Análise de Fluxos de Rede em Microsserviços Gerenciados com Kubernetes [1]

Matheus Baltasar de Carvalho Cercena[2], Maurício Aronne Pillon[3].

[1] Vinculado ao projeto "Impacto energético em nuvens IaaS frente a aplicação de técnicas de escalonamento e de virtualização."
[2] Acadêmico (a) do Curso de Ciência da Computação – CCT – Bolsista PROBIC
[3] Orientador, Departamento de Ciência da Computação – CCT – mauricio.pillon@udesc.br

Microsserviço se trata de uma nova abordagem à modularização de aplicações distribuídas, cuja característica é dividir funcionalidades de um sistema complexo em módulos menores, a fim de facilitar o desenvolvimento, implementação e entendimento dessa aplicação. No que tange a implementação, o grau de modularização impacta na escalabilidade do microsserviço e no desempenho. O desenvolvimento e modelagem do sistema são afetados na forma de organização de equipes de projetos, coordenação de pessoal e habilidades específicas de cada módulo. Microsserviços prezam pela independência conceitual dos módulos, mas não fogem da comunicação intermódulos por meio de transações de redes.

Conteinerizar os módulos independentes de microsserviços é um caminho de concretizar a implementação de uma aplicação microsserviços com ferramentas de gerenciamento escaláveis e flexíveis. O presente trabalho escolheu a técnica de conteinerização *Docker* e a plataforma de gerenciamento automático de contêineres *Kubernetes*. O *Docker* é um sistema capaz de lançar contêineres isolados dentro de uma rede, em que dentro desses componentes pode-se incluir partes de aplicações, de forma com que um usuário que deseja manipulá-la apenas precise estar conectado nessa mesma rede, sem a necessidade de passar por instalações ou configurações individuais. O *Kubernetes* possui o papel de orquestrar esses mesmos contêineres baseados em *Docker*, fazendo com que toda essa rede constitua um ecossistema completamente modularizado.

A arquitetura do *Kubernetes* se baseia em máquinas representadas como nós organizadas na forma de mestre/trabalhadores. Ao mestre, cabe o trabalho de atribuir e coordenar os serviços e, ao trabalhador, o processamento efetivos dos módulos da aplicação. Pode-se atribuir a cada nó trabalhador um ou mais serviços, com as mesmas funções ou não. Esses nós trabalhadores são generalizados unicamente para o nó principal através de uma camada de conexão virtual, que trata todos esses nós como uma única máquina. Nesses nós, estão presentes o menor componente dentro do *Kubernetes* denominado de *pod.* O *pod* é uma abstração da arquitetura que serve para delimitar o espaço de alocação de um ou mais contêineres que estão dispostos em um mesmo hospedeiro, podendo assim compor uma parte da aplicação por inteiro com seus respectivos módulos.

Para fins de manipulação e análise a respeito de como uma aplicação completa se comporta quando imerso em tal arquitetura, o foco deste trabalho, em um primeiro momento, se baseou em uma pesquisa estendida sobre aplicações já existentes voltadas para testagem que foram implementadas com o conceito de microsserviço desenvolvidas para *Docker* e gerenciadas por *Kubernetes*. Uma vez escolhida a aplicação, a estratégia consistiu em replicar a mesma nas máquinas virtuais disponibilizadas pela plataforma em nuvem do laboratório de pesquisa *LabP2D*, de forma com que partes dessa aplicação estejam distribuídas entre as máquinas dispostas em uma mesma rede. A partir desse ponto, foi possível analisar o fluxo de pacotes de redes que passam entre os módulos da aplicação e traçar seus caminhos dentro da arquitetura.

O microsserviço escolhido para referência se trata do *SockShop,* um projeto que simula uma aplicação *web* voltada para venda *online* de meias. O objetivo dessa aplicação é servir como plataforma base para geração de tráfego, simulando usuários que acessam conteúdos do *site*, sendo possível a visualização de um *benchmark* de recursos computacionais consumidos com os parâmetros dessa carga em uma ferramenta de monitoramento denominada *Prometheus*, que já está vinculada como um módulo a parte opcional desse projeto. Essa carga pode ser gerada com outra ferramenta também associada chamada de *Locust*, que é uma espécie de *library* do *Python* usada para escrever *scripts* que ditam o comportamento de usuários artificiais dentro da plataforma, que podem ser lançadas como mais um componente dentro da rede da aplicação.

A replicação nas máquinas da pesquisa necessitou de um passo adicional para garantir o aspecto distribuído, o qual envolvia uma modificação sutil nos arquivos de configuração *YAML* das aplicações usadas para lançá-la no *Kubernetes*, que basicamente precisou especificar diferentes nós a um conjunto de arquivos, assim não deixando o orquestrador lançar todos os módulos em somente um único nó. Com a aplicação executando, pode-se utilizar de uma ferramenta no sistema operacional usada para monitoramento de tráfego de pacotes em uma rede, o *tcpdump*, para checar a atividade dentro da interface de rede em que a aplicação foi alocada e traçar o caminho de alguns de seus módulos com base em *IPs* de destino e origem, contextualizado para a arquitetura do *Kubernetes*, como mostrado na figura 1.

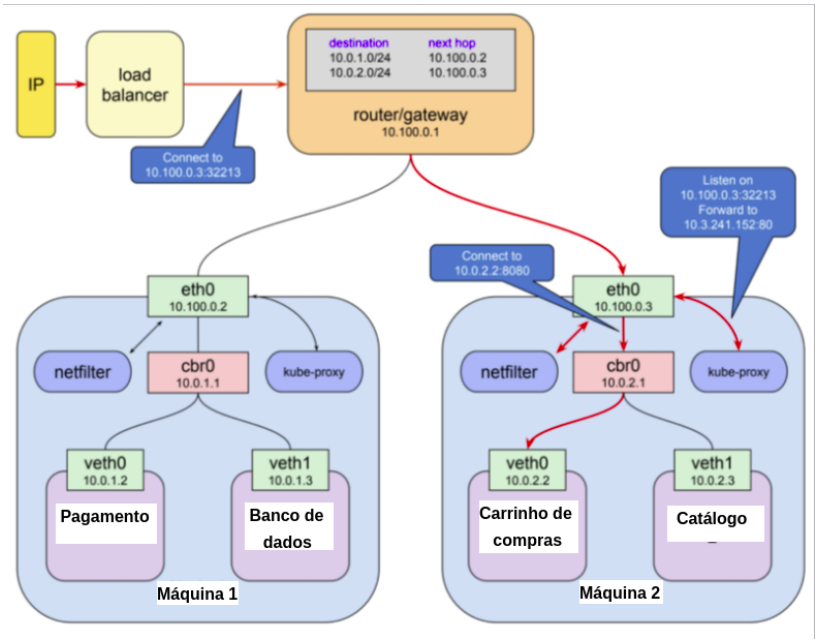

Figura 1. *Arquitetura de rede do Kubernetes exemplificada com o microsserviço SockShop*

A implantação da plataforma experimental e coleta de dados do fluxo foram concluídas com êxito. Como trabalho futuro, tem-se a definição de um plano de testes para o benchmark e a análise do comportamento da aplicação sob condições de saturação de recursos.

Palavras-chave: Microsserviços. Virtualização. Conteinerização. Docker. Kubernetes.