SELÇUK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SELÇUK
ÜNİVERSİTESİ

T.C.

SELÇUK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

KRİPTOLOJİ VE STEGANOGRAFİYLE GÜVENLİ İLETİŞİM SİSTEMİ TASARIMI

Hakan BUHURCU

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Ana Bilim Dalı

Ağustos-2022
KONYA

TEZ KABUL VE ONAYI

Hakan BUHURCU tarafından hazırlanan "***Kriptoloji ve Steganografiyle Güvenli İletişim Sistemi Tasarımı***" adlı tez çalışması 04/08/2022 tarihinde aşağıdaki jüri tarafından oy birliği ile Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Ana Bilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Doç. Dr. Murat SELEK	……..
Danışman Prof. Dr. Fatih BAŞÇİFTÇİ	……..
Üye Doç. Dr. Nurettin DOĞAN	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. Sait GEZGİN
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

İmza

Hakan BUHURCU

Tarih: 04.08.2022

ÖZET

YÜKSEK LİSANS TEZİ

KRİPTOLOJİ VE STEGANOGRAFİYLE GÜVENLİ İLETİŞİM SİSTEMİ TASARIMI

Hakan BUHURCU

Selçuk Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Ana Bilim Dalı

Danışman: Prof. Dr. Fatih BAŞÇİFTÇİ

2022, 155 Sayfa

Jüri
Prof. Dr. Fatih BAŞÇİFTÇİ
Doç. Dr. Murat SELEK
Doç. Dr. Nurettin DOĞAN

Bilgi ve iletişim güvenliği, verilere yetkisiz kişiler tarafından erişimi engelleme yollarının tümü olarak ifade edilebilir ve bu amaca ulaşmak için kriptoloji ve steganografi bilimleri ortaya çıkmıştır. Kriptoloji, çeşitli mantıksal ve matematiksel işlemlerle bilgilerin formunu değiştirerek onları anlamsız yapar. Steganografi ise bilginin yapısında herhangi bir değişim yapmadan onu başka bilgiler içerisine gizleyerek fark edilemez hale getirir. Yani, kriptoloji bilginin içeriğine ulaşılmasını, steganografi ise varlığının keşfedilmesini önler. Bu tez çalışmasında, kriptoloji ve steganografinin entegrasyonu ile güvenli bir iletişim modeli önerilmektedir. Aynı zamanda kriptolojide sorun teşkil eden konulardan birisi olan simetrik anahtarlı şifreleme algoritmalarındaki anahtar paylaşımı problemine de bir çözüm önerisi sunulmaktadır. Yapılan çalışmada, verilerin şifrelenmesi ve doğrulanması için simetrik anahtarlı Gelişmiş Şifreleme Standardı / Galois Sayaç Modu (Advanced Encryption Standart / Galois Counter Mode, AES / GCM) algoritması kullanılmıştır. Burada gizli anahtarın güvenli bir şekilde iletiminin sağlanabilmesi için bir asimetrik şifreleme algoritması olan Rivest Shamir Adleman (RSA)'dan faydalanılmıştır. Böylelikle simetrik ve asimetrik şifreleme algoritmalarının birlikte kullanılmasıyla güvenliğin, En Anlamsız Bit (Least Significant Bit, LSB) steganografi tekniğinden yararlanarak da gizliliğin sağlanması amaçlanmıştır. Ayrıca önerilen iletişim modeli kullanarak Android platformu üzerinde bir mesajlaşma uygulaması geliştirilmiştir ve mobil cihazlar üzerinde kriptoloji algoritmalarının performans değerlendirmesi yapılmıştır. Elde edilen sonuçlara göre AES GCM' nin verilerin içeriğini %99,61 oranında değiştirdiği ve 269,61 MB/s hızında şifreleme yaptığı tespit edilerek mobil cihazlar için en uygun şifreleme algoritması olduğu gözlemlenmiştir. Bunların yanı sıra uygulanan steganografi işlemlerinin başarımı da Ortalama Kare Hata (Mean Squared Error, MSE) ve Tepe Sinyal Gürültü Oranı (Peak Signal to Noise Ratio, PSNR) ölçütlerine göre değerlendirilmiştir. Literatürde, steganografi uygulamalarının başarılı olabilmesi için PSNR değerinin 40 desibel (dB) ve üzeri olması gerektiği ifade edilir. Bu çalışmada, farklı çözünürlükteki görüntüler üzerinde farklı boyutlardaki verilerle uygulanan steganografik işlemlerin sonucunda 51,37 – 82,39 dB aralığında PSNR değerleri elde edilmiştir. Bu açıdan gerçekleştirilen steganografi uygulamalarının literatürde tanımlanan eşik değere göre %28,43 – %105,98 oranında daha başarılı olduğu gözlemlenmiştir.

Anahtar Kelimeler: AES, GCM, Kriptoloji, LSB, RSA, Steganografi.

ABSTRACT

M.Sc. THESIS

DESIGNING OF A SECURE COMMUNICATION SYSTEM WITH CRYPTOLOGY AND STEGANOGRAPHY

Hakan BUHURCU

The Graduate School of Natural and Applied Science of Selçuk University
The Degree of Master of Science in Computer Engineering

Advisor: Prof. Dr. Fatih BAŞÇİFTÇİ

2022, 155 Pages

Jury
Prof. Dr. Fatih BAŞÇİFTÇİ
Assoc. Prof. Dr. Murat SELEK
Assoc. Prof. Dr. Nurettin DOĞAN

Information and communication security can be defined as all the ways to prevent access to data by unauthorized persons. To achieve this goal, the sciences of cryptology and steganography have emerged. Cryptology makes information meaningless by changing the form of information with various logical and mathematical operations. On the other hand, steganography makes it unnoticeable by hiding it in other information without making any changes in the structure of the information. That is, cryptography prevents access to the content of information, while steganography prevents its existence from being discovered. In this thesis, a secure communication model is proposed with the integration of cryptography and steganography. At the same time, a solution proposal is presented to the problem of key sharing in symmetric key encryption algorithms, which is one of the problematic issues in cryptology. In the study, Advanced Encryption Standard / Galois Counter Mode (AES / GCM) which is a symmetric key algorithm was used for encryption and verification of data. Here, Rivest Shamir Adleman (RSA) which is an asymmetric encryption algorithm was used to ensure the secure transmission of the secret key. Thus, it is aimed to ensure security by using symmetric and asymmetric encryption algorithms together, and to ensure confidentiality by using the Least Significant Bit (LSB) steganography technique. In addition, a messaging application has been developed on the Android platform using the proposed communication model and the performance of cryptology algorithms has been evaluated on mobile devices. According to the results obtained, it has been observed that AES GCM changes the content of the data by 99.61% and encrypts at 269.61 MB/s and is the most suitable encryption algorithm for mobile devices. In addition to these, the performance of steganography operations was evaluated according to Mean Squared Error (MSE) and Peak Signal to Noise Ratio (PSNR) criteria. In the literature, it is stated that the PSNR value should be 40 decibel (dB) and above for steganography applications to be successful. In this study, PSNR values in the range of 51.37 - 82.39 dB was obtained as a result of steganographic operations implemented with data of different sizes on images with different resolutions. In this respect, it has been observed that implemented steganography operations are 28.43% – 105.98% more successful than the threshold value defined in the literature.

Keywords: AES, Cryptology, GCM, LSB, RSA, Steganography.

ÖNSÖZ

Yüksek lisans sürecinde ve bu tez çalışmasındaki desteklerinden dolayı danışman hocam Sayın Prof. Dr. Fatih BAŞÇİFTÇİ' ye teşekkür ederim. Ayrıca destek oldukları için aileme de teşekkür ederim.

Hakan BUHURCU
KONYA-2022

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

Simgeler

cov(x,y)	: x ile y arasındaki kovaryans değeri
dB	: Desibel
GB	: Gigabayt
GHz	: Gigahertz
KB	: Kilobayt
kb/s	: Saniye başına kilobit
$\log_{10}$	: 10 tabanlı logaritma
MB	: Megabayt
MB/s	: Saniye başına megabayt
MP	: Megapiksel
ms	: Milisaniye
r_{xy}	: x ile y arasındaki korelasyon katsayısı
s	: Saniye
$\sum$	: Toplama fonksiyonu

Kısaltmalar

AAC	: Advanced Audio Coding (Gelişmiş Ses Kodlama)
AEAD	: Authenticated Encryption with Associated Data (İlişkili Verilerle Kimliği Doğrulanmış Şifreleme)
AES	: Advanced Encryption Standart (Gelişmiş Şifreleme Standardı)
API	: Application Programming Interface (Uygulama Programlama Arayüzü)
APK	: Android Package Kit (Android Paket Kiti)
ASCII	: American Standard Code for Information Interchange (Bilgi Değişimi için Amerikan Standart Kodlama Sistemi)
ASIC	: Application Specific Integrated Circuit (Uygulamaya Özel Tümleşik Devre)
AVD	: Android Virtual Devices (Android Sanal Cihazları)
BMP	: Bitmap
BPCS	: Bit Plane Complexity Segmentation (Bit Düzlem Karmaşıklık Segmentasyonu)
BSD	: Berkeley Software Distribution (Berkeley Yazılım Dağıtımı)
CBC	: Cipher Block Chaining (Şifre Bloğu Zincirleme)
CBC-MAC	: Cipher Block Chaining Message Authentication Code (Şifre Bloğu Zinciri Mesaj Doğrulama Kodu)
CFB	: Cipher Feedback (Şifre Geribildirim)
CTR	: Counter (Sayaç)
CUDA	: Compute Unified Device Architecture (Hesaplama Bileşik Cihaz Mimarisi)
CWC	: Carter Wegman Sayaç (Carter Wegman Counter)
DCT	: Discrete Cosine Transform (Ayrık Kosinüs Dönüşümü)
DES	: Data Encryption Standart (Veri Şifreleme Standardı)
DFT	: Discrete Fourier Transform (Ayrık Fourier Dönüşümü)
DOM	: Document Object Model (Belge Nesne Modeli)
DWT	: Discrete Wavelet Transform (Ayrık Dalgacık Dönüşümü)
3DES	: Triple Data Encryption Standart (Üçlü Veri Şifreleme Standardı)
EAX	: Encrypt Authenticate Translate (Şifrele Doğrula Çevir)
ECB	: Electronic Code Book (Elektronik Kod Kitabı)
ECDH	: Elliptic Curve Diffe Hellman (Eliptik Eğri Diffie Hellman)
ECDSA	: Elliptic Curve Digital Signature Algorithm (Eliptik Eğri Dijital İmza Algoritması
EEG	: Electroencephalography (Elektroensafolagram)
FFT	: Fast Fourier Transform (Hızlı Fourier Dönüşümü)

FPGA : Field Programmable Gate Array (Alanda Programlanabilir Kapı Dizileri)

FIPS-PUB : Federal Information Processing Standards Publications (Federal Bilgi İşleme Standartları Yayını)

GCM : Galois Counter Mode (Galois Sayaç Modu)

GIF : Graphics Interchange Format (Grafik Değişim Formatı)

GMAC : Galois Message Authentication Code (Galois Mesaj Doğrulama Kodu)

GPU : Graphics Processing Unit (Grafik İşlem Birimi)

HMAC : Hash-based Message Authentication Code (Özet Tabanlı Mesaj Doğrulama Kodu)

IBM : International Business Machines (Uluslararası İş Makineleri)

ICMP : Internet Control Message Protocol (İnternet Kontrol Mesaj Protokolü)

IoT : Internet of Things (Nesnelerin İnterneti)

IP : Internet Protocol (İnternet Protokolü)

IPSec : Internet Protocol Security (İnternet Protokolü Güvenliği)

ID : Identification Number (Kimlik Numarası)

IV : Initialization Vector (Başlangıç Vektörü)

İV : İlişkili Veri

JPEG : Joint Photographic Experts Group (Birleşik Fotoğraf Uzmanları Grubu)

JSON : JavaScript Object Notation (JavaScript Nesnesi Gösterimi)

KSA : Key Scheduling Algorithm (Anahtar Planlama Algoritması)

LSB : Least Significant Bit (En Önemsiz Bit)

LSBM : Least Significant Bit Matching (En Önemsiz Bit Eşleşmesi)

LSD : Least Significant Digit (En Önemsiz Basamak)

MARS : Çok Değişkenli Uyarlanabilir Regresyon Uzanımları (Multivariate Adaptive Regression Splines)

MD5 : Message Digest 5 (Mesaj Özeti 5)

MIT : Massachusetts Institute of Technology (Massachusetts Teknoloji Enstitüsü)

MP3 : Motion Pictures Experts Group Audio Layer 3 (Film Uzmanlar Grubu Ses Katmanı 3)

MQTT-SN : Message Queuing Telemetry Transport for Sensor Nodes (Sensör Ağları için Mesaj Kuyruğu Telemetri Aktarımı)

MQTT-TLS : Message Queuing Telemetry Transport for Transport Layer Security (Taşıma Katmanı Güvenliği için Mesaj Kuyruğu Telemetri Aktarımı)

MR : Magnetic Resonance (Manyetik Rezonans)

MSE : Mean Squared Error (Ortalama Kare Hatası)

NIST : National Institute of Standards and Technology (Ulusal Standartlar ve Teknoloji Enstitüsü)
NPCR : Number of Pixel Change Rate (Değişen Piksel Sayısı Oranı)
NSA : Ulusal Güvenlik Ajansı (National Security Agency)
NTRU : Number Theory Research Unit (Sayı Teorisi Araştırma Birimi)
OFB : Output Feedback (Çıktı Geribildirim)
OMAC : One-key Message Authentication Code (Tek-anahtar Mesaj Doğrulama Kodu)
OpenCV : Open Source Computer Vision Library (Açık Kaynak Bilgisayar Görüsü Kütüphanesi)
OPT : Orthopantomogram
PDF : Portable Document Format (Taşınabilir Belge Biçimi)
PEM : Privacy-Enhanced Mail (Gizliliği Artırılmış Posta)
PGP : Pretty Good Privacy (Oldukça İyi Gizlilik)
PNG : Portable Network Graphics (Taşınabilir Ağ Grafiği)
PRGA : Pseudo-Random Generation Algorithm (Sözde-Rastgele Üreteç Algoritması)
PSNR : Peak Signal to Noise Ratio (Tepe Sinyal Gürültü Oranı)
RC4 : Rivest Cipher 4 (Rivest Şifre 4)
RC6 : Rivest Cipher 6 (Rivest Şifre 6)
RGB : Red Green Blue (Kırmızı Yeşil Mavi)
RSA : Rivest Shamir Adleman
SDK : Software Development Kit (Yazılım Geliştirme Kiti)
SET : Secure Electronic Transaction (Güvenli Elektronik İşlem)
SHA : Secure Hash Algorithm (Güvenli Karma Algoritması)
S-HTTP : Secure Hypertext Transfer Protocol (Güvenli Köprü Metni Aktarım Protokolü)
S/MIME : Secure / Multipurpose Internet Mail Extensions (Güvenli / Çok Amaçlı İnternet Posta Uzantıları)
SNR : Signal to Noise Ratio (Sinyal Gürültü Oranı)
SQL : Yapılandırılmış Sorgu Dili (Structured Query Language)
SSIM : Structural Similarity (Yapısal Benzerlik)
SSH : Secure Shell (Güvenli Kabuk)
SSL : Secure Socket Layer (Güvenli Soket Katmanı)
S/WAN : Secure / Wide Area Network (Güvenli / Geniş Alan Ağı)
TCP : Transmission Control Protocol (İletim Denetimi Protokolü)
TLS : Transport Layer Security (Taşıma Katmanı Güvenliği)

UACI : Unified Average Changing Intensity (Birleşik Ortalama Yoğunluk Değişimi)

UDP : User Datagram Protocol (Kullanıcı Veri Bloğu Protokolü)

URL : Uniform Resource Locator (Tekdüzen Kaynak Bulucu)

WAV : Waveform Audio File Format (Dalga Formu Ses Dosyası Formatı)

WEP : Wired Equivalent Privacy (Kablolu Eşdeğer Gizlilik)

WPA : Wireless Fidelity Protected Access (Kablosuz Bağlantı Alanı Korumalı Erişim)

XML : Extensible Markup Language (Genişletilebilir İşaretleme Dili)

XOR : Exclusive Or (Özel Veya)

ŞEKİLLER LİSTESİ

ÇİZELGELER LİSTESİ

1. GİRİŞ

İletişim genel olarak anlama ve paylaşma süreci olarak tanımlanmaktadır. (Anonymous, 2015). Bir etkinlik, değişim veya davranışlar dizisinden meydana geldiği için bir süreç olarak kabul edilir. Çünkü sabit bir yapı değildir. İletişim kavramı, aynı zamanda anlam oluşturmak için mesaj kullanma süreci olarak da ifade edilir. Bu mesajlar, sözlü ve sözsüz sembollerden, işaretlerden ve davranışlardan oluşabilir. Örnek olarak birisi başka birisine gülümsediği zaman aslında bir mesaj göndermiş olur veya bir radyo spikeri son zamanlarda yaşanan bir olayın ciddiyetini vurgulayacak bir dil seçtiğinde bile yine bir mesaj oluşturmaktadır (Pearson ve ark., 2017). Tabi ki iletişim kavramı sadece insanlarla sınırlı değildir. Bilgisayarlar, mobil aygıtlar, Nesnelerin İnterneti (Internet of Things, IoT) cihazları vb. dijital platformlar da günümüzde iletişim sürecinin etkin katılımcılarıdır. Bu dijital katılımcılar da iletişim kurmak için dijital mesajlar oluştururlar ve bu mesajlarla birbirlerinin işleyişlerini etkilemeye ve yönlendirmeye çalışırlar. İnsanlar ise verdikleri mesajlar aracılığıyla ortak anlamlar üretmeyi umarlar. Anlam kavramı ise mesajın anlaşılması olarak ifade edilebilir. Oluşturulan her mesajla istenilen anlam üretilemeyebilir. Örnek olarak öğretmenler genellikle kendi alanlarıyla ilgili bir konu hakkında çok bilgilidir, ancak bu bilgileri aktarma yetenekleri birbirlerine göre büyük farklılıklar gösterebilir (Cherry, 1957).

İletişim; gönderici (kaynak), alıcı (hedef), mesajlar, kanallar, geri bildirimler, kodlar, kodlama ve kod çözme, gürültü ve durum olmak üzere 9 bileşenden meydana gelmektedir (Pearson ve ark., 2017). Gönderici ya da kaynak bir mesaj başlatır ve alıcı bu mesajın amaçlanan hedefidir. Katılımcılar bu iki rolü birbirinden bağımsız olarak yerine getirmezler. Aksine mesajların kaynakları ve alıcıları eş zamanlı ve sürekli olarak hareket ederler. Mesaj, bir kişinin (kaynak) başka bir kişiye veya bir grup insana (alıcılar) iletmek istediği fikir, düşünce veya duygunun sözlü ve sözsüz şeklidir (Küçük ve ark., 2012). Yani mesaj, etkileşimin içeriğidir. Mesajlar, düşünceleri iletmek için kullanılan kelime ve sembollerin yanı sıra jest ve mimikler, hareketler, fiziksel temas ve ses tonu gibi diğer sözsüz kodları da içerebilir (Anonymous, 2015).

Dijital dünyada ise mesaj kavramı, herhangi bir cihazdan başka bir cihaza iletilecek metin, resim, ses, video, komut ve işaret gibi analog ve dijital sinyallerle taşınabilecek her türlü veriyi kapsar. Kanal, bir mesajın kaynaktan hedefe iletildiği araçtır. Örnek olarak insanlar yüz yüze konuşurken ses dalgaları aracılığıyla birbirlerine mesaj gönderirler veya telefonla konuşurken radyo dalgaları sayesinde iletişim kurarlar

(Küçük ve ark., 2012). Kanal kavramı, kısa mesajları veya sosyal medyadaki durum güncellemeleri gibi çeşitli örnekleri de içerir (Anonymous, 2015). Bu örneklerin her biri katılımcıların birçok farklı kanalı kullanarak nasıl iletişim kurabileceğini göstermektedir. Bilgisayarlar içinse kablolar, kablosuz ağlar, fiber hatlar, elektromanyetik sinyaller vb. gibi bilgi iletilebilecek her türlü ortam kanal olarak değerlendirilebilir. Geri bildirim, alıcının kaynağın mesajına verdiği sözlü ve sözsüz yanıtları ifade etmektedir. Kaynağın mesajın istediği gibi alındığını bilmesi için geri bildirim gönderilir. Geri bildirim her iletişimde mutlaka bulunur. Tepki vermemek veya susmak bile bir geri bildirimdir. Katılımcıların iletişim sürecinde mesajların anlamlarını doğrulamaları geri bildirim yoluyla gerçekleşir (Pearson ve ark., 2017). Bilişim dünyasında geri bildirim için İnternet Kontrol Mesaj Protokolü (Internet Control Message Protocol, ICMP) örnek olarak verilebilir. ICMP, bilgisayar ağlarında temel olarak ilgili hata ve kontrol mesajlarını göndermek için zorunlu bir mekanizma olarak kullanılır. Protokol, ICMP mesajları göndererek iletişim ortamındaki sorunlar hakkında geri bildirim almayı sağlar (Rosen, 2014).

Bilgisayarlar mesajları, bir kablo veya fiber hat üzerinde ikili kodlar vasıtasıyla taşır. Benzer şekilde insanlar da birbirleriyle "dil" adı verilen bir kod kullanarak iletişim kurarlar (Cherry, 1957). Bir kod, başka bir kişi veya kişilerin zihninde anlamlar oluşturmak için kullanılan sembollerin sistematik bir düzenlemesidir. Kelimeler, deyimler ve cümleler, imgeleri, fikirleri ve düşünceleri başkalarının zihninde uyandırmak için kullanılan "semboller" haline gelir (Pearson ve ark., 2017). İletişim kodların kullanımını içeriyorsa, iletişim süreci bir kodlama veya kod çözme olarak görülebilir. Kodlama, bir fikri veya düşünceyi bir koda çevirme işlemidir. Kod çözme ise bu fikir veya düşünceye yüklenen anlamın çıkarılması sürecidir (Küçük ve ark., 2012). Kodlama ve kod çözme süreci dijital platformlar için daha somut bir şekilde ifade edilebilir. Çünkü dijital platformlarda bilgiler standartlaşmış belirli formatlara dönüştürülerek iletilir. Dolayısıyla kaynak tarafı bilgiyi ilgili formata dönüştürmek için kodlarken hedef taraf da kendisine ulaşan bilgiyi okumak için kod çözme sürecini işletir. İletişim sürecinde gürültü, kodlama ve kod çözme sürecinde veya iletim esnasında oluşabilecek mesajın netliğini azaltan herhangi bir parazittir. Örnek olarak kablolardaki elektronların hareketliliği, şehir şebekesi veya kablosuz haberleşmede şimşek, yıldırım gibi etmenler gürültüye sebep olmaktadır (Anonim, 2018). İnsanlar arasındaki iletişimde ise gürültü, yüksek sesler gibi fiziksel olmasının yanı sıra yoğun kalabalığın bulunduğu dikkat dağıtan ortamlar veya rahatsız edici davranışlar da olabilir. Kısaca gürültü; bir mesajın

alınmasını, yorumlanmasını veya bir mesaj hakkında geri bildirim sağlanmasını engelleyen her şey olabilir (Anonymous, 2015).

İletişimin son bileşeni durumdur ve iletişimin gerçekleştiği konumu ifade eder. Durum, iletişimin bulunduğu bağlama göre değişir. Bu bağlam kişilerarası iletişimden kitlesel iletişime kadar birçok farklı çeşitte olabilir. Her bağlam iletişim kurulduğunda farklı türde bir durum sağlar. Örneğin iki kişi arasındaki bir konuşma daha az resmi olma eğilimindedir, oysa yüzlerce kişinin önünde yapılan halka açık bir konuşma daha fazla resmi olabilir. Katılımcılar arasındaki ilişki de durumu etkiler. Örnek olarak insanlar patronlarıyla iş arkadaşlarından farklı bir şekilde iletişim kurar. Hatta kanal bile durumu etkileyebilmektedir. Yüz yüze iletişim sosyal medyanın bazı biçimlerine göre daha kişisel olabilir (Pearson ve ark., 2017). İletişim teknolojilerinde ise durum kavramı genellikle kullanılan kanalın donanımsal ve yazılımsal özellikleriyle iletişim ortamının fiziki ve beşeri coğrafi şartlarını ifade eder. Çünkü tüm bunlar iletişimin kalitesini etkiler. Örnek olarak fiber kabloyla döşenmiş bir hat üzerinde standart bakır kabloyla döşenmiş bir hatta göre daha hızlı veri aktarımı sağlanabilir ve daha kaliteli bir iletişim kurulabilir. Aynı şekilde kablosuz iletişimde olumsuz hava şartları iletişim kalitesini düşürebilir. Tüm bunlar değerlendirildiğinde durum kavramının etkileşimin genel şeklini belirlemek için iletişim sürecinin diğer unsurlarını birleştirdiği söylenebilir.

Günümüzün kompleks iletişim ortamında insanlar, kişilerarası iletişimde tercih edebilecekleri çok çeşitli kanal seçeneğine sahiptir. Yüz yüze iletişim insanlık tarihinin en eski iletişim yöntemi olmasına rağmen hala kişilerarası iletişim için günlük hayatta en yaygın kullanılan yöntemdir. İnsanların iletişim kurma kapasitesini artıran yeni iletişim araçları toplumsal hayata hızla entegre olmuştur. İlk olarak yazı icat edilmiştir ve sonra kişilerarası iletişim kanallarına telgraf ve telefon da eklenmiştir. Son olarak, cep telefonları ve internet kanalları (örneğin, e-posta veya anlık mesajlaşma) modern dünyada yeni ve heyecan verici iletişim araçları olarak birçok insanın hayatına girmiştir. İnsanlar bu iletişim kanallarını bilgi arama, eğlence, arkadaşlık ve kişiler arası iletişim gibi çeşitli amaçlar için kullanmaktadır (Tosun, 2010). Son yıllarda teknolojik iletişim kanallarının kullanımı insanların rutin sosyal yaşamlarının bir parçası haline gelmiştir.

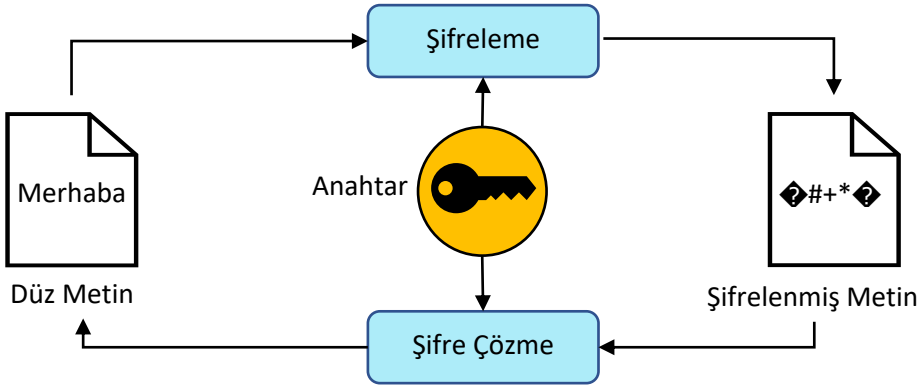
İletişim teknolojisi, birçok farklı şekilde kategorize edilebilen çok çeşitli donanım ve yazılım ürünlerini içerir. İletişim teknolojisinin en bilinen örnekleri e-posta, anlık mesajlaşma, sosyal medya ve mobil cihazlardır. Günümüzde iletişim ve ağ teknolojilerinin gelişmesiyle birlikte verilerin depolanması ve iletilmesi esnasında korunabilmesi için güvenlik önemli bir konu olmuştur. Güvenli olmayan kanallarda

verileri yok edilmekten veya yetkisiz erişimlerden korumanın yolu şifrelemedir. Kriptoloji olarak bilenen bu bilgi güvenliği uygulaması, kimlik doğrulama, bütünlük ve gizliliğin sağlanmasıyla ilgilidir (Imran ve ark, 2020). Bu kavramlar kriptolojiyle ulaşılmak istenen hedeflerdir. Kimlik doğrulama, özel bir kaynağa erişebilmesi için bir kişiye kimlik tanımlama sürecidir. Gizlilik sadece yetkisi bulunan kişilerin mesajı aldığını doğrulayan şifreleme sistemlerinin nihai hedefidir. Veri bütünlüğü, belirli bir gruba veya kişiye ait verileri modüle etme erişimine sahip olan işlemdir. Bunlara ek olarak kriptolojinin hedefleri arasında inkar edilemezlik ve erişim kontrolü de vardır. İnkar edilemezlik hem göndericinin hem de alıcının mesajın teslim edildiğini onaylamasını sağlar. Erişim kontrolü ise yalnızca yetkilendirilmiş kişilerin teslim edilen mesajın içeriğine erişebileceğini doğrular (Abood ve Guirguis, 2018).

Kriptografi, sözlük anlamında kod yazma ve çözme sanatı olarak tanımlanmıştır. Bu tarihsel olarak doğrudur, ancak alanın mevcut genişliğini veya bilimsel temellerini yansıtmaz. Bu tanım, yalnızca yüzyıllarca gizli iletişimi sağlamak için kullanılan kodlara odaklanmıştır. Ancak günümüzde kriptografi bundan çok daha fazlasını kapsamaktadır; veri bütünlüğünü sağlamak için mekanizmalar, gizli anahtarların değiş tokuşu için teknikler, kullanıcıların kimliğini doğrulamak için protokoller, elektronik oylama, kripto para ve daha fazlası ile ilgilenir. Tam olarak karakterize etmeye çalışmadan modern kriptografinin dijital bilgi ve sistemleri güvence altına almak için matematiksel teknikleri; yapılabilecek saldırılara karşı da karmaşık hesaplamaları içerdiğini söyleyebiliriz (Katz ve Lindell, 2021). Sözlük tanımı aynı zamanda kriptografiye bir sanat olarak atıfta bulunur. 20. yüzyılın sonlarına kadar kriptografi aslında büyük ölçüde bir sanattır. İyi kodlar oluşturmak veya mevcut olanları kırmak, üretkenliğe ve kodların nasıl çalıştığına dair gelişmiş bir anlayışa dayanmaktadır. Güvenilecek çok az teori vardır ve uzunca bir zaman iyi bir kodu neyin oluşturduğunu ifade eden bir tanım yoktur. 1970'ler ve 1980'lerden başlayarak, kriptografinin bu resmi kökten değişmiştir. Kriptografinin bir bilim ve matematik disiplini olarak titiz bir şekilde incelenmesini sağlayan zengin bir teori, ortaya çıkmaya başlamıştır. Bu bakış açısı, araştırmacıların bilgisayar sistemlerinin güvenliği hakkındaki düşüncelerini etkilemiştir. Klasik kriptografi (1980'lerden öncesi) ile modern kriptografi arasındaki bir diğer çok önemli fark, onun kullanımıyla ilgilidir. Tarihsel olarak, kriptografinin başlıca kullanım alanları askeri kuruluşlar ve hükümetlerdir. Bugün kriptografi her yerdedir. Bir parola yazarak kimliğinizi doğruladıysanız, internet üzerinden kredi kartıyla bir ürün satın aldıysanız veya işletim sisteminiz için doğrulanmış bir güncelleme indirdiyseniz kriptografi kullanmışsınızdır.

Günümüzde nispeten az deneyime sahip programcılardan bile yazdıkları uygulamaları kriptografik mekanizmalar kullanarak güvence altına almaları istenmektedir. Kısacası kriptografi, birkaç basit uygulama için gizli iletişim sağlamaya yönelik bir buluşsal teknikler grubu olmaktan çıkıp daha genel olarak dünyadaki insanlar için sistemlerin güvenliğini sağlamaya yardımcı olan bir bilime geçmiştir (Katz ve Lindell, 2021).

Kriptoloji aslında kriptografi ve kriptoanaliz kavramlarının birleşimidir. Şekil 1.1' de genel işleyişi gösterilen kriptografi, mesajları okunamaz hale getirmek için onları şifreleyerek güvenliği sağlamaya çalışır. Kriptoanaliz ise şifreli mesajların nasıl şifrelendiğini bilmeden tekrar okunabilir hale getirilmesi için kullanılan teknikleri ifade eder. Kriptografi ilk başlarda manuel teknikler kullanılarak uygulanmıştır. Daha sonraları gerçekleştirilen uygulamalarda kriptografi üzerinde birçok iyileştirme yapılmıştır. Daha önemlisi bu kriptografik algoritma ve fonksiyonları artık bilgisayarlar gerçekleştirmektedir ve böylece bu işlemleri uygulamak çok daha hızlı ve güvenlidir (Kahate, 2013). Kriptoloji aynı zamanda veri şifreleme ve şifre çözme işlemleri için matematik kullanan bir bilim olarak da tanımlanabilir. Şifreleme, anlamsız kodlar kullanarak verinin biçimini belirli sembollere dönüştürme işlemidir (Abood ve Guirguis, 2018). Böylece veriler, istenmeyen kişiler tarafından kolayca anlaşılamaz ve değiştirilemez hale gelir. Deşifreleme olarak da adlandırılan şifre çözme işlemi ise şifrelemenin tam tersi olup şifreli veriden tekrar orijinal verilerin elde edilme sürecini ifade eder. Her şifreleme ve deşifreleme işleminin algoritma ve anahtar olmak üzere iki temel bileşeni vardır. Şifreleme ve şifre çözme için kullanılan algoritma aynı olmalıdır. Aksi halde deşifreleme yapılarak orijinal mesaj geri elde edilemez. Şifreleme ve deşifreleme işlemlerinin ikinci bileşeni ise anahtardır. Anahtar, iletişim kuran taraflarca önceden paylaşılan ve başkaları tarafından bilinmeyen her kripto sistem için belirli uzunluklarda kullanılan gizli bir sayısal veridir. Gönderici bu anahtar yardımıyla orijinal mesajı şifreler ve bir kanal üzerinden alıcıya iletir. Alıcı da kendisine ulaşan şifreli mesajı yine bir anahtar vasıtasıyla deşifre ederek orijinal mesaja ulaşır. Alıcı ve göndericinin kullanmış olduğu bu anahtarlar tercih edilen algoritmaya göre birbirinin aynısı da olabilir veya birbirinden farklı da olabilir. Anahtarların aynı olması durumunda bu uygulamalar simetrik veya gizli anahtarlı kripto sistemler olarak; anahtarların farklı olması durumunda ise asimetrik veya açık anahtarlı kripto sistemler olarak adlandırılırlar. Simetrik anahtarlı sistemlerde şifreleme ve şifre çözme işlemleri için ortak olarak kullanılan tek bir gizli anahtar mevcuttur. Asimetrik anahtarlı sistemlerde ise biri açık biri gizli olmak üzere aralarında matematiksel bir bağıntı olan iki anahtar bulunur.



Şekil 1.1. Genel bir kripto sistem

Kriptografi, iletişim güvenliğini sağlamak için bir teknik olarak ortaya konmuştur. Bu amaç doğrultusunda verilerin yetkisiz kişiler tarafından anlaşılmasını engellemek için birçok farklı yöntem geliştirilmiştir. Ancak bazen bir mesajın sadece içeriğini gizlemek yeterli değildir ve mesajın varlığını gizli tutmak da gerekli olabilir. Bunu uygulamak için kullanılan tekniğe ise *steganografi* denir (Morkel ve ark, 2005). Steganografi kelimesi Yunancadaki kapak anlamına gelen *stego* ve yazı anlamına gelen *grafi* kelimelerinden türetilmiştir ve örtülü yazı olarak tanımlanır (Smitha ve Baburaj, 2016). Steganografi görünmez iletişim bilimi olarak tarif edilebilir ve istenmeyen bir alıcının gizli verilerin varlığından şüphelenmesini önler (Hussain ve Hussain, 2013). Bu da gönderilecek veriler, başka verilerin içerisine saklanarak yapılır ve böylece iletilen bilgilerin varlığı gizlenmiş olur. İçerisine bilgi saklanacak yapılar örtü, kapak veya taşıyıcı dosya olarak ifade edilir. Bu dosyalar metin formatında olabileceği gibi görüntü, ses veya video gibi multimedya ortamları da olabilir. Ancak dijital görüntüler ve sesler, internetteki sık kullanımlarından dolayı daha fazla tercih edilmektedir.

Temel olarak kullanılan üç tip steganografik protokol vardır. Bunlar; saf, gizli anahtarlı ve genel (açık) anahtarlı steganografidir (Sheelu ve Ahuja, 2013).

- Saf Steganografi, stego anahtarı olarak adlandırılan herhangi bir anahtarın kullanılmasını gerektirmeyen steganografik sistemleri ifade eder. Bu yöntem pek güvenli değildir çünkü gönderici ve alıcı diğer tarafların gizli mesajdan haberdar olmadığı varsayımına güvenir.
- Gizli anahtarlı steganografi, iletişimden önce gizli bir stego anahtarın değiş tokuşunu gerektiren bir steganografik sistem olarak tanımlanır. Bu yöntemde bilgiler gizli bir anahtar kullanılarak taşıyıcı dosyanın içerisine gömülür. Yalnızca gizli anahtarı bilen taraflar işlemi tersine çevirebilir ve gizli mesajı okuyabilir. Böylece taşıyıcı dosyada gizli bir bilginin varlığı fark edilirse bile

stego-anahtara sahip olmayan kişiler gizli mesajı taşıyıcı dosyadan çıkaramazlar.

- Genel (açık) anahtar steganografisi, gizlice iletişim kurmak isteyen taraflar arasındaki iletişimi güvence altına almak için biri açık biri özel olmak üzere iki farklı anahtar kullanır. Gönderici, kodlama işlemi sırasında açık anahtar kullanır ve yalnızca açık anahtar ile doğrudan matematiksel ilişkisi olan özel bir anahtar gizli mesajı deşifre edebilir. Bu yöntem, çok daha sağlam ve açık anahtar kriptografisinde araştırılmış bir teknolojiyi kullandığı için bir steganografik sistemi uygulamanın daha güçlü bir yolunu sağlar.

Etkili bir steganografik yöntem aşağıda maddeler halinde ifade edilen özelliklere sahip olmalıdır (Sheelu ve Ahuja, 2013):

- *Gizlilik:* Bir kişi, gizli anahtarı bilmeden taşıyıcı ortamdaki saklı bilgileri çıkaramamalıdır.
- *Algılanamazlık:* Gizli verilerin gömülü olduğu stego-ortam, orijinal ortamdan ayırt edilemez olmalıdır. Başka birileri ortamdaki gizli verilerin varlığından şüphelenmemelidir.
- *Yüksek kapasite:* Gömülebilecek maksimum mesaj uzunluğu, olabildiğince fazla olmalıdır.
- *Direnç:* Taşıyıcı dosyada bazı kayıplı sıkıştırma işlemleri gibi manipülasyonlar yapılsa bile gizli veriler bozulmadan varlığını sürdürebilmelidir.
- *Doğru çıkarma:* Ortamdan gizli verilerin çıkarılması doğru ve güvenilir olmalıdır.

Kriptografide bir mesajın yapısı, onu anlamsız ve anlaşılmaz hale getirmek için karıştırılır. Kodlanmış mesajı saklamak veya gizlemek için hiçbir girişimde bulunulmaz. Dolayısıyla herhangi bir taşıyıcı veya örtü ortamına ihtiyaç duyulmaz (Smitha ve Baburaj, 2016). Kriptografi, üçüncü bir kişinin gizli bilgileri okumasını engelleyecek şekilde kişiler arasında bilgi aktarma imkanı sunar. Kriptografinin amacı, verileri anlaşılamayacak bir forma dönüştürerek iletişimi güvence altına almaktır. Steganografi ise kriptografinin aksine gizli mesajın yapısını değiştirmez, mesajları fark edilemeyecek şekilde bir taşıyıcı dosyanın içerisine saklar. Steganografi bilgileri, varlıkları tespit

edilemeyecek şekilde bir iletişim kanalında saklama tekniğidir (Jain ve ark., 2012). Steganografinin ana hedefi tamamen algılanamaz şekilde güvenli bir iletişim sağlamak ve gizli verilerin iletilmesinde şüphe uyandırmaktan kaçınmaktır. Dolayısıyla steganografi ile kriptografi arasındaki farklardan bahsedecek olursak ilk olarak steganografide mesaj geçişi bilinmezken kriptografide mesaj geçişinin bilindiğini söyleyebiliriz. Steganografi, iletişimin varlığının keşfedilmesini önlerken kriptografi ise yetkisiz kişiler tarafından iletişimin içeriğinin keşfedilmesini önler (Morkel ve ark, 2005). Steganografi az bilinen bir teknolojidir fakat kriptografi ise yaygın kullanılan bir teknolojidir. Steganografi teknolojisi hala belirli formatlar için geliştirilmeye devam etmektedir. Kriptografide algoritmaların çoğu herkes tarafından bilinir. Steganografide dezavantaj olarak tespit edilen bir mesaj artık bilinir. Ancak kriptografi de mevcut güçlü algoritmalar saldırılara karşı dirençlidir ve bu algoritmaların kırılması için daha büyük ve pahalı bilgi işlem gücü gereklidir.

1.1. Tezin Amacı

Bu tez çalışmasında, yukarıda bahsedilenlerden hareketle kriptoloji ve steganografi bilimlerinin hem kendi içlerinde hem de birbirleriyle entegrasyonu gerçekleştirilerek iletişimde güvenliğin sağlanması ve artırılması amaçlanmıştır. Bu kapsamda kriptoloji ve steganografi bilimlerinin muhtevasında yer alan yöntemler ele alınıp değerlendirilmiş ve hem işlem zamanı hem de güvenlik açısından en iyi sonucu sağlayan uygulama belirlenmiştir. Bunun için öncelikli olarak kriptoloji biliminde güvenlik zafiyeti oluşturabilecek noktalar ifade edilmiştir. Bunlardan en önemlisi iletişim kuran taraflar arasında anahtar değişiminin nasıl yapıldığıdır. Anahtarın, alıcı ve gönderici arasında güvenli bir kanal üzerinden iletilmesi gerekmektedir. Ancak bu, her zaman mümkün değildir ve sürekli olarak aynı gizli anahtarın kullanılmasına ya da kullanılan gizli anahtarın seyrek aralıklarla değiştirilmesine sebep olmaktadır. İşte bu durum da başka bir güvenlik zafiyeti oluşturur. Uzun süre aynı anahtarın kullanılması gizli bilginin yetkisiz kişilerce deşifre edilerek ortaya çıkarılması ihtimalini artırır. Bu soruna bir çözüm yine kriptoloji biliminin kendi içerisinden gelmektedir. Bunun için asimetrik ve simetrik şifreleme algoritmaları birlikte kullanılmıştır. Asimetrik şifreleme algoritmalarında verilerin şifrelenmesi ve deşifre edilmesi için farklı anahtarlar kullanılmaktadır. Dolayısıyla anahtar değişimi için bir güvenlik sorunu söz konusu değildir. Bu durumda simetrik şifreleme algoritmasında kullanılan gizli anahtar, başka bir

asimetrik şifreleme algoritmasıyla şifrelenerek alıcıya iletilmektedir. Böylelikle simetrik şifreleme algoritmalarında kullanılan gizli anahtar rahatlıkla ve sıklıkla değiştirilebilmektedir.

Haberleşmede bilgi güvenliğinin sağlanması için şifrelemenin yanında bir de şifrelenmiş verilerin gizlenerek iletilmesi elbette güvenliği artırmaktadır. Bunun için kriptoloji algoritmalarıyla şifrelenen veriler, steganografi algoritmasıyla da bir örtü nesnenin içerisine gömülmektedir. Örtü nesnesi olarak görüntü, ses, video gibi bütün medya ortamları kullanılabilir. Burada kullanılacak ortamın ihtiyaca göre belirlenmesi önemlidir.

Bu tez çalışmasında, Android işletim sistemli mobil cihazlarda kullanılmak üzere güvenli ve gerçek zamanlı ya da başka bir deyişle anlık iletişim sağlayan bir mesajlaşma uygulamasının geliştirilmesi amaçlanmıştır. Bu amaç doğrultusunda kriptoloji ve steganografi tekniklerinden yararlanılarak iletişim güvenliğinin sağlanması hedeflenmiştir. Bu kapsamda tasarlanan mesajlaşma uygulamasında simetrik ve asimetrik kripto sistemlerin birlikte kullanımıyla mesajların güvenliğinin sağlanması ve buna ek olarak steganografiden faydalanılarak bu mesajların aynı zamanda gizliliğinin de korunması planlanmıştır.

1.2. Tezin Önemi

Bu tez çalışmasının önemini ifade eden üç ayrı nokta bulunmaktadır. Bunlardan ilki, günümüz iletişim teknolojisinin en önemli parçası haline gelen mobil cihazlarda iletişim güvenliğinin ve gizliliğinin sağlanması için kriptoloji ve steganografinin entegrasyonuyla gerçekleştirilmiş farklı bir bakış açısının uygulanmasıdır. İkincisi, bu bakış açısını uygularken aynı zamanda kriptoloji biliminde sorun teşkil eden önemli konulardan birisi olan simetrik anahtarlı algoritmalardaki anahtar paylaşım problemine de bir çözüm önerisi sunulmasıdır. Üçüncü önemli nokta ise yapılan bu tez çalışmasının hem kriptoloji hem de steganografinin detaylı bir araştırmasını içermesi ve bunun da bilgi güvenliği ile alakalı yapılacak gelecek çalışmalar için bir kılavuz ve referans niteliğinde olmasıdır.

1.3. Tezin Organizasyonu

Tez çalışmasının birinci bölümde ilk olarak iletişim kavramı tanımlanmış ve genel hatlarıyla açıklanmıştır. İletişimin nasıl sağlandığından, bileşenlerinden ve bu bileşenlerin birbirleriyle olan ilişkilerinden bahsedilmiştir. Ayrıca iletişim kavramı hem insanlar arası iletişimden hem de dijital platformlar arası iletişimden örnekler verilerek daha somut bir şekilde ifade edilmeye çalışılmıştır. Giriş bölümünün devamında ise iletişim güvenliğinin sağlanmasında kriptoloji ve steganografinin kullanımından bahsedilmiştir. Bu bağlamda hem kriptoloji bilimi hem de steganografi teknikleri temel olarak açıklanmış, bunların amaçları ve farkları ifade edilmiştir. Giriş bölümünün son kısmında ise bu bahsedilenlerden hareketle tezin amacı ve önemi açıklanmıştır.

İkinci bölümde kriptoloji ve steganografiyle yapılan çeşitli çalışmalardan bahsedilmiştir. Özellikle bilgi güvenliğinin sağlanmasında bu iki farklı tekniğin etkin kullanımı, örneklenmeye çalışılmıştır.

Üçüncü bölümde ise ilk olarak kriptoloji ve steganografi kavramları detaylı olarak açıklanmıştır. Hem kriptolojinin hem de steganografinin muhtevasında yer alan teknik ve yöntemler ayrıntılarıyla birlikte ifade edilmiştir. Dolayısıyla üçüncü bölümün, kriptoloji ve steganografinin genel literatürünü gerekli detaylarla birlikte açıklayan bir kılavuz olduğunu söyleyebiliriz. Bölümün devamında ise tez çalışması kapsamında tasarlanan mobil uygulamanın geliştirilmesi için kullanılan teknolojilerden bahsedilmiştir. Bu bağlamda Android Studio, Yazılım Geliştirme Kiti (Software Development Kit, SDK), Açık Kaynak Bilgisayar Görüsü (Open Source Computer Vision, OpenCV) kütüphanesi, Genymotion emülatör ve Firebase teknolojileri açıklanmıştır.

Dördüncü bölümde önerilen yaklaşım, blok ve akış diyagramlarıyla beraber detaylı olarak açıklanmıştır. Ayrıca önerilen yaklaşım kullanılarak geliştirilen Android tabanlı mobil mesajlaşma uygulamasından kullanıcı arayüzü görüntüleriyle beraber bahsedilmiştir ve gizli anahtarların cihaz içerisinde nasıl güvenle muhafaza edildiği ifade edilmiştir. Uygulamanın yanı sıra işin sunucu kısmı olan Firebase platformunda verilerin nasıl organize edilip işlendiği ve oluşturulan stego görüntülerin nasıl depolanıp bunlara ne şekilde erişim sağlandığı gibi iletişim sürecine ait tüm detaylar açıkça belirtilmiştir.

Beşinci bölümde ise ilk olarak uygulama üzerinde gerçekleştirilen steganografik işlemlerin başarımları değerlendirilmiştir. Daha sonra simetrik anahtarlı şifreleme algoritmaları, mobil cihazlar üzerindeki işlem zamanları açısından karşılaştırılmıştır. Ayrıca bu algoritmalarla görüntü şifreleme işlemleri yapılmış ve bu işlemler üzerinden

algoritmaların başarımları incelenmiştir. Bunların yanı sıra şifreli görüntülerin histogram ve korelasyon analizleri yapılarak şifreleme işlemlerinin görüntülerin karakteristiğini nasıl değiştirdiği somut verilerle birlikte ifade edilmiştir. Son bölümde ise genel olarak sonuçlar ve öneriler açıklanmıştır.

2. KAYNAK ARAŞTIRMASI

Literatür araştırması yapıldığında kriptoloji ve steganografiyle ilgili çeşitli çalışmalar görülmektedir. Genellikle, bu konularla alakalı algoritmaların incelenmesi karşılaştırılması üzerine olmakla beraber bu konulardan yararlanılarak ağ güvenliği uygulamaları, dosya şifreleme gibi çalışmalar da yapılmıştır. Adli bilişimde, sağlık bilimlerinde, ödeme sistemlerinde, kimliklendirme sistemlerinde vb. alanlarda da kriptoloji ve steganografinin kullanımlarına çeşitli örnekler vardır. Bu bölümde konuyla ilgili literatürdeki makale ve tez çalışmalarından, bunların amaçları ve elde edilen sonuçlardan bahsedilmiştir.

Cvejic ve Seppanen (2002), ses steganografisi için yeni bir En Önemsiz Bit (Least Significant Bit, LSB) metodu önermiştir. Önerilen yaklaşımda LSB ayarlaması için minimum hata değişimi Sinyal Gürültü Oranı (Signal to Noise Ratio, SNR)' nı azaltmak içinse değiştirilmiş hata yayma yöntemi kullanılmıştır. Bu metot, ses dosyalarına bilgi gizlemek amacıyla gizli veri kanalının kapasitesini 132,3 saniye başına kilobit (kb/s)' ten 176,4 kb/s' e çıkarak kapasiteyi %33 artırmıştır. Yapılan analizler önerilen yaklaşımın SNR değerini, düşük kapasiteli standart LSB metoduyla elde edilen SNR seviyesine yakın tutarken veri saklama kapasitesini artırmayı da başardığını göstermiştir.

Chan ve Cheng (2004), yaptıkları makale çalışmasında basit LSB yöntemini iyileştirerek yeni bir veri gizleme şeması önermişlerdir. Basit LSB yöntemiyle elde edilen stego-görüntüye, ekstradan düşük bir hesaplama maliyetiyle optimum piksel ayarı uygulanarak stego-görüntünün kalitesi büyük ölçüde iyileştirilmiştir. Deneysel sonuçlar, stego görüntünün orijinalinden görsel olarak ayırt edilemez olduğunu göstermektedir. Ayrıca önerilen yaklaşımın önceki çalışmalara göre görüntü kalitesi ve hesaplama verimliliği açısından daha başarılı olduğu gözlemlenmiştir.

Gürel, H. (2006), yaptığı tez çalışmasında LSB ve bir piksel içerisine bir Bilgi Değişimi İçin Amerikan Standart Kodlama Sistemi (American Standard Code for Information Interchange, ASCII) kodunun gömülmesi metotlarını inceleyerek transfer edilecek bilgilerin gizliliğin sağlanması için araştırmalar yapmıştır. Bu çalışmasında İletim Denetimi Protokolü / Kullanıcı Veri Bloğu Protokolü (Transmission Control Protocol / User Datagram Protocol, TCP / UDP)' nü kullanarak verilerin kablosuz ağ üzerinden gizlice gönderilmesini hedeflemiştir. Bu hedef doğrultusunda orijinal görüntü üzerinde fark edilir bir değişim olmadan verilerin taşıyıcı görüntüye gömülmesi

sağlanmıştır. İşlem zamanı açısından değerlendirme yapılarak ASCII kod gizleme metodunun en başarılı olduğu sonucuna varılmıştır.

Mousa, A. ve ark. (2006), yaptığı makale çalışmasında Rivest Şifre 4 (Rivest Cipher 4, RC4) şifreleme algoritmasının farklı parametrelerinin algoritmaya etkisini incelemiştir. RC4 algoritmasının performansını göstermek için bu parametrelerde değişikliklere dayanan bazı deneysel çalışmalar yapılmıştır. Çalışma zamanı, anahtar uzunluğu ve dosya boyutunun bir fonksiyonu olarak incelenmiştir. Farklı veri türleri analiz edilmiş ve veri türlerinin etkisi üzerinde durulmuştur. Sonuçlar analiz edilmiş ve yorumlanmıştır. Buna göre; incelenen veriler arasındaki ilişkiyi gösteren matematiksel denklemler ile algoritmanın farklı koşullar altında göstereceği performansın tahmin edilebileceği savunulmuştur.

Nair ve ark. (2011), ağ paketlerinin uzunluklarını değiştirerek veri gizleyen bir ağ steganografi yöntemi önermişlerdir. TCP paket uzunluklarındaki çeşitliliğin az olduğunu belirterek farklı uzunluktaki paketlerle çalışan UDP protokolünün veri gizleme için daha uygun olduğunu savunmuşlardır. UDP tabanlı bir sohbet uygulamasına ait paket uzunluk dağılımının doğası gereği rastgele olduğunu gözlemlemişlerdir. Bunun için bu uygulamanın ağ akışını taklit etmişlerdir. Deneysel çalışmalar için 50 farklı sohbet örneği kullanılmıştır. Yapılan uygulamalarda gizli bilgiler gömüldükten sonra bile normal ağ akışını takip eden bir uzunluk modeli elde edilmiştir. Sonuçlar önerilen yaklaşımın mevcut yöntemlerden daha üstün olduğunu göstermektedir.

Karim, M. ve ark. (2011), yaptıkları makale çalışmasında gizli bilgilerin güvenlik düzeyini iyileştirmek için LSB görüntü steganografi tekniğini geliştiren bir yaklaşım sunmuşlardır. Bu yaklaşımda mevcut steganografi tekniklerinden farklı olarak örtü resimde değişiklik yapılacak pikseller bir gizli anahtar yardımıyla belirlenmektedir. Genellikle LSB tekniklerinde, gizli bilgi örtü resmin spesifik pozisyonlarına saklanmaktadır. Bu nedenle bu teknikleri bilen herhangi biri bu gizli bilgiyi örtü resimden çıkarabilir. Bu yüzden bu çalışmada önerilen yaklaşımla veriler gizli bir anahtara bağlı olarak örtü resmin farklı pozisyonlarına saklanmaktadır. Bu da bilginin geri elde edilmesini zorlaştırmaktadır. Stego resmin kalitesini tespit etmek için Tepe Sinyal Gürültü Oranı (Peak Signal to Noise Ratio, PSNR) ölçütü kullanılmıştır. Önerilen yaklaşım mevcut tekniklere göre daha az sayıdaki pikselde değişiklik yaptığı için mevcut yöntemlere göre PSNR ölçütü daha iyi sonuç vermiştir.

Farah ve ark. (2012), Rivest Shamir Adleman (RSA) ElGamal ve Paillier algoritmalarını şifreleme ve şifre çözme zamanı, verimlilik, şifreli ve şifresi çözülmüş

dosya boyutları açısından karşılaştırmışlardır. Yapılan farklı deneyler sonucunda RSA' nın şifreleme zamanı açısından ElGamal algoritmasının ise şifre çözme zamanı açısından daha başarılı olduğu gözlemlenmiştir. Verimlilik ise herhangi bir algoritmanın performansını göstermek için en önemli parametredir. Elde edilen sonuçlar şifreleme sürecinde RSA'nın, şifre çözme sürecinde ise ElGamal'ın verimliliğinin diğer algoritmalardan daha iyi olduğunu göstermiştir. RSA'nın şifreli veriler için daha az bellek gereksinimine ihtiyaç duyduğu görülmüştür. Deşifre edilmiş dosya boyutlarının ise incelenen üç algoritma için de orijinal dosya boyutuyla aynı olduğu ifade edilmiştir. Genel olarak bu çalışmada incelenen parametreler açısından RSA'nın ElGamal ve Paillier algoritmalarına göre daha başarılı olduğu belirtilmiştir.

Bhowal ve ark. (2013), bilgi güvenliğinin sağlanması için hem kriptografi hem genetik algoritma tabanlı bir ses steganografisi kullanan yeni bir metot önermişlerdir. Bu kapsamda ses steganografisinde kullanılan yer değiştirme tekniğinin problemlerine yönelik bir çözüm önerisi sunulmuştur. Güvenliğin artırılması amacıyla gizli veriler ilk olarak RSA algoritması kullanılarak şifrelenir. Daha sonra da bu veriler genetik algoritma tabanlı LSB yöntemi kullanılarak bir ses dosyasına gizlenmektedir. Şifreli veriler, LSB katmanlarına rastgele olarak gömülür. Bu da dayanıklılığın artmasını sağlarken aynı zamanda gürültü oluşmasına da neden olur. Ama genetik algoritma operatörleri kullanılarak bu gürültü azaltılır ve yapılan optimizasyon sayesinde veri gömülmesi sonrasında taşıyıcı dosyada meydana gelen bozulmalar minimize edilir. Sonuçlar, önerilen yaklaşımın standart algoritmaya göre bit hata oranlarının daha düşük olduğunu ve steganalize karşı daha dayanıklı olduğunu göstermektedir.

Aydoğan M. ve ark. (2013), yaptıkları makale çalışmasında Diş hekimliğinde kullanılan ve Orthopantomogram (OPT) şeklinde isimlendirilen diş radyografilerinin gizlilik ve güvenliğini sağlamak amacıyla bir sistem geliştirmişlerdir. OPT, dişlerin ve çene kemiğinin görüntülenmesini sağlayan bir çeşit röntgendir. Dişlerin, kemik ve dişetlerinin muayene ile incelenemeyen kısımlarını görüntülemeyi sağlayan bir görüntüdür. Çalışmanın temeli hastalara ait bilgilerin önce şifrelenmesi ve sonra da OPT görüntülerine gizlenmesinden oluşmaktadır. Şifreleme için kendileri bir yöntem belirlemişlerdir. Bunun için klavyedeki her bir karaktere karşılık rastgele bir id tanımlanan bir tablo oluşturmuşlardır ve bu id değerine mod işlemi uygulayarak elde ettikleri tam ve kalan kısımları şifreli veri olarak kullanmışlardır. Bu şifreli verileri de OPT görüntüsünde herhangi bir anlam ifade etmeyen ve 0 renk koduna sahip piksellere

gömmüşlerdir. Böylece tıbbi alanda verilerin güvenliğinin sağlanmasında hem şifreleme hem de veri gizleme yöntemlerinin kullanımına bir örnek sunulmuştur.

Amritha ve Varkey. (2013) yaptıkları çalışmada biyometriğe dayalı bir steganografi tekniği uygulamışlardır. Steganografiyi uygulamak için kullandıkları biyometrik özellik görüntülerdeki cilt tonu bölgesidir. Görüntünün tamamını kullanmak yerine verileri yalnızca cilt bölgelerine gömmek verileri gizlemek için oldukça güvenli bir konum sağlar. Aynı zamanda veriler gömülmeden önce RC4 şifreleme algoritmasıyla şifrelenerek güvenlik seviyesi artırılır. Önerilen yaklaşım görünmezlik, kaliteli stego görüntü yüksek güvenlik ve başarılı bir PSNR değeri sağlamıştır.

Padmavathi ve Ranjitha Kumari (2013), ağ üzerinde iletilen verilerin güvenliğini sağlamak için kriptografi ve steganografi kombinasyonu üzerine odaklanmıştır. Bu çalışmada LSB steganografi algoritmasıyla birlikte Veri Şifreleme Standardı (Data Encryption Standart, DES), Gelişmiş Şifreleme Standardı (Advanced Encryption Standart, AES) ve RSA algoritması olmak üzere üç şifreleme tekniği uygulanmaktadır. Bu tekniklerin performansları şifreleme ve şifre çözme işlemi için harcanan zamana ve kullanılan tampon bellek boyutuna göre karşılaştırılmıştır. Deneysel sonuçlar AES algoritmasının şifreleme ve deşifreleme için en az zaman tüketen ve en az tampon bellek kullanan algoritma olduğunu göstermektedir. Ancak RSA algoritması şifreleme için daha fazla zaman tüketmektedir ve tampon bellek kullanımı da çok yüksektir. Simülasyon sonucunda AES algoritmasının DES ve RSA algoritmasından çok daha iyi olduğu gözlemlenmiştir.

Koley ve ark. (2014) hasta bilgilerinin gizliliğini artırmak, güvenli bir şekilde saklanmasını ve iletilmesini sağlamak için kriptografi ve dijital damgalama kombinasyonuyla biyomedikal görüntülerde yeni bir bilgi gizleme yöntemi önermektedir. Şifreleme için RSA algoritması ve bilgileri gizlemek için LSB tekniği kullanılmaktadır. Önerilen yöntemin sonucu, gizlenen bilgilerin taşıyıcı görüntüyü etkilemediğini ve bu bilgilerin bazı gürültülü görüntülerden bile verimli bir şekilde çıkarılabildiğini göstermektedir. Deneysel sonuçlar önerilen algoritmanın farklı saldırı türlerine karşı dayanıklı olduğunu göstermektedir.

Sekhar ve ark. (2015), yetkisiz kişiler tarafından yapılan saldırıları ve bilgi çalmaları önlemek amacıyla Sözde Rastgele Sayı Üreteçleri (Pseudo Random Number Generator, PRNG)'ni kullanarak ağ protokkolleri üzerinde verimli bir şekilde steganografi uygulaması gerçekleştiren yeni bir yöntem önermiştir. Bu yöntemde gizli veriler öncelikle şifrelenmiş ve sonrasında steganografi kullanılarak bir video dosyasının

içerisine gömülmüştür. Bu işlem sonucunda oluşan stego-nesne de tekrar şifrelenmiş ve sonra da sıkıştırılmıştır. Oluşan son dosya da ağ paketlerinin başlık alanlarına yine steganografi uygulanarak gizlenmiştir. Böylece steganografik bir ağ paketleri dizisi oluşturulmuş ve bu dizi ağ üzerinden alıcıya gönderilmiştir. Şifreleme işlemlerinde kullanılan anahtarın üretiminde rastlantısallık uygulanması sayesinde başlıklardan verilerin çıkartılmasındaki karmaşıklık seviyesi de artırılmıştır.

Yan ve Chen (2016), yaptıkları makale çalışmasında saldırganlar tarafından anahtarların kolayca elde edilebildiği tipik genişletme algoritmalarının dezavantajlarının üstesinden gelmek için AES algoritmasının geliştirilmiş bir halini sunmuşlardır. Bu amaçla alt anahtarların üretimini daha güvenli hale getirmek için iyileştirilmiş bir anahtar genişletme algoritması önerilmiştir. Önerilen yaklaşımının başarımını değerlendirmek için şifreleme işlemindeki difüzyon karakteristiğinin gelişimi analiz edilmiştir. Bunun için 128 bit uzunluğundaki düz metinler üzerinde her defasında daha fazla bit değişikliği yapılarak şifreleme işlemleri gerçekleştirilmiş ve yapılan bu değişimlerin şifreli metni ne kadar etkilediği incelenmiştir. Buna göre düz metinlerde yapılan 3 bitlik ve 4 bitlik değişim sonucunda standart AES algoritmasıyla elde edilen şifreli metinlerdeki ortalama değişimin sırasıyla 60,5 ve 61 bit olduğu, önerilen yaklaşımla oluşturulan şifreli metinlerdeki ortalama değişimin ise sırasıyla 64 ve 65 bit olduğu tespit edilmiştir. Bu sonuçlarda önerilen algoritmanın standart AES' e göre daha kararlı bir difüzyon karakteristiği ortaya koyduğunu göstermiştir. Ayrıca önerilen yaklaşım işlem zamanı açısından da değerlendirilmiş ve geliştirilen algoritmanın standart AES'e çok yakın bir performans gösterdiği görülmüştür. Sonuç olarak iyileştirilmiş AES algoritmasının verimliliği azaltmadan difüzyon ve güvenliği artırdığı gözlemlenmiştir.

Ahamad ve Abdullah (2016), kriptografi algoritmalarının multimedya içerikleri üzerinden performanslarını değerlendirmiştir. Şifreleme algoritmaları olarak RSA, AES, Blowfish ve Özel Veya (Exclusive Or, XOR,) multimedya olarak ise metin, fotoğraf, video ve ses ortamları kullanılmıştır. Simülasyon sonuçları AES algoritmasının diğer algoritmalardan daha başarılı olduğunu göstermiştir. Blowfish ve XOR' un ortalama bir performansa sahip olduğu gözlemlenmiştir. Blowfish, metin ve fotoğraf multimedya içerikleri için ikinci en iyi algoritma olarak tespit edilmiştir. XOR algoritmasının ise ses ve video ortamları için iyi performans gösterdiği görülmüştür. Asimetrik bir yöntem olan RSA' nın ise en çok zaman tüketen algoritma olduğu, simetrik algoritmaların asimetrik algoritmalardan daha az işlem zamanı harcadığı ve bu yüzden multimedya içeriklerin transferinde simetrik anahtarlı algoritmaların kullanılması gerektiği savunulmuştur.

Shakir (2016), yaptığı tez çalışmasında bilgi güvenliğinin sağlanmasında bir kriptografi ve steganografi entegrasyonunu anlatmıştır. Kriptografi açısından DES algoritmasının iyileştirilmiş bir hali uygulanmışken steganografide de LSB algoritması kullanılmıştır. Elde edilen sonuçlar, PSNR, SNR ve Ortalama Kare Hatası (Mean Squared Error, MSE) açısından değerlendirilmiştir. Dört farklı resim üzerinde veri gizleme işlemi yapılmış ve bu işlem sonucunda SNR değerleri 46,01 ile 47,40 dB (desibel) arasında, PSNR değerleri 46,02 ile 48,99 dB arasında ve MSE değerleri ise 0,023 ile 0,045 arasında değişim göstermiştir. Ayrıca çalışma kapsamında önerilen iyileştirilmiş DES algoritması ile normal DES algoritması da şifreleme ve deşifreleme performansı açısından test edilmiş ve önerilen iyileştirilmiş DES algoritmasının mevcut DES algoritmasına göre daha iyi sonuçlar verdiği görülmüştür. Ek olarak DES ve LSB' nin birleştirilmesiyle iki katmanlı güvenlik koruması sağlanacağı savunulmuştur.

Yasin (2017), yaptığı tez çalışmasında RC4 şifreleme algoritması ve Haar dalgacık dönüşümü kullanılarak gerçekleştirilen bir görüntü şifreleme şeması önermiştir. Görüntünün şifrelenip güvenliğinin sağlanması için RC4 algoritması sıkıştırılması içinse Haar dalgacık dönüşümü uygulanmıştır. Farklı görüntüler üzerinde yapılan deneysel uygulamalarda RC4' le yapılan şifreleme işlemi sonucunda görüntüdeki değişen piksel sayısı oranı %99,99 ile %99,98 arasında, görüntüdeki yoğunluk değişimi oranı ise %24,94 ile %40,39 arasında olmuştur. Bunlara ek olarak yapılan korelasyon analizi sonucunda ise orijinal görüntülere ait katsayıların 0,8665 ile 0,9767 arasında şifrelenmiş görüntülere ait katsayıların ise 0,0013 ile 0,0001 arasında değiştiği gözlemlenmiştir.

Gökrem ve ark. (2017), yaptıkları makale çalışmasında şifreli mesajlaşma yoluyla kişilerin iletişimini sağlayan bir mobil uygulama geliştirmişlerdir. Bu uygulamada, Çok Parçalı Sezar Şifreleme (Multi Fragmented Caesar Encryption) algoritması kullanılmıştır. Uygulama Android cihazlar için geliştirilmiştir. Bu uygulamada iki kullanıcının paylaşılan bir şifreleme anahtarı aracılığıyla birbirlerine güvenli bir şekilde mesaj göndermesi ile bilgi güvenliği sağlanmaya çalışılmıştır.

Yerlikaya ve Gençoğlu (2017), yaptıkları makale çalışmasında RSA algoritmasının yüksek hesaplama maliyetine sahip olmasından dolayı sınırlı kaynaklara sahip mobil cihazlarda doğrundan kullanımının uygun olmayacağını savunarak RSA'yı optimize etmişlerdir. Bu amaçla RSA algoritmasındaki modüler üs alma işlemini daha efektif hale getirmişlerdir. Önerdikleri yaklaşımı farklı platformlar üzerinde 1023 bit ve 1395 bitlik iki farklı şifreleme anahtarı kullanarak 65 bitle 865 bitlik iki farklı düz metni şifrelemek için uygulamışlardır. Bu uygulamaların işlem zamanlarını hesaplamışlar ve

önerilen yaklaşımla şifreleme sürecinin kullanılan platform veri ve anahtar boyutuna bağlı olarak 0,19 ile 1,55 sn arasında gerçekleştiğini gözlemlemişlerdir.

Hussain M. ve ark. (2018), yaptıkları makale çalışmasında belirli alanda görüntü steganografi tekniklerinin son 5 yıllık bir literatür taramasını sunmaktadır. Bu makalenin amacı, kapsamlı bir çalışma sağlamak ve görüntü steganografik sistemlerin tasarımında yer alan araştırmacılar için mevcut güncel tekniklerin artılarını ve eksilerini vurgulamaktır. Bu makalede steganografik sistemin genel yapısı ile görüntü steganografik tekniklerinin sınıflandırılması ve bunların belirli alanlardaki özellikleri araştırılmıştır. Ayrıca, farklı performans metrikleri ve steganaliz tespit saldırıları da incelenmiştir. Bu kapsamda incelenen tekniklerle alakalı uygulamalar yapılmış ve öneriler sunulmuştur.

Kodal, Sevindir, H. ve ark. (2018), yaptıkları makale çalışmasında bilgi güvenliğinin sağlanması amacıyla görsel kriptografi uygulamışlardır. Dalgacık dönüşümü kullanılarak veriler şifreli parçalara bölünmüştür. Daha sonra bu parçalar farklı görüntülere gizlenerek veriler fark edilemez hale getirilmiştir. Verilerden parçaların elde edilmesi için Shamir algoritması kullanılmıştır. Seçilen kapak görüntülerinin yeşil katmanına dalgacık dönüşümü uygulandıktan sonra elde edilen alt bantlara parçalar gömülerek veriler gizlenmiştir. Elde edilen stego-resimler, kapak resimlerle karşılaştırılarak PSNR değerleri hesaplanmıştır. Sonuç olarak ikili görüntülerde görüntüler arasında herhangi bir piksel kaybı olmamıştır. Format değişikliğinden ve piksel değer aralığından kaynaklı olarak Kırmızı Yeşil Mavi (Red Green Blue, RGB) formatı ile elde edilen görüntüler arasındaki PSNR değeri ise 15,8123 dB olarak elde edilmiştir.

Atalay, N. ve ark. (2019), yaptıkları makale çalışmasında kriptoloji algoritmalarının performansını görüntü şifreleme uygulamaları üzerinden değerlendirmişlerdir. Bu kapsamda DES, AES ve RC4 algoritmaları üzerinde durulmuştur. Bu algoritmalarla şifrelenen görüntüler histogram analizi, korelasyon analizi, diferansiyel saldırı analizi, anahtar uzay (alan) analizi, anahtar hassasiyet analizi ve zaman karmaşıklığı analizi performans metrikleri kapsamında incelenmiştir. Yapılan analizler neticesinde AES'in orijinal görüntü üzerinde gerçekleştirdiği %99,63' lük değişim oranıyla görüntü şifreleme uygulamaları için en başarılı algoritma olduğu gözlemlenmiştir.

Darbani ve ark. (2019), Birleşik Fotoğraf Uzmanları Grubu (Joint Photographic Experts Group, JPEG) formatlı görüntüler için bir steganografi yöntemi önermiştir. JPEG

sıkıştırma prosedüründe frekans değerlerinin ayrıştırılmasından sonra verilerin bir kısmının kaybolabilmesinden dolayı gömülecek mesajlar ayrıştırma aşamasından sonra görüntüye eklenmiştir. Bu amaçla görüntüdeki her pikselin iki en anlamsız biti veri gömmek için kullanılmıştır. Gömülecek mesajdaki bit değerlerine göre piksel bitleri, artırmak ve azaltmak suretiyle değiştirilmiştir. Gerçekleştirilen uygulamanın performansı gömülebilecek maksimum mesaj kapasitesi ve PSNR kriterlerine göre değerlendirilmiştir ve yapılan çalışma, literatürdeki iki farklı çalışmayla karşılaştırılmıştır. Sonuçlar önerilen yaklaşımın diğer çalışmalara göre hem daha yüksek PSNR değeri sağladığını hem de görüntülere daha fazla gizli veri gömme imkanı sunduğunu göstermiştir. Genel olarak önerilen yaklaşımın, stego-görüntünün kalitesinin neredeyse orijinaline benzer olmasını sağlarken veri saklama kapasitesini de artırdığı gözlemlenmiştir.

Goyal ve ark. (2019), yaptıkları makale çalışmasında IoT platformundaki kısıtlı cihazlar için uygun ve hafif özelliklere sahip iyileştirilmiş Present blok şifreleme algoritmasını önermiştir. Alanda Programlanabilir Kapı Dizileri (Field Programmable Gate Array, FPGA)' nden Uygulamaya Özel Tümleşik Devre (Application Specific Integrated Circuit; ASIC)'lere kadar Present şifreleme algoritmasının çeşitli platformlarda farklı uygulamaları açıklanmıştır. Güçlendirilmiş tasarımların hafif özellikler gösterdiği ve düşük kaynaklı cihazlar için uygun olduğu gözlemlenmiştir. Bu çalışmada da S-kutularının iyileştirilmesiyle Present algoritması güçlendirilmiş ve optimize edilmiştir. Sonuçlar, tasarımda kullanılan devre birimlerinin toplam sayısında önemli bir azalma olduğunu göstermiştir. Ayrıca tasarım için düşük çip alanı ayrılsa bile önerilen Present algoritması yeterli derecede güvenlik sağlamaktadır. Yapılan analizlerde güvenlik, alan, güç tüketimi ve verim açısından 128 bit anahtarlı Present algoritmasının diğer algoritmalardan üstün olduğu gözlemlenmiştir.

Forgae ve Oekay (2019), simetrik şifreleme için Evrişimli Sinir Ağları (Convolutional Neural Networks, CNN)' nın kullanımını gerçekleştirilen bir çalışma yapmışlardır. Gerçekleştirilen şifreleme uygulaması işlem zamanı, bellek kullanımı ve işlemci yükü üzerinden AES algoritmasıyla karşılaştırılmıştır. Bunun yanı sıra CNN şifreleme uygulamasının dayanıklılığının çalışma kapsamı dışında olduğunu ifade ederek gerçekleştirilen uygulamayı kripto ataklara karşı test etmemişlerdir. AES algoritmasının yüksek hızlı ve düşük bellek gereksinimine sahip olmasının yanı sıra yıllardır uygulamaları geliştirilen ve analiz edilen bir algoritma olmasından dolayı algoritmaya yönelik belgelenmiş ve yayınlamış birçok saldırı türü olduğunu belirtmişlerdir. Buna karşılık CNN' ye yönelik bir kripto analiz yönteminin bulunmadığını savunmuşlardır.

Deneysel sonuçlar bu çalışmada belirtilen şartlar altında CNN' nin şifreleme uygulamaları için bir potansiyel taşıdığını göstermiştir.

Balcı, D. (2019), yaptığı tez çalışmasında, hastalarla ilgili kişisel bilgiler, teşhis raporları ve tedaviyle ilgili verileri, güvenliklerinin sağlanması için bir steganografi yöntemi uygulayarak MR görüntülerine saklamıştır. Önerilen yaklaşımda, özer, şifreleme ve steganografi algoritmalarının bir kombinasyonu gerçekleştirilmiştir. Şifreleme algoritması olarak AES, steganografi algoritması olarak da LSB yöntemi kullanılmıştır. Önerilen yaklaşımın değerlendirilmesi için farklı büyüklükteki MR görüntülerine hasta bilgileri gizlenmiş ve elde edilen görüntüler, PSNR, MSE ve Yapısal Benzerlik (Structural Similarity, SSIM) değerleri üzerinden analiz edilmiştir ve literatürdeki diğer algoritmalar tarafından sağlanan değerlerle karşılaştırılmıştır. Bu kapsamda mevcut yöntemlerin PSNR değerleri ortalama olarak 69,64 dB ile 72,59 dB arasında değişirken önerilen hibrit yöntemin PSNR değeri 72,63 dB olarak hesaplanmıştır. Karşılaştırma sonuçlarına göre, önerilen yöntem yüksek PSNR ve SSIM değerlerine sahipken düşük MSE değerlerine sahiptir. Elde edilen sonuçlar önerilen yaklaşımın LSB yöntemini iyileştirdiğini göstermiştir. Ayrıca önerilen sistemde şifrelemeye ek olarak özet fonksiyonlarının kullanımıyla MR görüntülerine gizlenen verilerin doğruluğundan ve bütünlüğünden emin olunabileceği savunulmuştur.

Sharma ve ark. (2019), görüntü steganografisi için önceki çalışmalara göre çok daha fazla güvenli olan yeni bir metot önermiştir. Bu çalışmada bir görüntüyü başka bir görüntünün içerisine gizlemek için steganografi, kriptografi ve sinir ağları birlikte kullanılmıştır. Yaygın steganografi teknikleri kullanılmasına rağmen bunların kriptografi ve sinir ağları ile entegre edilmesi önerilen yöntemi saldırılara karşı daha dayanıklı yapmıştır. Ayrıca ek olarak şifreleme uygulanması taşıyıcı görüntünün erişebilir olması durumunda bile gizli bilginin güvende kalmasını sağlamaktadır.

Cihangir, S. (2020), yaptığı tez çalışmasında XOR tabanlı iyileştirilmiş LSB steganografi yöntemini tasarlamış ve gerçekleştirmiştir. Uygun PSNR ve MSE değerlerine sahip, olabildiğince az bozulmuş stego-görüntüler elde etmek için XOR işleminin doğal avantajı kullanılmıştır. Bu işlem piksellerin her renk katmanındaki en anlamsız bitlerin XOR' lanmasıyla gerçekleştirilmiş ve böylece iki gizli veri biti sadece bir renk katmanındaki tek bitlik değişimle görüntüye gizlenmiştir. Deneysel çalışmalarda hem klasik hem de iyileştirilmiş LSB yöntemleri kullanılarak 200 X 200 piksellik Lenna görseline 80.000 bit boyutunda gizli mesaj gömülmüştür. Bu işlem sonucunda klasik ve iyileştirilmiş yöntemle elde edilen stego-görüntülerin PSNR değerleri sırasıyla 52,9025

dB ve 54,1652 dB olarak hesaplanmıştır. Bu değerler de önerilen yaklaşımın klasik LSB metodundan daha başarılı olduğunu ispatlamıştır.

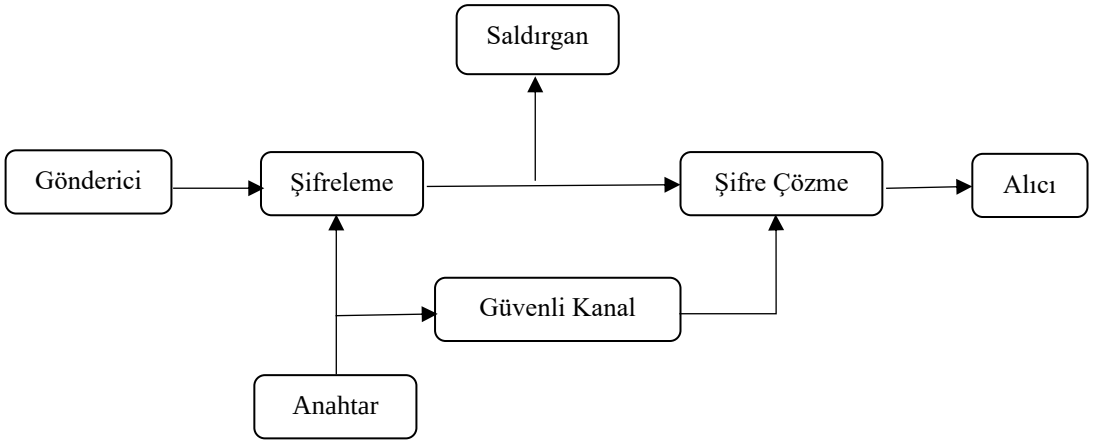
Kao ve ark. (2020), yaptıkları makale çalışmasında, Sensör Ağları için Mesaj Kuyruğu Telemetri Aktarımı (Message Queuing Telemetry Transport for Sensor Networks, MQTT-SN) tabanlı güvenli bir bağlantı ve uçtan uca şifreli iletim yöntemi sunmuştur. MQTT-SN' nin güvenli bir versiyonunu gerçekleştirmek amacıyla dijital imzalama için Eliptik Eğri Dijital İmzalama Algoritması (Elliptic Curve Digital Signature Algorithm, ECDSA), anahtar değişimi için Eliptik Eğri Diffie Hellman (Elliptic Curve Diffie Hellman, ECDH) ve kimliği doğrulanmış şifreleme için ChaCha20-Poly1305 algoritmasını kullanmışlardır. Bu algoritmaların uygulanması, güvenli MQTT-SN'nin sınırlı bilgi işlem gücüne sahip sensör düğümlerinde daha iyi performans göstermesini sağlamıştır. Zedboard geliştirme kartı üzerinde yapılan analizlerde Taşıma Katmanı Güvenliği için Mesaj Kuyruğu Telemetri Aktarımı (Message Queuing Telemetry Transport for Transport Layer Security, MQTT-TLS)' nin anlaşma zamanı 56,12 ms olarak hesaplanırken bu çalışmada sunulan güvenli MQTT-SN'nin anlaşma zamanı 24,78 ms olarak hesaplanmıştır.

3. MATERYAL VE YÖNTEM

Bu bölümde ilk olarak çalışma kapsamında önem arz eden kriptoloji ve steganografiden detaylı olarak bahsedilmiştir. Çünkü bu iki kavram çalışmanın temelini oluşturmaktadır. Bundan dolayı kriptoloji ve steganografinin muhtevasında yer alan algoritmalar, yöntem ve teknikler tüm yönleriyle ele alınmış, birbirleriyle karşılaştırılmış avantajları ve dezavantajları ifade edilmiştir. Yapılan bu inceleme ve değerlendirme neticesinde çalışmanın amacına en uygun olan algoritma ve yöntemler belirlenmiştir. Bölümün devamında Android tabanlı bir mobil uygulamanın geliştirilmesi, bu uygulamada kullanılacak verilerin depolanması ve gerçek zamanlı iletişim kurulması için gerekli araçlar incelenmiştir. Materyal olarak, bu konularla ilgili yayımlanmış makaleler, tezler, bilimsel çalışmalar, mobil uygulamanın geliştirilmesi için gerekli olan Android Studio, SDK, gerekli kütüphaneler, Genymotion emülatör, Java programlama dili, Firebase gerçek zamanlı veritabanı ve Firebase depolama hizmetleri, bunların kullanılabilmesi için Google ve Firebase ’in dokümantasyonları söylenebilir. Bu materyaller açıklandıktan sonra bahsedilen araçların, algoritma ve tekniklerin çalışmanın amacına uygun olarak nasıl bir araya getirildiği ve etkin bir sonuç elde etmek için bunların kullanılmasında nasıl bir yöntem izlendiği ifade edilmiştir.

3.1. Kriptoloji

Kriptoloji güvenlik, gizlilik, kimlik denetimi, bütünlük gibi bilgi güvenliği kavramlarını sağlamak için çalışan matematiksel yöntemler bütünüdür. Bu yöntemler, bir bilginin iletimi esnasında meydana gelebilecek saldırılardan bilgiyi, bilgi göndericisini ve alıcısını koruma amacı taşır. Bir başka ifadeyle kriptoloji, okunabilir durumdaki bir bilginin istenmeyen taraflarca okunamayacak bir hale dönüştürülmesinde kullanılan teknik ve yöntemlerin tamamıdır. Bu açıdan kriptoloji, kişiler arası veya özel/kamu kurum ve kuruluşlarındaki haberleşme sistemlerinde başta olmak üzere gizlilik ve güvenliğin gerekli olduğu her yerde kullanılmaktadır (Özyılmaz, 2014). Şekil 3.1’de bir şifreleme sisteminde olması gereken temel bileşenler gösterilmiştir.

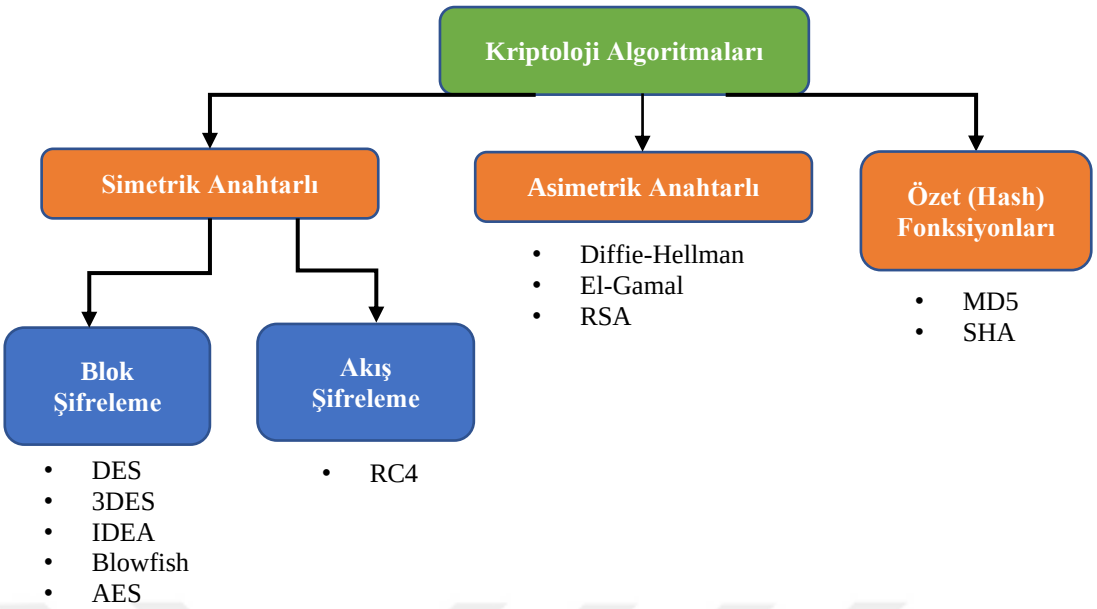


Şekil 3.1. Temel bir şifreleme sistemi

Bir şifreleme sistemi, düz metinlerden, şifreli metinlerden, anahtarlardan ve algoritmalardan oluşur. Şekil 3.1' de görüldüğü üzere kriptolojinin amacı güvensiz bir kanal üzerinden alıcı ve gönderici arasında herhangi bir üçüncü kişinin anlayamayacağı biçimde iletişim kurabilmeyi sağlamaktır. Örneğin bu kanal bir network veya bluetooth gibi herhangi bir iletişim hattı olabilir. Kriptoloji algoritması ise şifreleme ve şifre çözümü için kullanılan matematiksel işlemler topluluğudur.

Bir kriptoloji sisteminde gönderilmek istenen bilgi yani açık metin(plaintext), düz bir metin olabileceği gibi ses, fotoğraf, video gibi herhangi bir sayısal veri de olabilir. Gönderici önceden belirlediği bir anahtar kullanarak göndermek istediği bilgiyi yani açık metni şifreler ve kanal üzerine şifre metin olarak gönderir. Saldırgan, herhangi bir şekilde kanal üzerindeki bu şifrelenmiş metni ele geçirirse bunun açık metin halini yani gönderilen halini anlayamaz fakat şifreleme anahtarını önceden bilen alıcı, şifrelenmiş metni deşifre eder ve açık metne ulaşır.

Alıcı ve gönderici arasında bu protokol, belli bir kriptoloji algoritması kullanılarak çalıştırılır. Hepsinin kendine göre avantajı ve dezavantajı bulunan birçok kriptoloji algoritması mevcuttur. Bunlar belirli ölçütlere göre sınıflandırılmış ve farklı gruplara ayrılmıştır. Bu sınıflandırmadaki en temel ölçüt ise anahtar kullanımıdır Şekil 3.2' de bu sınıflandırma gösterilmiştir.



Şekil 3.2. Kriptoloji algoritmalarının sınıflandırılması

Kriptoloji algoritmaları temelde simetrik ve asimetrik şifreleme algoritmaları ile özet (hash) fonksiyonları olmak üzere üçe ayrılmaktadır. Simetrik (Gizli Anahtarlı) şifreleme algoritmaları, şifreleme ve şifre çözme işlemleri için aynı anahtarı kullanır. Gizli veri alışverişi yapacak kişi veya uygulamalar simetrik anahtarı kendi aralarında, emniyetli bir şekilde değiştirmelidir. Simetrik şifreleme algoritmasıyla şifrelenmiş bir verinin güvenliği, şifreleme işleminde kullanılmış olan anahtarın gizliliği ile doğrudan ilişkilidir.

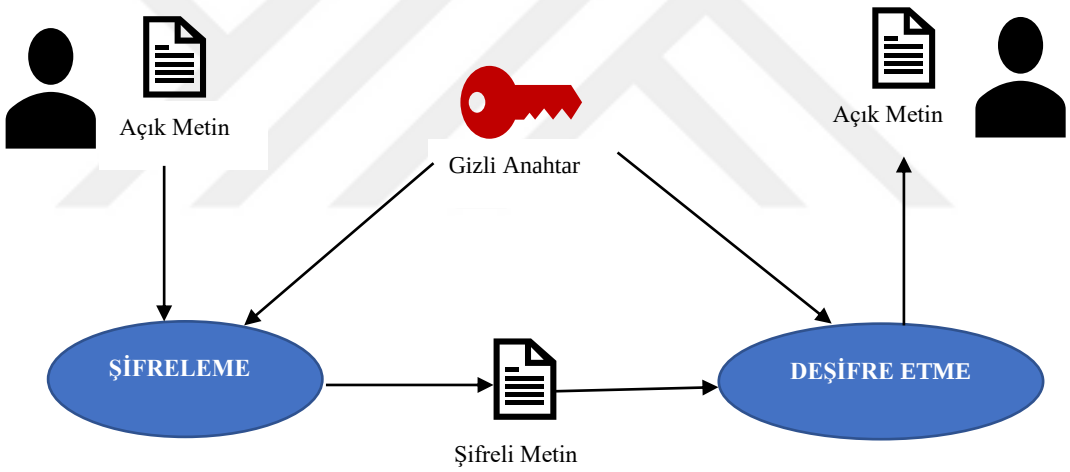
Simetrik anahtarlı algoritmalar da kendi içerisinde blok şifreleme ve akış şifreleme olmak üzere ikiye ayrılır. Akış şifreleme algoritmaları, belli bir anda bir bitlik acık metni şifreler, blok şifreleme algoritmaları ise açık metni blok olarak adlandırılan bit gruplarına bölerek işler (Mohan ve Reddy, 2011). Asimetrik anahtarlı sistemlerde ise biri gizli biri açık olmak üzere iki anahtar mevcuttur. Herkes tarafından bilinen açık anahtar şifreleme işleminde kullanılırken sadece alıcı tarafından bilinen gizli anahtar ise şifre çözme işleminde kullanılır.

Kriptolojinin esas bölümlerinden biri daha olan özet (hash) fonksiyonları ise şifreleme amaçlı değil, veri bütünlüğünün sağlanması amacıyla kullanılır. Bunlar özetleme işlemi yapar ve bu fonksiyonların deşifrelenmesi yoktur. Kısaca bahsedilen bu sınıflandırma, bölümün devamında daha detaylı olarak açıklanmıştır. Her bir grubun veya sınıfın özellikleri, avantaj ve dezavantajlarıyla birlikte bu gruplardaki popüler olan algoritmalarla beraber tek tek ifade edilmiştir.

3.1.1. Simetrik (Gizli anahtarlı) algoritmalar

Simetrik anahtarlı algoritmalar, aynı zamanda gizli anahtar algoritmaları, tek anahtar algoritmaları veya bir (aynı) anahtar algoritmaları diye de adlandırılır. Gönderici ile alıcının güvenle iletişime başlamadan önce bir anahtar üzerinde anlaşmalarını gerektirir. Bir simetrik algoritmanın güvenliği anahtara dayanır; anahtarın açığa çıkması herkesin mesajları şifreleyebileceği ve çözebileceği anlamına gelir. İletişimin gizli kalması gerektiği sürece, anahtar gizli kalmalıdır (Soyalıç, 2005).

Şekil 3.3' de simetrik şifreleme işlemi temel olarak gösterilmiştir. Görüldüğü üzere gönderici bir gizli anahtar kullanarak açık metni bir şifreleme algoritmasıyla şifreler ve bu şifreli metni herhangi bir kanal vasıtasıyla alıcıya iletir. Alıcı da kendisine gelen şifreli metinden yine aynı gizli anahtar yardımıyla şifreleme algoritmasını tersine çalıştırarak açık metni elde eder.



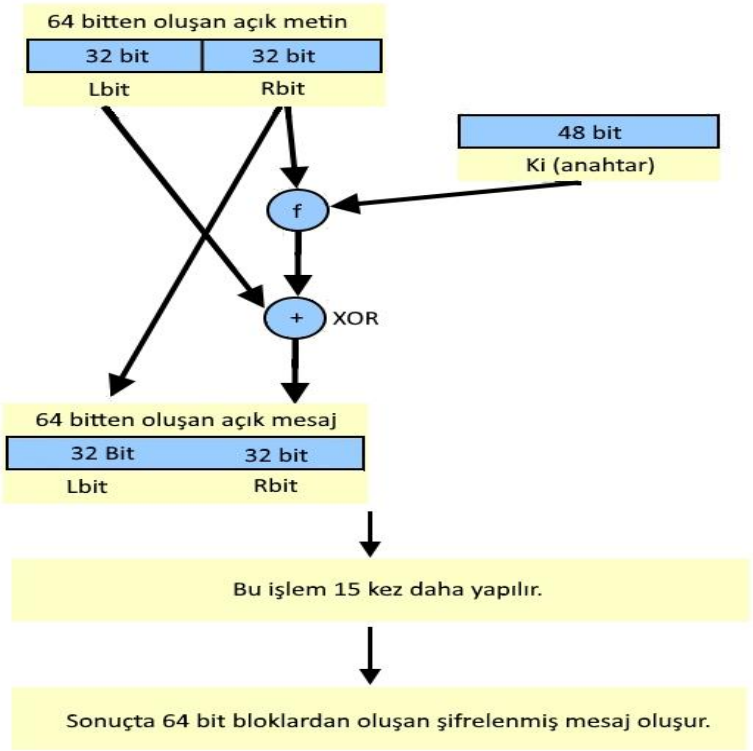
Şekil 3.3. Simetrik şifreleme modeli

Simetrik şifreleme algoritmaları kendi içerisinde de akış ve blok şifreleme olarak ikiye ayrılmaktadır. Akış şifreleme algoritmalarında, arka arkaya akarak gelen harf ya da bitlerin her seferinde tek biri işlenecek şekilde şifreleme yapılır. Gömülü sistemlerde ve cep telefonları gibi anlık iletişimin önemli olduğu yerlerde kullanılır (Ahmad ve ark., 2010). Blok şifreleme algoritmalarında ise düz metin blok adı verilen sabit uzunluklu parçalara bölünür ve her parça ayrı ayrı şifrelenip sonra tekrar bir araya getirilir. Bölümün devamında bu algoritmalardan ve özelliklerinden ayrı ayrı bahsedilmiştir.

3.1.1.1. Data Encryption Standart (DES) algoritması

Uluslararası İş Makineleri (International Business Machines, IBM) tarafından geliştirilen DES, Ulusal Güvenlik Ajansı (National Security Agency, NSA)' nın üzerinde uyguladığı değişikliklerden sonra bir şifreleme standardı olarak kabul edilmiştir. DES algoritmasında bilinen Feistel mimarisi kullanılmış ve şifreleme işlemi ile deşifreleme işlemi aynı şekilde uygulanmıştır (Şahin, 2015). Diğer modern şifreleme algoritmalarında olduğu gibi mesaj bitlere çevrilerek işlem yapılır ve blok şifreleme yapılır, yani veriler bir anahtar yardımıyla bloklar halinde şifrelenir. Şifreyi çözmenin zorluğu anahtar uzunluğuna bağlıdır. DES algoritmasında anahtar uzunluğu 64 bittir. Bu anahtar özellikle günümüz işlemci hızları göz önüne alındığında, kaba kuvvet (brute force) saldırılarına karşı zayıf kalmaktadır. Kısaca DES; öncelikle açık metni blok olarak adlandırılan sabit uzunluktaki parçalara böler, sonra da oluşan bütün parçaları ayrı ayrı şifreler. Sonuçta şifreli metni deşifre edebilmek için ise süreci tekrar ederek yine bölünmüş olan bloklar üzerinde bağımsız ve ayrı ayrı işlem yapar. Oluşturulan bloklar 64 bit uzunluğundadır (Furht ve ark., 2005).

Şekil 3.4' de görüldüğü üzere algoritmada ilk olarak 64 bitlik açık metin bloğu sol ve sağ 32 bitlik iki kısma ayrılır. Sağdaki 32 bit doğrudan sol 32 biti oluşturur. Aynı zamanda sağdaki 32 bit ile 64 bitlik anahtarın 48 bitlik bir kısmı ile özel bir fonksiyona girer, sonra çıkan sonuç soldaki 32 bit ile XOR lanır ve sağ 32 bitlik kısım ortaya çıkar. Bu işlem 16 kez tekrarlanır ve böylece ilk blok şifrelenmiş olur (Furht ve ark., 2005). Diğer bloklara da aynı işlemler uygulanarak metnin tamamı şifrelenir. Çıkan sonuç aynı algoritmaya tekrar girdi olarak verilirse de açık metne ulaşılır. DES algoritmasındaki anahtar uzunluğu 64 bit olmasına rağmen, 56 bitlik kısmı kullanılmaktadır. Her turda o kullanıma özel farklı bir anahtar oluşturması DES algoritmasının avantajlarındandır. Fakat günümüz teknolojisinde algoritmanın yavaş kalması ve kullandığı 56 bitlik anahtarın güvenlik ihtiyacını karşılayamaması da DES'in dezavantajlı tarafıdır.



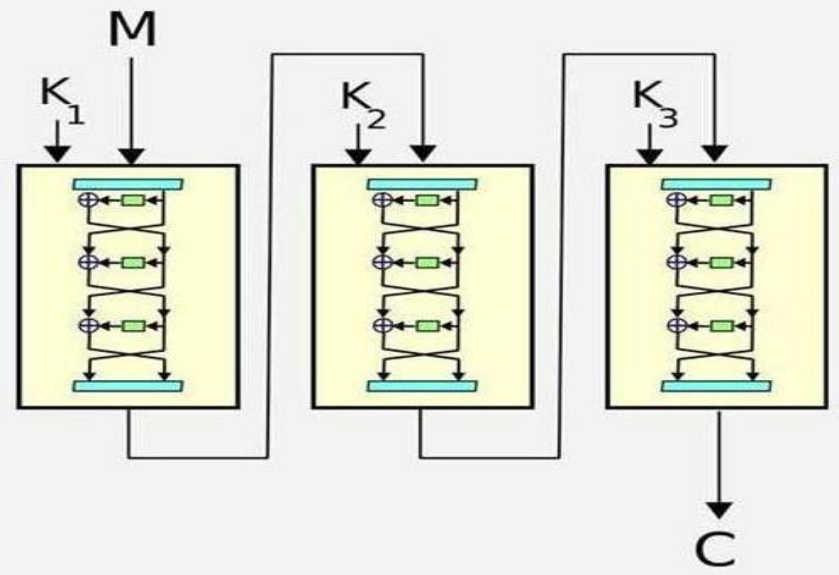
Şekil 3.4. DES algoritması blok diyagramı

DES zayıf yönleri dolayısıyla günümüzde önemini yitirmiştir ve yerini Üçlü Veri Şifreleme Standardı (Triple Data Encryption Standart, 3DES) olarak da adlandırılan daha güvenlikli bir algoritmaya bırakmıştır. 3DES algoritması; DES algoritmasının ardı ardına 3 defa çalıştırılması şeklinde uygulanmaktadır. Bu yüzden DES algoritmasına göre 3 kat daha yavaştır. 3DES algoritması şifreleme için 192 bitlik anahtar kullanır. 3DES algoritması da zamanla DES algoritması gibi günümüz teknolojisine ayak uyduramadığından ve yavaş kaldığından dolayı yerini, kendisine göre 6 kat daha hızlı çalışan AES algoritmasına bırakmıştır (Şahin, 2015).

3.1.1.2. Triple Data Encryption Standart (3DES) algoritması

3DES algoritması, DES algoritmasının arka arkaya üç defa çalıştırılması ile elde edilmiştir. Bu yüzden DES algoritmasına göre daha güvenlidir. 3DES algoritmasında iki ayrı anahtar kullanılmaktadır. İki anahtar kullanılması Brute Force (Kaba Kuvvet) saldırılarını önlemek için yeterli olmaktadır. Böylece DES algoritmasına göre iki kat daha güvenli çalışmaktadır (Chandra ve ark., 2014). Şekil 3.5’ de algoritmanın çalışma mekanizması gösterilmiştir. Üçlü-DES, iki tane 56-bitlik anahtar kullanır. Metin önce ilk

anahtar kullanılarak DES algoritması ile şifrelenir. İkinci anahtar ile şifrelenmiş metin üzerinde DES' in çözme algoritması uygulanır. İkinci anahtar çözmek için doğru bir anahtar olmadığından, bu çözme işlemi veriyi daha karmaşık bir hale getirecektir. İki kez karıştırılmış mesaj, son olarak ilk anahtar kullanılarak tekrar şifrelenir ve böylece şifreli metin elde edilir.



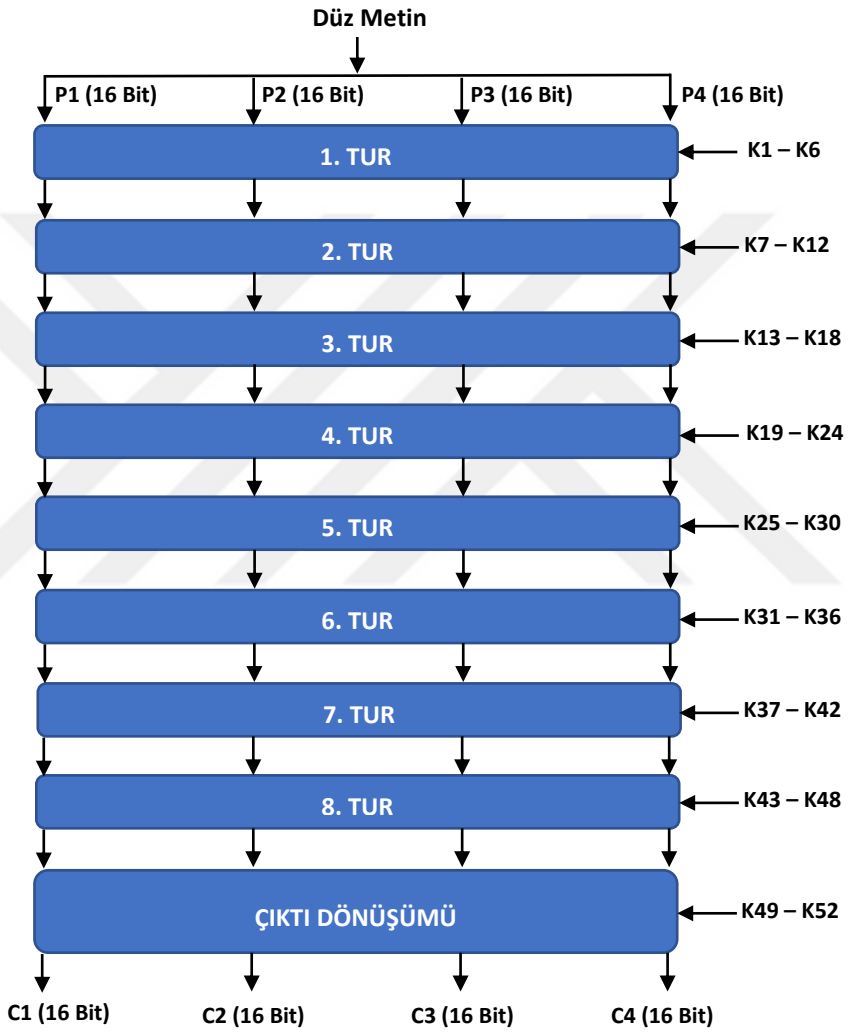
Şekil 3.5. 3DES algoritmasının blok diyagramı

Algoritma üç aşamadan meydana geldiği için Üçlü-DES olarak adlandırılmıştır. 3DES algoritması, DES algoritmasının art arda 3 defa çalıştırılması şeklinde uygulandığından DES algoritmasına göre 3 kat yavaş çalışır (Abood ve Guirguis, 2018). Algoritmanın güvenliği kullanılan anahtara bağlıdır. Anahtar ne kadar zayıfsa, algoritma da saldırılara o kadar açık hale gelir. 3DES, sonrasında geliştirilen ve daha başarılı olan AES algoritmasına göre 6 kat daha yavaş çalıştığından günümüzde tercih edilmemektedir.

3.1.1.3. International Data Encryption Algorithm (IDEA)

Uluslararası Veri Şifreleme Algoritması (International Data Encryption Algorithm, IDEA), İsviçre Federal Teknoloji Enstitüsü'ndeki Xuejia Lai ve James L. Massey tarafından tasarlanan ve ilk defa 1991'de tanıtılan bir blok şifreleme algoritmasıdır. Orijinal algoritma birkaç değişiklik geçirmiştir ve nihayetinde IDEA olarak adlandırılmıştır. Bahsedilen algoritma 64 bit düz metin ve şifreli metin bloğu

üzerinde çalışmaktadır. Şifreleme için 64-bit düz metin, dört adet 16 bitlik alt bloğa bölünür. Bu blokların her biri 8 turluk döngünden ve bir çıkış dönüşümü fazından geçer. Bu sekiz turun her birinde, bazı aritmetik ve mantıksal işlemler yapılır. 8 tur boyunca aynı ardışık işlemler tekrarlanır. Son aşamada, yani bir başka deyişle çıkış dönüşümünde sadece aritmetik işlemler yapılır (Basu, 2011). Şekil 3.6' da IDEA algoritmasının genel çalışma mekanizması gösterilmiştir.



Şekil 3.6. IDEA algoritmasına genel bakış

Şifreleme işleminin başlangıcında, 64 bit düz metin, Şekil 3.6' da P1 (16 Bit), P2 (16 Bit), P3 (16 Bit) ve P4 (16 Bit) olarak ifade edilen dört eşit boyutlu bloğa bölünür ve bu bloklar ilk tur için girdi olur. İlk turun çıktısı, 2.turun girişi ve benzer şekilde de 2.turun çıktısı 3. turun girdisi olur ve bu durum 8 tur boyunca devam eder. Nihayetinde 8.turun çıkışı, sonucunda 64 bit şifreli metin elde edilen çıkış dönüşümü fazına girdi olur. IDEA,

simetrik bir şifreleme algoritması olduğundan şifreleme ve şifre çözme işlemlerinde aynı anahtarı kullanır. Şifre çözme işlemi, alt anahtarların üretiminde farklı bir algoritmanın kullanılması dışında şifreleme işlemiyle aynıdır. Şifreleme anahtarı, 128 bittir. Tüm şifreleme sürecinde (1.turdan 8.tura ve çıkış dönüşümü fazına kadar), bu 128 bitlik anahtardan üretilen toplam 52 adet anahtar kullanılır. Her turda (1.turdan 8 tura kadar), 6 adet alt anahtar kullanılır ve her alt anahtar 16 bitten oluşur. Son olarak çıkış dönüşümü fazında ise 4 adet alt anahtar kullanılır (Furht ve ark., 2005).

3.1.1.4. Blowfish algoritması

Blowfish algoritması, DES'in yerini alması amacıyla Aralık 1993'teki Cambridge Güvenlik Çalıştayı'nda Schneier tarafından tasarlanmıştır. Algoritma, donanımda gerçeklenmesi için uygunluğu ve verimliliği de dahil olmak üzere çeşitli avantajlarından dolayı geniş çaplı analiz edilmiş ve kademeli olarak iyi ve güçlü bir şifreleme algoritması olarak kabul edilmiştir. Ayrıca patentsizdir ve bu nedenle herhangi bir lisans gerektirmemektedir. Blowfish algoritmasının temel operatörleri, dört S-kutusu (Substitution Box, Yerine koyma kutusu) ve bir P-dizisinden (Permutation, Yer değiştirme) oluşan tabloda arama, ekleme ve XOR işlemlerini içerir. Feistel mimarisine dayalı algoritma, benzer güvenlik, daha yüksek hız ve daha yüksek verimlilik sağlamak için DES 'te kullanılan ilkelerin basitleştirilmiş bir versiyonu olarak tasarlanan F-fonksiyonlarını kullanır (Schneier, 1994; Thakur ve Kumar, 2011).

Blowfish, benzer ve yinelenen şifreleme ve deşifreleme fonksiyonlarından oluşan 16 turluk bir simetrik blok şifreleme algoritmasıdır. Her Feistel yapısı özellikle donanımda çeşitli avantajlar sunar. Şifre çözme sürecinde tek gereksinim anahtar planlamasını tersine çevirmektir. Algoritma anahtar genişletme (key expansion) ve veri şifreleme (data encryption) olarak ikiye ayrılabilir (Nie ve Zhang, 2014).

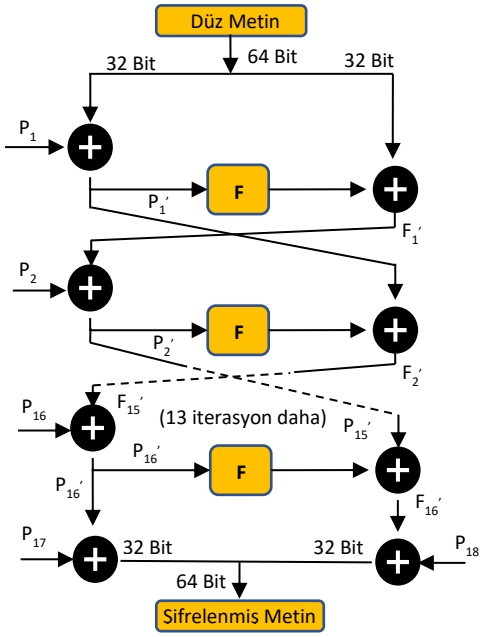
Anahtar genişletme kısmı, P-dizisi ve S-kutularının kullanılarak şifreleme ve şifre çözme işlemleri öncesinde ön hesaplama gerektiren birçok alt anahtarın oluşturulmasıyla başlar. P-dizisi 18 adet 32 bitlik alt anahtar içerir:

Bunlar; P1, P2, … P18, şeklinde ifade edilir. Bu bölümde maksimum 448 bitlik bir anahtar toplam 4168 bayta kadar birkaç alt anahtar dizisine dönüştürülür. Dört adet 32-bit S-kutusunun her birinde 256 eleman vardır. Bu elemanlar; S0, S1, ..., S255, şeklinde ifade edilir.

Alt anahtarların nasıl hesaplandığı aşağıda açıklanmıştır (Schneier, 1996):

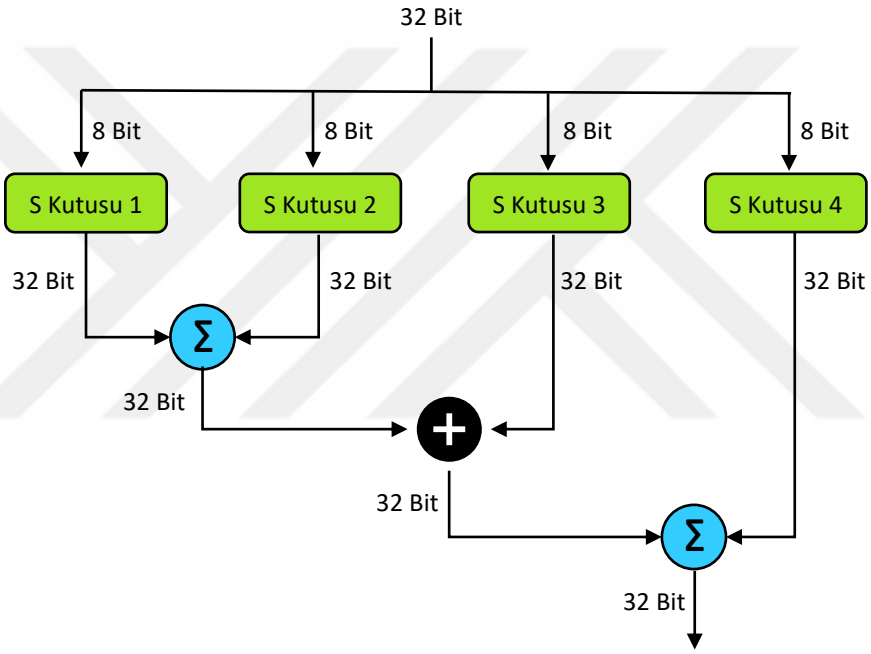
1. İlk olarak P dizisinin başlangıç değerleri, *pi* sayısının ilk 3 rakamı hariç heksadesimal dijitlerinden türetilen değerler olarak atanır. 4 adet S-kutusunun başlangıç değerleri de yine bu şekilde belirlenir.
2. Daha sonra anahtarın ilk 32 biti, P1 ile ikinci 32 biti P2 ile olacak şekilde XOR işlemine tabi tutulur. Bu döngü tüm P-dizisi anahtar bitleriyle XOR edilinceye kadar devam eder.
3. Sonrasında, 1. ve 2. adımda açıklanan tümü 0 dijitlerinden oluşan 64 bitlik bir metin Blowfish algoritması kullanılarak şifrelenir.
4. 3. adımda elde edilen 64 bitlik çıktı 32 bitlik iki parçaya ayrılır. İlk 32 bit P1'in, ikinci 32 bit ise P2'nin yeni değerleri olarak atanır.
5. Değiştirilen bu alt anahtarlar kullanılarak 3. adımda elde edilen çıktı, Blowfish algoritmasıyla şifrelenir.
6. P3 ve P4'ün değerleri de 5. adımda elde edilen sonuçla değiştirilir.
7. P-dizisinin tüm elemanları değiştirilene kadar bu işlem devam eder ve ardından 4 adet S-kutusunun tüm elemanları da değiştirilir.

Şekil 3.7, 16 turluk Blowfish algoritmasının mimarisini göstermektedir. Veri şifreleme aşaması, düz metinden alınan 64 bitlik bir blok parçasıyla başlar. İşlem sonucunda bu blok, 64 bitlik şifreli bir metne dönüşür



Şekil 3.7. Blowfish algoritmasının mimarisi

İlk olarak 64 bitlik düz metin, sağda ve solda olmak üzere 32 bitlik iki eşit parçaya ayrılır. Bir sonraki adımda 32 bitlik sol blok ile P dizisindeki ilk alt anahtara (P1), XOR işlemi uygulanır. Bu adımdan elde edilen 32 bitlik veri, permütasyon işlemi uygulanacak olan F-fonksiyonuna taşınır. Fonksiyonun çıktısı, 64 bitlik düz metnin 32 bitlik ikinci (sol) bloğu ile XOR işlemine girer. XOR işlemi tamamlandıktan sonra 32 bitlik sağ ve sol bloklar bir sonraki iterasyon için birbirleriyle yer değiştirir. Şifre çözme işlemi, veri şifreleme aşamasıyla benzerdir. Ama deşifreleme kısmında P-dizisini oluşturan alt anahtarlar ters sırada kullanılır (Schneier, 1994). Algoritmanın önemli bileşenlerinden biri olan F–fonksiyonlarının çalışma mekanizması Şekil 3.8' de gösterilmiştir.



Şekil 3.8. F-fonksiyonunun mimarisi

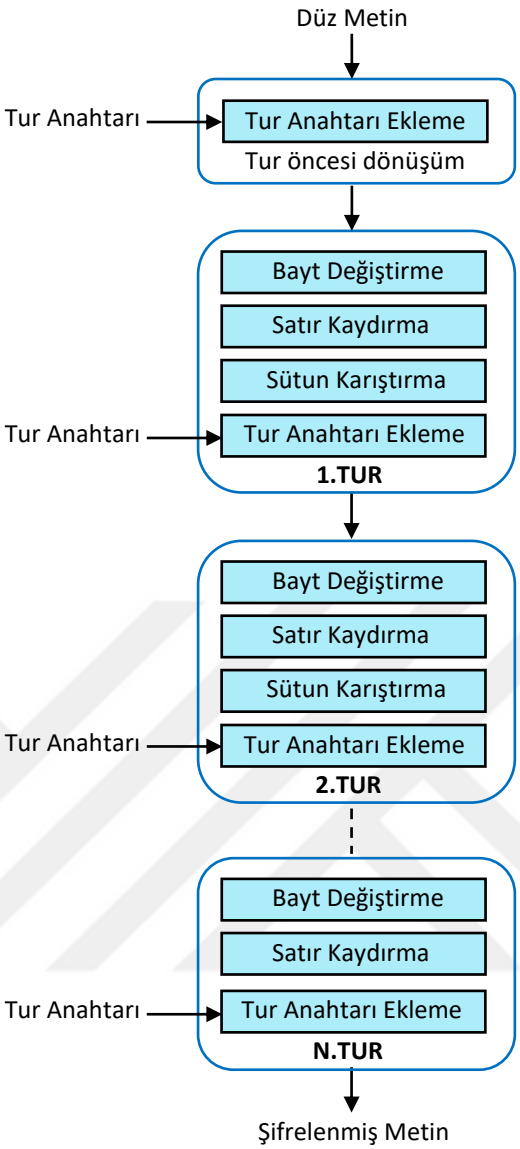
Blowfish'in F-fonksiyonu algoritmanın en karmaşık kısmıdır. Çünkü S-kutularının kullanıldığı tek kısımdır. 32 bitlik bir veri akışını girdi olarak alır ve dört eşit parçaya böler. Her 8 bitlik alt bölüm, S-kutusunda her birine karşılık gelen 32 bitlik veri akışına dönüştürülür. Elde edilen 32 bitlik veriler birbirleriyle XOR ve toplam işlemine tabi tutularak 32 bitlik final değeri elde edilir. Tüm toplama işlemleri $mod\ 2^{32}$'ye göre yapılır. F-fonksiyonuyla elde edilen çıktı, Blowfish algoritmasının permütasyon işlemlerinde kullanılır (Mousa, 2005).

3.1.1.5. Advanced Encryption Standart (AES) algoritması

1997 yılında Ulusal Standartlar ve Teknoloji Enstitüsü (National Institute of Standards and Technology, NIST), eskiyen DES algoritmasının yerini alacak bir veri şifreleme standardının geliştirilmesi ve belirlenmesi için bir program düzenledi. 1998 yılında, NIST, on beş aday algoritmanın kabul edildiğini duyurdu ve bu algoritmaların analiz edilmesi için kriptografik araştırma topluluğundan yardım istedi. Bu inceleme, her algoritma için güvenlik ve verimlilik özelliklerini içeriyordu. NIST, bu ön araştırmanın sonuçlarını gözden geçirdi ve finalist olarak Çok Değişkenli Uyarlanabilir Regresyon Uzanımları (Multivariate Adaptive Regression Splines, MARS), Rivest Şifre 6 (Rivest Cipher, RC6), Rijndael, Serpent ve Twofish algoritmalarını belirledi. Ekim 2000'de finalist algoritmaların genel analizlerini daha fazla inceledikten sonra NIST, Rijndael'i AES olarak önermeye karar verdi. Joan Daemen ve Vincent Rijmen tarafından tasarlanan Rijndael, basit ve kullanışlı bir yapıya sahipti (Mohan ve Reddy, 2011).

Rijndael olarak da bilinen AES, 128 bitlik veri bloklarını, 128, 192 veya 256 bitlik anahtarlar kullanarak şifreleyebilen bir simetrik blok şifreleme algoritmasıdır. AES özellikle işlem zamanını azaltmak amacıyla 3DES algoritmasının yerini alması için tanıtılmıştır. Yine de eğer güvenlik tek ölçüt olsaydı 3DES, onlarca yıldır kullanılan standart bir şifreleme algoritması olduğu için daha uygun bir seçim olacaktı. 3DES'in en büyük dezavantajı yazılımda uygulanmasında yavaş olmasıydı. Şifreleme sistemlerinde gerek verimlilik gerekse güvenlik için daha büyük bir blok boyutunun kullanılması istenir. Yüksek güvenlik düzeyi, hızı, uygulamadaki kolaylığı ve esnekliğinden dolayı Rijndael, 2001 yılında AES standardı olarak seçilmiştir (Mohan ve Reddy, 2011).

AES, yerine koyma-permütasyon bileşimi olarak bilinen bir tasarım ilkesine dayanmaktadır ve hem donanımda hem de yazılımda hızlıdır. Öncesindeki DES' ten farklı olarak AES, Feistel ağı kullanmaz. AES, durum matrisi olarak tanımlanan ve şifrelenecek verilerin baytlar halinde yerleştirildiği 4x4' lük bir matris üzerinde çalışır. Algoritmanın anahtar boyutu, tur sayısına bağlı olarak belirlenir. 10 tur için 128 bit, 12 tur için 192 bit ve 14 tur için 256 bit anahtar kullanılır (Al- Mamun ve ark., 2017). Her tur, farklı aşamalar içeren dört benzer işlem adımından oluşur. Bu turlardaki adımların ters sırayla uygulanmasıyla da şifreli metinden orijinal metin elde edilir. Şekil 3.9' da AES algoritmasının temel çalışma mekanizması gösterilmiştir.



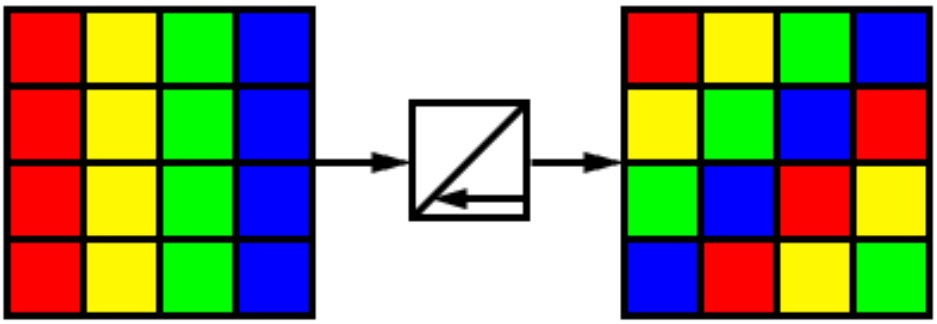
Şekil 3.9. AES algoritmasının genel yapısı

AES algoritmasında genel bir tur dört alt işlemden meydana gelir. Bunlar bayt değiştirme, satır kaydırma, sütun karıştırma ve tur anahtarı ekleme olarak adlandırılır. Sadece başlangıç aşamasında ve final turunda ara turlara göre bazı farklılıklar bulunmaktadır. Final turunda ara turlardan farklı olarak sütun karıştırma aşaması yoktur, bunun dışında sürecin geri kalanı ara turlarla aynıdır. Şekil 3.9' da görüldüğü üzere her turda bir tur anahtarı kullanılmaktadır. Bu tur anahtarları her tur için 128 bit uzunluğunda olmak üzere bir anahtar dizesi üreticisi tarafından sağlanır.

Başlangıç aşamasında sadece tur anahtarı ekleme işlemi yapılır. Bu adımda, 4x4'lük bir durum matrisi haline getirilen düz metin bloğu ile 128 bitlik tur anahtarına XOR işlemi uygulanır. Bu işlem sonucunda yeni durum matrisi elde edilmiş olur.

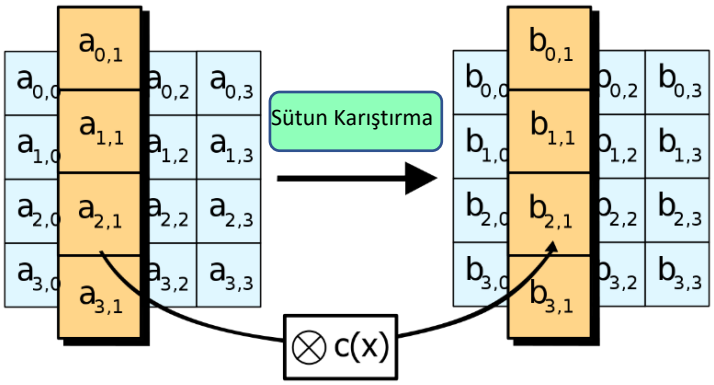
Başlangıç aşamasından sonra ara turlara geçilir. Bu turların ilk adımı daha önce de bahsedildiği üzere bayt değiştirme işlemidir. Bu adımda durum matrisinin her elemanı ($a_{i,j}$), 8 bitlik bir S-kutusundaki elemanlar $S(a_{i,j})$ ile yer değiştirilir. Bu işlem şifrelemede doğrusal olmamayı sağlar. Basit cebirsel özellikli saldırıları önlemek için S-kutusu, ters fonksiyon ile tersi alınabilir bir afin dönüşümünün kombinasyonu ile oluşturulmalıdır (Sachdev ve Bhansali, 2013). Bayt değiştirme adımından sonra satır kaydırma işlemine geçilir.

Satır kaydırma işlemi, durum matrisinin satırları üzerinde uygulanır. Şekil 3.10' da görüldüğü üzere her satırdaki elemanlar belirli bir kurala göre dairesel olarak kaydırılır. İlk satırda herhangi bir değişiklik yapılmaz. İkinci satırdaki her eleman bir sola kaydırılır. Benzer şekilde üçüncü ve dördüncü satırlar da sırasıyla iki ve üç adım sola kaydırılır. 128 bit ve 192 bit bloklar için bu kaydırma modeli aynıdır. Yani n. satır, n-1 bayt sola dairesel olarak kaydırılır. 256 bitlik blok için ise ilk satırda yine bir değişiklik yapılmaz fakat ikinci, üçüncü ve dördüncü satırlar sırasıyla 1 bayt, 3 bayt ve 4 bayt sola kaydırılır. Bu değişiklik sadece Rijndael algoritmasında ve 256 bitlik blok kullanıldığında uygulanır. Ama AES algoritmasında 256 bit blok kullanılmamaktadır (Rawal, 2016).



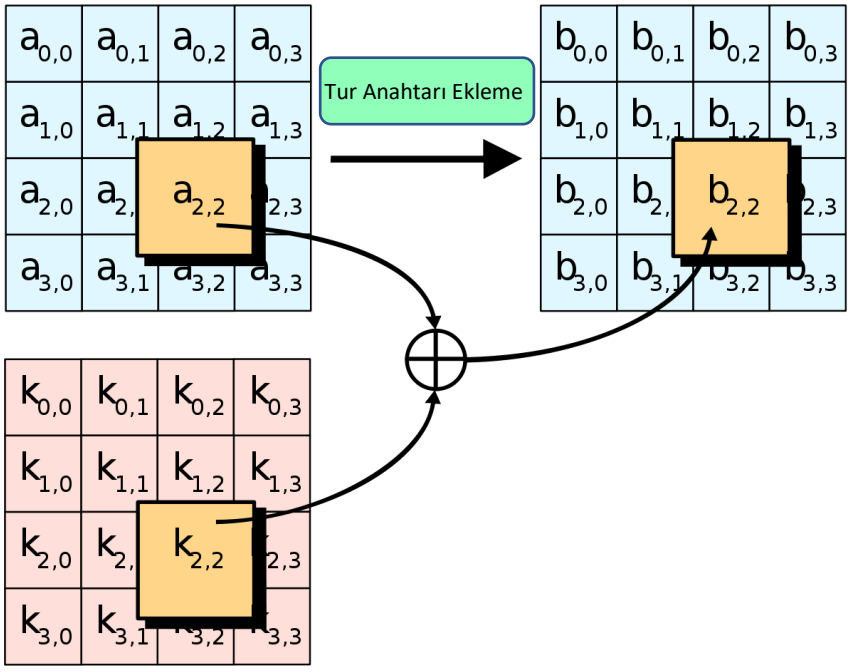
Şekil 3.10. Satır kaydırma işleminin uygulanması

Satır kaydırma işleminin tamamlanmasından sonra sütun karıştırma aşamasına devam edilir. Bu aşamada durum matrisinin her sütunundaki dört bayt, tersi alınabilir lineer bir dönüşüm kullanılarak işlenir. Sütun karıştırma fonksiyonu, girdi olarak dört bayt alır ve bunlar tüm dörtlü çıkış baytlarını etkiler. Sütun karıştırma, satır kaydırma ile birlikte şifrelemede difüzyon sağlar (Al- Mamun ve ark., 2017). Şekil 3.11' de gösterilen bu işlem boyunca durum matrisinin her bir sütunu, sabit bir polinom C(x) ile çarpılarak dönüştürülür.



Şekil 3.11. Sütun karıştırma işlemi

Turların son aşaması ise tur anahtarı ekleme işlemidir. Aslında bu işlem daha önce bahsedildiği üzere ara turlara geçmeden önce başlangıç aşamasında ilk kez uygulanır. Algoritmanın devamında da ara turların her birinde ve final turunda son işlem olarak uygulanmaktadır. Tur anahtarı ekleme aşamasında alt anahtar ya da başka bir deyişle tur anahtarı, durum matrisi ile kombine edilir. Her turda bir alt anahtar, Rijndael' in anahtar planlama algoritması kullanılarak ana anahtardan türetilir. Her alt anahtar, durum matrisi ile aynı boyuttadır. Şekil 3.12' de görüldüğü üzere durum matrisinin her bir baytı, alt anahtarda ona karşılık gelen bayt ile XOR işlemine tabi tutulur.



Şekil 3.12. Tur anahtarı ekleme işlemi

Bir AES şifreli metninin şifresini çözme süreci, şifreleme işleminin tersi gibidir. Şifrelemede her turda uygulanan dört işlem, şifre çözmede ters sıra ile uygulanır. Bundan

dolayı şifre çözme işleminin adımları sırasıyla tur anahtarı ekleme, sütun karıştırma, satır kaydırma ve bayt değiştirme olarak ifade edilir. Her turda alt işlemler ters sırada olduğu için Feistel şifrelemeden farklı olarak şifreleme ve deşifreleme işlemleri birbirleriyle yakından ilişkili olmalarına rağmen ayrı ayrı uygulanmak zorundadır (Sachdev ve Bhansali, 2013).

Günümüz kriptolojisinde AES hem donanım hem de yazılım için yaygın olarak kullanılmış ve desteklenmiştir. Esnek anahtar uzunluğuna sahip bir algoritmadır. Blok uzunluğunun artırılmış olması ve tur sayısının azaltılmış olması yerini alması için tasarlandığı DES algoritmasına göre hızlı olmasını sağlamıştır. Ancak, DES' te olduğu gibi AES, yalnızca doğru bir şekilde uygulanırsa ve iyi bir anahtar yönetimi sağlanırsa güvenli bir algoritmadır (Rawal, 2016).

3.1.1.6. Rivest Cipher 4 (RC4) algoritması

RC4, RSA Security kurucularından Ron Rivest tarafından 1987 yılında geliştirilen bir simetrik akış şifreleme algoritmasıdır. RC4'ün birden fazla açılımı mevcuttur; Ron's Code 4, Rivest Cipher 4. RC4 şifreleme algoritması başlangıçta gizli olarak tutulmuş ancak 1994 Eylül ayında bu algoritmanın kaynak kodları internet ortamında yayınlanmıştır. Buna rağmen RC4 ticari bir marka haline getirilmiştir ve ARCFOUR ve ARC4 gibi isimler takılmıştır (Yasin, 2017).

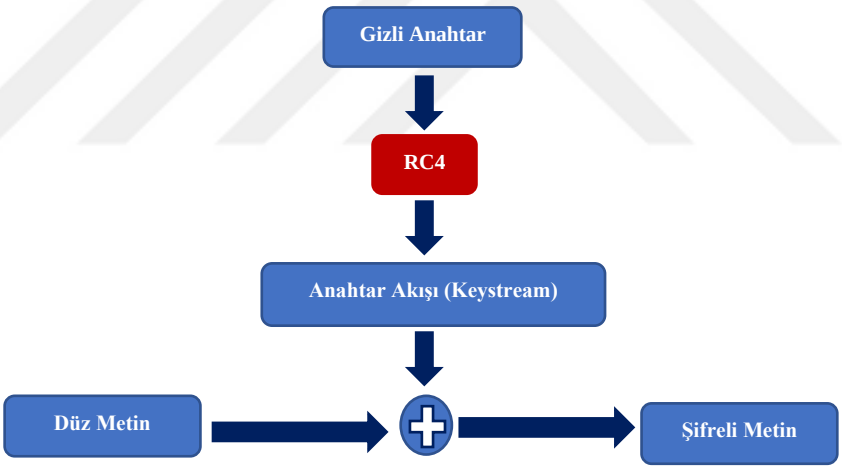
Birçok akış şifreleme algoritması, yazılımdan çok donanımda uygulanabilirliği kolay olması için tasarlanmıştır. RC4 algoritması ise yazılım uygulamalarında kullanılması için ideal bir algoritmadır, işlemler bayt'lar üzerinden yapılır. Durum dizilerinin tutulması için (S[0] … S[255]) 256 baytlık bir hafıza kullanılır. Hafızanın *k* baytı (anahtar[0] … anahtar[*k*-1]) anahtar ve tam sayı değişkenleri (*i*, *j*) için ayrılır (Fluhrer ve ark., 2001)**.** RC4, yazılım uygulamalarında en geniş uygulama alanına sahip olan şifreleme algoritmalarından birisidir. Bu kadar yaygın olarak kullanılmasının başlıca sebepleri yüksek hız, basitliği ve kolay uygulanabilirliğidir (Mammadov, 2010).

RC4 şifreleme algoritmasının kullanıldığı bazı uygulamalar aşağıda belirtilmiştir:

- Kablolu Eşdeğer Gizlilik (Wired Equivalent Privacy, WEP) 802.11b
- Kablosuz Bağlantı Alanı Korumalı Erişim (Wireless Fidelity Protected Access, WPA)
- BitTorrent protokol şifreleme

- Microsoft noktadan noktaya şifreleme
- Güvenli Soket Katmanı (Secure Sockets Layer, SSL)
- Taşıma Katmanı Güvenliği (Transport Layer Security, TLS)
- Güvenli Kabuk (Secure Shell, SSH)
- Uzaktan masaüstü erişim protokolü
- Oracle Secure Yapılandırılmış Sorgu Dili (Structured Query Language, SQL),
- Lotus Notes,

RC4, simetrik anahtarlı bir algoritmasıdır. Simetrik olması düz metnin şifrelenmesinde ve şifreli metnin çözülmesinde aynı anahtarın kullanılması anlamına gelmektedir. Şekil 3.13' de görüldüğü gibi gizli anahtar RC4 algoritmasına girdi olur ve RC4 algoritması ile anahtar dizisi üretilir. Anahtar dizisi ile düz metin bayt bayt XOR işleminden geçirilir ve şifreli metin oluşturulur. Şifreli metin bir kanal vasıtasıyla alıcı tarafa ulaştırılır.



Şekil 3.13. RC4 algoritması ile şifreleme

RC4, iki bölümden oluşmaktadır:

- Anahtar Planlama Algoritması (Key Scheduling Algorithm, KSA)
- Sözde Rastgele Üreteç Algoritması (Pseudo Random Generation Algorithm, PRGA)

KSA, 40 ile 2048 bit arasında bir uzunluğa sahip olan anahtarı, bir başlangıç *S* permütasyonuna dönüştürür. Bu kapsamda KSA, durum ve anahtar dizilerinin üretilmesi işlemlerini içerir; bu diziler için iki tane 256 bayt uzunluğunda dizi kullanılır. 256 olası

tüm baytlar için permütasyon işlemi yapılır (Rise ve ark., 2008). Şekil 3.14' de görüldüğü gibi KSA algoritmasında ilk etapta 256 bayt uzunluğunda S durum dizisinin tüm indislerine sırasıyla 0' dan 255' e kadar tam sayı değerler atanır. Aynı zamanda key anahtar dizisinden yine 256 bayt uzunluğunda bir K dizisi elde edilir. Daha sonraki aşamada bu K anahtar dizisi, S durum dizisiyle birlikte j değişkenini hesaplamak için kullanılır. Her adımda i indeksi bir arttırılır ve j indeksi yeniden hesaplanır. Böylece her adımda S dizisinin i ve j indeksli elemanları yer değiştirilerek bu dizinin içeriği rastgele sıralanmış değerler ile değiştirilmiş olur.

```
for i from 0 to 255
    S[i] = i
    K[i] = key[i mod keylength]
endfor
j = 0
for i from 0 to 255
    j = (j + S[i] + K[i]) mod 256
    swap(S[i], S[j])
endfor
```

Şekil 3.14. KSA algoritmasının kaba kodu

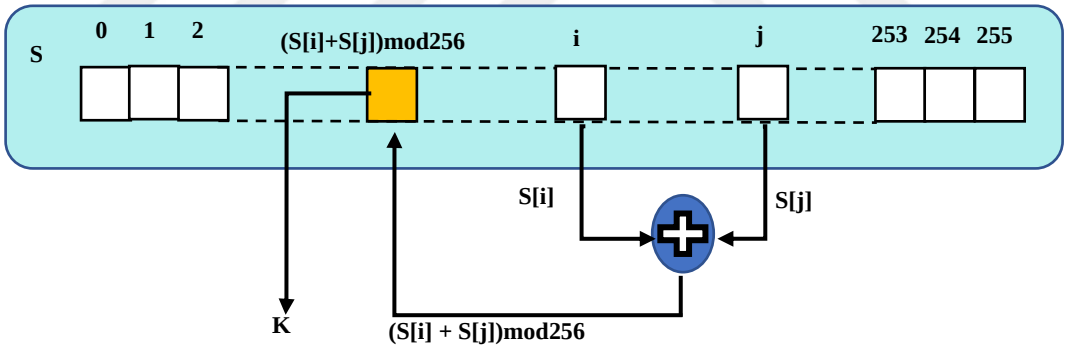
PRGA ise KSA aşamasında üretilen permütasyonu kullanarak bir sözde rastgele dizi üretimini gerçekleştirir (Gürkaş, 2005). Bu sözde rastgele dizi, anahtar dizisini oluşturmaktadır ve anahtar dizisi ile düz metin XOR işleminden geçirilerek şifreli metin elde edilir. Şekil 3.15' de PRGA algoritmasına ait bir pseudo kod yer almaktadır.

```
i = 0
j = 0
while GeneratingOutput:
    i = (i + 1) mod 256
    j = (j + S[i]) mod 256
    swap(S[i], S[j])
    output S[(S[i] + S[j]) mod 256]
endwhile
```

Şekil 3.15. PRGA algoritmasının kaba kodu

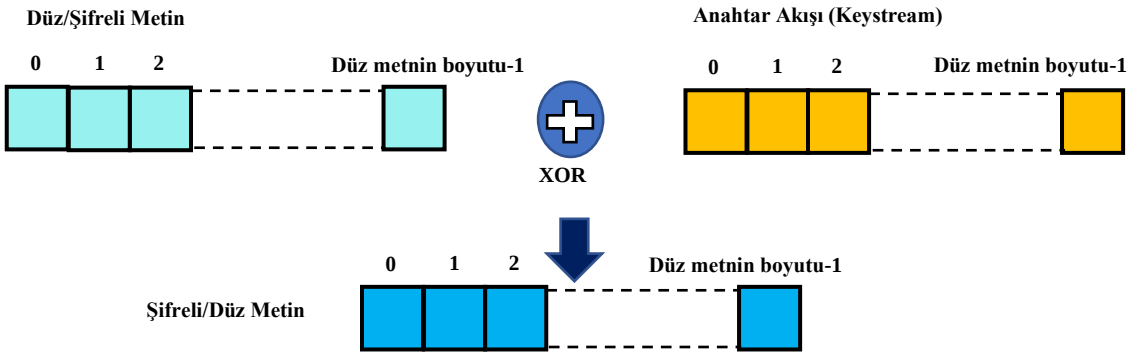
PRGA' da çıkış baytları *S[i]* ve *S[j]* değerlerinin toplamının 256' ya göre modunun alınmasıyla belirlenir. Algoritma bir yandan çıkış baytlarını üretirken bir yandan da *S* durum dizisini düzenli bir şekilde değiştirmeye devam eder. Bu işlemi de her iterasyonda durum dizisindeki *S[i]* ve *S[j]* elamanlarının yerlerini birbirleriyle değiştirerek yapar. İterasyon sayısı, şifrelenecek düz metnin boyutuna bağlıdır. Düz metin kaç baytsa bu döngü de o kadar kez devam eder. Çünkü PRGA algoritması, düz metindeki her karaktere karşılık bir çıkış baytı üretir.

Çıkış baytlarının üretimi Şekil 3.16' da görselleştirilmiştir. Algoritma, her bir iterasyonda S durum dizisindeki i ve j indeksli elemanların değerlerini toplar ve daha sonra bu toplamın 256'ya göre modunu alır. Bu işlem sonucunda elde edilen değer, S durum dizisinin hangi indeksli elemanın çıkış baytı olarak seçileceğini belirler (Mammadov, 2010). Her iterasyon sonucunda bu çıkış baytı, anahtar akışına eklenir ve anahtar akışı da daha sonra düz metni şifrelemek için kullanılır. Algoritmada 256'ya göre mod işlemi uygulanmasının sebebi S durum dizisinin 256 elemandan oluşuyor olmasıdır. Eğer mod işlemi alınmazsa S[i]+S[j] toplamının 256'dan büyük çıkması durumunda taşma hatası meydana gelir.



Şekil 3.16. PRGA algoritmasında çıkış baytlarının üretimi

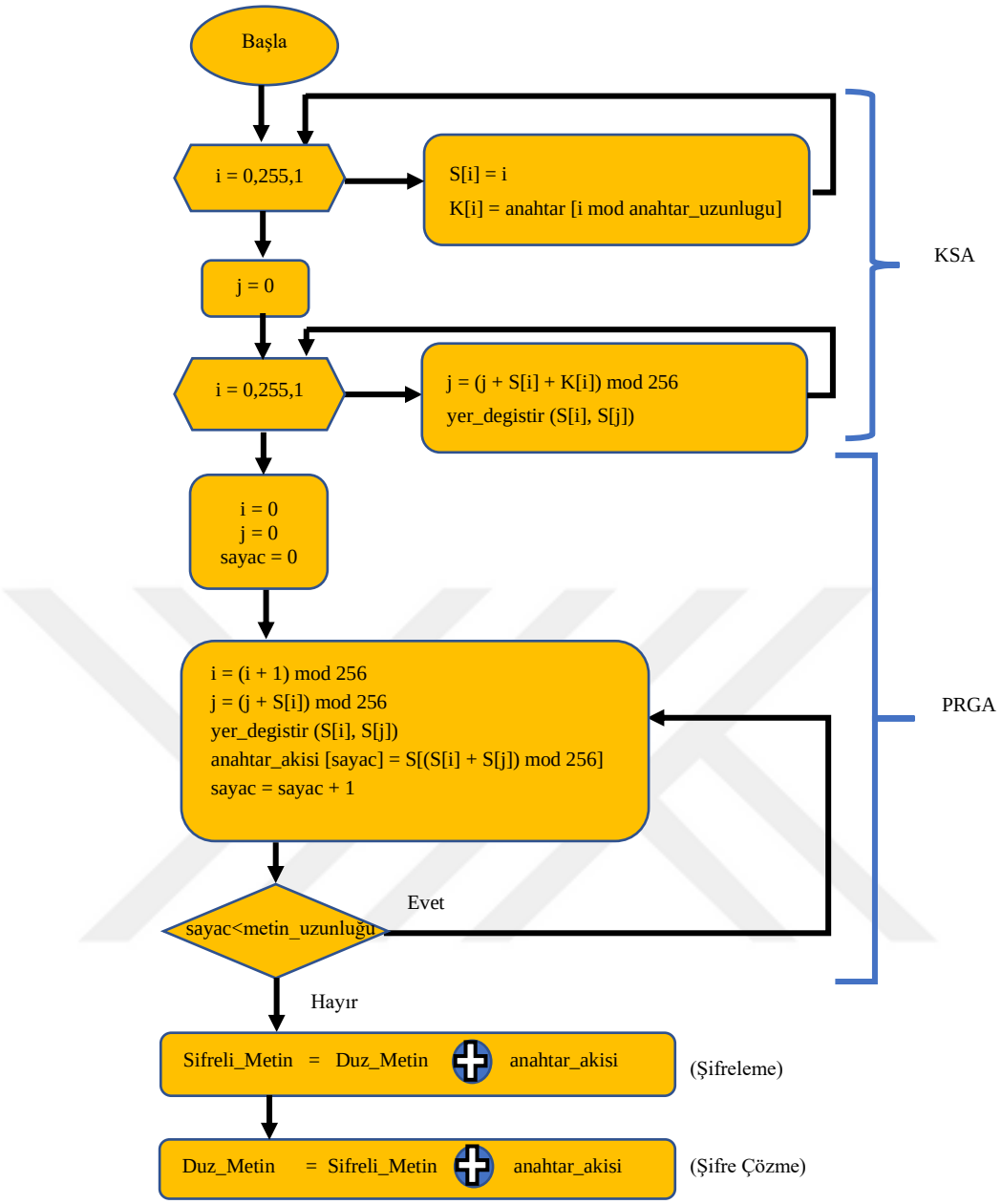
KSA aşamasındaki 256 iterasyon neticesinde karıştırılan *S* durum dizisi, PRGA aşamasında anahtar akışının üretilmesinde kullanılır. PRGA aşaması ise düz metindeki karakter sayısı kadar tekrar eder. Bu işlem sonucunda elde edilen anahtar akışı da şifreleme ve şifre çözme işlemlerinde kullanılır. Şekil 3.17' de görüldüğü üzere şifreleme ve şifre çözme işlemleri aynı şekilde gerçekleşmektedir.



Şekil 3.17. RC4 şifreleme ve şifre çözme işlemleri

Düz metnin, anahtar akışı ile bayt bayt XOR işlemine tabi tutulması sonucunda şifreli metin elde edilir. Bunun için düz metindeki her bir karakterin ASCII karşılığı alınır ve binary forma getirilir. Ardından anahtar akışındaki her bir baytın, düz metindeki her bir bayt ile karşılıklı olarak XOR işlemine tabi tutulması sonucunda şifreli metin elde edilir. Aynı şekilde şifreli metnin şifresinin çözülüp düz metnin elde edilmesi için de şifreli metinle anahtar akışı bayt bayt XOR işlemine girer. Şekil 3.18'de RC4 algoritmasının tüm aşamalarını içeren bir akış diyagramı görülmektedir.

RC4 algoritmasında öncelikle KSA aşaması çalışmaktadır. KSA' da ilk olarak S durum dizisi sırasıyla 0'dan 255'e kadar tam sayılarla doldurulmaktadır. Aynı zamanda 5-256 bayt arasında uzunluğa sahip anahtarın baytları sırasıyla tekrar ettirilerek 256 bayt boyutunda K anahtar dizisi oluşturulmaktadır. S durum ve K anahtar dizilerinin oluşturulmasından sonra S dizisinin elemanları kendi içerisinde yer değiştirilerek permütasyon işlemi uygulanmakta ve böylece KSA aşaması tamamlanmaktadır. Ardından PRGA aşamasına geçilmektedir. Bu aşamada düz metni şifrelemek için kullanılacak ve düz metinle aynı boyutta olacak bir anahtar akışı üretilir. Aynı zamanda da yine S durum dizisi karıştırılmaya devam edilir. KSA aşaması 256 kez tekrar ederken PRGA aşaması ise düz metindeki karakter sayısı kadar tekrar eder. PRGA neticesinde elde edilen anahtar akışı ile düz metin bayt bayt XOR işlemine tabi tutularak şifreli metin elde edilmiş olur. Şifre çözme işlemi de şifreleme aşamasında olduğu gibi şifreli metinle anahtar akışının XOR işlemine tabi tutularak düz metnin elde edilmesi şeklinde gerçekleşir.



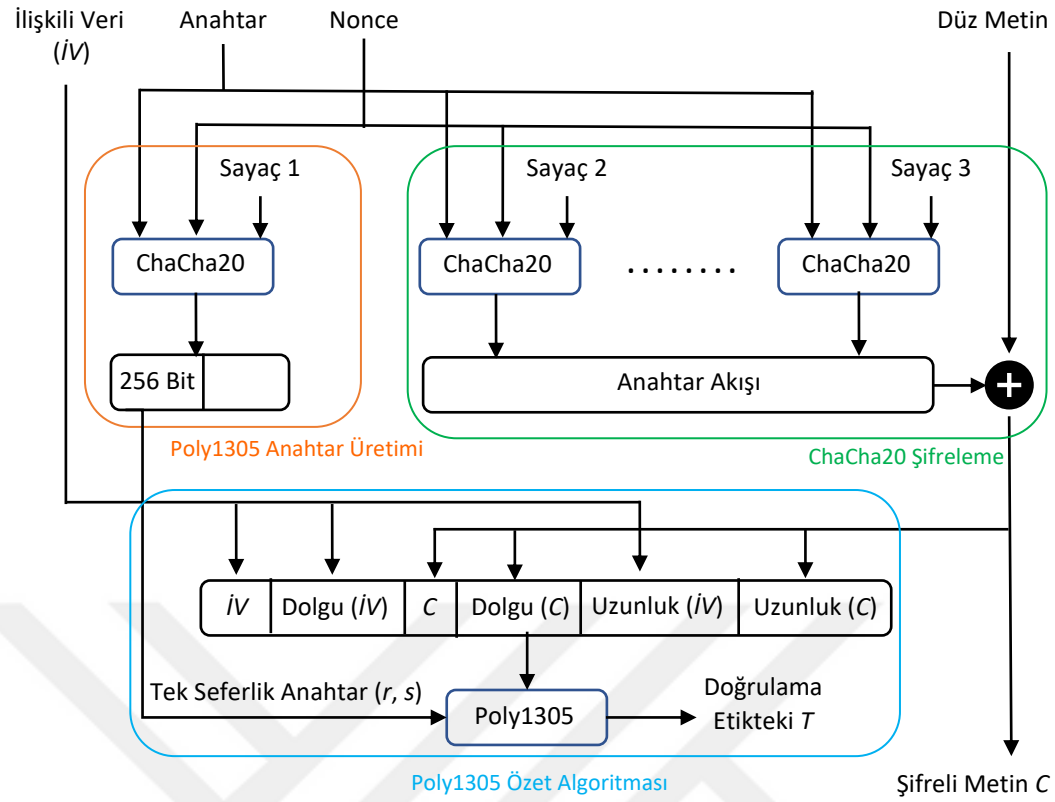
Şekil 3.18. RC4 algoritması akış diyagramı

3.1.1.7. ChaCha20 – Poly1305 algoritması

ChaCha20 akış şifreleme ve Poly1305 kimlik doğrulama algoritmaları hem çok çeşitli yazılım platformlarında yüksek performans elde etmek hem de yüksek güvenlik sağlamak amacıyla Bernstein (2008) tarafından tasarlanan kriptografik algoritmalardır. İnternet Mühendisliği Görev Gücü (Internet Engineering Task Force, IETF) grubu, onun yazılım platformlarındaki performansı ve güvenli uygulamaları gerçekleştirme kolaylığından dolayı TLS protokolünde ChaCha20 akış şifreleme ve Poly1305 kimlik

doğrulama algoritmalarının kullanımını ve standardizasyonunu teşvik etmek için 2015 ve 2016 yıllarında Request for Comments (RFC) 7539 ve RFC 7905 dokümantasyonlarını yayınlamıştır (Nir ve Langley, 2015; Langley ve ark., 2016). RFC 7539, verilerin gizliliğini, bütünlüğünü ve doğruluğunu sağlamak amacıyla kimliği doğrulanmış bir şifreleme şeması oluşturmak için ChaCha20 akış şifreleme ve Poly1305 kimlik doğrulama algoritmalarının nasıl bir araya getirileceğini belirtir. Bunun yanı sıra Poly1305 ve ChaCha20, Google gibi lider şirketler ve OpenSSH gibi küresel projeler tarafından resmi olarak desteklenmektedir (Fukushima ve ark., 2017). ChaCha algoritması, güvenliğe karşı performans tercihini desteklemek için algoritmanın farklı spesifikasyonlarına izin verir (Bernstein, 2008). Başka bir deyişle algoritmadaki anahtar ve sayaç gibi parametrelerin farklı uzunluklarda tanımlanabilmesinin yanı sıra algoritmanın farklı tur sayılarında çalıştırılabilmesine de imkan tanır. Bu kısımda işin güvenlik tarafına odaklanan ve RFC 7539' da belirtilen ChaCha20 algoritması açıklanmaktadır.

Şekil 3.19, ChaCha20 – Poly1305 algoritmasının çalışma mekanizmasını göstermektedir. Algoritma keyfi uzunluktaki bir İlişkili Veri (İV), 256 bitlik gizli anahtar, 96 bitlik nonce, 32 bitlik bir sayaç değeri ve düz metin olmak beş ayrı girdi almaktadır. Nonce, aynı anahtar kullanılarak yapılan her şifreleme işlemi için benzersiz olması gereken bir sayıdır. Bu yüzden rastgele üretilmemelidir. Bir sayıcı, nonce üretmek için iyi bir seçenek olabilir. Ama bir Doğrusal Geri Besleme Kaydırma Yazmacı (Linear Feedback Shift Register, LFSR) gibi diğer metotlar da nonce üretmek için kullanılabilir. ChaCha20 algoritması, sayıcı modunu kullanarak blok fonksiyonlarını çalıştırır ve işlem sonucunda bir anahtar akışı oluşturur. Bu anahtar akışının düz metinle XOR' lanmasıyla da şifreli metin elde edilir. Benzer şekilde şifreli metinle anahtar akışının XOR' lanmasıyla da şifre çözme işlemi gerçekleştirilir. ChaCha20 blok fonksiyonu anahtar, sayaç ve nonce değerlerini de kullanarak 512 bitlik bir durum dizisi oluşturur ve bu dizi üzerinde tur fonksiyonlarını iteratif olarak çalıştırıp toplam 20 turda işlem sürecini tamamlar. Her blok fonksiyonu sonucunda anahtar akışını meydana getiren 512 bitlik bloklar elde edilir (Langley ve ark., 2016).



Şekil 3.19. ChaCha20 – Poly1305 algoritmasının çalışma yapısı

Şekil 3.19' da görüldüğü üzere algoritmadaki ilk blok fonksiyonu anahtar akışını üretmek için değil, Poly1305' de kullanılacak tek seferlik anahtarı oluşturmak için çalıştırılır. Bu işlem sonucunda elde edilen 512 bitlik bloğun ilk 256 biti alınırken diğer 256 bitlik kısım ise atılır. Alınan 256 bitlik kısmın ilk 128 biti *r*, sonraki 128 biti *s* olarak atanır ve bu parametreler, 32 bayt uzunluğunda tek seferlik bir gizli anahtar olmak üzere Poly1305 algoritmasına girdi olarak verilir. Poly1305'e anahtarın yanı sıra girdi olarak bir de mesaj verilir. Bu mesajın nasıl oluşturulduğu Şekil 3.19' da gösterilmiştir. Mesajda yer alan *İV* ifadesi başta algoritmaya girdi olarak verilen ilişkili veriyi, Dolgu (*İV*) ifadesi ise *İV*' ye eklenecek dolgu baytlarını temsil eder. Dolgu yapılmasının amacı İV' nin uzunluğunu, 16 baytın tam katı haline getirmektir. Bunun için İV' ye en fazla 15 bayt uzunluğunda olacak şekilde 16 baytın tam katı olana kadar 0 eklenir. Eğer zaten İV' nin uzunluğu 16 baytın tam katıysa herhangi bir doldurma işlemi yapılmaz. İV için yapılan tüm bu işlemler şifreli metin için de uygulanır. Yani eğer şifreli metin, 16 baytın tam katı değilse tam katı olana kadar şifreli metne 0 dolgusu eklenir ve şifreli metinde Poly1305'e girdi olarak verilecek mesajın içerisine yerleştirilir. Ayrıca bu mesajın içerisine İV ve şifreli metne ait uzunluk bilgileri de eklenir. Ancak uzunluk bilgisi bunların kaç bitten oluştuğunu değil kaç oktetten oluştuğunu ifade eder. Başka bir deyişle uzunluk bayt

cinsinden ifade edilir ve bu uzunluk değeri, küçük endian sırasına göre kodlanmış 64 bitlik bir tam sayı olarak temsil edilir (Langley ve ark., 2015). Küçük endian kodlama sisteminde önemli baytlar en sağda olacak şekilde sıralama yapılır. Böylece, Poly1305 algoritması için gerekli olan mesaj hazırlanmış olur. Poly1305 ise bu mesajı ve (*r*, *s*) çiftinden oluşan 32 bayt uzunluğundaki gizli anahtarı kullanarak 128 bit uzunluğundaki kimlik doğrulama etiketi *T*' yi üretir (Bernstein, 2005). Bu etikette şifrelenmiş mesaja eklenerek alıcıya gönderilir. Alıcı ise, *T* etiketini şifre çözme işlemini doğrulamak için kullanır.

3.1.2. Asimetrik (Açık anahtarlı) algoritmalar

Asimetrik anahtarlı sistemler olarak da adlandırılan açık anahtarlı şifreleme sistemleri, genel olarak bilinen bir ortak anahtar ve sahibi tarafından saklanılan bir gizli anahtar olmak üzere iki farklı anahtar içerir. Bu sistemler, şifrelemede ve şifre çözmede farklı anahtarlar kullandığından asimetrik olarak adlandırılır. Veriler bir açık anahtarla şifrelenirse, yalnızca o açık anahtara karşılık gelen özel anahtar kullanılarak şifresi çözülebilir. Genelde bu şekilde olmakla birlikte bunun tersi de mümkündür. Tüm açık anahtarlı şifreleme sistemleri, bazı hesaplaması zor problemlere dayanmaktadır. Örneğin, RSA algoritması, büyük tamsayıları çarpanlarına ayırmanın zorluğuna dayanırken, El-Gamal kriptosistemi ayrık logaritma problemine dayanır (Sann ve ark., 2019).

Açık anahtarlı şifreleme sistemleri, anahtar paylaşımı için gizliliğe ihtiyaç duymaz. Bu da büyük bir gizli iletişim ağının oluşturulmasına olanak tanır. Ek olarak, açık anahtarlı sistemler, kriptolojinin tüm hedeflerine ulaşılmasının yollarını sağlar. Asimetrik şifrelemenin, açık anahtar sertifikasyonu ve güvenli hash (özet) fonksiyonları ile bir araya getirilmesiyle dijital imza, kimlik doğrulama ve veri bütünlüğünü sağlayan protokoller geliştirilir. İşlemci hızlarındaki artış ve modern kripto analizdeki gelişmelerden dolayı modern asimetrik sistemlerde kullanılan anahtar boyutu çok büyük hale gelmiştir. Örneğin, simetrik sistemlerden 128 bitlik anahtar kullanan DES algoritmasıyla sağlanan güvenlik seviyesi, asimetrik sistemlerdeki RSA algoritmasında 1024 bitlik anahtar uzunluğuyla sağlanmaktadır (Menezes ve ark., 1997). Bu durum açık anahtarlı sistemleri, gizli anahtarlı sistemlere kıyasla dezavantajlı yapmaktadır. Yani, açık anahtarlı şifreleme önemli ölçüde daha yavaştır ve büyük bir bellek kapasitesi ve hesaplama gücü gerektirir.

Bu sorunları çözmek için araştırmacılar çeşitli yaklaşımlar ortaya koymuşlardır. Anahtar boyutunun küçültülmesi, açık anahtarlı sistemlerin akıllı kartlar ve taşınabilir kablosuz cihazlar gibi hesaplama kapasitesi daha küçük ortamlarda kullanılabilmesini sağlar. 1985' de Neil Koblitz ve Victor Miller, temel bir açık anahtar yapısı olarak karmaşık bir eliptik eğri grubunun kullanılması fikrini ortaya atmıştır (Koblitz, 1987; Miller, 1986). Ayrık logaritma problemini çözmek, eliptik eğri grubunu çözmekten çok daha zor olduğu için eliptik eğri kriptografisi, daha küçük anahtar boyutu sağlar. Başka bir strateji, kafes tabanlı indirgeme gibi daha karmaşık bir hesaplama problemine dayanan bir açık anahtar kriptosistemi tasarlamaktı. Nispeten yeni olan Sayı Teorisi Araştırma Birimi (Number Theory Research Unit, NTRU) kriptosistemi, Hoffstein, Pipher ve Silverman tarafından tanıtılmıştır (Seidel ve ark., 2004). Bu sistem hız, bellek ve hesaplama maliyeti açısından bir simetrik sistemle karşılaştırabilecek tek açık anahtarlı sistemdir. Ancak, NTRU nispeten son zamanlarda önerildiği için onun güvenli tarafları henüz araştırılmamıştır (Howgrave-Graham ve ark., 2003). Açık anahtarlı bir sistemin hızını artırmak için en yaygın çözüm, simetrik ve asimetrik anahtarlı kripto sistemlerin birleştirildiği hibrit bir şema oluşturmaktır. Yani, düz metin hızlı bir simetrik anahtar algoritması kullanılarak şifrelenir ve yalnızca simetrik şifreleme için kullanılan gizli anahtar, RSA gibi yavaş bir açık anahtarlı sistemle şifrelenir. Bu şekilde kriptografinin tüm hedefleri çok daha iyi bir performansla sağlanmış olur. Bu bölümün devamında Diffie-Hellman, El-Gamal, RSA asimetrik kripto sistemlerden detaylı olarak bahsedilmiştir.

3.1.2.1. Diffie-Hellman algoritması

İlk pratik anahtar değişim protokolü Whitfield Diffie ve Martin Hellman tarafından 1976 yılında tasarlandı (Furht ve ark., 2005). Sıklıkla Diffie-Hellman protokolü olarak atıfta bulunulan bu protokol, iletişim kuran iki taraf arasında paylaşılan ortak bir giz üzerinde güvenle anlaşmaya olanak tanır. İletişim kuran tarafların önceden herhangi bir temas kurması veya daha önce paylaşılmış bir bilgiye sahip olması gerekmez. Ancak yine de taraflar güvensiz bir kanal üzerinden gizli veriyi güvenli bir şekilde paylaşabilirler. Diffie-Hellman protokolünün güvenliği ayrık logaritma problemini çözmenin zorluğu üzerine kuruludur (Furht ve ark., 2005). Diffie-Hellman SSL, SSH ve Internet Protocol Security (İnternet Protokolü Güvenliği, IPSec) de dahil olmak üzere çeşitli protokoller tarafından kullanılır (Gupta ve ark., 2012).

İletişim kuran taraflar A ve B, potansiyel olarak güvensiz bir kanal üzerinden mesaj alışverişi yapabilirler. Bunun için p, büyük bir asal sayı ve g de üstel fonksiyonda taban olarak kullanılacak bir tamsayı olmak üzere önceden belirlenir. Algoritmanın sonucunda ise sadece A ve B tarafından bilinen ortak bir K tamsayısı elde edilir. Bu tanımlamalar göz önünde bulundurularak Diffie Hellman protokolünün işlem adımları aşağıda ifade edilmiştir (Furht ve ark., 2005).

- A kişisi, $0 \leq x \leq p-1$ olacak şekilde rastgele bir x tamsayısı seçer.
- A kişisi, $a = g^x \pmod p$ değerini hesaplar ve B'ye gönderir.
- B kişisi, $0 \leq y \leq p-1$ olacak şekilde rastgele bir y tamsayısı seçer.
- B kişisi, $b = g^y \pmod p$ değerinin hesaplar ve A'ya gönderir.
- A kişisi, $K = b^x \pmod p$ değerini hesaplar.
- B kişisi, $K = a^y \pmod p$ değerini hesaplar.

A ve B kişileri, paylaşılan gizli K'yi oluşturduktan sonra bu K anahtarını, AES gibi bir simetrik anahtarlı kripto sistemle mesajları şifrelemek için kullanabilir. İletişim kanalını dinleyen saldırganlar, p, g, $a = g^x \pmod p$, $b = g^y \pmod p$ verilerini elde edebilir. Ancak saldırganlar gizli bilgiye ulaşmak için $K = b^x \pmod p = a^y \pmod p = g^{xy} \pmod p$ verisine ihtiyaç duyar. Dolayısıyla saldırganların, K' yı hesaplayabilmeleri için ayrık bir logaritma problemini çözerek x ve y değerlerini elde etmeleri gerekir. Ancak bu, hesaplama açısından zordur.

3.1.2.2. Rivest-Shamir-Adleman (RSA) algoritması

RSA şifreleme algoritması 1977 yılında R. Rivest, A. Shamir ve L. Adleman tarafından sunulmuş ve daha sonra asimetrik kripto sistemlere uygun şekilde düzenlenmiştir. Bu algoritma, asimetrik şifreleme uygulamalarında ve dijital imza işlemlerinde güvenli bir şekilde kullanılmaktadır. RSA algoritması ayrık logaritma problemi üzerine kuruludur. Algoritmanın güvenliği büyük sayıların üretilmesi ve işlenmesindeki karmaşıklığa dayanır (Sann ve ark., 2019). Anahtar üretimi asal sayılar kullanılarak gerçekleştirilir. Çizelge 3.1' de algoritmanın blok ve anahtar uzunluğu, şifreleme hızı gibi çeşitli özellikleri ifade edilmiştir.

Çizelge 3.1. RSA' nın genel özellikleri

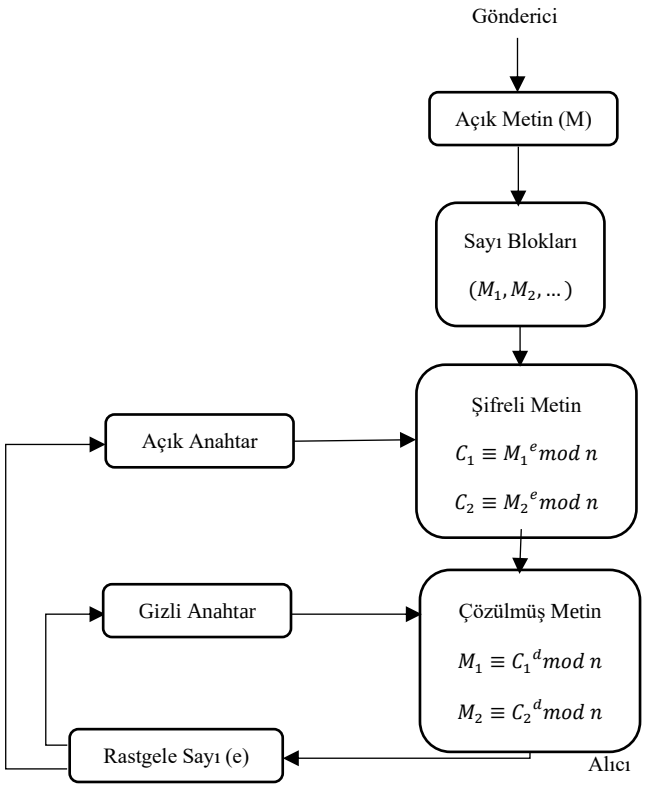
Özellik	**Açıklama**
Tarihi	1978
Anahtar Uzunluğu	1024 bit
Blok Uzunluğu	512 bit
Şifreleme ve Şifre Çözme İşlemi için Kullanılan Anahtar	Birbirinin aynısı değil
Algoritma Türü	Asimetrik şifreleme sistemi
Şifreleme işleminin Hızı	Düşük hız
Şifre Çözme işleminin Hızı	Düşük hız
Güç Tüketimi	Yüksek
Güvenilirliği	Oldukça güvenilir
Kullanılan Anahtar	İki farklı anahtar kullanılır.
Devir Sayısı	1
Başlama Hızı	Hızlı

Herkese açık olan anahtar ve sadece alıcısında olan gizli anahtarlarda gerekli güvenliği sağlayabilmek adına seçilen boyut, 1024 bitten büyük olmalıdır. Boyutun bu kadar büyük seçilmesi, güvenlik açısından avantaj iken şifreleme, bu anahtar üzerinden bazı matematiksel işlemler gerektireceğinden zaman alıcı olacaktır. Tersine şifreli metni çözme işleminde de bu sefer karşımıza çarpanlara ayırma zorluğu geleceğinden deşifreleme de zaman alacaktır. Halka açık olan ve herkesten gizli tutulan anahtar olmak üzere iki anahtar çifti barındıran bu algoritmanın başlama hızı ise hızlı olacaktır (Beşkirli ve ark., 2019).

RSA şifreleme algoritması metinleri şifreleyerek başkası tarafından okunmaz hale getirmenin yanı sıra dijital ortamda verilerin imzalanması için de kullanılan bir algoritmadır Günümüzde de RSA şifreleme algoritması hala güvenilirliğini korumaktadır. Çünkü RSA' nın temelini modüler matematik oluşturmaktadır, kripto analizi de büyük tamsayıları asal çarpanlarına ayırmaya dayandığı için kırılması zor bir algoritmadır. Bu nedenlerden dolayı RSA algoritması geniş bir kullanım alanına sahiptir. Bu alanlar aşağıda listelenmiştir (Engin, 2020):

- Askeri sistemler
- Cep telefonları
- Uzaktan kumandalar
- Online bankacılık
- Online alışveriş
- Uydu alıcıları
- Sim kartlar

Yukarıda bahsedildiği gibi yaygın bir kullanım alanı olan RSA algoritması, Microsoft Internet Explorer gibi web tarayıcısı programları tarafından kullanılan SSL'de, kredi kartı işlemleri için Güvenli Elektronik İşlem (Secure Electronic Transaction, SET) protokolü, Mastercard ile Visa' da ve ayrıca Güvenli Köprü Metni Aktarım Protokolü (Secure Hypertext Transfer Protocol, S-HTTP) ve Güvenli / Geniş Alan Ağı (Secure / Wide Area Network, S/WAN) uygulamalarında kullanılmaktadır. Bunlara ek olarak Güvenli / Çok Amaçlı İnternet Posta Uzantıları (Secure / Multipurpose Internet Mail Extensions, S/MIME), Gizliliği Artırılmış Posta (Privacy-Enhanced Mail, PEM) ve Oldukça iyi Gizlilik (Pretty Good Privacy, PGP) gibi gizli haberleşme protokolleri temel olarak RSA kullanır (Engin, 2020). RSA algoritmasında matematiksel yöntemler ile çalışan iki ayrı anahtar bulunmaktadır. Bu anahtarlardan biri açık diğeri gizli anahtardan oluşmaktadır. Açık anahtar herkese paylaşılır, şifreli metin göndermek isteyen bir kullanıcı bu açık anahtarı kullanarak metni şifreler ve gönderir. Ancak şifreli metnin çözülebilmesi gizli anahtara sahip olan kullanıcıya bağlıdır. Şekil 3.20'de algoritmanın işlem adımları görülmektedir. Burada gösterilen (e, n) tamsayıları açık anahtarı, (d, n) tamsayıları ise gizli anahtarı ifade etmektedir. Şifreleme ve şifre çözme bu anahtarlar kullanılarak yapılan modüler üs alma işlemleriyle gerçekleştirilir.



Şekil 3.20. RSA algoritmasının akış şeması

RSA algoritmasının anahtar oluşturma adımları aşağıdaki gibidir (Beşkirli ve ark., 2019):

- p ve q iki asal sayı olarak seçilir, C ve M ise sırasıyla şifreli ve düz metni ifade eder.
- Bu iki asal sayının çarpımı $n = p * q$ mod işleminde kullanılmak üzere hesaplanır.
- Euler sayısı seçilen iki asal sayının birer eksiltilip çarpılması şeklinde $\varphi(n) = (p-1).(q-1)$ ifadesi ile hesaplanır.
- $1 < e < \varphi(n)$ aralığında ve $EBOB(\varphi(n), e) = 1$ şartını sağlayan rastgele bir e sayısı türetilir.
- (e, n) çifti açık anahtardır ve düz metni şifrelemek için kullanılır.
- $1 < d < \varphi(n)$ aralığında $(d.e)\, mod\, \varphi(n) = 1$ olmak üzere bir d sayısı türetilir.
- (d, n) şifrelenmiş metni çözmek için kullanılan gizli anahtardır.

Anahtar oluşturma adımları tamamlandıktan sonra ilk olarak açık anahtar (e, n) alıcı tarafından göndericiye iletilir. Gönderici, bu acık anahtarı ve $C \equiv M^e mod\, n$ eşitliğini kullanarak mesajı şifreler ve şifreli mesajı alıcıya gönderir. Alıcı da kendisinde bulunan gizli anahtar d ile $M \equiv C^d mod\, n$ eşitliğini kullanarak orijinal mesajı elde eder (Prajapati ve Buddhdev, 2014).

3.1.2.3. El-Gamal algoritması

El-Gamal algoritması, Diffie-Hellman protokolüne dayanan açık anahtarlı bir algoritmadır. Bu kripto sistem, 1985 yılında Taher El-Gamal tarafından tanımlanmıştır (Yousif ve ark., 2020). EL-Gamal algoritması, ilk olarak dijital imza için kullanılmış; ama sonra şifreleme ve deşifreleme işlemlerinde kullanılması için güncellenmiştir (Iswari, 2016). Bu algoritma RSA algoritmasına bir alternatif sunmaktadır. El-Gamal ve RSA algoritmaları arasındaki en önemli fark, RSA algoritmasının güvenliğinin büyük asal sayıları çarpanlarına ayırmanın zorluğuna dayanması, El-Gamal algoritmasının güvenliğinin ise büyük asal sayıların ayrık logaritma modülünü hesaplamanın zorluğuna dayanmasıdır. El-Gamal algoritmasının işlem adımları üç aşamaya ayrılır: Anahtar üretimi, şifreleme ve deşifreleme (Imran ve ark., 2020).

Anahtar üretim aşamasının işlem adımları aşağıda tanımlanmıştır:

- Rastgele büyük bir p asal sayısı seçilir.
- Çarpımsal üreteç elemanı olarak adlandırılan rastgele bir g sayısı seçilir.
- 1 ile p-1 arasında olacak şekilde rastgele bir x sayısı gizli anahtar olarak belirlenir.
- $y = g^x \pmod p$ formülü kullanılarak bir y sayısı hesaplanır ve bu açık anahtar olarak kullanılır.
- p, g ve y değerleri açık anahtar olarak paylaşılırken x değeri gizli anahtar olarak saklanmalıdır.

Anahtar üretim aşamasının tamamlanmasının ardından şifreleme aşamasına geçilir. Şifreleme aşamasının işlem adımları da aşağıda açıklanmıştır:

- Alıcının açık anahtarı (p, g, y) elde edilir.
- Rastgele bir tamsayı k seçilir.
- Şifreli metin çifti $c_1 = g^k \bmod p$ ve $c_2 = (m * y^k) \bmod p$ olacak şekilde hesaplanır. Formülde geçen m değeri şifrelenmek istenen mesajı ifade etmektedir.
- Şifreli metin çifti (c_1, c_2) alıcıya gönderilir.

Deşifre etme işlemi ise aşağıda ifade edildiği şekilde uygulanmaktadır:

- Gizli anahtar x kullanılarak c_1^{p-1-x} parametresi hesaplanır.
- Bir önceki adımda hesaplanan parametre ve c_2 değerleri kullanılarak gizli mesaj $m = (c_1^{p-1-x}) * c_2 mod\ p$ formülünden elde edilir.

3.1.3. Özet (Hash) algoritmaları

Güvenli bir sistemde veri bütünlüğü önemli bir kavramdır. Bu kavramın sağlanması için özet (hash) algoritmaları kullanılmaktadır. Bir özet algoritması, farklı boyutlarda girdi alan ve her defasında sabit boyutlu çıktı üreten bir fonksiyondur. Kriptografik özet fonksiyonları sayesinde kullanıcılar verilerdeki yetkisiz değişiklikleri tespit edebilmek için bir mesaj özeti oluşturabilirler. Bu özellikle kritik sistemlerde ve hassas veri tabanlarında oldukça önemlidir. Verinin kaynağını doğrulamak için özet fonksiyonları, diğer kriptografik yöntemlerle entegre edilebilir. Özet algoritmaları

şifreleme ile birleştirildiğinde verilerin kaynağını tanımlayan özel mesaj özetleri üretebilirler. Bu özel özetler, Özet Tabanlı Mesaj Doğrulama Kodu (Hash-based Message Authentication Code - HMAC) olarak adlandırılır (Pittalia, 2019). HMAC, şu anda kullanılan standart bir algoritmadır. HMAC algoritması veri kaynağının doğrulanmasını sağlar ve saldırganların ataklarını önler.

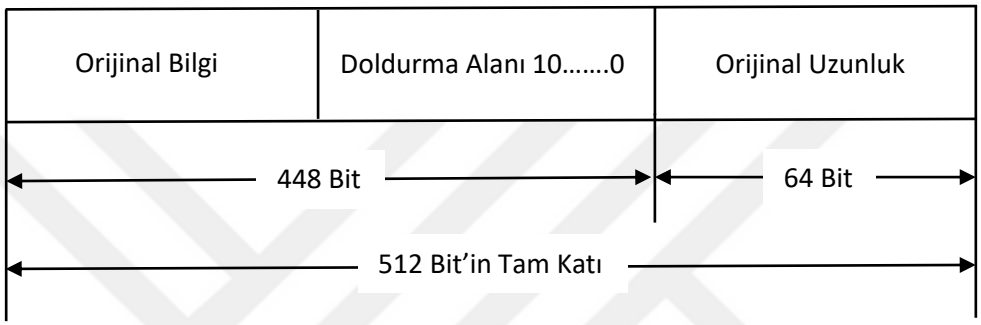
Bir özet fonksiyonu için karakter dizileri, binary dosyalar ve TCP paketleri gibi herhangi bir veri parçası girdi, yani mesaj olabilir. Tüm özet fonksiyonları için mesaj özetinden yani çıktıdan mesajın geri elde edilmesi imkansızdır. Dolayısıyla saldırganların oluşturulan özetten mesaja ulaşabilmelerinin hiçbir yolu yoktur. Bu özellik de özet algoritmalarını tek yönlü bir fonksiyon yapar. Özet fonksiyonlarının bir diğer özelliği de iki farklı mesajdan aynı özetin üretilmesinin çok zor olmasıdır. Bu da veri bütünlüğünün ve doğrulamasının sağlanması için oldukça büyük avantaj sağlamaktadır. Günümüzde en yaygın olarak kullanılan özet algoritmaları Mesaj Özeti 5 (Message Digest 5, MD5) ve Güvenli Karma Algoritması (Secure Hash Algorithm, SHA)' dir. Bölümün devamında bu algoritmalardan detaylı olarak bahsedilmiştir.

3.1.3.1. Message Digest 5 (MD5)

MD5, 20. yüzyılda 90'ların başında Massachusetts Teknoloji Enstitüsü (Massachusetts Institute of Technology, MIT) Computer Science Laboratuvarı ve RSA Data Security şirketinden Ronald L. Rivest tarafından ortaklaşa olarak geliştirilen bir özet algoritmasıdır (Zheng ve Jin, 2019). Algoritma, 90'lardan beri çeşitli programlama dillerinde veri bütünlüğünü sağlamak amacıyla yaygın olarak kullanılmıştır. Birçok veri tabanında, hatta Unix ve Linux işletim sistemlerinde bile kullanıcı parolalarını korumak için bu algoritmadan faydalanılır (Xia ve Ge, 2010). Ancak zaman geçtikçe algoritmanın güvenliği de zayıflamaktadır. MD5 algoritmasının tersi çevrilemez ve özet bilgiden orijinal bilgi tekrar elde edilemez. Bu yüzden algoritmanın güvenli olduğuna inanılmaktadır. Ancak bazı araştırmalar, MD5 algoritmasının çakışma ataklarıyla deşifre edilebileceğini göstermektedir ve bu durum algoritmanın uygulamadaki güvenliği açısından sorun teşkil etmektedir (Zheng ve Jin, 2019).

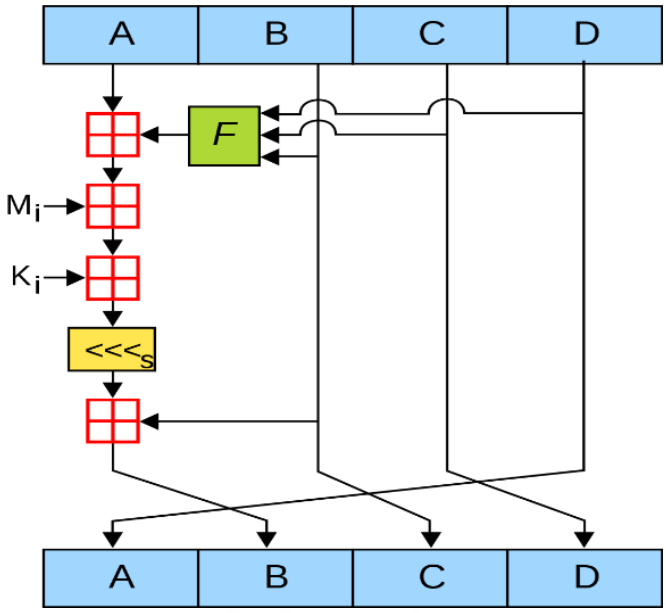
MD5 algoritması basit bir tasarıma sahiptir ve 128 bitlik bir özet üretir. Algoritma bazı başlangıç işlemlerinden sonra giriş metnini 16 tane 32 bitlik alt bloğa bölünmüş 512 bitlik bloklar halinde işler (Hossain ve ark., 2012). Algoritmanın çıktısı, birbirine bağlanan dört adet 32 bitlik blok kümesidir ve bunlar tek bir 128 bitlik özet değeri

oluşturur. Algoritma başlatılmadan önce giriş metninin uzunluğu 512 bitin katı olmalıdır. Bunun için mesajın sonuna önce 1 eklenir, sonra da gerektiği kadar 0 yerleştirilir (Pittalia, 2019). Bu ekleme işleminin sonucunda mesaj uzunluğu tam olarak 512'nin katının 64 eksiği olmalıdır. Çünkü daha sonra mesajın bu ekleme işleminden önceki uzunluğunun 64 bitlik temsili sonuca eklenir. Bu iki adım, farklı mesajların ekleme işleminden sonra aynı görünmemesini sağlarken aynı zamanda mesaj uzunluğunun da 512 bitin tam bir katı olmasını sağlar (Zheng ve Jin, 2019). Bu işlemden sonra oluşan veri modeli Şekil 3.21' de gösterilmiştir.



Şekil 3.21. MD5 algoritmasında kullanılan veri modeli

Başlangıç işlemlerinden sonra algoritmada zincirleme değişkenleri olarak adlandırılan dört adet değişken başlatılır. Bunların başlangıç değerleri; A = 0x01234567, B = 0x89abcdef, C = 0xfedcba98, D = 0x76543210 şeklindedir (Hossain ve ark., 2012). Her 512 bitlik mesaj bloğu için algoritma ayrı ayrı çalıştırılır. Ana döngü birbirine çok benzeyen dört turdan oluşur. Her tur da modüler toplama, sola kaydırma ve bir lineer olmayan fonksiyona dayalı 16 benzer işlem vardır (Aggarwal ve ark., 2014). Şekil 3.22'de bir turda yapılan işlemler gösterilmektedir.



Şekil 3.22. MD5 algoritmasının bir turu

Algoritmadaki tekrar eden işlemler toplamda 64 kez uygulanmaktadır ve 16 işlemden oluşan 4 turda gruplandırılmıştır. F, lineer olmayan bir fonksiyondur ve her turda farklı bir fonksiyon kullanılır. M_i, mesaj girdisinin 32 bitlik bir bloğunu göstermektedir. K_i ise 32 bitlik bir sabittir ve her işlem için farklıdır, s de s bitlik sola kaydırma işlemi yapılacağını ifade eder (Aggarwal ve ark., 2014). F fonksiyonu, B, C ve D değerlerini kullanan doğrusal olmayan bir fonksiyondur. Algoritma her 512 bitlik mesaj bloğunu bu değerleri değiştirmek için kullanır. Algoritmada kullanılan toplam 4 adet F fonksiyonu bulunmaktadır. Bu fonksiyonlar aşağıda ifade edilmiştir:

- F(B,C,D)=(B AND C) OR (NOT B AND D)
- G(B,C,D)= (B AND D) OR (C AND NOT D)
- H(B,C,D)= B XOR C XOR D
- I(B,C,D)= C XOR (B OR NOT D)

MD5 algoritmasının her turunda farklı bir fonksiyon kullanılır. Dolayısıyla 4 turun her biri için bir fonksiyon olmak üzere yukarıda ifade edilen 4 fonksiyon kullanılmaktadır. Bu 4 turun sonunda hash değeri olarak da ifade edilen 128 bitlik mesaj özetinin elde edilmesiyle algoritma tamamlanmış olur.

3.1.3.2. Secure Hash Algorithm (SHA)

Güvenli Karma Algoritması (Secure Hash Algorithm, SHA), NIST tarafından yayınlanan bir dizi kriptografik özet fonksiyonudur. NIST, 1993 yılında SHA-0 algoritmasını Federal Bilgi İşleme Standartları Yayını (Federal Information Processing Standards Publications, FIPS-PUB) 180" adlı dokümanla duyurmuştur. Sonrasında da 1995 yılında "FIPS PUB 180-1" adlı dokümanla SHA-0'ın yerine standart olarak kullanılması için revize edilmiş olan SHA-1 (SHA-160 olarak da bilinir) algoritmasını açıklamıştır. 2001 yılında ise NIST, SHA-160, SHA-256, SHA-384 ve SHA-512 olmak üzere dört algoritma içeren "FIPS PUB 180-2" adlı dokümanı yayınlamıştır. 2004 yılında bu doküman, 3DES ile aynı anahtar uzunluğuna sahip SHA-224 algoritmasının da eklenmesiyle güncellenmiştir (Lin ve ark, 2011).

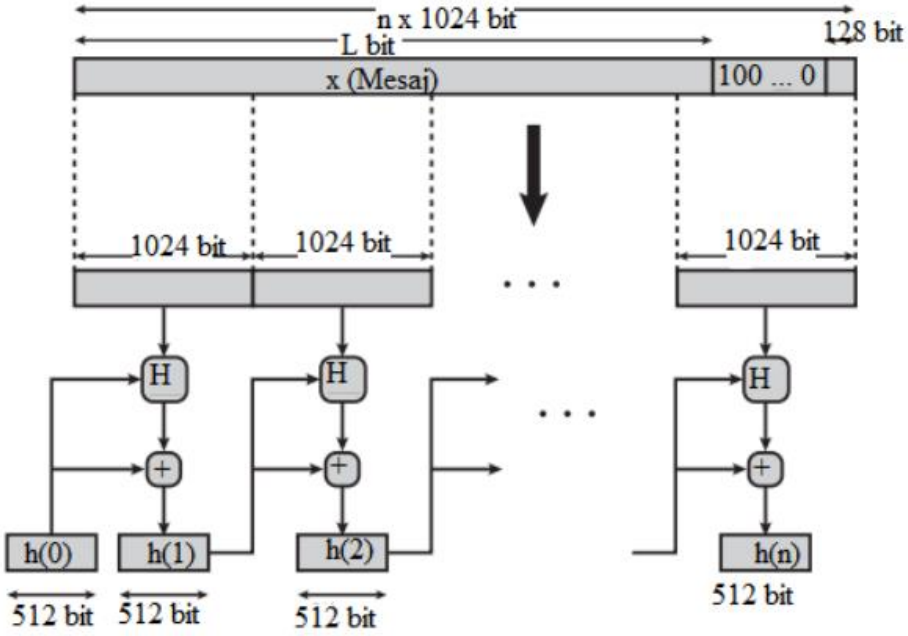
SHA' da işlem yapılabilecek maksimum mesaj uzunluğu, blok, kelime ve mesaj özeti boyutları algoritmanın varyasyonlarına göre değişiklik göstermektedir. Örnek olarak SHA-1, $2^{64} - 1$ bitlik mesaj girdisi kapasitesine karşılık 160 bitlik özet değeri üretmektedir (Sumagita ve Riadi, 2018). Diğer varyasyonların mesaj girdi kapasiteleri, kelime, blok ve özet değeri uzunlukları Çizelge 3.2' de gösterilmiştir.

Çizelge 3.2. SHA varyasyonlarının özellikleri

Algoritma	Mesaj Uzunluğu (bit)	Blok Uzunluğu (bit)	Kelime Uzunluğu (bit)	Mesaj Özeti Uzunluğu (bit)
SHA-1	$< 2^{64}$	512	32	160
SHA-256	$< 2^{64}$	512	32	256
SHA-384	$< 2^{128}$	1024	64	384
SHA-512	$< 2^{128}$	1024	64	512

SHA-1 algoritması yeterince güvenli değildir. Rijmen ve Oswald (2005), yaptıkları çalışmada SHA-1'in indirgenmiş versiyonunda (80 turun sadece 53'ünü kullanarak) uyguladıkları 2^{80} operasyon neticesinde algoritmanın ürettiği özet değerlerinde çakışma olduğunu saptamışlardır. SHA-256 ve SHA-384 algoritmaları, daha güvenli olmalarına rağmen özetleme işleminin uzun sürmesinden dolayı pek tercih edilmemektedir. SHA-512 ise SHA-1'in üzerinde MD4 tabanlı iyileştirme yapılarak geliştirilmiş bir versiyondur. SHA-512, bir hash fonksiyonunun üretebildiği en uzun özet değerine sahip olduğu için güvenli olarak nitelendirilebilir. Bu uzun hash değeri, SHA-512'yi saldırılara karşı diğer özet algoritmalarına göre daha dayanıklı hale getirir. Bu yüzden SHA 512 güçlü, sağlam ve hızlı bir özet fonksiyonu olarak kabul edilir (Sumagita

ve Riadi, 2018). SHA-512, Ron Rivest tarafından geliştirilmiş tek yönlü özet fonksiyonu kullanan bir algoritmadır ve 512 bit uzunluğunda bir özet değeri üretir. Şekil 3.23'te algoritmanın bu özet üretimini nasıl gerçekleştirdiği gösterilmiştir.



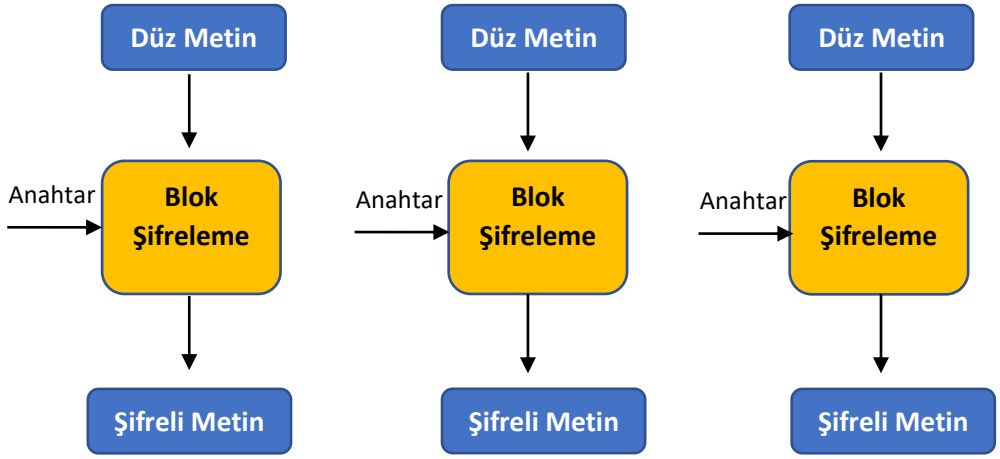
Şekil 3.23. SHA-512 algoritmasının çalışma yapısı

SHA-512 algoritması, 1024 bitlik bloklar üzerinde işlem yapar. Dolayısıyla yapılacak ilk işlem mesaj girdisinin 1024 bitin tam katı olacak şekilde düzenlenmesidir. Bunun için ilki "1", geri kalanı "0" olacak şekilde bir bit dizisi mesajın sonuna eklenir. Burada dikkat edilmesi gereken nokta, bu ekleme işleminden sonra mesaj uzunluğunun 1024'ün tam katının 128 eksiği kadar olması gerektiğidir. Çünkü bit dizisinin eklenmesinin ardından orijinal mesaj uzunluğunu ifade eden 128 bitlik bir değer daha mesajın sonuna eklenecek ve böylece mesaj girdisinin uzunluğu 1024'ün tam katı haline gelecektir (Stallings, 2011). Mesaj uzunluğu düzenlendikten sonra Şekil 3.23' te görüldüğü üzere bu mesaj 1024 bitlik alt bloklara bölünür ve her mesaj bloğu 80 turluk bir işlemden geçirilerek 512 bit uzunluğunda bir özet değeri elde edilir. Her bloktan elde edilen özet değeri bir sonraki bloğun işlenmesinde kullanılır ve sonuncu bloğun işlenmesiyle elde edilen değer nihai özet değerini oluşturur.

3.1.4. Blok şifre çalışma modları

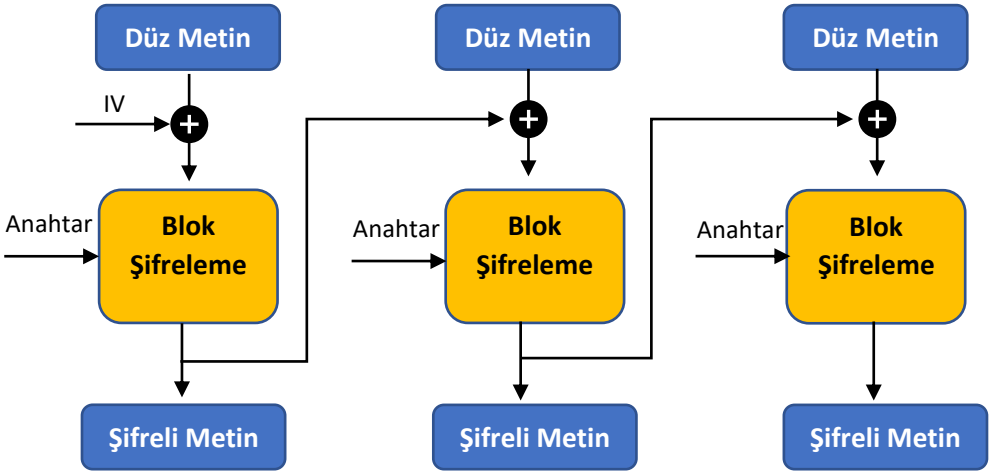
Blok şifre çalışma modları ya da diğer adıyla çalışma kipleri, DES veya AES gibi bir blok şifreleme algoritmasını, bazı basit işlemler ve geri bildirim mekanizmalarıyla birleştirerek kullanmanın özel yollarını ifade eder (Rogaway, 2011). Çeşitli şifreleme modları bulunmaktadır ve bunlar temelde deterministik ve olasılıksal olmak üzere ikiye ayrılır. Deterministik şifrelemede anahtar değişmezse, belirli bir düz metin sabit bir şifreli metne eşlenir. Diğer yandan olasılıksal şifreleme şemaları, deterministik olmayan şifreli metinler elde etmek için rastgelelik kullanır. Bu şifreleme şemaları kendi içerisinde blok ve akış şifreleme modları olarak ikiye ayrılır. Tüm bu çalışma modlarının ana amacı iletişim süreci boyunca gizliliği ve özgünlüğü sağlamak ve sürdürmektir (NIST, 1998).

Deterministik şemalardaki tek mod, Elektronik Kod Kitabı (Electronic Code Book, ECB)' dır. ECB, blok şifreleyici kullanmanın en basit yoludur. Düz metin, n bitlik bloklara bölünür ve bu bloklar DES veya AES gibi bir blok şifreleme algoritması kullanılarak birbirinden bağımsız olarak şifrelenir. Bu, senkronizasyona gerek olmadığı anlamına gelir, yani bloklar herhangi bir sırayla şifrelenebilir ve ardından birleştirilebilir. Benzer olarak şifre çözme işlemi bunun tersi şeklinde gerçekleşir ve Şekil 3.24, ECB ile şifreleme sürecini göstermektedir. ECB modunun avantajlarından birisi senkronizasyon gerektirmediği için alıcının kendisine ulaşan şifreli blokları diğerlerini almadan çözebilmesidir. Ayrıca, bazı iletim problemleriyle ilgili bit hataları sadece ilgili blokları etkiler. Bunların yanı sıra ECB'nin uygulanması, onun paralelleştirme yeteneğinden dolayı yeterince hızlı olarak kabul edilebilir. Başka bir deyişle farklı veri blokları farklı şifreleme birimlerinde eş zamanlı olarak şifrelenebilir. Bu paralelleştirme özelliği ve hızından dolayı veritabanı uygulamalarında kullanılmıştır. Çünkü yapılan girişlerin eklenmesi veya silinmesi diğer kayıtlardan bağımsız olarak gerçekleştirilebilmektedir (Menezes ve ark., 1997). Ancak ECB modu şifreleme yapmak için en iyi seçenek değildir. Şifrelemede kullanılan anahtar değişmediği sürece aynı düz metin blokları, aynı şifreli metin bloklarını üretir. Bu da onu fazlasıyla deterministik yapar. Deterministik olması da rastlantısallıktan uzak olması anlamına gelir. Ayrıca eğer şifreli metinde başlık veya alt bilgi gibi aynı yerlerde tekrar eden parçalar varsa bir saldırgan bu bilgileri kullanarak düz metne ulaşabilir.



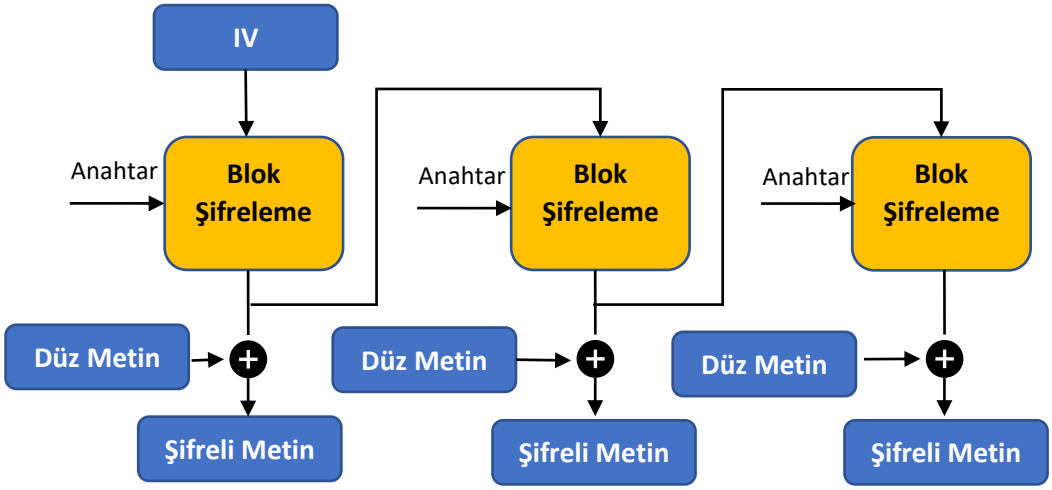
Şekil 3.24. ECB moduyla şifreleme

Bahsedildiği gibi determinizm şifrelemeyi saldırılara karşı savunmasız yapar ve bu yüzden şifrelemeyi olasılıksal bir şekilde uygulamak önemlidir. Başka bir deyişle, aynı düz metin, her şifrelendiğinde farklı şifreli metinler üretmelidir. Bu özellik, Şekil 3.25' te gösterilen Şifre Blok Zincirleme (Cipher Block Chaining, CBC) modu kullanılarak başarılabilir. Bu modda bloklar, bütün bir mesaj olarak kabul edilir ve birlikte zincirlenir. Öyle ki her düz metin bloğunun etkisi birçok şifreli metin bloğuna yayılır. CBC modu, şifrelemeyi olasılıksal yapmak için Başlangıç Vektörü (Initialization Vector, IV) olarak adlandırılan bir tür rastgelelik kullanır. IV, gizli olmak zorunda değildir ama sadece bir kez kullanılmalıdır. Bir rastgele sayı üretici kullanarak veya başka şekillerde üretilebilir (Dworkin, 2001). İlk düz metin bloğu IV ile XOR işlemine tabi tutulur ve ardından bir blok şifreleyici kullanılarak şifrelenir. Şekil 3.25' te görüldüğü gibi birbirini izleyen bloklar için şifreleme algoritmasına bir geri bildirim mekanizması vardır. Blok şifreleme algoritmasına bir sonraki girdiyi üretmek için daha önce üretilen şifreli metin geri beslenir ve düz metin bloğu ile XOR'lanır. Şifre çözme süreci de bu işlemlerin tersidir. CBC modu paralel olarak gerçekleştirilememesine rağmen en yaygın kullanılan şifreleme modudur. Bunun nedeni, her düz metin bloğunun, sonraki blokların şifrelenmesini etkilemesidir. Olası bit hatalarının sonraki şifreli metin blokları üzerinde büyük bir etkisinin olacağı düşünebilir. Ancak bu hatalar, şifre çözme işleminde kurtarılır ve aynı bit hatalarıyla düz metin üretilir. Bu, CBC modunun kendini kurtarma özelliği olarak adlandırılır ve şifre çözme işleminin paralelleştirilmesini mümkün hale getirir. ECB modunun aksine eğer IV her şifreleme işlemi için doğru şekilde seçilmişse yerine koyma atakları uygulanamaz (Rogaway, 2011). Ancak, şifreli metin üzerindeki herhangi bir değişiklik düz metinde istenmeyen bazı rastgele değişikliklere sebep olur.



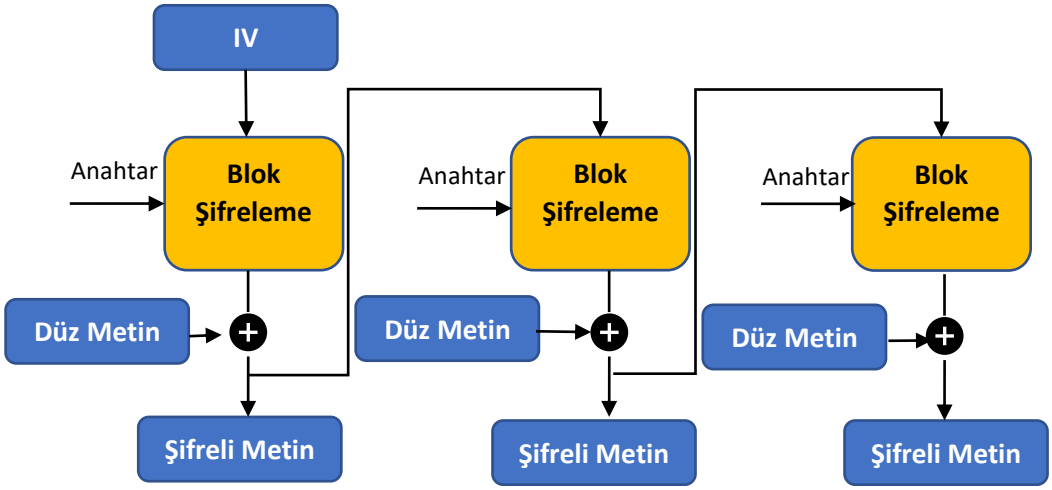
Şekil 3.25. CBC moduyla şifreleme

Blok şifreleme algoritmaları, Çıktı Geri Besleme (Output Feedback, OFB) ve Şifre Geri Besleme (Cipher Feedback, CFB) modlarıyla akış şifreleyicisi olarak kullanılabilir. Yani bu şifreleme şemaları, blok şifreleme algoritmalarını bir anahtar akışı üretici olarak kullanır. Blok şifreleyici için ilk girdi CBC modunda olduğu gibi yine başlangıç vektörüdür. N bit uzunluğundaki düz metin şifreleme operasyonu sonucunda üretilen n bitlik anahtar akışıyla XOR'lanır ve şifreli metin oluşturulur. Gelecek anahtar akışı, önceki üretilen akışın blok şifreleyiciye geri beslenmesiyle elde edilir. OFB şifreleme şeması bit düzeyinde değil, blok düzeyinde akış sağlar (NIST, 1998). OFB modu, bir blok şifreleme algoritmasını, onu standart bir akış şifreleme algoritmasına çok benzer kılacak şekilde senkronize bir akış şifreleyici olarak çalıştırır. Ne düz metin ne de şifreli metin oluşturulan anahtar akışını etkilemez. Şifreleme ve şifre çözme süreci tamamen aynıdır. Şekil 3.26, OFB moduyla yapılan şifreleme işlemini göstermektedir. Bu modun avantajlarından bir tanesi geri besleme mekanizmasının daha veri ulaştırılmadan önce çalıştırılabilmesidir. Diğer taraftan her anahtar akışı, önceki tüm akışlara bağlı olduğu için şifreleme ve şifre çözme işlemleri paralelleştirilemez (Menezes ve ark., 1997).



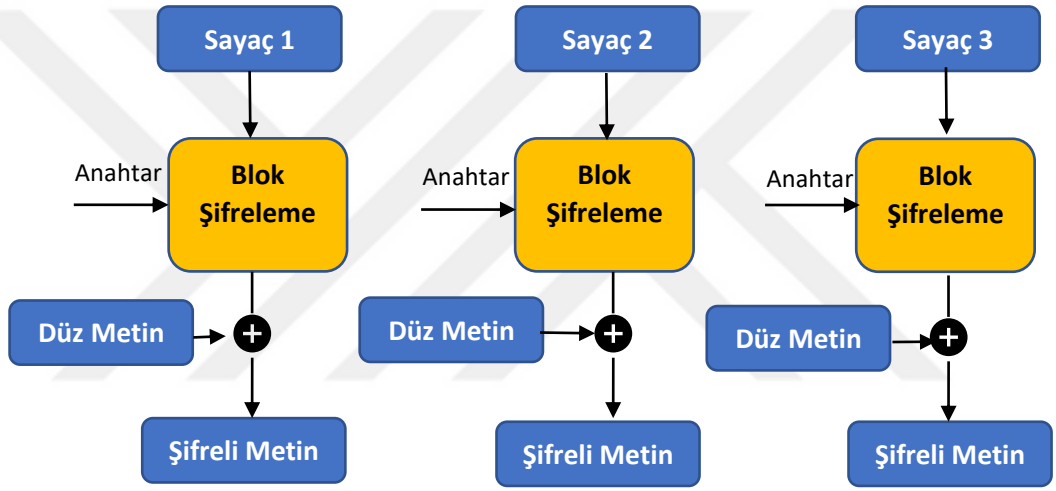
Şekil 3.26. OFB moduyla şifreleme

CFB şeması da OFB moduna çok benzerdir. Bu şema da OFB' de olduğu gibi bir blok şifreleyiciyi, bir akış şifre üretici olarak kullanır. Fakat OFB' den farklı olarak, Şekil 3.27' da görüldüğü gibi sonraki akışın üretilmesi için blok şifreleyiciye önceki anahtar akışı yerine şifreli metin geri beslenir. İlk n bitlik anahtar akışı, IV' nin şifrelenmesiyle oluşturulur ve şifrelenen IV, düz metinle XOR'lanır. Böylece n bitlik şifreli metin elde edilmiş olur. OFB şemasına benzer şekilde şifreleme ve şifre çözme süreçleri birbirinin aynısıdır ve şifreleme işlemi yine paralelleştirilemez. Ancak OFB' nin aksine anahtar akışının üretimi şifreli metnin bir fonksiyonu olduğundan CFB modu asenkronize bir akış şifreleyicisidir ve bu durum da şifre çözme işleminin paralelleştirilebilmesini mümkün kılar (Dworkin, 2001).



Şekil 3.27. CFB moduyla şifreleme

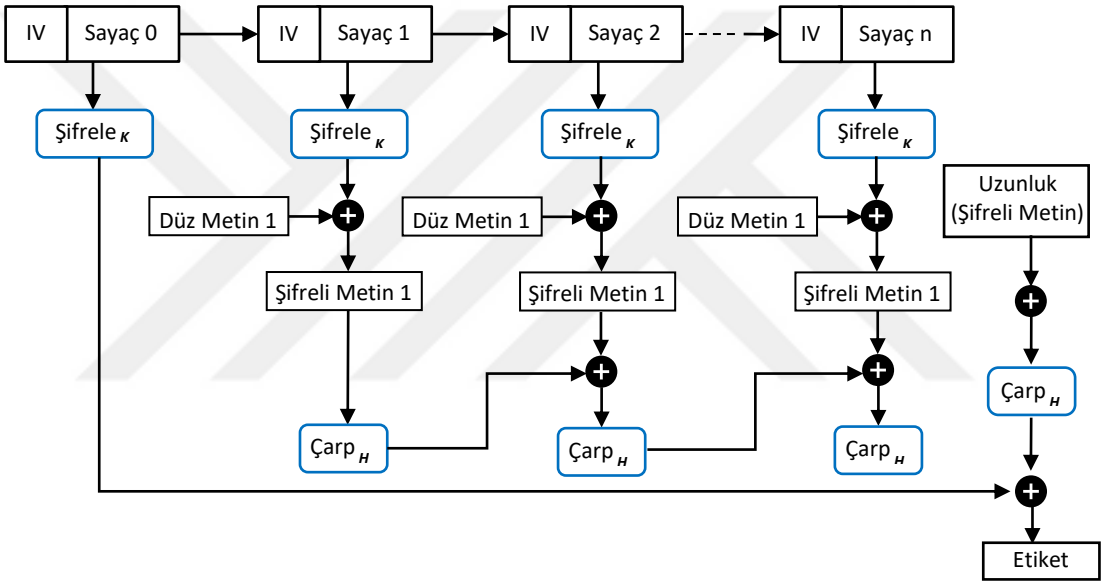
1979’ da Diffie ve Hellman tarafından tanıtılan Sayaç Modu (Counter Mode, CTR), OFB ve CFB şemalarına çok benzerdir. CTR’ de benzetildiği diğer şemalarda olduğu gibi bir blok şifreleme algoritmasını onun akış üreticisi olarak kullanır. Fakat Şekil 3.28’ de görüldüğü üzere diğerlerinden farklı şekilde blok şifreleme algoritmasına girdi olarak bir sayaç değeri verilir. Bu sayacın değeri, her yeni bir akış oluşturulduğunda değiştirilir. Sayaç değeri, sürekli olarak hem tekrar etmeyen hem de mümkün mertebe rastlantısal sayılar üreten matematiksel bir fonksiyona bağlıdır. CTR modu yaygın olarak kullanılır ve tavsiye edilir. Ek olarak CBC, CFB ve OFB şemalarının aksine hem şifreleme hem de şifre çözme işlemleri paralelleştirilebilir (Rogaway, 2011). Bu olanak da işlem açısından avantaj sağlar.



Şekil 3.28. CTR moduyla şifreleme

Simetrik kripto sistemlerde, kullanılan anahtarın ve şifresi çözülen verinin birbiriyle eşleşip eşleşmediğini kontrol etmenin bir yolu yoktur. Yani eldeki anahtar hatalı olsa bile yine de bir dizi şifreli veri çözülecektir. Ancak alıcı taraf, çözülen verinin doğru olup olmadığını bilemez. Bu yüzden alıcının şifre çözme sürecinin geçerliliğini kontrol edebilmesi için gönderici taraf, şifreleme işleminin yanı sıra bazı doğrulama metotlarını eklemeye ihtiyaç duyar. Genellikle doğrulama metodu olarak hash fonksiyonları kullanılır. İlişkili Verilerle Kimliği Doğrulanmış Şifreleme (Authenticated Encryption with Associated Data, AEAD), şifrelemeyle kimlik doğrulamayı birleştiren ve yaygın olarak kullanılan algoritmalardan birisidir (Kao ve ark., 2021). Galois Sayaç Modu (Galois Counter Mod, GCM), CTR modu şifrelemesini ve Galois Mesaj Doğrulama Kodu (Galois Message Authentication Code, GMAC) algoritmasını birleştirerek AEAD’ı gerçekleştirmeyi başarır. GCM, NIST tarafından Special Publication 800-38D adlı

dokümantasyonla tanıtılmış ve kimliği doğrulanmış şifreleme için standardize edilmiş bir blok şifreleme çalışma modudur (Dworkin, 2007). Şekil 3.29, GCM' nin çalışma mekanizmasını göstermektedir. 128 bit uzunluğundaki kimlik doğrulama anahtarı *H*, tüm bitleri 0 olan bir girdi bloğunun, asıl şifreleme anahtarı *K*'nın kullanılarak şifrelenmesiyle elde edilir. Daha sonra kimlik doğrulama etiketinin hesaplanması için her 16 baytlık veri bloğu için, sonlu alanda *H* ile 128 bitlik bir çarpma işlemi uygulanır. GCM kimlik doğrulaması için gereken temel işlem, gizli bir sabit eleman *H* ile Galois alan çarpmasıdır (Kasper ve Schwabe, 2009). Elde edilen çarpımlar, bir sonraki şifrelenmiş blokla XOR'lanıp yeniden bir çarpım işlemine beslenir ve bu süreç tüm bloklar için uygulanarak sonunda kimlik doğrulama etiketi elde edilir.



Şekil 3.29. GCM moduyla şifreleme

Şifreleme işleminden sonra oluşturulan kimlik doğrulama etiketi şifrelenmiş veriyle birlikte alıcıya gönderilir. Alıcı da kendisine ulaşan şifreli veriyi kullanarak göndericinin yaptığı gibi Galois alanında çarpım işlemlerini uygular ve bir etiket de kendisi türetir. Daha sonra göndericiden gelen etiketle kendi türettiği etiketi karşılaştırır ve eğer etiketler eşleşiyorsa kendisine ulaşan şifreli veriyi doğrulamış olur. Special Publication 800-38D adlı dokümantasyona göre GCM, AES algoritması kullanılarak çalıştırılmalıdır. Bu gereklilik kullanılacak blok şifreleme algoritmasının hem NIST onaylı olması hem de 128 bitlik blok boyutuna sahip olması şartlarından kaynaklanmaktadır ve bu gereksinimleri karşılayan tek algoritma ise AES'tir. IDEA gibi NIST onaylı diğer blok şifreleyeciler 64 bit bloklar üzerinde çalışmaktadır. GCM,

kanıtlanabilir iyi bir güvenlik garantisine sahiptir. IPSec, Medya Erişim Kontrolü Güvenliği (Media Access Control Security, MACSec) ve TLS de dahil olmak üzere yaygın olarak birçok yerde kullanılır. GCM hem donanımda hem de yazılımda verimli olabilir ve ayrıca Intel mikroişlemcilerde, GCM için doğrudan donanım desteği sunulmaktadır (Rogaway, 2011).

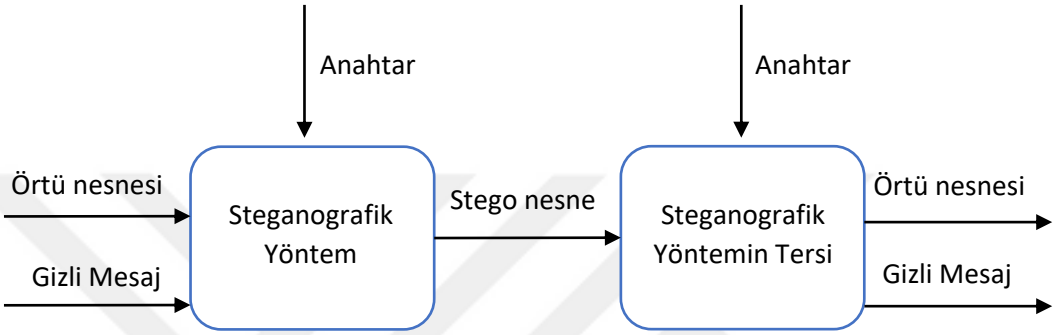
3.2. Steganografi

Steganografi, veri gizleme bilimidir. Gizli bilgiyi gözlemlenemeyeceği bir ortamda saklayarak onun güvenliğini sağlayan yöntemleri ifade eder. Son yıllarda steganografi yöntemleri fotoğraf, ses ve video verileri gibi farklı dijital ortamlarda kullanılmaya başlandı. Bu yöntemler, dijital ortamları bir örtü nesnesi olarak kullanmakta ve dijital formdaki gizli veriyi bu ortamların içerisine yerleştirmektedir. Örtü nesnesi olarak farklı dijital ortamlar seçilebildiği gibi gizlenecek bilgi de metin veya resim gibi her türlü veri olabilir (Smitha ve Baburaj, 2016). Steganografinin geniş bir uygulama alanı vardır. Askeri iletişim sistemleri gibi gizli iletişim sistemleri, yüksek seviye güvenliğe ihtiyaç duyar ve bu güvenliğin sağlanması için olası çözümlerden birisi de steganografidir (Petitcolas ve ark., 1999). Steganografinin yaygın kullanım alanlarından birisi de veri mülkiyeti ve telif haklarını korumak için uygulanan sayısal damgalama yöntemidir.

Steganografi, bilginin iletimini gizli olarak sağladığı için şifrelemeye göre daha az dikkat çeken bir yöntemdir. Bu yüzden birçok uygulamada bazı taraflarca şifreleme yerine tercih edilmektedir (Petitcolas ve ark., 1999). Her iki yöntemin de ortak unsurlardan birisi, bilgiyi güvence altına alacak bir anahtar olmasının gerekliliğidir. Anahtarın bilinmemesi durumunda, bilginin gizlenmesi veya şifrelenmesi için kullanılan yöntem bilinse bile o bilgi elde edilemez. Bu durum bilgi güvenliğinin artırılmasını sağlamak için kullanılan yöntemde bir anahtara sahip olunmasının önemini göstermektedir.

Steganografi için üç önemli kriter bulunmaktadır. Bunlar algılanamazlık, uygunluk ve geri çevrilebilirliktir. Gizli mesajın hem işitsel hem de görsel olmak üzere insan duyuları tarafından algılanamaz olması gerekmektedir. Örneğin örtü nesnesi bir fotoğrafsa bu fotoğrafa gizli mesaj yerleştirildikten sonra fotoğrafta meydana gelen değişiklik gözle fark edilemeyecek kadar olmalıdır veya örtü nesnesi bir ses dosyası olarak belirlenmişse stego-sesteki değişiklik kulakla ayırt edilemeyecek kadar olmalıdır.

Bir diğer kriter ise gizli mesajın saklanması için en uygun ortamın belirlenmesidir. Öyle ki yapılan gizleme işleminden sonra kullanılan örtü nesnesinin kalitesinde bir değişiklik olmamalıdır. Bunun için özellikle kullanılacak örtü nesnesinin boyutu, gizlenecek mesajın boyutuna göre belirlenmelidir. Başka bir kriter de geri çevrilebilirliktir, steganografiyle gizlenen mesaj, ihtiyaç duyulduğu zaman tekrar ortaya çıkarılabilmelidir. Şekil 3.30' da mesajın gizlenmesi ve gizlendikten sonra da geri elde edilmesinin nasıl gerçekleştirildiğini ifade eden temel bir steganografi modeli gösterilmiştir.



Şekil 3.30. Temel bir steganografi modeli

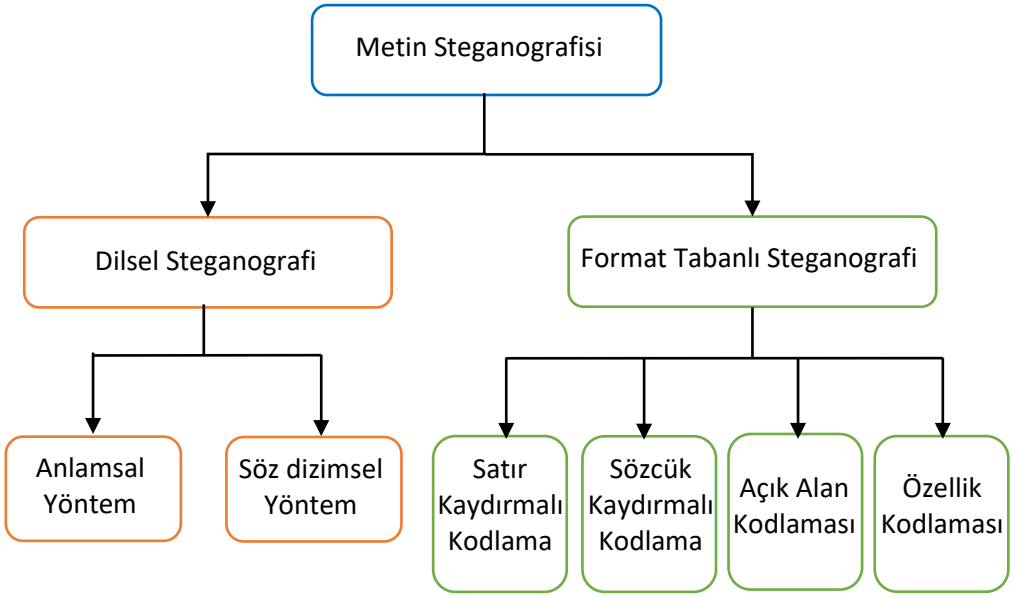
Bir steganografik sistem, temel olarak 5 bileşenden oluşur. Bunlar örtü nesnesi, gizli mesaj, anahtar, steganografik yöntem ve stego nesnedir. İlk olarak gizli mesajın yerleştirilebilmesi için uygun bir örtü nesnesi belirlenmelidir. Gizli mesaj bu örtü nesnesine, bir steganografik algoritma vasıtasıyla gömülür. Algoritmada, gizleme işleminin güvenliğini artırmak için bir stego-anahtar da kullanılabilir. Anahtar kullanılması durumunda gömülü mesajın tekrar çıkarılabilmesi için de yine bu anahtara ihtiyaç duyulur. Steganografi algoritmanın çıktısı, bir stego-nesnedir. Bu stego-nesne ve örtü nesnesinin birbiriyle aynı veri tipinde olması gerekir, fakat gömülü mesaj bunlardan farklı bir veri tipinde olabilir (Al-Ani ve ark., 2010). Gizleme işleminin tamamlanmasının ardından elde edilen stego-nesne alıcıya gönderilir. Alıcı da bu mesajın gizlenmesi için kullanılan steganografi algoritmasını tersine çalıştırarak gömülü mesajı stego-nesneden çıkartır. Steganografi teknikleri, kullanılan örtü nesnesinin türüne göre çeşitlere ayrılmaktadır. Temel olarak üç çeşit teknikten bahsedilebilir. Bunlar metin, ses ve görüntü steganografisidir. Bölümün devamında bu tekniklerin detaylarından bahsedilmiştir.

3.2.1. Metin (text) Steganografi

Metin içerisine bir başka metni gizleme işlemi, metnin formatını veya harfler gibi metin öğelerindeki belirli özellikleri değiştirerek gerçekleştirilebilir (Bennett, 2004). Dosya formatı, dokümanın içeriğini, sayfa düzenini ve TeX vb. gibi kullanılan dizgi sistemini tanımlar. Kodlama yöntemlerinin amacı gürültü içerseler bile saldırılarla deşifre edilemeyecek değişiklikler oluşturmaktır (Brassil ve ark., 1995). Bu yöntemler kolaylıkla çözülememeli ve meydana gelen görsel değişiklikler de minimum seviyede olmalıdır. Metin steganografisi, steganografinin en zor türüdür. Bu büyük ölçüde metin dosyalarının resim ve ses dosyalarına nazaran gürültü gibi gereksiz bilgileri daha az içermesinden kaynaklanmaktadır (Shirali-Shahreza, 2008). Resim gibi diğer dosya türlerinin yapısı bizim gözlemlediğimizden farklıyken metin dosyalarının yapısı bizim gözlemlediğimizle aynıdır. Bu nedenle metin dosyalarında ilgili çıktıda algılanabilir bir farklılık meydana getirmeden dosya yapısı değiştirilerek bilgiler gizlenmelidir.

Resimler, sesler ve video klipler gibi diğer medya ortamlarının aksine metin belgeleri, çok eski zamanlardan beri yaygındır (Popa, 1998). Matbaa makinesinin icadından sonra bile kitapların çoğu ve belgeler yalnızca metin içermekteydi. Günümüzde bu durum biraz değişmiş olsa da özellikle iletişimde metin formatının kullanılması hala çok yaygındır. Çünkü metinler daha az bellek gereksinimine ihtiyaç duyar ve böylece iletişim kurarken daha fazla bilgi paylaşılabilir (Sahni, 2019).

Metin steganografisi genel olarak dilsel (linguistic) ve format tabanlı (format based) olmak üzere ikiye ayrılır. Dilsel steganografi, kendi içerisinde anlamsal (semantic) ve sözdizimsel (syntactic) yöntem olarak ikiye ayrılır (Bennett, 2004). Format tabanlı steganografi ise kendi içerisinde satır kaydırmalı kodlama (line-shift encoding), sözcük kaydırmalı kodlama (word-shift encoding), açık alan kodlaması (open-space encoding) ve özellik kodlaması (feature encoding) olmak üzere dörde ayrılır (Popa, 1998). Bu sınıflandırma, Şekil 3.31’ de gösterilmiştir.



Şekil 3.31. Metin steganografisinin türleri

Format tabanlı yöntemler, boşluklar gibi metnin fiziksel biçimlendirmesini kullanarak bilgileri gizler. Bu yöntemler bilgiyi gizlemek için genellikle mevcut metni steganografik metin ile değiştirir. Boşluk veya görüntülenmeyen karakterlerin eklenmesi, baştan sona dağıtılan kasıtlı yazım hataları, metin ve yazı tiplerinin yeniden boyutlandırılması, metin steganografisinde kullanılan format tabanlı yöntemlerin bazılarıdır (Sahni, 2019). Kasıtlı yazım hataları ve boşluk ekleme gibi bu yöntemlerin bazıları, ara sıra yapılan yazım hatalarını görmezden gelen insanlar tarafından fark edilemeyebilir, ancak bir bilgisayar tarafından genellikle kolayca algılanır. Öte yandan bir bilgisayar, özellikle metnin yeniden boyutlandırılmasının beklenebileceği şekil vb. figürler içeren bir belgede yalnızca metnin içeriğine odaklanıyorsa yazı tipinin yeniden boyutlandırılmasını bir sorun olarak tanımlamayabilir. Ancak bir insan garip bir font boyutunu neredeyse anında algılayabilir. Ayrıca orijinal düz metin mevcutsa, bu düz metni şüpheli steganografik metinle karşılaştırmak, metnin manipüle edilmiş kısımlarını oldukça görünür hale getirecektir (Bennett, 2004).

Format tabanlı yöntemlerden bahsedeceğimiz ilki line-shift encoding olarak da bilinen satır kaydırmalı kodlamadır. Bu teknik, metin satırlarının konumlarını dikey olarak kaydırarak dokümanı değiştirir ve hem sayfa görüntüsü hem de dosya biçimi üzerine uygulanabilir (Popa, 1998). Belirli bir belgeye atanan kod kelimesi, bu belgeye eklenecek metin satırlarını belirtir. Bu kodlama aşağı kaydırılan satırlar için “0”, yukarı kaydırılan satırlar için “1” olacak şekilde belirlenebilir. Ama yukarı kaydırılmış bir satır

için “-1”, hareket ettirilmemiş bir satır için “0” ve aşağı kaydırılan bir satır için "+1" olacak şekilde bir kodlama da yapılabilir. Bu teknik, performans ve dayanıklılık elde etmek için diferansiyel kodlama tekniğini kullanabilir. Gizlenebilen her bir kod kelimesinin uzunluğu her satırı değiştiren tekniğe kıyasla azaltılmış olur, fakat gizleme kapasitesi hala yüksektir. Örneğin 40 satırlı sayfalardan oluşan bir belgede, sayfa başına $2^{20} = 1.048.576$ farklı kod kelimesi yerleştirilebilir. Kodlayıcı, dosyaya gömülmek istenen işaretlere göre satırları yukarı ve aşağı kaydırır. Kod çözücü de her iki komşu satırın arasındaki mesafeyi ölçer. Bu işlem iki farklı teknik kullanılarak yapılabilir. Kod çözücü ya komşu satırların taban çizgileri arasındaki mesafeyi ölçer ya da bu satırların merkezleri arasındaki mesafeyi ölçer (Sahni, 2019).

Bir diğer yöntem de word-shift encoding olarak da bilinen sözcük kaydırmalı kodlamadır. Bu teknik, işaretleri gömmek için satırlardaki sözcüklerin konumlarını yatay olarak kaydırarak dokümanı değiştirir. Bu tekniğin uygulanması için komşu sözcükler arasındaki boşluklar farklı olmalıdır (Shirali-Shahreza, 2008). Değişken boşluklardan dolayı bir dekoder, bu yöntemle işlenmiş bir dokümanı çözebilmek için dokümanın orijinaline veya orijinal belgedeki sözcük aralığıyla ilgili bir spesifikasyona ihtiyaç duyar. Kodlayıcı önce kodlama işleminin yapılabilmesi için bir satırın yeterli sayıda sözcüğe sahip olup olmadığını belirler, kısa satırlar kodlanmaz. Bu yöntemde kodlanabilir bulunan her satıra, diferansiyel kodlama tekniği uygulanır. İkinci, dördüncü, altıncı vb. sözcükler sol kenar boşluğundan kaydırılır. Sütun hizalamasını korumak için her satırdaki ilk ve son sözcükler kaydırılmaz. Sözcük kaydırma işlemi bittikten sonra belge dağıtılır. Dekoder belgeyi çözmek için orijinal doküman hakkında bilgiye ihtiyaç duyar. İhtiyaç duyulan bilgi, her sözcüğün başlangıç konumu veya merkezinin konumudur (Popa, 1998).

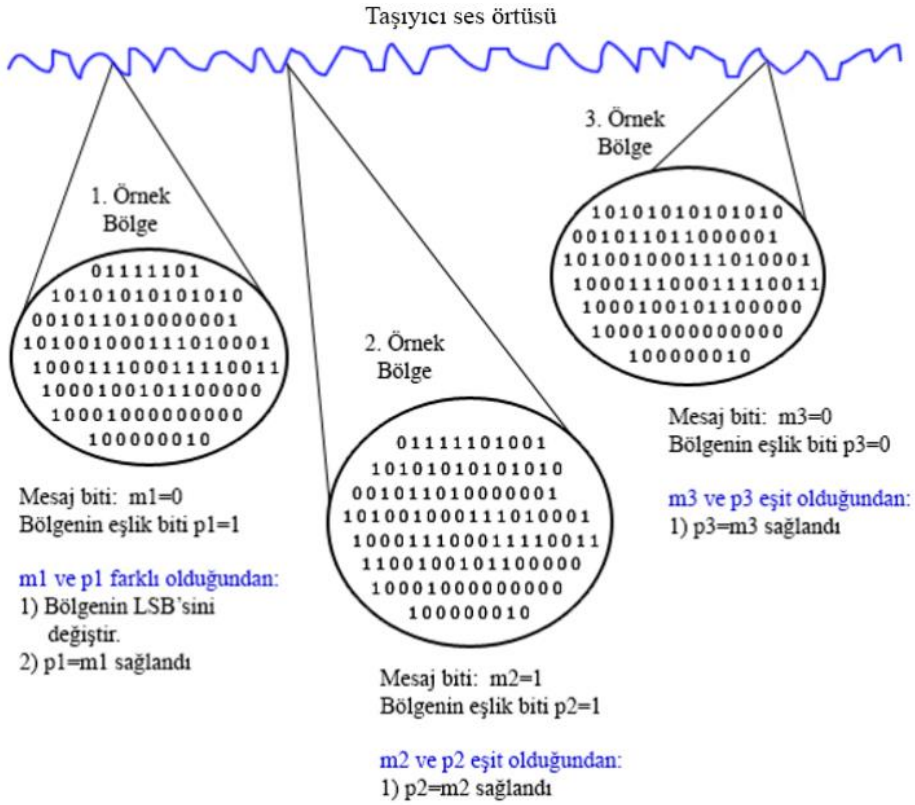
Özellik kodlama yöntemi ise belirli metin özelliklerini seçer ve bu özellikleri değiştirir (Huang ve Yan, 2001). Örnek olarak bu özellik b, d, h, k gibi harflerin dikey çizgileri olabilir. Bunların uzunluğu sıradan okuyucular için hatları algılanamayacak şekilde değiştirilebilir. Belirli bir yazı tipindeki karakter yükseklikleri ayrıca değiştirilebilir (Popa, 1998). Bunların yanı sıra sözcükleri kendi eş anlamlılarıyla değiştiren teknikler de vardır. Genelde iki çift eşanlamlı vardır ve bu eş anlamlılardan birini veya diğerini kullanmak “0” veya “1” gömmekle eşdeğerdir (Sahni, 2019). İletişime katılan taraflar bu eşanlamlı çiftleri birbirleriyle paylaşmalıdır.

3.2.2. Ses (audio) steganografi

Ses steganografisinde gizlenecek mesaj, bir ses dosyasının binary formunda basit bir değişiklik yapılarak o ses dosyasının içerisine yerleştirilir. Bu ses dosyası Film Uzmanlar Grubu Ses Katmanı 3 (Motion Pictures Experts Group Audio Layer 3, MP3), Dalga Formu Ses Dosyası Formatı (Waveform Audio File Format, WAV), Gelişmiş Ses Kodlama (Advanced Audio Coding, AAC) gibi çeşitli formatlarda olabilir. Ses steganografisiyle mesaj gizlemek, dijital resim gibi diğer medya ortamlarını kullanarak mesaj gizlemekten daha zordur. Bu sistemle bilgileri gizlemek amacıyla kullanılan birkaç teknik ve yöntem vardır. Bu yöntemler, basitten zora doğru geniş bir yelpazede yer alır. Yani mesajları ses dosyasındaki bir gürültü modeline basitçe yerleştiren yöntemlerden, bilgileri gizlemek için gelişmiş sinyal işleme üzerinde çalışan daha karmaşık tekniklere kadar birçok uygulama vardır. Bu yöntemlerden yaygın olarak kullanılanları LSB kodlama, parity (eşlik) kodlama, faz kodlama, spread spectrum (yayılmış spektrum) ve echo hiding (yankı gizleme) olarak ifade edebiliriz (Qiao ve ark., 2012).

Çok popüler bir metodoloji olan LSB, gizlenecek bilginin bayt dizisini örtü dosyasının bazı baytlarının en anlamsız bitleriyle yer değiştirir. Bu genellikle 24 bit bitmap resimlerde olduğu gibi LSB metodunun önemli bir kalite bozulmasına neden olmadığı durumlarda etkilidir. Bu işlem uygulanmadan önce hem örtü nesnesi olarak kullanılacak ses dosyası hem de bu örtü nesnesinin içerisine saklanacak gizli bilgi binary forma çevrilir. LSB metodu özellikle görüntü steganografisinde çok yaygın olarak kullanılmaktadır (Bandyopadhyay ve ark., 2008). Bu yüzden metotla ilgili detaylarda görüntü steganografisi bölümünde bahsedilecektir.

Başka bir ses steganografi yöntemi olan eşlik (parity) kodlaması, güçlü ses steganografi tekniklerinden biridir. Bu yöntem sinyali ayrı örneklere böler ve gizli mesajın her bitini bir eşlik bitine gömer. Seçilen bölgenin eşlik biti saklanacak gizli bitle eşleşmezse bölgedeki örneklerden birinin LSB'si tersine çevrilir. Bu sayede göndericinin gizli biti kodlamada daha fazla seçeneği olur (Dutta ve ark., 2009). Şekil 3.32' de eşlik kodlama tekniği görsel olarak ifade edilmiştir.



Şekil 3.32. Eşlik (parity) kodlama tekniği

Bir diğer ses steganografi yöntemi faz kodlama tekniğidir. Bu teknik, bir başlangıç ses segmentinin fazını gizli bilgiyi temsil eden referans bir fazla değiştirir. Geri kalan segmentlerin fazı, segmentler arasındaki faz farkını korumak için ayarlanır. Sinyal-gürültü oranı açısından, faz kodlama en etkili kodlama yöntemlerinden biridir. Fazda ciddi bir değişiklik olduğunda her frekans bileşeni arasındaki ilişki, fark edilir bir faz dağılımı meydana getirecektir. Ancak, fazdaki değişim yeterince küçük olduğu sürece, duyulamayan bir kodlama gerçekleştirilebilir (Cvejic ve Seppanen, 2002). Bu yöntem, sesin faz bileşenlerinin gürültü gibi insan kulağı tarafından algılanabilir olmadığı gerçeğine dayanır (Bandyopadhyay ve ark., 2008).

Faz kodlama tekniğiyle gizli bilginin saklanması için uygulanan işlem adımları aşağıda ifade edilmiştir (Dutta ve ark., 2009):

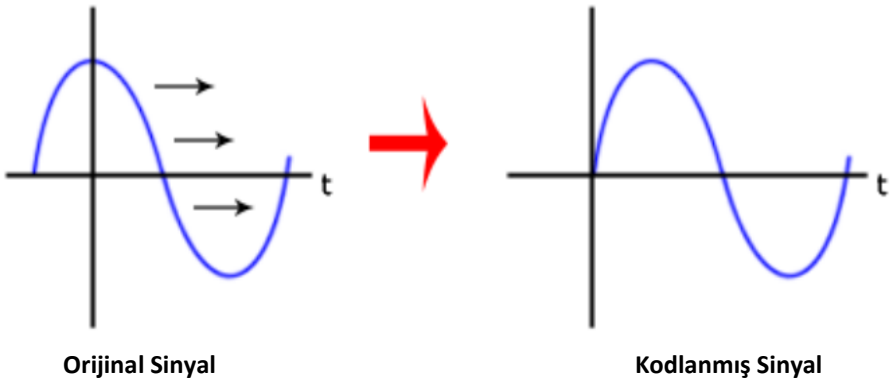
- Orijinal bir ses sinyali, uzunlukları kodlanacak mesajla aynı boyutta olacak şekilde daha küçük segmentlere ayrılır.
- Faz matrisi, Ayrık Fourier Dönüşümü (Discrete Fourier Transform, DFT) uygulanarak oluşturulur.
- Komşu segmentler arasındaki faz farkı hesaplanır.

- Komşu segmentler arasındaki faz kaymaları kolayca tespit edilebilir. Bu segmentlerin mutlak fazlarının değiştirebileceği ama komşu segmentler arasındaki faz farkının korunması gerektiği anlamına gelir. Bu yüzden gizli bilgi sadece ilk sinyal segmentinin faz vektörüne eklenebilir. Bu ekleme işlemi Şekil 3.33' de gösterilen fonksiyon kullanılarak gerçekleştirilebilir.

$$yeni\ faz = \begin{cases} \pi/2, & mesaj\ biti = 0\ ise \\ -\pi/2, & mesaj\ biti = 1\ ise \end{cases}$$

Şekil 3.33. Faz kodlama tekniğinde kullanılan bir fonksiyon

İlk segmente eklenen yeni faz ve orijinal faz farkları kullanılarak yeni bir faz matrisi oluşturulur. Ses sinyali, yeni faz matrisi ve orijinal büyüklük matrisi kullanılarak DFT'nin tersinin uygulanmasıyla yeniden yapılandırılır ve sonra ses segmentleri tekrar bir araya getirilerek birleştirilir. Alıcı, ses dosyasından gizli bilgiyi çıkarmak için segment uzunluğunu bilmek zorundadır. Sonra alıcı DFT'yi kullanarak fazları elde eder ve gizli bilgiyi çıkartır (Jayaram ve ark., 2011). Şekil 3.34' de faz kodlama yöntemi görsel olarak ifade edilmiştir.



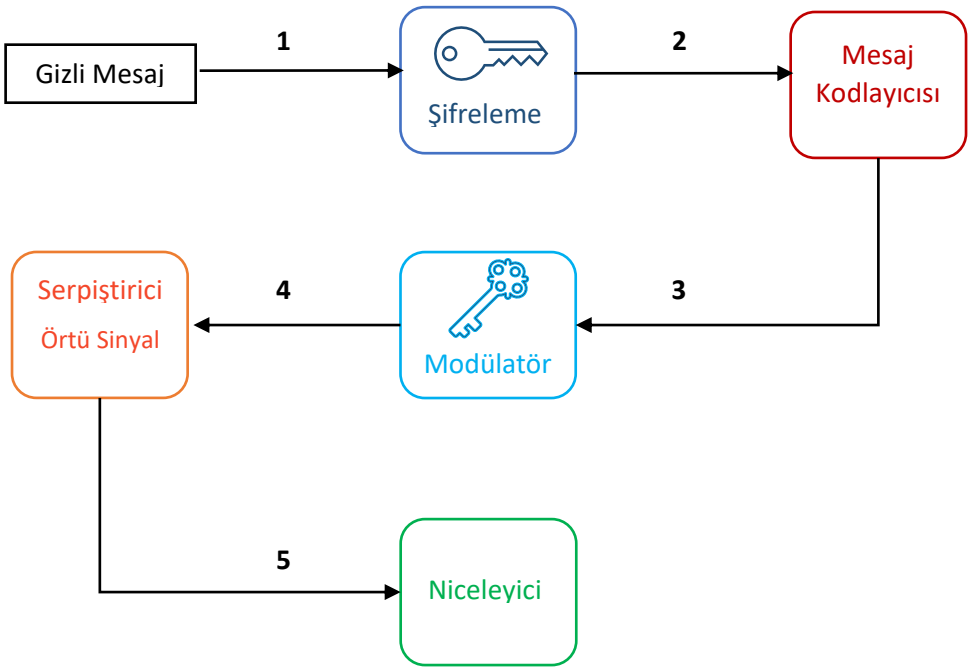
Şekil 3.34. Faz kodlama tekniğinin sinyal üzerinde gösterimi

Ses steganografisinde kullanılan metotlardan birisi de yayılmış spektrum (spread spectrum)' dur. Yayılmış spektrum yöntemi, gizli bilgileri ses sinyalinin frekans spektrumu boyunca yaymaya çalışır (Qiao ve ark., 2009). Bu, mesaj bitlerini tüm ses dosyasına rastgele yayan bir LSB metoduna benzer. Ancak LSB kodlamasının aksine

yayılmış spektrum yöntemi gizli bilgileri, gerçek sinyalden bağımsız bir kod kullanarak ses dosyasının frekans spektrumu üzerine dağıtır. Sonuç olarak, son sinyal gerçek sinyale göre iletim için daha fazla bant genişliğine ihtiyaç duyar. Yayılmış spektrum yöntemi, LSB kodlama, faz kodlama ve eşlik kodlama teknikleri ile karşılaştırıldığında veri aktarım hızı ve dayanıklılık açısından daha iyi performans göstermektedir. Ancak yayılmış spektrum yönteminin dezavantajı ise ses dosyasında gürültü meydana getirebilmesidir (Kaur ve Behal, 2014).

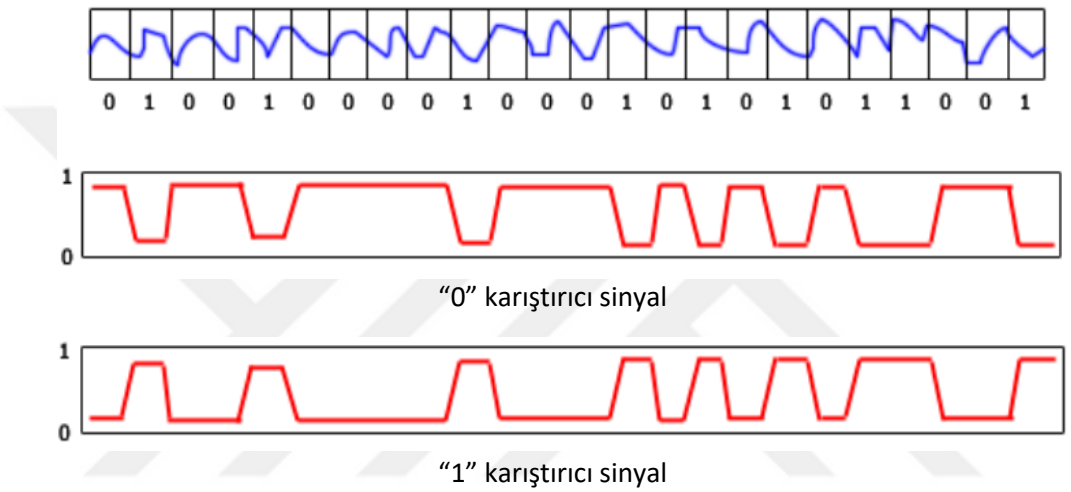
Şekil 3.35' de gösterilen yayılmış spektrum yönteminin işlem adımları aşağıda ifade edilmiştir (Al-Othmani ve ark., 2012):

1. Gizli mesaj bir simetrik anahtar kullanılarak şifrelenir.
2. Şifrelenen mesaj, düşük oranlı bir hata düzeltme kodu kullanılarak kodlanır ve bu işlem sistemin dayanıklılığını artırır.
3. Kodlanmış mesaj daha sonra ikinci bir simetrik anahtar oluşturulan sözde rastgele bir sinyal ile modüle edilir.
4. Ortaya çıkan ve mesaj içeren rastgele sinyal örtü sinyal üzerine dağıtılır.
5. Son sinyal, mesajı içeren yeni bir dijital ses dosyası oluşturmak için kuantize edilir.
6. Bu süreç mesajın geri elde edilmesi için tersine çevrilir.



Şekil 3.35. Yayılmış spektrum yönteminin işlem adımları

Ses steganografisi kapsamında bahsedeceğimiz son yöntem ise yankı gizleme (echo hiding) tekniğidir. Bu teknik ayrık sinyale bir yankı ekleyerek gizli bilgileri ses dosyasına gömer. Yankı gizleme tekniği diğer yöntemlerle karşılaştırıldığında yüksek veri iletim hızı ve üstün dayanıklılık avantajlarına sahiptir. Orijinal sinyalden yalnızca bir yankı üretildiğinde gizli bilginin sadece tek bir biti kodlanabilir. Bu nedenle, kodlama işlemine başlamadan önce orijinal sinyal bloklara bölünür. Kodlama işlemi tamamlandıktan sonra sinyalin son halini oluşturmak için bloklar tekrar bir araya getirilir (Dutta ve ark., 2009). Yankı gizleme metodu Şekil 3.36'da gösterilmiştir.



Şekil 3.36. Yankı gizleme (echo hiding) yöntemi

Bahsedilen yöntemlerin yanı sıra gizli bilgi bir yer değiştirme şemasıyla müzikal tonlar kullanılarak kodlanabilir. Örneğin bir ses tonu "0" bitini temsil ederken başka bir ses tonu "1" bitini temsil edebilir. Böylece gizli mesajı temsil edebilecek normal bir müzik parçası bestelenebilir veya mevcut bir parça, gizli mesajı temsil eden bir kodlama şemasıyla düzenlenebilir (Bandyopadhyay ve ark., 2008).

3.2.3. Görüntü (image) steganografi

Bir görüntü içerisine belirli mesajları gömmek için dikkat çekmeyen gürültülü alanlar oluşturulabilir veya gizli bilgiler görüntülerin doğal renk çeşitliliğinin fazla olduğu bölgelerine saklanabilir. Bu işlemi uygulamak gizli mesajları görüntünün çeşitli bölgelerine rastgele dağıtmakla aynı derecede mümkündür. Görüntü steganografisi için kullanılan birçok yöntem mevcuttur. Bunlar kendi içerisinde uzaysal / görüntü alan

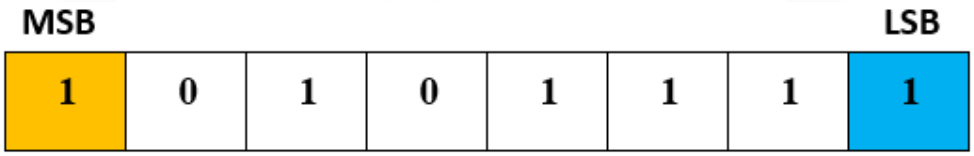
teknikleri (spatial / image domain techniques) ve frekans / dönüşüm alan teknikleri (frequency / transform domain techniques) olmak üzere iki ana kategoriye ayrılır (Morkel ve ark., 2005).

Uzaysal alan teknikleri, gizli bilgiyi doğrudan örtü görüntüsünün piksellerine gömer. Bu kategoride yer alan en popüler veri gizleme metodu LSB olarak da bilinen piksellerin en sağdaki bitlerini değiştirme yöntemidir. Bitmap, Taşınabilir Ağ Grafiği (Portable Network Graphics, PNG) ve 8 bit gri tonlamalı Grafik Değişim Formatı (Graphics Interchange Format, GIF) gibi kayıpsız görüntü biçimleri, LSB yöntemleri için kullanılabilir. LSB, gizli bir bilgiyi belirli bir görüntüde saklamanın en kolay yollarından biridir ve LSB yerleştirme tekniği kullanılarak oluşturulan bir stego-imgeyi deşifre etmek zor değildir (Cheddad ve ark., 2010). Bir diğer uzaysal alan tekniği ise Bit Düzlem Karmaşıklık Segmentasyonu (Bit Plane Complexity Segmentation, BPCS) ‘dur. İlk kez 1997 yılında Richard O. Eason ve Eiji Kawaguchi tarafından tanıtılan bu teknik, veri eklemek için her bit düzleminin hesaplama karmaşıklığını kullanır. BPCS, taşıyıcı ortamın tanımlanan bir eşik değerine göre ölçülen segmentlere bölünmesiyle uygulanır. Bilgiler gürültülü bölgeler gibi yüksek karmaşıklığa sahip bir segmente yerleştirilir. BPCS tekniği, en anlamsız tek bir biti değiştiren LSB yönteminin aksine sadece en düşük değerlikli bit üzerinde değil tüm bit düzlemi üzerinde işlem yapar (Yeshwanth Srinivasan, 2003).

Bilgi, uzaysal alan yöntemleri kullanılarak bir resmin içerisine gizlenirse bu resme sıkıştırma gibi görüntü işleme tekniklerinden herhangi biri uygulandığında saklanan bilgi kayba maruz kalır. Bu sorunu aşmak için bilgileri görüntünün frekans alanında güvence altına alan frekans / dönüşüm alan teknikleri kullanılır. Böylece gizli bilgiler frekans değerlerine eklenirken yüksek frekans kısmı da silinir. Bunun için ilk olarak gizlenmesi amaçlanan bilgiler ve görüntü bir dönüşüm işlemine tabi tutulur ve sonra dönüşüm katsayı değerleri değiştirilir. Üç farklı temel dönüşüm tekniği vardır. Bunlar; Hızlı Fourier Dönüşümü (Fast Fourier Transform, FFT), Ayrık Kosinüs Dönüşümü (Discrete Cosine Transform, DCT) ve Ayrık Dalgacık Dönüşümü (Discrete Wavelet Transform, DWT) yöntemleridir (Cheddad ve ark., 2010). FFT tekniği iki boyutlu olarak uygulanır. İlk olarak örtü resmine FFT dönüşümü uygulanır, sonra da gizli bitler önemli dönüşüm katsayılara yerleştirilir. Ayrıca bu teknik karmaşık matematiksel formüller içerir. Bu yüzden DCT steganografisine göre daha fazla matematiksel işlem içerir ve daha fazla işlem zamanı gerektirir. DCT tekniği ise örtü resmin iki boyutunu dönüştürmek için uygulanır ve sadece gerçek değerlerle çalıştığından FFT yöntemine göre daha az

hesaplama işlemi gerektirir. Başka bir yöntem olan DWT ise yüksek frekans bileşenlerinin düşük frekans bileşenlerinden ayrılması üzerine çalışır ve sonra yüksek frekanslı kısmı gizli verilerle değiştirir. Bu tekniğin gömme kapasitesi DCT steganografisinden çok daha yüksektir (Yeshwanth Srinivasan, 2003).

Bu tez çalışması kapsamında uzaysal alan tekniklerinden en popüleri olan LSB metodu kullanılacaktır. Bundan dolayı bölümün devamında LSB tekniği ayrıntılı olarak ifade edilecektir. Dijital sistemlerde herhangi bir bilgi bit dizisi formunda ifade edilebilir. Örnek olarak desimal formdaki 175 sayısının binary notasyonu $1 * 2^7 + 0 * 2^6 + 1 * 2^5 + 0 * 2^4 + 1 * 2^3 + 1 * 2^2 + 1 * 2^1 + 1 * 2^0 = 10101111$ olarak ifade edilir. Bu dizilimin en anlamlı ve en anlamsız dijitleri Şekil 3.37' de gösterilmiştir.

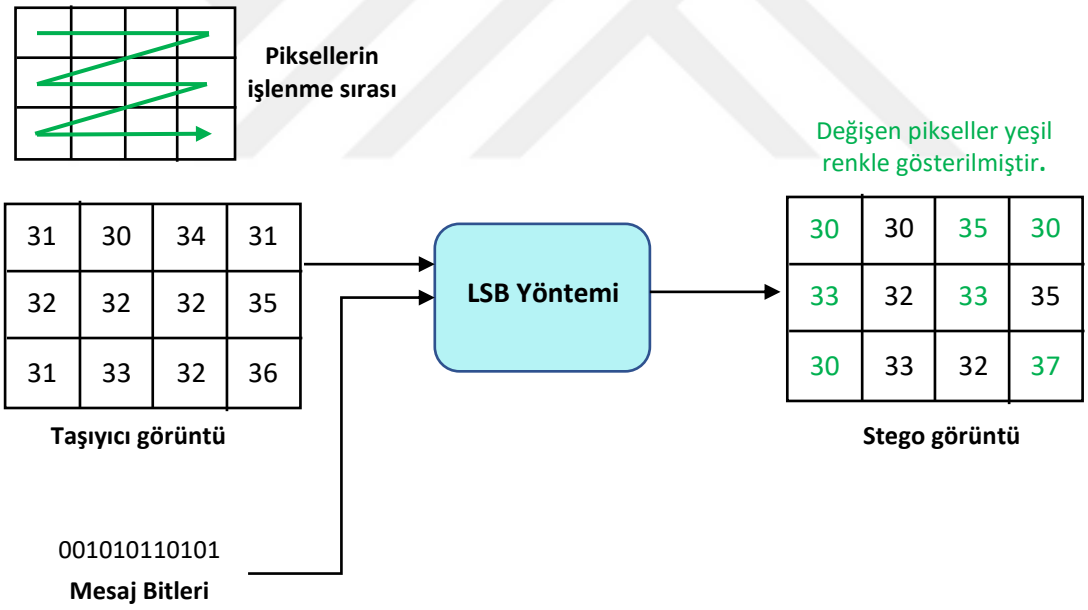


Şekil 3.37. 175 sayısının MSB ve LSB bitlerinin gösterimi

Bir sayının binary notasyonunda 2 tabanında en büyük üstel değerle temsil edilen ya da başka bir deyişle en soldaki dijit o sayı için En Önemli Bit (Most Significant Bit, MSB) olarak da adlandırılan en anlamlı değerdir. Buna karşın sayının 2 tabanında en küçük üstel değerle ifade edilen yani en sağdaki dijiti ise o sayı için LSB olarak da adlandırılan en anlamsız değerdir. Bu tanımlama, binary notasyonundaki bir sayının MSB dijitinin değiştirilmesinin o sayı üzerinde yapılabilecek en büyük değişim olduğunu ve LSB dijitinin değiştirilmesinin de o sayı üzerinde yapılabilecek en küçük değişim olduğunu ifade eder (Chan ve Cheng, 2004). Yine 175 sayını örnek olarak incelersek MSB dijitinin değiştirilmesi durumunda sayıda $2^7 = 128$ eksilme olacağından sayının yeni değeri 47 olurken LSB dijitinin değiştirilmesi durumunda ise sayıda sadece $2^0 = 1$ eksilme olacağından sayının yeni değeri 174 olur. Örnekten de anlaşıldığı üzere iki durumda da sadece tek bir dijitin değiştirilmesine rağmen MSB dijitini değiştirmek sayıda değer olarak çok büyük bir farklılığa neden olurken LSB dijitinin değiştirilmesi ise sayıda değer olarak çok küçük bir değişime neden olmaktadır (Gupta ve ark., 2012). LSB steganografi tekniği de işte bu temele dayanmaktadır. Bu teknikte taşıyıcı resmin piksellerinde LSB bitlerinin değiştirilmesiyle gizli bilgiler taşıyıcı resme yerleştirilerek saklanır. Piksel değerlerinin LSB bitlerinde değişiklik yapılması bu resimlerin görüntü kalitesinde insanlar tarafından algılanabilecek önemli bozulmalara sebep olmaz ve bu

bozulmalar gürültü olarak değerlendirilebilir. Bu nedenle, çoğu durumda stego-görüntü çıplak gözle orijinal (örtü resim) olanından ayırt edilemez ve tespit edilebilir herhangi bir kalite kaybı içeride gizli bir mesajın mevcudiyetine atfedilirse bile bu mesajın ortaya çıkarılması hala zordur.

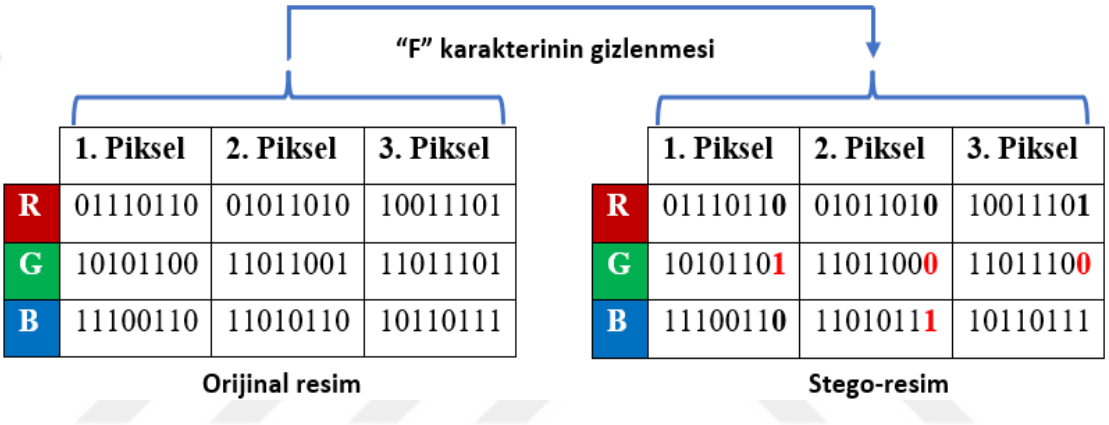
LSB yönteminde verilerin eklenme modu kullanılan taşıyıcı resmin 8 bit veya 24 bit olmasına bağlı olarak değişir. Gri tonlamalı 8 bit resimlerde her piksele yalnızca tek bir bit gizlenir. Mesaj bitleri, her pikselin en sağdaki bit değerleri üzerine art arda yazılır. Bu işlemin nasıl uygulandığı Şekil 3.38'de 12 piksellik bir blok görüntü örneği üzerinde gösterilmiştir. Örnekte görüldüğü gibi gömülecek bit ile bu bitin gömüleceği pikselin LSB' si aynı ise hiçbir değişiklik yapılmaz fakat bu değerler birbirinden farklı ise ilgili pikselin en anlamsız biti değiştirilir. Dolayısıyla aslında bazı piksellerde hiçbir değişiklik yapmadan o pikseller de bilgiyi saklamak için kullanılmış olur. Bu durumda gizlenecek bilginin bitleriyle kendi en önemsiz bitleri eşleşen piksel sayısı ne kadar fazla olursa taşıyıcı resimdeki kalite bozulması da o kadar az olur.



Şekil 3.38. LSB metoduyla veri gizleme işlemi

24 bit derinlikli renkli resimlerde her piksel kırmızı (red, R), yeşil (green, G) ve mavi (blue B) olmak üzere her biri bellekte 8 bitlik yer kaplayan üç renk bileşeninden meydana gelir. Bu sayede 24 bit derinlikli görüntülere bilgi gizlemek için bu üç renk bileşeni birlikte kullanılabilir. Dolayısıyla her bir renk bileşenine bir bitlik veri yerleştirilerek görüntünün her pikseline toplam 3 bitlik gizli bilgi saklanabilir. Örneğin

“F” karakterinin 24 bit derinlikli bir resim içerisine gizlenmesini ifade edecek olursak yapılacak ilk işlem bu karakterin ASCII tablosundaki karşılığının kullanılarak binary forma çevrilmesidir. ASCII, 7 bitlik bir karakter kodlama standardıdır ve $2^7 = 128$ tane karakter kodlayabilir. Daha sonrasında 8 bitlik kodlama yapan genişletilmiş ASCII standardı tanımlanmıştır (Peruginelli ve ark., 1992). Bu standarda göre her karakterin 0-255 aralığında desimal bir karşılığı vardır ve “F” karakterinin desimal karşılığı da 70’tir. Buna göre “F” karakteri binary formda 01000110 şeklinde temsil edilir. Şekil 3.39’ da “F” karakterinin 24 bit derinlikli renkli bir resme gizlenmesi görsel olarak ifade edilmiştir.

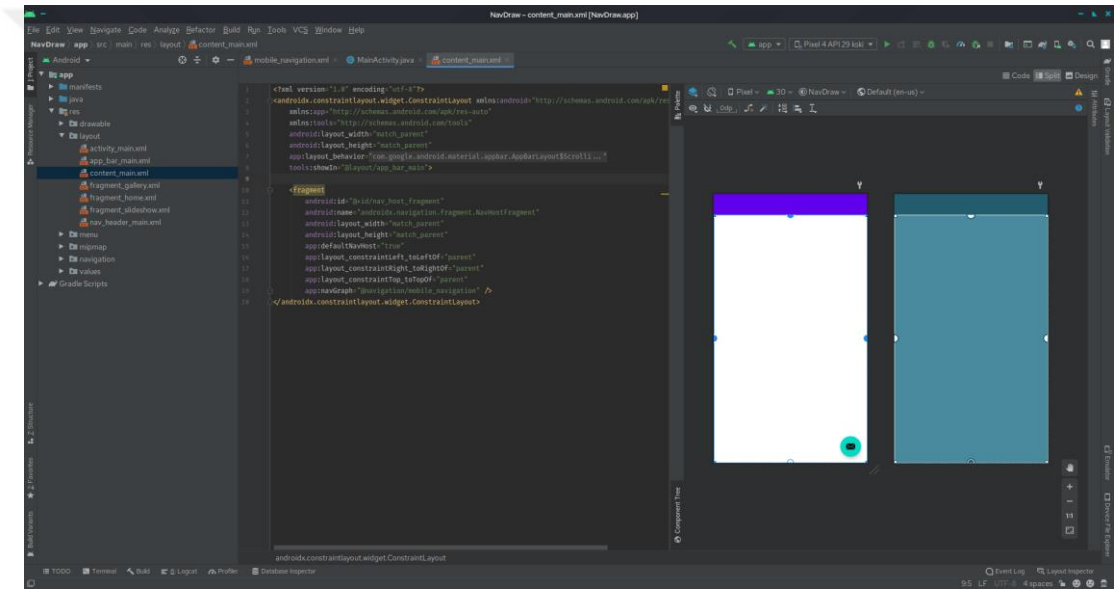


Şekil 3.39. 24 bit derinlikli bir görüntüde LSB yönteminin uygulanması

Şekil 3.39’ de yer alan tablolar örnek olarak 24 bit derinlikli renkli bir resme ait 3 pikseli temsil etmektedir. Şeklin sol tarafındaki tablo resmin orijinal halini, sağ tarafında stego-resim olarak ifade edilen tablo ise “F” karakterinin gizlendikten sonraki halini göstermektedir. Her pikselin, RGB değerleri binary formda ifade edilmiştir. “F” karakterinin bitleri, orijinal resmin 1.pikselinden başlamak üzere her pikselin RGB değerlerinin en anlamsız bitlerine sırasıyla yerleştirilmiştir. Bu yerleştirme işleminde sadece 4 bit değiştirilmiş diğerleri olduğu gibi kalmıştır. Bu örnekte olduğu gibi örtü veya taşıyıcı resmin tamamını kullanarak bir bilgi gizlenmek istendiğinde de ortalama olarak tüm en anlamsız bitlerin yarısı kadarında değişiklik yapmak yeterli olacaktır. En anlamsız bitler üzerinde yapılan bu değişiklikler de insanlar tarafından algılanamayacak kadar önemsiz olduğu için mesaj başarılı bir şekilde gizlenmiş olacaktır. Ayrıca görüldüğü gibi bilgiyi gizleme işleminde 3.pikselin mavi katmanı kullanılmamıştır. Dolayısıyla bu kısım, 3 piksele gizlenen 8 bitin doğruluğunu kontrol etmek için kullanılabilir ya da başka bir deyişle bu kısmın LSB’si eşlik biti olarak değerlendirilebilir.

3.3. Android Studio

Android Studio, Android işletim sistemli cihazlar için Google tarafından sunulan resmi tümleşik geliştirme ortamıdır. 16 Mayıs 2013 tarihinde Google I/O etkinliğinde tanıtılmıştır. Android Studio, JetBrains şirketi tarafından sunulan ve Java'da bilgisayar yazılımı geliştirmek için kullanılan IntelliJ IDEA temellidir. Android geliştirme için gerekli araç-gereçlerin bulunduğu IDE, resmi olarak Java ve Kotlin dillerini desteklemektedir. Android Studio, Apache lisansı ile lisanslanmıştır ve ücretsiz olarak edinilebilmektedir (Anonim, 2021). Şekil 3.40' da Android Studio' nun arayüzü görülmektedir.



Şekil 3.40. Android Studio geliştirme ortamı

Android Studio, birçok farklı Android cihaza uygulama geliştirmeyi hızlandıran araçlar bulundurmaktadır. Gradle sayesinde proje geliştirme aşamasında esneklik sağlayabilmektedir. Ekran çözünürlüğü gibi farklılıkları bulunan cihazlar için uygun Android Paket Kiti (Android Package Kit, APK) çıktısı verebilme özelliğine sahiptir. Kodu tamamlama, kod analizi yapma, kodun yeniden düzenlenebilmesi gibi geliştirme aşamasında gerçekleştirilen işlemleri oldukça hızlı yapabilmektedir. Hızlı kod yazımı için kod şablonları bulunmaktadır. Ayrıca versiyon kontrol sistemleri ile birlikte çalışabilmek için entegre edilmiş özellikler bulunmaktadır (Öztürk, 2021).

3.4. Android Software Development Kit (SDK)

SDK, yazılım ve donanım geliştiricilerine yönelik olarak hazırlanmış ve içerisinde bazı tasarım araçlarını barındıran bir paket programdır. SDK'ların içerisinde Uygulama Programlama Arayüzü (Application Programming Interface, API), rehber içerikler ve örnek çalışmalar bulunmaktadır (Anonim, 2020). Android SDK'nın içerisinde de Android API'lar, emülatör, sanal cihaz yöneticisi, dokümantasyon, geliştirme araçları, uygulama geliştirmede kılavuzluk yapacak örnek kodlar yer alır. Bu bileşenlerden en önemlisi Android API' lardır.

Android platformu geliştikçe ve yeni Android sürümleri yayınlandıkça, her Android sürümüne API düzeyi olarak adlandırılan benzersiz bir tamsayı tanımlayıcı atanır. Bu nedenle, her Android sürümü tek bir Android API düzeyine karşılık gelir. Geliştiricilerin uygulamalarını daha eski ve en son Android sürümlerinin yanı sıra, birden çok Android API düzeyiyle çalışacak şekilde tasarlaması gerekir. Android API'lar uygulama ve işletim sistemi arasındaki iletişimi ve etkileşimi sağlar. Her Android sürümü için farklı seviyelerde API paketleri bulunmaktadır. Her Android cihazı, sadece tek bir API düzeyinde çalışır. API düzeyi, uygulamaların çalışabileceği cihazları tanımlar.

Android ' de bir hedef veya en düşük API düzeyi seçilmeden önce, bu API düzeyine karşılık gelen Android SDK platform sürümü yüklenmelidir. Hedef Framework, minimum Android sürümü ve hedef Android sürümü için kullanılabilen seçeneklerin aralığı, yüklenilen Android SDK sürümlerinin aralığıyla sınırlıdır. Gerekli Android SDK sürümlerinin yüklendiğini doğrulamak için SDK yöneticisi kullanılabilir ve uygulamanın için ihtiyaç duyduğu yeni API düzeylerini yine SDK yöneticisi üzerinden yüklenebilir.

3.5. Open Source Computer Vision Library (OpenCV)

OpenCV, gerçek-zamanlı bilgisayar görüsü uygulamalarında kullanılan açık kaynaklı bir kütüphanedir. Intel tarafından geliştirilmiştir. Bu kütüphane çoklu platform ve Berkeley Yazılım Dağıtımı (Berkeley Software Distribution, BSD) lisansı altında açık kaynaklı bir yazılımdır. Yüz tanıma sistemi, hareket tanıma, insan-bilgisayar etkileşimi, nesne tanıma, resim segmentleme, hareket takibi, artırılmış gerçeklik gibi uygulamalarda kullanılmaktadır (Anonim, 2021). OpenCV, yapısındaki 500 üzerindeki fonksiyon yardımıyla görüntüler üzerinde birçok işlemi hızlı bir şekilde yapabilmektedir. OpenCV kütüphanesi; Windows, Linux, Mac OS X işletim sistemleri üzerinde çalışabilmektedir.

Ayrıca Android ve IOS mobil işletim sistemlerinde de kullanılabilmektedir. Bunlarla birlikte, Hesaplama Bileşik Cihaz Mimarisi (Compute Unified Device Architecture, CUDA) ara yüzü ile Grafik İşlemci Birimi (Graphics Processing Unit, GPU) üzerinde çalışabilme özelliği vardır. Özellikle C++ programla dili üzerine çok uyumludur. Ek olarak C, Python ve Java programlama dilleri üzerinde de çalışabilmektedir.

Bu çalışma kapsamında Steganografi algoritmasının, Android uygulama üzerinde gerçeklenebilmesi için bu kütüphaneden faydalanılacaktır. Android uygulamalarında OpenCV' yi kullanabilmek için OpenCV Android SDK' yı indirip projeye dahil etmek gerekir. OpenCV Android SDK, yerel kütüphaneler ve Android platformuna özel sınıflar içeren bir interface gibi düşünülebilir.

3.6. Genymotion Emülatör

Emülatör, herhangi bir işletim sistemi üzerinde başka bir işletim sisteminin çalıştırılmasını sağlayan yazılımlara denir (Anonim, 2021). Bu çalışma kapsamında geliştirilen uygulamanın farklı özellikteki cihazlarda sanal olarak test edilmesi için Genymotion emülatör kullanılacaktır. Genymotion, 2011 yılında Arnaud Dupuis tarafından kurulan ve şu anda Paris, Lyon ve San Francisco'da birimleri bulunan Genymobile tarafından geliştirilmiştir. Genymotion hem bulutta hem de masaüstünde birden fazla kanalda kullanılabilen tam teşekküllü bir Android platformu olarak çalışmaktadır.

Genymotion, sanal makine üzerinde çalışan bir emülatördür. Bilgisayarda oluşturulan sanal bir makine üzerine kurulu Android işletim sistemini kullanır. Android Studio' nun sunduğu dahili emülatör Android Sanal Cihazları (Android Virtual Devices, AVD) oldukça yavaş çalışmaktadır. Genymotion ile daha hızlı ve performanslı bir şekilde emülatör kullanılabilir ve uygulamalar daha verimli bir şekilde test edilebilir.

3.7. Firebase

Firebase, Google tarafından mobil ve web uygulamaları oluşturmak için geliştirilmiş bir platformdur (Anonim, 2021). Firebase ile uygulama yönetimi, veri depolama, bildirim gönderme, kullanım takibi gibi işlemler ekstra bir sunucuya ve sunucu taraflı kod yazmaya gerek olmadan gerçekleştirilebilmektedir. Firebase, bilinen tablolar

ve SQL yerine verileri root-child şeklinde organize edilen JavaScript Nesne Gösterimi (JavaScript Object Notation, JSON) veri parçacıklarında saklamaktadır.

JSON, bütün platformlar arasında, verilerin standartlaştırılmış formatta değişimini sağlayan bir metin biçimidir. Veriler yay ve ayraç gibi noktalama işaretleri kullanılarak düzenlenir. JavaScript'te bulunan bazı yapılardan temel alınarak tasarlanmıştır. JSON sahip olduğu ağaç yapısıyla Belge Nesnesi Modeli, (Document Object Model, DOM) ve Genişletilebilir İşaretleme Dili (Extensible Markup Language, XML) ile benzerlik gösterse de bunlardan tamamen farklı bir yapıdadır (Anonim, 2021). JSON, platform bağımsız kullanılabildiği için Firebase' de veriler bu formda saklanmaktadır.

Firebase, aynı zamanda Realtime Database (Gerçek Zamanlı Veri tabanı), Cloud Messaging (Bulut Mesajlaşma) gibi özelliklerle de donatılmıştır. Firebase, platform fark etmeksizin aynı veriye her cihazdan erişilebildiği, uygulamaların kullanım verilerinin analiz edilebildiği, kullanıcıya bildirim gönderme, uygulamayı test etme gibi işlemlerin rahatlıkla yönetilebileceği ve bu ihtiyaçları karşılayabilmek için uygulama geliştiricilerine ücretsiz kullanım da sunan bir platformdur.

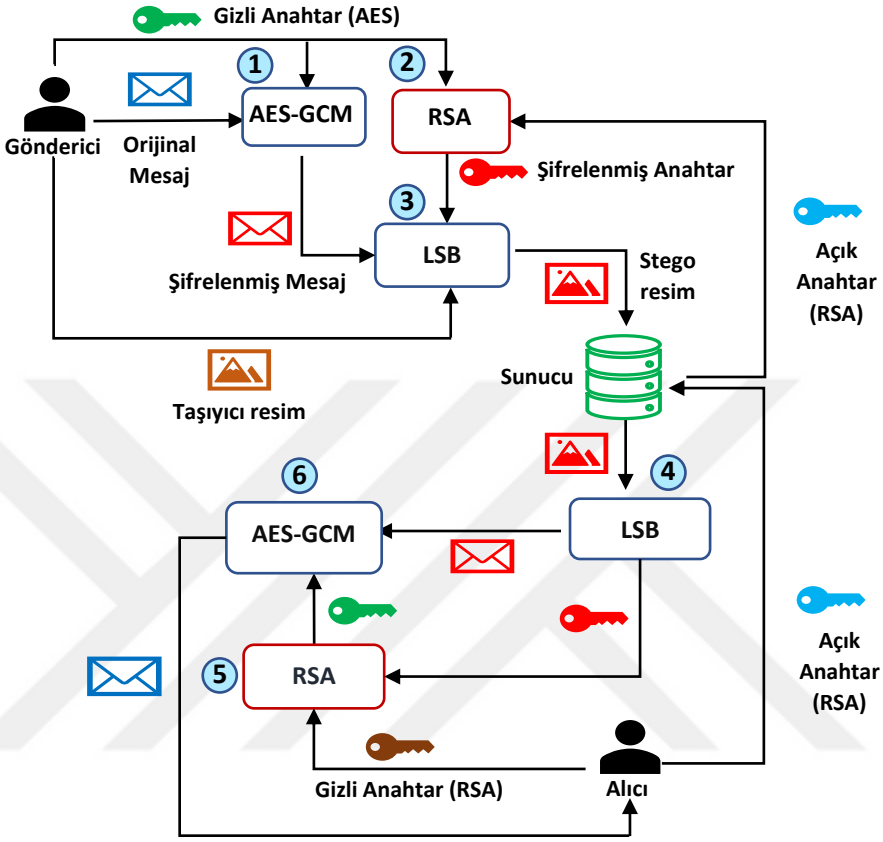
4. ÖNERİLEN YÖNTEM VE UYGULAMA

Bu tez çalışmasında iletişimde güvenliğin ve gizliliğin artırılması için kriptoloji ve steganografi tekniklerini entegre eden bir yaklaşım önerilmektedir. Bu amaçla kriptoloji algoritmalarının kullanımında güvenlik zafiyeti oluşturabilecek bazı sorunlara yönelik bir çözüm önerisi sunulmakta ve bu çözüm önerisinde steganografi biliminden de yararlanılmaktadır. Önerilen yaklaşım kullanılarak da mevcut uygulamalara alternatif oluşturabilecek anlık iletişim sağlayan bir mesajlaşma uygulaması geliştirilmiştir. Günümüzde iletişim için en çok mobil cihazlar kullanıldığından çalışmada mobil platformlar üzerinde gerçekleştirilmiştir. Bu bölümde ilk olarak önerilen yaklaşım detaylı bir şekilde açıklanmış ve sonrasında geliştirilen uygulamanın işleyişi ve çalışma mekanizmasından kullanıcı arayüzü görüntüleriyle birlikte bahsedilmiştir.

4.1. Önerilen Yöntem

Önerilen yöntem, Şekil 4.1’ de genel yapısı itibariyle gösterilmiştir. Bu yaklaşımda iletişim süreci, şifreleme ve gizleme olarak adlandırabileceğimiz iki temel işlemin ve bunların alt işlemlerinin yürütülmesiyle gerçekleşmektedir. Şifreleme süreci, mesajın veya gönderilecek verinin şifrelenmesi ve bunun için kullanılan gizli anahtarın şifrelenmesi işlemlerinden meydana gelir. Gizleme süreci ise şifreli verinin görüntüye gömülmesi ve görüntünün sıkıştırılması işlemlerinden oluşmaktadır. Şifreleme aşamasında literatürde yaygın olarak bilinen simetrik anahtarlı AES ve asimetrik anahtarlı RSA algoritması birlikte kullanılmaktadır. AES algoritması ile gönderilmek istenen mesaj veya veri şifrelenirken RSA algoritmasıyla da bu şifreleme işleminde kullanılan gizli anahtar şifrelenir. Böylece hem mesajın hızlı ve güvenli bir şekilde şifrelenmesi hem de anahtar paylaşımının gizli bir şekilde gerçekleşmesi sağlanmaktadır. Önerilen yaklaşımda AES algoritması GCM şifreleme moduyla birlikte uygulanmaktadır. AES-GCM hem yüksek hızda kimliği doğrulanmış şifreleme yapılmasını ve hem de veri bütünlüğünün korunmasını sağlar (Ahmad ve ark., 2018). Şifrelemeye ek olarak güvenliğin artırılması ve şifrelenen verilerin gizli bir şekilde iletilmesi için LSB steganografi tekniği uygulanmaktadır. Böylece gönderilmek istenen veriler fark edilemez hale getirilir. Hem saklama kapasitesinin yüksek olması hem de ses veya video ortamlarına göre daha düşük boyutlu olması sebebiyle steganografi için taşıyıcı ortam olarak fotoğraflar tercih edilmiştir. Öncelikle şifrelenen veriler daha sonrasında LSB

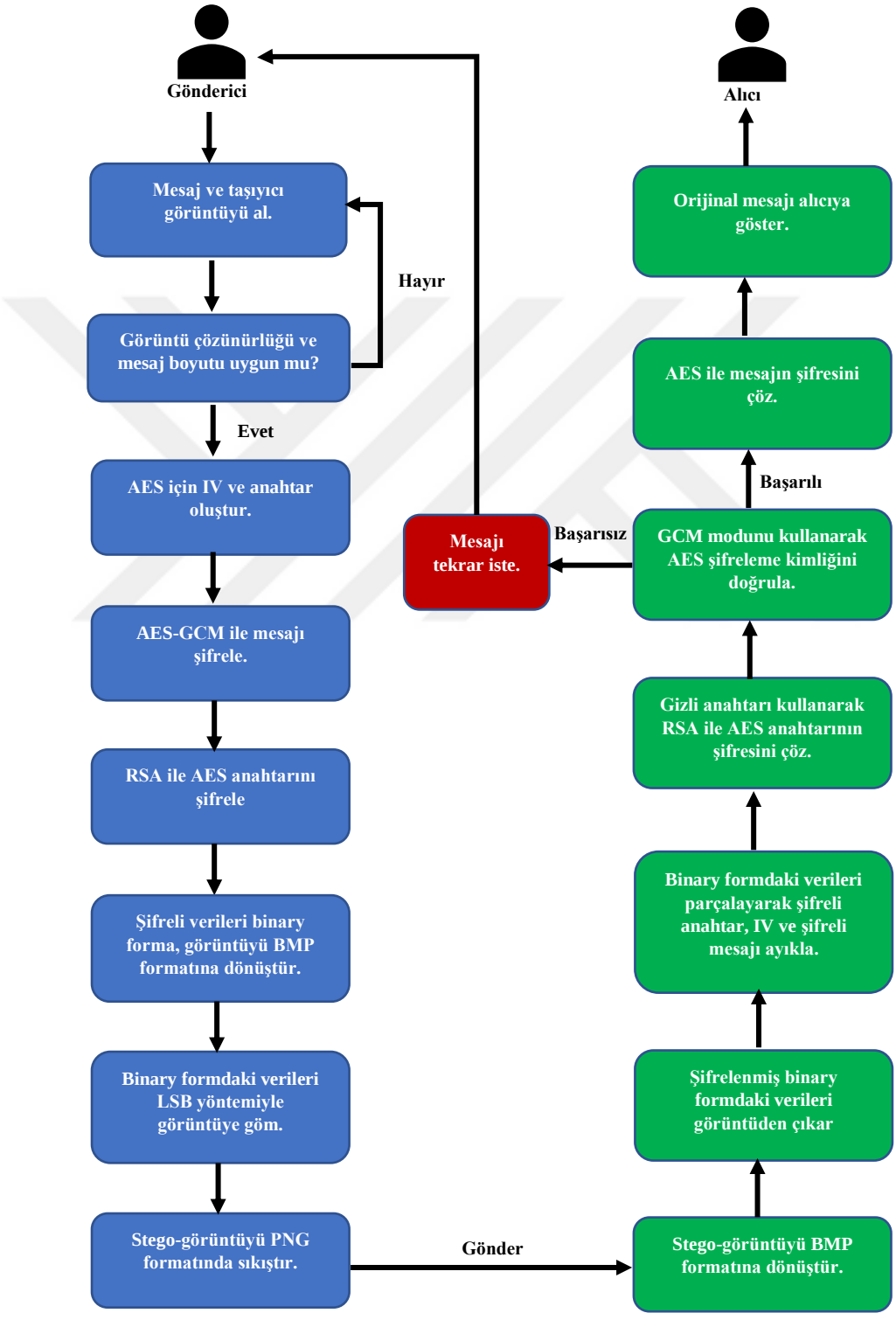
yöntemi kullanılarak herhangi bir fotoğrafa gömülür ve elde edilen bu stego-görüntü alıcıya iletilir. Alıcı kendisine ulaşan görüntüden önce şifreli verileri çıkartır, sonra da bu verilerin şifresini çözerek orijinal mesajı elde eder.



Şekil 4.1. Önerilen sisteminin genel gösterimi

İletişim sürecinin işlem adımları Şekil 4.2' deki akış diyagramında gösterilmiştir. Süreç, göndericiden mesajın ve taşıyıcı (kapak) görüntünün alınmasıyla başlar. İşlemlere devam etmeden önce görüntü çözünürlüğünün gönderilmek istenen mesajı saklamak için yeterli olup olmadığı kontrol edilir. Taşıyıcı olarak 24 bit derinlikli renkli görüntüler kullanılmaktadır ve dolayısıyla piksel başına 3 bit veri gizlenecektir. Bu yüzden mesaj boyutunun görüntünün toplam piksel sayısının 3 katından az olması gerekmektedir. Bu şartın sağlanması halinde işleme, AES algoritması için gerekli olan 96 bitlik başlangıç vektörü ve 256 bitlik anahtarın oluşturulmasıyla devam edilir. Başlangıç vektörü rastlantısal şifreli veriler elde etmek için gereklidir. Sonrasında AES-GCM algoritması kullanılarak mesaj şifrelenir. GCM modunun kullanılmasına bağlı olarak 128 bitlik kimlik doğrulama etiketi de şifrelenmiş mesaja eklenir ve mesaj artık bir fotoğrafa gizlenmesi için hazır hale gelir. Fakat alıcı tarafında şifreli mesajın çözülebilmesi için

kullanılan başlangıç vektörü ve anahtarında alıcıya güvenli bir şekilde ulaştırılması gerekir. Bunun için AES anahtarı da RSA algoritmasıyla şifrelenir. RSA algoritması için gerekli olan açık anahtar alıcıdan edinilir. Böylece sürecin şifreleme kısmı tamamlanmış olur.



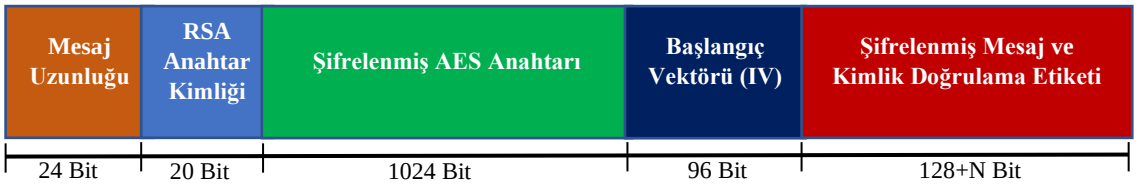
Şekil 4.2. Önerilen yaklaşımın akış diyagramı

Şifreleme aşamasına müteakiben steganografinin uygulanması için gerekli işlem adımları yürütülür. Bunun için ilk olarak taşıyıcı görüntü, veri gömülmeye uygun hale getirilmelidir. Çünkü dijital görüntüler genellikle JPEG veya GIF gibi sıkıştırılmış bir formatta saklanırlar ve piksel değerlerinin okunup değiştirilebilmesi ve değiştirildikten sonra da deformasyona uğramaması için sıkıştırma yapılmayan bir formata dönüştürülmeleri gerekir. Bu amaçla yapılan çalışmada göndericiden alınan taşıyıcı görüntü Bitmap (BMP) formatına dönüştürülür. BMP, grafik dosyalarını herhangi bir sıkıştırma yapmadan saklayan, yaygın olarak kabul gören ve kolay işlenebilen bir dosya formatıdır (Sindhu ve Rajkamal, 2009). Taşıyıcı görüntüde yapılan format değişikliğinin yanı sıra şifrelenmiş veriler de görüntüye gömülebilmesi için binary forma getirilir. Bu şifreli verilerin içerisinde mesajla birlikte şifreleme için kullanılan AES anahtarı, başlangıç vektörü ve kimlik doğrulama etiketi de mevcuttur. Bu şifrelenmiş verilerin her biti, sırasıyla taşıyıcı görüntünün her renk katmanındaki her pikselin en anlamsız bitiyle yer değiştirilir. Böylece önce şifrelenerek güvenliği sağlanan verilerin, sonra da taşıyıcı görüntüye gömülmesiyle gizliliği de sağlanır. Ancak oluşturulan stego-görüntü BMP formatındadır ve bu formatta verilere herhangi bir sıkıştırma işlemi uygulanmadığı için dosya boyutu çok büyük hale gelir. Bundan dolayı stego-görüntü, alıcıya gönderilmeden önce sıkıştırılır. Burada dikkat edilmesi gereken nokta sıkıştırma işlemi için hangi formatın kullanılacağıdır. Çünkü JPEG gibi kayıplı sıkıştırma yapan dosya formatları, görüntüye gizlenen verilerin deforme olmasına ve dolayısıyla bu verilerin görüntüden tekrar çıkarılamamasına sebep olmaktadır. Bu yüzden bu işlem için kayıpsız sıkıştırma yapan PNG formatı tercih edilmiştir. Elde edilen stego-görüntünün PNG formatına dönüştürülmesiyle birlikte görüntü, artık alıcıya gönderilmeye hazır hale gelir.

Alıcı ilk olarak kendisine ulaşan PNG formatındaki stego-görüntüyü, BMP formatına dönüştürür. Devamında ise stego-görüntüde gömülü bulunan binary formdaki şifreli verileri, görüntünün her renk katmanındaki piksellerin en az anlamsız bit değerlerini okuyarak çıkarır. Eldeki görüntünden kaç pikselin okunacağı ve kaç bitlik veri çıkarılacağı ise belli bir modele göre belirlenir. Bu modelin nasıl oluşturulduğundan bölümün devamında bahsedilecektir. Bu çıkarma işlemi sonucunda bir bit katarı elde edilir. Bu bit katarı öncelikle parçalara ayrılıp ayıklanarak şifreli mesaj, başlangıç vektörü ve AES anahtarının bit katarları ayrı ayrı elde edilir ve bunlar şifreleme algoritmaları tarafından işlenebilmeleri için byte formuna getirilir. Mesajın şifresini çözmeden önce kullanılacak anahtarın şifresinin çözülmesi gerekmektedir. Bunun için yalnızca alıcıda bulunan gizli anahtar kullanılarak RSA algoritmasıyla AES anahtarının şifresi çözülür.

Artık şifreli mesajın çözülebilmesi için gerekli veriler sağlanmıştır. Ancak şifre çözme işleminden önce şifreleme kimliğini doğrulamak gerekir. Bu aynı zaman da veri bütünlüğünün doğrulanmasını da sağlar. Çünkü bu işlemin uygulanmasıyla anahtarın ve şifreli verinin birbiriyle eşleşip eşleşmediği veya bunlardan herhangi birinde bir bitlik bile hata olup olmadığı kontrol edilir. Zira bunlardan herhangi birinde bir hata meydana gelmesi şifre çözme işleminin başarısız olmasına neden olacaktır. Bu doğrulama işlemini gerçekleştirmek için GCM çalışma kipi kullanılır. Uygulanan bu işlemle kimlik doğrulama etiketi olarak adlandırılan ve şifrelenen her bir veri için ona bağlı olan özgün bir değer elde edilir. Bu değer de şifreli mesaja eklenip alıcıya ulaştırılarak, alıcının şifreli veriyi kontrol etmesi ve çözülen verinin doğruluğundan emin olması sağlanır (Gövem, 2013). Doğrulama işlemi başarısız olursa göndericiden mesajı tekrar iletmesi istenir, başarılı olursa da görüntüden çıkarılan başlangıç vektörü ve RSA ile şifresi çözülen anahtar kullanılarak AES algoritmasıyla mesajın da şifresi çözülür ve alıcı orijinal mesaja ulaşmış olur.

Önerilen yaklaşımda uygulanan steganografi işleminde verilerin fotoğraflara doğru bir şekilde gömülmesi ve tekrardan doğru bir şekilde çıkarılabilmesi için standart bir veri modeli oluşturulmuştur. Şekil 4.3, bu veri modelini göstermektedir. Buna göre oluşturulan veri modeli, mesaj uzunluğunu belirten 24 bitlik bir kısım, AES anahtarının şifrelenmesi için hangi RSA anahtar çiftinin kullanıldığını temsil eden 20 bitlik bir kısım, 1024 bit uzunluğundaki şifrelenmiş AES anahtarı, 96 bitlik başlangıç vektörü, 128 bitlik kimlik doğrulama etiketi ve N bitlik şifrelenmiş mesaj olmak üzere beş ayrı parçadan oluşmaktadır. Mesaj uzunluğunu temsil eden kısmın 24 bit olması, gönderilebilecek maksimum mesaj boyutunun 16.777.215 bayt, yani yaklaşık olarak 16 Megabayt (MB) olduğunu göstermektedir. Bu değer, günümüzdeki mobil cihazların ortalama kamera çözünürlükleri göz önünde bulundurularak bu cihazlarla çekilen fotoğrafların içerisine 1 bit-LSB yöntemiyle gömülebilecek en büyük veri miktarının yaklaşık olarak tespit edilmesiyle belirlenmiştir. Günümüzde mobil cihazlar ortalama olarak 32-48 Megapiksel (MP) kamera çözünürlüklerine sahiptir. Bu değerler bu cihazlarla çekilen fotoğrafların, 32.000.000 ile 48.000.000 adet pikselden oluşacağı anlamına gelir. Her piksele 3 bit veri gizlenebileceğinden bu fotoğraflar içerisine gömülebilecek maksimum veri boyutu da yaklaşık olarak 12-18 MB olacaktır. Bu nedenle oluşturulan veri modelinde gönderilebilecek en büyük mesaj boyutu yaklaşık 16 MB olarak belirlenmiştir ve bu değer mesaj olarak gönderilebilecek birçok doküman, belge, fotoğraf vb. dosyaları şifreleyip gizlemek için gayet yeterlidir.



Şekil 4.3. Steganografide kullanılan veri modeli

Şekil 4.3' deki RSA anahtar kimliği kısmı ise AES anahtarını şifrelemek için kullanılan RSA anahtar çiftini temsil eden özgün bir değerdir. Çünkü RSA için kullanılan anahtar çiftleri, güvenliğin artırılması için belli aralıklarla güncellenmektedir. Bu güncelleme işleminde alıcı, oluşturduğu anahtar çiftlerinden açık olanı göndericiyle paylaşırken, gizli olanı da sadece kendisinin erişebileceği güvenli bir alanda saklamalıdır. Alıcı daha sonra muhafaza ettiği bu gizli anahtarları, kendisine ulaşan mesajlardaki AES anahtarlarının şifresini çözmek için kullanacaktır. Fakat bunun için AES anahtarlarının şifrelenmesinde hangi RSA anahtar çiftinin kullanıldığını bilmek gerekir. Bu amaçla alıcı tarafında oluşturulan her bir RSA anahtar çiftine özgün bir kimlik numarası atanır ve açık anahtarla beraber bu kimlik numarası da göndericiyle önceden paylaşılır. Böylece gönderici AES anahtarını şifrelerken hangi RSA açık anahtarını kullandığını belirtmek için bu kimlik numarasını da gömülecek veriler arasına ekler. Alıcıda mesaj kendisine ulaştığı zaman AES anahtarını deşifre etmek için hangi RSA gizli anahtarını kullanacağını bu kimlik numarasına bakarak anlar ve böylece AES anahtarının şifresini doğru bir şekilde çözebilir. Oluşturulan veri modelinde RSA anahtar kimliği için 20 bitlik bir alan ayrılmıştır. Bu da uygulamadaki bir kullanıcının diğer her bir kullanıcıyla mesajlaşması için 1.048.576 adet RSA anahtar çifti oluşturabileceği anlamına gelmektedir. Oluşturan veri modelinin üçüncü kısmı şifrelenmiş AES anahtarı için ayrılmıştır. AES anahtarının asıl uzunluğu 256 bittir. Fakat bu 256 bit AES anahtarı, 1024 bitlik bir açık anahtar kullanılarak RSA algoritmasıyla şifrelenmektedir ve RSA algoritmanın çıktısı anahtar boyutuna eşit olur. Bundan dolayı AES anahtarının RSA ile şifrelenmiş hali 1024 bit boyutunda olmaktadır. Son olarak başlangıç vektörü ve şifrelenmiş mesajın veri modeline eklenmesiyle birlikte işlem tamamlanır. AES-GCM algoritması çalışmasına bağlı olarak 128 bitlik kimlik doğrulama etiketi de şifrelenmiş mesajın içerisinde dahili olarak mevcuttur.

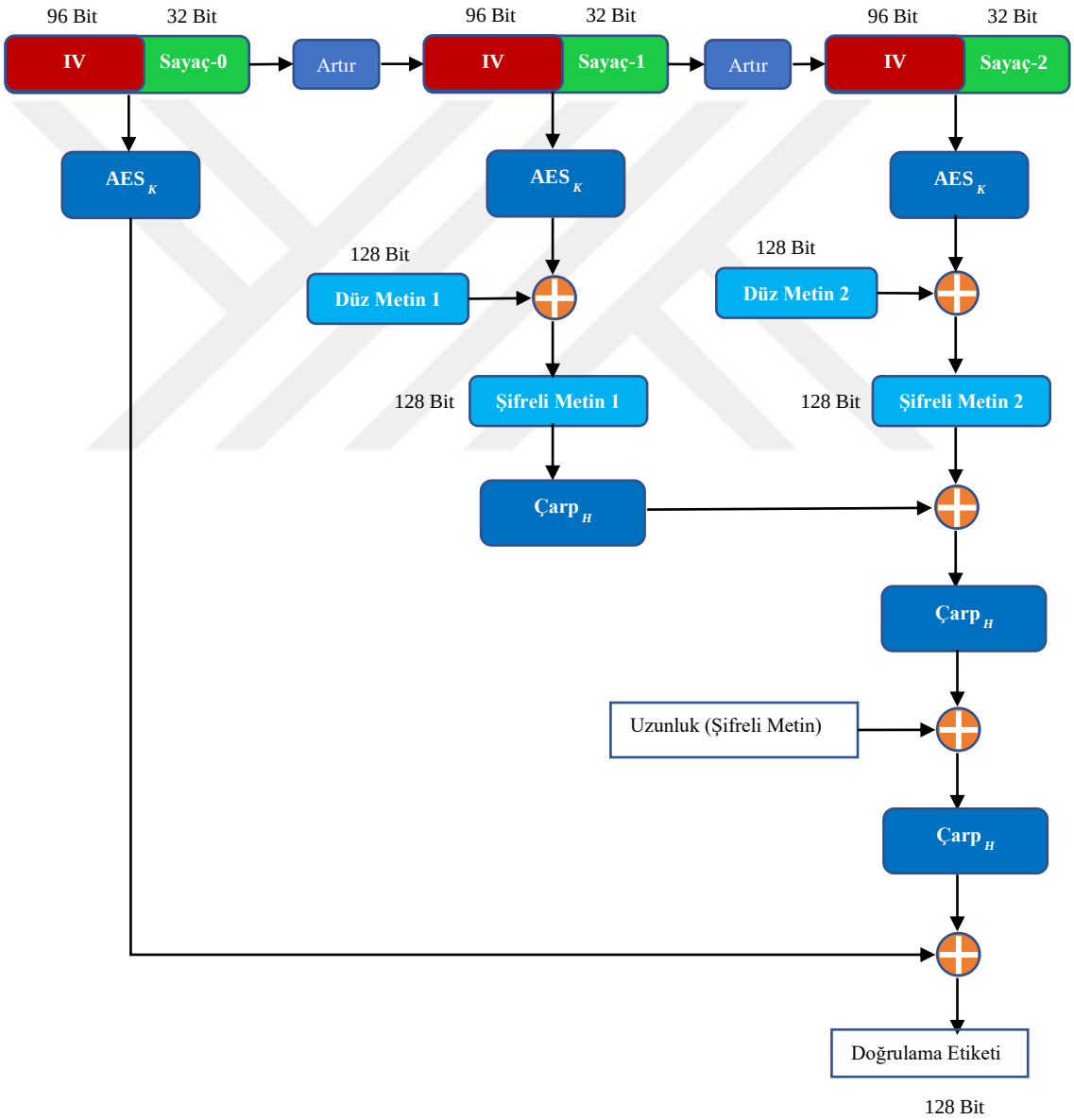
Bölümün devamında yukarıda genel yapısı itibariyle bahsedilen şifreleme ve steganografi uygulamalarının iç yapısı detaylarıyla birlikte açıklanmıştır. Önerilen metotta şifreleme işlemi, Şekil 4.4'te görüldüğü üzere GCM şeması ve AES algoritması

kullanılarak gerçekleştirilmiştir. GCM, şifreleme işlemi için sayaç modunu, kimlik doğrulama işlemi içinse ikili bir Galois (sonlu) alanı üzerinde evrensel hash fonksiyonunu kullanan bir blok şifreleme çalışma şemasıdır (Dworkin, 2007). Sayaç modu hem ardışık düzende hem de paralel olarak uygulanabilmektedir ve minimum hesaplama gecikmesine sahiptir. Bu yüzden yüksek hızlı şifreleme için en iyi seçenektir (Lipmaa ve ark., 2000). Şifreleme işleminin hızı kadar kimlik doğrulama işleminin hızı da önemlidir. Şifre Bloğu Zinciri Mesaj Doğrulama Kodu (Cipher Block Chaining Message Authentication Code, CBC-MAC) ve Tek-anahtar Mesaj Doğrulama Kodu (One-key MAC, OMAC) ile kimlik doğrulamalı şifreleme için bunları kullanan CBC-MAC' e sahip Sayaç (Counter with CBC-MAC, CCM) ve Şifrele Doğrula Çevir (Encrypt Authenticate Translate, EAX) gibi modlar paralelleştirilemez ve bu nedenle yüksek veri hızları için uygun değildir (Svenda, 2016). Carter–Wegman Sayaç (Carter–Wegman Counter, CWC) modu ise bu problemlere sahip olmasa da yüksek hızlı uygulamalar için daha az uygundur. Özellikle, CWC' nin mesaj doğrulama bileşeni, uygulama maliyetleri AES sayaç modunun bile üzerinde olan 127 bit tamsayı çarpma işlemlerini kullanır. Buna karşılık, GCM' de kimlik doğrulama sağlamak için kullanılan ikili alan çarpması, yüksek hızlardaki sayaç modu maliyetinin bir kısmıyla kolayca uygulanır (McGrew and Viega, 2004).

Önerilen metotta şifreleme işlemi için GCM şemasına dört girdi sağlanmaktadır. Şekil 4.4' de görüldüğü üzere bunlar 96 bitlik IV, 32 bitlik sayaç, 256 bitlik gizli anahtar ve düz metindir. Gizli anahtar, şekilde AES algoritmasında kullanılmak üzere *K* olarak temsil edilmiştir. IV' nin birincil amacı benzersiz olmaktır. Yani, sabit bir anahtar için şifreleme işleminin her çağrısında farklı olmalıdır. Başka bir deyişle bir IV aynı anahtarla birden fazla kullanılamaz. IV değerlerinin farklılığı yüksek olasılıksa IV' nin rastgele oluşturulması kabul edilebilir. NIST tarafından GCM üzerine hazırlanan 800-38D kodlu dokümantasyonda, kimliği doğrulanmış şifreleme işlevinin iki veya daha fazla farklı girdi veri kümesinde aynı IV ve aynı anahtarla çağrılma olasılığının 2^{-32}'den büyük olmaması gerektiği ifade edilmiştir. (Dworkin, 2007). Bu gereksinime uyum, GCM'nin güvenliği için çok önemlidir. Belirli bir anahtarla gerçekleştirilen şifreleme işlemlerinin tümünde, bir IV bile tekrarlanırsa, uygulama saldırılara karşı savunmasız olabilmektedir. Pratikte bu gereklilik neredeyse anahtarın gizliliği kadar önemlidir. Hedef kullanıcıları arasında oluşturulan herhangi bir GCM anahtarı ise yüksek olasılıkla yeni olmalıdır. Yani önceki herhangi bir anahtarla aynı olmamalıdır.

Şifreleme işleminde önce, IV ve 32 bitlik sayaç birleştirilerek 128 bitlik bloklar oluşturulur. Bunlar, AES algoritmasıyla şifrelenerek 128 bit uzunluğunda şifreli bloklar

elde edilir. Şifrelenecek her bir sonraki blok için sayaç değeri artırılarak devam edilir. Şifreli blokların 128 bitlik düz metin bloklarıyla XOR' lanmasıyla da şifreli metin blokları üretilmiş olur. Şifreleme işlemi sonucunda düz metinle aynı boyutta olan bir şifreli metin ve 128 bit uzunluğunda bir kimlik doğrulama etiketi olmak üzere iki farklı çıktı elde edilir. Kimlik doğrulama etiketini üretmek için ikili bir sonlu alan üzerinde çarpma işlemleri uygulanır. Sonlu bir alan, çarpma ve toplama işlemleriyle tanımlanır. Bu işlemler değişebilirlik, birleşebilirlik ve dağılabilirlik gibi çarpma ve toplamadan beklenen temel cebirsel özelliklere uyar (McGrew and Viega, 2004).



Şekil 4.4. Önerilen metotta uygulanan şifreleme işlemi

AES GCM ile bir n-bit düz metni şifrelemek için gereken blok şifreleme çağrılarının sayısı (n/128) + 1'e eşittir. Ayrıca, Galois alanı (2^{128}) üzerinde de aynı sayıda çarpma işlemi gerçekleştirilir. Bunların yanı sıra çarpma işlemlerinde kullanılmak üzere *H* hash anahtarını hesaplamak için ek bir blok şifre çağrısı gereklidir. Şekil 4.4' de Galois alanı üzerinde yapılan çarpma işlemleri Çarp$_H$ fonksiyonuyla ifade edilmiştir ve bu ifadedeki *H* simgesi, bahsedilen hash anahtarını temsil etmektedir. Hash anahtarı 128 bit uzunluğundadır ve 128 bitlik bir 0 vektörünün, *K* anahtarı kullanılarak AES algoritmasıyla şifrelenmesi sonucunda elde edilir. Çarp$_H$ fonksiyonu da bu hash anahtarını kendisine beslenen değerle çarparak 128 bitlik bir sonuç üretir.

Çarp$_H$ fonksiyonuna ilk çağrılışında, AES algoritması kullanılarak elde edilen ilk şifreli metin bloğu beslenir ve son çağrılışı hariç olmak üzere geri kalan tüm iterasyonlarda, bir önceki çarpımdan elde edilen sonuçla o iterasyondaki şifreli metin bloğunun XOR'lanması sonucu oluşan 128 bitlik değer beslenir. Son çağrılışında ise fonksiyona beslenen girdi, şifreli metnin uzunluğunu ifade eden bir tamsayının 128 bitlik temsili ile sondan bir önceki çarpım fonksiyonundan elde edilen 128 bitlik sonucun XOR' lanmasıyla oluşturulur. Çarp$_H$ fonksiyonun son çağrılışında elde edilen sonuçla, ilk iterasyonda oluşturulan sayaç bloğunun şifrelenmesiyle elde edilen değerin XOR 'lanmasıyla da 128 bitlik kimlik doğrulama etiketi üretilir.

Şifre çözme işlemine ise anahtar, IV, şifreli metin ve kimlik doğrulama etiketi olmak üzere beş farklı girdi sağlanır. Buna karşılık şifreli metinle aynı uzunlukta bir düz metin veya girdilerin doğru olmadığını gösteren özel bir başarısızlık sembolü olmak üzere tek bir çıktı vardır. Kimliği doğrulanmış şifre çözme işlemi, girdileri şifreleme işlemi tarafından aynı anahtarla oluşturulmadığında yüksek olasılıkla başarısız olarak dönecektir. Şifre çözme işlemi ile hesaplanan kimlik doğrulama etiketi, şifreli metin ile ilişkili kimlik doğrulama etiketi ile karşılaştırılır. İki etiket hem uzunluk hem de değer olarak eşleşirse, şifreli metin çözülür ve düz metin döndürülür. Aksi takdirde, özel başarısızlık sembolü döndürülür (McGrew and Viega, 2004).

Önerilen metotta şifreleme işlemleri dışında gerçekleştirilen bir diğer aşamada steganografi uygulamasıdır. Şekil 4.5'te, gerçekleştirilen bu uygulamanın kaba kodu verilmiştir. Algoritmada, şifreleme aşamasını müteakiben oluşturulan binary formdaki veri modeli ve kullanıcı tarafından seçilen yeterli çözünürlüğe sahip bir görüntü olmak üzere iki farklı girdi alınır. Bunlara karşılık içerisine veri gizlenmiş bir stego görüntü çıktı olarak elde edilir. Sürecin başlangıcında önce seçilen fotoğrafın çözünürlüğünün gizlenecek verinin boyutu için yeterli olup olmadığı kontrol edilmektedir. Bunun için ilk

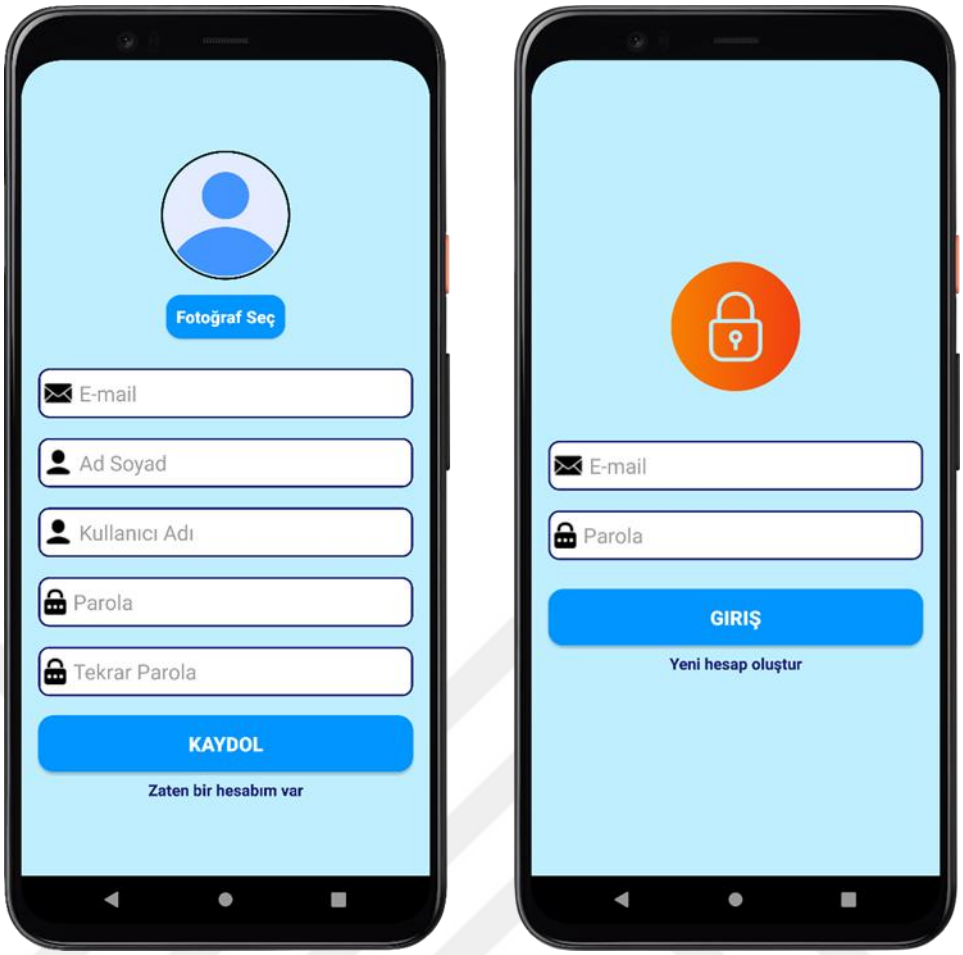
olarak gizlenecek verinin uzunluğu hesaplanır. Veri zaten binary formda bir dizeden oluştuğu için bit cinsinden uzunluğu hesaplamada dizedeki karakter sayısını tespit etmek yeterlidir. Daha sonrasında kullanıcıdan bir fotoğraf seçmesi istenir ve seçilen fotoğrafın yatay ve dikey çözünürlükleri kullanılarak görüntüdeki toplam piksel sayısı hesaplanır. Görüntüdeki pikseller kırmızı, mavi ve yeşil olmak üzere üç renk katmanından oluşmakta ve bir pikselin her renk seviyesi 0 ile 255 arasında değerler alan 8 bit uzunluğundaki pozitif bir tam sayıyla ifade edilmektedir. Üç renk katmanının her biri için bu 8 bitlik değerlerin en anlamsız bitleri veri gizlemek için kullanılmaktadır ve her renk katmanına sadece bir bit saklanmaktadır. Bu yüzden piksel başına 3 bit veri gizlenebilmektedir ve işlemin gerçekleştirilebilmesi için veri uzunluğunun görüntüdeki toplam piksel sayısının üç katından fazla olmaması gerekir. Şekil 4.5' de görüldüğü üzere bu şartın sağlanmaması halinde tekrardan başka bir fotoğraf seçimi gerçekleştirilir. Uygun bir görüntünün seçilmesinin ardından bu görüntü işlenmek üzere BMP formatına dönüştürülür. Bu dönüşümle her bir renk katmanı için görüntünün piksel değerlerini içeren birer matris elde edilir. Bu matrislerin elemanları üzerinde değişimler uygulanarak veri gizleme işlemi gerçekleştirilir. İlk olarak bu işlem için gereken piksel sayısı tespit edilir. Piksel başına üç bit yerleştirildiği için bu değer veri uzunluğunun üçe bölünmesiyle hesaplanır. Eğer uzunluk değeri üçe tam bölünmüyorsa sonuç, bölümün tam kısmı olur. Bu değer üç katmanı üzerinde de işlem yapılacak piksellerin sayısını ifade eder. Bu sayı adedince döngü kurularak veri gizlemeye (0,0) koordinatındaki ilk pikselden başlanır ve yatay yönde ilerleyerek devam edilir. Bu döngü boyunca işlenen her pikselin her renk katmanındaki en anlamsız biti, binary formdaki verinin sıradaki bitiyle değiştirilir. Bu değişim sırasında işlenen her pikselden sonra yatay çözünürlüğün sonuna gelinip gelinmediği kontrol edilir. Yatay çözünürlüğün sonuna gelinmişse dikey indeks bir artırılır ve yatay indeks de sıfırlanır. Şekil 4.5' de görüntünün yatay ve dikey indeksleri sırasıyla x ve y olarak temsil edilmiştir. Eğer veri uzunluğu üçün tam katıysa işlem burada tamamlanır; fakat aksi halde bir veya iki bit arta kalmış demektir. Artan bu bitler ise ekstra bir pikselin bir veya iki katmanına gömülür. Bu yüzden başlangıçta veri uzunluğunun üçe göre modu alınarak artan bit varsa bunların kaç tane olduğu tespit edilir ve bu değerin sıfırdan farklı olması durumunda ekstra bir piksel üzerinde daha işlem yapılır. Eğer artan bit sayısı birse işlenecek ekstra pikselin sadece kırmızı renk değeri, iki ise hem kırmızı hem de yeşil renk değerleri veri gizlemek için kullanılır. Son olarak da işlenen görüntünün kayıpsız sıkıştırma yapan PNG formatına dönüştürülmesiyle işlem tamamlanır ve stego görüntü elde edilir.

```
U = Uzunluk (binary_veri);
piksel_sayisi = U / 3;
artan_bit = U % 3;
index,x,y,yatay_cozunurluk = 0;
Goruntu = null;

do{
  Goruntu = Fotograf_Sec( );
  yatay_cozunurluk = Goruntu.Yatay;
  dikey_cozunurluk = Goruntu.Dikey
}while(U > yatay_cozunurluk * dikey_cozunurluk * 3)

Bitmap = BMP_Donusumu(Goruntu)
for (i = 0,.....,piksel_sayisi) {
    Bitmap.piksel(x,y).kirmizi(LSB) = binary_veri[index++];
    Bitmap.piksel(x,y).yesil(LSB) = binary_veri[index++];
    Bitmap.piksel(x,y).mavi(LSB) = binary_veri[index++];
    x++;
    if (x > yatay_cozunurluk - 1) {y++; x=0;}
}
if (artan_bit == 1)
   if (x > yatay_cozunurluk - 1) {y++; x=0;}
   Bitmap.piksel(x,y).kirmizi(LSB) = binary_veri[index++];
}
else if (artan_bit == 2) {
   if (x > yatay_cozunurluk - 1) {y++; x=0;}
   Bitmap.piksel(x,y).kirmizi(LSB) = binary_veri[index++];
   Bitmap.piksel(x,y).yesil(LSB) = binary_veri[index++];
}
Stego_goruntu = PNG_Formatlama(Bitmap);
```

Şekil 4.5. Önerilen metotta uygulanan steganografinin kaba kodu

4.2. Uygulama

Bu başlığa kadar olan kısımda önerilen yaklaşımın açıklaması tamamlanmıştır ve bölümün devamında bu yaklaşım kullanılarak geliştirilen uygulama açıklanacaktır. Uygulama, Java diliyle Android platformunda çalışmak üzere geliştirilmiştir. Anlık iletişim sağlanması için Firebase gerçek zamanlı veritabanı hizmeti ve sunucu ortamında saklanması gereken stego-görüntüler içinse Firebase depolama hizmeti kullanılmaktadır. Şekil 4.6, uygulamanın kayıt ve giriş ekranlarını göstermektedir. Uygulama ilk yüklendiğinde hesap oluşturulup sisteme kaydolunması gerekmektedir. Bunun için kayıt ekranına yönlendirilen kullanıcı, e-mail, ad, soyadı, kullanıcı adı, parola bilgilerini girip bir profil fotoğrafı seçerek hesap oluşturabilir.



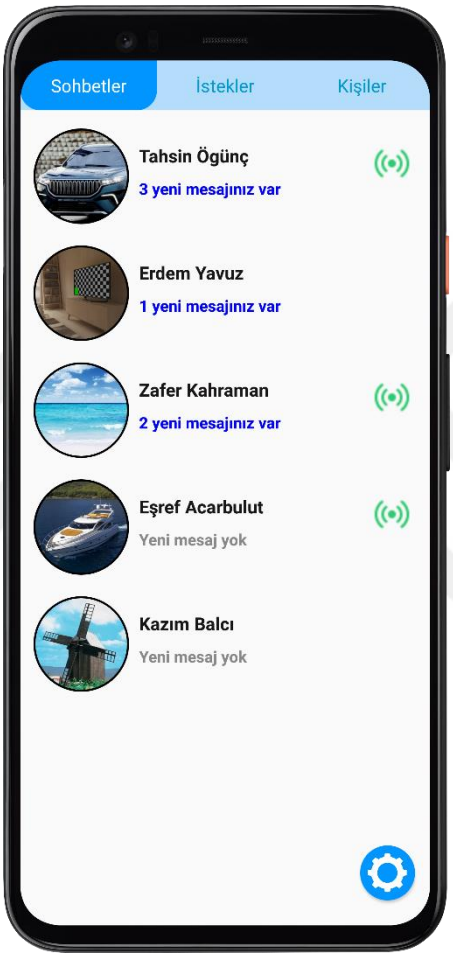
Şekil 4.6. Kayıt ve giriş ekranları

Hesap oluşturulduktan sonra kullanıcının doğrudan oturumu açılarak hesabına giriş yapması sağlanmaktadır. Mevcut hesabı bulunan kullanıcılar ise e-mail ve parola bilgilerini girerek sisteme giriş yapabilmektedir. Yetkilendirme işlemi ve oturum yönetimi de Firebase tarafından sunulan kimlik doğrulama (authentication) hizmeti kullanılarak gerçekleştirilmektedir. Bu hizmet sayesinde verilere erişim izinleri kolaylıkla kontrol edilebilir. Kullanıcılar, oturum açtıktan sonra Şekil 4.7' de yer alan ve sohbetler, istekler ve kişiler sekmelerinden oluşan uygulamanın ana sayfasına yönlendirilir. Kullanıcının başka bir katılımcıyla iletişim kurabilmesi için öncelikle bu sayfadaki kişiler sekmesi altından dilediği bir katılımcıya istek göndermesi ve gönderdiği kişinin de bu isteği kabul etmesi gerekmektedir. Kullanıcı kendisine gelen istekleri ise istekler sekmesi altından görüntüleyebilir ve bunları kabul veya reddedebilir.

Şekil 4.7. Kişiler ve istekler ekranı

Katılımcıların birbirlerine istek göndermesi aslında arka planda bir anahtar paylaşım sürecini başlatır. Bir katılımcı, başka bir katılımcıya istek gönderdiğinde öncelikle bir RSA anahtar çifti oluşturur. Bunun açık olanını mesajlaşma sürecinde kullanabilmesi için muhatabına gönderir, gizli olanını ise Android tarafından sunulan ve şifreleme anahtarlarını güvenle muhafaza etmeyi sağlayan anahtar deposu (keystore) sisteminde saklar. Bu sisteme giren bir anahtar, herhangi bir şekilde dışarı aktarılamaz olarak kalacak şekilde kriptografik işlemler için kullanılabilir. Anahtar deposu sistemi, anahtar materyalinin uygulama süreçlerinden ve bir bütün olarak Android cihazından çıkarılmasını engelleyerek anahtarın cihaz dışında yetkisiz kullanımını önler (Google, 2021). Kullanıcı anahtar deposunda sakladığı bu gizli anahtarı, daha sonra istek gönderdiği kişiden bir mesaj geldiğinde bu mesajın çözülebilmesi için kullanacaktır. İstek gönderilen kişi de bunu kabul etmesi halinde aynı şekilde istekte bulunan kişiyle kendi açık anahtarını paylaşır ve böylece taraflar bu açık anahtarları birbirleriyle mesajlaşmak için kullanırlar. İstek gönderilmesi ve kabul edilmesi sürecinin tamamlanmasının

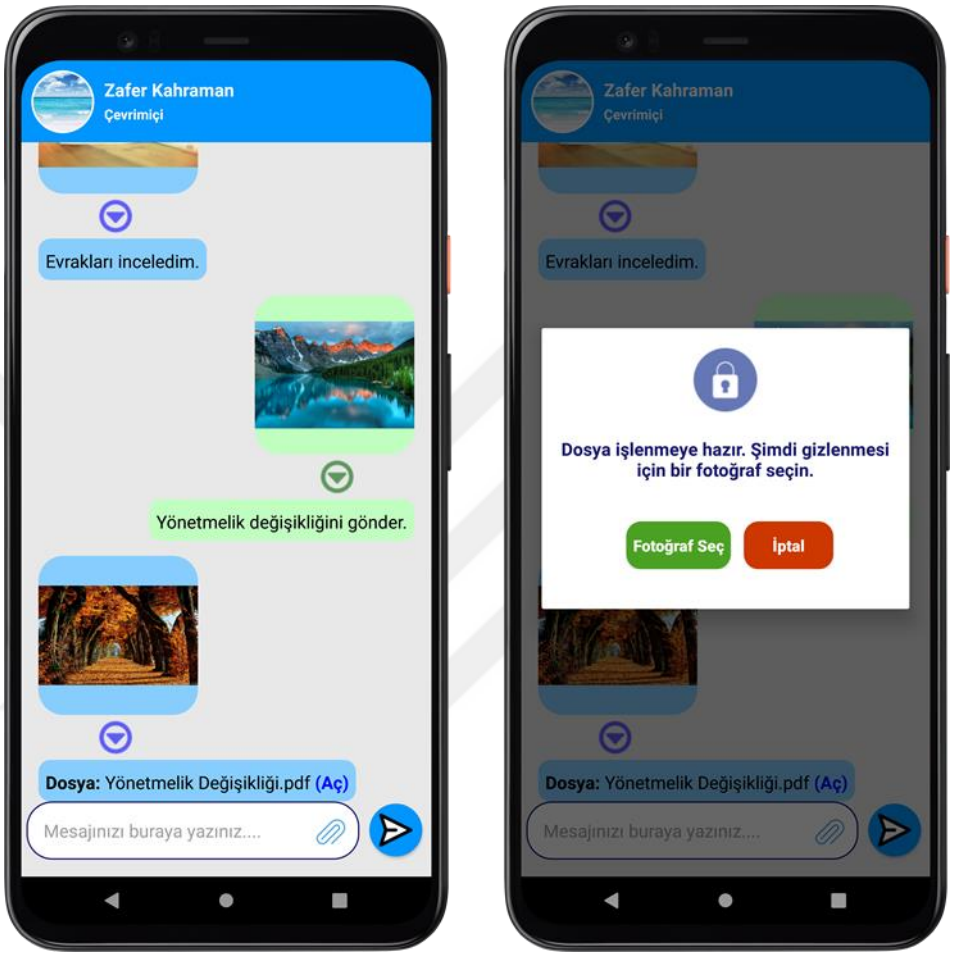
ardından taraflar birbirlerinin sohbetler sekmesi altında görüntülenmeye başlar. Buna ait ekran görüntüsü Şekil 4.8’ de gösterilmektedir. Katılımcılar bu sekme altından iletişim kurdukları kişilerin oturum durumlarından ve kendilerine gönderdikleri yeni mesajlardan haberdar olabilirler. Ekranda görünen yeşil renkli sinyal ikonu o katılımcının oturumunun aktif olduğunu ifade eder. Kişilerin isimlerinin altında ise kaç tane yeni mesaj gönderdikleri gösterilmektedir.



Şekil 4.8. Sohbet listesi ekranı

Şekil 4.8’ deki ekrandan bir kişinin seçilmesi halinde kullanıcı, o kişiyle iletişim kurabileceği Şekil 4.9’ da gösterilen mesajlaşma sayfasına yönlendirilir. Ekranın üst kısmında iletişim kurulan kişiye ait profil fotoğrafı ve isim bilgisi yer almaktadır. Bu bölümün altında ise o kişiden alınan ve o kişiye gönderilen mesajlar listelenmektedir. Mesajlar, içerisine gömülü olduğu stego-görüntüyle birlikte ekranda gösterilir. Sol taraftaki görüntüler alınan mesajları, sağ taraftaki görüntüler ise gönderilen mesajları barındırmaktadır. Stego-görüntünden mesaj çıkarımı sadece seçilen görüntüler için

gerçekleştirilir. Böylece sistemdeki iş yükü de hafifletilmiş olur. Kullanıcının bir stego-görüntünün içerisindeki mesajı görüntülemesi için o görüntünün üzerine dokunmasıyla mesaj çıkarım süreci başlatılır. İlk önce şifreli veriler, görüntü içerisinden okunarak elde edilir. Sonra da bu verilerin şifresinin çözülmesiyle asıl mesaj kullanıcıya gösterilir.



Şekil 4.9. Mesajlaşma ekranı ve dosya gönderimi

Uygulamada yazıyla mesajlaşmaya ek olarak dosya gönderimi de mevcuttur. Bu işlem için uygulanacak süreç, mesajlaşma ekranında yer alan ataş ikonuna dokunulmasıyla başlar. Kullanıcıdan öncelikli olarak göndermek istediği dosyayı seçmesi talep edilir. Bu dosya, Taşınabilir Belge Biçimi (Portable Document Format, PDF), Microsoft Word belgesi, JPEG, PNG formatlarında veya herhangi bir başka formatta olabilir. Dosya seçiminin ardından Şekil 4.9' da sol kısımda yer alan diyalog penceresi kullanıcıya gösterilir. Kullanıcı bu pencereden "Fotoğraf Seç" butonuna dokunduğunda bir taşıyıcı görüntü seçmek üzere fotoğraf galerisine yönlendirilir. Gönderilmek istenen dosya şifrelendikten sonra galeriden seçilen fotoğrafa gömülür ve bu fotoğraf alıcıya

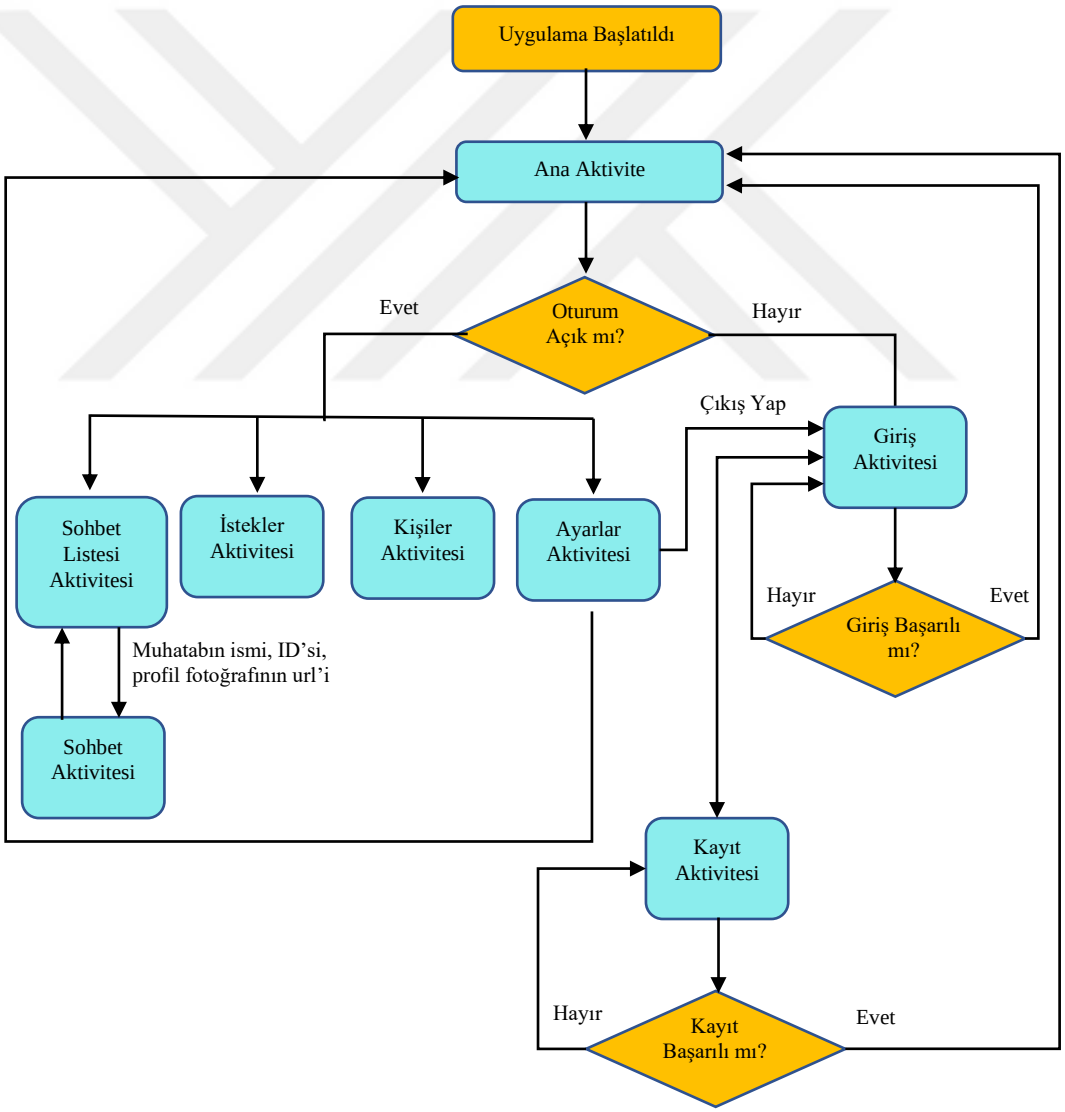
gönderilmek üzere sunucuya iletilir. Alıcıya ulaşan görüntüden çıkarılan şifreli dosyalar deşifre edildikten sonra cihazda dosya yöneticisi tarafında görüntülenebilmesi için hafızada belirli bir konuma kaydedilir. Mesajlaşma sürecinde dikkat edilmesi gereken nokta gönderilen mesajların yedeklenmesinin gerekliliğidir. Çünkü şifreleme işleminde simetrik bir yöntem olan AES' in yanı sıra asimetrik bir yöntem olan RSA algoritması da kullanılmaktadır. Buna bağlı olarak bir şifreli mesajı göndericisi dahi çözemez, sadece o mesajın alıcısı şifreyi çözebilir. Çünkü mesajı çözmek için gerekli olan gizli anahtar sadece o mesajın alıcısında mevcuttur. Bundan dolayı gönderilen mesajlarında uygulama içerisinde tekrar görüntülenebilmesi ancak yedeklenmesiyle mümkündür. Yedekleme işlemi için SQLite kullanılmaktadır. SQLite, verileri depolamak için kullanışlı bir gelişmiş veritabanı sistemidir. Kullanımı kolaydır ve çoklu platform tarafından desteklenir (Bhosale ve ark., 2015). Temelde daha çok android tabanlı uygulamalarda tercih edilmektedir. Alınan mesajların ise yedeklenmesine gerek yoktur, çünkü bunların daha sonra da tekrar çözülebilmesi için gerekli olan anahtarlar zaten alıcılarında mevcuttur.

Bölümün devamında uygulamanın yaşam döngüsü açıklanmıştır. Ancak bunu ifade edebilmek için Android uygulamalarının temel yapısından kısaca bahsetmek gerekir. Android uygulamalarının en temel bileşenleri aktivitelerdir. Bir aktivite, kullanıcıların arama yapmak için numara çevirmek, galeriden fotoğraf seçmek gibi çeşitli etkileşimlerde bulunabilecekleri bir ekran sağlar. Başka bir deyişle uygulamanın kullanıcı arabirimini çizdiği pencereyi oluşturur. Bu pencere tipik olarak ekranı doldurur, ancak ekrandan daha küçük olabilir ve diğer pencerelerin üzerinde yer alabilir. Çoğu uygulama birden çok ekran sunar, yani birden çok aktivite içerir. Tipik olarak, uygulamadaki bir aktivite, kullanıcı uygulamayı başlattığında görünen ilk ekran olan ana aktivite olarak belirtilir. Her aktivite daha sonra farklı eylemler gerçekleştirmek için başka bir aktivite başlatabilir. Örneğin, basit bir e-posta uygulamasındaki ana aktivite, gelen kutusunu gösteren bir ekran sağlayabilir. Buradan ana aktivite, e-posta yazma ve gelen e-postaları açma gibi farklı görevler için ekranlar sağlayan diğer aktiviteleri başlatabilir (Google, 2021). Şekil 4.10, geliştirilen uygulamanın yaşam döngüsünü göstermektedir ve görüldüğü üzere uygulama, 8 adet temel aktivite içermektedir. Şekilde bu aktivitelerin ne zaman başlatıldığı ve birbirleriyle olan etkileşimleri gösterilmiştir. Uygulamanın başlatılmasıyla beraber ilk olarak ana aktivite çalışır. Ana aktivite, Şekil 4.8'de görülen ekranın üst kısmındaki sekmeleri ve bu sekmelere bağlı sayfaları barındıran uygulamanın ana parçasıdır. Fakat sayfaların içeriği farklı aktiviteler tarafından sağlanır. Ana aktivite

başlatıldığı zaman ilk olarak oturum açmış mevcut kullanıcının olup olmadığını kontrol eder. Eğer oturum açmış kullanıcı yoksa oturum açılabilmesi için giriş aktivitesi başlatılır. Oturum açmış mevcut bir kullanıcı varsa da doğrudan sohbet listesi aktivitesi çalıştırılır. Bu aktivite, ana aktive altındaki sohbetler sekmesi altında yer alan sayfada kullanıcının mesajlaştığı kişileri, bunların oturum durumlarını ve gelen yeni mesajların sayısını gösteren bir ekran sağlar. Giriş aktivitesi ise kullanıcıya e-mail ve parola bilgilerini girmesi için gerekli bileşenlerini içeren bir arabirim sunar. Kullanıcının bu bilgileri girmesiyle birlikte oturum açma işleminin başarılı olup olmadığı kontrol edilir ve buna göre başarılı olması halinde ana aktivite tekrardan başlatılır. Bu durumda ana aktivite tekrardan oturum kontrolü yapar ve bu kontrolün başarılı olarak sonuçlanmasıyla birlikte sohbet listesi aktivitesini başlatır. Ayrıca mevcut hesabı bulunmayan kullanıcıların yeni hesap oluşturabilmeleri için giriş aktivitesi üzerinden kayıt aktivitesine bir yönlendirme linki bulunur. Bu linke tıklanması sonucunda kayıt aktivitesi başlatılır. Bu aktivitede kayıt işleminin başarılı olması durumunda doğrudan oturum açma işlemi gerçekleştirilir ve oturum açma işlemi de başarıyla sonuçlanırsa ana aktivite tekrardan başlatılır. Kayıt ve giriş aktivitelerinde yürütülen işlemler başarısız olursa yeni bir aktivite başlatılmaz ve hatayı açıklayan bir mesaj gösterilir.

Yukarıda bahsedildiği ve Şekil 4.10’ da görüldüğü üzere ana aktivite uygulamanın merkezi niteliğindedir. Sadece iki aktivite dışında diğer tüm aktiviteler ana aktivite tarafından başlatılır. Bunlar; sohbet listesi, istekler, kişiler, ayarlar ve giriş aktivitelerdir. Ana aktivite oturum açılmışsa ilk olarak sohbet listesi aktivitesini başlatsa da istekler ve kişiler aktiviteleri de ana aktivite tarafından sunulan sekmeler vasıtasıyla yine kendisi tarafından başlatılır. Kişiler aktivitesi, uygulamaya kaydolmuş tüm kullanıcıların listesini içeren ve bu kullanıcılara istek gönderme imkanı sunan bir ekran sağlar. Bu aktivite üzerinden diğer katılımcılara sohbet isteği talebinde bulunulabilir ve bu talebin kabul edilmesi halinde taraflar birbirleriyle mesajlaşmaya başlayabilir. İstekler aktivitesi ise kullanıcının istek aldığı diğer katılımcıların bir listesini ve bu isteklerin kabul veya reddedilebilmesi için gerekli arabirimleri içeren bir ekran hazırlar. Gönderilen ve alınan isteklerin kabul edilmesi halinde isteklerin muhatabı olan diğer kişiler sohbet listesi aktivitesi tarafından sağlanan ekranda gösterilir. Bu ekrandan bir muhatabın seçilmesiyle beraber muhatap kişiden gelen mesajları okumak ve mesaj göndermek için gerekli arabirimleri içeren sohbet aktivitesi başlatılır. Bu aktivite bir tek sohbet listesi aktivitesi tarafından başlatılabilir. Ana aktivite de dahil olmak üzere diğerleri üzerinden buraya yönlendirme imkanı yoktur. Ayarlar aktivitesi de yine ana aktivite tarafından başlatılan

aktivitelerden biridir. Kullanıcının kişisel bilgilerinin ve parolasının güncellenmesi işlemlerini gerçekleştirir. Ayrıca oturum sonlandırma işlemi de yine bu aktivite tarafından yürütülür. Ayarlar aktivitesi oturumun sonlandırması halinde giriş aktivitesini başlatır. Bunların yanı sıra kayıt aktivitesi de sohbet aktivitesiyle beraber ana aktivite tarafından başlatılmayan aktivitelerden birisidir. Çünkü giriş aktivitesi tarafından başlatılır. Ayrıca birbirini başlatabilen tek ikili ise kayıt ve giriş aktiviteleridir. Bunların sağladığı arabirimlerde birbirlerine yönlendirilebilmeleri için linkler bulunmaktadır. Ayrıca aktiviteler arası veri aktarımı bir tek sohbet listesi ve sohbet aktiviteleri arasında tek yönlü olarak gerçekleşir. Çünkü sohbet listesinden seçilen bir kişinin bilgileri mesajlaşma sürecinin başlaması ve devam etmesi için sohbet aktivitesine iletilmelidir.

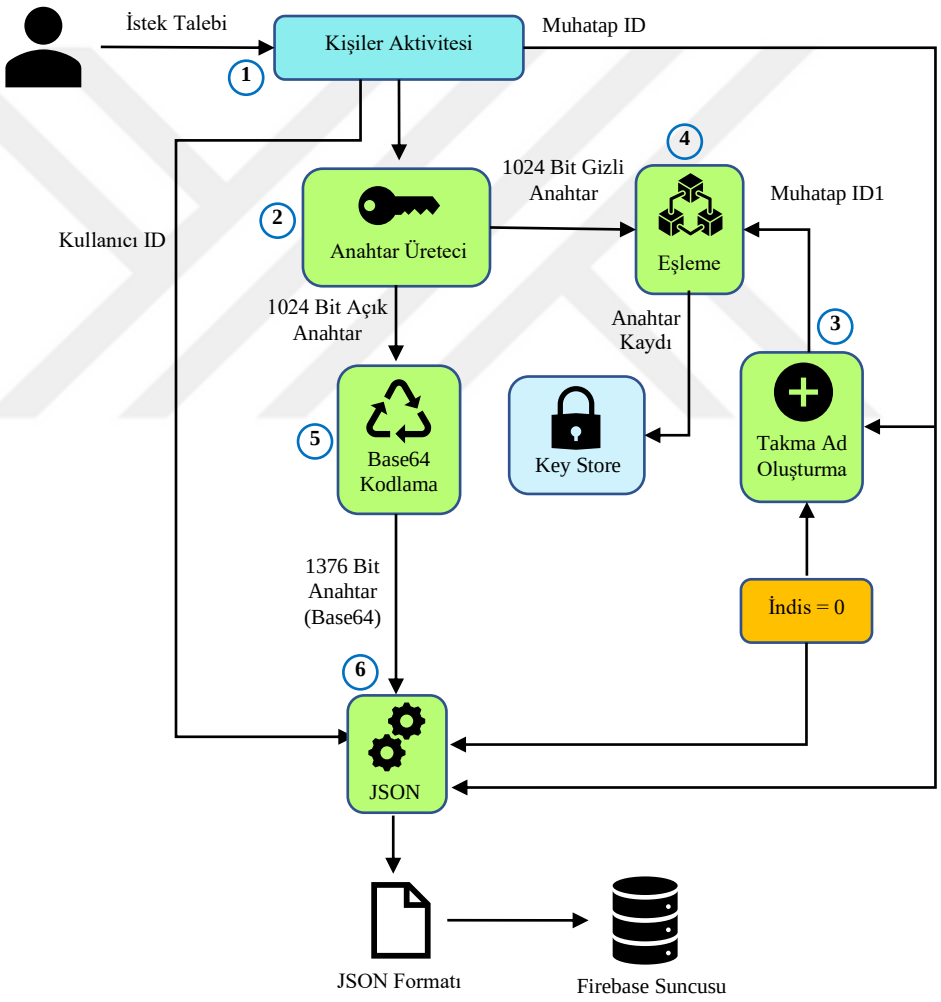


Şekil 4.10. Uygulamanın yaşam döngüsü

Uygulamada mesajlaşma süreci dışında gerçekleştirilen en önemli üç işlem, istek gönderme, gelen istekleri kabul etme ve asimetrik anahtar güncelleme işlemleridir. Çünkü bu işlemler, iletişim sırasında gerekli olan verilerin oluşturulması ve paylaşılmasıyla ilgili süreçleri içerir. İstek gönderme ve kabul etme işlemleri her ne kadar görünürde taraflar arasında mesajlaşmaya başlamaları için mutabakat oluşmasını sağlasa da aynı zamanda iletişimin başlangıcı için gerekli olan asimetrik anahtar paylaşımını da gerçekleştirir. Şekil 4.11' de istek gönderme işleminin nasıl yürütüldüğü görsel olarak ifade edilmiştir. Görüldüğü gibi süreç altı aşamadan meydana gelmektedir. İlk olarak kişiler aktivitesi üzerinden istek talebinde bulunulur. Bu taleple beraber kişiler aktivitesi, Android tarafından sağlanan anahtar üretecini kullanarak açık ve gizli olmak üzere 1024 bit uzunluğunda iki anahtar oluşturur. Bu anahtarlardan açık olanı mesajlaşma sürecinde AES anahtarının RSA algoritmasıyla şifrelenmesi için gönderici tarafından kullanılırken gizli olanı ise alıcı tarafından şifrelenmiş AES anahtarının çözülmesi için kullanılır. Gizli anahtar, üretildiği cihazdan çıkarılamayacak şekilde Android tarafından sunulan keystore sisteminde güvenle muhafaza edilir. keystore sisteminde saklanan her anahtara Kimlik Numarası (Identification Number, ID) niteliğinde bir takma ad verilir. Saklanan anahtarın kullanılması, değiştirilmesi ve silinmesi işlemleri, bu takma ad kullanılarak gerçekleştirilir. Dolayısıyla her anahtar için benzersiz bir takma ad oluşturulmalıdır. Uygulamada bu takma ad, bir ön ek ve bir indisin birleşiminden oluşturulmaktadır. Ön ek, o anahtar kullanılarak iletişim kurulacak kişinin kimliği, yani muhatabın ID'sidir. İndis ise o muhatap için oluşturulan anahtarın sıra numarasıdır. Yani muhatap alınan her kişi için üretilen anahtarlar ayrı ayrı numaralandırılır ve böylece iki muhatap için oluşturulan iki ayrı anahtarın aynı takma adı alma ihtimali ortadan kaldırılmış olur. İstek gönderme ve kabul etme işlemlerinde muhatap alınan kişi için ilk defa bir asimetrik anahtar çifti üretilir. Dolayısıyla bu işlemlerde üretilen anahtarların indisi doğrudan 0 olarak atanır. Daha sonra bu anahtarlar güncellediğinde yeni anahtarlar yeni indis numaralarıyla beraber keystore'a kaydedilir.

Gizli anahtarlar, alıcı tarafından keystore'da saklanırken açık anahtarlar da gönderici tarafından erişilebilmesi için sunucuya iletilir. Ama bu açık anahtarla beraber anahtarı oluşturan kişinin ve kullanacak kişinin ID' leri ile anahtarın indis numarası bilgileri de sunucuya gönderilir. Böylece gönderici taraf bir mesaj iletmek istediğinde alıcı tarafın onun için oluşturduğu açık anahtar sunucu tarafından kolaylıkla göndericiye sağlanır. Gönderilen mesaj alıcı tarafına ulaştığında alıcının bu mesajı çözebilmesi, gönderen kişi için oluşturduğu açık anahtarlardan hangisinin kullanılarak AES

anahtarının şifrelendiğini bilmesiyle mümkün hale gelir. Çünkü bu anahtarlar belirli aralıklarla güncellenmektedir ve farklı mesajlar için farklı açık anahtarlar kullanılmış olabilir. Bundan dolayı gönderici, mesajı hazırlarken bu mesajın içerisine RSA şifrelemesi için kullandığı açık anahtarın indis numarasını da ekler. İndis numarası bilgisi de açık anahtarla beraber sunucuya iletilmiş olduğu için gönderici taraf bu bilgiyi sunucudan kolaylıkla temin eder. Alıcı taraf da şifrelenmiş AES anahtarını çözerken bu şifreleme işleminde hangi RSA açık anahtarının kullanıldığını mesajın içerisindeki indis numarası bilgisi sayesinde anlar. Bu indis numarası ve muhatabın ID' si ile takma adı oluşturup ilgili açık anahtara karşılık gelen gizli anahtarı keystore' dan temin eder.



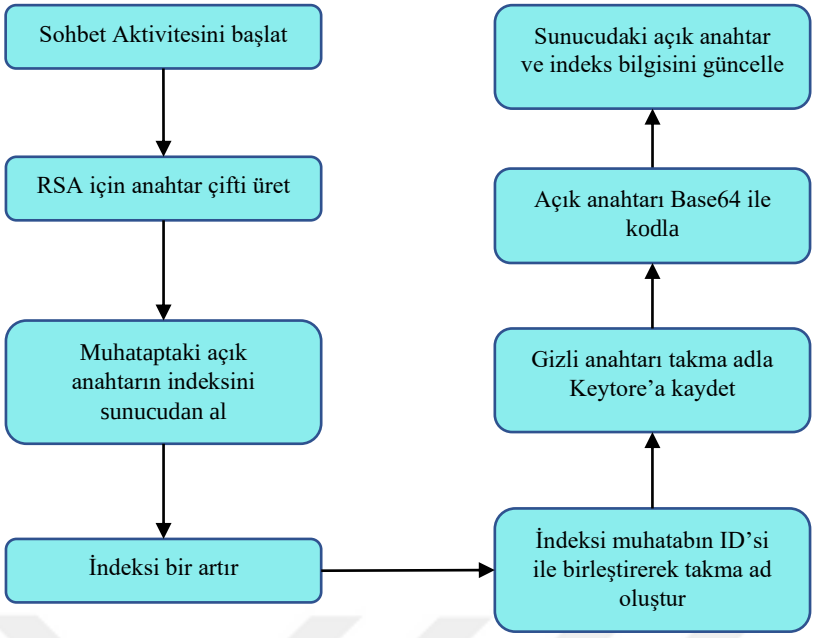
Şekil 4.11. İstek gönderme süreci

Açık anahtar, Firebase sunucusuna gönderilmeden önce Base64 formatında kodlanır. Bunun sebebi Firebase veri tabanının tüm verileri string tipinde saklaması ve buna bağlı olarak depolama alanı ihtiyacının en aza indirgenmesi açısından anahtarın da

string tipinde en kısa şekilde temsil edilmesinin gerekliliğidir. Bunun için açık anahtar RFC 4648’ de tanımlanan en büyük taban olan 64’lük tabanda kodlanır. Böylece 1024 bit uzunluğundaki açık anahtar en az karakter sayısıyla bir string dizgi olarak temsil edilir. Base64, karakter başına 6 bitin temsil edilmesini sağlayan US-ASCII'nin 65 karakterlik bir alt kümesini kullanır. Ekstra 65. karakter "=", özel bir işleme fonksiyonunu belirtmek için kullanılır. Kodlama işlemi için soldan sağa doğru ilerleyerek 3 adet 8 bitlik giriş grubunun birleştirilmesiyle 24 bitlik bir giriş grubu oluşturulur. Bu 24 bit, daha sonra her biri Base64 alfabesinde tek bir karaktere dönüştürülen 4 tane birleştirilmiş 6 bitlik grup olarak işlenir. Her 6 bitlik grup, 64 elemanlı karakter dizisine bir indeks olarak kullanılır. İndeks tarafından işaret edilen karakter, çıktı dizgisine yerleştirilir. Kodlanacak verinin sonunda kalan giriş grubu 24 bitten daha küçükse özel işleme fonksiyonu gerçekleştirilir. Bu durumda verinin son parçası 24 bit olana kadar en sağına 0 değerli bitler eklenir. Kodlanmış verideki dolgu "=" karakteri kullanılarak temsil edilir. Kodlanacak verinin uzunluğu 8’in tam katı olduğundan son giriş grubu 8, 16 veya 24 bit uzunluğunda olacaktır. Eğer son grubun uzunluğu 24 bitse zaten dolgu yapılmasına gerek yoktur ve kodlanmış dizgide "=" karakteri bulunmaz. Son giriş grubunun uzunluğu 8 bitse kodlanmış verinin son biriminde iki tane, 16 bitse bir tane dolgu karakteri bulunur (Josefsson, 2006). Açık anahtar 1024 bit uzunluğunda olduğu için kendisinden 42 tane 24 bitlik giriş grubu elde edilir ve son giriş grubunda ise 16 bit kalır. Bu 16 bitlik giriş grubu da 8 tane 0 değerli bitin eklenmesiyle 24’e tamamlanır ve bu durumda toplam 43 adet 24 bitlik giriş grubu oluşur. Her giriş grubu için 4 karakter olmak üzere kodlanmış çıktı 172 karakterden oluşan bir dizgi olarak elde edilir. Burada 8 bit doldurma yapıldığı için kodlanmış çıktıdaki son karakter, "=" karakteri olur. String bir ifadede Base64 alfabesinde kullanılan karakterlerin her biri 1 bayt yani 8 bit olarak yer kaplar. Bu durumda 172 karakterden oluşan Base64 formatında kodlanmış anahtarın uzunluğu da 1376 bit olur. Açık anahtarın kodlanmasıyla birlikte sunucuya iletilecek veriler için son bir işlem yapılır. Bunlar alan, değer ikilisi şeklinde organize edilerek bir JSON paketi haline getirilir ve Firebase sunucusuna gönderilir. Bunların yanı sıra gelen isteklerin kabul edilmesi de istek gönderme süreciyle aynı şekilde gerçekleşir. Çünkü istek gönderen kişi nasıl bu süreçte kendi açık anahtarını paylaşıyorsa isteği kabul eden kişi de kendi anahtarını paylaşır ve böylece iletişim sürecinin başlaması mümkün hale gelir. Farklı olarak kabul etme talebi kişiler aktivitesi üzerinden değil istekler aktivitesi üzerinden gerçekleşir. İsteği kabul eden kişi de gönderen kişi de olduğu gibi kendi açık ve gizli anahtarlarını oluşturur. Sonrasında gizli anahtarı, isteği gönderen kişinin ID’si ve

0 numaralı indisle bir takma ad oluşturarak Keystore'a kaydeder. Açık anahtarı da yine istek talebinde bulunan kişi gibi Base64 formatına dönüştürüp, kendisinin ve istekte bulunan kişinin ID'si ile birlikte bir JSON paketi haline getirerek sunucuya iletir. İsteği reddedilme durumu ise kabul edilmesi kadar önem arz etmez. Çünkü bu durumda taraflar birbiriyle iletişim kurmayacağı için anahtar paylaşım sürecini devam ettirmeye gerek kalmaz. İstekte bulunan kişi oluşturmuş olduğu anahtar ve bu anahtara ait indis numarası bilgisi de sunucudan silinir.

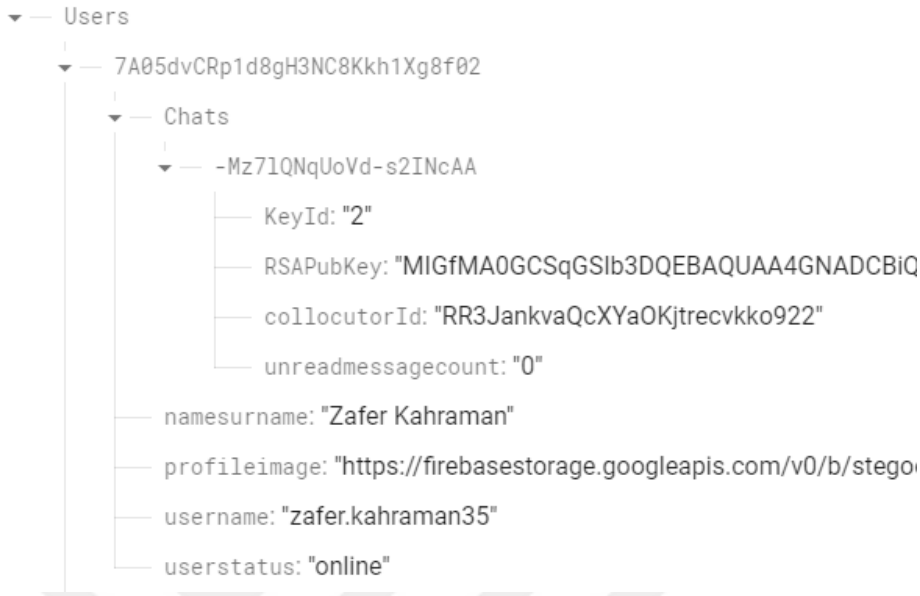
İstek gönderme ve kabul etme süreçlerinde ilk defa oluşturulan RSA anahtarları güvenliğin artırılması için kullanım sıklıklarıyla doğru orantılı olarak güncellenir. Bu işlemin nasıl gerçekleştirildiği Şekil 4.12'deki akış diyagramında gösterilmiştir. Görüldüğü üzere sohbet aktivitesi her başlatıldığında ilgili sohbet için kullanılan RSA anahtar çifti yenilenir. Dolayısıyla sohbet aktivitesinin başlatılmasıyla Android tarafından sunulan anahtar üreteciyle RSA algoritmasında kullanılmak üzere yeni bir asimetrik anahtar çifti üretilir. Bunla beraber gerekli güncellemelerin hem sunucuda hem de keystore'da yapılması gerekir. Anahtar üretimini müteakiben muhataptaki mevcut açık anahtarın indeks numarası sunucundan edinilir. Bu indeks bir artırılarak muhatabın ID' si ile birleştirilir ve böylece yeni anahtar çiftinin takma adı hazır hale gelir. Gizli anahtar bu takma ad kullanılarak keystore'a kaydedilir. Yeni açık anahtar ise önce Base64 formatında kodlanır, sonra da indeks numarasıyla birlikte sunucuya iletilerek sunucudaki eski anahtar bilgileri güncellenir. Fakat keystore'da eski anahtara ait kayıtlar silinmez. Çünkü anahtar güncellemeden önce gönderilen mesajların çözülebilmesi için önceki gizli anahtarlara ihtiyaç vardır. Bunlar önceki indis numaralı takma adlarıyla keystore'da muhafaza edilmeye devam eder. Zaten AES anahtarlarının şifrelenmesinde hangi RSA açık anahtarın kullanıldığını tanımlayan indis numaraları mesajların içerisinde mevcut olduğu için bu açık anahtarlara karşılık gelen gizli anahtarlar da keystore'dan her zaman güvenli bir şekilde temin edilerek şifrelenmiş AES anahtarları çözülür.



Şekil 4.12. RSA anahtarlarının güncellenmesinin işlem adımları

Uygulamada verileri saklamak ve yönetmek için Firebase gerçek zamanlı veritabanı hizmeti kullanılmaktadır. Aynı zamanda bu hizmet sayesinde uygulamada anlık iletişim de sağlanmaktadır. Firebase platformuda veriler JSON formatında tutulur. Bu formatta veriler, bir ağaç yapısına benzer şekilde en başta kök ve onun atında çocuk olarak adlandırılan veri veya veri gruplarının yerleştirilmesiyle yapılandırılır. Şekil 4.13' te uygulamadaki kullanıcı bilgilerinin JSON formatında organize edilmiş hali gösterilmektedir. En baştaki "Users" etiketi altında tüm kullanıcılar, birer çocuk düğüm olarak yer almaktadır. Oluşturulan her bir hesap için özgün bir kimlik değeri atanır ve bu değer "Users" etiketi altına bir çocuk olarak eklenir. Her kullanıcının kimlik etiketi altında da o kullanıcıya ait bilgiler yine birer düğüm olarak yer alır. Şekil 4.13' te kullanıcıya ait kişisel bilgilerin yanı sıra "Chat" kelimesiyle etiketlendirilmiş bir düğüm daha olduğu görülmektedir. Bu etiket altında ise o kullanıcının dahil olduğu sohbetler yer alır. En başta sohbeti tanımlayan kimlik etiketi olmak üzere onun altında okunmamış mesaj sayısı, muhatap olunan kişinin kimlik bilgisi, bu kişinin RSA açık anahtarı ve bu anahtara ait özgün kimlik numarası bilgileri mevcuttur. Bahsedilen açık anahtar 64'lük tabanda bir karakter dizisine dönüştürülerek veritabanında saklanmaktadır. Daha sonra bu anahtar uygulamada kullanılacağı zaman karakter dizisinden tekrar bir anahtar materyaline dönüştürülür. Anahtara ait özgün kimlik numarası ise şifrelemede hangi açık anahtarın kullanıldığını alıcıya bildirmek için gereklidir. Bu kimlik numarası da

mesajlaşma sürecinde taşıyıcı görüntüye gömülerek alıcıya iletilir ve alıcı da buna göre şifre çözmede hangi gizli anahtarı kullanacağını belirler.



Şekil 4.13. Firebase platformunda kullanıcı bilgileri

Firebase veritabanında kullanıcı mesajlarına ait bilgiler "Chats" etiketi, alınan istekler ise "ReceivedRequest" etiketi altında saklanmaktadır. Bu etiketlerin içeriğine ait örnekler Şekil 4.14' te gösterilmiştir. Chats etiketinin bir altındaki düğümler sistemde tanımlanan her sohbet için özgün bir kimlik değerini ifade etmektedir. Bu kimlik etiketleri altındaysa o sohbette iletilen mesajlar yer almaktadır. Her sohbetin olduğu gibi bir sohbetteki her mesajın da onu tanımlayan özgün bir kimlik değeri vardır. Her mesaj düğümünün içerisinde ise o mesajı gönderen kişinin kimlik bilgisi ve mesajın gömülü olduğu stego-görüntünün Tekdüzen Kaynak Bulucu (Uniform Resource Locator, URL) adresi mevcuttur. Görüldüğü gibi veritabanında mesajın içeriğine ait herhangi bir bilgi saklanmamaktadır. Mesaj içeriği şifrelenmiş bir şekilde stego-görüntüye gömülü olarak güvenle muhafaza edilir. Stego-görüntüler ise yine Firebase tarafından sunulan depolama alanında saklanmaktadır.

İstekler ise her bir kullanıcının kimlik etiketi altında istek aldığı kişilerin kimlik etiketlerinin düğüm olarak eklenmesi şeklinde düzenlemiştir. Aynı zamanda istek gönderen kişilerin RSA açık anahtarları ve bu anahtarlara ait özgün kimlik numaraları bu veri grupları içerisinde mevcuttur. Böylece bir kişi kendisine gelen bir isteği kabul etmesi halinde doğrudan bu açık anahtarı şifreleme işlemlerinde kullanarak mesaj göndermeye başlayabilir. Aynı zamanda isteği kabul eden taraf da kendi açık anahtarını istek gönderen

kişiye iletir ve karşılıklı olarak anahtar paylaşım süreci tamamlanmış olur. Paylaşılan bu açık anahtarlar ve bu anahtarlara ait kimlik numaraları her kullanıcının dahil olduğu her sohbet için ayrı ayrı olacak şekilde Firebase veritabanında saklanır ve güvenliğin artırılması için bu anahtarlar belli bir düzene göre güncellenir. Bu düzen, geliştirilen uygulamada kullanıcıların bir sohbete her katılımında RSA anahtarlarının değiştirilmesi olarak ayarlanmıştır.

Şekil 4.14. Firebase platformunda mesaj ve istek bilgileri

Firebase, Android platformuna özel bazı kütüphaneler sunmaktadır. Bu kütüphaneler arasından depolama ve gerçek zamanlı veritabanı hizmeti için gerekli olanları, geliştirilen uygulamada kullanılmıştır. Bu sayede hem oluşturulan stego-görüntüler Firebase sunucusuna kolaylıkla yüklenip depolanabilmekte hem de uygulama üzerinden verilerle anlık etkileşim kurulabilmektedir. Firebase kütüphaneleri kullanılarak uygulama üzerinden veritabanında tutulan veriler için farklı durumlara yönelik olay dinleyicileri tanımlanabilir. Böylece veriler üzerinde ekleme, güncelleme ve silme gibi herhangi bir değişiklik olduğunda bu olay dinleyicileri tarafından tetiklenen metotlar kullanılarak istenilen iş ve işlemler yürütülebilir. Uygulamada kullanıcıların birbirleriyle anlık olarak iletişim kurabilmesi de bu şekilde sağlanmaktadır.

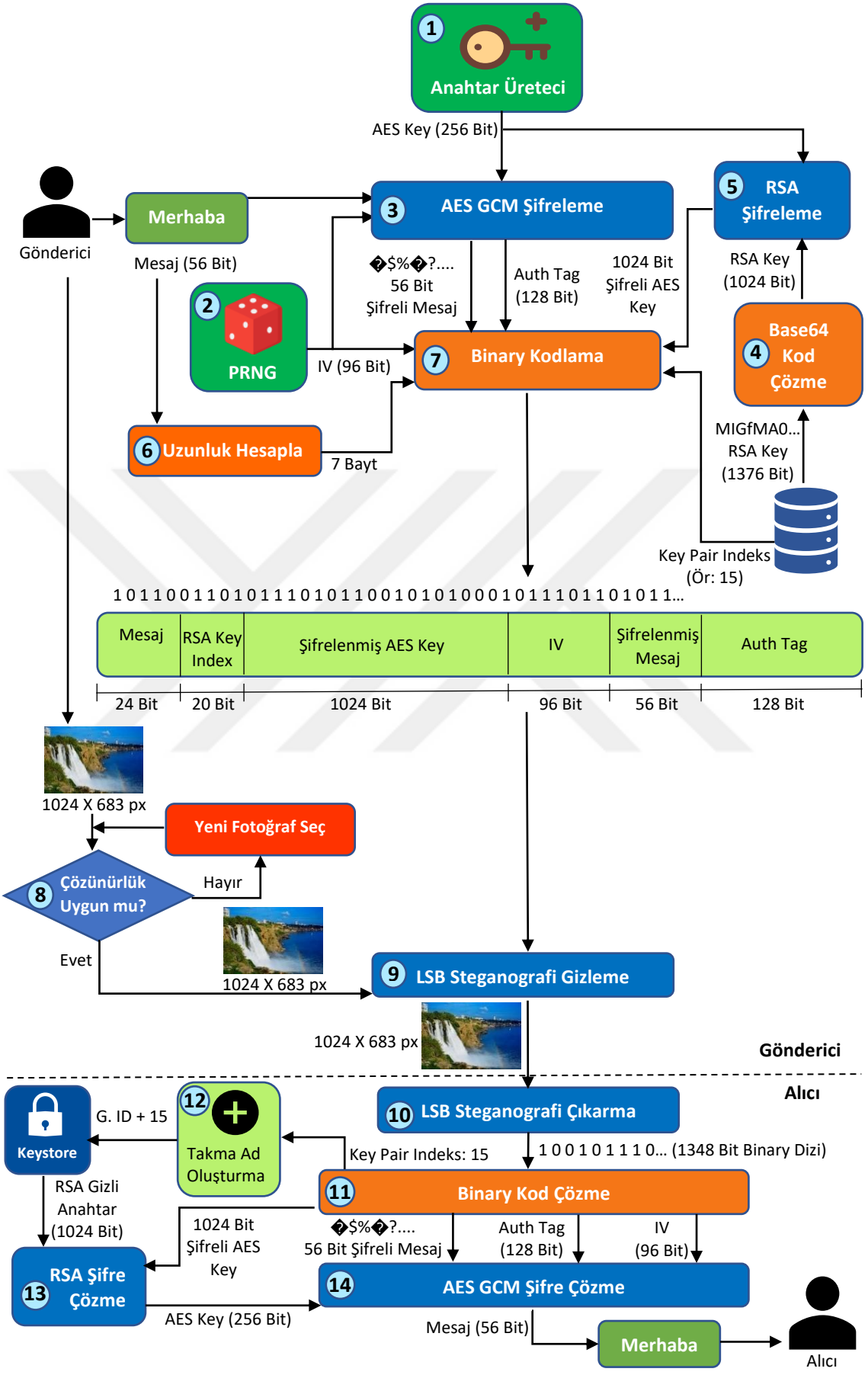
4.3. Örnek Uygulama

Bu bölümde geliştirilen uygulamanın çalışma mekanizması ve bir mesajın göndericisinden alıcısına ulaşana kadar geçirdiği tüm süreçler detaylarıyla birlikte bir örnek üzerinden anlatılmıştır. Şekil 4.15’ de tüm bu süreçler ve bunlara sağlanan

girdilerle bunlardan elde edilen sonuçlar görsel olarak ifade edilmiştir. Görüldüğü üzere süreç 14 işlem adımından meydana gelmektedir. Yapılan ilk temel işlemse 56 bit uzunluğundaki "Merhaba" mesajının AES GCM algoritmasıyla şifrelenmesidir. Ancak bunun öncesinde şifreleme işlemi için gerekli olan veriler üretilmelidir. Bunlar 256 bit uzunluğundaki AES anahtarı ve 96 bit uzunluğundaki IV'dir. IV parametresi, GCM şemasının çalışmasına bağlı olarak CTR moduyla şifreleme yapılması için gereklidir. CTR modunda 96 bitlik IV, 32 bitlik sayaç değeriyle birleştirilerek blok şifreleme fonksiyonlarına girdi olur. Sayaç değerlerinin üretimi zaten dahili olarak CTR modu tarafından gerçekleştirilir. IV ve sayaç parametreleri şifreleme işlemine rastlantısallık katarak aynı düz metin blokları için bile farklı şifrelenmiş bloklar üretilmesini sağlar. Böylece düşük korelasyona sahip şifrelenmiş veriler oluşturalarak güvenlik sağlanır. Geliştirilen uygulamada IV üretimi, Java'nın güvenlik kütüphanesinde bulunan SecureRandom sınıfı kullanılarak gerçekleştirilir. Bu sınıf, kriptografik olarak güçlü bir rastgele sayı üreteci sağlar. Üretilen bu rasgele sayılar, FIPS 140-2, Şifreleme Modülleri için Güvenlik Gereksinimlerinde belirtilen istatistiksel rasgele sayı üreteci testleriyle asgari düzeyde uyumludur. Pek çok SecureRandom uygulaması, Sözde Rastgele Sayı Üreteci (Pseudo Random Number Generator, PRNG) biçimindedir; bu, gerçek bir rastgele tohumdan sözde rastgele bir dizi üretmek için deterministik bir algoritma kullandıkları anlamına gelir (Oracle, 2014). Anahtar üretimi ise Java'nın kriptografi kütüphanesindeki KeyGenerator sınıfı kullanılarak gerçekleştirilir. Bu sınıf, gizli (simetrik) bir anahtar üretecinin fonksiyonunu sağlar. Bu fonksiyona şifreleme algoritması ve istenilen anahtar uzunluğu parametre olarak verilir. Anahtar üreticinin başlatmanın iki farklı yolu mevcuttur. Bunlardan birinde ayrıca SecureRandom argümanı bulunurken, diğeri rastgelelik kaynağı olarak en yüksek öncelikli kurulu sağlayıcının SecureRandom uygulamasını kullanır veya kurulu sağlayıcıların hiçbiri bir SecureRandom uygulaması sağlamıyorsa, sistem tarafından sağlanan bir rastgelelik kaynağı kullanılır (Oracle, 2014).

Simetrik anahtar ve IV' nin üretilmesini müteakiben AES GCM algoritması, 56 bit uzunluğundaki orijinal mesajı şifreleyerek yine 56 bit uzunluğunda şifrelenmiş bir mesaj ve 128 bit uzunluğunda bir kimlik doğrulama etiketi (Auth tag) üretir. Bu etiket, alıcı tarafında elde edilen AES anahtarı ve şifreli mesajın birbiriyle eşleşip eşleşmediğini kontrol etmek için kullanılır. Mesajın şifrelenmesinden sonraki bir temel işlemse mesajı şifrelemek için kullanılan 256 bitlik AES anahtarının RSA algoritmasıyla şifrelenmesidir. Ancak bunun için öncelikle alıcının önceden oluşturup sunucuya ilettiği 1024 bit

uzunluğundaki açık anahtar sunucudan temin edilir. Fakat bu anahtar sunucuda, 64'lük tabanda kodlanmış 1376 bitlik bir dizgi formunda depolanmaktadır. 1024 bitlik RSA açık anahtarı, Base64 formatında 172 karakterle ifade edilmektedir ve Base64 alfabesindeki her karakter string tipinde 1 bayta yani 8 bite tekabül ettiği için uzunluğu 1376 bit olmaktadır. Öncelikli olarak Base64 formatındaki bu dizginin RSA algoritması tarafından kullanılabilecek bir anahtar materyaline dönüştürülmesi gerekir. Bundan dolayı Base64 kod çözme işlemi uygulanarak 1024 bitlik RSA açık anahtarı elde edilir. Bu açık anahtarın elde edilmesine müteakiben RSA algoritması kullanılarak 256 bitlik AES anahtarı şifrelenir ve 1024 bit uzunluğunda şifrelenmiş bir veri haline gelir. RSA algoritmasının çıktısı şifreleme için kullanılan açık anahtarın boyutuna eşittir. Böylece şifreleme süreci tamamlanmış olur. Şifreleme işleminden sonra gönderilecek veri modeli, verilerin binary formda kodlanarak bir araya getirilmesiyle oluşturulur. Bunun için ilk önce orijinal veya şifrelenmiş mesajın uzunluğu hesaplanır. Bu değer verilerin fotoğrafın içerisine gömüldükten sonra tekrar çıkarılabilmesi için gereklidir. Çünkü veri modelinin uzunluğu tam olarak bilinmeli ve ona göre fotoğraftan verilerin çıkarılması işlemi yürütülmelidir. Uzunluk değeri, mesajın bayt cinsinden boyutunu bir tam sayıyla belirtir. Mesaj uzunluğu ifade eden tam sayı 24 bitlik bir binary dizgi olarak kodlanır. Bu tamsayı, 0 ile 16.777.215 arasında değerler alabilir. Bu durum, yaklaşık olarak en fazla 16 MB'a kadar mesaj gönderilebileceğini ifade eder. Zaten bir fotoğrafın içerisine de önerilen yöntemde gerçekleştirilen gizleme işlemiyle en fazla bu boyutlara kadar bir mesaj gizlenebilir. Gerçekleştirilen bu örnek uygulamada mesajın uzunluğu bayt cinsinden 7'dir. Veri modelindeki ikinci parça ise kullanılan RSA anahtarının indis numarasıdır. Böylece alıcı bu indis numarasıyla hangi gizli RSA anahtarını kullanarak şifrelenmiş AES anahtarını çözeceğini tespit eder. Örnekte kullanılan RSA açık anahtarına ait indis numarası sunucudan 15 olarak temin edilmiştir. Veri modelinde indis numarası için 20 bitlik alan ayrılmıştır ve bu da 1.048.576 adet farklı anahtar çiftini indeksleyebilir. Bu açıdan bir kullanıcı katılımcısı olduğu her sohbet için 1.048.576 adet RSA anahtar çifti oluşturabilir. Bu asimetrik anahtarların üretimi ilk defa istek gönderme ve kabul etme süreçlerinde ve sonrasında da her sohbet aktivitesinin başlatılmasında gerçekleştirilir. Veri modelinin üçüncü parçası ise 1024 bit uzunluğundaki şifrelenmiş AES anahtarıdır ve bunu 96 bitlik uzunluğundaki IV, 56 bitlik şifrelenmiş mesaj ve 128 bitlik kimlik doğrulama etiketi takip eder. Böylece bu örnek uygulama için toplam 1348 bit uzunluğu bir veri modeli hazırlanmış olur.



Şekil 4.15. Örnek uygulama şeması

Veri modelinin oluşturulmasından sonra bu verilerin gizlenmesi için kullanıcı tarafından bir fotoğraf seçimi gerçekleştirilir. Şekil 4.15' de görüldüğü üzere bu örnek uygulama için 1024 X 683 px çözünürlüğüne sahip bir fotoğraf seçilmiştir. İlk olarak bu fotoğrafın çözünürlüğünün 1348 bitlik veriyi gizlemek için yeterli olup olmadığı kontrol edilir. Bu amaçla 1348 bitlik veri modelinin gizlenebilmesi gerekli piksel sayısı uzunluk değerinin üçe bölünmesiyle tespit edilir. Çünkü görüntünün her renk katmanında bir bit olmak üzere piksel başına toplam 3 bit gömülür. 1348'in üçe bölümünden bölüm 449 ve kalan 1 olarak elde edilir. Bu durum, veri gizlemek için seçilen görüntüdeki ilk 449 pikselin üç renk katmanın da kullanılacağı ve bunun haricinde artan bir biti saklamak için de ekstra bir pikselin sadece kırmızı katmanı üzerinde işlem yapılacağı anlamına. Yani bu işlem toplam 500 piksele ihtiyaç vardır. Seçilen görüntünün yatay ve dikey çözünürlükleri çarpılarak görüntünün toplam piksel sayısı hesaplanır. Bu örnek için seçilen 1024 X 683 px çözünürlüğe sahip görüntünün toplam piksel sayısı 699.392' dir. Yani örnekteki 1348 bitlik veriyi gizlemek için fazlasıyla yeterlidir. Hatta içerisinde piksel sayısının üç katı kadar veri barındırabilir. Yani bu örnekte seçilen görüntüye toplam 2.098.176 bayt (yaklaşık olarak 2 MB) veri gömülebilir. Çözünürlüğün uygun olduğu tespit edildikten sonra 1348 bitlik veri, görüntünün ilk 500 pikseline yatay yönde ilerleyerek gömülür. Bu işlem için her pikselin her renk katmanındaki en anlamsız yani en sağdaki biti kullanılır. Gizleme işlemi sonucunda içerisinde 1348 veri modelini barındıran ve taşıyıcı görüntüyle aynı çözünürlüğü sahip, yani bu örnek için 1024 X 683 px bir stego görüntü elde edilir. Bu görüntü Firebase sunucusuna iletilir ve Firebase depolama hizmeti tarafından muhafaza edilir. Mesajın kimden alınıp kime gönderildiği bilgilerini içeren meta verisi ve Firebase depolama hizmeti tarafından muhafaza edilen stego görüntü için sağlanan URL bilgisi Firebase gerçek zamanlı veri tabanında saklanır. Bu gerçek zamanlı veri tabanı hizmeti sayesinde sunucuya yüklenen stego görüntü anlık olarak alıcısına ulaştırılır.

Alıcı tarafın ilk yapacağı iş mesaj uzunluğunu tespit etmektir. Bunun için kendisine ulaşan stego görüntüdeki ilk 8 pikselin her renk katmanındaki en anlamsız biti sırasıyla okuyarak bunları birleştirir ve 24 bit binary dizgi olarak kodlanmış mesaj uzunluğunu elde eder. Binary formdaki bu dizgiyi tam sayıya dönüştürür ve buna göre devamında görüntüden çıkarması gereken bit sayısını hesaplar, sonra geri kalan bitleri de bu örnek için stego görüntüdeki diğer 482 pikselden çıkarır. Bu işlemi yaparken geri kalan piksellerinin 481'nin tüm renk katmanlarındaki en anlamsız bitleri okurken 482. pikselin ise sadece kırmızı katmanının en anlamsız bitini okuyarak mesajın gömülmesi sırasında

artan o bir biti de çıkardığı veriye dahil etmiş olur. Sonuç olarak toplam 1348 bitlik bir binary dizgi stego görüntüden çıkarılmış olur. Bu 1348 bitin ilk 24 biti mesaj uzunluğunu temsil ettiğinden sadece mesajın çıkarılması aşamasında gereklidir ve sürecin devamında kullanılmaz. Bu yüzden sonraki işlemler 1348 bitlik verinin geri kalan 1324 bitlik kısmı kullanılarak gerçekleştirilir. Alıcının 56 bitlik şifrelenmiş mesajı çözebilmesi için öncesinde RSA ile şifrelenmiş olan AES anahtarını çözmesi gerekir. 1024 bit uzunluğundaki şifrelenmiş AES anahtarı verisi, 1348 bitten oluşan binary dizgide 45. karakterden başlayıp 1068. karakterle son bulan kısımda yer alır. Binary formundaki bu dizgi, RSA algoritması tarafından işlenebilmesi için bayt formuna getirilir. RSA ile şifre çözme işleminin gerçekleşmesi için AES anahtarının şifrelenmesinde kullanılmış RSA açık anahtarına karşılık gelen gizli anahtar, keystore' dan temin edilmelidir. Bunun temin edilmesi için de keystore sistemine ilgili anahtarın takma adı bildirilmelidir. Takma adın ikinci parçası olan indis numarası, zaten 1348 bitlik binary formundaki dizginin 25. karakteriyle başlayıp 44. karakteriyle biten 20 bitlik kısmından oluşmaktadır. Binary formdaki bu 20 bitlik dizginin desimale dönüşümüyle beraber indis numarasının tamsayı karşılığı da elde edilir. Örnekte bu değer 15 olarak elde edilmiştir. Takma adın ilk parçası ise mesajı gönderen kişinin yani muhatabın ID' sidir. Muhatabın ID'si zaten Şekil 4.10' da gösterildiği ve daha önce bahsedildiği gibi sohbet listesi aktivitesi tarafından sohbet aktivitesine iletilmiştir ve böylece temin edilen muhatap ID, indis numarasıyla birleştirilerek lazım olan gizli RSA anahtarının takma adı oluşturulur. Bu sayede 1024 bit uzunluğundaki gizli RSA anahtarı, keystore sisteminden temin edilir ve 1024 bitlik şifreli AES anahtarı verisinin çözülmesi için RSA algoritmasında kullanılır. Böylece 56 bitlik şifrelenmiş mesajın çözülmesi için gerekli olan simetrik AES anahtarı elde edilmiş olur. Yalnız mesajın şifresinin çözülebilmesi için 256 bitlik anahtarla beraber 128 bitlik kimlik doğrulama etiketi ve 96 bitlik IV de AES GCM algoritmasına girdi olarak sağlanmalıdır. IV, stego görüntüden çıkarılan binary dizginin 1069. karakteriyle başlayıp 1164. karakteriyle son bulan kısımdan oluşur. Ancak bu haliyle bir binary dizgi formundadır ve AES GCM algoritması tarafından işlenebilmesi için bayt formuna dönüştürülür. 56 bitlik şifrelenmiş mesaj ve 128 bitlik kimlik doğrulama etiketi zaten Java'nın kriptografi kütüphanesi tarafından sağlanan şifreleme fonksiyonlarına birleşik bir bayt dizisi şeklinde verilir. 1348 bitlik binary dizginin 1165. karakteriyle başlayıp 1348. karakteriyle biten son 184 bitlik kısmı şifreli metnin ve kimlik doğrulama etiketinin birleşimini içerir ve bu birleşim doğrudan bayt formuna dönüştürülerek AES GCM şifre çözme algoritmasına beslenir. AES GCM şifre çözme algoritması, şifreleme algoritmasında olduğu gibi şifreli

metni kullanarak 128 bit uzunluğundaki kimlik doğrulama etiketini üretir. Sonra, ürettiği bu etiketle kendisine girdi olarak sağlanan etiketi karşılaştırır ve bunlar birbirleriyle eşleşirse şifreleme işlemi onaylanmış olur. Yani şifre çözme algoritmasına sağlanan şifreli metnin yine bu algoritmaya sağlanan anahtar kullanılarak şifrelendiği doğrulanır. Aksi halde şifre çözme algoritmasına sağlanan anahtar, şifreleme için kullanılan anahtardan farklı bile olsa yine de şifreli veri çözülecektir. Ancak elde edilen orijinal verinin doğru olup olmadığı tespit edilemeyecektir. GCM şeması gerçekleştirmiş olduğu bu uygulamayla AEAD'ı sağlar. Eğer bahsedilen etiketler uzunluk veya değer açısından eşleşmezse şifreleme işlemi doğrulanamaz ve şifre çözme işleminin çıktısı olarak düz metin yerine özel bir başarısızlık sembolü döndürülür. Gerçekleştirilen bu örnek uygulamada etiketlerin eşleştiğinin doğrulanmasıyla birlikte 56 bitlik şifreli mesaj, AES GCM algoritması kullanılarak çözülür ve işlem sonucunda çıktı olarak elde edilen 56 bit uzunluğundaki "Merhaba" mesajı alıcıya gösterilir.

5. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Bu bölümde yapılan çalışma ve çalışmaya etki eden parametreler çeşitli kriterlere göre değerlendirilmiş ve elde edilen sonuçlar incelenmiştir. Buna yönelik olarak çalışma kapsamında yararlanılan kriptoloji algoritmalarının hangi kriterlere göre belirlendiği ve kullanılan steganografi yönteminin farklı durumlar altında ne derece başarı gösterdiği gibi çeşitli bulgulardan bahsedilmiştir. Bu amaçla ilk olarak geliştirilen uygulamada mesajların gizlenmesiyle oluşturulan stego-görüntüler, görüntü steganografi tekniklerinin değerlendirilmesinde kullanılan iki farklı ölçüt üzerinden incelenmiştir. Bunlardan ilki taşıyıcı görüntü ile stego görüntü arasındaki farkı ölçmek için kullanılan MSE' dir. MSE, orijinal ve stego görüntüler arasındaki kümülatif hataya eşittir (Cihangir, 2020). Bu değer iki görüntü birbirine ne kadar benzerse o kadar düşük, birbirinden ne kadar farklıysa o kadar yüksek çıkar. Dolayısıyla bir steganografi işleminin kalitesini doğrulamak için MSE değeri mümkün olduğunca düşük olmalıdır. MSE değeri, aşağıda ifade edilen eşitlik (5.1) ile hesaplanmaktadır. Bu eşitlikte yer alan I_1 ve I_2 ifadeleri sırasıyla taşıyıcı görüntü ve stego görüntüyü, *M* ve *N* ifadeleri ise sırasıyla kullanılan görüntünün yatay ve dikey çözünürlüğünü temsil etmektedir.

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} . \sum_{j=1}^{N} [I_1(i, j) - I_2(i, j)]^2 \quad (5.1)$$

Kullanılan bir diğer ölçüt ise PSNR' dir. PSNR, gizli mesajın görünmezliğini ölçer. Veri gizlenmiş stego görüntünün orijinal haline göre kalitesindeki değişimi ölçmek için kullanılır. PSNR aslında ölçülen MSE değerini tüm görüntü aralığına ölçekler (Goel ve ark., 2013). Bu yüzden PSNR görüntülerdeki bozulmaları değerlendirmek için iyi bir kriterdir. Dolayısıyla veri gömülen stego-görüntülerde meydana gelen bozulmaları ifade etmek için bu ölçüt kullanılmıştır. Görüntüdeki bozulma ne kadar fazlaysa PSNR değeri o kadar düşük, ne kadar azsa PSNR değeri o kadar yüksek olmaktadır. Bu açıdan bir steganografi işleminin başarısı PSNR değeriyle doğru orantılıdır. PSNR değeri eşitlik (5.2) ile hesaplanır. Bu eşitlikte yer alan *MAX* ifadesi bir pikselin alabileceği en yüksek değeri temsil eder ve bu değer 8 bitlik görüntüler için 255'tir.

$$PSNR = 10 \log_{10} \frac{MAX^2}{MSE} \quad (5.2)$$

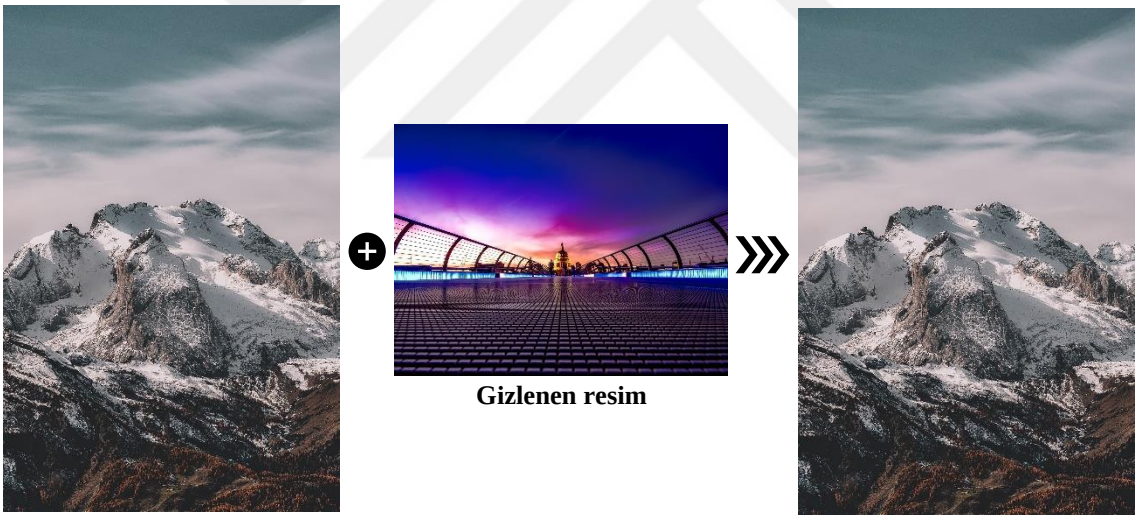
Çizelge 5.1’ de çözünürlükleri farklı dört taşıyıcı görüntüye belirtilen boyutlarda veri gömülmesiyle elde edilen stego görüntülerin MSE ve PSNR değerleri verilmiştir. Bir steganografi işlemi ne kadar başarılıysa hesaplanan MSE değeri o kadar düşük ve bunun aksine PSNR değeri de o kadar yüksektir. Çizelge 5.1’ deki değerlerden de görüntüye yüklenen veri boyutu artırıldığında MSE değerinin arttığı, bunun da PSNR değerini ters şekilde etkilediği görülmektedir. PSNR, genellikle dB cinsinden logaritmik bir ölçekte ifade edilir ve 30 dB’ in altındaki PSNR değerleri oldukça düşük bir kaliteye işaret eder. Başka bir deyişle görüntüdeki veri gizlenmesinden kaynaklı bozulma anlaşılır hale gelir. Yüksek kaliteli bir stego görüntünün PSNR değeri 40 dB ve üzeri olmalıdır (Hamid ve ark., 2012). Bu çalışmada elde edilen sonuçlar veri gömme işleminin daha az bozulma ve daha yüksek PSNR değeri sağladığını göstermektedir.

Kullanılan taşıyıcı görüntünün çözünürlüğü de steganografi işleminin başarımını önemli ölçüde etkilemektedir. Yüksek çözünürlüklü görüntüler yüksek veri saklama kapasitesine sahiptir. Çizelge 5.1’ de görüldüğü üzere aynı boyuttaki verilerin daha yüksek çözünürlüklü görüntülere gizlenmesi daha başarılı sonuçlar elde edilmesini sağlamıştır. Fakat 800x600 px gibi düşük çözünürlüklü bir görüntüde bile elde edilen PSNR değerinin oldukça yeterli olduğu görülmektedir.

Çizelge 5.1. Farklı görüntülere ait MSE ve PSNR değerlerinin karşılaştırması

Görüntüler	Gömülen veri boyutu	MSE	PSNR (dB)
 2160 x 1440 px	1 KB	0.0003	82.39
	100 KB	0.0370	62.45
	1 MB	0.3795	52.34
 1600 x 900 px	1 KB	0.0009	78.27
	100 KB	0.0949	58.36
	500 KB	0.4740	51.37
 1024 x 768 px	1 KB	0.0018	75.59
	100 KB	0.1750	55.70
	200 KB	0.3486	52.71
 800 x 600 px	1 KB	0.0029	73.48
	100 KB	0.2834	53.61
	150 KB	0.4252	51.84

Yapılan çalışmada güvenliğin sağlanabilmesi için taşıyıcı görüntüye gömülen şifrelenmiş mesajların algılanamaz olması önemlidir. Çizelge 5.1' deki değerler bu koşulun sağlandığını sayısal olarak ispatlar niteliktedir. Şekil 5.1' de ise bu durumun görsel olarak bir ispatı sunulmuştur. Bu şekilde, 1280 X 920 px çözünürlüğünde bir taşıyıcı görüntüye şekilde "gizlenen resim" olarak ifade edilen 1920 X 1440 px çözünürlüğündeki şifrelenmiş başka bir görüntünün gömülmesiyle elde edilen stego-görüntü gösterilmiştir. Görüldüğü gibi taşıyıcı görüntü ile oluşan stego görüntü arasında gözle algılanabilir hiçbir fark yoktur. Halbuki elde edilen stego görüntü kendi çözünürlüğünden daha yüksek çözünürlüğe sahip başka bir görüntüyü içerisinde barındırmaktadır ve hatta çok daha fazlasını da barındırabilir. Çünkü dijital resimler sıkıştırılmış formatta depolanmaktadır ve aslında hafıza kapladıkları bellek miktarlarına göre çok daha fazla bilgi saklarlar. Dolayısıyla bu bilgiler üzerinde değişiklik yapılarak kendi sıkıştırılmış boyutlarından çok daha fazla büyüklükteki veriler rahatlıkla bu görüntülerin içerisine gömülebilir.



Şekil 5.1. Taşıyıcı ve stego görüntü örneği

Şekil 5.1' de yer alan taşıyıcı görüntü toplam 2.457.600 adet pikselden oluşmaktadır. Dolayısıyla piksel başına 3 bit olmak üzere toplam 7.372.800 bit yani tam olarak 900 Kilobayt (KB) veri bu görüntü içerisine gömülebilir. Yine aynı şekilde "gizlenen resim" olarak ifade edilen görsel yaklaşık olarak 470 KB boyutundadır ve aslında bu boyutlarda bir resim daha aynı taşıyıcı görüntüye gizlenebilir. Bu gizleme işlemi sadece en az anlamlı bitler üzerinde değişim yapılarak gerçekleştirileceğinden yine

taşıyıcı görüntü üzerinde gözle algılanabilir hiçbir farklılık oluşmaz. Zaten Matlab ortamında yapılan ölçümlerde Şekil 5.1'de yer alan taşıyıcı ve stego görüntüye ait MSE değerinin 0,2611 ve PSNR değerinin ise 53,56 dB olduğu tespit edilmiştir. Bu değerlerden de uygulanan steganografi işleminin oldukça başarılı olduğu anlaşılmaktadır.

Yapılan çalışmada veriler steganografiyle bir görüntünün içerisine gömülmeden önce bir kriptoloji algoritması kullanılarak şifrelenmektedir. Çalışma kapsamında anlık iletişim sağlayan bir uygulama geliştirildiği için kullanılacak kriptoloji algoritmasının güvenli olmasının yanı sıra hızlı şifreleme yapması da gerekmektedir. Bu açıdan Java'nın kriptografi kütüphanesindeki mevcut simetrik şifreleme algoritmaları, Android platformu üzerinde test edilerek işlem zamanı açısından değerlendirilmiştir. Android, Java veya Kotlin dilindeki yerel kod performanslarını ve kaynak tüketimlerini hızlı bir şekilde ölçme imkanı sağlayan Jetpack Microbenchmark kütüphanesini sunmaktadır. Bu kütüphane, veri dönüştürme/işleme gibi birçok kez yürütülen Merkezi İşlem Birimi (Central Process Unit, CPU) çalışmalarını incelemek için oldukça kullanışlıdır. (Google, 2022). Bu yüzden şifreleme algoritmalarının performanslarını değerlendirmek için Microbencmark kütüphanesi kullanılmıştır. Test işlemleri, 1.6 Gigahertz (GHz) 8 çekirdekli işlemci ve 3 Gigabayt (GB) Rastgele Erişimli Hafıza (Random Access Memory, RAM)' ya sahip General Mobile GM21 model Android cihazı üzerinde gerçekleştirilmiştir. Yapılan test işleminde her bir algoritmayla 10 MB boyutunda veri şifrelenmiştir. Bu şifreleme işlemlerinde tüketilen zaman ve buna bağlı olarak hesaplanan şifreleme hızları Çizelge 5.2' de verilmiştir. Görüldüğü gibi AES algoritması diğerlerine göre daha başarılı olmuştur. Çünkü AES, yaygın bir simetrik şifreleme algoritması olduğundan donanım yongaları, AES'in performansını iyileştirmek için AES'e özel komut setleri içermektedir (Kao ve ark., 2021). AES algoritması en hızlı CTR kipinde çalışmaktadır. Ancak bu kipte kimliği doğrulanmış şifreleme yapılamaz. Buna karşılık GCM kipinde ise hem CTR modu kullanılarak şifreleme yapılır hem GMAC metodu kullanılarak yapılan bu şifreleme işlemi doğrulanır. Çizelge 5.2' den de görüldüğü gibi CTR ve GCM kipleri arasında işlem zamanı açısında çok büyük bir fark yoktur. Ama GCM kipi CTR' ye göre daha güvenli bir şifreleme sunmaktadır. Bu yüzden yapılan çalışmada mesajların şifrelenmesinde kullanılabilecek en ideal seçenek AES GCM algoritmasıdır. GCM kipinin kullanılmasına bağlı olarak kimliği doğrulanmış şifreleme sağlamak için oluşturulan şifreli veriye aynı zamanda 128 bit uzunluğunda bir kimlik doğrulama etiketi eklenmektedir. Ancak bu düşük bir boyut olduğundan mesajlaşma sürecini işlem zamanı açısından olumsuz olarak etkilemez.

Çizelge 5.2. Kriptoloji algoritmalarının şifreleme hızları

Algoritma	Mod	İşlem zamanı (ms)	Şifreleme Hızı (MB/sn)
AES 256 Bit	ECB	32,13	311,24
	CBC	22,87	437,25
	CTR	20,54	486,85
	CFB	1210,20	8,26
	OFB	1167,26	8,57
	GCM	37,09	269,61
Blowfish 256 Bit	ECB	463,82	21,56
	CBC	689,46	14,50
	CTR	880,54	11,36
	CFB	1057,43	9,46
	OFB	1011,78	9,88
DES 64 Bit	ECB	1035,71	9,66
	CBC	1265,15	7,90
	CTR	1457,13	6,86
	CFB	1626,65	6,15
	OFB	1580,90	6,33
RC4 256 Bit	ECB	139,78	71,54
ChaCha20	None	39,81	251,19
	Poly1305	88,59	112,88

Bu zamana kadar olan kısımda çalışmada kullanılan steganografi yönteminin başarımı değerlendirilmiş ve çalışmada kullanılabilecek şifreleme algoritmaları işlem zamanı açısından karşılaştırılmıştır. Ama şifreleme algoritmalarının hızlı olması kadar güvenli olması da önemlidir. Dolayısıyla şimdide bu algoritmalar şifreleme başarımları açısından karşılaştırılacak ve değerlendirilecektir. Bu amaçla AES, RC4 ve ChaCha20 algoritmaları farklı şifreleme modlarıyla beraber incelenmiştir. DES ve Blowfish ise işlem zamanı açısından çok yavaş olduğundan anlık iletişimde kullanmak için uygun değildir ve bu yüzden bu algoritmalar yapılan inceleme işleminin dışında tutulmuştur.

Bahsedilen kriptoloji algoritmalarının başarımlarını değerlendirmek için bu algoritmalar kullanılarak görüntü şifreleme işlemleri uygulanmıştır. Görüntü şifreleme uygulamalarında dijital resimler sıkıştırılmış formatta ve bir bayt katarı formunda işleme alınmaz. Buradaki uygulama, görüntünün içeriğinin yani piksel değerlerinin şifrelenmesi ve orijinal piksel değerleriyle şifrelenmiş hallerinin yer değiştirilerek yeni bir görüntünün elde edilmesidir. Bu sayede şifrelenmiş görüntüyle orijinal görüntü karşılaştırılarak kullanılan algoritma ve çalışma kiplerinin başarımlarıyla alakalı çıkarımlarda bulunulabilir. Görüntü şifreleme uygulamalarını değerlendirmek için kullanılan beş yaygın ölçüt bulunmaktadır. Bunlar; Değişen Piksel Sayısı Oranı (Number of Pixel

Change Rate, NPCR), Birleşik Ortalama Yoğunluk Değişimi (Unified Average Change Intensity, UACI), MSE, PSNR ve korelasyon sabitidir.

Shannon (1949), kriptolojide istatiksel analize dayalı güçlü saldırıları bertaraf etmek için karışıklık ve yayılma anlamlarına gelen konfüzyon ve difüzyon kavramlarını öne sürmüştür. NPCR, UACI, MSE ve PSNR kriterleri bir kripto sistemin difüzyon karakteristiğini ölçmek için kullanılır. Kripto sistemler için difüzyon kavramı düz metindeki bir bit değişikliğin şifreli metni ne kadar etkilediğini ifade eder. İyi bir kripto sistem iyi bir difüzyon sağlamalıdır, yani düz metindeki bir bitlik değişim, şifreli metni tamamen öngörülemeyen bir şekilde değiştirmelidir. Kripto sistemlerin bu özelliği çığ etkisi olarak bilinir. Çığ etkisi, tüm kriptografi algoritmaları için olması istenen bir özelliktir. Orijinal veride gerçekleşen bir bit değişimi, şifreli verinin en az yarısını değiştiriyorsa sert bir çığ etkisi meydana gelmiş demektir (Ahmad ve Ahmed, 2012). Görüntü şifreleme uygulamaları için difüzyon karakteristiği, şifreli görüntünün pikselleriyle orijinal görüntünün pikselleri arasındaki bağıntının çok karmaşık olması gerektiği anlamına gelir. Bu açıdan bir görüntü şifreleme uygulamasının başarılı olarak değerlendirilebilmesi için elde edilen şifreli görüntünün orijinalinden olabildiğince farklı olması gerekir. NPCR ve UACI kriterleri de bu farkın bir ölçüsünü ifade eder ve şifreleme algoritmalarının diferansiyel ataklara karşı dayanıklılığını tespit etmek için yaygın olarak kullanılır (Wu ve ark., 2011). NPCR, şifreleme işlemi sonucunda değişen piksel sayını ele alırken, UACI ise pikseller arasındaki ortalama değer farkına odaklanır. Şifreleme işleminin başarısı bu değerlerle doğru orantılıdır. Bu değeler ne kadar yüksekse şifreli ve orijinal görüntüler birbirinden o kadar bağımsız, yapılan şifreleme işlemi de o kadar başarılı demektir. NPCR ve UACI ölçütlerinin hesaplanışı sırasıyla eşitlik (5.3) ve (5.4)' te gösterilmiştir. Eşitlik (5.3)' teki $D(i,j)$ fonksiyonu şifreleme işleminden sonra o indeksteki pikselin değişip değişmediğini ifade eder ve eğer değişim olmuşsa 1, olmamışsa 0 değerini alır. Eşitlik (5.4)' teki I_o ifadesi orijinal görüntüyü, I_{enc} ifadesi ise şifrelenmiş görüntüyü temsil etmektedir.

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100\% \tag{5.3}$$

$$UACI = \frac{1}{W \times H} \cdot \left[\sum_{i,j} \frac{|I_o(i,j) - I_{enc}(i,j)|}{255} \right] \times 100\% \tag{5.4}$$

AES, ChaCha20 ve RC4 algoritmaları, farklı çalışma kipleriyle beraber kullanılarak birisi literatürdeki çalışmalarda görüntü işleme ve şifreleme uygulamalarında

yaygın olarak kullanılan “peppers” görseli olmak üzere 512 X 512 piksellik iki farklı fotoğraf üzerinde görüntü şifreleme işlemleri uygulanmıştır. Yapılan işlemler sonucu elde edilen şifreli görüntüler ve orijinalleri karşılaştırılarak her bir işlem için NPCR, UACI, MSE ve PSNR değerleri Matlab ortamında teker teker hesaplanmış ve bunların ortalama değerleri Çizelge 5.3’te gösterilmiştir. Görüldüğü gibi algoritmaların genel olarak başarılı olduğu ve birbirlerine yakın değerler elde edildiği gözlemlenmektedir. Aynı zamanda her algoritma belli ölçütlerde birbirine göre daha iyi sonuçlar vermiştir. Genel olarak değerlendirdiğimizde AES GCM algoritmasının NPCR açısından AES ECB’ye göre, UACI açısından ChaCha20 Poly1305 ve RC4’e göre, MSE ve PSNR açısından da AES ECB ve CBC’ ye göre daha iyi sonuçlar vermesinden dolayı nispeten daha başarılı olduğunu söyleyebiliriz.

Çizelge 5.3. Farklı algoritmaların görüntü şifreleme açısından karşılaştırılması

Algoritma	**NPCR**	**UACI**	**MSE**	**PSNR**
AES ECB	%99,60	%32,22	$1{,}0105 \times 10^4$	8,0855
AES CBC	%99,61	%32,21	$1{,}0101 \times 10^4$	8,0871
AES GCM	%99,61	%32,21	$1{,}0102 \times 10^4$	8,0870
ChaCha20 Poly1305	%99,61	%32,20	$1{,}0094 \times 10^4$	8,0903
RC4 ECB	%99,61	%32,20	$1{,}0093 \times 10^4$	8,0908

Bir görüntüye şifreleme algoritması uyguladığımızda, görüntünün piksel değerleri değişir. İyi bir şifreleme algoritması bu değişiklikleri düzensiz bir şekilde yapmalı ve aynı zamanda orijinal ve şifreli görüntüler arasındaki piksel değerlerindeki farkı en üst düzeye çıkarmalıdır. Ayrıca iyi şifrelenmiş bir görüntü, orijinal görüntünün özelliklerini açığa çıkarmayacak şekilde tamamen rastgele paternlerden oluşmalıdır ve şifrelenmiş görüntü orijinal görüntüden bağımsız olmalıdır (Elkamchouchi ve Makar, 2005). Bu bağımsızlığın derecesini ifade eden ve şifreleme algoritmalarının başarımlarını değerlendirmek için kullanılan bir diğer ölçütse korelasyon sabitidir. Bu yüzden kriptoloji algoritmalarının şifreleme kalitesini ölçmek için geliştirilen uygulama içerisinde şifrelenen görüntülerin korelasyon katsayıları hesaplanmıştır. Korelasyon, iki değişken arasındaki ilişkiyi belirler. Başka bir deyişle korelasyon, iki değişken arasındaki benzerlik derecesini hesaplayan bir ölçüdür (El-Fishawy ve Abu Zaid, 2007). Korelasyon sabiti herhangi bir kripto sistemin şifreleme kalitesini değerlendirmek için faydalı bir kriterdir. Eğer bir şifreleme algoritması, orijinal görüntünün tüm özelliklerini gizleyebiliyorsa ve elde edilen şifreli görüntü tamamen rastgele ve son derece ilişiksizse o şifreleme

algoritmasının iyi olduğu söylenebilir. Bu özelliğin sağlanabilmesi içinse elde edilen şifreli görüntüdeki komşu piksellerin birbirinden tamamen bağımsız ve ilişiksiz olması gerekir. Eşitlik (5.5)' teki r_{xy} ifadesi korelasyon katsayısını temsil etmektedir ve görüldüğü gibi bu katsayının hesaplanmasında eşitlik (5.6)' da $cov(x,y)$ ile ifade edilen kovaryans değerinden yararlanılmaktadır (Kamali ve ark., 2010). Kovaryansyonun ve korelasyonun hesaplanması için gerekli diğer ifadelere ait formüller eşitlik (5.7) ve (5.8)' de gösterilmiştir. Eşitliklerde yer alan x ve y değişkenleri görüntüdeki iki komşu pikselin renk yoğunluğu değerlerini, N ifadesi ise korelasyon hesabının yapılmasında kullanılmak üzere seçilen piksel çifti sayını temsil etmektedir. Bu çalışmada korelasyon hesabı yapılırken daha kesin sonuçlar elde edilmesi için görüntüden belli sayıda rastgele komşu piksel çiftleri seçmek yerine görüntüdeki tüm pikseller birden ele alınmıştır.

$$r_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{5.5}$$

$$cov(x,y) = E(x - E(x))(y - E(y)) \tag{5.6}$$

$$E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{5.7}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2 \tag{5.8}$$

Yapılan tez çalışmasında geliştirilen uygulamada, farklı algoritma ve farklı çalışma kiplerinin kullanılmasıyla elde edilen şifreli görüntülerin yatay, dikey ve çapraz yöndeki komşu piksellerine ait korelasyon katsayıları Matlab ortamında hesaplanmış ve sonuçlar Çizelge 5.4' te gösterilmiştir. Bu değerlerin elde edilmesi için şifrelenmiş görüntülerden her bir yön için ayrı olmak üzere tüm komşu pikseller işleme alınmış ve bunların korelasyon katsayıları hesaplanmıştır. Korelasyon sabiti -1 ile 1 arasında değerler almaktadır (Elkamchouchi ve Makar, 2005). Eğer komşu pikseller birbirinin aynısıysa mükemmel bir korelasyon içindedirler ve bu durumda korelasyon katsayısı 1 değerini alır. Genelde görüntülerde komşu pikseller birbirinin aynısı olmasa bile birbirine çok yakın olduğu için korelasyon katsayısı 1'e çok yakın bir değer çıkar. Örneğin Çizelge 5.4' teki değerleri elde etmek için kullanılan orijinal görüntünün tüm yönlerdeki ortalama korelasyon değeri 0,9664 olarak hesaplanmıştır. Güvenli bir şifreleme algoritmasının pikseller arasındaki korelasyonu elimine etmesi gerekir (Zhou ve ark., 2020). Eğer korelasyon katsayısı -1 ise komşu pikseller birbirinin tersi yani negatifidir ve dolayısıyla ters yönde kuvvetli bir korelasyon içerisindedirler. Korelasyon katsayısının 0 olması

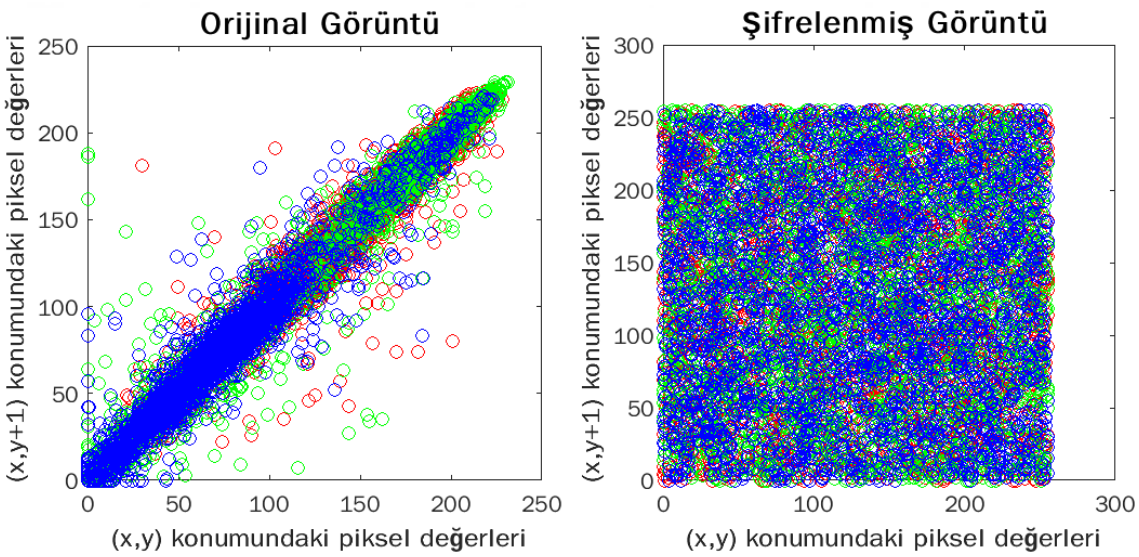
durumunda ise komşu pikseller tamamen ilişiksizdir ve birbirlerine hiçbir bağımlılığı yoktur. Bu yüzden başarıyla şifrelenmiş bir görüntünün tüm yönlerdeki komşu piksellerinin korelasyon katsayısı olabildiğince 0'a yakın olmalıdır. Çizelge 5.4' te görüldüğü üzere tüm algoritmalarla şifrelenen görüntülerin korelasyon katsayılarının 0'a yakın olduğu ama bazı algoritmaların diğerlerine göre daha iyi sonuçlar verdiği gözlemlenmektedir. Elde edilen değerlerin pozitif veya negatif olması şifreleme başarımı açısından önemli değildir, bu durum sadece komşu pikseller arasındaki ilişkinin yönünü ifade eder. Bizim için önemli olan bu değerlerin mutlak büyüklükleridir. Çünkü korelasyon değeri mutlak olarak ne kadar büyükse pikseller arasındaki ilişkinin de o kadar kuvvetli olduğu söylenebilir. Çizelge 5.2' de yer alan verilere göre her renk katmanındaki ve her yöndeki korelasyon katsayılarının mutlak değerlerini ortalama olarak ele aldığımızda AES CBC, GCM ve ChaCha20 Poly1305 algoritmaları için sırasıyla 0,0014, 0,0015 ve 0,0015 değerleri hesaplanmıştır. Dolayısıyla bu algoritmalarla şifrelenen görüntülerin neredeyse tamamen rastgele olduğu ve yapılan şifreleme işlemlerinin diğer algoritmalara göre daha başarılı olduğu söylenebilir.

Çizelge 5.4. Farklı algoritmalarla şifrelenmiş görüntülerin korelasyon katsayıları

Algoritmalar	Renk	Yatay	Dikey	Çapraz	Ortalama
AES ECB	Kırmızı	-0,0023	-0,0021	0,0022	0.0022
	Yeşil	-0,0026	-0,0008	-0,0037	0,0024
	Mavi	0,0016	-0,0037	-0,0017	0,0023
AES CBC	Kırmızı	-0,0021	0,0033	-0,0020	0,0025
	Yeşil	0,0011	-0,0003	-0,0001	0,0005
	Mavi	0,0017	-0,0013	-0,0009	0,0013
AES GCM	Kırmızı	0,0029	0,0010	-0,0004	0,0014
	Yeşil	-0,0019	0,0001	-0,0011	0,0010
	Mavi	-0,0010	0,0030	-0,0024	0,0021
RC4 ECB	Kırmızı	-0,0036	0,0008	0,0007	0,0017
	Yeşil	-0,0015	0,0027	-0,0023	0,0022
	Mavi	-0,0007	-0,0034	-0,0037	0,0026
ChaCha20 Poly1305	Kırmızı	0,0009	-0,0003	0,0031	0,0014
	Yeşil	-0,0001	-0,0030	-0,0020	0,0017
	Mavi	-0,0008	-0,0001	-0,0033	0,0014

Bir görüntünün pikselleri arasında doğrusal bir ilişki varsa o zaman pikseller arasında yakın bir korelasyon var demektir. Aynı şekilde pikseller arasında doğrusal olmayan bir ilişki mevcutsa o zaman pikseller arasındaki korelasyon azdır ve o görüntünün pikselleri birbirleriyle ilişiksizdir (Singh ve ark., 2019). Başarıyla şifrelenmiş bir görüntünün de pikselleri arasında doğrusal bir ilişki bulunmamalıdır. Çünkü pikseller

arasındaki yüksek korelasyon, içsel bir görüntü özelliğidir ve güvenli bir şifreleme sistemi bu ilişkiyi kırmalıdır (Qiao ve ark., 2020). Şekil 5.2, 512 X 512 piksellik bir görüntünün orijinal ve AES GCM algoritması kullanılarak şifrelemiş haline ait yatay yöndeki komşu piksellerinin korelasyon dağılımını göstermektedir. Görüntünün her renk katmandaki korelasyon dağılımı şekilde aynı renkle temsil edilmiştir. Görüldüğü gibi orijinal görüntüde pikseller arasında doğrusal bir ilişkinin varlığı ve piksellerin mükemmel bir korelasyon içerisinde olduğu gözlemlenirken şifreli görüntüde de bu ilişkinin tamamen kırıldığı ve piksel değerlerinin birbirinden bağımsız şekilde rastlantısal olarak dağıldığı fark edilmektedir. Dolayısıyla yapılan şifreleme işleminin gayet başarılı olduğu söylenebilir.

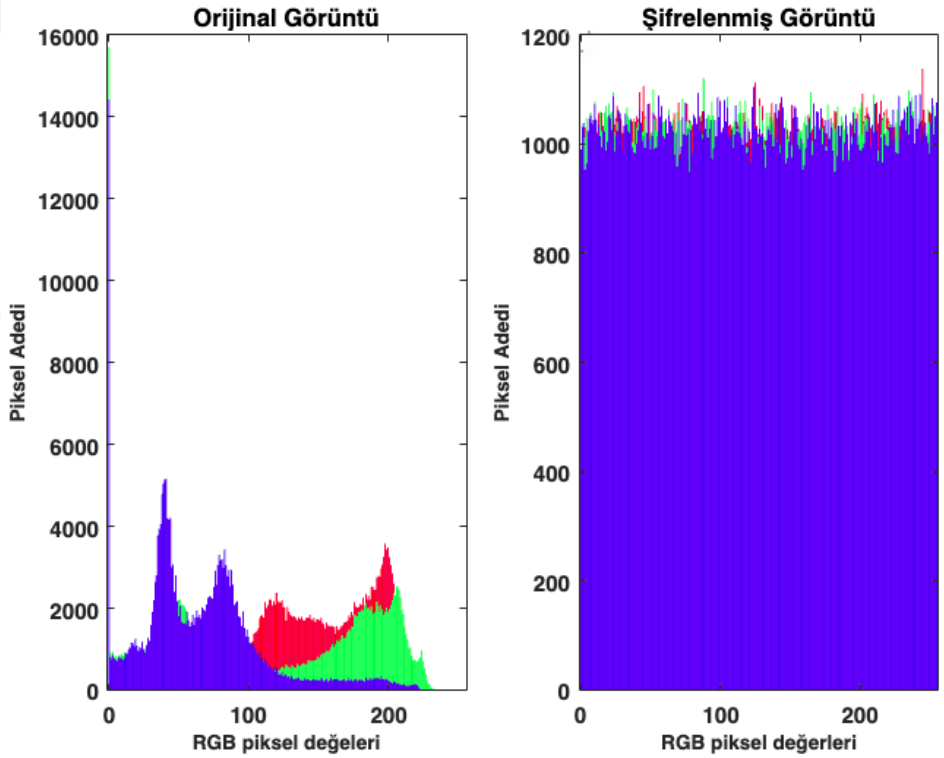


Şekil 5.2. Orijinal ve şifreli görüntülere ait korelasyon dağılımı

Herhangi bir şekilde saldırganlara bilgi sızmasını önlemek için şifrelenmiş görüntünün orijinaline ait hiç istatistiksel benzerlik içermemesi veya varsa bile bu benzerliğin olabildiğince az olması gerekmektedir. Bir görüntü histogramı, o görüntüdeki her renk yoğunluğu seviyesindeki piksel sayısını tespit edip bunu grafiğe dökerek görüntüdeki piksellerin dağılımını gösterir (Alsaffar ve ark., 2020). İyi bir şifreleme için şifrelenmiş bir görüntünün piksellerinin dağılımı tek tip veya dengeli olmalıdır ve bu, istatistiksel saldırılara karşı direnmek için sağlanması gereken temel bir koşuldur (Qiao ve ark., 2020).

Şekil 5.3, çalışma kapsamında AES GCM algoritması kullanılarak şifrelenen bir görüntünün ve onun orijinalinin histogram grafiğini göstermektedir. Grafikteki farklı

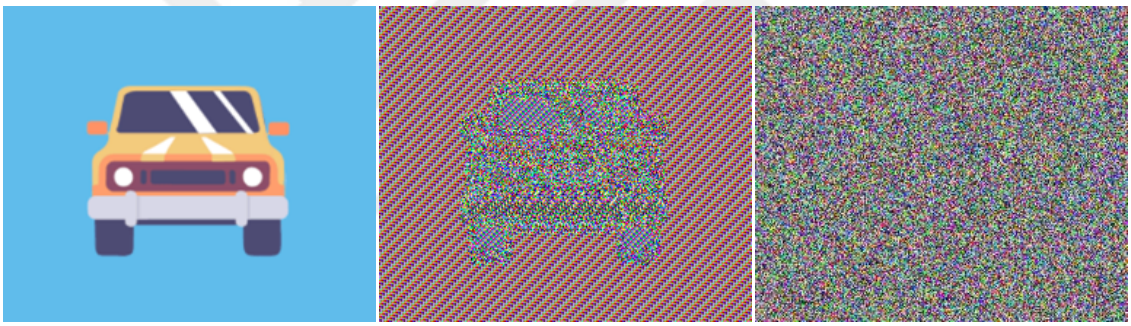
renkler görüntüdeki ilgili renk katmanına ait histogram değerlerini göstermektedir. Görüldüğü üzere orijinal görüntünün histogramı, büyük sivri uçlar ya da başka bir deyişle tepe noktaları içermektedir ve aynı zamanda bazı yerlerde de tam aksine düşük değerlerin olduğu gözlemlenmektedir. Bu durum normal bir görüntüde heterojen bir yapıya sahip olması, yani piksel değerlerinin bazı renk seviyelerinde yoğunlaşıp bazı seviyelerde seyrekleşmesinden kaynaklanmaktadır. Şifrelenmiş görüntünün histogramı ise tek düzedir, orijinalinden önemli ölçüde farklıdır ve orijinal görüntünün histogramına ait herhangi bir benzerlik taşımamaktadır. Çünkü şifreli görüntüde piksel değerleri tüm renk seviyeleri için olabildiğince eşit şekilde dağılım göstermiştir. Bu açıdan gerçekleştirilen şifreleme işleminin istatistiksel saldırılarda kullanılabilecek herhangi bir ipucu sağlamadığı ve dolayısıyla başarılı olduğu söylenebilir.



Şekil 5.3. Orijinal ve şifreli görüntülere ait histogram grafikleri

Yapılan analizlerde algoritmalarla beraber kullanılan şifreleme modlarının ya da diğer adıyla çalışma kiplerinin de hem şifreleme başarımını hem de işlem zamanını önemli ölçüde etkilediği gözlemlenmiştir. Çünkü şifreleme modları, herhangi bir blok şifreleyici algoritması kullanarak rastgele uzunluktaki mesajların olasılıksal şifrelenmesi için bir mekanizma tanımlamakta ve buna bağlı olarak şifreleme sürecindeki etkisi de büyük olmaktadır (Malozemoff ve ark., 2014). Bu kısma kadar yapılan

değerlendirmelerde blok şifreleme modlarının önemi, belli ölçütler kullanılarak sayısal verilerle ifade edilmiştir. Şekil 5.4' te blok şifreleme modlarının ne kadar etkili olduğunu ortaya koyan bir görsel sunulmuştur. Şekilde görülen 256 X 217 piksel çözünürlüğündeki araba animasyonu görseli AES algoritması kullanılarak hem ECB modunda hem de CBC modunda ayrı ayrı şifrelenmiş ve elde edilen şifreli görüntüler şekilde gösterilmiştir. Görüldüğü gibi iki şifreleme işleminde de aynı algoritma kullanılmasına rağmen algoritmanın çalışmasında tercih edilen şifreleme moduna bağlı olarak elde edilen sonuçlar çok büyük farklılık göstermektedir. CBC moduyla şifrelenmiş görüntüde pikseller, görüntünün her bölgesinde renk yoğunluğu seviyelerine homojen bir şekilde dağılmıştır ve şifrelenmiş görüntü, orijinal görüntüye ait herhangi bir ipucu içermemektedir. Ancak bunun aksine ECB modu kullanılarak şifrelenen görüntünün orijinal görüntüye ait paterni, net bir şekilde açığa çıkardığı gözlenmektedir ve bu durum güvenlik açısından büyük bir sorun teşkil etmektedir.



Şekil 5.4. Farklı modlarla şifrelenmiş görüntüler

ECB, 40 yılı aşkın bir süre önce FIPS 81' de DES için tanımlanan dört klasik şifreleme modundan birisidir. Ama klasik olması onu güvenilir yapmaz. ECB' nin iyi bilinen bazı temel sorunları vardır. Bu sorunlardan ilki aynı düz metin bloklarına karşılık aynı şifreli metin bloklarının üretilmesidir. Bu özellik ECB' nin istenilen gizliliği sağlayamayacağını ifade etmek için yeterlidir (Rogaway, 2011). Şekil 5.4' te şifreleme için kullanılan orijinal görüntüde arka plandaki piksellerin değerleri birbirleriyle aynıdır. Dolayısıyla bu görüntü, ECB modu kullanılarak şifrelendiği zaman arka plandaki tüm pikseller için elde edilen şifreli değerler aynı olmakta ve buna bağlı olarak da görüntünün modeli açıkça ortaya çıkmaktadır. Ancak CBC moduyla şifreleme yapıldığında böyle bir durum söz konusu değildir. Çünkü CBC modu zincirleme olarak adlandırılan ve düz metin bloklarının önceki şifreli metin bloklarıyla birleştirilmesini esas alan bir yöntem

uygulamaktadır (Dworkin, 2001). Ancak ilk düz metin bloğunu birleştirebilecek başka bir blok bulunmadığı için bir başlangıç vektörü gerektirmektedir. Bu yüzden CBC modu CFB ve OFB modlarıyla birlikte başlangıç vektörü tabanlı şifreleme şemaları olarak bilinir. CBC modu olasılıksal bir şifreleme şeması olarak güvenlidir ve rastgele bir IV kullanarak rastlantısal şifreli veriler üretir. Böylece aynı düz metin blokları için her defasında farklı şifreli veri blokları elde edilir. Şekil 5.4' te de görüldüğü üzere orijinal görüntünün büyük bir bölümünün aynı olmasına rağmen CBC modu kullanılarak rastlantısal bir şifreli görüntü elde edilmiştir. Burada dikkat edilmesi gereken nokta IV'nin yalnızca bir kez kullanılmasının yani şifrelenecek her veri için yeniden bir IV seçilmesinin gerekliliğidir.

Bu bölüm içerisinde yapılan incelemelerde steganografi işlemlerindeki başarı değerlendirmesinin yanı sıra tez çalışması kapsamında geliştirilen uygulamada kullanılabilecek en ideal şifreleme algoritmasının belirlenmesi için çeşitli analizler gerçekleştirilmiştir. Bu kapsamda simetrik anahtarlı şifreleme algoritmaları işlem zamanı, histogram ve korelasyon analizi gibi çeşitli açılardan ele alınmıştır. Tüm bu değerlendirmelere baktığımızda önerilen iletişim modeli ve geliştirilen anlık mesajlaşma uygulaması için en uygun ve avantajlı algoritmanın AES GCM olduğu tespit edilmiştir. Çünkü AES GCM hem işlem zamanı açısından göstermiş olduğu performans olarak anlık iletişim için uygun hem de mesajların güvenlik ve gizliliğin sağlanabilmesi açısından en başarılı algoritmadır. Bu başarıyı, hem görüntü şifreleme için kullanılan UACI, NPCR, MSE ve PSNR ölçütleri üzerinde hem de histogram ve korelasyon analizlerinde ispatlamıştır. Aynı zamanda sunmuş olduğu kimliği doğrulanmış şifreleme özelliği sayesinde mesajlaşma sürecinin daha güvenli gerçekleşmesini ve alınan mesajların doğruluğunun tespit edilebilirliğini sağlamıştır.

6. SONUÇLAR VE ÖNERİLER

6.1 Sonuçlar

Günümüzde hem sosyal yaşantıda hem de kurumsal alandaki en önemli problemlerden birisi bilgi ve iletişim güvenliğidir. Bu tez çalışmasında bu problemin çözümüne yönelik bir iletişim modeli sunulmuş ve bu model temel alınarak güvenlik ve gizliliği ön planda tutan, gerçek zamanlı haberleşme sağlayarak iletişim alanında piyasadaki muadillerine göre farklı bir seçenek sunan bir mobil mesajlaşma uygulaması geliştirilmiştir. Bu amaç doğrultusunda bilgi ve iletişim güvenliği üzerinde çalışan kriptoloji ve steganografi bilimleri detaylı olarak incelenmiştir. Bu kapsamda kriptoloji bilimin muhtevasında yer alan AES, DES, Blowfish, RC4, ChaCha20 algoritmaları, ECB, CBC, CFB, OFB, CTR, GCM ve Poly1305 çalışma kipleriyle beraber hem işlem zamanı açısından hem de görüntü şifreleme ölçütleri üzerinden karşılaştırılıp değerlendirilmiştir. Yapılan analizler sonucunda AES GCM algoritmasının hem orijinal verinin içeriğini %99,61 oranında değiştirerek gayet başarılı bir şifreme yaptığı hem de 269,61 saniye başına Megabayt (MB/s) şifreleme hızıyla işlem sürecini oldukça çabuk gerçekleştirdiği gözlemlenmiştir. İşlem zamanı açısından baktığımızda, AES algoritmasının 486,85 MB/s ile en hızlı CTR modunda çalıştığı görülmüştür. Zaten GCM modu da şifreleme için CTR kipini kullanmaktadır ve buna ek olarak özet elde etmek için GMAC metodunu uygulamaktadır (Kao ve ark., 2021). Bu metodun kullanımına bağlı olarak işlem hızında %44,62 oranında bir düşüş gerçekleşmektedir. Fakat GMAC metodu kimliği doğrulanmış şifreleme özelliğinin sağlanabilmesi için gereklidir ve bu özellik de güvenli bir iletişim süreci için vazgeçilemez niteliktedir. Çünkü iletişim sürecinde alıcı tarafın şifre çözme işlemini doğrulayabilmesi için özet fonksiyonlarının kullanılmasına ihtiyaç vardır ve GCM çalışma kipinde ise bu görevi GMAC algoritması gerçekleştirmektedir. Fakat işlem zamanı açısından verilen kayıp oran olarak bakıldığında yüksekmiş gibi gözükse de 269,61 MB/s' lik şifreleme hızı anlık iletişim için zaten fazlasıyla yeterlidir ve dolayısıyla iletişim sürecinin daha sağlıklı gerçekleşmesi adına verilen bu kayıp ihmal edilebilir. Bunların yanı sıra AES GCM algoritması kullanılarak yapılan görüntü şifreleme işlemi neticesinde elde edilen şifrelenmiş görüntünün yatay dikey ve çapraz yönlerdeki komşu piksellerine ait korelasyon katsayılarının ortalaması 0,0015 olarak hesaplanmıştır. Bu değer AES GCM' nin rastlantısal şifreli veriler üretme konusunda en başarılı algoritmalardan biri olduğunu göstermektedir. Elde edilen veriler ve yapılan

değerlendirmeler doğrultusunda AES GCM' nin yapılan çalışmada kullanılacak şifreleme algoritması için en doğru seçenek olduğu sonucuna varılmıştır.

Önerilen yaklaşımın başarıya ulaşması için yapılan şifrelemenin güvenliği kadar uygulanan steganografinin gizliliği de önemlidir. Bu yüzden geliştirilen uygulamada hem farklı çözünürlüklü görüntüler hem de farklı boyutta veriler kullanılarak steganografi işlemleri gerçekleştirilmiş ve bu işlemler sonucunda elde edilen stego-görüntüler, işlemlerin başarısını tespit etmek için MSE ve PSNR ölçütleri üzerinden değerlendirilmiştir. Literatürde bir steganografik işlemin başarılı olarak atfedilebilmesi için PSNR değerinin en az 40 dB olması gerektiği belirtilmiştir (Hamid ve ark., 2012). Yapılan çalışmada en düşük MSE 0,0003 ve en yüksek PSNR 82,39 dB olarak 2160 X 1440 px çözünürlüklü bir taşıyıcı görüntüye 1 KB boyutunda veri gömülmesiyle elde edilirken en yüksek MSE 0, 4740 ve en düşük PSNR 51,37 dB olmak üzere 1600 X 900 px çözünürlüklü bir taşıyıcı görüntüye 500 KB veri gömülmesiyle elde edilmiştir. Bu değerler, uygulanan steganografik işlemlerin literatürde tanımlanan eşik değere göre en az %28,43, en fazla %105,98 daha başarılı olduğunu göstermektedir. Dolayısıyla yapılan çalışmayı genel olarak değerlendirdiğimizde hem güvenliği hem de gizliliği sağlayan başarılı bir iletişim modelinin önerildiğini ve bunun başarıyla da uygulama geçirildiğini söyleyebiliriz.

6.2 Öneriler

Bu çalışmada uygulanan steganografi işlemleri sonucunda PNG formatında stego görüntüler elde edilmektedir. Gelecek çalışmalarda, JPEG formatlı stego görüntülerin de oluşturulabilmesi için OutGuess ve F5 gibi steganografi teknikleri de uygulanabilir. Ayrıca, önerilen yaklaşımda taşıyıcı ortam olarak ses, video gibi başka multimedya içerikleri veya az bellek gereksinimi ve düşük işlem zamanı sağlaması açısından metin dosyaları hatta ağ steganografisi uygulamak için TCP/UDP paketleri gibi birçok seçenek kullanılabilir. Bunlara ek olarak katılımcılara güvenliğe karşı performans tercihi sunabilmek için önerilen yaklaşımda kullanılan algoritma ve yöntemler açısından farklı spesifikasyonlar oluşturulabilir. Böylece kullanıcılara özel kişiselleştirilmiş bir iletişim sağlanabilir. Bunların yanı sıra önerilen yaklaşımdaki önemli konulardan birisi de anahtar paylaşımıdır. Yapılan çalışmada anahtar paylaşımı asimetrik bir yöntem olan RSA algoritmasıyla gerçekleştirilmektedir ve RSA' ya ait gizli anahtarlar, yüksek güvenlik için alıcı tarafında cihazdan çıkarılamayacak şekilde bir anahtar deposunda saklanmaktadır. Bu da kullanıcıların cihaz değiştirmeleri durumunda eski mesajlarının şifresini

çözemeyeceği anlamına gelir. Bu durumun oluşmaması için anahtar paylaşımında bir alternatif olarak Diffie - Helman protokolü uygulanabilir. Böylece bir anahtar saklama gereksinimi oluşmaz, ancak protokolde kullanılan parametrelerin bir sabit olarak tanımlanması gerekir. Ayrıca yine RSA kullanılması durumunda da güvenlikten biraz taviz verilerek gizli anahtarlar, uygulamanın yerel kütüphanelerinde veya kullanılan işletim sistemine özgü farklı veri gizleme stratejileriyle saklanabilir ve anahtarların yedeklenmesi mümkün hale getirilerek başka cihazlarda da önceden şifrelenmiş mesajlar çözülebilir.

Bu tez çalışmasındaki önerilen yaklaşımda uygulanan kriptoloji ve steganografi algoritmaları anlık iletişim için fazlasıyla hızlı çalışmaktadır. Ancak steganografik işlemlerin uygulanabilmesi için görüntüler önce BMP formatına çevrilip sonra işlendikten sonra tekrar sıkıştırılır ve bu sıkıştırma uygulaması için harcanan işlem zamanı, kriptoloji ve steganografi algoritmalarının göstermiş olduğu yüksek hızlı performansın önüne geçmektedir. Aynı zamanda kullanılan fotoğrafların yüksek boyutlu olması durumunda bunların sunucuya yüklenmesi gecikmelere sebep olmaktadır. Bu açıdan önerilen yaklaşımın işlem zamanı açısından performansının iyileştirilebilmesi için daha hızlı ve başarılı bir sıkıştırma yöntemi uygulanabilir.

KAYNAKLAR

Abood, O. G. and Guirguis S., 2018, A Survey on Cryptography Algorithms, *International Journal of Scientific and Research Publications*, 8(7), s. 495-516.

Aggarwal, S., Goyal N. and Aggarwal K., 2014, A review of Comparative Study of MD5 and SHA Security Algorithm, *International Journal of Computer Applications*, 104(14), s. 1-4.

Ahamad, M. M. and Abdullah, M. I., 2016, Comparison of encryption algorithms for multimedia. *Rajshahi University Journal of Science and Engineering*, 44, 131-139.

Ahmad, J. and Ahmed, F., 2010, Efficiency analysis and security evaluation of image encryption schemes, *International Journal of Video & Image Processing and Network Security*, 12 (4), 18-31.

Ahmad, N., Wei, L. M. and Jabbar M. H., 2018, Advanced Encryption Standard with Galois Counter Mode using Field Programmable Gate Array, *Journal of Physics: Conference Series*, vol. 1019, 1st International Conference on Green and Sustainable Computing, Kuching, Sarawak, Malaysia.

Ahmad, S., Beg, M. R., Abbas, Q., Ahmad, J. and Atif, S. Y., 2010, Comparative study between stream cipher and block cipher using RC4 and Hill Cipher, *International Journal of Computer Applications*, 1 (25), 15-21.

Al-Mamun, A., Shaon, T. A. and Hossain, A., 2017, Security Analysis of AES and Enhancing Its Security by Modifying S-Box with An Additional Byte, *International Journal of Computer Networks & Communications (IJCNC)*, 69-88.

Alsaffar, D. M., Almutiri, A. S., Alqahtani, B., Alamri, R. M., Alqahtani, H. F., Alqahtani, N. N. and Ali, A. A., 2020, Image Encryption Based on AES and RSA Algorithms, *In 3rd International Conference on Computer Applications & Information Security (ICCAIS)*, IEEE, 1-5.

Amritha G. and Varkey M., 2013, A security enhanced approach for digital image steganography using DWT and RC4 encryption, *International Journal of Computer Trends and Technology* 4 (6), 1710–1716.

Anonim, 2018, Gürültü (elektronik), *Wikipedia Özgür Ansiklopedi*, https://tr.wikipedia.org/wiki/G%C3%BCr%C3%BClt%C3%BC_(elektronik), [Ziyaret Tarihi: 25 Ağustos 2021].

Anonim, 2021, Android Studio, *Wikipedia Özgür Ansiklopedi*, https://tr.wikipedia.org/wiki/Android_Studio, [Ziyaret Tarihi: 17 Temmuz 2021].

Anonim, 2020, Yazılım geliştirme kiti, *Wikipedia Özgür Ansiklopedi*, https://tr.wikipedia.org/wiki/Yaz%C4%B1l%C4%B1m_geli%C5%9Ftirme_kiti, [Ziyaret Tarihi: 22 Temmuz 2021].

Anonim, 2021, OpenCV, *Wikipedia Özgür Ansiklopedi*, https://tr.wikipedia.org/wiki/OpenCV, [Ziyaret Tarihi: 25 Temmuz 2021].

Anonim, 2021, Öykünücü, *Wikipedia Özgür Ansiklopedi*, https://tr.wikipedia.org/wiki/%C3%96yk%C3%BCn%C3%BCc%C3%BC, [Ziyaret Tarihi: 27 Temmuz 2021].

Anonim, 2021, Firebase, *Wikipedia Özgür Ansiklopedi*, https://tr.wikipedia.org/wiki/Firebase, [Ziyaret Tarihi: 29 Temmuz 2021].

Anonim, 2021, JSON, *Wikipedia Özgür Ansiklopedi*, https://tr.wikipedia.org/wiki/JSON, [Ziyaret Tarihi: 29 Temmuz 2021].

Anonymous, Business Communication for Success, 2015, *University of Minnesota Libraries Publishing Edition*, https://open.lib.umn.edu/businesscommunication/, [Ziyaret Tarihi: 23 Ağustos 2021]. s. 7-12.

Atalay, N., Doğan, Ş., Tuncer, T., Akbal, E., 2019, İmge Şifreleme Yöntem ve Algoritmaları. *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, 10 (3), s. 815-831.

Atıcı, M., Sağıroğlu, Ş., 2016, Steganografi Tabanlı Yeni Bir Klasör Kilitleme Yaklaşımı ve Yazılımı Geliştirilmesi, *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, 31 (1), s. 129-144.

Aydoğan, M., Avcı, E., 2013, OPT Görüntülerine Kriptografik ve Steganografik İşlemlerin Uygulanması, *1st International Symposium on Digital Forensics and Security (ISDFS'13)*, Elâzığ, Türkiye, s. 240-243.

Balcı, D., 2019, Hibrid yöntemler kullanılarak tıbbi DICOM verilerinin güvenliğinin sağlanması, Yüksek Lisans Tezi, *Gazi Üniversitesi Fen Bilimleri Enstitüsü*.

Bandyopadhyay S.K., Bhattacharyya D., Ganguly D., Mukherjee S., Das P., 2008, A Tutorial Review on Steganography, *International Conference on Contemporary Computing (IC3-2008)*, Noida, India, s. 105-114.

Basu, S., 2011, International Data Encryption Algorithm (IDEA) – A Typical Illustration, *Journal of Global Research in Computer Science*, 2 (7), 116-118.

Bennett, K., 2004, Linguistic Steganography: Survey, Analysis and Robustness Concerns for Hiding Information in Text, *Purdue University, CERIAS Tech Report 2004-13, West Lafayette, Indiana, USA*, 12-13.

Bernstein, D. J., 2005 The Poly1305-AES Message-Authentication Code, *International Workshop on Fast Software Encryption*, Springer, Berlin, Heidelberg, 32-49.

Bernstein, D. J., 2008, ChaCha, a variant of Salsa20, *Workshop record of SASC*, 8 (1), 3-5.

Beşkirli, A., Özdemir, D. ve Beşkirli M., 2019, Şifreleme Yöntemleri ve RSA Algoritması Üzerine Bir İnceleme, *Avrupa Bilim ve Teknoloji Dergisi*, Özel Sayı, s. 284-291.

Bhowal K., Bhattacharyya D., Jyoti Pal A. and Kim T. H., 2013, A GA based audio steganography with enhanced security, *Telecommunication Systems*,52 (4). 2197-2204.

Brassil, J. T., Low, S., Maxemchuk, N. F., O'Gorman, L., 1995, Electronic marking and identification techniques to discourage document copying, *IEEE Journal on Selected Areas in Communications*, 13(8), s. 1495-1504.

Chan, C. K. and Cheng, L. M., 2004, Hiding data in images by simple LSB substitution, *Pattern recognition*, 37 (3), s. 469-474.

Chandra, S., Paira, S., Alam, S., Sanyal, G., 2014, A comparative survey of symmetric and asymmetric key cryptography, *International Conference on Electronics, Communication and Computational Engineering (ICECCE)*, Hosur, India, 83-93.

Cheddad, A., Condell, J., Curran, K., Kevitt, P., 2010, Digital image steganography: Survey and analysis of current methods, *Signal Processing*, 90(3), s. 727-752.

Cherry, C., 1957, On Human Communication: A Review, a Survey, and a Criticism, 2th. Edition, *Technology Press of Massachusetts Institute of Technology*, Brooklyn, NY. s. 3-7.

Cihangir, S., 2020, Design and implementation of an image based steganography, Yüksek Lisans Tezi, *Ankara Yıldırım Beyazıt Üniversitesi Fen Bilimleri Enstitüsü*.

Cvejic, N. and Seppanen, T., 2002, Increasing the capacity of LSB-based audio steganography, *IEEE Workshop on Multimedia Signal Processing*. St. Thomas, VI, USA, s. 336-338.

Darbani, A., AlyanNezhadi, M. M. and Forghani, M., 2019, A new steganography method for embedding message in JPEG images. *5th Conference on Knowledge Based Engineering and Innovation*, IEEE, 617-621.

Dutta, P., Bhattacharyya D., Kim T., 2009, Data Hiding in Audio Signal: A Review. *International Journal of Database Theory and Application*, 2(2), s. 1-8.

Dworkin, M., 2001, Recommendation for block cipher modes of operation, methods and techniques, *NIST Special Publication 800-38A Computer security Div*., Gaithersburg MD.

Dworkin, M. J., 2007, Recommendation for block cipher modes of operation: Galois/Counter mode (GCM) and GMAC, *NIST Special Publication 800-38D Computer security Div*., Gaithersburg MD.

El Fishawy, N. F. ve Zaid, O. M. A., 2007, Quality of encryption measurement of bitmap images with RC6, MRC6, and Rijndael block cipher algorithms. *International Journal of Network Security*, 5(3), 241-251.

Elkamchouchi, H. M., ve Makar, M. A., 2005, Measuring encryption quality for bitmap images encrypted with rijndael and kamkar block ciphers. *In Proceedings of the 22th National Radio Science Conference*, 277-284.

Engin, S., 2020, RSA Şifreleme Yöntemi ve Özellikleri Üzerine, Yüksek Lisans Tezi, *Erciyes Üniversitesi Fen Bilimleri Enstitüsü*, Kayseri, 24-33.

Farah, S., Javed, Y., Shamim, A. and Nawaz, T., 2012, An experimental study on performance evaluation of asymmetric encryption algorithms. *Recent Advances in Information Science, Proceeding of the 3rd European Conf. of Computer Science (EECS-12)* 121-124.

Fluhrer, S., Mantin, I., and Shamir, A., 2001, Weaknesses in the Key Scheduling Algorithm of RC4, *8th International Workshop on Selected Areas in Cryptography*, Toronto, Ontario, Canada, 1-24.

Forgae R. and Oekay, M., 2019, Contribution to Symmetric Cryptography by Convolutional Neural Networks. *Communication and Information Technologies (KIT*), IEEE.

Furht B., Muharemagic E. and Socek D., 2005, Multimedia encryption and watermarking, cilt 28, *Springer Science*, Boca Raton, FL, USA, 59-61, 77-78.

Fukushima, K., Xu, R., Kiyomoto, S. and Homma, N., 2017, Fault injection attack on Salsa20 and ChaCha and a lightweight countermeasure, *IEEE Trustcom/BigDataSE/ICESS*, Sydney, NSW, Australia, 1032-1037.

Goel S., Rana A. ve Kaur M., 2013, A Review of Comparison Techniques of Image Steganography, *Global Journal of Computer Science and Technology Graphics & Vision*, 13 (4).

Google, 2021, Android keystore system, *Android Developer Guides*, https://developer.android.com/training/articles/keystore, [Ziyaret Tarihi: 15.04.2022]

Google, 2021, Introduction to Activities, *Android Developer Guides*, https://developer.android.com/guide/components/activities/intro-activities, [Ziyaret Tarihi: 14.07.2022]

Goyal, T. K., Sahula, V., and Kumawat, D., 2019, Energy efficient lightweight cryptography algorithms for IoT devices. *IETE Journal of Research*, 1-14.

Gökrem L., Nasip O., 2017, An Encrypted Messaging Application with Multi Fragmented Caesar Encryption Method between Mobile Devices. *Journal of New Results in Science*, 6 (1), s. 1-10.

Gövem, B., 2013, Klonlanamaz Fonksiyonlar ve Yardımcı Bilgiler Kullanılarak Patent Hakları Korunması, Özgün Algoritma ve Donanımın Güvenliğinin Sağlanması, Yüksek Lisans Tezi, *İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü*, İstanbul, s. 23.

Gupta S., Sharma J., 2012, A hybrid encryption algorithm based on RSA and Diffie-Hellman, *2012 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, Coimbatore, India. s. 55-58.

Gupta S., Goyal A., Bhushan B., 2012, Information Hiding Using Least Significant Bit Steganography and Cryptography, *International Journal of Modern Education and Computer Science (IJMECS)*, 6, s. 27–34.

Gürel H., 2006, Sayısal resim içerisine veri gizleme uygulamaları, Yüksek Lisans Tezi, *Kocaeli Üniversitesi Fen Bilimleri Enstitüsü*.

Gürkaş, G. Z., 2005, Kablosuz Güvenlik Protokollerinin Karşılaştırmalı Analizi, Yüksek Lisans Tezi, *İstanbul Üniversitesi Fen Bilimleri Enstitüsü*, İstanbul, 33-35.

Hamid, N., Yahya, A., Ahmad, R. B. and Al-Qershi, O. M., 2012, Image steganography techniques: an overview, *International Journal of Computer Science and Security (IJCSS)*, 6 (3), s. 168-187.

Hossain A., Islam K., Das S. Nashiry A., 2019, Cryptanalyzing of Message Digest Algorithms MD4 and MD5, *International Journal on Cryptography and Information Security (IJCIS)*, 2(1), s. 1-13.

Howgrave-Graham N., Nguyen, P., Pointcheval, D., Proos, J., Silverman, J.H., Singer A. and Whyte, W., 2003, The Impact of Decryption Failures on the Security of NTRU Encryption, *Advances in Cryptology - CRYPTO 2003*, 226-246.

Huang, D. and Yan, H., 2001, Interword Distance Changes Represented by Sine Waves for Watermarking Text Images, *IEEE Transactions on Circuits and Systems for Video Technology*, 11 (2), 1237-1245.

Hussain M. and Hussain M., 2013, A Survey of Image Steganography Techniques, *International Journal of Advanced Science and Technology*, 54, 113-123.

Hussain M., Wahab A., Yamani I., Anthony T.S. Ho, and Ki-Hyun Jung, 2018, Image steganography in spatial domain: A survey, *Signal Processing: Image Communication*, vol. 65, s. 46-66.

Imran, O. A., Yousif, S. F., Hameed I. S., Al-Din Abed W. N. and Hammid A. T., 2020, Implementation of El-Gamal algorithm for speech signals encryption and decryption, *Procedia Computer Science* 167, 1028-1037.

Iswari, N. M. S., 2016, Key Generation Algorithm Design Combination of RSA and ElGamal Algorithm, *8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, Yogyakarta, Indonesia. 186-190.

Jain, V., Kumar, L., Sharma, M. M., Sadiq, M. and Rastogi, K., 2012, Public-Key Steganography Based on Modified LSB Method, *Journal of Global Research in Computer Science*, 3 (4), 26-29.

Jayaram, P., Ranganatha H. R. and Anupama, H. S., 2011, Information Hiding Using Audio Steganography-A Survey, *The International Journal of Multimedia & Its Applications (IJMA)*, 3 (3), 86-96.

Josefsson, S., 2006, The base16, base32, and base64 data encodings, RFC 4648, 4-6.

Kahate, A., 2013, Cryptography and Network Security 3th. Edition, *McGraw Hill Education*, New Delhi, 32-53.

Kamali, S. H., Shakerian, R. Ve Hedayati, M., 2010, A new modified version of advanced encryption standard based algorithm for image encryption, *In 2010 International Conference on Electronics and Information Engineer*ing, IEEE, cilt 1, 141-145.

Kao, T. L., Wang, H. C. ve Li, J. E., 2021, Safe MQTT-SN: a lightweight secure encrypted communication in IoT, *Journal of Physics: Conference Series*, 1, IOP Publishing.

Karim, M. ve Hossain, I., 2011, A new approach for LSB based image steganography using secret key, *14th International Conference on Computer and Information Technology (ICCIT)*, Dhaka, Bangladesh. 286-291.

Kasper, E. and Schwabe, P., 2009, Faster and timing-attack resistant AES-GCM. *International Workshop on Cryptographic Hardware and Embedded Systems* Springer, Berlin, Heidelberg, 1-17.

Katz J., ve Lindell, Y., 2021, Introduction to Modern Cryptography 3th Edition, *CRC Press*, Boca Raton, FL, 1-5.

Kaur, N. ve Behal S., 2014, A Survey on various types of Steganography and Analysis of Hiding Techniques, *International Journal of Engineering Trends and Technology (IJETT)*, 11 (8), 388-392.

Koblitz, N., 1987, Elliptic Curve Cryptosystems, *Mathematics of Computation*, 48 (177), 203-209.

Kodal, Sevindir, H., ve Sayın N., 2018, Dalgacık Dönüşümü Tabanlı Görsel Kriptoloji, *Afyon Kocatepe Üniversitesi Fen ve Mühendislik Bilimleri Dergisi*, 18 (3), 888-894.

Koley S., Pal K., Ghosh G. and Bhattacharya M, 2014, Secure transmission and recovery of embedded patient information from biomedical images of different modalities through a combination of cryptography and watermarking, *International Journal of Image, Graphics and Signal Processing*, 6 (4), 18–31,

Küçük M., Eriş U., Oğuz T., Dal A., Aydın C. H., Orhon E. N., 2012, İletişim Bilgisi 1.Baskı, *Anadolu Üniversites*i, Eskişehir, s. 6-12.

Langley, A., Chang, W., Mavrogiannopoulos, N., Strombergson, J. and Josefsson, S. 2016, ChaCha20-Poly1305 cipher suites for Transport Layer Security (TLS), *Internet Engineering Task Force (IETF), Request for Comments (RFC): 7905*, https://www.rfc-editor.org/rfc/rfc7905.txt, [Ziyaret Tarihi: 27.04.2022].

Lipmaa H., Rogaway P. and Wagner D., 2000, Comments to NIST concerning AES-modes of operations: CTR-mode encryption, *In Symmetric Key Block Cipher Modes of Operation Workshop*, Baltimore, Maryland, USA.

Lin C., Yeh Y., Chien S., Lee C., Chien H., 2011, Generalized secure hash algorithm: SHA-X, *IEEE EUROCON - International Conference on Computer as a Tool*, Lisbon, Portugal.

Malozemoff, A. J., Katz, J. ve Green, M. D., 2014, Automated analysis and synthesis of block-cipher modes of operation, In 27th Computer Security Foundations Symposium, IEEE, 140-152.

Mammadov, I., 2010, J2ME tabanlı mobil e-ticarette noktadan noktaya güvenlik, Yüksek Lisans Tezi, *İstanbul Üniversitesi Fen Bilimleri Enstitüsü*, İstanbul, 33-38.

McGrew, D. and Viega, J., 2004, The Galois/counter mode of operation (GCM), *Submission to NIST Modes of Operation Process*, 1-10.

Menezes, A.J., Oorschot, P. C. ve Vanstone, S. A., 1997, Handbook of Applied Cryptography", 5th. Edition, *CRC Press*, 25-32.

Miller, V.S., 1985, Use of Elliptic Curves in Cryptography, *Advances in Cryptology, CRYPT0 '85 Proceedings*, 417-426.

Mohan H. S. and Reddy, A. R., 2011, Performance Analysis of AES and MARS Encryption Algorithms, *IJCSI International Journal of Computer Science Issues*, 363-368.

Morkel, T., Eloff J. H. P. ve Olivier M. S., 2005, An Overview of Image Steganography, *Proceedings of the Fifth Annual Information Security South Africa Conference (ISSA2005)*, Sandton, South Africa, 1-11.

Mousa, A., 2005, Data encryption performance based on Blowfish, *47th International Symposium ELMAR*, Zadar, Croatia, 131-134.

Mousa A., Hamad A., 2006, Evaluation of the RC4 Algorithm for Data Encryption. *International Journal of Computer Sciences*, 3 (2), s. 44-56.

Nair A. S., Kumar A., Sur A. and. Nandi S., 2011, Length based network steganography using UDP protocol, *3rd International Conference on Communication Software and Networks*, Xi'an, China, 726-730.

National Institute of Standards and Technology, 1998, DES Modes of Operation, *FIPS PUB 81*, http://csrc.nist.gov/publications/fips/fips81/fips81.htm, [Ziyaret Tarihi: 25.04.2022]

Nie, T. ve Zhang, T., 2009, A study of DES and Blowfish encryption algorithm, *TENCON IEEE Region 10 Conference*, Singapore, 1067-1070.

Nir, Y., and Langley, A., 2015, ChaCha20 and Poly1305 for IETF Protocols, *Request for Comments: 7539*, https://www.rfc-editor.org/rfc/rfc7539.txt, [Ziyaret Tarihi: 27.04.2022]

Oracle, 2014, KeyGenerator (Java Platform SE 8), *Oracle Help Center*, https://docs.oracle.com/javase/8/docs/api/javax/crypto/KeyGenerator.html, [Ziyaret Tarihi: 10.07.2022]

Oracle, 2014, SecureRandom (Java Platform SE 8), *Oracle Help Center*, https://docs.oracle.com/javase/8/docs/api/java/security/SecureRandom.html, [Ziyaret Tarihi: 10.07.2022]

Öztürk, S., 2021, Android Studio nedir ve ne işe yarar? Android Studio kurulumu ve kullanımı hakkında bilgi, *Hürriyet Gazetesi*, https://www.hurriyet.com.tr/teknoloji/android-studio-nedir-ve-ne-ise-yarar-android-studio-kurulumu-ve-kullanimi-hakkinda-bilgi-41792487, [Ziyaret Tarihi: 4 Eylül 2021]

Özyılmaz, Ç., 2014, Kriptolojiye Giriş, Yüksek Lisans Tezi, *Karabük Üniversitesi Fen Bilimleri Enstitüsü*, Karabük, 9-14.

Padmavathi B. and. Ranjitha Kumari S., 2013, A survey on performance analysis of DES, AES and RSA algorithm along with LSB substitution technique, *International Journal of Science and Research*, 2 (4), 170–174.

Pearson J. C., Nelson E. N., Titsworth S., Hosek M. A., 2017, Human Communication, 6th. Edition, *McGraw-Hill Education*, 2 Penn Plaza, New York, NY, s. 8-12.

Peruginelli S., Bergamin G., Ammendola P., 1992, Character sets: towards a standard solution, *Program: electronic library and information systems*, 26(3), s. 215-223.

Pittalia P. P, 2019, A Comparative Study of Hash Algorithms in Cryptography, *International Journal of Computer Science and Mobile Computing*, 8(3), s. 147-152.

Popa, R., 1998, An Analysis of Steganographic Techniques, *The Politehnica University of Timisoara, Faculty of Automatics and Computers*, Romania, 20-25.

Prajapati, K. S. ve Buddhdev, B. V., 2014, Overview of Improvements and Modifications in RSA Algorithm, *Journal of Emerging Technologies and Innovative Research (JETIR)*, 1 (7), 656-659.

Qiao M., Sung A., Liu Q., 2009, Feature Mining and Intelligent Computing for MP3 Steganalysis, *International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing*, Shanghai, China s. 627-630.

Qiao, Z., El Assad, S. ve Taralova, I., 2020, Design of secure cryptosystem based on chaotic components and AES S-Box. *AEU-International Journal of Electronics and Communications*, 121, 153205.

Rijmen V., Oswald E., 2005, Update on SHA-1, *The Cryptographers' Track at the RSA Conference (CT-RSA)*, San Francisco, CA, USA, s. 58-71.

Rise, R., Cho, S. ve Kaylor, D., RC4 Encryption, *University of Washington*, https://sites.math.washington.edu/~nichifor/310_2008_Spring/Pres_RC4%20Encryption.pdf, [Ziyaret Tarihi: 4 Eylül 2021].

Rogaway, P., 2011, Evaluation of some blockcipher modes of operation, *Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan*,

Rosen R., 2014, Linux Kernel Networking, 1th. Edition, *Apress*, Berkeley, CA, s. 37-61.

Sachdev, A. ve Bhansali, B., 2013, Enhancing Cloud Computing Security using AES Algorithm, *International Journal of Computer Applications*, 67 (9), 19-23.

Sahni, M., 2019, An Analysis on Steganography, *International Journal of Research in Engineering*, 9, 153-165.

Sann, Z., Soe, T., Knin, K. W. M. ve Win, Z. M., 2019, Performance comparison of asymmetric cryptography (case study-mail message), *Journal on Computer Science and Information Technologies*, 4 (3), 105-111.

Seidel, T., Socek, D. and Sramka, M., 2004, Parallel Symmetric Attack on NTRU using Non-Deterministic Lattice Reduction, *Designs, Codes and Cryptography*, 32, 369-379.

Sekhar, A., Kumar, M. and Rahiman, M. A., 2015, A novel approach for hiding data in videos using network steganography methods, *Procedia Computer Science*, 70, 764-768.

Schneier, B., 1994, Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish), *In Fast Software Encryption Second International Workshop*, Leuven, Belgium, Proceedings, *Springer-Verlag*, 191-204.

Schneier, B., 1996, Applied Cryptography: Protocols, Algorithms and Source Code in C, 2nd Edition, *John Wiley & Sons*, New York, USA,

Shakir A. N., 2016, A robust encryption and data hiding technique by using hybrid DES and LSB algorithm, Yüksek Lisans Tezi, *Çankaya Üniversitesi Fen Bilimleri Enstitüsü*, Ankara.

Shannon, C. E., 1949, Communication theory of secrecy systems. *The Bell system technical journal*, 28(4), 656-715.

Sharma K., Aggarwal A., Singhania T., Gupta D. and A. Khanna, 2019, Hiding data in images using cryptography and deep neural network, *Journal of Artificial Intelligence and Systems*, 1,143-162.

Sheelu, Ahuja, B., 2013, An Overview of Steganography, *IOSR Journal of Computer Engineering*, 11 (1), 15-19.

Shirali-Shahreza, M., 2008, Text Steganography by Changing Words Spelling, *10th International Conference on Advanced Communication Technology*, Gangwon, Korea (South), 1912-1913.

Sindhu M. ve Rajkamal R., 2009, Images and its compression techniques - a review, *International Journal of Recent Trends in Engineering*, 2 (6), 71-75.

Singh, A., Agarwal, P. ve Chand, M., 2019, Image encryption and analysis using dynamic AES. *In 5th International Conference on Optimization and Applications (ICOA)* IEEE 1-6.

Smitha, G. ve Baburaj, E., 2016, A survey on image steganography based on Least Significant bit Matched Revisited (LSBMR) algorithm, *2016 IEEE International Conference on Emerging Technological Trends (ICETT)*, Kollam, India, 1-6.

Soyalıç, S., 2005, Kriptografik Hash Fonksiyonları ve Uygulamaları, Yüksek Lisans Tezi, *Erciyes Üniversitesi Fen Bilimleri Enstitüsü*, Kayseri, 1-2.

Stallings, W., 2011, Cryptography and Network Security Principles and Practice 5th. Edition, *Prentice Hall*, Upper Saddle River, NY, USA. 342-345.

Sumagita M., Riadi I., 2018, Analysis of Secure Hash Algorithm (SHA) 512 for Encryption Process on Web Based Application, *International Journal of Cyber-Security and Digital Forensics (IJCSDF)*, 7(4), s. 373-381

Svenda, P., 2016, Basic Comparison of Modes for Authenticated-Encryption (IAPM, XCBC, OCB, CCM, EAX, CWC, GCM, PCFB, CS), https://www.fi.muni.cz/~xsvenda/docs/AE_comparison_ipics04.pdf, [Ziyaret Tarihi: 10 Temmuz 2022]

Şahin, F., 2015, Modern Blok Şifreleme Algoritmaları, *İstanbul Aydın Üniversitesi Dergisi*, 26, 23-40.

Thakur, J. ve Kumar, N., 2014, DES, AES and Blowfish: Symmetric Key Cryptography Algorithms Simulation Based Performance Analysis, *International Journal of Emerging Technology and Advanced Engineering*, 1 (2), 6-12.

Tosun, L. P., 2010, Preference For Communication Technologies: Characteristics of Channels, Users and Communication Situations, Doktora Tezi, *Orta Doğu Teknik Üniversitesi*, Ankara, 1-3.

Xia Z., Ge Z., 2010, MD5 research, *2th International Conference on Multimedia and Information Technology*, Kaifeng, China, s. 271-273.

Yan, J. and Chen, F., 2016, An improved AES key expansion algorithm. *International Conference on Electrical, Mechanical and Industrial Engineering*, 113-116.

Yasin E. T., 2017, Image encryption based on Haar wavelet transformation and RC4 algorithm, Yüksek Lisans Tezi, *Yüzüncü Yıl Üniversitesi Fen Bilimleri Enstitüsü*, Van, 23-27.

Yerlikaya T., Gençoğlu H., 2017, Mobil Cihazlarda RSA Algoritmasının Performans Optimizasyonu, *Trakya Üniversitesi Mühendislik Bilimleri Dergisi*, 18 (1), s. 43-52.

Yeshwanth Srinivasan B. E., 2003, High Capacity Data Hiding System Using BPCS Steganography, Yüksek Lisans Tezi, *Texas Tech University*, Lubbock, Texas.

Yousif S. F., Abboud A. J., Radhi H. Y., 2020, Robust Image Encryption With Scanning Technology, the El-Gamal Algorithm and Chaos Theory, *IEEE Access*, 8, s. 155184 – 155209.

Wu, Y., Noonan, J. P. ve Agaian, S., 2011, NPCR and UACI randomness tests for image encryption. *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)*, 1(2), 31-38.

Zheng X., Jin J., 2012, Research for the Application and Safety of MD5 Algorithm in Password Authentication, *9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, Chongqing, China, s. 2216- 2219.

Zhou, S., He, P. ve Kasabov, N., 2020, A dynamic DNA color image encryption method based on SHA-512, *Entropy*, 22 (10), 1091.