ELİPTİK EĞRİ KRİPTOGRAFİSİ

Özge ÇETİN

YÜKSEK LİSANS TEZİ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MAYIS 2006
ANKARA

Özge ÇETİN tarafından hazırlanan ELİPTİK EĞRİ KRİPTOGRAFİSİ adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

Yrd.Doç.Dr. Erkan AFACAN
Tez Yöneticisi

Bu çalışma, jürimiz tarafından Elektrik Elektronik Mühendisliği Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir.

Başkan : Prof.Dr. Erdem YAZGAN

Üye : Doç.Dr. Çiğdem Seçkin GÜREL

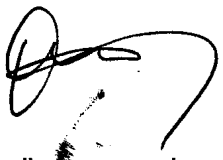
Üye : Yrd.Doç.Dr. Erkan AFACAN

Tarih : 26/04/2006

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.



Özge ÇETİN

ELİPTİK EĞRİ KRİPTOGRAFİSİ

(Yüksek Lisans Tezi)

Özge ÇETİN

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
Mayıs 2006

ÖZET

Bu tezde, eliptik eğrilerin matematiksel temelleri incelenmiş ve bir eliptik eğri şifreleme sisteminin simülasyonu gerçekleştirilmiştir. Ayrıca, eliptik eğri şifreleme sistemlerinin kripto analizine de değinilmiştir. Eliptik eğri kriptografisinin algoritması RSA ve DSA gibi diğer açık anahtarlı kripto sistemlerinin algoritmaları ile karşılaştırılmıştır. Açık anahtarlı şifreleme sistemlerinde, algoritmanın kullandığı matematiksel yöntemlerin kullanılan algoritmanın uzunluğundan daha önemli olduğu görülmüştür. Eliptik eğri şifreleme sisteminin simülasyonu için C++ ile bir program yazılmıştır.

Bilim Kodu : 905.1.011
Anahtar Kelimeler : Eliptik eğriler, Kriptografi, RSA, Özet fonksiyonu
Sayfa Adedi : 96
Tez Yöneticisi : Yrd.Doç.Dr. Erkan AFACAN

ELLIPTIC CURVE CRYPTOGRAPHY
(M.Sc. Thesis)

Özge ÇETİN

GAZİ UNIVERSITY

INSTITUTE OF SCIENCE AND TECHNOLOGY

May 2006

ABSTRACT

In this thesis, the mathematical foundations of elliptic curves are studied and the simulation of an elliptic curve crypto system is realized. Also, the cryptanalysis of elliptic curve crypto systems are covered. The algorithm of the elliptic curve cryptography is compared with the algorithms of other public key crypto systems such as RSA and DSA. In public key cryptography, the mathematical methods behind the algorithms are found to be more important than the length of the algorithm. For the simulation of the elliptic curve crypto system a program is written in C++.

Science Code : **905.1.011**
Key Words : **Elliptic curves, Cryptography, RSA, Hash function**
Page Number : **96**
Adviser : **Yrd.Doç.Dr. Erkan AFACAN**

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren Sayın Hocam Yrd.Doç.Dr. Erkan AFACAN'a, ayrıca çalışmalarımda beni hep destekleyen sevgili aileme ve manevi destekleriyle beni hiçbir zaman yalnız bırakmayan çok değerli Amirim Hv.Mu.Alb. H.Fehmi KIRCA ve arkadaşlarım Berna GÜNEŞ ve Ömer Faruk AYRANCI'ya teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

RESİMLERİN LİSTESİ

Resim **Sayfa**

SİMGELER VE KISALTMALAR

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

Simgeler	**Açıklama**
GF(p)	Galois cismi
φ(n)	Euler fonksiyonu
R	Gerçel sayılar
Q	Rasyonel sayılar

Kısaltmalar	**Açıklama**
AT&T	Amerika Telefon ve Telgraf Merkezi
C	Şifreli metin (ciphertext)
D	Deşifreleme (decipher (decode))
DSA	Dijital İmza Algoritması
DSE	Veri Şifreleme Standardı
E	Şifreleme (encipher (encode))
ECC	Eliptik Eğri Kriptografisi
HMAC	Özetlenmiş Mesaj Doğrulama Algoritması
IDEA	Uluslar arası Veri Şifreleme Algoritması
IP	Internet Protokolü
K	Anahtar (key)
MAA	Mesaj Doğrulama Algoritması
MAC	Mesaj Doğrulama Kodları
MCC	Mesaj Kontrol Karakteri
MD5	5. Mesaj Derleme Algoritması
NIST	Ağ Sistemleri ve Bilgi Teknolojisi
P	Açık metin (plaintext)

Kısaltmalar	Açıklama
PKDS	Açık Anahtar Dağıtım Şeması
PKS	Açık Anahtarlı Kripto Sistemleri
RC5	5. River Şifreleme
RSA	Ronald Rivest, Adi Shamir, Leonard Adleman
SHA	Güvenli Özet Algoritması
SAFER	Güvenli ve Hızlı Şifreleme Yöntemi

1. GİRİŞ

Günümüzde, teknolojinin gelişimiyle birlikte, bilgisayarlar ve internet hayatımızda çok büyük yer tutmaya başlamıştır. Daha çok insanın interneti kullanmaya başlamasından beri internet üzerinden işlemler yapmak kaçınılmaz bir hal almıştır. Bunun en önemli sonucu olarak e-ticaret büyük bir önem kazanmıştır. İnternet Protokol (IP) ağlarındaki sorunlardan biri güvenliktir. Güvenliği sağlamanın yolu da şifreleme ve kimlik denetiminden geçmektedir. E-ticaret ve bankacılık sisteminin gelişimi ile birlikte bu sistemlerin güvenliğinin sağlanması için şifreleme algoritmaları kullanılmaya başlanmıştır.

Şifreleme ve şifre çözme dönüşüm fonksiyonlarının tek ve aynı anahtar kullandığı simetrik kripto sistemler, hızlı ve birçok açıdan etkin olmalarına, mutlak ve koşulsuz güvenlik sunabilmelerine karşın, tüm sistem güvenliğinin kullanılan anahtarla belirlenmesi bu sistemlerin en zayıf yanını oluşturmaktadır.

Bilgisayar bilim ve teknolojisinin eriştiği yüksek düzey göz önüne alındığında, simetrik kripto sistemlerin mutlak biçimde korumak zorunda oldukları anahtarların koruma ve dağıtım maliyetinin ne kadar yüksek ve koruma işleminin ne kadar zor olduğu görülebilir. Sadece bu nedenle, karşılıklı haberleşme içinde olan iki tarafın güvenli dağıtım kanalları oluşturması özellikle güncel bankacılık sisteminde yaygın görülen bir örnektir.

Öte yandan, şifreleme ve şifre çözme dönüşüm fonksiyonlarının kullandıkları anahtarlar birbirinden ayrılarak anahtar güvenliği sorunu kesin biçimde çözülebilir. Anılan çözüm, anahtarların farklılığı nedeniyle Asimetrik Kripto Sistemi olarak bilinen ve ilk kez 1976'da Diffie ve Hellman (El-Gamal şifreleme algoritması) tarafından belirlenen yeni bir dönüşüm tekniğiyle elde edilmektedir. Şifreleme ve şifre çözme dönüşüm fonksiyonlarının birbirinden farklı anahtarlar kullanması, şifreleme anahtarının herkes tarafından bilinen

açık bir anahtar olmasıyla sonuçlanır. Şifre çözme anahtarı ise sadece yetkili alıcı tarafından bilinen gizli anahtar niteliğini kazanır. Şifre anahtarı halka açık tutulduğu için, asimetrik şifreleme algoritmaları aynı zamanda Açık Anahtarlı Kripto Sistemleri (Public Key Cryptosystem-PKS) olarak da bilinir.

Bu çalışmada asimetrik şifreleme algoritmalarının en önemlisi ve günümüzde de sıkça kullanılan, Eliptik Eğri Şifreleme algoritması (ECC) incelenmiştir.

İkinci bölümde, diğer bölümlerde anlatılacak olan eliptik eğri ve diğer şifreleme yöntemlerinde kullanılan karmaşık matematiksel yöntemlerin temelini oluşturan sayı teorileri hakkında yardımcı özet bilgiler verilmektedir [1-2].

Üçüncü bölümde, kriptolojide kullanılan temel kavramlar ve kriptolama güvenliği ile kriptoanaliz konusunda genel bilgi verilmektedir [3].

Dördüncü bölümde, Gizli Anahtarlı Kripto Sistemlerinden Blok ve Dizi Şifreleme yöntemi incelenmektedir. Daha sonra, eliptik eğri kripto sisteminin de içinde bulunduğu açık anahtarlı kripto sistemlerin genel ilke ve karakteristikleri açıklanmaktadır [4]. Açık anahtarlı kripto sistemlerden RSA ve El-Gamal'ın temel yapısından ve sayısal imza uygulamalarından bahsedilmektedir.

Beşinci bölümde, eliptik eğriler hakkındaki genel teoremlerden sonra, eliptik eğrinin seçimi, sonlu cisimlerdeki eliptik eğrilerin genel karakteristikleri, eliptik eğri üzerindeki noktaların oluşturacağı Grup'un elde edilmesi incelenmektedir.

Altıncı bölümde, hazırlanan tezin temelini oluşturan Eliptik Eğri Kriptografisi konusu incelenmektedir. Eliptik Eğri kullanarak açık metinlerin şifrelenmesi/şifresinin çözülmesi ve sayısal imza uygulamaları anlatılmaktadır.

Yedinci bölümde, eliptik eğri kriptografisi sayısal imza uygulamasında sıkça kullanılan özet (hash) fonksiyonlarının yapısı ve uygulama alanlarından bahsedilmektedir [5].

Sekizinci bölümde, kriptografik protokollerin özellikleri ve şifrelemede kullanılan protokol çeşitleri hakkında açıklama yapılmaktadır.

Son bölümde ise, eliptik eğri kriptografisinin temel özelliklerini simülasyon yöntemini kullanarak elde eden ve Borland C++ 6.0 ile geliştirilmiş programın açıklamasına yer verilmiştir.

2. SAYI TEORİSİ

2.1. Genel Kavramlar

2.1. Tanım

a, b ve n tamsayı ve $n \neq 0$ şartı için, eğer a ve b'nin farkı n'in k katı kadarsa bu şu şekilde gösterilebilir:

$a - b = k * n$
veya
$a \equiv b \bmod n$

2.1. Teorem

a_1, a_2 ve n tamsayı ve $n \neq 0$ şartı için,

$(a_1 \text{ op } a_2) \bmod n \equiv [(a_1 \bmod n) \text{ op } (a_2 \bmod n)] \bmod n$

denkliği gösterilebilir, burada op, " + " veya " * " şeklinde bir operatör olabilir.

Bir $a = b \bmod n$ eşitliği, a ve b aynı n ile bölündüğünde aynı kalanı verdiklerini ifade eder.

Örneğin,

$100 = 34 \bmod 11$

Genellikle $0 \leq b \leq n-1$ dir.

$2 \bmod 7 = 9 \bmod 7$

b'ye a mod n'nin kalanı denir.

Tamsayı modül n ile yapılan bütün aritmetiksel işlemlerde tüm sonuçlar 0 ve n arasında olur.

Toplama : a+b mod n

Çıkartma : a-b mod n = a+(-b) mod n

Çarpma : a*b mod n

- tekrarlanan toplamdan türetilir.
- ne a ne de b sıfır değil iken a*b = 0 olabilir.

2.1. Örnek

2*5 mod 10

Bölme : a/b mod n

- b nin tersi ile çarpmak gibidir: $a/b = a*b^{-1} \bmod n$
- eğer n asal ise $b^{-1} \bmod n$ vardır. $b*b^{-1} = 1 \bmod n$

2.2. Örnek

2*3=1 mod 5 bu nedenle 4/2=4*3 = 2 mod 5 dir.

Tamsayılarla modül n toplama ve çarpma aşağıdaki kurallar ile bir değişmeli halkadır.

Birleşme : (a+b)+c = a+(b+c) mod n

Değişme : $a+b = b+a \bmod n$

Dağılma : $(a+b)*c = (a*c)+(b*c) \bmod n$

Aynı zamanda tamsayı modül n'lerinde, istenirse önce işlem yapılıp sonra modül n'e indirgenebileceği gibi, önce modül n'e göre indirgenip daha sonra işlem yapılabilir.

$a+/-b \bmod n = [a \bmod n +/- b \bmod n] \bmod n$

$(a*b) \bmod n =((a \bmod n)*(b \bmod n)) \bmod n$

Eğer n, p doğal sayısı olmaya zorlanırsa bu form bir Galois cismi olur ve GF(p) ile gösterilir ve tüm tamsayı aritmetiğindeki normal kurallar geçerlidir.

2.2. Asal Sayılar

Asal sayılar, açık anahtarlı kripto sistemlerinde büyük rol oynarlar. Asal sayılarla ilgili karşımıza çıkan en önemli fonksiyonlar, asal bir sayının oluşturulması ve bir sayının asal olup olmadığının test edilmesidir. Asal sayı oluşturma, verilmiş bir $[r_1,r_2]$ tamsayılar aralığında asal sayı bulma işlemidir.

2.2. Tanım

$a^{n-1} \equiv 1 \bmod n$ şartını ve $1< a< n$ şartını sağlayan n tamsayısına a tabanına göre sözde asal (pseudo prime) sayı denir.

n tamsayısı için Euler fonksiyonu $\varphi(n)$, n den daha küçük olan ve n ile aralarında asal olan bütün pozitif tamsayıların sayısını verir.

p asal ise;

$\varphi(p) = p-1$ dir.

n =p*q ve p, q asal sayılar *ise* $\varphi(n) = \varphi(p)*\varphi(q) = (p-1)*(q-1)$ dir.

2.2. Teorem

Eğer p bir asal sayı ve OBEB(a,p)=1 ise p, a tabanına göre bir sözde asal sayıdır.

2.3. Tek Yönlü Fonksiyon

$F: X \longrightarrow Y$

f : x $\longrightarrow$ $f(x)= y$ yalnız ve yalnız aşağıdaki şartları taşıdığı takdirde tek yönlü bir fonksiyondur:

1. $f(x)$ bütün x değerleri için polinomsal zamanda çözümlenebilir olmalıdır.

2. Verilen bir y değeri için x değeri polinomsal zamanda bulunamamalıdır.

Örnek olarak verilirse a^m mod n ≡ x bir modüler üs alma işlemidir ve kolaylıkla yapılabilir, fakat var olan x değerinden m değerini bulmak ayrık logaritma problemine girer ve bunun da hesaplanma süresi polinomsal çözümleme süresinden çok daha uzundur.

2.4. Arka Kapılı Tek Yönlü Fonksiyonlar (Trapdoor One-Way Functions)

Arka kapılı tek yönlü fonksiyonlarda ise tek yönlü fonksiyonlara ek olarak analizciye başka bilgiler verilirse fonksiyon daha kolay tersinir hale getirilebilir.

Örneğin yalnız a^m mod n değerini bilmekten öte buradaki n değerinin iki asal sayının çarpımı olduğunu ve anahtarların bu sayılara bağlı olduğunu bilmek buradan m değerini bulma aşamasında analizciye ipucu vermiş olur.

2.5. Grup Teorisi

Verilen herhangi bir *G* grubu için bu gruba ait elemanların sayısına *G* grubunun mertebesi denir ve *ord (G)* = |G| sembolüyle gösterilir. Eğer *H* grubu *G* grubunun bir alt grubu ise |H| değeri |G| değerini böler. Böylece eğer G grubunun mertebesi bir asal sayıysa G'nin tek alt grubu kendisidir. Bu durumda G grubu çarpmalı olarak yazılabilir.
Grup üzerinde tanımlanan çarpma işleminin aşağıdaki şartları sağlaması gereklidir:

1) G'nin herhangi üç elemanı a,b,c için

a *(b*c) = (a*b)*c

eşitliği sağlanmalıdır,

2) G'nin öyle bir e elemanı vardır ki, G'deki herhangi bir a için

a * e = e * a = a

eşitliği sağlanır (yani e etkisiz elemandır), ve de e, G'de bu özelliği sağlayan tek elemandır,

3) G'deki her a elemanı için öyle bir b elemanı bulmak mümkündür ki

a*b = b*a = e

eşitliği sağlansın. Eğer bu eşitlik sağlanıyorsa b elemanına a elemanının tersi adı verilir.

Yukarıdaki tanımda dikkat edilmesi gereken bir nokta ise işlemimizin değişme özelliği olduğunu varsaymıyor oluşumuzdur. Yani bazı gruplarda öyle iki a ve b elemanı bulmak mümkündür ki a*b ≠ b*a olsun. Öte yandan eğer bir grupta fazladan değişme özelliği de varsa o gruba "Abel grubu" denir. Gruplar sonlu veya sonsuz sayıda eleman içerebilirler.

2.3. Teorem (Fermat teoremi)

p bir asal sayı olsun.

Her a tamsayısı için $a^p \equiv a \bmod p$ denkliği ve p ile bölünmeyen her a tamsayısı için ise $a^{p-1} \equiv 1 \bmod p$ denkliği her zaman doğrudur.

2.6. GF(p)'de Üs Alma İşlemi

1. Birçok kriptolama algoritması;

$b = a^e \bmod p$

ile verilen üs alma işlemini kullanılır.

2.3. Tanım (O notasyonu)

f(x) ve g(x) pozitif gerçel değerli iki fonksiyon olsun. Yeterince büyük her x için,

$f(x) < cg(x)$

eşitsizliğini sağlayacak şekilde bir c pozitif gerçel sayısı bulunabiliyorsa,

f(x) = O(g(x)) veya *f = O(g)*

yazılır.

2. Üs alma işlemi basit olarak bir n sayısı için *O(n)* çarpma olan tekrarlanan çarpmalardır.

3. Daha iyi bir yöntem kare alma ve çarpma algoritmasıdır.

4. Bir n sayısı için sadece $O(\log_2 n)$ çarpma yapılır.

2.7. GF(p)'de Ayrık Logaritmalar

Üs almada ters problem, bir modül p sayısının ayrık logaritmasının bulunmasıdır.

$a^i = b \bmod p$ işleminde a ve *i* verildiğinde b değerini hesaplamak nispeten kolay iken, a ve b değeri verildiğinde i değerini bulmak zor bir problemdir. $i = \log_a b$ ile gösterilir ve bu problem ayrık logaritma problemi olarak adlandırılır. Eğer p asal ise, herhangi bir b! = 0 için her zaman bir ayrık logaritması olan bir α olduğu gösterilebilir.

α'nın ardışıl kuvvetleri mod p ile grup oluşturur.

$\alpha \bmod p$, $\alpha^2 \bmod p$,....., $\alpha^{p-1} \bmod p$ farklıdır ve 1 ile p-1 arasında değer alır.

Bu durumda, α ya ilkel kök denir ve aynı zamanda bulmak nispeten zordur. Herhangi bir *b* tamsayısı ve p'nin ilkel kökü olan α için bir *i* üssü bulunabilir;
$b = \alpha^i \bmod p \quad 0 \le i \le (p-1)$ dir.

Üs *i* ayrık logaritma veya indeks olarak gösterilir.

2.8. En Büyük Ortak Bölen (Greatest Common Divisor)

2.4. Teorem

a ve n tamsayıları için, (a $\in$ {0,1,...n-1}); eğer a ve n aralarında asal iki sayıysa a'nın modül n'e göre yalnız bir tane tersi vardır ve a^{-1} sembolüyle gösterilir.

OBEB (a,n) = 1 $\Leftrightarrow \exists$ *b* $\in$ [a,n-1], 1= a*b mod n, yani b=a^{-1} dır.

a ve b'nin en büyük ortak böleni (a,b) a ve b'nin her ikisini de bölen en büyük sayıdır. Öklid Algoritması iki a ve n (a<n) sayısının en büyük ortak bölenini bulmak için kullanılır, eğer a ve b nin böleni d ise, d a-b ve a-2b yi böler.
Örneğin, OBEB(56,98)'i bulalım.

g_0=98
g_1=56
g_2 = 98 mod 56 = 42
g_3 = 56 mod 42 = 14
g_4 = 42 mod 14 = 0

sonuçta OBEB (56,98)=14 olarak bulunur.

2.5. Teorem (Euler genellemesi)

Her a ve n tamsayısı için;

OBEB (a,n) = 1 ise

$a^{\varphi(n)} \equiv 1 \bmod n$ dir. (2.1)

Buradan eğer a*x ≡ 1 mod n teoremini *OBEB (a,n)*=1 ile çalıştırırsak

$x \equiv a^{\varphi(n)-1} \bmod n$

ifadesi bize a sayısının modül n'e göre tersini verir.

2.6. Teorem (Çin Kalan Teoremi -Chinese Remainder Theorem)

Aralarında asal $n_1,n_2,n_3,...,n_k$ tamsayıları için:

$x \equiv a_1 \pmod{n_1}$
$x \equiv a_2 \pmod{n_2}$
.
.
.
$x \equiv a_k \pmod{n_k}$

ile verilen denklik sisteminin modül n'de, yalnız tek bir çözümü vardır ve $n = n_1n_2...n_k$ dir.

2.3. Örnek

$x \equiv 7 \pmod 8$, $x \equiv 3 \pmod 5$ denklik çiftinin mod (8*5) = mod 40' e göre tek bir çözüm kümesi vardır ve $x \equiv 23 \pmod{40}$.

2.9. Öklid Algoritması (Euclidean Algorithm)

a ve b şeklinde olan iki tamsayının en büyük ortak bölenini aritmetiğin esas teoreminde bahsedildiği gibi çarpanlarına ayırarak ve ortak çarpanların en büyüğünü alarak bulabiliriz. Eğer a ve b büyük sayılarsa bunların asal çarpanlarını bulmak zor olur; bunun sonucunda da en büyük ortak böleni bulmak da zorlaşır. Sayılar teorisinin önemli bir araştırma alanı da büyük tamsayıları daha çabuk çarpanlarına ayırma üzerine araştırmadır. Eğer a ve b nin asal çarpanları bilinmiyorsa, OBEB (a,b) yi bulmak için çabuk bir yol vardır. O da Öklid algoritmasıdır.

Öklid Algoritması şöyle çalışır:

- $a > b$ olmak üzere, a, b'ye bölünür. Bölüm q_1, kalan r_1 olsun

$$a = b * q_1 + r_1$$

- İkinci bölme işlemi gerçekleştirilir. b, r_1'e bölünür ve bölüm q_2, kalan ise r_2 olur.

$$b = q_2 * r_1 + r_2$$

- Üçüncü olarak r_1, r_2'ye bölünür ve bölüm q_3, kalan ise r_3 olur.

$$r_1 = q_3 * r_2 + r_3$$

.

.

.

- Son olarak r_{n-1}, r_n'e bölünür ve bölüm q_{n+1}, kalan ise $r_{n+1} = 0$ olur.

$$r_{n-1} = q_{n+1} * r_n + r_{n+1} \qquad (2.2)$$

- $r_{n+1} = 0$ olduğu için r_n değeri a ve b tamsayılarının en büyük ortak böleni olur. Yani OBEB(a,b) = r_n dir.

Bu algoritmadaki işlemler sonsuza kadar gitmez, çünkü 0 ile a tamsayısı arasında sonlu sayıda tamsayı vardır.

3. KRİPTOLAMANIN TEMEL KAVRAMLARI

Bilgisayar ağlarının ve haberleşme sistemlerinin güvenliğinin sağlanması için kullanılan en önemli işlem, verilerin şifrelenerek anlamsız hale getirilip hedefe gönderilmesi ve hedefte tersi işlem yapılarak tekrar eski hale getirilmesidir. Bir şifreli haberleşme için;

1. Şifreleme algoritması (E)
2. Şifre çözme algoritması (D)
3. Bir anahtar bilgisine (K)

ihtiyaç vardır.

3.1. Terminoloji ve Notasyon

Kriptoloji, latince gizli anlamına gelen *kryptos* ve yine latince sözcük anlamına gelen *logos* kelimelerinin birleşiminden oluşan gizli ve güvenli haberleşme bilimidir. Kriptoloji temelde iki kısımda incelenir. Birincisi kritik bilgilerin yetkisiz kişi ve/veya kurumlardan korunması amacıyla geri dönüşümü mümkün olarak anlaşılmaz hale getirilmesi yani şifrelenmesi için kripto sistemlerinin tasarlanması demek olan kriptografi bilimidir. İkinci kısım ise kodlanmış veya şifrelenmiş olan gizli bilgilerin bulunmasına yönelik çalışmaların yapılması demek olan kriptoanaliz bilimidir.

Kriptolojide daha çok bilginin güvenliği ve gizliliği üzerinde durulmaktadır. Bunun yolu genellikle bilgilerin veya mesajların bir takım transformasyonlara tabi tutulmasıyla olur. Daha sonra bu bilginin tekrar elde edilebilmesi için şifreli metne aynı transformasyonların tersi uygulanır.

Orijinal mesaj burada kısaca m harfiyle, mesajı transformasyona tabi tutma işlemi şifreleme adıyla, ortaya çıkan anlaşılmaz metin ise kısaca c harfi ile

gösterilecektir. Ters transformasyon işleminin şifreli metne uygulanıp tekrar orijinal mesajı elde etmeye yönelik yapılan işleme ise şifre çözme adı verilir.

3.2. Genel Kavramlar

Kriptografi (cryptography): Anlaşılır bir mesajı anlaşılmaz şekle dönüştürme ve mesajı tekrar eski orijinal haline geri dönüştürme prensipleri ve yöntemlerini içeren sanat veya bilimdir.

Açık metin (plaintext-P): Anlaşılır orijinal metin.

Şifreli metin (ciphertext-C): Dönüştürülen metin.

Şifreleyici (cipher): Anlaşılır bir metni, yerlerini değiştirme ve/veya yerine koyma yöntemlerini kullanarak anlaşılmaz şekle dönüştürmek için kullanılan bir algoritma.

Anahtar (key-K): Sadece gönderici ve alıcının bildiği şifreleyici tarafından kullanılan kritik bilgiler.

Şifreleme (encipher (encode)-E): Açık metni bir şifreleyici ve bir anahtar kullanarak şifreli metne dönüştürme süreci.

Şifre çözme (decipher (decode)-D): Şifreli metni bir şifreleyici ve bir anahtar kullanarak açık metne dönüştürme süreci.

Kriptoanaliz (cryptanalysis): Bilgi ve anahtar olmaksızın anlaşılmaz mesajı anlaşılır mesaj olarak geri dönüştürme prensipleri ve yöntemleridir. Aynı zamanda kod kırma (codebreaking) olarak da adlandırılır.

Kriptoloji (cryptology): Kriptografi ve kriptoanalizin her ikisi.

Kod (code): Anlaşılır bir mesajı bir kod kitabı kullanarak anlaşılmaz şekle dönüştürme için bir algoritma.

Şifreleme (Encryption) $c = E_K(m)$.

Şifre çözme (Decryption) $m = D_K(c)$.

E_K, kriptografik sistem olarak bilinen transformasyon ailesinden seçilir. Anahtar denilen K parametresi anahtar uzayından seçilir. Diğer bir deyişle, şifreleme işlemi $E_K(m)=c$ fonksiyonunu sağlayan bire-bir fonksiyondur. E_K fonksiyonunun tersi olan D_K fonksiyonu ise, $D_K(c)=m$ şartını sağlayan şifre çözme işlemini gerçekleştirir.

Burada yer alan bütün transformasyon işlemleri tersinir olduğundan açık bilginin şifreli bilgiden direkt olarak elde edilmesini önlemek için *E* ve *D* algoritmalarının gizli tutulması düşünülebilir. Şifreleme ve şifre çözme algoritmalarının herhangi bir şekilde yetkisiz kişilerin eline geçmesine karşı yalnızca mesajlaşacak kişilerin bilebileceği bir anahtar bilgisi, *K*, kullanılmalıdır. Dolayısıyla, mesajlaşmada önemli olan kriter kullanılan anahtarın gizliliği olacaktır. Sonuçta anahtar gizli tutulduğu halde algoritmalar açık olabilir.

3.3. Kripto Sistemleri

Kripto sistemlerinde kullanılan başlıca terimler kısaca şunlardır; *A* ile gösterilen Alfabe kavramı sonlu sayıda elemanlar kümesidir. Örneğin $A=\{0,1\}$ sık kullanılan ikili (binary) bir alfabedir. *P* ile gösterilen Açık Metin Uzayı (Plaintext Space) ise alfabeden alınmış sonlu sayıda eleman dizilerinden oluşur. Örneğin *P*, 0 ve 1 ler den meydana gelen bit dizilerini içerebilir. *C* ile gösterilen Şifreli Metin Uzayı (Ciphertext Space) ise yine *A* alfabesinden alınmış fakat *P* den farklı bir diziliş gösteren elemanlardan oluşur. *K* ise daha önce bahsettiğimiz Anahtar Uzayını (Key Space) ifade eder. Anahtar yine *A*

alfabesindeki elemanların belli uzunluklarda bir araya gelmiş elemanlarından oluşur [6].

3.1. Tanım

Bir kripto sistemi aşağıdaki şartları sağlayan (*P,C,K,E,D*) beşlisinden oluşur. Burada *E* şifreleme, *D* ise şifre çözme fonksiyonu veya algoritmasını gösterir.

$\forall k \in K$, $D_k \in D$ fonksiyonuna uyan bir $E_k \in E$ fonksiyonu vardır. Öyle ki;

$\forall E_k: P \rightarrow C$ ve $\forall D_k: C \rightarrow P$ ve her $x \in P$ için $D_k(E_k(x)) = x$

Kripto sistemleri genel olarak kullanılan protokole göre iki kategori de sınıflandırılır. Bunlardan birincisi tek ve gizli bir anahtarın ve simetrik bir algoritmanın kabul edildiği gizli anahtarlı kriptografi, ikincisi de biri açık biri ise gizli ve kişiye özel olmak üzere iki anahtarın kabul edildiği ve asimetrik algoritma kullanımı esasına dayanan açık anahtarlı kriptografi sistemidir.

3.4. Kriptolama Güvenliği ve Kriptoanaliz

Şifrelenen metnin ne kadar güvenli olduğu ve çözümlenmesi için yapılacak atak tiplerinin neler olduğunun bilinmesi önemlidir.

Çizelge 3.1'de şifrelenen mesajı çözmek için yapılan atak tipleri ve kripto analistin neler bildiği gösterilmiştir.

3.5. Mutlak ve Hesaplama Güvenliği

İki farklı temel yöntem ile şifreler güvenli olabilir.

1. Mutlak Güvenlik: Bilgisayar gücü ne kadar fazla olursa olsun şifre hiçbir şekilde kırılamaz.

2. Hesaplamaya bağlı güvenlik: Verilen bilgisayar gücü sınırları içinde, hesaplama zamanının çok uzun olmasından dolayı şifre kırılamaz.

Çizelge 3.1. Şifrelenen mesaja karşı yapılan atak tipleri

AtakTipi	Kriptoanalist'in bildiği
Sadece Şifreli Metin (ciphertext only)	Kriptolama algoritması Kodu çözülecek şifreli metin (istatistiksel atak, eksiksiz arama)
Bilinen Düz metin (known plaintext)	Kriptolama algoritması Kodu çözülecek şifreli metin Gizli anahtar ile şifrelenen bir veya daha fazla düz-şifreli metin çifti (Şifreye atak için kullanılır.)
Seçilen Düz metin (chosen plaintext)	Kriptolama algoritması Kodu çözülecek şifreli metin Kriptoanalist tarafından seçilen açık metin, bununla birlikte açık metnin gizli anahtar ile üretilen şifreli hali
Şeçilen Şifreli metin (chosen ciphertext)	Kriptolama algoritması Kodu çözülecek şifreli metin Kriptoanalist tarafından seçilen kuvvetle muhtemel şifreli metin ve karşılığı olan, gizli anahtar ile üretilen çözümlenmiş açık metin.
Şeçilen metin (chosen text)	Kriptolama algoritması Kodu çözülecek şifreli metin Kriptoanalist tarafından seçilen açık metin, bununla birlikte açık metnin gizli anahtar ile üretilen şifreli hali Kriptoanalist tarafından seçilen kuvvetle muhtemel şifreli metin ve karşılığı olan, gizli anahtar ile üretilen çözümlenmiş açık metin.

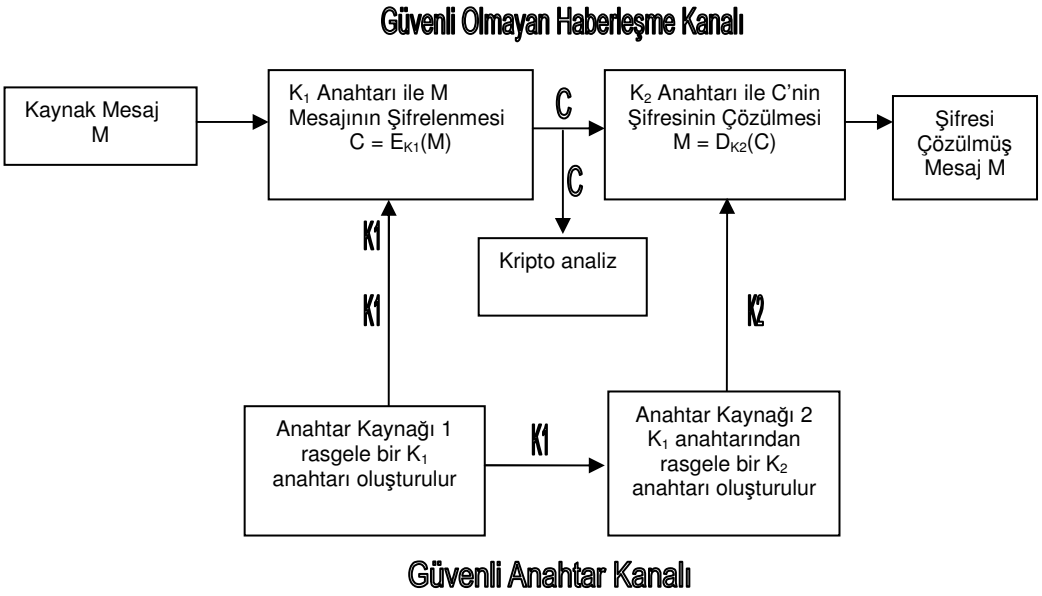
Hesaplamaya bağlı güvenlik için şifreleme algoritması ve kullanılan anahtar uzunluğu önemlidir. Çözümleme süresi için gerekli olacak zaman hesabı ortalama olarak alternatif şifre sayısının yarısı kadardır. Bilgisayar hesaplama gücünü ise paralel mimarili tasarım ile artırmak mümkün olmaktadır.

4. GİZLİ VE AÇIK ANAHTARLI KRİPTO SİSTEMLER

Bu bölüm, gizli ve açık anahtarlı şifrelemeye genel bir giriş mahiyetindedir. İlk önce, gizli anahtarlı kripto sistemlerden bahsedilmiştir. Daha sonra eliptik eğri kripto sistemi dışındaki açık anahtarlı şifreleme yöntemleri için, uygulanabilir olarak gösterilen en bilindik şifreleme/şifre çözme algoritmaları olan DSE, RSA ve El-Gamal algoritmaları incelenmiştir. Daha sonra da açık anahtarlı sistemler için, anahtar dağıtımı ve anahtar dağıtım yönetimleri incelenmiştir.

4.1. Gizli Anahtarlı (Simetrik) Kripto Sistemler

Gizli anahtarlı kriptografik sistemler tarihin ilk devirlerinden beri dünyada kullanımı süregelen kriptografik sistemlerdir. Bu sistemlerde şifreleme algoritması ve şifre çözme algoritması birbirinin tersi şeklindedir. Öncelikle haberleşecek iki grup aralarında gizli bir anahtar tespit ederler. Eğer bu iki grup birbirlerine yakın yerlerde yer almıyorlarsa güvenli bir haberleşme kanalı veya güvenilir bir kurye yoluyla anahtarları birbirlerine ulaştırabilirler. Bir taraf şifreleme algoritmasında girdi olarak açık metin (*P*) ve anahtarı (*K*) uygular, ardından şifreli metin (*C*) yi elde eder ve mesajın alıcısına gönderir. Mesaj alıcısı ise şifre çözme algoritmasının girdileri olarak şifreli metin (*C*) yi ve aynı (*K*) anahtarını kullanır ve ardından çıktı olarak açık metin (*P*) yi elde eder.

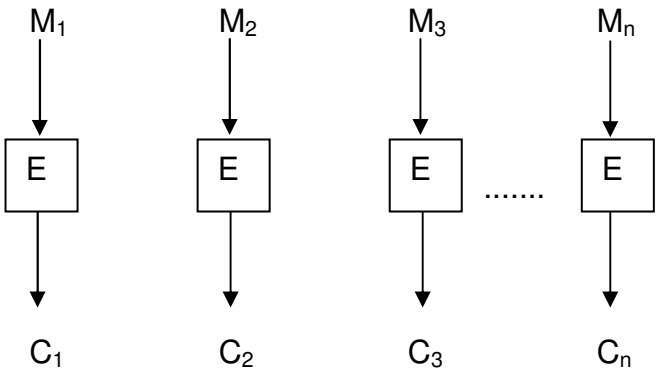


Şekil 4.1. Gizli anahtarlı kripto sistem ile haberleşme

Gizli anahtarlı kripto sistemleri uygulama sahalarında ikiye ayrılır;

4.1.1. Blok şifreleme

Şifreleme ve şifre çözme işleminde metinler sabit uzunluklu dizilere bölünüp blok blok işleme tabi tutulur (örneğin 8, 16, 32 bit veya bayt). Anahtar uzunluğu ise yine sabittir. Blok şifrelemeye örnek olarak IBM tarafından 1976 yılında tasarlanan ve A.B.D Teknoloji Standartları Enstitüsü NIST tarafından her dört yılda bir güvenliği onaylanan DES (Data Encryption Standard) algoritması verilebilir. DES algoritması şifrelenecek metni 64 bitlik bloklar halinde şifreler, kullandığı anahtar boyu ise yine 64 bittir. Yalnız burada anahtarın işaret bitlerinin ayıklanmaları durumunda anahtar boyunun 56 bite indiğini hatırlatmak gerekir (Şekil 4.2).



Şekil 4.2. Blok şifreleyici

4.1.2. Dizi şifreleme

Bu çeşit şifrelemede algoritmanın girdisi yalnızca anahtardır. Algoritma anahtardan rasgele bir diziye çok benzeyen kayan anahtar dizisi üretir. Daha sonra kayan anahtar dizisinin elemanları ile açık metin veya kapalı metin dizisinin elemanları ikili tabanda toplanarak şifreleme veya şifre çözme işlemi tamamlanır. Dizi şifreleme algoritmalarına örnek olarak RC4 algoritması gösterilebilir.

- Mesajı bit bit işler. (dizi olarak)
- En çok bilineni Vernam şifreleyicisidir (aynı zamanda tek kullanımlık sistem (one-time pad) de denir).
- 1917'de AT&T de çalışan Vernam tarafından geliştirilmiştir.
- Basit olarak mesaj bitlerini rasgele anahtar bitlerine ekler.
- Mesaj biti kadar anahtar biti gerekir. Pratikte zordur.
- Anahtar tamamen rasgele olduğu için koşulsuz güvenlik sağlanmıştır.

- Böyle büyük bir anahtar dağıtımı güç olduğu için anahtar dizisi daha küçük (taban) bir anahtardan üretilebilir. Bunun için rasgele sembol fonksiyonları kullanılır.

- Her ne kadar bu çok kolay gözükse de pratikte kriptografik olarak güçlü iyi bir rasgele fonksiyon bulmak çok zordur. Bu halen birçok araştırmanın konusudur.

4.2. Açık (Asimetrik) Anahtarlı Kripto Sistemler

Açık anahtarlı kriptografinin gelişmesi, kriptografi tarihindeki en büyük devrimdir. Başlangıcından günümüze kadar, bütün kriptografik sistemler, permütasyon işlemlerinin temel alınmasıyla oluşturulmuşlardır. Sadece elle hesaplanabilen algoritmalarla çalışabilme döneminden sonra, şifreleme/şifre çözme yapan rotor makinelerinin ortaya çıkması sonucunda, geleneksel kriptografide büyük bir gelişme kaydedilmiştir. Elektromekanik rotor, çok fazla inceliklere sahip ve karmaşık kriptografik sistemlerin geliştirilebilmesini sağlamıştır. Mevcut bilgisayarlarla daha karmaşık sistemler tasarlanmış ve en tanınanlarından olan -IBM´in- Lucifer girişimi gelişerek DES´i oluşturmuş ve DES dünyadaki kriptografi teknikleri arasında en yüksek seviyeye gelmiştir. Rotor makineleri ve DES, önemli avantajlar sunmalarına rağmen, halen permütasyon işlemlerine bağımlıdırlar.

Açık anahtarlı kriptografi, daha önceki gelişmelerden radikal bir kopuştur. Açık anahtarlı kriptografik sistemlerin en önemli özellikleri, permütasyondan çok matematiksel işlevler üzerine temellenmiş olmalarıdır. Daha da önemlisi, açık anahtarlı kriptografi, tek anahtar kullanan simetrik geleneksel şifreleme algoritmalarının tersine, iki ayrı anahtarın asimetrik kullanımını öngörür. Anahtar dağıtımı ve kimlik denetimi gibi gizlilik ve güven gerektiren durumlarda, iki anahtar kullanımı etkili sonuçlar ortaya koymuştur.

Açık anahtarlı şifreleme ile ilgili bazı yaygın, yanlış bilgilerden bahsetmek uygun olacaktır. Bu yanlış düşüncelerden birisi, açık anahtarlı şifrelemenin, kripto analize karşı geleneksel şifreleme yöntemlerinden daha dirençli olduğudur. Örneğin böyle bir iddia, Gardner´ın meşhur Scientific America adlı 1977 yılında yayınladığı makalesinde yapılmıştır. Aslında, şifrelemenin güvenliği, anahtarın uzunluğuna ve kırılan şifreli metnin içerdiği hesapsal işlemlerin karmaşıklığına dayanır. İster geleneksel ister açık anahtarlı şifreleme olsun, kriptoanaliz bakış açısına göre birini direğinden üstün tutmak doğru olamamaktadır.

Diğer bir yanlış düşünce de, genel amaçlı kullanım için geliştirilmiş bir teknik olan açık anahtarlı şifrelemenin, geleneksel şifrelemeyi modası geçmiş kıldığıdır. Tam tersine, geleneksel şifrelemeden vazgeçileceği sanısı, açık anahtarlı şifreleme yöntemlerinin, matematiksel fonksiyonlarından dolayı, ihtimal dışı gözükmektedir.

Son olarak, açık anahtarlı şifreleme kullanılırken, geleneksel şifrelemenin daha hantal anahtar dağıtım merkezleri ile karşılaştırıldığında, açık anahtarlı sistemlerin anahtar dağıtımının üzerinde kafa yorulması gerekmeyen, sıradan ve basit bir iş olduğuna dair yanlış bir anlayış vardır. Ancak bu kanının tersine açık anahtarlı şifreleme yöntemleri incelendiğinde, geleneksel şifreleme yöntemlerinin ihtiyaç duyduğu merkez temsilciler ve prosedürlerin, açık anahtarlı şifrelemenin ihtiyaç duyduklarından daha karmaşık ve etkili olmadığı görülmektedir.

4.2.1. Açık anahtarlı kripto sistemlerinin ilkeleri

Açık anahtarlı şifrelemenin genel amacı, gerçekleştireceği devrim ile geleneksel şifrelemenin en büyük iki problemine çözüm sağlamaktır. Bu problemlerden ilki gizli anahtarların dağıtımıdır. Gizli anahtar derken, geleneksel kriptografi uygulamalarının (DES, IDEA, Blowfish, CAST128, RC5, ...) kullandığı anahtarlar kastedilmektedir.

Geleneksel şifrelemeden yararlanarak birbirlerine şifrelenmiş metinler gönderecek olan taraflar, şifreleme ve şifre çözme işlemleri için, ya bir şekilde kendilerine ulaştırılmış olan anahtarı kullanacaklar, ya da, bir anahtar dağıtım merkezinden faydalanacaklardır. Açık anahtarlı kriptografinin mucitlerinden birisi olan Whitfield Diffie (diğeri de Stanford Üniversitesinden Martin Hellman'dır), kriptografinin özü olan, iletişimde %100 güvenlik esasını hiçe sayan bir anahtar dağıtım merkezi kullanma gerekliliğini ortadan kaldırmıştır. Çünkü tarafların kullanacakları gizli anahtarları bir anahtar dağıtım yetkilisinden almaları, istediği takdirde üçüncü parti bir kişinin iletişimi anlaşılır kılabileceği tehlikesini barındırmaktadır.

Diffie, üzerinde düşündüğü ikinci problem olan "sayısal imza" konusunun, yukarıda ifade edilenden farklı bir konu olduğunu görmüştür. Eğer kriptografinin kullanımı, sadece askeri konularda değil, özel ve kâr amaçlı uygulamalarda da kullanılacak kadar yaygın olsaydı, bu durumlar için kullanılacak elektronik belge ve dokümanlarda da, kâğıt dokümanlarda kullanılan kişisel imzalara gerek duyulacak ve böylece sayısal imzalar sayesinde, bir mesajı kimin gönderdiği kesinlikle bilinecek, bu da herkesi memnun eden bir yöntem olacaktır.

4.2.2. Açık anahtarlı kripto sistemlerin karakteristikleri

Açık anahtarlı şifreleme/şifre çözme algoritmaları, şifreleme için bir anahtara, şifre çözme içinse bu anahtarla ilişkisi olan ama bu anahtardan farklı ikinci bir anahtara ihtiyaç duyarlar. Bu durumda güvenlik sağlanmış olur. Bu algoritmalar şu önemli karakteristiğe sahiptirler:

- Sadece kriptografik algoritma ve şifre çözme anahtarı verilmişken, bir takım hesaplamalar yolu ile şifreleme anahtarını bulmak mümkün değildir.

Bununla beraber RSA gibi bazı algoritmalar şu karakteristikleri de gösterirler:

- Her iki benzer anahtar da şifreleme ve şifre çözme için kullanılabilir. Bununla beraber, bir anahtar şifreleme için kullanılmışsa, şifre çözme için diğer anahtar kullanılmalıdır.

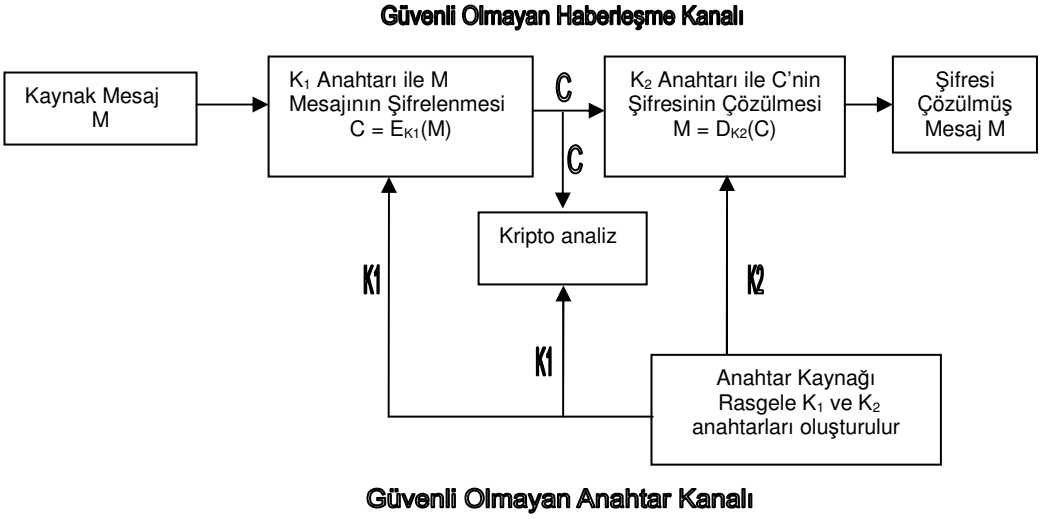
Şekil 4.3'de, açık anahtarlı şifreleme yöntemi gösterilmiştir. Başlıca adımlar şunlardır:

1. Her ağdaki her son sistem, mesaj alındığında şifreleme ve şifre çözme için kullanacak olduğu anahtar parçasını yaratır.

2. Her sistem, şifreleme anahtarını herkesçe erişilebilecek bir dosya ya da yazmaç içerisine kaydederek paylaştırır. Bu anahtarın, açık olan kısmıdır (public key). Özel anahtar saklı tutulur.

3. Eğer, A, B'ye bir mesaj yollamak isterse, mesajı B'nin açık anahtarını kullanarak şifreler.

4. B, mesajı aldığında, bu mesajı kendi özel anahtarını kullanarak şifreyi çözer. Diğer hiçbir alıcı mesajı şifreyi çözemez, çünkü mesajı çözecek olan özel anahtarı sadece B bilir.



Şekil 4.3. Açık anahtarlı kripto sistemi

Şekil 4.3'den de anlaşıldığı üzere her katılımcı, diğerlerinin açık anahtarlarına erişim hakkına sahiptir. Ve katılımcılar özel anahtarlarını yerel olarak yaratırlar. Bu yüzden, özel anahtarların paylaşılmasına gerek yoktur. Herhangi bir sebepten ötürü özel anahtarlar sahipleri tarafından değiştirilmek istenebilirler, bu durumda değişmiş olan yeni açık anahtar ilgili yerlere yeniden gönderilerek eskisi ile yer değiştirilir.

4.2.3. Açık anahtarlı kripto sistem uygulamaları

Açık anahtarlı algoritmaların üç önemli sınıfı vardır:

1. Açık anahtar Dağıtım Şeması (Public-Key Distribution Schemes PKDS), bilginin bir kısmının güvenli olarak değiştirilmesi için kullanılır (değer iki tarafa bağlıdır). Bu değer gizli anahtar şeması için bir oturum anahtarı olarak kullanılır.

2. İmza Şeması (Signature Schemes), sadece sayısal imza üretmek için kullanılır, burada gizli anahtar imzayı üretmekte, açık anahtar ise doğrulamakta kullanılır.

3. Açık anahtar Şeması (Public Key Schemes PKS), şifrelemek için kullanılır, burada açık anahtar mesajları şifreler, gizli anahtar mesajların şifresini çözer.

Herhangi bir açık anahtar şeması, gerekli olan oturum anahtarlı mesajı seçmek suretiyle PKDS olarak kullanılabilir. Çoğu açık anahtar şeması aynı zamanda imza şemasıdır (sağlanan şifreleme ve şifre çözme her iki sırada yapılabilir.).

4.2.4. RSA kripto sistemi

RSA kripto sistemi, 1978 yılında "Sayısal imza elde etme yöntemi ve açık anahtarlı kripto sistemler" adlı bir makale ile yayınlanmıştır. Adını yaratıcılarının (Ronald Rivest, Adi Shamir, Leonard Adleman) soyadlarının baş harflerinden alan RSA kripto sistemi, göndericinin bir yöntemle ve herkesçe bilinen açık bir anahtarla mesajlarını şifrelediği bir kripto sistemi olarak tanımlanır. Daha önceki gizli (simetrik) anahtarlı sistemlerin tersine anahtarı bilmek şifre çözme anahtarını ortaya çıkarmaz. Bu sistem hem gizlilik hem de sayısal imza sağlamak amaçlı kullanılabilir. Bu sistemin güvenliği tamsayılarda çarpanlara ayırma probleminin kolay olmaması temeline dayanır.

RSA kripto sisteminde kişilere şifreli mesaj gönderilebilmesi için o kişilerin açık anahtarlarına ihtiyaç vardır. Mesajı alan kişinin de mesajı okuyabilmesi için gizli bir anahtarının olması gerekir. Anahtar oluşturma aşağıdaki algoritmada ifade edilmiştir.

Anahtar oluşturma algoritması

Bir A kişisi anahtarını şu şekilde oluşturur:

- İki tane farklı, rasgele ve yaklaşık aynı uzunlukta olan p ve q asal sayıları seçer.
- n = pq ve ϕ = (p - 1)*(q-1) değerlerini hesaplar.

- 1 < e < ϕ ve OBEB (e, ϕ) = 1 olacak şekilde rasgele bir e sayısı seçer.

- Öklid algoritmasını kullanarak, 1 < d < ϕ ve ed ≡ 1 (mod ϕ) koşulunu sağlayan d sayısını hesaplar.

- A'nın açık anahtarı (n, e); A'nın gizli anahtarı ise d olur.

RSA anahtar oluşumunda e ve d tamsayıları sırasıyla şifreleme üssünü ve şifre çözme üssünü ve n ise mod sayısını gösterir. p ve q sayılarının onluk sistemde uzunluklarının 100 ve dolayısıyla da n nin uzunluğunun 200 olması beklenir. Fakat verilecek örneklerde kolaylık olması açısından küçük sayılar seçilecektir.

Şifreleme algoritması

1. B şahsı, A'ya bir m mesajı göndermek istiyor. B, m mesajını şifrelemek için aşağıdakileri yapar:

    - Öncelikle A'nın açık anahtarını (n,e) alır.

    - m mesajını [0, n -1] aralığında yazar.

    - Sonra c ≡ me (mod n) değerini hesaplar.

- Oluşan c şifresini A'ya gönderir.

2. Şifreli c metninden açık metni bulabilmek için A aşağıdaki işlemi uygular:

d gizli anahtarını kullanarak ve $m \equiv c^d \pmod{n}$ işlemini uygulayarak m açık metnine ulaşır.

NOT: Şifre çözme sisteminin çalışmasına $ed \equiv 1 \pmod{\phi}$ olduğu için $ed = 1 + k\phi$ eşitliğini sağlayan mutlaka bir k tamsayısı bulunur. Eğer OBEB (m, p) = 1 ise Fermat teoreminden dolayı;

$$m^{p-1} \equiv 1 \pmod{p} \qquad (4.1)$$

Eğer bu denkliğin her iki tarafının da k(q-1)'inci kuvvetlerini alırsak

$$m^{k(p-1)(q-1)} \equiv 1 \pmod{p} \qquad (4.2)$$

olur ve her iki tarafı da m ile çarptığımızda

$$m^{1+k(p-1)(q-1)} \equiv m \pmod{p} \qquad (4.3)$$

sonucuna ulaşırız.

Diğer taraftan, eğer OBEB (m, p) = p olursa yukarıdaki denklik yine geçerli olur; çünkü belli bir k tamsayısı için m = kp olduğunu varsayalım.

Bu durumda;

$m^{p-1} = (kp)(p-1) = k^{(p-1)}p^{(p-1)} \equiv p \pmod{p}$.
olur.

Eğer bu denkliğin her iki tarafının da k(q - 1)'inci kuvvetlerini alırsak

$m^{k(p-1)(q-1)} \equiv p^{k(p-1)(q-1)} \equiv p \pmod{p}$.

olur ve her iki tarafı da m ile çarptığımızda

$m^{1+k(p-1)(q-1)} \equiv mp = kp^2 \equiv kp = m \pmod{p}$.

buluruz.

İki durumda da

$m^{ed} \equiv m \pmod{p}$

olduğu görülür. Aynı şekilde,

$m^{ed} \equiv m \pmod{q}$ olur.

Sonuçta p ve q farklı asal sayılar olduğu için,

$m^{ed} \equiv m \pmod{n}$'dir. Böylelikle,

$c^d = m^{ed} \equiv m \pmod{n}$ (4.4)

olduğu görülür.

4.1. Örnek

1. Anahtar oluşturma: A şahsı p = 2357 ve q = 2551 olan iki tane asal sayı seçmiş olsun. Öncelikle A,

n = p * q = 6012707 ve

ϕ = (p- 1) * (q- 1) = 6007800

değerlerini hesaplar. A bir tane e = 3674911 değeri seçer. Bu e değeri, OBEB (e = 3674911, ϕ = 6007800) = 1 ve 1 < e = 3674911 < ϕ = 6007800 koşullarını sağlar.

Daha sonra Öklid algoritmasını kullanarak

e * d ≡ 1 (mod ϕ)

3674911 * d ≡ 1 (mod 6007800)

d = 422191 değerini hesaplar. A'nın açık anahtarı (n = 6012707; e = 3674911); gizli anahtarı da d = 422191 olur.

2. Şifreleme: B, m = 5234673 mesajını şifrelemek için A'nın açık anahtarını, yani (n = 6012707; e = 3674911), alır ve aşağıdaki şekilde olduğu gibi kapalı metin c'yi hesaplar:

$c \equiv m^e \pmod{n} = 5234673^{3674911} \pmod{6012707} \equiv 3650502$

ve bu değeri A'ya gönderir.

3. Şifre çözme: A, gelen c kapalı metninden m açık metnini aşağıdaki gibi hesaplar:

$m \equiv c^d \pmod{n} = 3650502^{422191} \pmod{6012707} \equiv 5234673$

RSA imza şeması

RSA kripto sistemi sayısal imzalar için de kullanılabilir. (n, e) A şahsının açık anahtarı, d sayısı da A'nın gizli şifre çözme üssü olsun. Öncelikle mesajın imzalanabilmesi için m mesajının {0,1,…, n-1} arasında olması istenir, daha sonra hesaplamalar yapılır.

İmzalama

A B'ye imzalı m mesajını göndermek isterse, mesaja kendisinin kapalı anahtarını uygular, yani

$\sigma = m^d \bmod n$

değerini hesaplar.

Daha sonra (m, σ) imzalı mesajı B'ye gönderir.

İmzayı doğrulama

B, A'dan aldığı (m, σ) imzalı mesajı doğrulamak için

$m = \sigma^e \bmod n$

değerini hesaplar. Çıkan sonuç m ise mesaj doğrulanmış olur.

4.2. Örnek

1. Anahtar Oluşturma: A kişisi p = 7927 ve q = 6997 asal sayılarını seçer ve n = pq = 55465219 ve ϕ = 7926* 6996 = 55450296 değerlerini hesaplar. Daha sonra A, ed = 5d ≡ 1 (mod 55450296) eşitliğinden d = 44360237

sayısını bulur. A'nın açık anahtarı (n = 55465219, e = 5); gizli anahtarı d = 44360237 olur.

2. İmzalama: m = 31229978 mesajını imzalamak için A şunu hesaplar:

$\sigma = m^d \bmod n = 31229978^{44360237} \bmod 55465219 \equiv 30729435$

ve (m = 31229978, σ = 30729435)'yi B'ye gönderir.

3. İmzayı Doğrulama: (m = 31229978, σ = 30729435)'yi alan B mesajı doğrulamak için şunu yapar:

$m = \sigma^e \bmod n = 30729435^5 \bmod 55465219 \equiv 31229978$

Çıkan sayı m olduğu için imza doğrulanmış olur.

4.2.5. Ayrık logaritma (Discrete logarithm)

RSA kripto sisteminde, RSA fonksiyonu m olarak verilen bir elemanın e. kuvvetini oluşturur. Bu fonksiyon bire-bir bir fonksiyondur ve etkili bir şekilde hesaplanır. Eğer n'nin çarpanlara ayrılması bilinmiyorsa, e. kökü hesaplamak için etkili bir algoritma yoktur. Sayılar teorisinde hesaplaması kolay fakat tersinin hesaplaması zor olan başka fonksiyonlar da vardır. Bunlardan en önemlilerinden biri de sonlu cisimlerde kuvvet almadır. Basit olarak sadece asal cisimler düşünülecektir.

p bir asal sayı ve g de Z_p^* de bir ilkel kök olsun. Ayrık kuvvet fonksiyonu (discrete exponential function)

Exp: $Z_{p-1} \rightarrow Z_p^*,\ x \mapsto g^x,$

tekrarlı karesini alma algoritması örneğinde olduğu gibi etkili bir şekilde hesaplanabilir. Kuvvetin logaritması fonksiyonunun tersini hesaplamak için etkili bir algoritma bilinmemektedir. Bu tahmine ayrık logaritma tahmini (discrete logarithm assumption) denir.

4.2.6. El-Gamal açık anahtarlı kripto sistemi

El-Gamal açık anahtarlı kripto sistemi, anahtar transferi modunda Diffie-Hellman anahtar anlaşması (Diffie-Hellman Key Agreement) olarak görülebilir. Güvenilirliği ayrık logaritma problemi ve Diffie-Helman probleminin kolay çözülememesi temeline dayanır. Temel El-Gamal ve genelleştirilmiş El-Gamal şifreleme şeması bu bölümde tanımlanmıştır.

<u>El-Gamal açık anahtarlı şifrelemede anahtar oluşturma algoritması</u>

Her kişi kendi açık anahtarını ve buna bağlı gizli anahtarını oluşturur. Bunu oluşturmak için A şahsı şunları uygular:

1. Çok büyük rasgele bir p asal sayısı ve mod p ye göre tamsayıların oluşturduğu çarpım grubu Z_p^* nin bir üreteci α yı oluşturur.

2. $1 \leq a \leq p - 2$ şeklinde olan bir a tamsayısı seçer ve $\alpha^a \bmod p$ değerini hesaplar.

3. A'nın açık anahtarı (p, α, α^a); A'nın gizli anahtarı ise a olur.

El-Gamal açık anahtarlı şifreleme algoritması

B şahsı A için m mesajını şifrelesin.

1. Şifreleme: B mesajı şifreleme için şunları yapar:

- A'nın açık anahtarını (p, α, α^a) alır.

- mesajı {0, 1, …, p-1} aralığında m tamsayısı olarak ifade eder.

- 1 ≤ k ≤ p -2'yi sağlayan rasgele bir k tamsayısı seçer.

- $\gamma = \alpha^k \bmod p$ ve $\delta = m * (\alpha^a)^k \bmod p$ değerlerini hesaplar.

- Son olarak $c = (\gamma, \delta)$ kapalı metnini A'ya gönderir.

2. Şifre çözme: c kapalı metninden m açık metine ulaşmak için A şunları yapar:

- a gizli anahtarını kullanarak $\gamma^{-a} \bmod p$ değerini hesaplar ($\gamma^{-a} = \alpha^{-ak} \bmod p$).

- $\gamma^{-a} * \delta \bmod p$ değerini hesaplayarak m'yi bulur.

$\gamma^{-a} * \delta \equiv \alpha^{-ak} * m\ \alpha^{ak} \equiv m \pmod p$

4.3. Örnek

1. Anahtar Oluşturma: A şahsı bir p = 2357 asal sayısı ve $\alpha = 2 \in Z^*$ bir üreteç seçer. Buna ilave olarak bir a = 1751 gizli anahtarı seçer ve

$\alpha^a \bmod p = 2^{1751} \bmod 2357 \equiv 1185$

değerini hesaplar. A'nın açık anahtarı (p = 2357, α = 2, α^a = 1185)'tir.

2. Şifreleme: m = 2035 mesajını şifrelemek için B şahsı rasgele bir k = 1520 tamsayısı seçer ve

$\gamma = 2^{1520} \bmod 2357 \equiv 1430$ ve

$\delta = 2035 * 1185^{1520} \bmod 2357 \equiv 697$

değerlerini hesaplar. Son olarak B ($\gamma = 1430$, $\delta = 697$)'yi A'ya gönderir.

3. Şifre çözme: A gelen kapalı metni çözmek için

$\gamma^{-a} = 1430^{-1751} \equiv 1430^{606} \bmod 2357 \equiv 872$ bulur ve m mesajına da

$m = 872 * 697 \bmod 2357 \equiv 2035$

böylece ulaşır.

El-Gamal imzası

El-Gamal kripto sisteminde imza RSA'da olduğu gibi mesajın doğru kişiden geldiğini kontrol etmek için kullanılır. Sadece kapalı metin yerine imzalanmış kapalı metin gönderilerek o kapalı metnin istenen kişiden gelip gelmediği de kontrol edilmiş olur. A şahsının açık anahtarı ($p, \alpha, \alpha^a = y$) ve gizli anahtarının da a olduğu düşünülsün.

İmza algoritması

m mesajının Z_p nin bir elemanı olduğu düşünülür. Eğer değilse özet fonksiyonu kullanılarak m mesajının Z_p nin elemanı olması sağlanır. A şahsı m mesajını şu şekilde imzalar:

1. Rasgele bir t tamsayısı seçer; t tamsayısı $1 \le t \le p - 2$ ve OBEB(t, p-1) = 1 koşulunu sağlamalıdır.

2. $r = \alpha^t$ ve $s = t^{-1}(m - ra) \bmod (p - 1)$ eşitliklerini kurar.

3. (m, r, s) A'nın imzalı mesajıdır.

Doğrulama

(m, r, s) imzalı mesajı alan B şahsı aldığı mesajın A'dan geldiğini şu şekilde doğrular:

1. Öncelikle $1 \leq r \leq p - 1$ olduğunu kontrol eder. Eğer değilse imzayı reddeder.

2. Daha sonra $v = \alpha^m$ ve $w = y^r r^s$ değerlerini hesaplar (Buradaki y sayısı A'nın açık anahtarındaki y sayısıdır.)

3. Eğer $v = w$ eşitliği sağlanıyorsa imza kabul edilir, aksi takdirde reddedilir.

4.4. Örnek

1. Anahtar Oluşturma: A şahsı bir p = 2357 asal sayısı ve $\alpha = 2 \in Z^*$ bir üreteç seçer. Buna ilave olarak bir a = 1751 gizli anahtarı seçer ve

$\alpha^a \bmod p = 2^{1751} \bmod 2357 \equiv 1185$

değerini hesaplar. A'nın açık anahtarı ($p = 2357$, $\alpha = 2$, $\alpha^a = 1185$) tir.

2. İmza Oluşturma: Basit olması açısından mesaj m = 1463 olarak seçilsin (Eğer mesaj p asal sayısından büyük olsaydı özet fonksiyonundan geçirilirdi). m = 1463 mesajını imzalamak için A önce rasgele bir t = 1529 sayısı seçer, daha sonra

$r = \alpha^t \bmod p = 2^{1529} \bmod 2357 \equiv 1490$ ve

$t^{-1} \bmod (p - 1) = 1529^{-1} \bmod (2356) \equiv 245$

$s = t^{-1}(m - ra) \bmod (p - 1) = 245(1463 – 1490 * 1751) \bmod 2356 \equiv 1777$

A'nın imzası (m = 1463, r = 1490, s = 1777)

3. İmzayı Doğrulama: B aldığı imzalı mesajı doğrulamak için önce

$v = \alpha^m \bmod p = 2^{1463} \bmod 2357 \equiv 1072$

değerini hesaplar. Daha sonra

$w = y^r r^s \bmod p = 1185^{1490}1490^{1777} \bmod 2357 \equiv 1072$

değerini hesaplar ve $v = w$ olduğu için imzayı kabul eder.

4.2.7. Diffie-Hellman anahtar anlaşması (Diffie-Hellman key agreement)

Diffie-Hellman anahtar anlaşması, anahtar dağıtma problemine ilk pratik çözümdür. Üs alarak anahtar değiştirme olarak da bilinen bu sistem daha önce hiç haberleşme sağlamamış iki tarafın açık kanal üzerinden mesajlarını birbirlerine göndererek ortak bir anahtar yaratması temeline dayanır.

p yeteri kadar büyük bir asal sayı olsun ve Z^*_p de ayrık logaritma problemini çözmek mümkün olmasın. Ayrıca, g Z^*_p' da ilkel bir kök olsun. p ve g herkes tarafından bilinsin. A ve B kişileri aşağıdaki yolu izleyerek ortak bir anahtar yaratabilirler:

<u>Diffie-Hellman anahtar anlaşması algoritması</u>

- A, $0 \le a \le p - 2$ eşitsizliğini sağlayan ve rasgele bir a sayısı seçer. $c = g^a$ değerini bulur ve bunu B'ye gönderir.

- B, $0 \leq b \leq p - 2$ eşitsizliğini sağlayan ve rasgele bir b sayısı seçer. $d = g^b$ değerini bulur ve bunu A'ya gönderir.

- A, ortak anahtar k'yı şu şekilde hesaplar:

$k = d^a = (g^b)^a$

- B, ortak anahtar k'yı şu şekilde hesaplar:

$k = c^b = (g^a)^b$

Böylelikle A ve B aralarında ortak bir anahtar olan k için anlaşmış olurlar.

5. ELİPTİK EĞRİ TEORİSİ

Bu bölümde, matematikteki birçok soyut cisimde yer alan eliptik eğriler hakkındaki genel teoriden bahsedilmiştir. Eliptik eğrilerin arkasında yatan temel kavramlar hakkında bilgi verilmesi amaçlanmaktadır [7].

Eliptik eğriler elips değildirler. Bu şekilde adlandırılmalarının sebebi, bir elipsin çemberinin hesaplanması için kullanılan kübik denkliklere benzer ifadeler ile gösterilmeleridir. Bir *K* cismini ele alırsak, *K* cismi, *R* Gerçel sayılar, *Q* Rasyonel sayılar, *C*- Kompleks sayılar veya p'nin asal bir sayı olduğunu kabul edersek, $q=p^r$ elemanlarından oluşan F_q –sonlu cismi olabilir. GF(2) sonlu cisminin karakteristiği 2, gerçel ve kompleks sayıların karakteristiği ise sonsuzdur.

5.1. Tanım (Karakteristik)

Bir cisimde 1 sayısına kendisini ekleyelim. Eğer 1+1 = 0 oluyorsa bu cismin karakteristiği 2 dir. Eğer 1+1+1 = 0 oluyorsa bu cismin karakteristiği 3 tür. Genel olarak $\underbrace{1+1+1+\ldots\ldots\ldots\ldots+1}_{n} = 0$ oluyorsa cismin karakteristiği n dir.

Eğer cisimde 1 sayısına kendisi sonsuz defa eklenebiliyorsa cismin karakteristiği 0 dır.

Herhangi bir *K* cismi için, eliptik eğrinin genel denklemi;

$$y^2+axy+by= x^3+cx^2+dx+e \tag{5.1}$$

ile verilir.

Eğer *K* cisminin karakteristiği Char(*K*) = 2 ise Eş. 5.1,

$$y^2+ay= x^3+bx+c \tag{5.2}$$

veya

$$y^2+xy= x^3+ax^2+b \quad (5.3)$$

denklemlerine dönüşür. Eğer *K* cisminin karakteristiği Char(*K*) = 3 ise Eş. 5.1,

$$y^2= x^3+ax^2+bx+c \quad (5.4)$$

denklemine dönüşür. Eğer *K* cisminin karakteristiği Char(*K*) ≠2 veya 3 ise Eş. 5.1,

$$y^2= x^3+ax+b \quad (5.5)$$

denklemine dönüşür.

Her bir denklem için Affine Dönüşümlerini kullanırız ve bu denklemde yer alan *x,y,a,b,c,d* ve *e* değerleri de *K* cisminin üstünde yer almaktadır.

Yukarıda belirtilen herhangi bir şartı sağlayan, *K* alanında yer alan ve (x,y) noktalarının kümesinden oluşan denkleme *E* eliptik eğri denklemi diyebiliriz.

5.1. Eliptik Eğriler

Genel olarak, bundan sonra inceleyeceğimiz eliptik eğri denklemlerinde Eş. 5.5'i kullanacağız:

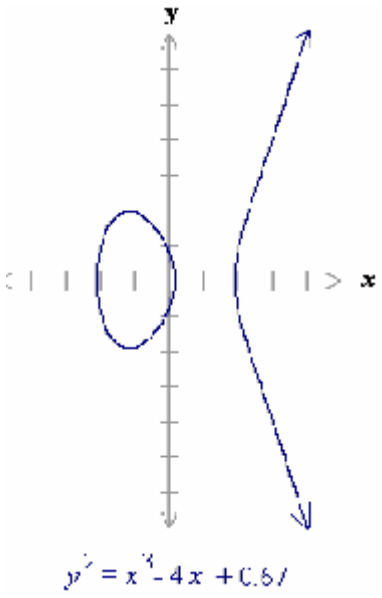
$$y^2 = x^3+ax+b$$

bu denklemdeki a ve b sayıları gerçel sayılardır ve x^3+ax+b denkleminin çoklu (multiple) kökü olmaması için $4a^3+27b^2\neq 0$ olmalıdır. Eğer bu koşulları sağlıyor ise $y^2 = x^3+ax+b$ eliptik eğridir deriz. Ayrıca, eliptik eğrinin tanımlamasında, daha sonra ayrıntılı şekilde inceleyeceğimiz, sonsuzluk ya

da sıfır nokta adı verilen bir *O* notasyonu vardır. En büyük dereceli üs 3 olduğundan dolayı bu tip denklemler kübik olarak adlandırılırlar.

5.1. Örnek

a=-4 ve b=0.67 olarak verilen eliptik eğri denkleminin grafiği:

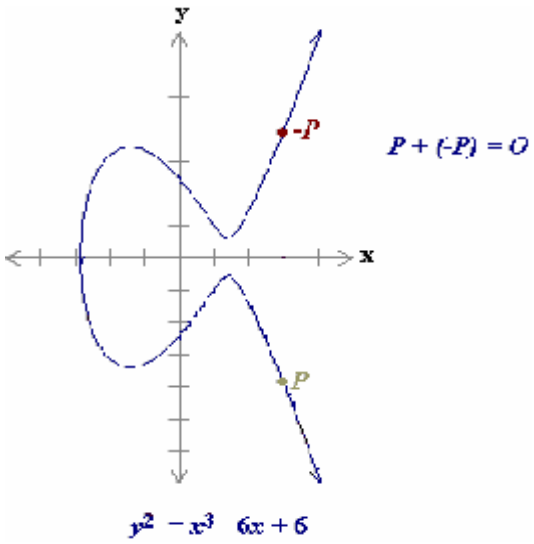


Şekil 5.1. $y^2= x^3-4x+0.67$ Eliptik eğrisi

Eğer bir eliptik eğrinin 3 noktası düz bir çizgi üzerinde bulunuyorsa, bunlar *O* sonsuzluk notasyonu olarak özetlenir. Bu açıklamadan yola çıkarak, bir eliptik eğri için şu kuralları tanımlayabiliriz:

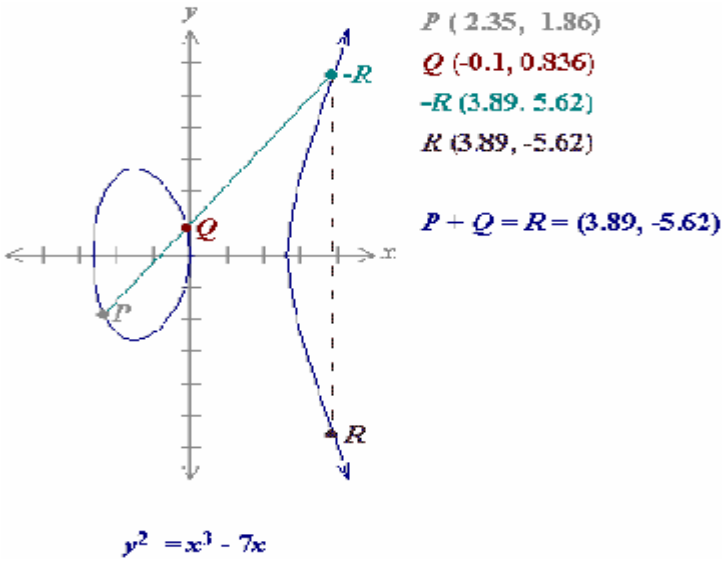
1. Eliptik eğri üzerindeki herhangi bir *P* noktası için, $P+O=P$ olur.

2. Bir dikey çizgi, aynı *x* değeri için eliptik eğriyi $P_1=(x,y)$ ve $P_2=(x, -y)$ gibi iki noktasında kesiyorsa, bu çizgi aynı zamanda eliptik eğriyi sonsuzluk noktasında da kesiyordur. Bu yüzden $P_1+P_2+O =O$ ve $P_1=-P_2$ olur. Böylece bir noktanın negatifi, *x* ekseni üzerinde aynı değeri alacak şekilde bir noktadır ve bu noktanın *y* ekseni üzerindeki değeri ilk noktanın negatifidir.



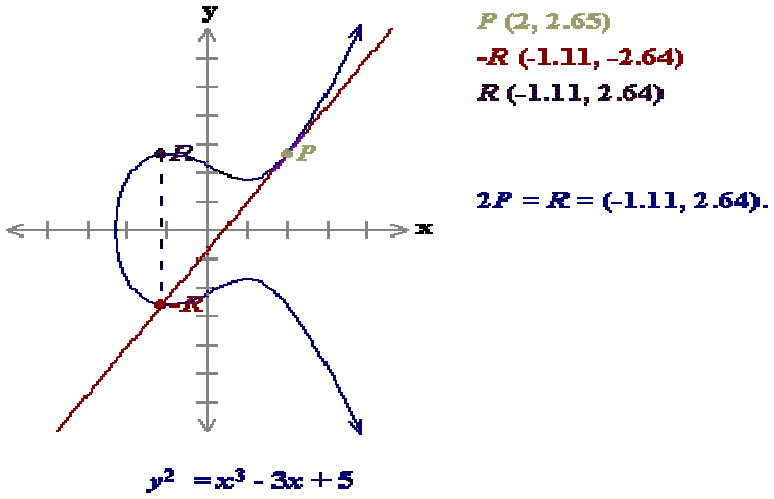
Şekil 5.2. $y^2= x^3-6x+6$ Eliptik eğrisi

3. x koordinatı farklı olan Q ve R noktası seçip bu iki noktadan geçen düz bir çizgi çizdiğimizde kesişimin üçüncü noktası olan P_1'i buluruz. P_1 noktası sadece bir tanedir (eğer çizdiğimiz doğru Q veya R noktalarından birisinden teğet geçiyorsa bu durumda $P_1=Q$ veya $P_2=Q$ alırız). Bu durumda $Q+R+P_1=O$ ve dolayısıyla $Q+R = -P_1$ olacaktır.



Şekil 5.3. $y^2= x^3-7$ Eliptik eğrisi üzerindeki iki noktanın toplamı

4. Bir *P* noktasının iki katını almak için, bir teğet çizgisi çizip eğriyi kestiği diğer noktayı buluruz. Eğer bu noktaya *–R* diyecek olursak *P+P=2P=R* eşitliği sağlanır.



Şekil 5.4. $y^2= x^3-3x+5$ Eliptik eğrisi üzerindeki P noktasının çift katı

5.2. Sonlu Cisimlerdeki Eliptik Eğriler

ECC için, eliptik eğrilerin, "sonlu cisimlerdeki eliptik eğriler" olarak tanımlanan bir formu ile ilgileneceğiz. Bu şu şekilde gösterilir: *p* bir asal sayı olsun ve *a* ve *b*, *p*'den küçük, negatif olmayan iki tamsayı olsun:

$$4a^3+27b^2 \pmod p \neq 0 \qquad (5.6)$$

Bu durumda, $E_F(a,b)$, *(x,y)*'nin *p*'den küçük negatif olmayan tamsayılar olduğu durum için, *O* sonsuz noktası ile beraber şu eşitliği ifade eder:

$$y^2 \equiv x^3+ax+b \pmod p \qquad (5.7)$$

Örneğin, *p=23* ve eliptik eğrimiz de $y^2 = x^3+x+1$ olsun. Bu durumda, *a=b=1* olur. Bu durumda, $4 \times 1^3 + 27 \times 1^2 \pmod{23} = 8 \neq 0$, bizim eliptik grubumuzun mod 23'e göre durumunu gösterir.

Eliptik grup için sadece (mod p'den dolayı), (0, 0)-(p, p) aralığında olan pozitif tamsayılar ile denklem oluştururuz. Çizelge 5.1'de, $E_{23}(1, 1)$ için *O* dışındaki noktaları listelenmiştir. Genel olarak liste aşağıdaki yolla oluşturulmuştur:

1. $0 \leq x \leq p$ koşulunu sağlayan her *x* değeri için $x^3+ax+b \ (mod\ p)$ denklemi hesaplanmıştır.

2. Önceki adımın her sonucu için, sonucun mod p ye göre çift katlı kökü olup olmadığına bakılır, eğer yoksa bu *x* değeri için, $E_F(a,b)$'nin bir değeri yoktur. Aksi takdirde, çift katlı kök koşulunu sağlayan iki adet *y* vardır (*y*'nin 0 olduğu durum haricinde). Bu (x,y) değerleri, $E_F(a,b)$'nin noktalarıdır.

Çizelge 5.1. $E_{23}(1, 1)$ Eliptik eğri üzerindeki noktalar

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

5.2. Örnek

$E_F(a,b)$ için bahsedilmiş kurallar, her $P,Q \in E_F(a,b)$ olacak şekilde alınan noktalar için aşağıdaki gibi gösterilebilir:

1. $P+O=P$

2. Eğer $P = (x,y)$ ise, $P+(x,-y) = O$ olur. $(x,-y)$ noktası, P'nin negatifidir ve $-P$ olarak gösterilir. Örneğin, $E_{23}(1, 1)$'de $P = (13,7)$ alalım. Bu durumda $-P = (13,-7)$ olacaktır. Fakat -7 mod 23 e göre 16 ettiğinden dolayı, bizim $-P$ noktamız aslında, yine $E_{23}(1, 1)$'de yer alan (13,16) noktasıdır.

3. Eğer $P = (x_1,y_1)$ ve $Q = (x_2,y_2)$ ise ve $P \neq -Q$ ise, bu durumda, $P+Q = (x_3,y_3)$ şu kuralla hesaplanır:

$$x_3 \equiv \lambda^2 - x_1 - x_2 \pmod p \tag{5.8}$$

$$y_3 \equiv \lambda (x_1 - x_3) - y_1 \pmod p \tag{5.9}$$

λ için koşul,

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{eğer } P \neq Q \text{ ise} \\ \dfrac{3x_1^2 + a}{2y_1} & \text{eğer } P = Q \text{ ise} \end{cases} \tag{5.10}$$

Eliptik eğri üzerinde yer alan bir noktanın k katını almak demek P noktasının k defa toplanması anlamına gelmektedir.

$$k * P = \underbrace{P+P+P\ldots\ldots\ldots\ldots+P}_{k}$$

5.3. Örnek

E eliptik eğrisinin $y^2 = x^3+x+6$ ve p asal sayısının 11 olduğunu kabul ederek, E eliptik eğrisinin üzerindeki noktaları hesaplayalım. Bunu yaparken verilen her x değerinin yukarıdaki kuralları sağlayıp sağlamadığını ve hangi aralıkta ise ona göre hesaplama yapılması gerektiğini unutmamak gerekir. Bunları dikkate alarak işlemleri yaparsak Çizelge 5.2'deki sonuçları buluruz:

Çizelge 5.2. $y^2 = x^3+x+6$ Eliptik eğri üzerindeki çift katlı kökler

x	x^3+x+6 mod 11	Karekökü var mı?	y
0	6	hayır	
1	8	hayır	
2	5	evet	4,7
3	3	evet	5,6
4	8	hayır	
5	4	evet	2,9
6	8	hayır	
7	4	evet	2,9
8	9	evet	3,8
9	7	hayır	
10	4	evet	2,9

E eliptik eğrisi üzerinde 13 nokta bulunmaktadır. Periyodik tekrara geçildikten sonra, sonsuzluktaki nokta dışındaki her hangi bir noktayı E' nin başlangıç noktası olarak kabul ediyoruz. Biz burada α = (2,7) noktasını başlangıç noktası olarak kabul edelim. Daha sonra α nın katlarını hesaplarız.

İlk önce 2 katını alırsak, 2α =(2,7)+(2,7)

$\lambda = (3 \times 2^2 + 1)(2 \times 7)^{-1} \bmod 11$

$= 2 \times 3^{-1} \bmod 11$

$= 2 \times 4 \bmod 11$

$= 8$

Daha sonra x_3 ve y_3 değerlerini bulursak,

$x_3 = 8^2 - 2 - 2 \bmod 11$

$= 5$

$y_3 = 8(2-5) - 7 \bmod 11$

$=2,$

ve böylelikle 2α = (5,2) olur.

buluruz.

Bir sonraki çarpımı $3\alpha = 2\alpha + \alpha = (5,2) + (2,7)$ şeklinde alırsak, tekrar λ değerini hesaplamalıyız. Bu da aşağıda belirtilen işlemlerle yapılır:

$\lambda = (7-2)(2-5)^{-1} \bmod 11$
$= 5 \times 8^{-1} \bmod 11$
$= 5 \times 7 \bmod 11$
$= 2.$

Daha sonra x_3 ve y_3 değerlerini bulursak,
$x_3 = 2^2 - 5 - 2 \bmod 11$
$= 8$
$y_3 = 2(5-8) - 2 \bmod 11$
$= 3,$

ve böylelikle $3\alpha = (8,3)$ olur.

Aynı şekilde diğer katlarını da hesaplarsak;

$\alpha = (2,7)$	$2\alpha = (5,2)$	$3\alpha = (8,3)$
$4\alpha = (10,2)$	$5\alpha = (3,6)$	$6\alpha = (7,9)$
$7\alpha = (7,2)$	$8\alpha = (3,5)$	$9\alpha = (10,9)$
$10\alpha = (8,8)$	$11\alpha = (5,9)$	$12\alpha = (2,4)$

buluruz.

Bu sonuçlardan da anlaşıldığı gibi $\alpha = (2,7)$ gerçekten başlangıç noktası olarak kabul edilir.

5.1. Teorem (Hasse Teoremi)

N'nin, F_q sonlu cisminde yer alan ve *E* eliptik eğrisi üzerindeki noktaların toplam sayısı olduğunu kabul edersek;

$$\left|N-(q+1)\right| \le 2\sqrt{q} \tag{5.11}$$

N değeri Eş. 5.11 ile sınırlandırılmaktadır.

6. ELİPTİK EĞRİ KRİPTOGRAFİSİ (ECC)

ECC'deki toplama işlemi ile RSA'daki modüler çarpım işlemi ve çoklu toplama işlemi ile de RSA'daki modüler üs alma işlemi birbirlerine çok benzemektedir. Eliptik eğriler kullanan bir kriptografik sistem oluşturabilmek için, bir sayıyı iki asal çarpanına ayırmak ya da ayrık logaritma almak gibi zor bir problem bulmamız gerekmektedir [8].

Diyelim ki, $P, Q \in E_F(a,b)$ ve $k<p$ iken, $Q=kP$ olsun. k ve P verildiğinde Q değerini hesaplamak nispeten kolay olduğu halde, Q ve P verildiğinde k değerini hesaplamak gerçekten çok zordur.

6.1. Diffie-Hellman Anahtar Değişimi Örneği

Eliptik eğriler kullanılarak anahtar değişimi aşağıdaki şekilde yapılabilir. Önce $p \approx 2^{180}$ olacak şekilde bir p asal sayısı ve $y^2 = x^3+ax+b$ denklemindeki eliptik eğri parametreleri olan a ve b seçilsin. Bu, eliptik noktalar grubu olan $E_F(a,b)$'yi oluşturur. Sonrasında $E_F(a,b)$ içerisinden, başlangıç noktası (generator point) olacak olan $\alpha=(x_1,y_1)$ seçilsin. A'nin seçilmesindeki en önemli kriter, $n\alpha=0$ eşitliğini sağlayan en küçük n değerinin çok bir büyük bir asal sayı olması gerekliliğidir. $E_F(a,b)$ ve α, kripto sistemin tüm katılımcılarca bilinecek parametreleridirler.

Bir A ve B kullanıcısı arasındaki anahtar değişimi aşağıdaki gibi gerçekleşir:

1. A, n'den küçük bir n_A tamsayısı seçer. Bu A'nın özel anahtarıdır. Daha sonra A, $P_A= n_A\alpha$ hesabıyla $E_F(a,b)$'nin bir noktası olan kendi açık anahtarını oluşturur.

2. B'de aynı metotla kendi açık anahtarı P_B'yi oluşturur.

3. *A* gizli anahtarı $K = n_A P_B$ ile, *B*'de gizli anahtarı $K = n_B P_A$ ile elde eder.

Üçüncü aşamadaki iki hesaplamanın sonucuda aynıdır. Çünkü,

$n_A P_B = n_A(n_B \alpha) = n_B(n_A \alpha) = n_B P_A$ eşitliği mevcuttur.

Bu yönteme bir atak gerçekleştirmek isteyen saldırgan, verilmiş α ve $k\alpha$ değerlerinden yola çıkara k değerini hesaplamak isteyecektir ve bu çok zordur.

Bu konuda bir örnek verelim: $p = 211$ olarak alalım. $E_F(0,-4)$, eliptik eğri $y^2 = x^3 - 4$ ve $\alpha=(2,2)$ olsun. Hesapladığımız takdirde, $241\alpha=0$ olacaktır. *A*'nın özel anahtarı $n_A=121$ ve bu durumda bu kullanıcının genel anahtarı $P_A = 121(2,2) = (115,48)$. *B*'nin özel anahtarı $n_B=203$ ve bu durumda bu kullanıcının genel anahtarı $P_B=203(2,2) = (130,203)$ olur. Bu koşullar altında paylaşılmış gizli anahtar $121(130,203) = 203(115, 48) = (161,169)$ olarak bulunur.

Görüldüğü gibi, anahtar iki parçadan oluşmaktadır. Eğer bu anahtar geleneksel şifreleme için bir oturum anahtarı olarak kullanılacaksa, sadece bir sayının yaratılması gerekir. Basit olarak sadece *x* koordinatını ya da *y* koordinatını anahtar olarak kullanabiliriz.

Literatürde, eliptik eğriler yardımıyla şifreleme/şifre çözme yapan birçok yöntem bulunmaktadır. Burada en basit yöntem incelenmiştir. Bu sistem içerisindeki ilk görev, açık metin olan *m*'yi, bir x-y koordinatı ile belirlenmiş P_m noktası şeklinde göndermek üzere kodlamaktır. Bu P_m noktası bir şifreli metin gibi şifrelenecek daha sonrasında da şifre çözülecektir. Bir mesajı x ya da sadece y koordinat noktası olarak basitçe kodlayamayız çünkü tüm olası noktalar $E_F(a,b)$ içinde bulunmayabilir.

Anahtar değişimi sisteminde olduğu gibi şifreleme/şifre çözme sistemi de parametre olarak bir α noktası ve bir $E_F(a,b)$ eliptik grubuna ihtiyaç duyar.

Her bir kullanıcı, bir n_A özel anahtarı seçer ve $P_A = n_A\alpha$ ile bir açık anahtar üretir.

P_m gibi bir mesajı şifrelemek ve bir B kullanıcısına göndermek isteyen bir A kullanıcısı, rasgele pozitif bir n_A tamsayısı seçer ve C_m şifreli metninin noktalarını aşağıdaki şekilde elde eder:

$$C_m = (n_A\alpha,\ P_m + n_A P_B) \tag{6.1}$$

A kullanıcısı şifreleme esnasında B'nin açık anahtarı olan P_B'den yararlanmaktadır. B aşağıdaki şekilde mesajın şifresini çözer:

$$P_m + n_A P_B - n_B(n_A\alpha) = P_m + n_A(n_B\alpha) - n_B(n_A\alpha) = P_m \tag{6.2}$$

A, P_m mesajını, ona $n_A P_B$ ekleyerek maskeler. n_A değerini bilmeyen kimse A'nın uyguladığı maskeyi kaldıramaz. Bunun yanında A, n_B özel anahtarını bilen bir kişinin maskeyi kaldırabilmesi için bir ipucu bırakmaktadır.

6.1. Örnek

Başlangıç noktasının α = (2,7) ve B kullanıcısının özel anahtarının $n_B = 7$ olduğunu kabul edersek;

B kullanıcısının genel anahtarı: $P_B = 7\alpha = (7,2)$ olur.

Daha sonra şifreleme işlemi;

$$C_m = (n_A(2,7),\ P_m + n_A(7,2))$$

biçiminde yapılır.

Burada $P_m \in E$ ve $0 \le n_B \le 12$ olarak verilmektedir. Şifre çözme işlemi de

$D_m = (P_m + n_A P_B) - n_B P_A$

biçiminde yapılır.

A kullanıcısı $P_m = (10,9)$ mesajını şifrelemek istesin (Mesaj, E eliptik eğrisi üzerinde bulunuyor). Eğer A kullanıcısı rasgele bir n_A=3 değerini seçerse, hesaplamaları yaparken;

$P_A = 3(2,7)$
$= (8,3)$ ve

$P_m + n_A P_B = (10,9) + 3(7,2)$
$= (10,9) + (3,5)$
$= (10,2)$
bulur.

Bunun sonucunda $C_m = ((8,3),(10,2))$ elde edilir. Şimdi B kullanıcısı aldığı şifreli metni, C_m, aşağıdaki şekilde şifreyi çözer:

$D_m = (10,2) - 7(8,3)$
$= (10,2) - (3,5)$
$= (10,2) + (3,6)$
$= (10,9)$

Sonuç olarak şifrelenmiş metin doğru olarak şifresi çözümlenmiştir.

6.2. Eliptik Eğri Kriptografisinin Kullanım Alanları

6.2.1. Sayısal imzalar

Sayısal imzalar özet algoritmaları kullanılarak oluşturulurlar. Özet algoritmaları geri dönüşümü yapılamayan algoritmalardır ve özet

fonksiyonları kullanılarak yapılan bir şifreleme işlemi sonucu ortaya çıkan bir şifrelenmiş veri şifre çözme işlemi ile ilk haline geri dönüştürülemez.

Şifreleme işlemleri içinde en güvenli olanı açık anahtarlı kripto sistemlerinden eliptik eğri kripto sistemidir. Burada kullanılan iki anahtardan birincisi olan açık anahtar herkese açıktır ve bu anahtarı kullanarak herkes bir veriyi şifreleyebilir. Ancak şifrelenen bu verinin şifresini çözecek olan gizli anahtar yalnızca verinin sahibi olan kişide bulunur ve ancak o kişi bu veriyi çözebilir. Bu sayede kendine gelen tüm verileri başkaları göremeden yalnızca kendisi görebilir.

Sayısal imzada ise özet algoritmaları bu özel anahtarı kullanarak şifreleme yaparlar ve şifre çözme işlemi yapılamayan bu işlem sonucunda yalnız kişiye ait bir sayısal imza elde edilmiş olur. Kişi bu sayısal imzasını verilere (dokümanlara, e-postalara, vb...) ekleyerek bir dokümanı imzalamış olur. Sayısal imza oluşturulurken yapılacak en küçük değişiklik karşı taraftan çözülemeyeceği için imza üzerinde oynanması ya da taklit edilmesi mümkün değildir. Sayısal imza ile imzalanmış bir doküman karşı tarafa ulaştığında önce alıcı elinde bulundurduğu açık anahtar ile imzanın şifresini çözer ve eline imzayı oluştururken kullanılan veri geçer. Daha sonra bu veriyi kullanarak gelen dokümanı özetler. Eğer imzanın şifresini çözdüğünde oluşan veri ile özet sonucu aynıysa doküman imza sahibinden gelmiş demektir. Aksi takdirde imza taklit edilmeye çalışılmış demektir. Bu sebepten dolayı, birçok ülke yasal düzenlemeler ile sayısal imzayı resmi imza olarak kabul etmiş ve imza sahiplerini imzayı içeren tüm belgelerden sorumlu tutmuşlardır.

E eliptik eğrisi F_q üzerinde olsun ve P noktası bu eliptik eğrinin mertebesi asal, N, olan bir noktası olsun. Her kullanıcı [1, N-1] aralığında rasgele bir x tamsayısı seçer. $Q=xP$ noktası bu kullanıcının açık anahtarı, x ise gizli anahtarıdır. Özet değeri H, $1<H<N-1$, olan bir M mesajını eliptik eğri sayısal imza algoritmasını kullanarak imzalamak için şunlar yapılmaktadır.

1. [1, *N*-1] aralığında rasgele bir k tamsayı seçilir.

2. $kP = (x_1, y_1)$ noktası hesaplanır ve $r = x_1 \ mod \ N$ değeri atanır. Burada *P* noktasının x_1 bileşeni, üzerinde tanımlanan sonlu cismin elemanı olup, tamsayıya çevrilmelidir. Eğer $r = 0$ çıkarsa birinci adıma dönülüp tekrar işlemlere başlanmalıdır.

3. $k^{-1} \ mod \ N$ hesaplanıp $s = k^{-1}*(H+xr) \ mod \ N$ değeri bulunur. Eğer $s = 0$ olursa birinci adıma tekrar gidilmelidir.

Mesajın eliptik eğri kullanılarak oluşturulan imzası *(r,s)* ikilisidir. Üretilen bu sayısal imzayı doğrulamak için ise aşağıdaki işlemler yapılmalıdır.

1. Mesajı imzalayıp gönderen kişinin *Q* açık anahtarı alınır.

2. *r* ve *s* sayılarının *[1, N-1]* aralığında olduğu doğrulanıp mesajın *H* özet değeri hesaplanır.

3. $u_1 = s^{-1}H \ mod \ N$ ve $u_2 = s^{-1}r \ mod \ N$ hesaplanır.

4. $u_1P+u_2Q = (x_0+y_0)$ hesaplanır ve $v = x_0 \ mod \ N$ bulunur.

5. Eğer $v = r$ ise imza doğrulanır. Aksi takdirde mesaj o kişiden gelmemiştir [9].

6.2.2. Açık metnin eliptik eğri üzerinde yer alan bir noktaya gömülmesi

Bu kısımda, açık metin birimlerinin E eliptik eğrisi üzerinde, sonlu cisim olarak tanımlanmış F_q üzerinde, noktalar halinde nasıl çözüleceği incelenmiştir [10]. Öncelikle "m" açık metin biriminin $0 \leq m \leq M$ arasında tamsayılar olarak uygun bir şekilde çözüldüğünü varsayıyoruz. Aradığımız tek koşul "m" in E üzerinde P_m tanımlama noktasında kolayca belirlenebilmesidir.

Başlangıçtan itibaren dikkat edilmesi gereken, eliptik eğri üzerinde, metin birimlerinin noktalar halinde yerleştirilmesi için bilinen bir "saptanabilir polinomsal" zaman algoritması olmamasıdır. Buna rağmen aşağıda tanımlayacağımız bir istatistiksel yöntem vardır. Tanımlayacağımız yöntemin $\frac{1}{2^K}$ kadar hata verme olasılığı vardır ve *K* genelde *30≤ K ≤50* arasında değer alan bir tamsayıdır. Burada *p* asal sayı ve $q=p^r$ 'nin büyük ve tek sayı olduğunu varsayıyoruz. F_q cismi *q>M*K* olacak şekilde seçilir. Böylece tamsayılar 1'den *M*K'* ya kadar *m*K+j, 1≤ j ≤K*, şeklinde yazılabilir ve tamsayılarla F_q'nun elemanlar kümesi arasında bir bire-bir örten fonksiyon oluşturabiliriz.

Verilen bir m metin birimini kullanarak, her bir *j=1,2,.,K* için, *m*K+j* değerine karşılık gelen bir $\alpha \in F_q$ elemanı elde ederiz. α'yı bir kez elde ettiğimizde, $y^2 = f(\alpha)$'yı formülünü kullanarak, $y^2=x^3+ax+b$ eşitliğinin sağ tarafını elde ederiz. Daha sonrada y'yi bulabiliriz. Elde ettiğimiz x ve y değerlerini yerine koyduğumuzda bize $y^2=f(x)=x^3+ax+b$ eliptik eğrisindeki $P_m = (x,y)$ noktasını verir. Eğer $y^2 = f(\alpha)$'nin p modülüne göre karekökleri yoksa o zaman *j* değerini bir artırıp işlemleri tekrardan yaparak F_q alanında bir $f(\alpha)$ değeri bulunur. Bu yaklaşımda dikkat edilmesi gereken husus, *j*'nin değeri *K*'dan büyük olursa P_m noktasından m metnini elde edemeyeceğimizdir.

Bir sonraki aşama, metin birimi m'in E üzerinde bulunduğu P_m tanımlama noktasından geri alınmasıdır. Bunu yapmak, P_m noktasındaki x koordinatına uyan tamsayının $\tilde{x}$ olduğu yerlerde $m=\left[\frac{\tilde{x}-1}{\kappa}\right]$ formülünü kullanarak çok kolay bir hale dönüşür.

6.2. Örnek

F_{751} cismi üzerinde E, $y^2=x^3-x+188$ eliptik eğrisi olsun. Bu eğri F_{751} üzerinde *N=727* tane noktaya sahiptir. Varsayalım ki düz mesaj üniteleri 0'dan 9'a

kadar desimal rakamları ve 10'dan 35'e kadar da A'dan Z'ye kadar olan harfleri simgelesin. *K*=20 alalım.

a) Eğride noktalar halinde verilen aşağıdaki mesajı çözmek için yukarda tanımlanan yöntemi kullanınız.

"GAZIUNIVERSITESI"

b) (283,54), (421,362), (283,54), (401,153), (581,20), (543,61), (484,161), (461,92), (361,8), (401,153) noktalarını cevap metni haline çeviriniz.

Çözüm

a) Mesaj "16 10 35 18 30 23 18 31 14 27 28 18 29 14 28 18" sayısal eşitliklerine sahiptir. *j=1*: m=16 ile başlıyoruz ve *x = 16*20+1= 321'* i deniyoruz. Görüyoruz ki *f(x)* ≡ 486 mod 751: $=y^2$ dir. Daha sonra f(x)'in mod 751'de karekökünün olup olmadığını bulalım. Gerçekte 321 mod 751'de kökü olmayan bir sayıdır. Bundan dolayı yerine *j=2*'yi deneriz ve bu sefer x = 322, y^2 =409 buluruz, ki bunlar da mod 751'de kökü olmayan bir sayıdır. Kökü bulana kadar j değerlerini arttırıyoruz, j=4 için x = 324, y^2 = 49 bulunur ki, bunlar da var olan değerlerdir ve karekökü de √49 mod 751 = 7'dir. Bundan sonra $y^2=x^3-x+188$ eğrisinde *(324,9)* noktası olarak, "G" 'i yerleştiririz. Benzer yöntemi kullanarak geri kalan (201,5), (701,203), (361,8), (603,164), (461,92), (361,8), (621,155), (283,54), (543,61), (562,201), (361,8), (581,20), (283,54), (361,8) noktalarını alırız.

b) Bu bölümde daha önce verilen formülü kullanarak cevap metnini geri alırız. Böylece ilk nokta için $m = \left[\frac{283-3}{20}\right] = 14 \rightarrow$E olur. Diğerlerine de aynı işlemi uyguladığımızda ELEKTRONIK açık metnini elde ederiz.

6.3. Eliptik Eğri Kriptografisinin Güvenliği

ECC'nin güvenliği, *kP* ve *P* verildiğinde *k* değerini elde etmenin zorluğuna bağlıdır. Bu durum, eliptik eğri logaritması problemi olarak adlandırılır. Sayısal imza algoritmasındaki (DSA) veya RSA'ye bağlı sayısal imzadaki güvenlik seviyesini yakalamak için *N* sayısı yaklaşık olarak 160 bit olmalıdır. Böylelikle eliptik eğri imzası daha küçük bir sonlu cisimde işlem yaparak ayrık imza algoritması ve aynı güvenliği sağlar. Çizelge 6.1'de, eliptik eğri yöntemi ile RSA yönteminde kullanılan imza büyüklüğüne bağlı olarak gerekli olan açık ve kapalı anahtar uzunlukları karşılaştırılmıştır. Çizelgeden de anlaşılacağı üzere, RSA'nın sağladığı direnci ECC, çok daha düşük anahtar boyutları ile sağlamaktadır. Bu yüzden ECC, düşük anahtar boyutu ile sağladığı yüksek güvenlik sayesinde RSA'ya karşı büyük bir hesapsal üstünlük sağlamaktadır [11].

Çizelge 6.1. ECC ile diğer kripto sistemler arasındaki güvenlik aralığı

	Bant Genişliği		Anahtar Uzunluğu	
	2000 bitlik uzun mesajlar için imza büyüklüğü (bit)	100 bit uzunluğundaki mesajın şifrelendikten sonrası uzunluğu	Açık Anahtar (bit)	Gizli Anahtar (bit)
RSA	1024	1024	1088	2048
DSA	320	2048	1024	160
ECC	320	321	161	160

7. ÖZET FONKSİYONLARI

Özet fonksiyonları h : {1, 2, . . . , 2^m } → {1, 2, . . . , 2^n } ve aşağıdaki özelliklere sahip olan fonksiyonlardır:

1. Sıkıştırma: *h* fonksiyonu, uzunluğu sonlu ve değişken olabilen girdiyi alıp sabit bir uzunlukta çıktı vermelidir,

2. Kolay hesaplanabilirlik: herhangi bir girdi için *h(x)* değerini hesaplamak kolay olmalıdır.

Özet fonksiyonları, anahtarsız ve anahtarlı özet fonksiyonları olmak üzere ikiye ayrılır:

1. Anahtarsız özet fonksiyonları $h : \{0, 1\}^* \rightarrow \{0, 1\}^n$

- Blok şifreleme sistemleri tabanlı
- Modüler aritmetik tabanlı
- İsteğe uyarlanmış (MD4, MD5, SHA-1, RIPE-MD, HAVAL)

2. Anahtarlı özet fonksiyonları $h_k : \{0, 1\}^* \rightarrow \{0, 1\}^n$

- Blok şifreleme sistemleri tabanlı
- Anahtarsız özet fonksiyonları tabanlı
- İsteğe uyarlanmış (MAA, MD5-MAC)
- Akan şifreler için üretilen

İsteğe uyarlanmış özet fonksiyonları sadece özet için kullanılan anahtarlı veya anahtarsız olarak üretilen özet fonksiyonlardır. Ayrıca güvenilirliği teorik olarak ispatlanan fakat pek pratik olmayan evrensel özet fonksiyonları da farklı bir grup olarak görülebilir.

Anahtarsız özet fonksiyonlarının üç temel özelliği aşağıda belirtilmiştir (h bir özet fonksiyonu, *x* ve *x*' girdileri, *y* ve *y'* çıktıları göstermektedir):

1. Ön görüntü Direnci (preimage resistance): *h(x) = y* değeri bilindiğinde, *x*'i hesaplamak sonlu zamanda mümkün değil. *y* biliniyor, h(x') = y olacak bir x' bulmak zor (hesaplamak sonlu zamanda mümkün değil).

2. İkincil Ön görüntü Direnci (2nd-preimage resistance): *h(x) = y* biliniyor, *h(x') = y* olacak farklı bir mesaj x ≠ x' bulmak zor.

3. Çarpışma Direnci (collision resistance): h(x) = h(x') olacak şekilde iki farklı mesaj x ve x' bulmak zor.

7.1. Örnek

Mod 32 kontrol toplamları (Mod-32 checksum). Mesajın içerisindeki bütün 32-bit'lik parçaların toplamı alınarak kullanılan fonksiyondur. Hesaplaması kolay, sıkıştırma var, fakat ön görüntü dirençli değildir.

7.2. Örnek

$g(x) = x^2 \bmod n = p*q$ p, q büyük asal sayılar (n'nin çarpanları bilinmiyorsa tek yönlü fonksiyondur.) Hesaplaması kolay, sıkıştırma yok, ön görüntü dirençli (çünkü ön görüntüyü bulmak n'yi çarpanlarına ayırmaya denk), fakat ikincil ön görüntü ve çarpışma var (x, -x).

7.3. Örnek

DES tabanlı tek yönlü fonksiyon. Sabit bir k anahtarı için, $f(x) = E_k(x) \oplus x$ olsun. E rasgele bir permütasyon olarak kabul edilirse, f fonksiyonu tek yönlü olur. y bilindiğinde E'nin rasgele olamamasından dolayı, $y = E_k(x) \oplus x$ olacak şekilde $x (= E_k^{-1} (x \oplus y))$ ve k değerlerini bulmak zordur. Dolayısıyla f tek

yönlü bir fonksiyondur. Fakat fonksiyon belli mesaj uzunlukları için çalışmaktadır.

- çarpışma dirençli ise ikincil ön görüntülü dirençlidir:

Fonksiyonumuzun çarpışma dirençli olduğunu kabul edelim. İkincil ön görüntülü dirençli değilse $\Rightarrow$ Sabit x; h(x) için h(x) = h(x') olan $x \neq x'$ bulabiliriz, fakat bu çarpışma direnci olamadığını gösterir, kabulümüzle çelişir.

- çarpışma dirençli ise ön görüntülü dirençli olmak zorunda değildir:

$g : (0, 1)^* \rightarrow (0; 1)^n$ çarpışma dirençli olsun, h fonksiyonu aşağıdaki şekilde tanımlanırsa ön görüntülü dirençli olmaz;

$$h(x) = \begin{cases} 1 \parallel x & \text{if } |x| = n \\ 0 \parallel g(x) & \text{if } |x| \neq n \end{cases} \qquad (7.1)$$

$h : (0, 1)^* \rightarrow (0, 1)^{n+1}$ n + 1 bit özet fonksiyon.

- Ön görüntülü dirençli ise ikincil ön görüntülü dirençli olmak zorunda değildir.

7.1. Ekstra Şartlar

1. İlintili olmayan (Non-correlation): Girdi ve çıktı bitleri arasında ilinti olmamalı, blok şifre sistemlerindeki gibi çığ (avalanche) özelliği sağlanmalı (bütün girdi bitleri bütün çıktı bitlerini etkilemeli),

2. Yakın çarpışma direnci (Near-collision resistance): $w(h(x) \oplus h(x'))$ küçük olacak farklı x ve x' çiftlerini bulmak zor olmalı (w:hamming ağırlığı),

3. Kısmi ön görüntü direnci (Partial-preimage resistance): Girdi bitlerinin bir kısmını dahi bulmak zor olmalı, girdinin t uzunluğundaki kısmını bulmak için yaklaşık 2^t - 1'lik hesaplama yapmak gerekmeli (girdinin belli bir kısmı bilinse dahi diğer kısmını bulmak zor olmalı).

Anahtarsız özet fonksiyonlarının çoğu girdi ve çıktı uzunluğu sabit olan bir *f* özet fonksiyonunun tekrarlı olarak uygulanmasıyla elde edilir. Bu fonksiyonlara yinelemeli özet fonksiyonları (h) adı verilir. Herhangi bir uzunluktaki x girdisi, sabit r-bit uzunluklara bölünür (x_i), x'in uzunluğunun r'nin katı olması için belli bir kurala bağlı olarak x'e bit eklenir (padding). Girdi parçaları x_i'ler sırasıyla f'ye sokulur, f'nin çıktısı ve x_{i+1} tekrar f'nin girdisi olarak kullanılır ve son girdi bloğuna kadar bu işlem tekrarlanır. Bu durumda aşağıdaki işlemler yapılmış olur: $x = x_1 x_2 \ldots x_t$,

$H_0 = IV; H_i = f(H_{i-1}, x_i), 1 \leq i \leq t; h(x) = g(H_t),$

IV: başlangıç değeri

Yinelemeli özet fonksiyonlarının genel ve detaylı yapıları aşağıdaki şekillerde verilmiştir:

Not: f fonksiyonunun çarpışma dirençli olması h fonksiyonunun çarpışma dirençli olmasını garantiler.

7.2. Anahtarsız Özet Fonksiyonları

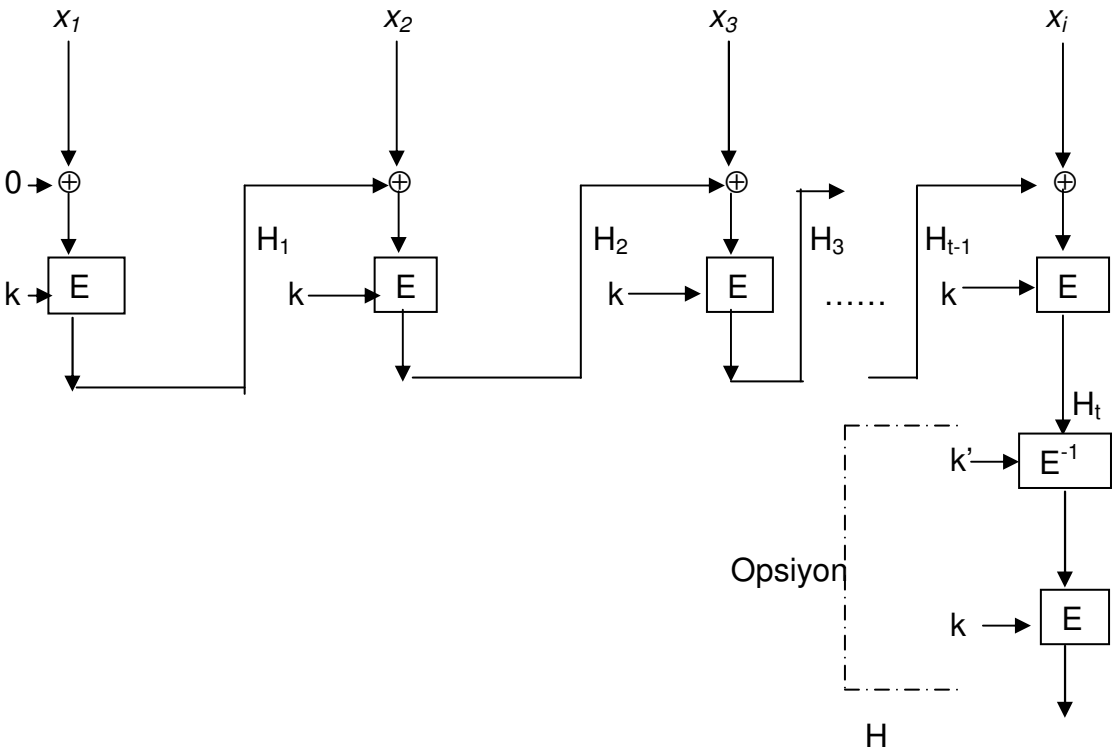
- Blok şifre sistemleri tabanlı: Yinelemede kullanılan *f* fonksiyonu herhangi bir blok şifre sistemi olarak seçilir. Kullanılan makinanın içinde bir blok şifreleme sistemi varsa özet fonksiyonu olarak da kullanılabilir.

- Modüler Aritmetik tabanlı: Yineleme fonksiyonu (f) mod M aritmetiğini baz alan bir fonksiyon olarak seçilir, çarpanlara ayırma ve ayrık logaritma problemlerini temel alan sistemler seçilebilir.

- İsteğe uyarlanmış: Özel olarak özet için tasarlanmış ve optimize hıza sahip olan fonksiyonlardır. Pratik olarak kullanılmaktadır, MD ailesi ve SHA örnek olarak verilebilir. Güvenilirlikleri hesaplama gücüne dayalı olarak ispatlanır, matematiksel olarak güvenilir oldukları ispatlanmamıştır.

7.3. Anahtarlı Özet Fonksiyonları

- Blok şifreleme sistemleri tabanlı: Şifre Blok Zincir Mod (CBC) tabanlı Mesaj Doğrulama Kodlar (MAC) örnek olarak verilebilir.



Şekil 7.1. CBC tabanlı MAC algoritması

- Anahtarsız özet fonksiyonları tabanlı: Gizli bir anahtarın anahtarsız özet fonksiyonlarının girdisinin bir parçası olarak kullanılmasıyla üretilen özet fonksiyonlarıdır.

Aşağıdaki yöntemler örnek olarak verilebilir (k anahtar ve h bir anahtarsız özet fonksiyonu olmak üzere):

1. Gizli önek yöntemi: $M(x) = h(k \| x)$ (7.2)

2. Gizli sontakı yöntemi: $M(x) = h(x \| k)$ (7.3)

3. Ekleme ile zarflama yöntemi: $h_k(x) = h(k \| p \| x \| k)$ p: ekleme $k \| p$ bir blok uzunluğunda olacak şekilde ekleme yapılıyor,

4. Özet tabanlı: $HMAC(x) = h(k \| p_1 \| h(k \| p_2 \| x))$, p_1, p_2 : ekleme, $k \| p_1$ ve $h(k \| p_2 \| x)$ birer tam blok uzunluğunda olacak şekilde bit ekleme yapılıyor.

- İsteğe uyarlanmış MAC'lar: Sadece özet yapmak için tasarlanmış ve içerisinde gizli anahtar barındıran özet fonksiyonlarıdır. Örnek olarak MAA ve MD5-MAC verilebilir.

8. KRİPTOGRAFİK PROTOKOLLER

8.1. Kriptografik Protokol Tanımı

Protokol, basitçe iki veya daha fazla kişi arasındaki önceden belirlenmiş belli bir amaca yönelik haberleşme yöntemi olarak tanımlanabilir. Kriptografik protokol ile kriptografik bir algoritma içeren bir protokol kastedilir [12].

8.2. Kriptografik Protokollerin Özellikleri

Kriptografik algoritmalarda olduğu gibi bir protokolün de güvensiz olduğunu ispatlamak güvenliliği ispatlamaya göre çok daha kolaydır. Bir protokolü incelerken yine tıpkı algoritmalardaki gibi protokolü nasıl bir cihazda kullanacağımızdan çok temel çalışma prensipleri ile ilgilenilir.

8.3. Hakem (Arbitrator) ve Düzenleyici (Adjudicator)

Kimi zaman protokollerde haberleşmenin düzgün bir şekilde işleyebilmesi için hakem olarak adlandırdığımız güvenilir 3. kişilere ihtiyaç duyulur. Hakemlerin en önemli özelliği tüm protokolün onların gözetiminde yürütülüyor olmasıdır. Hakemi çıkardığımız zaman protokol işlemez. Düzenleyici ise hakemin aksine sadece bir anlaşmazlık durumunda başvurulan güvenilir 3. kişilere denir.

8.4. Düzenleyicili (Adjudicated) Protokoller

Bu protokoller temelde tarafların dürüstlüğüne dayanır. Bir anlaşmazlık ya da birisinin hile yapması durumunda düzenleyici kişi bunu fark eder. İyi bir düzenleyicili protokolde düzenleyici aynı zamanda hile yapan tarafın kim olduğunu da fark eder. Dolayısı ile bu kişinin varlığı, hile önünde engel teşkil eder.

8.5. Kendi İşleyişini Zorlayan (Self-Enforcing) Protokoller

Protokol herhangi bir hileye yer bırakmaz. Protokolün sorunsuz işlemesi otomatik olarak herhangi bir hile yapılmadığı anlamına gelir. Hile halinde protokol durur. Devam edilemez bir hal alır ve bunu herhangi bir hakem ya da düzenleyiciye ihtiyaç olmadan sağlayacak şekilde tasarlanmıştır. Bir protokol için her zaman istenen bir özelliktir. Ancak her durumda kendi işleyişini zorlayan bir protokol bulmak mümkün olamayabilir.

8.6. Aktif Ve Pasif Ataklar (Attacks)

Bir protokol işlerken her zaman atak olması olasıdır. Pasif atakta kötü amaçlı kişi sadece arada gelip giden trafiği dinleyerek protokol tasarlanırken kendisinin ulaşması umulmayan bir bilgiye erişmeye çalışır. Aktif atakta ise kötü niyetli kişi sadece dinlemekle kalmaz. Mesajları ya da kayıtları okumanın ötesinde kesebilir, bozabilir ya da değiştirebilir. Aktif atak yapan kişi tamamen dışardan birisi olmak zorunda değildir. Sistemdeki başka yasal bir kullanıcı ve hatta sistem yöneticisi olabilir. Atak yapan kişinin protokolü uygulayan taraflardan birisi olması durumunda ise daha çok hile ve atağı uygulayan kişi içinse hilekâr tabirleri kullanılır. Atak yapan kişilerde olduğu gibi hilekârları da pasif hilekâr ve aktif hilekâr olarak ikiye ayırmak mümkündür.

8.7. Tipik Simetrik Algoritma Haberleşme Protokolü

Simetrik kriptografi kullanılarak yapılan tipik bir haberleşmede A ve B kişileri bir kripto sistem ve bir anahtar üzerinde anlaşırlar ve A kişisi bu anahtarı kullanarak şifrelediği mesajını B kişisine gönderir. B kişisi de yine aynı anahtarı kullanarak mesajın şifresini çözer ve haberleşme tamamlanır.

Bu sistemin önemli dezavantajları vardır. Öncelikle A ve B kişileri anahtar değiştirmek için bir araya gelmelidirler (anahtar değişimi için başka bir algoritma kullanılmadığı varsayılmaktadır). Araya giren kötü niyetli Z kişisi

mesajlara ulaştığı takdirde mesajın diğer tarafa gitmesini engelleyebilir ve hatta anahtarı bilmesi durumunda bu mesajı kendisininkilerle değiştirerek haberleşen iki tarafa da farkında olmaksızın bambaşka mesajlar gönderebilir.

8.8. Tipik Açık Anahtar Kripto Sistemi ile Haberleşme

A ve B kişileri bir açık anahtar kripto sistemi üzerinde anlaşırlar. Simetrik anahtarlı sistemde olduğu gibi burada kripto sistem üzerinde anlaşma gizli yapılmak zorunda değildir.

B kişisi A kişisine kendi açık anahtarını gönderir. A kişisi mesajını B'nin açık anahtarı ile şifreler ve gönderir. B kişisi ise kendi gizli anahtarı ile mesajı çözer ve mesaja ulaşır.

Açık anahtarlı tipik haberleşme sistemlerinin en önemli dezavantajı ortalama olarak simetrik bir algoritmadan 1000 kat yavaş olmalarıdır. Uzun mesajları bu yolla göndermek pratik değildir.

8.9. Tipik Karma (Hybrid) Kripto Sistemler

Bir karma şifre hem simetrik şifreyi hem de açık anahtarlı şifreyi kullanır. Bu mekanizma, bir simetrik şifreyi paylaştırmakta bir açık anahtarlı şifreyi kullanır. B, A'ya açık anahtarını gönderir. A bir oturumda kullanılmak üzere rasgele bir oturum anahtarı üretir. Oturum anahtarını B'nin açık anahtarı ile ve gönderilecek mesajı da oturum anahtarı ile şifreleyerek her ikisini de otomatik olarak bir paket içinde birleştirir ve B'ye gönderir. B kişisi ise gizli anahtarı aracılığıyla edindiği oturum anahtarını açar ve elde ettiği oturum anahtarı ile mesajın şifresini çözer.

Tahmin edileceği üzere bu sistemde taraflar, simetrik kripto sistem anahtarı belirlemek için bir araya gelmek zorunda kalmamaktadırlar.

8.10. Günlük Hayatta İmza

Günlük hayatta sıkça kullandığımız imzalarda aradığımız bizim için çok önemli özellikleri;

- İmza otentiktir (authentic), imzalayanın kimliğini gösterir.
- İmza çoğaltılamaz.
- İmza tekrar kullanılamaz.
- İmza değiştirilemez, üzerinde oynama yapılamaz.
- İmza atan kişi attıktan sonra imzasını inkâr edemez olmasıdır.

8.10.1. Açık anahtar kripto sistemlerle sayısal imza

Tipik bir açık anahtar kripto sisteminde A kişisi kendi gizli anahtarı ile imzalamak istediği dokümanı şifreler. B kişisi ise A'nın açık anahtarı ile dokümanı açar. Eğer açamazsa imza geçerli değildir, imza birisi tarafından ya da doğal etkenlerle bozulmuştur.

Bu basit imza protokolüne kötü taraftan bakacak olursak imzalı dokümanı alan kişi sayısal ortamda bu imzalı dokümanı dilediği kadar çoğaltabilir. İmzalı doküman bir anlaşma metni ise bu elbette ciddi bir sorun teşkil etmeyebilir ancak dokümanın para karşılığı olan bir çek olduğunu kabul edersek durum farklı olabilir. B kişisinin farklı zamanlarda farklı çeklermiş gibi aynı imzalı çeki bozdurmasını istemeyiz. Bu gibi tekrar kullanımları önlemek için sayısal dokümanlar çoğu zaman hangi tarihe ait olduklarını gösteren bir zaman pulu (timestamp) ile beraber kullanılır. Mesaj algoritma ile imzalanıp kapatılmadan önce içine tarih bilgileri de eklenir.

8.10.2. Açık anahtar kripto sistem ve özet fonksiyonuyla imza

A kişisi bu defa dokümanın kendisini (zaman pulu ile veya zaman pulsuz) imzalamak yerine dokümanın bir özet fonksiyonu sonucu üretilmiş özet değerini imzalar. A kişisi imzalamak üzere kullandığı dokümanla beraber imzalanmış özet değerini B'ye gönderir.

Alıcı olan B ise imzayı A'nın açık anahtarı ile açtıktan sonra dokümanın özetini kendisi hesaplar ve bu değerin gönderilen özet değeri ile uyup uymadığına bakar. Eğer uyuyorsa imza geçerlidir. Aksi halde geçersizdir.

Bu protokol dokümanın kendisi yerine sadece özet değerini imzaladığımız için dokümanın boyutuna bağlı olarak bir öncekine göre inanılmaz ölçüde hızlı olabilir. İki farklı dokümanın özet değerlerinin örneğin 160-bit'lik bir çıktı üreten bir özet fonksiyonunda $1/2^{160}$ olduğu düşünülürse özet değerini imzalamakla dokümanın kendisini imzalamanın rahatlıkla eşleştirilebileceği ortaya çıkar. Ancak özet fonksiyonunun temel özellikleri sağlaması gerektiği unutulmamalıdır. Şayet tek-yönlü olmayan bir özet fonksiyonu kullanılsaydı, bir doküman imzalandığı zaman onunla aynı özet değerini veren tüm dokümanlar da beraberinde aynı kişi tarafından imzalanmış olacaktı.

Bu yöntem ile A kişisi bir dokümanı açığa çıkarmadan dokümana sahip olduğunu ispatlayabilir. Örneğin 'copyright' hakkına sahip olmak isteyeceğimiz bir dokümanı dokümanın kendisini açığa çıkarmadan imzalayabiliriz.

8.10.3. Sayısal imzalarda inkar edememezlik (Nonrepudiation)

Yukarıdaki algoritmanın eksik bir yanı, belgeyi imzalayan kişiye her zaman hile olanağı vermesidir. Şöyle ki, eğer dokümanda bir zaman pulu mevcut değilse A kişisi dokümanı imzaladığını inkâr etmek istediği bir durumda her zaman için gizli anahtarını herhangi bir sayısal ortamda bırakarak daha sonra

da gizli anahtarının çalındığını ve dokümanın kendisi değil de imzasını çalan kişi tarafından imzalandığını öne sürebilir. Bu gibi durumlarda çoğu zaman 3. güvenilir kişi olan hakemlerin kontrolünde olan bir protokole ihtiyaç duyulabilir.

8.10.4. İnkar edilemeyen sayısal imzalar

Protokol bu defa bir parça daha karışıktır. 3. güvenilir T kişisine dayanır. A kişisi mesajı imzalar ve mesaja kendisini tanımlayan bir başlık ekler. Elde ettiği bu yeni birleştirilmiş mesajı tekrar imzalar ve T kişisine gönderir. 3. kişide herkesin açık ve gizli anahtarı bulunmaktadır. Dolayısıyla dıştaki mesajı A kişisinin açık anahtarı ile açarak A kişisinin kendisini tanımladığı eklentiye erişir. T daha sonra mesajı gerçekten A kişisinin gönderdiğini B'ye onaylamak için mesajın ne zamana ait olduğu bilgisini de ekleyerek imzalar ve hem A'ya, hem de B'ye gönderir. B kişisi T'nin imzasını açar ve A'nın kimlik bilgilerine ulaşarak imzayı onaylar. İmzanın A'ya da gönderiliyor olmasının sebebi bir anlamda A'ya "Senin adına şöyle bir mesaj gönderildi, şayet bunu gerçekten gönderen sen değil de (örneğin anahtarını çalan) bir başkası ise acele konuş, daha sonra bunu inkar edemezsin" denmesidir.

Bahsedilebilecek bir nokta da şudur ki bu protokolde zaman bilgileri açısından T'ye güvenilmektedir. Dolayısı ile T ile bir şekilde anlaşan birisi dokümanının tarihi hakkında hileye girişebilir. Protokolde B'nin A'dan mesajı aldıktan sonra gerçekten ona dair olup olmadığını onaylaması için T'ye sorması gibi değişiklikler yapılması suretiyle alternatifleri de türetilebilir.

8.10.5. Şifreleme ve sayısal imza

A ve B kişilerinin her ikisinin de açık ve gizli anahtarlarının bulunduğu bir ortamda bir mesajı hem şifrelemek ve hem de imzalamak istediğimiz bir durumda A kişisi, mesajı, basitçe söylemek gerekirse önce kendi gizli anahtarı ile imzalar ve daha sonra da imzalanmış mesajı B'nin açık anahtarı

ile şifreleyerek B'ye gönderir. B de mesajı kendi kapalı anahtarı ile şifreyi çözer ve daha sonra da A'nın açık anahtarı ile imzayı onaylar. Şayet mesajı sorunsuz bir şekilde aldığını A'ya iletmek isterse A'nın yaptığı işin tersini yaparak mesajı tekrar A'ya gönderebilir. Açıkça ifade etmek gerekirse, B kişisi mesajı kendi gizli anahtarı ile imzalayıp daha sonra da A'nın açık anahtarı ile imzayı şifreleyerek mesajı A'ya gönderebilir. A'nınsa geri ters operasyonlarla gönderdiği orijinal mesajı elde etmesi durumunda işlemlerin başarıyla gerçekleştiği kabul edilir.

Bu operasyonlarda mesajın şifrelenmeden önce imzalanması doğal gözükmektedir. Tersi bir sıra uygulanması durumunda tıpkı kağıt yerine zarfı imzalayan birisinin, bir başkasına zarfın içinden kağıdı çıkararak başka bir kağıt koyup, koyduğu kağıdı imzalanmış göstermesi gibi bir imkan verdiği açıktır.

Bu protokol de bir takım ataklara maruz kalabilir. Örneğin başkalarından gelen rasgele mesajların imzalanmaması gerekir. Aynı şekilde başkalarından gelen rasgele mesajların şifresi çözülüp sonucun başkalarına verilmesi de direkt olarak atağa meydan verebilmektedir. Ancak genel olarak şifreleme ve imzalama için farklı algoritmalar kullanıldığı takdirde ya da aynı algoritma kullanılsa bile bu iki operasyon için farklı anahtarlar kullanıldığı zaman bu gibi basit bir atağa meydan verilmediği söylenebilir. Benzer şekilde zaman pulu eklentisi de mesajları farklılaştırdığı için özet fonksiyonlarını burada kullanmak atağa karşı iyi bir çözüm olabilmektedir. Protokol, değişik amaçlar ve ortamlar doğrultusunda şekillendirilebilir.

8.11. Anahtar Değişimi (Key Exchange)

Kriptografide en temel uygulamalardan birisi anahtar değişimidir. Standart şifreli haberleşme için, karşılıklı anahtar değişimi başarı ile gerçekleşmişse, çoğu zaman herhangi bir güvenilir 128-bit (ya da daha geniş) blok şifre kullanılarak iletişim sağlanabilir. Ancak değinildiği gibi öncelikli olarak anahtar

değişiminin güvenilir bir şekilde gerçekleşmesi gereklidir. Sadece belli bir oturum (ya da haberleşme seansı) için kullanılan anahtara oturum anahtarı denir.
Anahtar değişimi için simetrik algoritmalar kullanılırsa, güvenilir üçüncü kişinin de yardımıyla anahtar değişimini gerçekleştirmek mümkündür. Ancak burada, ilk aşamada güvenilir 3. bir kişiye ihtiyaç bırakmayan, açık anahtar yöntemiyle gerçekleştirilen ve çok daha işlevsel olan bir yöntem ele alınacaktır.

8.11.1. Açık anahtar yöntemi ile anahtar değişimi

Bu uygulamada A kişisi önce B'nin açık anahtarını temin eder, rasgele oluşturduğu oturum anahtarını B'nin açık anahtarı ile şifreleyerek B'ye gönderir. B kişisi ise kendi gizli anahtarı ile A'nın mesajını açar ve haberleşmelerini gönderilen oturum anahtarını kullanarak, simetrik bir algoritma aracılığı ile yürütürler. Burada açık ve kapalı anahtarlar bir açık anahtar algoritmasına, oturum anahtarı ise bir simetrik şifreleme algoritmasına (DES, SAFER, Vigenere vb) aittir.

Burada pasif bir saldırgan için, yani herhangi bir müdahalede bulunmayan kötü niyetli kişi E için kullanılan açık anahtar algoritmasını kırmaya çalışmaktan başka bir yol görünmemektedir. Ancak aktif saldırgan için aynısı söz konusu değildir. Aktif saldırgan, eğer iki kişi arasındaki mesajlara erişme ve hatta değiştirme imkânına sahipse, B kişisi A'ya açık anahtarını gönderirken araya girip ona kendi açık anahtarını gönderebilir.

A kişisi de oturum anahtarını, B'nin anahtarı sandığı bu anahtarla şifreleyerek geri gönderdiğinde ise, kötü niyetli kişi zaten kendi açık anahtarı ile şifrelenmiş bu mesajı rahatlıkla açıp okuyabilir. Dahası E, mesajı B'nin gerçek açık anahtarı ile tekrar şifreleyerek B'ye göndererek onların hiçbir şey olmamışçasına anahtarı kendisinde olan bir simetrik algoritmayla haberleşmelerini sağlayabilir. Aktif saldırgan, hissedilir bir yavaşlamaya

sebep olmadığı sürece bu şekilde A ve B kişilerinin haberi bile olmadan onların şifreleyerek gönderdikleri tüm mesajları okuyabilir. Ortadaki 3. kişi tarafından gerçekleştirilen bu atağa "ortadaki-adam atağı" (man-in-the-middle attack) adı verilir.

Burada protokolün adım adım işlemesi yerine A kişisi tek bir gönderimle, B'nin açık anahtarı ile şifrelediği oturum anahtarını, o oturum anahtarı ile şifrelediği mesajla birlikte B'ye gönderebilir. B de önce kendi gizli anahtarı ile oturum anahtarını elde eder, sonra da bu anahtarı kullanarak simetrik algoritma ile şifrelenmiş mesajı açarak orjinal mesaja ulaşır.

8.12. Arakilit Protokolü (Interlock Protocol)

Yukarıda değinilen ortadaki-adam atağı, A ve B kişilerinin gerçekten birbirleri ile konuştuklarını onaylama şansları yokken kullanılabilir. Arakilit protokolünde A ve B kişileri birbirlerine karşılıklı olarak açık anahtarlarını gönderirler. Burada aktif saldırgan hala araya girip tarafların açık anahtarlarını kendi belirlediği bir açık anahtar ile değiştirebilir. Ancak sonrasında protokol farklı işler. A kişisi oturum anahtarını (veya herhangi bir mesajı da olabilir) yine B'nin açık anahtarı ile şifreler, ancak bu defa şifrelenmiş mesajın kendisi yerine yarısını B'ye gönderir. Aynı şekilde B kişisi de kendi şifrelediği mesajın ilk yarısını A'ya gönderir. Ve daha sonra, yine önce A kişisi ve sonra da B kişisi olmak üzere her iki taraf da şifrelediği mesajın kalan yarısını birbirine gönderirler ve iki yarımı birleştirerek kendi gizli anahtarlarıyla mesajın hepsinin şifresini çözer.

Burada mesajın ikinci yarısı olmadan ilk yarısı kullanışsızdır. Taraflar da ortadaki adam gibi mesajın ikinci yarısı gelmeden şifreli mesajı çözme şansına sahip değildirler.

Ancak burada açık anahtar ile giden mesaj, mesajın tamamı değildir. Dolayısıyla saldırgan, mesajı kendi gizli anahtarı ile açıp B kişisinin açık

anahtarı ile tekrar şifreleme şansına sahip değildir. Çünkü şifreli mesaj yarım olduğu için mesajı herhangi bir şekilde açma/şifre çözme şansına sahip değildir. Bu özelliği sağlayan yarım mesajın elde edilmesi ve mesajı iki "yarım"a bölmek için kullanılan mekanizma aşağıda belirtilmiştir.

Burada kullandığımız mesajı iki parçaya bölüp gönderme yöntemindeki parçalardan ilki şifreli mesajın bir özet fonksiyonu ile elde edilmiş bir özet değeri olurken ikincisi ise şifreli mesajın kendisi olabilir. Ya da gerçekten bir 64-bit blok şifre algoritması ile şifrelenmiş mesajın 32'şer bit uzunluğunda iki yarısı olabilir. Her iki durumda da ikinci mesaj gelmeden şifre çözme yapılamayacaktır. Ve yine her iki durumda da aktif saldırganın 2. şifreli mesaj yarısını da bilmeden mesajı değiştirmesine olanak yoktur. İkinci şifreli mesaj, taraflardan birisi tarafından gönderildiğinde ise çoktan mesajın onay için kullanılacak olan ilk yarısı diğer tarafın eline ulaşmıştır. Dolayısıyla ortadaki adam için atak vakti çoktan geçmiştir. Nitekim protokolün işleyişi bunu gerektirmektedir. Arakilit protokolü Rivest ve Shamir tarafından geliştirilmiştir.

8.13. Sayısal İmzalarla Anahtar Değişimi

Şu ana kadar mesajımızın kendisini ya da açık anahtarımızı gönderirken aktif bir saldırganın her zaman için araya girip tarafların açık anahtarlarını değiştirmek suretiyle çeşitli ataklarda bulunabileceği kabul edilmiştir. Bu sebeple açık anahtarları dağıtmanın nasıl daha güvenli bir şekilde olabileceği sorusuna cevap olarak 3. güvenilir T kişisinin imzası da protokole dâhil edilebilir. Basit bir ifade ile, A ve B kişilerinin anahtarlarını direkt olarak birbirlerine göndermeleri yerine, 3. güvenilir T kişisi onlar için anahtar belirleyip bu anahtarı imzalayabilir. Ve bu şekilde A ve B kişileri sadece T kişisinden imzalı anahtarlar kullanmak üzere karşılıklı anlaşmışlardır. Ve T kişisinin güvenli olduğunu, yani onun anahtar veri tabanına kimse tarafından erişilemediğini kabul edersek (bu belli ve tek bir anahtar dağıtıcısının güvenliğini sağlamak ayrı ayrı pek çok kullanıcının güvenliğini sağlamaktan

her zaman için daha kolaydır) aktif saldırgan araya girip açık anahtarları kendi anahtarı ile değiştiremez.

Bu arada her zaman için protokol detayları üzerinde düzenlemeler yapılarak protokolde faydalı değişiklikler yapabilmemizin olası olduğu unutulmamalıdır. Örneğin yukarıda açık ve gizli anahtarlarımızı ilk aşamada bizim için T kişisinin değil de kendimizin ürettiğini ve T kişisinin sadece bizim ürettiğimiz anahtarların imzası için protokolde yer aldığını düşünürsek, anahtarı göndereceğimiz B kişisi anahtarımızda her zaman T'nin imzasını arayacağı için ve aktif saldırgan da sayısal imza yöntemlerinin güvenilirliği ölçüsünde T kişisinin imzasını değiştiremeyeceği için açık anahtar gönderimlerimizi çok daha güvenilir bir şekilde yürütebiliriz. Üstelik bu durumda aktif saldırgan T'nin sayısal anahtarını ele geçirse bile bu anahtarı A ve B taraflarının mesajlarını okumakta kullanamaz, sadece yeni açık anahtarlar imzalayabilir. Çünkü T'de sadece bizim açık anahtarlarımızı imzalamada kullandığı bir sayısal imza anahtarı bulunmaktadır.

8.14. Otentikasyon (Kimlik doğrulama)

Bir bilgisayara kendimizi tanıtmak için kullandığımız yegâne yöntem parola kullanmaktır. Ancak ilk kez Needham ve Guy ikilisinin ortaya koyduğu şekliyle otentikasyon yapan bilgisayarda parolaların kendilerinin bulunmasına gerek yoktur. Dolayısıyla sistem yöneticisinin (ya da sizin parolalarınıza direkt erişme şansına sahip birisinin) parolanızı farklı amaçlar için kullanmasını (örneğin aynı parolayı kullandığınız başka bir sisteme sizin adınıza erişmesini) önemli ölçüde zorlaştırmaya yol açan gerçek, bilgisayar için gerekli olan içinde tüm kullanıcı parolalarının bulunması değil, sadece yanlış girilen parolaları diğerlerinden ayırt edebilmesidir. Bu da özet fonksiyonları sayesinde gerçekleşmektedir; kullanıcı otentikasyon yapılacak bilgisayara parolasını girer, bilgisayar bu parolanın özet değerini hesaplar ve kendi veri tabanındaki aynı kullanıcı için tutulan özet değeri ile uyup uymadığını kontrol eder.

Burada otentikasyon yapılan bilgisayara erişimi olan bir saldırgan hala şunu yapabilir; parola olarak kullanılabilecek çok miktarda sözcük üreterek bunların özet değerlerini hesaplar ve veri tabanındakilerle karşılaştırır. Bu yöntemle yeterli miktarda süreye sahip olduğu takdirde tüm olası kelimeleri deneyerek parolalara ulaşabilir. Bu operasyonu pratik olarak zorlaştırmak için tuzlama (salting) işlemi uygulanır. Tuzlama, parolalara özet fonksiyonundan geçirmeden önce rasgele metin eklemektir. Örneğin UNIX sistemlerde 12-bit tuzlama uygulanmaktadır. Tuzlama yöntemiyle otentikasyon işlemini önemli oranda yavaşlatmaksızın bir sözlük atağını çok ciddi biçimde zorlaştırmak mümkündür. Ancak her zaman için kolay tahmin edilebilir parolalar böyle bir atakta önce kırılanlar olacaktır ve tuzlama onları daha güçlü hale getirmemektedir.

8.14.1. Açık anahtar kriptografi ile otentikasyon

Otentikasyonun çok uzaktaki bir makina tarafından yapıldığı bir ortam düşünelim. Bizden birkaç ülke uzaklıkta olan bir internet sitesine kendimizi tanıtmak için kullandığımız parolamızı güvenli olarak nasıl gönderebiliriz? Bunun için yukarıdaki basit otentikasyon protokolümüzü açık anahtar sistemli otentikasyon kullanarak nasıl değiştiririz? Açık anahtar kullanan otentikasyon yönteminde biz parola yerine kendi gizli anahtarımızı bilmekteyiz. Makinada ise bizim açık anahtarımız bulunur. Otentikasyon şöyle işler; makina bize rasgele bir metin gönderir. Biz de gizli anahtarımızla bu metni şifreler ve makinaya geri göndeririz. Makina, bizim açık anahtarımız ile gönderdiğimiz mesajı açtığında kendi gönderdiği rasgele metni bulursa bize sistem erişimi verir. Bu yöntemin kullanıldığı ortamda, ne otentikasyon makinasının (gizli anahtarımız sadece bizde bulunuyor, makina veritabanında mevcut değil), ne de iletişim yolunun (açık anahtar kripto sisteminin temel özelliğinden dolayı) güvenli olması gerekmemektedir.

8.15. Sır Paylaştırma (Secret Splitting)

Bir bilgiyi iki kişi arasında paylaştırmak istediğimizi düşünelim. Öyle ki; her ikisi de bir araya gelmeden bilgiye ulaşamasınlar. Paylaştırılacak olan M mesajı için aynı bit uzunluğunda R gibi bir rasgele metin oluşturduğumuzu düşünürsek M ⊕ R = S şeklinde bir xorlama operasyonu ile S metnini elde ettiğimizi düşünelim. Bu durumda bir kişiye R'yi ve de diğer kişiye de S'yi verdiğimizi düşünürsek her ikisi de bir araya gelmeden M mesajını elde edemezler. Bir araya geldiklerinde ise R ⊕ S = M ile mesajı geri dönüştürebilirler.

Bu basit yöntemi ikiden fazla kişide uygulamak içinse bir sırrı 4 kişi arasında paylaştırmak istediğimizi düşünelim. Bu defa sır metni ile yine aynı uzunlukta R, S ve T rasgele metinlerine ihtiyaç vardır. M ⊕ R ⊕ S ⊕ T = U şeklinde bir operasyonla, R,S,T ve U metinlerini farklı 4 kişiye dağıtarak ancak bir araya geldiklerinde tüm sahip oldukları 4 metni xorlayarak M'i elde etmelerine izin verebiliriz. Dahası bu 4 kişiden birisini saf dışı bırakmak istediğimizde onun metinini diğerlerine dağıtmamız yeterli olacaktır. Bu sayede o olmadan da bu xor operasyonunu yaparak ana metne ulaşabilirler.

8.16. Saklı İletişim Yolu (Subliminal Channel)

Birisiyle haberleşmek istediğiniz, ancak bunun 3. bir başka kişi aracılığıyla gerçekleşmesi dışında mümkün olmadığı bir senaryo düşünelim. Bu 3. kişi de mesajın şifreli olmasına izin vermiyor, kendisi mesajınızı iletmeyi ancak o içeriğini okuyabildiği zaman kabul ediyor. İşte böylesi bir durumda bir saklı iletişim yolu yöntemi kullanımı çözüm olabilir. Yani ilettiğimiz mesajda gizli bir algoritmayla saklanmış, görünenin ardındaki başka bir mesajla hem diğer tarafa mesajımızı ulaştırıp hem de 3. aracı kişinin yalnızca görünürdeki mesajı gönderdiğimizi sanmasına yol açarak gerçekte ne gönderdiğimiz hakkında hiç bir fikri olmamasını sağlayabiliriz.

Çok basit bir saklı iletişim yolu yöntemi olarak, kurduğumuz cümlelerde tek sayıda kelime olması durumunda 1-bitini, çift sayıda kelime içeren cümleler ile de 0-bitini gönderdiğimizi düşünebiliriz. Saklı iletişim yolunda, tıpkı steganografide olduğu gibi herhangi bir anahtar bulunmamakta ve mesajın gizliliği tamamen algoritmanın gizliliğine dayanmaktadır. Saklı iletişim yollarına çok çeşitli uygulama alanları bulmak olasıdır. Örneğin sayısal imzalama sırasında saklı iletişim yolu yöntemleri ile mesajı bir taraftan imzalarken diğer taraftan imzanın içine "ben bu mesajı baskı altında imzalıyorum" gibi bir ifade saklamak mümkündür.

8.17. Bit Vadetme (Bit Commitment)

Şöyle bir senaryo düşünelim; Bir yatırımcıya hisse senedi önereceğiz. Yükselecek olan hisseleri önerebilirsek, karşılığında ondan bir iş alacağız. Ancak önerdiğimiz hisse senetlerini hemen bilmesini istemiyoruz, çünkü o zaman bize işi vermeden o senetlere kendisi para yatırıp bizi yüzüstü bırakabilir. Ancak diğer taraftan, yatırımcı da bizim samimiyetimize inanmak istiyor. Eğer hisse senedi sonuçları belli olduktan sonra ona açıklarsak, sonucu baştan tahmin ettiğimize dair onu ikna etme şansımız kalmaz. Özetle; A kişisi, B kişisine, belli bir tarihte belli bir bilgiyi bildiğini aradan bir süre geçtikten sonra ona ispatlamak istiyor ve kendisi izin verene kadar bilginin ne olduğuna dair B'ye ipucu vermek istemiyordur.

B kişisi rasgele bir R metni üretir ve bunu A'ya gönderir. A'nın belli bir grup bit içerisinden seçimini gösteren bite 'b' dersek A kişisi, simetrik bir algoritma kullanarak kendi belirlediği bir anahtar ile (R,b) ikilisini şifreler ve geri B'ye gönderir. Seçtiği biti açığa çıkarmak istediği gün geldiğinde ise şifrelemede kullandığı anahtarı B'ye vererek onun paketi açmasını sağlar. B kişisi ise kendi ilk başta gönderdiği rasgele R metninin pakette seçim biti ile beraber bulunup bulunmadığını kontrol eder. Eğer protokolde böyle bir rasgele R metni olmasaydı, söz konusu olan yalnızca bir bit olabileceği için A kişisi son aşamada gönderdiği paketi rahatlıkla istediği bite (veya hatta bit grubuna)

şifresi çözülebilen bir anahtar gönderebilirdi. Ancak paketin içinde fazladan bir rasgele metnin farklı anahtarlı şifre çözme işlemleri için konulmuş olması A kişisini böylesi bir hileden alıkoymaktadır.

8.18. Sıfır-Bilgi İspat Yöntemi (Zero-Knowledge Proof)

Sıfır bilgi ispat yöntemi, kısaca belli bir bilgiye sahip olduğumuzu, o bilginin ne olduğunu açığa çıkarmadan ispatlamamızı sağlayan yöntemdir. İspatlayacağımız problemi onunla eşyapıda (isomorphic) başka bir probleme dönüştürerek problemin kendisi yerine yeni problemi ispatlıyoruz. Ancak dönüştürdüğümüz problem, öyle bir şekilde seçilir ki bu yeni problemin çözümü orijinal problemin çözümü hakkında bilgi vermiyordur. Karşı tarafın yeni çözümü orijinal problemin çözümüne dönüştürmesi, en az orijinal problemin kendisini çözmesi kadar zor olmalıdır. Ancak diğer taraftan, bizim aslında çözümüne sahip olmadığımız bir probleme eşyapıdaki bir problem için çözüm sahibiymişiz gibi davranmamız olasılığına karşı diğer taraf her seferinde bize şu soruyu soruyor. "Ya bu eşyapıdaki problemin çözümünü bana ver, ya da onun asıl probleme eş yapıda olduğunu göster". Ve eğer iddiamızda haklı isek bu iki isteği de gerçekleştirebiliriz. Ancak bu iki önermeden birini gerçekten sağlayamıyor olmamız, yani hile yapmamız durumuna yer bırakmamak içinse bu işlemler n kez (karşı taraf ikna oluncaya kadar) tekrarlanır. Öyle ki, gerçekten orijinal problemin cevabını bilmediğimiz bir durumda her defasında sorulan soruya doğru cevabı verme olasılığımız 1/2 dir. n kez tekrarlanma durumunda ise bu şekilde hile ihtimalimiz 1/n'e düşer ki yeterli tekrardan sonra ihmal edilebilecek bir olasılıktır. Sıfır-bilgi ispat'ın dayandığı nokta ise bu olasılığın düşüklüğüdür. Dolayısıyla sıfır-bilgi ispatta yeterli miktarda tekrar için mutlak manada bir ispattan çok karşı tarafın iknası söz konusudur.

Sıfır-bilgi ispat yöntemleri ne yazık ki her matematiksel probleme aynı etkinlikle uygulanamamaktadır. Kimi zaman ispat için çözümün bir kısmını açığa çıkarmak zorunda olduğumuz problemler söz konusudur. Bu yüzden

sıfır-bilgi ispat yöntemleri, karşı tarafın sorusuna ne kadar sıklıkta doğru cevap verebileceğimiz ölçüsüne göre kusursuz, istatistiksel, hesapsal ya da kullanışsız gibi kategorilere ayrılmıştır.

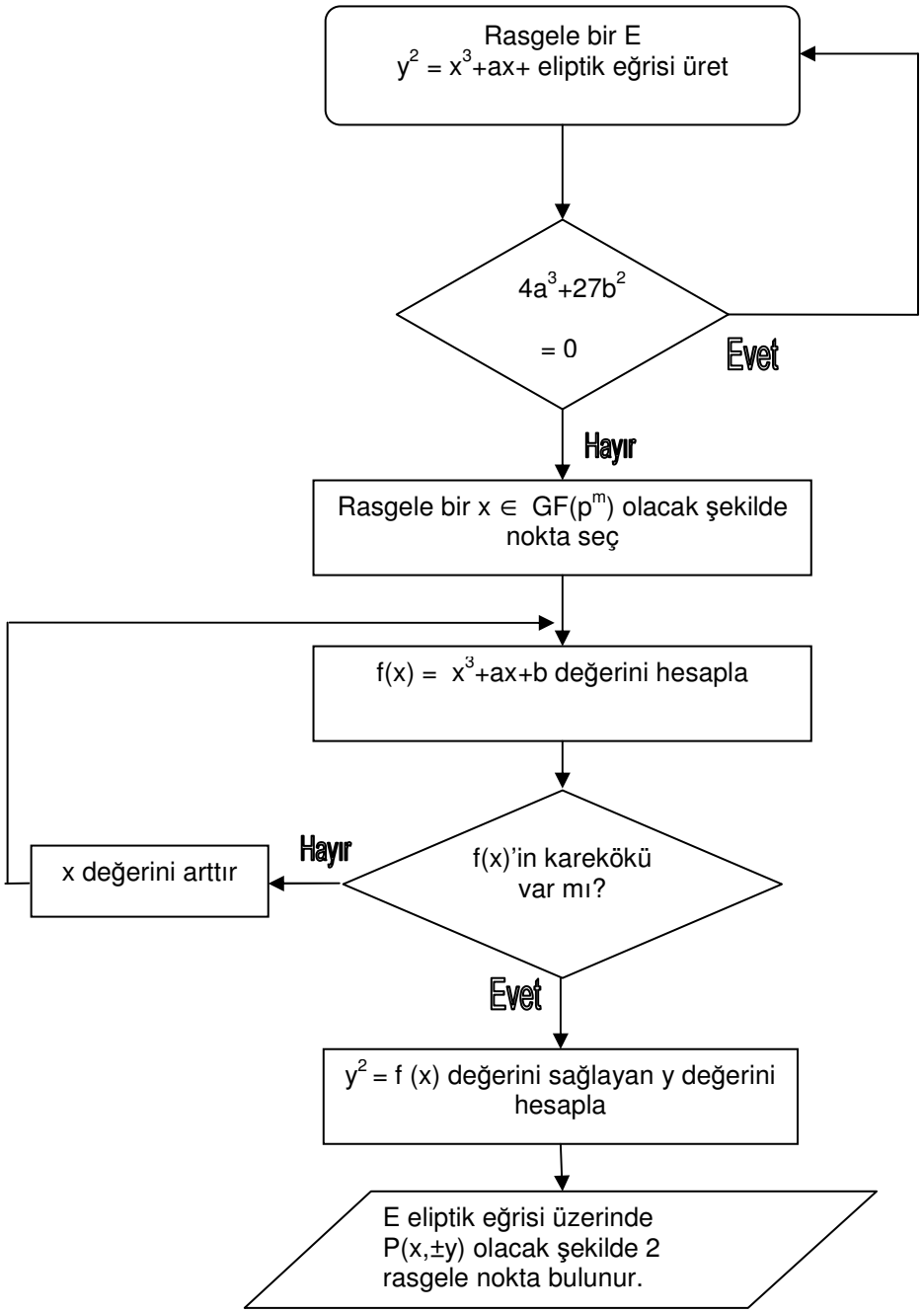
9. ELİPTİK EĞRİ KRİPTOGRAFİSİNİN C++ İLE SİMÜLASYONU

Bu bölümde, C++ programını kullanarak daha önceki bölümlerde verilen teorik bilgilerin uygulaması yapılmıştır. Eliptik eğri kriptografisi kullanılarak şifreleme ve şifre çözme programı hazırlanmıştır.

9.1. Eliptik Eğri Kriptografi Algoritmasının Açıklanması

9.1.1. Rasgele bir eliptik eğri'nin oluşturulması

Önce $p \approx 2^{180}$ olacak şekilde bir p asal sayısı ve $y^2 = x^3+ax+b$ denklemindeki eliptik eğri parametreleri olan a ve b girilecektir. Daha sonra bu değerlerin doğruluğu kontrol edildikten sonra, eliptik noktalar grubu olan $E_F(a,b)$ oluşturulacaktır. Sonrasında $E_F(a,b)$ içerisinden, başlangıç noktası olacak olan $\alpha=(x_1,y_1)$ seçilecektir. A'nin seçilmesindeki en önemli kriter, $n\alpha=0$ eşitliğini sağlayan en küçük n değerinin çok bir büyük bir asal sayı olması gerekliliğidir. Bu değer de seçildikten sonra, artık $E_F(a,b)$ ve α, kripto sistemin tüm katılımcılarına yayınlanabilir.



Şekil 9.1. E eliptik eğrisi üzerindeki noktaların bulunması

9.1.2. Eliptik eğri üzerinde yer alan bir noktaya açık metnin gömülmesi

Programda kullanacağımız yöntemin $\frac{1}{2^K}$ kadar hata verme olasılığı vardır ve *K* genelde *30≤ K ≤50* arasında değer alan bir tamsayıdır. Burada *p* asal sayı

ve $q=p^r$ 'nin büyük ve tek sayı olduğunu varsayıyoruz. F_q cismi $q>M*K$ olacak şekilde seçilir. Böylece tamsayılar 1'den $M*K'$ ya kadar $m*K+j$, $1 \leq j \leq K$, şeklinde yazılabilir ve tamsayılarla F_q'nun elemanlar kümesi arasında bir birebir örten fonksiyon oluşturabiliriz.

Verilen bir m metin birimi, her bir $j=1,2,.,K$ için, $m*K+j$ karşılık gelen bir $\alpha \in F_q$ elemanı elde ederiz. Bir kez α'yı elde ettiğimizde, $y^2 = f(\alpha)$'yı alarak bunu $y^2=x^3+ax+b$ eşitliğinin sağ tarafına gireriz. Sonrada y' yi bulabiliriz. x ve y'yi yerine koyduğumuzda bize $y^2=f(x)=x^3+ax+b$ eliptik eğrisinde $P_m = (x,y)$ noktasını verir. Eğer $y^2 = f(\alpha)$'nin p modülüne göre karekökleri yoksa, o zaman j değerini bir artırıp işlemleri tekrardan yaparak F_q alanında bir $f(\alpha)$ değeri bulunur. Bu yaklaşımda dikkat edilmesi gereken husus, j'nin değeri K'dan büyük olursa P_m noktasından m metnini elde edemiyeceğimizdir.

9.1.3. Anahtar değişimi

Bir A ve B kullanıcısı arasındaki anahtar değişimi aşağıdaki gibi gerçekleşir:

1. A, n'den küçük bir n_A tamsayısı seçer. Bu A'nın özel anahtarıdır. Daha sonra A, $P_A= n_A\alpha$ *hesabıyla* $E_F(a,b)$'nin bir noktası olan kendi açık anahtarını oluşturur.

2. B'de aynı metotla kendi açık anahtarı P_B'yi oluşturur.

3. A gizli anahtarı $K = n_AP_B$ ile, B'de gizli anahtarı $K = n_BP_A$ ile elde eder.

Üçüncü aşamadaki iki hesaplamanın sonucuda aynıdır. Çünkü,

$n_AP_B = n_A(n_B\alpha) = n_B(n_A\alpha) = n_BP_A$ eşitliği mevcuttur.

Bu yönteme bir atak gerçekleştirmek isteyen saldırgan, verilmiş *α* ve *kα* değerlerinden yola çıkara *k* değerini hesaplamak isteyecektir ve bu çok zordur.

9.1.4. Şifreleme

P_m gibi bir mesajı şifrelemek ve bir *B* kullanıcısına göndermek isteyen bir *A* kullanıcısı, rasgele pozitif bir n_A tamsayısı seçer ve C_m şifreli metninin noktalarını aşağıdaki şekilde elde eder:

$$C_m = (n_A\alpha, P_m + n_A P_B) \tag{9.1}$$

9.1.5. Şifre çözme

B kullanıcısı aşağıdaki şekilde mesajın şifresini çözer,

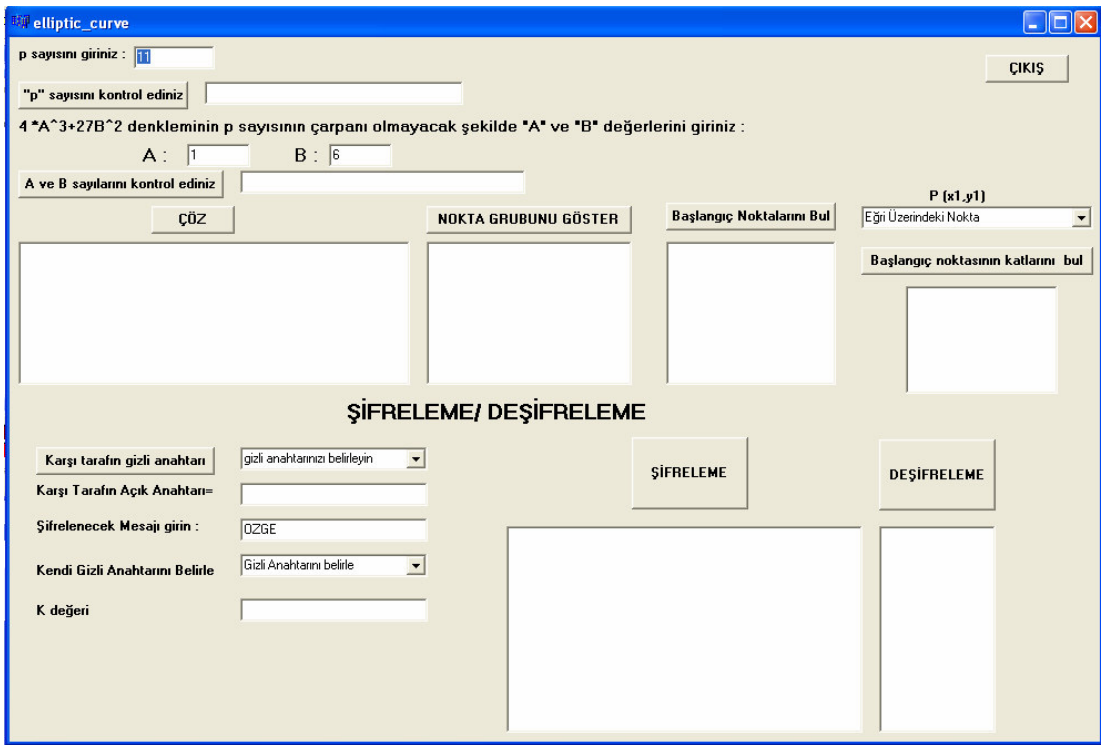
$$P_m + n_A P_B - n_B(n_A\alpha) = P_m + n_A(n_B\alpha) - n_B(n_A\alpha) = P_m \tag{9.2}$$

Bir sonraki aşama metin birimi m'in E üzerinde bulunduğu P_m tanımlama noktasından geri alınmasıdır. Bunu yapmak, P_m noktasındaki x koordinatına uyan tamsayının $\tilde{x}$ olduğu yerlerde $m = \left[\frac{\tilde{x} - 1}{\kappa} \right]$ formülünü kullanarak çok kolay bir hale dönüşmektedir.

9.2. Eliptik Eğri Kriptografisinin C++ Uygulaması

Bu bölüme kadar eliptik eğri kriptografisi algoritmasının uygulanması için gerekli tüm matematiksel ve kuramsal bilgiler verilmiştir. Bu bölümde eliptik eğri kriptografisi algoritmasının; Borland C++ 6 yardımı ile Windows işletim sistemi altında herhangi bir hazır şifreleme kütüphanesi kullanmaksızın uygulaması anlatılmıştır.

Program çalıştırıldığında aşağıdaki ekran belirir.



Resim 9.1. C++ 'daki programın ana arabirim ekran görüntüsü

Bundan sonraki alt bölümlerde, programın yapısı ve kullanılan fonksiyonlar hakkında bilgi verilmiştir.

9.2.1. ECC fonksiyonlarının açıklanması

Bu bölümde ECC şifrelemenin ihtiyaç duyduğu algoritmaların bulunduğu fonksiyonların açıklamaları yapılacaktır. Öncelikle şifreleme için gerekli anahtarların üretilmesi, daha sonra bu anahtarlar kullanılarak bir mesajın şifrelenebilmesi için kullanılan fonksiyonlar anlatılacaktır.

a ve b değerlerine bağlı olarak denklemin eliptik eğri olup olmadığının testi

$y^2 = x^3+ax+b$ denklemindeki *a* ve *b* sayıları gerçel sayılardır ve bazı basit koşulları sağlaması gerekir. x^3+ax+b ayrılabilen katsayılara sahip değilse

veya $4a^3+27b^2 \neq 0$ ise $y^2 = x^3+ax+b$ eliptik eğridir deriz. Bu testi de aşağıdaki fonksiyon ile yapabiliriz:

```
void __fastcall Telliptic_curve::Button3Click(TObject *Sender) /// a ve b Değerlerinin Kontrolü
{
        if (Button1->Tag==1)
         {
        a = StrToInt(Edit3->Text);
            b = StrToInt(Edit4->Text);
        if ((a>=p)||(b>=p))
        ShowMessage("asal sayıdan küçük bir sayı giriniz");
        else
        {
            test = ((4 * a * a * a) + (27 * b * b)) % p ;
        if (test == 0) {Button3->Tag=0; Edit5->Text="(4 * A^3 + 27 * B^2) "+IntToStr(p)+" Sayısının katıdır, Başka sayı giriniz!";}
            else{Button3->Tag=1; Edit5->Text="OK, (4 * A^3 + 27 * B^2) "+IntToStr(p)+"  Sayısının katı değildir."; }
        }  }
        else   ShowMessage("asal sayı giriniz");
```

Eliptik eğri üzerindeki noktaların hesaplanması

Rasgele seçtiğimiz E eliptik eğrisi üzerindeki noktaların hesaplanması için aşağıdaki fonksiyon kullanılabilir :

```
void __fastcall Telliptic_curve::Button4Click(TObject *Sender)
{
if(Button3->Tag==1)
{
ListBox1->Clear();
ListBox1->Items->Add("x          x^3+ "+ Edit3->Text+"* x + "+Edit4->Text+ " mod ("+IntToStr(p)+")       in QR("+IntToStr(p)+")?     y");
m=0;
for (int i = 0; i < p; i++)
      {
      k=0;
      long z = ((long) pow(i, 3) + (a * i) + b) % p;
      if(z!=0)  {
        long eulerTest = ((long)pow(z, ((p - 1,0)/ 2,0)));

        if(eulerTest==1) {
```

```
                for (int j = 0; j < p; j++)  {
                    long quad = (j * j) % p;
                    if (quad==z){
                        R[m]= j;
                        P[m]=i;
                        k=k+1;
                        ListBox1->Items->Add(IntToStr(i) + " " + IntToStr(z) + "
evet              "+ IntToStr(R[m]));
                        m=m+1; }          }
            }  //if(eu==1)
              } // if(z!=0)

      }  // for (int i = 0; i < p; i++)
       }  // if(Button3->Tag==1)
       else ShowMessage("Asal sayının katı başka sayı giriniz");
}
```

Elde ettiğimiz eliptik noktalar grubu $E_F(a,b)$'yi oluşturmaktadır.

Başlangıç noktasının hesaplanması

$E_F(a,b)$ içerisinden, başlangıç noktası (generator point) olacak olan $\alpha=(x_1,y_1)$ seçilsin. A'nin seçilmesindeki en önemli kriter, $n\alpha=0$ eşitliğini sağlayan en küçük n değerinin çok bir büyük bir asal sayı olması gerekliliğidir. Bunu sağlayan fonksiyon da aşağıda verilmiştir.

```
void __fastcall Telliptic_curve::Button12Click(TObject *Sender) //
BAŞLANGIÇ NOKTALARINI BUL
{
ListBox5->Clear();
for (int u=0;u<ListBox2->Items->Count;u++)
{ c=1;
y= u;
RR[t]= R[y];
PP[t]= P[y];
A[0]=  P[y];
B[0]=  R[y];
   while (((R[y]!=(p-RR[t]))||((PP[t]==0)&&(RR[t]==0)))&&(c<(2*p+1)))
```

```
 {  if((PP[t]!=P[y])||(RR[t]!=R[y]))
            {ust = (RR[t]-R[y])%p;
            alt = (PP[t]-P[y])%p;
            if(ust!=0)
            {do
              {  ust=ust+p;
              } while(!((ust%alt)==0));
            h=((long)ust/alt)%p;    }
            else h=0;       }
     else { ust=(3*(long)pow(PP[t],2)+a)%p;
            alt=(2*RR[t])%p;
            if(ust!=0)
            {do
              { ust=ust+p;
              } while (!((ust%alt)==0));
            h=((long)ust/alt)%p; }
            else h=0; }
     PP[t]= ((long)pow(h,2)-P[y]-PP[t])%p;
     while (PP[t]<0)    { PP[t]=p+PP[t];  };
     RR[t]=(h*(P[y]-PP[t])-R[y])%p;
     while (RR[t]<0)   {  RR[t]=p+RR[t]; };
     c++;
     if(c>2*p){ShowMessage("bu sayı sonsuz sayı değil, başka sayı seçin");}
      A[c-1]= PP[t];
      B[c-1]= RR[t];       };
      baslangic[u]=c;    }
int y=1;
if ((p == 0)||(p ==1)) { Edit2->Text="p bir asal sayı değildir"; }
else {for (int i=2;i<sqrt(p);i++)
            {  if((p%i)==0){
                        y=0;
                        break; }}
```

```
        if (y==0){Edit2->Text="p bir asal sayı değildir. Yeni bir sayı giriniz.";
Button1->Tag=0;}
        else {Edit2->Text="p asal sayıdır "; Button1->Tag=1;}   }
      enb=baslangic[0];
     for (int u=1;u<ListBox2->Items->Count;u++)
         { if(enb<baslangic[u])  {
          enb=baslangic[u];  }  // for (int u=0;u<ListBox2->Items->Count;u++)
     for (int u=0;u<ListBox2->Items->Count;u++)
         {   if(enb==baslangic[u])
        ListBox5->Items->Add("( "+IntToStr(P[u])+" , "+IntToStr(R[u])+" )");
          } // for (int u=0;u<ListBox2->Items->Count;u++)    }
```

Şifreleme

Başlangıç olarak, programa şifreleme yapılacak açık metin girilir. Daha sonra $q>M*K$ olacak şekilde K değeri seçilir. Genellikle K değeri $30 \leq K \leq 50$ arasında değer alan bir tamsayıdır. Girilen açık metin birimi her bir $j=1,2,..,K$ için, $m*K+j$ karşılık gelen bir $\alpha \in F_q$ elemanı elde ederiz. Bir kez α'yı elde ettiğimizde, $y^2 = f(\alpha)$'yı alarak bunu $y^2=x^3+ax+b$ eşitliğinin sağ tarafına gireriz. Sonrada y'yi bulabiliriz. x ve y'yi yerine koyduğumuzda bize $y^2=f(x)=x^3+ax+b$ eliptik eğrisinde $P_m = (x,y)$ noktasını verir. Bu da aşağıdaki fonksiyon ile yapılır :

```
mesaj = Edit9->Text;
for (int i=0;i<Edit9->GetTextLen();i++)
{
Edit9->SelStart = i;
Edit9->SelLength =1;

if ((Edit9->SelText=="A")||(Edit9->SelText=="a")) aaa[i]=10;
else if ((Edit9->SelText=="B")||(Edit9->SelText=="b")) aaa[i]=11;
else if ((Edit9->SelText=="C")||(Edit9->SelText=="c")) aaa[i]=12;
else if ((Edit9->SelText=="D")||(Edit9->SelText=="d")) aaa[i]=13;
else if ((Edit9->SelText=="E")||(Edit9->SelText=="e")) aaa[i]=14;
else if ((Edit9->SelText=="F")||(Edit9->SelText=="f")) aaa[i]=15;
else if ((Edit9->SelText=="G")||(Edit9->SelText=="g")) aaa[i]=16;
else if ((Edit9->SelText=="H")||(Edit9->SelText=="h")) aaa[i]=17;
```

```
else if ((Edit9->SelText=="I")||(Edit9->SelText=="i")) aaa[i]=18;
else if ((Edit9->SelText=="J")||(Edit9->SelText=="j")) aaa[i]=19;
else if ((Edit9->SelText=="K")||(Edit9->SelText=="k")) aaa[i]=20;
else if ((Edit9->SelText=="L")||(Edit9->SelText=="l")) aaa[i]=21;
else if ((Edit9->SelText=="M")||(Edit9->SelText=="m")) aaa[i]=22;
else if ((Edit9->SelText=="N")||(Edit9->SelText=="n")) aaa[i]=23;
else if ((Edit9->SelText=="O")||(Edit9->SelText=="o")) aaa[i]=24;
else if ((Edit9->SelText=="P")||(Edit9->SelText=="p")) aaa[i]=25;
else if ((Edit9->SelText=="Q")||(Edit9->SelText=="q")) aaa[i]=26;
else if ((Edit9->SelText=="R")||(Edit9->SelText=="r")) aaa[i]=27;
else if ((Edit9->SelText=="S")||(Edit9->SelText=="s")) aaa[i]=28;
else if ((Edit9->SelText=="T")||(Edit9->SelText=="t")) aaa[i]=29;
else if ((Edit9->SelText=="U")||(Edit9->SelText=="u")) aaa[i]=30;
else if ((Edit9->SelText=="V")||(Edit9->SelText=="v")) aaa[i]=31;
else if ((Edit9->SelText=="W")||(Edit9->SelText=="w")) aaa[i]=32;
else if ((Edit9->SelText=="X")||(Edit9->SelText=="x")) aaa[i]=33;
else if ((Edit9->SelText=="Y")||(Edit9->SelText=="y")) aaa[i]=34;
else if ((Edit9->SelText=="Z")||(Edit9->SelText=="z")) aaa[i]=35;
else if (StrToInt(Edit9->SelText)== 0) aaa[i]=0;
else if (StrToInt(Edit9->SelText)== 1) aaa[i]=1;
else if (StrToInt(Edit9->SelText)== 2) aaa[i]=2;
else if (StrToInt(Edit9->SelText)== 3) aaa[i]=3;
else if (StrToInt(Edit9->SelText)== 4) aaa[i]=4;
else if (StrToInt(Edit9->SelText)== 5) aaa[i]=5;
else if (StrToInt(Edit9->SelText)== 6) aaa[i]=6;
else if (StrToInt(Edit9->SelText)== 7) aaa[i]=7;
else if (StrToInt(Edit9->SelText)== 8) aaa[i]=8;
else if (StrToInt(Edit9->SelText)== 9) aaa[i]=9;
}
int w=1;
 for (int i = 0; i < Edit9->GetTextLen(); i++)
      {   int xx=1;
      w=1;
      do
      { k=0;
      xkoor[i]= aaa[i]*K+w;
      xkoor[i]=xkoor[i] %p;
      long z = ((long) pow(xkoor[i], 3) + (a * xkoor[i]) + b) % p;
                for (int j = 0; j < p; j++)  {
                    long quad = (j * j) % p;
                    if (quad==z){
                        MESAJ[i]= j;
                        k=k+1;
                        ListBox4->Items->Add("i=   "+   IntToStr(i)+"       mesaj   =
"+IntToStr(aaa[i])+ "    x = "+IntToStr(xkoor[i])+ " y2= " + IntToStr(z) + " y= "+
IntToStr(MESAJ[i]));
                        xx=0;
```

```
            break;  }}
        if(xx!=0)  {w=w+1;    }
           }while (xx!=0);  }  // for (int i = 0; i < p; i++)
```

Yukarıdaki fonksiyonu kullanarak elde ettiğimiz değer, girdiğimiz mesajın gömülü olduğu eliptik eğri grubunda yer alan nokta değeridir. Daha sonra bu noktanın anahtar değişimi yapılarak *A* kişisi tarafından şifrelenmesi sağlanır. Anahtar değişiminde *B* kullanıcısının genel anahtarı P_B ile *A* kullanıcısının özel anahtarı n_A çarpılır. Daha sonra *A* kullanıcısı, rasgele pozitif bir n_A tamsayısı seçer ve C_m şifreli metninin noktalarını $C_m = (n_A\alpha, P_m + n_A P_B)$ elde eder. Burada kullanılan fonksiyon eliptik eğri üzerindeki noktaların toplanması ile ilgili aşağıda yer alan temel denklemleri kullanarak kolaylıkla yapılır.

- $P= (x_1,y_1)$ ve Q$= (x_2,y_2)$ ise ve $P \neq -Q$ ise, bu durumda, $P+Q =(x_3,y_3)$

$x_3 \equiv \lambda^2 - x_1 - x_2 \ (mod\ p)$

$y_3 \equiv \lambda\ (x_1 - x_3) - y_1 \ (mod\ p)$

λ için koşul,

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & eğer \quad P \neq Q \quad ise \\ \dfrac{3x_1^2 + a}{2y_1} & eğer \quad P = Q \ \ ise \end{cases}$$

Şifre çözme

B kullanıcısı, A kullanıcısının gönderdiği şifreli metnin şifresini kendi özel anahtarını kullanarak çözer. Bunu da;

$P_m + n_A P_B - n_B(n_A\alpha) = P_m + n_A(n_B\alpha) - n_B(n_A\alpha) = P_m$

ile sağlar. Daha sonra aşağıdaki fonksiyonu kullanarak açık metni elde eder.

void __fastcall Telliptic_curve::Button11Click(TObject *Sender) // DEŞİFRELEME

```
{
for(int ii=0;ii<Edit9->GetTextLen();ii++)
{int ll=0;
do{
 ll=ll+1;
 mesa[ii] = (dsifre1[ii]-ll)/K;
  }while(((dsifre1[ii]-ll)%K)!=0);
 if (mesa[ii]== 10) ListBox6->Items->Add(" A " );
else if (mesa[ii]== 11) ListBox6->Items->Add(" B" );
else if (mesa[ii]== 12) ListBox6->Items->Add(" C " );
else if (mesa[ii]== 13) ListBox6->Items->Add(" D " );
else if (mesa[ii]== 14) ListBox6->Items->Add(" E " );
else if (mesa[ii]== 15) ListBox6->Items->Add(" F " );
else if (mesa[ii]== 16) ListBox6->Items->Add(" G " );
else if (mesa[ii]== 17) ListBox6->Items->Add(" H " );
else if (mesa[ii]== 18) ListBox6->Items->Add(" I " );
else if (mesa[ii]== 19) ListBox6->Items->Add(" J " );
else if (mesa[ii]== 20) ListBox6->Items->Add(" K " );
else if (mesa[ii]== 21) ListBox6->Items->Add(" L " );
else if (mesa[ii]== 22) ListBox6->Items->Add(" M " );
else if (mesa[ii]== 23) ListBox6->Items->Add(" N " );
else if (mesa[ii]== 24) ListBox6->Items->Add(" O " );
else if (mesa[ii]== 25) ListBox6->Items->Add(" P " );
else if (mesa[ii]== 26) ListBox6->Items->Add(" Q " );
else if (mesa[ii]== 27) ListBox6->Items->Add(" R " );
else if (mesa[ii]== 28) ListBox6->Items->Add(" S " );
else if (mesa[ii]== 29) ListBox6->Items->Add(" T " );
else if (mesa[ii]== 30) ListBox6->Items->Add(" U " );
else if (mesa[ii]== 31) ListBox6->Items->Add(" V " );
else if (mesa[ii]== 32) ListBox6->Items->Add(" W " );
else if (mesa[ii]== 33) ListBox6->Items->Add(" X " );
else if (mesa[ii]== 34) ListBox6->Items->Add(" Y " );
else if (mesa[ii]== 35) ListBox6->Items->Add(" Z " );
```

```
else if (mesa[ii]== 0) ListBox6->Items->Add(" 0 " );
else if (mesa[ii]== 1 ) ListBox6->Items->Add(" 1 " );
else if (mesa[ii]== 2) ListBox6->Items->Add(" 2 " );
else if (mesa[ii]== 3) ListBox6->Items->Add(" 3 " );
else if (mesa[ii]== 4) ListBox6->Items->Add(" 4 " );
else if (mesa[ii]== 5) ListBox6->Items->Add(" 5 " );
else if (mesa[ii]== 6) ListBox6->Items->Add(" 6 " );
else if (mesa[ii]== 7) ListBox6->Items->Add(" 7 " );
else if (mesa[ii]== 8) ListBox6->Items->Add(" 8 " );
else if (mesa[ii]== 9) ListBox6->Items->Add(" 9 " );}
}
```

10. SONUÇ

Günümüzde, bilim ve teknolojinin hızla gelişmesi sonucunda, bilgilerin korunması konusunda büyük sorunlar yaşanmaktadır. Buna bağlı olarak, simetrik ve asimetrik kripto sistemleri hızla yayılmaktadır. Asimetrik kripto sistemlerinin en güvenilir ve son zamanlardaki en popüler sistemi Eliptik Eğri Kripto Sistemidir. Ancak, eliptik eğri kripto sistemlerine dayalı uygulamalarda sistemin matematiğine dikkat edilmesi gerekmektedir. Bu sistemlerin uygulamaları hem büyük sayılarda aritmetik, hem de sonlu cisimlerin ve eliptik eğri noktalarının aritmetiğini gerektirmektedir.

Eliptik Eğrilere dayalı açık anahtarlı kripto sistemleri daha küçük anahtar uzunluğu ile, RSA ve DSA gibi diğer açık anahtar kripto sistemlerinin sahip olduğu güvenliği sağlamaktadır. Daha küçük anahtar boyu bu sistemlere büyük bir avantaj sağlamaktadır. Özellikle sayısal imza gibi akıllı kart kullanılan uygulamalarda eliptik eğri sistemlerinin gelecekte daha fazla yaygınlaşması beklenmektedir. Sayısal imza uygulamalarında en çok kullanılan yöntemlerden biri de özet fonksiyonuyla imzalamaktır. Bu yöntem çoğunlukla bankalarda para aktarımındaki güvenliği sağlamak ve verilen talimatın inkâr edilememesini sağlamak amacıyla kullanılır.

ECC, RSA şifreleme algoritmasının yerini alabilecek çok önemli avantajlara sahip bir açık anahtarlı şifreleme algoritmasıdır. Bununla ilgili çalışmalar hem Türkiye'de hem de bütün dünya da hızla devam etmektedir. İçinde bulunduğumuz yüzyılda en çok kullanılan ve güvenilir şifreleme yöntemi olarak hak ettiği yeri alacaktır.

KAYNAKLAR

1. Koblitz, N., "A Course in Number Theory and Cryptography 2nd ed.", ***Springer-Verlag,*** New York, 167-199, (1987).

2. Jones, A.G., Jones, J.M., "Quadratic Residues", Elementary Number Theory, 3rd ed., ***Spinger,*** London, 119-141 (1999).

3. Lenstra, H.W, Jr., "Factoring integers with elliptic curves", ***Annuals of Mathematics,*** 126: 649-673 (1987).

4. Akben, S.B., Subaşı, A., "RSA ve Eliptik Eğri Algoritmasının Performans Karşılaştırlması", ***KSÜ Fen ve Mühendislik Dergisi,*** 8(1): 35-38 (2005).

5. Stinson, D.R., "Cryptography Theory and Practice 2nd ed.", ***CRC Press***, Amerika, 183-190 (1995).

6. Hankerson, D., Menezes, A., Vanstone, S., "Guide to Elliptic Curve Cryptography", ***Spinger***, New York, 75-152 (2003).

7. Menezes, A., "Elliptic Curve Public Key Cryptosystems", **Kluwer Academic Publishers**, 65-78, (1993).

8. Koblitz, N., "Elliptic curve cryptosystems", ***Mathematics of Computation,*** 48: 203–209 (1987).

9. Menezes, A. J., Oorschot, P. C., Vanstone, S. A., "Handbook of Applied Cryptography", ***CRC Press***, Amerika, 425-534, (1996).

10. Crutchley, A.D., "Cryptography and Elliptic Curves", Yüksek Lisans Tezi, ***University of Southampton Faculty of Mathematical Studies,*** Southampton, 55-56 (1999).

11. Blake, I.F., Seroussi, G., Smart, N.P., "Advances in Elliptic Curve Cryptography", ***Cambridge University Press***, İngiltere, 3-19, (2005).

12. İnternet: Handbook of Applied Cryptography http://www.cacr.math.uwaterloo.ca/hac/ (2006).

13. Cenk, M., Özbudak, F., "Eliptik Eğri Kriptografi ve Aritmetiği", ***1. Ulusal Kriptoloji Sempozyumu,*** ODTÜ Ankara, 134-141 (2005).

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: ÇETİN, Özge
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 07.07.1979 Diyarbakır
Medeni hali	: Bekar
Telefon	: 0 (312) 402 42 74
Faks	: 0 (312) 402 44 04
e-mail	: cetinozge@hotmail.com.

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Lisans	Gazi Üniversitesi	
	Elkt.ve Elktr. Müh. Bölümü	2002
Lise	Ziya Gökalp Lisesi	1997

İş Deneyimi

Yıl	Yer	Görev
2003-2006	Milli Savunma Bakanlığı	Elktr.Sis.Prj.ve Kşf.Müh.

Yabancı Dil

İngilizce, Almanca

Hobiler

SCUBA, Yamaç Paraşütü, Dağcılık, Voleybol, Yüzme