AÇIK ANAHTAR KRİPTOGRAFİSİ İLE SAYISAL İMZA TASARIMI VE UYGULAMASI

Meryem KIRIMLI

YÜKSEK LİSANS TEZİ
ELEKTRONİK-BİLGİSAYAR EĞİTİMİ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OCAK 2007
ANKARA

Meryem KIRIMLI tarafından hazırlanan AÇIK ANAHTAR KRİPTOGRAFİSİ İLE SAYISAL İMZA TASARIMI VE UYGULAMASI adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

Yrd. Doç. Dr. O. Ayhan ERDEM
Tez Yöneticisi

Bu çalışma, jürimiz tarafından oy birliği ile Elektronik ve Bilgisayar Eğitimi Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir.

Başkan : Doç. Dr. M. Ali AKÇAYOL

Üye : Yrd. Doç. Dr. O. Ayhan ERDEM

Üye : Yrd. Doç. Dr. Mustafa BURUNKAYA

Tarih : 30/01/2007

Bu tez, Gazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygundur.

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Meryem KIRIMLI

AÇIK ANAHTAR KRİPTOGRAFİSİ İLE
SAYISAL İMZA TASARIMI VE UYGULAMASI
(Yüksek Lisans Tezi)

Meryem KIRIMLI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
Ocak 2007

ÖZET

Bu tezde, veri işlem güvenliğinde kullanılan şifreleme teknikleri ve sayısal imzalar hakkında genel bilgi verildikten sonra RSA şifreleme ve imzalama algoritması kullanılarak tasarlanmış olan Sağlık Ocağı Yönetim Sistemi (SOYS) uygulaması ele alınmıştır. SOYS programı ile web üzerinden il ya da ilçe merkezlerindeki tüm sağlık ocaklarına ait verilerin merkezi bir sunucuda toplanması sağlanmaktadır. Aynı zamanda yüksek risk oranına sahip olan bu verilerin kurum içi ya da kurum dışından gelebilecek saldırılara karşı korunması amaçlanmaktadır. Bu sebeple, geliştirilen java güvenlik yazılımı ile tüm verilerin sayısal imzalı olarak iletimi sağlanmakta ve bu şekilde hem kişilerin sayısal kimlikleri tespit edilerek kaynaklara erişim kontrol altına alınmakta hem de elektronik ortamda transfer edilen bu verilerin içeriklerinin korunması sağlanmaktadır.

Bilim Kodu : 702.3.006
Anahtar Kelimeler : Açık anahtar, Gizli anahtar, Sayısal İmza, Şifreleme, Şifre Çözme
Sayfa Adedi : 78
Tez Yöneticisi : Yrd. Doç. Dr. O. Ayhan ERDEM

DIGITAL SIGNATURE DESIGN AND APPLICATION WITH PUBLIC KEY CYRPTOGRAPY

(M.Sc. Thesis)

Meryem KIRIMLI

GAZİ UNIVERSITY

INSTITUTE OF SCIENCE AND TECHNOLOGY

January 2007

ABSTRACT

In this thesis, after being given a general information about cryptography methods and digital signatures, Village Clinic Control System (VCCS) which is projected with using RSA encryption and signature algorithm is taken up. Using this , Village Clinic Control System, datas, belonging to all village clinics in the cities and counties, is provided to gather with a central server through a web interface. At the same time, these datas, which have high risk, should be protected against probable attack from inside or outside of clinic. For this reason, all datas are sent with digital signature using a developed java security software. Therefore, reaching the sources is undercontroled with determining personal digital identity and also these data contents which are transfered on the electronic environment are protected.

Science Code : 702.3.006
Key Words : Public Key, Private Key, Digital Signature, Encryption Decryption
Page Number : 78
Adviser : Asisst. Prof. Dr. O. Ayhan ERDEM

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla ben yönlendiren Sayın danışmanım Yrd. Doç. Dr. O. Ayhan ERDEM'e ve manevi destekleriyle beni hiçbir zaman yalnız bırakmayan aileme teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklama
AES	Advanced Encryption Standard
DES	Data Encyption Standart
DSA	Digital Signature Algorithm
DSS	Digital Signature Standard
DPS	Digital Payment System
ECC	Eliptic Cilcium Criptograpyh
EDI	Electronic Data Interchange
EFT	Electronic Funds Transfer
ECDLP	Eliptic Curve Discrete Logarithm Problem
FSS	Fail-Stop Signature
IDEA	International Data Encyption Algorithm
ISS	Internet Security Services
MD	Message Digest
NIST	National Institute of Standart and Technology
RC	Rivest' s Cipher
RIPE	Race Integrity Primitives Evaluation
RSA	Rivest-Shamir-Adleman
SHA	Security Hash Algorithm
SHS	Security Hash Standart

Kısaltmalar	Açıklama
SOYS	Sağlık Ocağı Yönetim Sistemi
SSL	Secure Socket Layer
UEKAE	Ulusal Elektronik ve Kriptoloji Araştırma Enstitüsü
TCP/IP	Transport Control Protocol/Internet Protocol
www	World Wide Web

1. GİRİŞ

İnternet, bilgisayar ağlarından oluşan bir ağ sistemidir. Bilgisayar ağı ise, birden çok bilgisayarın, telefon hattı, özel hatlar ve uydular kullanılarak birbirine bağlanmasını ifade eder. Bu ağlar, giriş serbestliğine göre, kapalı ve açık ağlar olarak ikiye ayrılabilir. Kapalı bilgisayar ağlarında, sınırlı sayıda bilgisayar birbirine bağlanabilir ve ağa bağlanacak bilgisayarlar (kişiler) bellidir. Kapalı ağlarda güvenlik derecesi, açık ağlara göre daha yüksektir. Çünkü ağa erişim sınırlandırılmıştır. Kapalı ağ uygulamaları 1970'lerden beri kullanılmaktadır. Bu uygulamanın en belirgin örneklerini ise, EFT (Electronic Funds Transfer-Elektronik Fon Transferi) ve EDI (Electronic Data Interchange-Elektronik Veri Takası) oluşturmaktadır. Bankalar arasında, işletmeler arasında ve bankalarla işletmeler arasında yaygın olarak kullanılan bu sistemler, intranet ve extranet gibi kapalı ağlar üzerinden yapılmaktadır. Açık bilgisayar ağları, giriş kısıtlamasının veya yasağının olmadığı, isteyen herkesin serbestçe girebildiği bilgisayar ağlarıdır. Bu nedenle açık ağlarda güvenlik en büyük problemdir. Bunun en güzel örneğini ise internet oluşturmaktadır. İnternete bağlanmak için, herhangi bir ISS (Internet Services System-İnternet Servis Sağlayıcısı) den bağlantı numarası almak yeterlidir. Bilgisayarlar arasındaki iletişim, TCP/IP (Transport Control Protocol/Internet Protocol-Taşıma Kontrol Protokolü/İnternet Protokolü) olarak adlandırılan iletişim protokolü kullanılarak gerçekleştirilmektedir.

İnternet kullanımının ve elektronik ticaret uygulamalarının artmasının önündeki en önemli engellerden birisi de yine güvenlik sorunudur. Gerek kurumsal, gerekse bireysel kullanıcılar arasında yapılan araştırmalar da, güvenlik sorununun, internetin ticari amaçlı olarak yeterince kullanılmamasının başlıca nedenlerinden birisi olduğunu göstermektedir. İnternet kullanıcıları, kişisel ve finansal verilerinin yetkisiz kişilerce kullanılabileceğine ya da değiştirilebileceğine inandıkları takdirde, ticari amaçları için internet yerine geleneksel ticari araçları kullanmaya devam edeceklerdir.

İnternet üzerinden gerçekleştirilen faaliyetlerde karşılaşabilecek güvenlik sorunları;

- Yetkisiz kişilerin ağ kaynaklarına girmesi,
- Bilgi ve ağ kaynaklarına zarar verilmesi,
- Bilginin değiştirilmesi, silinmesi veya bilgiye yeni şeyler eklenmesi,
- Yetkisiz kişilere bilginin iletilmesi,
- Bilgi ve ağ kaynaklarının çalınması,
- Gönderilen veya alınan bilginin inkar edilmesi,
- Ağ hizmetlerinin kesilmesine ve bozulmasına neden olunması,
- Alınmayan veya gönderilmeyen bilgilerin alındığının veya gönderildiğinin iddia edilmesi şeklinde sıralanabilir

Ortaya çıkabilecek bu güvenlik sorunları ve bunlar için geliştirilen çözümler, yetkilendirme, internet güvenliği ve veri-işlem güvenliği olmak üzere üç kategoride sınıflandırılabilir.

Veri ve işlem güvenliği, ticari işlemlerde ve mesajlarda, elektronik para ve elektronik çek gibi çeşitli çevrimiçi (online) ödeme sistemlerinde gizliliği, bütünlüğü ve güvenliği sağlamak amacıyla çeşitli kriptoloji (şifreleme) yöntemlerinin ve araçlarının geliştirilmesidir.

Günümüzde veri ve işlem güvenliğinin sağlanmasında yaygın olarak kullanılan kriptoloji türleri, açık-anahtarlı kriptoloji ve tek/gizli anahtarlı kriptoloji olmak üzere iki çeşittir. Gizliliği ve güvenliği sağlamak amacıyla bu yöntemlerde kullanılan araçlar ise; sayısal (dijital) sertifikalar, sayısal-elektronik imzalar ve onay kurumlarıdır.

Sayısal imzaların kullanım alanı oldukça geniştir. Sayısal ortamda kimlik tespitinin şart olduğu her uygulama sayısal imzanın uygulama alanına girmektedir. Sayısal imzalar aynı zamanda veri bütünlüğünün test edilmesinde de yaygın olarak kullanılır. Yani hem gönderilen mesajların kime ait olduğunu hem de mesajın alıcıya doğru bir şekilde ulaşıp ulaşmadığını

belirler.

İnternet üzerinde kişilerin olduğu kadar programların da kimlik tespitine ihtiyaç duyulmaktadır. Bir yazılımın güvenli bir kaynaktan gelip gelmediğini belirlemek için sayısal imzalar kullanılmaktadır. Microsoft Inc. tarafından geliştirilen ActiveX teknolojisi bu tür bir uygulama içermektedir. ActiveX programları, geliştirici tarafından imzalanarak internet tarayıcıda (web browser) çalıştırılmadan önce isteğe göre bu imzalar kontrol edilmekte ve istenmeyen programların çalıştırılması engellenmektedir [1]. Aynı uygulama java appletlerinin imzalanması için de kullanılmaktadır. Bu şekilde hem yazılan programın kime ait olduğu hem de program içeriğinde herhangi bir değişiklik olup olmadığı tespit edilebilmektedir.

Sivil uygulama alanlarının yanında birçok askeri alanda da sayısal imzaların kullanımı söz konusudur. Askeri haberleşme sistemlerinin bazılarında sayısal imzalar ve diğer kriptoloji teknikleri kullanılarak güvenli bir altyapı kurulması sağlanmıştır. Şifreli telefon ağı bu uygulamalardan bir tanesidir. Bu sistemde bir akıllı kartta kaydedilen kişilere ait sertifikalar Diffie-Hellman anahtar takas protokolü kullanılarak değiş tokuş edilir ve sayısal imza doğrulaması yapılır.

Genel kullanımı internet üzerinde çalışan uygulama olan sayısal imzaların diğer uygulama alanı da kurum içi yazılımlarda kimlik tespitinin yapılmasının sağlanması ve kağıt üzerinden yürütülmekte olan birçok yazışma ve dokümantasyon işlemlerinin elektronik ortama taşınmasıdır.

İnternet üzerinden gönderilen ve alınan veri veya mesajların, yetkisiz kişiler tarafından okunmaması veya değiştirilmemesi için, şifreleme yöntemleri ile ilgili teknik ve yasal çalışmalar birçok ülke tarafından yapılmaktadır. Ülkemizde, kriptografi ile ilgili çalışmalar, TÜBİTAK'a bağlı olarak kurulan Ulusal Elektronik ve Kriptoloji Araştırma Enstitüsü (UEKAE) tarafından yapılmaktadır. UEKAE, ulusal bilgi güvenliği ve ileri elektronik teknolojileri alanlarında Türkiye'nin teknolojik bağımsızlığını sağlamaya katkıda bulunma ana hedefi doğrultusunda, temel ve uygulamalı araştırmalar yapmakta ve

ihtiyaç sahiplerine teknik destek sağlamaktadır.

Türkiye'de, elektronik ve sayısal imzayla ilgili çalışmalar, Dış Ticaret Müsteşarlığı'nın koordinasyonunda kurulan, Elektronik Ticaret Koordinasyon Kurulu (ETKK) tarafından başlatılmıştır. ETKK Hukuk Alt Çalışma Grubu, uluslararası uygulamaları ve AB mevzuatını göz önünde bulundurarak ve Türk hukuk sisteminin özelliklerini dikkate alarak "Elektronik Veri, Elektronik Sözleşme ve Elektronik İmza Kanunu Tasarısı" taslağını hazırlamış ve Başbakanlığa göndermiştir. Adalet Bakanlığı da elektronik imzanın düzenlenmesine ilişkin kanun taslağının hazırlanması için bir komisyon oluşturmuş ve oluşturulan komisyonun hazırladığı "Elektronik İmzanın Düzenlenmesi Hakkında Kanun Tasarısı" taslağı Başbakanlığa gönderilmiştir. Kasım ayında yapılan seçimler sonrasında 59. Hükümetin kurulması üzerine, Adalet Bakanlığı, hazırladığı Kanun Tasarısı taslağını yenileyerek Başbakanlığa tekrar göndermiş ve tasarı, TBMM'ye sevk edilmiştir. "Elektronik İmza Kanunu", 15 Ocak 2004 tarihinde TBMM'de kabul edilmiş ve 23 Ocak 2004 tarihinde 25353 sayılı Resmi Gazete'de yayımlanmıştır. Elektronik imzanın hukukî ve teknik yönleri ile kullanımına ilişkin esasları düzenleyen bu kanun, yayım tarihinden altı ay sonra yürürlüğe girmiştir [2].

Günümüzde internet ve diğer ağ sistemlerinde güvenliğin bu kadar önemli olması ve sayısal imza sistemlerinin de bu güvenliği sağlamada en etkin yöntemlerden biri haline gelmesi nedeniyle araştırılmak üzere bu tez konusu seçilmiş ve sayısal imza sistemleri ayrıntılı bir şekilde incelenerek bir uygulama programı gerçekleştirilmiştir.

Sağlık sektöründe kullanılan uygulama yazılımları ile sürekli olarak hasta ve kurumlara ilişkin bilgi ve veriler elektronik ortama aktarılmaktadır. Elektronik ortamda bulunan bilgilerin güvenliğinin yüksek risk oranına sahip olması, hasta ve kurumlara ait bu gizli ve özel bilgilerin güvenliğinin sağlanmasını gerekli kılmaktadır. Bu nedenle, tezin beraberinde geliştirilen güvenlik yazılımı sağlık ocakları için web tabanlı olarak geliştirilmiş bir uygulama

yazılımına entegre edilerek girilen hasta ve kuruma ait bilgilerin sayısal imza ile güvenliği sağlanmıştır.

Hazırlanan bu tez beş bölümden oluşmaktadır

Birinci bölüm giriş bölümü olup ikinci bölümde kripto sistemlerinde kullanılan temel tanımlara ve algoritmalara yer verilmiştir.

Üçüncü bölümde ise sayısal imza sistemleri ve kullanılan algoritmalar incelenmiştir. Ayrıca incelenen bu algoritmalar arasında karşılaştırmalara yer verilmiştir.

Dördüncü bölümde de yapılan incelemeler sonucunda uygun olan sayısal imza ve özetleme algoritmaları kullanılarak geliştirilen Sağlık Ocağı Yönetim Sistemi uygulama programına yer verilmiştir. Bu bölümde geliştirilen programın işleyişi ayrıntıları ile anlatılmıştır.

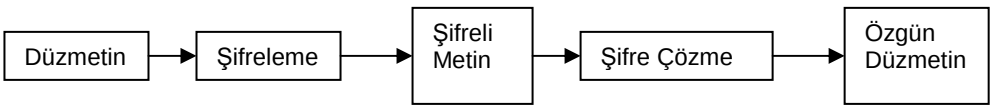
Beşinci bölüm olan sonuç bölümünde ise, hazırlanan tezin ve geliştirilen uygulama programının gelecekteki yapılacak çalışmalara ne gibi katkıları olabileceği ifade edilmiştir.

2. KRİPTO SİSTEMLER

Kriptografinin Türkçe adı şifre yazımdır. Kriptografi Yunanca gizli anlamına gelem "kript" ve yazı anlamına gelen “graf” dan türetilmiştir. Kriptoloji ise şifre bilimidir. Şifre kelimesi ise Fransızcadaki "chiffre" yani sayı kelimesinden gelmektedir. Kriptoloji, haberleşen iki veya daha fazla tarafın bilgi alışverişini emniyetli olarak yapmasını sağlayan, temeli matematiksel zor problemlere dayanan tekniklerin ve uygulamaların bütünüdür.

Kriptoanaliz ise; kriptografi sistemleri tarafından ortaya konan bir şifreleme sistemini inceleyerek zayıf ve kuvvetli yönlerini ortaya koymayı amaçlayan bir bilim dalıdır.

Bir göndericinin bir alıcıya açık ağlar üzerinden bir ileti göndermek istediği zaman, açık ağlardan gönderilen iletiler üçüncü şahıslar tarafından dinlenme ve değiştirilme tehdidi altındadırlar. Burada söz konusu ileti düz metindir (plaintext). Bir iletinin içeriğini saklamak üzere yapılan gizleme işlemi de şifrelemedir (encryption). Bu işlem düz metini şifreli metine dönüştürür. Bilginin içeriğinin başkalarının anlamayacağı hale gelir. Bu bilgi bir yere iletilmek amacıyla şifrelenen bir mesaj veya saklanmak amacıyla şifrelenen bir bilgi olabilir. Şifrelenmiş bir ileti şifreli metindir (ciphertext). Şifreli metini düz metine geri çevirme işlemi şifre çözümüdür (decrypt) [3]. Bu işlemler Şekil 2.1’de gösterilmektedir.



Şekil 2.1. Şifreleme ve şifre çözme blok diyagramı

Günümüzde elektronik güvenlik iki ana alanda ön plana çıkmaktadır. Bunlar, bilginin bir sistemden diğerine aktarıldığı andaki iletişim güvenliği ve bilginin bilgisayarda depolandığı süreç içerisindeki güvenliğidir. Her iki aşamada da güvenliği artırmak ve riskleri en aza indirmek için çeşitli kriptoloji yöntemleri uygulanmaktadır [4].

Modern kriptoloji algoritmaları kalem-kağıtla oluşturulan şifreli metinler kadar uzun değildir. Güçlü kriptoloji algoritmaları bilgisayar ya da özel donanım cihazları tarafından hesaplanabilecek şekilde tasarlanmıştır. Çoğu uygulamada bilgisayar yazılımının içerisinde şifreleme yapılmaktadır.

Kripto sistemleri üç ayrı bölümde kategorize etmek mümkündür.

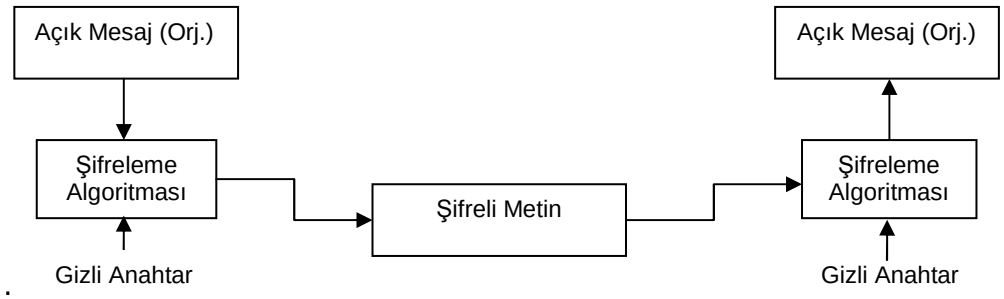
- *Açık metni şifreli metne çevirmekte kullanılan algoritma tipine göre:* Açık metindeki her elemanın başka bir elemana dönüştürülmesi ve açık metindeki elemanların yerlerinin değiştirilmesi şeklinde iki tür algoritma mevcuttur
- *Açık metnin işleniş yönetimine göre:* Blok ve Dizi şifreleme. Blok şifreleme sistemleri şifrelenecek olan veriyi bloklara ayırır ve her birim zamanda bir bloğu şifreler. Her bloğu şifreleme işlemi birbirinden bağımsız olarak gerçekleşir. Dizi şifreleme sistemler ise blok boyları bir olan blok dönüştürücüler olarak dönüştürülebilir. Dizi şifreleyicilerde her birim zamanda şifrelenecek olan verinin yalnızca bir biti şifrelenir, bu şifreleme işlemi bir xor kapısı ile gerçekleştirilir ve kullanılan anahtar uyarınca diğer bitler de sırasıyla şifrelenir [5].
- *Kullanılan anahtar sayısına göre:* Simetrik ve Asimetrik şifreleme

Sayısal imzanın dayanağı açık anahtarlı sistemler olduğundan bu tezde kripto sistemler kullanılan anahtar sayısına göre kategorize edilmiş şekilde incelenmiştir.

2.1. Simetrik Kripto Sistemler

Simetrik kripto sistemlerde şifreleme ve deşifreleme işlemlerinde aynı anahtar kullanılır ve bu anahtar simetrik anahtar veya gizli anahtar olarak da adlandırılmaktadır. Mesajlaşma yapılmadan önce anahtarlar iki taraf arasında ortak olarak belirlenir ve iletişime geçecek taraflar gizli anahtarlarını birbirlerine gönderirler

Şekil 2.2'de simetrik kripto sistemlerin çalışma şekli gösterilmiştir.



Şekil 2.2. Simetrik kripto sistemlerde mesaj alışverişi

Anahtar üretme ve birbirine güvenli bir şekilde iletme işlemi simetrik kripto sistemlerde üzerinde durulan en önemli konudur. Anahtar karşılıklı matematiksel ifadelerin takas edilmesi ile ortaklaşa oluşturulabildiği gibi, mesaj gönderen kişi tarafından oluşturulup asimetrik kripto sistemler yardımıyla şifrelenerek alıcıya iletilebilir. Kullanılan anahtarın gizli olmasından yola çıkılarak simetrik kripto sistemler için gizli anahtarlı kripto sistemler ismi de kullanılmaktadır.

Simetrik kripto sistemlerde kullanılan anahtar uzunlukları değişkenlik gösterir. Ne kadar uzun anahtar kullanılırsa o kadar güçlü bir şifreleme elde edilir. Ayrıca bu sistemler asimetrik sistemlere göre daha hızlıdırlar. Yeni algoritma üretiminin daha kolay olması sebebiyle asimetrik kripto sistemlere oranla daha çok gelişme göstermişlerdir [6].

Bilinen en meşhur simetrik algoritmalar DES (Data Encyption Standard-Veri Şifreleme Standardı), IDEA (International Data Encyption Algorithm- Uluslar arası Veri Şifreleme Algoritması), AES (Advanced Encryption Standard-Gelişmiş Şifreleme Standardı), RC2 (Rivest's Chipher-Rivest Şifresi) ve RC5 algoritmalarıdır.

2.1.1. DES algoritması

DES algoritması 1970'li yılların ortalarında NIST (National Institute of Standarts and Technology-Ulusal Standartlar ve Teknoloji Enstitüsü)

tarafından geliştirilmiştir. NIST tarafından zamanla standart haline getirilmiş ve dünya çapında birçok ülke tarafından benimsenmiştir [7].

DES 64 bit'lik bir blok şifrelemeli algoritmadır. Yani verileri 64 bit'lik bloklar halinde şifreler. Kullanılan anahtar ise 56 bit uzunluğundadır. Bu da algoritmanın ancak modern bilgisayar ve amaca yönelik donanımlarla yapılacak çok ayrıntılı anahtar arama işlemleri ile kırılabilmesine imkân tanır. DES hala çoğu şifre kırıcılar ve dış güçlere karşı oldukça güçlü bir algoritmadır ancak hükümetler, suç organizasyonları ve üst makamlarca kullanılan özel donanımlarla da kolaylıkla kırılabilmektedir. Genel olarak bakıldığında DES algoritmasının aşırı güvenlik gerektiren uygulamalar hariç diğer uygulamalarda rahatlıkla kullanılabileceği söylenebilir. Güvenlikle alakalı dikkat edilmesi gereken önemli bir husus zayıf anahtar kullanımından kaçınmaktır. DES için tespit edilen 4 zayıf, 16 yazı zayıf ve 53 muhtemel zayıf anahtar mevcuttur [7].

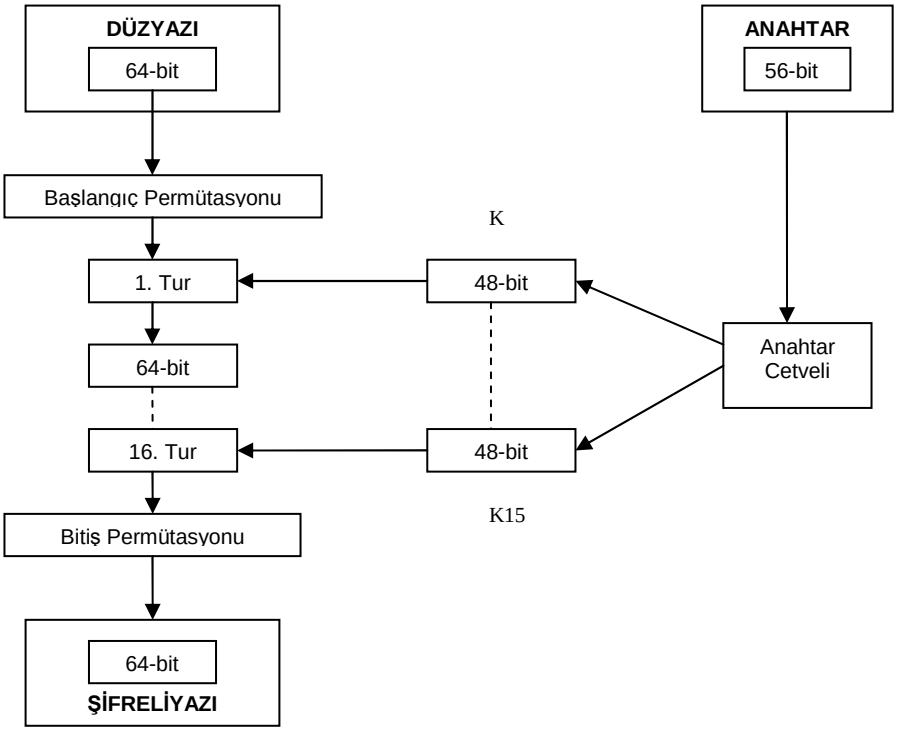
Kritik DES şifreleme işlemi S-kutuları (substitution boxes) kullanılarak yapılır. Doğrusal olmayan karmaşık işlemlerden oluşan S-kutuları DES içerisinde 4 satır ve 16 kolondan oluşan arama tabloları şeklinde tasarlanmıştır. S-kutuları bir anda 4 bit veri şifreleyebilir. Dolayısıyla 64 bit'lik bloğun şifrelenmesi için 16 döndürme işlemi gerekir. Tüm blok şifrelendikten sonra elde edilen sonuç tekrar permütasyon işlemine tabi tutulur. Bu işlemler Şekil 2.3.'de görülebilir.

DES bünyesinde bulunan tüm yerine koyma işlemleri, karmaşık XOR operasyonları ve permütasyonları şu iki özelliği sağlayacak şekilde seçilmişlerdir.

- Aynı algoritma hem şifreleme hem de deşifreleme için kullanılabilir.
- Basit operasyonlarda algoritma çok hızlı bir yapıya sahip olur.

DES algoritmasının hızı ile ilgili olarak RSA laboratuarlarında gerçekleştirilen araştırmalarda, DES'in RSA'ya oranla, yazılım şeklinde yapılan testlerde yüz

kat, donanım şeklindeki testlerde ise bin kat daha hızlı olduğu görülmüştür. Bunun sebebi DES içerisinde bulunan S-kutularının hızlı erişimli arama tabloları şeklinde tasarlanması RSA' da ise çok karmaşık bir tamsayı aritmetiğinin kullanılmasıdır [7].



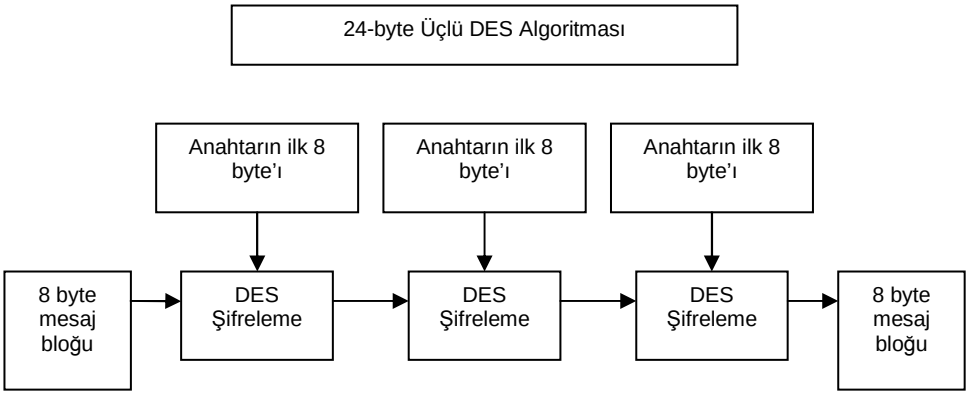
Şekil 2.3. DES algoritması [7]

2.1.2. Üçlü DES (3DES)

Simetrik şifreleme algoritmasıdır. DES'in 56 bit'lik anahtar uzunluğundan kaynaklanan güvenlik eksikliğini azaltmak için 3DES algoritması geliştirilmiştir. 3DES ile 168 (56x3) bit uzunluğunda kripto anahtarları kullanılır ve çalışma prensibi DES'e benzer. Şekil 2.4'de görüldüğü gibi 3DES ile şifrelemede DES algoritması birden çok anahtar ile arka arkaya uygulanır [8].

Normal DES algoritmasına göre 3 kat daha yavaş olan 3DES anahtar uzayını $2^{112}=1.65*10^{20}$ gibi oldukça büyük bir değere ulaştırmaktadır. Bunu yazabilmek için günümüz modern bilgisayarlarıyla bile yaklaşık 20 yıl sürecek bir anahtarlama hızına gerek vardır. Dolayısıyla 3DES oldukça güvenilir bir

algoritma olarak nitelendirilmektedir.

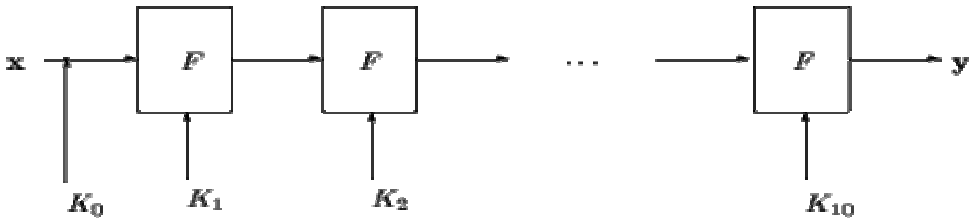


Şekil 2.4. 3DES [8]

2.1.3. AES algoritması

NIST tarafından 2000 yılı içerisinde şifreleme standardı olarak belirlenen AES, DES'e oranla daha büyük anahtar boyutu kullanmaktadır. Geliştiricileri olan Rijmen ve Daemen'in isimlerinden oluşan Rijndael, geleceğin gelişmiş şifreleme standartlarından biri olarak seçilmiştir.

Rijndael, bir blok şifrelemesinin, mesajların 128 bit'lik blok parçalarıyla şifrelenmesiyle olacağını ifade etmektedir. 128, 192 veya 256 bit anahtarlar kullanılan farklı seçenekler de bulunmaktadır [9].



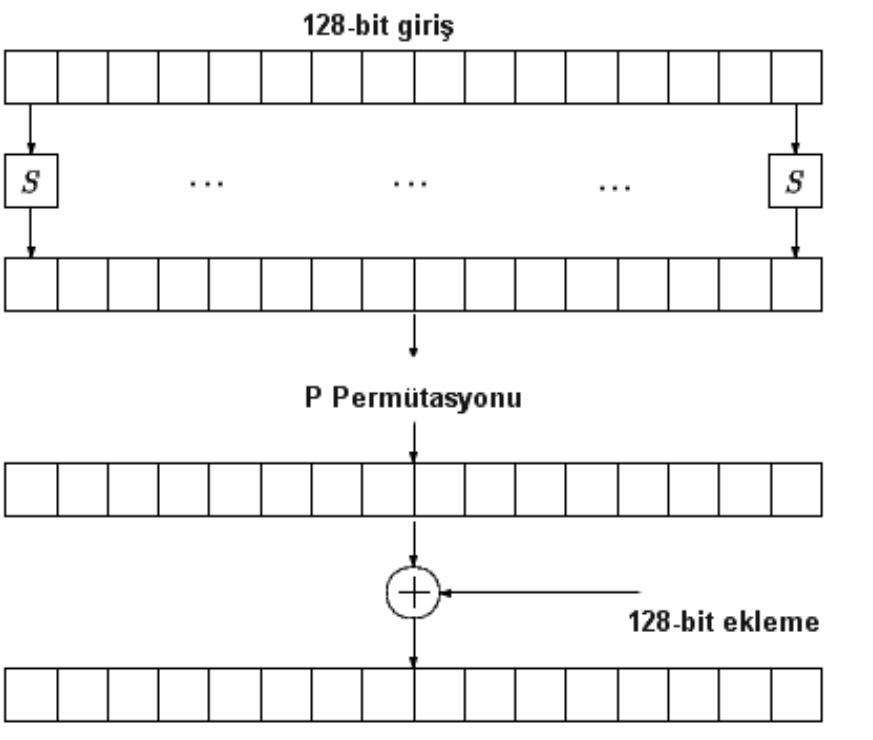
Şekil 2.5. AES tekrarlamaları [9]

AES' in işlemsel yolu Şekil 2.5' de gösterilmiştir. İlk olarak bir gizli anahtar K_0 bit seviyesinde mesaj ile XOR işlemine tabi tutulur. Daha sonra bütün blok şifrelere, F fonksiyonu K_0 anahtarından belirli bir yordam ile üretilen alt anahtarlar kullanılarak uygulanır.

AES için, F fonksiyonu 10 kez tekrarlanır.

• Şekil 2.6’da fonksiyonunun tekrarlanma süreci gösterilmektedir. 128 bit’lik blok, giriş olarak alınan 16 byte'ı alır. İlk olarak S (yerine koyma) her byte için uygulanır. Daha sonra 16 byte için ikinci bir P permutasyonu uygulanır. Anahtar genişletme yordamı ile oluşturulan i 128 bit’lik alt anahtar önceki sonuca bit seviyesinde eklenir.

• n. tekrarın K_i anahtarı, anahtar geliştirme yordamında (n-1). tekrarda oluşan $K_{(i-1)}$ alt anahtarı ile K_0 gizli anahtarının kullanımından elde edilir. Anahtar genişletme yordamında. $K_{(i-1)}$ anahtarının 16 baytı, 4 baytlık 4 parça halinde işleme alınırlar. Son 4 byte, S (yerine koyma) kullanılarak (F fonksiyonun uygulanan S) değiştirilir. Daha sonra, ilk 4 sonuç byte'ı alpha elementine eklenir. Bu element, önceden tanımlanan ve tekrar sayısına bağlı olan byte'dır. En son olarak, Ki 'yi elde etmek için 4 byte $K_{(i-1)}$'in ilk 4 byte'ına eklenir. Daha sonra sonuç diğer 4 byte' a eklenir.



Şekil 2.6. AES için F fonksiyonu uygulaması [9]

2.1.4. IDEA algoritması

IDEA, İsviçre'nin Zürih kentinde Xuejia Lai tarafından geliştirilmiştir. Anahtar uzunluğu 128 bit olan bu algoritma günümüzün en güvenilir şifreleme algoritmalarındandır. 8 döndürme işlemi içeren IDEA 64 bit blok şifreleme yapar. Şifreleme ve deşifreleme işlemleri DES algoritmasında olduğu gibi aynı algoritma kullanılarak gerçekleştirilir [7].

IDEA'ya yönelik çok yönlü atak denemeleri yapılmış ancak hiçbir aritmetiksel zayıflık tespit edilememiştir. Ayrıca algoritma diferansiyel kripto analiz ataklarına karşı da çok güvenli bir yapıya sahiptir. IDEA algoritmasının şu an için en büyük şansızlığı daha çok yeni olmasıdır.

IDEA hız olarak 3DES ve DES arası bir değerdedir. Yine IDEA'nın da 3IDEA sürümü mevcuttur.

2.1.5. RC2, RC4, RC5

RC2, DES'e alternatif bir metot olarak geliştirildi. Giriş ve Çıkış blokları 8 byte uzunluğundadır. RC2'nin düzgün çalışması için, giriş düzyazısı 8 byte'ın bir kaç katı olmalıdır. RC2 için giriş anahtarı 1 ile 1024 bit arası bir uzunlukta olabilir. Giriş anahtarı algoritma tarafından şifreleme işlemlerinde kullanılacak bir anahtar yaratmak için kullanılır. Şifreleme anahtarı bir bit'den 1024 bit'e kadar değerler alabilir. RC2'nin Amerika dışına ihracatı 40 ile 48 bit arası anahtar sınırlaması ile mümkündür [10].

RC4, 1987'de RSA Laboratuarları tarafından geliştirilen bir simetrik akış şifreleme metodudur. Pseudo-random numara sırasına dayalıdır. RC4'den elde edilen çıkış bilgi akışı ile XOR'lanır. Bu nedenle hiçbir RC4 anahtarı iki farklı bilgi akışını şifrelemek için kullanılmamalıdır. RC4, 1 ile 2048 bit arası uzunluklarda anahtarları destekler. Fakat Amerika dışına satılan yazılımlarda 40-bit sınırlaması uygulanır. 40 bit altındaki anahtarların RC4 ile kullanılması tavsiye edilmez. DES ile karşılaştırıldığında RC4 beş kat daha hızlıdır. Fakat geçmişte RC4 kırma girişimleri başarı ile sonuçlanmıştır. RC4'ü çalıştırmada

kullanılan kod DES'inkinin onda biri kadardır.

1994 yılında Ronald Rivest tarafından geliştirilen RC5 algoritması hızlı ve parametrik bir blok şifrelemeli simetrik algoritmadır. Anahtar uzunluğu, blok uzunluğu ve döndürme sayısı parametrik olarak ayarlanabilen RC5 değişik çapta güvenlik gerektiren uygulamalarda rahatlıkla uygulanabilir [10].

Tipik blok uzunluğu 32, 64 ve 128 bit, döndürme sayısı 0-255 arası bir değer, anahtar uzunluğu ise 0-2048 bit aralığında seçilebilmektedir. Algoritma anahtar genişletme, şifreleme ve deşifreleme olmak üzere üç ana bölümden oluşur.

Anahtar genişletme işleminde, kullanılan anahtar bir anahtar tablosunu doldurmak için kullanılır. Bu tablonun boyutu kullanılan döndürme parametresine göre değişir. Anahtar tablosu hem şifreleme hem de deşifreleme işlemlerinde kullanılır. Şifreleme işleminde üç temel operasyon vardır. Bunlar tamsayı ekleme, XOR ve değişken döndürme işlemleridir.

RC5 yayınlandığı günden günümüze kadar üzerinde birçok deneme testleri yapılmış ve algoritmanın kırılması için onlarca kripto analiz tekniği kullanılmıştır. Bütün bu araştırma sonunda RC5'in herhangi bir güvenlik problemine rastlanmamıştır.

2.2. Asimetrik Kripto Sistemler

Bir diğer kategori olan asimetrik kripto sistemlerdeki durum simetrik kripto sistemlere göre oldukça farklıdır. Bu sistemlerde, veri bir anahtarla şifrelenirken diğer bir anahtarla deşifrelenir. Böylelikle simetrik kripto sistemlerde gerekli olan anahtar takas işlemine gerek kalmamış olur [11].

Asimetrik kripto sistemlerde her kullanıcının iki adet anahtarı mevcuttur. Bu anahtarlardan birisi sadece kendisinin bildiği ve güvenli bir şekilde saklanması gereken gizli anahtar, diğeri ise herkesin kullanabilmesi için dağıtılan açık anahtardır. Mesaj gönderen kişi ya da parti mesajını karşı

tarafın açık anahtarıyla şifreleyerek gönderir. Bu mesaj ancak alıcının gizli anahtarıyla tekrar deşifre edilebilir. Şekil 2.7'de asimetrik sistemlerde mesaj alışverişinin nasıl gerçekleştiği görülmektedir.

Şekil 2.7. Asimetrik kripto sistemlerde mesaj alışverişi

Asimetrik kripto sistemlerin genel özellikleri şu şekildedir [12]:

- Açık anahtar (K_p, p:public) ve gizli anahtar (K_s, s:secret) çiftinin oluşturulması basit olmalı ve alıcı tarafından polinomiyal zaman içinde yapılabilmelidir.
- Şifreleme işlemi: $C=E_{\mathbf{Kp}}(M)$ olup, gönderici tarafından polinomiyal zamanda yapılabilmelidir.
- Alıcı tarafından yapılan deşifre işlemi, yine yalnız alıcı tarafından bilinen gizli anahtarla (K_s):$M= D_{Ks}(C)$ olarak ve polinomiyal zamanda gerçekleştirilebilmelidir.
- Düşman kriptanalisti açık anahtardan (K_p) giderek, gizli anahtarı (K_s) oluşturmaya kalktığında çözümsüz bir sorunla karşı karşıya gelmelidir.
- Düşman kriptanalist, açık anahtar ve kriptogram (K_p, C) ikilisinden hareketle açık metni (M) bulmaya çalıştığında, yine çözümsüz bir sorunla karşılaşmalıdır.

Asimetrik kripto sistemlerin gücü gizli anahtarın açık anahtardan türetilme ihtimalinin yok denecek kadar az olmasından kaynaklanmaktadır. Bu sistemlere açık anahtarlı kripto sistemler de denilmektedir.

Asimetrik kripto sistemler ilk defa Whitfield Diffie ve Martin Hellman tarafından 1975 yılında yayınlanan makalede tanıtılmıştır [13]. Diffie-Hellman anahtar takas algoritması, RSA (Rivest, Shamir and Adleman), ElGamal ve DSA (Digital Signaure Algorithm- Dijital İmza Algoritması) algoritmaları bilinen en meşhur asimetrik algoritmalardandır.

Çizelge 2.1'de simetrik ve asimetrik kripto sistemlerin bir tablo halinde karşılaştırılması görülmektedir.

Çizelge 2.1. Tek ve açık anahtarlı şifreleme yöntemlerinin karşılaştırılması

Özellikler	Tek/Gizli anahtar	Açık anahtar
Anahtarların Sayısı	Tek Anahtar	İki Anahtar
Anahtarların Türü	Anahtar Gizlidir	Bir anahtar gizli, diğeri açıktır
Anahtar Yönetimi	Basit fakat yönetimi zor	Dijital sertifikalara ve güvenilir üçüncü taraflara ihtiyaç vardır
Hız	Çok hızlı	Daha yavaş
Kullanımı	Büyük hacimli veri şifrelemesi için kullanılmaktadır	Küçük dokümanların şifrelenmesi veya mesajların imzalanması gibi az talep gören uygulamalarda kullanılır

2.2.1. Diffie-Hellman anahtar takas algoritması

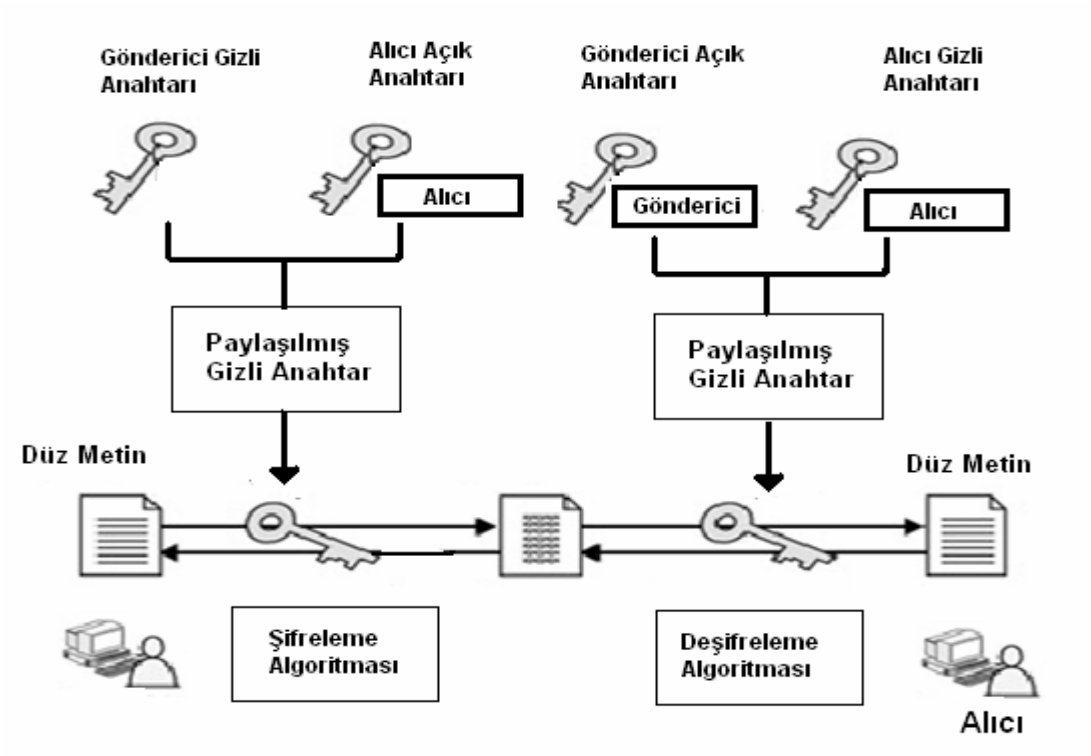
Diffie-Hellman anahtar takas algoritması yayınlanan ilk asimetrik algoritmadır. Simetrik kripto sistemlerde kullanılan gizli anahtarın güvenli bir şekilde takas edilmesi için geliştirilmiştir. Diffie-Hellman algoritması ayrık logaritma problemlerini çözmenin zorluğu üzerine kurulmuştur. Ayrık logaritma problemi modüler aritmetikte üst alma operasyonudur.

Diffie-Hellman anahtar takas algoritması birçok güvenlik yazılımında kullanılmaktadır. SSL (Secure Socket Layer- Güvenlik Soket Katmanı) bunlardan birisidir.

Her bir bağlantı, bir tanesi açık (public), diğeri özel (private) olmak üzere bir anahtar çifti alır. Aşağıda bu algoritmanın çalışma mantığı anlatılmıştır:

- Gönderen taraf istemcinin tüm bağlantılarında kullanacağı açık anahtarını öğrenir.
- Gönderen taraf özel anahtar ve istemcinin açık anahtarını birleştirerek bir hesaplama yapar ve bu hesabın sonucu olarak gizli bir anahtar saklanır.
- Mesaj bu ortak gizli şifre kullanarak şifrelenir.
- Şifrelenmiş mesaj istemciye gönderilir.
- Şifreli mesaj alındığında, istemci kendi özel anahtarı ve gönderen açık anahtarı ile bir hesaplama yapar gizli anahtarı üretir.

Bu işlemler Şekil 2.8'de gösterilmiştir.



Şekil 2.8. Diffie-Hellman algoritması

Bu algoritmanın temel mantığı, yetkisiz bir istemci şifreli mesajı ele geçirse bile özel anahtar saklı olduğu için orijinal bilgiye ulaşamayacak olmasıdır.

2.2.2. ElGamal algoritması

ElGamal şifreleme algoritması, bir açık anahtarlı şifreleme algoritması olarak 1985'de T. ElGamal tarafından tasarlandı. ElGamal hem şifreleme hem de imzalama işlemleri için kullanılmaktadır [14]. Günümüze kadar, ElGamal algoritmanın kullanıldığı bir sistem üzerine hiçbir başarılı saldırı bilinmemektedir. 1994'te bağımsız birkaç araştırma grubu, genel-anahtar, sayısal imza algoritmaları tabanlı tüm farklı logaritmaların, genelleştirilmiş "meta" algoritmanın farklı sonuçları olduğu bulgularını sundular. ElGamal'in içerdiği bu farklı logaritmalar, onun oldukça güvenli olduğunun bir göstergesidir [15,16]. Bu tip yöntemler, genel anahtardan, özel anahtarın zor bulunması prensibine dayanır. Bu zorluk, genel-özel anahtar çiftinin uzunluğuna ve ayrıca bu anahtar çiftinin ne olduğunu tahmin için yapılacak hesaplamaların güçlüğüne bağlıdır.

ElGamal şifreleme algoritmasının anahtar uzunluğu 256 bit'den, rasgele seçilmiş bir bit boyutuna kadar genişletilebilir. 1024 bit'den 2048 bite kadar değer alabilen bir anahtar uzunluğu gelecek 20 yıl için güvenli olarak düşünülmektedir. Bu tahmin günümüz hesaplama yöntemlerine, donanım alt yapısına ve gelecekteki kriptografi ve kripto analizdeki ilerlemelere bakılarak yapılmaktadır. Özel anahtar için baktığımızda ise 160 bit'den 240 bite kadar bir genişleyebilme özelliği bulunmaktadır. Yapılan analizler eşit uzunluklu anahtar değerleri için RSA ve ElGamal şifreleme algoritmalarının benzer güvenliğe sahip olduğunu göstermektedir.

ElGamal Sistemi farklı logaritmik problem üzerine dayandırılmış bir genel anahtarlama kripto sistemidir. Şifreleme ve sayısal imza algoritmalarından oluşur. Şifreleme algoritması temelde Diffie-Hellman anahtar protokolüne benzer sistem parametreleri, bir asal sayı olan p ve modül p'e göre elemanların sayısını veren g tamsayısından oluşur. Alıcı, bir özel anahtar olan a değerine ve bir genel anahtar olan y değerine sahiptir ve $y=g^a \pmod p$'dır. Varsayalım ki gönderici bir m mesajını alıcıya göndermek istiyor. Gönderici öncelikle p'den küçük olmak şartıyla rasgele bir k sayısı üretir.

Daha sonra Eş. 2.1'deki işlemi gerçekleştirerek y_1 ve y_2 değerlerini hesaplar.

$y_1 = g.k.(\text{mod } p)$ ve $y_2 = m \text{ xor } y.k$ (2.1)

Buradaki xor bit seviyesinde yapılan *özel veya* işlemini ifade etmektedir. Bu işlemden sonra gönderici (y_1,y_2) değer çiftini alıcıya gönderir. Şifrelenmiş mesajı alan alıcı, Eş. 2.2'deki işlemi gerçekleştirir.

$m = (y_1.a \text{ mod } p) \text{ xor } y_2$ (2.2)

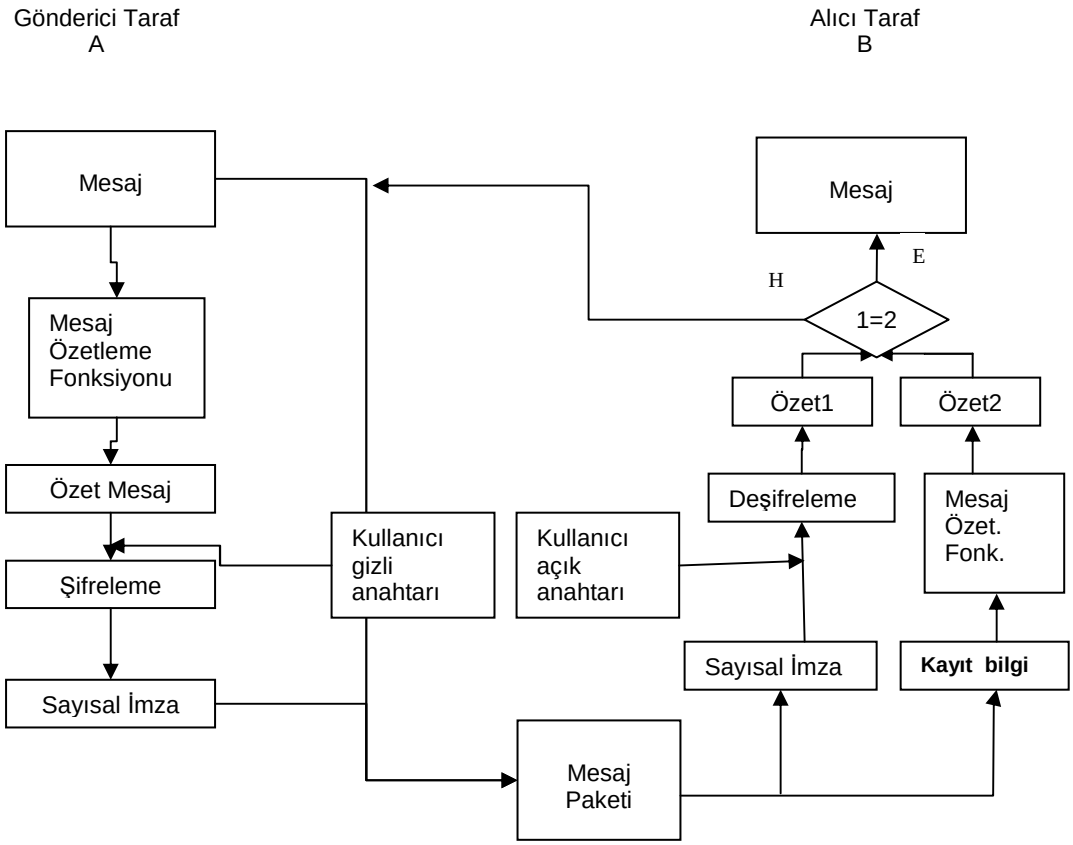
Bu hesaplamadan sonra orijinal metne ulaşılmış olunur.

3.SAYISAL İMZA SİSTEMLERİ

Sayısal imzalar, elle atılan imzaların sayısal olarak karşılığıdır. İş hayatında anlaşmaların geçerliliği imzalarla garanti altına alınmıştır. İmzalanmış bir k, kontratı elinde tutan kişinin gerektiğinde mahkemeye sunabileceği bir delil görevini üstlenir. Bu yöntemlerin tamamen sayısal bir karşılığını elde etmek için her kullanıcı, kaynağı herkes tarafından doğrulanabilen, ancak başka hiç kimse tarafından, hatta alıcı tarafından dahi üretilemeyen bir mesaj üretebilmelidir. Mesajı yazınca bir tek kişi üretebileceği ancak birçok kişi bu mesajı alabileceği için bu, bir yayın (broadcast) şifresi olarak algılanabilir [17].

Bir başka ifadeyle sayısal imza bir dokümandan özetleme fonksiyonları kullanılarak oluşturulan özetin doküman sahibinin gizli anahtarı ile şifrelemesi sonucu oluşan özel bir mesaj parçasıdır. Tanımdan da anlaşıldığı gibi bir sayısal imza hem imzalayıcı şahsa hem de imzalanan metne özel bir yapıya sahiptir. Bu iki özelliği sayesinde sayısal imza gerek kimlik tespiti gerekse mesaj bütünlüğünün korunması açısından oldukça etkili ve güvenli bir metot olarak kabul edilmektedir [18].

Her iki imza sisteminin de dayandığı ana nokta, tamamıyla iki imzaya sahip olan iki farklı insan bulma ihtimalinin çok küçük olmasıdır. Açık anahtarlı kripto sistemlerde kullanılan açık ve gizli anahtar çiftleri sayısal imzalarla bu özelliği kazandırmaktadır. Anahtarların birbirinden türetilme ihtimali yok denecek kadar azdır. Bu benzerliğin yanında sayısal imzaların bazı üstün yanları da mevcuttur. Örneğin altına normal imza atılan bir metnin üzerinde daha sonradan değişiklik yapabilmek mümkünken, sayısal imzalarda bu asla mümkün değildir. Çünkü imza imzalandığı metne göre değişiklik arz eder.



Şekil 3.1. Mesajların sayısal imza ile imzalanması

Genel olarak sayısal imza kullanımı şu şekilde gerçekleşmektedir [19].

- Gönderici taraf mesajını bir özetleme fonksiyonuna sokarak sabit uzunlukta bir özet mesaj elde eder.
- Bu özet mesaj göndericinin gizli anahtarıyla şifrelenerek sayısal imza oluşturulur ve mesaja eklenir.
- Gerekiyorsa tüm mesaj bir şifreleme algoritması kullanılarak gizli hale getirilir ve alıcıya gönderilir.
- Alıcı taraf gelen mesajı (eğer şifrelenmişse şifresini çözdükten sonra) açar. Mesajın içindeki sayısal imzayı göndericinin açık anahtarını kullanarak deşifre eder ve özet mesaja ulaşır.
- Orijinal mesajı göndericinin kullandığı özetleme fonksiyonunu kullanarak özetler ve bu özeti imzanın içerisindeki özetle karşılaştırır.
- Eğer iki özet birbirinin aynısı ise mesaj, doğru gönderici tarafından

gönderilmiş ve değişikliğe uğramadan kendisine ulaşmış demektir.

Sayısal imzada kullanılan asimetrik şifreleme, normal mesaj şifrelemenin tam tersi şeklinde işlemektedir. Mesaj alıcının açık anahtarıyla şifrelenip gizli anahtarıyla açılacağı yerde, göndericinin gizli anahtarıyla şifrelenip açık anahtarı ile açılmaktadır. Bu değişiklik sayısal imzanın doğasından kaynaklanmaktadır. Çünkü esas amaç gönderenin kimliğini imza içerisine yansıtmaktır.

3.1. Sayısal İmza Sistemleri

Sayısal imza sistemleri bir imzalama ve bir doğrulama prosedüründen oluşur. A ve B sırasıyla imzalayan ve alıcı, m imzalanacak mesaj, S_A A'ya ait imzalama transformasyonu, V_A A'ya ait doğrulama transformasyonu olsun. Burada S_A, A tarafından imza üretmek için kullanılacak ve gizli tutulacaktır. V_A ise herkese açık olup A tarafından üretilen imzaları doğrulamak için kullanılır ve sonuçta "doğru" ya da "yanlış" şeklinde çıkışlar verir.

S_A ve V_A dönüşümleri bir sayısal imza protokolü oluşturur. Bazen buna sayısal imza mekanizması adı da verilir [20].

3.1.1. İmzalama prosedürü

A tarafı (imzalayan) m mesajı için aşağıdaki gibi bir imza oluşturur:

1. $s=S_A(m)$ hesaplanır

2. (m,s) çifti B tarafına gönderilir. s' e m mesajının imzası adı verilir.

3.1.2. Doğrulama prosedürü

Bir mesaj üzerindeki s imzasının A tarafından üretildiğini doğrulamak için, B tarafı (doğrulayan) aşağıdaki işlemleri yapar.

1. A' nın doğrulama fonksiyonunu elde eder.

2. u=V_A (m,s)’ i hesaplar

3. Eğer u=”doğru” ise imzanın A tarafından üretildiği kabul edilir, aksi takdirde, eğer u=”yanlış” ise imza reddedilir.

Genellikle S_A ve V_A bir anahtarla karakterize edilir. Yani herkes tarafından bilinen bir imzalama ve doğrulama algoritması sınıfı vardır ve her algoritma bir anahtar ile tanımlanır. Böylece A’nın S_A imzalama algoritması bir k_A anahtarı ile belirtilir ve A’nın yalnızca bu anahtarı gizli tutması gerekir. Aynı şekilde A’nın doğrulama algoritması herkese açık bir I_A anahtarı ile belirlenir.

3.1.3. İmzalama ve doğrulama fonksiyonlarının özellikleri

İmzalama ve doğrulama dönüşümlerinde olması gereken bazı özellikler vardır:

1. s’in m mesajı üzerinde geçerli bir imza olabilmesi ancak ve ancak V_A (m)=”doğru” olması ile olabilir.

2.A’dan başka bir tarafın bir m, m mesajı için V_A (m,s)=”doğru” olacak şekilde bir s imzası bulması hesaplama açısından verimsiz olmalıdır.

İkinci maddedeki özelliği sağlayan sayısal imza protokollerinin mevcudiyeti henüz resmi olarak kanıtlanmamıştır. Ama bunu sağladığı düşünülen bir takım yöntemler vardır.

Sayısal imzaların elle atılan imzaların yerini alabilmesi için bazı şartları sağlaması gerekmektedir. Her sayısal imza yönteminin sağlaması gereken genel şartlar şunlardır [21]:

Kimlik sınaması (Authentication) : Ağ güvenliği açısından kimlik sınaması; alıcının, göndericinin iddia ettiği kişi olduğundan emin olmasıdır. Bunun yanında, bir bilgisayar programı kullanırken bir parola girmek de kimlik sınaması çerçevesinde değerlendirilebilir. Günümüzde kimlik sınaması, sadece bilgisayar ağları be sistemleri için değil, fiziksel sistemler için de

önemli biri hizmet haline gelmiştir. Akıllı karta ya da biyometrik teknolojilere dayalı kimlik sınama sistemlerinin kullanımına yaygın olarak başlanmıştır.

Veri bütünlüğü (Integrity) : Bu hizmetin amacı, veriyi göndericiden çıktığı haliyle alıcısına ulaştırmaktır. Bu durumda veri, haberleşme sırasında izlediği yollarda değiştirilmemiş, araya yeni veriler eklenmemiş, belli bir kısmı ya da tamamı tekrar edilmemiş ve sırası değiştirilmemiş şekilde verebiliriz. Şöyle ki; alıcıda iki tür bütünlük sınaması yapılabilir: Bozulma sınaması ya da düzeltme sınaması. Bozulma sınaması ile veririn göndericiden alıcıya ulaştırılması sırasında değiştirilip değiştirilmediğinin sezilmesi hedeflenmiştir. Düzeltme sınamasında ise, bozulma sınamasına ek olarak eğer veride değişik sezildiyse bunu göndericiden çıktığı haline döndürmek hedeflenmiştir.

İnkâr edememe (Non-repudiation): Bu hizmet sayesinde, ne gönderici alıcıya bir mesajı gönderdiğini ne de alıcı göndericiden bir mesajı aldığını inkâr edebilir. Bu hizmet, özellikle gerçek zamanlı işlem gerektiren finansal sistemlerde kullanım alanı bulmaktadır ve gönderici ile alıcı arasında ortaya çıkabilecek anlaşmazlıkların en aza indirilmesi sağlamaya yardımcı olmaktadır.

Bu hizmetler, zaman içerisinde bilgisayar sistemlerine karşı ortaya çıkmış tehditler ve yaşanmış olaylar sonucunda ortaya konulmuştur. Yani her bir hizmet, belli bir grup potansiyel tehdide karşı sistemi korumaya yöneliktir, denilebilir.

Gizlilik (Privacy): Bilginin yetkisiz kişilerin eline geçmesinin engellenmesidir. Gizlilik, hem kalıcı ortamlarda (disk, teyp, vb.) saklı bulunan veriler hem de ağ üzerinde bir göndericiden bir alıcıya gönderilen veriler için söz konusudur. Saldırganlar, yetkileri olmayan verilere birçok yolla erişebilirler. Parola dosyalarının çalınması, sosyal mühendislik, bilgisayar başında çalışan bir kullanıcının, ona fark ettirmeden özel bir bilgisini ele geçirme (parolasını girerken gözetleme gibi). Bunun yanında trafik analizinin, yani hangi gönderici ile hangi alıcı arasında haberleşmenin olduğunun belirlenmesine

karşı alınan önlemler de gizlilik hizmeti çerçevesinde değerlendirilir.

Bu özelliklere ek olarak yakın zamanda ortaya çıkan sayısal imza metotları şu özellikleri de taşımaktadır:

- İmzası taklit edilen bir imzacı, kanun karşısında imzanın bir taklit olduğunu kanıtlayabilmelidir. Bu, Taklit Edildiği Kanıtlanabilen imzaların bir özelliğidir.
- İmzanın alıcısı, imzalayanın yardımı olmadan imzayı başkalarına gösteremez. Bu yüzden, bu mekanizmalar "görünmez" şeklinde adlandırılabilir. Bu özellik kullanıcının imzaladığı mesajların gizli olması durumunda yararlı olacaktır. Ancak imzalayan imzayı kabul etmeye ya da reddetmeye zorlandığında gerçek imzalarını reddedememelidir (undeniable signature).
- Bir saldırganın sınırsız hesap gücüne sahip olsa bile rasgele bir sayı seçip bunun doğru imza olduğunu düşünmekten daha iyi taklitler üretemeyeceği imzalar vardır (şartsız güvenli imzalar-unconditionally secure signature). Ancak bu imza mekanizmalarının pratikte uygulanması çok zordur.

3.2. Sayısal İmzaların Karşı Karşıya Olduğu Tehditler

Sayısal imzalar söz konusu olduğunda bir "hasmın" hedefi imzayı taklit etmek yani başka bir kişinin imzası olarak kabul edilecek imzalar üretmek olacaktır. Aşağıda bir imza mekanizmasını kırmanın ne anlama geleceğine dair ölçütler verilmiştir [20].

Toplam kırma: Bir hasım ya da imzalayanın gizli anahtar bilgisini hesaplama yeteneğine sahiptir ya da geçerli imzalama algoritmasına fonksiyonel olarak eşdeğer olan bir imzalama algoritması bulur.

Seçici taklit: Bir hasım daha önceden seçilen belli bir mesaj ya da mesaj sınıfı için geçerli bir imza üretme yeteneğine sahiptir. İmzanın üretilmesi kanuni imzalayıcı ile doğrudan ilgili değildir.

Existansiyel taklit: Bir hasım, en az bir mesaj için geçerli bir imza üretmeyi

başarır. Hasım imzası elde edilen mesaj konusunda çok az kontrole sahiptir ya da hiç kontrole sahip değildir ve kanuni imzalayıcı bu olarak karışabilir.

Açık anahtarlı imza mekanizmalarına karşı iki temel saldırı türü vardır:

Yalnızca anahtardan oluşan saldırılar: Bu durumda hasım yalnızca imzalayanın açık anahtarını bilir.

Mesaj saldırıları: Burada hasım ya bilinen ya da seçilen mesajlara ilgili imzaları inceler. Bunlar da üçe ayrılır.

- Bilinen mesaj saldırısı, bir hasım kendisinin bildiği ama kendisi tarafından seçilmeyen bir grup mesajı elinde bulundurduğu saldırı türüdür.
- Seçilmiş mesaj saldırısı, imza mekanizmasını kırma teşebbüsünde bulunmadan önce hasımın, seçilmiş bir mesaj listesinden geçerli imzalar elde etmesine dayanan saldırı türüdür. Bu saldırı adaptif değildir çünkü mesajlar herhangi bir imza görülmeden seçilir.
- Adaptif seçilmiş mesaj saldırısında, bir hasım imzalayanın açık anahtarına ve daha önceden imza ve mesajlara dayanan mesajlar için imza talep edebilir.

Prensip olarak adaptif seçilmiş mesaj saldırısı önlenmesi en zor olan saldırı türüdür. Yeterince mesaj ve bunlarla ilgili imzalar verildiğinde hasmın bunlardan bir sonuç çıkarması ve imzayı taklit etmesi mümkün olabilir. Bu tür bir saldırı pratikte zor uygulanabilir olsa da iyi tasarlanmış bir imza mekanizması bu ihtimale karşı da koruyucu olmalıdır.

3.3. Sayısal İmza Sistemlerinin Sınıflandırılması

Sayısal imza sistemleri, imzalanan mesajın doğrulama algoritmasında giriş olarak kullanıp kullanılmamasına göre ikiye ayrılır. Çizelge 3.1’de sayısal imza sistemlerinin sınıflandırılmasında kullanılan küme ve fonksiyonlar listelenmiştir.

Çizelge 3.1. Sayısal imza sistemlerinin sınıflandırılmasında kullanılan notasyon [19]

M	Mesaj uzayı
M_S	İmzalama uzayı
S	İmza uzayı
R	M' den M_S' e 1-1 bir fonksiyon
M_R	R' nin görüntü kümesi
R-1	R' nin tersi
R	İmzalama için indisleme kümesi adı verilen bir küme
h	Tek yönlü bir özetleme fonksiyonu
M_h	H' nin görüntü kümesi

3.3.1. Sonuna eklemeli sayısal imza sistemleri

Bu tür sistemlerde orijinal mesaj doğrulama algoritmasının girişlerinden biri olmalıdır. Pratikte en çok kullanılan sayısal imza sistemleri bu türdendir. Bunlar daha çok kriptografik özetleme fonksiyonlarına dayanırlar ve egzistansiyel taklit saldırılarına karşı daha az duyarlıdırlar. Bunlara örnek olarak DSA ve ElGamal gösterilebilir.

3.3.2. Mesajın geri alınabildiği sayısal imza sistemleri

Bu tür imza sistemlerinde orijinal mesajın doğrulama algoritmasına giriş olarak verilmesi gerekmez. Mesaj imzanın kendisinden geri elde edilebilir. Bu sisteme örnek olarak RSA imza sistemleri verilebilir.

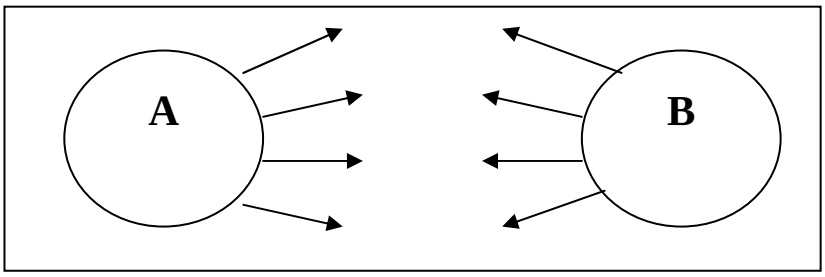
3.4. Anahtar Dağıtım Tekniği

Sayısal imzalarda kullanılan anahtarların dağıtımı için birçok teknik mevcuttur. Tüm bu teknikleri dört kategoride toplamak mümkündür [22].

- Açık Dağıtım
- Açık-Klasör Yapılanması
- Açık-Anahtar Otorite Kontrolü
- Açık-Anahtar Sertifikaları

3.4.1. Açık dağıtım

Ana çıkış noktası; iki anahtarlı kripto sistemlerde açık anahtarın adı üstünde herkese açık olmasıdır. Açık anahtar ortama Şekil 3.2'de görüldüğü gibi çoklu yayın (broadcast) olarak yayımlanır. Ortamdaki tüm taraflar anahtarlara erişebilir.



Şekil 3.2. Açık dağıtım

Bu yöntemin çok kolay ve kullanışlı olmasına karşın, büyük bir yetersizliği vardır: Herhangi birisi siz olduğunu iddia ederek açık anahtarını kişilere duyurabilir. Yani bir kullanıcı kendisini olmadığı halde A kullanıcısı gibi tanıtabilir ve kendi açık anahtarını A kullanıcısının açık anahtarıymış gibi kişilere ilan edebilir. Gerçek A kullanıcısı sahtekarlığın farkına varana kadar ya da bir başka kullanıcı onu uyarana dek, sahte A kullanıcısı gerçek A kullanıcısına gönderilmek üzere şifrelenmiş bütün mesajları okuyabilir [23].

Herkes Tarafından Erişilebilir Adres Rehberi

Herkesçe erişilebilir dinamik bir açık anahtar adres rehberinin iyi bir şekilde korunması ve organize edilmesiyle çok yüksek derecede güvenlik sağlanabilir. Adres rehberlerindeki anahtarların dağıtımı ve korunması güvenilir kimi kişi veya kuruluşların sorumluluğunda olmalıdır. Böyle bir

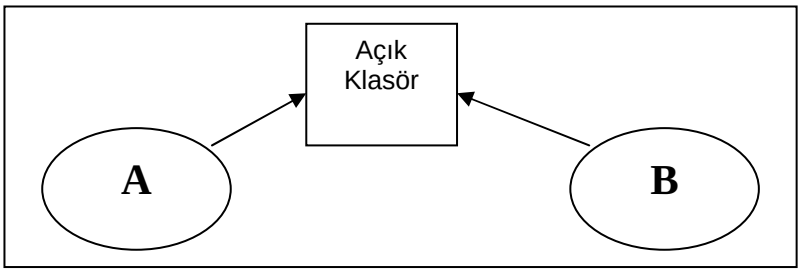
hizmet tasarlanırken aşağıdaki koşullar sağlanmış olunmalıdır:

- Adres rehberinin her elemanı için bulunacak (ad, açık anahtar) bölümleri iyi korunmalıdır.
- Her kullanıcı rehber yetkilileri ile bir açık anahtar kaydetmeli ve bu işlemi yüz yüze ya da kimlik kontrolü ile güvenli hale getirilmiş bir iletişim metodu ile yapmalıdırlar.
- Herhangi bir üye adres rehberindeki genel anahtarını, bu anahtarla çok fazla verinin şifrelenmiş olmasından dolayı ya da bu anahtarla ilişkili olan özel anahtarın herhangi bir sebepten ötürü güvenilirliğini yitirmesinden ötürü istediği bir zaman yenisi ile değiştirebilmelidir.
- Bu sistemin yönetimi, periyodik olarak adres rehberini güncellemeli, bir telefon rehberinde olduğu gibi isimleri ve genel anahtarlarının kopyasını saklamalı ve güncellemeli, geniş bir kitleye ulaşabilen yayın organı (gazete gibi) yoluyla diğer kullanıcılara haber vermelidir.
- Üyeler aynı zamanda adres rehberine elektronik yolla da ulaşabilmeli, bunun için de adres rehberinin yönetimi, gizlilik ve kimlik denetimi gerektiren güvenli bir iletişim metodu kullanılması zorunluluğunu yerine getirilmiş olmalıdır.

Bu yapının bireysel genel duyuru metodundan daha güvenli olduğu oldukça açıktır ancak halen kimi savunmasız noktalar bulunmaktadır. Eğer bir saldırgan, adres rehberi yöneticisinin özel anahtarını bir şekilde ele geçirir ya da onu hesapsal metotlarla bulmayı başarırsa, kolay bir şekilde güvenlik engellerini aşarak kişilerin açık anahtarlarını istediği açık anahtarlar ile değiştirip, onlara gelecek olan mesajları okuyabilir ve istediği kişiyi taklit ederek diğer üyelere onun adına mesaj atabilir

3.4.2. Açık klasör yapılanması

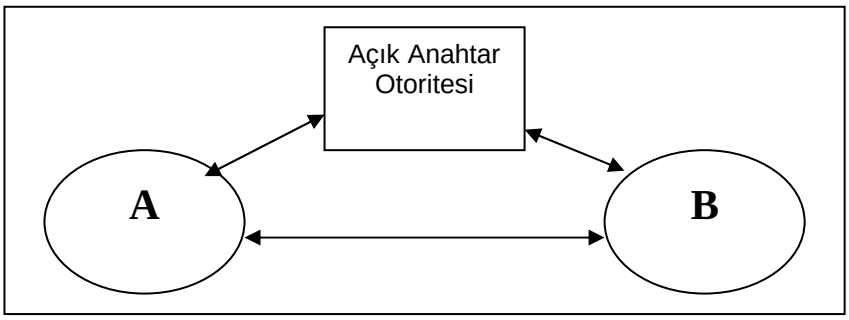
Açık dağıtımdan daha güvenli bir tekniktir. Belirli bir organizasyon tarafından idame edilen açık anahtarları içeren klasörlerin kullanıcılara Şekil 3.3’ de görüldüğü gibi açık olarak sunulmasıdır.



Şekil 3.3. Açık klasör yapılanması

3.4.3. Açık anahtar otorite kontrolü

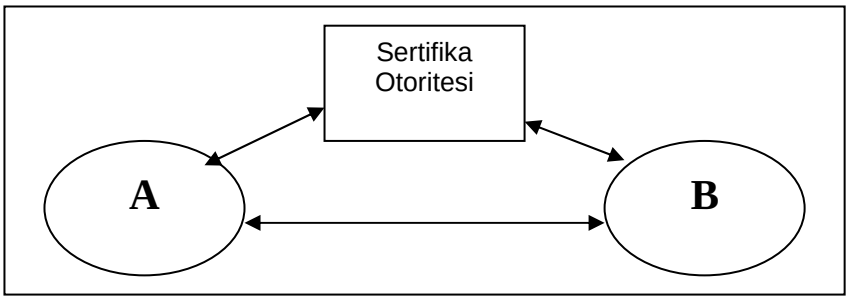
G. L. Popek ve C. S. Kline'ın 1979 yılında yayımladıkları "Encryption and Secure Computer Networks" isimli makaleleri [24] temel alınarak oluşturulmuş bu anahtar dağıtım tekniği açık klasör yapılanmasından daha güvenli bir tekniktir. Açık anahtarların bulunduğu klasör/dosyalara kontrollü erişim olanağı sağlar. Şekil 3.4'de görüldüğü gibi her kullanıcının erişebileceği bir açık anahtar otoritesi mevcuttur. Herhangi bir kullanıcı başka bir kullanıcının açık-anahtarı için otoriteye başvurur. Bu başvuru sonucunda ancak iletişime geçeceği diğer kullanıcının anahtarını elde edebilir.



Şekil 3.4. Açık anahtar otorite kontrolü

3.4.4. Açık anahtar sertifikaları

Herhangi bir açık anahtar otoritesine ihtiyaç duymadan iletişime geçecek iki kullanıcının, sertifika otoritesinden aldıkları sertifika yardımıyla birbirlerine açık anahtarlarını gönderme metodudur (Şekil 3.5) [25].



Şekil 3.5. Açık anahtar sertifikası

Alternatif bir yöntem, ilk kez L. M. Kohnfelder tarafından, 1978 yılında yayımlanan "A Method for Certification" isimli makalede sunulmuştur. Bu yönteme göre kullanıcılar sertifika kullanarak, bir genel anahtar yöneticisine başvurmadan güvenli bir yolla anahtarlarını değiştirebilmekte ve böylece güvenli iletişime başlayabileceklerdir. Bir sertifika yöneticisi/yetkilisi tarafından oluşturulmuş sertifika, bir genel anahtarı ve ek bilgiyi içerir ve bu sertifika üzerindeki genel anahtara uygun özel anahtar ile beraber kullanıcıya verilir. Bir kullanıcı diğer bir kullanıcıya açık anahtarını, sertifikasını göndererek gösterir. Diğer kullanıcı da, bu sertifikanın bir otorite tarafından yaratılmış olduğunu doğrulayabilir. Bu yöntemin gerekliliklerini şu şekilde sıralayabiliriz [23]:

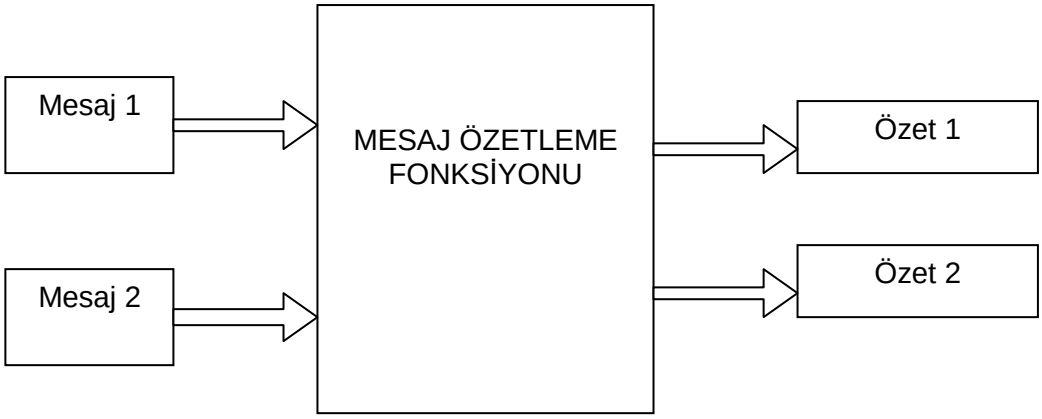
- Herhangi bir kullanıcı bir sertifikanın kullanıcısının adını ve açık anahtarını öğrenmek üzere okuyabilir.
- Herhangi bir kullanıcı bu sertifikanın, sertifika yöneticileri tarafından oluşturuluş orijinal bir sertifika olduğunu kanıtlayabilir.
- Sertifikaları sadece sertifika yöneticisi/otoritesi değiştirebilir ve yaratabilir.

Bu gereklilikler, Kohnfelder'in makalesinde geçen gereklilikler. Bu gerekliliklere D. E. Denning tarafından 1983 yılındaki "Cryptography and Data Security" isimli makalesi ile şu gereklilik de eklenmiştir:

“Herhangi bir kullanıcı bir sertifikanın geçerliliğini kanıtlayabilir.”

3.5. Mesaj Özetleme Fonksiyonları

Bir özetleme fonksiyonu, herhangi bir uzunluktaki metni, giriş değeri olarak alır ve sonuç olarak sabit uzunlukta bir değer üretir. Bu değere mesaj özeti (message digest) adı verilir. Burada üretilen özet, fonksiyona giren metnin karakterini taşımaktadır denilebilir. Giriş metninde yapılacak tek bir karakterlik değişikliği bile üretilecek özette büyük değişikliklere yol açar. Ayrıca, özetleme fonksiyonu tek yönlü olduğundan, özetten asıl metne geri dönüş yoktur. Özetleme fonksiyonları, uzun metinlerin, asimetrik bir algoritma ile şifrelenmeleri sırasında, asimetrik algoritmanın başarım dezavantajını ortadan kaldırmak amacıyla kullanılırlar. Tüm mesaj metni değil de yalnızca mesajın özeti alınarak asimetrik algoritmayla şifrelenir [21].



Şekil 3.6. Mesaj özetleme fonksiyonları

Matematiksel ifadesiyle h=H(m) şeklinde gösterilen bu sistemde, H(m) fonksiyonu tek yönlü, polinomsal zamanda hesaplanabilen ve birebir bir dönüşümü, m değişebilen uzunluktaki herhangi bir mesajı, h ise sabit uzunluktaki özet mesajı ifade etmektedir.

Kriptolojide kullanılan özetleme fonksiyonlarının şu şartları sağlaması gerekmektedir.

- Girdi değişen uzunlukta seçilebilir.
- Çıktı sabit uzunluktadır.

- H(m) fonksiyonu kolay hesaplanabilir bir fonksiyon olmalıdır.
- H(m) fonksiyonu tek yönlü bir fonksiyon olmalıdır.
- H(m) fonksiyonu çarpışmasız bir fonksiyon olmalıdır.

Bir fonksiyonun tek yönlü olması, fonksiyonun tersinin hesaplanmasının yani h=H(m) şartını sağlayan h değerleri için uygun m değerinin bulunması işleminin mümkün olmamasıdır.

Çarpışmasız fonksiyon, m girdisinin farklı her değeri için bir h çıktısı üreten fonksiyondur. Kripto sistemler açısından bu tanım şu şekilde yapılabilir. Bir mesaj özetleme fonksiyonu farklı iki mesajdan aynı özeti üretmemelidir.

Mesaj özetleme fonksiyonlarının kullanım alanı oldukça geniştir. Özellikle veri bütünlüğünün korunması ve mesaj değişikliklerinin tespit edilmesi amacıyla kullanılmaktadır. Sayısal imzalar oluşturulurken mesajın bir özeti bu fonksiyonlar yardımıyla oluşturularak imza içine konur. Böylelikle üretilen imza mesaj bağımlı bir imza haline getirilmiş olur.

Bilinen en meşhur özetleme fonksiyonları Rivest tarafından geliştirilen 128 bit'lik MD2 *(*Message Digest - Mesaj Özeti*),* MD4 ve MD5 ile NIST ve NSA tarafından ortaklaşa geliştirilen ve NIST'ın özetleme fonksiyonları standardı olarak kabul ettiği 160-bit lik SHS (Securiy Hash Standart – Güvenli Özetleme Standardı*)* içinde kullanılan SHA (Security Hash Algorithm – Güvenli Özetleme Algoritması) algoritmalarıdır.

3.5.1. Özetleme fonksiyonunun güvenirliliği

Eğer verilen bir özetleme değerine sahip bir katar üretilebilirse ya da özetleme değerini üreten iki farklı katar bulunabilirse o özetleme fonksiyonunun kırıldığı söylenir. Örnek olarak Ayşe ile Ahmet'in bir kontrat imzalamak için sayısal imzaları kullandığı bir örneği düşünelim. Ayşe özetleme değerini hesaplar ve bu değeri imzalar. Ahmet aynı özetleme değerine sahip başka bir kontrat üretebilir. Bu durumda sanki Ayşe bu

kontratı imzalamış gibi görünecektir ve bu kullanıcı sahte kontratı imzaladığını ispatlayamaz. Böyle bir mesajın bulunması, özetleme fonksiyonunun b bit uzunluğunda çıkış ürettiği durum için 2^{b-1} işlem sürer.

Eğer Ahmet aynı özetleme değerine sahip iki mesaj bulabilir ancak sonuçtaki özetleme değerini kendisi seçmez ise, farklı anlam taşıyan iki mesaj arayabilir. Sonra Ayşe'den ilk kontratı imzalamasını ister. Bu saldırı Ahmet için daha kolaydır çünkü iki mesajın bulunması ortalama $2^{b/2}$ işlem sürer. Ancak Ayşe kendisini protokolü değiştirerek koruyabilir. Örneğin özetleme değeri hesaplamadan önce mesaja rasgele bir katar ekler ve bunu imzalar.

3.5.2. Özetleme fonksiyonlarının uzunlukları

64 bit'lik özetleme fonksiyonları yukarıdaki türden saldırılardan kurtulmak için çok yetersizdir. Çoğu tek yönlü özetleme fonksiyonu 128 bit'lik özetleme değeri üretir. Bu da yukarıdaki saldırıya teşebbüs eden birini aynı özetleme değerini üreten iki doküman bulmak için 2^{64} dokümanı özetleme fonksiyonundan geçirmesini gerekli kılar. Ama bu da uzun süreli güvenlik açısından yetersiz bulunmuştur. NIST, SHS'de 160 bit 'ik bir özetleme değeri kullanır.

Verilen bir özetleme fonksiyonunun ürettiği bir özetleme değerinden daha uzun bir özetleme değeri ifade etmek için aşağıdaki metot önerilmiştir [26].

1. Çeşitli özetleme fonksiyonları kullanılarak mesajın özetleme değeri üretilir.
2. Özetleme değeri mesaja eklenir.
3. Mesajla özetleme değerinin birbirine eklenmesinden oluşan kataın özetleme değeri hesaplanır.
4. Birinci adımda üretilen özetleme değeriyle üçüncü adımda üretilen değerin birbirine eklenmesinden oluşan daha büyük bir özetleme değeri hesaplanır.
5. Birinci ve üçüncü adımlar istendiği kadar tekrar edilip birbirine eklenerek devam edilir.

Yukarıdaki metodun güvenli ya da güvensiz olduğu şu zamana kadar ispatlanmış değildir.

3.5.3. Özetleme algoritmaları

Günümüzde yaygın olarak kullanılan özetleme fonksiyonları aşağıda açıklanmıştır.

MD4 algoritması

MD4 1990 yılında Ronald L. Rivest tarafından geliştirilmiş tek yönlü bir özetleme fonksiyonudur. MD4 128 bit'lik bir özetleme değeri üretir [27].

Bu özetleme fonksiyonu şöyle uygulanır;

Giriş b bit'lik bir mesajdır. b, negatif olmayan, keyfi bir sayıdır; sıfır olabilir, b'nin sekizin bir katı olması gerekmez ve istenildiği kadar büyük olabilir. Mesajın bitlerinin aşağıdaki şekilde yazıldığı farz edilir (Eş. 3.1).

$$m_0 m_1 \ldots m_{b-1} \quad (3.1)$$

Mesajın özetleme değerini hesaplamak için dört adım uygulanır. Bunlar;

Mesajın sonuna bit eklenmesi: Mesaj, bit olarak uzunluğu 512 modülüne göre 448'e kongürant oluncaya kadar bir ekleme (padding) işlemine tabi tutulur.

Uzunluğun eklenmesi: Bir önceki adımın sonucuna b'nin 64 bit'lik bir ifadesi eklenir.

MD tamponunun ilklendirilmesi: Özetleme değerinin hesaplanması için dört sözcüklük (A,B,C,D) bir tampon kullanılır. Burada A,B,C,D birer 32-bit'lik saklayıcılardır. Bu saklayıcılar düşük anlamlı sekizliler önce gelecek şekilde ilklendirilir

Mesajın 16 sözcüklük bloklar halinde işlenmesi: Önce giriş olarak üç adet 32

bit'lik sözcük alan ve çıkışta 32 bit'lik bir sözcük üreten üç yardımcı fonksiyon tanımlanır. Gerçekleştirilen işlemler sonucunda oluşan özetleme değeri A,B,C ve D'dir. Yani A'nın düşük anlamlı sekizlisi ile başlar ve D'nin yüksek anlamlı sekizlisi ile biter.

Rivest algoritmanın tasarımında amaçladıklarını şöyle özetlemiştir:

"MD4 herhangi bir varsayıma dayanmaz. MD4 basit bir algoritmadır. Büyük veri yapıları kullanmaz. Mikroişlemci mimarileri için optimize edilmiştir ve 32 bit işlemciler üzerinde oldukça hızlı çalışmaktadır."

MD5 algoritması

Rivest'in MD5'i geliştirmesinin sebebi, MD4'ün çok kısa zamanda uygulamaya geçirilmesi ve çok hızlı hazırlandığı için kriptanalitik saldırılar riski açısından sınırda bulunmasıdır. MD5 biraz daha yavaştır. Daha fazla güvenlik için hızdan biraz fedakârlık eden bir algoritmadır.

Algoritmanın ilk üç adımı MD4 ile aynıdır. Sadece mesajın 16 sözcüklük bloklar halinde işlenmesi adımı farklılık gösterir.

MD2 algoritması

MD2 yine Rivest tarafından geliştirilmiş olan ve MD4 algoritmasına dayanan bir başka algoritmadır.

MD2 özetleme fonksiyonu şöyle uygulanır:

Giriş b bit'lik bir mesajdır. B, negatif olmayan, keyfi bir sayıdır; sıfır olabilir, b'nin sekizin bir katı olması gerekmez ve istenildiği kadar büyük olabilir. Mesajın bitlerinin aşağıdaki şekilde yazıldığı farz edilir.

$m_0 m_1 \ldots m_{b-1}$

Mesajın özetleme değerini hesaplamak için aşağıda verilen dört adım

uygulanır.

Mesajın sonuna bit eklenmesi: Mesaj, bit olarak uzunluğu 16 modülüne göre 0'a kongürant oluncaya kadar bir ekleme işlemine tabi tutulur.

Checksum eklenmesi: Önceki adımın sonucuna 16 byte'lik bir checksum eklenir. Bu adımda π'nin dijitlerinden üretilen 256 byte'lık rasgele permütasyon kullanılır.

MD tamponunun ilklendirilmesi: Özetleme değerinin hesaplanması için 48 byte'lik bir *X* tamponu kullanılır. Tampon sıfırlar ilklendirilir.

Mesajın 16 sözcüklük bloklar halinde işlenmesi: Bu adımda 2. adımda kullanılan 256 byte'lık *S* permütasyonunun aynısı kullanılır. Özetleme değeri X[0….15] dir.

MD4 ve MD5 arasındaki farklar

- MD5'de 4. bir çevrim eklenmiştir.
- Her adımda diğerinden farklı eklenen (additive) bir sabit vardır
- MD4'deki 2. çevrimdeki g fonksiyonu (XY or XZ or YZ) g'yi daha az simetrik yapmak için MD5' de (XZ or Y not(Z))'ye dönüştürülmüştür.
- Her adımda bir önceki adımın sonuçları eklenir. Bu çığ etkisi (avalanche effect)'nin daha hızlı olmasını sağlar
- 2. ve 3. çevrimlerde, giriş sözcüklerinin erişim sırası bunların birbirine daha az benzemesi için MD5'de değiştirilmiştir.
- Her çevrimdeki kaydırma (shift) miktarı hızlı bir çığ etkisi oluşturmak için yaklaşık olarak optimize edilmiştir. Farklı çevrimlerdeki kaydırmalar birbirlerinden farklıdır.

MD5, yine Boer ve Bosselaers sıkıştırma (compression) fonksiyonunun çakışmaya dirençli olmadığını gösterdiği halde bugün en sık kullanılan özetleme fonksiyonudur. Bulunan çakışma mesaj bloğunu değil, zincirleme değişkenlerini değiştirir. Bu MD5'in standart uygulamaları için bir tehdit

oluşturmaz ama yine de tasarım prensiplerinin çiğnendiğini gösterir.

SHA algoritması

SHA (Secure Hash Algorithm-Güvenli Özetleme Algoritması), NIST ve NSA tarafından Sayısal İmza Standardında (DSS-Digital Signature Standard) kullanılmak üzere geliştirilmiştir. SHA-0, SHA-1, SHA-256, SHA-384 ve SHA-512 olmak üzere beş türü vardır. SHA-1 bir çok uygulamada yaygın olarak kullanılmaktadır ancak Xiaoyun Wang, Yiqun Lisa Yin, ve Hongbo Yu'dan oluşan bir araştırma ekibi tarafından yapılan bir araştırma, SHA-1 kriptografik özet algoritmasında, normalde tahmin edilenden daha erken bir değerde çakışmaya rastlandığını ortaya koymaktadır. SHA-1'den sonraki algoritmalar SHA-2 olarak nitelendirilmektedir ve SHA-2 algoritmasına ilişkin henüz bir kırılma tehlikesi bildirilmemiştir [28].

Çizelge 3.2. Çeşitli çıkış boyutlarındaki SHA algoritmaları [28]

Algoritma	Çıkış boyutu (bits)	İfade edilen boyutu (bits)	Blok boyutu (bits)	Parça boyutu (bits)	Mesaj boyutu (bits)	Gerçekleştirilen operasyonlar	Çakışma (collision)
SHA-0	160	160	512	64	32	+,and,or,xor,rotl	Var
SHA-1	160	160	512	64	32	+,and,or,xor,rotl	Kusurlu
SHA-256/224	256/224	256	512	64	32	+,and,or,xor,shr,rotr	Yok
SHA-512/384	512/384	512	1024	128	64	+,and,or,xor,shr,rotr	Yok

SHA şu şekilde uygulanır:

Mesajın sonuna bit eklenmesi: Mesajın uzunluğunun 512'nin bir katı olması

için mesajın sonuna bit eklenmesi yapılır. Özetleme değerini üretirken SHA, mesajı 512 bit'lik bloklar halinde işler. Mesajın uzunluğunu 512'nin bir katı yapmak için mesajın sonuna bir "1", ondan sonra mesajın uzunluğunun 512'nin katı olmasına 64 bit kalıncaya kadar "0" eklenir. En son olarak da mesajın orijinal uzunluğunun 64 bit'lik ifadesi eklenir.

Bit eklenmiş mesajı M_1, M_2, … M_n ile gösterelim. Burada her M_i, 16 sözcüklük bir bloğu ifade eder.

Fonksiyonlar: SHA'da f_0, f_1,…f_{79} şeklinde bir grup mantıksal fonksiyon kullanılır. Her f_t fonksiyonu 32-bit'lik B,C,D sözcükleri üzerinde işlem yapar ve çıkış olarak 32 bit'lik sözcükler üretir. f_t aşağıdaki şekilde tanımlanır:

f_t(B,C,D)= (B and C) or ((not B) and D) $(0 \leq t \leq 19)$

f_t(B,C,D)= B xor C xor D $(20 \leq t \leq 39)$

f_t(B,C,D)= (B and C) or (B and D) or (C and D) $(40 \leq t \leq 59)$

f_t(B,C,D)= B xor C xor D $(60 \leq t \leq 79)$

Sabitler: Özetleme değeri bit eklenmiş mesajı kullanır. Değerin hesaplanmasında her biri beş adet 32 bit'lik sözcükten oluşan iki tampon ve seksen adet 32 bit'lik sözcük kullanılır. Beş sözcüklü ilk tamponun sözcükleri A,B,C,D,E şeklinde etiketlenir. İkinci tamponun sözcükleri ise H_0, H_1, H_2, H_3, H_4, şeklinde etiketlenir. 80 sözcüklük sekansın sözcükleri W_0, W_2,… W_{79}, ile gösterilir. Ayrıca tek bir TEMP tamponu kullanılır.

160 bit'lik özetleme değeri H_0, H_1, H_2, H_3, H_4, sözcüklerinin birbirine eklenmesinden oluşur.

SHA'nın güvenliği: SHA MD4'e çok benzer ancak 160 bit ve daha üzeri özetleme değerleri üretir. SHA'da bir önceki adımın çıkışının bir sonraki adıma eklenmesi daha hızlı bir çığ etkisi oluşmasını sağlamaktadır. Aynı şekilde SHA'nın MD5'den farkı, ek bir çevrim eklenmesi ve daha iyi bir çığ

etkisine sahip olmasıdır. Ayrıca SHA'nın daha yüksek bitlerde özetleme değeri ürettiği için brute-force saldırılarına karşı 128-bit'lik özetleme fonksiyonlarından daha güvenilirdir [26].

RIPEMD-160 algoritması

RIPEMD Avrupa Topluluğu'nun RIPE (Race Integrity Primitives Evaluation) projesi için geliştirilmiştir. Bu algoritma Rivest'in MD4 algoritmasına dayanır ve 128 bit'lik özetleme değeri üretir. RIPEMD temel olarak MD4'ün iki paralel sürümünden oluşur. Algoritmadaki kaydırmalarda ve mesaj sözcüklerinin sırasında bazı değişiklikler yapılmıştır. İki paralel kolun birbirinden tek farkı çevrimlerdeki sabitlerdir. Sıkıştırma fonksiyonunun sonunda sağ ve sol yarının sözcükleri birbirine eklenir.

1995'in ilk aylarında H. Dobbertin RIPEMD'nin üç çevriminin son ikisi için (ve daha sonra ilk ikisi için) çakışmalar bulmuştur. 1996 yılında RIPEMD'nin brute-force saldırısına karşı sınırlı güvenliği olduğu düşünülerek 160-bit'lik versiyonu geliştirilmiştir [34]. RIPEMD çeşitli bankacılık uygulamalarında kullanılmaktadır. RIPEMD-160'ı geliştiren araştırmacılar algoritmanın en az on yıl süreyle yeterince güvenli olmayacağını düşünmektedirler.

RIPEMD–160 32-bit'lik sözcükler üzerinde işlem yapar. Ekleme işlemi MD4'ünki ile aynıdır.

Başarım değerlendirmesi

Çizelge 3.3'de MD4 algoritmasına dayanan altı özetleme algoritmasının 90MHz'lik bir Pentium işlemcide gösterdiği performans görülmektedir. Örnekte 256 MB'lık veri 8K'lık bir tampon kullanılarak özetleme işleminden geçirilmiştir. RIPEMD–160 SHA'dan yaklaşık %15 daha yavaş, RIPEMD'nin yarısı hızda ve MD4'den dört kat yavaştır. Sözcük sayısının büyük-önce (big-endian) olduğu bir RISC makinede SHA ve RIPEMD–160 arasındaki hız farkı daha büyük olacaktır [29].

Çizelge 3.3. MD4 algoritmasına dayanan çeşitli özetleme fonksiyonlarının Performansları [29]

Algoritma	Performans (Mbit/s)	
	Assembly	C
MD4	165.7	81.4
MD5	113.5	59.7
SHA-1	46.5	21.2
RIPEMD	82.1	44.0
RIPEMD-128	63.8	35.6
RIPEMD-160	39.8	19.3

3.6. Sayısal İmza Algoritmaları

Asimetrik kripto sistemlerde mevcut olan tüm algoritmalar küçük bir modifikasyonla sayısal imza algoritması olarak kullanılabilirler. Modifikasyondan kasıt, gizli anahtar/açık anahtar değişimidir. RSA, ElGamal ve DSA yaygın olarak kullanılan sayısal imza algoritmalarındandır. Bu algoritmalardan DSA bir devlet birimi tarafından onaylanan ilk sayısal imza algoritmasıdır. ElGamal şifreleme algoritması Bölüm 2'de ayrıntılı bir şekilde açıklanmıştır.

3.6.1. DSA sayısal imza algoritması

1991 Ağustos ayında NIST, DSS'de kullanılmak üzere DSA'yı önermiştir [30]. DSA, bir hükümet tarafından önerilen ilk sayısal imza sistemidir. Bu algoritma ElGamal sisteminin bir değişik bir versiyonudur. DSA, bir özetleme algoritmasının uygulanmasını gerektirir. DSS, bu algoritmanın uygulanmasında SHA'nın kullanılmasını gerektirmektedir.

DSA'nın güvenliği ElGamal'da olduğu gibi ayrık logaritmaların hesaplanması problemine dayanır ve ElGamal'ın güvenliği için söylenenler burada da geçerlidir

İmza üretimi

Bir M mesajının imzası aşağıdaki eşitliklere göre üretilen r ve s sayılarıdır.

$r=(g^k \bmod p) \bmod q$ (3.2)

$s=(k^{-1} (SHA(M) + xr)) \bmod q$ (3.3)

Eş. 3.3.' daki k^{-1}, k' nın q modülüne göre çarpımsal tersidir. SHA(M), güvenli özetleme algoritmasının ürettiği 160 bit'lik bir katarıdır. S'in hesaplanmasında kullanılması için bu skalar bir tamsayıya dönüştürülmelidir.

İmzanın doğrulanması

İmzalanmış bir mesajdaki imzanın doğrulanmasından önce p,q ve g ile imzalayanın açık anahtarı doğrulamayı yapacak tarafa asıllaşmış şekilde duyurulmalıdır. M', r' ve s', sırasıyla M, r ve s'in alıcı tarafından alınmış türevleri olsun. Alıcı önce 0<r'<q ve 0<s'<q olup olmadığını kontrol eder. Eğer bu şartlar sağlanmıyorsa imza reddedilir. Eğer sağlanıyorsa alıcı aşağıdaki hesabı yapar.

$w=(s')^{-1} \bmod q,$ (3.4)

$u_1=(SHA(M')w) \bmod q$ (3.5)

$u_2=((r')w) \bmod q$ (3.6)

$v=(((g)^{U_1})(y)^{U_2}) \bmod p) \bmod q$ (3.7)

Eğer v=r' ise imza doğrulanmıştır. Aksi takdirde mesaj reddedilir.

DSA'nın güvenliği

DSA'nın güvenliği, ElGamal'da olduğu gibi ayrık logaritmaların hesaplanması problemine dayanır. DSA, s denklemi açısından ele alındığında ElGamal

sisteminin özel bir hali olduğu için ElGamal'ın güvenliği ile ilgili olarak söylenenler burada geçerlidir [20].

Yukarıda anahtar üretimi bölümünde q parametresinin sayısı 160 bit olarak belirlenmiştir. Ancak p'nin uzunluğu 512 ve 1024 arasında 64'ün bir katı olacak şekilde değişebilir. 512 bit'lik bir p saldırılara karşı yeterince güvenli olmayacaktır. p için tavsiye edilen uzunluk 768 ya da 1024 bittir [21,27].

DSA'nin başarım karakteristiği

p'nin DSA'da imza üretimi bir modüler üs alma (bu ortalama 240 modüler çarpma demektir), bir modüler üs ters alma (modül 160 bit'liktir), iki 160 bit'lik çarpma, bir toplama gerektirir. DSA'nın avantajı üs alma işlemlerinin önceden yapılabilmesi ve imza üretimi sırasında bunun yapılmasının gerekmemesidir. RSA sisteminde bu mümkün değildir. DSA'da imza doğrulamasının en önemli hesaplaması p modülüne göre 160 bit'lik üslerle üs alma işlemidir. Ortalama olarak bunlar da toplam 480 modüler çarpma gerektirir. Eğer iki üs alma işlemi paralel olarak yapılabilirse ortalama 280 modüler çarpma ile işlem tamamlanabilir [20].

Farklı uzunluklar için imza üretme ve imza doğrulamanın aldığı süreler Çizelge 3.4'de verilmiştir.

Çizelge 3.4. Farklı modül uzunlukları için DSA'nın SPARCII üzerindeki performansı [26]

İşlem	512 bit	786 bit	1024 bit
İmzalama	0.20 saniye	0.43 saniye	0.57 saniye
Doğrulama	0.35 saniye	0.80 saniye	1.27 saniye

3.6.2. RSA algoritması

RSA açık anahtar algoritması 1977 yılında R.Rivest , A.Shamir ve L.Adleman tarafından bulunmuş ve daha sonra açık anahtar kriptolojisine uygun bir biçimde geliştirilmiştir. Bu algoritma açık-anahtar şifreleme ve dijital imza

sistem işlemlerinde güvenli bir şekilde kullanılır. Birçok haberleşme protokolünde de yaygın olarak kullanılmaktadır [31].

RSA algoritması açık-anahtar şifreleme yönteminin temel bir uygulamasıdır. Bu şifreleme yönteminin diğer bir iyi tarafı ise önceden aralarında hiçbir görüşme yapmamış olan alıcı ve vericinin kendi aralarındaki iletişimin güvenli bir ortamda yapılmasıdır.

Görünüşte son derece basit matematiksel ilişkilerle çalışan bu yöntemde iki ayrı anahtar bulunmaktadır. Anahtarlardan birisi kamuya açık, birisi gizlidir. Herkes açık anahtarını yayınlar ve kendisine şifreli bir mesaj göndermek isteyen birisi bu anahtarı kullanarak mesajı şifreler ve gönderir. Gizli anahtar da sadece sahibinde bulunur. Böylece, herkes çözüm için gerekli anahtarı bilmeden, güçlü bir şifreyle mesajları gizleyebilir.

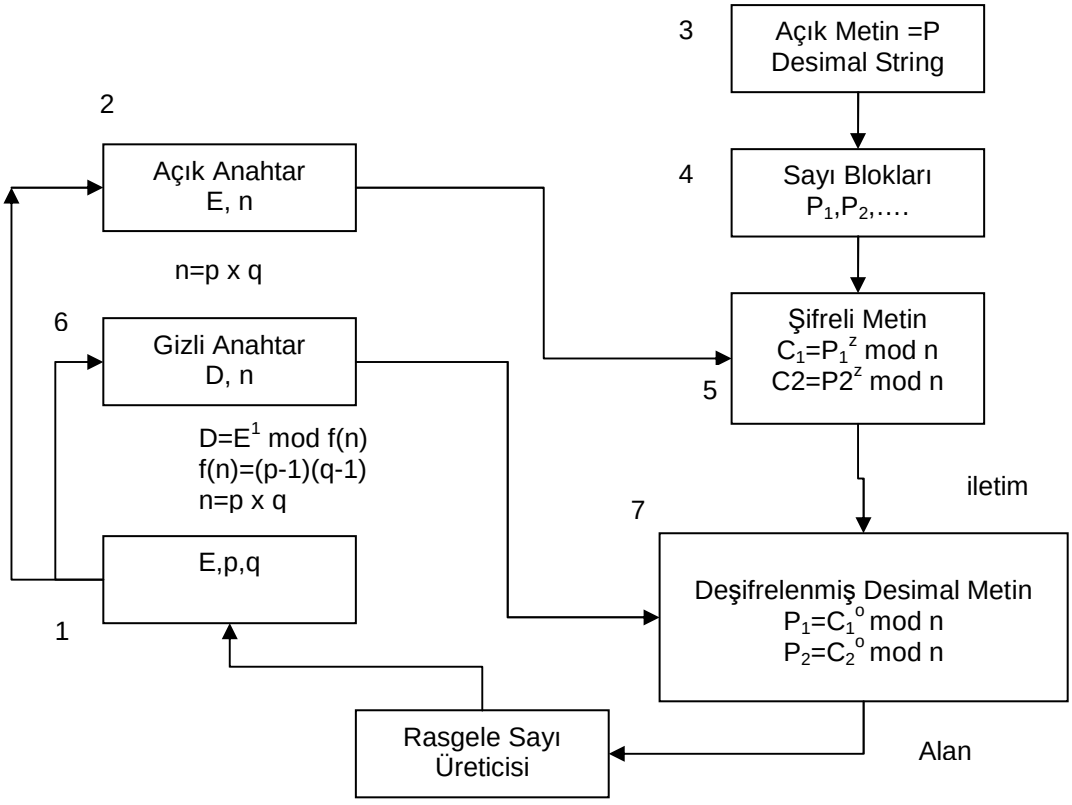
Daha önce hiç karşılaşmamış, birbirini tanımayan kişiler bile birbirlerine gizli mesajlar gönderebilirler. Internet' ten alışveriş yapan birisi, kendisini hiçbir şekilde tanımayan bir web sitesine giderek, sitenin kamuya açık anahtarını alır, kart numarasını bu anahtarla şifreleyerek gönderir. Şifreli bilgiyi gönderen dahil hiç kimse çözemez, sadece web sitesinde bulunan gizli anahtarla gelen kart numarasını web sitesi çözer. Böylece kart sahibi numarasının başkası tarafından okunmayacağından emin olacaktır. Ancak web sitesinin sahte olup olmadığından da emin olmak gerekir. Bu da sertifika yönetimiyle sağlanır.

RSA şifreleme algoritmasının çalışması Şekil 3.7.'de gösterilmiştir. Şekilde gösterildiği gibi öncelikle p ve q olmak üzere iki tane asal sayı üretilir. Bunların birbirleriyle çarpılmasıyla n=p*q'dan n elde edilir. Bundan sonra n sayısından küçük ve (p–1)(q–1) sayısıyla 1 dışında herhangi bir ortak böleni bulunmayan bir e sayısı seçilir.

Daha sonra (e*d=1) sayısının (p–1)(q–1) çarpımına tam olarak bölünmesini sağlayan bir d sayısı bulunur.

Buradaki e ve d değerleri, sırasıyla, açık ve gizli anahtarı temsil etmektedir. Açık anahtarı (n,e) çifti, gizli anahtarı ise (n,d) çifti oluşturur. p ve q sayıları ya yok edilmeli ya da gizli anahtar ile birlikte saklanmalıdır.

Gizli anahtar d sayısının (n,e) sayılarından elde edilmesi zor bir işlemdir. Eğer bir kişi n sayısını çarpanlarına ayırarak p ve q sayılarını elde edebilirse gizli anahtarı da kolaylıkla bulabilir. Bu sebeple RSA sisteminin güvenliği çarpanlarına ayırmak probleminin zorluğu temeline dayanır. Çarpanlarına ayırma işleminin kolay bir yönteminin bulunması, RSA algoritmasının kırılması anlamına gelir [32].



Şekil 3.7. RSA şifreleme algoritmasının yapısı [32]

RSA algoritmasının yapısı

RSA şifreleme algoritmasında şifrelenecek olan açık metni öncelikle [0,n–1] arasında pozitif tamsayı blokları haline dönüştürülür. Şekil 3.7'de ayrıntılı

matematiksel işlemleri görülmektedir.

Bundan sonraki işlem gizli ve açık anahtar çiftlerini elde etmektir. Bunun için p ve q şeklinde çok büyük iki tane birbirinden farklı iki asal sayı bulunur.

$$N=p*q \tag{3.8}$$

$$Z=(p-1)*(q-1) \tag{3.9}$$

Bunlara göre Eş. 3.8. ve Eş. 3.9. hesaplanır. Z ile ortak böleni 1 olacak şekilde bir E sayısı bulunur. Açık anahtar {E,n} olarak belirlenir.

$$D=E^{-1} \bmod Z \tag{3.10}$$

Eş. 3.10. ile de D sayısı bulunur. Gizli anahtar {D,n} olarak belirlenir.

Şifrelenecek mesaj m kabul edilirse bu mesaj ikili (binary) olarak $2^k < N$ olacak şekilde k bit'lik kısımlara ayrılır.

$$m=m(1) + m(2) + m(3) + \ldots + m(n) \tag{3.11}$$

Daha sonra şifreleme için her bir kısma $C(i)=m(i)^E \bmod N$ işlemi uygulanır.

Böylece şifreleme işlemi bitmiş olur. Girişte kullanılan m şifrelenmiş olarak C şeklinde elde edilir.

Deşifreleme

Belirlenen D gizli anahtarının elimizde bulunan şifrelenmiş C metnini çözümlemesi gerekir. Bunun için de şifrelemek için kullanılan bir matematiksel işlem kullanılır [36].

Gizli anahtar {D,n} kullanılarak şifre çözümü Eş. 3.12'de gösterilmiştir.

$$m(i)=C(i)^D \bmod N \tag{3.12}$$

RSA sisteminin güvenliği

RSA sisteminin kırılması birkaç değişik şekilde yorumlanabilir. Sisteme en çok zarar verecek saldırı bir kriptanalistin belli bir anahtara karşı gelen gizli anahtarı bulmasıdır. Bunu başarabilen biri hem şifrelenen bütün mesajları okuyabilir hem de imzaları taklit edebilir. Bunu yapmanın en akla gelen yolu n'nin asal çarpanlara ayrılmasıdır. RSA sisteminin güvenliği çok büyük sayıların asal çarpanlarına ayrılmasının zorluğu varsayımına dayanır. Büyük sayıların çarpanlarına ayrılmasının zorluğu ispatlanmış değildir. Son üç yüzyıl içerisinde Fermat ve Legendre gibi ünlü matematikçiler bu konuda çalışmalar yapmışlardır. Ancak n yeterince büyük seçilirse günümüz teknolojisiyle n'in çarpanlarına ayrılması yeterince uzun süreceği için bu yöntemle d'nin hesaplanması, hesaplama açısından verimsiz olacaktır. Rivest, Shamir ve Adleman'ın orijinal makalelerinde 200 haneli bir sayı olarak seçilmesi tavsiye edilmiştir [33].

RSA'yı kırmanın bir başka yolu da mod n'ye göre e'inci köklerin hesaplanmasıdır. $c=m^e$ olduğuna göre c'nin e'inci kökü m'dir. Böyle bir saldırı, gizli anahtar bilmese dahi şifrelenmiş mesajların deşifre edilmesini yada imzaların taklit edilmesini sağlayabilir. Böyle bir saldırının n'nin çarpanlarına ayrılmasına eşdeğer olup olmadığı bilinmemektedir. Şu ana kadar RSA yöntemini bu yolla kırmaya çalışan bir metoda rastlanmamıştır [33].

Bir başka yöntemde (p-1)(q-1)'in değerinin tahmin edilmesi olabilir. Bu da aynı şekilde n' in çarpanlarına ayrılmasından daha kolay değildir [34].

Rivest, Shamir ve Adleman'ın orijinal makalelerinde bahsedilen bir başka saldırı da n'i çarpanlarına ayırmadan Φ(n)'nin hesaplanmasıdır. Eğer bir kriptanalist Φ(n)'i hesaplayabilirse e'nin Φ(n) modülüne göre çarpımsal tersini hesaplayarak d' yi bulabilir. Ancak bu n' nin çarpanlarına ayrılmasından daha kolay değildir çünkü bu yolla kriptanalist Φ(n)'i kullanarak n'i kolaylıkla çarpanlarına ayırabilir. Bunun için Eş. 3.13.'da ki n bilindiği için (p+q)

hesaplanır.

$$\Phi(n)= n – 1 (p+q) + 1 \quad (3.13)$$

$$(p-q)=[(p+q)^2 – 4n] \quad (3.14)$$

Eş. 3.14.'den de (p-q) bulunur. (p+q) ve (p-q) değerlerinin bilinmesiyle de p ve q hesaplanabilir.

Yine orijinal makalede d'nin hesaplanmasının n'in çarpanlarına ayrılmasından daha kolay olmayacağı iddia edilmektedir. Çünkü eğer d bilinirse n kolaylıkla çarpanlara ayrılabilir.

RSA'nin pratikte kullanımı

Bir kullanıcı diğer bir kullanıcıya imzalı bir mesaj göndermek istediğinde önce mesaj üzerinde bir özetleme fonksiyonu kullanarak bir mesaj özü (digest) üretir. Bu, mesajın sayısal parmak izidir. Daha sonra gönderici taraf, bu mesaj özünü kendi RSA gizli anahtarı ile şifreler. Bu göndericinin karşı tarafa mesajla birlikte göndereceği sayısal imzadır. Alıcı taraf, mesajı ve imzayı alınca gönderici tarafın açık anahtarı ile deşifre ederek mesaj özünü elde eder. Ardından mesajı göndericinin kullandığı özetleme fonksiyonunun aynısından geçirir ve sonucu imzanın deşifrelenmesi sonucunda elde ettiği mesaj özüyle karşılaştırır. Eğer bu ikisi birbirine eşitse, imza kesin olarak doğrulanmış ve alıcı taraf, mesajın doğru göndericiden geldiğinden emin olabilir. Aksi taktirde ya mesajın kaynağı başkasıdır ya da mesaj iletim anında değiştirilmiştir.

RSA ve DSA algoritmaları karşılaştırıldığında, DSA'nın imzalama performansı oldukça iyidir. Bazı hesaplamaların imzalama öncesi yapılmasından dolayı daha avantajlı durumdadır. Ancak aynı durum imza doğrulama için geçerli değildir. RSA'da imza doğrulama işlemleri daha çabuk bir şekilde gerçekleştirilebilmektedir. Verilerin bir kez imzalanıp birçok kez doğrulandığı göz önünde bulundurulursa RSA'nın bir adım önde olduğu

düşünülebilir. Ancak bu durum ihtiyaçlar ve kullanılacak uygulamalar doğrultusunda değişebilir [35].

RSA sisteminin kullanım alanları

RSA sistemi imza doğrulanmasının imzalamadan daha fazla yapılacağı durumlarda uygundur. Ayrıca RSA yavaş bir algoritma olduğu için daha çok kısa mesajların, örneğin gizli anahtarlı şifreleme sistemlerinde taraflar arasında iletilen gizli anahtarın şifrelenmesinde kullanılır.

3.6.3. ECC algoritması

ECC (Eliptic Cilcium Criptograpyh-Eliptik Eğri Kripto sistemler) ilk olarak Washington üniversitesinden Neal Koblitz ve o zaman IBM, Yorktown Heights'ta çalışan Victor Miller tarafından 1985 yılında şifreleme sistemlerine dahil edilmiştir [36].

ECC algoritması ECDLP (Elliptic Curve Discrete Logarithm Problem- Eliptik Eğri Ayrık Logaritma Problemi) üzerine kurulmuştur. Eliptik eğri yaklaşımı standart RSA ve El-Gamal sistemlerinden daha zengin matematiksel prosedürler içermektedir. Bu kripto sistemin temel birimleri eliptik eğri üzerindeki (x,y) noktalarıdır ve Eş. 3.15'de gösterilmiştir

$$y^2=x^3+ax+b \tag{3.15}$$

$$x, y, a, b \in IFp = \{1, 2, 3, \ldots, p-2, p-1\} \tag{3.16}$$

Eş. 3.16' da şifreleme algoritması kısa ana hatları ile anlatılmıştır. Gösterilen her iki formülde "+" ve "–" işaretleri eliptik eğride toplama ve çıkarmayı referans etmektedir. Büyük harfler eğrideki noktaları, küçük harfler ise sayıları temsil etmektedir. Verilecek m mesajında, öncelikle çok büyük bir p sayısı seçilmelidir ve ona uyacak olan Eş. 3.16'da tanımlandığı şekilde eliptik eğri E (IF_p) seçilmelidir. Daha sonra mesaj, eğri üzerindeki p noktasına gömülecektir

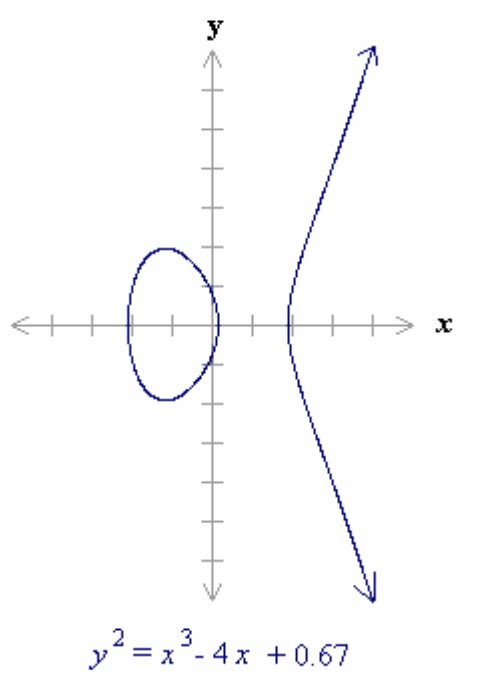
Herhangi bir kripto sistem için en temel bir durum, o sistemin matematiksel olarak kapalı olma özelliğidir bu kuralın eliptik eğrilerde sistemin kapalılığını sağlaması için standart olmayan toplama ve çıkarma işlemlerinin tanımlanması gereklidir.

Eş. 3.15'de ki x,y,a ve b sayıları gerçek sayılardır. a ve b'in her seçiminde farklı bir eliptik ürün çıkmaktadır.

Örneğin a=-4 ve b=0,67 alınırsa

$y^2=x^3-4x+0,67$

eliptik eğri denklemi meydana gelir. Bu eğrinin grafiği Şekil 3.8'de görülmektedir.



Şekil 3.8. Eliptik eğri grafiği [36]

Eğer (x^3 + ax + b) tekrarlamayan çarpanlar içerirse veya buna karşılık eğer ($4a^3 + 27b^2 \neq 0$) ise eliptik eğri ($y^2 = x^3$ + ax + b) grup formunda kullanılabilir. Gerçek sayılar üzerindeki eliptik eğri grubu benzer eliptik eğriler üzerindeki noktalardan oluşmakta ve beraberinde özel bir nokta olan O sonsuzluk

noktası olarak adlandırılır.

T. Yerlikaya, E. Buluş ve D. Arda'nın hazırlamış olduğu "Eliptik Eğri Şifreleme Algoritması Kullanan Dijital İmza Uygulaması" [36] adlı makalede ECC algoritmasının kriptanalizi ve bu algoritma kullanılarak geliştirilen sayısal imza uygulaması incelenebilir. Bu çalışmaya göre ECC şifreleme algoritması, diğer açık anahtar şifreleme algoritmalarının sağlamış olduğu güvenliği daha düşük anahtar değerleriyle sağlayabilmektedir. Bu özelliğinden dolayı ECC şifreleme algoritması kısıtlı bant genişliğinin kullanıldığı kablosuz ağlar için önem arz etmektedir.

3.7. Taklit Edildiği İspatlanabilen Sayısal İmzalar

Geleneksel sayısal imza sistemlerinin taklit edilemezliği karmaşıklık teorisiyle ilgili varsayımlara dayanmaktadır. Örneğin RSA sistemi asal çarpanları büyük olan tamsayıların çarpanlarına ayrılmasının zorluğuna, ElGamal sistemi de ayrık logaritmik hesaplamasının zorluğuna dayanır. Başka bir deyişle güvenlik olası bir saldırganın sayıyı çarpanlara ayırmak ya da ayrık logaritmaları hesaplamak için yeterince zamanı ya da yeterince iyi bir algoritması olmadığı varsayımına dayanır. Yani, en güvenli sistemler bile beklenmedik hesaplama yeteneklerine sahip bir hasım tarafından kırılabilir.

Taklit edildiği ispatlanabilen sayısal imzalar (Fail-Stop Signature-FSS), en iyi geleneksel imza mekanizması kadar taklit edilemezdir. Bunların geleneksel imzalardan farkı, imzanın taklit edilmesi halinde, gerçek imzalayanın bu sahtekârlığı ispatlayabilmesidir. Böylece esas imza sahibi imzanın sorumluluğundan kurtulabilecektir.

3.7.1. Taklit edilme durumunda doğacak sonuçlar

Eğer bir sayısal imza sistemi kırılırsa, gerçek imza sahibi, imzasının taklit edilmesi karşısında savunmasız kalacaktır. Çünkü taklit edilen imza aynen gerçek imzaya benzemektedir. Eğer bu imza mahkemeye sunulursa, mahkeme açık anahtar kullanarak imzanın geçerliliğini teyit edebilir. Böylece

aslında masum olan gerçek imza sahibi sorumlu tutulur. Ancak imzalı mesajın alıcısı tamamen güvendedir. Eğer imzanın açık anahtarla yapılan testi geçip geçmediğini kontrol ettiyse, imza sahte de gerçek de olsa, mahkemede de aynı sonucu alacağını bilmektedir.

Ancak bunun tersi bir durum da ortaya çıkabilir. İmzanın gerçek sahibi, sayısal imza mekanizmasının kanıtlanmamış bir varsayıma dayandığını ileri sürerek, imzasının taklit edildiğini iddia edebilir. Bu durumda da imzayı alan kişinin güvenliği kalmayacaktır. Varsayım tamamen doğru olsa ve herhangi bir imza taklit edilmemiş olsa bile gerçek imza sahipleri mahkemede imzaları, aynen bir önceki durumda imzası gerçekten taklit edilen kişinin yaptığı gibi reddedilebilir. Mahkemenin bu iki durumu ayırt etmesi mümkün değildir.

İşte yukarda sözü edilen durumların meydana gelmemesi için Michael Waidner ve Birgit Pfitzmann 1989 yılında FSS'yi bilim dünyasına sunmuşlardır [37].

3.7.2. FSS imzaların avantajları

FSS sistemlerinde mahkemeler bir imzanın sahte olup olmadığına daima karar verebilirler. FSS'leri geleneksel imzalarsan ayıran özellik budur.

Ayrıca, ilk taklitten sonra, sistemin diğer elemanları ve sistemi işleten imza sisteminin kırıldığını bilir. Böylece daha fazla zarar gelmeden bu sistem durdurulabilir ya da güvenlik parametresi artırılabilir. Zaten bu imza mekanizmalarının İngilizce adı (Fail-Stop) bu anlama gelmektedir. Hata toleransında (fault-tolarance) bu ad, hata halinde sabit bir durumda duran sistemlere verilir.

Tabii, taklitlerin belirlenebilmesi ne kadar güzel olsa da bu bütün problemleri çözemez: Eğer sahte imzanın alıcısı, sahtekârın kendisi değilse, bu imzaya güvenmiş demektir. Eğer imza taklit olduğu için geçersiz olursa, alıcı uygulamaya bağlı olarak belli bir zarara uğrayacaktır. Eğer zarar finansal ise sigorta gibi bir güvenceye başvurulabilir. Geleneksel imzalardan farklı olarak

sigorta zararın gerçekten ortaya çıkıp çıkmadığını kesin şekilde belirleyebilir [33].

3.7.3. FSS imzaların uygulama alanları

Sayısal Ödeme Sistemleri (Digital Payment System-DPS) FSS'lerin en önemli uygulama alanı olabilir. Böyle bir sistemde müşteriler FSS, banka ise geleneksel sayısal imzalar kullanır. Bu durumda müşteriler şartsız olarak güvencededir. Yani kriptografik varsayımlara güvenmek zorunda değildirler. Çünkü müşteriler FSS'lerin imzalayanı, geleneksel imzaların alıcılarıdırlar ve her iki rolde de şartsız olarak güvencededirler [33].

Burada bankanın şartsız güvencede olan taraf olma gereğinin sebebi bankanın güçlü taraf olmasıdır. Banka geleneksel imza mekanizmasını ve güvenlik parametresini uygun şekilde seçerek kendi güvenliğini sağlayabilir. Ayrıca, kriptografik varsayımın ne kadar güvenilir olduğu konusundaki gelişmelerden haberdar olabilir. Öte yandan, daha çarpanlara ayırmak ve ayrık logaritmaların hesaplanması varsayımlarını duyar duymaz pek çok müşteri bu konuda çekingen kalmayı tercih edecektir.

İkinci olarak eğer imzası taklit edilirse bankanın bir kanıta ihtiyacı olmaz: Sistemi işleten bankadır ve sistemi kendisi durdurabilir.

Üçüncü olarak, banka yalnızca geleneksel imzaları kullanan bir sayısal ödeme sistemine göre daha avantajlı olacaktır. Çünkü sahtekârlığın inkar edilemez şekilde kanıtlanabilmesi sistemin kanuni kabul edilebilirliğini artırabilir.

Aslında bu yalnızca bankalar için değil büyük bir kurumun birçok müşterisi olduğu ve imzanın iki müşteri arasında değil de kurum ile müşterileri arasında alınıp verildiği bütün durumlarda geçerli olacaktır.

4. SAĞLIK OCAĞI YÖNETİM SİSTEMİ

Sağlık sektörü bilgiye dayalı iş sahalarının en önemlilerinden birisidir. Daha iyi sağlık hizmeti üretebilmek için gerekli bilgi ve verilerin toplanması, kullanılması, paylaşılabilmesi ve bilgi üretiminin standart yöntemlerle gerçekleştirilmesi üretilen bilgiden en üst düzeyde yararlanmayı sağlar [38].

Hizmet türlerinin ve rollerinin çeşitliliği nedeniyle sağlık hizmeti sunan kurumlarda, hizmet üretiminin ve üretim yönetiminin planlanmasında ihtiyaç duyulan bilgiye erişim, oldukça zor ve karmaşık bir süreç gerektirmektedir. Bu nedenle bilgi üretimi ve bilgiye erişim yöntemleri en başta ve iyi bir şekilde tasarlanmalıdır.

Tüm sağlık kuruluşlarındaki bilgi sistemleri, bünyesinde hastalara ait gizli ya da özel bilgileri barındırmasının yanı sıra günlük olarak belli miktar para akışını içeren mali bilgileri de barındırmaktadır. Yüksek risk oranına sahip bu verilerin kurum içinden veya dışından oluşabilecek saldırılara karşı korunması için merkezi bir yerde aralıksız çalışan, yüksek güvenirlilik sağlayan ve sürekli bilgi akışını devam ettirebilen bir otomasyon sistemine sahip olunması gerekmektedir [38,39].

Elektronik ortamlardaki bilgilerin geniş kitlelerin erişebileceği bir şekilde bulunması bu bilgilerin risk oranlarını çok daha fazla artırmıştır. Yerel ve geniş alan ağlarını bünyesinde bulunduran sağlık kurumlarında bilgi sistemi uygulamalarının yaygınlaşması veri ve bilgi güvenliği açısından bazı problemleri de beraberinde getirmektedir. Bu ağlardaki veri ve bilgi taleplerinin her geçen gün artması gerek kişisel gerekse kurumsal bilgilerin gizlilik ve mahremiyeti açısından sakınca oluşturmaktadır. Özellikle hasta ve hastalık kayıtlarının gizlilik ve mahremiyeti önem arz etmektedir [38,40]. Güvenliğin önemli bir bileşeni de; medikal kayıtta kimin hangi bilgilere ulaşacağına ilişkin yetkilendirmelerdir. Bu yüzden verilerin korunması ve bu verilere erişimde farklı kişilerin farklı erişim haklarına sahip olmaları gerekmektedir [38,41].

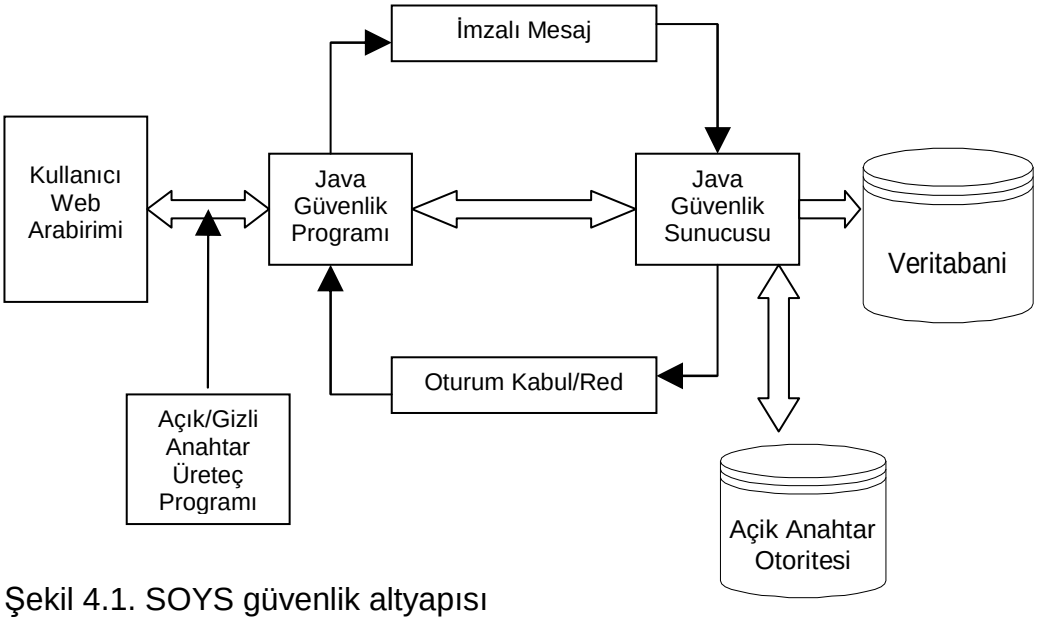
Bu çalışmada; hazırlanan Sağlık Ocağı Yönetim Sisteminde (SOYS) sayısal imza güvenlik teknolojisi kullanılarak geliştirilen güvenlik yazılımı ile işlemlerin etkin ve güvenilir bir ortamda yapılması sağlanmıştır. Oluşturulan kayıt bilgilerinin sayısal imzalı şekilde merkezi sunucuya aktarımı ile hem işlemi gerçekleştiren kişilerin sayısal kimlikleri tespit edilerek kaynaklara erişim kontrol altına alınmış, hem de özet algoritmaları ile bu bilgilerin içeriklerinin kontrol edilmesi sağlanmıştır.

4.1. Sistem Altyapısı

SOYS'de dört temel güvenlik gereksinimi karşılanmıştır. Bunlar;

- Kullanıcıların sisteme girişleri sayısal imza ile güvenli hale getirilmiştir
- Kullanıcı ile sunucu arasındaki veri akışı şifreli olarak gerçekleştirilmeye başlamıştır.
- Mesajların özetlerinin oluşturulmasıyla veri bütünlüğünün kontrolü sağlanmıştır
- Veritabanındaki kayıtlar üzerinde son değişiklik yapan kullanıcının sayısal imzası kaydedilmek suretiyle kayıt seviyesinde güvenlik sağlanmıştır. Bu şekilde aksi iddia edilme ihtimaline imkan vermeyecek şekilde değişikliklerin kim tarafından yapıldığı kaydedilmiştir.

Uygulama sistemindeki güvenlik oluşturulan Java Güvenlik Program ile sağlanmaktadır. Bu program girişi yapılan gizli anahtarın geçerli olup olmadığını kontrol etmektedir. Eğer geçerli ise oturumu kabul etmekte aksi halde bir mesajla girişi reddetmektedir. Java güvenlik sunucusu anahtarların geçerlilik kontrolünü açık anahtar otoritesinden aldığı bilgiye göre yapmaktadır. Kullanıcı tarafından web arabirimi ile yapılan bilgi girişi, güncelleme, silme gibi işlemler java güvenlik programı ile kontrol edilmektedir.



Şekil 4.1. SOYS güvenlik altyapısı

4.1.1. Java güvenlik platformu

Hazırlanan SOYS programında uygulama ve güvenlik bölümleri Java programlama dili ile yazılmıştır.

Uygulama geliştiriciler tarafından kullanılan dillerden biri olan Java ile, mikro ve makro uygulamalar daha kolay, etkin ve güvenli olarak yazılmaktadır. Bunu sağlayan gelişmiş özelliklerinden biri, güvenlik modelidir [42].

Java platformu, uygulama geliştiricilerine bütünleşmiş, esnek ve dağıtık kontrol sağlar. Bu kontrol ile applet, öğe, uygulama ve veriler güvenli olarak oluşturulur. Java güvenlik modeli, kullanıcılar, gruplar ve uygulama elemanları için politika bazlı erişim kontrol modelidir. Bu amaçla kullanılabilen üç uygulama aracı vardır [42] :

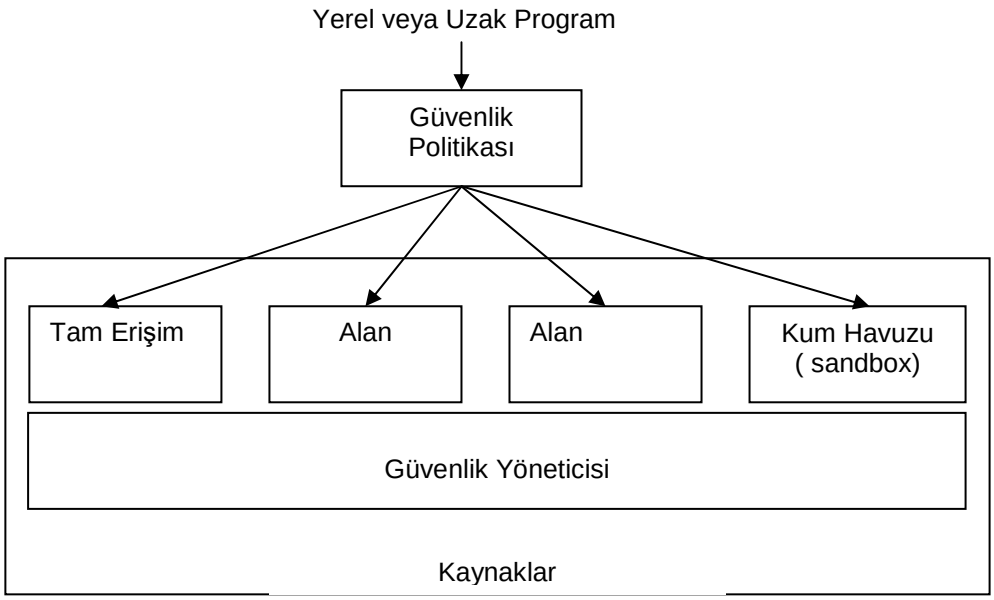
Politika aracı (policy tool), güvenlik politikalarını tanımlayan politika veritabanlarını oluşturmak ve değiştirmek için kullanılan araçtır.

Anahtar aracı (keytool), genel/özel anahtarları yönetmek ve imzalı dijital sertifikaları göstermek, içeri ve dışarı aktarmak için kullanılır.

Jar imzalayacı (jarsigner) ise dağıtık yazılım öğelerinin dijital imzalarının doğrulanması için kullanılır.

Java platformunun güvenlik çatısı (java.security.*) kimlik doğrulamada, X.509 v3 standart sertifikalarını destekler. Ayrıca, Java Kriptolama Uzantısı (Java Cyrptography Extension-JCE), Java Güvenli Soket Uzantısı (JSSE- Java Secure Socket Extension) ve Java Kimlik Doğrulama ve İzin Mekanizması (JAAS- Java Autentication and Authority System) güvenlik özellikleri de Java platformuna bütünleşmiştir. Java platformu, dağıtık işlemlerde güvenli veri servislerinin oluşturulmasına olanak sağlar. Örnek olarak dijital sertifikaların değiştirilmesi, SSL üzerinden iş akışlarının yapılması, programlanabilir genel/özel anahtar oluşturulması, saklanması ve yeniden erişimi, anahtar değişimi verilebilir.

Şekil 4.2' de java güvenlik modeli görülmektedir.



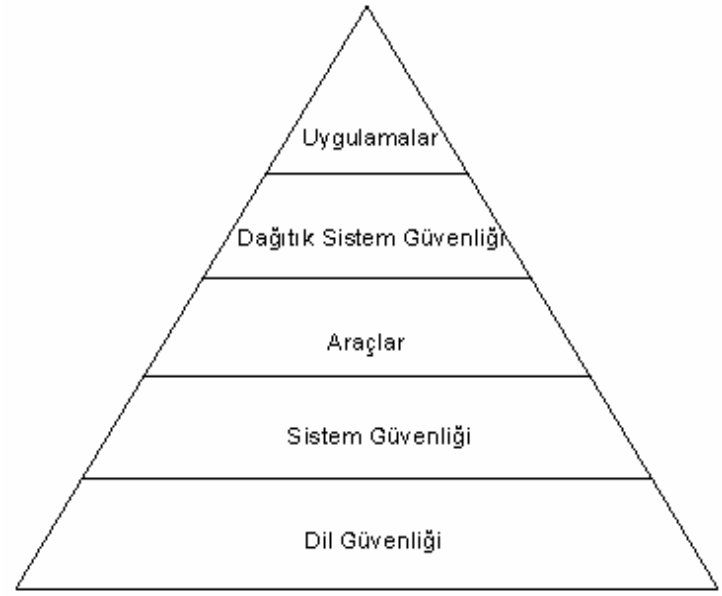
Şekil 4.2. Java platformu güvenlik modeli [42]

Java güvenlik modeli "KumHavuzu" (sandbox) modelini destekler, otomatik olarak kullanıcılar, appletler, uygulamalar ve kaynaklar güvenilir veya güvenilmez olarak tanımlanır. Ayrıca, Java koruyucu alanların (domain) kabul

edilmesi ile tam bir güvenli sistem haline gelmiştir. Karmaşık dağıtık uygulamalarda, alan politikaları belirlenmesi ile geliştiriciler modern güvenlik mekanizmalarını kullanabilirler; örneğin, haklar, şifreleme, dijital imzalar ve sertifikalar gibi.

Java'nın dağıtık güvenliği, akıllı-kartlardan, anabilgisayarlara kadar uzanan bir platform gibi davranır. Geliştiriciler, altyapıdaki yazılımı/donanımı bilmeden program geliştirebilirler.

Java programlama dili ile kullanılan yazılımların avantajı, yazılımlar yerel bilgisayar kaynakları ile etkileşebilir: kalıcı veri saklama gibi. Bilgi işlem geliştiricileri, en fazla kullanılan öğeleri yerelde saklayarak uygulamalarının performansını artırabilirler. Bu kalıcı veri saklama sırasında, Java güvenlik modeli kullanılır ve bu güvenlik modeli Şekil 4.3' de görülen güvenlik temellerini sağlamaktadır.



Şekil 4.3. Güvenlik temelleri [42]

Java platformunun önemli özelliklerinden birisi, tanecikli yapıyı desteklemesidir; Java güvenlik gereksinimlerini uygulamaların en alt düzeyine kadar indirgeme yeteneğini geliştiricilere sunar. Uygulama

geliştirici, kullanıcıları, grupları, programı ve işletim servisleri, nesne ve öğeleri içeren politikaları, hakları ve erişim mekanizmalarını kullanabilir.

Uygulama geliştiricileri Java platformu güvenlik özellikleri ile daha kolay ve dağıtık yapıda uygulama geliştirebilirler. Böylece, uygulamaların yazılım süresi kısaltılır ve uygulamalar daha performanslı çalışabilir.

4.1.2. Şifreleme ve imzalama algoritması

Açık anahtar kripto sistemi olarak hem şifreleme hem de sayısal imzayı destekleyen RSA algoritması kullanılmıştır.

RSA'nın yasal kullanımı ile kolayca hesaplanabilenler;

- Birbirine uyan açık/gizli anahtar çifti oluşturulabilir.
- Eğer açık-anahtar varsa, şifreleme ve imza oluşturma işlemleri yapılabilir.
- Eğer gizli-anahtar varsa, deşifreleme ve imzalama işlemleri yapılabilir.

RSA'nın yasalara aykırı kullanımı ile hesaplaması çok zor olanlar;

- Açık-anahtar varken gizli-anahtarın oluşturulması.
- Gizli-anahtar yokken şifreli mesajın çözülmesi.

RSA şifreleme algoritmasının güvenli olması şu şekilde açıklanabilir: Bu algoritmaya zarar vermek için bilinen en etkin yol n'in asal çarpanları olan p ve q'nun bulunmasıdır. Ancak p ve q çok büyük asal sayılar olduğundan ötürü bulunmaları neredeyse imkansızdır. Ayrıca açık-anahtar (e,n) belli iken d yi, p ve q olmadan bulmak çok zordur.

Uygulamanın gerektirdiği güvenlik seviyesine bağlı olarak kullanılacak bit sayısı ayarlanabilir. Normal şartlarda 1024 bit'lik anahtar kullanılarak şifreleme sisteminin oluşturulması yeterli güvenliği sağlamaktadır. Çizelge 4.1'de RSA algoritmasında farklı bit uzunluklarında anahtar oluşturma ve şifreleme süreleri gösterilmiştir. Test süreleri uygulamanın Pentium4 1.7GHz

Cpu'ya sahip bir bilgisayar çalıştırılmasıyla elde edilmiştir [43].

Çizelge 4.1. RSA algoritmasında farklı bit uzunluklarında anahtar oluşturma ve şifreleme süreleri [43]

Bit Sayısı	**Anahtar Oluşturma Süresi (sn)**	**Şifreleme Süresi (sn)**
64	0.021	0.011
128	0.026	0.013
256	0.083	0.015
512	0.307	0.018
1024	2.985	0.106
2048	50.432	0.766
4096	798.625	18.687

4.1.3. Mesaj özetleme fonksiyonu

Mesaj özetleme fonksiyonu olarak da SHA-2 kullanılmıştır. SHA-1 ile 160 bit'lik özetler oluşturulabilmekteydi. 256 bit'lik SHA-2 kullanılmasıyla bilgilerin güvenliği daha da artırılmış oldu.

Aşağıda özet oluşturma işlemi için tanımlanmış olan hashEkle() fonksiyonuna ilişkin program kodları verilmiştir.

```
private byte[] hashEkle(byte mesaj[])
{
Sha256 s = new Sha256();
s.update(mesaj);
byte hash[] = s.digest();
byte yeniolusanmesaj[] = new byte[hash.length + mesaj.length];
System.arraycopy(hash,0,yeniolusanmesaj,0,hash.length);
System.arraycopy(mesaj,0,yeniolusanmesaj,hash.length,mesaj.length);
return yeniolusanmesaj;
}
```

4.1.4. Anahtar dağıtım tekniği

Güvenlik modülünde 2. bölümde değinilen anahtar dağıtım tekniklerinden biri olan açık-anahtar otoritesi tekniği kullanılmıştır. Tüm kullanıcıların açık anahtarları bir otoritenin gözetiminde merkezi olarak saklanmaktadır. Açık anahtara ihtiyaç duyan kullanıcı (kayıt bilgilerini depolayan sunucu), açık anahtar otoritesinin gözetiminde diğer kullanıcının (tarayıcı üzerinden kayıt işlemlerini yapan istemci) açık-anahtarını elde etmektedir.

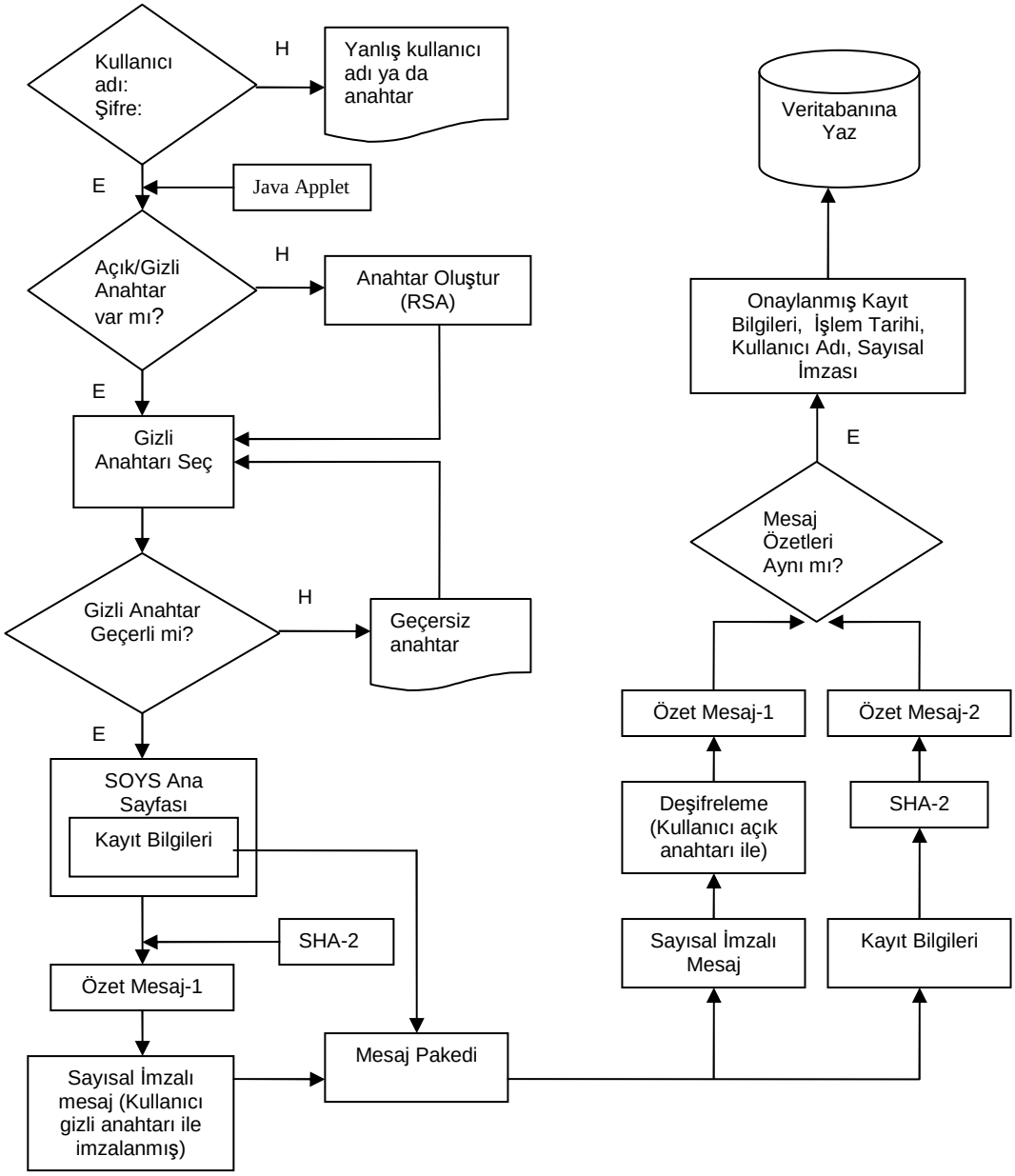
4.2. Sistemin İşleyişi

SOYS programının işleyişi temel olarak şifre giriş, anahtar üretimi, anahtar girişi, hasta ve kurum işlem sayfalarından oluşmaktadır. Bu sayfalar arasındaki işleyiş ve sayısal imzalama işlemi Şekil 4.5'de görülmektedir.

4.2.1. Şifre giriş ekranı

Kullanıcı kodu ve şifre ile sisteme girildiğinde (Şekil 4.4) sunucu tarafında çalışan servlet programı loginkontrol.jsp sayfası ile bu her iki değerin kontrolünü yapar. Herhangi birisinin hatalı olması durumunda ise sisteme giriş engellenir. Bu iki kontrol değeri doğru girilmeden diğer sayfaların adresleri bilinse dahi giriş engellenir ve giriş sayfasına otomatik yönlendirme yapılır. Kullanıcı kodu ve şifre doğru ise açık anahtar giriş modülüne geçilir.

Şekil 4.4. Şifre giriş ekranı



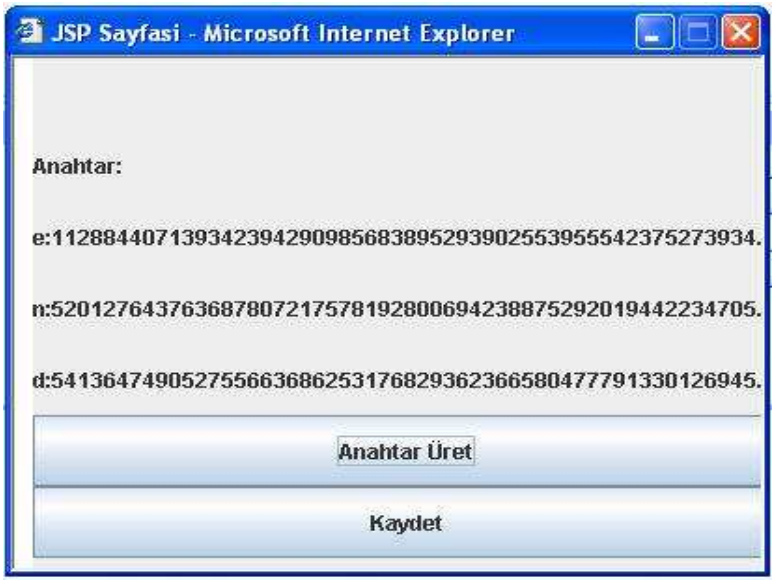
Şekil 4.5. SOYS'nin işleyiş akış diyagramı

4.2.2. Anahtar oluşturma ve kaydetme ekranı

Bu bölümde eğer kullanıcının bir anahtar çifti yok ise sunucu programı kullanıcı açık-gizli anahtar çifti oluşturma ekranına kullanıcıyı yönlendirir. Şekil 4.6'daki ekranda Anahtar Üret butonuna tıklanarak açık ve gizli anahtar oluşturulması sağlanır.

Anahtar üretimini sağlayan generateRSA fonksiyonuna ilişkin program kodları aşağıda verilmiştir.

```
private static BigInteger[] generateRSA(int bitnumber){
int KEYSTRENGHT = bitnumber;
Random random = new Random();
BigInteger TWO = new BigInteger("2",10);
if (random == null) { random = new Random(); }
BigInteger M = new BigInteger(2*KEYSTRENGHT,random);
while (M.mod(TWO).compareTo(BigInteger.ZERO) != 0){
M = new BigInteger(2*KEYSTRENGHT,random);}
int strenght = KEYSTRENGHT; BigInteger P,Q,N;
BigInteger D = new BigInteger(strenght,0,random);
BigInteger Eprime;
P = new BigInteger(strenght,99,random);
Q = new BigInteger(strenght,99,random);
while (!P.isProbablePrime(100)){
P = new BigInteger(strenght,99,random);}
while (!Q.isProbablePrime(100)){
Q = new BigInteger(strenght,99,random);}
N = P.multiply(Q);
BigInteger PHI =
P.subtract(BigInteger.ONE).multiply(Q.subtract(BigInteger.ONE));
do {  do { D = new BigInteger(N.bitLength()/4,random);
} while (D.gcd(PHI).compareTo(BigInteger.ONE) != 0);
BigInteger E = D.modInverse(PHI);
Eprime = (E.add(M)).mod(N.subtract(BigInteger.ONE));
} while(Eprime.gcd(PHI).compareTo(BigInteger.ONE) != 0);
BigInteger Dprime = Eprime.modInverse(PHI);
BigInteger out[] = new BigInteger[3];
out[0] = Eprime;out[1] = Dprime;out[2] = N;
return out;}
```



Şekil 4.6. Açık/gizli anahtar oluşturma ekranı

Yazılım oluşturulan açık anahtarı sunucudaki veri tabanına, gizli anahtarı da kullanıcının kendi bilgisayarında ya da herhangi bir depolama aygıtında yine kullanıcının istediği bir dosya ismiyle kaydeder. Bu isim Şekil 4.7'de görüldüğü gibi standart olarak GizliAnahtar.jsp olarak görünmektedir. Kullanıcının bu gizli anahtarı başka bilgisayarlarda da kayıt yapabilmesi için, ilgili bilgisayara taşıması gerekir. Bu nedenle gizli anahtarın herhangi bir taşınabilir aygıta kopyalanmasında fayda vardır.

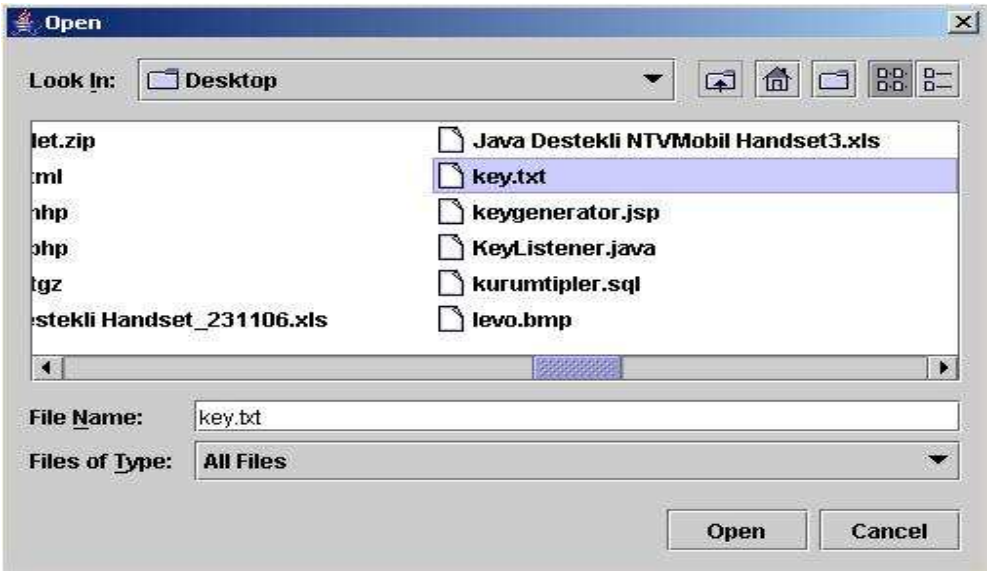
Kayıt işlemini gerçekleştiren fonksiyon kodları aşağıda görülmektedir.

```
<script language="javascript">
function indir(t,str) {
var textWin = window.open(t+"?KEY="+str,"");}
function control() {
var r = document.anahtaruretec.isOk();
if (r == "EVET"){
 var prikey = document.anahtaruretec.getPrivateKey();
indir("GizliAnahtar.jsp",prikey); } else {
setTimeout('control()',500);} }
```

Şekil 4.7. Gizli anahtar kayıt ekranı

4.2.3. Anahtar giriş ekranı

Açık ve gizli anahtarın oluşturulmasından sonra Şekil 4.9'daki Anahtar Seç butonu ile Şekil 4.8'de olduğu gibi gizli anahtar dosyasının bulunduğu kayıt ortamından seçilmesi sağlanır.



Şekil 4.8. Anahtar dosyasını seçme ekranı

Anahtar dosyasının seçilmesinden sonra dosya içeriği ilgili metin kutusuna otomatik olarak aktarılır ve devam butonuna tıklandığında kontrol() şeklinde tanımlanan bir fonksiyon ile girilen gizli anahtarın geçerliliği test edilir. Eğer doğru anahtar girilmemiş ise hata mesajı ile tekrar anahtar girişi yapması sağlanır. Anahtar testi doğrulandığında ise SOYS ana menüsüne giriş sağlanır.

Şekil 4.9. Gizli anahtar giriş ekranı

Aşağıda dosya seçim işlemi ve kontrol() fonksiyonuna ilişkin program kodları sırasıyla verilmiştir.

```
private JFileChooser fileChooser;
public String openFileDialog() {  try {
fileChooser = new JFileChooser();  fileChooser.showOpenDialog(null);
fileChooser.addActionListener(this);
File f = fileChooser.getSelectedFile();
FileInputStream fii = new FileInputStream(f);
int ava = fii.available();
if ( ava > 10000){ return "false:key çok uzun"; }
byte b[] = new byte[ava];  fii.read(b);
String key = new String(b); key = key.trim();
setUserPrivateKey(key); return "true";
} catch (Exception e) {
return "false: "+e;
}
}
```

```
<SCRIPT language="JavaScript">
function kontrol(){
var son =parent.applet.document.cripto.getStatusOfUserPrivateKey();
parent.index.document.sonuc.sonuc.value=son;
if (son == "ok") { parent.index.location = "anasayfa.jsp";
} else if (son == "keytesting"){
parent.index.setTimeout('kontrol()',1000); } else {
parent.index.location = "login3.jsp?hata="+son;}
}  </SCRIPT>
```

4.2.4. Anahtar giriş ekranı

Programın ana menüsü Şekil 4.10'da görülmektedir.

Şekil 4.10. SOYS ana sayfası

Ana sayfadaki ilgili menü seçeneklerinden hasta kaydı ve diğer işlemler yapılabilir. Sisteme ilk girişte kullanıcı gizli anahtarı yazılım içerisinde bulunan

java applete yüklendiğinden dolayı kullanıcı oturumu devam ettiği sürece bu anahtarda hafızada tutulur ve yapılan her işlemde kullanılır.

Kullanıcı tarafından applet vasıtasıyla gönderilen kayıt bilgileri, yazılım vasıtasıyla özet işlemine tabi tutularak bilgilerin bir özeti çıkartılır ve bilgiler kullanıcının gizli anahtarı işe şifrelenerek sayısal imza oluşturulur. Gönderilen bilgi ile birlikte oluşturulan bu sayısal imza da alıcıya gönderilmektedir. Sunucu tarafına gelen metin yeniden özet işlemine tabi tutulmakta ve yeni bir özet çıkarılmaktadır. Metnin arkasından gelen sayısal imza kullanıcının veritabanında saklı tutulan açık anahtarı ile deşifre edilerek metin özeti ile karşılaştırılır; iki özet birbirinin aynı ise metin kabul edilmekte ve bir tabloya gelen metin, sayısal imzası, o anki açık anahtar, kullanıcı kodu ve tarih kaydedilmektedir. İki özet birbirini tutmaz ise kullanıcının isteği bir hata mesajı ile geri çevrilmektedir.

Aşağıda applet vasıtasıyla girişi yapılan bilgilerin kullanıcı gizli anahtarı ile imzalanması ve sunucu açık anahtarı ile şifrelenmesine ilişkin program kodları verilmiştir.

```
public static byte[] Encript(byte message[],BigInteger e,BigInteger n){
return rsa_encode(e,n,message); }
private byte[] encript(byte data[]){
if (CriptoApplet.client_d == null){
System.out.println("CriptoApplet.client_d null");}
if (CriptoApplet.client_n == null){
System.out.println("CriptoApplet.client_n null");}
if (CriptoApplet.serverkey_e == null){
System.out.println("CriptoApplet.serverkey_e null");}
if (CriptoApplet.serverkey_n == null){
System.out.println("CriptoApplet.serverkey_n null");}
 System.out.println("CLIENT D:"+CriptoApplet.client_d.toString(10));
System.out.println("CLIENT N:"+CriptoApplet.client_n.toString(10));
System.out.println("SERVER E:"+CriptoApplet.serverkey_e.toString(10));
```

```
System.out.println("SERVER N:"+CriptoApplet.serverkey_n.toString(10));
data = rsa_encode(CriptoApplet.client_d,CriptoApplet.client_n,data);
System.out.println("IMZALANMIS HALI:["+new String(data)+"]");
data =  rsa_encode(CriptoApplet.serverkey_e,CriptoApplet.serverkey_n,data)
System.out.println("SIFRELENMIS HALI:["+new String(data)+"]");
return data;
}
```

4.2.5. SOYS bölümleri

SOYS uygulama programı altı bölümden oluşmaktadır:

- *Hasta İşlemleri:* Gelen hastalara ilişkin kimlik bilgilerinin kaydedildiği bölümdür.
- *Hasta Kart İşlemleri:* Kimlik bilgileri kaydedilen hastalara ilişkin muayene kart bilgilerinin girildiği bölümdür.
- *Poliklinik İşlemleri:* Kayıtlı kimlik bilgileri bulunan hastaya ilişkin sevk, tedavi işlemleri ile ilgili bilgilerin tutulduğu bölümdür.
- *Doktor Bilgileri:* Kurum içerisindeki doktorlara ait bilgilerinin kayıtlarının tutulduğu bölümdür. Hasta Kart dölümündeki doktor hanesine bilgi bu bölümden alınmaktadır
- *Personel Bilgileri:* Kurum içerisindeki tüm personele ait bilgilerin tutulduğu bölümdür.
- *Kurum Bilgileri:* Gelen hastaların sosyal güvencelerinin bulunduğu kurumlara ait bilgilerin tutulduğu bölümdür. Bu kısma kaydedilen kurumlar Hasta Kart İşlemleri bölümündeki Hizmetler analında listelenmektedir.

Tüm işlem bölümleri dört ekrandan oluşmaktadır. Bunlar; kayıt, liste, güncelleme ve silme ekranlarıdır.

Şekil 4.11. Hasta kayıt ekranı

Şekil 4.12. Hasta listesi

Şekil 4.13. Hasta güncelleme ekranı

Şekil 4.14. Hasta silme ekranı

5. SONUÇ VE ÖNERİLER

Hazırlanan bu çalışmanın ilk üç bölümünde kripto sistemler ve sayısal imza sistemleri detaylı bir şekilde incelenmiştir. Değişik şifreleme teknikleri üzerinde durulmuştur. Kripto sistemler şifrelemede kullanıldıkları anahtara göre iki ana kategoride incelenmiştir.

Simetrik kripto sistemlerde şifreleme ve deşifreleme aynı anahtar kullanılarak gerçekleştirilirken asimetrik kripto sistemlerde ise bu işlemler için açık ve gizli olmak üzere iki ayrı anahtar kullanılmaktadır. Bu anahtarlar kullanılan algoritmaya göre değişik uzunlukta olabilmektedirler.

Simetrik kripto sistemler asimetrik kripto sistemlere göre daha hızlı ve etkili olduğundan uzun mesajların şifrelenmesinde kullanılmaktadır. Asimetrik kripto sistemler ise simetrik sistemlerde kullanılan gizli anahtarın iki taraf arasında güvenli bir şekilde takas edilmesi ve sayısal imzalara temel oluşturması amacıyla kullanılmaktadır.

Sayısal imza, sayısal ortamda kimlik tespitinin yapılabilmesi için geliştirilmiş bir imzalama tekniğidir. Açık anahtarlı kripto sistemlerin ters çalıştırılmasıyla yani mesaj özetinin, göndericinin gizli anahtarıyla şifrelenip alıcıya gönderilmesi ve yine mesaj sahibinin açık anahtarıyla açılıp doğrulama yapılması şeklinde çalışırlar. Bu şekilde hem kimlik tespitinde hem de mesaj bütünlüğünün sağlanmasına yönelik hizmet verir.

Bu çalışmanın sonucunda merkezi bir sağlık ocağı ve buna bağlı alt sağlık ocakları tarafından ortaklaşa kullanılacak bir Sağlık Ocağı Yönetim Sistemi yazılımı ve Güvenlik Modülü hazırlanmıştır. Hazırlanan bu program sistemi sayesinde hem bir bölgedeki tüm sağlık ocaklarının kayıt işlemlerinin tek bir çatı altında toplanması sağlanmış, hem de oluşabilecek güvenlik sorunları kullanılan sayısal imza ve şifreleme sistemi sayesinde büyük ölçüde azaltılmıştır.

Hazırlanan bu uygulama programı Java programlama dili ile applet tekniği

kullanılarak hazırlanmış ve Java' nın Web tabanlı uygulamalardaki esnekliği ve platform bağımsızlığı özelliklerinden yararlanılmıştır. Java' nın üstün güvenlik özelliklerinin ve güvenlik kütüphanelerinin benzer uygulamalar için uygun bir yazılım geliştirme ortamı olduğu görülmüştür. SOYS uygulama ve güvenlik yazılımının java program kodları tezin beraberindeki CD'de verilmiştir.

Sistemin gerçeklenmesi sırasında sayısal imzanın kullanılan bilgisayara bağlı kalmadan her bilgisayarda sorunsuz kullanılabilmesi için güvenli bir şekilde taşınması zorunluluğu ortaya çıkmıştır. Bu problemi aşmak için akıllı kart, biometrik verileri, parmak izi ya da göz retinasından tanıma olanağı sağlayan donanımlardan yararlanılabilir. Ancak bu da varolan teknoloji, maliyet ve uygulanabilirlilik ölçütleri ışığında incelenmelidir.

KAYNAKLAR

1. Garfinkel S., Spafford G., "WEB Security & Commerce", ***O'Reilly & Associates Inc***., Cambridge, 43-45 (1997).

2. Anbar, A., "Veri Transferi Ve İşlem Güvenliğinin Sağlanmasında Kullanılan Şifreleme Yöntemleri Ve Sayısal İmza", **İş-Güç Endüstri İlişkileri ve İnsan Kaynakları Dergisi**, 6 (2): 12 (2004).

3. Saka, Y., "Bilgisayar ağ güvenliği ve şifreleme", Yüksek Lisans Tezi, ***Muğla Üniversitesi Fen bilimleri Enstitüsü***, Muğla, 3-7 (2000).

4. Yalçınkaya, L., "Malzeme Sipariş Sistemi Yazılımı ve Sayısal İmza Uygulaması", Yüksek Lisans Tezi, ***Gebze Yüksek Teknoloji Enstitüsü Mühendislik ve Fen Bilimleri Enstitüsü***, Gebze, 5-7(2001).

5. Doğan, A. Y., "SFINKS Dizi Şifreleme algoritmasının VHDL ile Yazılımı ve FPGA Üzerinde Gerçeklenmesi", ***İstanbul Üniversitesi Elektronik Mühendisliği Bölümü,*** 3-4 (2006).

6. Dahlgren, A., Jönsson, O., "IPSec, the Future of Network Security", Master of thesis, ***Göteborg University School of Economics and Cpmmertial Law***, 32-35 (2000)

7. Frösen, J., "Practical Cryptosystems and Their Strength"***, Proceedings of HUT Seminar on Network Security***, Helsinki University of Technology***,*** Helsinki, 38-45 (1995).

8. Smith, E. R., "Internet Cryptography", ***Addison-Wasley***, Massachusetts, 112 (2000).

9. Şahin, A., Buluş, E., Sakallı, M., T., "Modern Blok Şifreleme Algoritmalrının Gücünün İncelenmesi", ***Trakya Üniversitesi Bilgisayar Mühendisliği Bölümü***, 22100, Edirne, 2-6 (2005).

10. Rivest, R., "The RC5, RC5-CBC, RC5-CBC-Pad and RC5-CTS Algorithms", ***RSA Data Security Inc. RFC2040***, Cambridge, Londra, 12 (1996).

11. Çenesiz, F., Soğukpınar, İ., " Kurumsal Ağ Güvenliğinde Sayısal İmza Kullanımı: Tasarım ve Uygulama", ***5.Bilgisayar Ağları Sempozyumu BAS2000*** *,* Ankara, 198-207 (2000).

12. Yerlikaya, T., Buluş, E., "RSA Şifreleme Algoritmasının İncelenmesi ve Kriptanaliz", ***Trakya Üniversitesi Bilgisayar Mühendisliği Bölümü,*** 2-6 (2003).

13. Diffie, W., Hellman, M., E., "New Directions in Cryptography," ***IEEE Transactions on Information Theory*****,** 22 (6): 644-654 (1976).

14. ElGamal, T., "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", ***Hewlet Peckard Labs***, ***Springer-Verlag***, 10-18 (1985).

15. Stinson, D. R., “Cyrptography Theory and Practice”, ***CRC Press***, Florida, 182-190 (1995).

16. William, S., “Network And Interwork Security and Practice”, **Prientice-Hall Inc.,** NewJarsey, 202-210 (1995).

17. Weiner, M. J., “Cryptanalysis of Short RSA Secret Exponents”, ***IEEE Transac-tions on Information Theory***, 36(2): 553-558 (1990).

18. Çenesiz, F., Soğukpınar, İ., “Ağ Tabanlı Yazılımlarda Sayısal İmza ile Güvenlik Artırımı”, ***Bilişim 2000 Etkinlikleri***, İstanbul, 2-3 (2000).

19. Çenesiz, F., “Kurumsal Ağlarda Sayısal İmza İle Güvenlik”, Yüksek Lisans Tezi, ***Gebze Yüksek Teknoloji Enstitüsü Mühendislik ve Fen Bilimleri Enstitüsü,*** Gebze, 19 (2001).

20. Rabin, M. O., “Probabilistic Algosirthm for Testing Primality,” ***Journal of Number Theory***, 12 (1): 128-138 (Şubat 1980).

21. Dayıoğlu, B., “Bilişim Güvenliği”, ***Pro-G Bilişim Güvenliği ve Arş. Ltd.***, Ankara, 8-11 (2003).

22. Schneier, B., “Applied Cryptography Protocols Algorithms”, Second Edition, ***John Wiley & Sons Pres***, New York, 195 (1996).

23. Menezes, A., Oorschot, P., V.,Vanstone, S., "Handbook of Applied Cryptography", ***CRC Pres***, USA, 122-128 (1996).

24. Popek, G. J., Kline, C. S., “Encryption and Secure Computers Networks”, ***ACM Computer Surveys***, 28 (1): 335-356 (1979).

25. Kohnfelder, L., “Towards a Practical Public-Key Cryptosystem”, Bachelor’s Thesis, ***Massachusetts Institute of Technology Dept. of Electrical Engineering and Computer Science***, 58-60 (1978).

26. Solvay R., Strassen V., “A Fast Monte Carlo Test for Primality”, ***SIAM Journal on Computing,*** 6: 84-85 (1977).

27. Fiat A., Shamir A., “How to Prove Yourself: Practical Solutions to Identification and Signature Problems”, **Springer-Verlag**, New York, 186-194 (1987).

28. Chaum, D., “Security without Identification: Transaction Systems to Make Big Brother Obsolete”, ***Communications of the ACM***, 28 (10): 1030-1044 (1995).

29. Chaum D., Antwerpen H., V., “Advances in Cryptology - CRYPTO '89: Proceedings-Undeniable Signatures”, ***Springer***, Berlin, 440-443 (1990).

30. Aslan, G. B., “Sayısal İmza Sistemlerinin İncelenmesi”, Yüksek Lisans Tezi, ***İstanbul Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü,*** İstanbul, 58 (Şubat 1999).

31. Tora, M., Soyer, D., Naneci, E., “E-devlet Uygulamalarında Güvenlik ve güvenirlik Yaklaşımları”, ***TBD Kamu-BİB Bilişim Platformu VIII***, Antalya, 25-26 (25-28 Mayıs 2005).

32. Schneider, B., “Applied Cryptography: Protocols, Algorithms an Source Code in C”, Second Edition, ***John Wiley & Sons Pres,*** New York, 510 (1995).

33. Rivest R., Shamir A., and Adleman L., “A Method for Obtaining Digital Signature and Public-Key Cryptosystems”, ***Communications of the ACM***, 21:120-126 (1978).

34. Menezes, Alfred J., Van Ooorschot, Paul C. And Vanstone S. A., “Handbook of Applied Cryptogrsphy”, ***CRC Press***, 433-434 (1997).

35. Saygı, Z., Yeşil, S., “Açık Anahtar Altyapısı Konusunda Araştırma, Geliştirme ve Uygulamalar”, ***1. Ulusal Elektronik İmza Sempozyumu,*** Ankara, 28-34 (7-8 Aralık 2006).

36. Yerlikaya, T., Buluş, E., Arda, D., “Eliptik Eğri Şifreleme Algoritması Kullanan Dijital İmza Uygulaması”, ***Trakya Üniversitesi Müh-Mimarlık Fakültesi Bilgisayar Mühendisliği Bölümü***, 22100, Edirne, 1-5 (2005)

37. Pfitzmann B., Waidner M., “Fail-stop Signatures and Their Application”, ***9th Worldwide Congress on Computer and Communications Security***, Paris,145-160 (1991).

38. Çetin, M., Aydos, A., “Elektronik Sağlık Kayıtları Güvenliğinde IEEE 802.1x Standardının Kullanılması”, ***1. Ulusal Elektronik İmza Sempozyumu***, Ankara, 136-141 (2006).

39. Ünüvar N., “Bakanlığa Devredilecek Sağlık Birimlerinin Bilgi Sistemleri Konulu Genelge”, ***Sağlık Bakanlığı Bilgi İşlem Daire Başkanlığı Sayı:2005/23***, ***Ankara***, 1 (2005).

40. Ünüvar, N., “Veri Güvenliği Konulu Raporu”, ***Sağlık Bakanlığı Bilgi İşlem Daire Başkanlığı Sayı:2005/153***, ***Ankara***, 5-11 (2005).

41. T. C. Sağlık Bakanlığı, “Hasta Hakları Yönetmeliği”, ***Resmi Gazete***, ***Ankara*** , 23420: 5-9 (1998).

42. Çetinkaya, S., “Java2 Güvenlik Modeli”, ***Türkiye’de İnternet Konferansı***, İstanbul, 2-4 (2002)

43. Ülker, E., Fındık, O., Arslan, A., İşcan, H., “Selçuk Üniversitesi Sayısal İmza Uygulaması”, ***1. Ulusal Elektronik İmza Sempozyumu***, Ankara, 123-129 (2006).

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : KIRIMLI, Meryem

Uyruğu : T.C.

Doğum tarihi ve yeri : 05.05.1981 Suluova

Medeni hali : Bekar

Telefon : 0 (358) 417 42 24

e-mail : meryem_kirimli@hotmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Lisans	Gazi Üniversitesi/ Bilgisayar Sistemleri Öğretmenliği	2002
Lise	Suluova Süper Lisesi	1998

İş Deneyimi

Yıl	Yer	Görev
2002-2005	Merzifon Anadolu Kız Meslek ve Meslek Lisesi	Bilgisayar Öğretmeni
2005-2006	Amasya Bilim ve Sanat Merkezi	Bilgisayar Öğretmeni

Yabancı Dil

İngilizce