BAB III
METODOLOGI

3.1 Desain Penelitian

Penelitian ini menggunakan pendekatan eksperimen kuantitatif komparatif, yaitu metode yang menghasilkan data numerik terukur melalui pengujian langsung pada lingkungan yang terkontrol dan membandingkan hasil antar kondisi secara sistematis. Desain eksperimen dipilih karena memungkinkan pengukuran langsung terhadap variabel performa dan keamanan algoritma kriptografi dalam kondisi yang dapat direproduksi.

Penelitian ini membandingkan dua skema enkripsi, yaitu AES-GCM-SIV (skema yang diusulkan) dan AES-GCM (*baseline*), pada dua dimensi evaluasi. (1) performa kriptografi yang meliputi waktu enkripsi, waktu dekripsi, *throughput*, konsumsi memori, dan *overhead* ukuran data; serta (2) keamanan kriptografi yang meliputi ketahanan terhadap skenario *eavesdropping*, *tampering*, *replay attack*, *nonce reuse*, dan *Man-in-the-Middle* (MitM).

Penelitian ini diimplementasikan dalam bentuk *prototype* terdistribusi menggunakan bahasa pemrograman Rust, dijalankan dalam tiga node fisik terpisah yang saling terhubung, Alice, Bob, dan Server yang berperan ganda sebagai relay pesan antara Alice dan Bob sekaligus mensimulasikan peran penyerang (Eve) pada skenario pengujian kemanan. Simulasi dalam konteks penelitian ini merujuk pada representasi terprogram dari komunikasi dua entitas (Alice dan Bob) yang berkomunikasi melalui jaringan nyata dengan Server sebagai perantara, bukan pada pemodelan statistik atau pendekatan Monte Carlo.

Algoritma Kriptografi yang digunakan, yaitu ECDH Curve25519, HKDF-SHA256, dan AES-GCM-SIV, merupakan implementasi nyata melalui pustaka kriptografi tervalidasi dari ekosistem Rust, identik dengan yang digunakan dalam sistem produksi. Dengan demikian, hasil pengukuran yang diperoleh mencerminkan karakteristik algoritmik yang sesungguhnya dalam kondisi jaringan nyata, bukan aproksimasi atau model teoritis, dengan batasan bahwa ketiga node menggunakan spesifikasi perangkat keras yang berbeda. Perbandingan waktu eksekusi absolut lintas-node perlu mempertimbangkan variasi hardware ini, sementara perbandingan antara AES-GCM-SIV dan AES-GCM tetap valid karena kedua algoritma diuji secara bergantian pada node yang sama, sehingga hardware yang digunakan identik untuk setiap perbandingan.

3.1.1 Variabel Penelitian

Penelitian ini menetapkan tiga kategori variabel. Pertama adalah variabel bebas, yaitu faktor yang sengaja diubah untuk melihat pengaruhnya terhadap hasil pengujian. Dalam konteks ini, terdapat tiga variabel bebas, algoritma enkripsi yang digunakan (AES-GCM-SIV sebagai metode yang diusulkan dan AES-GCM sebagai pembanding), jenis data uji yang meliputi teks serta file media seperti gambar, audio, dan video, serta ukuran data yang dikelompokkan menjadi kecil, sedang, dan besar berdasarkan rentang yang telah ditentukan pada tahap pengumpulan data.

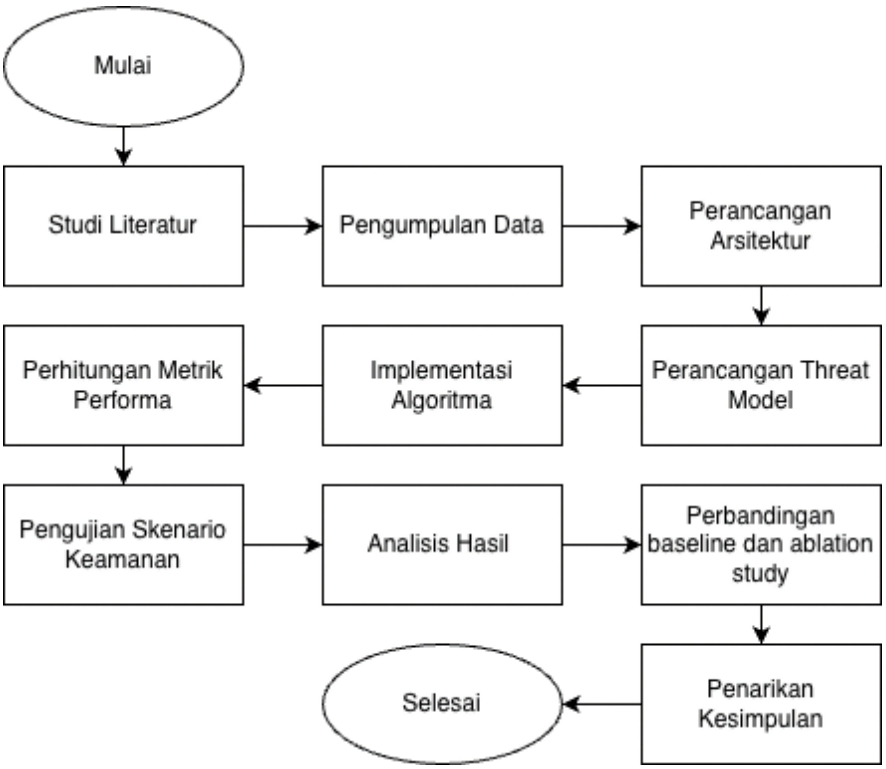
Selanjutnya, variabel terikat merupakan variabel yang diukur sebagai hasil dari eksperimen. Variabel ini mencakup waktu enkripsi dan dekripsi dalam milidetik (ms), *throughput* enkripsi dan dekripsi dalam MB/s, penggunaan memori selama

proses kriptografi dalam MB, tambahan ukuran data dalam satuan byte maupun persentase, hasil verifikasi tag autentikasi pada pengujian keamanan yang dinyatakan valid atau tidak, serta indikasi adanya kebocoran informasi pada skenario penggunaan ulang *nonce*.

Terakhir, variabel kontrol adalah faktor yang dijaga tetap konstan selama penelitian agar tidak memengaruhi hasil secara tidak terkontrol. Dalam penelitian ini, variabel kontrol meliputi spesifikasi perangkat keras yang digunakan, panjang kunci AES sebesar 256-bit, panjang *nonce* 12 byte, panjang authentication tag 16 byte, serta jumlah pengulangan pengujian sebanyak 30 kali untuk setiap kombinasi algoritma dan kategori data.

3.2 Tahapan Penelitian

Penelitian ini dilakukan melalui sepuluh tahap yang disusun secara terstruktur dan berurutan, sebagaimana diperlihatkan pada Gambar 3.1. Tahapan tersebut meliputi. (1) studi literatur, (2) pengumpulan data, (3) perancangan arsitektur sistem simulasi, (4) perancangan *threat model*, (5) implementasi algoritma kriptografi, (6) perhitungan metrik performa, (7) pengujian skenario keamanan, (8) analisis hasil, (9) perbandingan *baseline* dan *ablation study*, serta (10) penarikan kesimpulan. Setiap tahapan dirancang untuk saling mendukung sehingga hasil dari satu tahapan menjadi masukan bagi tahapan berikutnya.



Gambar III.1 Tahapan Penelitian

3.2.1 Studi Literatur

Studi literatur dilakukan untuk memperoleh landasan teoritis mengenai kriptografi modern, khususnya algoritma AES-GCM-SIV dan protokol pertukaran kunci ECDH dalam skema E2EE. Pada tahap ini juga dilakukan penelusuran terhadap jurnal internasional maupun nasional, standar kriptografi (RFC 8452 untuk AES-GCM-SIV dan RFC 7748 untuk Curve25519), serta dokumentasi teknis yang relevan.

Hasil studi literatur digunakan secara langsung sebagai dasar dalam tiga hal yang saling berkaitan. Pertama, penetapan *threat model* yang realistis berdasarkan celah keamanan yang ditemukan pada penelitian terdahulu, khususnya terkait kelemahan *nonce reuse* pada AES-GCM dan keterbatasan ECDH tanpa autentikasi kunci publik. Kedua, pemilihan parameter pengujian yang relevan dan telah

divalidasi dalam literatur, seperti jumlah iterasi 30 kali, variasi ukuran data, dan metrik performa yang digunakan. Ketiga, identifikasi celah penelitian yang menjadi justifikasi kebaruan penelitian ini, sebagaimana dipaparkan pada Tabel 2.1 dan Tabel 2.2.

3.2.2 Pengumpulan Data

Data uji yang digunakan dalam penelitian ini terdiri dari dua jenis, yaitu pesan teks dan file media, dengan variasi ukuran pada setiap kategori sebagaimana disajikan pada Tabel 3.1

Tabel III.1 Kategori Data Pengujian

Kategori	Subkategori	Ukuran
Pesan Teks	Pendek	100 Byte
	Sedang	350 Byte
	Panjang	2000 Byte
File Gambar	Kecil	76 KB
	Sedang	1 MB
	Besar	5 MB
File Audio	Kecil	700 KB
	Sedang	3 MB
	Besar	11 MB
File Video	Kecil	4 MB
	Sedang	15 MB
	Besar	33 MB

3.2.3 Arsitektur Sistem

Penelitian ini diimplementasikan pada arsitektur terdistribusi tiga entitas, Alice, Bob, dan Server, yang masing-masing berjalan sebagai proses independen pada tiga VPS (Virtual Private Server) terpisah, terhubung melalui jaringan internet publik. Topologi yang digunakan berbentuk star melalui Server, di mana seluruh komunikasi antara Alice dan Bob melewati Server sebagai relay, tanpa jalur komunikasi langsung antara kedua entitas tersebut.

Server dalam arsitektur ini menjalankan peran ganda, sebagai relay pesan terenkripsi antara Alice dan Bob, sekaligus sebagai simulasi Eve dalam pengujian keamanan (Subbab 3.2.4). Perancangan ini mencerminkan model ancaman E2EE yang realistis, di mana penyedia layanan atau infrastruktur relay tidak seharusnya perlu dipercaya untuk menjaga kerahasiaan dan integritas pesan, karena seluruh proses enkripsi dan dekripsi terjadi secara eksklusif pada endpoint Alice dan Bob. Server hanya meneruskan data terenkripsi tanpa memiliki akses terhadap kunci sesi maupun plaintext.

Alice bertindak sebagai pengirim yang melakukan enkripsi pesan dan file media, sementara Bob bertindak sebagai penerima yang melakukan dekripsi. Kedua entitas melakukan pertukaran kunci publik ECDH Curve25519 melalui Server pada fase inisialisasi sesi, kemudian menurunkan kunci sesi secara independen melalui HKDF-SHA256 tanpa pernah mengirimkan kunci privat maupun kunci sesi melalui jaringan.

Gambar 3.2 mengilustrasikan arsitektur ini, dengan garis penuh merepresentasikan pertukaran kunci publik dan garis putus-putus merepresentasikan pengiriman ciphertext.

Gambar III.2 Arsitektur Sistem Simulasi E2EE

3.2.4 *Threat Model*

Threat model mendefinisikan asumsi tentang kemampuan penyerang dan batas keamanan yang dijamin oleh sistem simulasi. Penelitian ini mengadopsi model Dolev-Yao yang dimodifikasi sebagai kerangka ancaman (Dolev & Yao, 1983). Model ini dipilih karena secara formal mengasumsikan bahwa penyerang memiliki kendali penuh atas kanal komunikasi, asumsi yang pada penelitian ini terealisasi secara langsung karena seluruh trafik antara Alice dan Bob memang diteruskan melalui Server sebagai relay, sehingga node yang sama dapat diprogram untuk berperan sebagai penyerang (Eve) pada pengujian keamanan.

Dalam model ini, penyerang diasumsikan mampu membaca, memodifikasi, menahan, menyisipkan, melakukan *replay attack*, mengeksploitasi kondisi *nonce reuse*, serta melancarkan serangan *Man-in-the-Middle* pada fase pertukaran kunci ECDH. Terhadap seluruh ancaman tersebut, sistem simulasi dirancang untuk memverifikasi lima properti keamanan secara kuantitatif, yaitu *confidentiality*, *integrity*, *authenticity*, *forward secrecy*, dan *nonce-misuse resistance*.

Batasan *threat model* ini perlu ditegaskan secara eksplisit. Penelitian ini tidak mengevaluasi ketahanan terhadap serangan pada lapisan sistem operasi, keamanan perangkat keras, maupun serangan *side-channel* terhadap implementasi kriptografi itu sendiri. Penelitian ini juga tidak mengevaluasi ancaman pada lapisan jaringan itu sendiri (misalnya ARP spoofing atau manipulasi protokol jaringan), karena peran Eve dijalankan secara terprogram pada Server, bukan melalui eksploitasi kerentanan jaringan yang sesungguhnya. Fokus evaluasi dibatasi pada ketahanan skema kriptografi terhadap ancaman-ancaman yang secara langsung relevan dengan kelima properti keamanan yang diklaim.

3.2.5 Implementasi Algoritma

Implementasi simulasi dibangun menggunakan bahasa pemrograman Rust sebagai program mandiri (standalone binary) yang mereplikasi seluruh alur kriptografi E2EE secara terprogram, mulai dari pembangkitan kunci ECDH, derivasi kunci melalui HKDF-SHA256, hingga enkripsi dan dekripsi menggunakan AES-GCM-SIV beserta algoritma pembandingnya AES-GCM. Rust dipilih karena menyediakan kontrol memori tingkat rendah tanpa garbage collector, sehingga pengukuran konsumsi memori dan waktu eksekusi lebih deterministik tanpa noise dari manajemen memori otomatis. Seluruh proses kriptografi (pembangkitan kunci, enkripsi, dekripsi) tetap dieksekusi secara lokal pada masing-masing node, sementara pertukaran kunci publik dan ciphertext dilakukan melalui jaringan LAN nyata via Server sebagai relay, sehingga hasil pengukuran waktu eksekusi mencerminkan karakteristik komputasi algoritma pada tiap perangkat, dengan tambahan latensi jaringan pada proses pengiriman data antar node. Spesifikasi ketiga node yang digunakan ditunjukkan pada Tabel 3.2.

Tabel III.2 Spesifikasi Lingkungan Pengujian Sistem

Komponen	Alice	Bob	Server
Sistem Operasi	Ubuntu 24.04 (LTS)	Ubuntu 24.04 (LTS)	Ubuntu 24.04 (LTS)
Prosesor	Intel vCPU 1 Core	vCPU 1 Core	vCPU 1 Core
RAM	2GB	2GB	2GB
Bahasa Pemrograman	Rust		
Pustaka Kriptografi	aes-gcm, aes-gcm-siv, hkdf, sha2, x25519-dalek, rand		

Pada implementasi pertukaran kunci, protokol ECDH dengan kurva Curve25519 digunakan melalui pustaka x25519-dalek untuk membangkitkan pasangan kunci secara independen di sisi masing-masing entitas simulasi. Kunci

privat tidak pernah keluar dari lingkup entitas yang bersangkutan, sedangkan kunci publik dipertukarkan melalui kanal simulasi. *Shared secret* hasil komputasi ECDH kemudian diproses menggunakan HKDF-SHA256 melalui pustaka hkdf dan sha2 untuk menghasilkan kunci simetris AES-256 beserta *nonce* dasar yang digunakan pada proses enkripsi, seluruhnya dilakukan secara lokal tanpa transmisi material kunci rahasia.

Pada implementasi enkripsi, *nonce* 12-byte yang berasal dari derivasi HKDF-SHA256 digunakan secara tetap per sesi pengujian. Data uji dienkripsi menggunakan AES-GCM-SIV hingga menghasilkan *ciphertext* beserta authentication tag 16-byte. Proses yang sama dijalankan secara paralel menggunakan AES-GCM sebagai *baseline* dengan konfigurasi kunci dan *nonce* yang identik, sehingga perbedaan hasil yang terukur dapat langsung diatribusikan pada perbedaan mode enkripsi AEAD. Proses dekripsi dilakukan dengan terlebih dahulu memverifikasi authentication tag; apabila tag tidak valid, dekripsi gagal dan data ditolak sebagai bentuk deteksi terhadap modifikasi atau pemalsuan.

3.2.6 Perhitungan Metrik

Pengukuran performa dilakukan secara langsung selama proses enkripsi dan dekripsi berlangsung dalam *pipeline* simulasi. Pengukuran waktu eksekusi menggunakan std::time::Instant yang tersedia secara native pada Rust dengan presisi hingga nanosecond dan tidak bergantung pada jam sistem. Pengukuran konsumsi memori dilakukan menggunakan pustaka stats_alloc yang menginstrumentasi global allocator Rust sebagai custom allocator wrapper, dengan mencatat selisih total bytes yang dialokasikan di *heap* sebelum dan sesudah proses

kriptografi. Pendekatan ini lebih akurat dibandingkan pengukuran berbasis OS karena mengukur alokasi pada level aplikasi secara langsung. Seluruh pengujian dilakukan sebanyak 30 iterasi per kombinasi algoritma dan kategori data untuk memperoleh nilai rata-rata yang representatif.

Parameter utama yang diukur meliputi waktu enkripsi, waktu dekripsi, *throughput*, konsumsi memori, dan *overhead* ukuran data menggunakan rumus-rumus yang telah didefinisikan pada Subbab 2.7, beserta standar deviasi waktu eksekusi sebagaimana didefinisikan pada Persamaan 7 untuk mengukur konsistensi hasil antar iterasi.

Untuk memastikan validitas hasil pengukuran, beberapa prosedur kontrol diterapkan secara konsisten. program dikompilasi dalam mode release dengan optimasi penuh; setiap kombinasi menjalankan lima iterasi pemanasan (warm-up) sebelum pengukuran dimulai; seluruh pengujian dilakukan dalam satu proses eksekusi yang sama tanpa intervensi manual; dan pengujian dilakukan pada kondisi sistem dengan beban latar belakang minimal. Prosedur ini bertujuan meminimalkan variasi dari faktor eksternal sehingga perbedaan performa yang terukur dapat diatribusikan secara valid pada perbedaan algoritma yang dibandingkan.

3.2.7 Analisis Data

Analisis data dalam penelitian ini dilakukan secara kuantitatif menggunakan pendekatan deskriptif dan komparatif terhadap data hasil pengukuran yang diperoleh dari program. Dari 30 iterasi per kombinasi algoritma dan kategori data, dihitung nilai rata-rata dan standar deviasi untuk setiap metrik performa, yaitu waktu enkripsi, waktu dekripsi, *throughput*, konsumsi memori, dan *overhead*

ukuran data. Nilai rata-rata digunakan sebagai representasi kinerja tipikal algoritma, sementara standar deviasi digunakan untuk mengukur konsistensi dan stabilitas hasil antar iterasi.

Perbandingan performa antara AES-GCM-SIV dan AES-GCM dilakukan pada setiap kombinasi jenis dan ukuran data uji. Untuk menentukan signifikansi statistik perbedaan yang terukur, digunakan uji beda berpasangan dengan tingkat kepercayaan 95% ($\alpha = 0{,}05$). Normalitas distribusi selisih diverifikasi menggunakan uji Shapiro-Wilk; apabila terpenuhi digunakan paired samples t-test, apabila tidak digunakan Wilcoxon signed-rank test. Besaran efek diukur menggunakan Cohen's d dengan interpretasi $|d| < 0{,}2$ = kecil, 0,2–0,5 = kecil-sedang, 0,5–0,8 = sedang, $> 0{,}8$ = besar. Analisis tren kinerja juga dilakukan berdasarkan variasi ukuran data untuk mengidentifikasi pola degradasi performa seiring bertambahnya beban komputasi.

Analisis tidak berhenti pada tataran deskriptif, melainkan juga mencakup interpretasi kritis dari perspektif teori kriptografi. Setiap perbedaan performa antara AES-GCM-SIV dan AES-GCM dikaitkan dengan perbedaan desain algoritma yang mendasarinya, khususnya karakteristik dua-pass pada AES-GCM-SIV dibandingkan *single-pass* pada AES-GCM, sehingga hasil empiris tidak sekadar disajikan sebagai angka melainkan dijelaskan secara teoritis.

Pada dimensi keamanan, analisis dilakukan berdasarkan hasil setiap skenario uji dengan mencatat apakah sistem berhasil mendeteksi serangan yang disimulasikan melalui kegagalan verifikasi authentication tag atau ketidakcocokan *plaintext*. Hasil *ablation study* dianalisis secara terpisah dengan membandingkan

konfigurasi penuh (ECDH + HKDF + AES-GCM-SIV) terhadap tiga konfigurasi yang dilepas satu komponennya untuk membuktikan secara empiris bahwa setiap komponen memberikan kontribusi yang tidak dapat dieliminasi tanpa mengorbankan keamanan atau performa sistem.

3.2.8 Skenario Uji

Pengujian keamanan dilakukan untuk membuktikan secara empiris bahwa skema E2EE yang diimplementasikan memenuhi properti keamanan yang diklaim berdasarkan *threat model* pada Subbab 3.2.4. Lima skenario uji dirancang dan dieksekusi dengan memanfaatkan peran ganda Server sebagai Eve, yang mencegat, memodifikasi, atau memanipulasi trafik nyata antara Alice dan Bob melalui jaringan, kemudian mengamati respons sistem. Setiap skenario dijalankan pada kedua algoritma secara paralel sehingga perbedaan respons antara AES-GCM-SIV dan AES-GCM dapat dibandingkan secara langsung.

Skenario pertama menguji properti *confidentiality* terhadap ancaman *eavesdropping*, dengan mencoba mendekripsi *ciphertext* menggunakan kunci yang salah. Keberhasilan sistem ditandai dengan kegagalan dekripsi dan ketidakmampuan penyerang merekonstruksi *plaintext*.

Skenario kedua menguji properti *integrity* terhadap ancaman *tampering*, dengan memodifikasi satu atau lebih bit pada *ciphertext* sebelum dekripsi. Keberhasilan sistem ditandai dengan ditolaknya seluruh *ciphertext* yang telah dimodifikasi melalui kegagalan verifikasi authentication tag, bahkan untuk modifikasi sekecil satu bit.

Skenario ketiga menguji ketahanan terhadap *replay attack*, dengan mengirimkan kembali *ciphertext* valid pada konteks sesi yang berbeda. Sistem mendeteksi dan menolak pesan yang dikirim ulang melalui pengecekan penomoran pesan atau *timestamp* yang disisipkan sebagai *associated data* pada proses enkripsi AEAD.

Skenario keempat merupakan pengujian utama yang membedakan AES-GCM-SIV dari AES-GCM, yaitu ketahanan terhadap *nonce reuse*. *Nonce* yang sama digunakan secara sengaja untuk mengenkripsi dua pesan berbeda pada kedua algoritma. Pada AES-GCM, kondisi ini memungkinkan rekonstruksi *plaintext* melalui operasi XOR antar *ciphertext*. Pada AES-GCM-SIV, mekanisme IV sintetis menghasilkan *ciphertext* yang berbeda meskipun *nonce* identik sehingga eksploitasi tidak dapat dilakukan, membuktikan properti *nonce-misuse resistance* secara empiris.

Skenario kelima menguji ketahanan terhadap serangan *Man-in-the-Middle* (MitM) pada fase pertukaran kunci, dengan mensimulasikan substitusi kunci publik ECDH oleh penyerang secara programatik. Substitusi ini menyebabkan *shared secret* yang dihasilkan Alice dan Bob tidak identik, sehingga kunci sesi yang diderivasi berbeda dan dekripsi gagal melalui kegagalan verifikasi authentication tag.

3.2.9 Perbandingan *Baseline* dan *Ablation Study*

Untuk menunjukkan posisi keunggulan dan *trade-off* skema yang diusulkan, *ablation study* dilaksanakan dengan pendekatan kumulatif, menambahkan satu komponen kriptografi secara bertahap pada setiap konfigurasi untuk mengisolasi

kontribusi masing-masing komponen terhadap performa dan keamanan sistem. Empat konfigurasi diuji secara paralel pada rangkaian pengujian performa dan kelima skenario keamanan yang sama.

Konfigurasi pertama (AES-GCM) merupakan *baseline*, menggunakan AES-256-GCM standar dengan kunci simetris statis yang dibagikan langsung antara Alice dan Bob tanpa mekanisme pertukaran kunci. Konfigurasi kedua (AES-GCM-SIV) menggantikan AES-GCM dengan AES-256-GCM-SIV, dengan seluruh komponen lain tetap sama, mengisolasi kontribusi mode AEAD Synthetic IV terhadap performa dan *nonce-misuse resistance*. Konfigurasi ketiga (AES-GCM-SIV + ECDH) menambahkan pertukaran kunci ECDH Curve25519, menggunakan shared secret hasil ECDH langsung sebagai kunci sesi tanpa derivasi HKDF, mengisolasi kontribusi ECDH terhadap *authenticity* dan *forward secrecy*. Konfigurasi keempat (AES-GCM-SIV + ECDH + HKDF) merupakan skema lengkap yang diusulkan, menambahkan derivasi kunci sesi melalui HKDF-SHA256 di atas shared secret ECDH.

Pendekatan kumulatif ini dipilih agar kontribusi tiap komponen dapat diukur secara bertahap dan diatribusikan secara jelas, penambahan AES-GCM-SIV terhadap *baseline* mengisolasi biaya dan manfaat mode Synthetic IV, penambahan ECDH pada konfigurasi ketiga mengisolasi kontribusinya terhadap ketahanan MITM dan *forward secrecy*, dan penambahan HKDF pada konfigurasi keempat mengisolasi biaya komputasi derivasi kunci tanpa mengubah properti keamanan yang sudah diverifikasi pada konfigurasi sebelumnya. Hasil pengujian keempat

konfigurasi terhadap kelima skenario keamanan dan tiga kategori ukuran pesan teks disajikan pada Subbab 4.6 dan dianalisis pada Subbab 4.7.

3.2.10 Penarikan Kesimpulan

Tahap akhir penelitian adalah penarikan kesimpulan berdasarkan hasil analisis data secara menyeluruh. Kesimpulan disusun untuk menjawab seluruh rumusan masalah dengan mengacu pada data kuantitatif hasil pengukuran performa, hasil verifikasi skenario uji keamanan, dan hasil *ablation study*.

Kesimpulan mencakup empat aspek sesuai rumusan masalah. (1) perancangan dan implementasi *pipeline* simulasi skema E2EE yang mengintegrasikan AES-GCM-SIV, ECDH Curve25519, dan HKDF-SHA256; (2) evaluasi empiris pemenuhan properti *confidentiality*, *integrity*, *authenticity*, *forward secrecy*, dan *nonce-misuse resistance* berdasarkan lima skenario uji; (3) analisis karakteristik performa AES-GCM-SIV dibandingkan AES-GCM pada seluruh parameter dan variasi ukuran data; serta (4) evaluasi kontribusi masing-masing komponen kriptografi melalui *ablation study* beserta *trade-off* skema yang diusulkan terhadap *baseline*.

Seluruh hasil disintesiskan dan dikaitkan dengan teori kriptografi serta penelitian terdahulu pada Bab II, sehingga terdapat benang merah yang jelas antara landasan teoritis, desain metodologi, dan temuan empiris. Apabila terdapat hasil yang tidak sesuai dengan prediksi teoritis, hal tersebut dibahas secara kritis dengan mengidentifikasi faktor penyebabnya. Pada bagian akhir, penelitian ini memberikan rekomendasi untuk penelitian lanjutan, antara lain pengujian pada arsitektur prosesor yang berbeda, eksplorasi integrasi protokol Signal atau Double Ratchet

untuk skenario multi-sesi, serta evaluasi skema pasca-kuantum sebagai alternatif ECDH dalam menghadapi ancaman komputasi kuantum.