



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



**ANALISIS MITIGASI SERANGAN *MAN-IN-THE-MIDDLE* PADA KOMUNIKASI LORA PEER-TO-PEER
BERBASIS *DYNAMIC KEY MANAGEMENT* DI
JARINGAN NON-INTERNET**

SKRIPSI
**POLITEKNIK
NEGERI
JAKARTA**

MOHAMMAD RIZKI FADILLAH (2207421016)

**PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK NEGERI JAKARTA**

2026



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



**ANALISIS MITIGASI SERANGAN *MAN-IN-THE-MIDDLE* PADA KOMUNIKASI LORA PEER-TO-PEER
BERBASIS *DYNAMIC KEY MANAGEMENT* DI
JARINGAN NON-INTERNET**

SKRIPSI

**POLITEKNIK
NEGERI
JAKARTA**

**Dibuat untuk Melengkapi Syarat-Syarat yang Diperlukan untuk
Memperoleh Diploma Empat Politeknik**

MOHAMMAD RIZKI FADILLAH

2207421016

**PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER
POLITEKNIK NEGERI JAKARTA**

2026



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertanda tangan di bawah ini:

Nama : Mohammad Rizki Fadillah
NIM : 2207421016
Jurusan / Program Studi : Teknik Informatika dan Komputer / Teknik Multimedia dan Jaringan
Judul Skripsi : Analisis Mitigasi Serangan *Man In The Middle* Pada Komunikasi LoRa Peer-to-Peer Berbasis *Dynamic Key Management* Di Jaringan Non-Internet

Menyatakan dengan sebenarnya bahwa skripsi ini benar-benar merupakan hasil karya saya sendiri, bebas dari peniruan terhadap karya dari orang lain. Kutipan pendapat orang lain dan tulisan orang lain ditunjuk sesuai dengan cara-cara penulisan karya ilmiah yang berlaku.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa dalam skripsi ini terkandung ciri-ciri plagiat dan bentuk-bentuk peniruan lain yang dianggap melanggar peraturan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

**POLITEKNIK
NEGERI
JAKARTA**

Depok, 5 Juni 2026

Yang membuat pernyataan



(Mohammad Rizki Fadillah)

NIM. 2207421016



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

LEMBAR PENGESAHAN

Skripsi diajukan oleh :

Nama : Mohammad Rizki Fadillah
NIM : 2207421016
Program Studi : Teknik Multimedia dan Jaringan
Judul Skripsi : Analisis Mitigasi Serangan *Man In The Middle* Pada Komunikasi LoRa *Peer-to-Peer* Berbasis *Dynamic Key Management* Di Jaringan Non-Internet

Telah diuji oleh tim penguji dalam Sidang Skripsi pada hari Jumat, Tanggal 19, Bulan Juni, Tahun 2026 dan dinyatakan **LULUS**

Disahkan Oleh

Pembimbing I : Defiana Arnaldy, S.T.P., M.Si. ()
Penguji I : Ayu Rosyida Zain, S.ST. M.T. ()
Penguji II : Fachroni Arbi Murrad, S.Kom., M.Kom. ()
Penguji III : Iik Muhamad Malik Matin, S.Kom., M.T. ()

**POLITEKNIK
NEGERI
JAKARTA**

Mengetahui :

Jurusan Teknik Informatika dan Komputer

Ketua



Hidayati, S.Kom., M. Kom.

NIP. 197908032003122003



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

KATA PENGANTAR

Puji syukur ke hadirat Allah SWT atas berkat dan rahmat-Nya sehingga penulis dapat menyelesaikan skripsi berjudul “Analisis Mitigasi Serangan *Man-In-The-Middle* Pada Komunikasi LoRa *Peer-to-Peer* Berbasis *Dynamic Key Management* Di Jaringan Non-Internet” sebagai salah satu syarat memperoleh gelar Sarjana Terapan Komputer di Politeknik Negeri Jakarta. Penulis menyadari bahwa kelancaran penyusunan skripsi ini tidak terlepas dari bantuan serta bimbingan berbagai pihak, sehingga pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Allah SWT atas kemudahan, kekuatan, petunjuk, dan karunia-Nya sehingga penulis bisa menyelesaikan skripsi ini tepat waktu.
2. Bapak Defiana Arnaldy, S.T.P., M.Si., selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan penulis dalam penyusunan skripsi ini;
3. Seluruh staf, pengajar, dan karyawan Jurusan Teknik Informatika dan Komputer Politeknik Negeri Jakarta yang telah memberikan ilmu dan bantuan administrasi selama masa perkuliahan;
4. Orang tua dan keluarga penulis yang telah memberikan bantuan dukungan material dan moral;
5. Rekan-rekan mahasiswa TMJ 8A 2022 yang telah banyak membantu penulis dalam memberikan ide serta dukungan dalam menyelesaikan ini.

Akhir kata, penulis berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu pengetahuan.

Depok, 6 Juni 2026

Mohammad Rizki Fadillah



© Hak Cipta milik Politeknik Negeri Jakarta

- Hak Cipta :**
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
 2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Politeknik Negeri Jakarta, saya bertanda tangan di bawah ini:

Nama : Mohammad Rizki Fadillah
NIM : 2207421016
Jurusan / Program Studi : Teknik Informatika dan Komputer /
Teknik Multimedia dan Jaringan

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Politeknik Negeri Jakarta Hak Bebas Royalti Eksklusif atas karya ilmiah saya yang berjudul:

Analisis Mitigasi Serangan *Man In The Middle* Pada Komunikasi LoRa Peer-to-Peer Berbasis *Dynamic Key Management* Di Jaringan Non-Internet

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksklusif ini Politeknik Negeri Jakarta berhak menyimpan, mengalihmediakan/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Depok, 5 Juni 2026

Yang menyatakan,

(Mohammad Rizki Fadillah)

NIM. 2207421016



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Analisis Mitigasi Serangan Man-In-The-Middle Pada Komunikasi LoRa Peer-to-Peer Berbasis Dynamic Key Management Di Jaringan Non-Internet

ABSTRAK

Pemanfaatan Internet of Things (IoT) di area blank spot sering mengandalkan komunikasi Point-to-Point (P2P) karena efisiensi daya dan jangkauannya. Namun, berbeda dengan LoRaWAN, LoRa P2P tidak memiliki standar keamanan bawaan sehingga rentan terhadap penyadapan, manipulasi pesan (Man-In-The-Middle/MITM), dan Replay Attack. Risiko ini memburuk dengan penggunaan kunci statis, di mana pencurian perangkat fisik dapat mengkompromikan seluruh jaringan. Penelitian ini bertujuan merancang sistem keamanan LoRa P2P menggunakan mikrokontroler ESP32 dengan menerapkan kriptografi hibrida dan manajemen kunci dinamis. Metode yang digunakan mencakup algoritma AES-128 untuk kerahasiaan data dan HMAC-SHA256 untuk verifikasi integritas pesan. Untuk mengatasi kelemahan kunci statis dan Replay Attack, dikembangkan mekanisme Dynamic Key Management yang memperbarui kunci enkripsi secara otomatis berdasarkan counter internal, serta Counter-based Nonce untuk memastikan kesegaran pesan. Hasil pengujian menunjukkan bahwa sistem yang diusulkan mampu memitigasi serangan MITM dengan mendeteksi perubahan tanda tangan HMAC dan menolak paket replay yang tidak valid. Kesimpulannya, sistem ini berhasil menjaga kerahasiaan dan integritas data secara dinamis, dengan tetap mempertahankan efisiensi komputasi pada perangkat berdaya rendah.

Kata Kunci: AES-128, Dynamic Key Management, ESP32, HMAC-SHA256, LoRa P2P, Man-In-The-Middle.

**POLITEKNIK
NEGERI
JAKARTA**



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR ISI

SURAT PERNYATAAN BEBAS PLAGIARISME	ii
LEMBAR PENGESAHAN	iii
KATA PENGANTAR.....	iv
SURAT PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS.....	v
ABSTRAK	vi
DAFTAR ISI	vii
DAFTAR TABEL.....	x
DAFTAR GAMBAR	xi
DAFTAR LAMPIRAN	xiii
BAB I PENDAHULUAN	1
1.1 Latar Belakang Masalah.....	1
1.2 Perumusan Masalah	2
1.3 Batasan Masalah.....	3
1.4 Tujuan dan Manfaat	3
1.5 Sistematika Penulisan	4
BAB II TINJAUAN PUSTAKA	6
2.1 <i>Internet of Things (IoT)</i>	6
2.1.1 End-Node	8
2.1.2 Gateway.....	8
2.1.3 Attacker Node	9
2.2 Teknologi LoRa (<i>Long Range</i>).....	9
2.2.1 Modul LoRa Ra-02 (SX1278).....	11
2.2.2 Perbandingan Arsitektur LoRaWAN dan LoRa P2P.....	11
2.3 Mikrokontroler	12
2.3.1 ESP32.....	13
2.4 DHT22	14



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.5	AMS1117	14
2.6	<i>Lightweight cryptography</i>	15
2.6.1	Advanced Encryption Standard (AES-128)	15
2.6.2	Hash-based Message Authentication Code (HMAC-SHA256)	16
2.6.3	Manajemen Kunci Dinamis (<i>Dynamic Key Management</i>)	17
2.7	Key Derivation Function (KDF)	18
2.8	Perangkat Lunak Pendukung.....	19
2.8.1	Visual Studio Code (VS Code)	19
2.8.2	Laragon	21
2.9	Penelitian Terdahulu.....	22
2.10	Kerangka Pemikiran.....	25
BAB III METODE PENELITIAN.....		27
3.1	Rancangan Penelitian	27
3.2	Tahapan Penelitian	28
3.3	Objek Penelitian	29
BAB IV HASIL DAN PEMBAHASAN.....		30
4.1	Analisis Kebutuhan	30
4.2	Perancangan Sistem	32
4.2.1	Rancangan Topologi Jaringan	33
4.2.2	Diagram Blok	34
4.2.3	Diagram Alur (<i>Flowchart</i>)	34
4.2.4	Desain Skematik dan Konfigurasi Pin	40
4.3	Implementasi Sistem	42
4.3.1	Implementasi Perangkat Keras.....	42
4.3.2	Implementasi Perangkat Lunak.....	44
4.4	Pengujian Sistem.....	50



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.4.1	Deskripsi Pengujian	50
4.4.2	Prosedur Pengujian	51
4.5	Data Hasil Pengujian.....	54
4.5.1	Data Hasil Pengujian Fase <i>Scanning</i>	54
4.5.2	Data Hasil Pengujian Sistem <i>Baseline</i>	55
4.5.3	Data Hasil Pengujian Keamanan Hanya AES-128.....	56
4.5.4	Data Hasil Pengujian Keamanan AES-128 dan HMAC (Kunci Statis)	57
4.5.5	Data Hasil Pengujian Sistem Mitigasi Penuh.....	59
4.5.6	Data Hasil <i>Quality of Service</i> (QoS)	61
4.6	Analisis Data dan Evaluasi Hasil Pengujian	63
4.6.1	Analisis Stabilitas Sistem Mitigasi Penuh.....	63
4.6.2	Analisis Kinerja Komputasi dan Kualitas Jaringan (QoS).....	64
4.6.3	Analisis Evaluasi Keamanan Sistem (Mitigasi Serangan)	65
BAB V	PENUTUP.....	68
5.1	Kesimpulan	68
5.2	Saran.....	69
DAFTAR PUSTAKA		70
DAFTAR RIWAYAT HIDUP PENULIS		74
LAMPIRAN		75



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR TABEL

Tabel 2.1 Penelitian Terdahulu	23
Tabel 4.1 Kebutuhan Perangkat Keras (<i>Hardware</i>)	30
Tabel 4.2 Kebutuhan Perangkat Lunak (<i>Software</i>)	31
Tabel 4.3 Konfigurasi <i>Pinout</i> Sistem	42
Tabel 4.4 Perbandingan Tingkat Keberhasilan Mitigasi Serangan	61
Tabel 4.5. Hasil Perbandingan <i>Quality of Service</i> (QoS).....	62
Tabel 4.6. Hasil Pengujian Stabilitas Kinerja Sistem Mitigasi Penuh Per Interval	63





Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR GAMBAR

Gambar 2.1 LoRa <i>Technologies</i>	9
Gambar 2.2 LoRa Ra-02	11
Gambar 2.3 ESP32	13
Gambar 2.4 Sensor Suhu dan Kelembapan DHT22.....	14
Gambar 2.5 Regulator Tegangan AMS1117	15
Gambar 2.6 Visual Studio Code.....	19
Gambar 2.7 PlatformIO.....	20
Gambar 2.8 C++.....	20
Gambar 2.9 Laragon.....	21
Gambar 2.10 MySQL.....	21
Gambar 4.1 Topologi Jaringan Komunikasi LoRa P2P dengan Intervensi <i>Attacker</i>	33
Gambar 4.2 Diagram Blok Sistem Keamanan LoRa P2P	34
Gambar 4.3 Flowchart Pengirim	35
Gambar 4.4 Flowchart Penerima.....	36
Gambar 4.5 Flowchart Penyerang.....	37
Gambar 4.6 Flowchart <i>Attacker</i> Melakukan Serangan	38
Gambar 4.7 Flowchart <i>Key Derivation Function</i>	39
Gambar 4.8 Rancangan Skematik Pengirim	40
Gambar 4.9 Rancangan Skematik <i>Gateway</i> (Penerima)	41
Gambar 4.10 Rancangan Skematik <i>Attacker</i> (Penyerang).....	41
Gambar 4.11 Rangkaian ESP32 dan Modul LoRa Ra-02.....	43
Gambar 4.12 Rangkaian ESP32 dan Sensor DHT22	43
Gambar 4.13 Rangkaian Regulator AMS1117.....	44
Gambar 4.14 Perakitan Keseluruhan Perangkat Keras	44
Gambar 4.15 Implementasi Fungsi Inisialisasi Modul LoRa.....	45
Gambar 4.16 Implementasi Algoritma Enkripsi AES-128 CTR.....	46
Gambar 4.17 Implementasi Deklarasi Variabel Kunci Keamanan.....	46
Gambar 4.18 Implementasi Inisialisasi Pustaka dan Kunci Dasar HMAC-SHA256	47
Gambar 4.19 Implementasi Kalkulasi Integritas Pesan dan Eksekusi Hash Akhir.....	47



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Gambar 4.20 Implementasi Deklarasi Variabel Kunci Keamanan.....	48
Gambar 4.21 Implementasi Kalkulasi KDF dan Pemotongan Kunci Sesi.....	49
Gambar 4.22 Implementasi Penerapan Kunci Sesi Dinamis pada Enkripsi AES-128	50
Gambar 4.23 Log Hasil <i>Scanning</i> Jaringan LoRa.....	55
Gambar 4.24 Log Hasil <i>Sniffing Baseline</i>	55
Gambar 4.25 Log Hasil Serangan <i>Tampering</i> Pada <i>Gateway</i>	56
Gambar 4.26 Log Hasil <i>Replay Attack</i> Pada <i>Gateway</i>	56
Gambar 4.27 Log Hasil <i>Sniffing</i> pada Skenario AES128.....	57
Gambar 4.28 Log Hasil <i>Tampering</i> pada Skenario AES128.....	57
Gambar 4.29 Log Hasil <i>Replay Attack</i> pada Skenario AES128.....	57
Gambar 4.30 Log Hasil Serangan <i>Sniffing Static Key</i>	58
Gambar 4.31 Log Hasil <i>Tampering</i> Pada <i>Gateway</i> dengan Kunci Statis.....	59
Gambar 4.32 Log Hasil <i>Replay Attack</i> Pada <i>Gateway</i> dengan Kunci Statis.....	59
Gambar 4.33 Log Hasil Serangan <i>Sniffing</i> Pada Kunci Dinamis.....	60
Gambar 4.34 Log Penolakan Serangan <i>Tampering</i> pada <i>Gateway</i>	60
Gambar 4.35 Log Penolakan <i>Replay Attack</i> pada <i>Gateway</i>	61
Gambar 4.36 Grafik Stabilitas Sistem (Interval Paket).....	64
Gambar 4.37 Grafik Perbandingan Kinerja Komputasi dan QoS.....	64
Gambar 4.38 Grafik Tingkat Keberhasilan Mitigasi Serangan (<i>Drop Rate</i>).....	66



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR LAMPIRAN

Lampiran 1. Dokumentasi Perangkat Keras.....	75
Lampiran 2. Dokumentasi Implementasi Sistem	75
Lampiran 3. Dokumentasi Pengujian Sistem	76
Lampiran 4. Spesifikasi Teknis Mikrokontroler ESP32.....	76
Lampiran 5. Spesifikasi Teknis LoRa Ra-02	77
Lampiran 6. Spesifikasi Teknis DHT22	78
Lampiran 7. Tautan Repositori Kode Program (<i>Source Code</i>)	78
Lampiran 8. Tautan Akses Dataset Pengujian.....	78





Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Perkembangan *Internet of Things* (IoT) telah mendorong adopsi teknologi *Low Power Wide Area Network* (LPWAN), khususnya LoRa, untuk berbagai aplikasi monitoring jarak jauh. Meskipun protokol LoRaWAN umum digunakan, keandalan jaringannya sangat bergantung pada infrastruktur gateway dan network server yang harus selalu terhubung ke internet, (Rosyida Zain et al., 2022). Selain itu, tingginya harga infrastruktur perangkat LoRaWAN Gateway komersial seringkali menjadi kendala besar bagi industri menengah dan kecil. Sebagai alternatif pengganti LoRaWAN, pengembangan gateway mandiri (*custom gateway*) berbiaya rendah menggunakan mode komunikasi *Point-to-Point* (P2P) menjadi solusi yang efisien dan mandiri, terutama di wilayah yang tidak terjangkau jaringan seluler (blank spot) (C. Sun et al., 2020).

Namun, kemandirian infrastruktur *custom gateway* ini membawa tantangan keamanan yang signifikan karena modulasi LoRa pada lapisan fisik tidak memiliki mekanisme keamanan bawaan. Berbeda dengan protokol LoRaWAN yang memiliki lapisan keamanan standar yang diatur di sisi server, komunikasi LoRa P2P mengirimkan data secara terbuka (*plaintext*), sehingga sangat rentan terhadap penyadapan. Hal ini diperkuat oleh penelitian yang membuktikan bahwa payload LoRa P2P dapat dengan mudah disadap di udara menggunakan perangkat *Software Defined Radio* (SDR) jika tidak dibekali mekanisme enkripsi yang memadai, (McWeeney et al., 2024).

Selain ancaman penyadapan, integritas data pada jaringan nirkabel jarak jauh juga menjadi perhatian kritis. Penyerang dapat melakukan manipulasi data di tengah jalur transmisi tanpa terdeteksi oleh penerima sah melalui teknik *packet injection*. Pernyataan ini didukung oleh riset yang berhasil mendemonstrasikan serangan *Man-In-The-Middle* (MITM) pada perangkat IoT berbasis LoRa, (Olazabal et al., 2022). Kerentanan ini sangat berbahaya jika diimplementasikan pada sektor kritis seperti smart metering atau infrastruktur publik, karena dapat menyebabkan data konsumsi energi atau status sensor dimanipulasi untuk keuntungan pihak tertentu, (Zaraket et al., 2020).



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Upaya pengamanan menggunakan algoritma kriptografi lightweight seperti AES-128 dan HMAC-SHA256 telah terbukti optimal untuk perangkat dengan sumber daya terbatas seperti ESP32, (Windya et al., 2021). Meskipun AES-128 mampu menjaga kerahasiaan payload, penggunaan kunci statis (*static key*) tetap menjadi titik lemah utama. Sebagaimana dijelaskan pada tinjauan keamanan jaringan berskala besar, manajemen kunci merupakan tantangan terbesar dalam IoT jangka panjang; kunci yang statis dan tidak pernah berubah sangat rentan terhadap eksploitasi melalui serangan replay dan pembongkaran kunci secara paksa, (Li & Cao, 2022). Hal ini diperkuat oleh penelitian lain yang menegaskan pentingnya mekanisme anti-replay (seperti sequence number atau timestamp) serta pembaruan kunci secara berkala untuk menjaga kepercayaan pada transmisi data rutin, (Hayati, 2024).

Oleh karena itu, diperlukan sebuah sistem mitigasi pada *custom gateway* LoRa P2P yang tidak hanya mengenkripsi data, tetapi juga mampu mengelola kunci secara dinamis tanpa harus bergantung pada koneksi internet atau server eksternal. Penelitian ini mengusulkan implementasi *dynamic key management* berbasis *Key Derivation Function* (KDF). Dengan mekanisme ini, kunci enkripsi akan diperbarui secara otomatis berdasarkan parameter dinamis, sehingga setiap sesi paket dilindungi oleh kunci sesi (*session key*) yang berbeda-beda, (Hakeem et al., 2021). Pendekatan ini diharapkan dapat memberikan perlindungan berlapis yang resilien terhadap serangan MITM, *sniffing*, dan *replay attack* pada jaringan LoRa P2P di area offline.

1.2 Perumusan Masalah

Berdasarkan identifikasi masalah yang telah dipaparkan pada latar belakang, terutama mengenai kerentanan komunikasi LoRa terhadap serangan MITM dan penggunaan kunci statis, maka perumusan masalah dalam penelitian ini adalah:

1. Bagaimana merancang dan mengimplementasikan mekanisme *dynamic key management* menggunakan metode *Key Derivation Function* (KDF) pada arsitektur jaringan LoRa P2P berbasis ESP32?
2. Sejauh mana tingkat efektivitas skema keamanan yang diusulkan dalam mendeteksi dan menolak gangguan yang berasal dari serangan *sniffing*, *packet injection* (MITM), dan *replay attack*?



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

3. Bagaimana dampak implementasi pengamanan (KDF, AES-128, dan HMAC-SHA256) terhadap performa mikrokontroler ESP32, khususnya pada indikator waktu pemrosesan komputasi (latensi) dan pemakaian memori (RAM usage)?

1.3 Batasan Masalah

Untuk mengantisipasi meluasnya pembahasan dan agar penelitian ini tetap terfokus pada solusi yang direncanakan, maka ditetapkan batasan masalah sebagai berikut:

1. Perangkat keras difokuskan pada penggunaan mikrokontroler ESP32 dan modul transceiver LoRa Ra-02 (SX1278) yang beroperasi pada frekuensi 433 MHz.
2. Implementasi sistem keamanan pada *custom gateway* LoRa P2P ini difokuskan pada studi kasus lingkungan *Industrial Internet of Things (IIoT)* atau *Smart Agriculture* yang berada di area blank spot tanpa infrastruktur LoRaWAN komersial.
3. Sistem pengamanan menggunakan algoritma kriptografi *lightweight* AES-128 untuk confidentiality, HMAC-SHA256 untuk *data integrity*, serta fungsi hash KDF untuk pembaharuan *session key*.
4. Parameter pengujian performa mikrokontroler (ESP32) dibatasi pada evaluasi waktu pemrosesan komputasi (*computational latency*), penggunaan memori (RAM) akibat penambahan beban komputasi dari algoritma kriptografi.

1.4 Tujuan dan Manfaat

Tujuan penelitian mengungkapkan sasaran yang ingin dicapai, sedangkan manfaat berisikan uraian hasil positif dari dilaksanakannya penelitian ini. Tujuan Penelitian Adapun tujuan yang ingin dicapai melalui pelaksanaan penelitian ini adalah:

1. Mengimplementasikan protokol pembaruan kunci secara dinamis (*dynamic key management*) guna menghindari penggunaan kunci statis yang berisiko tinggi pada jaringan LoRa P2P.
2. Menguji dan membuktikan ketahanan sistem *gateway* secara eksperimental dalam memblokir paket data tidak sah yang dikirimkan oleh pihak penyerang dalam skenario MITM.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

3. Menganalisis parameter performa secara kuantitatif untuk membuktikan kelayakan implementasi berlapis keamanan ini pada perangkat IoT dengan sumber daya yang sangat terbatas.
4. Menganalisis performa sistem keamanan yang diusulkan berdasarkan parameter keberhasilan penangkalan serangan, waktu komputasi (*computation overhead*), penggunaan memori pada perangkat dengan sumber daya terbatas (*constrained device*).

Melalui pencapaian tujuan tersebut, penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Menyediakan solusi infrastruktur Gateway LoRa yang efisien biaya (*low-cost*) dan aman, sehingga dapat dijangkau dan diterapkan oleh industri kecil atau sektor pertanian di area tanpa sinyal internet (*blank spot*).
2. Menghasilkan sistem keamanan yang terbukti mampu melindungi data sensor dari ancaman nyata seperti penyadapan (*sniffing*), manipulasi data (*Man-In-The-Middle*), dan pengiriman ulang paket (*replay attack*).
3. Memberikan referensi teknis mengenai penerapan algoritma kriptografi hibrida (AES-128 dan HMAC-SHA256) serta manajemen kunci dinamis yang optimal pada perangkat mikrokontroler dengan sumber daya terbatas.

1.5 Sistematika Penulisan

Sistematika penulisan skripsi ini disusun untuk memberikan gambaran menyeluruh mengenai alur pembahasan penelitian. Adapun sistematika penulisannya adalah sebagai berikut:

1. BAB I Pendahuluan

Bab ini menguraikan mengenai latar belakang pemilihan topik penelitian, identifikasi masalah terkait keamanan komunikasi LoRa P2P, perumusan masalah, batasan masalah, tujuan yang ingin dicapai, serta manfaat penelitian bagi pengembang sistem IoT dan instansi terkait.

2. BAB II Tinjauan Pustaka

Bab ini memaparkan teori-teori dasar yang menjadi landasan penelitian, meliputi teknologi LoRa P2P, algoritma kriptografi AES-128 dan HMAC-SHA256, mekanisme *Counter-based Nonce*, serta konsep *Key Derivation*



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Function (KDF). Selain itu, bab ini juga menyajikan tinjauan terhadap penelitian-penelitian relevan terdahulu.

3. BAB III Metodologi Penelitian

Bab ini menjelaskan mengenai kerangka pemikiran dan prosedur penelitian yang meliputi tahapan identifikasi masalah, studi literatur, perancangan sistem keamanan, teknik pengumpulan data melalui simulasi serangan MITM, serta teknik analisis data untuk menguji performa dan keamanan protokol yang dikembangkan.

4. BAB IV Hasil dan Pembahasan

Bab ini berisi tentang hasil dari implementasi protokol pada perangkat ESP32, hasil pengujian mitigasi serangan, serta analisis terhadap parameter waktu proses dan penggunaan memori pada perangkat.

5. BAB V Penutup

Bab ini berisi kesimpulan dari seluruh rangkaian penelitian yang telah dilakukan serta saran-saran bagi pengembangan penelitian selanjutnya di masa yang akan datang.

**POLITEKNIK
NEGERI
JAKARTA**

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB II

TINJAUAN PUSTAKA

2.1 *Internet of Things* (IoT)

Secara fundamental, *Internet of Things* (IoT) merupakan sebuah paradigma di mana segala sesuatu yang dikategorikan sebagai objek fisik atau "*things*" saling terhubung dan dapat berkomunikasi satu sama lain melalui jaringan internet (Windya et al., 2021). Ekosistem IoT memanfaatkan pemrograman untuk menghasilkan interaksi antar mesin (*Machine-to-Machine*) yang memungkinkan pengumpulan informasi dan pengambilan keputusan secara otomatis di mana saja dan kapan saja tanpa memerlukan campur tangan manusia secara langsung (Lestari et al., 2020). Perangkat fisik ini umumnya tertanam dengan berbagai sensor, perangkat lunak, dan teknologi lain untuk tujuan pertukaran data. Perkembangan pesat *Internet of Things* (IoT) dalam beberapa dekade terakhir telah memperluas jangkauan konektivitas perangkat ke berbagai sektor kritis, mulai dari otomasi industri hingga pertanian presisi (Li & Cao, 2022). Seiring dengan meningkatnya skala implementasi tersebut, teknologi *Low Power Wide Area Network* (LPWAN) muncul sebagai solusi utama untuk mengatasi keterbatasan jangkauan dari jaringan nirkabel konvensional (Z. Sun et al., 2022). Di antara berbagai teknologi LPWAN yang ada, LoRa (*Long Range*) menjadi protokol yang paling menjanjikan dan banyak diadopsi karena kemampuannya mentransmisikan data dalam jarak jauh (hingga belasan kilometer) dengan konsumsi energi yang sangat minim (McWeeney et al., 2024).

Dalam arsitektur IoT jarak jauh, terutama yang diaplikasikan di wilayah minim infrastruktur (*blank spot*), *gateway* memegang peranan krusial sebagai jembatan penghubung antara perangkat sensor (*end-node*) yang berada di lapangan dengan server pusat atau *cloud*. Tanpa adanya *gateway*, data dari sensor yang menggunakan protokol hemat daya seperti LoRa tidak dapat diteruskan ke jaringan internet global. Namun, tantangan utama dalam implementasi IoT skala luas, khususnya bagi industri kecil dan menengah, adalah tingginya biaya infrastruktur. Perangkat *gateway* LoRaWAN standar industri yang tersedia di pasaran memiliki harga yang relatif mahal (*expensive*), sehingga menjadi hambatan adopsi teknologi ini bagi pelaku usaha rintisan (*startup*) atau sektor pertanian (C. Sun et al., 2020). Sebagai



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

alternatif, konsep *gateway* berbiaya rendah (*low-cost gateway*) atau sering disebut sebagai *gateway* rancangan mandiri (*custom gateway*) mulai banyak dikembangkan. *Gateway* mandiri yang dibangun menggunakan perangkat komputasi terjangkau seperti Raspberry Pi atau ESP32 telah divalidasi mampu menjalankan fungsi krusial *gateway* yaitu menerima data dari sensor dan meneruskannya ke internet dengan efektivitas yang setara dengan solusi komersial namun dengan biaya investasi yang jauh lebih rendah (Guimarães, 2024).

Selain infrastruktur *gateway*, komponen utama lain dalam ekosistem IoT adalah perangkat ujung (*end-node*) yang bertugas melakukan *sensing* di lapangan. Salah satu karakteristik fundamental sekaligus tantangan terbesar dari perangkat keras penyusun IoT ini adalah sangat terbatasnya sumber daya komputasi dan energi yang dimiliki (Windya et al., 2021). Perangkat dengan limitasi ekstrem pada kapasitas memori (RAM dan *Flash*/ROM), daya pemrosesan komputasi, serta pasokan energi baterai ini diklasifikasikan sebagai perangkat terbatas atau *constrained devices* (*low-end IoT devices*) menurut standar *Internet Engineering Task Force* (IETF) dalam dokumen RFC 7228. Berdasarkan kapasitas memorinya, IETF mengklasifikasikan *constrained devices* ini ke dalam tiga kelas utama (Windya et al., 2021):

1. Class 0 (C0) merupakan perangkat dengan sumber daya paling kecil, yang memiliki memori RAM di bawah 10 kiB dan ruang penyimpanan *Flash* di bawah 100 kiB. Perangkat di kelas ini umumnya sangat terbatas dan tidak dapat menjalankan protokol keamanan jaringan standar secara utuh.
2. Class 1 (C1) merupakan perangkat dengan ukuran RAM sekitar 10 kiB dan *Flash* sekitar 100 kiB. Perangkat ini sudah mampu berkomunikasi dengan protokol yang dioptimalkan secara khusus untuk IoT.
3. Class 2 (C2) merupakan perangkat dengan ukuran RAM sekitar 50 kiB dan *Flash* sekitar 250 kiB. Perangkat ini mampu menjalankan protokol komunikasi standar secara utuh, namun tetap membutuhkan efisiensi beban komputasi yang tinggi.

Keterbatasan sumber daya komputasi dan memori pada *constrained devices* inilah yang memunculkan tantangan besar dari sisi keamanan data. Algoritma kriptografi standar yang biasanya digunakan di jaringan komputer konvensional tidak dapat



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

begitu saja diterapkan pada IoT karena memakan terlalu banyak memori dan mempercepat habisnya baterai. Hal ini menuntut pengembang sistem untuk merancang dan memilih metode keamanan serta protokol transmisi yang sangat ringan (*lightweight cryptography*) agar parameter keamanan yang krusial, seperti kerahasiaan (*confidentiality*), integritas (*integrity*), dan autentikasi (*authentication*), tetap dapat tercapai dengan sempurna tanpa melampaui batas kapabilitas mikrokontroler perangkat.

2.1.1 End-Node

Secara fundamental, end-node merupakan perangkat dasar yang beroperasi di lapisan fisik untuk memproses dan mengirimkan data secara nirkabel di dalam jaringan. Pada topologi komunikasi LoRa P2P, end-node difungsikan dan beroperasi dalam mode slave yang secara penuh bertanggung jawab atas transmisi sinyal dari lapangan serta penerimaan instruksi (Zhang et al., 2022). Dalam implementasinya di lapangan, perangkat end-node sangat dibatasi dalam hal ketersediaan sumber daya, seperti ukuran memori (RAM dan *Flash*), kemampuan komputasi, serta kapasitas daya. Karakteristik ini membuat perangkat end-node secara teknis diklasifikasikan ke dalam kategori perangkat terbatas atau *constrained devices* (Windya et al., 2021).

2.1.2 Gateway

Komunikasi nirkabel LoRa memiliki batasan teknis di mana sinyal radio tidak dapat mentransmisikan data secara langsung ke network server atau internet, sehingga membutuhkan kehadiran perangkat perantara berupa gateway (Rosyida Zain et al., 2022). Gateway memegang peranan sebagai komponen kunci yang bertanggung jawab untuk mengumpulkan informasi dari sensor, melakukan prapemrosesan, dan meneruskannya ke pusat data (C. Sun et al., 2020). Dalam implementasi jaringan LoRa P2P, perangkat gateway ini beroperasi sebagai master node yang mengontrol aliran perintah dan menentukan tindakan di dalam jaringan (Zhang et al., 2022). Mengingat perannya yang vital dalam ekosistem IoT, gateway menjadi tautan penting yang berfungsi mewujudkan interkoneksi langsung antara Wide-Area Network (WAN) dengan Local-Area Network (LAN) (Lv, 2021).



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.1.3 Attacker Node

Transmisi data pada lapisan fisik LoRa P2P pada dasarnya tidak memiliki enkripsi bawaan, sehingga kerahasiaan datanya bersifat opsional dan dapat disadap serta dibaca secara langsung oleh siapa saja yang berada pada frekuensi yang sama (McWeeney et al., 2024; Ramadhani et al., 2025). Dalam simulasi ancaman jaringan, peretas ini direpresentasikan sebagai *attacker node*. Perangkat penyerang tersebut mempraktikkan proses *sniffing* untuk memonitor dan menangkap semua lalu lintas paket jaringan yang lewat tanpa memedulikan kepada siapa paket tersebut sebenarnya ditujukan (Lestari et al., 2020). Berbekal lalu lintas data nirkabel yang terbuka itu, *attacker node* mampu meluncurkan serangan *Man-In-The-Middle* (MITM) untuk menginterupsi jalur komunikasi secara aktif, mencuri informasi sensitif, menyamar sebagai salah satu entitas sah, hingga memanipulasi data untuk mengeksploitasi sistem (Olazabal et al., 2022).

2.2 Teknologi LoRa (*Long Range*)



Gambar 2.1 LoRa Technologies

Sumber: (Semtech, n.d.)

LoRa (*Long Range*) adalah teknologi modulasi nirkabel turunan dari *Chirp Spread Spectrum* (CSS) yang dikembangkan oleh Semtech. Teknologi ini beroperasi pada lapisan fisik (*Physical Layer*) dan dirancang khusus untuk komunikasi jarak jauh dengan konsumsi daya yang sangat rendah, namun dengan konsekuensi laju data (*data rate*) yang kecil (Sun et al., 2022). Karakteristik utama LoRa adalah kemampuannya mempertahankan komunikasi di bawah tingkat kebisingan (*noise floor*) berkat teknik modulasi CSS, menjadikannya ideal untuk aplikasi IoT di area dengan banyak interferensi atau penghalang fisik (Li & Cao, 2022).

Penting untuk membedakan antara LoRa dan LoRaWAN. LoRa adalah lapisan fisik (modulasi sinyal), sedangkan LoRaWAN adalah protokol jaringan (*MAC Layer*) yang mengatur komunikasi antar perangkat secara terstandarisasi (Sun et al., 2022). Dalam implementasi *gateway* berbiaya rendah dan jaringan mandiri (*private*



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

network), penggunaan protokol LoRaWAN standar sering kali terlalu kompleks dan membutuhkan infrastruktur server yang membebani (Zain et al., 2021). Oleh karena itu, penelitian ini memanfaatkan modulasi LoRa pada mode "mentah" (*raw*) atau *transparent mode* untuk membangun komunikasi *point-to-point* (P2P) antara *node* dan *gateway* tanpa ketergantungan pada penyedia layanan jaringan LoRaWAN.

Secara teknis, performa transmisi dan modulasi LoRa pada lapisan fisik sangat bergantung pada konfigurasi empat parameter utama (Li & Cao, 2022), yaitu:

1. *Spreading Factor* (SF) menentukan jumlah kepingan (*chips*) per simbol yang dikirimkan. SF dapat diatur mulai dari SF7 hingga SF12 (Sun et al., 2022). Nilai SF yang lebih tinggi (misalnya SF12) akan memperpanjang durasi sinyal di udara (*Time-on-Air*) sehingga jangkauan komunikasi lebih jauh dan ketahanan terhadap *noise* meningkat. Namun, hal ini secara signifikan akan menurunkan laju data (*data rate*) serta meningkatkan konsumsi energi perangkat (Sun et al., 2022).
2. *Bandwidth* (BW) merupakan rentang frekuensi transmisi yang digunakan (umumnya 125 kHz, 250 kHz, atau 500 kHz). *Bandwidth* yang lebih lebar memberikan *data rate* yang lebih tinggi, namun rentan terhadap *noise* (Li & Cao, 2022).
3. *Coding Rate* (CR) merupakan tingkat penerapan *Forward Error Correction* (FEC) yang digunakan untuk mendeteksi dan mengoreksi *error* pada paket data yang diterima. Peningkatan nilai CR memberikan perlindungan ekstra terhadap interferensi, namun menambah panjang ukuran *payload* di udara (Sun et al., 2022).
4. *Transmission Power* (TP) merupakan besaran daya pancar sinyal radio yang digunakan oleh modul, yang berbanding lurus dengan jarak jangkauan komunikasi dan energi yang dikonsumsi (Li & Cao, 2022).



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.2.1 Modul LoRa Ra-02 (SX1278)



Gambar 2.2 LoRa Ra-02

Sumber: (Designer, n.d.)

Dalam perancangan sistem gateway berbiaya rendah, pemilihan modul transceiver menjadi faktor kunci. Penelitian ini menggunakan modul LoRa seri Ra-02 yang terintegrasi dengan chip Semtech SX1278. Berbeda dengan seri lain yang beroperasi di frekuensi tinggi, chip SX1278 dapat bekerja optimal pada pita frekuensi 433 MHz yang umum digunakan untuk berbagai aplikasi pada pita Industrial, Scientific, and Medical (ISM) (Sun et al., 2020).

Modul SX1278 memiliki sensitivitas penerimaan yang sangat tinggi hingga mencapai -148 dBm, serta mampu menghasilkan daya keluaran frekuensi radio (RF output power) hingga +20 dBm. Kombinasi sensitivitas tinggi dan daya pancar ini memungkinkan modul untuk mempertahankan keseimbangan optimal antara jarak jangkauan komunikasi yang jauh dan ketahanan terhadap interferensi. Untuk dapat berinteraksi dan dikendalikan oleh mikrokontroler (seperti ESP32), modul SX1278 memanfaatkan antarmuka Serial Peripheral Interface (SPI) dengan menggunakan pin-pin pendukung seperti MOSI, MISO, SCK, dan NSS guna memfasilitasi pertukaran data secara cepat.

2.2.2 Perbandingan Arsitektur LoRaWAN dan LoRa P2P

Perbedaan mendasar antara LoRaWAN dan LoRa *Peer-to-Peer* (P2P) terletak pada topologi dan lapisan protokolnya. Secara arsitektural, LoRaWAN adalah protokol jaringan (*MAC Layer*) yang mengadopsi topologi bintang (*star*), di mana *end-device* di lapangan harus berkomunikasi melalui sebuah *gateway* yang kemudian wajib meneruskan paket data tersebut ke sebuah *Network Server* terpusat menggunakan koneksi IP atau internet (Zain et al., 2021). LoRaWAN



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

menawarkan fasilitas manajemen perangkat yang lengkap serta menyediakan kerangka keamanan bawaan yang terstandarisasi melalui enkripsi di lapisan jaringan dan aplikasi (Simões et al., 2025).

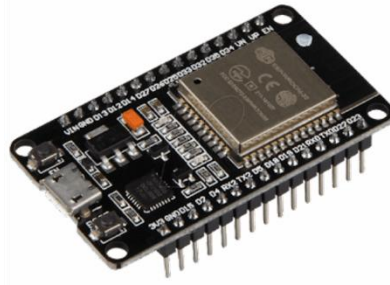
Sebaliknya, jaringan LoRa P2P memungkinkan *end-device* untuk berkomunikasi secara langsung (*single-hop*) satu sama lain murni di lapisan fisik tanpa memerlukan infrastruktur *gateway* yang terhubung ke internet maupun *network server*. Arsitektur P2P ini sangat menguntungkan secara biaya instalasi dan sangat cocok untuk diimplementasikan pada wilayah yang terisolasi (*blank spot*). Kendati demikian, kemandirian infrastruktur ini membawa tantangan tersendiri. Karena LoRa secara fisik (*mode raw*) tidak mengenkripsi data *payload* yang ditransmisikan, maka semua data dikirimkan dalam bentuk terbuka atau *plaintext* (Simões et al., 2025). Akibatnya, setiap perangkat penerima keras (seperti alat *Software Defined Radio* atau *sniffer*) yang berada dalam jangkauan sinyal dapat dengan mudah menyadap transmisi tersebut (Simões et al., 2025). Hal ini membuat jaringan LoRa P2P sangat rentan terhadap penyadapan dan manipulasi data (*Man-In-The-Middle*) jika lapisan keamanannya tidak dirancang secara mandiri oleh pengembang pada tingkat aplikasi.

2.3 Mikrokontroler

Mikrokontroler adalah sebuah sistem komputer fungsional dalam sebuah *integrated chip* (IC) yang berisi prosesor utama (CPU), memori (RAM/ROM), dan antarmuka input/output (I/O). Berbeda dengan mikroprosesor pada komputer konvensional yang membutuhkan komponen eksternal, mikrokontroler didesain secara khusus untuk sistem tertanam (*embedded system*) guna menjalankan fungsi kontrol spesifik secara otomatis. Dalam ekosistem *Internet of Things* (IoT), mikrokontroler berfungsi sebagai otak utama yang memproses akuisisi data dari sensor, menjalankan algoritma enkripsi, dan mengontrol modul komunikasi radio secara terintegrasi (Windya et al., 2021).



2.3.1 ESP32



Gambar 2.3 ESP32

Sumber: (Instiperjogja, 2021)

ESP32 adalah mikrokontroler berbiaya rendah (*low-cost*) dan hemat daya yang dirancang khusus untuk memenuhi kebutuhan implementasi jaringan IoT masa kini. Kemampuan mikrokontroler ini telah banyak divalidasi keandalannya dalam membangun infrastruktur jaringan secara mandiri. ESP32 sangat tangguh dan efektif digunakan sebagai otak komputasi utama, baik pada titik pengirim data (*transmitter end-node*) maupun pada titik penerima (*receiver gateway*) yang disandingkan dengan modul komunikasi LoRa (Setiowati et al., 2023).

Pemilihan ESP32 sebagai perangkat keras utama dalam penelitian ini juga didasarkan pada spesifikasi memori dan kapabilitas pemrosesannya. Dalam perancangan keamanan komunikasi nirkabel, keterbatasan kapasitas memori (seperti RAM dan Flash/ROM) pada sebuah perangkat terbatas (*constrained device*) menjadi faktor krusial yang sering kali menghambat eksekusi algoritma keamanan (Windya et al., 2021). Perangkat dengan sumber daya yang terlalu kecil (seperti mikrokontroler Kelas 0) akan mudah mengalami kegagalan sistem (*self restart*) apabila dipaksa untuk menjalankan algoritma enkripsi secara utuh (Windya et al., 2021).

Dengan kapasitas memori dan daya komputasi yang memadai pada arsitektur ESP32, proses pengamanan *payload* data sensor menggunakan algoritma ringan seperti *Advanced Encryption Standard* (AES) dan fungsi *Hash* dapat dieksekusi dengan efisien. Hal ini memastikan bahwa sistem keamanan kriptografi dapat berjalan secara optimal tanpa menyebabkan beban memori yang berlebih (*overload*) saat perangkat melakukan transmisi di jaringan LoRa P2P.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



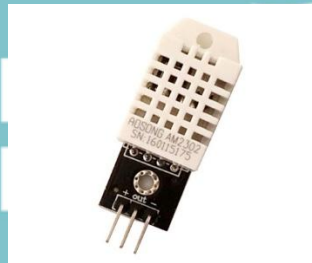
Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.4 DHT22

DHT22 merupakan sensor digital yang berfungsi untuk mengukur tingkat kelembapan relatif dan suhu udara secara bersamaan. Secara teknis, sensor ini menggunakan komponen kapasitor dan termistor untuk mengakuisisi kondisi udara di sekitarnya dan menghasilkan keluaran berupa sinyal digital pada pin datanya. Sensor DHT22 memiliki jangkauan pembacaan suhu yang luas, yaitu dari -40 hingga 80°C , dengan tingkat ketelitian (akurasi) sekitar $\pm 0.5^{\circ}\text{C}$. Kualitas pembacaan sensor ini dinilai sangat baik, dengan persentase rata-rata nilai kesalahan (error) yang sangat rendah jika dibandingkan dengan alat ukur thermohygrometer standar, yakni hanya sebesar 2,99% untuk kelembapan dan -2,31% untuk suhu (Puspasari et al., 2020).

Pada penelitian ini, sensor DHT22 diimplementasikan sebagai komponen input utama pada perangkat *end-node* pengirim. Data suhu dan kelembapan aktual yang dibaca oleh sensor ini akan bertindak sebagai data mentah (*plaintext*) atau muatan (*payload*), yang selanjutnya diproses melalui tahapan enkripsi dan pemberian tanda tangan digital sebelum ditransmisikan ke udara melalui antarmuka LoRa P2P. Bentuk fisik dari sensor DHT22 ditunjukkan pada Gambar 2.4.



Gambar 2.4 Sensor Suhu dan Kelembapan DHT22

Sumber: (Components101, 2018)

2.5 AMS1117

Integrated Circuit (IC) AMS1117 adalah komponen regulator tegangan *Low Dropout* (LDO) positif yang berfungsi sebagai *step-down direct current converter* untuk menurunkan dan menstabilkan tegangan. Varian AMS1117-3.3 dirancang untuk mengubah tegangan *input* yang lebih tinggi (seperti 5V hingga 7V) menjadi tegangan *output* 3,3V DC yang sangat stabil, serta memiliki kemampuan untuk menyalurkan rating arus keluaran hingga 1A (Artha Lesmana et al., 2023).

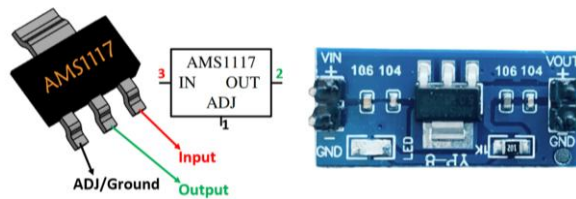


Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Regulator ini memiliki tingkat penurunan tegangan (*dropout*) yang efisien, yaitu tipikal 1,2V pada saat beroperasi dengan beban penuh 1A.

Sebagai langkah keandalan tambahan, *chip* AMS1117 juga dilengkapi dengan fitur proteksi keamanan internal terpadu, yaitu fitur pembatas arus (*current limiting*) dan pemutusan daya otomatis guna melindungi perangkat dari kerusakan akibat suhu sambungan yang berlebih (*thermal shutdown*). Dalam skripsi ini, AMS1117-3.3 diimplementasikan pada rangkaian perangkat *end-node* dan *gateway* untuk menurunkan tegangan 5V dari pin mikrokontroler ESP32 menjadi pasokan daya 3,3V DC secara eksklusif bagi modul *transceiver* LoRa Ra-02. Penggunaan regulator catu daya eksternal ini sangat krusial untuk mencegah terjadinya penurunan tegangan atau *restart* paksa pada sistem mikrokontroler akibat penarikan arus listrik yang masif pada saat proses transmisi paket radio sedang berlangsung. Bentuk fisik dari regulator AMS1117 ditunjukkan pada Gambar 2.5.



Gambar 2.5 Regulator Tegangan AMS1117

Sumber: (Tech, 2025)

2.6 Lightweight cryptography

Lightweight cryptography adalah algoritma kriptografi yang dirancang khusus untuk diimplementasikan pada perangkat terbatas (*resource-constrained devices*) seperti node sensor IoT, RFID, dan smart cards yang memiliki keterbatasan pada memori (RAM/ROM), daya komputasi, serta kapasitas baterai (Radhakrishnan et al., 2024).

2.6.1 Advanced Encryption Standard (AES-128)

Advanced Encryption Standard (AES) merupakan algoritma kriptografi simetris yang distandarisasi secara global untuk pengamanan data. Pada implementasi jaringan dengan sumber daya terbatas (*constrained device*) seperti perangkat *Internet of Things* (IoT), varian AES-128 jauh lebih diutamakan karena tidak



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

membutuhkan daya komputasi yang tinggi dan hemat konsumsi memori saat melakukan proses enkripsi maupun dekripsi (McWeeney et al., 2024).

Untuk memaksimalkan performa pada transmisi radio LoRa P2P, algoritma AES-128 dalam penelitian ini dioperasikan menggunakan mode *Counter* (CTR). Mode ini secara efektif mengubah karakteristik *block cipher* menjadi sebuah *stream cipher* dengan cara mengenkripsi sebuah nilai *counter* yang terus meningkat, yang kemudian aliran *cipher* tersebut di-XOR dengan *plaintext* untuk menghasilkan *ciphertext* akhir (McWeeney et al., 2024).

Penggunaan mode CTR memiliki keuntungan performa yang sangat signifikan bagi komunikasi nirkabel karena dapat memproses input data dengan panjang berapapun tanpa memerlukan penambahan *padding* atau bit pelengkap. Ketidadaan *padding* ini menjaga ukuran *payload* pesan tetap kecil sehingga kecepatan transmisi di udara meningkat dan konsumsi daya perangkat dapat ditekan. Selain itu, penggunaan *counter* yang selalu berubah memastikan bahwa *plaintext* yang sama tidak akan menghasilkan *ciphertext* yang berulang, sehingga menyulitkan peretas untuk mendeteksi pola data di udara. Namun, mode CTR ini menuntut solusi manajemen kunci dan *counter* yang sangat tangguh agar perangkat pengirim dan penerima tetap sinkron dan tidak mengalami kegagalan dekripsi saat paket data ditransmisikan (McWeeney et al., 2024).

2.6.2 Hash-based Message Authentication Code (HMAC-SHA256)

Meskipun enkripsi mampu menyembunyikan isi data dari pihak luar, mekanisme tambahan tetap wajib diterapkan untuk mencegah manipulasi pesan. *Hash-based Message Authentication Code* (HMAC) adalah prosedur kriptografi yang secara spesifik digunakan untuk memverifikasi bahwa pesan yang diterima di tujuan benar-benar utuh dan berasal dari sumber yang otentik (Windya et al., 2021).

Mekanisme HMAC bekerja dengan cara menggabungkan fungsi *hash* dengan sebuah kunci rahasia (*secret key*) untuk menghasilkan tanda tangan digital pada pesan. Jika terdapat pihak penyerang yang mencoba menyisipkan atau memodifikasi satu bit saja dari paket data di udara, nilai *hash* yang dihasilkan pada titik penerima akan berbeda secara drastis dengan nilai *hash* yang dikirimkan. Ketidacocokan tanda tangan digital ini memungkinkan sistem penerima untuk langsung menggugurkan paket tersebut, sehingga lalu lintas data



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

aman dari ancaman manipulasi pesan atau serangan *Man-In-The-Middle* di jaringan terbuka (Windya et al., 2021).

2.6.3 Manajemen Kunci Dinamis (*Dynamic Key Management*)

Kelemahan struktural dalam arsitektur keamanan perangkat IoT konvensional adalah penggunaan kunci statis (*static key*) yang ditanamkan secara permanen di dalam memori. Penggunaan kunci yang sama secara terus-menerus sangat rentan terhadap eksploitasi, di mana kebocoran satu kunci akan meruntuhkan seluruh kerahasiaan data secara permanen (Li & Cao, 2022). Hal ini semakin kritis mengingat mode AES-CTR membutuhkan manajemen nilai *counter* dan kunci yang sangat terstruktur agar proses sinkronisasi komunikasi tidak gagal (McWeeney et al., 2024).

Untuk mengatasi tantangan dan celah kerentanan tersebut, sistem ini mengimplementasikan pembaruan kunci secara dinamis menggunakan *Key Derivation Function* (KDF). Skema kriptografi ini bekerja dengan cara menurunkan kunci sesi (*session key*) baru secara berkala dari kunci utama dengan menginkorporasikan parameter dinamis berupa angka urut (*sequence number*) yang terus bertambah pada setiap transmisi (Hayati, 2024). Pendekatan KDF ini memungkinkan perangkat *end-node* dan *gateway* untuk selalu memperbarui kunci enkripsi mereka secara mandiri tanpa harus menukarkan kunci tersebut melalui jaringan nirkabel yang rentan disadap.

Penggabungan parameter *counter* yang selalu meningkat ini tidak hanya menjamin keunikan kunci pada setiap sesi, tetapi juga berfungsi ganda sebagai *nonce* (nilai yang hanya digunakan sekali) untuk memblokir serangan pengiriman ulang paket atau *replay attack* (Hayati, 2024). Dengan mekanisme pencatatan *counter* ini, *gateway* akan secara otomatis menolak dan membuang paket usang yang dikirimkan ulang oleh peretas karena paket tersebut memiliki nilai *counter* yang sama atau lebih rendah dari *counter* yang telah diterima sebelumnya.

Lebih dalam secara teoritis, metode *Key Derivation Function* (KDF) sangat ideal untuk diimplementasikan pada arsitektur jaringan berbasis perangkat terbatas (*constrained devices*). KDF mampu menyegarkan kunci sesi (*session key*) secara mandiri tanpa menggunakan operasi kriptografi asimetris (*public-key*) yang membebani daya pemrosesan mikrokontroler (Rakshit et al., 2025).



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Penerapan *entropy mixing* (pencampuran entropi) di dalam fungsi KDF ini menjamin terciptanya isolasi kunci per sesi (*per-session key isolation*), yang berarti setiap sesi komunikasi memiliki kunci independen yang terisolasi sepenuhnya dari sesi-sesi lainnya. Hal ini secara efektif mencegah sistem menggunakan *root keys* atau kunci utama jangka panjang secara langsung untuk melakukan enkripsi *payload*. Metode ini juga memastikan tercapainya properti keamanan *Perfect Forward Secrecy*, yaitu sebuah kondisi di mana kompromi atau kebocoran pada kunci utama tidak akan pernah mengekspos kunci-kunci sesi yang telah digunakan di masa lalu (Hakeem et al., 2021).

Rotasi dan pembaruan kunci rahasia secara rutin melalui KDF ini pada akhirnya merupakan mekanisme pertahanan yang sangat krusial. Selain menggagalkan teknik peretasan analisis pola, pembaruan kunci secara berkelanjutan memastikan bahwa jika terjadi suatu eksploitasi keamanan, dampak dari akses tidak sah tersebut akan sangat dibatasi rentang waktunya (*timeframe*), sehingga solusi ini sangat cocok untuk melindungi perangkat *Internet of Things* yang dibiarkan beroperasi dalam jangka waktu lama di area terbuka (Rana et al., 2023).

2.7 Key Derivation Function (KDF)

Key Derivation Function (KDF) merupakan sebuah skema kriptografi yang berfungsi untuk menghasilkan atau menurunkan kunci sesi (*session key*) baru secara berkala dari sebuah kunci utama. Pembaruan kunci secara dinamis pada sistem ini tidak dilakukan melalui pertukaran kunci baru di udara yang rentan disadap, melainkan dengan derivasi kunci menggunakan skema KDF berbasis algoritma HMAC-SHA256. Proses ini menghasilkan kunci sesi sementara (*ephemeral per-session encryption keys*) secara lokal dan independen di masing-masing perangkat pengirim dan penerima tanpa memerlukan operasi asimetris yang berat bagi mikrokontroler. Secara konseptual, logika fungsi KDF untuk menghasilkan kunci sesi yang baru dapat direpresentasikan ke dalam Persamaan 2.1 berikut.

$$K_{\text{session}} = [\text{HMAC SHA256}(K_{\text{master}}, \text{packetId})]_{0 \dots 15} \quad (\text{Persamaan 2.1})$$

Berdasarkan persamaan tersebut, KDF bekerja dengan cara menginkorporasikan parameter dinamis berupa angka urut transmisi (*Sequence Counter* atau ID paket) ke dalam *Master Key* statis sebagai materi pencampuran. Penerapan pencampuran



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

entropi (*entropy mixing*) di dalam fungsi KDF ini menjamin terciptanya isolasi kunci per sesi (*per-session key isolation*), yang berarti setiap sesi komunikasi memiliki kunci independen yang terisolasi sepenuhnya dari sesi-sesi lainnya.

Metode ini juga memastikan tercapainya properti keamanan *Perfect Forward Secrecy*, yaitu sebuah kondisi pertahanan di mana kompromi atau kebocoran pada kunci utama tidak akan pernah mengekspos kunci-kunci sesi yang telah digunakan di masa lalu. Rotasi dan pembaruan kunci rahasia secara rutin melalui KDF ini pada akhirnya merupakan mekanisme pertahanan yang sangat krusial. Selain menggagalkan teknik peretasan analisis pola sandi di udara, pembaruan kunci secara berkelanjutan memastikan bahwa dampak dari eksploitasi keamanan akan sangat dibatasi rentang waktunya (*timeframe*), sehingga sangat cocok diterapkan pada arsitektur perangkat *Internet of Things* (IoT).

2.8 Perangkat Lunak Pendukung

2.8.1 Visual Studio Code (VS Code)



Gambar 2.6 Visual Studio Code

Sumber: (Wikipedia, n.d.-b)

Visual Studio Code (VS Code) adalah sebuah teks editor yang ringan dan andal yang dirancang untuk dapat beroperasi pada sistem operasi lintas platform (multiplatform), seperti Linux, Mac, dan Windows (Ningsih et al., 2022). Perangkat lunak ini secara langsung mendukung bahasa pemrograman dasar dan dapat diekstensi untuk mendukung bahasa pemrograman lainnya seperti C, C++, Python, dan PHP melalui pemasangan *plugin* yang tersedia secara resmi di marketplace Visual Studio Code (Ningsih et al., 2022). Dalam perancangan sistem ini, VS Code difungsikan sebagai antarmuka utama penulisan kode sumber.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2.8.1.1 PlatformIO



Gambar 2.7 PlatformIO

Sumber: (PlatformIO, n.d.)

PlatformIO merupakan lingkungan pengembangan lintas platform yang dirancang secara khusus untuk menyederhanakan kompleksitas dalam pengembangan sistem tertanam (*embedded systems*) dan *Internet of Things* (IoT) (Shkoropad & Beshley, 2025). Platform ini memberikan kemudahan tingkat lanjut dalam manajemen dependensi perangkat lunak, otomatisasi proses kompilasi, serta integrasi yang sangat fleksibel dengan platform perangkat keras, khususnya mikrokontroler keluarga ESP32 (Shkoropad & Beshley, 2025).

2.8.1.2 Bahasa Pemrograman C++



Gambar 2.8 C++

C++ merupakan bahasa pemrograman tingkat menengah yang difungsikan sebagai bahasa utama untuk mengembangkan kode program (*firmware*) pada mikrokontroler ESP32. Visual Studio Code dapat diekstensi secara langsung untuk mendukung penulisan dan analisis sintaks bahasa C++ secara komprehensif melalui pemasangan *plugin* yang tersedia secara resmi di *marketplace* (Ningsih et al., 2022).

Dalam ekosistem pengembangan *Internet of Things* (IoT) pada penelitian ini, penulisan kode sumber berbasis C++ dilakukan di dalam lingkungan Visual Studio Code yang diintegrasikan dengan ekstensi PlatformIO guna menyederhanakan kompleksitas pengembangan sistem tertanam (*embedded*



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

systems) serta integrasi yang sangat fleksibel dengan mikrokontroler keluarga ESP32 (Shkoropad & Beshley, 2025). Penggunaan C++ sangat krusial karena instruksinya dieksekusi secara langsung (*low-level*) oleh ESP32 untuk mengelola komunikasi perangkat keras dan mengatur lapisan fisik radio LoRa. Selain itu, optimalisasi C++ dibutuhkan untuk mengeksekusi operasi algoritma *lightweight cryptography* secara efisien pada perangkat yang memiliki keterbatasan sumber daya komputasi dan memori (Radhakrishnan et al., 2024; Windya et al., 2021).

2.8.2 Laragon



Gambar 2.9 Laragon

Sumber: (Kreaweb, 2023)

Laragon adalah sebuah aplikasi yang dirancang untuk menyediakan lingkungan pengembangan web lokal (*local development environment*) yang tangguh, kuat, sekaligus mudah digunakan (Palcomtech, 2024). Dengan menggunakan Laragon, pengembang perangkat lunak dapat membuat peladen web lokal (*local web server*), mengelola basis data, dan menguji coba aplikasi secara independen tanpa perlu terhubung ke jaringan internet. Platform ini banyak dipilih karena kinerjanya yang sangat cepat, konsumsi sumber dayanya yang rendah, serta kemampuannya dalam mendukung berbagai basis data (Palcomtech, 2024).

2.8.2.1 MySQL



Gambar 2.10 MySQL

Sumber: (Wikipedia, n.d.-a)



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

MySQL merupakan suatu perangkat lunak sistem manajemen basis data relasional atau *Relational Database Management System* (RDBMS) yang didistribusikan secara gratis di bawah lisensi *General Public License* (GPL) (Ningsih et al., 2022). Sistem ini bekerja dengan cara menyusun data dan informasi yang kompleks menjadi beberapa kelompok dengan tipe data yang sejenis dalam bentuk tabel (entitas), di mana setiap datanya dapat saling berhubungan satu sama lain sehingga mudah dikelola dan diakses untuk keperluan sistem perekaman data (*logging*) (Ningsih et al., 2022).

2.9 Penelitian Terdahulu

Penelitian ini dikembangkan berdasarkan tinjauan terhadap beberapa studi terdahulu yang berfokus pada efisiensi infrastruktur dan keamanan jaringan komunikasi LoRa. Pengiriman data menggunakan infrastruktur LoRaWAN terbukti memiliki ketergantungan yang sangat tinggi pada ketersediaan koneksi internet dan jaringan *cloud server* untuk meneruskan paket data dari *gateway* (Rosyida Zain et al., 2022). Sebagai solusi untuk menekan mahalnya biaya infrastruktur *gateway* komersial tersebut, perancangan *single-channel* LoRa *gateway* berbiaya rendah (*low-cost*) berbasis perangkat mandiri telah banyak diimplementasikan (C. Sun et al., 2020). Kendati demikian, pengembangan *gateway* mandiri tersebut masih berfokus murni pada aspek efisiensi biaya dan membiarkan komunikasi data berjalan secara terbuka (*raw*) tanpa adanya lapisan enkripsi (C. Sun et al., 2020). Komunikasi nirkabel LoRa yang berjalan secara terbuka pada lapisan fisik memiliki kerentanan yang sangat fatal terhadap eksploitasi peretas. Jaringan LoRa tanpa mekanisme enkripsi yang memadai terbukti sangat rentan terhadap penyadapan lalu lintas data serta aksi manipulasi pesan melalui skenario serangan *Man-In-The-Middle* (Olazabal et al., 2022). Untuk memitigasi kerentanan tersebut, penerapan algoritma kriptografi ringan berupa *Advanced Encryption Standard* (AES) dan *Hash-based Message Authentication Code* (HMAC) terbukti efektif dalam memberikan keamanan ganda tanpa membebani memori dan komputasi perangkat keras yang memiliki keterbatasan sumber daya (Windya et al., 2021). Walaupun implementasi AES dan HMAC mampu menjaga kerahasiaan dan integritas pesan, mekanisme tersebut masih memiliki kelemahan mendasar karena menggunakan



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

konfigurasi kunci enkripsi statis (*static key*) yang tertanam secara permanen pada memori perangkat (Windya et al., 2021).

Dalam operasional sistem pemantauan jangka panjang, penggunaan satu kunci statis yang sama secara terus-menerus diidentifikasi sebagai titik kegagalan fatal (*fatal point of failure*) apabila perangkat fisik berhasil diretas atau dikuasai oleh pihak luar. Oleh karena itu, pembaruan kunci secara berkala melalui sistem manajemen kunci yang tangguh merupakan syarat mutlak untuk menjaga stabilitas kerahasiaan data *Internet of Things* (Li & Cao, 2022). Berdasarkan celah penelitian (*research gap*) dari studi-studi tersebut, penelitian ini hadir untuk menyempurnakan sistem keamanan komunikasi LoRa P2P. Solusi yang diusulkan adalah mengintegrasikan algoritma kriptografi AES-128 dan HMAC-SHA256 dengan mekanisme pembaruan kunci dinamis (*dynamic key management*) guna memastikan perlindungan data yang kebal terhadap serangan manipulasi pesan sekaligus meniadakan risiko kebocoran kunci statis di masa depan. Berikut adalah perbandingan penelitian ini dengan lima jurnal acuan (2020–2025) pada Tabel 2.1.

Tabel 2.1 Penelitian Terdahulu

Judul Penelitian	Metode/ Algoritma	Hasil Penelitian	Penjelasan
Design and Implementation of Cloud-based Single-channel LoRa IIoT Gateway Sun et al. (2020)	Custom Gateway Design	Menyatakan bahwa gateway komersial dan biaya layanan cloud LoRaWAN mahal bagi industri kecil.	Fokus pada koneksi cloud, sementara penelitian ini fokus pada keamanan di jaringan lokal.
Analisis Pengiriman Data dari LoRa Gateway ke Network Server	Analisis Protokol Gateway	Menjelaskan ketergantungan gateway pada internet dan infrastruktur cloud server untuk	Fokus pada performa gateway ke server, sementara penelitian ini fokus pada kemandirian P2P di area blank spot.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Zain et al. (2022)		transmisi data LoRaWAN.	
<i>Analysis of Payload Confidentiality for the IoT/LPWAN Technology 'LoRa'</i>	Analisis Risiko Keamanan P2P	Membuktikan urgensi proteksi P2P dari manipulasi data (MITM).	Melakukan implementasi teknis mitigasi MITM, bukan sekadar analisis risiko teoritis.
McWeeney et al. (2024)			
Penerapan Keamanan Komunikasi pada Jaringan LoRa (Long Range) Menggunakan Algoritma <i>Advanced Encryption Standard</i> (AES) dan Message Authentication Code (MAC)	AES-128 dan HMAC	Menyimpulkan kombinasi AES dan HMAC memberikan keamanan ganda yang ringan untuk perangkat ESP32.	Fokus pada keamanan kunci statis, sementara penelitian ini menerapkan <i>dynamic key management</i> .
Windya et al. (2021)			
LoRa Networking Techniques for Large-scale	Systematic Survey	Menegaskan bahwa kunci statis adalah celah keamanan fatal dan menyarankan	Jurnal ini memberikan landasan teori; penelitian ini mengimplementasikan



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

and Long-term IoT: A Down- to-top Survey Li & Cao (2022)		pembaruan kunci secara berkala.	mekanisme kunci dinamis tersebut.
--	--	------------------------------------	--------------------------------------

2.10 Kerangka Pemikiran

Kerangka pemikiran dalam penelitian ini disusun berdasarkan identifikasi masalah yang ditemukan pada implementasi infrastruktur IoT saat ini. Permasalahan utama bermula dari tingginya biaya investasi perangkat *gateway* LoRaWAN komersial, yang menyebabkan teknologi ini sulit dijangkau oleh sektor usaha kecil. Sebagai alternatif, penggunaan *gateway* mandiri (*custom gateway*) berbiaya rendah mulai marak dikembangkan. Namun, solusi alternatif ini sering kali mengabaikan aspek keamanan, di mana komunikasi data antara sensor dan *gateway* umumnya berjalan secara terbuka (*raw*) tanpa enkripsi. Kondisi ini membuat jaringan sangat rentan terhadap serangan siber, khususnya penyadapan (*sniffing*) dan serangan *Man-In-The-Middle* (MITM). Selain itu, penggunaan mekanisme pengamanan konvensional berbasis kunci statis (*static key*) dinilai memiliki risiko kerentanan yang tinggi jika perangkat fisik dicuri atau dikuasai oleh pihak yang tidak berwenang dalam operasional jangka panjang.

Berdasarkan permasalahan tersebut, penelitian ini mengusulkan pendekatan solusi berupa pengembangan *gateway* mandiri yang dibangun menggunakan perangkat keras yang efisien biaya, yaitu mikrokontroler ESP32 dan modul LoRa Ra-02 (SX1278). Fokus utama pengembangan terletak pada penerapan standar keamanan informasi (CIA *Triad*) yang komprehensif untuk melindungi data sensor. Untuk menjamin kerahasiaan (*confidentiality*), sistem menggunakan algoritma enkripsi AES-128 dengan mode *Counter* (CTR). Selanjutnya, aspek integritas dan autentikasi data dijaga menggunakan algoritma HMAC-SHA256 yang berfungsi mendeteksi adanya modifikasi data oleh pihak ketiga selama proses transmisi.

Guna mengatasi kelemahan kunci statis dan tuntutan sinkronisasi pada mode CTR, penelitian ini menerapkan mekanisme keamanan adaptif berupa *dynamic key management* dan penggunaan *Counter-based Nonce*. Mekanisme ini memungkinkan kunci enkripsi diturunkan dan diperbarui secara otomatis untuk



© Hak Cipta milik Politeknik Negeri Jakarta

setiap sesi pengiriman paket, sehingga sistem memiliki ketahanan terhadap serangan pengiriman ulang paket (*replay attack*) dan tetap aman meskipun kunci lama berhasil diretas. Melalui integrasi seluruh metode tersebut, diharapkan dapat tercipta purwarupa sistem keamanan pada *gateway* berbiaya rendah yang tangguh terhadap serangan siber tanpa membebani kinerja komputasi perangkat, sehingga dapat menjadi solusi infrastruktur IoT yang aman, andal, dan ekonomis.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB III

METODE PENELITIAN

3.1 Rancangan Penelitian

Penelitian ini mengadopsi skema penelitian eksperimental (*experimental research*) yang difokuskan pada pengembangan, implementasi, dan evaluasi protokol keamanan komunikasi data pada jaringan nirkabel LoRa dengan topologi *Point-to-Point* (P2P). Sistem komunikasi ini dirancang untuk beroperasi secara mandiri di lingkungan yang tidak memiliki akses internet (*blank spot*) tanpa bergantung pada infrastruktur *Network Server* LoRaWAN komersial. Mengingat transmisi radio LoRa P2P beroperasi secara terbuka pada lapisan fisik, seluruh perancangan mekanisme keamanan difokuskan pada modifikasi dan pemrosesan data di lapisan aplikasi (*application layer*) pada perangkat keras sebelum paket ditransmisikan ke udara.

Pendekatan yang digunakan dalam penelitian ini bersifat kuantitatif, di mana keandalan sistem dievaluasi melalui pengukuran data empiris secara langsung pada purwarupa (*prototype*) jaringan yang dibangun. Purwarupa tersebut mengintegrasikan mikrokontroler ESP32 dan modul *transceiver* LoRa Ra-02 (SX1278) yang dibekali dengan lapisan keamanan kriptografi. Mekanisme keamanan yang dirancang pada perangkat tersebut mencakup penggunaan *Advanced Encryption Standard* (AES-128) mode *Counter* (CTR) untuk kerahasiaan data, *Hash-based Message Authentication Code* (HMAC-SHA256) untuk integritas dan autentikasi pesan, serta pembaruan kunci sesi secara otomatis melalui *dynamic key management* berbasis *Key Derivation Function* (KDF).

Untuk membuktikan kelayakan dan ketangguhan rancangan keamanan tersebut, eksperimen dilakukan dengan menguji sistem pada kondisi operasional normal maupun di bawah simulasi skenario serangan siber secara langsung, seperti *sniffing*, *Man-In-The-Middle* (MITM), dan *replay attack*. Hasil dari pengujian eksperimental ini kemudian dianalisis secara kuantitatif secara komparatif (sebelum dan sesudah mitigasi diterapkan). Parameter terukur yang menjadi acuan analisis meliputi metrik kualitas jaringan (RSSI, SNR, *packet loss*, dan *latency*), metrik beban komputasi perangkat lunak (*processing time*, konsumsi RAM, dan *packet size*),



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

serta persentase keberhasilan sistem dalam mendeteksi dan menggugurkan paket data yang dimanipulasi oleh pihak penyerang.

3.2 Tahapan Penelitian

Tahapan penelitian disusun secara sistematis untuk memastikan ketercapaian tujuan penelitian, dengan rincian tahapan sebagai berikut:

1. Studi Literatur

Tahap awal ini dilakukan dengan mengkaji berbagai penelitian terdahulu dan literatur yang terkait dengan komunikasi nirkabel LoRa *Point-to-Point* (P2P), kerentanan keamanan *Internet of Things* (IoT), algoritma kriptografi AES-128 dan HMAC-SHA256, mekanisme manajemen kunci dinamis, serta teknik serangan eksploitasi seperti *Man-In-The-Middle* (MITM) dan *replay attack*.

2. Perancangan Arsitektur Sistem

Tahap ini berfokus pada kegiatan mendesain arsitektur jaringan eksperimental yang dibangun. Perancangan fisik melibatkan tiga *node* utama, yaitu perangkat pengirim (*sender*), *gateway* penerima, dan perangkat penyerang (*attacker*). Pendekatan tiga *node* ini dirancang khusus untuk mensimulasikan kondisi serangan siber secara nyata di lingkungan operasional.

3. Perancangan Mekanisme Keamanan

Tahap perancangan ini meliputi pembuatan logika pengamanan data pada lapisan aplikasi (*application layer*). Desain tersebut mencakup rancangan skema enkripsi kerahasiaan data, proses autentikasi integritas pesan, serta perhitungan matematis untuk pembaruan kunci sesi secara dinamis berbasis pada *Key Derivation Function* (KDF).

4. Implementasi Sistem

Pada tahap ini, seluruh rancangan arsitektur dan mekanisme keamanan direalisasikan secara fisik. Tahapan ini mencakup pengimplementasian algoritma keamanan ke dalam bentuk kode program (*source code*) yang ditanamkan pada mikrokontroler ESP32 serta integrasinya dengan perangkat keras modul *transceiver* LoRa Ra-02.

5. Pengujian Keamanan

Purwarupa sistem komunikasi yang telah berhasil diimplementasikan kemudian diuji secara eksperimental. Pengujian dilakukan dengan



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

menjalankan simulasi serangan secara langsung menggunakan *attacker node*, yang meliputi skenario penyadapan lalu lintas data (*sniffing*), manipulasi pesan di udara (MITM), dan pengiriman ulang paket kadaluwarsa (*replay attack*).

6. Analisis Data dan Evaluasi

Tahap terakhir adalah melakukan ekstraksi dan analisis terhadap seluruh data hasil pengujian. Evaluasi dilakukan secara kuantitatif berdasarkan parameter tingkat keberhasilan sistem dalam memitigasi serangan (*mitigation success rate*), serta analisis performa terkait latensi dan penggunaan memori pada mikrokontroler.

3.3 Objek Penelitian

Objek penelitian pada skripsi ini adalah sistem komunikasi nirkabel berbasis LoRa *Point-to-Point* (P2P) yang diimplementasikan pada perangkat mikrokontroler ESP32 dan modul transceiver LoRa Ra-02 (SX1278). Fokus utama dalam penelitian ini adalah rancang bangun serta evaluasi kinerja protokol keamanan *dynamic key management* yang menggabungkan algoritma kriptografi ringan AES-128 mode CTR dan *Hash-based Message Authentication Code* (HMAC-SHA256) guna memitigasi kerentanan lapisan fisik radio dari ancaman serangan *Man-In-The-Middle* (MITM) seperti *sniffing*, *tampering*, dan *replay attack* pada lingkungan jaringan yang beroperasi tanpa dukungan akses internet.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB IV

HASIL DAN PEMBAHASAN

4.1 Analisis Kebutuhan

Tahap analisis kebutuhan bertujuan untuk mendefinisikan seluruh spesifikasi perangkat keras (*hardware*) dan perangkat lunak (*software*) yang diperlukan agar sistem komunikasi LoRa P2P dapat beroperasi secara aman dan optimal. Kebutuhan perangkat keras dan perangkat lunak pada penelitian ini dijabarkan pada Tabel 4.1 dan Tabel 4.2.

Tabel 4.1 Kebutuhan Perangkat Keras (*Hardware*)

No	Perangkat Keras	Deskripsi / Fungsi Utama	Spesifikasi Utama
1	Mikrokontroler ESP32	Sebagai unit pemrosesan utama (<i>end-node</i> dan <i>gateway</i>) yang mengeksekusi algoritma kriptografi (AES-128 dan HMAC-SHA256) serta fungsi manajemen kunci dinamis (KDF).	Prosesor Xtensa <i>Dual-Core</i> 32-bit, <i>Hardware Cryptographic Accelerator</i> bawaan.
2	Modul LoRa Ra-02	Berfungsi sebagai modul <i>transceiver</i> untuk mentransmisikan dan menerima paket <i>ciphertext</i> secara nirkabel (<i>Point-to-Point</i>).	Chip Semtech SX1278, Frekuensi 433 MHz, Antarmuka SPI.
3	Sensor DHT22	Berfungsi sebagai komponen input pada <i>end-node</i> untuk mengakuisisi parameter	Sensor suhu dan kelembapan digital dengan tingkat akurasi tinggi.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

		lingkungan secara <i>real-time</i> .	
4	Regulator AMS1117	Modul <i>step-down</i> LDO untuk menurunkan tegangan 5V dari ESP32 menjadi 3,3V yang stabil untuk daya operasional modul LoRa.	<i>Input Voltage</i> 4.5V–15V, <i>Output Voltage</i> 3.3V DC, Arus maksimum 1A.
5	Kabel <i>Jumper</i>	Digunakan sebagai media transmisi kelistrikan <i>point-to-point</i> untuk menghubungkan pin antarkomponen (ESP32, LoRa, sensor, dan regulator) di atas papan rangkaian.	Kawat tembaga pejal, antarmuka tipe Male-to-Female dan Male-to-Male.
6	Papan Rangkaian (<i>Project Board</i> / PCB)	Digunakan sebagai papan media dasar untuk menyusun, menancapkan, dan mengintegrasikan seluruh kaki komponen purwarupa menjadi satu kesatuan sistem fisik.	Jalur konduktor tembaga, tipe matriks lubang standar.

Tabel 4.2 Kebutuhan Perangkat Lunak (*Software*)

No	Perangkat Lunak	Deskripsi / Fungsi
1	Visual Studio Code (VS Code)	Editor kode sumber utama yang digunakan sebagai lingkungan pengembangan terpadu (<i>integrated development environment</i>) untuk merancang arsitektur program sistem.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

2	PlatformIO <i>Extension</i>	Ekstensi manajemen proyek berbasis C/C++ pada VS Code yang digunakan untuk mengompilasi, mengelola <i>library dependencies</i> , dan melakukan <i>flashing</i> kode ke mikrokontroler ESP32.
3	<i>Library</i> mbedTLS	Pustaka kriptografi ringan bawaan dari ekosistem mikrokontroler ESP32 yang dieksekusi untuk menjalankan operasi enkripsi AES-128 mode CTR dan <i>hashing</i> SHA-256.
4	<i>Library</i> LoRa (Sandeep Mistry)	Pustaka pengontrol lapisan fisik radio yang digunakan untuk mengonfigurasi parameter modul Semtech SX1278 seperti frekuensi, <i>Spreading Factor</i> (SF), dan <i>Bandwidth</i> .
5	<i>Library</i> DHT <i>Sensor</i>	Pustaka pembaca parameter fisik yang digunakan untuk mengolah data sinyal analog dari sensor DHT22 menjadi nilai temperatur dan kelembapan digital.
6	<i>Library</i> Adafruit <i>Unified Sensor</i>	Pustaka dasar abstraksi perangkat keras (<i>hardware abstraction layer</i>) yang menyediakan paradigma seragam untuk mengelola sensor-sensor sekunder di bawah ekosistem Adafruit.
7	Laragon & MySQL Database	Perangkat lunak peladen lokal (<i>local server</i>) dan sistem manajemen basis data yang digunakan untuk menyimpan, menstrukturkan, dan mengelola log seluruh paket data hasil pengujian.

4.2 Perancangan Sistem

Perancangan sistem menjabarkan rancangan arsitektur dan alur kerja sistem keamanan sebelum direalisasikan ke dalam bentuk purwarupa (*prototype*).



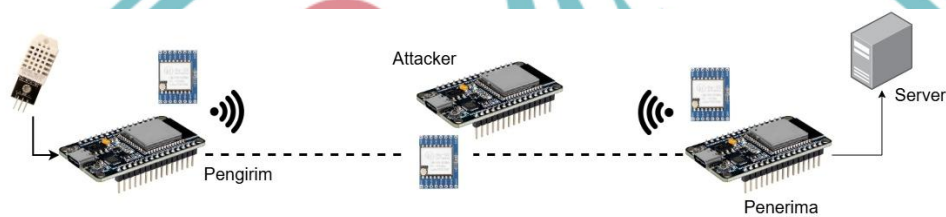
Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Perancangan ini mencakup pembuatan diagram blok (*block diagram*) dan diagram alur (*flowchart*), dan rangkaian skematik.

4.2.1 Rancangan Topologi Jaringan

Perancangan topologi jaringan digunakan untuk memvisualisasikan bagaimana hubungan komunikasi fisik dan logis antar node di udara (*Over-The-Air*). Karena penelitian ini tidak menggunakan infrastruktur internet atau *cloud server* komersial, topologi yang diterapkan adalah topologi nirkabel *Point-to-Point* (P2P). Rancangan topologi jaringan pada sistem ini ditunjukkan pada Gambar 4.1.



Gambar 4.1 Topologi Jaringan Komunikasi LoRa P2P dengan Intervensi *Attacker*

Berdasarkan topologi pada Gambar 4.1, arsitektur jaringan eksperimental ini secara fisik melibatkan tiga *node* utama yang beroperasi pada frekuensi LoRa 433 MHz. Perangkat pertama adalah *sender node* atau pengirim yang bertindak sebagai stasiun akuisisi data untuk membaca parameter lingkungan dari sensor DHT22. Pada titik ini, mikrokontroler secara mandiri mengenkripsi paket data dan memancarkannya secara nirkabel melalui gelombang radio. Perangkat kedua merupakan *receiver gateway* atau penerima yang bertindak sebagai pusat pemrosesan dan titik akhir komunikasi (*end-point*). Perangkat *gateway* ini berfungsi menerima pancaran sinyal dari udara, melakukan autentikasi keabsahan paket, mendekripsi data, dan menyimpannya secara lokal ke dalam *database*.

Selanjutnya, perangkat ketiga adalah *attacker node* atau penyerang yang direpresentasikan sebagai perangkat pihak ketiga (*rogue device*) yang menyusup ke dalam radius jangkauan sinyal LoRa. Perangkat penyerang ini bertugas secara aktif melakukan pemindaian frekuensi (*sniffing*) untuk menyadap paket yang mengudara, serta melakukan interupsi berupa manipulasi data (*tampering*) dan pengiriman ulang paket usang (*replay attack*) ke arah *gateway* guna menguji ketahanan algoritma keamanan sistem secara langsung.

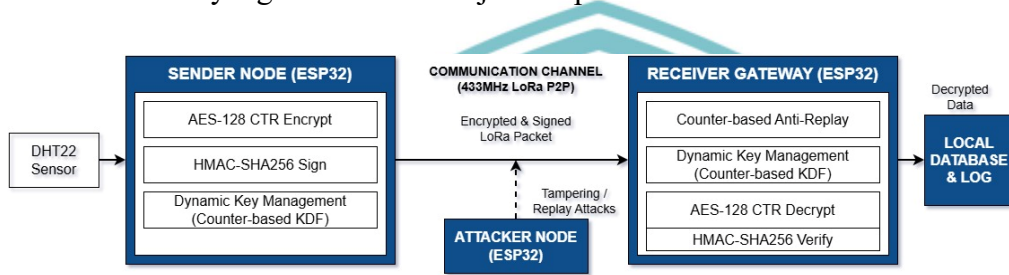


Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.2.2 Diagram Blok

Diagram blok menggambarkan rancangan topologi dan interaksi komponen pada sistem komunikasi LoRa P2P secara garis besar. Sistem ini dikelompokkan menjadi tiga bagian utama, yaitu input, process, dan output. Gambar diagram blok dari sistem yang diusulkan ditunjukkan pada Gambar 4.2.



Gambar 4.2 Diagram Blok Sistem Keamanan LoRa P2P

Berdasarkan Gambar 4.2, pada bagian input, sensor DHT22 membaca kondisi suhu dan kelembapan lingkungan. Data tersebut kemudian masuk ke bagian process di mana mikrokontroler ESP32 (sebagai *end-node*) melakukan pemrosesan tingkat lanjut pada lapisan aplikasi. Pada tahap ini, ESP32 membangkitkan *session key* baru, mengenkripsi data dengan AES-128, dan menambahkan tanda tangan digital menggunakan HMAC-SHA256. Pada bagian output, paket data yang telah tersegel aman ditransmisikan melalui kanal radio oleh modul LoRa Ra-02 ke Gateway penerima.

4.2.3 Diagram Alur (*Flowchart*)

Secara keseluruhan, alur kerja sistem komunikasi LoRa P2P pada penelitian ini direpresentasikan melalui *flowchart* yang dipecah menjadi empat proses utama untuk memudahkan pemahaman. Proses tersebut meliputi alur kerja pada perangkat pengirim, perangkat penerima, simulasi perangkat penyerang, dan alur pembaruan kunci dinamis.

4.2.3.1 Diagram Alur *End-Node* (Pengirim)

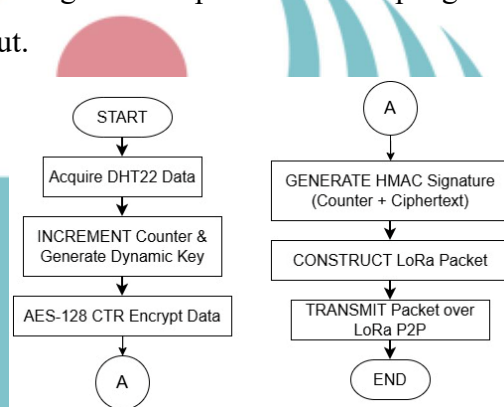
Diagram alur pada perangkat end-node pengirim memvisualisasikan tahapan pemrosesan data sejak pembacaan sensor hingga transmisi ke udara. Proses dimulai dengan inisialisasi parameter fisik modul LoRa dan pembacaan nilai suhu serta kelembapan melalui sensor DHT22. Setelah data berhasil dibaca, sistem akan memperbarui nilai *sequence counter* (ID Paket). Sistem kemudian mengecek kondisi kelipatan pengiriman, apabila ID paket



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

merupakan kelipatan 10, maka fungsi *Key Derivation Function* (KDF) akan dieksekusi untuk memperbarui *session key* secara dinamis. Setelah kunci sesi dipastikan kebaruannya, data sensor (*plaintext*) dienkripsi menggunakan algoritma AES-128 mode CTR. Selanjutnya, sistem mengkalkulasi tanda tangan digital menggunakan algoritma HMAC-SHA256 berdasarkan kombinasi *sequence counter* dan *ciphertext*. Pada tahap akhir, *payload* yang berisi *counter*, *ciphertext*, dan *hash* HMAC dirakit menjadi satu paket utuh dan ditransmisikan melalui frekuensi radio menuju *gateway*. Secara lebih terperinci, tahapan diagram alur pada *end-node* pengirim dapat dilihat pada Gambar 4.3 berikut.



Gambar 4.3 Flowchart Pengirim

4.2.3.2 Diagram Alur *Gateway* (Penerima)

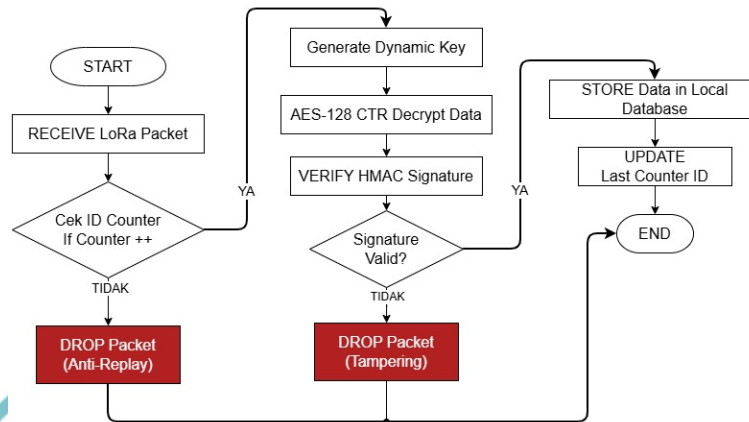
Pada sisi penerima, diagram alur dirancang dengan mengutamakan filtrasi keamanan berlapis sebelum data disimpan. Sesuai dengan desain *flowchart*, ketika *gateway* menerima paket LoRa, sistem pertama kali masuk ke dalam blok pengecekan ID *Counter*. Sistem memvalidasi apakah *sequence counter* yang masuk nilainya lebih besar dari paket sebelumnya. Jika kondisi tidak terpenuhi (TIDAK), maka paket langsung di-*drop* atau dibuang karena terindikasi sebagai *Replay Attack*. Jika kondisi terpenuhi (YA), sistem melanjutkan ke blok *Generate Dynamic Key* untuk memproduksi kunci sesi yang sinkron dengan pengirim, lalu melakukan dekripsi AES-128 CTR. Setelah data menjadi *plaintext*, sistem masuk ke blok *Verify HMAC Signature*. Pada tahap *decision* ini, apabila tanda tangan tidak valid atau berubah (TIDAK), maka paket di-*drop* karena terindikasi sebagai modifikasi data (*Tampering*). Apabila tanda tangan valid (YA), barulah data sensor dinyatakan sah, disimpan ke dalam *Local Database*, dan sistem memperbarui



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

memori *Last Counter ID* sebelum proses diakhiri. Tahapan diagram alur pada *gateway* penerima dapat dilihat pada Gambar 4.4 berikut.



Gambar 4.4 Flowchart Penerima

4.2.3.3 Diagram Alur Attacker (Penyerang)

Diagram alur pada perangkat penyerang (*attacker node*) dirancang untuk mensimulasikan ancaman keamanan nirkabel yang secara garis besar terbagi menjadi dua fase utama, yaitu fase pencarian celah parameter fisik (*Reconnaissance*) dan fase eksekusi serangan aktif (*Attacking*). Masing-masing fase memiliki alur kerjanya tersendiri sebagai berikut:

- a. Fase Pencarian Celah (*Reconnaissance*) perangkat penyerang berfokus pada pengintaian di udara untuk menemukan celah parameter jaringan fisik. Adapun tahapan kerjanya adalah sebagai berikut:
 1. Inisialisasi & Pengaturan Parameter: Sistem menginisialisasi *hardware* mikrokontroler ESP32 dan modul LoRa Ra-02, kemudian mengatur *Scanning Pool Parameters* yang meliputi kombinasi Frekuensi, *Spreading Factor* (SF), *Bandwidth* (BW), dan *Coding Rate* (CR).
 2. Proses *Sniffing* & Penguncian Jaringan: Perangkat mulai mendengarkan udara pada saluran yang ditentukan (*Sniffing Channel*). Apabila tidak ada paket terdeteksi, perangkat akan mengganti parameter ke *pool* berikutnya. Namun, jika ada transmisi yang terdeteksi, perangkat akan menangkap *raw byte packet* dan mengunci parameter fisik jaringan target tersebut



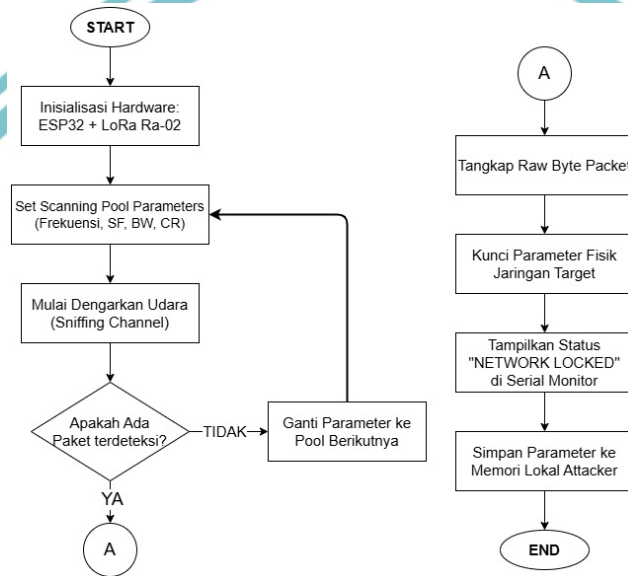
Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

(ditandai dengan munculnya status "NETWORK LOCKED" di *Serial Monitor*).

3. Penyimpanan Parameter Target: Pada akhir fase ini, penyerang menyimpan parameter jaringan yang berhasil diretas tersebut ke dalam memori lokal untuk digunakan pada fase serangan.

Secara lebih jelas, tahapan fase *Reconnaissance* ini dapat dilihat pada Gambar 4.5 berikut.



Gambar 4.5 Flowchart Penyerang

- b. Fase Eksekusi Serangan Aktif (*Attacking*) setelah parameter jaringan berhasil dikunci pada fase pertama, penyerang kemudian memasuki fase kedua yaitu eksekusi serangan. Adapun tahapan simulasi eksploitasi ini adalah sebagai berikut:

1. Proses *Sniffing* Lanjutan ketika penyerang mulai mendengarkan lalu lintas udara secara *real-time*. Apabila sebuah paket sah terbaca, penyerang akan menangkap dan menyalin *raw byte* dari paket valid tersebut untuk dijadikan bahan serangan.
2. Skenario *Replay Attack* apabila penyerang memilih jalur serangan ini, penyerang akan menyimpan paket valid tanpa melakukan perubahan *payload*, memberikan jeda waktu tertentu (*Delay Trigger*), dan mengirimkan ulang paket usang tersebut ke *gateway*. Skenario ini akan menghasilkan status kegagalan, di



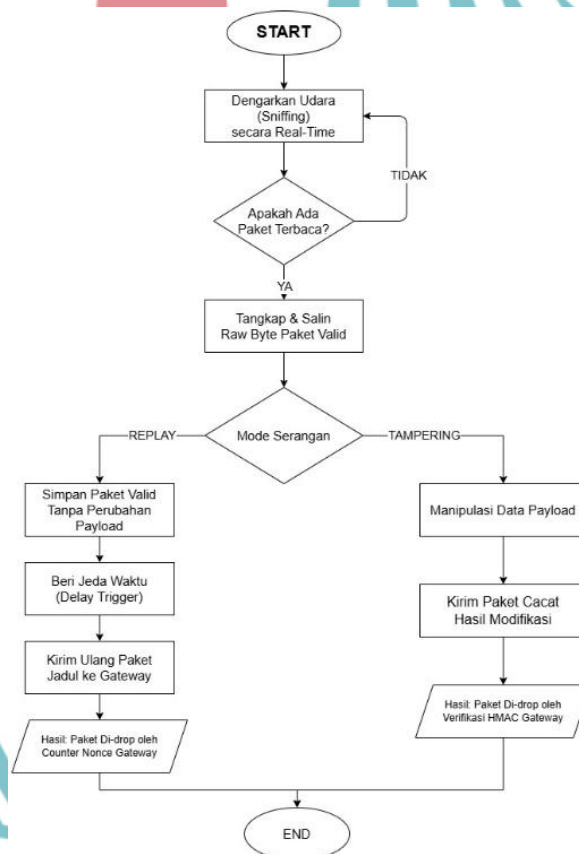
Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

mana paket di-drop (ditolak) secara otomatis oleh lapisan validasi *Counter Nonce* pada gateway.

3. Skenario *Tampering* apabila penyerang memilih jalur serangan ini, penyerang akan memanipulasi *data payload* (mengubah susunan *byte ciphertext*) dan mengirimkan paket modifikasi tersebut kembali ke udara. Serangan ini juga berujung pada kegagalan, di mana paket di-drop oleh lapisan verifikasi tanda tangan HMAC pada gateway.

Rincian langkah kerja dari fase eksekusi serangan pada perangkat penyerang ini dapat dilihat pada Gambar 4.6 berikut.



Gambar 4.6 Flowchart *Attacker* Melakukan Serangan

4.2.3.4 Diagram Alur Key Derivation Function (KDF)

Diagram alur KDF memvisualisasikan langkah-langkah prosedural dalam menghasilkan kunci sesi dinamis (*Dynamic Key Management*) yang dieksekusi secara lokal pada perangkat. Merujuk pada persamaan matematis Key Derivation Function (Persamaan 2.1) yang telah dijabarkan pada Bab 2,



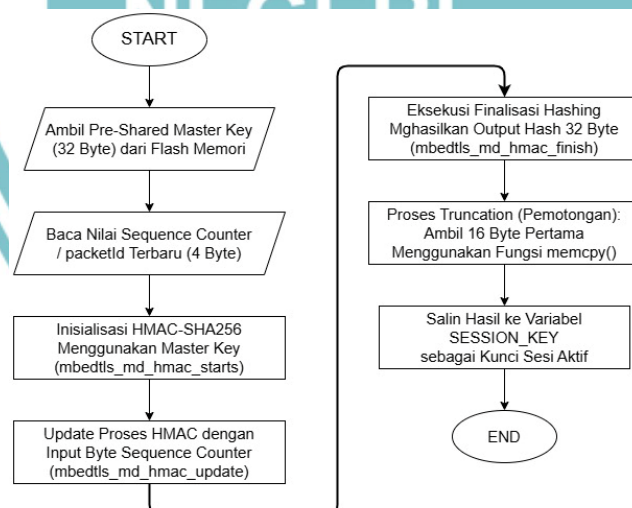
Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

proses ini diawali dengan tahapan pengambilan dua parameter krusial sebagai *Input Keying Material*. Sistem pertama-tama mengambil *Pre-Shared Master Key* (K_{master}) berukuran 32-Byte yang tersimpan aman di dalam *flash memory*, kemudian dilanjutkan dengan membaca nilai *Sequence Counter* atau *packetId* terbaru yang telah diekstrak menjadi susunan *array* 4-Byte sebagai representasi dari parameter dinamis (*packetId* 0...15).

Sesuai dengan rumusan persamaan tersebut, kedua bahan baku ini kemudian dicampur melalui proses pencampuran entropi (*entropy mixing*) dan dimasukkan ke dalam ruang komputasi *hashing*. Sistem mengeksekusi algoritma HMAC-SHA256 untuk memproduksi keluaran hash (*digest*) acak yang memiliki panjang penuh 32-Byte.

Oleh karena algoritma enkripsi AES-128 secara spesifik hanya membutuhkan format kunci berukuran 128-bit (16-Byte), sistem kemudian diinstruksikan untuk melakukan proses pemotongan (*truncation*). Pada blok ini, sistem secara presisi mengekstrak 16-Byte pertama dari output HMAC menggunakan fungsi memori *memcpy()*. Susunan 16-Byte yang berhasil dipotong inilah yang pada tahap akhir disalin dan dideklarasikan ke dalam variabel *SESSION_KEY* untuk digunakan sebagai kunci sesi aktif ($K_{session}$) pada operasi enkripsi atau dekripsi selanjutnya. Adapun urutan langkah kerja KDF ini dapat dilihat secara terperinci pada Gambar 4.7 berikut.



Gambar 4.7 Flowchart *Key Derivation Function*

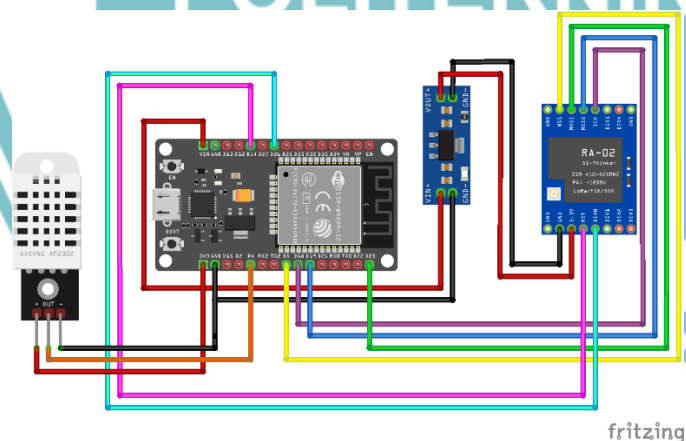


Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.2.4 Desain Skematik dan Konfigurasi Pin

Rancangan skematik memvisualisasikan arsitektur perangkat keras penyusun sistem komunikasi LoRa P2P pada penelitian ini. Mengingat adanya perbedaan peran fungsional dalam alur komunikasi, perancangan perangkat keras direpresentasikan ke dalam dua skema kelistrikan utama. Skematik yang pertama adalah arsitektur untuk perangkat pengirim (*end-node*). Perangkat ini direpresentasikan sebagai titik awal akuisisi data sekaligus pusat pengamanan kriptografi. Pada rangkaian ini, mikrokontroler ESP32 bertindak sebagai otak pemroses yang bertugas membaca data lingkungan dari sensor DHT22. Data tersebut kemudian diproses secara internal untuk dienkripsi menggunakan algoritma AES-128 dan diberi tanda tangan digital HMAC-SHA256. Setelah paket siap, ESP32 menyerahkan data tersebut kepada modul *transceiver* LoRa Ra-02 melalui jalur komunikasi *Serial Peripheral Interface* (SPI). Modul LoRa inilah yang kemudian mengambil peran pada lapisan fisik untuk memodulasi data digital menjadi sinyal radio dan memancarkannya ke udara secara nirkabel. Untuk menunjang stabilitas pancaran sinyal yang rawan menarik lonjakan arus listrik, rangkaian pengirim ini juga dilengkapi dengan regulator daya AMS1117 guna memastikan suplai listrik ke modul LoRa tetap stabil di angka 3,3V. Konfigurasi skematik perangkat pengirim ini dapat dilihat pada Gambar 4.8.



Gambar 4.8 Rancangan Skematik Pengirim

Skematik selanjutnya adalah arsitektur untuk perangkat penerima (*gateway*). Pada perangkat ini (*gateway*), modul LoRa bertugas menangkap sinyal radio yang dikirimkan oleh *end-node*, lalu menyerahkan paket tersebut ke ESP32 untuk

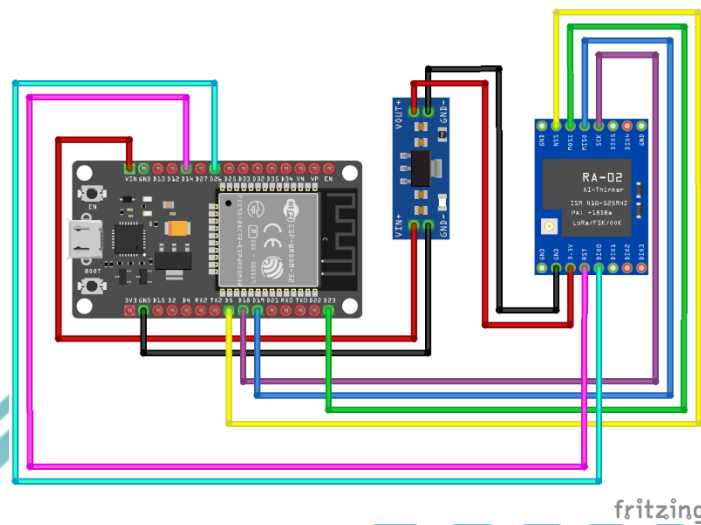


© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

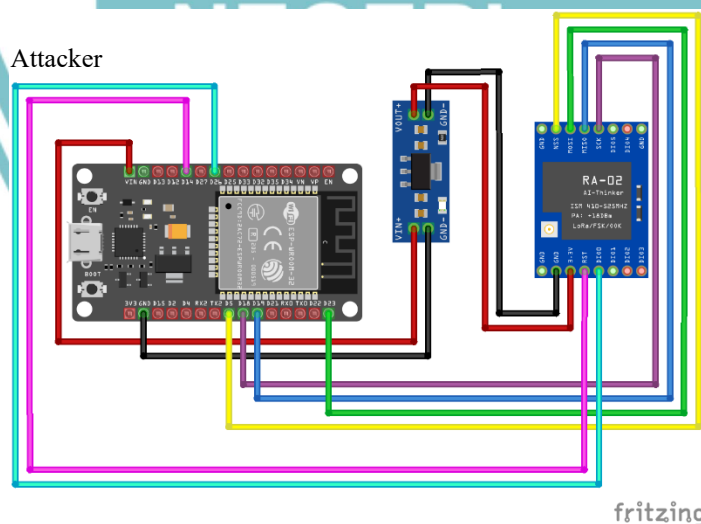
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

dieksekusi tahapan validasi keasliannya serta proses dekripsi. Skematik *gateway* dapat dilihat pada Gambar 4.9.



Gambar 4.9 Rancangan Skematik *Gateway* (Penerima)

Yang terakhir adalah perangkat penyerang, rancangan perangkat keras yang sama dimanfaatkan untuk mengeksploitasi jaringan. Modul LoRa pada penyerang dialihfungsikan sebagai *sniffer* untuk menyadap paket yang melintas, yang kemudian data sadapan tersebut digunakan oleh ESP32 untuk mengeksekusi simulasi serangan manipulasi data (*Tampering*) maupun pengiriman ulang (*Replay Attack*). Konfigurasi skematik untuk kedua perangkat ini dapat dilihat pada Gambar 4.10.



Gambar 4.10 Rancangan Skematik *Attacker* (Penyerang)



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Untuk mempermudah proses perakitan fisik, konfigurasi pemetaan pin (*pinout*) antara mikrokontroler ESP32, modul LoRa Ra-02, sensor DHT22, dan regulator AMS1117 dirangkum secara rinci pada Tabel 4.3.

Tabel 4.3 Konfigurasi *Pinout* Sistem

Komponen Asal	Pin Asal	Komponen Tujuan	Pin Tujuan
ESP32	D18	LoRa Ra-02	SCK
ESP32	D19	LoRa Ra-02	MISO
ESP32	D23	LoRa Ra-02	MOSI
ESP32	D5	LoRa Ra-02	NSS/CS
ESP32	D14	LoRa Ra-02	RST
ESP32	D26	LoRa Ra-02	DIO0
AMS1117	Vout (+)	LoRa Ra-02	3.3V
AMS1117	Vout (-)	LoRa Ra-02	GND
ESP32	Vin 5V	AMS1117	Vin (+)
ESP32	GND	AMS1117	Vin (-)
ESP32	3.3V	DHT 22	VCC
ESP32	GND	DHT 22	GND
ESP32	Data / Out	DHT 22	D4

4.3 Implementasi Sistem

Implementasi sistem menjelaskan tahapan-tahapan dalam pembuatan sistem nyata berdasarkan perancangan arsitektur yang telah dibuat. Tahapan ini terdiri dari implementasi perangkat keras (*hardware*) untuk merealisasikan purwarupa fisik, dan implementasi perangkat lunak (*software*) untuk memprogram logika komunikasi dan keamanan kriptografi pada perangkat.

4.3.1 Implementasi Perangkat Keras

Implementasi perangkat keras mencakup perakitan komponen mikrokontroler ESP32 dengan modul komunikasi, sensor, dan catu daya pada *Project Board*.

4.3.1.1 Perangkaan Modul LoRa Ra-02

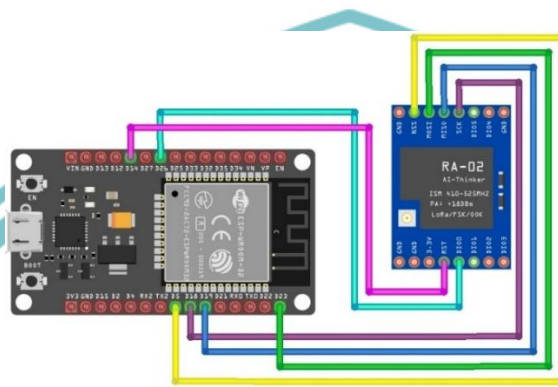
Modul LoRa Ra-02 digunakan sebagai antarmuka komunikasi nirkabel utama pada *end-node*, *secure gateway*, maupun *attacker node*. Modul ini



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

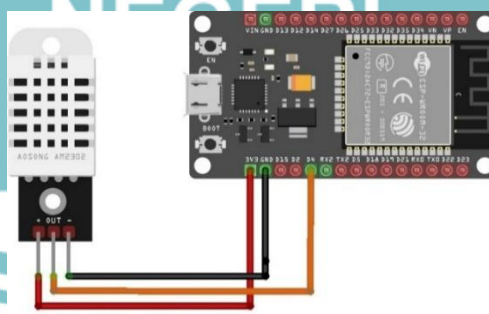
dihubungkan dengan ESP32 menggunakan jalur komunikasi Serial Peripheral Interface (SPI). Pin MOSI, MISO, SCK, dan NSS pada LoRa secara berurutan dihubungkan ke pin GPIO 23, 19, 18, dan 5 pada ESP32. Hasil implementasi rangkaian ESP32 dan modul LoRa ditunjukkan pada Gambar 4.11.



Gambar 4.11 Rangkaian ESP32 dan Modul LoRa Ra-02

4.3.1.2 Perangkaian Sensor DHT22

Sensor DHT22 diimplementasikan khusus pada perangkat pengirim (*end-node*) sebagai komponen input untuk mengakuisisi data suhu dan kelembapan secara real-time. Pin data (*OUT*) pada sensor DHT22 dihubungkan ke pin GPIO 4 pada mikrokontroler ESP32, sedangkan pin VCC dan GND dihubungkan ke jalur catu daya. Hasil implementasi rangkaian sensor DHT22 ditunjukkan pada Gambar 4.12.



Gambar 4.12 Rangkaian ESP32 dan Sensor DHT22

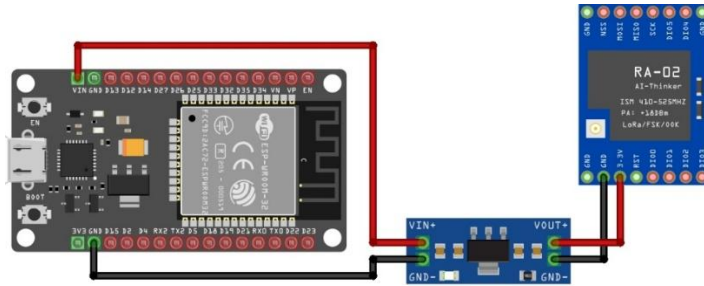
4.3.1.3 Perangkaian Regulator AMS1117

Modul step-down regulator AMS1117 digunakan untuk menurunkan tegangan 5V dari pin Vin ESP32 menjadi tegangan 3,3V yang stabil. Daya keluaran 3,3V ini secara khusus dialokasikan untuk menyuplai pin daya pada modul LoRa Ra-02. Hal ini bertujuan untuk mencegah terjadinya penurunan

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

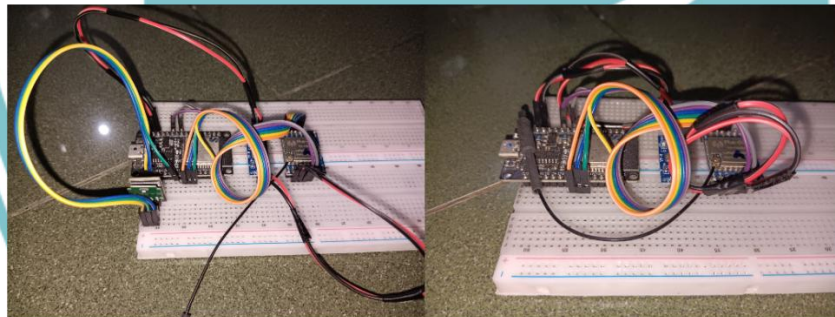
daya (*voltage drop*) atau restart paksa pada ESP32 saat proses transmisi radio sedang berlangsung. Rangkaian implementasi regulator ditunjukkan pada Gambar 4.13.



Gambar 4.13 Rangkaian Regulator AMS1117

4.3.1.4 Perakitan Keseluruhan pada *Project Board*

Setelah seluruh komponen dipastikan berfungsi dengan baik secara individual, komponen-komponen tersebut dirakit secara permanen di atas purwarupa papan sirkuit (*Project Board*). Perakitan ini dilakukan untuk menghasilkan wujud fisik dari perangkat *end-node*, *gateway*, dan *attacker node*. Hasil perakitan purwarupa perangkat keras secara keseluruhan ditunjukkan pada Gambar 4.14.



Gambar 4.14 Perakitan Keseluruhan Perangkat Keras

4.3.2 Implementasi Perangkat Lunak

Implementasi perangkat lunak terdiri dari penulisan kode program (*source code*) ke dalam ESP32 menggunakan environment Visual Studio Code (VS Code) yang diintegrasikan dengan ekstensi PlatformIO. PlatformIO dipilih karena kemampuannya dalam manajemen library dan proses kompilasi kode berbasis bahasa C++ yang lebih efisien dibandingkan editor standar. Implementasi ini difokuskan pada pengaturan lapisan radio LoRa dan penanaman algoritma keamanan kriptografi ringan (*lightweight cryptography*).



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.3.2.1 Konfigurasi Komunikasi LoRa

Pengaturan lapisan fisik radio dilakukan menggunakan pustaka LoRa.h. Ketiga entitas perangkat dikonfigurasi untuk beroperasi pada frekuensi 433 MHz dengan parameter *Spreading Factor* (SF) dan *bandwidth* (BW) yang seragam. Hal ini dilakukan agar seluruh perangkat berada pada kanal medium komunikasi yang sama. Potongan implementasi kode inisialisasi modul LoRa ditunjukkan pada Gambar 4.15.

```

87  if (!LoRa.begin(433E6)) {
88      Serial.println("LoRa init FAILED!");
89      while (1);
90  }
91
92  // Konfigurasi LoRa (Spreading Factor, Bandwidth, Coding Rate, Sync Word, Tx Power)
93  LoRa.setSpreadingFactor(7);
94  LoRa.setSignalBandwidth(125E3);
95  LoRa.setCodingRate4(5);
96  LoRa.setSyncWord(0x12); // Default
97  LoRa.setTxPower(17);
  
```

Gambar 4.15 Implementasi Fungsi Inisialisasi Modul LoRa

4.3.2.2 Implementasi Enkripsi AES-128 Mode CTR

Kerahasiaan data sensor dijamin melalui enkripsi AES-128 CTR. Sebelum data dienkripsi, sistem menyusun vektor inisialisasi (*Initialization Vector* / IV) berukuran 16-byte. *Initialization Vector* ini dirakit secara dinamis dengan menggabungkan 12-byte *BASE_NONCE* statis dan 4-byte nilai *packetId* saat itu yang disisipkan pada indeks ke-12 hingga ke-15 dari *array* IV. Selanjutnya, mikrokontroler memanggil fungsi *mbedtls_aes_setkey_enc* dengan memasukkan *SESSION_KEY* dinamis yang telah dibuat oleh KDF sebelumnya. Teks asli (*plaintext*) yang berisi data suhu, kelembapan, dan status peringatan (seperti "PANAS" atau "NORMAL") kemudian dienkripsi menjadi teks acak menggunakan fungsi *mbedtls_aes_crypt_ctr*. Implementasi penyusunan *Initialization Vector* dan enkripsi AES-128 ini ditunjukkan pada Gambar 4.16.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

```

143 // 2. Siapkan IV / Nonce (12 Byte Base + 4 Byte Packet ID)
144 uint8_t iv[16];
145 memcpy(iv, BASE_NONCE, 12);
146 iv[12] = (packetId >> 24) & 0xFF;
147 iv[13] = (packetId >> 16) & 0xFF;
148 iv[14] = (packetId >> 8) & 0xFF;
149 iv[15] = packetId & 0xFF;
150
151 // Array counter khusus untuk disematkan di udara (4 Byte)
152 uint8_t counter_bytes[4] = {iv[12], iv[13], iv[14], iv[15]};
153
154 // 3. Proses Enkripsi AES-128 CTR (Menggunakan SESSION_KEY Dinamis)
155 mbedtls_aes_context aes;
156 mbedtls_aes_init(&aes);
157 mbedtls_aes_setkey_enc(&aes, SESSION_KEY, 128); // <-- Memakai kunci dinamis
158
159 uint8_t ciphertext[pt_len];
160 size_t nc_off = 0;
161 uint8_t stream_block[16] = {0};
162
163 mbedtls_aes_crypt_ctr(&aes, pt_len, &nc_off, iv, stream_block, plaintext, ciphertext);
164 mbedtls_aes_free(&aes);

```

Gambar 4.16 Implementasi Algoritma Enkripsi AES-128 CTR

4.3.2.3 Implementasi HMAC-SHA256

Sistem menerapkan lapisan keamanan tambahan berupa *Message Authentication Code* (MAC) untuk menjamin aspek integritas pesan (*integrity*) serta mendeteksi secara mutlak adanya manipulasi data di udara (*tampering attack*). Sebelum algoritma kriptografi ini dijalankan, sistem pada tahap awal mendeklarasikan variabel kunci yang bersifat statis. Bersamaan dengan pendefinisian kunci utama, sistem juga menyiapkan variabel konstanta bernama *HMAC_KEY* berukuran 32-byte. Kunci statis ini ditanamkan secara permanen di dalam memori mikrokontroler dan bertindak sebagai sandi rahasia eksklusif untuk membentuk tanda tangan digital. Kunci ini memastikan bahwa hanya perangkat yang memiliki kecocokan sandi yang mampu menghasilkan kode *hash* yang valid di sisi penerima. Potongan kode deklarasi pendefinisian kunci keamanan ini ditunjukkan pada Gambar 4.17.

```

29 // =====
30 // SECURITY KEYS & CONFIG
31 // =====
32 // Master Key & HMAC Key (Statis, ditanam di memori, tidak pernah dikirim ke udara)
33 const char* MASTER_KEY = "MasterKeySkripsiRizki1234567890!"; // 32 Bytes
34 const char* HMAC_KEY = "IniKunciRahasiaHMACSkripsiRizki!"; // 32 Bytes

```

Gambar 4.17 Implementasi Deklarasi Variabel Kunci Keamanan

Berdasarkan rancangan diagram alur (*flowchart*), proses perlindungan dieksekusi menggunakan algoritma HMAC-SHA256 yang difasilitasi secara bawaan oleh pustaka kriptografi *mbedtls*. Mikrokontroler memulai operasi autentikasi dengan mendeklarasikan sebuah variabel *array* bernama



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

hmac_result berukuran 32-byte sebagai wadah penampung hasil akhir tanda tangan digital. Sistem kemudian menginisialisasi ruang komputasi *hashing* dengan mengatur parameter tipe operasi secara spesifik pada algoritma SHA-256. Setelah mesin kalkulasi tersebut aktif, tahapan awal HMAC dimulai dengan menyuntikkan variabel *HMAC_KEY* ke dalam fungsi eksekusi sebagai kata sandi dasar perhitungan. Potongan kode inisialisasi ruang komputasi HMAC ini ditunjukkan pada Gambar 4.18.

```
165 // 4. Proses Autentikasi HMAC-SHA256
166 uint8_t hmac_result[32];
167 mbedtls_md_context_t ctx;
168 mbedtls_md_init(&ctx);
169 mbedtls_md_setup(&ctx, mbedtls_md_info_from_type(MBEDTLS_MD_SHA256), 1);
170 mbedtls_md_hmac_starts(&ctx, (const unsigned char*)HMAC_KEY, 32);
```

Gambar 4.18 Implementasi Inisialisasi Pustaka dan Kunci Dasar HMAC-SHA256

Setelah fondasi perhitungan dan kata sandi dasar dipersiapkan, mikrokontroler melangkah pada tahapan krusial berikutnya yaitu pencampuran data ke dalam fungsi *hashing*. Sesuai dengan skenario mitigasi keamanan, tanda tangan digital yang dibentuk diwajibkan untuk mengikat nilai urutan waktu pesan agar sistem kebal terhadap teknik pemalsuan tingkat lanjut. Oleh karena itu, sistem memanggil fungsi pembaruan kalkulasi untuk menyuntikkan *array* empat byte dari variabel pengurut pesan (*counter_bytes*), dan dilanjutkan dengan menyisipkan struktur susunan data teks tersandi (*ciphertext*) beserta ukuran panjang muatannya (*pt_len*). Seluruh elemen vital tersebut kemudian diolah secara matematis melalui instruksi penyelesaian akhir. Operasi ini secara instan memproduksi sekumpulan karakter *hash* acak yang langsung diekstrak ke dalam memori *hmac_result* untuk nantinya disematkan bersama paket transmisi LoRa. Setelah rangkaian pembentukan tanda tangan digital ini dinyatakan selesai, mikrokontroler menjalankan perintah pembersihan memori konteks *mbedtls* guna mencegah terjadinya risiko kebocoran memori RAM. Tahapan kalkulasi data dan penyelesaian HMAC ini divisualisasikan pada Gambar 4.19.

```
172 // HMAC mengkalkulasi Counter + Ciphertext (Anti-Tampering)
173 mbedtls_md_hmac_update(&ctx, counter_bytes, 4);
174 mbedtls_md_hmac_update(&ctx, ciphertext, pt_len);
175 mbedtls_md_hmac_finish(&ctx, hmac_result);
176 mbedtls_md_free(&ctx);
```

Gambar 4.19 Implementasi Kalkulasi Integritas Pesan dan Eksekusi Hash Akhir



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.3.2.4 Implementasi Dynamic Key Management (KDF)

Sistem dirancang secara khusus untuk tidak menggunakan kunci enkripsi statis guna memitigasi serangan pengiriman ulang paket dari pihak peretas. Pembaruan kunci perlindungan dieksekusi secara otomatis dan mandiri oleh mikrokontroler menggunakan skema *Key Derivation Function* (KDF). Pada tahap implementasi awal, sistem mendeklarasikan variabel-variabel kunci yang menjadi fondasi keamanan struktural. Variabel konstanta *MASTER_KEY* berukuran 32-byte bersifat statis dan ditanam permanen di dalam mikrokontroler tanpa pernah ditransmisikan melalui jaringan udara terbuka. Kunci ini merupakan bahan baku utama yang sangat rahasia. Terdapat pula variabel *HMAC_KEY* berukuran 32-byte yang difungsikan murni sebagai sandi rahasia pembentuk tanda tangan digital guna mendeteksi kecurangan manipulasi data. Selanjutnya, sistem menyiapkan sebuah variabel array bernilai kosong bernama *SESSION_KEY* dengan ukuran 16-byte atau 128-bit. Variabel ini difungsikan sebagai wadah penampung kunci dinamis yang secara spesifik akan dipanggil pada saat eksekusi operasi AES-128. Potongan kode deklarasi pendefinisian kunci keamanan ini ditunjukkan pada Gambar 4.20.

```

29 // =====
30 // SECURITY KEYS & CONFIG
31 // =====
32 // Master Key & HMAC Key (Statis, ditanam di memori, tidak pernah dikirim ke udara)
33 const char* MASTER_KEY = "MasterKeySkripsiRizki1234567890!"; // 32 Bytes
34 const char* HMAC_KEY = "IniKunciRahasiaMACSkripsiRizki!"; // 32 Bytes
35
36 // Variabel Kunci Sesi yang akan terus berubah (Dinamis)
37 uint8_t SESSION_KEY[16]; // 16 Byte (128 bit) untuk AES-128

```

Gambar 4.20 Implementasi Deklarasi Variabel Kunci Keamanan

Berdasarkan rancangan diagram alur (*flowchart*), tahapan pembaruan kunci diawali dengan persiapan ruang komputasi *hashing*. Mikrokontroler mengaktifkan pustaka kriptografi *mbedtls* untuk menyiapkan kalkulasi menggunakan algoritma SHA-256. Pada tahap inisialisasi awal ini, variabel *MASTER_KEY* disuntikkan ke dalam ruang operasi sebagai sandi dasar untuk memulai perhitungan. Guna memastikan kunci sesi yang diproduksi selalu memiliki struktur acak yang baru pada setiap siklus, sistem menerapkan teknik pencampuran entropi (*entropy mixing*) dengan memanfaatkan nilai urutan transmisi paket (*Sequence Counter*) sebagai material yang dinamis.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Nilai variabel *packetId* yang bertipe data integer 32-bit wajib dipecah secara sistematis menjadi susunan *array* 4-byte bernama *counter_bytes* melalui instuksi penggeseran bit. Pemecahan dan penyisipan urutan paket ini amat vital peranannya untuk memberikan kepastian bahwa nilai akhir kalkulasi kunci akan berubah drastis secara matematis pada setiap sesi komunikasi data nirkabel.

Setelah parameter dinamis tersebut sukses disiapkan, mikrokontroler menginstruksikan fungsi komputasi untuk segera mengaduk materi kunci utama dengan *array* urutan paket. Keluaran dari instruksi pembaruan ini menghasilkan sebuah susunan karakter hash acak sepanjang 32-byte yang ditampung sementara pada variabel *hmac_result*. Oleh karena algoritma enkripsi AES-128 secara baku hanya diformat untuk beroperasi dengan rentang ukuran kunci 128-bit, keseluruhan proses komputasi ini harus diakhiri dengan eksekusi teknik pemotongan (*truncation*). Fungsi manipulasi memori diinstruksikan untuk mengekstrak serta menyalin tepat 16-byte pertama dari output hash secara akurat demi menimpa data usang yang ada di dalam variabel penampung *SESSION_KEY*. Keseluruhan alur kerja fungsi perombakan dan turunan kunci secara dinamis ini divisualisasikan pada Gambar 4.21.

```

44 // =====
45 // FUNGSI KDF (Dynamic Key Management)
46 // =====
47 void updateSessionKey() {
48     uint8_t hmac_result[32];
49     mbedtls_md_context_t ctx;
50     mbedtls_md_init(&ctx);
51     mbedtls_md_setup(&ctx, mbedtls_md_info_from_type(MBEDTLS_MD_SHA256), 1);
52
53     // HMAC-SHA256 menggunakan Master Key
54     mbedtls_md_hmac_starts(&ctx, (const unsigned char*)MASTER_KEY, 32);
55
56     // Masukkan packetId sebagai parameter dinamis (material KDF)
57     uint8_t counter_bytes[4];
58     counter_bytes[0] = (packetId >> 24) & 0xFF;
59     counter_bytes[1] = (packetId >> 16) & 0xFF;
60     counter_bytes[2] = (packetId >> 8) & 0xFF;
61     counter_bytes[3] = packetId & 0xFF;
62
63     mbedtls_md_hmac_update(&ctx, counter_bytes, 4);
64     mbedtls_md_hmac_finish(&ctx, hmac_result);
65     mbedtls_md_free(&ctx);
66
67     // Ekstrak 16 Byte pertama dari hasil Hash untuk jadi Session Key AES-128
68     memcpy(SESSION_KEY, hmac_result, 16);
69
70     Serial.print("[~] KDF TRIGGERED! Session Key baru dibuat pada Packet ID: ");
71     Serial.println(packetId);
72 }

```

Gambar 4.21 Implementasi Kalkulasi KDF dan Pemotongan Kunci Sesi



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Tahap final dari serangkaian siklus manajemen dinamis ini adalah pembuktian langsung penerapan hasil ekstraksi KDF pada lapisan aplikasi enkripsi di perangkat. Pada detik-detik sebelum perangkat merakit dan melempar payload ke frekuensi radio, mikrokontroler mengarahkan antarmuka parameter AES-128 untuk mengimpor muatan nilai keamanan terbaru dari variabel `SESSION_KEY`. Konfigurasi ini secara mutlak mematikan efektivitas kunci statis pada skenario komunikasi nirkabel terbuka, sekaligus merespons ancaman peretas. Penggunaan parameter dinamis berantai tersebut menjamin bahwa seluruh paket teks tersandi yang tertangkap di udara bebas akan dilapisi dengan kompleksitas kriptografi baru yang tidak mungkin dipetakan menggunakan pola penyadapan masa lampau. Tataan implementasi penyematan kunci sesi dinamis ke dalam ruang lingkup kalkulasi AES ini dapat dilihat pada Gambar 4.22.

```
153 // 3. Proses Enkripsi AES-128 CTR (Menggunakan SESSION_KEY Dinamis)
154 mbedtls_aes_context aes;
155 mbedtls_aes_init(&aes);
156 mbedtls_aes_setkey_enc(&aes, SESSION_KEY, 128); // Memakai kunci dinamis
```

Gambar 4.22 Implementasi Penerapan Kunci Sesi Dinamis pada Enkripsi AES-128

4.4 Pengujian Sistem

Pengujian sistem menjelaskan tentang cara dan tahapan bagaimana setiap komponen pada sistem diuji sehingga dapat dianalisis hasil dan kinerjanya. Bagian ini terdiri dari deskripsi pengujian, prosedur pengujian, data hasil pengujian, serta analisis data dan evaluasi hasil pengujian.

4.4.1 Deskripsi Pengujian

Pengujian dalam penelitian ini difokuskan pada dua parameter utama. Pertama, pengujian kinerja transmisi dan komputasi untuk mengetahui tingkat overhead yang dihasilkan dari penambahan algoritma enkripsi AES-128 dan HMAC-SHA256 terhadap waktu pemrosesan, ukuran payload, penggunaan memori (RAM), dan latensi jaringan. Kedua, pengujian keamanan sistem untuk membuktikan efektivitas mekanisme *dynamic key management* dalam mendeteksi dan memitigasi serangan siber di udara (*Over-The-Air*), yang meliputi skenario serangan *sniffing*, *tampering*, dan *replay attack*.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

4.4.2 Prosedur Pengujian

Prosedur pengujian menjelaskan langkah-langkah eksperimental yang dilakukan untuk memastikan sistem pertahanan bekerja sesuai rancangan. Pengujian ini melibatkan tiga entitas perangkat keras: *end-node* sebagai pengirim data sensor, *secure gateway* sebagai penerima sah, dan *attacker node* yang dikendalikan sebagai pihak penyerang. Pengambilan sampel dilakukan dengan merekam log transmisi pada terminal *Serial Monitor*.

4.4.2.1 Prosedur Pengujian Fase *Scanning* (Pengintaian)

Pengujian ini bertujuan untuk membuktikan bahwa penyerang dapat memindai lalu lintas udara dan mengunci parameter komunikasi jaringan LoRa target sebelum melancarkan serangan. Prosedur pengujian yang dilakukan adalah sebagai berikut:

1. Menyiapkan dan menyalakan perangkat *attacker node* yang telah diprogram dengan kode *scanning*.
2. Membiarkan perangkat memindai pool parameter frekuensi, *Spreading Factor* (SF), *Bandwidth* (BW), dan *Coding Rate* (CR) secara terus-menerus di udara.
3. Mengobservasi log pada *Serial Monitor* penyerang hingga perangkat berhasil menemukan paket transmisi sah dan menampilkan status "*NETWORK LOCKED*".

4.4.2.2 Prosedur Pengujian Sistem *Baseline*

Pengujian ini mengevaluasi tingkat kerentanan sistem bawaan LoRa ketika mentransmisikan data tanpa mekanisme perlindungan apa pun. Prosedur pengujian yang dilakukan adalah sebagai berikut:

1. Menyalakan *end-node* dan *gateway* menggunakan program *baseline* (tanpa enkripsi).
2. Menggunakan *attacker node* untuk melakukan penyadapan (*sniffing*) secara pasif guna melihat data plaintext secara telanjang di udara.
3. Melakukan injeksi serangan aktif berupa modifikasi nilai *payload* (*tampering*).
4. Melancarkan serangan pengiriman ulang paket usang (*replay attack*) menggunakan data yang telah disadap sebelumnya.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

5. Memverifikasi *log* pada *Serial Monitor gateway* dan *database MySQL* untuk memastikan bahwa seluruh paket serangan tersebut berhasil masuk tanpa penolakan. Merekam *log gateway* untuk membuktikan bahwa sistem rentan karena tetap mendekripsi paket modifikasi tersebut menjadi data sampah (*garbage data*) lalu menyimpannya ke basis data.

4.4.2.3 Prosedur Pengujian Keamanan Hanya AES-128

Pengujian ini mengevaluasi sistem yang hanya dilengkapi dengan algoritma enkripsi AES-128 untuk kerahasiaan data, tanpa didampingi mekanisme validasi integritas (HMAC) maupun perlindungan sequence counter. Rincian prosedur serangannya adalah:

1. Pada prosedur serangan *sniffing*, penyerang menyadap lalu lintas udara. Keberhasilan perlindungan dievaluasi dengan memastikan penyerang tidak dapat membaca *plaintext*, melainkan hanya melihat *ciphertext* acak di terminal.
2. Pada prosedur serangan *tampering*, penyerang menginjeksikan *ciphertext* yang *byte*-nya telah dimanipulasi secara acak. Kelemahan sistem dibuktikan melalui *log gateway* yang tetap mendekripsi pesan cacat tersebut menjadi data sampah (*garbage data*) dan menyimpannya ke basis data, karena ketiadaan mekanisme HMAC untuk menolaknya.
3. Pada prosedur serangan *replay attack*, penyerang menyuntikkan ulang paket *ciphertext* usang yang valid. Bukti kelemahan dievaluasi dengan melihat *gateway* yang masih rentan menerima paket duplikasi tersebut karena tidak adanya pengecekan kebaruan data.

4.4.2.4 Prosedur Pengujian AES-128 dan HMAC (*Static Key*)

Pengujian ini mengevaluasi sistem yang telah dilengkapi enkripsi dan validasi HMAC-SHA256, namun masih menggunakan kunci rahasia yang bersifat statis (tetap). Rincian prosedur serangannya adalah:

- a. Pada prosedur *Serangan Tampering* penyerang menginjeksikan paket modifikasi ke arah *gateway*. Peningkatan keamanan dibuktikan melalui *gateway* yang kini mampu mendeteksi adanya manipulasi tersebut dan secara mutlak membuang (*drop*) paket cacat dengan status penolakan



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

"*HMAC Invalid*". Hal ini membuktikan bahwa penggunaan HMAC mampu menjaga integritas pesan meskipun menggunakan kunci statis.

- b. Pada prosedur Serangan *Replay Attack* penyerang menyuntikkan ulang paket usang (berserta HMAC valid yang berhasil disadap sebelumnya) secara bertubi-tubi ke gateway. Bukti kelemahan sistem keamanan ini dievaluasi ketika gateway berhasil dijebol dan menerima eksploitasi *replay attack* tersebut. Parameter kunci yang statis serta ketiadaan lapisan perlindungan *sequence counter* membuat gateway tidak mampu mendeteksi bahwa paket tersebut adalah data duplikasi yang sudah kedaluwarsa.

4.4.2.5 Prosedur Pengujian Sistem Mitigasi Penuh

Tahap ini merupakan puncak pengujian pada arsitektur keamanan penuh (*Full Security*) yang diusulkan, di mana *Session Key* diperbarui secara otomatis menggunakan KDF setiap kelipatan 10 paket. Rincian prosedur serangannya adalah:

1. Prosedur Serangan Tampering Penyerang secara masif melakukan eksploitasi modifikasi susunan byte paket di udara. Keberhasilan mitigasi dievaluasi dengan memastikan sistem gateway membuang 100% paket cacat tersebut melalui validasi "*HMAC Invalid*".
2. Prosedur Serangan *Replay Attack* Penyerang menahan paket valid dan mengirimkan ulang paket kadaluwarsa tersebut di sesi komunikasi berikutnya. Keberhasilan sistem (keterbaruan) dievaluasi melalui *Serial Monitor gateway* yang menolak mutlak seluruh transmisi ulang tersebut dengan memunculkan status "*Replay Detected*", membuktikan bahwa mekanisme perlindungan *Counter-based Nonce* dan pembaruan kunci dinamis berjalan secara efektif.

4.4.2.6 Prosedur Pengujian *Quality of Service* (QoS)

Pengujian QoS (*Quality of Service*) dilakukan untuk mengetahui kualitas layanan transmisi data antara perangkat dan klien setelah diberikan beban komputasi dari algoritma keamanan (KDF, AES-128, dan HMAC-SHA256). Parameter utama yang diukur dalam pengujian ini meliputi delay (latensi),



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

throughput, dan packet loss. Rincian prosedur pengujiannya adalah sebagai berikut:

1. Mengonfigurasi perangkat *end-node* untuk mengirimkan paket data sensor secara terus-menerus menuju *gateway* dalam skenario jumlah paket tertentu.
2. Mencatat waktu awal pengiriman paket pada *end-node* (*transmit time*) dan waktu saat paket berhasil didekripsi di *gateway* (*receive time*) untuk menghitung rata-rata *delay* komputasi.
3. Menghitung total byte data (*payload*) yang berhasil ditransmisikan dan membaginya dengan total waktu pengiriman untuk mendapatkan nilai throughput.
4. Membandingkan total jumlah paket yang dikirimkan oleh *end-node* dengan total paket valid yang sukses masuk ke database untuk mengevaluasi persentase paket yang hilang atau packet loss.
5. Membandingkan hasil perhitungan ketiga parameter tersebut dengan standar kualitas jaringan telekomunikasi untuk menyimpulkan tingkat kelayakan sistem secara empiris.

4.5 Data Hasil Pengujian

Data hasil pengujian menampilkan rekapitulasi data kuantitatif yang diperoleh dari seluruh skenario pengujian yang telah dijalankan. Data ini mencakup parameter kinerja komputasi mikrokontroler, kinerja jaringan LoRa, serta efektivitas keamanan sistem dalam memblokir serangan.

4.5.1 Data Hasil Pengujian Fase *Scanning*

Sebelum dapat mengeksekusi serangan aktif pada skenario pengujian, perangkat penyerang terlebih dahulu diuji kemampuannya dalam memindai (*scanning*) lalu lintas udara. Hasil pengujian pada fase *scanning* menunjukkan bahwa attacker node yang dibangun menggunakan perangkat keras yang identik berhasil mendeteksi dan mengunci parameter fisik jaringan target secara otomatis. Penyerang berhasil menyinkronkan lapisan radionya pada frekuensi pusat 433 MHz dengan *Spreading Factor* (SF) 7. Keberhasilan pengintaian ini dibuktikan dengan munculnya status log "*NETWORK LOCKED*" pada antarmuka *Serial Monitor* penyerang, yang menandakan bahwa peretas telah bersiap di kanal yang



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

tepat untuk menyadap dan menginjeksikan paket serangan. Bukti *log* keberhasilan pada fase *scanning* tersebut ditunjukkan pada Gambar 4.23.

```
[SYSTEM] INITIALIZING ATTACKER NODE... DONE
[SYSTEM] HARDWARE: ESP32 dual-core | TRANSCIEVER: LoRa Ra-02 (SX1278)
[RECON] STARTING VULNERABILITY SCANNING ON PIT P2P CHANNELS...

[SCAN] | proc_time=106158us | freq=431.00MHz | sf=SF7 | status=NO_CARRIER_DETECTED
[SCAN] | proc_time=106251us | freq=431.00MHz | sf=SF8 | status=NO_CARRIER_DETECTED
[SCAN] | proc_time=105796us | freq=431.00MHz | sf=SF9 | status=NO_CARRIER_DETECTED
[SCAN] | proc_time=106553us | freq=431.00MHz | sf=SF10 | status=NO_CARRIER_DETECTED
[SCAN] | proc_time=106310us | freq=432.00MHz | sf=SF7 | status=NO_CARRIER_DETECTED
[SCAN] | proc_time=106023us | freq=432.00MHz | sf=SF8 | status=NO_CARRIER_DETECTED
[SCAN] | proc_time=105539us | freq=432.00MHz | sf=SF9 | status=NO_CARRIER_DETECTED
[SCAN] | proc_time=105940us | freq=432.00MHz | sf=SF10 | status=NO_CARRIER_DETECTED
[SCAN] | proc_time=106224us | freq=433.00MHz | sf=SF7 | status=NO_CARRIER_DETECTED

NETWORK LOCKED
FREQUENCY      : 433.00 MHz
SPREADING FACTOR (SF) : SF7
BANDWIDTH      : 125.00 kHz
```

Gambar 4.23 Log Hasil *Scanning* Jaringan LoRa

4.5.2 Data Hasil Pengujian Sistem *Baseline*

Pengujian ini mengevaluasi sistem komunikasi bawaan LoRa secara murni ketika mentransmisikan data sensor tanpa adanya penerapan mekanisme perlindungan kriptografi apa pun pada lapisan aplikasi. Pada tahap awal, penyerang melancarkan serangan penyadapan (*sniffing*) untuk memonitor lalu lintas udara. Karena data dikirimkan tanpa enkripsi, penyerang dapat dengan mudah melihat format dan struktur *payload* asli yang berisi nilai suhu dan kelembapan secara terbuka (*plaintext*). Bukti hasil penyadapan data secara terbuka pada fase *baseline* ini dapat dilihat pada Gambar 4.24.

```
[SNIFF] 3,31.0,70.1,PANAS
[SNIFF] 4,31.0,70.1,PANAS
[SNIFF] 5,31.1,70.0,PANAS
[SNIFF] 6,31.1,69.8,PANAS
[SNIFF] 7,31.0,69.9,PANAS
[SNIFF] 8,31.1,70.0,PANAS
[SNIFF] 9,31.0,69.8,PANAS
[SNIFF] 10,31.0,69.8,PANAS
[SNIFF] 11,31.1,69.7,PANAS
[SNIFF] 12,31.0,69.9,PANAS
```

Gambar 4.24 Log Hasil *Sniffing Baseline*

Setelah penyerang mengetahui format struktur data yang tidak terenkripsi tersebut, penyerang melancarkan serangan injeksi dengan memanipulasi nilai sensor pada *payload* secara paksa (*tampering*). Karena sistem gateway pada mode *baseline* tidak dilengkapi dengan protokol validasi integritas (*Message Authentication Code*), paket cacat hasil modifikasi tersebut sama sekali tidak



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumpukan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

dapat dideteksi. Gateway memproses paket manipulasi tersebut dan menyimpannya secara permanen ke dalam basis data sebagai data yang sah. Bukti kerentanan sistem yang menerima data palsu hasil modifikasi ini ditunjukkan pada Gambar 4.25.

[RX]	id=126	payload=126,31.2,70.5,PANAS	proc_rx=32330us	packet_size=19	ram=333516	rssl=-42	snr=9.50
[RX]	id=126	payload=126,36.6,94.0,PANAS	proc_rx=32354us	packet_size=19	ram=333516	rssl=-45	snr=9.28
[RX]	id=127	payload=127,31.2,70.5,PANAS	proc_rx=32278us	packet_size=19	ram=333516	rssl=-42	snr=9.50
[RX]	id=127	payload=127,42.4,17.9,DINGIN	proc_rx=32496us	packet_size=20	ram=333516	rssl=-42	snr=9.39
[RX]	id=128	payload=128,31.2,70.5,PANAS	proc_rx=32316us	packet_size=19	ram=333516	rssl=-42	snr=9.50
[RX]	id=128	payload=128,22.2,20.8,PANAS	proc_rx=32278us	packet_size=19	ram=333516	rssl=-43	snr=9.49
[RX]	id=129	payload=129,31.2,70.5,PANAS	proc_rx=32466us	packet_size=19	ram=333516	rssl=-42	snr=9.50
[RX]	id=129	payload=129,14.6,43.5,NORMAL	proc_rx=32247us	packet_size=20	ram=333516	rssl=-40	snr=9.72
[RX]	id=130	payload=130,31.2,70.5,PANAS	proc_rx=32528us	packet_size=19	ram=333516	rssl=-42	snr=9.50
[RX]	id=130	payload=130,28.0,64.4,PANAS	proc_rx=32225us	packet_size=19	ram=333516	rssl=-41	snr=9.28
[RX]	id=131	payload=131,31.2,70.5,PANAS	proc_rx=32620us	packet_size=19	ram=333516	rssl=-42	snr=9.50
[RX]	id=131	payload=131,20.1,85.6,PANAS	proc_rx=32599us	packet_size=19	ram=333516	rssl=-39	snr=9.28

Gambar 4.25 Log Hasil Serangan *Tampering* Pada Gateway

Selanjutnya pada skenario pengujian pengiriman ulang paket (*replay attack*), penyerang menahan sebuah paket *plaintext* yang valid dan menginjeksikannya kembali ke *gateway* secara berulang-ulang dengan jeda waktu tertentu. Karena ketiadaan parameter *sequence counter* sebagai *nonce* untuk memvalidasi kebaruan pesan, *gateway* secara pasif memproses seluruh paket usang tersebut. Hal ini mengakibatkan terjadinya duplikasi data secara fatal di dalam basis data. Bukti terjadinya pencatatan data duplikat akibat serangan *replay* tersebut ditunjukkan pada Gambar 4.26.

[RX]	id=2	payload=2,31.2,70.5,PANAS	proc_rx=32555us	packet_size=17	ram=333516	rssl=-42	snr=9.50
[RX]	id=2	payload=2,31.2,70.5,PANAS	proc_rx=32550us	packet_size=17	ram=333516	rssl=-46	snr=9.45
[RX]	id=3	payload=3,31.2,70.5,PANAS	proc_rx=32522us	packet_size=17	ram=333516	rssl=-42	snr=9.50
[RX]	id=3	payload=3,31.2,70.5,PANAS	proc_rx=32684us	packet_size=17	ram=333516	rssl=-44	snr=9.01
[RX]	id=4	payload=4,31.2,70.5,PANAS	proc_rx=32641us	packet_size=17	ram=333516	rssl=-42	snr=9.50
[RX]	id=4	payload=4,31.2,70.5,PANAS	proc_rx=32411us	packet_size=17	ram=333516	rssl=-40	snr=9.11
[RX]	id=5	payload=5,31.2,70.5,PANAS	proc_rx=32357us	packet_size=17	ram=333516	rssl=-42	snr=9.50
[RX]	id=5	payload=5,31.2,70.5,PANAS	proc_rx=32419us	packet_size=17	ram=333516	rssl=-40	snr=9.78
[RX]	id=6	payload=6,31.2,70.5,PANAS	proc_rx=32740us	packet_size=17	ram=333516	rssl=-42	snr=9.50
[RX]	id=6	payload=6,31.2,70.5,PANAS	proc_rx=32514us	packet_size=17	ram=333516	rssl=-39	snr=9.76
[RX]	id=7	payload=7,31.2,70.5,PANAS	proc_rx=32531us	packet_size=17	ram=333516	rssl=-42	snr=9.50
[RX]	id=7	payload=7,31.2,70.5,PANAS	proc_rx=32405us	packet_size=17	ram=333516	rssl=-44	snr=9.12
[RX]	id=8	payload=8,31.2,70.5,PANAS	proc_rx=32639us	packet_size=17	ram=333516	rssl=-42	snr=9.50
[RX]	id=8	payload=8,31.2,70.5,PANAS	proc_rx=32371us	packet_size=17	ram=333516	rssl=-39	snr=9.61
[RX]	id=9	payload=9,31.2,70.5,PANAS	proc_rx=32381us	packet_size=17	ram=333516	rssl=-42	snr=9.50
[RX]	id=9	payload=9,31.2,70.5,PANAS	proc_rx=32346us	packet_size=17	ram=333516	rssl=-39	snr=9.89
[RX]	id=10	payload=10,31.2,70.5,PANAS	proc_rx=32293us	packet_size=18	ram=333516	rssl=-42	snr=9.50
[RX]	id=10	payload=10,31.2,70.5,PANAS	proc_rx=32572us	packet_size=18	ram=333516	rssl=-42	snr=9.46

Gambar 4.26 Log Hasil *Replay Attack* Pada Gateway

4.5.3 Data Hasil Pengujian Keamanan Hanya AES-128

Pada skenario ini, sistem diuji dengan menerapkan algoritma enkripsi AES-128 tanpa didampingi mekanisme validasi integritas HMAC-SHA256 maupun *sequence counter*. Berdasarkan hasil pengujian penyadapan (*sniffing*), kerahasiaan data berhasil dilindungi karena penyerang hanya melihat deretan



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

ciphertext acak di udara. Bukti hasil penyadapan *ciphertext* pada skenario ini ditunjukkan pada Gambar 4.27.

```
[SNIFF] payload=47EDEBE46DF5D28C8947CA70912357CF | rssi=-41 | snr=8.58
[SNIFF] payload=513D18612C9C4FA6930200974018E6C7 | rssi=-41 | snr=9.44
[SNIFF] payload=1E07552F1166F8C71A467F566115E935 | rssi=-45 | snr=9.50
[SNIFF] payload=C8DC118438F861106F0FD48057CA6679 | rssi=-40 | snr=9.55
[SNIFF] payload=AAD21381B97D14B491473E5BA394CAB0 | rssi=-41 | snr=8.61
[SNIFF] payload=249A61D70C1142B6D8132B28CECC387F | rssi=-45 | snr=9.21
[SNIFF] payload=DF5961B9C2276A0E87C2E3A58213068C | rssi=-40 | snr=9.99
[SNIFF] payload=323FDC72D3C985424559E0C15806CC89 | rssi=-44 | snr=9.15
```

Gambar 4.27 Log Hasil *Sniffing* pada Skenario AES128

Akan tetapi, sistem terbukti rentan pada pengujian serangan modifikasi data (*tampering*). Perubahan susunan *byte* pada *ciphertext* yang dilakukan oleh penyerang tidak dapat dideteksi oleh *gateway*. Akibatnya, *gateway* tetap mengeksekusi proses dekripsi pada pesan cacat tersebut yang mengubahnya menjadi data sampah (*garbage data*) dan menyimpannya ke dalam *database*. Bukti kegagalan sistem dalam memitigasi manipulasi *byte* ini ditunjukkan pada Gambar 4.28.

```
[RX] id=6 | payload=6,31.2,70.5,PANAS | proc_rx=32740us | packet_size=17 | ram=333516 | rssi=-42 | snr=9.50
[RX] id=6 | payload=6,31.2,70.5,PANAS | proc_rx=32514us | packet_size=17 | ram=333516 | rssi=-39 | snr=9.76
[RX] id=7 | payload=7,31.2,70.5,PANAS | proc_rx=32531us | packet_size=17 | ram=333516 | rssi=-42 | snr=9.50
[RX] id=7 | payload=7,31.2,70.5,PANAS | proc_rx=32405us | packet_size=17 | ram=333516 | rssi=-44 | snr=9.12
[RX] id=8 | payload=8,31.2,70.5,PANAS | proc_rx=32639us | packet_size=17 | ram=333516 | rssi=-42 | snr=9.50
[RX] id=8 | payload=8,31.2,70.5,PANAS | proc_rx=32371us | packet_size=17 | ram=333516 | rssi=-39 | snr=9.61
[RX] id=9 | payload=9,31.2,70.5,PANAS | proc_rx=32381us | packet_size=17 | ram=333516 | rssi=-42 | snr=9.50
[RX] id=9 | payload=9,31.2,70.5,PANAS | proc_rx=32346us | packet_size=17 | ram=333516 | rssi=-39 | snr=9.89
[RX] id=10 | payload=10,31.2,70.5,PANAS | proc_rx=32293us | packet_size=18 | ram=333516 | rssi=-42 | snr=9.50
[RX] id=10 | payload=10,31.2,70.5,PANAS | proc_rx=32572us | packet_size=18 | ram=333516 | rssi=-42 | snr=9.46
```

Gambar 4.28 Log Hasil *Tampering* pada Skenario AES128

Selain itu, eksploitasi pengiriman ulang paket usang (*replay attack*) juga berhasil menembus *gateway*. Ketiadaan mekanisme pengecekan kebaruan data membuat paket duplikasi diterima begitu saja. Bukti kerentanan sistem terhadap serangan *replay* pada skenario ini ditunjukkan pada Gambar 4.29.

```
[RX] id=74 | payload=74,1.8,24.6,NORMAL | proc_rx=2019us | packet_size=54 | ram=350176 | rssi=-55 | snr=9.75
[RX] id=4259914 | DROP: HMAC INVALID (Data Tampering Detected!)
[RX] id=16714 | DROP: HMAC INVALID (Data Tampering Detected!)
[RX] id=75 | payload=75,1.8,24.6,NORMAL | proc_rx=2022us | packet_size=54 | ram=350176 | rssi=-55 | snr=9.50
[RX] id=74 | DROP: HMAC INVALID (Data Tampering Detected!)
[RX] id=74 | DROP: HMAC INVALID (Data Tampering Detected!)
[RX] id=76 | payload=76,1.8,24.6,NORMAL | proc_rx=2017us | packet_size=54 | ram=350176 | rssi=-54 | snr=10.50
[RX] id=76 | DROP: HMAC INVALID (Data Tampering Detected!)
```

Gambar 4.29 Log Hasil *Replay Attack* pada Skenario AES128

4.5.4 Data Hasil Pengujian Keamanan AES-128 dan HMAC (Kunci Statis)

Pengujian ini mengevaluasi sistem enkripsi AES-128 yang menggunakan satu kunci rahasia secara permanen (kunci statis) tanpa didampingi mekanisme *Key Derivation Function* (KDF) maupun validasi integritas HMAC. Pada tahap awal,



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

penyerang meluncurkan serangan penyadapan (*sniffing*). Guna membuktikan kelemahan kunci statis secara empiris, pengujian pada skenario ini dikondisikan menggunakan data sensor suhu yang dikunci pada nilai konstan. Meskipun *payload* telah disandikan menjadi ciphertext dan isi pesannya tidak dapat dibaca secara transparan, penggunaan kunci statis yang dipadukan dengan input data konstan menyebabkan ciphertext yang tertangkap di udara membentuk pola byte yang identik dan berulang secara prediktif. Pola sandi kembar ini terbukti membuat sistem sangat rentan dieksploitasi melalui teknik analisis frekuensi (*frequency analysis*), terutama apabila perangkat pengirim mengalami restart yang me-reset parameter transmisi. Bukti hasil penyadapan ciphertext statis dengan pola berulang tersebut ditunjukkan pada Gambar 4.30.

[SNIFF]	payload=18C54BFE12671A08D41D0984CE6AF249		rss=-41		snr=9.79
[SNIFF]	payload=18C44BFE12671A08D41D0984CE6AF249		rss=-42		snr=9.79
[SNIFF]	payload=18C74BFE12671A08D41D0984CE6AF249		rss=-45		snr=9.31
[SNIFF]	payload=18C64BFE12671A08D41D0984CE6AF249		rss=-43		snr=8.58
[SNIFF]	payload=18C14BFE12671A08D41D0984CE6AF249		rss=-44		snr=9.68
[SNIFF]	payload=18C04BFE12671A08D41D0984CE6AF249		rss=-40		snr=9.89
[SNIFF]	payload=18C34BFE12671A08D41D0984CE6AF249		rss=-41		snr=9.27
[SNIFF]	payload=18C24BFE12671A08D41D0984CE6AF249		rss=-44		snr=9.08
[SNIFF]	payload=18CD4BFE12671A08D41D0984CE6AF249		rss=-45		snr=9.75
[SNIFF]	payload=18CC4BFE12671A08D41D0984CE6AF249		rss=-41		snr=8.89
[SNIFF]	payload=18C54BFE12671A08D41D0984CE6AF249		rss=-45		snr=10.00
[SNIFF]	payload=18C44BFE12671A08D41D0984CE6AF249		rss=-41		snr=9.98

Gambar 4.30 Log Hasil Serangan *Sniffing Static Key*

Selanjutnya, penyerang meluncurkan serangan modifikasi data (*tampering*) dengan mengubah susunan byte ciphertext yang melintas di udara secara paksa. Karena ketiadaan mekanisme *Message Authentication Code* (MAC) untuk memverifikasi integritas paket di sisi penerima, gateway gagal mendeteksi adanya manipulasi tersebut. Akibatnya, *gateway* secara pasif tetap mendekripsi paket cacat tersebut, yang pada akhirnya menghasilkan data rusak (*garbage data*) dan menyimpannya ke basis data. Bukti kegagalan sistem dalam memitigasi manipulasi byte ini ditunjukkan pada Gambar 4.31.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

[RX]	id=16	payload=16,31.2,70.5,PANAS	proc_rx=32627us	packet_size=18	ram=333516	rss=-42	snr=9.50
[RX]	id=16	payload=16,46.8,27.2,DINGIN	proc_rx=32730us	packet_size=19	ram=333516	rss=-40	snr=9.63
[RX]	id=17	payload=17,31.2,70.5,PANAS	proc_rx=32432us	packet_size=18	ram=333516	rss=-42	snr=9.50
[RX]	id=17	payload=17,43.0,28.6,PANAS	proc_rx=32329us	packet_size=18	ram=333516	rss=-43	snr=8.79
[RX]	id=18	payload=18,31.2,70.5,PANAS	proc_rx=32722us	packet_size=18	ram=333516	rss=-42	snr=9.50
[RX]	id=18	payload=18,43.2,76.9,PANAS	proc_rx=32273us	packet_size=18	ram=333516	rss=-42	snr=9.67
[RX]	id=19	payload=19,31.2,70.5,PANAS	proc_rx=32674us	packet_size=18	ram=333516	rss=-42	snr=9.50
[RX]	id=19	payload=19,13.5,69.5,NORMAL	proc_rx=32270us	packet_size=19	ram=333516	rss=-47	snr=9.93
[RX]	id=20	payload=20,31.2,70.5,PANAS	proc_rx=32232us	packet_size=18	ram=333516	rss=-42	snr=9.50
[RX]	id=20	payload=20,40.1,48.6,NORMAL	proc_rx=32648us	packet_size=19	ram=333516	rss=-48	snr=9.60
[RX]	id=21	payload=21,31.2,70.5,PANAS	proc_rx=32473us	packet_size=18	ram=333516	rss=-42	snr=9.50
[RX]	id=21	payload=21,33.8,51.5,NORMAL	proc_rx=32242us	packet_size=19	ram=333516	rss=-45	snr=8.64
[RX]	id=22	payload=22,31.2,70.5,PANAS	proc_rx=32449us	packet_size=18	ram=333516	rss=-42	snr=9.50
[RX]	id=22	payload=22,22.0,62.8,NORMAL	proc_rx=32579us	packet_size=19	ram=333516	rss=-44	snr=9.08

Gambar 4.31 Log Hasil Tampering Pada Gateway dengan Kunci Statis

Pada pengujian terakhir, penyerang mengeksekusi pengiriman ulang paket (*replay attack*) dengan menahan *ciphertext* yang valid dan menginjeksikannya kembali ke gateway pada sesi berikutnya. Walaupun paket tersebut terenkripsi, *gateway* tetap mendekripsi dan menerimanya secara mutlak karena ketiadaan parameter *sequence counter* (*nonce*) yang berfungsi untuk memvalidasi dan menolak paket usang. Hal ini membuktikan bahwa penerapan enkripsi statis murni tidak mampu mencegah terjadinya duplikasi data akibat injeksi peretas. Bukti rekaman *log* serangan *replay* pada skenario ini ditunjukkan pada Gambar 4.32.

[RX]	id=2	payload=2,31.2,70.5,PANAS	proc_rx=32215us	packet_size=17	ram=333516	rss=-47	snr=9.41
[RX]	id=3	payload=3,31.2,70.5,PANAS	proc_rx=32421us	packet_size=17	ram=333516	rss=-42	snr=9.50
[RX]	id=3	payload=3,31.2,70.5,PANAS	proc_rx=32733us	packet_size=17	ram=333516	rss=-39	snr=9.95
[RX]	id=4	payload=4,31.2,70.5,PANAS	proc_rx=32519us	packet_size=17	ram=333516	rss=-42	snr=9.50
[RX]	id=4	payload=4,31.2,70.5,PANAS	proc_rx=32646us	packet_size=17	ram=333516	rss=-46	snr=9.37
[RX]	id=5	payload=5,31.2,70.5,PANAS	proc_rx=32230us	packet_size=17	ram=333516	rss=-42	snr=9.50
[RX]	id=5	payload=5,31.2,70.5,PANAS	proc_rx=32292us	packet_size=17	ram=333516	rss=-47	snr=8.63
[RX]	id=6	payload=6,31.2,70.5,PANAS	proc_rx=32289us	packet_size=17	ram=333516	rss=-42	snr=9.50
[RX]	id=6	payload=6,31.2,70.5,PANAS	proc_rx=32556us	packet_size=17	ram=333516	rss=-45	snr=9.07
[RX]	id=7	payload=7,31.2,70.5,PANAS	proc_rx=32701us	packet_size=17	ram=333516	rss=-42	snr=9.50
[RX]	id=7	payload=7,31.2,70.5,PANAS	proc_rx=32543us	packet_size=17	ram=333516	rss=-46	snr=9.85

Gambar 4.32 Log Hasil Replay Attack Pada Gateway dengan Kunci Statis

4.5.5 Data Hasil Pengujian Sistem Mitigasi Penuh

Pengujian ini merupakan tahap validasi puncak yang mengaktifkan seluruh arsitektur keamanan usulan secara bersamaan, meliputi enkripsi AES-128 mode CTR, rotasi kunci dinamis (*Key Derivation Function*), serta verifikasi integritas pesan berbasis HMAC-SHA256.

Berbeda dengan skenario sebelumnya yang rentan terhadap analisis frekuensi, penerapan KDF pada sistem ini memastikan bahwa *session key* selalu diperbarui secara otomatis menggunakan nilai *sequence counter*. Hal ini mengeliminasi terbentuknya pola *ciphertext* yang sama di udara. Dengan demikian, upaya penyadapan (*sniffing*) oleh penyerang hanya akan menghasilkan deretan *byte* acak yang selalu berubah pada setiap kelipatan sesi, sehingga kerahasiaan



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengemukakan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

payload terlindungi sepenuhnya. Bukti hasil penyadapan *ciphertext* dengan pola *byte* acak dinamis tersebut ditunjukkan pada Gambar 4.33.

```
[SNIFF] payload=6CCE7516393545EB7F01D9E9DB73C61D | rssi=-40 | snr=8.84
[SNIFF] payload=23AFC30B29BE468FA359FF17567F473E | rssi=-42 | snr=9.20
[SNIFF] payload=540BA13DE51A25551016FB1B5A72F80E | rssi=-42 | snr=9.20
[SNIFF] payload=8FF75B1A9F1E23F998B52B7CB3E5A2BF | rssi=-45 | snr=9.43
[SNIFF] payload=6CB8AD975716071A73AF2D1F3A7C2132 | rssi=-42 | snr=9.42
[SNIFF] payload=7CE81F5F10795F914148204ABEB91C0E | rssi=-42 | snr=8.82
[SNIFF] payload=F46CB4DD882373F1594358F0D783DABD | rssi=-40 | snr=8.90
[SNIFF] payload=B523BD168CE469244E51837C181FA871 | rssi=-45 | snr=9.54
```

Gambar 4.33 Log Hasil Serangan *Sniffing* Pada Kunci Dinamis

Selanjutnya, ketika penyerang meluncurkan serangan modifikasi data (*tampering*) dengan mencoba mengubah susunan *byte ciphertext* secara paksa di udara, sistem pertahanan lapis kedua langsung merespons. Manipulasi sekecil apa pun pada struktur *payload* secara otomatis merusak keabsahan tanda tangan digital di sisi penerima. Gateway secara mutlak menolak dan membuang (*drop*) paket cacat tersebut dengan menampilkan status peringatan kegagalan verifikasi HMAC, sehingga integritas basis data tetap terjaga dan terhindar dari rekaman data rusak (*garbage data*). Bukti log penolakan mutlak oleh sistem *gateway* terhadap serangan *tampering* ini ditunjukkan pada Gambar 4.34.

```
[RX] id=70 | payload=70,1.8,24.7,NORMAL | proc_rx=2015us | packet_size=54 | ram=350176 | rssi=-55 | snr=9.50
[RX] id=66 | DROP: HMAC INVALID (Data Tampering Detected!)
[RX] id=69 | DROP: HMAC INVALID (Data Tampering Detected!)
[-] KDF SYNCED! Session Key aktif untuk Base ID: 71
[RX] id=71 | payload=71,1.8,24.7,NORMAL | proc_rx=2218us | packet_size=54 | ram=350176 | rssi=-55 | snr=9.50
[RX] id=71 | DROP: HMAC INVALID (Data Tampering Detected!)
[RX] id=72 | payload=72,1.8,24.7,NORMAL | proc_rx=2020us | packet_size=54 | ram=350176 | rssi=-56 | snr=9.75
[RX] id=71 | DROP: HMAC INVALID (Data Tampering Detected!)
[RX] id=73 | payload=73,1.8,24.7,NORMAL | proc_rx=2020us | packet_size=54 | ram=350176 | rssi=-55 | snr=9.50
[RX] id=73 | DROP: HMAC INVALID (Data Tampering Detected!)
[RX] id=74 | payload=74,1.8,24.6,NORMAL | proc_rx=2019us | packet_size=54 | ram=350176 | rssi=-55 | snr=9.75
[RX] id=4259914 | DROP: HMAC INVALID (Data Tampering Detected!)
```

Gambar 4.34 Log Penolakan Serangan *Tampering* pada Gateway

Selanjutnya, pada skenario pengujian serangan pengiriman ulang paket (*replay attack*), penyerang mencoba menginjeksi ulang *ciphertext* usang yang sebelumnya berhasil disadap ke arah *gateway*. Namun, berkat integrasi parameter *sequence counter* sebagai *nonce* pada proses autentikasi pesan, *gateway* mampu mengidentifikasi bahwa nilai *counter* pada paket tersebut sudah kedaluwarsa karena bernilai sama atau lebih rendah dari yang tersimpan di memori sistem. Sistem *gateway* langsung menggugurkan seluruh transmisi ulang tersebut, yang membuktikan bahwa mekanisme pembaruan kunci dinamis berjalan sangat efektif dalam mencegah terjadinya eksploitasi duplikasi data.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Bukti rekaman *log* penolakan serangan *replay* oleh *gateway* ditunjukkan pada Gambar 4.35.

```
[RX] id=18 | payload=18,1.7,24.8,NORMAL | proc_rx=2031us | packet_size=54 | ram=350176 | rssi=-55 | snr=-10.00
[RX] id=17 | DROP: REPLAY DETECTED (Paket Expired atau Sudah Pernah Diterima)
[RX] id=17 | DROP: REPLAY DETECTED (Paket Expired atau Sudah Pernah Diterima)
[RX] id=18 | payload=18,1.7,24.8,NORMAL | proc_rx=2029us | packet_size=54 | ram=350176 | rssi=-55 | snr=-9.25
[RX] id=19 | DROP: REPLAY DETECTED (Paket Expired atau Sudah Pernah Diterima)
[RX] id=19 | DROP: REPLAY DETECTED (Paket Expired atau Sudah Pernah Diterima)
[RX] id=19 | DROP: REPLAY DETECTED (Paket Expired atau Sudah Pernah Diterima)
[RX] id=19 | DROP: REPLAY DETECTED (Paket Expired atau Sudah Pernah Diterima)
[RX] id=20 | payload=20,1.7,24.8,NORMAL | proc_rx=2023us | packet_size=54 | ram=350176 | rssi=-54 | snr=-9.50
[-] KDF SYNCED! Session Key aktif untuk Base ID: 21
```

Gambar 4.35 Log Penolakan *Replay Attack* pada *Gateway*

Untuk memberikan gambaran komprehensif mengenai signifikansi peningkatan keamanan yang dicapai, hasil perbandingan tingkat keberhasilan sistem dalam memitigasi serangan (*drop rate*) antara sistem *baseline*, sistem AES statis, dan sistem mitigasi penuh dirangkum pada Tabel 4.4. berikut.

Tabel 4.4 Perbandingan Tingkat Keberhasilan Mitigasi Serangan

Jenis Serangan	Jumlah Injeksi	Sistem <i>Baseline</i>	Keamanan hanya AES-128	Keamanan AES-128 dan HMAC	<i>Full Security</i>
<i>Sniffing</i>	50 Paket	100% (<i>Plain</i>)	100% (<i>Cipher</i>)	100% (<i>Cipher</i>)	100% (<i>Cipher</i>)
<i>Replay Attack</i>	50 Paket	100% (<i>Lolos</i>)	0% (<i>Lolos</i>)	100% (<i>Lolos</i>)	100% (<i>Drop</i>)
<i>Data Tampering</i>	50 Paket	100% (<i>Lolos</i>)	0% (<i>Lolos</i>)	100% (<i>Drop</i>)	100% (<i>Drop</i>)

4.5.6 Data Hasil *Quality of Service* (QoS)

Pengujian *Quality of Service* (QoS) dilakukan untuk mengevaluasi kualitas layanan transmisi data secara nirkabel pada jaringan LoRa *Point-to-Point* (P2P) antara perangkat *End-Node* dan *Gateway*. Parameter utama yang diukur dalam pengujian jaringan ini meliputi waktu tunda transmisi (*Latensi/Delay*), laju transfer data (*Throughput*), dan persentase paket yang hilang di udara (*Packet Loss*). Hasil pengujian komparatif yang membandingkan kinerja jaringan komunikasi pada kondisi normal antara sistem *baseline*, sistem dengan enkripsi murni, dan sistem mitigasi penuh ditunjukkan pada Tabel 4.5.



Tabel 4.5. Hasil Perbandingan *Quality of Service* (QoS)

Skenario Pengujian	Ukuran <i>Payload</i>	Latensi (<i>Delay</i>)	<i>Throughput</i>	<i>Packet Loss</i>
Sistem <i>Baseline</i> (Tanpa Keamanan)	18,63 <i>Bytes</i>	42,58 ms	437,53 <i>Bytes/s</i>	0%
Keamanan hanya AES128	22 <i>Bytes</i>	49,22 ms	537,33 <i>Bytes/s</i>	0%
Enkripsi AES128 dan HMAC	35,59 <i>Bytes</i>	55,86 ms	637,13 <i>Bytes/s</i>	0%
Sistem Mitigasi Penuh (<i>Full Security</i>)	54,38 <i>Bytes</i>	69,65 ms	780,76 <i>Bytes/s</i>	0%

Merujuk pada Tabel 4.5, implementasi algoritma keamanan kriptografi secara logis memengaruhi performa jaringan transmisi. Waktu latensi pada sistem baseline tercatat paling responsif sebesar 42,58 ms. Ketika sistem mitigasi penuh diterapkan (kombinasi AES-128, HMAC-SHA256, dan KDF), rata-rata latensi transmisi mengalami peningkatan menjadi 69,65 ms. Peningkatan nilai latensi ini merupakan konsekuensi langsung dari penambahan struktur keamanan berlapis yang menyebabkan penambahan ukuran *payload* data dari 18,63 *Bytes* menjadi 54,38 *Bytes* di udara. Meskipun mengalami peningkatan waktu latensi, metrik *throughput* pada sistem mitigasi penuh justru mencatatkan angka yang optimal, yaitu mencapai 780,76 *Bytes/s*. Di samping itu, tingkat stabilitas pengiriman dinyatakan sangat tangguh dengan nilai *packet loss* yang konsisten berada di angka absolut 0% pada ketiga skenario, menandakan transmisi berjalan lancar tanpa ada data sensor sah yang hilang di jalan.

Selain mengukur rata-rata performa jaringan secara umum, pengujian stabilitas (*stability testing*) juga dilakukan secara spesifik pada skenario Sistem Mitigasi Penuh (*Full Security*). Pengujian ini dilakukan dengan cara mengirimkan 200, 400, dan 600 paket data secara terus-menerus. Tujuannya adalah untuk memverifikasi konsistensi jaringan dan memastikan tidak adanya kebocoran

- Hak Cipta :**
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
 2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

memori (*memory leak*) pada mikrokontroler saat proses pembaruan kunci dinamis berjalan dalam durasi yang panjang. Hasil rekapitulasi pengujian stabilitas ini ditunjukkan pada Tabel 4.6.

Tabel 4.6. Hasil Pengujian Stabilitas Kinerja Sistem Mitigasi Penuh Per Interval

Interval	Ukuran Payload	Latensi	Throughput	Packet Loss	Sisa RAM
200 Paket	53,4 Bytes	69,75 ms	772,10 B/s	0%	350.204
400 Paket	54,1 Bytes	69,90 ms	774,80 B/s	0%	350.204
600 Paket	54,8 Bytes	70,03 ms	777,50 B/s	0%	350.204
Rata-rata Keseluruhan	54,1 Bytes	69,89 ms	774,80 B/s	0%	350.204

4.6 Analisis Data dan Evaluasi Hasil Pengujian

Analisis data dilakukan berdasarkan hasil pengujian yang telah dipaparkan sebelumnya. Evaluasi ini bertujuan untuk membedah korelasi antara penambahan algoritma keamanan dengan peningkatan *overhead* perangkat, serta membuktikan keandalan sistem dalam menggugurkan ancaman siber.

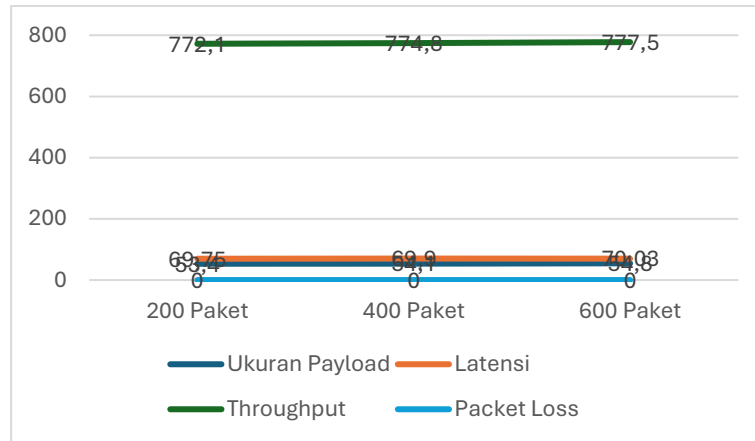
4.6.1 Analisis Stabilitas Sistem Mitigasi Penuh

Pengujian stabilitas (*stability testing*) bertujuan untuk memverifikasi konsistensi jaringan dan memastikan tidak adanya kebocoran memori (*memory leak*) pada mikrokontroler saat proses pembaruan kunci dinamis (KDF) dan komputasi HMAC berjalan dalam durasi yang panjang. Gambar 4.36. menyajikan rekaman data stabilitas yang diukur secara berkala setiap interval 200 paket.



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

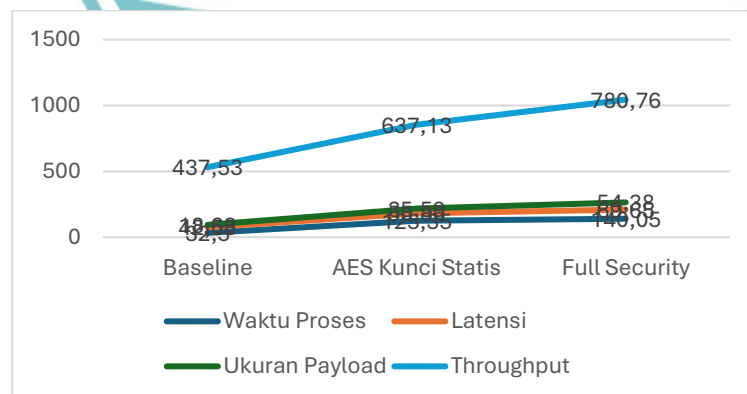


Gambar 4.36 Grafik Stabilitas Sistem (Interval Paket)

Data pada grafik menunjukkan bahwa kinerja sistem keamanan usulan memiliki tingkat stabilitas yang sangat tangguh. Terdapat eskalasi yang sangat linier dan tipis pada rata-rata ukuran payload (dari 53,4 *Bytes* menjadi 54,8 *Bytes*) dan latensi jaringan (dari 69,75 ms menjadi 70,03 ms). Parameter tersebut mengonfirmasi bahwa eksekusi pembaruan kunci dinamis (*Key Derivation Function*) setiap kelipatan 10 paket tidak memicu bottleneck atau penumpukan beban pemrosesan yang masif di antrean mikrokontroler. Konsistensi ketersediaan sisa memori yang tidak berubah di angka 350.204 *Bytes* sejak interval pertama hingga 600 paket menjadi bukti nyata bahwa arsitektur perangkat lunak sukses mengelola sumber daya secara efisien tanpa ancaman *memory leak*.

4.6.2 Analisis Kinerja Komputasi dan Kualitas Jaringan (QoS)

Penerapan keamanan komputasi secara alamiah akan memberikan overhead terhadap performa jaringan. Grafik analisis kinerja komputasi dan kualitas jaringan dapat dilihat pada Gambar 4.37.



Gambar 4.37 Grafik Perbandingan Kinerja Komputasi dan QoS



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

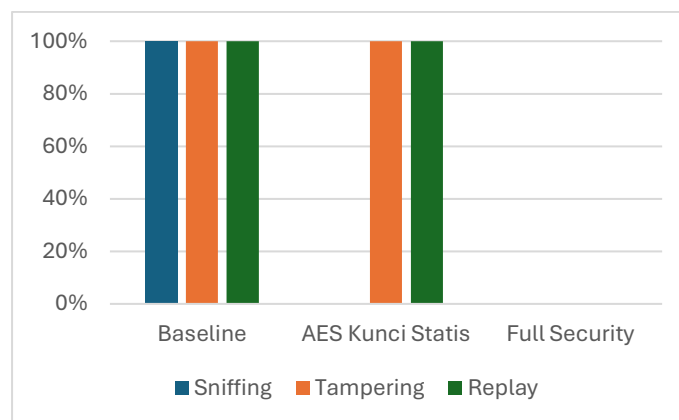
Waktu pemrosesan internal mikrokontroler pada sistem *baseline* yang awalnya hanya membutuhkan 32,50 ms mengalami lonjakan menjadi 125,35 ms saat enkripsi AES-128 murni diterapkan, yang merupakan imbas langsung dari kalkulasi matematis *matrix ciphering*. Pada sistem mitigasi penuh yang mengalkulasi AES-128, HMAC-SHA256, dan KDF secara simultan, waktu pemrosesan menyentuh angka rata-rata 140,05 ms. Penambahan *delay* komputasi ini sangat rasional dan tetap berada dalam spesifikasi operasional responsif untuk skenario pemantauan *Internet of Things* (IoT).

Ditinjau dari beban jaringan di udara, ukuran *payload* bertambah secara signifikan dari 18,63 *Bytes* pada *baseline* menjadi 54,38 *Bytes* pada sistem mitigasi penuh. Pembengkakan dimensi ini merupakan konsekuensi logis dari kewajiban sistem untuk membawa *ciphertext*, *Counter ID*, dan 32-Byte *digest* HMAC-SHA256. Kendati demikian, ukuran paket secara keseluruhan masih berada jauh di bawah limitasi *Maximum Transmission Unit* (MTU) perangkat radio LoRa fisik yang berada di rentang 255 *Bytes*, sehingga dijamin tidak memicu terjadinya fragmentasi paket di udara.

Menariknya, sistem mitigasi penuh justru mencatatkan performa *throughput* tertinggi di angka 780,76 *Bytes/s*. Fenomena ini membuktikan bahwa optimasi lapisan kriptografi yang diterapkan mampu memproses ukuran paket data yang lebih masif dengan efisiensi transfer bit yang sangat optimal di kanal medium transmisi, didukung penuh dengan tingkat keberhasilan pengiriman mencapai 100%.

4.6.3 Analisis Evaluasi Keamanan Sistem (Mitigasi Serangan)

Evaluasi tingkat keamanan sistem diukur secara absolut dari kemampuan secure gateway dalam memfilter dan menolak ancaman dari berbagai vektor serangan. Persentase analisis keamanan system dapat dilihat pada Gambar 4.38.



Gambar 4.38 Grafik Tingkat Keberhasilan Mitigasi Serangan (*Drop Rate*)

Sistem *baseline* dan arsitektur dengan enkripsi AES-128 statis terbukti sangat rapuh terhadap intervensi eksploitasi aktif. Ketiadaan lapisan validasi parameter keaslian pesan memicu *gateway* secara buta menerima injeksi paket usang maupun cacat dari peretas. Pada insiden AES-128 statis, penyerang sukses melancarkan sabotase modifikasi *byte* (*bit-flipping*), dan *gateway* tetap mengeksekusi proses dekripsi yang mengubahnya menjadi data sampah (*garbage data*), sehingga mencederai integritas nilai basis data secara permanen.

Implementasi Sistem Mitigasi Penuh yang diusulkan sukses mengisolasi seluruh celah kerentanan tersebut secara mutlak dengan pencapaian angka *drop rate* 100%. Untuk proses mitigasi *tampering*, implementasi penyegelan akhir HMAC-SHA256 menjamin bahwa modifikasi sekecil satu *bit* di udara akan melahirkan efek longsor matematis (*avalanche effect*) yang menggugurkan kecocokan struktur *hash* di sisi penerima. Kegagalan verifikasi tanda tangan ini memicu *gateway* untuk langsung membuang (*drop*) paket cacat tersebut sebelum memasuki antrean dekripsi AES.

Selanjutnya, mitigasi ancaman *replay attack* sukses dimentahkan secara sepihak berkat sinkronisasi mekanisme *Counter-based Nonce* dan fitur penyegaran *Key Derivation Function* (KDF). Sistem memori *gateway* memblokir upaya pengiriman ulang paket usang secara akurat karena mendeteksi injeksi nilai *counter* yang sudah kedaluwarsa. Keberhasilan validasi menyeluruh ini menegaskan bahwa arsitektur hibrida keamanan pada



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

sistem LoRa P2P memberikan perisai data yang tangguh di lingkungan yang terisolasi dari jaringan internet komersial.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan perancangan, implementasi, dan hasil pengujian komparatif terhadap mitigasi serangan *Man-In-The-Middle* (MITM) pada komunikasi LoRa *Point-to-Point* (P2P) berbasis ESP32, dapat ditarik simpulan sebagai berikut:

1. Implementasi *dynamic key management* menggunakan metode *Key Derivation Function* (KDF) berbasis algoritma HMAC-SHA256 berhasil menggantikan kerentanan sistem kunci statis (*static key*). Sistem secara mandiri dan otomatis mampu melakukan pembaruan kunci sesi (*session key*) setiap kelipatan 10 paket transmisi menggunakan parameter nilai *Sequence Counter*, sehingga memastikan setiap sesi komunikasi memiliki kunci yang unik tanpa perlu melakukan pertukaran kunci secara terbuka di udara.
2. Skema keamanan *Full Security* yang diusulkan terbukti memiliki efektivitas yang absolut dalam memitigasi tiga model serangan siber. Algoritma AES-128 mode CTR sukses menyembunyikan *plaintext* dari serangan penyadapan (*sniffing*). Integrasi HMAC-SHA256 berhasil mencapai *drop rate* 100% terhadap serangan manipulasi data (*tampering* / MITM) karena sistem mampu mendeteksi ketidaksesuaian nilai hash di udara. Selain itu, penggunaan *Counter-based Nonce* juga sukses mencapai *drop rate* 100% dalam memblokir dan menolak serangan pengiriman ulang paket usang (*replay attack*).
3. Penerapan algoritma keamanan kriptografi secara logis memberikan beban komputasi tambahan terhadap performa mikrokontroler ESP32, yang ditandai dengan peningkatan waktu pemrosesan dari 32,50 ms pada sistem *baseline* menjadi 140,05 ms. Walaupun terjadi peningkatan, latensi ini dinilai masih sangat responsif untuk aplikasi IoT. Dari segi penggunaan memori, pustaka kriptografi mbedTLS terbukti sangat ringan (*lightweight*) dengan sisa kapasitas RAM yang bertahan konstan di angka 350.204 *Bytes* selama pengujian transmisi ratusan paket data berdurasi panjang, sehingga mengonfirmasi bahwa sistem mampu mengelola memori secara efisien dan terbebas dari kebocoran memori (*memory leak*).



© Hak Cipta milik Politeknik Negeri Jakarta

5.2 Saran

Meskipun sistem yang dikembangkan telah mencapai tujuan penelitian, terdapat beberapa aspek yang dapat disempurnakan. Oleh karena itu, saran yang dapat diberikan untuk penelitian dan pengembangan selanjutnya perlu melakukan analisis konsumsi daya (*power profiling*) secara langsung untuk mengukur dampak konsumsi baterai (dalam satuan *miliampere* atau *Joule*) pada perangkat *end-node* akibat beban penambahan siklus komputasi kriptografi, sehingga masa pakai (*lifetime*) baterai untuk implementasi *Internet of Things* (IoT) jangka panjang dapat dipastikan secara optimal.

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta





DAFTAR PUSTAKA

- Artha Lesmana, D. D., Arsa Suyadnya, I. M., & Shandyasa, I. W. (2023). Rancang Bangun Perangkat Keras Sistem Smart Lampu Penerangan Jalan Umum Berbasis Internet of Things Guna Mendukung Implementasi Smart City. *Jurnal SPEKTRUM*, 10(3), 21. <https://doi.org/10.24843/spektrum.2023.v10.i03.p3>
- Components101. (2018). *DHT22 – Temperature and Humidity Sensor*. <https://components101.com/sensors/dht22-pinout-specs-datasheet>
- Designer, C. (n.d.). *How to Use LoRa Ra-02 SX1278: Examples, Pinouts, and Specs*. Retrieved June 7, 2026, from <https://docs.cirkitdesigner.com/component/ea67bc39-2543-48d1-915c-5686eed5bac4/lora-ra-02-sx1278>
- Guimarães, A. (2024). *Gateway IoT usando Raspberry Pi*.
- Hakeem, S. A. A., El-Kader, S. M. A., & Kim, H. W. (2021). A Key Management Protocol Based on the Hash Chain Key Generation for Securing LoRaWAN Networks. *Sensors (Basel, Switzerland)*, 21(17), 1–35. <https://doi.org/10.3390/s21175838>
- Hayati, N. (2024). The Design of Security Framework for LoRaWAN FUOTA. *Journal of Electrical Technology UMY*, 7(2), 82–88. <https://doi.org/10.18196/jet.v7i2.22360>
- Instiperjogja, R. (2021). *Mikrokontroler ESP32 (IoT Device)*. <https://robotics.instiperjogja.ac.id/post/ESP32>
- Kreaweb. (2023). *LARAGON, an easy and yet powerful Local Web Development Server*. <https://www.kreaweb.be/laragon/>
- Lestari, R. I., Suryani, V., & Wardana, A. A. (2020). Implementasi Pengamanan Pada Jaringan LoRaWAN Untuk Mengatasi Serangan Sniffing Dengan Menggunakan Metode Digital Signature. *EProceedings ...*, 7(2), 7983–7994. <https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/13079>
- Li, C., & Cao, Z. (2022). LoRa Networking Techniques for Large-scale and Long-term IoT: A Down-to-top Survey. *ACM Computing Surveys*, 55(3). <https://doi.org/10.1145/3494673>

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

- Lv, Z. (2021). Security of Internet of Things edge devices. *Software - Practice and Experience*, 51(12), 2446–2456. <https://doi.org/10.1002/spe.2806>
- McWeeney, B., Mudritskiy, I., & Verbruggen, R. (2024). Analysis of Payload Confidentiality for the IoT/ LPWAN Technology ‘Lora.’ *International Conference on Information Systems Security and Privacy, I(Icissp)*, 91–102. <https://doi.org/10.5220/0012271400003648>
- Ningsih, K. S., Aruan, N. J., & Siahaan, A. T. A. A. (2022). Aplikasi Buku Tamu Menggunakan Fitur Kamera Dan Ajax Berbasis Website Pada Kantor Dispora Kota Medan. *SITek: Jurnal Sains, Informatika, Dan Teknologi*, 1, 94–99.
- Olazabal, A. A., Kaur, J., & Yeboah-Ofori, A. (2022). Deploying Man-In-the-Middle Attack on IoT Devices Connected to Long Range Wide Area Networks (LoRaWAN). *ISC2 2022 - 8th IEEE International Smart Cities Conference*. <https://doi.org/10.1109/ISC255366.2022.9922377>
- Palcomtech. (2024). *Laragon Sebuah Platform Pengembangan Lokal Terkini untuk Pengembang Web*. <https://palcomtech.ac.id/laragon-sebuah-platform-pengembangan-lokal-terkini-untuk-pengembang-web/>
- PlatformIO. (n.d.). *PlatformIO*. Retrieved June 2, 2026, from <https://platformio.org/>
- Puspasari, F., Satya, T. P., Oktawati, U. Y., Fahrurrozi, I., & Prisyanti, H. (2020). Analisis Akurasi Sistem sensor DHT22 berbasis Arduino terhadap Thermohygrometer Standar. *Jurnal Fisika Dan Aplikasinya*, 16(1), 40. <https://doi.org/10.12962/j24604682.v16i1.5776>
- Radhakrishnan, I., Jadon, S., & Honnavalli, P. B. (2024). Efficiency and Security Evaluation of Lightweight Cryptographic Algorithms for Resource-Constrained IoT Devices. *Sensors*, 24(12). <https://doi.org/10.3390/s24124008>
- Ramadhani, E. H., Wulandari, A., & Hikmaturokhman, A. (2025). Perancangan Sistem Keamanan Komunikasi Data pada Jaringan LoRA Menggunakan Algoritme PRESENT. *Jurnal Telematika*, 19(2), 72–81. <https://journal.ithb.ac.id/index.php/telematika/article/view/674>
- Rosyida Zain, A., Hudi, S. A. R., & Neforawati, I. (2022). Analisis Pengiriman Data dari LoRa Gateway ke Network Server. *Multinetics*, 7(1), 21–29. <https://doi.org/10.32722/multinetics.v7i1.3971>



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

- Semtech. (n.d.). *Marks and Logos*. Retrieved June 7, 2026, from <https://www.semtech.com/company/brand-resources>
- Setiowati, S., Riandini, R., Sari, V. A., Purwanti, I. L., & Andriansyah, N. (2023). LoRa Communication in the Service Level Monitoring Satu Duit Bogor Bridge. *JITCE (Journal of Information Technology and Computer Engineering)*, 7(01), 19–28. <https://doi.org/10.25077/jitce.7.01.19-28.2023>
- Shkoropad, Y., & Beshley, H. (2025). Development of Embedded Software for Esp32-Based Lora Modules With Adaptive Configuration and Link Quality Monitoring. *Information and Communication Technologies, Electronic Engineering*, 5(2), 25–37. <https://doi.org/10.23939/ictee2025.02.025>
- Simões, T., Cruz, T., Sousa, B., & Simões, P. (2025). A Security-Conscious Primer on LoRa and LoRaWAN Technologies. *European Conference on Information Warfare and Security, ECCWS*, 638–646. <https://doi.org/10.34190/eccws.24.1.3575>
- Sun, C., Zheng, F., Zhou, G., & Guo, K. (2020). Design and Implementation of Cloud-based Single-channel LoRa IIoT Gateway Using Raspberry Pi. *Chinese Control Conference, CCC, 2020-July*, 5259–5263. <https://doi.org/10.23919/CCC50068.2020.9189480>
- Sun, Z., Yang, H., Liu, K., Yin, Z., Li, Z., & Xu, W. (2022). Recent Advances in LoRa: A Comprehensive Survey. *ACM Transactions on Sensor Networks*, 18(4). <https://doi.org/10.1145/3543856>
- Tech, F. (2025). *AMS1117 Voltage Regulator IC: Pinout, Features & Applications*. <https://www.flywing-tech.com/blog/ams1117-voltage-regulator-ic-pinout-features-applications/>
- Wikipedia. (n.d.-a). *MySQL*. Retrieved June 1, 2026, from <https://id.wikipedia.org/wiki/MySQL>
- Wikipedia. (n.d.-b). *VS Code*. Retrieved June 3, 2026, from https://en.wikipedia.org/wiki/Visual_Studio_Code
- Windya, P. A., Suryani, V., & Wardana, A. A. (2021). Penerapan keamanan komunikasi pada jaringan LoRa(long range) menggunakan algoritma advanced encryption standard(AES) dan message authentication code(MAC). *E-Proceeding of Engineering*, 8(2), 3406.



© Hak Cipta milik Politeknik Negeri Jakarta

Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Zaraket, C., Papageorgas, P., Aillerie, M., Agavanakis, K., & Salame, C. (2020). Cyber security vulnerabilities of smart metering based on LPWAN wireless communication technologies. *AIP Conference Proceedings*, 2307. <https://doi.org/10.1063/5.0032709>

Zhang, J. C., Khieng, D. H. T., & Huang, N. F. (2022). A Long Range Reliable Precise Irrigation System Based on LoRaP2P Protocol. *International Conference on Information Networking*, 2022-Janua, 107–112. <https://doi.org/10.1109/ICOIN53446.2022.9687187>



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumunkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

DAFTAR RIWAYAT HIDUP PENULIS



Mohammad Rizki Fadillah

Lahir pada 2 Oktober 2004, saat ini berdomisili di Jakarta Selatan. Putra pertama dari dua bersaudara. Lulus dari Sekolah Dasar (SD) tahun 2016, lulus dari Sekolah Menengah Pertama (SMP) tahun 2019, dan dari Sekolah Menengah Kejuruan (SMK) dengan kompetensi keahlian Teknik Komputer dan Jaringan pada tahun 2022.

Berkesempatan untuk melanjutkan pendidikan Diploma IV di Program Studi Teknik Multimedia dan Jaringan, Jurusan Teknik Informatika dan Komputer, Politeknik Negeri Jakarta dari tahun 2022 hingga tahun 2026. Selama masa perkuliahan, Penulis memiliki ketertarikan yang mendalam dan sangat berminat untuk mendalami bidang Network Engineering, khususnya pada implementasi infrastruktur jaringan, konfigurasi perangkat multi-vendor, penyusunan arsitektur routing, serta analisis sistem keamanan jaringan nirkabel.

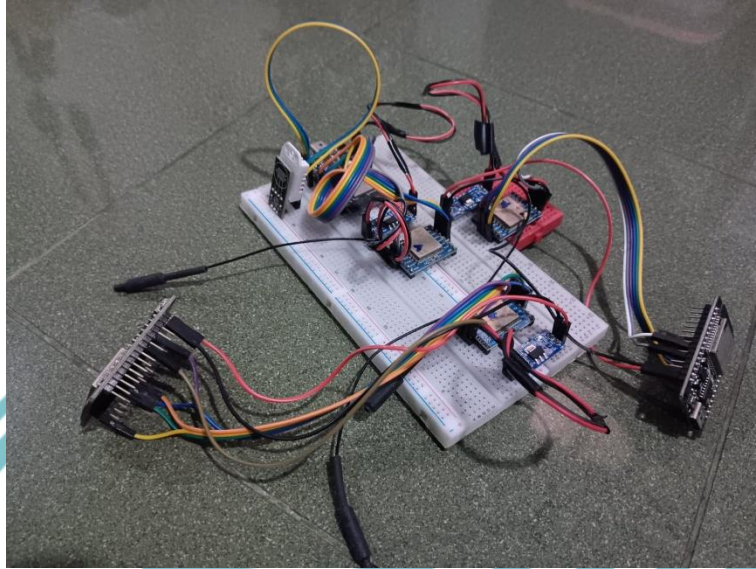
**POLITEKNIK
NEGERI
JAKARTA**

Hak Cipta :

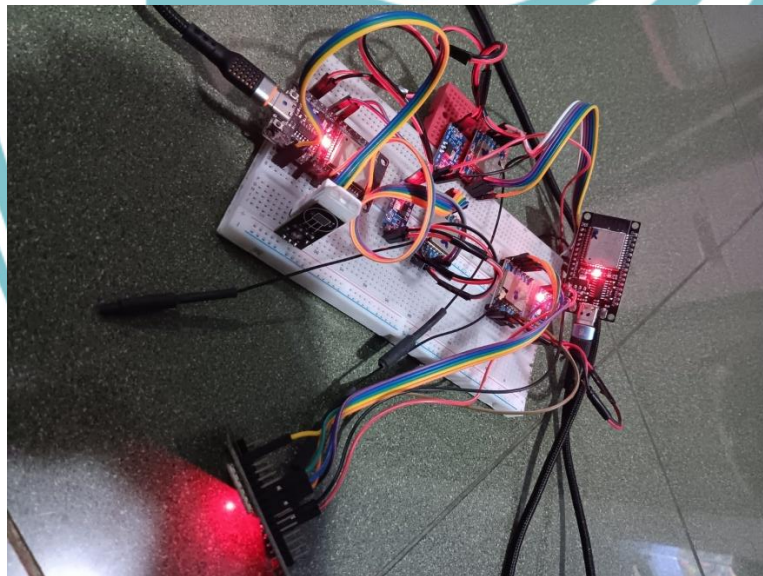
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

LAMPIRAN

Lampiran 1. Dokumentasi Perangkat Keras



Lampiran 2. Dokumentasi Implementasi Sistem



1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

The image shows a Windows command prompt window with a C++ program for a Denial of Service (DoS) attack. The program is named `dos.cpp` and is located in the `C:\Users\VL61\1\platforms\user\scripts\platforms\os\device monitor -port cmd` directory. The program defines several constants: `IP` (192.168.1.100), `PORT` (8080), `PACKET_SIZE` (1024), and `ATTACK_RATE` (1000). It then enters a loop to send raw packets to the target IP address.

```

1  #include <iostream>
2  #include <string>
3  #include <sys/types.h>
4  #include <unistd.h>
5  #include <arpa/inet.h>
6  #include <netinet/ip.h>
7  #include <netinet/ip_icmp.h>
8  #include <netinet/tcp.h>
9  #include <sys/socket.h>
10 #include <netdb.h>
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <string.h>
14 #include <unistd.h>
15 #include <sys/types.h>
16 #include <arpa/inet.h>
17 #include <netinet/ip.h>
18 #include <netinet/ip_icmp.h>
19 #include <netinet/tcp.h>
20 #include <sys/socket.h>
21 #include <netdb.h>
22 #include <stdio.h>
23 #include <stdlib.h>
24 #include <string.h>
25 #include <unistd.h>
26 #include <sys/types.h>
27 #include <arpa/inet.h>
28 #include <netinet/ip.h>
29 #include <netinet/ip_icmp.h>
30 #include <netinet/tcp.h>
31 #include <sys/socket.h>
32 #include <netdb.h>
33 #include <stdio.h>
34 #include <stdlib.h>
35 #include <string.h>
36 #include <unistd.h>
37 #include <sys/types.h>
38 #include <arpa/inet.h>
39 #include <netinet/ip.h>
40 #include <netinet/ip_icmp.h>
41 #include <netinet/tcp.h>
42 #include <sys/socket.h>
43 #include <netdb.h>
44 #include <stdio.h>
45 #include <stdlib.h>
46 #include <string.h>
47 #include <unistd.h>
48 #include <sys/types.h>
49 #include <arpa/inet.h>
50 #include <netinet/ip.h>
51 #include <netinet/ip_icmp.h>
52 #include <netinet/tcp.h>
53 #include <sys/socket.h>
54 #include <netdb.h>
55 #include <stdio.h>
56 #include <stdlib.h>
57 #include <string.h>
58 #include <unistd.h>
59 #include <sys/types.h>
60 #include <arpa/inet.h>
61 #include <netinet/ip.h>
62 #include <netinet/ip_icmp.h>
63 #include <netinet/tcp.h>
64 #include <sys/socket.h>
65 #include <netdb.h>
66 #include <stdio.h>
67 #include <stdlib.h>
68 #include <string.h>
69 #include <unistd.h>
70 #include <sys/types.h>
71 #include <arpa/inet.h>
72 #include <netinet/ip.h>
73 #include <netinet/ip_icmp.h>
74 #include <netinet/tcp.h>
75 #include <sys/socket.h>
76 #include <netdb.h>
77 #include <stdio.h>
78 #include <stdlib.h>
79 #include <string.h>
80 #include <unistd.h>
81 #include <sys/types.h>
82 #include <arpa/inet.h>
83 #include <netinet/ip.h>
84 #include <netinet/ip_icmp.h>
85 #include <netinet/tcp.h>
86 #include <sys/socket.h>
87 #include <netdb.h>
88 #include <stdio.h>
89 #include <stdlib.h>
90 #include <string.h>
91 #include <unistd.h>
92 #include <sys/types.h>
93 #include <arpa/inet.h>
94 #include <netinet/ip.h>
95 #include <netinet/ip_icmp.h>
96 #include <netinet/tcp.h>
97 #include <sys/socket.h>
98 #include <netdb.h>
99 #include <stdio.h>
100 #include <stdlib.h>
101 #include <string.h>
102 #include <unistd.h>
103 #include <sys/types.h>
104 #include <arpa/inet.h>
105 #include <netinet/ip.h>
106 #include <netinet/ip_icmp.h>
107 #include <netinet/tcp.h>
108 #include <sys/socket.h>
109 #include <netdb.h>
110 #include <stdio.h>
111 #include <stdlib.h>
112 #include <string.h>
113 #include <unistd.h>
114 #include <sys/types.h>
115 #include <arpa/inet.h>
116 #include <netinet/ip.h>
117 #include <netinet/ip_icmp.h>
118 #include <netinet/tcp.h>
119 #include <sys/socket.h>
120 #include <netdb.h>
121 #include <stdio.h>
122 #include <stdlib.h>
123 #include <string.h>
124 #include <unistd.h>
125 #include <sys/types.h>
126 #include <arpa/inet.h>
127 #include <netinet/ip.h>
128 #include <netinet/ip_icmp.h>
129 #include <netinet/tcp.h>
130 #include <sys/socket.h>
131 #include <netdb.h>
132 #include <stdio.h>
133 #include <stdlib.h>
134 #include <string.h>
135 #include <unistd.h>
136 #include <sys/types.h>
137 #include <arpa/inet.h>
138 #include <netinet/ip.h>
139 #include <netinet/ip_icmp.h>
140 #include <netinet/tcp.h>
141 #include <sys/socket.h>
142 #include <netdb.h>
143 #include <stdio.h>
144 #include <stdlib.h>
145 #include <string.h>
146 #include <unistd.h>
147 #include <sys/types.h>
148 #include <arpa/inet.h>
149 #include <netinet/ip.h>
150 #include <netinet/ip_icmp.h>
151 #include <netinet/tcp.h>
152 #include <sys/socket.h>
153 #include <netdb.h>
154 #include <stdio.h>
155 #include <stdlib.h>
156 #include <string.h>
157 #include <unistd.h>
158 #include <sys/types.h>
159 #include <arpa/inet.h>
160 #include <netinet/ip.h>
161 #include <netinet/ip_icmp.h>
162 #include <netinet/tcp.h>
163 #include <sys/socket.h>
164 #include <netdb.h>
165 #include <stdio.h>
166 #include <stdlib.h>
167 #include <string.h>
168 #include <unistd.h>
169 #include <sys/types.h>
170 #include <arpa/inet.h>
171 #include <netinet/ip.h>
172 #include <netinet/ip_icmp.h>
173 #include <netinet/tcp.h>
174 #include <sys/socket.h>
175 #include <netdb.h>
176 #include <stdio.h>
177 #include <stdlib.h>
178 #include <string.h>
179 #include <unistd.h>
180 #include <sys/types.h>
181 #include <arpa/inet.h>
182 #include <netinet/ip.h>
183 #include <netinet/ip_icmp.h>
184 #include <netinet/tcp.h>
185 #include <sys/socket.h>
186 #include <netdb.h>
187 #include <stdio.h>
188 #include <stdlib.h>
189 #include <string.h>
190 #include <unistd.h>
191 #include <sys/types.h>
192 #include <arpa/inet.h>
193 #include <netinet/ip.h>
194 #include <netinet/ip_icmp.h>
195 #include <netinet/tcp.h>
196 #include <sys/socket.h>
197 #include <netdb.h>
198 #include <stdio.h>
199 #include <stdlib.h>
200 #include <string.h>
201 #include <unistd.h>
202 #include <sys/types.h>
203 #include <arpa/inet.h>
204 #include <netinet/ip.h>
205 #include <netinet/ip_icmp.h>
206 #include <netinet/tcp.h>
207 #include <sys/socket.h>
208 #include <netdb.h>
209 #include <stdio.h>
210 #include <stdlib.h>
211 #include <string.h>
212 #include <unistd.h>
213 #include <sys/types.h>
214 #include <arpa/inet.h>
215 #include <netinet/ip.h>
216 #include <netinet/ip_icmp.h>
217 #include <netinet/tcp.h>
218 #include <sys/socket.h>
219 #include <netdb.h>
220 #include <stdio.h>
221 #include <stdlib.h>
222 #include <string.h>
223 #include <unistd.h>
224 #include <sys/types.h>
225 #include <arpa/inet.h>
226 #include <netinet/ip.h>
227 #include <netinet/ip_icmp.h>
228 #include <netinet/tcp.h>
229 #include <sys/socket.h>
230 #include <netdb.h>
231 #include <stdio.h>
232 #include <stdlib.h>
233 #include <string.h>
234 #include <unistd.h>
235 #include <sys/types.h>
236 #include <arpa/inet.h>
237 #include <netinet/ip.h>
238 #include <netinet/ip_icmp.h>
239 #include <netinet/tcp.h>
240 #include <sys/socket.h>
241 #include <netdb.h>
242 #include <stdio.h>
243 #include <stdlib.h>
244 #include <string.h>
245 #include <unistd.h>
246 #include <sys/types.h>
247 #include <arpa/inet.h>
248 #include <netinet/ip.h>
249 #include <netinet/ip_icmp.h>
250 #include <netinet/tcp.h>
251 #include <sys/socket.h>
252 #include <netdb.h>
253 #include <stdio.h>
254 #include <stdlib.h>
255 #include <string.h>
256 #include <unistd.h>
257 #include <sys/types.h>
258 #include <arpa/inet.h>
259 #include <netinet/ip.h>
260 #include <netinet/ip_icmp.h>
261 #include <netinet/tcp.h>
262 #include <sys/socket.h>
263 #include <netdb.h>
264 #include <stdio.h>
265 #include <stdlib.h>
266 #include <string.h>
267 #include <unistd.h>
268 #include <sys/types.h>
269 #include <arpa/inet.h>
270 #include <netinet/ip.h>
271 #include <netinet/ip_icmp.h>
272 #include <netinet/tcp.h>
273 #include <sys/socket.h>
274 #include <netdb.h>
275 #include <stdio.h>
276 #include <stdlib.h>
277 #include <string.h>
278 #include <unistd.h>
279 #include <sys/types.h>
280 #include <arpa/inet.h>
281 #include <netinet/ip.h>
282 #include <netinet/ip_icmp.h>
283 #include <netinet/tcp.h>
284 #include <sys/socket.h>
285 #include <netdb.h&gt
```

Pin diagram of the ESP32 showing pin numbers 1-40 and their functions. The diagram is a square with pins numbered 1 to 40 around the perimeter. Functions are listed next to each pin number. The center of the chip is labeled 'ESP32'.

Pin	Function
1	32K_XN
2	GPIO15
3	GPIO16
4	GPIO17
5	GPIO18
6	GPIO19
7	GPIO20
8	GPIO21
9	GPIO22
10	GPIO23
11	GPIO24
12	GPIO25
13	GPIO26
14	GPIO27
15	GPIO28
16	GPIO29
17	GPIO30
18	GPIO31
19	GPIO32
20	GPIO33
21	GPIO34
22	GPIO35
23	GPIO36
24	GPIO37
25	GPIO38
26	GPIO39
27	GPIO40
28	GPIO41
29	GPIO42
30	GPIO43
31	GPIO44
32	GPIO45
33	GPIO46
34	GPIO47
35	GPIO48
36	GPIO49
37	GPIO50
38	GPIO51
39	GPIO52
40	GPIO53

ESP32 is equipped with hardware accelerators of general algorithms, such as AES (FIPS PUB 197), SHA (FIPS PUB 180-4), and RSA. The chip also supports independent arithmetic, such as large-number modular multiplication and large-number multiplication. The maximum operation length for RSA, large-number modular multiplication, and large-number multiplication is 4096 bits.

Jurusan Teknik Informatika dan Komputer – Politeknik Negeri Jakarta



Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Lampiran 5. Spesifikasi Teknis LoRa Ra-02

7SEMI
SX1278 LoRa Ra-02

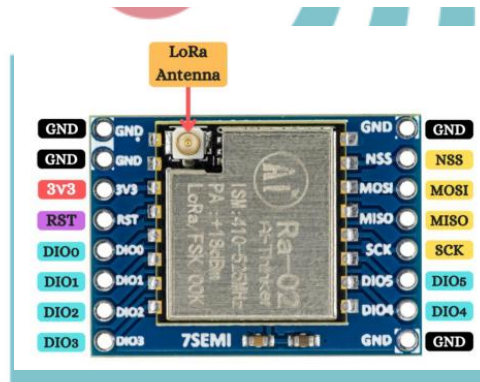
Manual
Version 1.0

2. Technical Specification

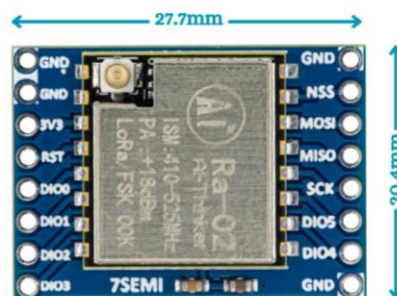
The **Technical Specification** table provides detailed information about 7Semi SX1278 **LoRa Ra-02**, including its operating voltage, current consumption, and electrical characteristics. This data helps users understand the power requirements, communication parameters, and performance capabilities of the sensor. It ensures compatibility with different microcontrollers and embedded systems while providing guidelines for efficient integration into various applications.

SX1278 Specifications

- Chipset: Semtech SX1278
- Frequency Range: 410 MHz – 525 MHz (Typically 433 MHz)
- Transmission Distance: Up to 10 km (in open areas with ideal conditions)
- Operating Voltage: 1.8V – 3.3V (Typically 3.3V)
- Operating Temperature: -40°C to +85°C
- Antenna Interface: IPEX (U.FL) Connector
- Sensitivity: Up to -148 dBm
- Module Breakout Size: 27.7mm x 20.4mm.
- Weight:



Pin	Name	Description
1	GND	Ground connection
2	3V3	Power supply input (3.3V)
3	RST	Module Reset
4	DIO0	Digital I/O for interrupts
5	DIO1	Digital I/O for interrupts
6	DIO2	Digital I/O for interrupts
7	DIO3	Digital I/O for interrupts
8	DIO4	Digital I/O for interrupts
9	DIO5	Digital I/O for interrupts
10	SCK	SPI Clock
11	MISO	SPI Data Output
12	MOSI	SPI Data Input
13	NSS	SPI Chip Select

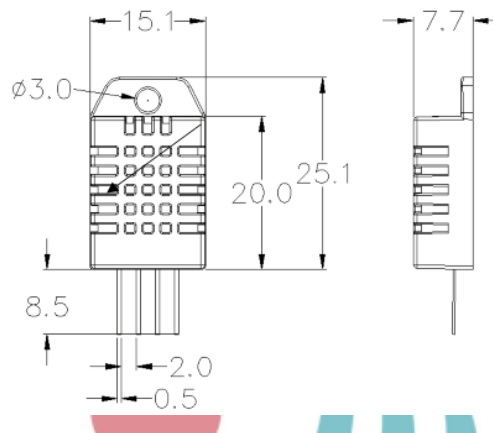




Hak Cipta :

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber :
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penulisan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Politeknik Negeri Jakarta
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Politeknik Negeri Jakarta

Lampiran 6. Spesifikasi Teknis DHT22 Aosong Electronics Co.,Ltd



Pin sequence number: 1 2 3 4 (from left to right direction).

Pin	Function
1	VDD—power supply
2	DATA—signal
3	NULL
4	GND

Lampiran 7. Tautan Repositori Kode Program (*Source Code*)

Seluruh kode program implementasi algoritma AES-128 Mode CTR, HMAC-SHA256, serta mekanisme *Dynamic Key Management* (KDF) menggunakan pustaka mbedTLS pada perangkat ESP32 diunggah pada platform GitHub. Kode program untuk Node Pengirim dan Node Penerima tersebut dapat diakses secara publik melalui tautan berikut: <https://github.com/rizkifrtma11/TA-LoRa-P2P-Security/tree/main>

Lampiran 8. Tautan Akses Dataset Pengujian

Data hasil pengujian disimpan pada layanan *cloud storage* Google Drive. Folder ini memuat data mentah (*raw data log*) hasil pengukuran performa komputasi (*processing time*), nilai RSSI, SNR, serta pencatatan error (*packet drop* atau *HMAC invalid*) selama pengujian mitigasi serangan *replay* dan *tampering*. Seluruh dataset dapat diakses melalui tautan berikut: <https://s.pnj.ac.id/DatasetHasilPengujian>.