Do confinamento de processos ao microservices com Docker e Kubernetes

William Lima Reiznautt
william@unicamp.br

Do confinamento de processos ao microservices com Docker e Kubernetes

- Confinamento de processos
- Docker
  - Engine
  - Imagens
  - Execução e dicas
  - Dockerfile
  - Stack / Compose
  - Introdução Swarm
- Docker vs Containerd
- Microservices
- Kubernetes
  - Engine e Arquitetura
  - CNI
  - Pods
  - Services
  - Deployment
  - Replicaset
  - Volumes persistente
  - AutoScale e HPA
  - Sidecar
- Service Mesh
  - Logs e Monitoramento
  - Ingress

O que é virtualização ?

O que é virtualização?

Particionamento de recursos, dividindo uma única máquina física em múltiplas "máquinas lógicas", mais conhecidas como máquinas virtuais.

Cada máquina virtual pode rodar um sistema operacional com seus aplicativos de maneira independente e isolada, ou até mesmo rodar apenas aplicativos isoladamente.

- Virtualização de Sistema Operacional.
- Virtualização de aplicação / linguagens de programação.
- Emulação de hardware.

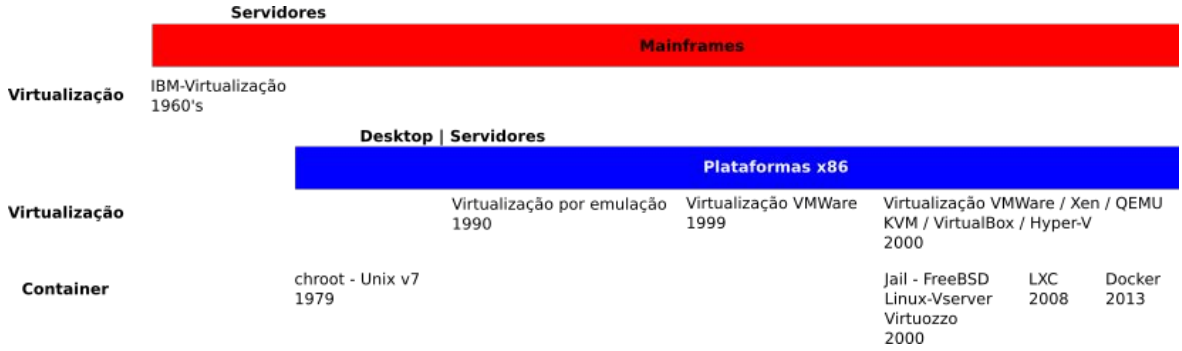
Jargões

Jargões:

Host - Máquina física
Guest - Máquina virtual
Container - Conjunto de processos confinado

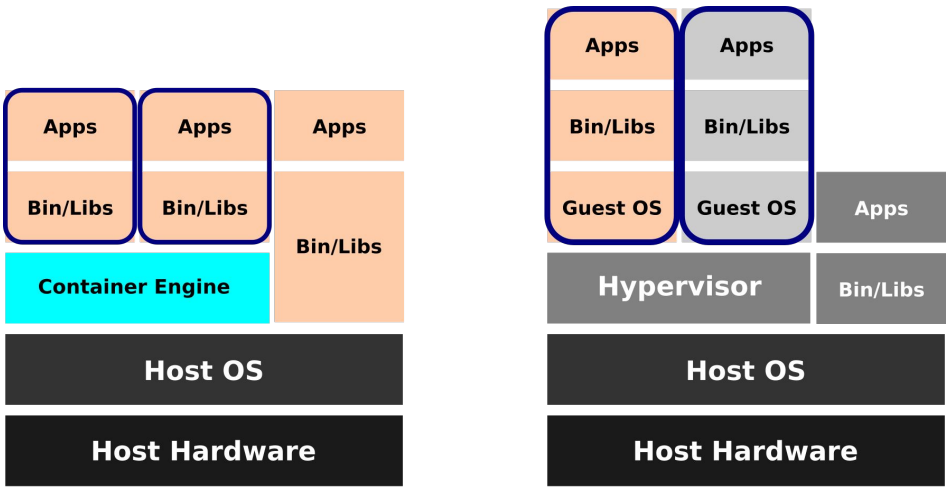
História e evolução da Virtualização e Container

Timeline História da Virtualização e Container



IBM * Time sharing

Virtualização e Container



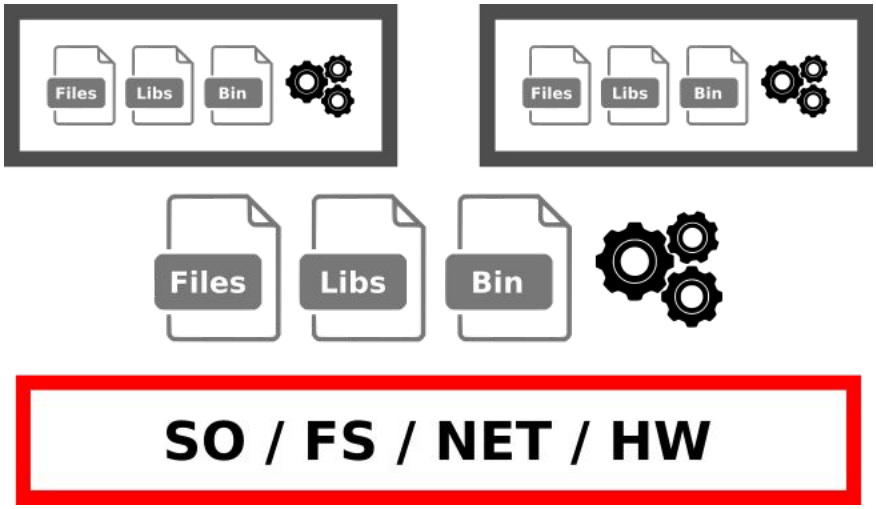
Container

Virtualização em nível de sistema operacional

Criação de um isolamento “confinamento” de um ou mais processos a um ou mais recurso do Host

Introdução ao confinamento de processos (container)

- **O que é o confinamento de processo ?**



Introdução ao confinamento de processos (container)

- **O que é o confinamento de processo ?**

Tabela de Processos

```
systemd(1)─┬─ModemManager(531)─┬─{gdbus}(588)
           │                   └─{gmain}(586)
           ├─NetworkManager(544)─┬─dnsmasq(708)
           │                     ├─{gdbus}(687)
           │                     └─{gmain}(685)
           ├─accounts-daemon(506)─┬─{gdbus}(512)
           │                      └─{gmain}(510)
           ├─agetty(1164)
           ├─anacron(520)
           ├─apache2(1030)──apache2(1033)─┬─{apache2}(1041)
           │                              └─{apache2}(1042)
           └─ bash (2000) ──{apache2}(2001)
```

Confinamento

```
bash (2000) ──{apache2}(2001)
```

Confinamento de processos (container)
chroot

chroot (criado em 1979 – Unix v7)

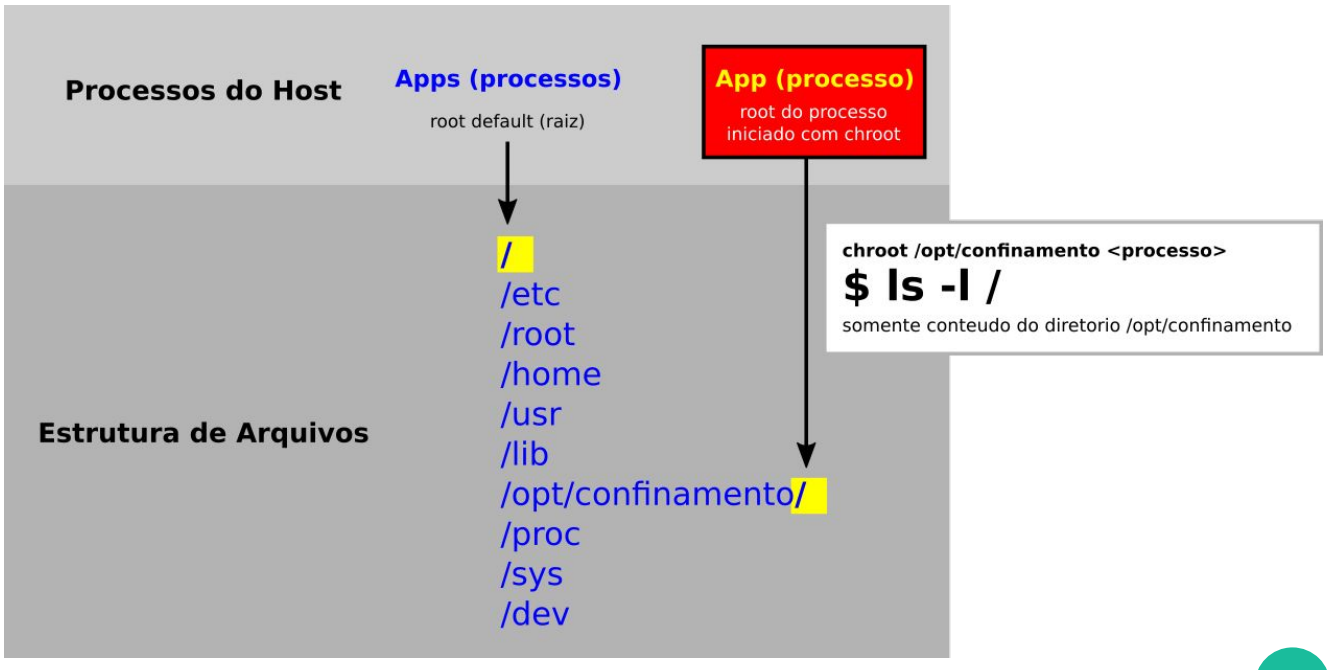
Foi criado no intuito de segregar acessos de diretórios e evitar que o usuário possa ter acesso à estrutura raiz (“/” ou root), ou seja isolando qualquer processo em um diretório

Porém não segregava rede e nem limitava os acessos de superusuários aos processos, Isso foi implementado mais tarde pelo JAIL do FreeBSD no anos 2000.

** JAIL não fazia limitação de recurso (cpu/memoria)

Confinamento de processos (container)
chroot

chroot



Confinamento de processos (container)
chroot

Exemplo chroot (somente bash)

- Criando o diretório para seu container

```
# mkdir -p /container/test/lib64 /container/test/bin
/container/test/lib/x86_64-linux-gnu
```

- Copiando um programa para seu container

```
# cp -l $(ldd /bin/bash | egrep -o "/lib64/.* ") /container/test/lib64/
# cp /bin/bash /container/test/bin/

# cp -l $(ldd /bin/ls | egrep -o "/lib64/.* ") /container/test/lib64/
# cp /bin/ls /container/test/bin/

# cp /lib/x86_64-linux-gnu/*.*  /container/test/lib/x86_64-linux-gnu
```

- Acessando seu container (ambiente confinado)

```
# chroot /container/test/
# export PATH=/bin
```

** Explore os comandos (pwd / ls / cd) dentro e fora do container

Confinamento de processos (container)
namespace

Segregação de recurso por processos e sub-processo (Kernel).

- Namespace é são um recurso do kernel do Linux que particiona os recursos do kernel, de modo que um conjunto de processos vê um conjunto de recursos, enquanto outro conjunto de processos vê um conjunto diferente de recursos.
- Namespaces são um aspecto fundamental dos contêineres no Linux.
- Um sistema Linux começa com um único namespace de cada tipo, usado por todos os processos. Os processos podem criar namespaces adicionais e ingressar em diferentes namespaces.
- Os Namespaces do Linux foram criados em 2002 no kernel 2.4.19

Confinamento de processos (container)
namespace

A partir do kernel versão 4.10, existem 7 tipos de namespaces.

Mount (mnt) : Pontos de montagem /
ID do processo (PID) : sub-processos isolados e sub-ids.
Network (NET) : uma nova pilha de rede.
Comunicação Interprocessual (IPC): no-root acesso entre processo
UTS : Nomes de host e domínio diferentes para os processos.
User ID (USER) : Mapa de identificador de usuários diferentes.
Grupo de controle (cgroup) : Permissões e controle a recursos.

http://man7.org/linux/man-pages/man7/namespaces.7.html

Confinamento de processos (container)
cgroup

Cgroup : Histórico

- Começou a ser desenvolvido em 2006 pela Google.
- Entrou no kernel Linux 2.6.24 e ajudou a adoção do LXC.
- Algumas características ainda estão em desenvolvimento.

Confinamento de processos (container)
cgroup

Cgroup : Funcionalidade

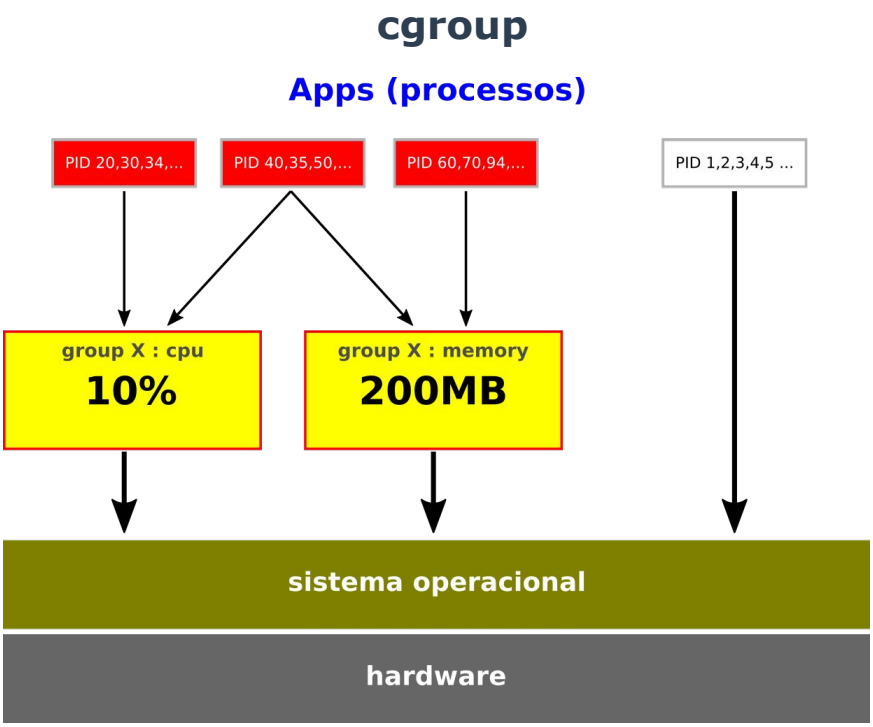
- Controle de acesso restrito a dispositivos, e quais recursos podem ser usados por cada cgroup.
- Limitação de Recursos: memory, CPU, device accessibility, block I/O, etc.
- Priorização: quem obtém mais prioridade na CPU, no acesso a memória, etc.
- Contabilização: uso dos recursos por cgroup
- Controle: permite o congelamento e checkpoint
- Injeção: marcação de pacotes

Confinamento de processos (container)
cgroup

Cgroup : Uso

- A funcionalidade cgroup é exposta pelo sistema operacional como “controladores de recursos” (em “subsystems”
- Esse recurso de Subsystems é montado com um Sistema de Arquivo
- O subsystem montado no nível-top é o cgroup “root”, que é global para todos os processos do host
- Os sub-diretórios a partir do top-level são criados por cgroup.
- Processos são colocados no arquivo “tasks” para associar ao group.
- A interface é feita via leitura / escrita nos arquivos do cgroup.

Confinamento de processos (container)
cgroup



Confinamento de processos (container)
cgroup

Limitação de recurso para processos (cgroup).

```
# apt install cgroup-tools
```

- Criando estrutura de segmentação

```
# cgcreate -g cpu:/teste
```

- Atribuindo limites de recurso memória e cpu

```
# echo 100 > /sys/fs/cgroup/cpu/teste/cpu.shares
```

- Executando a aplicação com a limitação.

```
# cgexec -g cpu:teste /root/math.py &

# cat /root/math.py
#!/usr/bin/python
while True: 42*42
```

Confinamento de processos (container)
cgroup

Limitação de recurso para processos (cgroup).

- Criando estrutura de segmentação.

```
# cgcreate -g memory:/teste2
```

- Atribuindo limites de recurso de memória.

```
# echo 300000000 > /sys/fs/cgroup/memory/teste2/memory.limit_in_bytes
```

- Executando a aplicação com a limitação de memory.

```
# curl www.ic.unicamp.br/~william/scripts/memory.py > memory.py
# chmod 755 memory.py
# cgexec -g memory:teste2 ./memory.py
```

Confinamento de processos (container)
cgroup

- Formas para atribuir limites de recurso a um grupo:

```
# echo 100 > /sys/fs/cgroup/cpu/teste/cpu.shares
```

- ou

```
# cgset -r cpu.shares=512 meu-container
```

- Executando a aplicação com a limitação de cpu e memoria.

```
# cgexec -g memory,cpuset,cpu:meu-container <aplicação> <argumentos>
```

Confinamento de processos (container)
cgroup

Alguns subsistemas disponíveis do CGROUP:

blkio - limites no acesso de entrada / saída para dispositivos de bloco (discos ou USB).
cpu – limites no scheduler da CPU.
cpuacct - relatórios automáticos dos recursos da CPU usados pelas tarefas no CGROUP.
cpuset - atribui CPUs individuais (em um sistema multicore) e nós de memória a tarefas em um cgroup.
devices - permite ou nega acesso a dispositivos por tarefas em um cgroup.
freezer - suspende ou retoma tarefas em um cgroup.
memory - define limites no uso da memória por tarefas em um cgroup e gera relatórios automáticos sobre os recursos de memória usados por essas tarefas.
net_cls - marca os pacotes de rede com um identificador de classe (classid) que permite que o controlador de tráfego Linux (tc) identifique pacotes originados de uma tarefa cgroup específica.
net_prio - fornece uma maneira de definir dinamicamente a prioridade do tráfego de rede por interface de rede.
ns - o subsistema dos namespace.
perf_event - identifica a associação de grupos de tarefas e pode ser usado para análise de desempenho.

Confinamento de processos (container)
segragação de recursos

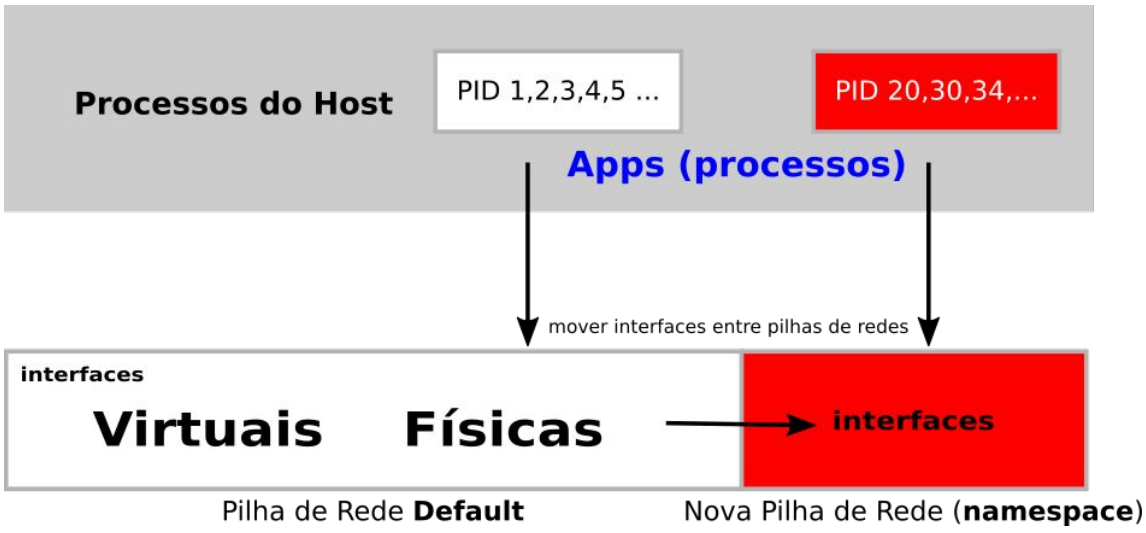
Segregação de recursos

- **Cgroup**
  (limitar recurso de CPU, memória e I/O de discos)
- **Namespace ID Processo**
  (Nova hierarquia da arvore de processo)
- **Namespace / chroot / selinux / apparmor**
  (Novo root na estrutura de arquivos “/”)
- **Namespace Network**
  (É criado um outra pilha de rede dentro do kernel)

Todas essas características são implementas no kernel do Linux

Confinamento de processos (container)
namespace network

Namespace Network



Confinamento de processos (container)
namespace network

Segregação de redes (*namespace network - netns*).

- O namespace de rede cria uma *nova pilha de rede*.
- Na criação, um namespace de rede contém apenas uma interface de *loopback*.
- Cada interface de rede (física ou virtual) está presente em exatamente 1 namespace e pode ser *movida entre namespaces*.
- Cada namespace terá um conjunto particular de *endereços IP*, com sua própria *tabela de roteamento*, *listagem de soquete*, *tabela de controle de conexão, firewall* e outros recursos relacionados à rede.
- Em sua destruição, um namespace de rede destruirá quaisquer interfaces virtuais dentro dele e moverá quaisquer interfaces físicas de volta para o namespace de rede inicial.

Confinamento de processos (container)
namespace network

Segregação de redes (namespace).

- Criando uma nova pilha de rede

```
# ip netns add net-meu-container
```

- Criando duas interfaces virtuais interligadas

```
# ip link add veth0 type veth peer name veth1
```

- Incluindo interface dentro da pilha de rede

```
# ip link set veth1 netns net-meu-container
```

- Executando aplicações dentro da pilha de rede

```
# ip netns exec net-meu-container /bin/bash
```

Confinamento de processos (container)
exemplo

Exemplo de um “container”
com segregação de recurso e uma pilha de rede.

```
# cgexec -g memory,cpu:net-meu-container \
   ip netns exec net-meu-container \
    chroot /container/meu-container/ /bin/bash
```

Confinamento de processos (container)
exemplo

Exemplo de como criar um container Linux a partir de uma ISO

```
# mkdir /isos ; cd /isos
# wget http://mirror.ufscar.br/centos/7/isos/x86_64/CentOS-7-x86_64-NetInstall-1708.iso

# mkdir /mnt/centosiso /mnt/squash /mnt/root
```

- Montando a estrutura da ISO

```
# mount -o loop -o ro /isos/CentOS*.iso /mnt/centosiso/
# mount -o loop -o ro /mnt/centosiso/LiveOS/squashfs.img /mnt/squash/
# mount -o loop -o ro /mnt/squash/LiveOS/rootfs.img /mnt/root/
```

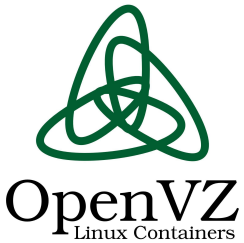
- Copiando a estrutura básica do SO

```
# cp -aR /mnt/root/* /<seu caminho>/seu-container/
```

Fedora e Centos em : http://legolas.lab.ic.unicamp.br/containers/

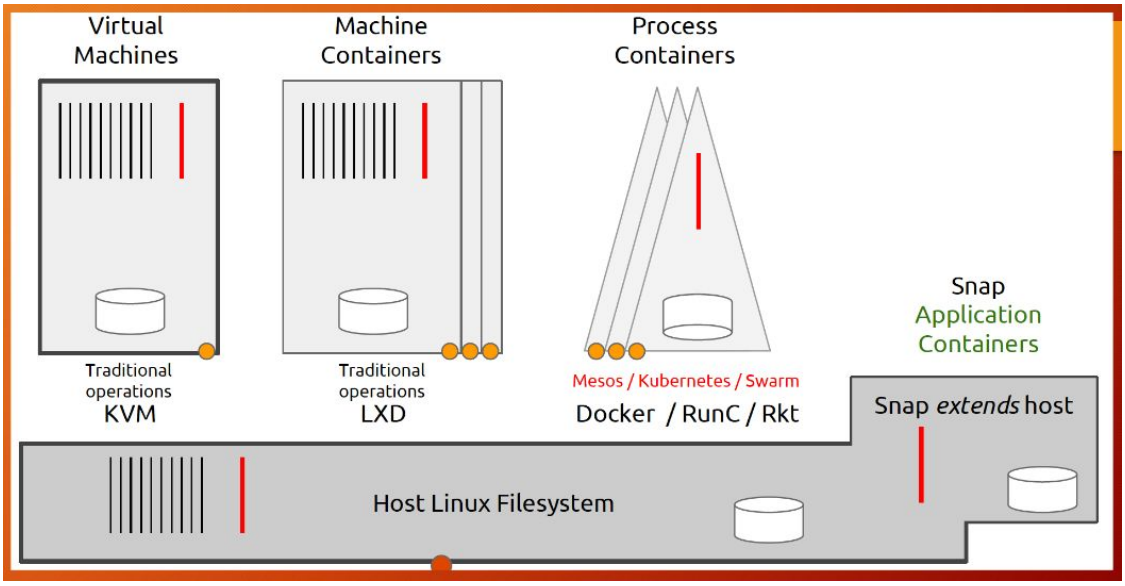
Confinamento de processos (container)
softwares para container

Ferramentas para deploy/gerenciamento de container

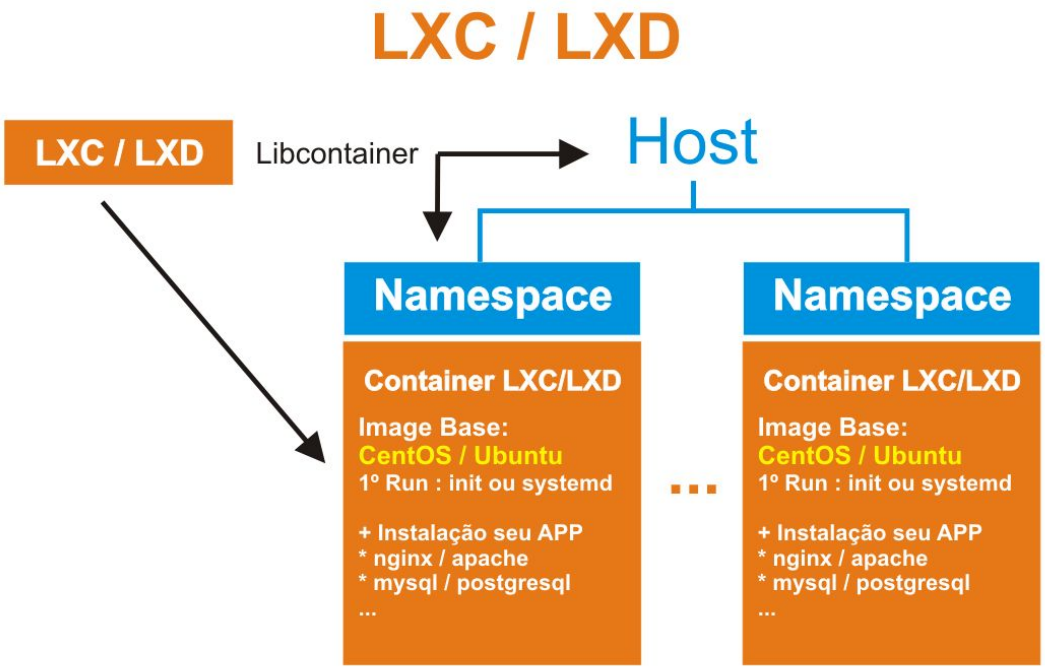


Confinamento de processos (container)
VMs vs Linux Container vs App Container

VM / Containers



Confinamento de processos (container)
LXC / LXD



Confinamento de processos (container)
Docker