Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.20241127446
Vol. 11, No. 2, April 2024, hlm. 275-284 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

PENGEMBANGAN ALGORITMA ECDSA DENGAN MODIFIKASI PERKALIAN SKALAR MENGGUNAKAN DOUBLE BASE CHAIN

Hafiz Abdul Fatah Kurniawan[*1], Shelvie Nidya Neyman[2], Sony H Wijaya[3]

[1,2,3]Institut Pertanian Bogor, Bogor
Email: [1]hafizkurniawan@apps.ipb.ac.id, [2]shelvie@apps.ipb.ac.id, [3]sony@apps.ipb.ac.id
[*]Penulis Korespondensi

(Naskah masuk: 05 Juli 2023, diterima untuk diterbitkan: 04 April 2024)

Abstrak

Penelitian ini mengeksplorasi modifikasi parameter perkalian skalar dalam algoritma *Elliptic Curve Digital Signature Algorithm* (ECDSA) menggunakan metode *Double Base Chain* (DBC) dalam Era Revolusi 4.0. ECDSA, sebagai algoritma kriptografi asimetris, umum digunakan untuk memberikan integritas dan otentikasi pada data digital. Penelitian ini menilai apakah penggunaan DBC dapat meningkatkan performa ECDSA dalam hal waktu komputasi dan penggunaan memori. Hasil penelitian menunjukkan bahwa meskipun modifikasi ECDSA dengan DBC tidak selalu memberikan performa yang superior. Waktu yang dibutuhkan untuk *key generation*, *signing*, dan *verification* cenderung lebih lama, dan penggunaan memori bervariasi tergantung pada konfigurasi dan parameter tertentu. Faktor-faktor seperti struktur kurva, panjang kurva, parameter kurva, dan implementasi memiliki dampak signifikan terhadap performa. Pengujian *avalanche effect* menunjukkan variasi dalam keteracakan kunci privat pada berbagai jenis kurva. Meskipun belum optimal, penelitian ini menyoroti potensi peningkatan keamanan dengan fokus pada peningkatan keteracakan *private key* pada ECDSA. Temuan ini memberikan dasar bagi eksplorasi lebih lanjut dalam pengembangan algoritma kriptografi yang lebih aman dan efisien, dengan mempertimbangkan keseimbangan antara performa dan keamanan. Dalam konteks Era Revolusi 4.0, di mana keamanan informasi menjadi semakin penting, penelitian ini memberikan wawasan berharga untuk pengembangan teknologi keamanan yang lebih baik.

Kata kunci: *Kriptografi, Double base chain, ECDSA, Enkripsi, Dekripsi, SHA-1*

DEVELOPMENT OF ECDSA ALGORITHM WITH SCALAR MULTIPLICATION MODIFICATION USING DOUBLE BASE CHAIN

Abstract

This study explores the modification of scalar multiplication parameters in the Elliptic Curve Digital Signature Algorithm (ECDSA) using the Double Base Chain (DBC) method in the Era of the Fourth Industrial Revolution (Industry 4.0). ECDSA, as an asymmetric cryptographic algorithm, is commonly used to provide integrity and authentication to digital data. The research evaluates whether the use of DBC can enhance ECDSA performance in terms of computational time and memory usage. The results indicate that, although modifying ECDSA with DBC does not always yield superior performance, the time required for key generation, signing, and verification tends to be longer, and memory usage varies depending on specific configurations and parameters. Factors such as curve structure, curve length, curve parameters, and implementation significantly impact performance. Avalanche effect testing reveals variations in the traceability of private keys across different curve types. Despite not achieving optimal results, the study highlights the potential for improving security by focusing on enhancing the traceability of private keys in ECDSA. These findings provide a foundation for further exploration in the development of more secure and efficient cryptographic algorithms, considering the balance between performance and security. In the context of the Fourth Industrial Revolution, where information security is increasingly crucial, this research offers valuable insights for the advancement of better security technologies.

Keywords: C*ryptography, Double base chain, ECDSA, Encryption, Description, SHA-1*

1. PENDAHULUAN

Era revolusi 4.0 dengan kehadiran teknologi yang semakin canggih dan tingkat keamanan informasi merupakan salah satu isu yang semakin penting saat ini (Cahyadi, 2020) Tanda tangan digital merupakan konsep yang memiliki peran kunci dalam menjamin keamanan informasi saat ini (Genc and Afacan, 2021). Ini digunakan pada banyak bidang

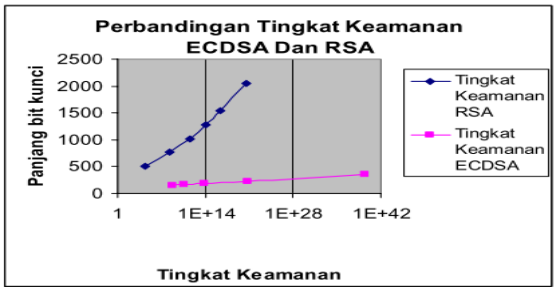
seperti kesehatan, perbankan, perdagangan, internet, pemungutan suara elektronik. Tanda tangan digital digunakan untuk memberikan integritas, otentikasi, dan non-denial (Al-Zubaidie et al., 2019) Kriptografi adalah ilmu yang mempelajari bagaimana pesan dan data yang kita gunakan aman (Prabowo and Afrianto, 2017), selain itu kriptografi memiliki pengertian lain, yakni suatu ilmu tentang teknik enkripsi naskah asli (*plaintext*) yang diacak memanfaatkan sebuah kunci enkripsi sehingga naskah asli tersebut berubah menjadi naskah yang sulit dibaca (*chipertext*) oleh pihak yang tidak memiliki kunci dekripsi (Menezes and Stebila, 2021). Untuk menjaga pesan, maka pesan tersebut dapat diubah menjadi sebuah kode yang tidak dapat dimengerti pihak lain (Triwinarko, 2005).

Sistem kriptografi yang cocok digunakan untuk tanda tangan digital adalah sistem kriptografi kunci-publik. Hal ini disebabkan karena skema tanda tangan digital berbasis sistem kunci-publik dapat menyelesaikan masalah *non-repudiation* (baik penerima dan pengirim pesan mempunyai pasangan kunci masing-masing) (Prabowo and Afrianto, 2017). Sistem kriptografi kunci publik mempunyai tingkat keamanan (*security level*) yang sebanding dengan jumlah kunci (bit) yang dipakai, atau dengan kata lain semakin panjang ukuran kunci maka semakin tinggi pula tingkat keamanannya (Triwinarko, 2005).

Dalam pengamanan informasi dan data ada beberapa metode, salah satu antara lain merupakan algoritma pengkodean ataupun penyandian data (Tchernykh et al., 2020). Ada pula metode algoritma pengkodean atau penyandian data yang kerap digunakan merupakan kriptografi yang dibagi jadi 2 jenis algoritma, baik simetris maupun asimetris. pengamanan informasi memakai algoritma asimetris seperti algoritma *Elliptic Curve Digital Signature Algorithm* (ECDSA) yang menggunakan 2 teknik yaitu kriptografi dan Hash . Algoritma asimetris ini mempunyai *key* yang berbeda saat proses enkripsi serta deskripsinya, yaitu *public key* dan *private key* (Alanazi, Khan and Gutub, 2021). kunci yang didistribusikan dalam algoritma asimetris merupakan *public key* yang tidak dibutuhkan kerahasiaanya, sebaliknya *private key* merupakan kunci yang tetap dijaga kerahasiaannya ataupun tidak didistribusikan. Tiap orang yang memiliki *public key* dapat melaksanakan enkripsi informasi namun hasil dari enkripsi tersebut cuma dapat dibaca oleh orang yang mempunyai kunci individu (Khan and Khan, 2018).

Pada Algoritma *Elliptic Curve Digital Signature Algorithm*, pihak yang akan melakukan tanda tangan digital, mempunyai parameter domain kurva eliptik, pasangan kunci kunci rahasia, dan kunci publik (Triwinarko, 2005). Algoritma ECC menawarkan persamaan asimetrik terdekat dengan enkripsi simetris dalam hal kinerja. ukuran kunci *ECDSA* meningkat secara linear dengan kunci enkripsi simetris dengan kinerja yang sama (seperti 2 × ukuran kunci simetris) (Doerner et al., 2019). Ukuran kunci *ECDSA* yang lebih kecil berarti bahwa enkripsi yang lebih kuat dapat dicapai dengan daya komputasi dan *bandwidth* jaringan yang lebih sedikit daripada *RSA*. ECDSA telah disahkan oleh Institut Nasional Standar dan Teknologi (NIST) AS, dan saat ini disetujui oleh Badan Keamanan Nasional AS (*NSA*) untuk perlindungan informasi rahasia dengan ukuran kunci 384 bit (setara dengan 7680- bit kunci *RSA*).

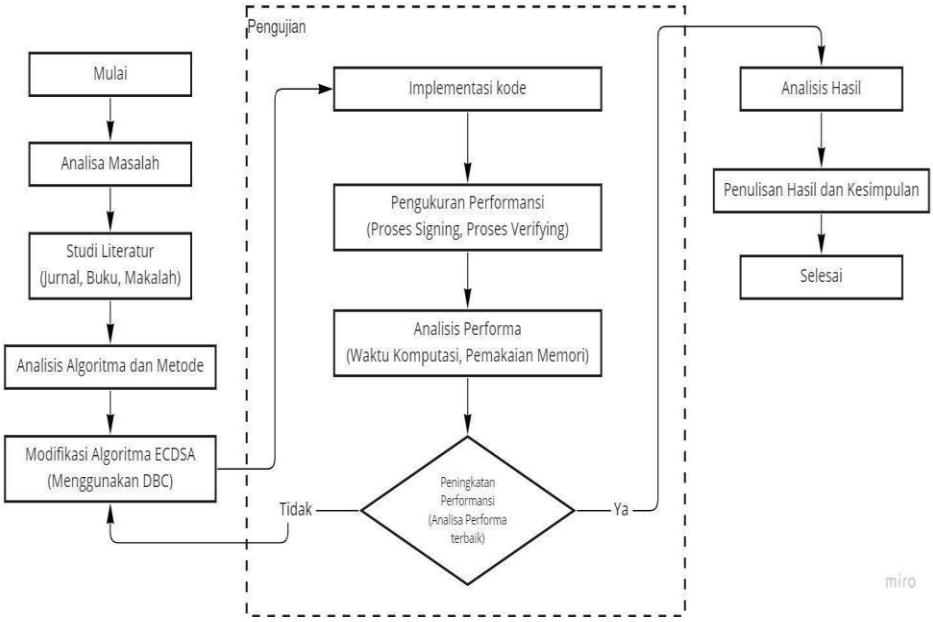
Pada saat ini penggunakan ECDSA sebagai algoritma dalam pembuatan tanda tangan digital sangat banyak dikarenakan memiliki kelebihan dari algoritma terdahulu seperti *RSA*. dapat dilihat pada gambar 1, tingkat keamanan ECDSA dengan kunci yang yang pendek menghasilkan tingkat keamanan yang tinggi, sedangkan pada RSA dengan kunci yang lebih Panjang dari ECDSA tidak jauh lebih aman dibandingkan dengan ECDSA terlihat pada gambar 1 (Triwinarko, 2005).



Gambar 1. Perbandingan Keamanan Skema ECSA dan RSA

Verifikasi tanda tangan dalam algoritma ECDSA membutuhkan 2 perkalian skalar yang membutuhkan operasi komputasi yang kompleks (Yu et al., 2020). Dalam dekade terakhir, DBC secara luas diselidiki untuk mempercepat perkalian skalar dan pemasangan kunci. Pentingnya improvisasi pada ECDSA berkaitan dengan perhitungan perkalian skalar yang mana perkalian tersebut membutuhkan waktu lebih dari 80% untuk menjalankan operasi kunci ECDSA. (Setyawan, Pinandito and Pradana, 2015).

Dengan kekurangan dan kelebihan tersebut banyak peneliti yang mengembangkan ECDSA agar menjadi lebih baik dan lebih sempurna lagi, seperti pengembangan ECDSA menggunakan metode D&A (*Double and Add*) pada jurnal (Al-Zubaidie et al., 2019) menghasilkan 21% tingkat keefektifan algoritma berdasarkan *Scalar multiplication*, selanjutnya penggunaan algoritma NAF dapat meningkatkan 13,5%, dan penggunaan metode ternary/binary dalam modifikasi 5,4%.



Gambar 2. Tahapan Penelitian

Dari permasalahan tersebut penelitian ini ingin melakukan modifikasi parameter perkalian skalar pada algoritma ECDSA sehingga dapat meningkatkan performa berupa waktu komputasi dan penggunaan memori menggunakan metode *double base chain*, dengan hipotesa awal bahwa penggunaan *double base chain* dapat meningkatkan proses pembangkitan tanda tangan, dengan modifikasi parameter perkalian skalar berdasarkan ECC dan diterapkan pada tanda tangan digital, hasil modifikasi parameter akan dibandingkan dengan algoritma ECDSA awal, hingga dari perbandingan tersebut akan mendapatkan membuktikan bahwa apakah modifikasi tersebut akan meningkatkan performa berupa waktu komputasi dan penggunaan memory pada algoritma ECDSA.

2. METODE PENELITIAN

2.1 Tahapan Penelitian

Penelitian ini terdiri dari beberapa tahapan, tahapan tersebut berupa analisis masalah, studi literatur, analisis algoritma dan metode, dan modifikasi algoritma ECDSA. Selanjutnya pada tahapan pengujian yaitu implementasi kode, pengukuran performansi, analisis performa, peningkatan performansi. Pada tahapan terakhir yaitu analisis hasil, penulisan hasil dan kesimpulan. Semua tahapan penelitian ini dapat dilihat pada gambar 2

2.2 Analisis Masalah

Tahap awal pada penelitian ini akan dimulai dengan analisis masalah, masalah yang ditemukan adalah permasalahan waktu dan performa dari algoritma ECDSA, bahwa dalam algoritma ECDSA banyak waktu yang dibutuhkan dalam perkalian skalar, sehingga ini menjadi masalah utama dalam penelitian ini.

2.3 Studi Literatur

Pada tahap ini peneliti akan mengumpulkan bahan bahan bacaan seperti jurnal, buku dan makalah agar peneliti dapat mengumpulkan informasi-informasi yang relevan dengan penelitian yang ingin dilakukan. Hal ini perlu dilakukan agar penelitian dapat berlangsung dengan terarah dan terstruktur sesuai dengan teori-teori yang didapatkan untuk menghasilkan output yang diharapkan

2.4 Analisis Algoritma dan Metode

Langkah awal dalam melakukan modifikasi atau pengembangan algoritma adalah melakukan analisis terkait dengan hal-hal yang berkaitan dengan pengembangan tersebut, yaitu :

1. Tanda tangan digital yang merupakan skema matematis yang digunakan untuk membuktikan keaslian pesan atau dokumen digital. Skema ini menjadi jaminan bahwa data dan informasi benar-benar berasal dari sumber yang benar.
2. *Elliptic Curve Digital Signature Algorithm (*ECDSA*)*, merupakan algoritma yang diterapkan dalam pembuatan tanda tangan digital yang menggunakan analogi kurva elips, yang mana ECDSA juga didasarkan dengan perhitungan *ECC* atau *Elliptic Curve Cryptography* yang mana proses ini adalah proses *enkripsi end-to-end*.
3. *Double base chain* merupakan metode yang yang memiliki dasar nilai 2 dan 3 untuk mengurangi waktu perhitungan titik pada kurva elips.

2.5 Modifikasi Algoritma ECDSA

Penggunaan *double base chain* dapat diterapkan pada proses signing yaitu perhitungan nilai penandatangan atau nilai *curva*, proses tanda tangan

itu sendiri terdapat 2 nilai yaitu nilai r dan s, untuk implementasi *double base chain* pada parameter ECDSA dapat diterapkan pada perhitungan nilai *r* dan *s*, untuk implementasi *double base chain* pada parameter ECDSA, parameter *r* dan *s* merupakan proses perhitungan pada ECDSA untuk proses tandatangan proses tersebut dapat dilihat pada gambar 3. dapat diterapkan pada perhitungan nilai

$$r = k - 1 \, mod \, n \quad (1)$$

r : nilai tanda tangan
k : meupakan *private key*
n : merupaka orde grup kurva eliptik

yang mana proses ini adalah proses sebelum dilakukan proses hashing. Menghitung nilai r merupakan salah satu perhitungan yang akan menjadi nilai dari penanda tanda tangan. Pada proses *verifying* akan terjadi perubahan perhitungan signing dikarenakan perhitungan nilai *curva* berubah, sehingga akan terjadi perubahan perhitungan (Doche and Habsieger, 2016).

2.6 Implementasi *code*

Pada tahapan ini adalah proses pembuatan kode program untuk menjalankan fungsi dari algoritma ECDSA yang sudah dimodifikasi menggunakan *double base chain*, sehingga dari proses ini dapat menghasilkan data yang akan dianalisa.

2.7 Pengukuran Performansi

Pengukuran Performansi diukur dari waktu komputasi dan penggunaan memori pada proses *signing* dan *verifying* dari algoritma yang sudah dimodifikasi menggunakan *double base chain*.

2.8 Analisis Performa

Tahapan selanjutnya adalah analisis performa algoritma yang dikembangkan dengan algoritma awal, ada 3 perbandingan performa yang akan diuji yaitu :

1. Waktu Komputasi
   Hasil algoritma yang dikembangkan akan dilakukan pengujian dengan melihat perbandingan waktu yang dibutuhkan pada proses signing dan verifying, perbandingan tersebut dibandingkan dengan algoritma awal ECDSA.
2. Penggunaan Memori
   Dari tingkat penggunaan memori apakah modifikasi algoritma ini akan membutuhkan memori yang lebih banyak dari algoritma awal ECDSA.
3. Tingkat Keamanan
   Dari tingkat keamanan apakah modifikasi algoritma ini akan menghasilkan jenis algoritma yang lebih aman diandingkan dengan algoritma sebelumnya.

3. HASIL DAN PEMBAHASAN

3.1 Identifikasi Masalah

Salah satu permasalahan dalam *elliptic curve digital signature algorithm* (ECDSA) berkaitan dengan perhitungan perkalian skalar yang mana perkalian tersebut membutuhkan waktu lebih dari 80% untuk menjalankan operasi kunci ECDSA. Penelitian ini dibuat untuk menghasilkan model algoritma ECDSA yang dapat meningkatkan performansi dan algoritma yang aman yang dapat diterapkan pada *digital signature* nantinya, dengan melakukan modifikasi menggunakan double base chain dapat meningkatkan performansi ECDSA dari sisi *key generation*, *signing*, *verifying*, penggunaan *memory*, dan tingkat keamanan kunci.

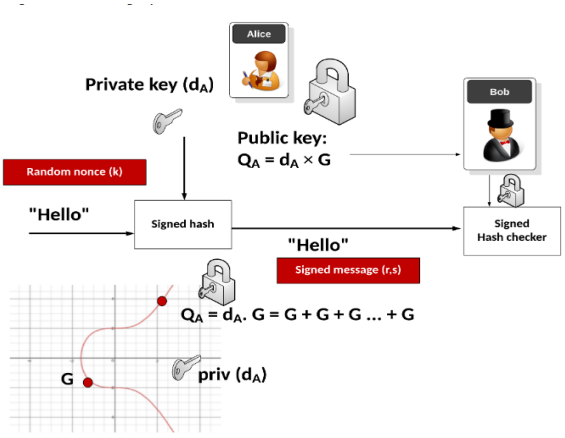
3.2 Analisis ECDSA

Elliptic curve digital signature algorithm (ECDSA) Merupakan algoritma yang diterapkan dalam pembuatan tanda tangan digital yang menggunakan analogi kurva elips, yang mana ECDSA juga didasarkan pada perhitungan *Elliptic Curve Cryptograph*y yang mana proses ini adalah proses *enkripsi end-to-end* (Pramukantoro et al., 2019). Pada *library* python *pure* ECDSA menggunakan 19 jenis kurva yang terdapat pada tabel 1.

Tabel 1 Jenis kurva ECDSA

Curve	*Length*	*Proposed By*
NIST192p	192	NIST / NSA
NIST224p	224	NIST / NSA
NIST256p	256	NIST / NSA
NIST384p	384	NIST / NSA
NIST521p	521	NIST / NSA
BRAINPOOLP160r1	160	Brainpool
BRAINPOOLP192r1	192	Brainpool
BRAINPOOLP224r1	224	Brainpool
BRAINPOOLP256r1	256	Brainpool
BRAINPOOLP320r1	320	Brainpool
BRAINPOOLP384r1	384	Brainpool
BRAINPOOLP512r1	521	Brainpool
SECP112r1	112	SECG
SECP112r2	112	SECG
SECP128r1	128	SECG
SECP160r1	160	SECG
SECP256k1	256	SECG
Ed25519	255	Others
Ed448	448	Others

Data didapatkan pada tabel 1 menggunakan *library* ECDSA dengan bahasa pemrograman python sehingga didapatkan data mentah untuk proses *signing* dan *verifikasi* menggunakan fungsi SHA-1 dan menggunakan beberapa jenis kurva yang berbeda.Data pada pengujian sebelumnya menggunakan *database* kurva standar seperti dari NIST, certicom, BSI dan yang lainnya. Pada *library* python *pure* ECDSA menggunakan 19 jenis kurva.



Gambar 3. Algorma ECDSA

Pada gambar 3 merupakan tahap pembuatan tanda tangan, pesan melalui proses hash menjadi nilai *e*. Kemudian, dibuat bilangan acak k dan dihitung titik (x^1, y^1) pada kurva eliptik sebagai hasil perkalian k dengan titik basis *G* (Zeadally et al., 2019). Nilai *r* dihitung sebagai sisa pembagian x^1 dengan orde grup *N*, dengan syarat *r* tidak boleh nol. Selanjutnya, nilai *s* dihitung menggunakan invers modular *k* dan variabel lain seperti *h* (sebagian kiri dari *e*) dan *dA* (kunci *privat*) . Jika *s* juga bukan nol, maka tanda tangan yang dihasilkan adalah pasangan (*r, s*). Pada tahap verifikasi, pesan di-hash menjadi nilai *e* seperti sebelumnya. Dalam proses ini, *c* dihitung sebagai *invers* modular *s*. Nilai u^1 dan u^2 dihitung menggunakan *c*, *h*, dan *r*. Dengan menggunakan titik basis *G* dan kunci publik *QA*, titik (x^1, y^1) dihitung sebagai hasil perkalian u1 dengan *G* dan u^2 dengan *QA*. Jika titik (x^1, y^1) bukan titik tak hingga *O*, maka tanda tangan dianggap valid jika *r* sama dengan sisa pembagian x^1 dengan orde grup *N* (Sharma, 2019).

3.3 Perancangan Modifikasi Algoritma

1. Percobaan 1

   Berdasarkan studi literatur dari algoritma ECDSA dan *double base chain*, Modifikasi ECDSA menggunakan *double base chain* dilakukan berdasarkan survey penelitian rujukan, yaitu dengan menggunakan algoritma *greedy* yang dimodifikasi dengan mengubah bentuk nilai *k* menjadi *double base number system* (DBNS), evaluasi dan pengujian tersebut dilakukan pada koordinat *x* dalam komputasi *2P+Q*, dengan mengganti menggunakan *4P+Q* dapat meningkatkan efisiensi *scalar multiplication* (SM). Kemudian dilakukan modifikasi fungsi *hash* yang berupa SHA-256 menjadi SHA-1 dengan 19 jenis kurva yang berbeda dan panjang yang berbeda,

2. Percobaan 2

   Modifikasi selanjutnya dengan penurunan ½ pada perkalian skalar secara *monotonic decreasing* (Cheng et al., 2006). Maka penggandaan titik dan pengali empat dapat diganti dengan setengah titik. Pertama-tama, kalikan *k* dengan pangkat *2* yang besar, dengan nilai, 2^q Dari hasil tersebut mendapatkan nilai 2^q untuk menjadi nilai di sekitar ukuran bidang kurva. Kemudian didapatkan nilai sisa k^0 setelah modulo ukuran bidang *p*, maka persamaan yang didapatkan berupa :

   $$k' = 2^q \; k \bmod p \qquad (1)$$

   k : nilai *base*
   q : nilai eksponen
   p : operasi modulo

Kemudian dilakukan modifikasi fungsi *hash* yang berupa *SHA*-256 menjadi *SHA*-1 dengan 19 jenis kurva yang berbeda dan panjang yang berbeda,

3.4. Hasil Evaluasi Komparasi Kinerja

1. Pengukuran *memory usage*

   Pada tahapan ini dilakukan modifikasi pada *library* ECDSA dengan menambahkan *library Psutil* yang berfungsi untuk melakukan traking berbagai *resource* serta penggunaan *memory*. Dengan menambahkan *library Psutil* setiap fungsi dalam sebuah file dapat diukur berdasarkan konsumsi *memory*. *Library* ini dilakukan dengan menambahkan fungsi dekorator dan fungsi proses *memory* ke dalam kode sehingga menghasilkan perhitungan *signing* dan *verifying* beserta konsumsi *memory*.

2. Modifikasi Hash

   Pada algoritma sebelumnya hash yang digunakan adalah *SHA*-256 sehingga *hash* tersebut harus diganti menjadi *SHA*-1.

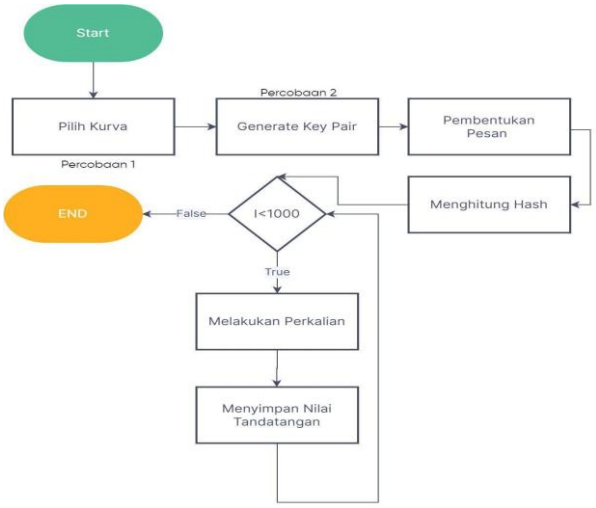
3. *Avalanche Effect Mode Blowfish*

   Avalanche Effect mengacu pada properti sistem kriptografi di mana perubahan kecil pada *input* (seperti satu bit) menghasilkan perubahan signifikan pada *output*. *Avalanche Effect* ditunjukkan dengan menerapkan fungsi kriptografi ke *plaintext* dan kemudian membuat perubahan kecil pada *plaintext* dan mengamati perubahan yang dihasilkan dalam *ciphertext* (Verma and Sharma, 2020). Sifat ini penting untuk meningkatkan keamanan algoritma kriptografi karena membuat serangan terhadap enkripsi lebih sulit, karena setiap bit informasi dari pesan atau kunci berkontribusi pada seluruh *output* (Mahindrakar, 2014).

   Blowfish adalah algoritma yang cepat dan sederhana yang menggunakan kunci dengan panjang variabel, dari 32 bit hingga 448 bit, kunci tersebut digunakan untuk mengenkripsi dan mendekripsi data.

4. Komparasi percobaan 1

   Implementasi perubahan berdasarkan metode *double base chain* percobaan 1 pada gambar 4, yang merubah perkalian pada *coordinate x* pada algoritma ECDSA pada maka kurva tersebut`dalam komputasi 2P+Q, dengan mengganti menggunakan 4P+Q dapat meningkatkan efisiensi *scalar multiplication* (*SM*)



Gambar 4 Flowchart modifikasi ECDSA

Tabel 2 Hasil percobaan 1

Curva	*Consume Memory (%)*	*Keygen Generate (%)*	*Signing Time Generate (%)*	*Verifying Time Generate (%)*
NIST192p	**0,126**	45,443	52,278	64,369
NIST224p	**0,136**	22,006	107,009	131,419
NIST256p	**0,158**	40,788	**1,599**	13,106
NIST384p	**0,225**	34,806	31,517	**1,281**
NIST521p	**0,022**	**19,749**	**8,275**	1,900
SECP256k1	0,089	71,545	**26,473**	46,789
BRAINPOOLP160r1	0,111	37,761	28,597	64,386
BRAINPOOLP192r1	0,066	116,444	18,283	10,432
BRAINPOOLP224r1	**0,066**	17,391	17,325	16,110
BRAINPOOLP256r1	**0,220**	77,739	104,146	312,643
BRAINPOOLP320r1	**0,241**	140,798	54,216	83,785
BRAINPOOLP384r1	**0,305**	822,143	201,931	114,280
BRAINPOOLP512r1	**0,303**	409,016	89,009	28,390
SECP112r1	**0,258**	28,428	33,639	58,478
SECP112r2	**0,322**	25,987	29,582	22,804
SECP128r1	**0,322**	36,800	37,310	60,256
SECP160r1	**0,301**	40,265	31,502	**6,598**
Ed25519	**0,326**	**63,873**	**49,423**	14,154
Ed448	**0,498**	11,002	**2,150**	8,124

Pada tabel 2 pada text tebal merupakan nilai modifikasi yang menghasilkan penurunan performa.

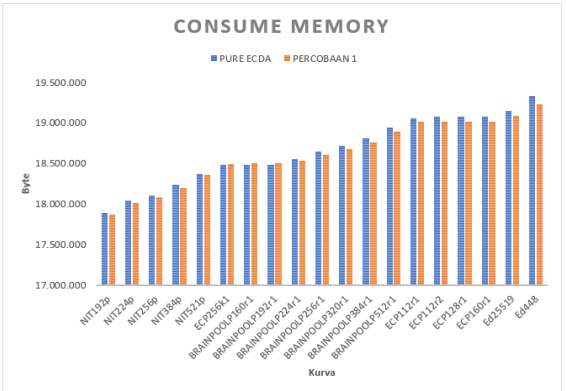
5. Komparasi percobaan 2

Implementasi Percobaan 2 dilakukan dengan penurunan ½ dan 3. perubahan tersebut menggunakan persamaan matematika $k' = 2^q \; k \; mod \; p$, modifikasi tersebut diimplementasikan pada proses *signing* di *library* ECDSA.

Tabel 3 Hasil percobaan 2

Curva	*Consume Memory (%)*	*Keygen Generate (%)*	*Signing Time Generate (%)*	*Verifying Time Generate (%)*
NIST192p	0,160	66,536	131,013	94,892
NIST224p	0,278	102,726	135,242	114,566
NIST256p	0,322	47,864	53,137	6,091
NIST384p	0,067	50,057	58,318	48,872
NIST521p	0,151	298,890	27,692	37,498
SECP256k1	0,111	80,458	57,182	105,941
BRAINPOOLP160r1	0,111	58,065	98,561	25,654
BRAINPOOLP192r1	0,150	48,889	110,942	93,754
BRAINPOOLP224r1	0,210	120,736	121,074	22,921
BRAINPOOLP256r1	0,088	124,014	97,595	227,814
BRAINPOOLP320r1	0,066	1402,331	190,096	121,297
BRAINPOOLP384r1	0,000	113,338	44,957	56,954
BRAINPOOLP512r1	0,000	371,391	90,108	54,941
SECP112r1	0,000	104,013	81,651	80,969
SECP112r2	**0,064**	58,553	606,431	76,351
SECP128r1	**0,043**	86,667	78,426	170,513
SECP160r1	**0,285**	88,469	102,564	37,317
Ed25519	**0,369**	**50,187**	**9,670**	77,475
Ed448	**0,286**	**70,746**	94,073	67,993

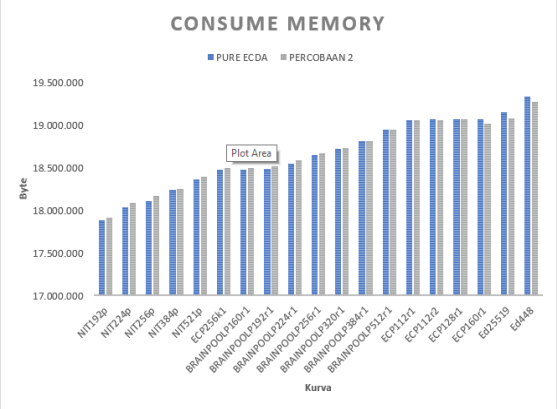
Pada tabel 3 pada text tebal merupakan nilai modifikasi yang menghasilkan penurunan performa. Pengujian dan evaluasi dilakukan untuk mendapatkan hasil dan modifikasi terbaik terhadap modifikasi algoritma ECDSA menggunakan *double base chain*. terdapat 5 pengujian yaitu *key generate*, *signing*, *verifying*, *consume memory*, dan *avalanche effect* (keacakan kunci) setiap algoritma dan kurva yang digunakan.



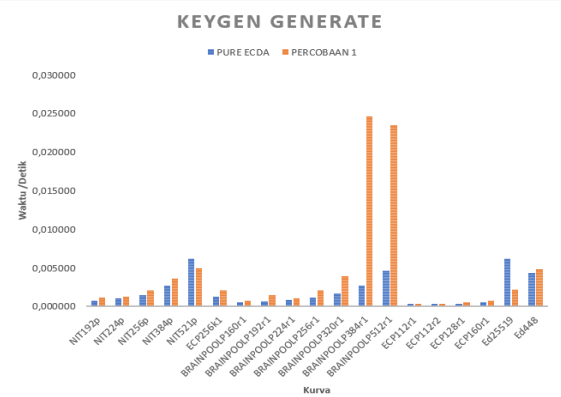
Gambar 5 Hasil performa *consume memory* percobaan 1

Tabel 4 Rata-rata perhitungan performa

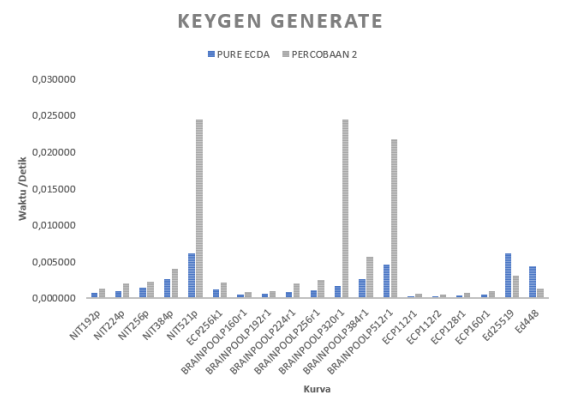
Object	ECDSA	**Percobaan 1**	**Percobaan 2**
Consume memory	18,657,280	18,621.871	18,663,208
Keygen time generate	0,001972	0,004287	0,005349
Sign time generate	0,001807	0,002468	0,003192
Verifying time generate	0,003414	0,004833	0,005674



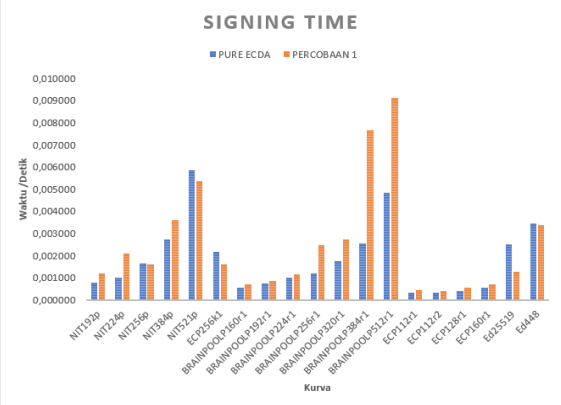
Gambar 6 Hasil Performa *consume memory* percobaan 2



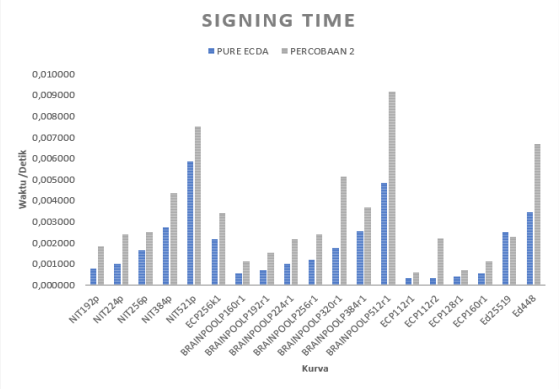
Gambar 7 Hasil Performa *keygen generate* percobaan 1



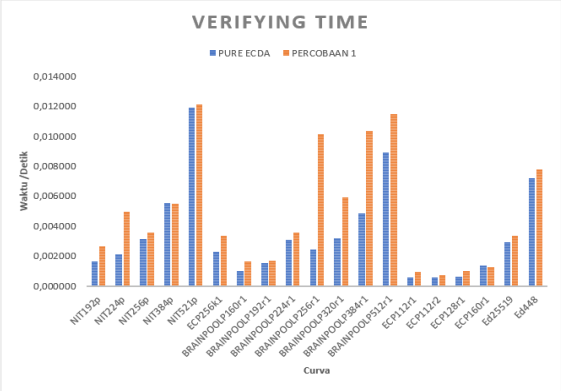
Gambar 8 Hasil Performa *keygen generate* percobaan 2



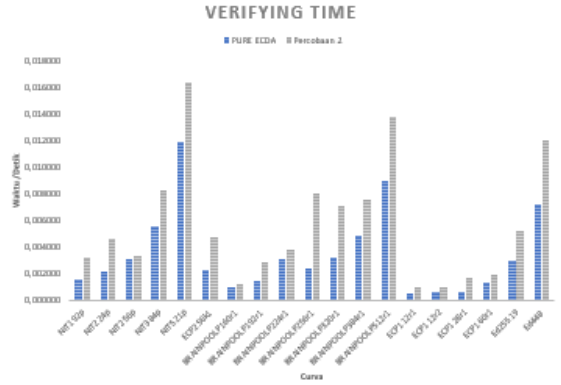
Gambar 9 Hasil Performa *signing time* percobaan 1



Gambar 10 Hasil Performa *signing time* percobaan 2



Gambar 11 Hasil Performa *verifying time* percobaan 1



Gambar 12 Hasil Performa *verifying time* percobaan 2

Dari data gambar hasil performa menunjukkan hasil pengujian performa pada beberapa jenis kurva berbeda dalam konteks ECDSA. Pengujian dilakukan dengan membandingkan *consume memory*, *key generation*, *signing time* dan *verification time* dan waktu verifikasi pada dua percobaan yang berbeda dengan *pure* ECDSA sebagai referensi.

Pada gambar 5, konsumsi memori dari tiap jenis kurva pada percobaan 1 dibandingkan dengan *pure* ECDSA. Hasil pengujian menunjukkan bahwa percobaan 1 pada modifikasi ECDSA menghasilkan *consume memory* yang lebih sedikit daripada *pure* ECDSA. Pada gambar 6 Menunjukkan bahwa percobaan 2 pada modifikasi ECDSA menghasilkan konsumsi memori yang lebih besar daripada *pure* ECDSA.

Pada gambar 7 dan 8, waktu *key generation* pada tiap jenis kurva pada percobaan 1 dan 2 dibandingkan dengan *pure* ECDSA. Hasil pengujian menunjukkan bahwa waktu *key generation* pada tiap jenis kurva pada percobaan 1 dan 2 memiliki perbedaan yang signifikan tergantung pada panjang kurva yang digunakan. Pada percobaan 1, waktu terlama terjadi pada kurva *BRAINPOOLP384r1*, sedangkan pada percobaan 2, waktu terlama terjadi pada kurva *NIST521p*.

Pada gambar 9 dan 10, *signing time* pada tiap jenis kurva pada percobaan 1 dan 2 dibandingkan dengan *pure* ECDSA. Hasil pengujian menunjukkan bahwa *signing time* pada tiap jenis kurva pada percobaan 1 dan 2 memiliki perbedaan yang signifikan tergantung pada panjang kurva yang digunakan. Pada percobaan 1 dan 2, waktu terlama terjadi pada kurva *BRAINPOOLP512r1*.

Pada gambar 10 dan 11, *verification time* pada tiap jenis kurva pada percobaan 1 dan 2 dibandingkan dengan *pure* ECDSA. Hasil pengujian menunjukkan bahwa *verification time* pada tiap jenis kurva pada percobaan 1 dan 2 memiliki perbedaan yang signifikan. Pada percobaan 1, rata-rata waktu terendah terjadi pada kurva dengan panjang 192 bit, sementara pada percobaan 2, rata-rata waktu terendah terjadi pada kurva dengan panjang 256 bit.

Secara keseluruhan, hasil pengujian menunjukkan bahwa modifikasi ECDSA tidak selalu menghasilkan performa yang lebih baik dibandingkan dengan *pure* ECDSA dapat dilihat pada tabel 4 Terdapat variasi performa yang signifikan tergantung pada jenis kurva dan konteks pengujian.

Tabel 5 Hasil evaluasi model keamanan *avalanche effect*

NO	*Curva*	***Pure*** ECDSA	**Percobaan 1**	**Percobaan 2**
1	NIST192p	0,014732	0,013393	0,0125
2	NIST224p	0,015179	0,014732	0,015625
3	BRAINPOOLP160r l	0,0125	0,015625	0,013839
4	BRAINPOOLP192rl	0,012946	0,016964	0,016071
5	BRAINPOOLP 224r l	0,011607	0,016964	0,013839
6	SECP1 12r l	0,014732	0,0125	0,016964
7	SECP1 12r2	0,011161	0,016071	0,012946
8	SECP128r l	0,012054	0,012946	0,013839
9	SECP160r l	0,012054	0,013839	0,017411

Pengujian model keamanan dilakukan dengan menghitung keteracakan kunci dengan algoritma *avalanche effect mode blowfish* dengan menghitung *Cipher-Block Chaining Mode* (CBC) dalam setiap *private key* pada 3 jenis modifikasi ECDSA terdapat pada tabel 5, kurva yang digunakan adalah kurva dengan panjang 32 - 448bit didapatkan hasil modifikasi algoritma ECDSA meningkatkan tingkat keteracakan kunci *private*.

Secara keseluruhan, hasil pengujian menunjukkan bahwa modifikasi ECDSA tidak selalu menghasilkan performa yang lebih baik dibandingkan dengan pure ECDSA dapat dilihat pada tabel 6. Terdapat variasi performa yang signifikan tergantung pada jenis kurva dan konteks pengujian.

Tabel 6 Rata-rata perhitungan performa

No	**Objek**	***Pure* ECDSA**	**Percobaan 1**	**Percobaan 2**
1	*Consume memory*	18.657.280/ byte	18.621.871/ byte	18.663.208 /byte
2	*Keygen time generate*	0,001972/s	0,004287/s	0,005349/s
3	*Sign time generate*	0,001807/s	0,002468	0,003192
4	*Verifying time generate*	0,003414/s	0,004833/s	0,005674/s

4. KESIMPULAN

Modifikasi algoritma ECDSA dengan *double base chain* (DBC) dapat efektif meningkatkan tingkat keamanan pada *private key*, tetapi belum dapat meningkatkan performa ECDSA secara keseluruhan. Evaluasi hasil penelitian menunjukkan bahwa penggunaan DBC dalam ECDSA menghasilkan peningkatan waktu eksekusi pada tahap *key generation, signing time, dan verification time*. Selain itu, penggunaan memori juga bervariasi tergantung pada konfigurasi dan parameter yang digunakan.

Namun, penelitian ini juga mempertimbangkan pentingnya faktor-faktor seperti struktur kurva, panjang kurva, parameter kurva, serta implementasi algoritma dan perangkat kriptografi yang digunakan dalam pengembangan lebih lanjut. Penggunaan DBC pada ECDSA dapat dipertimbangkan jika keamanan *private key* menjadi prioritas, tetapi perlu dipertimbangkan bahwa ini dapat memengaruhi performa algoritma secara keseluruhan.

DAFTAR PUSTAKA

ALANAZI, N., KHAN, E. AND GUTUB, A., 2021. Efficient security and capacity techniques for Arabic text steganography via engaging Unicode standard encoding. Multimedia Tools and Applications, 80(1), pp.1403–1431. https://doi.org/10.1007/s11042-020-09667-y.

AL-ZUBAIDIE, M., ZHANG, Z. AND ZHANG, J., 2019. Efficient and secure ECDSA algorithm and its applications: A survey. International Journal of Communication Networks and Information Security, https://doi.org/10.17762/ijcnis.v11i1.3827.

CAHYADI, T.N., 2020. Aspek Hukum Pemanfaatan Digital Signature Dalam. Jurnal Rechtsvinding, 9, pp.219–236.

CHENG, L.M., WONG, K.W., LEE, E.C.W. AND LIAO, X., 2006. Fast Elliptic Scalar Multiplication using New Double-base Chain and Point Halving. Image integrity and copyright protection View project ITS Project on Smart Card View project Fast Elliptic Scalar Multiplication using New Double-base Chain and Point Halving. [online] Available at: <https://www.researchgate.net/publication/220336386>.

DOCHE, C. AND HABSIEGER, L., 2016. A Tree-Based Approach for Computing Double-Base Chains A Tree-Based Approach for Computing Double-Base Chains. In Australasian Conference on Information Security and Privacy (pp. 433-446). Springer, Berlin, Heidelberg., (June 2008), pp.433–446.

DOERNER, J., KONDI, Y., LEE, E. AND SHELAT, A., 2019. Threshold ECDSA from ECDSA Assumptions: The Multiparty Case. Proceedings - IEEE Symposium on Security and Privacy, 2019-May, pp.1051–1066. https://doi.org/10.1109/SP.2019.00024.

GENC, Y. AND AFACAN, E., 2021. Design and implementation of an efficient elliptic curve digital signature algorithm (ECDSA). 2021 IEEE International IOT, Electronics and Mechatronics Conference, IEMTRONICS 2021 - Proceedings. https://doi.org/10.1109/IEMTRONICS52119.2021.9422589.

KHAN, S. AND KHAN, R., 2018. Elgamal elliptic curve based secure communication architecture for microgrids. Energies, 11(4), pp.1–14. https://doi.org/10.3390/en11040759.

MAHINDRAKAR, M.S., 2014. Evaluation of Blowfish Algorithm based on Avalanche Effect. International Journal of Innovations in Engineering and Technology (IJIET), .

MENEZES, A. AND STEBILA, D., 2021. Challenges in Cryptography. IEEE Security and Privacy, 19(2), pp.70–73. https://doi.org/10.1109/MSEC.2021.3049730.

PRABOWO, E.C. AND AFRIANTO, I., 2017. Penerapan Digital Signature dan Kriptografi Pada Otentikasi Sertifikat Tanah Digital. Jurnal Ilmiah Komputer dan Informatika (KOMPUTA), 6(2), pp.83–90.

PRAMUKANTORO, E.S., BAKHTIAR, F.A., AJI, A.L.B. AND DEWA, D.H.P., 2019. Implementasi Mekanisme End-To-End Security pada IoT Middleware. Jurnal Teknologi Informasi dan Ilmu Komputer, 6(3), p.335. https://doi.org/10.25126/jtiik.2019631401.

SETYAWAN, G., PINANDITO, A. AND PRADANA, F., 2015. Performance Calculation of Hash Sha-1 in Embedded System Using Arduino. Jurnal Penelitian dan Pengembangan Komunikasi dan Informatika, 5(3), p.122769.

SHARMA, A.K., 2019. Design and Mathematical Structure of Cryptographic Hash Function SHA-512. 11(2), pp.41–47.

TCHERNYKH, A., BABENKO, M., CHERVYAKOV, N., MIRANDA-LOPEZ, V., AVETISYAN, A., DROZDOV, A.Y., RIVERA-RODRIGUEZ, R., RADCHENKO, G. AND DU, Z., 2020. Scalable Data Storage Design for Nonstationary IoT Environment with Adaptive Security and Reliability. IEEE Internet of Things Journal, 7(10), pp.10171–10188. https://doi.org/10.1109/JIOT.2020.2981276.

TRIWINARKO, A., 2005. Elliptic Curve Digital Signature Algorithm (ECDSA). Bandung.

VERMA, R. AND SHARMA, A.K., 2020. Cryptography: Avalanche effect of AES and RSA. International Journal of Scientific and Research Publications (IJSRP), 10(4), p.p10013. https://doi.org/10.29322/ijsrp.10.04.2020.p10013.

YU, W., MUSA, S. AL AND LI, B., 2020. Double-base chains for scalar multiplications on elliptic curves. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 12107 LNCS, pp.538–565. https://doi.org/10.1007/978-3-030-45727-3_18.

ZEADALLY, S., KUMAR, A. AND SKLAVOS, N., 2019. Cryptographic technologies and protocol standards for Internet of Things. Internet of Things, [online] (xxxx), p.100075. https://doi.org/10.1016/j.iot.2019.100075.

Halaman ini sengaja dikosongkan.