IMPLEMENTASI ALGORITME AES PADA PENGIRIMAN DATA SENSOR DHT11 MENGGUNAKAN PROTOKOL KOMUNIKASI HTTP

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Aulia Rizqy Pratama
NIM: 135150301111107

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI ALGORITME AES 128 BIT PADA PENGAMANAN DATA SENSOR DHT11 MENGGUNAKAN PROTOKOL HTTP

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Aulia Rizqy Pratama
NIM: 135150301111107

Penelitian ini telah diuji dan dinyatakan lulus pada
26 Desember 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Mochammad Hannats Hanafi Ichsan,S.ST.,M.T	Ari Kusyanti,ST.,M.Sc
NIK: 201405 881229 1 001	NIP. 19831228 201803 200 2

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP. 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 26 Desember 2018

Aulia Rizqy Pratama

NIM: 135150301111107

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas limpahan rahmat dan petunjuk-Nya, sehingga penulis dapat menyelesaikan skripsi dengan judul "**Implementasi Algoritme AES pada Pengiriman Data Sensor DHT11 Menggunakan Protokol HTTP**". Skripsi ini disusun untuk memenuhi sebagian persyaratan untuk memperoleh gelar Sarjana Komputer di Fakultas Ilmu Komputer Universitas Brawijaya Malang.

Penyusunan dan penulisan skripsi ini tidak lepas dari dukungan moral dan materil dari berbagai pihak, maka penulis ingin menyampaikan rasa hortma serta mengucapkan terima kasih penulis kepada:

1. Bapak Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya Malang.
2. Bapak Tri Astoto Kurniawan, S.T., M.T., P.hD. selaku Ketua Jurusan Teknik Informatika Universitas Brawijaya Malang.
3. Bapak Dahnial Syauqy, S.T., M.T., M.Sc. selaku Ketua Program Studi Teknik Komputer Universitas Brawijaya Malang.
4. Bapak Mochammad Hannats Hanafi Ichsan, S.T., S.ST. selaku Dosen Pembimbing I yang telah memberikan pengarahan dan bimbingan kepada penulis, sehingga dapat menyelesaikan penulisan skripsi ini.
5. Ibu Ari Kusyanti, S.T., M.Sc. selaku Dosen Pembimbing II yang telah memberikan pengarahan dan bimbingan kepada penulis, sehingga dapat menyelesaikan penulisan skripsi ini.
6. Ayahanda Anil Mujai dan Ibunda Sri Anggraini atas segala nasehat, dukungan, kasih sayang, perhatian dan kesabarannya dalam memberikan semangat kepada penulis, serta doa yang tiada hentinya diberikan kepada penulis sehinnga penulisan skripsi ini dapat terselesaikan.
7. Adik tercinta Nadya Andini, Karina Dewyn Senda dan Anindi Faraz Cantika atas dukungan, semangat dan motivasi yang diberikan kepada penulis sehingga dapat menyelesaikan penulisan skripsi ini.
8. Keluarga besar Abbas dan Ali Badar atas dukungan dan semangat yang diberikan kepada penulis untuk menyelesaikan penulisan skripsi ini.
9. Sahabat Agung Prasetyo, S.Kom, Fatkrurrosi, S.Kom, Rinaldi Albert, S.Kom, Wildan Alaudin, S.Kom dan seluruh sahabat-sahabat penulis atas dukungan dan bantuan yang diberikan kepada penulis dalam penulisan skripsi ini.
10. Keluarga besar Baseball-Softball Universitas Brawijaya atas dukungan dan saran serta kekeluargaan kepada penulis sehingga dapat menyelesaikan penulisan skripsi ini.

11. Seluruh civitas akademika Informatika Universitas Brawijaya Malang dan terkhusus untuk teman-teman Teknik Komputer Universitas Brawijaya Malang 2013 yang telah banyak membantu dan mendukung serta memberikan motivasi kepada penulis selama penulis menempuh studi di Teknik Komputer Universitas Brawijaya Malang dan selama penyelesaian penulisan skripsi ini.
12. Seluruh pihak yang membantu penelitian, penulis mengucapkan terima kasih atas segala bentuk dukungan dan doa sehingga penulisan skripsi ini dapat terselesaikan.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih terdapat banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap penelitian ini dapat membawa manfaat bagi semua pihak yang membutuhkan.

Malang, 26 Desember 2018

Aulia Rizqy Pratama
auliarizqypratama@gmail.com

ABSTRAK

HTTP merupakan protokol jaringan pada lapisan aplikasi TCP/IP yang digunakan untuk komunikasi atau pertukaran data antara server dengan client. Terdapat kelemahan pada protokol HTTP dimana pada pengiriman data atau informasi tidak terdapat proses pengamanan atau enkripsi yang dimana dapat membahayakan informasi atau data yang dikirim pada suatu jaringan. Pada tahun 2001 National Institute of Standard and Technology (NIST) mempublikasikan standar algoritme kriptografi terbaru yang digunakan untuk menggantikan algoritme DES yang sudah tidak efisien penggunaannya yaitu algoritme AES. Penelitian ini mengusulkan ide untuk memanfaatkan algoritme AES sebagai pengamanan pengiriman data sensor DHT11 menggunakan protokol HTTP. Pada penelitian ini menggunakan Arduino Uno sebagai mikrokontroler yang digunakan untuk mengolah data sensor DHT11 dan NI Labview untuk menampilkan keluaran dari sistem. HTTP server dan proses enkripsi diimplementasikan di Arduino IDE dan HTTP client dan proses deskripsi pada NI Labview. Dari hasil pengujian didapatkan total error rate pada enkripsi 0%, deskripsi 24% dan pengiriman data 0%. Pada nilai rata-rata waktu pemrosesan enkripsi dan deskripsi sebesar 13,91 detik dengan rata-rata delay pengiriman pada protokol HTTP sebesar 0,00276 detik. Dari hasil pengujian disimpulkan bahwa algoritme AES dapat diimplementasikan untuk pengamanan pengiriman data dari sensor DHT11 menggunakan protokol HTTP.

Kata Kunci: protokol HTTP, algoritme AES, pengiriman data, Arduino, NI Labview

ABSTRACT

HTTP is a network protocol on the TCP / IP application layer that is used for communication or data exchange between the server and the client. There is a weakness in the HTTP protocol where the data or information transmission does not have a security or encryption process which can endanger information or data sent on a network. In 2001 the National Institute of Standard and Technology (NIST) published the latest cryptographic algorithm standard that was used to replace the DES algorithm that was not efficient in the using with the AES algorithm. This study proposes an idea to utilize the AES algorithm as a safeguard for sending DHT11 sensor data using the HTTP protocol. In this study using Arduino Uno as a microcontroller used to process data from DHT11 sensor and NI Labview to display the output of the system. The HTTP server and encryption process are implemented on the Arduino IDE and HTTP client and the description process in NI Labview. From the test results, the total error rate is 0% encryption, 24% description and 0% data transmission. On the average value of encryption processing time and description is 13.91 seconds with the average delivery delay on the HTTP protocol by 0.00276 seconds. From the test results it can be concluded that the AES algorithm can be implemented to secure the sending of data from the DHT11 sensor using the HTTP protocol.

***Keyword**: HTTP protocol, AES algorithym, data transmission, Arduino Uno, NI Labview*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Internet merupakan salah satu terapan dari teknologi telekomunikasi yang banyak menggunakan protokol jaringan berbasiskan *Transmission Control Protocol/Internet Protocol* (TCP/IP), salah satunya adalah *Hypertext Transfer Protocol* (HTTP). HTTP merupakan protokol jaringan pada lapisan aplikasi TCP/IP yang digunakan untuk komunikasi atau pertukaran data antara server dengan *client* (Susana, 2015). HTTP juga merupakan sebuah format standar yang biasanya digunakan oleh pembuat situs agar komputer *client* dapat mengakses sebuah halaman pada *browser* atau *website*. Akan tetapi terdapat kekurangan pada faktor keamanan pertukaran data atau informasi pada protokol HTTP di mana dalam proses pengiriman datanya tidak terdapat proses enkripsi atau pengamanan pada datanya yang dimana hal ini dapat membahayakan informasi atau data yang dikirimkan dalam suatu jaringan (Musliyana, 2018).

Penelitian sebelumnya yang dilakukan oleh Zuhar Musliyana dengan judul penelitian *"Improvement of Data Exchange Security on HTTP using Client-side Encryption"* yang menjelaskan tentang penggunaan algoritme AES untuk pengamanan protokol HTTP pada sisi *client*. Di mana pada penelitian ini dijelaskan algoritme AES digunakan untuk mengamankan pengiriman informasi *client*/*web browser* berupa *username* dan *password* yang di mana informasi itu akan dienkripsi sebelum dikirim menuju *web server* dan informasi yang diterima oleh *web server* berupa *chipertext* lalu *chipertext* tersebut akan dideskripsi pada *web server* dan diubah menjadi *plaintext*. Dan dari hasil pengujian didapatkan bahwa algoritme AES dapat diimplementasikan pada protokol HTTP untuk menyelesaikan masalah keamanan pengiriman informasi pada *web browser* (Musliyana, 2018). Terdapat penelitian lain yang dilakukan oleh Adimas Fiqri Ramdhansya dengan judul Implementasi *Advanced Encryption Standart* (AES) Pada Sistem Kunci Elektronik Kendaraan Berbasis Sistem Operasi Android dan mikrokontroler Arduino yang menjelaskan tentang implementasi algoritme AES untuk mengenkripsi dan mendeskripsi pesan antara mikrokontroler Arduino dengan sistem operasi *mobile* Android. Dari hasil analisis dan pengujian didapatkan bahwa transmisi antara telepon genggam dengan Arduino melalui Bluetooth menjadi lebih aman dan pesan yang terkirim tidak dapat dibaca oleh pihak lain.

Penulis mengusulkan untuk mengimplementasikan algoritme enkripsi AES sebagai algoritme keamanan pada pengiriman data sensor DHT11 dengan menggunakan protokol HTTP. Algoritme AES dipilih dengan mempertimbangkan bahwa algoritme AES mempunyai keseimbangan dalam hal keamanan dan fleksibilitas di berbagai macam perangkat lunak dan perangkat keras. Algoritme AES merupakan algoritme kunci simetris yang menggunakan satu kunci bersama antara pengirim dan penerima yang digunakan baik untuk enkripsi dan deskripsi, algoritme AES sendiri di publikasikan oleh *National Institute of Standarts and Technology* (NIST) sebagai pengganti algoritme DES yang sudah tidak efisien

penggunaannya dan juga algoritme AES digunakan sebagai standar algoritme kunci simetris sampai saat ini (Musliyana, 2018). Algoritme AES akan diimplementasikan untuk pengamanan pada pengiriman data sensor DHT11 yang akan diimplementasikan pada Arduino dan NI Labview di mana proses enkripsi dan server dari protokol HTTP akan diimplementasikan pada Arduino dan proses deskripsi dan *client* protokol HTTP akan diimplementasikan pada NI Labview. Sensor DHT11 sendiri merupakan sensor pengukuran suhu dan kelembapan secara serempak dengan nilai keluarannya berupa dijital yang di mana tidak perlu lagi melakukan proses konversi dari sinyal analog (Saptadi, 2015). Arduino sendiri adalah pengendali mikro *single board* yang bersifat *open source* yang dirancang untuk memudahkan pengguna elektronik dalam berbagai bidang. Arduino sendiri mempunyai perangkat lunak sendiri yaitu Arduino IDE yang bersifat *open source* yang menggunakan Bahasa pemrograman C untuk AVR dengan dukungan *libraries* bahasa C memudahkan dalam pengembangan lebih lanjut (Manaloe, 2015). NI Labview adalah perangkat lunak dengan konsep pemrograman berbasis grafis atau blok diagram dengan sebutan *virtual instrument*. Dalam membentuk instruksi pada Labview menggunakan berbagai macam ikon serta pemrograman yang dilakukan menggunakan metode dataflow dengan menentukan alur data melalui banyak instruksi untuk dapat di implementasikan (Anadiansyah, 2017). Kedua *platform* ini dipilih dengan alasan pada Arduino terdapat *library* untuk pemrograman sensor DHT11 yang mana pada NI Labview masih tidak kompatibel untuk program sensor DHT11. NI Labview disini digunakan selain untuk implementasi dekripsi dan HTTP *client* juga untuk *output* dari sistem ini dimana pada NI Labview terdapat sebuah antar muka yang dinamakan *front panel*. *Front panel* pada NI Labview berfungsi untuk menampilkan setiap fungsi yang didefinisikan pada *block diagram* pada NI Labview dimana fungsi yang didefinisikan pada *block diagram* adalah HTTP *client* dan dekripsi dari algoritme AES. Pengujian dan analisis kinerja berdasarkan total *error rate* dari enkripsi dan deskripsi algoritme AES, waktu yang dibutuhkan oleh algoritme AES untuk melakukan proses enkripsi dan deskripsi dan *delay* dari protokol HTTP. Penelitian ini diharapkan akan menjadi sebuah solusi keamanan pada pengiriman data sensor menggunakan protokol HTTP yang diimplementasikan pada perangkat yang berbeda yaitu Arduino dan NI Labview.

1.2 Rumusan Masalah

Bagian ini berisi pertanyaan penelitian. Adapun rumusan masalah yang dapat diambil dari latar belakang di atas adalah sebagai berikut:

1. Bagaimana perancangan algoritme AES pada pengiriman data sensor DHT11 menggunakan protokol HTTP?
2. Bagaimana cara mengimplementasikan algoritme AES sebagai algoritme enkripsi pada pengiriman data sensor DHT11 menggunakan protokol HTTP pada Arduino dan NI Labview?
3. Bagaimana kinerja algoritme AES yang diimplementasikan pada pengiriman data sensor DHT11 menggunakan protokol HTTP?

1.3 Tujuan

Bagian ini berisi tujuan yang ingin didapat dari skripsi ini. Adapun beberapa tujuan penelitian yang akan dilaksanakan sebagai berikut:

1. Untuk merancang algoritme AES pada pengiriman data sensor DHT11 menggunakan protokol HTTP.
2. Mengimplementasikan algoritme AES pada pengiriman data sensor DHT11 menggunakan protokol HTTP pada Arduino dan NI Labview.
3. Mengetahui kinerja algoritme AES pada pengiriman data sensor DHT11 menggunakan protokol HTTP.

1.4 Manfaat

Memberikan peningkatan keamanan data algoritme enkripsi AES pada pengiriman data sensor menggunakan protokol HTTP juga dapat memberikan ide kepada para *developer* IoT untuk pengembangan protokol atau enkripsi data kedepannya.

1.5 Batasan Masalah

Bagian ini dapat dituliskan untuk membantu menjelaskan ruang lingkup masalah penelitian dengan menyatakan hal-hal yang menjadi batasan dan asumsi-asumsi yang digunakan untuk menyelesaikan masalah yang sudah dirumuskan. Pada penelitian ini batasan masalah adalah sebagai berikut:

1. Pengimplementasian algoritme AES pada pengiriman data sensor menggunakan protokokol HTTP.
2. Sistem ini menggunakan algoritme AES versi 128 bit dengan key sepanjang 16 *byte* yang dimasukkan secara manual.
3. Data yang dikirimkan berasal dari hasil *sensing* Sensor DHT11 berupa suhu dan kelembapan.
4. Sistem ini menggunakan Arduino Uno yang ditambahkan Ethernet Shield sebagai mikrokontroler dan komunikasi jaringan.
5. Sistem yang digunakan harus terhubung secara serial dengan PC Komputer.

1.6 Sistematika Pembahasan

Bagian ini berisi struktur skripsi ini mulai Bab Pendahuluan sampai Bab Penutup dan deskripsi singkat dari masing-masing bab. Diharapkan bagian ini dapat membantu pembaca dalam memahami sistematika pembahasan isi dalam skripsi ini. Berikut sistematika pembahasan:

BAB I : PENDAHULUAN

Pada bab ini menjelaskan informasi umum yaitu latar belakang penelitian, perumusan masalah, tujuan dan manfaat penelitian, batasan penelitian dan sistematika penelitian.

BAB II : LANDASAN TEORI

Pada bab ini terdapat teori-teori tentang sistem yang dibuat berdasarkan kutipan buku, jurnal maupun web yang dibuat oleh lembaga resmi yang dapat dipertanggung jawabkan berupa pengertian dan definisi. Bab ini juga menjelaskan tentang konsep dasar sensor DHT11, HTTP, AES, dan informasi lain yang berkaitan dengan batasan masalah.

BAB III : METODE PENELITIAN

Pada bab ini terdapat langkah-langkah yang dilakukan pada penelitian yaitu: menguraikan dan menjelaskan langkah kerja yang dilakukan dalam penulisan skripsi yang terdiri dari studi literatur, analisis kebutuhan, pengumpulan data, desain sistem, implementasi, pengujian, pengolahan data dan analisis hasil penarikan kesimpulan.

BAB IV : REKAYASA KEBUTUHAN

Pada bab ini berisikan bahasan semua yang dibutuhkan oleh sistem. Mulai dari kebutuhan perangkat keras dan lunak hingga kebutuhan fungsional yang dibutuhkan.

BAB V : PERANCANGAN DAN IMPLEMENTASI

Pada bab ini membahas mengenai perancangan sistem, implementasi HTTP, AES, dan tampilan sistem pada NI Labview.

BAB VI : PENGUJIAN DAN ANALISIS

Pada bab ini berisikan hasil-hasil pengujian yang dilakukan dan analisis terhadap sistem yang telah direalisasikan.

BAB VII : PENUTUP

Pada bab ini terdapat kesimpulan akhir dari pembuatan dan pengujian perangkat keras dan perangkat lunak yang dikembangkan dalam skripsi ini serta saran-saran untuk pengembangan lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepustakaan berisi uraian dan pembahasan tentang teori, konsep, model, metode, atau sistem dari literatur ilmiah, yang berkaitan dengan tema, masalah, atau pertanyaan penelitian. Dalam landasan kepustakaan terdapat landasan teori dari berbagai sumber pustaka yang terkait dengan teori dan metode yang digunakan dalam penelitian. Jika dibutuhkan sesuai dengan karakteristik penelitiannya dan syarat kecukupan khusus keminatan tertentu, bisa juga terdapat kajian pustaka yang menjelaskan secara umum penelitian-penelitian terdahulu yang berhubungan dengan topik skripsi dan menunjukkan persamaan dan perbedaan skripsi tersebut terhadap penelitian terdahulu yang dituliskan.

2.1 Tinjauan Pustaka

Berikut ini beberapa penelitian terkait implementasi algoritme AES pada pengiriman data sensor DHT11 menggunakan protokol HTTP seperti penggunaan algoritme AES untuk pengamanan pengiriman data pada sisi *client* protokol HTTP atau implementasi algoritme enkripsi untuk mengamankan pengiriman data sensor pada protokol jaringan menggunakan Arduino yang dapat dilihat pada Tabel 2.1.

Tabel 2.1 Kajian Pustaka

No	Nama Penulis, Tahun dan Judul	Persamaan	Perbedaan	
			Penelitian Terdahulu	Rencana Penelitian
1.	Zuhar Mulyana(2018), "*Improvement of Data Exchange Security on HTTP using Client-side Encryption*"	Menggunakan algoritme AES dan protokol HTTP	Implementasi algoritme AES untuk pengamanan informasi pengguna menggunakan Protokol HTTP pada *web*	Menggunakan Algoritme Kriptografi AES untuk pengamanan pengiriman data sensor menggunakan protokol HTTP pada Arduino dan NI Labview
2.	First Wanita , Rancang Bangun Sistem Enkripsi dan Deskripsi Pengiriman Informasi Menggunakan Algoritme RSA	Implementasi pengamanan pengiriman data sensor menggunakan protokol jaringan	Implementasi menggunakan algoritme RSA	Implementasi menggunakan algoritme AES dan protokol HTTP

Tabel 2.1 Kajian Pustaka(Lanjutan)

No	Nama Penulis, Tahun dan Judul	Persamaan	Perbedaan	
			Penelitian Terdahulu	Rencana Penelitian
3	Mohammed Erritali(2011) , *"A CONTRIBUTION TO SECURE THE ROUTING PROTOKOL "GREEDY PERIMETER STATELESS ROUTING'' USING A SYMMETRIC SIGNATUREBASED AES AND MD5 HASH"*	Implementasi menggunakan algoritme AES	Menggunakan Algoritme kriptografi AES dan MD5 untuk pengamanan protokol *routing GPRS* pada VANET	Menggunakan algoritme AES untuk pengamanan pengiriman data sensor DHT11 menggunakan protokol HTTP

2.2 HTTP (*HyperText Transfer Protokol)*

HTTP merupakan sebuah protokol yang didesain untuk mentrasfer informasi dalam bentuk hipermedia antara server dengan sebuah *client*. HTTP juga mentransfer data suatu informasi melalui *header*-nya. *Header* HTTP ini merupakan bentuk pengembangan dari *Multipurpose Internet Mail Extentions* (MIMEs). Pengembangan ini memungkinkan HTTP untuk mentransfer informasi dalam bentuk biner dan informasi dalam bentuk tidak standar yang berhasil dinegosiasi antara server dan *client*. Secara umum, *delay* akan terjadi pada saat melakukan suatu negosiasi sebelum proses transfer data. Karena lama dari *delay* yang disebabkan oleh *overhead* akan menjadi relatif lebih besar pada transfer data yang sebetulnya singkat. HTTP merupakan protokol yang bersifat *stateless*, sehingga server akan memproses setiap *request* dari *user* secara terpisah dari *request* yang lain, independen terhadap *request* yang sebelumnya. HTTP menggunakan 8 bit untuk mentransfer semua tipe data yang mungkin. Mekanisme yang terjadi pada HTTP bila suatu *client* menginginkan layanan dari server dibagi dalam 4 langkah, yaitu:

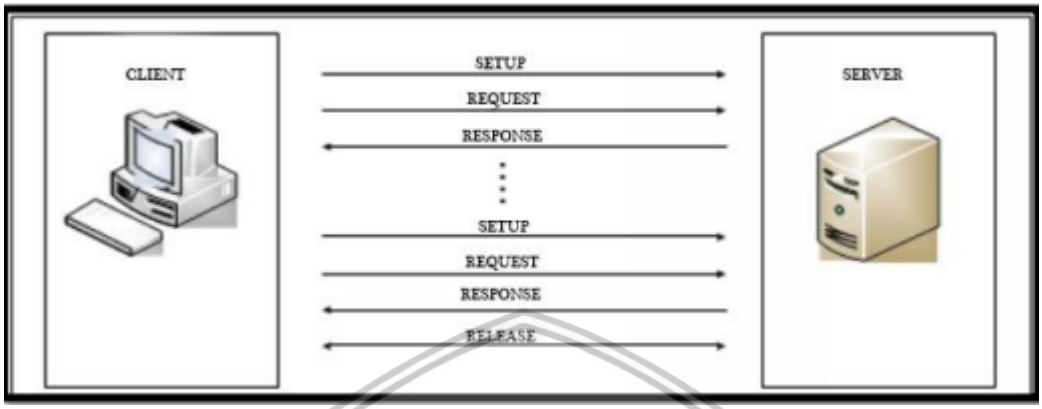
1. *Connection Setup Client* mengakses sebuah server dengan menggunakan alamat Internet dan *port number*. *Default* dari *port number* adalah 80.

2. *Request Client* mengirimkan pesan berupa informasi dari metode transaksi dan kapabilitas *client*.

3. *Response Server* mengirimkan *response* kepada *client* sesudah *client* itu menyelesaikan *request*-nya. *Response message* meliputi informasi dari transaksi dan data yang diminta.

4. *Connection Release Client* mengakhiri koneksi ke server.

Request dari *client* ke server dapat terjadi lebih dari satu kali. Dari setiap *request* yang dilakukan oleh *client*, server akan mengirimkan *response*. Setelah *request-response* selesai, terjadi proses *release* antara *client* dan server. (Indriawan, 2011)



Gambar 2.1 Proses Koneksi Protokol HTTP

Sumber: (Indriawan, 2011)

2.3 Algoritme Enkripsi AES

Kriptografi merupakan salah satu solusi atau metode pengamanan data yang tepat untuk menjaga kerahasiaan dan keaslian data, serta dapat meningkatkan aspek keamanan suatu data atau informasi. Metode ini bertujuan agar informasi yang bersifat rahasia dan dikirim melalui suatu jaringan, seperti LAN atau internet, tidak dapat diketahui atau dimanfaatkan oleh orang atau pihak yang tidak berkepentingan. Kriptografi mendukung kebutuhan dua aspek keamanan informasi, yaitu perlindungan terhadap kerahasiaan data informasi dan perlindungan terhadap pemalsuan dan pengubahan informasi yang tidak diinginkan.

Untuk mengetahui apakah suatu algoritme kriptografi dapat mengamankan data dengan baik dapat dilihat dari segi lamanya waktu proses pembobolan untuk memecahkan data yang telah disandikan. Seiring dengan perkembangan teknologi komputer yang semakin canggih, maka dunia teknologi informasi membutuhkan algoritme kriptografi yang lebih kuat dan aman. Saat ini, AES *(Advanced Encryption Standard*) merupakan algoritme *cipher* yang cukup aman untuk melindungi data atau informasi yang bersifat rahasia. Pada tahun 2001, AES digunakan sebagai standar algoritme kriptografi terbaru yang dipublikasikan oleh NIST (*National Institute of Standard and Technology*) sebagai pengganti algoritme DES (*Data Encryption Standard*) yang sudah tidak efisien penggunaannya. Algoritme AES adalah algoritme kriptografi yang dapat mengenkripsi dan mendekripsi data dengan panjang kunci yang bervariasi, yaitu 128 bit, 192 bit, dan 256 bit. (Musliyana, 2016)

Input dan *output* dari algoritme AES terdiri dari urutan data sebesar 128 bit. Urutan data yang sudah terbentuk dalam satu kelompok 128 bit tersebut disebut juga sebagai blok data atau *plaintext* yang nantinya akan dienkripsi menjadi

ciphertext. *Cipher key* dari AES terdiri dari *key* dengan panjang 128 bit, 192 bit, atau 256 bit. Perbedaan panjang kunci akan mempengaruhi jumlah ronde yang akan diimplementasikan pada algoritme AES ini. Berikut ini adalah Tabel 2.2 yang memperlihatkan jumlah ronde/putaran (Nr) yang harus diimplementasikan pada masing-masing panjang kunci.

Tabel 2.2 Perbandingan Jumlah Round dan *Key*

	Jumlah *Key* (Nk)	Jumlah Blok (Nb)	Jumlah Putaran (Nr)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Sumber: (Surian, 2006)

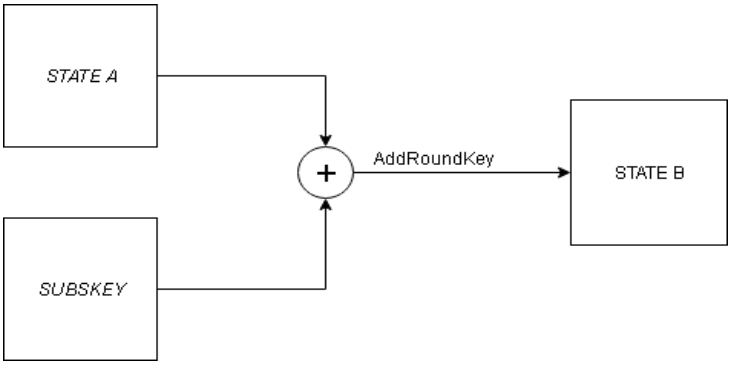
2.3.1 *KeySchedule*

Proses *key schedule* diperlukan untuk mendapatkan *subkey-subkey* dari kunci utama agar cukup untuk melakukan enkripsi dan dekripsi. Proses ini terdiri dari beberapa operasi, yaitu:

1. Operasi *Rotate*, yaitu operasi perputaran 8 bit pada 32 bit dari kunci.
2. Operasi *SubBytes*, pada operasi ini 8 bit dari *subkey* disubstitusikan dengan nilai dari *S-Box*.
3. Operasi *Rcon*, operasi ini dapat diterjemahkan sebagai operasi pangkat 2 nilai tertentu dari pengguna. Operasi ini menggunakan nilai-nilai dalam *Galois field*. Nilai-nilai dari *Rcon* kemudian akan di-XOR dengan hasil operasi *SubBytes*.
4. Operasi XOR dengan w[i-Nk] yaitu word yang berada pada Nk sebelumnya (Surian, 2006).

2.3.2 *AddRoundKey*

Pada proses ini *subkey* digabungkan dengan *state*. Proses penggabungan ini menggunakan operasi XOR untuk setiap *byte* dari *subkey* dengan *byte* yang bersangkutan dari *state*. Untuk setiap tahap, *subskey* dibangkitkan dari kunci utama dengan menggunakan proses *key schedule*. Setiap *subkey* berukuran sama dengan *state* yang bersangkutan (Surian, 2006). Proses *AddRoundKey* diperlihatkan pada Gambar 2.2.



Gambar 2.2 Proses *AddRoundKey*

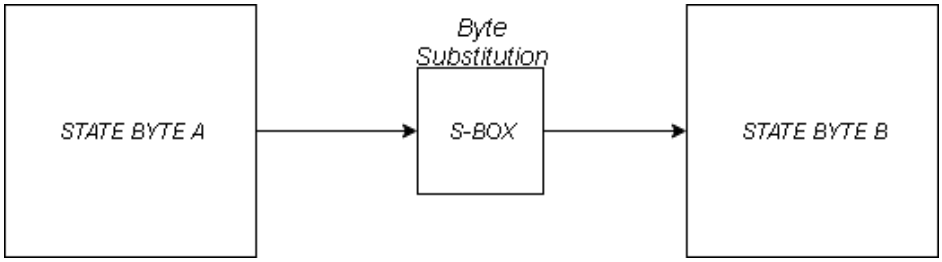
Sumber: (Surian, 2006)

2.3.3 SubBytes

Proses *SubBytes* adalah operasi yang akan melakukan substitusi tidak linear dengan cara mengganti setiap *byte state* dengan *byte* pada sebuah tabel yang dinamakan tabel *S-Box*. Sebuah tabel *S-Box* terdiri dari 16x16 baris dan kolom dengan masing-masing berukuran 1 byte. Tabel *S-Box* diperlihatkan pada Tabel 2.3 sedangkan proses *SubBytes* diperlihatkan pada Gambar 3.

Tabel 2.3 *S-Box*

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
01	ca	82	c9	7d	fa	59	47	f0	Ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	dl	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	0d
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

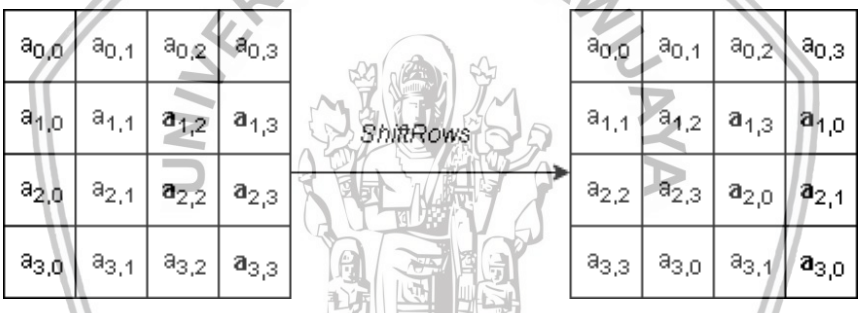


Gambar 2.3 Proses Subtitusi *Byte*

Sumber: (Surian, 2006)

2.3.4 *ShiftRows*

Proses *Shift Rows* akan beroperasi pada tiap baris dari tabel *state*. Proses ini akan bekerja dengan cara memutar *byte-byte* pada 3 baris terakhir (baris 1, 2, dan 3) dengan jumlah perputaran yang berbeda-beda. Baris 1 akan diputar sebanyak 1 kali, baris 2 akan diputar sebanyak 2 kali, dan baris 3 akan diputar sebanyak 3 kali. Sedangkan baris 0 tidak akan diputar. Proses *ShiftRows* dapat dilihat pada Gambar 2.4.



Gambar 2.4 *ShiftRows*

Sumber: (Surian, 2006)

2.3.5 *MixColumns*

Proses *MixColumns* akan beroperasi pada tiap kolom dari tabel *state*. Operasi ini menggabungkan 4 *bytes* dari setiap kolom tabel *state* dan menggunakan transformasi linier Operasi *Mix Columns* memperlakukan setiap kolom sebagai polinomial 4 suku dalam *Galois field* dan kemudian dikalikan dengan c(x) modulo (x4+1), di mana $c(x)=3x^3+x^2+x+2$. Kebalikkan dari polinomial ini adalah $c(x)=11x^3+13x^2+9x+14$. Operasi *MixColumns* juga dapat dipandang sebagai perkalian matrix. Langkah *MixColumns* dapat ditunjukkan dengan mengalikan 4 bilangan di dalam *Galois field* oleh matrix yang ditunjukkan pada Persamaan 2.1.

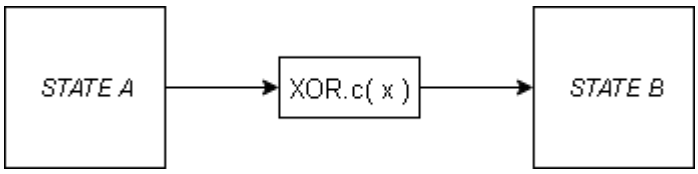
$$r_0=2a_0+a_3+a_2+3a_1$$

$$r_1=2a_1+a_0+a_3+3a_2$$

$$r_2=2a_2+a_1+a_0+3a_3 \qquad (2.1)$$

$$a_2=2a_3+a_2+a_1+3a_0$$

Operasi penjumlahan di atas dilakukan dengan operasi XOR, sedangkan operasi perkalian dilakukan dalam *Galois field*. Operasi perkalian dapat dilihat pada Gambar 2.5.



Gambar 2.5 Proses *MixColumns*

Sumber: (Surian, 2006)

2.4 Manualisasi Algoritme AES

Algoritme AES mempunyai *input* data sebesar 128 bit dengan panjang kunci yang bervariasi tergantung penggunaannya. Pada penelitian ini menggunakan AES-128 atau algoritme AES dengan panjang kunci 128 bit, pada AES-128 jumlah *round* atau ronde yang akan diimplementasikan sebanyak sepuluh. Berikut adalah proses enkripsi untuk merupah *plaintext* menjadi *chipertext*.

2.4.1 *KeySchedule*

Langkah awal dari proses enkripsi pada algoritme AES adalah *Keyschedule* yang dimana pada proses ini dilakukan untuk mendapatkan *roundkey* agar cukup untuk melakukan proses enkripsi pada setiap rondenya. Pertama disini terdapat sebuah kunci "Thats my Kung Fu" dan *plaintext* "Two One Nine Two", kunci dan *plaintext* ini akan diubah terlebih dahulu kedalam bentuk *hexadecimal* seperti pada Gambar 2.6 dan Gambar 2.7 berikut ini.

T	h	a	t	s		m	y		K	u	n	g		F	u
54	68	61	74	73	20	6D	79	20	4B	75	6E	67	20	46	75

Gambar 2.6 *Key* dalam bentuk HEX

T	w	o		O	n	e		N	i	n	e		T	w	o
54	77	6F	20	4F	6E	65	20	4E	69	6E	65	20	54	77	6F

Gambar 2.7 *Plaintext* dalam bentuk HEX

Selanjutnya memecah kunci sepanjang 128 bit menjadi empat bagian sepanjang 32 bits sebagai berikut:

Key in Hex (128 bits) : 54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75

w[0]=(54,68,61,74), w[1]=(73,20,6D,79) (2.2)

w[2]=(20,4B,75,6E), w[3]=(67,20,46,75)

Langkah selanjutnya adalah melakukan operasi *Rotate, SubBytes, Rcon* dan XOR untuk mendapatkan *RoundKey*. Operasi pertama adalah *Rotate* dimana kunci 32 bit tadi akan diputar sebanyak 8 bit. Setelah itu dilakukan operasi kedua yaitu operasi *SubsBytes* dimana para operasi ini kunci disubstitusikan dengan tabel substitusi atau *S-Box.* Setelah itu operasi selanjutnya adalah operasi *Rcon* atau *round constant* dimana kunci tadi akan ditambahkan dengan nilai konstan yang ada pada *Galais field*. Selanjutnya adalah operasi terakhir yaitu operasi XOR dimana w[i-Nk] XOR dengan Nk sebelumnya. Setelah semua operasi dilakukan maka didapatkan *roundkey* pertama yaitu "E2 32 FC F1 91 12 91 88 B1 59 E4 E6 D6 79 A2 93".

g(w[3]):

circular byte left shift w[3] : (20, 46, 75, 67)

Byte Substitution (S-Box) : (B7, 5A, 9D, 85)

Adding round constant (01,00,00,00) to g(w[3])= (B6,5A,9D,85)

Operasi XOR : (2.3)

w[4] = w[0] ⊕ g(w[3]) = (54,68,61,74) ⊕ (B6,5A,9D,85) = (E2,32,FC,F1)

w[5] = w[4] ⊕ w[1] = (E2,32,FC,F1) ⊕ (73,20,6D,79) = (91,12,91,88),

w[6] = w[5] ⊕ w[2] = (91,12,91,88) ⊕ (20,4B,75,6E) = (B1,59,E4,E6),

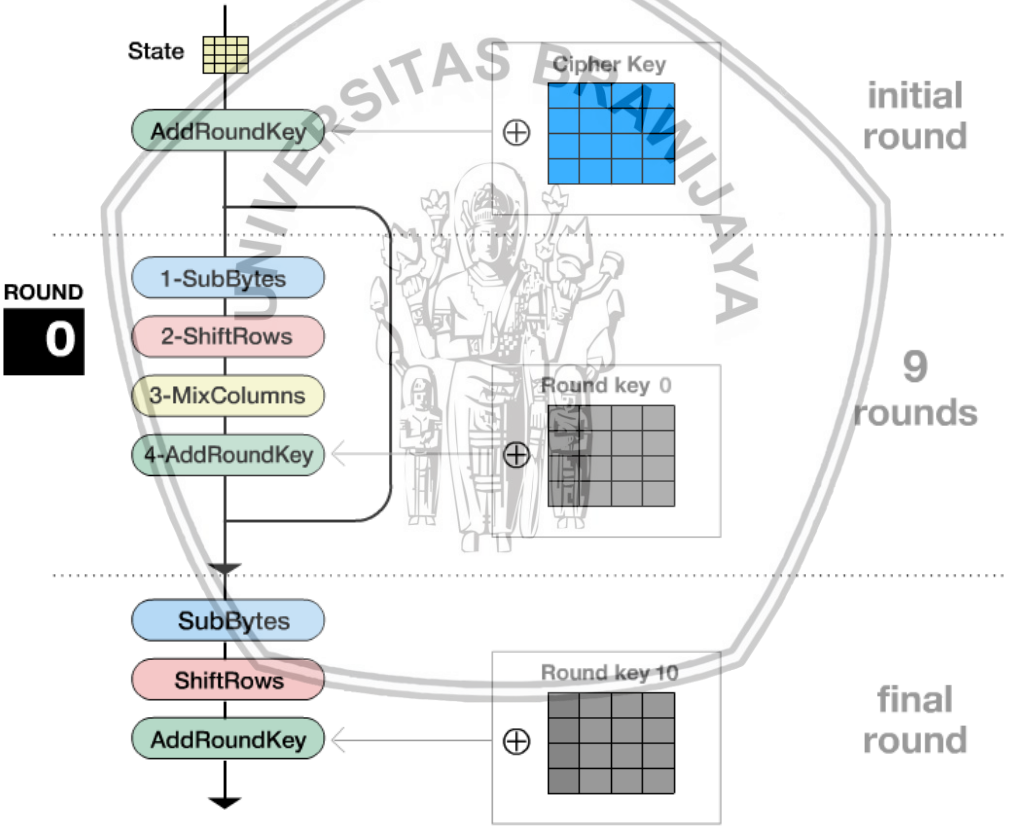
w[7] = w[6] ⊕ w[3] = (B1,59,E4,E6) ⊕ (67,20,46,75) = (D6,79,A2,93)

Tabel 2.4 *RoundKey*

Ronde ke-	*Key*
0	*54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75*
1	*E2 32 FC F1 91 12 91 88 B1 59 E4 E6 D6 79 A2 93*
2	*56 08 20 07 C7 1A B1 8F 76 43 55 69 A0 3A F7 FA*
3	*D2 60 0D E7 15 7A BC 68 63 39 E9 01 C3 03 1E FB*
4	*A1 12 02 C9 B4 68 BE A1 D7 51 57 A0 14 52 49 5B*
5	*B1 29 3B 33 05 41 85 92 D2 10 D2 32 C6 42 9B 69*
6	*BD 3D C2 B7 B8 7C 47 15 6A 6C 95 27 AC 2E 0E 4E*
7	*CC 96 ED 16 74 EA AA 03 1E 86 3F 24 B2 A8 31 6A*

8	*8E 51 EF 21 FA BB 45 22 E4 3D 7A 06 56 95 4B 6C*
9	*BF E2 BF 90 45 59 FA B2 A1 64 80 B4 F7 F1 CB D8*
10	*28 FD DE F8 6D A4 24 4A CC C0 A4 FE 3B 31 6F 26*

Proses ini diulang sebanyak sepuluh kali sehingga menghasilkan sepuluh macam *roundkey* untuk proses enkripsi Tabel 2.4 menunjukkan semua *roundkey* yang dihasilkan dari sepuluh kali proses pembuatan *KeyScheduled*. Proses selanjutnya adalah proses enkripsi untuk menghasilkan *chipertext* dari perubahan *plaintext*. Pada tahap ini akan dilakukan empat transformasi yaitu *AddRoundkey, SubsBytes, ShiftRows dan MixColumns*.



Gambar 2.8 Proses Enkripsi

2.4.2 Proses *AddRoundKey*

Pada proses ini *plaintext* atau biasa disebut *state* pada algoritme AES akan digabungkan dengan *roundkey* dengan menggunakan operasi XOR yang akan menghasilkan *state* yang baru.

Plaintext : 54 77 6F 20 4F 6E 65 20 4E 69 6E 65 20 54 77 6F (2.4)

Roundkey 0 : 54 68 61 74 73 20 6D 79 20 4B 75 6E 67 20 46 75

$$\begin{pmatrix} 54 & 4F & 4E & 20 \\ 77 & 6E & 69 & 54 \\ 6F & 65 & 6E & 77 \\ 20 & 20 & 65 & 6F \end{pmatrix} \oplus \begin{pmatrix} 54 & 73 & 20 & 67 \\ 68 & 20 & 4B & 20 \\ 61 & 6D & 75 & 46 \\ 74 & 79 & 6E & 75 \end{pmatrix} = \begin{pmatrix} 00 & 3C & 6E & 47 \\ 1F & 4E & 22 & 74 \\ 0E & 08 & 1B & 31 \\ 54 & 59 & 0B & 1A \end{pmatrix}$$

2.4.3 Proses *SubsBytes*

Proses selanjutnya adalah transformasi *SubsBytes* dimana pada proses ini *state* matrix yang dihasilkan dari proses sebelumnya disubstitusikan dengan *byte* yang ada pada tabel substitusi atau *S-Box,* contohnya pada *state* matrix byte 6E disubtitusikan dengan masukkan dari *S-Box* baris 6 dan kolomE dimana pada *S-Box* baris 6 dan kolom E adalah 9F.

$$\begin{pmatrix} 00 & 3C & 6E & 47 \\ 1F & 4E & 22 & 74 \\ 0E & 08 & 1B & 31 \\ 54 & 59 & 0B & 1A \end{pmatrix} = \begin{pmatrix} 00 & 3C & 9F & 47 \\ 1F & 4E & 22 & 74 \\ 0E & 08 & 1B & 31 \\ 54 & 59 & 0B & 1A \end{pmatrix} = \begin{pmatrix} 63 & EB & 9F & A0 \\ C0 & 2F & 93 & 92 \\ AB & 30 & AF & C7 \\ 20 & CB & 2B & A2 \end{pmatrix} \quad (2.5)$$

Tabel 2.5 Transformasi *SubsByte*

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
01	ca	82	c9	7d	fa	59	47	f0	Ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	dl	00	Ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	Fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	Ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	Dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	0d
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

2.4.4 Proses *ShiftRows*

Proses selanjutnya adalah transformasi *ShiftRows* dimana pada proses ini *byte* pada *state* akan diputar pada tiga baris terakhir dengan jumlah perputaran baris satu diputar satu kali, baris kedua diputar dua kali dan baris ketiga akan diputar tiga kali.

$$\begin{pmatrix} 63 & EB & 9F & A0 \\ C0 & 2F & 93 & 92 \\ AB & 30 & AF & C7 \\ 20 & CB & 2B & A2 \end{pmatrix} = \begin{pmatrix} 63 & EB & 9F & A0 \\ 2F & 93 & 92 & C0 \\ AF & C7 & AB & 30 \\ A2 & 20 & CB & 2B \end{pmatrix} \quad (2.6)$$

2.4.5 Proses *MixColumns*

Proses selanjutnya adalah transformasi *MixColumns* yaitu pada proses ini mengalikan setiap kolom pada *state* dengan sebuah matrix dari *Galios field*.

$$\begin{pmatrix} 02\,03\,01\,01 \\ 01\,02\,03\,01 \\ 01\,01\,02\,03 \\ 03\,01\,01\,02 \end{pmatrix}$$

Gambar 2.9 Matrix *Galios field*

$$\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} 63 & EB & 9F & A0 \\ 2F & 93 & 92 & C0 \\ AF & C7 & AB & 30 \\ A2 & 20 & CB & 2B \end{pmatrix} = \begin{pmatrix} BA & 84 & E8 & 1B \\ 75 & A4 & 8D & 40 \\ F4 & 8D & 06 & 7D \\ 7A & 32 & 0E & 5D \end{pmatrix}$$

Ex. BA hasil dari (02 • 63) $\oplus$ (03 • 2F) $\oplus$ (01 • AF) $\oplus$ (01 • A2):

02 • 63 = 00000010 • 01100011 = 11000110

03 • 2F = (02 • 2F) $\oplus$ 2F = (00000010 • 00101111) $\oplus$ 00101111 = 01110001

01 • AF = AF = 10101111 dan 01 • A2 = A2 = 10100010 (2.7)

Jadi 11000110

01110001

10101111

10100010 $\oplus$

10111010

Setelah semua proses tranformasi dilakukan, maka tranformasi *AddRoundKey* dilakukan lagi untuk mendapatkan keluaran untuk ronde pertama dengan melakukan operasi XOR antara *state* matrix dari operasi *MixColumns* dengan *roundkey* nomor satu.

$$\begin{pmatrix} BA & 84 & E8 & 1B \\ 75 & A4 & 8D & 40 \\ F4 & 8D & 06 & 7D \\ 7A & 32 & 0E & 5D \end{pmatrix} \oplus \begin{pmatrix} E2 & 91 & B1 & D6 \\ 32 & 12 & 59 & 79 \\ FC & 91 & E4 & A2 \\ F1 & 88 & E6 & 93 \end{pmatrix} = \begin{pmatrix} 58 & 15 & 59 & CD \\ 47 & B6 & D4 & 39 \\ 08 & 1C & E2 & DF \\ 8B & BA & E8 & CE \end{pmatrix}$$

Didapatkan hasil keluaran dari ronde pertama 58 47 08 8B 15 B6 1C BA 59 D4 E2 E8 CD 39 DF CE.

2.4.6 Proses Pembuatan *Chipertext*

Semua proses transformasi sebelumnya diulangi sebanyak ronde yang diimplementasikan untuk menghasilkan *chipertext* yang dimana dilakukan transformasi *SubBytes, ShiftRows, MixColumns* dan *AddRoundKey* dimana pada ronde ke sepuluh transfromasi *MixColumns* tidak dilakukan. Berikut merupakan proses pembuatan *chipertext* dari transformasi pada tiap rondenya.

- *Round 2*
- Setelah operasi *SubsByte:* $\begin{pmatrix} 6A & 59 & CB & BD \\ A0 & 4E & 48 & 12 \\ 30 & 9C & 98 & 9E \\ 3D & F4 & 9B & 8B \end{pmatrix}$
- Setelah operasi *ShiftRows:* $\begin{pmatrix} 6A & 59 & CB & BD \\ 4E & 48 & 12 & 40 \\ 98 & 9E & 30 & 9B \\ 8B & 3D & F4 & 9B \end{pmatrix}$
- Setelah operasi *MixColumns:* $\begin{pmatrix} 15 & C9 & 7F & 9D \\ CE & 4D & 4B & C2 \\ 89 & 71 & BE & 88 \\ 65 & 47 & 97 & CD \end{pmatrix}$
- Setelah operasi *AddRoundKey:* $\begin{pmatrix} 43 & 0E & 09 & 3D \\ C6 & 57 & 08 & F8 \\ A9 & C0 & EB & 7F \\ 62 & C8 & FE & 37 \end{pmatrix}$
- *Round 3*
- Setelah operasi *SubsByte:* $\begin{pmatrix} 1A & AB & 01 & 27 \\ B4 & 5B & 30 & 41 \\ D3 & BA & E9 & D2 \\ AA & E8 & BB & 9A \end{pmatrix}$
- Setelah operasi *ShiftRows:* $\begin{pmatrix} 1A & AB & 01 & 27 \\ 5B & 30 & 41 & B4 \\ E9 & D2 & D3 & BA \\ A9 & AA & E8 & BB \end{pmatrix}$
- Setelah operasi *MixColumns:* $\begin{pmatrix} AA & 65 & FA & 88 \\ 16 & 0C & 05 & 3A \\ 3D & C1 & DE & 2A \\ B3 & 4B & 5A & 0A \end{pmatrix}$

- Setelah operasi *AddRoundKey:* $\begin{pmatrix} 78 & 70 & 99 & 4B \\ 76 & 76 & 3C & 39 \\ 30 & 7D & 37 & 34 \\ 54 & 23 & 5B & F1 \end{pmatrix}$

- *Round 4*

- Setelah operasi *SubsByte:* $\begin{pmatrix} BC & 51 & EE & B3 \\ 38 & 38 & EB & 12 \\ 04 & FF & 9A & 18 \\ 20 & 26 & 39 & A1 \end{pmatrix}$

- Setelah operasi *ShiftRows:* $\begin{pmatrix} BC & 51 & EE & B3 \\ 38 & EB & 12 & 38 \\ 9A & 18 & 04 & FF \\ A1 & 20 & 26 & 39 \end{pmatrix}$

- Setelah operasi *MixColumns:* $\begin{pmatrix} 10 & BC & D3 & F3 \\ D8 & 94 & E0 & E0 \\ 53 & EA & 9E & 25 \\ 24 & 40 & 73 & 7B \end{pmatrix}$

- Setelah operasi *AddRoundKey:* $\begin{pmatrix} B1 & 08 & 04 & E7 \\ CA & FC & B1 & B2 \\ 51 & 54 & C9 & 6C \\ ED & E1 & D3 & 20 \end{pmatrix}$

- *Round 5*

- Setelah operasi *SubsByte:* $\begin{pmatrix} C8 & 30 & F2 & 94 \\ 74 & B0 & C8 & 37 \\ D1 & 20 & DD & 50 \\ 55 & F8 & 66 & B7 \end{pmatrix}$

- Setelah operasi *ShiftRows:* $\begin{pmatrix} C8 & 30 & F2 & 94 \\ B0 & C8 & 37 & 74 \\ DD & 50 & D1 & 20 \\ B7 & 55 & F8 & 66 \end{pmatrix}$

- Setelah operasi *MixColumns:* $\begin{pmatrix} 2A & 26 & 8F & E9 \\ 78 & 1E & 0C & 7A \\ 1B & A7 & 6F & 0A \\ 5B & 62 & 00 & 3F \end{pmatrix}$

- Setelah operasi *AddRoundKey:* $\begin{pmatrix} 9B & 23 & 5D & 2F \\ 51 & 5F & 1C & 38 \\ 20 & 22 & BD & 91 \\ 68 & F0 & 32 & 56 \end{pmatrix}$

- *Round 6*

- Setelah operasi *SubsByte:* $\begin{pmatrix} 14 & 26 & 4C & 15 \\ D1 & CF & 9C & 07 \\ B7 & 93 & 7A & 81 \\ 45 & 8C & 23 & B1 \end{pmatrix}$

- Setelah operasi *ShiftRows:* $\begin{pmatrix} 14 & 26 & 4C & 15 \\ CF & 9C & 07 & D1 \\ 7A & 81 & B7 & 93 \\ B1 & 45 & 8C & 23 \end{pmatrix}$
- Setelah operasi *MixColumns:* $\begin{pmatrix} A9 & 37 & AA & F2 \\ AE & D8 & 0C & 21 \\ E7 & 6C & B1 & 9C \\ F0 & FD & 67 & 3B \end{pmatrix}$
- Setelah operasi *AddRoundKey:* $\begin{pmatrix} 14 & 8F & C0 & 5E \\ 93 & A4 & 60 & 0F \\ 25 & 2B & 24 & 92 \\ 77 & E8 & 40 & 75 \end{pmatrix}$
- *Round 7*
- Setelah operasi *SubsByte:* $\begin{pmatrix} FA & 73 & BA & 58 \\ DC & 49 & D0 & 76 \\ 3F & F1 & 36 & 4F \\ F5 & 9B & 09 & 9D \end{pmatrix}$
- Setelah operasi *ShiftRows:* $\begin{pmatrix} FA & 73 & BA & 58 \\ 49 & D0 & 76 & DC \\ 36 & 4F & 3F & F1 \\ 9D & F5 & 9B & 09 \end{pmatrix}$
- Setelah operasi *MixColumns:* $\begin{pmatrix} 9F & 37 & 51 & 37 \\ AF & EC & 8C & FA \\ 63 & 39 & 04 & 66 \\ 4B & FB & B1 & D7 \end{pmatrix}$
- Setelah operasi *AddRoundKey:* $\begin{pmatrix} 53 & 43 & 4F & 85 \\ 39 & 06 & 0A & 52 \\ 8E & 93 & 3B & 57 \\ 5D & F8 & 95 & BD \end{pmatrix}$
- *Round 8*
- Setelah operasi *SubsByte:* $\begin{pmatrix} ED & 1A & 84 & 97 \\ 74 & 6F & 67 & 00 \\ D3 & DC & E2 & 5B \\ 55 & 41 & 2A & 7A \end{pmatrix}$
- Setelah operasi *ShiftRows:* $\begin{pmatrix} ED & 1A & 84 & 97 \\ 6F & 67 & 00 & 12 \\ E2 & 5B & 19 & DC \\ 7A & 4C & 41 & 2A \end{pmatrix}$
- Setelah operasi *MixColumns:* $\begin{pmatrix} E8 & 8A & 4B & F5 \\ 74 & 75 & EE & E6 \\ D3 & 1F & 75 & 58 \\ 55 & 8A & 0C & 38 \end{pmatrix}$
- Setelah operasi *AddRoundKey:* $\begin{pmatrix} 66 & 70 & AF & A3 \\ 25 & CE & D3 & 73 \\ 3C & 5A & 0F & 13 \\ 74 & A8 & 0A & 54 \end{pmatrix}$

- *Round 9*
- Setelah operasi *SubsByte:* $\begin{pmatrix} 33 & 51 & 79 & 0A \\ 3F & 8B & 66 & 8F \\ EB & BE & 76 & 7D \\ 92 & C2 & 67 & 20 \end{pmatrix}$
- Setelah operasi *ShiftRows:* $\begin{pmatrix} 33 & 51 & 79 & 0A \\ 8B & 66 & 8F & 3F \\ 76 & 7D & EB & BE \\ 20 & 92 & C2 & 67 \end{pmatrix}$
- Setelah operasi *MixColumns:* $\begin{pmatrix} B6 & E7 & 51 & 8C \\ 84 & 88 & 98 & CA \\ 34 & 60 & 66 & FB \\ E8 & D7 & 70 & 51 \end{pmatrix}$
- Setelah operasi *AddRoundKey:* $\begin{pmatrix} 09 & A2 & F0 & 7B \\ 66 & D1 & FC & 3B \\ 8B & 9A & E6 & 30 \\ 78 & 65 & C4 & 89 \end{pmatrix}$
- *Round 10*
- Setelah operasi *SubsByte:* $\begin{pmatrix} 01 & 3A & 8C & 21 \\ 33 & 3E & B0 & E2 \\ 3D & B8 & 8E & 04 \\ BC & 4D & 1C & A7 \end{pmatrix}$
- Setelah operasi *ShiftRows:* $\begin{pmatrix} 01 & 3A & 8C & 21 \\ 3E & B0 & E2 & 33 \\ 8E & 04 & 3D & B8 \\ A7 & BC & 4D & 1C \end{pmatrix}$
- Operasi *MixColumns* tidak dilakukan pada *round* terakhir.
- Setelah operasi *AddRoundKey:* $\begin{pmatrix} 29 & 57 & 40 & 1A \\ C3 & 14 & 22 & 02 \\ 50 & 20 & 99 & D7 \\ 5F & F6 & B3 & 3A \end{pmatrix}$

Setelah semua proses dilakukan pada sepuluh ronde dihasilkan *chipertext* yaitu 29 C3 50 5F 57 14 20 F6 40 22 99 B3 1A 02 D7 3A.

2.5 Sensor

Sensor adalah suatu peralatan yang berfungsi mendeteksi gejala-gejala atau sinyal-sinyal yang berasal dari perubahan suatu energi listrik, energi fisika, energi kimia, energi bilogi, energi mekanik dan sebagainya (D Sharon dkk, 1982).

2.5.1 DHT-11

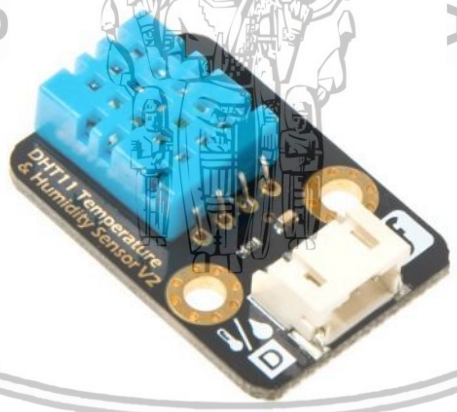
Sensor suhu dan kelembaban merupakan perangkat elektronika yang berguna untuk mengukur besaran suhu pada satuan celsius ($_o$C) dan kelembapan dalam satuan persen (%). Sensor suhu dan kelembaban biasanya digunakan untuk keperluan sistem *monitoring* ataupun sistem antisipasi terjadinya bencana

di dalam atau di luar ruangan. Salah satu jenis sensor suhu dan kelembaban adalah DHT11. Sensor DHT11 merupakan sensor dijital dengan tingkat stabilitas yang sangat baik serta fitur kalibrasi yang sangat akurat. Sensor DHT11 juga menyediakan *library* khusus yang bernama DHT *Library*, *library* tersebut berguna untuk memudahkan pengguna memprogram di mikrokontroler. Berikut adalah spesifikasi sensor DHT11 dan Gambar 2.6 merupakan gambar sensor DHT11. (Akhmalia, 2016)

Tabel 2.6 Tabel Spesifikasi Sensor DHT11

Power Suplay	3-5 v DC
Rentang pengukuran	Kelembapan (20% – 90% RH) Suhu (0-50 Celcius)
Akurasi	Kelembapan (4 RH) Suhu (2.0 Celcius)
Sensitivitas	Kelembapan (1% RH) Suhu(0.1 Celcius)
Periode *Sensing*	Rata-rata: 2 s

Sumber: (Manaloe, 2015)



Gambar 2.10 Sensor Suhu DHT11

Sumber: (Manaloe, 2015)

2.6 NI Labview

LabVIEW adalah sebuah perangkat lunak pemograman yang diproduksi oleh National Instruments dengan konsep yang berbeda. Seperti bahasa pemograman lainnya yaitu C++, Matlab atau Visual basic , LabVIEW juga mempunyai fungsi dan peranan yang sama, perbedaannya bahwa LabVIEW menggunakan bahasa pemrograman berbasis grafis atau blok diagram sementara bahasa pemrograman lainnya menggunakan basis *text*. Program LabVIEW dikenal dengan sebutan VI atau *Virtual instruments* karena penampilan dan operasinya dapat meniru sebuah

instrument. Pada LabVIEW, user pertama-tama membuat user interface atau front panel dengan menggunakan control dan indikator, yang dimaksud dengan kontrol adalah *knobs*, *push buttons*, *dials* dan peralatan *input* lainnya sedangkan yang dimaksud dengan indikator adalah *graphs*, LEDs dan peralatan *display* lainnya. Setelah menyusun antarmuka pengguna, lalu pengguna menyusun blok diagram yang berisi kode-kode VIs untuk mengontrol front panel (Wibowo, 2010). Perangkat lunak LabVIEW terdiri dari tiga kompone utama yaitu :

1. *Front Panel*

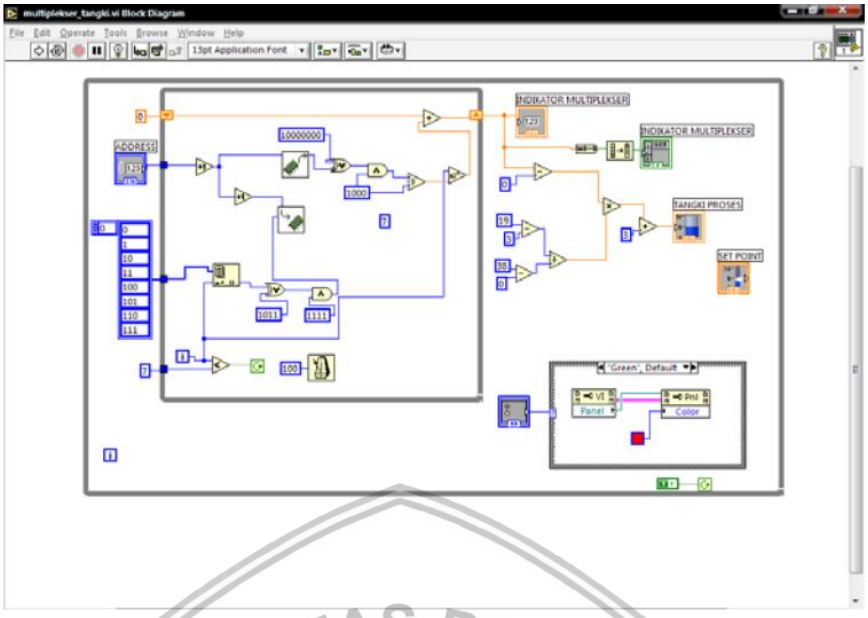
Front panel adalah bagian *window* yang berlatar belakang abu-abu serta mengandung kontrol dan indikator. *Front panel* digunakan untuk membangun sebuah VI, menjalankan program dan mendebug program. Tampilan dari *front panel* dapat dilihat pada Gambar 2.7.

Gambar 2.11 Tampilan *Front Panel*

Sumber: (Wibowo, 2010)

2. Blok Diagram dari *Virtual Instrument*

   Blok diagram adalah bagian *window* yang berlatar belakang putih berisi *source code* yang dibuat dan berfungsi sebagai instruksi untuk *front panel*. Tampilan dari blok diagram dapat dilihat pada Gambar 2.8.



Gambar 2.12 Tampilan Blok Diagram

Sumber: (Wibowo, 2010)

3. *Controls Pallete*

   Control Pallete merupakan tempat beberapa kontrol dan indikator *pada front panel*, *control pallete* hanya tersedia di *front panel*, untuk menampilkan *control pallete* dapat dilakukan dengan **mengkilk windows >> show control pallete** atau klik kanan pada *front panel*. Contoh *control pallete* ditunjukkan pada Gambar 2.9.

Gambar 2.13 Tampilan *Clontrol Pallete*

4. *Functions Pallete*

Functions Pallete di gunakan untuk membangun sebuah blok diagram, *functions pallete* hanya tersedia pada blok diagram, untuk menampilkannya dapat dilakukan

dengan **mengklik *windows* >> *show control pallete*** atau klik kanan pada lembar kerja blok diagram. Contoh dari *functions pallete* ditunjukkan pada Gambar 2.10.

Gambar 2.14 Tampilan *Functions Pallete*

Sumber: (Wibowo, 2010)

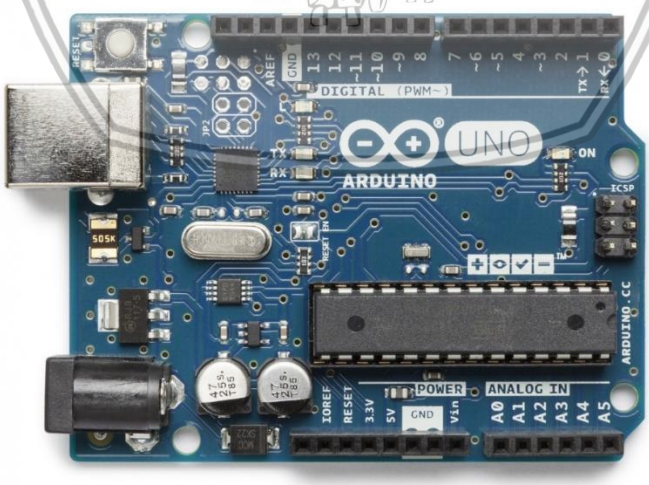
2.7 Arduino

Arduino merupakan sebuah *platform* komputasi fisik yang bersifat *open source* pada *board* masukan dan keluaran sederhana. *Platform* komputasi merupakan sistem fisik yang interaktif dengan penggunaan perangkat lunak dan perangkat keras yang dapat mendeteksi dan merespon situasi dan kondisi yang ada didunia nyata. Nama Arduino tidak hanya digunakan untuk menamai *board* rangkaiannnya saja tetapi juga untuk menamai bahasa dan perangkat lunak pemrogramannya, serta lingkungan pemrogramannya atau IDE (*Integrated Development Environment*). Ada beberapa jenis modul Arduino yang biasa digunakan, pada penelitian ini menggunakan *board* Arduino Uno sebagai mikrokontroler yang menghubungkan dari perangkat keras ke antarmuka komputer. Jenis-jenis dari Arduino sangatlah banyak salah satunya Arduino Uno. Arduino Uno adalah piranti mikrokontroler yang menggunakan ATmega328, merupakan penerus Arduino Duemilanove. Arduino Uno memiliki 14 pin I/O dijital di mana 6 pin *input* tersebut dapat digunakan sebagai *output* PWM dan 6 pin *input* analog, 16 MHz *osilator* kristal, koneksi USB, *jack power*, *ICSP header*, dan tombol *reset*. Arduino juga mempunyai kompiler sendiri, bahasa pemrograman yang digunakan adalah C/C++ tetapi sudah menggunakan konsep pemrograman berbasis objek atau biasa disebut *Object Oriented Programing* (OOP). *Compiler* bersifat gratis, dan dapat diunduh di *website* arduino.cc. Kelebihan lain dari *compiler* arduino ini adalah dia bersifat *cross-platform* atau dapat berjalan di semua sistem operasi, sehingga walaupun pengguna Windows, Linux, ataupun Macintos bisa menggunakan perangkat ini. Kelebihan Arduino dari *platform* perangkat keras mikrokontroler lain adalah:

1. IDE Arduino merupakan *multiplatform*, yang dapat dijalankan di berbagai sistem operasi, seperti windows, dan linux.
2. Pemrograman Arduino menggunakan kabel yang terhubung dengan *port* USB, bukan *port serial*.
3. Arduino adalah perangkat keras dan perangkat lunak *open source*, pembaca bisa mengunduh perangkat lunak dan gambar rangkaian arduino tanpa harus membayar ke pembuat arduino.
4. Tidak perlu perangkat *chip* programmer karena di dalamnya sudah ada *bootloader* yang akan menangani *upload* program dari komputer.
5. Sudah memiliki sarana komunikasi USB, sehingga pengguna laptop yang tidak memiliki *port* serial/RS323 bisa menggunakannya.
6. Bahasa pemrograman relatif mudah karena perangkat lunak Arduino di lengkapi dengan kumpulan *library* yang cukup lengkap.
7. Memiliki modul siap pakai (*shield*) yang bisa ditancapkan pada *board* Arduino. Misalnya *shield* GPS, Ethernet, SD Card, dll. (Hartari, 2014)

2.7.1 Arduino UNO

Arduino Uno adalah *board* mikrokontroler berbasis ATmega328. Memiliki 14 pin I/O di mana 6 pin *input* tersebut dapat digunakan sebagai *output* PWM dan 6 pin *input* analog, 16 MHz *osilator* kristal, koneksi USB, *jack power*, *ICSP header*, dan tombol *reset* . Untuk mendukung mikrokontroler agar dapat digunakan, cukup hanya dengan menghubungkan *board* Arduino Uno ke komputer dengan menggunakan kabel USB. Arduino Uno berbeda dengan semua *board* sebelumnya dalam hal koneksi USB *to serial* yaitu menggunakan fitur Atmega8U2 yang diprogram sebagai konverter USB *to serial* berbeda dengan *board* sebelumnya yang menggunakan *chip* FTDI *driver* USB *to serial*.



Gambar 2.15 Arduino UNO

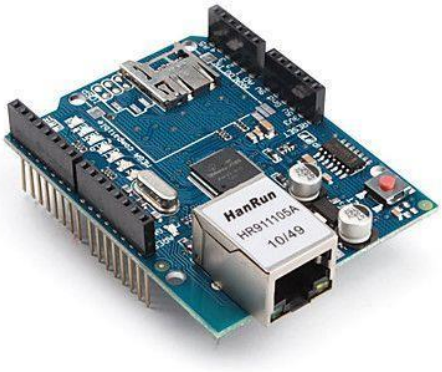
Sumber: (Manaloe, 2015)

Adapun spesifikasi modul Arduino Uno adalah sebagai berikut:

1. Daya Modul Arduino Uno dapat diaktifkan melalui koneksi USB atau dengan catu daya eksternal (otomatis). Eksternal (non-USB) daya dapat berasal dari AC ke adaptor DC atau baterai. Adaptor ini dapat dihubungkan dengan menancapkan *plug jack* pusat-positif ukuran 2.1mm konektor "*POWER"*. Ujung kepala dari baterai dapat dimasukkan ke dalam Gnd dan Vin pin *header* dari konektor "*POWER"*. Kisaran kebutuhan daya yang disarankan untuk *board* Arduino Uno adalah 7 sampai dengan 12 volt, jika diberi daya kurang dari 7 volt kemungkinan pin 5v Arduino Uno dapat beroperasi tetapi tidak stabil kemudian jika diberi daya lebih dari 12V, regulator tegangan bisa panas dan dapat merusak *board* Arduino Uno.
2. Memori ATmega328 memiliki 32 KB (dengan 0,5 KB digunakan untuk *bootloader*), 2 KB dari *SRAM* dan 1 KB *EEPROM*.
3. *Input* dan *Output* masing-masing dari 14 pin dijital modul Arduino Uno dapat digunakan sebagai *input* atau *output*, dengan menggunakan fungsi *pinMode*(), *dijitalWrite*(), dan *dijitalRead*(), beroperasi dengan daya 5 volt. Setiap pin dapat memberikan atau menerima maksimum 40 mA dan memiliki *internal pull-up resistor* (secara default terputus) dari 20-50 kΩ. Selain itu, beberapa pin memiliki fungsi khusus seperti serial 0 (RX) dan 1 (TX). Digunakan untuk menerima (RX) dan mengirimkan (TX) TTL data *serial*. 13 Pin ini dihubungkan ke pin yang berkaitan dengan *chip* serial ATmega8U2 *USB-to-TTL*.

2.8 Ethernet Shield

Ethernet Shield menambah kemampuan Arduino *board* agar terhubung ke jaringan komputer. Ethernet shield berbasiskan *chip* Ethernet Wiznet W5100. *Ethernet library* digunakan dalam menulis program agar Arduino *board* dapat terhubung ke jaringan dengan menggunakan Arduino Ethernet Shield. Arduino *board* berkomunikasi dengan W5100 SPI (*Serial Peripheral Interface*). Komunikasi ini di atur oleh *library SPI.h* dan *Ethernet.h*. *Bus* SPI menggunakan pin dijital 11, 12 dan 13 pada Arduino Uno. Pin dijital 10 digunakan untuk memilih *chip* W5100. Pin-pin yang sudah disebutkan sebelumnya tidak dapat digunakan untuk *input*/*output* umum ketika kita menggunakan Ethernet Shield . DFRduino Ethernet Shield adalah sebuah *clone* dari Arduino Ethernet Shield yang dibuat oleh DFRobot. (Irwan Dinata, 2015).



Gambar 2.16 Ethernet Shield

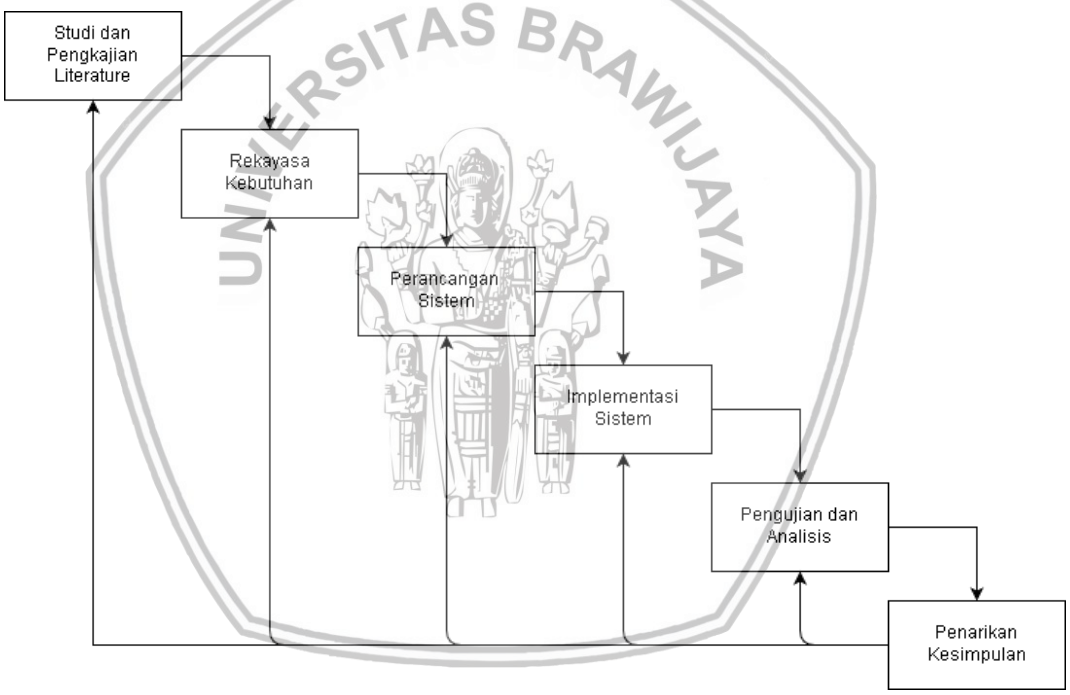
Sumber: (Manaloe, 2015)

BAB 3 METODOLOGI

Bab ini akan menjelaskan metode yang digunakan untuk melakukan penelitian. Tipe penelitian ini adalah implementatif yaitu bersifat observasi menggunakan rancang bangun sederhana.

3.1 Metode Penelitian

Langkah awal yang dilakukan adalah mengumpulkan teori-teori pendukung dan menjelaskan secara singkat pada studi literatur. Kemudian, proses selanjutnya adalah melakukan pengumpulan data. Setelah data dikumpulkan sistem didesain, lalu perangkat keras dan perangkat lunak diimplementasikan sesuai dengan perancangan. Kemudian dilakukan pengujian dan analisis pada perancangan yang telah dibuat. Kesimpulan dan saran dimuat sebagai catatan atas perancangan dan kemungkinan arah pengembangan selanjutnya. Langkah-langkah dalam penelitian ini ditunjukan pada Gambar 3.1 .



Gambar 3.1 Diagram Alir Penelitian

3.2 Studi Literatur

Studi literatur yang digunakan untuk menambah studi pustaka dan pengetahuan yang akan dilakukan untuk mengerjakan penulisan laporan dan penelitian. Studi literatur didapat dengan mengumpulkan teori dan pustaka yang berkaitan dengan penelitian ini. Hal yang perlu dijelaskan dalam metodologi penelitian adalah:

1. Protokol Komunikasi HTTP (*HyperText Transfer Protokol)*
2. Sensor suhu dan kelembapan DHT11

3. Mikrokontroler Arduino Uno
4. Perangkat lunak Pemrograman Arduino IDE
5. Algoritme Enkripsi AES
6. Pemrograman NI Labview

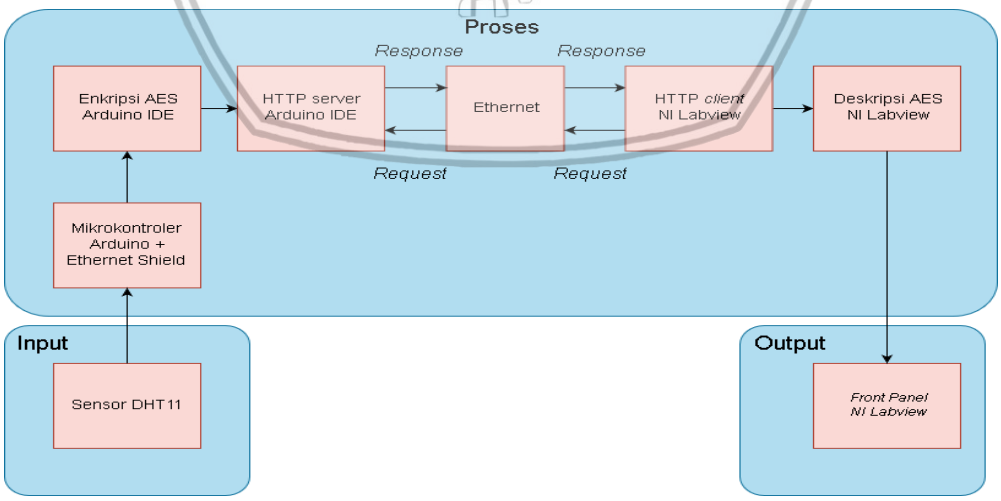
3.3 Rekayasa Kebutuhan

Dalam penelitian ini kebutuhan utama yang dibutuhkan meliputi perangkat keras dan perangkat lunak. Perangkat keras yang digunakan pada penelitian ini adalah seperangkat laptop atau sejenisnya yang memiliki spesifikasi sebagai berikut:

1. Prosesor : Intel Core i3-4005U CPU @ 1.70GHz
2. RAM : 4 GB
3. Sistem Operasi : Windows 10
4. Sensor Suhu dan Kelembapan DHT 11
5. Mikrokontroler Arduino Uno
6. Pemrograman NI Labview
7. Algoritme Enkripsi AES

3.4 Perancangan Sistem

Pada perancangan sistem ini menjelaskan tentang perangkat yang digunakan berserta spesifikasinya serta alur komunikasi sistem hingga sistem dapat berjalan. Model perancangan sistem menjelaskan mengenai cara kerja sistem secara terstruktur. Alur kerja sistem dapat dilihat pada Gambar 3.2.



Gambar 3.2 Diagram Blok Rancangan Sistem

Berdasarkan Gambar 3.2 di atas perancangan sistem yang akan dibuat adalah sensor DHT11 sebagai *node* sensor akan melakukan *sensing* dan mendapatkan data dari hasil *sensing* lalu dikirimkan ke mikrokontroler Arduino Uno untuk pengolahan datanya. Hasil data *sensing* yang diproses Arduino akan dikirimkan ke server protokol HTTP, data yang dikirimkan oleh protokol HTTP ke HTTP *client* akan dienkripsi dulu menggunakan algoritme AES. Data yang ada pada HTTP server berupa *chipertext* dari data *sensing* DHT11 yang dienkripsi oleh algoritme AES. NI Labview di sini berperan sebagai HTTP *client* yang di mana sudah terdapat deskripsi algoritme AES. Proses pengiriman data dari server ke *client* yaitu saat *client* mengirim *request* ke HTTP server setelah itu HTTP server mengolah permintaan *client*, lalu HTTP server membalas dengan pesan *response* berupa data yang diminta oleh *client* yaitu data *sensing* DHT11 yang sudah dienkripsi. Setelah menerima data dari HTTP server, *client* yaitu NI Labview akan mendeskripsikan terlebih dahulu data *sensing* DHT11 yang sudah dienkripsi oleh algoritme AES sebelum ditampilkan pada *output front panel* pada NI Labview.

Implementasi dilakukan berdasarkan perancangan sistem yang dijelaskan pada Gambar 3.2 langkah yang harus dilakukan yaitu:

3.4.1 Konfigurasi Perangkat Keras

Pada tahapan konfigurasi perangkat keras, yang harus dilakukan adalah dengan merancang sensor DHT11 untuk disambungkan ke mikrokontroler Arduino Uno. Sensor akan disambungkan melalui kabel jumper *male to female* dan *male to male*. Penyambungan dilakukan berdasarkan *datasheet* masing-masing perangkat keras tersebut.

3.4.2 Konfigurasi Perangkat Lunak

Pada tahap konfigurasi perangkat lunak dilakukan instalasi perangkat lunak NI Labview yang dapat diunduh dari *website* resmi National Instrument dan Arduino IDE yang juga dapat diunduh dari *website* resmi Arduino Lab. Setelah itu, dilakukan pemrograman untuk server HTTP dan algoritme enkripsi AES menggunakan Arduino IDE dan pemrograman pengambilan data *sensing* dari sensor DHT11 lalu diimplementasikan ke mikrokontroler Arduino Uno, lalu pemrograman *client* HTTP pada perangkat lunak NI Labview dan algoritme deskripsi AES beserta *output interface*.

3.5 Pengujian dan Analisis Hasil

Pengujian dilakukan dengan pengiriman data *sensing* yaitu pengukuran suhu dan kelembapan dari HTTP server dengan data yang sudah dienkripsi oleh algoritme AES, lalu *client* meminta data ke HTTP server dan HTTP server mengirim data *sensing* berupa *chipertext* ke *client* yaitu NI Labview di mana *chipertext* akan dideskripsi oleh NI Labview lalu hasil deskripsi akan ditampilkan pada *output* di *front panel* NI Labview. Berdasarkan hasil dari pengujian akan dilakukan analisis dari tingkat *error rate* enkripsi dan deskripsi, waktu pemrosesan dan *delay* pengiriman protokol HTTP.

3.6 Kesimpulan

Pada tahap pengambilan kesimpulan dapat dilakukan setelah semua tahap pada bab 3 ini selesai. Kesimpulan didapat berdasarkan pada hasil pengujian dan analisis terhadap sistem yang dibangun oleh penulis. Kesimpulan yang dibuat diharapkan menjadi referensi untuk pengembangan dan penyempurnaan sistem.

BAB 4 REKAYASA KEBUTUHAN SISTEM

4.1 Deskripsi Umum

Sistem yang dibangun pada penelitian ini yaitu mengimplementasikan algoritme enkripsi AES pada pengiriman data sensor DHT11 menggunakan protokol komunikasi *Hyper Text Transfer Protokol* (HTTP) pada perangkat lunak National Instrument Labview dan Arduino. Penelitian ini menggunakan Arduino Uno sebagai mikrokontroler dan sensor suhu dan kelembapan DHT11 sebagai data masukan. Sensor DHT11 dipilih sebagai sumber data dari sistem ini dimana pada sensor DHT11 data yang dihasilkan sudah dikonversi kedalam bentuk dijital yang tidak memerlukan lagi konversi secara manual dari analog ke dijital. Algoritme AES juga memerlukan masukkan yang mana pada sistem ini masukkan dari algoritme AES dibuat secara otomatis atau tidak perlu dimasukkan secara manual oleh peneliti. DHT11 akan melakukan *sensing* dan data dari hasil *sensing* DHT11 akan dikirimkan ke server protokol HTTP. Mikrokontroler Arduino Uno digunakan karena kebutuhan akan pengolahan dari data sensor DHT11 untuk dikirimkan ke HTTP server, pada Arduino Uno akan ditambahkan dengan modul Ethernet Shield sebagai penyedia jalur komunikasi berupa LAN (*Local Area Network*). Pada Arduino Nano atau Promini tidak menyediakan pin tambahan untuk menyambungkan Ethernet Shield ke papan Arduino sedangkan pada Arduino Uno ada. Protokol HTTP sudah diprogram dengan Arduino Uno dan NI Labview yang dijalankan pada mikrokontroler Arduino Uno di mana data yang sudah diambil dari sensor DHT11 akan dikirimkan ke server HTTP dan server akan mengirimkan data ke *client* ketika *client* mengirim *request* ke server HTTP. Algoritme keamanan AES akan mengamankan data yang dikirimkan oleh server HTTP ke *client* HTTP di mana data tersebut yang berupa *plaintext* akan dienkripsi menjadi *chipertext* sebelum dikirimkan ke *client* HTTP dan ketika data sudah terkirim ke *client* HTTP maka data tersebut akan dideskripsi terlebih dahulu agar *client* bisa membaca data tersebut. Sistem ini membutuhkan beberapa spesifikasi kebutuhan agar sistem dapat berjalan sesuai dengan yang diharapkan.

4.2 Kebutuhan Sistem

Pada bagian ini dijelaskan tentang apa saja yang dibutuhkan oleh sistem untuk penelitian yang dilakukan, sehingga dapat dilakukan sesuai dengan yang diharapkan. Kebutuhan sistem dibagi menjadi beberapa bagian yaitu kebutuhan fungsional, kebutuhan perangkat keras dan kebutuhan perangkat lunak. Kebutuhan sistem dapat mempermudah proses perancangan dan implementasi.

4.2.1 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan tentang apa saja yang dapat dilakukan oleh sistem, informasi apa saja yang harus ada dan bisa dihasilkan oleh sistem. Berikut ini adalah kebutuhan fungsional dari sistem ini, yaitu:

1. Sistem dapat mengimplementasikan algoritme AES pada pengiriman data sensor DHTT menggunakan protokol HTTP.
2. Sistem dapat menjalankan HTTP server pada Arduino IDE dan HTTP *client* pada NI Labview yang diolah oleh mikrokontroler Arduino Uno dengan menggunakan jalur komunikasi Ethernet (LAN).
3. Sistem dapat menjalankan protokol HTTP di mana HTTP *client* melakukan *request* ke HTTP server dan HTTP server akan membalas pesan *request* dengan pesan *response* yang berisikan data sensor.
4. Sistem dapat mengenkripsi data sensor DHT11 yang akan dikirimkan oleh HTTP server ke HTTP *client* dan data yang diterima oleh HTTP *client* dapat dideskripsi dan ditampilkan pada *output* sistem.

4.2.2 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yatu spesifikasi minimal komputer yang digunakan untuk implementasi sistem. Komputer/laptop dengan spesifikasi:

1. CPU : 4th Generation Intel® Core™ i3-4005U *processor* (3M *Cache*, 1.7 GHz)
2. *Operating System* : Windows 10 (64bit)
3. *Video Card* : NVIDIA® GeForce® 820M 1GB DDR3L
4. *Hard Drive* : 500 GB3 (standard*) with options up to* 1 TB3 *and* HHD *options up to* 1 TB3 / 8 GB *Cache*
5. *Display* : 14.0*" LED Backlit Display with Truelife and HD resolution* (1366 x 768)

Arduino Uno dengan spesifikasi sebagai berikut:

1. Mikrokontroler : ATmega328
2. *Operating Voltage* : 5V
3. *Input Voltage* (disarankan) : 7-12V
4. *Input Voltage* (batas) : 6-20V
5. Dijital I / O : Pins 14 (di mana 6 memberikan *output* PWM)
6. Analog *Input* : Pins 6
7. DC *Current* per I / O : Pin 40 mA
8. DC *Current* for 3.3V : Pin 50 mA
9. *Flash Memory* : 32 KB (ATmega328) yang 0,5 KB digunakan oleh *bootloader*
10. *SRAM* : 2 KB (ATmega328)
11. *EEPROM* : 1 KB (ATmega328)
12. *Clock Speed* : 16 MHz

1. *Input Voltage*
   Arduino Uno ini dapat beroperasi pada tegangan eksternal dari 6-20 volt. Jika diberikan tegangan kurang dari 7V, maka arduino ini mungkin akan

menjadi tidak stabil. Jika menggunakan lebih dari 12V, regulator *voltage* bisa panas dan merusak *board*. Rentang yang dianjurkan adalah 7-12 volt.

2. Memori
   ATmega328 ini memiliki memori sebesar 32 KB (0,5 KB dari memori tersebut digunakan untuk *bootloader*) dan juga memiliki memori sebesar 2 KB dari *SRAM* dan 1 KB dari *EEPROM*.
3. *Input* dan *Output*
   Masing-masing dari 14 pin dijital pada Uno dapat digunakan sebagai *input* atau *output*, dengan menggunakan fungsi pinMode(), dijitalWrite(), dan dijitalRead(). Setiap pin dapat memberikan atau menerima maksimum 40 mA dan memiliki resistor *pull-up* internal (terputus secara *default*) dari 20-50 kOhms.

Ethernet Shield dengan spesifikasi:

1. IEEE802.3af *compliant*
2. Low *output* ripple and noise (100mVpp)
3. *Input* voltage *range* 36V to 57V
4. *Overload and short-circuit protection*
5. 9V *Output*
6. *High efficiency* DC/DC *converter*: typ 75% @ 50% *load*
7. 1500V *isolation* (*input* to *output*)

Shield saat ini memiliki *Power over Ethernet* (PoE) modul di rancang untuk mengambil listrik dari *twisted pair* Kategori 5 kabel Ethernet konvensional.

4.2.3 Kebutuhan Perangkat Lunak

Kebutuhan Perangkat lunak yaitu seluruh perangkat lunak yang digunakan untuk implementasi sistem.

1. Windows 10 64 bit : Sebagai sistem operasi yang digunakan sistem.
2. NI Labview : sebagai perangkat lunak yang digunakan menjadi *client* protokol HTTP dan deskripsi algoritme AES juga menampilkan *output* sistem.
3. Arduino IDE : sebagai perangkat lunak yang digunakan untuk server protokol HTTP dan enkripsi algoritme AES dan program *sensing* sensor DHT11.

4.3 Batasan Sistem

Perancangan sistem memuat langkah-langkah perencanaan dan perancangan atas suatu sistem yang dapat memenuhi kebutuhan berdasarkan analisis kebutuhan yang telah dilakukan. Batasan-batasan sistem yang terkait pada kebutuhan sistem yaitu seperti berikut ini:

1. Sistem ini menggunakan Arduino Uno sebagai mikrokontroler.
2. Sistem ini hanya menggunakan satu sensor DHT11 sebagai *input* data dan data yang diambil adalah kelembapan dan suhu.

3. Enkripsi algoritme AES dan HTTP server akan diimplementasikan pada Arduino IDE.
4. Deskripsi algoritme AES dan HTTP *client* akan diimplementasikan pada NI Labview dan *output* sistem menggunakan *front panel* pada NI Labview

BAB 5 PERANCANGAN DAN IMPLEMENTASI

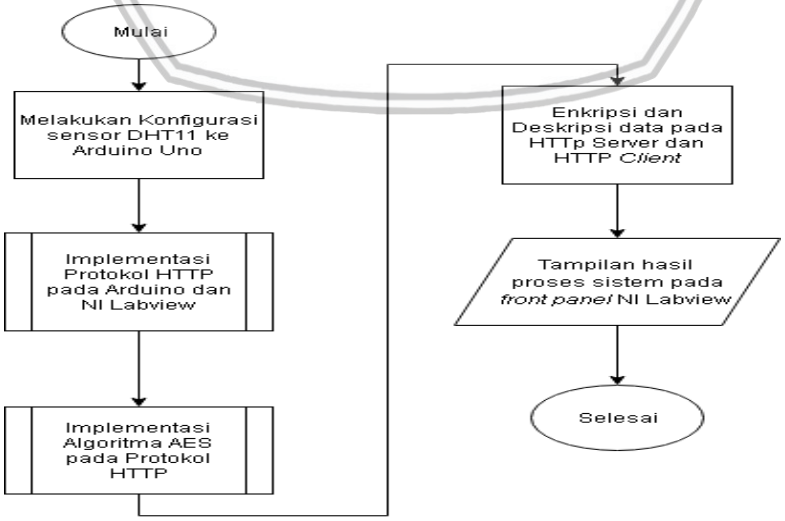
Pada bagian ini akan dibahas mengenai perancangan dan implementasi dari sistem yang akan dibuat yaitu "Implementasi Algoritme AES 128 bit pada Protokol Komunikasi HTTP". Pada perancangan terdapat perangkat lunak (*software*) dan perancangan perangkat keras (*hardware*). Sedangkan pada implementasi sistem terdapat implementasi perangkat lunak (*software*) dan implementasi perangkat keras (*hardware*).

5.1 Perancangan Sistem

Perangkat yang dirancang adalah di mana sistem menggunakan algoritme AES (*Advance Encrption* System) untuk mengenkripsi pengiriman data sensor DHT11 dengan menggunakan protokol komunikasi HTTP (*Hypertext Transport Protocol*). *Input* data yang digunakan adalah sensor suhu dan kelembapan DHT11 dengan jenis data yang diambil adalah suhu dan kelembapan. HTTP server berada pada Arduino Uno di mana sebelum dikirim ke HTTP server data *input* yaitu data sensor DHT11 akan dienkripsi oleh algoritme AES. Untuk metode pengirimannya menggunakan Ethernet yang berada pada *local network*. HTTP *client* berada pada NI Labview di mana pada *client* NI Labview data yang diambil dari HTTP server yaitu berupa *chipertext* akan dideskripsi oleh *client* menggunakan algoritme AES. *Output* dari sistem ditampilkan pada *front panel* yang terdapat pada NI Labview. Hasil dari sistem berupa data suhu dan kelembapan dari sensor DHT11 yang sudah dideskripsi.

5.1.1 Diagram Alir Sistem

Pada bagian ini akan dijelaskan perancangan sistem dimulai dengan tahapan yang tertera pada bab sebelumnya hingga pengujian yang dilakukan terhadap sistem. Berikut Gambar 5.1 diagram alir perancangan sistem.



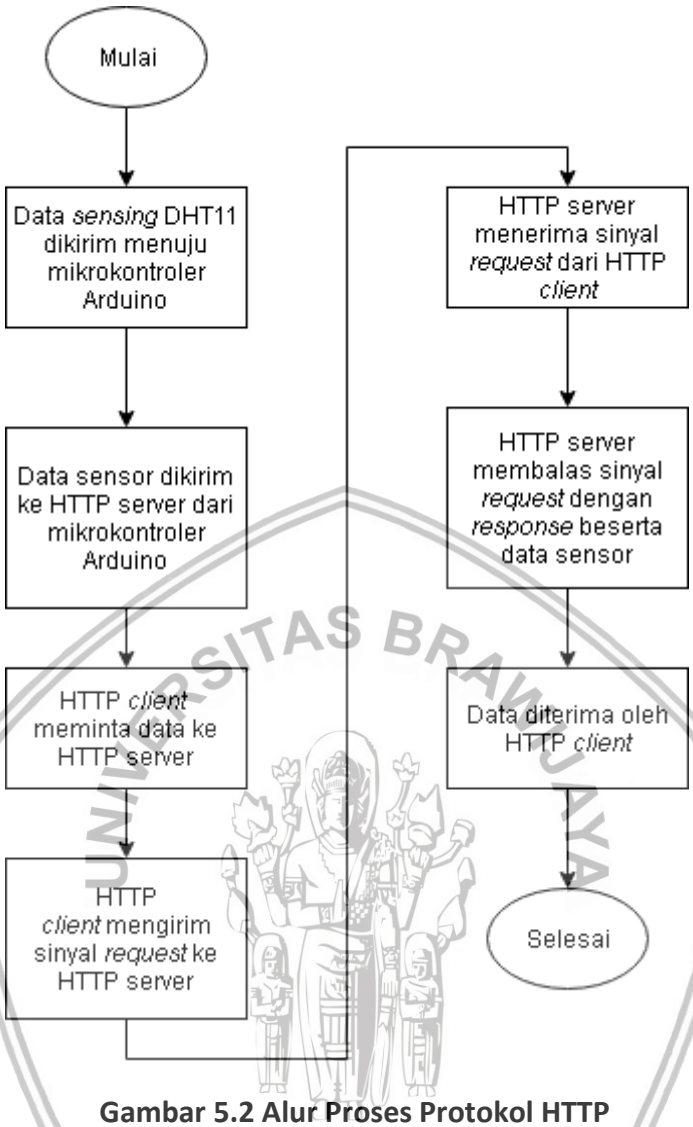
Gambar 5.1 Diagram Alir Perancangan Sistem

Berikut penjelasan pada Gambar 5.1 di atas, dapat dijelaskan alur dari sistem secara keseluruhan adalah sebagai berikut:

1. Hal pertama yang dilakukan sistem adalah konfigurasi sensor suhu dan kelembapan DHT11 pada Arduino di mana konfigurasi dilakukan agar Arduino dapat mengambil nilai dari suhu dan kelembapan yang di-*sensing* oleh sensor DHT11.
2. Implementasi protokol HTTP pada sistem akan diimplementasikan pada dua platform yang berbeda yaitu Arduino IDE dan NI Labview. Pada protokol HTTP terdapat 2 fungsi yaitu HTTP server dan HTTP *client*. Untuk HTTP server akan diimplementasikan pada Arduino IDE sedangkan untuk HTTP *client* akan diimplementasikan pada NI Labview lebih jelasnya akan dijelaskan pada kategori protokol HTTP.
3. Implementasi algoritme AES pada sistem dibagi dalam dua fungsi yaitu enkripsi (*encrypt)* dan deskripsi (*decrypt*). Fungsi enkripsi akan diimplementasikan pada HTTP server yang dikonfigurasi pada Arduino dan untuk fungsi deskripsi akan diimplementasikan pada HTTP *client* yang dikonfigurasi pada NI Labview. Data yang dienkripsi berupa nilai suhu dan kelempaban yang dihasilkan oleh sensor DHT11 lebih jelasnya akan dijelaskan pada kategori algoritme AES.
4. Setelah algoritme AES diimplementasikan pada sistem maka dilakukan proses pengiriman data oleh protokol HTTP di mana server akan mengirimkan data sensor ke *client*. Sebelum dikirimkan data yang akan dikirim oleh server akan dienkripsi terlebih dahulu dan proses deskripsi akan dilakukan setelah data diterima oleh *client*.
5. Setelah data diterima oleh HTTP *client* dan berhasil dideskripsi oleh algoritme AES maka data tersebut akan ditampilkan pada *output* sistem yang ada pada NI Labview berupa nilai suhu dan kelembapan sensor DHT11.

5.1.1 HTTP *Server* dan HTTP *Client*

Pada perancangan sistem ini juga menggunakan protokol komunikasi HTTP sebagai protokol komunikasi yang digunakan antara server dan *client*. Berikut berupa *flowchart* untuk menjelaskan bagaimana protokol HTTP bekerja pada sistem ini berdasarkan Gambar 5.2.



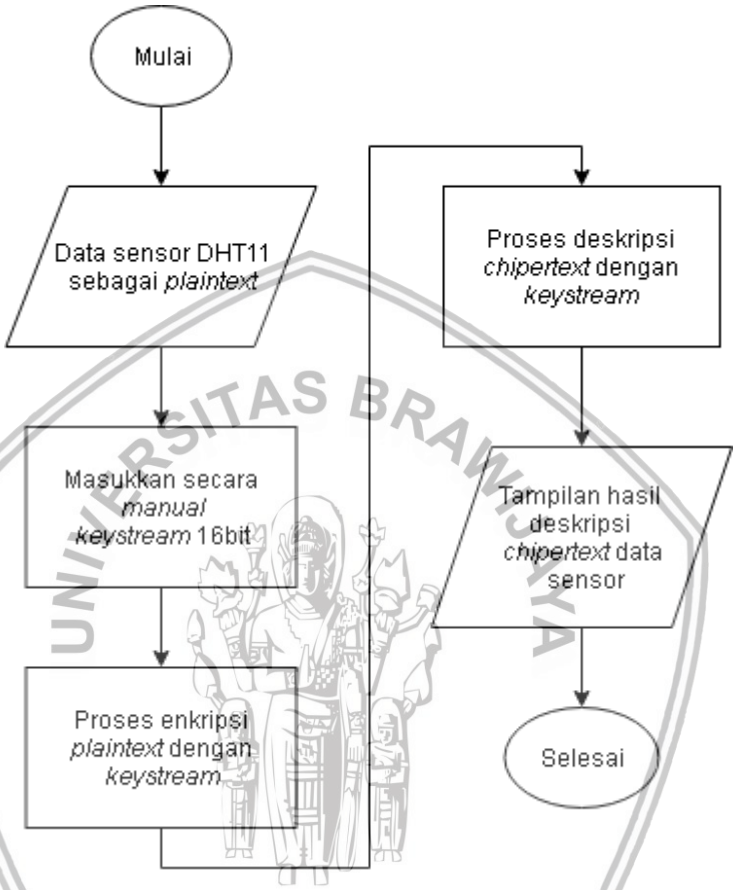
Gambar 5.2 Alur Proses Protokol HTTP

Berikut penjelasan pada Gambar 5.2 di atas adalah *flowchart* bagaimana proses protokol HTTP berjalan pada sistem ini. Berikut penjelasan untuk *flowchart* dari Gambar 5.2:

1. Saat sistem awal dijalankan sensor akan mengambil nilai suhu dan kelembapan setelah nilai diambil data dari nilai sensor akan dikirim ke mikrokontroller Arduino Uno untuk diolah.
2. Setelah data diterima dan diolah oleh mikrokontroler Arduino Uno selanjutnya masuk ke proses awal pada protokol HTTP di mana data dari sensor tadi akan dikirim ke HTTP server dan disimpan pada server.
3. Setelah data diterima oleh server selanjutnya HTTP *client* akan meminta data ke HTTP server. Di mana HTTP *client* akan mengirimkan pesan *request* ke HTTP server dan saat HTTP server menerima sinyal *request* dari HTTP *client* untuk permintaan data maka HTTP server akan membalas sinyal *request* tadi dengan sinyal *response* beserta data yang diminta oleh HTTP *client*. Untuk koneksi yang digunakan di sini menggunakan LAN (*Local Area Network*).

5.1.2 Enkripsi dan Deskripsi Data

Pada perancangan sistem ini juga memiliki fitur tambahan enkripsi dan deskripsi data yaitu berupa nilai dari sensor DHT11. Berikut berupa *flowchart* untuk menjelaskan bagaimana nilai yang akan di proses pada sistem ini berdasarkan Gambar 5.3.



Gambar 5.3 Enkripsi dan Deskripsi Algoritme AES

Berikut penjelasan pada Gambar 5.3 di atas adalah *flowchart* bagaimana cara enkripsi dan deskripsi algoritme AES pada sistem ini:

a. Di sini sistem menggunakan nilai suhu dan kelembapan dari sensor DHT11 sebagai *plaintext* yang akan digunakan oleh algoritme AES. *Plaintext* di sini adalah sebutan pada algoritme kriptografi untuk sebuah pesan atau data yang masih dapat dibaca.

b. Setelah sensor DHT11 dikonfigurasikan sebagai *plaintext* selanjutnya adalah memasukkan *key* atau kunci yang digunakan untuk mengubah *plaintext* ke *chipertext* yaitu pesan yang tidak dapat dibaca atau sudah dienkripsi. Karena algoritme AES merupakan algoritme kriptografi bermodel *stream chipper* maka kunci yang digunakan untuk enkripsi sama dengan kunci yang digunakan untuk deskripsi. Kunci yang digunakan di sini mempunyai panjang 16 *byte* dengan tipe data adalah *hexadecimal* yang ditentukan secara manual.

c. Setelah memasukkan *key* atau kunci selanjutnya sistem akan melakukan proses enkripsi *plaintext* yang berupa data dari nilai suhu dan kelembapan dari sensor DHT11 menjadi *chipertext* sebelum dikirimkan oleh HTTP server ke HTTP *client*. Data yang akan diterima oleh HTTP *client* adalah berupa *chipertext* dari hasil penggabungan *plaintext* dengan *key* sepanjang 16 *byte* dengan tipe data *hexadecimal*.

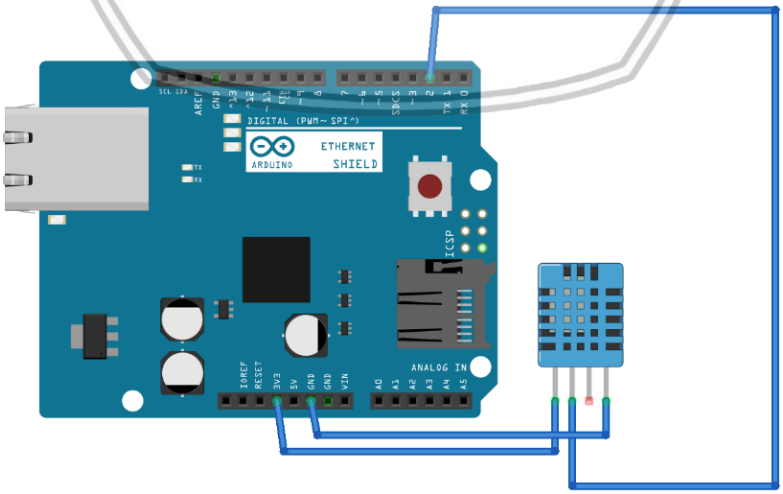
d. Setelah data diterima oleh HTTP *client* selanjutnya sebelum data ditampilkan pada *output* sistem maka data yang berupa *chipertext* tadi akan dideskripsikan dulu dengan menggunakan *key* atau kunci yang sama pada proses enkripsi. Setelah *chipertext* berhasil dideskripsikan data hasil deskripsi tersebut akan ditampilkan pada *output* sistem yaitu berupa nilai suhu dan kelembapan.

5.2 Implementasi

Pada bab sebelumnya sudah membahas tentang konsep perancangan, yang mana akan menjadi dasar dalam melakukan implementasi sistem. Di dalam tahap ini akan dijelaskan bagaimana implementasi sensor DHT11 sebagai data *input* untuk protokol HTTP dan *plaintext* algoritme AES, implementasi protokol HTTP di mana HTTP server berjalan pada Arduino dan HTTP *client* berjalan pada Labview dan Implementasi algoritme kriptografi AES yang algoritme enkripsinya berada pada HTTP server dan algoritme deskripsinya berada pada HTTP *client*.

5.2.1 Implementasi Perangkat Keras

Implementasi perangkat keras berupa penyatuan Ethernet Shield ke Arduino Uno dan penyambungan DHT11 ke pin dijital pada Arduino Uno. Pin dijital yang digunakan adalah pin D2 yang dihubungkan ke sensor DHT11 dan untuk koneksi jaringan menggunakan Ethernet LAN (*Local Area Network)* yang dihubungkan ke laptop. Implementasi perangkat keras dapat ditunjukkan pada Gambar 5.4.



Gambar 5.4 Skema Perancangan Perangkat keras

Setelah perangkat keras sudah selesai dirancang selanjutnya adalah inisialisasi alamat ip yang akan digunakan oleh HTTP server dan HTTP *client.*

Alamat ip di sini berguna untuk menghubungkan HTTP server dan HTTP *client* melalui jaringan LAN (*Local Area Network*). Untuk alamat ip yang digunakan oleh HTTP server adalah 192.0.0.101 dan untuk HTTP *client* adalah 192.0.0.102.

5.2.2 Implementasi Perangkat Lunak

Implementasi perangkat lunak pada tugas akhir merupakan implementasi sistem dari hasil yang dibuat sebelumnya. Dalam implementasi perangkat lunak ini direpresentasikan dalam bentuk kode program Arduino Uno dan NI Labview.

5.2.2.1 Implementasi Sensor DHT 11 sebagai Input

Pada sistem ini penulis menggunakan sensor suhu dan kelembapan DHT11 sebagai data *input* yang digunakan oleh protokol HTTP dan juga sebagai data *plaintext* untuk algoritme AES. Proses yang pertama adalah mendeklerasikan pin yang digunakan oleh sensor DHT11.

Tabel 5.1 Deklarasi pin sensor DHT11

No.	Kode Program
1	`#include "DHT.h"`
2	`#define DHTPIN 2`
3	`#define DHTTYPE DHT11`

Pada Tabel 5.1 penulis menggunakan pin 2 dijital pada Arduino Uno dan tipe sensor suhu dan kelembapan DHT11. Selanjutnya adalah implementasi program untuk pengambilan data pada sensor DHT11, data yang digunakan adalah nilai suhu (*temperature)* dan kelembapan (*humidity*). Program dapat dilihat pada Tabel 5.2 berikut:

Tabel 5.2 Kode program *sensing* data sensor DHT11

No.	Kode Program
1	`dht.begin();`
2	`delay(5000);`
3	`int h = dht.readHumidity();`
4	`// Read temperature as Celsius (the default)`
5	`int t = dht.readTemperature();`
6	`if (isnan(h) || isnan(t)) {`
7	`Serial.println("Failed to read from DHT sensor!");`
8	`return;`
	`}`

Berdasarkan Tabel 5.2 diberikan *delay* selama 5 detik karena sensor DHT11 memerlukan waktu untuk melakukan pembacaan dan pengambilan data.

5.2.2.2 Implementasi HTTP Server pada Arduino IDE

Pada penelitian ini penulis menggunakan protokol HTTP sebagai protokol komunikasi yang digunakan sistem. Pada protokol HTTP terdapat dua fungsi yaitu HTTP server dan HTTP *client*. HTTP server di sini berfungsi sebagai penyedia data dari sensor DHT11 yang nantinya akan dikirimkan ke HTTP *client.* Program untuk deklarasi fungsi HTTP server dapat dilihat pada Tabel 5.3 berikut ini:

Tabel 5.3 Kode program deklarasi fungsi HTTP *Server*

No.	Kode Program
1	#include <Ethernet.h>
2	*byte* mac[] = {
3	0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED
4	};
5	IPAddress ip(192, 0, 0, 101);
6	EthernetServer Server(80);

Di sini penulis menggunakan Ethernet atau *local network* untuk komunikasi jaringan yang digunakan oleh sistem. Dapat dilihat pada Tabel 5.3 *IPAddress* 192.0.0.101 adalah alamat ip yang digunakan untuk mengakses HTTP server dan "Server(80)" adalah *port default* yang digunakan oleh HTTP server. Selanjutnya adalah fungsi untuk memulai koneksi oleh HTTP server, program dapat dilihat pada tabel 5.4:

Tabel 5.4 Kode program memulai koneksi HTTP *Server*

No.	Kode Program
1	Ethernet.begin(mac, ip);
2	*Server*.begin();
3	Serial.print("*Server* is at ");
4	Serial.println(Ethernet.localIP());

5.2.2.3 Implementasi Enkripsi Algoritme AES

Pada sistem ini penulis menggunakan algoritme kriptografi AES sebagai algoritme keamanan yang diimplementasikan untuk enkripsi pengiriman data sensor DHTT11 menggunaka protokol HTTP. Pada algoritme kriptografi terdapat dua fungsi yaitu enkripsi (*encryption*) dan deskripsi (*decryption*) sama halnya pada algoritme AES. Pada sistem ini enkripsi akan dijalankan pada HTTP server di mana sebelum pengiriman data ke HTTP server, data akan dienkripsi oleh algoritme AES yang mana akan diubah menjadi *chipertext* dengan menggunakan *key* sepanjang 16 *byte*. Selanjutnya adalah membuat *key* sepanjang 16 *byte*, di sini penulis memasukkan kunci secara manual ke dalam program yaitu 0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f dan diubah kedalam bentuk *hexadecimal*. Setelah memasukkan *key* proses selanjutnya adalah proses enkripsi data yang akan dikirimkan menuju HTTP server. Untuk program proses enkripsi dapat dilihat pada Tabel 5.5 berikut ini:

Tabel 5.5 Proses enkripsi data

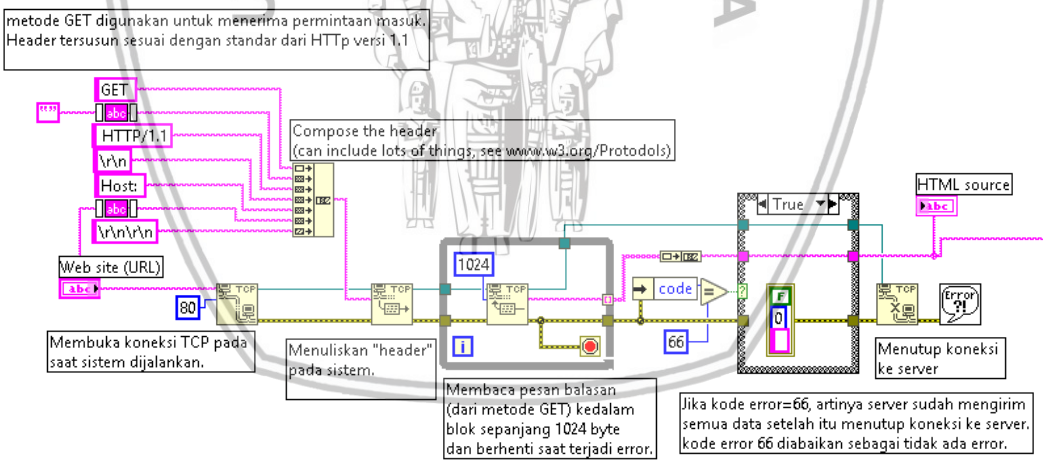
No.	Kode Program
1	char data[16];
2	Serial.print(h);
3	Serial.print(",");
4	Serial.println(t);
5	sprintf(data, "%02d,%02d", h, t);
6	AES128_enc_single(*key*, data);
7	for (int i=0; i<16; i++){
8	*Client*.print(String((*byte*)data[i], HEX));

9	}
10	break;
11	}

Dapat dilihat pada Tabel 5.5, di sini penulis menggunakan algoritme AES 128-bit dengan panjang *key* 16*byte*, di mana *plaintext* yang digunakan adalah data sensor DHT11 yaitu kelembapan (*humidity*) dan suhu (*temperature*). Pada proses enkripsi algoritme AES terdiri dari empat jenis transformasi *byte*, yaitu *SubBytes, ShiftRows, MixColumns* dan *Add*Ronde*Keys*. Proses ini akan dilakukan berulang-ulang selama sepuluh ronde di mana pada ronde terakhir proses *MixColumns* tidak akan dilakukan. Semua proses tersebut dipanggil menggunakan *library AES128_enc_single(key,data)*, di mana *library* tersebut berguna untuk memanggil fungsi enkripsi algoritme AES 128 bit. Setelah dilakukan proses enkripsi, *plaintext* tadi akan diubah menjadi *chipertext* di mana *chipertext* akan dikirim ke HTTP *client*, data yang dikirim dengan format *hexadecimal*.

5.2.2.4 Implementasi HTTP Client Pada NI Labview

Pada sistem ini penulis mengimplementasikan HTTP *client* pada NI Labview. Sebelumnya HTTP server diimplementasikan pada Arduino dan untuk HTTP *client* diimplementasikan pada NI Labview. Program HTTP *client* dapat dilihat pada Gambar 5.5 berikut ini:



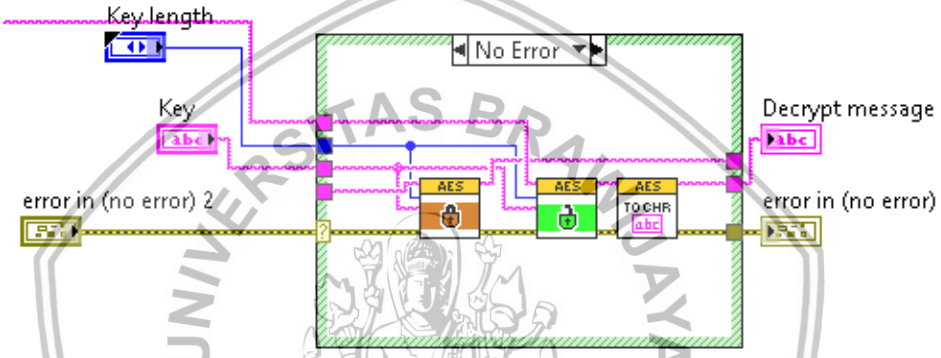
Gambar 5.5 Implementasi HTTP *Client* pada Labview

Pada Gambar 5.5 dapat dilihat penulis menggunakan metode *HTTP Request GET* di mana pada metode ini HTTP *client* melakukan *request* untuk mengambil data dari HTTP server tanpa memperbarui data tersebut. Untuk versi HTTP yang digunakan adalah versi 1.1. Untuk *web site (URL)* diisi dengan alamat ip dari HTTP server, digunakan untuk menghubungkan HTTP *client* dengan HTTP server. Protokol HTTP berkomunikasi melalui TCP/ IP, di mana HTTP *client* terhubung ke HTTP server menggunakan TCP pada *port* 80. Selanjutnya proses kerja protokol TCP/IP, proses awal data yang dikirim oleh HTTP server akan

dibungkus dengan informasi *header* yang dibutuhkan, kemudian data dikirimkan ke lapisan IP yang mana akan ditambahkan *header* IP setelah itu IP mengarahkan data tersebut ke tujuan. HTTP *client* sebagai penerima membaca *response* dari HTTP *server* berupa informasi *header* lalu membuka informasi *header* tersebut dan melakukan *checksum* untuk memeriksa adanya *error* atau tidak. Jika tidak ada *error* dan data sudah diterima, maka tutup koneksi TCP/IP.

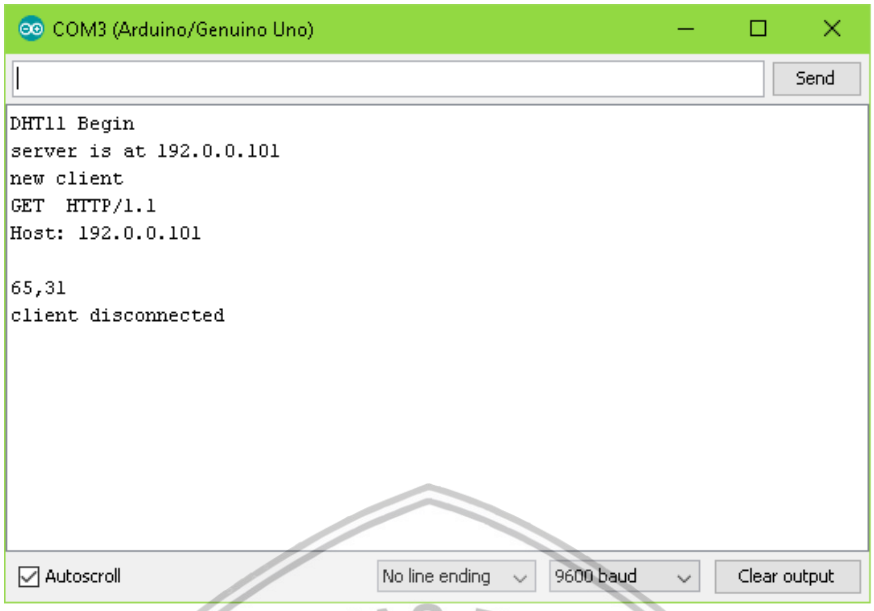
5.2.2.5 Implementasi Deskripsi Algoritme AES pada NI Labview

Data yang diterima oleh HTTP *Client* masih berupa *chipertext* dari hasil enkripsi algoritme AES, maka dibutuhkan proses deskripsi untuk mengubah *chipertext* tadi menjadi *plaintext* yang dapat dibaca. Program untuk deskripsi *chipertext* tersebut dapat dilihat pada Gambar 5.6 berikut ini:



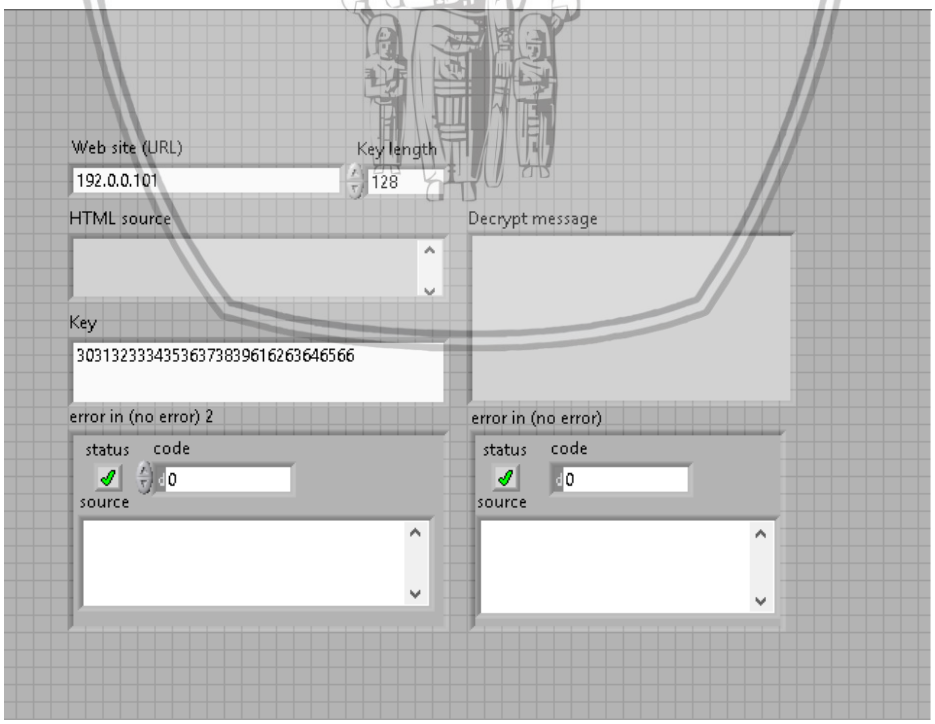
Gambar 5.6 Implementasi Deskripsi pada NI Labview

Pada algoritme kriptografi AES *key* yang digunakan untuk enkripsi dan deskripsi sama yaitu "0123456789abcdef" dengan panjang kunci 128 bit (*key length)*. Untuk masukkan *key* adalah hasil data yang sudah diterima oleh HTTP *client* yaitu berupa *plaintext* yang akan diproses untuk diubah menjadi *chipertext* (*Decrypt message*) pada program ini. Untuk tampilan *output* sistem menggunakan *front panel* pada aplikasi NI Labview. Menu yang ada pada *output* sistem berupa *"keylength"* untuk menentukan panjang *key* yang akan digunakan, *"key"* untuk memasukkan *key* yang akan digunakan untuk deskripsi, *"web site(URL)"* untuk memasukkan alamat ip dari HTTP server dan *"Decrypt message"* hasil dari deskripsi *chipertext* menjadi *plaintext* yang berupa data sensor kelembapan dan suhu DHT11.



Gambar 5.7 Tampilan *Serial Monitor* pada Arduino

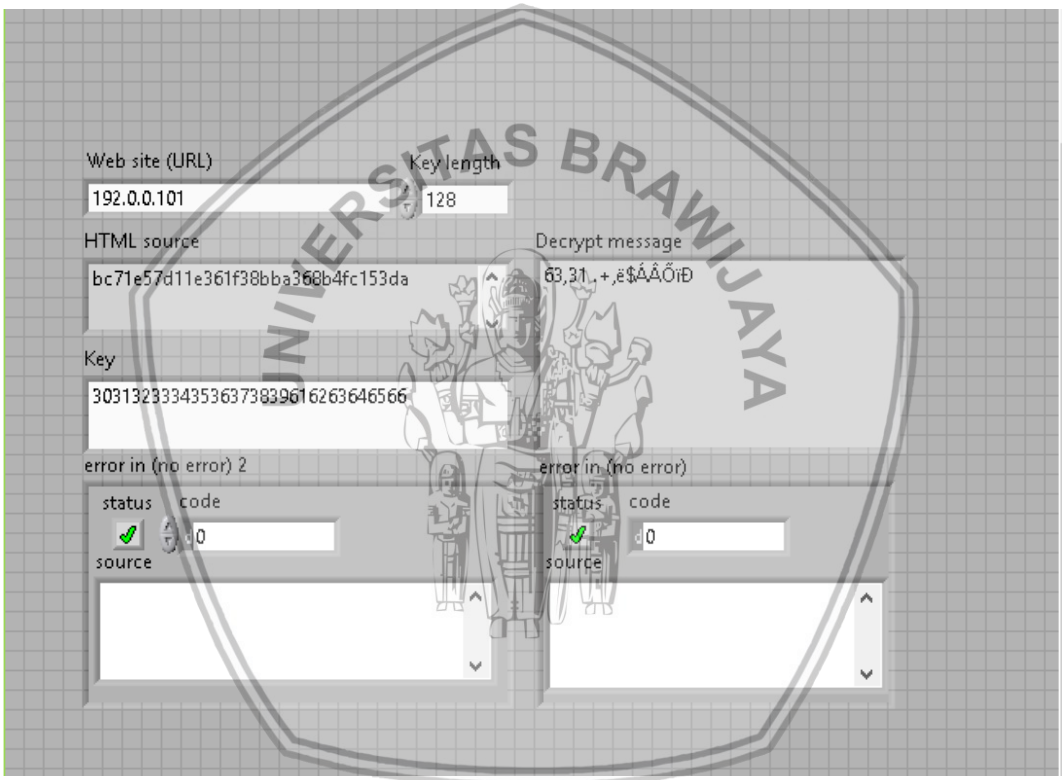
Gambar 5.7 menunjukkan tampilan *serial monitor* pada Arduino IDE saat sistem dijalankan, dapat dilihat "*DHT11 Begin*" menunjukkan bahwa sensor DHT11 mulai melakukan *sensing* dan mengambil nilai suhu dan kelembapan. HTTP server sendiri mulai berjalan setelah sensor selesai mengambil data dengan alamat ip 192.0.0.101 untuk nilai sensor berupa suhu dan kelembapan yaitu 65,31 akan ditampilkan ketika HTTP *client* me-*request* data ke pada HTTP server.



Gambar 5.8 Tampilan *Output* Sistem pada *FrontPanel* NI Labview

Selanjutnya pada Gambar 5.8 adalah tampilan *output* sistem yang berada pada *front panel* NI Labview di mana terdiri dari "*Web site(URL)*" yang berfungsi

untuk memasukkan alamat ip yang di tuju oleh HTTP *client* di sini alamat ip yang akan di tuju adalah alamat ip HTTP server yaitu 192.0.0.101 lalu ada "*HTML source*" yang berfungsi untuk menampilkan data yang diterima oleh HTTP *client* dari HTTP server berupa *chipertext*. Selanjutnya ada "*Key*" di sini berfungsi untuk memasukkan kunci yang akan digunakan untuk mendeskripsikan *chipertext* yaitu berupa pesan unik yang tidak bisa dibaca menjadi *plaintext* yaitu pesan yang dapat dibaca di sini penulis menggunakan *key* yang sama dengan yang digunakan untuk mengenkripsi *plaintext* pada HTTP server lalu ada "*Decrypt message*" yang berfungsi untuk menampilkan hasil deskripsi *chipertext* yang diterima oleh HTTP *client* dari HTTP server yaitu berupa nilai suhu dan kelembapan. Untuk melihat hasil keselurahan dari berjalannya sistem pada HTTP *client* bisa dilihat pada Gambar 5.9.



Gambar 5.9 Tampilan *Output* Sistem Saat Berjalan

BAB 6 PENGUJIAN DAN ANALISIS

Bab ini akan membahas mengenai perancangan dan implementasi dari sistem yang akan dibuat untuk penelitian yang akan dilakukan yaitu "Implementasi Algoritme AES pada Pengiriman Data Sensor DHT11 Menggunakan Protokol Komunikasi HTTP". Pada pengujian dan analisis terdapat pengujian pengiriman dan enkripsi-deskripsi data sensor DHT11 pada protokol HTTP, pengujian waktu enkripsi dan deskripsi dan pengujian *delay* pada protokol HTTP.

6.1 Pengujian Enkripsi dan Deskripsi pada Pengiriman Data Sensor

Dalam sistem ini digunakan algoritme enkripsi AES yang diimplementasikan untuk pengaman pengiriman data sensor DHT11 menggunakan protokol HTTP. Pengujian dilakukan dengan menggunakan Arduino Uno yang sudah ditanamkan Ethernet Shield sebagai mikrokontroler dan jalur komunikasi jaringan berupa LAN. HTTP server dan enkripsi akan diimplementasikan pada Arduino IDE untuk HTTP *client* dan deskripsi akan diimplementasikan pada NI Labview.

6.1.1 Tujuan Pengujian

Pengujian dilakukan dengan tujuan untuk mengetahui kinerja algoritme AES pada enkripsi data sensor DHT11 dengan menggunakan protokol HTTP pada *platform* yang berbeda. Untuk algoritme enkripsi diimplementasikan pada HTTP server di Arduino dan untuk algoritme deskripsi diimplementasikan pada HTTP *client* pada NI Labview. Hasil yang diharapkan dari pengujian ini adalah data sensor yang dienkripsi pada HTTP *server* dapat dikirimkan ke HTTP *client* dan berhasil dideskripsi untuk ditampilkan pada *output* sistem. Hasil dari setiap percobaan akan dicatat dan dianalisis dalam bentuk total *error rate* untuk mengetahui kinerja sistem.

6.1.2 Prosedur Pengujian

Berikut langkah-langkah pengujian enkripsi dan deskripsi pada protokol HTTP.

1. Menghubungkan Arduino Uno+Ethernet Shield ke laptop.
2. Membuka program HTTP server+enkripsi pada Arduino IDE.
3. Membuka program HTTP *client*+deskripsi pada NI Labview.
4. Memasukkan *key* pada program.
5. Menjalankan program HTTP server+enkripsi pada Arduino IDE dilanjutkan program HTTP *client*+deskripsi pada NI Labview.
6. Mencatat keberhasilan sistem melakukan enkripsi dan deskripsi data serta pengiriman antara HTTP server dan HTTP *client*.

7. Percobaan dilakukan sebanyak 25 kali dengan jeda saat *output* sistem menampilkan hasil deskripsi sistem dan dimuat ulang.

6.1.3 Hasil dan Analisis

Tabel 6.1 Pengujian Enkripsi dan Deskripsi pada Protokol HTTP

percobaan ke-	Data		Enkripsi	Deskripsi	Terkirim
	Humidity	Temperature			
1	0	0	✓	✓	✓
2	63	27	✓	×	✓
3	95	28	✓	✓	✓
4	63	28	✓	✓	✓
5	63	27	✓	✓	✓
6	64	27	✓	×	✓
7	65	27	✓	✓	✓
8	65	27	✓	✓	✓
9	95	28	✓	×	✓
10	95	29	✓	×	✓
11	67	27	✓	×	✓
12	66	27	✓	×	✓
13	65	27	✓	✓	✓
14	66	27	✓	✓	✓
15	66	27	✓	✓	✓
16	66	27	✓	✓	✓
17	68	28	✓	✓	✓
18	62	29	✓	✓	✓
19	62	29	✓	✓	✓
20	62	29	✓	×	✓
21	61	29	✓	×	✓
22	70	30	✓	✓	✓
23	42	42	✓	✓	✓
24	46	32	✓	×	✓
25	53	30	✓	✓	✓
Total			25	19	25
Total Error Rate			0%	24%	0%

Dengan menggunakan persamaan total *error rate*, maka dapat menghasilkan nilai error dari setiap data yang diperoleh. Berikut adalah perhitungan total *error rate* dari melakukan 25 kali percobaan, data dapat dilihat pada Tabel 6.1.

$$\textit{Total Error Rate} = \frac{Jumlah\ Error - Total\ Pengujian}{Total\ Pengujian}\ x\ 100$$

$$\textit{Total Error Rate} = \frac{19 - 25}{25}\ x\ 100 \qquad (6.1)$$

Total Error Rate = 24%

Total *error rate* didapatkan melalui jumlah keberhasilan sistem dalam mengenkripsi, mengdeskripsi dan pengiriman data. Untuk menghitung total *error rate* yakni dengan mengurangkan total error dengan total pengujian dan dikali 100. Didapatkan persentase error untuk pengiriman data 0%, enkripsi data 0% dan deskripsi data 24%. Penyebab error pada deskripsi karena dalam proses enkripsi di Arduino saat proses pembuatan *chipertext* panjang datanya terkadang tidak mencapai 16 *byte* atau 128 bit sedangkan untuk proses deskripsi di NI Labview dibutuhkan data sepanjang 16 *byte* atau 128 bit hal ini terjadi karena data sensor tidak mencapai 128 bit dan perlu dilakukan *padding* untuk menambahkah jumlah bit pada data tersebut hingga mencapai 128 bit lalu dikonversikan ke bentuk *hexadecimal* lalu dikirimkan ke HTTP *client* untuk dideskripsi menjadi *plaintext*.

6.2 Pengujian Performa Waktu Enkripsi dan Deskripsi

Dalam sistem ini menggunakan Algoritme Enkripsi AES 128 bit, di mana pada versi 128 bit untuk proses pembuatan *chipertext* dari *plaintext* membutuhkan waktu yang lebih cepat dari pada versi 192 bit dan 256 bit. Pengujian dilakukan dengan menambahkan program untuk menghitung setiap proses enkripsi dari *plaintext* ke *chipertext* dan deskripsi dari *chipertext* ke *plaintext*.

6.2.1 Tujuan Pengujian

Pengujian dilakukan dengan tujuan untuk mengetahui waktu yang dibutuhkan sistem untuk melakukan proses enkripsi untuk merubah *plaintext* ke *chipertext* dan proses deskripsi untuk merubah *chipertext* ke *plaintext.* Hasil dari setiap percobaan akan dicatat dan dihitung untuk mendapatkan rata-rata waktu pemrosesan.

6.2.2 Prosedur Pengujian

Berikut langkah-langkah pengujian waktu pemrosesan enkripsi dan deskripsi.

1. Menghubungkan Arduino Uno+Ethernet Shield ke laptop.
2. Membuka program HTTP server+enkripsi pada Arduino IDE.
3. Membuka program HTTP *client*+deskripsi pada NI Labview.
4. Menambahkan program untuk menghitung waktu enkripsi dan deskripsi sistem.
5. Memasukkan *key* pada program.
6. Menjalankan program HTTP server+enkripsi pada Arduino IDE dilanjutkan program HTTP *client*+deskripsi pada NI Labview.
7. Mencatat waktu proses enkripsi dan deskripsi
8. Pengujian dilakukan sebantak 15 kali dengan jeda saat *output* sistem menampilkan hasil deskripsi sistem dan dimuat ulang.

Berikut adalah persamaan yang digunakan untuk menghitung waktu pemrosesan dari waktu enkripsi dan waktu deskripsi.

Waktu Pemrosesan = Waktu Enkripsi + Waktu Deskripsi (6.2)

6.2.3 Hasil dan Analisis

Tabel 6.2 Pengujian Waktu Proses Enkripsi dan Deskripsi

Percobaan ke-	Plaintext	Chipertext	Waktu Pemrosesan(s)
1	68,26	4b8f37323ab428d635d492ab8a47a3d	14.6
2	68,26	2656e88fa3be88b637fa9978b6c21bd	13.7
3	69,26	a89fb2da3aae7951b676e9e0329c594d	12.9
4	70,26	dccaefa5e0eae1a378a4fee720b6dada	13.7
5	70,26	cbdeaa604e09ddc9a126987581328c	13
6	70,26	96fabda2639548ff2e2284c70b76c57	14.1
7	72,27	2b9e4042c0ed1b88c74be8f1edccd	15.5
8	70,26	58b05591e094c0afb01ce398b17ea82b	14
9	71,26	784ea13e4cced2f0757de350efbc224a	16.1
10	66,28	f5be9489c5b401117624b8f13fa2ab	15.2
11	63,29	84ec6d58bb7147376a6b9fdcd4e78f44	12.2
12	62,29	6ca559a6241724f3bcd347bb3c1bbb65	16.7
13	63,27	d3e226c62eef82d58a894fbe5da02fa6	11.9
14	64,27	2caa2f56e01baaa297181566fd8fa9b6	12.7
15	64,27	177697d9f2b2e6d62274803119c4e449	12.3
Rata-Rata			13.91

Berdasarkan Tabel 6.2, sistem membutuhkan rata-rata waktu pemrosesan sebesar 13.91 detik untuk melakukan enkripsi dan deskripsi data. Dapat dilihat pada setiap percobaan total waktu pemrosesan sekitar 12.9 – 16.1 detik dengan selisih waktu 3 detik antara waktu tercepat dengan waktu terlama. Hal ini membuktikan bahwa sistem mempunyai waktu pemrosesan yang cukup stabil dalam pemrosesan enkripsi dan deskripsi. Hal ini terjadi karena sistem memproses data yang tidak berjumlah besar berupa satu sensor DHT11 dan juga versi yang digunakan adalah AES 128 bit yang jumlah ronde-nya lebih sedikit dari pada versi lain.

6.3 Pengujian *Delay* pada Protokol HTTP

Pada protokol HTTP ada dua fungsi yang digunakan yaitu HTTP server dan HTTP *client*. Dalam protokol HTTP pasti ada selisih waktu pengiriman dari HTTP server ke HTTP *client* atau biasa disebut dengan *delay*. *Delay* digunakan untuk melihat berapa lama selisih pengiriman data antara HTTP server dan HTTP *client*. Pengujian *delay* tersebut dilakukan sebanyak 15 kali dan akan dihitung rata-rata *delay*-nya.

6.3.1 Tujuan Pengujian

Pengujian dilakukan dengan tujuan untuk mengetahui berapa *delay* pengiriman data dari HTTP server menuju HTTP *client*. Sehingga dapat dilihat berapa *delay* protokol HTTP pada sistem ini.

6.3.2 Prosedur Pengujian

Berikut langkah-langkah pengujian *delay* antara HTTP server menuju HTTP *client*.

1. Menyiapkan *Wireshark* sebagai alat bantu pengujian.
2. Menghubungkan Arduino Uno+Ethernet Shield ke laptop.
3. Membuka program HTTP server+enkripsi pada Arduino IDE.
4. Membuka program HTTP *client*+deskripsi pada NI Labview.
5. Memasukkan *key* pada program.
6. Menjalankan program HTTP server+enkripsi pada Arduino IDE dilanjutkan program HTTP *client*+deskripsi pada NI Labview.
7. Mengukur waktu pengiriman paket dari HTTP server ke HTTP *client* menggunakan bantuan *Wireshark*.
8. Percobaan dilakukan sebanyak 15 kali dengan jeda saat *output* sistem menampilkan hasil deskripsi sistem dan dimuat ulang.

Berikut adalah persamaan yang digunakan untuk menghitung *delay* pengiriman pada protokol HTTP.

$$Delay = \frac{Waktu\ Paket\ Diterima - Waktu\ Paket\ Dikirim}{Jumlah\ Paket} \quad (6.3)$$

6.3.3 Hasil dan Analisis

Tabel 6.3 Pengujian *Delay* Protokol HTTP

Percobaan ke-	HTTP *Server*(s)	HTTP *Client*(s)	Jumlah Paket Terkirim	*Delay*(s)
1	5.30235	5.393823	33	0.002772
2	5.299169	5.390807	33	0.002777
3	5.299334	5.390139	33	0.002752
4	5.296356	5.386961	33	0.002746
5	5.296425	5.388156	33	0.00278
6	5.301498	5.392412	33	0.002755
7	5.296321	5.388098	33	0.002781
8	5.302255	5.393845	33	0.002775
9	5.296006	5.386188	33	0.002733
10	5.296115	5.386274	33	0.002732
11	5.299016	5.391558	33	0.002804
12	5.305611	5.396516	33	0.002755

13	5.305927	5.39651	33	0.002745
14	5.306458	5.397428	33	0.002757
15	5.302423	5.392643	33	0.002734
Rata-rata				0.00276

Dapat dilihat pada Tabel 6.3 rata-rata *delay* dari pengiriman protokol HTTP adalah 0.00276 detik dengan jumlah paket yang dikirim 33 pada sekali percobaan. Pada Tabel 6.3 terlihat bahwa perbedaan *delay* dari percobaan pertama sampai percobaan ke-lima belas tidak terlalu signifikan dikarenakan jumlah paket yang tidak terlalu besar. Dari hasil pengujian di atas dapat disimpulkan proses enkripsi dan deskripsi tidak terlalu mempengaruhi *delay* pengiriman pada protokol HTTP dikarenakan proses enkripsi dilakukan sebelum data sensor dikirim dari HTTP server menuju HTTP *client*.

BAB 7 PENUTUP

Bab ini memuat penarikan kesimpulan dan saran yang diharapkan dapat berguna untuk pengembangan selanjutnya berdasarkan tahapan yang telah dilakukan sebelumnya dari penelitian ini.

7.1 Kesimpulan

Kesimpulan yang diperoleh berdasarkan implementasi dan pengujian adalah sebagai berikut:

1. Sistem ini dirancang menggunakan algoritme AES untuk mengamankan pengiriman dari data sensor DHT11 yang dimana protokol HTTP digunakan sebagai protokol komunikasi untuk menghubungkan server dengan *client*. Sistem menggunakan Arduino Uno sebagai mikrokontroler untuk mengolah data dari sensor DHT11 yang ditambahkan dengan Ethernet Shield sebagai perangkat yang menyediakan jalur komunikasi Ethernet atau *wired*. Proses enkripsi pada algoritme AES dan HTTP server diimplementasikan pada pemrograman perangkat lunak Arduino IDE dan untuk proses dekripsi dan HTTP *client* diimplementasikan pada pemrograman perangkat lunak NI Labview. *Output* sistem diimplementasikan pada *front panel* pada NI Labview.
2. Algoritme AES berhasil diimplementasikan pada pengiriman data sensor DHT11 menggunakan protokol HTTP pada dua *platform* yang berbeda yaitu Arduino dan NI Labview dimana data suhu dan kelembapan DHT11 dienkripsi menjadi *chipertext* lalu dikirim ke HTTP *client* melalui HTTP server yang di program pada Arduino IDE setelah itu HTTP *client* yang di program pada NI Labview melakukan *request* ke HTTP server dan server membalas pesan *request* dengan pesan *response* beserta data yaitu berupa data sensor DHT11, selanjutnya data dideskripsi menjadi *plaintext* dan ditampilkan pada *output* sistem yaitu *front panel* pada NI Labview.
3. Dari hasil seluruh pengujian yang dilakukan pada penelitian ini didapatkan hasil performa di mana pada proses enkripsi dan deskripsi data sensor DHT11 menggunakan protokol HTTP didapatkan persentase *error rate* sebesar 24% pada proses deskripsi sedangkan pada proses pengiriman dan enkripsi 0%, dan untuk waktu pemrosesan enkripsi dan deskripsi didapatkan rata-rata waktu pemrosesan sebesar 13.91 detik dengan menggunakan algoritme AES versi 128 bit, lalu untuk rata-rata *delay* pengiriman pada protokol HTTP didapatkan nilai sebesar 0.00276 detik dengan jumlah paket terkirim sebanyak 33 pada setiap percobaannya. Dari hasil tersebut menandakan implementasi sistem algoritme AES untuk enkripsi pengiriman data sensor DHT11 menggunakan protokol HTTP berjalan dengan stabil dan baik.

7.2 Saran

Berdasarkan kesimpulan yang didapatkan terdapat beberapa saran untuk penelitian selanjutnya yang akan mengembangkan sistem ini, berikut beberapa saran tersebut:

1. Untuk pengembangan penelitian selanjutnya dapat menggunakan Algoritme enkripsi atau protokol jaringan yang berbeda seperti algoritme A5, DES atau grain chipper dan protokol MQTT, CoAP atau XMPP.
2. Dapat mengubah jalur komunikasi jaringan yang lain menjadi nirkabel seperti Wi-Fi atau *Bluetooth* dengan menambahkan jumlah data atau sensor yang digunakan atau dengan menggunakan tipe data lain seperti *file*, gambar atau video+.
3. Pada proses penentuan *key* diharapkan pada pengembangan selanjutnya proses untuk pembuatan *key* bisa secara otomomatis dengan menyesuaikan masukan yang digunakan pengguna.

DAFTAR PUSTAKA

Akhmalia, R., 2016. *INTEGRASI TIMING-SYNC PROTOCOL FOR SENSOR NETWORK DENGAN TIME DIVISION MULTIPLE ACCSESS PADA PROTOKOL MQTT-SN,* Malang: Fakultas Ilmu Komputer Universitas Brawijaya.

Aminatus, J., 2018. *Implementasi dan Analisa Sistem Keamanan Di Jaringan Sensor Nirkabel Pada Standar Zigbee,* Bandung: Universitas Telkom .

Anadiansyah, 2017. *Robotika Modul untuk myRio dan roboRio,* Malang: universitas Muhammadiyah Malang.

Developer, N., 2007. *Ch.36 AES Encryption and Decryption on the GPU.* [Online] Available at: https://developer.nvidia.com/gpugems/GPUGems3/gpugems3_ch36.html [Accessed 9 8 2018].

Erritali, M., 2011. *A Contribution to Secure the Routing Protocol "Greedy Perimeter Stateless Routing" Using A Symmetric Signature Based AES and MD5 Hash,* s.l.: s.n.

Fachrurozi, M. F., 2006. *ENKRIPSI PESAN RAHASIA ALGORITMA (Advance Encryption Standard) AES:RIJNDAEL,* Jakarta: Universitas Islam Negeri Syarif Hidayatulah.

Hartari, B. S., 2014. *IMPLEMENTASI SISTEM KONTROL PID PADA PENGENDALIAN INTENSITAS CAHAYA LAMPU TERHADAP PENERANGAN RUANGAN BERBASIS LABVIEW DAN ARDUINO,* Yogyakarta: Universitas Gadjah Mada.

Indriawan, 2011. *PERANCANGAN DAN IMPLEMENTASI MAIL SERVER PADA CV. SANJAYA ANUGERAH SEJAHTERA (ISP JOGJAKARTA) BERBASIS OPEN SOURCE,* Yogyakarta: Unikom.

Irwan Dinata, W. S., 2015. *IMPLEMENTASI WIRELESS MONITORING ENERGI LISTRIK BERBASIS WEB DATABASE,* Bangka Belitung: Universitas Bangka Belitung.

Manaloe, P., 2015. *Perangkat Pada Mikrokontroler,* s.l.: s.n.

Musliyana, 2018. *Improvement of Data Exchange Security on HTTP using Client-side Encryption,* Indonesia: IOP Publishing.

Musliyana, Z., 2016. *Peningkatan Sistem Keamanan Autentikasi Single Sign On (SSO) Menggunakan Algoritma AES dan One-Time Password Studi Kasus: SSO Universitas Ubudiyah Indonesia,* Banda Aceh: Research Gates.

Muzammil, P., 2010. *Network Security lect4.* [Online] Available at: https://www.slideshare.net/pakmuzammil/network-security-lec4 [Accessed 9 8 2018].

Pitchaiah, M., 2012. *Implementation of Advanced Encryption Standard Algorithm.* s.l.:International Journal of Scientific & Engineering Research.

Samura, A., 2017. *SISTEM KONTROL DAN MONITORING KUALITAS AIR TAMBAK UDANG WINDU DENGAN METODE FUZZY LOGIC CONTROL MENGGUNAKAN MICROCONTROLLER NI MYRIO ,* Malang: s.n.

Saptadi, A. H., 2015. *Perbandingan Akurasi Pengukuran Suhu dan kelembapan Antara Sensor DHT11 dan DHT22,* Purwokerto: Sekolah Tinggi Teknologi Telematika Telkom Purwokerto.

Stricot-Tarboton, S., n.d. *Taxonomy of Man-In-The-Middle Attacks on HTTPS,* Hamilton, New Zealand: Cyber of Security Lab, University of Waikato .

SUKARMAN, 2008. *AKUSISI DATA LEWAT PROTOCOL TCP/IP BERBASIS LABVIEW,* Yogyakarta: Sekolah Tinggi Teknologi Nuklir-BATAN.

Surian, D., 2006. *Algoritma Kriptografi AES Rijndael,* Jakarta: Tesla.

Susana, R., 2015. *Perancangan dan Realisasi Web-Based Data Logging System menggunakan ATmega16 melalui Hypertext Transfer Protocol(HTTP),* Bandung: Portal Garuda.

Sutantyo, D. K., 2006. *Implementasi Embedded Web Server Via Modem Berbasiskan Mikrokontroler,* Salatiga: Universitas Kristen Satya Wacana.

Wanita, F., n.d. *Rancang Bangun Sistem Enkripsi dan Deskripsi Pengiriman Informasi Menggunakan Algoritmas RSA Berbasis Wifi,* Makassar: STMIK Handayani Makassar.

Wibowo, S. A., 2010. *Pengendalian Level Ketinggan Air Dengan menggunakan Kendali Logika Fuzzy,* Bandung: Perpustakaan Pusat Unikom.