Analisis Elliptic Curve Cryptography (ECC) dalam Pembuatan dan Verifikasi Tanda Tangan Digital pada Transaksi Blockchain

Manuel Thimoty Silalahi - 13524102
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: manuelsilalahi06@gmail.com , 13524102@std.stei.itb.ac.id

***Abstract*— Kehidupan digital manusia yang setiap tahun terus meningkat mendorong beberapa perubahan pada cara mereka berinteraksi. Salah satu perubahan yang dirasakan adalah dalam hal transaksi digital. Sistem blockchain yang lahir melalui kemajuan ilmu teknologi dan pengetahuan, serta mengedepankan keamanan pengguna berhasil menjadi sistem revolusioner peradaban umat manusia, khususnya bagaimana mereka melakukan transaksi. Elliptic Curve Cryptography (ECC) memainkan peran yang sangat penting dalam perwujudan sistem ini. Selain itu, pada setiap transaksi blockchain, diperlukan sistem keamanan tambahan yaitu tanda tangan digital. Dengan menggunakan ECC, kita dapat menentukan tanda tangan digital untuk suatu pesan transaksi. Proses ini juga dinamakan sebagai Elliptic Curve Digital Signature Algorithm (ECDSA). Dengan demikian, keamanan transaksi pengguna dapat terjamin dan sistem transaksi secara desentralisasi dapat semakin prospek transaksi yang bagus untuk ke depannya.**

I. PENDAHULUAN

A. Latar Belakang

Dunia saat ini tengah bergerak pesat ke arah digital, ditandai dengan adopsi teknologi informasi dan komunikasi secara massif dalam berbagai kehidupan perekonomian dan bisnis. Perubahan ini juga mengubah paradigma dunia terhadap sistem perekonomian, dimana sistem perekonomian semakin banyak dilakukan secara digital, mulai dari sektor perdagangan, hiburan, bisnis, transportasi dan sektor – sektor lainnya.

Transformasi ekonomi digital melibatkan beberapa tahap :

a. Digitalisasi : Konversi data dari fisik menjadi digital.
b. Integrasi : Penggabungan data digital ke dalam suatu sistem yang terintegrasi.
c. Automatisasi : Pemanfaatan teknologi untuk mengotomatisasi suatu sistem.
d. Inovasi : Pengembangan produk, layanan, dan model bisnis berbasis teknologi.

Transaksi online yang menjadi tulang punggung dalam ekonomi digital menawarkan kecepatan dan keamanan, serta jangkauan yang luas. Kecepatan dan keamanan dapat diibaratkan sebagai 2 saudara yang saling melengkapi satu sama lain. Sebuah sistem yang cepat, tetapi tidak aman sama saja tidak berarti, begitupun sebaliknya.

Oleh karena itu, dibutuhkan sistem transaksi berbasis *kriptografi*, sebuah sistem dimana setiap transaksi diubah menjadi kode – kode unik dan bersifat *irreversible* atau tidak bisa dibalikkan. Dibutuhkan sebuah sistem dimana pengguna manapun tidak bisa melakukan komputasi terhadap data pribadi pengguna lain dan tanpa melibatkan pihak ketiga.

Satoshi Nakamoto berhasil memecahkan masalah ini dengan memperkenalkan suatu transaksi *peer-to-peer* dan terdesentralisasi. Dia mengenalkan *bitcoin* pada tahun 2009. Tidak seperti mata uang pada umumnya, penemuan ini menggunakan jaringan P2P ke jurnal transaksi dan menggunakan kriptografi untuk menyediakan fungsi – fungsi keamanan. Inilah dasar dari sistem *blockchain.*

B. Rumusan Masalah

Berikut adalah rumusan masalah dari makalah ini :

- Bagaimana peran Elliptic Curve Cryptography (ECC) dalam pembuatan dan verifikasi tanda tangan pada Blockchain?
- Bagaimana karakteristik dari ECC? Bagaimana tingkat keamanannya?
- Bagaiman perbandingan algoritma pembentukan tanda tangan dengan ECC atau ECDSA dengan algoritma lain, seperti RSA?

C. Tujuan Penulisan

Adapun tujuan dari penulisan adalah sebagai berikut :

- Mengetahui peran ECC dalam pembuatan dan verifikasi tanda tangan pada Blockchain.
- Mengetahui karakteristik dari ECC dan sistem keamanan dibaliknya.

- Mengetahui perbanding algoritma menggunakan ECC dengan algoritma lain.

D. *Manfaat Penulisan*

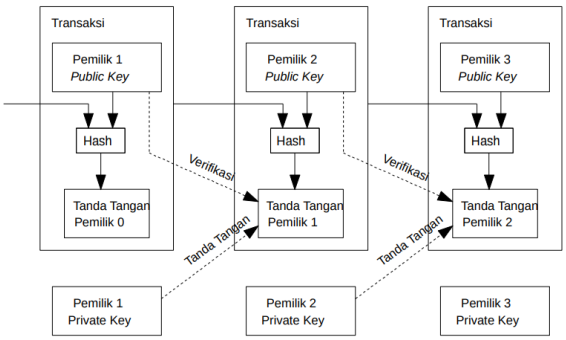
Adapun manfaat dari penulisan adalah sebagai berikut :

- Penulis dapat mengetahui sistem dibalik transaksi blockchain.
- Penulis dapat mempercayai keamanan dari transaksi blockchain dan ranah yang mengadopsi sistem tersebut.

II. TINJAUAN PUSTAKA

A. *Blockchain dan Transaksi Digital*

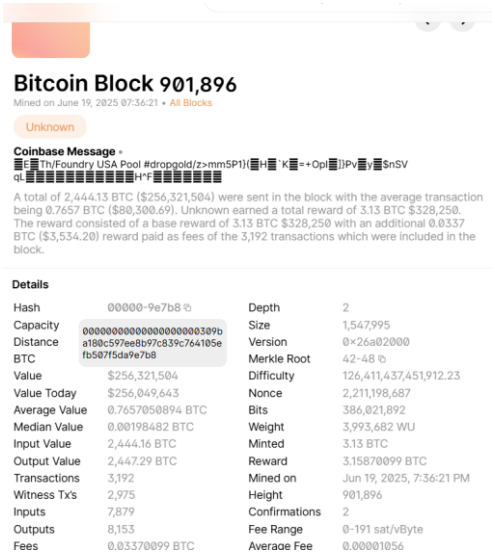
Blockchain dapat didefinisikan sebagai transaksi besar yang menggunakan sistem buku terdesentralisasi. Sistem ini berbeda dengan sistem transaksi konvensional yang bersifat sentralisasi karena menggunakan pihak ketiga untuk melakukan transaksi, seperti bank negara. Pada dasarnya, blockchain yang diambil dari kata *block* (blok) dan *chain* (rantai) memiliki arti blok – blok data yang saling terhubung satu sama lain. Setiap blok tersebut berisi beberapa data, yaitu : data transaksi, hash (yang akan kita bahas setelah ini) dari blok sebelumnya, dan tanda tangan digital (topik utama dari makalah ini).



Gambar 1. Diagram Sistem Blockchain

Ketika Anda memasuki *network*, jaringan dimana sistem blockchain bekerja, Anda diberikan 2 kunci, yaitu *kunci privat* dan *kunci publik.* Kunci privat hanya diketahui oleh Anda sendiri, sementara kunci publik diketahui semua orang. Misalkan Anda ingin membuat transaksi ke Bob, teman Anda. Anda membuat transaksi : "Saya mengirim 1 BTC dari alamat Saya ke Alamat teman saya". Lalu, Anda menggunakan kunci privat Anda untuk membuat tanda tangan digital sebagai tanda bahwa Anda yang benar – benar melakukan transaksi teersebut.

Setelah itu, transaksi Anda akan masuk ke dalam *mempool*, sebuah "ruang tunggu" sebelum masuk ke jaringan global. Lalu, disinilah *miner* melakukan pekerjaannya. Miner akan mengecek transaksi Anda. Jika transaksi Anda valid, maka transaksi Anda akan dikategorikan sebagai kandidat blok. Setelah semua transaksi sudah dicek dan valid, maka disinilah konsep Proof of Work dalam dunia blockchain berjalan. Mereka akan mengerjakan sejumlah permasalahan matematika yang membutuhkan daya komputasi yang besar dan waktu yang lama. Permasalahan ini adalah menemukan sebuah angka 'keberuntungan', atau yang disebut sebagai *nonce,* yang dimana jika nonce tersebut dan data transaksi digabukan kemudian di hash, menghasilkan angka dengan sejumlah bit 0 di depannya.

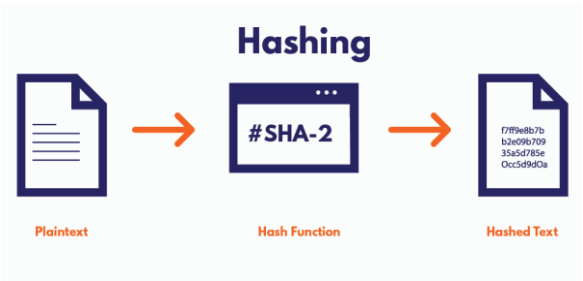


Gambar 2. Salah satu contoh dari block pada sistem *blockchain.* Sumber : www.blockchain.com

Jika berhasil, maka miner akan menyiarkan blok yang sudah lengkap (yang dimana permasalahan matematikanya telah selesai) ke jaringan, kemudian akan diverifikasi oleh sistem. Setelah itu, blok ini akan ditambahkan sebagai blok baru di sistem blockchain. Proses ini akan terus berulang, sampai ditemukan rantai blok yang "sudah cukup panjang untuk dinyatakan sah.

Setelah *block* dinyatakan sah dan sudah ditambah ke rantai, maka *miner* akan mendapatkan reward berupa *bitcoin*. Inilah sebuah proses besar yang dinamakan dengan *bitcoin mining*. Diketahui bahwa hanya ada sekitar 21.000.000 bitcoin yang bisa di-*mining* yang menyebabkan harga bitcoin dari tahun ke tahun cenderung naik.

B. *Fungsi Hash*



Gambar 3. Diagram bagaimana fungsi *hash bekerja. Sumber* : https://khalilstemmler.com/blogs/data-structures-algorithms/hash-tables/

Misalkan saja ada sebuah pesan m. Fungsi hash (h) memetakan nilai – nilai ke sebuah string yang panjangnya *fix*. Misalkan hasil fungsi hash adalah f, dengan kata lain

$$h(m) = f$$

Perhatikan bahwa kita hampir mustahil untuk menemukan 2 nilai yang berbeda untuk hasil hash yang sama, atau secara matematis akan sangat sulit (hampir mustahil) menemukan m_1, m_2 dengan $m_1 \neq m_2$ sehingga $h(m_1) = h(m_2)$. Dengan kata lain, hasil dari fungsi hash bersifat unik.

Sifat dari fungsi hash yang lain salah satunya adalah *irreversible.* Misalkan kita mempunyai hasil hash f. Lalu, kita ingin mencari message m sehingga $h(m) = f$. Kita lakukan dengan menghitung $m = h^{-1}(f)$. Perhatikan bahwa ini tidak bisa kita lakukan karena fungsi hash bersifat unik dan tidak memiliki pola tertentu. Salah satu trik yang dapat dilakukan adalah dengan *brute force,* atau mencoba satu persatu kemungkinan nilai m. Untuk *bit* hasil hash yang besar, misalkan saja SHA-256, hal ini tentunya sangat sulit dilakukan dan perlu daya komputasi yang tinggi.

Fungsi hash merupakan salah satu fundamental dari transaksi blockchain, khususnya dalam pembuatan dan verifikasi tanda tangan. Misal *user* A ingin mengirim sebuah pesan m ke user B. Lalu, user A membuat tanda tangan digital sebagai "penanda" bahwa transaksi tersebut dilakukan oleh A sendiri. Namun, sebelum membuat tanda tangan digital, message tersebut harus di *hash* terlebih dahulu menggunakan fungsi hash, salah satu contohnya adalah *SHA-256.* Katakan hasil dari hash ini adalah *message digest.* Kemudian, dengan menggunakan kunci public dan suatu algoritma, yang akan kita bahas, kita dapat meng-*generate* tanda tangan digital. Lalu, message yang diterima oleh *B* kemudian diverifikasi dengan meng-*hash* message tersebut. Lalu, tanda tangan tersebut lalu diverifikasi dengan algoritma khusus dan kunci publik. Jika hasil hash cocok, dan verifikasi berhasil, maka message tersebut valid.

C. Tanda Tangan Digital

Tanda tangan digital adalah sebuah skema matematika yang digunakan sebagai bukti bahwa suatu pesan/transaksi tidak diubah atau dipalsukan dan menunjukkan bahwa pesan tersebut benar – benar berasal dari pengirim. Ini adalah salah satu fundamental penting dalam dunia *blockchain.*

Misalkan Alice ingin mengirim pesan ke Bob. Alice akan meng-*hash* pesannya, lalu dienkripsi dengan kunci privatnya. Alice pun akan mengirim pesan tersebut lengkap dengan hasil enkripsi dari hash tersebut. Saat Bob menerima pesannya, Bob akan menggunakan kunci publik Alice untuk mendeskripsi hash dari pesan Alice. Selain itu, Bob juga akan meng-*hash* pesan dari Alice. Jika kedua *hash* ini cocok, maka Bob bisa yakin bahwa pesan dari Alice ini tidak 'rusak'. Jika tidak cocok, maka pesan dari Alice dapat dipastikan berubah/dipalsukan/rusak.

D. Elliptic Curve Cryptography (ECC)

Pada suatu field K, Elliptic Curve didefinisikan sebagai nilai – nilai x, y dengan $x, y \in K$ dan memenuhi persamaan untuk suatu $a, b \in K$.

$$y^2 = x^3 + ax + b$$

Perhatikan juga bahwa persamaan kubik $x^3 + ax + b$ harus tidak mempunyai banyak akar. Kita definisikan juga sebuah titik O sebagai titik "infinity".

Sekarang kita akan meninjau untuk $K = \mathbb{R}$. Kita Katakan dua titik pada kurva E, P dan Q dengan $P(x_P, y_P)$ dan $Q(x_Q, y_Q)$. Penjumlahan kedua titik tersebut, dengan hasilnya adalah Q atau didefinisikan secara matematis

$$P + Q = R$$

Dapat diperoleh dengan menggambar garis lurus yang melewati dua titik P dan Q. Perpotongan garis ini dengan E adalah $-R$. Dengan demikian R adalah pencerminan titik $-R$ terhadap sumbu x. Secara matematis,

$$x_R = s^2 - (x_P + x_Q)$$
$$y_R = s(x_P - x_R) - y_P$$

Dengan $s = \frac{y_P - y_Q}{x_P - x_Q} = \frac{y_R - y_Q}{x_R - x_Q}$.

Bukti. Tinjau bahwa persamaan garis linear yang melalui P dan Q adalah

$$l(x) = s(x - x_P) + y_P$$

Sekarang, karena P, Q, R ada di kurva E, kita punya persamaan

$$y_P^2 = x_P^3 + Ax_P + B \ldots\ldots (1)$$
$$y_Q^2 = x_Q^3 + Ax_Q + B \ldots\ldots (2)$$
$$y_R^2 = x_R^3 + Ax_R + B \ldots\ldots (3)$$

Dengan mengurangkan (1) dengan (2), diperoleh

$$(y_P - y_Q)(y_P + y_Q)$$
$$= (x_P - x_Q)(x_P^2 + x_Q^2 + x_P x_Q) + A(x_P - x_Q)$$

Atau ekuivalen dengan

$$s(y_P + y_Q) = x_P^2 + x_Q^2 + x_P x_Q + A \ldots\ldots (4)$$

Dengan cara yang sama, untuk (2) dan (3),

$$s(y_Q + y_R) = x_Q^2 + x_R^2 + x_Q x_R + A \ldots\ldots (5)$$

Mengurangkan (4) dengan (5) memberikan hasil

$$s(y_P - y_R) = (x_P - x_R)(x_P + x_R) + x_Q(x_P - x_R)$$

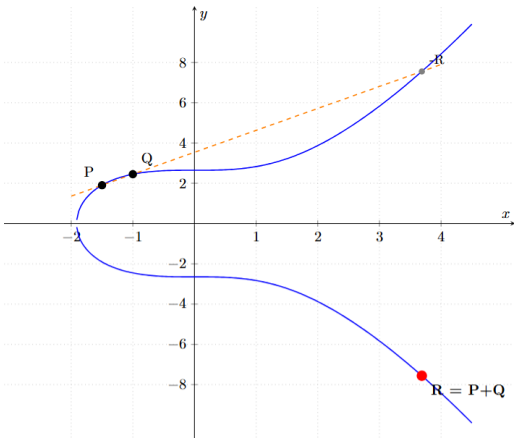
Bagikan dengan $x_P - x_R$ memberikan persamaan

$$s^2 = x_P + x_Q + x_R$$
$$x_R = s^2 - (x_P + x_Q)$$

Perhatikan bahwa kita ingin mengambil pencerminan dari titik potong antara garis lurus dengan kurva E. Maka,

$$y_R = -l(x_R) = -(s(x_R - x_P) + y_P)$$
$$y_R = s(x_P - x_R) - y_P$$

Sebagaimana dengan apa yang ingin kita buktikan.



Gambar 4. Ilustrasi dari *Point Addition.*

Dengan konsep yang sama seperti point addition, sebuah titik $R = 2P$ dapat diperoleh dengan menggunakan algoritma point addition dengan $P = Q$. Dengan demikian, kita dapat melakukannya dengan mencari garis tangen (garis singgung)

terhadap titik P. Kemudian, perpotongan garis tersebut dengan E adalah titik $-R$. Dengan merefleksikannya terhadap sumbu x, diperoleh R.

Asumsikan kita mempunyai titik P. Kita akan mencari titik Q dengan $Q = kP$. Untuk melakukan hal ini, kita dapat menggunakan algoritma dari Point Doubling dan Addition sekaligus. Misal, untuk $k = 19$, kita dapat menghitung Q,

$$Q = 19P = 2\big(2\big(2(2P)\big) + P\big) + P$$

Namun, kita tidak bisa me-reverse proses ini. Misalkan kita mengetahui Q dan P. Tujuan kita adalah ingin mencari nilai k. Namun, karena proses Point Multiplication menghasilkan titik yang hampir tidak memiliki pola sama sekali, maka untuk mencari nilai $k = \frac{Q}{P}$ hanya bisa dilakukan dengan mencoba satu persatu semua nilai k yang mungkin. Untuk k yang sangat besar, hal ini pasti sangatlah sulit. Selain itu, untuk setiap k yang berbeda, mustahil untuk menemukan nilai Q yang sama. Dengan kata lain titik – titik hasil dari point multiplication bersifat unik.

Katakan sebuah titik $\mathcal{O}$ sebagai **titik netral** jika dia memenuhi aturan berikut :

$$P + \mathcal{O} = P$$

Dalam geometri, titik $\mathcal{O}$ tidak memiliki absis dan ordinat. Titik ini bisa dikatakan berada di "tak terhingga". Misalkan saja sebuah titik di kurva Elliptic, X. Kemudian pencerminan dari X terhadap sumbu-x menghasilkan $-X$. Bagaimana point addition untuk $X + (-X)$? Titik ini "tidak ditemukan" atau berada di titik tak terhingga, atau dengan kata lain,

$$X + (-X) = \mathcal{O}$$

E. Implementasi ECC pada Sistem Blockchain

Dalam sistem blockchain, ECC adalah salah satu fundamental. Akan diambil contoh dalam implementasi pada *bitcoin,* sebuah koin kripto yang mengadopsi sistem blockchain dalam sistem transaksinya. Didefinisikan suatu tuple dalam sebuah finite field F_p (titik yang terdefinisi pada ECC dibuat dalam modulo p)

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

Dengan spesifikasi sebagai berikut :

- p adalah modulo p.
- a, b adalah konstanta dari koefisien kurva, atau dengan kata lain memenuhi $y^2 = x^3 + ax + b \ (mod\ p)$.
- G adalah generator, atau point dasar yang digunakan untuk menghasilkan kunci.
- n adalah order dari titik G, yaitu nilai terkecil sehingga $nG = \mathcal{O}$.
- h adalah kofaktor, didefinisikan sebagai $h = \frac{\#E}{n}$, dimana $\#E$ adalah jumlah titik pada kurva dan n adalah order dari G. Biasanya nilai $h = 1$ atau $h = 2$

Untuk setiap data transaksi/pesan, pasti ada kunci public dan kunci privat yang kegunaannya sudah dijelaskan sebelumnya. Pada kurva elliptic, kunci publik sendiri di-*generate* dengan :

$$K_p = K_s \times G$$

Dimana K_p adalah kunci public dan K_s adalah kunci privat. Sama seperti sebelumnya, kita tidak bisa mencari nilai K_s menggunakan suatu algoritma tertentu. Cara satu – satunya adalah melakukan *brute force* yang dimana untuk nilai K_s yang sangat besar ini sangatlah lama dilakukan. Inilah yang menjadi alasan mengapa penerapan Eclipctic Curve dalam sistem blockchain bisa menjamin keamanan transaksi.

III. METODOLOGI

Metodologi yang digunakan adalah metode deskriptif analitis dengan pendekatan studi literatur dan simulasi. Data dikumpulkan melalui beberapa sumber yang bisa mendeskripsikan bagaimana dunia blockchain bisa bekerja. Untuk itu, akan dilakukan juga analisis dari Eclipctic Curve Cryptography dan implementasinya dalam pembuatan dan verifikasi tanda tangan. Sebagai tambahan, analisis dan pembahasan akan dilakukan juga dalam bahasa pemograman C.

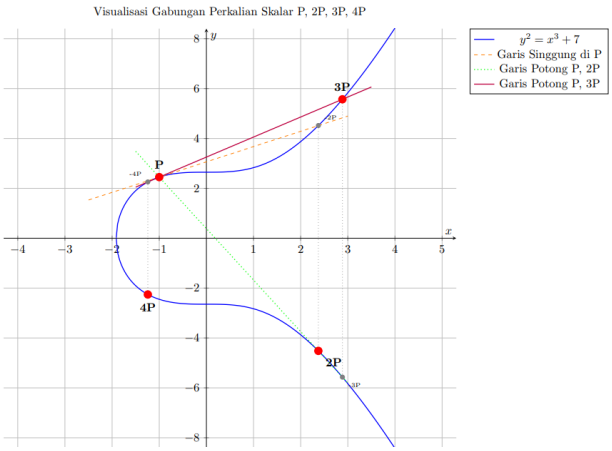
A. Analisis Sistem Generating Public Key oleh ECC

Sebelumnya, telah dijelaskan bahwa public key dapat di-*generate* dengan cara

$$K_P = K_S \times G$$

Selain itu, dijelaskan juga bahwa untuk setiap K_S yang berbeda, nilai K_P juga pasti berbeda. Fakta ini penting untuk memastikan kemanan nilai K_S. Hal ini akan kita bahas lebih detail nanti.

Sekarang, kita akan menganalisis keunikan dari titik – titik yang dihasilkan. Misalkan kurva $y^2 = x^3 + 7$. Kita akan melihat titik – titik $P, 2P, 3P, 4P$ dari kurva ini.



Gambar 5. Penggambaran titik $P, 2P, 3P, 4P$ pada kurva $y^2 = x^3 + 7$.

Dapat dilihat bahwa titik – titik tersebut di-generate dengan pola yang tidak menentu. Namun, titik kP untuk k yang kecil masih dapat dilihat mudah bahwa nilainya pasti berbeda. Bagaimana untuk k yang besar? Untuk menyelesaikan persoal ini, saya kaan menggunakan bahasa pemrograman C. Berikut aalah *code* yang saya gunakan.

```c
#include <stdio.h>

#define Eltype double

typedef struct {
    Eltype x;
    Eltype y;
    int is_identity; // Flag untuk titik netral
} Point;
Point point_identity() {
    return (Point){0, 0, 1}; // Representasi titik netral
}
int is_equal(Point P, Point Q) {
    return (P.x == Q.x && P.y == Q.y && P.is_identity == Q.is_identity);
}
Point point_add(Point P, Point Q) {
    if (P.is_identity) return Q;
    if (Q.is_identity) return P;
    if (P.x == Q.x && P.y == -Q.y)
        return point_identity(); // P + (-P) = O
    if (is_equal(P, Q)) {
        if (P.y == 0)
            return point_identity(); // Titik vertikal, hasilnya O
        double s = (3 * P.x * P.x) / (2 * P.y); // Misal a = 0
        Point R;
        R.x = s * s - 2 * P.x;
        R.y = s * (P.x - R.x) - P.y;
        R.is_identity = 0;
        return R;
    }
    double s = (Q.y - P.y) / (Q.x - P.x);
    Point R;
    R.x = s * s - (P.x + Q.x);
    R.y = s * (P.x - R.x) - P.y;
    R.is_identity = 0;
    return R;
}
```

Gambar 6. Potongan kode C, berisi deklarasi *struct* Point, fungsi *point_identity(), is_equal(),* dan *point_add().*

```c
Point pointmulti(long long n, Point G) {
    Point R = point_identity(); // Harus mulai dari titik netral
    Point Q = G;

    while (n > 0) {
        if (n % 2 == 1) {
            R = point_add(R, Q);
        }
        Q = point_add(Q, Q);
        n /= 2;
    }
    return R;
}
```

Gambar 7. Potongan kode C berisi deklarasi dari fungsi *pointmulti().*

Kemudian, kita akan menjalankannya dengan isi code dari fungsi int main() sebagai berikut :

```c
int main(){
    Point G;
    printf("Masukkan koordinat titik G (x y): ");
    scanf("%lf %lf", &G.x, &G.y);
    G.is_identity = 0; // Titik G bukan titik netral
    long long n;
    printf("Masukkan jumlahj iterasi: ");
    scanf("%lld", &n);
    Point result;
    Point res_arr[1000];
    for(int i=1;i<=n;i++){
        result=pointmulti(i,G);
        if (result.is_identity) {
            printf("Untuk iterasi ke %d : O\n",i); // Titik netral
        } else {
            printf("Untuk iterasi ke %d :(%.2lf,%.2lf)\n",i, result.x, result.y);
        }
        res_arr[i-1] = result; // Simpan hasil ke array
    }
    int count=0;
    for(int i=0;i<n;i++){
        for(int j=i+1;j<n;j++){
            if(is_equal(res_arr[i],res_arr[j])){
                printf("Hasil iterasi ke %d dan %d adalah sama.\n", i+1, j+1);
                count++;
            }
        }
    }
    printf("Total dua titik yang sama: %d\n", count);
}
```

Gambar 8. Potongan kode C berisi fungsi main.

Output dari *code* ini akan dicantumkan di *section* selanjutnya beserta dengan pembahasannya.

B. Analisis Penerapan ECC dalam Pembuatan dan Verifikasi Tanda Tangan

Penerapan ECC dalam tanda tangan digital dikemas dalam sebuah algoritma terkenal, yaitu Elliptic Curve Digital Siganture Algorithm (ECDSA). Spesifikasi dan detail dari algoritma ini akan dijelaskan setelah ini. Untuk *bitcoin,* beberapa konstanta tupel $T = (p, a, b, G, n, h)$ adalah konstanta kurva secp256k1 dengan fungsi hashsnya adalah sha-256. Berikut adalah nilai dari setiap konstanta :

Konstanta	Nilai
p	$2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$
a	0
b	7
G	02 79BE667E F9$DCBBAC$ 55A06295 CE870B07 029$BFCDB$ 2DCE28D9 59F2815B 16F81798
n	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 F48A03BB FD25E8CD 0364141
h	1

Tabel 1. Konstanta yang digunakan pada kurva secp265k1.

Akan dilakukan analisis singkat dengan menggunakan bahasa pemrograman. Dengan mempertimbangkan efisiensi baris kode dan penulisan yang baik, akan digunakan bahasa *python*. Library yang digunakan adalah *hashlib* dan *ecdsa* untuk mempermudah pekerjaan. Berikut adalah potongan *code*-nya.

```python
from ecdsa import SigningKey, VerifyingKey, SECP256k1
import hashlib

# Fungsi untuk hash pesan
def hash_message(msg: str) -> bytes:
    return hashlib.sha256(msg.encode()).digest()

# Generate keypair (private dan public key)
sk = SigningKey.generate(curve=SECP256k1)
vk = sk.verifying_key

# Pesan yang akan ditandatangani
message = "Ini adalah pesan penting"
hashed = hash_message(message)

# Tanda tangan pesan (hasil: bytes dari r dan s)
signature = sk.sign_digest(hashed)

# Verifikasi tanda tangan
is_valid = vk.verify_digest(signature, hashed)

print("Pesan:", message)
print("Tanda tangan (hex):", signature.hex())
print("Verifikasi berhasil?" , is_valid)
```

Gambar 9. Potongan kode Python untuk generate signature dan verifikasinya.

IV. HASIL DAN PEMBAHASAN

Pada *section* sebelumnya, telah dijelaskan bagaimana keamanan transaksi dapat dikemas oleh Elliptic Curve Cryptography. Selain itu, dijelaskan juga bahwa ECC berperan

penting dalam pembuatan dan verifikasi tanda tangan digital. Dengan demikian, transaksi yang dilakukan dalam sistem blockchain dapat terjamin keamanannya. Berikut adalah hasil analisis dari Bab III.

A. Hasil Analisis Sistem Generating Public Key pada ECC

Berikut adalah output dari *code* pada analisis ECC dan pembuatan public key. Akan dipilih titik G(5,3) dengan jumlah iterasi sebanyak 30.

```
Masukkan koordinat titik G (x y): 5 3
Masukkan jumlah iterasi: 30
Untuk iterasi ke 1 :(5.00,3.00)
Untuk iterasi ke 2 :(146.25,-1768.62)
Untuk iterasi ke 3 :(6.06,10.34)
Untuk iterasi ke 4 :(36.57,-220.93)
Untuk iterasi ke 5 :(8.72,23.40)
Untuk iterasi ke 6 :(16.31,-65.00)
Untuk iterasi ke 7 :(14.82,56.01)
Untuk iterasi ke 8 :(9.34,-26.43)
Untuk iterasi ke 9 :(31.66,177.84)
Untuk iterasi ke 10 :(6.34,-11.77)
Untuk iterasi ke 11 :(110.63,1163.63)
Untuk iterasi ke 12 :(5.09,-3.94)
Untuk iterasi ke 13 :(6521.43,526640.97)
Untuk iterasi ke 14 :(4.94,2.08)
Untuk iterasi ke 15 :(202.30,-2877.42)
Untuk iterasi ke 16 :(5.82,9.03)
Untuk iterasi ke 17 :(42.73,-279.10)
Untuk iterasi ke 18 :(8.18,20.75)
Untuk iterasi ke 19 :(18.05,-75.95)
Untuk iterasi ke 20 :(13.53,48.56)
Untuk iterasi ke 21 :(10.04,-29.91)
Untuk iterasi ke 22 :(27.68,145.23)
Untuk iterasi ke 23 :(6.65,-13.33)
Untuk iterasi ke 24 :(86.61,805.90)
Untuk iterasi ke 25 :(5.20,-4.93)
Untuk iterasi ke 26 :(1630.36,65830.12)
Untuk iterasi ke 27 :(4.90,1.19)
Untuk iterasi ke 28 :(298.05,-5145.48)
Untuk iterasi ke 29 :(5.62,7.82)
Untuk iterasi ke 30 :(50.58,-359.59)
Total dua titik yang sama: 0
```

Gambar 10. Output dari program C pada BAB III.

Dari sini, bisa dilihat bahwa tidak ada kelipatan dari *P* yang menghasilkan 2 titik yang sama. Untuk jumlah iterasi 30, ini berhasil membuktikan pernyataan kita tentang sistem keamanan kunci privat pada kurva elliptik. Mengapa?

Misalkan saja, ada titik $Q = k_1P$ dan $Q = k_2P$ di kurva elliptic E pada finite field F_p. Ini memenuhi jika dan hanya jika $k_1 \equiv k_2 (mod\ p)$. Dengan kata lain, ada banyak kunci privat (atau nilai privat yang hanya diketahui oleh Anda) untuk satu kunci publik. Ini bisa mengancam keamanan dari sistem blockchain!

B. Implementasi Algoritma Pembuatan dan Verifikasi Tanda Tangan

Selanjutnya, topik utama dari makalah ini, kita akan membahasa bagaimana Elliptic Cuve Cryptography dapat digunakan dalam pembuatan dan verifikasi tanda tangan digital. Perlu diperhatikan bahwa tanda tangan digital adalah salah satu unsur penting dalam dunia blockchain. Dengan adanya tanda tangan, penerima dapat mengetahui bahwa pesan yang diterima dari pengirim adalah pesan yang benar – benar dibuat oleh pengirim, bukan palsu ataupun duplikat.

Berikut adalah *output* dari kode yang telah dibuat pada Bab III, tentang bagaimana menggunakan ECC dalam pembuatan dan verifikasi tanda tangan.

```
Pesan: Ini adalah pesan penting
Tanda tangan (hex):
96216339e28b3a6e75dfaf6033b6b79752af2fb48262478fa4
4659ca13ddaac584011ffe11ae7c5a9f6ea4227f817683c5d2
4c8561ece045c0cbddeec43c8ed5
Verifikasi berhasil? True
```

Sebenarnya, bagaimana cara kerja dari algoritma ini? Bagaimana program ini dapat menghasilkan suatu tanda tangan dalam bentuk heksadesimal dan langsung memverifikasinya? Jawabannya, ini adalah algoritma terkenal dalam sistem blockchain. *Bitcoin* merupakan salah satu *cryptocurrency* yang mengadopsi algoritma ini. Algoritma ini dikenal dengan Elliptic Curve Digital Siganture Algorithm (ECDSA).

Misalkan suatu tupel kurva elliptic pada finite field F_p, seperti pada section sebelumnya adalah $T = (p, a, b, G, n, h)$. Misalkan Alice ingin mengirim pesan m ke Bob. Algoritma pembentukan tanda tangan digital pesan tersebut adalah sebagai berikut :

1. Ambil sembarang bilangan bulat k dengan $1 \leq k \leq n-1$.
2. Hitung $kG = (x_1, y_1)$
3. Misalkan $r = x_1\ (mod\ n)$. Jika $r = 0$, Kembali ke Langkah 1.
4. Hitung hash dari pesan m, $e = HASH(m)$.
5. Hituing $s = k^{-1}(e + rK_s)\ (mod\ n)$ jika $s = 0$, Kembali ke Langkah 1.
6. Maka, diperoleh tanda tangannya adalah (r, s)

Setelah message berhasil dikirim, Bob ingin memverifikasi apakah message tersebut valid. Berikut adalah algoritmanya dalam ECDSA untuk proses verifikasi :

1. Cek apakah r dan s berada di interval $[1, n-1]$.
2. Hitung $e = HASH(m)$, dimana m adalah message yang diterima.
3. Hitung $w = s^{-1}(mod\ n)$.
4. Hitung $u_1 = ew(mod\ n)$ dan $u_2 = rw\ (mod\ n)$.
5. Hitung $X = u_1G + u_2K_P$.
6. Jika $X = \mathcal{O}$, maka tanda tangan tidak valid. Jika tidak, Misalkan $X(x_1, y_1)$ dan hitung $v = x_1(mod\ n)$.
7. Valid jika dan hanya jika $v = r$.

Perhatikan juga bahwa nilai k disini sangatlah krusial. Mengapa? Untuk tanda tangan yang berbeda, kita harus memilih nilai k yang berbeda juga. Jika tidak, seseorang dapat menghitung nilai dari K_s atau kunci privat kita.

Perhatikan, jika ada 2 pesan berbeda, m_1 dan m_2 dengan tanda tangan yang berbeda, sebut saja (r, s_1) dan (r, s_2). Sekarang kita punya 2 persamaan :

$$s_1 = k^{-1}(e_1 + rK_S)\ (mod\ n)$$
$$s_2 = k^{-1}(e_2 + rK_S)\ (mod\ n)$$

Dari sini, diperoleh bahwa $s_1 - s_2 = k^{-1}(e_1 - e_2)\ (mod\ n)$ atau ekuivalen dengan $k = (e_1 - e_2)(s_1 -$

$s_2)^{-1}\ (mod\ n)$. Karena $s_1 = k^{-1}(e_1 + rK_s)$, kita dapat mencari

$$K_s = (s_1 k - e_1) r^{-1}\ (mod\ n)$$

C. *Perbandingan ECDSA vs RSA*

RSA adalah sebuah algoritma pembentukan tanda tangan digital yang dinamai berdasarkan penemunya : Rivest, Shamir, dan Adleman. Ini adalah salah satu algoritma kriptografi tertua yang diadopsi. RSA menggunakan bilangan prima yang besar dalam melakukan enkripsi pesan dan pembentukan tanda tangan digitalnya. RSA membutuhkan daya komputasi yang lebih besar dari EDCSA karena menggunakan kunci yang ukurannya lebih besar.

ECDSA dan RSA merupakan 2 algoritma dalam kriptografi dalam pembuatan tanda tangan digital. Namun, ada beberapa perbedaan yang membuat ECDSA dan RSA berbeda. Beriktut adalah tabel perbandingan dari ECDSA dan RSA.

Fitur	ECDSA	RSA
Ukuran kunci	Lebih kecil	Lebih besar
Kecepatan	Lebih cepat	Lebih lambat
Keamanan	Sangat aman dengan kunci yang kecil	Sangat aman dengan kunci yang lebih besar
Penggunaan sumber	Lebih sedikit	Lebih banyak
Adopsi	Terus meningkat	Sudah banyak digunakan

Tabel 2. Perbandingan ECDSA dan RSA.

V. KESIMPULAN & SARAN

Blockchain merupakan salah satu sistem revolusioner yang menggambarkan bagaimana peradaban digital telah berkembang pesat. Penerapan ilmu matematika serta algoritma tidak pernah lepas dari hal itu, salah satunya Elliptic Curve Cryptography (ECC) yang telah dibahas pada makalah ini. Kombinasi dari ECC dan dunia kriptografi menghasilkan Elliptic Curve Digital Signature Algorithm yang digunakan sebagai tanda keamanan oleh suatu transaksi blockchain.
Banyak penerapan dari algoritma kriptografi dan ECC ini pada dunia blockchain secara langsung. Salah satunya adalah *cryptocurrency* Bitcoin yang mengadopsi algoritma ini dengan spesifikasi secp265k1.

Untuk ke depannya, saya berharap supaya dunia blockchain bisa semakin berkembang dan dapat diadopsi secara umum, bukan hanya pada kalangan tertentu saja. Selain itu, saya juga berharap keamanan dari blockchain ini juga semakin dapat dipastikan, mengingat perkembangan teknologi yang pesat tidak menutup kemungkinan untuk data bocor karena daya komputasi yang sangat kuat. Sebut saja sebuah rumor teknologi revolusioner, *quantum computer.*

a.

VIDEO LINK AT YOUTUBE

Video makalah dapat dilihat pada link berikut :

https://youtu.be/HBCpsJ0Ag0A

(catatan : maaf saya tidak bisa oncam karena kamera laptop saya tiba – tiba rusak 😊)

VI. DAFTAR PUSTAKA

[1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Tersedia di : https://bitcoin.org/bitcoin.pdf

[2] N. Mistry, "An Introduction to Bitcoin, Elliptic Curves and the Mathematics of ECDSA," Project Report, MAC200, 2015.

[3] "Blockchain | Elliptic Curve Digital Signature Algorithm (ECDSA)," *GeeksforGeeks*, May 24, 2024. [Online]. Tersedia di: https://www.geeksforgeeks.org/computer-networks/blockchain-elliptic-curve-digital-signature-algorithm-ecdsa/

[4] Johnson, Don; Menezes, Alfred (1999). "The Elliptic Curve Digital Signature Algorithm (ECDSA)". *Certicom Research. Canada*.

[5] "Elliptic Curve Digital Signature Algorithm," *Bitcoin Wiki*, Jun. 13, 2024. [Online]. Tersedia: https://en.bitcoin.it/wiki/Elliptic_Curve_Digital_Signature_Algorithm. [Accessed: Jun. 18, 2025].

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 1 Juni 2025

Manuel Thimoty Silalahi 13524102