



Unioeste - Universidade Estadual do Oeste do Paraná
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
Colegiado de Ciência da Computação
Curso de Bacharelado em Ciência da Computação

AP02 - Implementação de Sistemas de Containers
RunC, LXC, Docker e Podman

Tecnologia de Análise de Sistemas

Aluno: Fernando Seiji Onoda Inomata
Turma: 2º Ano
Professor: Leonardo Medeiros

7 de Dezembro de 2023

Etapa 1 - Runc

Instalação do Runc:

```
sudo apt install runc;
```

Criação de diretórios:

```
Runc, rootfs, ambos usando Mkdir;
```

Instalação do SO no diretório /rootfs:

```
sudo debootstrap stable ./rootfs
```

Criação do arquivo config.json:

```
runc spec;
```

Alterações no Arquivo .json:

```
"terminal": true -> "terminal": false;  
"args": [          "args": [  
    "sh"           ->    "sleep", "5"  
],                ],
```

Arquivo .json criado e alterado, e diretórios já criados; Criação do container:

No Diretório Runc:

Criar Container:

```
sudo runc create ContainerRunc;
```

Verificação da Criação:

```
sudo runc list;
```

Inicialização do Container:

```
sudo runc start ContainerRunc;
```

Verificação da Criação:

```
sudo runc list;
```

Deletar o Container:

```
sudo runc delete ContainerRunc;
```

Com isso, consegui mexer na configuração do root utilizando o “sudo debootstrap stable ./rootfs”, e nas configurações do config.json;

Etapa 2 - LXC

Instalação do VirtManager:

```
sudo apt-get install lxc  
sudo apt install lxd-installer  
sudo apt install virt-manager libvirt-daemon-system libvirt-clients  
sudo usermod -aG libvirt $(whoami)
```

Iniciar o LibVirt:

```
sudo systemctl start libvirtd  
sudo systemctl enable libvirtd
```

Criei o container:

```
sudo lxc-create -t download -n "ContainerLXC"
```

Durante a Criação, é nos pedido 3 informações:

```
Distribution: busybox  
Release: 1.36.1  
Architecture: amd64
```

Iniciei o Container:

```
sudo lxc-start -n 'ContainerLXC'
```

E Iniciei o Console do Container:

```
sudo lxc-console -n 'ContainerLXC'
```

Tentei mas não consegui continuar...;

Etapa 3 - Dockers

Primeiramente comecei realizando a instalação do Docker, e utilizando o seguinte comando disponibilizado no site deles:

```
sudo apt-get update
```

```
sudo apt-get install ca-certificates curl gnupg
```

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor  
-o /etc/apt/keyrings/docker.gpg
```

```
sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

```
echo \  
"deb [arch=$(dpkg --print-architecture)  
signed-by=/etc/apt/keyrings/docker.gpg]  
https://download.docker.com/linux/ubuntu \  
$(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \  
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null  
sudo apt-get update";
```

Logo em seguida utilizei esse outro “sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin” para instalar a última versão e para verificar “sudo docker run hello-world”

```
“sudo systemctl enable --now docker docker.socket containerd”
```

```
“sudo su”
```

```
“docker pull wordpress” e “docker pull ubuntu”
```

Após esses comandos, ao digitar “docker images” aparece 3 repositórios, o wordpress, o ubuntu e o hello-world, com essas imagens, e ainda no modo root, fui até o diretório que tinha criado para realizar o Docker e digitei o seguinte “docker run --name teste -p 8080:80 -d wordpress” e usando o comando “docker stop [IDdoContainer]”, conseguimos parar a execução do container e se trocar o “stop” por “start”, vai iniciá-lo novamente; Digitando no navegador, “<http://localhost:8080>”, abriu a tela do wordpress, onde aparece para escolher a linguagem e para logar nele.

Etapa 4 - Podman

Instalação:

```
sudo apt-get install -y podman
```

Verificação da Versão do Podman:

```
podman --version (3.4.4)
```

Entrando no Podman:

```
podman run -it --rm alpine /bin/sh
```

Para adicionar arquivos dentro do Podman:

```
apk add
```

Verificação da Versão do Alpine:

```
cat /etc/os-release (3.18.5)
```

Verificação do Container:

```
podman ps
```

Tentativa de criar o Container:

```
podman run -d -p 8080:80 nginx
```

Porém já tinha cadastrado um container nesse id, então:

```
sudo docker ps -> Para ver se era nesse que estava o  
container, e era, peguei o id;
```

o container antes...

```
sudo docker stop <id>  
sudo docker rm <id>
```

```
podman run -d -p 8080:80 nginx
```

Testando com "<http://localhost:8080>", abre a página inicial da nginx;

Etapa 5 - Github e CodeBerg

Github

Inicialmente verifiquei se já havia instalado o git na minha máquina:

```
git --version (2.34.1)
```

E seguindo as etapas do próprio github, criei um Readme.txt:

```
echo "# FernandoInomata-TDS-AP02" >> README.md  
git init  
git add README.md
```

Comecei adicionando a pasta do Docker:

```
git add Docker/  
git push -u origin main
```

Quando digitei o segundo comando, pediu para digitar o nome do usuário e a senha, porém ao digitar a senha, aparecia a seguinte mensagem:

```
remote: Support for password authentication was removed on  
August 13, 2021.
```

```
remote: Please see
```

```
https://docs.github.com/en/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for information on currently  
recommended modes of authentication.
```

fatal: Authentication failed for
'https://github.com/FernandoInomata/FernandoInomata-TDS-AP02.git/'

Para resolver isso, tive que criar um Token no github, depois de criar ele coloquei no lugar da senha e funcionou;

Refiz o mesmo processo para colocar a pasta do Runc no github

Link: <https://github.com/FernandoInomata/FernandoInomata-TDS-AP02>

Codeberg

Para o Codeberg foi mais simples, o próprio codeberg já tem uma função para importar projetos do github direto pra ele.

Link: <https://codeberg.org/FernandoInomata/FernandoInomata-TDS-AP02>