T.C.

DİCLE ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

KRİPTOLOJİDE ELİPTİK EĞRİ ALGORİTMASI

Aziz Mahmut YÜCELEN

YÜKSEK LİSANS TEZİ

MATEMATİK ANABİLİM DALI

DİYARBAKIR

Haziran - 2011

T.C

DİCLE ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜ

DİYARBAKIR

Aziz Mahmut YÜCELEN tarafından yapılan bu çalışma, jürimiz tarafından Matematik Anabilim Dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir.

Jüri Üyesinin

Ünvanı	Adı Soyadı
Başkan	: Prof.Dr.Ali YILMAZ
Üye	: Prof.Dr.H.İlhan TUTALAR
Üye	: Yrd. Doç. Dr.Abdullah BAYKAL

Yukarıdaki bilgilerin doğruluğunu onaylarım.

/ /2011

Prof. Dr. Hamdi TEMEL

ENSTİTÜ MÜDÜRÜ

(MÜHÜR)

TEŞEKKÜR

Tez çalışmam süresince büyük yardımlarını gördüğüm, bilgi ve deneyiminden yararlandığım değerli hocam sayın Yrd. Doç. Dr. Abdullah BAYKAL'a, destek ve ilgisinden dolayı sayın Prof. Dr. Hasan İlhan TUTALAR'a manevi desteklerinden dolayı eşim Esra YÜCELEN'e ve sevimli kızım Berra YÜCELEN'e, teşekkürlerimi sunmayı bir borç bilirim.

İÇİNDEKİLER

ÖZET

KRİPTOLOJİDE ELİPTİK EĞRİ ALGORİTMASI

YÜKSEK LİSANS TEZİ

Aziz Mahmut YÜCELEN

DİCLE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MATEMATİK ANABİLİM DALI

2011

Bu tezde eliptik eğri şifrelemenin matematiksel temelleri ve tanımları yapılmış olup, El-Gamal eliptik eğri şifreleme uygulaması geliştirilmiştir. Eliptik eğri şifreleme algoritması RSA şifreleme algoritması ile karşılaştırılmış ve açık anahtarlı şifrelemede, algoritmayı oluşturan temel matematiksel yapının, özel anahtar uzunluğundan daha önemli olduğu ve eliptik eğri şifreleme algoritmasının RSA algoritmasından daha yüksek güvenlik sağladığı görülmüştür. Temel java bilgileri ile El-Gamal eliptik eğiri şifreleme örnek programı yazılmıştır.

Anahtar Kelimeler : Eliptik Eğri Algoritması, Kriptoloji

ABSTRACT

ELLIPTIC CURVE ALGORITHM OF CRYPTOLOGY

MSc THESIS

Aziz Mahmut YÜCELEN

DEPARTMENT OF MATHMATİCS
INSTITUTE OF NATURAL AND APPLIED SCIENCES
UNIVERSITY OF DICLE

2011

In this thesis, the mathematical basis and definition of elliptic curves are studied and the application of El-Gamal elliptic curve crypto system is realized. The algorithm of the elliptic curve cryptography is compared with the powerful algorithms of public key crypto systems such as RSA and in the public key cryptography, the essential mathematical function behind the algorithms are found to be more important than the length of the private key and elliptic curve algorithm is secure than RSA . Example desktop application of the elliptic curve crypto system is written in basic java programming.

Key Words: Elliptic Curve Algorithm, Cryptology

ÇİZELGE LİSTESİ

ŞEKİL LİSTESİ

KISALTMA VE SİMGELER

EE	: Eliptik eğri
EEŞ	: Eliptik eğri şifreleme
DHADA	: Diffie-Helmann anahtar değişimi algoritması
EEESİA	: Eliptik eğri El-Gamal sayısal imzalama algoritması

1. GİRİŞ

Kriptoloji, geçmişten beri insanların her türlü iletişiminde gizlilik, reddedilemezlik ve doğruluk ihtiyacını karşılamak üzere düşünülmüş ve uygulamaya geçilmiştir.

Günümüze kadar birçok kriptoloji sistemi geliştirilmiş fakat geliştirilen sistemler o günün koşullarına ayak uydurup, değişen dünyanın gelişen teknolojisine ayak uydurmak için modern bilimlerde olduğu gibi matematik biliminin de bir alt araştırma konusu olarak gücellenip insanların ilgisini çekmeye devam etmiştir. Teknoloji ilerledikçe şifreleme, önemini daha belirgin bir şekilde hissettirmekte ve birçok ülkede araştırmacılar ve özel şirketler bu konu üzerine yoğunlaşıp değişik ve kırılması zor algoritmala üretmek için ciddi kaynaklar harcamaktadırlar. Şifrelemenin temelini oluşturan gizliliğin sağlanması ilkesi günümüzde bilgisayarlar veya elektronik sistemlerde uygulanarak hayatımızı kolaylaştırmaya ve ilgili işlemleri güven içinde yapmamızı sağlamaktadır.

İster iş amaçlı isterse sosyal olarak kullanılan internet, bilgi paylaşımının yoğun oldugu bir yapı olması, şifrelemenin önemini açık bir şekilde ortaya koymaktadır. Günümüzde şifrelemeler bankacılık, e-devlet uygulamaları, uzaktan eğitim sistemleri, güvenlik güçleri iletişim sistemleri, uydu , kara ve deniz harp sistemlerinde, kimlik doğrulama ve daha bir çok cisimda aktif bir şekilde kullanılmaktadır.

Kriptoloji biliminin temel mantığı bilinenin aksine kırılamazlık değil kırılmanın güçlüğü üzerine kuruludur. Bu anlamda geliştirilen algoritmalar bu kırılma zorluğunu elde edebilmek için tek yönlü matematiksel foksiyonlar üretmekte iken diğer taraftan matematiksel analizler ile kırılma yolları ve kırılma algoritmaları üzerine de çalışmalar ve yaklaşımlar geliştirilmektedir. Yakın zamana kadar şifreleme için kullanılan açık anahtar sistemine sahip RSA, DES, 3DES gibi bir çok algoritma, aynı anahtar uzunluğuna sahip eliptik eğri şifrelemenin sağladığı güvenliğin yarısını sağlayabildiği gerçeği, araştırmacıları eliptik eğriler cismina itmiştir fakat bu avantaj diğer konularda dezavantaj olarak kendini gösterebilmektedir. Bu nedenle eliptik eğri şifrelemeninde diğer şifreleme tekniklerine göre işlemci gücü ve buna bağlı sorunların varlığı, araştırmacıları bu dezavantajları ortadan kaldırmaya yönelik çalışmaya sevketmektedir.

Bu tez, eliptik eğri şifreleme algoritmalarının araştırılması üzerine bir çalışmadır. Algoritmaların oluşturulması, oluşturulurken kullanılan matematiksel tanımlamalar ve teoremler konusunda bilgi verilmesi amaçlanmıştır.

2. KAYNAK ÖZETLERİ

Diffie ve Hellman`ın açık anahtarlı kriptografi olarak tanımladıkları 1976 yılında yayımlanmış "New Directions in Cryptography" isimli makalelerinde yer aldı ve bu algoritma kriptografik sisteme örnek olarak Diffie-Hellman Anahtar değişimi idi. Birçok ticari uygulama bu anahtar değişimini kullandı. Algoritmanın amacı, iki kullanıcının bir anahtarı güvenli şekilde birbirlerine iletmeleri ve daha sonrasında da bu anahtar yardımı ile şifreli mesajları birbirlerine gönderebilmelerini sağlamaktı. Algoritma anahtar değişimi ile sınırlıdır. Diffie-Hellman ortak gizli anahtar oluşturma sistemi ayrık logaritma problemini üzerine kurulmuş ve güvenirliği çok büyük asal sayıları seçmeye dayanmaktadır.

Taher Elgamal 1985 te "Açık anahtrar kriptosistemi ve Ayrık Logaritma Problemine Dayanan İmza Şeması" isimli yayınını sundu, daha sonra Elgamal imza şeması, NIST tarafından Dijital İmza Standardı olarak önerilen Dijital İmza Algoritmasının temelini oluşturdu.

Hankerson D. Kriptoloji ve Diferansiyel Denklemler Cisminda çalışmaktadır. Kriptoloji cisminda 'Eliptik Eğri Kriptoloji Klavuzu' isimli kitabı mevcuttur.

Washington L.C. Sayılar Teorisi, Cyclotomic Cisimlar, Eliptik Eğriler ve Kriptoloji cisminda çalışmaktadır. Kriptoloji cisminda 'Eliptik Eğri Sayılar Teorisi ve Kriptolojisi' isimli kitabı mevcuttur.

Andreas Enge, 'Eliptik Eğriler ve Kriptolojideki Uygulamaları', 'Hipereliptik Kriptosistemler' isimli, kriptoloji cisminda kitapları mevcuttur.

Lindell Yehuda, Kriptoloji ve Güvenli Protokoller üzerine çalışmaktadır. Kriptoloji cisminda 'Modern Kriptolojiye Giriş' isimli bir kitabı bulunmaktadır.

Francisco Rodríguez-Henríquez, kriptoloji cisminda 'Yeniden Ayarlanabilen Donanımlarda Kriptografik Algoritmalar' isimli kitabı mevcuttur.

Babinkostova Liljana, Kümeler Teorisi, Topoloji ve Kriptoloji cisminda çalışmaktadır.

3. MATERYAL VE METOT

3.1. Matematiksel Temeller

3.1.1. Grup

Tanım 3.1.: G boş olmayan bir küme ve "Δ"da G üzerinde tanımlı bir ikili işlem olsun. Aşağıdaki işlemlerin sağlanması durumunda $<G, \Delta>$ sistemine bir grup denir.

(i) $\forall a, b \in G$ için $a \Delta b \in G$ dir. (Kapalılık özelliği)

(ii) $\forall a, b, c \in G$ için $(a \Delta b) \Delta c = a \Delta (b \Delta c) \in G$ dir. (Birleşme özelliği)

(iii) $\forall a \in G$ için $a \Delta e = e \Delta a = a$ olacak şekilde bir $e \in G$ vardır. (Birim eleman özelliği)

(iv) $\forall a \in G$ için $a \Delta b = b \Delta a = e$ olacak şekilde bir $b \in G$ vardır. (Ters eleman özelliği)

Tüm bunlara ek olarak ikili işlem değişme özelliğini sağladığı taktirde $< G, \Delta >$ grubuna değişmeli grup denir.

(v) $\forall a, b \in G$ için $a \Delta b = b \Delta a$ (Değişme özelliği)

Örnek olarak tamsayılar kümesi "+" işlemi için <Z,+> bir (değişmeli gruptur. Benzer olarak "0" dan "n-1" 'e kadar olan tamsayıların oluşturduğu küme, modülo n toplama işlemi , $< Z_n$,+> bir değişmeli gruptur.

3.1. 2. Halka

Tanım 3.2. : $<R, \Delta, \Diamond>$ birimli halkası, bir R kümesi ve bu kümenin elemanları arasında tanımlanmış olan 'Δ' ve '$\Diamond$' gibi iki adet ikili işlemden oluşmaktadır.

R kümesinin ve üzerinde tanımlanmış işlemlerin bir halka olabilmesi aşağıdaki özellikleri sağlaması durumunda gerçekleşir.

(i) $<R, \Delta>$ bir değişmeli grup olmalıdır. Etkisiz öğesine halkanın sıfırı denilmektedir ve genellikle '0' ile gösterilmektedir.

(ii) ‘ ◊ ’ işlemi R üzerinde, kapalılık ve birleşme özelliğini sağlamalıdır. ‘ ◊ ’ işlemi için birim eleman tanımlanabilmelidir. Birim genellikle ‘1’ ile gösterilir ve aksi gösterilmedikçe $0 \neq 1$ varsayılmaktadır.

(iii) $\forall$ a,b,c $\in$ R için (a Δ b) ◊ c=(a ◊ c) Δ (b ◊ c) olmalıdır. Buna ◊ işleminin Δ işlemi üzerine sağdan dağılma kuralı diyoruz. Sağdan dağılma kuralı da sağlanmalıdır. Her ikisi birden kısa dağılma kuralı veya dağılma özelliği olarak ifade edilecektir.

(iv) Ek olarak ‘ ◊ ’ işleminin değişme özelliği varsa <R, Δ , ◊ > halkası ‘Değişmeli Halka’ adını alır. $R = \{0\}$ özel durumunda, Halkaya ‘Sıfır Halka’ adı verilir. Bu tezde halka denince $R \neq \{0\}$ anlaşılacaktır.

3.1. 3. Cisim

Tanım 3.3. : R sayı kümesi üzerinde tanımlanmış olan ‘ Δ ’ ve ‘ ◊ ’ işlemeriyle birlikte aşağıdaki aksiyomları sağlıyorsa bir ‘Cisim’ oluşturur.

(i) <R, Δ> bir değişmeli grup olmalıdır.

(ii) <R, ◊ > bir değişmeli grup olmalıdır. Sadece ‘ Δ ’ işleminin sıfır elemanı için bir ters eleman bulunmaz.

(iii) <R, Δ , ◊ > bir halka olmalıdır. Yani yukarıdaki her iki koşula ek olarak ‘ Δ ’ işleminin ‘ ◊ ’ işlemi üzerine dağılma özelliği olmalıdır.

Bu tanıma bir örnek gösterecek olursak gerçel sayılar kümesi toplama ve çarpma işlemleri ile birlikte bir ‘Cisim’ oluşturmaktadır.

3.1.4. Sonlu Cisim

Tanım 3.4. : Yukarıda tanımı verilen cisimler sonlu sayıda eleman içeriyorsa bu cisime ‘Sonlu cisim’ denir.

Sonlu cismin eleman sayısı o cismin derecesidir. Aynı sonlu sayıda elemana sahip, derecesi aynı cisimler eş yapıdadırlar sadece elemanların gösterilişi farklıdır.

Bir sonlu elemanın var olabilmesi için o sonlu cismin derecesinin p ve m asal tamsayı olmak üzere $q = p^m$ şeklinde bir asalın kuvveti olması gerekir ve gösterimi de F_q şeklindedir (SEC 2000).

3.2. Eliptik Eğrilere Giriş

Tanım 3.5. : Bir K cismi üzerinde tanımlanmış bir E eliptik eğri (EE) denklemi ve $a_1, a_2, a_3, a_4, a_5 \in K$ olmak üzere

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_5 \tag{1}$$

şeklinde gösterilir. E eliptik eğrisinin diskriminantı Δ ile gösterilir ve d_2, d_4, d_6, d_8 katsayıları

$$\begin{aligned} d_2 &= a_1^2 + 4a_2 \\ d_4 &= 2a_4 + a_1a_3 \\ d_6 &= a_3^2 + 4a_5 \\ d_8 &= a_1^2a_5 + 4a_2a_5 - a_1a_3a_4 + a_2a_3^2 - a_4^2 \end{aligned} \tag{2}$$

olarak alınırsa Δ diskriminantı

$$\Delta = -d_2^2d_8 - 8d_4^3 - 27d_5^2 + 9d_2d_4d_5 \tag{3}$$

şeklinde tanımlanır. Dikkat edilirse $\Delta \neq 0$ olmaktadır.

Eğer L, K'nın genişletilmiş bir cismi ise E üzerindeki *L-rasyonel noktaların* kümesi, '∞'sonsuz(daki) noktasını göstermek üzere

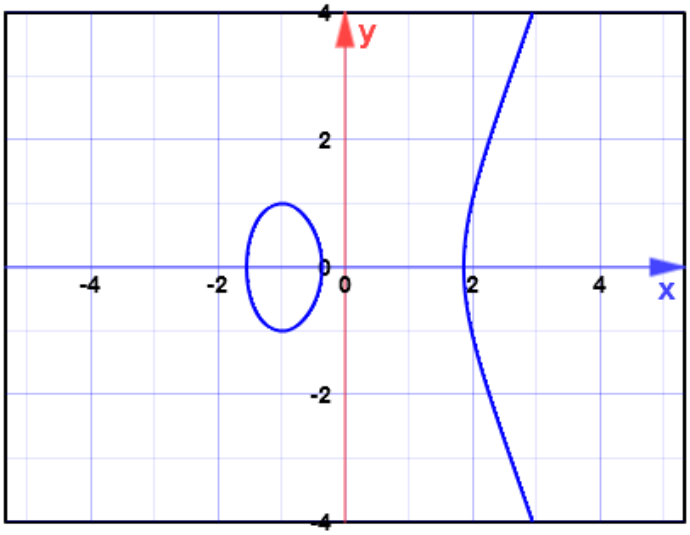
$$E(L) = \{(x, y) \in L \times L : y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6 = 0\} \cup \{\infty\} \tag{4}$$

şeklinde tanımlanır (Hankerson ve ark. 2003).

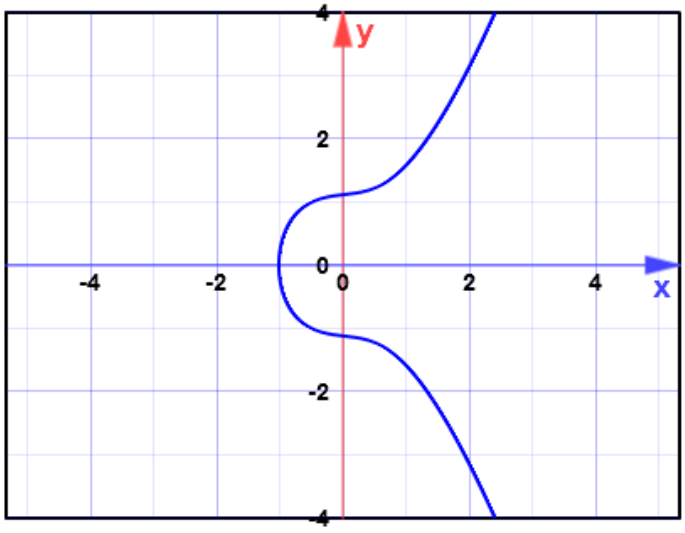
Örnek 3.1 : Gerçel sayıların R cismi üzerinde tanımlanmış olan E_1 *ve* E_2 eliptik eğrilerinin denklemi

$$E_1 : y^2 = x^3 - 2x - 2$$
$$E_2 : y^2 = x^3 + \mathrm{x} + 1 \qquad (5)$$

olarak verilsin. Bu eğrilerin grafikleri Şekil 4.1 ve Şekil 4.2. deki gibidir.



Şekil 4.1. Gerçel sayılar üzerinde eliptik eğrinin $\Delta < 0$ durumu.



Şekil 4.2. Gerçel sayılar üzerinde eliptik eğrinin $\Delta > 0$ durumu.

3.2.1. Basitleştirilmiş Weierstrass Denklemleri

Tanım 3.6. : E_1 *ve* E_2 **,** $\Re$ *veya* C gibi bir K cismi üzerinde tanımlanmış iki EE olsun ve Weierstrass denklemleri

$$E_1 : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$
$$E_2 : y^2 + \overline{a_1}xy + \overline{a_3}y = x^3 + \overline{a_2}x^2 + \overline{a_4}x + \overline{a_6} \quad (6)$$

ile verilsin. Eğer E_1 denkleminden E_2 denklemine

$$(x, y) \to \left(u^2x + r, u^3y + u^2sx + t\right) \quad (7)$$

şeklinde

ki dönüşümü sağlayacak $u, r, s, t \in K, u \neq 0$ var ise E_1 ve E_2 denklemlerinin K cismi üzerinde izomorfik olduğu söylenir. Buradaki (7) dönüşümü, *uygun değişken değişimi* olarak adlandırılır.

K üzerinde tanımlanan

$$E \ : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_5 \quad (8)$$

bir Weierstrass denklemi, uygun değişken değişimi yapılarak daha da basitleştirilebilir. Karakteristiği 2 veya 3'e eşit olan veya 2 veya 3 ten farklı olan öncelikli K cisimleri ayrı birer durum olarak düşünülebilir.

1. Eğer K'nın karakteristiği 2 veya 3 ten farklı ise o zaman uygun bir değişken değiştirilmesi

$$(x, y) \to \left(\frac{x - 3a_1^2 - 12a_2}{36}, \frac{y - 3a_1x}{216} - \frac{a_1^3 + 4a_1a_2 - 12a_3}{24} \right) \quad (9)$$

ile E'den

$$y^2 = x^3 + ax + b \qquad a, b \in K \quad (10)$$

eğrisine dönüşür. Bu eğrinin diskriminantı $\Delta = -16\left(4a^3 + 27b^2\right)$ şeklindedir.

2. Eğer K'nın karakteristiği 2 ise o zaman iki durum göz önüne alınır. Eğer $a_1 \neq 0$ ise o zaman uygun değişken değişimi

$$(x, y) \to \left(a_1^2 x + \frac{a_3}{a_1}, a_1^3 y + \frac{a_1^2 a_4 + a_3^2}{a_1^3} \right) \tag{11}$$

ile E'den

$$y^2 + xy = x^3 + ax + b \qquad a, b \in K \tag{12}$$

eğrisine dönüşür. Böylece eğrinin süpersingüler olmadığı söylenir ve diskriminantı $\Delta = b$ dir. Eğer $a_1 = 0$ ise o zaman uygun değişken değişimi

$$(x, y) \to (x + a_2, y) \tag{13}$$

ile E'den

$$y^2 + cy = x^3 + ax + b \qquad a, b, c \in K \tag{14}$$

eğrisine dönüşür. Böylece eğrinin süpersingüler olduğu söylenir ve diskriminantı $\Delta = c^4$ tür.

3. Eğer K'nın karakteristiği 3 ise o zaman iki durum dikkate alınır. Eğer $a_1^2 \neq -a_2$ ise o zaman uygun değişken değiştirme

$$\begin{aligned} &d_2 = a_1^2 + a_2 \quad ve \quad d_4 = a_4 - a_1 a_3 \quad için \\ &(x, y) \to \left(x + \frac{d_4}{d_2}, y + a_1 x + a_1 \frac{d_4}{d_2} + a_3 \right) \end{aligned} \tag{15}$$

ile E'den (10) eğrisine dönüşür. Böylece eğrinin süpersingüler olmadığı söylenir ve diskriminantı $\Delta = -a^3 b$ dir. Eğer $a_1^2 = -a_2$ ise o zaman uygun değişken değiştirme

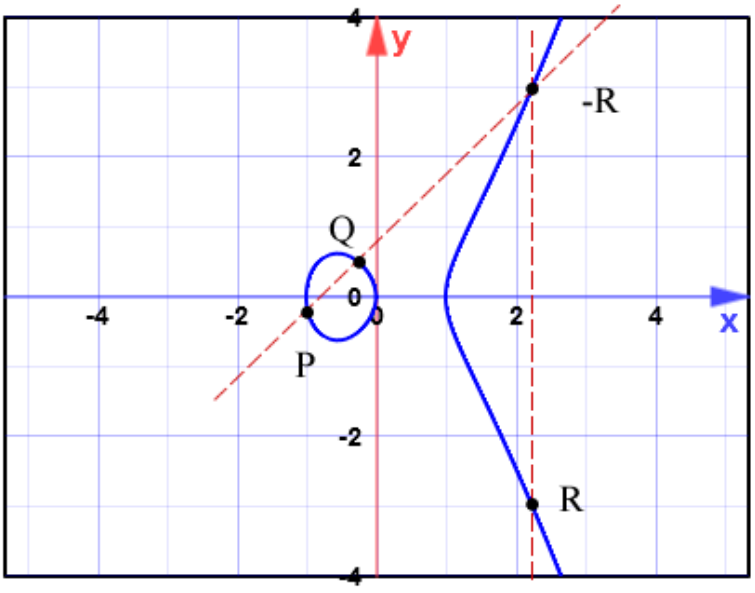
$$(x, y) \to (x, y + a_1 x + a_3) \tag{16}$$

ile E'den yine (10) eğrisine dönüşür. Böylece eğrinin süpersingüler olduğu söylenir ve diskriminantı $\Delta = -a^3$ dir (Hankerson ve Ark. 2003).

3.2.2 Eliptik Eğrilerde Grup Kanunu

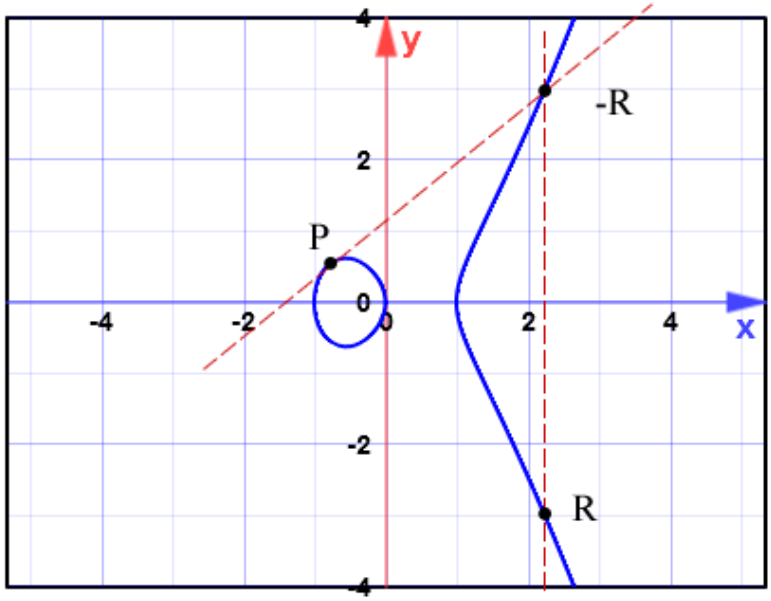
E, K cismi üzerinde tanımlı bir EE olsun. E(K) üzerindeki iki noktanın toplanmasıyla yine aynı E(K) üzerinde üçüncü bir noktanın elde edilmesi için bir doğru-ve-tanjan kuralı vardır. Bu toplama işlemi ile birlikte E(K) noktalar kümesi bir abelyen gurup oluşturur ve bu gurubun bu işlemdeki etkisiz elemanı ∞ dir. Bu gurup eliptik eğri şifreleme (EEŞ) sistemlerinin temelini oluşturur.

Toplama kuralı geometrik olarak kolayca açıklanabilir. $P = (x_1, y_1)$ ve $Q = (x_2, y_2)$ aynı bir E, EE üzerinde ayrık iki nokta olsun. O zaman P ve Q'nun toplamı olan R noktası aşağıdaki şekilde tanımlanır. İlk olarak P ve Q'yu birleştiren bir doğru çizilir; bu doğru EE'yi üçüncü bir noktada keser. Bu noktanın x eksenine göre simetriği olan nokta üçüncü nokta olan R noktasıdır. Bu durum Şekil 4.3. de gösterilmektedir.



Şekil 4.3. Eliptik eğridefarklı iki noktanın toplama kuralı.

P noktasının çiftlenmesiyle oluşan R noktası şöyle tanımlanır. İlk olarak P noktasında EE'ye bir tanjant doğrusu çizilir. Bu doğru eliptik eğriyi ikinci bir noktada keser. Bu durumda bu noktanın x eksenine göre simetriği R noktasıdır. Bu durum Şekil 4.4. de gösterilmektedir.



Şekil 4.4. Eliptik eğride aynı iki noktanın toplama kuralı.

1. *(Birim).* $\forall P \in E(K)$ *için* $P + \infty = \infty + P = P$

2. *(Negatiflik)* Eğer $P = (x, y) \in E(K)$ ise o zaman $(x, y) + (x, -y) = \infty$ dur. $(x, -y)$ noktası –P ile gösterilir ve P'nin negatifi olarak adlandırılır. Ayrıca $-\infty = \infty$ dir.

3. *(Nokta ekleme)* $P = (x_1, y_1) \in E(K)$ ve $P \neq \pm Q$ için $Q = (x_2, y_2) \in E(K)$ olsun. O zaman $P + Q = (x_3, y_3)$

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2 \text{ ve } y_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1 \qquad (17)$$

şeklindedir.

4. (*Nokta Çiftleme*) $P \neq -P$ olacak şekilde bir $P = (x_1, y_1) \in E(K)$ noktası alalım. O zaman $2P = (x_3, y_3)$

$$x_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1 \; ve \; y_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)(x_1 - x_3) - y_1 \tag{18}$$

şeklindedir (Washington 2008).

Örnek 3.2. (F_{23} asal cismi üzerinde EE) p=23 a=1 ve b=2 alalım. O zaman EE

$$E : y^2 = x^3 + x + 2$$

şeklinde F_{23} cismi üzerinde tanımlanır. Diskriminantı

$$\begin{aligned} \Delta &= -16\left(4a^3 + 27b^2\right) \\ &= -16\left(4.1^3 + 27.2^2\right) \\ &= -16 \neq 0 \left(\text{mod } 23\right) \end{aligned}$$

olduğundan dolayı E bir eliptik eğridir. $E\left(F_{23}\right)$ eliptik eğrisinin noktaları Çizelge 4.1. de gösterilmiştir.

Çizelge 4.1. Eliptik eğri noktaları.

(1,2)	(1,35)	(2,7)	(2,30)	(4,12)
(4,25)	(5,13)	(5,24)	(8,2)	(8,35)
(9,0)	(11,7)	(11,30)	(12,15)	(12,22)
(15,5)	(15,32)	(16,9)	(16,28)	(17,14)
(17,23)	(20,17)	(20,20)	(21,16)	(21,21)
(22,4)	(22,33)	(24,7)	(24,30)	(25,1)
(25,36)	(27,18)	(27,19)	(28,2)	(28,35)
(29,0)	(34,3)	(34,34)	(36,0)	

Tanım kümesinde y<0 da yer cisim noktalar Z_{23} *de* $\left(x, -y \pmod{23}\right)$ noktalarına taşınır.

3.2.3.Eliptik Eğrilerde Grup Derecesi

Tanım 3.7. : F_q üzerinde tanımlı EE, E olmak üzere, $E(F_q)$ daki noktaların sayısı $\#E(F_q)$ ile gösterilir ve bu gösterim F_q üzerinde tanımlı E eliptik eğrisinin *derecesi* olarak adlandırılır. Weierstrass denkleminin $\forall\ x \in F_q$ için en fazla iki çözüm bulunduğundan dolayı , $\#E(F_q) \in [1\ ,\ 2q+1]$ olduğu biliniyor. Hasse teoremi sınırlı cisimlarda tanımlı bir eliptik eğrinin kaç noktaya sahip olduğunu sınar.

Teorem 3.1 (Hasse)**:** $\#E(F_q), F_q$ üzerinde tanımlı bir E eliptik eğrisi üzerindeki noktaların toplam sayısı olmak üzere

$$q+1-2\sqrt{q} \le \#E(F_q) \le q+1+2\sqrt{q} \tag{19}$$

dir. $\left[q+1-2\sqrt{q}, q+1+2\sqrt{q}\right]$ aralığı Hasse aralığı olarak adlandırılır ve böylece $\#E(F_q)$ değeri

$$\left|\#E(F_q)-(q+1)\right| \le 2\sqrt{q} \tag{20}$$

ile sınırlandırılır (Enge 1999)(McReynolds 2008).

3.2.4. Eliptik Eğri Şifrelemeye Giriş

Günümüzde eliptik eğrilerin şifrelemedeki kullanımı giderek artmakta ve birçok açık anahtarlı şifreleme tekniklerinde eliptik eğriler önemli rol oynamaktadır. EEŞ'nin temellerinden biri de düz metindeki karakterlerin yani gönderilecek mesajın, EEŞ'nin mantığı olan iki boyutlu uzaydaki noktalara dönüştürülmesi olayıdır. Bu dönüşüm sadece verilen mesajın EE üzerindeki noktalara dönüştürülmesiyle kalmaz, EE üzerindeki noktalara dönüştürülmüş mesajın tekrar düz metini oluşturacak karakterlere ve böylece orijinal metinin oluşmasını sağlar.

3.2.4.1. Mesajların Eliptik Eğriye Yerleştirilmesi:

Şifrelenmek üzere metni oluşturan karakterlerin her biri Çizelge 5.1. 'e göre her bir karaktere birer doğal sayı gelecek şekilde birebir bir eşleme yaptırılır. Bu durum Çizelge 5.2.'e göre a=0, b=1, c=2, … z=29 şeklinde eşleme yaptırılarak karakterlerin her biri bir anlamda numarcisimdırılır.

Çizelge 5.1. Örnek karakterler tablosu.

A	b	c	d	e	f	g	h	ı	i
J	k	l	m	n	o	ö	p	q	r
S	ş	t	u	ü	v	w	x	y	z

Çizelge 5.2. Harf ve sayı eşleştirme tablosu.

a=0	b=1	c=2	d=3	e=4	f=5	g=6	h=7	ı=8	i=9
j=10	k=11	l=12	m=13	n=14	o=15	ö=16	p=17	q=18	r=19
s=20	ş=21	t=22	u=23	ü=24	v=25	w=26	x=27	y=28	z=29

Bu eşlemeyi yaparken seçilecek cisim noktalarının sayısı, Çizelge 5.1. karakter sayısı için yeterli olmalıdır.

$q = p^s$ olacak şekilde Z_q da tanımlı EE için *s* asal olmalıdır. *m,* bir mesajdaki her bir karakterin sayısal değerini ve M ise karakter tablosundaki karakterlerin sayısını göstermek üzere $0 \le m \le M$ ve $q > Mk$ ölçütleri alınır. Yeterince büyük seçilen bir $k \in N$ doğal sayısı, genelde $10 \le k \le 30$ şeklinde bir değerdir. Mesajların noktalara dönüştürülmesi işleminde kullanılan bu $k \in N$ değeri için oluşacak başarısızlık olasılığı $\frac{1}{2^k}$ dir. Aşağıdaki şekilde tanımlanmış kümenin

$$X = \{x : x = mk + j,\ 1 \le j \le k,\ \ k, j, m \in N\}$$

$x \in X$ değerlerinin her biri $y^2 = f(x) = x^3 + ax + b \bmod(q)$ şeklindeki denklemde yerine yazılarak $f(x)$'in bir tam karesi elde edilmeye çalışılır. Böylece bir m mesajı P_m şeklinde bir noktaya $1-1$ olarak dönüştürülmüş olur.

Örnek 3.3. : F_{631} üzerinde tanımlı $y^2 = x^3 + x + 1$ şeklindeki EE'ye yukarıdaki örnek karakter tablosunda bulunan 'e' karakterini yerleştirelim.

İlk olarak verilen eliptik eğrinin şifrelemeye uygunluğunu araştıralım. $4*1^3 + 27*1^2 = 31 \neq 0 (\text{mod}\, 631)$ olduğundan eliptik eğri süpersingüler değildir, böylece eliptik eğrinin şifrelemeye uygun olduğu görülür.

Şimdi Hasse teoremi ile eliptik eğri üzerindeki noktaların sayısının, mesaj yerleştirme işlemi için yeterliliğini kontrol edelim.

$$|N - (q+1)| \leq 2\sqrt{q}$$

$$|N - (769+1)| \leq 2\sqrt{769} = 55{,}4616$$

$$-55{,}4616 \leq N - 770 \leq 55{,}4616$$

$$714 \leq N \leq 825$$

$714 \leq N \leq 825$ nokta sayısı aralığı, mesaj karakterlerini eliptik eğriye gömmek için yeterlidir. Örnek karakter tablosundaki karakter sayısı M=30, dönüştürülmesini istediğimiz karakter ve eşlendiği sayı 'e=5', son olarak 'k =12' olarak alındığında $m = 5,\ k = 12,\ 0 < j \leq k$ *için*

$mk + j$	51 52 53 54 55 56 57 58 59 60
x	51 52 53 54 55 56 57 58 59 60

elde edilir.

Herbir x değeri teker teker verilen eliptik eğri denklemine yerleştirilip kontrol edilerek tamsayı bir 'y' değerine ulaşana kadar belirtilen aralıklarda x'in değeri bir arttırılır.

$$x = 51 \ \textit{için} \quad y^2 = x^3 + 3x$$

$$y^2 = 51^3 + 3*51$$

$$= 132804$$

$$= 294 (\text{mod}\, 631)$$

Bulunur ki $y^2 = 294 (\mathrm{mod}\, 631)$ olacak şekilde herhangi bir $y \in F_{631}$ bulunamaz, bu durumda verilen x değeri artırılarak

$$x = 55 \ \textit{için} \quad y^2 = x^3 + 3x$$

$$y^2 = 55^3 + 3*55$$
$$= 166540.0$$
$$= 587 (\mathrm{mod}\, 631)$$

Bulunur ki $y^2 = 587 (\mathrm{mod}\, 631)$ olacak şekilde

$$y = 43$$
$$y^2 = 43^2$$
$$= 1849$$
$$= 587 (\mathrm{mod}\, 631)$$

bir $y \in F_{631}$ bulunur, böylece "e=5" mesajı şekil 6 daki $y^2 = x^3 + 3x$ eliptik eğrisi üzerinde $P_e = (55,43)$ noktasına dönüşmüş olur.

3.2.4.2. Yerleşik Noktalardan Mesajın Elde Edilmesi

Mesajların EE'ye yerleştirilmesinin aksine, EE üzerindeki bir $P(x, y)$ noktasına karşılık gelecek şekilde yerleşik bir karakteri , $P(x, y)$ noktasından yararlanarak bulmak için

$$m = \left[\frac{x-1}{k} \right] \tag{19}$$

denkleminden yararlanılır.

Örnek 3.4. : Yukarıdaki örnekte 'e=5' mesajı, $y^2 = x^3 + 3x$ EE'sine $P_e = (55,43)$ olarak yerleştirilmişti, şimdi bu noktaya karşılık gelen mesaj

$$m = \left[\frac{x-1}{k} \right] = \left[\frac{55-1}{10} \right] = \left[\frac{54}{10} \right] = [5] = e$$

olarak bulunur.

3.2.4.3. Eliptik Eğri Tabanlı Geometri

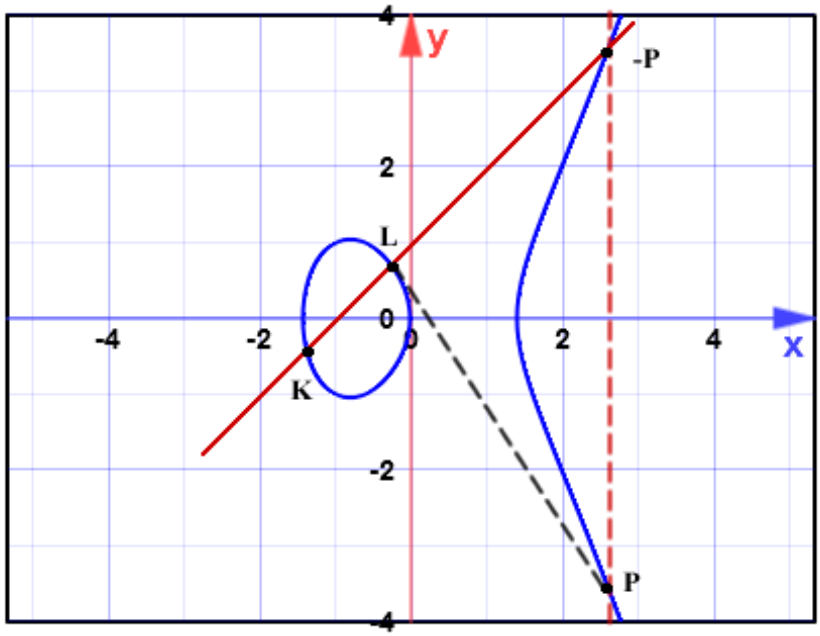
EEŞ diğer şifreleme tekniklerine göre matematiksel arkaplanı daha zengin olduğu için, eğrinin tanımlanacağı cisima ek olarak standart dışı toplama ve çarpma işlemeleri de tanımlanmaktadır.

Nokta Ekleme

$K(x_K, y_K)$ *ve* $L(x_L, y_L)$, EE üzerinde biribirinden farklı iki noktanın eklenerek $P(x_P, y_P)$ şeklinde başka bir noktanın oluşturulması işlemidir.

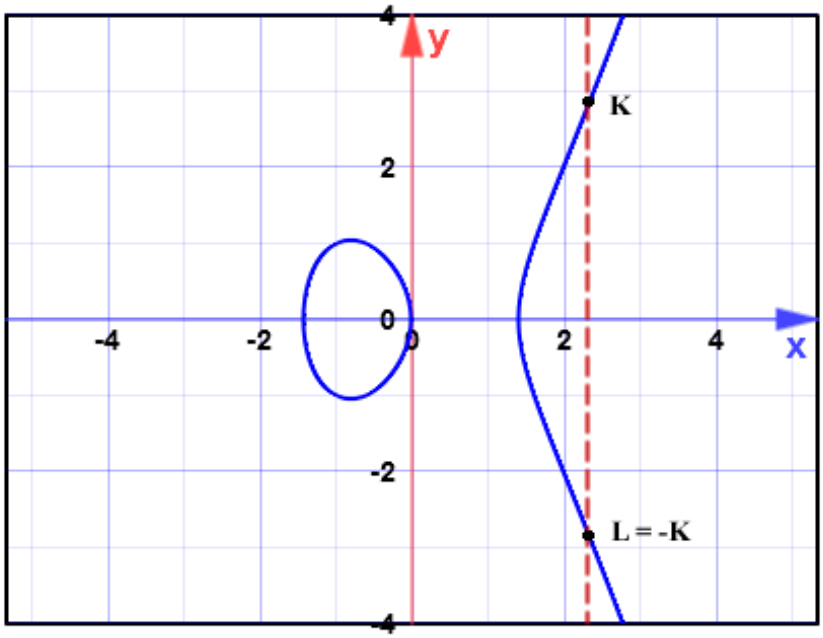
Geometrik Yaklaşım

$K(x_K, y_K)$, $L(x_L, y_L)$ *ve* $P(x_P, y_P)$, $K \neq L$ olmak üzere, eğer $x_K \neq x_L$ ise K ve L noktalarını birleştiren doğru, eğriyi üçüncü bir –P noktasında keser, bu –P noktasının x-eksenine göre simetriği olan P noktası K+L işleminin sonucudur. $K + L = P$ işlemi geometrik olarak Şekil 5.1. de gösterilmiştir.



Şekil 5.1 Nokta ekleme.

Eğer $x_K = x_L$ ise K ve L noktaları x-eksenine göre simetrik olur, bu durumda K ve L noktaları eğriyi sonsuzdaki 'O' noktasında keser. Böylece $K + L = O$ bulunur ve bu durum Şekil 5.2. de geometrik olarak gösterilmiştir.



Şekil 5.2. Nokta ekleme sonsuz durumu.

Aritmetik Yaklaşım

$K(x_K, y_K)$, $L(x_L, y_L)$, $P(x_P, y_P)$, $K \neq L$ aynı eliptik eğri üzerindeki noktalar , $m_{K,L}$ ise

$$m_{K,L} = \frac{y_L - y_K}{x_L - x_K} \tag{20}$$

şeklinde K ve L noktalarını birleştiren doğrunun eğimi olmak üzere $K + L = P$ işlemini sağlayan $P(x_P, y_P)$ noktası, $x_K \neq x_L$ ise

$$\begin{aligned} x_P &= m_{K,L}^2 - x_L - x_K \\ y_P &= m_{K,L}(x_P - x_L) + y_L \end{aligned} \tag{20}$$

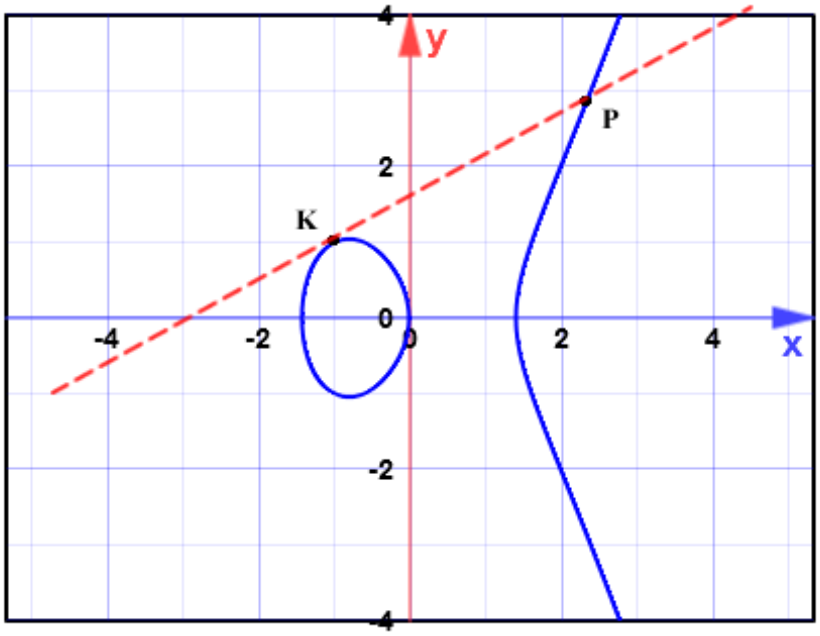
$x_K = x_L$ *ve* $y_K \neq y_L$ ise $L = -K$ dır ve $m_{K,L} = \infty$ olduğu için bu iki noktayı birleştiren doğru y-eksenine paraleldir dolayısıyla bu doğru eğriyi sonsuzdaki O noktasında keser böylece $K + L = O$ sonucuna ulaşılır.

Nokta Çiftleme

Nokta çiftleme, EE üzerindeki herhangi bir K noktasının kendisine eklenmesiyle farklı noktaların oluşturulmasıdır.

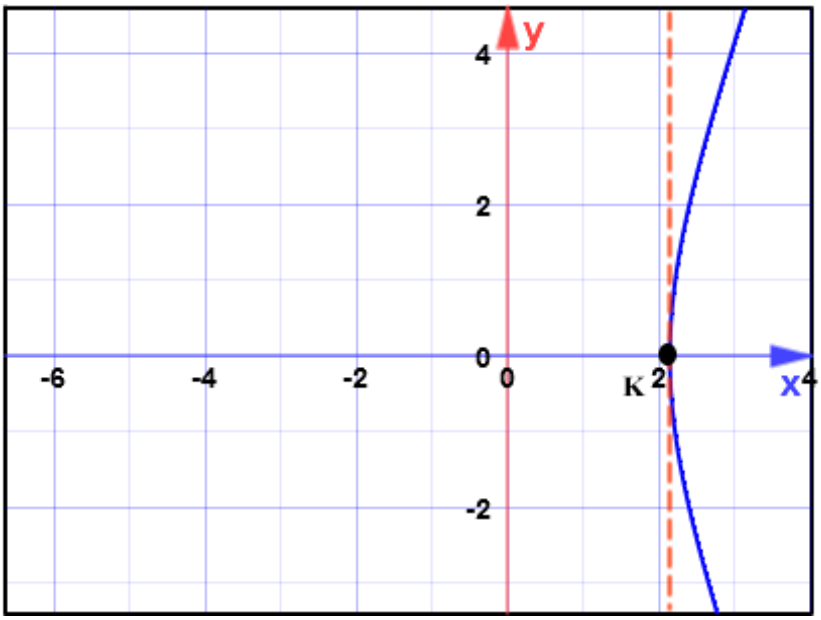
Geometrik Yaklaşım

$y_k \neq 0$ olmak üzere bir $K(x_k, y_k)$ noktasının kendisi ile toplamı, eğrinin $K(x_k, y_k)$ noktasındaki teğetinin eğriyi kestiği ikinci bir noktanın x-eksenine göre simetriğidir ve bu durum Şekil 5.3.te ifade edildiği gibi $K + K = 2K$ olur.



Şekil 5.3 Nokta çiftleme.

Eğer $y_k = 0$ ise $K(x_k, y_k)$ noktasındaki eğrinin teğeti y-eksenine paralel olduğu için bu teğet eğriyi sonsuz noktada keser bu durum Şekil 5.4. te ifade edildiği gibi $K + K = O$ dur.



Şekil 5.4 Nokta çiftleme sonsuz durumu.

Aritmetik Yaklaşım

EE üzerindeki $K(x_k, y_k)$ noktasının kendisi ile toplamı olan $K+K=2K=P(x_P, y_P)$ noktası eğer $y_k \neq 0$ ise

$$\begin{aligned} x_P &= m_{K,L}^2 - 2x_K \\ y_P &= m_{K,L}(x_P - x_K) + y_K \end{aligned} \tag{21}$$

şeklinde bulunur.

Eğer $K(x_k, y_k)$ noktası için $y_k = 0$ ise eliptik eğrinin bu noktadaki teğet doğrusu y-eksenine paralel olduğu için nokta çiftleme işleminin sonucu $K+K=O$ şeklinde sonsuz noktadır.

Nokta Çarpımı

Eliptik eğrilerde nokta çarpımı;

(i) Nokta ekleme ile K,L gibi farklı iki nokta toplanarak K+L=Q gibi diger bir nokta elde edilerek

(ii) Nokta çiftleme ile K gibi bir noktadan yararlanarak K+K=2K gibi bir farklı nokta elde edilerek

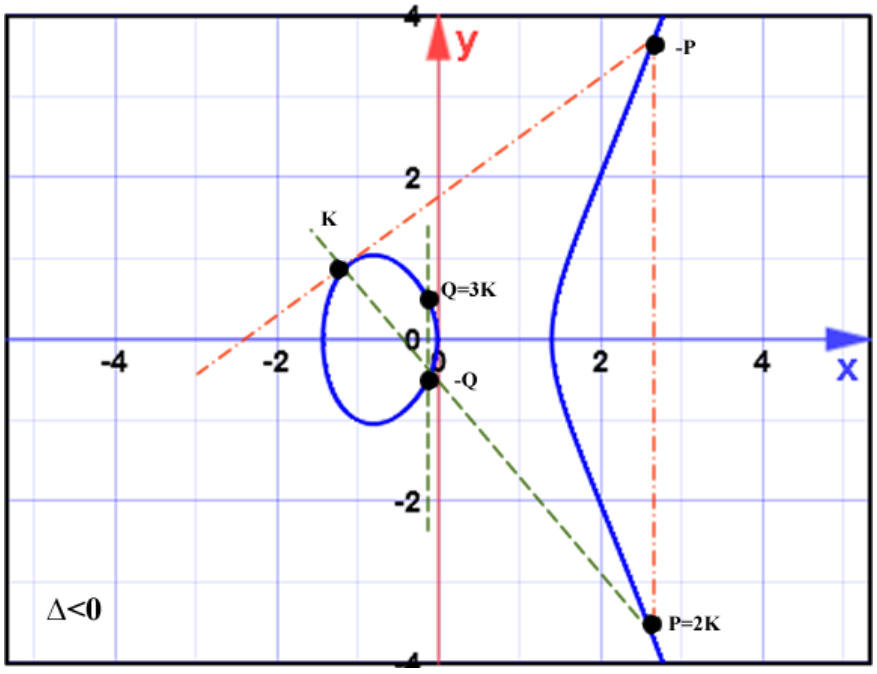
yapılabilir.

$Q \in E(a,b)$, $k \in Z^+$ *ve* $0 < k < n-1$ $n = \#E(a,b)$ için $K = kQ$ şeklinde farklı bir noktaya tekrarlı olarak nokta ekleme ve çiftleme işlemi yapılarak istenen sonuca en az işlem yapılarak ulaşılabilir.

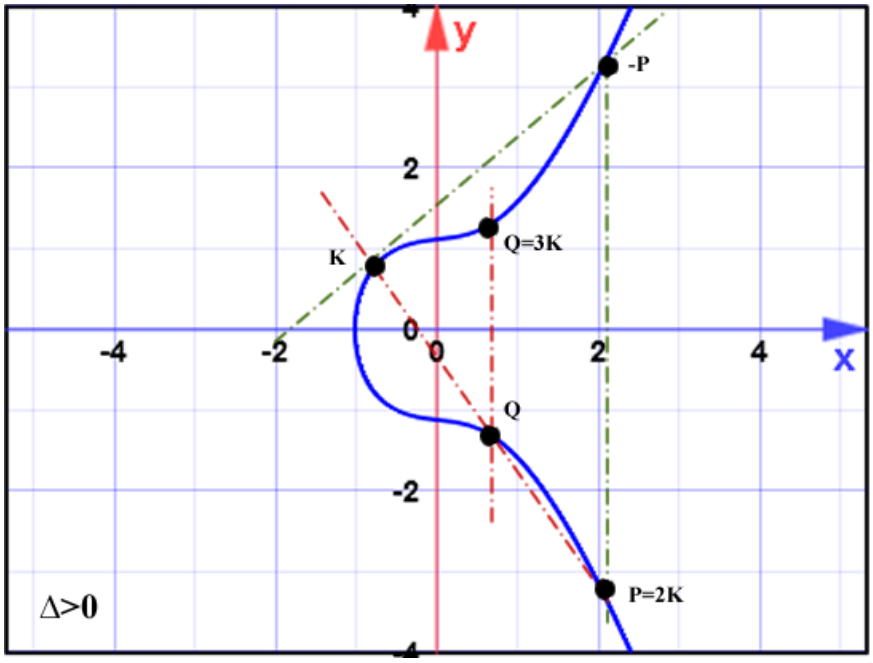
Örnek 3.5. : k=21 ve Q üreteç noktası ile $kQ = 21Q$ noktasına

$$Q + Q = 2Q$$
$$2Q + 2Q = 4Q$$
$$4Q + Q = 5Q$$
$$4Q + 4Q = 8Q$$
$$8Q + 8Q = 16Q$$
$$16Q + 5Q = 21Q$$

ile ulaşılabilinir. Eliptik eğrilerde nokta çarpım işlemi, Şekil 5.5. ve Şekil 5.6. de geometrik olarak gösterilmiştir.



Şekil 5.5. $\Delta < 0$ eğriler için nokta çarpımı.



Şekil 5.6 $\Delta > 0$ eğriler için nokta çarpımı.

3.2.5.Eliptik Eğri Tabanlı El-Gamal Şifreleme ve Deşifrelemesi (Friswell 2010)

Eliptik eğri tabanlı şifrelemeler açık anahtar sistemine dayalı olduğundan bu sistem güvenli olmayan bir ortamda karşılıklı ön bilgilerin gönderilmesiyle başlar, bu başlangıç işlemindeki ön bilgiler, *cisim parametreleri* olarak adlandırılır. Bu sistemin kırılmasının zorluğu eliptik eğri ayrık logaritma problemine dayanır (Yavuz 2008).

3.2.5.1. Başlangıç Cisim Parametreleri:

El-Gamal eliptik eğri şifrelemede kullanılacak olan parametreleri (p,a,b,L,n) olmak üzere;

-p eliptik eğride kullanılacak olan asal sayı

-(a,b) $y^2 = x^3 + ax + b$ şeklinde, bir F cismi üzerinde tanımlı E eliptik eğrisini oluşturan skalerler

-L ise E eliptik eğrisi üzerindeki üreteç noktası

-n asal sayısı ise üreteç noktasının derecesi yani L üreteç noktasından başlayıp sonsuz O noktasına kadar olan noktaların sayısı şeklindedir.

3.2.5.2. Anahtar Oluşturma:

Bir ‘p’ asal sayısı için F_p üzerinde tanımlı bir $L \notin E(F_p)$ noktasının derecesi, n asal sayısı olmak üzere, L noktasıyla oluşturulan devirli alt grup

$$\langle L \rangle = \{L, 2L, 3L, 4L, ...(n-1)L\} \tag{22}$$

şeklindedir. Bu devirli alt guruptaki n sayısı, L noktasının derecesi olup anahtar oluşturmak için gerekli olan ve $1 \le d \le n-1$ aralığında rasgele ‘d’ tamsayısı seçilerek $K = dL$ şeklinde ikinci bir nokta olan K noktası elde edilir. Bu K noktası şifrelemede kullanılacak olan açık anahtar, d tamsayısı ise özel anahtardır.

3.2.5.3. Mesaj Şifreleme:

Şifreleme işleminde ilk olarak gönderici L üreteç noktası ve ‘d’ özel anahtarıyla $K = dL$ şeklinde kendi açık anahtarını üretir daha sonra alıcının açık anahtarı Q ve EE üzerindeki noktalara gömülen M mesajı ile

$$\begin{aligned} U_1 &= dL \\ U_2 &= M + dQ \\ U &= (dL, M + dQ) \end{aligned} \tag{23}$$

şeklinde $U = (dL, M + dQ)$ gönderilecek şifreli metin oluşturulur.

3.2.5.4. Mesaj Deşifreleme:

Deşifreleme işleminde alıcı kendi açık anahtarını oluşturmak için kullandığı özel anahtar k ve göndericiden alınan şifrelenmiş M metnini

$$\begin{aligned} &U = (dL, M + dQ) \text{ için } U_1 = dL \text{ ve } U_2 = M + dQ \text{ olmak üzere} \\ &\Rightarrow kU_1 = k(dL) = d(kL) = dQ \\ &\Rightarrow M = U_2 - kU_1 = (M + dQ) - dQ \text{ olarak bulur.} \end{aligned} \tag{24}$$

Örnek 3.6. : Z_{769} üzerinde tanımlı $y^2 = x^3 + x + 1$ eliptik eğrisi ile “e=5” mesajı, k=10 için eliptik eğri üzerindeki (53,323) noktasına yerleşir, şimdi bu mesajı karşı tarafa şifreleyerek gönderelim.Öncelikle gönderici kendi açık anahtarını oluşturmak için

rasgele bir özel anahtar seçer, bu örnek için gönderici özel anahtar d_G=2 seçilsin, üreteç nokta L=(1,100) olarak seçilirse açık anahtar

$$K = d_G L = 2L$$

formülünden K = 2L bulunur, burada nokta çiftleme yöntemiyle L = (1,100) noktası için

$$egim = m = \frac{(3x_1 + a)}{2y_1} \mod(769)$$

$$= \frac{3*1^2 + 1}{2*100} = \frac{770}{200} \mod(769) = 325$$

bulunur ve eğimden yararlanarak K noktasının apsisi

$$x_K = m^2 - 2x_1 \ için$$

$$= (325)^2 - 2*1 \mod(769) = 512$$

ve ordinatı

$$y_K = m(x_1 - x_K) - 2x_1 \mod(631) \ için$$

$$= (325(1-512) - 2) \mod(769)$$

$$= 182$$

olarak elde edilir. Böylece $K = (512,182)$ bulunmuş olur, benzer olarak alıcı da kendi açık anahtarını aynı yöntemle elde eder, rasgele bir özel anahtar seçer, bu örnek için gönderici özel anahtar d_A=6 olarak seçilirse açık anahtar

$$Q = d_A P = 2P$$

formülünden Q=6P bulunur, burada iki nokta çiftleme ve bir toplama işlemi kullanılır , bu yöntem ile P=(1,100) üreteç noktası seçildiğinde;

P+P=2P işlemi için

$$egim = m = \frac{(3x_1^2 + a)}{2y_1} \mod(769)$$

$$= \frac{3*1^2 + 1}{2*100} = \frac{770}{200} \mod(769) = 325$$

olarak bulunur ve eğimden yararlanarak 2P noktasının apsisi

$$x_K = m^2 - 2x_1 \ için$$

$$= (325)^2 - 2*1 \mod(769) = 512$$

ve ordinatı

$$y_K = m(x_1 - x_K) - 2x_1 \ \text{mod}(631) \ \textit{için}$$
$$= (325(1-512) - 2)\,\text{mod}(769)$$
$$= 182$$
$$2P = (512,182) \ \textit{dir}.$$

2P+2P=4P işlemi için

$$\textit{egim} = m = \frac{\left(3x_1^2 + a\right)}{2y_1} \ \text{mod}(769)$$
$$= \frac{3*(512)^2 + 1}{2*182} = \frac{786433}{364}\,\text{mod}(769) = 350$$

bulunur ve eğimden yararlanarak 4P noktasının apsisi

$$x_K = m^2 - 2x_1 \ \textit{için}$$
$$= (350)^2 - 2*512 \ \ \text{mod}(769) = 743$$

ve ordinatı

$$y_K = m(x_1 - x_K) - 2x_1 \ \text{mod}(631) \ \textit{için}$$
$$= (350(512-743) - 2*512)\,\text{mod}(769)$$
$$= 482$$
$$4P = (743,482) \ \textit{dir}.$$

4P+2P=6P işlemi için

$$\textit{egim} = m = \frac{y_2 - y_1}{x_2 - x_1} \ \text{mod}(769)$$
$$= \frac{482-182}{743-512} = \frac{300}{231}\,\text{mod}(769) = 231$$

bulunur ve eğimden yararlanarak 6P noktasının apsisi

$$x_K = m^2 - x_1 - x_2 \textit{için}$$
$$= (231)^2 - 512 - 743 \ \ \text{mod}(769) = 583$$

ve ordinatı

$$y_K = m(x_1 - x_K) - 2y_1 \ \text{mod}(769) \ \textit{için}$$
$$= (231(512-583) - 182)\,\text{mod}(769)$$
$$= 335$$

olarak elde edilir. Böylece alıcının açık anahtarı $6P = (583,335)$ bulunmuş olur. Metnin şifrelenmesi için alıcı ve gönderen kendi aralarında açık anahtarlarını paylaşır daha sonra gönderen

$$\begin{aligned} U &= (dL, M + dQ) \\ &= (2*(1,100), (53,323) + 2(583,335)) \\ &= ((512,182), (53,323) + (243,65)) \\ &= ((512,182), (463,598)) \end{aligned}$$

şeklinde gerekli işlemleri yaparak şifrelenmiş mesajı $((512,182),(463,598))$ olarak elde eder. Deşifreleme işleminde ise alıcı, göndericinin açık anahtarını kendi özel anahtarı ile çarpar ve

$U = (dL, M + dQ) = ((512,182), (463,598))$ *ve* $P = (1,100)$ *için*

$U_1 = dL = 2P \quad U_2 = M + dQ = (463,598)$ *ve* $k = 6$ *alılı özel anahtarı*

$\Rightarrow kU_1 = k(dL) = 6(2P) = 12P = d(kL) = dQ = (243,65)$

$\Rightarrow M = U_2 - kU_1 = (M + dQ) - dQ$

$= (463,598) - (243,65)$

$= (463,598) + (243,-65)$

$= (463,598) + (243,704)$

$= (53,323)$

bulunur.

$\frac{[x-1]}{k} = \frac{[53-1]}{10} = [5] = e$

orijinal mesajı elde edilir.

3.2.6. Eliptik Eğri Tabanlı Diffie-Hellman Anahtar Değişimi Algoritması

Diffie-Hellman anahtar değişimi algoritması (DHADA) sonlu cisimlerin çarpımsal gurupları üzerinde tanımlı olup bu sistem, gönderici ve alıcı tarafından seçilecek asal sayı,sonlu cisim , eliptik eğri ve bu eliptik eğri üzerindeki bir noktanın belirlenmesi için anlaşarak başlar. Gönderici ve alıcı kendi aralarında, ayrık logaritma probleminin zorluk derecesini göz önde bulunduracak şekilde bir F_p cismi üzerinde tanımlı bir eliptik eğri seçer. Her iki taraf $K \in E(F_P)$ olacak şekilde ortak bir üreteç nokta seçer, üreteç noktanın derecesi ne kadar büyükse bu noktanın üreteç olarak eliptik eğri

uygulamalarındaki kullanım uygunluğu o kadar artar. Kullanılacak cismin, eliptik eğrinin ve üreteç noktasının seçiminden sonra gönderici ve alıcı sırasıyla α, β olacak şekilde rasgele bir sayı seçerek

$$\begin{aligned} P_G &= \alpha K \\ P_A &= \beta K \end{aligned} \tag{25}$$

şeklinde kendi açık anahtarlarını oluştururlar ve bu açık anahtarı biribirilerine gönderirler. Sırasıyla gönderici ve alıcı

$$M = \alpha P_A = \alpha(\beta K) = \beta(\alpha K) = \beta P_G \tag{26}$$

şeklinde paylaşımlı gizil (shared secret) olarak bilinen aynı M noktasını elde etmiş olurlar(Rodrigez ve Ark. 2007).

3.2.6.1. Şifreleme:

DHADA ile bir düz mesaj, eliptik eğri üzerindeki Q noktalarına dönüştürülüp, anahtar değişimiyle üretilen bir M paylaşımlı gizil noktası ile

$$Y = (u, j) \ , \ Q = (t, d) \ ve \ M = (f, c) \tag{27}$$

$$(u, j) = (f, c) + (t, d) \tag{28}$$

şeklinde şifreli $Y = (u, j)$ noktasına dönüştürülür.

3.2.6.2. Deşifreleme:

DHADA ile şifreli mesaj, anahtar değişimiyle üretilen bir M paylaşımlı gizil noktası ile (27) için

$$(t, d) = (u, j) - (f, c) \tag{29}$$

hesaplanarak orijinal metin olan $Q = (t, d)$ noktası elde edilir.

3.2.7. Eliptik Eğri Tabanlı Sayısal İmza Algoritması

Eliptik eğri sayısal imza algoritması için p asal ve $E(a,b)$ $F(p)$ üzerinde tanımlı bir eliptik eğri, $T=(r,t)$ ise derecesi q gibi ANSI X9.62 gereği ikilik tabanda $q>2^{160}$ şeklinde en az 160 bitlik uzunluğa sahip bir asal tamsayı olan bir üreteç nokta olsun(Rodrigez ve Ark. 2007).

3.2.7.1. İmzalama

Öncelikle gönderen $1<o_G<q-1$ olacak şekilde bir o_G özel anahtar seçer ve

$$P_G = o_G T \tag{30}$$

gibi bir P_G açık anahtarı oluşturur. Bir Q mesajının gönderen tarafından imzcisimması işlemi için $1<z<q-1$ olacak şekilde başka bir rasgele sayı seçilir ve ardından Q mesajı

$$\begin{aligned} U &= zT = (w,e) \\ h &= w \mod (q) \\ c &= (Q + o_G h)/z \mod (q) \end{aligned} \tag{31}$$

hesaplamaları ile bulunan (h,c) çifti ile imzcisimmış olur, burada dikkat edilmesi gereken husus (h,c) çiftinin her bir parametresinin sıfırdan farklı olmasıdır, olmaması durumunda rasgele seçilen z tamsayısı, ilgili parametreler sıfır olmaktan kurtarılana kadar değiştirilir ve işlemler tekrarlanır.

3.2.7.2. İmza Doğrulama

Alıcı tarafı mesaj ile gelen imzayı doğrulamak için $1<a<q-1$ olacak şekilde bir a rasgele tamsayısını seçer ve daha sonra imza olarak gönderilen (h,c) çiftinin her bir parametresi için

$$1 < h, c < q \tag{32}$$

doğrulamasını yapar ve

$$\begin{aligned} b &= Q/c \mod(q) \\ n &= h/c \mod(q) \\ bT + no_G &= (i, j) \\ m &= i \mod(q) \end{aligned} \tag{33}$$

değerlerini hesaplar, eğer m=h bulunursa imza doğrulanmış olur.

3.2.8. Eliptik Eğri Tabanlı El-Gamal İmzalama Şeması

Eliptik eğri El-Gamal sayısal imza algoritması (EEESİA) için p asal ve $E(a,b)$ $F(p)$ üzerinde tanımlı bir eliptik eğri, $T = (r,t)$ ise derecesi q gibi asal olması gerekmeyen bir üreteç nokta olsun (Babinkostova 2011).

3.2.8.1. İmzalama

Öncelikle gönderen $1 < o_G < q-1$ olacak şekilde bir o_G özel anahtar seçer ve (30) denklemindeki gibi bir P_G açık anahtarı oluşturur. Bir Q mesajının gönderen tarafından imzcisimması işlemi için $1 < z < q-1$ ve $(z,q) = 1$ olacak şekilde başka bir rasgele sayı seçilir ve ardından Q mesajı

$$\begin{aligned} U &= zT = (w, e) \\ h &= w \mod (q) \\ c &= (Q - o_G h)/z \mod (q) \end{aligned} \tag{33}$$

hesaplamaları ile bulunan (h,c) çifti ile imzcisimmış olur, burada dikkat edilmesi gereken ilk husus EEESİA'nda imza hesaplanırken kullanılan

$$c = (Q + o_G h)/z \mod (q)$$

yerine

$$c = (Q - o_G h)/z \mod (q)$$

kullanılmasıdır, ikinci husus ise (h,c) çiftinin her bir parametresinin sıfırdan farklı olmasıdır, olmaması durumunda rasgele seçilen z tamsayısı, ilgili parametreler sıfır olmaktan kurtarılana kadar değiştirilir ve işlemler tekrarlanır.

3.2.8.2. İmza Doğrulama

Alıcı tarafı mesaj ile gelen imzayı doğrulamak için $1 < a < q-1$ olacak şekilde bir a rasgele tamsayısını seçer ve daha sonra imza olarak gönderilen (h,c) çiftinin her bir parametresi için (32) ile doğrulamasını yapar ve

$$b = Q/c \mod(q)$$
$$n = h/c \mod(q)$$
$$bT - no_G = (i, j)$$
$$m = i \mod(q)$$

değerlerini hesaplar , yine burada EEESİA'nın doğrulama işlemlerindeki

$$bT + no_G = (i, j)$$

yerine

$$bT - no_G = (i, j)$$

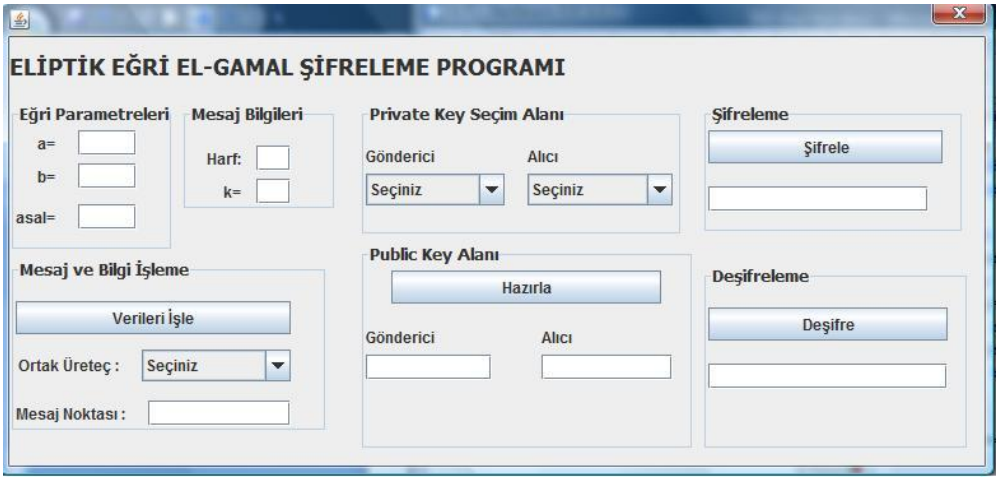
olduğuna dikkat edilmelidir. Eğer m=h bulunursa imza doğrulanmış olur.

3.2.9. Ayrık Logaritma ve Eliptik Eğri Ayrık Logaritma Problemi

Ayrık logaritma problemi, $m^x = y$ şeklindeki bir eşitlik için $x = \log_m^y$ şeklindeki x degerlerinin bulunması temeline dayanmaktadır (Coultas 2008)(Katz ve Ark. 2007). Eliptik eğri ayrık logaritma problemi ise kısaca bir eliptik eğri üzerindeki P ve Q noktası bilindiğinde $Q = kP$ şeklindeki bir denklemden k tamsayısının elde edilmesi problemidir(Cohen ve Ark. 2005). Bu noktadan hareketle eliptik eğri El-Gamal şifre çözme ve Diffie-Hellman anahtar degisim protokolü ayrık logaritma problemine dayanan başlıca algoritmalardır.

3.2.10. Eliptik Eğri El-Gamal Şifreleme Programı Tanıtımı

Örnek şifreleme programı, java programlama dili ile standart kriptoloji kütüphaneleri kullanılmadan temel düzeye indirgenip NetBeans IDE 6.9 kullanılarak yazılmıştır. İlk olarak masaüstü uygulaması çalıştırıldığında Şekil 5.7 görüntülenir.



Şekil 5.7. Java uygulamasının ekran görüntüsü

Sırasıyla, eliptik eğri parametreleri olan a,b ve asal sayı girildikten hemen sonra 'Mesaj Bilgileri' sekmesinde şifrelenecek harf ve uygun bir 'k' değeri girilir, 'Mesaj ve Bilgi işleme' sekmesindeki 'Verileri İşle' butonuna basılır, bu butonun görevi , girilen parametrelerin EEŞ için uygun olup olmadığını sınar, üreteç noktası oluşturur, şifrelenecek harfi EE üzerindeki bir noktaya dönüştürür ve son olarak gönderici ve alıcının private key bilgilerini oluşturur. Üreteç noktası, gönderenin ve alıcının private keyleri belirlenmesinin ardından 'Public Key' sekmesindeki 'Hazırla' butonuna basılarak iki tarafında public keyleri oluşturularak tüm safhalar tamamlanır, böylece 'Şifreleme' sekmesindeki 'Şifrele' butonuna basılarak EE noktası şifrelenir, 'Deşifreleme' sekmesindeki 'Deşifrele' butonuna basılarakta şifrelenmiş metin deşifre edilir.

EEŞ sisteminin simüle edilmiş örnek programının özünde kullanılan metodlar ve açıklamalar aşağıda verilmiştir.

'Parametrekontrol' metodu, EE domain parametrelerinin EEŞ uygunluğunu kontrol eder.

```
public  bolean  Parametrekontrol(int asal,int a,int b){
  Bolean sonuc= true;
  For(int i=2;i<asal;i++){
```

```
 if(asal%i==0){  sonuc=false;  }
 }
if(((-16)*(4*(a*a*a)+27*(b*b)))%asal==0){  sonuc=false;  }
return sonuc;
}
```

'Donusturapsis' ve 'Donusturordinat' metodları, 'k' parametresi ile şifrelenecek harfi EE üzerinde karşılık gelecek ilgili noktaya dönüştürme görevi görür.

```
public  int Donusturapsis(int sayi,int k, int a,int b,int modn){
   int islem1,y=0,i,j;
   int start=(k*sayi)+1;
   int end =(k*sayi)+k+1;
   for (i=start;i<end;i++){
       islem1 = ((i*i*i)+a*i+b);
       islem1 = islem1%modn;
      for(j=0;j<modn;j++){
         int p= (j*j)%modn;
         if(p==islem1){
            y=j;
          break;
         }
      }
      if(y!=0){
      break;
      }
      if(i==(end-1)&&y==0){ System.out.println("k="+k+" parametresi yetersiz değiştiriniz!!"); }
   }
   return i;
   }
public  int Donusturordinat(int sayi,int k, int a , int b, int modn){
   int islem1,y=0,i,j=0;
   int start=(k*sayi)+1;
   int end =(k*sayi)+k+1;
   for (i=start;i<end;i++){
       islem1 = ((i*i*i)+a*i+b);
       islem1 = islem1%modn;
      for(j=0;j<modn;j++){
```

```
            int p= (j*j)%modn;
            if(p==islem1){
               y=j;
             break;
            }
        }
        if(y!=0){
        break;
        }
        if(i==(end-1)&&y==0){ System.out.println("k="+k+" parametresi yetersiz değiştiriniz!!"); }
    }
    return j;
    }
```

'Noktabul' metodu, EE domain parametreleri ile oluşturulacak gurubun elemanlarını bulur.

```
public void Noktabul(int sayi , int a, int b,int diziuzunluğu){
long y,p,q;
int i,j,z=0;
Object noktadizisi[] = new Object[diziuzunluğu+1];
for(i=0;i<sayi;i++)
{
y=(i*i*i)+a*i+b;
y=y%sayi;
p=(i*i*i)+a*i+b;
q=p%sayi;
for(j=0;j<sayi;j++){
          if((j*j)%sayi==y){
            z++;
           noktadizisi[z]=i+","+j;
          }
       }
}
jComboBox1.setModel(new javax.swing.DefaultComboBoxModel(noktadizisi));

}
```

'Noktasayisi' metodu, EE domain parametreleri ile oluşturulacak gurubun eleman sayısını bulur.

```
public int Noktasayisi(int sayi , int a, int b){
long y,p,q;
int i,j,z=0;
for(i=0;i<sayi;i++)
{
y=(i*i*i)+a*i+b;
y=y%sayi;
p=(i*i*i)+a*i+b;
q=p%sayi;
for(j=0;j<sayi;j++){
        if((j*j)%sayi==y){
          z++;
        }

     }


}
return z;
}


public void noktatopla(int modn, int x, int y, int a,int noktasayisi){
         double x3,y3,egim = 0.0;
         for(int k=0;k<modn;k++){
           egim = ((double )((((double) 3*x*x)+ (double) a)+ k*modn)/(double) (2*y));
           if (egim==Math.round(egim)){
            break;
           }
         }
        x3 = ((egim*egim)-2*x)%modn;
        y3 =(egim*(x-x3)-y);
        if((y3%modn)<0){
          y3=(y3%modn)+modn;
```

```
            }
            y3=y3%modn;
        for(int i=3;i<=noktasayisi;i++){
            if(x!=x3&&y!=y3){
               for(int k=0;k<modn;k++){
                   egim = (double )((double)(((double) y3)- (double) y)+k*modn)/ (double) ((double)(x3)-
(double)(x));
                    if (egim==Math.round(egim)){
                     break;
                    }
                  }
                  x3 = ((egim*egim)-x-x3)%modn;
                  y3 =(egim*(x-x3)-y);
                  if(y3<0){
                  for(int g=1;g<modn;g++){
                     double c =y3+g*modn;
                       if(c>0){
                       y3=c%modn;
                       break;
                       }
                     }
                  }else{
                     y3=y3%modn;
                  }
            }

            if(x==x3&&y==y3&&y!=0){
                for(int k=0;k<modn;k++){
               egim = (((double)(((double) 3*x*x)+ (double) a)+ k*modn)/(double) (2*y));
                  if (egim==Math.round(egim)){
                    break;
                  }
                }
              x3 = ((egim*egim)-2*x)%modn;
              y3 =(egim*(x-x3)-y);
              if(y3<0){
                  for(int g=1;g<modn;g++){
                     double c =y3+g*modn;
                       if(c>0){
```

```
                    y3=c%modn;
                    break;
                    }
                  }
                }else{
                   y3=y3%modn;
                }
            }
        }
    }
```

'Alicipkbul' metodu, üreteç, alıcı private key ve diğer bilgiler ile alıcının public key noktasını oluşturur.

```
public String Alicipkbul(String uretec,int egrinoktasayisi,int aliciprivatekey,int modn,int a ){
int x,y;
Pattern pat = Pattern.compile(",");
String strs[] = pat.split(uretec);
x=Integer.parseInt(strs[0]);
y=Integer.parseInt(strs[1]);
double x3,y3,egim = 0.0;
x3=Integer.parseInt(strs[0]);
y3=Integer.parseInt(strs[1]);
String arr[]=new String[egrinoktasayisi+1];
arr[1]=x3+","+y3;
for(int i=2;i<=aliciprivatekey;i++){
        if(x!=x3&&y!=y3){
           for(int k=0;k<modn;k++){
              egim = (double )((double)(((double) y3)- (double) y)+k*modn)/ (double) ((double)(x3)-
(double)(x));
                if (egim==Math.round(egim)){
                  break;
                }
               }
               x3 = ((egim*egim)-x-x3)%modn;
               y3 =(egim*(x-x3)-y);
               if(x3<0){
               for(int g=1;g<modn;g++){
```

```
            double l =x3+g*modn;
              if(l>0){
              x3=l%modn;
              break;
              }
            }
        }else{
            x3=x3%modn;
        }
        if(y3<0){
        for(int g=1;g<modn;g++){
            double c =y3+g*modn;
              if(c>0){
              y3=c%modn;
              break;
              }
            }
        }else{
            y3=y3%modn;
        }
        int aa = (int)x3;
        int bb = (int)y3;
          arr[i]=aa+","+bb;
}
if(x==x3&&y==y3&&y!=0){
     for(int k=0;k<modn;k++){
    egim = (((double)(((double) 3*x*x)+ (double) a)+ k*modn)/(double) (2*y));
        if (egim==Math.round(egim)){
            break;
        }
     }
   x3 = ((egim*egim)-2*x)%modn;
   y3 =(egim*(x-x3)-y);
   if(y3<0){
        for(int g=1;g<modn;g++){
            double c =y3+g*modn;
              if(c>0){
              y3=c%modn;
              break;
```

```
                    }
                   }
                }else{
                   y3=y3%modn;
                }
                int aa = (int)x3;
                int bb = (int)y3;
                 arr[i]=aa+","+bb;
            }
        }
     return arr[aliciprivatekey];
    }
```

'Gondericipkbul' metodu, üreteç, gönrerici private key ve diğer bilgiler ile göndericinin public key noktasını oluşturur.

```
    public String Gondericipkbul(String uretec,int egrinoktasayisi,int gondericiprivatekey,int modn,int a
){
    int x,y;
    Pattern pat = Pattern.compile(",");
    String strs[] = pat.split(uretec);
    x=Integer.parseInt(strs[0]);
    y=Integer.parseInt(strs[1]);
    double x3,y3,egim = 0.0;
    x3=Integer.parseInt(strs[0]);
    y3=Integer.parseInt(strs[1]);
    String arr[]=new String[egrinoktasayisi+1];
    arr[1]=x3+","+y3;


    for(int i=2;i<=gondericiprivatekey;i++){
        if(x!=x3&&y!=y3){
           for(int k=0;k<modn;k++){
               egim = (double )((double)(((double) y3)- (double) y)+k*modn)/ (double) ((double)(x3)-
(double)(x));
                 if (egim==Math.round(egim)){

                  break;
```

```
            }
        }
        x3 = ((egim*egim)-x-x3)%modn;
        y3 =(egim*(x-x3)-y);
        if(x3<0){
        for(int g=1;g<modn;g++){
            double l =x3+g*modn;
              if(l>0){
              x3=l%modn;
              break;
              }
            }
        }else{
            x3=x3%modn;
        }
        if(y3<0){
        for(int g=1;g<modn;g++){
            double c =y3+g*modn;
              if(c>0){
              y3=c%modn;
              break;
              }
            }
        }else{
            y3=y3%modn;
        }
        int aa = (int)x3;
        int bb = (int)y3;
          arr[i]=aa+","+bb;
}
if(x==x3&&y==y3&&y!=0){
     for(int k=0;k<modn;k++){
    egim = (((double)(((double) 3*x*x)+ (double) a)+ k*modn)/(double) (2*y));
        if (egim==Math.round(egim)){
            break;
        }
     }
   x3 = ((egim*egim)-2*x)%modn;
   y3 =(egim*(x-x3)-y);
```

```
            if(y3<0){
                for(int g=1;g<modn;g++){
                   double c =y3+g*modn;
                    if(c>0){
                    y3=c%modn;
                    break;
                    }
                   }
                }else{
                   y3=y3%modn;
                }
                int aa = (int)x3;
                int bb = (int)y3;
                 arr[i]=aa+","+bb;
        }
    }
 return arr[gondericiprivatekey];
}
```

'Carp' metodu , EE üzerindeki bir noktayı verilen diğer bir skaler ile çarpma işlemini yerine getirir.

```
public String Carp(String uretec,int egrinoktasayisi,int carpan,int modn,int a ){
int x,y;
Pattern pat = Pattern.compile(",");
String strs[] = pat.split(uretec);
System.out.println("uretec="+uretec);
x=Integer.parseInt(strs[0]);
y=Integer.parseInt(strs[1]);
System.out.println("x="+x+"  y="+y);
double x3,y3,egim = 0.0;
x3=Integer.parseInt(strs[0]);
y3=Integer.parseInt(strs[1]);
String arr[]=new String[egrinoktasayisi+1];

arr[1]=x3+","+y3;
for(int i=2;i<=carpan;i++){
        if(x!=x3&&y!=y3){
```

```
for(int k=0;k<modn;k++){
 egim = (double )((double)(((double) y3)- (double) y)+k*modn)/ (double) ((double)(x3)-
(double)(x));
  if (egim==Math.round(egim)){

   break;
  }
 }
 if(egim<0){
 for(int g=1;g<modn;g++){
  double l =egim+g*modn;
   if(l>0){
   egim=l%modn;
   break;
   }
  }
 }else{
  egim=egim%modn;
 }

 x3 = ((egim*egim)-x-x3)%modn;
 y3 =(egim*(x-x3)-y);
 if(x3<0){
 for(int g=1;g<modn;g++){
  double l =x3+g*modn;
   if(l>0){
   x3=l%modn;
   break;
   }
  }
 }else{
  x3=x3%modn;
 }
 if(y3<0){
 for(int g=1;g<modn;g++){
  double c =y3+g*modn;
   if(c>0){
   y3=c%modn;
   break;
```

```
            }
           }
        }else{
           y3=y3%modn;
        }
        int aa = (int)x3;
        int bb = (int)y3;
         arr[i]=aa+","+bb;
}
if(x==x3&&y==y3&&y!=0){
    for(int k=0;k<modn;k++){
   egim = (((double)(((double) 3*x*x)+ (double) a)+ k*modn)/(double) (2*y));
        if (egim==Math.round(egim)){
           break;
        }
    }
        if(egim<0){
        for(int g=1;g<modn;g++){
           double l =egim+g*modn;
            if(l>0){
            egim=l%modn;
            break;
            }
           }
        }else{
           egim=egim%modn;
        }
  x3 = ((egim*egim)-2*x)%modn;
  y3 =(egim*(x-x3)-y);
  if(y3<0){
        for(int g=1;g<modn;g++){
           double c =y3+g*modn;
            if(c>0){
            y3=c%modn;
            break;
            }
           }
        }else{
           y3=y3%modn;
```

```
                }
                if(x3<0){
                for(int g=1;g<modn;g++){
                   double l =x3+g*modn;
                    if(l>0){
                    x3=l%modn;
                    break;
                    }
                   }
                }else{
                   x3=x3%modn;
                }
                int aa = (int)x3;
                int bb = (int)y3;
                 arr[i]=aa+","+bb;          }
    }
  return arr[carpan];
}
```

'Topla' metodu verilen iki EE noktasını toplar, bu toplama işlemi yapılırken iki noktanın kendi aralarındaki benzerlik durumları kontrol edilerek her bir duruma özel işlemler gerçekleştirilir.

```
public String Topla(String ilknokta,String sonnokta,int modn,int a ){
double ilk_x,ilk_y,son_x,son_y;
Pattern pat = Pattern.compile(",");
String ilk_strs[] = pat.split(ilknokta);
ilk_x=Integer.parseInt(ilk_strs[0]);
ilk_y=Integer.parseInt(ilk_strs[1]);
Pattern pat1 = Pattern.compile(",");
String son_strs[] = pat.split(sonnokta);
son_x=Integer.parseInt(son_strs[0]);
son_y=Integer.parseInt(son_strs[1]);
double x3=0.0,y3=0.0,egim = 0.0;

        if(ilk_x!=son_x&&ilk_y!=son_y){
           for(int k=0;k<modn;k++){
```

```
                egim = (double )((double)(((double) son_y)- (double) ilk_y)+k*modn)/ (double)
((double)(son_x)-(double)(ilk_x));
                  if (egim==Math.round(egim)){
                    break;
                  }
                }
                if(egim<0){
                for(int g=1;g<modn;g++){
                   double l =egim+g*modn;
                     if(l>0){
                     egim=l%modn;
                     break;
                     }
                   }
                }else{
                   egim=egim%modn;
                }
                x3 = ((egim*egim)-ilk_x-son_x)%modn;
                y3 =(egim*(ilk_x-x3)-ilk_y);
                if(x3<0){
                for(int g=1;g<modn;g++){
                   double l =x3+g*modn;
                     if(l>0){
                     x3=l%modn;
                     break;
                     }
                   }
                }else{
                   x3=x3%modn;
                }
                if(y3<0){
                for(int g=1;g<modn;g++){
                   double c =y3+g*modn;
                     if(c>0){
                     y3=c%modn;
                     break;
                     }
                   }
                }else{
```

```
            y3=y3%modn;
        }

}
if(ilk_x==son_x&&ilk_y==son_y&&ilk_y!=0){
     for(int k=0;k<modn;k++){
    egim = (((double)(((double) 3*ilk_x*ilk_x)+ (double) a)+ k*modn)/(double) (2*ilk_y));
        if (egim==Math.round(egim)){
            break;
        }
     }
        if(egim<0){
        for(int g=1;g<modn;g++){
            double l =egim+g*modn;
              if(l>0){
              egim=l%modn;
              break;
              }
            }
        }else{
            egim=egim%modn;
        }
   x3 = ((egim*egim)-2*ilk_x)%modn;
   y3 =(egim*(ilk_x-x3)-ilk_y)%modn;

        if(x3<0){
        for(int g=1;g<modn;g++){
            double l =x3+g*modn;
              if(l>0){
              x3=l%modn;
              break;
              }
            }
        }else{
            x3=x3%modn;
        }

        if(y3<0){
        for(int g=1;g<modn;g++){
```

```
                double c =y3+g*modn;
                  if(c>0){
                  y3=c%modn;
                  break;
                  }
                }
            }else{
                y3=y3%modn;
            }
        }
            int aa = (int)x3;
            int bb = (int)y3;

     return aa+","+bb;
    }
```

‘Toplavecarp’ metodu, verilen iki EE noktasını topla ve çarp alt metoduyla toplar ve çarpar. Bu metod mesajın gönderilmeye hazır hale gelmesini sağlar.

```
    public void Toplavecarp(String gondericipk,int egrinoktasayisi,int gondericiprivatekey,String
alicipk,String mesajnoktasi,int a,int b,int modn ){
    int alicipk_x,alicipk_y,mesaj_x,mesaj_y,gondericipk_y,gondericipk_x;
    double x3,y3,egim = 0.0,sonuc1_x,sonuc1_y;
    String carpilannokta,toplanannokta;
    carpilannokta=carp(alicipk, egrinoktasayisi,gondericiprivatekey, modn, a) ;
    toplanannokta = topla(carpilannokta, mesajnoktasi, modn, a);
    sifrelitext.setText(gondericipk+","+toplanannokta);
    }
```

‘Desifrele’ metodu, girilen şifreli metini diğer ek parametreler ile işleme tabi tutarak şifrelenmeden önce ki EE üzerindeki noktalara geri dönüştürür.

```
    public void Desifrele (String sifrelimetin,int modn,int k,int a, int egrinoktasayisi,int carpan){
    int ilk_x,ilk_y,son_x,son_y;
    Pattern pat = Pattern.compile(",");
    String ilk_strs[] = pat.split(sifrelimetin);
    ilk_x=Integer.parseInt(ilk_strs[0]);
    ilk_y=Integer.parseInt(ilk_strs[1]);
```

```
String uretec = ilk_x+","+ilk_y;
String ilknokta = carp(uretec, egrinoktasayisi, carpan, modn, a);
Pattern pat2 = Pattern.compile(",");
String nokta_strs[] = pat2.split(ilknokta);
ilk_x=Integer.parseInt(nokta_strs[0]);
ilk_y=modn-Integer.parseInt(nokta_strs[1]);
ilknokta = ilk_x+","+ilk_y;
son_x=Integer.parseInt(ilk_strs[2]);
son_y=Integer.parseInt(ilk_strs[3]);
String sonnokta = son_x+","+son_y;
String sonuc = topla(ilknokta,sonnokta, modn, a);
Pattern pat3 = Pattern.compile(",");
String x_strs[] = pat3.split(sonuc);
int desifre_x=Integer.parseInt(x_strs[0])/k;
String sifresizmetin = numbertotext(desifre_x);
desifretext.setText(sifresizmetin);
}
```

Texttonumber metodu girilen harfin sayı karşılığını döndürür, böylece şifreleme işleminin ilk dönüşümü sağlanmış olur.

```
public int texttonumber(char  harf){
int number=30;
   switch(harf){
   case 'a' :number=0; break;
   case 'b' : number=1; break;    case 'c' : number=2; break;    case 'ç' : number=3; break;
   case 'd' : number=4; break;    case 'e' : number=5; break;    case 'f' : number=6; break;
   case 'g' : number=7; break;    case 'ğ' : number=8; break;    case 'h' : number=9; break;
   case 'ı' : number=10; break;    case 'i' : number=11; break;    case 'j' : number=12; break;
   case 'k' : number=13; break;    case 'l' : number=14; break;    case 'm' : number=15; break;
   case 'n' : number=16; break;    case 'o' : number=17; break;    case 'ö' : number=18; break;
   case 'p' : number=19; break;    case 'r' : number=20; break;    case 's' : number=21; break;
   case 'ş' : number=22; break;    case 't' : number=23; break;    case 'u' : number=24; break;
   case 'ü' : number=25; break;    case 'v' : number=26; break;    case 'y' : number=27; break;
   case 'z' : number=28; break;    default:            System.out.println("deger bulunamadı");

 }
return number;
```

```
}
```

Numbertotext metodu girilen sayının harf karşılığını döndürür, bu durum deşifreleme işleminin son adımıdır.

```
public String Numbertotext(int  sayi){
int number=30;
String deger=null;
   switch(sayi){
   case 0 :   deger="a";  break;    case 1 : deger="b";break;    case 2 : deger="c"; break;
   case 3 : deger="ç";break;    case 4 : deger="d"; break;    case 5 : deger="e";break;
   case 6 : deger="f";break;    case 7 : deger="g";break;    case 8 : deger="ğ";break;
   case 9 : deger="h";break;    case 10 : deger="ı";break;    case 11 : deger="i";break;
   case 12 :deger="j";break;    case 13 :deger="k";break;    case 14 :deger="l";break;
   case 15 : deger="m";break;    case 16 : deger="n";break;    case 17 : deger="o";break;
   case 18 : deger="ö";break;    case 19 : deger="p";break;    case 20 : deger="r";break;
   case 21 : deger="s";break;    case 22 : deger="ş";break;    case 23 : deger="t";break;
   case 24 : deger="u";break;    case 25 : deger="ü";break;    case 26 : deger="v";break;
   case 27 : deger="y";break;    case 28 : deger="z";break;
   default: System.out.println("deger bulunamadı");
 }
return deger;
}
```

4. BULGULAR VE TARTIŞMA

EEŞ sisteminin RSA ve diğer güçlü şifreleme tekniklerine göre üstünlüğü, aynı güvenlik seviyesini daha az anahtar uzunluğu ile gerçekleştirmesidir.

Çizelge 7.1 Anahtar uzunluğu ve güvenlik karşılaştırılması (Cohen ve Ark. 2005)

	Bant Genisligi		Anahtar Uzunlugu	
	2000 bitlik uzun mesajlar için imza büyüklügü (bit)	100 bit uzunlugundaki mesajın sifrelendikten sonrası uzunlugu	Açık Anahtar (bit)	Gizli Anahtar (bit)
RSA	1024	1024	1088	2048
ECC	320	321	161	160

Çizelge 7.1'de bu durum açıkça ortaya çıkmakta, anahtar uzunluğunun şifreleme sistemlerine göre kıyaslanması verilmektedir (Cohen ve Ark. 2005).

5. SONUÇ VE ÖNERİLER

Günümüz teknolojisinin vaz geçilmez unsurlarından biri olan güvenilirliğin, elektronik iletişimde önemi git gide artamakta ve buna paralel olarak hızına yetişilmekte güçlük çekilecek ilerlemelere sebep olmaktadır. Elektronik ve diğer ilgili iletişim sistemlerinde kullanılan şifreleme sistemleri uygulama cisimlerini genişleterek insanların hayatlarını kolaylaştıracak birçok yeniliğe temel oluşturmuş ve buna devam etmektedir.

Teknolojinin ilerlemesine paralel olarak RSA, DES, 3DES ve diğer şifreleme sistemlerine ECC Eliptik Eğri Şifreleme gibi güçlü sistemler geliştirilmiştir. Eliptik eğri şifreleme bu anlamda diğer güvenli ve hızlı olarak nitelendirilen şifreleme tekniklerine göre aynı güvenlik seviyesinde düşük anahtar uzunluğunu kullanması, geleceğin bu sistemlere ihtiyaç duyacagını göstermektedir.

6. KAYNAKLAR

Babinkostova L. (2011). http://math.boisestate.edu/~liljanab/Crypto2Spring10/ec eg1.htm,(.Sayfa 19-23,24).

Cohen H.,Frey G.,Avanzi R.,Doche C.,Lange T.,Nguyen K., ve Vercauteren F. (2005), Handbook of Elliptic and Hyperelliptic Curve Cryptography, Chapman & Hall/CRC, CRC Press Company, T&F Group (Sayfa 1)

Coultas M.(2008),Elliptic Curves and Cryptography,Rapor,(Sayfa 13)

Enge A.(1999), Elliptic Curve and Their Applications to Cryptography, Kluwer Academic Publisher Group (Sayfa 95)

Friswell R.(2010),Elliptic Curves, Cryptography and Factorisation, Durham University Department of Mathmatical Sciences, 4. Proje, (Sayfa 65-72)

Hankerson D. , Menezes A. , Vanstone S. (2003) Guide to Elliptic Curve Cryptography. Springer-Verlag,(Sayfa 76,78-79)

Katz J.,Lindell Y. (2007),Introduction to Modern Cryptography, Chapman & Hall/CRC, CRC Press Company.(Sayfa 277)
McReynolds J.(2008),Elliptic Curves and Cryptography, Rapor,(Sayfa 34).

Rodríguez-Henríquez F., Pérez A., Saqib A.N., ve Koç Ç.K. (2007). Cryptographic Algorithms on Reconfigurable Hardware. Springer,(.Sayfa 19-23,24).

Standarts for Efficent Cryptography (2000),SEC 1: Elliptic Curve Cryptography, Araştırma,(Sayfa 3)

Yavuz İ. (2008), Eliptik Eğri Kriptosisteminin FPGA Üzerinde Gerçeklenmesi. Yüksek Lisans Tezi (Sayfa 9)

Washington L.C.(2008),Elliptic Curves Number Theory and Cryptography Second Edition, Chapman & Hall/CRC T&F Group,(Sayfa 12-18)

ÖZGEÇMİŞ

Adı Soyadı : Aziz Mahmut YÜCELEN

Doğum Yeri : Diyarbakır

Doğum Tarihi : 06.06.1980

Medeni Hali : Evli

Yabancı Dil : İngilizce

Eğitim Durumu (Kurum ve Yılı)

Lise	:	Yunus Emre Lisesi	1994-1997
Lisans	:	Dicle Üniversitesi –Fen Fakültesi Matematik	1999-2003

Çalıştığı Kurum/Kurumlar ve Yıl:

D.Ü. Bilgi İşlem	2003-2004
Karacadağ Dershanesi	2005-2007
Final Dershanesi	2007-2008
Dicle Üniversitesi	2008- …