Pemodelan dan Analisis End-to-End Encryption pada Aplikasi Whatsapp

Felix Chandra 13523012[#1]

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13523012@std.stei.itb.ac.id
[1]mainaccfelixchandra@gmail.com

***Abstrak*— Kemajuan teknologi dalam komunikasi jarak jauh telah mempermudah kegiatan komunikasi antar manusia sehingga kita dapat berkomunikasi dimanapun dan kapanpun, namun ancaman keamanan seperti pencurian pesan pun tidak dihindari karena pesan dikirim melalui internet. Maka dari itu, untuk menjaga privasi pengguna meskipun pesan dicuri, aplikasi perpesanan instan seperti WhatsApp telah menerapkan Enkripsi End-to-End (E2EE). Kunci E2EE adalah algoritma Elliptic Curve Diffie-Hellman (ECDH), yang menukar kunci dengan aman tanpa mengirimkannya. ECDH menggunakan konsep matematika diskrit, khususnya kurva elips, untuk menghasilkan kunci simetris yang hanya dapat diakses oleh pengirim dan penerima. Enkripsi dan dekripsi lokal pada perangkat pengguna memastikan kerahasiaan di jaringan publik. Tulisan ini mengeksplorasi sistem enkripsi pesan E2EE WhatsApp, dengan fokus pada algoritma ECDH dalam pertukaran kunci. Selain itu, Makalah ini juga akan memberikan gambaran mekanisme E2EE di dalam program Python.**

***Kata Kunci*— End-to-End Encryption, Whatsapp, Elliptic Curve Diffie-Hellman (ECDH), Python.**

I. Pendahuluan

Perkembangan teknologi komunikasi telah memungkinkan manusia untuk saling bertukar informasi secara cepat dan efisien tanpa batasan jarak. Namun, seiring dengan kemudahan ini, risiko terhadap keamanan data yang dikirim melalui internet juga semakin meningkat. Salah satu ancaman utama adalah pencurian pesan oleh pihak yang tidak bertanggung jawab, seperti hacker, yang dapat mengakses informasi sensitif dengan berbagai metode. Untuk mengatasi masalah ini, aplikasi pesan instan seperti WhatsApp mengintegrasikan teknologi End-to-End Encryption (E2EE) guna melindungi privasi pengguna.

Teknologi E2EE memastikan bahwa pesan hanya dapat diakses oleh pengirim dan penerima yang dituju, tanpa ada pihak ketiga, termasuk penyedia aplikasi, yang dapat membaca isi pesan tersebut. Salah satu inti dari sistem E2EE adalah penggunaan algoritma kriptografi berbasis Elliptic Curve Diffie-Hellman (ECDH). Metode ini memanfaatkan prinsip matematika diskrit, seperti logaritma diskrit dan bilangan prima besar, untuk menghasilkan kunci enkripsi yang aman secara kriptografis. Dengan ECDH, kunci enkripsi dan dekripsi dihitung secara dinamis pada perangkat pengirim dan penerima tanpa perlu dikirimkan melalui jaringan, sehingga mengurangi risiko penyadapan.

Makalah ini membahas secara mendalam cara kerja E2EE yang diterapkan pada WhatsApp, dimulai dari proses pertukaran kunci menggunakan ECDH hingga enkripsi dan dekripsi pesan menggunakan algoritma simetris seperti AES (Advanced Encryption Standard). Selain itu, makalah ini juga memodelkan proses enkripsi dan dekripsi pesan dalam E2EE menggunakan bahasa pemrograman Python. Pemodelan ini mencakup generasi kunci publik dan privat, pertukaran kunci menggunakan ECDH, derivasi kunci enkripsi dengan fungsi hash, serta implementasi enkripsi dan dekripsi pesan menggunakan algoritma AES dengan mode operasi yang aman.

Melalui eksplorasi ini, makalah bertujuan untuk memberikan pemahaman yang lebih mendalam tentang teknologi E2EE, khususnya penerapan ECDH dalam menjamin keamanan komunikasi digital. Hasil pemodelan menunjukkan bahwa penggunaan E2EE yang benar dapat memberikan perlindungan efektif terhadap ancaman keamanan siber dalam komunikasi berbasis internet.

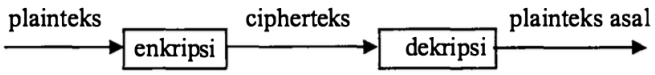
II. Dasar Teori

A. Kriptografi

Kriptografi adalah ilmu dan seni yang digunakan untuk menjaga keamanan dan kerahasiaan dari suatu pesan. Kemanan pesan diperoleh dengan menyandikannya menjadi pesan yang tidak bermakna. Kriptografi digunakan untuk menyamaekan informasi rahasia dari pihak yang tidak berhak membacanya.

Pesan awal yang ingin dirahasiakan diberi nama plaintext yang kemudian diubah menjadi ciphertext setelah melewati proses penyandian. Pesan yang disandikan dapat dikembalikan lagi ke bentuk semula atau aslinya dengan menggunakan kunci penyandian. Proses penyandian plaintext menjadi

ciphertext disebut dengan proses enkripsi sedangkan proses mengembalikan ciphertect menjadi plaintext diesbut dengan proses dekripsi.



Gambar 1. Proses Enkripsi dan Dekripsi
(Munir, Rinaldi (2005), *Matematika Diskrit* (Edisi Ketiga), Bandung: Informatika Bandung)

Sebagai contoh, diberikan sebuah pesan rahasia sebagai berikut:

Matematika Diskrit IF1220

Kemudian dilakukan proses penyandian menjadi ciphertext dengan salah satu teknik kriptografi menjadi:

b'\x08\xad\xde\xa8\x18\x14\xb7\x83\xf4)$&~\xba/\xadP\xf07
O\xd6\xf8o\xd3\xb2'

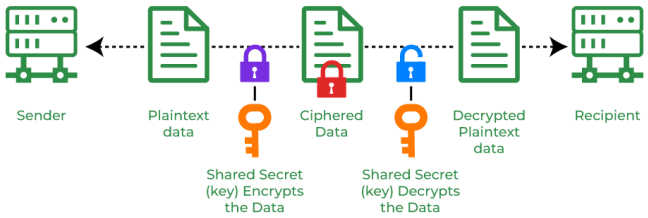
Dapat dilihat bahwa meskipun tidak dirahasiakan, ciphertext tidak dapat dimengerti oleh orang yang tidak memiliki hak untuk membaca isi pesan. Jadi hanya orang yang berhak membaca pesan tersebut yang mampu mengartikan asti dari ciphertext tersebut.

Setelah dilakukan proses dekripsi menggunakan kunci, didapatkan kembali pesan semula sebagai berikut:

Matematika Diskrit IF1220

Metode kriptografi yang umumnya digunakan terbagi atas 2 yaitu:

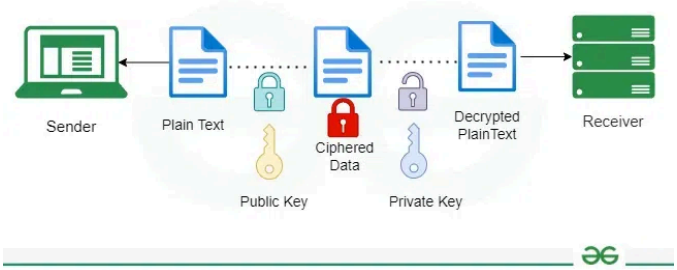
a.Symmetric Key Criptography



Gambar 2. Symmetric Key Criptography
(https://www.geeksforgeeks.org/cryptography-and-its-types/)

Metode kriptografi ini melibatkan hanya satu buah kunci untuk menenkripsi dan dekripsi pesan. Keuntugan dari metode kriptografi ini adalah cepat dan efisien sehingga cocok untuk digunakan untuk data dalam jumlah yang besar. Kekurangan dari metode ini adalah membutuhkan distribusi kunci yang mana sulit untuk diterapkan dalam skala besar. Contoh algoritma yang menerapkan metode symmetric key cryptography antara lain adalah: Advanced Encryption Standard dan Data Encryption Standard.

b. Asymmetric Key Cryptography



Gambar 3. Asymmetric Key Criptography
(https://www.geeksforgeeks.org/cryptography-and-its-types/)

Metode kriptografi in menggunakan pasagan kunci publik dan privat yang berbeda. Kunci publik ini dibagikan secara bebas. Sedangkan kunci privat dirahasiakan. Keuntungan dari kriptografi ini adalah tidak memerlukan saluran aman untuk distribusi kunci. Namun metode ini memilki kekurangan dimana Ia lebih lambat apabila dibandingkan dengan kriptografi simetris. Contoh algoritma yang menerapkan asymmetric key cryptography ini adalah RSA (Rivest-Shamir-Adleman) dan juga ECC (Elliptic Cutve Cryptography).

Dalam sistem modern seperti WhatsApp, protokol kriptografi sering menggabungkan algoritma simetris dan asimetris untuk mendapatkan manfaat dari keduanya:

- Pertukaran Kunci: Kriptografi asimetris (seperti ECDH) digunakan untuk berbagi kunci simetris secara aman.
- Enkripsi Data: Setelah kunci simetris diterima, algoritma seperti AES digunakan untuk mengenkripsi data, karena lebih cepat untuk data besar.

B. End-to-End Encryption (E2EE)

End-to-End Encryption (E2EE) adalah metode pengamanan komunikasi yang memastikan bahwa pesan hanya dapat diakses oleh pengirim dan penerima yang dituju. Dalam sistem ini, pesan dienkripsi pada perangkat pengirim dan hanya dapat didekripsi pada perangkat penerima menggunakan kunci yang unik untuk setiap komunikasi. Dengan kata lain, tidak ada pihak ketiga, termasuk penyedia layanan seperti WhatsApp, yang memiliki akses ke isi pesan, bahkan jika data tersebut melewati server mereka. E2EE menjadi standar dalam komunikasi digital modern untuk melindungi privasi pengguna dari ancaman seperti penyadapan, peretasan, atau penyalahgunaan data oleh pihak yang tidak berwenang.

Proses E2EE dimulai dengan pertukaran kunci enkripsi antara pengirim dan penerima. Dalam konteks WhatsApp, proses ini dilakukan menggunakan protokol Signal, yang mengimplementasikan algoritma Elliptic Curve Diffie-Hellman (ECDH). Setiap perangkat menghasilkan pasangan kunci privat dan kunci publik menggunakan kurva eliptik yang telah ditentukan. Kunci publik ditukar melalui server, sementara kunci privat tetap disimpan secara lokal di perangkat. Setelah pertukaran kunci, kedua perangkat secara independen menghitung shared secret menggunakan kunci privat mereka sendiri dan kunci publik dari pihak lain. Shared secret ini tidak pernah dikirimkan melalui jaringan, sehingga keamanan data tetap terjaga meskipun saluran komunikasi tidak aman.

Setelah kunci enkripsi berhasil diperoleh, pesan yang dikirimkan dienkripsi menggunakan algoritma simetris, seperti Advanced Encryption Standard (AES), dengan panjang kunci 256-bit. Enkripsi simetris ini memungkinkan pengolahan data yang lebih cepat dibandingkan algoritma asimetris, yang sangat penting untuk pengiriman pesan dalam jumlah besar secara real-time. Pesan yang telah dienkripsi dikirimkan sebagai ciphertext melalui internet. Ketika pesan ini tiba di perangkat penerima, kunci simetris yang sama digunakan untuk mendekripsi pesan kembali menjadi plaintext yang dapat dibaca.

Salah satu keunggulan utama dari sistem E2EE adalah forward secrecy, yang diterapkan melalui penggunaan kunci enkripsi yang bersifat sementara (session keys). Setiap sesi komunikasi atau bahkan setiap pesan memiliki kunci enkripsinya sendiri, yang dihasilkan secara dinamis menggunakan algoritma ECDH. Dengan demikian, meskipun salah satu kunci dalam komunikasi berhasil diretas, pesan-pesan sebelumnya tetap aman karena setiap pesan menggunakan kunci yang berbeda. Selain itu, kunci yang digunakan untuk komunikasi ini tidak disimpan dalam server WhatsApp, sehingga meminimalkan risiko kebocoran data akibat serangan pada server.

Gambar 4. Mekanisme Autentifikasi Identitas E2EE Whatsapp (https://www.eff.org/deeplinks/2016/04/whatsapp-rolls-out-end-end-encryption-its-1bn-users)

Implementasi E2EE pada WhatsApp juga mencakup mekanisme autentikasi identitas untuk mencegah serangan man-in-the-middle (MITM). Setiap perangkat memiliki identitas unik yang direpresentasikan oleh QR code. Sebelum memulai komunikasi, pengguna dapat memverifikasi QR code perangkat pasangan komunikasi mereka untuk memastikan bahwa tidak ada pihak ketiga yang menyusup dalam proses pertukaran kunci.

C. Elliptic Curve Diffie-Hellman (ECDH)

Elliptic Curve Diffie-Hellman (ECDH) adalah sebuah algoritma untuk pertukaran kunci yang aman, yang merupakan varian dari algoritma Diffie-Hellman tradisional namun menggunakan kurva eliptik dalam perhitungannya. Tujuan dari ECDH adalah memungkinkan dua pihak yang tidak memiliki saluran komunikasi aman untuk berbagi kunci secara aman yang dapat digunakan untuk enkripsi dan dekripsi pesan. Algoritma ini memanfaatkan sifat matematika dari kurva eliptik, yang memungkinkan pembuatan kunci kriptografi dengan ukuran yang lebih kecil dibandingkan dengan metode lain seperti RSA, tetapi tetap menawarkan tingkat keamanan yang setara atau bahkan lebih tinggi.

Prinsip dasar ECDH berlandaskan pada proses pertukaran kunci publik yang dilakukan oleh dua pihak yang ingin berkomunikasi. Setiap pihak memulai dengan memilih sebuah titik pada kurva eliptik yang sudah disepakati bersama dan menghasilkan pasangan kunci privat dan publik. Kunci privat adalah nilai yang hanya diketahui oleh masing-masing pihak, sementara kunci publik adalah hasil dari operasi perkalian kunci privat dengan titik pada kurva tersebut, yang dapat dibagikan secara terbuka. Meskipun kunci publik ini dapat dilihat oleh pihak ketiga, keamanan ECDH dijamin oleh kesulitan untuk menghitung kunci privat dari kunci publik yang diberikan.

Setelah kedua pihak saling bertukar kunci publik, mereka dapat menggunakan kunci privat mereka masing-masing bersama dengan kunci publik pihak lain untuk menghasilkan shared secret yang identik. Shared secret ini tidak pernah ditransmisikan melalui jaringan, melainkan dihitung secara independen oleh kedua pihak menggunakan metode matematika yang sangat efisien. Hasil akhirnya adalah kunci bersama yang digunakan untuk enkripsi simetris, seperti AES, untuk menjaga kerahasiaan pesan.

Keunggulan utama dari ECDH adalah efisiensinya, terutama dalam hal ukuran kunci. Dengan menggunakan kurva eliptik, ECDH dapat mencapai tingkat keamanan yang setara dengan algoritma Diffie-Hellman tradisional dengan menggunakan kunci yang jauh lebih pendek. Sebagai contoh, kunci 256-bit pada ECDH dianggap setara dengan kunci 3072-bit pada RSA dalam hal tingkat keamanannya. Selain itu, karena ECDH menggunakan operasi matematika yang lebih sederhana dan lebih cepat daripada algoritma berbasis bilangan besar seperti

RSA, ia lebih cocok untuk digunakan pada perangkat dengan keterbatasan sumber daya, seperti ponsel pintar.

Kurva eliptik yang digunakan dalam ECDH didefinisikan oleh persamaan:

$$y^2 = x^3 + ax + b \pmod p$$

di mana a, b, dan p adalah konstanta yang ditentukan untuk kurva tertentu, dan x serta y adalah koordinat titik pada kurva tersebut. p adalah bilangan prima yang menentukan modulus untuk operasi di dalam bidang finite, yaitu modulo p. Dalam implementasi ECDH yang umum digunakan, seperti Curve25519, kurva ini sudah ditentukan dengan nilai-nilai tetap untuk a, b, dan p.

Langkah-Langkah dalam ECDH adalah sebagai berikut:

a. Pemilihan Titik Generasi (Generator Point)
Pihak pertama dan kedua memilih sebuah titik generator G pada kurva eliptik yang sudah disepakati sebelumnya. Titik generator ini digunakan untuk menghasilkan pasangan kunci privat dan publik.

b. Pembuatan Pasangan Kunci
Kunci Privat: Pihak pertama dan pihak kedua masing-masing memilih kunci privat secara acak anggap d1 dan d2. Kunci privat ini adalah bilangan acak dalam rentang $1 <= d <= n-1$, di mana n adalah orde grup pada kurva eliptik. Kunci Publik: Kunci publik dihitung dengan mengalikan titik generator G dengan kunci privat yang dipilih. Secara matematis, kunci publik dihitung sebagai:

$$Q1 = d1 \cdot G \text{ dan } Q2 = d2 \cdot G$$

Di sini, Q1 adalah kunci publik dari pihak pertama, dan Q2 adalah kunci publik dari pihak kedua. Operasi ini disebut dengan *point multiplication* pada kurva eliptik.

c. Pertukaran Kunci Publik
Kunci publik Q1 dan Q2 ini kemudian dipertukarkan melalui saluran yang tidak aman, meskipun demikian, pesan tetap aman karena kunci privat masing-masing tetap terjaga dan tidak dibagikan.

d. Menghitung Shared Secret
Setelah pertukaran kunci publik, masing-masing pihak menggunakan kunci privatnya dan kunci publik yang diterima untuk menghitung shared secret yang identik. Ini dilakukan dengan mengalikan kunci publik pihak lain dengan kunci privat masing-masing:

Pihak pertama:

$$S1 = d1 . Q2$$

Pihak kedua:

$$S2 = d2 \cdot Q1$$

Perhatikan bahwa hasil akhirnya adalah sama dimana S1 = S2 karena operasi perkalian titik pada kurva eliptik bersifat komutatif. Dengan demikian, kedua pihak akhirnya memiliki shared secret yang sama. Shared secret S yang telah dihitung ini kemudian digunakan untuk menghasilkan kunci simetris yang akan digunakan untuk enkripsi dan dekripsi pesan antara kedua pihak. Salah satu metode yang umum digunakan untuk menghasilkan kunci simetris dari shared secret adalah dengan menggunakan fungsi hash, seperti SHA-256, untuk menghasilkan kunci dengan panjang yang sesuai dengan algoritma enkripsi simetris, seperti AES.

D. Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) adalah algoritma enkripsi simetris yang sangat digunakan dalam berbagai aplikasi keamanan modern, termasuk dalam aplikasi pesan seperti WhatsApp untuk mengamankan pesan yang dikirim antar pengguna. AES mengandalkan kunci simetris, artinya kunci yang sama digunakan untuk proses enkripsi dan dekripsi pesan. AES memiliki tiga panjang kunci yang umum digunakan: 128-bit, 192-bit, dan 256-bit. Dalam WhatsApp, AES-256 digunakan untuk memberikan tingkat keamanan yang lebih tinggi.

Enkripsi pesan whatsapp menggunakan AES mengggunakan kunci simetris yang diperoleh dari proses ECDH sebelumnya. Sebelum pesan dienkripsi, data pesan akan dibagi menjadi blok-blok berukuran 128-bit (16 byte). Setelah itu, setiap blok pesan akan diproses melalui serangkaian operasi substitusi dan permutasi yang dilakukan dalam beberapa putaran (tergantung pada panjang kunci: 10 putaran untuk AES-128, 12 putaran untuk AES-192, dan 14 putaran untuk AES-256). Proses ini dimulai dengan tahap Initial Round, diikuti oleh beberapa putaran (jumlah putaran tergantung jenis AES) yang meliputi langkah-langkah berikut:

a.SubBytes: Penggantian byte menggunakan tabel substitusi yang disebut S-box. Setiap byte dari blok diubah sesuai dengan nilai dalam S-box.

b.ShiftRows: Pergeseran baris dalam blok data, di mana byte dalam setiap baris dipindahkan ke kiri.

c.MixColumns: Operasi matrix yang melibatkan perkalian blok dengan sebuah matriks tetap untuk mencampur byte di dalam kolom.

d.AddRoundKey: Menambahkan kunci ronde saat itu ke dalam data yang sedang diproses dengan operasi XOR.
Setelah beberapa putaran ini, blok yang sudah dienkripsi akan dikirimkan melalui server.

Pada pihak penerima, proses dekripsi dilakukan dengan menggunakan kunci yang sama untuk mendapatkan kembali pesan asli. Dekripsi AES adalah kebalikan dari proses enkripsi, di mana blok yang telah dienkripsi akan diproses kembali dengan langkah-langkah yang berlawanan, termasuk substitusi, pergeseran, pencampuran kolom, dan penambahan kunci ronde, hingga pesan asli berhasil dikembalikan.

III. Pemodelan Sistem E2EE Whatsapp Menggunakan Python

A. Pembuatan Kunci Privat Dan Publik

```python
# Generate ECDH private and public keys
def generate_keys():
    private_key = ec.generate_private_key(ec.SECP256R1())
    public_key = private_key.public_key()
    return private_key, public_key
```

Gambar 5. Fungsi Pembuatan Kunci privat Dan Publik
(arsip penulis)

Menghasilkan pasangan kunci privat dan publik menggunakan metode ECDH dan kurva eliptik SECP256R1. Kunci privat digunakan secara lokal dan tidak dibagikan sedangkan kunci publik dibagikan kepada pihak lain untuk proses pertukaran kunci.

B. Serialisasi Dan Deserialisasi Kunci Publik

```python
# Serialize public key for sharing
def serialize_public_key(public_key):
    return public_key.public_bytes(
        encoding=serialization.Encoding.PEM,
        format=serialization.PublicFormat.SubjectPublicKeyInfo,
    )
# Load serialized public key
def load_public_key(serialized_public_key):
    return serialization.load_pem_public_key(serialized_public_key)
```

Gambar 6. Fungsi Serialisasi Dan Deserialisasi Kunci Publik
(arsip penulis)

Serialisasi: Mengubah kunci publik menjadi format teks (PEM) untuk memungkinkan transmisi melalui jaringan (server whatsapp).

Deserialisasi: Mengonversi kembali teks PEM menjadi objek kunci publik untuk digunakan dalam proses pertukaran kunci.

C. Pertukaran Kunci dan Derivasi Shared Secret

```python
# Derive shared secret using ECDH
def derive_shared_secret(private_key, peer_public_key):
    return private_key.exchange(ec.ECDH(), peer_public_key)
```

Gambar 7. Fungsi Derivasi Shared Secret
(arsip penulis)

Menggunakan algoritma ECDH untuk menghitung shared secret dengan mengombinasikan kunci privat pengguna lokal dan kunci publik pihak lain. Proses ECDH ini terjadi di dua perangkat pengguna.

D. Derivasi Kunci Enkripsi

```python
# Derive encryption key from shared secret
def derive_encryption_key(shared_secret):
    return HKDF(
        algorithm=SHA256(),
        length=32,
        salt=None,
        info=b"encryption-key",
    ).derive(shared_secret)
```

Gambar 8. Fungsi Derivasi Kunci Enkripsi
(arsip penulis)

Menggunakan HKDF (HMAC-based Key Derivation Function) dengan algoritma SHA-256 untuk menghasilkan kunci enkripsi simetris dari shared secret. Kunci ini akan digunakan dalam algoritma AES untuk enkripsi dan dekripsi pesan secara simetris.

E. Enkripsi dan Dekripsi Pesan

```python
# Encrypt message using AES
def encrypt_message(encryption_key, message):
    iv = urandom(16)  # Initialization Vector
    cipher = Cipher(algorithms.AES(encryption_key), modes.CFB(iv))
    encryptor = cipher.encryptor()
    ciphertext = encryptor.update(message.encode()) + encryptor.finalize()
    return iv, ciphertext
```

Gambar 9. Fungsi Enkripsi Pesan
(arsip penulis)

Menggunakan algoritma AES dengan mode CFB (Cipher Feedback). Pesan dienkripsi menggunakan kunci simetris yang dihasilkan dari ECDH. Initialization Vector (IV): Data acak yang memastikan ciphertext berbeda meskipun pesan dan kunci sama.

```python
# Decrypt message using AES
def decrypt_message(encryption_key, iv, ciphertext):
    cipher = Cipher(algorithms.AES(encryption_key), modes.CFB(iv))
    decryptor = cipher.decryptor()
    plaintext = decryptor.update(ciphertext) + decryptor.finalize()
    return plaintext.decode()
```

Gambar 10. Fungsi Dekripsi Pesan
(arsip penulis)

Menggunakan algoritma AES dan IV yang sama untuk mengembalikan ciphertext menjadi pesan asli. Dekripsi hanya berhasil jika kunci simetris benar.

F. Simulasi Proses E2EE

```python
# A generates keys
A_private_key, A_public_key = generate_keys()
# B generates keys
B_private_key, B_public_key = generate_keys()
# Serialize and share public keys
A_serialized_public_key = serialize_public_key(A_public_key)
B_serialized_public_key = serialize_public_key(B_public_key)
# Load received public keys
B_loaded_public_key = load_public_key(A_serialized_public_key)
A_loaded_public_key = load_public_key(B_serialized_public_key)
```

Gambar 11. Pembuatan, Serialisasi dan Pembagian Kunci Kriptografi
(arsip penulis)

Pengguna A dan B menghasilkan padangan kunci privat dan publik masing-masing. Setelah itu, kunci publik diserialisasi, ditransmisikan, dan di-deserialisasi oleh masing-masing pihak.

```python
# Derive shared secrets
A_shared_secret = derive_shared_secret(A_private_key, A_loaded_public_key)
B_shared_secret = derive_shared_secret(B_private_key, B_loaded_public_key)
# Derive encryption keys
A_encryption_key = derive_encryption_key(A_shared_secret)
B_encryption_key = derive_encryption_key(B_shared_secret)
```

Gambar 12. Perhitungan Shared Secret dan Derviasi kunci enkripsi
(arsip penulis)

Masing-masing pihak menghasilkan shared secret yang sama dan menderviasi kunci enkripsi.

```
# Message to encrypt
message = "IF1220 Matematika Diskrit"
# A encrypts the message
iv, ciphertext = encrypt_message(A_encryption_key, message)
```

Gambar 13. A Menenkripsi Dan Mengirim Pesan ke B
(arsip penulis)

A menulis pesan IF1220 Matematika Diskrit dan melakukan enkripsi menggunakan kunci enkripsi terlebih dahulu sebelum mengirimkan ciphertext ke perangkat B.

```
# B decrypts the message
decrypted_message = decrypt_message(B_encryption_key, iv, ciphertext)
```

Gambar 14. B Mendekripsi Pesan Ciphertext yang diterima dari A
(arsip penulis)

B mendekripsi pesan yang diterima dari A sehingga memperoleh pesan asli kembali.

```
Original Message: Matematika Diskrit
Ciphertext: b'\xb0E\xbf.uLEu\x9a\x8b\xe8\x14\xde\xb9\xbc\x80_\xc4'
Decrypted Message: Matematika Diskrit
```

Gambar 15. Hasil Simulasi Program Sistem End-to-End Encryption
(arsip penulis)

Baris pertama berisi plaintext sebelum dienkripsi yang ditulis oleh A untuk B. Baris kedua berisi ciphertext yang telah dienkripsi perangkat A dan diterima oleh B. Baris ketiga menunjukkan teks asli kembali yang didekripsi oleh perangkat B.

IV. Kesimpulan

Kriptografi adalah teknik penting untuk menjaga keamanan dan kerahasiaan data, khususnya dalam komunikasi digital. Dengan menggunakan metode enkripsi seperti symmetric key cryptography dan asymmetric key cryptography, data dapat diamankan dari pihak yang tidak berwenang. Dalam aplikasi modern seperti WhatsApp, protokol End-to-End Encryption (E2EE) menggabungkan kekuatan kedua metode ini untuk memastikan keamanan pesan dengan efisiensi tinggi. Proses ini melibatkan pertukaran kunci melalui algoritma Elliptic Curve Diffie-Hellman (ECDH) dan enkripsi pesan menggunakan Advanced Encryption Standard (AES). Implementasi ini tidak hanya melindungi privasi pengguna tetapi juga mencegah serangan seperti man-in-the-middle. Dengan terus berkembangnya teknologi kriptografi, standar keamanan seperti E2EE menjadi krusial dalam menjaga kepercayaan pengguna di era digital.

V. Lampiran

Repository Github:

https://github.com/Felix-Chandra-13523012/End-to-EndEncryption

Ucapan Terima Kasih

Pertama sekali, Penulis ingin memanjatkan puji syukur kepada Tuhan Yang Maha Esa sehingga dengan rahmat dan kuasa-Nya penulis mampu menyelesaikan makalah ini. Penulis juga ingin mengucapkan terima kasih kepada para dosen mata kuliah Matematika Diskrit IF1220 terutama dosen pengampuh kelas penulis, Bapak Rila Mandala atas bimbingannya dan pembelajaran yang diberikan sehingga penulis dapat menyelesaikan makalah ini. Penulis juga tidak lupa ingin mengucapkan terima kasih kepada kedua orangtua karena tanpa dukungan biaya dan juga mental, penulis tidak mungkin akan mampu menyelesaikan makalah ini. Terakhir penulis ingin mengucapkan banyak terima kasih kepada para pembaca yang telah mau meluangkan waktunya untuk membaca makalah yang telah dibuat oleh Penulis.

Daftar Pustaka

[1] Munir, Rinaldi (2005), *Matematika Diskrit* (Edisi Ketiga), Bandung: Informatika Bandung

[2] https://www.geeksforgeeks.org/cryptography-and-its-types/ , diakses 5 Januari 2025 pukul 16

[3] https://www.ibm.com/topics/end-to-end-encryption, diakses 5 Januari pukul 16

[4] Certicom Research, Standards for efficient cryptography, SEC 1: Elliptic Curve Cryptography, Version 2.0, May 21, 2009.

[5] https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf, diakses 5 Januari pukul 19

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis adalah hasil kerja saya sendiri, bukan hasil terjemahan karya orang lain maupun hasil plagiasi.

Bandung, 8 Januari 2025

Felix Chandra (13523012)