Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 12, Desember 2018, hlm. 7521-7527 http://j-ptiik.ub.ac.id

Implementasi Protokol MQTT Untuk Sistem *Monitoring* Perangkat *IoT*

Zavero Brillianata Abilovani[1], Widhi Yahya[2], Fariz Andri Bakhtiar[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]zavero93@gmail.com, [2]widhi.yahya@ub.ac.id, [3]fariz@ub.ac.id

Abstrak

Perkembangan era internet pada masa kini mengembangkan teknologi yang bernama *Internet of Things* atau yang biasa dikenal sebagai *IoT*. Dimana *IoT* mampu untuk mentransmisikan data secara otomatis dari komputer ke komputer tanpa harus adanya inisiasi dari manusia. Namun dengan meningkatnya jumlah beserta ukuran data yang akan di transmisi akan berdampak negatif terhadap pada kinerja perangkat *IoT*. Pada permasalahan tersebut, solusi yang ditawarkan berupa sebuah sistem yang mampu memanajemen perangkat serta dapat mengawasi dan menjaga ketersediaan dari perangkat *IoT*. Salah satu cara adalah dengan membuat sebuah sistem *monitoring* pada perangkat *IoT*. Tujuan yang dimiliki adalah untuk mengawasi perangkat ketika terdapat gangguan. Dengan menerapkan prinsip yang terdapat pada protokol MQTT yaitu mekanisme *publish-subscribe* dan berjalan di *layer* aplikasi . Protokol MQTT dapat melakukan proses mengirim dan menerima pesan untuk mengatur dan me*monitoring* dengan yang dihendaki oleh *user*. Dalam mengirim dan menerima pesan ditentukan berdasarkan pada topik-topik. Berdasarkan hasil daripada pengujian, resource yang dibutuhkan lebih kecil. Dibandingkan protokol UDP ukuran paket data yang dimiliki ternyata lebih kecil dari protokol SNMP. Sistem *monitoring* dengan protokol MQTT pada perangkat *IoT* juga memiliki *delay* yang kecil yaitu 0.008634 *sec* dan nilai *throughput 9,2 MBit/sec*. Sehingga sistem tersebut dapat berjalan pada *bandwith* rendah beserta *latency* tinggi.

Kata kunci: *monitoring, mqtt, internet of things, protocol.*

Abstract

Internet era at this time has developed a technology called Internet of Things as known as IoT. Where IoT is able to transmit data automatically from computer to computer without the initiation from humans. However, with size and number of data that increased will be transmitted make IoT devices performance has a negative result. Efforts are being made to overcome these problems, namely the need for a system that is able to manage devices and can monitor and maintain availability of IoT devices. One way is to make an IoT device monitoring system that aims to monitor when there is a disruption to the device. By implementing the MQTT protocol which is a protocol that implements the publish-subscribe mechanism and runs on the application layer. The MQTT protocol can process the send and receive of messages to manage and monitor according to the user's wishes, because the send and receive of messages by predetermined topics. Based on results rather than testing, the MQTT protocol has a smaller resource requirement than the UDP protocol and has a smaller data packet size than the SNMP protocol. The IoT device monitoring system that uses the MQTT protocol also has a small delay of 0.008634 sec and a throughput value of 9.2 MBit / sec. So that the system can run on low bandwidth and high latency.

Keywords: *monitoring, mqtt, internet of things, protocol.*

1. PENDAHULUAN

Kemajuan yang sangat cepat pada era teknologi sangatlah cepat. Diantaranya yaitu *Internet of Things* atau yang biasa dikenal dengan *IoT*. IoT adalah sebuah objek baik itu benda maupun perangkat dengan kemampuan menerima serta mengirim data di dalam jaringan dengan tidak disertai adanya interaksi dari manusia dengan manusia bahkan manusia dengan komputer (Rose,et all.,2015). Perangkat IoT sendiri saat ini sudah banyak digunakan dan diterapkan dalam berbagai aspek

kehidupan, contohnya yaitu pada bidang transportasi, bidang keamanan, bidang kesehatan, bidang pertanian serta bidang pemerintahan (Gubbi.J., 2013). Penelitian serta riset pada *IoT* saat ini terus dilakukan.

Saat ini perkembangan *IoT* yang pesat memiliki banyak tantangan. Untuk mengatasi tantangan ini, diperlukan sebuah proses manajemen sistem pada perangkat untuk mengawasi ketersediaan pada perangkat *IoT* digunakan sistem *monitoring*. Sistem monitoring pada perangkat *IoT* adalah sebuah sistem yang digunakan untuk *monitoring* perangkat IoT itu sendiri baik dari resource, *network* maupun *disk*. Sehingga apabila pada perangkat terjadi gangguan dan masalah, dapat diketahui segera masalah yang terjadi pada perangkat *IoT* itu. Proses pengumpulan data dari perangkat itu sendiri dilakukan oleh sistem monitoring yang kemudian memaksimalkan sumber daya yang dimiliki dengan menganalisis nya.(Prasetyo, 2013).

Berbagi macam protokol jaringan yang digunakan dalam penelitian Iot diantaranya adalah SNMP atau *Simple Network Management Protokol*. Akan tetapi *monitoring* perangkat IoT yang menggunakan protokol SNMP ternyata kurang efisien. Banyak batasan yang dijumpai oleh protokol SNMP. Baik dari segi kinerja pada CPU, aspek ukuran penggunaan memori dengan jumlah besar saat melakukan pengamatan pada perangkat IoT (Jacquout.,2010).

Oleh sebab itu pada monitoring jaringan diperlukan sebuah solusi yang efektif. Metode menggabungkan protokol lain seperti MQTT atau COAP dengan SNM dapat menjadi salah satu contohnya. Pada agent dilakukan perubahan pada proses mengirim data.(Savic,2016).

Kemudian pada sistem monitoring untuk perangkat IoT dilakukan pengembangan yang memakai protokol MQTT. Protokol ini dinilai sesuai dengan perangkat *IoT* yang bersifat *lightweighted message* serta dirancang pada perangkat dengan sumber daya minimum. Prinsip *publish/subscribe* yang dimiliki protokol MQTT dan dapat beradaptasi dalam mengirim dan menerima pesan dalam *monitoring* yang berdasarkan dengan topik yang dikehendaki. Alasan utama dipilihnya protokol MQTT yang diterapkan pada perangkat *IOT* ialah karena jumlah penggunaan energi yang jauh lebih sedikit bila dibandingkan dengan protokol yang lain dan dapat bekerja dengan kondisi bandwith kecil serta latency tinggi.(Kim, 2015).

2. LANDASAN KEPUSTAKAAN

Pada pengembangkan sistem *monitoring* perangkat *IoT* diperlukan beberapa sumber acuan baik itu dari buku,jurnal maupun riset sebagai acuan dalam tahap merancang dan mengimplementasikan. Dasar penting pada penelitian ini antara lain, *Raspberry Pi2 model B*, Twisted, protokol MQTT, *Psutil, JSON, Flask, SQLite* beserta *monitoring* jaringan

2.1 Raspberry Pi 2 model B

Raspberry Pi 2 model B merupakan nama lain dari perangkat CPU berukuran kecil yang dirilis oleh The *Raspberry Pi Foundation*. Diharapkan dengan adanya Raspberry Pi meningkatkan minat pendidikan sains komputer. Komputer berukuran kecil dengan nilai yang bias dijangkau oleh masyarakat ini bisa menjadi alat yang multifungsional. Terdapat juga konektor *input* dan *output* serta perangkat keras komputer itu sendiri pada papan sirkuit cetak.

Gambar 1. Perangkat Raspberry Pi 2 Model B

Raspberry Pi model B ialah sebuah komputer mini seukuran kartu atm dengan spesifikasi *Broadcom* jenis *BCM2835* pada *chip (SoC)* pada sistemnya. *SoC* ini dilengkapi prosesor jenis ARM1176JZFS 32-bit, juga *clock* berukuran 700MHz, dan PU Videocore 4. Perangkat ini dilengkapi dengan RAM 256 MB dengan POP paket di atas SoC. Selain itu Raspberry Pi juga disertai dengan charger AC dengan daya 5V yang setara dengan 4 buah baterai jenis AA. Sementara *CPU* ARM yang dimiliki menghasilkan kemampuan yang tidak jauh dari *Pentium* 2 300MHz. Inti grafis yang dimiliki yaitu GPU *Broadcom*, mampu mengolah beberapa jenis video dengan definisi yang tinggi.(Zhao S.C., 2015)

Raspberry Pi model B ini juga memiliki *port* HDMI dan keluaran video komposit, empat buah *port* USB tipe 2.0, satu buah *port* Ethernet 10/100, dan *slot* kartu microSD, serta konektor bertipe GPIO serta pada audio disertai output audio bertipe *analog*.

2.2 *Twisted*

Twisted merupakan sebuah jenis *framework* dengan metode *event-based programming*. Metode *Event-based programming* menggunakan trigger untuk menjalankan banyak event dalam waktu yang bersamaan. Ketika berjalan, proses yang dihasilkan nantinya mampu berjalan secara efektif dan efisien. Selain itu program juga dapat berjalan dengan ringan. Pendekatan ini. (Paykin.,2016).

2.3 MQTT

Message Queue Telemetry Transport atau yang biasa disebut *MQTT* yaitu protokol untuk komunikasi yang bersifat *machine to machine* atau M2M dan bekerja di *layer* ketujuh atau aplikasi dan bersisifat *lightweight message*. Meskipun koneksi dalam keadaan terputus, semua pesan yang dikirim akan terjamin oleh protokol MQTT. Metode komunikasi *publish*/*subscribe* merupakan metode pengiriman yang digunakan oleh protokol MQTT. Pesan pada MQTT dikirim ke *broker* dan berisi topik yang dikirimkan oleh *publisher*. Kemudian topik tadi diolah untuk diteruskan ke *subscriber* berdasarkan dari permintaan pengguna.

2.4 Psutil

Psutil merupakan sebuah jenis library yang berorientasi dalam bentuk bahasa pemrograman *python* dengan kegunaan mengambil sejumlah jenis data dari dalam proses yang sedang berjalan pada perangkat berdasarkan pada prosesnya. Data yang diambil merupakan data dalam bentuk proses yang sedang berjalan, data bisa berupa disk, CPU, memory, *network*, dan jenis data lainnya.

2.5 *JSON*

JavaScript Object Notation atau yang dikenal dengan JSON merupakan sebuah format teks yang tidak memiliki ketergantungan terhadap bahasa pemrograman tertentu serta dapat digunakan sebagai sarana dalam proses pertukaran data pada *platform* tertentu. Selain itu JSON juga memiliki sebuah keuntungan jika menggunakannya mudah untuk dimengerti dan dibaca oleh pengguna karena tersusun berpasang-pasang.

2.6 Flask

Flask merupakan sebuah web microframework yang mengacu pada python sebagai bahasa dalam pemrograman. Beberapa fungsi yang dimiliki oleh flask sangat tepat jika diterapkan pada program yang memiliki jumlah memori dan energi yang terbatas. Sebagai salah satu *framework* dengan kategori ringan, terdapat juga kegunaan flask yang lain yaitu mempunyai kemampuan yang baik jika di*integrasi*kan dengan SQLAlchemy pada *database*nya. Kemampuan tersebut dapat memaksimalkan kinerja dari aplikasi yang menggunakan *database* untuk menyimpanan sejumlah data.

2.7 SQLite

SQLite merupakan aplikasi *database* yang bisa diterapkan pada berbagai macam aplikasi mulai dari personal maupun komersial. Prinsip yang dimiliki oleh SQLite adalah *open source* dan ditulis oleh D. Richard Hipp dengan bahasa pemrogaman C. Jika dilihat secara garis besar dengan mesin *database* yang lainnya misalnya pada Oracle, SQL Server dan lain sebagainya. SQLite dapat dikategorikan ringan sebagai jenis *database*.(Bhosale.,2015).

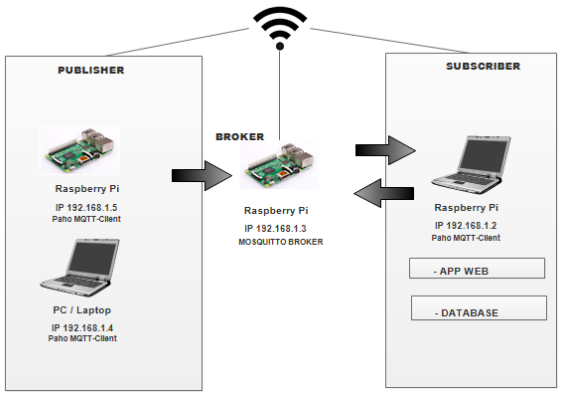
2.8 *Monitoring* Jaringan

Monitoring pada jaringan yaitu kemampuan dalam mengamati, melakukan kontrol serta mengambil analisa pada sebuah sistem atau pada jaringan komputer. Salah satu pokok penting dari *monitoring* jaringan adalah proses *monitoring* pada komputer atau perangkat lainnya. Dasar dari konsep manajemen pada jaringan ini memiliki dasar konsep adanya *agent* yaitu segala jenis perangkat yang dapat diamati serta *manager* yaitu perangkat yang menjalankan proses *monitoring* dan (Pradikta.,2013).

3. PERANCANGAN

Dalam bagian perancangan protokol MQTT ke dalam sistem *monitoring*, ada tiga

peran penting di dalamnya antara lain *broker*, *publisher* dan *subscriber* dan *broker*. *Subscriber* dirancang sebagai peran yang digunakan oleh pengguna dalam meminta data hasil *monitoring* dari perangkat *IoT*. Terdapat *paho* MQTT pada subscriber sebagai *client* dalam sistem *monitoring*. Dilengkapi juga dengan SQLite *database* yang berfungsi sebagai penyimpan data *monitoring*. Kemudian terdapat juga aplikasi *web* yang bertugas mengelola informasi dan data untuk diolah dan ditampilkan pada pengguna dengan alamat yang sudah disediakan. Sedangkan pada *broker* dipasang *mosquitto broker* dengan peran sebagai *broker* di dalam arsitektur MQTT. Broker inilah yang berfungsi sebagai pengolah data berdasarkan *topic*. Kemudian yang ketiga ialah *publisher* yang juga di dalamnya dilengkapi dengan *paho* MQTT sebagai *client* dalam sistem *monitoring*.



Gambar 2. Rancangan Topologi pada Sistem *Monitoring*

Seperti yang terlihat pada gambar 2 sebagai rancangan dari topologi, pada sistem *monitoring* ini dirancang dengan dua buah perangkat *IoT* yaitu Raspberry Pi dan laptop sebagai *publisher* yang akan diamati oleh pengguna. Kemudian terdapat Raspberry Pi sebagai *broker* yang berada diantara *publisher* dan *subscriber* sebagai penerima dan pengolah data yang dikirimkan dan kemudian dilanjutkan dengan pengiriman kepada *subscriber*. Untuk perancangan pada subscriber menggunakan laptop yang langsung mengolah dan menampilkan data kepada pengguna. Di dalamnya juga terdapat *database* di desain dengan empat kategori tabel sebagai penyimpanan data. Pada tabel pertama yaitu tabel yang terkoneksi pada jaringan dan menyimpan data dari host yang sedang aktif. Fungsi yang dimiliki tabel ini sebagai proses autentifikasi. Kemudian tabel kedua sebagai tabel yang berfungsi sebagai penyimpan dari perintah untuk proses *monitoring*. Dasar dari *manager* merupakan fungsi dari tabel ini yaitu mengirim topik yang diminta oleh pengguna disertai dengan durasi dan *interval* di dalamnya. Tabel untuk menyimpan data merupakan tabel yang ketiga yaitu kategori *resource*. Sedangkan tabel untuk menyimpan data merupakan tabel yang keempat untuk menyimpan kategori *network*.

Dibutuhkan sebuah aplikasi web dalam menampilkan hasil data dari proses *monitoring*. Tampilan pada sistem *monitoring* dirancang melalui aplikasi *web*. Oleh sebab itu fungsi menerima dan memasukkan data ke dalam *database* harus dimiliki oleh aplikasi *web* ini.

4. IMPLEMENTASI

Dalam tahap implementasi pada *publisher*, python digunakan sebagai bahasa pemrograman dengan *library psutil* dan *framework twisted* untuk mengambil data di dalam proses *monitoring*. Pada *publisher* rancangan yang telah dibuat disesuaikan kemudian diimplementasikan.

Tabel 1 Tabel Pada Topik *Resource*

Isi Data Pada Topik *Resource*	**Fungsi *library* psutil yang digunakan**
CPU Usage	Psutil cpu percent
Memory Used	Psutil virtual memory used
Memory Available	Psutil virtual memory available
Swap	Psutil virtual memory free

Pada tabel diatas di dapatkan data dalam bentuk *resource*, data yang diambil berupa *cpu usage*, *memory used*, *memory available* dan *swap*.

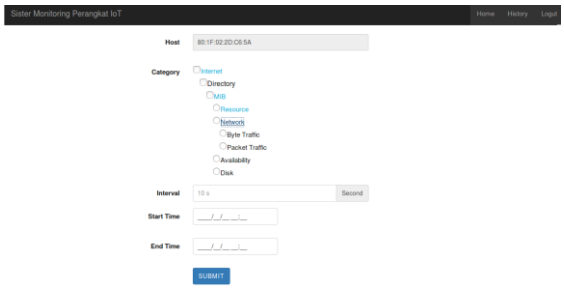
Tabel 2 Tabel Pada Topik *Network*

Isi Data Pada Topik Network	**Fungsi *library* psutil yang digunakan**

Byte Sent	Psutil net io counters bytes sent
Byte Receive	Psutil net io counters bytes recv
Packet Sent	psutil.net io counters packets sent
Packet Receive	Psutil net io counters packets recv

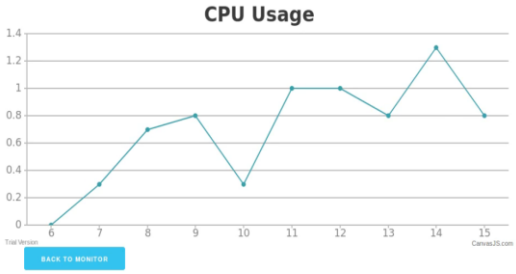
Dari Tabel 2 data yang didapatkan merupakan topik *network*, didapatkan data antara lain *byte sent*, *byte receive*, paket *sent*, serta paket *receive*.

Setelah diperoleh hasil monitoring, data akan diubah sebagai topic-topic pada monitoring yakni resource dan network. Selanjutnya broker akan menerima massage dalam format. Pada implementasi broker terdapat Mosquitto broker yang berfungsi sebagai penerima data dari publisher kemudian, mengirimkan topic tersebut kepada subscriber sesuai dengan request.Selanjutnya data diterima oleh *subscriber* dan disimpan ke dalam *database*, data yang diperoleh akan diolah selanjutnya ditampilkan pada aplikasi *web* halaman monitor. Implementasi Aplikasi Web pada Sistem *Monitoring* menggunakan web *framework Flask.*



Gambar 3. Halaman Utama Sistem *Monitoring*

Halaman utama sistem monitoring yang terdapat pada gambar 3 merupakan halaman sebagai penambah perintah *monitoring* yang terdapat di dalam Sistem *Monitoring* IoT yaitu fitur *Add Monitor*. Pada halaman ini terdapat beberapa isian *form seperti host address, monitoring category OID, monitoring interval*, dan durasi dari waktu serta tanggal pengiriman proses *monitoring*. Apabila tombol *submit* dipilih oleh pengguna, maka dalam *database* akan menyimpan parameter perintah yang sudah di*set* oleh pengguna.. Selanjutnya manager akan mengambil parameter perintah yang sudah diterima untuk dikirimkan ke *agent.*



Gambar 4. Halaman yang menampilkan CPU Usage

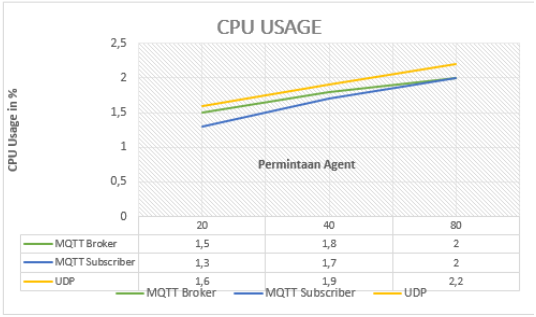
Pada database akan menyimpan hasil data dari sesuai dengan *topic* yang di *request* oleh *subscriber.* Selanjutnya akan ditampilkan grafik pada halaman *monitoring* dari hasil pengolahan data.

5. PENGUJIAN DAN ANALISIS

Tahapan yang dilalui ketika selesainya proses implementasi yaitu menguji dan melakukan analisa terhadap sistem yang berjalan. Berbagai macam pengujian dilakukan antara lain, menguji ukuran paket data, menguji performa pada sistem, menguji fungsi pada aplikasi *web*, menguji jumlah nilai pada *delay* serta pada *throughput*. Aplikasi wireshark digunakan dalam mengolah dan melakukan pengambilan analisa dari pengujian tersebut.

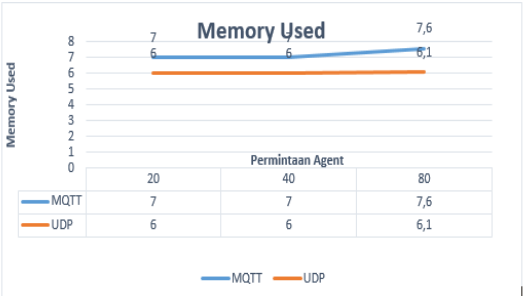
Dalam menguji ukuran besar paket data yang dimiliki dengan melakukan filter pada wireshark diperoleh nilai sebagai berikut. Protokol MQTT memiliki jumlah nilai yang lebih kecil dari protokol SNMP. Akan tetapi protokol MQTT bila dibandingkan dengan protokol UDP ternyata memiliki jumlah ukuran paket data yang lebih besar disebabkan lebih banyaknya proses yang terjadi dalam satu kali transfer pada MQTT.

Kemudian pengujian selanjutnya adalah menguji dari performa yang dimiliki oleh protokol MQTT dan UDP sesuai dengan jumlah agent yang ditentukan. Pada MQTT, fungsi on_publish digunakan sebagi pengganti thread yang ada dalam pengujian performa pada protokol UDP. Fungsi tersebut bekerja seperti layaknya sebuah thread yang dapat menjalankan beberapa proses bersamaan.



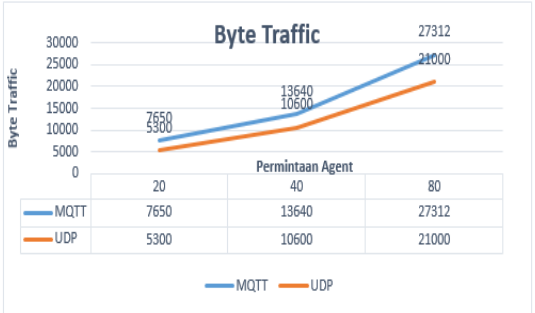
Gambar 5. Hasil pengujian besar paket data

Pada gambar 5 diatas dapat diamati bahwa grafik tersebut menunjukkan data dari CPU *usage* yang ketika jumlah agent bertambah maka nilai pada grafik juga akan mengalami peningkatan. Hasil yang terlihat sangat berbeda ini sesuai dengan jumlah *agent* yang meningkat dua kali lipat setiap pengambilan data. Kinerja yang dimiliki dipengaruhi oleh permintaan jumlah publisher.



Gambar 6. Hasil Pengujian Penggunaan Memori

Pada gambar 6 terlihat perubahan kenaikan pada penggunaan memori yang disebabkan oleh meningkatnya jumlah *publisher*. Hasil lain yang diperoleh ternyata menunjukkan protokol UDP ternyata menggunakan lebih sedikit memori jika dibandingkan protokol MQTT.



Gambar 7. Hasil Pengujian Byte Traffic.

Grafik pada *byte traffic* di gambar 7 meningkat akibat pengaruh jumlah *publisher*. Hal yang sama terjadi sesuai dengan hasil percobaan bahwa grafik yang ditampilkan bersifat *linier* yaitu meningkat sebanding dengan semakin banyaknya *publisher*.

Melalui pengujian performa pada sistem dangan perbandingan antara kinerja yang dihasilkan dari protokol MQTT dengan protokol UDP diperoleh dua buah kesimpulan. Yang pertama, di dalam proses satu kali transaksi, nilai penggunaan sumber daya CPU pada protokol MQTT lebih sedikit bila dibandingkan dengan protokol UDP.

Kemudian yang kedua yaitu protokol MQTT yang membutuhkan memori dan byte traffic yang lebih banyak jika dibandingkan dengan protokol UDP. Hasil tersebut akibat dari jumlah proses yang terjadi pada protokol MQTT dalam satu kali transaksi data. Oleh sebab itu semakin tinggi jumlah publisher maka akan terbukti peningkatan pada grafik.

Selanjutnya pada aplikasi web dilakukan pengujian untuk mengetahui apakah aplikasi web sudah berjalan baik dan menghasilkan nilai keluaran sesuai dengan keinginan pengguna.

Melalui pengujian uji coba fungsi pada aplikasi web dihasilkan informasi yang diolah melalui tabel tiga. Dalam tabel tersebut ditunjukkan bahwa nilai keluaran yang dihasilkan sudah berjalan dengan baik. Demikian pula jika uji coba ini dilakukan untuk mencoba apakah aplikasi web mampu menampilkan grafik dari data yang telah diolah.

Tabel 3 Pengujian fungsional aplikasi web

No	Deskripsi	Masukan	Keluaran yang diharapkan
1	Pengujian menampilkan *list device*	Table *host* pada *database*	Sistem menampilkan *list device*
2	Pengujian menampilkan status *device*	Table *host* pada *database*	Sistem menampilan status *device* (up atau *down*)
3	Pengujian menghapus *device*	Tekan tombol *delete host*	Sistem menghapus *host* dari *database*
4	Pengujian memilih *device*	Tekan tombol monitor	Sistem menampilkan form add monitor dengan nama host sesuai yang dipilih
5	Pengujian penyimpanan perintah *monitoring*	Tekan tombol submit	Sistem menyimpan perintah ke database dan muncul tulisan *Add Success*
6	Pengujian filter berdasarkan host yang dipilih	Memilih *host* pada *host address* dan tekan tombol *filter*	Sistem menampilkan grafik berdasarkan host
7	Pengujian filter berdasarkan kategori komponen yang dimonitoring	Memilih komponen pada *monitoring category*	Sistem menampilkan grafik berdasarkan komponen kategori

Pada tahap terakhir di pengujian yaitu mengenai pengujian pada *QoS* yaitu *delay* dan *throughput*. Berdasarkan perhitungan yang

didapatkan saat menguji nilai *delay* dan *throughput* dari protokol MQTT pada sistem *monitoring* perangkat *IoT* tidak terlalu besar. Nilai pada *delay* yang dihasilkan yaitu 0.008634 sec, Kemudian nilai pada *throughput* yang diperoleh ialah sebesar *9,2 MBit/sec.* Melalui pengamatan tersebut dapat diperoleh kesimpulan bahwa protokol MQTT yang digunakan sistem *monitoring* perangkat *IoT* mampu berjalan dengan kondisi bandwith kecil dan latency tinggi.

6. KESIMPULAN

Protokol MQTT mampu berjalan dengan baik dan lancar ketika diimplementasikan pada sistem *monitoring* perangkat *IoT*. Prinsip *publish-subscribe* yang diterapkan mampu berjalan dengan baik pada proses transfer data dan bisa memperoleh topik yang berisi informasi yang diminta oleh pengguna berdasarkan pengamatan dari perangkat yang ditentukan.

Diperoleh juga kesimpulan mengenai *resource* dari hasil pengujian, bahwa kebutuhan *resource* pada protokol MQTT lebih sedikit dibanding protokol UDP. Hasil tersebut akan sama jika dibandingkan dengan protokol SNMP. Protokol MQTT yang diterapkan pada sistem *monitoring* perangkat *IoT* juga memiliki *delay* kecil yaitu 0.008634 *sec* dan nilai *throughput 9,2 MBit/sec*. Sehingga dapat disimpulkan bahwa sistem tersebut mampu berjalan pada keadaan *bandwith* yang rendah dan *latency* yang tinggi.

.

7. DAFTAR PUSTAKA

Eclipse, 2018. *iot.eclipse.org*. [Online] Available at: https://iot.eclipse.org/resources/white-papers/Eclipse%20IoT%20White%20Paper%20-%20The%20Three%20Software%20Stacks%20Required%20for%20IoT%20Architectures.pdf

Flask. 2018. Flask(A Python Microframework). In *http://www.flask.pocoo.org*..

Gubbi,J .Rajkummar, B. Marusic, S. Palaniswami M. 2013. *Internet of Things (IoT): A vision, architectural elements, and future directions. Elsevier B.V.*

Jacquout, A. et al., 2010. *A New Management Method. IEEE IFIP Annual Mediterranean Ad Hoc Networking Worshop*.

Json. (2018). Inroducing JSON. In *http://www.json.org*.

Kim, S.-M., Choi, H.-S. & Rhee, W.-S., 2015. *IoT* Home Gateway for Auto-Configuration and.

Mihajlo, Savić. (2016). Bridging The Snmp Gap: Simple Network Monitoring The Internet Of Things. Facta universitatis-series: Electronics and Energetics.

Pradikta, R., Affandi, A., & Setijadi, E. 2013. Rancang Bangun Aplikasi *Monitoring* Jaringan dengan Menggunakan Simple Network Management Protocol.

Prasetyo, I. 2013. Pengenalan *Monitoring* Jaringan Komputer.

Python. (2018). psutil Python Package Index. In http://pypi.python.org/pypi/psutil.

Rose, K., Eldridge, S., & Chapin, L. 2015. The Internet Of Things: An Overview. Understanding the Issues and Challenges of a More Connected World.

SQLite, 2018. *SQlite Home Page.* [Online] Available at: http://sqlite.org [Diakses 15 Maret 2018].

Zhao, S. C. 2015. Exploring IOT Application Using Raspberry Pi . *International Journal of Computer Networks and Applications*.