Estratégia Concursos

Aula 00

UnDF (Professor e Tutor - Arquitetura de Sistemas da Computação - Código 131 e 331) Desenvolvimento de Software - 2022 (Pós-Edital)

Autor:
Diego Carvalho

29 de Junho de 2022

Índice

NOÇÕES DE DEVOPS

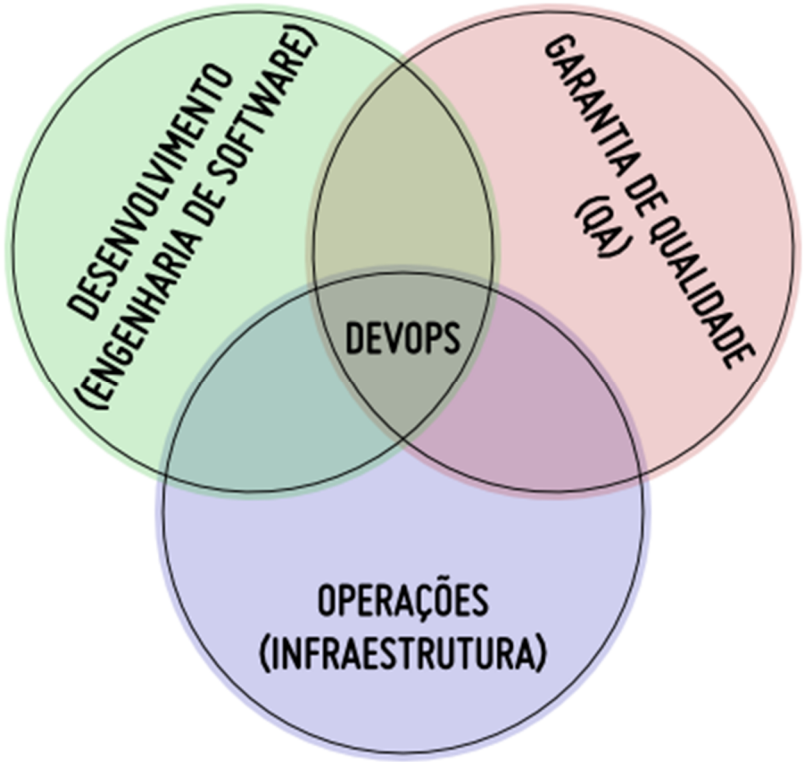
Conceitos Básicos

INCIDÊNCIA EM PROVA: MÉDIA

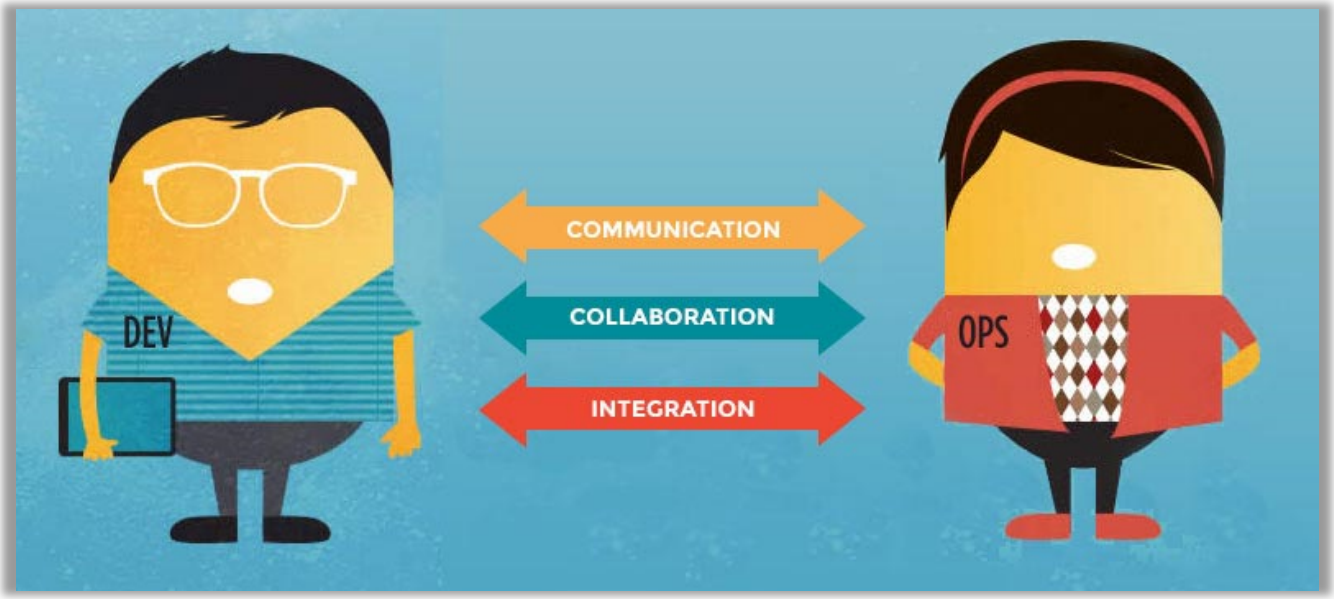
De um tempo para cá, nós temos vivenciado novos paradigmas em tecnologias de software. **As metodologias ágeis, por exemplo, vieram com um conjunto de práticas que desencadearam diversas outras práticas, de forma a obter software de maneira mais ágil e mais adaptável a possíveis mudanças**. No entanto, o grande lance é que as mudanças ocorrem com um intervalo de tempo cada vez menor.

Vocês já devem ter percebido isso! Antigamente, softwares lançavam atualizações em intervalos de 18 a 24 meses (ou até mais). **Atualmente, a dinâmica de consumo de aplicações de tecnologia da informação sofreu uma reviravolta e a demanda dos clientes é insana.** As empresas de software atualmente são duramente pressionadas para lançar e atualizar aplicativos no mercado o mais rápido possível.

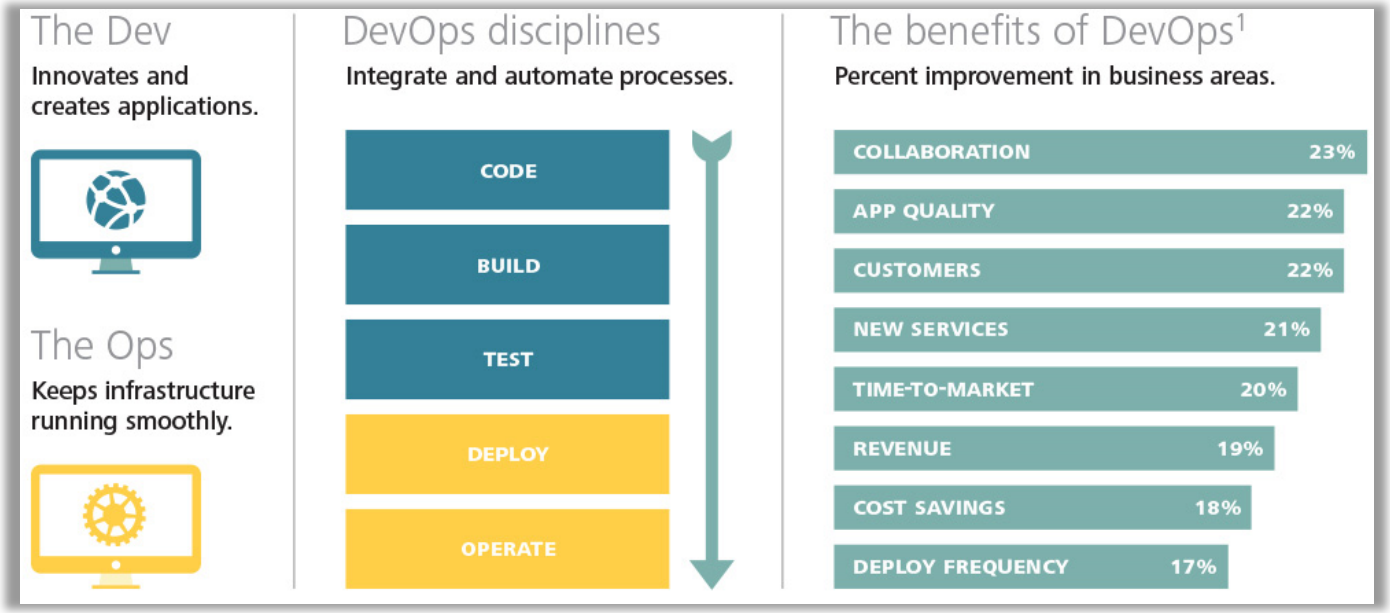
Pois é, o mundo mudou! O ciclo para a criação de novos aplicativos de software dura cerca de três meses para uma versão inicial e mais seis meses para o conjunto completo de recursos. **Não só o ciclo de vida foi encurtado, mas os aplicativos se tornaram muito mais complexos e exigem colaboração e integração cruzada entre os diversos componentes de tecnologia da informação, como Dev, Ops e Q&A.**



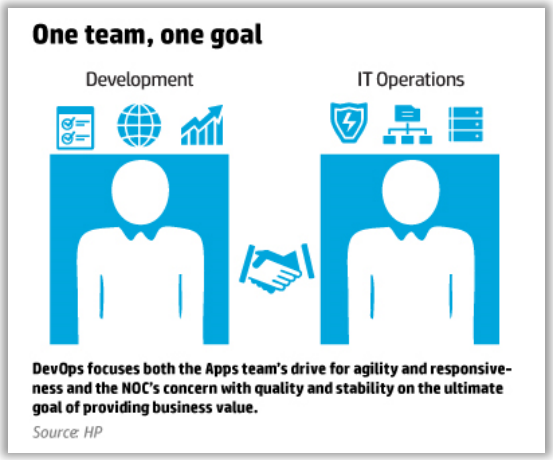
Professor, espera um pouco aí! O que acontece se eu juntar conceitos de Desenvolvimento de Software, Operação de Sistemas e Garantia de Qualidade? Surgirá, então, o conceito de DevOps! **Trata-se de um conceito muito simples, como apresenta a imagem acima. Essas ideias são tratadas em conjunto, em vez de separadas.** O lance é ter uma maior comunicação, colaboração e integração entre essas áreas.



Professor, o que é exatamente DevOps? É uma cultura? É uma técnica? É uma metodologia de desenvolvimento de software? Ainda não existe uma resposta precisa para essas perguntas! *Por que, professor?* **Porque foi um movimento que começou ao mesmo tempo em diversos lugares diferentes, tratando de infraestrutura e desenvolvimento, mas não houve um manifesto formal, como o manifesto ágil.**



Parece simples fazer a infraestrutura conversar de forma harmônica com o desenvolvimento, mas não é tão fácil! *Qual é o papel da infraestrutura?* É sustentar os sistemas em produção; monitorar o funcionamento e a performance; cuidar da estabilidade, segurança, níveis de serviço; planejar mudanças, minimizando riscos; entre outros. *O que acontece se uma aplicação em produção parar de funcionar?*



Isso pode significar prejuízo financeiro ou institucional. Em suma, podemos dizer que a infraestrutura se preocupa em proteger o valor de negócio. *E o desenvolvimento?* Esses caras se preocupam com inovação e criatividade, baseado nos requisitos do usuário. Desenvolvedor fica louco quando sai uma biblioteca, componente ou tecnologia nova. A consequência disso é que cada inovação significa um novo Deploy (feito pela rapaziada da infraestrutura).

E se ocorrer algum problema? Deve ser realizado um rollback (também pelo pessoal da infraestrutura). **Podemos afirmar, então, que o desenvolvedor se preocupa em aumentar o valor do negócio.** Vocês já devem ter notado que há um conflito interessante nessa conversa. Ora, o desenvolvedor quer colocar suas aplicações no ar o mais rápido possível para que fique disponível para o cliente.

No entanto, a galera da infraestrutura quer ter certeza de que a aplicação está suficientemente estável para ir para produção sem gerar incidentes que parem o que já está funcionando. *O que ocorria antigamente?* As empresas permitiam apenas um deploy por semana ou por mês! *Tem coisa mais não-ágil que isso?* Vai totalmente de encontro aos ideais do manifesto ágil. *Lembram-se dos conceitos de entrega, integração e teste contínuos?*

Pois é, a infraestrutura teve que se adaptar a realizar deploys diários. No entanto, os desenvolvedores – muitas vezes – se esqueciam de considerar algumas diferenças importantes entre ambientes de desenvolvimento e produção. **Isso gerava alguns incidentes, o cliente reclamava e começava uma briga muito comum representada pelas duas imagens anteriores.** Sim, galera... rola essa briga!

Desenvolvedores afirmando que a Infraestrutura é engessada, lenta e que não oferece um ambiente adequado para o desenvolvimento de aplicações; já a Infraestrutura afirmava que os desenvolvedores faziam código ruim e instável, e que a culpa não era deles. **Pessoal, voltem agora para a primeira imagem e percebam o que DevOps tenta integrar: Desenvolvimento, Infraestrutura e Qualidade!**

Parece briga de marido e mulher, mas ambos os lados têm que reconhecer seus erros, ceder e se adaptar! **O Desenvolvimento precisa pensar mais na Infraestrutura e controlar as fases de deploy (Ex: Deployment Pipeline). Já a Infraestrutura tem que evoluir para o mundo ágil.** Começar a trabalhar de forma automatizada e dinâmica, ser mais veloz para subir ambientes; reconstruir ou duplicar ambientes de acordo com as necessidades do Desenvolvimento.

Galera, as principais características do DevOps são: colaboração entre equipes; fim de divisões; relação saudável entre áreas; teste, integração e entrega contínuos; automação de deploy; controle e monitoração; gerenciamento de configuração; orquestração de serviços; avaliação de métricas e desempenho; logs e integração; velocidade de entrega; feedback intenso; e comunicação constante.

Em suma, podemos dizer que ele é um movimento, um conceito, uma cultura, uma abordagem que trata do feedback, comunicação e colaboração entre áreas de tecnologia da informação com o objetivo de garantir qualidade do software, com menor custo, mais rapidez, menor risco e maior eficiência. **É como se fosse a utilização de metodologias ágeis no desenvolvimento e na infraestrutura.** *Bacana?*

Para o DevOps, o código gerado pela infraestrutura é apenas mais um artefato qualquer de desenvolvimento e, não, uma parte separada. Trata-se de uma prática importante, já que não basta somente o desenvolvedor escrever o código do sistema, é necessário que a área de infraestrutura também atue para liberar, controlar e entregar a versão do sistema de forma contínua, periódica e preferencialmente automática.

TODAY'S DEVOPS

DEV + OPS

PART DEV

Application Performance

Modern developers use APM tools to decrease latency, have complete visibility into code, databases, caches, queues, and third-party services.

End User Analytics

A great developer understands end users have the best feedback and analytics play an enormous part of understanding users. Developers are constantly monitoring end user latency and checking performance by devices and browsers.

Quality Code

Developers need to ensure their deployments and new releases don't implode or degrade the overall performance.

Code-Level Errors

When you have a large distributed application it is vital to lower MTTR by finding the root cause of errors and exceptions.

PART OPS

Application Availability

The applications need to be up and running and it's Ops responsibility to ensure uptime and SLAs are in order.

Application Performance

Classic Ops generally rely on infrastructure metrics - CPU, memory, network and disk I/O, etc. Modern Ops correlate all of those metrics with application metrics to solve problems 10x faster.

End User Complaints

The goal is to know about and fix problems before end users complain, reduce the number of support tickets, and eliminate false alerts.

Performance Analytics

Automatically generated baselines of all metrics help Ops understand what has changed and where to focus their troubleshooting efforts. Alerts based upon deviation from observed baselines improve alert quality and reduce alert noise.

Vamos resumir: **a preocupação é com os objetivos quase diametralmente opostos da galera de Desenvolvimento (DEV) e Operações (OPS).** Os desenvolvedores querem programar novos recursos, melhorar o produto; corrigir bugs, etc. As operações querem colocar tudo em funcionamento e nunca mudar, já que as alterações causam novos erros, bugs, problemas de desempenho, entre outros.

O objetivo do DevOps é aliviar a tensão entre esses dois campos! **Ter pessoas de Operações nas trincheiras de Desenvolvimento é a principal maneira de alcançar esse objetivo.** Seu trabalho é facilitar ao máximo que desenvolvedores e operações façam o que precisam fazer. *Como assim?* Por exemplo: fornecendo ambientes idênticos de desenvolvimento, testes, homologação, *staging* e qualidade; configuração de pipelines de teste e implementação automáticos.

Além de estar envolvido no processo de desenvolvimento para que eles estejam mais preparados para lidar com erros de produção. **Tradicionalmente, as operações são quase alheias à base de código, uma vez que se preocupam apenas com sua infraestrutura.** O objetivo é tornar todo o processo transparente de forma que você possa fazer milhares de implantações de produção por dia; ao contrário dos métodos tradicionais de configuração de uma janela de implantação uma vez a cada trimestre ou por mais tempo.

Claro que para fazer isso, é necessário utilizar um conjunto de práticas e ferramentas. *Quais, professor?* Ferramentas de Controle de versão (Git, CVS, Tortoise), Servidores de Integração Contínua (Jenkins, Bamboo, Travis), Docker/Vagrant, Gerenciamento de Configuração (SaltStack, Chef, Puppet). Isso reduz a dor de cabeça tanto para os desenvolvedores quanto para os operadores e reduz a quantidade de problemas de desempenho e erros de codificação.

(STF – 2013) Integração contínua, entrega contínua, teste contínuo, monitoramento contínuo e feedback são algumas práticas do DevOps.

_____

Comentários: as principais características do DevOps são: colaboração entre equipes; fim de divisões; relação saudável entre áreas; teste, integração e entrega contínuos; automação de deploy; controle e monitoração; gerenciamento de configuração; orquestração de serviços; avaliação de métricas e desempenho; logs e integração; velocidade de entrega; feedback intenso; e comunicação constante (Correto).

(STF – 2013) Teste contínuo é uma prática do DevOps que, além de permitir a diminuição dos custos finais do teste, ajuda as equipes de desenvolvimento a balancear qualidade e velocidade.

_____

Comentários: teste, integração e entrega contínuos são realmente práticas do DevOps que permitem reduzir custos de teste e ajuda a equipe de desenvolvimento a balancear a qualidade e velocidade (Correto).

(TCU – 2015) De acordo com a abordagem DevOps (development – operations), os desafios da produção de software de qualidade devem ser vencidos com o envolvimento

dos desenvolvedores na operação dos sistemas com os quais colaboraram no desenvolvimento.

_____

Comentários: essa questão permite duas interpretações de 'envolvimento': (1) no sentido de interação e colaboração entre equipes; (2) no sentido de mão na massa mesmo - na operação dos sistemas. Por conta da ambiguidade, a questão foi anulada (Anulada).

(TRE-PE – 2017) O DevOps consiste em:

a) um processo similar ao IRUP (IBM Rational Unified Process), que tem como objetivo dividir o processamento em fases e disciplinas de software para paralelizar as ações de desenvolvimento e de manutenção das soluções.

b) uma plataforma aberta cuja função é substituir a virtualização de aplicações e serviços em containers e, com isso, agilizar a implantação de soluções de software.

c) um aplicativo que permite o gerenciamento de versões de códigos-fonte e versões de programas, bem como a implantação da versão mais recente de um software em caso de falha.

d) um processo de promoção de métodos que objetivam aprimorar a comunicação, tornando a colaboração eficaz especialmente entre os departamentos de desenvolvimento e teste e entre os departamentos de operações e serviço para o negócio.

e) uma metodologia ágil que, assim como a XP (extreme programming) e o Scrum, tem foco na gestão de produtos complexos relativos à equipe de desenvolvimento.

_____

Comentários: DevOps não é um processo similar ao iRUP, não é uma plataforma aberta, não é um aplicativo e não é uma metodologia ágil. DevOps é um processo de promoção de métodos que objetivam aprimorar a comunicação, tornando a colaboração eficaz especialmente entre os departamentos de desenvolvimento e teste e entre os departamentos de operações e serviço para o negócio (Letra D).

(TRT-PA e AP – 2016) Acerca de DevOps, assinale a opção correta.

a) O DevOps concentra-se em reunir diferentes processos e executá-los mais rapidamente e com mais frequência, o que gera baixa colaboração entre equipes.

b) O DevOps tem como princípio produzir, a partir da avaliação dos times de desenvolvimento do serviço, grandes mudanças e farta documentação com valor agregado para os usuários, assemelhando-se, por isso, com objetivos dos métodos iterativos e em cascata.

c) A infraestrutura de nuvem de provedores internos e externos vem restringindo o uso de DevOps pelas organizações.

d) O DevOps parte da premissa de adoção de grandes equipes de especialistas, com a menor interação possível, visando à padronização de processos e à mínima automação de atividades.

e) Atividades típicas em DevOps compreendem teste do código automatizado, automação de fluxos de trabalho e da infraestrutura e requerem ambientes de desenvolvimento e produção idênticos.

_____

Comentários: (a) Errado, isso gera alta colaboração entre equipes; (b) Errado, não se assemelha com objetivos de métodos em cascata, visto que essa metodologia possui entregas menos frequentes e é menos dinâmica; (c) Errado, é justamente o inverso – elas usam com cada vez mais frequência; (d) Errado, recomenda-se a maior interação possível entre as equipes; (e) Correto, testes automatizados, automação de fluxos e infraestrutura são atividades frequentes que realmente exigem ambientes de desenvolvimento e produção idênticos (Letra E).

(STJ – 2015) DevOps é um conceito pelo qual se busca entregar sistemas melhores, com menor custo, em menor tempo e com menor risco.

_____

Comentários: perfeito, perfeito, perfeito – todas são características do DevOps (Correto).

(STJ – 2015) O profissional especialista em DevOps deve atuar e conhecer as áreas de desenvolvimento (engenharia de software), operações e controle de qualidade, além de conhecer, também, de forma ampla, os processos de desenvolvimento ágil.

_____

Comentários: ele é a combinação entre desenvolvimento, operação e controle de qualidade – portanto questão perfeita (Correto).

(SLU-DF – 2019) Em DevOps, o princípio monitorar e validar a qualidade operacional antecipa o monitoramento das características funcionais e não funcionais dos sistemas para o início do seu ciclo de vida, quando as métricas de qualidade devem ser capturadas e analisadas.

_____

Comentários: o princípio monitorar e validar a qualidade operacional transfere o monitoramento para o início do ciclo de vida do software, para identificar antecipadamente problemas de qualidade que podem ocorrer no desenvolvimento. Na implantação e teste, as métricas de qualidade são capturadas e analisadas, sendo que essas métricas devem ser entendidas por todos os stakeholders (Correto).

(MPE-PI – 2018) A infraestrutura como código é uma prática DevOps caracterizada pela infraestrutura provisionada e gerenciada por meio de técnicas de desenvolvimento de código e de software, como, por exemplo, controle de versão e integração contínua.

_____

Comentários: trata-se realmente de uma prática em que a infraestrutura provisionada e gerenciada por meio de técnicas de desenvolvimento de código e de software – ela oferece controle de versão, entrega e integração contínua (Correto).

(STJ – 2018) Apesar de ser um processo com a finalidade de desenvolver, entregar e operar um software, o DevOps é incompatível com a aplicação de métodos ágeis como o Scrum ou, ainda, com o uso de ferramentas que permitam visualizar os fluxos do processo.

_____

Comentários: pelo contrário, é completamente compatível com as aplicações de métodos ágeis como o Scrum ou, ainda, com o uso de ferramentas que permitam visualizar os fluxos do processo (Errado).

(STJ – 2018) O gerenciamento de desenvolvimento de software por meio do Scrum pode ser combinado com o ciclo de vida do DevOps, haja vista que o DevOps combina práticas e ferramentas que aumentam a capacidade de uma organização de distribuir aplicativos e serviços; logo, a integração contínua do software pode ser realizada na sprint do Scrum junto com a operação dos serviços da organização.

_____

Comentários: ele realmente combina práticas e ferramentas que aumentam a capacidade de uma organização de distribuir aplicativos e serviços e pode ser realizado na sprint do Scrum por meio da integração contínua (Correto).

DOCKER

Conceitos Básicos

INCIDÊNCIA EM PROVA: MÉDIA

Caros alunos (as), o Docker é um software usado como contêiner da empresa Docker, Inc, que **fornece uma camada de abstração e automação para virtualização de sistema operacional no Windows e no Linux**.

Como ele faz isso? Segundo o fornecedor: *o Docker emprega isolamento de recurso do núcleo do Linux e espaços de nomes do núcleo, e um sistema de arquivos com recursos de união, como OverlayFS criando contêineres independentes para executar dentro de uma única instância do sistema operacional, evitando a sobrecarga de manter máquinas virtuais (VM).*

O docker então é uma **alternativa de virtualização** em que o kernel da máquina hospedeira é compartilhado com a máquina virtualizada ou o software em operação, portanto um desenvolvedor pode agregar a seu software a possibilidade de levar as bibliotecas e outras dependências do seu programa junto ao software com menos perda de desempenho do que a virtualização do hardware de um servidor completo. **Assim, o docker torna operações em uma infraestrutura como serviços web mais intercambiável, eficientes e flexíveis.**

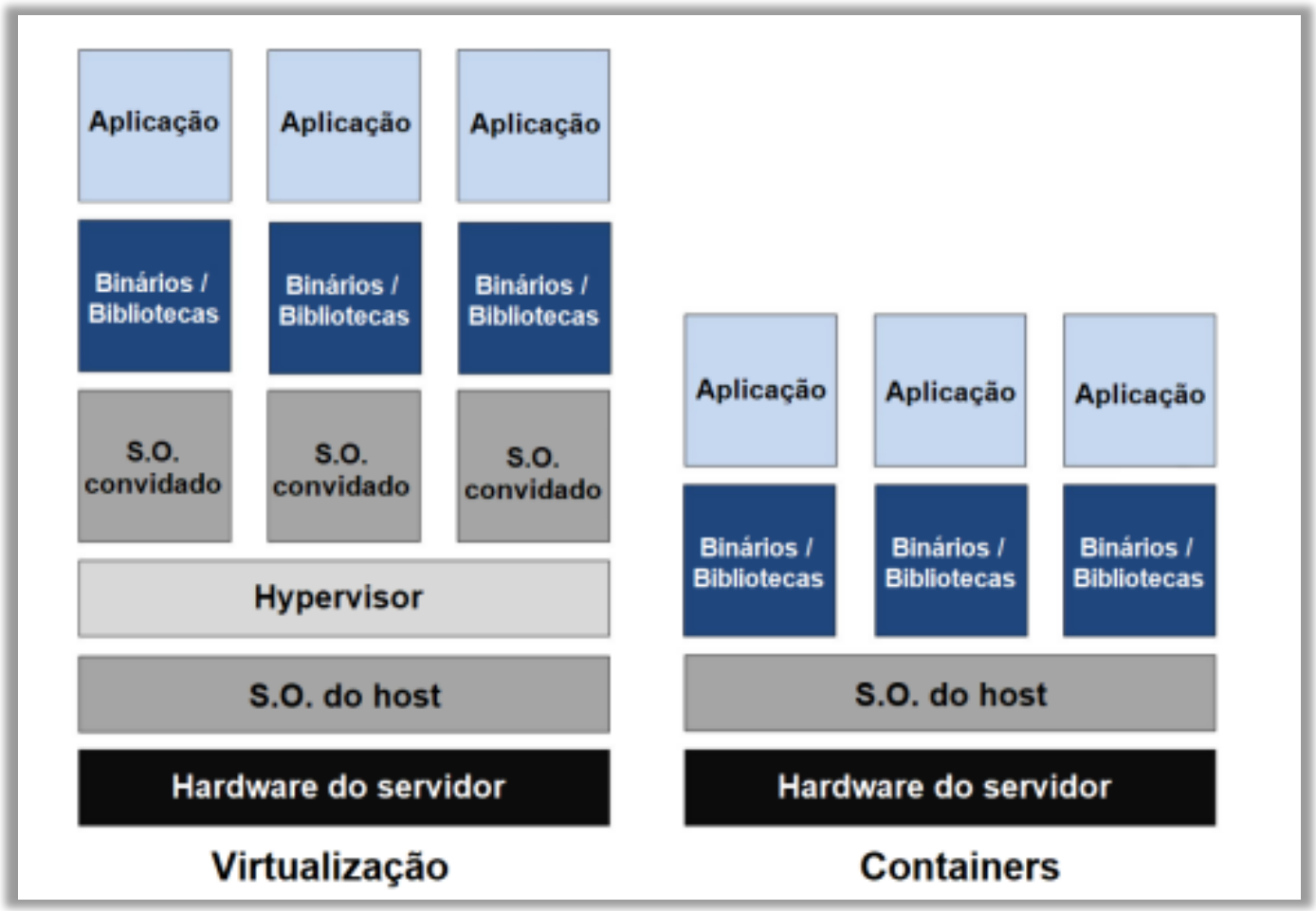
Na prática o docker é utilizado como uma ferramenta que pode **empacotar um aplicativo e suas dependências em um recipiente virtual que pode ser executado em qualquer servidor que contenha o Docker instalado**. Isso ajuda a permitir flexibilidade e portabilidade de onde o aplicativo pode ser executado, quer nas instalações, nuvem pública, nuvem privada, entre outros.
Então o que exatamente o Docker faz e por que utilizamos ele? Docker é uma plataforma que utiliza containers como solução arquitetural, para aumentar a portabilidade e eficiência. Perai professor eu ainda não sei o que são containers.

O QUE SÃO, DE ONDE VIERAM E POR QUE USAMOS CONTAINERS?

Pessoal, um container é um ambiente isolado, que representa um conjunto de processos executados a partir de uma imagem, a imagem é o local onde estão todos os arquivos necessários para execução do container, logo os containers compartilham o kernel e isolam os processos das aplicações do resto do sistema.

Por exemplo: se você está desenvolvendo uma aplicação para um cliente, você pode fazer suas configurações nessa aplicação. Mas, em um ambiente convencional, você precisará replicar estas configurações para os outros ambientes de execução. Ok?!

Com o Docker, você pode ir fazendo isso em um ambiente isolado e, por causa da facilidade para replicação de containers, você acaba criando ambientes padronizados, tanto em desenvolvimento como em produção, por exemplo. Assim, você pode disponibilizar essa arquitetura toda para seu cliente, onde ele estiver: basta replicar os containers, que serão executados da mesma maneira em qualquer lugar. Logo, podemos já derivar uma diferença fundamental entre Containers e Virtualização, veja:

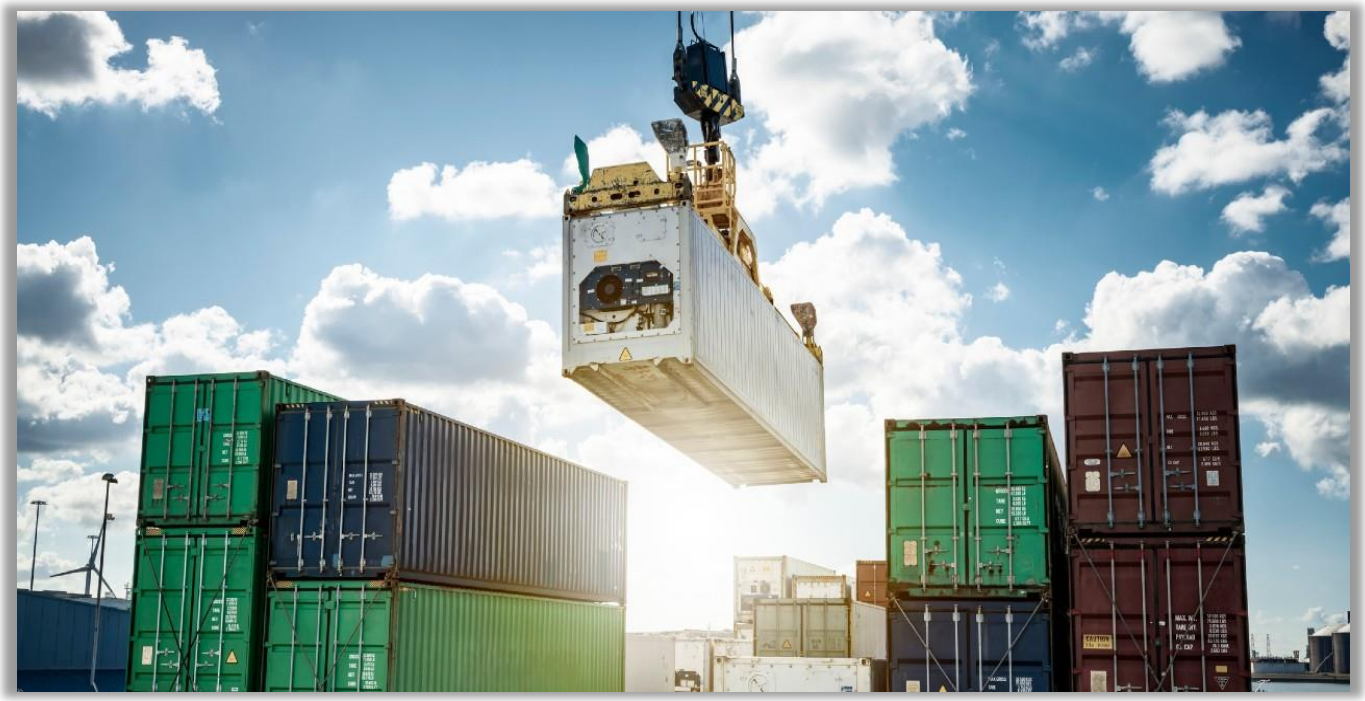


Como o container possui uma imagem que **contém todas as dependências de um aplicativo, ele é portátil** e consistente em todas as etapas de desenvolvimento. **Essa imagem é um modelo de**

somente leitura que é utilizada para subir um container. O Docker nos permite construir nossas próprias imagens e utilizá-las como base para os containers.

Pessoal, Docker é então uma **plataforma open source escrito em Go**, que é uma linguagem de programação de alto desempenho desenvolvida dentro do Google, que facilita a criação e administração de ambientes isolados. O Docker possibilita o empacotamento de uma aplicação ou ambiente inteiro dentro de um container, **e a partir desse momento o ambiente inteiro torna-se portável para qualquer outro host que contenha o Docker instalado**. O Docker pode ler instruções através de um arquivo texto que contém instruções para montar uma imagem (**dockerfile**).

Isso **reduz drasticamente o tempo de deploy** (implantação) de alguma infraestrutura ou até mesmo aplicação, pois não há a necessidade de ajustes de ambiente para o correto funcionamento do serviço. O ambiente é sempre o mesmo, basta configurar uma vez e replicar quantas vezes quiser! Outra facilidade do Docker é poder criar suas imagens (containers prontos para deploy) a partir de arquivos de definição, os chamados **Dockerfiles**.



Outro fator importante é que o Docker utiliza como backend padrão o LXC (Linux Containers), com isso é possível definir limitações de recursos por container (memória, CPU, E/S etc.). O LXC é um método de virtualização a nível de sistema operacional que permite executar múltiplos Sistemas Linux (denominados containers) usando um único kernel. Pessoal, importante ressaltar um dos principais comandos do Docker, que serve para visualizar informações referentes ao consumo de recursos de todos os containers, deve-se executar no terminal: **Docker container stats.**

Assim, pode-se criar ambientes de teste e/ou produção utilizando o LXC de forma ágil e segura (isolada). **Não há a necessidade de se preocupar com tantos detalhes como na virtualização tradicional.**

Quanto as instruções Docker, uma se destaca por ser responsável por indicar ao docker a porta que o container deve utilizar em tempo de execução, trata-se da EXPOSE. Outras instruções importantes, seriam:

- **ADD:** Adiciona um arquivo ou diretório dos sistemas de arquivo local para a imagem;
- **COPY:** Copia arquivos remotos e/ou locais para a imagem.
- **CMD:** Comando padrão a ser executado pela imagem;
- **ENTRYPOINT:** Permite configurar o contêiner ou apenas definir o comando a ser executado (Sobrepõe o CMD);
- **ENV:** Define variáveis de ambiente;
- **FROM:** Inicia a imagem a partir de outra imagem: Ex "FROM debian:8";
- **RUN:** Roda um comando no sistema operacional da imagem;
- **ARG:** Define variáveis de ambiente, mas permite que no momento da construção da imagem seja passado o valor para a variável especificada. Útil para quando se deseja permitir que o usuário construa imagens para mais de uma versão do mesmo software usando o mesmo DockerFile.

Como o Docker trabalha utilizando cliente e servidor (**toda a comunicação entre o Docker Daemon e Docker client é realizada através de API**), basta apenas ter instalado o serviço do Docker em um lugar, e apontar em o Docker Client para esse servidor. A plataforma do Docker em si utilizada alguns conjuntos de recursos, seja para a criação ou administração dos containers.

Entre esses conjuntos podemos destacar a biblioteca **libcontainer**, que é responsável pela comunicação entre o Docker Daemon e o backend utilizado, sendo ela a responsável pela criação do container, e através dela é possível configurar os limites de recursos por container.

IMPORTANTE

Vale lembrar que, apesar do Docker ter sido desenvolvido inicialmente com base na tecnologia LXC (Linux Containers – sendo, portanto, mais associado aos containers Linux), hoje essa tecnologia tornou-se independente de sistema operacional: podemos utilizar o Docker em ambientes Linux, Windows e até mesmo MacOS.

(CRA-PR - 2022) Em um sistema Linux Ubuntu com o Docker instalado, para visualizar informações referentes ao consumo de recursos de todos os containers, deve-se executar no terminal: Docker container stats.

_____

Comentários: Perfeito! Conforme vimos em aula, o Docker container stats mostra os recursos de todos os containers, logo em um sistema Linux Ubuntu com o Docker instalado, para visualizar informações referentes ao consumo de recursos de todos os containers, deve-se executar no terminal: Docker container stats (Correto).

QUESTÕES COMENTADAS – DOCKER

1. **(FEPESE / Prefeitura de Florianópolis - 2019)** O Docker pode ler instruções através de um arquivo texto que contém instruções para montar uma imagem (dockerfile). Nesse contexto, qual a palavra-chave ou instrução que indica ao docker a porta que o container deve utilizar em tempo de execução?

   a) HTTP
   b) PORT
   c) EXPOSE
   d) DOCKER
   e) SERVERPORT

Comentários:

Galera, a instrução que indica ao Docker a porta que o container deve utilizar em tempo de execução é o EXPOSE. Conforme vimos: "Quanto as instruções do Docker, uma se destaca por ser responsável por indicar ao docker a porta que o container deve utilizar em tempo de execução, trata-se da EXPOSE".

Gabarito: Letra C

---

2. **(COMPERVE / TJ-RN - 2020)** Os volumes são mecanismos utilizados para persistir os dados gerados e usados pelos containers do Docker. Embora as montagens de ligação dependam da estrutura de diretórios da máquina *host*, os volumes são completamente gerenciados pelo Docker. Considerando que um analista queira criar um volume de nome *my-volume* dentro de um docker, ele deve executar o comando:

   a) docker volume create my-volume
   b) docker create volume my-volume
   c) docker run create volume my-volume
   d) docker create run volume my-volume

Comentários:

Pessoal, caso a ideia seja criar um volume de nome "my-volume" a sintaxe no Docker deve ser: docker volume create my-volume. Bem simples, né?

Gabarito: Letra A

---

3. **(AOCP / MJSP - 2020)** O Docker possibilita que uma imagem com todos os aplicativos e configurações realizadas em um contêiner sejam transferidos para outro host, bastando que este tenha o Docker instalado. Assinale a alternativa que apresenta o nome dessa operação:

a) Restore.
b) Dump.
c) Drill down Contêiner.
d) Portabilidade.
e) Drill up Contêiner.

Comentários:

Trata-se da operação de Portabilidade, quando o Docker possibilita que uma imagem com todos os aplicativos e configurações realizadas em um contêiner sejam transferidos para outro host.

Gabarito: Letra D

---

4. **(COMPERVE / TJ-RN - 2020)** Uma imagem do Docker é criada a partir de uma série de camadas, onde cada uma representa uma instrução no Dockerfile da imagem. Considerando que um analista do Tribunal de Justiça queira listar as camadas (layers) da imagem docker mailserver, ele deve executar o comando:

a) docker expose mailserver
b) docker layers mailserver
c) docker history mailserver
d) docker image mailserver

Comentários:

Para listar as layers da imagem Docker mailserver, o comando a ser executado é o Docker history mailserver.

Gabarito: Letra C

---

5. **(FURB / Prefeitura De Blumenau - 2022)** Sobre Docker, marque a alternativa CORRETA:

a) Docker é uma plataforma de software livre que permite que os desenvolvedores empacotem aplicativos em contêineres, que consistem em componentes executáveis e padronizados que combinam o código-fonte do aplicativo com as bibliotecas e dependências do sistema operacional (S.O.) que são necessárias para executar esse código em qualquer ambiente.

b) É projetado para ser flexível, ele estabelece um padrão para a construção de interfaces com o usuário do lado do servidor. A arquitetura define claramente uma separação entre a lógica da

aplicação e a apresentação, enquanto torna mais fácil ligar a camada de apresentação ao código do aplicativo.

c) É uma plataforma que estende a plataforma SE do Java e é o padrão para desenvolvimento de aplicativos empresariais entre a comunidade Java. Em outras palavras, trata-se de framework web MVC para a construção de interfaces de usuário baseadas em componentes.

d) Para começar a codificar o Docker, não é necessária nenhuma configuração especial ao servidor Glassfish. Basta instalar o Netbeans com suporte à tecnologia Java EE, que por padrão já será instalado o servidor Glassfish pré configurado.

e) É um popular framework para projetos JSF que pode ser usado para desenvolver rapidamente aplicações sofisticadas para empresas ou no desenvolvimento de sites padrão. Tecnicamente, é uma biblioteca de componentes de interface gráfica para as aplicações web baseadas em JSF.

Comentários:

Definição de Docker como uma plataforma de software livre que permite que os desenvolvedores empacotem aplicativos em contêineres, que consistem em componentes executáveis e padronizados que combinam o código-fonte do aplicativo com as bibliotecas e dependências do sistema operacional (S.O.) que são necessárias para executar esse código em qualquer ambiente, está perfeita.

Gabarito: Letra A

---

6. **(CESPE / DPE-RO - 2022)** A tecnologia Docker usa o *kernel* do Linux e recursos do *kernel* para segregar processos; as ferramentas baseadas nos contêineres Linux oferecem aos usuários acesso sem precedentes a aplicações, além da habilidade de implantar com rapidez e de ter total controle sobre as versões e distribuição. A respeito da tecnologia Docker, assinale a opção correta.

a) O Docker é uma tecnologia proprietária que automatiza a implantação da aplicação dentro do ambiente de contêiner.

b) O Docker oferece vantagem em questões como a limpeza de processos netos (grandchild) após o encerramento dos processos filhos (child).

c) O Docker fornece as mesmas funcionalidades oferecidas pelos contêineres Linux tradicionais, incluindo a capacidade de usar processos como cron ou syslog dentro do contêiner, junto à aplicação.

d) As instâncias de aplicativos em contêiner Docker usam mais memória do que as máquinas virtuais, sendo inicializadas e interrompidas mais lentamente.

e) Os contêineres Docker facilitam a colocação rápida de novas versões de software, com novos recursos de negócios, e a rápida reversão para uma versão anterior, se necessário.

Comentários:

Pessoal, os contêineres Docker facilitam a colocação rápida de novas versões de *software*, com novos recursos de negócios, e a rápida reversão para uma versão anterior, se necessário. Essa é uma das mais relevantes vantagens de se utilizar Docker.

Gabarito: Letra E

---

7. **(COPERV / UFSC - 2018)** A respeito da solução de contêiner Docker, analise as afirmativas abaixo e assinale a alternativa correta.

I. Uma imagem pode ser versionada com múltiplos commits.
II. O arquivo Dockerfile contém variáveis, comandos e/ou operações para criar uma nova instância Docker.
III. Depois de uma imagem ser criada, para alterá-la é necessário reexecutar o processo de criação.

a) Somente as afirmativas I e III estão corretas.
b) Somente a afirmativa II está correta.
c) Somente as afirmativas II e III estão corretas.
d) Somente as afirmativas I e II estão corretas.
e) Somente a afirmativa I está correta.

Comentários:

Das alternativas somente a I está correta. No Docker uma imagem pode ser versionada com múltiplos *commits*. Não temos a criação de uma nova instância Docker como dito no item II e sim de uma nova imagem e o item III está errado também, já que a imagem pode ser modificada no momento da execução.

Gabarito: Letra E

---

8. **(CESPE / SLU - 2019)** O Docker é uma ferramenta open source que permite a criação de ambientes virtuais por meio de Linux Containers, sendo uma das vantagens dos contêineres Docker fornecer uma virtualização em nível de sistema operacional, o que isola as aplicações em execução e não utiliza tantos recursos da máquina quanto as máquinas virtuais.

Comentários:

Perfeito! O Docker é uma ferramenta open source que permite a criação de ambientes virtuais por meio de Linux Containers, sendo uma das vantagens dos contêineres Docker fornecer uma virtualização em nível de sistema operacional, o que isola as aplicações em execução e não utiliza tantos recursos da máquina quanto as máquinas virtuais.

Gabarito: Correto

---

9. **(FCC / DPE-RS - 2017)** Considere, por hipótese, que a equipe de analistas da Defensoria Pública tenha optado pelo uso do Docker. Esta decisão foi motivada pelo fato de o Docker:

a) estar ganhando espaço como um gerenciador de máquinas virtuais no ambiente GNU/Linux e não ter bibliotecas próprias, mantendo as bibliotecas nativas utilizadas para gerenciar o LXC.

b) não utilizar Namespaces do Linux, o que permite prover espaços de trabalho isolados para os contêineres. Desta forma, quando um contêiner é criado, automaticamente é criada uma camada de isolamento para grupos de processos.

c) utilizar hypervisors, compatíveis com diversas plataformas, para executar máquinas virtuais que virtualizam hardware físico como parte de um desenvolvimento multiplataforma para testes e implementação de fluxo de trabalho.

d) permitir portabilidade de contêineres. É possível criar uma imagem de toda a configuração e aplicativos instalados em um contêiner e transferi-lo para outro host que tenha um Docker previamente instalado.

e) obter o mesmo desempenho da virtualização baseada em hypervisor, em que cada contêiner é executado em seu próprio sistema operacional, o que reduz a utilização de recursos de disco, embora os contêineres utilizem mais memória.

Comentários:

Docker permite a portabilidade de contêineres, vimos isso durante toda aula. Logo, com Docker é possível criar uma imagem de toda a configuração e aplicativos instalados em um contêiner e transferi-lo para outro *host* que tenha um Docker previamente instalado.

Gabarito: Letra D

---

10. **(FGV / CGU – 2022)** Uma das estratégias para reduzir o tamanho de imagens Docker consiste em:

a) combinar comandos RUN em um único comando;
b) substituir a imagem base por uma versão mais recente;
c) separar comandos RUN complexos em comandos menores;

d) reordenar os comandos de forma que o cache seja utilizado com maior frequência;
e) compactar arquivos a serem copiados para a imagem e descompactá-los durante a sua geração;

Comentários:

Conforme verificamos uma das estratégias para reduzir o tamanho de imagens Docker consiste em combinar comandos RUN em um único comando.

Gabarito: Letra A

---

11. (CEPUERJ / UERJ – 2021) Considerando-se os conceitos de docker, é correto afirmar que se trata de:

a) um complemento para virtualização completa, associado ao VMware ESXi
b) um complemento para virtualização completa, associado ao VMware
c) uma ferramenta para criar e manter máquinas virtuais
d) uma ferramenta para criar e manter containers

Comentários:

Pessoal, conforme estudamos e segundo: aws.amazon.com.pt/docker/

"*O Docker é uma plataforma de software que permite a criação, o teste e a implantação de aplicações rapidamente. O Docker cria pacotes de software em unidades padronizadas chamadas de contêineres que têm tudo o que o software precisa para ser executado, inclusive bibliotecas, ferramentas de sistema, código e runtime. Ao usar o Docker, é possível implantar e escalar rapidamente aplicações em qualquer ambiente e ter a certeza de que o seu código será executado.*"

Gabarito: Letra D

---

12. (CESPE / SLU – 2019) O Docker é uma ferramenta *open source* que permite a criação de ambientes virtuais por meio de Linux Containers, sendo uma das vantagens dos contêineres Docker fornecer uma virtualização em nível de sistema operacional, o que isola as aplicações em execução e não utiliza tantos recursos da máquina quanto as máquinas virtuais.

Comentários:

De fato pessoal, conforme vimos o Docker é uma ferramenta *open source* que permite a criação de ambientes virtuais por meio de Linux Containers, sendo uma das vantagens dos contêineres Docker fornecer uma virtualização em nível de sistema operacional, o que isola as aplicações em execução e não utiliza tantos recursos da máquina quanto as máquinas virtuais.

Gabarito: Correto

---

13. (IADES / BRB – 2021) O Docker é uma plataforma aberta para desenvolvimento, entrega e execução de aplicações. A respeito da funcionalidade *tmpfs mounts*, assinale a alternativa correta.

a) Assim como os volumes e bind mounts, é possível compartilhar o tmpfs mounts entre diversos containers.
b) Essa funcionalidade está presente apenas quando se executa o Docker no Linux.
c) O tmpfs mounts pode ser armazenado em unidade de armazenamento do tipo SSD do servidor que hospeda o Docker.
d) O tmpfs mounts permanece persistente após a parada do container.
e) Não é possível limitar o seu tamanho; por padrão, ele é ilimitado.

Comentários:

A funcionalidade tmpfs mounts está presente apenas quando se executa o Docker no Linux. Penso que esse tipo de conhecimento está muito mais atrelado ao SO Linux, mas sabe como é a banca né? Pede qualquer coisa sobre o tema na questão.

Gabarito: Letra B

---

14. (COMPERVE / TJ-RN – 2020) O processo de virtualização é possibilitado por um hypervisor, que é um software instalado em cima de um servidor físico e que, a partir dele, é possível a criação de máquinas virtuais que podem, cada uma, conter sistemas operacionais diferentes. Analise as seguintes afirmativas sobre o uso de Máquinas Virtuais e Docker.

I. A virtualização permite o isolamento total do ambiente da sua aplicação, pois ela não emula a máquina virtual por completo.

II. O Docker permite "empacotar" uma aplicação ou sistema dentro de um container, sendo que este container pode posteriormente ser executado em qualquer máquina que tenha o Docker instalado.

III. Vários containers podem ser executados na mesma máquina e compartilhar o kernel do SO com outros containers, cada um executando como processos isolados no espaço do usuário.

IV. Em um sistema de virtualização tradicional, o sistema operacional é isolado dos demais instalados dentro da máquina host.

Estão corretas apenas as afirmativas:

a) III e IV

b) II e IV
c) I, II e III
d) II, III e IV

Comentários:

A questão extrapola o conhecimento somente em Docker, porém trouxe aqui para vocês para nos atentarmos somente a alternativa II que está correta e foi a definição que vimos na aula. O Docker permite "empacotar" uma aplicação ou sistema dentro de um container, sendo que este container pode posteriormente ser executado em qualquer máquina que tenha o Docker instalado.

Gabarito: Letra D

---

15. (CESPE / STJ – 2015) *Dockerfile* é um arquivo de texto que contém todos os comandos, em ordem, necessários para construir uma determinada imagem *Docker*. Sobre as instruções contidas em um *Dockerfile*, assinale a alternativa correta.

a) A instrução VOLUME configura o tamanho da imagem.
b) A instrução ENV adiciona metadados para uma imagem.
c) A instrução WORKDIR permite a criação de um diretório no host onde ficam armazenados os dados do container.
d) A instrução EXPOSE informa ao Docker que o container escuta nas portas de rede especificadas em tempo de execução.
e) A instrução FROM configura qual será a aplicação principal do container, sendo executada após a inicialização do container.

Comentários:

Dockerfile e não FileDocker é um arquivo de texto que contém todos os comandos, em ordem, necessários para construir uma determinada imagem Docker. Logo, conforme estudamos, a instrução EXPOSE informa ao *Docker* que o container escuta nas portas de rede especificadas em tempo de execução.

Gabarito: Letra D

---

16. (COMPERVE / TJ-RN-2020) Uma imagem de container do Docker é um pacote de software leve, independente e executável que inclui tudo o que é necessário para executar uma aplicação. Na criação de um arquivo Dockerfile, a instrução EXPOSE:

a) mapeia uma porta externa para uma porta interna à rede Docker.
b) divulga uma porta (TCP ou UDP) para os hosts externos à rede Docker.
c) expõe um serviço do container para a rede Docker default.
d) documenta quais portas se pretende publicar.

Comentários:

Conforme vimos, a instrução EXPOSE do Docker documenta quais portas se pretende publicar.

Gabarito: Letra D

---

17. (CESPE / SEFAZ-CE – 2021) As alterações efetuadas em arquivos e diretórios copiados de uma camada base para dentro de um container docker, por padrão, são vistas pelos múltiplos containers do mesmo sistema de arquivos.

Comentários:

As alterações efetuadas em arquivos ou diretórios dentro de um container são isoladas e não podem ser vistas fora dele.

Gabarito: Errado

LISTA DE QUESTÕES – DOCKER

1. **(FEPESE / Prefeitura de Florianópolis - 2019)** O Docker pode ler instruções através de um arquivo texto que contém instruções para montar uma imagem (dockerfile). Nesse contexto, qual a palavra-chave ou instrução que indica ao docker a porta que o container deve utilizar em tempo de execução?

   a) HTTP
   b) PORT
   c) EXPOSE
   d) DOCKER
   e) SERVERPORT

2. **(COMPERVE / TJ-RN - 2020)** Os volumes são mecanismos utilizados para persistir os dados gerados e usados pelos containers do Docker. Embora as montagens de ligação dependam da estrutura de diretórios da máquina *host*, os volumes são completamente gerenciados pelo Docker. Considerando que um analista queira criar um volume de nome *my-volume* dentro de um docker, ele deve executar o comando:

   a) docker volume create my-volume
   b) docker create volume my-volume
   c) docker run create volume my-volume
   d) docker create run volume my-volume

3. **(AOCP / MJSP - 2020)** O Docker possibilita que uma imagem com todos os aplicativos e configurações realizadas em um contêiner sejam transferidos para outro host, bastando que este tenha o Docker instalado. Assinale a alternativa que apresenta o nome dessa operação.

   a) Restore.
   b) Dump.
   c) Drill down Contêiner.
   d) Portabilidade.
   e) Drill up Contêiner.

4. **(COMPERVE / TJ-RN - 2020)** Uma imagem do Docker é criada a partir de uma série de camadas, onde cada uma representa uma instrução no Dockerfile da imagem. Considerando que um analista do Tribunal de Justiça queira listar as camadas (layers) da imagem docker mailserver, ele deve executar o comando:

   a) docker expose mailserver
   b) docker layers mailserver
   c) docker history mailserver

d) docker image mailserver

5. **(FURB / Prefeitura De Blumenau - 2022)** Sobre Docker, marque a alternativa CORRETA:

a) Docker é uma plataforma de software livre que permite que os desenvolvedores empacotem aplicativos em contêineres, que consistem em componentes executáveis e padronizados que combinam o código-fonte do aplicativo com as bibliotecas e dependências do sistema operacional (S.O.) que são necessárias para executar esse código em qualquer ambiente.

b) É projetado para ser flexível, ele estabelece um padrão para a construção de interfaces com o usuário do lado do servidor. A arquitetura define claramente uma separação entre a lógica da aplicação e a apresentação, enquanto torna mais fácil ligar a camada de apresentação ao código do aplicativo.

c) É uma plataforma que estende a plataforma SE do Java e é o padrão para desenvolvimento de aplicativos empresariais entre a comunidade Java. Em outras palavras, trata-se de framework web MVC para a construção de interfaces de usuário baseadas em componentes.

d) Para começar a codificar o Docker, não é necessária nenhuma configuração especial ao servidor Glassfish. Basta instalar o Netbeans com suporte à tecnologia Java EE, que por padrão já será instalado o servidor Glassfish pré configurado.

e) É um popular framework para projetos JSF que pode ser usado para desenvolver rapidamente aplicações sofisticadas para empresas ou no desenvolvimento de sites padrão. Tecnicamente, é uma biblioteca de componentes de interface gráfica para as aplicações web baseadas em JSF.

6. **(CESPE / DPE-RO - 2022)** A tecnologia Docker usa o *kernel* do Linux e recursos do *kernel* para segregar processos; as ferramentas baseadas nos contêineres Linux oferecem aos usuários acesso sem precedentes a aplicações, além da habilidade de implantar com rapidez e de ter total controle sobre as versões e distribuição. A respeito da tecnologia Docker, assinale a opção correta.

a) O Docker é uma tecnologia proprietária que automatiza a implantação da aplicação dentro do ambiente de contêiner.

b) O Docker oferece vantagem em questões como a limpeza de processos netos (grandchild) após o encerramento dos processos filhos (child).

c) O Docker fornece as mesmas funcionalidades oferecidas pelos contêineres Linux tradicionais, incluindo a capacidade de usar processos como cron ou syslog dentro do contêiner, junto à aplicação.

d) As instâncias de aplicativos em contêiner Docker usam mais memória do que as máquinas virtuais, sendo inicializadas e interrompidas mais lentamente.

e) Os contêineres Docker facilitam a colocação rápida de novas versões de software, com novos recursos de negócios, e a rápida reversão para uma versão anterior, se necessário.

7. **(COPERV / UFSC - 2018)** A respeito da solução de contêiner Docker, analise as afirmativas abaixo e assinale a alternativa correta.

I. Uma imagem pode ser versionada com múltiplos commits.
II. O arquivo Dockerfile contém variáveis, comandos e/ou operações para criar uma nova instância Docker.
III. Depois de uma imagem ser criada, para alterá-la é necessário reexecutar o processo de criação.

a) Somente as afirmativas I e III estão corretas.
b) Somente a afirmativa II está correta.
c) Somente as afirmativas II e III estão corretas.
d) Somente as afirmativas I e II estão corretas.
e) Somente a afirmativa I está correta.

8. **(CESPE / SLU - 2019)** O Docker é uma ferramenta open source que permite a criação de ambientes virtuais por meio de Linux Containers, sendo uma das vantagens dos contêineres Docker fornecer uma virtualização em nível de sistema operacional, o que isola as aplicações em execução e não utiliza tantos recursos da máquina quanto as máquinas virtuais.

9. **(FCC / DPE-RS - 2017)** Considere, por hipótese, que a equipe de analistas da Defensoria Pública tenha optado pelo uso do Docker. Esta decisão foi motivada pelo fato de o Docker:

a) estar ganhando espaço como um gerenciador de máquinas virtuais no ambiente GNU/Linux e não ter bibliotecas próprias, mantendo as bibliotecas nativas utilizadas para gerenciar o LXC.

b) não utilizar Namespaces do Linux, o que permite prover espaços de trabalho isolados para os contêineres. Desta forma, quando um contêiner é criado, automaticamente é criada uma camada de isolamento para grupos de processos.

c) utilizar hypervisors, compatíveis com diversas plataformas, para executar máquinas virtuais que virtualizam hardware físico como parte de um desenvolvimento multiplataforma para testes e implementação de fluxo de trabalho.

d) permitir portabilidade de contêineres. É possível criar uma imagem de toda a configuração e aplicativos instalados em um contêiner e transferi-lo para outro host que tenha um Docker previamente instalado.

e) obter o mesmo desempenho da virtualização baseada em hypervisor, em que cada contêiner é executado em seu próprio sistema operacional, o que reduz a utilização de recursos de disco, embora os contêineres utilizem mais memória.

10. **(FGV / CGU – 2022)** Uma das estratégias para reduzir o tamanho de imagens Docker consiste em:

a) combinar comandos RUN em um único comando;
b) substituir a imagem base por uma versão mais recente;
c) separar comandos RUN complexos em comandos menores;
d) reordenar os comandos de forma que o cache seja utilizado com maior frequência;
e) compactar arquivos a serem copiados para a imagem e descompactá-los durante a sua geração;

11. **(CEPUERJ / UERJ – 2021)** Considerando-se os conceitos de docker, é correto afirmar que se trata de:

a) um complemento para virtualização completa, associado ao VMware ESXi
b) um complemento para virtualização completa, associado ao VMware
c) uma ferramenta para criar e manter máquinas virtuais
d) uma ferramenta para criar e manter containers

12. **(CESPE / SLU – 2019)** O Docker é uma ferramenta *open source* que permite a criação de ambientes virtuais por meio de Linux Containers, sendo uma das vantagens dos contêineres Docker fornecer uma virtualização em nível de sistema operacional, o que isola as aplicações em execução e não utiliza tantos recursos da máquina quanto as máquinas virtuais.

13. **(IADES / BRB – 2021)** O Docker é uma plataforma aberta para desenvolvimento, entrega e execução de aplicações. A respeito da funcionalidade *tmpfs mounts*, assinale a alternativa correta.

a) Assim como os volumes e bind mounts, é possível compartilhar o tmpfs mounts entre diversos containers.
b) Essa funcionalidade está presente apenas quando se executa o Docker no Linux.
c) O tmpfs mounts pode ser armazenado em unidade de armazenamento do tipo SSD do servidor que hospeda o Docker.
d) O tmpfs mounts permanece persistente após a parada do container.
e) Não é possível limitar o seu tamanho; por padrão, ele é ilimitado.

14. **(COMPERVE / TJ-RN – 2020)** O processo de virtualização é possibilitado por um hypervisor, que é um software instalado em cima de um servidor físico e que, a partir dele, é possível a criação de máquinas virtuais que podem, cada uma, conter sistemas operacionais diferentes. Analise as seguintes afirmativas sobre o uso de Máquinas Virtuais e Docker.

I. A virtualização permite o isolamento total do ambiente da sua aplicação, pois ela não emula a máquina virtual por completo.

II. O Docker permite "empacotar" uma aplicação ou sistema dentro de um container, sendo que este container pode posteriormente ser executado em qualquer máquina que tenha o Docker instalado.

III. Vários containers podem ser executados na mesma máquina e compartilhar o kernel do SO com outros containers, cada um executando como processos isolados no espaço do usuário.

IV. Em um sistema de virtualização tradicional, o sistema operacional é isolado dos demais instalados dentro da máquina host.

Estão corretas apenas as afirmativas

a) III e IV
b) II e IV
c) I, II e III
d) II, III e IV

15. (CESPE / STJ – 2015) *Dockerfile* é um um arquivo de texto que contém todos os comandos, em ordem, necessários para construir uma determinada imagem *Docker*. Sobre as instruções contidas em um *Dockerfile*, assinale a alternativa correta.

a) A instrução VOLUME configura o tamanho da imagem.
b) A instrução ENV adiciona metadados para uma imagem.
c) A instrução WORKDIR permite a criação de um diretório no host onde ficam armazenados os dados do container.
d) A instrução EXPOSE informa ao Docker que o container escuta nas portas de rede especificadas em tempo de execução.
e) A instrução FROM configura qual será a aplicação principal do container, sendo executada após a inicialização do container.

16. (COMPERVE / TJ-RN-2020) Uma imagem de container do Docker é um pacote de software leve, independente e executável que inclui tudo o que é necessário para executar uma aplicação. Na criação de um arquivo Dockerfile, a instrução EXPOSE:

a) mapeia uma porta externa para uma porta interna à rede Docker.
b) divulga uma porta (TCP ou UDP) para os hosts externos à rede Docker.
c) expõe um serviço do container para a rede Docker default.
d) documenta quais portas se pretende publicar.

17. **(CESPE / SEFAZ-CE – 2021)** As alterações efetuadas em arquivos e diretórios copiados de uma camada base para dentro de um container docker, por padrão, são vistas pelos múltiplos containers do mesmo sistema de arquivos.

GABARITO – DOCKER

1. CORRETO
2. ERRADO
3. ERRADO
4. CORRETO
5. ERRADO
6. CORRETO
7. ERRADO
8. CORRETO
9. ERRADO
10. CORRETO
11. ERRADO
12. CORRETO
13. LETRA B
14. LETRA D
15. LETRA D
16. ERRADO
17. ERRADO

KUBERNETS

Conceitos Básicos

INCIDÊNCIA EM PROVA: MÉDIA

O Kubernetes é um sistema de código aberto que foi desenvolvido pelo Google **para gerenciamento de aplicativos em containers através de múltiplos hosts de um cluster**. Tem como principal objetivo facilitar a implantação de aplicativos baseados em microservices. Ele foi baseado na experiência do Google de muitos anos trabalho com containers, adaptando-o para se trabalhar com Docker.

kubernetes

O Kubernetes é uma ferramenta de **orquestração que oferece recursos de gerenciamento para containers**, foi muito útil para ser utilizado até o Docker Swarm 1.0, pois disponibilizava muitos recursos que o Docker não disponibilizava até aquele momento, entre eles: Balanceamento de carga e movimento de containers sem perda de dados.

A principal vantagem que se tem ao utilizar o Kubernetes é que você não está preso as limitações da API do Docker, logo você tem total liberdade já que o Kubernetes não foi desenvolvido especialmente para o Docker, você pode trocar a sua estrutura de Docker para Rockets e vice-versa por exemplo.

Hoje o Kubernetes é mantido pela Cloud Native Computing Foundation e tem como papel principal prover o gerenciamento de clusters que executam as aplicações. Esses clusters podem ser hots disponíveis em nuvem, logo kubernetes é ideal para cloud native apps que podem exigir escalabilidade rápida.

Para ficar mais claro, imagine o seguinte: Você precisa fazer deploy de uma nova versão de uma aplicação em nuvem, milhares de coisas podem dar errado e você pode precisar realizar rollback para a versão anterior. Porém além disso, a aplicação é composta por dezenas de microsserviços, cada um com um ciclo de vida diferente do outro, releases diferentes, tecnologias diferentes, etc.

Pensando nisso a arquitetura de microsserviços adotou o Kubernetes como uma referência, já que na prática as aplicações baseadas em arquitetura de microsserviços, incorporam automação em todo seu ciclo de vida, logo diferente de aplicações monolíticas, com forte acomplamento, as APIs com microsserviços tem características descentralizadas e fracamente acopladas. Logo, basta encapsular as principais características dos serviços em containers.

Como esses contêineres possuem o serviço e todas as suas dependências, eles se tornam independentes da infraestrutura, podendo ser facilmente migrados de uma cloud para outra, por exemplo, facilitando escalabilidade e deploy.

Dentro desse contexto, o Kubernetes oferece várias funcionalidades, dentre eles temos a ideia central, isto é, o estado da aplicação. **Para o Kubernetes existes dois estados de uma aplicação: o atual e o desejado.** O **estado atual** da aplicação descreve a realidade. Por exemplo, quantas réplicas de um determinado serviço estão em execução, qual a versão em produção de cada serviço e por aí vai.

Já o **estado desejado** descreve como o time ou a pessoa responsável pela aplicação deseja que ela esteja naquele momento. O Kubernetes implementa uma série de loops que ficam constantemente verificando se o estado atual é igual ao estado desejado. Esse papel é desempenhado pelos chamados **Controllers.**

Quando um controller identifica que o estado atual é diferente do estado desejado, ele aciona outros componentes do sistema para fazer novamente com que o estado atual se iguale ao estado desejado. Todo esse processo de monitoramento e gestão do estado da aplicação, sem contar a execução da aplicação em si, exige uma série de componentes. É por isso que a arquitetura de um ambiente Kubernetes é baseada em um cluster de máquinas.

PROFESSOR, COMO FUNCIONA NA PRÁTICA O KUBERNETES?

Pessoal, o Kubernetes é composto por uma vários componentes, cada um com um propósito diferente. Para garantir que exista uma separação de responsabilidades e que o sistema seja resiliente, o kubernetes utiliza um cluster de máquinas para ser executado, logo as máquinas de um cluster são separadas em três tipos: **Node, Etcd e Master**.

MÁQUINAS KUBERNETES	DESCRIÇÃO
NODE	O papel de um node é executar os contêineres que encapsulam as aplicações.

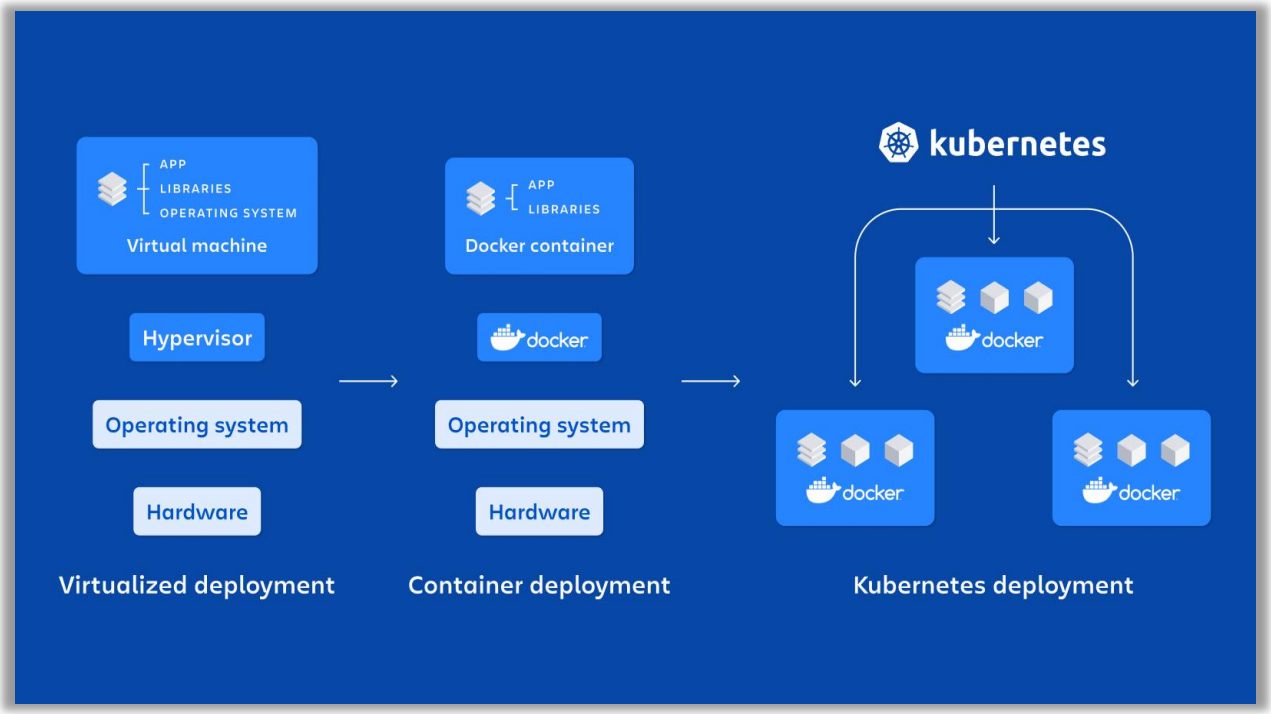
ETCD	O etcd é a base de dados distribuída que é utilizada para armazenar tudo o que acontece dentro do cluster, incluindo o estado da aplicação.
MASTER	O master é responsável pelos principais componentes do kubernetes, como o scheduler, que tem a responsabilidadae de controlar a alocação de recursos no cluster.

PROFESSOR, ONDE ENTRA OS MICROSSERVIÇOS NO KUBERNETES?

Como já vimos as aplicações que utilizam microsserviços, constumam utilizar containers para encapsulamento dos microsserviços. No entanto, quando falamos sobre aplicações sendo executadas em um cluster Kubernetes, não falamos sobre contêineres diretamente, mas sim sobre **Pods**. No Kubernetes, temos o **kubelet** que é uma pequena aplicação localizada em um nó que se comunica com o plano de controle, assegurando que os containers estejam em execução em um pod, que consiste no menor e mais simples objeto do Kubernetes.

Pods são a unidade básica de um cluster. **Os Pods encapsulam um ou mais containers de uma aplicação e representam um processo dentro do cluster**. Logo, quando fazemos o deploy de uma aplicação no Kubernetes, estamos criando uma ou mais Pods. No Kubernetes, **kubelet** é uma pequena aplicação localizada em um nó que se comunica com o plano de controle, assegurando que os containers estejam em execução em um pod, que consiste no menor e mais simples objeto do Kubernetes.

Por fim, para garantir que o acesso a um microsserviços esteja sempre disponível, temos um objeto no Kubernetes, chamado de **Service**, o **Service é que encapsula um ou mais Pods e é capaz de encontra-los dinamicamente em qualquer lugar do cluster**. A seguir conseguimos didaticamente ver a diferença da virtualização, do emprego de Docker e do Kubernetes:



Precisamos por últimos falar do CNFC (Cloud Native Computing Foundation):

Pessoal, o CNFC é parte do Linux foundation e oferece suporte para projetos nativos em nuvem, como o Kubernetes, o Envoy e o Prometheus. Como fundação a CNFC é responsável por diversas certificações em Kubernetes, que é seu principal expoente, além de promover a cultura DevOps mundo a fora.

O que a Cloud Native Computing Foundation busca é promover o desenvolvimento e evolução do emprego de containers no desenvolvimento de soluções, principalmente em soluções nativas de nuvem.

(TJ-MS - 2017) Kubernetes é uma ferramenta de orquestração que oferece recursos de gerenciamento para containers, como balanceamento de carga e migração sem perda de dados.

_____

Comentários: Perfeito! Foi a primeira que vimos: Kubernetes é uma ferramenta de orquestração que oferece recursos de gerenciamento para containers, como balanceamento de carga e migração sem perda de dados (Correto).

QUESTÕES COMENTADAS – KUBERNETES

1. **(FGV / FUNSAÚDE-CE – 2021)** *Pod* é uma unidade atômica de escalonamento, implantação e isolamento na execução de um grupo de contêineres no Kubernetes, analise as afirmativas a seguir.

   I. Um *Pod* garante uma mesma localização para os seus contêineres, graças a isso eles têm diversas formas de interagir com bom desempenho, por exemplo, através de troca de arquivos, uso de interface de redes ou de mecanismos de comunicação entre processos.
   II. Um *Pod* tem um endereço IP, um nome e uma faixa de portas compartilhadas por todos os contêineres pertencentes a ele. Isso significa que os contêineres de um mesmo *Pod* devem ser configurados cuidadosamente a fim de evitar conflitos de portas.
   III. Um *Pod* é um elemento persistente no tempo, ele resiste às operações de redimensionamento, falhas de verificação de sanidade de contêineres e migrações entre nós.

   Está correto o que se afirma em:

   a) I, somente.
   b) II, somente.
   c) III, somente.
   d) I e II, somente.
   e) I e III, somente.

Comentários:

Um Pod tem um endereço IP, um nome e uma faixa de portas compartilhadas por todos os contêineres pertencentes a ele. Isso significa que os contêineres de um mesmo Pod devem ser configurados cuidadosamente a fim de evitar conflitos de portas. Além disso, um Pod garante uma mesma localização para os seus contêineres, graças a isso eles têm diversas formas de interagir com bom desempenho, por exemplo, através de troca de arquivos, uso de interface de redes ou de mecanismos de comunicação entre processos.

Gabarito: Letra D

---

2. **(CESPE / SEFAZ-CE – 2021)** Com a implantação do Kubernetes, é obtido um cluster com pelo menos um nó de trabalho (worker node); os nós de trabalho, por sua vez, hospedam vários componentes da carga de trabalho do aplicativo.

Comentários:

Pessoal, com a implantação do Kubernetes, é obtido um cluster com pelo menos um nó de trabalho (worker node); os nós de trabalho, por sua vez, hospedam vários componentes da carga de trabalho do aplicativo.

Gabarito: Correto

---

3. **(CESPE / SERPRO - 2020)** A camada de gerenciamento do Kubernetes possui o componente etcd, cuja função é observar pods que foram criados sem nenhum node atribuído e selecionar um node para execução.

Comentários:

Conforme vimos, o etcd é a base de dados distribuída que é utilizada para armazenar tudo o que acontece dentro do cluster, incluindo o estado da aplicação. Não tem função de observar pods que foram criados sem nenhum noda atribuído e selecionar um node para execução.

Gabarito: Errado

---

4. **(IADES / BRB – 2021)** Kubernetes é uma plataforma de código aberto, portável e extensiva para o gerenciamento de cargas de trabalho e serviços distribuídos em contêineres, que facilita tanto a configuração declarativa quanto a automação. Ele possui um ecossistema grande e de rápido crescimento. Serviços, suporte e ferramentas para Kubernetes estão amplamente disponíveis. Disponível em: <https://kubernetes.io/pt-br/docs/concepts/overview/>. Acesso em: 25 jun. 2021, com adaptações.

   Com base no texto apresentado e considerando o contexto do Kubernetes, assinale a alternativa que corresponde à ferramenta disponibilizada para realizar operações nos *clusters* Kubernetes, por meio de interface de linha de comando, pela qual é possível realizar a implantação de aplicações, inspecionar e gerenciar recursos do *cluster* e visualizar *logs*.

   a) Kubeadm
   b) Minikube
   c) Kubectl
   d) Kind
   e) Kubelet

Comentários:

Kubernetes é uma plataforma de código aberto, portável e extensiva para o gerenciamento de cargas de trabalho e serviços distribuídos em contêineres, que facilita tanto a configuração declarativa quanto a automação, a ferramenta para operações nos clusters, que é utilizada em linha de comando é o Kubectl.

Gabarito: Letra C

---

5. **(CESPE / SEFAZ-CE – 2021)** No Kubernetes, kubelet é uma pequena aplicação localizada em um nó que se comunica com o plano de controle, assegurando que os containers estejam em execução em um pod, que consiste no menor e mais simples objeto do Kubernetes.

Comentários:

Exato! Literalmente o vimos na aula: No Kubernetes, temos o kubelet que é uma pequena aplicação localizada em um nó que se comunica com o plano de controle, assegurando que os containers estejam em execução em um pod, que consiste no menor e mais simples objeto do Kubernetes

Gabarito: Correto

---

6. **(PUC-RS / TJ-MS - 2017)** Sistemas virtualizados e containers são conceitos importantes para computação na nuvem. Para gerenciar grande número de servidores físicos, virtualizados e containers, utilizam-se ferramentas especializadas de configuração remota. Indique a afirmativa que descreve de forma **CORRETA** os conceitos relativos a sistemas virtualizados e *containers* e as ferramentas de gerenciamento disponíveis.

a) *Puppet* e *Ansible* são ferramentas que tem a finalidade de simplificar o processo de gerenciamento de servidores remotos. Essas ferramentas funcionam apenas com servidores físicos ou virtualizados. Elas não suportam containers.

b) *Puppet* e *Ansible* podem ser usados para gerenciar serviços virtualizados. *Ansible* é preferível por ser uma ferramenta multi-plataforma, enquanto *Puppet* funciona apenas para Linux, pois todos os seus comandos remotos são executados via SSH.

c) *Containers* e máquinas virtuais são sinônimos, pois ambos são usados para virtualizar o *hardware* que hospeda um sistema operacional completo, que pode ser diferente do sistema operacional da máquina física.

d) *Containers* do tipo Dockers podem ser orquestrados apenas pelo *Docker Swarm*, que foi desenvolvido especificamente para suportar essa tecnologia de container.

e) *Kubernetes* é uma ferramenta de orquestração que oferece recursos de gerenciamento para containers, como balanceamento de carga e migração sem perda de dados.

Comentários:

A questão extrapola no conhecimento e só a alternativa E foca no nosso assunto, Kubernetes, sem mais enrolação, está perfeita a definição: Kubernetes é uma ferramenta de orquestração que

oferece recursos de gerenciamento para containers, como balanceamento de carga e migração sem perda de dados.

Gabarito: Letra E

---

7. **(CESPE / SEFAZ-CE - 2021)** No Kubernetes, kubelet é uma pequena aplicação localizada em um nó que se comunica com o plano de controle, assegurando que os containers estejam em execução em um pod, que consiste no menor e mais simples objeto do Kubernetes.

Comentários:

Conforme vimos, o kuberlet é uma aplicação localizada em um nó que se comunica com o plano de controle, assegurando que os containers estejam em execução em um pod, que consiste no menor e mais simples objeto do Kubernetes.

Gabarito: Correto

---

8. **(CESPE / SERPRO - 2021)** Para obter o status de um node nomeado como node1 em um cluster, deve ser executado o comando a seguir: Kubectl describe node node1.

Comentários:

É impossível decorar todos os comandos do Kubectl, parece que dessa vez a banca entendeu isso.

Gabarito: Anulada

---

9. **(CESPE / SERPRO - 2021)** A camada de gerenciamento possui o componente etcd, cuja função é observar pods que foram criados sem nenhum node atribuído e selecionar um node para execução.

Comentários:

O componente etcd, como vimios não tem a função de observar pods, veja:

MÁQUINAS KUBERNETES	DESCRIÇÃO
NODE	O papel de um node é executar os contêineres que encapsulam as aplicações.
ETCD	O etcd é a base de dados distribuída que é utilizada para armazenar tudo o que acontece dentro do cluster, incluindo o estado da aplicação.
MASTER	O master é responsável pelos principais componentes do kubernetes, como o scheduler, que tem a responsabilidadae de controlar a alocação de recursos no cluster.

A questão trata do kube scheduler que é o componente cuja função é observar pods que foram criados sem nenhum node atribuído e selecionar um node para execução.

Gabarito: Errado

---

10. (FGV / TJ-RO – 2021) A equipe de desenvolvimento de sistemas de um tribunal de contas está guiando a implantação de um Webservice REST. A implantação será dividida nos seguintes grupos distintos de containers *Docker*:

- Grupo A: responsável pela execução da aplicação do Webservice REST
- Grupo B: responsável pela execução do Sistema Gerenciador de Banco de Dados utilizado pelo Webservice REST Os Grupos A e B terão seu próprio contexto de armazenamento e rede a serem orquestrados por um cluster de *Kubernetes.*

Para que a conexão de rede entre os containers dos Grupos A e B seja bem-sucedida pelo orquestrador, independentemente dos endereços IP a eles atribuídos, deverá ser configurado um novo:

a) pod
b) replicaSet
c) service
d) ingress
e) statefulSet

Comentários:

Deverá ser configurado um novo service. Pessoal, essa questão vai além na orquestração de clusters no Kubernetes, mas tudo bem. Note que o Pod já é um conjunto de containers, porém para que possamos cria-lo como serviço de rede, basta configurar um serviço (service).

Gabarito: Letra C

---

11. (CESPE / STJ – 2015) A orquestração automatiza a implantação, o gerenciamento, a escala e a rede dos contêineres. As ferramentas de orquestração de contêineres fornecem um *framework* para gerenciar arquiteturas de microsserviços e contêineres em escala, e muitas delas são usadas no gerenciamento do ciclo de vida dos contêineres; entre elas, o Docker Swarm é uma plataforma

a) que permite utilizar diversos recursos e ferramentas, como Apache e PHP, porém tudo rodando em um mesmo sistema operacional.

b) de código aberto criada pelo Google para operações de implantação de contêiner, aumento e redução e automação em clusters de *hosts.*

c) de orquestração de contêiner de código aberto, sendo o mecanismo de clusterização nativo para e pelo Docker, utilizando sua mesma linha de comando.

d) que roda sobre o Kubernetes instalado em sistema operacional na versão Enterprise da Red Hat, agregando opções de monitoramento, integração e entrega contínua.

e) usada pela Amazon para fornecer outros serviços aos clientes, como DNS, balanceamento, segurança e monitoramento, se integrando nativamente.

Comentários:

O correto seria dizer que o Docker Swarm é plataforma de orquestração de contêiner de código aberto, sendo o mecanismo de clusterização nativo para e pelo Docker, utilizando sua mesma linha de comando. Docker Swarm e Kubernets são orquestradores de container independentes.

Gabarito: Letra C

LISTA DE QUESTÕES – KUBERNETES

1. **(FGV / FUNSAÚDE-CE – 2021)** *Pod* é uma unidade atômica de escalonamento, implantação e isolamento na execução de um grupo de contêineres no Kubernetes, analise as afirmativas a seguir.

   I. Um *Pod* garante uma mesma localização para os seus contêineres, graças a isso eles têm diversas formas de interagir com bom desempenho, por exemplo, através de troca de arquivos, uso de interface de redes ou de mecanismos de comunicação entre processos.
   II. Um *Pod* tem um endereço IP, um nome e uma faixa de portas compartilhadas por todos os contêineres pertencentes a ele. Isso significa que os contêineres de um mesmo *Pod* devem ser configurados cuidadosamente a fim de evitar conflitos de portas.
   III. Um *Pod* é um elemento persistente no tempo, ele resiste às operações de redimensionamento, falhas de verificação de sanidade de contêineres e migrações entre nós.

   Está correto o que se afirma em

   a) I, somente.
   b) II, somente.
   c) III, somente.
   d) I e II, somente.
   e) I e III, somente.

2. **(CESPE / SEFAZ-CE – 2021)** Com a implantação do Kubernetes, é obtido um cluster com pelo menos um nó de trabalho (worker node); os nós de trabalho, por sua vez, hospedam vários componentes da carga de trabalho do aplicativo.

3. **(CESPE / SERPRO - 2020)** A camada de gerenciamento do Kubernetes possui o componente etcd, cuja função é observar pods que foram criados sem nenhum node atribuído e selecionar um node para execução.

4. **(IADES / BRB – 2021)** Kubernetes é uma plataforma de código aberto, portável e extensiva para o gerenciamento de cargas de trabalho e serviços distribuídos em contêineres, que facilita tanto a configuração declarativa quanto a automação. Ele possui um ecossistema grande e de rápido crescimento. Serviços, suporte e ferramentas para Kubernetes estão amplamente disponíveis. Disponível em: <https://kubernetes.io/pt-br/docs/concepts/overview/>. Acesso em: 25 jun. 2021, com adaptações.

   Com base no texto apresentado e considerando o contexto do Kubernetes, assinale a alternativa que corresponde à ferramenta disponibilizada para realizar operações nos *clusters* Kubernetes, por meio de interface de linha de comando, pela qual é possível realizar a implantação de aplicações, inspecionar e gerenciar recursos do *cluster* e visualizar *logs*.

a) Kubeadm
b) Minikube
c) Kubectl
d) Kind
e) Kubelet

5. **(CESPE / SEFAZ-CE – 2021)** No Kubernetes, kubelet é uma pequena aplicação localizada em um nó que se comunica com o plano de controle, assegurando que os containers estejam em execução em um pod, que consiste no menor e mais simples objeto do Kubernetes.

6. **(PUC-RS / TJ-MS - 2017)** Sistemas virtualizados e containers são conceitos importantes para computação na nuvem. Para gerenciar grande número de servidores físicos, virtualizados e containers, utilizam-se ferramentas especializadas de configuração remota. Indique a afirmativa que descreve de forma **CORRETA** os conceitos relativos a sistemas virtualizados e *containers* e as ferramentas de gerenciamento disponíveis.

a) *Puppet* e *Ansible* são ferramentas que tem a finalidade de simplificar o processo de gerenciamento de servidores remotos. Essas ferramentas funcionam apenas com servidores físicos ou virtualizados. Elas não suportam containers.

b) *Puppet* e *Ansible* podem ser usados para gerenciar serviços virtualizados. *Ansible* é preferível por ser uma ferramenta multi-plataforma, enquanto *Puppet* funciona apenas para Linux, pois todos os seus comandos remotos são executados via SSH.

c) *Containers* e máquinas virtuais são sinônimos, pois ambos são usados para virtualizar o *hardware* que hospeda um sistema operacional completo, que pode ser diferente do sistema operacional da máquina física.

d) *Containers* do tipo Dockers podem ser orquestrados apenas pelo *Docker Swarm*, que foi desenvolvido especificamente para suportar essa tecnologia de container.

e) *Kubernetes* é uma ferramenta de orquestração que oferece recursos de gerenciamento para containers, como balanceamento de carga e migração sem perda de dados.

7. **(CESPE / SEFAZ-CE - 2021)** No Kubernetes, kubelet é uma pequena aplicação localizada em um nó que se comunica com o plano de controle, assegurando que os containers estejam em execução em um pod, que consiste no menor e mais simples objeto do Kubernetes.

8. **(CESPE / SERPRO - 2021)** Para obter o status de um node nomeado como node1 em um cluster, deve ser executado o comando a seguir: Kubectl describe node node1

9. **(CESPE / SERPRO - 2021)** A camada de gerenciamento possui o componente etcd, cuja função é observar pods que foram criados sem nenhum node atribuído e selecionar um node para execução.

10. **(FGV / TJ-RO – 2021)** A equipe de desenvolvimento de sistemas de um tribunal de contas está guiando a implantação de um Webservice REST. A implantação será dividida nos seguintes grupos distintos de containers *Docker*:

- Grupo A: responsável pela execução da aplicação do Webservice REST
- Grupo B: responsável pela execução do Sistema Gerenciador de Banco de Dados utilizado pelo Webservice REST Os Grupos A e B terão seu próprio contexto de armazenamento e rede a serem orquestrados por um cluster de *Kubernetes.*

Para que a conexão de rede entre os containers dos Grupos A e B seja bem-sucedida pelo orquestrador, independentemente dos endereços IP a eles atribuídos, deverá ser configurado um novo:

a) pod
b) replicaSet
c) service
d) ingress
e) statefulSet

11. **(CESPE / STJ – 2015)** A orquestração automatiza a implantação, o gerenciamento, a escala e a rede dos contêineres. As ferramentas de orquestração de contêineres fornecem um *framework* para gerenciar arquiteturas de microsserviços e contêineres em escala, e muitas delas são usadas no gerenciamento do ciclo de vida dos contêineres; entre elas, o Docker Swarm é uma plataforma

a) que permite utilizar diversos recursos e ferramentas, como Apache e PHP, porém tudo rodando em um mesmo sistema operacional.

b) de código aberto criada pelo Google para operações de implantação de contêiner, aumento e redução e automação em clusters de *hosts.*

c) de orquestração de contêiner de código aberto, sendo o mecanismo de clusterização nativo para e pelo Docker, utilizando sua mesma linha de comando.

d) que roda sobre o Kubernetes instalado em sistema operacional na versão Enterprise da Red Hat, agregando opções de monitoramento, integração e entrega contínua.

e) usada pela Amazon para fornecer outros serviços aos clientes, como DNS, balanceamento, segurança e monitoramento, se integrando nativamente.

GABARITO – KUBERNETES

1. CORRETO
2. CORRETO
3. CORRETO
4. CORRETO
5. ERRADO
6. ERRADO
7. CORRETO
8. ANULADA
9. ERRADO
10. LETRA C
11. LETRA C

ESSA LEI TODO MUNDO CONHECE:
PIRATARIA É CRIME.

Mas é sempre bom revisar o porquê e como você pode ser prejudicado com essa prática.

1. 

Professor investe seu tempo para elaborar os cursos e o site os coloca à venda.

2. 

Pirata divulga ilicitamente (grupos de rateio), utilizando-se do anonimato, nomes falsos ou laranjas (geralmente o pirata se anuncia como formador de "grupos solidários" de rateio que não visam lucro).

3. 

Pirata cria alunos fake praticando falsidade ideológica, comprando cursos do site em nome de pessoas aleatórias (usando nome, CPF, endereço e telefone de terceiros sem autorização).

4. 

Pirata compra, muitas vezes, clonando cartões de crédito (por vezes o sistema anti-fraude não consegue identificar o golpe a tempo).

5. 

Pirata fere os Termos de Uso, adultera as aulas e retira a identificação dos arquivos PDF (justamente porque a atividade é ilegal e ele não quer que seus fakes sejam identificados).

6. 

Pirata revende as aulas protegidas por direitos autorais, praticando concorrência desleal e em flagrante desrespeito à Lei de Direitos Autorais (Lei 9.610/98).

7. 

Concurseiro(a) desinformado participa de rateio, achando que nada disso está acontecendo e esperando se tornar servidor público para exigir o cumprimento das leis.

8. 

O professor que elaborou o curso não ganha nada, o site não recebe nada, e a pessoa que praticou todos os ilícitos anteriores (pirata) fica com o lucro.

Deixando de lado esse mar de sujeira, aproveitamos para agradecer a todos que adquirem os cursos honestamente e permitem que o site continue existindo.