Docker - Infraestrutura como código, com autonomia e replicabilidade

[1]**Rafael Gomes,** [1]**Rodrigo Souza**

[1]Superintendência de Tecnologia da Informação – Universidade Federal da Bahia (UFBA)
Av. Adhemar de Barros, s/n – Campus Ondina – Salvador, BA – Brasil

{rafael.gomes, rodrigorgs}@ufba.br

Resumo. *O presente artigo demonstra a utilização da ferramenta Docker para disponibilização ágil, padronizada e altamente parametrizada de ambientes para desenvolvimento, teste, homologação de produção de soluções de TI. Essa nova estrutura oferece maior autonomia, documentação e replicabilidade para o desenvolvedor de software, que através do acesso a uma interface web, pode gerenciar parte dos ambientes disponibilizados.*
Palavras-chave*: Autonomia, Docker, Desenvolvimento, Ágil, Replicabilidade*

1. Introdução

No ambiente de TI a padronização é um desafio crescente e complexo, pois demandas de ambientes distintos e com versões de software diferentes são cada vez mais comuns e muitas vezes até estimuladas [1]. Tais demandas surgem por diversos motivos, seja pela falta de equipe para manter a solução nas condições especificadas em um padrão, ou pelo advento de um software de terceiro que, mesmo poupando trabalho, pode prejudicar a padronização do ambiente.

Este artigo apresenta uma arquitetura de provisionamento de plataforma como serviço para equipes de desenvolvimento através da ferramenta Docker. Demonstra-se os desafios de manter ambientes de desenvolvimento, seja para disponibilizar novas soluções, como para atualizar as existentes. Na Seção 3, a arquitetura do Docker é apresentada, em conjunto com outras ferramentas que facilitam o gerenciamento do ambiente. Na Seção 4 os resultados preliminares deste trabalho são apresentados. Por fim, na Seção 5, são apresentadas as conclusões e os trabalhos futuros.

2. Desafios para manutenção de ambientes

2.1. Velocidade na disponibilização e atualização de ambientes

A implantação de um novo ambiente para a equipe de desenvolvimento normalmente é fornecida pela equipe de infraestrutura da organização, que entre suas atribuições tem diversas outras tarefas e prioridades de execução.

Nesse cenário, qualquer nova solicitação de ambiente em uma infraestrutura não automatizada pode demorar o suficiente para ser um dos caminhos críticos de um projeto de desenvolvimento de software.

Quando o assunto é atualizar esse ambiente, a situação se agrava ainda mais, pois sem documentação confiável da infraestrutura aliado ao demasiado esforço necessário para se criar ambiente de simulação dessa atualização, geralmente opta-se por não atualizar a infraestrutura.

2.2. Múltiplos ambientes com diversas versões de software

No contexto de desenvolvimento de software, múltiplas versões de interpretadores e compiladores podem ser necessárias para atender a expectativa dos desenvolvedores, pois nem sempre é possível se limitar a uma determinada versão, uma vez que podem existir incompatibilidade de bibliotecas com determinadas versões de interpretadores e compiladores, assim como necessidade específicas de versões antigas.

Em alguns casos atualizar uma aplicação feita com uma versão antiga de um interpretador ou compilador é muito custoso, e assim opta-se por manter esse legado, dessa forma a padronização do ambiente pode ser comprometida.

Quando se fala em múltiplos ambientes, isso se materializa em várias máquinas, em alguns casos com diversas instâncias no mesmo servidor. No que tange ao uso dos recursos isso é um problema, pois são novos ativos com uma camada completa de sistema operacional, que muitas vezes é perfeitamente igual de uma máquina para outra.

3. Arquitetura da solução

A solução proposta usa as ferramentas Docker e Rancher, como explicado a seguir.

3.1. Docker

O Docker[1] é uma plataforma que automatiza a implantação de aplicações dentro de ambientes isolados denominados *containers* [1]. Trata-se, portanto, de uma solução para desenvolvedores e administradores de sistema desenvolverem, embarcarem, integrarem e executarem aplicações rapidamente. Seu principal objetivo é proporcionar múltiplos ambientes isolados dentro do mesmo servidor, mas acessíveis externamente via tradução de portas.

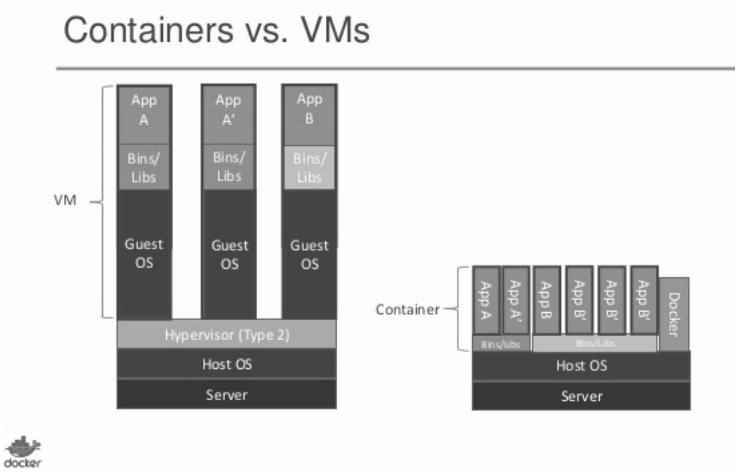

Figura 1. Comparação entre ambiente Docker e virtualizado

Como podemos perceber na Figura 1, a solução pode aparentar uma semelhança à estrutura de ambiente virtualizado, porém difere no que tange à necessidade de uma

[1]Disponível em `https://www.docker.com/`.

camada intermediária de sistema operacional entre o hospedeiro e as aplicações hospedadas, que no caso do Docker é desnecessária, pois ela utilizar o mesmo kernel, criando ambientes isolados a nível de disco, memória e processamento.

O Docker também estabelece um padrão de empacotamento de soluções. Ou seja, uma vez criada a estrutura do ambiente ela pode ser facilmente replicada, usada como referência para a criação de novas estruturas.

O `Dockerfile`, demonstrado na Listagem 1, é o arquivo que contém todos os comandos que normalmente seriam usados para configurar manualmente a implantação do ambiente. A imagem será criada com base nos parâmetros especificados nesse arquivo.

Listagem 1. ***Exemplo de arquivo*** **`Dockerfile`.**

```
FROM debian
MAINTAINER rafael.gomes@ufba.br
EXPOSE 8080

ENV TOMCAT_VERSION 7.0.62
ENV DEPLOY_DIR /maven

# Baixar e descompactar o Tomcat
RUN wget
http://archive.apache.org/dist/tomcat/tomcat-7/v\${TOMCAT_VERSION}/bin/
apache-tomcat-\${TOMCAT_VERSION}.tar.gz -O /tmp/catalina.tar.gz && tar xzf
/tmp/catalina.tar.gz -C /opt && ln -s /opt/apache-tomcat-\${TOMCAT_VERSION}
/opt/tomcat && rm /tmp/catalina.tar.gz

ENV CATALINA_HOME /opt/tomcat

ADD deploy-and-run.sh /opt/tomcat/bin/deploy-and-run.sh

CMD /opt/tomcat/bin/deploy-and-run.sh
```

3.2. Interface web para gerenciamento de *containers*

Para fornecer uma interface web de gerencia dos ambientes, mostrada na Figura 2, usamos o software Rancher[2]. A interface web de gerência de *containers* possibilita o controle dos ambientes de desenvolvimento e teste para os líderes da equipe de desenvolvedores.

3.3. Viabilizando infraestrutura rapidamente

Através da criação de uma infraestrutura replicável e de rápido provisionamento é possível disponibilizar ambientes de desenvolvimento, teste, homologação e produção para aplicações. Uma vez criada a imagem Docker, cada ambiente é instanciado com suas configurações particulares parametrizadas em variáveis de ambiente específicas para cada *container*.

Tendo acesso à interface é possível ligar, desligar e reiniciar *containers*, assim como criar novos *containers* a partir de imagens predefinidas, de plataformas previamente oferecidas.

4. Resultados alcançados

Controle completo dos ambientes instalados. Como cada plataforma é configurada através de parâmetros bem definidos, há uma maior confiança sobre o ambiente no qual a

[2]Disponível em `http://rancher.com/rancher-io/`.

aplicação está hospedada. Desta forma, a transição do ambiente de desenvolvimento até o ambiente de produção ocorre com maior confiança.

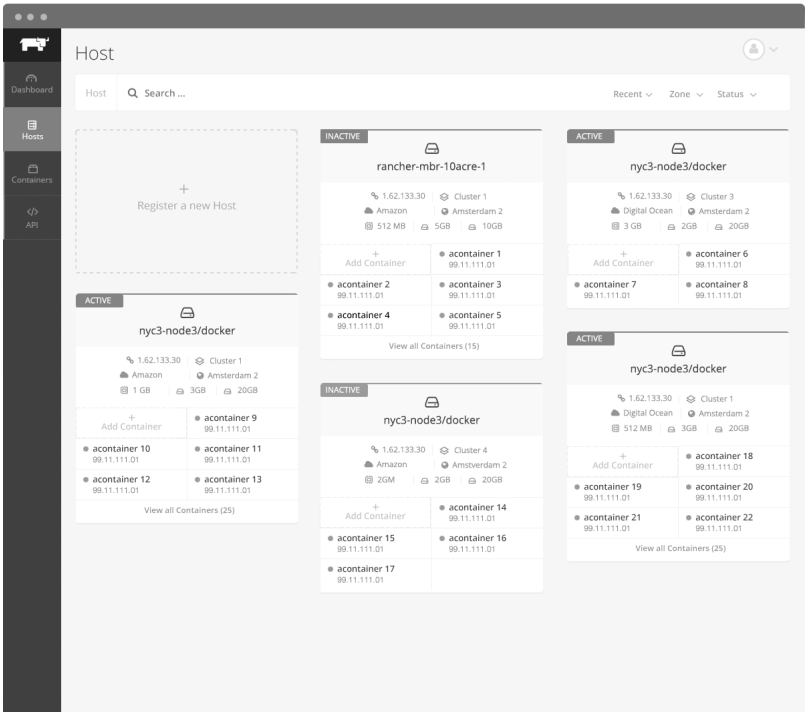

Figura 2. Exemplo da interface de gerenciamento

Autonomia da equipe de desenvolvimento A criação de um ambiente para aplicações web, desde a solicitação até a entrega, que normalmente demorava 2 dias, agora é feita em 2 minutos. O líder da equipe de desenvolvimento concentra essa atividade e assim a infraestrutura deixa de ser um possível caminho crítico para o projeto, ao menos no que tange à gerência da plataforma de desenvolvimento e teste, que são as que normalmente demandam uma maior urgência.

5. Conclusão e trabalhos futuros

Levando em consideração que o Dockerfile tem os parâmetros exatos para criação da máquina, podemos considerar que é um tipo de documentação que será sempre atual, assim como possibilitar a sua replicabilidade. Para os desenvolvedores foi possível proporcionar maior autonomia, o que impactou diretamente na velocidade do desenvolvimento, bem como uma segurança com relação ao ambiente usado desde o desenvolvimento até a produção.

Como outro trabalho futuro, planeja-se investigar a utilização do Docker como mecanismo de isolamento no provisionamento de ambientes web para sites de usuários da Universidade. Tal isolamento é importante para evitar que um site comprometido interfira no funcionamento dos demais, e viabilizá-la através de virtualização é inviável dado o custo de infraestrutura necessária para cada site.

Referências

[1] James Turnbull. *The Docker Book: Containerization is the new virtualization*. James Turnbull, 2014.