SURVEILLANCE SYSTEM PADA RUANG *SERVER* BERBASIS ARDUINO ETHERNET SHIELD MENGGUNAKAN PROTOKOL MQTT

SKRIPSI

Oleh

Aditya Juli Prasetyo
NIM 131910201072

PROGRAM STUDI STRATA-1 TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2018

SURVEILLANCE SYSTEM PADA RUANG *SERVER* BERBASIS ARDUINO ETHERNET SHIELD MENGGUNAKAN PROTOKOL MQTT

SKRIPSI

diajukan guna melengkapi skripsi dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Teknik Elektro (S1)
dan mencapai gelar Sarjana Teknik

Oleh

Aditya Juli Prasetyo
NIM 131910201072

PROGRAM STUDI STRATA-1 TEKNIK ELEKTRO
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS JEMBER
2018

PERSETUJUAN PEMBIMBING

Skripsi berjudul ”*Surveillance System* Pada Ruang *Server* Berbasis Arduino Ethernet Shield Menggunakan Protokol MQTT” telah disetujui pada:

hari, tanggal : Jum’at, 5 Januari 2018

tempat : Fakultas Teknik Universitas Jember.

Dosen Pembimbing Utama,	Dosen Pembimbing Anggota
Ike Fibriani, S.T., M.T.	Dodi Setiabudi, S.T., M.T.
NIP. 19800207 201504 2 001	NIP. 19840531 200812 1 004

PERSEMBAHAN

Dengan rasa syukur saya panjatkan kepada Allah SWT atas segala karunia-Nya hingga saya dapat menyelesaikan skripsi ini. Skripsi ini merupakan langkah awal kesuksesan yang saya raih untuk mendekati masa depan dan meraih cita-cita di dalam hidup saya. Dengan penuh rasa syukur dengan ketulusan hati saya persembahkan karya ini kepada :

1. Allah SWT Yang Maha atas segalanya;
2. Nabi besar Muhammad SAW yang menjadi suri tauladan bagi seluruh umat manusia;
3. Kedua orang tua, Ibu Tri Kusmawati dan Bapak Mardiono yang tercinta dan tersayang dan adik saya ica yang sangat saya banggakan;
4. Dosen Pembimbing Utama Ibu Ike Fibriani, S.T., M.T. dan Bapak Dodi Setiabudi, S.T., M.T. atas kesabaran dan keikhlasan dalam membimbing saya menyelesaikan skripsi ini;
5. Keluarga perantuan H2-12A dan Purbalingga Perwira (IMAGARA);
6. Keluarga besar Laboratorium Teknologi Informatika Fakultas Teknik Universitas Jember;
7. Keluarga INTEL UJ 13 “Ikatan Teknik Elektro Universitas Jember 2013”;
8. Almamater Fakultas Teknik Universitas Jember.

MOTTO

Sembahlah Allah dan janganlah kamu mempersekutukan-Nya dengan sesuatupun. Dan berbuat baiklah kepada dua orang ibu-bapa, karib-kerabat, anak-anak yatim, orang-orang miskin, tetangga yang dekat dan tetangga yang jauh, dan teman sejawat, ibnu sabil dan hamba sahayamu. Sesungguhnya Allah tidak menyukai orang-orang yang sombong dan membangga-banggakan diri.[*)]

وَالصَّلاَةِبِالصَّبْرِوَاسْتَعِينُواْ

Mintalah pertolongan dengan sabar dan shalat.[**)]

"Akal dan belajar adalah itu seperti raga dan jiwa. Tanpa raga, jiwa hanyalah udara hampa. Tanpa jiwa, raga adalah kerangka tanpa makna." [***)]

---

*)", (Q.S An Nisaa', 4:36)..
**)QS Al-Baqarah: 45.
***)(Kahlil Gibran)

PERNYATAN

Saya yang bertanda tangan di bawah ini:

Nama : Aditya Juli Prasetyo

NIM : 131910201072

Menyatakan dengan sesungguhnya bahwa proyek akhir yang berjudul "*Surveillance System* Pada Ruang *Server* Berbasis Arduino Ethernet Shield Menggunakan Protokol MQTT" adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan dalam institusi mana pun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak mana pun serta bersedia mendapat sanksi akademik jika ternyata dikemudian hari pernyataan ini tidak benar.

Jember, 24 Januari 2018

Yang menyatakan,

Aditya Juli Prasetyo
NIM 131910201072

SKRIPSI

SURVEILLANCE SYSTEM PADA RUANG *SERVER* BERBASIS ARDUINO ETHERNET SHIELD MENGGUNAKAN PROTOKOL MQTT

Aditya Juli Prasetyo
NIM 131910201072

Pembimbing :

Dosen Pembimbing Utama : Ike Fibriani, ST., MT.

Dosen Pembimbing Anggota : Dodi Setiabudi, S.T., MT.

PENGESAHAN

Skripsi berjudul "*Surveillance System* Pada Ruang *Server* Berbasis Arduino Ethernet Shield Menggunakan Protokol MQTT" karya Aditya Juli Prasetyo telah diuji dan disahkan pada :

hari, tanggal :

tempat : Fakultas Teknik Universitas Jember.

Tim Penguji:

Ketua,	Anggota I,
Ike Fibriani, S.T., M.T. NIP. 19800207 201504 2 001	Dodi Setiabudi, S.T., M.T. NIP. 19840531 200812 1 004
Anggota II,	Anggota III,
Khairul Anam, ST., MT., Ph.D NIP. 19780405 200501 1 002	Widya Cahyadi, S.T., M.T. NIP. 19851110 201404 1 001

Mengesahkan
Dekan,

Dr. Ir. Entin Hidayah M.U.M
NIP. 19661215 199503 2 001

RINGKASAN

"*Surveillance System* Pada Ruang *Server* Berbasis Arduino Ethernet Shield Menggunakan Protokol MQTT"; Aditya Juli Prasetyo 131910201072; 2018: 114 halaman; Program Studi Strata 1 (S1) Teknik, Jurusan Teknik Elektro, Fakultas Teknik Universitas Jember.

Ruang *server* adalah ruangan dimana terdapat banyak informasi yang disimpan, data penting, dan berbagai peralatan yang terhubung dalam jaringan pada suatu perusahaan. Kondisi ruangan ini harus sesuai dengan standar yang telah ditentukan badan standarisasi yang di khususkan menstandarisasi kondisi pada ruang server. Keadaan ruangan ini juga harus terjaga dari ancaman baik dari luar ruangan maupun dari dalam ruangan. Pengamanan yang akan digunakan adalah pengamanan fisik pada ruangan ini, bertujuan untuk memonitoring suhu, kelembaban, dan objek di depan *server*.

Kendala yang dihadapi ketika membangun *surveillance system* yaitu pada pengiriman data terkadang tidak sesuai dengan keinginan kita, sehingga hasil yang diharapkan dari sistem ini tidak sesuai target. Pemilihan protokol merupakan salah satu cara agar mendapatkan performa transmisi data yang di inginkan. Untuk penelitian ini menggunakan protokol MQTT sebagai protokol utama yang akan diuji performanya dan akan dibandingkan dengan protokol HTTP untuk mendapatkan nilai performansi yang di inginkan.

Pada penelitian ini dilakukan pemberian *noise* yang diharapkan akan dapat melihat perbandingan antara protokol yang dipilih yaitu protokol MQTT dan protokol HTTP. Perubahan pada *noise* yang akan dilakukan yaitu berupa waktu sebesar 1 s dan 13 s dengan skema elektronika yang sama. Sebelumnya dilakukan pengujian ethernet shield, sensor suhu, sensor kelembaban, sensor jarak, server broker, monitoring client, QoS level 0, QoS level 1, QoS level 2. Pengujian QoS level ini bertujuan untuk mendapatkan data *packet loss* dan *delay*.

Penerapan protokol MQTT pada *surveillance system* ini bertujuan untuk mengetahui *packet loss* dan *delay* serta menunjukkan bahwa protokol ini lebih baik daripada protokol HTTP. Pada penelitian sebelumnya yang di kutip dari Priatmoko (2016) dan Tetsuya Yokotani (2016) pada kedua penelitian itu menggunakan protokol MQTT dan HTTP

sebagai perbandingan performa, untuk penelitian sebelumnya menggunakan simulasi sehingga hasil sesuai dengan teori untuk sensor juga menggunakan sensor virtual yang datanya langsung dapat dikirim ke *server*. Melalui penerapan protokol MQTT pada sistem ini diharapkan dapat meraih protokol dengan *packet loss* rendah dan *delay* yang rendah juga. Dari hasil pengujian yang telah di lakukan, semakin tinggi *level* QoS maka semakin bagus nilai *packet loss* nya. Maka dapat disimpulkan bahwa nilai *packet loss* berbanding terbalik dengan *level* QoS yang digunakan. Untuk kualitas dari nilai *packet loss* berbanding lurus dengan *level* QoS yang digunakan. Sedangkan untuk *delay* semua QoS *level* bernilai bagus yaitu 0 s.

SUMMARY

Surveillance System on the Server Room Based on Arduino Ethernet Shield Using MQTT Protocol; Aditya Juli Prasetyo, 131910201072; 2018; 114 pages; Departement of Electrical Engineering, Faculty of Engineering, Jember University.

Server room is a room where there is a lot of information stored, important data, and various devices connected in the network in a company. The condition of this room must be in accordance with the standards set by the standard company in particular standardize the conditions on the server room. The condition of this room must also be protected from threats both from outside the room or from the room. The security that will be used is physical security in this room, aiming to monitor temperature, humidity, and object in front of server.

Constraints faced when building a surveillance system that is on the delivery of data sometimes not in accordance with our wishes, so that the expected results of this system does not match the target. Selection of protocols is one way to get the performance of data transmission in want. For this study use MQTT protocol as the main protocol to be tested its performance and will be compared with HTTP protocol to get the desired performance value.

In this research, noise is expected to be able to see the comparison between the selected protocols of MQTT protocol and HTTP protocol. Changes in the noise that will be done is a time of 1 s and 13 s with the same electronics scheme. Previously tested ethernet shield, temperature sensor, humidity sensor, proximity sensor, server broker, client monitoring, QoS level 0, QoS level 1, QoS level 2. QoS testing this level aims to get packet loss and delay data.

Implementation of MQTT protocol on surveillance system aims to find out packet loss and delay and indicate that this protocol is better than HTTP protocol. In the previous study quoted from Priatmoko (2016) and Tetsuya Yokotani (2016) in both studies used the MQTT and HTTP protocols as performance comparisons, for previous studies using simulations so that the results according to the theory for sensors also use virtual sensors whose data can directly sent to the server. Through

the application of MQTT protocol on this system is expected to reach the protocol with low packet loss and low delay as well. From the test results that have been done, the higher the level of QoS the better the value of its packet loss. It can be concluded that the value of packet loss is inversely proportional to the level of QoS used. For the quality of the packet loss value is directly proportional to the level of QoS used. As for the delay all QoS level is good 0 s.

PRAKATA

Bismillahirohmanirohim.

Puji syukur kehadirat Allah SWT atas limpahan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "*Surveillance System* Pada Ruang *Server* Berbasis Arduino Ethernet Shield Menggunakan Protokol MQTT". Skripsi ini disusun sebagai salah satu syarat untuk menyelesaikan program studi strata satu (S1) Jurusan Teknik Elektro Fakultas Teknik Universitas Jember.

Selama penyusunan skripsi ini penulis mendapat bantuan dari berbagai pihak, untuk itu penulis mengucapkan terima kasih kepada:

1. Dosen Pembimbing Utama Ibu Ike Fibriani, S.T., M.T., dan Bapak Dodi Setiabudi, S.T., M.T. yang telah meluangkan waktu, pikiran, dan perhatian dalam penulisan skripsi ini;
2. Khairul Anam, ST., MT., Ph.D selaku Dosen Penguji I, Widya Cahyadi, S.T., M.T. selaku Dosen Penguji II yang telah memberikan kritik dan saran yang sangat membangun demi penyempurnaan skripsi ini;
3. Dr. Ir. Bambang Sri Kaloko, S.T., M.T., selaku Dosen Pembimbing Akademik yang telah membimbing selama penulis menjadi mahasiswa;
4. Ibu Tri Kusmawati dan Bapak Mardiono selaku orangtua tercinta saya yang telah memberikan dukungan moril dan materiil serta kasih sayang yang tak terhingga sepanjang masa;
5. Keluarga H2-12A Kaliurang Green Garden "*Service Center and Design*" yaitu Anjar, Bayu, Rizki, Wandi, Okman yang telah memberi warna dan arti persahabatan dalam perantauan;
6. Keluarga Besar Purbalingga yang berada di jember Ilham, Koko, Maryam, Sugi, Eli, Rima, Estu, Lianti, Imah, Mika, Aji dan teman IMAGARA;
7. Sahabat saya (teman main game apapun) Dhyanika Indrawan yang telah meluangkan waktu untuk mendengarkan keluh kesah saya dan memberi solusi selama skripsi ini dikerjakan;
8. Keluarga Lab TI terkhusus Pak Danang yang telah memberikan seluruh ilmu, pengalaman hidup, pengalaman kerja dan pengalaman organisasi;

9. Seluruh Dosen Teknik Elektro Universitas Jember yang telah memberikan ilmu pengetahuan dan bimbingan selama mengikuti pedidikan di Universitas Jember;
10. Keluarga besar Teknik Elektro khususnya angkatan 2013 (INTEL UNEJ), terimakasih atas dukungan dan motivasi yang kalian berikan;
11. Ny. Melinda Astuti yang telah memberi warna serta semangat dalam proses pengerjaan skripsi ini;
12. Sahabat dari sejak SD, SMP hingga SMK sampai sekarang yang selalu memberikan dukungan untuk menyelesaikan skripsi;
13. Teman-teman Aslab Patrang yang telah membantu memberikan ide, waktu serta tenaganya dalam pengerjaan skripsi ini;
14. Serta semua pihak yang tidak dapat disebutkan satu per satu, terimakasih banyak yang mana telah mendukung dan memberikan semangat dalam penyelesaian skripsi ini;

Penulis menyadari bahwa kesempurnaan hanya milik-Nya sehingga sebagai manusia biasa, penulis selalu terbuka terhadap masukan dan menerima segala kritik dan saran dari semua pihak yang sifatnya membangun demi kesempurnaan skripsi ini. Penulis berharap, semoga skripsi ini dapat bermanfaat bagi pembaca dan tidak lupa juga penulis menyampaikan permohonan maaf yang sebesar-besarnya jika terdapat kesalahan dan kekeliruan di dalam skripsi ini.

Jember, 24 Januari 2018

Penulis

DAFTAR ISI

Halaman

DAFTAR TABEL

Halaman

DAFTAR GAMBAR

Halaman

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Ruang *server* yaitu ruangan yang dikhususkan untuk menyimpan perangkat *server*, jaringan dan aplikasi penting yang digunakan *user*. Peranannya kurang mendapat perhatian, sehingga standardisasi perangkat, desain ruangan, pemeliharaan kurang sesuai. TIA-942-A menyatakan bahwa standar ruang komputer harus sesuai dengan kelas 1 dari TIA-569-C, yang diselaraskan dengan ASHRAE (Badan Standarisasi). Suhu dan kelembaban pada ruang *server* adalah 18° – 27° C (64° - 81° F) dan kelembaban relatif maksimal adalah 60% dengan titik embun maksimum 15° C (59° F) dan tingkat perubahan suhu maksimum: 5° C (9° F) per jam (ASHRAE, 2017).

Sistem keamanan informasi pada ruang *server* dapat dilihat dari dua sisi, yang pertama yaitu *physical security* yang meliputi bagaimana keamanan fisik sebuah ruang *server* dapat terjaga dengan baik dan yang kedua adalah *cyber security* yang meliputi bagaimana ruang *server* aman dari ancaman non-fisik seperti serangan *denial of service attack*, *virus*, *hacker*, *malware*, dll. *Physical security* sering diabaikan dan dianggap tidak penting, perusahaan lebih mewaspadai ancaman *cyber terrorist*, *hacker*, dan *virus*, padahal *vandalism*, pencuri, pegawai yang tidak puas, dan musuh perusahaan berpotensi menimbulkan kerusakan fisik yang akan lebih sulit untuk diperbaiki daripada serangan *virus*, *malware* atau *hacker* dan akan lebih banyak mengeluarkan biaya untuk memperbaikinya. Pada pengaplikasiannya, *cyber security* yang baik harus diimbangi dengan *physical security* yang baik juga agar keamanan informasi aman. *Physical security* merupakan keamanan tahap dasar dari *computer security*. Jika *physical security* tidak terjaga dengan baik, maka informasi, data-data, bahkan *hardware computer* tidak dapat diamankan (Priatmoko, et al., 2016).

Keamanan untuk saat ini menjadi hal yang penting. Banyak cara dilakukan untuk meningkatkan tingkat keamanan, baik untuk keamanan pada perusahaan atau instansi maupun tempat pribadi seperti rumah. Salah satu solusi untuk mencegah adanya tindakan kriminalitas adalah dengan menggunakan "*Surveillance System*"

pada tempat-tempat tertentu. Khususnya tempat yang di dalamnya terdapat barang berharga dan mempunyai nilai lebih yang harus dilindungi. Atau dapat juga diletakkan di tempat yang sering terjadi tindakan kriminal. Karena itu, pada suatu instansi pasti terdapat sistem komunikasi *monitoring* yang disebut "*Surveillance System*". Dengan sistem itu, diharapkan semua aktivitas ataupun perilaku yang mencurigakan dapat dipantau dan amati dari kejauhan dengan peralatan elektronik (Suryansyah, et al., 2014).

Penyampaian suatu informasi membutuhkan sumber informasi, transmisi dan penerima. Informasi yang dikirim dapat berupa teks, gambar, audio, maupun video. Data yang dikirim berupa informasi mempunyai ukuran yang beragam sesuai besarnya *bit* yang dimiliki suatu data atau informasi yang akan dikirim. Oleh karena itu suatu sistem penyampain komunikasi membutuhkan inovasi dalam hal pengiriman data dari pengirim ke penerima, diantaranya yaitu pengiriman informasi yang cepat sehingga setiap informasi dapat dilihat pada saat itu juga (*real time*) (Yokotani & Sasaki, 2016).

Untuk transmisi data dibutuhkan sebuah aturan atau protokol agar sistem berjalan dengan lancar. Salah satu aturan yang digunakan dalam *Surveillance System* adalah protokol MQTT. Protokol MQTT adalah *Messaging protocol* yang "ringan" berjalan diatas *TCP/IP protocol*, didesain untuk daerah/ lokasi dengan *resource* jaringan yang terbatas. MQTT adalah protokol komunikasi yang ringan, terbuka dan sederhana, lebih modern daripada HTTP yang memiliki *protocol overhead* yang rendah, berjalan pada *latency* yang tinggi dan dapat berjalan pada *bandwidth* yang rendah. (Satria, et al., 2015).

Pada penelitian sebelumnya pengujian implementasi protokol MQTT pada OpenMTC dengan parameter *packet loss* dari virtual sensor ke *server* OpenMTC dengan protokol HTTP dan MQTT yaitu 0% dengan topologi LAN dan WLAN sehingga dapat disimpulkan akurasi pengiriman data sensor pada lingkungan pengujian ini adalah 100%. (Satria, et al., 2015).

Pada penelitian ini dimaksudkan untuk menerapkan dan menganalisa protokol MQTT untuk diterapkan pada "*Surveillance System*" ruang *server* yang berada pada Fakultas Teknik Universitas Jember. Metode yang akan digunakan dalam penelitian ini yaitu metode MQTT *publish* dan *subscribe,* serta pengiriman data menggunakan layanan QoS tingkat 0, tingkat 1, dan tingkat 2. Pengujian performansi dilakukan dengan dua parameter uji yaitu *delay* dan *packet loss.* Pada penelitian ini akan digunakan protokol MQTT untuk membuat suatu *prototype* pada "*Surveillance System*" Fakultas Teknik Universitas Jember yang diharapkan pada protokol MQTT memberikan hasil yang baik.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, berikut ini adalah beberapa masalah yang akan diselesaikan dalam penelitian ini yaitu:

1. Bagaimana rancangan *Surveillance System* dengan menggunakan protokol MQTT?
2. Bagaimana analisa hasil rancangan *Surveillance System* dengan menggunakan protokol MQTT berdasarkan parameter QoS *level* 0, 1 dan 2 dengan *delay* dan *packet loss*?

1.3 Batasan Masalah

Berdasarkan rumusan masalah penelitian diatas, agar pembahasan tidak terlalu luas maka diperlukan suatu pembatasan masalah sebagai berikut.

1. Hanya membahas tentang Sistem telekomunikasi data dan sistem kerja alat.
2. Tidak membahas tentang elektronika pada alat.
3. Ruang yang digunakan yaitu Ruang *Server* Fakultas Teknik Universitas Jember.
4. Pada pembahasan analisa hanya membahas data pada parameter QoS.
5. Performansi yang akan di ukur adalah parameter QoS *level* 0, 1 dan 2 *delay* dan *packet loss*.
6. Sensor yang digunakan adalah sensor suhu dan kelembaban DHT11 dan sensor ultrasonik.

7. Arduino yang digunakan adalah Arduino Uno.
8. Penghubung jaringan pada Arduino menggunakan Arduino Ethernet Shield.
9. PC dengan OS Windows untuk *server* MQTT.

1.4 Tujuan Penelitian

Penelitian yang diusulkan dalam proposal ini memiliki beberapa tujuan sebagai berikut:

1. Merancang dan membuat alat *Surveillance System* yang dapat di akses pada jaringan yang sudah terintegrasi.
2. Menganalisa dengan mengetahui parameter suhu, kelembaban, dan jarak objek untuk *Surveillance System* menggunakan protokol MQTT.

1.5 Manfaat Penelitian

Penelitian ini diharapkan bermanfaat sebagai:

1. Langkah awal dalam pengembangan sistem *Surveillance System* pada ruang *server* berbasis Arduino Ethernet Shield menggunakan protokol MQTT Fakultas Teknik Universitas Jember.
2. Mengetahui kualitas dengan menganalisa cara kerja *Surveillance System* dengan menggunakan protokol MQTT berdasarkan parameter QoS.

BAB 2. TINJAUAN PUSTAKA

Bab tinjauan pustaka ini, diulas berbagai publikasi resmi yang berhubungan dengan konsep Alat *Surveillance System* Pada Ruang *Server* Berbasis Arduino Ethernet Shield Menggunakan Protokol MQTT dan mencakup aspek masalah dan penjelasan faktor-faktor yang diduga berkaitan dengan penelitian ini. Seluruh teori dan konsep pada tinjauan pustaka ini pada akhirnya nanti akan digunakan untuk menunjang analisis pembahasan terhadap hasil penelitian yang akan dilakukan. Berikut ini adalah teori dan konsep yang berhubungan dengan masalah studi analisis yang akan dibahas.

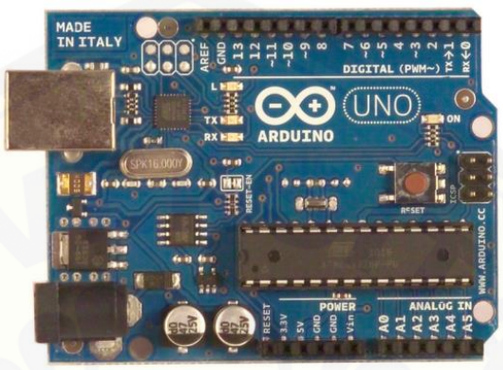
2.1 Pengertian *Surveillance*

Surveillance adalah pemantauan yang dapat digambarkan sebagai kesadaran (*awareness*) tentang apa yang ingin kita ketahui, pemantauan berkadar tingkat tinggi dilaksanakan supaya dapat membuat pengukuran pada waktu yang akan menunjukkan pergerakan ke arah tertentu atau menjauh dari arah tersebut. *Monitoring* akan memberikan info tentang status dan kecenderungan bahwa pengukuran dan evaluasi yang telah diselesaikan secara berulang dari waktu ke waktu, pemantauan pada umumnya dilakukan untuk tujuan tertentu, untuk memeriksa terhadap proses dan objek atau untuk mengevaluasi kondisi dan kemajuan menuju tujuan hasil manajemen berdasarkan efek tindakan untuk mempertahankan manajemen yang sedang berjalan.

2.2 Arduino Uno R3

Arduino berasal dari Italia (bahasanya), *ardui* yang berarti sulit dan *no* yang berarti tidak sehingga arti kata Arduino adalah tidak sulit atau mudah. Arduino adalah *platform prototype* berbasis *open-source* elektronik yang dapat digunakan baik dari *hardware* maupun *software* dan perangkatnya memiliki *processor* Atmel AVR Atmega 328. Arduino mempunyai *input* yang dapat menerima dari berbagai sensor dan *output* sebagai pengendali seperti motor, lampu, dan sensor lainnya. Arduino Uno diprogram menggunakan bahasa pemrograman berdasarkan *wiring*

berbasis bahasa C yang disederhanakan dengan bantuan *library* dan dalam lingkup pengembang berasaskan *processing*. Arduino bisa bekerja mandiri atau dapat berkomunikasi dengan perangkat lain seperti komputer.



Gambar 2.1 Arduino Uno R3 (sumber : Anonim, 2015)

2.2.1 Spesifikasi Arduino Uno R3

Arduino memiliki spesifikasi yang baik walaupun merupakan *microcontroller* yang mudah untuk digunakan. Berikut merupakan spesifikasi Arduino (Sumber: *Datasheet* Arduino Uno R3 Atmega328).

a. *Microcontroller* Atmega 328
b. *Input* daya 5 v *Volt*age
c. *Input* tegangan yang disarankan 7 - 12 V
d. *Input* tegangan dengan batas 6 - 20 v
e. Digital I/O 14 pin dimana pin 6 memberikan *output* PWM
f. Analog *input* pin 6
g. DC lancar per I/O pin 40 mA
h. *Flash memory* 32 KB (Atmega 328) yang 0,5 KB digunakan oleh *bootloader*
i. SRAM 2 KB (Atmega 328)
j. EEPROM 1 KB (Atmega 328)

2.3 Ethernet Shield

Ethernet Shield dapat menambah kemampuan Arduino Uno agar terhubung ke jaringan komputer. Ethernet Shield menggunakan *chip* Ethernet Wiznet W5100. Ethernet *library* digunakan dalam *write* program agar Arduino Uno dapat terhubung pada jaringan dengan menggunakan Arduino Ethernet Shield. Pada Ethernet Shield terdapat satu buah *slot micro*-SD yang dapat digunakan untuk menyimpan *file* yang dapat diakses dari jaringan. *Onboard micro*-SD diakses dengan menggunakan SD *library* pada Arduino. Arduino dengan tipe W5100 dan SD *card* mengunakan SPI (*Serial Peripheral Interface*). Komunikasi ini diatur oleh pustaka SPI.h dan Ethernet.h. SPI menggunakan pin digital 13, 12, dan 11 pada Arduino. Pin digital 10 digunakan untuk memilih tipe W5100 dan pin digital 4 digunakan untuk memilih jalur SD *card.*

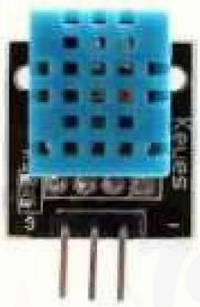
Gambar 2.2 Arduino Ethernet Shield (sumber : Anonim, 2015)

2.4 DHT11

DHT11 adalah sensor suhu dan kelembaban (*air temperature sensor*), dia memiliki keluaran sinyal digital yang sudah dikalibrasi dengan menggunakan sensor suhu dan kelembaban yang kompleks. Teknologi ini memastikan keandalan yang tinggi dan stabilitasnya sangat baik dalam jangka panjang. *Microcontroller* terhubung pada kinerja tinggi sebesar 8 *bit.* Sensor ini termasuk elemen yang resistif dan perangkat pengukur suhu NTC. DHT11 mempunyai kemampuan anti gangguan, respon cepat, kualitas yang sangat baik dan keuntungan biaya tinggi kinerja.

2.4.1 Spesifikasi DHT11

a. Pasokan *Voltage*: 5 V
b. Rentang temperatur :0-50° C kesalahan ± 2° C
c. Kelembaban :20-90% RH ± 5% RH *error*
d. *Interface:* Digital



Gambar 2.3 Sensor DHT11 (sumber : Santoso, 2015)

Gambar diatas merupakan bentuk fisik dari sensor DHT11.Pada saat suhu ruangan berubah maka, nilai resistansinya sensor DHT11 akan berubah. Sensor ini sebagai pengindra yang merupakan elemen yang pertama-tama menerima energi dari media untuk memberi keluaran berupa perubahan energi.

2.5 Sensor Ultrasonik HC-SR04

Sensor Ultrasonik adalah sensor yang bekerja berdasarkan prinsip pantulan gelombang suara yang digunakan untuk mendeteksi keberadaan suatu objek tertentu di depannya. Cara kerja sensor ini berdasarkan pada prinsip dari pantulan suatu gelombang suara sehingga dapat digunakan untuk menafsirkan eksistensi (jarak) suatu benda dengan frekuensi tertentu (Santoso, 2015).

Dengan menggunakan pantulan gelombang suara, sensor ini dapat digunakan sebagai sensor jarak. Cara kerja dari sensor ultrasonik untuk mengukur jarak adalah sebagai berikut (Santoso, 2015). Sinyal tersebut dipancarkan oleh pemancar ultrasonik dengan frekuensi tertentu dan durasi waktu tertentu. Sinyal tersebut berfrekuensi diatas 20kHz. Untuk mengukur jarak benda, frekuensi yang

umum digunakan adalah 40kHz. Sinyal yang dipancarkan akan merambat sebagai gelombang bunyi dengan kecepatan 340 m/s. Ketika menabrak suatu benda, maka sinyal tersebut akan dipantulkan oleh benda tersebut.

HC-SR04 mempunyai 4 pin, GND, ECHO, TRIG, dan VCC. VCC dihubungkan dengan 5 V dari Arduino, GND dengan GND pada Arduino, TRIG terhubung pada pin digital 12 dan ECHO disambungkan dengan pin digital 13. (Zerfani Yulias, 2011)

Gambar 2.4 Sensor Ultrasonik HC-SR04 (sumber : Santoso, 2015)

Sensor ini adalah tranceiver, bertindak sebagai pengirim sekaligus sebagai penerima. Cara kerjanya mirip kelelawar, yaitu dengan menembakkan sinyal ultrasonik lalu setelah terpantul benda didepannya, sinyal tersebut akan diterima kelelawar untuk menentukan jarak antara dirinya dengan benda didepannya. Untuk menghubungkan HC-SR04 dengan Arduino sangat mudah tanpa perlu komponen lainnya seperti resistor atau kapasitor. (Ibnu Kusumayadi, 2014)

2.5.1 *Datasheet* Sensor Ultrasonik HC-SR04 :

a) *Working Volt*age: DC 5 V
b) *Working Current*: 15 mA
c) *Working Frequency*: 40 Hz
d) *Max Range*: 4 m
e) *Min Range*: 2 cm
f) *Measuring Angle*: 15 *degree*
g) *Trigger Input* Signal: 10 µS TTL *pulse*

h) *Echo Output* Signal *Input* TTL *level signal and the range in proportion*
i) *Dimension* 45 * 20 * 15 mm

2.5.2 Spesifikasi dari HC-SR04 :

a) *Supply* tegangan 5 V DC.
b) Arus *Quiescent* < 2 mA.
c) Sudut efektif < 15°.
d) Jarak pengukuran 2 – 500 cm.
e) Resolusi 0.3 cm.

2.5.3 Pin dari HC-SR04 :

a) VCC : *Input* supply 5 V
b) Trig : *Input* untuk memberikan pulsa *trigger*
c) Echo : *Output* untuk pulsa *Echo*
d) GND : *Input* supply *Ground*

2.6 *Quality of Service* (QoS)

QoS dikhususkan untuk membantu *user* agar lebuh produktif dengan memastikan bahwa *user* mendapatkan kinerja yang handal dari aplikasi – aplikasi berbasis jaringan. QoS ini sendiri mengacu pada kemampuan pada jaringan untuk menyediakan layanan yang baik pada jalur jaringan tertentu melalui teknologi yang berbeda - beda. QoS merupakan tantangan yang besar dalam jaringan berbasis *internet* secara keseluruhan. QoS adalah kemampuan suatu jaringan untuk menyediakan layanan yang baik dengan mengatasi *delay* dan *jitter* serta menyediakan kapasitas jaringan. Pada teknologi Qos ini memungkinkan dalam pengoptimalan jaringan yang terjadi (Yonathan at al, 2011).

2.6.1 Parameter Qos

Parameter QoS adalah parameter dimana kualitas dari jaringan itu diukur. Parameter-parameter tersebut adalah sebagai berikut.

a. Packet Loss

Adalah suatu parameter yang menampilkan suatu kondisi yang menunjukan jumlah total paket yang hilang, hal tersebut dapat terjadi karena *collision* dan *congestion* pada jaringan. Untuk nilai *packet loss* sesuai dengan versi *Telecomunications and Internet Protocol Harmonuzation Over Network* (TIPHON) sebagai berikut:

Tabel 2.2 *Packet Loss*

Kategori Degradasi	*Packet Loss*	*Indeks*
Sangat Bagus	0	4
Bagus	3	3
Sedang	15	2
Jelek	25	1

(sumber : Sutarman, 2009)

Persamaan perhitungan *packet loss*:

$$Packet\ loss\ (\%) = \frac{Paket\ total\ tercapture - paket\ terkirim}{paket\ total\ tercapture}\ x\ 100\% \quad (2.1)$$

b. Delay

Yaitu waktu yang dibutuhkan data untuk menempuh jarak dari asal ke tujuan. *Delay* dapat dipengaruhi oleh kongesti, media fisik, jarak, atau waktu proses yang lama.

$$Delay\ (s) = \frac{waktu\ paket\ yang\ dikirim}{jumlah\ paket} \quad (2.2)$$

2.7 Protokol

Protokol dalam ilmu komputer adalah satu set aturan atau prosedur untuk mentransmisikan data antar perangkat elektronik, seperti komputer agar dapat bertukar atau mendapatkan informasi dan harus ada kesepakatan yang sudah ada mengenai bagaimana informasi akan terstruktur dan bagaimana masing - masing

pihak akan mengirim dan menerimanya. Tanpa sebuah protokol, sebuah komputer mentransmisikan paket data 8-*bit* ke komputer penerima, tetapi komputer penerima mengharapkan data dalam paket 16-*bit*. Protokol dibuat oleh organisasi internasional atau industri.

2.8 HTTP

HTTP merupakan suatu protokol *request*/ *reply* antara *client* dan *server*. Untuk *client* HTTP seperti *web browser*, memulai permintaan dengan membuat sambungan TCP/IP ke *port* tertentu di *localhost* yang jauh (biasanya *port* 80). Sebuah *server* HTTP memberikan *listen* pada *port* tersebut menunggu *client* mengirim *request*, seperti "GET / HTTP/1.1" yang akan meminta halaman yang sudah ditentukan, dilanjutkan dengan pesan MIME yang memiliki beberapa informasi pada *header* yang menjelaskan aspek dari permintaan tersebut. Beberapa *header* juga bebas ditulis atau tidak, sementara lainnya (*localhost*) diperlukan oleh protokol HTTP/1.1. Begitu menerima *request* (pesan, bila ada), *server* mengirim kembali *reply*, seperti "200 OK", dan sebuah pesan yang diminta, atau pesan *error* atau pesan lainnya.

Pengembangan HTTP dikoordinasi oleh Konsorsium World Wide Web (W3C) dan grup bekerja Internet Engineering Task Force (IETF), bekerja dalam publikasi satu seri RFC, yang paling terkenal RFC 2616, yang menjelaskan HTTP/1.1, versi HTTP yang digunakan umum sekarang ini (Abdul Zabar & Novianto, 2015).

2.9 MQTT

Message Queuing Telemetry Transport (MQTT) merupakan protokol transport yang bersifat *client server publish/ subscribe*. Protokol yang terbuka, sederhana, ringan, dan dirancang agar mudah diimplementasikan. Karakteristik ini membuat MQTT bisa digunakan di banyak kondisi, termasuk penggunaanya dalam komunikasi *machine-to-machine* (M2M) dan *Internet of Things* (IoT). Protokol ini berjalan pada jaringan TCP/IP. Protokol MQTT membutuhkan transportasi yang menjalankan perintah MQTT, *byte stream* dari *client* ke *server* atau *server* ke *client.*

Protokol *transport* yang digunakan adalah TCP/IP. TCP/IP dapat digunakan untuk MQTT, selain itu TLS dan WebSocket juga dapat menggunakan TCP/IP. (Satria, Satrya, & Herutomo, 2015)

Berikut merupakan fitur protokol MQTT:

a. *Publish/ subscribe message pattern* yang menyediakan distribusi *message* dari satu ke banyak dan *decoupling* aplikasi.
b. *Messagging transport* yang *agnostic* dengan isi dari *payload.*
c. Menggunakan TCP/IP sebagai konektivitas dasar jaringan.
d. Terdapat tiga *level Qualities of Service* (Qos) dalam penyampaian pesan :
   1) "*At most once*", di mana pesan dikirim dengan upaya terbaik dari jaringan TCP/IP. Kehilangan pesan atau terjadi duplikasi dapat terjadi.
   2) "*At least once*", dapat dipastikan pesan tersampaikan walaupun duplikasi dapat terjadi.
   3) "*Exactly once*", dimana pesan dapat dipastikan tiba tepat satu kali.

2.9.1 MQTT Client

Setiap objek IOT dapat menjadi MQTT *client* yang mengirim atau menerima data telemetri. Jenis MQTT *client* (*subscriber* atau *publisher*) tergantung pada perannya dalam sistem serta dapat memproduksi atau mengumpulkan data telemetri. Dalam kedua kasus, MQTT *client* harus pertama kali terhubung ke *server* menggunakan jenis tertentu dari pesan. Setelah sambungan berhasil dibuat, klien harus mendeklarasikan dirinya sendiri apakah dia adalah *subscriber* atau *publisher.* Dalam urutan untuk membedakan data yang dikirim oleh publisher, digunakan topik untuk membedakannya. Misalnya, klien dapat mempublikasikan suhu dan kelembaban menggunakan nilai *string* yang berbeda untuk setiap nilai (misalnya temp/ 25 atau hum/ 40). Di sisi lain, jika *client* MQTT ingin menerima data, ia harus *subscribe* topik tertentu. Untuk membuat MQTT *client* dari sebuah perangkat, diperlukan *library* yang sudah terinstal dan terkoneksi ke MQTT *broker* dengan penetapan jenis jaringan. Misalnya, MQTT *client* bisa menjadi komputer kecil (Arduino atau Raspberry Pi) yang terhubung ke jaringan *wireless* dengan *library* yang terinstal ke minimum atau komputer yang menjalankan program grafis. *Client*

yang mengimplementasi protokol MQTT menjadi sederhana. *Library* pada *client* dapat menyederhanakan proses *writing* MQTT pada aplikasi *client. Library* MQTT tersedia untuk berbagai jenis bahasa pemrograman dan *platform*, misalnya, JavaScript, PHP, C, C ++, Android, iOS dll. Untuk implementasi awal, Python *library* akan digunakan. (Grgić, et al., 2016)

2.9.2 MQTT *Broker*

MQTT *broker* adalah perangkat sentral dalam model ini. Tanggung jawab utama dari sebuah *broker* MQTT adalah penanganan komunikasi antara MQTT *client* dan mendistribusikan pesan di antara mereka. MQTT *broker* dapat menangani ribuan MQTT *client* yang terhubung pada waktu yang sama. Ketika pesan diterima, *broker* harus menemukan semua *client* yang memiliki *subscription* ke topik yang diterima. Dalam aplikasinya, ia bertanggung jawab untuk menerima pesan dari sensor yang terhubung ke perangkat dengan menerapkan MQTT *client library* dan *forwarding* ke perangkat *mobile* yang ditunjuk. Ada banyak tugas lain dan tanggung jawab yang ditangani oleh *broker*. Pertama-tama, ada otentikasi dan otorisasi *client* untuk tujuan keamanan. Seorang klien dapat mengirim username dan password dalam pesan yang terhubung dan akan diperiksa oleh *broker*. Pada topik sisi *broker* izin diimplementasikan untuk membatasi klien untuk *publish* atau *subscribe*. Selanjutnya, untuk keamanan komunikasi yang terenkripsi antara *broker* dan *client* menggunakan *Transport Layer Security* (TLS) dan *Secure Sockets Layer* (SSL), dimana protokol keamanan yang sama yang dapat digunakan oleh protokol HTTP. Hal ini juga memungkinkan untuk menerapkan kustom otentikasi atau logika otorisasi ke dalam sistem. Ada beberapa *broker* pesan yang menerapkan protokol MQTT seperti Mosquitto - open source MQTT v3.1 / v3.1.1. *broker* dan HiveMQ - *broker* perusahaan MQTT. (Grgić, et al., 2016)

2.9.3 MQTT *over WebSockets*

Menggunakan MQTT dengan *WebSockets* pada setiap *browser* dapat menjadi perangkat MQTT. Karena *publish*/ *subscribe* pola MQTT dengan komunikasi *real-time* antara *end devices* (misalnya suhu sensor) dan perangkat

yang me*monitor* (misalnya aplikasi web) tercapai. Menggunakan QoS 1/2 pesan akan tiba pada *client* atau *broker* setidaknya sekali. *Broker* akan membuat antrian semua pesan jika *client* tidak tersambung. Pesan yang dipertahankan pada *server* akan disampaikan ketika *client* melakukan *subscribe* ke salah satu topik secara langsung. (Grgić, et al., 2016)

2.9.4 Format pesan

Header pesan untuk setiap pesan perintah MQTT berisi *header* tetap dengan panjang 2 *byte* dan pesan khusus yang opsional dengan variabel panjang *header* dan *message payload*. (Grgić, et al., 2016)

Tabel 2.3 Format pesan MQTT

Bit	7	6	5	4	3	2	1	0
Byte 1	*Message type*				DUP	QoS		RETAIN
Byte 2	*Remaining length*							

(sumber : Grgić, et al., 2016)

Byte 1 berisi jenis pesan yang diwakili sebagai 4-*bit* nilai yang belum ditandai. Ada 14 jenis pesan yang ditentukan dalam versi 3.1 protokol MQTT, diantaranya adalah :

a. CONNECT (permintaan *client* untuk terhubung ke *server*).
b. PUBLISH (*publish* pesan).
c. PUBACK (*publish* pengakuan atau *acknowledgement*).
d. SUBSCRIBE (permintaan *client* untuk *subscribe*).
e. CONNACK (*Connection Acknowledgement*).
f. PUBACK (QoS 1 merespon pesan PUBLISH).
g. PUBREC (Bagian pertama pada QoS 2 *message flow).*
h. PUBREL (Bagian kedua pada QoS 2 *message flow).*
i. PUBCOMP (Bagian terakhir pada QoS 2 *message flow).*
j. SUBACK (Pengakuan pada pesan SUBSCRIBE).

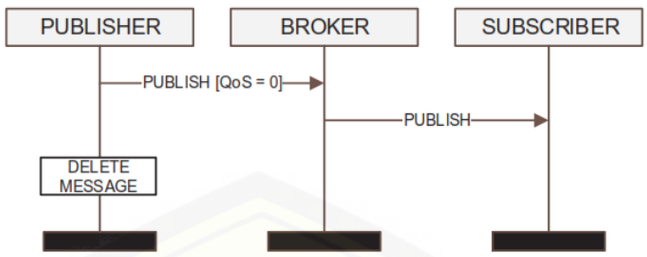
k. UNSUBSCRIBE (Pesan yang digunakan *clients* untuk melakukan pembatalan *subscribe* dari topik tertentu.
l. UNSUBACK (pengakuan untuk pesan UNSUBSCRIBE).
m. PINGREQ (*request* koneksi).
n. PINGRESP (respon pada *request* koneksi).
o. DISCONNECT (memberi pesan *request* sebelum melakukan pemutusan jaringan).

Flag DUP diatur ketika *client* atau *server* mencoba untuk kembali memberikan PUBLISH, PUBREL, SUBSCRIBE atau UNSUBSCRIBE. Hal ini berlaku untuk pesan di mana nilai QoS lebih besar dari nol (0), dan pengakuan diperlukan. QoS *flag* menunjukkan tingkat jaminan untuk pengiriman pesan PUBLISH. Tingkat QoS dijelaskan dalam bab berikutnya. Jika RETAIN *flag* diatur ke *true,* pesan MQTT yang normal menjadi dipertahankan. *Broker* akan mengembalikan pesan ditahan dengan kesesuaian QoS untuk topik tertentu. Setiap klien yang berlangganan pola topik ini akan segera menerima pesan yang dipertahankan. *Broker* akan menyimpan satu pesan ditahan per topik. *Remaining length* mewakili jumlah *byte* tersisa dalam pesan, termasuk data di variabel *header* dan *payload*. (Grgić, et al., 2016)

2.9.5 QoS *level*

a. *Level* 0

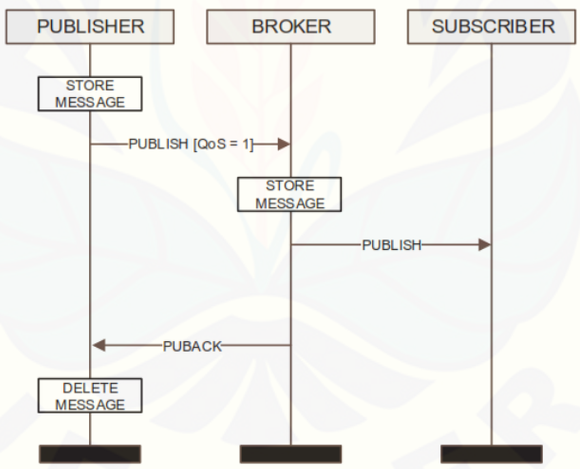
Pesan dikirim hanya satu kali. Data pesan yang terkirim tergantung dari reliabiliti stack TCP alias tergantung kondisi *network* dan tidak ada usaha untuk mengirim pesan kembali (Grgić, et al., 2016).



Gambar 2.6 QoS *level* 0 (sumber : Grgić, et al., 2016)

b. *Level* 1

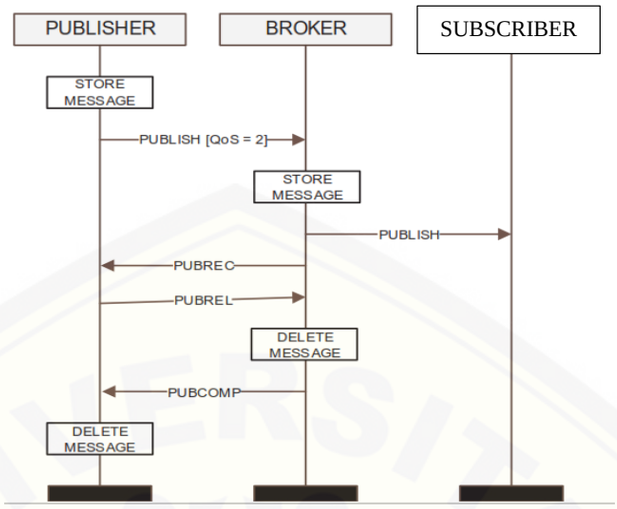
Pesan dikirim setidaknya satu kali pengiriman. Jadi klien setidaknya akan menerima pesan satu kali. Jika *subscriber* tidak mengakui pesan (*acknowledge*) maka *broker* akan mengirimkan pesan sampai pengirim (*publisher)* menerima status pengakuan pesan dari *client* (Grgić, et al., 2016).



Gambar 2.7 QoS *level 1* (sumber : Grgić, et al., 2016)

c. *Level* 2

Pesan pasti diterima sekali. Protokol dengan *level* ini dapat memastikan bahwa pesan pasti tersampaikan dan tidak terjadi duplikasi data yang terkirim (Grgić, et al., 2016).



Gambar 2.8 QoS *level* 2 (sumber : Grgić, et al., 2016)

2.10Wireshark

Wireshark adalah sebuah aplikasi perangkat lunak *(software)* yang dipakai untuk dapat melihat macam - macam paket pada jaringan dan berusaha untuk menampilkan semua informasi di paket tersebut sedetail mungkin. *Open Source* dari Wireshark menggunakan *Graphical User Interface* (GUI). Wireshark sering disebut juga *Network packet analyzer*. Dengan adanya wireshark ini semua sangat dimudahkan dalam hal *monitoring* dan menganalisa paket yang lewat di jaringan (Sihombing & Zulfin, 2013).

2.11Standar Ruang *Server*

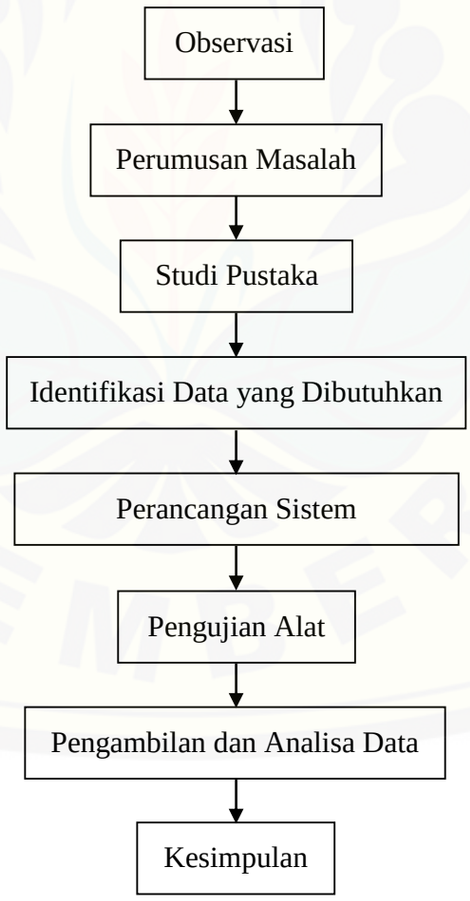
ASHRAE adalah badan yang menentukan standar suhu dan kelembaban ruang *server*. TIA-942-A telah diperbarui untuk memodifikasi rentang operasi yang lebih rendah untuk kelembaban berdasarkan titik embun daripada kelembaban relatif. Hasilnya adalah bahwa TIA-942-A menyatakan bahwa ruang komputer harus sesuai dengan kelas 1 dari TIA-569-C, yang diselaraskan dengan ASHRAE. Suhu dan kelembaban pada ruang *server* adalah 18-27°C (64-81°F) dan kelembaban relatif maksimal adalah 60% dengan titik embun maksimum 15°C (59°F) dan tingkat perubahan suhu maksimum: 5°C (9°F) per jam (ASHRAE, 2017).

BAB 3. METODE PENELITIAN

Pembahasan pada bab metode penelitian ini dijelaskan beberapa hal pokok yaitu obyek penelitian, tahap penelitian, tempat dan waktu penelitian, alat dan bahan, langkah-langkah dalam pengambilan data dan manajemen penelitian di lapangan, pengolahan data serta *software* yang digunakan dalam penelitian.

3.1 Tahap Penelitian

Penyusunan laporan ini memiliki beberapa tahap untuk memperoleh informasi yang dibutuhkan, adapun tahap pengambilan data yang digunakan dalam penelitian ini adalah sebagai berikut :

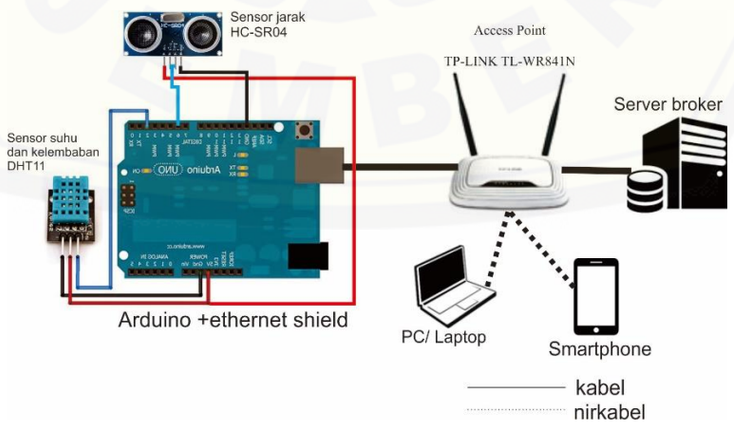


Gambar 3.1 Kerangka penelitian

Tahap penelitian ini dilakukan observasi dengan melihat dan menentukan tema objek yang akan diteliti. Selanjutnya merumuskan suatu masalah dari beberapa masalah yang terdapat dalam kehidupan sehari-hari yang berhubungan dengan objek yang diteliti. Tahap berikutnya mengumpulkan dan mempelajari literatur tentang metode *Surveillance System* dengan menggunakan protokol MQTT. Setelah mengumpulkan landasan teori hal yang perlu dilakukan adalah mengidentifikasi data yang dibutuhkan untuk menganalisa performa dari alat yang akan di buat. Setelah itu kita dapat menentukan *software* dan *user interface* yang digunakan. Tahap selanjutnya merancang atau membuat alat *Surveillance System* dengan menggunakan protokol MQTT. Jika alat telah selesai dirancang tahap selanjutnya yaitu pengujian sensor dengan cara kalibrasi. Selanjutnya untuk pengambilan data dilakukan dengan parameter QoS *level* 1, 2, dan 3. Kemudian diberi *delay* sebagai *noise*. Tahap akhir dari penelitian ini yaitu membuat kesimpulan.

3.2 Perancangan Alat

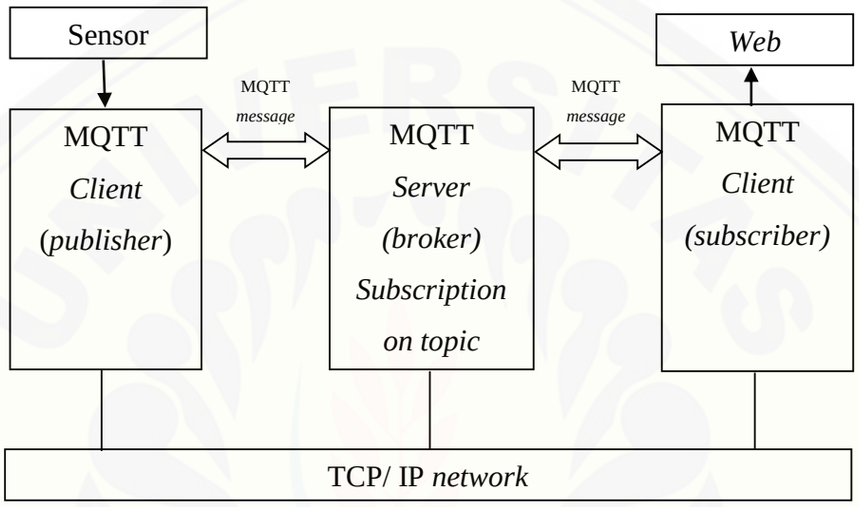
Perlu dilakukan perencanaan dengan membuat perancangan alat, topologi yang digunakan, skenario pengujian, dan hasil pengujian. Pada perancangan alat terdapat blok diagram, diagram alir, algoritma, dan desain alat. Blok diagram menjelaskan tentang rancangan sistem. Pada diagram alir dijelaskan proses jalannya alat mulai dari *on* hingga *off*. Pada desain alat di jelaskan perancangan pada *Surveillance System* dengan menggunakan protokol MQTT.



Gambar 3.2 Topologi jaringan pengujian protokol MQTT

Pada Gambar 3.2 dijelaskan rancangan elektonika pada alat. Data sensor akan dikirimkan melalui Arduino Ethernet Shield dengan media kabel LAN. Data yang dikirim akan diteruskan menuju *access point* dan dikirim lewat media nirkabel menuju *server broker*. Dari *server broker* dapat diambil oleh *end devices*.

3.2.1 Blok Diagram

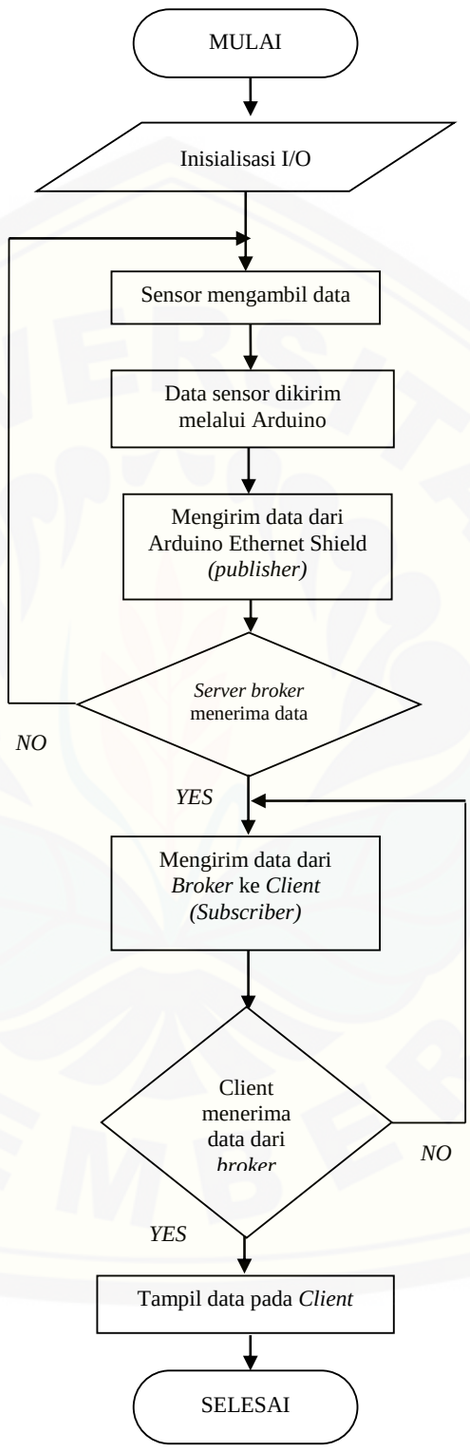


Gambar 3.3 Blok diagram *surveillance system*

Gambar 3.3 merupakan gambar diagram blok dari skema yang akan di buat, alat tersebut akan terbagi menjadi tujuh bagian, dengan fungsi sebagai berikut :

a. Sensor ultrasonik dan DHT11 yang berfungsi sebagai *monitoring* pada ruang *server* Fakultas Teknik Universitas Jember.
b. MQTT *Client (publisher)* sebagai jembatan antara sensor dengan *server*, disini menggunakan Arduino Ethernet Shield.
c. MQTT *Server* (*broker*) sebagai penampung data dari publisher dan sebagai penampung request dari subscriber.
d. MQTT *Client (subscriber)* sebagai jembatan antara *end user* dengan *server*, disini menggunakan laptop/ pc/ smartphone.
e. *Web Browser* sebagai tampilan dari data sensor ultasonik dan DHT11.
f. TCP/ IP *network* sebagai penghubung antara *publisher, broker*, dan *subscriber*.

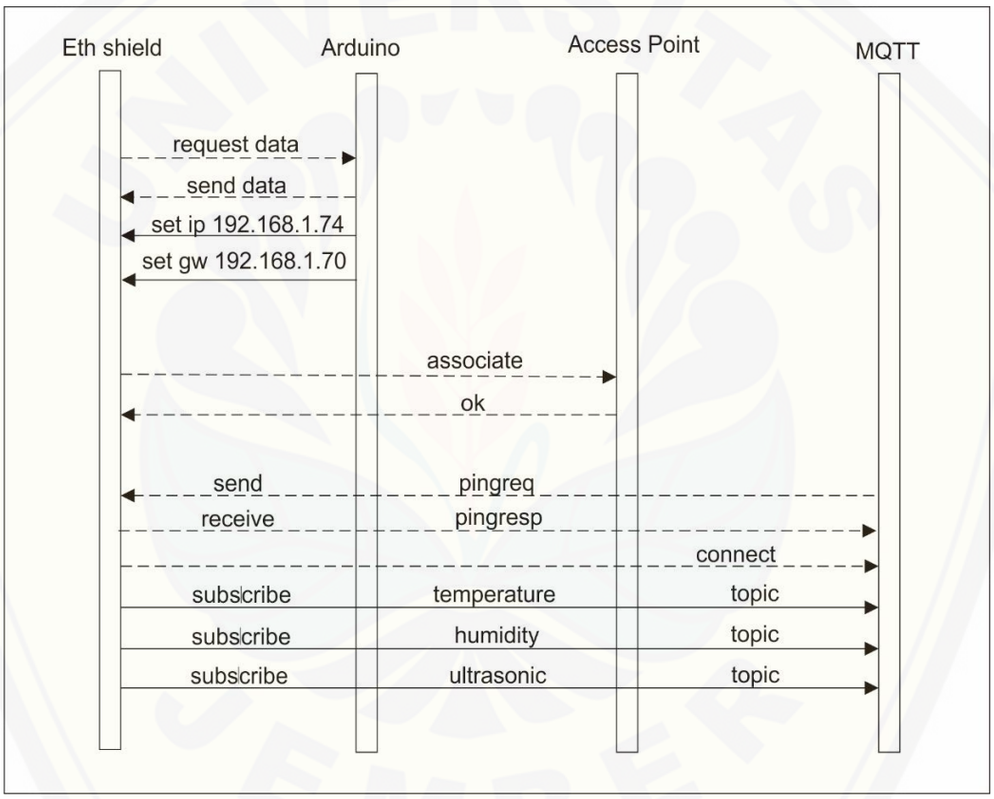
3.2.2 *Flowchart*



Gambar 3.4 *Flowchart*

Gambar 3.4 merupakan gambar langkah-langkah dalam pengambilan data. Sensor akan mendeteksi suhu pada ruang *server* dengan mengawali inisialisasi *input* dan *output*. Data tersebut lalu dikirim melalui Arduino Ethernet Shield menuju *server broker* MQTT. Setelah itu data yang sudah berada pada *server broker* MQTT akan dikirim ke *client* agar data dari sensor dapat dibaca oleh *Administrator*. Dari data yang didapat akan dianalisis pada QoS *level* 0, 1, dan 2.

3.2.3 Algoritma MQTT



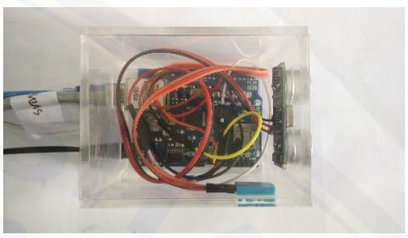
Gambar 3.5 Algoritma protokol MQTT

Pada Protokol MQTT menggunakan Algoritma yang terbentuk oleh *surveillance system* yang dibuat. Untuk komunikasi pada protokol ini menggunakan Arduino Ethernet Shield, *access point*, dan *server broker* MQTT. Untuk set IP pada Ethernet Shield digunakan pemrograman pada Arduino. Untuk komunikasi pada sensor juga melewati Arduino terlebih dahulu. Pada komunikasi Arduino Ethernet

Shield dengan *server broker* MQTT menggunakan PINGREQ dan PINGRESP, setelah *device* dikenali maka akan mengirimkan data dengan topik tertentu yang sudah dibuat.

3.2.4 Desain Alat

(b) Tampak samping



(a) Tampak atas

(c) Tampak depan

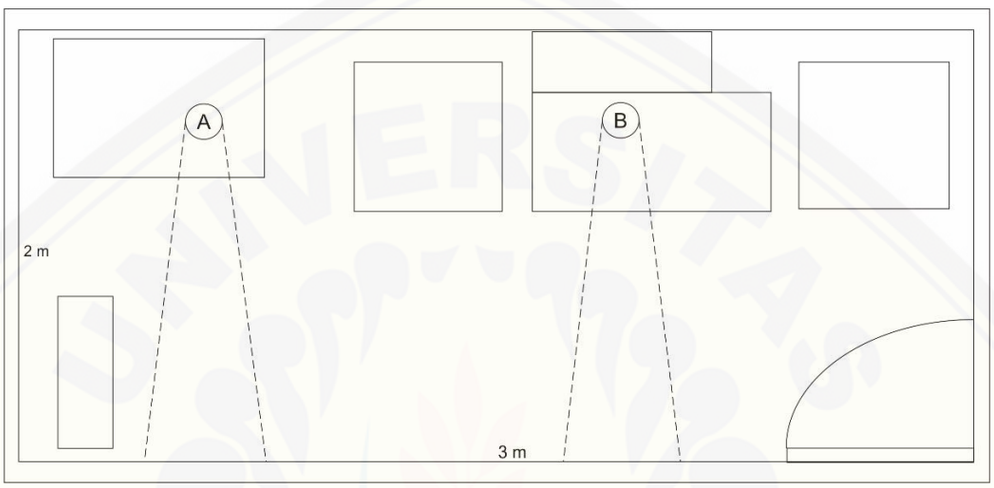
Gambar 3.6 Desain fisik Alat

Pada sensor ultasonik dan DHT11 akan digabungkan dengan Aduino Uno dengan Arduino Ethernet Shield sebagai media transmisinya. Untuk sensor DHT11 disambungkan pada pin 7 pada Arduino Ethernet Shield. Sedangkan untuk sensor ultrasonik disambungkan pada pin 6 pada Arduino Ethernet Shield. Untuk mengirim data ke server broker menggunakan kabel LAN tipe straight yang di sambungkan ke *access point*. Pada *access point* menyambungkan ke server dengan menggunakan koneksi *wireless*. Lalu pada *client* yang akan mengakses *server broker* bisa menggunakan PC/ laptop/ smartphone yang sudah terintegrasi pada satu jaringan ini.

3.3 Skenario Pengujian

Pengujian dalam penelitian ini perlu dilakukan penempatan titik *surveillance*, skema pengujian, *software* dan jaringan yang digunakan, serta hasil pengujian.

3.3.1 Penempatan titik *surveillance*

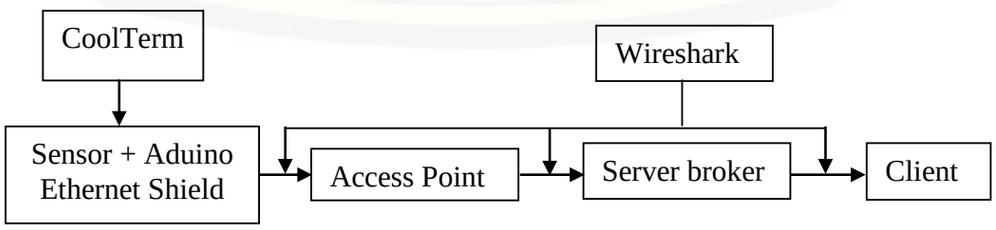


Gambar 3.7 Skenario pengujian *surveillance*

Pada skenario pengujian *surveillance* dilakukan penempatan alat pada titik titik tertentu dimana dapat mengawasi objek yang melewati jarak jangkauan sensor. Tempat pengujian dilakukan pada ruang *server* yang berukuran 2 x 3 m. Untuk alat disini di tempatkan pada 2 titik yang dapat dilihat pada Gambar 3.7. Pada pengujian kali ini menggunakan 2 alat yang di beri nama titik A dan titik B.

3.3.2 Skema Pengujian protokol MQTT

Pengujian pertama yaitu dengan menggunakan protokol MQTT dengan skema seperti berikut:

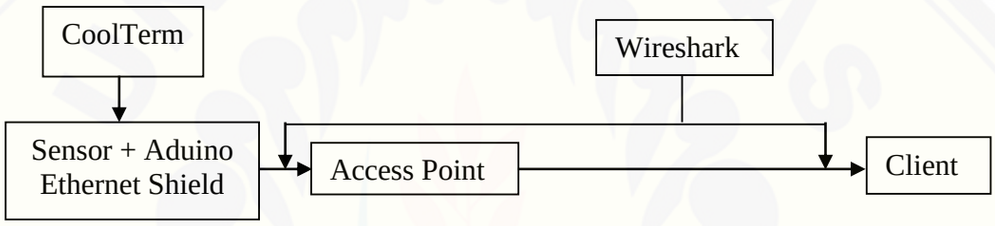


Gambar 3.8 Skema pengujian protokol MQTT

Pada skema pengujian protokol MQTT menggunakan *software* CoolTerm untuk mengambil data waktu dan pengiriman data pada sensor dan Arduino Uno. Setelah itu data dikirim melalui media kabel LAN menuju *access point* lalu ke *server broker* kemudian menuju *client*. Pada saat pengiriman data digunakan *software* Wireshark untuk mengamati paket paket yang lewat melalui jaringan tersebut agar data yang diterima dapat diketahui dan dianalisa.

3.3.3 Skema Pengujian protokol HTTP

Pengujian kedua yaitu dengan menggunakan protokol HTTP dengan skema seperti berikut:



Gambar 3.9 Skema pengujian protokol HTTP

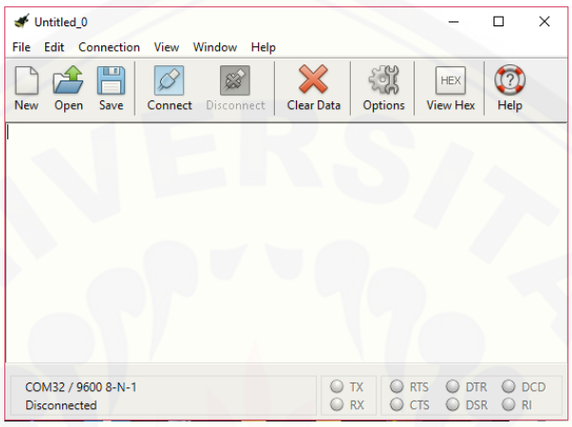
Pada skema pengujian protokol HTTP menggunakan *software* CoolTerm untuk mengambil data waktu dan pengiriman data pada sensor dan Arduino Uno. Setelah itu data dikirim melalui media kabel LAN menuju *access point* kemudian menuju *client*. Pada saat pengiriman data digunakan *software* Wireshark untuk mengamati paket paket yang lewat melalui jaringan tersebut agar data yang diterima dapat diketahui dan dianalisa.

3.3.4 *Software* dan Jaringan

a. CoolTerm

Yaitu aplikasi perangkat lunak yang digunakan untuk melihat apa yang ada dalam komunikasi kabel baik itu kabel *serial* maupun USB. Dalam penelitian ini *software* ini digunakan untuk melihat riwayat pengiriman dari sensor yang dilihat melalui serial monitor pada Arduino Uno. *Software* ini termasuk *open source* dimana tidak memerlukan lisensi berbayar buntuk menggunakannya. Kelebihan

dari *software* ini adalah dapat membaca data melalui USB *Serial* pada Arduino Uno dan menampilkannya seperti halnya serial monitor, tetapi pada *software* ini dapat menyimpan dan menambahkan log tanggal, sehingga Arduino tidak memerlukan perangkat tambahan untuk menyimpan log tanggal. Dibawah ini adalah tampilan dari *software* CoolTerm.

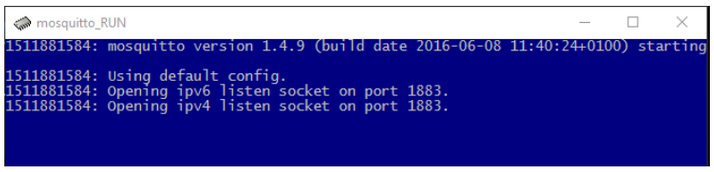


Gambar 3.10 Tampilan pada *software* CoolTerm

Pada pengaplikasian *software* ini dilakukan penyambungan koneksi antara arduino dan PC yang sudah terpasang *software* ini. Penyambungan koneksi dilakukan pada USB yang tersedia pada Arduino Uno. Untuk mencatat hasil dari log pengiriman data sensor yang terdapat pada Arduino dilakukan pengaturan dengan menggunakan tanggal dan kemudian di save sebagai *file* teks.

b. Mosquitto

Mosquitto adalah *broker* pesan yang open source berlisensi EPL (Eclipse Public License) yang menerapkan protokol MQTT versi 3.1 dan 3.1.1. *Broker* ini digunakan untuk penerapan protokol MQTT pada sistem operasi baik Windows maupun Linux. MQTT pada Mosquitto menyediakan metode ringan untuk olah pesan menggunakan metode publish/ subscribe, hal ini membuat *software* ini cocok untuk sensor dengan *microcontroller* seperti Arduino.

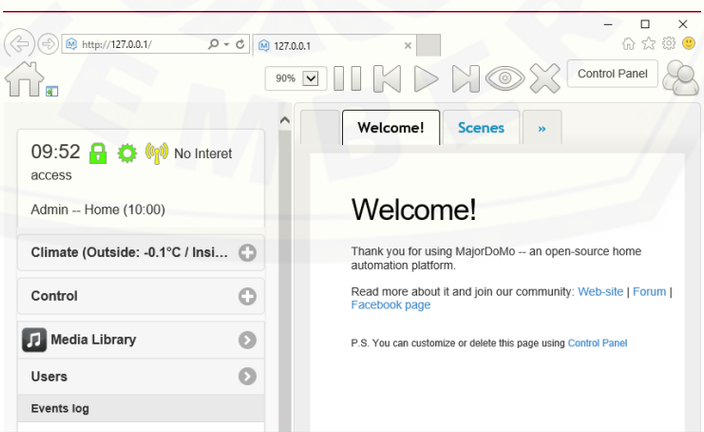


Gambar 3.11 Tampilan awal pada Mosquitto

Server broker pada *software* Mosquitto dijalankan pada sistem operasi Windows 10 64*bit* dengan menggunakan ijin dari Administrator untuk menjalankan *software* tersebut agar dapat berjalan lancar. Pada penelitian kali ini menggunakan *software* Mosquitto versi 1.4.9 dengan *build date* 2016-06-08 11:40:24+0100. Mosquitto ini menggunakan pengaturan dasar atau *default*. Pada saat menjalankan program ini akan membuka koneksi untuk transmisi pada Ipv6 dengan *port* 1883 dan pada Ipv4 dengan *port* 1883.

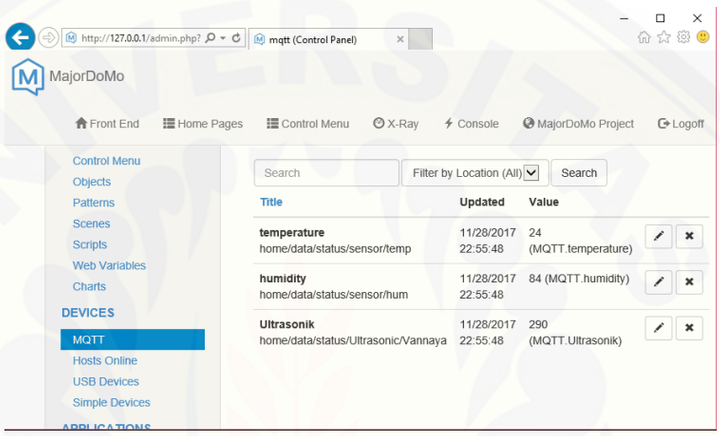
c. MajorDomo

MajorDomo adalah *software open source* yang digunakan untuk proyek *smart home* dengan memanfaatkan beberapa *software* dan dijadikan sebagai satu kesatuan utuh untuk menjalankan tugasnya. MajorDomo menggabungkan tiga aplikasi kedalam satu paket, yaitu Apache, MySQL,dan PHPMyAdmin dan diberi batasan dalam pengaturan di dalamnya karena dikhususkan untuk proyek smarthome. Pada penelitian kali ini saya menggunakan *software* Majordomo untuk menampilkan data pada client dan menghubungkan pada *server broker* Mosquitto.



Gambar 3.12 Tampilan awal MajorDomo

Setelah membuka *software* tersebut dilakukan pengaturan agar data dari *broker* dapat terhubung dengan baik. Berikut adalah tampilan bagian pengaturan pada *software* MajorDomo. Setelah tampilan utama maka dilakukan pengaturan lanjutan dengan menuju link http://*192.168.1.70/admin.php*. Setelah itu menambahkan objek pada menu *object* dengan MQTT yang akan dibuat. Kemudian pada menu *devices* ditambakan pengaturan MQTT untuk pengalamatan dimana data akan ditempatkan pada direktori *server broker*.



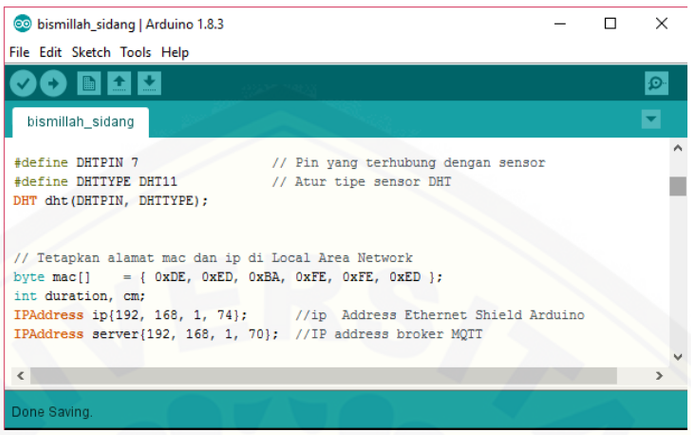
Gambar 3.13 Tampilan administrator MajorDomo

Pada Gambar 3.13 dapat dilihat pada menu *devices* terdapat sub menu MQTT yang di dalamnya terdapat pengaturan lanjutan untuk mengatur letak direktori yang akan digunakan untuk menyimpan data yang akan dikirim pada *server broker*. Untuk data suhu diletakkan pada direktori home\data\status\sensor\temp. Sedangkan untuk data sensor kelembaban diletakkan pada direktori home\data\status\sensor\hum. Untuk sensor ultrasonik sendiri akan diletakkan pada direktori home\data\status\sensor\Vannaya.

d. Jaringan

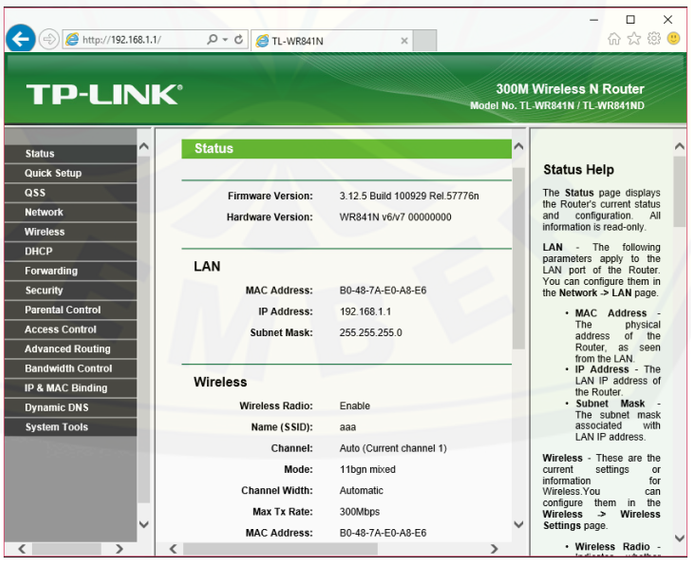
Jaringan pada penelitian ini menggunakan jaringan kabel LAN untuk koneksi dari sensor ke *Access Point* dan *wireless* untuk koneksi dari *Access Point* ke *server broker*. Untuk akses pada *client* digunakan media apa saja dengan catatan sudah terintegrasi dengan jaringan pada *server broker*. Disini ada beberapa

konfigurasi jaringan yaitu konfigurasi jaringan pada Arduino Ethernet Shield, *Access Point*, *Server Broker*, dan *Client*.



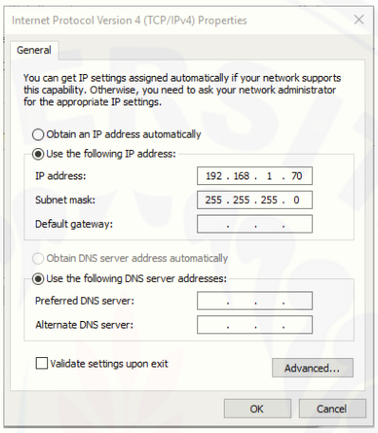
Gambar 3.14 Konfigurasi jaringan pada Arduino Ethernet Shield

Pada Gambar 3.14 dapat dilihat bahwa pengaturan jaringan pada Arduino Ethernet Shield pada program Arduino IDE diatur dengan IP statis. Untuk MAC *Address* menggunakan alamat {0xDE, 0xED, 0xBA, 0xFE, 0xED}. Untuk IP Address pada Arduino Ethernet Shield menggunakan alamat 192.168.1.74. Sedangkan pada *server broker* menggunakan alamat IP 192.168.1.70.



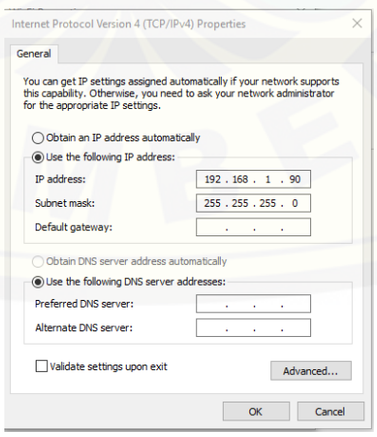
Gambar 3.15 Konfigurasi pada *Access Point*

Pada konfigurasi *access point* disini menggunakan TP-LINK Model No. TL-WR841N/ TL-WR841ND dengan versi *firmware* 3.12.5 Build 100929 Rel.57776n dan versi *hardware* WR841N v6/v7 00000000. MAC *Address pada access point* ini adalah B0-48-7A-E0-A8-E6 dengan alamat IP sebagai berikut 192.168.1.1/255.255.255.0. Nama SSID menggunakan nama ‘aaa’ dengan mode 11bgn *mixed* dan *channel automatic*.



Gambar 3.16 Konfigurasi IP pada *server broker*

Pada konfigurasi *server broker* diatur pada *setting* Windows dengan pengaturan IP *Address* 192.168.1.70. *Subnet mask* menggunakan 255.255.255.0. Untuk *gateway*, *preffered dns* dan *alternate dns* tidak di *setting* atau di kosongkan. Pengaturan ini menggunakan IP versi 4.

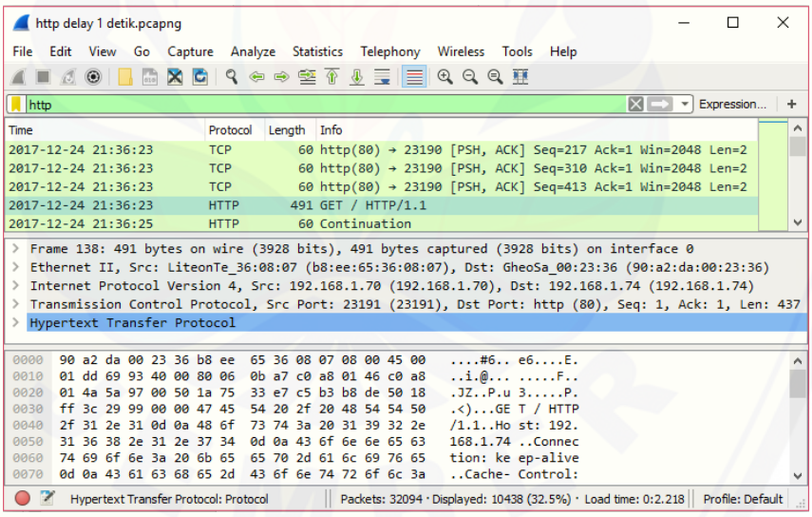


Gambar 3.17 Konfigurasi IP *Client*

Pada konfigurasi IP *client* diatur pada *setting* Windows dengan pengaturan IP *Address* 192.168.1.90. *Subnet mask* menggunakan 255.255.255.0. Untuk *gateway*, *preffered dns* dan *alternate dns* tidak di *setting* atau di kosongkan. Pengaturan ini menggunakan IP versi 4. Atau dapat diatur Obtain karena pada *access point* sudah dapat IP dari DHCP.

e. Wireshark

Pada penelitian ini menggunakan Wireshark sebagai aplikasi perangkat lunak *(software)* untuk dapat melihat dan mencoba menangkap paket-paket jaringan dan berusaha untuk menampilkan semua informasi di paket tersebut sedetail mungkin. *Open Source* dari Wireshark menggunakan *Graphical User Interface* (GUI). Wireshark disebut juga *Network packet analyzer* yang berfungsi menangkap paket-paket jaringan dan berusaha untuk menampilkan semua informasi dipaket tersebut sedetail mungkin.



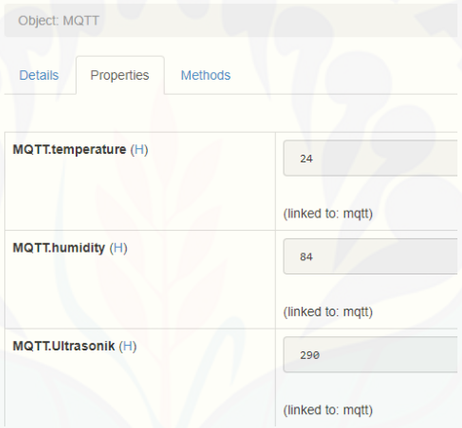
Gambar 3.18 Tampilan *software* Wireshark ketika menangkap data

Pada penelitian kali ini menggunakan *software* Wireshark untuk *monitoring* jaringan agar data dapat tercapture oleh *software* ini. Selain *monitoring* juga dapat untuk *filter* paket apa saja yang ingin di lihat. Pada penelitian ini menggunakan *filter* protokol MQTT dan protokol HTTP. Untuk skenario pengambilan data pada *software* ini menggunakan I/O *graph* untuk menampilkan data dengan format

grafik, kemudian data di analisa dengan perhitungan *packet loss* dan perhitungan *delay*. Pada data yang ter*capture* dapat dilihat waktu kapan data tersebut diterima oleh c*lient/* destinasi.

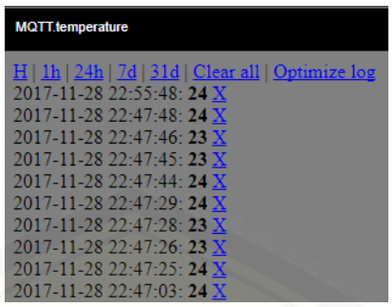
3.3.5 Hasil pengujian

Pada penelitian kali ini yang dilakukan adalah *monitoring* suhu dan kelembaban ruang *server* serta objek di depan *server*. Untuk hasil dari penelitian ini adalah data yang dapat di tampilkan pada *client*. Hasil penelitian dikatakan sukses apabila sudah muncul data dari semua sensor pada tampilan *client*. Dibawah ini adalah contoh hasil yang didapat setelah sukses.



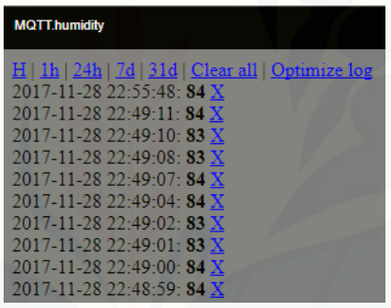
Gambar 3.19 Tampilan hasil pada *Client*

Pada tampilan *client* yang di tampilkan adalah pada menu *object* pada MQTT ditampilkan 3 sub yaitu MQTT.temperature yang merepresentasikan objek sensor suhu, MQTT.humidity yang akan merepresentasikan objek sensor kelembaban, dan MQTT.Ultrasonik yang merepresentasikan objek sensor jarak. Untuk setiap bagian bagian dari hasi tersebut dapat di tampilkan masing masing dalam bentuk angka untuk setiap snya. Berikut adalah gambar pada masing – masing bagian dari hasil tersebut.



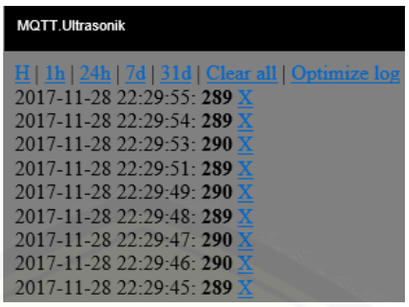
Gambar 3.20 Hasil dari MQTT suhu

Pada tampilan MQTT suhu terdapat informasi berupa tanggal dengan format YYYY-MM-DD, dan jam dengan format HH:MM:SS, setelah itu di ikuti dengan nilai pada sensor suhu dengan satuan derajat *celsius*. Tanda X pada samping data suhu berfungsi untuk menghapus log data tersebut.



Gambar 3.21 Hasil dari MQTT kelembaban

Pada tampilan MQTT kelembaban terdapat informasi berupa tanggal dengan format YYYY-MM-DD, dan jam dengan format HH:MM:SS, setelah itu di ikuti dengan nilai pada sensor kelembaban dengan satuan persentase RH. Tanda X pada samping data kelembaban berfungsi untuk menghapus log data tersebut. *Clear* data pada menu digunakan untuk menghapus semua data. *Optimize* log digunakan agar log tampil dengan optimal.



Gambar 3.22 Hasil dari MQTT ultrasonik

Pada tampilan MQTT ultrasonik terdapat informasi berupa tanggal dengan format YYYY-MM-DD, dan jam dengan format HH:MM:SS, setelah itu di ikuti dengan nilai pada sensor ultrasonik dengan satuan sentimeter. Tanda X pada samping data ultrasonik berfungsi untuk menghapus log data tersebut. *Clear* data pada menu digunakan untuk menghapus semua data. *Optimize* log digunakan agar log tampil dengan optimal.

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitan yang telah dilakukan dapat diambil beberapa kesimpulan diantaranya:

1. Pada pengujian MQTT dimana paket data yang dikirim dari sensor berukuran 3 *byte* (sensor suhu) 3 *byte* (sensor kelembaban) 4 *byte* (sensor ultrasonik) untuk sekali pengiriman.
2. Dalam pengujian QoS *level* 0 saat diberi *noise* 1 s untuk parameter suhu dan kelembaban pada 10 data pertama didapat 3 data yang berhasil diterima oleh *server broker*, dan pada data lainnya gagal di tampilkan oleh *client* yang berarti data tersebut tidak sampai pada *broker*. *Packet loss* yang terjadi sebesar 70% hal ini dikarenakan data 1, 5, 6, 7, 8, 9, 10 tidak diterima oleh *server broker*. (Tabel 4.3 halaman 48 untuk sensor suhu dan Tabel 4.4 halaman 49 untuk sensor kelembaban)
3. Semakin tinggi *level* QoS maka semakin bagus nilai *packet loss* nya. Maka dapat disimpulkan bahwa nilai *packet loss* berbanding terbalik dengan *level* QoS yang digunakan. Untuk kualitas dari nilai *packet loss* berbanding lurus dengan *level* QoS yang digunakan. (Gambar 4.19 halaman 72 dan Gambar 4.20 halaman 73)
4. Pada perbandingan *delay* protokol MQTT dengan *noise* 13 s lebih unggul dibandingkan dengan protokol HTTP. (Gambar 4.22 halaman 75)

5.2 Saran

Untuk penelitian selanjutnya, penulis menyarankan beberapa hal agar penelitian ini dapat dikembangkan dan lebih bermanfaat. Adapun beberapa saran dari penulis adalah sebagai berikut:

1. Pada penelitian selanjutnya diharapkan dapat mengembangkan sistem yang lebih besar (*online*).
2. Penulis mengharapkan adanya perbandingan dengan protokol lainnya agar dapat dikembangkan kedepannya.

DAFTAR PUSTAKA

Abdul Zabar, A. & Novianto, F., 2015. *KEAMANAN HTTP DAN HTTPS BERBASIS WEB MENGGUNAKAN SISTEM OPERASI KALI LINUX*. Jurnal Ilmiah Komputer dan Informatika (KOMPUTA) , 4(2), pp. 69-74.

Abdur R, H., Primananda, R. & Nurwasito, H., 2017. *Sistem Kendali Berbasis Mikrokontroler Menggunakan Protokol MQTT pada Smarthome.* Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer , 1(06), pp. 445-455.

Anonim, 2015. *Arduino Ethernet Shield.* [Online] Available at: http://www.arduino.cc/en/Main/arduinoBoardEthernetShield [Diakses 11 Januari 2017].

Anonim, 2015. *Arduino Uno*. [Online] Available at: http://www.arduino.cc/en/Main/arduinoBoardUno [Diakses 11 Januari 2017].

ASHRAE, 2017. *Telecommunications Infrastructure Standard for Data Centers.* Quebec City, QC, CAN , Paten No. ANSI/TIA-942-A.

Bagas, Q., 2016. *Lightweight Push Messaging dengan MQTT*. [Online] Available at: https://developers.kudo.co.id/2016/07/18/lightweight-push-messaging-dengan-mqtt/

Budianto, H. & Windardi, S., 2012. *RANCANG BANGUN DAN WEB MONITORING PENGUKUR TEMPERATUR SUHU UNTUK PERINGATAN PADA RUANG SERVER MENGGUNAKAN SENSOR DHT 11 DENGAN MODUL KOMUNIKASI ARDUINO UNO*. pp. 1-10.

Budioko, T., 2016. *SISTEM MONITORING SUHU JARAK JAUH BERBASIS INTERNET OF THINGS MENGGUNAKAN PROTOKOL MQTT.* Seminar Riset Teknologi Informasi (SRITI), pp. 353-358.

Cahyawan, A. K. A., 2011. *Sistem Monitor Dan Kendali Ruang Server Dengan Embedded Ethernet.* LONTAR KOMPUTER VOL. 2 JUNI, pp. 64-68.

Candra, R., 2006. *ALAT PEMANTAU SUHU RUANGAN MELALUI WEB BERBASISKAN MIKROKONTROLER AT89S51*. Proceeding, Seminar Ilmiah Nasional Komputer dan Sistem Intelijen (KOMMIT 2006), pp. 533-538.

Grgić, K., Špeh, I. & Heđi, I., 2016. *A Web-Based IoT Solution for Monitoring Data Using MQTT Protocol*. IEEE, pp. 249-253.

Hanifah, S., Rizqika A, S. & Amron, K., 2017. *Implementasi Quality of Service pada Protokol Message Queue Telemetry Transport – Sensor Network (MQTT-SN) Berbasis Arduino dan NRF24L01*. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer , 2(6), pp. 2131-2140.

Kim, J. et al., 2016. *Standard-Based Iot Platforms Interworking : Implementation, Experiences, and Lessons Learned.* IEEE Communications Magazine - Communications Standards Supplement, July, pp. 48-54.

Nur Ilmiyati, R., t.thn. *SISTEM MONITORING DAN KONTROL OTOMATIS INKUBATOR BAYI DENGAN VISUAL BASIC 6.0 BERBASIS ARDUINO.*

Oktafia F, S., Imantia S, H. & Hutagalung, M., t.thn. *Pengukuran Kinerja Sistem Publish/ Subscribe Menggunakan Protokol MQTT (Message Queuing Telemetry Transport).* Jurnal Telematika, 9(1), pp. 26-30.

Popovic, G. et al., 2013. *Overview, Characteristics and Advantages of IP Camera Video Surveillance Systems Compared to Systems with other Kinds of Camera.* International Journal of Engineering Science and Innovative Technology (IJESIT), 2(5), pp. 356-362.

Prada, M. A. et al., 2016. *Communication with resource-constrained devices through MQTT for control education.* IFAC-PapersOnLine 49-6, pp. 150-155.

Priatmoko, D. B., Astuti, E. S. & Riyadi, 2016. *ANALISIS PENERAPAN SISTEM KEAMANAN FISIK PADA DATA CENTER UNTUK MELINDUNGI DATA ORGANISASI.* Jurnal Administrasi Bisnis (JAB), 40(1), pp. 160-169.

Santoso, 2015. *Sensor DHT11*. [Online] Available at: http://www.elangsakti.com/2015/05/sensor-dht11.html [Diakses 20 januari 2017].

Santoso, 2015. *Sensor Ultrasonik.* [Online] Available at: http://www.elangsakti.com/2015/05/sensor-ultrasonik.html [Diakses 20 januari 2017].

Satria, G. O., Satrya, G. B. & Herutomo, A., 2015. *IMPLEMENTASI PROTOKOL MQTT PADA SMART BUILDING BERBASIS OPENMTC*. Universitas Telkom.

Sihombing, R. O. L. & Zulfin, M., 2013. *ANALISIS KINERJA TRAFIK WEB BROWSER DENGAN WIRESHARK NETWORK PROTOCOL ANALYZER PADA SISTEM CLIENT-SERVER.* SINGUDA ENSIKOM, 2(3), pp. 96-101.

Suryansyah, Saleh, M. & Pontia, F. T., 2014. *RANCANG BANGUNSISTEM MONITORING KEAMANAN LINGKUNGAN BERBASIS WIFI MENGGUNAKAN IP CAMERA.*

Sutarman, 2009. *Pengantar Teknologi Informasi*. Jakarta: PT Bumi Aksara.

Weku, H. S., Poekoel, V. C. & Robot, R. F., 2015. *Rancang Bangun Alat Pemberi Pakan Ikan Otomatis Berbasis Mikrokontroler.* E-journal Teknik Elektro dan Komputer, 5(7), pp. 54-64.

Yokotani, T. & Sasaki, Y., 2016. *Comparison with HTTP and MQTT on Required Network Resources for IoT*. International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC), pp. 1-6.

LAMPIRAN

Program pada arduino :

Program protokol MQTT

```
/*Sketsa untuk menampilkan informasi dari sensor ke Port Monitor (DHT11,
HC-SR04)
   dan mengirimkan data suhu, kelembaban dan data jarak dalam sentimeter
   ke broker atau MajorDoMo oleh MQTT.*/
#include <SPI.h>                // Library SPI
#include <Ethernet.h>           // Library Ethernet
#include <PubSubClient.h>       // Library MQTT
#include <DHT.h>                // Library untuk sensor DHT11
#include <Kalman.h>
#define DHTPIN 7                // Pin yang terhubung dengan sensor
#define DHTTYPE DHT11           // Atur tipe sensor DHT
DHT dht(DHTPIN, DHTTYPE);
// Tetapkan alamat mac dan ip di Local Area Network
byte mac[]    = { 0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED };
int duration, cm;
IPAddress ip{192, 168, 1, 74};      //ip  Address Ethernet Shield
Arduino
IPAddress server{192, 168, 1, 70};  //IP address broker MQTT
// Fungsi (Callback)
void callback(char* topic, byte* payload, unsigned int length);
EthernetClient ethClient;                                  //Inisialisasi
Ethernet client
PubSubClient client(server, 1883, callback, ethClient);   //Inisialisasi
MQTT client
int echoPin = 6;       // Koneksi pin 6 - Echo (Range finder HC-SR04)
// Fungsi Callback
void callback(char* topic, byte* payload, unsigned int length)
{
  // Alokasi memori yang diperlukan untuk salinan payload
  byte* p = (byte*)malloc(length);
  // Salin payload ke buffer baru
  memcpy(p, payload, length);
  client.publish("home/data/status/sensor", p, length);
  // Membebaskan memori
  free(p);
}
void setup()
{
  // 1 baud sama dengan 0,8 bps
  // 1 bit / detik sama dengan 1,25 baud
  Serial.begin(9600);             // Atur kecepatan baudrate
  Serial.println("DHT11 test!");  // Pesan uji saat buka Monitor Port
  dht.begin();
  Ethernet.begin(mac, ip);        // Menginisialisasi mac, ip
}
void loop()
{
```

```
  // Pengolahan data dari range finder HC-SR04
   pinMode(echoPin, OUTPUT);        // pin output
  digitalWrite(echoPin,LOW);
  delayMicroseconds(2);
  digitalWrite(echoPin,HIGH);
  delayMicroseconds(5);
   digitalWrite(echoPin,LOW);
   pinMode(echoPin,INPUT);
  duration = pulseIn(echoPin, HIGH);
    //Tetapkan variabel untuk durasi dan Lihat.
    cm = duration / 58;                  // Menghitung dalam sentimeter
  Serial.print(cm);                    // Nilai akhir dalam Ref.
  Serial.println(" cm");
  static char char_Zvyk[10];    // Variabel untuk terjemahan (HC-SR04)
  dtostrf(cm, 4, 0, char_Zvyk); // Terjemahan dari int ke char (HC-SR04)
  // Parameter sensor DHT11
  int h = dht.readHumidity();     // Variabel tipe int untuk Kelembaban
  int t = dht.readTemperature();  // Variabel tipe int untuk Temperatur
    T = (0,9206*t) + 2,9627;
    H = (1,0482*h) - 1,3564;
  // Mengkonversi variabel untuk dikirim ke MQTT to Broker
  static char char_temp[10];      // Variabel untuk terjemahan dari int
ke char
  dtostrf(t, 3, 0, char_temp);    // Terjemahan dari int ke char
  static char char_hum[10];
  dtostrf(h, 3, 0, char_hum);
  if (isnan(t) || isnan(h))     // Periksa apakah membaca dengan DHT11
berhasil
  {
    Serial.println("Failed to read from DHT11");  // Tidak bisa membaca
DHT11
  }
  else
  {
    Serial.print("Humidity: ");
    Serial.print(h);
    Serial.print(" %\t");
    Serial.print("Temperature: ");
    Serial.print(t);
    Serial.println(" *C");
  }
  //Mengirimkan data pada MQTT ke Broker atau MajorDoMo ...
  if (client.connect("DHT_UltrasonicClient"))
  {
    //Mengirim data dari sensor DHT11
    client.publish("home/data/status/sensor/temp", char_temp); //kirim
ke nilai Broker
    client.publish("home/data/status/sensor/hum", char_hum);
    //Mengirim data dari Rangefinder Ultrasonic HC-SR04
    client.publish("home/data/status/Ultrasonic/Vannaya", char_Zvyk);
    delay(13000);      // Keterlambatan dalam hitungan detik
    client.disconnect();      // Disonnect
  }
}
```

Program protocol HTTP

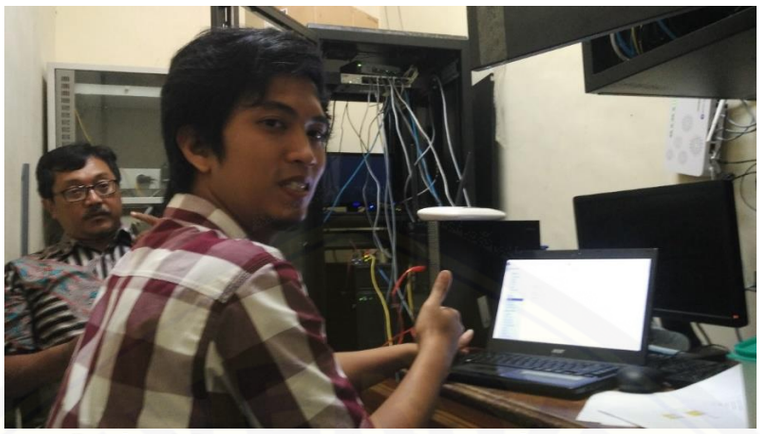
```
#include <DHT.h> // memanggil library sensor DHT
#include <SPI.h> // memanggil library spi
#include <Ethernet.h> // memanggil library Ethernet shield
#define DHTPIN 7     // PIN data yang masuk dari DHT 11 menuju board
arduino
#define DHTTYPE DHT11   // DHT 11
DHT dht(DHTPIN, DHTTYPE);
byte mac[] = { 0x90, 0xA2, 0xDA, 0x00, 0x23, 0x36 }; //MAC Address pada
boks Ethernet shield anda.
int duration, cm;
IPAddress ip(192, 168, 1, 74);
IPAddress gateway(192, 168, 1, 70);
IPAddress subnet(255, 255, 255, 0);
EthernetServer server(80); // port default HTTP adalah 80,
inisialisasikan dengan Ethernet server library
int echoPin = 6;
void setup() {
Serial.begin(9600);
dht.begin();
Ethernet.begin(mac, ip);
server.begin();
Serial.print("server is at ");
Serial.println(Ethernet.localIP());
}
void loop() {
  // Pengolahan data dari range finder HC-SR04
   pinMode(echoPin, OUTPUT);        // pin output
  digitalWrite(echoPin,LOW);
  delayMicroseconds(2);
  digitalWrite(echoPin,HIGH);
  delayMicroseconds(5);
  digitalWrite(echoPin,LOW);
  pinMode(echoPin,INPUT);
  duration = pulseIn(echoPin, HIGH);
    //Tetapkan variabel untuk durasi dan Lihat.
    cm = duration / 58;                  // Menghitung dalam sentimeter
  Serial.print(cm);                    // Nilai akhir dalam Ref.
  Serial.println(" cm");
float h = dht.readHumidity();
float t = dht.readTemperature();
if (isnan(t) || isnan(h)) {
Serial.println("Failed to read from DHT");
} else {
Serial.print("Kelembapan: ");
Serial.print(h);
Serial.print(" %\t");
Serial.print("Suhu: ");
Serial.print(t-2);
Serial.println(" *C");
}
delay(1000);
```

```
EthernetClient client = server.available();
if (client) {
Serial.println("new client");
boolean currentLineIsBlank = true;
while (client.connected()) {
if (client.available()) {
char c = client.read();
Serial.write(c);
if (c == '\n' && currentLineIsBlank) {
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println("Connection: close");  // the connection will be closed
after completion of the response
client.println("Refresh: 1");  // refresh halaman browser tiap 10 detik
client.println();
client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.println("<title>");
client.print("Suhu dan Kelembapan");
client.println("</title>");
client.println("<table border=’1’ align=’center’>");
client.println("<tr>");
client.println("<td>");
client.println("<H2>");
client.print("Kelembapan: ");
client.println("</H2>");
client.println("</td>");
client.println("<td>");
client.println("<p />");
client.println("<H1>");
client.print(h); // panggil data kelembapan
client.print(" %\t");
client.println("</H1>");
client.println("<p />");
client.println("</td>");
client.println("</tr>");
client.println("<tr>");
client.println("<td>");
client.println("<H2>");
client.print("Suhu: ");
client.println("</H2>");
client.println("</td>");
client.println("<td>");
client.println("<p />");
client.println("<H1>");
client.print(t-2); // panggil data suhu
client.println(" &#176;");
client.println("C");
client.println("</H1>");
client.println("<p />");
client.println("</td>");
client.println("</tr>");
```

```
client.println("<tr>");
client.println("<td>");
client.println("<H2>");
client.print("Jarak: ");
client.println("</H2>");
client.println("</td>");
client.println("<td>");
client.println("<p />");
client.println("<H1>");
client.print(cm); // panggil data jarak
client.println(" cm");
client.println("</H1>");
client.println("<p />");
client.println("</td>");
client.println("</tr>");
client.println("</table>");
client.println("</html>");
break;
}
if (c == '\n') {
currentLineIsBlank = true;
}
else if (c != '\r') {
currentLineIsBlank = false;
}
}
}
delay(13000);
client.stop();
Serial.println("client disonnected");
}
}
```

Dokumentasi kegiatan

Gambar 1 Proses pengambilan data pada ruang server



Gambar 2 Proses pengujian sistem pada ruang server

Data monitoring suhu ruang server :

07/12/2017	13.30.00	20	07/12/2017	13.29.27	21	07/12/2017	13.28.51	20
07/12/2017	13.30.59	21	07/12/2017	13.29.26	20	07/12/2017	13.28.50	21
07/12/2017	13.29.58	21	07/12/2017	13.29.20	20	07/12/2017	13.28.30	21
07/12/2017	13.29.57	20	07/12/2017	13.29.19	21	07/12/2017	13.28.29	20
07/12/2017	13.29.56	20	07/12/2017	13.29.12	21	07/12/2017	13.28.28	20
07/12/2017	13.29.54	21	07/12/2017	13.29.11	20	07/12/2017	13.28.26	21
07/12/2017	13.29.53	21	07/12/2017	13.29.10	20	07/12/2017	13.25.12	21
07/12/2017	13.29.52	20	07/12/2017	13.29.08	21	07/12/2017	13.25.11	20
07/12/2017	13.29.51	20	07/12/2017	13.29.07	21	07/12/2017	13.25.10	20
07/12/2017	13.29.49	21	07/12/2017	13.29.06	20	07/12/2017	13.25.09	21
07/12/2017	13.29.48	21	07/12/2017	13.29.04	20	07/12/2017	13.21.48	21
07/12/2017	13.29.47	20	07/12/2017	13.29.03	21	07/12/2017	13.21.46	20
07/12/2017	13.29.35	20	07/12/2017	13.29.02	21	07/12/2017	13.21.44	20
07/12/2017	13.29.34	21	07/12/2017	13.29.00	20	07/12/2017	13.21.43	21
07/12/2017	13.29.33	21	07/12/2017	13.28.59	20	07/12/2017	13.19.05	21
07/12/2017	13.29.31	20	07/12/2017	13.28.58	21	07/12/2017	13.19.04	20
07/12/2017	13.29.30	20	07/12/2017	13.28.54	21	07/12/2017	13.16.36	20
07/12/2017	13.29.29	21	07/12/2017	13.28.53	20			

Data monitoring kelembaban ruang server :

07/12/2017	13.30.01	57	07/12/2017	13.29.51	57	07/12/2017	13.29.26	57
07/12/2017	13.30.00	56	07/12/2017	13.29.49	56	07/12/2017	13.29.20	57
07/12/2017	13.29.58	56	07/12/2017	13.29.48	56	07/12/2017	13.29.19	55
07/12/2017	13.29.57	57	07/12/2017	13.29.47	57	07/12/2017	13.29.12	55
07/12/2017	13.29.56	57	07/12/2017	13.29.35	57	07/12/2017	13.29.11	57
07/12/2017	13.29.54	56	07/12/2017	13.29.34	56	07/12/2017	13.29.10	57
07/12/2017	13.29.53	56	07/12/2017	13.29.27	56	07/12/2017	13.29.08	55
07/12/2017	13.29.52	57						

07/12/2017	13.29.07	55	07/12/2017	13.25.43	55	07/12/2017	13.24.39	55
07/12/2017	13.29.06	56	07/12/2017	13.25.42	55	07/12/2017	13.24.36	55
07/12/2017	13.28.59	56	07/12/2017	13.25.41	56	07/12/2017	13.24.34	56
07/12/2017	13.28.58	55	07/12/2017	13.25.39	56	07/12/2017	13.24.31	56
07/12/2017	13.28.57	55	07/12/2017	13.25.38	55	07/12/2017	13.24.29	55
07/12/2017	13.28.55	56	07/12/2017	13.25.37	55	07/12/2017	13.24.03	55
07/12/2017	13.28.51	56	07/12/2017	13.25.36	56	07/12/2017	13.24.02	56
07/12/2017	13.28.50	55	07/12/2017	13.25.27	56	07/12/2017	13.23.42	56
07/12/2017	13.28.30	55	07/12/2017	13.25.26	55	07/12/2017	13.23.40	55
07/12/2017	13.28.29	56	07/12/2017	13.25.25	55	07/12/2017	13.23.31	55
07/12/2017	13.28.28	56	07/12/2017	13.25.23	56	07/12/2017	13.23.30	56
07/12/2017	13.28.26	55	07/12/2017	13.25.18	56	07/12/2017	13.23.29	56
07/12/2017	13.25.52	55	07/12/2017	13.25.16	55	07/12/2017	13.23.27	55
07/12/2017	13.25.51	56	07/12/2017	13.25.12	55	07/12/2017	13.23.10	55
07/12/2017	13.25.50	56	07/12/2017	13.25.11	55	07/12/2017	13.23.09	56
07/12/2017	13.25.48	55	07/12/2017	13.25.10	55	07/12/2017	13.23.06	56
07/12/2017	13.25.47	55	07/12/2017	13.25.09	55	07/12/2017	13.23.04	55
07/12/2017	13.25.46	56	07/12/2017	13.24.44	55	07/12/2017	13.22.54	55
07/12/2017	13.25.44	56	07/12/2017	13.24.43	56	07/12/2017	13.22.53	56
			07/12/2017	13.24.41	56			

Data monitoring jarak ruang server :

07/12/2017	13.30.00	77	07/12/2017	10.44.27	64	07/12/2017	10.43.54	51
07/12/2017	10.44.59	132	07/12/2017	10.44.25	65	07/12/2017	10.43.49	50
07/12/2017	10.44.58	77	07/12/2017	10.44.24	64	07/12/2017	10.43.48	49
07/12/2017	10.44.56	168	07/12/2017	10.44.22	64	07/12/2017	10.43.46	50
07/12/2017	10.44.55	76	07/12/2017	10.44.21	65	07/12/2017	10.43.44	64
07/12/2017	10.44.54	169	07/12/2017	10.44.21	65	07/12/2017	10.43.43	58
07/12/2017	10.44.50	80	07/12/2017	10.44.19	62	07/12/2017	10.43.42	56
07/12/2017	10.44.49	83	07/12/2017	10.44.17	64	07/12/2017	10.43.40	51
07/12/2017	10.44.48	168	07/12/2017	10.44.16	66	07/12/2017	10.43.39	56
07/12/2017	10.44.46	80	07/12/2017	10.44.15	72	07/12/2017	10.43.37	58
07/12/2017	10.44.44	117	07/12/2017	10.44.11	50	07/12/2017	10.43.36	54
07/12/2017	10.44.43	71	07/12/2017	10.44.11	50	07/12/2017	10.43.34	50
07/12/2017	10.44.42	101	07/12/2017	10.44.08	51	07/12/2017	10.43.33	167
07/12/2017	10.44.39	168	07/12/2017	10.44.08	50	07/12/2017	10.43.32	166
07/12/2017	10.44.38	78	07/12/2017	10.44.06	51	07/12/2017	10.43.28	51
07/12/2017	10.44.37	88	07/12/2017	10.44.05	168	07/12/2017	10.43.27	55
07/12/2017	10.44.35	77	07/12/2017	10.44.03	166	07/12/2017	10.43.27	52
07/12/2017	10.44.34	82	07/12/2017	10.44.02	50	07/12/2017	10.43.24	54
07/12/2017	10.44.30	73	07/12/2017	10.44.01	51	07/12/2017	10.43.23	57
07/12/2017	10.44.30	78	07/12/2017	10.43.58	50	07/12/2017	10.43.22	62
07/12/2017	10.44.28	72	07/12/2017	10.43.54	50	07/12/2017	10.43.21	56

07/12/2017	10.43.19	52	07/12/2017	10.43.07	59	07/12/2017	10.42.57	167
07/12/2017	10.43.18	53	07/12/2017	10.43.05	61	07/12/2017	10.42.55	167
07/12/2017	10.43.17	56	07/12/2017	10.43.04	57	07/12/2017	10.42.53	69
07/12/2017	10.43.13	77	07/12/2017	10.43.00	167	07/12/2017	10.42.52	167
07/12/2017	10.43.11	72	07/12/2017	10.42.59	167			
07/12/2017	10.43.08	84	07/12/2017	10.42.58	57			

Data monitoring suhu ruang server delay 1 detik :

07/12/2017	13.30.00	20	07/12/2017	13.29.27	21	07/12/2017	13.28.51	20
07/12/2017	13.30.59	21	07/12/2017	13.29.26	20	07/12/2017	13.28.50	21
07/12/2017	13.29.58	21	07/12/2017	13.29.20	20	07/12/2017	13.28.30	21
07/12/2017	13.29.57	20	07/12/2017	13.29.19	21	07/12/2017	13.28.29	20
07/12/2017	13.29.56	20	07/12/2017	13.29.12	21	07/12/2017	13.28.28	20
07/12/2017	13.29.54	21	07/12/2017	13.29.11	20	07/12/2017	13.28.26	21
07/12/2017	13.29.53	21	07/12/2017	13.29.10	20	07/12/2017	13.25.12	21
07/12/2017	13.29.52	20	07/12/2017	13.29.08	21	07/12/2017	13.25.11	20
07/12/2017	13.29.51	20	07/12/2017	13.29.07	21	07/12/2017	13.25.10	20
07/12/2017	13.29.49	21	07/12/2017	13.29.06	20	07/12/2017	13.25.09	21
07/12/2017	13.29.48	21	07/12/2017	13.29.04	20	07/12/2017	13.21.48	21
07/12/2017	13.29.47	20	07/12/2017	13.29.03	21	07/12/2017	13.21.46	20
07/12/2017	13.29.35	20	07/12/2017	13.29.02	21	07/12/2017	13.21.44	20
07/12/2017	13.29.34	21	07/12/2017	13.29.00	20	07/12/2017	13.21.43	21
07/12/2017	13.29.33	21	07/12/2017	13.28.59	20	07/12/2017	13.19.05	21
07/12/2017	13.29.31	20	07/12/2017	13.28.58	21	07/12/2017	13.19.04	20
07/12/2017	13.29.30	20	07/12/2017	13.28.54	21	07/12/2017	13.16.36	20
07/12/2017	13.29.29	21	07/12/2017	13.28.53	20			

Data monitoring kelembaban ruang server delay 1 detik:

07/12/2017	13.30.01	57	07/12/2017	13.29.27	56	07/12/2017	13.28.51	56
07/12/2017	13.30.00	56	07/12/2017	13.29.26	57	07/12/2017	13.28.50	55
07/12/2017	13.29.58	56	07/12/2017	13.29.20	57	07/12/2017	13.28.30	55
07/12/2017	13.29.57	57	07/12/2017	13.29.19	55	07/12/2017	13.28.29	56
07/12/2017	13.29.56	57	07/12/2017	13.29.12	55	07/12/2017	13.28.28	56
07/12/2017	13.29.54	56	07/12/2017	13.29.11	57	07/12/2017	13.28.26	55
07/12/2017	13.29.53	56	07/12/2017	13.29.10	57	07/12/2017	13.25.52	55
07/12/2017	13.29.52	57	07/12/2017	13.29.08	55	07/12/2017	13.25.51	56
07/12/2017	13.29.51	57	07/12/2017	13.29.07	55	07/12/2017	13.25.50	56
07/12/2017	13.29.49	56	07/12/2017	13.29.06	56	07/12/2017	13.25.48	55
07/12/2017	13.29.48	56	07/12/2017	13.28.59	56	07/12/2017	13.25.47	55
07/12/2017	13.29.47	57	07/12/2017	13.28.58	55	07/12/2017	13.25.46	56
07/12/2017	13.29.35	57	07/12/2017	13.28.57	55	07/12/2017	13.25.44	56
07/12/2017	13.29.34	56	07/12/2017	13.28.55	56	07/12/2017	13.25.43	55
						07/12/2017	13.25.42	55

07/12/2017	13.25.41	56	07/12/2017	13.25.10	55	07/12/2017	13.23.40	55
07/12/2017	13.25.39	56	07/12/2017	13.25.09	55	07/12/2017	13.23.31	55
07/12/2017	13.25.38	55	07/12/2017	13.24.44	55	07/12/2017	13.23.30	56
07/12/2017	13.25.37	55	07/12/2017	13.24.43	56	07/12/2017	13.23.29	56
07/12/2017	13.25.36	56	07/12/2017	13.24.41	56	07/12/2017	13.23.27	55
07/12/2017	13.25.27	56	07/12/2017	13.24.39	55	07/12/2017	13.23.10	55
07/12/2017	13.25.26	55	07/12/2017	13.24.36	55	07/12/2017	13.23.09	56
07/12/2017	13.25.25	55	07/12/2017	13.24.34	56	07/12/2017	13.23.06	56
07/12/2017	13.25.23	56	07/12/2017	13.24.31	56	07/12/2017	13.23.04	55
07/12/2017	13.25.18	56	07/12/2017	13.24.29	55	07/12/2017	13.22.54	55
07/12/2017	13.25.16	55	07/12/2017	13.24.03	55	07/12/2017	13.22.53	56
07/12/2017	13.25.12	55	07/12/2017	13.24.02	56			
07/12/2017	13.25.11	55	07/12/2017	13.23.42	56			

Data monitoring jarak ruang server delay 1 detik:

07/12/2017	13.30.00	77	07/12/2017	10.44.19	62	07/12/2017	10.43.33	167
07/12/2017	10.44.59	132	07/12/2017	10.44.17	64	07/12/2017	10.43.32	166
07/12/2017	10.44.58	77	07/12/2017	10.44.16	66	07/12/2017	10.43.28	51
07/12/2017	10.44.56	168	07/12/2017	10.44.15	72	07/12/2017	10.43.27	55
07/12/2017	10.44.55	76	07/12/2017	10.44.11	50	07/12/2017	10.43.27	52
07/12/2017	10.44.54	169	07/12/2017	10.44.11	50	07/12/2017	10.43.24	54
07/12/2017	10.44.50	80	07/12/2017	10.44.08	51	07/12/2017	10.43.23	57
07/12/2017	10.44.49	83	07/12/2017	10.44.08	50	07/12/2017	10.43.22	62
07/12/2017	10.44.48	168	07/12/2017	10.44.06	51	07/12/2017	10.43.21	56
07/12/2017	10.44.46	80	07/12/2017	10.44.05	168	07/12/2017	10.43.19	52
07/12/2017	10.44.44	117	07/12/2017	10.44.03	166	07/12/2017	10.43.18	53
07/12/2017	10.44.43	71	07/12/2017	10.44.02	50	07/12/2017	10.43.17	56
07/12/2017	10.44.42	101	07/12/2017	10.44.01	51	07/12/2017	10.43.13	77
07/12/2017	10.44.39	168	07/12/2017	10.43.58	50	07/12/2017	10.43.11	72
07/12/2017	10.44.38	78	07/12/2017	10.43.54	50	07/12/2017	10.43.08	84
07/12/2017	10.44.37	88	07/12/2017	10.43.54	51	07/12/2017	10.43.07	59
07/12/2017	10.44.35	77	07/12/2017	10.43.49	50	07/12/2017	10.43.05	61
07/12/2017	10.44.34	82	07/12/2017	10.43.48	49	07/12/2017	10.43.04	57
07/12/2017	10.44.30	73	07/12/2017	10.43.46	50	07/12/2017	10.43.00	167
07/12/2017	10.44.30	78	07/12/2017	10.43.44	64	07/12/2017	10.42.59	167
07/12/2017	10.44.28	72	07/12/2017	10.43.43	58	07/12/2017	10.42.58	57
07/12/2017	10.44.27	64	07/12/2017	10.43.42	56	07/12/2017	10.42.57	167
07/12/2017	10.44.25	65	07/12/2017	10.43.40	51	07/12/2017	10.42.55	167
07/12/2017	10.44.24	64	07/12/2017	10.43.39	56	07/12/2017	10.42.53	69
07/12/2017	10.44.22	64	07/12/2017	10.43.37	58	07/12/2017	10.42.52	167
07/12/2017	10.44.21	65	07/12/2017	10.43.36	54			
07/12/2017	10.44.21	65	07/12/2017	10.43.34	50			

Data monitoring suhu ruang server delay 13 detik:

13/12/2017	17.45.30	21	13/12/2017	17.33.31	23	13/12/2017	17.25.16	22
13/12/2017	17.41.17	21	13/12/2017	17.33.17	23	13/12/2017	17.25.03	23
13/12/2017	17.41.04	22	13/12/2017	17.33.04	24	13/12/2017	17.24.10	23
13/12/2017	17.39.44	22	13/12/2017	17.32.37	24	13/12/2017	17.23.56	22
13/12/2017	17.39.31	23	13/12/2017	17.32.24	23	13/12/2017	17.22.48	22
13/12/2017	17.38.37	23	13/12/2017	17.32.11	23	13/12/2017	17.22.34	23
13/12/2017	17.38.24	24	13/12/2017	17.31.58	24	13/12/2017	17.18.02	23
13/12/2017	17.36.38	24	13/12/2017	17.31.31	24	13/12/2017	17.17.48	22
13/12/2017	17.36.24	23	13/12/2017	17.31.18	23	13/12/2017	17.15.34	22
13/12/2017	17.36.11	23	13/12/2017	17.29.58	23	13/12/2017	17.15.20	21
13/12/2017	17.35.58	24	13/12/2017	17.29.44	22	13/12/2017	17.15.06	23
13/12/2017	17.35.17	24	13/12/2017	17.29.18	22	13/12/2017	17.14.52	21
13/12/2017	17.35.04	23	13/12/2017	17.29.04	23	13/12/2017	17.12.12	21
13/12/2017	17.34.51	23	13/12/2017	17.28.51	22	13/12/2017	17.11.59	20
13/12/2017	17.34.37	24	13/12/2017	17.27.30	22	13/12/2017	17.10.39	20
13/12/2017	17.33.44	24	13/12/2017	17.27.16	23	13/12/2017	16.54.37	21
			13/12/2017	17.27.03	22	13/12/2017	16.54.21	21

Data monitoring kelembaban ruang server delay 13 detik:

13/12/2017	17.45.30	50	13/12/2017	17.33.57	53	13/12/2017	17.27.30	53
13/12/2017	17.43.17	50	13/12/2017	17.33.44	52	13/12/2017	17.27.16	52
13/12/2017	17.43.04	49	13/12/2017	17.33.17	52	13/12/2017	17.27.03	51
13/12/2017	17.40.37	49	13/12/2017	17.33.04	54	13/12/2017	17.26.49	50
13/12/2017	17.40.24	50	13/12/2017	17.32.51	55	13/12/2017	17.26.36	52
13/12/2017	17.39.44	50	13/12/2017	17.32.37	54	13/12/2017	17.25.30	52
13/12/2017	17.39.31	49	13/12/2017	17.32.24	52	13/12/2017	17.25.16	53
13/12/2017	17.39.04	49	13/12/2017	17.32.11	54	13/12/2017	17.25.03	53
13/12/2017	17.38.51	50	13/12/2017	17.31.58	55	13/12/2017	17.24.50	54
13/12/2017	17.38.37	50	13/12/2017	17.31.31	55	13/12/2017	17.24.36	55
13/12/2017	17.38.24	51	13/12/2017	17.31.18	54	13/12/2017	17.24.23	53
13/12/2017	17.37.57	51	13/12/2017	17.30.24	54	13/12/2017	17.24.10	54
13/12/2017	17.37.44	50	13/12/2017	17.30.11	53	13/12/2017	17.23.56	51
13/12/2017	17.35.58	50	13/12/2017	17.29.58	52	13/12/2017	17.23.43	54
13/12/2017	17.35.44	51	13/12/2017	17.29.44	51	13/12/2017	17.23.02	54
13/12/2017	17.35.31	51	13/12/2017	17.29.31	52	13/12/2017	17.22.48	55
13/12/2017	17.35.17	50	13/12/2017	17.29.04	52	13/12/2017	17.22.34	54
13/12/2017	17.34.37	50	13/12/2017	17.28.51	53	13/12/2017	17.22.20	55
13/12/2017	17.34.24	51	13/12/2017	17.28.38	52	13/12/2017	17.22.06	56
13/12/2017	17.34.11	52	13/12/2017	17.27.57	52	13/12/2017	17.21.38	56
			13/12/2017	17.27.43	53	13/12/2017	17.21.24	55

13/12/2017	17.21.10	55
13/12/2017	17.20.56	56
13/12/2017	17.20.15	56
13/12/2017	17.20.02	57
13/12/2017	17.19.35	57
13/12/2017	17.19.22	58
13/12/2017	17.19.09	52
13/12/2017	17.18.55	58
13/12/2017	17.18.42	59
13/12/2017	17.18.29	58
13/12/2017	17.18.16	59
13/12/2017	17.18.02	59
13/12/2017	17.17.48	60
13/12/2017	17.16.55	60
13/12/2017	17.16.42	61
13/12/2017	17.16.15	61
13/12/2017	17.16.01	62
13/12/2017	17.15.34	62
13/12/2017	17.15.20	61
13/12/2017	17.15.06	63
13/12/2017	17.14.52	61
13/12/2017	17.14.38	59
13/12/2017	17.14.25	55
13/12/2017	17.14.12	61
13/12/2017	17.13.45	61
13/12/2017	17.13.32	62
13/12/2017	17.12.12	62
13/12/2017	17.11.59	63
13/12/2017	17.10.39	63

Data monitoring jarak ruang server delay 1 detik:

13/12/2017	17.45.30	75
13/12/2017	17.45.17	96
13/12/2017	17.45.03	101
13/12/2017	17.44.50	129
13/12/2017	17.44.37	96
13/12/2017	17.44.23	95
13/12/2017	17.44.10	97
13/12/2017	17.43.57	96
13/12/2017	17.43.17	96
13/12/2017	17.43.04	98
13/12/2017	17.42.50	96
13/12/2017	17.42.37	98
13/12/2017	17.42.24	109
13/12/2017	17.42.10	96
13/12/2017	17.41.57	97
13/12/2017	17.41.44	107
13/12/2017	17.41.30	129
13/12/2017	17.41.17	138
13/12/2017	17.41.04	139
13/12/2017	17.40.50	96
13/12/2017	17.39.44	96
13/12/2017	17.39.31	95
13/12/2017	17.39.17	97
13/12/2017	17.39.04	94
13/12/2017	17.38.51	100
13/12/2017	17.38.37	170
13/12/2017	17.38.24	249
13/12/2017	17.38.11	96
13/12/2017	17.37.57	145
13/12/2017	17.37.44	96
13/12/2017	17.37.31	145
13/12/2017	17.37.17	185
13/12/2017	17.37.04	126
13/12/2017	17.36.51	94
13/12/2017	17.36.38	94
13/12/2017	17.36.24	137
13/12/2017	17.36.11	95
13/12/2017	17.35.58	93
13/12/2017	17.35.44	90
13/12/2017	17.35.31	98
13/12/2017	17.35.17	106
13/12/2017	17.35.04	137
13/12/2017	17.34.51	141
13/12/2017	17.34.37	96
13/12/2017	17.34.24	113
13/12/2017	17.34.11	142
13/12/2017	17.33.57	101
13/12/2017	17.33.44	92
13/12/2017	17.33.31	252
13/12/2017	17.33.17	249
13/12/2017	17.33.04	117
13/12/2017	17.32.51	95
13/12/2017	17.32.37	169
13/12/2017	17.32.24	120
13/12/2017	17.32.11	95
13/12/2017	17.31.58	170
13/12/2017	17.31.44	102
13/12/2017	17.31.31	100
13/12/2017	17.31.18	357
13/12/2017	17.31.04	94
13/12/2017	17.30.51	98
13/12/2017	17.30.38	45
13/12/2017	17.30.24	90
13/12/2017	17.30.11	95
13/12/2017	17.29.58	98
13/12/2017	17.29.44	42
13/12/2017	17.29.31	116
13/12/2017	17.29.18	95
13/12/2017	17.29.04	96
13/12/2017	17.28.51	94
13/12/2017	17.28.38	114
13/12/2017	17.28.25	95
13/12/2017	17.28.11	116
13/12/2017	17.27.57	124
13/12/2017	17.27.43	95
13/12/2017	17.27.30	91
13/12/2017	17.27.16	196
13/12/2017	17.27.03	95
13/12/2017	17.26.49	98
13/12/2017	17.26.36	94
13/12/2017	17.26.23	101
13/12/2017	17.26.10	89
13/12/2017	17.25.56	95
13/12/2017	17.25.43	145
13/12/2017	17.25.30	115
13/12/2017	17.25.16	101
13/12/2017	17.25.03	133
13/12/2017	17.24.50	96
13/12/2017	17.24.36	95
13/12/2017	17.24.23	71
13/12/2017	17.24.10	135
13/12/2017	17.23.56	174
13/12/2017	17.23.43	96
13/12/2017	17.23.30	6

13/12/2017	17.23.16	0
13/12/2017	17.20.56	0
13/12/2017	17.20.42	95
13/12/2017	17.20.29	95
13/12/2017	17.20.15	90
13/12/2017	17.20.02	136

Data monitoring
HTTP delay 1 detik

21.36.23	491
21.36.26	400
21.36.27	491
21.36.29	400
21.36.30	491
21.36.32	400
21.36.33	491
21.36.36	400
21.36.37	491
21.36.39	400
21.36.40	491
21.36.42	400
21.36.43	491
22.37.37	434
22.37.51	400
22.38.06	491
22.38.20	400
22.38.33	491
22.38.47	400
22.39.00	491
22.39.15	400
22.39.28	491
22.39.42	400
22.39.55	491
21.36.46	400
21.36.47	491
21.36.49	400
21.36.50	491
21.36.52	400
21.36.53	491
21.36.56	400
21.36.57	491
21.36.59	400
21.37.00	491
21.37.02	400
21.37.03	491
21.37.06	400
21.37.07	491
21.37.09	400
21.37.10	491
21.37.12	400
21.37.13	491
21.37.16	400
21.37.17	491
21.37.19	400
21.37.20	491
22.40.10	400
22.40.23	491
22.40.37	400
22.40.50	491
22.41.32	400
22.41.45	491
22.42.00	400
22.42.13	491
22.42.27	400
22.42.40	491
22.42.55	400
21.37.22	400
21.37.23	491
21.37.26	400
21.37.27	491
21.37.29	400
21.37.30	491
21.37.32	400
21.37.33	491
21.37.36	400
21.37.36	491
21.37.39	400
21.37.40	491
21.37.42	400
21.37.43	491
21.37.45	400
21.37.46	491
21.37.49	400
21.37.50	491
22.43.08	491
22.43.22	400
22.43.35	491
22.43.50	400
22.44.03	491
22.44.17	400
22.44.30	491
22.44.45	400
22.44.58	491

Data monitoring
HTTP 13 detik