PROJE TASARIM RAPORU: MOD-64 ŞİFRELEME ALGORİTMASI

Furkan Akkamış **Tarih:** 12.12.2025

1. Algoritma Künyesi

- **Algoritma Adı:** MOD-64
- **Algoritma Türü:** Simetrik Blok Şifreleme (Feistel Ağı Mimarisi)
- **Blok Boyutu:** 64-bit (32-bit Sol Blok + 32-bit Sağ Blok)
- **Anahtar Boyutu:** 64-bit
- **Tur Sayısı:** 8 Tur

2. Tasarım Felsefesi ve Gerekçe

MOD-64 algoritması tasarlanırken, kriptografinin temel prensipleri olan **Karıştırma (Confusion)** ve **Yayılma (Diffusion)**ilkeleri gözetilmiştir.

- **Neden Feistel Ağı?** Şifreleme ve deşifreleme işlemleri için aynı yapının kullanılmasına olanak tanıdığı için seçilmiştir. Bu yapı, kodlama karmaşıklığını azaltırken, kullanılan "F-Fonksiyonu"nun tersine çevrilebilir olma zorunluluğunu ortadan kaldırır. Bu da daha karmaşık matematiksel fonksiyonların güvenle kullanılmasını sağlar.
- **Saldırı Direnci:**
  - **Frekans Analizi:** Standart bir XOR şifrelemesi harf frekanslarını korurken, bu tasarımda kullanılan Modüler S-Kutusu ((5x+3)(mod16)), harf dağılımını lineer olmayan bir şekilde değiştirerek frekans analizini zorlaştırır.
  - **Brute-Force (Kaba Kuvvet):** 64-bit anahtar uzayı, proje kapsamındaki basit saldırı senaryoları için yeterli güvenlik marjını sağlar.

3. Matematiksel Model

Algoritma, 64 bitlik düz metin bloğunu Li (Sol 32-bit) ve Ri (Sağ 32-bit) olarak ikiye böler. Her tur (i) için şifreleme denklemi şöyledir:

Ana Döngü Denklemi:

$$L_{i+1} = R_i$$

$$R_{i+1} = L_i \oplus F(R_i, K_i)$$

Burada F fonksiyonu (Karıştırma Fonksiyonu) şu adımlardan oluşur:

1. **Anahtar Karışımı (XOR):**
2. $T = R_i \oplus K_i$

3. **S-Kutusu (İkame):** T değeri 4 bitlik parçalara (x) bölünür ve şu dönüşüm uygulanır:
4. $S(x)=(5x+3) \bmod 16$
5. **Permütasyon (Bit Kaydırma):** Sonuç sola 5 bit dairesel kaydırılır.
6. $Output=(S(x) \ll 5)$

4. Akış Şeması (Flowchart)

Algoritmanın genel çalışma prensibi aşağıdaki Feistel yapısına dayanmaktadır.

Rapor için Yazılı Akış Tarifi:

1. **Başlangıç:** 64-bit Düz Metin ve Anahtar alınır.
2. **Bölme:** Metin, Sol (L0) ve Sağ (R0) olarak ikiye ayrılır.
3. **Döngü (Tur 1'den 8'e kadar):**
   - R0 kopyalanır ve Anahtar ile XOR işlemine girer.
   - Çıkan sonuç S-Kutusundan (modüler formül) geçirilir.
   - Sonuç sola kaydırılır (Permütasyon).
   - Elde edilen karışık veri, L0 ile XOR yapılır.
   - Eski R0, yeni L1 olur; Yeni hesaplanan değer R1 olur.
4. **Sonuç:** Sol ve Sağ blok birleştirilir (Swap yapılmaz).

5. Elle İzleme (Manual Trace) - Örnek Senaryo

Algoritmanın çalışırlığını ispatlamak için **"KALE"** metni ve **"AN"** anahtarı kullanılarak 1 turluk işlem aşağıda gösterilmiştir.

Veriler:

- **Düz Metin:** "KALE"
- **Anahtar:** "AN"

Adım 5.1: Bloklara Ayırma (ASCII Değerleri)

Metin ortadan ikiye bölünür:

- **Sol Blok (L0):** "KA" → K(75), A(65)
- **Sağ Blok (R0):** "LE" → L(76), E(69)
- **Anahtar (K0):** "AN" → A(65), N(78)

Adım 5.2: F-Fonksiyonu İşlemleri

Sadece Sağ Blok (R0) işleme girer.

A) XOR İşlemi ($R0 \oplus K0$):

- L(76) ⊕ A(65):
  - 01001100 XOR 01000001 = 00001101 (Decimal: **13**)
- E(69) ⊕ N(78):
  - 01000101 XOR 01001110 = 00001011 (Decimal: **11**)

B) S-Kutusu Uygulaması ((5x+3)mod16): Elde edilen sonuçların (13 ve 11) formülden geçirilmesi:

- Girdi **13** için: (5×13+3)=68. → 68mod16=4
- Girdi **11** için: (5×11+3)=58. → 58mod16=10
- *F-Fonksiyonu Çıktısı:* [4, 10]

Adım 5.3: Feistel Birleştirme (XOR)

F-Fonksiyonu çıktısı, bekleyen Sol Blok (L0) ile işleme girer.

- Eski Sol **K (75)** ⊕ Çıktı **4** = **79** (ASCII: 'O')
- Eski Sol **A (65)** ⊕ Çıktı **10** = **75** (ASCII: 'K')

Sonuç (1. Tur Sonunda)

- **Yeni Sol (L1):** Eski Sağ Blok → **"LE"**
- **Yeni Sağ (R1):** Yeni Hesaplanan → **"OK"**
- **Oluşan Ara Metin:** "LEOK"

Bu işlem, belirlenen tur sayısı kadar tekrarlanarak nihai şifreli metne ulaşılır.