İSTANBUL TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM FAKÜLTESİ

TEK KULLANIMLIK PAROLA ÜRETEN SİSTEM

Bitirme Ödevi

Semih Dinçer

040060191

Bölüm: Bilgisayar Mühendisliği

Danışman: Prof. Dr. Eşref ADALI

Haziran 2013

İSTANBUL TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM FAKÜLTESİ

TEK KULLANIMLIK PAROLA ÜRETEN SİSTEM

Bitirme Ödevi

Semih Dinçer

040060191

Bölüm: Bilgisayar Mühendisliği

Danışman: Prof. Dr. Eşref ADALI

Haziran 2013

Özgünlük Bildirisi

1. Bu çalışmada, başka kaynaklardan yapılan tüm alıntıların, ilgili kaynaklar referans gösterilerek açıkça belirtildiğini,
2. Alıntılar dışındaki bölümlerin, özellikle projenin ana konusunu oluşturan teorik çalışmaların ve yazılım/donanımın benim tarafımdan yapıldığını bildiririm.

İstanbul, 07/04/2013

Semih Dinçer

Bu çalışmada deney düzeneğinin gerçekleştirilmesi konusunda bana çok değerli desteklerini veren Araştırma Görevlisi ***Hasan Ünlü****'ye ve bitirme projesi boyunca bana yardımcı olan danışman hocam* ***Prof. Dr. Eşref Adalı****'ya teşekkürlerimi sunarım.*

TEK KULLANIMLIK PAROLA ÜRETEN SİSTEM

(ÖZET)

Bankacılık ve bilişim sektöründe güvenlik her zaman önemli bir konu olmuştur. Güvenlik konusunda çeşitli şifreleme algoritmaları ve kimlik belgeleme teknolojileri geliştirilmiştir. Kullanıcılarına kimliklerini elektronik olarak doğrulamaları için statik parolalar soran sistemler en basit güvenlik sistemlerinin bir örneğidir. Zaman içinde statik parolalar dışında başka güvenlik yöntemleri geliştirilmiştir. Yeni geliştirilen kimlik doğrulama yazılımları ve cihazları çeşitli kimlik doğrulama teknolojileri kullanırlar.

Tek kullanımlık parola üreten sistemler kullanıcıların güvenliğini arttırmak için geliştirilmiş teknolojilerden biridir. OTP sistemlerinde tek bir bağlanma işlemi veya hesap işlemi için bir parola üretilir ve bu parola her işlemde değişir. Parola her bir işlem için yeniden üretilir. Bu durum daha önce hesap işlemi yapmak için kullanılmış bir parolayı bir şekilde elde eden kötü niyetli kimselerin güvenliği aşarak hesaba giriş yapabilmesini olanaksız kılar, çünkü daha önce hesaba giriş yapmak için kullanılmış olan parola artık geçersizdir ve hesaba giriş yapmak için gerekli olan parola değişmiştir.

Bu çalışmada tek kullanımlık parola teknolojileri incelenmiştir ve tek kullanımlık parola üreten bir sistem gerçekleştirilmiştir. Tek kullanımlık parola üretmek için çeşitli yöntemler vardır. Tek kullanımlık parola üretmek için iki temel yöntem zaman senkronizasyonu ve matematiksel algoritmalardır. Bu çalışmada matematiksel bir algoritma olan DES yöntemi tek kullanımlık parolalar oluşturmak için kullanılmıştır. Tek kullanımlık parolalar daha önce üretilen parolalar üzerinden üretilebilir. Özel bir başlangıç değeri üzerinde matematiksel bir fonksiyon ile işlemler yapılır ve seri olarak parolalar üretilebilir. Üretilen her bir parola daha önce üretilen parolaya ve seçilen matematiksel fonksiyona bağlıdır. DES yönteminde de parolalar buna benzer bir şekilde oluşturulur. Bu çalışmada 3-DES algoritması gerçekleştirilmiştir. 3-DES algoritması DES algoritmasının özel bir halidir. DES algoritması sabit uzunlukta anahtarlar kullanarak verilen bir mesajı şifreleyebilen bir algoritmadır.

Bu çalışma iki bölümden oluşmaktadır. Çalışmanın ilk bölümünde 3-DES algoritması İTÜ Bilgisayar Mühendisliği bölümünde mikroişlemci derslerinde öğretilen örnek MİB üzerine gerçeklenmiştir. Yazılım aynı mikroişlemci derslerinde kullanılan MikBil simulasyon paketi üzerinde yazılmıştır ve test edilmiştir. Yapılan testlerde 3-DES algoritmasının verilen bir mesajı başarılı bir şekilde şifreleyebildiği gözlemlenmiştir. Çalışmanın ikinci bölümünde 3-DES algoritması MC9S12C32 mikrodenetçisi üzerinde gerçeklenmiştir. Bu amaçla MC9S12C32 mikrodenetçisini kullanan CSM12C32 geliştirme kiti kullanılmıştır. Bu kite yerleştirilen kod çalıştırıldığında kit üzerindeki SW1 düğmesine her basıldığında bir tek kullanımlık parola üretilir ve bu parola HY-1602 LCD gösterge üzerinde gösterilir. Bu projede 3-DES algoritması her bir turda 16 haneli bir parola üretir. Bu 16 haneli parola her turda basit bir Hash fonksiyonuna girer ve sonuçta 12 haneli bir parola üretilmiş olur. Bu 12 haneli parola LCD göstergede gösterilir. Hash fonksiyonu güvenliğin arttırılması için gereklidir. Hash fonksiyonu tek kullanımlık parolalar üretmek için kullanılan 3-DES algoritmasının kullandığı anahtarları ve son üretilen tek kullanımlık parolayı ele geçiren birinin bir sonraki üretilecek parolayı bilmesini engeller. Sonuç olarak tek kullanımlık parolalar bilişim sistemlerinde güvenliği arttırırlar.

ONE TIME PASSWORD SYSTEM

(SUMMARY)

Security has long been an important concern in information systems. Various forms of security measures have been used in information systems such as static passwords. A password is a string of characters used for user authentication to control access to a resource. Passwords are used to prove identity and for access approval to gain access to a system. Static passwords are an effective method of access control in information systems and computer networks. As time passed new methods and technologies related to security systems have emerged.

One time password systems are one of the technologies developed for increasing the security of the users of information systems. One time password technology is an innovative method of controlling access to a resource. They strongly reinforce information systems in respect of security. A one time password is a password that is valid for only one transaction or login session. Each time a transaction would be made, a new one time password is produced exclusively for that transaction. This prevents intruders from gaining access to one's account in a system by password replay attacks since the password that was used previously for a transaction is no longer valid and a new password is required in order to do transaction.

In this project, a system producing one time passwords has been developed. One time password systems use various techniques to produce one time passwords. One time password systems make use of pseudorandomness in order to make it difficult to predict future one time passwords by oberving previous ones. The two common methods to produce one time passwords are time-synchronization method and mathematical algorithms. In time-synchronization method, the users are provided with a security token that produces one time passwords based on the value of a clock inside the token which has been synchronized with the clock on the proprietary authentication server. In these systems, time is an important part of the password generation algorithm since the generated passwords are based on current time rather than, or in addition to, previous passwords or a secret key. Mathematical algorithms are another method of producing one time passwords. In the method of mathematical algorithms, the generated passwords are based on a mathematical function in addition to previous passwords. The one time password system works by starting with an initial seed and then producing passwords as many times as necessary.

In this project, a system running with a block cipher algorithm is designed. The method of mathematical algorithms is used to generate one time passwords in this project. The chosen algorithm for this task is Triple DES algorithm. Triple DES algorithm is a special variant of Data Encrytion Standard algorithm known as DES. In cryptography, Triple DES is the common name for the Triple Data Encryption Algorithm block cipher which applies Data Encryption Standard cipher algorithm three times to each data block. When the DES algorithm was designed, the original key size of 56 bits was enough to protect users, but the

availability of increasing computational power made brute force attacks feasible. Triple DES algorithm provides a way of increasing key size to protect against such attacks without the need to design a completely new block cipher algorithm. In Triple DES algorithm, a key bundle of two or three keys each of which is 56 bits is used. In this project, two keys are used to generate one time passwords. First a given message is encrypted with the first key. Then, the word resulting from this operation is decrypted using the second key. Since the second key is different from the first key, what is done in the second operation is encrypting the message again rather than decrypting it. So the message is encrypted twice using the two different keys. Lastly the message is encrypted again with the first key to produce the final result. So the message is encrypted three times with the DES algorithm. Triple DES works this way.

This project is divided into two main parts. In the first part of the project, which comprises the first half of the estimated project timeline, the software for one time password system was developed on the sample microprocessor used in the Microprocessor Systems lectures of Istanbul Technical University. The code was written and tested on MikBil simulation program which is also used in these lectures. In the test phase of development cycle which comprised the first half of the project timeline, it was observed that the system successfully generated one time passwords based on the 3-DES algorithm. In the second half of the project timeline, the one time password system was to be implemented on a real microprocessor. Motorola HCS12 processor was selected for this purpose. Freescale MC9S12C32 general purpose microcontroller has this processor in its architecture and it is used in this project. The passwords are generated when the user presses a button, so a push button was needed. So a complete evaluation board was selected for this project. The evaluation board CSM12C32 was chosen for this project. The generated passwords are displayed on the LCD module named HY-1602. HY-1602 is a HD44780 compatible LCD screen with backlight and it has the capability to display two lines of 16 characters. The connection between the CSM12C32 kit and the LCD is provided by a 40-wire connector cable which is the same cable used with hard disk drives in personal computers. In the one password system in this project, a password is generated when the user presses a button. When the user presses the SW1 user button on the CSM12C32 kit, the triple DES algorithm produces a 16 character password and that password is then passed to a cryptographic hash function. The cryptographic has function designed for this project is a simple one, it uses every three characters out of every four characters in the password generated by 3-DES algorithm to produce a final password of 12 characters. So the output of the hash function is a 12 character password. This password is displayed on the LCD screen. The cryptographic hash function is needed because of why an intruder who manages to record previous passwords and also knows the keys used by the algorithm can predict future passwords. Hash function prevents this from happening. So the hash function is an important part of the password generation process. So one time password systems are effective in securing transactions done by users of information systems. In conclusion, one time password systems effectively reinforce information systems in respect of security.

İÇİNDEKİLER

1 GİRİŞ

Bankacılık ve bilişim sektöründe güvenlik her zaman önemli bir konu olmuştur. Güvenlik konusunda çeşitli kimlik belgeleme teknolojileri ve şifreleme algoritmaları geliştirilmiştir. Kullanıcılarına kimliklerini elektronik olarak doğrulamaları için statik parolalar soran sistemler en basit güvenlik sistemlerinin bir örneğidir. Zaman içinde statik parolalar dışında başka güvenlik yöntemleri geliştirilmiştir. Yeni geliştirilen kimlik doğrulama yazılımları ve cihazları çeşitli kimlik doğrulama teknolojileri kullanırlar. T-FA(Two factor authentication), tek kullanımlık parolalar, PC kartları, akıllı kartlar, akıllı kart temelli USB cihazları ve benzer teknolojiler bilişim sistemlerinin katmanlı güvenlik yapısına yeni katmanlar eklemek ve böylece güvenliği arttırmak amacıyla geliştirilmiştir [3]. Bu bitirme ödevinde bu teknolojilerden tek kullanımlık parola teknolojisi incelenmiştir.

1.1 Projenin Amacı

Bu çalışmada amaç tek kullanımlık parolalar üretebilen bir sistem gerçekleştirmektir. Bu amaçla, verilen bir mesajı sabit uzunlukta anahtarlar kullanarak şifreleyebilen bir blok şifreleyici tek kullanımlık parolalar üretmek için kullanılmıştır.

OTP adı verilen tek kullanımlık parola üreten sistemler güvenliği arttırmak amacıyla tasarlanmışlardır. OTP sistemlerinde tek bir bağlanma işlemi veya hesap işlemi için bir parola üretilir ve bu parola her işlemde değişir. Parola her bir işlem için tekrar üretilir. Bu durum daha önce hesap işlemi yapmak için kullanılmış bir parolayı bir şekilde elde eden kötü niyetli kimselerin güvenliği aşarak hesaba giriş yapabilmesini olanaksız kılar, çünkü daha önce hesaba giriş yapmak için kullanılmış olan parola artık geçersizdir ve değişmiştir [2]. OTP sistemleri genellikle diğer güvenlik sistemleri ile birlikte kullanılır ve katmanlı bir güvenlik sisteminin güçlü bir katmanını oluştururlar. Örneğin OTP parolalar statik parolalar ile birlikte kullanılabilir. Kullanıcı hesap işlemi yapmak için sisteme bağlanırken öncelikle statik bir parola ile sisteme girebilir ve ardından sistem kullanıcıdan elindeki tek kullanımlık parolayı sisteme girmesini isteyebilir. Böylece çok katmanlı bir güvenlik sistemi kurulabilir.

1.2 Projenin Kapsamı

Bu çalışmada tek kullanımlık parolalar üretebilen bir sistem gerçekleştirilmiştir. Tek kullanımlık parolalar üretmek için DES blok şifreleme algoritması kullanılmıştır ve bu algoritmanın özel bir hali olan 3-DES yazılım ile gerçekleştirilmiştir. Projede tasarlanan yazılım bir mikroişlemci veya mikrodenetçiye yerleştirilmiştir ve bu noktada yazılım ve donanım test edilmiştir. Proje, DES algoritmasının yazılım ile gerçekleştirilmesi, yazılımın ilgili mikroişlemciye yerleştirilmesi ve tek kullanımlık parola üretilmesi için tasarlanan ek donanımın mikroişlemci ile bağlantısının yapılması süreçlerini kapsar.

1.3 Kullanım Alanları

Tek kullanımlık parola üreten sistemler yaygın olarak bankaların elektronik ödeme sistemlerinde kullanılırlar [4]. Ayrıca internet üzerinden alışveriş yapılan sitelerde de tek kullanımlık parolalar hesap erişimi için kullanılırlar.

Tek kullanımlık parola üreten sistemlerin temel fonksiyonu bilgisayar ağları ve banka sistemlerinde güvenliği arttırmaktır. Tek kullanımlık parola sistemleri çok katmanlı bir güvenlik sisteminin güçlü bir katmanını oluştururlar ve kullanıcıları parola tekrarlama saldırılarından korurlar. Tek kullanımlık parolalar her hesap işlemi için yeniden üretilirler, daha önce kullanılan bir parola tekrar geçerli olmaz. Bu nedenle daha önce kullanılan bir parola kötü niyetli kimseler tarafından ele geçirilmiş olsa bile bu parola tekrar geçerli olmayacağı için bu kimseler güvenliği aşarak sisteme giremezler. Böylece kullanıcılar parola tekrarlama saldırılarından korunurlar.

1.4 Başarım Kriterleri

Projenin temel başarım kriterleri üretilen tek kullanımlık parolaların tahmin edilme olasılığının düşük olması ve parolaların tekrarlanma sıklığının yani aynı parolanın tekrar üretilmesi sıklığının düşük olmasıdır. Kullanıcı tarafında tek kullanımlık parola üreten donanım parçalarını kullanan sistemlerin temel kısıtlaması üretilen parolaların tam anlamıyla rastgele olmaması ve aslında sözde rastgele parolalar olmasıdır. Tam anlamıyla rastgele parola üreten bir sistemde parolalar ancak sunucu sistem tarafında üretilebilir ve kullanıcıya çeşitli yöntemler ile ulaştırılır. Ancak kullanıcının elindeki cihaz tarafından parola üretiliyorsa bu parolanın aynı zamanda sunucu tarafında da üretilebilmesi gerekir. Sunucu ve kullanıcı arasında bir bağlantı olmadığı için ve aynı parolanın hem sunucu hem de kullanıcı tarafında aynı anda üretilmesi gerektiği için bu durum parolaların bir noktaya kadar tahmin edilebilir olmasına neden olur. Bu tek kullanımlık parola üreten sistemlerde önemli bir kısıtlamadır.

1.5 Projeye İlişkin Kestirimler

Projenin bütçesi ve boyutları tahmin edilebilir. Projede yaklaşık olarak 2000 satırlık düşük seviyeli programlama dili ile yazılmış kod olması bekleniyor. Projenin boyutu yaklaşık olarak 2000 LOC olarak tahmin edilebilir. Projede kullanılacak mikrodenetleyici ve LCD gösterge devresinin bir maliyeti olabilir. Projede kullanılacak donanımın maliyeti yaklaşık olarak 100 TL olabilir. Proje yaklaşık olarak 1 yıl sürecektir. Proje, 2012 Ekim tarihinden 2013 Haziran tarihine kadar sürecektir. Projede 1 kişi çalışacaktır, projenin insan kaynağı 1 kişiden oluşmaktadır. Projenin boyutları, maliyeti, süresi ve insan kaynakları ile ilgili tahminler bunlardır.

1.6 Çözüm Stratejileri ve Uygulanacak Yöntemler

Projede tek kullanımlık parolalar üretebilen bir sistem tasarlanacaktır. Tek kullanımlık parolalar üretmek için DES algoritması kullanılacaktır. Projenin başlangıçta iki modülden oluşması planlanıyordu. Bu modüller kullanıcıların hesaplarına erişmek için kullanacakları sunucu sistem ve tek kullanımlık parolalar üreten kullanıcı cihazlarıydı. Projede tek kullanımlık parolalar üreten kullanıcı cihazları 3-DES algoritmasını kullanmak üzere tasarlanmıştır. Ancak daha sonra projenin sunucu parçasının yapılmamasına karar verilmiştir.

Projede kullanıcı cihazları tek kullanımlık parolalar üretmek için 3-DES algoritmasını kullanacaktır. Projenin bu parçası öncelikle örnek MİB üzerinde gerçekleştirilmiştir ve MikBil simülasyon paketi üzerinde test edilmiştir [1]. Daha sonra algoritma gerçek bir mikroişlemci veya mikrodenetçi üzerinde gerçekleştirilecektir. Proje için Motorola HCS12 işlemcisini kullanan MC9S12C32 mikrodenetçisi kullanılabilir. Örnek MİB üzerinde yazılan yazılım gerçek bir işlemci üzerinde yeniden yazılacaktır ve gerekli donanım ile birlikte test edilecektir.

Projede sunucu sisteminin de gerçekleştirilmesi durumunda test sürecinde sunucu sistem ve kullanıcı cihazları birlikte test edilecektir. Test sürecinde projenin sunucu bölümünde kullanıcı hesapları açılacaktır. Bu kullanıcıların her birine tek kullanımlık parola üreten donanım parçaları verilecektir. Kullanıcılar kullanıcı numarası ve statik parolaları ile sisteme bağlanacaklardır ve sistem kullanıcılardan tek kullanımlık parolalar isteyecektir. Kullanıcılar ellerindeki tek kullanımlık parola üreten cihazların o anda ürettiği tek kullanımlık parolaları sisteme gireceklerdir ve bu noktada sistemin kullanıcılara giriş izni verip vermemesine bakılacaktır.

1.7 Daha Önce Bu Konuda Yapılan Çalışmalar

Tek kullanımlık parola üretimi konusunda önemli çalışmalar yapılmıştır. Tek kullanımlık parola üretmek için çeşitli yöntemler geliştirilmiştir. Tek kullanımlık parola üretmek için iki temel yöntem zaman senkronizasyonu ve matematiksel algoritmalardır. Zaman senkronizasyonu yönteminde genellikle özel bir donanım parçası tek kullanımlık parola üretmek için kullanılır. Donanım parçasının içinde, hesap işlemleri yapılan sistemin zamanı ile senkronize edilmiş bir saat bulunmaktadır [2]. Bu sistemlerde zaman parola üreten algoritmanın önemli bir parçasıdır. Bu sistemlerde tek kullanımlık parola üretimi için o anki zaman değeri ve sabit gizli bir anahtar bir arada kullanılabilir. Böylece belli bir zaman periyodu ile değişen parolalar üretilir. Bu parolalar her bir hesap işlemi için tekrar üretilirler ve böylece her işlem için farklı parola kullanılmış olur. Parola üretimi için özel bir donanım parçası kullanılabileceği gibi cep telefonları da özel yazılımlar sayesinde tek kullanımlık parola üretimi için kullanılabilirler. Tek kullanımlık parola üretmenin diğer bir yöntemi matematiksel algoritmalardır. Tek kullanımlık parolalar daha önce üretilen parolalar üzerinden üretilebilir. Özel bir başlangıç değeri üzerinde matematiksel bir fonksiyon ile işlemler yapılır ve seri olarak parolalar üretilebilir [2]. Üretilen her bir parola daha önce üretilen parolaya ve seçilen matematiksel fonksiyona bağlıdır.

Tek kullanımlık parola üreten sistemler çeşitli blok şifreleme algoritmaları kullanırlar. Bu algoritmalardan biri de DES algoritmasıdır. Bu çalışmada DES algoritması incelenmiştir. DES algoritması sabit uzunlukta bir bitler dizisini alır ve bu dizi üzerinde çeşitli operasyonlar

yaparak aynı uzunlukta şifrelenmiş bir mesaj üretir [7]. DES algoritmasında blok uzunluğu 64 bittir. DES algoritması şifreleme işlemi için özel bir anahtar kullanır. Bu anahtar yalnızca şifreleme işlemini yapan kurum tarafından bilinir. Parola çözme işlemi yalnızca bu anahtarı bilen kimseler tarafından yapılabileceği için, kullanıcılar güvenli bir şekilde hesap işlemlerini yapabilirler. DES algoritmasında 64 bitlik anahtarlar kullanılır; ancak bu 64 bitin yalnızca 56 biti şifreleme işleminde kullanılır. Anahtar içindeki 8 bit parite kontrolü için kullanılır.

Zaman içinde DES dışında başka blok şifreleme algoritmaları geliştirilmiştir. Ünlü blok şifreleme algoritmalarından bazıları şunlardır: Lucifer, IDEA, RC5, Rijndael / AES, Blowfish. Lucifer algoritması ilk sivil blok şifreleme algoritmasıdır [5]. Lucifer algoritması IBM şirketinde 1970'li yıllarda geliştirilmiştir. Bu algoritma Horst Feistel'in yaptığı çalışmalara dayanır. Bu algoritmanın biraz değiştirilmiş bir versiyonu Amerika Birleşik Devletleri tarafından federal bilgi işleme standardı olarak belirlenmiştir ve veri şifreleme standardı (DES) adını almıştır. Lucifer algoritması DES algoritmasının temelini oluşturur. DES algoritması ABD ulusal standartlar dairesi tarafından kamusal bir davet üzerine seçilmiştir ve 1976 yılında yayınlanmıştır. DES algoritması yayınlandığı zamandan beri yaygın bir şekilde kullanılmaktadır. IDEA algoritmasi diğer bir blok şifreleme algoritmasıdır. IDEA algoritması ETH Zurich üniversitesinden James Massey ve Xuejia Lai tarafından geliştirilmiştir. IDEA algoritması 1991 yılında DES algoritmasının yerini alacak bir algoritma olarak ortaya atılmıştır. IDEA algoritması 64 bitlik bloklar üzerinde çalışır ve 128 bitlik anahtarlar kullanır [5]. IDEA algoritması birbiri ile aynı 8 dönüşüm ve son bir çıkış dönüşümünden oluşur. IDEA algoritmasında şifreleme ve şifre çözme işlemleri birbirine çok benzerdir. IDEA algoritması, sağladığı güvenliği farklı farklı işlemlerin sonuçlarını bir arada kullanmasına borçludur. IDEA algoritması modüler toplama ve çarpma işlemlerini ve bitler üzerinde YADA işlemini bir arada kullanır. Bu işlemler cebirsel olarak birbiri ile uyumsuz işlemler olmasına rağmen IDEA algoritmasında bir arada kullanılmaktadırlar. Bir diğer blok şifreleme algoritması RC5 algoritmasıdır. RC5 algoritması Ronald Rivest tarafından 1994 yılında tasarlanmıştır. Bu algoritma diğer pekçok algoritmanın aksine değişken blok büyüklüğü, anahtar uzunluğu ve tur sayısına sahiptir [5]. RC5 algoritmasının önemli bir özelliği veriye bağımlı döndürme işlemleridir. RC5 aynı zamanda bir dizi modüler toplama ve bitler üzerinde YADA işlemlerinden oluşur. Algoritmanın genel yapısı bir Feistel ağına benzer. Algoritmanın basitliği ve yenilikçi veriye bağımlı döndürme işlemleri bu algoritmayı ilginç bir çalışma konusu yapmıştır. Rijndael / AES diğer bir blok şifreleme algoritmasıdır. DES algoritmasının yerini bir Birleşik Devletler federal standardı olarak AES algoritması almıştır [5]. AES algoritması 2001 yılında NIST tarafından 5 yıllık bir kamusal rekabet sürecinden sonra kabul edilmiştir. AES algoritması Joan Daemen and Vincent Rijmen tarafından tasarlanmıştır. AES algoritması 128 bitlik sabit bir blok uzunluğuna ve 128, 192 veya 256 bitlik anahtar uzunluğuna sahiptir. AES algoritması 4*4 boyutunda ve baytlardan oluşan durum matrisleri üzerinde çalışır. Blowfish diğer bir blok şifreleme algoritmasıdır. Blowfish algoritması 1993 yılında Bruce Schneier tarafından tasarlanmıştır. Bu algoritma 64 bit blok uzunluğu ve 1 bitten 448 bite kadar değişen anahtar uzunluğuna sahiptir [5]. Bu algoritma 16 turluk bir Feistel blok şifreleyicisidir ve anahtara bağımlı büyük S kutuları kullanır. Algoritmanın en önemli özelliği anahtara bağımlı S kutuları ve oldukça karmaşık anahtar işlemleridir.

1.8 Yapılan Çalışma ve Sonuçları

Proje iki bölüme ayrılmıştır. Çalışmanın ilk bölümünde 3-DES algoritması İTÜ Bilgisayar Mühendisliği bölümünde mikroişlemci derslerinde öğretilen örnek MİB üzerine gerçekleştirilmiştir. Yazılım aynı mikroişlemci derslerinde kullanılan MikBil simulasyon paketi üzerinde yazılmıştır ve test edilmiştir. MikBil simulasyon paketi örnek MİB üzerinde kod yazılmasını ve yazılan kodların test edilmesini sağlayan bir platformdur [1]. Yapılan testlerde 3-DES algoritmasının verilen bir mesajı başarılı bir şekilde şifreleyebildiği gözlemlenmiştir.

Projenin ikinci bölümünde 3-DES algoritması gerçek bir mikroişlemci üzerinde gerçekleştirilmiştir. Algoritmaya ilişkin kod yazılmıştır ve bu kod tek kullanımlık parolalar üretmek için gerekli olan ek devre ile birlikte test edilmiştir. Projenin ikinci bölümü için Motorola HCS12 işlemcisini kullanan MC9S12C32 mikrodenetçisi tercih edilmiştir. Bu nedenle bu mikrodenetçiyi kullanan CSM12C32 geliştirme kartı kullanılmıştır. Projede üretilen tek kullanımlık parolaları göstermek için HY-1602 LCD gösterge seçilmiştir.

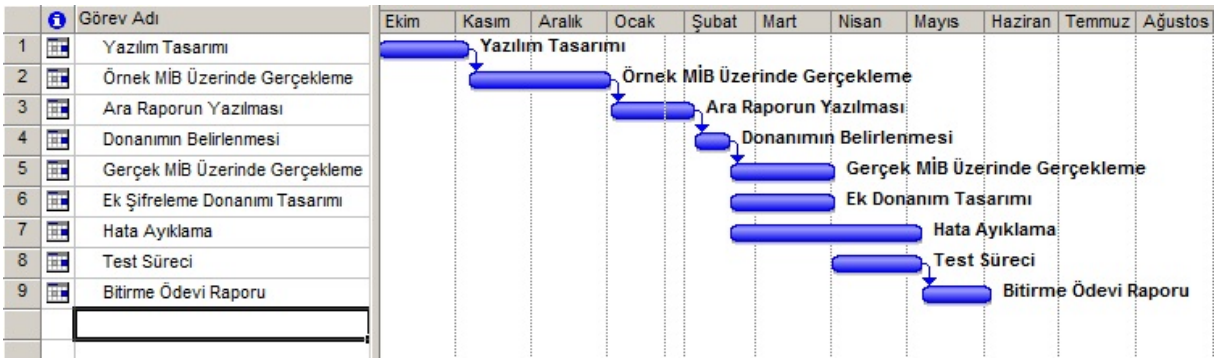
Projede tek kullanımlık parolalar üretmek için 3-DES algoritması kullanılmıştır. Bu algoritma DES algoritmasının özel bir halidir. Bu algoritma DES algoritmasının 3 defa farklı anahtarlar ile çalıştırılması ile uygulanır. Bu nedenle öncelikle DES algoritması yazılmıştır ve daha sonra 3-DES algoritmasını gerçekleştirmek için yazılan kod içinde DES algoritmasına ilişkin altprogram 3 defa çağrılmıştır. Deneyde yazılan kod CSM12C32 geliştirme kitine yerleştirildiğinde ve kod çalıştırıldığında kit üzerindeki SW1 düğmesine her basıldığında bir tek kullanımlık parola üretildiği ve bu parolanın LCD ekranda gösterildiği görülmüştür.

2 PROJENİN TANIMI VE PLANI

Bu çalışmada tek kullanımlık parolalar üretebilen bir sistem tasarlanmıştır. Tek kullanımlık parolalar üretmek için DES algoritmasının özel bir hali olan 3-DES algoritması kullanılmıştır. Bu bölümde projenin hangi modüllerden oluştuğu anlatılmıştır ve ayrıntılı iş planı incelenmiştir. Proje tek kullanımlık parolalar üretmek için kullanılacak ek devreler ve mikroişlemci içine yerleştirilecek şifreleme algoritmasından oluşur.

Projenin en önemli parçası şifreleme algoritmasıdır. Şifreleme algoritması mikroişlemci içine yerleştirilecek ve tek kullanımlık parolalar üretmek için kullanılacak kodları içerir. Bu kodlar bir ana program, bir şifreleme altprogramı ve bir de parola çözme altprogramından oluşur. Ana program içinde gerek duyulduğunda altprogramlar çağrılır ve böylece şifreleme veya parola çözme işlemleri gerçekleştirilir. 3-DES algoritması DES algoritmasının özel bir halidir. 3-DES algoritmasında iki adet 56 bitlik anahtar kullanılır. İlk anahtar verilen bir mesajı şifrelemek için kullanılır. İkinci anahtar ise şifrelenen mesajı çözmek için kullanılır. İkinci anahtar birinci anahtardan farklı olduğu için ikinci işlemde parola çözme işlemi yerine tekrar şifreleme işlemi gerçekleştirilmiş olur. Böylece verilen mesaj art arda iki kez şifrelenmiş olur. Son olarak mesaj birinci anahtar kullanılarak tekrar şifrelenir ve böylece başlangıçta elimizde bulunan mesaj art arda üç kez şifrelenmiş olur. 3-DES algoritması bu şekilde çalışır. Bu nedenle algoritma ana program, şifreleme altprogramı ve parola çözme altprogramından oluşacak şekilde tasarlanmıştır.

Proje ile ilgili ayrıntılı iş planı aşağıda Şekil 2.1'de verilmiştir. Projede öncelikle tek kullanımlık parolalar oluşturmak için kullanılacak olan yazılımın tasarımı yapılmıştır. Tek kullanımlık parolalar üretmek için DES algoritmasının özel bir hali olan 3-DES algoritması kullanılmıştır. Gerekli yazılımın tasarımı yapıldıktan sonra yazılım İTÜ Bilgisayar Mühendisliği bölümünde mikroişlemci sistemleri dersinde öğretilen örnek MİB üzerinde gerçekleştirilmiştir. Yazılım aynı derslerde öğretilen MikBil simülasyon paketi üzerinde yazılmıştır ve test edilmiştir. Yapılan testlerde 3-DES algoritmasının verilen bir mesajı başarılı bir şekilde şifreleyebildiği görülmüştür. Algoritma örnek MİB üzerinde gerçekleştirildikten sonra gerçek bir MİB üzerinde gerçekleştirilmesi için kullanılacak mikroişlemci veya mikrodenetçi belirlenmiştir. Projenin bu bölümü için Motorola HCS12 işlemcisini kullanan MC9S12C32 mikrodenetçisi tercih edilmiştir. Bu nedenle bu mikrodenetçiyi kullanan CSM12C32 geliştirme kartı kullanılmıştır. Projede üretilen tek kullanımlık parolaları göstermek için HY-1602 LCD gösterge seçilmiştir. Donanım belirlendikten sonra algoritma gerçek MİB üzerinde gerçekleştirilmiştir. Bu aşamada ayrıca tek kullanımlık parolalar üretmek için kullanılacak ek devre de tasarlanmıştır, LCD gösterge ve CSM12C32 kiti arasındaki bağlantılar yapılmıştır. Bu süreçle beraber hata ayıklama süreci de başlamıştır ve bu süreç kısa zamanda tamamlanmıştır.



Şekil 2.1: Projenin Ayrıntılı İş Planı

Algoritma gerçek MİB üzerinde gerçekleştirildikten sonra hata ayıklama ve test süreçleri yaşanmıştır. Son olarak Mayıs ayı boyunca bitirme ödevi raporu yazılmıştır. Bitirme ödevi ile ilgili ayrıntılı iş planı budur.

3 KURAMSAL BİLGİLER

Bu projede tek kullanımlık parolalar üretebilen bir sistem tasarlanmıştır. Tek kullanımlık parolalar üretmek için DES algoritmasının özel bir hali olan 3-DES algoritması kullanılmıştır. Bu bölümde DES algoritmasını ve onun özel bir hali olan 3-DES algoritmasını anlatacağız. DES algoritmasının nasıl çalıştığı ile ilgili bilgiler J. Orlin Grabbe'nin "The DES Algorithm Illustrated" isimli yazısında bulunmaktadır [6]. Bu bölümde bulunan şekiller ve tablolar bu kaynaktan alınmıştır.

DES algoritması bir blok şifreleme algoritmasıdır ve sabit uzunluklu bloklar üzerinde çalışır. DES algoritması 64 bitlik bir blok için 2^64 farklı düzenleniş biçimi içinde bir permütasyon ile sonuçlanır [6]. Öncelikle şifrelenmesi istenen 64 bitlik mesaj 32 bitlik iki parçaya bölünür ve bu parçalara L ve R isimleri verilir. Şifrelenmesi istenen mesaj M olsun. Mesaj M, 16 sayı tabanında örneğin M = 0123456789ABCDEF şeklinde verilebilir. Mesaj 2 sayı tabanında yazılırsa M aşağıda görüldüğü gibi olur:

M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
L = 0000 0001 0010 0011 0100 0101 0110 0111
R = 1000 1001 1010 1011 1100 1101 1110 1111

M olarak verilen mesajın ilk biti 0 ve son biti de 1'dir. Bu çalışmada bitler soldan sağa doğru okunur.

DES algoritması 64 bitlik bloklar üzerinde 56 bitlik anahtarlar kullanarak çalışır. Bu çalışmada bit indeksleri soldan başlayarak ve sağa doğru ilerleyerek 1'den 64'e kadar numaralandırılmıştır. DES algoritmasında anahtarlar bellekte 64 bitlik değerler olarak kayıtlıdır; ama DES hesaplamalarda 8 ve sekizin katları olan sıralardaki bitleri kullanmaz.

K olarak verilen anahtar 16 sayı tabanında K = 133457799BBCDFF1 şeklinde verilmiş olsun. K anahtarı ikilik tabanda aşağıda görülen değere sahiptir:

K = 00010011 00110100 01010111 01111001 10011011 10111100 11011111 11110001

DES algoritması aşağıda belirtilen adımları uygulayarak çalışır. Öncelikle K anahtarı kullanılarak her biri 48 bit uzunluğunda 16 adet geçici anahtar üretilir. 64 bitlik anahtar PC-1 tablosunda verilen değerlere göre permütasyona uğrar ve sonuç olarak K+ anahtarı üretilmiş olur.

Tablo 3.1: PC-1 Tablosu

PC-1

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Tablo 3.1'de PC-1 tablosundaki ilk değer 57'dir. Bunun anlamı şudur, K anahtarının soldan 57. biti permütasyon sonucunda oluşan K+ anahtarının ilk biti olacaktır. K anahtarının soldan 49. biti K+ anahtarının soldan ikinci biti olur ve K anahtarının 4. biti de K+ anahtarının son biti olur. 64 bitlik K anahtarının Tablo 3.1'deki değerlere göre permütasyona uğraması sonucunda K+ anahtarı oluşur ve K+ anahtarı aşağıda görülen değeri alır.

K = 00010011 00110100 01010111 01111001 10011011 10111100 11011111 11110001

K+ = 1111000 0110011 0010101 0101111 0101010 1011001 1001111 0001111

Daha sonra K+ anahtarı C_0 ve D_0 olmak üzere iki parçaya bölünür. Her bir parça 28 bit uzunluğundadır. C_0 sol taraftaki parçadır ve D_0 da sağ taraftaki parçadır.

$\boldsymbol{C_0}$ = 1111000 0110011 0010101 0101111
$\boldsymbol{D_0}$ = 0101010 1011001 1001111 0001111

Daha sonra $\boldsymbol{C_0}$ ve $\boldsymbol{D_0}$ değerleri kullanılarak 16 adet $\boldsymbol{C_n}$ ve $\boldsymbol{D_n}$ blok çiftleri oluşturulur. Her bir $\boldsymbol{C_n}$ ve $\boldsymbol{D_n}$ blok çifti bir önceki $\boldsymbol{C_{n-1}}$ ve $\boldsymbol{D_{n-1}}$ blok çiftinden oluşturulur ve bu işlem n=1'den n=16'ya kadar aşağıdaki döndürme döngüsü tablosu kullanılarak yapılır.

Tablo 3.2: Döndürme Döngüsü Tablosu

Iteration Number	Number of Left Shifts
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Tablo 3.2'de her bir C_n ve D_n çiftinin n=1,2,3…16 için nasıl oluşturulacağı belirtilmiştir. Her bir C_n ve D_n çifti bir önceki C_n ve D_n çiftinin sola döndürülmesi ile elde edilir. Sola döndürme işlemi şu şekilde yapılır; C_n veya D_n değerindeki her bir bit bir kez sola ötelenir ve soldan birinci bit soldan sonuncu bit olur. Tablo 3.2'ye göre C_3 ve D_3 değer çifti, C_2 ve D_2 değer çiftinin iki kez sola döndürülmesi ile elde edilir.Aynı tabloya göre C_{16} ve D_{16} değer çifti de C_{15} ve D_{15} değer çiftinin bir kez sola döndürülmesi ile elde edilir. C_0 ve D_0 değer çifti üzerinde döndürme işlemleri yapılarak aşağıdaki değerler elde edilir.

$\boldsymbol{C_0}$ = 11110000110011001010101011111
$\boldsymbol{D_0}$ = 01010101011001100111100011111

$\boldsymbol{C_1}$ = 11100001100110010101010111111
$\boldsymbol{D_1}$ = 10101010110011001111100011110

$\boldsymbol{C_2}$ = 11000011001100101010101111111
$\boldsymbol{D_2}$ = 01010101100110011111000111101

C_3 = 00001100110010101010111111111
D_3 = 01010110011001111000111110101

C_4 = 00110011001010101011111111100
D_4 = 01011001100111110001111010101

C_5 = 11001100101010101111111110000
D_5 = 01100110011110001111010101010

C_6 = 00110010101010111111111000011
D_6 = 10011001111000111110101010101

C_7 = 11001010101011111111100001100
D_7 = 01100111100011111010101010110

C_8 = 00101010101111111110000110011
D_8 = 10011110001111010101011001

C_9 = 01010101011111111100001100110
D_9 = 00111100011110101010110011

C_{10} = 01010101111111110000110011001
D_{10} = 11110001111010101011001100

C_{11} = 01010111111110000110011001010
D_{11} = 11000111101010101100110011

C_{12} = 01011111111000011001100101010
D_{12} = 00011110101010101100110011111

C_{13} = 01111111100001100110010101010
D_{13} = 01111010101010110011001111100

C_{14} = 11111110000110011001010101010
D_{14} = 11101010101011001100111100011

C_{15} = 11111000011001100101010101111
D_{15} = 10101010101100110011111000111

C_{16} = 11110000110011001010101011111
D_{16} = 01010101011001100111100011111

Daha sonra 1<=n<=16 için her bir birleştirilmiş C_nD_n değeri için K_n anahtarları bulunur. Bu amaçla her bir C_nD_n değerine Tablo 3.3'deki permütasyon uygulanır. Her bir C_nD_n değeri 56 bit uzunluğundadır, ancak PC-2 tablosu bunun 48 bitini kullanır.

Tablo 3.3: PC-2 Tablosu

PC-2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Tablo 3.2'ye göre CnDn değerinin soldan 14. biti K_n anahtarının ilk bitidir. Yine aynı tabloya göre CnDn değerinin soldan 17. biti K_n anahtarının soldan ikinci bitidir ve C_nD_n anahtarının soldan 32. biti de K_n anahtarının soldan 48. biti, yani sonuncu bitidir. Birinci anahtar için elde aşağıdaki değer vardır:

C_1D_1 = 1110000 1100110 0101010 1011111 1010101 0110011 0011110 0011110

Bu değer üzerinde PC-2 permütasyonu uygulanarak K_1 anahtarı elde edilir.

K_1 = 000110 110000 001011 101111 111111 000111 000001 110010

Benzer şekilde diğer geçici anahtarlar da elde edilir. Bu örnek için diğer geçici anahtarların değerleri aşağıda görülmektedir.

K_2 = 011110 011010 111011 011001 110110 111100 100111 100101
K_3 = 010101 011111 110010 001010 010000 101100 111110 011001
K_4 = 011100 101010 110111 010110 110110 110011 010100 011101
K_5 = 011111 001110 110000 000111 111010 110101 001110 101000
K_6 = 011000 111010 010100 111110 010100 000111 101100 101111
K_7 = 111011 001000 010010 110111 111101 100001 100010 111100
K_8 = 111101 111000 101000 111010 110000 010011 101111 111011
K_9 = 111000 001101 101111 101011 111011 011110 011110 000001
K_{10} = 101100 011111 001101 000111 101110 100100 011001 001111
K_{11} = 001000 010101 111111 010011 110111 101101 001110 000110
K_{12} = 011101 010111 000111 110101 100101 000110 011111 101001

K_{13} = 100101 111100 010111 010001 111110 101011 101001 000001
K_{14} = 010111 110100 001110 110111 111100 101110 011100 111010
K_{15} = 101111 111001 000110 001101 001111 010011 111100 001010
K_{16} = 110010 110011 110110 001011 000011 100001 011111 110101

Böylece geçici anahtarlar oluşturulmuştur. Bu geçici anahtarlar şifrelemek istenen mesaj üzerinde yapılacak işlemlerde kullanılacaktır. Şimdi mesaj üzerinde yapılacak işlemlere geçelim.

Öncelikle şifrelemek istenen mesaja IP permütasyonu uygulanır. Şifrelemek istenen mesaj M olarak verilsin.

M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

Mesaj M üzerinde Tablo 3.4'de verilen değerlere göre permütasyon uygulanır ve böylece IP değeri elde edilir.

Tablo 3.4: IP Tablosu

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Tablo 3.4'e göre M değerinin soldan 58. biti IP değerinin birinci biti olur. Yine aynı tabloya göre M değerinin soldan 50. biti IP değerinin soldan ikinci biti olur ve M değerinin soldan 7. biti de IP değerinin sonuncu biti olur.

M değerine IP permütasyonu uygulanarak IP değeri elde edilir. IP değeri aşağıda görüldüğü gibidir.

M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
IP = 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000 1010 1010 1111 0000 1010 1010

Daha sonra 64 bitlik IP değeri yarıya bölünür ve sol tarafta 32 bitlik L_0 ve sağ tarafta da 32 bitlik R_0 değeri elde edilir.

L_0 = 1100 1100 0000 0000 1100 1100 1111 1111
R_0 = 1111 0000 1010 1010 1111 0000 1010 1010

Şimdi 16 döngü adımından geçilir ve her bir adımda f fonksiyonu çalıştırılır. Fonksiyon f 32 bitlik bir veri bloğu ve 48 bitlik $\mathbf{K_n}$ anahtarı üzerinde çalışır ve 32 bitlik bir sonuç üretir. Her bir döngü adımı için aşağıdaki değerler hesaplanır.

$L_n = R_{n-1}$
$R_n = L_{n-1} + f(R_{n-1},K_n)$

Burada görülen + işareti YADA işlemini temsil etmektedir. Bu denklemlere göre 1<=n<=16 için her bir döngü adımında $\mathbf{L_n}$ değeri $\mathbf{R_{n-1}}$ değerine eşitlenir ve $\mathbf{R_n}$ değeri de $\mathbf{L_{n-1}}$ değeri ile f fonksiyonunun sonucunun YADA işlemine girerek oluşturdukları sonuç değerine eşitlenir. Bu döngünün sonunda n=16 için $\mathbf{L_{16}R_{16}}$ değeri elde edilir.

Bu döngüde n=1 için aşağıdaki değerler elde edilir:

K_1 = 000110 110000 001011 101111 111111 000111 000001 110010
$L_1 = R_0$ = 1111 0000 1010 1010 1111 0000 1010 1010
$R_1 = L_0 + f(R_0,K_1)$

Şimdi f fonksiyonunun nasıl çalıştığı açıklanacaktır. Bu fonksiyonun sonucunu hesaplamak için öncelikle $\mathbf{R_{n-1}}$ değeri 32 bitten 48 bite genişletilir. Bu $\mathbf{R_{n-1}}$ içindeki bazı bitleri tekrarlayan bir seçim tablosu kullanarak yapılır. Bu işleme E fonksiyonu adı verilir. $\mathbf{E}(R_{n-1})$ fonksiyonu 32 bitlik bir giriş bloğu üzerinde işlem yapar ve 48 bitlik bir çıkış bloğu üretir.

Tablo 3.5: E bit Seçim Tablosu

E BIT-SELECTION TABLE

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Tablo 3.5'e göre $\mathbf{E}(R_{n-1})$ değerinin soldan ilk üç biti R_{n-1} değerinin soldan 32. biti, birinci biti ve ikinci bitidir. Yine aynı tabloya göre $\mathbf{E}(R_{n-1})$ değerinin son iki biti R_{n-1} değerinin soldan 32. biti ve birinci bitidir.

Tablo 3.5'deki değerlere göre bit seçim işlemleri yapılarak $\mathbf{E}(R_0)$ değeri aşağıdaki gibi hesaplanır:

R_0 = 1111 0000 1010 1010 1111 0000 1010 1010
$\mathbf{E}(R_0)$ = 011110 100001 010101 010101 011110 100001 010101 010101

Daha sonra f fonksiyonunun hesaplanmasında $\mathbf{E}(R_{n-1})$ değeri K_n anahtarı ile YADA işlemine girer.

$$K_n + \mathbf{E}(R_{n-1})$$

K_1 ve $\mathbf{E}(R_0)$ değerleri için (K_1)YADA($\mathbf{E}(R_0)$) değeri aşağıdaki gibi hesaplanır.

K_1 = 000110 110000 001011 101111 111111 000111 000001 110010
$\mathbf{E}(R_0)$ = 011110 100001 010101 010101 011110 100001 010101 010101
K_1+$\mathbf{E}(R_0)$ = 011000 010001 011110 111010 100001 100110 010100 100111

Henüz f fonksiyonunun hesaplanması tamamlanmamıştır. Şu ana kadar R_{n-1} değeri 32 bitten 48 bite genişlemiştir ve genişletilmiş değer de K_n anahtarı ile XOR işlemine girmiştir. Şimdi elde 48 bitlik bir değer vardır. Bu 48 bitlik değer 8 adet 6 bitlik gruplardan oluşmaktadır. Şimdi her 6 bitlik grup S kutuları adı verilen tablolarda adresler olarak kullanılır. Her 6 bitlik grup farklı bir S kutusunda bir adres verecektir. Bu adreste 4 bitlik bir değer bulunur. Bu 4 bitlik değer 6 bitlik grubun yerini alacaktır. Bu şekilde eldeki 48 bitlik değerdeki her 6 bitlik grup için bu işlem yapılır ve böylece 48 bitlik değerden 32 bitlik bir sonuç oluşturulur.

Elde bulunan 48 bitlik değer 6 bitlik gruplar halinde yazılırsa:

$$K_n + \mathbf{E}(R_{n-1}) = B_1B_2B_3B_4B_5B_6B_7B_8,$$

Şimdi amaç aşağıdaki değeri hesaplamaktır.

$$S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8)$$

Burada her bir $S_i(B_i)$ değeri i. sıradaki S kutusu tablosunun sonucunu temsil etmektedir. S_1 değerini hesaplamak için kullanılacak tablo aşağıda verilmiştir ve gerekli açıklamalar yapılmıştır.

Tablo 3.6: S1 Tablosu

S1

Row No.	Column Number															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_1 Tablo 3.6'ya göre tanımlanan bir fonksiyon ve B de 6 bitlik bir blok olmak üzere $S_1(B)$ aşağıda açıklandığı gibi hesaplanır. B değerinin ilk ve son bitleri 2 sayı tabanında 0 ve 3 arasında bir değere karşılık gelir(2 sayı tabanında 00 değerinden 11 değerine kadar). Bu değer i olsun. B değerinin ortadaki 4 biti 2 sayı tabanında 0 ve 15 arasında bir değere karşılık gelir(2 sayı tabanında 0000 değerinden 1111 değerine kadar). Bu değer de j olsun. Şimdi S tablosunda i. satır ve j. sütundaki değere bakılır. Bu 0 ve 15 arasında bir değerdir ve 4 bitlik bir blok ile temsil edilir. Bu blok B girişi için S_1 tablosundaki $S_1(B)$ çıkışıdır.
Örneğin, B = 011011 girişi için ilk ve son bitler 0 ve 1'dir. Bu durumda satır değeri 01 olur. Bu S kutusu tablosunda birinci satıra karşılık gelir. B girişi için ortadaki 4 bit 1101 değerindedir. Bu durumda sütun değeri 2 sayı tabanında 1101 olur ve bu da S kutusu tablosunda 13. sütuna karşılık gelmektedir. Tabloda birinci satırda ve 13. sütunda 5 değeri bulunmaktadır. Bu değerin 2 sayı tabanında karşılığı 0101 değeridir. Böylece S fonksiyonunun sonuç değeri bulunmuş olur. $S_1(011011) = 0101$

Burada S_1 tablosu için hesaplamalar yapılmıştır. Benzer şekilde diğer tablolar için de hesaplamalar yapılır ve böylece $S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8)$ ifadesinin değeri bulunur. Aşağıda Tablo 3.7 ve Tablo 3.8'de $S_1,...,S_8$ tabloları görülmektedir.

Tablo 3.7: S Tabloları

S1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

Tablo 3.8: S Tabloları Devamı

S4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

S kutusu tabloları ile yapılan hesaplamalarda n=1 için ilk turda aşağıdaki sonuç elde edilir:

K_1 + **E**(R_0) = 011000 010001 011110 111010 100001 100110 010100 100111.

$S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8)$ = 0101 1100 1000 0010 1011 0101 1001 0111

DES algoritmasında f fonksiyonunun hesaplanması için yapılacak son işlem, S kutusu tablolarının verdiği sonuç değeri üzerinde bir P permütasyonu yapmaktır. S kutusu tablolarının verdiği sonuç üzerinde P permütasyonu yapılarak f fonksiyonunun vereceği sonuç değeri bulunur.

$$f = \mathbf{P}(S_1(B_1)S_2(B_2)...S_8(B_8))$$

P permütasyonu Tablo 3.9'da tanımlanmıştır. P permütasyonu 32 bitlik bir giriş değeri üzerinde işlem yapar ve 32 bitlik bir çıkış üretir.

Tablo 3.9: P Tablosu

P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

S kutusu tablolarının verdiği sonuç üzerinde P permütasyonu yapılarak aşağıdaki sonuç elde edilir:

$S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8)$ = 0101 1100 1000 0010 1011 0101 1001 0111

f = 0010 0011 0100 1010 1010 1001 1011 1011

$R_1 = L_0 + f(R_0 , K_1)$

= 1100 1100 0000 0000 1100 1100 1111 1111
+ 0010 0011 0100 1010 1010 1001 1011 1011
= 1110 1111 0100 1010 0110 0101 0100 0100

Bir sonraki turda n=2 için $L_2 = R_1$ olur. Daha sonra $R_2 = L_1 + f(R_1, K_2)$ değerinin hesaplanması gerekir. Bu şekilde 16 tur boyunca gerekli hesaplamalar yapılır ve 16. turun sonunda L_{16} ve R_{16} blokları elde edilir. Daha sonra bu blokların sırası ters çevrilir ve böylece 64 bitlik $R_{16}L_{16}$ bloğu elde edilir.

DES algoritmasında yapılacak son işlem $R_{16}L_{16}$ bloğuna $\mathbf{IP^{-1}}$ permütasyonunu uygulamaktır. $\mathbf{IP^{-1}}$ permütasyonu Tablo 3 .10'da görülmektedir.

Tablo 3.10: IP^{-1} Permütasyonu

IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Tablo 3.10'a göre bu permütasyonun giriş değerinin soldan 40. biti çıkış değerinin ilk bitidir, soldan 8. biti çıkış değerinin ikinci bitidir ve soldan 25. biti de çıkış değerinin son bitidir.

Elde bulunan örnekte, 1<=n<=16 için gerekli hesaplamalar yapılarak sonuncu turda aşağıdaki değerler elde edilir:

L_{16} = 0100 0011 0100 0010 0011 0010 0011 0100
R_{16} = 0000 1010 0100 1100 1101 1001 1001 0101

Daha sonra bu blokların sırası ters çevrilir ve sırası ters çevrilmiş bloğa da $\mathbf{IP^{-1}}$ permütasyonu uygulanır.

$R_{16}L_{16}$ = 00001010 01001100 11011001 10010101 01000011 01000010 00110010 00110100

IP^{-1} = 10000101 11101000 00010011 01010100 00001111 00001010 10110100 00000101

Böylece DES algoritması verilen bir mesajı şifrelemiş bulunmaktadır. Şifrelenmiş mesaj $\mathbf{IP^{-1}}$ değeridir. Elde bulunan $\mathbf{IP^{-1}}$ değerinin 16 sayı tabanındaki karşılığı 85E813540F0AB405 değeridir. Bu değer başlangıçta elde bulunan **M** = 0123456789ABCDEF mesajının şifrelenmiş halidir.

C = 85E813540F0AB405

DES algoritması şifreleme işlemini bu şekilde gerçekleştirir. Parola çözme işlemi için aynı adımlar uygulanır; ancak bu işlemde geçici anahtarlar sondan başa doğru ve şifreleme işlemine göre ters sırada uygulanır.

3-DES algoritması DES algoritmasının özel bir halidir. 3-DES algoritmasında iki adet 56 bitlik anahtar kullanılır. İlk anahtar verilen bir mesajı şifrelemek için kullanılır. İkinci anahtar ise şifrelenen mesajı çözmek için kullanılır. İkinci anahtar birinci anahtardan farklı olduğu için ikinci işlemde parola çözme işlemi yerine tekrar şifreleme işlemi gerçekleştirilmiş olur. Böylece verilen mesaj art arda iki kez şifrelenmiş olur. Son olarak mesaj birinci anahtar kullanılarak tekrar şifrelenir ve böylece başlangıçta elimizde bulunan mesaj art arda üç kez şifrelenmiş olur. 3-DES algoritması bu şekilde çalışır.