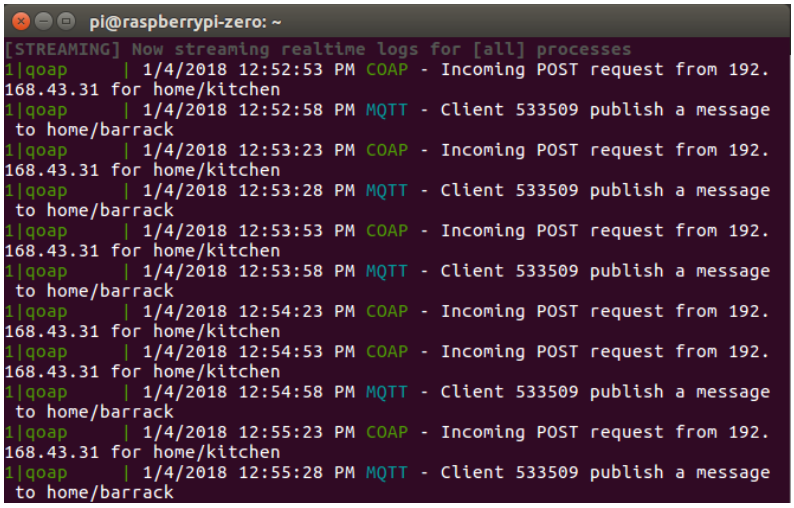
BAB 5 IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dilakukan implementasi sistem dan pengujian berdasarkan perancangan yang dibuat pada bab sebelumnya dan dijelaskan proses pengambilan data kemudian akan dibahas pada bab selanjutnya.

5.1 Pembangunan Testbed

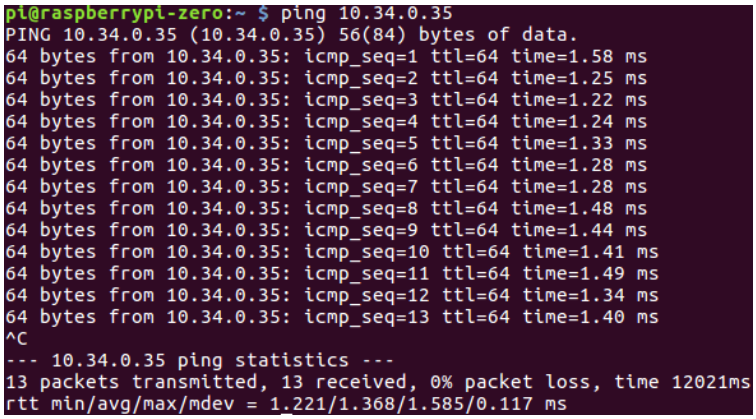
Gambar 5.1 Implementasi Sistem

Pada Gambar 5.1 menunjukkan bahwa *middleware* dapat dibangun pada raspberry pi zero dengan menggunakan tambahan usb port menggunakan USB Hub sehingga *middleware* nantinya dapat memiliki dua *interface* jaringan yaitu wlan0 dengan menggunakan *wireless adapter* sebagai *access point* dan eth0 dengan menggunakan USB to LAN untuk terhubung ke data center sehingga *middleware* dapat mengirimkan data ke aplikasi. Setelah sistem berhasil diimplementasikan dilakukan pengujian testbed untuk mengetahui bahwa rancangan topologi jaringan sudah berjalan sesuai dengan harapan dan dapat saling terhubung satu dengan yang lainnya. Pada Gambar 5.2 menunjukkan bahwa *publisher* CoAP berhasil mengirim data suhu dan kelembapan setiap 30 detik sekali dengan melakukan "POST" data dengan topik home/kitchen ke alamat coap://192.168.43.1/r/home/kitchen melalui port 5683 dan *publisher* MQTT berhasil mengirim data suhu dan kelembapan setiap 30 detik sekali dengan melakukan "*PUBLISH*" data dengan topik home/barrack ke alamat mqtt://192.168.43.1 melalui port 1883.



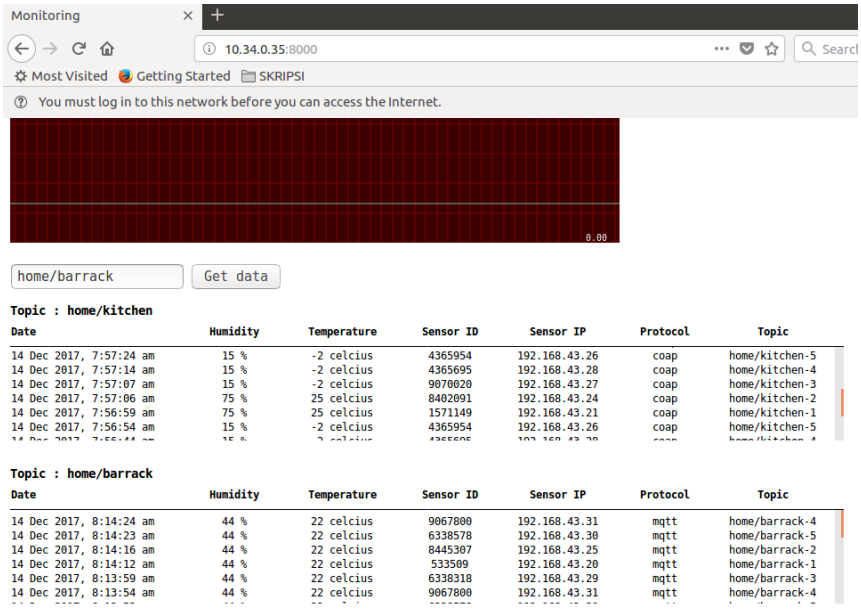
Gambar 5.2 Data *Publisher*

Untuk mengetahui apakah *middleware* telah terhubung dengan data center maka dilakukan tes ping ke alamat IP data center yaitu 10.34.0.35. Pada Gambar 5.3 menunjukkan bahwa *middleware* telah terhubung dengan data center dan dapat melakukan pertukaran data.



Gambar 5.3 Tes Ping ke Datacenter

Setelah *middleware* dapat terhubung dengan datacenter, maka data yang dikirimkan oleh *middleware* dapat dipantau secara *realtime* melalui web-app dengan alamat 10.34.0.35:8000. Pada Gambar 5.4 menunjukkan bahwa data dapat ditampilkan pada web-app. Dengan demikian, pengiriman data dari *publisher* melalu *middleware* hingga ke web-app berjalan sesuai dengan harapan.



Gambar 5.4 Data Web-app

5.2 Metode Pengujian

Metode pengujian yang digunakan dalam penelitian ini sama dengan metode pengujian pada penelitian sebelumnya sehingga data yang diperoleh dapat digunakan sebagai perbandingan.

5.2.1 Pengujian Penggunaan CPU dan Memori

Untuk mendapatkan data kinerja dari CPU dan Memori pada *middleware* digunakan program sebagai berikut:

```
var usage = require('usage');
var pid = 886
setInterval(function() {
            var options = { keepHistory:true};
      usage.lookup(pid, options, function(err, stat) {
            console.log(err, stat);
            console.log(new Date().toISOString());
      });
},15000);
```

Kode Program 5.1 Penggunaan CPU dan Memori Raspberry Pi Zero

Program tersebut akan mencatat log penggunaan CPU dan memori berdasarkan PID (*process ID*) dari *middleware* setiap 15 detik sekali yang dilakukan selama 3 jam. Data dari penggunaan CPU dan memori disimpan dalam log file dengan format .txt.

5.2.2 Pengujian *Delay* dan Data Transfer

Pada pengujian ini digunakan program tcpdump untuk menangkap lalu lintas data pada *interface* wlan0 (Kode Program 5.2) dan eth0 (Kode Program 5.3) yang kemudian dilakukan analisis lebih lanjut dengan menggunakan program wireshark. Nilai *delay* didapatkan dengan cara mengurangi waktu paket saat diterima dengan waktu paket saat dikirimkan. Untuk mendapatkan nilai end-to-end *delay* adalah dengan menjumlahkan *delay* dari NodeMCU ke *middleware* dan *delay* dari *middleware* ke aplikasi. Data transfer adalah berapa banyak data yang berhasil dikirim ke *middleware* oleh *publisher* dan ke aplikasi oleh *middleware*.

```
tcpdump -i wlan0 -w <nama-file>
```

Kode Program 5.2 Tcpdump wlan0

```
tcpdump -i eth0 -w <nama-file>
```

Kode Program 5.3 Tcpdump eth0

5.2.3 Pengujian Skalabilitas

Pada pengujian skalabilitas digunakan package async dari npm yang dapat menjalankan program secara asynchronous. Program pengiriman data (*publish*) dan program meminta data (*subscribe*) dimasukkan kedalam program async seperti pada Kode Program 5.4. Pada pengujian ini ditentukan jumlah *publisher* dan *subscriber* dengan variasi 100, 500, 1000.

```
async.times (1000, function(n, next) {
      Publisher ()
{
       next ();
   };
}, function (err) {
});
```

Kode Program 5.4 Skalabilitas Raspberry Pi Zero

5.3 Skenario Pengujian

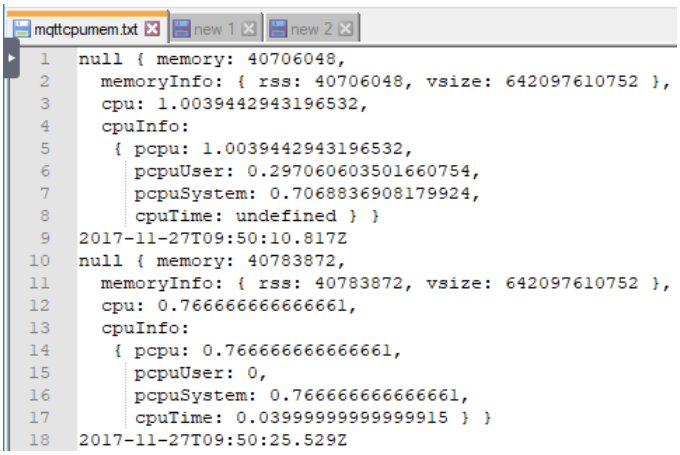
Dalam penelitian ini dilakukan 5 skenario pengujian yaitu:

1. Skenario 1 untuk pengujian penggunaan CPU dan memori adalah dengan menggunakan 1 nodeMCU MQTT yang akan dilakukan selama 3 jam.
2. Skenario 2 untuk pengujian penggunaan CPU dan memori adalah dengan menggunakan 1 nodeMCU CoAP yang akan dilakukan selama 3 jam.

3. Skenario 3 untuk pengujian penggunaan CPU dan memori adalah dengan menggunakan 1 nodeMCU MQTT dan 1 nodeMCU CoAP yang akan dilakukan selama 3 jam.
4. Skenario 4 untuk pengujian *delay* dan data transfer adalah dengan menggunakan 5 nodeMCU MQTT dan 5 nodeMCU CoAP yang mengirim data ke *middleware* secara bersamaan yang akan dilakukan selama 1 jam.
5. Skenario 5 untuk pengujian skalabilitas adalah dengan menjalankan program async dengan variasi *publisher*/susbscriber sebanyak 100, 500, dan 1000.

5.3.1 Pengujian Skenario 1

Pada pengujian skenario 1 ini dilakukan pengujian dengan menggunakan 1 nodeMCU MQTT dengan durasi pengujian selama 3 jam untuk mengetahui bagaimana kinerja penggunaan cpu dan memori dari *middleware* jika mengunakan raspberry pi zero.

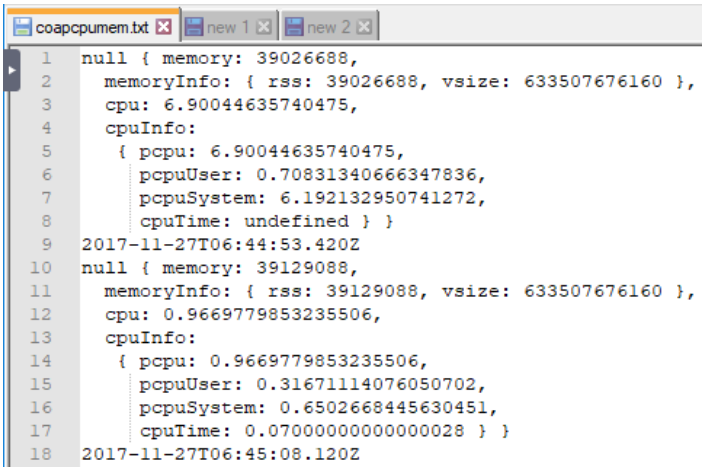


Gambar 5.5 Pengujian Skenario 1

Gambar 5.5 menunjukkan cuplikan hasil log penggunaan cpu dan memori selama pengujian skenario 1 berlangsung. Log ini merupakan output dari program monitoring cpu dan memori pada bab sebelumnya. Tujuan dari monitoring adalah untuk mengetahui seberapa besar beban penggunaan cpu dan memori.

5.3.2 Pengujian Skenario 2

Pada pengujian skenario 2 ini dilakukan pengujian dengan menggunakan 1 nodeMCU CoAP dengan durasi pengujian selama 3 jam untuk mengetahui bagaimana kinerja penggunaan cpu dan memori dari *middleware* jika mengunakan raspberry pi zero.

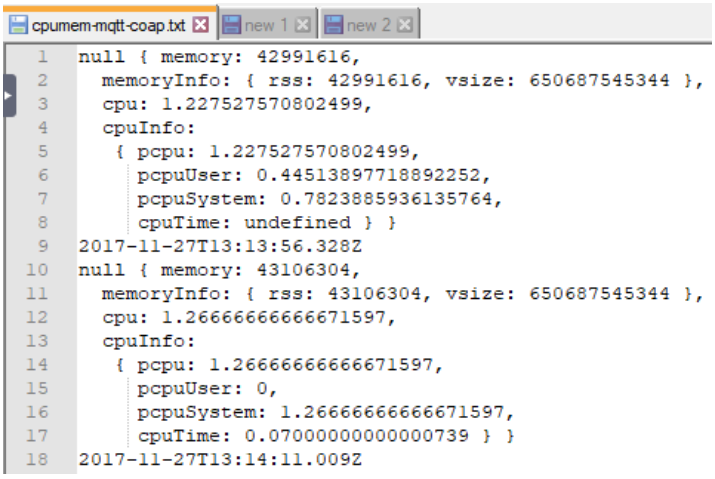


Gambar 5.6 Pengujian Skenario 2

Gambar 5.6 menunjukkan cuplikan hasil log penggunaan cpu dan memori selama pengujian skenario 2 berlangsung. Log ini merupakan output dari program monitoring cpu dan memori pada bab sebelumnya. Tujuan dari monitoring adalah untuk mengetahui seberapa besar beban penggunaan cpu dan memori.

5.3.3 Pengujian Skenario 3

Pada pengujian skenario 3 ini dilakukan pengujian dengan menggunakan 1 nodeMCU MQTT dan 1 nodeMCU CoAP dengan durasi pengujian selama 3 jam untuk mengetahui bagaimana kinerja penggunaan cpu dan memori dari *middleware* jika mengunakan raspberry pi zero.



Gambar 5.7 Pengujian Skenario 3

Gambar 5.7 menunjukkan cuplikan hasil log penggunaan cpu dan memori selama pengujian skenario 3 berlangsung. Log ini merupakan output dari program monitoring cpu dan memori pada bab sebelumnya. Tujuan dari monitoring adalah untuk mengetahui seberapa besar beban penggunaan cpu dan memori.

5.3.4 Pengujian Skenario 4

Pada pengujian skenario keempat ini, digunakan sebanyak 5 nodeMCU MQTT dan 5 nodeMCU CoAP sebagai *publisher* dengan topik yang berbeda-beda. Mekanisme pengujian ini adalah dengan membuat 10 nodeMCU mengirimkan data secara bersamaan selama 1 jam dan dilakukan capture wireshark pada saat pengujian berlangsung. Pada Tabel 5.1 dan Tabel 5.2 dapat dilihat IP untuk masing-masing protokol MQTT dan CoAP.

Tabel 5.1 IP Protokol MQTT

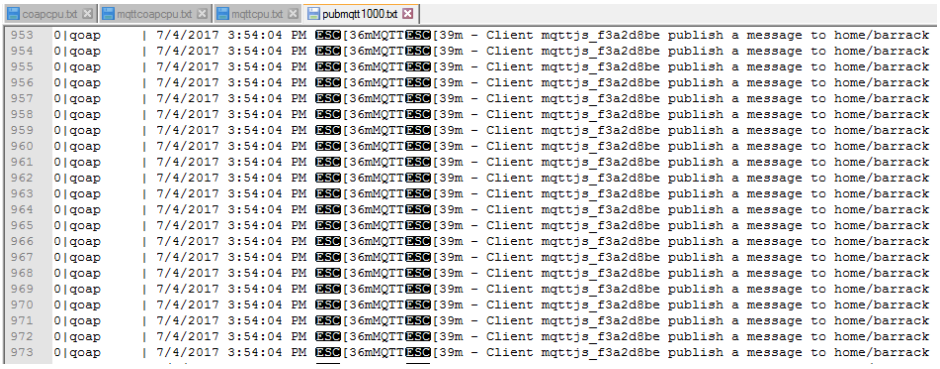
Source	Destination	Protocol	Info
192.168.43.20 (barrack-1)	192.168.43.1	MQTT	*Publish* Message
192.168.43.24 (barrack-2)	192.168.43.1	MQTT	*Publish* Message
192.168.43.26 (barrack-3)	192.168.43.1	MQTT	*Publish* Message
192.168.43.31 (barrack-4)	192.168.43.1	MQTT	*Publish* Message
192.168.43.29 (barrack-5)	192.168.43.1	MQTT	*Publish* Message

Tabel 5.2 IP Protokol CoAP

Source	Destination	Protocol	Info
192.168.43.21 (kitchen-1)	192.168.43.1	CoAP	CON, MID:23346, POST, TKN:6e 6f 64 65, coap://192.168.43.1/r/home/kitchen
192.168.43.25 (kitchen-2)	192.168.43.1	CoAP	CON, MID:24031, POST, TKN:6e 6f 64 65, coap://192.168.43.1/r/home/kitchen
192.168.43.28 (kitchen-3)	192.168.43.1	CoAP	CON, MID:63061, POST, TKN:6e 6f 64 65, coap://192.168.43.1/r/home/kitchen
192.168.43.30 (kitchen-4)	192.168.43.1	CoAP	CON, MID:23327, POST, TKN:6e 6f 64 65, coap://192.168.43.1/r/home/kitchen
192.168.43.31 (kitchen-5)	192.168.43.1	CoAP	CON, MID:63083, POST, TKN:6e 6f 64 65, coap://192.168.43.1/r/home/kitchen

5.3.5 Pengujian Skenario 5

Pengujian skenario 5 adalah pengujian skalabilitas yang dilakukan untuk mengetahui kemampuan *middleware* dalam menangani peningkatan jumlah *publisher* ataupun *subscriber*. Dalam pengujian ini jumlah *publisher* dan *subscriber* yang digunakan adalah 100, 500, 1000. Hasil dari log pengujian skalabilitas seperti pada Gambar 5.8.



Gambar 5.8 Log Skalabilitas

Pengujian skalabilitas dilakukan dengan menggunakan program skalabilitas dengan package async dari npm dan dilakukan perhitungan berapa banyak *publisher* atau *subscriber* yang dapat ditangani oleh *middleware* dalam satu detik.

5.4 Pengambilan Data

5.4.1 Pengambilan Data CPU dan Memori

```
null { memory: 42991616,
  memoryInfo: { rss: 42991616, vsize: 650687545344 },
  cpu: 1.227527570802499,
  cpuInfo:
   { pcpu: 1.227527570802499,
     pcpuUser: 0.44513897718892252,
     pcpuSystem: 0.7823885936135764,
     cpuTime: undefined } }
```

Gambar 5.9 Log CPU dan Memori

Proses pengambilan data log cpu dan memori dicatat setiap 15 detik sekali berdasarkan program yang telah dibuat. Pada Gambar 5.9 data yang didapat adalah data dari pengggunaan memori dalam satuan *byte* yang nantinya dibagi dengan jumlah memori pada raspberry pi zero yaitu 455200768 *byte* dan data penggunaan cpu dalam satuan persentase (%). Pencatatan log dilakukan selama 3 jam untuk skenario 1, skenario 2, dan skenario 3. Setelah pencatan log selesai dilakukan pengumpulan data dan dihutung rata-rata penggunaan cpu dan memori untuk masing-masing skenario.

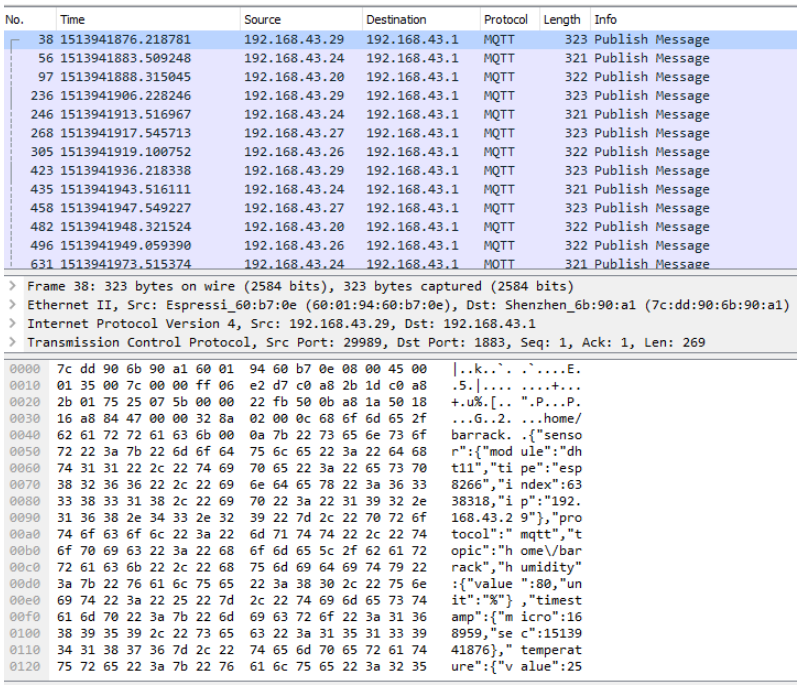
```
pi@raspberrypi-zero:~ $ free -b
             total       used       free     shared    buffers     cached
Mem:     455200768   87035904  368164864    4653056   10100736   40783872
-/+ buffers/cache:   36151296  419049472
Swap:    104853504          0  104853504
```

Gambar 5.10 Jumlah Memori (byte) Raspberry Pi Zero

5.4.2 Pengambilan Data *Delay* dan Data Transfer

Untuk mendapatkan nilai dari *delay* pada penelitian ini adalah dengan mengurangi waktu pada saat paket diterima dengan waktu pada saat paket dikirimkan. Waktu paket diterima dapat dilihat pada wireshark seperti pada

Gambar 5.11 pada bagian kolom *time*. Sedangkan untuk waktu pengiriman dapat dilihat pada info masing-masing paket yang terdapat keterangan *time*stamp. Sebagai contoh pada Gambar 5.11 waktu pengiriman adalah 1513941876.168959 (format epoch *time*) dan waktu paket diterima adalah 1513941876.218781. Jadi *delay* yang didapat adalah 1513941876.218781-1513941876.168959 = 0.049822 detik. Dan akan dilakukan perhitungan *delay* untuk setiap paket yang dikirimkan oleh nodeMCU yang kemudian akan dihitung nilai rata-rata dari *delay*. Cara yang sama juga dilakukan untuk menghitung *delay* dari *middleware* ke aplikasi. Sehingga untuk menghitung nilai *end-to-end delay* adalah dengan menjumlahkan *delay* dari nodeMCU ke *middleware* dengan *delay* dari *middleware* ke aplikasi.



Gambar 5.11 Log *Delay*

Data transfer yang dimaksud dalam penelitian ini adalah seberapa banyak data yang sukses dikirimkan oleh sensor ke *middleware*. Untuk mendapatkan hasil data transfer adalah dengan melakukan filter IP topik tertentu pada log wireshark dan dilakukan perhitungan seberapa banyak paket yang berhasil di*publish* seperti pada Gambar 5.12.

mqtt&&ip.src==192.168.43.20

No.	Source	Destination	Protocol	Length	Time delta	Info
2458	192.168.43.20	192.168.43.1	MQTT	324	0.000000000	Publish Message
7436	192.168.43.20	192.168.43.1	MQTT	324	30.006808000	Publish Message
13282	192.168.43.20	192.168.43.1	MQTT	76	35.238990000	Connect Command
18134	192.168.43.20	192.168.43.1	MQTT	324	30.220287000	Publish Message
23053	192.168.43.20	192.168.43.1	MQTT	324	30.005449000	Publish Message
28481	192.168.43.20	192.168.43.1	MQTT	76	32.555175000	Connect Command
33489	192.168.43.20	192.168.43.1	MQTT	324	30.369245000	Publish Message
38437	192.168.43.20	192.168.43.1	MQTT	324	30.000537000	Publish Message
43278	192.168.43.20	192.168.43.1	MQTT	324	29.998377000	Publish Message
48023	192.168.43.20	192.168.43.1	MQTT	324	29.999324000	Publish Message
52959	192.168.43.20	192.168.43.1	MQTT	324	30.000621000	Publish Message
57824	192.168.43.20	192.168.43.1	MQTT	324	30.001615000	Publish Message
62869	192.168.43.20	192.168.43.1	MQTT	324	30.004071000	Publish Message
67827	192.168.43.20	192.168.43.1	MQTT	324	30.003893000	Publish Message
72655	192.168.43.20	192.168.43.1	MQTT	324	29.999534000	Publish Message
77443	192.168.43.20	192.168.43.1	MQTT	324	30.000934000	Publish Message
82369	192.168.43.20	192.168.43.1	MQTT	324	30.102074000	Publish Message
87263	192.168.43.20	192.168.43.1	MQTT	324	29.888518000	Publish Message
92136	192.168.43.20	192.168.43.1	MQTT	324	30.002207000	Publish Message

Gambar 5.12 Log Data Transfer

5.4.3 Pengambilan Data Skalabilitas

Data skalabilitas diperoleh dari log file dalam bentuk .txt yang merupakan output dari program skalabillitas. Pada log file tersebut dicatat waktu kapan proses *publish*, *post*, atau *subscribe* dilakukan. Waktu inilah yang digunakan untuk menghitung seberapa banyak *concurrent publish/subscribe* dalam satu detik. Perhitungan dilakukan untuk seluruh variasi jumlah *publish/subscribe* yakni 100, 500, 1000. Contoh log file skalabilitas dengan *publisher* coap berjumlah 100 seperti pada Gambar 5.13.

coap100.txt

```
71  [STREAMING] Now streaming realtime logs for [all] processes
72  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
73  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
74  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
75  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
76  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
77  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
78  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
79  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
80  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
81  l|qoap     | 12/19/2017 6:14:27 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
82  l|qoap     | 12/19/2017 6:14:32 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
83  l|qoap     | 12/19/2017 6:14:32 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
84  l|qoap     | 12/19/2017 6:14:32 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
85  l|qoap     | 12/19/2017 6:14:32 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
86  l|qoap     | 12/19/2017 6:14:32 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
87  l|qoap     | 12/19/2017 6:14:32 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
88  l|qoap     | 12/19/2017 6:14:32 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
89  l|qoap     | 12/19/2017 6:14:32 PM [32mCOAP[39m - Incoming POST request from 192.168.43.18 for home/kitchen
```

Gambar 5.13 Log Data Skalabilitas

5.5 Pengolahan Data

5.5.1 Pengolahan Data CPU dan Memori

Dari hasil log file program monitoring cpu dan memori yang dilakukan selama 3 jam diperoleh data sebanyak 720 karena program mencatat log penggunaan cpu dan memori setiap 15 detik sekali untuk skenario 1 hingga skenario 3. Dari 720 data tersebut dibagi menjadi tiga bagian masing-masing 240 data yang mana bagian pertama menunjukkan penggunaan cpu dan memori pada satu jam

pertama, bagian kedua menunjukkan penggunaan cpu dan memori pada satu jam kedua, dan bagian ketiga menunjukkan penggunaan cpu dan memori pada satu jam ketiga. Dari hasil pembagian tersebut diperoleh rata-rata penggunaan cpu dan memori mulai dari satu jam pertama hingga satu jam ketiga seperti pada Tabel 5.3 untuk rata-rata penggunaan cpu dan Tabel 5.4 untuk rata-rata penggunaan memori. Data selengkapnya dapat dilihat pada Lampiran A Penggunaan CPU dan Lampiran B Penggunaan Memori.

Tabel 5.3 Rata-Rata Penggunaan CPU

	MQTT	CoAP	MQTT+CoAP
1st hour	0,87%	0,91%	1,25%
2nd hour	0,84%	0,88%	1,22%
3rd hour	0,87%	0,84%	1,27%
Average	0,86%	0,88%	1,25%

Tabel 5.4 Rata-Rata Penggunaan Memori

	MQTT	CoAP	MQTT+CoAP
1st hour	10,10%	9,63%	12,58%
2nd hour	10,24%	9,73%	12,78%
3rd hour	10,37%	9,89%	12,99%
Average	10,24%	9,75%	12,79%

5.5.2 Pengolahan Data *Delay* dan Data Transfer

Data *delay* dan data transfer diperoleh dari hasil *capture traffic* menggunakan program tcpdump. Terdapat dua data *capture traffic*, yang pertama adalah *capture traffic* dari nodeMCU ke *middleware* dan yang kedua adalah *capture traffic* dari *middleware* ke aplikasi/web-app. Data yang diperoleh tersebut dilakukan analisis menggunakan program wireshark. Untuk memperoleh nilai *delay* adalah dengan mengurangi waktu saat paket diterima oleh dengan waktu saat paket dikirimkan. Sedangkan untuk mendapatkan hasil data transfer adalah dengan melakukan filter pada wireshark untuk masing-masing topik berdasarkan IP yang didapat kemudian dihitung berapa banyak data yang berhasil dikirimkan. Hasil yang didapat dari pengolahan data *delay* dan data transfer ditunjukkan oleh Tabel 5.5, Tabel 5.6, dan Tabel 5.7. Data selengkapnya dapat dilihat pada Lampiran C *Delay* dan Data Transfer.

Tabel 5.5 Data MQTT

MQTT						
Topic	Expected	Actual	Invalid	Loss Rate	Success rate	Average *Delay*(s)
barrack-1	120	120	0	0%	100%	0,357

MQTT						
Topic	Expected	Actual	Invalid	Loss Rate	Success rate	Average *Delay*(s)
barrack-2	120	120	0	0%	100%	0,566
barrack-3	120	120	0	0%	100%	0,123
barrack-4	120	120	0	0%	100%	0,578
barrack-5	120	120	0	0%	100%	0,242
Total	600	600	0	0%	100%	0,373

Tabel 5.6 Data CoAP

CoAP						
Topic	Expected	Actual	Invalid	Loss Rate	Success rate	Average *Delay* (s)
kitchen-1	120	120	0	0%	100%	0,776
kitchen-2	120	120	0	0%	100%	0,647
kitchen-3	120	120	0	0%	100%	0,653
kitchen-4	120	120	0	0%	100%	0,128
kitchen-5	120	120	0	0%	100%	0,131
Total	600	600	0	0%	100%	0,467

Tabel 5.7 Data Websocket

Websocket			
Expected	Actual	Overhead	Average *Delay* (s)
1200	1200	0%	0,495

5.5.3 Pengolahan Data Skalabilitas

Pada log file data skalabilitas terdapat log waktu kapan proses *publish/subscribe* dilakukan. Berdasarkan waktu tersebut dapat diketahui berapa banyak *publisher/subscriber* yang dapat ditangani oleh *middleware* dalam satu detik dengan variasi 100, 500, dan 1000 klien. Dilakukan perhitungan berapa banyak data yang memiliki log waktu yang sama sehingga menghasilkan data *concurrent publish/subscribe* dan menghitung jeda waktu saat proses *publish/subscribe* dimulai hingga waktu proses *publish/subscribe* selesai sehingga menghasilkan data *time publish/subscribe*.

Tabel 5.8 Skalabilitas CoAP

CoAP			
	100 Clients	500 Clients	1000 Clients
Publisher	41	35	39
Time Publish (s)	6	36	72

Tabel 5.9 Skalabilitas MQTT

MQTT			
	100 Clients	500 Clients	1000 Clients
Publisher	81	55	56
Time Publish (s)	3	10	20

Tabel 5.10 Skalabilitas Websocket

Websocket			
	100 Clients	500 Clients	1000 Clients
Subscriber	50	52	54
Time Subscribe (s)	3	12	22

Pada Tabel 5.8, Tabel 5.9, Tabel 5.10 perhitungan *publisher/subscriber* diambil berdasarkan jumlah maksimal *publisher/subscriber* dalam satu detik yang terdapat dalam log file skalabilitas. Sedangkan *time publish/subscribe* adalah jeda waktu saat proses *publish/subscribe* dimulai hingga waktu proses *publish/subscribe* selesai. Data selengkapnya dapat dilihat pada Lampiran D Skalabilitas.