REDES VIRTUALIZADAS COM DOCKER

Genileide Gabriella de Carvalho Ferreira[1], Samyta Ramiza Vicente da Silva[1], Maria Aparecida de Oliveira Varela[1], Natan Paulo Fernandes[1], Victor Carvalho Galvão de Freitas[1], Diego da Silva Pereira[1]

[1]Laboratório de Pesquisa em Redes e Sistemas Computacionais (NOCS LAB), Instituto Federal do Rio Grande do Norte, Parnamirim, Brasil (gabriella.g@escolar.ifrn.edu.br)

Resumo: A pesquisa demonstra a criação de redes virtualizadas com Docker, permitindo a comunicação segura entre containers. Foram comparados os métodos manual e Docker Compose, sendo este último mais eficiente por automatizar a configuração e reduzir erros. Os resultados confirmaram o isolamento da rede e a comunicação entre containers por meio do DNS interno do Docker, destacando vantagens como leveza, portabilidade e otimização de recursos.

Palavras-chave: Docker; Virtualização; Redes; Containers; Docker Compose.

INTRODUÇÃO

Serão apresentados conceitos como comunicação entre containers, isolamento de serviços e os principais tipos de redes do Docker, além de uma breve demonstração prática de criação e configuração de redes virtuais. O trabalho também destaca vantagens como flexibilidade, organização, escalabilidade e otimização de recursos em ambientes computacionais.

O Docker é uma plataforma de código aberto que automatiza o deploy de aplicações em containers. Foi criado pela DotCloud, empresa especialista em PaaS. Tendo como diferencial uma plataforma de deploy de aplicações sendo executada em um ambiente virtualizado de container (TURNBULL, 2014).

Containers são uma tecnologia de virtualização que permite replicar um ambiente computacional isolado, apto a executar aplicações de maneira consistente. Onde cada um deles é uma unidade virtual capaz de reunir todos os recursos necessários para o funcionamento de um serviço ou processo, podendo ser, códigos, bibliotecas, dependências e recursos de desenvolvimento (ROMERO, 2018).

O objetivo é a criação de uma rede virtual simples para que os containers possam conversar entre si de forma isolada e segura. O isolamento é garantido pelos recursos do Linux chamados **Namespaces** (isolamento de processos e rede) e **Cgroups** (limitação de hardware). A conexão é feita através de um driver bridge que funciona como um switch virtual conectando os containers. A maior vantagem da rede própria é que o Docker ativa um DNS interno, permitindo o uso apenas do nome sem precisar do IP (GOMES, 2019).

MATERIAL E MÉTODOS

A metodologia utilizada nesta pesquisa teve como base a exploração da ferramenta Docker. O Docker é amplamente usado no gerenciamento de infraestrutura de aplicações de redes de computadores e temas correlatos. Seu manuseio é feito através do terminal de comando que permitiu a agilização dos processos de criação, manutenção e modificação dos serviços. Para o desenvolvimento, utilizaram-se o sistema operacional Windows 11 e a versão do Docker 29.4.3.

A criação da rede virtual foi feita utilizando-se de dois métodos, o manual e o Docker Compose, podendo haver um comparativo no nível de automação e organização da infraestrutura de ambos.

Na configuração manual, o primeiro passo foi a criação da rede, onde no terminal foi usado o comando: *docker network create rede-virtual*. A rede cria um ambiente de comunicação isolado. O segundo passo foi a criação da imagem, para isso foi preciso em uma pasta criar o arquivo Dockerfile com as instruções da imagem. E depois dentro da pasta onde está o arquivo executar o comando *docker image build -t minha-imagem .*, para construir a imagem. O último passo foi a criação dos 3 containers executando no terminal o comando *docker container run -d --name cont1 --net rede-virtual minha-imagem, docker container run -d --name cont2 --net rede-virtual minha-imagem* e *docker container run -d --name cont3 --net rede-virtual minha-imagem.* E por último foi feito o teste de

comunicação, fazendo o ping do container cont1 para o cont3 no terminal usando *docker container exec cont1 ping cont3*.

O segundo método foi utilizar o Docker Compose. O primeiro passo foi a criação, dentro de uma pasta, do arquivo docker-compose.yml com informações da imagem e dos containers. O próximo passo foi a criação da rede e dos containers executando no terminal, dentro da pasta do arquivo, o comando *docker-compose up -d*. E, por último, foi feito o teste de conexão executando o comando *docker-compose exec cont1 ping cont3*.

RESULTADOS E DISCUSSÃO

Na comunicação manual, ao usar o comando *docker network create,* foi possível observar o isolamento da rede virtual em relação à rede padrão Docker0 do host.

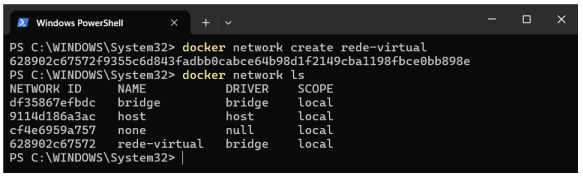

Figura 1. Criação e verificação de rede virtual no terminal.

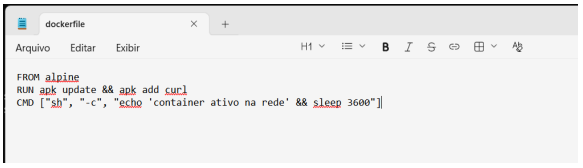

Figura 2. Criação da imagem no arquivo dockerfile.

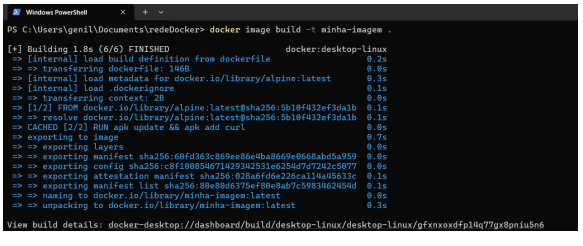

Figura 3. execução do comando build para construção da imagem.

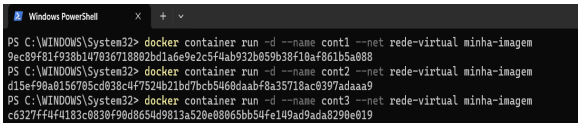

Figura 4. Criação dos 3 containers.

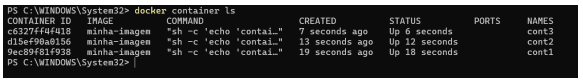

Figura 5. listagem dos containers ativos na rede.

Foi possível notar como a criação de uma rede utilizando-se do arquivo *docker-compose.yml,* simplificou o processo de subir múltiplos serviços de forma simultânea, fazendo com que erros de configuração manuais de rede fossem reduzidos.

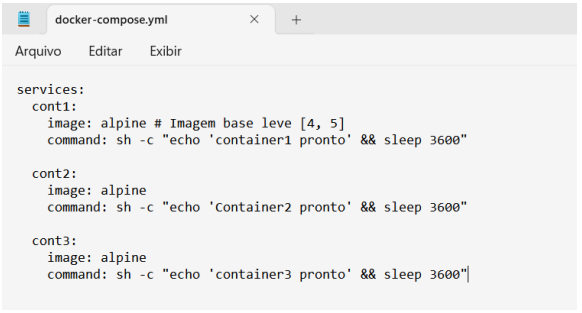

Figura 6. Definição da estrutura Docker compose.

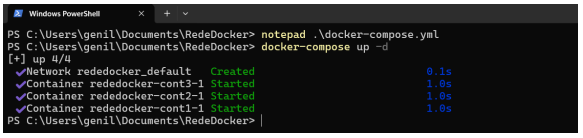

Figura 7. Execução automatizada.

Ao que diz respeito ao teste de ping por nome, foi obtido sucesso sem a necessidade de mapeamento do endereçamento ip manual, isso se deu pelo fato da utilização do DNS interno, característica tanto de redes manuais quanto no compose, que permite que os containers se comuniquem pelo nome.

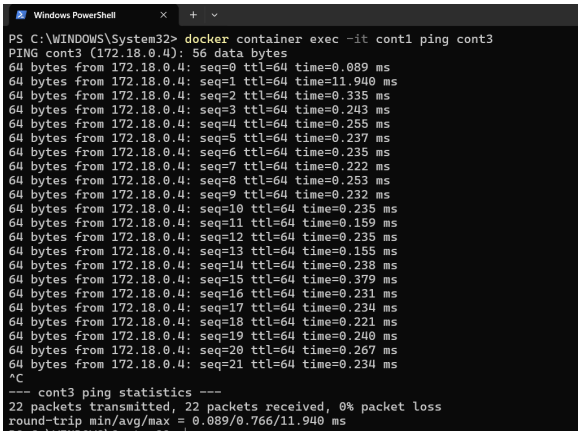

Figura 8. Teste de conectividade entre containers utilizando DNS interno.

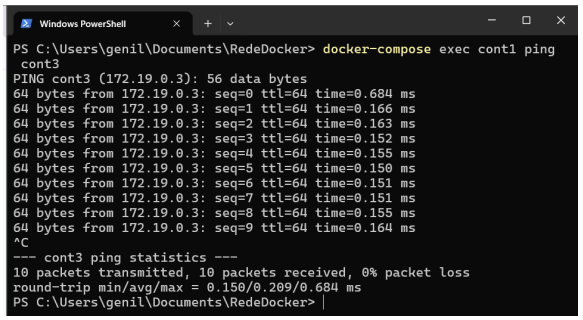

Figura 9. Teste de conectividade com método compose.

CONCLUSÃO

Por compartilhar o kernel do host, a virtualização por containers se torna mais leve do que as máquinas virtuais tradicionais. Além da utilização de redes customizadas com Docker Compose que garante portabilidade, facilidade de gerenciamento, permitindo que a aplicação seja replicada em diferentes ambientes de forma fiel.

Foi possível confirmar, como proposto na introdução, a criação de uma rede virtual simples para comunicação isolada e segura. A implementação demonstrou sucesso no isolamento através de *Namespaces* e na conectividade eficiente via driver *bridge*.

No comparativo entre os dois métodos, Manual e Docker Compose, foi possível evidenciar no método manual o isolamento em relação à rede padrão do host. Já o Docker Compose se mostrou superior na automação e organização, reduzindo erros de configuração humana ao subir múltiplos serviços simultaneamente.

As redes customizadas se mostraram vantajosas pela ativação do DNS interno, permitindo a comunicação entre os containers apenas pelo nome, simplificando o gerenciamento da infraestrutura.

Foi possível concluir que a leveza da virtualização por containers, em comparação com as máquinas virtuais tradicionais, traz mais portabilidade e a otimização de recursos necessários para ambientes de redes modernas.

AGRADECIMENTOS

Agradecemos ao professor Diego da Silva Pereira pelos conhecimentos compartilhados, pelo apoio e pelo incentivo durante o desenvolvimento deste trabalho. As orientações recebidas foram fundamentais para a realização desta pesquisa e contribuíram significativamente para o nosso aprendizado e crescimento acadêmico.

E ao NOCS Lab e ao IFRN Parnamirim pelo apoio no desenvolvimento da pesquisa.

REFERÊNCIAS

GOMES, Rafael. **Docker para desenvolvedores**. Leanpub, 2019.

ROMERO, Daniel. **Containers com Docker: Do desenvolvimento à produção**. Casa do Código, São Paulo, 2018.

TURNBULL, James. **The Docker Book**. Creative Commons, 2014.