## 1. GİRİŞ

Bilgi güvenliği sistemlerinde (anahtar üretimi, şifreleme protokolleri ve dijital imzalar gibi) birçok alanda tahmin edilemezlik hayati önem taşır. Bu çalışmada, deterministik bilgisayarların ürettiği "sözde rastgele" sayıların güvenliğini sağlamak amacıyla **Blum Blum Shub (BBS)** algoritması seçilmiştir.

Standart algoritmaların aksine, BBS algoritması **Kriptografik Olarak Güvenli Sözde Rastgele Sayı Üreteci (CSPRNG)** sınıfına girer ve güvenliğini matematiksel zorluk problemlerinden alır.

## 2. MATEMATİKSEL ALTYAPI VE ALGORİTMA

Blum Blum Shub algoritması, sayılar teorisindeki "Karesel Kalanlar" prensibine dayanır. Algoritmanın güvenliği, çok büyük iki asal sayının çarpımından oluşan bir modülün çarpanlarına ayrılmasının zorluğuna bağlıdır.

**Algoritma Parametreleri:**

- **p ve q:** İki büyük asal sayı. Bu asallar $p \equiv 3 \pmod 4$ şartını sağlamalıdır.
- **M (Modül):** $M = p \cdot q$.
- **$x_o$ (Seed/Tohum):** M ile aralarında asal olan rastgele bir başlangıç sayısı.

**Üretim Formülü:** Sayı dizisinin bir sonraki elemanı şu formülle elde edilir:

$$x_{n+1} = x_n^2 \bmod M$$

Bu işlem sonucunda elde edilen sayının en düşük anlamlı biti veya parite biti alınarak rastgele bit dizileri oluşturulur.

Çarpanların büyüklüğü arttıkça, çözüm için gereken süre ve işlem gücü **üstel** olarak artar. Bu asimetri (çarpmanın kolay, bölmenin imkansız olması), modern kriptografinin ve BBS algoritmasının temel güvenlik duvarıdır.

## 3. SÖZDE KOD (PSEUDOCODE)

Algoritmanın mantıksal akışı aşağıda sunulmuştur:

```
BAŞLA
    GİRDİ: p, q (Büyük Asal Sayılar), seed (Tohum)
```

```
    ADIM 1: Modül Hesapla
       M = p * q

    ADIM 2: Başlangıç Durumunu (State) Ayarla
       state = (seed * seed) MOD M

    FONKSİYON bit_uret():
       state = (state * state) MOD M
       cikis_biti = state MOD 2 (Tek/Çift kontrolü)
       DÖNDÜR cikis_biti

    DÖNGÜ (İstenen bit sayısı kadar):
       bit = bit_uret()
       YAZDIR bit
BİTİR
```

## 4. KOD ANALİZİ VE UYGULAMA

Çalışma kapsamında geliştirilen Python uygulamasında, BBS algoritmasının temel çalışma prensibi simüle edilmiştir.

(Kaynak kodlara şu adresten ulaşılabilir: github.com/kullaniciadi/repo)

Uygulamada p ve q değerleri, algoritmanın doğruluğunu test etmek amacıyla örnek asal sayılar olarak seçilmiştir. next() fonksiyonu her çağrıldığında karesel mod alma işlemi uygulanarak durum güncellenmekte ve kriptografik açıdan güçlü bitler üretilmektedir.

## 5. GÜVENLİK ANALİZİ VE DEĞERLENDİRME

BBS algoritmasının seçilme nedeni, diğer basit üreteçlere kıyasla sahip olduğu şu üstünlüklerdir:

1. **Geriye Dönük Tahmin Zorluğu:** BBS'de bir sonraki sayıyı tahmin etmek, M sayısını çarpanlarına ayırmak kadar zordur. M sayısı yeterince büyük seçildiğinde (RSA şifrelemesinde olduğu gibi) bu işlem hesaplama gücü açısından imkansıza yakındır.
2. **İstatistiksel Başarım:** p ve q sayılarının özel seçimi, üretilen döngünün uzun olmasını garanti eder.

**Dezavantaj Analizi:** Algoritmanın en büyük dezavantajı hesaplama maliyetidir. Her adımda modüler üs alma işlemi yapıldığı için, basit aritmetik işlemler yapan algoritmalara göre daha yavaştır. Bu nedenle BBS, hızın değil güvenliğin ön planda olduğu (anahtar üretimi gibi) yerlerde tercih edilmelidir.