UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

Escalonamento em aplicações na plataforma Kubernetes utilizando algoritmos de previsão temporal

PROPOSTA DE TRABALHO DE GRADUAÇÃO

Aluno: André Luís Damázio de Sales Júnior (aldsj@cin.ufpe.br)
Orientador: Adriano Augusto de Moraes Sarmento (aams@cin.ufpe.br)
Área: Aprendizagem de máquina, Computação em nuvem

14 de Junho de 2021.

Resumo

Em pouco mais de uma década, a nuvem consolidou-se como uma das plataformas mais importantes da computação moderna. A escalabilidade é uma característica fundamental para essa consolidação, visto que esta oferece capacidade de crescimento sob demanda para diversos serviços em diferentes estágios de desenvolvimento. Para obter-se escalabilidade, faz-se necessário entender em que momento escalar recursos computacionais. Nesse projeto, é feita a implementação e avaliação do uso de soluções baseadas em análise de séries temporais para previsão de carga em ambiente Kubernetes, e baseado nessas previsões um sistema de auto escalonamento é proposto e avaliado. Além disso, é feito um comparativo com a solução nativa da plataforma.

Introdução

Conteinerização é uma tecnologia utilizada para virtualização de aplicações de maneira leve e que vem resultando em avanços na forma como aplicações em nuvem estão sendo implantadas [1]. A utilização dessa tecnologia traz facilidades no ciclo de desenvolvimento e de implantação da aplicação. As principais vantagens incluem, portabilidade, consistência, e aceleração no ciclo de desenvolvimento. Em contraponto a utilização de máquinas virtuais que necessitam que todo o sistema operacional esteja incluso, a conteinerização permite a virtualização de aplicativos autocontidos e suas dependências, que compartilham recursos de um sistema operacional em comum, o que torna essas aplicações mais leves e portáteis. Como orquestrar a criação e implantação dessas aplicações em ambientes em nuvem vem sendo um dos grandes desafios da área, para tal, ferramentas como Docker e Kubernetes vem sendo utilizadas para resolver esse problema, de forma escalável e performática.

A partir da demanda por ferramentas capazes de prover facilidade no desenvolvimento de aplicações conteinerizadas, surgiu o Docker, que consiste em um mecanismo de conteinerização de aplicações de código aberto [2], simples e prático para desenvolver e implantar aplicações, onde sua popularização advém principalmente da facilidade de implantação e versionamento de aplicações. Em um ambiente de contêiner onde as aplicações são executadas, o Docker adiciona uma camada extra de implantação, que facilita a testagem antes do sistema ser implantado em produção. Por ser uma ferramenta de conteinerização, o mesmo possibilita o empacotamento de uma aplicação juntamente com todas as suas dependências em contêineres , tornando o ambiente inteiro portável para qualquer outro Host que contenha o Docker instalado. Esses contêineres então são executados compartilhando de forma isolada apenas o kernel do sistema operacional em comum, dessa forma é possível manter o isolamento entre os serviços que estão sendo executados em um mesmo ambiente [3].

Dado que as aplicações nesses contêineres estão isoladas e autocontidas, surge a necessidade de gerenciamento das mesmas. O gerenciamento do ciclo de vida destes contêineres é feito por aplicações conhecidas como orquestradores. Esses sistemas são capazes de lidar com centenas de milhares de aplicações distribuídas em milhares de máquinas de maneira autônoma. E para que isso seja possível, é necessário que esses sistemas sejam capazes de prover escalabilidade, resiliência a falhas e disponibilidade, utilização eficiente de recursos entre outros [4]. O Kubernetes surge como uma plataforma de orquestração de contêineres de código aberto que define um conjunto de ferramentas que provêem um mecanismo para implantação, manutenibilidade, escalabilidade e monitoramento de aplicações em contêineres. Este busca abstrair a complexidade de orquestração**,** enquanto gerencia a disponibilidade dos contêineres ao qual está encarregado [5].

Escalabilidade

Computação em nuvem oferece inúmeros benefícios para o desenvolvimento de produtos de software, talvez sendo o maior deles a habilidade de escalar o seu ambiente sob-demanda [7]. Escalabilidade é o termo utilizado para definir a capacidade que um software tem em lidar com variações de carga, buscando mitigar o impacto dessas variações na experiência do usuário, no caso de aplicações web [6]. Em termos de computação em nuvem, pode-se definir duas formas comuns de escalabilidade, vertical e horizontal.

Ao lidarmos com aumentos no número de requisições em uma aplicação web, por exemplo, o aumento no tempo de resposta pode vir a acontecer devido a incapacidade que os recursos de hardware da presente infraestrutura têm para processar tais requisições. O aumento desses recursos (cpu, memória, armazenamento, etc) é conhecido com escalabilidade vertical. Uma das desvantagens de escalar-se verticalmente acontece pois existem limitações físicas impostas pelo estado da arte dos computadores atuais, um outro problema decorrente é o tempo de inatividade necessário para execução da mudança em tempo de execução.

Outra alternativa para lidar com alterações de carga é a escalabilidade horizontal. Nessa forma, a infraestrutura como um todo é replicada e interconectada, atuando como um único sistema. Sendo necessário que as aplicações sejam independentes entre si, de forma que possam ser requisitadas separadamente já que existem réplicas de infraestrutura [7].

O auto escalonamento é uma funcionalidade presente na maioria dos provedores de nuvem. Estes são capazes de escalonar recursos baseando-se em métricas pré-definidas pelo usuário. Uma das ferramentas de escalonamento presentes no kubernetes é o *horizontal pod autoscaler* (hpa) [8]. Essa ferramenta escalona o número de pods baseado na utilização de CPU ou em métricas customizadas. O funcionamento do hpa é regido pela seguinte equação:

$$numeroReplicasDesejado = teto\left[numeroReplicas\ x \frac{valorMetricaAtual}{valorMetricaDesejado}\right]$$

A inicialização de um novo Pod pode ser dividido em quatro fases, sendo estas, disparo do hpa para expansão, cálculo do número de réplicas necessárias, agendamento de novos pods, criação e inicialização de um novo pod [9]. Na abordagem padrão do hpa, um grande número de requisições se acumula durante o tempo de escalonamento composto pelas quatro fases, o que leva a um fenômeno de enfileiramento de requisições em espera. Esse enfileiramento aumenta o tempo de resposta do usuário, degradando a qualidade do serviço.

Objetivos

Devido a resultados encontrados em trabalhos similares que buscaram resolver o problema de tempo de resposta durante o escalonamento da fase de expansão. Esse projeto busca estudar e entender melhor a estratégia de auto-escalonamento do Kubernetes, e alternativamente propor a utilização de uma estratégia de predição baseada em análise de séries temporais para resolver o problema supracitado. Para tal, uma aplicação de auto escalonamento Kubernetes será desenvolvida.

Como objetivos específicos do desenvolvimento, tem-se:

- O desenvolvimento de uma aplicação de leitura de métricas Kubernetes capaz de escalar Pods de um serviço qualquer, de maneira autônoma.
- Estudo e identificação de algoritmos de previsão de séries temporais que se adequem ao problema de escalabilidade no tempo de expansão.
- Definição de métricas relevantes para serem utilizadas na resolução do problema.
- Comparativo qualitativo entre as soluções desenvolvidas e a forma padrão do hpa presente.
- Investigação de soluções de benchmark para serviços web com integração Kubernetes.

Metodologia

Como metodologia para o desenvolvimento do projeto, inicialmente será feito o estudo das soluções já presentes, assim com o estudo aprofundado do funcionamento do hpa nativo, possível, visto que a tecnologia é de código aberto.

Após o levantamento das soluções e dos resultados, se faz necessário entender quais soluções adequam-se melhor ao problema. A partir daí poderemos iniciar o desenvolvimento da aplicação Kubernetes, para testes iniciais de leitura de métricas genéricas e escalonamento, e em seguida definir as métricas relevantes para o trabalho.

Tendo o sistema de escalonamento com funcionalidades básicas implementado, pode-se então implementar os algoritmos encontrados e validar o funcionamento. Após a etapa anterior, uma busca será feita em trabalhos similares à procura de soluções de Benchmarking de aplicações web quanto a escalabilidade, preferencialmente que tenham sido utilizadas em aplicações Kubernetes.

A partir daí, um comparativo será feito entre os resultados encontrados na utilização dos algoritmos e a solução padrão presente no hpa do Kubernetes.

Durante toda a elaboração do trabalho serão feitas reuniões remotas com o orientador visando definição de quaisquer mudanças necessárias no escopo do trabalho, assim como acompanhamento para eventuais dúvidas técnicas, na escrita e apresentação final do trabalho.

Cronograma

	Período							
Atividade	Junho	Julho				Agosto		
Revisão bibliográfica	X							
Implementação de aplicação base	X	X	X					
Estudo de algoritmos			X					
Implementação de algoritmos			X	X				
Testes de performance				X	X			
Escrita do TG					X	X		
Preparação da apresentação							X	X

Referências

[1] Pahl, Claus & Brogi, Antonio & Soldani, Jacopo & Jamshidi, Pooyan. (2017). Cloud Container Technologies: a State-of-the-Art Review. IEEE Transactions on Cloud Computing. PP. 1-1. 10.1109/TCC.2017.2702586.

[2] Bashari Rad, Babak & Bhatti, Harrison & Ahmadi, Mohammad. (2017). An Introduction to Docker and Analysis of its Performance. IJCSNS International Journal of Computer Science and Network Security. 173. 8.

[3]Paul Rubens. What are containers and why do you need them? 2020. Disponivel em: <https://www.cio.com/article/2924995/what-are-containers-and-why-do-you-need-them.html>

[4] M. A. Rodriguez and R. Buyya, "Container-based Cluster Orchestration Systems: A Taxonomy and Future Directions," Software: Practice and Experience, pp. 698--719, 2019.

[5] Vayghan, Leila & Saied, Mohamed & Toeroe, Maria & Khendek, Ferhat. (2019). Kubernetes as an Availability Manager for Microservice Applications.

[6] Martin Kleppmann. *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems Paperback* – Illustrated, April 11, 2017.

[7]Sarah Vonnegut. Scalability in the Cloud: How Organizations Win with the Cloud 2020. Disponivel em : <https://www.stratoscale.com/blog/cloud/scalability-cloud-organizations-win-cloud/>

[8] KUBERNETES. Horizontal Pod Autoscaler 2020. Disponível em: <https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale/>

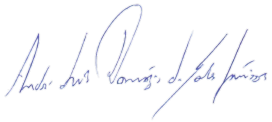
[9] Zhao, Anqi & Huang, Qiang & Huang, Yiting & Zou, Lin & Chen, Zhengxi & Song, Jianghang. (2019). Research on Resource Prediction Model Based on Kubernetes Container Auto-scaling Technology. IOP Conference Series: Materials Science and Engineering. 569. 052092. 10.1088/1757-899X/569/5/052092.

Possíveis Avaliadores

Prof. Germano Vasconcelos

Assinaturas

Recife, 14 de Junho de 2021



André Luís Damázio de Sales Júnior

(Aluno)

Adriano Augusto de Moraes Sarmento

(Orientador)