# END TO END ENKRIPSI MENGGUNAKAN ADVANCED ENCRYPTION STANDARD PADA PERANGKAT INTERNET OF THINGS

END TO END ENCRYPTION USING ADVANCED ENCRYPTION STANDARD ON INTERNET OF THINGS DEVICES

**Noprianto[1)], Vivi Nur Wijayaningrum[2)]**
E-mail : [1)]noprianto@polinema.ac.id , [2)]vivinurw@polinema.ac.id

[1,2)] Jurusan Teknologi Informasi, Politeknik Negeri Malang

**Abstrak**

Penerapan teknologi Internet of Things (IoT) di berbagai bidang di dalam kehidupan sehari-hari menyebabkan diperlukannya sebuah teknik pengamanan data untuk mencegah terjadinya tindak kejahatan yang dapat merugikan pengguna. Penggunaan algoritma kriptografi Advanced Encryption Standard (AES) dapat dimanfaatkan untuk mengamankan data yang tersimpan pada perangkat IoT sehingga data tersebut tetap dalam keadaan aman pada saat proses pengiriman dilakukan dengan perangkat IoT lainnya. Penerapan AES dilakukan pada sistem IoT dengan menguji mekanisme monitoring dan controlling untuk memastikan keamanan data. Hasil pengujian menunjukkan bahwa penggunaan AES mode Cipher Block Chaining (CBC) dapat memberikan pengamanan data saat melakukan proses pengiriman data dari perangkat IoT sampai dengan aplikasi dashboard monitoring.
**Kata kunci:** *dekripsi, enkripsi, internet of things, keamanan, perangkat*

**Abstract**

*The application of Internet of Things (IoT) technology in various fields in everyday life causes the need for a data security technique to prevent crimes that can harm users. The use of the Advanced Encryption Standard (AES) cryptographic algorithm can be used to secure data stored on IoT devices so that the data remains safe when the delivery process is carried out with other IoT devices. The application of AES is carried out on the IoT system by testing monitoring and controlling mechanisms to ensure data security. The test results show that the use of AES Cipher Block Chaining (CBC) mode can provide data security when carrying out the process of sending data from IoT devices to dashboard monitoring applications.*
**Keywords:** *decryption, encryption, internet of things, security, device*

## 1. PENDAHULUAN

Perkembangan teknologi Internet of Things (IoT) semakin meningkat selama beberapa tahun terakhir. Penerapan IoT dapat berupa kendaraan pintar (*smart vehicles*), bangunan pintar (*smart buildings*), sistem pemantauan kesehatan, rantai pasokan makanan, dan sebagainya [1]. Dengan demikian, sistem berbasis IoT harus mampu menangani data dalam jumlah yang cukup besar. Hal ini yang menyebabkan diperlukannya algoritma yang cukup efisien untuk melakukan pemrosesan dan analisis data dengan jumlah yang cukup besar tersebut [2][3].

Sistem berbasis IoT sering menjadi target penyerangan bagi pihak ketiga yang tidak bertanggung jawab. Sebagian besar penyerangan dilakukan untuk mencuri informasi mengenai lokasi, keuangan, catatan terkait kesehatan, atau informasi penting lainnya. Oleh karena itu, mekanisme pengamanan data perlu dilakukan pada sistem

berbasis IoT untuk meminimalkan terjadinya tindak kejahatan pencurian data. Salah satu teknik yang dapat digunakan untuk mengamankan data adalah menggunakan algoritma Advanced Encryption Standard (AES) untuk proses enkripsi dan dekripsi data. AES merupakan skema kriptografi menggunakan cipher simetris dengan tingkat keamanan yang cukup tinggi. Keamanan AES cukup kuat dan implementasinya sederhana, baik pada perangkat lunak maupun perangkat keras [4].

Beberapa penelitian mengenai pemanfaatan AES untuk mengamankan data pada perangkat IoT telah dilakukan oleh peneliti-peneliti sebelumnya. Endrayanto menerapkan algoritma AES-128 untuk digunakan pada modul IoT Particle Photon yang belum mempunyai hardware accelarator. Penelitian tersebut menunjukkan bahwa penggunaan AES-128 dapat berjalan dengan baik dengan waktu enkripsi terlama sebesar 398 ms [5]. Pada penelitian lainnya, AES 128 bit juga diterapkan untuk melakukan pengamanan data pada perangkat berbasis IoT untuk model bercocok tanam hidroponik [6].

Pada penelitian ini, algoritma AES digunakan untuk melakukan enkripsi dan dekripsi data pada sistem IoT. Algoritma AES dipilih karena kemampuannya yang sangat baik ketika digunakan pada sistem dengan sumber daya yang terbatas. Hal ini tentunya sangat sesuai untuk digunakan pada perangkat IoT dengan keterbatasan sumber daya.

## 2. METODOLOGI

Sistem yang diusulkan pada penelitian ini merupakan sebuah perangkat IoT lengkap yang terdiri dari sensor, aktuator, message broker, dan platform IoT (Node-RED). Gambar 1 merupakan ilustrasi perangkat IoT yang diusulkan.


**Gambar 1. Ilustrasi perangkat IoT**

Pada Gambar 1, terdapat tiga komponen utama yang digunakan yaitu perangkat IoT, Message Broker, dan Platform IoT (Node-RED). Komunikasi yang dilakukan antara perangkat IoT, Message broker, dan Node-RED menggunakan jaringan nirkable (Wifi).

Pada penelitian ini, perangkat IoT menggunakan NodeMCU yang sudah dilengkapi dengan sensor-sensor seperti Light Dependent Resistor (LDR), Ultrasonik, dan DHT11. Ketiga sensor tersebut masing-masing berfungsi untuk mendeteksi intensitas cahaya, mengetahui jarak objek, dan mengetahui kelembaban dan suhu di sekitar. Selain terdapat sensor, pada perangkat IoT juga terdapat beberapa aktuator seperti relay, buzzer, FAN (kipas), dan LED (lampu). Relay merupakan sebuah aktuator yang digunakan untuk menghubungkan dan memutuskan arus listrik, sementara buzzer digunakan untuk menghasilkan suara [7].

Data-data yang berasal dari sensor yang dihasilkan oleh perangkat IoT akan dikirimkan (*publish*) melalui jaringan Wifi ke sebuah message broker dengan menggunakan protokol Message Queuing Telemetry Transport (MQTT). Message broker merupakan sebuah sistem dan layanan yang digunakan untuk berkomunikasi dan bertukar informasi antara perangkat IoT dengan perangkat yang lain. Pada penelitian ini, protokol MQTT digunakan untuk pertukaran data antar perangkat IoT karena dianggap ringan pada saat proses pertukaran data [8] sehingga proses pertukaran data menjadi lebih cepat jika dibandingkan dengan protokol lainnya.

Pada saat melakukan proses pengiriman data atau pertukaran informasi, protokol MQTT menggunakan konsep *publish* dan *subscribe* dengan topik tertentu. *Publish* merupakan proses pengiriman data ke message broker, sedangkan *subscribe* merupakan proses yang dilakukan untuk menerima data ketika ada *client* atau perangkat yang melakukan *publish*. Untuk melakukan *publish* atau *subscribe*, harus digunakan topik yang bertujuan untuk membedakan data. Dengan demikian, semua perangkat IoT yang melakukan *subscribe* dengan topik yang sama akan mendapatkan data yang sama ketika terdapat perangkat lain yang melakukan *publish* dengan topik tersebut [9]. Pada penelitian ini, Mosquitto digunakan sebagai message broker. Data-data yang dikirimkan (*publish*) atau diterima (*subscribe*) yang berasal dari perangkat IoT ke Mosquitto atau sebaliknya, diamankan menggunakan algoritma Advanced Encryption Standard (AES).

Selanjutnya, data-data sensor atau aktuator disajikan dalam sebuah tampilan dashboard menggunakan platform IoT yaitu Node-RED. Platform IoT ini berfungsi untuk mengumpulkan, menyimpan, menyajikan, dan mengolah data dari perangkat IoT. Node-RED harus melakukan *subscribe* data-data sensor ataupun *publish* data aktuator dalam keadaan data telah terenkripsi menggunakan AES. Mosquitto dan Node-RED pada penelitian ini masih terpasang pada satu perangkat komputer, tetapi penerapannya dapat dipasang secara terpisah sehingga beban kerja tidak akan saling memberatkan.

### 2.1 NodeMCU

NodeMCU merupakan sebuah perangkat IoT yang bertugas untuk mengambil data di lingkungan sekitar dengan bantuan sensor. Sensor merupakan sebuah alat yang digunakan untuk mengubah besaran fisis menjadi besaran listrik. Besaran fisis bisa berupa panas, suara, intensitas cahaya, tekanan, kemiringan, magnetis, kelembaban, dan gravitasi. Setelah mendapatkan data-data tersebut, NodeMCU dapat mengendalikan keadaan lingkungan dengan bantuan aktuator. Aktuator merupakan alat yang digunakan untuk mengubah besaran listrik menjadi besaran fisis.

NodeMCU disebut sebagai single board microcontroller dengan chip ESP8266 processor Tensilica Xtensa LX106 berkapasitas 128Kb dan memiliki penyimpanan 4Mb. Koneksi pada NodeMCU memanfaatkan bluetooth dan koneksi Wifi sehingga dapat terhubung ke dalam jaringan. Sementara sumber tegangan yang digunakan pada NodeMCU diperoleh melalui USB. Dengan menggunakan spesifikasi komponen-komponen tersebut, pada NodeMCU dapat dipasang berbagai macam sensor dan aktuator sesuai dengan kebutuhan.

### 2.2 Mosquitto

Mosquitto merupakan sebuah message broker yang bersifat *open source* menggunakan protokol MQTT. Selain itu, Mosquitto bersifat ringan sehingga cocok digunakan pada semua perangkat mulai dari *single board computer* dengan sumber daya terbatas sampai dengan perangkat *full server*. Mosquitto ini merupakan bagian dari Eclipse Foundation yang disponsori oleh cedalo.com.

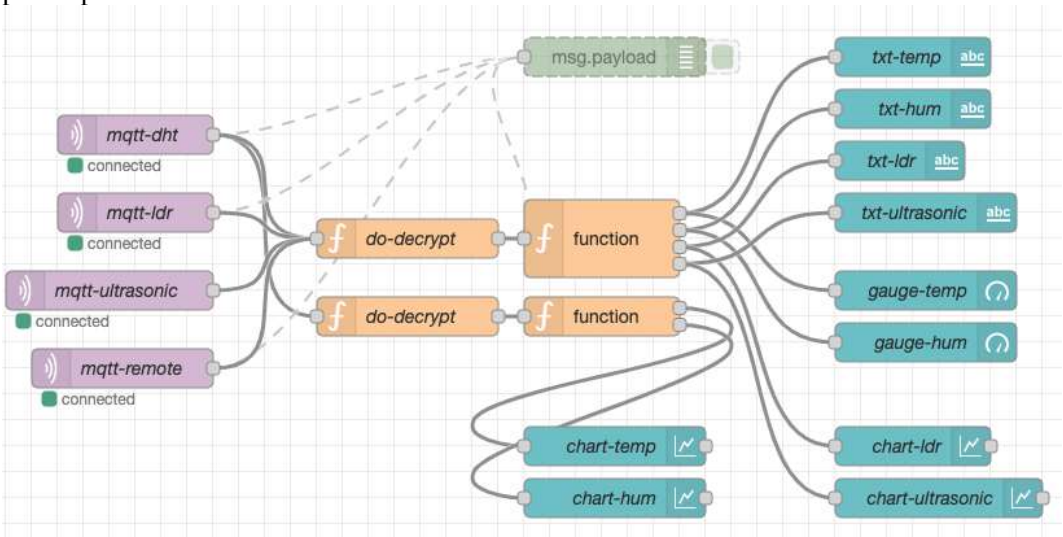
Protokol MQTT yang digunakan pada Mosquitto menyediakan sebuah fungsi yang ringan untuk melakukan pertukaran data dengan model *publish* dan *subscribe*. Hal ini sangat cocok digunakan untuk *messaging* pada IoT seperti sensor dengan *low energy* atau perangkat mobile seperti *handphone, embedded computer* atau mikrocontroller. Pada saat melakukan pengiriman data dari client (perangkat IoT) ke message broker (Mosquitto) digunakan proses *publish*, sementara untuk menerima data dari client (perangkat IoT) yang lain digunakan proses *subscribe*. Kedua proses tersebut dapat dilakukan dengan menggunakan topik tertentu sebagai identifikasi data yang unik. Misalnya terdapat sebuah client yang melakukan publish sensor cahaya dengan topik "/ldr", maka semua client yang melakukan subscribe dengan topik "/ldr" akan menerima data sensor cahaya tersebut.

### 2.3 Node-RED

Node-RED adalah salah satu platform IoT yang sangat terkenal karena kemudahannya saat digunakan. Dengan menggunakan konsep *drag and drop*, node yang

digunakan dapat disesuaikan dengan kebutuhan. Meskipun hampir 80% pemanfaatan Node-RED dilakukan menggunakan drag and drop, namun untuk melakukan fungsi yang lebih kompleks dibutuhkan kode program berbasis Javascript.

Sebuah node pada Node-RED menyatakan sebuah fungsi tertentu, misalnya menampilkan data atau mengolah data. Salah satu contoh node yang dapat digunakan adalah Gauge. Node tersebut digunakan untuk menampilkan data dalam bentuk gauge. Di dalam satu dashboard monitoring sistem, terdapat lebih dari satu node yang akan dihubungkan satu sama lain sehingga membentuk *flow*. Gambar 2 merupakan ilustrasi penerapan Node-RED.




**Gambar 2. Ilustrasi Penerapan Node-RED**

Pada Gambar 2, terdapat beberapa node-node yang dihubungkan satu sama lain. Node-node yang telah terhubung tersebut berfungsi untuk menampilkan data ke dalam sebuah halaman web. Hasil akhir dari Node-RED dapat dilihat melalui web browser.

## 2.4 Advanced Encryption Standard (AES)

AES merupakan algoritma enskripsi yang diperkenalkan untuk menggantikan algoritma enkripsi yang sudah ada sebelumnya yaitu Data Encryption Standard (DES). AES banyak digunakan karena kemampuan yang dimilikinya antara lain memberikan tingkat keamanan yang lebih akurat dan mudah diimplementasikan [10]. Selain itu, AES juga dianggap sebagai algoritma paling efektif dalam menyediakan keamanan pada saat transmisi pesan dibandingkan dengan algoritma lainnya seperti Rivest-Shamir-Adleman (RSA), DES, Triple Data Encryption Standard (3DES) [11][12].

Sebagai algoritma simetris, AES menggunakan key yang sama antara penerima dan pengirim pesan pada saat proses enkripsi dan dekripsi. Mekanisme algoritma AES dilakukan secara iteratif, yang dikenal dengan istilah putaran (round). Banyaknya putaran menyesuaikan dengan panjang key yang digunakan. Blok data 128 bit terlebih dahulu disusun dalam bentuk matriks array byte berukuran 4×4 dengan total 16 byte [13]. Selanjutnya, pada setiap putaran, dilakukan empat transformasi byte yang terdiri dari SubByte, ShiftRow, MixColumn, dan AddRoundKey [14][15]. Operasi SubByte merupakan proses mengubah setiap byte key dengan byte yang bersesuaian pada Substitution box (S-box). Operasi ShiftRow dilakukan untuk menggeser seluruh byte secara siklis ke kiri dengan nilai offset yang berbeda. MixColumn merupakan operasi penggantian nilai setiap byte pada setiap kolom dengan hasil perkalian menggunakan Galois Field. Operasi terakhir yaitu AddRoundKey merupakan operasi XOR antara matriks state dengan matriks round key.

## 3. HASIL DAN PEMBAHASAN

Pengujian yang dilakukan pada penelitian dilakukan secara *end to end*, yaitu dengan melakukan pengujian perangkat IoT (NodeMCU) yang sudah terpasang sensor dan aktuator sampai dengan data-data tersebut ditampilkan pada sebuah tampilan dashboard. Konektivitas dilakukan dengan menggunakan jaringan Wifi lokal area atau tanpa terhubung ke dalam internet karena berfokus pada pengamanan data menggunakan AES. Message broker dan Node-RED dipasang pada sebuah laptop sehingga keduanya memiliki alamat yang sama yaitu Localhost (127.0.0.1).

### 3.1 Proses Enkripsi dan Dekripsi Data NodeMCU

Pada NodeMCU, terdapat proses enkripsi dan deskripsi yang bertujuan untuk mengamankan data yang digunakan selama proses pertukaran informasi. Proses enkripsi merupakan proses untuk mengubah data sensor berbentuk plainteks yang dapat dibaca menjadi data terenkripsi yang tidak dapat dibaca. Tujuannya adalah agar data sensor yang akan di-*publish* ke message broker menjadi lebih aman pada saat diakses oleh pihak ketiga yang tidak bertanggung jawab, sehingga data tersebut sampai di tujuan dan dapat diterima oleh *subscriber*. Sebaliknya, proses dekripsi berfungsi untuk mengubah data yang terenkripsi menjadi data plainteks. Data tersebut dikirimkan oleh *publisher*, dalam hal ini adalah Node-RED, untuk mengontrol aktuator pada NodeMCU. Gambar 3 menunjukkan NodeMCU yang sudah berjalan.


**Gambar 3. NodeMCU**

Pada Gambar 3 tersebut, setiap kali melakukan *publish* data sensor ke message broker dan subscribe data untuk menerima perintah dari Node-RED, misalnya menggunakan data sensor cahaya dengan nilai 137, maka perintah yang digunakan adalah sebagai berikut:

```
{"iv":"f92kjvqlK0mP3gje+htWug==","msg":"iMDWmDeMrcKG0x7hwhbkQA=="}
```

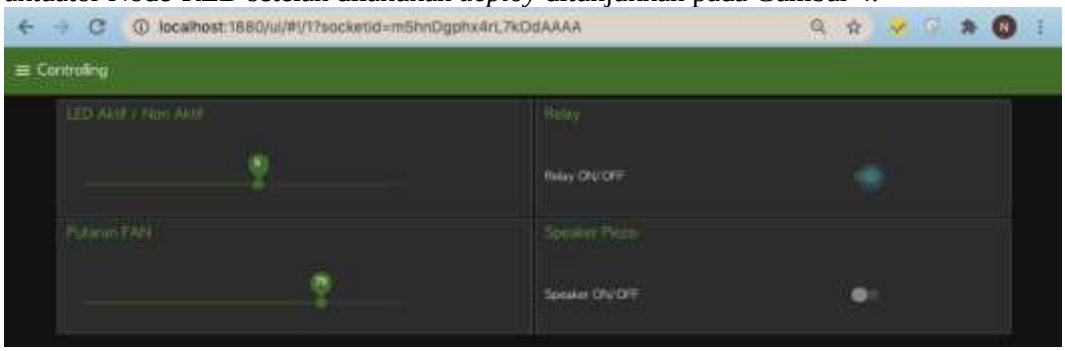
NodeMCU melaukan *publish* data sensor cahaya ke message broker menggunakan topik "/ldr". Penamaan topik dapat disesuaikan dengan kebutuhan, namun sebaiknya topik yang digunakan dapat mendeskripsikan data yang akan dikirimkan dan cara penulisannya seperti pengalamatan direktori yaitu menggunakan tanda garis miring (slash). Data dikirimkan dalam format JavaScript Object Notation (JSON) karena dianggap ringan pada saat proses pertukaran data, serta mudah untuk dibaca ataupun ditulis.

Pada perintah *publish* tersebut, terdapat dua komponen yaitu *iv* dan *msg* yang masing-masing sudah di-*encode* menggunakan Base64 setelah dilakukan enkripsi. Hal ini bertujuan agar format data menjadi ASCII, bukan hexadesimal. Initialization Vector (*iv*) adalah sebuah blok *dummy* yang digunakan untuk melakukan proses skripsi. Blok *dummy* tersebut dibangkitkan secara acak setiap kali melakukan proses enkripsi. Pada saat melakukan *publish*, *iv* tersebut akan dikirimkan karena dapat dimanfaatkan untuk proses

dekripsi. Komponen kedua yaitu *msg*, berisi data enkripsi atau nilai sensor yaitu 137. Proses enkripsi dilakukan menggunakan AES dengan panjang key 128 bit dan mode Cipher Block Chaining (CBC). Mode ini dilakukan dengan melakukan operasi XOR pada setiap blok plainteks dengan blok ciperteks (teks terenkripsi) hasil enkripsi sebelumnya, kemudian dilanjutkan dengan proses enkripsi. Dengan demikian, setiap cipherteks dari masing-masing blok akan bergantung pada hasil cipherteks dari blok-blok sebelumnya [16]. Pada proses ini, dibutuhkan Initialization Vector pada blok pertama untuk operasi XOR. Sementara key yang digunakan untuk melakukan enkripsi pada nilai sensor tersebut adalah 2B7E151628AED2A6ABF7158809CF4F3C.

### 3.2 Proses Enkripsi dan Dekripsi Data Node-RED

Sama halnya dengan NodeMCU, proses enkripsi dan dekripsi juga terdapat pada Node-RED. Proses enkripsi dilakukan ketika Node-RED melakukan kontrol aktuator dan melakukan *publish* sesuai dengan topik tertentu. Pada Node-RED, terdapat proses enkripsi pada aktuator antara lain untuk menyalakan LED 1-9, menyalakan putaran FAN, mengaktifkan RELAY, dan membunyikan BUZZER. Tampilan untuk mengontrol aktuator Node-RED setelah dilakukan *deploy* ditunjukkan pada Gambar 4.




**Gambar 4. Tampilan Controlling Aktuator Node-RED**

Pada Gambar 4, terlihat bahwa untuk melakukan kontrol, slider dapat digeser ke kanan ataupun ke kiri. Sebagai contoh, pada slider LED Aktif / Nonaktif, apabila slider digeser sampai menampilkan angka 5, maka LED yang terdapat pada controller akan menyala 5 buah seperti ditunjukkan pada Gambar 5.


Gambar 5. Tampilan Controlling LED

Ketika pada node-RED dilakukan penggeseran posisi pada slider, maka nilai slider akan di-*publish* ke message broker (Mosquitto) menggunakan topik (ledanim) yang berarti pada controller akan dilakukan *subscribe* dengan topik yang sama sehingga dapat dilakukan kontrol untuk menghidupkan 5 LED. Data yang di-*publish* tersebut dienkripsi sebagai berikut:

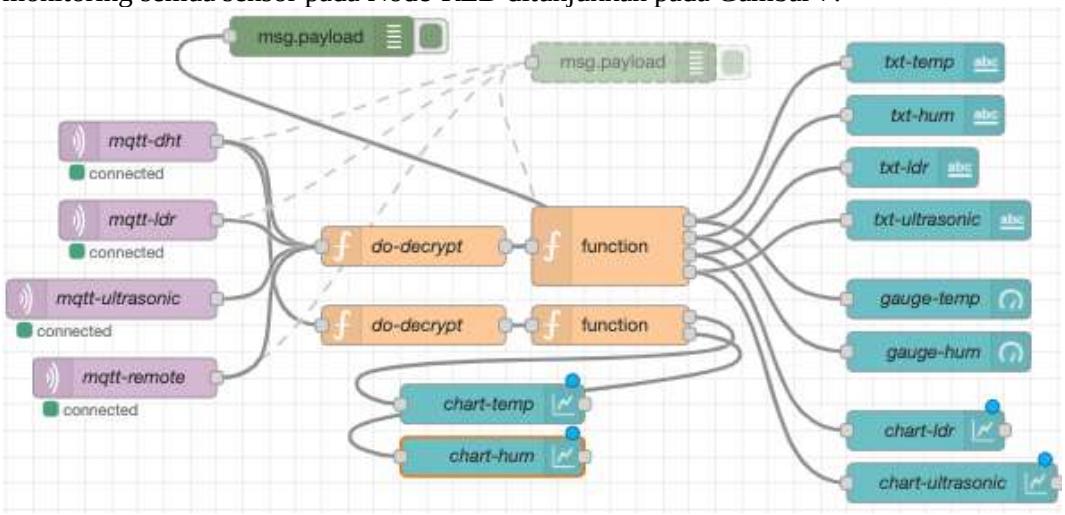
{"iv":"CigZ78pgTCTw9OM/LPnjJA==","msg":"eKPqmmqc5i/1jCT3dzcJEg=="}

Proses enkripsi pada Node-RED sama seperti pada NodeMCU, yaitu data yang di-*publish* dalam bentuk JSON yang terdiri dari komponen *iv* dan *msg* dengan format Base64. Komponen *iv* merupakan sebuah *initialization vector* yang digunakan saat proses dekripsi pada controller. Apabila dilakukan dekripsi cipherteks pada controller, maka hasil yang diperoleh adalah 5, yang artinya angka 5 ini digunakan untuk menyalakan LED sebanyak 5 buah.

Selanjutnya, untuk melakukan proses enkripsi pada Node-RED, perlu ditambahkan node *function* yang selanjutnya ditambahkan kode program menggunakan Javascript untuk melakukan proses enkripsi tersebut. Tampilan desain dari Node-RED ditunjukkan pada Gambar 6.

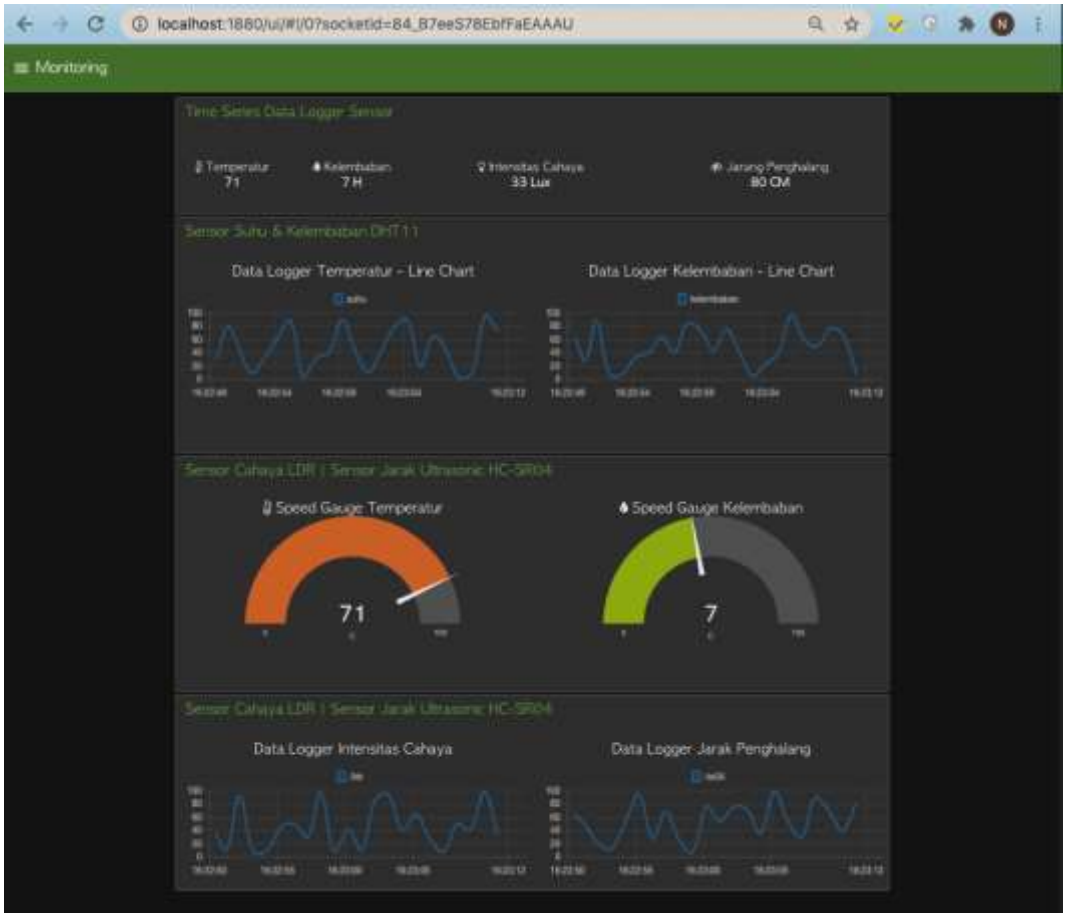

**Gambar 6. Desain Node-RED**

Node *slider-led* (warna biru) menunjukkan node untuk interaksi pengguna pada Node-RED. Node *do-encrypt* (warna oranye) berisikan kode program Javascript untuk melakukan proses enkripsi. Node *mqtt-led* (warna ungu) atau *mqtt-out* merupakan node yang digunakan untuk melakukan *publish* data ke message broker. Satu aliran proses tersebut difungsikan saat slider digeser ke kiri atau ke kanan dengan nilai tertentu, kemudian nilai tersebut akan dienkripsi dan di-*publish* ke message broker. Tampilan monitoring semua sensor pada Node-RED ditunjukkan pada Gambar 7.




**Gambar 7. Tampilan Monitoring Node-RED**

Gambar 7 menunjukkan sebuah flow untuk menampilkan sensor-sensor yang terdapat pada controller. Salah satu contoh yang ditunjukkan pada Gambar 7 tersebut adalah sensor DHT, yaitu untuk mendeteksi kelembaban dan suhu yang ditunjukkan pada node *mqtt-dht* (warna ungu) yang dihubungkan ke node *do-decrypt* dan node *function* (warna oranye), serta pada bagian terakhir dihubungkan ke node *txt-temp* dan *txt-hum* (warna biru). Node *mqtt-dht* merupakan sebuah node untuk melakukan *subscribe* data dari sensor DTH11. Data yang diperoleh tersebut masih dalam bentuk cipherteks, sehingga perlu dilakukan transformasi data ke dalam bentuk plainteks menggunakan node *do-decrypt*. Selanjutnya, node *function* digunakan untuk melakukan format pada data sebelum ditampilkan pada node *txt-temp* atau *txt-hum*. Hasil dari desain pada Gambar 7 ketika ditampilkan pada sebuah web browser ditunjukkan pada Gambar 8.




**Gambar 8. Hasil Penerapan Desain pada Web Browser**

### 3.3 Pengujian Fungsionalitas AES

Pengujian fungsional AES bertujuan untuk mengetahui keberhasilan proses enkripsi dan dekripsi yang dilakukan oleh controller (NodeMCU) ataupun oleh Node-RED. Mekanisme pengujian terdiri dari pengujian proses enkripsi dan enkripsi monitoring serta pengujian proses enkripsi dan dekripsi controlling.

Pengujian pada monitoring artinya data yang dienkripsi pada controller (NodeMCU) selanjutnya didekripsi agar data dapat ditampilkan. Beberapa skenario pengujian yang dilakukan pada proses monitoring ditunjukkan pada Tabel 1.

**Tabel 1. Hasil Pengujian Monitoring**

| No | Sensor | Cipherteks | Plainteks | Status | Hasil |
|---|---|---|---|---|---|
| 1 | DHT11 | {"iv":"YRjCdxBbpdXNQq1SImod+Q==", "msg":"B8v96Wd3S2Yg+6yXROeFYcosfeTN 2XvftfezObT6hiTW2Q3tc2W/Ma9q6WJleIIv"} | {"suhu":28, "kelembaban":67} | Data berhasil dienkripsi dan tampil pada Node-RED | Sesuai |
| 2 | LDR | {"iv":"VQwYG3s08wCkmoWkRbZJuQ==", "msg":"nAcwM6oMZ0qxJE64grp5yQ=="} | 119 | Data berhasil dienkripsi dan tampil pada Node-RED | Sesuai |
| 3 | Ultrasonic | {"iv":"z48NT2/cEVSl00lqcPCHZw==", "msg":"nP4JugPOjs2+QHaXB+hc Dg=="} | 31 | Data berhasil dienkripsi dan tampil pada Node-RED | Sesuai |

Hasil pengujian pada Tabel 1 menunjukkan bahwa proses enkripsi dan dekripsi berhasil dilakukan, yaitu hasil yang diberikan sesuai dengan yang diharapkan. Misalnya

untuk sensor DHT11, nilai kelembaban dan suhu dapat ditampilkan. Pengujian dianggap berhasil jika nilai yang ditampilkan pada Node-RED berhasil dilakukan dekripsi dan nilai tersebut sesuai dengan yang ditampilkan pada LCD OLED pada controller (NodeMCU).

Selanjutnya, pengujian pada controlling juga dilakukan dengan melakukan proses enkripsi ataupun controlling untuk mengontrol aktuator yang terdapat pada controller (NodeMCU) menggunakan Node-RED. Beberapa skenario pengujian pada proses controlling ditunjukkan pada Tabel 2.

**Tabel 2. Hasil Pengujian Controlling**

| No | Sensor | Cipherteks | Plainteks | Status | Hasil |
|---|---|---|---|---|---|
| 1 | LED | {"iv":"sbNtyZvRL54t05 wam2XveA==", "msg":"HZJ8YbIvMgOFK8 WpGJ/o1g=="} | 1 | Data berhasil didekrip dan LED akan menyala 1 buah | Sesuai |
| 2 | Relay | {"iv":"EN5S+FkGWL 9imSs3hOY9BA==", "msg":"eOkuW3cxU9JUm wkw65cYKw=="} | on | Data berhasil didekrip dan relay akan status on | Sesuai |
| 3 | FAN | {"iv":"Iaja0iYUq6KV Erc6uKQTZg==", "msg":"gVvWGAWJtpx8e2 +3D866yQ=="} | 50 | Data berhasil didekrip dan FAN akan berputar dengan 50 RPM | Sesuai |
| 4 | Speaker Piezo/buzzer | {"iv":"Ey+JP+uOGvQc 3Z3QEILm1w==", "msg":"HULENX+WHY KgOihtpzUskQ=="} | on | Data berhasil didekrip dan buzzer akan berbunyi | Sesuai |

Pengujian controlling dilakukan dengan menjalankan beberapa aktivitas pada Node-RED yang berupa aksi-aksi controlling, misalnya LED digeser nilainya menjadi 1 sehingga data akan dienkripsi dan di-*publish*. Setelah itu, data akan di-*subscribe* oleh controller (NodeMCU). Selanjutnya, data tersebut akan didekripsi sehingga LED dapat dinyalakan sesuai nilai yang diterima tersebut.

## 4. KESIMPULAN DAN SARAN

Berdasarkan hasil pengujian yang telah dilakukan, dapat disimpulkan kesimpulan bahwa penggunaan Advanced Encryption Standard mode Cipher Block Chaining (CBC) dapat memberikan pengamanan data saat melakukan proses pengiriman data dari perangkat IoT sampai dengan aplikasi dashboard monitoring. Hal ini dibuktikan dengan bentuk data yang tidak dapat secara langsung dibaca dan diterjemahkan karena perlu ada mekanisme lain untuk menerjemahkan data tersebut, meskipun hanya dengan menggunakan key statis sepanjang 128 bit. Untuk menambahkan keamanan pada data, diterapkan initialization vector (*iv*) yang nilainya selalu berubah-ubah saat proses enkripsi dan mengubah base. Dengan demikian, proses pengamanan data menjadi lebih optimal.

Untuk meningkatkan keamanan data, pada penelitian selanjutnya dapat digunakan mekanisme pengamanan data menggunakan key yang bersifat asimetris. Dengan menggunakan key asimetris, key yang digunakan untuk melakukan proses enkripsi dan dekripsi dibedakan sehingga data yang dikirimkan akan menjadi lebih aman dari penyalahgunaan pihak ketiga yang tidak bertanggung jawab. Salah satu contoh penerapan key asimetris yang dapat digunakan adalah algoritma Riverst-Shamir-Adleman (RSA).

## 5. DAFTAR RUJUKAN

[1] A. Mosenia and N. K. Jha, "A comprehensive study of security of internet-of-things," *IEEE Trans. Emerg. Top. Comput.*, vol. 5, no. 4, pp. 586–602, 2017.

[2] C. Stergiou, K. E. Psannis, A. P. Plageras, Y. Ishibashi, and B.-G. Kim, "Algorithms for efficient digital media transmission over IoT and cloud

networking," *J. Multimed. Inf. Syst.*, vol. 5, no. 1, pp. 27–34, 2018.
[3] V. A. Memos, K. E. Psannis, Y. Ishibashi, B. G. Kim, and B. B. Gupta, "An Efficient Algorithm for Media-based Surveillance System (EAMSuS) in IoT Smart City Framework," *Futur. Gener. Comput. Syst.*, vol. 83, pp. 619–628, 2018.
[4] U. Farooq, N. Ul Hasan, I. Baig, and N. Shehzad, "Efficient adaptive framework for securing the Internet of Things devices," *Eurasip J. Wirel. Commun. Netw.*, vol. 1, pp. 1–13, 2019.
[5] R. K. Endrayanto, A. Muttaqin, and R. A. Setyawan, "Advanced Encryption Standard (AES) pada Modul Internet of Things (IoT)," *TELKA - Telekomun. Elektron. Komputasi dan Kontrol*, vol. 5, no. 2, pp. 103–113, 2019.
[6] R. Ravida and H. A. Santoso, "Advanced Encryption Standard (AES) 128 Bit for Hydroponic Plant Internet of Things (IoT) Data Security," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 4, no. 6, pp. 1157–1164, 2020.
[7] D. Suprianto, R. Agustina, and T. A. Izzuddin, "' Best Practice ' Pengembangan Aplikasi Internet of Things," 2021.
[8] I. Maulana, A. Kusyanti, and A. Bhawiyuga, "Implementasi Algoritme Grain V1 Pada Protokol MQTT Menggunakan Raspberry Pi Untuk Mengamankan Data IoT," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 2, no. 12, pp. 6542–6549, 2018.
[9] S. Mulyono, M. Qomaruddin, and M. Syaiful Anwar, "Penggunaan Node-RED pada Sistem Monitoring dan Kontrol Green House berbasis Protokol MQTT," *J. Transistor Elektro dan Inform. (TRANSISTOR EI)*, vol. 3, no. 1, pp. 31–44, 2018.
[10] P. Jindal, A. Kaushik, and K. Kumar, "Design and Implementation of Advanced Encryption Standard Algorithm on 7th Series Field Programmable Gate Array," in *2020 7th International Conference on Smart Structures and Systems (ICSSS)*, 2020, pp. 1–3.
[11] G. Singh and S. Supriya, "A Study of Encryption Algorithms (RSA, DES, 3DES and AES) for Information Security," *Int. J. Comput. Appl.*, vol. 67, no. 19, pp. 33–38, 2013.
[12] A. Ginting, R. R. Isnanto, and I. P. Windasari, "Implementasi Algoritma Kriptografi RSA untuk Enkripsi dan Dekripsi Email," *J. Teknol. dan Sist. Komput.*, vol. 3, no. 2, p. 253, 2015.
[13] S. Oukili and S. Bri, "High speed efficient advanced encryption standard implementation," in *2017 International Symposium on Networks, Computers and Communications (ISNCC)*, 2017, pp. 1–4.
[14] S. U. Jonwal and P. P. Shingare, "Advanced Encryption Standard (AES) implementation on FPGA with hardware in loop," in *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, 2017, pp. 64–67.
[15] A. Purwinarko and W. Hardyanto, "A Hybrid Security Algorithm AES and Blowfish for Authentication in Mobile Applications," *Sci. J. Informatics*, vol. 5, no. 1, pp. 76–80, 2018.
[16] A. Pramudita, "Penggunaan Enkripsi AES Dengan Mode Operasi CBC dan CTR pada Javascript dengan Library PidCrypt."