**2. Metodoloji**

Bu çalışmada geliştirilen görüntü şifreleme sistemi, RGB formatındaki dijital görsellerin çok katmanlı bir şifreleme sürecinden geçirilerek güvenli bir biçimde saklanmasını ve yalnızca yetkilendirilmiş kullanıcılar tarafından tekrar çözülebilmesini amaçlamaktadır. Metodoloji; sırasıyla DNA tabanlı kodlama, XOR şifreleme, kaotik Logistic Map kullanımı ve RSA ile metadata şifreleme olmak üzere dört ana bileşenden oluşmaktadır. Bu bölümde, her bir adım hem teorik hem de uygulamalı olarak örneklerle birlikte detaylandırılmıştır.

**2.1. DNA Kodlama**

**Teorik Arka Plan**

DNA kodlama yöntemi, dijital verinin biyolojik bir temsil üzerinden dönüştürülmesini sağlar. 8-bitlik bir byte değeri, 2-bitlik parçalara bölünerek her parça bir DNA bazıyla eşleştirilir. Bu eşleştirme işlemi önceden tanımlanmış 8 farklı DNA eşleme kuralına göre gerçekleştirilir. Örneğin, aşağıdaki eşleme kurallarından birine göre:

00 → A

01 → G

10 → C

11 → T

**Uygulama Aşaması**

1. RGB görüntü H x W x 3 şeklinde bir NumPy dizisine sahiptir.
2. Bu görüntü flatten() metodu ile tek boyutlu hale getirilir.
3. Her byte değeri (0–255) ikilik (binary) forma çevrilir.
4. Binary string 2-bit’lik parçalara ayrılır.
5. Her parça, DNA kuralına göre bir harfe çevrilir.

**Örnek**

Girdi Byte: 65 → Binary: 01000001

2-bit Parçalar: 01 00 00 01 → DNA: G A A G

**Fonksiyonel Gösterim**

Let:

* B = {b\_1, b\_2, ..., b\_n}: Byte dizisi
* f\_dna: {00,01,10,11} → {A,T,G,C}: DNA kuralı eşlemesi

Kodlama işlemi:

D=fdna(split(bits(B),2))D = f\_dna(split(bits(B), 2))

Çıktı: DNA harf dizisi D

**2.2. XOR Şifreleme**

**Teorik Arka Plan**

XOR işlemi, iki bit arasındaki eşitlik durumuna göre 0 veya 1 döndüren bir mantıksal işlemdir. Simetrik kriptografi için uygundur çünkü aynı işlem hem şifrelemede hem de çözmede kullanılabilir.

**Uygulama Aşaması**

1. DNA ile kodlanmış harfler, ASCII değerleri alınarak tekrar byte dizisine çevrilir.
2. Her byte, xor\_key (0–255 arası rastgele sayı) ile XOR işlemine tabi tutulur.
3. XOR işlemi: E\_i = P\_i ⊕ K

**Örnek**

DNA: GAAG → ASCII: 71 65 65 71

XOR key: 123

Şifreli: 60 58 58 60

**Fonksiyonel Gösterim**

Let:

* P = {p\_1, ..., p\_n}: DNA sonrası ASCII byte dizisi
* K: XOR anahtarı

Ei=Pi⊕KE\_i = P\_i \oplus K

Çıktı: XOR şifrelenmiş byte dizisi E

**2.3. Logistic Map ile Kaotik Şifreleme**

**Teorik Arka Plan**

Logistic map, kaotik sistemlerin en bilinen örneklerinden biridir. Aşağıdaki denklemle ifade edilir:

Xn+1=rXn(1−Xn)X\_{n+1} = r X\_n (1 - X\_n)

Burada:

* x\_0: [0.4, 0.6] aralığında bir rastgele başlangıç değeri
* r: [3.9, 4.0] aralığında bir kaos katsayısıdır

Oluşan X dizisi, her bir değer 256 ile çarpılıp tamsayıya dönüştürülerek byte dizisi haline getirilir.

**Uygulama Aşaması**

1. x0 ve r rastgele oluşturulur.
2. Görüntü boyutu kadar (n) iterasyon yapılır.
3. X\_n dizisi hesaplanarak int(X\_n \* 256) % 256 biçiminde sayısallaştırılır.
4. XOR ile elde edilen her byte değeri ayrıca Logistic dizisi ile XOR işlemine tabi tutulur.

**Örnek**

xor\_key ile şifreli: 60

logistic\_key[i]: 157

final: 60 ⊕ 157 = 161

**Fonksiyonel Gösterim**

Let:

* E = {e\_1, ..., e\_n}: XOR sonrası byte dizisi
* L = logistic\_map(x0, r, n)

Ci=Ei⊕LiC\_i = E\_i \oplus L\_i

Çıktı: Nihai şifreli görüntü verisi C

**2.4. RSA ile Metadata Şifreleme**

**Teorik Arka Plan**

RSA, büyük asal sayıların çarpımına dayalı bir asimetrik şifreleme algoritmasıdır. Bu sistemde, şifreleme ve çözme işlemleri farklı anahtarlarla yapılır:

* Public Key: Şifreleme için
* Private Key: Çözümleme için

Bu projede RSA sadece **metadata** (anahtarlar ve parametreler) için kullanılmıştır.

**Metadata İçeriği**

{

"xor\_key": 123,

"x0": 0.489,

"r": 3.976,

"shape": (H, W, 3),

"dna\_rule": 4

}

**Uygulama Aşaması**

1. RSA 2048-bit anahtar çifti üretilir.
2. Yukarıdaki metadata pickle ile serileştirilir.
3. Public key ile metadata şifrelenir.
4. Private key .pem dosyasına yazılır.

**Fonksiyonel Gösterim**

Let:

* M: Metadata verisi
* PK: Public Key
* C\_m: Şifrelenmiş metadata

Cm=RSA\_Encrypt(PK,serialize(M))C\_m = RSA\\_Encrypt(PK, serialize(M))

Çıktı: Şifrelenmiş metadata C\_m ve private.pem

**2.5. Şifreleme Akışı**

1. RGB görüntü yüklenir.
2. Flatten edilerek 1D byte dizisine çevrilir.
3. DNA kodlama yapılır.
4. DNA → ASCII → XOR şifreleme
5. XOR sonucu, Logistic dizisi ile tekrar XOR’lanır.
6. Metadata yapısı hazırlanır ve RSA ile şifrelenir.
7. Şifreli görüntü ve metadata .bin dosyasına kaydedilir.
8. RSA public ve private anahtarları .pem dosyaları olarak saklanır.

Her işlem, bir öncekinden bağımsız olarak çözülemez yapıda olup; sadece şifrelenmiş veriyi değil, şifreleme adımlarını da koruma altına almaktadır.