Service Mesh para Gerenciamento de Configuração

Pablo Yuri S. Souza[1], Gessica Franciéle M. Azevedo[1], Jordi Pujol Ricarte[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{gessica.azevedo,jpricarte,pablo.souza}@inf.ufrgs.br

Abstract. *This article presents the setup of a local environment using Minikube to establish a Kubernetes cluster and the installation of Istio as a Service Mesh. It details the environment, covering the installation of kubectl and Istio, emphasizing installation profiles. It explores the implementation of the BookInfo application, from namespace labeling to ingress configuration, highlighting Service Mesh functionality in inter-service communication. Additionally, it showcases Istio's traffic shifting for controlled microservices version migration. This practical configuration is pertinent to real-world scenarios and contributes to the discussion on network configuration management.*

Resumo. *Este artigo apresenta a configuração de um ambiente local com Minikube, para estabelecer um cluster Kubernetes, e a instalação do Istio como Service Mesh. Detalha o ambiente, incluindo a instalação do kubectl e do Istio, com ênfase nos perfis de instalação. Explora a implementação do aplicativo BookInfo, desde a rotulagem do namespace até a configuração do ingress, evidenciando o funcionamento do Service Mesh na comunicação entre serviços. Além disso, ilustra o uso do traffic shifting do Istio para uma migração controlada de versões de microsserviços. Esta configuração, discutida de maneira prática, é relevante para cenários reais e integra um debate sobre gerenciamento da configuração de rede.*

1. Introdução

Este artigo apresenta uma investigação prática sobre a implementação de um Service Mesh para o gerenciamento de configuração, utilizando o Minikube para criar um cluster Kubernetes local e o Istio como a solução de Service Mesh, mais especificamente, iremos replicar o processo de roteamento de requisições de um serviço de avaliações literárias.

A tecnologia de service mesh serve para controlar diferentes componentes de uma aplicação que compartilham dados entre si. Ao contrário de outros sistemas de gerenciamento de comunicação, essa tecnologia implementa uma infraestrutura dedicada a um serviço específico, algo que facilita o gerenciamento da rede de uma aplicação baseada em microsserviços.

Inicialmente, iremos configurar o nosso ambiente Kubernetes, como descrito na seção 2. Na seção 3, iremos abordar a configuração de roteamento das requisições seguindo o tutorial disponibilizado pelo Istio. Por fim, discutiremos como podemos expandir as ideias vistas para aplicações reais, criando cenários onde o roteamento de requisições descrito aqui resolverá problemas vistos no dia a dia de um serviço.

2. Configuração do Ambiente

2.1. Utilização do Minikube para criar um cluster Kubernetes local

No primeiro passo, realizou-se o download das ferramentas necessárias para a execução da nossa aplicação. São elas o Docker para a criação de clusters, o Kubernetes para o gerenciamento de containers e o kubectl, que permite interagir com os containers utilizando o terminal. O processo de instalação foi baseado no descrito na especificação, utilizando os gerenciadores de pacotes dos sistemas para facilitar o processo.

Completando a instalação, utilizamos o Minikube para criar um cluster Kubernetes local. utilizando o comando start do minikube, conseguimos configurar o driver (docker), os cores e memória utilizadas de acordo com a capacidade da máquina.

2.2. Instalação e configuração do Istio como Service Mesh.

Para a configuração do service mesh, instalamos o Istio, assim como os arquivos de exemplo necessários para a execução do tutorial disponibilizado.Após a conclusão bem-sucedida da instalação, fizemos a instalação efetiva do Istio usando o comando istioctl install --set profile=demo. Esta configuração específica escolheu o perfil de instalação de demonstração, incorporando recursos padrão adequados para ambientes de teste.

3. Implementação de Exemplos

3.1. Implementação de microsserviços de exemplo

Implementamos o aplicativo BookInfo de exemplo do Istio. As instruções detalhadas abaixo fornecem uma visão geral do processo, desde a rotulagem do namespace até a verificação do acesso à aplicação.

No primeiro passo, o namespace padrão foi rotulado para permitir a injeção automática do Istio nos pods. O comando utilizado foi “kubectl label namespace default istio-injection=enabled.” Isso configura o ambiente para que o Istio injete automaticamente sidecars nos pods do namespace padrão.

Em seguida, foram aplicados os recursos do BookInfo utilizando os arquivos de configuração “platform/kube/bookinfo.yaml” e “networking/bookinfo-gateway.yaml” que apontavam para os arquivos YAML relativos à configuração da aplicação e do gateway da mesma. O resultado disso é o deploy da aplicação com arquitetura da Figura 2.

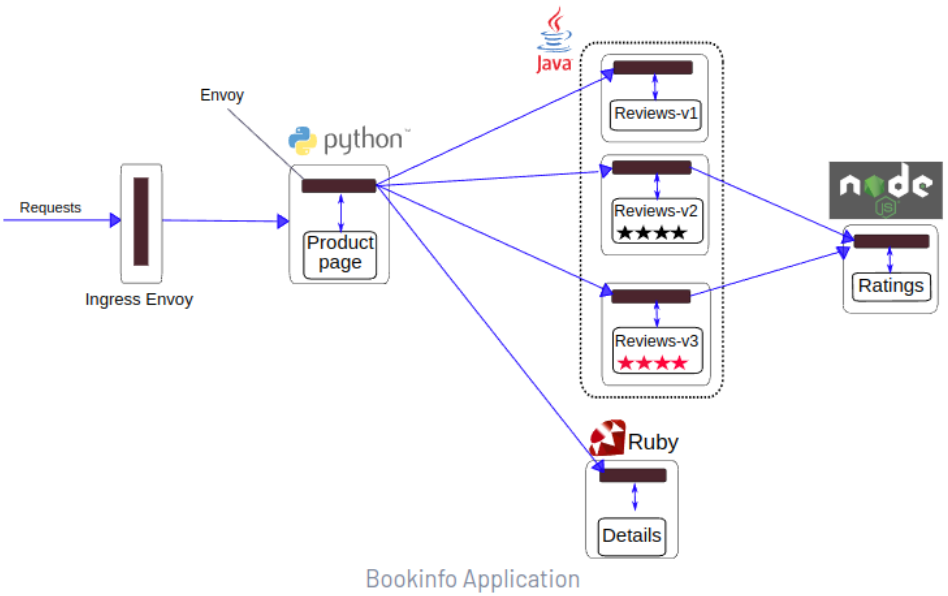

Figura 1. Arquitetura do projeto BookInfo

A configuração do ingress para o Istio foi o terceiro passo, determinamos o IP e as portas do ingress, executando o comando “minikube tunnel” e exportando as informações necessárias para o acesso ao ingress. Essas variáveis foram essenciais para configurar o ambiente Minikube para acessar o Istio Ingress Gateway.

Finalmente, com a configuração do gateway pronta obtivemos acesso à aplicação verificando ao direcionar o tráfego para a página de produtos do BookInfo usando o endereço de loopback. Este passo assegurou que a instalação e configuração do Istio no Minikube fossem bem-sucedidas.

3.2. Implementação das regras de roteamento de requisição em um Service Mesh.

Com a nossa aplicação rodando corretamente, aplicamos as regras de roteamento do Istio para todos os serviços do BookInfo usando o arquivo “networking/destination-rule-all.yaml” efetivamente conectando todos microsserviços e obtendo o resultado presente na Figura 3.

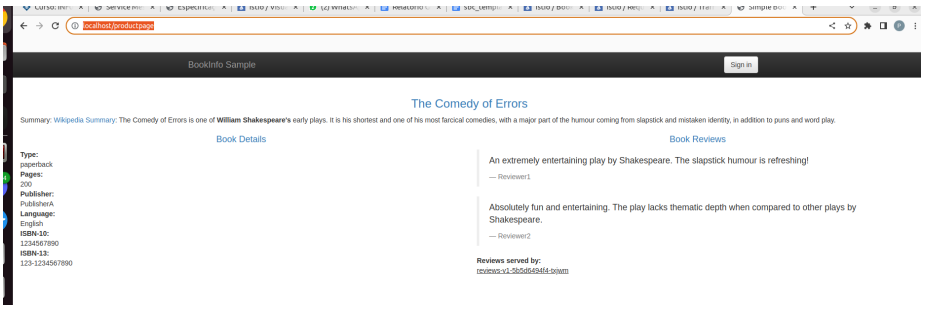

Figura 2. Página de Product Page rodando localmente com microsserviço Reviews-v1

Com nosso Service Mesh funcionando corretamente, podemos criar regras de roteamento utilizando os chamados “Virtual Services”. Esse modelo permite especificar regras para um determinado host.

No tutorial que seguimos nesse trabalho, geramos a seguinte regra de roteamento: Caso o usuário seja “Jason”, ele será redirecionado diretamente para a v2 do nosso aplicativo conforme demonstrado na Figura 4. Caso contrário, ele será redirecionado para a v1 ou v3 do mesmo, com 50% de chance de ir para cada uma dessas conforme demonstrado na Figura 3.

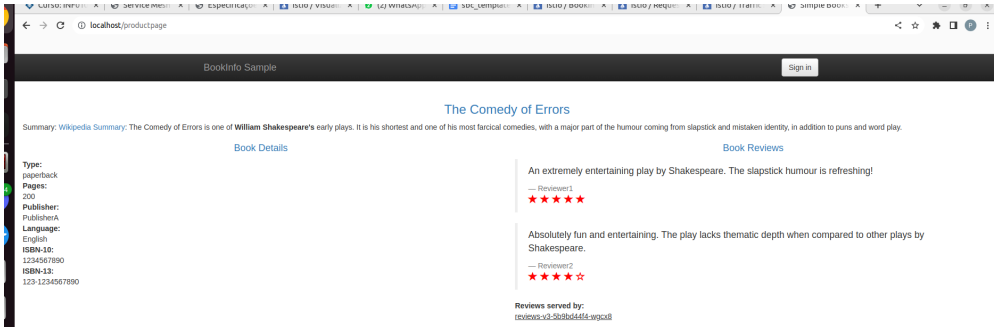

Figura 3. Página de Product Page rodando localmente com microsserviço Reviews-v3

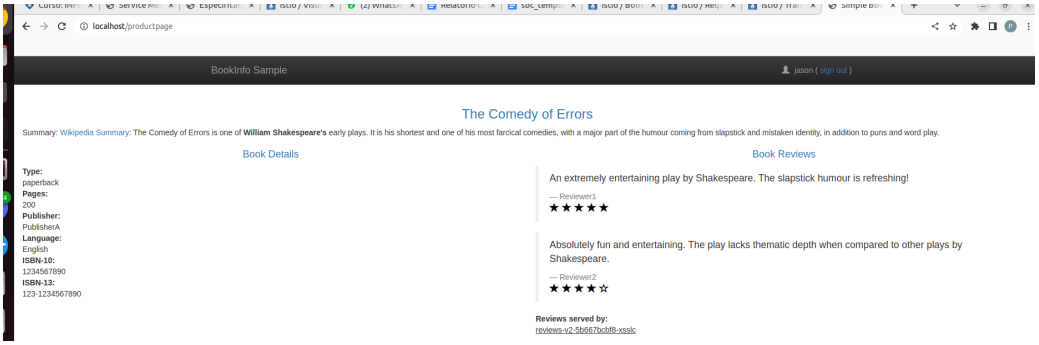

Figura 4. Página de Product Page rodando localmente com microsserviço Reviews-v2

4. Conclusões:

O processo descrito acima traz um exemplo simples de como podemos configurar nossa rede para distribuir os acessos a um serviço. Podemos expandir a ideia aqui descrita para fazer roteamentos mais complexos, a depender da demanda que chegue até o gerente do projeto.

Um exemplo de como podemos expandir esse modelo seria caso o nosso serviço de reviews fizesse uma nova versão para um cliente específico (por exemplo uma

livraria). Caberia a equipe de desenvolvimento desenvolver uma versão específica do aplicativo para essa livraria, enquanto a equipe de redes poderia usar as regras aqui dispostas para rotear todos os acessos de logins dessa livraria para essa versão. Outra opção seria definir um prefixo no endereço de acesso, que teria o mesmo objetivo de usar uma tag no cabeçalho.

Outra possível aplicação seria para, em conjunto com uma equipe de testes, implementar um teste A/B de uma nova versão da plataforma. O gerente dessa plataforma poderia utilizar uma tag específica para definir a que versão do sistema o usuário será encaminhado, de forma transparente ao cliente.

Essas são apenas algumas das diversas possibilidades que se abrem quando utilizamos um service mesh. Isso mostra como o uso dessa tecnologia pode ser útil para simplificar o gerenciamento de configuração de rede onde, com um arquivo simples e de fácil leitura, podemos criar regras robustas de roteamento.

References

Istio / Bookinfo Application,

https://istio.io/latest/docs/examples/bookinfo/#define-the-service-versions, December.

Istio / Request Routing,

https://istio.io/latest/docs/tasks/traffic-management/request-routing/, December.

Istio / Traffic Shifting,

https://istio.io/latest/docs/tasks/traffic-management/traffic-shifting/, December.