DOMINE DOCKER DO ZERO AO DEPLOY! | Curso Essencial

Aula 1 - INTRODUÇÃO

1. Introdução

O nome ***Docker*** vem do inglês e possui dois significados principais que se relacionam diretamente com a função da tecnologia: - **Estivador (*Docker*):** Em inglês, "docker" é um termo para "estivador", que é o profissional responsável por carregar e descarregar navios, especialmente os contêineres. Essa é uma analogia perfeita, já que o Docker, como plataforma, é o "estivador" que empacota, transporta e executa os "contêineres" de software. - **Contêiner (*Container*)**: Embora "docker" não signifique diretamente "contêiner", a tecnologia ***Docker*** é fundamentalmente sobre contêineres de *software*. Esses contêineres são pacotes padronizados que incluem tudo o que um *software* precisa para rodar (código, bibliotecas, configurações, etc.), garantindo que ele funcione de forma consistente em qualquer ambiente.

A logo do ***Docker***, uma baleia carregando contêineres, também reforça essa ideia de transporte e gerenciamento de contêineres, consolidando a analogia com o trabalho de um estivador.

Embora tanto ***Docker*** quanto máquinas virtuais (do Inglês, *Virtual Machines, VMs*) sejam tecnologias de virtualização que permitem isolar ambientes para a execução de softwares, eles operam em camadas diferentes e, por isso, possuem características e casos de uso distintos.

1.1 Máquinas Virtuais (VMs)

Uma **máquina virtual (MV)** é uma emulação de um computador físico completo. Pense nela como um computador dentro do seu computador.

- **Camada de Virtualização:** Uma MV virtualiza o *hardware* (CPU, memória, disco, rede). Para isso, ela usa um *software* chamado ***Hypervisor*** (ou monitor de MV). O *Hypervisor* cria e gerencia as *VMs*, alocando recursos do *hardware* físico para cada uma delas.
- **Sistema Operacional (SO) Convidado:** Cada MV executa um Sistema Operacional (SO) convidado completo (Windows, Linux, macOS, etc.) sobre o *Hypervisor*. Isso significa que cada MV tem seu próprio *kernel*, seus próprios processos de inicialização e suas próprias bibliotecas.
- **Isolamento:** As *VMs* oferecem um isolamento robusto. Se uma MV for comprometida por um vírus, por exemplo, as outras *VMs* no mesmo *host* não serão afetadas, pois cada uma tem seu próprio SO isolado.
- **Recursos:** Cada MV precisa de uma quantidade significativa de recursos (CPU, RAM, espaço em disco) para rodar seu SO convidado, mesmo que a aplicação em si seja pequena. Isso as torna mais "pesadas" e lentas para inicializar.

- **Portabilidade:** *VMs* são menos portáteis. Mover uma MV entre diferentes *hypervisors* ou provedores de nuvem pode ser um desafio e, muitas vezes, requer reconfiguração.

1.2 Docker (Contêineres)

O Docker é uma plataforma que utiliza a tecnologia de contêineres. Ao contrário das *VMs*, os contêineres não virtualizam o *hardware* completo, mas sim o sistema operacional.

- **Camada de Virtualização:** O Docker virtualiza a camada de aplicação e o espaço do usuário do sistema operacional. Ele roda diretamente sobre o SO *host* e compartilha o *kernel* do SO *host* entre os contêineres.
- **Sistema Operacional (SO) Convidado:** Contêineres não incluem um SO convidado completo. Eles empacotam apenas o código da aplicação, suas dependências, bibliotecas e configurações necessárias para rodar.
- **Isolamento:** O isolamento dos contêineres é feito através de recursos do próprio *kernel* do SO *host* (como *namespaces* e *cgroups*). Embora ofereçam um bom nível de isolamento, não é tão completo quanto o de uma MV, pois compartilham o mesmo *kernel*.
- **Recursos:** Contêineres são muito mais leves e eficientes em termos de recursos. Como compartilham o *kernel* do SO *host* e não precisam inicializar um SO completo, eles iniciam em segundos (ou milissegundos) e consomem menos CPU e memória. Isso permite rodar muito mais contêineres em uma mesma máquina física do que VMs.
- **Portabilidade:** Contêineres são altamente portáteis. Uma vez que uma aplicação é *"dockerizada"* (empacotada em um contêiner), ela pode ser executada de forma consistente em qualquer ambiente que tenha o Docker instalado (seu *laptop*, um servidor, a nuvem), independentemente do SO subjacente (desde que o Docker seja compatível com ele).

1.3 Conclusão

A escolha entre Docker e VMs depende das suas necessidades.

- Use ***VMs*** se você precisar de:
  - **Isolamento de segurança máximo** entre ambientes.
  - Executar **diferentes sistemas operacionais** no mesmo *hardware* (ex: Windows em um host Linux).
  - Compatibilidade com **aplicações legadas** que exigem um SO específico.
- Use **Docker** se você precisar de:
  - **Eficiência de recursos** e alta densidade de aplicações por host.
  - **Inicialização rápida** e escalabilidade ágil.
  - **Portabilidade** e consistência do ambiente de desenvolvimento para produção.
  - Trabalhar com **microsserviços** e DevOps.

É importante notar que **Docker pode ser executado dentro de uma Máquina Virtual**. Por exemplo, é comum executar o Docker Desktop em um Mac ou Windows, que por sua vez, usa uma MV Linux leve para rodar os contêineres. Isso combina a segurança da MV com a flexibilidade e eficiência dos contêineres.

---

2. Instalação do Docker no Linux

Instalar o Docker no Linux é um processo bem direto, mas os comandos exatos podem variar um pouco dependendo da distribuição que você usa (Ubuntu, Fedora, CentOS, etc.).

2.1 Antes de Começar: Desinstalar Versões Antigas

É uma boa prática remover qualquer instalação antiga do Docker para evitar conflitos. Use os comandos abaixo, se aplicável à sua distribuição:

Para Debian/Ubuntu:

Bash

```
sudo apt remove docker docker-engine docker.io containerd runc
```

Para CentOS/Fedora:

Bash

```
sudo yum remove docker docker-client docker-client-latest docker-
common docker-latest docker-latest-logrotate docker-logrotate docker-
engine
sudo dnf remove docker docker-client docker-client-latest docker-
common docker-latest docker-latest-logrotate docker-logrotate docker-
engine # Para Fedora mais recente
```

2.2 Processo de instalação:

A forma mais recomendada é usar o repositório oficial do Docker para garantir que sempre o recebimento das versões mais recentes e seguras.

Seguir o tutorial apresentado no seguinte endereço eletrônico: Install Docker Desktop on Linux