T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Kriptoloji Dersi Proje Ödevi

Ders Hocası	**:**	**Ünal Çavuşoğlu**
Grup	**:**	**1-A**

BEYZANUR ODABAŞ/ B201210100

ZEYNEP İNAN/B201210004

TEVHİDE DEMET DİRİK /B191210064

1.1 LİTERATÜR TARAMASI

Literatürdeki popüler blok şifreleme algoritmaları incelenmiştir. Bu incelemeler DES,3DES,AES,Blowfish,Twofish,RC5,IDEA,Serpent,Camellia,CAST-128 algoritmaları üzerinde yapılmıştır.

DES

"Des" (Data Encryption Standard), veri şifreleme standardı anlamına gelir. DES, 1970'lerin sonlarına doğru IBM tarafından geliştirilen ve daha sonra Amerikan Ulusal Standartlar Enstitüsü (ANSI) tarafından federal verilerin şifrelenmesi için standart olarak kabul edilen bir şifreleme algoritmasıdır. DES, öncelikle elektronik veri iletiminde ve depolanmasında güvenliği sağlamak amacıyla kullanılmıştır. DES, 56-bit uzunluğunda anahtarları kullanır.
64-bit bloklar halinde veri işler. Her bir 64-bitlik blok, şifreleme veya deşifreleme işlemine tabi tutulur. DES, Feistel ağı yapısını kullanır. Bu yapı, anahtarın yarıları üzerinde bağımsız işlemler gerçekleştirerek veriyi şifreler.

Avantajları:

DES, özellikle donanım üzerinde hızlı çalışan bir şifreleme algoritmasıdır. Geçmişte geniş bir kullanım alanına sahipti ve birçok sistemde standart şifreleme protokolü olarak kabul edildi.

Dezavantajları:

Des şu anda kabul edilen güvenlik standartlarına göre oldukça kısa bir anahtar uzunluğudur.
Bu nedenle DES'in 56-bitlik anahtar uzunluğu güvenlik açısından yetersiz ve brute force saldırılarına karşı savunmasız hale geldi. Paralelleştirme açısından sınırlıdır. Güvenli bir şifreleme sistemi için kritik öneme sahip olan anahtarların güvenli bir şekilde dağıtılması ve paylaşılması DES'te kullanılamıyor. Anahtar yönetimi özellikle anahtarın elle değiştirilmesi gereken durumlar için DES günümüz standartlarına uygun değildir.

AES

AES (Advanced Encryption Standard), şu anda dünyanın dört bir yanında geniş bir kullanım bulan, güçlü bir simetrik şifreleme algoritmasıdır. Bu algoritma, önceden yaygın olarak kullanılan DES'in yerine geçmiş ve modern güvenlik gereksinimlerini karşılamak üzere tasarlanmıştır.
AES, 128, 192 veya 256 bit uzunluğunda anahtarları destekler. Bu, özellikle uzun anahtarlar kullanıldığında yüksek güvenlik sağlar. AES, 128-bit bloklar halinde veriyi işler. Bu, hem hızlı işlem sağlar hem de güvenli bir şifreleme düzeyi sunar. AES, bir dizi matematiksel dönüşüm (substitution-permutation network) kullanır. Bu, kriptografik güvenliği artırmaya yönelik bir dizi işlemdir. AES, çeşitli kriptoanalitik saldırılara karşı dirençlidir ve güvenliği uzun yıllar boyunca kanıtlanmıştır.

Avantajları:

AES, yüksek güvenlik sağlamak için tasarlanmıştır ve günümüz kripto analiz tekniklerine karşı dirençlidir. Hem yazılım hem de donanım üzerinde hızlı çalışabilir. Bu, geniş bir uygulama yelpazesi için uygundur. AES, NIST tarafından belirlenen bir standart olarak dünya genelinde kabul görmüştür. Bu nedenle, çeşitli uygulamalarda ve endüstrilerde yaygın olarak kullanılmaktadır.

3DES

Triple DES (3DES), DES (Data Encryption Standard) adlı eski bir şifreleme algoritmasının güvenliğini artırmak amacıyla geliştirilmiş bir şifreleme algoritmasıdır. 3DES, DES'in güvenlik zayıflıklarını kapatmak için üç kez uygulama ilkesine dayanır. Bu, daha yüksek bir güvenlik düzeyi sağlar.

- İlk DES Şifreleme Aşaması: Açık metin, ilk anahtarla DES algoritması kullanılarak şifrelenir.

- İkinci DES Deşifreleme Aşaması: İlk şifreleme aşamasının çıktısı, ikinci anahtar kullanılarak DES algoritması ile deşifre edilir. Bu adım şifreleme değil, deşifreleme işlemidir.

- Üçüncü DES Şifreleme Aşaması: İkinci deşifreleme aşamasının çıktısı, üçüncü anahtar kullanılarak tekrar şifrelenir. Bu adım şifreleme işlemidir. 3DES, DES'in güvenlik zayıflıklarını kapatmak için üç kez uygulama ilkesine dayanır. Bu, daha yüksek bir güvenlik düzeyi sağlar.

Avantajları:

1. **Güvenlik Artışı:** Triple DES, orijinal DES algoritmasının anahtar uzunluğunu artırarak güvenlik seviyesini yükseltmeye çalışır. Orijinal DES 56-bit anahtarlara sahiptirken, Triple DES, genellikle 168-bit anahtar uzunluğuyla kullanılır (3 adet 56-bit anahtarın kullanılmasıyla elde edilir).

2. **Uyumlu Geçiş:** Triple DES, orijinal DES ile uyumlu olduğu için mevcut sistemlerde kullanılabilir. Yani, DES ile iletişim kuran sistemler, Triple DES kullanıldığında uyumluluk sorunu yaşamaz.

3. **Makul Performans:** Triple DES, önceki DES algoritmasına göre daha güvenli olmasına rağmen, yeterli anahtar uzunluğuyla (168-bit) kullanıldığında makul bir performans sunabilir.

Dezavantajları:

1. **Yavaşlık:** Triple DES, önceki DES algoritmasına göre daha güvenli olmasına rağmen, üç kez tekrarlanan şifreleme işlemi nedeniyle daha yavaşdır. Özellikle yüksek hacimli veri şifrelemesi gerektiren uygulamalarda performans sorunlarına neden olabilir.

2. **Anahtar Yönetimi:** Anahtar yönetimi, Triple DES'in kullanılması sırasında dikkate alınması gereken bir dezavantajdır. Anahtarlar özenle yönetilmelidir ve güvenlik için düzenli olarak değiştirilmelidir.

3. **Modern Alternatiflere Göre Zayıf Güvenlik:** Günümüzde, daha güçlü ve hızlı şifreleme algoritmaları bulunmaktadır. Özellikle, AES (Advanced Encryption Standard) gibi modern algoritmalar, Triple DES'e göre daha hızlı ve güvenli bir performans sunar. Bu nedenle, yeni uygulamalarda genellikle AES tercih edilmektedir.

Blowfish

Blowfish, Bruce Schneier tarafından 1993 yılında tasarlanan bir simetrik blok şifreleme algoritmasıdır. Blowfish, anahtarlama esnekliği ve hızlı işlem yapma yetenekleri ile bilinir.

Avantajları:

1. **Anahtar Uzunluğu:** Blowfish, 32 bit ile 448 bit arasında değişen anahtar uzunluklarına izin verir. Bu, algoritmanın anahtar uzunluğunu isteğe bağlı olarak ayarlamak ve gelecekteki güvenlik ihtiyaçlarına uyum sağlamak için esneklik sunar.

2. **Performans:** Blowfish, genellikle hızlı bir şifreleme algoritması olarak kabul edilir. Özellikle donanım tabanlı uygulamalarda ve kaynak sınırlı cihazlarda kullanılmak üzere tasarlanmıştır.

3. **Hafif ve Basit:** Blowfish algoritması, karmaşık olmayan bir yapıya sahiptir, bu da hafif olmasını ve uygulama için kolay entegrasyon sağlamasını sağlar.

4. **Geniş Kullanım Alanları:** Blowfish, özellikle güvenli iletişim, veri şifreleme ve diğer uygulamalarda geniş bir kullanım alanına sahiptir. Ayrıca, kriptografik yazılımlar ve protokoller tarafından sıklıkla tercih edilmektedir.

Dezavantajları:

1. **Standart Olarak Kabul Edilmeme:** Blowfish, diğer bazı popüler simetrik şifreleme algoritmaları kadar geniş ölçüde kullanılmamıştır. Bu, özellikle Blowfish'in genellikle bir öneri olarak kalması ve standartlarla geniş bir şekilde kabul edilmemesi anlamına gelir.

2. **Açık Anahtarlı Şifreleme İle İlgilenmez:** Blowfish, temelde bir simetrik şifreleme algoritmasıdır ve açık anahtarlı şifreleme (public-key cryptography) gibi diğer şifreleme ihtiyaçlarına cevap vermez.

3. **Anahtar Değişimi:** Blowfish, anahtar değişimini ele almada diğer bazı modern algoritmalar kadar etkili olmayabilir. Anahtar yönetimi önemli bir kriptografik zorluktur ve Blowfish bu konuda bazı sınırlamalara sahip olabilir.

Twofish

Twofish, simetrik bir anahtar algoritmasıdır, yani aynı anahtar hem şifreleme hem de şifre çözme işlemlerinde kullanılır. Anahtar uzunluğunu 128, 192 veya 256 bit olarak kabul eder.128-bit blok boyutu vardır.

Avantajları:

1. **Anahtar Esnekliği:** Twofish, kullanıcılara 128, 192 veya 256 bit gibi farklı anahtar uzunlukları seçme esnekliği sunar. Bu, güvenlik ihtiyaçlarına uygun olarak anahtar uzunluğunu belirlemenizi sağlar.

2. **Performans:** Twofish, hızlı bir şifreleme algoritması olarak bilinir. Donanım tabanlı uygulamalarda ve kaynak sınırlı sistemlerde etkili bir performans sergileyebilir.

3. **Anahtar Bağımsızlığı:** Bloklar arasında bağımsızlık özelliği, algoritmanın güvenlik özelliklerini artırabilir ve diferansiyal saldırılara karşı direncini güçlendirebilir.

4. **Uyumluluk:** Twofish, birçok platformda kullanılabilir ve birçok programlama dili ve ortamında uygulanabilir.

Dezavantajları:

1. **Standardizasyon:** Twofish, standartlarla geniş bir şekilde kabul edilmemiştir. Bu, algoritmanın popülerlik ve güvenilirlik açısından bazı sınırlamalara sahip olmasına neden olabilir.

2. **Yavaş Anahtar Değişimi**: Anahtar değişimi işlemi Twofish'te biraz yavaş olabilir, bu da anahtar yönetimi konusunda bazı sınırlamalara işaret edebilir.

RC5:

RC5, simetrik bir anahtar algoritmasıdır. Anahtar uzunluğu kullanıcı tarafından seçilebilir ve genellikle 0 ila 2040 bit arasında değişir. RC5'nin blok boyutu da değiştirilebilir ancak genellikle 32 veya 64 bit olarak kullanılır.

Avantajları:

1. **Anahtar Uzunluğu Esnekliği:** RC5, kullanıcılara anahtar uzunluğunu ihtiyaca bağlı olarak seçme esnekliği tanır. Bu, güvenlik seviyesini artırmak veya performansı optimize etmek isteyen kullanıcılara uygun bir çözüm sunar.

2. **Bağımsızlık İlkesi:** Bloklar arasında bağımsızlık, algoritmanın güvenlik direncini artırabilir ve çeşitli saldırılara karşı daha dayanıklı hale getirebilir.

3. **Parametre Ayarlanabilirliği:** Algoritmanın çeşitli parametrelerini ayarlayabilme yeteneği, RC5'i farklı kullanım senaryolarına adapte etme esnekliği sağlar.

4. **Makul Performans:** RC5, genellikle makul bir performansa sahiptir ve donanım kaynaklarından tasarruf sağlayabilir.

Dezavantajları:

1. **Standartlaşma:** RC5, diğer popüler şifreleme algoritmaları kadar geniş bir standart kabul görmemiştir. Bu, güvenilirlik ve geniş bir kullanım alanı konusunda bazı sınırlamalara neden olabilir.

2. **Optimizasyon Gerekliliği:** RC5, performansın optimum düzeye çıkarılabilmesi için iyi bir şekilde optimize edilmelidir. İyi bir optimizasyon olmaksızın, diğer bazı şifreleme algoritmalarına göre daha yavaş olabilir.

IDEA

IDEA (International Data Encryption Algorithm), James L. Massey ve Xuejia Lai tarafından geliştirilmiş bir simetrik anahtarlı blok şifreleme algoritmasıdır. IDEA, 64-bit bloklar üzerinde çalışan ve 128-bit anahtar kullanabilen bir şifreleme algoritmasıdır.

Avantajları:

1. **Güvenlik Seviyesi:** IDEA, tasarımındaki matematiksel karmaşıklık ve anahtar uzunluğu nedeniyle güvenli kabul edilen bir şifreleme algoritmasıdır.

2. **Yaygın Kullanım:** IDEA, özellikle finansal işlemlerde ve güvenli iletişim uygulamalarında kullanılmıştır. Özellikle PGP (Pretty Good Privacy) gibi güvenli e-posta ve dosya şifreleme uygulamalarında kullanım bulmuştur.

3. **Lineer Olmayan Yapı:** IDEA'nın matematiksel yapısı lineer olmayan bir yapıya sahiptir, bu da saldırılara karşı dirençli olmasına yardımcı olabilir.

Dezavantajları:

1. **Anahtar Yönetimi:** IDEA'nın anahtar yönetimi, algoritmanın kullanılmasını karmaşık hale getirebilir. Anahtarların güvenli bir şekilde yönetilmesi önemlidir.

2. **Hız:** IDEA, diğer bazı şifreleme algoritmalarına göre daha yavaş kabul edilebilir, özellikle daha modern ve optimize edilmiş algoritmaların ortaya çıkmasından sonra.

3. **Patent Sorunu:** IDEA'nın başlangıçta patentli olması ve kullanımının sınırlanması, onun geniş çapta kullanılmasını engelleyebilmiştir. Ancak, patent süresi dolmuş durumda.

4. **Anahtar Uzunluğu:** 128-bit anahtar uzunluğu, günümüz standartlarına kıyasla daha kısa kabul edilebilir.

Serpent

Serpent, Bruce Schneier ve diğer kriptografi uzmanları tarafından tasarlanan bir blok şifreleme algoritmasıdır.

Avantajları:

1. **Güvenlik Seviyesi:** Serpent, tasarımındaki matematiksel karmaşıklık ve ayrıştırılmış yapısı nedeniyle güvenli bir şifreleme algoritması olarak kabul edilir.

2. **Yüksek Performans:** Serpent, performans açısından etkili bir şekilde çalışabilir. Bilgi karıştırma operasyonlarına dayalı yapısı, güvenliği artırırken etkili performans sağlamayı amaçlar.

3. **Anahtar Uzunluğu Esnekliği:** Serpent, 128, 192 veya 256-bit anahtar uzunluklarını destekleyerek güvenlik ihtiyaçlarına uygun olarak anahtar uzunluğunu seçme esnekliği sunar.

4. **Geniş Kullanım Alanları:** Serpent, genellikle şifreleme uygulamalarında ve güvenli iletişim sistemlerinde kullanılmıştır.

Dezavantajları:

1. **Yavaş Hareketli Kutular:** Yavaş hareketli kutular nedeniyle, Serpent diğer bazı algoritmalarla kıyaslandığında işlem hızında bir miktar yavaşlık gösterebilir.

2. **Popülerlik:** Serpent, diğer bazı popüler şifreleme algoritmaları kadar geniş bir kabul görmemiştir. Bu, algoritmanın genel olarak daha az kullanılabilir olmasına neden olabilir.

3. **Yavaş Anahtar Değişimi:** Anahtar değişimi işlemi, Serpent'te biraz yavaş olabilir, bu da anahtar yönetimi konusunda bazı sınırlamalara neden olabilir.

Camellia

Camellia, Mitsubishi Electric ve NTT (Nippon Telegraph and Telephone Corporation) tarafından birlikte geliştirilen bir blok şifreleme algoritmasıdır. Camellia, 128, 192 veya 256-bit anahtar uzunluklarına ve 128-bit blok uzunluğuna sahiptir.

Avantajları:

1. **Güvenlik:** Camellia, güvenli bir şifreleme algoritması olarak kabul edilir. Anahtar bağımsızlığı ve diğer kriptografik özellikler, güvenlik seviyesini artırır.

2. **Performans:** Camellia, donanım tabanlı uygulamalarda ve yazılım implementasyonlarında iyi performans sergileyebilir. Özellikle hızlı ve etkili bir şekilde çalışması avantajlıdır.

3. **Standartlaşma:** Camellia, ISO/IEC ve diğer standart organizasyonları tarafından standartlaştırılmış ve önerilmiştir. Özellikle Japon hükümeti tarafından kullanılmaktadır.

4. **Uyumlu Implementasyonlar:** Camellia'nın açık ve standart bir tasarımı olduğu için, farklı platformlarda uyumlu implementasyonlar yapmak daha kolaydır.

Dezavantajları:

1. **Daha Az Popüler:** Camellia, özellikle AES'in (Advanced Encryption Standard) popülerliği karşısında daha az bilinen bir şifreleme algoritmasıdır. Bu, kullanımı ve desteklenirliği etkileyebilir.

2. **Anahtar Yönetimi:** Anahtar yönetimi, her şifreleme algoritması için önemli bir konudur. Camellia'nın anahtar yönetimi konusundaki zorlukları, kullanıcıların dikkate alması gereken bir dezavantaj olabilir.

1.2 TEMEL PRENSİPLERİN BELİRLENMESİ

Blok Boyutu:

Blok boyutu, şifreleme algoritmasının güvenlik seviyesini etkileyebilir. Genelde, daha büyük blok boyutları daha güçlü bir direnç sağlayabilir, ancak bununla birlikte performansı etkileyebilir.Blok boyutu, algoritmanın performansını etkiler. Daha küçük blok boyutları genellikle daha hızlı işleme imkanı sağlar, ancak güvenlik seviyesi azalabilir. Daha büyük blok boyutları ise daha güvenli olabilir, ancak işlemleri daha yavaş hale getirebilir.Şifreleme algoritması hangi tür uygulamalar için kullanılacaksa, blok boyutu buna uygun olmalıdır. Örneğin, bir veritabanı şifreleme uygulaması için farklı bir blok boyutu seçilebilirken, bir metin mesajı şifreleme uygulaması için başka bir blok boyutu daha uygun olabilir.Şifreleme algoritmasının tasarımında kullanılan yapısal özellikler de blok boyutunu etkileyebilir.
Örneğin, Feistel yapısında genellikle daha küçük blok boyutları kullanılır.

Şifreleme algoritması genellikle belirli bir yürütme ortamında çalışacaksa (örneğin, donanım tabanlı uygulamalar), blok boyutu bu ortama uygun olmalıdır. Blok boyutunun güvenlik analizi üzerinde etkisi vardır. İyi bir şifreleme algoritması tasarlarken, blok boyutunun matematiksel analizlerle değerlendirilmesi önemlidir. Genel olarak, blok boyutu bir dengeleme eylemidir. Daha büyük blok boyutları genellikle daha güvenli olabilir, ancak performansı etkileyebilir. Tasarım sürecinde, belirli uygulama gereksinimlerini, güvenlik ihtiyaçlarını ve performans hedeflerini dikkate alarak blok boyutunu seçmek önemlidir. Ayrıca, kriptografi topluluğundan ve standartlardan gelen rehberliklere de başvurmak faydalı olabilir.

Algoritmada 128 bitlik blok boyutu kullanılmıştır. 128 bit, günümüzde genellikle kabul edilen bir güvenlik seviyesidir. Daha büyük blok boyutları genellikle daha fazla güvenlik sağlayabilir, ancak 128 bit, birçok uygulama için yeterli güvenlik düzeyini temsil eder. 128-bit blok boyutları, modern işlemcilerle uyumlu bir performans sağlar. Çoğu donanım ve yazılım altyapısı, 128 bitlik işlemleri hızlı bir şekilde gerçekleştirecek şekilde optimize edilmiştir. Blok boyutu arttıkça, şifreleme ve çözme işlemleri karmaşık hale gelebilir ve daha fazla kaynak gerektirebilir. 128 bit, hem güvenlik hem de performans açısından genellikle etkili bir denge sağlar. Birçok şifreleme standardı, özellikle Advanced Encryption Standard (AES), 128-bit blok boyutunu kullanır. Bu standartlar, güvenilirliği ve uyumluluğu artırmak için 128 bit blok boyutunu önerir. 128 bit, kriptografik analizlere dayalı olarak güvenli kabul edilen bir blok boyutudur. Bu, şifreleme algoritmalarının kriptanalistler tarafından incelendiğinde, saldırılara karşı dirençli olduğunu gösterir. 128-bit blok boyutları, özellikle donanım tabanlı uygulamalarda etkili bir şekilde çalışabilir. Bu, güvenlik sistemleri ve cihazları için uygun bir seçenek yapar.

Anahtar Boyutu:

Şifreleme algoritması tasarlarken anahtar boyutu seçimi, algoritmanın güvenlik düzeyini doğrudan etkileyen kritik bir karardır. Anahtar boyutunun belirlenmesi, bir dizi faktöre dayanmalı ve algoritmanın dayanıklılığı, direnci ve genel güvenliği göz önüne alınarak yapılmalıdır. Anahtar boyutu, algoritmanın güvenlik seviyesini belirler. Genellikle daha uzun anahtarlar, daha yüksek güvenlik düzeyleri sağlar. Ancak, anahtar uzunluğu arttıkça işleme süresi de artabilir. Artan anahtar boyutu, genellikle daha fazla işlem gücü gerektirir. Anahtar uzunluğu ile işlem hızı arasında bir denge kurulmalıdır. Çok uzun anahtarlar, işlemleri yavaşlatabilir ve sistem kaynaklarını daha fazla tüketebilir. Anahtar boyutu seçilirken, gelecekteki bilgisayar gücü gelişmeleri de göz önüne alınmalıdır. Şu anki güvenliği sağlamak için günümüz bilgisayar gücü yeterli olabilir, ancak algoritmanın uzun vadeli direncini düşünmek önemlidir. Anahtar uzunluğu, kriptoanalitik saldırılara karşı direnci belirler. Kriptoanalistlerin geliştirdiği yeni saldırı tekniklerine karşı direnç sağlamak amacıyla güncel bilimsel ve matematiksel çalışmalara göre belirlenmelidir.

Algoritmada 128 bitlik anahtar kullanılmıştır. 128-bitlik anahtarlar, daha uzun anahtar uzunluklarına kıyasla daha hızlı işleme imkanı sağlar. Bu, şifreleme ve çözme işlemlerinin daha verimli bir şekilde gerçekleştirilmesini sağlar. 128-bitlik anahtarlar, hem donanım hem de yazılım tabanlı uygulamalarda genellikle uyumlu bir şekilde çalışabilir. Bu, farklı platformlarda ve cihazlarda kullanım esnekliği sağlar. Birçok kriptografik uygulama ve protokol, 128-bitlik anahtar uzunluğunu temel alır. Örneğin, Advanced Encryption Standard (AES) gibi popüler şifreleme algoritmaları 128-bit anahtarları kullanır. 128-bit anahtarlar, çoğu günlük uygulama ve iletişim için yeterli güvenlik düzeyini sağlar. Özellikle genel kullanıma yönelik uygulamalarda, daha uzun anahtarlar gereksiz olabilir ve performansı düşürebilir.

128-bit anahtar uzunluğu, birçok standart ve öneri tarafından kabul görmüştür. Bu standartlar, anahtar uzunluğunu belirlerken genellikle bir dengeleme sağlar.

Döngü Sayısı:

Şifreleme algoritması tasarlarken döngü sayısı (round count), algoritmanın güvenlik düzeyini ve dayanıklılığını belirleyen önemli bir faktördür. Döngü sayısı, anahtar ile veri arasındaki ilişkiyi karmaşıklaştırarak saldırılara karşı direnci artırır. Ancak, döngü sayısını belirlerken bir denge bulunmalıdır, çünkü fazla döngü, performansı düşürebilir ve kaynakları aşırı kullanabilir. Daha fazla döngü genellikle daha yüksek bir güvenlik düzeyi anlamına gelir, ancak bu aynı zamanda işlemleri daha yavaş hale getirebilir. Güvenlik düzeyini belirlerken belirli bir risk profiline göre değerlendirme yapılmalıdır. Döngü sayısı arttıkça, şifreleme ve çözme işlemleri daha karmaşık hale gelir ve daha fazla kaynak gerektirir. Bu nedenle, döngü sayısı performansı doğrudan etkiler. Uygulama bağlamında kabul edilebilir bir performans düzeyi bulunmalıdır. Döngü sayısının arttırılması, kriptoanalitik saldırılara karşı direnci artırabilir. Ancak, bu sadece döngü sayısını artırmanın tek faktörü değildir; algoritmanın diğer matematiksel ve yapısal özellikleri de kriptoanalitik dirence etki eder. Uygulama genellikle belirli bir donanım veya yazılım platformunda çalışacaksa, döngü sayısı bu platformlarla uyumlu olmalıdır. Donanım tabanlı implementasyonlar genellikle daha hızlı çalışabilir. Belirli sektörlere ve uygulama türlerine yönelik standartlar belirli döngü sayılarını öneriyor olabilir. Bu standartlara uymak, algoritmanın genel kabulünü artırabilir. Döngü sayısının bir ölçüsü, algoritmanın gelecekteki bilgisayar gücü gelişmelerine ve yeni kriptoanalitik saldırılara karşı ne kadar esnek olduğunu belirleyebilir.

Algoritmada döngü sayısı 10 olarak belirlenmiştir. Bu döngü sayısı genellikle daha hızlı işlemler anlamına gelir. Bu, algoritmanın genel performansını artırabilir ve daha hızlı şifreleme ve çözme işlemleri sağlayabilir. algoritmanın kaynakları daha etkin bir şekilde kullanmasına olanak tanır. Özellikle sınırlı kaynaklara sahip sistemlerde (örneğin, mobil cihazlar), bu önemlidir. Enerji tasarrufu sağlayabilir. Mobil cihazlar, IoT (Nesnelerin İnterneti) cihazları gibi enerji kaynakları sınırlı olan sistemlerde bu avantaj önemli olabilir.

Blok Şifreleme:

Algoritmada blok şifreleme kullanılmıştır. Blok şifreleme, kriptografi alanında yaygın olarak kullanılan bir şifreleme türüdür. Bu yöntem, belirli bir blok boyutunda (örneğin, 128 bit) veriyi işler ve genellikle aynı boyutta bir anahtar kullanır. Blok şifreleme, belirli bir blok boyutunda veriyi işleyen bir algoritma türüdür. Her blok, şifreleme veya çözme işlemi için ayrı ayrı işlenir. Blok şifreleme algoritmaları genellikle bir anahtar kullanır. Aynı anahtar, şifreleme ve çözme işlemlerinde kullanılır. Anahtarın değiştirilmesi, farklı blokların farklı şekilde işlenmesini sağlar. Çoğu blok şifreleme algoritması, döngü yapısını kullanarak blokları işler. Döngü yapısı, belirli bir blok boyutundaki veriyi karmaşık hale getirerek güvenliği artırır. Birçok blok şifreleme algoritması, Feistel yapısını benimser. Bu yapıda, blok iki yarıya bölünür ve ardından belirli bir döngüde gerçekleşen permütasyon ve substitüsyon adımları uygulanır. Blok şifreleme algoritmaları, genellikle matematiksel yapıları nedeniyle yarı grup özelliklerine sahiptir. Ayrıca, anahtar bağımsızlığı özelliği sayesinde anahtarın değiştirilmesi, farklı anahtarlarla güvenli bir şekilde çalışmasını sağlar.

Blok şifreleme algoritmaları, genellikle paralel işlem yeteneklerine sahiptir. Bu, aynı anda birden çok bloğun işlenmesine olanak tanır, bu da performansı artırabilir.

Blok şifreleme algoritmaları, geniş bir uygulama yelpazesine sahiptir. Örneğin, veritabanı şifreleme, disk şifreleme, iletişim güvenliği ve daha pek çok alanda kullanılabilirler.
Birçok blok şifreleme algoritması, kriptografik standartlar ve protokoller tarafından desteklenir ve yaygın olarak kabul görmüştür. Bu standartlar, algoritmaların güvenilirliğini ve güvenliğini artırabilir. İyi tasarlanmış ve güvenilir bir blok şifreleme algoritması, güçlü bir güvenlik düzeyi sağlayabilir. Blok şifreleme algoritmaları genellikle çeşitli kriptoanalitik saldırılara karşı dirençlidir. Blok şifreleme algoritmaları genellikle modüler bir yapıya sahiptir, bu da onları farklı uygulamalara entegre etmeyi kolaylaştırır.

1.3 ALGORİTMA TASARIMI

1-Anahtar Üretimi

Anahtar üretiminde PBKDF2HMAC algoritması kullanılmıştır. PBKDF2HMAC, güvenli ve öngörülemeyen anahtarlar üretmek için tasarlanmıştır. Bu, şifrelerin daha güvenli ve dayanıklı olmasını sağlar.PBKDF2HMAC, key stretching olarak bilinen bir teknik kullanır. Bu, saldırılara karşı direnci artırmak amacıyla, kullanıcı tarafından belirtilen başlangıç anahtarından türetilen anahtarın, belirli bir algoritma ve iterasyon sayısı kullanılarak daha uzun bir süreçte türetilmesidir. Bu özellik, saldırılara karşı dayanıklılığı artırarak güvenlik seviyesini yükseltir. PBKDF2HMAC, salting yöntemini destekler. Bu, aynı şifre kullanılarak türetilmiş anahtarların birbirinden farklı olmasını sağlar. Saldırılara karşı daha dirençli olmak için kullanıcıya özgü saltlarla birlikte çalışabilir. PBKDF2HMAC'nin birçok parametresi, kullanıcı tarafından belirlenebilir. Bu, kullanıcılara özel güvenlik gereksinimlerine uygun bir anahtar türetme işlemi gerçekleştirmelerine olanak tanır.

2-S-box Üretimi

S-Box (Substitution Box), blok şifreleme algoritmalarında bir tür substitüsyon (yerine koyma) işlemi uygulamak için kullanılan bir yapıdır. S-Box'lar, giriş bitlerini belirli bir mantıksal kurala göre çıkış bitlerine dönüştüren bir yapıya sahiptir. Bu, şifreleme algoritmasının matematiksel karmaşıklığını artırarak güvenlik düzeyini yükseltmeye yardımcı olur. S-Box'lar genellikle şifreleme algoritmalarının temel yapı taşlarından biridir ve güvenlikleri üzerinde önemli bir etkiye sahiptir. Bu nedenle, güvenli ve etkili bir S-Box üretimi, kriptografik algoritmaların genel güvenliği için kritik bir faktördür.

3-Data Kontrolü

Girilen datanın bit uzunluğu 128’den küçük olduğu taktirde eksik yerler 0 ile doldurulur.

4-Blok Ayırma

Data R0 ve L0 olmak üzere iki parçaya bölünür.

5-XOR İşlemi

Elde edilen 32 bitlik değer, başlangıçta sol yarı ile (L) XOR işlemine tabi tutulur. Bu işlem, yeni sağ yarının (R') oluşturulmasını sağlar.

6-Yeni Blok Oluşturma

Yeni blok oluşturulurken, sol yarının (L) değeri eski sağ yarının (R) değeri olurken, sağ yarının (R') değeri ise Feistel fonksiyonunun çıkış değeri olur. Bu adımlar, 16 tur boyunca tekrarlanır.

7-State Matris Oluşturma

4x4'lük bir matris oluşturulur. Her bir eleman, giriş bloğundan elde edilen 8-bitlik grupların işlenmiş halini içerir.Giriş bloğundaki bitler arasında dolaşmak için Her seferinde 8 bitlik bir grup alınır. 8 bitlik grup, ikili sayıya dönüştürülür ve birleştirilerek bir string haline getirilir. İlk 4 bitlik parça, ondalık bir sayıya çevrilir. İkinci 4 bitlik parça, ondalık bir sayıya çevrilir. Her iki sayı, hex formatına dönüştürülür. Bu hex sayılar, state matrisinin ilgili elemanına yerleştirilir.

8-S-Box İşlemleri

Bu adım, bir state matrisindeki her bir elemanı S-Box olarak adlandırılan bir matrisle değiştirir. Bunun için state matrisinin ilgili elemanından bir hexadecimal string alınır. Hexadecimal string, ondalık bir sayıya çevrilir. Bu değer, S-Box matrisindeki satırı belirlemek için kullanılır. Belirlenen satır ve sütundan S-Box matrisindeki ilgili değer alınır. Yeni değer, iki karakterlik bir hexadecimal stringe dönüştürülür. '02x' formatı, her zaman iki karakterlik bir çıktı üretecektir. State matrisindeki ilgili eleman, S-Box ile değiştirilmiş değerle güncellenir.

9-Add Key

Giriş bloğundaki her bir byte'ı hexadecimalden ikili sayıya çevirir. Sonuç, bir ikili string olan içinde birleştirilir. İkili stringi ondalık bir sayıya çevrilir. Bu, giriş bloğunu temsil eden bir tamsayıdır. Anahtar blokları üzerinde bir döngü başlatılır. Her bir blok, bir öncekine eklenerek bir tamsayı oluşturulur. Her bir anahtar bloğu, bir öncekine eklenirken bir byte sola kaydırılır ve ardından bloğun içeriği eklenir.Giriş bloğu ile anahtarın XOR'lanması yapılır. XOR işlemi sonrası elde edilen sonuç tamsayısı, 32 karakterlik bir ikili stringe dönüştürülür. '032x' formatı, 32 karakter uzunluğunda bir hexadecimal string oluşturur.

10-Performans Ölçümleri

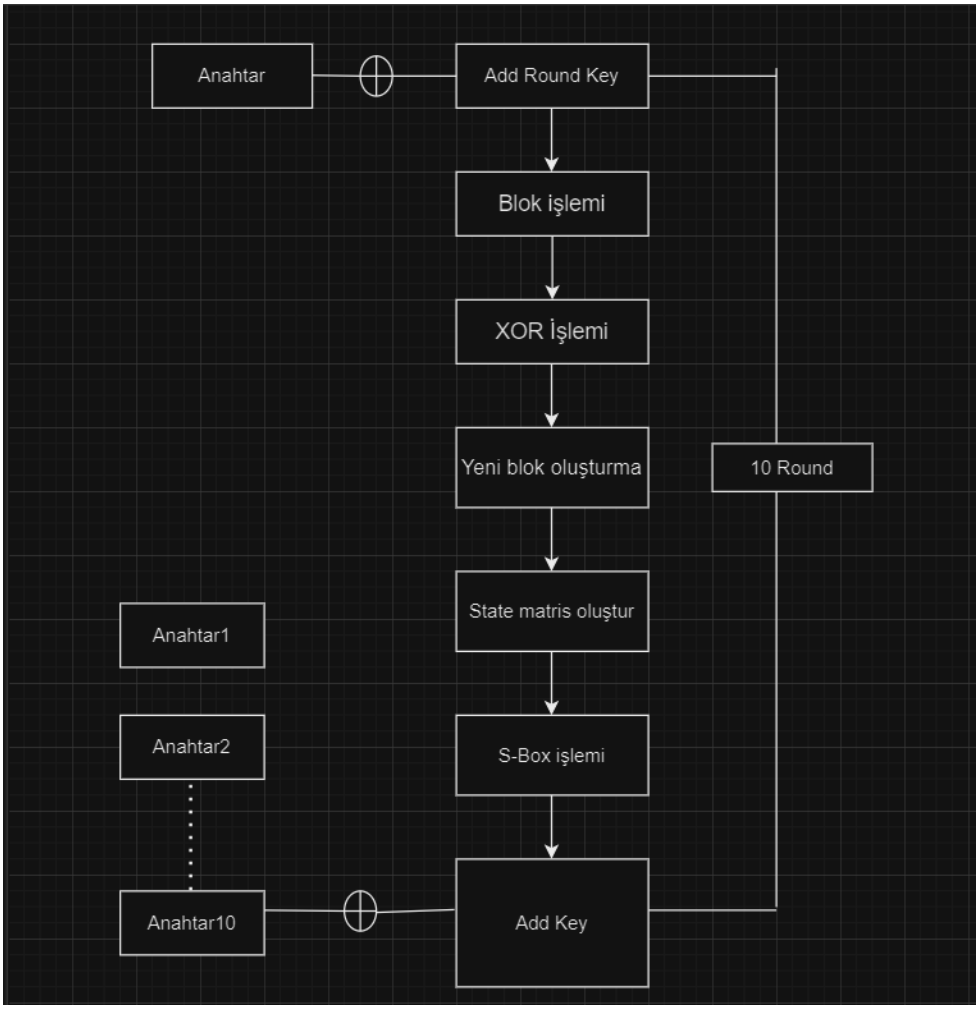
Algoritmanın anahtar üretme süresi, bellek kullanımı ve hızı hesaplanmıştır.

1.4 Anahtar Üretimi

Anahtar türetme işlemi GenerateKey sınıfı içinde gerçekleştiriliyor.Bu sınıf, bir şifre ve gerekirse özelleştirilmiş bir salt kullanarak anahtar türetme işlemini gerçekleştirir.
Kullanıcıdan alınan şifre, salt uzunluğu, anahtar boyutu ve iterasyon sayısı gibi parametreleri sınıf özelliklerine atanır. “generate_salt” fonksiyonu ile sistem saatini kullanarak rastgele bir salt oluşturur. Salt’ın bir kısmı sistem saatinden elde edilirken, geri kalan kısmı rastgele baytlarla doldurulur. “key_derivation” fonksiyonu PBKDF2HMAC algoritmasını kullanarak anahtar türetilir. Bu fonksiyon, şifre ve tuz kullanarak anahtarı türetir. “generate_key” fonksiyon, salt oluşturma ve anahtar türetme işlemlerini başlatır. Bu aşamada süre de hesaplanır ve sınıfın key_generation_time özelliğine atanır. Anahtar türetme işlemi sırasında PBKDF2HMAC algoritması kullanılır. Bu algoritma, şifreleme anahtarını türetirken saldırılara karşı daha güvenli olmak için kullanıcı tarafından belirtilen başlangıç anahtarından türetilen anahtarın belirli bir algoritma ve iterasyon sayısı kullanılarak daha uzun bir süreçte türetilmesini sağlar.

Anahtar türetme işlemi tamamlandığında, elde edilen anahtar ve salt, sınıfın key ve salt özelliklerine atanır ve bu değerler, şifreleme ve çözme işlemlerinde kullanılır.

1.5 BLOCK DİYAGRAMI

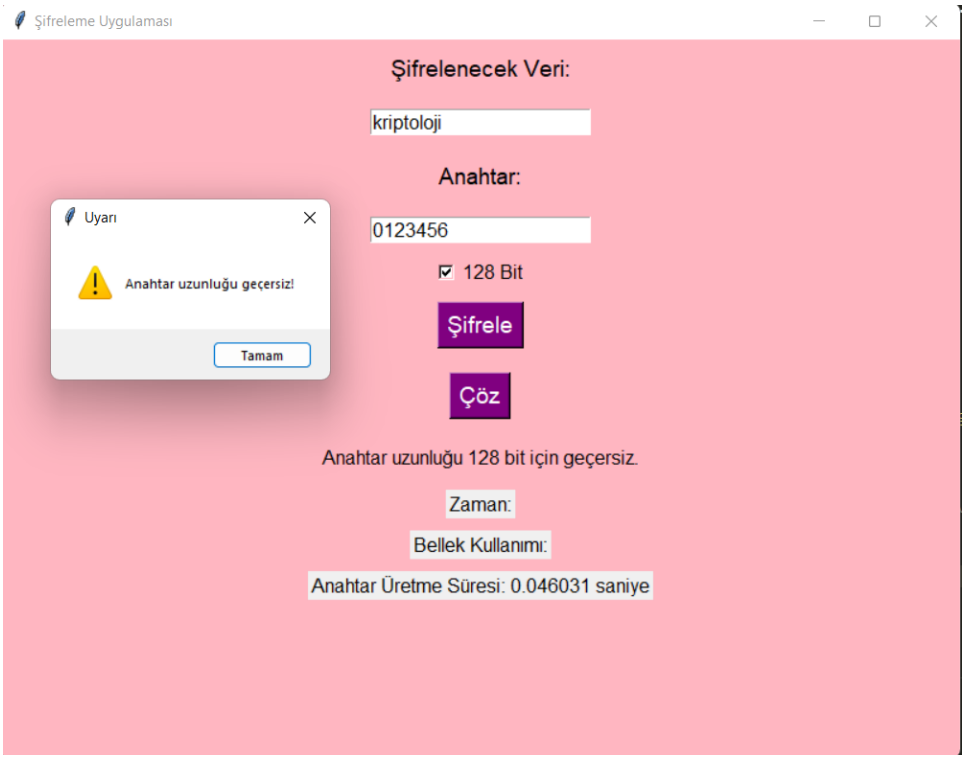


1.6 PROJE ARAYÜZÜ

Projede kullanıcıdan “şifrelenecek veri” texti içerisinde şifrelemek istediği veriyi almaktayız. Daha sonra şifreleme işleminde 128 bit anahtar boyutu kullandığımız için bu uzunlukta bir anahtar girmesini bekliyoruz. Eğer daha küçük veya büyük anahtar boyutu girerse bunu kontrol ederek uyarı mesajı veriyoruz.

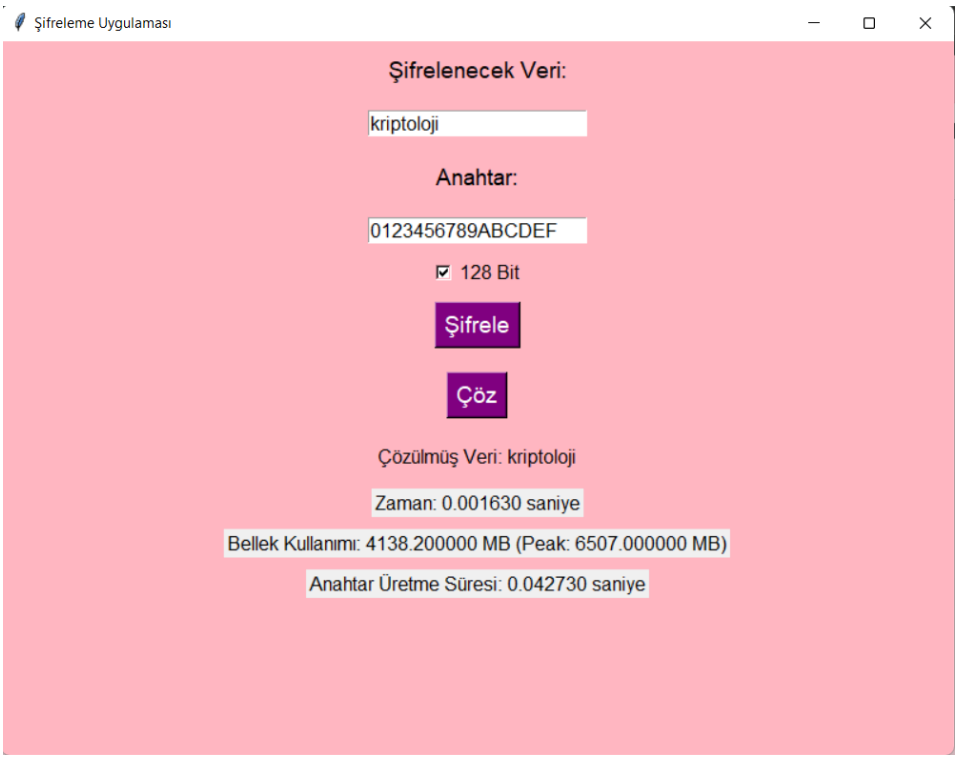
Şifreleme İşlemi :

Kullanıcı 128 Bit Anahtar Boyutu Girmezse Hata Mesajı :



Deşifreleme İşlemi :

Bu kısımda ise şifreleme kısmında yaptığımız adımları tersine uygulayarak çzöe işlemi gerçekleştiriyoruz. Kullanıcı “çöz” butonuna tıkladığı zaman şifreleme işleminde kullanılan aynı anahtar ile deşifreleme işlemi gerçekleştirilir.



1.7 KAYNAKÇA

[1]. https://dergipark.org.tr/tr/pub/iaud/issue/30074/324620
[2]. https://siberkuvvet.com/kutuphane/oku/des-algoritmasi-ve-adim-adim-ornekleri
[3]. ORDU L., AES Algoritmasının FPGA Üzerinde Gerçeklenmesi ve Yan Kanal Analizi Saldırılarına Karşı Güçlendirilmesi, Y.Lisans, Đstanbul Teknik Üniversitesi ,Fen Bilimleri Enstitüsü, 2006.
[4] SMART N., Cryptography: An Introduction, McGraw – Hill Companies, Inc, 2003.
[5]. YERLĐKAYA, T., Şifreleme Algoritmalarının Analizi, Doktora, Trakya Üniversitesi, Bilgisayar Mühendisliği Bölümü, Sf. 8, 22-27, 2006.
[6]. YILDIRIM, M., DES ve DES Benzeri Şifreleme Sistemlerinin Diferansiyel Kripto Analizi, Y.Lisans, Đstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, sf.8-13, 1995.
[7].https://acikerisim.sakarya.edu.tr/bitstream/handle/20.500.12619/80357/T04682.pdf?sequence=1
[8]. http://tr.wikipedia.org/wiki/Blowfish, Ağustos 2010