Elliptic Curve Cryptography

Elif Karataş
Computer Engineering
Dokuz Eylul University
İzmir
elif.karatas@ceng.deu.edu.tr

Berk Adalı
Computer Engineering
Dokuz Eylul University
Izmir
berk.adali@ceng.deu.edu.tr

***Abstract*—Project on the operation of Elliptic-Curve Cryptography, its usage areas and its use with El-Gamal Encryption Decryption.**

***Keywords*—asymmetric cryptography, elliptic-curve cryptography, prime numbers, curve, el-gamal, encryption, decryption**

I. Introduction

Elliptic curve cryptography, one of the asymmetric cryptographies is also known as public

key cryptography.

On finite fields, elliptic curve cryptography is a plane curve that satisfies the equation y** = x*** + a*x + b. The curve consists of points.

Asymmetric cryptography uses key pairs(public and private key), unlike symmetric

cryptography. Symmetric cryptography uses a private key. In asymmetric cryptography, each communicating party has a key pair.

The public key is used for the sender to encrypt the information, while the private key is used for the receiver to decrypt it, in public key cryptography.

In asymmetric cryptography, there are mathematically linked key pairs. Because of that, key lengths are longer than symmetric cryptography. RSA can be given as an example of this but in elliptic curve cryptography, key sizes are less, although equivalent encryption strength is provided. In ECC, 256-bit key size can be realized with 3072-bit key size in RSA.

Although symmetric cryptography is used in terms of performance, asymmetric cryptography is more suitable for security. The closest or even better to symmetric cryptography in performance is elliptical curve cryptography, which is asymmetric cryptography. When evaluated in terms of both security and performance, with the new technology, elliptic curve cryptography has started to be used commonly. Cryptocurrency and blockchain uses are examples of this technology. It is also advantageous for low-powermobile devices and IoT devices that are widely used and where speed is important. Elliptic curve cryptography also consumes less memory. Therefore, smart card systems can also be seen in its use.

Large technology firms such as Motorola, IBM, Sun Microsystems, Microsoft and Hewlett-Packard are also investing in crypto systems that use elliptic curve cryptography.

II. Related Works

A. *Symmetric and Asymmetric Cryptography*

There are two kinds of cryptography: Symmetric key and public key (asymmetric).

Public key algorithms are:

- Diffie-Hellman (DH), DSA, RSA, Elliptic Curves (ECC)

Public key features are:

- Two keys, public and private
- Slower then symmetric key crypto

Public key uses are:

- digital signatures
- key Exchange
- pseudo random numbers

Elliptic-curve cryptography can us efor all of these above. Differently, slower keys than the other public keys crypto.

The difficulty in elliptic curve cryptography is reliesed on the Diffie-Hellman Problem (DLP).

Generally, If DLP adapts ECC, We gets:

- Given points P and nP on an elliptic curve
- Easy to find nP given n and P
- Hard to find n given P and nP
- Public key: nP, Private key: n [1].

B. *Elliptic-Curve Cryptography*

Elliptic curves satisfy the real equation y^2**= x*** + a*x + b and exist a different curve for each value of a, b. They are symmetrical curves with respect to the X axis.

The most basic function of these groups is addition.

Below you can see two examples of summation in elliptic curves.

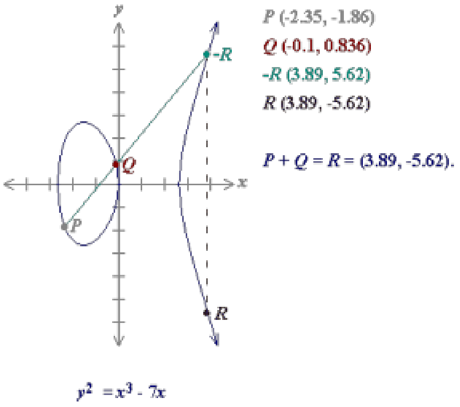


Fig. 1. Addition in Elliptic-Curve

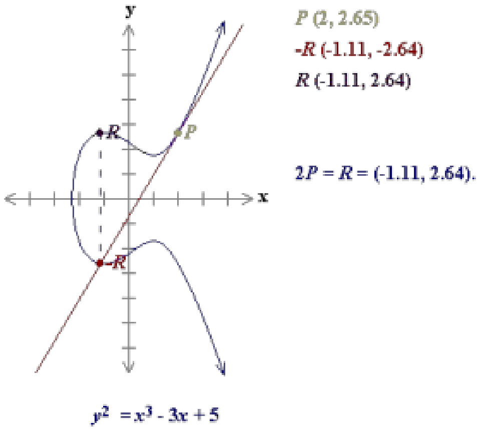


Fig. 2. Multiplication in Elliptic-Curve

In the second curve, unlike the first curve, it is seen that the line passes through one point. This situation creates uncertainty. The angle of the line is uncertain. The solution to this situation is to take the tangent value at that point [2].

C. Discrete Logarithm Problem

Elliptic-curve cryptography is an encryption technique built on the Elliptic curve discrete logarithm problem. Cryptanalysis on ECC can be realized by analyzing the solution of this problem. The most important of these studies are Pollard's Rho algorithm, Gaudry – Hess – Smart - Attack, and Weil Descent. General and the most known algorithm to solve the EC's discrete logarithm problem is Pollard's rho algorithm [3].

D. Program with ECC

A simple program has been developed to test elliptic curve encoding. In general, according to the program, the operations performed in ECC are as follows:

- Curve parameters (getParams (); a, b, prime) and Message parameters and operations (plainTextProc ();) are entered.
- Then, the curve point operations (findDot (); findNumDots (); messagePointMatch ();) are performed.
- Key operations (receiverPrvKey (); recievertPkFind (); senderPrvKey (); senderPvFind ();) are then performed [4].
- Finally, encryption and decryption operations are done.

You can see the UML diagram below:

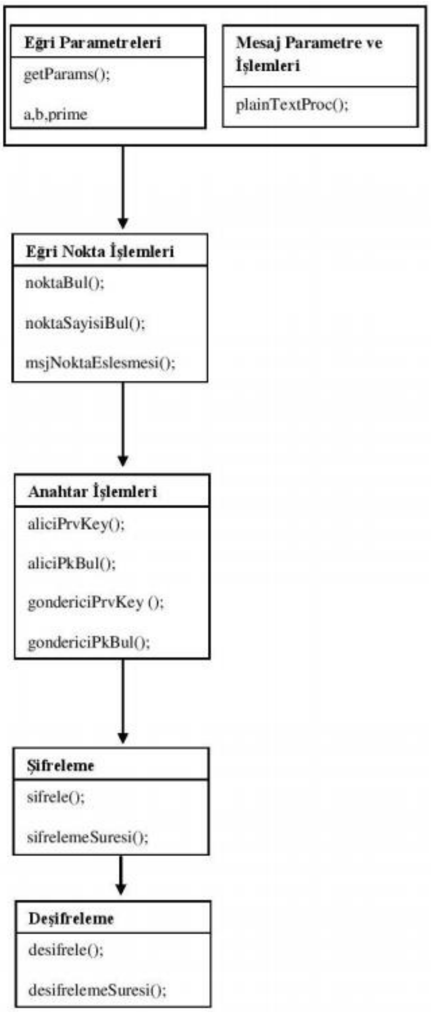


Fig. 3. UML Diagram

E. RSA and ECC

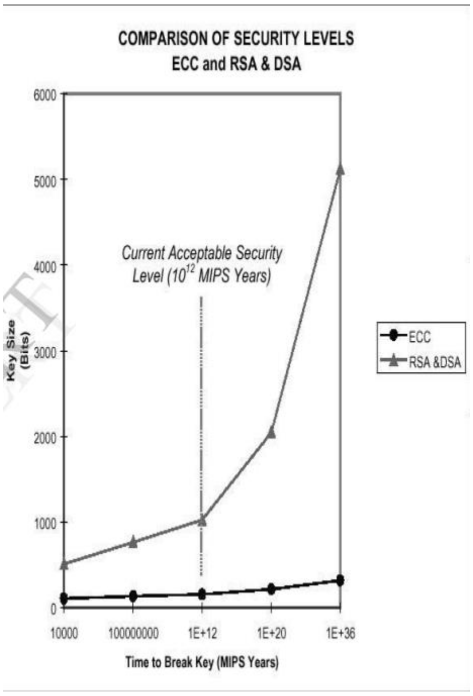


Fig. 4. RSA and ECC

A can be seen from the graph above, it is much more difficult to break an ECC with a key of the same size than an RSA & DSA with a key of the same size. For example, 109-bit encryption key was broken with 10,000 computers running all day for 549 days in 2002. Many people thinks that ECDLP is harder than the RSA & DSA. The security ECC depends on how difficult it becomes to find k when kP and P are given [5].

F. *El-Gamal Digital Signature Application*

El-Gamal is a digital signing algorithm based on the discrete logarithms. It consists of three stages. These:
Signing Protocol:

- P prime number is selected.
- A value g that satisfies g € Zp is chosen by the receiver.
- The value y = g**x % p is calculated.
- The public key is (p,gy) and the private key x is kept.

Key Generation:

- The number k is chosen such that gcd (k, p − 1) = 1.
- The value of r≡g**k (% p) is computed.
- The operation s≡ (m-xr) k ** +(-1) (% p +(-1)) is calculated.
- If there is s = 0 equality, it is returned to the beginning. The (r, s) pair is a digital signature.

Verification:

- The signature produced must be accepted by the person verifying it.
- The signature g**h(m) ≡ y**r x r**s (mod p) in the form is verified.
- According to v1 ≡ (p**r) x (r**s)≡ g**m≡ v2, if the value of v1≡v2 is provided, the signature is accepted and correct [6].

The above steps are applied to Elliptic Curve Cryptography to obtain public and private keys.

G. *ECC on Smart Cards*

Elliptic-curve cryptography is used in many digital signature areas and digital certificates. One of these is smart card systems. It is aimed to create a key pair on the smart card system. The created key is sent openly to the system that is logged in. The information on the smart card is read and paired in the system. If the matches are correct, a certificate is generated. The created certificate is recorded both on the smart card and in the system. The generated keys are generated according to elliptic curve cryptography. The aim is to prevent the person's information from being seized by harmful resources while sending to the system [7].

H. *ECC in Blockchain*

The main purpose of elliptic curve cryptography is to securely store data sent or received using large prime numbers. Cryptographies are used to prevent malicious software from getting the information while transmitting and receiving transactions on the blockchain. The important thing is that the hash value on the blockchain is generated as a secret key. Elliptic curve algorithm is used in this section [8].

İ. *Asymmetric Authentication Protocol for Mobile Devices Using EC*

Today, the most personal exchange of people is the internet environment. In this way, the number of e-commerce sites is increasing day by day. It is very important that information is kept securely when shopping on mobile devices. Many attacks are carried out on mobile devices. Elliptic curve cryptology is used against these attacks. The information in the authentication part of e-commerce sites is produced asymmetrically both openly and confidentially and the necessary protocols are applied [9].

J. *Elliptic Curve based Signature Method to Control Fake Paper based Certificates*

Certificates are used in many areas of our lives. They are used in job applications, university documents, commercial agreements and many other places. With the transfer of these certificates to the internet environment, attacks on certificates have increased. Elliptic curve cryptography is used for the security of certificates. Discrete algorithm methods are used for the control of fake certificates. In this way, it is aimed to bring the calculation time to the most ideal level [10].

III. Elliptic-Curve Cryptography with El-Gamal Encryption / Decryption

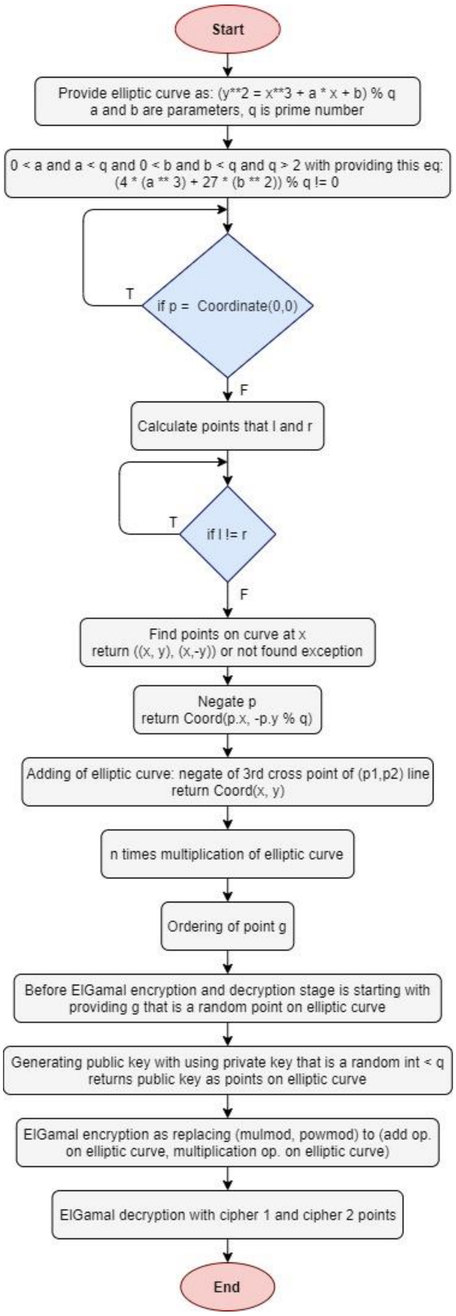

Start
Provide elliptic curve as: (y**2 = x**3 + a * x + b) % q
a and b are parameters, q is prime number
0 < a and a < q and 0 < b and b < q and q > 2 with providing this eq:
(4 * (a ** 3) + 27 * (b ** 2)) % q != 0
T
if p = Coordinate(0,0)
F
Calculate points that l and r
T
if l != r
F
Find points on curve at x
return ((x, y), (x,-y)) or not found exception
Negate p
return Coord(p.x, -p.y % q)
Adding of elliptic curve: negate of 3rd cross point of (p1,p2) line
return Coord(x, y)
n times multiplication of elliptic curve
Ordering of point g
Before ElGamal encryption and decryption stage is starting with
providing g that is a random point on elliptic curve
Generating public key with using private key that is a random int < q
returns public key as points on elliptic curve
ElGamal encryption as replacing (mulmod, powmod) to (add op.
on elliptic curve, multiplication op. on elliptic curve)
ElGamal decryption with cipher 1 and cipher 2 points
End

Fig. 5. Flowchart of the project

El-Gamal is an asymmetric encryption algorithm. It is based on the Diffie-Hellman method. It consists of three stages. These are:

Key Generation:

- P prime number is selected.
- A value g that satisfies g € Zp is chosen by the receiver.
- The value h = g ^ x is calculated.
- The public key h is released. The private key x is kept. Encryption:
- A public key is required to encrypt plain text. It is encrypted with the public key.
- The value c2 = m * h ^ y (modp) is calculated by the sender.
- (c1, c2) cipher text is sent to the receiver.

Decryption:

- The private key is used to decryption c1 and c2 values.
- Plain text is reached by performing m = c2 / c1 ^ x (modp) operation by the receiver [6].

The above steps are applied to Elliptic Curve Cryptography to obtain public and private keys.

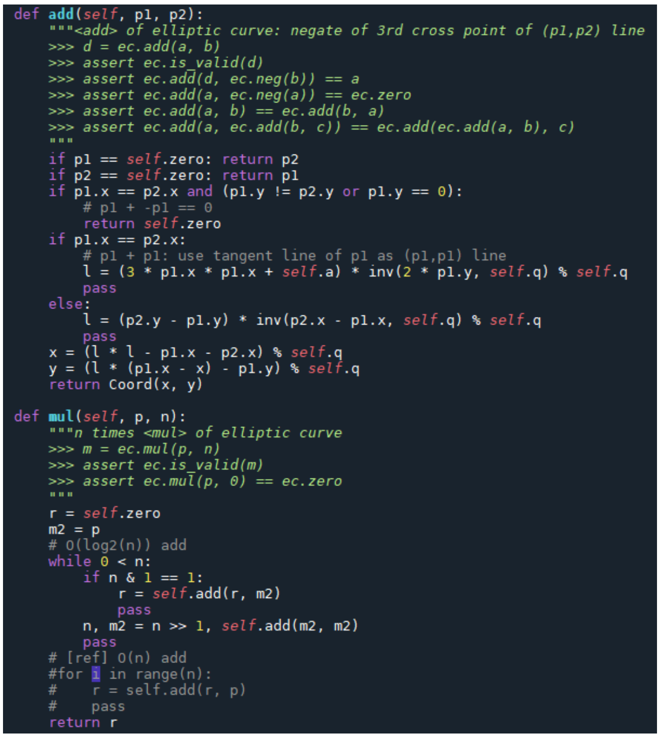
IV. Test Results and Discussion

In this program, we produced the public key to be used in El-Gamal Encrption and Decryption with ECC.

When program is started, these values generated randomly:

- a and b parameter to be used to create EC
- private key
- x value on EC to generate g point to be used in El-Gamal encryption
- r value to be used in multiplication on EC and El-Gamal encryption.

Here is addition and multiplication operations on EC.

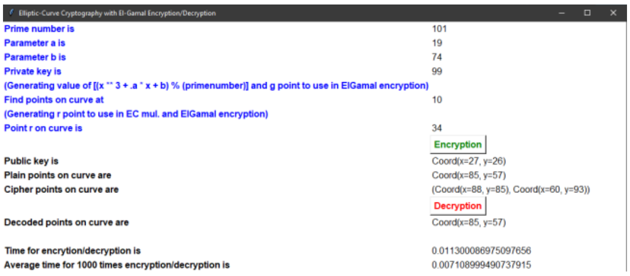


```
def add(self, p1, p2):
    """<add> of elliptic curve: negate of 3rd cross point of (p1,p2) line
    >>> d = ec.add(a, b)
    >>> assert ec.is_valid(d)
    >>> assert ec.add(d, ec.neg(b)) == a
    >>> assert ec.add(a, ec.neg(a)) == ec.zero
    >>> assert ec.add(a, b) == ec.add(b, a)
    >>> assert ec.add(a, ec.add(b, c)) == ec.add(ec.add(a, b), c)
    """
    if p1 == self.zero: return p2
    if p2 == self.zero: return p1
    if p1.x == p2.x and (p1.y != p2.y or p1.y == 0):
        # p1 + -p1 == 0
        return self.zero
    if p1.x == p2.x:
        # p1 + p1: use tangent line of p1 as (p1,p1) line
        l = (3 * p1.x * p1.x + self.a) * inv(2 * p1.y, self.q) % self.q
        pass
    else:
        l = (p2.y - p1.y) * inv(p2.x - p1.x, self.q) % self.q
        pass
    x = (l * l - p1.x - p2.x) % self.q
    y = (l * (p1.x - x) - p1.y) % self.q
    return Coord(x, y)

def mul(self, p, n):
    """n times <mul> of elliptic curve
    >>> m = ec.mul(p, n)
    >>> assert ec.is_valid(m)
    >>> assert ec.mul(p, 0) == ec.zero
    """
    r = self.zero
    m2 = p
    # O(log2(n)) add
    while 0 < n:
        if n & 1 == 1:
            r = self.add(r, m2)
            pass
        n, m2 = n >> 1, self.add(m2, m2)
        pass
    # [ref] O(n) add
    #for i in range(n):
    #    r = self.add(r, p)
    #    pass
    return r
```

Fig. 6. EC addition and multiplication.

The program is run until the random numbers meet the conditions specified in the algorithm. The program starts when suitable values are found.

To calculate the encryption and decryption time of the program, the program is run 1000 times and the average time is taken.



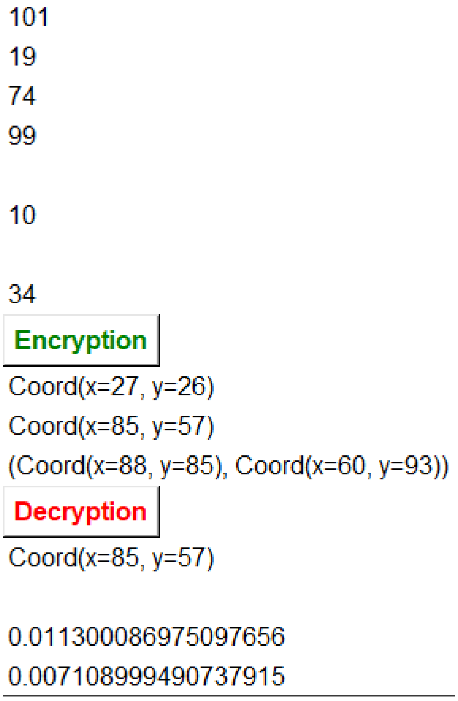
```
Elliptic-Curve Cryptography with El-Gamal Encryption/Decryption
Prime number is                                                                          101
Parameter a is                                                                           19
Parameter b is                                                                           74
Private key is                                                                           99
(Generating value of [(x ** 3 + .a * x + b) % (primenumber)] and g point to use in ElGamal encryption)
Find points on curve at                                                                  10
(Generating r point to use in EC mul. and ElGamal encryption)
Point r on curve is                                                                      34
                                                                                         Encryption
Public key is                                                                            Coord(x=27, y=26)
Plain points on curve are                                                                Coord(x=85, y=57)
Cipher points on curve are                                                               (Coord(x=88, y=85), Coord(x=60, y=93))
                                                                                         Decryption
Decoded points on curve are                                                              Coord(x=85, y=57)

Time for encrytion/decryption is                                                         0.011300086975097656
Average time for 1000 times encryption/decryption is                                     0.007108999490737915
```

Fig. 7. GUI of the program.

Where it is printed in blue, respectively:

- Prime number
- parameter a
- parameter b
- private key
- x value
- r value

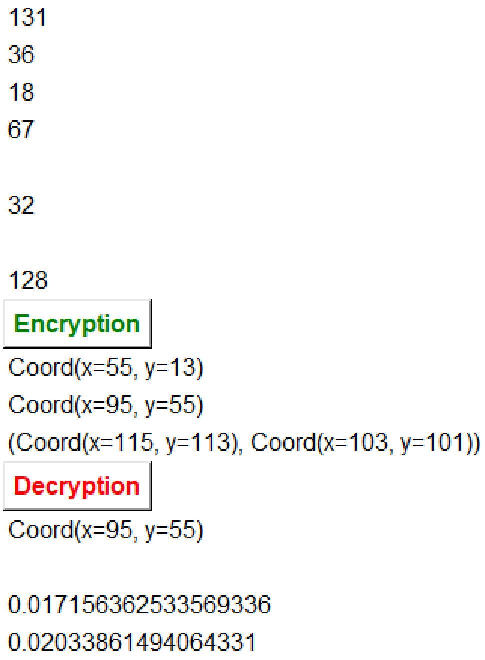
are produced.



```
101
19
74
99

10

34
Encryption
Coord(x=27, y=26)
Coord(x=85, y=57)
(Coord(x=88, y=85), Coord(x=60, y=93))
Decryption
Coord(x=85, y=57)

0.011300086975097656
0.007108999490737915
```

Fig. 8. Values and results.

When the program was run with the above random values, public key, plain, and cipher coordinates were created. Then plain points were obtained by making decryption. The time taken for a single encryption and decryption with random values generated as above is 0.0114. When it is run 1000 times, the average time is 0.0072.



```
131
36
18
67

32

128
Encryption
Coord(x=55, y=13)
Coord(x=95, y=55)
(Coord(x=115, y=113), Coord(x=103, y=101))
Decryption
Coord(x=95, y=55)

0.017156362533569336
0.02033861494064331
```

Fig. 9. Values and results.

With the above generated values, encryption and decryption time has also changed, but there is not much difference.

The time taken for a single encryption and decryption with random values generated as above is 0.0172. When it is run 1000 times, the average time is 0.0204

As the prime number increases, the elapsed time increases and encryption is slower than decryption because of encrypiton has many arguments while decryption has single entity.

V. CONCLUSION

As a result, small sized devices are becoming widespread with the development of technology. Therefore, there is a need for fast encryption algorithms on small sized devices. ECC is one of them. The biggest advantage of providing high security with low key size compared to other encryption algorithms. Another advantage is that it can be used in many different areas.

In the future, with the development of technology, many more different encryption algorithms will be produced. No encryption algorithm is unbreakable, but ensuring maximum security is essential.

REFERENCES

[1] OWASP Portland Maine Local Chapter Meetup | OWASP Foundation. (2021). Retrieved 20 January 2021, from https://owasp.org/www-chapter-portland-me/

[2] Şadi Evren ŞEKER, Eliptik Eğriler, 2009

[3] Tarık YERLİKAYA, Ercan BULUŞ, Derya ARDA, Eliptik Eğri Şifreleme Algoritmasını Kullanan Dijital İmza Uygulaması

[4] Mühendislik Dergisi, Aziz Mahmut YÜCELEN, Abdullah BAYKAL, Cengiz COŞKUN, Kriptolojide eliptik eğri algoritmasının uygulanması, 2017

[5] P.K. Sahoo, Dr. R. K. Chhotray, Dr. Gunamani Jena, Dr. S. Pattnaik, International Journal of Engineering Research & Technology (IJERT), An Implementation Of Elliptic Curve Cryptography, 2013

[6] - Siber Vatan. (2021). Retrieved 20 January 2021, from https://www.sibervatan.org/makale/elgamal-sifreleme/40

[7] ELİPTİK EĞRİ TABANLI KRİPTOGRAFİK PROTOKOL ve AKILLI KART ÜZERİNDE BİR UYGULAMA - (2021). Retrieved 20 January 2021, from https://docplayer.biz.tr/4303498-Eliptik-egri-tabanli-kriptografik-protokol-ve-akilli-kart-uzerinde-bir-uygulama.html

[8] Burak Selim Şenyurt | Blockchain Eliptik Eğri Şifreleme Algoritması. (2021). Retrieved 20 January 2021, from https://www.buraksenyurt.com/post/Blockchain-Eliptik-Egri-Sifreleme-Algoritmas%C4%B1

[9] S. Prasanna Ganesan, "An asymmetric authentication protocol for mobile devices using elliptic curve cryptography," *2010 2nd International Conference on Advanced Computer Control*, Shenyang, 2010, pp. 107-109, doi: 10.1109/ICACC.2010.5486928.

[10] SGK Murthy, MV Ramana Murthy, A Chandrasekhara Sarma, “Elliptic Curve based Signature Method to Control Fake Paper based Certificates”, *2011*.