BAB 1 PENDAHULUAN

1.1 Latar belakang

Teknologi informasi dan komunikasi telah berkembang sangat pesat dan menghadirkan suatu konsep yang dinamakan dengan *Internet of Things* (IoT). IoT dapat diartikan sebagai konsep untuk menghubungkan "*Things*" atau benda-benda nyata di dunia dengan *internet* dan saling berbagi informasi. Iot mengubah benda-benda fisik di dunia dari benda tradisional menjadi *smart* (Al-Fuqaha, et al., 2015). Konsep IoT memungkinkan benda-benda fisik untuk melihat, mendengar, berpikir, berbagi informasi dan mengkoordinasikan keputusan. Iot diharapkan dapat berkontribusi pada peningkatan kualitas hidup masyarakat luas seperti contohnya pada *smart-homes*.

Untuk mengirimkan informasi antar *"Things"* melalui *internet*, Iot membutuhkan protokol komunikasi yang ideal dan efisien dalam hal *resource* ketika mengirimkan pesan. Hal itu disebabkan karena pada umumnya Iot dibangun dengan *constrained devices* yang memiliki keterbatasan dalam hal *memory*, *storage* dan *power* (Petersen, Bacelli & Wahlisch, 2014). Salah satu protokol yang dianggap cocok dengan model komunikasi IoT adalah protokol *Message Queueing Telemetry Transport* (MQTT). Protokol MQTT mengungguli *Hypertext Transfer Protocol* (HTTP) dalam komunikasi IoT dikarenakan memiliki *overhead protocol* yang lebih rendah dan lebih efisien dalam hal penggunaan *bandwidth* serta *network resources* ketika berkomunikasi (Yokotani & Sasaki, 2016). Hal itu disebabkan karena protokol MQTT adalah protokol yang secara spesifik didesain untuk *devices* yang memiliki keterbatasan *resources* dengan *high-latency* dan *low bandwidth* (Happ & Wolisz, 2016).

Pada model komunikasi protokol MQTT, *broker* memegang peranan penting pada keberhasilan proses komunikasi yang terjadi. Hal itu disebabkan karena komunikasi antara *publisher* dan *subscriber* terjadi secara asinkron yang artinya komunikasi yang terjadi harus melalui perantara *broker* (Hayun & Wibisono, 2017). *Broker* adalah komponen yang menjembatani komunikasi dan memastikan pesan dapat diterima dari *publisher* kepada *subscriber* (Hunkeler, Truong & Stanford-Clark, 2008). Akan tetapi, *broker* yang menjadi jembatan komunikasi memiliki kemungkinan tidak dapat tersedia. Hal itu dapat disebabkan karena *broker* memiliki permasalahan pada jaringan atau perangkat keras yang digunakan. *Publisher* dan *subscriber* akan kehilangan *service* ketika broker tidak tersedia yang menyebabkan *publisher* tidak dapat melakukan *publish* informasi dan *subscriber* tidak dapat melakukan *subscribe* suatu topik.

Fokus penelitian ini adalah menanggulangi permasalahan terhentinya komunikasi antara *publisher* dan *subscriber* dengan *broker* dalam waktu yang lama ketika *broker* mengalami kegagalan. Mekanisme yang dapat digunakan adalah dengan mengimplementasikan *failover* pada *broker*. *Failover* atau juga disebut *High-Availability clusters* adalah kumpulan dari beberapa komputer yang mendukung aplikasi *server* dengan *downtime* yang minimal karena adanya grup

yang terdiri dari beberapa *server* (Kahanwal & Singh, 2012). *Failover* dapat meminimalisir *downtime* yang terjadi karena adanya komponen lain dalam grup akan menggantikan komponen yang gagal secara otomatis tanpa adanya intervensi dari *user*.

Untuk dapat mengimplementasikan *failover*, *broker* dari MQTT harus menggunakan *message broker* yang memiliki fitur *broker cluster*. Hal itu disebabkan karena *failover* bekerja dengan membentuk sebuah *cluster* yang terdiri dari minimal 2 *node* (Pribadi, 2013). *Cluster* digunakan agar dapat menggunakan redundansi yang mendeteksi kegagalan pada salah satu *node* dalam *cluster* dan menggantikan perannya dengan *node* lain dalam *cluster*. Salah satu *message broker* yang memiliki fitur *broker cluster* adalah *Apache ActiveMQ*. *ActiveMQ* adalah *open source message broker* yang dimiliki oleh *Apache Software Foundation*. *ActiveMQ* memiliki kemampuan konfigurasi *clustering* yang dinamakan dengan *network of brokers* seperti contohnya pada konfigurasi *master-slave* (Magnoni, 2015). Oleh karena itu, penelitian ini menggunakan *message broker ActiveMQ* yang memiliki fitur *broker cluster* agar dapat mengimplementasikan *failover*.

Beberapa peneliti telah melakukan penelitian terkait dengan *failover* untuk mengatasi permasalahan *availability* suatu sistem. Penelitian dari Pribadi (2013) membahas mengenai ketersediaan data dan layanan pada perusahaan sangat dibutuhkan dalam mendukung proses bisnis. Berdasarkan hasil penelitian, implementasi *failover* memastikan bahwa layanan tetap tersedia dengan adanya *primary* dan *secondary server*. Penelitian selanjutnya membahas tentang implementasi *failover clustering* pada *web server* untuk mengatasi permasalahan ketersediaan layanan suatu *website* yang dibutuhkan untuk memberikan informasi. Implementasi dilakukan dengan membangun *cluster* yang terdiri dari *server* aktif dan pasif yang mengimplementasikan *failover* untuk mengatasi kegagalan pada *server*. Berdasarkan hasil penelitian, implementasi *failover* dengan *cluster* dapat meminimalisir *downtime* yang dihasilkan yaitu 5 detik (Juliharta, Supedana & Hostiadi, 2015). Berdasarkan pada beberapa penelitian sebelumnya, *failover* dapat dijadikan solusi untuk mengatasi permasalahan ketersediaan layanan suatu sistem dengan adanya 2 *node* yang meminimalisir waktu *downtime*.

Berdasarkan latar belakang yang diuraikan pada paragraf sebelumnya, penulis mengangkat judul "Implementasi Metode *Failover* Pada *Broker* Protokol MQTT Dengan *ActiveMQ*". Penelitian ini diharapkan dapat meminimalisir kendala pada komunikasi antara *publisher* dan *subscriber* ketika *broker* tidak dapat tersedia. Penelitian ini juga diharapkan dapat meningkatkan tingkat *availability* dari *broker* yang menjadi media komunikasi dalam protokol MQTT.

1.2 Rumusan masalah

Berdasarkan latar belakang, berikut adalah rumusan masalah yang tercakup pada penelitian ini:

1. Bagaimana mengimplementasikan *failover* pada *broker* protokol MQTT dengan *ActiveMQ*?
2. Bagaimana *availability* dan performa protokol MQTT yang mengimplementasikan metode *failover*?

1.3 Tujuan

Tujuan dari penelitian ini adalah:

1. Mengimplementasikan metode *failover* pada protokol MQTT dengan *ActiveMQ* untuk meningkatkan *availability*.
2. Menguji tingkat *availability* pada *broker* protokol MQTT yang mengimplementasikan metode *failover*.

1.4 Manfaat

Manfaat dari penelitian ini adalah:

1. Untuk umum dapat membantu pengembangan penelitian mengenai topik protokol MQTT.
2. Untuk peneliti dapat memahami sistem kerja dari *publish* dan *subscribe* dari protokol MQTT yang menerapkan metode *failover*.

1.5 Batasan masalah

Batasan masalah pada penelitian ini antara lain yaitu:

1. Sistem mengimplementasikan *failover* yang digunakan untuk menangani kegagalan pada perangkat keras serta lunak *broker* dan tidak menangani kegagalan yang diakibatkan oleh kondisi jaringan.
2. Pengujian tingkat *availability* dilakukan dengan mengukur nilai *downtime*.
3. Sistem hanya mengimplementasikan *failover* tanpa adanya *failback*.

1.6 Sistematika pembahasan

BAB I : PENDAHULUAN
Bab ini membahas tentang latar belakang, rumusan masalah penelitian, tujuan dari penelitian, manfaat dari penelitian, batasan masalah dari penelitian dan sistematika dari penelitian.

BAB II : LANDASAN KEPUSTAKAAN
Bab ini membahas tentang dasar teori dan kajian pustaka yang menjadi landasan dalam implementasi metode *failover* pada *broker* dari protokol MQTT.

BAB III : METODOLOGI
Bab ini menjelaskan tentang metode yang digunakan pada penelitian ini dan langkah penerapan nya yang diantaranya

adalah studi literatur, analisis kebutuhan sistem, perancangan, implementasi, pengujian dan kesimpulan.

BAB IV : PERANCANGAN

Bab ini menjelaskan tentang perancangan setiap komponen yang berperan dalam sistem. Perancangan dijelaskan mulai dari perancangan sistem, perancangan MQTT *publisher*, *subscriber* dan *broker*. Perancangan juga menjelaskan alur yang dilakukan ketika mengimplementasikan *failover*.

BAB V : IMPLEMENTASI

Bab ini menjelaskan tentang implementasi dari sistem pada penelitian ini. Perancangan dijelaskan mulai dari implementasi yang dilakukan pada masing-masing komponen sistem yaitu MQTT *publisher*, *subscriber* dan *broker*. Implementasi juga menjelaskan implementasi *failover* pada *broker* dan batasan dari implementasi.

BAB VI : PENGUJIAN

Bab ini menjelaskan tentang pengujian dan analisa hasil pengujian dari sistem pada penelitian ini. Pengujian dilakukan dengan beberapa skenario untuk mengetahui kebutuhan fungsional dan non-fungsional sistem telah terpenuhi.

BAB VII : PENUTUP

Bab ini menjelaskan tentang kesimpulan dan saran dari sistem berdasarkan pelaksanaan penelitian dari proses perancangan, implementasi, dan pengujian.