Bab I
Pendahuluan

1.1. Latar Belakang

Tidak tersedianya sebuah sistem yang disebabkan oleh *Distributed Denial of Service* (DDoS) menjadi ancaman utama bagi sebuah industry TI yang berjalan [1]. DDoS adalah serangan berbahaya yang dapat menghabiskan sumber daya server (*Bandwidth*, Memori, CPU, dll.) dan dapat menyebabkan sebuah server mati dan sebuah sistem tidak tesedia bagi user [5], dengan tidak tersedianya sebuah layanan bagi pengguna tentu saja bisa membuat bisnis TI tidak berjalan dan menimbulkan kerugian. Masalah ketersediaan *server* dan aplikasi sangat penting bagi pengguna. Ketersediaan *server* sangat berhubungan erat dengan kualitas yang ditawarkan oleh sebuah aplikasi, karena ketika sebuah *server* dan aplikasi tidak dapat melayani suatu permintaan, maka layanan tersebut dapat dikatakan kurang baik, ditambah lagi, ketika sebuah *server* atau sebuah aplikasi mati, layanan secara otomatis akan berhenti dan perlu dilakukan tindakan manual bagi tim operaion untuk menyalakannya kembali. Salah satu platform yang dapat digunakan untuk mendukung ketersediaan tinggi sebuah server dan aplikasi adalah Kubernetes[4]

Containerization adalah proses membundel sebuah aplikasi bersama dengan semua file konfigurasi, library, dan dependensi yang diperlukan agar aplikasi dapat berjalan secara efisien, sebuah bandel tersebut disebut *container* [6]. Sebuah *container* adalah sebuah wadah tertutup untuk perangkat lunak yang didalamnya memiliki semua files dan *libraries* yang dibutuhkan untuk menjalankan aplikasi [7], teknologi *container* yang populer saat ini adalah Docker [6].

Kubernetes adalah sistem manajemen *container* atau yang biasa disebut *container orchestrator* yang bisa melakukan deployment aplikasi di beberapa server. kubernetes menggunakan teknologi docker untuk menjalankan, melakukan penskalaan *container* di dalam cluster. Kubernetes dapat membantu menghindari masalah seperti *server downtime*, kerusakan fisik, kegagalan sistem operasi, dll. Kubernetes dapat memastikan keandalan dan ketersediaan sebuah aplikasi [4].

Pada penelitian ini penulis menganalisis sistem yang dibangun pada *cloud computing* yaitu *container orchestration* Kubernetes dari segi skalabilitas dan performa. Parameter perbandingan Kubernetes adalah besarnya serangan DDoS untuk skalabilitas, penggunaan sumber daya, *response time*, dan juga *availability test* dalam menjaga ketersediaan sebuah aplikasi. Untuk lebih memanfaatkan fitur yang tersedia pada Kubernetes, disini penulis menggunakan Kubernetes yang berbasis pada layanan *cloud* yaitu Amazon EKS (Elastic Kubernetes Service).

1.2. Perumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, terdapat beberapa rumusan masalah, yaitu:

1. Bagaimana mengimplementasi Kubernetes pada layanan *cloud computing* Amazon Web Services (AWS)?
2. Bagaimana menjamin dan menganalisa ketersediaan aplikasi yang berada dibawah serangan DDoS menggunakan Amazon EKS?
3. Bagaimana membangun infrastruktur yang mempunyai *high availability* dan *scalable*?

1.3. Tujuan

Tujuan yang ingin dicapai pada eksperimen yang dilakukan, yaitu:

1. Mengimplementasi Kubernetes pada layanan *cloud computing* AWS.
2. Menganalisis dan menjamin ketersediaan aplikasi yang berada dibawah serangan DDoS menggunakan Amazon EKS.
3. Membangun infrastruktur yang mempunyai *high availability* dan *scalable*.

1.4. Batasan Masalah

Batasan dari permasalahan yang ada pada penelitian ini, yaitu:

1. *Container Orchestration* yang digunakan adalah Kubernetes khususnya yang berada di cloud provider Amazon yaitu Amazon EKS.

2. Layanan cloud yang digunakan adalah AWS (Amazon Web Services)
3. Layanan di dalam AWS (Amazon Web Services) yang digunakan adalah:
    a) Virtual Private Cloud
    b) Internet Gateway
    c) Subnet
    d) Security Group
    e) Elastic Load Balancer
    f) Auto Scaling Group
    g) Target Group
    h) Elastic Kubernetes Services
    i) Identity Access Management
    j) Elastic Compute Engine
    k) Docker image yang digunakan disimpan di dalam docker hub.
4. Analisa hasil simulasi terbatas pada penggunaan CPU, penggunaan memori, dan *response time*.
5. Serangan DDoS dibuat menggunakan Low Orbit Ion Cannon (LOIC) menggunakan metode HTTP Flood yang dilakukan secara bertahap mulai dari 5.000, 10.000, 15.000, 30.000 serangan ke aplikasi, yang mana masing-masing serangan dijalankan 10 kali.
6. *Scaling* yang dilakukan memiliki batas maksimal yaitu sebanyak 5 server.
7. Serangan DDoS diciptakan menggunakan 1 buah laptop menggunakan tools LOIC.