

Um estudo de caso sobre a utilização de containers para aplicações em nuvem de alta disponibilidade

Leonardo B. Pelegrini¹, Marcos J. Vissoto Corino¹, Roger Sá da Silva¹

¹Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul – *Campus*
Avançado Veranópolis - Veranópolis - RS - Brasil

i32.leo@gmail.com

Resumo. Atualmente, plataformas de computação em nuvem oferecem serviços sob medida para negócios de qualquer escala. Dada a importância da internet no cenário atual, onde ela assume papel fundamental no funcionamento de inúmeros serviços, é importante projetar aplicações capazes de garantir a operação pelo maior tempo possível. Portanto, este trabalho tem como objetivo desenvolver uma aplicação de alta disponibilidade em uma plataforma de computação em nuvem utilizando containers, de modo a verificar a viabilidade do uso desta tecnologia para este tipo de infraestrutura. A aplicação desenvolvida atendeu aos critérios necessários para ser considerada de alta disponibilidade, oferecendo balanceamento de carga, escalabilidade e servidores alocados em diferentes data centers. Os containers se mostraram viáveis para serem utilizados em aplicações deste tipo, pois oferecem total compatibilidade com os serviços das plataformas, além de serem mais leves e rápidos que máquinas virtuais.

Abstract. Nowadays, cloud computing platforms offer on-demand services for businesses of any size. Given the current importance of the internet, which is fundamental for running many services, it is important to design applications that are able to guarantee the operation for most of the time as possible. Therefore, this work aims to develop a high availability application in a cloud computing platform using containers, to check the viability of this technology when working with an infrastructure of this kind. The application met the high availability requirements, by having load balancing, scalability and servers in different data centers. Containers proved to be viable to be used in applications of this type, because they offer full compatibility with cloud services, and are lighter and faster than virtual machines.

1. Introdução

Com o aumento do uso da internet para as mais diversas finalidades, surgem novos desafios para os profissionais da área de infraestrutura, que devem garantir a disponibilidade de sites e aplicações web pelo maior tempo possível. Sabendo disso, plataformas de computação em nuvem oferecem serviços modernos, resilientes e sob medida para negócios de qualquer escala, onde o cliente paga conforme a utilização de cada recurso. No entanto, para aproveitar estas vantagens, é preciso usar tecnologias compatíveis. Este artigo irá abordar o uso de containers como alternativa para servir aplicações em nuvem, os quais permitem usar técnicas de alta disponibilidade oferecidas pelas plataformas.

De acordo com Perry (2022), alta disponibilidade é a capacidade de um sistema em garantir a continuidade dos serviços mesmo quando ocorrerem falhas (ex.: hardware, software, energia), e também em ser resiliente quando submetido a cenários críticos, como quantidades variáveis de tráfego ou falhas de comunicação com o servidor de uma

determinada região. Numa aplicação completa, há várias tecnologias que precisam oferecer alta disponibilidade, como os servidores, banco de dados e armazenamento de arquivos. Neste artigo, será abordada a parte dos servidores, e como a utilização de containers funciona em conjunto com os serviços de uma plataforma em nuvem. Ainda de acordo com Perry, para ser ter alta disponibilidade nos servidores, é necessário:

- Escalabilidade: capacidade da infraestrutura de aumentar e diminuir recursos computacionais dinamicamente, conforme a necessidade;
- Zonas de disponibilidade: disponibilizar servidores em mais de uma região geográfica, para que o tráfego possa ser desviado caso uma apresente problemas;
- Balanceamento de carga: distribuir o tráfego em múltiplos servidores.

Portanto, este trabalho tem como objetivo disponibilizar uma aplicação numa plataforma de computação em nuvem utilizando containers, de modo a verificar a viabilidade da utilização desta tecnologia ao se trabalhar com uma infraestrutura de alta disponibilidade. Dentro desse escopo, teremos os seguintes objetivos específicos:

- Desenvolver uma aplicação protótipo utilizando containers;
- Configurar uma infraestrutura com duas regiões de disponibilidade, balanceamento de carga e escalabilidade;
- Verificar a viabilidade do uso de containers para se trabalhar com este tipo de infraestrutura.

Este projeto será hospedado na plataforma Amazon Web Services (AWS)¹, pois oferece todos os serviços necessários para rodar aplicações de alta disponibilidade. Já o software utilizado para desenvolvimento e execução da aplicação em nuvem será o Docker.²

A seguir, encontra-se a fundamentação teórica, que elucida os principais conceitos envolvidos e traz alguns trabalhos relacionados com a área. Na sequência, a sessão de métodos mostra o detalhamento do projeto, explicando como foi feito o desenvolvimento da aplicação. Por fim, são apresentados e discutidos os resultados obtidos ao longo do trabalho, juntamente com as considerações finais.

2. Revisão da Literatura

Entre várias definições para a computação em nuvem, algumas características básicas são amplamente aceitas:

- A computação em nuvem é um modelo que permite acesso à rede de forma onipresente, conveniente e sob demanda a um conjunto compartilhado e distribuídos de recursos de computação configuráveis que podem ser rapidamente alocados e liberados com o mínimo de esforço de gerenciamento ou interação com o prestador de serviço [NIST, 2022];
- Possui autoatendimento sob demanda e amplo acesso de serviços de rede. Os recursos computacionais são usados para servir múltiplos usuários, sendo

¹ <https://aws.amazon.com/pt/>

² <https://www.docker.com/>

alocados e realocados dinamicamente conforme a demanda. Oferece elasticidade rápida e serviços mensuráveis, sendo transparente com o cliente [NIST, 2022];

- Computação em nuvem é uma plataforma que dinamicamente provê, configura, reconfigura e libera servidores de acordo com as necessidades e que emprega grandes data centers e potentes servidores, nos quais hospeda aplicações e serviços para serem utilizados via internet [DIÓGENES, VERAS, 2014];
- Computação em nuvem é um estilo de computação no qual recursos de TI, massivamente escaláveis, são disponibilizados sob a forma de serviços, por meio da internet, para múltiplos consumidores externos [GARTNER GROUP, 2022];

2.1 Motivação e relevância

Infraestruturas de alta disponibilidade são distribuídas e escaláveis. Portanto, para rodar uma aplicação em múltiplos servidores, com máquinas de diferentes configurações e em múltiplas regiões, faz-se necessário utilizar uma tecnologia capaz de padronizar ambientes e facilitar a entrega do produto final, o software.

Para atender a estas necessidades, o Docker foi criado em 2013 pela empresa DotCloud. Em sua palestra, onde apresentou o Docker, o criador Solomon Hykes explica que à medida que o software fica mais complexo, é difícil para o programador manter ambientes de desenvolvimento e produção iguais. Ainda de acordo com Hykes, “containers são unidades de software que funcionam em qualquer máquina, desde um simples laptop até um servidor gigantesco, e rodam igualmente pois possuem processos isolados e sistema de arquivos próprio” [HYKES, 2013]. De acordo com Carey (2021), a popularidade dos containers no mundo do desenvolvimento se deve basicamente ao Docker, que foi o responsável por transformar a forma de desenvolver, executar e disponibilizar aplicações, tornando esta tecnologia o padrão da indústria.

Portanto, faz-se necessário analisar aspectos desta tecnologia a fim de determinar a viabilidade de se trabalhar com ela em aplicações em nuvem, dada a importância de entregar software de alta disponibilidade.

2.2 Como *containers* funcionam

A virtualização em nível de sistema operacional, também conhecida como containerização, refere-se a um recurso do sistema operacional no qual o núcleo permite a existência de várias instâncias isoladas de espaço de usuário. Essas instâncias são chamadas de containers [CAREY, 2021]. Para Vitalino e Castro (2016), um container é agrupamento de uma aplicação junto com suas dependências, que compartilham o kernel do sistema operacional do host, ou seja, da máquina (seja ela virtual ou física) onde está rodando. Ainda de acordo com os autores, os containers são bem similares às máquinas virtuais, porém mais leves e mais integrados ao sistema operacional da máquina host, uma vez que compartilha o seu kernel, proporcionando melhor desempenho por conta do gerenciamento único dos recursos.

Quando utiliza-se máquinas virtuais, emula-se um novo sistema operacional completo que emprega mais recursos da máquina física, o que não ocorre ao usar containers, já que os recursos são compartilhados. O ganho disso é uma eficiência muito

maior, além da capacidade de rodar mais containers em um único host, se comparado com a quantidade que se conseguiria com máquinas virtuais [RUBENS, 2017].

A Figura 1 mostra três aplicações sendo executadas, primeiro nativamente, depois com máquinas virtuais e, por fim, com containers. Pode-se perceber que diferente das máquinas virtuais, não é necessário emular um novo sistema operacional ao utilizar containers.

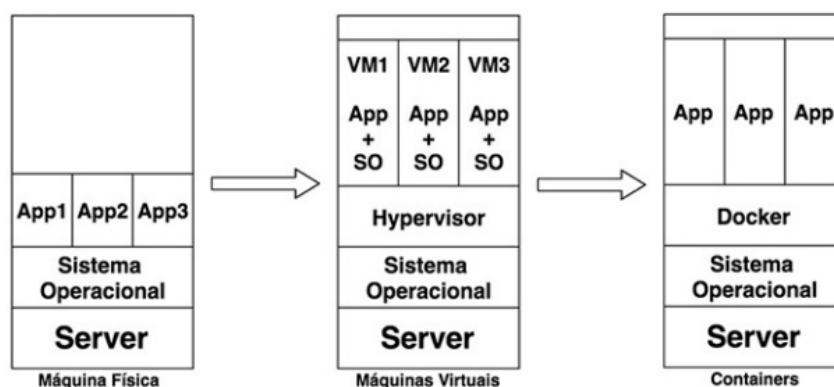


Figura 1. Diferenças de aplicações executadas nativamente, em máquinas virtuais e containers [Vitalino e Castro, 2016].

Um exemplo prático de isolamento é ter as aplicações 1, 2 e 3 rodando em diferentes versões do Java (linguagem de programação para aplicações ao lado do servidor). Com o uso de containers, não haveria problema algum, pois cada aplicação fica em seu ambiente isolado. Todas as dependências, incluindo o Java na versão de cada aplicação estão instalados no container, sem que eles afetem uns aos outros. Já na máquina nativa, isso é muito mais complexo, já que realizar qualquer instalação ou configuração no host afeta diretamente todas as aplicações que estão sendo executadas, causando problemas imediatos de compatibilidade entre elas. Na máquina virtual não haveria problemas de compatibilidade e as aplicações também estariam isoladas. Porém, o desempenho seria muito inferior, já que a máquina física estaria consumindo muito mais recursos ao executar três sistemas operacionais completos sobre o sistema operacional nativo.

2.3 Padronização e compatibilidade

Por oferecerem um ambiente isolado e garantirem execução consistente, containers podem ser executados na maioria dos sistemas operacionais e plataformas de computação em nuvem. Independente de onde estiver rodando, o ambiente será igual para todos, eliminando problemas de compatibilidade e de diferenças entre o ambiente de desenvolvimento e produção [RUBENS, 2017].

Conforme a documentação oficial do Docker, ele pode ser executado nos sistemas operacionais Linux, MacOS (em arquiteturas x86-64), ARM e outras, e em Windows (x86-64). Esta abrangência dos principais sistemas operacionais permite a portabilidade entre a equipe de desenvolvimento, já que é possível padronizar o ambiente da aplicação independente do sistema operacional utilizado.

Em relação à compatibilidade com as plataformas de computação em nuvem, as maiores e principais empresas já oferecem recursos para execução de containers há anos. As três maiores plataformas do mercado (AWS, Google Cloud³ e Azure⁴) possuem verdadeiros ecossistemas em torno de containers, oferecendo inúmeros serviços para hospedagem, execução, orquestração e gerenciamento. Isso permite que uma empresa possa migrar ou mesmo utilizar várias plataformas, sem se preocupar em instalar a aplicação manualmente, visto que o container oferece um ambiente completo, isolado e compatível com qualquer serviço [FOSTAINI, 2020]. Abaixo, há exemplos de serviços fornecidos por estas empresas por finalidade:

- AWS Elastic Container Service (ECS) e Azure Container Instances: Serviços para execução de containers;
- AWS Elastic Container Registry (ECR) e Azure Container Registry (ACR): Serviços para registro de imagens para containers;
- AWS Fargate, AWS Elastic Kubernetes Service (EKS), Azure Kubernetes Service (AKS) e Google Kubernetes Engine (GKE): Serviços para orquestração de containers.

2.4 Imagens

Uma imagem é um modelo que contém um conjunto de instruções necessárias para criar e executar containers [SCHENKER, 2020]. É possível comparar uma imagem de container a um snapshot de uma máquina virtual. Uma imagem Docker contém o código da aplicação e todas as dependências necessárias para executá-la. Em um cenário onde há um aumento de usuários acessando uma aplicação, é necessário aumentar o número de containers para atender a todas as requisições, os quais são criados a partir de uma imagem.

2.5 Regiões e Zonas de disponibilidade da AWS

De acordo com a documentação oficial da AWS, uma região é um local físico onde estão agrupados *data centers*. Uma zona de disponibilidade (*AZ - Availability Zone*) é um grupo de *data centers* que pertencem a uma região.

Cada AZ tem energia, refrigeração e segurança física independentes e está conectada por meio de redes redundantes de latência ultrabaixa. Os clientes da AWS, focados em alta disponibilidade, podem projetar seus aplicativos para serem executados em várias AZs, a fim de obter tolerância a falhas ainda maior. As regiões de infraestrutura da AWS atendem aos mais altos níveis de segurança, conformidade e proteção de dados [AWS, 2022].

Portanto, a vantagem de uma arquitetura que suporte múltiplas regiões geográficas é garantir maior disponibilidade, uma vez que a aplicação não depende de apenas um *data center* para garantir seu funcionamento.

2.6 Balanceamento de carga

De acordo com Piper e Clinton (2021), um balanceador de carga é um serviço que fica à frente de uma infraestrutura, sendo responsável por atender a qualquer requisição por

³ <https://cloud.google.com/?hl=pt-br>

⁴ <https://azure.microsoft.com/pt-br/>

conteúdo. Ao receber uma requisição, o balanceador vai distribuir o tráfego para os servidores e responder de volta para o cliente. Dessa forma, garante-se maior estabilidade a uma aplicação, já que haverá múltiplos servidores processando requisições. Além disso, o balanceador pode distribuir o tráfego de forma inteligente, sempre buscando equalizar a utilização de CPU de cada servidor para não sobrecarregar máquinas específicas.

2.7 Escalabilidade

De acordo com Fowler (2009), um sistema escalável é aquele que permite adicionar *hardware* e obter uma melhora proporcional no desempenho, como dobrar o número de servidores para dobrar o *throughput*. Existem duas modalidades de escalabilidade, a vertical e a horizontal. Ainda de acordo com Fowler (2009), a escalabilidade vertical é adicionar mais poder a um único servidor (por exemplo, acrescentar memória). Já a escalabilidade horizontal, significa adicionar mais servidores.

Atualmente as aplicações estão cada vez mais leves, robustas e fáceis de se adaptar a cargas de trabalho inconstantes, devido ao uso de containers e suas características. Para Sedaghat et al. (2013), sistemas que podem realizar o gerenciamento de forma automática são fundamentais, principalmente quando se fala de infraestruturas na nuvem. Isto ocorre principalmente por causa da complexidade e pela proporção da infraestrutura propriamente dita, como também pelo fato de que as ações que são tomadas no gerenciamento devem ser realizadas de maneira muito rápida.

Segundo Taherizadeh e Stankovski (2017), a provisão dinâmica de aplicações de nuvem de acordo com uma carga de trabalho inconstante pode ser realizada por meio de dois tipos de tecnologias de virtualização: baseada em máquinas virtuais (VMs) ou em containers. Atualmente, há um grande crescimento na utilização dos containers, pois máquinas virtuais precisam rodar um sistema operacional completo para cada instância, fazendo que ocupem mais espaço, sejam mais lentas para inicializar e mais difíceis para manter, atualizar e executar em diferentes ambientes.

2.8 Orquestração

A orquestração de containers é a automação de tarefas necessárias para executar aplicações containerizadas, como a instalação, escalonamento, balanceamento de carga, monitoramento e provisionamento de recursos [Red Hat, 2022]. Como já mencionado, escalabilidade significa a capacidade de ajustar os recursos de infraestrutura para atender quantidades variáveis de usuários. Utilizando os recursos de forma inteligente, é possível reduzir custos e tornar o sistema resiliente quando muitos acessos ocorrem simultaneamente [CASALICCHIO, 2016].

Ainda de acordo com o autor, os orquestradores servem para gerenciar o ciclo de vida dos containers. De maneira geral, a orquestração permite configurar como os containers irão se comportar de acordo com a demanda de serviço. Tarefas como disponibilidade e redundância de containers, adicionar ou remover containers de acordo com o uso e promover ambiente de implantação adequado são algumas ações que são previamente definidas de acordo com as políticas adotadas.

2.9 Trabalhos relacionados

No artigo *High Availability in Clouds*, os autores realizaram uma pesquisa sobre alta disponibilidade em nuvem, mostrando quais são os desafios presentes nesta área e também fazendo um panorama geral do cenário. Na pesquisa, apontam diversos casos onde a interrupção de um serviço ocasionou significativos prejuízos financeiros, mostrando assim a importância da discussão do tema [Endo, P.T. et al. 2016].

Já no artigo *Using Docker in high performance computing applications*, os autores explicam que a virtualização tem papel fundamental na computação em nuvem. Portanto, realizam um estudo com aplicações de alta performance, comparando o desempenho delas ao serem executadas em containers Docker e máquinas virtuais [M. T. Chung. et al. 2016].

O artigo de Silva e Carvalho (2018) aborda que a computação em nuvem cresceu muito ao longo dos anos e, com ela, a necessidade de prover infra estruturas que se adaptem à demanda dinamicamente. Assim, realizou-se de forma teórica uma análise de mecanismos de *auto scaling* em aplicações baseadas em containers, mostrando os diferentes tipos de escalabilidade, ferramentas e serviços, além de reunir as principais características dos mesmos, em busca de ajudar a apoiar a tomada de decisão de qual método deve-se escolher em determinada situação.

No artigo *A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments*, os autores fazem uma avaliação de diferentes estratégias de escalabilidade para ambientes de computação em nuvem. Neste sentido, abordam a necessidade de configurar corretamente uma estratégia conforme os requisitos específicos de uma aplicação. Assim, mostram que estratégias mal configuradas podem reduzir a performance da aplicação e também levar ao desperdício de recursos [M. A. S. Netto et al. 2014].

3. Métodos

Este trabalho inicia sua metodologia com o princípio de estudo de caso. De acordo com Yin (2015), o estudo de caso é uma investigação contemporânea em profundidade e em seu contexto de mundo real, especialmente quando os limites entre o fenômeno e o contexto podem não ser claramente evidentes.

A orientação de pesquisa para nortear esse trabalho é a qualitativa. De acordo com Gerhardt e Silveira (2009, p. 32), a pesquisa qualitativa não se preocupa com representatividade numérica, mas sim com o aprofundamento da compreensão. Nesse tipo de pesquisa, o cientista é ao mesmo tempo o sujeito e o objeto de suas pesquisas. Ainda segundo as autoras, o desenvolvimento das pesquisas é imprevisível. Preocupa-se com aspectos da realidade que não podem ser quantificados, centrando-se na compreensão e explicação da dinâmica das relações sociais.

Já para o desenvolvimento da parte prática, utilizou-se o método exploratório. O objetivo da pesquisa exploratória, de acordo com , é proporcionar mais informações sobre o assunto que será investigado, possibilitando sua definição e delineamento. Esse tipo de pesquisa possui planejamento flexível, o que permite o estudo do tema sob diversos ângulos e aspectos, dada a necessidade de compreender as tecnologias envolvidas buscando a melhor alternativa para o projeto.

3.1 Aplicação

Para demonstrar o funcionamento de uma arquitetura de alta disponibilidade, foi desenvolvida uma aplicação para calcular o retorno de investimentos em renda fixa, conforme a Figura 2. Para tal, foi utilizada a linguagem de programação PHP 8.1 e o servidor HTTP NGINX. Em seguida, foi gerada uma imagem Docker desta aplicação, contendo o PHP, o NGINX e o código-fonte da aplicação.



Figura 2. Interface da aplicação [autor].

3.2 Infraestrutura

Em plataformas de computação em nuvem, a infraestrutura é configurada pelo próprio usuário. As redes são definidas via software (paradigma SND - *Software Defined Networking*), permitindo alta possibilidade de customização. Para esta aplicação, a rede foi configurada conforme a Figura 3.

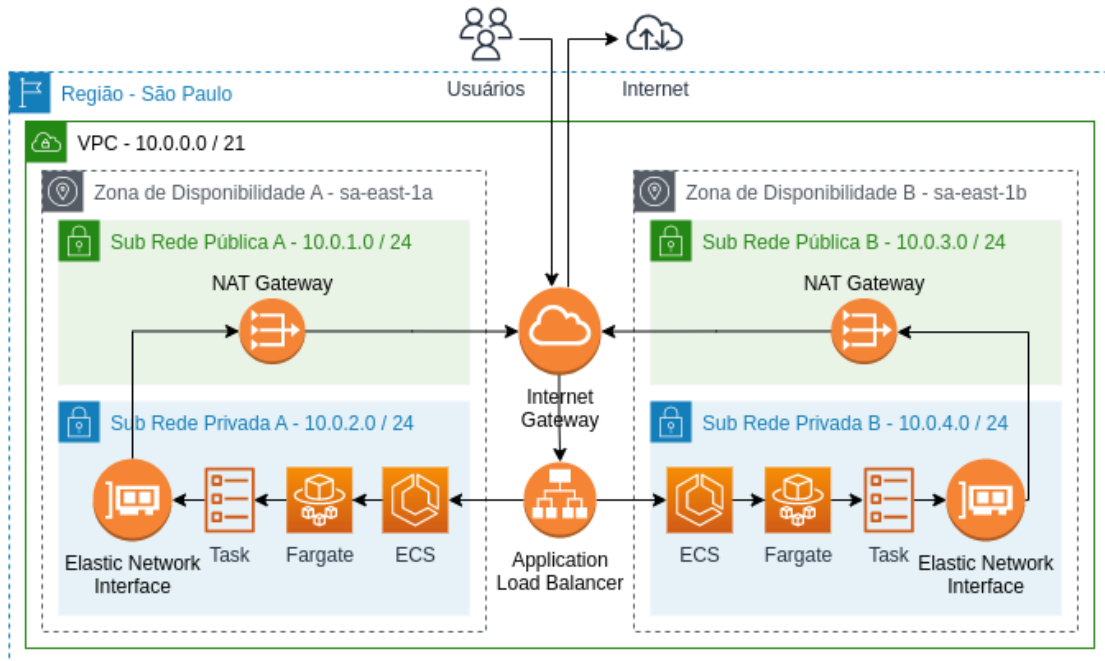


Figura 3. Diagrama da infraestrutura em nuvem da aplicação [autor].

3.3 Tecnologias e serviços

Nesta infraestrutura, faz-se necessário a utilização de vários serviços que se comunicam entre si para processarem requisições. Primeiramente, temos uma VPC (*Virtual Private Cloud*), que é um serviço que permite iniciar recursos da AWS em uma rede virtual isolada logicamente. Nela é disponibilizado controle total sobre o ambiente de redes virtuais, incluindo a seleção de intervalo de endereços IP, a criação de sub-redes e a configuração de tabelas de rotas e *gateways* de rede [AWS, 2022].

Atendendo ao primeiro requisito de infraestrutura de alta disponibilidade, configurou-se duas zonas de disponibilidade independentes. Para distribuir o tráfego e atender ao segundo requisito, criou-se o balanceador de carga. A AWS oferece este serviço por meio do Elastic Load Balancing, que de acordo com a documentação, é capaz de distribuir as requisições entre múltiplos destinos, como máquinas, containers e endereços de IP. Estes recursos podem estar em uma ou múltiplas zonas de disponibilidade. Além disso, ele também atua monitorando o status da infraestrutura. Caso algum servidor apresente problemas e pare de funcionar, o balanceador automaticamente direciona requisições apenas para máquinas saudáveis [AWS, 2022].

Para gerenciar o ciclo de vida dos containers, utilizou-se o serviço AWS Fargate, que faz parte de outro serviço chamado Elastic Container Service (ECS). O AWS Fargate é um mecanismo de computação sem servidor e com pagamento conforme o uso que permite disponibilizar aplicações sem gerenciar servidores [AWS, 2022]. O AWS Fargate foi escolhido pois elimina a necessidade de gerenciar a infraestrutura física. Basta configurá-lo informando as características desejadas (quantidade mínima e máxima de máquinas, memória, CPU) e se encarrega de disponibilizar máquinas físicas para executar os containers.

Em seguida, temos a Task (definição de tarefa), entidade onde é configurada a execução de um container, determinando a imagem modelo, a capacidade de CPU, memória, logs e comportamentos ao escalar instâncias [AWS, 2022]. Nesta etapa, a imagem da aplicação foi enviada para o serviço Elastic Container Registry da AWS (ECR), de modo que pudesse ser integrada com a definição de tarefa,

Para que uma Task possa acessar a sub rede pública, a AWS disponibiliza automaticamente um serviço de interface de rede (Elastic Network Interface). De acordo com Piper e Clinton (2021), o *Internet Gateway* possui um endereço de IP público, e permite que determinados recursos possam receber requisições externas e também acessar a internet. Já o *Nat Gateway* funciona de forma similar ao *Internet Gateway*, porém opera apenas em uma via, permitindo que recursos de uma sub rede privada acessem serviços fora da VPC, e não permitindo que serviços externos se conectem diretamente com estes recursos [Patel, 2019].

3.4 Regra de Escalabilidade

Para a configuração da escalabilidade, foram definidas as seguintes regras: aumentar a infraestrutura em uma instância quando a utilização de CPU for maior que 75% durante 3 minutos; diminuir a infraestrutura em uma instância quando a utilização de CPU for menor que 25% durante 3 minutos; manter o mínimo de duas instâncias, independente do uso de CPU; aumentar até o máximo de oito instâncias. Através do uso de containers,

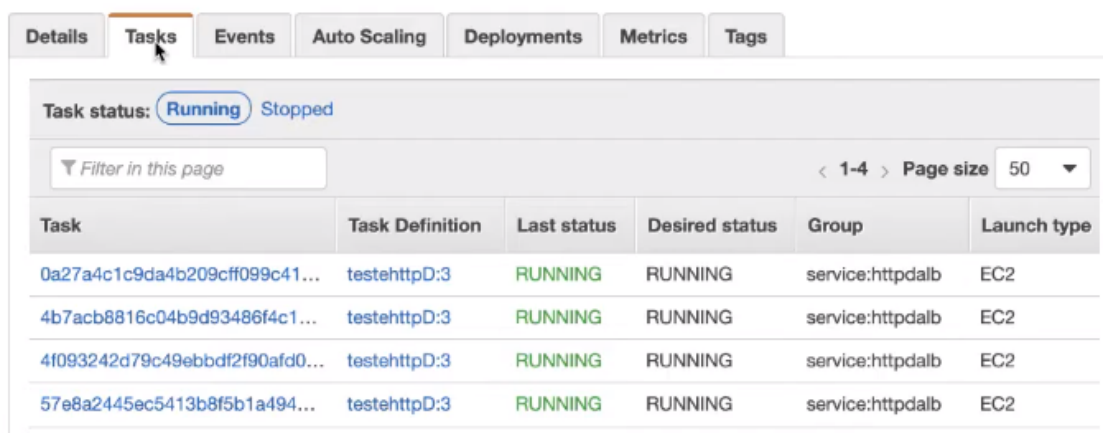
foi possível configurar a política de escalabilidade de modo que o serviço pudesse ajustar a infraestrutura automaticamente, utilizando a imagem da aplicação como modelo para criar novos servidores.

Visto que o objetivo do estudo consiste em verificar a viabilidade do uso de containers para uso em infraestruturas escaláveis, os parâmetros foram arbitrários, apenas para que a política pudesse ser estabelecida. Testes de carga para comprovarem o ajuste automático requerem análises específicas conforme o caso, pois o percentual de uso de CPU depende não somente da finalidade da aplicação, mas de muitos outros fatores como a própria capacidade de CPU, linguagem de programação, sistema operacional, etc. Além disso, um teste de carga agressivo pode fazer com que as máquinas deixem de responder ao serviço de escalonamento, fazendo com que o sistema suspenda todas as máquinas e crie novas no lugar, causando indisponibilidade momentânea. Para evitar este tipo de ataque conhecido como negação de serviço (DDoS), a AWS oferece o serviço AWS Shield, o qual não foi possível testar devido ao elevado custo da versão configurável.

No entanto, através do painel de configuração, é possível realizar o ajuste da quantidade de servidores de forma manual. Desta forma, verificou-se que ao alterar a quantidade de Tasks desejadas, o serviço de escalonamento logo providenciava o alinhamento da infraestrutura, sendo possível observar através do painel, em tempo real, a quantidade de máquinas ativas correspondendo à configuração. Além disso, o sistema apresentou logs de eventos, deixando registrado o momento exato em que realizou determinada ação, seguido de justificativa e detalhes do ocorrido.

4. Resultados

Através desta pesquisa e desenvolvimento, foi possível implementar uma aplicação de alta disponibilidade usando containers conforme havia sido planejado. Neste processo, conforme apresentado na Seção 3.2, foi constatado que a infraestrutura atendeu aos critérios necessários para ser considerada de alta disponibilidade, pois possui balanceamento de carga, escalabilidade, e servidores em diferentes *data centers*.



Task	Task Definition	Last status	Desired status	Group	Launch type
0a27a4c1c9da4b209cff099c41...	testehttpD:3	RUNNING	RUNNING	service:httpdalb	EC2
4b7acb8816c04b9d93486f4c1...	testehttpD:3	RUNNING	RUNNING	service:httpdalb	EC2
4f093242d79c49ebbd2f90afd0...	testehttpD:3	RUNNING	RUNNING	service:httpdalb	EC2
57e8a2445ec5413b8f5b1a494...	testehttpD:3	RUNNING	RUNNING	service:httpdalb	EC2

Figura 4. Painel AWS: visualização da lista de Tasks em execução [autor].

Na Figura 4, pode-se observar o painel de controle da AWS, que nesta seção apresenta a lista de Tasks em execução. Neste instante, haviam quatro containers ativos.

Além do monitoramento em tempo real, o painel disponibiliza a aba *Events*, que mostra os registros de atividades executadas pelo serviço ECS (*Elastic Container Service*).

Os containers se mostraram viáveis para serem utilizados neste tipo de infraestrutura, e suas características de leveza e portabilidade fazem com que levem vantagem em relação às máquinas virtuais. Por terem total compatibilidade com serviços de alta disponibilidade, mostram-se uma das melhores tecnologias disponíveis atualmente.

Utilizando o Docker, a aplicação funcionou corretamente tanto em ambiente de desenvolvimento quanto em ambiente de produção. Dessa forma, foi possível padronizar os ambientes, pois não foi necessário fazer qualquer adaptação ao instalar a aplicação no servidor em nuvem.

5. Considerações finais

Este trabalho teve como objetivo implementar uma infraestrutura em nuvem de alta disponibilidade usando containers, permitindo analisar tanto aspectos técnicos como a viabilidade do uso desta tecnologia para esta finalidade.

A respeito da infraestrutura, observou-se que ao trabalhar com arquiteturas de alta disponibilidade surgem vários elementos aos quais o desenvolvedor precisa ter conhecimento, como fundamentos de redes, redes definidas via software, gerenciamento de serviços em nuvem e orquestração de containers. Observou-se também que a interação entre os elementos torna-se mais complexa em relação a uma infraestrutura que não conta com serviços de alta disponibilidade, visto que para utilizá-los é necessário alocar recursos em zonas e sub redes diferentes, e trabalhar com mecanismos de rede para conectá-los.

Apesar da íngreme curva de aprendizado para dominar todas as tecnologias envolvidas, observou-se que o seu uso é indispensável em sistemas que atingem uma escala maior e necessitam de alta disponibilidade. Além disso, pôde-se observar nas pesquisas que cada vez mais empresas têm adotado infraestruturas que seguem este modelo, o que o torna ainda mais atrativo do ponto de vista de mercado de trabalho [TI Inside, 2022].

Neste trabalho, por se tratar de uma análise qualitativa de aspectos tecnológicos das ferramentas e serviços abordados, os processos de configuração de infraestrutura e instalação da aplicação foram feitos de forma manual. Numa aplicação comercial, seria necessário automatizar estes processos, adaptando algumas rotinas da organização. Para isso, seria necessário incorporar algumas ferramentas e conceitos do conjunto de atividades conhecido como DevOps. O DevOps permite à empresa aumentar a capacidade de distribuir aplicativos e serviços, otimizando e aperfeiçoando produtos em um ritmo mais rápido do que o das empresas que usam processos tradicionais de desenvolvimento de *software* e gerenciamento de infraestrutura [AWS, 2022].

No entanto, conforme já citado na Seção 1, existem outras tecnologias que precisam ser implantadas para garantir a alta disponibilidade completa, como banco de dados e armazenamento de arquivos. Estas duas são indispensáveis para a maioria das aplicações, e podem representar graves problemas de performance se não forem

projetadas pensando em alta disponibilidade. Portanto, conhecendo estas possíveis debilidades, faz-se necessário desenvolver futuros projetos de pesquisa que apontem soluções para estes problemas.

6. Referências bibliográficas

- AWS. Recursos do Amazon Virtual Private Cloud. AWS, 2022. Disponível em: <<https://aws.amazon.com/pt/vpc/features/>>. Acesso em: 21 nov. 2022.
- AWS. Elastic Load Balancing. Distribua o tráfego de rede para melhorar a escalabilidade da aplicação AWS, 2022. Disponível em: <<https://aws.amazon.com/pt/elasticloadbalancing/>>. Acesso em: 21 nov. 2022.
- AWS. AWS Fargate. Computação sem servidor para contêineres. AWS, 2022. Disponível em: <<https://aws.amazon.com/pt/fargate/>>. Acesso em: 21 nov. 2022.
- AWS. ECS Task Definitions. 2022. Disponível em: <https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task_definitions.html>. Acesso em: 21 nov. 2022.
- AWS. Regiões e zonas de disponibilidade. AWS, 2022. Disponível em: <https://aws.amazon.com/pt/about-aws/global-infrastructure/regions_az/>. Acesso em: 21 nov. 2022.
- AWS. O que é o DevOps? AWS, 2022. Disponível em: <<https://aws.amazon.com/pt/devops/what-is-devops/>>. Acesso em: 28 nov. 2022.
- CAREY, Scott. What is Docker? The spark for the container revolution. InfoWorld, 2021. Disponível em: <<https://www.infoworld.com/article/3204171/what-is-docker-the-spark-for-the-container-revolution.html>>. Acesso em: 18 jun. 2022.
- CASALICCHIO, Emiliano. Autonomic Orchestration of Containers: Problem Definition and Research Challenges. 2016.
- DIÓGENES, Yuri; VERAS, Manoel. Certificação Cloud Essentials: Guia preparatório para o exame CLO-0001. Rio de Janeiro: Nova-terra, 2014.
- Endo, P.T., Rodrigues, M., Gonçalves, G.E. et al. High availability in clouds: systematic review and research challenges. J Cloud Comp 5, 16 (2016).
- FOSTAINI, Renicius Pagotto. Docker: Desmistificando a containerização de Aplicações. Disponível em: <<https://renicius-pagotto.medium.com/docker-desmistificando-a-containeriza%C3%A7%C3%A3o-de-aplica%C3%A7%C3%B5es-f0cbe208005b>>. Acesso em: 17 abril 2022.
- FOWLER, Martin. Padrões de Arquitetura de Aplicações Corporativas. São Paulo: Bookman, 2009.
- GARTNER GROUP. What is cloud computing? Disponível em: <<https://www.gartner.com/en/topics/cloud>>. Acesso em: 29 nov. 2022.
- GERHARDT, Tatiana Engel; SILVEIRA, Denise Tolfo. Métodos de pesquisa. Porto Alegre: Editora UFRGS, 2009.
- HYKES, Solomon. Why we built Docker. YouTube, 1 ago. 2013. Disponível em: <<https://www.youtube.com/watch?v=3N3n9FzebAA>>. Acesso em: 27 nov. 2022.

- M. A. S. Netto, C. Cardonha, R. L. F. Cunha and M. D. Assuncao, "Evaluating Auto-scaling Strategies for Cloud Computing Environments," 2014 IEEE 22nd International Symposium on Modelling, Analysis & Simulation of Computer and Telecommunication Systems, 2014, pp. 187-196, doi: 10.1109/MASCOTS.2014.32.
- M. T. Chung, N. Quang-Hung, M. -T. Nguyen and N. Thoai, "Using Docker in high performance computing applications," 2016 IEEE Sixth International Conference on Communications and Electronics (ICCE), 2016, pp. 52-57, doi: 10.1109/CCE.2016.7562612.
- NIST - NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. The NIST Definition of Cloud Computing. Disponível em: <<https://csrc.nist.gov/publications/detail/sp/800-145/final>>. Acesso em: 29 nov. 2022.
- PATEL, Ashish. AWS - Difference between Internet Gateway and NAT Gateway. Disponível em: <<https://medium.com/awesome-cloud/aws-vpc-difference-between-internet-gateway-and-nat-gateway-c9177e710af6>>. Acesso em: 27 nov. 2022.
- PIPER, Ben; CLINTON, David. AWS Certified Solutions Architect Study Guide: Associate (SAA-C02) Exam. 3.Ed. Indianapolis, Indiana: Sybex, 2021.
- PERRY, Yifat. (2021). AWS High Availability: Compute, SQL and Storage. Disponível em: <<https://bluexp.netapp.com/blog/understanding-aws-high-availability-compute-sql-and-storage>>. Acesso em 13 nov. 2022.
- PRODANOV, Cleber Cristiano; FREITAS, Ernani Cesar de. Metodologia do Trabalho Científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico - 2ª Edição. Editora Feevale. Rio Grande do Sul. 2013.
- RED HAT. What is container orchestration? [S.I] [2015]. Disponível em: <<https://www.redhat.com/en/topics/containers/what-is-container-orchestration>> Acesso em: 12 nov. 2022.
- RUBENS, Paul. What are containers and why do you need them? CIO, 2017. Disponível em: <<https://www.cio.com/article/247005/what-are-containers-and-why-do-you-need-them.html>>. Acesso em: 17 abr. 2022.
- SCHENKER, Gabriel N. Docker - Fundamentals of Docker 19.x. Birmingham: Packt Publishing, 2020.
- SEDAGHAT, Mina; Hernandez Rodriguez, Francisco; Elmroth, Erick. (2013). A virtual Machine Re-packing Approach to the Horizontal vs. Vertical Elasticity Trade-off for Cloud Autoscaling. Disponível em: <https://www.researchgate.net/publication/242335395_A_Virtual_Machine_Re-packing_Approach_to_the_Horizontal_vs_Vertical_Elasticity_Trade-off_for_Cloud_Autoscaling>. Acesso em: 17 jun. 2022.
- SILVA, Van Eyck; CARVALHO, Marcus. Análise de Mecanismos de Autoscaling para Aplicações Baseadas em Containers. 2018.
- TAHERIZADEH, Salman; Stankovski, Vlado. (2017). Auto-scaling Applications in Edge Computing: Taxonomy and Challenges. Disponível em: <<https://dl.acm.org/doi/10.1145/3175684.3175709>>. Acesso em: 17 jun. 2022.

TI INSIDE. Impulso disponibiliza 5 mil bolsas de estudos com foco em DevOps e AWS. 2022. Disponível em: <<https://tiinside.com.br/27/10/2022/impulso-disponibiliza-5-mil-bolsas-de-estudos-com-foco-em-devops-e-aws/>>. Acesso em: 1 dez. 2022.

VITALINO, Jeferson Fernando Noronha; CASTRO, Marcus André Nunes. Descomplicando o Docker. Rio de Janeiro: Brasport, 2016

YIN, Robert K. Estudo de Caso - Planejamento e Métodos. Porto Alegre: Bookman Editora. 5.Ed. 2015.