

ii. Kullanıcı $B$, $X_B < q$ şartını sağlayan rastgele bir $X_B$ sayısı seçer ve $Y_B = \alpha^{X_B} \bmod q$ hesaplar ve $A$'ya gönderir.

iii. Kullanıcı $A$, ortak anahtar olarak $K = (Y_B)^{X_A} \bmod q$ hesaplar, $B$ de aynı şekilde $K = (Y_A)^{X_B} \bmod q$ hesaplar. $A$ ile $B$ ortak $K$ anahtarında anlaşmış olurlar.

## 2.6 Eliptik Eğri Diffie-Hellman Anahtar Değişimi

Eliptik eğrilerin kriptografide kullanılması birbirinden bağımsız olarak 1986'da Miller ve 1987'de Koblitz tarafından önerilmiştir.

Günümüzde açık anahtarlı kriptografik uygulamalar başlıca üç ana matematiksel probleme dayandırılarak geliştirilmektedir.

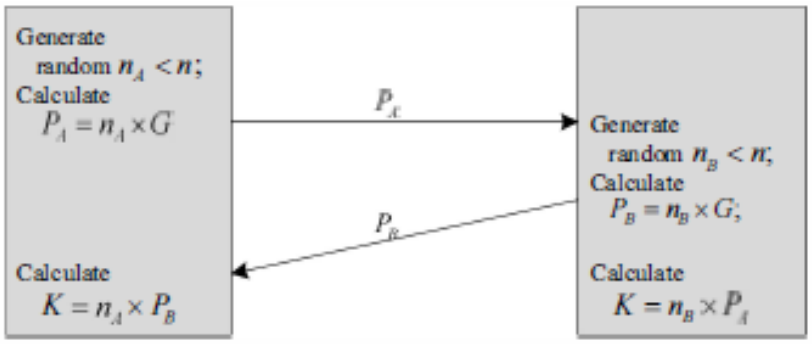
Bunlar;

- Bir sayının çarpanlarına ayrılması problemi

- Ayrık logaritma problemi
- Eliptik eğri ayrık logaritma problemi

Son dönemlerde kurum ve kuruluşlar, güvenlik gereksinimlerini karşılamak üzere daha yüksek boyutlu anahtarlara ihtiyaç duymuşlar, bu gereksinim ise gerek hafıza ihtiyacı, gerekse işlem yükü açılarından maliyet ile doğru orantılı olarak kurum ve kuruluşların sistemlerine büyük yük getirmiştir. Eliptik eğri temeline dayanan kriptosistemler, anahtar boyutlarında ve veri transfer hızlarında büyük gelişmelere olanak sağlamaktadır.

Eliptik eğri yönteminin diğer yöntemlere karşı en büyük avantajı, daha küçük boyutlu anahtarlar kullanılarak yapılan işlemlerin diğer yöntemlerde kullanılan büyük boyutlu anahtarlarla yapılan işlemler gibi yüksek güvenlik sağlayabilmesidir. Bu sayede işlem süresi, bellek kullanımı ve bant genişliği gereksinimi azalmaktadır. Eliptik Eğri Diffie-Hellman, Diffie-Hellman gibi taraflar arasında ortak gizli anahtar oluşturmak amacıyla kullanılmaktadır. Ancak Diffie-Hellman ile AES-256 da kullanılacak olan anahtarı oluşturmak için kullanılacak olan anahtarlar çok daha büyük boyutta olmaktadır.




Şekil 2.22 Eliptik Eğri Diffie-Hellman (Stallings 2011)

$y^2=x^3+ax+b$ denkleminde her a ve b için farklı eliptik eğriler meydana çıkmaktadır. Taraflar a ve b noktalarını belirleyerek bir eğri denklemi üzerinde anlaşırlar ve bu eğri üzerinde olan $G$ = (x, y) noktasını belirlerler.

i. *A* kullanıcısı, $1 < n_A < n$ şartını sağlayan bir $n_A$ sayısını gizli anahtar olarak seçer, seçmiş olduğu $n_A$ gizli anahtarını kullanarak $P_A = n_A$ x $G$ açık anahtarını belirler ve *B* kullanıcısına gönderir.

ii. $B$ kullanıcısı benzer bir şekilde $n_B$ gizli anahtarını seçer, $P_B = n_B$ x $G$ açık anahtarını oluşturur ve $A$ kullanıcısına gönderir.

iii. $A$ kullanıcısı $K_A = n_A$ x $P_B$ ve $B$ kullanıcısı da $K_B = n_B$ x $P_A$ ortak gizli anahtarı oluşturur.

$K_A = K_B$ olur, çünkü $n_A$ x $P_B = n_A$ x ($n_B$ x $G$) = $n_B$ x ($n_A$ x $G$) = $n_B$ x $P_A = K$

Çizelge 2.5'de yer alan algoritmalar farklı uzunluklardaki anahtar boyları ile eşdeğer güvenlik seviyesini sağlamaktadır. Çünkü bu kriptosistemler farklı matematiksel problemler üzerine kurulmuştur ve matematiksel problemlerin her birisi farklı karmaşıklık seviyesindedir. Eşdeğer güvenlik düzeyinin sağlanması için 256 bit uzunluğunda anahtar kullanan simetrik kriptosisteme karşın; faktörizasyon problemini kullanan RSA veya ayrık logaritma problemini kullanan DH için anahtar uzunluğunun bundan 60 kat ve eliptik eğri kriptosistemininse 2 kat daha uzun anahtar kullanması gerekmektedir. Öte yandan; yine simetrik anahtarlarda 256 bit ile sağlanan güvenliği sağlamak için, RSA-DH ikilisinin gereksinim duyduğu anahtar uzunluğu eliptik eğri tabanlı kriptosistemin 30 katıdır.

Çizelge 2.5 EEDH ile diğer algoritmaların anahtar uzunluklarının karşılaştırılması

| **Güvenlik (Bit)** | **Simetrik Şifreleme Algoritmaları** | **DH** | **RSA** | **EEDH** |
|---|---|---|---|---|
| 80 | Skipjack | 1024 | 1024 | 160 |
| 112 | 3DES | 2048 | 2048 | 224 |
| 128 | AES-128 | 3072 | 3072 | 256 |
| 192 | AES-192 | 7680 | 7680 | 384 |
| 256 | AES-256 | 15360 | 15360 | 512 |

## 2.7 Özetleme Fonksiyonları

Farklı uzunluklardaki veriyi işleyerek, sabit uzunlukta çıktı oluşturma işlemine, özetleme; bu işlemi gerçekleştiren algoritmalara da özetleme fonksiyonları denir. Özetleme fonksiyonları, kimlik doğrulama, bütünlük kontrolü, e-imza ve güvenli e-posta uygulamalarında kullanılmaktadır. Genellikle veri bütünlüğünü garanti etmesi, hızlı olması, sabit uzunlukta çıktı vermesi ve yüksek performanslı bir haberleşme sağlaması bu yaklaşımın üstünlükleridir.

Bir özetleme fonksiyonu, H, mesajın sayısal karşılığı olan M'yi kullanarak, sabit uzunluğa sahip mesaj özetini, $h = H(M)$ , oluşturur. İlk olarak mesaj, $m_1$, $m_2$, …$m_n$ olmak üzere eşit uzunlukta bloklara ayrılır. Gerektiği durumlarda son blok, belirli bir fonksiyon kullanılarak blok uzunluğuna tamamlanır.

Özetleme fonksiyonunun güvenli olarak kullanılabilmesi için sağlaması gereken temel özellikler şunlardır:

- Özetlenecek veri herhangi bir boyutta olabilir.
- Veri özeti sabit uzunluktadır.
- Elde edilecek özet değer tekil olmalıdır.
- Verilen herhangi bir veri için özetin hesaplanması kolay, özet değerinden veriyi elde etmek zor olmalıdır (Tek yönlü olmalıdır).
- Farklı veriden aynı özet değerinin üretilememesi gereklidir (Çakışmalara dayanıklı olmalıdır).

En çok bilinen hash fonksiyonları MD4, MD5 ve SHA ailesidir. MD4 ve MD5 günümüzde kullanılamaz hale gelmiştir. SHA-1 üzerinde yapılan kriptoanaliz çalışmaların SHA-1'in çakışmalara karşı zayıf kaldığını göstermiştir ve bu sebepten SHA-2 algoritmalarına geçiş önerilmeye başlanmıştır.

### 2.7.1 MD4

MD4, Ron Rivest tarafından 1990 yılında tasarımlanmış hash fonksiyonudur. MD4 veri girdisi için 128 bitlik hash değeri üretir.

MD4'te 512 bitlik veri blokları, tekrarlı yapı içinde her bloğu üç farklı döngüde işlemden geçirilir. Aynı hash değerine sahip iki veriyi bulmak yaklaşık $2^{64}$ uygulama gerektirir. Daha önceden belirlenmiş hash değerini sağlayan girdi bulmak ise $2^{128}$ uygulama gerektirir.

Ancak MD4 üretildikten kısa bir süre sonra kırılmış ve Rivest tarafından MD4 güçlendirilerek MD5 geliştirilmiştir.

### 2.7.2 MD5

MD5, MD4'ün güçlendirilmiş şeklidir. MD4 üç döngüden oluşurken MD5 dört döngüden oluşur, çalışma mekanizmaları benzerdir ve MD5 de MD4 gibi 128-bitlik hash değerleri üretir.

Günümüzde MD5 de kırılmış ve kullanılamaz hala gelmiştir.

### 2.7.3 SHA

SHA, NSA tarafından DSS için üretilmiştir. SHA'nın SHA-0, SHA-1 ve SHA-2 olmak üzere üç versiyonu vardır. 1993 yılında NIST tarafından SHA-0 yayınlanmıştır. 1995 yılında bir güvenlik eksikliğinden dolayı geri çekilen SHA-0'ın yerine SHA-1 yayınlanmıştır. SHA-1, SHA'nın kullanımı en yaygın olan sürümüdür.

#### 2.7.3.1 SHA-1

SHA-1, uzunluğu en fazla $2^{64}$ bit olan mesajları girdi olarak kullanır ve 160 bitlik mesaj özeti üretir. Bu işlem sırasında, ilk önce mesajı 512 bitlik bloklara ayırır ve gerekirse son bloğun uzunluğunu 512 bite tamamlar. SHA-1 çalışma prensibi olarak R. Rivest tarafından tasarlanan MD5 özetleme fonksiyonuna benzer ve iteratif bir yapısı vardır. Her iterasyonda bir sıkıştırma fonksiyonu kullanır. Bu fonksiyon, mesajın 512 bitlik bloğunu alır ve 16 tane 32 bitlik kelimeye ( $m_0$, $m_1$,…$m_{15}$ ) çevirir. Daha sonra bu kelimeler,

$m_i = (m_{i-3} + m_{i-8} + m_{i-14} + m_{i-16}) << 1$ fonksiyonu kullanılarak 2560 bite genişletilir ve her biri 20 matematiksel fonksiyon içeren 4 tur çalıştırılarak 160 bitlik mesaj özeti elde edilir.

Özet algoritmalarını kullanışlı kılan özelliklerden biri de veri özetlerinin farklı olmasıdır. Veri özeti aynı olan iki farklı verinin bulunması durumuna çakışma adı verilmektedir. $2^{n/2}$ lik karmaşıklıktan daha az işlemle çakışma bulan bir algoritma özet fonksiyonunu kırmış sayılır (“n”, veri özeti bit sayısını ifade ediyor).

SHA-1 algoritmasına 2005 yılında $2^{80}$ işlemden daha az işlem ile çakışma oluşturuldu. Daha sonra bir grup araştırmacı tarafından $2^{69}$ işlemden daha az işlem ile çakışma oluşturan saldırılar duyuruldu. 2005 yılı sonlarında düzenlenen CRYPTO konferansında çakışma oluşturmak için gerekli saldırı sayısının $2^{63}$ işleme düşürüldüğü belirlendi. Genellikle elektronik imzalarda kullanılan bir algoritmanın bu seviye çakışma vermesi büyük bir güvenlik açığı olarak kabul edilebilir. Örnekleyecek olursak, 160 bitlik çıktı üreten algoritmanın çakışmalara karşı 80 bitlik güvenlik sağlaması gerekir. Ancak son dönemlerde SHA-1 algoritması için bu sayı 39 bite kadar düşürülmüştür. Yaşanan bu gelişmelerden sonra, 2010 yılından itibaren SHA-2 algoritmasına geçilmesi önerilmeye başlanmıştır.

### 2.7.3.2 SHA-2

SHA-1 algoritmasının verdiği kırılma tedirginliği ile SHA-2 algoritmaları geliştirilmiştir. SHA özet fonksiyon kapsamında veri özeti uzunluğu daha uzun olan ve aralarında ufak farklar içeren ve SHA-2 ailesi olarak adlandırılan dört farklı algoritma (SHA-224, SHA-256, SHA-384, SHA-512) bulunur. SHA-0 ve SHA-1 için geliştirilmiş ataklar SHA-2 versiyonları için herhangi bir zayıflık belirtmemiştir. Ancak SHA-2 algoritmasının da kırılma ihtimaline karşı araştırmacılar yeni bir algoritma tasarlamaya karar vermişlerdir ve SHA-3 algoritması gündeme gelmiştir.

Kriptolojinin temelinde olduğu gibi elektronik imzada da algoritmaların hızlı çalışması önemlidir. Bunun için de verinin kendisi değil özeti imzalanır. Elektronik imza uygulamalarında SHA-1 algoritmasının kullanılması önerilmektedir. Ancak 2005 yılında, SHA-1 ile ilgili zayıflıkların yakalanması SHA-2 algoritmalarının geliştirilmesine zemin hazırlamıştır ve 2010 yılından itibaren SHA-2'ye geçilmesi önerilmeye başlanmıştır.

SHA-1 ve SHA-2 ailesine ait algoritmaların bir kısım özellikleri çizelge 2.6'da verilmiştir.

Çizelge 2.6 SHA Algoritmalarının karşılaştırması (Stallings 2011)

| | **SHA-1** | **SHA-256** | **SHA-384** | **SHA-512** |
|---|---|---|---|---|
| Mesaj Özet Boyutu | 160 | 256 | 384 | 512 |
| Mesaj Boyutu | $< 2^{64}$ | $< 2^{64}$ | $< 2^{128}$ | $< 2^{128}$ |
| Blok Boyutu | 512 | 512 | 1024 | 1024 |
| Kelime Boyutu | 32 | 32 | 64 | 64 |
| Adım Sayısı | 80 | 64 | 80 | 80 |
| Güvenlik | 80 | 128 | 192 | 256 |

Not: 1. Bütün veriler bit cinsindendir.
2. Güvenlik, doğum günü atağına karşı sağladığı güvenliği ifade etmektedir. Doğum günü atağına göre *n* boyutundaki bir mesaj özeti için mesajların çakışma ihtimali $2^{n/2}$ olur.

## 2.8 Sayısal İmza

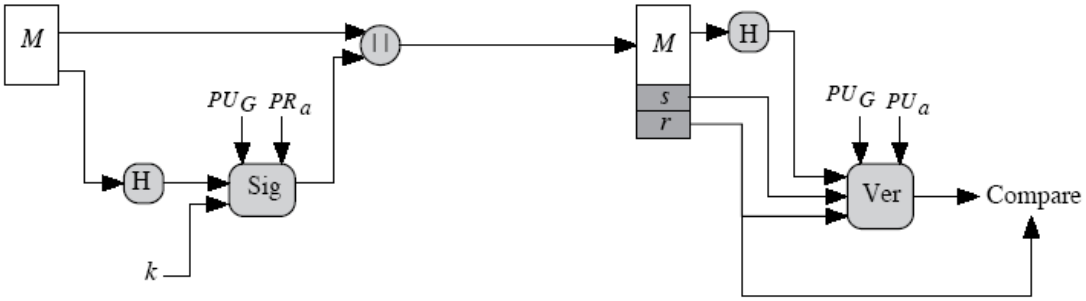
Sayısal imza elektronik ortamda bir mesajın imzalanması işlemi olup geleneksel imzanın elektronik ortamdaki karşılığı olarak düşünülebilir. Sayısal imza elektronik ortamdaki bir mesaja bazı algoritmalar ile eklenir. Sayısal imza imzalanan metne göre değişiklik gösterir. Sayısal imza üç temel özelliğe sahiptir.

Bunlar:

i. Veri Bütünlüğü: Verinin izinsiz değiştirilmesinin önlenmesi.
ii. Kimlik Doğrulama: Veriyi alan ve gönderen tarafların kimliklerinin doğrulanması.
iii. İnkâr Edilemezlik: Bireylerin elektronik ortamda gerçekleştikleri işlemleri inkâr etmelerini önlenmesi.

Sayısal imza; imzanın oluşturulması ve imzanın doğrulanması olarak iki kısımdan oluşur. Bir *A* kullanıcısı bir *m* mesajının sayısal imzasını elde etmek için özet fonksiyonları yardımı ile mesajın özetini kendi gizli anahtarı ile şifreler ve mesaja ekleyerek mesajı gönderir (gerekirse mesajı da şifreler ancak performans açısından tavsiye edilmez). Mesajı alan *B* kullanıcısı (eğer mesaj şifrelenmişse mesajı deşifreler) mesajın içindeki sayısal imzayı *A*'nın açık anahtarı ile deşifreleyerek özet mesajı elde eder ve orijinal mesaja *A*'nın uyguladığı özet fonksiyonu uygulayarak bir özet elde eder. İmzanın içindeki özet ile *B*'nin bulduğu özet aynı ise mesaj göndericisinin doğruluğu ve mesajın gönderilme esnasında değişmediği anlamına gelir.

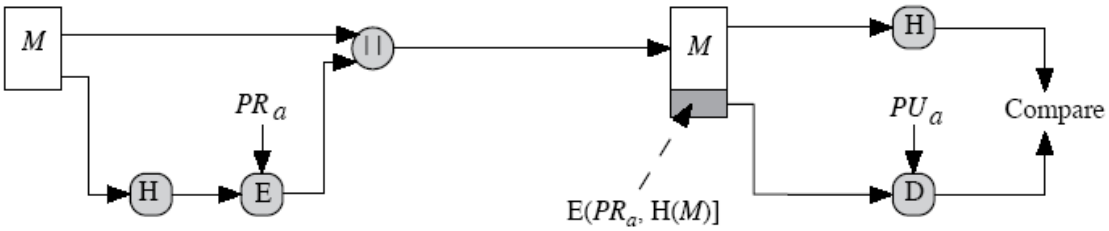
### 2.8.1 DSA sayısal imza sistemi




Şekil 2.23 DSA sayısal imza sistemi (Stallings 2011)

M: İmzalanacak mesaj
H: Özetleme fonksiyonu
Sig: DSA imzalama işlemi
$PR_a$: Gönderenin gizli anahtarı
Ver: DSA doğrulama İşlemi
$PU_a$: Gönderenin açık anahtarı
s, r: Gönderenin mesaj özeti üzerindeki imzası
$PU_G$:Global açık anahtar bileşenleri

### 2.8.2 RSA sayısal imza sistemi




Şekil 2.24 RSA sayısal imza sistemi (Stallings 2011)

M: İmzalanacak mesaj
H: Özetleme fonksiyonu
E: RSA gizli anahtar ile imzalama işlemi
$PR_a$: Gönderenin gizli anahtarı
D: RSA açık anahtar ile doğrulama işlemi
$PU_a$: Gönderenin açık anahtarı

# 3. KAYNAK ÖZETLERİ

Yapılan literatür araştırmaları sonucunda gizli anahtarın eliptik eğri Diffie-Hellman yöntemi ile taraflar arasında ortak bir şekilde oluşturulması ve gizliliğin yanı sıra elektronik imza ile kimlik denetimi, bütünlük, inkar edememe gibi güvenlik unsurlarını da sağlamaya yönelik bir çalışmaya rastlanılmamış olup güvenli veri iletimine yönelik bazı çalışmalara mevcuttur. Söz konusu çalışmalarda önerilen yöntemlerin bazıları aşağıda özetlenmiştir.

Mantoro ve Zakariya (2012) yaptıkları çalışmada, mobil cihazlar kullanılırken karşılaşılabilecek güvenlik zafiyetlerinden bahsetmiş bunlara yönelik bir yöntem önermişlerdir. Android işletim sistemi için geliştirdikleri uygulamada simetrik ve asimetrik yöntemleri bir arada kullanan melez kriptosistem üzerinde çalışmışlar, şifreleme ve deşifreleme işlemlerini simetrik kriptosistem ile şifrelemede kullanılacak olan gizli anahtarı veriyi gönderen kişi alıcının açık anahtarını kullanarak asimetrik kriptosistem ile şifrelemekte ve şifreli veri ile beraber karşı tarafa iletmektedir. Bu noktada şifrelemede kullanılacak anahtarın açık anahtarlı şifreleme yöntemi ile şifrelenmesi ve tekrar aynı şekilde deşifre edilmesi performans zafiyetine sebep olmaktadır.

Gupta ve Sharma (2012) yaptıkları çalışmada, RSA ve Diffie-Hellman yöntemlerinin bir arada kullanıldığı hibrid kriptosistem üzerinde durmuşlardır. Önerilen hibrid yöntemde kullanıcı RSA ile açık ve gizli anahtar çiftini üretiyor ve bu anahtarları Diffie- Hellman parametreleri olarak kullanıyor daha sonra anahtar ile veriyi XOR işleminden geçirerek şifreleme işlemini gerçekleştiriyor.

Rasool, Sridhar ve Kumar (2011) yaptıkları çalışmada, internet üzerinden veri transferinde güvenlik zafiyetine karşı bir mekanizma önermişlerdir. Önerdikleri mekanizmada simetrik ve asimetrik yöntemleri birlikte kullanmışlardır. Şifreleme ve deşifreleme işlemlerinde performansı sebebiyle AES'i tercih etmişlerdir. Burada kullanılacak gizli anahtarın iletişim yapan diğer kullanıcıya iletilmesi aşamasında açık anahtarlı kriptosistemden yararlanmışlardır.

Ren ve Miao (2010) yaptıkları çalışmada, DES ve RSA yöntemlerinin bir arada kullanıldığı melez kriptosistem üzerinde durmuşlardır. Önerilen melez yöntemde veri 64 bit anahtar kullanılarak DES algoritması ile şifreleniyor ve şifrelemede kullanılan anahtar veriyi alacak olan kişinin açık anahtarıyla RSA algoritması kullanılarak şifreleniyor. Son olarak RSA ile şifrelenmiş oturum anahtarı ve DES ile şifrelenmiş veri karşı tarafa iletiliyor. Şifreli veriyi ve şifreli anahtarı alan taraf, anahtar üzerindeki şifreyi kendi RSA gizli anahtarı ile deşifreliyor ve elde ettiği anahtarı DES algoritmasında kullanarak şifreli veriyi deşifreliyor.

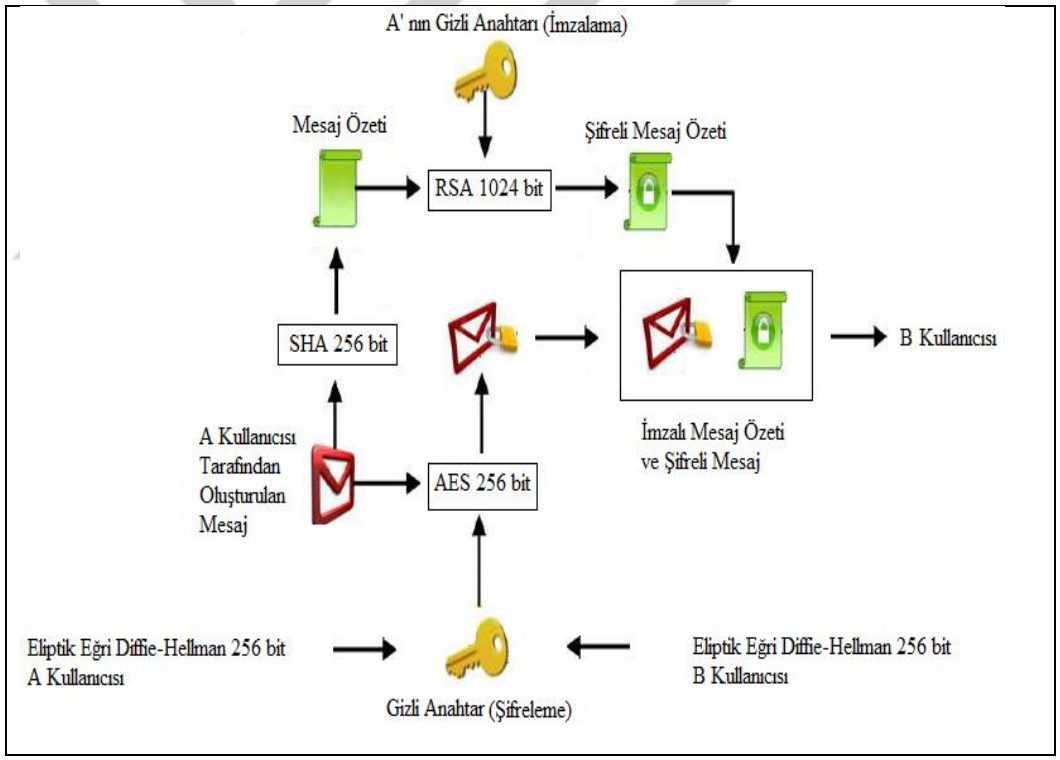
Yapılan çalışmalar birtakım güvenlik problemlerine çözüm üretmesine rağmen; kullanılan tekniklerin günümüzde yeterli güvenliği sağlayamaması, verinin içeriğinin değiştirilmesi, verinin kim tarafından gönderildiğinin bilinememesi, veriyi gönderenin inkâr edebilmesi gibi bazı yönlerden halen güvenlik zafiyetleri yaşatabilmektedir.

Ayrıca şifreleme ve deşifreleme işlemlerinde kullanılacak olan anahtarın RSA algoritması gibi yüksek anahtar boyutu gerektiren bir algoritma ile şifrelenip veriyi alacak tarafa gönderilmesi ve veriyi alan tarafın da şifreli anahtarı aynı şekilde deşifrelemesi mobil cihazlar gibi işlem kapasitesi daha düşük olabilen aygıtlarda performans problemi teşkil edebilmektedir.

# 4. ÖNERİLEN YÖNTEM VE GELİŞTİRİLEN UYGULAMA

## 4.1 Önerilen Yöntem

Bu tez çalışmasında, şifreleme ve deşifreleme işlemlerinde kullanılacak olan gizli anahtarın Diffie-Hellman yöntemi ile taraflar arasında ortak oluşturulması ve bu aşamada eliptik eğri kullanılarak daha düşük anahtar boyutları ile aynı güvenlik seviyesine ulaşılması böylece önerilen yöntemin performansa önemli derecede katkı sağlayacağı düşünülmektedir. Aynı zamanda önerilen yönteme elektronik imza entegrasyonu ile gizliliğin yanı sıra bütünlük, inkâr edememe, kimlik doğrulama gibi bilgi güvenliği unsurlarının da sağlanması düşünülmektedir.




Şekil 4.1 Önerilen yöntemin mimarisi

- Önerilen yöntemde iletişimi gerçekleştirecek olan *A* ve *B* kullanıcıları eliptik eğri Diffie-Hellman yöntemi ile 256 bit anahtarı ortak bir şekilde üretirler,
- Kullanıcı *A* tarafından oluşturulan verinin 256 bit SHA ile özeti oluşturulur,

- Oluşturulan özet kullanıcı *A*'nın 1024 bit RSA gizli anahtarı ile imzalanır ve imzalı veri özeti elde edilir,
- Kullanıcılar tarafından eliptik eğri Diffie-Hellman yöntemi ile ortak bir şekilde üretilen 256 bit anahtar ve AES algoritması ile oluşturulan veri şifrelenir,
- Şifrelenen veri ve imzalı veri özeti kullanıcı *B*'ye iletilir,
- Kullanıcı *B* kendisine gelen şifreli veriyi, eliptik eğri Diffie-Hellman yöntemini kullanarak kullanıcı *A* ile ortak bir şekilde oluşturduğu 256 bit anahtar ve AES algoritması ile deşifreler ve açık veriyi elde eder,
- Kullanıcı *B* elde ettiği açık veriden 256 bit SHA kullanarak gelen verinin özetini çıkarır,
- Kullanıcı *B* açık verinin gerçekten kullanıcı *A*'dan geldiğini ve verinin kullanıcı *A*'dan çıktıktan sonra içeriğinin değiştirilmediğinden emin olmak için kendisine gelen imzalı veri özetini kullanıcı *A*'nın açık anahtarı ile deşifreler ve veri özetini elde eder,
- Kullanıcı *B* son iki adımda elde ettiği veri özetlerini karşılaştırır. Eğer veri özetleri eşleşiyorsa veri gerçekten kullanıcı *A*'dan gönderilmiştir ve hiçbir şekilde değişmemiştir.

Literatürde şifrelemede kullanılacak anahtarın Diffie-Hellman yöntemi ile taraflar arasında ortak bir şekilde oluşturulması bu işlemin eliptik eğri kullanılarak gerçekleştirilmesi, oluşturulan anahtarın AES algoritması tarafından şifreleme-deşifreleme işlemlerinde kullanılması ve elektronik imza entegrasyonu yapılarak güvenlik unsurlarından gizliliğin yanı sıra bütünlük, inkâr edememe, kimlik doğrulama gibi unsurların sağlanması yönünde bir çalışma görülmemiştir ayrıca böyle bir yöntemin mobil uygulamalarda da verimli bir şekilde çalışabilmesi önerilen yöntemin çok yönlülüğünü ortaya koymaktadır.

## 4.2 Geliştirilen Uygulama

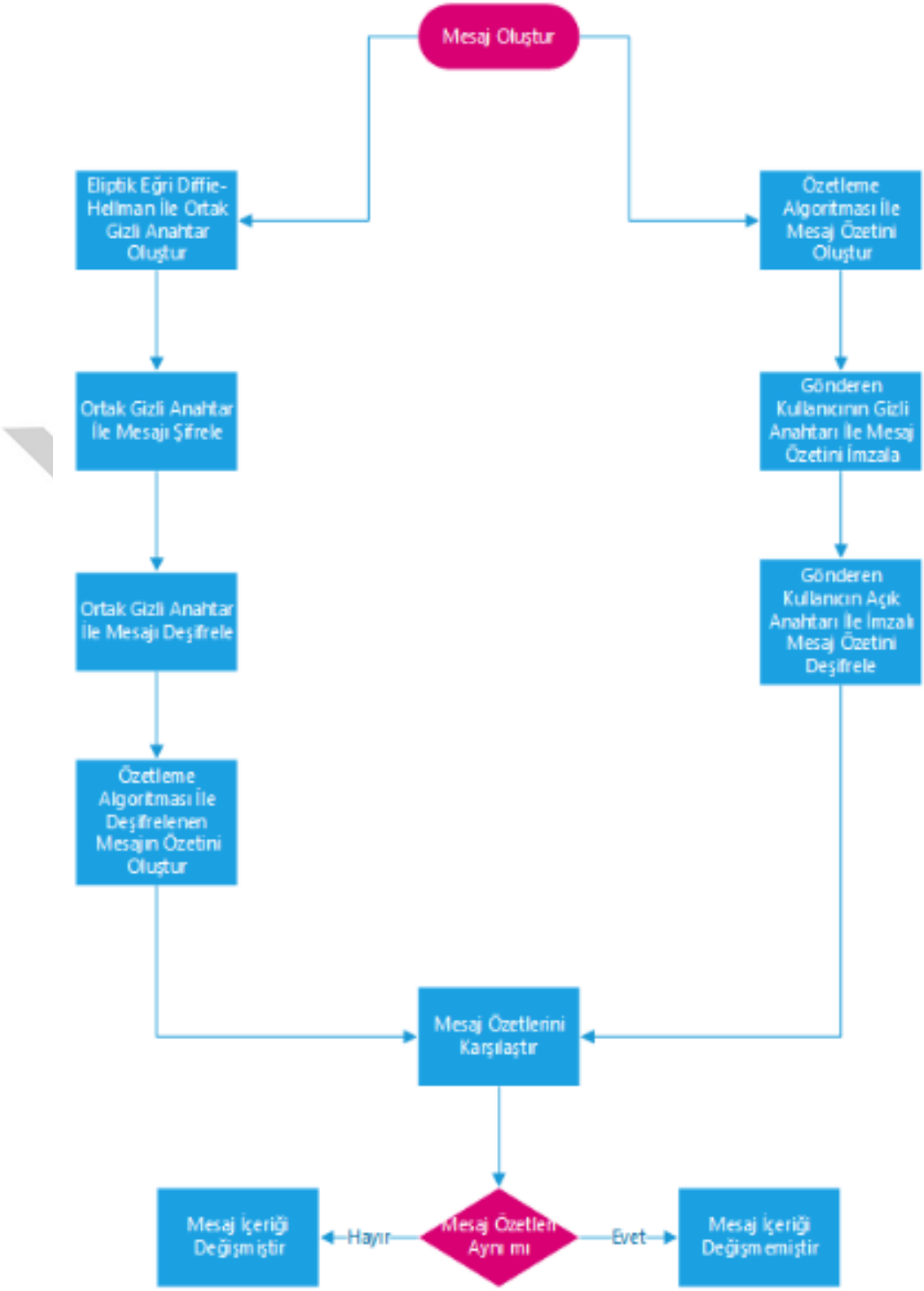
Mobil cihazlar (telefon, tablet) gibi işlem kapasitesi daha düşük olan ortamlarda uygulamaların performansı önemli bir konudur. Güvenlik önlemlerinin sonucu olarak yaşanabilecek performans problemlerine karşı etkili çözüm getirmek çok önemlidir.

Simetrik Anahtarlı Sistemler, şifreleme ve deşifreleme işlemlerinde küçük anahtar boyutu ve algoritmaların matematiksel işlemler yerine yer değiştirme işlemlerini kullanması sebebiyle asimetrik sistemlere göre çok daha hızlı çalışırlar. Ancak simetrik anahtarlı sistemlerde şifreleme ve deşifreleme işlemleri aynı anahtarla yapıldığı için bu anahtarın güvenli dağıtımı bir problem olarak karşımıza çıkmaktadır. Bu noktada Diffie-Hellman anahtar değişim sistemi ile simetrik anahtarlı sistemlerde şifreleme ve deşifreleme işlemleri için kullanılacak olan gizli anahtar bağımsız olarak taraflar arasında üretilir.

Yapılan çalışmada simetrik anahtarlı algoritma için AES, AES'te kullanılacak gizli anahtarı tarafların birlikte üretmesi için Diffie-Hellman anahtar değişimi kullanılmış ve bu işlemi daha yüksek hız ile gerçekleştirmek için eliptik eğrilerden faydalanılmıştır. Ayrıca geliştirilen uygulamaya elektronik imza entegrasyonu yapılarak bilgi güvenliği servislerinden gizliliğin yanı sıra, bütünlük, inkâr edememe ve kimlik doğrulama servislerini de kapsaması sağlanmıştır.

Böylece iletişimde kullanılacak olan ağa yönelik dinleme saldırıları yapılsa bile iletişim ağ üzerinden şifreli bir şekilde yapılacağı için saldırganlar şifreli iletişimi dinleseler de önerilen yöntemdeki yüksek güvenlik sebebiyle veri üzerindeki şifreyi çözemeyecekler ve taraflar arasında güvenli bir iletişim gerçekleşmiş olacaktır.

Ayrıca, geliştirilen uygulama sadece önerilen yöntemi gerçeklemek için düşünülmüş olup, önerilen yöntem her türlü veri iletişim ortamında ve uygulamada çalışabilecek verimliliğe ve güvenlik seviyesine sahiptir. Bunun yanı sıra karşılaşılabilecek ağ dinleme saldırılarına karşı da güvenli bir mekanizma ortaya koymaktadır.




Şekil 4.2 Geliştirilen yöntemin akış şeması

Bu tez çalışmasında, yazılım geliştirme aracı olarak Visual Studio platformu ve programlama dili olarak C# kullanılmıştır.

Geliştirilen uygulamaya ANTI SNIFFING adı verilmiştir.

Geliştirilen uygulamanın altyapısı şu şekildedir:

### 4.2.1 Uygulama platformu

Önerilen yöntemi geliştirmek için kullanılan PC platformu ve yazılım araçları aşağıdaki gibidir:

- Windows 8.1 Enterprise
- Intel Core i7
- 8 GB Ram
- Visual Studio – C#
- SQL Server

Önerilen yöntem sadece PC uygulamaları için değil aynı zamanda mobil uygulamalar için de çözüm sunmaktadır. Bu kapsamda, önerilen yöntem için geliştirilen uygulama Windows Phone işletim sistemi üzerinde yapılmıştır.

Windows Phone işletim sistemine yazılım geliştirmek için kullanılan emülatörün özellikleri aşağıdaki gibidir:

- Windows Phone 8.0
- 512 MB Ram

### 4.2.2 Veritabanı mimarisi

Geliştirilen uygulama için aşağıdaki tablolar kullanılmıştır.

**a. Kullanıcı tablosu**

Kullanıcı tablosunda kullanıcının adı ve RSA algoritması ile imzalanan veriyi deşifrelemek için kullanılacak olan RSA açık anahtarı tutulmaktadır.

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | ID | int | ☐ |
| | Ad | nvarchar(50) | ☑ |
| | RSAPublicKey | nvarchar(MAX) | ☑ |

| | ID | Ad | RSAPublicKey |
|---|---|---|---|
| 1 | 1 | Ertan | <RSAKeyValue><Modulus>vE3MSPh1zEQhX27j2mt6LyBAc8... |

**b. Açık veri tablosu**

Açık veri tablosunda gönderen kullanıcı ve veri tutulmaktadır.

| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | Id | int | ☐ |
| | Gonderen | int | ☑ |
| | Veri | nvarchar(500) | ☑ |

| | Id | Gonderen | Veri |
|---|---|---|---|
| 1 | 1 | 1 | Hava çok soguk |

**c. Şifreli veri tablosu**

Şifreli veri tablosunda; imzalı veri özeti, şifrelenmiş veri, gönderen kullanıcı, AES algoritması için kullanılan başlangıç vektörü (IV) bilgileri tutulmaktadır.

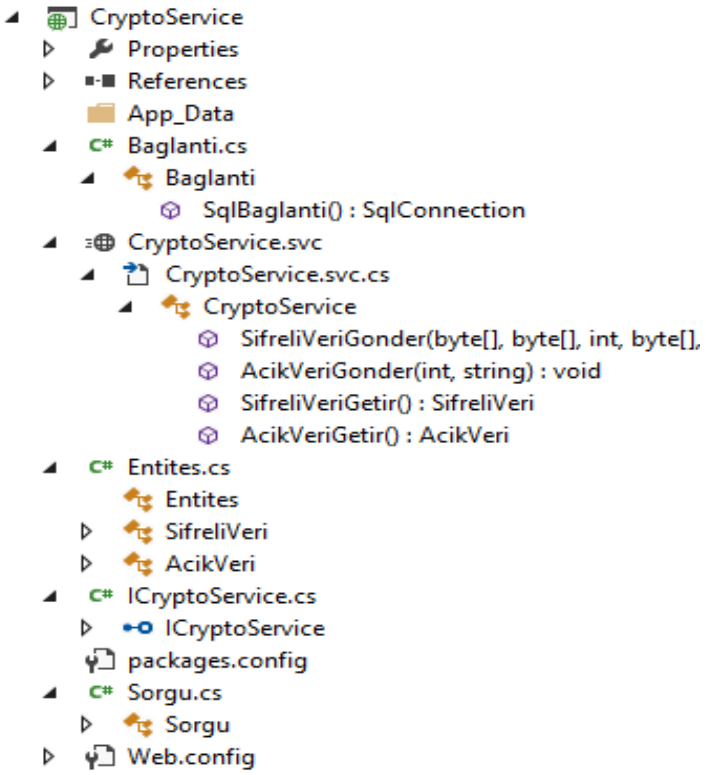
| | Column Name | Data Type | Allow Nulls |
|---|---|---|---|
| 🔑 | ID | int | ☐ |
| | SignedHash | varbinary(MAX) | ☑ |
| | EncryptedData | varbinary(MAX) | ☑ |
| | Gonderen | int | ☑ |
| | AesIV | varbinary(MAX) | ☑ |

| | ID | SignedHash | EncryptedData | Gonderen | AesIV |
|---|---|---|---|---|---|
| 1 | 1 | 0x87572065371O2DE5C143E20929ED1E647B21A4FCD80184... | 0x0D6E9A78F5DD42E741DCE7CAA279AF4F0A5A730A64BCF7... | 1 | 0x2B914DCD25FAEF76CB6E6712BC29998F |

### 4.2.3 Web servis mimarisi

Bu bölümde gönderilen veri web servis aracılığıyla veritabanına kaydedilir ve alması gereken kişiye iletilir.

- Veritabanı bağlantısı için “Baglanti”
- Veritabanı tablo kolonlarının yazılım tarafındaki karşılıkları için “Entities”
- Gönderilen ve alınan verilerin taraflara iletimi için “Sorgu” olmak üzere 3 adet Class kullanılmıştır.
- Web Servis üzerinde kullanılacak metotların yönetimi için “ICryptoService” Interface’i kullanılmıştır.

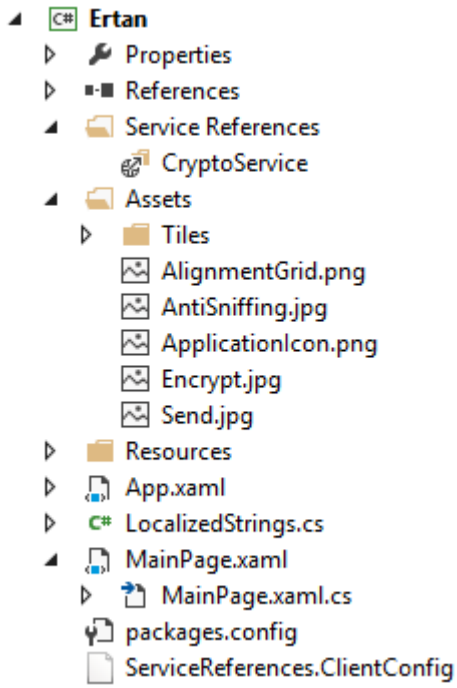



Şekil 4.3 ANTI SNIFFING mimari

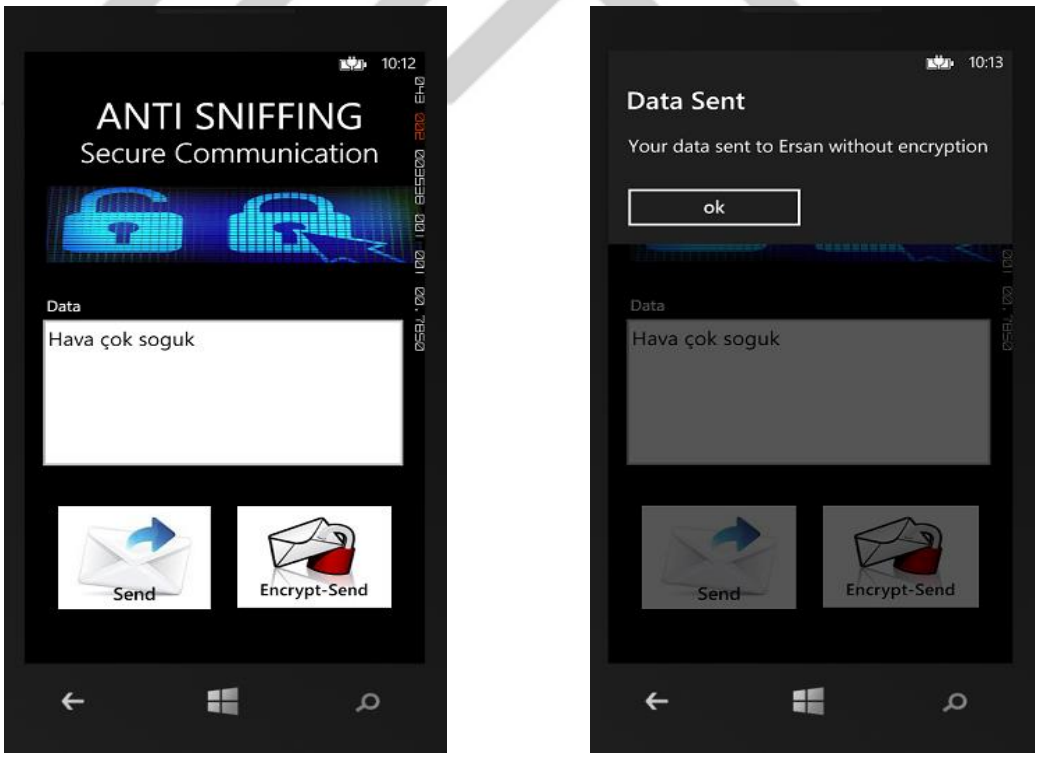
### 4.2.4 İletişim adımları

**İletişimi başlatan kullanıcı**

İletişimi başlatan kullanıcı veriyi açık veya şifreli bir şekilde karşı tarafa iletebilir. Eğer şifreli bir şekilde gönderiyorsa verinin özetine imzasını da ekler ve karşı tarafın verinin kimden geldiğine emin olmasını sağlar.

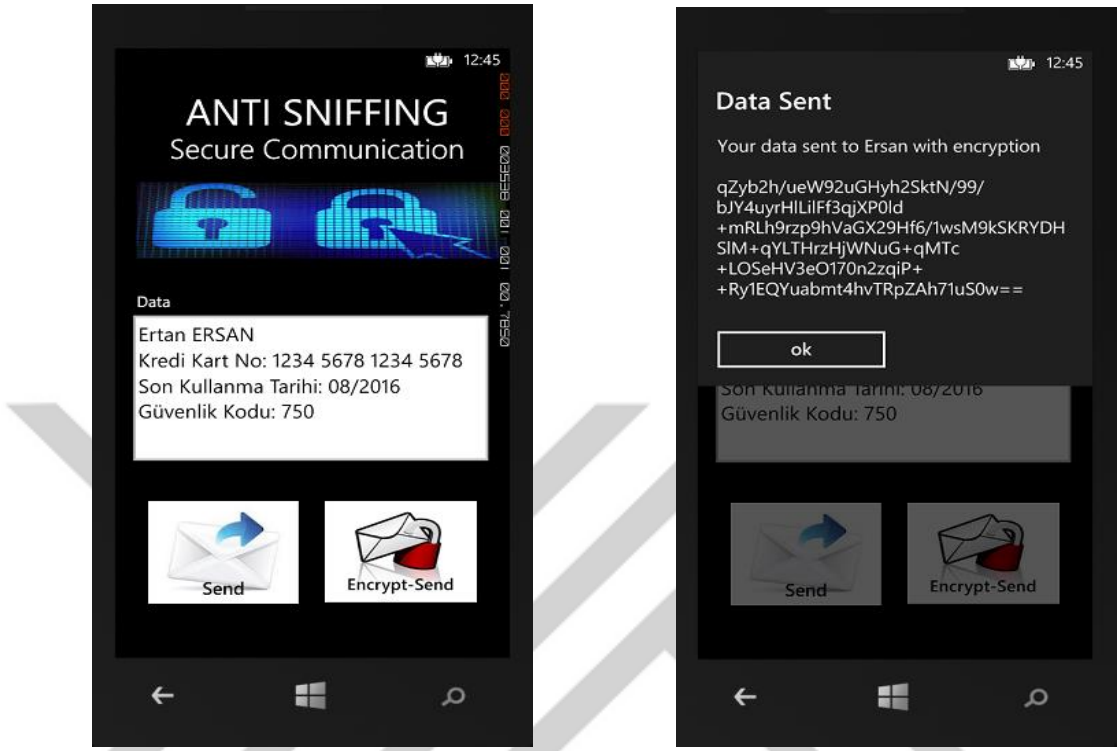



Şekil 4.4 İletişimi başlatan kullanıcı mimari




Şekil 4.5 ANTI SNIFFING şifresiz veri gönderme ekranları

ANTI SNIFFING uygulaması ile iletişimi başlatan taraf başkalarının görmesinin sorun oluşturmayacağı veriyi şifresiz bir şekilde karşı tarafa gönderebilir.

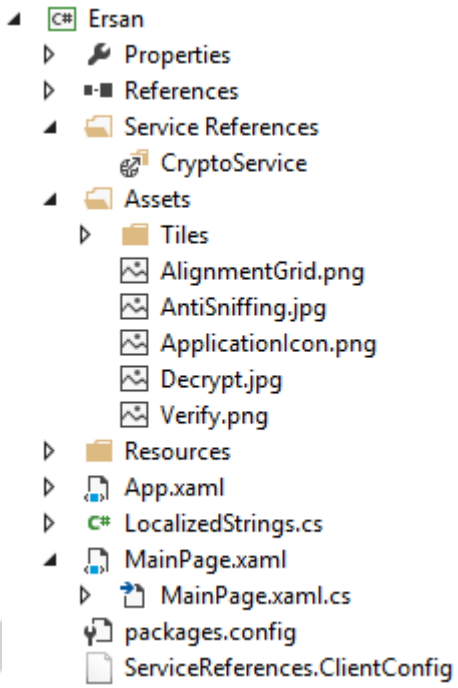



Şekil 4.6 ANTI SNIFFING şifreli veri gönderme ekranları

ANTI SNIFFING uygulaması ile iletişimi başlatan kullanıcı gizliliği önemli olan veriyi şifreli ve imzalı bir şekilde karşı tarafa iletebilir.

**İletişimi sonlandıran kullanıcı**

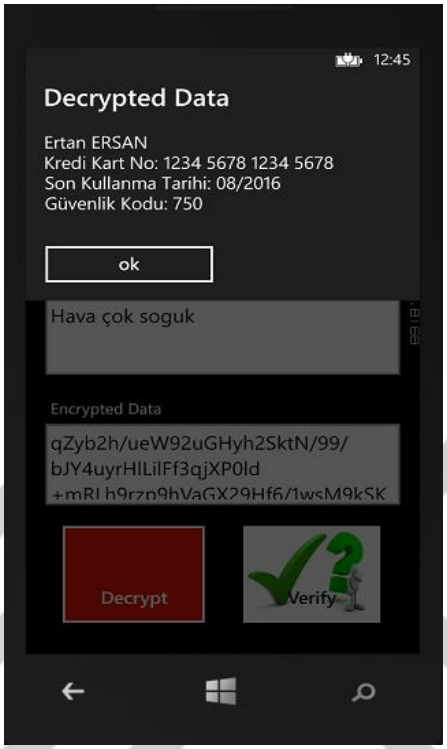
İletişimi başlatan kullanıcı tarafından açık veya şifreli bir şekilde gönderilen veri, iletişimi sonlandıran kullanıcı tarafından; veri açık bir şekilde gönderildiyse direk, şifreli bir şekilde gönderildiyse deşifre edilerek görülür. Ayrıca veri şifreli gönderildiyse gönderen kullanıcının imzasını doğrulayarak gerçekten o kullanıcı tarafından gönderildiğinden emin olur.




Şekil 4.7 İletişimi sonlandıran kullanıcı mimari

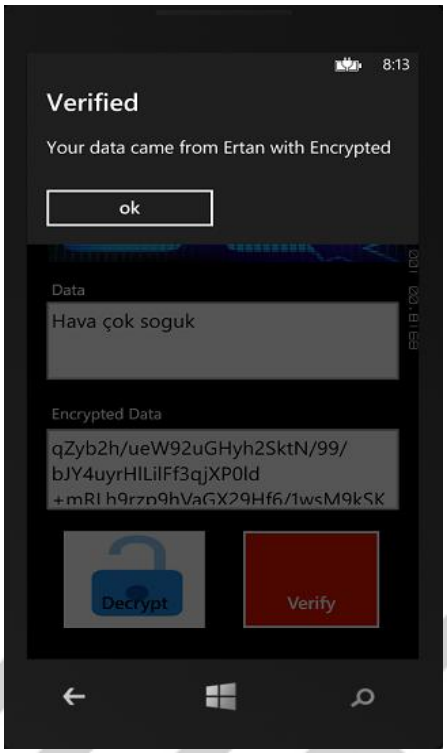

Şekil 4.8 ANTI SNIFFING gelen veri ekranı (Şifreli ve Şifresiz)

ANTI SNIFFING uygulaması ile iletişimi sonlandıran kullanıcı kendisine gelen şifreli veya şifresiz veriyi şekil 4.8'deki gibi görebilir.




Şekil 4.9 ANTI SNIFFING veri deşifreleme ekranı

Eğer veri şifreli iletildiyse veri üzerindeki şifreyi şekil 4,9'daki gibi çözer ve veri üzerinde gerekli doğrulama işlemlerini şekil 4.10'daki gibi gerçekleştirerek veriyi gönderen kişiden ve verinin iletim esnasında değişmeden ulaştığından emin olur.

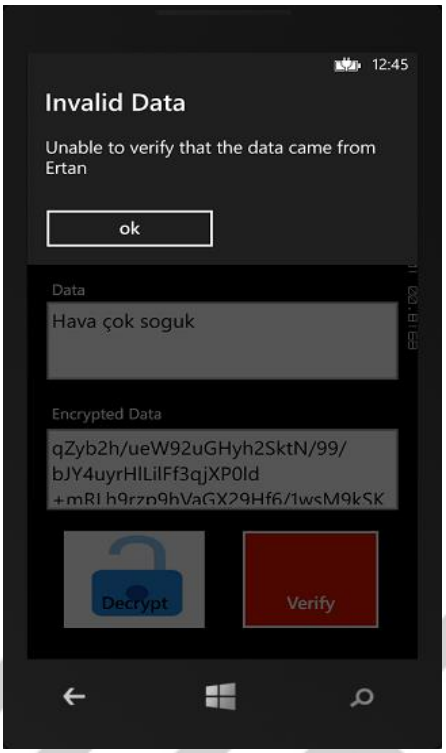



Şekil 4.10 ANTI SNIFFING veri doğrulama ekranı

İmzalanmış veri özeti üzerinde aşağıdaki gibi bir karakterinde bile değişiklik olduğu zaman veya veri özetini imzalayan RSA gizli anahtarıyla ilişkili olmayan farklı bir RSA açık anahtarıyla doğrulama işlemi yapılmak istendiği zaman şekil 4.11'deki gibi veri doğrulama işlemi yapılamaz.

```
SignedHash=0x8757206537102DE5C143E20929ED1E647B21A4FCD80184BCD3CFCADD65458F562...

update SifreliVeri set
SignedHash=0x9757206537102DE5C143E20929ED1E647B21A4FCD80184BCD3CFCADD65458F562...
```

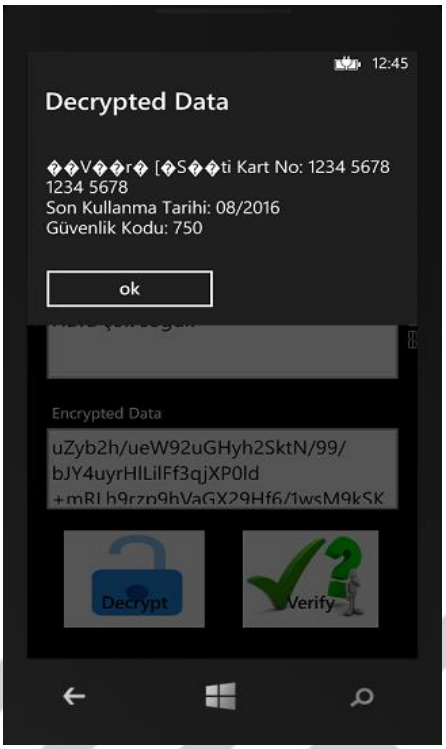



Şekil 4.11 ANTI SNIFFING veri doğrulayamama ekranı

Şifreli veri üzerinde aşağıdaki gibi bir karakterinde bile değişiklik olduğu zaman veri deşifreleme işlemi şekil 4.12'deki gibi doğru bir şekilde gerçekleştirilemez ve aynı zamanda veri doğrulama işlemi yapılamaz.

```
EncryptedData=0xA99C9BDA1FEE796F76B861F28764A4B4DFFDF7F6C9638BB2AC794B8A515FDE...

update SifreliVeri set
EncryptedData=0xB99C9BDA1FEE796F76B861F28764A4B4DFFDF7F6C9638BB2AC794B8A515FDE...
```




Şekil 4.12 ANTI SNIFFING içeriği değiştirilmiş veri deşifreleme ekranı

## 4.3 Bulgular

Yapılan çalışmada, AES algoritmasında kullanılacak olan gizli anahtarın dağıtımındaki güvenlik zafiyetine karşı önerilen yöntemlere alternatif olarak Diffie-Hellman yöntemi kullanılmıştır. Bu noktadaki avantaj, RSA algoritması ile şifreleme-deşifreleme işlemlerinde kullanılacak olan anahtarın tekrar şifrelenip, deşifrelenmesi gereksiniminin ortadan kaldırılmasıdır. Burada RSA gibi açık anahtarlı şifreleme sisteminin kullanılmasının ortadan kaldırılmasıyla AES'te kullanılacak olan 256 bit anahtarın şifrelenmesinde, kullanılan RSA anahtar uzunluğuna göre çizelge 4.1 ve 4.2'de belirtilen işlem maliyetleri ortadan kalkmaktadır. Buradaki önemli nokta RSA'de kullanılan anahtar uzunlukları artıkça işlem sürelerinin de katlanarak artmasıdır.

Çizelge 4.1 RSA çalışma süreleri (Milisaniye)

| **İşlem** | **128 bit** | **1024 bit** |
|---|---|---|
| Gizli Anahtar Oluşturma | 66.8 | 200.6 |
| Açık Anahtar Oluşturma | 66.6 | 199.5 |
| Şifreleme | 111.2 | 332.3 |
| Deşifreleme | 112.1 | 335.2 |

Gerekli olan bellek miktarları incelendiğinde ise RSA'de kullanılan anahtar uzunluğu arttıkça sisteme getirdiği yük de sürekli olarak artmaktadır. Bu da bir noktada sistemlerin durmasına sebep olabilmektedir.

Çizelge 4.2 RSA gerekli bellek boyutu (Kb)

| **İşlem** | **128 bit** | **1024 bit** |
|---|---|---|
| Gizli Anahtar Oluşturma | 283.3 | 850.4 |
| Açık Anahtar Oluşturma | 283.3 | 850.4 |
| Şifreleme | 580.1 | 1520.6 |
| Deşifreleme | 585.0 | 1535.3 |

Çizelge 4.3'te görüldüğü gibi Diffie-Hellman yönteminde de büyük anahtar boyutlarına gereksinim olması sebebiyle bu noktada da bir performans zafiyeti karşımıza çıkmaktaydı. Yapılan çalışmada Diffie-Hellman yöntemi eliptik eğri ile kullanılarak aynı güvenlik gereksinimini daha düşük anahtar boyutlarıyla elde edilebildiği görüldü. Bu noktada da önemli derecede işlem süresi kısaltılmış oldu.

Çizelge 4.3 Diffie-Hellman ve Eliptik Eğri Diffie-Hellman anahtar boyutları

| **Simetrik Şifreleme Algoritmaları ile Sağlanan Güvenlik (Bit)** | **Diffie-Hellman** | **Eliptik Eğri Diffie-Hellman** |
|---|---|---|
| 80 | 1024 | 160 |
| 112 | 2048 | 224 |
| 128 | 3072 | 256 |
| 192 | 7680 | 384 |
| 256 | 15360 | 512 |

160 bit Eliptik Eğri Diffie-Hellman / 1024 bit Diffie-Hellman güvenlik seviyesi için platforma ve optimizasyon seviyesine göre değişebilmekle birlikte Eliptik Eğri Diffie-Hellman, Diffie-Hellman'a göre 5-15 kat daha hızlıdır. Bu fark 256 bit Eliptik Eğri Diffie-Hellman / 3072 bit Diffie-Hellman seviyesine gelindiğinde 20-60 kata çıkar. 256 bit AES anahtarını güvenlik zafiyeti vermeden taşıyabilmek için gerekli olan 512 bit Eliptik Eğri Diffie-Hellman / 15360 bit Diffie-Hellman güvenlik seviyesinde ise bu fark ortalama 400 kattır. İşlem süresi, bellek tüketimi gibi parametrelerin hesaplanması kullanılan cihazın özelliklerine göre değişebilmektedir.

# 5. SONUÇ VE ÖNERİLER

Hayatın her alanında bilgisayarların yaygın kullanımı ve özellikle bilgisayar ağlarının gelişmesi, dünyanın dört bir yanındaki bilgiye erişmeyi kolaylaştırırken, güvenlik sorunlarını beraberinde getirmektedir. Bilginin korunması ve güvenli bir şekilde iletilmesi çok önemli bir konu haline gelmiştir.

Günümüzde hem kurumsal anlamda hem de kişisel anlamda verileri izinsiz ele geçirmeye yönelik çok ciddi saldırılar yapılmaktadır. Geliştirilen yöntem her türlü bilgisayar uygulamaları ve mobil uygulamalar ile birlikte çalışabilecek altyapıya sahip olup, yapılabilecek ağ dinleme saldırılarına karşı etkili bir çözüm sunmaktadır.

Yapılan çalışmada, taraflar arasındaki iletişimin en hızlı ve en güvenli bir şekilde gerçekleştirilebilmesi amaçlanmıştır. Bu noktada şifreleme işlemi için simetrik anahtarlı AES tercih edilmiştir. Ancak AES'te gizli anahtarın taraflar arasında paylaşımı problem oluşturmaktadır. Bu problem, gizli anahtarı tarafların ortak bir şekilde oluşturabileceği Diffie-Hellman anahtar değişim sistemi ile aşılmış, ayrıca mobil cihazlar gibi bilgisayarlara kıyasla daha düşük kapasiteye sahip ortamlarda da performansı en üst seviyede tutmak için Diffie-Hellman anahtar değişimi, eliptik eğriler kullanılarak gerçekleştirilmiştir.

Bununla birlikte geliştirilen mobil uygulamaya elektronik imza entegrasyonu yapılarak bilgi güvenliği servislerinden gizliliğin yanı sıra; bütünlük, inkâr edememe ve kimlik doğrulama servislerini de kapsaması sağlanmıştır. Böylece oluşturulan melez sistem ile karşılaşılabilecek bütün tehditlere karşı güvenli bir yapı sağlanmıştır.

Performans açısından bakıldığında ise; AES, şifreleme işlemleri için en uygun yöntemdir. Bu noktada AES'te kullanılacak gizli anahtarın, veri iletişimini gerçekleştiren taraflar arasında paylaşımını sağlamak amacıyla kullanılan Eliptik Eğri Diffie-Hellman anahtar değişim sistemi de oldukça etkili bir şekilde gerçekleştirilmiştir.

Elektronik imza açısından bakıldığında ise, performans ve güvenlik bakımından RSA diğer yöntemlere göre daha çok tercih edilmektedir.

Bu çalışmada, gizli verinin paylaşımı ve bu noktada uygulanabilecek güvenlik önlemleri ele alınmıştır. Şifreleme algoritmalarının gücünün, algoritmanın yapısıyla birlikte, algoritmada kullanılan anahtara da bağlı olduğu bilinmektedir. Bir şifreleme sisteminin en önemli noktalarından biri, gizli anahtarın üçüncü şahıslardan korunmasıdır. Bu koruma işlemi; yüksek maliyet ve güvenlik zafiyeti ortaya çıkarabilir. Yapılan çalışmada; maliyet en aza indirilerek, gizli veri paylaşımının güvenli bir şekilde yapılması sağlanmaktadır.

Sonuç olarak; bilginin korunmasını, güvenli ve verimli bir şekilde taşınmasını sağlamak için; gelişen bilgisayar teknolojilerine paralel olarak şifreleme algoritmalarının güçlendirilmesi ve gizli anahtar paylaşımına yönelik çalışmaların sürdürülmesi büyük önem arz etmektedir.

Bu çalışma, söz konusu problemlerin çözümü yönünde atılan bir adım olarak değerlendirilebilir.

## KAYNAKLAR


Altan, K., Kaşkaloğlu, K., Saygı, Z., Yıldırım, E. ve Yıldırım, M. 2004. Kriptolojiye Giriş. ODTÜ Uygulamalı Matematik Enstitüsü, Ankara.

Anonymous. 1999. NIST. Data Encryption Standard. Federal Information Processing Standards Publication No. 46-3.

Anonymous. NIST. 2001. Advanced Encryption Standard. Federal Information Processing Standards Publication No. 197.

Başkök, M. 2007. AES Şifreleme Algoritmasının Modellenmesi. Yüksek Lisans Tezi, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Elektrik-Elektronik Mühendisliği Anabilim Dalı, Ankara.Branovic, I., Giorgi, R. and Martinelli, E. 2013. Memory Performance of Public Key Cryptography Methods in Mobile Environments. University of Siena.

Buluş, H.N. 2006. Temel Şifreleme Algoritmaları ve Kriptoanalizlerin İncelenmesi. Yüksek Lisans Tezi. Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, Edirne.

Çalık, Ç., Sönmez, M. ve Yüce, Z. 2006. E-İmzada SHA-1 Özetleme Algoritmasının Kullanımı. ODTÜ Uygulamalı Matematik Enstitüsü, Ankara.

Daemen, J. and Rijmen, V. 2002. The Design of Rijndael. Springer, 238 p., USA.

Diffie, W. and Hellman, M. 1976. New Directions in Cryptography. IEEE, 22(6); 644-654.

Ersin, G. 2006. Açık Anahtar Altyapısı Eğitim Kitabı. TÜBİTAK UEKAE, 84 s., Kocaeli.

Gupta, S. and Sharma, J. 2012. A Hybrid Encryption Algorithm Based on RSA and Diffie-Hellman. IEEE, December 2012.

Gupta, V., Gupta, S. and Chang, S. 2002. Performance Analysis of Elliptic Curve Cryptography for SSL. ACM Workshop Wireless Security.

Hussain, A. and Khan, A. M. 2012. A Competitive Analysis of Cryptography Tecniques Over Block Cipher. Asian Journal of Computer Science and Information Technology, 11(2); 291-293.

Knudsen, L. and Robshaw, M. 2011. The Block Cipher Companion. Springer, 270 p., USA.

Li, N. 2010. Research on Diffie-Hellman Key Exchange Protocol. IEEE, 4(635); 634-637.

Mahajan, P. and Sachdeva, A. 2013. A Study of Encryption Algorithms AES, DES and RSA for Security. Global Journal of Computer Science and Technology, 13(15).

Mantoro, T. and Zakariya, A. 2012. Securing E-Mail Communication Using Hybrid Cryptosystem on Android-Based Mobile Devices. TELKOMNIKA, December

2012; 807-814.

Menezes, A., Oorschot, P. V. and Vanstone, S. 1996. Handbook of Applied Cryptography. CRC-Press, 780 p., USA.

Oppliger, R. 2005. Contemporary Cryptograpy. Artech House, 510 p., USA.

Paar, C. and Pelzl, J. 2010. Understanding Cryptography. Springer, 372 p., USA.

Palmgren, K. 2006. Diffie-Hellman Key Exchange: A Non-Mathematician's Explanation. Information Systems Security Association Journal, October 2016; 30-33.

Raju, S. and Akbani, R. 2003. Elliptic Curve Cryptosystem and its Applications. IEEE, 1540-1543.

Rasool, S., Sridhar, G. and Kumar, P. 2011. Enhanced Secure Algorithm for Message Communication. IJNSA, September 2011; 33-42.

Ren, W. and Miao, Z. 2010. A Hybrid Encryption Algorithm Based on DES and RSA in Bluetooth Communication. IEEE, August 2010; 221-225.

Ryabko, B. and Fionov, A. 2005. Basics of Contemporary Cryptography for IT Practitioners. World Scientific Publishing Company, 197 p., USA.

Sağıroğlu, Ş. 2005. Her Yönüyle Elektronik İmza. Grafiker, Ankara.

Sakallı, M. 2006. Modern Şifreleme Yöntemlerinin Gücünün İncelenmesi. Doktora Tezi. Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği, Edirne.

Saxena, N. and Chaudhari, N. 2012. Secure Encryption with Digital Signature Approach for Short Message Service. IEEE, 803-806.

Schneier, B. 1996. Applied Cryptography. Wiley, 784 p., USA.

Silverman, J. 1994. Advanced Topics in the Arithmetic of Elliptic Curves. Springer, 528 p., USA.

Stallings, W. 1998. Cryptography and Network Security: Principles and Practise (2nd Edition). Prentice Hall, 569 p., USA.

Sumathy, V. and Navaneethan, C. 2012. Enhanced AES Algorithm for Strong Encryption. International Journal of Advances in Engineering and Technology, 4(2); 547-553.

Trujillo, A., Morales, M. and Ruiz, M. 2012. Elliptic Curve Cryptography on Windows CE Devices. IEEE, April 2012; 224-229.

Vanstone, S., Menezes, A. and Hankerson, D. 2004. Guide to Elliptic Curve Cryptography. Springer, 312 p., USA.

Wang, X., Yu, H. and Yin, Y. 2005. Efficient Collision Search Attacks on SHA-0. Springer, 1-16 p.

Zhou, X. and Tang, X. 2011. Research and Implementation of RSA Algorithm for Encryption and Decryption. IEEE, 1118-1121.