YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELİPTİK EĞRİ ŞİFRELEME KULLANARAK GÜVENLİ SOKET KATMANI PROTOKOLÜ'NÜN GERÇEKLENMESİ VE PERFORMANSININ DEĞERLENDİRİLMESİ

Yük. Bilgisayar Müh. Ömer Özgür Bozkurt

FBE Bilgisayar Mühendisliği Anabilim Dalında Hazırlanan

DOKTORA TEZİ

Tez Danışmanı : Prof. Dr. Oya KALIPSIZ

İSTANBUL, 2005

İÇİNDEKİLER **Sayfa**

SİMGE LİSTESİ

a, b, c, k, l	Tamsayı
x, y	Gerçek sayı
p, q	Asal sayı
n	Bileşik tamsayı
Z_p	*p* asal sayısı tarafından belirlenen tamsayılar kümesi
F_p	*p* asal sayısı tarafından belirlenen alan
$E(F_p)$	F_p tarafından belirlenen eliptik eğri
# $E(F_p)$	Eliptik eğrinin düzeyi
P, Q, R	Eliptik eğri noktaları
O	Eliptik eğri sonsuz noktası
O(n)	Algoritma karmaşıklığı
(*x*,*y*)	Koordinat ekseni üzerinde *x* ve *y* değerleri ile belirlenen nokta

KISALTMA LİSTESİ

ADA	Anahtar Değişim Algoritması (bkz. KEA)
ALP	Ayrık Logaritma Problemi (bkz. DLP)
ANSI	American National Standards Instute (Amerikan Ulusal Standartlar Enstitüsü)
CA	Certificate Authority (bkz. SS)
DH	Diffie-Hellman
DES	Digital Encryption Standard (Sayısal Şifreleme Standardı)
DSA	Digital Signature Algorithm (bkz. SİA)
EEALP	Eliptik Eğri Ayrık Logaritma Problemi (bkz. EEDLP)
EEDH	Eliptik Eğri Diffie Hellman
EESİA	Elliptik Eğri Sayısal İmza Algoritması
GSK	Güvenli Soket Katmanı (bkz. SSL)
HTTP	Hyper Text Transfer Protocol (Hiper Metin Aktarım Protokolü)
IEEE	Institute of Electrical and Electronics Engineers (Elektrik ve Elektronik Mühendisleri Enstitüsü)
IMAP	Internet Message Access Protocol (Internet İleti Erişim Protokolü)
IP	Internet Protocol (Internet Protokolü)
ISO	International Organization for Standardization (Uluslararası Standartlar Enstitüsü)
LDAP	Lightweight Directory Access Protocol (Dizin Erişim Protokolü)
KEA	Key Exchange Algorithm (bkz. ADA)
MAC	Message Authentication Code (İleti Doğrulama Kodu)
MD	Message Digest (İleti Özeti)
MIPS	Million Instructions Per Second (Saniyede Milyon İşlem))
NAF	Non Adjacent Form (Doğrusal Olmayan Biçim)
NIST	National Institute of Standards and Technology (Ulusal Standartlar ve Teknoloji Enstitüsü)
RFM	Radio Frequency Modulation (Radyo Frekans Modülasyonu)
RSA	Rivest Shamir Adleman
SS	Sertifika Sağlayıcı (bkz. CA)
SHA	Secure Hash Algorithm (Güvenli hash algoritması)
SİA	Sayısal İmza Algoritması (bkz. DSA)
SSL	Secure Sockets Layer (bkz. GSK)
TCP	Transport Control Protocol (Aktarım Kontrol Protokolü)
TÇAP	Tamsayı Çarpanlarına Ayırma Problemi
VPN	Virtual Private Network (Sanal Özel Ağ)

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

İletişimin başlangıcı ile birlikte, aktarılan verilerin güvenliği ile ilgili kaygılar da başlamıştır. Özellikle askeri bilgilerin güvenliğini sağlama çabaları ile Sezar şifresinden Enigma'ya uzanan; ardınan iletişimin her alanında ihtiyaç duyulan ve kullanılmaya başlanan şifreleme yöntemleri tarihin akışı içerisinde kaydadeğer bir aşama kaydetmiştir. Çocukluğumda 'kuş dili' konuşarak sağlamaya çalıştığımız iletişim güvenliği, tez çalışmam ile benim açımdan zirveye ulaşmış bulunmaktadır.

Tez çalışmam süresince beni yönlendirdikleri, bilgi, görüş ve deneyimlerini bana aktardıkları için danışmanım Sayın Prof. Dr. Oya KALIPSIZ'a ve doktora çalışmalarına başladığım dönemdeki danışmanım Sayın Prof. M. Yahya KARSLIGİL'e;.

Engin bilgi ve deneyimlerini benimle paylaşıp, yol gösterdikleri ve yardımcı oldukları için, tez izleme komitesinin değerli üyeleri Sayın Prof. Dr. Bülent ÖRENCİK ve Sayın Yrd. Doç. Dr. A. Gökhan YAVUZ'a;

Tezin hazırlanması aşamasında bana destek olan ve yardımlarını esirgemeyen sevgili mesai arkadaşlarım Sırma YAVUZ, Göksel BİRİCİK ve Ekin Su UĞURLU ile Yıldız Teknik Üniversitesi Bilgisayar Mühendisliği Bölümündeki bütün arkadaşlarıma,

Akademisyenlik yolunda ilerlemem için beni destekleyen, eğitim hayatım süresince maddi-manevi her türlü destekle yanımda olacağını defalarca vurgulayan, ancak bir trafik kazası sonucu bizlere erkenden veda ederek doktora çalışmalarımı göremeyen sevgili babama,

Hayatım boyunca desteğini hiçbir zaman eksik etmeyen, beni yüreklendiren ve bana güvenen sevgili anneme,

Doktora çalışmalarımı kendisine rakip olarak görmeyen, her zaman yanımda olduğunu hissettiren, desteğini esirgemeyen sevgili eşime,

Teşekkürü bir borç bilirim.

ÖZET

Günümüzde iletişim çoğunlukla internet omurgası üzerinden gerçekleştirilmektedir. Bu iletişim çok basit anlamda sohbet etmekten, en mahrem bilgilerin aktarılmasına kadar oldukça farklılık gösterebilmektedir. Internet gibi kolayca erişilebilen bir ortamda verilerin güvenliğinin sağlanması bir sorun olarak ortaya çıkmakta, sağlanan güvenliğin maliyeti ise performans kaybı olmaktadır. Güvenli Soket Katmanı, internet üzerinden aktarılan verilerin güvenliğinin sağlanması için en yaygın kullanılan protokoldür. Hizmet verdiği üst katman protokolünden bağımsız olarak işlemesi nedeniyle her platformda kullanılabilmekte ve dolayısı ile yaygın kullanım alanı bulabilmektedir. Güvenli Soket Katmanı'nın sağladığı güvenliğin maliyeti, performans kaybı olarak karşımıza çıkmaktadır. Yapılan çalışmalar, güvenli soket katmanının sunucu performansında %85e varan kayıplara neden olabildiğini göstermiştir.

Güvenli Soket Katmanında performans kaybının en temel nedeni, Tokalaşma Protokolü işleyişi sırasında açık anahtar şifreleme algoritmalarının getirdiği ek yük olarak gözükmektedir. Mevcut olarak güvenli soket katmanında en yaygın kullanılan açık anahtar şifreleme algoritması RSA algoritmasıdır. RSA algoritmasının sağladığı güvenliği daha yüksek performans ile sağlayan bir şifreleme Güvenli Soket Katmanı performansında belirgin bir artış sağlayabilecektir.

Her türlü dış müdahaleye açık işlemekte olan duyarga ağları da güvenli iletişime ihtiyaç duyulan ortamların başında gelmektedir. Duyarga ağlarında güvenlik, uygulamalar tarafından gerçekleştirileceği varsayılarak açık bırakılmıştır. Özellikle farklı amaçlara hizmet eden cihazların oluşturduğu ağlarda güvenliğin uygulama tarafından gerçekleştirilmesi beklenemez. Duyarga ağ cihazlarının sınırlı kapasiteleri de klasik şifreleme algoritmalarının duyarga ağlarında kullanılmasına olanak vermemektedir. Bu durumda aktarılan verilerin şifrelemesi için simetrik şifreleme, gerektiğinde şifreleme anahtarlarının değiştirilmesi için de açık anahtar şifreleme yöntemlerinin kullanılması seçenek olarak karşımıza çıkmaktadır. Bu amaçla anahtar değişimi için eliptik eğri şifreleme algoritmalarının nasıl kullanılabileceği irdelenmiştir.

Whitfield Diffie ve Martin Hellman tarafından 1976 yılında açık anahtar şifreleme tekniklerinin ortaya atılmasından bu yana çok sayıda açık anahtar şifreleme yöntemi önerilmiştir. Belirli bir matematiksel problemin çözümündeki zorluğa dayananan bütün bu yöntemler içerisinde eliptik eğri ayrık logaritma problemine dayanan yöntemlerin geçmişi çok daha kısadır. Eliptik eğriler üzerinde 19. yüzyıldan beri çalışıla gelmiş olmasına rağmen, şifreleme amacıyla kullanımı 1985 yılından daha önce değildir.

Eliptik eğri şifreleme algoritmaları temelde, yaygın olarak kullanılan klasik şifreleme algoritmalarının, tamsayı alanlarından eliptik eğrilere taşınması temeline dayanmaktadır ve eliptik eğri şifreleme algoritmalarının dayandığı matematiksel problem olan eliptik eğri ayrık logaritma probleminin çözüm karmaşıklığı, klasik şifreleme algoritmalarının dayandığı matematiksel problemler olan ayrık logaritma problemi ve tamsayı çarpanlarına ayırma probleminin karmaşıklığından çok daha yüksektir. Bu durumda, eliptik eğri şifreleme, çok daha kısa anahtarlarla, klasik şifreleme algoritmalarının sağladığı güvenliğe eşdeğer güvenlik sağlayabilmektedir.

Bu tez çalışmasında, sağladığı güvenliğin diğer açık anahtar şifreleme yöntemlerinden daha yüksek olduğu kanıtlanmış olan eliptik eğri şifreleme kullanarak; Güvenli Soket Katmanı vasıtasıyla daha yüksek performansla güvenli iletişim sağlanabileceği ve duyarga ağlarında simetrik şifreleme anahtarlarını dağıtmanın mümkün olduğu gösterilmiştir.

ABSTRACT

Contemporarily, communication is generally realized through internet. This type of communication can vary considerably, from simple chatting to exchange of private information. Ensuring data security in an easily accessible environment like the Internet could present a problem. Usually it is required not to compromise on performance while ensuring data security. Secure Sockets Layer is the most common protocol used to ensure the security of the data transmitted over the Internet. Being independent from the protocol of the upper layer it served, it can be used on any desired platform and is widely deployed. The cost of the security provided by the Secure Sockets Layer is degradation in performance. Previous studies showed that secure sockets layer can cause a loss in server performance up to %85.

In this study, by using elliptic curve cryptography method, which is proved to be more secure than the other public key cryptography methods, it is showed that secure communication can be achieved with higher performance using Secure Sockets Layer.

1. GİRİŞ

Yarı iletkenlerin ortaya çıkışı ile üç ayrı teknoloji dalgasının oluşumunu sağladığı söylenmektedir; birinci dalga 70lerin ortalarında büyük bilgisayarların dönemi, ikinci dalga 80li yıllarda kişisel bilgisayar devrimi ve üçüncü dalga ise 90larda yaşanan Internet patlamasıdır. Her bir dalga öncekinin beş katı bir büyümeyi de beraberinde getirmiştir. Bunların ardından gelmekte olan dalga ise "evrensel bağlanabilirlik" ya da "iletişim ve bağlanabilirlik" olarak adlandırılabileceğimiz dördüncü dalgadır. Gelen ya da yaşamakta olduğumuz dalga için anahtar, uygulamalara güvenilir, hatasız, ucuz ve güvenli bağlanabilirliğin sağlanmasıdır [1].

Bilgisayarlar ya da benzer şekilde iletişim kuran gömülü cihazlar için artık bire bir kablo bağlantısına gerek duyulmamaktadır. Çoğunlukla, iletişim amacıyla internet omurgası üzerinden kurulan açık bağlantılar ya da sanal özel ağ (VPN) bağlantıları, uygulamaların farklılaştığı durumlarda ise kablosuz iletişim ihtiyaçları karşılamaktadır. Her iki durumda da üçüncü şahısların araya girerek hassas bilgilere erişimini önlemek önemli bir sorun olarak ortaya çıkmaktadır. İnternet üzerinden iletişim söz konusu olduğunda güvenli soket katmanı ilk akla gelen güvenlik sağlama yöntemlerinden birisi olmaktadır. İnternet üzerinden güvenli iletişim olanağı sağlayan sunucular Güvenli Soket Katmanı'ndan faydalanmakta, bu ise kullanıcıların yanıt alma sürelerini düşürmektedir. Düşen performans zaman zaman işlemlerin sonuçlandırılmamasına bile neden olabilmektedir (Clark 2000). Güvenlik kaygısıyla tamamlanmayan işlemlere ilave olarak bir de bu şekilde performans düşüklüğü nedeniyle sonlandırılamayan işlemler eklendiğinde dikkat çekici değerlere ulaşılmaktadır.

Kolayca araya girilebilen, aktarılan verilerin üçüncü şahısların kontrolüne rahatlıkla geçebildiği ortamlardan birisi de duyarga ağlarıdır. Duyarga ağları; doğal gözlem, sağlık durumunun gözlenmesi, acil sağlık desteği, araç takibi ve askeri uygulamalar benzeri ortamlarda kullanım için öngörülmektedir (Lorincz vd 2002). Bu ortamların tümünde de yetkilendirme, bütünlük, gizlilik ve güvenlik gereksinimlerinin bileşimlerine ihtiyaç duyulmaktadır. Duyarga ağlarında kullanılan cihazların sınırlı kapasiteleri mevcut güvenlik uygulamalarını bu tür ortamlar için uygulanamaz kılmaktadır (Perrig vd 2001). Bu tür cihazlar için, çok fazla işlem gücü ya da enerji gerektiren işlemler uygun olmadığından açık anahtar şifreleme algoritmaları güvenliği sağlamak için kullanışlı gözükmemektedir. Açık anahtar şifreleme, veri aktarım güvenliğinin sağlanması için kullanılamazken, simetrik şifreleme anahtarlarının dağıtılması için kullanılabilir. Açık anahtar şifreleme yöntemleri ile ilgili sıkıntı

ise anahtar boylarının düşük kapasitedeki cihazların işlem gücünün çok üzerinde boyutlarda olmasıdır.

Whitfield Diffie ve Martin Hellman (1976) tarafından açık anahtar şifreleme tekniklerinin ortaya atılmasından bu yana çok sayıda açık anahtar şifreleme yöntemi önerilmiştir. Bu yöntemlerin tümü belirli bir matematiksel problemin çözümündeki zorluğa dayanmaktadır.

Bu yöntemler içerisinde eliptik eğri ayrık logaritma problemine dayanan yöntemlerin geçmişi çok daha kısadır. Eliptik eğriler üzerinde 19. yüzyıldan beri çalışıla gelmiş olmasına rağmen, şifreleme amacıyla kullanımı 1985 yılından daha önce değildir. 1985 yılında eliptik eğriler üzerindeki noktalar grubunu kullanarak farklı ayrık logaritma şifreleme yöntemlerinin gerçeklenebileceği ortaya konmuş (Koblitz 1987a; Miller 1986), ardından, tamsayı çarpanlarına ayırma problemine dayanan açık anahtar şifreleme yöntemlerinin eliptik eğriler üzerinde de gerçekleştirilebileceği ve eliptik eğri gerçekleştirimlerinin daha üst düzeyde güvenlik sağladığı ispatlanmıştır.

Güvenli Soket Katmanı'nın performans kaybının önüne geçecek, duyarga ağ cihazlarının işleyebileceği boyutlarda anahtarlarla yüksek güvenlik sağlayabilecek çözüm, açık anahtar şifreleme yöntemleri arasında, sağladığı güvenlik ile ön plana çıkan eliptik eğri şifreleme olarak gözükmektedir. Bu çalışma eliptik eğri şifreleme kullanarak Güvenli Soket Katmanı Protokolü'nün gerçeklenebileceğini ve performansının da aynı zamanda arttırılabileceğini göstermeyi amaçlamaktadır. Eliptik eğrilerin sağladığı olanakların Güvenli Soket Katmanı Protokolü'ne özgü olmadığını göstermek amacıyla duyarga ağlarında eliptik eğri şifreleme kullanılarak anahtar dağıtımı gerçekleştirilmiştir.

Tezin kalan kısmı aşağıdaki şekilde düzenlenmiştir:

2. Bölümde, açık anahtar şifreleme sistemlerinin altyapısını oluşturan matematiksel problemler açıklanmakta, çözüm algoritmaları hakkında bilgi verilmektedir. Eliptik eğri şifreleme algoritmalarının dayandığı matematiksel problem ile diğerleri arasındaki fark açıklanarak, eliptik eğri şifrelemenin sağladığı güvenlik gösterilmesi amaçlanmıştır.

3. Bölümde, eliptik eğriler açıklanmış, ardından eliptik eğrilerin şifreleme alanında nasıl kullanıldığı anlatılmıştır. Bu amaçla öncelikle eliptik eğrilerin temelini oluşturan matematiksel altyapı özetlenmiştir.

4. Bölümde, Güvenli Soket Katmanı hakkında genel bilgiler verilmiş, işleyişi açıklanmıştır. Güvenli Soket Katmanı Protokolü'nün desteklediği şifreleme yöntemleri belirtilmiş ve özellikle Tokalaşma Protokolü işleyişi üzerinde durulmuştur.

5. Bölümde, duyarga ağları hakkında genel bilgi verilmiş, güvenlik gereksinimleri özetlenmiştir.

6. Bölümde, geliştirilen eliptik eğri kütüphanesi açıklanmış, ardından kütüphanenin Güvenli Soket Katmanı ve duyarga ağları uyarlamaları açıklanarak performans değerlendirmeleri yapılmıştır.

Sonuç bölümünde de, tez çalışmasında elde edilenler özetlenmiş, eliptik eğri şifreleme sistemlerinin halefi olduğu şifreleme sistemlerine olan avantajları vurgulanmıştır.

Tez çalışmasında kullanılan eliptik eğrilere ilişkin parametreler Ek 1'de sunulmuştur.

2. AÇIK ANAHTAR ŞİFRELEME SİSTEMLERİ

Açık anahtar şifreleme sistemlerinin Whitfield Diffie ve Martin Hellman tarafından ilk olarak ortaya atıldığı 1976 yılından günümüze pek çok açık anahtar şifreleme sistemi önerilmiştir. Bu sistemlerin tümü belirli bir matematiksel problemin çözümündeki zorluğa dayanmaktadır.

Geçen süre içerisinde önerilen açık anahtar şifreleme sistemlerinin birçoğu kırılmış, pek çoğunun ise kullanışlı olmadığı ispatlanmıştır. Günümüzde sadece üç tür şifreleme sistemi hem güvenli hem de etkin kullanılabilir olarak değerlendirilmektedir. Bu sistemlerin başlıcaları ve dayandıkları matematiksel problemler aşağıda verilmiştir:

- Tamsayı Çarpanlarına Ayırma Problemi (TÇAP) (Odlyzko 1995). RSA (Rivest 1978) ve Rabin-Williams (Rabin 1979) şifreleme yöntemleri
- Ayrık Logaritma Problemi (ALP) (Odlyzko 2000). Sayısal İmza Algoritması (SİA) (NIST 1994), Diffie Hellman Anahtar Değişimi (DH) (Diffie ve Hellman 1976), El Gamal Şifreleme ve İmza Yöntemleri (ElGamal 1985), Schnorr İmza Yöntemi (Schnorr 1991) ve Nyberg-Rueppel İmza Yöntemi (Nyberg vd. 1996)
- Eliptik Eğri Ayrık Logaritma Problemi (EEALP) (Menezes vd. 1993) . Eliptik Eğri Sayısal İmza Algoritması (EESİA) (ANSI 1998), Eliptik Eğri Diffie Hellman (EEDH) Anahtar Değişimi (ANSI 1999), Eliptik Eğri ElGamal Şifreleme ve İmza Yöntemi (Hankerson vd. 2004), Eliptik Eğri Schnorr İmza Yöntemi (Pointcheval ve Stern 1996) ve Eliptik Eğri Nyberg-Rueppel İmza Yöntemi (IEEE 2000)

Bu yöntemlerin hiçbirisi için etkin çözüm sağlayan bir yöntem olmadığı ispatlanabilmiş değildir. Yıllar süren yoğun çalışmalar sonucunda böyle bir çözüm bulunamadığından, çözümsüz olduğuna inanılmaktadır.

2.1 Tamsayı Çarpanlarına Ayırma Problemi (TÇAP)

TÇAP, yıllar süren çalışmalara rağmen, kesin çözüm elde edilememiş matematiksel problemlerden birisi olarak pek çok şifreleme sistemine temel teşkil etmiştir. Aşağıda TÇAP tanımı ve çözümü için yapılan çalışmalar verilmiştir.

2.2 Tanım

Büyük iki asal sayı, p ve q, çarpımından oluşan bileşik n sayısı verildiğinde p ve q sayılarının bulunması TÇAP olarak adlandırılır.

Büyük asal sayıların bulunması göreceli olarak basit bir iş olmasına rağmen, asal sayılar dikkatli seçilmişse, çarpımlarının çarpanlarına geri ayrılması problemi işlemsel olarak zordur. Problemin zorluğuna dayanarak RSA ve Rabin-Williams şifreleme yöntemleri geliştirilmiştir.

TÇAP, başta Fermat ve Gauss olmak üzere yüz yıllardır matematikçilerin ilgisini çekmesine rağmen, çözümü için en büyük mesafe son 20 yılda kat edilmiştir. Bunun iki nedeni bulunmaktadır: Birincisi 1978 yılında RSA şifreleme sisteminin ortaya çıkışının matematikçilerin dikkatini çekmesi; ikincisi ise karmaşık algoritmaların gerçekleştirim ve sınanması için yüksek hızlı bilgisayarların kullanıma sunulmasıdır.

2.2.1 Çözüm Yöntemleri

Özel amaçlı ve genel amaçlı olmak üzere temel iki tür çarpanlara ayırma algoritması bulunmaktadır. Özel amaçlı çarpanlara ayırma algoritmaları, çarpanlara ayrılan *n* sayısının belirli özelliklerinden faydalanmaktadır. Bunun yanında genel amaçlı çarpanlara ayırma algoritmaları sadece çarpanlarına ayrılan *n* sayısının boyuna bağlıdır.

Özel amaçlı çarpanlarına ayırma algoritmalarının en kuvvetli olanlarından birisi, 1985 yılında Herndrik Lenstra (1985) tarafından ortaya atılan, eliptik eğri çarpanlarına ayırma yöntemidir. Bu yöntemin karmaşıklığı *n* sayısının asal çarpanlarının büyüklüğüne bağlı olduğundan, önce küçük çarpanları bulmaya yönelir. 1998 yılında başlatılan ECMNET projesi [2] 50 basamaklı (166 ikil) asal çarpanlara başarıyla ulaşmıştır. Eliptik eğri çarpanlara ayırma yöntemini kullanılarak 180 basamaklı bileşik sayı, Bruce Dodson tarafından 2005 Nisan ayında 66 ve 114 basamaklı asal çarpanlarına ayrılmıştır [2].

RSA şifreleme sistemi ortaya atılıncaya kadar bilinen en iyi genel amaçlı çarpanlarına ayırma yöntemi, en çok 40 basamaklı (133 ikil) sayıları çarpanlarına ayırabilen sürekli çarpan algoritması (Morrison ve Brillhart 1975) idi. Bu algoritma asal sayılardan oluşan çarpan tabanı kullanımı ve çözüldüğünde çarpanlara ayırmayı tamamlayan, bu tabanla eşlenmiş doğrusal denklemler üretme fikrine dayanmaktadır. Günümüzde kullanılan en iyi genel amaçlı algoritmalar olan ikinci dereceden denklem eleği ve sayı alanı eleği algoritmalarının dayandığı fikir de aynıdır. Her iki algoritma da iş istasyonları arasında paralel işleyerek çarpanlara ayırmaya olanak sağlamakta ve süper bilgisayar ihtiyacını ortadan kaldırmaktadır.

İkinci dereceden denklem eleği (Pomerance 1985) Carl Pomerance tarafından 1984 yılında geliştirilmiştir. Başlangıçta 70 basamaklı (233 ikil) sayıların çarpanlarına ayrılmasında kullanılmaktayken, Arjen Lenstra (Atkins 1995) liderliğindeki bir grup tarafından 129 basamaklı (429 ikil) RSA yarışmasının çözümü için kullanılmıştır. Çarpanlarına ayırma

işlemi dünyanın değişik yerlerine dağılmış 1600 bilgisayar tarafından 8 ayda tamamlanmıştır. Çarpanlarına ayırma işleminin 5000 MIPS yıl gerektirdiği hesaplanmaktadır.

Sayı alanı eleği (Lenstra 1993) 1989 yılında ilk geliştirildiğinde, belirli özellikleri taşıyan sayılar üzerinde en iyi sonucu vermekteydi ve 155 basamaklı (513 ikil) ($2^{512} + 1$) sayısının çarpanlarına ayrılmasında kullanılmıştı. Daha sonra genel bir çarpanlarına ayrıma algoritması olarak genişletilmiştir (Buhler 1993). Başlangıçta ikici dereceden denklem eleği algoritmasından daha yavaş olduğu değerlendirilmekteyken, 120 basamaktan (400 ikil) büyük sayılar için çok daha etken olarak işlediği ortaya konmuştur. 129 basamaklı RSA yarışması çözümü için hesaplanan 5000 MIPS yıl işlem gücünün %15i ile çözümün elde edilebileceği hesaplanmış, 155 basamak içinse bunun sadece 5 katı işlem gücünün yeterli olacağı iddia edilmiştir. Daha sonra 1999 yılında 155 basamaklı RSA yarışma sayısının toplam 8000 MIPS yıl işlem gücü ile 3,7 ayda çözüldüğü duyurulmuştur. Çizelge 2.1'de tamsayı çarpanlarına ayırma ile ilgili gelişmeleri gösteren tarihsel veriler gösterilmiştir.

Çizelge 2.1 Tamsayı çarpanlarına ayırma probleminin geçmişi

Yıl	Basamak Sayısı	İkil Sayısı	MIPS Yıl
1984	71	236	0,1
1988	106	352	140
1993	120	399	825
1994	129	429	5000
1995	119	395	250
1996	130	432	750
1999	140	466	2000
1999	155	512	8000
2001	160	530	--
2003	174	576	--
2005	200	663	--

Çizelgede boş olan hücreler için MIPS değerleri hesaplanmamış ya da hesaplanamamıştır [3]. Çizelge incelendiğinde 512 ikil bileşik sayı, *n*, kullanan RSA şifreleme ile sadece belirli bir düzeye kadar güvenlik sağlanabilmekte olduğu görülmektedir. Yeterli güvenliği sağlamak için 1024 ikil ya da daha fazlasına ihtiyaç duyulmaktadır.

2.3 Ayrık Logaritma Problemi (ALP)

Aşağıda ALP tanımı ve çözüm yaklaşımları açıklanmıştır.

2.3.1 Tanım

p asal sayı ise, Z_p, toplama ve çarpmanın modülo *p* gerçekleştirildiği {0, 1, 2, ... , *p*-1} tamsayılar kümesidir. Z_p'nin üreteci olarak bilinen ve Z_p'nin tüm elemanlarını üssel değerleri ile veren bir $\alpha \varepsilon Zp$ değeri bulunmaktadır. Bir asal sayı, *p*; Z_p'nin üreteci α ve sıfırdan farklı bir $\beta \varepsilon Zp$ verilmişken $\alpha^l \equiv \beta \pmod p$ denkleminin biricik çözümü olan l , $0 \leq l \leq p-1$, değerinin bulunması ALP olarak adlandırılır.

Bu problemin zorluğuna dayanarak, Diffie-Hellman anahtar değişimi yöntemi (Diffie ve Hellman 1976) ortaya çıkmıştır. Ardından ALP kullanarak SİA (NIST 1994), ElGamal Şifreleme ve İmza Yöntemi (ElGamal 1985), Schnorr imza yöntemi (Schnorr 1991) ve Nyberg-Ruepel (Nyberg 1996) imza yöntemi başta olmak üzere pek çok yöntem ortaya atılmıştır. Bu uygulamalara duyulan ilgi nedeniyle son 20 yıldır matematikçiler tarafından yaygın olarak incelenmiştir.

2.3.2 Çözüm Yöntemleri

Tamsayı çarpanlara ayırma probleminde olduğu gibi ayrık logaritma probleminin çözümü için de iki tür algoritma bulunmaktadır. Özel amaçlı algoritmalar, *p* asal sayısının belirli özelliklerinden faydalanmaktadır. Bunun yanında genel amaçlı algoritmalar sadece *p* asal sayısının boyuna bağlıdır.

Bilinen en hızlı genel ayrık logaritma problemi çözümü algoritması dizin hesabı (index calculus) (Schirokauer vd. 1996) olarak adlandırılmaktadır. Bu yöntemde, küçük asal sayılar bir veri tabanında saklanmakta ve karşılığı olan logaritma değerleri oluşturulmakta, bu şekilde gerekli diğer alan elemanları kolayca elde edilebilmektedir. Bu tamsayı çarpanlara ayırma için kullanılan çarpan tabanı yöntemlerinin benzeridir. Bu nedenle TÇAP ya da ALP çözüm yöntemlerinden birisi için bir iyileştirme söz konusu olduğunda kısa bir süre sonra diğer problem içinde benzer çözüm iyileştirmeleri beklenebilir. Çarpanlara ayırma yöntemlerinde olduğu gibi, dizin hesabı yöntemi de kolaylıkla paralel olarak işletilebilir.

Çarpanlarına ayırmada olduğu gibi, ALP için de bilinen en iyi çözüm algoritması sayı alanı eleğidir (Gordon 1993, Rabin 1979). TÇAP'de karşılık gelen algoritma ile aynı karmaşıklığa sahiptir. Kabaca, *k*-ikil asal modülo *p* için logaritma bulmak ile *k*-ikil bileşik sayının asal çarpanlarına ayrılması aynı derecede zordur denilebilir.

ALP algoritmalarının gerçekleştirimi, tamsayı çarpanlarına ayırma algoritmalarına nazaran geride kalmıştır. 1990 yılında Brian LaMAcchia ve Andrew Odlzko (1991) Gauss tamsayı yöntemi adı verilen dizin hesabı yönteminin bir yöntem kullanarak 191 ikil modülo ayrık logaritma hesaplamayı başarmışlardır.

Weber ve Denny (1998), 129 basamak modülo ayrık logaritma probleminden oluşan McCurley Diffie-Hellman yarışmasına çözüm elde ettiklerini duyurdular. Kullanılan yöntem, asal sayı alanı eleğine uygun özellikler taşıyordu. Crypto '98'de yayımlanan sonuçlar, ayrık logaritma problemi altyapısına dayanan şifreleme yöntemleri için 1024 ikil ya da daha büyük modülo *p* kullanmak gerektiğini ortaya koymaktadır.

2.4 Eliptik Eğri Ayrık Logaritma Problemi (EEALP)

Matematiksel işlemlerin tamsayı alanlarından eğri üzerindeki noktalara taşınmasıyla ortaya çıkan EALP, klasik ALP'den çok daha zor gözükmektedir. Aşağıda bu problemin açıklaması ve çözümü için gerçekleştirilenler açıklanmaktadır.

2.4.1 Tanım

q asal üstel sayı ise, F_q, *q* elaman içeren sonlu alanı ifade eder. Uygulamalarda *q*, 2'nin kuvveti (2^m) ya da asal sayı olarak farklı biçimlerde kullanılabilmektedir. F_q, alanında tanımlanmış bir E eliptik eğrisi, ile bu eğri üzerindeki P ve Q noktaları verilmişken, E eliptik eğrisinin düzeyi olmak üzere, $Q = lP$ denkleminin biricik çözümü olan *l*, $0 \leq l \leq n-1$, değerinin bulunması EEALP olarak adlandırılır. Bu problemin çözümünün zorluğuna dayanarak 1985 yılında Neal Koblitz (Kolbitz 86) ve Victor Miller (Miller 87) bir birinden bağımsız olarak eliptik eğri üzerindeki noktaları kullanarak farklı ayrık logaritma sistemlerinin gerçeklenebileceğini göstermişlerdir.

2.4.2 Çözüm Yöntemleri

Sıklıkla TÇAP'nin EEALP'den çok daha kolay anlaşılır olduğu iddia edilmektedir. Bu genel anlamıyla geçerli olduğu halde, yanlış bir şekilde TÇAP üzerinde yüz yıllardır çalışıldığı inancını doğurmaktadır. Yukarıda açıklandığı gibi, TÇAP ile ilgili ciddi gelişmeler 1985 yılından sonra yaşanmıştır ki; bu yıllarda EEALP üzerinde de benzer çalışmalar yapılmaktaydı. Bu nedenle "EEALP'nin çözümünün TÇAP çözümünden daha zor

gözükmesinin nedeni, üzerinde yeterli çalışmanın yapılmamış olmasıdır" yaklaşımı tamamen yanlıştır.

Yukarıda açıklanan matematiksel problemlerde olduğu gibi EEALP'de de genel ve özel çözüm yöntemleri bulunmaktadır. Özel yöntemler; eğri üzerindeki noktaların eşdeğerlik kümelerine bölünebildiği durumlardan faydalanan eşdeğerlik kümeleri yöntemi (Gallant 2000 Wiener 1999); süper tekil olarak adlandırılan, *q* asal üssel sayısı kadar noktaya sahip olan ve istisna eğriler olarak adlandırılan eğrilerin özelliklerinden faydalanan istisna eğriler yöntemi (Semaev 1998); *q* asal sayısının üssel değerlerinin uzun süre 1 değeri vermediği, süper tekil olarak adlandırılan eğrilerin özelliklerinden faydalanan MOV yöntemi (Menesez vd. 1993) sayılabilir. Bu tür eğriler oldukça seyrek (örneğin, seçilen bir eğrinin istisna eğrilerden olması ihtimali 1/4√q dur) ve seçilen eğrilerin bu sınıflara girip girmediğinin tespit edilmesi olanaklı olduğundan, eliptik eğri ayrık logaritma probleminin çözümü için çok fazla yardımcı olamamaktadır. Genel yöntemler aşağıda açıklanmıştır.

2.4.2.1 Kaba Çözüm Yaklaşımı

ECDLP çözümü için en temel yöntem üretici Q'nun katlarının hesaplanması ve bunların bir tabloda tutulmasıdır. Her bir işlemin eliptik eğri toplama olduğu akılda tutulmak üzere, bu yöntem tablonun oluşturulması için O(*n*) süre ve tablonun tutulması için O(*n*) alan gerektirir. Bu en kötü durum sınırlamasının üzerine aşağıdaki yöntemlerle oldukça ilerleme sağlanmıştır.

2.4.2.2 Pohlig-Hellman Yöntemi

Bu yöntem (Pohlig 1978), taban noktasının düzeyi olan *n*'nin çarpanlara ayrılmasından faydalanmaktadır. Yöntem, *l* değerinin bulunmasını *l* mod (*n*'nin asal çarpanlarının her birisi) düzeyine indirmektedir. Her bir çarpan için hesaplama yapıldıktan sonra "chinese remainder" teoremi ile *l* değeri hesaplanabilir. Bu yöntem, düzeyi çok büyük asal çarpanlara sahip bir eliptik eğri için işlevselliğini yitirmektedir.

2.4.2.3 Shank Yöntemi (Bebek Adımı, Dev Adımı)

Daniel Shanks tarafından ortaya konmuş olan yöntem (Shanks 1971) karmaşıklığı $O(n^{1/2}\log n)$ ve gereken alanı O(√n) düzeylerine indirmektedir. Bu yöntem *p* asal sayısının tam değerini değil, yaklaşık değerlerini kullanmaktadır. Bu algoritmanın eliptik eğri uyumlaması şu şekildedir: Q'nun *n* düzeyinde bir üretici olduğunu varsayalım, A noktası verildiğinde *k*Q = A olacak şekilde *k* değerini bulmak istiyoruz. m = √n olsun. (m - 1) ≥ *j* ≥ 0 için L1 listesinde (*j*, *jm*Q) çiftlerini tutulur. Bu liste çiftlerin ikinci elemanlarına göre sıralanır. (m - 1) ≥ *i* ≥ 0 (m - 1) için (*i*, -*i*Q + A) çiftlerinden oluşan ikinci bir liste, L2 yaratılır ve bu liste de ikinci

elemanlarına göre sıralanır. Listeler (j, P) ∈ L1 ve (i, P) ∈ L2 bulununcaya kadar taranır. Bu durumda:

- p = 3 mod 4 ise ($p = 4u + 3$ gibi bir değere sahipse); (2.1)
  - $y = a^{u+1}$ mod p
  - eğer $y^2 = a$ ise, sonuç bulunmuştur, aksi takdirde y değeri yoktur.

- p = 5 mod 8 ise ($p = 8u + 5$ gibi bir değere sahipse); (2.2)
  - $y = a \cdot (2a)^u \cdot [(2a)^{2u+1} - 1]$ mod p
  - eğer $y^2 = a$ ise, sonuç bulunmuştur, aksi taktirde y değeri yoktur.

- p = 1 mod 4 ise ($p = 4u + 1$ gibi bir değere sahipse); (2.3)
  - Q = g; 0 ≤ P ≤ p olmak üzere rasgele bir değer alınır.
  - Lucas Sıra Elamanları bulunur
    - $U_0 = 0$, $U_1 = 1$, … $U_k = PU_{k-1} - QU_{k-1}$ ($2 \le k$)
    - $V_0 = 0$, $V_1 = P$, … $V_k = PV_{k-1} - QV_{k-1}$ ($2 \le k$)
  - $U = U_{2u+1}$, $V = V_{2u+1}$ mod p elemanları hesaplanır.
  - $V^2 = 4Q$ ise $y = V/2$ mod p olur.
  - U ≠ 1 ise y değeri yoktur.
  - Başka bir P değeri ile işlemler baştan tekrarlanır.

$$P = jmQ$$
$$P = -iQ + A$$
$$jmQ = -iQ + A$$
$$(jm + i)\,Q = A$$

olur ve $k = (jm + i)$ olarak bulunmuştur.

2.4.2.4 Pollard'ın ρ (ro) Yöntemi

Pollard'ın ρ yöntemi (Pollard 1978), EEALP çözümü için bilinen en iyi yöntemdir. Karmaşıklığı O(√n), gerektirdiği bellek alanı ise 6 değerin tutulmasına kadar indirgenmiştir.

A, katsayısı bilinmeyen nokta; Q, n seviyesinde eğrinin üreteci olsun. A=xQ olduğuna inanıyor ve x'i arıyorsak;

$a_0, q_0 \in Z_n$ seçilir. $x_0 = q_0 + a_0$A atanır. $\{x_i\}$, $\{a_i\}$ ve $\{q_i\}$ aşağıdaki şekilde oluşturulur:

$$x_i = \begin{cases} Q + x_{i-1} & x_{i-1} \in S_0 \text{ ise} \\ 2x_{i-1} & x_{i-1} \in S_1 \text{ ise} \\ A + x_{i-1} & x_{i-1} \in S_2 \text{ ise} \end{cases} \quad (2.5)$$

$$a_i = \begin{cases} a_{i-1} & x_{i-1} \in S_0 \text{ ise} \\ 2a_{i-1} & x_{i-1} \in S_1 \text{ ise} \\ 1 + a_{i-1} & x_{i-1} \in S_2 \text{ ise} \end{cases} \quad (2.6)$$

$$q_i = \begin{cases} 1 + q_{i-1} & x_{i-1} \in S_0 \text{ ise} \\ 2q_{i-1} & x_{i-1} \in S_1 \text{ ise} \\ q_{i-1} & x_{i-1} \in S_2 \text{ ise} \end{cases} \quad (2.7)$$

$x_i = x_j$ olduğunda:

$$a_iA + q_iQ = a_jA + q_jQ$$
$$(a_i - a_j)A = (q_j - q_i)Q$$
$$A = [(q_j - q_i)/(a_i - a_j)]Q$$

$(a_i - a_j)$ tersi alınabilir bir değer ise sonuç bulunmuştur. Bellek yönünden $x_i = x_j$ bulmak için en etken işleyiş $x_i = x_{2i}$ bulmaya çalışmaktır. Bu şekilde ilk $x_i = x_j$ bulunamayabilir, ancak sadece 6 değerin, x_i, x_{2i}, a_i, a_{2i}, q_i ve q_{2i}, tutulması yeterli olmaktadır. Son çalışmalar (Gallant 2000; Wiener 1999) bu yöntemin $\sqrt{2}$ kat hızlandırılabileceğini göstermiştir. Bu durumda geliştirilmiş Pollard'ın ρ yöntemi için karmaşıklık $O(\sqrt{(\pi n)/2})$ olmaktadır.

Odlyzko'nun (1995) hesaplamalarına göre dünyadaki tüm bilgisayarların binde biri birlikte çalışacak olursa, 2014 yılında 1010 ila 1011 MIPS yıl toplam işlem gücüne erişecektir. Aşağıdaki çizelgede (Çizelge 2.2) Pollard'ın ρ yöntemi kullanılarak EEALP çözümü için öngörülen MIPS tahminleri verilmiştir[2].

Çizelge 2.2 Pollard'ın ρ yöntemi kullanılarak EEALP çözümü için gereken işlem gücü.

q (ikil)	MIPS yıl
160	$8{,}5 \times 10^{11}$
186	$7{,}0 \times 10^{15}$
234	$1{,}2 \times 10^{23}$
354	$1{,}3 \times 10^{41}$
426	$9{,}2 \times 10^{51}$

3. ELİPTİK EĞRİ ŞİFRELEME

Eliptik eğriler üzerinde yüzyılı aşkın bir süredir çalışılmış olmasına rağmen, şifreleme alanında kullanımı sadece 20 yıllık bir geçmişe sahiptir. 1985 yılında Neal Koblitz (1987a) ve Victor Miller (1986) ilk defa eliptik eğrilerin şifreleme alanında kullanılabileceğini öngörmüşlerdir. Her ikisi de yeni bir şifreleme algoritması ortaya atmak yerine, eliptik eğrilerin mevcut açık anahtar şifreleme algoritmalarıyla kullanılabileceğini göstermişlerdir.

Eliptik eğrilerin şifreleme alanında kullanılabilmesi, grup oluşturmak üzere 'elemanlar' ve 'birleştirme kuralları' tanımlamayı sağlayan yöntemlere sahip olması sayesindedir. Bu gruplar, şifreleme için kullanılmasına olanak sağlayacak özelliklere sahipken şifre çözmeyi kolaylaştırıcı belirli özelliklere sahip değildir.

Bu bölümde, eliptik eğriler açıklanacak, ardından eliptik eğrilerin şifreleme alanında nasıl kullanıldığı anlatılacaktır. Bu amaçla öncelikle eliptik eğrilerin temelini oluşturan matematiksel altyapı kısaca özetlenecektir.

3.1 Matematiksel Altyapı

EEŞ, eliptik eğri üzerindeki noktalar üzerinde işlem yapmasına rağmen, eğri tanımı ve noktalar ile ilgili işlemleri gerçekleştirebilmek için sonlu alanlarda tanım ve işlemlere ihtiyaç duyulduğundan, öncelikle sonlu alanların tanıtılması ile gerekmektedir.

3.1.1 Sonlu Alanlar

Bir sonlu alan, 'eleman'ları olarak adlandırılan sonlu eleman nesneleri kümesi ile bu alan eleman çiftleri üzerinde tanımlı iki işlemden (toplama ve çarpma) oluşur. Bu işlemlerin belirli özellikleri taşıması gerekir (Hankerson vd. 2004).

q elemana sahip bir sonlu alan için q'nun bir asal sayının üssü (kuvveti) olması gerekir ve böyle her bir q için tek bir sonlu alan bulunmaktadır. q elemana sahip bir alan F_q ile gösterilmektedir.

Eliptik eğrilerde iki tür sonlu alan F_q kullanılmaktadır; asal sonlu alan olarak adlandırılan F_p ($q=p$ ve p asal bir sayı olmak üzere) ve karakteristik 2 olarak adlandırılan F_{2^m} ($q=2^m$ ve $m \geq 1$) (Schneier 1996).

Eliptik eğrilere dayanan şifreleme yöntemleri için bu alanların tanımını sağlam bir şekilde oturtmak gerektiğinden aşağıdaki kısımlarda bu alanların tanımı yapılmıştır:

3.1.1.1 Sonlu Alan Fp

Sonlu Alan F_p; p eleman içeren asal sonlu alandır. Her bir asal p için tek bir asal alan F_p olduğu halde, sonlu asal alanının elemanlarını ifade etmek için birden çok yol bulunmaktadır. Burada elemanlarının tamsayılar kümesi olarak gösterilmesi gerekmektedir:

$$\{0, 1, 2, ..., p\text{-}1\}$$

Toplama ve çarpma işlemleri de aşağıdaki şekilde tanımlanmıştır:

- Toplama: Eğer $a, b \in F_p$ ise, $a + b = r$ F_p alanındadır.

  ($r \in [0, p\text{-}1]$, $a + b$ tamsayısının p ile bölünmesinden kalandır). Bu işlem mod p toplama olarak bilinir ve

  $$a + b \equiv r \pmod{p} \text{ olarak yazılır.} \qquad (3.1)$$

- Çarpma: Eğer $a, b \in F_p$ ise, $a \,.\, b = s$ F_p alanındadır.

  ($s \in [0, p\text{-}1]$, $a \,.\, b$ tamsayısının p ile bölünmesinden kalandır). Bu işlem mod p çarpma olarak bilinir ve

  $$a \,.\, b \equiv s \pmod{p} \text{ olarak yazılır} \qquad (3.2)$$

F_p alanında toplama ve çarpma bilinen tamsayı aritmetiğinde kullanılan algoritmalar vasıtasıyla etkin bir şekilde gerçekleştirilebilir. Bu F_p gösteriminde, toplamada etkisiz eleman, tamsayı 0 (sıfır)dır ve çarpma etkisiz elemanı tamsayı 1 (bir)dir. Normal tamsayı aritmetiğinde olduğu gibi alan elemanları için çıkartma ve bölme işlemlerinin de tanımlanması yerinde olacaktır. Bunun için, alanının toplamaya göre ve çarpmaya göre ters elamanlarını tanımlamak gereklidir:

- Toplamaya göre ters eleman: $a \in F_p$ ise, a'nın toplamaya göre tersi ($-a$) F_p alanın elemanıdır ve $a + x \equiv 0 \pmod{p}$ denkleminin biricik çözümüdür.
- Çarpmaya göre ters eleman: $a \in F_{p,}$ $a \neq 0$ ise, a'nın çarpmaya göre tersi (a^{-1}) F_p alanın elemanıdır ve $a \,.\, x \equiv 1 \pmod{p}$ denkleminin biricik çözümüdür.

F_p alanındaki toplamaya ve çarpmaya göre ters elamanlar kolaylıkla hesaplanabilir. Çarpmaya göre ters elemanlar genişletilmiş Öklid algoritmasına göre hesaplanmaktadır. Bölme ve çıkartma çarpmaya ve toplamaya göre ters elemanlar kullanılarak tanımlanmaktadır:

$$a - b \bmod p \rightarrow a + (-b) \bmod p \text{ ve} \qquad (3.3)$$

$$a / b \bmod p \;\rightarrow\; a \,.\, b^{-1} \bmod p. \tag{3.4}$$

Hesaplama ve iletişim kolaylığı sağlayabilmek için, p kelime genişliğine bağlı olduğundan, talep edilen güvenlik düzeylerini de karşılayabilmek üzere, burada kullanılacak asal alan F_p aşağıdaki şekilde sınırlandırılmalıdır (Menezes 1993):

$$\lceil \mathrm{Log}_2 p \rceil \;\in \{112,128,160,192,224,256,384,521\}$$

$\lceil \mathrm{Log}_2 p \rceil = 512$ yerine $\lceil \mathrm{Log}_2 p \rceil = 521$ kullanılmasının nedeni, başta (NIST 1998) olmak üzere tavsiye edilen EEŞ parametrelerine uygunluk sağlamaktır.

3.1.1.2 Sonlu Alan F_{2^m}

Sonlu Alan F_{2^m}, 2^m eleman içeren karakteristik 2 sonlu alandır. $m > 1$ olmak üzere ikinin kuvveti olan her bir 2^m için tek bir karakteristik 2 F_{2^m} bulunduğu halde, karakteristik 2 alanının elemanlarını ifade etmek için birden çok yol bulunmaktadır (Husemoller 2004). Burada elemanların m-1 ya da daha düşük dereceli ikili polinomlar kümesi olarak ifade edilmesi anlaşılırlığı kolaylaştırması açısından uygun olmaktadır.

$$\{a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_1x + a_0 : a_i \in \{0,1\}\}$$

Toplama ve çarpma işlemleri de m derecesinde indirgenemez polinom $f(x)$ terimleriyle aşağıdaki şekilde tanımlanmıştır:

- Toplama:

  Eğer

  $$a = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_1x + a_0,$$

  $$b = b_{m-1}x^{m-1} + b_{m-2}x^{m-2} + ... + b_1x + b_0 \in F_{2^m} \text{ ise,}$$

  $$r = r_{m-1}x^{m-1} + r_{m-2}x^{m-2} + ... + r_1x + r_0 \text{ olmak üzere,}$$

  $$a + b = r \;\; (r_i = a_i + b_i \,(\mathrm{mod}\, 2)) \;\; F_{2^m} \text{ alanındadır.} \tag{3.5}$$

- Çarpma:

  Eğer

  $$a = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_1x + a_0,$$

  $$b = b_{m-1}x^{m-1} + b_{m-2}x^{m-2} + ... + b_1x + b_0 \in F_{2^m} \text{ ise,}$$

  $$s = s_{m-1}x^{m-1} + s_{m-2}x^{m-2} + ... + s_1x + s_0 \text{ olmak üzere,}$$

  $$a \,.\, b = s, \; F_{2^m} \text{ alanındadır.} \tag{3.6}$$

s, aynı zamanda $a \cdot b$ polinomunun, tüm katsayılar modülo 2 işlenmek üzere, bölümünden kalandır.

F_{2^m} alanında toplama ve çarpma bilinen tamsayı ve polinom aritmetiğinde kullanılan algoritmalar kullanılarak etkin bir şekilde gerçekleştirilebilir. Bu F_{2^m} gösteriminde, toplamada etkisiz eleman ya da sıfır, 0 polinomu, çarpma etkisiz elemanı 1 polinomudur. Normal tamsayı aritmetiğinde olduğu gibi alan elemanları için çıkartma ve bölme işlemlerinin de tanımlanması yerinde olacaktır. Bunun için, alanının toplamaya göre ve çarpmaya göre ters elamanlarını tanımlamak gereklidir:

- Toplamaya göre ters eleman:

  $a \in F_{2^m}$ ise, *a*'nın toplamaya göre tersi ($-a$) F_{2^m} alanın elemanıdır ve

  $a + x = 0$ denkleminin biricik çözümüdür. (3.7)

- Çarpmaya göre ters eleman:

  $a \in F_{2^m}$, $a \neq 0$ ise, *a*'nın çarpmaya göre tersi (a^{-1}) F_{2^m} alanın elemanıdır ve

  $a \cdot x = 1$ denkleminin biricik çözümüdür. (3.8)

F_{2^m} alanındaki toplamaya ve çarpmaya göre ters elamanlar genişletilmiş Öklid algoritması ile kolaylıkla hesaplanabilir. Bölme ve çıkartma, çarpmaya ve toplamaya göre ters elemanlar kullanılarak tanımlanmaktadır:

$a - b = a + (-b)$ ve (3.9)

$a / b = a \cdot b^{-1}$ dir. (3.10)

F_p asal alanında bahsedilen nedenlerle, burada da kullanılabilecek karakteristik 2 F_{2^m} alanı için aşağıdaki kısıtlamalar söz konusudur:

$m \in \{113,131, 163, 193, 233, 239, 283, 409, 571\}$ (3.11)

ve F_{2^m} alanında toplama ve çarpma aşağıda (Çizelge 3.1) verilen indirgenemez polinomlar kullanılarak yapılmalıdır.

Çizelge 3.1 F_2m Alanları İndirgeme polinomları (NIST 1999)

Alan	İndirgeme Polinomları
F_2^{113}	$f(x) = x^{113} + x^9 + 1$
F_2^{131}	$f(x) = x^{131} + x^8 + x^3 + x^2 + 1$
F_2^{163}	$f(x) = x^{163} + x^7 + x^6 + x^3 + 1$
F_2^{193}	$f(x) = x^{193} + x^{15} + 1$
F_2^{233}	$f(x) = x^{233} + x^{74} + 1$
F_2^{239}	$f(x) = x^{239} + x^{36} + 1$ ya da $f(x) = x^{239} + x^{158} + 1$
F_2^{283}	$f(x) = x^{283} + x^{12} + x^7 + x^5 + 1$
F_2^{409}	$f(x) = x^{409} + x^{87} + 1$
F_2^{571}	$f(x) = x^{571} + x^{10} + x^5 + x^2 + 1$

Kabul edilebilir *m* değerlerini seçmek için kullanılan kural; yukarıdaki kümede (3.11) verilen aralıklarda, bulunabiliyorsa seviyesi F_2^m üzerinde bir asal sayının 2 ya da 4 katı olan bir Koblitz eğrisi (Koblitz eğrisi, $a, b \in (0,1)$ olan, F_2^m üzerinde bir eliptik eğridir), bulunamıyorsa, en küçük asal sayının alınmasıdır. Burada $m = 239$ değerinin dahil edilmesinin nedeni, mevcut çalışmalarda 239 değerinin yaygın olarak kullanılmasıdır.

Kullanılan $f(x)$ polinomları kuralı ise; *m* derecesinde üç elemanlı, indirgenemez ikili polinomu bulunabiliyorsa ($f(x) = x^m + x^k + 1$; $m > k > 1$), olabildiğince küçük k değeri ile bu polinomu kullan; bulunamıyorsa, *m* derecesinde beş elemanlı, indirgenemez ikili polinomu ($x^m + x^{k_3} + x^{k_2} + x^{k_1} + 1$; $k_3 > k_2 > k_1 > 1$) olabildiğince küçük k_3 değeri (k_3 verilmişken olabildiğince küçük k_2 ve k_2 verilmişken olabildiğince küçük k_1 değeri) ile kullan.

3.1.2 Eliptik Eğriler

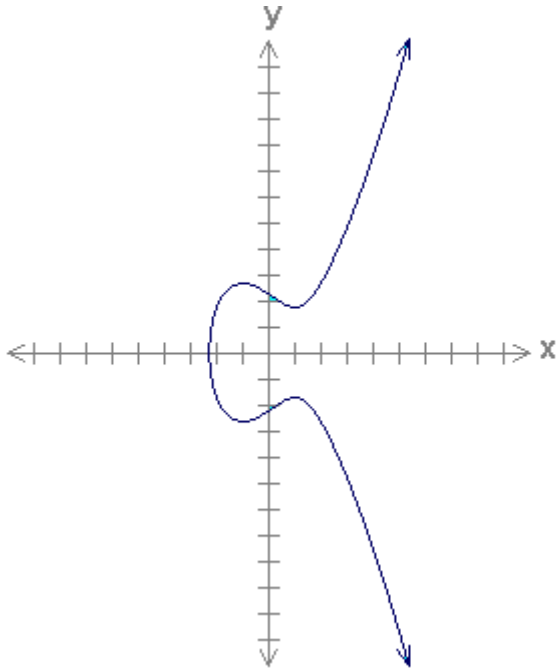
Gerçek sayılar üzerinde bir eliptik eğri tanımı;

"$y^2 + axy + by = x^3 + cx^2 + dx + e$ eliptik eğri eşitliğini sağlayan (x,y) noktalarının kümesi"

olarak yapılabilir. *a, b, c, d, e* katsayılarının farklı değerleri, farklı eliptik eğriler üretmektedir.

Eliptik eğri denkleminin bu biçimde gösterilmesi “uzun Weierstrass biçimi” (Menezes 1993) olarak adlandırılmaktadır.

Örneğin, *a, b, d* = 0, *c* = -3, *e* = 5 alınırsa, denklem $y^2 = x^3 - 3x + 5$ biçimini alır. Bu denklemin ifade ettiği eğrinin çizimi aşağıda (Şekil 3.1) verilmiştir.



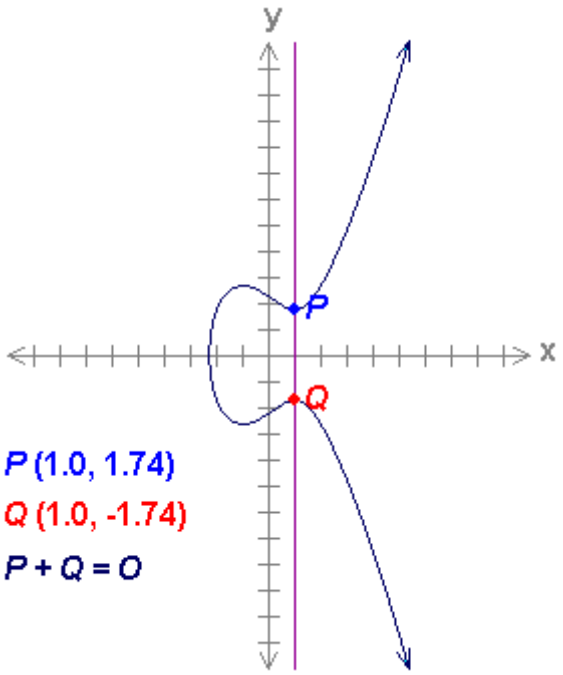
Şekil 3.1 $y^2 = x^3 - 3x + 5$ Eliptik Eğrisi

Gerçek sayılar üzerinde bir eliptik eğri, karşılık gelen eğri üzerindeki noktalar ve sonsuzdaki nokta olarak adlandırılan *O* noktasının birleşiminden oluşur. Bu noktaların oluşturduğu küme, iki noktanın toplamı yine aynı eğri üzerinde bir nokta verecek şekilde tanımlanırsa bir grup oluşturulmuş olur.

Eliptik eğri grupları temel işlemi toplama olan gruplardır. Eliptik eğrilerde iki noktanın toplamı geometrik olarak tanımlanmıştır: Eliptik eğrilerde iki farklı nokta toplanırken, iki noktadan geçen bir doğru çizilir, bu doğrunun eğriyi kestiği noktanın x ekseninde yansıtılması, iki noktanın toplamını verir. Bir birine göre tersi olan iki noktanın toplamını bulmak için çizilen doğru, eğriyi hiçbir noktada kesmeyecektir, bu nedenle eliptik eğri grupları O noktasını içermektedir. Tanım olarak P, eliptik eğri üzerinde bir nokta olmak üzere

“P + (-P) = *O*”dur ve *O* eliptik eğri grubunun toplama için etkisiz elemanıdır. (3.12)

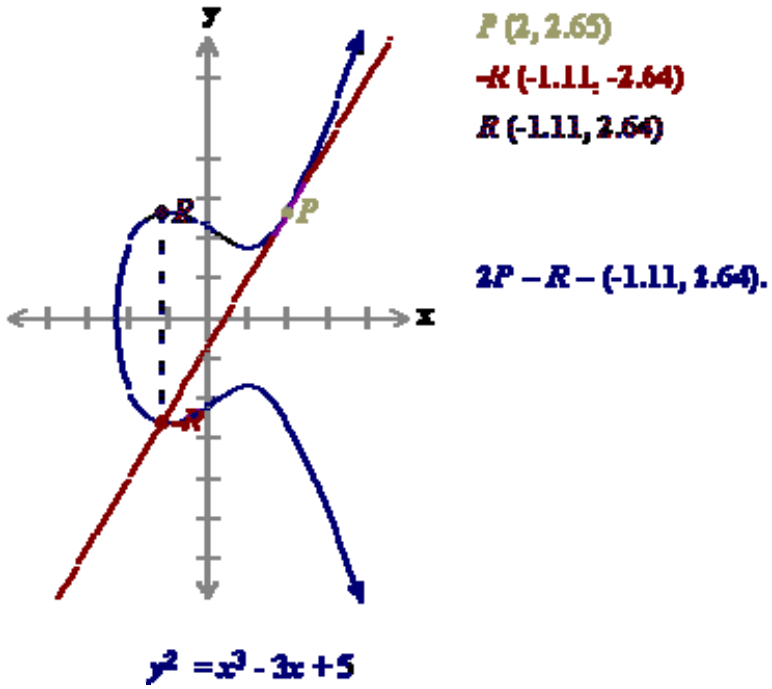
Bir birine göre tersi olan iki noktanın toplanması Şekil 3.2‘de gösterilmiştir.



Şekil 3.2 $y^2 = x^3 - 3x + 5$ Eliptik Eğrisi üzerinde P+ (-P) Toplamı

P + P ise, P noktasından çizilen teğet çizgisinin eğriyi kestiği noktanın x eksenine göre tersinin alınması işlemiyle bulunur.

Örneğin: $y^2 = x^3 - 3x + 5$ eliptik eğrisi üzerinde P(2, 2.65) noktasının grafik olarak kendisi ile toplanması aşağıdaki şekilde (Şekil 3.3) gösterilmiştir:



Şekil 3.3 $y^2 = x^3 - 3x + 5$ Eliptik Eğrisi Üzerinde Noktanın İki Katının Bulunması

Bu geometrik işlemlerin aritmetik karşılıkları şu şekilde verilebilir:

$P = (x_1, y_1)$

$Q = (x_2, y_2)$

$R = (x_3, y_3)$ ve

$P \neq Q$

$P \neq -Q$ olmak üzere, $y^3 = x^2 + ax + b$ için

- $O + O = O$ (3.13)
- $P + O = O + P = P$ (3.14)
- $P + (-P) = O$ (3.15)
- P + Q = R ise;

$$x_3 = (y_2 - y_1 / x_2 - x_1)^2 - x_1 - x_2 \quad (3.16)$$

$$y_3 = (y_2 - y_1 / x_2 - x_1)^2 * (x_1 - x_3) - y_1 \quad (3.17)$$

- P + P = R ise;

$$x_3 = ((3x_1^2 + a)/2y_1)^2 - 2x_1 \quad (3.18)$$

$$y_3 = ((3x_1^2 + a)/2y_1) * (x_1 - x_3) - y_1 \text{ olur.} \quad (3.19)$$

Eliptik eğrilerde kullanılan önemli işlemlerden birisi de skaler çarpımdır. Skaler çarpım ile, verilen bir noktanın belirli bir değer ile çarpımı sonucunda eğri üzerinde yer alan başka bir nokta elde edilmektedir. Skaler çarpım için kullanılan yöntem verilen noktanın skaler kadar kendisiyle toplanmasından ibarettir.

3.1.2.1 Fp Sonlu Alanında Eliptik Eğriler

$y^2 + axy + by = x^3 + cx^2 + dx + e$ eğrisinde y yerine $y-(ax+b)/2$ yazılacak olursa denklem :

$$[y-(ax+b)/2]^2 + ax[y-(ax+b)/2] + b[y-(ax+b)/2] = x^3 + cx^2 + dx + e \quad (3.20)$$

$$\Rightarrow \quad y^2 - axy - by + a^2x^2/4 + abx/2 + b^2/4 + axy - a^2x^2/2 + by - b^2/2 = x^3 + cx^2 + dx + e$$

$$\Rightarrow \quad y^2 - a^2x^2/4 + abx/2 - b^2/4 = x^3 + cx^2 + dx + e \text{ biçimini alır.} \quad (3.22)$$

xy ve y terimleri denklemden kaybolduğuna göre bu terimlerin katsayısı (a ve b) 0 olmalıdır. Denklemde a ve b yerine 0 koyduğumuzda

$$y^2 = x^3 + cx^2 + dx + e \quad (3.23)$$

elde ederiz. Burada x yerine x-c/3 yazarsak:

$$y^2 = (x-c/3)^3 + c(x-c/3)^2 + d(x-c/3) + e \quad (3.24)$$

$$\Rightarrow \quad y^2 = x^3 - cx^2 + c^2x/3 - c^3/27 + cx^2 - 2c^3x/3 + c^2/9 + dx - cd/3 + e \quad (3.25)$$

$$\Rightarrow \quad y^2 = x^3 - c^2x/3 + dx + 2c^3/27 - cd/3 + e \quad (3.26)$$

$$\Rightarrow \quad y^2 = x^3 - (c^2/3 - d)x + (2c^3/27 - cd/3 + e) \quad (3.27)$$

$-(c^2/3 - d) = A$, $2c^3/27 - cd/3 + e = B$ koyarsak

$$\Rightarrow \quad y^2 = x^3 + Ax + B \text{ biçimini alır.} \quad (3.28)$$

Bu denklemde y'nin birden çok kökünün olabilmesi için $4A^3 + 27B^2 \neq 0$ olması gerekir.

Tanım:

p asal, F_p asal sonlu alan olmak üzere $a,b \in F_p$; $4a^3+27b^2 \neq 0 \pmod p$ şartını sağlıyor ise, F_p üzerinde $E(F_p)$ eliptik eğrisi,

$$y^2 \equiv x^3 + ax + b \pmod p \text{ denklemi için,} \quad (3.29)$$

$a, b \in F_p$ parametreleri ile tanımlanan, $x, y \in F_p$, $P = (x, y)$ noktalarından ve sonsuz olarak adlandırılan O noktasından oluşur.

$y^2 = x^3 + ax + b \pmod p$ denklemi $E(F_p)$ eliptik eğrisinin tanım denklemi olarak adlandırılır. Verilen bir $P = (x_p, y_p)$ noktası için, x_p, P noktasının X koordinatı; y_p, P noktasının Y koordinatı olarak adlandırılır.

$E(F_p)$ eliptik eğrisinin üzerindeki noktaların sayısı $\#E(F_p)$ ile gösterilir ve Hasse Teoremi (Koblitz 1987b)

$p + 1 - 2\sqrt{p} \leq \#E(F_p) \leq p + 1 + 2\sqrt{p}$ olduğunu ortaya koymaktadır.

$E(F_p)$ üzerinde toplama ve çarpma işlemleri yukarıda "Eliptik Eğriler" başlığı altında anlatıldığı şekilde tanımlanabilir.

Toplama kuralları ışığı altında $E(F_p)$ bir grup oluşturmaktadır. Daha da ötesinde $P, R \in F_p$ için, $P + R = R + P$ olduğu göz önüne alınırsa Abelian grup oluşturmaktadır. Eliptik eğri şifreleme sistemleri, eliptik eğri üzerindeki bir noktanın skaler çarpımına dayanmaktadır. Bir k sayısı ve $P \in F_p$ noktası verildiğinde, kP, P noktasının k defa kendisi ile toplanmasıyla elde edilmektedir.

3.1.2.2 F_2m Sonlu Alanında Eliptik Eğriler

$y^2 + axy + by = x^3 + cx^2 + dx + e$ eğrisinde y yerine $a^3y+[(a^2d+b)/a^3]$ ve,

x yerine $a^2x + b/a$ yazılacak olursa denklem

$y^2 + xy = x^3 + ax^2 + b$ biçimini alır. (3.30)

F_{2^m}, karakteristik 2 sonlu alan ve $a,b \in F_{2^m}$; $b \neq 0$ olmak üzere, F_{2^m} üzerinde $E(F_{2^m})$ eliptik eğrisi F_{2^m}'de, (3.30) denklemi için $a,b \in F_{2^m}$ parametreleri ile tanımlanan, $x,y \in F_{2^m}$, P = (x,y) noktalarından ve sonsuz olarak adlandırılan O noktasından oluşur.

$y^2 + xy = x^3 + ax^2 + b$ (mod p) denklemi $E(F_{2^m})$ eliptik eğrisinin tanım denklemi olarak adlandırılır. Verilen bir $P = (x_p, y_p)$ noktası için, x_p, P noktasının x koordinatı, y_p P noktasının y koordinatı olarak adlandırılır.

$E(F_{2^m})$ eliptik eğrisinin üzerindeki noktaların sayısı $\#E(F_{2^m})$ ile gösterilir ve Hasse Teoremi (Koblitz 1987b)

$2^m + 1 - 2\sqrt{2^m} \leq \#E(F_{2^m}) \leq 2^m + 1 + 2\sqrt{2^m}$ olduğunu ortaya koymaktadır.

$E(F_{2^m})$ üzerinde toplama ve çarpma işlemleri yukarıda Eliptik Eğriler başlığı altında anlatıldığı şekilde tanımlanabilir. Benzer şekilde $E(F_{2^m})$ bir Abelian grup oluşturmaktadır ve skaler çarpım $E(F_p)$ ile aynıdır.

3.1.2.3 Eliptik Eğri Üzerindeki Noktaların Elde Edilmesi

Eliptik eğrilerle ilgili sorunlardan bir tanesi de eliptik eğri üzerinde bir noktanın özellikle de taban noktasının (skaler çarpımla eğri üzerindeki tüm noktaları veren nokta) bulunmasıdır. Doğrudan yaklaşım bir x değeri belirleyip bu x'e eliptik eğri denklemini sağlayan y değeri olup olmadığına bakılmasıdır. Ele aldığımız x için y elde edilemiyorsa bir başka x değeri ile devam edilir. Varsayalım ki; y değeri sağlayabilen bir x değeri seçtik. Bu durumda

$$y^2 = a \bmod p \qquad (3.31)$$

denkleminden y değerinin hesaplanması gerekmektedir. Bu işlemin deterministik bir karmaşıklıkta çözümü bulunmamaktadır. "Shank'ın yaklaşım yöntemi" ile çözüm $O(\ln^3 p)$ adım gerektirmektedir. y değerinin hesaplanmasının zorluğundan öte, asıl zorluk eliptik eğri denkleminde yerinde konulduğunda kare sonuç veren x'leri bulmaktadır.

3.1.2.4 Eliptik Eğri Düzeyi

Bir eliptik eğrinin düzeyi, o eliptik eğri üzerindeki noktaların sayısını ifade etmektedir. Verilen bir noktanın düzeyi ise o noktayı kullanarak eğri üzerindeki noktalardan kaç tanesinin elde edilebileceğinin sayısıdır.

3.1.2.5 Eliptik Eğri Alan Parametreleri

Üzerinde işlem yapılan eliptik eğriyi belirleyebilmek için, eğri bir takım özelliklerini ortaya koyan değerlerle ifade edilmelidir. İki tür eliptik eğri alan parametresinden bahsedilebilir: F_p üzerinde eliptik eğri alan parametreleri ve F_{2^m} üzerinde eliptik eğri alan parametreleri. Takip eden kısımlarda bu parametreler açıklanmıştır.

F_p üzerinde eliptik eğri alan parametreleri

F_p üzerinde eliptik eğri alan parametreleri bir altılıdan oluşmaktadır:

$$T\ (p, a, b, G, n, h)$$

Burada;

p : Sonlu alan F_p'yi belirleyen tamsayı,

a,b : $y^2 \equiv x^3 + ax + b \pmod p$ denklemi ile tanımlanan eğriyi belirleyen elemanlar,

G : $G = (x_G, y_G)$ taban noktası,

n : G noktasının derecesini veren asal sayı

h : $\#E(F_p) / n$

Burada verilen parametreler, EEŞ'ye dayanan bir açık anahtar şifreleme sistemi için mutlak gerekli olan F_p üzerinde bir eliptik eğriyi ve taban noktasını kesin olarak tanımlamaktadır.

F_{2^m} üzerinde eliptik eğri alan parametreleri

F_{2^m} üzerinde eliptik eğri alan parametreleri bir yediliden oluşmaktadır:

$$T\ (m, f(x), a, b, G, n, h)$$

Burada;

m : Sonlu alan F_{2^m}'yi belirleyen tamsayı,

$f(x)$: m dereceli F_{2^m}'yi simgeleyen indirgenemez polinom

a,b : $y^2 + xy \equiv x^3 + ax^2 + b$ denklemi ile tanımlanan eğriyi belirleyen elemanlar,

G : $G = (x_G, y_G)$ taban noktası,

n : G noktasının derecesini veren asal sayı

h : $\#E(F_2^m) / n$

Burada verilen parametreler, EEŞ'ye dayanan bir açık anahtar şifreleme sistemi için mutlak gerekli olan F_2^m üzerinde bir eliptik eğriyi ve taban noktasını kesin olarak tanımlamaktadır.

3.1.2.6 Nokta Sıkıştırma

Eğri üzerindeki her bir nokta x ve y koordinatlarını ifade eden ikili ile simgelenmektedir ve *y* değeri eğri denkleminden *x* değerine göre elde edilmektedir. Ancak, eğri üzerindeki her bir x koordinatı değerine karşılık bir çift y koordinatı değeri bulunmaktadır. Bu durumda sadece *x* değerinin ve beraberinde hangi *y* değerinin kullanılacağını belirten bir ikil kullanılarak eğri üzerinde her hangi bir nokta tam olarak ifade edilebilmektedir. Eğri üzerinde *x* değerini yerine koyarak *y* değerinin elde edilmesi işlemsel olarak çok pahalı olmadığından, tutulacak ya da aktarılacak verilerin kısaltılması amacıyla nokta sıkıştırma kullanılabilmektedir (Hankerson 2004).

3.1.2.7 Eliptik Eğri Nokta Gösterimleri

Eliptik eğri noktaları, eğriler üzerinde gerçekleştirilen işlemlerde kolaylık sağlayacak şekilde faklı gösterimler kullanılarak ifade edilebilmektedir. Bundan önceki kısımlarda kullanılan P(x,y) gösterimi doğrusal (affine) gösterim olarak adlandırılmaktadır. Bir diğer gösterim şekli, izdüşümsel gösterim olarak adlandırılan, iki boyutlu düzlemsel gösterimden üç boyutlu gösterime geçişi sağlayan sistemdir.

Doğrusal gösterimde eğriyi ifade eden Weierstrass biçimdeki ,

$$E : y^2 + axy + by = x^3 + cx^2 + dx + e \text{ eğrisi}$$

İzdüşümsel gösterimde,

$$E: Y^2Z + aXYZ + bYZ^2 = X^3 + cX^2Z + dXZ^2 + eZ^3 \ (Z \neq 0) \text{ (Husemuller 2004)}$$

şeklini almakta, P(x,y) noktası P(X/Z,Y/Z) ile eşdeğer olmakta ve P(Z:Y:Z) şeklinde ifade edilmektedir.

İzdüşümsel gösterimin biraz daha geliştirilmiş biçimi olan Jacobian ve Chudnovsky İzdüşümesel gösterimleri de (Chudnovsky 1986) kullanılabilmektedir. Bu gösterimlerde Weierstrass biçimi sırasıyla aşağıdaki biçime dönüşmekte.

$$E: Y^2 + aXYZ + bYZ^3 = X^3 + cX^2Z^2 + dXZ^4 + eZ^6 \ (Z \neq 0)$$

$$E: Y^2 + aXYZ + bYZ^2 = X^3Z + cX^2Z^2 + dXZ^3 + eZ^4 \ (Z \neq 0)$$

ve sırasıyla doğrusal gösterimdeki P(x,y) noktasının karşılığı sırasıyla $P(X/Z^2, Y/Z^3)$ ile $P(X/Z, Y/Z^2)$ olmaktadır.

3.2 Eliptik Eğri Şifreleme

EEŞ için eliptik eğri gruplarının asal sayılarda ya da 2^m alanında tanımlanan eğrilerle kullanılmaktadır. Veri alış verişinde bulunan iki taraf, Alıcı (A) ve Gönderici (B), arasında aşağıdaki gibi bir işlemler dizisi gerçekleştirilir:

A, bir anahtar çifti belirler ve açık anahtarını B'ye gönderir.

B, mesaj göndermek istediği zaman, A'nın açık anahtarı ile şifreler ve mesajı A'ya gönderir.

A, mesajı aldığı zaman, kendi anahtarı ile mesajın şifresini çözer.

Şifreleme için kullanılan anahtar, *e*, ile eliptik eğri üzerindeki noktalara karşılık gelen mesaj şifrelenir (ortaya çıkan şifreli metin de eğri üzerinde bir noktadır) ve şifre anahtarının çarpmaya göre tersi olan deşifre anahtarı, *d,* kullanılarak şifreli mesaj çözülür.

3.2.1 Eliptik Eğri Anahtar Çiftleri

Eliptik eğri alan parametreleri, T, verildiğinde, T ile eşlenen [1, *n*-1] aralığında gizli anahtar *d* ve eliptik eğri açık anahtarı $Q(x_Q, y_Q)$'dan oluşan bir (*d*,Q) eliptik eğri anahtar çifti bulunmaktadır. Burada aslında Q = *d*G'dir. *d*, *n* değerine göre rasgele seçilmekte, Q ise verilenlerden elde edilmektedir (SECG 2000a).

3.2.2 Şifreleme Sistemleri

Sonlu alanlar için ALP'nin varsayılan zorluğuna dayanan birçok şifreleme sisteminin, eliptik eğrilere uygulaması bulunmaktadır. Bu sistemlerde açık metnin her bir karakteri alanın bir elemanı ile eşlenmektedir. Eliptik eğrilerde ise, karakter dizilerinin eliptik eğri üzerindeki noktalarla eşlenmesi gerekmektedir.

3.2.2.1 Eliptik Eğri Diffie Helman (EEDH)

A ve B çiftinin, iletişim için bir gizli anahtar üzerinde anlaşmak istediklerini varsayalım. İlk olarak her ikisi açık olarak bir F_q sonlu alanı ve bu alan üzerinde eliptik eğri E belirlerler. Yine açık olarak E üzerinde rasgele bir nokta P belirlenir. A, gizli olarak bir tamsayı *a* seçer ve *a*P'yi hesaplar. Benzer şekilde, B, gizli olarak bir tamsayı *b* seçer ve *b*P'yi hesaplar. A ve B, sırasıyla *a* ve *b* tamsayılarını gizli tutarken, *a*P ve *b*P açıklanır. A gizli *a* değerine, B gizli *b* değerine sahiptir. Hem A, hem de B *ab*P'yi hesaplayabilmektedirler ve bu anlaşılmış ortak

gizli anahtardır. Sadece P, *a*P ve *b*P verilmişken, *ab*P'nin hesaplanmasının EEALP'yi çözmeyi gerektirdiğine inanılmaktadır (ANSI 1999).

3.2.2.2 Eliptik Eğri ElGamal

Bu bir başka açık anahtar şifre sistemidir. F_q sonlu alanı üzerinde eliptik eğri E ve bu eğri üzerinde P noktası açıktır. A ve B sırasıyla a_A ve a_B gizli tamsayılarını üretir ve saklarlar. Ardından a_AP ve a_BP değerlerini hesaplayıp yayımlarlar. A, M metnini B'ye göndermek istediğinde, gizli bir *k* hesaplar ve B'ye (*k*P, M +ka_BP) ikilisini gönderir. B elinde bulunan a_B ile eline geçen ikilinin ilk değerini çarparak ka_BP değerini hesaplar. A'dan gelen ikilinin ikinci değerinden elde ettiği sonucu çıkartarak M metnini elde eder (Hankeson vd. 2004).

3.2.2.3 Eliptik Eğri Anahtar Değişim Algoritması

ADA, SKIPJACK algoritmasının açıklanmasıyla duyulan ABD'nin Anahtar Değişim Algoritmasıdır (KEA - Key Exchange Algorithm). Algoritmanın eliptik eğrilerle uygulanması şu şekildedir: F_p üzerinde E eliptik eğrisi ve *q* asal düzeyinde E eğrisi üzerinde bir P noktası açıktır. A, gizli $x_A \in Z_q$ seçer, x_AP değerini hesaplar ve açıklar. B gizli $x_B \in Z_q$ seçer, x_BP değerini hesaplar ve açıklar. B ile haberleşmek için A, rasgele bir değer $r_A \in Z_q$ seçer ve r_AP değerini hesaplayarak B'ye gönderir. B, rasgele bir değer $r_B \in Z_q$ seçer ve r_BP değerini hesaplayarak B'ye gönderir. A artık r_A, x_A ve r_BP değerlerini, benzer şekilde B r_B, x_B ve r_AP değerlerini bilmektedir. A, $w_A = r_A(x_B P) + x_A(r_B P)$ değerini; B, $w_B = r_B(x_A P) + x_B(r_A P)$ değerini hesaplar. $w_A = w_B$ olduğundan, A ve B gizli anahtar üzerinde anlaşmıştır (NIST 2000).

3.2.2.4 Eliptik Eğri Sayısal İmza Algoritması

Eliptik Eğri Sayısal İmza Algoritması (EESİA) 1998 yılında Amerikan Ulusal Standartlar Enstitüsü (ANSI) X9.62 Standardı (ANSI 1998) ile veri ve mesaj güvenliğinin sağlanması ve amacıyla imza yaratılması ve onaylanması için standartlaştırılmıştır. Algoritmanın işleyişi için A ve B eliptik eğri alan parametreleri, eğri üzerinde açık anahtar olarak kullanılacak bir Q noktası ve bir hash fonksiyonu üzerinde anlaşırlar; A bir gizli anahtar *d* belirler Anlaşma sağlandıktan sonra A geçici kullanım için bir tamsayı *k* ve eğri üzerinde R(x_R, y_R) noktası belirler. $r = \neg x_R$ ve gönderilecek ileti hash fonksiyonundan geçirilerek elde edilen değer *k* ve *r.d* değerleriyle işleme alınarak *s* elde edilir. Elde edilen S = (*r*, *s*) imzası ve M iletisi B'ye gönderilir. B aldığı iletideki M değerini hash fonksiyonundan geçirdikten sonra s ile çarparak u_1 ve *r* . *s* çarpımını yaparak u_2 değerlerini elde eder. Buradan $R = (x_R, y_R) = u_1G + u_2Q$ elde

edilir. Eğer elde edilen $x_R = \neg r$ ise imza doğrulanmıştır yoksa geçersizdir. İşlemler aşağıda özetlenmiştir.

Parametreler : T (m, f(x), a, b, G, n, h) veya T (p, a, b, G, n, h) eliptik eğri parametreleri

Q noktası ve imzalayacak taraf için d değeri

İmza, S(r, s) : Geçici k değeri ve yine geçici R = (x_R, y_R) noktası elde edilir.

$r = \neg x_R$

e = hash (M)

$s = k^{-1}.(e+r.d) \bmod n$

İmza Onayı : e = hash (M)

$u_1 = e \,.\, s^{-1}$

$u_2 = r \,.\, s^{-1}$

$R = (x_R, y_R) = u_1G + u_2Q$

$v = \neg x_R$

$r = v$ ise, imza doğrulanmıştır.

3.3 Şifreleme Yöntemlerinin Karşılaştırılması

Eliptik eğrilere dayanan şifreleme yöntemlerinin dayandıkları şifreleme algoritmalarının klasik yöntemlerine nazaran daha kısa anahtar boyları ile eşdeğer güvenlik sağladıkları yapılan araştırmalarla ortaya konmuştur (ANSI 1998). Aşağıdaki çizelgede (Çizelge 3.2) ihtiyaç duyulan güvenlik düzeyi için gerekli anahtar boyları gösterilmiştir.

Çizelge 3.2 Eş Güvenlik Düzeylerindeki Anahtar Boyları (ANSI 1998), [1, 4]

Simetrik anahtar (ikil)	EE şifreleme (ikil olarak n)	DH/RSA (ikil olarak) modülo
56	112	512
80	160	1024
112	224	2048
128	256	3072
192	384	7680
256	512	15360

NIST tarafından yayımlanan bir çalışma [4], gelecek otuz yıllık dönem için işlemci gücündeki artışları göz önüne alarak gerekli güvenlik düzeylerini belirlemiştir. Simetrik anahtarların ikil olarak uzunluğu referans olarak alındığında, ihtiyaç duyulan anahtar boyları Çizelge 3.3'de verilmiştir.

Çizelge 3.3 İhtiyaç Duyulan Güvenlik Düzeyleri [4]

Asgari güvenlik düzeyi (ikil)	80	112	128
Zaman Dilimi	2010'a kadar	2011-2035	2035'den sonra

Çizelge 3.2 ve Çizelge 3.3'de verilen değerler birlikte değerlendirildiğinde, günümüz için 1024 ikil açık anahtar şifreleme yeterli olabilmekteyken 2035 yılından sonrası için bunun üç katı öngörülmektedir. Eliptik eğri tabanlı sistemlerde ise günümüzde 160 ikil 2035 yılından sonrası içinse bunun yaklaşık %50 fazlası, 256 ikil yeterli olmaktadır. Şifreleme ve şifre çözme işlemleri anahtar boyunun artışıyla doğru orantılı olduğuna göre, artan güvenlik ihtiyaçlarını karşılamak için 3 kat daha fazla işlem gücü yerine sadece 1,5 kat işlem gücü yeterli olabilecektir. Sorun sadece işlem gücünde değil, bu kadar büyük değerlerin işlenmesi ve aktarımında da ortaya çıkmaktadır.

3.4 Eliptik Eğri Şifreleme Gerçekleştirimleri

Eliptik eğri şifreleme gerçekleştirimi için yapılması gereken pek çok tercih bulunmaktadır. Bunların arasında eliptik eğri parametrelerinin belirlenmesi (kullanılacak sonlu alan, alanın nasıl ifade edileceği, eğri denklemi), alan aritmetiğinde kullanılacak algoritmalar, eliptik eğri matematiği algoritmaları ve protokol ile ilgili matematik algoritmaları en başta gelmektedir (Joye vd. 2001). Tercihleri güvenlik gereksinimleri, uygulama ortamı, gerçekleştirim ortamıyla ilgili kısıtlar ve iletişim ortamının kısıtları yönlendirebilmektedir. En iyi tercihler kümesini oluşturmak imkansız değilse de oldukça zordur. Örneğin, kişisel bilgisayar ortamında yapılacak bir gerçekleştirim ile duyarga ağları ortamında yapılacak bir gerçekleştirim için kullanılabilecek algoritmalar da, eliptik eğri değişkenleri de bir birlerinden oldukça farklı olabilmektedir.

Gerçekleştirimler sırasında rasgele üretilen eğriler kullanılabileceği gibi standartlaşmış eğriler de kullanılabilmektedir. Rasgele üretilen eğri kullanımı, üretilen eğrilerin şifreleme amacıyla kullanımına uygun olup olmadığının sınanmasını gerektirdiğinden tercih edilmemektedir. Ayrıca birlikte işleyebilirliği sağlayabilmek amacıyla da standart eğriler tercih edilmektedir. Bu çalışmada da NIST (1999) ve SECG (2000b) tarafından tavsiye edilen eğriler kullanılmıştır. Bu eğrilerle ilgili değerler Ek-1'de sunulmuştur.

4. GÜVENLİ SOKET KATMANI

GSK Protokolünün temel amacı iletişim kuran iki uygulama arasında gizlilik ve güvenirlik sağlamaktır. Protokol iki katmandan oluşmaktadır. En alt katmanda, güvenilir bir aktarım protokolünün (TCP gibi) üzerinde işleyen GSK Kayıt Protokolü (**SSL Record Protocol)** bulunmaktadır. GSK Kayıt Protokolü, farklı üst düzey protokollerin sarmalanması için kullanılmaktadır. GSK Tokalaşma Protokolü (**SSL Handshake Protocol)** ise**,** sunucu ile istemcinin bir birinin aslıyla aynılığını kanıtlamasına ve uygulama programı veri aktarımına başlamadan önce şifreleme algoritma ve anahtarının belirlenmesine olanak sağlar. GSK'nın en önemli avantajı protokolden bağımsız olmasıdır. GSK Protokolü, üst katman protokollerinin altında saydam bir şekilde yer alabilir. GSK protokolü üç temel özelliği olan bağlantı güvenliği sağlar (Freier vd. 1996):

- Bağlantı gizlidir. İletişim başlayıp şifre anahtarları belirlendikten sonra, şifreleme kullanılır.
- Karşı tarafın kimliği açık anahtar şifreleme kullanılarak doğrulanabilir.
- Bağlantı güvenilirdir. İleti aktarımı, ileti bütünlüğünü kontrol eden mekanizmalar içermektedir.

4.1 GSK Protokolü

TCP/IP, internet üzerinden verinin aktarımı ve yönlendirilmesi ile sorumludur. HTTP, LDAP, IMAP vb protokoller uygulama taleplerini yerine getirmek için TCP/IP üzerinde işlerler.

GSK, TCP/IP'nin üzerinde, üst düzey protokollerin altında yer alır. Üst düzey protokoller adına TPC/IP servislerini kullanır ve GSK uyumlu bir sunucunun aslıyla aynılığını GSK uyumlu istemciye kanıtlamasını, aynı şekilde istemcinin aslıyla aynılığını sunucuya kanıtlamasını ve her iki taraf arasında şifreli bir iletişim kurulmasını sağlar (Apostopolous vd. 1999).

Bu olanaklar Internet ve diğer TCP/IP ağları üzerinden iletişim ile ilgili bir takım hususların dikkate alınmasını gerektirir:

GSK sunucu aslıyla aynılığını kanıtlama, istemcinin sunucu kimliğinden emin olmasını sağlar. GSK uyumlu istemci yazılımı, sunucunun sertifikası ile açık kimliğinin (ID) geçerli

olduğunu ve istemcinin güvenilir Sertifika Sağlayıcı'ları (SS, CA - Certificate Authority) listesinde bulunan bir SS tarafından sağlandığını standart açık anahtar şifreleme tekniklerini kullanarak kontrol edebilir.

GSK istemci aslıyla aynılığını doğrulama, sunucunun, kullanıcı kimliğinden emin olmasını sağlar. Sunucu doğrulama için kullanılan tekniklerin aynılarını kullanarak, GSK uyumlu sunucu yazılımı, istemcinin sertifikası ile açık kimliğinin geçerli olduğunu ve sunucunun güvenilir SS'ler listesinde bulunan bir SS tarafından sağlandığını kontrol edebilir. Bu doğrulama, sunucunun müşterisine hesap bilgilerini göndermekte olan bir banka olduğunda müşteri kimliğini kontrol etmek istemesine benzer durumlarda önem kazanmaktadır.

Şifreli GSK bağlantısı, istemci ile sunucu arasında gönderilen tüm bilginin gönderici tarafından şifrelenmesi ve alıcı tarafından şifrenin açılmasını gerektirerek üst düzeyde güvenlik sağlamaktadır. İlave olarak, güvenli GSK bağlantısı üzerinden gönderilen tüm verinin aktarım sırasında değiştirilip değiştirilmediğini otomatik olarak tespit etmeyi sağlayan bir mekanizma içermektedir.

GSK protokolü iki alt protokol içermektedir: GSK Kayıt Protokolü ve GSK Tokalaşma Protokolü**.** GSK Kayıt Protokolü, veri aktarımı için kullanılacak biçemi belirlemektedir. Her ikisi de GSK uyumlu olan istemci sunucu arasında GSK bağlantısı kurulurken, Tokalaşma Protokolü, Kayıt Protokolü'nü kullanarak bir iletiler dizisi alış verişi gerçekleştirir. Bu ileti alış verişi aşağıdakileri sağlamak için tasarlanmıştır:

- Sunucunun aslıyla aynılığının istemciye kanıtlanması.
- İstemci ile sunucunun şifreleme algoritması seçimine olanak sağlanması
- İstenirse, istemcinin aslıyla aynılığının sunucuya kanıtlanması.
- Paylaşılan sırların üretilmesi için açık anahtar şifreleme tekniklerinin kullanılması.
- Şifreli GSK bağlantısının kurulması.

4.1.1 GSK Tokalaşma

GSK protokolü açık anahtar ve simetrik anahtar şifreleme yöntemlerinin birleşimini kullanır (Coarfa 2002). Simetrik anahtar şifreleme, açık anahtar şifreleme yönteminden çok daha

hızlıdır, ancak, açık anahtar şifreleme daha iyi doğrulama teknikleri sağlamaktadır. Bir GSK oturumu daima, GSK Tokalaşma adı verilen ileti alış verişi ile başlar. Tokalaşma, sunucunun aslıyla aynılığını istemciye onaylatmak için açık anahtar şifreleme yöntemleri kullanmasını ve ardından takip edecek oturum sırasında sunucu ile istemci arasında hızlı şifreleme, şifre çözme ve aktarılan veriler üzerindeki değişiklikleri izleme amacıyla kullanılacak olan simetrik anahtarların beraberce belirlenmesine olanak sağlar. Tokalaşma, gerekirse, istemcinin aslıyla aynılığını sunucuya kanıtlamasını da sağlamaktadır.

GSK Tokalaşma işleyişinin adımları, şifreleme yöntemi olarak RSA anahtar değişimi kullanıldığı varsayılarak, aşağıda verilmiştir (Freier vd. 1996):

1. İstemci, sunucuya GSK sürüm numarasını, şifreleme ayarlarını, rasgele üretilmiş veriyi ve sunucunun istemci ile GSK kullanarak haberleşmesini sağlayacak diğer bilgileri aktarır.

2. Sunucu, istemciye GSK sürüm numarasını, şifreleme ayarlarını rasgele üretilmiş veriyi ve istemcinin sunucu ile GSK kullanarak haberleşmesini sağlayacak diğer bilgileri aktarır. Sunucu ayrıca kendi sertifikasını gönderir ve gerekiyorsa, istemcinin sertifikasını talep eder.

3. İstemci, sunucunun gönderdiği bilgilerin bir kısmını kullanarak sunucunun aslıyla aynılığını kanıtlar. Sunucu onaylanamazsa, kullanıcı problem hakkında uyarılır ve şifreli ve aslıyla aynılığı kanıtlanmış bir bağlantının mümkün olmadığı bildirilir. Sunucu onaylanırsa, sonraki adıma geçilir.

4. Tokalaşma sırasında bu aşamaya kadar üretilen tüm veriyi kullanarak istemci oturum için bir öncül sır üretir, sunucunun açık anahtarını kullanarak şifreler ve şifreli öncül sırrı sunucuya gönderir.

5. Sunucu istemcinin aslıyla aynılığının kanıtlanmasına ihtiyaç duyarsa, istemci, sadece sunucu ve istemci tarafından bilinen ve bu tokalaşma sürecine özgü olan başka bir veri parçasını imzalar. Bu durumda istemci şifreli öncül sır ile birlikte imzalanmış veriyi ve kendi sertifikasını gönderir.

6. Sunucu, istemcinin aslının aynılığıyla kanıtlanmasını talep etmişse, istemciyi doğrulamaya çalışır. İstemci onaylanamıyorsa, oturum sonlandırılır. İstemci

onaylanırsa, sunucu kendi anahtarını kullanarak öncül sır şifresini açar ve öncül sırrı bir takım işlemlere tabi tutarak asıl sırrı oluşturur. Bu sırada istemci de elinde bulunan öncül sırrı aynı işlemlere tabi tutarak asıl sırrı oluşturmaktadır.

7. Hem istemci hem de sunucu asıl sırrı kullanarak, GSK oturumu sırasında verinin şifrelenmesini, şifreli verinin açılmasını ve aktarılan verinin bütünlüğünün kontrolünü sağlamak için kullanılacak olan birer simetrik anahtar olan oturum anahtarlarını elde ederler.

8. İstemci, sunucuya bir ileti göndererek bundan sonraki iletişimin şifreli gerçekleşeceğini bildirir. Ardından şifreli bir ileti göndererek Tokalaşma işleminin istemci tarafının bittiğini bildirir.

9. Sunucu istemciye bir ileti göndererek bundan sonraki iletişimin şifreli gerçekleşeceğini bildirir. Ardından şifreli bir ileti göndererek Tokalaşma işleminin sunucu tarafının bittiğini bildirir.

10. GSK Tokalaşma kısmı artık sonlanmış ve GSK oturumu başlamıştır. İstemci ve sunucu bir birlerine gönderdikleri verileri şifrelemek, açmak ve bütünlüğünü kontrol etmek için oturum anahtarlarını kullanırlar.

Hem istemci hem de sunucu doğrulamanın açık-gizli anahtar çiftinin birisiyle bir parça verinin şifrelenmesini, diğeriyle de şifrenin açılmasını gerekmektedir:

- Sunucu aslıyla aynılığını kanıtlama durumunda, istemci öncül sırrı sunucunun açık anahtarı ile şifreler. Sadece karşılık gelen gizli anahtar doğru olarak sırrı açabilir, bu şekilde, istemci belirli bir düzeyde açık anahtarla eşlenmiş kimliğin istemcinin bağlı olduğu sunucuya ait olduğu konusunda emin olabilir. Aksi takdirde, sunucu öncül sırrı açamaz ve oturum için gereken simetrik anahtarları üretemez ve oturum sonlanır.

- İstemci aslıyla aynılığını kanıtlama durumunda, istemci kendi gizli anahtarıyla bir miktar rasgele veriyi şifreler. İstemcinin sertifikasındaki açık anahtar, yalnız karşılık gelen gizli anahtar kullanılmışsa sayısal imzayı doğrulayabilir. Aksi takdirde, sunucu sayısal imzayı doğrulayamaz ve oturum sonlandırılır.

4.1.2 Sunucu Aslıyla Aynılığını Kanıtlama

Sunucu aslıyla aynılığını kanıtlama için GSK Tokalaşma ikinci adımda anlatıldığı şekilde sunucu istemciye aslıyla aynılığını kanıtlamak için sertifikasını gönderir. İstemci bu sertifikayı üçüncü adımda, sunucunun kimliğinin, sertifika tarafından temsil edilen kimlik olup olmadığını doğrulamakta kullanır.

Açık anahtarla, açık anahtarı içeren sertifikanın temsil ettiği sunucu arasındaki ilişkiyi doğrulamak için istemcinin aşağıdaki 4 soruya olumlu yanıt alması gerekir:

- Sertifikanın geçerliliği devam ediyor mu? İstemci sunucu sertifikasının geçerlilik dönemini inceler. Günün tarihi geçerlilik döneminin dışındaysa, doğrulama süreci devam etmez. Mevcut tarih sertifikanın geçerlilik dönemine dahilse, bir sonraki adıma geçilir.

- SS, Güvenilir SS'ler listesinde mi? Her bir GSK uyumlu istemci, hangi sertifikaları kabul edebileceğini gösteren, güvenilir SS'ler listesi tutmaktadır. Eğer sertifikayı sağlayan SS, istemcinin güvenilir SS'ler listesinde yer alıyorsa, bu sorunun yanıtı olumludur. Sertifika sağlayan SS, istemcinin listesinde yer almıyorsa, istemci sunucunun aslıyla aynılığını kanıtlayamaz.

- Sertifika sağlayan SS'nin açık anahtarı sayısal imzayı doğruluyor mu? İstemci SS'nin sertifikasından aldığı açık anahtarı kullanarak sunucunun sağladığı sertifikadaki SS'nin sayısal imzasını kontrol eder. SS'nin sayısal imzası, SS tarafından imzalandıktan sonra değiştirilmişse ya da SS tarafından sunucu sertifikasını imzalamak için kullanılan gizli anahtar açık anahtar ile uyumlu değilse, istemci sunucuyu doğrulamayacaktır. SS'nin imzası doğrulanabilirse, istemci sunucunun sertifikasını geçerli olarak kabul eder ve bu noktadan sonra devam etmeden önce dördüncü adımı gerçekleştirmek istemcinin sorumluluğundadır.

- Sunucunun sertifikasındaki alan adı, sunucunun kendi alan adıyla uyumlu mu? Bu adım, sunucunun, gerçekten de sunucu sertifikasında verilen ağ adresinde bulunduğundan emin olmayı sağlamaktadır. Aslında bu adım GSK protokolünün bir parçası değildir, ancak, araya girme (man-in-the-middle attack) saldırısından korunmak için gereklidir. İstemciler bu aşamayı gerçekleştirmeli ve sunucunun alan

adı sertifikada sağlanan alan adı ile uyumlu değilse bağlantıyı reddetmelidir. Bu aşamada da başarı sağlanırsa beşinci adıma geçilebilir.

Araya Girme Saldırısı (Man-in-the-Middle Attack): Araya girme, GSK kullanarak iletişim kurmak isteyen istemci ile sunucu arasına üçüncü bir şahsın girerek kendisini istemciye sunucu, sunucuya istemci olarak tanıtması, bu sırada da gelip giden anahtarlar yerine kendi anahtarlarını koyması ve istediği bilgiler üzerinde değişiklik yapmasıdır. İstemci, bilginin sunucudan, sunucu ise istemciden geldiğini düşündüğünden tüm mahrem bilgilerini paylaşabilmektedir. Dolayısıyla, sertifikada yer alan, alan adıyla gerçek alan adının istemci tarafından karşılaştırılması istemcinin kiminle görüştüğünden emin olması için çok önemlidir.

Sunucu doğrulanmıştır. İstemci GSK Tokalaşma işlemine devam edebilir. Eğer istemci her hangi bir nedenle beşinci adıma kadar gelemezse, sertifikanın temsil ettiği sunucu onaylanamaz ve kullanıcı sorun hakkında uyarılarak şifreli ve doğrulanmış bağlantı kurulamayacağı hakkında bilgilendirilir. Sunucu istemcinin aslıyla aynılığının kanıtlanmasını gerektiriyorsa, istemci doğrulama adımlarına geçilir.

4.1.3 İstemci Aslıyla Aynılığını Kanıtlama

GSK destekleyen sunucular istemcinin aslıyla aynılığını kanıtlama yönünde düzenlenebilirler. İstemciler arasında pek çok sertifikanın düzenlenmesi ve izlenmesi sorun olduğundan genellikle istemci aslıyla aynılığını kanıtlama gerçekleştirilmez, istemci aslıyla aynılığını kanıtlama, GSK güvenli bağlantısı üzerinden kullanıcı adı şifre doğrulama ya da benzeri şekilde uygulama katmanına bırakılır. Bu şekilde düzenlenmiş bir sunucu, aslıyla aynılığını kanıtlama talep ederse, istemci sertifikasının yanı sıra kendisini doğrulamak üzere bir parça imzalanmış veri gönderir. Sunucu imzalanmış veriyi, sertifikadaki açık anahtarla kontrol ederek istemcinin kimliğini doğrulamış olur.

Açık anahtarla açık anahtarı içeren sertifikanın temsil ettiği istemci arasındaki bağlantıyı doğrulamak için sunucunun aşağıdaki 4 soruya olumlu yanıt alması gerekir:

- İstemcinin açık anahtarı istemcinin imzasını doğruluyor mu? Sunucu istemci tarafından gönderilen imzalanmış veriyi, sertifikadaki açık anahtarla kontrol ederek istemcinin kimliğini doğrulamaya çalışır. İmzanın doğrulanmış olması istemcinin doğrulanması anlamına gelmemektedir, sunucuyu yanıltmak üzere bir sertifika yaratılmış olabilir, sertifikanın geçerliliğinin de kontrol edilmesi gereklidir.

- Sertifikanın geçerliliği devam ediyor mu? Sunucu istemci sertifikasının geçerlilik dönemini inceler. Günün tarihi geçerlilik döneminin dışındaysa, doğrulama süreci devam etmez. Mevcut tarih sertifikanın geçerlilik dönemine dahilse, bir sonraki adıma geçilir.

- SS, Güvenilir SS'ler listesinde mi? Her bir GSK uyumlu sunucu hangi sertifikaları kabul edebileceğini gösteren, güvenilir SS'ler listesi tutmaktadır. Eğer sertifikayı sağlayan SS, sunucunun güvenilir SSler listesinde yer alıyorsa, bu sorunun yanıtı olumludur. Sertifika sağlayan SS, sunucunu listesinde yer almıyorsa, sunucu istemcinin aslıyla aynılığını kanıtlayamaz.

- Sertifika sağlayan SS'nin açık anahtarı sayısal imzayı doğruluyor mu? Sunucu SS'nin sertifikasından aldığı açık anahtarı kullanarak istemcinin sağladığı sertifikadaki SS'nin sayısal imzasını kontrol eder. SS'nin sayısal imzası, SS tarafından imzalandıktan sonra değiştirilmişse ya da SS tarafından istemci sertifikasını imzalamak için kullanılan gizli anahtar açık anahtar ile uyumlu değilse, sunucu istemciyi doğrulamayacaktır. SS'nın imzası doğrulanabilirse, sunucu istemcinin sertifikasını geçerli olarak kabul eder.

- İstemci sertifikasındaki alan adı, istemcinin kendi alan adıyla uyumlu mu? Bu adım, istemcinin, gerçekten de istemci sertifikasında verilen ağ adresinde bulunduğundan emin olmayı sağlamaktadır. Aslında bu adım GSK protokolünün bir parçası değildir, ancak, araya girme saldırısından korunmak için gereklidir. Sunucular bu aşamayı gerçekleştirmeli ve istemcinin alan adı sertifikada sağlanan alan adı ile uyumlu değilse bağlantıyı reddetmelidir. Bu aşamada da başarı sağlanırsa istemci doğrulanmış olacaktır.

4.1.4 GSK ile Kullanılan Şifreleme Yöntemleri

GSK Protokolü, sunucu ve istemcinin bir birine aslıyla aynılığının kanıtlanması, sertifika aktarımı ve oturum anahtarları kurulması gibi işlemlerde kullanılmak üzere farklı şifreleme yöntemleri desteklemektedir. İstemci ya da sunucular, destekledikleri GSK sürümü, kabul edilebilir şifre düzeyi ve GSK uyumlu yazılımların ihracatındaki kısıtlamalar gibi etmenlere bağlı olarak, farklı şifreleme yöntemleri kümesi destekleyebilirler. Diğer işlevlerinin yanında, GSK Tokalaşma protokolü, sunucu ve işlemcinin bir birlerini doğrulamak, sertifikaları

aktarmak ve oturum anahtarları kurmak için kullanacakları şifreleme yöntemini belirleme görüşmelerini nasıl yapacaklarını da ortaya koyar.

GSK sunduğu şifreleme, kaynak doğrulama ve bütünlük korumanın yanı sıra anahtar belirleme, şifreleme ve hash işlemleri için farklı algoritmaları destekleyebilecek esnekliğe sahiptir. Ancak, bu algoritmaların belirli birleşimlerinin kullanılmasına olanak sağlanmaktadır. Bu şifreleme ve hash algoritmalarının birleşiminden oluşan birleşimler şifre süiti olarak adlandırılmaktadır ve sağladığı güvenlik düzeyi belirlidir. Örneğin, *TLS RSA WITH RC4 128 SHA* şifre süiti anahtar değişimi için RSA, veri şifreleme için 128-ikil RC4 ve hash fonksiyonu olarak SHA kullanmaktadır. Çizelge 4.1 ve Çizelge 4.2'de bu süitlerle ilgili detaylı bilgi verilmektedir.

Şifreleme yöntemleri açıklamaları aşağıdaki algoritmaları içermektedir (Dierks ve Allen 1999):

DES – Data Encryption Standart, ABD Hükümeti tarafından kullanılan şifreleme algoritmasıdır.

DSA – Digital Signature Algorithm, ABD Hükümeti tarafından kullanılan sayısal doğrulama standardının bir parçasıdır.

KEA – Key Exchange Algorithm, ABD Hükümeti tarafından anahtar değiş tokuşu için kullanılan algoritmalardan birisi.

MD5 – Rivest tarafından geliştirilen Message Digest algoritması.

RC2 ve RC4 – RSA Data security için geliştirilmiş Rivest şifreleme yöntemi.

RSA – Hem şifreleme hem de doğrulama için kullanılan açık anahtar algoritması.

RSA – RSA algoritmasını temel alan GSK anahtar değiş tokuşu algoritması.

SHA-1 Secure Hash Algorithm, ABD Hükümeti tarafından kullanılan hash işlevi.

SKIPJACK. ABD Hükümeti tarafından kullanılan, FORTEZZA uyumlu donanımlarda gerçekleştirilmiş simetrik anahtar algoritması.

Triple-DES. DES algoritmasının üç kez yinelenmesi.

KEA ve RSA gibi anahtar değişim algoritmaları, sunucu ve istemcinin, her ikisinin de GSK oturumu sırasında kullanacakları simetrik anahtarları belirlemelerini yönetir. En yaygın kullanılan GSK şifreleme yöntemi RSA anahtar değişimidir(Goldberg vd. 1998).

GSK Protokollerinin değişik sürümleri örtüşen şifreleme yöntemlerini desteklemektedir. Bunun yanında, hem istemci hem de sunucu tarafında desteklenen şifreleme yöntemlerinin etkinleştirilmesi ya da kapatılması mümkün olmaktadır. Bir istemci ile sunucu GSK Tokalaşma sırasında bilgi alış verişinde, destekledikleri şifreleme yöntemlerinden ortak olanların en kuvvetlisini belirler ve GSK oturumu için bunu kullanırlar.

4.1.4.1 RSA Anahtar Değişimi ile kullanılan Şifreleme Yöntemleri

Çizelge 4.1'de GSK tarafından desteklenen, RSA anahtar değişimi algoritması kullanan şifreleme yöntemleri en kuvvetliden en zayıfa doğru listelenmiştir (Freier 1996)

Çizelge 4.1 RSA Anahtar Değişimi Kullanan Şifreleme Yöntemleri

Kuvvet sınıfı ve Tavsiye edilen kullanım	Şifreleme Yöntemleri
En kuvvetli şifreleme yöntemi. Sadece ABD dahilinde kullanımına izin verilmektedir. Bu şifreleme yöntemi yüksek hassasiyete sahip veri ile işlem yapan bankalar ve diğer kuruluşlar için uygundur.	168 ikil şifreleme destekleyen Triple DES ile SHA–1 ileti doğrulama. Triple DES GSK tarafından desteklenen en güçlü şifreleme yöntemi olmakla beraber, RC4 kadar hızlı değildir. Triple DES, standart DES'in üç katı uzunluğunda anahtar kullanmaktadır. Anahtarın bu kadar uzun olması sayesinde diğer şifreleme yöntemlerinden daha fazla (yaklaşık $3.7 \cdot 10^{50}$ adet) anahtar üretilebilmektedir.
Kuvvetli şifreleme yöntemi. Sadece ABD dahilinde kullanımına izin verilmektedir. Bu şifreleme yöntemi şirket ve devlet ihtiyaçlarının çoğunu karşılamak için yeterlidir.	128 ikil şifreleme kullanan RC4 ile MD5 ileti doğrulama. RC4 desteklenen şifreleme yöntemlerinin en hızlısıdır.
	128 ikil şifreleme kullanan RC2 ile MD5 ileti doğrulama. RC2 RC4 kadar hızlı değildir. RC2 ve RC4 şifreleme yöntemleri kullandıkları 128 ikil anahtar uzunluğu ile (yaklaşık $3.4 \cdot 10^{38}$ adet) anahtar üretilebilmektedir.
İhraç edilebilir şifreleme yöntemleri. Bu şifreleme yöntemleri yukarıdakiler kadar	40 ikil şifreleme kullanan RC4 ile MD5 ileti doğrulama. RC4 desteklenen şifreleme yöntemlerinin en hızlısıdır.

güçlü değildir.	40 ikil şifreleme kullanan RC2 ile MD5 ileti doğrulama. RC2, RC4 kadar hızlı değildir. RC2 ve RC4 şifreleme yöntemleri kullandıkları 40 ikil anahtar uzunluğu ile (yaklaşık $1.1 \cdot 10^{12}$ adet) anahtar üretilebilmektedir.
En zayıf şifreleme yöntemi. Bu şifreleme yöntemi doğrulama ve aktarım bütünlüğünün korunmasını sağlamakta, ancak, şifreleme sağlamamaktadır.	Şifreleme yok, sadece MD5 ileti doğrulama. Bu yöntem aktarılan iletilerde değişiklik yapılıp yapılmadığını tespit edebilmek için MD5 kullanır. Tipik olarak sunucu ile istemci arasında ortak başka bir şifreleme yöntemi yoksa kullanılır.

4.1.4.2 FORTEZZA Şifreleme Yöntemleri

FORTEZZA, ABD hükümet kuruluşlarının sınıflandırılmamış hassas verinin yönetiminde kullanılan şifreleme yöntemidir. Federal hükümet tarafından geliştirilmiş, iki gizli donanım gerçekleştirimi bulunmaktadır. FORTEZZA KEA ve SKIPJACK. Çizelge 4.2'de GSK'de Fortezza şifreleme kullanımı gösterilmiştir.

Çizelge 4.2 FORTEZZA Kullanan Şifreleme Yöntemleri

Kuvvet sınıfı ve Tavsiye edilen kullanım	Şifreleme Yöntemleri
Kuvvetli FORTEZZA şifreleme yöntemi. Sadece ABD dahilinde kullanımına izin verilmektedir. Bu şifreleme yöntemi hükümet kuruluşlarının çoğunun ihtiyaçlarını karşılamak için yeterlidir.	128 ikil şifreleme kullanan RC4 ile SHA–1 ileti doğrulama. RC4 desteklenen şifreleme yöntemlerinin en hızlısıdır. Yaklaşık $3.4 \cdot 10^{38}$ adet anahtar üretilebilmektedir.
	SKIPJACK, 80 ikil şifrelemeli RC4 ile SHA-1 ileti doğrulama. SKIPJACK şifreleme yöntemi, FORTEZZA uyumlu donanımda gerçekleştirilmiş, sınıflandırılmış simetrik anahtar şifreleme algoritmasıdır.
En zayıf FORTEZZA şifreleme yöntemi. Bu şifreleme yöntemi doğrulama ve aktarım bütünlüğünün korunmasını sağlamakta, ancak, şifreleme sağlamamaktadır.	Şifreleme yok, sadece SHA–1 ileti doğrulama. Bu yöntem aktarılan iletilerde değişiklik yapılıp yapılmadığını tespit edebilmek için SHA–1 kullanır.

4.2 Sertifikalar

GSK, 1998 yılında belirlenen ITU X.509 v3 sertifika standardına uymaktadır (Dierks ve Allen 1999). Sertifikaların içeriği genel itibarıyla kullanıcıların ilgilenmesini gerektirmez, kullanıcılar sertifikaları güvenilir yetkili kurumlardan elde etmektedirler.

4.2.1 Distinguished Name

X.509 v3 sertifikası **distinguished name (DN)** ile açık anahtar arasında bir ilişki kurar. DN isim-değer çiftleri serisidir. Örneğin uid=ozgurb, sertifika konusu varlığını belirlemektedir. Örneğin, aşağıda verilen bir YTU çalışanı için tipik bir DN örneğidir:

uid=ozgurb, e=ozgurb@yildiz.edu.tr, cn=Ozgur Bozkurt, o=Yildiz Teknik U., c=TR

Her bir eşittir işaretinden önce gelen kısaltmanın anlamı aşağıda verilmiştir[5]:

uid : kullanıcı adı (user ID)
e : elmek adresi
cn : kullanıcın adı (common name)
o : kurum (organization)
c : ülke (country)

DN'ler çok farklı isim-değer ikilileri içerebilir. Bu ikililer hem sertifika konunun hem de LDAP protokolünü destekleyen dizinlerdeki kayıtları ifade etmektedir. DN oluşturma kuralları oldukça karmaşıktır ve detayları RFC 2253 (Wahl vd. 1997) dokümanında verilmiştir.

X.509 sertifika iki kısımdan oluşmaktadır:

- Aşağıdaki bilgileri içeren veri kısmı:
  - Sertifika tarafından desteklenen X.509 standardı sürüm numarası.
  - Sertifika seri numarası. Bir SS tarafından dağıtılan tüm sertifikalar o SS tarafından dağıtılan sertifikaları diğerlerinden ayırt eden bir seri numarası içermektedir.
  - Bilgi
  - Kullanıcının açık anahtarı ve kullanılan şifreleme algoritması hakkında bilgi.
  - Sertifikayı sağlayan SS'nın DN'i
  - Sertifikanın geçerlilik dönemi
  - Sertifika konusunun DN'i
  - Seçimlik sertifika eklentileri; istemci ya da sunucu tarafından kullanılan ilave bilgileri içeren alan. Örneğin, sertifika türü eklentisi, sertifikanın istemci GSK sertifikası mı, sunucu GSK sertifikasını mı elmek imza sertifikası mı olduğunu belirlemek için kullanılmaktadır.
- Aşağıdaki bilgileri içeren imza kısmı:
  - SS'nin kendi sayısal imzasını yaratmak için kullandığı şifreleme yöntemi.
  - SS'nin gizli anahtarı ile sertifikadaki tüm bilgileri şifreleyerek oluşturduğu SS'nin sayısal imzası.

Bir sertifikanın okunabilir biçimi Şekil 4.1'de verilmiştir, aynı sertifikanın 64 ikil olarak yazılımlar tarafından yorumlandığı biçimi şekil Şekil 4.2'de verilmiştir.

```
Certificate:
    Data:
        Version: v3 (0x2)
        Serial Number: 3 (0x3)
        Signature Algorithm: PKCS #1 MD5 With RSA Encryption
        Issuer: OU=Ace Certificate Authority, O=Ace Industry, C=US
        Validity:
            Not Before: Fri Oct 17 18:36:25 1997
            Not  After: Sun Oct 17 18:36:25 1999
        Subject: CN=Jane Doe, OU=Finance, O=Ace Industry, C=US
        Subject Public Key Info:
            Algorithm: PKCS #1 RSA Encryption
            Public Key:
                Modulus:
                    00:ca:fa:79:98:8f:19:f8:d7:de:e4:49:80:48:e6:2a:2a:86:
                    ed:27:40:4d:86:b3:05:c0:01:bb:50:15:c9:de:dc:85:19:22:
                    43:7d:45:6d:71:4e:17:3d:f0:36:4b:5b:7f:a8:51:a3:a1:00:
                    98:ce:7f:47:50:2c:93:36:7c:01:6e:cb:89:06:41:72:b5:e9:
                    73:49:38:76:ef:b6:8f:ac:49:bb:63:0f:9b:ff:16:2a:e3:0e:
                    9d:3b:af:ce:9a:3e:48:65:de:96:61:d5:0a:11:2a:a2:80:b0:
                    7d:d8:99:cb:0c:99:34:c9:ab:25:06:a8:31:ad:8c:4b:aa:54:
                    91:f4:15
                Public Exponent: 65537 (0x10001)
        Extensions:
            Identifier: Certificate Type
                Critical: no
                Certified Usage:
                    SSL Client
            Identifier: Authority Key Identifier
                Critical: no
                Key Identifier:
                    f2:f2:06:59:90:18:47:51:f5:89:33:5a:31:7a:e6:5c:fb:36:
                    26:c9
    Signature:
        Algorithm: PKCS #1 MD5 With RSA Encryption
        Signature:
            6d:23:af:f3:d3:b6:7a:df:90:df:cd:7e:18:6c:01:69:8e:54:65:fc:06:
            30:43:34:d1:63:1f:06:7d:c3:40:a8:2a:82:c1:a4:83:2a:fb:2e:8f:fb:
            f0:6d:ff:75:a3:78:f7:52:47:46:62:97:1d:d9:c6:11:0a:02:a2:e0:cc:
            2a:75:6c:8b:b6:9b:87:00:7d:7c:84:76:79:ba:f8:b4:d2:62:58:c3:c5:
            b6:c1:43:ac:63:44:42:fd:af:c8:0f:2f:38:85:6d:d6:59:e8:41:42:a5:
            4a:e5:26:38:ff:32:78:a1:38:f1:ed:dc:0d:31:d1:b0:6d:67:e9:46:a8:
             dd:c4
```

Şekil 4.1 Okunabilir Bir Sertifika İçeriği [5]

```
-----BEGIN CERTIFICATE-----
MIICKzCCAZSgAwIBAgIBAzANBgkqhkiG9w0BAQQFADA3MQswCQYDVQQGEwJVUzER
MA8GA1UEChMITmV0c2NhcGUxFTATBgNVBAsTDFNlcHJpeWEncyBDQTAeFw05NzEw
MTgwMTM2MjVaFw05OTEwMTgwMTM2MjVaMEgxCzAJBgNVBAYTAlVTMREwDwYDVQQK
EwhOZXRzY2FwZTENMAsGA1UECxMEUHViczEXMBUGA1UEAxMOU3Vwcml5YSBTaGV0
dHkwgZ8wDQYJKoZIhvcNAQEFBQADgY0AMIGJAoGBAMr6eZiPGfjX3uRJgEjmKiqG
7SdATYazBcABulAVyd7chRkiQ3lFbXFOGD3wNktbf6hRo6EAmM5/RlAskzZ8AW7L
iQZBcrXpc0k4du+2Q6xJu2MPm/8WKuM0nTuvzpo+SGXelmHVChEqooCwfdiZywyZ
NMmrJgaoMa2MS6pUkfQVAgMBAAGjNjA0MBEGCWCGSAGG+EIBAQQEAwIAgDAfBgNV
HSMEGDAWgBTy8gZZkBhHUfWJMloxeuZc+zYmyTANBgkqhkiG9w0BAQQFAAOBgQBt
I6/z07Z635DfzX4XbAFpjlRl/AYwQzTSYx8GfcNAqCqCwaSDKvsuj/vwbf9lo3j3
UkdGYpcd2cYRCgKi4MwqdWyLtpuHAHl8hHZ5uvi00mJYw8W2wU0sY0RC/a/IDy84
hW3WWehBUqVK5SY4/zJ4oTjx7dwNMdGwbWfpRqjdlA==
-----END CERTIFICATE-----
```

Şekil 4.2 64 İkil Sertifika İçeriği [5]

4.2.2 SS Sertifikalarının Güven Kurmak için Kullanılması

SS'ler varlıkları doğrulayan ve sertifika sağlayan kurumlardır. Bağımsız üçüncü parti sertifika sağlayıcılar kullanılabileceği gibi; kurumlar, kurum içerisinde kendi sertifika sağlama yazılımlarını da kullanabilmektedirler. Sertifika destekleyen istemci ve sunucu yazılımlarının güvenilir SS sertifikaları koleksiyonuna sahip olması gerekmektedir. Bu SS sertifikaları yazılımın hangi sertifikalarını onaylayabileceğinin, başka bir deyişle yazılımın hangi sağlayıcıların sertifikalarına güvenebileceğinin bir göstergesidir.

5. DUYARGA AĞLARI

Duyarga ağları; doğal gözlem, sağlık durumunun gözlenmesi, acil sağlık desteği, araç takibi ve askeri uygulamalar benzeri ortamlarda kullanım için öngörülmektedir (Loricz vd. 2004). Bu ortamların tümünde de yetkilendirme, bütünlük, gizlilik ve güvenlik gereksinimlerinin farklı bileşimlerine ihtiyaç duyulmaktadır

Duyarga cihazları sınırlı kapasiteye sahip, ucuz, düşük enerjili, boyut olarak küçük, kısa mesafeler için kablosuz iletişim olanağına sahip cihazlardır. (Akyildiz vd. 2002). Bir duyarga düğümü tipik olarak bir güç ünitesi, bir algılama birimi, bir işleme birimi, bir veri saklama birimi ve bir telsiz alıcı/verici biriminden oluşmaktadır. Duyarga ağı ise, sınırlı enerji, işlem saklama ve iletişim olanaklarına sahip çok sayıda duyarga cihazından oluşmaktadır. (Chong ve Kumar 2003) Duyarga cihazlarının konumlandırıldığı ortam ev, ofis gibi kontrollü bir ortam olabileceği gibi, düşman ya da afet bölgesi, zehirli alanlar gibi kontrolsüz ortamlar da olabilir. Ortam biliniyor ve kontrollü ise, cihazların dağıtımı belirli bir altyapı oluşturacak şekilde elle gerçekleştirilebilir. Ancak, düğüm sayısı arttıkça ve ortam kontrolü kayboldukça elle dağıtım imkansızlaşmaktadır. Bu durumda dağıtım cihazların hedef bölgeye rasgele serpiştirilmesiyle gerçekleştirilmektedir. Belirli bölgelerde cihazların yoğun olmasını sağlamak mümkündür ancak, düğümlerin tam yerini belirlemek mümkün olmamaktadır. Bu durumda topoloji bilgisi hareketli duyarga düğümleri ve kendinden yerleşme protokolleri kullanılarak belirlenebilir (Callaway 2004).

Telsiz duyarga ağlarında güvenlik 6 sorunla karşılaşmaktadır (Ilyas ve Mahgoub 2005)

- İletişimin kablosuz doğası
- Düğümlerin sınırlı kaynakları
- Çok geniş alana dağılmışlık ve farklı yoğunluk
- Sabit altyapı bulunmaması
- Dağıtım öncesi kesin bilinemeyen ağ topolojisi
- Gözetimsiz düğümlerin fiziksel saldırılara açık olması Daha da ötesinde düşman ortamlarda duyarga düğümleri saldırgan ortamlarda işlemek durumundadır.

Bu tür uygulamalar için güvenlik kuvvetli ve etken anahtar dağıtımı mekanizmalarına ihtiyaç duymaktadır. Kontrolsüz ortamlarda çok sayıda düğümü dolaşarak ayarlarını değiştirmek çoğunlukla olanaksızdır. Tüm duyarga ağında paylaşılan tek bir anahtar kullanılması durumunda da düşman ellere geçecek bir cihaz, tüm ağın güvenliğini tehlikeye atacağından,

çözüm olmaktan uzaktır. Bu durumda duyarga cihazlarının bulundukları ortama uyum sağlaması ve güvenliği sağlamak için:

- Önceden verilmiş anahtar ya da benzeri mekanizmaları kullanması
- Yakın komşuları ile bilgi alış verişinde bulunması
- İletişim kurma olanağı bulunan düğümlerle bilgi alış verişinde bulunması gereklidir.

Farklı doğası gereği, duyarga ağlarında güvenliği sağlamak için farklı yöntemlere gereksinim duyulduğu aşikar olmasına rağmen, özellikle şifreleme alanında yeni bir yöntem ortaya atmanın ve bu yöntemin güvenilir olduğunun ispatlanması için geçmesi gereken sürecin uzunluğu nedeniyle geleneksel algoritmalara başvurmaktan başka yol kalmamaktadır. Öte yandan duyarga ağlarında kullanılan cihazların sınırlı kapasiteleri mevcut güvenlik uygulamalarını bu tür ortamlar için uygulanamaz kılmaktadır. Bu tür cihazlar için, çok fazla işlem gücü ya da enerji gerektiren işlemler uygun olmadığından açık anahtar şifreleme algoritmaları güvenliği sağlamak için kullanışlı gözükmemektedir(Malan vd. 2004). Açık anahtar şifreleme, veri aktarım güvenliğinin sağlanması için kullanılamazken, simetrik şifreleme anahtarlarının dağıtılması için uygun bir çözüm olabilir

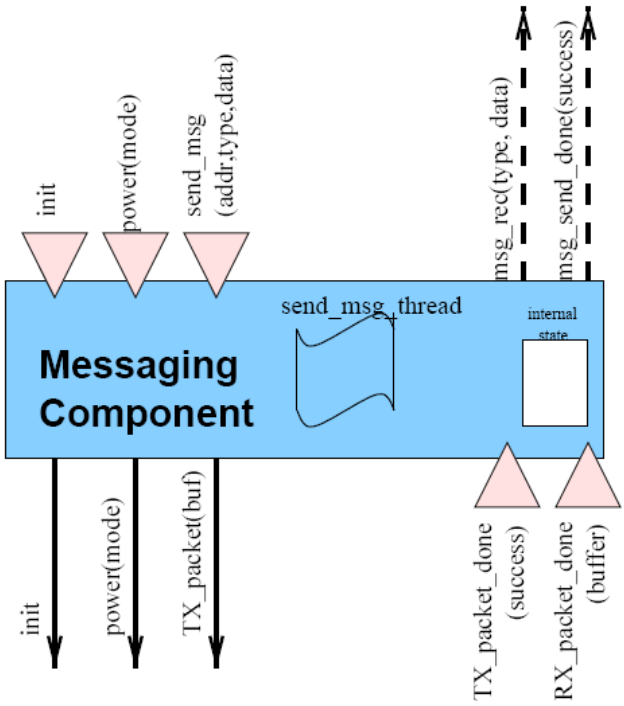
Duyarga ağının her bir cihazının taşıması gereken birçok özellik bulunmaktadır. Belleğinin sınırlı olması, üzerinde çalışan yazılımın çok fazla tampon alan kullanamayacağı anlamına gelmektedir. Bu nedenle talep-yanıt ikilisi yerine yazılımın akışı işleyebilecek yapıda olması gereklidir. Maliyet ve enerji kısıtlamaları nedeniyle cihaz üzerinde bulunabilecek donanım sınırlıdır, dolayısıyla, yazılımın kısıtlı donanım kaynaklarına rağmen yoğun olarak gerçekleşen eş zamanlı işlemlere yanıt verebilmesi gerekmektedir. Ağın bir parçası olan her bir cihazın uygulamalarının birbirinden oldukça farklı olması beklenmektedir, dolayısıyla, cihazların işletim sistemlerinin bir birinden oldukça farklı olan uygulamaları destekleyebilmesi gerekmektedir. Son olarak, sistemin bütününün, bileşenlerindeki bir parçanın sorun çıkartması ya da arızalanması durumunda, işleyişine sorunsuz devam edebilmesi gerekmektedir.

Bu gereksinimleri karşılayabilmek için, Kaliforniya (Berkeley) Üniversitesinde, mikro-görevler destekleyen küçük bir işletim sistemi, TinyOS (Levis vd. 2004), tasarlanmıştır.

5.1 TinyOS

TinyOS, eklenebilir programlardan oluşan, olay güdümlü bir işletim sistemidir. Kısıtlı donanım kaynaklarına rağmen çok sayıda eş zamanlı işlemi karşılamak üzere tasarlanmış ve C dilinde geliştirilmiştir.

TinyOS temel soyutlama düzeyi yazılım bileşenidir. Bileşen bir birinden bağımsız dört parçadan oluşmaktadır: Komut kotarıcısı (handler), olay kotarıcısı, sabit boyutlu çerçeve (frame) ve bir basit görevler demeti. Aşağıdaki şekilde (Şekil 5.1) bileşenin grafik gösterimi verilmiştir. Komutlar, bileşenle arayüzü oluştururlar ve tıkanmasız (non-blocking) olmaları gereklidir. Olaylar, tamamlanmayı bildiren bir mekanizma içerir. Sabit olarak atanan çerçeve, her bir bileşenin durumunu tutar ve atanmanın statik olması, derleme sırasında belleğin yeterli olup olmayacağının belirlemesinin yanı sıra, işletim sırasında bellek yönetimi için ek yük gelmesini önler. Tasarım olarak, görevler sonlandırılmak üzere işletime başlarlar, ancak olaylar tarafından işletim sonu beklenmeden sonlandırılabilirler. Tüm sistem düzenleşimi (configuration) bileşen çizgesi ve basit bir görev düzenleyicisinden oluşmaktadır.



Şekil 5.1 TinyOS Yazılım Bileşeni Yapısı(Levis vd. 2003)

Bileşenler üç sınıfa ayrılmaktadır: Donanım soyutlamaları, sentetik donanım ve üst düzey yazılım. Donanım soyutlaması, fiziksel donanımı bileşen modeline eşlemektedir. Sentetik donanım gerçek donanımın işleyişi benzetimi için kullanılmaktadır. Yazılım bileşeni; kontrol, yol atama ve veri aktarımı işlevlerini gerçekleştirmektedir. Bu bileşen yapısı, yazılım/donanım sınırlamasını kırılmasını sağlayarak taraflar arasında kolay geçişe olanak

imkan tanır. Olay tabanlı model altında işleyen donanımı tamamlayıcı bir yapıdadır (Levis vd. 2003).

TinyOS işletim sisteminde görevler ve olaylar arası etkileşim çift düzeyli düzenleyici olarak bilinmektedir. "Aktif İletiler" olarak bilinen iletişim modeliyle tamamen uyumlu işleyebilmektedir. Bu iletişim modelinin ardındaki fikir, iletişim ilkellerini alttaki donanıma eşlemek ve iletişim ile işleyişi tümleştirmektir. Gecikmeleri asgari düzeye indirmek için üst düzey paralel işlem bağlamında geçekleştirilmiştir. Modelin üst düzeydeki etkenliği ve asgari tampon alan gereksimi sayesinde cihazlar arası iletişim için iyi bir araç haline gelmiştir. Her bir aktif ileti, ulaştığında hedef düğümde uyarılan ve veri alanında parametreleri aktarılan kullanıcı düzeyinde olay kotarıcısı adını içerir. [6]

TinyOS işletim sistemi, aktarınla verileri TinySEC olarak adlandırılmış modülü ile gerçekleştirmektedir. TinySEC hakkında bilgi vermeden önce duyarga ağlarında güvenlik ile ilgili sorunların incelenmesi faydalıdır.

5.2 Duyarga Ağlarında Güvenlik Zaafları

Dinamik olarak değişen topolojisi, belirli bir alt yapının bulunmaması ve merkezi olmayan karakteri nedeniyle, duyarga ağlarında güvenliğin sağlanması oldukça güçtür. Güvenlik, duyarga ağ uygulamalarının her birinde gerçekleştirilmiş bir özellik olmak zorundadır. Duyarga ağlarında çok farklı uygulamalar olduğu dikkate alınırsa, ağların çok fazla açık noktası olduğu rahatça görülebilir. Bu açıkların bazıları aşağıda verilmiştir (Ilyas ve Mahgoub 2005) .

5.2.1 Zayıf Fiziksel Koruma

Klasik ağ uygulamalarında, düğümlerin fiziksel koruması genellikle oldukça kolay ve tam olarak gerçekleştirilebilir. Düğümler, yetkisiz kişilerin giremeyeceği korumalı ortamlarda bulunmaktadır. Öte yandan, duyarga ağlarında, açık arazide bulunan bir düğümün kolayca düşman güçlerin eline geçebileceği aşikardır. Böyle bir düşmanca ortamda mükemmel bir fiziksel koruma sağlamak mümkün değildir. Duyarga ağ sistemi gerçekleştirimi sırasında cihazların tehdit altında olduğu dikkate alınmak zorundadır.

5.2.2 Sınırlı Olanaklar

Duyarga ağ cihazları işlemci gücü, pil ömrü, ve aktarım bant genişliği gibi konular başta olmak üzere sınırlı olanaklara sahiptir. Bu tür kısıtlı kaynaklar servis reddi saldırısına hedef olabilmektedir. İşlemci gücü ve aktarım bant genişliğine karşı servis reddi saldırısı, klasik ağlarda çok yakından bilinmektedir. Duyarga ağlarında cihazlar özellikle pil ömrünün

tüketilmesi şeklinde servis reddi saldırılarına açık bulunmaktadır. Bir düğümün pilinin tüketilmesi, o düğümü sürekli devre dışı bırakmak anlamına gelmektedir.

5.2.3 Birlikte İşleme Zorunluluğu

Duyarga ağında bulunan bir düğüme veri iletebilmek için diğer düğümlerin katılımı zorunludur. Tek bir amaca hizmet eden bir duyarga ağında diğer düğümlerin katılımını sağlamak zor değildir, ancak, özellikle değişik kullanıcılara hitap eden karmaşık ağlarda, kullanıcıların bencilliklerinden dolayı (örneğin pil ömrünü uzatmak, vb. amaçlarla) birlikte işletime katılma konusunda düğümler sorun yaşayabilirler. Bu durumda ağın işleyebilirliği ile kaynakların korunması arasında bir ikilem ortaya çıkmaktadır. Ağın her bir düğümünün işleyişe katkıda bulunmasının sağlanması gerekmektedir.

5.2.4 Kablosuz Ortamların Zaafları

Kablosuz iletişim ortamında aktarılan verilere dışarıdan erişim, verinin bozulması, verinin değiştirilmesi, kablolu ortamlara göre çok daha kolay olmaktadır.

5.2.5 Ağ Katmanı Saldırıları

Duyarga ağlarının dinamik değişen topolojisi nedeniyle, yol atamanın dinamik yapılması gerekmektedir. Bu durum aşağıdaki zafiyetleri ortaya çıkarmaktadır (Karlof ve Wagner 2003):

5.2.5.1 Hatalı Yönlendirme

Bir düğüm, bazı nedenlerden ötürü sadece belirli paketleri ya da sadece belirli düğümlere ait paketleri yönlendirmeyi gerçekleştirirken diğerlerini reddedebilir. Daha da ötesinde diğer düğümlerin yol taleplerine verdikleri yanıtları değiştirerek tüm ağın genel performansını da etkileyebilir.

5.2.5.2 Trafik Sapması

Zararlı bir düğüm yanıltıcı şekilde çekici yol duyuruları yaparak diğer düğümlerin paketleri kendi üzerinden göndermelerini sağlayabilir. Saldırganlar bu yöntemi, bilgi toplamak, ağ yollarını etkilemek ve bazı paketlerin aktarımını önlemek için kullanabilir.

5.2.5.3 Yol Güncellemeleri Taşması

Kısa aralıklarla ağa yol güncellemeleri göndermek ağın aşırı yüklenmesine neden olabilir. Bu servis reddi saldırılarının bir başka yöntemidir.

5.2.5.4 Kara Delik Saldırıları

Hatalı yönlendirme ve trafik sapması saldırılarının birleşiminden oluşur. Belirli bir düğümün paketlerini ele geçirmek isteyen bir saldırgan yanıltıcı yol duyuruları ile bu düğüme en uygun yolun kendi üzerinden geçtiğini duyurur. Bu noktada pek çok düğüm en etkin yol olarak gördükleri için paketlerini bu düğüm üzerinden göndermeye başlayacaklardır. Araya giren düğüm bundan sonra kendine gelen paketleri atar.

5.2.5.5 Gri Delik Saldırıları

Kara delik saldırılarının özel bir biçimidir. Araya giren düğüm paketlerin bazılarını seçici bir şekilde atarken bazılarının aktarımını sağlar.

5.2.5.6 Solucan Yuvası Saldırıları

Bu saldırıda, aralarında özel bir bağlantı olan birden fazla düğüm işin içindedir. Düğümlerden birisi aldığı paketi yönlendirmek yerine özel bağlantı üzerinden solucan yuvasının diğer ucuna gönderir ve diğer uçtaki düğüm paketi yeniden ağa bırakır. Bu şekilde paketlerin yönlendirilmesinde kısa devre yaratılmakta ve muhtemelen oldukça yoğun bir trafik yaratılmaktadır.

5.3 Duyarga Ağlarında Güvenlik Sağlama Yaklaşımları

Duyarga ağ elemanları üzerindeki fiziksel kısıtlamalar yüksek işlem gücüne sahip cihazlar üzerinde işlemek üzere tasarlanmış olan mevcut güvenlik algoritmalarının kullanımını olanaksız kılmaktadır. Örneğin, bir mote cihazının sahip olduğu bellek miktarı şifreleme algoritmalarının gerektirdiği parametreleri tutmak için bile yeterli değildir.

Önemli bir sorun da yetkilendirilmiş verinin tüm ağa yayınlanmasıdır. Yetkilendirilmiş yayın paketleri için mevcut öngörüler, duyarga ağları için kullanışlı olmaktan çok uzaktır. Öncelikle bu öngörüler, yetkilendirme için simetrik sayısal imza yöntemine dayanmaktadır. Kullanışlı olmamasının nedeni, her bir paket için 50 – 1000 sekizli arası ek yük getiren yüksek iletişim yükü ve sayısal imzanın yaratılması/doğrulanması için gereken işlen yüküdür.

Yayın yetkilendirilmesi ayrı bir sorun oluşturmaktadır. Rohatgi imza yöntemi (Rohatgi 1999) gibi tamamen simetrik çözümler bile asgari 300 sekizli gerektirmektedir.

Duyarga ağ elemanlarının en önemli özelliklerinden birisi olan sınırlı enerji kaynağına sahip olmaları, her bir düğümün gerçekleştirebileceği işlem ve iletişimin düzeyini belirler. Enerji kullanımını asgari tutabilmek için güvenlik alt sisteminin işlemci üzerinde mümkün olduğunca az yük oluşturması ve aktarılan her bir mesaja eklediği bilginin asgari tutulması

gerekmektedir. Öte yandan düğümlerin sınırlı ömürleri, kullanılabilir anahtarların ömrünü de kısıtlamaktadır.

Duyarga ağlarında veri güvenliği sağlama yöntemlerinden birisi TinyOS işletim sisteminin bir alt parçası olan TinySEC (Karlof vd. 2004) olarak karşımıza çıkmaktadır.

5.3.1 TinySEC

TinySEC, bir bağlantı katmanı şifreleme mekanizması sağlamaktadır. Çekirdeğini bir blok şifreleme ve anahtarlama mekanizması oluşturmaktadır. TinySEC mevcut olarak, bir grup düğüme dağıtılmış tek bir simetrik anahtar kullanmaktadır. Paket aktarımından önce her bir düğüm paketi şifreler ve veri bütünlüğünü sağlamak üzere bir mesaj yetkilendirme kodu ekler (MAC). TinySEC paket formatı ile TinyOS paket formatı karşılaştırması Şekil 5.2'de verilmiştir. Alıcı bu MACi doğrulayarak mesajın bozulmadan geldiğini anlar ve şifresini çözer. TinySEC üç hedef göz önüne alınarak geliştirilmiştir: Güvenlik, performans ve kullanım kolaylığı.

Hedef (2)	AM (1)	Boy (1)	Kaynak (2)	Sayac (2)	Veri (0..29)	MAC (4)

Hedef (2)	AM (1)	Boy (1)	Grp (1)	Veri (0..29)	CRC (2)

Şekil 5.2 TinySec Paket Formatı ve TinyOS Paket Formatı (Alan genişlikleri sekizli olarak verilmiştir) (Levis vd. 2003)

5.3.1.1 Güvenlik

Bağlantı katmanı güvenlik protokolü olarak TinySEC üç temel güvenlik özelliğini sağlamaktadır; erişim kontrolü, ileti bütünlüğü ve ileti gizdeşliği.

Erişim kontrolü sadece yetkili düğümler ağa katılabilmesi anlamına gelmektedir. Ağın bağlı olan düğümler yetkisiz düğümlerden gelen iletileri ayırt edip reddetme olanağına sahip olmalıdırlar. Yetkilendirilmiş düğümler, paylaşılan anahtara sahip olduğu varsayılarak iletişime dahil edilirler.

İleti bütünlüğü, ileti yetkilendirilmesi ile yakından ilgilidir. Bir ileti aktarım sırasında araya giren bir düğüm tarafından değiştirilmiş ise, alıcının bunu fark edebilmesi gerekir. TinySEC, ileti bütünlüğü kontrolünü paketlerdeki MAC alanı ile denetlemektedir.

İleti gizdeşliği, yetkisiz düğümlerin aktarılan paketlerden hiç bir bilgi elde edememesi anlamına gelmektedir. Bir şifreleme algoritmasının verilerin şifrelenip, şifrelerin açılmasının yanı sıra, şifreli paketlerin kısmi bilgi sağlamasını da önlemesi gerekmektedir. Bu amaçla veri şifreleme sırasında, TinySEC paketinde veri alanından önce gelen alanlar da veri ile birlikte şifrelenerek gönderilir. Veriye eklenen bu değer, paket içerisinde açık olarak da bulunduğu halde, şifreli veri hakkında hiç bir bilgi sağlamazken, aynı verinin iki kere arka arkaya şifrelenmesi durumunda bile farklı şifreli sonuçlar üreterek araya girenlerin aktarılan paketten bilgi elde etmelerinin önüne geçer.

TinySEC TOSSIM benzetim ortamının yanı sıra mica ve mica2 motelarında çalışabilmektedir. TOSSIM, TinyOS işletim sisteminde koşturulmak üzere hazırlanan yazılımların gerçek ortamına aktarılmadan önce sınanması için hazırlanmış benzetim ortamıdır. Sonraki kısımda TOSSIM hakkında detaylı bilgi verilmiştir.

5.4 TOSSIM

TOSSIM (Levis vd. 2003) TinyOS duyarga ağları için ayrık bir olay benzetim ortamıdır. Kullanıcılar TinyOS uygulamalarını mote cihazları yerine kişisel bilgisayarlar üzerinde çalışan TOSSIM sistemine derleyebilirler. Bu, kullanıcılara kontrollü ve yinelenebilir ortamda algoritmaları test etme ve kontrollü işletme olanağı sağlar. TOSSIM kişisel bilgisayarlar üzerinde işletildiği için kullanıcılar geliştirme araçlarını kullanarak TinyOS kodlarını inceleyebilirler.

TOSSIM benzetim ortamının temel amacı, TinyOS uygulamalarının yüksek doğrulukta benzetimini sağlamaktır. Bu nedenle gerçek dünyanın benzetimini sağlamak yerine TinyOS ve işletiminin benzetimine yoğunlaşmaktadır. TOSSIM gerçek dünyada gözlenen davranışların nedenlerini anlamak için kullanılabilir, ancak, tüm davranışları yakalayamayacağından tüm ölçümler için kullanılamaz.

TOSSIM her zaman doğru benzetim ortamı olamayabilir; tüm benzetim ortamları gibi, bazı davranışları kesinleştirip bazılarını basitleştirerek, pek çok varsayımda bulunur. TOSSIM için en önemli soru belirli bir sorun için uygun benzetim ortamını sağlayıp sağlayamayacağının belirlenmesidir. Bu sorunun cevabı TOSSIM ortamının özelliklerinin incelenmesiyle verilebilir:

5.4.1 Doğruluk

TOSSIM TinyOS davranışlarını en alt düzeyde yakalar. Ağ benzetimi ikil seviyesinde gerçekleştirir. Sistemdeki her bir kesilmenin benzetimini sağlar.

5.4.2 Zaman

TOSSIM kesilmeleri hassas bir şekilde zamanlarken, işletim zamanını modellememektedir. TOSSIM bakış açısından kod parçası anlık olarak işletilmektedir. Kod anlık olarak işletildiğinden, kod işletimi tamamlanmadan spin durdurulmasını sağlayacak olay asla gerçekleşmeyecektir.

5.4.3 Modeller

TOSSIM, gerçek dünyayı modellememektedir. Bunun yerine belirli gerçek dünya olaylarının soyutlamalarını sağlamaktadır. Benzetim ortamının haricinde sağlanan araçlarla, kullanıcılar ihtiyaç duydukları modeller için bu soyutlamalardan ihtiyaç duyduklarını kullanabilirler. Karmaşık modelleri benzetim ortamından hariç tutarak, TOSSIM doğru olanı gerçekleştirmek için çalışmak yerine, tüm kullanıcıların ihtiyaçlarını karşılayacak esnekliği sağlamaktadır. İlave olarak benzetim ortamını basit ve etkin kılar.

5.4.4 Radyo

TOSSIM radyo sinyal yayılımını modellemez, bunun yerine iki düğüm arasında yönlendirilmiş bağımsız ikil hatalarından radyo soyutlaması sağlar. Harici bir program istenilen radyo modelini ve ikil hatlarına eşlenmesini sağlayabilir.

5.4.5 Güç/Enerji

TOSSIM çekilen gücü ya da harcanan enerjiyi modellememektedir. Bunun yanında, bileşenlere güç durumunda değişiklik olduğunda bilgi verecek eklemeleri yapmak oldukça kolaydır. Bir benzetim işletildikten sonra kullanıcı bu geçişlere bir güç ya da enerji modeli uygulayarak toplam harcanan enerjiyi belirleyebilir. TOSSIM işlemci işlem süresini modellemediğinden, işlemci tarafından harcanan enerji hakkında tutarlı bir bilgi sağlayamaz.

5.4.6 Yaratma

TOSSIM doğrudan TinyOS kodu kullanarak benzetimi yaratmaktadır. Bir protokol ya da sistemin benzetimi için TinyOS kodunun yazılmış olması gerekmektedir. Bu bir taraftan soyut bir benzetimden zor iken bir taraftan da benzetim sonrası gerçekleştirimin doğrudan motelar üzerinde işletilebileceği anlamına gelmektedir.

5.4.7 Eksiklikler

TOSSIM TinyOS davranışlarını en alt düzeyde yakalamasına rağmen basitleştirmeye yönelik pek çok varsayımda bulunmaktadır. Bu benzetim ortamında işleyen bir kodun gerçek cihazlar üzerinde işlemeyebileceği anlamına gelmektedir. Örneğin, TOSSIM kesilmeleri durdurulabilir değildir. Gerçek bir mote üzerinde ise bir kod parçası çalışırken kesilmeler araya girebilir.

Kodun işletiminin kesilmesi mote işletimini belirsiz bir duruma sokuyor ise, benzetim ortamındaki cihazlar sorunsuz çalışırken, gerçek cihazlar sorun yaşayabilir. Ayrıca kesilme kotarıcılarının işletimi çok uzun sürüyorsa, TOSSIM benzetimlerinde işletim anlık olup sorun çıkartmazken, gerçek cihazlar kilitlenebilir.

5.4.8 İletişim

TOSSIM, MAC, kodlama, zamanlama ve zaman uyumlu geribildirim içeren, 40Kb RFM mica ağ yığıtı benzetimi sağlamaktadır. ChipCon CC1000 yığıtı benzetimi bulunmamaktadır. [7]

6. ELİPTİK EĞRİ GERÇEKLEŞTİRİMLERİ

Eliptik eğrilerin performansının değerlendirilebilmesi için, iki farklı ortamda uygulaması gerçekleştirilmiştir: GSK ve duyarga ağları. Bu amaçla öncelikle bir eliptik eğri kütüphanesi oluşturulmuş, daha sonra bu kütüphane ilgili ortamlara uyarlanmıştır. Aşağıdaki kısımlarda bu gerçekleştirimlerle ilgili detaylı bilgi verilmektedir.

6.1 Eliptik Eğri Kütüphanesi

Bir eliptik eğri gerçekleştirimi için öncelikle yapılması gereken bir takım tercihler olduğu daha önceden açıklanmıştı. Çalışmaya başlarken, belirli bir platformu hedefemeyen yazılım geliştirme amacıyla, algoritmalar ve algoritmalar arası performans farkları incelenip en uygun olanları belirlenmiştir. Bunu gerçekleştirken işlem hızı, bellek alanı vb. sınırlamalara bağlı kalınmamış, farklı gösterimlerdeki farklı eğri türlerinin desteklenmesi hedeflenmiş, özel bir şifreleme donanımı desteği düşünülmemiştir. Ortamdan bağımsız uygulama gerçekleştirimi için geliştirme aracı olarak C programlama dili (ISO/IEC, 1999) ve gnuC derleyicisi tercih edilmiştir.

Öncelikle kullanılacak eğrilerin seçilmesi, bu eğrilere uygun algoritmaların belirlenmesi için gereklidir. Birlikte işleyebilirlik ve standartlara uyum amacıyla rasgele eğri üretmek yerine, SECG (SECG 2000) ve NIST (NIST 1999) tarafından önerilen eliptik eğrilerin kullanılması tercih edilmiştir. Bu eğriler ile ilgili parametreler Ek.1'de verilmiştir. Her iki dokümanda da üç farklı eğri türü parametreleri ve bu eğrilere uygun şifreleme algoritmaları sağlanmıştır. Standart konumunda olan dokümanlar olması itibarıyla şifreleme algoritmaları olarak bu dokümanlarda sağlanan algoritmaların kullanılması uygun görülmüştür. NIST tarafından önerilen eğrilerin bir avantajı da, tanımlanan eğri asal alanlarını oluşturan asal sayıların "Mersenne Asalı" ya da "taklit Mersenne Asalı" (Konstantinou vd. 2003) biçiminde olmasıdır. Mersenne Asalları, EEALP çözümü için bir avantaj sağlamamakla beraber, eğri işlemleri sırasında fayda sağlamaktadır. Eliptik eğriler üzerinde yapılan işlemlere ilişkin algoritmalar aşağıdaki kısımlarda verilmiştir.

6.1.1 Asal Alan Tamsayı Aritmetik İşlemleri

İşlemler çok büyük sayılar üzerinde gerçekleştirildiğinden doğrudan gerçekleştirilmesi mümkün olamamaktadır. Bu durumda sayıların uygun büyüklükte parçalara bölünmesi ve işlemlerin buna göre yapılması gerekmektedir.

6.1.1.1 Modüler Toplama

Modüler toplama algoritması verilen iki sayıyı toplayıp, modülo değerinden büyükse bu değeri toplamdan çıkartarak sonucu vermektedir. Yapılan işlem $c = (a + b) \bmod p$ işlemidir.

Algoritma 6.1 Modüler Toplama (Rosing 1999)
Giriş: Modülo p ve tamsayılar $a, b \in [0, p - 1]$.
Çıkış: $c = (a + b) \bmod p$.
1. $c_0 \leftarrow$ topla(a_0, b_0).
2. i = 1'den $t - 1$'e: $c_i \leftarrow$ eldeli_topla(a_i, b_i).
3. Elde ikili kurulu ise, $c = (c_{t-1}, \ldots, c_2, c_1, c_0)$'den p çıkar.
4. $c \geq p$ *ise* $c \leftarrow c - p$.
5. Döndür(c).

6.1.1.2 Modüler Çıkarma

Modüler çıkartma algoritması verilen ikinci sayıyı birinciden çıkartıp, sonuç negatif çıkarsa modülo değerini ekleyerek sonucu vermektedir. Yapılan işlem $c = (a - b) \bmod p$ işlemidir.

Algoritma 6.2 Modüler Çıkarma (Rosing 1999)
Giriş: Modülo p ve tamsayılar $a, b \in [0, p - 1]$.
Çıkış: $c = (a - b) \bmod p$.
1. $c_0 \leftarrow$ Çıkart(a_0, b_0).
2. i = 1'den $t - 1$'e: $c_i \leftarrow$ eldeli_çıkart(a_i, b_i).
3. Elde ikili kurulu ise, $c = (c_{t-1}, \ldots, c_2, c_1, c_0)$'ye p ekle.
4. Döndür(c).

6.1.1.3 Tamsayı Çarpma

Çarpma işlemi ikil tabanlı olarak değil, işlemi hızlandırabilmek amacıyla t sayıda 32 ikil parçalara bölünen sayıları çarparak 64 ikil parçalar elde ederek sonuca ulaşacak şekilde gerçekleştirilmektedir. Elde edilen sayının modüler değeri daha sonra verilen indirgeme algoritmalarından birisi kullanılarak elde edilmelidir.

Algoritma 6.3 Tamsayı Çarpma (Gura vd. 2004)
Giriş: Tamsayılar $a, b \in [0, p - 1]$.
Çıkış: $c = a \cdot b$.
1. $r_0 \leftarrow 0$, $r_1 \leftarrow 0$, $r_2 \leftarrow 0$.
2. k = 0'dan $2(t - 1)$'e
    2.1 $\{(i, j) \mid i + j = k, 0 \leq i, j < t\}$'nin her elemanı için
        $(uv) = a_i \cdot b_j$.
        $r_0 \leftarrow$ topla(r_0, v), $r_1 \leftarrow$ eldeli_topla(r_1, u), $r_2 \leftarrow$ eldeli_topla($r_2, 0$).
    2.2 $c_k \leftarrow r_0$, $r_0 \leftarrow r_1$, $r_1 \leftarrow r_2$, $r_2 \leftarrow 0$.
3. $c_{2t-1} \leftarrow r_0$.
4. Döndür(c).

6.1.1.4 Kare Alma

Kare alma işlemi modüler çarpma algoritmasının değiştirilmesi ile elde edilmektedir. Kare alma için aşağıda verilen genel kare alma algoritmasının en iyileştirilmiş şeklinin kullanımı

daha uygun bulunmuştur. Karşılaştırma sağlayabilmek amacıyla Gujardo ve Paar (1997) tarafından önerilen kare alma algoritması da gerçekleştirilmiş, genel kare alma algoritmasından daha yüksek performans sağlamadığı için tercih edilmemiştir.

Algoritma 6.4 Klasik Kare Alma (Hankerson vd. 2004)
Giriş: Tamsayı $a \in$ [0, $p-1$].
Çıkış: $c = a^2$.
1. $r_0 \leftarrow 0$, $r_1 \leftarrow 0$, $r_2 \leftarrow 0$.
2. k = 0'dan $2(t-1)$'e
    2.1 $\{(i, j) \mid i + j = k,\ 0 \le i \le j < t\}$'nin her elemanı için
        $(uv) = a_i \cdot a_j$.
        $(i < j)$ ise $(uv) << 1$, $r_2 \leftarrow$ eldeli_topla(r_2, 0).
        $r_0 \leftarrow$ topla(r_0, v), $r_1 \leftarrow$ eldeli_topla(r_1, u), $r_2 \leftarrow$ eldeli_topla(r_2, 0).
    2.2 $c_k \leftarrow r_0$, $r_0 \leftarrow r_1$, $r_1 \leftarrow r_2$, $r_2 \leftarrow 0$.
3. $c_{2t-1} \leftarrow r_0$.
4. Döndür(c).

Algoritma 6.5 Kare Alma (Gujardo ve Paar 1997)
Giriş: Tamsayı $a \in [0, p-1]$.
Çıkış: $c = a^2$.
1. i = 0'dan $2t-1$'e: $c_i \leftarrow 0$.
2. i = 0'dan $t-1$'e
    2.1 $(uv) \leftarrow c_{2i} + a^2 i$, $c_{2i} \leftarrow v$, $C1 \leftarrow u$, $C2 \leftarrow 0$.
    2.2 $j = i + 1$'den $t-1$'e
        $(uv) \leftarrow c_{i+j} + a_i a_j + C1$, $C1 \leftarrow u$, $(uv) \leftarrow v + a_i a_j + C2$, $c_{i+j} \leftarrow v$, $C2 \leftarrow u$.
    2.3 $(uv) \leftarrow C1 + C2$, $C2 \leftarrow u$, $(uv) \leftarrow c_{i+t} + v$, $c_{i+t} \leftarrow v$.
    2.4 $c_{i+t+1} \leftarrow C2 + u$.
3. Döndür(c).

6.1.1.5 İndirgeme

Uygulamada hedef olarak NIST tarafından önerilen eğriler seçildiğinden, NIST eğri parametrelerinin özelliklerini kullanan özelleştirilmiş algoritmaların yanı sıra genel eğrileri destekleyebilmek üzere Barret indirgeme algoritması da kullanılmıştır.

Algoritma 6.6 Barrett İndirgeme (Bosselaers vd. 1994)
Giriş: $b > 3$, p, $k = \lfloor \log_b p \rfloor + 1$, $0 \le x < b^{2k}$, $\mu = \lfloor b^{2k} / p \rfloor$.
Çıkış: $x \bmod p$.
1. q' $\leftarrow \lfloor \lfloor x/b^{k-1} \rfloor \cdot \mu / b^{k+1} \rfloor$, $r \leftarrow (x \bmod b^{k+1}) - (q' \cdot p \bmod b^{k+1})$.
2. $r < 0$ ise $r \leftarrow r + b^{k+1}$.
3. $r \ge p$ iken: $r \leftarrow r - p$.
4. Döndür(r).

Algoritma 6.7 Hızlı Modülo İndirgeme ($p_{192} = 2^{192} - 2^{64} - 1$ için) (Solinas 2000)
Giriş: Tamsayı $c = (c_5, c_4, c_3, c_2, c_1, c_0)$, her bir c_i 64-ikil genişliğinde, $0 \le c < p^2_{192}$.
Çıkış: $c \bmod p_{192}$.
1. 192-ikillik tamsayılar tanımla: $s_1 = (c_2, c_1, c_0)$, $s_2 = (0, c_3, c_3)$, $s_3 = (c_4, c_4, 0)$, $s_4 = (c_5, c_5, c_5)$.
2. Döndür($s_1 + s_2 + s_3 + s_4 \bmod p_{192}$).

6.1.1.6 Ters Alma

Ters alma işlemi için genişletilmiş Öklid algoritmasının (Algoritma 6.22) değiştirilmiş biçimi kullanılmıştır.

Algoritma 6.8 İkili Ters Alma Algoritması (Brown vd. 2001)
Giriş: Asal p, $a \in [1, p-1]$.
Çıkış: $a^{-1} \bmod p$.
1. $u \leftarrow a$, $v \leftarrow p$, $A \leftarrow 1$, $C \leftarrow 0$.
2. $u \neq 0$ iken:
    2.1 u çift iken:
        $u \leftarrow u/2$. A çift ise $A \leftarrow A/2$; değilse $A \leftarrow (A + p)/2$.
    2.2 v çift iken:
        $v \leftarrow v/2$. C çift ise $C \leftarrow C/2$; değilse $C \leftarrow (C + p)/2$.
    2.3 $u \geq v$ ise $u \leftarrow u - v$, $A \leftarrow A - C$; değilse $v \leftarrow v - u$, $C \leftarrow C - A$.
3. Döndür($C \bmod p$).

Aşağıdaki çizelgede (Çizelge 6.1) yukarıda verilen algoritmaların gerçekleştirim performansları gösterilmiştir. Performans değerlendirmesi Intel Pentium III, 700 MHz kişisel bilgisayar ortamında, Red Hat Linux 9 işletim sisteminde gerçekleştirilmiştir.

Çizelge 6.1 NIST Asal Alanları için algoritma performans değerleri (1000 işlem/ms)

	F_{p192}	F_{p224}	F_{p256}	F_{p384}	F_{p521}
Toplama	0,135	0,160	1,722	2,269	2,364
Çıkarma	0,138	0,165	1,751	2,274	2,378
Barret İndirgeme	2,09	2,533	2,614	4,554	7,919
Hızlı İndirgeme	0,128	0,154	0,326	0,457	0,503
Klasik Çarpma*	0,727	0,918	1,452	2,611	4,770
Karatsuba Çarpma*	1,511	1,953	2,912	--	--
Klasik Kare Alma*	0,981	1,283	1,820	3,491	5,201
Kare Alma*	1,221	1,602	2,762	4,519	7,741
Ters Alma	80,328	110,492	148,910	298,95	574,372

* Bu işlemler hızlı indirgeme işlemini de içermektedir.

6.1.2 Asal Alan Eliptik Eğri Nokta İşlemleri

Eliptik eğrilerde noktalar üzerinde yapılan işlemler, bir skaler ile noktanın çarpılması temeline dayanmasına rağmen, tercih edilen nokta gösterim biçimine göre uygulanması gereken algoritma değişebilmektedir. Bunun da ötesinde, bazı durumlarda daha yüksek performans sağlayabilmek için gösterim biçimleri arasında geçişler yapılması da söz konusu olabilmektedir (Cohen vd. 1998). Gösterim biçimleri arasında geçişlerin getireceği ek yüklerden kaçınmak için gösterimler arası dönüşüm kullanılmamıştır. Aşağıdaki çizelgede (Çizelge 6.2) farklı gösterim biçimlerinde nokta toplama ve iki katını alma için gerekli işlem sayısı verilmiştir. Ters alma işleminin diğer işlemlerden 10 kattan çok daha yavaş bir işlem olduğu göz önüne alındığında, doğrusal gösterimin avantajlı olmadığı ortaya çıkmaktadır. Kare alma işlemi de çarpma işlemine göre az da olsa yavaş kaldığından, izdüşümsel gösterim toplama için çok avantajlı görünmesine rağmen, iki katını alma işleminde Jacobian gösteriminden çok geride kalmaktadır. Chudnovsky gösterimi performans olarak Jacobian ve İzdüşümsel gösterimlerin arasında yer almaktadır. Eğri işlemleri arasında iki katını bulma, nokta toplama işleminden daha fazla ihtiyaç duyulan bir işlem olduğu için, bu konuda en iyi performansı sağlayan Jacobian gösterimi tercih edilmiştir.

Çizelge 6.2 Eliptik Eğri Nokta Toplama ve İki Katını Alma için İşlem Sayıları (Brown vd. 2001)

Gösterim	Toplama	2 Katını Bulma
Doğrusal	1 Ters Alma + 2 Çarpma + 1 Kare	1 Ters Alma + 2 Çarpma + 2 Kare
İzdüşümsel	12 Çarpma + 2 Kare Alma	7 Çarpma + 3 Kare Alma
Jacobian	12 Çarpma + 4 Kare Alma	4 Çarpma + 4 Kare Alma
Chudnovsky	11 Çarpma + 3 Kare Alma	5 Çarpma + 4 Kare Alma

6.1.2.1 Nokta Çarpımı

kP çarpımını hesaplamak için kullanılabilecek akla gelen ilk yöntem, k'nın ikillerini soldan sağa doğru incelerken, her bir 1 olan ikil için P noktasını sürekli iki ile çarpılmakta olan ara değere eklemektir. Bu işlemin algoritması aşağıda verilmiştir.

Algoritma 6.9 (Soldan sağa) nokta çarpımı için ikil yöntem (Rosing 1999)
Giriş: $k = (k_{m-1}, \ldots, k_1, k_0)_2$, $P \in E(\mathrm{F}_p)$.
Çıkış: kP.
1. $Q \leftarrow O$.
2. $i = m - 1$'den 0'a
    2.1 $Q \leftarrow 2Q$.

2.2 $k_i = 1$ ise $Q \leftarrow Q + P$.
3. Döndür(Q).

k sayısı daha az sayıda 1 içerecek şekilde ifade edilebilirse, bu durumda yapılacak toplama işlemi sayısı daha az olacağından, çarpma işleminin performansı arttırılabilir. Bu tür gösterime işaretli basamak gösterimi adı verilmektedir ve en etkin yöntemlerden birisi bitişik olmayan biçim (non-adjacent form – NAF) gösterimidir (Hankerson vd. 2000). Bu gösterimde, k (-1), 0 ve (+1)'ler dizisi olarak ifade edilir ve k'nın boyu en fazla bir ikil artmaktadır. Örneğin, 7 sayısı, her birisi 1 olan üç ikil 111 (4 + 2 + 1 = 7)ile ve sadece 2 ikili 0'dan farklı olan 4 ikil 100(-1) (8 + 0 + 0 - 1 = 7) ile gösterilebilir. NAF hesaplama algoritması aşağıda verilmiştir.

Algoritma 6.10 Pozitif bir tamsayının NAF'ını hesaplamak (Miyaji vd 1997).
Giriş: k pozitif tamsayısı.
Çıkış: NAF(k).
1. $i \leftarrow 0$.
2. $k \geq 1$ iken
    2.1 k tek sayı ise: $k_i \leftarrow 2 - (k \bmod 4)$, $k \leftarrow k - k_i$;
    2.2 Değilse: $k_i \leftarrow 0$.
    2.3 $k \leftarrow k/2$, $i \leftarrow i + 1$.
3. Döndür($(k_{i-1}, k_{i-2}, \ldots, k_1, k_0)$).

k sayısının NAF gösterimi NAF(k) şeklindedir ve k sayısının içerdiği 1 sayısını ortalama 1/3'üne indirmektedir (Miyaji vd 1997). Bu şekilde nokta çarpımı sırasında kullanılan toplama işlemlerinin sayısı da aynı oranda azalmaktadır. NAF gösterimi kullanarak nokta çarpımı sağlayan algoritma aşağıda verilmiştir.

Algoritma 6.11 Nokta çarpımı için ikil NAF yöntemi (Miyaji vd 1997)
Giriş: NAF(k) = $\sum_{i=0}^{l-1} k_i 2^i$, $P \in E(\mathrm{F}_p)$.
Çıkış: kP.
1. $Q \leftarrow O$.
2. $i = l - 1$'den 0'a
    2.1 $Q \leftarrow 2Q$.
    2.2 $k_i = 1$ ise $Q \leftarrow Q + P$.
    2.3 $k_i = -1$ ise $Q \leftarrow Q - P$.
3. Döndür(Q).

Yukarıda verilen algoritma, NAF(k), w genişliğinde pencerelere bölünerek ve her seferinde pencere genişliği kadar işlemi bir arada yaparak, hızlandırılabilmektedir. Ancak, bu durumda başlangıçta hesaplanan değerlerin tutulması gereği nedeniyle bellek gereksinimi artmaktadır. Ortamdan bağımsız gerçekleştirim hedefi nedeniyle bu algoritma gerçekleştirilmemiş, ancak burada verilmiştir.

Algoritma 6.12 Nokta çarpımı için pencere NAF yöntemi (Miyaji vd 1997)
Giriş: Pencere genişliği w, NAF$w(k) = \sum_{i=0}^{l-1} k_i 2^i$, $P \in E(\mathrm{F}p)$.
Çıkış: kP.
1. $i \in \{1, 3, 5, ..., 2^{w-1} - 1\}$ için $P_i = iP$ hesapla.
2. $Q \leftarrow O$.
3. $i = l - 1$'den 0'a
    3.1 $Q \leftarrow 2Q$.
    3.2 $k_i \neq 0$ ise:
        $k_i > 0$ ise $Q \leftarrow Q + P_{k_i}$; değilse $Q \leftarrow Q - P_{k_i}$.
4. Döndür(Q).

6.1.2.2 Sabit Nokta Çarpımı

Yukarıda verilen algoritmalar, çarpımı yapılan noktanın önceden bilinmediği durumlar için kullanılmaktadır. Noktanın önceden bilinmesi durumunda bir takım hesaplamalar önceden yapılarak algoritmaların başarımı arttırılabilmektedir. Noktanın önceden bilinmesi durumunda kullanılabilecek farklı iki nokta çarpımı algoritması aşağıda verilmiştir.

Algoritma 6.13 Sabit-taban pencereleme yöntemi (Brown vd. 2001)
Giriş: Pencere genişliği w, $d = \lceil m/w \rceil$, $k = (k_{d-1}, \ldots, k_1, k_0)_{2^w}$, $P \in E(\mathrm{F}_p)$.
Çıkış: kP.
1. *Ön hesaplama*. $Pi = 2^{wi}P$'yi hesapla, $0 \le i \le d - 1$.
2. $A \leftarrow O$, $B \leftarrow O$.
3. $j = 2^w - 1$'den 1'e
    3.1 $k_i = j$ olan her i için: $B \leftarrow B + P_i$. {B'ye Q_j ekle}
    3.2 $A \leftarrow A + B$.
4. Döndür(A).

Algoritma 6.14 İki tablolu sabit-taban tarak yöntemi (Brown vd. 2001)
Giriş: Pencere genişliği w, $d = \lceil m/w \rceil$, $k = (k_{m-1}, \ldots, k_1, k_0)_2$, $P \in E(\mathrm{F}_p)$.
Çıkış: kP.
1. *Ön hesaplama*. $e = \lceil d/2 \rceil$ olsun. Tüm $(a_{w-1}, \ldots, a_1, a_0) \in \{0, 1\}^w$ 'ler için $[a_{w-1}, \ldots, a_0]P$'yi ve $2^e[a_{w-1}, ..., a_0]P$'yi hesapla.
2. Gerekirse k'nin solunu 0 ile doldur, $k = K^{w-1} ||\cdots||K^1||K^0$'ı yaz. Her K^j , d uzunluğunda bir ikil katarıdır. K^j_i , K^j 'nin i. ikilini ifade eder.
3. $Q \leftarrow O$.
4. $i = e - 1$'den 0'a
    4.1 $Q \leftarrow 2Q$.
    4.2 $Q \leftarrow Q + [K_i^{w-1}, \ldots, K_i^1, K_i^0]P + 2^e[K_{i+e}^{w-1}, \ldots, K_{i+e}^1, K_{i+e}^0]P$
5. Döndür(Q).

6.1.3 İkili Alan Aritmetik İşlemleri

F_{2m} alanında eğriler için polinom gösterimi kullanılmakta olduğundan, buradaki işlemlerin de polinom tabanlı olması gerekmektedir ve asal alan işlemlerinden farklılık göstermektedir. Burada asal alan algoritmalarından farklı olanlar açıklanmıştır.

6.1.3.1 Polinom Toplama

Eliptik eğri polinomları vektör olarak tutulmakta olduğundan, toplama işlemi sadece 2 (m-1)-ikil vektörün toplanmasıdır. Gerçekleştirimin hedef aldığı platformun desteklediği kelime genişliğine göre bu vektörlerin kaç kelime olacağı değişmektedir. 32-ikil kelime genişliğinin desteklendiğini varsayarsak, $t = \lceil m/32 \rceil$ adet eldeli toplama işlemi sonuca ulaşmak için yeterli olacaktır.

6.1.3.2 Polinom Çarpımı

Polinomun katsayıları 0 ve 1'lerden oluştuğu için, çarpan polinomlardan birisinin her bir ikili kontrol edilerek, 1 olduğunda diğer çarpan polinom kaydırılıp eklenme suretiyle polinomların çarpımına ulaşılabilir (Algoritma 6.15) Ancak, bu yöntem, çok fazla kaydırma ve toplama gerektirmektedir. Çarpım polinomunu elde ederken önceden elde edilmiş değerlerin daha sonraki elemanların elde edilmesinde kullanılabilmesi durumunda, çarpma işlemi hızlanacaktır. Bu özellikten faydalanan iki algoritma aşağıda verilmiştir. İlk algoritma, ilk çarpanı sağdan sola tararken, diğer algoritma soldan sağa taramaktadır. Her iki algoritmanın başarımı da ilk verilen çarpım algoritmasından yüksektir, ancak, sağdan sola çarpım algoritmasında kaydırılmakta olan B'nin boyutu diğerinde kaydırılan C'nin yarısı olduğundan başarımı artmaktadır.

Algoritma 6.15 Sağdan sola kaydır-ve-ekle alan çarpımı (Rosing 1999)
Giriş: En çok $m - 1$ derecesinden $a(x)$ ve $b(x)$ ikil polinomları.
Çıkış: $c(x) = a(x) \cdot b(x) \bmod f(x)$.
1. $a_0 = 1$ ise $c \leftarrow b$; değilse $c \leftarrow 0$.
2. i= 1'den $m-1$'e
    2.1 $b \leftarrow b \cdot x \bmod f(x)$.
    2.2 $a_i = 1$ ise $c \leftarrow c + b$.
3. Döndür(c).

Algoritma 6.16 Polinom çarpımı için sağdan sola tarak yöntemi (Hankerson vd. 2004)
Giriş: En çok $m - 1$ derecesinden $a(x)$ ve $b(x)$ ikil polinomları.
Çıkış: $c(x) = a(x) \cdot b(x)$.
1. $C \leftarrow 0$.
2. k = 0'dan 31'e
    2.1 j = 0'dan $t - 1$'e
        $A[j]$'nin k. ikili 1 ise $C\{j\}$'e B ekle.
    2.2 $k \neq 31$ ise $B \leftarrow B \cdot x$.
3. Döndür(C).

Algoritma 6.17 Polinom çarpımı için soldan sağa tarak yöntemi (Hankerson vd. 2004)
Giriş: En çok $m - 1$ derecesinden $a(x)$ ve $b(x)$ ikil polinomları.
Çıkış: $c(x) = a(x) \cdot b(x)$.
1. $C \leftarrow 0$.
2. k = 31'den 0'a

2.1 j = 0'dan t − 1'e
$A[j]$'nin k. ikili 1 ise $C\{j\}$'e B ekle.
2.2 $k \neq 31$ ise $C \leftarrow C \cdot x$.
3. Döndür(C).

Yukarıda verilen algoritmalardan Algoritma 6.16'nın başarımı en yüksek olmasına rağmen, Algoritma 6.17, pencere kullanımı ile, bellek sıkıntısı olmadığı durumlarda daha da hızlandırılarak hız başarımı en yüksek çarpma algoritması olarak kullanılabilir. Pencere kullanarak soldan sağa tarama algoritması aşağıda verilmiştir.

Algoritma 6.18 w = 4 pencere genişliğine sahip soldan sağa tarak yöntemi(Hankerson vd. 2004)
Giriş: En çok m – *1* derecesinden $a(x)$ ve $b(x)$ ikil polinomları.
Çıkış: $c(x) = a(x) \cdot b(x)$.
1. En çok 3 dereceli tüm $u(x)$ polinomları için $B_u = u(x) \cdot b(x)$'i hesapla.
2. $C \leftarrow 0$.
3. k = 7'den 0'a
3.1 j = 0'dan t − 1'e
$u = (u_3, u_2, u_1, u_0)$ ve u_i, $A[j]$'nin $(4k + i)$. ikili olsun. $C\{j\}$'e B_u ekle.
3.2 $k \neq 0$ ise $C \leftarrow C \cdot x^4$.
4. Döndür(C).

6.1.3.3 Kare Alma

İkili polinomlarda kare alma işlemi, iki polinomun çarpımından çok daha hızlıdır. Kare alma işleminde yapılan, değeri x olan her bir ikili 0x şeklinde yazmaktan ibarettir. Bu işlemi yapan algoritma aşağıda verilmiştir.

Algoritma 6.19 Kare Alma (Schroeppel 1995)
Giriş: $a \in F_{2m}$.
Output: $a^2 \bmod f(x)$.
1. *Ön hesaplama*. Her $v = (v_7, \ldots, v_1, v_0)$ sekizlisi için, 16-ikil $T(v) = (0, v_7, \ldots, 0, v_1, 0, v_0)$ değerini hesapla.
2. i =0'dan t −1'e
2.1 Her u_j bir sekizli, $A[i] = (u_3, u_2, u_1, u_0)$ olsun.
2.2 $C[2i] \leftarrow (T(u_1), T(u_0))$, $C[2i+1] \leftarrow (T(u_3), T(u_2))$.
3. $b(x) = c(x) \bmod f(x)$'i hesapla.
4. Döndür(b).

6.1.3.4 İndirgeme

İlgili polinomu, her seferinde bir ikil indirgeyerek, indirgenemeyecek hale getirmek hem basit hem de performans açısından tatmin edicidir. NIST eğrileri de dahil olmak üzere bir takım eğriler için her seferinde bir ikil yerine bir kelime indirgeme yapmak daha yüksek başarım sağlayabilmektedir. Ancak, bu algoritma her bir eğri indirgeme polinomu için özelleştirilmek durumundadır. Aşağıda (Algoritma 6.21) bir örneği verilmiştir.

Algoritma 6.20 Modüler İndirgeme (her seferinde bir ikil) (Hankerson vd. 2000)
Giriş: En çok $2m-2$ derecesinden bir ikil $c(x)$ polinomu.
Çıkış: $c(x) \bmod f(x)$.
1. *Ön hesaplama*. $u_k(x) = x^k r(x)$'i hesapla $(0 \le k \le 31)$.
2. $i = 2m-2$'den m'ye
    2.1 $c_i = 1$ ise
        $j = \lfloor (i-m)/32 \rfloor$ *ve* $k = (i-m) - 32j$ olsun.
        $C\{j\}$'e $u_k(x)$'i ekle.
3. Döndür((*C*[*t* − 1], …, *C*[1], *C*[0])).

Algoritma 6.21 Modüler indirgeme (her seferinde bir kelime) (Hankerson vd. 2000)
Giriş: En çok 324 derecesinden bir ikil $c(x)$ polinomu.
Çıkış: $c(x) \bmod f(x)$, $f(x) = x^{163} + x^7 + x^6 + x^3 + 1$.
1. i = 10'dan 6'ya {*C*[*i*] modulo $f(x)$'i indirge}
    1.1 $T \leftarrow C[i]$.
    1.2 $C[i-6] \leftarrow C[i-6] \oplus (T << 29)$.
    1.3 $C[i-5] \leftarrow C[i-5] \oplus (T << 4) \oplus (T << 3) \oplus T \oplus (T >> 3)$.
    1.4 $C[i-4] \leftarrow C[i-4] \oplus (T >> 28) \oplus (T >> 29)$.
2. $T \leftarrow C[5]$ AND 0xFFFFFFF8. {*C*[5]'in 0, 1 ve 2. ikillerini temizle}
3. $C[0] \leftarrow C[0] \oplus (T << 4) \oplus (T << 3) \oplus T \oplus (T >> 3)$.
4. $C[1] \leftarrow C[1] \oplus (T >> 28) \oplus (T >> 29)$.
5. $C[5] \leftarrow C[5]$ AND 0x00000007. {*C*[5]'in kullanılmayan ikillerini temizle}
6. Döndür((*C*[5], *C*[4], *C*[3], *C*[2], *C*[1], *C*[0])).

6.1.3.5 Ters Alma

Ters alma işlemi için genişletilmiş Öklid algoritması kullanılmıştır. Ters alma için "neredeyse tersi" (Almost Inverse) (Algoritma 6.23) ya da geliştirilmiş biçimi (Algoritma 6.24) de kullanılabilir, ancak bu algoritmalar sadece sınırlı durumlarda daha iyi performans sağlayabilmektedirler. Bu nedenle genişletilmiş Öklid algoritması tercih edilmiştir.

Algoritma 6.22 F_2m'de tersini alma için genişletilmiş Öklid Algoritması (Hnakerson vd. 2000)
Giriş: $a \in F_{2m}$, $a \neq 0$.
Çıkış: $a^{-1} \bmod f(x)$.
1. $b \leftarrow 1$, $c \leftarrow 0$, $u \leftarrow a$, $v \leftarrow f$.
2. derece(u) ≠ 0 iken
    2.1 $j \leftarrow$ derece(u) − derece(v).
    2.2 $j < 0$ ise $u \Leftrightarrow v$, $b \Leftrightarrow c$, $j \leftarrow -j$.
    2.3 $u \leftarrow u + x^j v$, $b \leftarrow b + x^j c$.
3. Döndür(b).

Algoritma 6.23 F_2m'de tersini alma için yaklaşık tersini alma algoritması (Schroeppel 1995)
Giriş: $a \in F_{2m}$, $a \neq 0$.
Output: $b \in F_{2m}$ ve $k \in [0, 2m-1]$ öyle ki $ba \equiv x^k \pmod{f(x)}$.
1. $b \leftarrow 1$, $c \leftarrow 0$, $u \leftarrow a$, $v \leftarrow f$, $k \leftarrow 0$.
2. x, u'yu böldüğü sürece
    2.1 $u \leftarrow u/x$, $c \leftarrow cx$, $k \leftarrow k+1$.

3. $u = 1$ ise döndür(b, k).
4. derece(u) < derece(v) ise $u \Leftrightarrow v$, $b \Leftrightarrow c$.
5. $u \leftarrow u + v$, $b \leftarrow b + c$.
6. 2. adıma git.

Algoritma 6.24 F_2m'de tersini alma için değiştirilmiş yaklaşık tersini alma algoritması (Hankerson vd. 2004)
Giriş: $a \in F_{2m}$, $a \neq 0$.
Çıkış: $a^{-1} \bmod f(x)$.
1. $b \leftarrow 1$, $c \leftarrow 0$, $u \leftarrow a$, $v \leftarrow f$.
2. x, u'yu böldüğü sürece
    2.1 $u \leftarrow u / x$.
    2.2 x, b'yi bölerse $b \leftarrow b/x$; bölmezse $b \leftarrow (b + f) / x$.
3. $u = 1$ ise döndür(b).
4. derece(u) < derece(v) ise $u \Leftrightarrow v$, $b \Leftrightarrow c$.
5. $u \leftarrow u + v$, $b \leftarrow b + c$.
6. 2.adıma git.

Aşağıdaki çizelgede (Çizelge 6.1) yukarıda verilen algoritmaların gerçekleştirim performansları gösterilmiştir. Performans değerlendirmesi Intel Pentium III, 700 MHz kişisel bilgisayar ortamında, Red Hat Linux 9 işletim sisteminde gerçekleştirilmiştir.

Çizelge 6.3 İkili Alan Eğrileri için Algoritma Performans Değerleri (1000 işlem/ms)

	m=163	m=233	m=283	m=409	m=571
Toplama	0,058	0,071	0,078	0,122	0,161
Modüler İndirgeme	0,103	0,129	0,196	0,354	0,919
Kaydır ve ekle*	9,51	14,637	19,823	37,956	51,234
Sağdan Sola Tarak*	3,627	7,046	8,75	14,645	20,12
Soldan Sağa Tarak*	4,931	7,258	9,015	16,842	23,71
Pencereli Tarak*	1,911	3,204	3,61	5,921	7,091
Karatsuba Çarpma*	2,571	4,353	4,547	--	--
Kare Alma*	0,233	0,328	0,448	0,608	0,861
Genişletilmiş Öklid	18,85	29,031	39,13	74,833	98,109
Yaklaşık Ters Alma	25,922	36,54	57,095	104,709	137,052
G. Yaklaşık Ters Alma	24,835	33,692	51,881	95,968	127,354

6.1.4 İkili Alan Eliptik Eğri Nokta İşlemleri

Asal alan eğrilerinde olduğu gibi, ikili alan eliptik eğrilerinde de tercih edilen nokta gösterim biçimine göre uygulanması gereken algoritma değişebilmektedir. Aşağıdaki çizelgede (Çizelge 6.4) farklı gösterim biçimlerinde nokta toplama ve iki katını alma için gerekli işlem sayısı verilmiştir.

Kare almanın maliyeti çok düşük olduğu için çizelgeye yansıtılmamıştır. Çizelge incelendiğinde en avantajlı gösterim biçiminin Chudnovsky gösterimi olduğu ortaya çıkmaktadır. Ancak, asal alan nokta gösteriminde tercih edilen Jacobian gösterim ile çok fazla bir performans farkı bulunmadığından uyumlu olması açısından Jacobian gösteriminin kullanılması tercih edilmiştir.

Çizelge 6.4 Eliptik Eğri Nokta Toplama ve İki Katını Alma için İşlem Sayıları (Hankerson 2000)

Gösterim	Toplama	2 Katını Bulma
Doğrusal	1 Ters Alma + 2 Çarpma	1 Ters Alma + 2 Çarpma
İzdüşümsel	13 Çarpma	7 Çarpma
Jacobian	14 Çarpma	5 Çarpma
Chudnovsky	14 Çarpma	4 Çarpma

6.1.4.1 Nokta Çarpımı

Asal alan nokta çarpımında anlatılan algoritmalara ilave olarak ikili alan eğrileri nokta çarpımı için Montgomery nokta çarpımı algoritması kullanılabilmektedir. Montgomery algoritması (Koç ve Acar 1998), noktanın x ve y koordinatlarının ayrı ayrı hesaplanmasına dayanmakta ve işlem sayısını azaltmaktadır.

Algoritma 6.25 Montgomery Nokta Çarpımı (Koç ve Acar 1998)
Giriş: $k = (k_{t-1}, \ldots, k_1, k_0)_2$ ($k_{t-1} = 1$), $P = (x, y) \in E(\mathrm{F}_{2m})$.
Çıkış: kP.
1. $X_1 \leftarrow x$, $Z_1 \leftarrow 1$, $X_2 \leftarrow x^4 + b$, $Z_2 \leftarrow x^2$. {(P, $2P$)'yi hesapla}
2. $i = t - 2$'den 0'a
    2.1 $k_i = 1$ ise
        $T \leftarrow Z_1$, $Z_1 \leftarrow (X_1Z_2 + X_2Z_1)^2$, $X_1 \leftarrow xZ_1 + X_1X_2TZ_2$.
        $T \leftarrow X_2$, $X_2 \leftarrow X^4{}_2 + bZ^4{}_2$, $Z_2 \leftarrow T^2Z^2{}_2$.
    2.2 Değilse
        $T \leftarrow Z_2$, $Z_2 \leftarrow (X_1Z_2 + X_2Z_1)^2$, $X_2 \leftarrow xZ_2 + X_1X_2Z_1T$.
        $T \leftarrow X_1$, $X_1 \leftarrow X^4{}_1 + bZ^4{}_1$, $Z_1 \leftarrow T^2Z^2{}_1$.
3. $x_3 \leftarrow X_1 / Z_1$.
4. $y_3 \leftarrow (x + X_1 / Z_1)[(X_1 + xZ_1)(X_2 + xZ_2) + (x^2 + y)(Z_1Z_2)](xZ_1Z_2)^{-1} + y$.
5. Döndür((x_3,y_3)).

6.1.4.2 Koblitz Eğrileri için Nokta Çarpımı

Koblitz eğrilerinin ikili alan eliptik eğrilerinin özel bir biçimi olduğu daha önceden açıklanmıştı. Eliptik eğri işlemleri arasında en sık kullanılan işlem olan nokta çarpımı için Koblitz eğrilerinde işlemleri hızlandıran yöntemler bulunmaktadır. Bunlardan bir tanesi t-adic NAF (TNAF) gösterimi (Solinas 2000) kullanarak nokta çarpımı yapmaktır. TNAF gösteriminin elde edilmesi için kullanılan algoritma aşağıda verilmiştir.

Algoritma 6.26 $Z[\tau]$'deki bir elemanın TNAF'ının hesaplanması (Solinas 2000)
Giriş: $\kappa = r_0 + r_1\tau \in Z[\tau]$
Çıkış: TNAF(κ).
1. $i \leftarrow 0$.
2. $r_0 \neq 0$ veya $r_1 \neq 0$ iken
    2.1 r_0 tek ise: $u_i \leftarrow 2 - (r_0 - 2r_1 \bmod 4)$, $r_0 \leftarrow r_0 - u_i$;
    2.2 değilse: $u_i \leftarrow 0$.
    2.3 $t \leftarrow r_0$, $r_0 \leftarrow r_1 + \mu r_0 / 2$, $r_1 \leftarrow -t / 2$, $i \leftarrow i+ 1$.
3. Döndür($(u_{i-1}, u_{i-2}, \ldots, u_1, u_0)$).

Ancak, burada elde edilen TNAF(k), k'nın iki katı uzunluğa sahiptir. $\rho = k \bmod \delta$ ise, $\rho P = kP$ olacağından, TNAF(k) yerine kullanıldığında azami k ikil uzunluğunda değerlerle işlem yapılmış olur. TNAF(ρ) elde etme algoritması aşağıda verilmiştir.

Algoritma 6.27 Kısmi modülo δ indirgeme (Solinas 2000)
Giriş: $k \in [1, n-1]$, $C \geq 2$, $s_0 = d_0 + \mu\upsilon d_1$, $s_1 = -d_1$, $\delta = d_0 + d_1\tau$.
Çıkış: $\rho' = k \bmod \delta$.
1. $k' \leftarrow \lfloor k / 2^{a-C+(m-9)/2} \rceil$.
2. i = 0'dan 1'e
    2.1 $g' \leftarrow s_i \cdot k'$, $j' \leftarrow V_m \cdot \lfloor g' / 2^m \rceil$, $\lambda_i \leftarrow \lfloor (g' + j') / 2^{(m+5)/2} + \tfrac{1}{2} \rfloor / 2^C$.
    2.2 $f_i \leftarrow \lfloor \lambda_i + \tfrac{1}{2} \rfloor$, $\eta_i \leftarrow \lambda_i - f_i$, $h_i \leftarrow 0$.
3. $\eta \leftarrow 2\eta_0 + \mu\eta_1$.
4. $\eta \geq 1$ ise
    4.1 $\eta_0 - 3\mu\eta_1 < -1$ ise $h_1 \leftarrow \mu$; değilse $h_0 \leftarrow 1$.
    Değilse
    4.2 $\eta_0 + 4\mu\eta_1 \geq 2$ ise $h_1 \leftarrow \mu$.
5. $\eta < -1$ ise
    5.1 $\eta_0 - 3\mu\eta_1 \geq 1$ ise $h_1 \leftarrow \mu$; değilse $h_0 \leftarrow 1$.
    Değilse
    5.2 $\eta_0 + 4\mu\eta_1 < 2$ ise $h_1 \leftarrow \mu$.
6. $q_0 \leftarrow f_0 + h_0$, $q_1 \leftarrow f_1 + h_1$, $r_0 \leftarrow k - (s_0 + \mu s_1)q_0 - 2s_1q_1$, $r_1 \leftarrow s_1q_0 - s_0q_1$.
7. Döndür($r_0 + r_1\tau$).

TNAF(ρ) kullanılarak nokta çarpımı, Algoritma 6.28'de gösterilmiştir.

Algoritma 6.28 Nokta çarpımı için TNAF yöntemi
Giriş: TNAF(ρ') = $\sum_{i=0}^{l-1} u_i\tau^i$, $\rho' = k \bmod \delta$, $P \in E_a(F_{2m})$.
Çıkış: kP.

1. $Q \leftarrow O$.
2. $i = l - 1$'den 0'a
   2.1 $Q \leftarrow \tau Q$.
   2.2 $u_i = 1$ ise $Q \leftarrow Q + P$.
   2.3 $u_i = -1$ ise $Q \leftarrow Q - P$.
3. Döndür(Q).

TNAF(ρ) kullanarak nokta çarpımı pencere kullanımı ile daha fazla bellek kullanarak daha yüksek başarım sağlayacak hale getirilebilir. Pencereli TNAF(ρ) algoritması ve bunu kullanan çarpma algoritması aşağıda verilmiştir.

Algoritma 6.29 $Z[\tau]$'deki bir elemanın w genişliğinde TNAF'ının hesaplanması (Solinas 2000)
Giriş: w, t_w, $u \in \{1, 3, \ldots, 2^{w-1} - 1\}$ için $\alpha_u = \beta_u + \gamma_u \tau$, $\rho = r_0 + r_1 \tau \in \square Z[\tau]$.
Çıkış: TNAF$_w(\rho)$.
1. $i \leftarrow 0$.
2. $r_0 \neq 0$ veya $r_1 \neq 0$ olduğu sürece
   2.1 r_0 tek ise
      $u \leftarrow r_0 + r_1 t_w \bmod 2^w$.
      $u > 0$ ise $s \leftarrow 1$; değilse $s \leftarrow -1$, $u \leftarrow -u$.
      $r_0 \leftarrow r_0 - s\beta_u$, $r_1 \leftarrow r_1 - s\gamma_u$, $u_i \leftarrow s\alpha_u$.
   2.2 Değilse: $u_i \leftarrow 0$.
   2.3 $t \leftarrow r_0$, $r_0 \leftarrow r_1 + \mu r_0 / 2$, $r_1 \leftarrow -t / 2$, $i \leftarrow i + 1$.
3. Döndür($(u_{i-1}, u_{i-2}, \ldots, u_1, u_0)$).

Algoritma 6.30 Nokta çarpımı için pencere TNAF yöntemi (Solinas 2000)
Giriş: TNAF$_w(\rho') = \sum_{i=0}^{l-1} u_i \tau^i$, $\rho' = k \bmod \delta$, $P \in E_a(F_{2m})$.
Çıkış: kP.
1. $u \in \{1, 3, 5, \ldots, 2^{w-1} - 1\}$ için $P_u = \alpha_u P$'yı hesapla.
2. $Q \leftarrow O$
3. $i = l - 1$'den 0'a
   3.1 $Q \leftarrow \tau Q$.
   3.2 $u_i \neq 0$ ise:
      u öyle ki, $\alpha_u = u_i$ *veya* $\alpha_{-u} = -u_i$ olsun.
      $u > 0$ ise $Q \leftarrow Q + P_u$;
      Değilse $Q \leftarrow Q - P_{-u}$.
4. Döndür(Q).

6.1.4.3 Sabit Nokta Çarpımı

Noktanın önceden bilinmesi durumunda, ikili eğri nokta çarpımı için Algoritma 6.30 'da verilen algoritmanın ilk satırındaki hesaplamanın önceden yapılarak saklanması algoritma işleyişinde oldukça performans arttırıcı olacaktır.

6.1.5 Şifreleme Algoritmaları

Eliptik eğri kullanan şifreleme algoritmalarından EEDH ve EESİA gerçekleştirimleri için SEC1 (SECG, 2000a) ve ANSI X9.62 (ANSI 1998) standardı dokümanlarında verilen algoritmalar tercih edilmiştir.

6.1.5.1 EEDH Algoritmaları

Şifreleme algoritmalarının detayına girmeden önce EEŞ sistemlerinde kullanılan eliptik eğri parametrelerini hatırlamak faydalı olacaktır. Asal alan eğrileri ve ikili alan eğrileri için farklı alan parametreleri bulunmaktadır:

F_p üzerinde eliptik eğri alan parametreleri bir altılıdan oluşmaktadır:

$$T\ (p, a, b, G, n, h)$$

Burada;

p : Sonlu alan F_p'yi belirleyen tamsayı,

a,b : $y^2 \equiv x^3 + ax + b \pmod p$ denklemi ile tanımlanan eğriyi belirleyen elemanlar,

G : $G = (x_G, y_G)$ taban noktası,

n : G noktasının derecesini veren asal sayı

h : $\#E(F_p) / n$

F_{2^m} üzerinde eliptik eğri alan parametreleri bir yediliden oluşmaktadır:

$$T\ (m, f(x), a, b, G, n, h)$$

Burada;

m : Sonlu alan F_{2^m}'yi belirleyen tamsayı,

$f(x)$: m dereceli F_{2^m}'yi simgeleyen indirgenemez polinom

a,b : $y^2 + xy \equiv x^3 + ax^2 + b$ denklemi ile tanımlanan eğriyi belirleyen elemanlar,

G : $G = (x_G, y_G)$ taban noktası,

n : G noktasının derecesini veren asal sayı

h : $\#E(F_{2^m}) / n$

Algoritma 6.31 EEDH Anahtar Yaratma (SECG 2000a)
Giriş : Geçerli EE Parametreleri T = (p, a, b, G, n, h) ya da (m, f(x), a, b, G, n, h)
Çıkış: T ile eşlenmiş, eliptik eğri anahtar çifti.

1. Rasgele d ε (1,n-1) belirle
2. Q = dG hesapla
3. Döndür (d,Q)

Algoritma 6.32 EEDH Anahtar Anlaşma (SECG 2000a)
Giriş : 1.Geçerli EE Parametreleri T = (p, a, b, G, n, h) ya da (m, f(x), a, b, G, n, h),
2. Kendi gizli anahtarı e,
3. Karşı tarafın açık anahtarı P = dQ,
4. Üzerinde anlaşılmış, ortak nokta Q.
Çıkış: e, Q ile eşlenmiş, açık anahtar
1. E = e·Q; D = e·P hesapla, .
2. Döndür (E,D).

Algoritma 6.33 EEDH şifreleme ve çözme
Giriş : 1. Geçerli EE Parametreleri T = (p, a, b, G, n, h) ya da (m, f(x), a, b, G, n, h)
2. M şifreli metin.
3. e gizli anahtarı.
Çıkış: M açık metin
1. M = e·M'.
2. Döndür (M')

6.1.5.2 EESİA Algoritmaları

EESİA, ANSI (1998) tarafından standardı yayımlanan bir sayısal imza yöntemi olduğu için bu dokümanda verilen algoritmaların kullanımı tercih edilmiştir.

Algoritma 6.34 EESİA İmza Yaratma (ANSI 1998)
Giriş : 1. Geçerli EE Parametreleri T = (p, a, b, G, n, h) ya da (m, f(x), a, b, G, n, h),
2. Gizli anahtar d ve imzalanacak ileti M
Çıkış: İmza (r,s)
1. Rasgele k ε (1,n-1) belirle
2. kG = (x,y) hesapla
3. r = x mod n hesapla
4. $s= k^{-1}(Hash(e) + d.r)$ hesapla.
5. Döndür (r,s)

Algoritma 6.35 EESİA İmza Onaylama (ANSI 1998)
Giriş : 1. Geçerli EE Parametreleri T = (p, a, b, G, n, h) ya da (m, f(x), a, b, G, n, h),
2. Açık anahtar Q ve imzalanmış metin M ve imza (r,s)
Çıkış: Geçerli/Geçersiz
1. e = Hash (M)
2. $w = s^{-1}$ mod n
3. u1 = ew mod n ve u1 = rw mod n hesapla
4. X = u1G+ u2Q = (x, y)
5. v = x mod n
6. v = r ise döndür (Geçerli) değilse döndür (geçersiz)

EESİA imza onay algoritmasının (Algoritma 6.35) 4. satırında iki noktanın çarpımı toplanmaktadır. Algoritma 6.36, çarpımların ayrı ayrı yapılıp daha sonra sonuçların toplanması yerine tüm işlemi birden gerçekleştirerek, performansı yükseltmektedir.

Algoritma 6.36 Eş zamanlı çoklu nokta çarpımı (Hankerson vd. 2004)
Giriş: Pencere genişliği w, $k = (k_{m-1}, \ldots, k_1, k_0)_2$, $l = (l_{m-1}, \ldots, l_1, l_0)_2$, $P, Q \in E(\mathrm{F}p)$.
Çıkış: $kP + lQ$.
1. Tüm $i, j \in [0, 2^w - 1]$ için $iP + jQ$'yu hesapla.
2. $k = (k^{d-1}, \ldots, k^1, k^0)$ ve $l = (l^{d-1}, \ldots, l^1, l^0)$'ı yaz. Her k^i ve l^i, w uzunluğunda bir ikil katarıdır. $d = \lceil t/w \rceil$.
3. $R \leftarrow O$.
4. $i = d - 1$'den 0'a: $R \leftarrow 2^w R$, $R \leftarrow R + (k^i P + l^i Q)$.
5. Döndür(R).

6.2 Güvenli Soket Katmanı'nda Eliptik Eğri Şifreleme Gerçekleştirimi

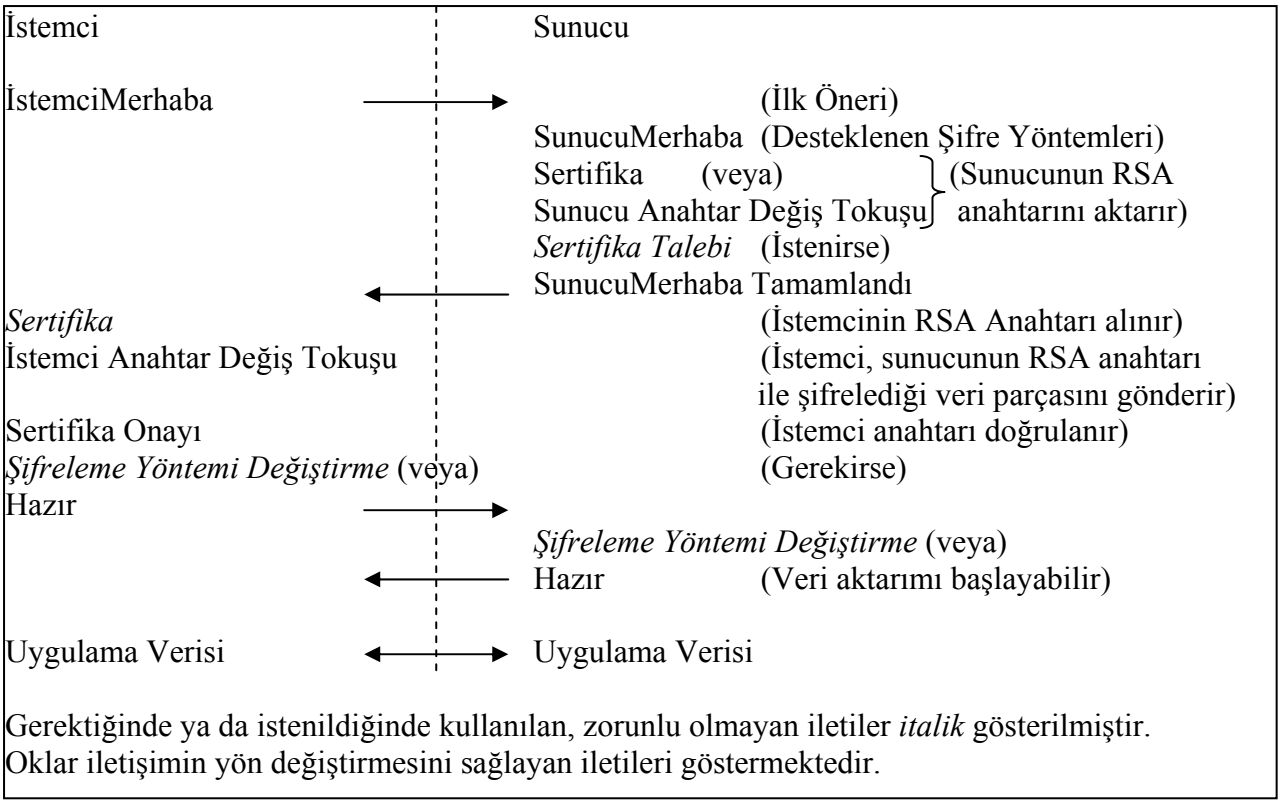
GSK'de EEŞ performansını değerlendirebilmek için, yukarıda gerçekleştirimi anlatılan EEŞ kütüphanesi, açık kod GSK gerçekleştirimi olan OpenSSL [8] yazılımına uyarlanmıştır. Gerçekleştirimin hazırlandığı tarihte tam işlevsel sürüm olan OpenSSL0.9.6.b kullanılmıştır.

EEŞ, açık anahtar şifreleme mekanizması olduğundan sadece Tokalaşma Protokolü etkilenmekte, Kayıt Protokolü işleyişinde değişiklik olmamaktadır. RSA tabanlı ve EEŞ tabanlı Tokalaşma Protokolü işleyişleri arasındaki farklar aşağıda kısaca açıklanmıştır.

6.2.1 RSA Tabanlı Tokalaşma

Error! Reference source not found.'de RSA tabanlı GSK anlaşma işleyişi gösterilmiştir. Bu anlaşma türünde istemci ve sunucu ilk önce rasgele oluşturdukları veri parçacıklarını gönderir (tekrar –replay– saldırısından korunmak için) ve İstemciMerhaba ve SunucuMerhaba iletileri ile şifre grubu üzerinde anlaşma sağlarlar. Sunucu, imzalı RSA açık anahtarını Sertifika ya da SunucuAnahtarDeğişimi iletilerinden birisi ile gönderir. Sunucunun RSA anahtarını doğrulamak için istemci RSA açık anahtar işlemi gerçekleştirir. Ardından istemci, rasgele ürettiği 48 ikil bir sayıyı (öncül sır) sunucunun açık anahtarını kullanarak şifreler ve İstemci Anahtar Değişimi iletisi içerisinde gönderir. Sunucu kendi gizli anahtarını kullanarak öncül şifreyi çözer. Her iki uç, öncül anahtarları kullanarak asıl anahtarı oluşturur. Daha önceden aktarılan anlamsız veriler kullanılarak Kayıt Katmanı tarafından veri şifrelemede kullanılacak olan şifre anahtarlarını ve MAC (Message Authentication Code – İleti Aslıyla Aynılığını Kanıtlama Kodu) belirlenir. Sunucu, gerekirse, kabul edilebilir sertifikaların ve tanınan SS'lerin listesini içeren Sertifika Talebi iletisi göndererek istemcinin aslıyla aynılığını kanıtlamasını talep edebilir. Bu durumda, istemci yukarıda açıklanan işlemleri gerçekleştirerek İstemci Sertifikası içerisinde RSA açık anahtarını ve karşılık gelen gizli

anahtara sahip olduğunu gösteren şifrelemiş olduğu veri parçasını gönderir. Bu imzanın yaratılması da istemcinin bir RSA işlemi yapmasını gerektirmektedir.



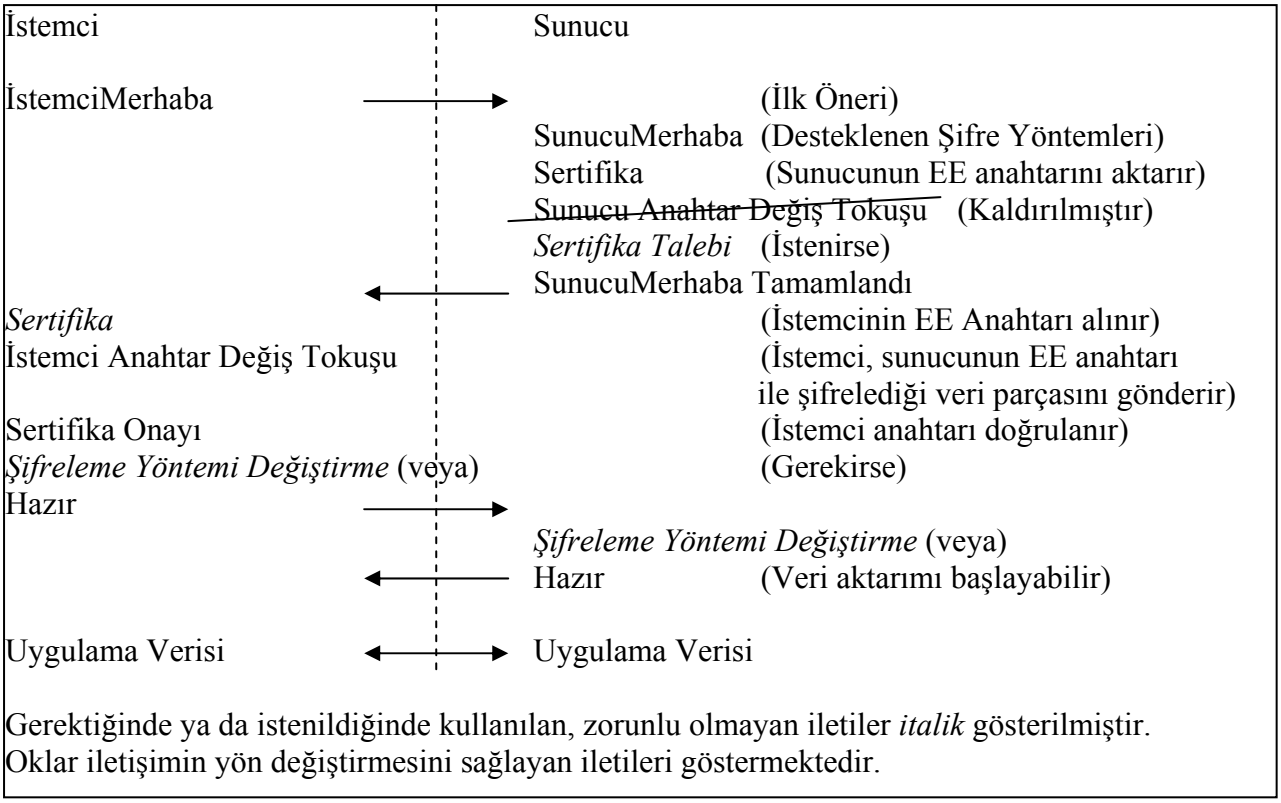
Şekil 6.1 RSA Tabanlı Sertifika Değişimi (RFC 2246'dan uyarlanmıştır)

6.2.2 EEŞ Tabanlı Tokalaşma

Error! Reference source not found.'de EEŞ tabanlı GSK tokalaşma işleyişi gösterilmiştir. Bu işleyiş TLS'de EEŞ kullanımını açıklayan IETF taslağına [9] dayandırılmıştır. RSA ile aynı şekilde işleyen ilk iki ileti ile istemci ve sunucu bir EEŞ tabanlı şifre takımı üzerinde anlaşma sağlar. Farklı olarak sunucu sertifikası, güvenilir bir SS tarafından EESİA kullanılarak imzalanmış, EEDH açık anahtarı içermektedir. Uygulamada kullanılan sertifika OpenSSL yazılımı içerisinde bulunan sertifika yazılımları vasıtasıyla yaratılmıştır. EESİA imzasının doğrulanmasından sonra istemci kendi EEDH açık anahtarını İstemci Anahtar Değişimi iletisi içerisinde sunucuya aktarmaktadır. Bu aşamada her iki tarafta da kendi EEDH gizli anahtarını ve diğerinin EEDH açık anahtarını kullanarak paylaşılan öncül sır elde edilir. Asıl sır ve simetrik anahtarların elde edilmesi RSA tokalaşma işleminin aynısı olarak yapılır.

İstemcinin aslının aynılığıyla kanıtlanması işlemi halen isteğe bağlı olmakla beraber, sunucu tarafından talep edilen aslıyla aynılığını kanıtlama türüne ve istemcinin sahip olduğu sertifikaya göre değişmektedir. Eğer istemcinin sertifikasında uzun vadeli bir EESİA anahtarı bulunuyorsa, geçici kullanımlık bir EEDH anahtarı oluşturulur ve EESİA anahtarı ile

imzalanarak Sertifika Onay iletisi içerisinde gönderilir. İstemci sertifikası uzun vadeli EEDH anahtarı içeriyorsa, açık EEDH anahtarı İstemci Sertifika iletisi içerisinde gönderilir. EEDH anahtarının doğrulanması, istemcinin sunucu açısından geçerli olan bir Hazır iletisi yaratabilmesi ile sağlanmaktadır.



Şekil 6.2 Eliptik Eğri Tabanlı Sertifika Değişimi [9]

Açıklanan ikinci istemcinin aslıyla aynılığının kanıtlama işlemi diğerine göre işlemsel olarak daha ucuz olmasına rağmen, iletişim de bulunduğu sunucunun sertifikasında bulunan ile tamamen aynı eliptik eğri parametrelerini kullanıyor olmasını gerektirir. Bu durumda istemcinin iletişim kuracağı her bir sunucu için uygun sertifikaya sahip olması gerekmektedir. Gerçekleştirim EESİA anahtar içeren sertifika desteğine sahip olmamasına rağmen gerçekleştirilen işlemlerde bir farklılık bulunmadığından aşağıda verilen kıyaslamalarda dahil edilmiştir.

6.2.3 GSK Açık Anahtar Şifreleme İşlemleri

GSK işleyişinde performans değerlendirmesi yaparken, arka arkaya GSK bağlantısı kuran istemci açısından şifreleme işlemleri için geçen toplam süre, eş zamanlı olarak GSK bağlantıları kurmak durumunda olan sunucu tarafından birim zamanda gerçekleştirilebilen şifreleme işlemi sayısını incelemeyi gerektirmektedir. Bu değerlere ulaşabilmek için yukarıda

açıklanan her bir sertifika değişimi işleminde gerçekleştirilen RSA tabanlı ya da EE tabanlı bir takım şifreleme, imza, şifre açma ya da imza onayı işleminin maliyetine bakmak gereklidir. Aşağıda istemci ve sunucu tarafından gerçekleştirilen açık anahtar şifreleme işlemleri açıklanmıştır

6.2.3.1 İstemci Aslının Aynılığının Kanıtlanmadığı Durumda

RSA Tokalaşma

İstemci tarafından bir RSA açık anahtar işlemi ile sunucunun sertifikası onaylanır (RSA_onay) ve bir diğer RSA açık anahtar işlemi ile de sunucunun açık anahtarı kullanılarak öncül sır şifrelenir (RSA_şifrele). Sunucu tarafında ise öncül sır şifresini çözmek üzere bir RSA gizli anahtar işlemi (RSA_şifreçöz) gerçekleştirilir.

EEDH/EESİA Tokalaşma

İstemci, sunucunun sertifikasını onaylamak üzere bir EESİA onay işlemi (EESİA_onay) ve öncül sır'ı elde etmek için sunucunun açık anahtarını ve kendi gizli anahtarını kullanarak bir EEDH şifreleme işlemi gerçekleştirir (EEDH_şifrele). Sunucu tarafında ise öncül sır şifresini çözmek üzere bir EEDH gizli anahtar işlemi (EEDH_şifreçöz) gerçekleştirilir.

6.2.3.2 İstemci Aslının Aynılığının Kanıtlandığı Durumda

RSA Tokalaşma

Önceki durumda olduğu gibi bir RSA_onay ve bir RSA_şifrele işlemine ilave olarak SertifikaOnay iletisi üretmek için bir RSA gizli anahtar şifreleme (RSA_şifrele) işlemi gerçekleştirilir. Sunucu tarafında istemci sertifikasının onaylanması için bir RSA açık anahtar işlemi (RSA_onay), SertifikaOnay iletisini onaylamak için bir başka RSA açık anahtar işlemi (RSA_onay) ve bir de öncül sır'ı elde etmek için RSA gizli anahtar işlemi gerçekleştirilir (RSA_şifreçöz).

EEDH/EESİA Tokalaşma

İstemci EEDH sertifikası kullanıyorsa, her iki taraf da EESİA doğrulama işlemi (EESİA_onay) ve ardından öncül sır'ı elde etmek için EEDH işlemi (istemci EEDH_şifrele; sunucu EEDH_şifreçöz) gerçekleştirir.

İstemci EESİA sertifikası kullanıyorsa, istemci tarafında sunucunun sertifikasını onaylamak üzere bir EESİA doğrulama işlemi (EESİA_onay) öncül sır'ı elde etmek için EEDH şifreleme (EEDH_şifrele) ve Sertifika Onay iletisini yaratmak için bir EESİA işlemi (EESİA_imza) gerçekleştirilir. Sunucu tarafında ise öncül sır'ı elde etmek için bir EEDH şifre çözme işlemi

(EEDH_şifreçöz) ve biri istemcinin sertifikasını, diğeri Sertifika Onay iletisini onaylamak amacıyla iki EESİA doğrulama (EESİA_onay) işlemi gerçekleştirilir.

EEDH_şifrele ve EEDH_şifreçöz işlemleri bir birinin tamamen aynısı işlemler olduğundan her ikisini de ifade etmek üzere EEDH_işlem kullanılması uygundur. Yukarıda anlatılan işlemler aşağıdaki çizelgelerde özetlenmiştir (Çizelge 6.5 ve Çizelge 6.6)

Çizelge 6.5 Sadece Sunucunun Aslıyla Aynılığının Kanıtlanması

	RSA	EEDH-EESİA
İstemci	RSA_onay + RSA_şifrele	EESİA_onay + EEDH_işlem
Sunucu	RSA_şifreçöz	EEDH_işlem

Çizelge 6.6 Hem Sunucu hem de İstemcinin Aslıyla Aynılığının Kanıtlanması

	RSA	EEDH-EESİA
İstemci	RSA_onay + RSA_şifrele + RSA_imza	EESİA_onay + EEDH_işlem
		EESİA_onay + EESİA_imza + EEDH_işlem
Sunucu	RSA_şifreçöz	EESİA_onay + EEDH_işlem
		2 x EESİA_onay + EEDH_işlem

Yukarıda anlatılan işlemler için performans karşılaştırması yapmak amacıyla OpenSSL yazılımının bir bileşeni olan speed kullanılmıştır. Speed aracı, her bir şifreleme algoritması için 10sn süreyle yapılan işlemleri saymakta ve bu sürede gerçekleştirilen işlem sayısına göre ortalama bir değer üretmektedir. Speed ile Pentium III – 700Mhz işlemcili bilgisayarda Red Hat Linux 9.0 işletim sisteminde ölçülen şifreleme algoritmaları başarım değerleri Çizelge 6.7 ila Çizelge 6.10'da verilmiştir.

Çizelge 6.7 RSA Şifreleme Performans Ölçümleri

RSA (ikil)	İmza(ms)	Onay(ms)	İmza/sn	Onay/sn
512	2,0	0,2	499,6	4612,8
1024	9,7	0,6	102,8	1763,4
2048	57,8	1,8	17,3	562,7
4096	384,4	6,1	2,6	165,1

Çizelge 6.8 Sayısal İmza Algoritması Performans Ölçümleri

SİA (ikil)	İmza(ms)	Onay(ms)	İmza/sn	Onay/sn
512	1,7	2,0	589,1	493,8
1024	4,9	5,9	204,8	168,5
2048	16,2	20,1	61,8	49,7

Çizelge 6.9 Eliptik Eğri Sayısal İmza Performans Ölçümleri

EESİA	İmza(ms)	Onay(ms)	İmza/sn	Onay/sn
160 ikil SEC asal	1,3	5,9	768,8	170,8
192 ikil NIST asal	1,4	6,2	732,0	160,5
224 ikil NIST asal	1,7	8,3	575,0	120,1
256 ikil NIST asal	2,1	10,6	466,3	94,8
384 ikil NIST asal	4,9	25,9	205,2	38,6
521 ikil NIST asal	9,7	50,6	103,6	19,7
163 ikil NIST Koblitz	4,9	12,1	203,1	82,6
233 ikil NIST Koblitz	9,7	23,9	102,8	41,8
283 ikil NIST Koblitz	15,1	44,3	66,4	22,6
409 ikil NIST Koblitz	34,7	107,2	28,8	9,3
571 ikil NIST Koblitz	78,1	251,1	12,8	4,0
163 ikil NIST 2^m	4,9	13,2	204,7	76,0
233 ikil NIST 2^m	9,7	26,5	102,9	37,8
283 ikil NIST 2^m	15,0	50,3	66,5	19,9
409 ikil NIST 2^m	34,8	123,6	28,7	8,1
571 ikil NIST 2^m	78,4	289,9	12,8	3,4

Çizelge 6.10 Eliptik Eğri Diffie Helman Performans Ölçümleri

EEDH	İşlem (ms)	İşlem/sn
160 ikil SEC asal	4,9	202,3
192 ikil NIST asal	5,3	189,0
224 ikil NIST asal	7,1	140,6
256 ikil NIST asal	8,8	113,1
384 ikil NIST asal	22,4	44,6
521 ikil NIST asal	42,3	23,7
163 ikil NIST Koblitz	5,9	170,8
233 ikil NIST Koblitz	11,7	85,5
283 ikil NIST Koblitz	22,0	45,5
409 ikil NIST Koblitz	53,1	18,8
571 ikil NIST Koblitz	125,1	8,0
163 ikil ecdh NIST 2m	6,4	156,4
233 ikil ecdh NIST 2m	12,8	78,3
283 ikil ecdh NIST 2m	24,9	40,2
409 ikil ecdh NIST 2m	61,4	16,3
571 ikil ecdh NIST 2m	144,2	6,9

Yukarıdaki çizelgelerde verilen değerlere göre Tokalaşma Protokolü performansı Çizelge 6.11 – Çizelge 6.16'da verilmiştir. RSA_onay ile RSA_şifrele ve RSA_imza ile RSA_şifreçöz aynı işlemlere karşılık gelmektedir.

Çizelge 6.11 Sadece Sunucunun Aslıyla Aynılığının Kanıtlanması RSA Performansı

	1024 ikil	2048 ikil	4096 ikil
İstemci	0,6 + 0,6 = 1,2ms	1,8 + 1,8 = 3,6 ms	6,1 + 6,1 = 12,2ms
Sunucu	9,7 ms	57,8 ms	384,4 ms

Çizelge 6.12 Sunucunun ve İstemcinin Aslıyla Aynılığının Kanıtlanması RSA Performansı

	1024 ikil	2048 ikil	4096 ikil
İstemci	0,6 + 0,6 + 9,7 = 10,9ms	1,8 + 1,8 + 57,8 = 61,4 ms	6,1 + 6,1 + 384,4 = 396,6ms
Sunucu	9,7 ms	57,8 ms	384,4 ms

Çizelge 6.13 Sadece Sunucunun Aslıyla Aynılığının Kanıtlanması Asal EE Performansı

	160 ikil	224 ikil	384 ikil
İstemci	5,9 + 4,9 = 10,8ms	8,3 + 7,1 = 15,4ms	25,9 + 22,4 = 48,3ms
Sunucu	4,9 ms	7,1 ms	22,4 ms

Çizelge 6.14 Sunucunun ve İstemcinin Aslıyla Aynılığının Kanıtlanması Asal EE Performansı

	160 ikil	224 ikil	384 ikil
İstemci	5,9 + 4,9 = 10,8ms	8,3 + 7,1 = 15,4ms	25,9 + 22,4 = 48,3ms
	5,9 + 1,3 + 4,9 = 12,1ms	8,3 + 1,7 + 7,1 = 17,1ms	25,9 + 4,9 + 22,4 = 53,2ms
Sunucu	5,9 + 4,9 = 10,8ms	8,3 + 7,1 = 15,4ms	25,9 + 22,4 = 48,3ms
	5,9 + 5,9 + 4,9 = 15,7ms	8,3 + 8,3 + 7,1 = 23,7ms	25,9 + 25,9 + 22,4 = 74,2ms

Çizelge 6.15 Sadece Sunucunun Aslıyla Aynılığının Kanıtlanması Tokalaşma Performans Karşılaştırması

	1024 ikil RSA(ms)	160 ikil EEŞ(ms)	2048 ikil RSA(ms)	224 ikil EEŞ(ms)	4096 ikil RSA(ms)	384 ikil EEŞ*(ms)
İstemci	1,2	10,8	3,6	15,4	12,2	48,3
Sunucu	9,7	4,9	57,8	7,1	384,4	22,4

* 384 ikil EEŞ, 7168 ikil RSA ile eşdeğer olmasına rağmen, fikir oluşturması için 4096 ikil RSA ile karşılaştırılmıştır.

Çizelge 6.16 Sunucunun ve İstemcinin Aslıyla Aynılığının Kanıtlanması Tokalaşma Performans Karşılaştırması

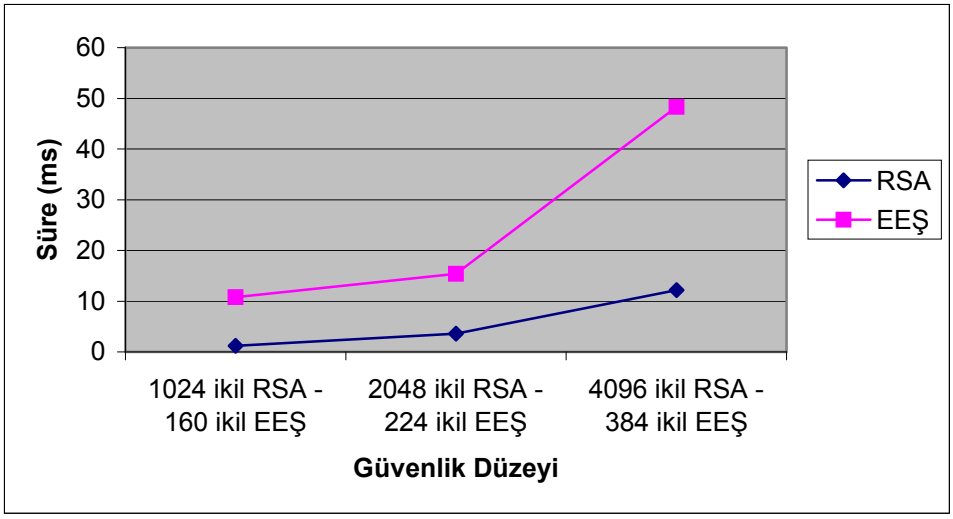
	1024 ikil RSA(ms)	160 ikil EEŞ(ms)	2048 ikil RSA(ms)	224 ikil EEŞ(ms)	4096 ikil RSA(ms)	384 ikil EEŞ*(ms)
İstemci	10,9	10,8	61,4	15,4	396,6	48,3
Sunucu	9,7	10,8	57,8	15,4	384,4	48,3

* 384 ikil EEŞ, 7168 ikil RSA ile eşdeğer olmasına rağmen, fikir oluşturması için 4096 ikil RSA ile karşılaştırılmıştır.

Yukarıdaki çizelgelerden de görülebileceği gibi, sadece sunucunun aslıyla aynılığının onaylandığı durumdaki istemci performansı haricindeki tüm durumlarda EEŞ performansı, RSA performansından daha yüksektir. Nadir güvenli bağlantı kuran istemci yanında sürekli güvenli bağlantı taleplerine yanıt veren sunucu performansının yüksek olması önemlidir. Özellikle, artan güvenlik düzeylerindeki performans farkı çok daha fazladır. Güvenlik düzeyine göre performans karşılaştırmaları aşağıdaki şekillerde (Şekil 6.3 – Şekil 6.6) verilmiştir.

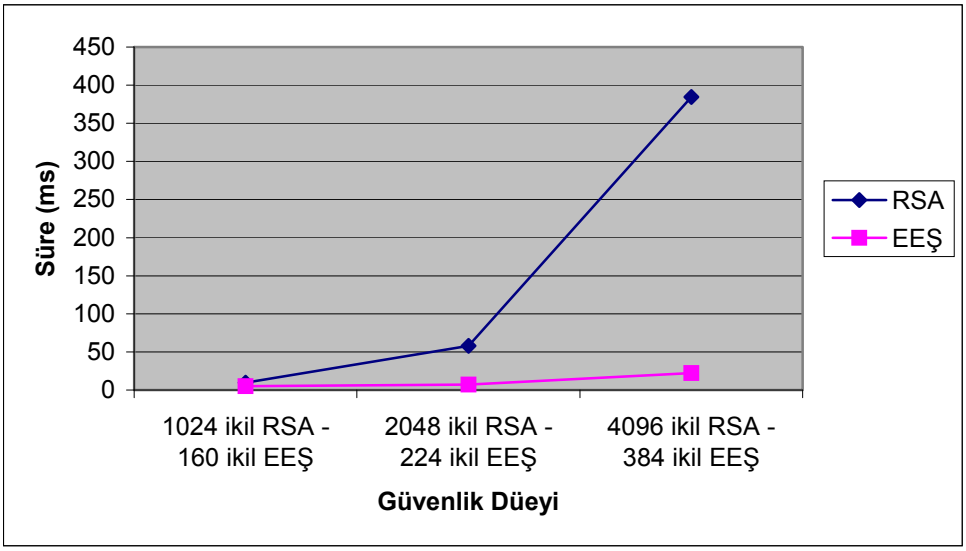
Şekil 6.3'de görüldüğü gibi sadece sunucunun onaylanmasında EESİA onay işleminin getirdiği yük nedeniyle, istemci performansında düşüş yaşanmakta, RSA kullanımı çok daha avantajlı gözükmektedir.

Şekil 6.3 Sadece Sunucunun Aslıyla Onaylanması İstemci Performansı



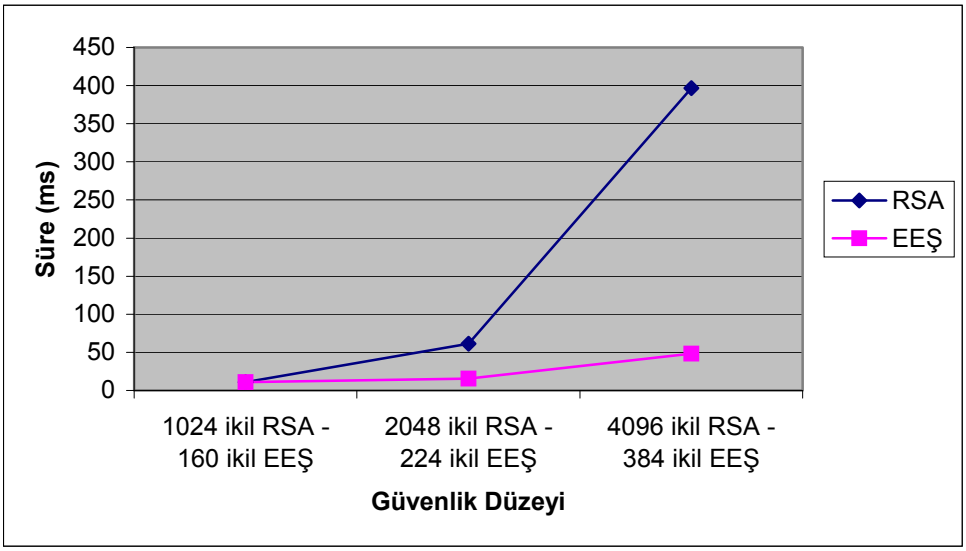
Şekil 6.4'de istemci performansından farklı olarak sunucu performansında belirgin bir artış olmaktadır. Özellikle 7168 ikil RSA düzeyinde güvenlik sağlayan 384-ikil EEŞ, sağladığı yaklaşık iki kat fazla güvenliğe rağmen 17 kat gibi yüksek bir performans göstermektedir.

Şekil 6.4 Sadece Sunucunun Aslıyla Onaylanması Sunucu Performansı

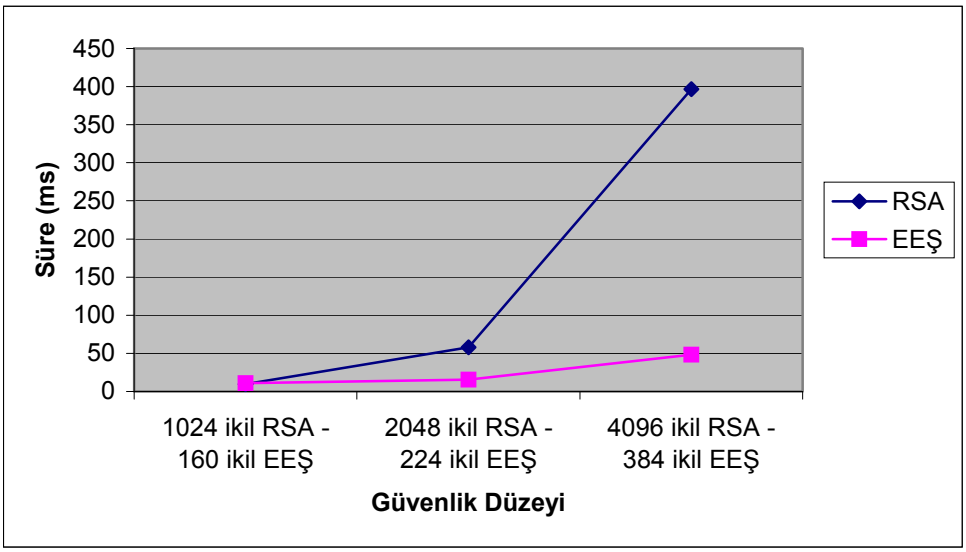


Sunucu ve istemcinin onaylanması durumunda hem istemci performansında (Şekil 6.5) hem de sunucu performansında (Şekil 6.6) EEŞ şifreleme, sadece sunucunun onaylanmasında ve hem istemci hem de sunucunun aslıyla aynılığının kanıtlanmasında, sunucu performansı olarak eşdeğer sonuçlar sağlamaktadır.

Şekil 6.5 Sunucun ve İstemcinin Aslıyla Onaylanması İstemci Performansı



Şekil 6.6 Sunucu ve İstemcinin Aslıyla Onaylanması Sunucu Performansı



Yukarıda verilen açıklama ve şekiller ışığında, EEŞ'nin, Güvenli Soket Katmanı Tokalaşma Protokolü'nde mevcut kullanılan RSA şifreleme yönteminden çok daha yüksek performanslarda çalışabileceği gösterilmiştir. Artan güvenlik gereksinimleri doğrultusunda, daha yüksek boyutlu anahtarlar kullanımı gereği duyulduğunda RSA performansının çok daha kabul edilemez düzeylerde kalacağı da aşikardır. Bugün olmasa bile, yakın gelecekte daha etkin bir şifreleme algoritması bulunamazsa, RSA yerine EEŞ kullanımı, bir gereksinim olarak karşımıza çıkacaktır.

6.3 Duyarga Ağlarında Eliptik Eğri Şifreleme Kullanarak Anahtar Dağıtımı

Duyarga ağ elemanları üzerindeki fiziksel kısıtlamalar yüksek işlem gücüne sahip cihazlar üzerinde işlemek üzere tasarlanmış olan mevcut güvenlik algoritmalarının kullanımını

olanaksız kılmaktadır. Örneğin, bir mote cihazının sahip olduğu bellek miktarı şifreleme algoritmalarının gerektirdiği parametreleri tutmak için bile yeterli değildir.

Duyarga ağ elemanlarının en önemli özelliklerinden birisi olan sınırlı enerji kaynağına sahip olmaları, her bir düğümün gerçekleştirebileceği işlem ve iletişimin düzeyini belirler. Enerji kullanımını asgari tutabilmek için güvenlik alt sisteminin işlemci üzerinde mümkün olduğunca az yük oluşturması ve aktarılan her bir mesaja eklediği bilginin asgari tutulması gerekmektedir. Öte yandan düğümlerin sınırlı ömürleri, kullanılabilir anahtarların ömrünü de kısıtlamaktadır.

Mote cihazlarının sınırlı işlem gücü ve enerjileri göz önüne alındığında, açık anahtar şifreleme yetkilendirme, bütünlük, gizlilik ve güvenlik sağlamak için kullanılamaz durumdadır. Ancak, hareketli cihazlara anahtar dağıtımı için uygun gözükmektedir. Mevcut sistemlerde, simetrik şifreleme için anahtarlar cihazlara başlangıçta yazılmakta ve cihazların ömrü boyunca bu anahtarlar kullanılmaktadır. Bir şekilde anahtarların değiştirilmesi gerektiğinde, dağıtılmış cihazların toplanarak anahtarların yazılması gerekmektedir (Malan vd. 2004).

Duyarga ağlarında, TinyOS, erişim kontrolü, yetkilendirme, bütünlük ve gizlilik sağlayabilmek için TinySEC kullanmaktadır. İleti yetkilendirme ve bütünlük kontrolü, ileti yetkilendirme kodlarıyla, gizlilik şifrelemeyle ve erişim kontrolü paylaşılan grup anahtarlarıyla sağlanmaktadır. TinySEC, bağlantı katmanında, 80 ikil simetrik şifreleme yapan SKIPJACK algoritması kullanmaktadır. Bu şekilde şifreli bir mesajın çözülmesi ortalama 2^{79} deneme gerektirmektedir. İlave olarak TinySEC tarafından kullanılan 4 sekizli ileti yetkilendirme kodu, mesajın 2^{-32} ihtimalle doğru kaynaktan geldiğini garanti etmektedir. TinyOS tarafından tanımlanmış olan CRC ve Grup kimliği alanları da TinySEC tarafından kullanılmaktadır. Bu şekilde 29 sekizli aktarılacak veri için; TinyOS 36 sekizli aktarırken, TinySEC 41 sekizli aktarım yapmakta, bu şekilde yaklaşık % 14 ek yük getirmektedir. Ölçümler göstermektedir ki, TinySEC paket aktarım süresine ortalama 2ms (%3), paketin komşu düğüme gidiş dönüş süresine ise, ortalama 5 ms (%3) ek yük getirmektedir. TinySEC paket geribildirim başarımını saniyede 0,28 paket azaltabilmektedir (Karlof vd. 2004)

TinySEC işletim sırasında yaklaşık 8 KB (yaklaşık 7 KB program ve 1 KB veri) bellek alanı gerektirmektedir [10] ki; bu bellek miktarı özellikle 4 KB RAM ve 128 KB ROM içeren MICA2 mote cihazları için sorun yaratan bir miktar değildir.

Duyarga ağlarda kullanılan cihazların ömrü genellikle enerji kaynaklarının ömürleriyle sınırlı olduğu için, cihazların fiziksel güvenliğini sağlamak anlamlı gözükmemektedir. Böyle bir

ortamda düğümleri oluşturan cihazların birisinin ele geçirilerek bir şekilde kullandığı simetrik şifre anahtarının ele geçirilmesi, tüm ağın kullandığı anahtarın ele geçirilmesi anlamına gelmektedir. Herhangi iki düğüm arasında farklı simetrik anahtarlar kullanımı, tüm ağın güvenliğinde oldukça yükselen bir güvenlik sağlayabilmektedir, ancak bu durumda, ağı oluşturan n cihaz varsa, her bir cihaza n^2 adet 80 ikil anahtarın yazılması gerekir ki, duyarga ağ cihazlarında bu düzeyde bir verinin tutulması olası bile değildir. Bu durumda çözüm TinySEC tarafından desteklenmekte olan anahtar değiştirebilme seçeneğinin değerlendirilmesinde yatmaktadır. Gizli simetrik anahtarların dağıtılması ise, tüm cihazları toplayıp işleme imkanının olmadığı düşünülecek olursa ancak açık anahtar şifreleme yöntemleri ile olabilecektir.

Bu yöntemler arasında en uygulanabilir olanı Diffie-Hellman anahtar değişimi algoritması olarak ortaya çıkmaktadır. Diffie-Hellman anahtar değişimi protokolünün duyarga ağlara uyarlanmış bir varyasyonu, istasyondan istasyona (STS – station to station) protokolü (Diffie vd. 1992) sorunsuz bir şekilde açık anahtar şifreleme kullanımına olanak sağlamaktadır. Buradaki sıkıntı, 80 ikil simetrik şifreleme tarafından sağlanan güvenlik düzeyinin altına düşmeden anahtarların dağıtılmasıdır. Bahsedilen düzeyde güvenlik ise ancak 1024 ikil açık anahtar şifreleme kullanarak sağlanabilmektedir. 8 ikil işlemci kullanan cihazlarda 1024 ikil değerler üzerinde asgari 160 ikil üssel işlemler yapmak kabul edilebilir sınırların çok üzerindedir. Bu noktada 1024 ikil güvenlik düzeyini 163 ikil kullanarak sağlayabilen EEŞ çok daha mantıklı bir çözüm olarak karşımıza çıkmaktadır.

6.3.1 Gerçekleştirim

Platformdan bağımsız olarak geliştirilmiş olan eliptik eğri kütüphanesini duyarga ağlarına uyarlamak üzere TOSSIM benzetim ortamı hedeflenmiş, C dilinde hazırlanmış olan kütüphane TinyOS için C uyarlaması olan NesC diline uyarlanmıştır. Bu amaçla, 80 ikil simetrik şifreleme ile eşdeğer güvenliği sağlayan, eliptik eğri nokta işlemlerinin daha yüksek performans ile gerçekleştirilebildiği 163 ikil Koblitz eğrisi kullanılmıştır. Eğri parametreleri Ek-1'de verilmiştir.

İkili alan üzerinde yapılan tamsayı işlemleri 8 ikil kelime genişliğine uyarlanmıştır. Bellek kısıtlı olduğu halde çarpma işlemlerinde taşmaları önlemek amacıyla, sayılar ihtiyaç duyulan 21 tane 8-ikil yerine, 42 tane 8-ikil olarak ifade edilmiştir. Farklı boyutta tamsayı tanımlanması bellek açısından belirgin bir iyileştirme getirmezken, gerçekleştirimin karmaşıklaşması ve artacak işlem sayısı nedeniyle performans üzerinde getirdiğinden fazlasına mal olacağından tercih edilmemiştir.

Nokta işlemleri Koblitz Eğrilerine özgü işlemler ile gerçekleştirilmiş, bu şekilde bölüm 6.1'de açıklanan performans değerlerinin üzerine çıkılması hedeflenmiştir. EEŞ kütüphanesinin nokta işlemleri modülü bu uyarlamalarla yeniden hazırlanmıştır. Örneğin, polinomların katsayıları daima mod 2 olduğundan, aritmetik işlemler eldesiz yapılabilmekte, nokta sıkıştırma, ve sıkıştırılmış biçimden açık biçime dönüştürme işlemleri ikili alanlarda çok daha süratli yapılabilmektedir.

Kod, olabildiğince elden geçirilmiş, bellek ve hız açısından optimize edilmiştir. Başlangıçta 4 dakikadan fazla süren açık anahtar üretimi işlemini kabul edilebilir düzeylere indirmek için sık kullanılan işlevler "inline" olarak tanımlanmış, bu şekilde işlev çağırma ve geri dönüş için zaman kaybedilmemiştir. İşlev ve yordamlarda fazladan yapılmakta olan kontrol kaldırılmak ya da kısaltılmak suretiyle hem hız hem de bellekten kazanılmıştır. Belirli döngülerin küçük indisten büyük indise doğru yapılması yerine, büyük indisten küçük indise doğru yapılmasının da performans üzerinde katkısı olmuştur.

Anahtar yaratma sırasında ihtiyaç duyulan rasgele sayı üretimi için TinyOS RandomLFSR modülünün kullanılması değerlendirilmiş, ancak çekirdek fonksiyonu olarak mote cihazının ID'sini kullanan bu modülün ürettiği rasgele sayıların bir güvenlik zaafı oluşturması nedeniyle, duyarga cihazının topladığı verileri tuttuğu bellek alanında belirli bir adresin rasgele sayı çekirdeği olarak kullanılması uygun görülmüştür. Mevcut gerçekleştirim, verilerin tutulduğu alan olmasına bakmaksızın belirli bir adresdeki değerleri okuyarak rasgele sayı üretiminde kullanmaktadır. Rasgele sayı çekirdeği üretiminin işlev olmaktan çıkarak tek bir okumaya dönüştürülmesi de performans açısından avantaj sağlamaktadır.

6.3.1.1 Duyarga Ağlarında EEDH İşleyişi

A düğümü rasgele seçtiği bir sayı ile gizli anahtarı d_A'yı, ve eğri taban noktası G ile gizli anahtarının çarpımından elde ettiği açık anahtarı $P_A=(x_A, y_A)$'yı oluşturur.

Bu sırada B düğümü aynı şekilde ve eş zamanlı olarak d_B ve $P_B=(x_B, y_B)$'yi oluşturmaktadır

A iki ayrı paketle x_A ve y_A'yı, B de aynı şekilde x_B, y_B'yi yayınlar.

Karşı tarafın paketlerini alan düğüm kendi gizli anahtarı ile karşı tarafın açık anahtarını çarparak ortak anahtarı ($P_O = d_A . d_B . G$) elde eder.

6.3.1.2 Duyarga Ağlarında EEDH Performansı

Yukarıda açıklanan işlemler için zaman ölçümleri ve bellek değerlendirmeleri aşağıda verilmiştir. Bu değerler, işlemleri 100 kere tekrarladıktan sonra elde edilen değerlerin

ortalaması alınarak elde edilmiştir. TOSSIM harcanan enerji ölçümü için doğrudan destek sağlamadığı için, enerji açısından değerlendirme yapılmamıştır.

Çizelge 6.17 Duyarga Ağı EEDH Performansı

Gizli Anahtar Yaratma	402 ms
Açık Anahtar Yaratma	58 sn.
Kullanılan RAM	2,086 B
Kullanılan EPROM	56,546 B

Yukarıda verilen değerler, bellek gereksinimi açısından bakıldığında, 128 KB ROM ve 4 KB RAM içeren mica2 mote cihazları için kabul edilebilir sınırlar içerisinde kalmaktadır. TinySEC tarafından kullanılan 7KB ROM ve 1 KB RAM'da bu değerlere eklenildiğinde, cihazlar üzerinde diğer programlar için 1 KB RAM ve 64 KB'den fazla ROM kalmaktadır. Bu değerler mevcut duyarga uygulamaları için yeterli olmaktadır.

Harcanan süre açısından bakıldığında, açık anahtar yaratma eliptik eğri nokta çarpımı işlemi gerektirdiğinden kısıtlı kaynaklar üzerinde uzun bir süre almaktadır. Ancak, anahtar değişimi işleminin çok sık yapılmayacağı göz önüne alınacak olursa, sağlanacak güvenlik için bu süre, dolayısıyla harcanacak enerji göze alınabilir.

7. SONUÇ

Bu tez çalışmasının amacı, Internet iletişiminde güvenliği sağlamak üzere yoğun olarak kullanılan Güvenli Soket Katmanı performansını, eliptik eğri şifreleme kullanarak arttırılabileceğinin ve duyarga ağlarında şifreli iletişim için kullanılan anahtarların, ihtiyaç duyulduğunda değiştirilebilmesinin yine eliptik eğri şifreleme kullanarak mümkün olduğunun gösterilebilmesidir.

Güvenli Soket Katmanı Protokolü, veri aktarımı sırasında verileri korumak üzere simetrik şifreleme algoritmaları kullanmaktadır. Ancak, simetrik şifreleme algoritma anahtarlarının belirlenmesi için kullanılan Güvenli Soket Katmanı Tokalaşma Protokolü, açık anahtar şifreleme mekanizmalarından faydalanmakta ve genel olarak açık anahtar şifreleme protokolü olarak RSA kullanmaktadır. Günümüz koşullarında yeterli güvenlik ancak 1024 ikil RSA şifreleme ile sağlanmakta, bu değerin artan işlem gücü ile arttırılması gerekmektedir. Anahtar boyu arttıkça işlemler daha yavaş olmakta, bu durumda da performans doğal olarak düşmektedir.

Eliptik eğri şifreleme algoritmaları temelde, yaygın olarak kullanılan klasik şifreleme algoritmalarının, tamsayı alanlarından eliptik eğrilere taşınması temeline dayanmaktadır ve eliptik eğri şifreleme algoritmalarının dayandığı matematiksel problem olan eliptik eğri ayrık logaritma probleminin çözüm karmaşıklığı, klasik şifreleme algoritmalarının dayandığı matematiksel problemler olan ayrık logaritma problemi ve tamsayı çarpanlarına ayırma probleminin karmaşıklığından çok daha yüksektir. Bu durumda, eliptik eğri şifreleme, çok daha kısa anahtarlarla, klasik şifreleme algoritmalarının sağladığı güvenliği sağlayabilmektedir.

Duyarga ağlarında aktarılan verinin güvenliğini sağlama yöntemlerinden birisi, duyarga ağları için tasarlanmış TinyOS işletim sisteminin bir alt bileşeni olan TinySEC kullanımıdır. TinySEC, Güvenli Soket Katmanında olduğu gibi veri iletişimi sırasında güvenliği simetrik şifreleme kullanarak yapmaktadır. TinySEC, kullanılan simetrik şifre anahtarlarının değiştirilmesine olanak sağlayan bir altyapıya sahip olmasına rağmen, anahtar değişimini sağlayan bir mekanizma içermemektedir. Güvenli soket katmanından örnek alacak olursak, duyarga ağlarında simetrik şifreleme anahtarlarının değişimi için açık anahtar şifreleme algoritmaları kullanılabilir. Açık anahtar şifreleme algoritmaları içerisinde eliptik eğri tabanlı olanlar, kullanılacak anahtar boylarının kısalığı avantajı nedeniyle daha elverişli görünmektedir (Bozkurt 2005a, 2005c).

Yapılan tez çalışması ile Güvenli Soket Katmanı Tokalaşma Protokolü'ne eliptik eğri şifreleme yöntemleri eklenmiştir. Yapılan performans ölçümleri, eliptik eğri şifreleme algoritmaları kullanarak gerçekleştirilen anahtar değişimi işlemlerinin, sunucu açısından bakıldığında her durumda daha yüksek performans sağladığını; istemci açısından bakıldığında ise, istemcinin gerçekleştirdiği eliptik eğri sayısal imza işlemleri nedeniyle, sadece sunucunun aslıyla aynılığının kanıtlandığı durumda RSA tabanlı yönteme göre olumsuz performans sergilediğini göstermiştir.(Bozkurt 2005b, 2005c) Bunun yanında güvenlik düzeyinin yükseldiği durumlarda eliptik eğri şifrelemenin performans üstünlüğü çok daha belirginleşmektedir.

Duyarga ağlarında ise, TinySEC tarafından kullanılan 80 ikil simetrik şifreleme ile eşdeğer güvenlik düzeyini sağlayan 163 ikil eliptik eğri şifreleme kullanarak anahtar dağıtımının mümkün olduğu gösterilmiştir. RSA ile eşdeğer güvenlik düzeyi olan 1024 ikil şifreleme işlemlerinin, 4 KB belleğe sahip duyarga ağ cihazlarının kısıtlı olanaklarıyla gerçekleştirilmesi olası gözükmemektedir. EEŞ ile RSA karşılaştırmasını duyarga ağları üzerinde gerçekleştirme amaçlı bir çalışmada (Gura vd., 2004) daha yüksek kapasiteli duyarga cihazlarını (Atmel Atmega 128) kullanmış ve RSA gerçekleştiriminin EEŞ gerçekleştiriminden 13,5 kat daha düşük performans sergilediğini göstermişlerdir. Gerçekleştirim ortamı kısıtlarına uyum sağlayabilmesi açısından da eliptik eğri şifreleme algoritmalarının klasik şifreleme algoritmalarından avantajlı olduğu ortaya çıkmaktadır.

Eliptik eğri şifreleme işlemleri asal alanlar üzerinde daha hızlı gerçekleştirimler sağlayabilmektedir. Duyarga ağlarında ikili alan Koblitz eğrileri kullanılmasının nedeni, bu eğrilerin özellikleri itibarıyla, en iyileştirmeye açık olması ve bu eğriler üzerinde daha hızlı işlemler yapabilen algoritmaların bulunmasıdır. Asal alanlar için de benzer algoritmalar gerçekleştirilebilirse, duyarga ağları için anahtar dağıtımı daha yüksek performans ile gerçekleştirilebilir.

KAYNAKLAR

Akyıldız, I.F., Su, W., Sankarasubramaniam, Y. ve Çayırcı, E., (2002), "Wireless Sensor Networks: A Survey", Computer Networks, 38:393-422.

ANSI, (1998), X9.62 Public Key Cryptography For the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm, American National Standards Institute.

ANSI, (1999), X9.63 Public Key Cryptography For the Financial Services Industry: The Key Agreement and Key Transport using Elliptic Curve Cryptography, American National Standards Institute.

Apostolopoulos, G., Peris, V. ve Saha, D., (1999), "Transport Layer Security, How Much does it Really Cost?" Eighteenth Conference on Computer Communications, Mart 1999, NY.

Atkins, D., Graff M., Lenstra, A.K. ve Leyland, P.C., (1995), "The Magic Words are SQUEAMISH OSSIFRAGE", LNCS, 917: 263-277.

Badia, L., (2001), Real World SSL Benchmarking, Rainbow Technologies Whitepaper.

Bosselaers, A., Govaerts, R. ve Vandewalle, J., (1994), "Comparison of Three Modular Reduction Functions", LNCS, 773:175-186.

Bozkurt, Ö.Ö., (2005a), "Tasarsız Ağlarda Eliptik Eğri Şifreleme Kullanarak Anahtar Dağıtımı", Ağ ve Bilgi Güvenliği Sempozyumu (ABG2005), 9-11 Haziran 2005, İstanbul Teknik Üniversitesi, İstanbul.

Bozkurt, Ö.Ö., (2005b), "Internet Communication Security using Elliptic Curve Cryptography", Innovations in Intelligent Systems and Applications (INISTA2005), 15-18 Haziran 2005, Yıldız Teknik Üniversitesi, İstanbul.

Bozkurt, Ö.Ö., (2005c), "Performance Considerations for Elliptic Curve Cryptography in Communications", Advances in Computer Science and Engineering : New Trends in Computer Networks, 1: 207-217

Brown, M., Hankerson, D., Lopez, J. ve Menezes A., (2001), "Software Implementation of the NIST Elliptic Curves over Prime Fields", LNCS, 2020:250-265.

Buhler, J.P., Lenstra H.W. ve Pomerance C., (1993), "Factoring Integers with the Number Field Sieve", The Development of the Number Field Sieve, LNCS, 1554:43-100.

Callaway, E.H., (2004), Wireless Sensor Networks: Architectures and Protocols, CRC Press, Boca Raton, FL, ABD.

Chong, C. ve Kumar, S.P., (2003), "Sensor Networks: Evolution, Opportunities and Challenges", Proceedings of the IEEE, 41(8):1247-1256.

Chudnovsky, D.V. ve Chudnovsky, G.V., (1986), "Sequences of Numbers Generated by Addition in Formal Groups and New Primality and Factorization Tests", Advances in Applied Mathematics, 7(4):385-434.

Clark, D., (2000), "Encryption Advances to Meet Internet Challenges", IEEE Computer, 33(8):20-24.

Coarfa, C., Druschel, P. ve Wallach, D., (2002), "Performance Analysis of TLS Web Servers", Network and Distributed Systems Security Symposium '02, Şubat 2002, San Diego, California.

Cohen, H., Miyaji, A. ve One, T., (1998), "Efficient Elliptic Curve Exponentiation Using Mixed Coordinates", LNCS, 1514:51-65.

Dierks, T. ve Allen, C., (1999), "The TLS Protocol – Version 1.0", IETF RFC 2246.

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

Diffie, W., VanOorschot, P.C. ve Wiener, M.J., (1992), "Authentication and Authenticated Key Exchanges", Designs, Codes and Cryptography, 2:107-125.

ElGamal, T., (1985), "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory, 31:469-472.

Freier, A.O., Karlton, P. ve Kocher, P.C., (1996), "The SSL Protocol v.3.0", Internet Draft.

Gallant, R., Lambert, R. ve Vanstone, S., (2000), "Improving the Parallelized Pollard Lambda Search on Binary Anomalous Curves", Mathematics of Computation, 69:1699-1705.

Goldberg, A., Buff, R. ve Schmitt, A., (1998), "Secure Web Server Performance Dramatically Improved by Caching SSL Session Keys", Workshop on Internet Server Performance, June 1998, Madison, Wisconsin.

Gordon, D., (1993), "Discrete Logarithms in GF(p) using the Number Field Sieve", SIAM Journal on Discrete Mathematics, 6:124-138.

Gujardo J. ve Paar, C., (1997), "Efficient Algorithms for Elliptic Curve Cryptosystems", LNCS, 1294:342-356.

Gura, N., Patel, A., Wander, A., Eberle, H. ve Shantz, S.C., (2004), "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs", LNCS, 3158:119-132.

Hankerson, D., Hernandez, J. ve Menezes, A., (2000), "Software Implementation of Elliptic Curve Cryptography over Binary Fields", LNCS, 1965:1-24.

Hill J., Szewczyk, R., Woo, A., Hollar, S., Culler, D. ve Pister, K., (2000), "System Architecture Directions for Networked Sensors", 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000), 12-15 Kasım 2000, Cambridge, MA, ABD.

Hodjat, A. ve Verbauwhede, I., (2002), "The Energy Cost of Secrets in Ad-hoc Networks", IEEE CAS Workshop on Wireless Communications and Networking, 5-6 Eylül 2002, Pasadena, CA, ABD.

Husemoller, D., (2004), Elliptic Curves, Second Ed., Springer-Verlag, Berlin.

IEEE, (2000), "P1363 - Standard Specifications for Public Key Cryptography", Institute of Electrical and Electronics Engineers.

Ilyas, M. ve Mahgoub, I., (2005), Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems, CRC Press, Boca Raton, FL, ABD.

ISO/IEC, (1999), Programming Languages, International Standard.

Joye, M., Quisquater, J.J., ve Tagaki, T., (2001), "How to Choose Secret Parameters for RSA-Type Cryptosystems over Elliptic Curves", Designs, Codes and Cryptography, 23:297-316.

Karlof, C. ve Wagner, D., (2003), "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures", IEEE International Workshop on Sensor Network Protocols and Applications 2003, 11 Mayıs 2003, Berkeley, CA, ABD.

Karlof, C., Sastry, N. ve Wagner, D., (2004), "TinySec: A Link Layer Security Architecture for Wireless Sensor Networks", 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys 2004), 3-5 Kasım 2004, Baltimore, MA, ABD.

Koblitz, N., (1987a), "Elliptic Curve Cryptosystems", Mathematics of Computation, 48:203-209.

Koblitz, N., (1987b), A Course in Number Theory and Cryptography, Springer-Verlag, NY.

Koblitz, N. ve Menezes, A., (2000), "The State of Elliptic Curve Cryptography", Designs, Codes and Cryptography, 19:173-193.

Koç, Ç.K. ve Acar, T., (1998), "Montgomery Multiplication in GF(2^k)", Designs, Codes and Crptography, 14:57-69.

Konstantinou, E., Stamatiou Y.C. ve Zaroliagis, C., (2003), "On the Construction of Prime Order Elliptic Curves", LNCS, 2904:309-22.

LaMacchia, B.A. ve Odlyzko, A.M. (1991), "Computation of Discrete Logarithms in Prime Fields", Designs, Codes and Cryptography, 1:47-62.

Lauter, K., (2004), "The Advantages of Elliptic Curve Cryptography for Wireless Security", IEEE Wireless Communications Magazine, 11(1):62-67.

Law, Y.W., Dulman, S., Etalle, S. ve Havinga, P., (2003), "Assessing Security Critical Energy Efficient Sensor Networks", 18th IFIP International Information Security Conference, 26-28 Mayıs 2003, Atina, Yunanistan.

Lenstra A.K., Lenstra, H.W., Manasse M.S. ve Pollard J.M., (1993), "The Number Field Sieve", The Development of the Number Field Sieve, Lecture Notes in Mathematics, 1554:11-42.

Lenstra, H.W., (1987), "Factoring Integers with Elliptic Curves", Annals of Mathematics, 126:649-673.

Levis, P., Lee, N., Welsh, M. ve Culler, D., (2003), "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications", First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003), 5-7 Kasım 2003, Los Angeles, CA, ABD.

Levis, P., Madden S., Gay, D., Polastre, J., Szewczyk, R., Woo, A., Brewer, E. ve Culler, D., (2004), "The Emergence of Networking Abstractions and Techniques in TinyOS", USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI 2004), Mart 2004, San Francisco, CA, ABD.

Lopez, J. ve Dahab, R., (1999), "Improved Algorithms for Elliptic Curve Arithmetic in GF(2^n)", LNCS, 1556:201-212.

Lopez, J. ve Dahab, R., (2000), "High Speed Software Multiplication in F_{2m}", LNCS, 1977:203-212.

Lorincz, K., Malan, D., Fulford-Jones, T.R.F., Najov, A. ve Welsh, M., (2004), "Sensor Networks for Emergency Response: Challenges and Opportunities" IEEE Pervasive Networks, 3(4):16-23.

Malan, D. J., Welsh, M. ve Smith M.D., (2004), "A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography", First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, (IEEE SECON 2004), 4-7 Ekim 2004, Santa Clara, CA, ABD.

Menezes, A., (1993), Elliptic Curve Public Key Cryptosystems, Kluwer Academic Publishers, Boston.

Menezes, A., Okamoto, T. ve Vanstone, S.A., (1993), "Reducing Elliptic Curve Logarithms to Logarithms in Finite Fields", IEEE Transactions on Information Theory, 39:1639-1646.

Menezes, A., VanOorschot, P., ve Vanstone, S.A., (1996), Handbook of Applied Cryptography, CRC Press, Boca Raton, FL, ABD.

Miller, V., (1986), "Uses of Elliptic Curves in Cryptography", LNCS, 218: 417-426.

Miyaji, A., Ono, T. ve Cohen, H., (1997), "Efficient Elliptic Curve Exponentiation", LNCS, 1334:282-290.

Morain, F. ve Olivos, J., (1990), "Speeding up the Computations on an Elliptic Curve Using Addition-Subtraction Chains", Informatique Théorique et Applications, 24:531-544.

Morrison, M.A. ve Brillhart, J., (1975), "A Method of Factoring and the Factorization of F7", Mathematics of Computation, 29:183-205.

NIST, (1994), FIPS 186, Digital Signature Standard, National Institute for Standards and Technology.

NIST, (1999), Recommended Elliptic Curves for Federal Government Use, National Institute of Standards and Technology.

NIST, (2000), FIPS 186-2, Digital Signature Standard (+change notice), National Institute for Standards and Technology.

Nyberg, K. ve Rueppel, R.A., (1996), "Message Recovery for Signature Schemes Based on the Discrete Logarithm Problem", Designs, Codes and Cryptography, 7:61-81.

Odlyzko, A., (1995), "The Future of Integer Factorization", CryptoBytes - The Technical Newsletter of RSA Laboratories, 1(2):5-12.

Odlyzko, A., (2000), "Discrete Logarithms: the Past and the Future", Designs, Codes and Cryptography, 19(2):129-145.

Perrig, A., Szewczyk, R., Wen, V., Culler, D., ve Tygar J.D., (2001), "SPINS: Security Protocols for Sensor Networks", Mobile Computing and Networking, 189-199.

Pohlig, S. ve Hellman M., (1978), "An Improved Algorithm for Computing Logarithms over GF(p) and Its Cryptographic Significance", IEEE Transaction on Information Theory, 24:106-110.

Pointcheval, D., ve Stern, J., (1996), "Security Proofs for Signature Schemes", LNCS, 1070:387-398.

Pollard, J., (1978), "Monte Carlo Methods for Index Computation mod p", Mathematics of Computation, 32:918-924.

Pomerance, C., (1985), "The Quadratic Sieve Factoring Algorithm", LNCS, 209:169-182.

Rabin, M.O., (1979), "Digitalized Signatures and Public-Key Functions as Intractable as Factorization", Teknik Rapor, MIT Laboratory for Computer Science.

Rivest, R., Shamir, A. ve Adleman, L., (1978), "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM, 21(2):120-126.

Rohatgi, R, (1999), "A Compact and Fast Hybrid Signature Scheme for Multicast Packet Authentication", 6th ACM Conference on Computer and Communications Security, 1-4 Kasım 1999, Singapur.

Rosing, M., (1999), Implementing Elliptic Curve Cryptography, Maning Publications.

Schirokauer, O., Weber, D. ve Denny, T.F., (1996), "Discrete Logarithms: the Effectiveness of the Index Calculus Method", LNCS, 1122:337-361.

Schneier, B., (1996), Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code in C, John Wiley and Sons, Inc. New York.
Schnorr, C.P., (1991), "Efficient Signature Generation by Smart Cards", Journal of Cryptology, 4:161-174.

Schroeppel, R., Orman, H., O'Malley, S. ve Spatscheck, O., (1995), "Fast Key Exchange with Elliptic Curve Systems", LNCS, 963:43-56.

SECG, (2000a), "SEC1: Elliptic Curve Cryptography", Standards for Efficient Cryptography Group.

SECG, (2000b), "SEC2: Recommended Elliptic Curve Domain Parameters", Standards for Efficient Cryptography Group.

Semaev, I., (1998), "Evaluation of Discrete Logarithms in a Group of P-torsion Points of an Elliptic Curve in Characteristic p". Mathematics of Computation, 67:353-356.

Shanks, D., (1971), "Class Number, a Theory of Factorization and Genera", Proceedings of Symposium on Pure Mathematics, American Mathematical Society, Providence, 20:415-440.

Solinas, J.A., (2000), "Efficient Arithmetic on Koblitz Curves", Designs, Codes and Cryptography, 19:195-249.

Stinson, D., (1995), Cryptography: Theory and Practice, CRC Press, Boca Raton, FL, ABD.

Wahl, M., Kille, S. ve Howes, T., (1997), "Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names", IETF RFC 2253.

Weber, D. ve Deny, T.F., (1988), "The Solution of McCurley's Discrete Log Challenge", LNCS, 1962:458-471.

Internet Kaynakları

[1] http://www.certicom.com

[2] http:www.loria.fr/~zimmerma/records/**ecmnet**.html

[3] http://www.**rsa**security.com

[4] http://csrc.ncsl.nist.gov/

[5] http://docs.sun.com/source/816-6154-10/contents.htm

[6] http://www.tinyos.net/tinyos-1.x/doc/

[7] http://www.cs.berkeley.edu/~pal/research/**tossim**.html

[8] http://www.openSSL.org

[9] hhttp://www.ietf.org/proceedings/ 04mar/I-D/draft-ietf-tls-ecc-05.txt

[10] TinySEC Kullanıcı el Kitabı. http://www.tinyos.net/tinyos-1.x/doc/tinysec.pdf

Ek 1. Eliptik Eğri Parametreleri

Asal Alan Eğrileri

NIST tarafından önerilen eğriler, $y^2 = x^3 - 3x + b$ ile belirtilmekte, b değerinin değişimi ile farklı eğriler oluşmaktadır. Eliptik eğri parametreleri belirlenirken F_p üzerinde eliptik eğri alan parametrelerini oluşturan altılının yanı sıra, hash algoritmasına verilecek çekirdek değeri s, ve hash algoritmasından dönecek değer c de belirlenmiştir. Alan parametreleri:

T (p, a, b, G, r, h)

p : Sonlu alan F_p'yi belirleyen tamsayı,

a,b : $y^2 \equiv x^3 + ax + b \pmod{p}$ denklemi ile tanımlanan eğriyi belirleyen elemanlar,

G : $G = (G_x, G_y)$ taban noktası,

r : G noktasının derecesini veren asal sayı

h : $\#E(F_p) / n$

P-192 Eliptik Eğrisi

p = 6277101735386680763835789423207666416083908700390324961279

r = 6277101735386680763835789423176059013767194773182842284081

s	= 3045ae6f	c8422f64	ed579528	d38120ea	e12196d5	
c	= 3099d2bb	bfcb2538	542dcd5f	b078b6ef	5f3d6fe2	c745de65
b	= 64210519	e59c80e7	0fa7e9ab	72243049	feb8deec	c146b9b1
G_x	= 188da80e	b03090f6	7cbf20eb	43a18800	f4ff0afd	82ff1012
G_y	= 07192b95	ffc8da78	631011ed	6b24cdd5	73f977a1	1e794811

P-224 Eliptik Eğrisi

p = 26959946667150639794667015087019630673557916260026308143510066 298881

r = 2695994 666715063979466701508701962594045780771442439172168272236806 1

s	= bd713447	99d5c7fc	dc45b59f	a3b9ab8f	6a948bc5	
c	= 5b056c7e 218291fb	11dd68f4	0469ee7f	3c7a7d74	f7d12111	6506d031
b	= b4050a85 2355ffb4	0c04b3ab	f5413256	5044b0b7	d7bfd8ba	270b3943
G_x	= b70e0cbd 115c1d21	6bb4bf7f	321390b9	4a03c1d3	56c21122	343280d6
G_y	= bd376388 85007e34	b5f723fb	4c22dfe6	cd4375a0	5a074764	44d58199

P-256 Eliptik Eğrisi

p = 1157920892103562487626974469494075735300861434152903141955336313088670 97853951

r = 1157920892103562487626974469494075735299969552241357603424222590610 685 12044369

s	= c49d3608	86e70493	6a6678e1	139d26b7	819f7e90	
c	= 7efba166 af317768	2985be94	03cb055c	75d4f7e0	ce8d84a9	c5114abc
b	= 5ac635d8 3bce3c3e	aa3a93e7	b3ebbd55	769886bc	651d06b0	cc53b0f6
G_x	= 6b17d1f2 f4a13945	e12c4247	f8bce6e5	63a440f2	77037d81	2deb33a0

G_y = 4fe342e2 fe1a7f9b 8ee7eb4a 7c0f9e16 2bce3357 6b315ece cbb64068 37bf51f5

P-384 Eliptik Eğrisi

p = 39402006196394479212279040100143613805079739270465446667948293404245721771496870329047266088258938001861606973112319

r = 39402006196394479212279040100143613805079739270465446667946905279627659399113263569398956308152294913554433653942643

s = a335926a a319a27a 1d00896a 6773a482 7acdac73

c = 79d1e655 f868f02f ff48dcde e14151dd b80643c1 406d0ca1 0dfe6fc5 2009540a 495e8042 ea5f744f 6e184667 cc722483

b = b3312fa7 e23ee7e4 988e056b e3f82d19 181d9c6e fe814112 0314088f 5013875a c656398d 8a2ed19d 2a85c8ed d3ec2aef

G_x = aa87ca22 be8b0537 8eb1c71e f320ad74 6e1d3b62 8ba79b98 59f741e0 82542a38 5502f25d bf55296c 3a545e38 72760ab7

G_y = 3617de4a 96262c6f 5d9e98bf 9292dc29 f8f41dbd 289a147c e9da3113 b5f0b8c0 0a60b1ce 1d7e819d 7a431d7c 90ea0e5f

P-521 Eliptik Eğrisi

p = 6864797660130609714981900799081393217269435300143305409394463459185543183397656052122559640661454554977296311391480858037121987999716643812574028291115057151

r = 6864797660130609714981900799081393217269435300143305409394463459185543183397655394245057746333217197532963996371363321113864768612440380340372808892707005449

s = d09e8800 291cb853 96cc6717 393284aa a0da64ba

c = 0b4 8bfa5f42 0a349495 39d2bdfc 264eeeeb 077688e4 4fbf0ad8 f6d0edb3 7bd6b533 28100051 8e19f1b9 ffbe0fe9 ed8a3c22 00b8f875 e523868c 70c1e5bf 55bad637

b = 051 953eb961 8e1c9a1f 929a21a0 b68540ee a2da725b 99b315f3 b8b48991 8ef109e1 56193951 ec7e937b 1652c0bd 3bb1bf07 3573df88 3d2c34f1 ef451fd4 6b503f00

G_x = c6 858e06b7 0404e9cd 9e3ecb66 2395b442 9c648139 053fb521 f828af60 6b4d3dba a14b5e77 efe75928 fe1dc127 a2ffa8de 3348b3c1 856a429b f97e7e31 c2e5bd66

G_y = 118 39296a78 9a3bc004 5c8a5fb4 2c7d1bd9 98f54449 579b4468 17afbd17 273e662c 97ee7299 5ef42640 c550b901 3fad0761 353c7086 a272c240 88be9476 9fd16650

İkili Alan Eğrileri

İkili alan eğrileri belirlenirken, bir $y^2 + xy = x^3 + x^2 + b$ biçiminde yarı rasgele eğri (B-xxx) bir de $y^2 + xy = x^3 + ax^2 + 1$ biçiminde Koblitz eğrisi (K-xxx) belirlenmiştir. Eğri parametreleri belirlenirken, ikili alan eliptik eğri alan parametrelerini oluşturan T (m, f(x), a, b, G, r, h) yedilisine ilave olarak, eğrinin Gösterim biçimini belirleyen T ve normal gösterimlerde hash fonksiyonunun çekirdeği de verilmektedir.

Burada;

m : Sonlu alan F_{2^m}'yi belirleyen tamsayı,

f(x) : m dereceli F_{2^m}'yi simgeleyen indirgenemez polinom

a,b : $y^2 + xy \equiv x^3 + ax^2 + b$ denklemi ile tanımlanan eğriyi belirleyen elemanlar,

G : $G = (x_G, y_G)$ taban noktası,

r : G noktasının derecesini veren asal sayı

h : $\#E(F_{2^m}) / n$

163. derece İkili Alan

T = 4

$p(t) = t^{163} + t^7 + t^6 + t^3 + 1$

K-163 Eliptik Eğrisi

a = 1

r = 5846006549323611672814741753598448348329118574063

Polinom Tabanlı Gösterim :

G_x =	2	fe13c053	7bbc11ac	aa07d793	de4e6d5e	5c94eee8
G_y =	2	89070fb0	5d38ff58	321f2e80	0536d538	ccdaa3d9

Normal Gösterim :

G_x =	0	5679b353	caa46825	fea2d371	3ba450da	0c2a4541
G_y =	2	35b7c671	00506899	06bac3d9	dec76a83	5591edb2

B-163 Eğrisi

r = 5846006549323611672814742442876390689256843201587

Polinom Tabanlı Gösterim :

b =	2	0a601907	b8c953ca	1481eb10	512f7874	4a3205fd
G_x =	3	f0eba162	86a2d57e	a0991168	d4994637	e8343e36
G_y =	0	d51fbc6c	71a0094f	a2cdd545	b11c5c0c	797324f1

Normal Gösterim :

s =	85e25bfe	5c86226c	db12016f	7553f9d0	e693a268	
b =	6	645f3cac	f1638e13	9c6cd13e	f61734fb	c9e3d9fb
G_x =	0	311103c1	7167564a	ce77ccb0	9c681f88	6ba54ee8
G_y =	3	33ac13c6	447f2e67	613bf700	9daf98c8	7bb50c7f

233. Derece İkili Alan

T = 2

$p(t) = t^{233} + t^{74} + 1$

K-233 Eliptik Eğrisi

a = 0

r = 3450873173395281893717377931138512760570940988622521263280870247413 43

Polinom Tabanlı Gösterim :

G_x =	172	32ba853a	7e731af1	29f22ff4	149563a4	19c26bf5
	0a4c9d6e	efad6126				
G_y =	1db	537dece8	19b7f70f	555a67c4	27a8cd9b	f18aeb9b
	56e0c110	56fae6a3				

Normal Gösterim :

G_x =	0fd	e76d9dcd	26e643ac	26f1aa90	1aa12978	4b71fc07
	22b2d056	14d650b3				
G_y =	064	3e317633	155c9e04	47ba8020	a3c43177	450ee036
	d6335014	34cac978				

B-233 Eğrisi

r = 6901746346790563787434755862277025555839812737345013555379383634485463

b =	066	647ede6c	332c7f8c	0923bb58	213b333b	20e9ce42
	81fe115f	7d8f90ad				

Polinom Tabanlı Gösterim :

G_x = 0fa c9dfcbac 8313bb21 39f1bb75 5fef65bc 391f8b36 f8f8eb73 71fd558b

G_y = 100 6a08a419 03350678 e58528be bf8a0bef f867a7ca 36716f7e 01f81052

Normal Gösterim :

s = 74d59ff0 7f6b413d 0ea14b34 4b20a2db 049b50c3

b = 1a0 03e0962d 4f9a8e40 7c904a95 38163adb 82521260 0c7752ad 52233279

G_x = 18b 863524b3 cdfefb94 f2784e0b 116faac5 4404bc91 62a363ba b84a14c5

G_y = 049 25df77bd 8b8ff1a5 ff519417 822bfedf 2bbd7526 44292c98 c7af6e02

283. Derece İkil Alan

T = 6

$p(t) = t^{283} + t^{12} + t^7 + t^5 + 1$

K-283 Eliptik Eğrisi

a = 0

r = 3885337784451458141838923813647037813284811733793061324295874997529815829704422603873

Polinom Tabanlı Gösterim :

G_x = 503213f 78ca4488 3f1a3b81 62f188e5 53cd265f 23c1567a 16876913 b0c2ac24 58492836

G_y = 1ccda38 0f1c9e31 8d90f95d 07e5426f e87e45c0 e8184698 e4596236 4e341161 77dd2259

Normal Gösterim:

G_x = 3ab9593 f8db09fc 188f1d7c 4ac9fcc3 e57fcd3b db15024b 212c7022 9de5fcd9 2eb0ea60

G_y = 2118c47 55e7345c d8f603ef 93b98b10 6fe8854f feb9a3b3 04634cc8 3a0e759f 0c2686b1

B-283 Eliptik Eğrisi

r = 77706755689029162836778476272940756265696259243769048891091965267700442777787378692871

Polinom Tabanlı Gösterim :

b = 27b680a c8b8596d a5a4af8a 19a0303f ca97fd76 45309fa2 a581485a f6263e31 3b79a2f5

G_x = 5f93925 8db7dd90 e1934f8c 70b0dfec 2eed25b8 557eac9c 80e2e198 f8cdbecd 86b12053

G_y = 3676854 fe24141c b98fe6d4 b20d02b4 516ff702 350eddb0 826779c8 13f0df45 be8112f4

Normal Gösterim:

s = 77e2b073 70eb0f83 2a6dd5b6 2dfc88cd 06bb84be

b = 157261b 894739fb 5a13503f 55f0b3f1 0c560116 66331022 01138cc1 80c0206b dafbc951

G_x = 749468e 464ee468 634b21f7 f61cb700 701817e6 bc36a236 4cb8906e 940948ea a463c35d

G_y = 62968bd 3b489ac5 c9b859da 68475c31 5bafcdc4 ccd0dc90
5b70f624 46f49c05 2f49c08c

409. Derece İkili Alan

T = 4
$p(t) = t^{409} + t^{87} + 1$

K-409 Eliptik Eğrisi

a = 0
r = 330527984395124299475957654016385519914202341482140609642324395022880711289249191050673258457777458014096366590617731358671

Polinom Tabanlı Gösterim:

G_x = 060f05f 658f49c1 ad3ab189 0f718421 0efd0987 e307c84c
27accfb8 f9f67cc2 c460189e b5aaaa62 ee222eb1 b35540cf
e9023746
G_y = 1e36905 0b7c4e42 acba1dac bf04299c 3460782f 918ea427
e6325165 e9ea10e3 da5f6c42 e9c55215 aa9ca27a 5863ec48
d8e0286b

Normal Gösterim:

G_x = 1b559c7 cba2422e 3affe133 43e808b5 5e012d72 6ca0b7e6
a63aeafb c1e3a98e 10ca0fcf 98350c3b 7f89a975 4a8e1dc0
713cec4a
G_y = 16d8c42 052f07e7 713e7490 eff318ba 1abd6fef 8a5433c8
94b24f5c 817aeb79 852496fb ee803a47 bc8a2038 78ebf1c4
99afd7d6

B-409 Eliptik Eğrisi

r = 661055968790248598951915308032771039828404682964281219284648798304157774827374805208143723762179110965979867288366567526771

Polinom Tabanlı Gösterim:

b = 021a5c2 c8ee9feb 5c4b9a75 3b7b476b 7fd6422e f1f3dd67
4761fa99 d6ac27c8 a9a197b2 72822f6c d57a55aa 4f50ae31
7b13545f
G_x = 15d4860 d088ddb3 496b0c60 64756260 441cde4a f1771d4d
b01ffe5b 34e59703 dc255a86 8a118051 5603aeab 60794e54
bb7996a7
G_y = 061b1cf ab6be5f3 2bbfa783 24ed106a 7636b9c5 a7bd198d
0158aa4f 5488d08f 38514f1f df4b4f40 d2181b36 81c364ba
0273c706

Normal Gösterim:

s = 4099b5a4 57f9d69f 79213d09 4c4bcd4d 4262210b
b = 124d065 1c3d3772 f7f5a1fe 6e715559 e2129bdf a04d52f7
b6ac7c53 2cf0ed06 f610072d 88ad2fdc c50c6fde 72843670
f8b3742a
G_x = 0ceacbc 9f475767 d8e69f3b 5dfab398 13685262 bcacf22b
84c7b6dd 981899e7 318c96f0 761f77c6 02c016ce d7c548de
830d708f
G_y= 199d64b a8f089c6 db0e0b61 e80bb959 34afd0ca f2e8be76
d1c5e9af fc7476df 49142691 ad303902 88aa09bc c59c1573
aa3c009a

571. Derece İkil Alan

T = 10
$p(t) = t^{571} + t^{10} + t^5 + t^2 + 1$

K-571 Eliptik Eğrisi

a = 0
r = 1932268761508629172347675945465993672149463664853217499328617625725759
57114478021226813397852270671183470671280082535146127367497406661731192
968242161709250355573368527 6673

Polinom Tabanlı Gösterim:

G_x =	26eb7a8	59923fbc	82189631	f8103fe4	ac9ca297	0012d5d4
	60248048	01841ca4	43709584	93b205e6	47da304d	b4ceb08c
	bbd1ba39	494776fb	988b4717	4dca88c7	e2945283	a01c8972
G_y =	349dc80	7f4fbf37	4f4aeade	3bca9531	4dd58cec	9f307a54
	ffc61efc	006d8a2c	9d4979c0	ac44aea7	4fbebbb9	f772aedc
	b620b01a	7ba7af1b	320430c8	591984f6	01cd4c14	3ef1c7a3

Normal Gösterim:

G_x =	04bb2db	a418d0db	107adae0	03427e5d	7cc139ac	b465e593
	4f0bea2a	b2f3622b	c29b3d5b	9aa7a1fd	fd5d8be6	6057c100
	8e71e484	bcd98f22	bf847642	37673674	29ef2ec5	bc3ebcf7
G_y =	44cbb57	de20788d	2c952d7b	56cf39bd	3e89b189	84bd124e
	751ceff4	369dd8da	c6a59e6e	745df44d	8220ce22	aa2c852c
	fcbbef49	ebaa98bd	2483e331	80e04286	feaa2530	50caff60

B-571 Eliptik Eğrisi

r = 3864537523017258344695351890931987344298927329706434998657235251451519
14228956042453614399938941577308313388112192694448624687246281681307023
45282883033324113931911052857 03

Polinom Tabanlı Gösterim:

b =	2f40e7e	2221f295	de297117	b7f3d62f	5c6a97ff	cb8ceff1
	cd6ba8ce	4a9a18ad	84ffabbd	8efa5933	2be7ad67	56a66e29
	4afd185a	78ff12aa	520e4de7	39baca0c	7ffeff7f	2955727a
G_x =	303001d	34b85629	6c16c0d4	0d3cd775	0a93d1d2	955fa80a
	a5f40fc8	db7b2abd	bde53950	f4c0d293	cdd711a3	5b67fb14
	99ae6003	8614f139	4abfa3b4	c850d927	e1e7769c	8eec2d19
G_y =	37bf273	42da639b	6dccfffe	b73d69d7	8c6c27a6	009cbbca
	1980f853	3921e8a6	84423e43	bab08a57	6291af8f	461bb2a8
	b3531d2f	0485c19b	16e2f151	6e23dd3c	1a4827af	1b8ac15b

Normal Gösterim:

s =	2aa058f7	3a0e33ab	486b0f61	0410c53a	7f132310	
b =	3762d0d	47116006	179da356	88eeaccf	591a5cde	a7500011
	8d9608c5	9132d434	26101a1d	fb377411	5f586623	f75f0000
	1ce61198	3c1275fa	31f5bc9f	4be1a0f4	67f01ca8	85c74777
G_x =	0735e03	5def5925	cc33173e	b2a8ce77	67522b46	6d278b65
	0a291612	7dfea9d2	d361089f	0a7a0247	a184e1c7	0d417866
	e0fe0feb	0ff8f2f3	f9176418	f97d117e	624e2015	df1662a8
G_y =	04a3642	0572616c	df7e606f	ccadaecf	c3b76dab	0eb1248d
	d03fbdfc	9cd3242c	4726be57	9855e812	de7ec5c5	00b4576a
	24628048	b6a72d88	0062eed0	dd34b109	6d3acbb6	b01a4a97

ÖZGEÇMİŞ

Doğum tarihi	24.05.1972	
Doğum yeri	Ankara	
Lise	1983 – 1990	Samsun Anadolu Lisesi
Lisans	1990 – 1994	Hacettepe Üniversitesi Mühendislik Fak. Bilgisayar Bilimleri Mühendisliği Bölümü
Yüksek Lisans	1996 – 1997	Univ. of Southwestern Louisiana, Graduate School Center for Advanced Computer Studies
Doktora	2000 – 2005	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı

Çalıştığı kurumlar

1994 – 1996	Ankara Emniyet Müdürlüğü Bilgi İşlem Şb. Md.
1998 – 1999	Kara Kuvvetleri Komutanlığı Bilgi Sistemleri D.
1998 – ...	YTÜ Elektrik-Elektronik Fak. Araştırma Görevlisi