DOCKER: CONSTRUINDO AMBIENTES SEGUROS E SUSTENTÁVEIS

DOCKER: BUILDING SAFE AND SUSTAINABLE ENVIRONMENTS

Anthony Vitor dos Santos Genesino
Faculdade de Tecnologia de Americana – Ministro Ralph Biasi
anthony.genesino@fatec.sp.gov.br

Renan André Vasconcellos
Faculdade de Tecnologia de Americana – Ministro Ralph Biasi
renan.vasconcellos@fatec.sp.gov.br

Richard Coelho Cunha
Faculdade de Tecnologia de Americana – Ministro Ralph Biasi
richard.cunha@fatec.sp.gov.br

João Emmanuel D Alkmin Neves
Faculdade de Tecnologia de Americana – Ministro Ralph Biasi
joao.neves11@fatec.sp.gov.br

Resumo

Este artigo investiga a relação entre sustentabilidade e Segurança da Informação utilizando *Docker*, a tecnologia de conteinerização de serviço mais popular do mercado. O objetivo deste estudo é analisar as contribuições do *Docker* nesses aspectos, com base em pesquisas e comparações com cenários onde a tecnologia não foi implementada. A metodologia utilizada no desenvolvimento é embasada em fontes bibliográficas de bases de dados fidedignas. Essas pesquisas têm como propósito verificar se o *Docker* pode promover a sustentabilidade e manter a segurança dos ambientes. Portanto, os resultados deste trabalho científico fizeram uma contribuição significativa para a área de Tecnologia da Informação, garantindo a segurança dos sistemas informatizados e promovendo a sustentabilidade ambiental através do uso otimizado de recursos, proporcionado por essa ferramenta.

Palavras-chave: Conteinerização; TI Verde; Segurança da Informação.

Abstract

This article investigates the relationship between sustainability and Information Security using Docker, the most popular service containerization technology on the market. The objective of this study is to analyze Docker's contributions in these aspects, based on research and comparisons with scenarios where the technology has not been implemented. The methodology used in development is based on bibliographic sources from reliable databases. This research aims to verify whether Docker can promote sustainability and maintain the security of environments. Therefore, the results of this scientific work have made a significant contribution to the field of Information Technology, ensuring the security of computerized systems and promoting environmental sustainability through the optimized use of resources enabled by this tool.

Keywords: *Containerization; Green IT; Information Security.*

1. Introdução

Nos últimos anos, a Tecnologia da Informação (TI) tem sido cada vez mais associada à sustentabilidade nas empresas, visando otimizar e reduzir os gastos e desperdícios energéticos (Neves, 2021). Muitas empresas de tecnologia adotam soluções de virtualização de servidores para maximizar os recursos de máquina, cortar custos de TI e obter maior flexibilidade para atender às demandas do negócio (Almeida, 2011). No entanto, conforme destacado por Moura e D'Alkmin Neves (2021), é preocupante que muitas empresas não atribuam a devida importância à Segurança da Informação, o que representa uma falha crítica diante das legislações nacionais e internacionais existentes sobre o assunto. A não conformidade com essas regulamentações pode resultar em sérias consequências legais e financeiras.

Além disso, ataques cibernéticos podem exercer um impacto substancial nas operações empresariais, considerando a crescente dependência da TI para a continuidade dos negócios. Segundo Neves (2018), na contemporaneidade, caracterizada como a Era do Conhecimento, a TI desempenha um papel crucial não apenas na gestão de processos internos, mas também na interação com clientes e fornecedores, na proteção de dados sensíveis e na inovação de produtos e serviços. Qualquer interrupção ou comprometimento dos sistemas de TI devido a atividades maliciosas pode resultar em perdas financeiras significativas, danos à reputação da empresa e interrupções operacionais que comprometem a competitividade e a sustentabilidade dos negócios (Vicentine *et al.*, 2022). Portanto, a cibersegurança tornou-se uma prioridade estratégica para as organizações que buscam manter a integridade no ambiente digital.

Muitas vezes, apenas virtualizar não é a melhor alternativa porque pode originar um uso ineficiente dos recursos do servidor. No mercado atual, existem plataformas como o *Docker*, que utilizam os recursos da máquina de maneira mais inteligente e eficaz. Com o *Docker*, em vez de adicionar uma camada adicional de virtualização, são criados *containers* que proporcionam ambientes enxutos e altamente otimizados. Dessa forma, o servidor não apenas mantém uma camada de virtualização de máquinas, mas também se beneficia de uma camada de *containers* do *Docker*, resultando em melhor desempenho, maior eficiência e uma gestão mais simplificada dos ambientes de desenvolvimento e produção.

No entanto, como é possível criar e manter um ambiente seguro e, ao mesmo tempo, sustentável e eficiente relativamente ao uso de recursos computacionais e meio ambiente, por meio da ferramenta *Docker*?

O *Docker* pode afetar positivamente no uso consciente de recursos energéticos de *data centers*, uma vez que, a solução consegue realizar o uso inteligente dos recursos no local onde foi implementado, além de ter também os seus mecanismos de segurança, para proteger os *containers* de ameaças e ataques que podem surgir. Essa hipótese estuda como o *Docker* pode estar a contribuir para estes dois temas simultaneamente, entendendo o seu funcionamento e vantagens do seu uso em ambientes de produção e desenvolvimento.

Portanto, o objetivo desse estudo é investigar de que maneira esses dois tópicos, TI verde e Segurança da Informação, podem ser contribuídos nas empresas com a implementação e uso correto da solução de conteinerização, *Docker*, com o seu aproveitamento efetivo dos recursos e oferecendo certo nível de Segurança da Informação.

A justificativa deste estudo se diz respeito no quão grande é a responsabilidade das grandes empresas com o meio ambiente, diminuindo cada vez mais a degradação ambiental, descarte excessivo de rejeitos (servidores e equipamentos legados), uso eficiente de energia e entre outros, e, ao mesmo tempo, proteger e assegurar a integridade, confidencialidade e disponibilidade dos ambientes informatizados.

2. Referencial Teórico

Esta seção visa fornecer uma exposição elucidativa sobre o *Docker* e os tópicos pertinentes que serão discutidos neste artigo, com o intuito de promover uma compreensão mais

acessível ao leitor. Nesse contexto, destaca-se a crescente relevância do *Docker*, uma das principais inovações tecnológicas contemporâneas, assim como a *Blockchain*, a Internet das Coisas, a Inteligência Artificial e a Agricultura de Precisão, que têm sido amplamente reconhecidas na promoção da sustentabilidade e no aprimoramento das práticas de Segurança da Informação (Santos *et al.*, 2020; Neves *et al.*, 2023; Pedro *et al.*, 2024; Souza *et al.*, 2024). Essa importância se fundamenta na capacidade do *Docker* em possibilitar uma gestão mais eficiente dos recursos, mitigando o desperdício e fomentando a conservação ambiental. A adoção dessa tecnologia tem exercido influência nas operações empresariais, conferindo à Segurança da Informação e à busca pela sustentabilidade um status não meramente relevante, mas essencial para a competitividade e viabilidade no cenário de mercado contemporâneo.

2.1. Virtualização

A virtualização é uma técnica que representa a criação de ambientes virtuais independentes, visando o uso de recursos computacionais, como processamento, armazenamento, redes e entre outros. Essa prática tornou-se essencial devido aos seus principais objetivos, que incluem aumentar a eficiência operacional, maximizar a utilização de recursos, facilitar a escalabilidade, promover a flexibilidade na alocação de recursos e simplificar a gestão de ambientes computacionais complexos. Essa técnica desempenha um papel crucial em *data centers*, nuvem computacional e ambientes de desenvolvimento, trazendo benefícios como redução de custos, melhor recuperação de desastres e agilidade na implantação de serviços. Em resumo, apresenta diversas vantagens que contribuem para o avanço da tecnologia (Almeida, 2011).

Para se virtualizar um ambiente, utilizam-se as Máquinas Virtuais (VM), do inglês *Virtual Machines*, onde primeiramente deve-se criar uma VM, após isso, o hipervisor irá criar um espaço de memória virtual e selecionará os recursos necessários, como Unidade Central de Processamento (CPU), do inglês *Central Processing Unit*, memória, rede e armazenamento.

A estrutura de VM consiste em três partes:

- *Hardware*: Os recursos físicos que serão utilizados pelas VMs.
- Hipervisor: *Software* responsável por criar e executar as VMs, baseados nos recursos de *hardware*.

- Sistema Operacional e Aplicações: VM em si, com seu sistema operacional e suas aplicações em execução.

2.2. *Container*

Os *containers* representam uma unidade padrão de *software* que realiza o empacotamento do código e todas as suas dependências, para que a aplicação seja executada de forma confiável, rápida e íntegra por meio de um ambiente de computação para outro. Essa ação de empacotamento é feita através da segregação de processos no mesmo *kernel*, de forma que o processo seja isolado da melhor maneira em todo o ambiente. Uma imagem de *container* do *Docker* é um pacote de *software* autônomo e executável, incluindo tudo aquilo necessário para a execução da aplicação (Rad *et al.*, 2017).

Os *containers* sempre serão executados da mesma maneira, independentemente da infraestrutura. Eles isolam o *software* do seu ambiente e garantem que a sua funcionalidade seja unificada, apesar das diferenças entre o desenvolvimento e a preparação (Docker, 2024).

2.3. Diferença entre uma máquina virtual e os *containers*

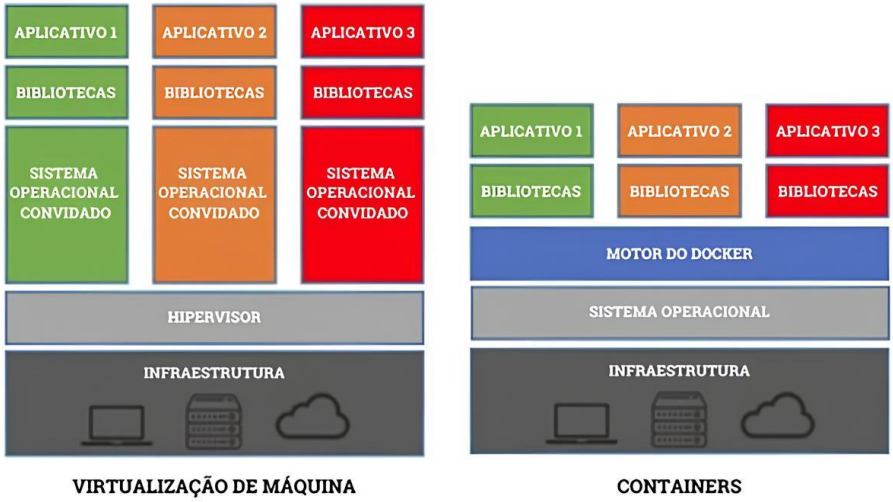
Os *containers* e as VMs são tecnologias de virtualização, porém elas possuem maneiras diferentes de funcionamento. Segundo Waldspurger (2002), para haver a distribuição de recursos de maneira mais eficiente, os *containers* fazem a distribuição e gestão desses recursos. No *container* há a virtualização feita ao nível de sistema operacional, compartilhando o *kernel* da máquina *host* e fazendo o isolamento do restante do ambiente, consequentemente, resultando em menor sobrecarga dos sistemas. Já as máquinas virtuais, empregam uma virtualização ao nível de *hardware*, portanto, cada VM terá o seu próprio *kernel* e sistema operacional, podendo resultar em maior uso adicional de recursos. Assim, as VMs proporcionam um isolamento mais robusto, o *container* oferece um bom desempenho e uma maior portabilidade (Waldspurger, 2002).

O *container* utiliza menos recursos físicos de uma máquina física ou até mesmo virtual (em cenários de grandes empresas com *softwares* de virtualização), uma vez que ele não precisa virtualizar o seu próprio *kernel*, mas sim utilizando *kernel* do hospedeiro. Além disso, pode oferecer um nível de isolamento utilizando o *cgroups*, sendo executado no modo *rootless mode.*

Por possuir um tempo de execução rápido e prático, também permite interromper um *container* em caso de incidentes de segurança, corrigir falhas, atualizar o microsserviço etc.

Para facilitar o entendimento, a Figura 1 ilustra a estrutura de uma máquina virtual e de um *container*, mostrando as diferenças na estrutura e no uso de recursos.

Figura 1 - Estrutura Máquina Virtual e *Containers*



Fonte: Elaborado pelos autores (2024) com base em Mendonça (2019)

- Na VM utiliza-se um hipervisor (ou até mesmo outro sistema operacional) para virtualizar outros sistemas operacionais completos e suas dependências;
- No *container* utiliza-se uma ferramenta de conteinerização onde são conteinerizados apenas os microsserviços e suas dependências, utilizando do mesmo *kernel* do sistema operacional hospedeiro.

Segundo a Absam (2024), VMs demandam mais recursos da infraestrutura do que *containers*, uma vez que não é necessário virtualizar outro sistema operacional para executar cada aplicação. Dessa forma, os *containers* são mais vantajosos, como, por exemplo:

- Isolamento: Protege dados evitando ataques, minimiza o impacto de problemas e garante a disponibilidade dos serviços e otimiza o uso de recursos do sistema;
- Gerenciamento Inteligente: Simplifica a orquestração e agiliza o desenvolvimento e permite escalabilidade automática e alta disponibilidade;

- Escalabilidade: Ajuste da capacidade de processamento e armazenamento conforme as necessidades, otimiza o uso de recursos existentes, escalonamento rápido para atender às demandas de carga e adaptação da capacidade às demandas da aplicação;
- Economia de Custos: Redução de investimentos em infraestrutura, otimização do consumo de energia e melhoria da lucratividade.

O *Docker* oferece benefícios distintivos, como a capacidade de isolar recursos e a eficiência na gestão de custos de infraestrutura, especialmente relevante para aplicações na Internet das Coisas. Mesmo compartilhando o mesmo núcleo do Sistema Operacional, o *Docker* consegue segregar efetivamente recursos de CPU, memória e armazenamento, graças à sofisticação de suas ferramentas integradas.

O modo de operação *rootless* destaca-se por fortalecer a cibersegurança do ambiente. Além disso, o *Docker* permite otimizar o uso de servidores existentes, maximizando recursos computacionais sem a necessidade de adquirir novos equipamentos, resultando em economia de recursos físicos e virtuais (Moraes *et al.*, 2022; Vertigo, 2019).

2.4. *Docker*

O *Docker* surgiu como uma inovação no ramo de virtualização, sendo apresentado ao mundo por Solomon Hykes, engenheiro, fundador e diretor executivo da empresa dotCloud, em uma apresentação no evento *Python Developers Conference*, em 15 de março de 2013 (Serra Junior e Lima, 2020).

Sendo assim, o *Docker* é uma plataforma *open source* que permite a criação, execução e testes de implantação de aplicações distribuídas dentro de *containers* executáveis. Nos *containers* de *software*, há a combinação do código-fonte de aplicações com as bibliotecas e estruturas do sistema operacional necessárias para a execução e implantação em qualquer ambiente, de forma rápida, confiável e estável. Atualmente, o *Docker* é utilizado principalmente com foco em microsserviços e serviços (Serra Junior e Lima, 2020).

Para executar em um ambiente com o *Docker*, é necessário ter:

- Infraestrutura: Servidor local ou em nuvem, por exemplo;
- Sistema operacional hospedeiro: *Software* base para o *Docker* poder ser instalado (Windows ou Linux, por exemplo);
- *Docker* por si próprio;
- Aplicativos em *containers*: São os ambientes segregados propriamente ditos, apartando cada ambiente do outro.

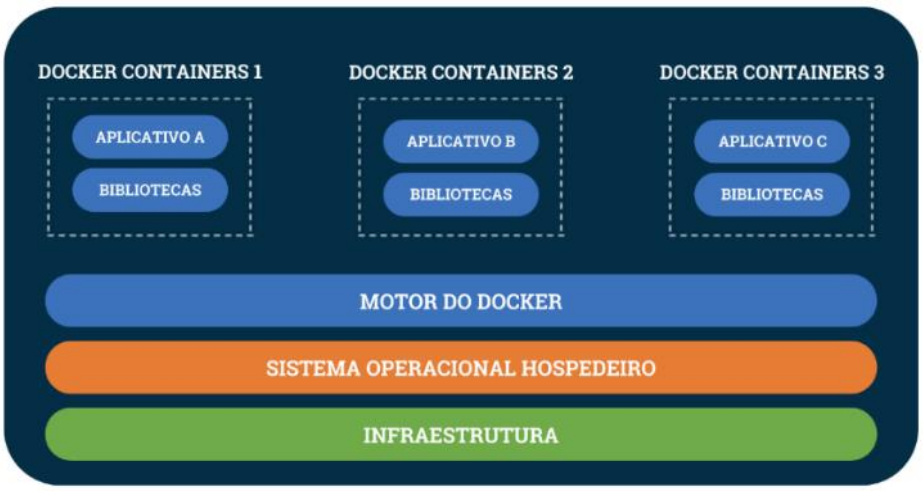
O *Docker* utiliza a virtualização ao nível do sistema operacional para permitir a execução de vários processos isolados, chamados de *containers*, no mesmo *host*. Para evitar a exaustão dos recursos da máquina por um único *container*, o *Docker* faz uso da funcionalidade *cgroups* do *kernel*, possibilitando a limitação do uso de *hardware*. Isso permite a coexistência de diferentes *containers* no mesmo *host* sem que um impacte negativamente o desempenho dos outros devido ao uso excessivo de recursos compartilhados (Gomes, 2020).

Os c*ontainers* do *Docker* destacam-se por:

- Padronização: Estabelecendo um padrão universal para *containers*. Isso proporciona segurança ao desenvolver e implementar aplicativos em quaisquer ambientes.
- Leveza: Compartilhando o *kernel* do sistema operacional, os *containers* são extremamente leves, evitando sobrecarga de recursos, resultando em eficiência, redução de custos e agilidade na implementação e provisionamento de aplicativos.
- Segurança: Oferecendo um alto nível de segurança ao isolar completamente os *containers* uns dos outros e do sistema operacional. Isso garante que problemas de segurança em um *container* não afetem outros *containers*.

Como exibido na Figura 2, a primeira camada é a infraestrutura, que seria o servidor no qual o sistema operacional hospedeiro está instalado. No sistema operacional hospedeiro, o motor do *Docker* é instalado e executado para que, assim, os *containers* de fato sejam executados e suas aplicações e dependências estejam separados do sistema hospedeiro.

Figura 2 - *Docker containers*



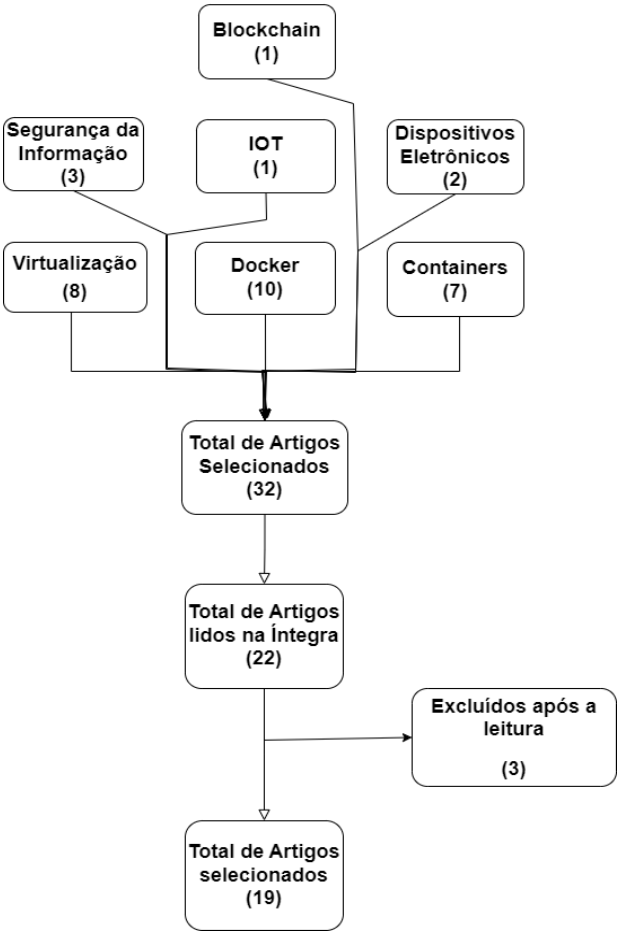
Fonte: Elaborado pelos autores (2024) com base em Altexsoft (2022)

3. Metodologia

Este artigo se fundamenta na análise e investigação de documentos teóricos que esclarecem os tópicos abordados, tais como a natureza da virtualização, a essência do *Docker*, a função dos *namespaces*, a definição de *containers*, além da intersecção entre *Docker*, Segurança da Informação e práticas de TI verde. Dessa forma, foi utilizado uma abordagem metodológica bibliográfica e qualitativa. A pesquisa foi direcionada para compreender a virtualização e *Docker*, visando a relação da sua aplicação contemporânea na Segurança da Informação e sustentabilidade, chegando na conclusão de que, com base na teoria, ele tem o objetivo de otimizar os recursos. Além disso, investigaram-se as distinções entre uma VM convencional e um *container*, destacando os benefícios do último em várias situações.

Com relação às pesquisas na Figura 3, foram considerados artigos entre os anos de 2008 a 2023. Após a análise dos títulos, resumos e leitura na íntegra dos artigos, os que não atingiram a expectativa inicial da temática proposta foram excluídos. Foram revisados 32 artigos e destes artigos selecionados, 22 foram lidos na íntegra e apenas 19 foram selecionados.

Figura 3 – Fluxograma (PRISMA) do processo de artigos de seleção dos artigos pesquisados



Fonte: Elaborado pelos autores (2024)

4. Resultados e Discussões

Nesta seção do artigo serão apresentadas as relações que o *Docker* possui com Segurança da Informação e sustentabilidade.

4.1. Segurança da Informação no *Docker*

A Segurança da Informação enfrenta muitos desafios nos ambientes virtuais. O *Docker*, uma plataforma de virtualização baseada em *containers*, fornece recursos que ajudam a melhorar a segurança dos aplicativos. Diferente da virtualização baseada em hipervisores, que cria VM completas, o *Docker* utiliza *containers* que compartilham o mesmo sistema operacional, permitindo maior densidade e melhor desempenho. Apesar disso, os *containers* mantêm um alto nível de segurança, mesmo com a configuração padrão (Bui, 2015).

Com base nas pesquisas feitas, os pontos apresentados na Tabela 1 destacam como o *Docker* pode fornecer segurança robusta em comparação com a virtualização tradicional baseada em hipervisor. Os *containers*, por serem leves e compartilharem o *kernel* do *host*, permitem uma gestão de segurança mais eficiente, ao mesmo tempo que mantém um isolamento forte entre as aplicações (Bui, 2015).

Tabela 1 – Diferenças entre *Docker* e VMs

Aspecto de Segurança	***Docker***	**VMs**
Isolamento de Aplicações	*Containers* são isolados entre si e do *host*, o que minimiza os riscos de interferência.	VMs são isoladas completamente, mas requerem hipervisores complexos para gerenciar esse isolamento.
Gerenciamento de Recursos	Usa *cgroups* para limitar recursos, prevenindo que um *container* consuma todos os recursos do *host*.	Hipervisores alocam recursos fixos para cada VM, o que pode levar a desperdício de recursos.
Atualizações de Segurança	Atualizações são aplicadas ao *host*, refletindo em todos os *containers*, facilitando a manutenção.	Cada VM requer atualizações individuais, aumentando a complexidade da gestão de *patches*.
Menor Superfície de Ataque	*Docker* tem uma superfície de ataque reduzida devido ao compartilhamento do *kernel* do *host*.	VMs possuem uma superfície de ataque maior, devido à necessidade de múltiplos *kernels*.
Controle de Acessos	Implementa controle granular de permissões e capacidades para cada *container*.	Controle de acessos depende das configurações de cada VM, o que pode ser mais complexo.
Restrições de Privilégios	Mesmo com acesso *root* dentro do *container*, os privilégios são restritos para mitigar danos.	*Root* nas VMs tem acesso total ao sistema, o que aumenta o risco caso comprometido.

Fonte: Elaborado pelos autores (2024) com base em Bui (2015) e Docker (2024)

Por exemplo, no *Docker*, mesmo que um invasor consiga acesso superusuário num *container*, os privilégios são limitados e não equivalem ao acesso superusuário no *host*. Isso é possível graças à implementação de capacidades restritas e ao mapeamento de usuários comuns fora do *container*, introduzido no *Docker* 1.10, que ajuda a mitigar os riscos de escape de *containers* (Docker, 2024).

Em resumo, ao adotar o *Docker*, as organizações podem aproveitar um modelo de segurança mais simples e eficiente, que facilita a manutenção e gestão de recursos, ao mesmo tempo em que proporciona um isolamento forte e reduz a superfície de ataque, resultando em uma Segurança da Informação mais robusta (Docker, 2024).

4.2. *Docker* e Sustentabilidade

O *Docker* está associado à sustentabilidade na área de Tecnologia da Informação devido ao seu uso eficiente de recursos físicos, o que resulta frequentemente em uma redução do gasto energético. Compartilhando o mesmo *kernel* e sistema operacional, o *Docker* otimiza a utilização dos recursos da máquina, resultando em uma pegada de carbono menor em comparação a outras formas de virtualização (Junior, 2023). Além disso, a rápida inicialização e desligamento dos *containers* contribuem para a eficiência energética ao reduzir o consumo contínuo de energia.

A eficiência no uso de *containers*, como no *Docker*, não só reduz resíduos eletrônicos ao aproveitar melhor o *hardware*, mas também promove a eficiência energética e escalabilidade. Isso se traduz em menor descarte de equipamentos eletrônicos e menor consumo de energia, já que menos servidores ativos são necessários. Além disso, os *containers* simplificam o empacotamento de aplicativos e suas dependências, otimizando a gestão de infraestrutura e evitando desperdícios de *hardware*. A utilização de *cgroups* pelo *Docker* para impor limites de recursos em *containers* garante um ambiente equilibrado e seguro, evitando a monopolização de recursos por uma única aplicação (Junior, 2023).

A Tabela 2 demonstra como o *Docker* pode oferecer uma solução mais sustentável em comparação com as VMs. O uso de *cgroups* permite ao *Docker* controlar e limitar recursos de forma eficiente, evitando que uma aplicação monopolize os recursos do sistema. De outra forma, as VMs requerem um sistema operacional completo para cada instância, resultando em maior consumo de recursos e energia (Junior, 2023).

Tabela 2 - Diferenças entre *Docker* e VMs de forma sustentável

Aspecto de Sustentabilidade	***Docker***	**Máquinas Virtuais**
Consumo de Recursos Físicos	Compartilha *kernel* e sistema operacional, otimizando recursos e reduzindo a pegada de carbono.	Cada VM requer um sistema operacional completo, resultando em maior consumo de recursos e energia.
Eficiência Energética	Inicialização e desligamento rápidos, menor consumo contínuo de energia.	Inicialização mais lenta, maior consumo de energia durante a operação.
Redução de Resíduos Eletrônicos	Menos *hardware* obsoleto devido à escalabilidade e eficiência.	Tendência a utilizar mais *hardware*, levando a maior geração de resíduos eletrônicos.
Escalabilidade Horizontal	Facilita a escalabilidade, reduzindo a necessidade de servidores ativos continuamente.	Requer mais servidores para escalabilidade, aumentando o consumo de energia.
Gestão de Infraestrutura	Simplifica a gestão e otimiza o uso de recursos disponíveis.	Gestão mais complexa, maior probabilidade de desperdício de recursos.
Limitação de Recursos	Utiliza *cgroups* para limitar CPU, memória e E/S, garantindo equilíbrio e segurança.	Menos controle granular, potencial para monopolização de recursos por uma VM.

Fonte: Elaborado pelos autores (2024) com base em Junior (2023).

O *Docker* oferece um ambiente de *containers* isolados que operam de forma eficiente e independente, resultando em menor consumo de energia em comparação com as VMs. A sua capacidade de escalabilidade horizontal reduz a necessidade de servidores ativos constantes, promovendo uma solução mais ecológica. Além disso, a gestão simplificada de infraestrutura e a redução de resíduos eletrônicos destaca-o como uma ferramenta valiosa para promover a sustentabilidade na área de TI, utilizando tecnologias de isolamento e gestão de recursos do *kernel* Linux. Em resumo, o *Docker* é uma alternativa superior às VMs em termos de eficiência energética, redução de resíduos eletrônicos e otimização de recursos (Junior, 2023).

5. Considerações Finais

Com base na análise das informações reunidas e expostas, é cabível afirmar que o *Docker* proporciona uma otimização do consumo de recursos computacionais, o que impulsiona a sustentabilidade ao facilitar a implementação de múltiplos microsserviços em um único servidor ou em servidores virtualizados. Tal eficiência concorre para a redução do consumo energético, tanto por meio da diminuição do uso dos servidores propriamente ditos, quanto pela mitigação da exigência de sistemas de refrigeração de alta capacidade.

Além disso, o uso de *Docker* pode reduzir o descarte de *hardware* obsoleto, pois diminui a necessidade de novos servidores físicos, resultando em um ambiente de TI mais enxuto e controlado. Dessa forma, tem como resultado um impacto ambiental reduzido, devido à menor produção de resíduos eletrônicos e à diminuição da demanda por novos equipamentos.

No que diz respeito à Segurança da Informação, o *Docker* implementa diversas medidas para assegurar a proteção dos *containers*. Essas práticas de segurança englobam um isolamento eficaz e a aplicação de privilégios mínimos durante a execução dos *containers*, contribuindo para a redução da superfície de ataque e a preservação da integridade do sistema. A documentação oficial do *Docker* oferece diretrizes sobre as melhores práticas de segurança e sustentabilidade, capacitando os usuários a manterem seus sistemas atualizados e seguros. Isso garante que, mesmo em ambientes complexos, seja viável manter a eficiência e a segurança.

Por fim, ao utilizar *Docker*, as organizações podem melhorar a gestão e operação de seus recursos de TI e contribuir significativamente para a sustentabilidade ambiental. A utilização de *containers* permite uma alocação mais eficiente de recursos, minimizando o desperdício e maximizando a utilização do *hardware* disponível. Com menos necessidade de *hardware* adicional, há uma diminuição na produção de resíduos eletrônicos e na demanda por novos equipamentos. Além disso, a menor necessidade de energia para operação e refrigeração dos servidores resulta em uma pegada de carbono reduzida. Assim, o uso inteligente da tecnologia através do *Docker* não só melhora a eficiência operacional, mas também demonstra um compromisso das organizações com a responsabilidade ambiental e a sustentabilidade a longo prazo. Esse compromisso é cada vez mais valorizado no mercado atual, onde consumidores e *stakeholders* estão mais conscientes e exigentes em relação às práticas ambientais das empresas.

Referências

ABSAM BLOG. **Vantagens de utilizar Docker para montar seu ambiente.** 2024. Disponível em: https://absam.io/blog/vantagens-de-utilizar-docker-para-montar-seu-ambiente/. Acesso em: 10 maio 2024.

ALMEIDA, A. **Virtualização**. 2011. 57 f. Dissertação (Mestrado) - Faculdade de Engenharia da Universidade do Porto, Porto, 2011. Disponível em: https://repositorio-aberto.up.pt/bitstream/10216/61585/1/000148684.pdf. Acesso em: 3 dez. 2023.

ALTEXSOFT. **The Good and the Bad of Docker Containers.** 2022. Disponível em: https://www.altexsoft.com/blog/docker-pros-and-cons/. Acesso em: 5 maio 2024.

BUI, T. **Analysis of Docker Security**. 2015. Disponível em: https://arxiv.org/abs/1501.02967. Acesso em: 3 dez. 2023.

DOCKER. **Use Containers to Build, Share and Run your applications**. 2024. Disponível em: https://www.docker.com/resources/what-container/. Acesso em: 5 maio 2024.

GOMES. R. **Docker para Desenvolvedores**. Salvador: Instruct, 2020. 174 p. Disponível em: https://leanpub.com/dockerparadesenvolvedores. Acesso em: 6 dez. 2023.

JUNIOR, V. **Como funciona o Isolamento de containers utilizado pelo Docker**. 2023. LinkedIn. Disponível em: https://www.linkedin.com/pulse/como-funciona-o-isolamento-de-containers-utilizado-pelo-valdir-junior/?originalSubdomain=pt. Acesso em: 4 dez. 2023.

MENDONÇA, P. A. S. **Virtualização, Containers e Docker**. 2019. Disponível em: https://pauloandresoares.com/2019/10/29/virtualizacao-containers-e-docker.html. Acesso em: 2 mai. 2024.

MORAES, J. M. de; QUIRINO, C.; ALMEIDA, R. M. de; NEVES, J. E. D. **Internet das Coisas (IoT)**: Casa inteligente, definições e aplicações. Revista Brasileira em Tecnologia da Informação, [S. l.], v. 4, n. 2, p. 31 - 37, 2022. Disponível em: https://www.fateccampinas.com.br/rbti/index.php/fatec/article/52. Acesso em: 21 maio. 2024.

MOURA, T. M.; D' ALKMIN NEVES, J. E. **Análise de Segurança em Dispositivos Internet das Coisas**. Revista Interface Tecnológica, [S. l.], v. 18, n. 2, p. 15–27, 2021. Disponível em: https://doi.org/10.31510/infa.v18i2.1174. Acesso em: 3 jun 2024.

NEVES, J. E. D. **Estudo dos parâmetros do modelo de Mason para cerâmicas piezelétricas utilizando algoritmos genéticos**. Dissertação (Mestrado em Tecnologia) - Faculdade de Tecnologia, Universidade Estadual de Campinas. Limeira, p. 129. 2018. Disponível em: https://doi.org/10.47749/T/UNICAMP.2018.995710. Acesso em: 12 maio 2024.

NEVES, J. E. D. **Modelo Baseado em Agentes para Simulação de Consumo de Energia Elétrica em Função do Comportamento Humano**. Revista Eletrônica Anima Terra, v. 12, 2021. Disponível em: https://fatecmogidascruzes.com.br/pdf/animaTerra/edicao12/artigo7.pdf. Acesso em: 10 maio 2024.

NEVES, J. E. D.; PEDRO, P. S. M.; HERNANDEZ, M. F. G.; FABRI JUNIOR, L. A. **Simulation of the Implementation of Domestic Solar Systems Using Multi-agent Systems from Web Scraping**. Smart Innovation, Systems and Technologies. 1ed.: Springer International Publishing, 2023, v. 1, p. 88-96. Disponível em: https://doi.org/10.1007/978-3-031-04435-9_8. Acesso em: 10 de maio 2024.

PEDRO, A. M.; TURCI JUNIOR, M.; MONTEIRO, A. S.; ESPERANDIO, A. A. M.; BASTOS, C. V.; NEVES, J. E. D. **Blockchain como Fator de Transparência**. Revista Brasileira em Tecnologia da Informação, v. 5, p. 79-95, 2024. Disponível em: https://www.fateccampinas.com.br/rbti/index.php/fatec/article/104. Acesso em: 30 maio 2024.

RAD, B. B. AHMADI, M; BHATTI, H. J.; **An Introduction to Docker and Analysis of its Performance**. 2017. Disponível em: https://www.researchgate.net/publication/318816158_An_Introduction_to_Docker_and_Analysis_of_its_Performance. Acesso em: 6 dez. 2023.

SANTOS, B. S.; FONSECA, L. M. B.; SERNAGLIA, L.; NEVES, J. E. D. **Automação de casas e estabelecimentos comerciais através de microcontroladores**. REVISTA TECNOLÓGICA DA FATEC DE AMERICANA, v. 8, p. 70-80, 2020. Disponível em: https://ric.cps.sp.gov.br/handle/123456789/6732. Acesso em: 20 maio 2024.

SERRA JUNIOR, F. C.; LIMA, B. S. N. M. **Tecnologia Docker**. 2020. Disponível em: https://www.academia.edu/42175912/TECNOLOGIA_DOCKER. Acesso em: 5 dez. 2023.

SOUZA, A. L. O.; BASTOS, C. V.; SANTOS, P. M. S.; SOARES, N. M.; NEVES, J. E. D. **Cibersegurança na Agricultura de Precisão: Exploração à Aplicação de Medidas Preventivas**. Advances in Global Innovation & Technology, v. 2, p. 61-73, 2024. Disponível em: https://doi.org/10.29327/2384439.2.2-5. Acesso em: 20 de maio 2024.

VERTIGO. **Containers vs Máquinas Virtuais**. 2019. Disponível em: https://vertigo.com.br/containers-vs-maquinas-virtuais/. Acesso em 10 jan. 2024.

VICENTINE, A. L.; SILVA, G. C. M.; NASCIMENTO, J. P. D. G.; NEVES, J. E. D. Software anti-cheat. Revista Brasileira em Tecnologia da Informação, v. 4, p. 1-10, 2022. Disponível em: https://www.fateccampinas.com.br/rbti/index.php/fatec/article/54. Acesso em: 12 maio 2024.

WALDSPURGER, C. A. **Memory Resource Management in VMware ESX Serv. Boston: Usenix**, 2002. 15 p. Disponível em: https://www.usenix.org/legacy/event/osdi02/tech/waldspurger/waldspurger.pdf. Acesso em: 6 dez. 2023.