Penggunaan Algoritma *Enhance Elliptic Curve* Diffie-Hellman (E-ECDH) Untuk Keamanan Penyimpanan File Pada sistem *Multi-Tenant Cloud*

Ken Azizan
18221107
Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
18221107@std.stei.itb.ac.id

Abstrak – Penerapan teknologi cloud computing, khususnya *multi-tenant cloud*, telah memfasilitasi perusahaan dengan berbagai keuntungan. Namun, teknologi ini juga menghadirkan tantangan baru terkait keamanan data. Salah satu pendekatan untuk mengatasi tantangan ini adalah dengan mengintegrasikan algoritma keamanan yang canggih. Makalah ini membahas penggunaan *Enhanced Elliptic Curve Diffie-Hellman* (E-ECDH) dalam konteks *multi-tenant cloud* untuk meningkatkan keamanan penyimpanan *file*. E-ECDH menggabungkan pertukaran kunci rahasia dengan enkripsi dan pembuatan kode autentikasi pesan, memastikan aspek kerahasiaan, integritas, dan autentikasi data dalam sistem cloud. Penelitian ini menggunakan metode analisis pustaka untuk menjelaskan konsep *multi-tenant cloud*, *elliptic curve cryptography*, algoritma E-ECDH, serta implementasi dan manfaatnya dalam meningkatkan keamanan sistem *cloud.*

Kata Kunci – ***Multi-Tenant Cloud, Enhanced Elliptic Curve Diffie-Hellman (E-ECDH), Keamanan Data, Enkripsi, Pertukaran Kunci Rahasia***

I. PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi yang terjadi secara progresif menyebabkan banyak perusahaan melakukan transformasi digital, salah satunya adalah menggunakan layanan *cloud computing*. Layanan *cloud computing* menyediakan tempat penyimpanan data sehingga perusahaan dapat mengakses data tersebut melalui jaringan internet dan data tersimpan secara fisik pada penyedia layanan. Penggunaan *cloud computing* memberikan manfaat diantaranya adalah mengurangi biaya operasional, meningkatkan fleksibilitas & skalabilitas, meningkatkan aksesibilitas pada data, serta meningkatkan keandalan sistem dalam menanggapi permintaan [6]. Walaupun terdapat banyak manfaat yang ditawarkan oleh *cloud computing,* tetapi masih terdapat isu terhadap keamanan data ketika menggunakan layanan tersebut.

Cloud computing memiliki kerentanan pada beberapa aspek keamanan, yakni *confidentiality, integrity,* dan, *authentication* [4]. *Integrity* memastikan data bersifat akurat dan tidak diubah oleh pengguna tanpa akses. Authentication diperlukan untuk memverifikasi pengguna tertentu ketika mengakses sistem. *Integrity* menjadi sulit dikelola pada lingkungan *cloud* karena sifat *cloud* yang *multi-tenancy* dan memiliki infrastruktur terdistribusi. Di sisi lain, *cloud* yang diakses melalui internet menyebabkan *cloud* menjadi rawan terhadap *cyber attacks* seperti *brute force* atau *phishing*. Oleh karena itu, perlu dikembangakan metode untuk mengamankan data pada *cloud*. Untuk menangani aspek *integrity* dapat menggunakan *hash function* [8]. Pada aspek *confidentiality* dapat diselesaikan dengan menggunakan kriptografi untuk melakukan enkripsi pada data yang tersimpan [8]. Penggunaan *Message Authentication Code* (MAC) dapat digunakan untuk memenuhi aspek *authentication*. Pada makalah ini akan dibahas pengembangan algoritma *Elliptic Curve Diffie-Hellman* yang terintegrasi dengan enkripsi dan pembuatan MAC pada *multi-tenant cloud*. Pengembangan *Elliptic Curve Diffie-Hellman* dilakukan agar sistem *cloud* dapat memenuhi aspek *confidentiality, integrity,* dan, *authentication*.

1.2 Rumusan Masalah

Berikut merupakan rumusan masalah berdasarkan latar belakang yang dipaparkan.

1. Apa itu *multi-tenant cloud* ?
2. Apa itu *elliptic curve*?
3. Bagaimana penggunaan *elliptic curve* pada algoritma *Diffie-Hellman*?
4. Bagaimana pengembangan *elliptic curve* pada algoritma *Diffie-Hellman*
5. Bagaimana cara implementasi sistem yang diajukan?

1.3 Tujuan

Berikut adalah tujuan dari pembuatan makalah ini.

1. Mengetahui apa itu *multi-tenant cloud*.
2. Mengetahui apa itu *elliptic curve*?
3. Mengetahui penggunaan penggunaan *elliptic curve* pada algoritma *Diffie-Hellman*.
4. Mengetahui pengembangan algoritma *Elliptic Curve* Diffie-Hellman.
5. Mengetahui implementasi sistem yang diajukan.

1.4 Metodologi

Pada makalah ini, metode penelitian yang dipakai adalah analisis kajian pustaka. Penelitian ini menggunakan referensi berupa jurnal-jurnal yang relevan terkait topik bahasan.

II. PEMBAHASAN

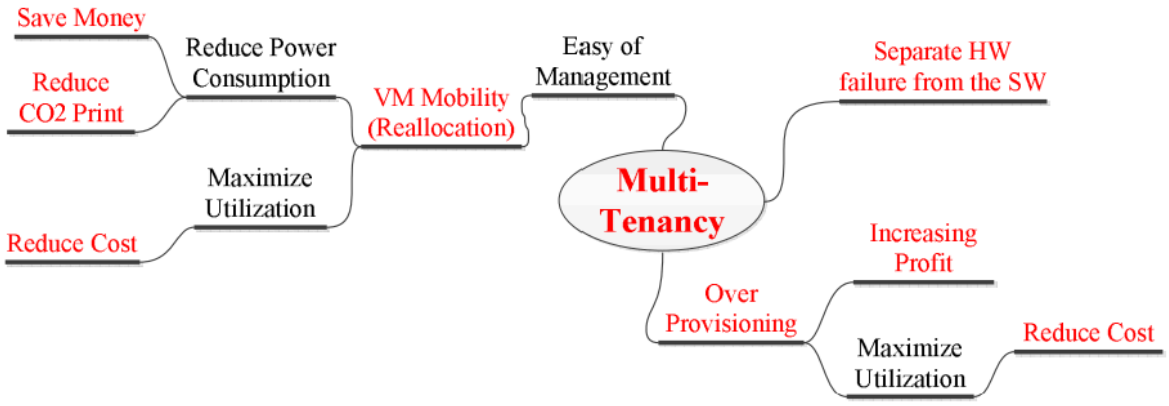
2.1 Multi-Tenant Cloud

Multi-tenant cloud mengacu pada sistem *cloud* yang menerapkan pembagian sumber daya dan setiap objek sumber daya dapat digunakan ulang pada infrastruktur *cloud* [7]. *Multi-tenant cloud* juga dapat diartikan sebagai pengguna dari organisasi yang berbeda berbagi infrastruktur perangkat keras yang digunakan oleh penyedia layanan *cloud* [9]. *Tenant* merupakan partisipan yang memiliki tampilan yang sama pada suatu aplikasi [5]. Oleh karena itu, pada sistem *cloud* dilakukan virtualisasi untuk mengisolasi

tampilan untuk setiap *tenant* [5]. Berikut merupakan tiga karakteristik *multi -tenant cloud* [5].

- Infrastruktur perangkat keras yang digunakan bersama.
- *Multi-tenant cloud* memiliki kemampuan konfigurasi yang tinggi, sehingga dapat melakukan kustomisasi untuk setiap *tenant*.
- Hanya menggunakan satu aplikasi pada setiap *runtime* yang dapat dikonfigurasi dan menggunakan satu *database* terpusat untuk seluruh *tenant*.

multi-tenant cloud merupakan hasil perkembangan teknologi untuk mendapatkan keuntungan ekonomi karena penggunaan bersama infrastruktur [7]. Penggunaan sistem *multi-tenant* memberikan manfaat untuk penyedia layanan dan pengguna. Bagi penyedia layanan, sistem *multi-tenant* memberikan kemudahan dalam pengelolaan, menghindari dari kegagalan *software* & *hardware,* serta tentunya mengurangi biaya. Pengguna mendapatkan keuntungan berupa harga layanan *cloud* yang murah karena berbagi infrastruktur dengan pengguna lainnya. Keuntungan penggunaan *multi-tenant cloud* ditunjukkan oleh Gambar 1.



Gambar 1. *Mind Map* Keuntungan *Multi-tenant cloud* [7]

Selain memiliki keuntungan, *multi-tenant cloud* tentunya memiliki kekurangan. Permasalahan utama *multi-tenant cloud* adalah keamanan data [2]. Sistem *multi-tenant cloud* sangat rentan dengan kebocoran data karena seluruh pengguna mengakses *server* yang sama dan semua data terkumpul pada satu tempat. Walaupun sudah ada kontainer virtual sebagai pemisah pada lapisan aplikasi, tetapi belum ada pemisah pada lapisan

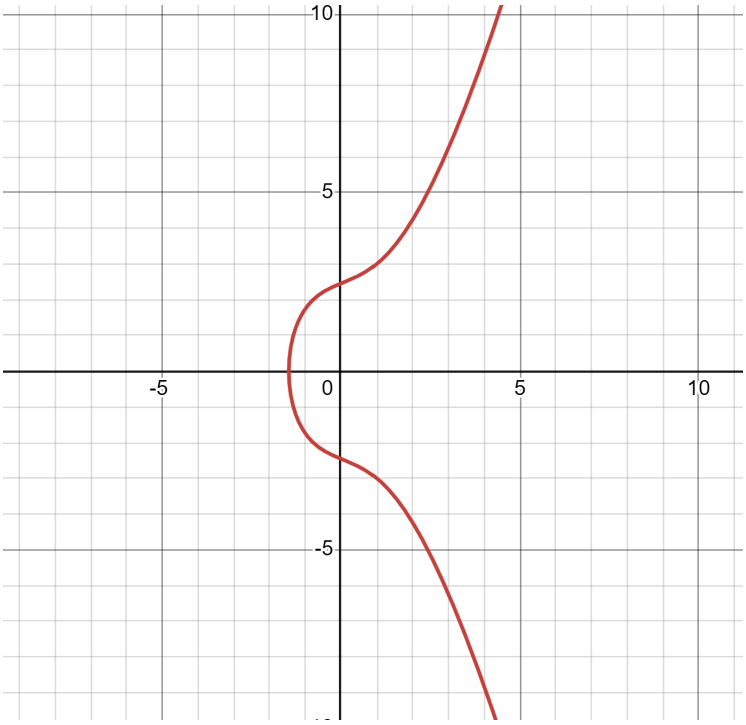
hardware [7]. Penggunaan ulang objek dari sumber daya *cloud* juga dapat menyebabkan pelanggaran pada aspek kerahasian dengan melakukan *data remanence*, ketika *tenant* membuat permintaan tempat penyimpanan kepada penyedia layanan dan melakukan pemindaian untuk mendapat data *tenant* lainnya [7].

2.2 Elliptic Curve

Elliptic curve bukan berarti kurva yang digambarkan merupakan kurva yang berbentuk elips. *Elliptic curve* yang diimplementasikan menggunakan perhitungan matematika akan menghasilkan kurva halus dengan titik O yang terdefinisi pada ketakhinggaan [1]. Berikut merupakan persamaan dari *elliptic curve*.

$$y^2 = x^3 + ax + b. \qquad (1)$$

Setiap nilai a dan b yang dimasukkan akan menciptakan bentuk kurva yang berbeda. Anggap a memiliki nilai sebesar 2 dan b memiliki nilai sebesar 6. Maka *elliptic curve* yang dihasilkan akan seperti pada Gambar 2.



Gambar 2. *Elliptic Curve*

Dalam penerapan *elliptic curve* pada kriptografi digunakan perkalian titik skalar dalam bentuk $k \cdot P$ dengan k sebagai nilai integer positif dan P sebagai titik yang berada pada *elliptic curve* [10]. Perhitungan $k \cdot P$ akan menghasilkan titik Q yang juga berada pada *elliptic curve*. Jika hanya diberikan nilai P dan Q, maka sangat sukar untuk mendapatkan nilai k sedemikian sehingga $Q = k \cdot P$ yang dikenal sebagai *Elliptic Curve Discrete Logarithm Problem* (ECDLP) bahkan sampai saat ini belum ada yang dapat memecahkannya [10]. Oleh karena itu, penggunaan *elliptic curve* sangat cocok untuk diterapkan dalam pertukaran kunci publik karena memiliki tingkat keamanan yang tinggi dan memiliki ukuran kunci yang lebih kecil daripada algoritma pertukaran kunci tradisional seperti RSA yang ditunjukan pada Tabel 1.

Tabel 1. Tabel Perbandingan Ukuran Key yang Dibangkitkan [10]

Symmetric Key	*Elliptic Curve Cryptography*	*Traditional Key Exchange*
64 bit	128 bit	700 bit
80 bit	160 bit	1024 bit
128 bit	256 bit	2048 - 3072 bit

2.3 Elliptic Curve Diffie-Hellman (ECDH)

Diffie-Hellman merupakan algoritma pertukaran kunci sehingga seluruh partisipan mendapatkan kunci rahasia bersama untuk digunakan. Namun, pertukaran kunci rahasia dilakukan pada *channel* secara langsung, sehingga lebih rentan dengan serangan *man-in-the-middle* [8]. Penggunaan dari *Elliptic Curve* Diffie-Hellman (ECDH) dapat mengatasi permasalahan tersebut dengan setiap partisipan melakukan komputasi kunci rahasia bersama pada masing-masing perangkat. Pertukaran pada algoritma ECDH yang dilakukan pada *channel* hanyalah kunci publik, sehingga dipastikan lebih aman daripada algoritma Diffie-Hellman biasa. Berikut adalah proses penggunaan algoritma ECDH. Anggap saja terdapat dua partisipan yaitu Alice dan Bob.

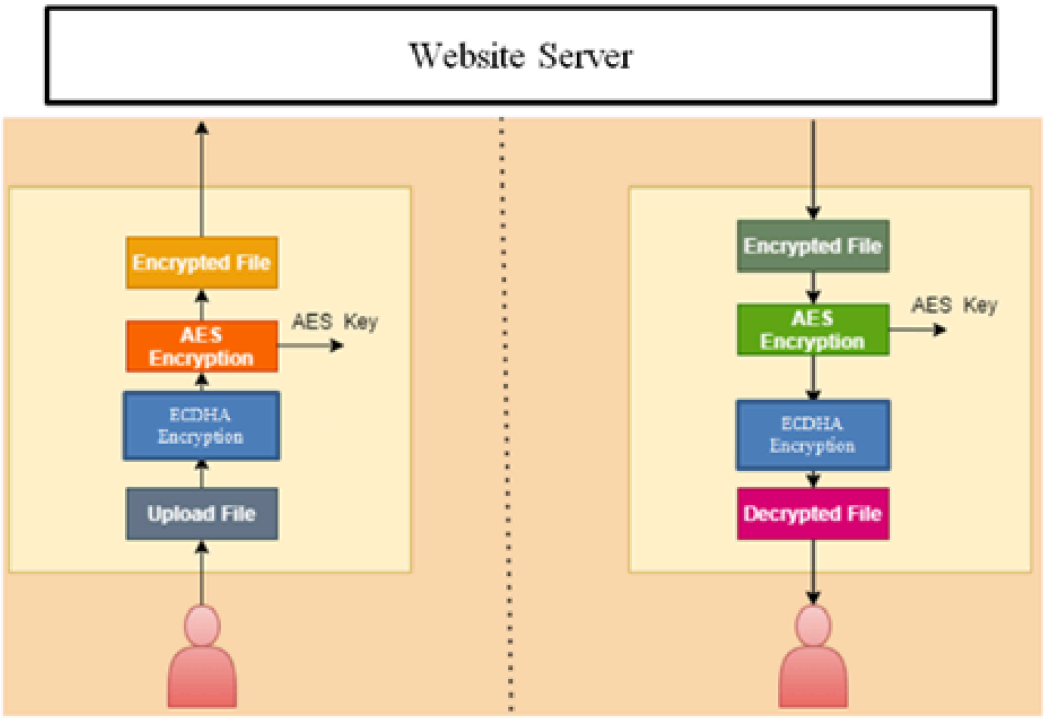
- Langkah 1: Alice dan Bob menyepakati kurva *elliptic Curve* $y^2 = x^3 + ax + b \bmod N$ dan sebuah titik Q yang berada pada kurva, kurva eliptik dan titik Q bersifat publik.
- Langkah 2: Alice dan Bob menentukan suatu nilai yang bersifat privatnya masing-masing, a sebagai kunci Alice dan b sebagai kunci privat Bob. kedua kunci privat tersebut harus berada pada $\{1, \ldots, N-1\}$.
- Langkah 3: Setiap partisipan menentukan kunci publik miliknya, untuk kunci publik Alice adalah $P_A = a \cdot Q$ dan untuk kunci publik bob adalah $P_B = b \cdot Q$, Alice dan Bob akan saling bertukar kunci publik
- Langkah 4: Alice dan Bob menghitung kunci rahasia bersama, untuk Alice bernilai $K_{AB} = a \cdot P_B = a \cdot (b \cdot Q)$ dan untuk Bob bernilai $K_{AB} = b \cdot P_A = b \cdot (A \cdot Q)$. Didapatkan kunci rahasia bersama seperti yang ditunjukan pada Persamaan 3

$$K_{AB} = a \cdot P_B = b \cdot P_A. \quad (3)$$

Namun, untuk melakukan proses enkripsi dan dekripsi pesan harus dilakukan secara terpisah menggunakan kriptografi kunci simetri seperti algoritma AES. Kunci rahasia bersama yang dibangkitkan akan digunakan sebagai kunci pada algoritma kriptografi yang dipilih [1]. Berikut rangkaian kegiatan yang dilakukan dalam melakukan enkripsi dan dekripsi *file* pada sistem *cloud* menggunakan AES sebagai algoritma kriptografi dengan pertukaran kunci ECDH.

- Perlu dilakukan pembangkitan kunci menggunakan *Elliptic Curve* Diffie Hellman (ECDH). Penggunaan algoritma ECDH memastikan bahwa kunci rahasia bersama yang akan digunakan sebagai kunci enkripsi AES tidak secara langsung dikirimkan pada suatu *channel* untuk meningkatkan keamanan.
- Pengirim dapat melakukan *upload file* pada *cloud*. *File* tersebut akan dienkripsi menggunakan algoritma AES sebelum akhirnya disimpan pada *cloud*.
- Untuk mendapatkan *file* yang tersimpan pada *cloud*, penerima perlu melakukan dekripsi AES menggunakan kunci yang dikirimkan.

Proses enkripsi dan dekripsi dengan metode kriptografi yang diajukan antara pengirim dan pengguna juga ditunjukan pada Gambar 3.



Gambar 3. Enkripsi dan Dekripsi AES dengan pembangkitan kunci Diffie-Hellman [1]

Proses enkripsi dan dekripsi yang terpisah tentunya membuat waktu yang dibutuhkan semakin lama [1]. Proses ini juga rentan dengan serangan *chosen plaintext* dan *chosen ciphertext* [3]. Selain itu, proses ini juga tidak memastikan integritas pesan yang disampaikan dari pengirim kepada pengguna. Untuk keamanan yang lebih baik algoritma ECDH perlu menghasilkan kunci yang ukurannya lebih besar sehingga kompleksitas komputasi juga semakin meningkat [3].

2.4 Enhanced Elliptic Curve Diffie-Hellman (E-ECDH)

Enhanced Elliptic Curve Diffie-Hellman (E-ECDH) merupakan pengembangan dari ECDH dengan melakukan enkripsi dan *hashing* yang dilakukan dengan *elliptic curve*. Proses enkripsi yang terintegrasi dan optimasi ukuran kunci yang dibangkitakan membuat waktu enkripsi dan dekripsi semakin menjadi lebih cepat. Penerapan *hashing* dilakukan untuk memastikan integritas data-data yang disimpan pada proses enkripsi.

E-ECDH juga menggunakan MAC (*Message Authentication Code*) untuk memastikan autentikasi pesan yang dikirim. [3]

Pada pembangkitan kunci digunakan *Key Generation Function (*KGF) yang ditujukan pada Persamaan 4 [3],

$$K = KGF(p^{ab}). \qquad (4)$$

Variabel p^a merupakan kunci publik penerima dan a merupakan kunci privat penerima. Kunci publik penerima akan dikirimkan pada pengirim. Selanjutnya Pengirim akan membangkitkan kunci privat yaitu b dan kunci publik berupa p^b khusus untuk suatu sesi. Setelah itu pengirim menggunakan KGF untuk menghasilkan kunci enkripsi dan melakukan enkripsi pesan yang akan dikirimkan $cipher = E(K \mid pesan)$. Pengirim akan mengirimkan *cipher* hasil enkripsi dan kunci publiknya kepada penerima untuk didekripsi.

Algoritma E-ECDH melakukan enkripsi data lebih efisien dengan mengurangi kompleksitas dan waktu yang dibutuhkan pada enkripsi [3]. Penggunaan *hash function* sebagai metode pengaman dari serangan eksternal. Algoritma ini juga mengoptimalkan penggunaan parameter untuk pembangkitan kunci, sehingga mengurangi biaya komunikasi dan *overhead* [3]. Proses enkripsi dan dekripsi E-ECDH diuraikan sebagai berikut.

a. Enkripsi Enhanced Elliptic Curve Diffie-Hellman (E-ECDH)

- Langkah 1: Pengirim menentukan nilai r bersifat rahasia dan berada pada $\{1, \ldots, n-1\}$.
- Langkah 2: Menentukan titik Q yang berada pada *elliptic curve*.
- Langkah 3: Mendapatkan kunci publik pengirim menggunakan Persamaan 5.

  $$K_r = r \cdot Q \quad (5)$$

- Langkah 4: Mendapatkan kunci rahasia bersama, K_R merupakan kunci publik penerima dengan Persamaan 6.

$$privateShared = (pub_a, pub_b) = r \cdot K_R \quad (6)$$

- Langkah 5: Mendapatkan kunci dari KGF untuk melakukan enkripsi dan *hashing* untuk pembuatan MAC ditunjukan pada Persamaan 7.

$$K_e || K_m = KGF(privateShared, privateShared_1) \quad (7)$$

- Langkah 6: Melakukan enkripsi pesan ditunjukan pada Persamaan 8.

$$Cipher = E(K_e : Message) \quad (8)$$

- Langkah 7: Membuat MAC ditunjukan pada Persamaan 9 menggunakan SHA-3.

$$Mac = H(Message || K_m) \quad (9)$$

- Langkah 8: kunci publik P_r, hasil enkripsi $Cipher$, dan kode autentikasi MAC dikirimkan kepada penerima.

b. Dekripsi Enhanced Elliptic Curve Diffie-Hellman (E-ECDH)

- Langkah 1: Mendapatkan kunci rahasia bersama, K_r merupakan kunci publik pengirim dengan Persamaan 10.

$$privateShared = (pub_a, pub_b) = r \cdot K_r \quad (10)$$

- Langkah 2: Mendapatkan kunci dari KGF untuk melakukan dekripsi dan *hashing* untuk pembuatan MAC ditunjukan pada Persamaan 11.

$$K_e || K_m = KGF(privateShared, privateShared_1) \quad (11)$$

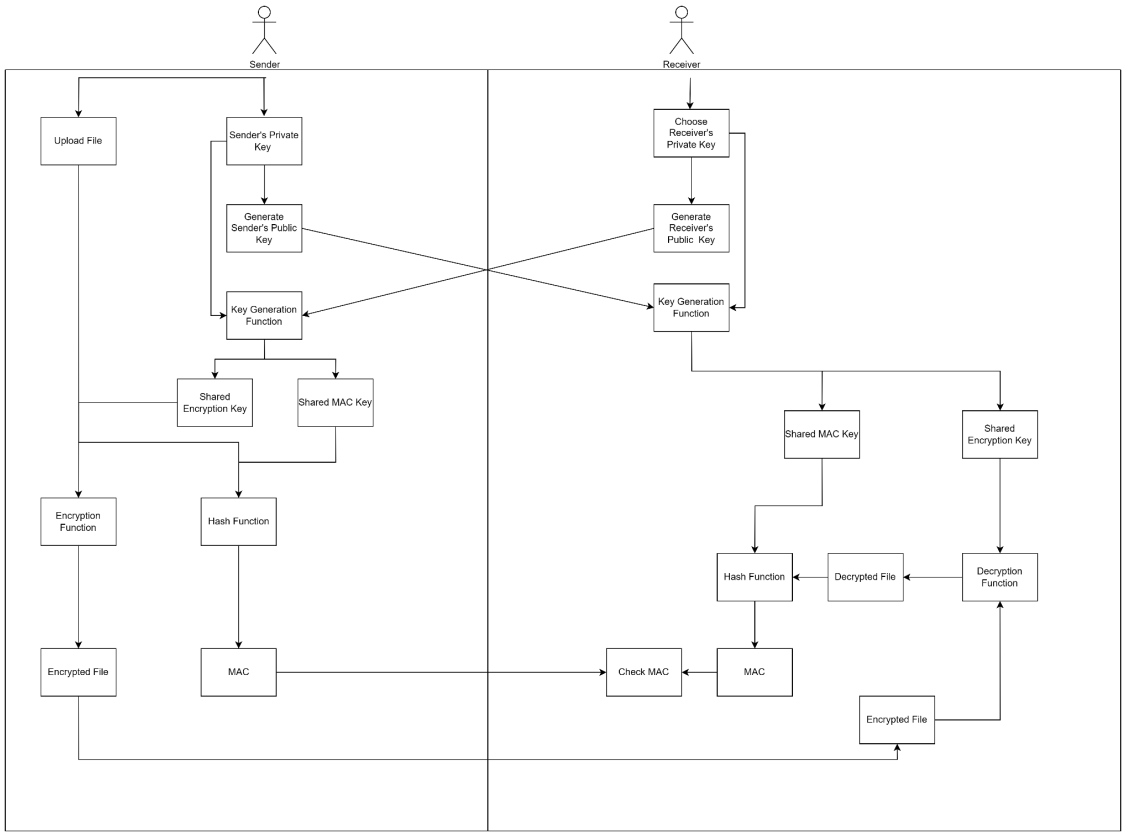
- Langkah 3: Melakukan dekripsi *cipher* yang dikirimkan menggunakan kunci yang dibangkitkan yang ditunjukkan pada Persamaan 12.

$$Plain = E(K_e : Cipher) \quad (12)$$

- Langkah 4: Melakukan pengecekan hasil dekripsi $Cipher$ yang dikirimkan dengan melakukan verifikasi MAC, apakah MAC dekripsi $Cipher$ sama dengan MAC yang dikirimkan yang ditujukan pada Persamaan 13.

$$Mac == H(Plain || K_m) \quad (13)$$

Berikut adalah alur dari algoritma *Enhanced Elliptic Curve* Diffie-Hellman(E-ECDH).



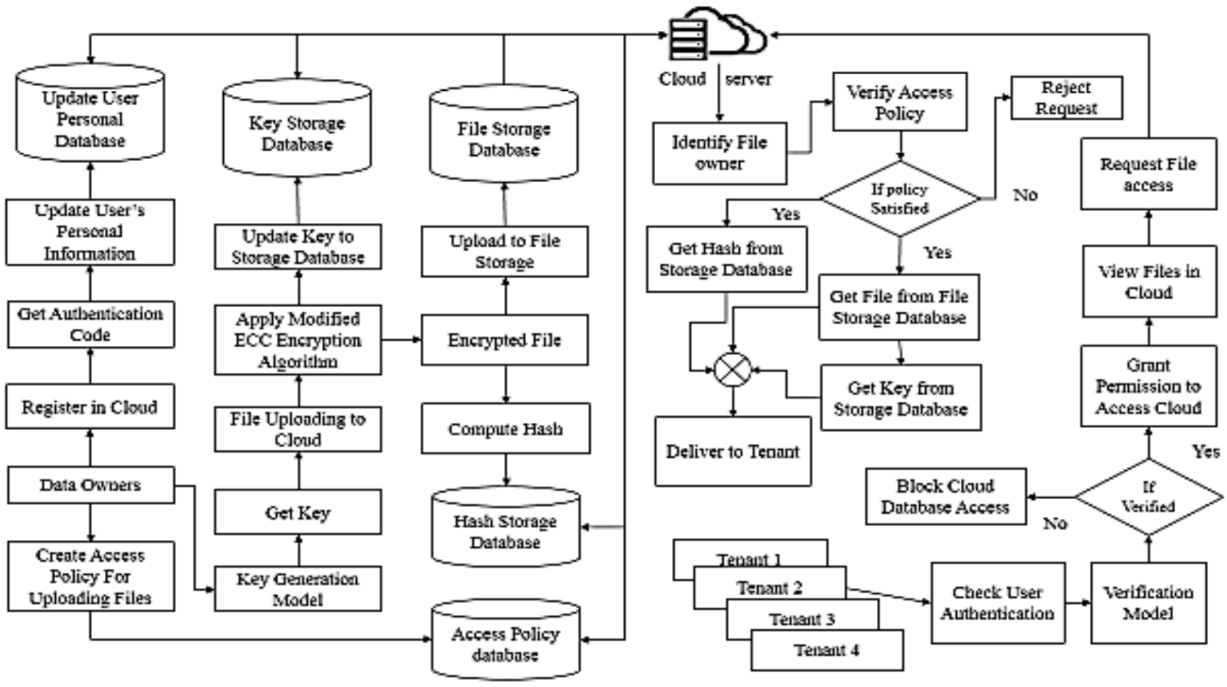
Gambar 4. Proses Enkripsi dan Dekripsi Menggunakan E-ECDH

2.5 Implementasi Sistem yang Diajukan

Penggunaan *multi-tenant cloud* memberikan dampak positif untuk penyedia layanan dan pengguna. Pengguna *cloud* mendapatkan harga yang lebih terjangkau karena biaya pengelolaan dan pemeliharaan sumber daya terbagi-bagi ke seluruh pengguna. Pengelolaan sumber daya *cloud* dapat dioptimasi menyesuaikan *demand* dari pengguna, sehingga tidak ada sumber daya yang tidak terpakai sia-sia. Akan tetapi, tantangan terbesar pada *multi-tenant cloud* adalah keamanan pada sistem.

Untuk dapat mengatasi permasalahan tersebut maka digunakan skema keamanan sistem *cloud* yang sudah pernah digunakan sebelumnya ditunjukan pada Gambar 4. Skema keamanan *cloud* ini dikembangkan untuk melakukan enkripsi dan dekripsi data

menggunakan *modified elliptic curve cryptography* dan memastikan keamanan pembangkitan dan penyaluran kunci [2]. Oleh karena itu, dengan sedikit modifikasi, skema ini dapat digunakan untuk E-ECDH yang juga merupakan pengembangan dari *elliptic curve cryptography*. Selain itu, skema keamanan *cloud* yang digunakan dapat memastikan aspek *confidentiality, integrity, dan authentication* data yang tersimpan pada *server*.



Gambar 5. *Skema* Keamanan Sistem *Cloud* yang Diajukan [2]

Data Owner dapat berupa orang ataupun perusahaan yang menyimpan banyak *file* rahasia pada *cloud*. Setiap *database* yang ada pada skema menggunakan modul komunikasi, sehingga membentuk satu kesatuan sistem. Pada skema yang digunakan hanya terdapat dua jenis *query* yang dapat digunakan oleh pengguna sistem yaitu, *read-write* dan *read*. Oleh karena itu, data-data pada *database* tidak dapat diubah. Semua *database* dikelola dengan *Database Management System* (DBMS) untuk pengelolaan sumber daya secara efektif. [2]

Data owner sebagai pengguna *cloud* terotorisasi dapat melakukan penyimpanan *file* pada sistem. *Data owner* perlu memasukkan kredensial terlebih dahulu untuk masuk ke dalam sistem *cloud*. *File* yang disimpan akan dienkripsi menggunakan E-ECDH sebelum disimpan pada *file storage database*. Setiap *tenant* memiliki kunci privat dan kunci publiknya masing-masing, begitu pula *data owner*. *Data owner dapat* mengatur akses setiap *tenant* setiap kali untuk mengunggah *file* ke *cloud server*. Pada *access policy database* disimpan kunci publik *tenant* beserta nama *file* yang dapat diakses oleh *tenant*. Jika *tenant* memiliki akses pada *file,* kunci publik *tenant* akan digunakan dalam pembangkitan kunci bersama untuk melakukan enkripsi dan pembuatan MAC. Setelah itu dilakukan enkripsi pada pesan menggunakan kunci enkripsi menjadi *cipher*. Kemudian, *cipher* akan dimasukkan ke dalam *hash function* beserta dengan kunci untuk pembuat MAC. MAC akan disimpan *hash storage database*. Kunci publik *data owner* juga disimpan pada *key storage database*. [2]

Tenant sebelum masuk ke dalam sistem *cloud* akan dilakukan pengecekan autentikasi. Jika *tenant* terverifikasi, *tenant* dapat masuk ke dalam sistem *cloud*. *Tenant* yang sudah berada di dalam sistem dapat meminta akses untuk mengunduh *file*. *Server* terlebih dahulu mengecek pemilik *file* yang diminta oleh *tenant*. Kemudian, sistem akan mengecek kebijakan akses *file*. Apabila *tenant* memiliki akses, sistem akan mengambil *file* yang terenkripsi dari *file storage database* serta MAC dari *hash storage database*. Modifikasi pada skema adalah sistem akan melakukan pembangkitan kunci rahasia bersama menggunakan kunci privat *tenant* dan kunci publik *data owner*. Kunci publik *data owner* didapatkan dari *key storage database*. *File* akan dilakukan dekripsi terlebih dahulu menggunakan kunci rahasia bersama. Setelah itu, *file* hasil dekripsi dimasukkan ke dalam *fungsi hash* untuk mendapatkan MAC. *File hasil* dekripsi, MAC yang berasal dari *database,* dan MAC hasil *hash function* dengan kunci yang dibangkitkan akan diberikan kepada *tenant*.

Algoritma E-ECDH melakukan pembangkitan kunci lebih optimal, sehingga tidak memerlukan penggunaan sumber daya pada sistem [3]. Oleh karena itu, sistem *multi-tenant cloud* dapat memfokuskan penggunaan sumber dayanya pada pekerjaan lain,

seperti pembacaan *query,* penulisan *query*, pengelolaan transaksi, dan penyelesaian *deadlock*. Penggunaan E-ECDH pada skema *multi-tenant* terbukti meringankan *workload database* pada sistem *cloud* dibandingkan algoritma lainnya [3]. Perbandingan *workload* tiap algoritma ditunjukkan pada Tabel 2.

Tabel 2. Tabel Perbandingan Hasil *Workload Database* Selama 30 Menit [3]

Functions	SHAMC2	SHAMC3	SHAMC4	Crypt-DB	MONOMI	SDB	Amazon RDS	E-ECDH
Read Queries	1.1955	1.3518	1.4212	0.7826	0.9856	0.7521	1.3996	1.4424
Write Queries	0.2152	0.2549	0.2864	0.1056	0.1685	0.1041	0.4665	0.6908
Transaksi	7.6512	7.9691	8.1251	6.0519	6.5849	6.0551	9.9882	10.284
Deadlock	1265	1143	1095	1521	1582	1438	846	795
TPS	7468	7823	8512	5125	5816	5049	10659	10856
Success Rate(%)	99.84	99.86	99.87	99.75	99.76	99.76	99.92	99.95

Penerapan algoritma E-ECDH pada skema *multi-tenant cloud* juga meningkatkan keamanan sistem. Sebelumnya, skema *multi-tenant cloud* hanya memfasilitasi kegiatan enkripsi dan *hashing* saja, sehingga hanya dapat memenuhi aspek *confidentiality* dan *integrity*. Adanya algoritma E-ECDH yang terintegrasi dengan MAC memungkinkan untuk melakukan verifikasi partisipan yang menyimpan *file* pada *cloud* ketika *tenant* mengunduh *file*. Selain itu, pada skema awal pembangkitan kunci rahasia hanya dilakukan oleh *data owner* saat menyimpan *file*. Kunci rahasia tersebut akan dikirim kepada *tenant* melalui *channel* lain, sehingga ada kemungkinan penyadapan. Pada algoritma E-ECDH, *tenant* juga melakukan pembangkitan kunci rahasia bersama menggunakan kunci privat *tenant* dan kunci publik *data owner*. Kunci publik *data owner* tetap dikirimkan ke *tenant* melalui *channel*. Hal ini, tidak menjadi masalah karena kunci yang dikirim bukan bersifat rahasia. [2]

III. Penutup

3.1 Kesimpulan

Sistem keamanan *cloud* yang diajukan memenuhi ketiga aspek kerentanan keamanan data, yakni *confidentiality, integrity,* dan *authentication*. Ketiga aspek tersebut dipenuhi oleh algoritma *Enhanced Elliptic Diffie-Hellman* (E-ECDH) yang melakukan enkripsi data dan pembuatan MAC. Selain itu, sistem keamanan *cloud* yang diajukan juga sudah memiliki sistem autentikasi tersendiri dengan memasukkan kredensial akun. Pembangkitan kunci yang lebih optimal pada algoritma E-ECDH menyebabkan sumber daya pada sistem dapat mengerjakan tugas lain, seperti pembacaan *query,* penulisan *query*, pengelolaan transaksi, dan penyelesaian *deadlock*. Oleh karena itu, *workload database* pada sistem *cloud* lebih ringan daripada penggunaan algoritma lainnya.

Referensi

[1] R. B. Deokar, “File Transfer on Cloud using Diffie-Hellman Key Exchange in Conjunction with AES Encryption,” Master’s Thesis, Dublin, National College of Ireland, 2023. [Online]. Available: https://norma.ncirl.ie/6467/

[2] Bharathiar University, Coimbatore, Tamil Nadu, S. U. Chandrika, and T. P. Perumal, “Modified ECC for Secure Data Transfer in Multi-Tenant Cloud Computing,” *Int. J. Comput. Netw. Inf. Secur.*, vol. 14, no. 6, pp. 76–88, Dec. 2022, doi: 10.5815/ijcnis.2022.06.06.

[3] A. Patil, “Enhanced-Elliptic Curve Diffie Hellman Algorithm for Secure Data Storage in Multi Cloud Environment,” *Int. J. Intell. Eng. Syst.*, vol. 11, no. 2, pp. 184–191, Apr. 2018, doi: 10.22266/ijies2018.0430.20.

[4] P. R. Kumar, P. H. Raj, and P. Jelciana, “Exploring Data Security Issues and Solutions in Cloud Computing,” *Procedia Comput. Sci.*, vol. 125, pp. 691–697, 2018, doi: https://doi.org/10.1016/j.procs.2017.12.089.

[5] I. Odun-Ayo, S. Misra, O. Abayomi-Alli, and O. Ajayi, “Cloud Multi-Tenancy: Issues and Developments,” in *Companion Proceedings of the10th International Conference on Utility and Cloud Computing*, Austin Texas USA: ACM, Dec. 2017, pp. 209–214. doi: 10.1145/3147234.3148095.

[6] A. Sether, “Cloud Computing Benefits,” *SSRN Electron. J.*, 2016, doi: 10.2139/ssrn.2781593.

[7] H. AlJahdali, A. Albatli, P. Garraghan, P. Townend, L. Lau, and J. Xu, “Multi-tenancy in Cloud Computing,” in *2014 IEEE 8th International Symposium on Service Oriented System Engineering*, Oxford, United Kingdom: IEEE, Apr. 2014, pp. 344–351. doi: 10.1109/SOSE.2014.50.

[8] P. Rewagad and Y. Pawar, “Use of Digital Signature with Diffie Hellman Key Exchange and AES Encryption Algorithm to Enhance Data Security in Cloud Computing,” in *2013 International Conference on Communication Systems and Network Technologies*, Gwalior: IEEE, Apr. 2013, pp. 437–439. doi: 10.1109/CSNT.2013.97.

[9] M. A. P. Leandro and T. J. Nascimento, “Multi-Tenancy Authorization System with Federated Identity for Cloud-Based Environments Using Shibboleth,” 2012.

[10] M. Amara and A. Siad, “ELLIPTIC CURVE CRYPTOGRAPHY AND ITS APPLICATIONS,” *Int. Workshop Syst. Signal Process. Their Appl. WOSSPA*, Jun. 2011, doi: 10.1109/WOSSPA.2011.5931464.