İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

RSA ve ELGAMAL KISMİ HOMOMORFİK KRİPTO SİSTEMLERİN VERGİ ÖDEME SİSTEMİNE UYGULANMASI VE BU UYGULAMALARIN PERFORMANS ANALİZLERİ

YÜKSEK LİSANS TEZİ

Hasan ÇETİNKAYA

Bilişim Uygulamaları Anabilim Dalı

Bilgi Güvenliği Mühendisliği ve Kriptografi Programı

Tez Danışmanı: Doç. Dr. Enver ÖZDEMİR

HAZİRAN 2019

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

RSA ve ELGAMAL KISMİ HOMOMORFİK KRİPTO SİSTEMLERİN VERGİ ÖDEME SİSTEMİNE UYGULANMASI VE BU UYGULAMALARIN PERFORMANS ANALİZLERİ

YÜKSEK LİSANS TEZİ

Hasan ÇETİNKAYA

(707161007)

Bilişim Uygulamaları Anabilim Dalı

Bilgi Güvenliği Mühendisliği ve Kriptografi Programı

Tez Danışmanı: Doç. Dr. Enver ÖZDEMİR

HAZİRAN 2019

İTÜ, Bilişim Enstitüsü'nün 707161007 numaralı Yüksek Lisans Öğrencisi Hasan ÇETİNKAYA, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "RSA ve ELGAMAL KISMİ HOMOMORFİK KRİPTO SİSTEMLERİN VERGİ ÖDEME SİSTEMİNE UYGULANMASI ve BU UYGULAMALARIN PERFORMANS ANALİZLERİ" başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı:	**Doç. Dr. Enver ÖZDEMİR** İstanbul Teknik Üniversitesi	.....
Jüri Üyeleri :	**Doç. Dr. Ergün YARENERİ** İstanbul Teknik Üniversitesi	.....
	Yrd. Doç. Dr. Deniz SARIER TÜBİTAK	.....

Teslim Tarihi : 03 Mayıs 2019

Savunma Tarihi : 13 Haziran 2019

Aileme,

ÖNSÖZ

Yüksek lisans eğitimim ve tez süresi boyunca hiçbir konuda yardımını esirgemeyen danışmanım Doç. Dr. Enver Özdemir'e çok teşekkürlerimi borç bilirim.

Zorlu ve uzun bir yolculuk yaparak gerçekleştirdiğim yüksek lisans eğitimi süresince beni destekleyen yöneticilerime, çalışma arkadaşlarıma, özellikle eşime ve kızıma desteklerinden için teşekkür ederim.

Mayıs 2019 Hasan ÇETİNKAYA

İÇİNDEKİLER

Sayfa

KISALTMALAR

$\boldsymbol{M}$	**:** Açık Metin (Mesaj)
$\boldsymbol{C}$	**:** Kapalı Metin
$\boldsymbol{E}$	**:** Kapama İşlemi (Şifreleme)
$\boldsymbol{D}$	**:** Açma İşlemi (Şifreyi çözme)
$\boldsymbol{m}$	**:** Düz metin, şifrelenmemiş metin, şifrelenmemiş mesaj
$\boldsymbol{c}$	**:** Şifrelenmiş metin, şifreli metin, şifreli mesaj
$\boldsymbol{p_k}$	**:** Açık Anahtar
$\boldsymbol{s_k}$	**:** Gizli, kapalı Anahtar
NIST	**:** National Institute of Standards and Technology (TSE'nin karşılığı)
SGK	**:** Sosyal Güvenlik Kurumu
KİK	**:** Kamu İhale Kurumu
OVÖS	**:** Online Vergi Ödeme Sistemi
AES	**:** Advanced Encryption Standard; Gelişmiş Şifreleme Standardı
RSA	**:** Açık anahtarlı, bulanların isimlerinin baş harfiyle gösterilen kripto sistem.
RSA-n	**:** n bit Uzunluğunda Açık Anahtarlı RSA kripto sistem
ElGamal-n	**:** n bit Uzunluğunda Açık Anahtarlı ElGamal kripto sistem
FHE	**:** Tam Homomorfik Şifreleme
PHE	**:** Kısmi Homomorfik Şifreleme
SWHE	**:** Bir Dereceye Kadar Tam Homomorfik Şifreleme
$\boldsymbol{ebob}$	**:** Sayıların ortak bölenlerinin en büyüğü
$\boldsymbol{ekok}$	**:** Sayıların ortak katlarının en küçüğü
$\boldsymbol{ord}$	**:** Grubun veya grup elemanın mertebesi, derecesi
Enc	**:** Encryption (Şifreleme), ifadesinin kısaltılmış hali.
e-imza	**:** Elektronik imza, dijital imza
e-voting	**:** Elektronik oylama
sdk	**:** Java'nın Standart Development Kit kütüphanesi
Mikro saniye	**:** Saniyenin Milyonda Biri
Mili saniye	**:** Saniyenin Binde Biri

API **:** Application Programming Interface (Uygulama Programlama Ara yüzü)

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

RSA VE ELGAMAL KISMİ HOMOMORFİK KRİPTO SİSTEMLERİN VERGİ ÖDEME SİSTEMİNE UYGULANMASI VE BU UYGULAMALARIN PERFORMANS ANALİZLERİ

ÖZET

Açık anahtarlı kripto sistemlerin Diffie ve Hellman tarafından 1976'da bulunmasıyla [1], dijital verinin gizliliği önem kazanmış, özellikle internetin iş ve özel hayatta vazgeçilmez bir hâl almasıyla bu önem daha da artmıştır. Bunun sonucunda 1978'ten bu yana geçen yıllarda birçok güvenli kripto sistemler bulunmuştur. İnternet uygulamalarında, örneğin çevrimiçi (online) bankacılık sisteminde, elektronik oylama sisteminde, çevrimiçi (online) vergi ödeme sisteminde ve ayrıca özellikle son yıllarda popülaritesi artan bulut bilişimde güvenli veri iletimini sağlamanın yollarından biri homomorfik şifrelemedir.

Homomorfik kripto sistemler ilk olarak Rivest, Adleman ve Dertouzous tarafından 1978 yılında yayımlanan makalede ele alınmıştır [2]. Makalelerinde, şifreli veriler deşifre edilmeden bir birtakım matematiksel işlemler yapılabileceğini göstermişlerdir. Bu makalenin yayımlanmasıyla homomorfik kripto sistemler üzerine çalışmalar başlamıştır.

Homomorfik şifrelemenin temel amacı haberleşmedeki verilerin ve veri tabanlarında veya bulut bilişimindeki verilerin gizliliğini sağlamaktır. Geçen süre zarfında birçok çarpmaya göre homomorfik (ElGamal [3] 1984, RSA [2]) veya toplamaya göre (Paillier [4] 1999, Goldwasser- Micali 1984 [5]) kısmi homomorfik kripto sistemler geliştirilmiştir. 1991 yılında Feigenbaum yayınlamış olduğu bir makaleyle [6] tam homomorfik kripto sistemlere olan ilgi artmıştır. Makalesinde; "Acaba Enc(x+y) ve Enc(x.y) ifadelerini Enc(x) ve Enc(y) türünden ifade eden Enc() şifreleme metoduna sahip kripto sistem geliştirilebilir mi?" şeklindeki sorusunu 2009 yılında Craig Gentry, yayınladığı doktora tezinde ispatlamış ve kendisi tam homomorfik kripto sistemini [7] geliştirerek Feigenbaum' un sorusunu yanıtlamıştır.

Bu tezde RSA ve ElGamal açık anahtarlı kripto sistemlerine genel bir bakıştan sonra homomorfik özellikleri gösterilecek ve işbu özellikler Java programlama dilinde gerçeklenecektir. Sonrasında çevrimiçi (online) vergi ödeme sisteminde her ikisinin homomorfik uygulaması ele alınıp performans karşılaştırılması yapılacaktır. Ancak bunları daha iyi kavrayabilmek için öncelikle tezimizin ilk bölümünde "Sayılar Teorisi ve Soyut Cebirdeki Matematiksel Kavramlar konusu üzerinde durulacaktır.

İkinci bölümde, açık anahtarlı (asimetrik) kripto sistemlerden ilki olan RSA ve ElGamal ele alınacak ve sonrasında günümüzdeki kullanım alanlarından biri olan, elektronik imzadan (*e-imza*) bahsedilecektir.

Üçüncü bölümde ise, tezimizin asıl konusu olan homomorfik (eş şekilli) şifrelemenin açık anahtarlı kripto sistemler için matematiksel tanımı yapılarak homomorfik şifrelemenin türleri anlatılacak ve sonrasında bunlara örnekler verilecektir. Ayrıca 2010 yılı sonrasında geliştirilen tam homomorfik kripto sistemlerden bahsedilecektir.

Tezin dört ve beşinci bölümünde ise RSA ve ElGamal'ın çarpımsal homomorfik özellikleri matematiksel olarak gösterilecek, akabinde bu özellikler Java'da gerçekleşecektir.

Tezin son bölümünde ise RSA ve ElGamal'ın homomorfik özellikleri vergi ödeme sistemine uygulanacak; her iki kripto sistemin çarpımsal homomorfik özelliklerini performans açısından karşılaştırılacaktır. Uygulamamız, değişik anahtar uzunlukları, 1024-bit, 2048-bit ve 3072-bit kullanılarak Java'da gerçeklenecek ve bu anahtarların her birinde, şifreleme, deşifreleme ve homomorfik işlemlerin toplam gecikme süreleri beş ayrı deneme yapılarak hesaplanacaktır.

Tezde yapılan araştırmada, RSA ve ElGamal açık anahtarlı kripto sistemlerin her ikisi de çarpımsal homomorfik özellikleri sağlamakta olup, yapılan karşılaştırmalar neticesinde RSA'nın daha performanslı ve kullanışlı olduğu sonucuna varılmıştır. RSA, ElGamal kripto sistemine göre, şifreleme, deşifreleme ve homomorfik işlemlerin bütününde yaklaşık 4 daha hızlı olduğu görülmüştür. Kullanılan anahtar boyutu ne olursa olsun, RSA ve ElGamal ile şifrelenen aynı metin için, ElGamal da elde edilen şifreli metin, RSA da elde edilen şifreli metinin iki katıdır. Bu da ElGamal için ayrı bir dezavantajdır. Kurumlar (tüzel kişiler) için vergi ödeme sisteminde avantajlı olan RSA açık anahtarlı kripto sistem aynı zamanda, gerçek kişiler SGK dan maaş aldıkları için vergi ödeme sistemi olarak SGK içinde uygulanabilir. Sadece K.İ.K yerine SGK

kullanılarak, gerçek kişilerin ödemesi gereken vergi miktarları hesaplanabilir. Böylelikle kişilerin maaşları sadece kendileri tarafından bilinir ve ödemesi gereken vergi miktarı da ancak kendisi tarafından bilinir.

APPLICATION OF RSA AND ELGAMAL PARTIAL HOMOMORPHIC CRYPTO SYSTEM TO TAX PAYMENT SYSTEM AND PERFORMANCE ANALYSIS OF THESE APPLICATIONS

SUMMARY

It is very important that personal data is only known by the authorized people. For example, it is very important to know and display the person's health information, salary information, the tender information that companies have made and the amount of tax that they need to pay. In order to ensure the privacy of these data, it must be interrogated by the web service, encryption of data in the application server, and then the mathematical operations, such as multiptication on the encrypted data must be performed on the client side. In addition to this, decryption of ciphertext by using private key is performed on the client side. Hence, privacy of data is satisfied.

With the presentation of public key crypto systems by Diffie and Hellman in 1976 [1], the confidentiality of digital data has become more important, especially when the Internet becomes indispensable in business and private life. As a result, many secure crypto systems have been developed in the years since 1978. Homomorphic encryption is one of the ways to ensure secure data transmission in Internet applications, for example in the online banking system, keeping medical records,in the electronic voting system, in the online tax payment system, and also in the growing cloud computing, especially in recent years. The execution of transactions on the encrypted data may prevent the environment of mistrust and ensure data confidentiality. At this point, using a homomorphic encryption during processing of the data, operations can be performed without the need for decryption, and only the user can see the decrypted result of the operations. A reliable calculation chain can be created by performing different services in different companies by means of homomorphic encryption.

Homomorphic crypto systems were first discussed in the article published by Rivest, Adleman and Dertouzous in 1978 [2]. In that article, they showed that some mathematical operations can be performed without deciphering encrypted data.

Studies on homomorphic crypto systems have begun with the publication of this article.

The main purpose of homomorphic encryption is to ensure the privacy of data during communication and the confidentiality of data in databases or in public cloud systems. In the meantime, several partial homomorphic crypto systems have been developed, for example methods which are homomorphic for only multiplication operation (ElGamal [3] 1984, RSA [2]) or for only addition (Paillier [4] 1999, Goldwasser-Micali 1984 [5]). With the article published in 1991 by Feigenbaum [6] the interest in fully homomorphic crypto systems increased. In that article the author asked the following question; "Is it possible to develop a crypto system with Enc() encryption function that expresses Enc(x + y) and Enc (x.y) in terms of Enc(x) and Enc(y)?". In 2009, Craig Gentry, in his doctoral thesis, proved the existence of such crypto systems, and he developed the fully homomorphic crypto system [7] and answered Feigenbaum's question.

In this thesis, RSA and ElGamal will be investigated and their homomorphic features will be presented right after giving an overview of public key crypto systems. The implementation will be performed in Java programming environment. Then, in the online tax payment system, the homomorphic application of both will be discussed and performance comparison will be presented. However, in order to get a better understanding of them, we will first begin by explaining mathematical concepts in number theory and abstract algebra in the first part of our thesis.
In the second part, RSA and ElGamal, the first of the open-key (asymmetric) crypto systems, will be discussed, and then one of the current usage areas, electronic signature or digital signature will be mentioned.

In the third chapter, the mathematical definition of homormorphic encryption, which is the main subject of our thesis, with public key cryptosystem, will be given. Full homomorphic crypto systems developed after 2010 will be discussed.

In the four and fifth part of the thesis, the mathematical properties of RSA and ElGamal are shown mathematically. In the last part of the thesis, we apply the homomorphic properties of RSA and ElGamal to our tax payment system and compare the multiplicative homomorphic characteristics of both cryptosystems in terms of performance. Our application will be implemented in Java using different key lengths, 1024-bits, 2048-bits and 3072-bits, and in each of these switches, the total latency of

encryption, decoding and homomorphic operations will be calculated by five separate trials.

As a result of the research, both RSA and ElGamal public-key crypto systems provide both multiplicative homomorphic properties and it was concluded that RSA is more efficient and more useful in terms of encryption, decryption and homomorphic multiplicative operations on encrypted texts. In general, RSA pulic key crypto system seems to have been about four times faster than ElGamal public key crypto system. Moreover, in the same key size, the RSA is four times faster, and also the encrypted text is half that of the encrypted text in ElGamal. Our tax payment system, which is developed by using homomorphic crypto systems, can be applied to the Social Security Institution for salary payment. Thus, salaries of individuals are known only by themselves and the amount of tax that they have to pay is known only by themselves.

1. GİRİŞ

Bu bölümde açık anahtarlı kripto sistemleri ve bunların homomorfik özelliklerini anlatılacaktır. Bu kapsamda sayılar teorisi ve soyut cebirdeki tanım ve teoremlerden yararlanılacak olup; gerektiğinde örneklendirmeler yapılacaktır. Konuyla ilgili ispatlara girilmeyecektir. Özellikle grup, halka ve homomorfizma kavramları, homomorfik şifreleme modelleri, kısmi ve tam homomorfik şifreleme için önem arz etmektedir.

1.1. Grup

Tanım: G boş olmayan bir küme ve $*$, G de bir ikili işlem olsun.

$(G,*)$ cebirsel ifadesi eğer aşağıdaki üç koşulu gerçekliyorsa grup denir.

$G \times G \to G,\ (a,b) \to\ a * b$ olmak üzere,

$i)$ *Birleşme Özelliği:* $\forall\ a,b,c \in G\ için\ a * (b * c)\ = (a * b) * c$

$ii)$ *Etkisiz Elemanın varlığı*: $\exists\ e \in G, \forall\ a \in G\ için\ a * e = e * a$

$iii)$*Ters Elemanın varlığı*: $\forall\ a\ \in G\ için \exists\ a' \in G$ $öyle\ ki\ a' * a = a * a'\ = e$

$(G,*)$ grup ve $*$ işleminin değişme özelliği varsa, yani $a * b = b * a$ ise değişmeli grup denir.

Verilen herhangi bir sonlu G grubu için eleman sayısına grubun mertebesi (derecesi) denir ve $|G|$ ile gösterilir.

Eğer $H,\ G$'nin bir alt grubu ise $|H|$ ifadesi $|G|$'yi böler. Yani H nin mertebesi, G'nin mertebesini böler. Diğer bir ifadeyle, G'nin mertebesi H'nin mertebesinin bir tamsayı katıdır.

Bir grup G sonlu sayıda eleman içeriyorsa, grubun eleman sayısına grubun derecesi denir ve $|G|$ ile gösterilir.

$a \in G$ olmak üzere, $a^n\ = e$ ise a ya sonlu dereceli eleman denir ve en küçük n sayısına a nın mertebesi (derecesi) denir ve $|a| = n$ ile gösterilir.

$H,\ G$'nin bir alt kümesi ise H bir alt-gruptur.

Devirli Grup

Tanım: G grup olsun a grubun elemanı olmak üzere

$\langle a\rangle = \{\ a^n\ ,\ n \in \mathbb{Z}\}$ kümesine a tarafından üretilen G'nin bir devirli alt grubu denir.

Eğer $G = \langle a\rangle$ ise G ye a tarafından üretilen devirli grup denir ve a ya G nin üreteci denir.

Teorem (Lagrange)

G sonlu grup ve $H,\ G$' nin alt grubu olsun.

Bu durumda H nin mertebesi(derecesi) G'nin mertebesini (derecesini) böler.

Özellikle herhangi bir $x \in G$'nin derecesi G grubunun derecesini böler.

Yarı Grup

$(G,*)$ cebirsel yapısı,

■ $(*)$ işlemi kapalılık özelliğini sağlar.

■ $(*)$ işlemi birleşme özelliğini sağlar.

■ G grubunun $(*)$ işlemine göre etkisiz(birim) elemanı vardır.

■ G grubunda her elemanın $(*)$ işlemine göre tersi vardır.

Yukarıda verilen, Grup tanımından sadece ilk ikisini sağlıyorsa yarı grup denir.

1.2. Halkalar

Halkalar, gruplardan farklı olarak iki işlemli cebirsel yapılardır.

Tanım: $(R,+,\ \cdot)$ iki işlemli cebirsel yapı, aşağıdaki özellikleri sağlıyorsa bu cebirsel yapıya halka denir.

■ $(R,+)$ bir değişmeli gruptur.

■ $(R,\ \cdot)$ bir yarı gruptur.

■ $(\cdot)$ işleminin $(+)$ işlemi üzerine sağdan ve soldan dağılma özelliği vardır. Yani,

$\forall\ a,b,c \in R$ için

$a \cdot (b+c) = a \cdot b + a \cdot c$ (soldan dağılma) (1.1)

$(a+b)\cdot c = a\cdot c + b\cdot c$ (sağdan dağılma) (1.2)

1.3. Homomorfizmalar

1.3.1. Grup homomorfizması:

Grup işlemlerini koruyan fonksiyonlara grup homomorfizması denir.

Tanım: $(G,\odot)$ ve $(G',\otimes)$ iki grup olmak üzere,

$f: G \to G'$ bir fonksiyon olsun.

Eğer, $\forall\, a,b \in G$ için,

$$f(a \odot b) = f(a) \otimes f(b) \quad (1.3)$$

eşitliği sağlanıyorsa, bu durumda f'ye grup homomorfizması denir.

Örnek1: Üstel fonksiyonlar tanımlı oldukları grupta bir, grup homomorfizmasıdır.

$exp: (\mathbb{R},+) \to (\mathbb{R},\ \cdot)$, x ve $y \in \mathbb{R}$ olmak üzere

$exp(x+y) = e^x \cdot e^y = exp(x)\cdot exp(y)$

Örnek2:

$f: \mathbb{Z} \to \mathbb{Z}$ olmak üzere

$f(x) = 2\cdot x$, fonksiyonu bir grup homomorfizmasıdır. Çünkü

$f(x+y) = 2\cdot(x+y) = 2\cdot x + 2\cdot y = f(x) + f(y)$ dir.

1.3.2. Halka homomorfizması

Halka işlemlerini koruyan fonksiyonlara, halka homomorfizması denir.

R ve S iki halka, $(+,\ \cdot)$ işlemleri üzerinde halka oluştursunlar.

$f: R \to S$, bir fonksiyon olsun. Eğer $\forall\, a,b \in R$ için

$f(a+b) = f(a) + f(b)$

$f(a\cdot b) = f(a)\cdot f(b)$

eşitlikleri sağlanıyorsa, f fonksiyonuna halka homomorfizması denir.

Örnek:

$n \in \mathbb{Z}^+$ olmak üzere,

$f(a)=\bar{a}$ ile tanımlı fonksiyon (kalan fonksiyonu)

$f: \mathbb{Z} \rightarrow \mathbb{Z}_n$ fonksiyonu halka homomorfizmasıdır.

$f(a+b)=\overline{(a+b)} =\bar{a} +\bar{b} = f(a) + f(b)$

$f(a \cdot b)=\overline{(a \cdot b)} =\bar{a} \cdot \bar{b} = f(a) \cdot f(b)$

eşitlikleri sağlanır.

$\therefore f$ fonksiyonu halka homomorfizmasıdır.

1.4. Denklik Bağıntıları

Açık anahtarlı kripto sistemlerin birçoğunda denklik bağıntı teorisi önemli bir yere sahiptir.

$S \times S$ Kartezyen çarpım kümesinin alt kümelerine R bağıntısı denir ve (x, y) ikilileri $x\ R\ y$ ile gösterilir.

Tanım: R bir A kümesi üzerinde bağıntı olsun. Eğer

$i) \forall\ x \in A$ için $(x, x) \in R$ ise o zaman R ye yansıma özelliğine sahiptir,

$ii) (x, y) \in R$ şartını sağlayan $\forall\ x, y \in A$ için $(y, x) \in R$ ise o zaman R simetri özelliğine sahiptir,

$iii) (x, y) \in R$ ve $(y, x) \in R$ şartlarını sağlayan $\forall\ x, y \in A$ için $x = y$ oluyorsa R' ye ters simetri özelliğine sahiptir,

$iv) (x, y) \in R$ ve $(y, z) \in R$ şartlarını sağlayan $\forall\ x, y, z \in A$ için $(x, z) \in R$ ise o zaman R'ye geçişme özelliğine sahiptir denir.

Eğer R bağıntısı bu özelliklerden ilk üçünü sağlıyorsa R ye denklik bağıntısı denir.

Tanım: n pozitif tamsayı olmak üzere; $a, b \in \mathbb{Z}$

$a \equiv b\ (mod\ n)$ eğer $n \mid (a-b)$ ise a ve b denktir modül n denir.

Genel bir deyişle, $a - b = k \cdot n,\ k \in \mathbb{Z}$

Modül bağıntısının yukarıdaki ilk üç özelliği sağladığını çok rahatlıkla gösterebiliriz.

Yansıma: $a \equiv a\ (mod\ n)$

Simetri: Eğer $a \equiv b\ (mod\ n)$ ise $b \equiv a\ (mod\ n)$

Geçişme: Eğer $a \equiv b\ (mod\ n)$ ve $b \equiv c\ (mod\ n)$ ise, $a \equiv c\ (mod\ n)$ dir.

Tanım: Verilen x tamsayısının denklik sınıfı

$\overline{x} = \{y \in \mathbb{Z} \mid y \equiv x\ (mod\ n)\} = x + n\mathbb{Z}$,

Ayrıca modül n de tamsayılar;

$\mathbb{Z}_n = \{0,1,2 \ldots n-1\}$ biçiminde gösterilir. Ve $|\mathbb{Z}_n| = n$ dir.

$\mathbb{Z}_n$ kümesine n ye bölümünden kalanlar kümesi denir.

Matematik bölümünü önemli çokça kullanılacak bir grup yapısıyla bitirelim.

p bir asal sayı olmak üzere (bir sonraki kısımda asal sayı tanımı verilecek)

$\mathbb{Z}_p{}^* = \{1,2,3 \ldots p-1\}$ kümesi çarpma işleminde bir grup oluşturur. Çünkü grup oluşturmadaki üç kriteri sağlar. $(\mathbb{Z}_p{}^*, \cdot)$ ile gösterilir. ($*$ işaretinin anlamı kalanlar kümesinde 0 elemanı hariç demektir)

Bunu bir örnekle görelim.

Örnek: $\mathbb{Z}_7{}^* = \{1,2,3 \ldots 6\}$ kümesinin etkisiz elemanı 1 dir.

Kapalılık özelliğini sağlar aşikardır.

Her elemanında çarpmaya göre tersi vardır. Çünkü $\forall$ a $\in \mathbb{Z}_7{}^*$ için a' nın tersi vardır.

$(a, 7) = 1$ yani a ile 7 aralarında asaldır.

$4^{-1} = 2$, çünkü $4 \cdot 2 \equiv 1\ (mod\ 7)$. Benzer şekilde, $\mathbb{Z}_7{}^*$ deki diğer elemanların tersi,

$2^{-1} = 4$, $3^{-1} = 5$, $5^{-1} = 3$ ve $6^{-1} = 6$ dir.

1.5. Sayılar Teorisinden Bazı Teoremler

Tanım: $n \in \mathbb{Z}^+$ için, n den küçük ve n ile aralarında asal olan sayıların sayısı $\varphi(n)$ ile gösterilir ve Euler Phi fonksiyonu olarak adlandırılır.

Özellikleri:

$i)$ p asal sayı ise $\varphi(p) = p - 1$

$ii)$ $ebob(m,n) = 1$ ve $m, n \in \mathbb{N}$ ise $\varphi(m \cdot n) = \varphi(m) \cdot \varphi(n)$

$n = {p_1}^{e_1} \cdot {p_2}^{e_2} \ldots {p_n}^{e_n}$ ise $\varphi(n) = \mathrm{n}\left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \ldots \left(1 - \frac{1}{p_n}\right)$

Teorem (Euler): $n \geq 2$ tamsayı olmak üzere, $a \in \mathbb{Z}_n$ ve $ebob(a,b) = 1$ ise

$$a^{\varphi(n)} \equiv 1 \ (mod\ n) \tag{1.4}$$

Teorem (Fermat): $ebob(a,p) = 1$ ise $a^{p-1} \equiv 1 \ (mod\ p)$

Aslında Fermat teoremi Euler teoreminin özel halidir. n nin asal halidir.

Tanım: p , 1'den büyük pozitif tamsayı olsun. Eğer p nin 1 ve p den başka pozitif böleni yoksa p ye bir asal sayı denir. 1'den büyük ve asal olmayan tamsayılara ise birleşik sayı denir. Sonsuz sayıda asal sayı vardır.

0 dan büyük her tamsayının en az bir asal böleni vardır.

Teorem: Eğer n birleşik sayı ise n nin $\sqrt{n}$ yi geçmeyen bir asal çarpanı yani böleni vardır.

Tanım: a, b iki tamsayı olmak üzere $(a,b) = 1$ yani $ebob(a,b) = 1$ ise a ve b ye aralarında asal denir.

Teorem (Öklid Algoritması): a, b tamsayılar ve $a > 0$ olsun. Bölme algoritması art arda uygulanarak aşağıdaki eşitlikler elde edilir;

$$b = q_0 \cdot a + r_0$$
$$a = q_1 \cdot r_0 + r_1$$
$$\vdots$$
$$r_{n-1} = q_{n+1} \cdot r_n + r_{n+1}$$

elde edilir ve r_{n+1} =0 olduğunda r_n ifadesi $ebob(a,b)$ ye eşit olur.

Öklit algoritması modüler aritmetikte ters elemanı bulmak için kullanılır. Aynı zamanda, bilgisayar programlarında iki sayının en büyük ortak bölenini bulmak için yani $ebob$ larını bulmak kullanışlı bir algoritmadır.

Teorem (Çin Kalan Teoremi -Chinese Remainder Theorem):

Aralarında asal $n_1, n_2, n_3 \ldots n_k$ tamsayıları için:

$x \equiv a_1 \ (mod\ n_1)$

$x \equiv a_2 \ (mod\ n_2)$

$\vdots$

$x \equiv a_k \ (mod\ n_k)$

ile verilen denklik sisteminin $(mod\ n)$ de, yalnız tek bir çözümü vardır ve

$n = n_1 . n_2 . n_3 \ldots n_k$, dir

1.6. Ayrık Logaritma Problemi:

$\mathbb{Z}_p^*$=$\{1,2,3, \ldots . , p-1\}$ olmak üzere, burada p tamsayısı asal bir tamsayıdır.

$\mathbb{Z}_p^*$ de, g ve y verilmiş olsun. Yani $g, y \in \mathbb{Z}_p^*$ olsun.

Bu durumda,

$g^x \equiv$ y $(mod\ p\ )$ (1.5)

Denkliğini sağlayan x değerinin bulmak çok zahmetli olup, bu probleme ayrık logaritma problemi denir. Bir örnekle açıklayalım.

Örnek:

$\mathbb{Z}_{17}^*$ de, $g = 3\ ve\ y = 11$ değerlerini alalım.

$3^x \equiv 11\ (mod\ 17\ )$, denkliğinde x değerini bulmaya çalışalım.

$\mathbb{Z}_{17}^*$=$\{1,2,3 \ldots 16\}$ kümesindeki tüm değerler denenir. p asalının çok büyük değerleri için bu problemi çözmek oldukça vakit alır. Verilen denklikte, 1'den başlayarak x değerlerini yerine koyalım bu işleme ta ki denklik sağlanana kadar devam etmeliyiz.

$3^1 \equiv 3\ (mod\ 17)$ $3^2 \equiv 9\ (mod\ 17)$

$3^3 \equiv 10\ (mod\ 17)$ $3^4 \equiv 13\ (mod\ 17)$

$3^5 \equiv 5\ (mod\ 17)$ $3^6 \equiv 15\ (mod\ 17)$ $3^7 \equiv 11\ (mod\ 17)$ bulunur.

$x = 7$ için yukarıdaki denklem sağlanmış olur.

Görüldüğü gibi, iki asalın çarpımından oluşan bileşik sayıların hangi iki asalın çarpımından oluştuğuna dair şu an bir algoritma olmadığı gibi ayrık logaritma probleminde x değerini bulan bir matematik algoritması şimdilik yoktur.

1.7. Tek Yönlü Fonksiyon ve Ek Bilgili (Trapdoor) Tek Yönlü Fonksiyon

$f: X \to Y$ bire-bir fonksiyon olsun.

f fonksiyonu aşağıdaki özellikleri sağlıyorsa f ye tek yönlü fonksiyon denir.

■ $\forall x \in X$ için $f(x)$ değerini hesaplamak kolaydır. (Bilgisayarda yazılacak bir algoritma(program) ile hesaplanması kolaydır.)

■ $y = f(x)$ olmak üzere, y değeri verildiğinde x değerini hesaplamak zordur. (Bilgisayarda bilinen algoritmalar kullanılarak hesaplamak güç, imkansızdır.)

$f: X \to Y$, tek yönlü bir fonksiyon olsun.

Ek bilgi (trapdoor, arka kapı) kullanılarak $y = f(x)$ eşitliğindeki y değeri verildiğinde, x değerini hesaplamak kolay oluyorsa f fonksiyonuna ek bilgili (trapdoor) tek yönlü fonksiyon denir.

Örnek1:

p ve q asal sayılar olmak üzere,

$f(p, q) = p \cdot q = N$

fonksiyonu ek bilgili (trapdoor) fonksiyondur. Çünkü p ve q verildiğinde, N birleşik tamsayısını hesaplamak kolaydır. Fakat büyük p ve q asalları için, N birleşik tamsayı değeri verildiğinde bilinen matematiksel algoritmalar kullanılarak, p ve q asallarını bulmak zordur.

Eğer ek bilgi olarak, $\varphi(N)$ değeri verilirse, yani N birleşik tamsayının Euler phi fonksiyonu değeri verilirse, p ve q asalları bulunabilir.

Örnek2:

p , asal sayı olmak üzere

$\left(\mathbb{Z}_p^*\right)$ devirli çarpımsal bir gruptur.

$a \in \mathbb{Z}_p^*$ ve primitif kök olsun.

$a^x \equiv b \ (mod\ p)$ denkliğinde, (1.6)

a , b ve p değerleri bilinirken, $0 \leq x \leq p-2$ olacak biçimde x değerinin bulmak zordur.

Bu problem ayrık logaritma problemi olarak adlandırılır.

Tanım:

G sonlu bir çarpımsal grup olsun. $g \in G$ olmak üzere g nin mertebesi(derecesi)

g^m=1 eşitliğini sağlayan en küçük m pozitif tamsayısıdır.

Örnek:

$\mathbb{Z}_{11}^{*} = \{1,2,3 \ldots 10\}$ grubu verilsin.

Grubun mertebesi (derecesi) 10 dur. Yani grubun eleman sayısı. $|\mathbb{Z}_{11}| = 10$ ile gösterilir.

$2^m \equiv 1 \ (mod\ 11)$ sağlayan m pozitif tamsayısı,

m pozitif tamsayısı grubun derecesini böler (Lagrange Teoremi).

Yani m nin alabileceği değerler kümesi $\{1,2,5,10\}$ dir.

Hesaplamalar yapıldığında,

$2^{10} \equiv 1(mod\ 11)$ olduğu görülür.

Bu durumda, $2 \in \mathbb{Z}_{11}^{*}$ in mertebesi 10 , $ord(2) = 10$ şeklinde gösterilir.

Şimdi de $ord(3)$ ifadesinin kaça eşit olduğuna bakalım.

$3^m \equiv 1 \ (mod\ 11) \Rightarrow m = 5$ olduğu bulunur.

$3 \in \mathbb{Z}_{11}^{*}$ in mertebesi 5 tir ve 5 sayısı grubun mertebesini, yani 10'u tam böler.

Lagrange Teoremi:

G mertebesi (derecesi) n olan çarpımsal bir grup ve $g \in G$ olsun. Bu durumda g nin mertebesi G nin mertebesini böler.

Tanım: p asal bir sayı olmak üzere, mertebesi $p - 1$ olan $g \in \mathbb{Z}_p^{*}$ sayısına ilkel kök (primitif kök) denir.

Not: p modunda ilkel köklerin sayısı $\varphi(p-1)$ dir.

Örnek:

$\mathbb{Z}_{11}^{*}$'de $2^{10} \equiv 1(mod\ 11)$ olduğundan, $2 \in \mathbb{Z}_{11}^{*}$ ilkel köktür.

Diğer ilkel kökler; 6, 7 ve 8'dir.

Bu durumda ilkel köklerin kümesi $\{ 2, 6, 7, 8 \}$ ve

$$\varphi(11-1) = \varphi(10) = \varphi(5) \cdot \varphi(2) = 4 \cdot 1 = 4$$

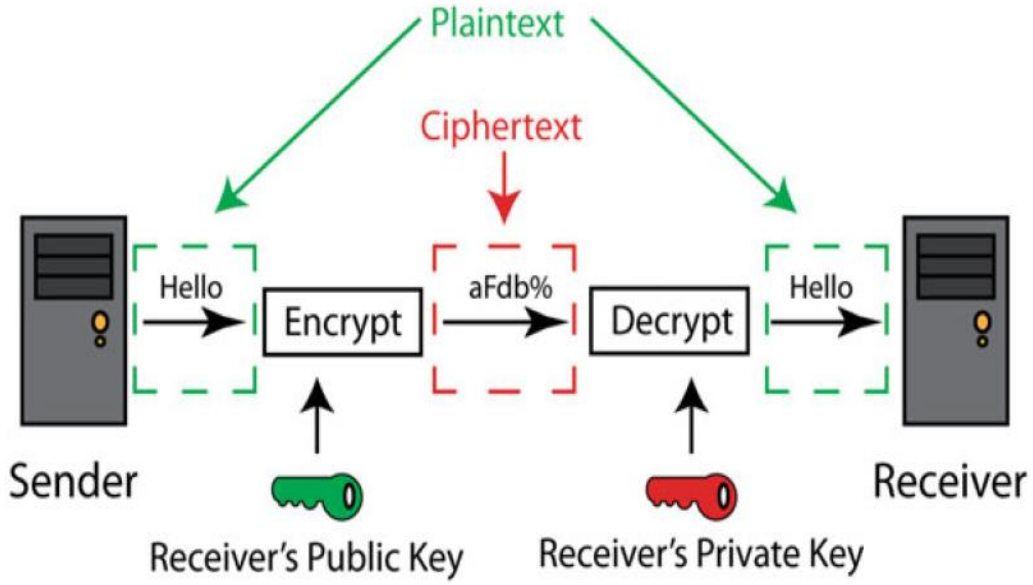
bize ilkel kökler kümesinin eleman sayısını verir.

2. AÇIK ANAHTAR KRİPTO SİSTEMLER (ASİMETRİK ŞİFRELEME)

2.1. Açık Anahtar Şifreleme Modeli

Şifreleme tarihinin başlangıcında, iki taraf güvenli bir şekilde paylaştıkları bir anahtarla, düz metinleri şifreleyip, şifrelerken kullandıkları anahtarla şifreli metinleri çözerek haberleşiyorlardı. Bu anahtarın gönderici ve alıcı tarafından gizli tutulması gerekiyordu. Bu şifreleme modeli, simetrik şifreleme olarak adlandırılır. Simetrik anahtarlı şifrelemede hem şifreleme hem de şifre çözme için aynı anahtar kullanılmaktadır [8]. Açık anahtar şifreleme, diğer bir deyişle asimetrik şifreleme, simetrik şifreleme modelinin anahtar dağıtımı problemini çözmüştür. Açık anahtar kripto sistemlerde paylaşılan anahtar üzerinde önceden anlaşmadan haberleşmek mümkündür. Açık anahtar kriptografisi, temelde herkesin birbirlerini tanımadan bile gizli bir şekilde haberleşmesi demektir. Bu tanım kriptografi için dönüm noktası olmuştur. Açık Anahtar şifreleme modeli, aynı zamanda asimetrik şifreleme olarak da adlandırılır. Aşağıdaki şekilde 2. 1'de gösterilen açık anahtar şifreleme modeli, ilk kez 1976 yılında Whitfield Diffie ve Martin Hellman [1] ve bu iki bilim insanının çalışmasından etkilenip açık anahtar dağıtımı üzerine çalışan Ralph Merkle tarafından önerilmiştir.

Daha önce de belirttiğimiz gibi açık anahtar şifreleme, asimetrik şifreleme olarak da adlandırılmaktadır. Bu nedenle bazen açık anahtar şifreleme yerine asimetrik şifreleme tabirini de kullanacağız. Asimetrik şifreleme sistemlerinde, simetrik şifreleme sistemlerindeki gibi güvenli yollardan bir anahtar dağıtım işlemine gerek duyulmaz. Hatta asimetrik şifreleme, simetrik şifreleme sistemlerinde kullanılacak gizli (kapalı) anahtarın iletimi için kullanılır. Birçok şifreli mesajlaşma uygulamasında, örneğin WhatsApp'ta, oturumda kullanılacak gizli (kapalı) anahtar karşıdakinin telefonuna asimetrik şifreleme ile şifrelenerek iletilir. Sonrasında da mesajlaşma iletilen anahtar kullanılarak simetrik şifreleme kullanılarak mesajlaşma yapılır.



Şekil 2.1:Açık Anahtar Şifreleme-Deşifreleme Modeli [25].

Asimetrik şifreleme sistemlerinde, kripto sistemlerinde yukarıdaki şekilde de görüldüğü üzere iki ayrı anahtar bulunur. Bunlardan biri gizli, diğeri ise açık anahtardır. İsimlerinden de anlaşılacağı üzere, bu anahtarlardan biri gizli tutulur, diğeri ise herkese gösterilir. Fakat bu iki anahtar birbiriyle matematiksel olarak ilişkilidir. Yani matematiksel bir algoritma ile açık anahtardan gizli anahtar bulunur. Açık anahtarlar, düz metinleri, mesajları şifrelemek için kullanılırken, gizli anahtarlar şifreli metinleri çözmek için kullanılırlar. Asimetrik şifreleme denmesinin nedeni, şifreleme ve deşifrelemede farklı anahtarların kullanılmasıdır. Simetrik şifreleme modelinde ise şifreleme ve deşifreleme için aynı anahtar kullanılır.

Açık Anahtarlı Kripto sistemler genel olarak;

K: anahtar uzayı

M: düz metin

C: şifreli metin

olmak üzere üç aşamadan oluşur. Bunlar anahtar üretimi, şifreleme algoritması ve şifre çözme (deşifre) algoritmasıdır. Şimdi sırasıyla bunların kısaca tanımlarını ve gösterimlerini verelim.

Anahtar üretim algoritması (KG):

(p_k, s_k) anahtar çifti üretilir. Burada s_k, gizli anahtar, p_k ise açık anahtardır. Gizli anahtar s_k, sadece anahtar sahibi tarafından bilinirken, p_k açık anahtarı ise herkese yayınlanır, gösterilir. Çünkü düz metinler, mesajlar alıcının p_k sı ile şifrelenip alıcaya gönderici tarafından gönderilir.

Şifreleme Algoritması:

$m \in M$, m düz metin veya mesaj olmak üzere,

p_k açık anahtar

$c = E(m, p_k), c \in C$ işlemine şifreleme algoritması denir. Şifreleme algoritması ve açık anahtar kullanılarak m düz metinden, c şifreli metin elde edilir.

Şifre Çözme (Deşifreleme) Algoritması:

Verilen şifreli metin $c = E(m, p_k)$ ve s_k gizli anahtar için, $m = D(c, s_k)$ algoritmasına, yani şifreli metinden, düz metinin yani ilgili mesajın elde edilmesidir.

E şifreleme algoritması, M düz metin uzayından, C şifreli metin uzayına tek yönlü trapdoor fonksiyondur. Yani tüm şifreli metinler $c = E(m, p_k)$ için, makul sürede c ve p_k bilinerek, mesajın hesaplanması mümkün değildir. Fakat s_k, gizli anahtar kullanılarak, E şifreleme algoritmasının tersi olan D deşifreleme algoritması ile alıcı tarafından kolaylıkla hesaplanabilir.

Açık anahtarlı sistemlerde güvenlik tek yönlü (trapdoor) fonksiyonlara dayandırılmaktadır. Bu fonksiyonların kendisinin hesaplanması kolay iken tersinin hesaplanması polinomsal süre içinde imkansızdır [15]. Trapdoor fonksiyonları, bazı matematiksel problemlerin zorluğuna dayanmaktadır. Bunlar, tam sayıların asal çarpanlarına ayrılması ve ayrık logaritma problemleridir.

Kullanıcı için şifreleme ve deşifrelemede kullanılacak anahtarları üretmek oldukça kolaydır. Trapdoor fonksiyonların zorluğu ise, sadece açık anahtardan yola çıkılarak ek bir bilgi verilmeden, deşifrelemede kullanılan gizli anahtarı hesaplamak hemen hemen imkansızdır. Bu yüzden açık anahtarlı sistemlerde, açık anahtar herkese yayınlanırken gizli anahtar sadece ilgili kişi tarafından bilinmelidir. Ayrıca açık anahtarlı kripto sistemlerde, simetrik anahtarlı kripto sistemlerdeki gibi gizli anahtar paylaşımı yoktur.

Açık anahtarlı sistemlerin en ilk ve en çok kullanılanı ve sayıların asal çarpanlarına ayrılması problemine dayanan RSA açık anahtarlı kripto sistemi inceleyelim.

2.2. RSA Açık Anahtarlı Kripto Sistem

Diffie ve Hellman açık anahtar kripto sistemler fikrini 1976 yılında yayınladılar, fakat pratiksel bir uygulaması üzerine çalışmadılar. Açık anahtar sistemlerin ilk pratiksel uygulaması, RSA [2], Ron **R**ivest, Adi **S**hamir ve Leonard **A**dleman tarafından, 1977 yılında yayınlandı. RSA kripto sisteminin adı, bu üç bilim insanının isimlerinin baş harfinden oluşmaktadır. RSA kripto sistemi, biri açık diğeri gizli olarak tutulan açık anahtarlı kripto sistemin güvenliği, iki büyük asal sayının çarpımından oluşan sayının çarpanlarına ayrılmasının zorluğuna dayanmaktadır. Bu sistem hem gizlilik, hem de dijital imza sağlamak amacıyla kullanılabilir. Bu sistemin güvenliği tamsayılarda çarpanlara ayırma probleminin kolaylıkla olmamasına dayanır [9].

Ünlü İngiliz matematikçisi, Clifford Cocks 1973 yılında RSA ya benzer, açık anahtarlı kripto sistem tasarlamıştır fakat bunu 1997 yılına kadar yayınlamamıştır. Çalıştıkları projenin gizlilik içermesinden dolayı 1997 yılına kadar yayınlamamışlardır. Ancak 1997 yılında duyurusu yapılmıştır [10].

RSA açık anahtarlı kripto sistemi daha öncede bahsedildiği üzere üç algoritmadan oluşmaktadır. Bunlar anahtar üretimi, şifreleme ve deşifreleme (şifre çözme). Sırasıyla bunları ele alalım.

Anahtar Üretimi:

RSA açık anahtarlı kripto sistemi olduğu için biri açık diğeri gizli olmak üzere bir çift anahtar içerir. Herkes tarafından bilinen açık anahtar ile mesaj, düz metin, şifrelenir. Gizli anahtar ile makul bir sürede şifreli metinler deşifre edilerek, çözülerek ilgili düz metin elde edilir. RSA açık anahtarlı kripto sistemlerde anahtarlar aşağıdaki gibi üretilir:

1. Birbirinden farklı aynı uzunlukta rastgele oldukça büyük en az her biri 512 bit (yaklaşık 160) basamaklı p , q iki asal sayı seçilir. Bu asal sayıların asallığı da ayrıca test edilir. Bu anahtar uzunluğu 1024-bit olan, RSA-1024 için geçerlidir. Eğer anahtar uzunluğu 2048, RSA-2048 anahtar üretilmek istenirse p ve q asallarının uzunlukları 1024-bit (yaklaşık 320 basamaklı) iki asal sayı seçilir. NIST'in bu on yıl için, yani 2020 ye kadar belirlediği anahtar uzunluğu 2048 -bit, yani her bir asalın uzunluğu 1024-bit olmalıdır. Bu bölümün sonunda güvenlik açısından NIST in belirlediği anahtar uzunluğunun yıllara göre bir tablosunu ve Moore yasasından kısaca bahsedeceğiz.

2. $n = p \cdot q$

Sayısı açık ve gizli anahtarlar için modül olarak kullanılır, bit türünden ifade edilir. Yukarıda ifade ettiğimiz gibi, n modülü bize kripto sistemin anahtar uzunluğunu verir. RSA-1024, RSA-2048 ve RSA-3072 gibi.

3. $\varphi(n) = \varphi(p) \cdot \varphi(q) = (p-1) \cdot (q-1)$

φ: Euler phi fonksiyonu olmak üzere

$\varphi(n)$: n den küçük n ile aralarında asal olan pozitif tam sayıların sayısı

4. $1 < e < \varphi(n)$ ve $ebob(e\,, \varphi(n)) = 1$

Olacak biçimde, e tamsayısı seçilir. e tamsayısı ile $\varphi(n)$ nin en büyük ortak böleni 1 dir. Diğer bir ifadeyle, e ile $\varphi(n)$ aralarında asaldır. e sayısı açık anahtar üsteli olarak da bilinir. e sayısı genel olarak küçük seçilir ve bu sayı genel olarak

$2^{16} + 1 = 65537$ olarak alınır. Fakat e nin $(e = 3)$ küçük değerler alınması güvenlik açıklarına [11] neden olmaktadır.

5. $d = e^{-1}\ (mod\ \varphi(n))$

d sayısı e'nin modül $\varphi(n)$'de çarpımsal tersidir. Bu hesaplama genel olarak Genişletilmiş Euclid (Öklid) Algoritması ile yapılır.

Sonuç olarak; açık anahtar, modül n ve açık anahtar üstel e sayısından yani

(n, e) ikilisinden oluşurken, gizli anahtar ise sadece gizli d üstel sayısından oluşur. p, q ve $\varphi(n)$ sayıları gizli tutulmalıdır. d sayısını hesaplamak için kullanılırlar.

Açık anahtar: (n, e)

Gizli anahtar, kapalı anahtar: (d)

RSA da Şifreleme

A kişisi açık anahtarı (n, e) ikilisini yayınlarken gizli anahtarı (d)'yi gizler. B kişisi, A şahsına M mesajını göndermek istesin. M mesajını m tamsayısına dönüştürür, m sayısı

$0 < m < n$ eşitsizliğini sağlar.

B kişisi şifreli mesaj,

$c = m^e\ (mod\ n)$ ifadesini hesaplar.

Bu hesaplamada üst almada kare metodu kullanılarak hızlı bir şekilde hesaplanarak A şahsına, yani alıcıya gönderilir.

RSA da Deşifreleme (Şifre çözme)

A şahsı, B kişisinin göndermiş olduğu şifreli metini $c = m^e \ (mod\ n)$ kendi gizli anahtarı d yi kullanarak,

$m = c^d \ (mod\ n)$ ifadesini hesaplar. m tamsayısı tekrar ascii kodlar kullanılarak M mesajına dönüştürülür.

RSA tersten kullanılırsa, şifreleme gizli anahtar, deşifreleme açık anahtarla, bu durumda mesaja kişinin dijital imzası, *e-imzası* atılmış olur. Tabi burada hash (özet) fonksiyonları da kullanılır. Çünkü mesajın tamamını imzalamak çok zahmetlidir. Bu yüzden mesajın özeti alınır. Bu genelde 160 bit uzunluğunda olur ve bu özet imzalanır. Ayrıca imza sadece kişinin gizli anahtarıyla atıldığı için, böylece kişinin açık anahtarıyla herkes kontrol edebilir [9].

RSA da dijital imzayı ilerleyen bölümlerde ele alacağız.

RSA kripto sisteminin şifreleme, deşifreleme algoritmasının matematiksel olarak açıklaması:

$$m^{\varphi(n)} \equiv 1 \ (mod\ n) \qquad (2.1)$$

$$e \cdot d \equiv 1 \left(mod\ \varphi(n)\right) \implies e \cdot d = k \cdot \varphi(n) + 1$$

$$c^d \equiv (m^e)^d \equiv m^{e \cdot d} \equiv m^{k \cdot \varphi(n)+1} \equiv m^{k \cdot \varphi(n)} \cdot m^1 \equiv \left(m^{\varphi(n)}\right)^k \cdot m^1 \equiv m$$

Şimdi RSA açık anahtar kripto sisteminin şifreleme ve deşifreleme algoritmasını bir örnekle açıklayalım.

Örnek:

1. Birbirinden farklı $p = 53$ ve $q = 61$ asal sayıları seçilir.

2. $n = p \times q = 53 \times 61 = 3233$ modülü hesaplanır.

$n = 3233$ bulunur.

3. Euler phi fonksiyonu $\varphi(n) = \varphi(p) \cdot \varphi(q) = (p-1) \cdot (q-1)$ kullanılarak, $n = 3233$

$\varphi(3233) = (53-1) \times (61-1) = 3120$

değeri hesaplanır.

4. $1 < e < 3120$ ve $ebob(e,\ 3120) = 1$ olacak biçimde, yani 3120 ile aralarında asal olacak biçimde e tamsayısı, yani e sayısı 3120'yi bölmeyen bir tamsayı seçilir.

$e = 17$ seçelim.

5. d üstel tamsayısı, yani kapalı anahtar,

e nin $(mod\ 3120)$ de çarpımsal tersi olacak biçimde seçilir. Bu hesaplama yapılırken Genişletilmiş Öklit algoritması kullanılır.

$d = 2753$ bulunur. Çünkü, $e \cdot d \equiv 1\ (mod\ 3120)$ dir.

Anahtar üretimi, açık veya kapalı anahtarlar üretimi; 1'den 5'e kadar tüm aşamalar tamamlanmış olur.

Açık anahtar, $p_k = (n = 3233, e = 17)$ ikilisinden oluşur.

Kapalı anahtar, $s_k = (d = 2753)$ dir.

Şimdi $M =$ "A" mesajını şifreleyelim.

$M =$ "A" mesajı, m=65 tamsayısına ascii kodla çevrilir.

Alıcının açık anahtarı $p_k = (n = 3233, e = 17)$ ile şifrelenerek,

$c \equiv m^e \equiv 65^{17}\ (mod\ 3233) \Longrightarrow c \equiv 2790\ (mod\ 3233)$ bulunur.

Şifreli c =2790 sayısı alıcıya iletilir. Alıcı kendi gizli, yani kapalı anahtarı d=2753 ile

Rahatlıkla $m \equiv 2790^{2753}\ (mod\ 3233) \equiv 65$ bulunur. Ascii kod karşılığı alınarak;

$M =$ "A" mesajının kolaylıkla deşifreleme (mesaj çözme) algoritmasını kullanarak hesaplar.

Şimdi de RSA'nın güncel hayatta önemli bir uygulamasından bahsedelim.

2.3. RSA ile Dijital İmza

RSA İmza Şeması:

RSA kripto sistemi dijital imza (*e- imza*, elektronik imza) içinde kullanılır.

$(n = modül\,, e = üstel\ sayı)$ ikilisi A şahsının mesaj şifrelenmesi için herkese açık anahtarı, d üstel tamsayısı da A kişisinin deşifreleme de kullandığı kapalı (gizli) anahtarı olsun. İlk önce M mesajının imzalanabilmesi için,

$0 < m < n$ aralığındaki bir m tamsayısına dönüştürülmesi gerekir.

RSA ile İmzalama:

A şahsı B kişisine m mesajını göndermek isterse, kendisinin gizli (kapalı) anahtarını uygular. Yani, $\sigma \equiv m^d \ (mod\ n)$ hesaplanır.

Daha sonra $(m\ ,\ \sigma\ )$ imzalı mesajı B ye gönderir. Böylelikle *e-imza*, elektronik imza, tamamlanmış olur. Dikkat edilirse imzalama için ilgili kişi sadece kendisinin bildiği gizli (kapalı) anahtarını kullanır. Uzun mesajlar için, açık anahtarlı kripto sistemlerin şifreleme algoritması yavaş olduğu için mesajın özeti alınıp imzalanır.

İmza Doğrulama:

B kişisi A dan aldığı $(m\ ,\ \sigma\ )$ imzalı mesajı doğrulamak için, A'nın açık anahtarını kullanır.

$m \equiv \sigma^e \ (mod\ n)$ değerini hesaplar.

Eğer çıkan sonuç m ise imza doğrulanmış olur.

RSA da *e-imza* kullanımı ve doğrulamayı bir örnekle açıklayalım.

Örnek:

İmzalamaya, anahtar oluşturmayla başlayalım. A kişisi, $p = 6997$ ve $q = 7927$ asal sayılarını seçer.

$n = p \cdot q = 55465219$ modülünü,

ve $\varphi(n) = \varphi(554665219) = (69997 - 1).(7927 - 1) = 55450296$ değerlerini hesaplar.

Daha sonra A kişisi, kendisinin gizli(kapalı) anahtarını,

$e \cdot d = 5 \cdot d\ (mod\ 55465219)$ eşitliğinden d =44360237 sayısını bulur.

A'nın açık anahtarı $(n = 55465219, e = 5)$;

A'nın gizli anahtarı (d =44360237) olur.

A kişisi "·TCN" mesajının imzalamak istesin.

M= "·TCN" mesajını m=31229978 sayısına dönüştürür.

A kişisi;

$\sigma = m^d \ (mod\ n) = 31229978^{44360237}\ (mod\ 55465219) = 30729435$ bulur, ve

$(m = 31299978, \sigma = 30729435)$ ikilisini B'ye gönderir.

İmzayı doğrulamak için;

$(m = 31299978, \sigma = 30729435)$ yi alan B kişisi, A nın açık anahtarını kullanır,

$M = \sigma^e \ (mod\ n) = 30729435^5 = 31229978\ (mod\ 55465219\ )$

Çıkan sayı m sayısı ile eşit olduğu için imza doğrulanmış olur.

2.4. RSA ya Yapılan Ataklar

RSA ya yapılan birçok atak (saldırı) vardır. Bunlar;

1. Mesajı şifrelerken kullanılan şifreleme üstel tamsayısının, e nin küçük sayılar seçilmesidir. Örneğin $e = 3$ ve $m < n^{1/e}$ olacak biçimde seçilirse $c = m^e$ değeri modül değerinden küçük olur. Bu durumda c şifrelenmiş metin kolaylıkla e ninci dereceden kökü alınarak m mesajı kolaylıkla bulunabilir.

2. Eğer aynı mesaj, aynı e üstel sayısı kullanılarak, farklı p ve q dolayısıyla farklı modül n kullanılarak birçok alıcıya gönderilmesi Çinlilerin kalan teoremi kullanılarak şifreli metinler çözülebilir. Bu atak, saldırı John Hastad tarafından bulunmuştur ve sonrasında Don Smith tarafından da daha da geliştirilmiştir [13].

3. RSA kripto sistem deterministik şifreleme sistemidir. Diğer bir ifadeyle rastgele bir bileşen (anahtar üretimi hariç) içermediği için olasılı (probabilistik) şifreleme sistemi değildir. Daha sonra inceleyeceğimiz ElGamal kripto sistemi ise olasılı şifreleme sistemidir ki aynı düz mesaj hiçbir zaman aynı şifreli metini aynı anahtarlar kullanılarak vermez. RSA ise aynı anahtarlar kullanıldığında her defasında aynı mesaj için aynı şifreli metini verir. RSA kripto sistemine, benzer düz metinler kullanılarak aynı şifreli metin elde edilip edilmediği test edilerek atak (saldırı) yapılabilir.

RSA kripto sistemin güvenliği, çok büyük tamsayıların çarpanlara ayrılma problemine dayanmaktadır. İki büyük asalı çarparak n büyük sayısını hesaplamak kolaydır. Fakat bu n büyük sayının hangi iki asalın çarpımına eşit olduğunu bulmak çok zordur. Günümüzde bilinen en hızlı bilgisayar bir saniyede 367 trilyon işlem gücüne sahiptir.

Açık anahtar kriptografisinin en çok bilinen kripto sistemlerinden olan RSA nın süresel karmaşıklığı çok yüksektir. Dünyanın en hızlı bilgisayarı ile RSA-1024 de kullanılan 1024-bit uzunluğunda n sayısının çarpanlarına ayrılması onlarca yıl alacaktır [12].

2.5. ElGamal Açık Anahtarlı Kripto Sistem

Şimdi de ayrık logaritma probleminin zorluğuna dayanan 1985 yılında Taher ElGamal tarafından önerilen, ElGamal açık anahtar kripto sistemini inceleyelim. ElGamal şifreleme şeması [11] açık anahtarlı kripto sistem olup, Diffie-Hellman anahtar değişimi prensibine dayanır. ElGamal şifreleme şeması devirli G grubu üzerinde tanımlanır. ElGamal'ın güvenliği ise G grubunda tanımlanan ve G grubuyla ilişkili ayrık logaritma problemine dayanır. ElGamal açık anahtarlı şifre sistemi, anahtar transferi açısından Diffie-Hellman anahtar anlaşması olarak görülebilir. Güvenirliği ayrık logaritma problemi ve Diffie-Hellman probleminin kolay çözülememesinin temeline dayanmaktadır [9]. ElGamal şifreleme modeli de yine RSA'da olduğu gibi üç ayrı aşamadan oluşur. Bunlar anahtar üretimi, şifreleme algoritması ve deşifreleme algoritmasıdır. Şimdi bunları sırasıyla ele alalım.

1. Anahtar Üretimi

G mertebesi (derecesi) q olan bir sonlu grup olmak üzere ve g sayısı da G nin üreteci (ilkel kökü) olmak üzere,

A şahsı random(rastgele) $x \in \{1,2,3,4 \dots q-1\}$ sayısını seçer. Bu sayı

$x \in \{1,2,3,4 \dots [\![\log q]\!]\}$ olarak da seçilebilir [14].

A şahsı $y = g^x$ ifadesini hesaplar. A şahsı;

(G, q, g, y) yi açık anahtarı olarak yayınlar, x tamsayısı ise A nın gizli (kapalı) anahtarıdır.

2. Şifreleme

M mesajını şifrelemek için A nın yayınlanan açık anahtarı (G, q, g, y) nin yanında random, rastgele $r \in \{1,2,3,4 \dots q-1\}$ sayısını seçer. Yine $r \in \{1,2,3,4 \dots [\![\log q]\!]\}$seçilebilir.[14]. B şahsı, $c_1 = g^r$ yı hesaplar. Aynı zamanda $s = y^r$ yi hesaplar ve M mesajını $m \in$ G olarak dönüştürür. $c_2 = m \cdot s$ ifadesini hesaplar ve B şahsı,

A ya şifreli metin $(c_1, c_2) = (g^r,\ m \cdot y^r)$ gönderir. Eğer m bilinirse y^r ifadesi kolaylıkla hesaplanabilir. Bu yüzden, her bir şifreleme için random r üretilir. r ye geçici anahtar da denir. Bu yüzden ElGamal açık anahtar kripto sistemi probabilistik yani olasılı şifrelemedir. Her defasında şifrelenen aynı mesajdan farklı bir şifreli metin elde edilir. RSA'da ise bu durum öyle değildir. RSA deterministik şifreleme sistemidir; şifrelenen metin her defasında aynı anahtarla aynı şifreli metini verir.

Güvenli şifreleme için RSA'da olduğu gibi anahtar uzunluğu en az 1024-bit [14] olarak seçilmeli yani grubun mertebesi (derecesi) q (310 basamaklı) olarak seçilmelidir. ElGamal-2048, ElGamal-3072 için G grubunun mertebesi yaklaşık olarak sırasıyla 620 ve 930 basamaklı bir tamsayı seçilmelidir. r geçici anahtar uzunluğu ve x gizli (kapalı) anahtar, 1024-bit uzunluğundaki q için, $q-1$, yani 1023 bit uzunluğunda ≈310 basamaklı veya $[log_2{}^q]$ ≈310 basamaklı bir sayı seçilmelidir. ElGamal-2048 ve ElGamal-3072 için sırasıyla, ElGamal şifreleme sisteminde kullanılan tüm parametrelerin büyüklükleri sırasıyla, yaklaşık olarak her biri yaklaşık olarak 620 ve 930 basamaklı seçilmelidir.

3. Deşifreleme

$(c_1, c_2) = (g^r,\ m \cdot y^r)$ şifreli mesajı çözmek için A kişisi, sadece kendisinin bildiği x gizli (kapalı) anahtarını kullanır. A şahsı ilk önce $t = {c_1}^x$ ifadesini hesaplar. Sonrada

$m = c_2 \cdot t^{-1}$ ifadesini hesaplar. t^{-1} ifadesi, G grubunda t nin çarpımsal tersidir.

Bu durumda;

$c_2 \cdot t^{-1} \equiv (m \cdot s) \cdot {c_1}^{-x} \equiv m \cdot y^r \cdot g^{-xr} \equiv m \cdot g^{xr} \cdot g^{-xr} \equiv m$ olur. m sayısı ascii kod kullanılarak M mesajına dönüştürülebilir.

ElGamal açık anahtar kripto sisteminin en önemli dezavantajlarından biri şifreleme algoritması ile düz metin şifrelenerek iki katına çıkar. Çünkü düz metin sadece şifreli metin c ye dönüştürülmez, c_1 ve c_2 değerlerine dönüştürülmesinden kaynaklanır. RSA da böyle bir durum yoktur, mesajın uzunluğuna eşdeğer şifreli mesaj elde edilir. ElGamal kripto sistemi için, anahtar oluşturma, şifreleme ve deşifreleme algoritmalarını örneklendirelim.

Örnek:

A kişisi ilk önce modül olarak kullanacağı p asalını ve g , $\mathbb{Z}_p{}^*$ grubunun üreteci olacak biçimde $g \in \{2,3,4 \ldots p-2\}$ sayılarını seçer.

p = 2889 ve g = 2585 seçilsin.

İkinci olarak, A gizli (kapalı) anahtarı $x \in \{2,3,4 \ldots p-1\}$ olarak seçer.

x = 47 olarak seçilsin. x, A nın gizli anahtarıdır.

A, sonrasında $y = g^x$ değerini hesaplar. y değeri;

$y \equiv g^x \equiv 2585^{47} \equiv 2826 \ (mod\ 2879)$ olarak bulur.

A, açık anahtarını (p, g, y) = (2879, 2585, 2826) olarak yayınlar.

B şahsı M= "M" mesajını, m=77 sayısına ascii kodu kullanarak dönüştürdükten sonra, r geçici sayısını random olarak seçer. r =65 olarak seçilsin. Bu durumda, B kişisi, şifreli metin (c_1, c_2) ifadesini hesaplayarak A kişisine gönderir. $c_1 \equiv g^r \equiv 2585^{77} \equiv 319 \ (mod\ 2879)$

Ve $c_2 \equiv m \cdot y^r \equiv 77 \cdot 2826^{65} \equiv 472 \ (mod\ 2879)$ değerlerini bulur.

A şahsı almış olduğu şifreli metini kullanarak;

$^{c_1}/_{c_2{}^x}$ ifadesini $mod\ 2879$ da hesaplar. Burada x sayısı A nın gizli (kapalı) anahtarıdır.

$m = {}^{c_1}/_{c_2{}^x} \equiv {}^{472}/_{319^{47}} \equiv 77 \ (mod\ 2879)$, x =47 gizli anahtarını kullanarak kolayca hesaplar.

Sonra ascii kod kullanılarak m =77 mesajını, M = "M" ifadesine dönüştürür. Bilindiği üzere, ElGamal açık anahtar şifreleme sistemi Diffie-Hellman anahtar değişimi prensibine dayanmaktadır. Kriptografinin en temel uygulamalarından biri olan anahtar değişiminden biraz bahsedelim.

2.6. ElGamal ile Dijital İmza

ElGamal kripto sisteminde imza, RSA da olduğu gibi mesajın doğru kişi tarafından gönderildiğini kontrol etmek için kullanılır. Sadece şifreli metin yerine, imzalanmış metin gönderilerek, şifreli metnin istenen kişi tarafından gönderilip gönderilmediğini

kontrol etmiş olur. A kişisinin açık anahtarı $p_k = (q,\ g, y = g^x)$ ve gizli(kapalı) anahtarı $s_k = x$ olsun.

İmza Algoritması

m mesajı, $m \in \mathbb{Z}_q{}^*$ olsun. Eğer değilse, daha önceden de belirttiğimiz gibi özet(hash), örneğin SHA-3, fonksiyonu kullanılarak, $m \in \mathbb{Z}_q{}^*$ olması sağlanır. A şahsı, m mesajının aşağıdaki şekilde imzalar:

1. Rasgele $t \in \{1,2,3 \dots q-2\}$ ve $ebob(t\ , q-1) = 1$ olacak biçimde seçilir. Yani t sayısı ile q aralarında asal, ortak bölenleri 1.

2. $r \equiv\ g^t\ (mod\ q)$ ve $s \equiv\ t^{-1}(m - r \cdot x)\ (mod\ q)$ eşitliklerini sağlar.

3. (m, r, s) A'nın imzalı mesajıdır.

İmza Doğrulama:

(m, r, s) imzalı mesajı alan B şahsı aldığı mesajın A ya ait olup olmadığını, şu şekilde doğrular:

1. Öncelikle $1 < r < q - 1$ olup olmadığını kontrol ederi yoksa reddeder.

2. Sonra $\vartheta = g^m$ ve $\omega = y^r \cdot r^s$ değerlerini $(mod\ q)$ da hesaplar. (Buradaki y değeri A'nın açık anahtarındaki y değeri olup, $y = g^x$ dir)

3. Eğer ϑ=ω ise imza kabul edilir yoksa reddedilir.

Örnek:

A'nın açık anahtarı, $(q,\ g, y = g^x)$= (2357, 2, 1185) ve gizli anahtarı x=1751 olmak üzere, burada g =2, $\mathbb{Z}_{2357}{}^*$ nin bir ilkel köküdür, yani grubun jeneratörüdür.

İmza oluşturma:

m=1463 olarak seçelim eğer $m > q$ olsaydı, hash(özet) fonksiyonunu kullanırdık.

A şahsı, m =1463 mesajını imzalamak için rastgele t=1529 sayısını seçer.

$r \equiv g^t\ (mod\ q) \equiv 2^{1529}\ (mod\ 2357)$ hesaplar. $r\ \equiv 1490\ \ (mod\ 2357)$ bulunur.

Ve sonrasında,

$t^{-1} \equiv (mod\ q-1) \equiv 1529^{-1} \equiv 245\ (mod\ 2356)$ bulunur.

$s \equiv t^{-1} \cdot (m - r \cdot x)\ (mod\ q-1)$

$s \equiv 245 \cdot (1463 - 1490 \cdot 1751)\ \ (mod\ 2356)$, burada x A nın gizli anahtarıdır. Dikkat edilirse imzalama A'nın gizli anahtarı kullanılarak yapılmaktadır.

$s \equiv 1777$ bulunur.

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

İmzayı Doğrulama:

B kişisi aldığı imzalı mesajın A dan geldiğini doğrulamak için ilk önce,

$\vartheta = g^m\ (mod\ q)$ ifadesini hesaplar.

$2^{1463} \equiv 1072\ (mod\ 2357)$ değerini hesaplar. Sonrasında da,

$\omega = y^r \cdot r^s\ \ (mod\ q)\ \Longrightarrow 1185^{1490} \cdot 1490^{1777}\ \ (mod\ 2357) \equiv 1072\ (mod\ 2357)$ değerini hesaplar.

$\vartheta = \omega$ olduğu için imzayı kabul eder, ilgili mesajın A kişisi tarafından kendisine gönderildiğini kontrol etmiş olur.

2.7. Diffie-Hellman Anahtar Anlaşması

Diffie-Hellman anahtar anlaşması, simetrik kripto sistemlerde, örneğin AES, anahtar paylaşımı için önerilmiş ilk pratik çözümdür. Üs alınarak anahtar iletimi olarak anılan bu sistem, hiç haberleşme sağlamamış taraflar için açık kanal üzerinden mesajlarını birbirlerine göndererek ortak anahtar oluşturabilirler.

p oldukça büyük bir asal sayı, g sayısı da $\mathbb{Z}_p^*$ de ilkel (primitif) kök olsun, p ve g herkese açık anahtar.

Bu durumda A ve B kişileri simetrik kripto sistemleri (AES) için kullanmak istedikleri, ortak s_k gizli anahtarlarını aşağıdaki biçimde oluşturabilirler.

1. A kişisi, $1 < a < p-2$ koşulunu sağlayan rastgele a gizli (kapalı) anahtarını, sayısını seçer ve $c = g^a$ yı hesaplar ve bu ifadeyi haberleşmek istediği B kişisine gönderir.

2. B kişisi de $1 < b < p - 2$şartını sağlayan b gizli(kapalı) anahtarını, sayısını seçer ve

$d = g^b$ yi hesaplayıp bunu A kişisine gönderir.

3. A kişisi simetrik kripto sistemi için ortak kullanacağı $s_k = k$ gizli anahtarını

$k \equiv d^a \equiv (g^b)^a \equiv g^{a \cdot b}$ ifadesini hesaplar.

4. B kişisi simetrik kripto sistemi için ortak kullanacağı $s_k = k$ gizli anahtarını

$k \equiv c^b \equiv (g^a)^b \equiv g^{a \cdot b}$ ifadesini hesaplar.

Böylelikle A ve B kişileri, simetrik kripto sistemlerinde kullanmak istedikleri, gizli (kapalı) ortak anahtar için anlaşmış olur.

g^a ve g^b ifadelerinden gizli olarak seçtikleri a ve b nin bulunamaması da ayrık logaritma probleminin zorluğuna dayanır. ElGamal kripto sistemi de Diffie-Hellman anahtar değişimi prensibine dayanır. Diffie-Hellman anahtar değişimini bir örnekle açıklayalım.

Örnek:

Herkese açık p ve g sayılarını p=13 ve g=2 olarak seçelim.

A kişisi gizli anahtarı olan a sayısını, g=5 seçsin.

B şahsı da gizli anahtarını, b sayısını, b=4 olarak seçsin.

A kişisi $c \equiv g^a \equiv 2^5 \ (mod\ 13)$

B kişisi $d \equiv g^b \equiv 4 \ (mod\ 13)$

Hesapladıkları c ve d sayılarını birbirlerine gönderirler.

A kişisi oturum için kullanacakları $s_k = k$ gizli anahtarını,

$k \equiv d^a \equiv 3^5 \ (mod\ 13)$,

B ise k gizli anahtarı, $k \equiv c^b \equiv 6^4 \ (mod\ 13)$ olarak hesaplar.

Böylelikle k =9 oturum gizli anahtarıyla A ve B kişileri birlerinden haberdar olmadan, k gizli anahtarını kullanarak şifreli olarak mesajlaşabilirler.

Not: 2 sayısının $\mathbb{Z}_{13}{}^*$de primitif kök olduğuna dikkat edelim.

$2^1 \equiv 2\ (mod 13)$ $2^4 \equiv 3\ (mod 13)$ $2^7 \equiv 11\ (mod 13)$ $2^{10} \equiv 10\ (mod 13)$

$2^2 \equiv 2\ (mod 13)$ $2^5 \equiv 6\ (mod 13)$ $2^8 \equiv 9\ (mod 13)$ $2^{11} \equiv 7\ (mod 13)$

$2^3 \equiv 2\ (mod 13)$ $2^6 \equiv 12\ (mod 13)$ $2^9 \equiv 5\ (mod 13)$ $2^{12} \equiv 1\ (mod 13)$

2.8. Moore Yasası ve Kripto Sistemlerin Anahtar Boyutları

Intel Şirketinin kurucularından Gordon Moore'un adıyla anılan bu yasa, 1965 yılında Electronics Magazine dergisinde yayımlanmıştır. Kendisi tarafından yasa olarak görülmeyen bu yasa, temel olarak, mikroişlemciler içindeki transistör sayısı her yıl iki katına çıkacaktır biçiminde tanımlanabilir [23]. Bu yasaya göre, bilgisayarların işlemci gücü her sene ikiye katlanmaktadır. Bilgisayar dünyasındaki, bilgisayarların işlemci gücünün her yıl iki katına çıkması ve maliyetlerinin değişmeden kalması hatta düşmesinden, kriptografi de nasibini almıştır. Nasibini almış derken, Kriptografinin iyi yönde gelişmesinden bahsediyoruz. Bu durumda kriptografide simetrik veya asimetrik kripto sistemler de kullanılan anahtar boyutları da, bilgisayarların işlemci gücünün artmasından dolayı artırılması gerekmektedir. Örneğin 2000'li yıllarda RSA için 512-bit uzunluğundaki anahtar boyutu verinin güvenli bir şekilde iletimi için yeterli iken şimdilerde, güncel bilgisayarlar tarafından oluşturulan dağıtık bir sistemle kırılabilmektedir. Aynı şekilde hash (özet) algoritmalarından biri olan MD-5 (Message Digest -5) artık tarihe karışmıştır. Bundan dolayı, açık anahtar kriptografisinde kullanılan anahtar boyutları da yıllara göre değişmektedir. NIST in bu alanda yapmış olduğu çalışmaya göre, verilerin güvenliği açısından kripto sistemlerde, yıllara göre kullanılması gereken anahtar boyutları da bit bazında aşağıdaki tabloda verilmiştir.

Çizelge 2.1:Yıllara göre, kripto sistemlerde kullanılması gereken anahtar uzunlukları [24].

Güvenlik Seviyeleri / Algoritmalar	Sınıflandırılmamış	Gizli (Kısa Süreli Koruma)	Gizli (Orta Süreli Koruma)	Gizli (Uzun Süreli Koruma)	Gizli (Askeri Süreli Koruma)
Simetrik					
2DES	80	-	-	-	-
3DES	-	112	-	-	-
AES	-	-	128	192	256
Asimetrik					
DSA[1]	1024-160	2048-224	3072-256	7680-384	15360-512
RSA	**1024**	**2048**	**3072**	**7680**	**15360**
ECDSA	160	224	256	384	512
EC	160	224	256	384	512
ELGAMAL	**1024**	**2048**	**3072**	**7680**	**15360**
Özet Fonksiyonları					
RIPEMD	160	-	-	-	-
SHA	-	224	256	384	512
Son Geçerlilik Tarihi*	2010	2020	2030	2050	**

[1] İlk değer grubun boyutunu, ikinci değer anahtar boyutunu göstermektedir.

* Günümüz koşullarına uygun olarak yaklaşık değerler verilmiştir.

** Kuantum bilgisayarlara karşı iyi koruma sağlamaktadır.

3. HOMOMORFİK ŞİFRELEME

Homomorfik (Eş şekilli) şifreleme, şifreli metinler üzerinde bir takım matematiksel hesaplamalar yapılabilmesine (çarpma, toplama vs.) olanak sağlayan şifreleme sistemleridir. Diğer bir deyişle, örneğin şifreli metinler üzerinde çarpma işlemi uygulanıp (uygulanabilirse) elde edilen çarpımsal şifreli metin, deşifre edildiğinde düz metinler çarpımına eşittir. RSA homomorfik özelliği sağlayan ilk açık anahtarlı kripto sistemdir. RSA açık anahtarlı kripto sistem, sadece çarpımsal homomorfik şifreleme özelliğini sağlar.

Sadece bir tane matematiksel işlem(operasyon) yapılmasına olanak sağlayan şifreleme sistemlere kısmi homomorfik şifreleme sistemler denir. İki tane matematiksel işlemi(operasyonu) sağlayan kripto sistemlere tam homomorfik şifreleme sistemler denir. RSA ve ElGamal çarpımsal homomorfik özelliğini sağlayan, biri sayının asal çarpanlarına ayrılma zorluğuna diğeri ise ayrık logaritma problemine dayanan açık anahtarlı kısmi homomorfik sistemlerdir. Biz bu iki sistemin homomorfik özelliklerini ele aldıktan sonra, bir sonraki bölümde Online Vergi Ödeme Sistemine (OVÖS) uygulamasını geliştireceğiz. Online Vergi Ödeme Sisteminde (OVÖS) performans açısından hangisinin daha kullanışlı olduğunu değişik boyutlardaki anahtar uzunluğunda, 1024-bit, 2048-bit ve 3072-bit uzunluğunda anahtarlarda sadece Java nın BigInteger sınıfını kullanarak, anahtar üretimi, şifreleme ve deşifreleme algoritmalarını yazacağız. Sonrasında, bu kripto sistemlerin çarpımsal homomorfik özelliklerini gerçekleyeceğiz. Aynı gerçeklemeyi, meşhur 3.Parti, kriptografi uygulamalarında anahtar üretimi, şifreleme, deşifreleme için kullanılan Bouncy Castle jar dosyasının son bir sürümünü kullanarak gerçekleyeceğiz. Bouncy Castle, kriptografide kullanılan bir API koleksiyonudur. Hem Java hem de C# programlama dilleri için API'ler içerir. Uygulamalara geçmeden önce homomorfik şifrelemenin tanımı, türlerinden genel bahsettikten sonra, sırasıyla RSA, ElGamal kripto sistemlerin çarpımsal homomorfik özelliklerini bir örnekle göstereceğiz. Sonra bu çarpımsal homomorfik özelliği Java'da sadece BigInteger sınıfını kullanarak gerçekleyeceğiz. Bunu Bouncy Castle jar dosyasının son bir sürümünü kullanarak da gerçekleyeceğiz.

Sonra kısmi homomorfik şifrelemeye, toplamsal homomorfik özelliğini sağlayan Paillier açık anahtarlı kripto sistemin bu homomorfik özelliğini gösterip bir örnek verdikten sonra, Java'da BigInteger sınıfını kullanarak bunu yazılımsal olarak gerçekleyeceğiz. Tam homomorfik şifrelemeyi, tamsayılar üzerinde anlatıp, bir örnekle açıkladıktan sonra yine yazılımsal olarak Java'da BigInteger sınıfını kullanarak bunu gerçekleyeceğiz.

Soyut cebirden aşina olduğumuz üzere, iki cebirsel yapı arasında, yapısal özellikleri koruyan fonksiyonlara homomorfizma denir. Eğer bu iki cebirsel yapı grup ise grup homomorfizması denir. Kısmi homomorfik şifrelemeler aslında birer grup homomorfizmasıdır. Daha önceden matematiksel altyapı bölümünde cebirdeki grup kavramına değinildiği için burada bahsedilmeyecektir. Şimdi grup homomorfizmasının tanımını verelim.

3.1. Grup Homomorfizması ve Kısmi Homomorfik Tanımı

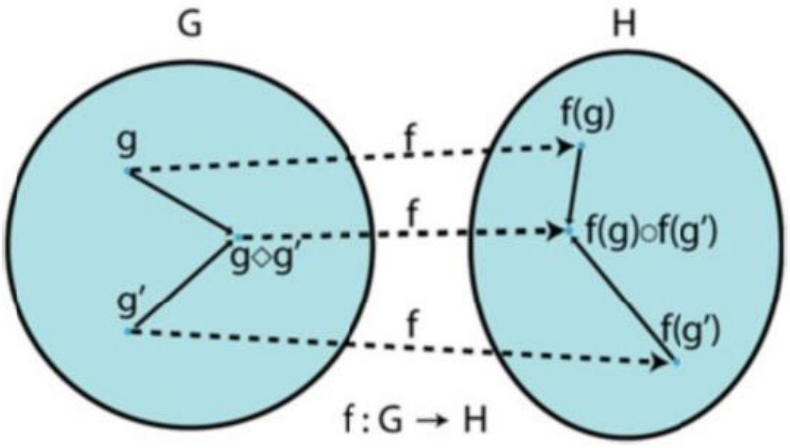
$(G,\diamond)$ ve $(H,\circ)$ grupları verilsin.

$f : G \to H$ fonksiyonu, $\forall\, g, g' \in G$ için,

$$f(g \diamond g') = f(g) \circ f(g') \tag{3.1}$$

eşitliği sağlanıyorsa, f fonksiyonuna grup homomorfizması denir.

Grup homomorfizması aşağıdaki şekildeki gibidir.



Şekil 3.1: Grup Homomorfizması [25].

Şimdi de kısmi homomorfik şifrelemenin veya genel olarak homomorfik şifrelemenin ne anlama geldiğini açıklayalım. Grup homomorfizmasının, kriptografideki karşılığını açıklayalım.

(P, C, K, E, D) şifreleme şemasında (beşlisinde);

P, C sırasıyla düz metin şifreli metini, K anahtar uzayını, E ve D sırasıyla şifreleme ve deşifreleme (veri çözme) algoritmalarını göstermektedir.

$(P,\diamond)$ ve $(C,\circ)$ düz metinler ve şifreli metinler ilgili işlemler üzerinde bir grup oluşturmaktadırlar. Şifreleme algoritması E, P grubundan C grubuna bir fonksiyon oluşturur.

$E_k: P \rightarrow C$ biçiminde gösterilir. $k \in K$ nın bir elemanı olup, eğer simetrik şifreleme ise gizli(kapalı) anahtar, eğer asimetrik şifreleme ise açık anahtar kümesinin bir elemanıdır.

Bütün $a, b \in P$ ve $k \in K$ için,

$$E_k(a) \circ E_k(b) = E_k(a \circ b) \qquad (3.2)$$

oluyorsa, bu şifreleme (P, C, K, E, D) sistemine homomorfik şifreleme veya kısmi homomorfik şifreleme denir. Eğer homomorfik şifreleme şeması, sadece bir matematiksel işlemi (çarpma, toplama vs.) den birini sağlıyorsa bu homomorfik şifreleme kısmi homomorfik şifreleme olarak adlandırılır.

Aslında kısmi homomorfik şifreleme sistemlerinin her birinde E şifreleme algoritması bir grup homomorfizmasıdır.

Şimdi ilk önce kısmi homomorfik şifreleme sistemlerinden, çarpımsal homomorfik özelliğini sağlayan RSA ve ElGamal kripto sistemlerinin, homomorfik özelliklerini matematiksel olarak gerçekleyip birer örnek verelim.

3.2. RSA Kripto Sisteminin Kısmi Çarpımsal Homomorfik Özelliği ve Java İmplementasyonu

RSA açık anahtarlı kripto sisteminde, açık anahtar $p_k=(n, e)$

$(P, \cdot)$ grubunda düz metinler grubu oluşturur.

Aynı şekilde, $(C, \cdot)$ şifreli metinler grubunu oluşturur.

$\forall m_1, m_2 \in P$ için

$E(m_1, p_k) \cdot E(m_2, p_k) = E(m_1 \cdot m_2, p_k)$ (3.1) sağlanır.

Daha açık bir ifadeyle,

$E(m_1, p_k) \equiv {m_1}^e \ (mod\ n)$

$E(m_2, p_k) \equiv {m_2}^e \ (mod\ n)$

Eşitlikleri, (3.1) nolu denklemde yerine yazılırsa;

$E(m_1, p_k) \cdot E(m_2, p_k) \equiv {m_1}^e \cdot {m_2}^e (mod\ n)$

$\equiv (m_1 \cdot m_2)^e \ (mod\ n)$

$\equiv E(m_1 \cdot m_2, p_k)$

olduğu gösterilmiş olur.

Sonuç olarak, RSA kripto sistemi çarpımsal homomorfik özelliğe sahiptir.

Örnek:

$p_k = (e, n) = (5437, 189781)$ açık anahtar.

$s_k = (d) = (49269)$ gizli (kapalı) anahtar.

Mesajlar $m_1 = 56947$, $m_2 = 64413$ olsun.

Açık anahtarlar kullanılarak mesajlar şifrelenirse;

$c_1 \equiv {m_1}^e \equiv (56947)^{5437} \equiv 96068\ (mod\ 189781)$

$c_2 \equiv {m_2}^e \equiv (64413)^{5437} \equiv 149380\ (mod\ 189781)$

Şifreli metinler c_1 ve c_1 nin çarpımı,

$c_1 \cdot c_2 \equiv 96068 \cdot 149380 \equiv 157744\ (mod\ 189781)$ (3.2)

Düz metinler m_1 ve m_2 nin çarpım sonucunun şifrelenmesi;

$m_1 \cdot m_2 \equiv 56947 \cdot 64413 \equiv 39943\ (mod\ 189781)$

$E(m_1 \cdot m_2) \equiv (39943)^{5437} \equiv 157744\ (mod\ 189781)$ (3.3)

Bu sonuç, (3.3) numaralı sonuç, yukarıda bulduğumuz şifreli metinler çarpımına, (3.2) nolu sonuca eşittir.

Sonuç olarak;

$E(m_1 \cdot m_2)=E(m_1)\cdot E(m_2) = c_1 \cdot c_2$ bulunur.

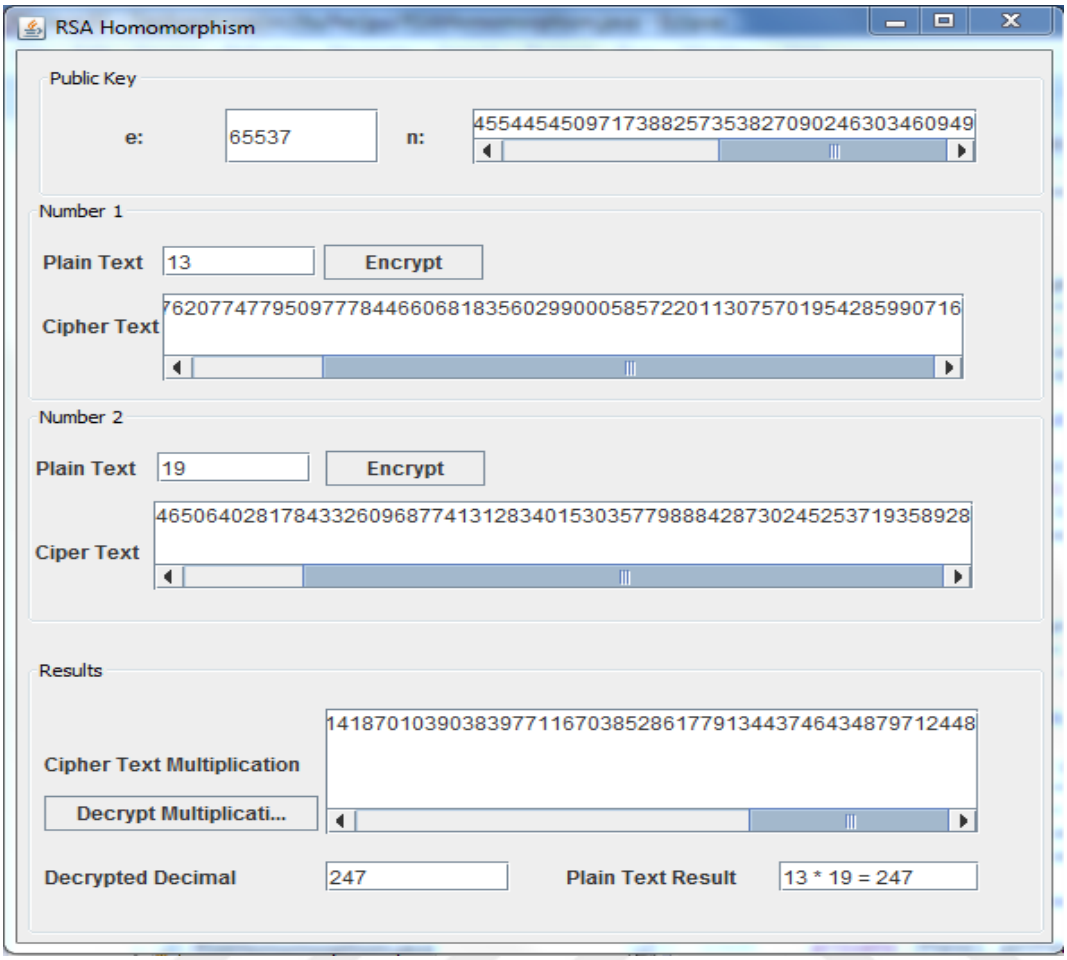
Java programlama diliyle, Java Swing platformunda, RSA açık anahtar kripto sisteminin çarpımsal homomorfik özelliğini gerçekleyelim. Homomorfik çarpımsal özelliğini iki farklı biçimde gerçekledik.

Birincisi, uygulamada sadece Java'nın Standart Development Kit (*sdk*) kütüphanelerinde (Java standart kütüphanesi), Math paketini kullandık, hatta sadece bu pakette bir sınıfı kullandık. Kullandığımız sınıf, BigInteger (büyük sayılar) sınıfıdır. Bu sınıf bize istediğimiz bit uzunluğunda asal sayılar üretilmesine olanak sağlar. Bu belirlediğimiz asalların çarpımındaki modüler aritmetik işlemlerini gerçekleyen metotlar mevcuttur. Aynı zamanda, burada kullandığımız modüler işlemlerden biri kuvvet alma, diğeri ise sayının verilen modül de tersini almaktır. Bu uygulamada sadece BigInteger sınıfı kullanılarak, anahtar üretimi, şifreleme ve deşifreleme algoritmaları, son olarak da homomorfik çarpımsal özelliği gerçeklendi.

İkinci uygulamamızı, Bouncy Castle jarını kullanarak gerçekledik. Bouncy Castle jarının son versiyonunu, Java Swing projemize gösterdik. Burada anahtar oluşturma evresini, Bouncy Castle jarının sunmuş olduğu metotları kullanarak gerçekledik. Şifreleme, deşifreleme ve homomorfik özelliklerini ise Java'nın BigInteger sınıfını kullanarak gerçekleştirdik.

Her iki uygulama için, anahtar uzunluğu 1024-bit seçildi. Üstel şifreleme açık anahtarı, $e = 65337$ seçildi [15]. Her iki uygulamada, 1024-bitlik anahtarı üretmek için 512-bitlik (160 basamaklı) rasgele asal sayı üretildi. Sonuçta, rasgele 1024-bitlik (yaklaşık 320 basamaklı) modül n sayısı üretilmiş oldu. Euler phi fonksiyonu kullanılarak, ilgili sınıflardan bu algoritma çağırılarak, şifreleme üstel e sayısına karşılık, gizli anahtar olan deşifrelemede kullanılacak d sayısını hesapladık.

Anahtar oluşturma işleminden sonra, uygulamalardan rasgele girilecek a ve b tamsayıları, şifrelenir. Şifrelenmiş sayılar, ilgili text alanlarında gösterilir. Homomorfik çarpımsal özelliğini görmek için, ilgili butona basılarak, şifrelenmiş sayılar çarpımı alınır ve deşifre edildiğinde, düz metinler çarpımına eşit olduğu görülür.



Şekil 3.2: RSA nın Çarpımsal Homomorfik Özelliğinin Java Swing İmplementasyonu.

Şimdi de ElGamal kripto sistemin homomorfik özelliğini araştıralım.

3.3. ElGamal Kripto Sistemin Kısmi Çarpımsal Homomorfik Özelliği ve Java İmplementasyonu

ElGamal kripto sistemiyle m_1 , m_2 düz metinleri şifrelenirse;

$E(m_1)=(c_{11}, c_{12}) =(g^{r_1}, m_1 \cdot y^{r_1})$ ve

$E(m_2)= (c_{21}, c_{22}) =(g^{r_2}, m_2 \cdot y^{r_2})$ eşitlikleri elde edilir.

$r_1 , r_2 \in \{1,2,3, \ldots . q-1\}$ ve $m_1 , m_2 \in$ G

$E(m_1) \cdot E(m_2)= (c_{11}, c_{12}) \cdot (c_{21}, c_{22}) = (c_{11} \cdot c_{21} , c_{12} \cdot c_{22})$

$= \left(g^{r_1} \cdot g^{r_2}, (m_1 y^{r_1}) \cdot (m_2 y^{r_2})\right)$

$= (g^{r_1+r_2}, (m_1 \cdot m_2) \cdot y^{r_1+r_2})=E(m_1 \cdot m_2)$ elde edilir.

Sonuç olarak şifreli metinlerin çarpımsal hali, $m_1 \cdot m_2$ nin şifrelenmiş haline eşit olduğu görülür.

∴ ElGamal da RSA gibi çarpımsal homomorfik özelliğe sahiptir. Şimdi bunu basit bir örnekle gösterelim

Örnek:

$s_k = x$ gizli anahtar olmak üzere, $x = 4$ olsun.

$q = 13, g = 2$ ve $y \equiv g^x \equiv 2^4 \equiv 16 \equiv 3 \ (mod\ 13)$ ise,

açık anahtar $p_k = (q, g, y) = (13, 2, 3)$

Dikkat edilirse burada $g = 2$, $\mathbb{Z}_{13}{}^*$ ün bir üreteci, jeneratörü yani ilkel(primitif) köküdür.

Mesajlar, m_1=4 ve m_2=3 olsun. Geçici anahtarlar, r_1=6 ve r_2=8 olsun.

Bu durumda,

$E(m_1) = (c_{11}, c_{12}) = (g^{r_1}, m_1 \cdot y^{r_1}) \equiv (2^6, 4 \cdot 3^6) \ (mod\ 13) \equiv (12, 4)$ bulunur.

Aynı şekilde,

$E(m_2) = (c_{21}, c_{22}) = (g^{r_2}, m_2 \cdot y^{r_2}) \equiv (2^8, 3 \cdot 3^8) \ (mod\ 13) \equiv (9, 1)$ olarak bulunur.

$E(m_1) \cdot E(m_2) = (c_{11}, c_{12}) \cdot (c_{21}, c_{22}) = (c_{11} \cdot c_{21}, c_{12} \cdot c_{22}) = (12 \cdot 9, 4 \cdot 1)$

$\equiv (4, 4) \ (mod\ 13)$ bulunur.

Şimdi de $m_1 \cdot m_2 \equiv 4 \cdot 3 \equiv 12 \ (mod\ 13)$ çarpımını şifreleyelim.

$r = r_1 + r_2 = 14$

$(c_1, c_2) = (2^{14}, 12 \cdot 13^4) \equiv (4, 12 \cdot 9) \ (mod\ 13) \equiv (4, 4)$ bulunur.

Sonuç olarak, ElGamal kripto sistemi de RSA gibi kısmi homomorfik olup, çarpımsal homomorfik şifreleme sistemidir.

Java programlama diliyle, Java Swing platformunda, ElGamal açık anahtar kripto sisteminin çarpımsal homomorfik özelliğini gerçekleyelim. Homomorfik çarpımsal özelliğini iki farklı biçimde gerçekledik.

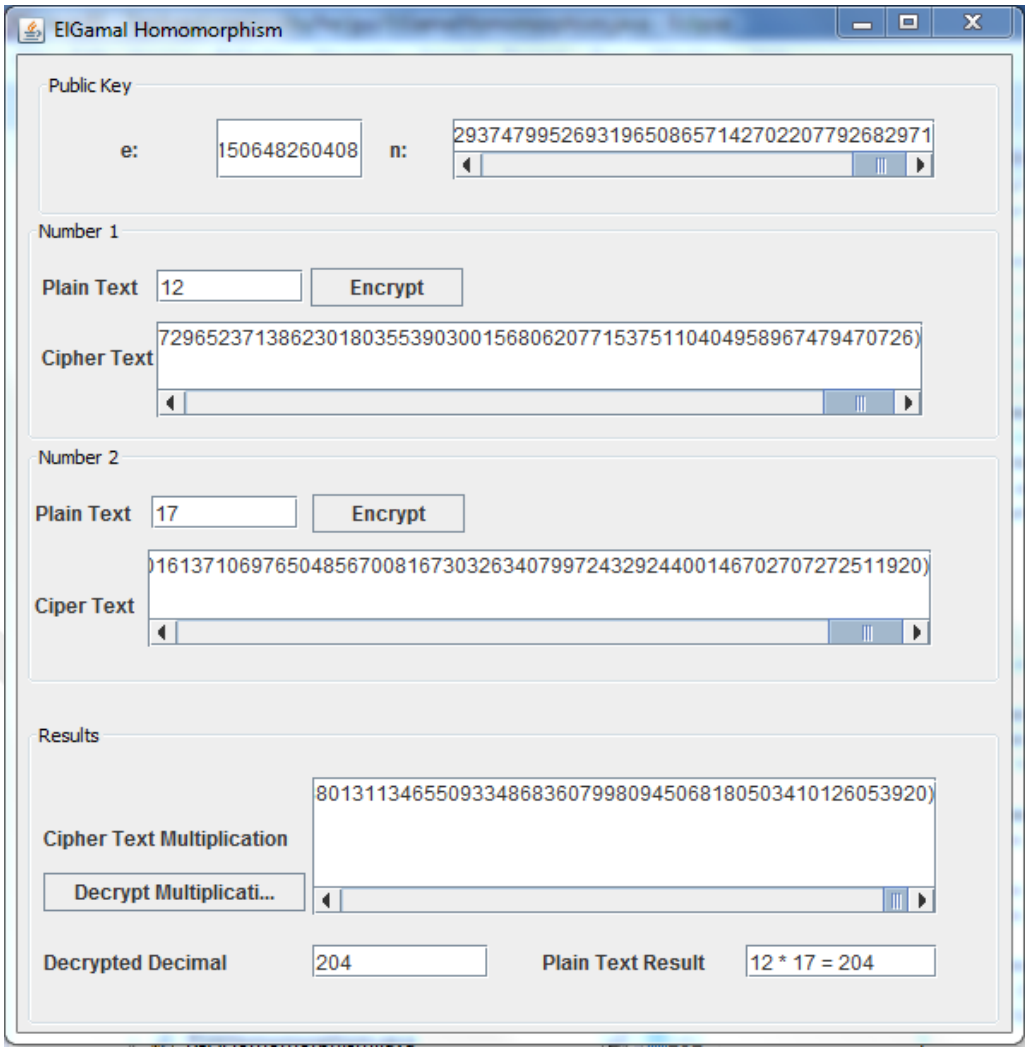
Birincisi, uygulamada sadece Java *sdk* kütüphanelerinde (Java'nın Standart Kütüphanesi), Math paketini kullandık, hatta sadece bu pakette bir sınıfı kullandık. Kullandığımız sınıf, BigInteger (büyük sayılar) sınıfıdır. Bu sınıf bize istediğimiz bit

uzunluğunda random olarak asal sayılar üretilmesine olanak sağlar. Bu belirlediğimiz asalların çarpımındaki modüler aritmetik işlemlerini gerçekleyen metotlar mevcuttur. Aynı zamanda, burada kullandığımız modüler işlemlerden biri kuvvet alma, diğeri ise sayının verilen modülde tersini almaktır. Bu uygulamada sadece BigInteger sınıfı kullanılarak, anahtar üretimi, şifreleme ve deşifreleme algoritmaları, son olarak da homomorfik çarpımsal özelliği gerçeklendi.

İkinci uygulamamızda, bunu Bouncy Castle jarını kullanarak gerçekledik. Bouncy Castle jarının son versiyonunu, Java Swing projemize dahil ettik, böylelikle jarın bize sunmuş anahtar üretimi metodunu kullandık. Burada anahtar oluşturma evresini, Bouncy Castle jarının sunmuş olduğu metotları kullandık. Şifreleme, deşifreleme ve homomorfik özelliklerini Java'nın BigInteger sınıfını kullanarak gerçekledik.

Her iki uygulama için, anahtar uzunluğu 1024-bit seçildi. İlk uygulamamızda r geçici değişken değerinin uzunluğu yaklaşık olarak $\approx 0.66\,p$-bit uzunluğunda (660 bit), Bouncy Castle jarını kullanarak implemente ettiğimiz ikinci uygulamamızda ise, r değişken değerini 1023-bit uzunluğunda aldık. Geçici r nin uzunluğu deşifreleme süresini etkiledi.

Anahtar oluşturma işleminden sonra, uygulamalardan girilecek her iki a ve b tamsayıları, şifrelenir. Şifrelenmiş sayılar, ilgili metin kutucuklarında, metin alanlarında gösterilir. Deşifrelenmiş sayılar, homomorfik çarpımsal özelliğin görmek için, ilgili butona basılarak, deşifrelenmiş sayılar çarpımı alınır ve deşifre edildiğinde, düz metinler çarpımına eşit olduğu görülür.



Şekil 3.3: ElGamal'ın Homomorfik Özelliğinin Java Swing İmplementasyonu.

Şimdi de toplamaya göre kısmi homomorfik olan, ElGamal gibi olasılıksal olan açık anahtarlı Paillier kripto sistemiyle devam edelim.

3.4. Paillier Şifreleme ve Homomorfik Özelliği ve Java İmplementasyonu

Olasılıksal açık anahtarlı bir şifreleme olan Paillier, 1999 yılında Pascal Paillier tarafından geliştirilmiştir. Paillier açık anahtarlı kripto sistem, kısmi homomorfik şifreleme sistemler içerisinde yer alır. Paillier Şifreleme sistemi toplama işlemine göre homomorfiktir. Homomorfik şifreleme, şifreli mesaj üzerinde bir takım hesaplama yapabilmesine olanak sağlar [15]. Paillier şifreleme sistemi güvenli oylama, elektronik oylama, *e-voting* (oylama) sistemlerinde kullanılması önerilmiştir.

Paillier şifreleme şeması da RSA ve ElGamal gibi, anahtar oluşturma, şifreleme ve deşifrelemeden (veri çözme) oluşur.

Anahtar Oluşturma:

p ve q oldukça büyük asal sayıları rastgele seçilir.

$ebob(p \cdot q, (p-1) \cdot (q-1))=1$ (3.4) eşitliğini sağlamalı.

Eğer p ve q eşit uzunlukta seçilirse, (3.4) nolu denklem sağlanır. (3.4) nolu denklem sağlanmazsa yeniden p ve q asalları rastgele seçilerek (3.4) nolu denklem sağlanana kadar devam edilir.

$n = p \cdot q,\ \lambda = ekok(p-1, q-1)$ hesaplanır.

Rastgele $g \in \mathbb{Z}_{n^2}{}^*$ olacak biçimde, ElGamal'daki gibi olasılıksal özelliği sağlayan geçici anahtar seçilir. Burada n sayısı g nin mertebesini(derecesini) bölmelidir. Yani g sayısının mertebesi n nin bir katı olmalıdır.

$\mu = \left(L\left(g^{\lambda}\ (mod\ n^2)\right)^{-1}\right)\ (mod\ n)$, burada L fonksiyonu $L(u) = \frac{u-1}{n}$ biçiminde tanımlıdır.

Açık ve kapalı (gizli) anahtarlar Paillier de aşağıdaki biçimde tanımlıdır.

Açık Anahtar: $p_k = (n, g)$

Gizli (Kapalı) Anahtar: $s_k = (\lambda, \mu)$

Kolay olması açısından;

p ve q eşit uzunlukta asallar,

$g = n + 1,\ \lambda = \varphi(n)$ (Euler phi fonksiyonu), $\mu = \varphi(n)^{-1}(mod\ n)$ seçilir.

Şifreleme Algoritması:

m mesaj ve $m \in \mathbb{Z}_n$ olmak üzere,

$r \in \mathbb{Z}_{n^2}{}^*$, rastgele geçici tamsayı seçilir. Bu durum, Paillier'in olasılıksal açık anahtarlı kripto sistem olmasını sağlar.

Şifreli mesaj c,

$c = g^m \cdot r^n\ (mod\ n^2)$ şeklinde hesaplanır.

Deşifreleme Algoritması:

c şifreli mesaj, $c \in \mathbb{Z}_{n^2}{}^*$ olmak üzere,

$m = L(\ c^{\lambda}\ (mod\ n^2\ )) \cdot \mu\ (mod\ n)$ olarak m düz metini bulunur.

Örnek:

Paillier şifreleme şeması anlamak için küçük parametreler kullanalım.

$p = 11, q = 7 \Rightarrow n = p \cdot q = 11 \cdot 7 = 77$ ve $n^2 = 77^2 = 5929$ bulunur.

g sayısı, geçici anahtar olup, $g \in \mathbb{Z}_{n^2}{}^*$ ve g nin mertebesi n modülünün bir katı olmalıdır.

$g = 5652$ seçilirse, bu şart sağlanır.

$\mathbb{Z}_{n^2}{}^*$ grubunda, g nin mertebesi, $ord(g) = 2310 = 30 \cdot 77$ olup, $n = 77$'nin bir katıdır.

Açık anahtar: $p_k = (n, g) = (77, 5652)$

$\lambda = ekok(p-1, q-1) = (10, 6) = 30$ ve $L(u) = \frac{u-1}{n}$ olmak üzere,

$k = L\left(g^{\lambda}\ (mod\ n^2)\right) = L\left(5652^{30}\ (mod\ 5929)\right) = L(3928) = \frac{(3928-1)}{77} = 51$ bulunur.

k nın çarpmaya göre tersi $(mod\ 77)$ de;

$\mu = k^{-1}\ (mod\ n) \Longrightarrow$

$\mu = 51^{-1}\ (mod\ 77)$ hesaplanırsa, $\mu = 74$ bulunur. Bu durumda; (λ, μ)

Kapalı(gizli) anahtar: $(\lambda, \mu) = (30, 74)$ olarak bulunur. Anahtar oluşturma işlemi tamamlanmış oldu.

Şimdi $M =$ "*" mesajını şifreleyelim.

M'nin ascii kodu alınarak m=42 sayısına dönüştürülür. Dikkat edilirse $m = 42 \in \mathbb{Z}_{77}{}^*$ dir.

r geçici sayısı $r \in \mathbb{Z}_{77}{}^*$ olacak biçimde seçelim. r geçici anahtarı, r=23 olsun.

$c = g^m \cdot r^n\ (mod\ n^2)$ şifreleme algoritması uygulanırsa,

$c = 5652^{42} \cdot 23^{77}\ (mod\ 5929)$

$c = 4624$ şifreli mesajı bulunur.

Şimdi de $c = 4624$ şifreli mesajı deşifre edelim.

$m = L\left(c^{\lambda}\ (mod\ n^2)\right) \cdot\ \mu\ (mod\ n)$ deşifre algoritması uygulanırsa,

$L\,(4624^{30}\,(mod\ 5929))\cdot 74\,(mod\ 77) = L\,(4852)\cdot 74\,(mod\ 77) = 42$ bulunur.

$m = 42$ sayısı ascii kod kullanılarak, $M =$ " * " mesajı bulunur.

Şimdi de Paillier açık anahtarlı kripto sistemin, homomorfik özelliğini ele alalım.

Homomorfik Özelliği:

Şifreleme algoritması kullanılarak, m_1 ve m_2 mesajları şifrelenirse,

$E(m_1, p_k) = g^{m_1}\cdot {r_1}^n\ (mod\ n^2)$ ve

$E(m_2, p_k) = g^{m_2}\cdot {r_2}^n\ (mod\ n^2)$ şifreli mesajları bulunur. Burada, r_1 ve r_2 rastgele geçici tamsayılar ve r_1 , $r_2 \in \mathbb{Z}_{n^2}{}^*$ dir.

$E(m_1, p_k)$ ve $E(m_2, p_k)$ ifadeleri çarpılırsa,

$E(m_1, p_k)\cdot E(m_2, p_k) \equiv (\ g^{m_1}\cdot {r_1}^n)\cdot(\ g^{m_2}\cdot {r_2}^n)\ \ (mod\ n^2)\ \Longrightarrow$

$E(m_1, p_k)\ \cdot\ E(m_2, p_k) \equiv\ \ g^{m_1+m_2}\cdot (r_1\cdot r_2)^n\ (mod\ n^2) \equiv E(m_1 + m_2, p_k)$ (3.5)

bulunur.

Bu durumda Paillier toplama işlemine göre kısmi homomorfiktir.

Örnek:

Paillier için anahtar oluşturmada bahsettiğimiz kolay parametre değerleri seçilirse,

$p=17$, $q=13$ alalım.

Bu durumda, n modül değeri;

$n = 17\cdot 13 = 221$ bulunur.

Euler phi fonksiyonu, gizli anahtardaki λ parametresi seçilebileceği için;

$\varphi(n) = (p-1)\cdot(q-1) = (17\text{-}1)\cdot(13-1) = 16\cdot 12 = 192$ bulunur.

$ebob\big(p\cdot q, (p-1)\cdot(q-1)\big) = ebob\big(n, \varphi(n)\big) = ebob\ (221,192) = 1$ yani aralarında asal.

Bu durumda kolay parametreler için,

$g = n + 1 = 221+1 = 222$ ve $\lambda = \varphi(n) = 192$ alınabilir. Şimdi de μ parametresini hesaplayalım.

$\mu \equiv L\big(g^{\lambda}\ mod\ n^2\big)^{-1}\ (mod\ n)$ ve $L(u) = \frac{u-1}{n}$ dir.

$\mu \equiv L(222^{192}\ mod\ 48841)^{-1}\ (mod\ 221) \Longrightarrow$

$\mu \equiv L(42433)^{-1}$ (mod 221), L (u) fonksiyonunda yerine koyarsak,

$\mu \equiv \frac{42433-1}{221} = \frac{42432}{221} = 192^{-1}\ (mod\ 221)$

$\mu \equiv 160$ bulunur. Anahtar oluşturma işlemi tamamlanmış oldu. Bu durumda,

Açık anahtar değeri: $p_k=(n,g)=(222,221)$

Gizli(kapalı) anahtar değeri: $s_k=(\lambda,\mu)=(192,160)$ olarak setlendi.

Mesajlarımız sırasıyla,

$m_1=148$ ve $m_2=18$ olsun.

Geçici anahtar değerlerini sırasıyla r_1=75 ve r_2=57 seçelim.

Bu durumda şifreleme algoritması kullanılırsa;

$E(m_1,p_k)=g^{m_1}\cdot {r_1}^n\ (mod\ n^2) \equiv 222^{148}\cdot 75^{221}\ (mod\ 48841)$

$\equiv 32709\cdot 15357\ (mod\ 48841) \Longrightarrow c_1 = 31269$ bulunur.

Aynı şekilde,

$E(m_2,p_k)=g^{m_2}\cdot {r_2}^n\ (mod\ n^2)\ \equiv 222^{18}\cdot 57^{221}\ (mod\ 48841)$

$\equiv 3979\cdot 38940\ (mod\ 48841) \Longrightarrow c_2 = 18608$ bulunur.

Artık homomorfik toplama özelliğini gösterebiliriz. (3.5) nolu denklem kullanılırsa,

$E(m_1,p_k)\cdot E(m_2,p_k) \equiv\ g^{m_1+m_2}\cdot (r_1\cdot r_2)^n (mod\ n^2) \equiv E(m_1+m_2,p_k) \Longrightarrow$

$31269\cdot 18608 \equiv E(148+18,p_k)\ \ (mod\ 48841)$ denkliği yer değiştirerek yazarsak

$\Longrightarrow E(166,p_k)\ \ ) \equiv 31269\cdot 18608\ (mod\ 48841) \equiv 10719\ (mod\ 48841)$ (3.6)

(3.6) numaralı denklemi, deşifreleme algoritmasını kullanarak,

$D\big(E(166,p_k)\big) \equiv D(10719) \Longrightarrow D\ (10719) = 166$ (3.7) bulunur.

(3.6) nolu denklemin sol tarafı için, deşifreleme algoritması kullanarak denkliği gösterelim.

$L\left(c^{\lambda}\ (mod\ n^2)\right)\cdot \mu\ (mod\ n)$

$D(c,s_k) \equiv m \equiv L\left(c^{\lambda}\ (mod\ n^2)\right)\cdot \mu\ (mod\ n) \Longrightarrow$

$D(10719) \equiv m \equiv L(10719^{192} \pmod{48841}) \cdot 160 \pmod{221} \Longrightarrow$

$m \equiv L\,(10609) \cdot 160 \pmod{221}$, $L(u)=\frac{u-1}{n}$ fonksiyonunda yerine yazılırsa;

$m \equiv \frac{10609-1}{221} \cdot 160 \pmod{221} \equiv 48 \cdot 160 \pmod{221} \Longrightarrow m \equiv 166$ bulunur.

$\therefore m = m_1 + m_2 = 148+18=166$ değerleri birbirine eşittir.

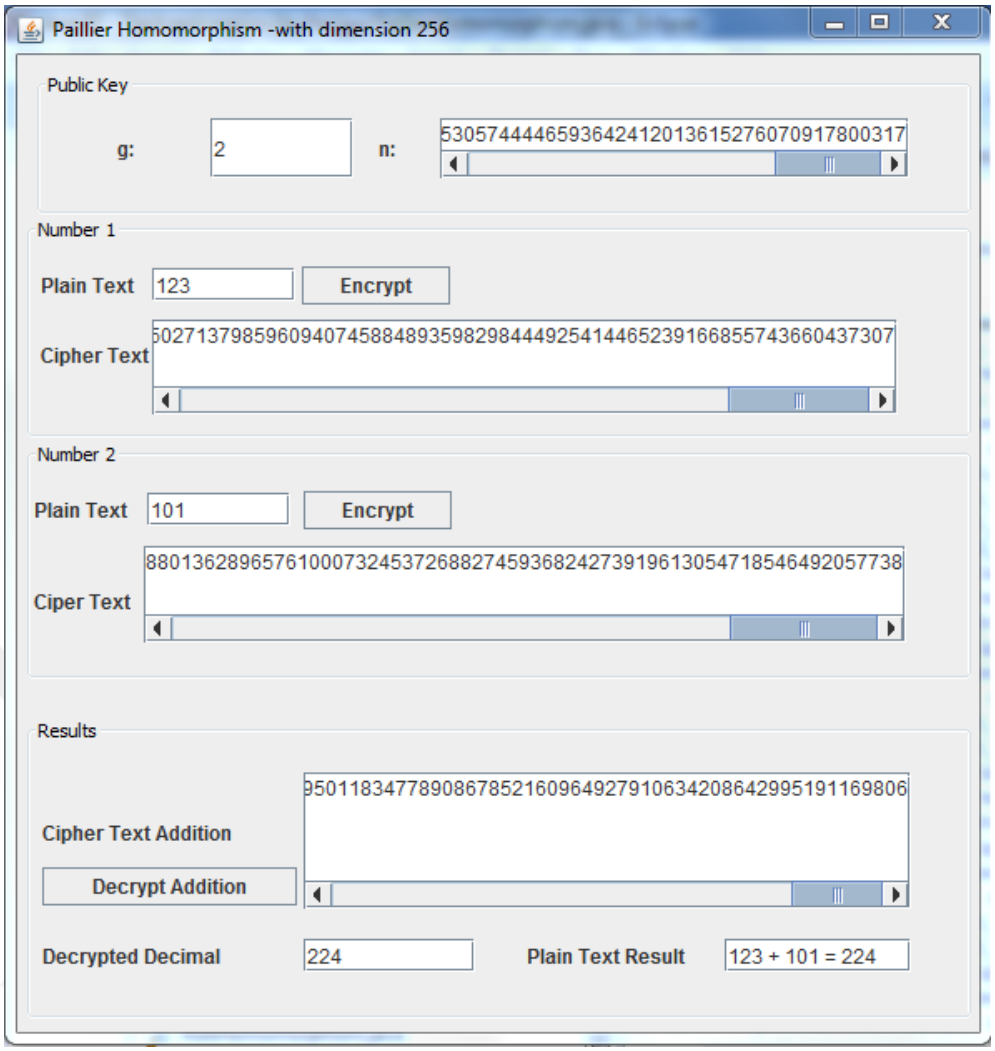
Java programlama diliyle, Java Swing platformunda, Paillier açık anahtar kripto sisteminin homomorfik toplama özelliğini gerçekleyelim.

Homomorfik toplama özelliğini, Java standart kütüphanesindeki Math paketi içindeki sadece bir sınıfı kullanarak gerçekledik. Kullandığımız sınıf, BigInteger (büyük sayılar) sınıfıdır. Bu sınıf bize istediğimiz bit uzunluğunda rasgele asal sayılar üretilmesine olanak sağlar. Bu belirlediğimiz asalların çarpımındaki modüler aritmetik işlemlerini gerçekleyen metotlar mevcuttur. Aynı zamanda, burada kullandığımız modüler işlemlerden biri kuvvet alma, diğeri ise sayının verilen modülde tersini almaktır. Bu uygulamada sadece BigInteger sınıfı kullanılarak, anahtar üretimi, şifreleme ve deşifreleme algoritmaları, son olarak da homomorfik toplama özelliği gerçeklendi.

Uygulamada, anahtar uzunluğu 1024-bit seçildi. 1024-bitlik anahtarı üretmek için 512-bitlik (165 basamaklı) random asal sayı üretir. Sonuçta, random(rastgele) modül n sayısı üretilir.

Geçici r değişkeni, 32-bitlik bir sayı seçildi.

Anahtar oluşturma işleminden sonra, uygulamalardan girilecek her iki a ve b tamsayıları, şifrelenir. Şifrelenmiş sayılar, ilgili text alanlarında gösterilir. Deşifrelenmiş sayılar, homomorfik toplama özelliğini görmek için, ilgili butona basılarak, deşifrelenmiş sayılar toplamı alınır ve deşifre edildiğinde, düz metinler toplamına eşit olduğu görülür.



Şekil 3.4: Paillier'in Homomorfik Özelliğinin Java Swing İmplementasyonu.

3.5. Tam Homomorfik Şifrelemeye Genel Bir Bakış

Homomorfik şifreleme, bulut bilişimde ve gizliliğin önemli olduğu bankacılık sektöründe birçok uygulaması yapılabilecek çok kullanışlı bir araçtır.

Homomorfik şifreleme bize, işlemlerin şifreli metinler üzerinde gerçekleştirilmesi, oluşan güvensizlik ortamını engelleyebilecek ve veri gizliliğini sağlayacaktır. Bu noktada, verilerin işlenmesi sırasında homomorfik şifreleme kullanılarak, şifre çözümüne gerek kalmadan, üzerinde işlemler yapılabilmekte ve işlemlerin şifresiz sonucunu sadece ilgili kullanıcı görebilmektedir [16]. Şifreli mesajlar üzerinde birtakım matematiksel işlemler gerçekleştirme fikri ilk olarak Rivest, Adleman ve Dezorous tarafından önerilmiştir [16].

Homomorfik şifreleme, uygulanabilecek matematiksel işlem sayısına göre, kısmi (PHE) ve tam (FHE) homomorfik olarak adlandırılmaktadır. Şifreli metinler üzerinde

gerçekleştirilebilen işlem sayısı tek ise kısmi homomorfik, iki tane ise tam homomorfik şifreleme denir.

Daha önceki kısımlarda, şifreli metinler üzerinde sadece bir işlemi gerçekleyen (toplama veya çarpma) kısmi homomorfik (PHE) şifrelemelerden bahsettik. Çarpma ve toplama gibi her iki işlemi birden, şifreli metinler üzerinde yapabildiğimiz şifreleme sistemlere, tam homomorfik şifreleme sistemler denir ve FHE ile gösterilir. FHE (tam homomorfik şifreleme) kavramı ilk kez Rivest tarafından [17] önerilmiştir. Bu problem 2009'a kadar çözümsüz kalmış ve Gentry [6] tarafından çözümü sunulmuş ve kriptografi tarihinde önemli bir buluş olmuştur. Craig Gentry [18] tarafından tez çalışmasında kriptografi dünyasının gündemine getirilen tam homomorfik şifreleme hem toplama hem çarpma işlemini bir arada sağlar. Craig Gentry bu çalışmasını "İdeal kafesler" ve tamsayılar üzerinde bir dereceye kadar tam homomorfik üzerine yapmıştır. Bölümün bu kısmında sadece tam homomorfik (FHE)'nin tanımını ve tamsayılar üzerinde FHE'yi verip, Java'da bunu gerçekleyeceğiz.

3.6. Halka Homomorfizması ve Tam Homomorfik Şifreleme Tanımı

FHE şifreleme, halka homomorfizması olarak düşünülebilir. Halka homomorfizması, verilen iki halka R ve S arasında tanımlanan f fonksiyonu için,

$f: R \to S, \forall\, a, b \in R$ için,

$$f(a+b) = f(a) + f(b) \tag{3.3}$$

$$f(a \cdot b) = f(a) \cdot f(b) \tag{3.4}$$

oluyorsa f ye halka homomorfizması denir.

Kısmi Homomorfik Şifrelemeler (PHE) grup homomorfizması, Tam Homomorfik Şifrelemeler (FHE) halka homomorfizması olarak düşünülebilir.

Örnek:

$f: \mathbb{Z}_2 \to \mathbb{Z}_2$olmak üzere, $f(x) = x^2$ ve $x \in \{0,1\}$

f'nin halka homomorfizması olduğunu gösterelim.

$f(x+y) = (x+y)^2 = x^2 + +y^2 = x^2 + y^2 = f(x) + f(y), 2xy \;\; = 0$ çünkü $\mathbb{Z}_2$ deyiz.

$$f(x \cdot y) = (x \cdot y)^2 = x^2 \cdot y^2 = f(x) \cdot f(y) \tag{3.5}$$

$\therefore$ f fonksiyonu halka homomorfizmasıdır.

Grup homomorfizmasında olduğu gibi, halka homomorfizma kavramını kriptografiye uyarlayalım.

(P, C, K, E, D) kripto sistemi için P, C sırasıyla şifreleme ve deşifreleme (veri çözme) algoritmalarını, K ise anahtar uzayını göstermektedir.

$(P, \oplus_p, \otimes_p)$ ve $(C, \oplus_c, \otimes_c)$ kripto sistemlerde iki ayrı halka olmak üzere,

$E_k: P \to C$, burada $k \in K$, k gizli(kapalı) veya açık anahtar. Bu durumda

E_k şifreleme algoritması(fonksiyonu), tam homomorfik fonksiyondur.

$\forall\, a, b \in P$ ve $k \in K$ için,

$$E_k(a) \oplus_c E_k(b) = E_k(a \oplus_p b) \tag{3.6}$$

$$E_k(a) \otimes_c E_k(b) = E_k(a \otimes_p b)$$

Şifreleme şeması yukarıdaki eşitlikleri sağlıyorsa, tam homomorfik şifreleme denir.

Craig Gentry [19] tam homomorfik şifreleme çalışmasını 2009'da soyut cebirdeki ideal kafes kavramını kullanarak göstermiştir. Kafes tabanlı kriptografi ile tam homomorfik şifrelemeyi burada göstermemiz, kafes noktaları ile ilgili birçok teorik bilgiyi vermemiz gerektiğinden dolayı tam homomorfik şifrelemeyi tamsayılar üzerinde gerçekleyeceğiz.

3.7. Tamsayılar Üzerinde Kapalı (Gizli) Anahtar ile Tam Homomorfik Şifreleme (SWHE)

Tam homomorfik şifrelemenin (FHE) kolay anlaşılması için, kapalı (gizli) anahtar ile tamsayılar üzerinde bir dereceye kadar tam homomorfik (SWHE) şifrelemeden bahsedeceğiz. Açık anahtarlı şifreleme sistemlerinde olduğu gibi bu da üç aşamadan oluşur.

Anahtar Oluşturma:

Burada sadece gizli(kapalı) anahtar oluşturacağız.

p: tek tamsayı olmak üzere, $p \in [2^{\eta-1}, 2^{\eta}]$

η: tamsayı olan kapalı anahtarın bit türünden uzunluğu

gizli anahtar: $s_k = p$

Şifreleme Algoritması:

$c = E(m, s_k)$ ile gösterilir.

s_k : gizli anahtar

m: mesaj, $m \in \{0,1\}$

c: şifreli mesaj

Şifreleme algoritması aşağıdaki biçimde tanımlanır.

$E(m, s_k) = c = p \cdot q + 2 \cdot r + m$

q ve r rastgele seçilen, geçici anahtarlar, birer tamsayı olup $2 \cdot r < \frac{p}{2}$ dir.

Deşifreleme Algoritması:

$m = D(c, s_k)$ deşifreleme algoritması ve gizli anahtar $s_k = p$ olmak üzere

$D(c, s_k) \equiv m \equiv (c \ (mod\ p))\ (mod\ 2)$

biçiminde tanımlanmıştır. Çünkü,

$(c \ (mod\ p))\ (mod\ 2) \equiv (p \cdot q + 2 \cdot r + m \ \ (mod\ p))\ (mod\ 2)$

$= (2 \cdot r + m\ )\ (mod\ 2) \equiv m$ bulunur.

Şimdi bunu bir örnekle açıklayalım.

Örnek:

$s_k = p = 17$ kapalı anahtarı ve $m = 1$ bit mesajı verilsin.

$c = p \cdot q + 2 \cdot r + m = 17 \cdot 2 + 2 \cdot 3 + 1 = 41$, $r = 3$ ve $q = 2$ alındı

$c = 41$ şifreli metini bulunur.

Şimdi şifreyi çözelim:

$(c \ (mod\ p))\ (mod\ 2) \equiv (41\ (mod\ 17))\ (mod\ 2) \equiv 7\ (mod\ 2) \equiv 1$ bulunur.

Sonuç olarak $m = 1$ bulunur.

Tam Homomorfik Özelliği:

$E(m_1, s_k) = c_1 = p \cdot q_1 + 2 \cdot r_1 + m_1$

$E(m_2, s_k) = c_2 = p \cdot q_2 + 2 \cdot r_1 + m_2$

Şifreli metinleri verilsin, sırasıyla şifreli metinler toplamını ve çarpımını bulalım.

$c_1 + c_2 = (q_1 + q_2)\cdot p + 2\cdot(r_1 + r_2) + (m_1 + m_2)$, şifreli metinler toplamı.

$c_1 \cdot c_2 = (p \cdot q_1 \cdot q_2 + 2\cdot q_1 \cdot r_2 + 2\cdot q_2 \cdot r_1 + m_1 \cdot q_2 + m_2 \cdot q_1) \cdot p$

$+ 2\cdot (2 \cdot r_1 \cdot r_2 + m_1 \cdot r_2 + m_2 \cdot r_1) + m_1 \cdot m_2$, şifreli metinler çarpımı.

Burada $r_1 + r_2 < \frac{p}{2}$ ve $2\cdot r_1 \cdot r_2 + m_1 \cdot r_2 + m_2 \cdot r_1 < \frac{p}{2}$ dir.

Şifreli toplam ve çarpımlara sırasıyla deşifre algoritması uygulanırsa,

$(c_1 + c_2 \pmod p) \pmod 2 = m_1 + m_2$ bulunur, çünkü

$c_1 + c_2 = (q_1 + q_2)\cdot p + 2\cdot (r_1 + r_2) + (m_1 + m_2)$, toplamında

aşağıdaki terimler sıfıra denktir;

$(q_1 + q_2)\cdot p \equiv 0 \pmod p$ ve $2\cdot(r_1 + r_2) \equiv 0 \pmod 2$.

Aynı şekilde,

$(c_1 \cdot c_2 \pmod p) \pmod 2 = m_1 \cdot m_2$ bulunur, çünkü

$c_1 \cdot c_2 = (p \cdot q_1 \cdot q_2 + 2\cdot q_1 \cdot r_2 + 2\cdot q_2 \cdot r_1 + m_1 \cdot q_2 + m_2 \cdot q_1) \cdot p$

$+ 2\cdot (2 \cdot r_1 \cdot r_2 + m_1 \cdot r_2 + m_2 \cdot r_1) + m_1 \cdot m_2$, çarpımında aşağıdaki terimler sıfıra denktir;

$(p \cdot q_1 \cdot q_2 + 2\cdot q_1 \cdot r_2 + 2\cdot q_2 \cdot r_1 + m_1 \cdot q_2 + m_2 \cdot q_1) \cdot p \equiv 0 \pmod p$ ve

$2\cdot (2 \cdot r_1 \cdot r_2 + m_1 \cdot r_2 + m_2 \cdot r_1) \equiv 0 \pmod 2$

∴ Tamsayılar üzerinde yukarıdaki biçimde tanımlanmış, gizli anahtarlı kripto sistem tam homomorfiktir (FHE).

Burada işlemler $\mathbb{Z}_2$ de gerçeklendiği için;

$+$ işlemi: veya (or) ($\vee$) işlemidir.

$\cdot$ işlemi: ve (and) ($\wedge$) işlemidir.

Bu kripto sistemde şifreleme bit bit yapılır ve şifreli metin üzerinde veya (or), ve (and) işlemleri sağlanır. Tam homomorfik özelliğini bir örnekle açıklayalım.

Örnek:

Gizli anahtar $s_k = p = 17$,

Mesajlar m_1=0 ve m_2=1 olsun.

Geçici anahtarlar r_1 , r_2 , q_1, q_2 değerlerini de

r_1 =q_1=1 ve r_2 =q_2=2 alalım.

Bu durumda mesajları şifreleme algoritma ile sırasıyla şifrelersek;

$E(m_1, s_k)$= c_1= $p \cdot q_1$+2·r_1 +m_1 $\Longrightarrow E(0, s_k$=17) = c_1 =17·1 +2·1 +0=19

$E(m_2, s_k)$= c_2= $p \cdot q_2$+2·r_2 +m_2 $\Longrightarrow E(1, s_k$=17) =c_2= 17·2 +2·2 +1=39 değerleri bulunur.

Sırasıyla bu şifreli metinlere toplama ve çarpma işlemleri uygulayalım ve sonrasında deşifre edelim.

$(c_1 + c_2 \pmod p) \pmod 2 = m_1 + m_2 \equiv (39 + 19 \pmod{17}) \pmod 2 \equiv 7 \pmod 2 \equiv 1 \equiv 0 + 1 \equiv m_1 + m_2 \equiv m_1$ or $m_2 \equiv m_1 \vee m_2$ bulunur.

$(c_1 \cdot c_2 \pmod p) \pmod 2 = m_1 \cdot m_2 \equiv (39 \cdot 19 \mod 17)) \pmod 2 \equiv 10 \pmod 2 \equiv 0 \equiv 0 \cdot 1 \equiv m_1 \cdot m_2 \equiv m_1$ and $m_2 \equiv m_1 \wedge m_2$ bulunur.

Not: $r_1 + r_2 \geq max\,(r_1, r_2)$

$2 \cdot r_1 \cdot r_2 + m_1 \cdot r_2 + m_2 \cdot r_1 \geq max\,(r_1, r_2)$

r ifadesi, imaj işlemedeki gibi gürültü parametresi olarak adlandırılabilir. r gürültü parametresinden dolayı art arda toplama yapıldığında her defasında 2 katına çıkar.

Çarpmada ise karesi, küpü gibi kuvvetleri biçiminde artar. Bundan dolayı şifreli metin aşırı derecede büyük değerlere ulaşır.

3.8. Tamsayılar Üzerinde Açık Anahtar ile Tam Homomorfik Şifreleme (SWHE) ve Java İmplementasyonu

Kapalı anahtarlı bir dereceye kadar tam homomorfik şifrelemede mesajı şifrelemek için her iki algoritmada, şifreleme veya deşifrelemede, s_k=p kapalı anahtarı kullanılmıştı.

Şimdi, açık anahtarlı bir dereceye kadar homomorfik şifrelemelerden bahsedelim.

Bu şifreleme şeması birçok parametreden oluştuğu için, ilk önce parametre tanımlarını verelim.

Parametreler:

γ: tamsayı olan açık anahtarın bit türünden uzunluğu.

η : tamsayı olan kapalı anahtarın bit türünden uzunluğu.

ρ: tamsayı olan gürültü ifadesinin bit türünden uzunluğu.

τ: açık anahtardaki tamsayıların sayısı.

Uygun parametre değerleri:

$\rho = \lambda\,, \quad = 2.\lambda, \qquad \eta = O(\lambda^2), \quad \gamma = O(\lambda^5)$ ve $\tau = \gamma + \lambda$

Anahtar Oluşturma:

η – bit uzunluğunda $s_k = p$ gizli(kapalı) anahtarı seçilir.

Açık anahtar;

$x_i = \; p \cdot q_i + \; r_i,$ burada $q_i \in \left[0, {2^\lambda}/{p}\;\right)$ ve $r_i \in [-2^\rho, 2^\rho\;)$

her defasında random seçilen q ve r değerleri ile x tamsayıları üretilir. Üretilen x değerleri açık anahtarı aşağıdaki biçimde oluşturur.

Açık anahtar $p_k = \langle x_0, x_1, \ldots., x_\tau \rangle$, burada x_0 tamsayısı $x_0, x_1, \ldots., x_\tau$ tamsayı değerlerinden büyük seçilir.

Şifreleme Algoritması

$c = E(m, p_k)$ şifreleme algoritması, p_k açık anahtar,

m mesaj ve $m \in \{0,1\}$ ve $S \subset \{1,2,3 \ldots, \tau\}$ olmak üzere;

$E(m, p_k) = c = (m + 2{\cdot}r + 2{\cdot}\sum_{i \in S} \boldsymbol{x_i}\,)\;\; (mod\; x_0)$

Deşifreleme Algoritması

$m = D(c, s_k)$, burada c şifreli metin ve $s_k = p$ kapalı anahtar olmak üzere,

deşifreleme algoritması aşağıdaki biçimde tanımlanır.

$m = D(c, s_k) = (\; c \; (mod\; p))\; (mod\; 2)$ olarak tanımlıdır.

Aynı zamanda, $c \; (mod\; p) \equiv c - p \cdot \left[{c}/{p}\right]$ olarak tanımlanır.

Not: $\left[{c}/{p}\right]$ ifadesi, ${c}/{p}$ den küçük en büyük tamsayı değeri, yani tamdeğer fonksiyonu olarak tanımlanır.

Homomorfik özelliği kapalı anahtarda olduğu gibi gösterilebilir. İlk önce toplama, sonrasında da çarpma işlemine göre homomorfik özelliğini gösterelim.

$c_1 = (m_1 + 2{\cdot}r_1 + 2{\cdot}\sum_{i\in S} x_i\ )$

$c_2 = (m_2 + 2{\cdot}r_2 + 2{\cdot}\sum_{i\in S} x_i\ )$, şifreli metinleri verilsin.

$c_1\ +\ c_2 = 2{\cdot}\sum_{i\in S} x_i\ +\ 2{\cdot}\sum_{t\in S} x_t\ + 2{\cdot}\ (\ r_1\ +\ r_2\ )\ +\ (\ m_1 + m_2\ )$, şifreli metinler toplamı.

Bu eşitliğe, deşifreleme algoritması uygulanırsa;

$(c_1\ +\ c_2\ \ (mod\ p))\ (mod\ 2) = m_1 + m_2$ bulunur, çünkü toplamdaki aşağıda verilen terimleri sıfıra denktir.

$2{\cdot}\sum_{i\in S} x_i + 2{\cdot}\sum_{t\in S} x_t \equiv 0\ (mod\ p)(\ mod\ 2)$ ve $2{\cdot}(\ r_1 + r_2\ ) \equiv 0\ (mod\ 2)$

$c_1\ \cdot\ c_2 = (m_1 + 2{\cdot}r_1\ + 2{\cdot}\sum_{i\in S} x_i\ )\cdot\ (m_2 + 2{\cdot}r_2\ + 2{\cdot}\sum_{t\in S} x_t\ )$, ifadesini açmadan direkt modül alma işlemleri alınırsa,

$c_1\ \cdot\ c_2 = m_1.m_2$ olduğu görülür.

∴ Tamsayılar üzerinde yukarıdaki biçimde tanımlanmış, açık anahtarlı kripto sistem tam homomorfiktir (FHE).

Burada işlemler $\mathbb{Z}_2$ de gerçeklendiği için;

$+$ işlemi: veya (or) ($\vee$) işlemidir.

$\cdot$ işlemi: ve (and) ($\wedge$) işlemidir.

Örnekle homomorfik özelliğinin sağlandığını görelim.

Örnek:

Kapalı anahtar $s_k = p = 10001$ ve $\tau = 4$ için

Random(rastgele) seçilen q ve r değerleri sırasıyla,

$(q_0,\ q_1,\ q_2,\ q_3) = (36,27,34,6)$

$(r_0,\ r_1,\ r_2,\ r_3) = (8, 5, 4, 2)$ olmak üzere,

$x_i = \ p \cdot q_i + r_i$ eşitliği kullanılarak açık anahtar p_k hesaplanırsa,

$p_k = (x_0,\ x_1,\ x_2,\ x_3) = (360044, 270032, 340038, 60008)$ olarak bulunur.

Dikkat edilirse x_0 değeri en büyük olacak biçimde x_0 ve r_0 değerleri seçildi.

Anahtarları oluşturduk. Şimdi de m_1=0 ve m_2 =1 bit mesajlarını şifreleyelim.

Şifreleme algoritmasını kullanarak m_1 =0 mesajı şifrelenirse;

Geçici r değerini, $r_1 = 31$ alalım.

$E(m_1, p_k)$= c_1= $(m_1 + 2{\cdot}r_1 + 2{\cdot}\sum_{i\in S} x_i)$ $(mod\ x_0)$, burada $S = \{1,3\}$ alınırsa,

c_1= 0+2·31+2· $(270032 + 60008)$ =660142 ≡ 30098$(mod\ 360044)$ bulunur.

Aynı şekilde m_2=1 bit mesajının şifreleyelim.

Geçici r değerini, $r_2 = 11$ alalım.

$E(m_2, p_k)$ =$c_2 = (m_2 + 2{\cdot}r_2 + 2{\cdot}\sum_{i\in S} x_i)$ $(mod\ x_0)$, burada S={2,3} alınırsa

c_2 = 1+2·11+2· $(340038 + 60008)$ = 800115≡80027 $(mod\ 360044)$ bulunur.

Şifreli mesajlar sırasıyla, c_1=30098 ve c_2=80027 bulunur.

Homomorfik toplam ve çarpım değerlerini bulmak için şifreli mesajları toplayalım ve çarpalım sonrasında da bulduğumuz bu değerlere deşifreleme algoritmasını uygulayalım.

c_1=30098, c_2=80027 ⟹

$c_1 + c_2$ =300098 + 80027, bu toplam ifadesine deşifreleme algoritmasını uygularsak,

$c_1 + c_2$ =300098 +80027 $(mod\ s_k{=}p{=}10001)$ ≡ 87 $(mod\ 2)$ ≡ 1≡0+1= m_1+ m_2 bulunur.

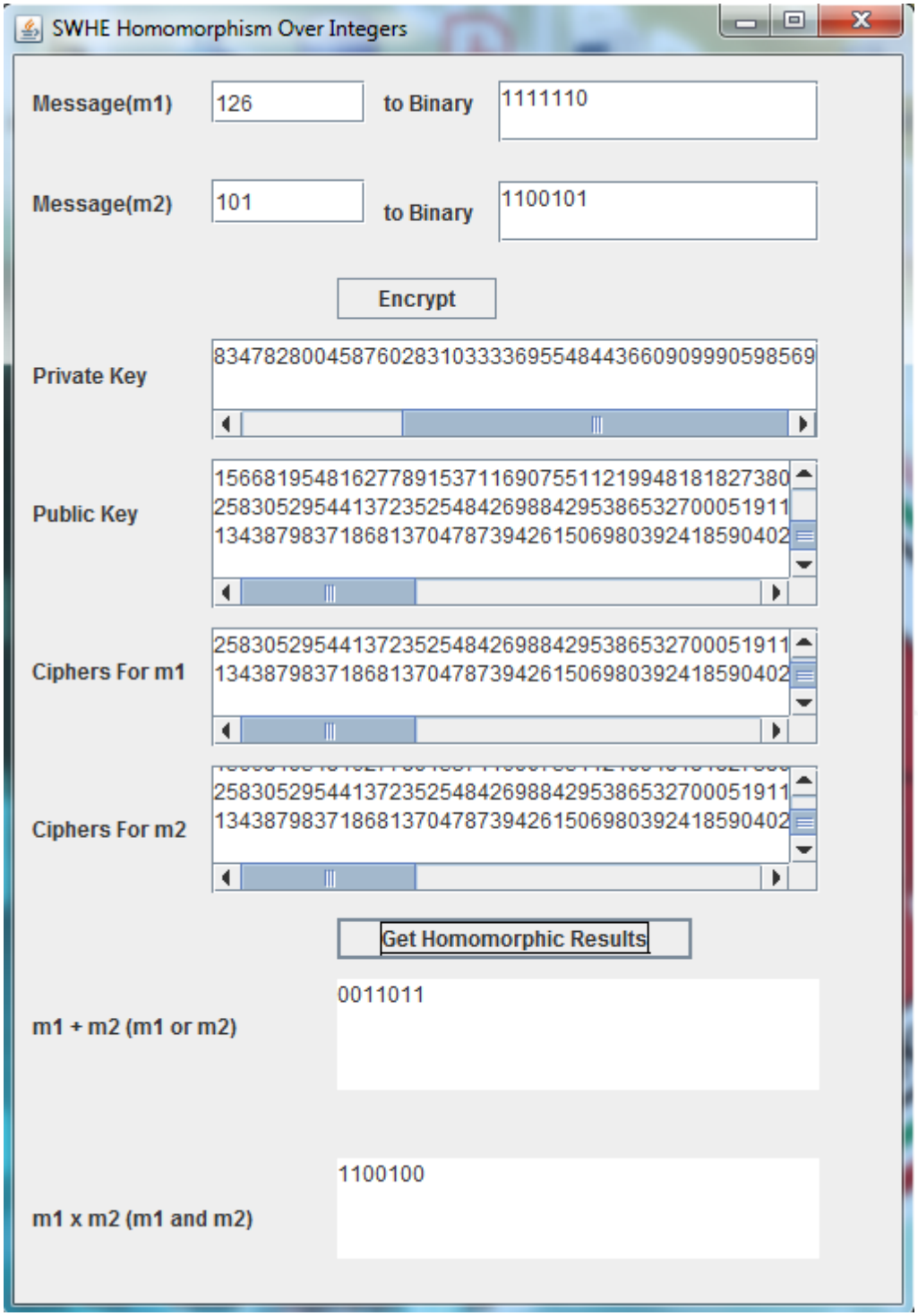
$c_1 \cdot c_2$ =300098 ·80027, bu çarpım ifadesine deşifreleme algoritmasını uygularsak,

$c_1 \cdot c_2$ =300098 ·80027 $(mod\ p = 10001)$ ≡1292 $(mod\ 2)$ ≡0 ≡0·1= $m_1.m_2$ bulunur.

∴ Homomorfik özelliklerinin sağlandığı örnek üzerinden de görülmüş olur.

Tamsayılar üzerinde Homomorfik toplam ve çarpım (or ve and) özelliklerini, Java programlama diliyle, Java swing platformun da yine sadece BigInteger sınıfını kullanarak gerçekledik. Parametre değerlerini; $\lambda = 15$, kapalı anahtarın uzunluğu η $=O(\lambda^2)$=225 olarak aldık. Açık anahtardaki x tamsayılarının sayısını, girilen sayıların bit uzunluğu kadar aldık. Girilen herhangi iki tamsayı, bit türünden ekranda

gösterilmiş olup bit bit şifrelenmiştir. Girilen sayıların toplamı ve çarpımı deşifre edildiğinde, düz metinler toplam (or) ve çarpım (and) sonucuna eşit olduğu görülür.



Şekil 3.5: Tam Sayılarda Tam Homomorfik Özelliğin Java Swing İmplementasyonu.

4. RSA VE ELGAMAL HOMOMORFİK ŞİFRELEMELERİN VERGİ ÖDEME SİSTEMİ UYGULAMASINDAKİ PERFORMANS ANALİZİ VE DEĞERLENDİRMESİ

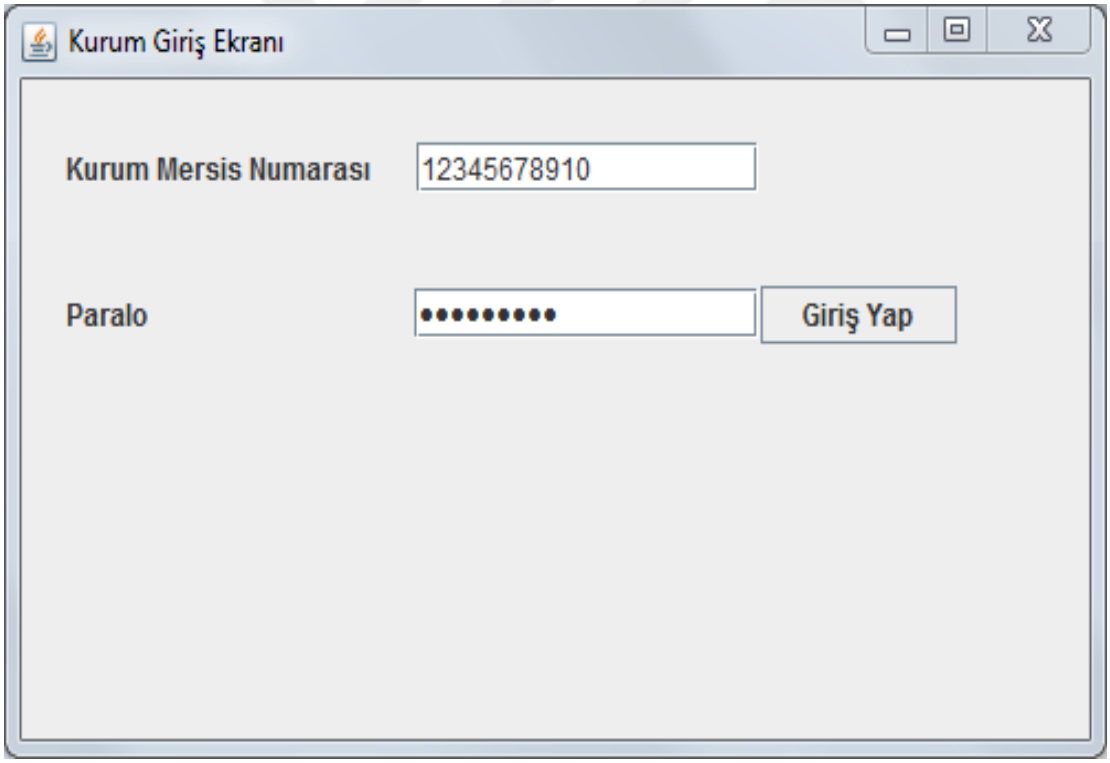
Uygulamamızda, web servis aracılığı ile kurumlar giriş yaptıkları ara yüzden ödemeleri gereken vergi miktarını güvenli bir şekilde görebilmektedirler. Uygulamada kurumların Kamu İhale Kurumunda (KİK), ihale bedelleri tutulmaktadır. Maliye Bakanlığı Gelir İdaresi Başkanlığında ise kurumun ödemesi gereken vergi oranı tutulmaktadır. Her iki yerde sadece bu bilgiler tutulmakta ve ilgili kurum ancak mersis numarası ile bu bilgilere ulaşabilmektedir.

Buradaki amacımız kurumun ait ödemesi gereken vergiyi sorgulayabileceği bir sistem tasarlamak. Bu nu yaparken de çarpmaya göre homomorfik olan RSA ve ElGamal Kripto sistemleri kullanmak. Vergi bildiğimiz üzere sözleşme bedeli ile vergi oranının çarpımının 100' e bölünmesiyle hesaplanabilir. İlgili kurum web ara yüzünde, mersis numarası ve parolasını girer ve sorgulama sisteminde sorgula butonuna bastığında sistem ilgili kurumun mersis numarası ve açık anahtarı ile web servis kullanılarak ilk önce Kamu İhale Kurumuna oradan da Vergi Dairesi Başkanlığından ilgili veriler şifrelenerek kullanıcı ara yüzüne döner. Şifreli veriler kullanıcı bilgisayarında Homomorfik çarpımsal işleminden sonra veriler çözülerek güvenli bir şekilde gösterilir. Buradaki amacımız web te verilerin şifrelenerek kullanıcıya ulaşmasını sağlamak. Gerçi Https protokolü da kullanılarak bu yapılabilirdi. Buradaki amacımız homomorfik çarpımsal özelliklere sahip RSA ve ElGamal açık anahtarlı sistemlerin performans açısından karşılaştırılması. Sistem şifreleme ve şifre çözmede Java'nın sadece BigInteger sınıfı kullanılarak; ayrıca 3. parti meşhur yazılımlarda Bouncy Castle jar'ı kullanılarak gerçeklendi. Her iki sonuçta bize RSA'nın performans açısından hem veri şifreleme ve veri çözmede ElGamal'dan daha kullanışlı olduğu sonucunu verdi. Ayrıca ElGamal ile veri şifrelendiğinde aynı anahtar uzunluğuna sahip RSA' ya göre 2 katı kadar şifreli veri elde edilmektedir. ElGamal'ın RSA'ya göre avantajı ElGamal'ın olasılı açık anahtar kripto sistem olmasıdır yani ayni düz metin aynı anahtar ile şifrelendiğinde farklı şifreli metinler elde edilmesidir. İlk

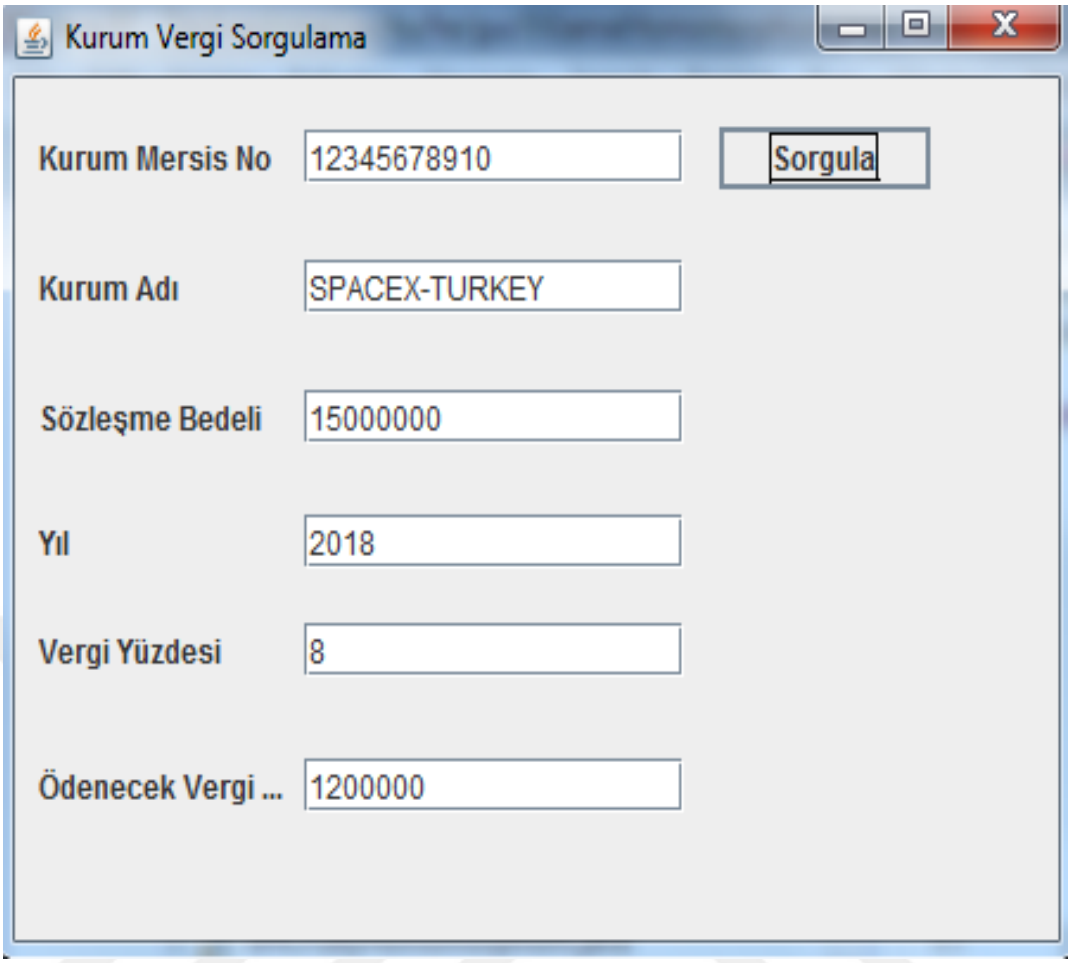
uygulamada ve Bouncy Castle jar'ı kullanılarak gerçeklenen ikinci uygulamamızda, sözleşme bedeli 15000000 (15 milyon) lira ve vergi oranı tamsayı olup 8 olarak alınmıştır. 1024-bit, 2048-bit ve 3072-bit uzunluğundaki anahtarlar kullanılarak sadece Java'nın BigInteger sınıfı kullanılarak RSA ve ElGamal açık anahtarlı kripto sistemler gerçeklenmiştir.

Bu uygulamalardan elde edilen bulgular tablo ve grafiklerle sonuçlandırılmıştır. Elde edilen sonuçlar mikro saniye (saniyenin milyonda biri) türünden verilmiş olup, beş ayrı deneme sonuçları tablolara aktarılmıştır. Sonrasında her bir tablonun sonuçların grafikleri çizilmiştir. İlgini sonuçlar, işlemcisi Intel(R) Core (TM) i5-4300U CPU @1.90 GHz 2.50 GHz olan bir bilgisayar üzerinde çalıştırılan uygulamadan elde edilmiştir.

İlgili kurum giriş ekranından sonra vergi borcu sorgulama ekranına yönlendirilir.



Şekil 4.1: Online Vergi Ödeme Sistemi (OVÖS) Giriş Ekranı.



Şekil 4.2: Online Vergi Ödeme Sistemi (OVÖS) Vergi Sorgulama Ekranı.

Sırasıyla anahtar bazında uygulamalardan elde edilen sonuçları görelim. Sonuçlara, 1024-bit uzunluğunda anahtar kullanıldığında her iki uygulama sonucunda elde edilen veriler tablosu ve grafikleri ile başlayalım.

4.1. İlk Uygulamada, 1024-bit Anahtar Kullanımı ve Elde Sonuçlar

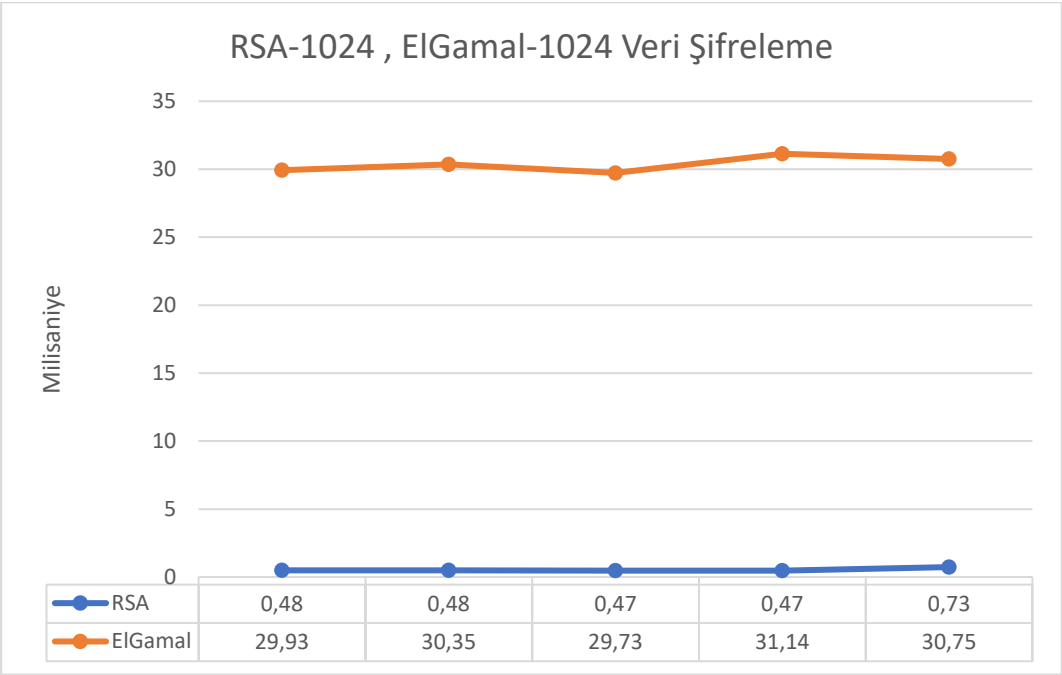
Uygulamanın 1024-bit anahtar kullanılmasıyla elde edilen sonucun çizelgesi aşağıdaki gibidir.

Çizelge 4:1. Uygulama ile RSA-1024, ElGamal-1024 için elde edilen veriler.

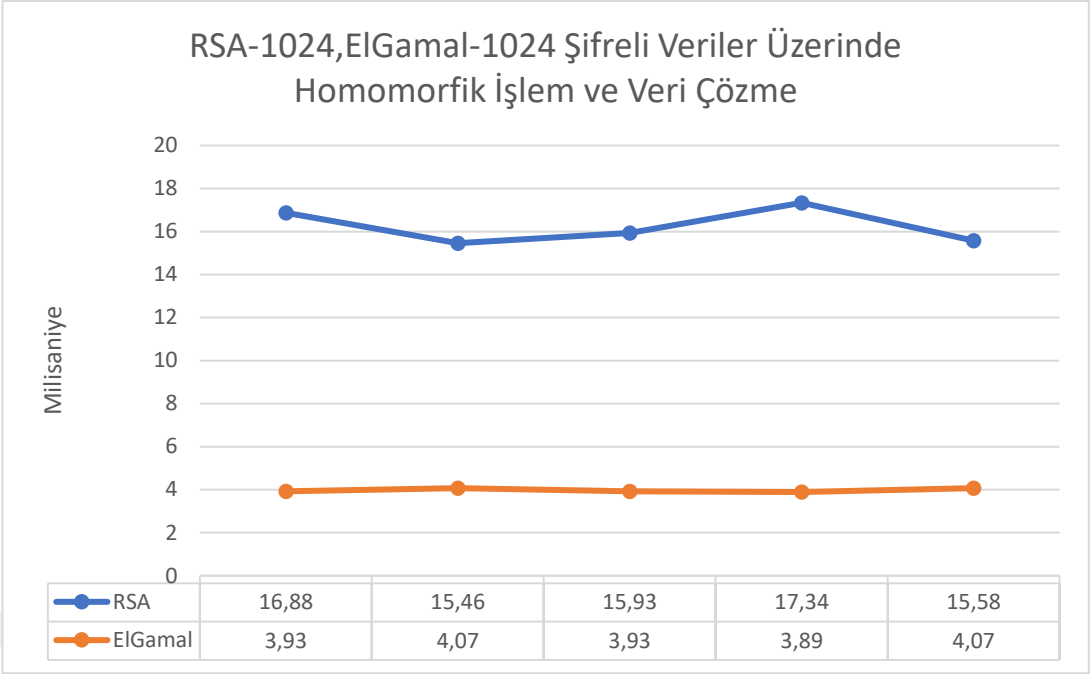
Mikro Saniye Türünden Sorgu Sonuçları		RSA-1024	ElGamal-1024
1.Sorgu	Şifreleme	481	29931
Sonucu	Homomorfik İşlem ve Veri Çözme	16889	3931
	Toplam Süre	17370	33862
2.Sorgu	Şifreleme	481	30351
Sonucu	Homomorfik İşlem ve Veri Çözme	15463	4070
	Toplam Süre	15944	34421
3.Sorgu	Şifreleme	479	29732
Sonucu	Homomorfik İşlem ve Veri Çözme	15935	3935
	Toplam Süre	16414	33667
4.Sorgu	Şifreleme	477	31146
Sonucu	Homomorfik İşlem ve Veri Çözme	17345	3897
	Toplam Süre	17822	35043
5.Sorgu	Şifreleme	734	30756
Sonucu	Homomorfik İşlem ve Veri Çözme	15585	4071
	Toplam Süre	16319	34827

İlgili çizelgede veri şifreleme, homomorfik işlem ve veri çözme, tüm işlemler sonucunda yani veri şifreleme, homomorfik işlem ve veri çözme sonucunda verinin gecikme süresi verilmiştir.

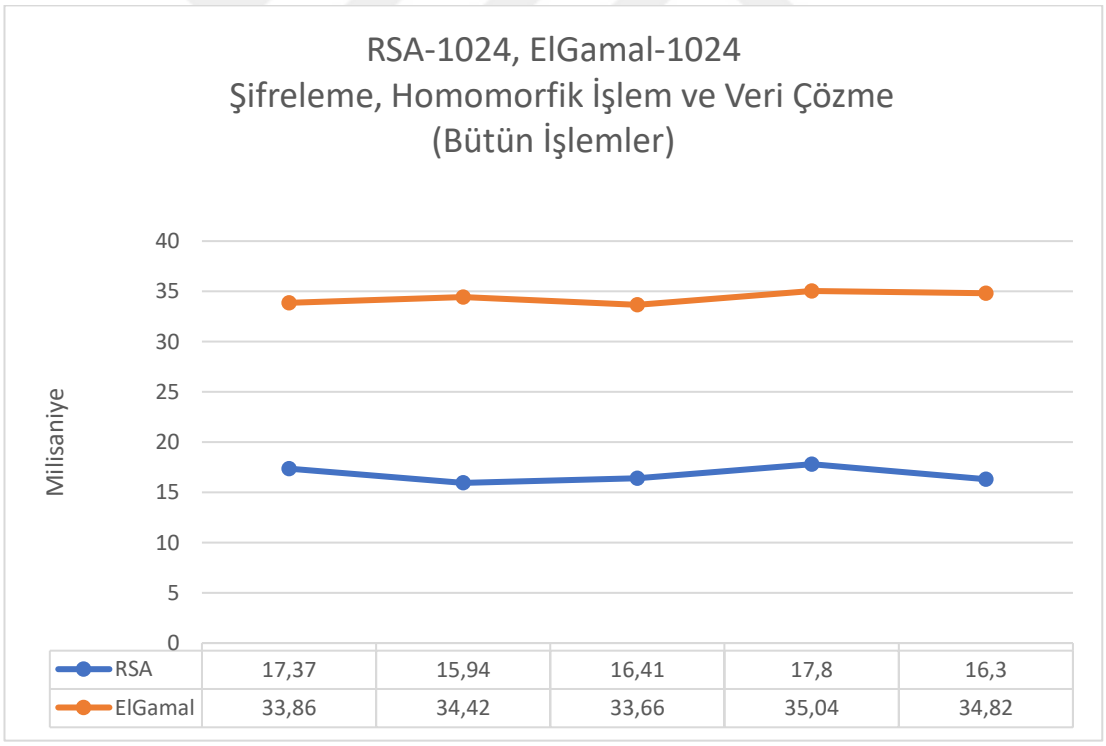
Çizelgeden elde edilen verilerin grafiksel gösterimleri aşağıdaki gibidir.



Şekil 4.3: RSA-1024, ElGamal-1024 Veri Şifreleme (Sunucu Tarafı).



Şekil 4.4: RSA-1024, ElGamal-1024 Homomorfik İşlem ve Veri Çözme (İstemci).



Şekil 4.5: RSA-1024, ElGamal-1024 Tüm İşlemler, Toplam Gecikme Süresi.

Çizelge ve Şekiller incelendiğinde; RSA'nın tartışmasız olarak veri şifreleme ve toplam süre açısından oldukça ElGamal'e göre daha tercih edilebilir olduğu görülmektedir. Rastgele üretilen k değerinin uzunluğu burada $\log p$ [14] olarak seçildiği için deşifrelemede ElGamal'ın RSA'dan daha iyi olduğu görülmektedir. Her

defasında üretilen bu k değerinden dolayı ElGamal olasılı açık anahtar kriptolama sistemidir. Şifreleme işlemleri sunucu tarafında, homomorfik çarpma işlemi ve şifre çözme kullanıcı tarafında yapıldığı düşünülürse, sunucuların işlemcisi uygun seçildiğinde ElGamal da düşünülebilir. ElGamalda toplam gecikme süresi ortalama 30-35 milisaniye iken RSA da bu gecikme süresi 15-20 milisaniyedir. ElGamalda sunucuda yapılacak iyileştirme ile bu fark kapanabilir.

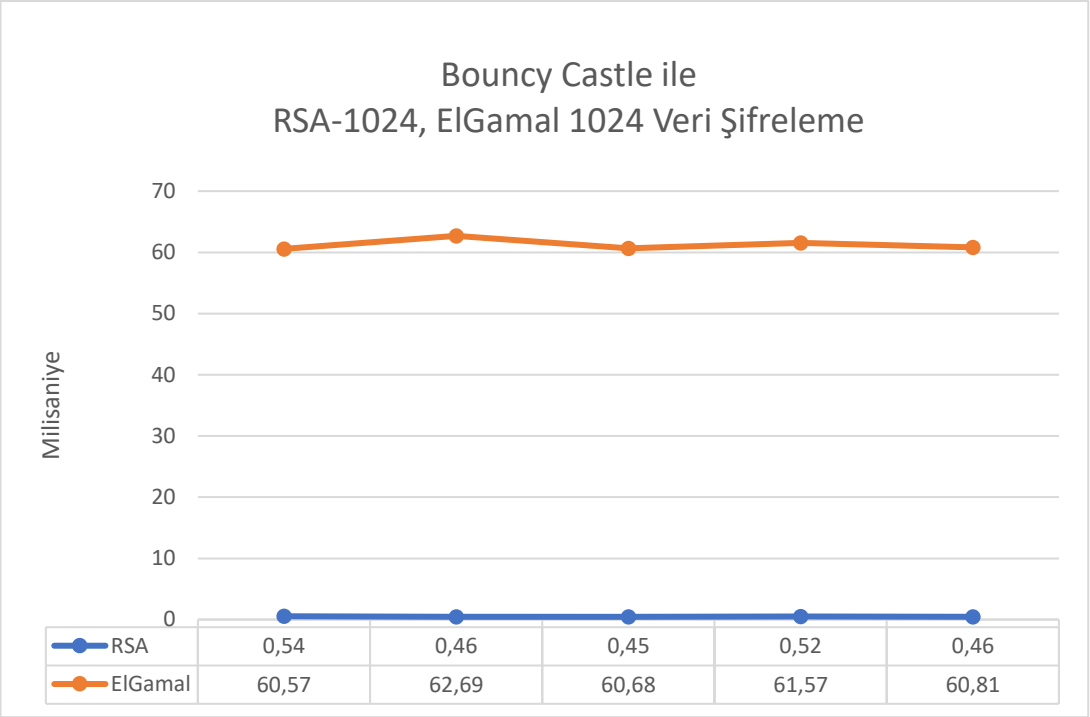
4.2. İkinci Uygulamada (Bouncy Castle), 1024-bit Anahtar Kullanımı ve Elde Sonuçlar

İkinci uygulama yani Bouncy Castle jar'ı kullanılarak, anahtar uzunluğu 1024 alınarak elde edilen verilerin çizelgesi ve grafikleri aşağıdaki gibidir.

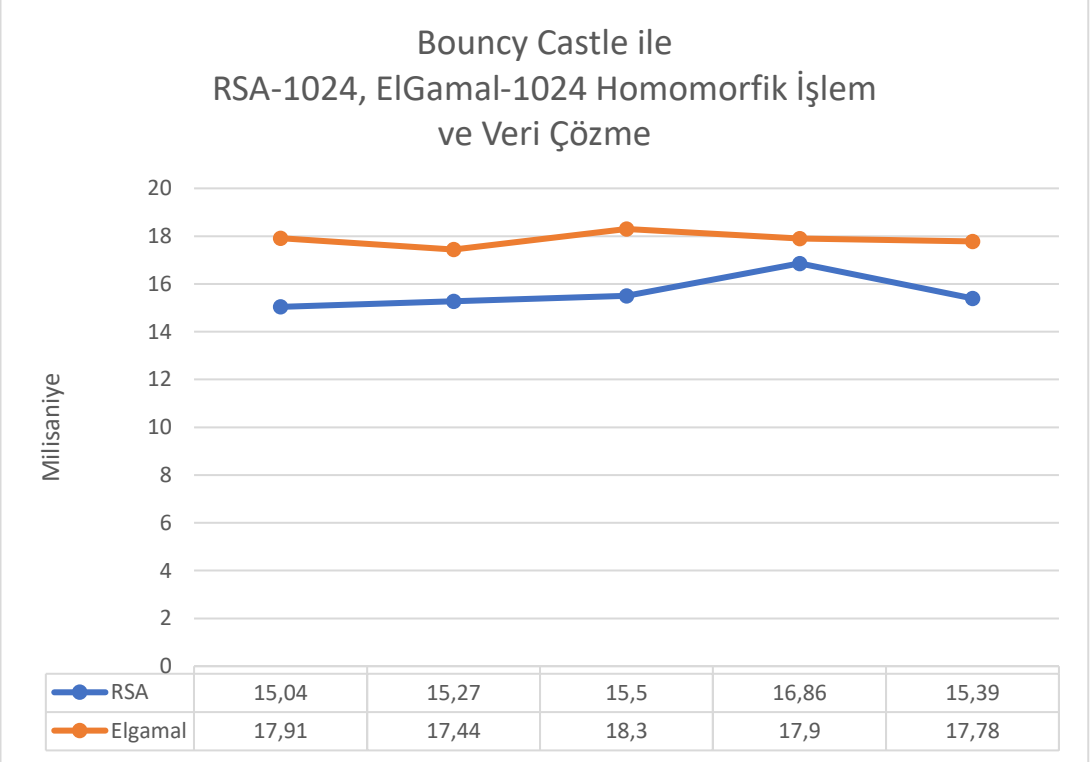
Çizelge 4.1: Bouncy Castle ile RSA-1024, ElGamal-1024 için elde edilen veriler.

Mikro Saniye Türünden Sorgu Sonuçları (Bouncy Castle)		RSA-1024	ElGamal-1024
1.Sorgu	Şifreleme	541	60578
Sonucu	Homomorfik İşlem ve Veri Çözme	15046	17914
	Toplam Süre	15588	78492
2.Sorgu	Şifreleme	463	62694
Sonucu	Homomorfik İşlem ve Veri Çözme	15273	17447
	Toplam Süre	15737	80141
3.Sorgu	Şifreleme	459	60686
Sonucu	Homomorfik İşlem ve Veri Çözme	15506	18302
	Toplam Süre	15965	78988
4.Sorgu	Şifreleme	523	61573
Sonucu	Homomorfik İşlem ve Veri Çözme	16864	17908
	Toplam Süre	17387	79481
5.Sorgu	Şifreleme	461	60819
Sonucu	Homomorfik İşlem ve Veri Çözme	15396	17781
	Toplam Süre	15858	78601

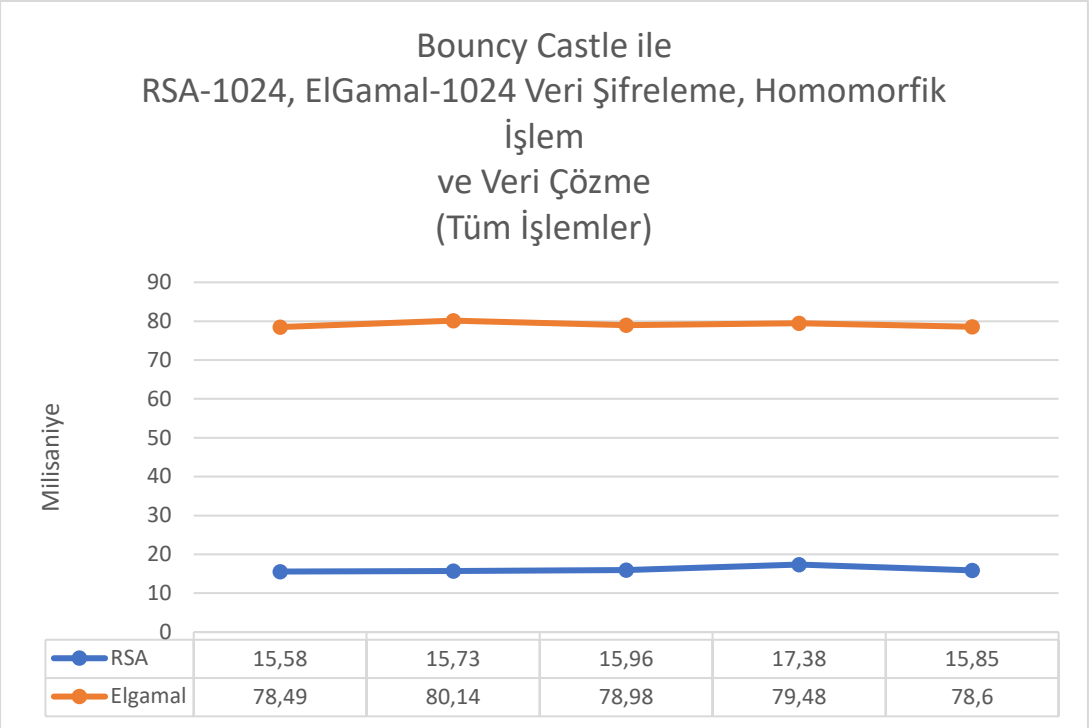
Çizelgenin, veri şifreleme, homomorfik işlem ve veri çözme ve toplam gecikme süresi bazında grafiksel gösterimleri aşağıdaki gibidir.



Şekil 4.6: Bouncy Castle ile RSA-1024, ElGamal-1024 Veri Şifreleme (Sunucu Tarafı).



Şekil 4.7: Bouncy Castle ile RSA-1024, ElGamal-1024 Homomorfik İşlem ve Veri Çözme (İstemci).



Şekil 4.1: Bouncy Castle ile RSA-1024, ElGamal-1024 Tüm İşlemler, Toplam Gecikme Süresi.

İkinci uygulamada, Bouncy Castle jar'ı kullanılmıştır. Burada RSA'nın veri şifrelemenin ve toplamdaki üstünlüğünün yanında veri çözme ve homomorfik işlemdeki az bir farkla üstünlüğü göze çarpmaktadır. Bu da Bouncy Castle rasgele üretilen k nın uzunluğunu $\log p$ yerine $p-1$ **[20]** yani daha uzun almasıdır. Burada sonuç olarak RSA'da ortalama gecikme süresi toplamda ortalama 15-20 milisaniye olurken ElGamal'da ortalama gecikme süresi 80 milisaniye olmuştur. İlk uygulamadaki ile RSA'nın gecikme süresi aynı kalırken, ElGamal'ın toplam gecikme süresi 2 katına çıkmıştır. Tabi ki bunun nedeni k nın uzunluğunun $\log p$ yerine $p-1$ seçilmesidir. 1. Uygulamada k'nın uzunluğu $\log p$ yerine $p-1$ seçilirse RSA'nın ElGamal'dan, 2. Uygulamada olduğu gibi veri çözmede biraz daha iyi olduğu görülür

Şimdide anahtar uzunluğu 2048-bit seçildiğinde, 1. Uygulama ile 2. Uygulamanın sonuçlarının sırasıyla yukarıda yaptığımız gibi inceleyelim.

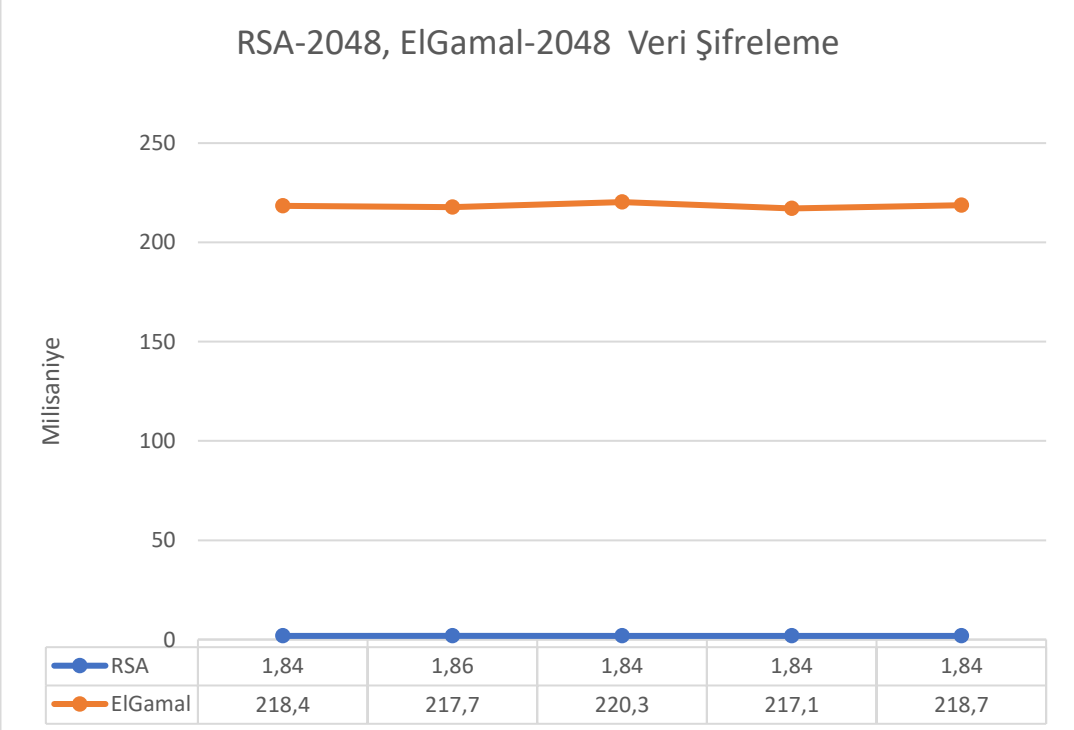
4.3. İlk Uygulamada, 2048-bit Anahtar Kullanımı ve Elde Sonuçlar

Uygulamada 2048-bit anahtar kullanılmasıyla elde edilen sonucun çizelgesi aşağıdaki gibidir.

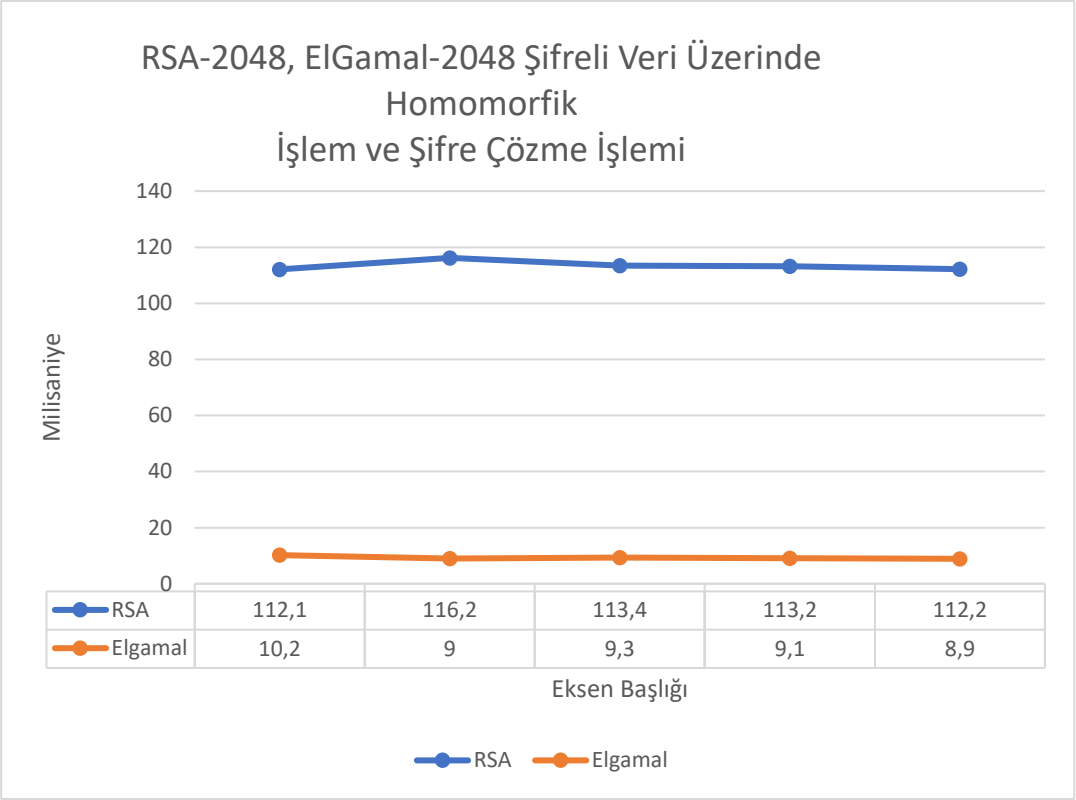
Çizelge 4.3: 1. Uygulama ile RSA-2048, ElGamal-2048 için elde edilen veriler.

Mikro Saniye Türünden Sorgu Sonuçları		RSA-2048	ElGamal-2048
1.Sorgu Sonucu	Şifreleme	1842	218440
	Homomorfik İşlem ve Veri Çözme	112143	10262
	Toplam Süre	113985	228702
2.Sorgu Sonucu	Şifreleme	1869	217792
	Homomorfik İşlem ve Veri Çözme	116226	9000
	Toplam Süre	118095	226792
3.Sorgu Sonucu	Şifreleme	1842	220312
	Homomorfik İşlem ve Veri Çözme	113478	9351
	Toplam Süre	115320	229663
4.Sorgu Sonucu	Şifreleme	1843	217144
	Homomorfik İşlem ve Veri Çözme	113263	9187
	Toplam Süre	115106	226331
5.Sorgu Sonucu	Şifreleme	1840	218744
	Homomorfik İşlem ve Veri Çözme	112217	8913
	Toplam Süre	114057	227657

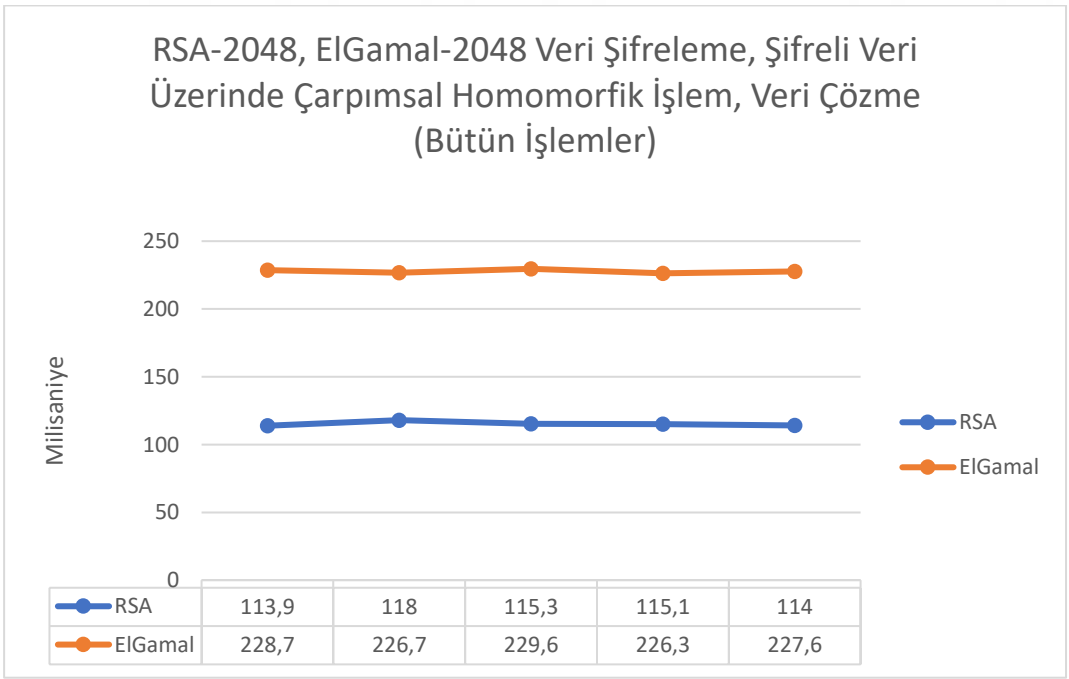
Çizelgenin, veri şifreleme, homomorfik işlem ve veri çözme ve toplam gecikme süresi bazında grafiksel gösterimleri aşağıdaki gibidir.



Şekil 4.9: RSA-2048, ElGamal-2048 Veri Şifreleme (Sunucu Tarafı).



Şekil 4.10: RSA-2048, ElGamal-2048 Homomorfik İşlem ve Veri Çözme (İstemci).



Şekil 4.11: RSA-2048, ElGamal-2048 Tüm İşlemler, Toplam Gecikme Süresi.

Tablo ve grafikler incelendiğinde; RSA'nın tartışmasız olarak veri şifreleme ve toplam süre açısından oldukça ElGamal e göre daha tercih edilebilir olduğu görülmektedir.

Rastgele üretilen k değerinin uzunluğu burada $\log p$ [14] olarak seçildiği için deşifrelemede ElGamal'ın RSA dan daha iyi olduğu görülmektedir. Her defasında üretilen bu k değerinden dolayı ElGamal açık anahtarlı kripto sistemi, olasılı açık anahtar kriptolama sistemi olarak bilinmektedir. Şifreleme işlemleri sunucu tarafında, homomorfik çarpma işlemi ve şifre çözme kullanıcı tarafında yapıldığı düşünülürse, sunucuların işlemcisi uygun seçildiğinde ElGamal da düşünülebilir. ElGamalda toplam gecikme süresi ortalama 225 milisaniye iken RSA'da bu gecikme süresi yaklaşık ortalama 115 milisaniyedir. ElGamal gecikme süresi RSA'nın gecikme süresinin yaklaşık 2 katıdır. ElGamal'da sunucuda yapılacak iyileştirme ile, sunucunun işlemci gücü ile bu fark kapanabilir. Ayrıca RSA ve ElGamal 1024- bitlik anahtar ile karşılaştırıldığında gecikme süreleri 6-7 katına çıkmıştır.

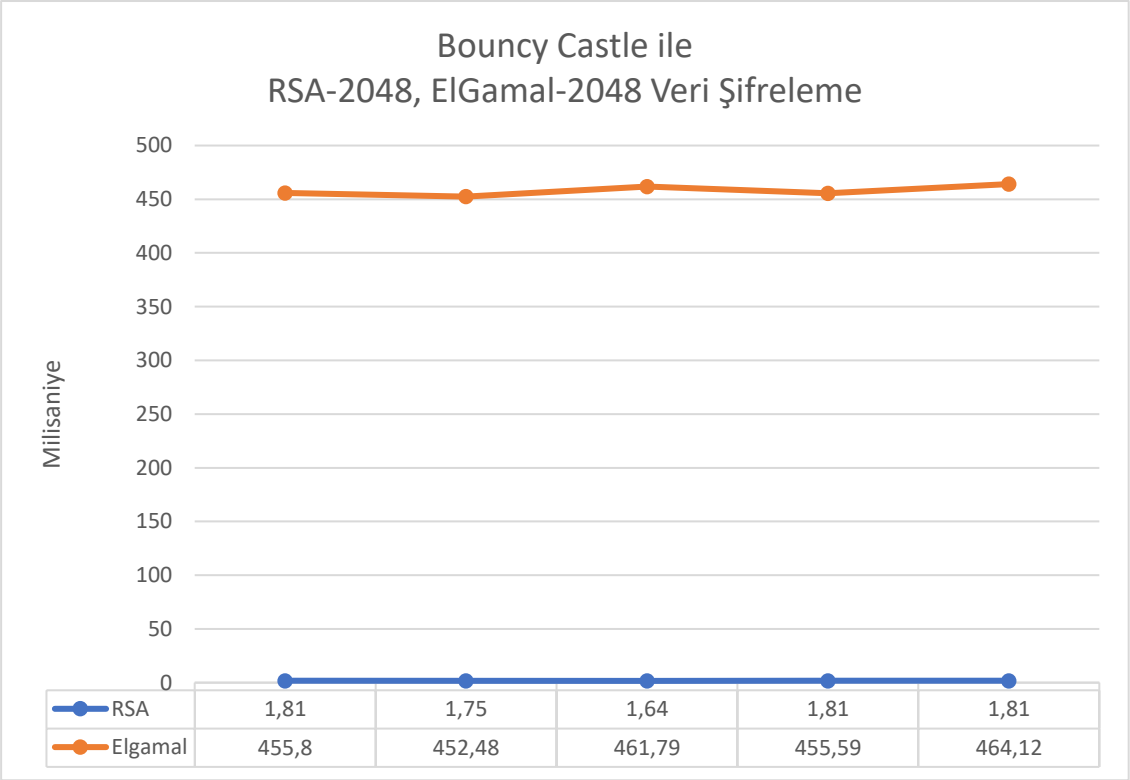
4.4. İkinci Uygulamada (Bouncy Castle), 2048-bit Anahtar Kullanımı ve Elde Sonuçlar

Vergi ödeme sisteminin Bouncy Castle ve 2048-bitlik anahtarlar kullanarak elde edilen verilerin çizelge ve şekillerini verelim ve sonuca bakalım.

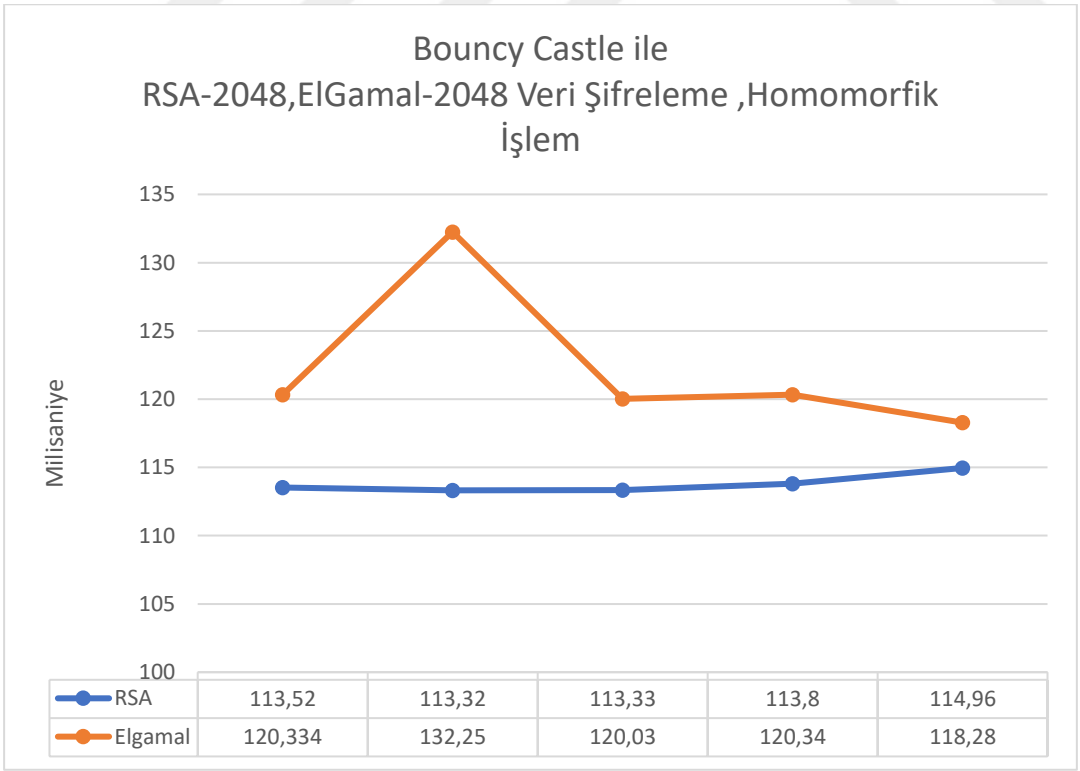
Çizelge 4.4: Bouncy Castle ile RSA-1024, ElGamal-1024 için elde edilen veriler.

Mikro Saniye Türünden Sorgu Sonuçları (Bouncy Castle)		RSA-2048	ElGamal-2048
1.Sorgu Sonucu	Şifreleme	1801	455804
	Homomorfik İşlem ve Veri Çözme	113522	120334
	Toplam Süre	115323	576139
2.Sorgu Sonucu	Şifreleme	1754	452488
	Homomorfik İşlem ve Veri Çözme	113322	132250
	Toplam Süre	118095	584738
3.Sorgu Sonucu	Şifreleme	1648	461798
	Homomorfik İşlem ve Veri Çözme	113332	120037
	Toplam Süre	114981	581835
4.Sorgu Sonucu	Şifreleme	1810	455598
	Homomorfik İşlem ve Veri Çözme	113800	120347
	Toplam Süre	115610	575945
5.Sorgu Sonucu	Şifreleme	1815	464128
	Homomorfik İşlem ve Veri Çözme	114963	118287
	Toplam Süre	116778	582416

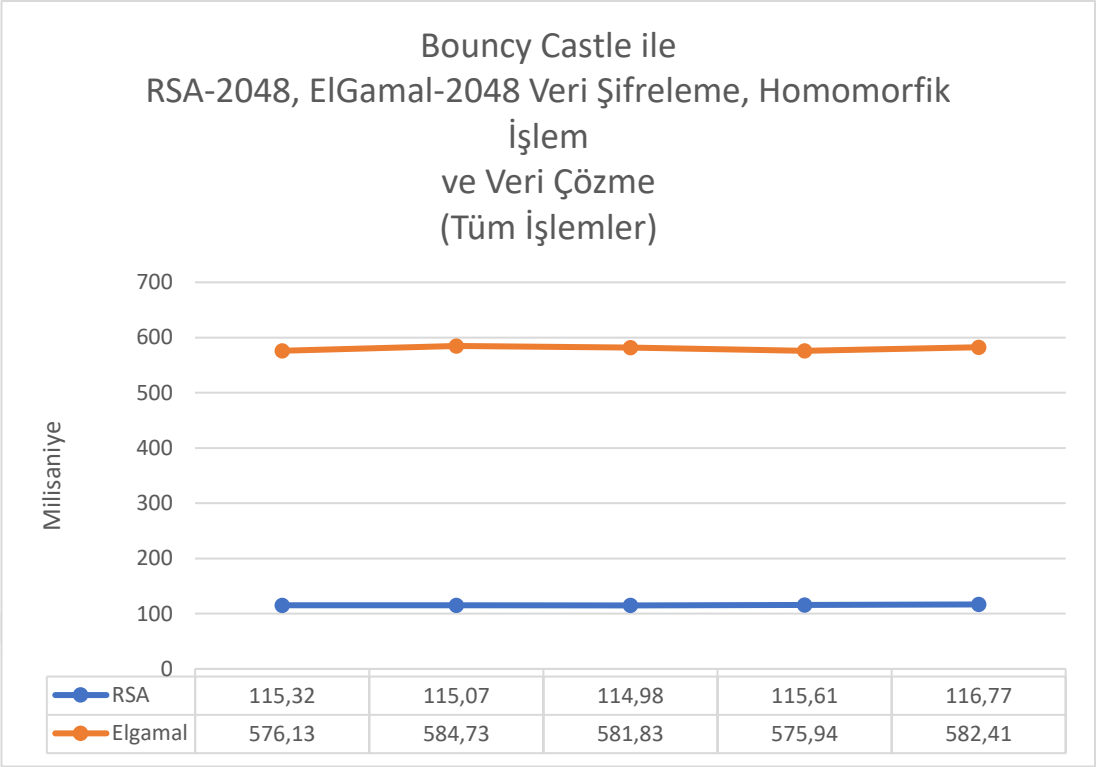
Çizelgenin, veri şifreleme, homomorfik işlem ve veri çözme ve toplam gecikme süresi bazında şekilsel gösterimleri aşağıdaki gibidir.



Şekil 4.12: Bouncy Castle ile RSA-2048, ElGamal-2048 Veri Şifreleme (Sunucu Tarafı).



Şekil 4.13: Bouncy Castle ile RSA-2048, ElGamal-2048 Homomorfik İşlem ve Veri Çözme (İstemci).



Şekil 4.14: Bouncy Castle ile RSA-2048, ElGamal-2048 Tüm İşlemler, Toplam Gecikme Süresi.

Bouncy Castle jar'ının ve 2048-bit uzunluğunda anahtarların kullanıldığı bu uygulamada, RSA' nın veri şifrelemenin ve toplamdaki üstünlüğünün yanında veri çözme ve homomorfik işlemdeki az bir farkla üstünlüğü göze çarpmaktadır. Bu da Bouncy Castle random üretilen k nın uzunluğunu $\log p$ yerine $p-1$ [20] yani daha uzun almasıdır. Burada sonuç olarak RSA da ortalama gecikme süresi toplamda ortalama 115 milisaniye olurken ElGamalda ortalama gecikme süresi 575 milisaniye olmuştur. İlk uygulamadaki ile RSA'nın gecikme süresi aynı kalırken, ElGamal'ın toplam gecikme süresi 2 katına çıkmıştır. Tabi ki bunun nedeni k nın uzunluğunun $\log p$ yerine $p-1$ seçilmesidir. 1. Uygulamada k nın uzunluğu $\log p$ yerine $p-1$ seçilirse RSA nın ElGamal'dan, 2. Uygulamada olduğu gibi veri çözmede biraz daha iyi olduğu görülür

Son olarak 3072-bit uzunluğunda anahtarlar kullanılarak her iki uygulamadan elde edilen sonuçları gözden geçirelim.

4.5. İlk Uygulamada, 3072-bit Anahtar Kullanımı ve Elde Sonuçlar

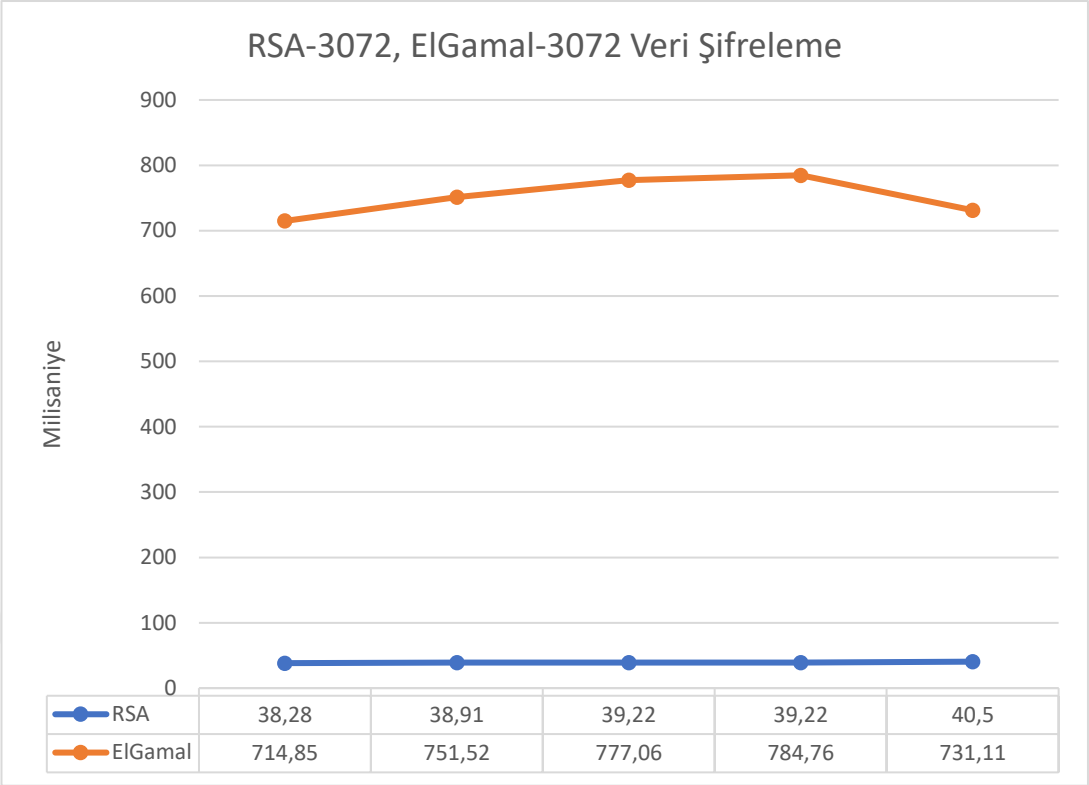
İlk uygulamada, 3072-bit uzunluğunda anahtar kullanılarak elde edilen sonuçların sırasıyla çizelge ve şekilleri aşağıdaki gibidir.

Çizelge 4.5: 1. Uygulama ile RSA-3072, ElGamal-3072 için elde edilen veriler.

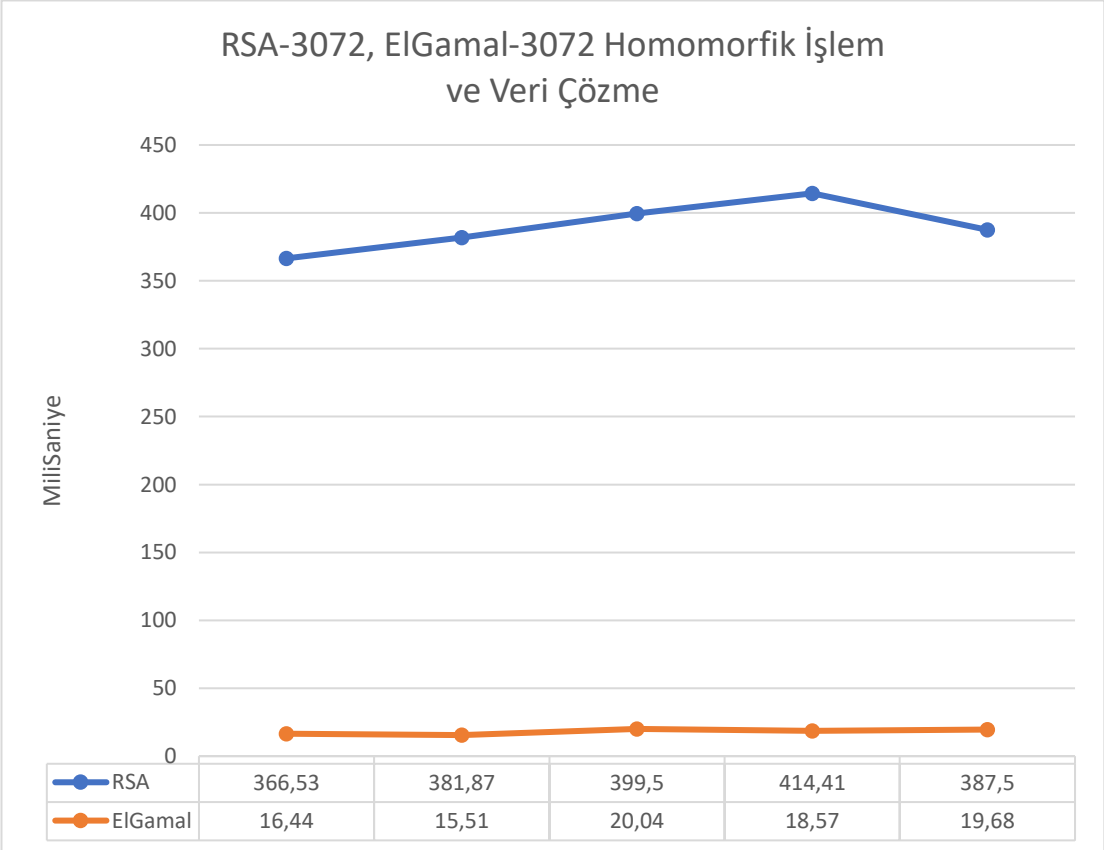
Mikro Saniye Türünden Sorgu Sonuçları		RSA-3072	ElGamal-3072
1.Sorgu	Şifreleme	3828	714854
Sonucu	Homomorfik İşlem ve Veri Çözme	366539	16443
	Toplam Süre	370367	731297
2.Sorgu	Şifreleme	3891	751526
Sonucu	Homomorfik İşlem ve Veri Çözme	381872	15512
	Toplam Süre	385763	767038
3.Sorgu	Şifreleme	3922	777063
Sonucu	Homomorfik İşlem ve Veri Çözme	399506	20042
	Toplam Süre	403428	797105
4.Sorgu	Şifreleme	3922	784769
Sonucu	Homomorfik İşlem ve Veri Çözme	414410	18574
	Toplam Süre	418332	803343
5.Sorgu	Şifreleme	4050	731114
Sonucu	Homomorfik İşlem ve Veri Çözme	387053	19688
	Toplam Süre	391103	750802

Çizelgenin veri şifreleme, homomorfik işlem ve veri çözme ve toplam gecikme sürelerinin grafikleri milisaniye (saniyenin binde biri) türünden aşağıdaki gibidir.

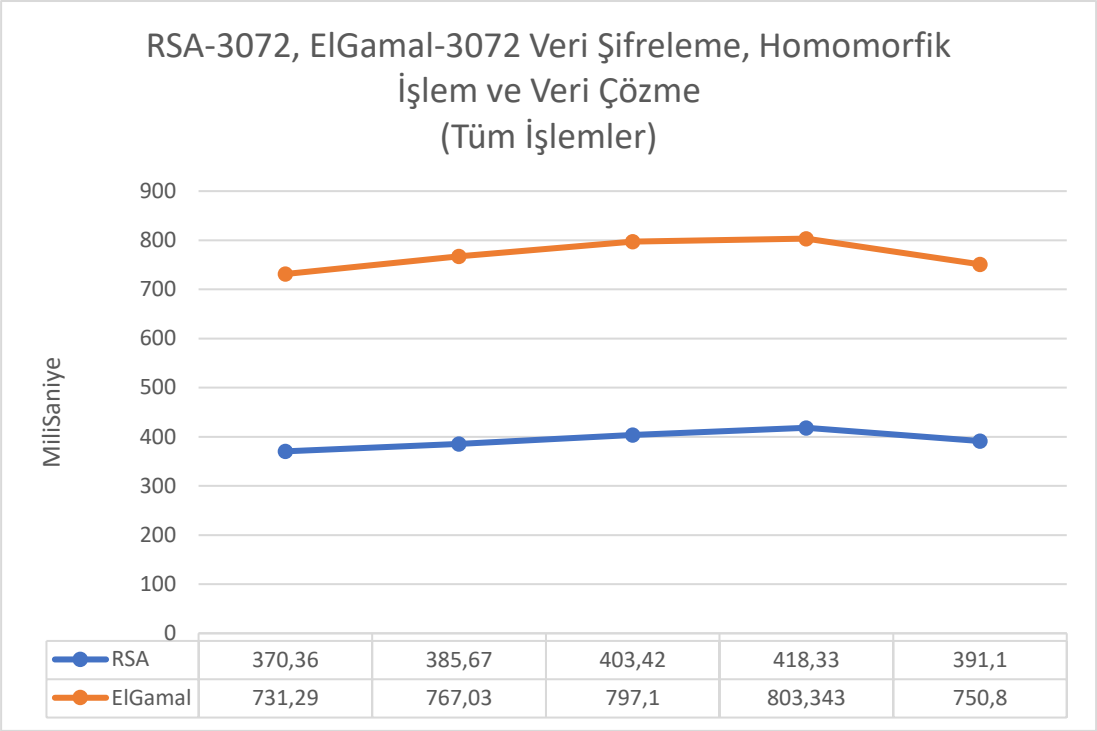
Önemli not: Veri şifreleme, veri çözme ve homomorfik işlemlerinin ayrı ayrı incelenmesinin nedeni veri şifreleme sunucu tarafında, homomorfik işlem ve veri çözme kullanıcı bilgisayarlarında (client) yapıldığı için bu şekilde bir hesaplanmıştır. Ayrıca toplam gecikme süresi de hesaplanmıştır çünkü eğer şifreleyerek verileri internette iletmenin maliyeti milisaniye türünden hesaplanarak değip değmeyeceğine karar vermektir.



Şekil 4.15: RSA-3072, ElGamal-3072 Veri Şifreleme (Sunucu Tarafı).



Şekil 4.16: RSA-3072, ElGamal-3072 Homomorfik İşlem ve Veri Çözme (İstemci).



Şekil 4.17: RSA-3072, ElGamal-3072 Tüm İşlemler, Toplam Gecikme Süresi.

Şekiller ve çizelgeler incelendiğinde; ElGamal ile verilerin şifrelenerek ve kullanıcı bilgisayarında homomorfik işlemden sonra deşifre edilmesinin maliyeti ortalama 750 milisaniye olurken RSA' da yaklaşık 400 milisaniyedir. Sonuç olarak Vergi ödeme sistemi için tercih edeceğimiz kripto sistem RSA olmadır. Güvenli anahtar uzunluğunu da NIST'in önerdiği gibi 3072-bit olmalı. Kısacası vergi ödeme sistemini RSA-3072 kullanarak yapmalıyız.

Şimdi son olarak bunu Bouncy Castle jar'ı kullanarak gerçeklersek elde edeceğimiz sonuçlara bakalım.

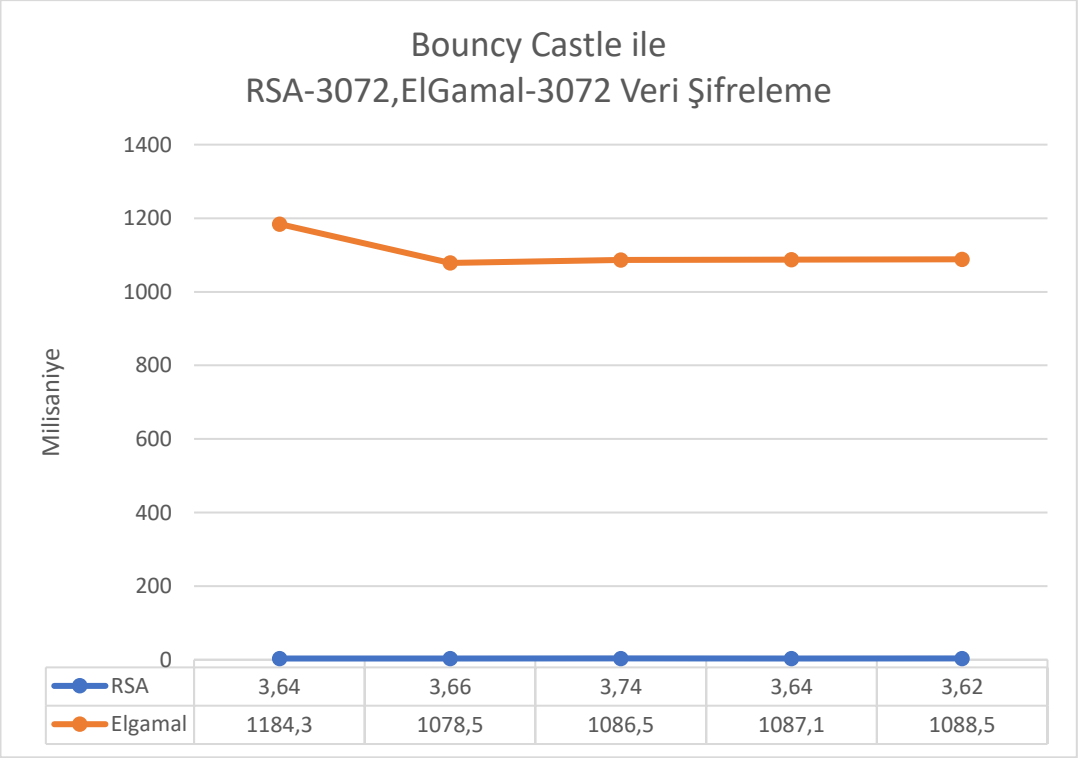
4.6. İkinci Uygulamada (Bouncy Castle), 3072-bit Anahtar Kullanımı ve Elde Sonuçlar

Bu ikinci uygulamada gizli anahtarlar değişken olduğu için gerçekte sanki örneğimizde 5 ayrı kurumun sisteme bağlanmasını gerçeklemiş oluyoruz.

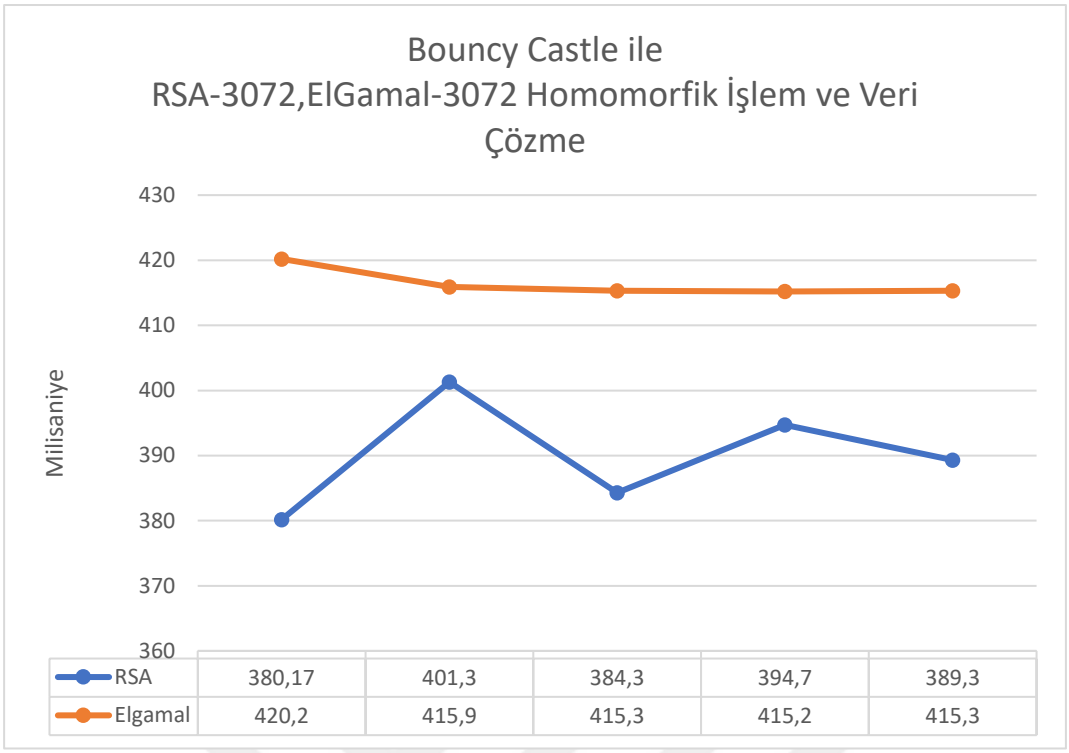
Çizelge 4.6: Bouncy Castle ile RSA-3072, ElGamal-3072 için elde edilen veriler.

Mikro saniye Türünden Sorgu Sonuçları (Bouncy Castle)		RSA-3072	ElGamal-3072
1.Sorgu Sonucu	Şifreleme	3644	1184383
	Homomorfik İşlem ve Veri Çözme	380173	420245
	Toplam Süre	383818	1604628
2.Sorgu Sonucu	Şifreleme	3661	1078519
	Homomorfik İşlem ve Veri Çözme	401392	415901
	Toplam Süre	405054	1494420
3.Sorgu Sonucu	Şifreleme	3744	1086011
	Homomorfik İşlem ve Veri Çözme	384343	415386
	Toplam Süre	388088	1501397
4.Sorgu Sonucu	Şifreleme	3643	1087150
	Homomorfik İşlem ve Veri Çözme	394717	415243
	Toplam Süre	398360	1502393
5.Sorgu Sonucu	Şifreleme	3627	1088585
	Homomorfik İşlem ve Veri Çözme	389396	415380
	Toplam Süre	393023	1503966

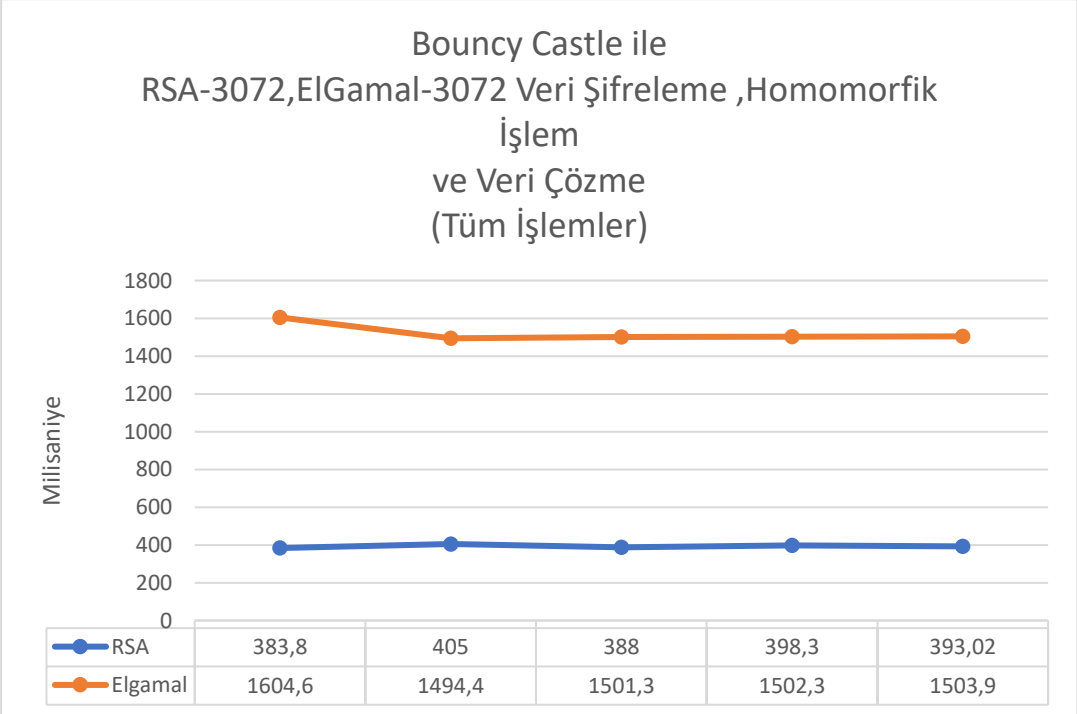
Çizelgenin grafiklerle gösterimi aşağıdaki gibidir.



Şekil 4.18: Bouncy Castle ile RSA-3072, ElGamal-3072 Veri Şifreleme (Sunucu Tarafı).



Şekil 4.19: Bouncy Castle ile RSA-3072, ElGamal-3072 Homomorfik İşlem ve Veri Çözme (İstemci).



Şekil 4.20: Bouncy Castle ile RSA-3072, ElGamal-3072 Tüm İşlemler, Toplam Gecikme Süresi.

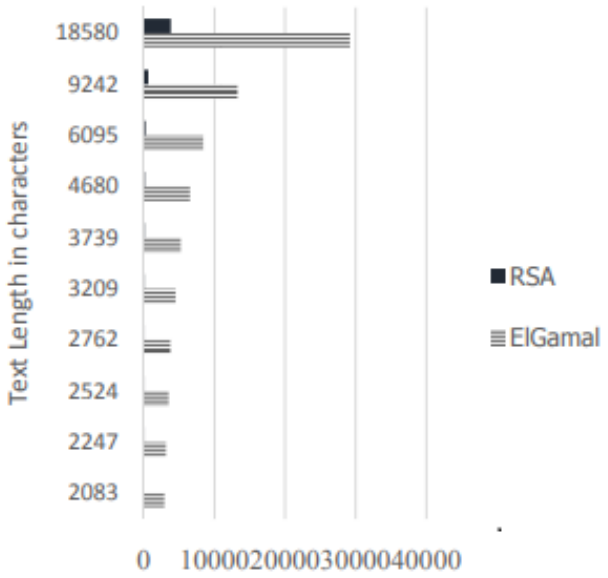
Sonuç olarak; RSA ile toplamda gecikme süresi 400 milisaniye olurken ElGamal ile RSA'nın 4 katı yaklaşık 1600 milisaniye yaklaşık 1,5 saniye ki bu veri iletimi için web uygulaması için biraz fazla görünüyor.

Her iki ayrı uygulama sonucunda vergi ödeme sisteminde kullanmamız gereken açık anahtarlı kripto sistemlerden RSA olmalıdır.

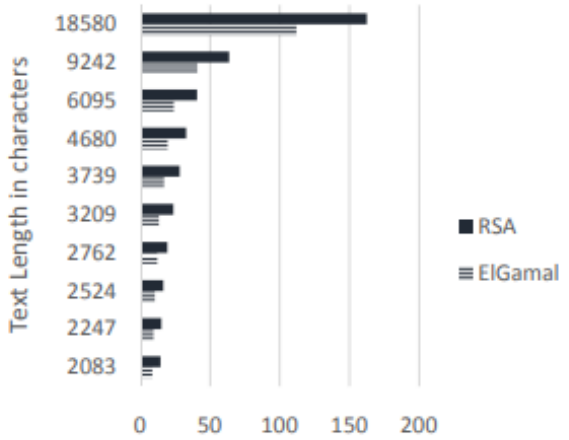
4.7. RSA, ElGamal Kripto Sistemleri ile Yapılmış Benzer Çalışmaların Sonuçları

Bu alanda yapılan çalışmalarda bu sonucu desteklemektedir. Bu çalışmalar sonucun da RSA algoritmasının şifreleme ve şifre çözme süresi ElGamal'dan daha iyidir sonucuna varılmıştır [21]. Diğer bir benzer çalışmada da genel olarak şifreleme ve deşifremele açısından RSA ile ElGamal kripto sistemleri karşılaştırılmış ve RSA kripto sistemi toplamda şifreleme ve deşifrelemenin toplam süresinde ElGamal kripto sisteminden oldukça üstün bir performansa sahip çıkmıştır [22]. Bu çalışmada RSA, şifreleme ve toplamda ElGamal'a göre oldukça performanslıdır. ElGamal ise, deşifreleme işleminde RSA'ya göre performanslıdır. Bu sonuçlar bizim çalışmamızı destekler sonuçlardır.

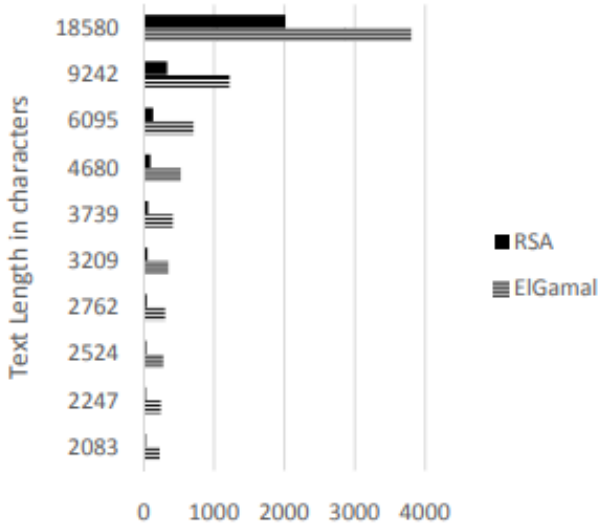
İlgili çalışmanın grafiksel sonuçları:



Şekil 4.21: RSA, ElGamal ile Veri Şifreleme ve İmzalama [22].



Şekil 4.22: RSA, ElGamal ile Veri Çözme [22].



Şekil 4.23: RSA, ElGamal ile İmza Doğrulama [22]

5. SONUÇ VE ÖNERİLER

Online Vergi Ödeme Sistemi'nde (OVÖS) istemci (client), Kamu İhale Kurumu ve Maliye Bakanlığı ile web servis aracılığıyla haberleşir. Sisteme bilgileri doğrulanarak giren bir kişinin, Kamu İhale Kurumundaki ihale sözleşme bedeli ve Maliyedeki ödemesi gereken vergi yüzdesi (tamsayı olarak kabul edildi) ilgili kişinin açık anahtarı ile şifrelendi. Internet ağı boyunca, şifreli olarak ilgili istemciye (client) iletildi. Bu şifreli metinler, istemcinin bilgisayarında iletilen şifreli sözleşme bedeli ile şifreli vergi oranı (yüzde olarak) arasında homomorfik çarpımsal işlem yapıldı. Elde edilen homomorfik çarpımsal sonuç, şifreli ödenmesi gereken vergi miktarı olup, istemcinin kapalı anahtarı ile deşifre edilirse, istemcinin ödemesi gereken vergi miktarı bulunur. Sistemimizde RSA ve ElGamal kripto sistemlerini kullandık, çünkü her ikisi de homomorfik çarpımsal özelliği sağlamaktadır. Bize de bu iki verinin, şifreli verinin çarpımı gereklidir. Değişik anahtar uzunluklarında bu iki kripto sistemin şifreleme, homomorfik işlem ve deşifreleme süreleri milisaniye türünden hesaplanmış ve bir tablo oluşturulup grafikleri çizilmiştir. Şifreleme işlemi, server (sunucu) tarafında, homomorfik işlem ve veri çözme (deşifreleme) istemci (client) tarafında olduğu için bir incelenmiştir. Ayrıca toplam olarak şifreleme, homomorfik işlem ve veri çözme (deşifreleme) de toplamda gecikme süresini milisaniye bazında görmek için incelenmiştir. NIST in bize her on yıl için önerdiği anahtar boyutlarında ele alınmıştır. 1024-bit, şimdiki on yıl için 2048-bit ve sonraki on yıl için 3072-bit uzunluğundaki anahtarlar he iki kripto sistem için seçilmiştir. Verilen iki sayının asal çarpanlarına ayrılmasının zorluğuna dayanan RSA kripto sistemi, ayrık logaritma probleminin zorluğuna dayanan ElGamal kripto sistemine göre her anahtar boyutunda oldukça performanslı olup milisaniye türünden ortalama, ElGamal'ın %25 i çıkmıştır. Sonuç olarak, bu şekilde geliştirilecek Vergi Ödeme Sistemi için kesinlikle RSA kripto sistemi kullanılmalıdır. Ayrıca anahtar boyutu da NIST in bize önerdiği anahtar boyutu seçilmelidir. Bu on yıl için, yani 2020 ye kadar önerdiği anahtar boyutu bit türünden 2048-bit olmalı, 2020 sonrası içinse 3072-bit seçilmelidir.

Sadece kurumların ihale bedeli üzerinden, ihaleyi gerçeklemeleri durumunda ödemeleri gereken vergi miktarı ele alınmıştır. Bu uygulama gerçek kişiler için de uygulanabilir. Kamuda veya özelde çalışan kişiler, maaşını Sosyal Güvenlik Kurumu (SGK) tarafından alanların maaşı ilgili kişinin açık anahtarı ile şifrelenerek aynı Kamu İhale Kurumu (KİK) da olduğu gibi yapılabilir. Aynı zamanda kişinin ödemesi gereken vergi oranı Maliye Bakanlığından şifrelenerek, istemcinin bilgisayarında homomorfik işlem yapılarak, ödenmesi gereken vergi miktarı şifreli olarak hesaplanır. Sonrasında bulunan şifreli vergi miktarı, kişinin özel anahtarı ile deşifre edilerek, yalnızca kendi tarafından görülmüş olur. Uygulamamızda, kurumlar için ödemesi gereken vergi miktarı, KİK ve Maliye Bakanlığı kurumlarından veriler web servis aracılığı ile şifreli olarak alınıyordu. Önerdiğimiz sistemde de kurumlar yerine gerçek kişilerin, maaşları üzerinden ödemesi gereken vergi miktarı, SGK ve Maliye Bakanlığından web servis aracılığı ile alınan verilerin ilgili kişilerin açık anahtarları kullanılarak şifrelenen verilerin, istemcinin bilgisayarında homomorfik çarpımsal işlem ve sonrasında ilgili kişinin (istemcinin) gizli anahtarı ile deşifrelenerek ödemesi gereken vergi miktarı, güvenli biçimde görüntülenmiş olur. Sonuç kısmında da belirtildiği gibi kullanılması gereken kripto sistemin RSA ve anahtar boyutunun 2048-bit veya 3072-bit olması gereklidir.

Gelecekte, birçok sisteme, özellikle bankacılık ve finans sektöründe, homomorfik şifrelemeler kullanılarak birçok sistem için verilerin güvenliğini sağlayan yazılımlar geliştirmek istiyorum. Ayrıca tam homomorfik şifrelemeler ve uygulamaları üzerine çalışmak istiyorum.

KAYNAKLAR

[1] **W. Diffie and M. Hellman** (1976). New directions in cryptography. IEEE Transactions on Information Theory. (s. 644-654)

[2] **R. L. Rivest, A. Shamir and L. Adleman** (1978). A method for obtaining digital signatures and public-key cryptosystems. (s. 120-126)

[3] **T. ElGamal** (1984). A public key cryptosystem and a signature scheme based on discrete logarithms. In Proceedings of CRYPTO 84 on Advances in cryptology, (s.10-18) Springer-Verlag New York.

[4] **P. Paillier** (1999). Public-key cryptosystems based on composite degree residuosity classes. Advances in Cryptology Eurocrypt. (s. 223-238)

[5] **S. Goldwasser and S. Micali** (1984). Probabilistic encryption. Journal of Computer and System Sciences. (s.270-297)

[6] **J. Feigenbaum and M. Merritt** (1991). DIMACS Series in Discrete Mathematics and Theoretical Computer Science, volume 2, chapter Open Questions, Talk Abstracts, Summary of Discussions. (s. 1-40)

[7] **C. Gentry** (2009). Fully homomorphic encryption using ideal lattices. In Proceedings of the 41st annual ACM symposium on Theory of computing. (s.169-178).

[8] **Erkan Afacan** (2016). Kriptografiye Giriş, şifreleme teorisi, epos yayınları. (s. 67-77)

[9] **Ersan Akyıldız**, Uygulamalı Matematik Enstitüsü UYGULAMALI MATEMATiK ENSTiTSÜ Kriptografi Bölümü ODTÜ. Alındığı tarih:27.04.2019, adres: https://iam.metu.edu.tr/system/files/iamData/LectureNotes/kriptolojiye_giris_ders_notlari.pdf

[10] **Eşref Adalı** (2016). Bilgisayar ve Bilgi Güvenliği ve Yönetimi, Ulusal Yazılım ve Sertifikasyon Merkezi. (s.112-152)

[11] **D. Boneh** (1999). Twenty years of attacks on the RSA cryptosystem. (s.203–213)

[12] **Ersan Akyıldız, Canan Çimen, Sedat Akleylek** (2014). Şifrelerin Matematiği: KRİPTOGRAFİ, ODTÜ Yayıncılık. (s.67-100)

[13] **D. Coppersmith** (1997). Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. J. Cryptol. (s.233–260)

[14] **Christof Paar, Jan Pelzl** (2010). Understanding Cryptography, Springer. (s.226-246)

[15] **Hüseyin Bodur** (2016). Java Diliyle Kriptoloji Uygulamaları, abaküs. (s.155-199)

[16] **Esra Çalık, Hüseyin Aşkın Erdem, M. Ali Aydın** (2014). Bulut Bilişim Güvenliği için Homomorfik Şifreleme

[17] **N. Smart, F. Vercauteren** (2010). Fully homomorphic encryption with relatively small key and ciphertext sizes, in Proceedings of PKC. (s. 420–443)

[18] **C. Gentry, S. Halevi** (2011). Implementing Gentry fully-homomorphic encryption scheme, in Proceedings of Advances in Cryptology. (s. 129–148)

[19] **C. Gentry** (2009). Fully Homomorphic Encryption Using Ideal Lattices. PhD thesis,

[20] **J.Menezes** (1996). HANDBOOK of APPLIED CRYPTOGRAPHY, crc press.

[21] **Andysah Putera, Utama Siahaan** (2018). Comparative Analysis of RSA and ElGamal Cryptographic Public-key Algorithms

[22] **A. E. Okeyinka** (2015). Computational Speeds Analysis of RSA and ElGamal Algorithms on Text Data

[23] Moore Yasası (Moores' Law) Wikipedia. Alındığı tarih:25.04.2019, adres: https://tr.wikipedia.org/wiki/Moore_yasası

[24] **Cihangir Tezcan** (2015). Kriptografi ve Siber Güvenlik-ODTÜ Seminer.

[25] **Xun Yi, Russell Paulet, Elisa Bertino** (2014). Homomorphic Encryption and Applications, Springer, (s. 15)

ÖZGEÇMİŞ

Ad-Soyad : Hasan ÇETİNKAYA

Doğum Tarihi ve Yeri : 12/05/1974, Muğla

E-posta : hasen.cetinkaya@gmail.com

ÖĞRENİM DURUMU:

- **Lisans** : ODTÜ- Matematik, 1998
  Atılım Üniversitesi-Yazılım Mühendisliği, 2014
  Atılım Üniversitesi- Bilgisayar Mühendisliği, 2015
- **Yüksek Lisans** : Bilgi Güvenliği Mühendisliği ve Kriptografi, 2019

MESLEKİ DENEYİM VE ÖDÜLLER:

- 1998-2001 yılları arasında Erciyes Üniversitesi'nde araştırma görevlisi olarak çalıştım.
- 2009-2011 yılları arasında Maliye Bakanlığı'nda yazılım mühendisi olarak çalıştım.
- 2011 yılından itibaren Adalet Bakanlığı'nda yazılım mühendisi olarak çalışmaya devam etmekteyim.