BIL 470 KRİPTOGRAFİ ve BİLGİSAYAR GÜVENLİĞİ DÖNEM ÖDEVİ

ESRA NUR ARICAN
161044028

Araştırma :

OSI temel referans modelinin uygulama katmanında (katman 7), ağ katmanında (katman 3) ve taşıma katmanında (katman 4) kriptografik protokollerin uygulanmasının göreli avantajları ve dezavantajlarını araştırarak öneklerle karşılaştırmalı olarak açıklayın.

OSI Modeli Nedir?

OSI modeli (Open System Interconnection) yedi katmandaki protokolleri uygulamak için bir bilgisayar ağ çerçevesi tanımlar. Ağ oluşturma terimlerindeki bir protokol, bir tür müzakere ve iki ağ kuruluşu arasında kuraldır. Amaç iki bilgisayar arasındaki iletişimin nasıl olacağını tanımlamaktır.

OSI Modeli herhangi bir donanım ya da bilgisayar ağı tipine göre değişiklik göstermez. OSI' nin amacı ağ mimarilerinin ve protokollerinin bir ağ ürünü bileşeni gibi kullanılmasını sağlamaktır. OSI modeli 7 katmana ayrılmıştır.

Bu katmanlar şöyle sıralanmaktadır:

1- Fiziksel Katman (Physical Layer)
2- Veri Bağı Katmanı (DataLink Layer)
3- Ağ Katmanı (Network Layer
4- Taşıma Katmanı (Transport Layer)
5- Oturum Katmanı (Session Layer)
6- Sunum Katmanı (Presantation Layer)
7- Uygulama Katmanı (Application Layer)

UYGULAMA KATMANI (KATMAN 7)

Uygulama katmanı bilgisayar uygulaması ile ağ arasında bir arabirim sağlar. Bu katman, kullanıcıların ve programların ağı kullanabilmesi için araçlar sunar. OSI katmanları arasında sadece bu katman diğer katmanlara servis sağlamaz. Uygulamaların ağ üzerinde çalışması sağlanır.
Uygulama katmanı ağ servisini kullanacak olan programdır. Bu katman kullanıcıların gereksinimini karşılar. SSH, telnet, FTP, TFTP, SMTP, SNMP, HTTP, DNS, HTTPS protokolleri ve tarayıcılar bu katmanda çalışır.
E-posta ve veritabanı gibi uygulamalar bu katman aracılığıyla yapılır.

HTTPS (Güvenli Metin Aktarma İletişim Protokolü) Protokolü

Güvenli metin aktarma iletişim protokolü olarak bilinen bu protokol, HTTP kullanımlarına eklenen SSL (Güvenli Soket Katmanı) ile oluşturulmuş bir sistemdir.

Web siteleri ve kullanıcıların arasındaki güvenliği sağlayan bir şifreleme protokolüdür. İnternet sunucularının ve web sayfalarının güvenli olması, e-ticaret sistemlerinde kullanıcıların banka hesap bilgileri, kredi kart bilgilerinin alınmaması açısından gerekli bir uygulamadır.

HTTPS protokolü kullanan sitelerde, alışveriş ve ödeme esnasında 3. kişiler ya da sistemler tarafından bilgilerin güvenliği çalınmaması için şifreleme sistemi sağlanır. SSL eklenen yani HTTP protokolünden HTTPS protokolüne geçilen sitelerde, güvenlik ve şifreleme teknikleri sağlanmış olur.

Avantajları

1.Veri Şifreleme

HTTP, istemci ve sunucu arasında veri aktarırken verileri şifrelemez. Herhangi biri / bilgisayar korsanı araya girerse, veriler bilgisayar korsanının eline geçebilir. HTTPS üzerinden aktarılan veriler şifrelenir. Bilgisayar korsanı veri aktarımı sırasında verileri alsa bile, veriler zaten şifrelenmiş olduğundan hiçbir şey yapamaz.

2.Veri Koruma

HTTPS, istemci sistemde herhangi bir tür veriyi (veya çok sınırlı) kaydetmez. Böylece HTTPS, veri korumasını sağlar.

3.Sunucu Kimlik Doğrulaması

HTTPS kullanıyorsanız, adres çubuğunda bir asma kilit görebilirsiniz. Üzerine tıklarsanız, sunucunun kimliğini doğrulayabilirsiniz. Sisteminizde depolanan tüm çerezleri, yetkili sertifikaları ve sistem kaynaklarına erişim iznini kontrol edebilirsiniz.

Sertifika ve web sitesi politikaları uyuşmuyorsa, kullanıcı bildirimi güvenli olmayan bir bağlantı olarak alacak ve devam etmek için izninizi isteyecektir.

Özetle, HTTPS herhangi bir kişisel veriyi geçersiz bir sunucuya beslemeden önce ayrılma fırsatı verir.

4.Veri Aktarımından Önce Anlaşma

HTTPS protokolünde verileri aktarmak için tokalaşma yapılması gerekir.

Bu, gönderen, alıcı ve tüm ara bileşenler dahil olmak üzere tüm veri aktarımlarının ve bileşenlerin doğrulanmasını sağlar. Doğrulamalar başarılı olursa, veri aktarımı gerçekleşir. Doğrulama başarısız olursa, tüm veri aktarım etkinliğini iptal edilir.

5.Arama Motoru Görünürlüğü

HTTPS, Google ve diğer arama motorları tarafından arama motoru görünürlüğü için HTTP'den daha fazla tercih edilir. Kullanıcı bir web sitesi kuruyorsa ve para transferi, kullanıcı adı ve şifre gibi önemli verilerle ilgileniyorsa; HTTP yerine HTTPS kullanması daha avantajlı olur.

Dezavantajları

Veri aktarımının fazladan bir ek yükü olur. Bu yük, verileri şifreleme ve şifresini çözme süresini, şifrelenmiş veriler için fazladan başlık girişini ve verileri aktarmadan önce el sıkışmayı içerir. Ayrıca HTTPS kullanımında SSL sertifikası için ödeme yapılması gerekir.

TAŞIMA KATMANI (KATMAN 4)

Bu katman kaynak tarafından gönderilen veriyi parçalara (segment) bölerek hedefe iletilmesini sağlar bu katman diğer katmanlar tarafından yapılan hata denetimi işlemlerinin sonucu olarak hata düzeltme işlemini yerine getirir. Kaynak ve hedef arasındaki akış denetimi bu katmanda yapılır. Üst katmanlara taşıma servisi sağlamasının yanında ayrıca ağın servis kalitesini arttırır. Verinin uçtan uca iletimini sağlar, zamanında ulaşıp ulaşmadığını kontrol eder. Bu katmanın iki önemli protokolü TCP (İletim Kontrol Protokolü) ve UDP (Kullanıcı Veribloğu İletişim Protokolü) dir. Her ikisinin de farklı kullanım alanları ve amaçları vardır. Her uygulama için farklı farklı tanımlanan port adresleri bu katmanın önemli bileşenlerinden biridir. Bu katmanda kullanılan önemli protokoller şunlardır:

TCP, UDP, SPX, RTP, SIP, H.323

İLETİM KONTROL PROTOKOLÜ (TCP)

TCP bağlantı tabanlı olduğu için, herhangi bir şey yapmadan önce bir bağlantı üzerinde görüşmemiz gerekir. Bu prosedür, üç yönlü el sıkışma olarak bilinir.

İlk olarak, başlatan alıcıya bir bağlantı kurmak isteyip istemediğini soracaktır.

Alıcı, isteğe cevap verecektir ve başlatıcı bu yanıtı aldığında, alıcıya bağlantının şimdi kurulduğunu onaylayan bir paket gönderecektir. Benzer prosedür, bir bağlantı kapatıldığında gerçekleşir.

Avantajları

TCP her zaman üç şeyi garanti eder : verileriniz hedefine ulaşır, oraya zamanında ulaşır ve oraya kopyalanmadan ulaşır.

TCP'de, tüm iş işletim sistemi tarafından yapılır, hata ayıklama bile işletim sistemi tarafından halledilir.

Verileri otomatik olarak paketlere ayırır.

1.Yeniden İletim

Bir gönderici, belirli bir süre sonra bir onay almadığında, paketin yolda kaybolduğunu varsayacaktır. Böylece tekrar gönderecek.

2.Siparişte Teslimat

Segmentler TCP'de sıralandığından, sırayla teslimatı gerçekleştirir.

Paketler sıra dışı olsa da TCP bunları uygulamaya göndermeden önce yeniden düzenler.

3.Tıkanıklık Kontrolü

Ağ yoğun olduğunda iletimi geciktirir.

4.Hata Tespiti

Dezavantajları

Veriler her zaman anında gönderilmez. Buna tıkanıklık kontrolünün yan etkisi diyebiliriz. Bu nedenle gerçek zamanlı sohbet uygulamaları istenildiği gibi gerçek zamanlı hissettirmeyebilir. Tıkanıklık kontrolü bu açıdan bakılınca avantajlı veya dezavantajlı olabilir.

Tüm işlemler işletim sistemi tarafından yapıldığından, işletim sisteminizde hatalar varsa, internette gezinme ve içerik indirme gibi birçok sorunla karşılaşabilirsiniz.

TCP, yayın ve çok noktaya yayın bağlantıları için kullanılamaz.

Paketlerin yeniden iletimi ve paketlerin kabulüyle daha büyük ek yük oluşur. Örnek vermek gerekirse HD video göndermek için çok fazla bant genişliğine ihtiyacımız vardır. Ancak TCP kullanıyorsanız, yeniden iletim ve onay için bu bant genişliğinden daha da fazlasına ihtiyaç duyulur. Böyle bir durumda TCP yerine UDP kullanmak mantıklıdır.

AĞ KATMANI (KATMAN 3)

Ağ katmanı veri paketine farklı bir ağa gönderilmesi gerektiğinde yönlendiricilerin kullanacağı bilginin eklendiği katmandır. Bu katmanda veriler paket olarak taşınır.

Ağ katmanında iki istasyon arasında en ekonomik yoldan verinin iletimi kontrol edilir. Bu katman sayesinde verinin yönlendiriciler aracılığıyla yönlendirilmesi sağlanır.

Ağ aşamasında mesajlar adreslenir ayrıca mantıksal adresler fiziksel adreslere çevrilir. Bu aşamada ağ trafiği, yönlendirme gibi işlemler de yapılır.

IP protokolü bu katmanda çalışır. Bu katman ağ adresi (IP adresi) verinin kaynaktan hedefe ulaşmasını sağlar. Bu katman verinin kaynak ve hedef IP adresi eklenmiş şekline paket ismi verilir. IP paketi adreslerle birlikte paketin toplam boyutu, TTL, servis tipi, versiyon, hata denetimi gibi bilgiler barındırır. Ağ ortamında veriyi yönlendiren yönlendiriciler ve yönlendirme algoritmaları bu katmanda çalışır.

Bu katmanda kullanılan protokollerden bazıları şunlardır; IP, IPX, IPSec, AppleTalk DDP, ARP, RARP, ICMP, RIP, EIGRP.

İNTERNET PROTOKOLÜ GÜVENLİĞİ (IPsec)

“Internet Protokolü (IP) Güvenliği” veya “IP Güvenlik Protokolü” olarak da bilinen IPsec, IP ağ trafiği için oluşturulmuş bir güvenlik hizmeti altyapısını tanımlar. IPsec, IP katmanında veri güvenliğinin sağlanmasına yönelik bir çerçeve ortaya koyar, bu güvenliği sağlamak için IP ağ paketlerinin kimlik doğrulaması ve şifrelenmesi yoluyla tasarlanmış protokolleri ifade eder. IPsec, paketleri şifreleme, şifresini çözme ve kimlik doğrulamasını yapmada kullanılan şifreleme algoritmalarını tanımlar. Bunun yanı sıra güvenli anahtar değişimi ve anahtar yönetimi için gerekli olan protokoller de IPsec kapsamındadır.

IPsec IP paketlerinde güvenliği sağlamak için ilk olarak iki mekanizma tanımlamıştır: IP paketlerinde verileri şifrelemenin bir yolu olan Encapsulating Security Payload (ESP) protokolü ve IP paketlerini dijital olarak imzalamanın bir yolu olan Authentication Header (AH) protokolü. IPsec sunucuları tarafından kullanılan kriptografik anahtarları yönetmek için ise Internet Key Exchange (IKE) protokolü kullanılır.

IPsec’in temel kullanım amacı, ağ verisini korumaktır. Bu, her VPN bağlantısında olduğu gibi IPsec tünelleme kullanarak, iki uç nokta arasında iletilen verinin şifrelendiği bağlantılar kurarak yapılabilir. Bunun yanı sıra IPsec, ağ verisini uygulama katmanında şifreleyerek veya verilerini herkese açık internet üzerinden ileten router özellikli cihazların güvenliğini sağlayarak da koruyabilir. Dahası, şifreleme olmadan da IPsec kimlik doğrulamada kullanılabilir, örneğin verinin bilinen bir göndericiden geldiğini teyit edebilir.

İnternet trafiği, bir sunucu ile diğeri arasında IPsec kullanılmadan da güvenlik altına alınabilir. Örneğin, uygulama katmanında (OSI modelinin 7. katmanı) HTTP Secure (HTTPS) ile veya aktarım katmanında (OSI modelinin 4. katmanı) Transport Layer Security (TLS) protokolüyle veri şifrelenerek güvenliği sağlanabilir. Ancak, bu yüksek katmanlarda şifreleme veya kimlik doğrulaması kullanıldığında, tehdit unsurları, gizli kalması gereken verileri ortaya çıkarabilecek protokol bilgilerine müdahale edebilir.

Avantajları

1. Ağ katmanı güvenliği

IPSec, ağ katmanı olan katman 3'te çalışır. Sonuç olarak, daha yüksek ağ katmanı üzerinde hiçbir etkisi yoktur. Diğer bir deyişle, IPSec'in en büyük avantajlarından biri, uygulamalara karşı şeffaf olmasıdır. Son kullanıcının IPSec veya yapılandırması hakkında endişelenmesine gerek yoktur.

Ek olarak, ağ katmanında çalıştığı için IPSec, ağ üzerinden geçen tüm trafiğin izlenmesine izin verir. Bu nedenle, Destek Mühendislerimiz ağa giren ve çıkan tüm trafik için korumaya ihtiyaç duyan müşteriler için IPsec tabanlı VPN'leri önermektedir.

2. Gizlilik

Benzer şekilde, IPSec'in ikinci avantajı da gizlilik sağlamasıdır. Herhangi bir veri alışverişi sırasında IPSec, gizli verilerin güvenli bir şekilde aktarılmasına yardımcı olan genel anahtarları kullanır. Sonuç olarak, anahtarların güvenliğini sağlamak, güvenli veri aktarımını garanti eder. Ayrıca bu anahtarlar, verilerin doğru ana bilgisayardan geldiğini doğrulamaya yardımcı olur. Bu nedenle, veri paketlerini taklit etmek oldukça imkansız hale gelir. Bu nedenle, Sunucu Yöneticilerimiz genel anahtarları gönderirken her zaman güvenliği sağlar.

3. Uygulamaya sıfır güvenilirlik

IPSec güvenliği ağ katmanında uygulanmaktadır. Böylece kullanılan uygulamalara bağlı değildir.

IPSec yalnızca işletim sisteminde değişiklik yapılmasını gerektirir. Sonuç olarak, IPsec tabanlı VPN'lerin de uygulama türü konusunda endişelenmesine gerek kalmaz. Bireysel uygulamalarda değişiklik yapılmasını gerektiren SSL tabanlı VPN'lerde durum böyle değildir. Bu, IPSec'in popülaritesinin bir başka nedenidir.

Dezavantajları

1. Geniş erişim aralığı

IPSec'in en büyük dezavantajlarından biri geniş erişim aralığıdır. IPSec tabanlı ağda tek bir cihaza erişim verilmesi, diğer cihazlar için de erişim ayrıcalıkları sağlayabilir.

Örneğin, IPSec tabanlı ev ağından kurumsal bir ağa bağlandığımızda burada ev ağındaki herhangi bir bilgisayarda kötü amaçlı yazılım varsa, kurumsal ağdaki bilgisayarlara kolayca yayılabilir.

Özel güvenlik mekanizmaları olmadığı sürece, IP katmanında bulunan güvenlik açıkları IPSec tüneli üzerinden kurumsal ağa aktarılacaktır.

2. Uyumluluk sorunları

IPSec, yazılımla da birkaç uyumluluk sorunu getirir. Bu, yazılım geliştiricileri IPSec standartlarına uymadığında gerçekleşir. Benzer şekilde, zaten IPSec tabanlı VPN üzerindeyken, başka bir ağa bağlanmak, güvenlik duvarlarındaki kısıtlamalar nedeniyle oldukça imkansız olacaktır.

Yine IPsec, çoklu protokol ve IP çoklu yayın trafiği için destek sağlamaz.

3. CPU Ek Yükü

Ne yazık ki IPSec, yüksek CPU kullanımıyla tanınır. Sunucudan geçen tüm verileri şifrelemek ve şifresini çözmek için oldukça fazla işlem gücü gerektirir. Veri paketi boyutu küçük olduğunda, IPsec tarafından kullanılan büyük ek yük nedeniyle ağın performansı düşer.

4. Kırık Algoritmalar

IPSec'te kullanılan belirli algoritmaların güvenliği bir endişe kaynağıdır. Birisi bu bozuk algoritmaları kullanırsa, sunucu bilgisayar korsanları tarafından ele geçirilme riski altında olacaktır. Neyse ki, bilinen güvenlik açıklarının üstesinden gelen, kullanıma hazır daha yeni ve karmaşık algoritmalar vardır. Bilgisayar korsanlığı riskinden kaçınmak için, IPSec kullanırken, doğru algoritmalar seçmek gerekir.

Programlama Projesi:

C veya phyton ile gerçekleştirilecek olan bu araçta şifreleme/deşifreleme ve Özüt alma, dosya bütünlüğünün denetimi yöntemleri bizzat gerçeklenecek olup, arşiv/API kullanılmayacaktır. Gerçeklenenen Programların kaynak kodları açıklamalı olarak verilecektir;

a) AES şifreleme algoritmasının gerçeklenmesi ve şifreleme/deşifrelemede kullanılması (test verileri ile birlikte).

b) Gerçeklenen Şimetrik şifreleme algoritması kullanılarak CBC ve OFB modlarında çalışmayı gerçekleyip testlerini yapacak şekle getiriniz.

c) Herhangi bir doküman (.doc/.docx, .pdf, ppt, xls vs) üzerinde değişiklik yapılıp yapılmadığını ve yapanın kimliğini anlamak için, özütünü alacak ve sadece işlem yapan kişinin bildiği bir anahtar ile şifreleyip dosyanın sonuna ekleyecek bir araç (b şıkkındaki gerçeklemeyi özüt fonk. Olarak kullanınızkullanınız)

d) Dosyanın bütünlüğünün değişip değişmediğinin kontrolü için, c)deki işlemleri yaparak ilk üretilen özüt değer ile karşılaştıran doğrulama aracını gerçekleyerek örnek testleri gösteriniz.

Kodların açıklanması

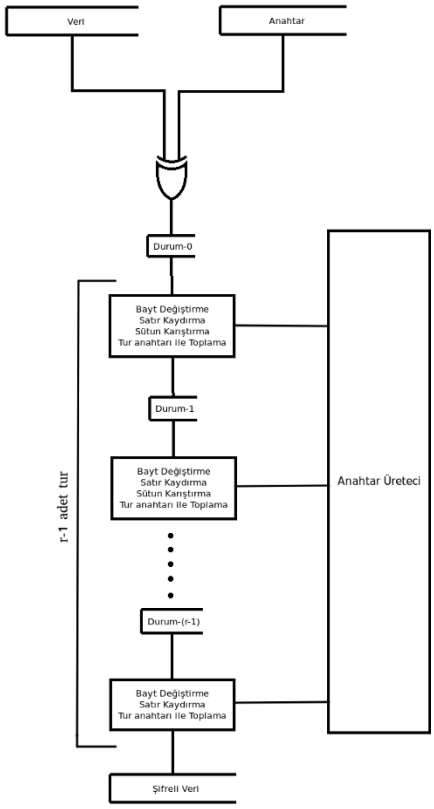
AES simetrik bir şifreleme algoritmasıdır yani hem şifreleme hem şifre çözme işlemleri için aynı anahtar kullanılır. AES için girdi ve çıktı matrisleri her zaman 128 bit olmak zorundadır ancak anahtar uzunluğu 128, 192 veya 256 bit olabilir.

```
class AES(object):
```
AES algoritmasını gerçekleştirecek sınıf oluşturuldu

Daha sonra anahtar boyutları belirlendi.

```
keySize = dict(SIZE_128=16, SIZE_192=24, SIZE_256=32)
```

AES algoritması Bayt Değiştirme, Satır Kaydırma, Sütun Karıştırma ve Tur Anahtarı ile Toplama gibi adımların tekrar etmesiyle oluşturulur. 128 bit AES şifrelemesi için 10, 192 bit için 12, 256 için ise 14 tur sonunda şifrelenmiş mesaja ulaşılır.



Yandaki şekil AES algoritmasının genel akışını gösterir.

```
sbox =  [0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67,
        0x2b, 0xfe, 0xd7, 0xab, 0x76, 0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59,
        0x47, 0xf0, 0xad, 0xd4, 0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0, 0xb7,
        0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc, 0x34, 0xa5, 0xe5, 0xf1,
        0x71, 0xd8, 0x31, 0x15, 0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05,
        0x9a, 0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75, 0x09, 0x83,
        0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29,
        0xe3, 0x2f, 0x84, 0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b,
        0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf, 0xd0, 0xef, 0xaa,
        0xfb, 0x43, 0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c,
        0x9f, 0xa8, 0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5, 0xbc,
        0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2, 0xcd, 0x0c, 0x13, 0xec,
        0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19,
        0x73, 0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee,
        0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb, 0xe0, 0x32, 0x3a, 0x0a, 0x49,
        0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79,
        0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4,
        0xea, 0x65, 0x7a, 0xae, 0x08, 0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6,
        0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd, 0x8b, 0x8a, 0x70,
        0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9,
        0x86, 0xc1, 0x1d, 0x9e, 0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e,
        0x94, 0x9b, 0x1e, 0x87, 0xe9, 0xce, 0x55, 0x28, 0xdf, 0x8c, 0xa1,
        0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d, 0x0f, 0xb0,
        0x54, 0xbb, 0x16]
```

Bayt değiştirme işlemi durum matrisindeki baytların farklı baytlara dönüştürülmesi işlemidir. Bu işlem için gerekli S-box tabloları kod içerisinde tanımlanmıştır.

Daha sonra satır kaydırma, sütun karıştırma, tur anahtarı ile toplama ve tur anahtarının oluşturulması aşamaları kod üzerinde gerçekleştirilmiştir.

```
def createRoundKey(self, expandedKey, roundKeyPointer):
    #Creates a round key.
```

```
def addRoundKey(self, state, roundKey):
    #Adds (XORs) the round key to the state.
```

```
# each iteration shifts the row to the left by 1
def shiftRow(self, state, statePointer, nbr, isInv):
```

```
def mixColumn(self, column, isInv):
```

Deşifreleme

Deşifreleme algoritmasının akışı yukarıda AES algoritması için verilen şekildeki akış adımlarının terslerini içermektedir. Ters Bayt Değiştirme, Ters Satır Kaydırma, Ters Sütun Karıştırma ve Tur Anahtarıyla Toplama adımlarını içerir ancak Tur Anahtarıyla Toplama adımı genişletilmiş anahtarın içerisindeki son anahtardan başlayarak geriye doğru ilerler.
Ters bayt değiştirme işlemi için bayt değiştirme işleminde olduğu gibi bir tablodan faydalanılır. Bu tablodaki veriler bayt değiştirme tablosundaki verilerin tersleridir.

```
# Inverted S-box
s_invertBox = [0x52, 0x09, 0x6a, 0xd5, 0x30, 0x36, 0xa5, 0x38, 0xbf, 0x40, 0xa3,
        0x9e, 0x81, 0xf3, 0xd7, 0xfb , 0x7c, 0xe3, 0x39, 0x82, 0x9b, 0x2f,
        0xff, 0x87, 0x34, 0x8e, 0x43, 0x44, 0xc4, 0xde, 0xe9, 0xcb , 0x54,
        0x7b, 0x94, 0x32, 0xa6, 0xc2, 0x23, 0x3d, 0xee, 0x4c, 0x95, 0x0b,
        0x42, 0xfa, 0xc3, 0x4e , 0x08, 0x2e, 0xa1, 0x66, 0x28, 0xd9, 0x24,
        0xb2, 0x76, 0x5b, 0xa2, 0x49, 0x6d, 0x8b, 0xd1, 0x25 , 0x72, 0xf8,
        0xf6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xd4, 0xa4, 0x5c, 0xcc, 0x5d,
        0x65, 0xb6, 0x92 , 0x6c, 0x70, 0x48, 0x50, 0xfd, 0xed, 0xb9, 0xda,
        0x5e, 0x15, 0x46, 0x57, 0xa7, 0x8d, 0x9d, 0x84 , 0x90, 0xd8, 0xab,
        0x00, 0x8c, 0xbc, 0xd3, 0x0a, 0xf7, 0xe4, 0x58, 0x05, 0xb8, 0xb3,
        0x45, 0x06 , 0xd0, 0x2c, 0x1e, 0x8f, 0xca, 0x3f, 0x0f, 0x02, 0xc1,
        0xaf, 0xbd, 0x03, 0x01, 0x13, 0x8a, 0x6b , 0x3a, 0x91, 0x11, 0x41,
        0x4f, 0x67, 0xdc, 0xea, 0x97, 0xf2, 0xcf, 0xce, 0xf0, 0xb4, 0xe6,
        0x73 , 0x96, 0xac, 0x74, 0x22, 0xe7, 0xad, 0x35, 0x85, 0xe2, 0xf9,
        0x37, 0xe8, 0x1c, 0x75, 0xdf, 0x6e , 0x47, 0xf1, 0x1a, 0x71, 0x1d,
        0x29, 0xc5, 0x89, 0x6f, 0xb7, 0x62, 0x0e, 0xaa, 0x18, 0xbe, 0x1b ,
        0xfc, 0x56, 0x3e, 0x4b, 0xc6, 0xd2, 0x79, 0x20, 0x9a, 0xdb, 0xc0,
        0xfe, 0x78, 0xcd, 0x5a, 0xf4 , 0x1f, 0xdd, 0xa8, 0x33, 0x88, 0x07,
        0xc7, 0x31, 0xb1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xec, 0x5f , 0x60,
        0x51, 0x7f, 0xa9, 0x19, 0xb5, 0x4a, 0x0d, 0x2d, 0xe5, 0x7a, 0x9f,
        0x93, 0xc9, 0x9c, 0xef , 0xa0, 0xe0, 0x3b, 0x4d, 0xae, 0x2a, 0xf5,
        0xb0, 0xc8, 0xeb, 0xbb, 0x3c, 0x83, 0x53, 0x99, 0x61 , 0x17, 0x2b,
        0x04, 0x7e, 0xba, 0x77, 0xd6, 0x26, 0xe1, 0x69, 0x14, 0x63, 0x55,
        0x21, 0x0c, 0x7d]
```

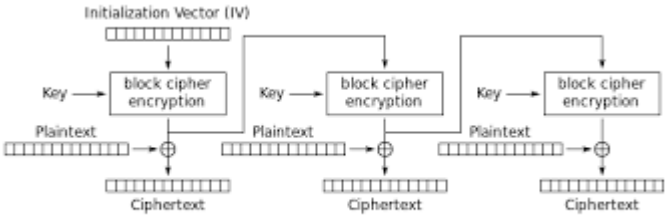
Ters Değiştirme Tablosu

Ters satır kaydırma adımında, satır kaydırma adımında yapılan işlemler sağa kaydırılarak tekrarlanır.

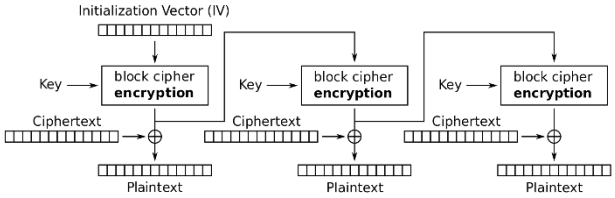
Kod üzerinde AES algoritmasının AES sınıfında gerçekleştirilmesinden sonra, bu algoritma kullanılarak CBC ve OFB modlarında çalışacak bir sınıf tasarlandı. Ve bu sınıf içinde AES algoritmasını gerçekleştirdiğimiz sınıf tipinde bir obje oluşturuldu.

```
class AESModeOfOperation(object):
    #Handles AES with plaintext consisting of multiple blocks.
    #Choice of block encoding modes:  OFB,CBC
    aes = AES()
```

Daha sonra şifreleme ve deşifreleme işlemlerini yapacak fonksiyonlar yazıldı ve bunların içinde AES algoritmasında yazılan şifreleme ve deşifreleme yöntemleri kullanıldı.
OFB MODU: OFB (Çıktı Geri Bildirim) çalışma modu ayrıca bir blok şifreleyicinin bir akış şifreleyici olarak kullanılmasını sağlar. Ayrıca dolgu verilerine ihtiyaç duymaz. Metni datalara bölmektedir. Şifreleme algoritması düz metin üzerinde değil de IV (İlklendirme vektörü: çeşitli kipler tarafından şifrelemeyi rassallaştırmak için kullanılan ve dolayısıyla aynı açık metnin tekrar şifrelendiği durumlarda anahtar değiştirmeden farklı şifreli metin üretilmesini sağlayan bir bit bloğudur) ve anahtar(key) üzerinde uygulanmaktadır.

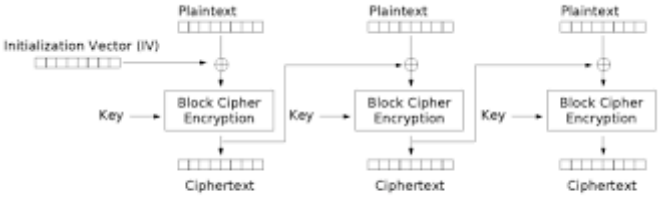


Output Feedback (OFB) mode encryption

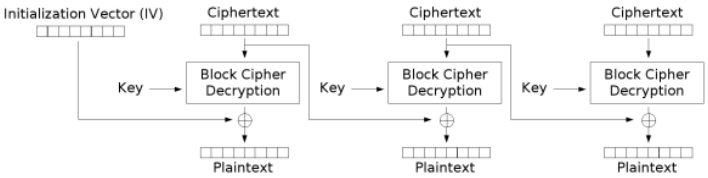


Output Feedback (OFB) mode decryption

CBC MODU: (Şifreleme Bloğu Zincirleme) Veriyi doldurma işlemi yapmaz, veri ile aynı büyüklükte çıktı verir. CBC, her bloğun kendisinden sonra ki blok ile, şifreleme işlemi öncesi bire bir XOR işlemine girmesidir.



Cipher Block Chaining (CBC) mode encryption



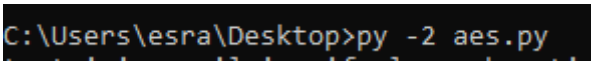
Cipher Block Chaining (CBC) mode decryption

Bu modların gerçekleştirilmesi işlemi şu şekilde yapılmıştır:

```
if mode == self.modeOfOperation["OFB"]:
    if firstRound:
        output = self.aes.encrypt(IV, key, size)
        firstRound = False
    else:
        output = self.aes.encrypt(input, key, size)
    for i in range(16):
        if len(output)-1 < i:
            plaintext[i] = 0 ^ ciphertext[i]
        elif len(ciphertext)-1 < i:
            plaintext[i] = output[i] ^ 0
        elif len(output)-1 < i and len(ciphertext) < i:
            plaintext[i] = 0 ^ 0
        else:
            plaintext[i] = output[i] ^ ciphertext[i]
    for k in range(end-start):
        chrOut.append(chr(plaintext[k]))
    input = output
elif mode == self.modeOfOperation["CBC"]:
    output = self.aes.decrypt(ciphertext, key, size)
    for i in range(16):
        if firstRound:
            plaintext[i] = IV[i] ^ output[i]
        else:
            plaintext[i] = input[i] ^ output[i]
    firstRound = False
    if originalsize is not None and originalsize < end:
        for k in range(originalsize-start):
            chrOut.append(chr(plaintext[k]))
    else:
        for k in range(end-start):
            chrOut.append(chr(plaintext[k]))
    input = ciphertext
```

Programın Test Sonuçları

Yazılan program Windows ortamında python 2 kullanarak çalıştırılmıştır.



CBC modu ile çalıştırılan program çıktısı:

```
C:\Users\esra\Desktop>py -2 aes.py
test icin yazilmis sifrelenecek metin
Rastgele anahtar testi mod: CBC
Acik metin: test icin yazilmis sifrelenecek metin
Anahtar: [243, 178, 207, 249, 40, 101, 164, 166, 153, 249, 53, 225, 12, 248, 130, 16]
Sifreleyici: [115, 37, 144, 253, 201, 117, 112, 21, 101, 145, 218, 252, 22, 88, 94, 241, 119, 16, 51, 30, 198, 246, 53, 32, 13, 80, 234, 137, 248, 13, 198, 65, 69, 104,
 216, 8, 108, 173, 35, 169, 237, 228, 208, 55, 230, 21, 147, 170, 91, 112, 122, 23, 255, 138, 156, 131, 177, 218, 83, 150, 142, 254, 97, 85]
Desifreleyici: test icin yazilmis sifrelenecek metin

C:\Users\esra\Desktop>
```

OFB modu ile çalıştırılan program çıktısı:

```
C:\Users\esra\Desktop>py -2 aes.py
test icin yazilmis sifrelenecek metin,2. deneme
Rastgele anahtar testi mod: OFB
Acik metin: test icin yazilmis sifrelenecek metin,2. deneme
Anahtar: [54, 107, 5, 64, 92, 134, 187, 169, 255, 142, 87, 93, 168, 2, 247, 31]
Sifreleyici: [5, 38, 114, 0, 192, 70, 44, 143, 209, 162, 141, 241, 148, 104, 30, 7, 232, 239, 192, 135, 204, 225, 2, 3, 64, 5, 59, 223, 192, 36, 109, 194, 12, 106, 156,
 171, 118, 36, 245, 255, 234, 164, 88, 0, 75, 178, 5, 248, 19, 236, 222, 3, 70, 27, 144, 87, 58, 155, 232, 91, 167, 124, 148]
Desifreleyici: test icin yazilmis sifrelenecek metin,2. deneme

C:\Users\esra\Desktop>
```

Referanslar

https://bidb.itu.edu.tr/seyir-defteri/blog/2013/09/07/osi-katmanlar%C4%B1

https://medium.com/bili%C5%9Fim-hareketi/osi-modeli-ve-7-katman-7c3bb467798c

https://tr.wikipedia.org/wiki/Blok_%C5%9Fifre_%C3%A7al%C4%B1%C5%9Fma_kipleri