SİVİL HAVA TRAFİK RADARLARINDA YER-YER HATLARINDAN GÖNDERİLEN HAVA RESİMLERİNİN ŞİFRELENMESİ

Ayhan ÖZEL

Yüksek Lisans Tezi

Bilgisayar Mühendisliği Anabilim Dalı

Temmuz 2006

JÜRİ VE ENSTİTÜ ONAYI

Ayhan ÖZEL'in "**Sivil Hava Trafik Radarlarında Yer-Yer Hatlarından Gönderilen Hava Resimlerinin Şifrelenmesi"** başlıklı **Bilgisayar Mühendisliği** Anabilim Dalındaki, Yüksek Lisans Tezi 23.06.2006 tarihinde, aşağıdaki jüri tarafından Anadolu Üniversitesi Lisansüstü Eğitim Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

	Adı-Soyadı	**İmza**
Üye (Tez Danışmanı)	**: Prof. Dr. YAŞAR HOŞCAN**	**……**
Üye	**: Yard. Doç. Dr. CÜNEYT AKINLAR**	**……**
Üye	**: Yard. Doç. Dr. HAKAN KAĞNICIOĞLU**	**……**

Anadolu Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ……. tarih ve …… sayılı kararıyla onaylanmıştır.

Enstitü Müdürü

ÖZET

Yüksek Lisans Tezi

SİVİL HAVA TRAFİK RADARLARINDA YER-YER HATLARINDAN GÖNDERİLEN HAVA RESİMLERİNİN ŞİFRELENMESİ

Ayhan ÖZEL

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Yaşar HOŞCAN
2006, 100 sayfa

Günümüzde şifreleme, bilgi güvenliği için anahtar faktör haline gelmektedir. Önemsemediğimiz bir çok bilgi başkalarının dikkatini çekebilir. Sivil Hava Trafik Radarlarının yer-yer hatlarındaki haberleşme bunlardan biridir. Bu haberleşme hassas bilgiler içermektedir ve uçuş emniyeti için çok önemlidir. Haberleşmedeki herhangi bir kontrolsüz değişiklik veya kesinti, havada beklenmedik facialara yol açabilir. Korunmayan verilerin sonucu olarak ortaya çıkabilecek herhangi bir kazayı önlemek için, Sivil Hava Trafik Kontrol Merkezi ile radarlar arasındaki hava resmi haberleşmesi yazılım tabanlı araçlarla, alma ve göndermeden önce şifrelenebilir ve şifresi çözülebilir. Bu çalışmada, veri transferi için, gizli anahtarlı şifreleme yordamı olarak İleri Şifreleme Standardı (Advanced Encryption Standart) yordamı; anahtar transferi için, açık anahtarlı şifreleme yöntemi olarak, RSA (Ronald **R**ivest, Adi **S**hamir ve Leonard **A**dleman) yordamı seçilmiştir.

Çalışmanın bulguları, Sivil Hava Trafik Kontrol Merkezi ile radarlar arasındaki hava resminin, bilgi güvenliğini ve uçuş emniyetini sağlamak amacıyla, herhangi bir kabul edilmeyecek gecikmeye sebebiyet vermeden, yazılım tabanlı şifreleme araçları ile şifrelenebileceğini göstermektedir.

Anahtar Kelimeler : Hava Trafik Kontrol Merkezi, Hava Resmi, Şifreleme, Gizli Anahtarlı Şifreleme, Açık Anahtarlı Şifreleme.

ABSTRACT

Master of Science Thesis

ENCRYPTION OF AIR PICTURE THAT IS TRANSMITTED THROUGH CIVIL AIR TRAFFIC CONTROL RADARS' GROUND TO GROUND LINES

Ayhan ÖZEL

Anadolu University
Graduate School of Sciences
Computer Engineering Program

Supervisor: Prof. Dr. Yaşar HOŞCAN
2006, 100 pages

Nowadays, cryptography is becoming a key factor for information security. Most of the data that we don't care may take attention of others. Civil Air Traffic Control Radars' ground to ground communication is one of them. This communication is very important for flight safety and includes sensitive data to be protected. Any uncontrolled change and break on communication may cause unexpected catastrophe in the skies. In order to prevent any accident which may occur as a result of unprotected data, the air picture communication between Civil Air Traffic Control Center and radars can be encrypted and decrypted with software based tools before sending and receiving air picture. In this study, Advanced Encryption Standard algoritm is selected as symetric encryption algoritm for data transfer and RSA (Ronald **R**ivest, Adi **S**hamir and Leonard **A**dleman) algoritm is selected as public encryption algoritm for key transfer.

The findings of the study show that the air picture between Civil Air Traffic Control Center and radars can be encrypted with software tools without causing any unacceptable delay to provide information security and fligt safety.

Keywords: Air Traffic Control Center, Air Picture, Encryption, Symmetric Encryption, Public Key Encryption.

İÇİNDEKİLER

Sayfa

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

SÖZLÜK ve KISALTMALAR DİZİNİ

Türkçe Açık Hali	Kısaltma	İngilizce Açık Hali	Kısaltma
ABD Federal Veri İşleme Standardı	ABD FVİS	Federal Information Processing Standard	FISP
ABD Milli Standartlar ve Teknoloji Enstitüsü	ABD MSTE	National Institute of Standarts and Technology	NIST
Açık Anahtar		Public Key	
Açık Anahtarlı Şifreleme		Public Key Encryption	
Açık Metin		Plaintext	
Açık Metine Dayanan Saldırılar		Plaintext Attacks	
Akım Şeklinde Şifreleme		Stream Cipher	
Anahtar Dağıtım Sorunu		Key Distribution Problem	
Asal Sayı		Prime Number	
Ayrıcalıklı Veya		Exclusive OR	XOR
Bağımsızlık		Independence	
Blok Halinde Şifreleme		Block Cipher	
Çarpışma		Collision	
Çarpışma Saldırısı		Collision Attack	
Çekirdek Dökümü		Core Dump	
Çıktı Geri Besleme	ÇGB	Output Feedback	OFB
Çırpı		Hash	
Değiştirge		Parameter	
Depolama		Storage	
Diffie-Hellman	DH	Diffie-Hellman	DH
Dijital İmza Standardı	DİS	Digital Signature Standard	DSS
Doğrusal Şifre Çözümleme Saldırısı		Linear Cryptanalysis Attack	
Doğum Günü Saldırısı		Birthday Attack	
Dost Düşman Teşhisi	IFF	Identification Friend or Foe	IFF
Döndürme		Rotation	
Elektronik Kod Kitabı	EKK	Electronic Codebook	ECB
Eliptik Eğri		Elliptic Curve	
En Büyük Ortak Bölen	EBOB	Greatest Common Divisor	GCD
Feistel Ağı		Feistel Network	
Gizleme		Whitening	
Gizli Anahtar		Secret Key	

Gizli Anahtarlı Şifreleme		Symetric Encryption	
Güç Analizi Saldırısı		Power Analysis Attack	
Güvenli Çırpı Yordamı	GÇY	Secure Hashing Algorithm	SHA
Hata Ayıklayıcılar		Debugger	
Hava Resmi		Air Picture	
Hava Trafik Kontrol Merkezi	HTKM	Air Traffic Control Center	ATC
İkame		Substitute	
İki-Balık		Twofish	
İkil		Bit	
İleri Şifreleme Standardı (Rijndael Yordamı)	İŞS	Advanced Encryption Standard (Rijndael Algorithm)	AES
İlgili Anahtar Saldırısı		Related-Key Attack	
Kaydırma Yazmaçları		Shift Registers	
Manyetik Alan		Magnetic Field	
Meksefe, Kondansatör		Capacitor, Condenser	
Merkezi İşlem Birimi	MİB	Central Processing Unit	CPU
Mesaj Özeti 5	MÖ5	Message-Digest Algorithm 5	MD5
Ortada Buluşturma Saldırısı		Meet In The Middle Attack	
Ortadaki Adam Saldırısı		Man In The Middle Attack	
Özel Anahtar		Private Key	
Rassal Sayı		Random Number	
RC6 Yordamı	RC6	Ron's Code 6, Rivest Cipher 6	RC6
Ronald Rivest, Adi Shamir ve Leonard Adleman	RSA	Ronald Rivest, Adi Shamir and Leonard Adleman	RSA
Sabah Yıldızı, Venüs		Lucifer	
Sahte Rassal Sayı		Pseudorandom Number	
Seçilen Anahtar Saldırısı		Chosen-Key Attack	
Sıralandırma		Permutation	
Sırt Çantası		Knapsack	
S-Kutusu		S-Box	
Şifre Blok Zincirleme	ŞBZ	Cipher Block Cahining	CBC
Şifre Çözme		Decryption	
Şifre Geri Besleme	ŞGB	Cipher Feedback	CFB
Şifrebilim		Cryptology	
Şifreleme		Encryption, Cryptography	
Şifrelemeye Karşı Saldırılar		Cryptographic Attacks	

Şifrelenmiş Metin		Ciphertext	
Şifrelenmiş Metine Dayanan Saldırılar		Ciphertext Attacks	
Şifreli Metin Çalma	ŞMÇ	Cipher Text Stealing	CTS
Tam Güç Saldırısı		Brute Force Attack	
Tam Güç Saldırısı		Brute Force Attack	
Tekrar Üretilememe		No Reliable Reproduction	
Tektip Dağılım		Uniform Distribution	
Tohum, Tohumlama		Seed	
Türevsel Şifre Çözümleme Saldırısı		Differential Cryptanalysis Attack	
Uçan-Balık		Blowfish	
Uluslararası Veri Şifreleme Yordamı	UVŞY	International Data Encryption Algorithm	IDEA
Üçlü VŞS		Triple DES	
Veri Şifreleme Standardı	VŞS	Data Encryption Standard	DES
Yan-Kanal Saldırıları		Side-Channel Attacks	
Yeniden Tohumlama		Reseed	
Yılan		Serpent	
Yordam		Algorithm	
Yönlendirici		Router	
Zamanlama Saldırısını		Timing Attack	

1. GİRİŞ

Gelişen teknoloji, iş yaşamına ve günlük hayata birçok konuda yeni kolaylıklar katmakta ancak, birçok sorunu da beraberinde getirmektedir. Bu sorunlardan bir tanesi, internet dünyasından da karşımıza çıkan bilgi güvenliğidir. Bilgi güvenliği, özellikle finansal ve askeri uygulamalarda hayati öneme sahip olduğundan, özel yazılım ve donanıma ihtiyaç duymaktadır. Bilgi güvenliğinin sağlanabilmesi için birçok yöntemden faydalanılmaktadır. Bu yöntemlerden bir tanesi de şifrelemedir. Şifreleme, temel olarak, haberleşme sırasında bilgilerin, istenmeyen kişilerin eline geçmesine engel olmaktadır.

Sivil Hava Trafik Konrol Merkezi ile radarlar arasındaki haberleşme, istenmeyen kişilerce elde edilmemesi ve değiştirilmemesi gereken bilgiler içermektedir. Bu bilgiler, kalkış ve iniş meydanları arasında korunmasız seyahat eden uçakların, anlık koordinat ve irtifa bilgilerinden, teşhis amacıyla kullandıkları IFF mod değerleri ve çağrı adlarına kadar bir çok bilgiden oluşmaktadır. Hava trafiğinin yönetilmesi, bu bilgilere tamamen bağımlı olduğundan; bu bilgilerdeki herhangi bir kötü amaçlı değişiklik, bütün trafiğin felç olmasına sebep olabilir ve telafi edilmesi mümkün olmayan sonuçlar doğurabilir. 7 gün 24 saat üzerinden devam eden hava trafiğinin, bu bilgiler vasıtasıyla, istenmeyen kişilerce takip edilmesi, sivil yolcu uçağı uçuşlarından Devlet Başkanı uçuşlarına kadar, tüm hava faaliyetini, terörist eylemlere karşı bir hedef haline getirebilir. Bu nedenle, Sivil Hava Trafik Radarlarının kendi aralarında hava resmini göndermek için oluşturdukları veri haberleşmesinin, bilgi güvenliği açısından koruma altına alınması gerekmektedir.

Bu tez kapsamında, radarların yer-yer hatlarından Sivil Hava Trafik Kontrol Merkezlerine gönderdikleri hava resimlerinin, bilgi güvenliğinin sağlanması ve bu konuda yeni bir model oluşturulması amacıyla, yazılım tabanlı bir teknik ile şifrelenmesi üzerinde durulmuştur. Kuşkusuz, %100 güvenli olduğu iddia edilen bir sistemi oluşturmak, çoğu zaman zor ve hatta imkansızdır. Ancak

güvenli olduğu düşünülen veya güvenli olduğu kabul edilen bir sistem oluşturmak daha mantıklı bir yaklaşımdır. Bu nedenle, yapılan tez çalışmasında, uygulamaya yönelik güvenli bir sistem tasarlanmaya çalışılmıştır.

1.1. Araştırmanın Problemi

Bu tez kapsamında yapılan araştırmanın problemi, radarların, halen yer-yer hatlarından Sivil Hava Trafik Kontrol Merkezlerine açık bir şekilde gönderdikleri hava resimlerinin, gönderilmeden önce, uluslararası düzeyde emniyetli olduğu kabul edilen teknikler ile şifrelenmesi ve alındıktan sonra da şifrelerinin çözülerek kontrol sistemlerine dahil edilmesidir. Bu bağlamda, gönderilen hava resimlerinin, istenmeyen üçüncü şahıslar tarafından elde edilmesinin engellenmesi ve açık bilgi formatındaki verilerin, uçuş emniyetini tehlikeye atacak değişikliklere karşı korunması amaçlanmıştır.

Şifreleme ve şifre çözme konusunda yapılan çalışmaların, güvenlik ve ticari nedenlerden dolayı yaygın ve açık bir şekilde duyurulmaması ve özellikle bu konuda Türkçe kaynak sıkıntısının bulunması dikkate alındığında, yapılan tez çalışmasının, uygulamaya yönelik çalışmalara, bilgi birikimi açısından, küçük de olsa bir katkı sağlayacağı düşünülmüştür.

Tez çalışmasının uygulama bölümünde, karşılıklı Kimlik Tanıtma ve Gizli Anahtarlı Şifrelemede kullanılacak anahtarının gönderilmesi safhalarında gerekli olan Açık Anahtarlı Şifrelemeye ait özel anahtarın, Diffie-Hellman tekniği kullanılarak “TCP/IP” protokolü ile veya şifre mutemedi ile karşı tarafa ulaştırıldığı varsayılmıştır. Gönderme ve almada kullanıldığı varsayılan program parçasının kaynak kodu 3.2 numaralı paragrafta açıklanmıştır.

Bu araştırmada; uluslararası düzeyde kabul gören ve şimdiye kadar kırıldığına dair herhangi bir bir bilgiye rastlanmamış olan ve bundan dolayı da güvenli olduğu kabul edilen RSA (Rivest-Shamir-Adleman) ve İleri Şifreleme

Standardı - İŞS (Advanced Encryption Standard – AES - Rijndael) yordamları kullanılmıştır.

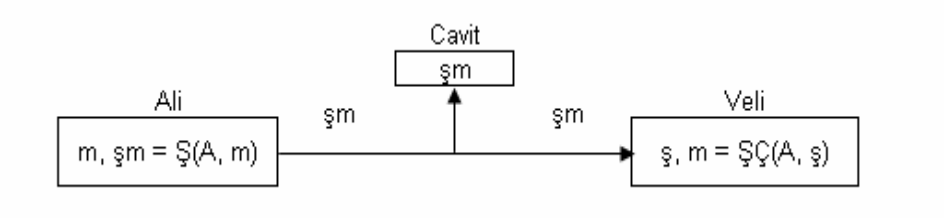
1.2. Şifreleme Hakkında Genel Bilgi

Şifreleme (cryptography), açık metin halinde bulunan herhangi bir bilgiyi, istenmeyen kişiler tarafından anlaşılamayacak ve çözülemeyecek hale getirilmesi ile ilgilenen bilim dalıdır.

Esasen emniyetli haberleşme konusunu inceleyen şifrebilimin (cryptology) bir alt dalı olmakla beraber bilginin gizliliğinin korunması ve/veya gerçeklik, doğruluk ve güvenirliğinin ispat edilmesi amacıyla şifreleme ve şifre çözme yordamları (algoritma) ile ilgilenen bilim dalıdır (Menezes ve ark., 1996, s.4, s.15).

Son yirmi yıldır elde edilen tecrübeler, cebirsel işlemlere dayanan şifreleme yordamlarının neredeyse tamamının kırıldığını göstermektedir. Cebirsel ifadeler, saldırılara çok açık olduğundan, herhangi bir matematikçi tarafından rahatlıkla çözülebilmektedir. Bu yüzden, şifreleme yordamlarında, cebirsel ifadelerden kaçınmak gerekir. Herhangi bir cebirsel eşitlik kullanılması durumunda, bu eşitliklerin etkileri, çırpı (hash) fonksiyonları ile ortadan kaldırılması zorunludur (Ferguson ve Schneier, 2003, s.268).

Şekil 1.1 genel anlamda şifrelemenin nasıl yapıldığını göstermektedir. Ali ile Veli kendi aralarında şifreli bir şekilde haberleşiyorken, Cavit araya girerek gönderilen mesajları almaya çalışmaktadır. Cavit, gönderilen mesajları çözmekle kalmayıp, Ali'nin gönderdiği mesajların yerine, Veli'ye kendi istediği sahte mesajları gönderebilir. Şekildeki m, açık metini; şm, şifreli metini; Ş, şifreleme fonksiyonunu; ŞÇ, şifre çözme fonksiyonunu ve A, şifrelemede kullanılan anahtarı ifade etmektedir. Şifrelerin istenmeyen kişiler tarafından çözülmesi ile ilgili yöntemler, Şifrelemeye Karşı Saldırı bölümünde ayrıca incelenmiştir.



Şekil 1. 1. Genel anlamda şifreleme (Ferguson ve Schneier, 2003, s.22)

1.3. Şifrelemenin Kısa Tarihçesi

Şifreleme (Cryptography), yapı olarak Yunanca iki kelimenin birleşiminden oluşmaktadır ve "gizli yazı" anlamını taşıyan derinlemesine matematiksel bir konudur (Bishop, 2003, s.217).

Şifrelemenin ilk bilinen kullanım yeri, M.Ö. 1900 yıllarındaki mezar kitabeleridir. Haberleşme amacıyla, ilk defa, M.Ö. 475 yılında kullanılmıştır. Haberleşme güvenliği ve şifreleme üzerine ilk yazı, M.Ö. 350 yılında yayımlanmıştır. Askeri manada şifreleme, ilk defa, Sezar tarafından M.Ö. 60 yılında kullanılmıştır. Şifre çözme ile ilgili bilinen en eski bilimsel ilmi eser, 1412 yılında, Mısırlı bilim adamı El-Kalkashandi tarafından yayımlanmıştır. Günümüzdeki şifrelemeye yönelik kaydadeğer gelişmeler, 1970'li yıllardan itibaren ortaya çıkmıştır. IBM, 1971 yılında Sabah Yıldızı (Venüs – Lucifer) olarak bilinen şifreleme tekniğini geliştirmiştir. 2000'li yıllara kadar uzun süre önemini koruyan, Veri Şifreleme Standardı - VŞS (Data Encryption Standard - DES) 1975 yılında geliştirilmiş ve 1977 yılında da onaylandığı duyurulmuştur. Açık Anahtarlı Şifreleme tekniği ise ilk defa 1976 yılında Diffie-Hellman tarafından ortaya atılmıştır. 1978 yılında, Merkle tarafından Sırt Çantası (Knapsack) adıyla bilinen şifreleme tekniği; Rivest – Shamir – Adleman tarafından günümüzde de hala kullanılan ve kendi isimlerinin baş harfleri ile isimlendirilmiş olan RSA yordamı geliştirilmiştir (Stallings, 1995, s.108).

Şifreleme ile ilgili, son yılların en önemli gelişmesi İŞS geliştirme çalışmalarıdır. Bu amaçla, 1997 yılında, ABD Milli Standartlar ve Teknoloji

Enstitüsü - MSTE (National Institute of Standarts and Technology – NIST) tarafından, yeni bir Gizli Anahtarlı Şifreleme standardı için gerekli kriterler belirlenmiş ve seçim amacıyla tüm dünyaya duyurulmuştur (Stallings, 2003a, s.141). Birçok bilim adamı tarafından, aday olarak gönderilen şifreleme yordamları elemeye tabi tutulmuş ve içlerinden 5 tanesi finalist olarak seçilmiştir. 2000 yılında da, bunlardan Rijndael yordamı yeni İŞS olarak kabul edilmiştir (Nechvatal ve ark., 2000, s.13).

Halen birçok kurum ve bilim adamı yeni teknikler üzerinde çalışmaktadır. Özellikle finansal işlemlerin bilgisayar üzerinden yapılmaya başlanması, bilgi güvenliği konusunu ön plana çıkarmış ve şifreleme tekniklerini daha da önemli hale getirmiştir. Günümüzde, el ile atılan imzalar, yerlerini yavaş yavaş elektronik imzalara bırakmakta ve günlük hayatımızın bir çok alanına şifreleme ile ilgili konular girmektedir. Getirdikleri kolaylıklar nedeniyle günlük hayatın vazgeçilmez unsurları haline gelen bilgisayarlar, şifrelemenin de dahil olduğu bilgi güvenliği konularını olmazsa olmazlar arasına sokmaktadır.

1.4. Şifrelemenin Kullanım Alanları

Şifreleme, elektronik donanımın var olduğu birçok sistemde kullanılmaktadır. Başlangıçta askeri amaçlarla kullanılan şifreleme teknikleri, günümüzde bilgi güvenliği gerektiren sistemlerin neredeyse tamamında vazgeçilmez bir unsur olarak yerini almıştır.

Şifrelemenin kullanım alanları her geçen gün artmakla birlikte, başlıca şu alanlarda kullanıldığı göze çarpmaktadır:

- Otomatik para çekme makinalarında
- Şifreli ve kartlı elektronik giriş kapılarında
- Cep telefonlarında

- Internet üzerinden yapılan ve bilgi güvenliği gerektiren işlemlerde
- Askeri uygulamalarda
- Şifreli televizyon yayınlarında
- Yönlendiriciler (router) ile emniyetli haberleşme bağlantıları oluşturmada
- Bilgisayarlarda, şifre çizelgelerinin oluşturulmasında
- Sabit disk ve işletim sistemlerinin korumalı hale getirilmesinde

Bu kullanım alanlarına ek olarak, şifrelemenin bir alt çeşidi olan Açık Anahtarlı Şifreleme yordamlarının hangi amaçlarla kullanıldığı Çizelge 1.1'de gösterilmektedir (Stallings, 2003b, s.71).

Çizelge 1. 1. Açık Anahtarlı Şifreleme yordamlarının kullanım alanları (Stallings III, s71)

Yordam	Şifreleme / Şifre Çözme	Dijital İmza	Anahtar Dağıtımı
RSA	Evet	Evet	Evet
Diffie - Hellman	Hayır	Hayır	Evet
Dijital İmza Standardı	Hayır	Evet	Hayır
Eliptik Eğri	Evet	Evet	Evet

2. MATERYAL VE YÖNTEM

Yapılan tez çalışması, literatür taraması ve uygulama olmak üzere iki safhada hazırlanmıştır. Araştırmayı yönlendirecek esas veriler, konu hakkında yayınlanmış kitaplardan elde edilmiştir. Uygulama bölümünde ise, MSDN kütüphanesinin yardım menüsündeki ilgili açıklamalardan faydalanılmıştır.

Uygulama bölümünde, yazılım tabanlı teknikler üzerinde durulmuş ve donanım tabanlı tekniklere göre daha yavaş olmalarına rağmen, uygulamayı anlamlı ve gerçekleştirilebilir bir hızda çalıştırabildikleri görülmüştür. Uygulama safhasında hazırlanan program, aynı zamanda donanım tabanlı olarak da gerçekleştirilebilir. Ancak donanım tabanlı uygulamalar daha çok Elektronik Mühendisliği ile ilgili olduğundan, uygulama bölümünde bu konuya yer verilmemiştir.

Araştırmanın, metaryel ve yöntem kısmında, ABD MSTE tarafından onaylanmış ve uluslararası düzeyde kabul gören şifreleme teknikleri incelenmiş ve bunların içinden, tez kapsamında kullanılabilecek olanları üzerinde durulmuştur. Bu nedenle, öncelikle Şifreleme ve Şifre Çözme teknikleri araştırılmış ve bu yordamlarda kullanılan temel kavramlar açıklanmıştır.

2.1. Şifreleme Ve Şifre Çözme Teknikleri

Şifreleme ve Şifre Çözme (Encryption and Decryption) tekniklerinde iki farklı yöntem mevcuttur. Bunlardan ilki Gizli Anahtarlı Şifreleme (Symetric Encryption), ikincisi de Açık Anahtarlı Şifreleme (Public Key Encryption) yöntemidir. Her iki yönteminde de farklı bilim adamları tarafından oluşturulmuş çeşitli yordamları vardır.

Gizli Anahtarlı Şifreleme, bazı kaynaklarda Simetrik Şifreleme olarak; Açık Anahtarlı Şifreleme de Asimetrik Şifreleme olarak geçmektedir (Ferguson ve Schneier, 2003, s.28) ancak, bu tez kapsamında karışıklığa sebep olmamak için birincil isimleri kullanılmıştır.

Gizli Anahtarlı Şifreleme ile Açık Anahtarlı Şifrelemeyi birbirlerinden ayıran en önemli nokta kullandıkları anahtar tipleridir. Gizli Anahtarlı Şifrelemede gizli anahtar (secret key) olarak adlandırılan tek bir anahtar kullanılırken, Açık Anahtarlı Şifrelemede açık anahtar (public key) ve özel anahtar (private key) olarak adlandırılan iki ayrı anahtar kullanılmaktadır. Gizli Anahtarlı Şifrelemede kullanılan gizli anahtar ile Açık Anahtarlı Şifrelemede kullanılan özel anahtarın her ikisi de gizli olmasına karşın, karıştırılmaması için ayrı ayrı isimlerle temsil edilmektedir (Stallings, 1995, s.111).

Açık Anahtarlı Şifreleme, Gizli Anahtarlı Şifrelemeye göre daha yavaştır. Bu yüzden, mesaj yoğunluğu fazla olan haberleşmede ve açık metinlerin şifrelenmesi işlemlerinde daha çok Gizli Anahtarlı Şifreleme, anahtar alış-verişi ve küçük veri gruplarının şifrelenmesinde de Açık Anahtarlı Şifreleme kullanılmaktadır (Schneier, 1996, s.461).

Bu tez kapsamında, Gizli Anahtarlı Şifreleme ve Açık Anahtarlı Şifrelemeye geçilmeden önce, şifreleme açısından bilinmesi gereken temel konular olan Rassal Sayı ile Asal Sayı Üretimlerinden bahsedilmiştir.

2.2. Rassal Sayı Üretimi

Rassal sayılar (random numbers), herhangi bir kurala bağlı olmadan dizilen, her birinin dizilişlerindeki yerleri kendinden öncekilerden bağımsız olarak belirlenen sayılardır.

Sadece bilgisayar işlemcilerini kullanarak tam anlamıyla rassal bir sayı üretmek mümkün olmamakla beraber, kabul edilebilir seviyede rastgele olma

özelliği taşıyan sayılar üretmek mümkündür. Bu tipte, gerçek rassal sayılara çok yakın özellik taşıyan sayılara sahte rassal sayılar (rassalımsı, rastgelemsi, pseudorandom) denmektedir.

Herhangi bir sayı dizisinin (belli sınırlar içersinde bulunan sayılardan elde edilmiş, içlerinden bir şekilde seçilmiş veya üretilmiş sayı grubuna ait olan sayıların), rassal olduğundan bahsedilebilmesi için, aşağıdaki özelliklere sahip olması gerekir:

- **Tektip Dağılım (Uniform Distribution):** Seçilen sayı dizisinin içinde bulunan her sayının tekrarlanma adedinin yaklaşık olarak aynı olması (Stallings, 1995, s.97).

- **Bağımsızlık (Independence):** Sayı dizisinin içindeki sayılardan diğer sayıları elde edebilecek herhangi bir kural veya yordamın bulunmaması (Stallings, 1995, s.97).

- **Tekrar Üretilememe (No Reliable Reproduction):** Aynı girdileri ve yöntemi kullanarak aynı sayının elde edilememesi (Schneier, 1996, s.46).

- **Tahmin Edilemezlik (Unpredictability):** Bir sonraki sayının hangi sayı olacağının tahmin edilememesi (Schneier, 1996, s.45).

Rassal sayı üretimi için çeşitli teknikler ve yordamlar mevcuttur. Radyasyon yayılımı, gaz boşaltma tüplerinin kaçakları ve sızıntılı meksefelerden (kondansatör) elde edilen veriler (Stallings, 1995, s.98) rassal sayı üretiminde kullanılabilir. Bu değişkenler, ortamdaki diğer değişkenlerden etkilendikleri için sürekli farklı değerler almaktadırlar. Algılayıcılar vasıtasıyla elde edilen değişkenlerin, sabit değerlerden oluşan en büyük basamakları kesildikten sonra kalan en hassas kısımları rassal sayı olarak kullanılabilmektedir.

Bunların haricinde, klavye tuşlarına dokunma ve farenin hareketlerinden elde edilen veriler de çok bilinen ve rassal sayı üretiminde kullanılan örneklerdir. Bilgisayar diskine erişim zamanındaki salınımlardan rassal sayı elde etme

çalışmaları da bu konuda devam eden araştırmalara örnek olarak gösterilebilir. Ancak, bu değişkenler, rassal sayı ile ilgili saldırı da bulunan kişiler tarafından da kontrol edilebileceğinden ve etkilenebileceğinden, kısmen şüpheli veriler olarak değerlendirilmektedir (Ferguson ve Schneier, 2003, s.156).

Rassal sayı üretimi, şifreleme işlemlerinde önemli bir yer tutmaktadır. Özellikle şifrelemede kullanılacak anahtarların elde edilmesinde rassal sayılara ihtiyaç duyulmaktadır. Rassal sayılar, üretilen anahtarların, başkalarınca pratik bir şekilde tahmin edilememesinde etkin bir rol oynamaktadır.

Rassal sayıların kullanım alanları, aşağıdaki gibi sıralanabilir:

- Oyunlarda (Schneier, 1996, s.44)
- Frekans tabanlı çalışan cihazların (radar, telsiz vb.) elektronik karıştırmaya karşı korunmasında
- Parola (password) üretiminde (Ferguson ve Schneier, 2003, s.349)
- Tıbbi bilimsel araştırmalarda (Uysal ve ark., 2005) olduğu gibi, tüm araştırma kütlesini temsil edecek ve anlamlı bir sonuç elde edilmesine katkıda bulunacak istatistiksel örneklemlerin, rassal bir şekilde elde edilmesinde (Kara, 2000, s.266)
- Asal sayıların elde edilmesi (Welschenbach, 2001, s.229).
- Şifreleme işlemleri (Shparlinski, 1999, s.131) için anahtar üretimi ve güvenlik protokollerinin çalıştırılması (Welschenbach, 2001, s.229).

Şifreleme açısından, rassal sayı üretimi oldukça önemlidir. RSA, VŞS gibi birçok şifreleme yordamında anahtarların güvenliği rassal sayı üretimine bağlıdır. Rassal sayı üretiminde oluşacak bir hata veya risk, yapılan şifrelemenin güvenliğini tamamen ortadan kaldırmaktadır (Ferguson ve Schneier, 2003, s.159). Bu yüzden, tezin başlangıç kısmında rassal sayı üretiminden bahsedilmiştir.

128 ikillik (bitlik) bir rassal sayının tüm bitleri sadece 1 veya sadece 0 değerinden oluşuyorsa, sayı, rassal şekilde üretilmiş olsa bile, şifreleme güvenliği açısından emniyetli değildir. Tam Güç Saldırısı (Brute Force Attack) yapacak biri için, ilk denenecek değerler, sıfırdan başlayıp en büyük değere doğru artan veya en büyük değerden sıfıra doğru azalan değerlerdir. Bu durumda daha ilk işlemde elde edilen rassal sayı, saldıran kişi tarafından denenmiş olur, hemen hemen hiç zaman kaybedilmez, saldırı başarıyla sonuçlanır ve şifrelemede kullanılan anahtar tespit edilir.

Şifreleme ile ilgili programlarda, gerçek rassal sayı elde etmek oldukça güç olduğundan, sahte rassal sayı olarak adlandırılan "pseudorandom" sayılar kullanılmaktadır. Rassal sayı üretiminde, her defasında farklı bir değer elde etmek için yeniden tohumlama (reseed) yapılması gerekmektedir. Bu işlem sırasında kullanılan tohum (seed), saldıran kişiler tarafından tesbit edilmemesi için, çırpı (hash) fonksiyonları ile karıştırılır. Saldıran kişi bu karıştırmanın sonucuna ulaşsa bile, ters işlemi yapıp tohumu elde edemez (Ferguson ve Schneier, 2003, s.165). Böylelikle rassal sayı üretiminde kullanılan tohumun değeri emniyet altına alınmış olur.

Rassal sayı üretimi ile ilgili olarak takip edilebilecek yöntemlerden bir tanesi aşağıda verilmiştir.

T: Tohum, A: Anahtar (başlangıç değeri = 0), S: Sayaç (başlangıç değeri = 0), "||" işareti birleştirme olmak üzere;

(2.1)

$$A \leftarrow GÇY_{256}(A \,||\, T)$$

$$S \leftarrow S + 1$$ (Ferguson ve Schneier, 2003, s.165)

Tohumlanmış anahtar değeri elde edilmiş olur. $GÇY_{256}$ ifadesi, Güvenli Çırpı Yordamı (Secure Hashing Algorithm - SHA) olarak bilinen ve 256 ikillik çıktısı bulunan bir tür karıştırma yordamıdır.

Bundan sonraki aşamada, aşağıda belirtildiği gibi, rassal sayı için gerekli olan bloklar oluşturulur.

R: Rassal dizgi (string) (başlangıç değeri = 0x00), Ş: Şifreleme fonksiyonu olmak üzere;

(2.2)

assert $S \neq 0$

for i = 1 ,…., k **do**

{R ← R || Ş (A, S)

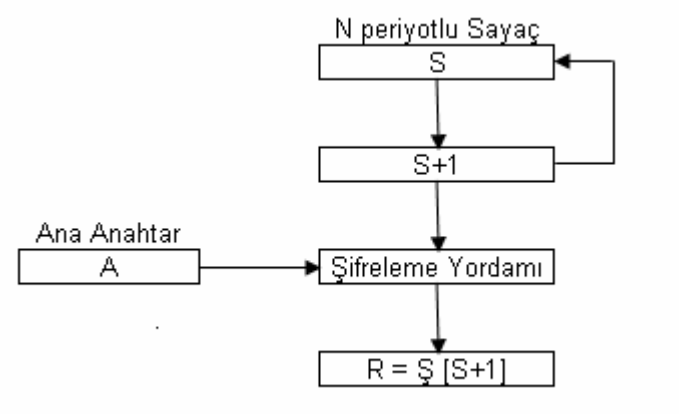
S ← S + 1}

return R; (Ferguson ve Schneier, 2003, s.166)

Bu işlemlerin sonucunda R dizgisi elde edilmiş olur. Bu dizginin ilk **n** byte kesilerek alınır ve uygun fonksiyonlar kullanılarak sayıya dönüştürülür. En sonunda elde edilen değer şifreleme açısından güvenilir bir **rassal sayı**dır. Yeniden bir değer elde edilmek istendiğinde kullanılan tohum yenilenir ve (2.1) numaralı basamaktaki sayaç değerleri değiştirilmeden devam edilir.

Bu bilgilere ek olarak, rassal sayıların anahtar olarak kullanılabilmesi için **n** byte değerinin yeterince geniş olması gerekmektedir. Bu konu tezin ilerleyen bölümlerinde ayrıca açıklanacaktır.

Rassal sayı üretiminde kullanılan (2.1) ve (2.2) numaralı basamaklar Şekil 2.1'de kabaca ifade edilmiştir.



Şekil 2. 1. Bir sayaçtan sahte rassal sayı üretimi (Stallings, 1995, s.101)

2.3. Asal Sayı Üretimi

Asal sayılar (Prime numbers), 1 ve kendinden başka herhangi bir sayıya bölünemeyen sayılardır. RSA ve Diffie-Hellman gibi bir çok şifreleme yordamında asal sayılar kullanılmaktadır. Adı geçen yordamlar da dahil olmak üzere, birçok şifreleme yordamı, bir veya birden fazla çok büyük asal sayıya ihtiyaç duymaktadır (Stallings, 2003a, s.243). Bu yüzden asal sayı üretiminden de tezin başlangıç aşamasında bahsedilmiştir.

Asal sayı elde edilmesinde kullanılan yöntemlerin çoğu, aşağıdaki basit iki basamaktan oluşan asal sayı üretme tekniğine (Tilborg, 2000, s.182) dayanmaktadır.

(1) U ikil uzunluğunda, tek (çift olmayan), herhangi bir R rassal bir sayısı elde edilir.

(2) R sayısının, asal olup olmadığı test edilir, değilse ilk basamak yeniden uygulanır.

Bu yöntem ifade edilmesi basit ancak gerçekleştirilmesi oldukça karmaşık işlemler içermektedir.

Bundan dolayı, öncelikle asal sayıların matematiksel ifadelerinin incelenmesi gerekmektedir.

$\prod(x)$: x ve kendisinden küçük asal sayıların çarpımı olmak üzere,

$$\lim_{x\to\infty} = \frac{\prod(x)}{x / \ln x} \qquad \textbf{(2.3)}$$

olarak tanımlanmaktadır (Tilborg, 2000, s.183).

Yukarıdaki ifadenin biraz değiştirilerek sözel halde açılması sonucunda, bizi aynı sonuca daha anlaşılır bir şekilde ulaştıran, aşağıdaki bakış açısı elde edilmiştir.

Bilinen bütün (sonsuz sayıdaki) asal sayı birbirleriyle çarpılsın ve sonucuna 1 eklensin, elde edilen bu yeni sayı, bilinen asal sayılardan hiçbirine kalansız bölünemez. Bölünmesi durumunda kalan daima 1'dir. Diğer bir deyişle bulduğunuz bu yeni sayı da asaldır (Kaufman ve ark., 2002, s.186). Bu yaklaşımda önemli olan nokta, seçilen asal sayıya kadar olan asal sayıların - kendisi de dahil olmak üzere - birbirleriyle çarpılması ve sonucuna 1 sayısının eklenmesidir. Aksi takdirde seçilen asal sayıya kadar olan asal sayıların hepsi çarpılmaz, rassal bir şekilde içlerinden seçilenler birbirleriyle çarpılıp, sonuca 1 eklenirse sonuç her zaman bir asal sayı olmayabilir. 11 sayısı asal sayı sınırı olarak kabul edilsin. 11'e kadar olan asal sayılardan 5 ve 7 asal sayıları seçilmiş olsun. Bunların çarpımları olan 35 sayısına 1 eklendiğinde; sonucun asal olmadığı kolaylıkla görülür. O yüzden, Kaufman'ın bahsettiği yöntemde seçilen sınıra kadar olan asal sayıların ***tamamının*** birbirleriyle çarpılması ve sonuca 1 eklenmesi, elde edilecek sayının asal olmasında en önemli etkendir. Aşağıdaki örnek bu yöntemin doğruluğunu ispat etmektedir.

Örnek:

x: Asal sayı olmak üzere

$$S = \prod(x) + 1 ;$$

2,3,5,7,11,13,17,19,23,29 asal sayılarının çarpımı + 1 ;

6469693230 + 1 = 6469693231'dir. Bu sayının 2-29 arasındaki herhangi bir asal sayıya bölümünde kalan daima 1 olmaktadır. 29 ile 6469693231 arasında kalan diğer asal sayılara da kalansız bölünememektedir. 29 ile 6469693231 arasında kalan diğer asal sayılarla yapılan bölme işleminde kalan 1'den farklı bir rakam olmaktadır. Sonuç olarak, 6469693231 sayısı da, kendinden küçük herhangi bir asal sayıya kalansız bölünememektedir, dolayısıyla asal olduğu ispatlanmaktadır. Buna Öklid'in asal sayı ispatı denmektedir (Weisstein, 2006).

Belirli sınırlarda (0 ile 999999 gibi), küçük asal sayıların, asal olup olmadıklarını test etmenin en basit yollarından bir tanesi de, 2'den başlayıp bütün sayıların katlarının (2,4,6,8 ….; 3,9,15…..gibi) asal sayılar kümesinden çıkartılması ve istenilen sınıra ulaşıldığında kalan sayıların asal sayılar olarak kayda geçirilmesidir. Bu yöntemle asal sayı bulma işlemine "Eratosfen yöntemi" denmektedir (Alfeld, 1998). Ancak çok zaman alıcıdır ve şifreleme açısından tercih edilen bir yöntem değildir. Çünkü, 1000-9999 ikillik herhangi bir sayının asal olup olmadığını incelemek istediğimizde bu yöntem gereksiz birçok sayının daha asal olup olmadığıyla ilgilenmekte ve en sonunda bize istediğimiz sayının asal olup olmadığı sonucunu vermektedir. Birçok şifreleme yordamının, bir asal sayı elde etmek için bu kadar çok Merkezi İşlem Birimi - MİB (central processing unit - CPU) zamanı harcaması imkansızdır, o yüzden bu yöntem pratik olmaktan çıkmaktadır.

2.3.1. Fermat Asallık Testi

Fermat, başta matematik olmak üzere ve birçok alanda araştırmalar yapmış bir bilim adamıdır. Yaptığı araştırmalardan özellikle bir tanesi, yaşadığı dönemden (1601-1665) (Cosgrave, 2004) yaklaşık 300 yıl sonra şifreleme

biliminde kullanılmaya başlanmıştır. Bu araştırma kendisinin adıyla anılmakta olup, **Fermat'ın küçük teoremi** olarak (Fermat'ın son teoremi veya Fermat teoremi olarak bilinen başka bir teoremi daha vardır) adıyla kayıtlara geçmiştir:

p bir asal sayı ve a da bir tamsayı olmak üzere;

$a^p \equiv a \pmod p$ teoremidir (Tilborg, 2000, s.357).

Fermat'ın bu teoremi $F_n = 2^{2^{\wedge}n}$ olarak bilinen Fermat sayılarını ortaya çıkarmıştır. Ancak daha sonradan, bu sayıların tamamının asal olmadıkları ispatlanmıştır. Fermat'ın küçük teoremi, kendisinden sonra Euler için bir kaynak olmuş ve şifrelemede kullanılan Euler Φ(n) olarak gösterilen Euler Totient fonksiyonunu ortaya çıkarmıştır (O'connor ve Robertson, 2005).

Femat'ın küçük teoremi ve Euler Totient fonksiyonu RSA şifreleme yordamı bölümünde detaylı olarak açıklandığından, bu bölümde genel bilgi olarak verilmiştir.

2.3.2. Rabin-Miller Asallık Testi

Herhangi bir sayının asal olup olmadığını test etmek için kullanılan diğer bir yöntem de Rabin-Miller testidir. Kolay olduğundan dolayı yaygın olarak kullanılmaktadır (Schneier, 1996, s.259).

Rabin-Miler testinin hata oranı da oldukça düşüktür. Testin kaç defa çalıştırılacağını k sayısı ile ifade edildiğinde; Solovay-Strassen assallık testi 2^{-k} hata düzeyinde asal olmayan bir sayıyı asalmış gibi göstermekte, buna karşın Rabin-Miller asallık testi 4^{-k} hata düzeyinde asal olmayan bir sayıyı asalmış gibi göstermektedir (Welschenbach, 2001, s.221). Bu sebeplerden dolayı Rabin-Miller testine bu bölümde ayrıca yer verilmiştir.

r: herhangi bir rassal sayı,

b: (r – 1) değerini kalansız bölen 2^b değerindeki üs sayısı,

m: $(r - 1) / 2^b$ olmak üzere

(1) r sayısından küçük ikinci bir rassal a sayısı alınır

(2) $j = 0$ ve $z = a^m \bmod r$ işlemleri yapılır

(3) $z = 1$ veya $z = r - 1$ ise; r testi geçer ve asaldır

(4) $j > 0$ ve $z = 1$ ise; r asal değildir

(5) $j = j + 1$ işlemi yapılır. $j < b$ ve $z \neq r - 1$ ise; $z = z^2 \bmod r$ işlemi yapılır ve (4) numaralı basamağa geri gidilir. $z = r - 1$ ise; r testi geçer ve asaldır.

(6) $j = b$ ve $z \neq r - 1$ ise; r asal değildir (Schneier, 1996, s.260).

Rabin-Miller testinin haricinde, bir sayının asal olup olmadığını, çarpanlarına ayırarak da anlayabiliriz. Ancak bu yöntem oldukça zaman alıcı ve çoğu büyük sayı için imkansızdır. Zaten şifreleme yöntemlerinin esas dayanaklarından birisi de bu zorluk ve imkansızlıktır. RSA şifreleme yöntemi, *p* ve *q* asal sayı olmak üzere, $n = pq$ değerinin mod işlemine tabi tutulmasına dayanmaktadır ve *p* ve *q* asallarının dolayısıyla *n* sayısının yeterince büyük olmasını gerektirmektedir ki, *n* sayısı çarpanları olan *p* ve *q* sayılarına ayrılamasın (Menezes ve ark., 1996, s.149). Zaten 1000-9999 ikillik bir sayının çarpanlarına ayırmak, günümüz teknoloji ile mümkün olsa idi, mevcut birçok şifreleme yordamının hiçbir güvenliği kalmazdı.

2.3.3. Mersenne Sayıları ve Lucas-Lehmer Asallık Testi

Mersenne sayıları, $s \geq 2$ olmak üzere; $m = 2^s - 1$ formatında olan sayılardır. Lucas-Lehmer asallık testi, çok yoğun işlem ve buna bağlı olarak MİB zamanı gerektiren bir yöntem olduğu için, Rabin-Miller testinden önce kullanılmamalıdır (Menezes ve ark., 1996, s.142).

Mersenne sayıları, özel bir sayı grubunu oluşturduğundan, bu sayılarla yapılan işlemler tüm sayılar için genelleştirilemez. Mersenne sayılarından herhangi biri için yapılan test sadece o sayının asal olup olmadığını gösterir. Diğer bir ifade ile; $s \geq 2$ olmak üzere; $m = 2^s - 1$ olarak ifade edilen m sayısı için yapılan test sonucunda 3 ile m arasındaki bütün asal sayılar elde edilemez. Örneğin 13 asal sayı olmasına rağmen Mersenne sayı formatına uymamaktadır. Bundan dolayı asal sayı elde etmek için yapılan işlemlerde, tüm asal sayıların, Mersenne sayı formatına uymadığının bilinmesi ve Mersenne sayı grubuna girmeyen asal sayıların tespiti için diğer yöntemlerin uygulanması gerekmektedir.

Mersenne sayılarının asallığını test etmekte kullanılan Lucas-Lehmer asallık testi aşağıdaki basamaklardan oluşmaktadır.

$n = 2^s - 1$ ve $s \geq 3$ olmak üzere;

(1) s sayısının 2 ile $\lfloor \sqrt{s} \rfloor$ arasında herhangi bir çarpanı var mıdır? Çarpanı yoksa asaldır.

(2) $u = 4$

(3) k 1'den (s-2)'ye kadar ; $u \leftarrow (u^2-u) \bmod n$ işlemi yapılır

(4) En sonunda u = 0 ise asaldır, değilse birleşiktir (Menezes ve ark., 1996, s.142).

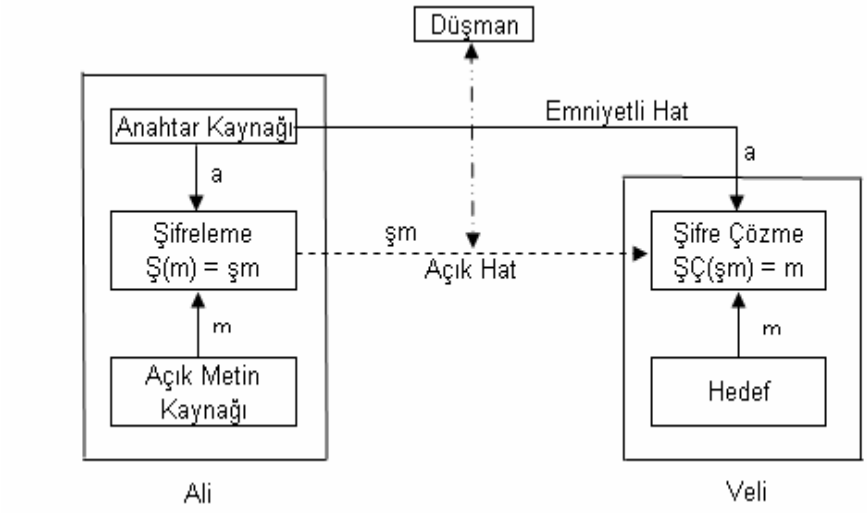
2.3.4. Diğer Testler Ve Asal Sayı Üretme Yöntemleri

Yukarıda bahsedilen Rabin-Miller ve Mersenne sayılarına uygulanan Lucas-Lehmer asallık testinin dışında Solovay-Strassen, Lehmann, Cohen-Lenstra, çarpanlarına ayırma, eliptik eğriler ile asallık testleri ve Gordon ve Maurer asal sayı üretim yordamları bulunmaktadır (Menezes ve ark., 1996, s.137-154). Bunlardan, Gordon ve Maurer yordamları uygulama olarak Rabin-Miller yöntemine kısmen bağımlı olup, daha verimli ve anlamlı olmaları için öncelikle Rabin-Miller yönteminin uygulanmasına ihtiyaç duymaktadır.

Bu yüzden adı geçen asallık testleri ve asal sayı üretim yordamları daha fazla işlem gerektirmeleri ve/veya daha az kesinlik içermelerinden dolayı, yapılan tez çalışması kapsamında ayrıca ele alınmamıştır.

2.4. Gizli Anahtarlı Şifreleme (Symmetric Encryption)

Gizli Anahtarlı Şifreleme, alıcı ile göndericinin aynı anahtarı kullanmalarına dayanan, diğer bir deyişle her iki uçtan bakıldığında simetrik görünen bir tekniktir. Gizli Anahtarlı Şifreleme; tek-anahtarlı şifreleme (single-key) veya geleneksel (conventional) şifreleme olarak da adlandırılmaktadır (Stallings, 1995, s.24). Gizli Anahtarlı Şifreleme ile ilgili şema Şekil 2.2.'de gösterilmektedir.



Şekil 2. 2. Gizli Anahtarlı Şifrelemeyi gösteren şema (Menezes ve ark., 1996, s.16)

Gizli Anahtarlı Şifrelemenin mantığı, çok genel manada aşağıdaki basamakları takip etmektedir (Schneier, 1996, s.28):

(1) Ali ile Veli herhangi bir şifreleme sistemi üzerinde anlaşırlar.

(2) Ali ile Veli kullanacakları anahtar üzerinde anlaşırlar.

(3) Ali ilgili mesajı alır ve aldığı mesajı anlaştıkları şifreleme sistemi ve anahtarıyla şifreler. Bunun sonucunda şifrelenmiş mesaj ortaya çıkar.

(4) Ali şifrelenmiş mesajı Veli'ye gönderir.

(5) Veli aldığı mesajı Ali'nin kullandığı aynı anahtar ve şifreleme sistemi ile çözer.

Yukarıdaki basamaklar basit gibi gözükse de, gerçekleştirilmesi oldukça karmaşık ve zordur. Zorlukların içersinde özellikle gizli anahtar üzerinde anlaşılması ve bu anahtarın karşı tarafa iletilmesi önemli bir yer tutmaktadır. Bu probleme Anahtar Dağıtım Sorunu (Key Distribution Problem) denmekte olup (Menezes ve ark., 1996, s.16), gerçekleştirilmesi için Diffie-Hellman gibi yordamlar kullanılmaktadır. Gizli Anahtarlı Şifrelemenin önemli bir problemi olan Anahtar Dağıtım Sorunu geniş bir konu olduğundan bu tez kapsamında detaylı bir şekilde incelenmemiş ancak, konunun çözümünde kullanılan yöntemlerden kısmen bahsedilmiştir.

Gizli Anahtarlı Şifreleme yöntemlerinin çeşitli avantaj ve dezavantajları bulunmaktadır. Özel donanım kullanıldığında 100 MB/sn, özel yazılım kullanıldığında ise 1 MB/sn seviyesinde şifreleme yapabilmektedir. Gizli Anahtarlı Şifreleme yöntemleri, şifrelemenin haricinde, çırpı (hash) fonksiyonlarının oluşturulmasında da kullanılmaktadır. Ancak, Gizli Anahtarlı Şifrelemede, kullanılan anahtarın her iki uçta da gizliliğini koruyacak şekilde saklanması gerekmektedir. Çok büyük ağ yapılarında, her kanal için ayrı bir anahtar kullanılma zorunluluğu olduğundan, çok sayıda anahtar üretilmesi ve bunların ilgili birimlere dağıtılması sorunuyla karşılaşılmaktadır (Menezes ve ark., 1996, s31). Gizli Anahtarlı Şifrelemenin bu zorluğu, çok kanallı ağ yapılarında, Açık Anahtarlı Şifreleme tekniklerinin kullanımını öne çıkarmakta ve çoğu durumda zorunlu hale getirmektedir.

Bu tez kapsamında açıklanan Gizli Anahtarlı Şifreleme ve Açık Anahtarlı Şifreleme yöntemlerinin isimleri, Türkçe'ye çevrilerek verilmiş ancak, yaratıcılarının kendi isimlerini taşıyan yordamlar, orjinal isimlerinde olduğu gibi kullanılmıştır.

Bu bağlamda, literatür araştırması yapılmış ve Gizli Anahtarlı Şifreleme yöntemi olarak geliştirilmiş birçok yöntem olduğu tesbit edilmiştir. Ancak Gizli Anahtarlı Şifreleme yordamı olduğu iddia edilen yöntemlerin hepsi genel anlamda kabul gören teknikler olmadığı belirtilmektedir. Bu tez kapsamında, Veri VŞS hariç olmak üzere, VŞS'nın yerini alması için, kriterleri ABD MSTE tarafından belirlenen İŞS gizli anahtarlı şifreleme finalistleri üzerinde durulmuştur. Diğer bir deyişle, güvenirliği tam olarak test edilmemiş, bir çok açıdan tartışmaya açık, içinde güvenlik açığı içermesi muhtemel ve kişisel hataya daha çok ihtimal bırakan şifreleme teknikleri incelemeye alınmamıştır.

Bu tezde incelenen Gizli Anahtarlı Şifreleme teknikleri, ABD MSTE tarafından geliştirilmesi istenen yeni İŞS tekniği için, aday olarak gösterilmiş olan teknikler içersinden seçilmiştir. İŞS için toplamda 15 geçerli olabilecek başvuru yapılmış, ancak bunlardan sadece 5 tanesi İŞS finalisti olarak kalabilmiştir. ABD MSTE'nün oluşturduğu kurul tarafından finalist olarak seçilen Yılan (Serpent), İki-Balık (Twofish), RC6, MARS ve Rijndael yordamlarından, Rijndael yordamı yeni İŞS olarak seçilmiştir (Ferguson ve Schneier, 2003, s.55). Bundan dolayı yeni kaynaklarda İŞS ifadesiyle karşılaşıldığında, bahsedilmek istenen yöntemin Rijndael yordamı olduğu anlaşılmalıdır. Kalan 4 yöntem ise daha çok İŞS adayı veya finalisti olarak adlandırılmakta ve herhangi bir saldırı ile kırılabildiği ispatlanamadığı için şifreleme yöntemi olarak kullanılmasında güvenlik açısından bir sakınca görülmemektedir (Nechvatal ve ark., 2000, s.8, s.17).

İŞS finalisti olarak adlandırılan beş yöntemin dördünde, AxB ikil büyüklüğüne sahip, sıralama kutusu (S-Kutusu, S-Box) olarak tanımlanan kutular kullanılmış ve bunlar A adet giriş ikilini B adet çıkış ikiline dönüştüren, yeniden sıralandırma (permütasyon) kutuları olarak ifade edilmiştir (Nechvatal ve ark., 2000, s.8).

Yukarıda belirtilen hususlardan dolayı, bu tez kapsamında hazırlanan programda gizli anahtarlı şifreleme tekniği olarak **İŞS (Rijndael)** yordamı kullanılmış ve ilgili bölümde İŞS (Rijndael) yordamı daha detaylı bir şekilde incelenmiştir.

2.4.1. Veri Şifreleme Standardı

VŞS (Data Encryption Standard - DES) yordamı Profesör Edward Schaefer tarafından geliştirilmiş olan, daha sonra IBM'in yönettiği Tuchman-Meyer projesi kapsamında 1977 yılında, ABD MSTE tarafından istenen standartlara ulaştırılmış eski bir şifreleme tekniğidir (Stallings, 2003a, s.56, s.73). Bu tez kapsamında, İŞS finalisti olmamasına karşın, ilk Gizli Anahtarlı Şifreleme yöntemi olduğundan ayrıca incelenmiş, ancak artık güvenli bir şekilde kullanılamadığından kısaca açıklanması tercih edilmiştir.

VŞS yordamının oluşturulmasında ABD Milli Güvenlik Ajanslığının görünmeyen bir desteği olduğu ve bundan dolayı gerçekte kendileri için 128 ikillik olarak tasarlanan anahtarın 56 ikillik anahtara düşürüldüğü belirtilmektedir (Schneier, 1996, s.266-267). Bu durum, şifreleme yordamlarının matematiksel yapısının kırılamayacak kadar güçlü olduğu iddia edilse bile, kullanacak birimlerin kendilerine göre ayrı önlemler almaları gerektiği fikrini uyandırmaktadır.

VŞS ilk ortaya çıktığı zamanlarda çok popüler olmasına karşın, 1993 yılında milyon dolar seviyesinden daha az bir maliyetle Michael Wiener tarafından oluşturulan, VŞS için özel, tam güç saldırısı (brute-force attack) ile önemini yitirmiştir (Schneier, 1996, s.153). Dolayısıyla, anahtar büyüklüğü artırılsa veya üçlü VŞS yöntemi oluşturulsa bile gerçek şifreleme işlemlerinde kullanılmasının uygun olmadığı düşünülmektedir. Bu yüzden, tez içersinde detaylı bir şekilde incelenmemiş ve hazırlanan programda kullanılmamıştır.

2.4.2. Yılan (Serpent)

Yılan, Ross Anderson, Eli Biham ve Lars Knudsen tarafından İŞS finalisti olması için geliştirilmiş olan ve VŞS'ındaki ne benzer Sıralama Kutusu kullanan, ancak 128, 192 ve 256 ikillik olmak üzere üç değişik uzunlukta anahtar seçeneği bulunan yeni ve güvenilir bir gizli anahtarlı şifreleme yöntemidir. Bu yöntem, ABD MSTE tarafından belirlenen, üçlü VŞS'dan daha güvenilir ve daha hızlı, 128 ikillik blok uzunluğuna sahip, 256 ikillik anahtar ile çalışan (ancak 128 ve 192 ikillik anahtarları da destekleyebilen) kriterler kapsamında geliştirilmiştir (Anderson ve ark., 1997).

Yılan yöntemi, tamamıyla güvenliği ön plana çıkartan ve bu konuda daha tutucu olan gizli anahtarlı bir şifreleme yöntemidir. Yılan ile ilgili, bilinen şifre kırma saldırıları, toplam 32 basamağın ancak 10uncu basamağına kadar ulaşabilmiştir. Ancak bu kadar güvenli olmasına karşın, İŞS şifreleme yönteminin sadece üçte biri hızındadır ve bundan dolayı tercih edilmemektedir (Ferguson ve Schneier, 2003, s.58). Buna karşın, İŞS şifreleme yöntemi seçimlerinde, bütün ciddi şifreciler, İŞS adayları içersinde en emniyetli yöntemin Yılan olduğunda mutabık kalmışlardır (Ferguson ve Schneier, 2003, s.64).

Yılan sahip olduğu 128 ikillik blok uzunluğundan dolayı, şifreleyeceği açık metinleri 128 ikillik parçalar halinde işleme tabi tutmaktadır. Bu yöntemde şifreleme (Anderson ve ark., 1997);

(1) Başlangıç sıralaması (permütasyonu),

(2) Anahtar karıştırma (32 basamaktan sonuncu basamak hariç) ve Sıralama Kutularından (S-Box) geçirme, sonuncu basamakta ise önceki basamaklarda doğrusal dönüşüm ile elde edilmiş olan anahtarın başka bir karıştırma ile farklılaştırılması ve

(3) Final sıralamasından oluşmaktadır.

Yılan, bu tez kapsamında hazırlanan programda kullanılmadığı için, daha detaylı bir şekilde açıklanmamıştır.

2.4.3. İki-Balık (Twofish)

İki-Balık yordamı, Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall ve Niels Ferguson tarafından ortaklaşa geliştirilmiş, herhangi bir patent ve telif hakkı kısıtlaması bulunmayan, kişisel ve ticari kullanıma açık, İŞS finalisti yeni bir şifreleme tekniğidir (Turgeon, 2003).

İki-Balık yordamı, MARS ve RC6 ile beraber, beş İŞS finalisti içersinde, Feistel ağını (Feistel network) kullanan üç yordamdan biridir (Nechvatal ve ark., 2000, s.23-24). Feistel ağı, bu üç yeni şifreleme yordamının haricinde, VŞS ve Uçan-Balık (Blowfish) gibi eski yordamlarda da kullanılmış olan (Schneier, 1996, s.347) blok halinde şifreleme tekniğidir. Blok halinde şifreleme (block cipher), açık metini belirli uzunlukta bir bütün olarak alıp, aynı uzunluktaki şifrelenmiş metin haline dönüştürme tekniğidir. Blok halinde şifrelemenin haricinde bir de akım şeklinde (stream cipher) şifreleme tekniği mevcut olup, bu yöntemde, açık metin ikil veya byte şeklinde bölünerek alınmakta ve şifrelenmiş metine dönüştürülmektedir (Stallings, 2003a, s.63).

Feistel ağı, eski yordamlarla beraber, yeni üç İŞS finalistinde de kullanılmıştır. Bunlar İki-Balık, MARS ve RC6 yordamlarıdır. 1970'lerin başlarında geliştirilen bu teknik, açık metini iki eşit parçaya bölüp, bu parçaları ilgili anahtar ile "ayrıcalıklı veya" (XOR) işlemine tabi tutma mantığına dayanmaktadır. Açık metin uzunluğu çift bir sayı olan n sayısı olarak ifade edildiğinde, L ve R parçaları n/2 büyüklüğündeki iki adet metinden oluşmaktadır (Schneier, 1996, s.347).

Feistel ağının, şifreleme sırasında nasıl çalıştığı Şekil 2.3'te gösterilmektedir (Stallings, 2003a, s.68).

Bu teknikte, şifreleme sırasında kullanılan eşitliklerin matematiksel ifadesi, L sol yarım, R sağ yarım değeri ve A_i de ilgili turun alt anahtarı olmak üzere;

$L_i = R_{i-1}$

$R_i = L_{i-1} \oplus f(R_{i-1}, A_i)$ şeklinde (Schneier, 1996, s.347); şifre çözme sırasında kullanılan eşitliklerin matematiksel ifadesi de Ş, şifreleme; ŞÇ, şifre çözme olmak üzere;

$LŞÇ_1 = RŞÇ_0 = LŞ_{16} = RŞ_{15}$

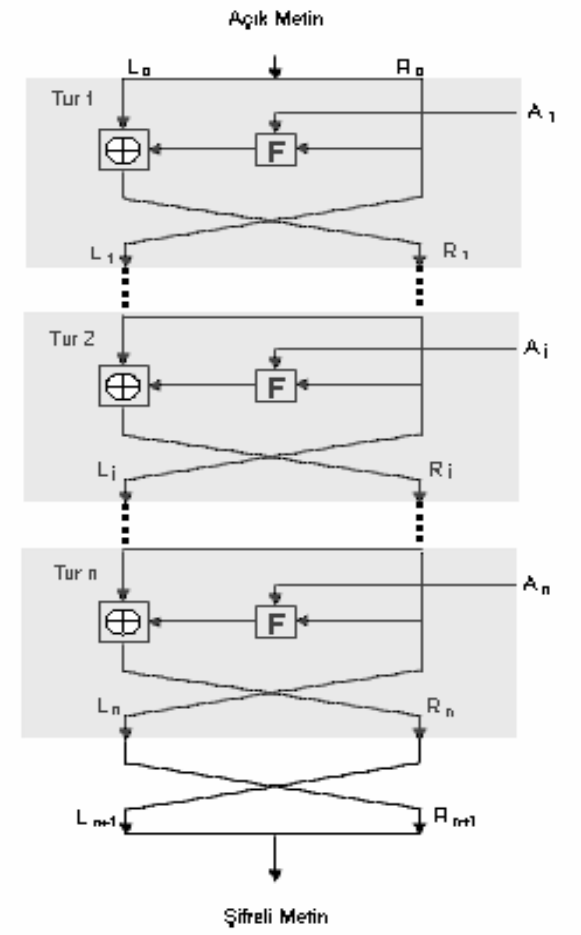
$RŞÇ_1 = LŞÇ_0 \oplus f(RŞÇ_0, A_{16})$

$= RŞ_{16} \oplus f(RŞ_{15}, A_{16})$

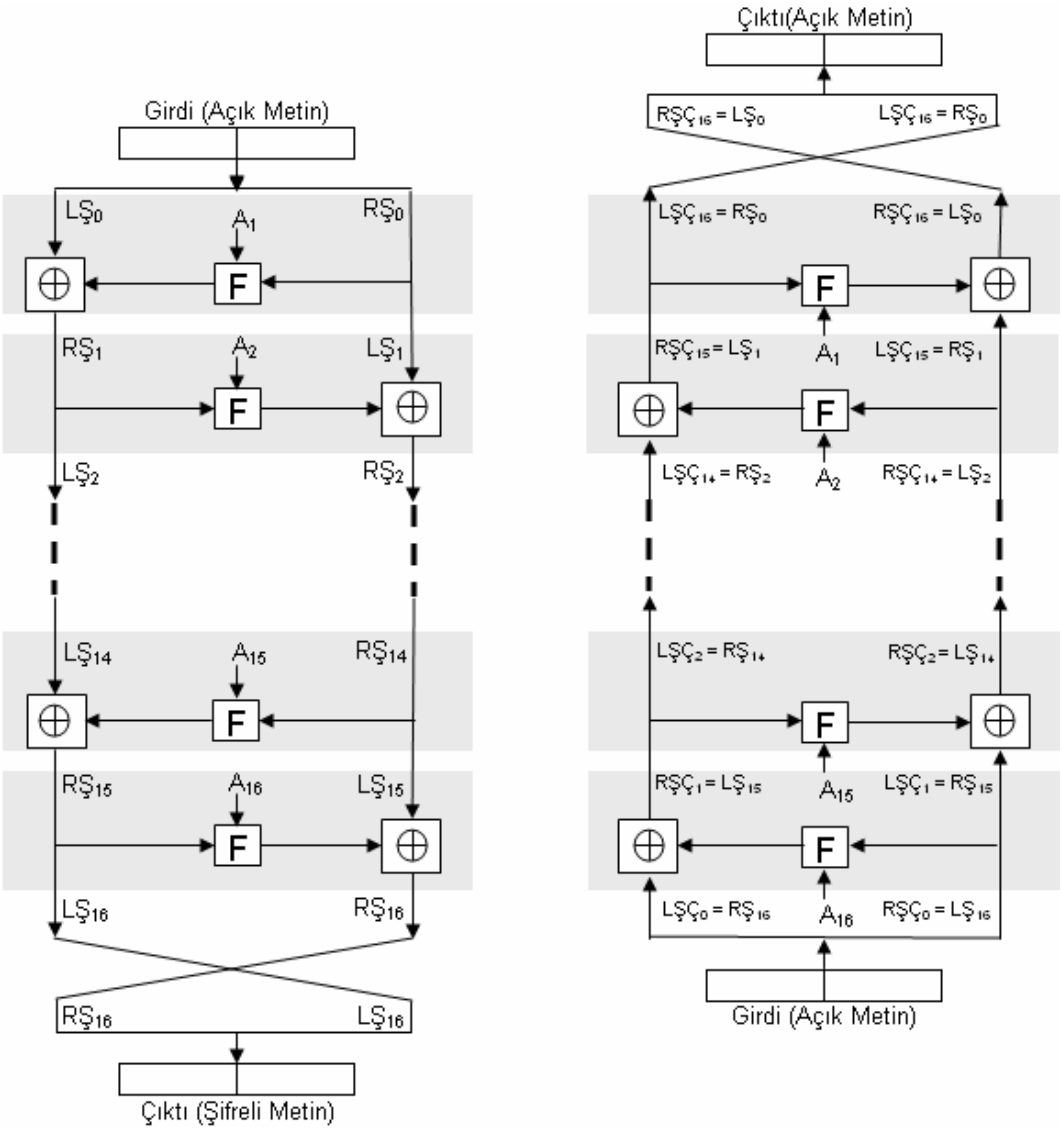
$= [\ LŞ_{15} \oplus f(RŞ_{15}, A_{16})\ ] \oplus f(RŞ_{15}, A_{16})$ şeklinde ifade edilmektedir (Stallings, 2003a, s.70).

Feistel ağı, şifre çözme sırasında Şekil 2.4'te gösterildiği gibi çalışmaktadır (Stallings, 2003a, s.71).

İki-Balık yordamı toplam 16 turluk Feistel ağından oluşmakta ve 1 ikillik döndürme (rotation) işlemi ile değiştirilmektedir (Nechvatal ve ark., 2000, s.9). Bu yordamda kullanılan Feistel ağının her bir turu iki adet aynı işi yapan g foksiyonundan oluşmakta ve bu fonksiyon 128 ikillik açık metini 32 ikillik kelimelere bölerek (Ferguson ve Schneier, 2003, s.59), 4 adet 8x8 büyüklüğündeki anahtara bağımlı S-kutusu içersinde yeniden sıralandırılmakta ve sonrasında iki g kutusundan elde edilen çıktılar toplanmakta ve en sonunda anahtar eklenmektedir (Nechvatal ve ark., 2000, s.9).



Şekil 2. 3. Klasik Feistel ağı (Stallings, 2003a, s.68)



Şekil 2. 4. Feistel ağında şifreleme ve şifre çözme (Stallings, 2003a, s.70)

2.4.4. MARS

MARS yordamı, anahtar ekleme, 8 turluk anahtarsız ileriye doğru karıştırma, 8 turluk anahtarlı ileriye doğru dönüştürme, 8 turluk anahtarlı geriye doğru dönüştürme, 8 turluk anahtarsız geriye doğru karıştırma ve anahtar çıkarma işlemlerinden oluşan, İŞS finalisti yeni bir tekniktir. IBM firması tarafından, İŞS seçimleri için geliştirilmiştir. Anahtarsız turlar, 8x32 ikil büyüklüğündeki S-kutularını kullanmakta ve anahtar ekleme ile “ayrıcalıklı veya” işlemleri

içermektedir. Anahtarlı turlar ise, 32 ikil anahtar çarpımı ve veriye bağlı döndürme ile anahtar ekleme işlemlerini içermektedir. Her iki tipteki turlarda, esasen, Feistel ağı, 1/4'lük veri bloğunun kalan 3/4'lük veri bloğunu değiştirmesi yöntemiyle orjinal halinden farklılaştırılmıştır (Nechvatal ve ark., 2000, s.9).

MARS yordamı, çok çeşitli sayıda farklı işlem yapmakta ve çalıştırılması diğer İŞS finalistlerine göre, donanım açısından çok fazla maliyet getirmektedir. Bundan daha önemlisi, bu yordamın kullandığı S-kutularının birinin içinde, geliştiricilerinin kendilerinin S-kutuları için koydukları ölçütlere uymayan bir yazılım hatası bulunmaktadır. Daha ciddi bir şekilde bakıldığında, MARS geliştiricilerinin doğrusal şifre çözümleme saldırılarına karşı koydukları ölçüt, bu yordam için bir güvenlik çatlağına dönüşmüştür. Şu ana kadar MARS yordamının kırıldığına dair herhangi bir bilgi olmamasına karşın, ciddi bir şifreleme yordamının içermemesi gereken bir hata barındırdığı için yeterli derecede güvenli olduğu düşünülmemektedir. Bunların yanında, MARS ve RC6 yordamları iyi birer şifreleme yordamı olarak görülseler de Rijndael, Yılan ve İki-Balık daha iyidir ve bunların seçilmesinin daha uygun olacağı ifade edilmektedir (Ferguson ve Schneier, 2003, s.62). Bu sebeplerden ve İŞS finalisti olarak seçilemediğinden dolayı tez kapsamında detaylı bir şekilde incelenmemiştir.

2.4.5. RC6

RC6 yordamı (Ron's Code 6 veya Rivest Cipher 6), RSA Labaratuarları tarafından İŞS seçimleri için geliştirilmiş yeni ve güvenli bir şifreleme tekniğidir. Bu yordam da Feistel yapısını kullanmakta ve 20 turdan oluşmaktadır. Şifreleme turlarında, verilerin karesel fonksiyonları ile düzenlenmiş çeşitli döndürme işlemleri yapılmaktadır. Her bir turda, 32 ikil büyüklüğünde modüler çarpım, toplama, "ayrıcalıklı veya" işlemleri ile tur öncesinde ve sonrasında bir tür gizleme (whitening) ve anahtar ekleme işlemleri yapılmaktadır (Nechvatal ve ark., 2000, s.8).

Toplam 20 turdan oluşan RC6 yordamı, İŞS seçimlerindeki saldırılarda 17nci turuna kadar kırılabilmiştir. Bu durum, RC6 yordamının, güvenlik açısından 5 İŞS finalisti içersine girmesine engel olmasa da, tüm aşamaları ile birlikte kırılmaya çok yakın bir noktada olduğundan rahatsız edicidir (Ferguson ve Schneier, 2003, s.62). Bu sebepten ve İŞS finalisti olarak seçilemediğinden dolayı tez kapsamında detaylı bir şekilde incelenmemiştir.

2.4.6. İŞS

İŞS - Rijndael yordamı, ABD MSTE tarafından yapılan elemede 15 adet aday yordam arasından son 5 içine giren ve final seçiminde de İleri Şifreleme Standardı olarak seçilen yeni gizli anahtarlı şifreleme standardıdır (Ferguson ve Schneier, 2003, s.55).

Bu yordam, ABD MSTE tarafından 1997 yılında İleri Şifreleme Standardı seçimi duyurusuna katılım amacıyla, Belçikalı bilim adamları Dr. Vincent Rijmen ve Dr. Joan Daemen tarafından geliştirilmiştir (Stallings, 2003a, s.140-141).

İŞS olarak şeçilmiş olan Rijndael yönteminde, ilk basamakta, açık metin 16 byte uzunluğunda olacak şekilde alınmakta ve 128 ikillik (bit) anahtar ile “ayrıcalıklı veya” işlemine tabi tutulmaktadır. “Ayrıcalıklı veya” iş1eminin sonucunda elde edilen yeni 16 byte, S-kutularına 8 adet ikil giriş olarak kullanılmakta ve çıktı olarak girişten farklı 8 adet ikil oluşturulmaktadır. 16 adet 8 ikil daha sonra, 4 adet karıştırma grubuna girmektedir. Bu işlem anahtar büyüklüğüne bağlı olarak, 10 ile 14 defa tekrarlanarak şifreli metin elde edilmektedir (Ferguson ve Schneier, 2003, s.55-56).

İŞS yordamı detaylı bir şekilde incelendiğinde, 4 farklı basamaktan oluşan ve kullanılan anahtarın büyüklüğüne göre 10, 12 veya 14 tur içeren bir şifreleme tekniği olduğu görülmektedir (Stallings, 2003a, s.145-146). Anahtar büyüklüklerini de içeren İŞS değiştirgeleri Çizelge 2.1’de verilmiştir.

Çizelge 2. 1. İŞS değiştirgeleri

Anahtar büyüklüğü	4/16/128	6/24/192	8/32/256
Açık metin öbek büyüklüğü	4/16/128	4/16/128	4/16/128
Tur sayısı	10	12	14
Tur anahtarı büyüklüğü	4/16/128	4/16/128	4/16/128
Genişletilmiş anahtar büyüklüğü	44/176	52/208	60/240

Herbir turda gerçekleştirilen işlemler sırasıyla aşağıdaki 4 basamaktan oluşmaktadır;

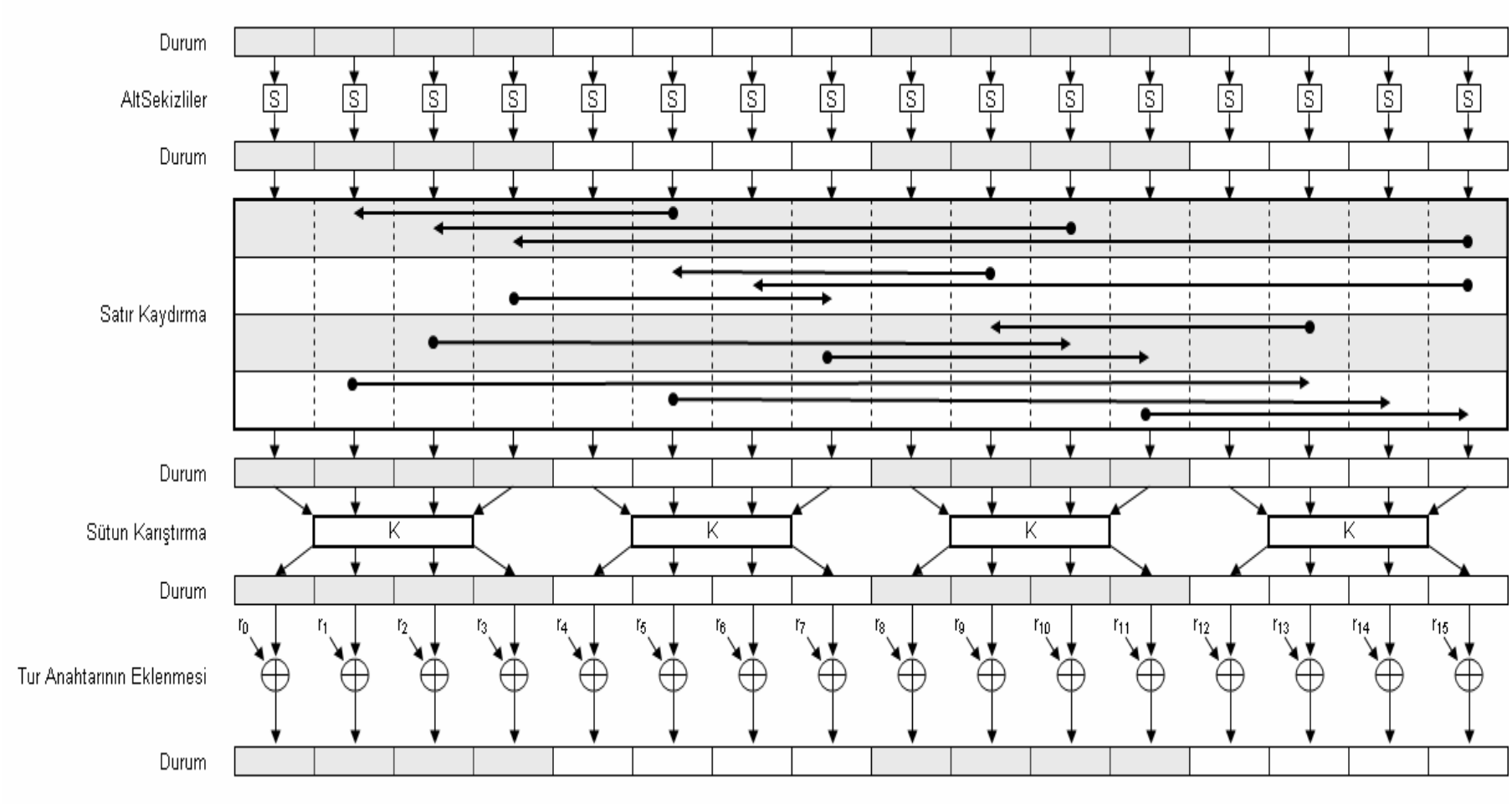
- **İkame (Substitute) byte'ların oluşturulması**: S-kutuları olarak tanımlanan bu basamak, 16 byte'tan oluşan 4x4 matris şeklindeki açık metinin ikame açık metin haline dönüştürülmesidir.

- **Satırların kaydırılması:** Basit bir yeniden sıralama (permütasyon) işlemidir.

- **Kolonların karıştırılması:** Bu basamakta, 4x4 matris büyüklüğündeki ara byte'ların kolonları karıştırılarak yeni ikame byte'lar elde edilmektedir.

- **Tur anahtarının eklenmesi:** Bu basamak, şifrelemedeki esas güvenliği sağlayan anahtarın, "ayrıcalıklı veya" işlemi ile eklendiği basamaktır. Eklenen anahtar, genişletilmiş anahtar olarak tanımlanan, esas anahtardan türetilmiş anahtarın bir parçasıdır.

İŞS yordamında icra edilen her bir tur Şekil 2.5'te ifade edilmektedir.



Şekil 2. 5. İŞS şifreleme turu (Stallings, 2003a, s.149)

Rijndeal yordamında, şifreleme işleminin son basamağını ve şifre çözme işleminin de ilk basamağını oluşturan, tur anahtarının eklenmesi işleminin haricindeki bütün basamaklar, şifre kırıcılar tarafından ters yönde çalıştırılarak kırılabilir. Ancak tur anahtarının "ayrıcalıklı veya" işlemi ile eklenmesi veya çıkartılması, bu açıkları kapatmakta ve yordamın tamamen güvenli hale gelmesini sağlamaktadır. "Ayrıcalıklı veya" işleminin şifrelemede de şifre çözmede de kullanılmasına imkan sağlayan kolaylık şu şekilde açıklanabilir. M, açık metinden elde edilmiş ara metin; A, ilgili tur anahtarı; Ş, şifrelenmiş metin olmak üzere; M $\oplus$ A = Ş şifrelenmiş metini veriyorsa, M $\oplus$ A $\oplus$ A = M olduğundan, Ş $\oplus$ A = M işlemi şifre çözme yaparak açık metini vermektedir (Stallings, 2003b, s.41).

32 ikil işlemcili bir bilgisayarda, $a_{i,j}$ açık metin, $k_{i,j}$ de tur anahtarı olmak üzere, Rijndael yordamını oluşturan basamaklar Çizelge 2.2'de olduğu gibi ifade edilmektedir. Çizelgede, satır kaydırma işlemindeki kolon numaraları, "mod 4" işlemine tabi tutulmaktadır.

Her bir basamağa ait ifadeler, tek bir eşitlikte toplanıp, $T_{0...3}[...]$ ile ifade edilen matris işlemleri ile ilgili sadeleştirmeler yapıldığında, Çizelge 2.3'teki şifrelenmiş metini gösteren eşitlik elde edilmektedir (Stallings, 2003a, s.166). Bu eşitlik Rijndael yordamının en kısa şekilde ifade edilmiş hali olmakla beraber, işlem adedi ve karmaşıklığını gösterme açısından yeterli bilgi içermemektedir.

VŞS, Üçlü VŞS (Triple DES), Uluslararası Veri Şifreleme Yordamı (International Data Encryption Algorithm – IDEA), Uçan-Balık (Blowfish) ve RC5 gibi eski gizli anahtarlı şifreleme yordamları şifrelemeye giriş bloğu olarak 64 ikil büyüklükte açık metin kullanıyorken, İŞS olarak seçilmiş olan Rijndael yordamı 128 ikillik açık metin kullanmaktadır (Stallings, 2003b, s.42). Bu durum, her şifreleme işleminin, eskilerine göre iki kat fazla bilgiyi şifrelediğini göstermektedir.

Çizelge 2. 2. Rijndael yordamında hesaplama (Stallings, 2003a, s166)

Alt Sekizliler	$b_{ij} = S[a_{ij}]$
Satır Kaydırma	$\begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix} = \begin{bmatrix} b_{0,j} \\ b_{1,j-1} \\ b_{2,j-2} \\ b_{3,j-3} \end{bmatrix}$
Sütun Karıştırma	$\begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix}$
Tur Anahtarının Eklenmesi	$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ e_{3,j} \end{bmatrix} = \begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix} \oplus \begin{bmatrix} k_{0,j} \\ k_{1,j} \\ k_{2,j} \\ k_{3,j} \end{bmatrix}$

Çizelge 2. 3. Rijndael yordamının sadeleştirilmiş hali (Stallings, 2003a, s166)

$T_0[x] = \left(\begin{bmatrix} 02 \\ 01 \\ 01 \\ 03 \end{bmatrix} \bullet S[x] \right)$	$T_1[x] = \left(\begin{bmatrix} 03 \\ 02 \\ 01 \\ 01 \end{bmatrix} \bullet S[x] \right)$	$T_2[x] = \left(\begin{bmatrix} 01 \\ 03 \\ 02 \\ 01 \end{bmatrix} \bullet S[x] \right)$	$T_3[x] = \left(\begin{bmatrix} 01 \\ 01 \\ 03 \\ 02 \end{bmatrix} \bullet S[x] \right)$

İŞS finalist yordamlarının şifreleme ve şifre çözme açısından performansları Çizelge 2.4'teki verilerde ve anahtar üretimi açısından

performansları Çizelge 2.5'teki verilerde gösterilmektedir (Nechvatal ve ark., 2000, s.34).

Çizelge 2. 4. Rijndael yordamı şifreleme ve şifre çözme performansı (Nechvatal ve ark., 2000, s.34)

	32-ikil (C)	32-ikil (Java)	64-ikil (C ve Çevirici)	8-ikil (C ve Çevirici)	32-ikil (Akıllı Kart)	Sayısal Sinyal İşlemcileri
MARS	II	II	II	II	II	II
RC6	I	I	II	II	I	II
Rijndael	II	II	I	I	I	I
Yılan	III	III	III	III	III	III
İki Balık	II	III	I	II	III	I

Çizelge 2. 5. Rijndael yordamı anahtar üretim performansı (Nechvatal ve ark., 2000, s.34)

	32-ikil (C)	32-ikil (Java)	64-ikil (C ve Çevirici)	8-ikil (C ve Çevirici)	Sayısal Sinyal İşlemcileri
MARS	II	II	III	II	II
RC6	II	II	II	III	II
Rijndael	I	I	I	I	I
Yılan	III	II	II	III	I
İki Balık	III	III	III	II	III

Rijndael yordamında en çok zaman alan işlem, anahtarın üretilmesidir. Anahtar üretimi, şifreleme işleminde harcanan zamanın %85'ini oluşturmaktadır (Nechvatal ve ark., 2000, s.62). Bu açıdan bakıldığında, gerçekte şifreleme işleminin çok hızlı yapılabildiği, ancak anahtar üretiminin toplam şifreleme zamanını çok uzattığı anlaşılmaktadır. Bütün İŞS finalist yordamları, hem anahtar üretimi hem de şifreleme – şifre çözme performansları açısından, incelendiğinde Çizelge 2.6'da ifade edilen genel performans verilerine ulaşılmıştır (Nechvatal ve ark., 2000, s.34).

Çizelge 2. 6. Rijndael yordamı genel performansı (Nechvatal ve ark., 2000, s.34)

	Şifreleme / Şifre Çözme	Anahtar Üretimi
MARS	II	II
RC6	II	II
Rijndael	I	I
Yılan	III	II
İki Balık	II	III

Yukarıdaki verilere dayanarak, 5 finalist içersinden İŞS yordamı olarak seçilmiş olan Rijndael yordamı, tez kapsamında esas şifreleme tekniği olarak kullanılmış ve hazırlanan programın içersine Microsoft Visual C++ kütüphane fonksiyonları vasıtasıyla yerleştirilmiştir.

2.5. Açık Anahtarlı Şifreleme (Public Key Encryption)

Açık Anahtarlı Şifreleme, aynı zamanda asimetrik şifreleme olarak da adlandırılmakta olup, şifrelemede kullanılan bir adet açık anahtar ve buna bağlı olarak hesaplanan ve şifre çözmede kullanılan bir adet özel anahtardan oluşmakta ve şifreleme ile şifre çözme işlemlerinin her biri için ayrı fonksiyonlar ihtiva etmektedir (Menezes ve ark., 1996, s.283).

Açık Anahtarlı Şifreleme yöntemi, ilk olarak Diffie-Hellman tarafından ortaya atılmış ve uygun şekilde çalışması için gerekli olan kriterler belirlenmiştir (Seberry ve Pieprzyk, 1989, s.90). Bu kriterleri şu şekilde sıralamak mümkündür (Stallings, 1995, s.115):

- Açık anahtarı ile özel anahtar kolayca hesaplanarak üretilebilmelidir.
- Gönderici tarafından açık metin kolaylıkla şifrelenmiş metin haline dönüştürülebilmelidir.

- Alıcı tarafından, özel anahtar kullanılarak, şifrelenmiş metinden açık metin kolaylıkla elde edilebilmelidir.

- Üçüncü şahıslar, açık anahtarı elde etmeleri durumunda, herhangi bir şekilde bu anahtarı kullanarak özel anahtarı hesaplayamamalıdırlar.

- Üçüncü şahıslar, açık anahtarı elde etmeleri durumunda, şifrelenmiş metinden açık metini elde edememelidirler.

- Şifrelemede açık anahtarın yerine özel anahtar, şifre çözmede de özel anahtarın yerine açık anahtar kullanılabilmelidir.

Bu yöntemin geliştirilmesi, Gizli Anahtarlı Şifrelemede olmayan iki eksikliğin giderilmesine dayanmaktadır (Tilborg, 2000, s.105):

- Gizli Anahtarlı Şifrelemede kullanıcı sayısına bağlı olarak kullanılacak anahtar sayısının artması.

- Bilgisayar kontrollü haberleşme sistemlerinde imzanın elektronik karşılığına ihtiyaç duyulması.

Açık Anahtarlı Şifreleme de ağır matematiksel işlemlere dayanmakla beraber, temelde aşağıdaki basitleştirilmiş mantığı kullanmaktadır (Ferguson ve Schneier, 2003, s.27):

m = Açık Metin;

Ş( ) Şifreleme Fonksiyonu, ŞÇ( ) Şifre Çözme Fonksiyonu;

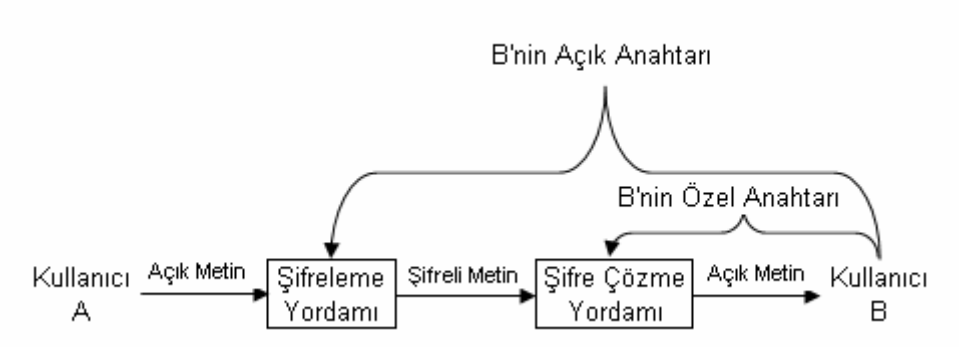
H_{Ali} Ali'nin de sahip olduğu açık anahtar, $Ö_{Ali}$ Ali'nin sahip özel anahtar olmak üzere;

$ŞÇ(Ö_{Ali} , Ş(H_{Ali} , m)) = m$

Bu bağlamda, RSA yordamının, Açık Anahtarlı Şifreleme yöntemleri içersinde en önemli yordam olduğu belirtilmekte (Stallings, 1995, s.109); buna ek

olarak RSA yordamının DH ile beraber en yaygın kullanımı olan Açık Anahtarlı Şifreleme yordamları olduğu ifade edilmektedir (Stallings, 2003b, s.72).

Açık Anahtarlı Şifrelemenin basitleştirilmiş haldeki ifadesi Şekil 2.6'da gösterilmektedir.



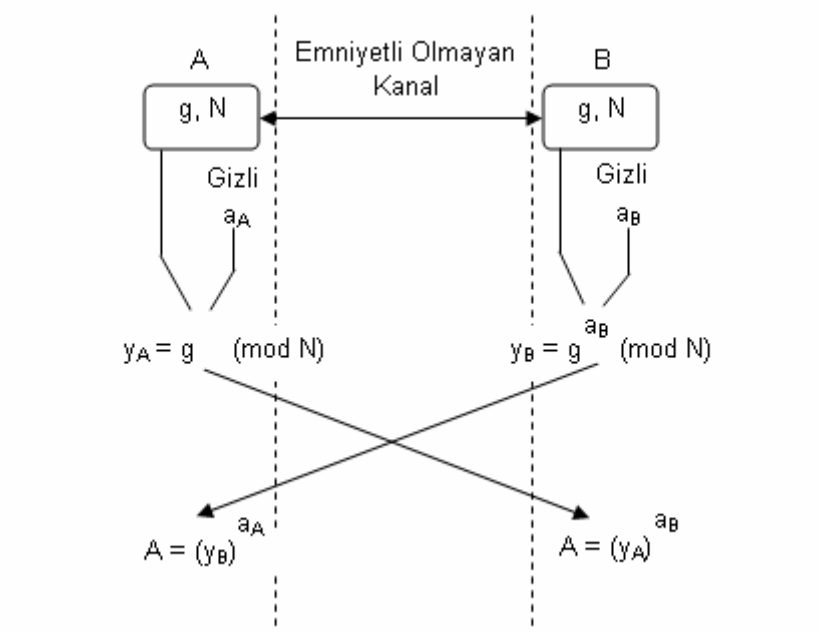
Şekil 2. 6. Açık Anahtarlı Şifrelemenin basitleştirilmiş hali (Stallings, 1995, s.110)

2.5.1. DH (Diffie-Hellman)

Açık Anahtarlı Şifreleme ilk olarak, Whitfield Diffie ile Martin Hellman tarafından 1976 yılında yayınlanan "Şifrelemede Yeni Eğilimler" adlı makalede ortaya atılmıştır (Ferguson ve Schneier, 2003, s.207). DH şifreleme tekniği, yayınlandığından bu yana önemini kaybetmeden kullanılmıştır. Genellikle Diffie-Hellman anahtar alış-veriş yordamı olarak bilinen (Stallings, 2003a, s.293) bu yöntem, herkesin takip edebileceği açık bir kanaldan gizli olan bir anahtarın emniyetli bir şekilde karşıya geçirilmesini sağlamaktadır (Kaufman ve ark., 2002, s.166).

Bu yöntem, mod işleminin sonucunda elde edilen sayının hangi sayıdan elde edildiğinin sadece kalana bakılarak bulunamamasına ve asal sayıların çarpanlarına ayrılamamasına dayanmaktadır. Mod işlemine bağlı olan bilinmezliği, X mod 7 = 3 işlemini inceleyerek daha kolay anlamak mümkündür. Bu işlemde X değeri; 7 ve 3 rakamları bilinmesine rağmen sonsuz adet değer alabilmekte ve ne değer alacağı tahmin edilememektedir.

Diffie-Hellman yordamı, bazı kaynaklar tarafından ne Gizli Anahtarlı Şifreleme ne de Açık Anahtarlı Şifreleme yöntemine dahil edilmektedir (Seberry ve Pieprzyk, 1989, s.98), ancak bir çok kaynak (Stallings, 2003a, s.293; Bishop, 2003, s.233; Ferguson ve Schneier, 2003, s.207) Açık Anahtarlı Şifreleme yöntemi olarak kabul etmektedir. Bu tez kapsamında ise, özel ve açık anahtar içermesi nedeniyle Açık Anahtarlı Şifreleme yöntemi olarak incelenmiş ve anahtar alış-verişi işleminde Şekil 2.7'de gösterilen haliyle (Seberry ve Pieprzyk, 1989, s.99) kullanıldığı değerlendirilmiştir.



Şekil 2. 7. Anahtar bilgisini üssel ifade ile gösteren bir model (Seberry ve Pieprzyk, 1989, s.99)

Normal olarak g^x ifadesinden x ifadesinin elde edilmesi işlemi logaritma olarak tanımlanmaktadır, Diffie-Hellman yordamında ise ayrık logaritma (discrete logarithm) kavramı mevcuttur. Bu kavram (Stallings, 2003a, s.294);

a asal sayı olmak üzere,

$b = t^i \bmod a$ ifadesindeki i'ye karşılık gelmektedir.

Küçük değerler için a sayısının hesap edilmesi mümkün olsa bile, değerler büyüdükçe zorluk üssel bir şekilde artmaktadır (Bishop, 2003, s.230).

Diffie-Hellman yordamı mantık açısından basit gibi görünen ancak hesaplama açısından karmaşık basamaklardan oluşan bir yordamdır. Bu basamaklar şu şekilde ifade edilmektedir (Stallings, 1995, s.341):

Ali tarafında;

- Bir a asal sayısı üretilir;
- t < a olacak şekilde rassal taban t sayısı seçilir,
- Elde edilen t ve a sayıları açık bir kanaldan Ali'den ile Veli'ye gönderilir
- $Ö_{Ali}$ < a olacak şekilde Ali'ye ait (rassal) özel $Ö_{Ali}$ anahtarı seçilir,
- $H_{Ali} = t^{Öali}$ mod a işlemi ile Ali'ye ait H_{Ali} açık anahtarı hesaplanır ve Veli'ye gönderilir,

Veli tarafında;

- $Ö_{Veli}$ < a olacak şekilde Veli'ye ait (rassal) özel $Ö_{Veli}$ anahtarı seçilir,
- $H_{Veli} = t^{Öveli}$ mod a işlemi ile Veli'ye ait H_{Veli} Açık anahtarı hesaplanır ve Ali'ye gönderilir,

Ali tarafında;

- $G = (H_{Veli})^{Öali}$ mod a olacak şekilde gizli anahtar hesaplanır,

Veli tarafında;

- $G = (H_{Ali})^{Öveli}$ mod a olacak şekilde gizli anahtar hesaplanır.

Yukarıdaki işlemlerin sonunda Ali ile Veli, aynı değerlere sahip olan gizli anahtarı elde etmiş olurlar. Bu gizli anahtar, Gizli Anahtarlı Şifreleme yöntemlerinde hem şifrelemede ve hem de şifre çözmede; Açık Anahtarlı Şifreleme yöntemlerinde ise sadece şifre çözmede kullanılmak üzere gizliliğini koruyacak şekilde saklanır. Ali ile Veli'nin açık kanallardan birbirlerine gönderdikleri açık anahtarlar ise RSA gibi Açık Anahtarlı Şifreleme yöntemlerinde şifreleme yapmak için kaydedilir.

Herhangi bir özel anahtar üzerinde anlaşmak için kullanılan Diffie-Hellman yordamının matematiksel olarak basit bir örneği aşağıdadır (Stallings, 2003a, s.295);

- Ali, asal a sayısını a = 353 ve ilgili taban sayısını t = 3 olarak üretsin,
- Üretilen a ve t sayıları açık bir kanaldan Ali'den Veli'ye gönderilsin,
- Ali, $Ö_{Ali} < a$ olacak şekilde rassal $Ö_{Ali} = 97$ sayısını üretsin,
- Ali, $H_{Ali} = t^{Öali} \bmod a$ formülüne uygun olarak $3^{97} \bmod 353 = 40$ sayısını hesaplasın ve Veli'ye H_{Ali} açık anahtarı olarak göndersin,
- Veli, $Ö_{Veli} < a$ olacak şekilde rassal $Ö_{Veli} = 233$ sayısını üretsin,
- Veli, $H_{Veli} = t^{Öveli} \bmod a$ formülüne uygun olarak $3^{233} \bmod 353 = 248$ olarak hesaplasın ve Ali'ye H_{Veli} açık anahtarı olarak göndersin,
- Ali, $G = (H_{Veli})^{Öali} \bmod a$ olacak şekilde $248^{97} \bmod 353 = 160$ gizli anahtar değerini hesaplasın,
- Veli, $G = (H_{Ali})^{Öveli} \bmod a$ olacak şekilde $402^{233} \bmod 353 = 160$ gizli anahtar değerini hesaplasın.

Yukarıdaki basamakların sonucunda elde edilen 160 sayısı, gizli anahtar olarak, tahmin edilebilecek büyüklükteki bir değerdir. Ancak a asal sayısı 2000 ile 4000 ikil uzunlukta olduğunda, elde edilecek gizli anahtarın, pratikte

hesaplanamayacağı öngörülmektedir (Ferguson ve Schneier, 2003, s.208). Bu yüzden şifrelemede büyük sayılar ile işlem yapma ihtiyacı mevcuttur. Bu tez kapsamında kullanılan Microsoft Visual C++ programlama lisanı ve benzeri lisanlarda, mevcut değişken sınırları izin vermediğinden dolayı, 2000 ile 4000 ikil uzunluktaki sayılarla doğrudan matematiksel işlemler yapılamamaktadır. Tamamen ayrı bir konu olan, büyük sayılarla hızlı matematiksel işlemler, bu tez kapsamında incelenmemiştir. Şifrelemede kullanılan hızlı hesaplama tekniklerinin nasıl uygulanacağı, henüz dilimize çevrilmemiş olan “The Art Of The Computer Programming” (Knuth, 1969, s.347-351) ve benzeri kaynaklarda detaylı bir şekilde incelenmiştir ancak, uygulama amacıyla hazırlanan programda, ilgili matematiksel işlemler, MSDN kütüphane fonksiyonları aracılığıyla gerçekleştirilmiştir ve bu yüzden tez içersinde DH, RSA ve İŞS yordamlarındaki matematiksel işlemlerin bilgisayar aracılığı ile nasıl yapılacağı üzerinde durulmamıştır.

DH yordamındaki en önemli konulardan bir tanesi, seçilen asal sayının yeterli emniyeti sağlayacak şekilde büyük olup olmadığıdır. Bu açıdan bakıldığında, DH yordamının, binlerle ifade edilen ikil büyüklükte anahtar kullandığı, buna karşın gizli anahtarlı şifreleme yöntemlerinin 128, 192 veya 256 ikil büyüklükte anahtar kullandığı belirtilmekte ve açık anahtarların büyüklüğünün, her zaman, gizli anahtarların büyüklüğünden fazla olduğu ifade edilmektedir (Ferguson ve Schneier, 2003, s.216).

2.5.2. RSA

RSA yordamı, geliştiricileri Ron Rivest, Adi Shamir and Len Adleman’ın soyadlarının **R**ivest-**S**hamir-**A**dleman olarak kısaltılması ile isimlendirilmiş ve yayımlandığı yıl olan 1970’ten bu yana, Açık Anahtarlı Şifreleme yöntemleri içersinde en geniş kullanım alanı bulmuş olan yöntemdir (Stallings, 2003a, s.266).

RSA yordamı, bu tez kapsamında hazırlanan programda kullanıldığı için detaylı bir biçimde incelenmiş ve programın kullandığı MSDN kütüphane fonksiyonlarına temel teşkil eden hesaplamalar basit bir şekilde açıklanmıştır.

Bu yöntem, 512 ikillik ve daha büyük sayıların çarpanlarına ayrılamamasından ve yayımlandığından bu yana herhangi bir şekilde kırıldığına dair bir bilgi ile karşılaşılmamasından dolayı, güvenli bir yöntem olarak kabul edilmektedir. En iyi çarpanlarına ayırma yöntemlerinin bile bu büyüklükteki bir sayıyı, günümüz teknolojisi ile bin yıllarla ifade edilen zamanda çarpanlarına ayırabileceği tahmin edilmektedir (Kaufman ve ark., 2002, s.153).

Şifrelemeye genel olarak bakıldığında, temelde iki şartı yerine getirmesi beklenmektedir. Bunlardan ilki şifreleme ve şifre çözmenin hızlı bir şekilde yapılabilmesi, ikincisi de şifrelemede kullanılan anahtarların ve açık metinin bir başkası tarafından hesaplanamaması ve/veya hesaplanmasının kabul edilebilir zaman diliminde yapılamamasıdır. Bundan dolayı ve elde edilecek güvenliğin pratik bir şekilde uygulanabilir olması için, şifreleme ve şifre çözme yöntemlerinde kullanılan yordamların, oldukça hızlı çalışması gerekmektedir. Dolayısıyla matematiksel olarak formüle edilen işlemler, sadelikten daha çok, hızlı hesaplama yöntemlerine dayandırılmalıdır. Buna göre, 123^{54} mod 678 işlemini gerçekleştirmek için (Kaufman ve ark., 2002, s.154-155); 123 sayısının kendisini 54 defa katlaması ve sonunda ortaya çıkacak yaklaşık 100 ikillik sayının 678'e bölünmesi yönteminin kullanılmasının, hız açısından uygun olmayan bir hesaplama olduğu açıktır. Bunun yerine,

$54 = (((((1)2+1)2)2+1)2+1)2$ ise

$123^{54} = (((((123)^2\ 123)^2)^2\ 123)^2\ 123)^2 = 87 \bmod 678$ diğer bir ifade ile

$123^{54} = (123^{27})^2 = 171^2 = 29241 = 87 \bmod 678$ yönteminin kullanılmasının,

8 adet çarpma ve 8 adet bölme işleminin haricinde, gereksiz çarpma ve bölme işlemlerini ortadan kaldırdığı için, hız açısından daha etkin bir yöntem olduğu değerlendirilmektedir.

Yukarıda bahsedilen 100 ikil büyüklükteki sayılarla yapılan işlemlerde, boşa harcanan MİB zamanını azaltmak için takip edilen yöntem ile, 2000 veya daha fazla ikil büyüklükteki sayılarla yapılan işlemlerde boşa harcanan MİB zamanını azaltmak için takip edilen yöntem, matematiksel açıdan farklılık göstermektedir. 2000 veya daha büyük ikil sayılarla yapılan işlemlerde, hızı artırmak için takip edilen yöntemler, logaritmik işlemlerden Fourier dönüşümlerine kadar birçok matematiksel ifadeyi içermekte ve hesaplama açısından daha da karmaşık hale gelmektedir (Knuth, 1969, s.258-279).

RSA yordamı, hesaplama açısından; 512 ikilden daha büyük asal sayılarla yapılan işlemlere, Euler Totient fonksiyonuna ve en büyük ortak bölen işlemine dayanmaktadır.

p ve q asal sayı,

n = pq , 0 < m < n ve k tamsayı olmak üzere,

$$m^{k\,\phi(n)\,+\,1} = m^{k\,(p\,-\,1)\,(q\,-\,1)\,+\,1} = m \bmod n$$

eşitliği, RSA yordamı için temel teşkil etmektedir. Bu eşitlikte kullanılan $\phi(pq) = (p-1)(q-1)$ ifadesi Euler Totient fonksiyonu olarak bilinmekte ve bu fonksiyonun özelliği kullanılarak aşağıdaki eşitlik ve denklikler elde edilmektedir (Stallings, 2003a, s.269);

$$ed = k\,\phi(n) + 1,$$

$$ed \equiv 1 \bmod \phi(n),$$

$$d \equiv e^{-1} \bmod \phi(n)$$

Bu denklikler; d ve dolayısıyla e sayıları $\phi(n)$ ile aralarında asal ise, diğer bir ifade ile EBOB $(\phi(n), d) = 1$ ise geçerlidir.

RSA yordamında n ve e değerleri Açık anahtarı; p, q, ϕ(n) ve d değerleri de özel anahtar olarak tanımlanmaktadır. n sayısı p ve q çarpanlarına ayrıldığı takdirde, buradan ϕ(n) ve d değerleri de elde edilebilmektedir. Bu yüzden, şifrelemede çarpanlarına ayırma işleminden sıklıkla bahsedilmektedir (Ferguson ve Schneier, 2003, s.232).

Örnek olarak yukarıda incelenen $123^{54} = 87 \bmod 678$ işleminin, herhangi sayı değerleri ile en hızlı şekilde hesaplanabilmesi için, aşağıdaki matematiksel formül ve yordam kullanılmaktadır (Stallings, 1995, s.125):

m sayısı pozitif bir tam sayı olmak üzere,

$m = b_k b_{k-1} \ldots\ldots b_0$ olacak şekilde ikilik sayıya dönüştürülsün,

Formül;

$$m = \sum_{b_i \neq 0} 2^i$$

olacağından,

$$a^m = a^{\left(\sum_{b_i \neq 0} 2^i\right)} = \prod_{b_i \neq 0} a^{(2^i)}$$

$$a^m \bmod n = \left(\prod_{b_i \neq 0} a^{(2^i)}\right) \bmod n = \prod_{b_i \neq 0} \left(a^{(2^i)} \bmod n\right)$$

şeklinde ifade edilmektedir.

Bu durumda, yordam;

c ← 0 ; d ← 1

for i ← k **downto** 0

do c ← 2 x c

d ← (d x d) mod n

if b_i = 1

then c ← c + 1

d ← (d x a) mod n

return d

basamaklarından oluşmaktadır.

En iyi bilinen çarpanlarına ayırma yöntemi bile oldukça yavaş çalıştığından, 512 ikillik bir sayının, bilgisayar veya özel elektronik donanımla 30.000 yılda çarpanlarına ayrılabileceği öngörülmektedir. Ancak çarpanlarına ayırma işlemi, gelişen teknolojinin getirdiği imkanlar kullanılarak, çok daha hızlı yapılabilirse, RSA yordamının kırılması mümkün olabilecektir (Kaufman ve ark., 2002, s.153).

RSA yordamı, kullandığı büyük asal sayılar nedeniyle, karmaşık işlemler içermekte ve çözüm için özel anahtara bağımlılık yaratmaktadır. Bu durum, RSA kullanılarak yapılan şifreleme işlemlerinin güvenirliğini sağlamakta ve açık metinin veya özel anahtarın istenmeyen kişilerin eline geçmesini engellemektedir. Yapılan işlemlerin basite indirgenmiş hali, aşağıdaki örnekte gösterilmektedir (Bishop, 2003, s.235-236).

$c = m^e \bmod n$ ve $m = c^d \bmod n$ olduğuna göre;

p = 7, q = 11, n = 77, $\phi(n) = 60$, e = 17, d = 53 değerleri alınsın;

A'dan Z'ye İngilizce harfler 00 ile 26 değerlerinden oluşsun;

Gönderilecek mesaj: "HELLO WORLD " olsun;

Ali, Açık Anahtar olarak n = 77 kullanarak;

07^{17} mod 77 = 28

04^{17} mod 77 = 16

11^{17} mod 77 = 44

…

03^{17} mod 77 = 75

işlemlerinin sonucunda 28 16 44 44 42 38 22 42 19 44 75 şifreli metini elde eder ve bunu Veli'ye gönderir.

Veli, d = 53 Özel Anahtarını kullanarak;

28^{53} mod 77 = 07

16^{53} mod 77 = 04

44^{53} mod 77 = 11

…

75^{53} mod 77 = 03

işlemlerinin sonucunda 07 04 11 11 14 26 22 14 17 11 03 sayılarına karşılık gelen "HELLO WORLD" açık metinini elde eder.

Yukarıdaki işlemler incelendiğinde, açık metinde, üç adet 11 (L harfine) sayısına karşılık, şifrelenmiş metinde, üç adet 44 sayısı görülmektedir. Bu şekilde basit ve düz olarak RSA yordamını kullanarak, bir kitabı harf harf şifreleyip gönderdiğimizde, şifre kırıcı kişi ve programlar, özellikle sözlük saldırısını kullanarak, şifrelenmiş metinden, herhangi bir Özel Anahtar kullanmadan kitabın orijinal halini elde edebilir. O yüzden harf harf şifreleme yerine, blok halinde şifreleme veya akım şeklinde şifreleme teknikleri kullanılmaktadır.

2.5.3. Dijital İmza Standardı – DİS (Digital Signature Standard – DSS)

Dijital İmza Standardı, ABD MSTE tarafından, El Gamal imza şemasına dayanarak geliştirilmiştir ve ondan daha hızlı çalışan yeni bir imza standardıdır (Kaufman ve ark., 2002, s.172).

Bu yöntem, aşağıdaki değişkenleri kullanmaktadır;

Açık Anahtar Değişkenleri:

(1) p; 512 ikilden 1024 ikile kadar değişen uzunlukta bir asal sayı,

(2) q; 160 ikil uzunlukta ve p-1 sayısının çarpanı olan bir asal sayı,

(3) $h < p-1$ ve $h^{(p-1)/q} \bmod p > 1$ olmak üzere, $g = h^{(p-1)/q} \bmod p$,

(4) $y = g^x \bmod p$ (y Açık Anahtar),

Özel Anahtar:

(5) $x < q$ (160 ikil uzunlukta) bir sayı,

İmzalama:

(6) $k < q$ olan bir rassal sayı,

(7) $r \text{ (imza)} = (g^k \bmod p) \bmod q$,

H (m), GÇY olmak üzere,

(8) $s \text{ (imza)} = (k^{-1} (H(m) + xr)) \bmod q$,

Doğrulama:

(9) $w = s^{-1} \bmod q$,

(10) $u_1 = (H(m) * w) \bmod q$,

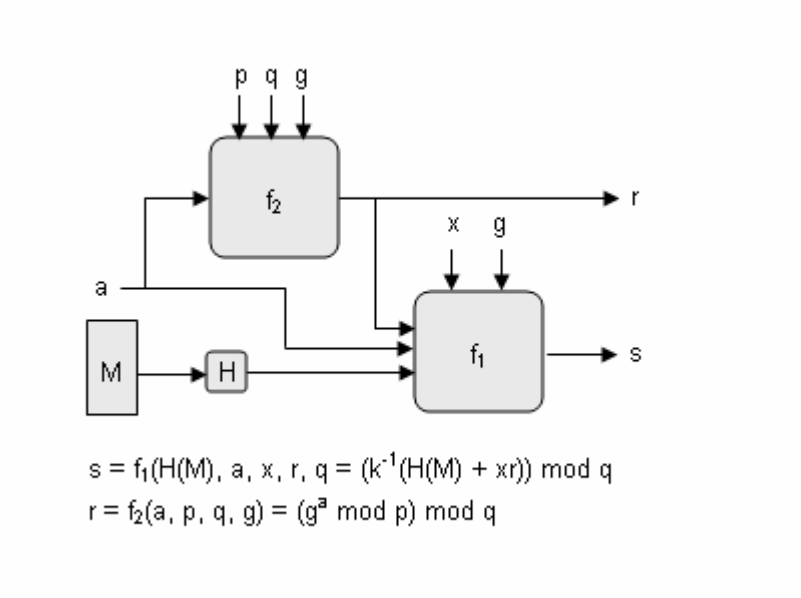
(11) $u_2 = (rw) \bmod q$,

(12) $v = ((g^{u1} * y^{u2}) \bmod p) \bmod q)$

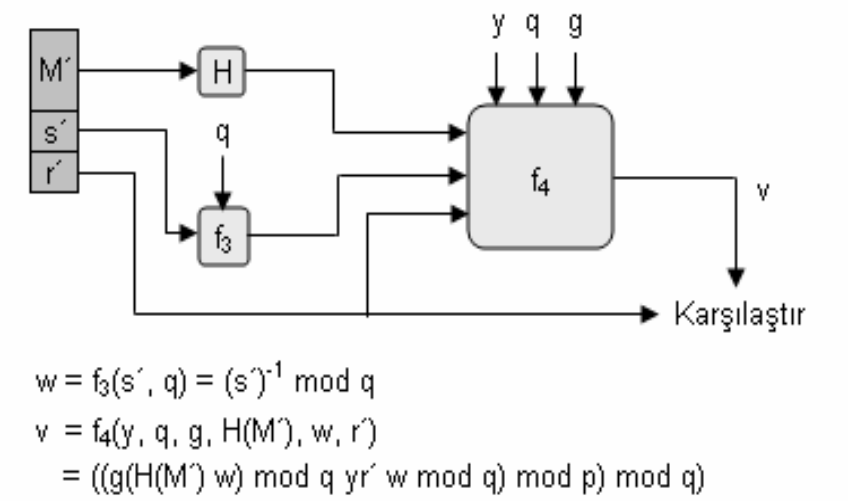
(13) v = r ise imza doğrulanmıştır.

Bu yöntemde öncelikle p, q ve g değişkenleri belirlenerek açık olarak kullanıcılara duyurulur. Özel Anahtar x, Açık Anahtarı da y olarak belirlenir. Daha sonra mesaj m'den, (6) ve (7) numaralı basamaklardaki gibi r ve s imzaları elde edilerek karşı tarafa gönderilir. Karşı taraf da, aldığı imzaları kullanarak (9), (10), (11), (12) ve (13) numaralı işlemleri yaparak imzanın doğru olup olmadığına karar verir (Schneier, 1996, s.488).

Şekil 2.8 ve 2.9, Dijital İmza Standardının imzalama ve imza ile ilgili sağlama yapılması sırasında kullanılan işlemleri, basitleştirilmiş halde göstermektedir (Stallings, 2003a, s.395).



Şekil 2. 8. Dijital imzanın oluşturulması



Şekil 2. 9. Alınan dijital imzanın sağlaması

Dijital İmza Standardı, imzalama işleminin Özel Anahtarını açığa çıkarmayacağından; kimsenin, verilen bir mesaj için, Özel Anahtarı bilmeden aynı imzaları üretemeyeceğinden; kimsenin, verilen imzalara karşılık gelen mesajları üretemeyeceğinden ve kimsenin, imzalanmış olan bir mesajı, aynı imzayı sabit tutacak şekilde değiştiremeyeceğinden, güvenli bir yöntem olarak kabul edilmektedir (Kaufman ve ark., 2002, s.174).

DİS, güvenli olmasına karşın, şifrelemede ve anahtar dağıtımında kullanılamayacağından; içinde muhtemel bir açık kapı bulundurabileceğinden; RSA yordamına göre daha yavaş olduğundan; RSA yordamının fiili olarak kullanımda olan bir standart olmasından; halka açık bir şekilde geliştirilmemesinden ve bir standart olarak duyurulmadan önce analiz için yeterli zaman verilmemesinden; diğer patentler ile sorun yaşabileceğinden ve anahtar büyüklüğünün çok küçük olmasından dolayı eleştirilmektedir (Schneier, 1996, s.484-486).

2.5.4. Eliptik Eğri (Elliptic Curve)

Eliptik Eğri ile şifreleme yöntemi, Victor Miller ve Neil Koblitz tarafından geliştirilmiş ve üzerinde çok sayıda araştırma yapılarak çeşitli endüstri standartlarına uyumu sağlanmış olan (Vanstone, 2004, s.2) ve RSA yöntemine göre, MİB zamanını daha az kullanan (Stallings, 2003b, s.78) Açık Anahtarlı Şifreleme yöntemidir.

Şifrelemede kullanılan Eliptik Eğriler gerçekte elips şekline sahip değildir. a, b, c, d ve e değerleri gerçek bir sayı olmak üzere;

(1) $y^2 + axy + by = x^3 + cx^2 + dx + e$ eşitliğini, daha basit haliyle ise

(2) $y^2 = x^3 + ax + b$ eşitliği gibi benzer 3ncü dereceden denklemleri kullanan bir yöntemdir (Stallings, 2003a, s.298).

Açık Anahtarlı Şifreleme yöntemleri içersinde, son yıllarda en çok Eliptik Eğri ile şifreleme yönteminin kullanımın arttığı bildirilmektedir. ABD Milli Güvenlik Ajansı, 23 Ekim 2004 tarihinde Eliptik Eğri ile şifrelemeyi can alıcı bir teknoloji olarak tanımlamış ve milli güvenliği ilgilendiren yazılımlarda kullanılmasını istemiştir. Bundan dolayı ABD'de, özellikle anahtar yönetimi konusu Eliptik Eğri yöntemine doğru kaydırılmaktadır (Vanstone, 2004, s.2).

Bu yöntem esasen, verilen gizli bir anahtar ile Eliptik Eğri üzerindeki açık bir noktayı elde etme mantığı ile çalışmakta olup, ayrık logaritma problemi üzerine kurulmuştur. Asal sayının büyüklüğü ve noktaların çokluğu, konunun güvenliğini oluşturduğundan, kabul edilebilir bir zamanda tahmin edilemeyen bir asal sayı seçilmeli ve çok miktarda nokta elde edilmelidir (Atay, 2005, s.2).

Eliptik Eğri ile şifreleme yönteminde, 2^n formundaki ikil eğriler ile asal sayılardan oluşan eğriler kullanılmaktadır. Bunlardan asal sayılardan oluşan eliptik eğriler daha çok yazılım temeline dayanan şifreleme yöntemlerinde kullanılmakta; 2^n formundaki ikil eliptik eğriler ise donanım temelli şifreleme yöntemlerinde kullanılmaktadır (Stallings, 2003a, s.301).

Asal sayılardan oluşan eliptik bir eğri oluşturmak amacıyla;

p asal sayı 23, a = b = 1, x = 5, y = 4 olmak üzere,

$y^2 \bmod p = (x^3 + ax + b) \bmod p$ eşitliği,

$4^2 \bmod 23 = (5^3 + 5 + 1) \bmod 23$,

$16 \bmod 23 = 131 \bmod 23$,

16 = 16 şeklinde çözümlenir ve,

$E_{p=23}(a = 1, b = 1)$ eliptik eğrisinin, (0,0)'dan (p-1, p-1)'e kadar Çizelge 2.7'de gösterilen değerleri aldığı görülür (Stallings, 2003a, s.301-s302).

Bu yöntem, tez çalışması kapsamında hazırlanan programda kullanılmadığından, detaylı bir şekilde incelenmemiştir.

Çizelge 2. 7. E_{23} (1, 1) Eliptik eğrisi üzerindeki noktalar

(0, 1)	(6, 4)	(12, 19)
(0, 22)	(6,19)	(13, 7)
(1, 7)	(7, 11)	(13, 16)
(1, 16)	(7, 12)	(17, 3)
(3, 10)	(9, 7)	(17, 20)
(3, 13)	(9, 16)	(18, 3)
(4, 0)	(11, 3)	(18, 20)
(5, 4)	(11, 20)	(19, 5)
(5, 19)	(12, 4)	(19, 18)

2.6. Şifrelemeye Karşı Saldırı Teknikleri

Şifrelemeye Karşı Saldırılar (Cryptographic Attacks), pasif ve aktif olmak üzere ikiye ayrılmaktadır. Pasif saldırılar, bilginin gizliliğinin ortadan

kalkmasına sebep olan saldırılardır. Aktif saldırılar ise hem bilginin gizliliğinin ortadan kalkmasına sebep olmakta, hem de bilginin silinmesi, değiştirilmesi ve veri bütünlüğünün bozulmasına sebep olmaktadır (Menezes ve ark., 1996, s.41).

2.6.5 nolu paragrafta açıklanan Ortadaki Adam Saldırısı ve benzeri saldırılar, gelen-giden mesajlarda değişiklik yaptığından dolayı aktif bir saldırı olarak kabul edilmekte buna karşın, şifreli metinin elde edilmesi sırasında yapılan dinleme işlemi ve benzeri saldırılar, pasif bir saldırı olarak değerlendirilmektedir (Bishop, 2003, s.7).

Şifrelemeye karşı yapılan saldırılarda kullanılan yordamın ne olduğunun bilinmesinin büyük önemi vardır. Hangi yordamın kullanıldığının bilinmesi; açık metinin bilinmesi veya anahtar büyüklüğünün bilinmesi veya açık metin-şifreli metin ikililerinin bilinmesi gibi veriler ile birleştirildiğinde şifrenin kırılmasını ve anahtarın elde edilmesini kolaylaştırmaktadır.

Şifrelemeye karşı saldırı, genellikle şifreleme işleminden daha çok zaman almaktadır. Gerçekte, günümüz şifreleme yöntemlerinin güvenliğini sağlayan esas nokta da, aradaki zaman farkının fazlalığı ve şifrelemeye karşı kullanılan saldırı yöntemlerinin anahtarları ve/veya açık metni elde etmelerinin kabul edilebilir zaman diliminde yapılamamasıdır. Bu yüzden, matematiksel manada şifreleme yöntemleri güvenilir olsa bile, uygulamada oluşabilecek zayıf yönlerin karşı saldırı yöntemlerine açık vermeden kapatılması ve hangi karşı saldırı yönteminin, ne gibi zayıf noktalardan faydalandığının iyi bilinmesi gerekmektedir.

Şifrelemeye karşı saldırıda iki kriter göz önünde bulundurulmaktadır. Bunlardan ilki, şifre çözme maliyetinin, şifreli bilginin değerinden daha düşük olması; ikincisi ise, şifreli bilginin, değerini, şifre (haberleşilen kişi tarafından) çözüldüğünde hala koruyor olmasıdır (Stallings, 1995, s.26).

Şifrelemeye karşı kullanılan saldırı yöntemlerinin karmaşıklığı aşağıdaki üç ana kritere göre belirlenmektedir (Schneier, 1996, s.9):

- **Veri karmaşıklığı:** Saldırı için ne kadar veriye girdi olarak ihtiyaç duyulduğu.

- **Hesaplama karmaşıklığı:** Saldırının başarılı olması için gereken zaman.

- **Depolama (storage) ihtiyacı :** Saldırı için gerekli olan hafıza**.**

Bütün saldırılar için geçerli olan öncelikli konu, zincirin en zayıf halkasının bulunmasıdır. Bundan sonraki basamağı, en zayıf halkanın, en hızlı şekilde kırılmasını sağlayacak yöntemin seçilmesi oluşturmaktadır.

2.6.1. Tam Güç Saldırısı (Brute Force Attack)

Tam Güç Saldırısı, bütün olasılıkların denenmesine dayanan saldırı şeklidir. Bu yöntemde, kullanılan anahtarın yeterince büyük olması, harcanan zamanı pratik olmayacak şekilde artırmaktadır. Şifre çözücüler, diğer yöntemlere, tam güç yönteminin pratik olmadığı durumlarda geçmektedirler (Stallings, 1995, s.24).

Bu saldırı yönteminin başarılı olabilmesi için, şifreli metin üzerinde, bütün anahtarlar sırayla denenir ve elde edilen sonucun anlamlı bir metin olup olmadığına bakılır. Saldırı hızını, kırılmaya çalışılan anahtarın uzunluğu ile her bir anahtarın kırılmasında harcanan zaman belirlemektedir. Çizelge 2.8’de, 1995 yılında kullanımda olan donanımlar ile Tam Güç Saldırısı yöntemini kullanarak bir anahtarın kırılmasının maliyeti verilmiştir (Ferguson ve Schneier, 2003, s.152-s153).

Çizelge 2. 8. 1995 yılındaki Donanımsal Tam Güç Saldırısı için gereken ortalama zaman tahminleri (Ferguson ve Schneier, 2003, s.153)

	Anahtarların İkil Uzunlukları					
Maliyet (TL)	40	56	64	80	112	128
135×10^3	2 saniye	35 saat	1 yıl	70.000 yıl	10^{14} yıl	10^{19} yıl
$1,35 \times 10^6$	0,2 saniye	3,5 saat	37 gün	7.000 yıl	10^{13} yıl	10^{18} yıl
$13,5 \times 10^6$	0,02 saniye	21 dakika	4 gün	700 yıl	10^{12} yıl	10^{17} yıl
135×10^6	2 milisaniye	2 dakika	9 saat	70 yıl	10^{11} yıl	10^{16} yıl
$1,35 \times 10^9$	0.2 milisaniye	13 saniye	1 saat	7 yıl	10^{10} yıl	10^{15} yıl
$13,5 \times 10^9$	0,02 milisaniye	1 saniye	5,4 saniye	245 gün	10^9 yıl	10^{14} yıl
135×10^9	2 mikrosaniye	0,1 saniye	32 saniye	24 gün	10^8 yıl	10^{13} yıl
$1,35 \times 10^{12}$	0.2 mikrosaniye	0,01 saniye	3 saniye	2,4 gün	10^7 yıl	10^{12} yıl
$13,5 \times 10^{12}$	0.02 mikrosaniye	1 milisaniye	0,3 saniye	6 saat	10^6 yıl	10^{11} yıl

Not: 1 $ = 1,35 TL olarak alınmıştır.

Çizelge 2.9'da da, başka bir kaynakta benzer bir şekilde belirtilmiş olan; (Stallings, 1995, s.26) şifrelenmiş bir metinden anlamlı bir metin elde edilinceye kadar uygulanan ve gerektiğinde tüm anahtar ihtimallerinin denenmesini sağlayan tam güç saldırısı ve buna temel teşkil edebilecek, anahtar büyüklükleri ve bu anahtarların kırılması için gerekli olan zaman süreleri gösterilmektedir.

Çizelge 2. 9. Tam Güç Saldırısı için gereken zaman ve diğer veriler (Stallings, 1995, s.26)

Anahtar Büyüklüğü	Alternatif Anahtar Adedi	Mikrosaniyede Bir Şifreleme	Mikrosaniyede 10^6 Şifreleme
32 ikil	$2^{32} = 4,3 \times 10^9$	2^{31} µs = 35.3×10^9	2,15 ms
56 ikil	$2^{56} = 7,2 \times 10^{16}$	2^{55} µs = 1142 yıl	10,01 saat
128 ikil	$2^{128} = 3,4 \times 10^{38}$	2^{127} µs = $5,4 \times 10^{24}$	$5,4 \times 10^{18}$ yıl
26 karakter	$26! = 4,03 \times 10^{26}$	2×10^{26} µs= $6,4 \times 10^{12}$ yıl	$6,4 \times 10^6$ yıl

2.6.2. Açık Metine Dayanan Saldırılar (Plaintext Attacks)

Açık Metine Dayanan Saldırılar, bir şekilde elde edilmiş açık metin ve şifreli metin üzerinden kullanılan anahtarın ele geçirilmesini amaçlayan saldırılardır (Ferguson ve Schneier, 2003, s.31). Saldırının başarılı olması

sonucunda elde edilen anahtar ile, şifreleyen taraf kullandığı anahtarı değiştirmediği sürece, şifrelenmiş bütün mesajlar rahatlıkla çözülebilir. Dolayısıyla şifreli haberleşmenin herhangi bir güvenliği kalmaz, en azından yeni anahtar üretilinceye kadar geçecek zaman süresince yapılan tüm haberleşme, saldırıyı yapan tarafın eline geçmiş olur.

Açık Metine Dayanan Saldırılar, açık metin ile buna karşılık gelen şifreli metinin elde edilmesiyle yapılan saldırılardır. Açık metinin elde edilmesi şifreli metinin elde edilmesine göre daha zordur ancak, açık metinin sonsuza kadar saklanmasının da mümkün olmadığı durumlar mevcuttur. Örneğin; saldırı yapılacak bir şehrin ismi, şifreli olarak gönderilse bile, şehrin ismi saldırı gerçekleştikten sonra kendiliğinden ifşa olup açık metin haline dönüşür. Şifreli metini bir şekilde elde eden ve bunu açık metin elde edilinceye kadar elinde tutan üçüncü kişiler bu saldırı yöntemini rahatlıkla kullanabilirler (Kaufman ve ark., 2002, s.46).

Bu saldırı tekniğinde, belirli bir uzunluktaki açık metin ile buna karşılık gelen şifreli metin ikilisi kullanılabileceği gibi, birden fazla açık metin – şifreli metin ikilisi de kullanılabilir. Harfleri birebir şifreleyen çok basit bir yordam kullanıldığı düşünüldüğünde, bu teknikten faydalanılarak, açık metinde kullanılan 5 adet A harfinin şifreli metinde 5 adet X harfine karşılık geldiği tespit edilebilir. Bu şekilde her harfin karşılığının elde edilmesi, özellikle birden fazla açık metin - şifreli metin ikilisinin kullanılmasıyla çok kolay hale gelir (Welsh, 1988, s.111).

Saldırının başarıya ulaşması için, açık metin ve ilgili şifreli metin ikilileri için ayrı ayrı iki çizelge oluşturulur ve bu çizelgeye karşılık gelebilecek anahtarlar sırayla denenir. Açık metinden ilgili şifreli metini elde eden anahtar bulununcaya kadar, işlem devam ettirilir (Stallings, 1995, s.67-68). Şifreleme ve şifre çözmede kullanılan anahtarın elde edilmesi için gereken zaman, n açık metin-şifreli metin adedi olmak üzere aşağıda ifade edilen formül ile hesaplanmaktadır (Stallings, 1995, s.69).

$$(2^{56})\frac{2^{64}}{n} = 2^{120 - \log_2 n}$$

2.6.3. Şifrelenmiş Metine Dayanan Saldırılar (Ciphertext Attacks)

Şifrelenmiş Metine Dayanan Saldırılar, saldıran tarafta sadece, bir şekilde elde edilmiş şifreli metin varsa, uygulanabilecek en zor saldırı yöntemidir (Ferguson ve Schneier, 2003, s.31).

Bu saldırı yöntemi, şifrelenmiş metinin elde edilmesi imkansız olmadığından hiçbir zaman göz ardı edilemez. Bu yöntemde uygulanabilecek tekniklerden bir tanesi, bütün anahtarların sırayla denenerek, bir şekilde elde edilmiş şifreli metinin açık metin haline dönüştürülmeye çalışılmasıdır. Ancak bu yöntemde, deneme sırasında elde edilen metinin gerçek açık metin olup olmadığının anlaşılması, ayrı bir zorluk içerir. Gerçek çözümün bulunabilmesi için, açık metin olduğu düşünülen verinin anlamlı bir veri olup olmadığı her defasında test edilmelidir. Anlamlı bir verinin elde edilip edilmediğinin anlaşılması, herhangi bir dilde yazılmış bir metin olduğunda başka bir zorlukta, tamamen teknik, sayısal ve konuşma-yazma lisanlarının dışında olduğunda ayrı bir zorluktadır. Şifreleme yöntemleri, zaten bu saldırıya karşı çok güçlü olacak şekilde geliştirildiklerinden, bu saldırı ile aşılamayacaktır ancak, şifre kırıcılar, şifrelenmiş metinin yanında, başka veriler de elde edip bu saldırı yöntemiyle beraber kullanabileceğinden bilinmesi gereken bir saldırı yöntemi olarak ele alınmıştır (Kaufman ve ark., 2002, s.45).

Bu saldırı tekniğinde esas amaç, açık metinin elde edilmesi olmakla beraber, mümkün olan durumlarda ilgili anahtarın da elde edilmesiyle de uğraşılır (Bishop, 2003, s.218.)

Bu yöntemde, şifreleme sırasında, alfabedeki harflerin yer değiştirmesine dayanan Sezar şifresi veya benzerinin kullanılması durumunda, Açık Metine Dayanan Saldırılardaki tekniğe benzeyen bir teknik uygulanmaktadır. Elde edilen

şifreli metin öncelikle, karakter bazında frekans analizine tabi tutulmakta ve takiben İngilizce (veya ilgili lisanın) harf dağılımlarını temsil eden frekans tablolarıyla karşılaştırılmaktadır. Sonuçlar içersinde en yüksek korelasyon oranına sahip, açık metin olabilecek muhtemel metinler incelenerek, İngiliz dilinde (veya ilgili lisanda) anlamlı bir ifadeye karşılık gelip gelmediğine bakılmaktadır (Bishop, 2003, s.219). Bu yöntem istatistiksel bir yöntem olarak kabul edilmektedir ancak, günümüzde Sezar şifreleme yöntemine benzer teknikler kullanılmamaktadır, bu yüzden çok fazla uygulama alanı olmadığı değerlendirilmektedir.

2.6.4. Doğum Günü Saldırısı (Birthday Attack)

Aynı salonda, 23 kişide bir, aynı doğum gününe (365 gün düşünüldüğünde) sahip olma olasılığı %50'yi geçmektedir (Ferguson ve Schneier, 2003, s.33). Bu durum, birinci saldırıda belirli bir doğum gününe sahip olan bir kişinin seçilip, ikinci saldırıda da aynı doğum gününe sahip ikinci bir kişinin bulunması olasılığına dayanan Doğum Günü Saldırısı olarak adlandırılmaktadır. Bu saldırı, saniyede milyon adet mesaj çırpılayabilen bir makinenin, 64 ikillik çırpı çıktısını hangi mesaja ait olduğunun 600.000 yılda tespit edilmesi yerine, hangi iki mesajın (örneğin; 1nci mesaj ile 23ncü mesajın) aynı çırpı çıktısına sahip olduğunun 1 saat gibi kısa bir sürede tespit edilmesi üzerine kurulmuştur (Schneier, 1996, s.166).

Diğer bir ifade Doğum Günü Saldırısı, "kaç (k) adet mesaj girişi yapıldığında aynı çırpı sonucunu Ç(x) = Ç(y) veren iki adet x ve y mesajı bulunur?" sorusunun cevaplanması yoluyla yapılmaktadır. Matematiksel olarak aşağıdaki gibi ifade edilen bu saldırı, kullanılması düşünülen anahtar büyüklüğünün ikil (bit) değer olarak 2 katına çıkarılmasıyla önlenebilmektedir. "1 ile n arasında değer alabilen tam sayılar kümesinden, k adet örneklem alındığında, kaç adet seçimden sonra, daha önceden seçilmiş bir sayı tekrar seçilebilir? Diğer bir ifade ile aynı sayının tekrar seçilme olasılığı nedir?" sorusunu sorduğumuzda

O(n, k) olasılık değerinden bahsedilmekte ve bu olasılığın yaklaşık olarak √n değerine eşit olduğu bildirilmektedir (Stallings, 1995, s.204-205).

Bu saldırının, elektronik ortamda Ali ile Veli'nin kendilerini birbirlerine tanıtma (authentication) işlemi sırasında olduğunu ve saldırıyı yapan kişinin sırayla deneyerek x ve y mesajlarının aynı çırpı sonucunu verdiğini hesapladığını varsayalım; bu durumda, saldırıyı yapan kişi, Ali'nin gönderdiği mesajın yerine kendi ürettiği mesajı koyarak, Ali'nin yerine Veli'ye, kendini tanıtmış olur (Ferguson ve Schneier, 2003, s.91). Dolayısıyla saldırıyı takiben Veli'nin aldığı mesajlar gerçek Ali'den değil, Doğum Günü Saldırısını yapan sahte Ali'den gelmeye başlar. Karşılığında, Veli'nin güvenli olduğunu düşünerek gönderdiği mesajlar da, sahte Ali'nin eline geçer. Bu saldırının internet üzerinden yürütülen bankacılık işlemlerine karşı yapıldığı düşünüldüğünde, büyük bir finansal sorun yaratacağı rahatlıkla görülebilir. Bu saldırıya karşı tedbir olarak, dijital imza yöntemleri geliştirilmiş ve dijital imzayı internet kullanıcılarına güvenli bir şekilde temin eden firmalar ortaya çıkmıştır. Dijital imza firmaları, tarayıcıların kaynak kodlarının içine koydukları kendi firmalarının imzaları dahil, bir çok güvenlik gereksinimini karşılamaktadır.

2.6.5. Ortada Buluşturma Saldırısı (Meet In The Middle Attack)

Ortada Buluşturma Saldırısı, bir uçtan şifreleme yapılırken, diğer uçtan da şifre çözme yapılması ve sonuç verilerinin karşılaştırılarak, uyuşan iki veri grubunun bulunup bulunmadığına karar verilmesine dayanan saldırı yöntemidir (Schneier, 1996, s.358).

Çift VŞS şifrelemesinin kırılması için, 2 x 56 = 112 ikillik tam güç saldırısına ihtiyaç duyulmasına karşın, çift VŞS şifrelemesi Ortada Buluşturma Saldırısı ile daha az basamakta daha az zaman harcayarak kırılabilmektedir (Kaufman ve ark., 2002, s.111).

Bu yöntemde hesaplamalar, açık metin = m, şifreli metin = şm olmak üzere, bilinen (m, şm) ikilisi üzerinden yapılmaktadır. Ş( ) Şifreleme Fonksiyonu,

ŞÇ( ) Şifre Çözme Fonksiyonu olduğu ve çift şifreleme yapıldığı varsayıldığında, gözlemlere dayanarak;

$şm = Ş_{Anahtar-2}(Ş_{Anahtar-1}(m))$ eşitliğinin olduğu ve buna bağlı olarak,

$x = Ş_{Anahtar-1}(m) = ŞÇ_{Anahtar-2}(şm)$ eşitliklerinin belirlenebildiği bilinmektedir.

Şifrenin kırılması için, m açık metini, 2^{56} adet Anahtar-1 değeri ile şifrelenir ve sonuçlar bir tabloya kaydedilir ve daha sonra sıralandırılır. Takiben, şm şifreli metini, 2^{56} adet Anahtar-2 değeri ile şifre çözme işlemine tabi tutulur ve sonuçlar ayrı bir tabloya kaydedilir ve daha sonra sıralandırılır. Her iki tablodaki değerler birbirleriyle karşılaştırılıp, eşleşen değerler tesbit edilirse, kullanılan iki adet anahtara karşılık, bir adet (m, şm) şifreli metin - açık metin ikilisi elde edilir. Böylece, Ortada Buluşturma Saldırısı başarıya ulaşmış olur (Stallings, 2003a, s.176). Saldırının sonucunda ilgili Anahtar-1 ve Anahtar-2 değerleri, saldırı yapan kişinin eline geçtiğinden, yapılan şifreleme işleminin herhangi bir önemi kalmamakta ve aynı anahtarlar kullanılmaya devam edildiği sürece, gönderilen şifreli metinlerin gizliliği ortadan kalkmaktadır. Anahtar değerleri yenilenmesi durumunda ise, saldırı yeniden yapılmakta ve yeni anahtarlar da elde edilebilmektedir. Bundan dolayı, çift VŞS kullanılarak şifreleme yapıldığında, toplam anahtar uzunluğu iki katına çıkmış gibi ve şifreleme işlemi daha güvenilir hale gelmiş gibi görünse de, bu saldırı nedeniyle, yapılan haberleşmenin herhangi bir gizliliğinin ve güvenliğinin kalmadığı değerlendirilmektedir.

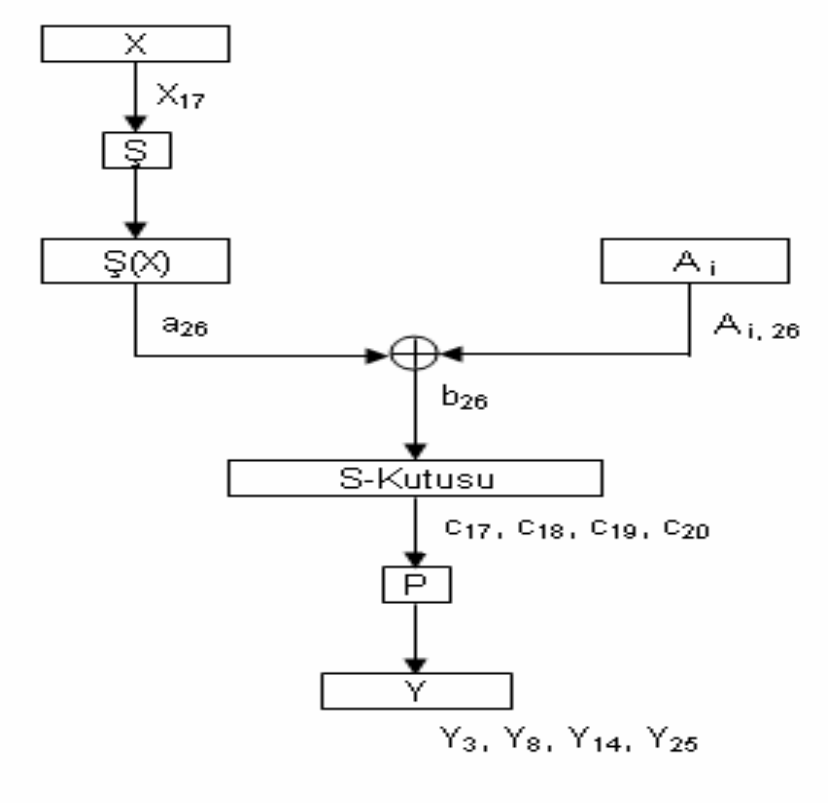
2.6.6. Doğrusal Şifre Çözümleme Saldırısı (Linear Cryptanalysis Attack)

Doğrusal Şifre Çözümleme Saldırısı, Mitsui Matsui tarafından geliştirilmiş olan, bir şekilde elde edilmiş şifreli metinler ile ilgili açık metinlerin analiz edilerek, kullanılan anahtara ait bazı ikillerin tahmin edilmesi veya anahtarın tamamen elde edilmesi işlemidir. Bu yöntemde, eğer anahtarın tamamı elde edilemiyorsa, tahmin edilemeyen kısmı için sırayla deneme yapılmaktadır. VŞS ile ilgili olarak, anahtarın bir ikillik kısmını elde etmeye yönelik böyle bir

saldırı yapıldığında, 1/2 - 5/16 olasılıkla başarı sağlanmaktadır (Schneier, 1996, s.290-293). VŞS için 1 turluk doğrusal tahmin işlemi Şekil 2.10'da gösterilmektedir.

Burada yapılan işlem basitleştirilecek olursa, açık metine ait bir ikil ve bunun karşılığında elde edilmiş şifreli metine ait 4 ikilin, "ayrıcalıklı veya" işlemine tabi tutulduğu görülür; X_{17} açık metinine ait ikil ve Y_3, $Y_{8,}$ $Y_{14,}$ Y_{25} ilgili şifreli metine ait ikil oldukları kabul edildiğinde;

$X_{17} \oplus Y_3 \oplus Y_8 \oplus Y_{14} \oplus Y_{25} = A_{i,\,26}$ eşitliğinin sonucu olarak anahtara ait bir ikil tesbit edilebilmektedir. Bu işlem, anahtara ait üç ikilin elde edilmesi için yapıldığında başarı şansı ona göre azalarak 1/2 - 0.0061 olmaktadır (Schneier, 1996, s.292).



Şekil 2. 10. VŞS için 1 turluk doğrusal tahmin işlemi (Schneier, 1996, s.291)

Aynı işlem diğer bir şekilde ifade edildiğinde, M açık metin, ŞM şifreli metin, A anahtar ve a,b,c sayıları ikil göstergeler olmak üzere;

$M[a_1, a_2, ... ,a_n] \oplus ŞM[b_1, b_2, ... , b_m] = A[y_1, y_2, ... , y_x]$ olacak şekilde açık metin ve ilgili şifreli metin işleme tabi tutulduğunda, sonuç yarıdan fazla 0 çıkıyorsa $A[y_1, y_2, ... , y_x] = 0$; 1 çıkıyorsa $A[y_1, y_2, ... , y_x] = 1$ kabul edilir. Bu işlemde, ne kadar çok açık metin - şifreli metin ikilisi kullanılırsa doğru sonuca ulaşma ihtimali o kadar artmaktadır. Şifrelemede kullanılan anahtarın tamamının tesbit edilmesi amacıyla, her ikil anahtar değeri için bu işlem tekrarlanır. Bu saldırı, doğrusal eşitliklerle ilgili olduğundan, her defasında anahtara ait bir ikil değerinin belirlenmesinin daha uygun bir yaklaşım olduğu değerlendirilmektedir (Stallings, 2003a, s.85).

2.6.7. Türevsel Şifre Çözümleme Saldırısı (Differential Cryptanalysis Attack)

Türevsel Şifre Çözümleme Saldırısı, 1990 yılında Eli Biham ve Adi Shamir tarafından ortaya atılmış ve açık metinleri birbirlerinden farklı olan, ancak şifreli metin halleri birbirleri ile aynı olan eş şifreli metin çiftlerine dayanan bir saldırı tekniğidir (Schneier, 1996, s.285).

Bu saldırı yöntemi, VŞS şifreleme algoritmasını 2^{55} adımdan daha az adımda kırabilen ilk yöntemdir (Stallings, 1995, s.55).

Türevsel Şifre Çözümleme Saldırısını oluşturan basamaklar, basitleştirilmiş bir yaklaşımla, aşağıdaki gibi ifade edilmektedir. Öncelikle, birbirlerinden belirli farklılıkları olan iki açık metin seçilir. Seçim işlemi rassal olarak da yapılabilir. Seçilen iki açık metin arasındaki farklılık, çıktı olarak elde edilen şifreli metinde de farklılığa sebep olacaktır. Aynı işlem, A_i (anahtarın i'nci ikili) için 0 ve 1 değerleri verilerek, sırayla farklı açık metin ikilileri için yapılır ve sonuç olarak elde edilen farklı şifreli metinler ayrı ayrı tablolara kaydedilir. Daha sonra, "ayrıcalıklı veya" kullanılarak açık metinler arasındaki farklılıklar elde edilir ve ayrı bir tablonun satırına yazılır. Bu tablonun sütunlarına şifreli

metinlerin “ayrıcalıklı veya”ları kaydedilir ve tablonun içi, “hangi açık metin “ayrıcalıklı veya”sına karşılık hangi şifreli metin “ayrıcalıklı veya”sı kaç defa elde edilmiş” sorusunu cevaplayack şekilde doldurulur (Schneier, 1996, s.285-286). Böylelikle A_i’nin muhtemel değeri tahmin edilmiş olur. Yani belli bir olasılık dahilinde, $A_i = 0$ veya $A_i = 1$ olacak şekilde, anahtarın ilgili ikili bulunur.

Bu işlemler ile anahtarın tamamının elde edilmesi, çok uzun zaman alacak olursa, anahtarın belli bir kısmının, kabul edilebilir zaman dilimi içinde, Türevsel Şifre Çözümleme Saldırısı ile, kalan kısmının ise Tam Güç Saldırısı ile tesbit edilmesi yöntemi uygulanır (Schneier, 1996, s.288).

Bu saldırı, VŞS için aşağıdaki detaylandırıldığı gibi ifade edilmektedir. Şifrelenecek açık metin, m_0 ve m_1 iki yarım parçasından oluşan m, A_i ilgili anahtar ikili, Ş şifreleme fonksiyonu olmak üzere;

$m_{i+1} = m_{i-1} \oplus Ş\ (m_{i,}\ A_i)$ eşitliği ara safhada elde edilen yarım mesajı ifade etmekte ve iki adet açık metin (m ve m') ile bunların arasındaki “ayrıcalıklı veya” ile ifade edilen farkı gösteren $\Delta m = m \oplus m'$ eşitliği ile beraber kullanılmaktadır. Buradan aşağıdaki ifade elde edilmektedir;

$$\Delta m_{i+1} = m_{i+1} \oplus m'_{i+1}$$

$$= [\ m_{i-1} \oplus\ Ş(m_{i,}\ A_i)\ ] \oplus [m'_{i-1} \oplus Ş(m'_{i,}\ A_i)]$$

$$= \Delta m_{i-1} \oplus [Ş(m_{i,}\ A_i) \oplus Ş(m'_{i,}\ A_i)]$$

Farkları aynı olan birçok açık metin çifti olduğu düşünülürse, aynı farkın, aynı alt anahtar parçası ile aynı çıktı farkına karşılık geleceği kabul edilebilir. Daha sembolik bir ifade ile, X “ayrıcalıklı veya” girdi farkı, O olasılıkla, Y “ayrıcalıklı veya” çıktı farkına neden olmaktadır denebilir. Bundan dolayı, eğer Δm_{i-1} ile Δm_i yüksek bir olasılıkla biliniyorsa, yüksek bir olasılıkla Δm_{i+1} değeri de elde edilebilir. Bu şekilde, belirli bir sayıda fark ortaya çıkarıldığında, Ş fonksiyonunda kullanılan alt anahtar parçası da tahmin edilebilir hale gelmektedir (Stallings, 2003a, s.84).

2.6.8. Ortadaki Adam Saldırısı (Man In The Middle Attack)

Ortadaki Adam Saldırısı, Ali ile Veli haberleşiyorken üçüncü bir kişinin hattı dinlemesi ve bununla yetinmeyip karşılıklı gönderilen mesajları, silerek, değiştirerek veya yerine kendi mesajını koyarak, sahte Ali ve sahte Veli oluşturması işlemidir. Bu saldırı anahtar alış-verişi sırasında yapılmaktadır ve aşağıdaki basamaklardan oluşur (Schneier, 1996, s.48).

- Ali, Veli'ye kendine ait Açık Anahtarı gönderir. Üçüncü şahıs bu anahtarı önler ve yerine kendi Açık Anahtarını koyarak Veli'ye gönderir.

- Veli, Ali'ye kendine ait Açık Anahtarı gönderir. Üçüncü şahıs bu anahtarı da önler ve yerine kendi Açık Anahtarını koyarak Ali'ye gönderir.

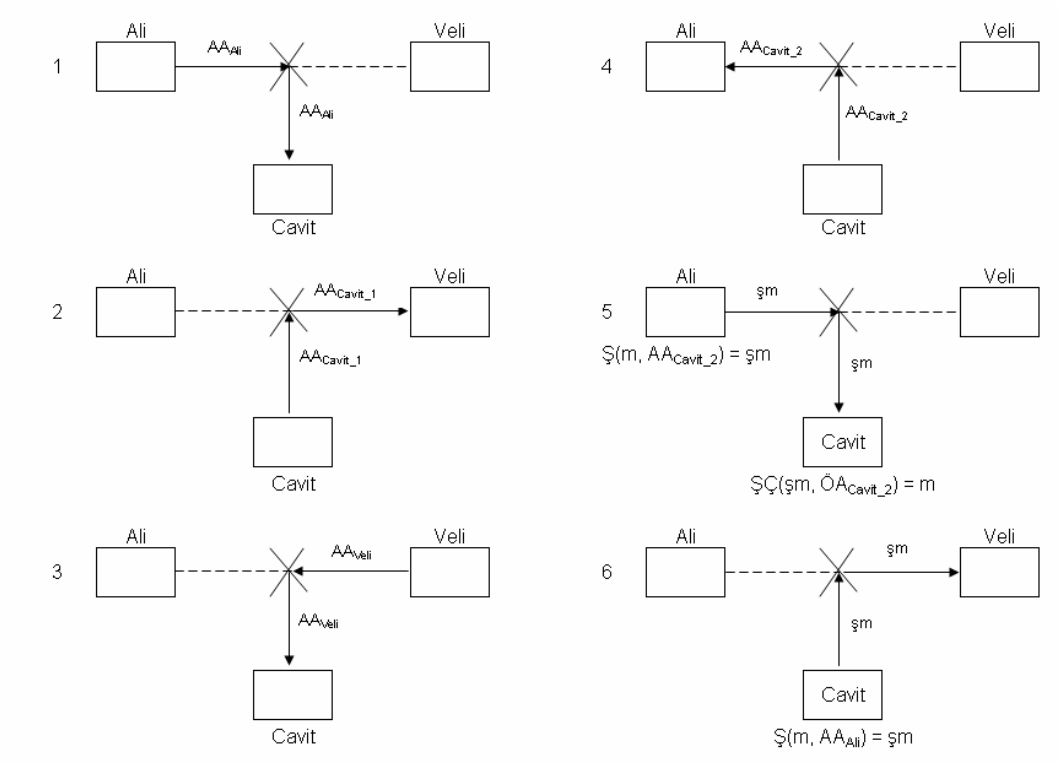
- Ali, Veli'ye mesaj gönderirken sahte Veli'nin Açık Anahtarını kullanarak mesaj gönderir. Sahte Veli olan üçüncü şahıs, gönderilen mesajı kendi Özel Anahtarı ile çözer ve Veli'ye gerçek Ali'nin Açık Anahtarı ile şifreleyerek gönderir.

- Bu durum Veli, Ali'ye mesaj gönderilirken de yapılır ve bir önceki basamakta sahte Veli olan aynı üçüncü şahıs, bu defa sahte Ali olarak görev yapar.

Şekil 2.11, Ortadaki Adam Saldırısının, yukarıda bahsedilen son basamağı hariç olmak üzere, nasıl gerçekleştirildiğini göstermektedir. Ali ile Veli, karşılıklı anahtar alış-verişi sırasında, bu tip bir saldırıya uğrarlarsa, gönderilen anahtarların gerçek Ali'den mi veya gerçek Veli'den mi veya sahte bir kişiden mi geldiğini anlayamazlar (Bishop, 2003, s.252).

Bu saldırı, fark edilebilecek herhangi bir gecikmeye neden olmazsa; saldırıya maruz kalan Ali ile Veli, güvenli bir şekilde haberleştiklerini zannederler (Schneier, 1996, s.49). Şifrelemede oluşabilecek en tehlikeli durumlardan bir tanesidir ki, farkedilmediği sürece çok uzun zaman uygulanabilir.

Ortadaki Adam Saldırısına maruz kalan göreceli şifreli haberleşmede, Ali ile Veli'nin birbirlerine gönderdikleri mesajlar aynen ulaşır ancak, şifreli olarak gönderdiklerini düşündükleri mesajlar, sanki açık bir mesajmış gibi araya giren üçüncü bir şahıs tarafından elde edilir. Bu da bütün güvenliğin ve gizliliğin ortadan kalkması anlamındadır.



Şekil 2. 11. Ortadaki adam saldırısı

2.6.9. Diğer Saldırı Yöntemleri

Diğer saldırı yöntemlerine, İlgili Anahtar Saldırısı (Related-Key Attack) , Seçilen Anahtar Saldırısı (Chosen-Key Attack), Çarpışma Saldırısı (Collision Attack), Güç Analizi Saldırısı (Power Analysis Attack) ve Zamanlama Saldırısını (Timing Attack) dahil etmek mümkündür.

İlgili Anahtar Saldırısında, kullanılan anahtarlar arasındaki bağlantılardan faydalanılmaktadır. Saldırıyı yapan kişi, anahtarları bilmese bile, şifreleme fonkisyonlarına erişimi varsa ve anahtarların aralarında bir bağlantı olduğunu

biliyorsa bu saldırı ile başarıya ulaşma şansı vardır. Yeni anahtarın, bir önceki anahtar değerinin bir artırılması ile elde edildiği gerçek ve benzeri durumlarda, bu saldırı yöntemi ile, takip eden anahtarlar tesbit edilebilmektedir. Seçilen Anahtar Saldırısında ise anahtarın belli bir kısmı sabit kalacak şekilde bir seçim yapılır ve kalan kısmı da İlgili Anahtar Saldırısı ile tesbit edilmeye çalışılır (Ferguson ve Schneier, 2003, s.45).

Çarpışma Saldırısı, aynı anahtar değerinin çok uzun süre kulanılması sonucunda farklı iki açık metinin aynı şifreli metine karşılık geldiğinin tespit edilmesi sonucunda başarıya ulaşmaktadır. Böyle bir durumu fark eden kişi, gönderilen açık metini aynı şifreli metin çıktısını veren farklı bir açık metin ile değiştirebilmektedir (Ferguson ve Schneier, 2003, s.100). Bu da şifreleme güvenliğini ortadan kaldırmaktadır.

Başka bir saldırı tekniği de bilgisayarların kontrollü alanlarından anahtar ve diğer gizli bilgilerin elde edilmesidir. Modern işletim sistemleri, hata ayıklayıcılar (debugger) tarafından kullanılabilecek özelliklere sahip olduklarından, hafızanın okunmasına izin vermekte ve/veya neden olmaktadırlar. Buna benzer bir şekilde, Unix işletim sisteminde çekirdek dökümü (core dump) olarak bilinen, kullanıcılara ait gizli veriler kullanılarak istenilen bilgilerin ilgili hafıza alanından alınması işlemi de bir saldırı şeklidir. Diğer büyük bir tehlike de, işletim sistemlerindeki hesaplardır. Yönetici yetkisine sahip olan kullanıcılar, şifreleme yordamlarının çalıştırıldığı bilgisayarlara girip, Unix işletim sisteminde olduğu gibi istedikleri hafıza alanını okuyabilmektedirler. Bu da yapılan şifrelemenin önemini ortadan kaldırmaktadır. Dolayısıyla, şifreleme yordamları, kendilerini bu tip saldırıların tamamına karşı koruyabilecek özellikte olmalıdır (Ferguson ve Schneier, 2003, s.144).

Güç analizi saldırısı (power analysis attack), akıllı kart gibi cihazlar ile işlem yaparken, harcanan gücün her işlem için tek tek gözlemlenmesine dayanan bir saldırı yöntemidir. Harcanan güç hangi işlemin yapıldığına dair bilgi vermektedir. Çarpma işlemi ve sıfır rakamının yazılması, toplamaya ve bir rakamının yazılmasına göre daha çok güç harcadığından, yapılan işlemler üçüncü

şahıslar tarafından güç analizi yöntemiyle çözümlenebilmektedir (Stallings, 2003a, s.143).

Zamanlama saldırısı (timing attack), fiziksel saldırı yöntemlerinden biridir. Bu saldırı, RSA yordamında bahsedilen, $c = m^e \bmod n$ ve $m = c^d \bmod n$ eşitliklerini kullanan ve akıllı kart gibi cihazlarla işlem yapan donanıma karşı yapılmaktadır. Saldırıda donanımın çevreye yaydığı radyasyon, güç harcaması ve/veya kullanılan (geçen) işlemci zamanı incelenir. Ç: Çarpma, K: Kare Alma olmak üzere, $(((((((a)^2)^2a)^2)^2a)^2)^2a)^2)a$ işlemi K.K.Ç.K.K.Ç.K.K.Ç.K.Ç olduğundan a^{171} sayısının üs ifadesinin 171 olduğu bulunmaktadır. RSA yordamı açısından bakıldığında, kare alma ve çarpma işlemlerinin ne kadar kullanıldığı tespit edilerek e veya d özel anahtarı elde edilmektedir (Tilborg, 2000, s.180). Buna benzer saldırıların önlenebilmesi için, donanımın fiziki olarak koruma altına alınması gerekmektedir.

Güç Analizi Saldırısı ile Zamanlama Saldırısının haricinde Manyetik Alan (Magnetic Fields) ve Radyo Frekansı Yayılımına dayanan Yan-Kanal Saldırıları (Side-Channel Attacks) gibi saldırılar da vardır ancak, fazla etkin olmamalarından dolayı (Ferguson ve Schneier, 2003, s.151), bu tez kapsamında detaylı bir biçimde incelenmemiştir.

3. BULGULAR

3.1. Sivil Trafik Radarlarının Yer-Yer Hatlarından Hava Resmi Gönderilmesi

Günümüzde, askeri uçuşlarla beraber sivil uçuşların da sayısı da önemli ölçüde artmıştır. Gelişen ekonomilerin sağladığı refah düzeyi, kara ve deniz yolunun yanı sıra hava yolunun da yolcu ve kargo taşımacılığında kullanılmasına imkan tanımaktadır. Ancak, sivil uçaklarla yapılan ve tarihe 11 Eylül saldırıları olarak geçen, terörist eylemler, hava yollarındaki uçuşları bir süreliğine durdurmuş ve güvenliğin yeniden ele alınmasına neden olmuştur. Bu bağlamda, özellikle hava alanlarında yapılan kontroller artırılmış ve uçaklarda silahlı sivil güvenlik görevlilerinin bulundurulmasına kadar bir dizi yeni önlem alınmıştır.

Bu önlemlerle birlikte, sivil hava trafiğinin kontrol altında tutulması ve gelişen olaylara anında müdehale edilmesi, ayrı bir sorun olarak ele alınmaya başlanmış ve bunun için uçaklardaki ve yerdeki seyrüsefer kolaylıklarının ne gibi yeni nesil teçhizat ile yenilebileceği üzerinde durulmaya başlanmıştır.

Dünyadaki sivil havacılık alanındaki gelişmelerin paralelinde, Avrupa Birliği, gelecekte Türkiye'nin de dahil olacağı ve sivil hava trafiğini tüm Avrupa için tek merkezden kontrol edip yönetebileceği bir proje oluşturmuş ve çalışmalarını 11 Eylül saldırılarından sonra hızlandırmıştır. Yapılan çalışmalar ile, sivil hava trafik radarlarından gelen hava resimleri bir merkezde birleştirilecek ve tüm Avrupa Hava Sahasındaki hava yolları tek elden kontrol edilebilecektir. Projenin gerçekleşmesiyle beraber, maliyetlerin azaltılması ve ani durumlarda karar alma süresinin kısaltılması gibi birçok konuda büyük aşama kaydedilecek ve özellikle hava sahasındaki sivil uçuşlardan kaynaklanan güvenlik sorunlarına, zamanında müdehale edilerek büyük facialarının yaşanmasının önüne

geçilebilecektir. Ancak, bu gelişmelerin, kötü amaçlı kullanımı da beraberinde getirebileceği değerlendirilmektedir.

Sivil hava trafik radarlarının, kendi aralarındaki mevcut hava resmi alış – verişi açık hatlardan yapılmaktadır ve buna bağlı olarak, gelecekte kurulacak Avrupa Sivil Hava Trafik Kontrol Merkezi ile hava resmi alış – verişinin de açık hatlardan yapılacağı düşünülmektedir. Bu durum; terörist faaliyetler kapsamında ele alındığında, hatlardan gelen hava resimlerinin bir şekilde elde edilmesi sonucunda, sayıca daha fazla sivil uçakla, daha organize saldırı yapma olasılığını ortaya çıkarmaktadır. Bu olasılık; sivil hava trafik radarlarından merkeze gönderilen hava resminde, gerçek izlerin, normal hava yolunu takip ediyormuş izlenimini veren sahte izlerle değiştirilmesi ile birleştirildiğinde, ortaya çıkacak sorunun ciddiyetini daha da artırabilir. Merkeze gönderilen izlerin içinde sahte iz olması veya izlerin yanlış koordinatlarda gelmesi tüm hava trafiğinin sabote edilmesine neden olacak önemli güvenlik açıklarındandır. Bunlara benzer güvenlik açıklarının örneklerini artırmak mümkündür. Sivil hava trafiğini tehdit eden bu tip güvenlik açıklarının, radarlar ile merkezler arasındaki açık hatların şifrelenmesi ile kapatılabileceği değerlendirilmektedir.

Sivil hava trafik radarları, hava sahasındaki uçuşları, çevreye yaydığı radar enerjisinin, havadaki uçaklardan yansıyıp geri gelmesi prensibine göre tesbit etmektedir. Radar anteni, enerjisini 360 derece dönerek yaymaktadır. Dolasısıyla bir hava cisminin pozisyonu her turda bir defa tesbit edebilmektedir. Antenden gelen ham elektronik veriler, bilgisayarlar vasıtasıyla işlenmekte ve hava resmi oluşturulmaktadır. Oluşan hava resmi, açık hatlardan belli aralıklarla ilgili Hava Trafik Kontrol Merkezlerine aktarılmaktadır. Hava Trafik Kontrol Merkezleri, radarlardan gelen bu bilgilere dayanarak uçuşları takip etmekte ve hava sahasındaki trafiği uçuş emniyetini sağlayacak şekilde yönlendirmektedir.

Uçuş emniyeti, birçok parametreye bağlı olan ve her türlü havacılık faaliyetinde “olmazsa-olmaz” kelimeleri ile nitelendirilen önemli bir olgudur. Uçuş emniyetini etkileyen değiştirgeler içersinde, hava trafiğinin yönetilmesi önemli bir yer tutmaktadır. Hava trafiğinin yönetilmesi ise, radarlar tarafından

üretilen hava resimlerinin düzenli olarak kontrol merkezlerine gönderilmesine bağlıdır. Kontrol merkezlerinin ihtiyaç duyduğu hava resmi, sürekliliği olan bir veri trafiğinin tesis edilmesini öngörmektedir. Süreklilik gösteren veri akışı, gönderilen verilerin şifrelenmesi söz konusu olduğunda, gizli anahtarlı şifreleme tekniklerinin kullanılmasını gerektirmektedir.

Bu tezin uygulama safhasında, radarlardan kontrol merkezlerine açık hatlardan gönderilen hava resminin şifrelenmesi ile ilgili bir model geliştirilmesi üzerinde durulmuştur.

3.2. Problemin Çözümünde Kullanılan Yöntemin Açıklanması

Tez kapsamında hazırlanan program; karşılıklı kimlik tanıtma, şifreleme anahtarının gönderilmesi, şifreleme ve şifre çözme bölümlerinden oluşmaktadır. Esas işlemlerin şifreleme ve şifre çözme bölümlerinde yapılmasına karşın, ön hazırlık kapsamında karşılıklı olarak kimlik tanıtma işlemi yapılmakta ve esas şifreleme işlemini yapacak olan gizli anahtarlı şifreleme yordamının anahtarı karşı tarafa gönderilmektedir.

Bu işlemler yapılırken, güvenlikle ilgili oluşabilecek zayıf noktalara dikkat edilmeye çalışılmış ve literatür taramasından elde edilen bilgiler kapsamında, yapılan işlemlerin, bütünsel olarak emniyetli olması üzerinde durulmuştur. Ancak program içersinde kullanılan mantık çevrimi, birden fazla konunun uzmanı tarafından güvenlik açısından test edilip uygun görülmedikçe herhangi bir şifreleme işleminde kullanılmamalıdır. Herhangi bir şifreleme programının, güvenlik açısından test edilip onaylanması o programın en üst seviyede güvenli olduğunun varsayılması anlamına gelmektedir. Ancak bu varsayım, programın %100 güvenli olduğunu göstermez. İlgili programı veya yordamı kıran veya kırabilecek bir kişi olabilir ve bunu halka açık bir şekilde duyurmayabilir. Bu yaklaşımın doğruluğu, teknolojideki ve şifreleme yordamlarındaki ilerlemelere paralel olarak daha önceden güvenli sayılan VŞS

gibi yordamların artık güvenli sayılmaması ile teyit edilmektedir. Ancak bu tez kapsamında hazırlanan program, finansal uygulamalar gibi mutlak kesinlik isteyen uygulamalarda kullanılmasa bile, en azından, bir şekilde şifrelenerek gönderilmesi açık şekilde gönderilmesinden daha iyi olan verilerin karşı tarafa ulaştırılmasında kullanılabilir.

Bu amaçla hazırlanan programın içersinde iki konuda varsayımda bulunulmuştur. Bunlardan ilki, RSA yordamı ile kullanılacak özel anahtarların karşı tarafa Diffie – Hellman yordamıyla veya şifre mutemetliği tarafından ulaştırıldığı varsayımıdır. Diğeri de, karşılıklı olarak çalıştırılan iki programın verilerini TPC/IP ile aktardıkları varsayımıdır. TCP/IP ile ilgili varsayım, her iki tarafın da aşağıdaki kodu çalıştırarak ve karşılıklı olarak haberleşme kanalı oluşturduklarına dayanmaktadır.

```
Int32 port = 13000;

IPAddress* localAddr = IPAddress::Parse(S"XXX. XXX.XXX.XXX");

TcpListener* server = new TcpListener( localAddr,port);

server->Start();

TcpClient* client = server->AcceptTcpClient();

NetworkStream* stream = client->GetStream();
```

Buna ek olarak, TCP/IP ile ilgili varsayım her veri gönderiminde karşılıklı olarak aşağıda belirtilen yazma ve okuma nesnelerini kullanma mantığına dayanmaktadır.

```
StreamWriter SWriter = new StreamWriter (stream);

Byte SendData[] = Encoding::UTF8 -> GetBytes("Gönderilen Veri”);

int numBytesToWrite = (int) SWriter.Length;
```

```
int numBytesWrite = 0;

SReader.Write (SendData, numBytesWrite, numBytesToWrite);

StreamReader SReader = new StreamReader (stream);

Byte ReadData[];

int numBytesToRead = (int) SReader.Length;

int numBytesRead = 0;

SReader.Read(bytes,numBytesRead,numBytesToRead);
```

Uygulama safhasında, yukarıda çalıştırıldığı varsayılan program parçasının haricinde, gerçekte şifreleme yordamlarında bulunmayan görsel özellikler eklenmiş ve hazırlanan programın anlaşılabilir hale gelmesi ve sonuçlarının daha kolay görülebilmesi için ekran çıktıları kullanılmıştır.

3.2.1. Yeni Bir Model Önerisi Ve Bu Modelde Kullanılan Şifreleme Teknikleri

Tez kapsamında hazırlanan programda kullanılan teknikler, ABD MSTE tarafından onaylanan ve uluslar arası alanda kabul gören yordamlardan seçilmiştir.

İnternet üzerinde yapılan araştırmada, amatör programcıların yazdıkları son derece iyi tasarlanmış şifreleme yordamları olduğu görülmüş ancak, bu programların kullandıkları şifreleme anahtarlarının kolaylıkla başkalarınca elde edilebileceği fark edilmiştir. Şifrelemenin en önemli özelliğinin, başkalarınca elde edilmesi istenmeyen bilgilerin ve dolaylı olarak da bunları şifreleyen anahtarların koruma altına alınması olduğu bilindiğinden, bu tip bir yordam yazma yoluna gidilmemiş, bunun yerine güvenlik açıklarını uzman bir yaklaşımla kapatacağı düşünülen Microsoft'un kütüphane desteğinden faydalanılmıştır.

Bu bağlamda, MSDN kütüphane fonksiyonlarının kullanılması, tez kapsamında kullanılacak yordamlarının bütün kriterleri karşılayacak şekilde yeni baştan yazılmasını ortadan kaldırmış ve daha modüler program yazmaya imkan tanımıştır. MSDN fonksiyonlarının kullanılmaması durumunda, sadece RSA yordamının bir kısmının bile gerekli kriterlere göre yazılmasının, 1000 – 4000 ikillik sayıların birbirleri ile çarpılıp mod işlemine tabii tutulmasını içerdiğinden, ayrı bir araştırma konusu olarak ele alınmasını gerektirmektedir. Bunlardan dolayı ve tezin esas amacından ve sınırlarından uzaklaşılmaması ve karmaşık bir konu olan şifrelemenin anlaşılır bir tarzda incelenmesini sağlamak maksadıyla, program içersinde MSDN fonksiyonları kullanılmıştır.

MSDN kütüphane fonksiyonlarının kullanılmasının diğer bir nedeni de, kişisel seçim hatasını önlemek ve kullanıcıları adına, şimdiye kadar üretilmiş şifreleme fonksiyonlarını inceleyerek, içlerinden en uygun olanları gerekli elemelerden geçirip, seçmiş olan Microsoft firmasının tecrübelerinden faydalanmaktır.

3.2.1.1. RSA

Daha önceki paragraflarda nasıl çalıştığına dair bilgi verilen RSA yordamına, bu paragrafta, uygulama safhasında kullanılan teknikler arasına neden alındığının açıklanması amacıyla tekrar yer verilmiştir.

Bu yordam hali hazırda, dünya üzerinde en çok kulladıldığı düşünülen ve en çok bilinen Açık Anahtarlı Şifreleme yöntemidir. Hem Dijital İmza hem de Açık Anahtarlı Şifreleme yapabilme kabiliyetine sahip olup, bir çok kişiyi büyülemekte ve üzerinde çalışmaya sevk etmektedir. Yordamın güvenliği, büyük sayıların çarpanlarına ayrılma zorluğuna dayanmaktadır (Ferguson ve Schneier, 2003, s.223).

RSA yordamının ilginç bir özelliği, Açık anahtarı ve özel anahtar büyüklüklerinin artırılması sonucunda daha da güvenli hale gelmesidir. Bu durum, zaman içersinde RSA yordamının, mevcut anahtar büyüklüklerine göre çalışan

uygulamaları kırılsa bile, anahtar büyüklüklerinin artırılması sonucunda, geçerliliğini koruyabileceğini göstermektedir.

RSA yordamı, bütün dünya tarafından kullanılmakta ve ortaya konduğundan bu yana geçen zaman içersinde kırıldığına dair herhangi bir veri bulunmamaktadır. Bu nedenlerden dolayı uygulama programında kullanılmak üzere seçilmiş ve ilgili safhaların içersinde yer almıştır.

3.2.1.2. Dijital İmza

Kullanıcıların el ile attığı imza, kolaylıkla kopyalanabilmektedir ve güvenlik açısından, internet üzerinden yapılan işlemlerde kullanılamamaktadır. Bunun yerine elektronik ortamda, karşıdaki kişinin doğru kişi olup olmadığını tespit etmeye yardımcı olan Dijital İmza yordamı kullanılmaya başlanmıştır.

Dijital İmza, özellikle bilgisunar üzerinden yapılan bankacılık işlemlerinden dolayı iyi bilinmekte ve hem “Microsoft Internet Explorer” hem de diğer tarayıcılar tarafından desteklenmektedir. Ancak bu tez kapsamında kullanılan dijital imza, arada dijital imza firmalarının hakemliğine ihtiyaç duyan imza türünde değildir. Karşılıklı olarak daha önceden mutabık kalınmış tanıtıcı verilere dayanan dijital imza şeklidir.

RSA yordamı, bir önceki bölümde güvenli bir yöntem olarak seçildiğinden, karşılıklı kimlik tanıtma işleminde dijital imza amacıyla da kullanılması uygun görülmüştür.

3.2.1.3. Güvenli Çırpı Yordamı (Secure Hash Algorithm - SHA)

Güvenli Çırpı Yordamı – GÇY, ABD MSTE tarafından 1993 yılında geliştirilmiş ve ABD Federal Veri İşleme Standardı – FVİŞ-180 (FISP-180) olarak kullanıma verilmiştir. 1995 yılında güncellenerek standart numarası 180-1 olarak değiştirilmiştir. Bu yeni çırpı tekniği GÇY-1 (SHA-1) olarak isimlendirilmiştir (Stallings, 2003a, s.357).

GÇY, 256, 384 ve 512 ikil büyüklükte çıktı verebilen bir çırpı yordamıdır ve 128, 196 ve 256 ikil anahtar büyüklükleri ile çalışan İŞS yordamı ile beraber kullanılmak üzere geliştirilmiştir (Schneier, 1996, s.89).

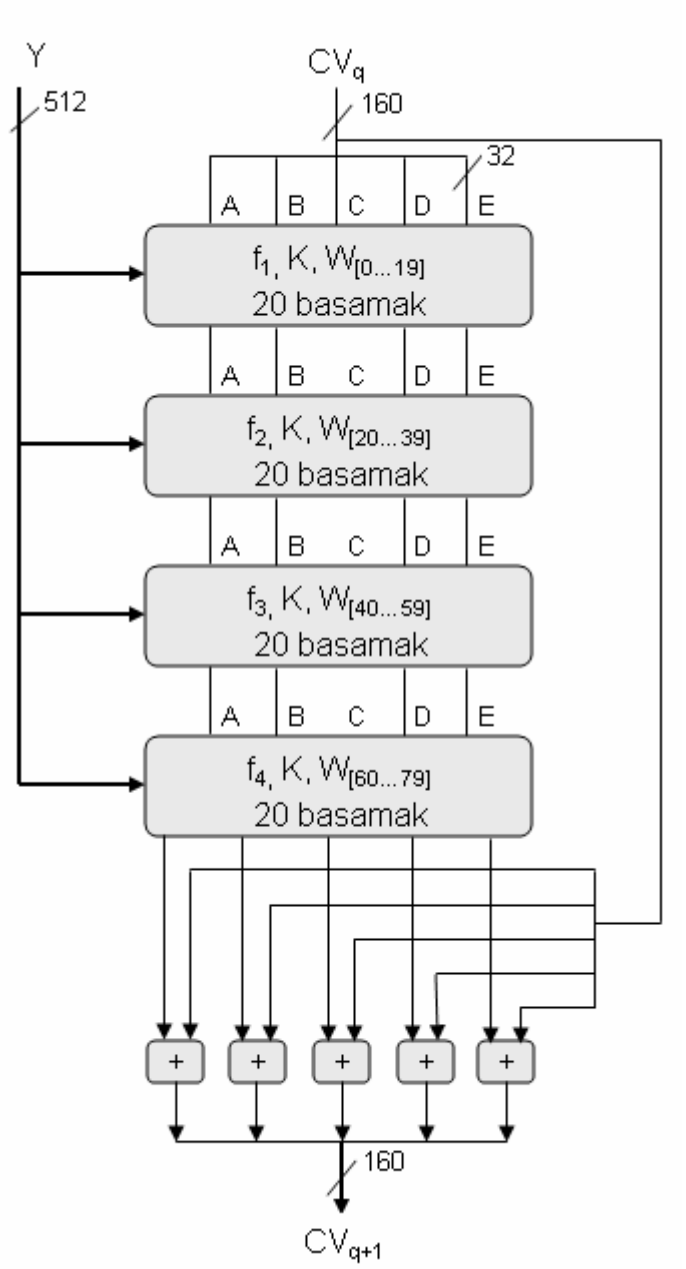
GÇY-1, 2^{64} ikil uzunluğa kadar mesajları, 512 ikillik bloklar halinde işleme tabi tutup 160 ikillik çıktı vermektedir. Mesajlar, uzunluk = 448 mod 512 olacak şekilde yastıklanmaktadır (padding). Bu işlemlerin sonucunda, her bloğun uzunluğu 512, son bloğun uzunluğu ise 448 ikil büyüklükte olacak şekilde ayarlanmaktadır. Elde edilen çıktıya, ikinci basamakta 64 ikillik ilk mesajın uzunluğu eklenmektedir. Daha sonra A,B,C,D ve E parçalarına bölünmüş, 5x32 = 160 sabit ikil başlangıç değerine sahip olan tampon (buffer) üretilmektedir. Sonraki dordüncü aşamada, alınan mesajlara her bir turu 20 basamaktan oluşan 4 farklı fonksiyonda, 512 ikil büyüklükte blok işlem yapılmaktadır (Stallings, 2003a, s.358). Şekil 3.1 dördüncü basamakta yapılan işlemleri göstermektedir.

Şekilde kullanılan K_{f1}, K_{f2}, K_{f3}, K_{f4} değerleri birbirlerinden farklı özel sabit değerlerdir. “+” işareti ile ifade edilen işlem ise mod 2^{32} işlemidir. Dördüncü basamağın sonunda, A,B,C,D ve E parçalarından oluşan CV_q giriş değeri ile son f_4 fonksiyonundan elde edilen değer, parçalar halinde ayrı ayrı toplanarak mod 2^{32} işlemine tabi tutulmakta ve 160 ikil büyüklüğünde CV_{q+1} değeri elde edilmektedir.

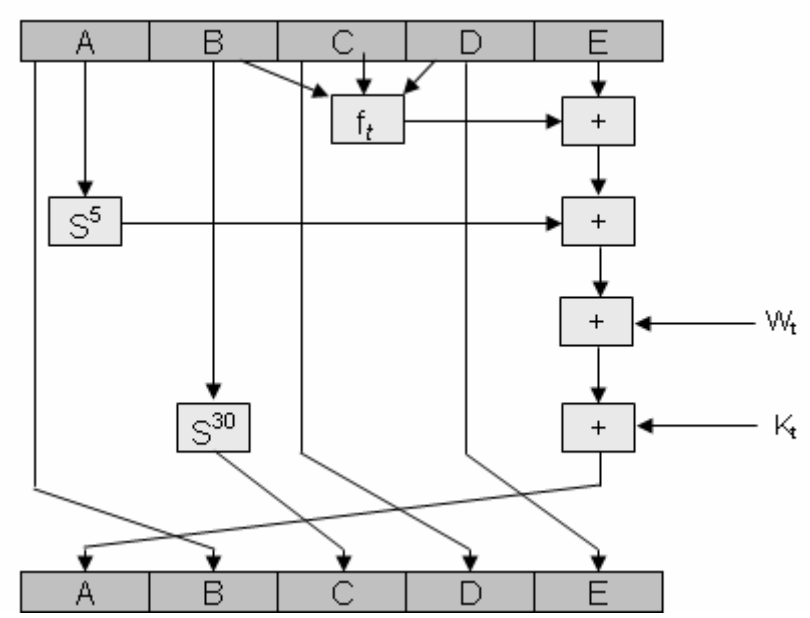
Bu işlem, L defa bütün mesajı oluşturan 512 ikil büyüklüğündeki bloklara uygulandığında, 160 ikil büyüklüğünde çırpılmış mesajlar elde edilmektedir. (Stallings, 2003a, s.360)

Şekil 3.1’deki (1) – (4) numaralı basamaklar daha detaylı bir şekilde Şekil 3.2’de gösterilmektedir.

S^k olarak ifade edilen işlem, 32 ikillik kelimenin sola doğru dairesel bir şekilde k ikil döndürülmesi işlemidir. Şekil 3.1 ve Şekil 3.2’de gösterilen f_t fonksiyonlarının yaptığı işlemler basamak numaralarına göre Çizelge 3.1’de ifade edilmektedir.



Şekil 3. 1. GÇY-1 yordamının 512 ikillik bir bloğu sıkıştırarak işlemesi (Stallings, 2003a, s.359)



Şekil 3. 2. Tek bir basamaktaki GÇY işlemleri (Stallings, 2003a, s.361)

Çizelge 3. 1. GÇY tarafından kullanılan fonksiyon ifadeleri (Stallings, 2003a, s.361)

Basamak	Fonksiyon İsmi	Fonksiyon Değeri
$(0 \leq t \leq 19)$	f1 = f(t, B, C, D)	$(B \wedge C) \vee (\overline{B} \wedge D)$
$(20 \leq t \leq 39)$	f2 = f(t, B, C, D)	$B \oplus C \oplus D$
$(30 \leq t \leq 59)$	f3 = f(t, B, C, D)	$(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$
$(40 \leq t \leq 79)$	f4 = f(t, B, C, D)	$B \oplus C \oplus D$

Yukarıda bahsedilen GÇY-1, ABD MSTE tarafından 180-2 numaralı standart olarak daha sonra güncellenmiş ve daha güvenli olduğu bildirilen GÇY-256, GÇY-384 ve GÇY-512 standartları ayrıca duyurulmuştur. Bu tez kapsamında hazırlanan programda ise, MSDN kütüphane fonksiyonlarının (MSDN Library Visual .NET 2003) sadece GÇY-1 ve Mesaj Özeti (MÖ5)'i desteklemesinden dolayı, GÇY-1 çırpısı kullanılmıştır. Ancak, GÇY-1 yordamı RSA modülünün içersinde kullanılmış ve RSA yordamı tek başına yeterli güvenliği sağladığından, programın ilgili aşamasında herhangi bir güvenlik açığı oluşmamıştır.

3.2.1.4. İŞS - Rijndael

Daha önceki paragraflarda nasıl çalıştığına dair bilgi verilen İŞS yordamına, bu paragrafta, uygulama safhasında kullanılan teknikler arasına neden alındığının açıklanması amacıyla, tekrar yer verilmiştir.

Rijndael yordamının kırılabilmesi için, 2^{120} adımlık işleme ve 2^{100} byte büyüklüğünde hafızaya ihtiyaç duyulduğu bildirilmektedir. Bu durum, günümüz teknolojisi ile bu yordamın kırılmasının mümkün olmadığını göstermektedir. Rijndael yordamının seçilmesindeki en önemli faktör, ABD MSTE tarafından yeni İŞS olarak duyurulmuş olmasıdır (Schneier, 1996, s.57-58). Yapılan çalışmada Rijndael yordamının seçilmesindee, standart bir yordamın seçilmesinin, kişisel seçim hatalarını ortadan kaldıracağı fikri etkili olmuştur.

İŞS – Rijndael yordamı, akım şeklinde süreklilik arzeden şifreleme işlemlerinde, Açık Anahtarlı Şifreleme yöntemlerine göre daha hızlı çalışmakta ve bir defada daha büyük veri gönderebildiği için daha avantajlı bir yöntem olarak görülmektedir. Bu özellik, diğer bütün gizli anahtarlı şifreleme yöntemleri için geçerlidir ancak, İŞS – Rijndael yordamı, gizli anahtarlı şifreleme bölümünde bahsedilen üstünlüklerinden dolayı ve uluslararası düzeyde kabul görmesinden ötürü seçilmiştir.

3.2.2. Kaynak Kodun İncelenmesi

Tez kapsamında hazırlanan programda, şifreleme için gerekli olan basamaklar, en az güvenlik açığı oluşturacak şekilde kaynak kod haline dönüştürülmüştür. Gerçekte bu tip programlar görsel modül içermemektedirler ancak, yapılan işlemlerin daha iyi takip edilmesi ve verilerin daha iyi kontrol edilmesi amacıyla, girdileri ve çıktıları gösteren görsel mesaj pencereleri kullanılmıştır.

Program başla butonuna basıldıktan sonra sadece bir defa çalışmakta ve istenen veriyi şifreleyerek karşı tarafa göndermektedir. Karşı tarafın da alınan

mesajı açması ile tamamlanmakta ve yeniden başa dönmektedir. Başlama butonuna tekrar basıldığında bütün anahtarlar yeniden üretilmekte ve aynı işlemler tekrar edilmektedir.

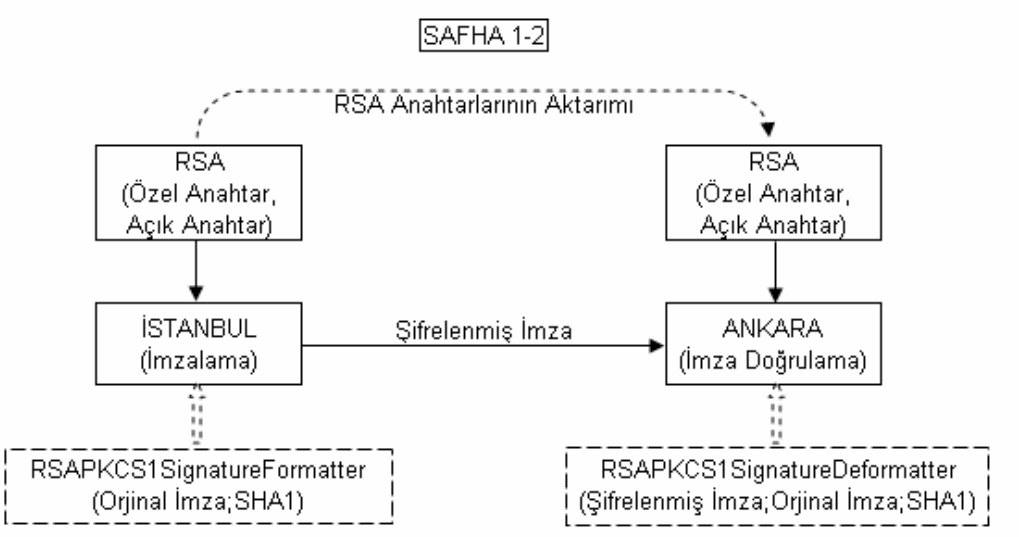
Kaynak kodun içinde Safha_1'e kadar olan kısım, ilgili döngünün kurularak başlatma butonuna basılması ile ilgilidir ve Microsoft Visual C++ araçlarının kullanılması sonucunda otomatik olarak üretilmiştir. Bundan dolayı, kaynak kodun incelenmesi bölümünde ayrıca açıklanmamıştır.

3.2.2.1. Karşılıklı Kimlik Tanıtma

Şifrelemenin ilk basamağı olan kimlik tanıtma, gönderilecek anahtar değerlerinin gerçek kişiye ulaştırılmasında temel teşkil etmektedir. Burada yapılacak bir hata, göndermek istediğimiz mesajların, gerçek kişi yerine hattı dinleyen üçüncü bir şahsa gitmesine neden olacaktır.

Programın ilgili safhasında, kimlik tanıtma işleminin güvenli olması için, RSA yordamından faydalanılmış ve ihtiyaç duyulan Dijital İmza üretilmiştir. Dijital İmza üretilirken kullanılan, GÇY-1 yordamı, imzaya esas teşkil eden metinin, her ihtimale karşı çırpılarak işleme girmesini sağlamıştır. Böylelikle, hattı dinleyen birinin imzayı çözmeye çalışırken, anlamlı bir metine ulaşıp ulaşmadığı gizlenmeye çalışılmıştır. Burada önemli olan bir nokta GÇY-1 yordamının güvenlik açısından risk taşımasıdır. Bu risk, çıktı olarak 160 ikillik bir veri göndermesidir. Herhangi bir çarpışma (collision) durumunda, bu büyüklükteki bir mesaj 2^{80} adımda kırılabilmektedir. 2^{80} adımın yeterli güvenlik sağlamamasının esas nedeni, gizli anahtarlı şifrelemede güvenli olaral kabul edilebilecek şifre anahtarı büyüklüğünün 128 ikilden başlamasıdır (Ferguson ve Schneier, 2003, s.88). Ancak GÇY-1 yordamı, açık anahtarlı RSA yordamı ile beraber kullanıldığı için, ilgili yordamdaki riskin varlığı dikkate alınmamıştır.

Kimlik tanıtma işlemi, programın Safha_1 ve Safha_2 aşamalarında gerçekleştirilmiş ve nasıl yapıldığı Şekil 3.3'te gösterilmiştir.

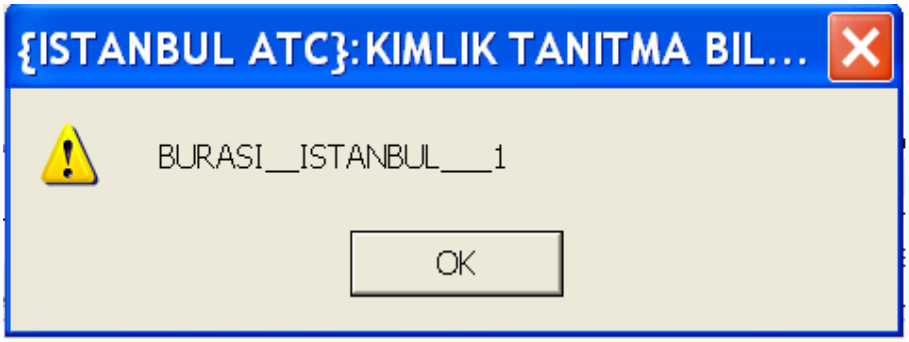


Şekil 3. 3. Safha_1 ve Safha_2'deki kimlik tanıtma işlemi

Kimlik tanıtma işleminin iki taraflı yapılması gerekmektedir. Safha_1'de İstanbul Hava Trafik Kontrol Merkezinden (ATC) kimlik tanıtma bilgisi, Ankara'ya gönderilmektedir. Burada kullanılan verilerin görsel olabilmesi için ASCII çevrimi kullanılmıştır. İstanbul'dan gönderilen kimlik tanıtma bilgisi (dataAuthenticationString) iki bölümden oluşmaktadır. Birinci bölüm, "BURASI__ISTANBUL___" verisidir ve esas kimlik bilgisini oluşturmaktadır. İkinci bölüm, sayaç bilgisidir ve esas kimlik bilgisi ile birleştirilmiştir. Sayaç oluşturma ve esas kimlik bilgisi ile birleştirme işlemi, hazır veri olarak girilmiştir. Sayaç verisi, normalde, imza karşı tarafta tamamen çözüldükten sonra, esas imzadan ayrılmalı ve bir artırılarak (veya uygun görülen başka bir matematiksel işlem yapılarak) karşı tarafa cevap olarak gönderilecek imzaya eklenmelidir (Ferguson ve Schneier, 2003, s.119-125). Bu durum, cevap olarak alınan imzanın gerçekten Ankara tarafından gönderildiğinin ikinci bir kontrolüdür. Aksi takdirde, karşı tarafın gönderilen imzayı doğru çözüp cevap verdiğinden emin olamayız. Arada hattı dinleyen üçüncü bir şahıs, İstanbul'dan gelen mesajı siler ve Ankara'ya kendi mesajını gönderebilir ve aynı işlemi Ankara'dan İstanbul'a gönderilen mesaj için de yapabilir. Başarılı olduğu takdirde, şifrelemede kullanılacak anahtarı elde eder ki, bu durum güvenliğin ortadan kalkması

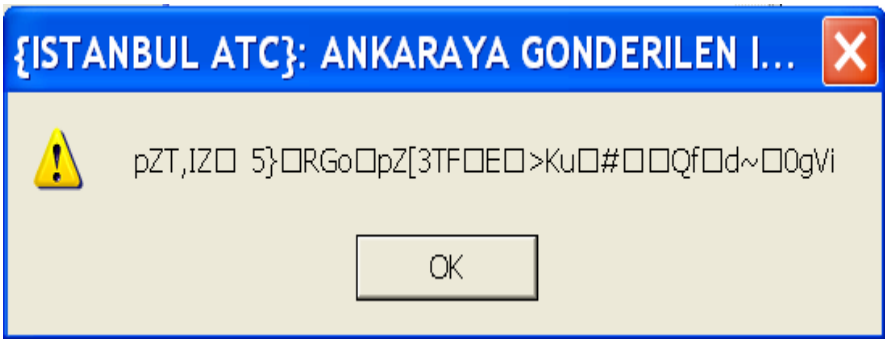
anlamına gelmektedir. Bu açığı kapatmak için, gerekli olan sayaç oluşturma ve esas kimlik bilgileri ile birleştirme işlemlerinin, İstanbul ve Ankara tarafından yapıldığı kabul edilmiş ve Safha_1 ile Safha_2'de gerekli olan veriler hazır değerler olarak girilmiştir.

Safha_1'deki program parçasının, kimlik tanıtma bilgisi olarak kullanılan veriyi gösteren ilk çıktısı, Şekil 3.4'te verilmiştir.



Şekil 3. 4. Safha_1'deki program parçasının ilk çıktısı

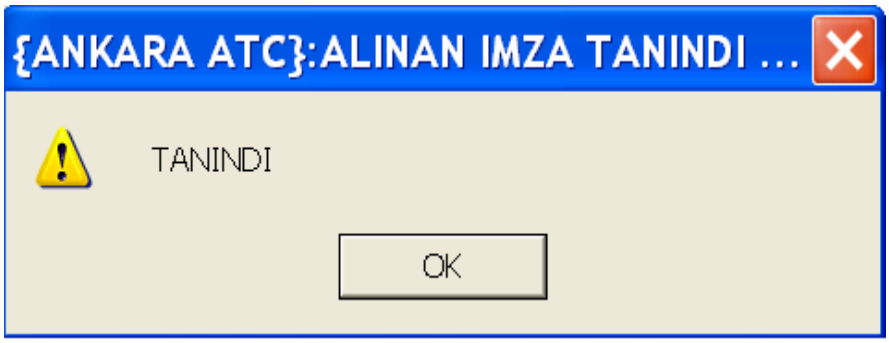
Safha_1'deki program parçasının, kimlik tanıtma bilgisine ait dijital imzayı gösteren şifrelenmiş ikinci çıktısı, Şekil 3.5'te verilmiştir. Şifrelenmiş ikinci çıktı, programın her çalıştırılışında farklı değerler almaktadır.



Şekil 3. 5. Safha_1'deki program parçasının ikinci çıktısı

İstanbul tarafından Ankara'ya gönderilen imza alındığında, olması gereken imza ile karşılaştırılmakta ve imzanın gerçek İstanbul tarafından

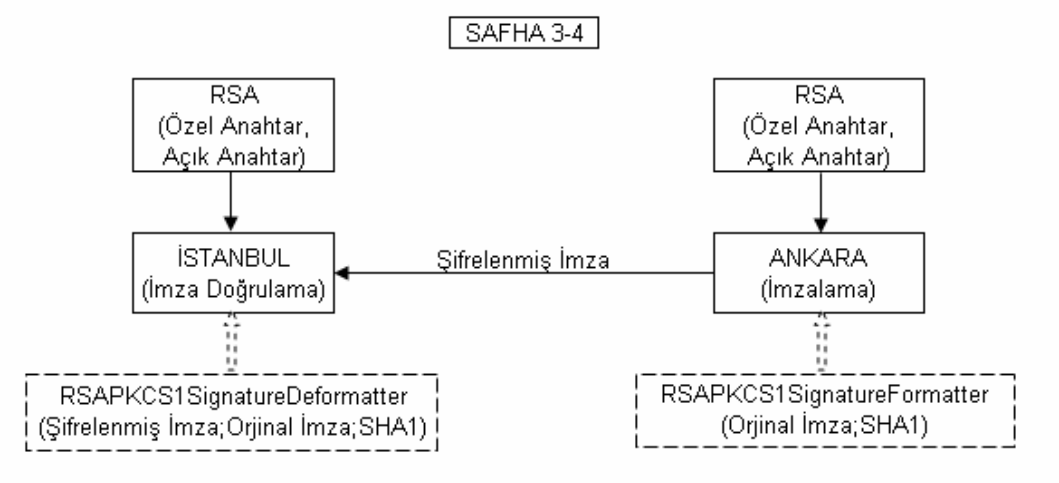
gönderilip gönderilmediği kontrol edilmektedir. Doğru olması durumunda, Safha_2'deki program parçası, Şekil 3.6'daki çıktıyı, yanlış olması durumunda da imzanın tanınmadığını ve programın durdurulması gerektiğini belirten mesajı yayınlamaktadır. Safha_2'de, İstanbul tarafından gönderilen imza ile birleştirilmiş olan sayaç değerinin ayrıldığı ve Ankara'dan İstanbul'a gönderilecek imzaya eklenmek üzere değerinin bir artırıldığı (veya uygun görülen başka bir matematiksel işlem yapıldığı) varsayılmıştır.



Şekil 3. 6. Safha_2'deki program parçasının çıktısı

Safha_3 ve Safha_4, aynı işlemlerin Ankara'dan İstanbul'a doğru yapıldığıyla ilgilidir. Şekil 3.7, bu yöndeki kimlik tanıtma işleminin nasıl yapıldığını göstermektedir. Üçüncü ve dördüncü safhalar, her iki tarafın da karşılıklı olarak birbirlerini tanımaları için şarttır. Aksi takdirde, kimlik tanıtma işlemi tek taraflı yapılmış olmakta ve karşılıklı bilgi alış verişinin başlatılması sakıncalı hale gelmektedir. Şekil 3.8 ve 3.9 Safha_3'teki çıktıları, Şekil 3.10 da Safha_4'teki çıktıyı göstermektedir.

Kimlik tanıtma işleminde kullanılabilecek başka bir yöntem de, esas kimlik bilgisine tarih ve zaman bilgisinin eklenmesidir. Bunun için belirli bir pencere değer (Örneğin 30 dk) sınır olarak kabul edilmekte ve şifreleme başlamadan önce her iki taraf da, sınır içersinde kalacak şekilde zamanlarını ayarlamaktadırlar (Stallings, 2003a, s.381-385). Ancak bu yöntem, matematiksel olarak hesaplanabilen sayaç değeri yöntemine göre daha az kesinlik taşıdığı için tercih edilmemiştir.



Şekil 3. 7. Safha 3 ve Safha_4'teki kimlik tanıtma

Şekil 3. 8. Safha_3'teki program parçasının ilk çıktısı

Şekil 3. 9. Safha_3'teki program parçasının ikinci çıktısı

Şekil 3. 10. Safha_4'teki program parçasının çıktısı

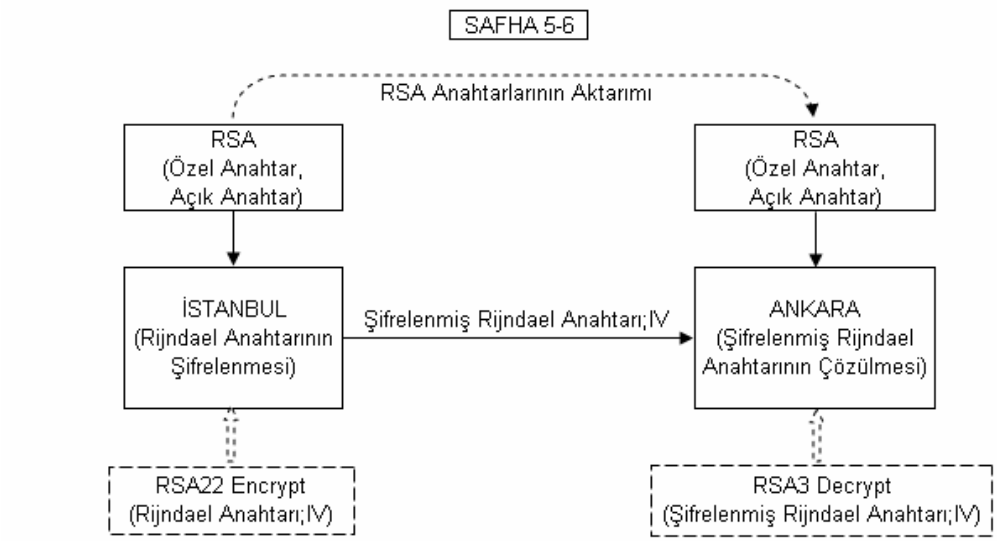
Kimlik tanıtma bölümleri olan Safha_1 ile Safha_2'de; RSACryptoServiceProvider* RSA = new RSACryptoServiceProvider() ve RSACryptoServiceProvider* RSA2 = new RSACryptoServiceProvider() işlemleri ile oluşturulan RSA ve RSA2 nesneleri farklı anahtar değerleri kullanmaktadır. Gerçek durumda bu anahtar değerlerinin, her iki tarafa karşılıklı işlemler sonucunda geçirilmesi veya anahtar mutemetliği vasıtasıyla başka bir yoldan temin edilmesi gerekmektedir. Bu durum, daha önceki bölümlerde bahsedilen Anahtar Dağıtımı Sorununu oluşturmaktadır. Anahtar dağıtımı başlı başına ayrı bir konu olduğundan, RSA ve RSA2 nesnelerine ait ilgili anahtarların, İstanbul ve Ankara arasında Diffie-Hellman yöntemi ile aktarıldığı varsayılmıştır.

Her iki tarafta da gerçekleştirilen imza alış-verişi işlemlerinde, şifrelenmiş olarak gelen mesaj çözüldüğünde, beklenen cevap ile karşılaştırılmakta ve aynı olup olmadığına dair, ekranda, görsel uyarı mesajı yayınlanmaktadır. Dijital imzadan elde edilen veri ile beklenen veri aynı değilse, programın kullanıcı tarafından durdurulması gerektiğini belirten ikaz yayınlanmaktadır. Gerçekte, böyle bir durumda, programın otomatik olarak durdurulması ve hattın muhtemel üçüncü bir şahıs tarafından dinleniyor olduğuna dair bir mesaj yayınlanması veya haberleşmede teknik bir arıza olduğuna dair bir ikaz yayınlanması gerekmektedir. Takiben de, kullanıcıya, programı bir süre sonra yeniden çalıştırması gerektiği bildirilmelidir.

3.2.2.2. Şifreleme Anahtarının Gönderilmesi

Şifreleme anahtarından kasıt, gizli anahtarlı şifrelemede kullanılacak olan İŞS – Rijndael anahtarıdır. Anahtarın gönderilmesinde, kimlik tanıtma safhasında olduğu gibi RSA yordamından yararlanılmıştır. RSA yordamının kullandığı, açık anahtar ve özel anahtar, Safha_5'te;

RSAParameters RSAKeyInfo22 = RSA22-> ExportParameters(true) işlemi ile ihraç edilmiştir. RSA yordamına ait açık anahtar ve özel anahtarın, Safha_5 ile Safha_6 arasında, Diffie – Hellman yordamıyla veya şifre mutemetliğiyle karşı tarafa aktarıldığı varsayılmıştır. İhraç ve ithal işlemleri bu varsayıma göre yapılmıştır. Sonuç olarak, gizli anahtarlı şifrelemede kullanılacak anahtarı şifreleyen açık anahtarlı yordamın, anahtarı, karşı tarafa sorunsuz bir şekilde ulaştırdığı kabul edilmiş ve şifrelenmiş olarak gelen İŞS – Rijndael yordamının gizli anahtarı çözülerek kullanılmıştır. Şekil 3.11'de bu işlemin nasıl yapıldığı gösterilmektedir.



Şekil 3. 11. Safha_5 ve Safha_6'daki Rijndael anahtarının aktarımı

Safha_5'in başında, Rijndael yordamı ile ilgili gizli anahtar ve vektör oluşturulmuştur. Vektör, gizli anahtar oluşturulabilmesi için gerekli olan rassal

sayı üretiminde kullanılmakta olup, olmaması halinde yordam, rassal bir sayı üretmek yerine her defasında aynı sabit sayıyı ürecetek şekilde işlem yapmaktadır. Bunun sonucunda da şifre anahtarı her defasında aynı değere sahip olmaktadır ki, bu durum istenmeyen bir güvenlik açığıdır. Vektör değeri bu açığı ortadan kaldırmak için kullanılmakta ve her defasında farklı değer alacak şekilde yenilenmektedir.

Safha_5'te bahsedilen diğer iki önemli nokta, gizli anahtarlı şifrelemenin hangi modda çalışacağının belirlenmesi ve yastıklamanın nasıl yapılacağına karar verilmesidir. Gizli anahtarlı şifreleme yordamları, istendiğinde 5 değişik modda çalışabilmekte ve bu modlar programa eklenebilmektedir. Bu modlardan ilki, Şifre Blok Zincirleme – ŞBZ modudur ve bir önceki bloğun şifrelenmiş metini ile sıradaki bloğun açık metininin "ayrıcalıklı veya" işlemine tabi tutulmasıyla oluşturulmaktadır. İkinci mod, Şifre Geri Besleme - ŞGB modudur ve her defasında açık metinin ilgili bloğunu byte vb. bölümlere ayırıp, şifreleme yapılması işlemidir. Üçüncü mod, Şifreli Metin Çalma – ŞMÇ modudur ve çıktı olarak elde edilen şifreli metinin uzunluğunun, başlangıçtaki açık metinin uzunluğu ile aynı olmasını sağlayan işlemlerden oluşmaktadır. Dördüncü mod, Elektronik Kod Kitabı – EKK modudur ve her bloğu ayrı ayrı şifreleyen ve aynı açık metin girdilerine karşılık aynı şifreli metin çıktılarını üretmeye yarayan işlemlerden oluşmaktadır. Beşinci mod, Çıktı Geri Besleme – ÇGB modudur ve kaydırma yazmaçlarının (shift register) doldurma tekniğindeki farklılık haricinde, ŞGB modu ile aynı mantıkta çalışan işlemlerden oluşmaktadır. Çalışma modlarının haricinde girilebilen başka bir değişken de yastıklama verisidir. Yastıklama tekniği olarak iki farklı teknik bulunmaktadır. Bu tekniklerden ilki sıfır ile yastıklama ikincisi de PKCS7 tekniği ile yastıklamaktır. Sıfır ile yastıklama basit bir işlemdir ve boş kalan kısımların sıfır ile doldurulmasını ifade etmektedir. PKCS7 tekniği ile yastıklama ise, yastıklama miktarı için gereken byte adedi ile yastıklama yapılmasına dayanan bir tekniktir. Örneğin, 24 ikil (3 byte) yastıklama yapılacaksa, yastıklama verisi "030303" değerlerinden oluşmaktadır (MSDN Library, 2003). RijndaelManaged sınıfında, çalışma modu

ve yastıklama tekniği değerlerini girmek mecburi olmayıp, farklı güvenlik önlemleri olarak şifreleme yordamına ayrıca dahil edilebilmektedir.

Programın Safha_6'daki kısmında, yeni bir RSA nesnesi oluşturulmuş, bu nesnenin ilgili anahtarları ithal edilmiş ve açık anahtarlı RSA yordamı ile şifrelenmiş biçimde alınmış olan gizli anahtarlı Rijndael anahtarı ve vektörü, açılarak DecryptedSymmetricKey ve DecryptedSymmetricIV değişkenlerine aktarılmıştır.

3.2.2.3. Şifreleme

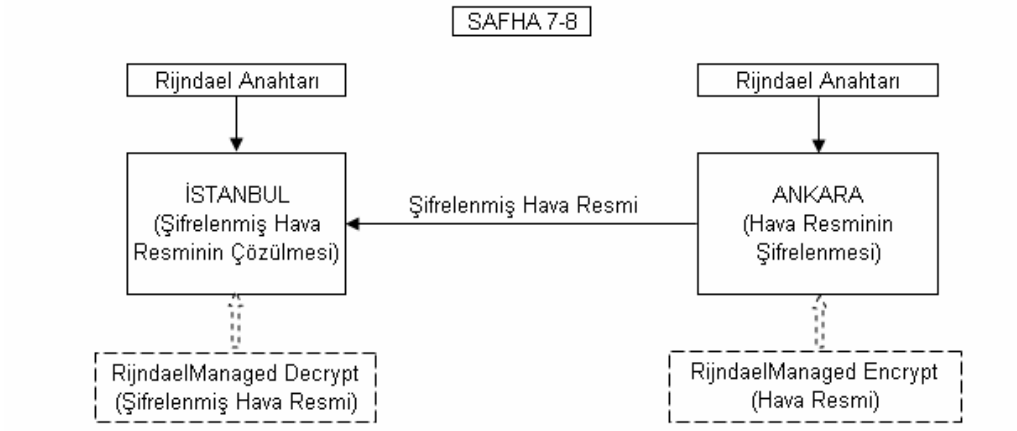
Sahfa_7'nin başında, şifrelemede kullanılacak olan Rijndael nesnesi oluşturulmuş ve ilgili değerleri bir önceki safhadan gelen değişkenlere eşitlenmiştir. Bu sayede, her iki tarafın da kullandığı şifreleme anahtarı ve vektörünün de aynı değerlerden oluşması sağlanmıştır.

Rijndael anahtarı, diğer gizli anahtarlı şifreleme yordamlarında olduğu gibi, üretildiği andan itibaren, herbir gizli anahtarlı şifreleme basamağı bir işlem olmak üzere, işlem adedi anahtar büyüklüğünün yarısına ulaşıncaya kadar kullanılabilmektedir. İşlem adedi anahtar büyüklüğünün yarısını geçtiğinde ise yeniden anahtar üretilmesi gerekmektedir. Örneğin, 256 ikil büyüklüğünde bir anahtar kullanıldığında, 2^{128} işlem sonra, güvenlik açısından anahtarın yeniden üretilmesine ihtiyaç duyulmaktadır.

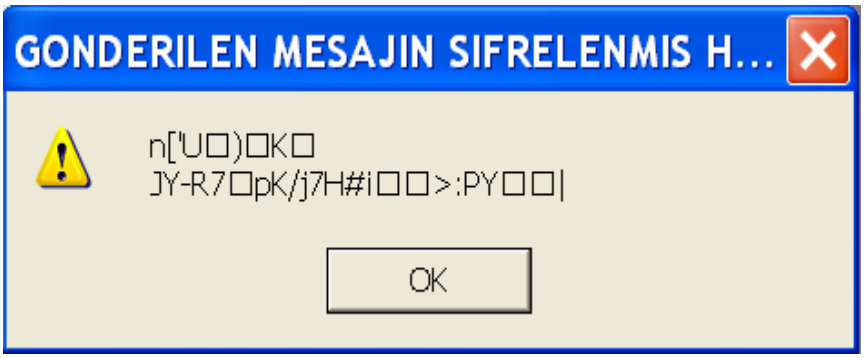
Bu durum, dijital imza safhasında olduğu gibi ayrı bir sayaçla kontrol edilebilmektedir. Sayaç değeri her gizli anahtarlı şifreleme işleminden sonra bir artırılmakta ve gönderilmek istenen açık metin ile birleştirilmektedir. Birleştirme işlemine, açık metinin uzunluk değeri de eklenmektedir. Bu işlem, alınan mesajın karşı taraf tarafından nasıl ayrılacağını belirtmektedir. Sayaç değerinden, başlangıçta üretilen anahtarın beklenenden daha fazla kullanılıp kullamadığını tespit etmekte faydalanıldığı gibi, gönderilen mesajların sıra numarasının ne olduğunun anlaşılmasında da faydalanılabilmektedir. Sayaç değeri, sıra numarası olarak kullanıldığında, kaybolan mesaj olup olmadığı hakkında bilgi elde

edilmekte ve/veya üçüncü bir kişinin gönderilen mesajlarla ilgili işlem yapıp yapmadığı anlaşılmaktadır (Ferguson ve Schneier, 2003, s.117-118). Bu kontrol, sayaç değeri çok büyük olduğundan ve test edilmesi günlerle ifade edilen zaman alacağından, program içersinde yapılmamış ancak, gerçek durumda yapılacağı varsayılmıştır.

Safha_7'nin en önemli kısmında, gönderilecek mesaj için, Rijndael yordamı ile birlikte kullanılmak üzere hafıza akımı ve şifre akımı oluşturulmaktadır. Daha sonra da esas işlem olan Rijndael yordamı ile şifreleme yapılmakta ve akıma dahil edilmektedir. Safha_7 ve Safha_8'deki şifreleme ve şifre çözme işleminin nasıl yapıldığı Şekil 3.12'de gösterilmiştir. Sonuç olarak elde edilen şifrelenmiş mesaj Şekil 3.13'te gösterilmiştir. Şifrelenmiş mesaj, kullanılan açık metin girdisine ve anahtara göre her defasında farklı değerler almaktadır. Şifrelenmiş mesajlar incelendiğinde, açık metine ait herhangi bir bölümü aynen içermediği ve açık metin girdisi bir anlam ifade etse bile, şifrelenmiş mesajların herhangi bir anlam ifade etmeyen verilerden oluştuğu görülmektedir. Bu açıdan bakıldığında, üçüncü şahıslar tarafından pratik bir zaman dilimi içinde çözülemeyecek karmaşıklıkta şifrelenmiş metin elde edildiği anlaşılmaktadır.



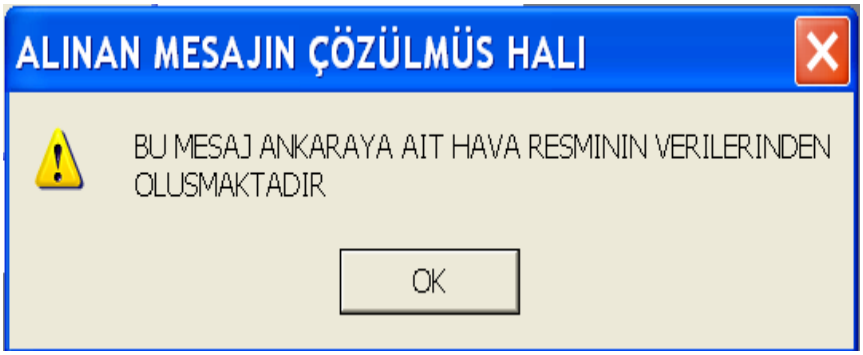
Şekil 3. 12. Safha_7 ve Safha_8'deki şifreleme ve şifre çözme



Şekil 3. 13. Safha_7'deki program parçasının ürettiği şifreli mesaj

3.2.2.4. Şifre Çözme

Safha_8'de, alınan mesajın şifresinin çözülerek, açık metinin elde edilebilmesi için, bir önceki safhada yapılan işlemlerin tersi yapılmaktadır. Bunun için, şifre çözmede kullanılacak hafıza ve şifre akımları oluşturulmakta ve alınan mesajın şifresi bu akımlar vasıtasıyla çözülmektedir. Daha sonra çözülen mesaj, bu akımlardan okunarak, Şekil 3.14'te görüldüğü gibi yayınlanmaktadır.



Şekil 3. 14. Safha_8'deki program parçasının çözdüğü mesaj

4. SONUÇ VE ÖNERİLER

Sivil Hava Trafik Radarlarının yer-yer hatlarından göderdikleri verilerin, istenmeyen kişilere karşı koruma altına alınması amacıyla yapılan bu çalışmanın sonucunda, verilerin gönderilmeden önce şifrelenmesi ve alındıktan sonra da şifrelerinin çözülmesine yönelik oluşturulan modelin başarıyla çalıştırılabileceği değerlendirilmiştir.

Tezin uygulama safhasında kullanılan İŞS-Rijndael yordamı, 128,192 ve 256 ikil büyüklüğündeki anahtarlarla çalıştırılabilmektedir. Bu anahtar büyüklüklerinden, istenilen hız durumuna göre, herhangi birinin kullanılabileceği sonucuna ulaşılmıştır.

Tezin uygulama safhasında hazırlanan programda, en çok zaman alan safhanın, anahtar üretimi ile ilgili program parçası olduğu gözlemlenmiş ve anahtar üretiminin ayrıca yapılması veya daha önceden üretilmiş hazır anahtarların kullanılması durumunda şifreleme programının çok daha hızlı çalıştırılabileceği anlaşılmıştır.

Hazır anahtar kullanılması durumunda, gizli anahtarlı şifrelemenin kullandığı anahtarın, RSA yordamı ile karşı tarafa gönderilmesi safhasına ve RSA yordamının kullandığı anahtarın, Diffie-Hellman yordamı ile karşı tarafa gönderilmesi safhasına ihtiyaç duyulmayacağından, uygulama programının daha da kısa hale getirilmesi mümkün olabilecektir.

Gönderme ve dolayısıyla şifreleme işleminin her 10 saniyede bir yapıldığı varsayıldığında, program tarafından 1 yılda 1.903.040 defa şifreleme işlemi gerçekleştirilmekte ve bu da 22 ikil büyüklüğünde bir sayıya tekabül etmektedir. Şifreleme ile ilgili saldırılara karşı alınması gereken güvenlik önlemleri göz önüne alındığında, bir anahtar, ikil büyüklüğünün yarısına kadar (anahtar 256 ikil büyüklüğünde ise 2^{128} defa) kullanılabilmekte ve 2^{22} sayısı ile 128, 192 ve 256 ikil büyüklüğündeki İŞS anahtarları karşılaştırıldığında, yapılan

uygulamanın istenilen şartların çok daha üstünde bir güvenlik sağladığı anlaşılmaktadır. Anahtarın yılda bir defa değiştirilmesi yerine, 3 ayda veya ayda bir defa değiştirilmesi durumunda ise güvenlik sınırının daha da yukarıya çekilebileceği değerlendirilmektedir.

Bu tez çalışması; artan bilgisayar hızlarından dolayı, donanım tabanlı şifreleme ile yazılım tabanlı şifreleme arasındaki hız farkının, düşük bant genişliklerinde gönderme yapan uygulamalarda, göz ardı edilebileceğini göstermiştir.

Yapılan tez çalışması ile; radarların yer-yer hatlarından Sivil Hava Trafik Kontrol Merkezlerine hava resmi aktarımı işlemlerinde, yazılım tabanlı şifrelemenin rahatlıkla kullanılabileceği sonucuna ulaşılmıştır.

KAYNAKLAR

Alfeld, P. (1998), *The Sieve of Eratosthenes*, http://www.math.utah.edu/~pa/Eratosthenes.html.

Anderson, R., Biham, E. ve Knudsen, L. (1997), *Serpent: A Proposal for the Advanced Encryption Standard*, http://kremlinencrypt.com/algorithms.htm#Serpent.

Atay, S. (2005), *Eliptik Eğri Tabanlı Kriptografik Protokol Ve Akıllı Kart Üzerinde Bir Uygulama*, http://www.certicom.com/index.php?action=res,cc_index.

Bishop, M. (2003), Computer Security Art and Science,Pearson Education Inc., Boston, A.B.D.

Cosgrave, J. (2004), *Fermat's Little Theorem*, http://www.spd.dcu.ie/johnbcos/fermat's_little_theorem.htm.

Ferguson, N. ve Schneier, B. (2003), *Practical Cryptography*, Wiley Publishing Inc., Indianapolis, Indiana, A.B.D.

Kara, İ. (2000), *Olasılık*, Bilim Teknik Yayınevi, Ankara

Kaufman, C., Perlman, N. ve Speciner, M.(2002), *Network Security Private Communication in a Public World*, Prentice Hall PTR Pearson Education Inc., Upper Saddle River, New Jersey, A.B.D.

Knuth, D.E. (1969), *The art Of Computer Programming Vol.2*, Addison Wesley Publishing, Massachusetts, A.B.D.

Menezes, A., Oorschot, P.v. ve Vanstone, S. (1996), *Handbook of Applied Cryptography*, CRC Press, Boca Raton, Florida, A.B.D..

MSDN Library Visual Studio.NET 2003 Help Menu, CipherMode Enumaration, Padding Mode Enumaration, Microsoft Corporation.

Nechvatal, J., Barker, E., Bassham, L., Burr, W., Dworkin, M., Foti, J. ve Roback, E., (2000), *Report on the Development of the Advanced Encryption Standard*, http://csrc.nist.gov/CryptoToolkit/aes/round2/r2report.pdf.

O'connor, J.J. ve Robertson, E.F. (2005), *Prime Numbers*, http://www-groups.dcs.st-and.ac.uk/~history/HistTopics/Prime_numbers.html.

Schneier, B. (1996), *Applied Cryptography Second Edition: protocols, algorithms, and source code in C*, John Wiley & Sons. Inc., New York, A.B.D.

Seberry, J. ve Pieprzyk, J. (1989), *Cryptography: An Introduction to Computer Security*, Prentice Hall of Australia Pty Ltd., Sydney, Australia.

Shparlinski, I. (1999), *Number Theoretic Methods in Cryptography Complexity lower bounds*, Birkhauser Verlag, Basel, İsviçre.

Stallings, W. (1995), *Network And Internetwork Security Principles and Practice*, Prentice Hall , Upper Saddle River, New Jersey, A.B.D.

Stallings, W. (2003a), *Cryptograghy And Netwrok Security Principles And Practice, Third Edition*, Prentice Hall Pearson Education International, Upper Saddle River, New Jersey, A.B.D.

Stallings, W. (2003b), *Network Security Essentials Applications And Standards, Second Edition*, Prentice Hall Pearson Education International, Upper Saddle River, New Jersey, A.B.D.

Tilborg, H.C.A v. (2000), *Fundamentals of Cryptology A Professional Reference and Interactive Tutorial*, Kluwer Academic Publishers, Norwell, Massachusetts, A.B.D.

Turgeon, G. (2003), *PowerBasic Crypto Archives: Twofish*, http://www.pbcrypto.com/view.php?algorithm=twofish.

Uysal, Ö., Şenocak, M. ve Süt, N. (2005), *Seçkin Türk Tıp Dergilerindeki Randomize Kontrollü Araştırmaların Yöntembilimsel Kalitesi*, VIII. Ulusal Biyosistatistik Tıp Kongresi Sözlü Bildirisi, http://bio2005.uludag.edu.tr/files/kitap-3.pdf.

Vanstone, S. (2004), *Public-Key Cryptography: Where is it Going?*, Code and Cipher Vol.1 No.3, http://www.certicom.com/index.php?action=res,cc_index.

Weisstein, E.W., "Primorial Prime." From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/PrimorialPrime.html.

Welschenbach, M. (2001), *Cryptography in C and C++*, Springer-Verlag Apress CA, New York, A.B.D.

Welsh, D. (1988), *Codes and Cryptogarphy*, Oxford University Press, Oxford, İngiltere.

Ek Programın Kaynak Kodu

```
// SAHFA_0
#pragma once

namespace DENEME_6
{
    using namespace System;
    using namespace IO;
    using namespace Text;
    using namespace System::ComponentModel;
    using namespace System::Collections;
    using namespace System::Windows::Forms;
    using namespace System::Data;
    using namespace System::Drawing;
    using namespace System::Security::Cryptography;
    //using namespace System::Object;
    using namespace System::Net::Sockets;
    using namespace System::Net;
/// <summary>
/// Summary for Form1
///
/// WARNING: If you change the name of this class, you will need to change
/// the 'Resource File Name' property for the managed resource compiler tool
/// associated with all .resx files this class depends on.  Otherwise, the
/// designers will not be able to interact properly with localized
/// resources associated with this form.
/// </summary>

public __gc class Form1 : public System::Windows::Forms::Form
    {
      public:
    Form1(void)
            {
              InitializeComponent();
               //Setup();
            }

protected:
    void Dispose(Boolean disposing)
    {
      if (disposing && components)
        {
            components->Dispose();
```

```
        }
        __super::Dispose(disposing);
    }
private: System::Windows::Forms::FontDialog *  fontDialog1;
private: System::Windows::Forms::Button *  button1;
private: System::Windows::Forms::FontDialog *  fontDialog2;

private:
/// <summary>
/// Required designer variable.
/// </summary>

System::ComponentModel::Container * components;

/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>

void InitializeComponent(void)
    {
            this->fontDialog1 = new System::Windows::Forms::FontDialog();
            this->button1 = new System::Windows::Forms::Button();
            this->fontDialog2 = new System::Windows::Forms::FontDialog();
            this->SuspendLayout();
            this->button1->Location = System::Drawing::Point(128, 80);
            // button1
            this->button1->Name = S"button1";
            this->button1->TabIndex = 0;
            this->button1->Text = S"button1";
            this->button1->Click += new System::EventHandler(this,
            button1_Click);
            // Form1
            this->AutoScaleBaseSize = System::Drawing::Size(5, 13);
            this->ClientSize = System::Drawing::Size(292, 266);
            this->Controls->Add(this->button1);
            this->Name = S"Form1";
            this->Text = S"Form1";
            this->Load += new System::EventHandler(this, Form1_Load);
            this->ResumeLayout(false);
    }
private: System::Void
button1_Click(System::Object*sender,System::EventArgs*e)
{
////////////
// SAFHA_1
////////////
```

```
Byte Key[];
Byte IV[];
Byte cipherbytes[];
Byte fromEncrypt[];

//Yeni bir nesnenin yaratılması
RSACryptoServiceProvider.RSACryptoServiceProvider* RSA = new
RSACryptoServiceProvider();

ASCIIEncoding* ByteConverter = new ASCIIEncoding();

//İmzalanacak çırpı
String* dataAuthenticationString = "BURASI__ISTANBUL___1";

MessageBox::Show(dataAuthenticationString, S"{ISTANBUL ATC}:KIMLIK
TANITMA BILGISI", MessageBoxButtons::OK,
MessageBoxIcon::Exclamation);

Byte originalData[]= ByteConverter->GetBytes(dataAuthenticationString);

RSAPKCS1SignatureFormatter* RSAFormatter = new
RSAPKCS1SignatureFormatter(RSA);

//Çırpı yordamının SHA1 olarak ayarlanması
RSAFormatter->SetHashAlgorithm(S"SHA1");

//Çırpı değeri için imza oluşturulması
Byte SignedHash[] = RSAFormatter->CreateSignature(originalData);

String* message2 = ByteConverter->GetString(SignedHash);

MessageBox::Show(message2, S"{ISTANBUL ATC}: ANKARAYA
GONDERILEN IMZA", MessageBoxButtons::OK,
MessageBoxIcon::Exclamation);
////////////
// SAFHA_2
//////////
// RSAPKCS1SignatureDeformatter nesnesinin yaratılarak,
// RSACryptoServiceProvider nesnesine anahtar değerlerinin için aktarılması
RSAPKCS1SignatureDeformatter* RSADeformatter = new
RSAPKCS1SignatureDeformatter(RSA);

RSADeformatter->SetHashAlgorithm(S"SHA1");

//Çırpının doğrulanıp, sonuçlarının ekranda gösterilmesi
if (RSADeformatter->VerifySignature(originalData,SignedHash))
{
```

```
MessageBox::Show(S"TANINDI", S"{ANKARA ATC}:ALINAN IMZA
TANINDI MI?", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);
}
else
{
MessageBox::Show(S"!!__DIKKAT__!!__IMZA_TANINMADI__PROGRA
MI_DURDUR!!", S"{ANKARA ATC}:ALINAN IMZA TANINDI MI?",
MessageBoxButtons::OK, MessageBoxIcon::Exclamation);
}
////////////
// SAFHA_3
////////////
RSACryptoServiceProvider* RSA2 = new RSACryptoServiceProvider();
ASCIIEncoding* ByteConverter2 = new ASCIIEncoding();

//İmzalanacak çırpı
String* dataAuthenticationString2 = "BURASI___ANKARA____2";

Byte originalData2[]= ByteConverter2->GetBytes(dataAuthenticationString2);

// RSACryptoServiceProvider nesnesine anahtar değerlerinin aktarılması için
// RSAOPKCS1SignatureFormatter nesnesinin oluşturulması
RSAPKCS1SignatureFormatter* RSAFormatter2 = new
RSAPKCS1SignatureFormatter(RSA);

//Çırpı yordamının SHA1 olarak ayarlanması
RSAFormatter2->SetHashAlgorithm(S"SHA1");

//Çırpı değeri için imza oluşturulması
Byte SignedHash2[] = RSAFormatter2->CreateSignature(originalData2);

MessageBox::Show(dataAuthenticationString2, S"{ANKARA ATC}:KIMLIK
TANITMA BILGISI", MessageBoxButtons::OK,
MessageBoxIcon::Exclamation);

String* message22 = ByteConverter2->GetString(SignedHash2);
MessageBox::Show(message22, S"{ANKARA ATC}: ISTANBULA
GONDERILEN IMZA", MessageBoxButtons::OK,
MessageBoxIcon::Exclamation);
////////////
// SAFHA_4
////////////
// RSACryptoServiceProvider nesnesine anahtar değerlerinin aktarılması için
//RSAPKCS1SignatureDeformatter nesnesinin yaratılması
RSAPKCS1SignatureDeformatter* RSADeformatter2 = new
RSAPKCS1SignatureDeformatter(RSA);
```

```
RSADeformatter2->SetHashAlgorithm(S"SHA1");

// Çırpının doğrulanıp, sonuçlarının ekranda gösterilmesi
if (RSADeformatter2->VerifySignature(originalData2, SignedHash2))
{
MessageBox::Show(S"TANINDI", S"{ISTANBUL ATC}: IMZA TANINDI
MI?", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);
}
else
{
MessageBox::Show(S"!!__DIKKAT__IMZA TANINMADI__PROGRAMI
DURDUR__!!", S"{ISTANBUL ATC}: IMZA TANINDI MI?",
MessageBoxButtons::OK, MessageBoxIcon::Exclamation);
}
////////////
// SAFHA_5
////////////
RijndaelManaged* myRijndael = new RijndaelManaged();
myRijndael->GenerateIV();
myRijndael->GenerateKey();
Key = myRijndael->Key;
IV = myRijndael->IV;
myRijndael->Mode;
myRijndael->Padding;

//Gizli anahtarlı şifrelemeye ait anahtar değerlerinin tutan değişkenler
Byte EncryptedSymmetricKey[];
Byte EncryptedSymmetricIV[];

// Yeni bir RSACryptoServiceProvider nesnesinin yaratılması
RSACryptoServiceProvider* RSA22 = new RSACryptoServiceProvider();

// Yeni bir RSAParameters nesnesinin yaratılması.
RSAParameters RSAKeyInfo22 = RSA22->ExportParameters(true);

//Gizli anahtar ve vektörünün şifrelenmesi
EncryptedSymmetricKey = RSA22->Encrypt(Key, false);
EncryptedSymmetricIV = RSA22->Encrypt(IV, false);
////////////
// SAFHA_6
////////////
// EncryptedSymmetricKey ve EncryptedSymmetricIV değerlerinin, karşı
// tarafa, network bağlantısı üzerinden gönderildiği kabul edilmiştir.Bunun
// yanında ihraç edilen RSA anahtar çiftinden özel anahtarın, karşı tarafa, Diffie
// Hellman anahtar değişim yordamıyla gönderildiği varsayılmıştır.

//Yeni bir RSACryptoServiceProvider nesnesinin yaratılması
```

```
    RSACryptoServiceProvider* RSA3 = new RSACryptoServiceProvider();

    //RSA yordamına anahtar değerlerinin ithal edilmesi
    RSA3->ImportParameters(RSAKeyInfo22);

    Byte DecryptedSymmetricKey[];
    Byte DecryptedSymmetricIV[];

DecryptedSymmetricKey = RSA3->Decrypt(EncryptedSymmetricKey, false);
DecryptedSymmetricIV  = RSA3->Decrypt(EncryptedSymmetricIV , false);
////////////
// SAFHA_7
////////////
    RijndaelManaged* myRijndael2 = new RijndaelManaged();
    myRijndael2->Key = DecryptedSymmetricKey;
    myRijndael2->IV  = DecryptedSymmetricIV;
    myRijndael2->Mode;
    myRijndael2->Padding;

MemoryStream* ms = new MemoryStream();
CryptoStream* cs = new CryptoStream(ms, myRijndael->
CreateEncryptor(),CryptoStreamMode::Write);
Byte PlainBytes[] = Encoding::UTF8->GetBytes("BU MESAJ ANKARAYA AIT
HAVA RESMININ VERILERINDEN OLUSMAKTADIR");

cs->Write(PlainBytes, 0, PlainBytes->Length);
cs->FlushFinalBlock();

ASCIIEncoding* ByteConverter3 = new ASCIIEncoding();
cipherbytes = ms->ToArray();

String* cipherbytes3 = ByteConverter3->GetString(cipherbytes);

MessageBox::Show(cipherbytes3, S"GONDERILEN MESAJIN SIFRELENMIS
HALI", MessageBoxButtons::OK, MessageBoxIcon::Exclamation);
////////////
// SAFHA_8
////////////
// Bu safhada kullanılan şifrelenmiş metinin network üzerinden karşı tarafa
// gönderildiği varsayılmıştır.

MemoryStream* msDecrypt = new MemoryStream(cipherbytes);
CryptoStream* csDecrypt = new CryptoStream(msDecrypt, myRijndael2-
>CreateDecryptor(), CryptoStreamMode::Read);

fromEncrypt = new Byte[cipherbytes->Length];
```

```
// Verinin şifre akımından okunması
StreamReader* SReader = new StreamReader(csDecrypt);
String* Serhat = SReader->ReadToEnd();

MessageBox::Show(Serhat, S"ALINAN MESAJIN ÇÖZÜLMÜS HALI",
MessageBoxButtons::OK, MessageBoxIcon::Exclamation);
}

private: System::Void textBox1_TextChanged(System::Object *  sender,
System::EventArgs *  e)
{
}

private: System::Void Form1_Load(System::Object *  sender, System::EventArgs
*  e)
{
 }

};
}
```