1. PENDAHULUAN

1.1. Latar Belakang

Pada era modern ini internet sudah menjadi bagian dari kehidupan sehari-hari. Sebanyak 5.40 milliar orang di dunia menggunakan internet pada awal quartal tahun 2023 atau sama dengan 65.7% dari total populasi dunia. Pengguna internet bertumbuh seiring waktu, dengan data terbaru yang mengindikasikan bahwa pengguna internet telah bertumbuh pada 12 bulan terakhir menuju oktober 2023 hingga 189 juta orang [1]. Data tersebut mengindikasikan bahwa betapa berpengaruhnya internet di dunia ini, dengan internet setiap individu di dunia dapat mengakses berbagai macam informasi dan berinteraksi dengan sesama melalui akses internet dan perangkat yang menunjang dalam pengaksesan internet. Oleh karena itu pemanfaatan internet di era modern menjadi semakin luas salah satunya adalah komputasi awan (*Cloud Computing*) [2].

Cloud Compuing merupakan model atau cara pengaksesan layanan komputasi atau sumber daya komputasi secara virtual seperti *server*, *database*, *networking*, dan *software* melalui internet atau dalam arti lain ”*cloud*” atau bisa disebut *data center* yang tersedia untuk pengguna secara online [3]. *Cloud* computing memberikan berbagai keuntungan dimana pengguna tidak perlu menyediakan sumber daya fisik untuk menjalankan suatu aplikasi atau dependensi lainnya melainkan cukup dengan koneksi dan akses internet. *Cloud* computing menawarkan *on demand availability* dari setiap sumber daya komputasi dan kombinasi dari *hardware* dan *software* secara online. Beberapa perusahaan yang menyediakan layanan *Cloud Computing* diantaranya adalah Google (Google *Cloud* Platform) dan Amazon (Amazon Web Services). Layanan pada *cloud* computing disebut dengan *Cloud Services* yang merupakan suatu layanan sumber daya virtual yang bisa digunakan untuk berbagai macam kebutuhan komputasi [4]. Penggunaan layanan *cloud* sudah banyak diadopsi salah satunya pada teknologi Internet of Things (IoT).

Menghubungkan berbagai objek ke internet telah menciptakan pergeseran signifikan dalam ekonomi berbasis informasi. Aliran informasi memainkan peran yang krusial pada ekosistem teknologi informasi. *Internet of Things (IoT)* merupakan konektivitas dari berbagai tipe objek fisik fisik, sensor, dan perangkat akhir ke internet. Objek fisik ini dapat berupa perangkat apa pun yang ditandai dengan sensor (misalnya ponsel pintar, peralatan listrik pintar, peralatan kantor pintar, mobil, dan sebagainya). Jumlah perangkat IoT yang terkoneksi di seluruh dunia di perkirakan meningkat dari 8,74 juta perangkat pada 2020 menjadi lebih dari 25,4 juta perangkat IoT pada tahun 2030. Data yang dihasilkan oleh perangkat IoT nantiya di kirim melalui jaringan kabel atau nirkabel menuju server yang berlokasi di *cloud* atau *on-premise* [5]. Karena perangkat IoT(sensor, aktuator, dan lain-lain) pada umumnya memiliki keterbatasan sumber daya (misalnya kemampuan penyimpanan dan pemrosesan), paradigma *IoT* berbasis *cloud(IoT-Cloud)* diperkenalkan dan menjadi pendekatan dalam menangani limitasi pada perangkat penunjang IoT [6].

Pada saat ini aplikasi dengan skala besar dan berperforma tinggi terutama dengan sistem

berbasis *cloud-native*, dikembangkan dengan arsitektur *Event-driven* berbasis microservices. Selain itu, penggunaan arsitektur berbasis *Event-driven* memungkinkan memproses data dalam waktu nyata [7]. Arsitektur *Event-driven* menggunakan *message broker* sebagai media komunikasi antar komponen dalam suatu sistem. Pada konteks IoT *message broker* digunakan dalam berbagai kasus salah satunya yaitu pada sistem notifikasi atau *Event notification* [8]–[10]. Dalam lingkungan IoT, banyak perangkat dan sensor mengumpulkan data yang penting, maka dari itu aplikasi atatu layanan *data processor* digunakan untuk memproses data dari *message broker* misalnya untuk di proses dan disimpan ke database, sistem notifikasi, email dan proses lainya [11]–[13].

Sistem *Event-Driven* seperti IoT mengandalkan pertukaran informasi secara *real-time* dan efisien antara berbagai perangkat dan *cloud*, yang sering kali membutuhkan layanan *data processor* yang optimal agar para pemangku kepentingan tetap mendapatkan informasi. *Message broker* biasanya digunakan untuk memfasilitasi komunikasi dalam sistem Event-Driven, tetapi ada berberapa permasalahan. Penggunaan *Message Broker* terkadang bisa jadi sulit, karena kompleksitas yang terkait dengan penanganan data, skalabilitas, dan manajemen sumber daya. Salah satu tantangan yang sering muncul dalam sistem berbasis IoT adalah pengelolaan antrian *(queue)* atau pesan *(messsage)* dalam *message broker*. Antrian *(Queue)* ini sering kali menumpuk sejumlah pesan yang cukup besar, yang menyebabkan inefisiensi dan potensi kemacetan data atau pesan yang berada pada *message broker*. *Message* dapat tidak diproses karena keterbatasan sumber daya yang tersedia. Situasi ini dapat memiliki konsekuensi yang luas, termasuk pemberitahuan yang tertunda, kehilangan data, dan, dalam kasus yang ekstrim, sistem memasuki kondisi pemblokiran atau non-pemrosesan [14].

Untuk mengatasi tantangan ini, penulis mengusulkan solusi inovatif yang bertujuan untuk mengoptimalkan kinerja layanan pemrosesan data dalam sistem berbasis IoT. Solusi ini memanfaatkan *Horizontal Pod Autoscaler* (HPA) pada *Kubernetes* yang merupakan platform orkestrasi kontainer merupakan salah satu teknologi *cloud*-native yang mendukung *High Availability (HA)* dan Elastisitas [14]. Penilitian kali ini penulis menggunakan *Kubernetes Event-Driven Autoscaler* (*KEDA*) untuk memastikan antrean pesan dikelola dan diproses secara efisien. Dengan demikian, penulis bertujuan untuk meningkatkan keandalan dan daya tanggap sistem *IoT* sekaligus memitigasi masalah yang terkait dengan antrean pesan yang besar dan tidak diproses serta keterbatasan sumber daya. Solusi ini menyesuaikan alokasi sumber daya secara dinamis berdasarkan beban sistem, sehingga memungkinkan pemrosesan pesan yang efisien dan mencegah sistem memasuki kondisi pemblokiran atau non-pemrosesan.

Sebelumnya sudah dilakukan penelitian terkait dengan *Horizontal Pod Autoscaler* (HPA) dengan membandingkan *metric* terbaik dalam melakukan *autoscaling* menggunakan HPA yaitu dengan membandingkan *metric Kubernetes* Resource *Metric* (KRM) dan Prometheus Custom *Metric* (*PCM*) dan mengevaluasi perilaku HPA berdasarkan kedua *metric* tersebut [15]. Penelitian selanjutnya membahas mengenai pengenalan HPA pada *single-board computer* untuk nantinya dapat diimplementasikan pada IoT [14]. Kedua penelitian diatas masih membahas HPA dengan *metric* yang diambil dari CPU dan Memory hal ini menjadi masalah pada kasus tertentu terutama dalam hal responsivitas dikarenakan HPA hanya

scaling berdasarkan kedua *metric* tersebut sedangkan dalam teknologi IoT banyak faktor yang memungkinkan suatu sistem terjadi lambat atau ter-blok seperti penumpukan *message* pada *message* queue, maka dari itu penelitian kali penulis mengimplementasikan HPA mennggunakan *metric* yang lebih luas menggunakan *Kubernetes Event-Driven* Autosclaer (*KEDA*) yang *scaling* suatu *Pod* dengan *metric* yang diambil dari *message broker*.

1.2. Perumusan Masalah

Rumusan masalah dari tugas akhir ini diantaranya:

1) Bagaimana perilaku *HPA Kubernetes* dengan menggunakan *Kubernetes Event-Driven Autoscaler* ?
2) Bagaimana menerapkan *Kubernetes* sebagai platform manajemen kontainer untuk sistem *Data processor* dalam meningkatkan kemampuan pemrosesan?

1.3. Batasan Masalah

Batasan masalah pada tugas akhir kali ini yaitu:

1) Implementasi dilakukan pada sistem *data processor* berbasis *cloud* dan *Kubernetes*.
2) *Message broker* yang digunakan adalah *Apache Kafka*
3) Menggunakan *Kubernetes Event-driven autoscaler* sebagai scaler HPA
4) Pengujian berfokus pada skalabilitas *data processor* dan *Message* rate
5) Metode autoscaling menggunakan metrik yang diambil dari *KEDA (Kubernetes Event-Driven Autoscaler)* dengan rata-rata metrik sebagai ambang batas autoscaling (*Average-Based Metric*) dan *PCM (Prometheus Custom Metric)* dengan jumlah total metrik sebagai ambang batas autoscaling *(Count Based Metric)*.

1.4. Tujuan

Penelitian ini bertujuan untuk mengevaluasi perilaku *HPA* menggunakan *KEDA (Kubernetes Event-Driven Autoscaler)*. Penelitian ini juga bertujuan untuk meningkatkan kemampuan *data processor* untuk menangani peningkatan beban kerja secara dinamis dengan menyesuaikan alokasi sumber daya komputasi berdasarkan banyaknya data yang masuk.