Introdução ao Docker – Basico

O que é , Vantagens , Porque devo usar ? , Ainda não sei se devo usar , Infraestrutura de servidores , Conhecendo a arquitetura , Entendendo o conceito , Comandos basicos , Criando imagem propria , Imagem com Dockerfile ,Listando imagens e containers, Subindo imagem ao Dockerhub , Armazenamento , Trabalhando com redes e Docker-compose

Docker – O que é?

- O docker é uma plataforma aberta
- Facilitar no desenvolvimento
- Agilizar a implementação
- E a possibilidade de ter aplicações em ambientes isolados
- Facilidade de gerenciar
- Agilidade para subir um novo ambiente de desenvolvimento
- Simplicidade de realizar modificações

Tudo isso ajuda muitos desenvolvedores e syadmins no seu dia a dia.

Docker - Vantagens

- Extremamente leve
- Isolado e ajudando com projetos legados
- Dependencias isoladas de cada container
- Facilicidade em criar e exportar ambientes
- Podemos ter diversas imagens e versões
- Facilidade em testar em outros ambientes

Docker – Porque devo usar ?

- Chega de na minha maquina funciona
- Facilidade em disponibilizar uma imagem
- Facilidade em criar um novo ambiente
- Facilidade em encontrar exemplos
- Docker é considerado um “idioma”

Docker - Ainda não sei se devo usar

Para o desenvolvedor

- Diversos ambientes , foco no desenvolvimento
- Não é necessário se preocupar com dependências
- Desenvolver uma vez e pode executar em qualquer local

Para o sysadmin

- Só configura uma vez e eliminar incertezas na entrega
- Infraestrutura escalavel , mais eficiente e agil

Para o Usuario

- Softwares em um ambiente isolado e evitar conflitos

Docker – Infraestrutura de servidores

Antigamente

- Maquina virtual tem um sistema operacional Projeto muito mais lento
- Pesado e até aumentando o custo

Containers

- Processos trabalhando de forma isolada
- Usa a funcionalidade de cgroups do kernel
- Funcionalidade do kernel denominada de namespaces

Docker – Conhecendo a arquitetura

Imagens

- Entender como formas de bolo ou template
- Imagens não são containers , mais dão a base
- Tem imagens oficiciais ou da comunidade
- Podemos armazenar no Dockerhub ou registry

Containers

- São baseados nas images
- Mantem o isolamento da aplicação e de recursos
- São como bolos prontos , precisamos de uma imagem

Docker – Entendendo o conceito

Docker Engine

- O docker engine é um daemon
- Nos auxilia na construção
- no envio
- executar nossos containers

Docker Client

- Já o Docker client é responsável por receber as entradas do usuário e as enviar para a engine.
  - Podemos ter um client e o engine na mesma maquina.
  - Porem eles podem ser executados em hosts diferentes.

Docker – Entendendo o conceito

Docker Registry

- O Docker registry é responsável por armazenar nossas imagens , temos o dockerhub ou até criar nosso próprio

Dockerfile

- O dockerfile é um arquivo que auxiliar na criação de uma imagem.
- É usado instruções e elas são aplicadas em uma determinada imagem para ue outra imagem seja criada baseada nas modificações.

Docker – Comandos basicos

Run

- Docker run é usado para iniciar um novo container
- sudo docker run --help

Pull

- Docker pull é resposavel por obter uma imagem
- sudo docker pull debian

Images

- Docker images é resposavel por listar as imagens que temos em nossa maquina
- Sudo docker images

Docker – Comandos basicos

Executando em background

- Já vimos como iniciar uma maquina com o **-d** podemos executar ele em background
- sudo docker run -d debian

Acesso a shell e excluindo quando sair

- Podemos excluir nosso container assim que terminar o uso usando docker run -it –-rm debian bash

Listando containers

- É possivel listar quais containers estão em funcionamento usando
- sudo docker ls

Docker – Comandos basicos

Passando um nome ao container

- Podemos dar o nome a um container usando
- sudo docker container run --name "web_server" httpd

Volumes

- Podemos realizar o mapeamento de volumes , lincar diretorios do host para o container
- docker container run -it --rm -v "$ {PWD}:/var/www/html" debian

Docker – Comandos basicos

Passando uma porta ao container

- Como o container é isolado precisamos passar uma porta para ele se comunicar
- Sudo docker container run -p8080:80 httpd

Conhecendo o exec

- Depois de um container estar funcionando , podemos interagir com ele usando o exec
- sudo docker exec -it web_server

Docker – Comandos basicos

Recursos (Ram e CPU)

- Ao iniciar um container é possivel setar alguns limites na ultilização dos recursos
- docker container run -it --rm -m 512M debian
- docker container run -it --rm -c 512 debian

Gerenciando containers

- sudo docker container start meu_container
- sudo docker container stop meu_container

Docker – Criando imagem propria

Pesquisando por imagens

- https://hub.docker.com
- https://hub.docker.com/u/greenmind/

As tags

- As TAGS são como versão de uma imagem
- https://hub.docker.com/_/debian?tab=tags

Docker – Criando imagem propria

Criando imagem usando commit

- Vamos usar o Debian
- Vamos instalar o NGINX nele
- sudo docker container run -it --name container_criado debian:9.7 bash
- apt-get update && apt-get install nginx -y && exit
- sudo docker container stop container_criado
- sudo docker commit container_criado "greenmind/debian:nginx"

Docker – Imagem com Dockerfile

Criando imagem

- A instrução FROM é resposavel por passar o nome da imagem e da tag que vamos usar no arquivo Dockerfile.
- From mysql:5.5

- O maintainer é a instrução resposavel por passar o resposavel por criar a imagem e pode ser usado caso alguem queria relatar algum problema/melhorar imagem. MAINTAINER email@contato.com

Docker – Imagem com Dockerfile

Criando imagem

- Essa instrução nos auxilia na troca de diretorios e podemos usar da seguinte forma.
- WORKDIR /root

- O copy é uma instrução usada para copiar algo para dentro da imagem
- COPY . /application

Docker – Imagem com Dockerfile

Criando imagem

- O comando run é resposavel por passar algum comando como instrução
- RUN apt-get update

- A instrução EXPOSE é usada para expor uma determinada porta da nossa imagem
- EXPOSE 80

Docker – Imagem com Dockerfile

Criando imagem usando commit

- O entrypoint é um argumento para ser usado caso queira usar sempre um comando ao iniciar o container
- ENTRYPOINT ["nmap"]

- A instrução CMD eu vejo ela como um complemento do ENTRYPOINT e caso não passe nada rodar o CMD
- ENTRYPOINT ["nmap"]
- CMD ["--version"]

Docker – Imagem com Dockerfile

Entendo a ordem das instruções

- Mão na massa
- https://github.com/ABase-BR/Docker-pentesters/tree/master/pompem

Docker – Listando imagens e containers

Listando imagens

- sudo docker images
- Repository , Tag , Image , Created e size

Listando containers

- sudo docker ps
- Container ID Image , Command , Created , Status , Ports,Names

Docker – Subindo imagem ao Dockerhub

- Conhecendo o Dockerhub
- Criando conta no Dockerhub
- Entendo sobre stars
- Imagem oficial e não oficial
- Subindo imagem para o Dockerhub
- Obtendo imagem usando pull

Docker – Armazenamento

Host directoty as a data volume

- Volume é salvo no Host.
- Não é escalável
- sudo docker run -d --name server_web -v /home/user/site:/usr/local/apache2/htdocs -p 80:80 httpd

- https://github.com/ABase-BR/Docker-intro/tree/master/11-Trabalhando-com-armazenamento

Docker – Armazenamento

Data-only Container

- O volume é portavel e não é atrelado ao host.
- Volatil caso container seja removido ou falhar.
- Iniciamente vamos usar o docker create , ele é resposavel por criar nosso container e deixar ele em stand by.

- sudo docker create -v "/usr/local/apache2/htdocs" --name datasite ubuntu:14.04
- docker run --name web_volumes -d --volumes-from datasite -p 9092:80 httpd

Docker – Armazenamento

Shared-store Volume

- Ele tem um problema , maior overhead de disco.
- Porem ele tem uma maior segurança dos dados.
- Compativel com plugin local , NFS , CIFS e Cluster.
- sudo docker volume create --name data
- sudo docker volume ls
- sudo docker run -it --name cont_volume -v "data:/tmp" ubuntu:14.04

Docker – Trabalhando com redes

Linking - Linkando containers

- Simples
- Porem seu uso é considerado absoleto e possivelmente as proximas versões nao vão suportar mais.
- Ele é muito usado para disponibilizar microserviços.

Docker – Trabalhando com redes

Linking - Linkando containers

- docker run -d --name database -e MYSQL_ROOT_PASSWORD=123 -e MYSQL_DATABASE=teste -e MYSQL_USER=user -e MYSQL_PASSWORD=pass mysql:5.5
- sudo docker run -d -p 8080:80 --name php --link database:db -v /home/user/site:/var/www/html php:5.6-apache

Docker – Trabalhando com redes

User-defined networks

- Esse metodo é o sucesso do linking , temos a possibilidade de criar diversas redes proprias e ainda permitir a comunicação entre containers atraves do uso de nomes.

•

- É um metodo seguro , a rede isolada para comunicar dos containers.
- Acaba sendo um pouco complexo pois precisamos criar nossas redes bridges.

Docker – Trabalhando com redes

User-defined networks

- Primeiro vamos criar a rede
- sudo docker network create redeA

- Em seguida podemos criar duas maquinas para se comunicar com a redeA.
- sudo docker run -itd --name ubuntu --network=redeA ubuntu
- sudo docker run -itd --name ubuntu2 --network=redeA ubuntu

Docker – Trabalhando com redes

User-defined networks

- Vamos desconectar as duas maquinas que criamos.
- sudo docker network disconnect redeA ubuntu
- sudo docker network disconnect redeA ubuntu2

- Depois de desconectar as duas maquinas podemos usar
- sudo docker network rm redeA
- Podemos checar se for removida usado

- sudo docker network ls

Docker – Usando o docker-compose

Docker compose

- O docker compose é uma ferramenta para nos auxiliar na criação de múltiplos containers Docker
- Podemos definir determinados serviços , podemos setar portas abertas , variáveis de ambiente , volumes , configurar redes e muitas possibilidade que não conseguimos apenas com o Dockerfile.

- https://hub.docker.com/_/wordpress

Docker para pentesters

Dirb , nmap , netcat , recon , mysql-client , ftp-client , openssl ,openvas , nessus , Pompem , CeWL , Dirsearch , DVWA , Amass , sharingmyip.

Docker – Dirb

- O Dirb é um projeto que tem como objetivo realizar o reconhecimento de diretorios usando brute force
- Conta com wordlist padrão
- Podemos usar a nossa wordlist personalizada

Docker - Nmap

NMAP nos ajuda em

- Scan de portas
- Scanner
- Ataques a formularios
- E uma infinidade de soluções
- Junto com o NMAP temos o NSE

Site oficial e documentação

- nmap.org/nsedoc/
- https://nmap.org/book/nse-usage.html

Docker - Netcat

- Netcat é considerado o canivete suíço TCP/IP
- Realiza conexões
- Scanner de portas
- Buscando headers

Docker – Recon

- + de 18 ferramentas para ajudar no reconhecimento
- Whois,Nmap,Netcat,Fierce,Shodan,DnsRecon,TCP Dump e muitas outras

Docker – Mysql client

- Podemos usar uma imagem com o cliente mysql
- Podemos se conectar a um banco
- Enumerar serviços
- Não tendo a necessidade de instalar um client

Docker – FTP Client

- Podemos usar um client FTP para enumerar serviços
- Não temos a necessidade de instalar

Docker - OpenSSL

- Podemos usar um cliente openssl para analisar sites com HTTPS e assim facilitando o uso em testes
- Não temos a necessidade de instalar no host

Docker - Nessus

- Nessus é uma ferramenta que nos auxilia com a analise de vulnerabilidades

Pontos positivos

- Não é instalado em nossa maquina
- Podemos criar mais de uma maquina

Pontos negativos

- Podemos ser prejudicados devida a performance

Docker - OpenVas

- OpenVAS é um framework de vários serviços
- Oferece uma solução de varredura e Gerenciamento de vulnerabilidade
- Software livre !

Docker – Pompem

- O Pompem é desenvolvido em Python
- Auxilia na busca de exploits públicos

Fontes de informações

- PacketStorm security
- CXSecurity
- ZeroDay
- Vulners
- National Vulnerability Database
- WPScan Vulnerability Database

Docker - CeWL

- O CeWL é um projeto que tem como meta auxiliar na criação de wordlists personalisadas
- Ferramenta desenvolvida em Ruby

Veja o projeto no Github

- https://github.com/digininja/CeWL/

Docker - Dirsearch

- O dirsearch realiza o brute force em aplicações

Nos ajuda a encontrar

- Diretórios
- Arquivos
- Paginas
- Muito mais rápido que o Dirb

Link do projeto

- https://github.com/maurosoria/dirsearch

Docker - DVWA

- DVWA é um projeto vulneravel que tem a meta de ajudar profissionais e estudantes a estudar tecnicas de web hacking.
- https://github.com/opsxcq/docker-vulnerable-dvwa/
- Docker build -t “dvwa:1” .
- Docker run -p 80:80 “dvwa:1”

Docker - Amass

- O Amass nos ajuda na enumeração de alvos usando OSINT e fontes de informações
- Enumeração DNS
- Mapeamento de Rede
- Subdominios
- Netblocks e ASNs
- DNS
- Web Archive , APIs , Certificados

Docker - Sharingmyip

- Sharingmyip é um site que nos auxilia mostrando quais sites dividem o mesmo IP
- Script desenvolvido em Python