Penggunaan *QR code* Berbasis Kriptografi Menggunakan Algoritma *Elliptic Curve Criptography*

Julieta Adhellia Pratiwi[1], Asmunin[2]
[1,2]Jurusan Teknik Informatika Fakultas Teknik Universitas Negeri Surabaya
[1]julieta.18011@mhs.unesa.ac.id
[2]asmunin@unesa.ac.id

Abstrak – Media digital dan internet banyak dimanfaatkan semua orang untuk mengirim informasi pada saat pandemi seperti ini Media digital menjadi sarana menyebarkan informasi dengan bebas. Beberapa informasi berupa URL (*Unifrom Resource Locator*) yang dibagikan. Namun dengan membagikan URL tersebut ke orang yang kita kehendaki sangat tidak aman saat diperjalanan. Sehingga perlu adanya keamanan. Maka kita perlu keamanan yaitu kriptografi. Ada dua jenis algoritma kriptografi yaitu simetris menggunakan dua kunci dan asimetris menggunakan satu kunci. Algoritma yang digunakan yaitu ECC (*Elliptic Curve Cryptography*). Algoritma ECC Merupakan Enkripsi Asimetris. Enkripsi Asimetris Adalah dua kunci enkripsi berbeda yang secara matematis terkait satu sama lain. Salah satu kunci ini dikenal sebagai *public key* dan *private key*. Keuntungan pertama dari jenis enkripsi ini adalah keamanan yang diberikannya. Dalam metode ini, *public key* yang tersedia untuk umum untuk mengenkripsi data, sedangkan dekripsi data dilakukan menggunakan *private key* yang harus disimpan dengan aman. Ini memastikan bahwa data tetap terlindungi dari serangan *man-in-the-middle* (MiTM). Kemudian hasil dari *ciphertext* tersebut kita generate menjadi *QR code (Quick Response)*. Cara memindai *QR code* dapat dilakukan dengan mudah. Caranya dengan scan *QR code* tersebut ke aplikasi kemudian diproses menjadi tautan halaman atau dokumen berupa *ciphertext*. Algoritma ECC memiliki kinerja waktu yang berbeda dengan masukan karakter yang panjangnya yang tidak sama.

Kata Kunci – Kriptografi, Enkripsi, Dekripsi, Algoritma ECC, *QR code*

I. PENDAHULUAN

Mulai berkembang sangat pesatnya media digital sekarang. Media digital dan internet banyak dimanfaatkan semua orang untuk mengirim informasi pada saat pandemi seperti ini. Media digital menjadi sarana menyebarkan informasi dengan bebas. Beberapa informasi berupa URL yang dibagikan. URL adalah sebuah alamat yang mengarah ke sebuah file pada website atau internet[1]. URL berfungsi untuk mengetahui mekanisme pengambilan dan yang tersedia pada dokumen, gambar, maupun file lainnya dan untuk menunjukkan alamat suatu situs. URL juga merupakan sumber daya akses alamat jarak jauh. URL disediakan online oleh Internet.

Namun dengan membagikan URL tersebut ke orang yang kita kehendaki sangat tidak aman saat diperjalanan. Sehingga perlu adanya keamanan. Maka kita perlu keamanan yaitu kriptografi. Kriptografi adalah suatu ilmu untuk mempelajari aspek-aspek keamanan informasi yaitu kerahasiaan, integritas data dan otentikasi menggunakan teknik-teknik matematika[2]. Ada dua jenis algoritma kriptografi yaitu simetris menggunakan dua kunci dan asimetris menggunakan satu kunci.

Algoritma yang digunakan yaitu ECC. ECC merupakan algoritma asimetris yang menggunakan public key dan private key. Metode kriptografi ini menggunakan kurva ellips dengan koefisiennya terbatas dan semua variabel pada elemen dari suatu Galois Field[3]. Pada standar efisiensi kriptografi kurva elips merupakan acuan Kurva ellips . ECC ini ditemukan oleh Victor Miller dan Neil Koblitz pada tahun 1985 sebagai implementasi kriptografi asimetris untuk mekanisme alternatif. Pada algoritma asimetris enkripsi lebih sederhana dalam manajemen kunci, namun hanya dapat digunakan pada data berukuran kecil untuk pengiriman data melalui internet.

ECC memiliki kelebihan memiliki keamanan yang sama dengan algoritma yang lain pada distribusi kunci akan tetapi menajemen kunci lebih baik karena ukuran kunci yang lebih kecil jika dibandingkan dengan algoritma yang lain dan usia dari algoritma ECC ini lebih baru[4]. Algoritma ECC Merupakan Enkripsi Asimetris. Enkripsi Asimetris Adalah dua kunci enkripsi berbeda yang secara matematis terkait satu sama lain. Salah satu kunci ini dikenal sebagai public key dan private key.

Keuntungan pertama dari jenis enkripsi ini adalah keamanan yang diberikannya. Dalam metode ini, *public key* yang tersedia untuk umum untuk mengenkripsi data (*encryption)*, sedangkan dekripsi (*decryption*) data dilakukan menggunakan *private key*; yang perlu disimpan dengan aman. Ini memastikan bahwa data tetap terlindungi dari serangan MiTM. Kemudian hasil dari *ciphertext* tersebut kita generate menjadi *QR code* .

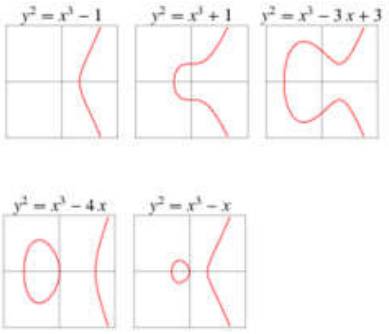
QR code adalah penyimpan informasi yang mampu menyimpan ribuan karakter alfanumerik dari suatu kode matrik dua dimensi[5]. Cara memindai *QR code* dapat dilakukan dengan mudah. Caranya dengan scan *QR code* tersebut ke aplikasi kemudian diproses menjadi *ciphertext*.

II. METODE PENELITIAN

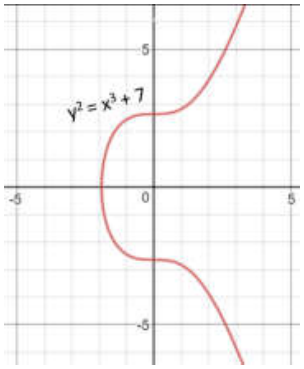
Pada penelitian ini, metode yang digunakan adalah Metode algoritma ECC. Kemudian *QR code* menjadi medianya. Metodologi penelitian adalah suatu proses untuk keperluan penelitian dengan menggunakan data. Pada penelitian ini mempunyai beberapa tahapan yaitu:

A. Identifikasi kebutuhan dan System
Pada tahap ini adalah menganalisa beberapa hal yang diperlukan untuk pembuatan sistem.
1. Analisis Kebutuhan
Penelitian ini menggunakan metode algoritma ECC yang enkripsi dan dekripsi sebuah URL menbutuhkan sebuah sistem yang mendukung penelitian yaitu :
   a. Perangkat Keras
   Perangakat keras yang dibutuhkan adalah laptop dengan spesifikasi yaitu :
      1) Processor : Intel Core I3
      2) RAM : 8,00 GB
      3) *System Type* : 64 bit- *operating system*
   b. Perangkat Lunak
   Perangkat lunak yang dibutuhkan yaitu :
      1) Sistem Operasi : Microsoft Windows 10
      2) Bahasa Pemrograman : *Python*
2. Analisis System
Penelitian ini menggunakan algoritma ECC yang mengubah *plaintext* menjadi *ciphertext* dan sebaliknya. Setelah menjadi *ciphertext* dibuat menjadi *Qr code* dan sebaliknya. Penelitian ini melakukan enkripsi menggunakan *public key* dan melakukan dekripsi menggunakan *private key*. *Plaintext* yang diproses merupakan URL.
merupakan URL.

B. Studi Literatur
Tahap ini menjelaskan penerapan proses Enkripsi dan Dekripsi menggunakan algoritma ECC yaitu :
1. Penerapan Algoritma ECC untuk *QR Code*
Semua titik yang memenuhi persamaan ECC dengan bentuk $y^2 = x^3+ax+b$ dan beberapa contoh *e* ECC memenuhi persamaan pada Gbr. 1 kurva simetris simetris dengan sumbu x, kemudian pada Gbr. 2 ECC menggunakan secp256k1 adalah visualisasi algoritma ECC yang menggunakan secp256k1 didasarkan pada persamaan $y^2 = x^3 + 7$ (dari persamaan diatas a = 0, b = 7). Secp256k1 merupakan perameter yang mengacu pada algoritma ECC.

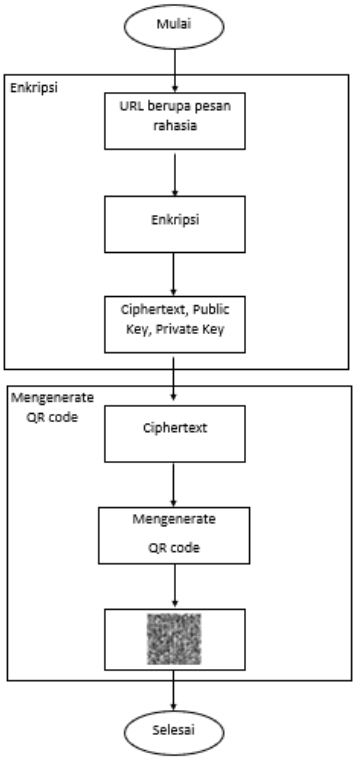


Gbr 1. kurva simetris dengan sumbu x



Gbr 2. ECC menggunakan secp256k1

2. Proses Enkripsi metode ECC



Gbr 3. *flowchart* enkripsi algoritma

Penjelasan proses enkripsi pada Gbr. 3 *flowchart* enkripsi algoritma menggunakan ECC yaitu :

a. Masukan *plaintext* yang merupakan pesan rahasia yang akan dikirimkan berupa teks. Teks tersebut digunakan untuk enkripsi data.
b. Pada proses enkripsi yaitu:
   1) Masukan text URL yang akan dienkripsi
   2) Sistem men-generate private key dan public key menggunakan fungsi urandom dengan key size 32. dengan ketentuan lebih dari ZERO dan kurang dari GROUP_ORDER

```
GROUP_ORDER = (

b'\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xf
f'

b'\xfe\xba\xae\xdc\xe6\xafH\xa0;\xbf\xd2^\x8c\xd06
AA'
)
ZERO = b'\x00'
KEY_SIZE = 32

def get_valid_secret() -> bytes:
    while True:
        secret = urandom(KEY_SIZE)
        if ZERO < secret < GROUP_ORDER:
            return secret

def generate_eth_key() -> keys.PrivateKey:
    return keys.PrivateKey(get_valid_secret())

def generateKey():
    eth_k = generate_eth_key()
    privKey = eth_k.to_hex()
    pubKey = eth_k.public_key.to_hex()

    return [privKey, pubKey]
```

Gbr 5. *Source Code private key* dan *public key*

3) Enkripsi menggunakan library yang sudah tersedia di python yaitu eciespy. Text diencode kemudian dienkrip dengan public key dan menghasilkan ciphertext.

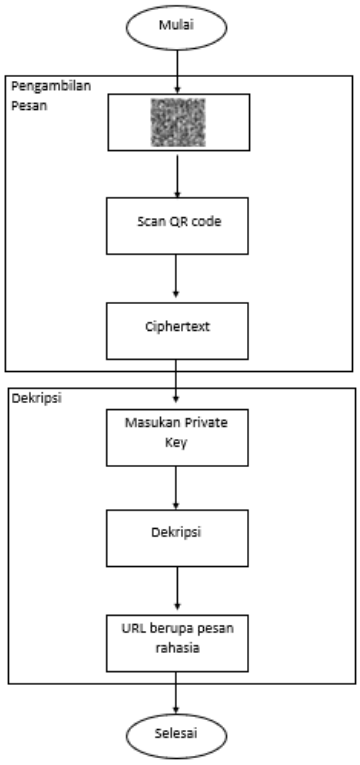
```
from ecies import encrypt, decrypt

def encryptText(publicKey, text):
    return encrypt(publicKey, text.encode("utf8")).hex()
```

Gbr 4. *Source Code* Enkripsi

4) *Ciphertext* terdiri dari 65 byte *public key* dan nilai enkripsi.

c. *Ciphertext* yang dibuat menjadi *QR code*. *Public key* untuk enkripsi data. *Private key* untuk mendekripsi ciphertext menjadi plaintext.
d. Untuk membuat *ciphertext* menjadi *QR code* menggunakan aplikasi QR code Generator.
e. Tahap ini *QR code* sudah siap dikirimkan oleh pengirim.

3. Proses Dekripsi metode ECC



Gbr 6. *flowchart* dekripsi algoritma

Penjelasan proses dekripsi pada Gbr. 6 *flowchart* dekripsi algoritma
menggunakan ECC yaitu :

a. Tahap ini memasukkan *QR code* yang telah dikirimkan untuk dipindai.
b. Proses scan *QR code* menggunakan aplikasi QR & Barcode Scanner untuk mendapatkan *ciphertext*
c. Tahap ini ciphertext telah dipindai untuk proses dekripsi
d. Masukkan private key untuk mendekripsi *ciphertext* menjadi *Plaintext*

e. Pada proses dekripsi yaitu:
   1) Masukan *ciphertext*
   2) Konversi *ciphertext* ke bytes menggunakan fungsi fromhex

```
chipper_text = bytes.fromhex(input_chipper)
```

   3) Masukan private key
   4) Cek valid atau tidak private key
   5) Mengambil 65 byte pertama ciphertext yang sudah dikonversi ke byte menjadi public key dan sisanya sebagai nilai enkripsi

```
def decryptText(privKey,
chipperText):
    try:
        dec= decrypt(privKey,
chipper_text).decode('utf-8')
        return dec
    except:
return "Private key atau chipper text
salah"
```

Gbr 7. *Source Code* Dekripsi

   6) Byte dikonversi dan mengembalikan nilai bertipe string

f. *Plaintext* berupa URL yang merupakan pesan rahasia yang telah dikirimkan berupa text.

C. Pengujian

Data yang sudah dianalisis kemudian diterjemahkan mengunakan desain yang dimengerti oleh pengguna, sehingga dapat mengetahui alur system dan jalannya system tersebut. Pengujian dari sytem menggunakan algoritma ECC yaitu sebagai berikut :

1. Pengujian system Enkripsi

Gbr 8. *input* text URL

Gbr 9. *output* Enkripsi

Gbr 10. *QR code* 1

Pada Gbr. 8 *Input* text URL terdapat 2 pilihan yaitu enkripsi dan dekripsi, jika memilih enkripsi maka masukan angka 1. Kemudian masukan text URL.

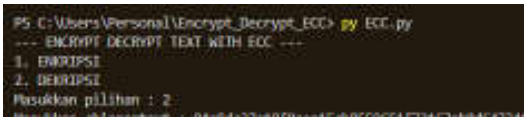
Gbr. 9 *output* Enkripsi merupakan output setelah memasukkan text URL. *Output* dari enkripsi yaitu:

Masukkan text URL : https://www.youtube.com/
Private Key :
0x19859f69bae69545db79cb92a509ca91c212f3c27
6882e6f9297426f601cd99d
Public Key :
0x49befa4e50f241a6675f82c3fa27452a80f5d96aaeb
9aa4d784e5778f4887099bd4b136e6d04972204c29b
0fecd432bc245c202b7476cef3a387644bd78e6157
Ciphertext :
04e9dc37e18f0aee15cb8559651f73167ab0d647346
dfd6e5f1e529c8ae30c2393cd1989101d1e980f6bf44
d0f7809c5940b6223282b4b40c2e405135978383cd1
c9641d2cf3fbfa6dd2d57ca9ea34c97343c6b973e9b4
2a2afc32535507354eccd8fbf0c4fb6496f7f9db27f48
e6a69bfca92d97f9fcb4c37
Time Execute : 0.012966632843017578 seconds

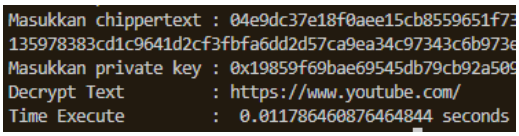
Teks URL yang telah di enkripsi akan dibuat menjadi *QR code* pada Gbr. 10 *QR code*.

2. Pengujian system Dekripsi

Gbr 11. *QR code* 2



Gbr 12. *input* data Dekripsi



Gbr 13. *output* Dekripsi

Pada Gbr. 11 *QR code* 2 kemudian dipindai dan menghasilkan *ciphertext*. Pada Gbr. 12 *input* data Dekripsi terdapat 2 pilihan yaitu enkripsi dan dekripsi, jika memilih dekripsi maka masukan angka 2. Kemudian Masukkan *Ciphertext dan Private Key* untuk dekripsi menjadi *plaintext* atau URL berikut:

Decrypt Text : https://www.youtube.com/
Time Execute : 0.011786460876464844 seconds

output dari dekripsi akan menghasilkan plaintext atau URL pada gambar Gbr. 13 *output* Dekripsi.

D. Data Penelitian

Pada penerapan *Algoritma* ECC digunakan untuk enkripsi dan dekripsi URL beberapa data yang harus dirahasiakan adalah tiket perjalanan, KTP/SIM, selfie dengan KTP/SIM, Dokumen keuangan, dan dokumen pribadi, data yang digunakan dalam pengujian ini terdapat 15 data.

III. HASIL DAN PEMBAHASAN

Penerapan Algoritma ECC pada program yaitu:

A. Pengujian Algoritma ECC untuk *QR code*

1. Panjang masukan

Panjang masukan yang telah diujikan menggunakan *plaintext*. *Plaintext* mempunyai 24 karakter membutuhkan waktu 0.012966632843017578 detik, 97 karakter membutuhkan waktu 0.008981466293334961 detik, dan 237 karakter membutuhkan waktu 0.010006904602050781 detik. Maka semakin banyak jumlah masukan semakin lama proses enkripsi dan dekripsinya.

2. Variasi kunci

Pengujian ini untuk mengetahui variasi kunci dari *ciphertext, public key dan private key.* Waktu yang dibutuhkan untuk memvariasi kunci 15 sampel dari 24 – 237 karakter adalah 0.012966632843017578 detik - 0.010006904602050781 detik.

3. Kenerja waktu

Pengujian ini menunjukkan jumlah rata – rata waktu yang dibutuhkan dalam enkripsi dengan menggunakan 15 sampel yaitu sebesar 0.008950471878051757866666666666667 detik. Waktu yang dibutuhkan untuk mendekripsi yang menggunakan 15 sampel memiliki rata- rata waktu sebesar 0.015878629684448242213333333333333 detik.

B. Performa Algoritma Elleptic Curve Criptografy (ECC) untuk *QR Code*

1. Uji kecepatan Enkripsi dan kecepatan Dekripsi

Tabel 1. Hasil uji kecepatan enkripsi

No.	Panjang karakter plaintext	Kecepatan Enkripsi
1.	24	0.012966632843017578
2.	25	0.009470939636230469
3.	27	0.008974790573120117
4.	27	0.005583286285400391
5.	26	0.005583286285400391
6.	131	0.006985664367675781
7.	126	0.008975982666015625
8.	123	0.009781837463378906
9.	108	0.00967097282409668
10.	97	0.008981466293334961
11.	187	0.010970115661621094
12.	196	0.006360054016113281
13.	189	0.009971141815185547
14.	176	0.009974002838134766
15.	237	0.010006904602050781

Pada uji kecepatan enkripsi pada Tabel 1 diatas merupakan uji enkripsi yang dilakukan pada 15 sampel yang memiliki panjang karakter URL berbentuk teks yang berbeda-beda yang menghasilkan kenerja waktu yang berbeda-beda. Hasil tersebut akan dibandingan dengan hasil uji kecepatan dekripsi.

Tabel 2. Hasil uji kecepatan dekripsi

No.	Panjang karakter plaintext	Kecepatan Dekripsi
1.	24	0.011786460876464844
2.	25	0.015523910522460938
3.	27	0.01695418357849121
4.	27	0.015957117080688477
5.	26	0.016962766647338867
6.	131	0.01695418357849121
7.	126	0.016066551208496094
8.	123	0.019590139389038086
10.	97	0.0189511775970459
11.	187	0.015070915222167969
12.	196	0.01977825164794922
13.	189	0.01695728302001953
14.	176	0.0027322769165039062
15.	237	0.017940044403076172

Pada uji kecepatan dekripsi pada Tabel 2 diatas merupakan uji dekripsi yang dilakukan pada 15 sampel yang memiliki panjanga karakter URL berbentuk teks yang berbeda-beda yang menghasilkan kenerja waktu yang berbeda-beda. Perbadingan uji kecepatan antara enkripsi dan dekripsi memiliki kecepatan yang tidak jauh beda.

2. Pengukuran Ukuran Ciphertext

Tabel 3. Hasil uji peforma pemnambhan plaintext dan ciphertext

No.	Panjang karakter Plaintext	Panjang karakter Ciphertext	Penambahan Ukuran
1.	24	242	9.917 %
2.	25	244	10.245 %
3.	27	250	10.8 %
4.	27	250	10.8 %
5.	26	246	10. 569 %
6.	131	456	28.778 %
7.	126	448	28.125 %
8.	123	442	27.828 %
9.	108	410	26.341 %
10.	97	390	24.871 %
11.	187	570	32.807 %
12.	196	586	33.447 %
13.	189	572	33.041 %
14.	176	548	32.116 %
15.	237	668	35.479 %

Hasil uji performa untuk membandingkan penambahan plaintext dan *Ciphertext*. Penambahan ukuran plaintext ke *ciphertext* mempunyai penambahan yang teratur. Jadi semakin sedikit plaintext maka akan semakin sedikit *ciphertext* dan bergitu sebaliknya. Penambahan sekitar 10 – 35 %.

IV. KESIMPULAN

Berdasarkan penelitian ini terdapat beberapa kesimpulan yaitu:

1. Pada pengujian panjang masukan semakin banyak jumlah masukan karakter semakin lama proses enkripsi da dekripsi. Waktu yang dibutuhkan untuk enkripsi memiliki rata-rata waktu 0.008950471878051757866666666666667 detik dan waktu yang dibutuhkan untuk dekripsi memiliki rata- rata waktu 0.015878629684448242221333333333333 detik.
2. Pada uji kecepatan enkripsi dan dekripsi memiliki panjang karakter yang berbeda-beda maka akan mendapatkan kenerja waktu yang berbeda.

3. Padauji performa semakin sedikit karakter *plaintext* maka akan semakin sedikit *ciphertext* dan begitu sebaliknya.

UCAPAN TERIMAKASIH

Ucapan terimakasuh dan syukur saya sampaikan kepada :

1. Allah SWT atas segala rahmat dan hidayah, sehingga penulis bisa menyelesaikan penelitian ini.
2. Orang tua dan keluarga penulis yang senantiasa menberikan dukungan untuk selalu melakukan apapun yang terbaik.
3. Bapak Asmunin, S.Kom., M.Kom. selaku Dosen Pembimbing yang dengan sabar membimbing penelitian ini dari awal hingga akhir.
4. Bapak I Made Suartana, S.Kom., M.Kom. dan bapak Ricky Eka Putra, S.kom.,M.Kom selaku dosen penguji yang selalu memberikan saran yang baik
5. Sahabat-sahabat dan teman-teman yang selalu memberikan semangat dan dukungan.

REFERENSI

[1]Sipayung Syafmi Giffari. 2019. Analisa Keamanan URL yang Menggunakan Algoritma 3DES. STMIK Budi Darma,

[2] Adiwiganda M R. 2019. Implementasi Algoritma Kriptografi Tiny Encryption Algorithm Pada Keamanan Komunikasi Dan Data Smart Card

[3] Masita. 2020. Perbandingan Algoritma Ecdh Dan Algoritma Ecc Dalam Mengamankan Pesan Gambar. Jurnal Informasi dan Teknologi Ilmiah (INTI).

[4] Aziz armirara refa. 2017. Implementasi Enkripsi Text Menggunakan Algoritma Advanced Encryption Strandard Dan Elliptic Curve Criptografy, Tugas Akhir Institut Teknologi Negeri Sepuluh November

[5] Nuraeni Fitri. 2020. Implementasi Skema QR-Code dan Digital Signature menggunakan Kombinasi Algoritma RSA dan AES untuk Pengamanan Data Sertifikat Elektronik . Sekolah Tinggi Teknologi Garut

[6] Akbar Suhadak,dkk. 2019. Implementasi Algoritme COZMO untuk Enkripsi dan Dekripsi Data pada QR Payment. Universitas Brawijaya.

[7] Ibrahim Saleh dkk. 2020. Efficient Image Encryption Schene Using Henon Map, Dynamic S-Boxes and Elleptic Curve Criptografy. IEEE

[8] Maulana Dhika dkk. 2021. Implementasi Algoritma Advanced Encryption standard (AES) untuk keamanan QR-Code sebagai Digital Signature Pada Aplikasi E-Surat. Journal of Economic, Business and Engineering (JEBE) Vol. 3

[9] Fatmala Yuniar Siska dkk. 2018. Implementasi Algoritma Speck untuk Enkripsi Dan Dekripsi pada QR Code. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer

[10] Rahmad Cahya dkk. 2020. Rancang Bangun Sistem Presensi Qr Code Yang Dipadukan Dengan Digital Signature Rsa Dan Face Detection Menggunakan Metode Template Matching Berbasis Android. Politeknik Negeri Malang

[11] https://pypi.org/project/eciespy/. (2022,Juni 6). Elliptic Curve Integrated Encryption Scheme for secp256k1 in Python.