Department of Digital Business
Journal of Artificial Intelligence and Digital Business (RIGGS)
Homepage: https://journal.ilmudata.co.id/index.php/RIGGS
Vol. 5 No. 2 (2026) pp: 9601- 9608
P-ISSN: 2963-9298, e-ISSN: 2963-914X

Penerapan Enkripsi Homomorfik pada Jaringan SDN untuk Meningkatkan Keamanan Data Real-Time pada Edge Computing

Muhajirin[1], Lilik Widyawati[2], Khairan Marzuki[3]
[12]Program Studi Ilmu Komputer, Fakultas Teknik, Universitas Bumigora
[3]Program Studi Sistem Ilmu Komputer, Fakultas Teknik, Universitas Bumigora
muhajirindr@gmail.com, lilikwidyawati@universitasbumigora.ac.id , khairanmarzuki@universitasbumigora.ac.id

Abstrak

Peningkatan kebutuhan keamanan data pada jaringan modern mendorong pengembangan metode yang mampu menjaga kerahasiaan informasi tanpa mengurangi efektivitas komunikasi dan kinerja jaringan. Software Defined Networking (SDN) dan Edge Computing merupakan teknologi yang banyak digunakan untuk mendukung pengelolaan jaringan yang fleksibel serta pemrosesan data secara real-time dengan latensi rendah. Namun, penerapan kedua teknologi tersebut masih menghadapi berbagai ancaman keamanan, seperti sniffing, man-in-the-middle (MitM), dan serangan terhadap SDN controller yang berpotensi menyebabkan kebocoran informasi sensitif. Penelitian ini bertujuan untuk mengimplementasikan dan menganalisis penerapan Homomorphic Encryption pada jaringan SDN berbasis Edge Computing guna meningkatkan keamanan data real-time. Metode penelitian yang digunakan adalah Network Development Life Cycle (NDLC) yang meliputi tahapan analisis, perancangan, simulasi, implementasi, dan monitoring sistem. Implementasi dilakukan menggunakan Mininet sebagai emulator jaringan, RYU Controller sebagai pengendali SDN, serta library Microsoft SEAL/TenSEAL dengan skema enkripsi Cheon-Kim-Kim-Song (CKKS). Pengujian dilakukan dengan membandingkan skenario pengiriman data tanpa enkripsi dan dengan Homomorphic Encryption berdasarkan parameter latensi, throughput, penggunaan CPU, penggunaan memori, serta tingkat keamanan menggunakan Wireshark. Hasil penelitian menunjukkan bahwa Homomorphic Encryption mampu menjaga kerahasiaan data tanpa memerlukan proses dekripsi selama transmisi. Meskipun terjadi peningkatan latensi dan penurunan throughput akibat proses enkripsi, kinerja sistem masih berada dalam batas yang dapat diterima untuk aplikasi real-time. Selain itu, paket data terenkripsi tidak dapat dibaca oleh pihak ketiga, sehingga memberikan perlindungan yang efektif terhadap penyadapan. Integrasi SDN, Edge Computing, dan Homomorphic Encryption berpotensi menjadi solusi keamanan jaringan masa depan pada lingkungan industri.

Kata Kunci: *Software Defined Networking, Edge Computing, Homomorphic Encryption, Keamanan Data, Real-Time*

1. Latar Belakang

Kelancaran pertukaran informasi digital dan efisiensi infrastruktur jaringan menjadi faktor penting dalam mendukung operasional organisasi [1]. Perkembangan transformasi digital menuntut sistem yang mampu menangani transmisi data secara real-time dengan tingkat keamanan yang tinggi [2]. Namun, pertukaran data sensitif masih menghadapi berbagai ancaman, seperti *sniffing*, *man-in-the-middle* (MitM), serta risiko kebocoran informasi akibat tingginya lalu lintas data dan pengelolaan jaringan yang kurang adaptif [3],[4]. Untuk mengatasi permasalahan tersebut, Software-Defined Networking (SDN) menawarkan pengelolaan jaringan yang lebih fleksibel melalui pemisahan *control plane* dan *data plane* [5], [6]. Sementara itu, Edge Computing memungkinkan pemrosesan data dilakukan lebih dekat ke sumber sehingga mampu menurunkan latensi komunikasi [2]. Meskipun demikian, lingkungan edge yang bersifat *zero-trust* tetap rentan terhadap berbagai ancaman keamanan [1], [3]. Oleh karena itu, Homomorphic Encryption (HE) menjadi salah satu solusi yang menjanjikan karena memungkinkan pemrosesan data dilakukan langsung pada ciphertext tanpa perlu proses dekripsi terlebih dahulu [7], [8].

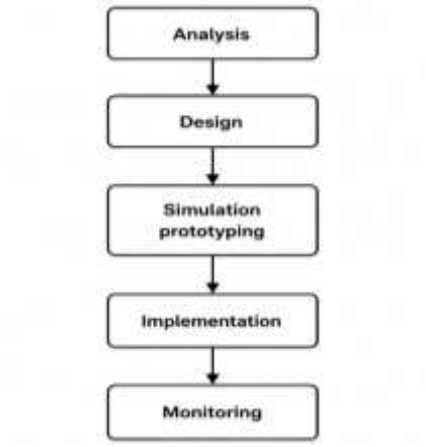
Beberapa penelitian terdahulu telah menunjukkan efektivitas penerapan Homomorphic Encryption pada berbagai bidang. Jin et al. [9] melalui FedML-HE berhasil mengurangi overhead komunikasi hingga 10 kali lipat dan menekan ukuran transmisi dari 129,75 MB menjadi 16,37 MB pada sistem pembelajaran mesin terdistribusi. Jia et al. [10] menunjukkan bahwa metode *chunk-based partitioning* mampu meningkatkan efisiensi Fully Homomorphic Encryption (FHE) dengan akurasi klasifikasi mencapai 97,1%. Pada aspek implementasi perangkat keras, Lee et al. [8] melaporkan bahwa optimasi skema CKKS menggunakan Number Theoretic Transform (NTT) mampu memproses polinomial berderajat tinggi dalam hitungan milidetik, sedangkan Kiran Kumar et al. [4]

membuktikan bahwa modifikasi cipher SLIM pada FPGA dapat menghasilkan efisiensi sumber daya yang tinggi. Di lingkungan edge, Chan et al. [11] melalui HHEML berhasil membagi proses enkripsi antara edge dan cloud sehingga mempercepat proses komputasi. Pada jaringan SDN, Gilliard dan Liu [6] melalui arsitektur CALIS mampu menurunkan waktu *handshake* mTLS dari 1403 ms menjadi 195 ms. Selain itu, Yu et al. [12] menunjukkan bahwa integrasi pembelajaran kolaboratif cloud-edge terenkripsi dapat mempercepat konvergensi pelatihan hingga 32%, sedangkan Ali et al. [13] membuktikan bahwa kombinasi HE dan blockchain mampu menjaga validitas data IoT tanpa mengungkap kunci rahasia. Penelitian lain oleh Park et al. [14] dan Bouabidi et al. [15] juga menunjukkan bahwa optimasi parameter CKKS mampu meningkatkan efisiensi inferensi dan pemrosesan data terenkripsi pada lingkungan komputasi modern.

Meskipun berbagai penelitian tersebut berhasil meningkatkan efisiensi dan keamanan data terenkripsi, sebagian besar masih berfokus pada pembelajaran mesin, layanan cloud, IoT, maupun optimasi perangkat keras. Implementasi Homomorphic Encryption pada jaringan SDN yang terintegrasi dengan Edge Computing untuk melindungi data real-time masih relatif terbatas, khususnya pada lingkungan perusahaan. Oleh karena itu, penelitian ini bertujuan mengimplementasi dan menganalisis penerapan Homomorphic Encryption berbasis skema CKKS pada jaringan SDN untuk meningkatkan keamanan data real-time di lingkungan Edge Computing menggunakan metode Network Development Life Cycle (NDLC) pada PT. Dukuh Raya. Evaluasi dilakukan berdasarkan parameter latensi, penggunaan memori, tingkat keberhasilan transmisi paket (*packet delivery*), serta stabilitas pemrosesan ciphertext guna menilai efektivitas sistem dalam mengurangi risiko penyadapan dan menjaga kerahasiaan informasi perusahaan.

2. Metode Penelitian

Metode yang digunakan dalam penelitian ini adalah Network Development Life Cycle (NDLC) yang diadaptasi sebagai kerangka kerja pengembangan sistem jaringan. NDLC dipilih karena memiliki tahapan yang terstruktur dan sesuai untuk proses perancangan, implementasi, serta evaluasi jaringan komputer. Pada penelitian ini, tahapan NDLC yang digunakan terdiri atas Analysis, Design, Simulation Prototype, Implementation, dan Monitoring. Setiap tahapan dilakukan secara berurutan untuk memastikan sistem jaringan SDN yang terintegrasi dengan Edge Computing dan Homomorphic Encryption dapat dibangun serta dievaluasi secara sistematis. Diagram alir tahapan penelitian ditunjukkan pada Gambar 1.



Gambar 1. Metodologi Penelitian

2.1. Analysis

Tahap *analysis* dilakukan untuk mengidentifikasi kebutuhan sistem jaringan serta permasalahan keamanan data yang terdapat pada lingkungan PT. Dukuh Raya [5]. Proses analisis mencakup pengamatan terhadap mekanisme pertukaran data antar divisi, infrastruktur jaringan yang digunakan, serta potensi ancaman keamanan seperti *sniffing* dan *man-in-the-middle* (MitM) yang dapat memengaruhi kerahasiaan dan integritas data [1], [3]. Selain itu, pada tahap ini dilakukan identifikasi kebutuhan penerapan *Software Defined Networking* (SDN), *Edge Computing*, dan *Homomorphic Encryption* sebagai teknologi utama yang diintegrasikan dalam penelitian [5], [6], [7]. Komponen kontroler terpusat dianalisis untuk mengatasi rigiditas jaringan konvensional [5], sementara kapabilitas komputasi tepi diidentifikasi untuk mendekatkan proses pemrosesan ke sumber data IoT [2]. Hasil analisis menjadi dasar dalam menentukan arsitektur jaringan, kebutuhan perangkat lunak, serta parameter pengujian sumber daya komputasi yang akan digunakan pada tahap selanjutnya [15].

DOI: https://doi.org/10.31004/riggs.v5i2.10086

2.2. Design

Tahap *design* dilakukan dengan merancang arsitektur jaringan berbasis *Software Defined Networking* (SDN) yang terintegrasi dengan *Edge Computing* dan *Homomorphic Encryption*. Perancangan ini mencakup penyusunan topologi jaringan, penentuan fungsi setiap perangkat, serta perancangan alur komunikasi data yang akan digunakan dalam sistem. Topologi yang dirancang menggambarkan proses pertukaran data antar divisi perusahaan dengan memanfaatkan pengelolaan jaringan terpusat melalui SDN dan pemrosesan data pada lapisan edge. Arsitektur jaringan terdiri atas host sebagai pengirim dan penerima data, *OpenFlow switch* sebagai perangkat penerus paket (*forwarding*), *RYU Controller* sebagai komponen pengendali terpusat (*control plane*), serta *edge node* yang berfungsi sebagai lokasi pemrosesan data dan penerapan mekanisme enkripsi [5], [6], [11]. Pada tahap ini juga dirancang mekanisme pengamanan data menggunakan skema *Cheon-Kim-Kim-Song* (CKKS) yang diimplementasikan melalui pustaka TenSEAL [8], [10]. Data yang dikirim akan terlebih dahulu diproses pada *edge node* sebelum diteruskan melalui jaringan SDN dalam bentuk ciphertext. Dengan rancangan tersebut, sistem diharapkan mampu menyediakan pengelolaan jaringan yang fleksibel sekaligus menjaga kerahasiaan data selama proses transmisi pada lingkungan *Edge Computing* [5], [14].

2.3 Simulation Prototyping

Tahap *simulation prototype* dilakukan dengan membangun model simulasi jaringan menggunakan *Mininet* sebagai emulator jaringan SDN [14]. Simulasi digunakan untuk merepresentasikan kondisi jaringan, penjaluran OpenFlow, serta pemetaan antrean paket yang akan diterapkan pada lingkungan penelitian [5], [6]. Pada tahap ini dilakukan konfigurasi fungsional *OpenFlow switch*, integrasi kanal komunikasi menuju *RYU Controller*, serta pengujian konektivitas dasar melalui inspeksi transmisi antar *host* pada topologi star yang telah dirancang [5], [14]. Selain itu, mekanisme *Homomorphic Encryption* berbasis skema CKKS dipersiapkan dengan memetakan parameter kedalaman multiplikatif, konstanta bit pengali, dan pembuatan pasangan kunci kunci publik, kunci rahasia, serta kunci evaluasi Galois untuk diintegrasikan ke dalam saluran pipa komunikasi data [7], [8], [12]. Tahap simulasi bertujuan untuk memastikan bahwa seluruh komponen sistem dapat berinteraksi dan bertukar byte biner sesuai dengan rancangan kearsitekturan yang telah dibuat [11].

2.4 Implementation

Tahap *implementation* merupakan proses penerapan rancangan sistem ke dalam lingkungan pengujian fungsional [3]. Implementasi dilakukan pada sistem operasi *Ubuntu Linux* dengan memanfaatkan *Mininet* sebagai emulator data plane, *RYU Controller* sebagai pengontrol instruksi OpenFlow, dan library *TenSEAL* berbasis dependensi *NumPy* sebagai implementasi teknis komponen *Homomorphic Encryption* [8], [10], [14]. Pada tahap ini dilakukan konfigurasi topologi jaringan secara riil, pembuatan aturan aliran (*flow rules* sekuensial) pada pengontrol, serta implementasi proses enkripsi data numerik pada sisi perangkat tepi [6], [14]. Korpus data mentah yang dilepaskan oleh klien diproses dan diubah menjadi representasi struktur koefisien polinomial acak menggunakan skema CKKS melalui metode serialisasi biner, sehingga informasi yang ditransmisikan melintasi jalur switch berada dalam bentuk *ciphertext* utuh selama proses komunikasi berlangsung [6], [7], [10].

2.5 Monitoring

Tahap *monitoring* dilakukan untuk mengamati dan merekam aktivitas jaringan selama proses pengujian [3]. *Monitoring* bertujuan untuk memperoleh data yang diperlukan dalam proses evaluasi sistem [14]. Pengamatan dilakukan menggunakan beberapa perangkat lunak pendukung, seperti *Wireshark* untuk analisis paket jaringan dan utilitas sistem Linux untuk pemantauan sumber daya komputasi [3], [14]. Parameter yang diamati dalam penelitian ini meliputi latensi, *throughput*, penggunaan CPU, penggunaan memori, keberhasilan pengiriman paket, serta keamanan data terhadap potensi serangan penyadapan (*sniffing*) [1], [3], [14]. Data yang diperoleh dari tahap *monitoring* selanjutnya digunakan sebagai dasar analisis pada tahap hasil dan pembahasan.

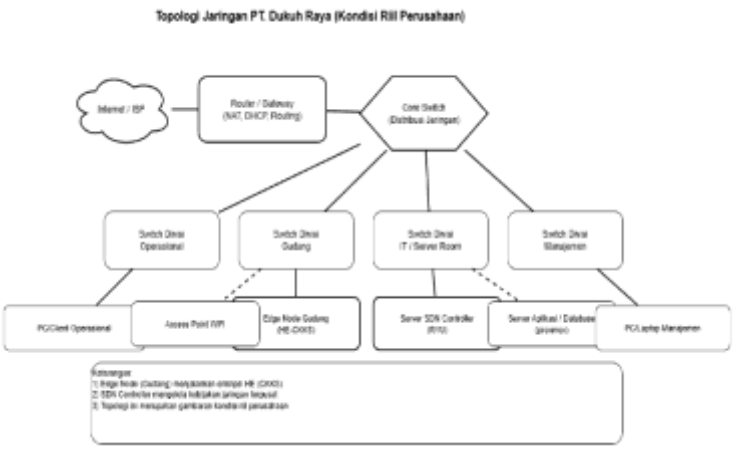
3. Hasil dan Diskusi

3.1 Hasil

Implementasi sistem pada penelitian ini dilakukan berdasarkan tahapan *Network Development Life Cycle* (NDLC) yang telah dijelaskan pada bagian metodologi penelitian. Sistem yang dibangun mengintegrasikan *Software Defined Networking* (SDN), *Edge Computing*, dan *Homomorphic Encryption* untuk meningkatkan keamanan data real-time pada lingkungan jaringan PT. Dukuh Raya. Implementasi dilakukan menggunakan Mininet pada sistem operasi Ubuntu Linux sebagai emulator jaringan SDN sehingga mampu merepresentasikan kondisi jaringan yang mendekati implementasi nyata [5], [14].

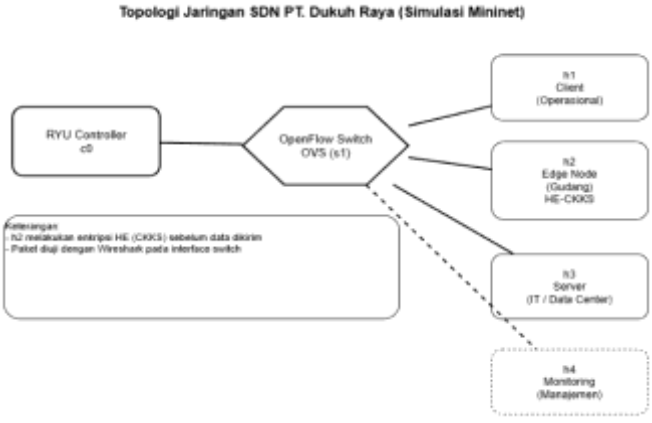
DOI: https://doi.org/10.31004/riggs.v5i2.10086

Hasil tahap *design* menghasilkan dua rancangan topologi jaringan, yaitu topologi jaringan eksisting PT. Dukuh Raya dan topologi jaringan SDN yang diusulkan. Topologi jaringan eksisting menggambarkan kondisi jaringan perusahaan yang terdiri atas koneksi internet, router, *core switch*, serta beberapa *switch* distribusi yang melayani divisi operasional, gudang, IT/*server room*, dan manajemen. Topologi ini digunakan sebagai dasar dalam proses analisis kebutuhan dan perancangan sistem yang dikembangkan pada penelitian ini. Topologi jaringan eksisting ditunjukkan pada Gambar 2.



Gambar 2. Topologi Jaringan PT. Dukuh Raya (Kondisi Riil Perusahaan)

Berdasarkan analisis terhadap jaringan eksisting, dirancang topologi jaringan berbasis SDN sebagai solusi yang diusulkan dalam penelitian ini. Topologi tersebut terdiri atas RYU Controller sebagai *control plane*, OpenFlow *switch* sebagai *data plane*, *host client* sebagai sumber data, *edge node* sebagai lokasi penerapan *Homomorphic Encryption*, *server* sebagai tujuan pengiriman data, serta *host monitoring* untuk melakukan pengamatan lalu lintas jaringan [5], [6], [14]. Arsitektur ini digunakan sebagai lingkungan simulasi untuk mengimplementasikan konsep SDN dan *Edge Computing* pada penelitian. Topologi jaringan SDN yang diusulkan ditunjukkan pada Gambar 3.



Gambar 3. Topologi Jaringan SDN PT. Dukuh Raya (Simulasi Mininet)

Penelitian ini menggunakan RYU Controller untuk mengelola *flow rules* dan mengatur jalur komunikasi antar perangkat secara terpusat [5], [6]. Implementasi *Homomorphic Encryption* dilakukan menggunakan bahasa pemrograman Python dengan memanfaatkan pustaka TenSEAL dan NumPy. Skema enkripsi yang digunakan adalah *Cheon-Kim-Kim-Song* (CKKS) yang memungkinkan data numerik diproses dalam bentuk ciphertext [8], [10]. Pada mekanisme yang diterapkan, data yang dikirim dari host *client* terlebih dahulu diproses dan dienkripsi pada *edge node* sebelum diteruskan melalui jaringan SDN menuju *server* [7], [11].

Untuk mengevaluasi efektivitas sistem yang dibangun, pengujian dilakukan menggunakan dua skenario utama, yaitu pengiriman data tanpa enkripsi dan pengiriman data menggunakan *Homomorphic Encryption*. Pada skenario tanpa enkripsi, data dikirim dalam bentuk *plaintext* sehingga isi paket dapat diamati secara langsung selama proses transmisi. Sebaliknya, pada skenario dengan *Homomorphic Encryption*, data dikirim dalam bentuk ciphertext

DOI: https://doi.org/10.31004/riggs.v5i2.10086

sehingga informasi yang ditransmisikan tidak dapat dibaca secara langsung oleh pihak yang tidak berwenang [7], [11]. Selain pengujian keamanan, penelitian ini juga melakukan pengujian performa jaringan untuk mengetahui pengaruh penerapan *Homomorphic Encryption* terhadap sistem SDN berbasis *Edge Computing*. Parameter yang digunakan dalam pengujian meliputi latensi (*latency*), *throughput*, penggunaan CPU, dan penggunaan memori [1], [3]. Pengujian latensi dilakukan untuk mengukur waktu yang diperlukan paket data untuk mencapai tujuan pada kedua skenario pengujian [14]. Hasil pengujian menunjukkan bahwa rata-rata latensi pada skenario tanpa enkripsi sebesar 1,2 ms, sedangkan pada skenario dengan penerapan *Homomorphic Encryption* sebesar 9,8 ms. Ringkasan hasil pengujian latensi ditunjukkan pada Tabel 1.

Tabel 1. Hasil Pengujian Latensi

Skenario	Rata-rata Latensi (ms)
Tanpa Enkripsi	1,2 ms
Dengan HE	9,8 ms

Selanjutnya, dilakukan pengujian *throughput* untuk mengukur kemampuan jaringan dalam mentransmisikan data selama proses komunikasi berlangsung [1], [3], [14]. Hasil pengujian menunjukkan bahwa pada skenario tanpa enkripsi diperoleh nilai *throughput* pada rentang 95–100 Mbps, sedangkan pada skenario yang menerapkan *Homomorphic Encryption* (CKKS) diperoleh nilai *throughput* pada rentang 60–70 Mbps. Nilai tersebut merepresentasikan kemampuan transfer data yang dicapai pada masing-masing skenario pengujian. Ringkasan hasil pengujian *throughput* ditunjukkan pada Tabel 2.

Tabel 2. Hasil Pengujian Throughput

Skenario	Throughput (Mbps)
Tanpa Enkripsi	95 – 100 Mbps
Dengan HE	60 – 70 Mbps

Pengujian penggunaan memori dilakukan untuk mengetahui sumber daya memori yang digunakan sistem selama proses pengiriman data [3], [14]. Pada skenario tanpa enkripsi, total memori sistem tercatat sebesar 3931 MB dengan penggunaan memori (*used memory*) sebesar 1227 MB, memori bebas (*free memory*) sebesar 1551 MB, *buffer/cache* sebesar 1152 MB, dan memori tersedia (*available memory*) sebesar 2412 MB. Sementara itu, pada skenario dengan *Homomorphic Encryption* (CKKS), total memori tetap sebesar 3931 MB, dengan penggunaan memori sebesar 1086 MB, memori bebas sebesar 2075 MB, *buffer/cache* sebesar 769 MB, dan memori tersedia sebesar 2587 MB. Ringkasan hasil pengujian penggunaan memori disajikan pada Tabel 3.

Tabel 3. Hasil Pengujian Memori

Skenario	Total (MB)	Used (MB)	Free (MB)	Buff/Cache (MB)	Available (MB)
Tanpa Enkripsi	3931	1227	1551	1152	2412
Dengan HE	3931	1086	2075	769	2587

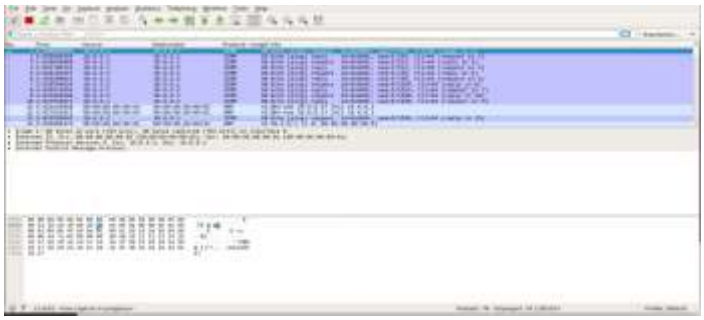
pengujian penggunaan memori, dilakukan pula pengujian penggunaan CPU untuk mengetahui tingkat pemanfaatan sumber daya prosesor selama sistem beroperasi [14]. Hasil pengujian menunjukkan bahwa penggunaan CPU pada skenario tanpa enkripsi sebesar 1,3%, sedangkan pada skenario yang menerapkan *Homomorphic Encryption* (CKKS) mencapai 13%. Perbedaan nilai tersebut menunjukkan tingkat penggunaan sumber daya prosesor yang tercatat pada masing-masing skenario pengujian. Ringkasan hasil pengujian penggunaan CPU disajikan pada ditunjukkan pada Tabel 4.

Tabel 4. Hasil Pengujian CPU

Skenario	Penggunaan CPU (%)
Tanpa Enkripsi	1,3%
Dengan HE	13%

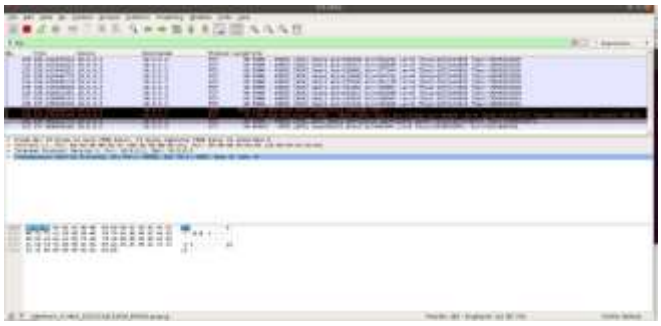
DOI: https://doi.org/10.31004/riggs.v5i2.10086

Selain pengujian performa, penelitian ini juga melakukan pengujian keamanan untuk mengamati bentuk data yang ditransmisikan pada jaringan. Pengujian dilakukan menggunakan aplikasi Wireshark dengan membandingkan paket data yang dikirim tanpa enkripsi dan dengan penerapan *Homomorphic Encryption* (CKKS). Berdasarkan hasil paket menggunakan Wireshark, data yang dikirim masih dapat dibaca secara langsung dalam bentuk *plaintext* sehingga informasi yang ditransmisikan dapat diamati oleh pihak yang melakukan pemantauan lalu lintas jaringan. Hasil pengujian keamanan pada skenario tanpa enkripsi ditunjukkan pada Gambar 4.



Gambar 4. Hasil Pengamatan Paket Data Tanpa Enkripsi Menggunakan Wireshark

Sementara itu, pada skenario yang menerapkan *Homomorphic Encryption* (CKKS), data yang ditransmisikan tampak dalam bentuk *ciphertext* sehingga informasi yang dikirim tidak dapat dibaca secara langsung melalui proses *packet capture*. Hasil tersebut menunjukkan bahwa data telah melalui proses enkripsi sebelum dikirimkan melalui jaringan. Bentuk data yang ditampilkan berbeda dengan skenario tanpa enkripsi, di mana informasi masih dapat diamati dalam bentuk *plaintext*. Visualisasi hasil pengujian keamanan pada skenario dengan penerapan *Homomorphic Encryption* (CKKS) disajikan pada Gambar 5.



Gambar 5. Hasil Capture Paket Data dengan Homomorphic Encryption (CKKS) Menggunakan Wireshark

3.2 Diskusi

Berdasarkan hasil pengujian yang telah dilakukan, penerapan *Homomorphic Encryption* (CKKS) pada lingkungan SDN berbasis *Edge Computing* memberikan pengaruh terhadap aspek keamanan dan performa sistem [6], [14], [15]. Pengujian keamanan menggunakan *Wireshark* menunjukkan bahwa pada skenario tanpa enkripsi, data yang ditransmisikan masih dapat diamati dalam bentuk *plaintext*. Sebaliknya, pada skenario yang menerapkan *Homomorphic Encryption*, data ditampilkan dalam bentuk *ciphertext* sehingga informasi yang dikirim tidak dapat dibaca secara langsung selama proses transmisi [6], [7], [11]. Hasil ini menunjukkan bahwa mekanisme enkripsi yang diterapkan mampu meningkatkan kerahasiaan data *real-time* pada jaringan yang dibangun.

Dari sisi performa jaringan, penerapan *Homomorphic Encryption* menyebabkan peningkatan nilai latensi dibandingkan skenario tanpa enkripsi. Kondisi ini terjadi karena data harus melalui proses pembangkitan kunci, enkripsi pada *edge node*, serta proses pemrosesan ciphertext sebelum diteruskan ke server tujuan [7], [11], [12]. Tahapan tambahan tersebut membutuhkan waktu komputasi sehingga menambah waktu yang diperlukan paket untuk mencapai tujuan. Meskipun demikian, nilai latensi yang diperoleh masih berada pada tingkat milidetik sehingga komunikasi data tetap dapat berlangsung secara real-time. Hasil pengujian *throughput* menunjukkan adanya penurunan kemampuan transfer data setelah penerapan *Homomorphic Encryption*. Penurunan ini dapat disebabkan oleh bertambahnya ukuran data hasil enkripsi dan adanya proses komputasi tambahan yang harus dilakukan sebelum data dikirimkan melalui jaringan. Pada skema CKKS, ciphertext memiliki ukuran yang lebih

DOI: https://doi.org/10.31004/riggs.v5i2.10086

besar dibandingkan data asli (*plaintext*), sehingga jumlah data yang dapat ditransmisikan dalam satuan waktu tertentu menjadi lebih rendah dibandingkan skenario tanpa enkripsi [8], [9].

Pada pengujian penggunaan CPU, terlihat adanya peningkatan pemanfaatan sumber daya prosesor pada skenario yang menerapkan *Homomorphic Encryption*. Kondisi tersebut menunjukkan bahwa proses enkripsi membutuhkan operasi matematika yang lebih kompleks dibandingkan pengiriman data biasa [4], [12]. Algoritma CKKS melakukan berbagai operasi kriptografi pada data numerik sehingga beban komputasi sistem menjadi lebih tinggi [8], [12]. Peningkatan penggunaan CPU ini merupakan konsekuensi yang umum terjadi pada implementasi teknik kriptografi modern yang berorientasi pada peningkatan keamanan data [3], [4]. Sementara itu, hasil pengujian penggunaan memori menunjukkan perubahan yang relatif kecil antara kedua skenario pengujian. Hal ini mengindikasikan bahwa penerapan *Homomorphic Encryption* pada penelitian ini tidak memberikan dampak yang signifikan terhadap konsumsi memori sistem [14]. Dengan demikian, kebutuhan sumber daya yang paling terpengaruh oleh proses enkripsi berada pada aspek komputasi prosesor dibandingkan penggunaan memori.

4. Kesimpulan

Berdasarkan hasil implementasi dan pengujian yang telah dilakukan, integrasi *Software Defined Networking* (SDN), *Edge Computing*, dan *Homomorphic Encryption* (CKKS) berhasil diterapkan pada lingkungan simulasi jaringan PT. Dukuh Raya menggunakan Mininet dan RYU Controller. Penerapan *Homomorphic Encryption* pada *edge node* mampu meningkatkan keamanan data selama proses transmisi, yang ditunjukkan melalui hasil pengujian menggunakan Wireshark. Pada skenario tanpa enkripsi, data masih dapat diamati dalam bentuk *plaintext*, sedangkan pada skenario dengan *Homomorphic Encryption*, data ditransmisikan dalam bentuk *ciphertext* sehingga tidak dapat dibaca secara langsung. Hasil pengujian performa menunjukkan bahwa penerapan *Homomorphic Encryption* memberikan dampak terhadap kinerja sistem. Nilai latensi meningkat dari 1,2 ms menjadi 9,8 ms, sementara *throughput* menurun dari rentang 95–100 Mbps menjadi 60–70 Mbps. Selain itu, penggunaan CPU meningkat dari 1,3% menjadi 13%, sedangkan penggunaan memori tidak mengalami perubahan yang signifikan. Meskipun demikian, sistem tetap mampu menjalankan komunikasi data dengan baik pada lingkungan jaringan yang dibangun. Dengan demikian, penelitian ini menunjukkan bahwa penerapan *Homomorphic Encryption* pada arsitektur SDN berbasis *Edge Computing* dapat menjadi solusi yang efektif untuk meningkatkan keamanan data real-time. Hasil penelitian juga menunjukkan adanya kompromi antara peningkatan keamanan dan performa sistem, namun dampak yang ditimbulkan masih berada pada batas yang dapat diterima sehingga pendekatan ini berpotensi untuk diterapkan pada lingkungan jaringan perusahaan yang membutuhkan tingkat keamanan data yang lebih tinggi.

Referensi

[1] J. K. Lastre and Y. Ko, “Evaluating Transport Layer Security 1 . 3 Optimization Strategies for 5G Cross-Border Roaming : A Comprehensive Security and Performance Analysis,” *Sensors*, vol. 25, no. 19, pp. 1–33, 2025, doi: 10.3390/s25196144.
[2] D. S. Excel, “Edge Computing : Opportunities and Challenges,” *World J. Adv. Res. Rev.*, vol. 23, no. September, pp. 585–596, 2024, doi: 10.30574/wjarr.2024.23.3.2723.
[3] M. El-hajj, H. Mousawi, and A. Fadlallah, “Analysis of Lightweight Cryptographic Algorithms on IoT Hardware Platform †I,” *Futur. Internet*, vol. 15, no. 2, pp. 1–29, 2023, doi: 10.3390/fi15020054.
[4] S. R. K, “A Novel Lightweight Cryptographic Approach for IoT Using Modi ed SLIM Cipher on FPGA A Novel Lightweight Cryptographic Approach for IoT Using Modified SLIM Cipher on FPGA,” *Res. Sq.*, 2026, doi: 10.21203/rs.3.rs-9119520/v1.
[5] R. Wainwright, M. Bagheri, A. Salama, and R. Saatchi, “Software-Defined Networking Security Detection Strategies and Their Limitations with a Focus on Distributed Denial- of-Service for Small to Medium-Sized Enterprises Software-Defined Networking Security Detection Strategies and Their Limitations with a Foc,” *Appl. Sci.*, vol. 15, pp. 1–21, 2025, doi: 10.3390/app152312389.
[6] S. I. J. King and C. Inf, “CALIS : AI-driven context-aware encryption for SDN-enabled smart-home IoT IN IN,” *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 38, 2026, doi: 10.1007/s4444 3-025-00404-9.
[7] Y. H. Chan *et al.*, “HHEML : Hybrid Homomorphic Encryption for Privacy-Preserving Machine Learning on Edge,” *arVix*, 2025.
[8] J. L. P. N. D. H. Lee, “Configurable Encryption and Decryption Architectures for CKKS-Based Homomorphic Encryption,” *Sensors*, vol. 23, pp. 1–13, 2023, doi: 10.3390/s23177389.
[9] W. Jin *et al.*, “FEDML-He: An Efficient Homomorphic-encryption-based Privacy-preserving Federated Learning System,” *arVix*, 2024.
[10] H. Jia *et al.*, “Efficient and privacy - preserving image classification using homomorphic encryption and chunk - based convolutional neural network,” *J. Cloud Comput.*, 2023, doi: 10.1186/s13677-023-00537-0.
[11] X. Yu, D. Tang, and W. Zhao, “Privacy - preserving cloud - edge collaborative learning without trusted third - party coordinator,” *J. Cloud Comput.*, pp. 1–11, 2023, doi: 10.1186/s13677-023-00394-x.
[12] C. S. R. Huang, “Secure Convolution Neural Network Inference Based on Homomorphic Encryption,” *Appl. Sci.*, vol. 13, no. 1, pp. 1–15, 2023, doi: 10.3390/app13106117.
[13] A. Ali, B. A. S. Al-rimy, F. S. Alsubaei, A. A. Almazroi, and A. A. Almazroi, “HealthLock : Blockchain-Based Privacy Preservation Using Homomorphic Encryption in Internet of Things Healthcare Applications,” *Sensors*, vol. 23, no. 15, pp. 1–29, 2023, doi: 10.3390/s23156762.
[14] J. Park *et al.*, “Toward Practical Privacy-Preserving Convolutional Neural Networks,” *arXiv*, pp. 10–12, 2023.
[15] S. Reports, “Machine learning-driven adaptive parameter selection for homomorphic encryption in edge computing IN IN,” *Sci. Rep.*,

DOI: https://doi.org/10.31004/riggs.v5i2.10086

vol. 16, 2026, doi: 10.1038/ s41598-026-52587-4.

DOI: https://doi.org/10.31004/riggs.v5i2.10086