

DESVENDANDO O FUNCIONAMENTO POR TRÁS DO DOCKER

**Palestrante:
Pedro Filho**



QUEM É O PALESTRANTE?

Professor Mestre e pesquisador do Instituto Federal da Paraíba (IFPB) desde 2014. Tem mais de 20 anos de experiência na área de Redes de Computadores executando e construindo projetos que envolvem sistemas virtualizados, roteamento, linux, práticas de segurança, cloud, Devops, entre outros.

Também é membro do grupo OWASP da paraíba

Instrutor oficial da academia CISCO com habilitação para formar outro instrutores.



<https://www.linkedin.com/in/pedro-batista-de-carvalho-filho-92b95768/>



+55 (83) 98725-0006





Docker



HISTÓRIA DAS TECNOLOGIAS DE CONTAINERS NO LINUX



UNIX V7
→ CHROOT

1979



FREEBSD
JAIL

2000



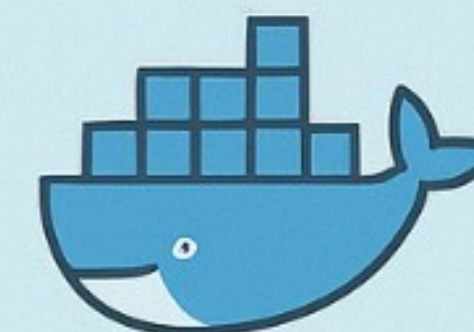
NAMESPACES
(MNT, UTS, IPC,
PID, NET, USER)

2002-2013



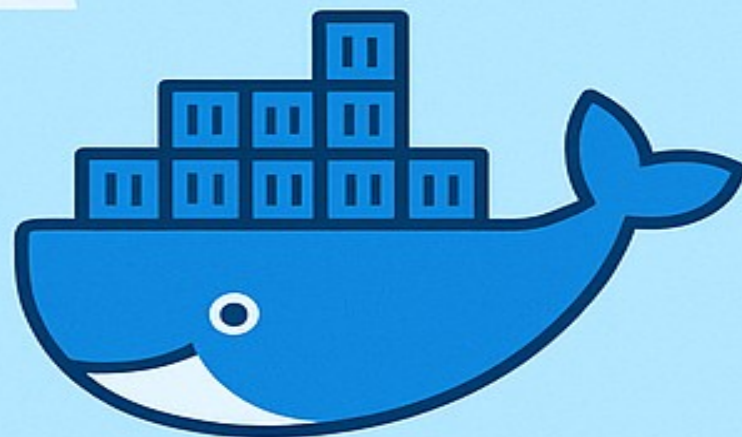
CGROUPS
LXC

2006-2007



DOCKER

2013



Docker no Linux

Namespaces

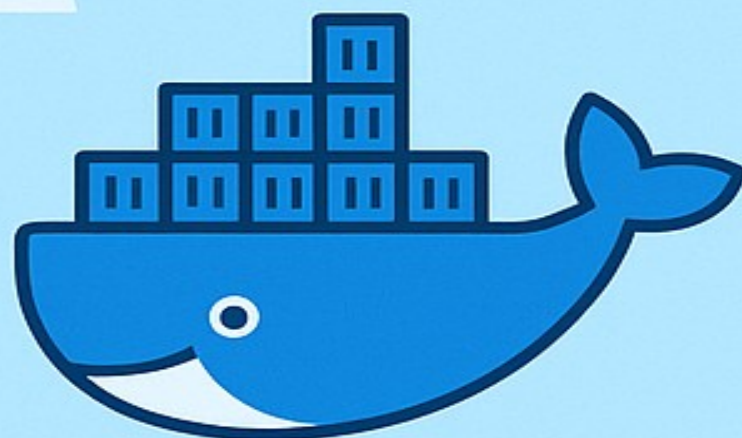
cgroups

Capabilities

Seccomp

chroot

UnionFS



Docker no Linux

Isolamento de
recursos



Namespaces

cgroups

Capabilities

Seccomp

chroot

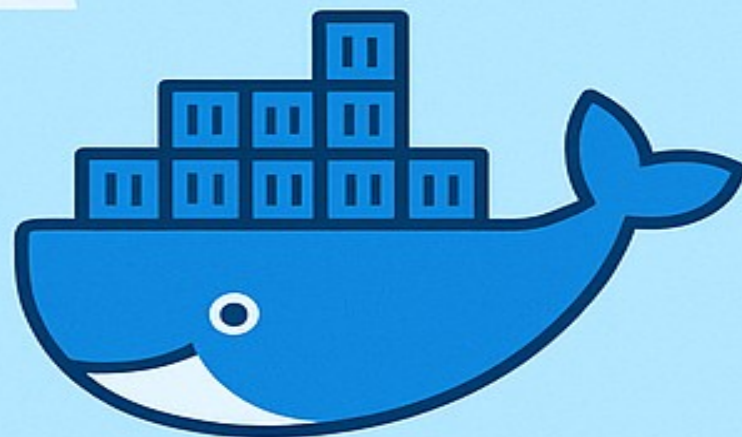
UnionFS



Segurita
y

Proporciona dinamismo
entre imagens e container





Docker no Linux

Nosso foco
será em

Namespaces

cgroups

Capabilities

Seccomp

chroot

UnionFS

Do dockerfile ao container ...



Camadas da imagem Docker

Dockerfile

```
FROM debian:latest
RUN echo "Texto" > /file01.txt
CMD ["tail", "-f", "/dev/null"]
```

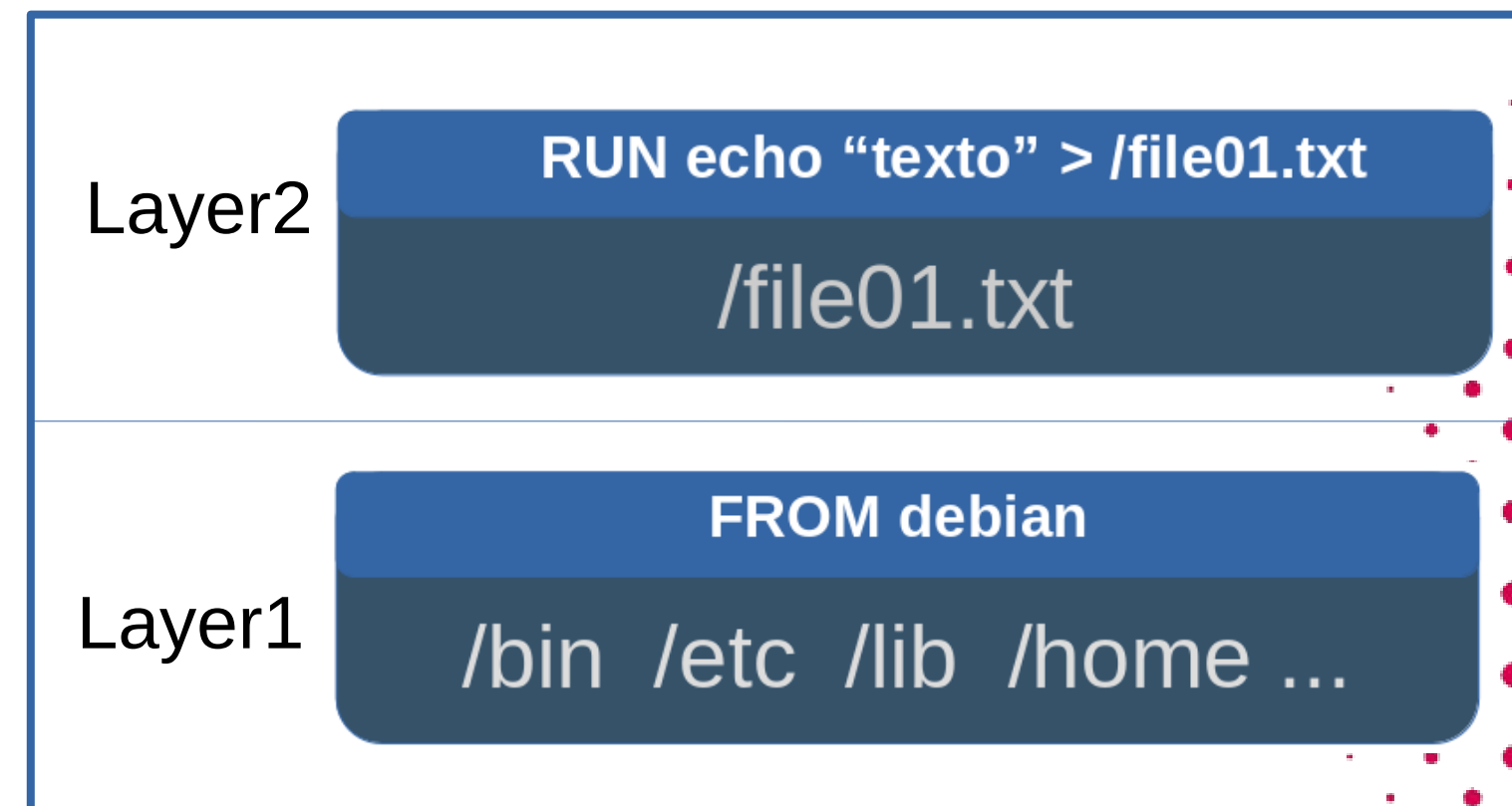
Docker build

....

```
Step 1/3 : FROM debian:latest
--> c93961f83a1e
Step 2/3 : RUN echo "Texto" > /file01.txt
--> Running in 449e5043eba7
--> Removed intermediate container 449e5043eba7
--> c565509506de
Step 3/3 : CMD ["tail", "-f", "/dev/null"]
--> Running in fdaff621c92c
--> Removed intermediate container fdaff621c92c
--> 3d7a820d52c6
Successfully built 3d7a820d52c6
Successfully tagged union-simples:latest
```



Image Docker



Cada Layer é um diretório
em:
`/var/lib/docker/overlay2/`

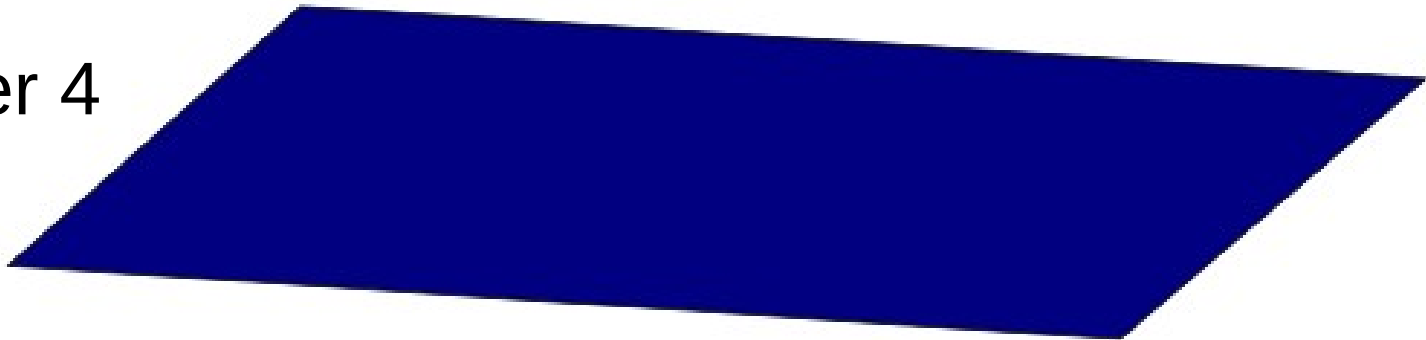
A estrutura base das imagens são diretórios que representam as camadas

Camadas do UnionFS (driver overlay2)

Container



Layer 4



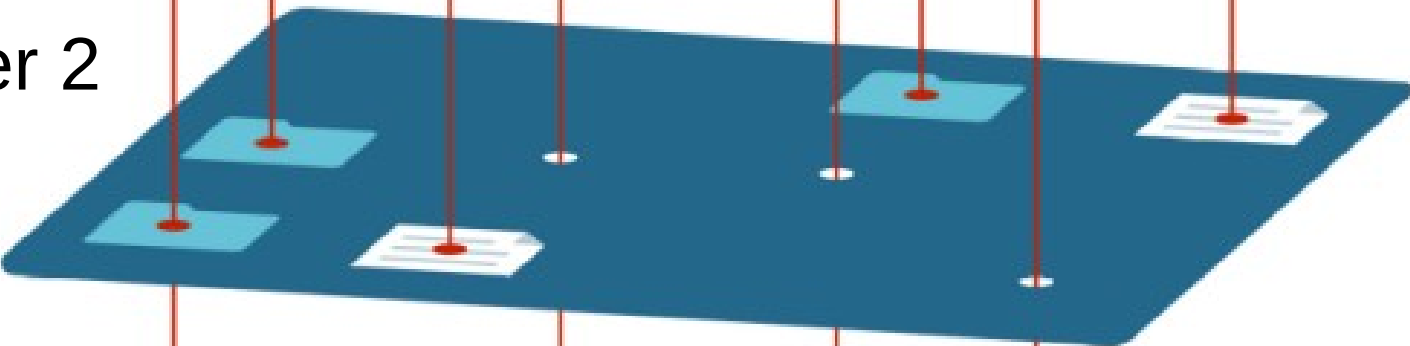
UpperDir

Layer 3



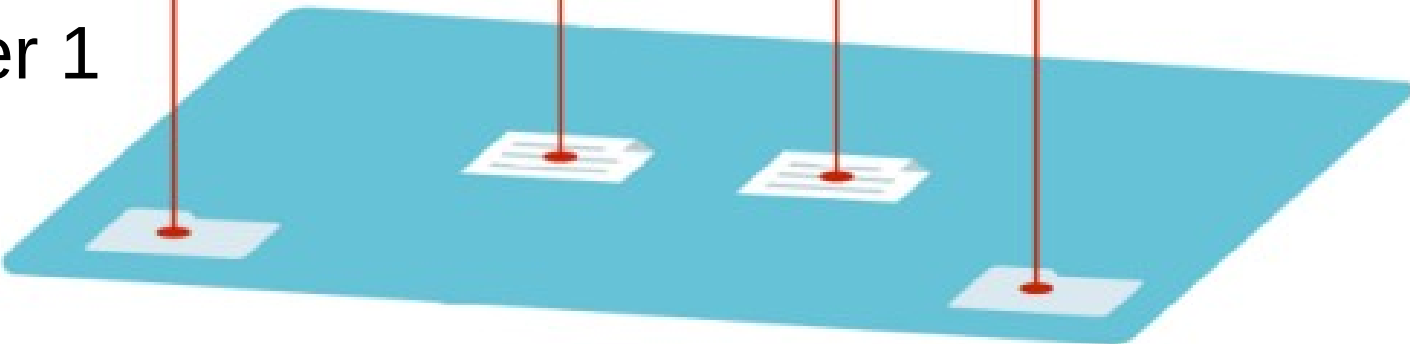
MergedDir

Layer 2



LowerDir

Layer 1



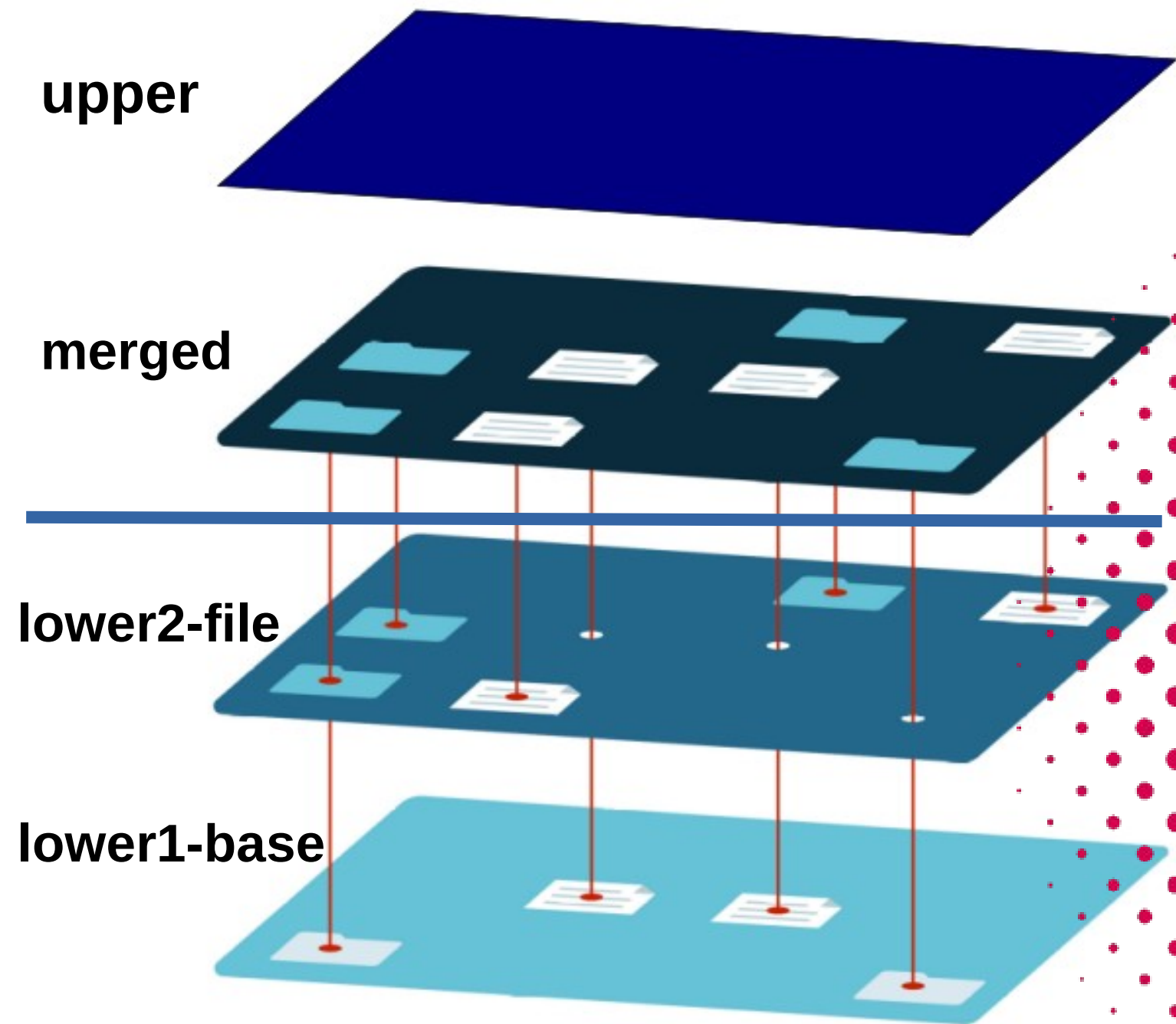
Image

Criando as camadas do UnionFS

```
-----  
== Diretorios do unionFS  
mkdir -p /overlay/{lower1-base,lower2-file,upper,work,merged}  
  
lower1-base lower2-file merged upper work
```

```
-----  
== Listando as camadas  
ls --color /overlay/{lower1-base,upper,work,merged}  
  
/overlay/lower1-base:  
bin dev home lib64 mnt proc run srv tmp var  
boot etc lib media opt root sbin sys usr  
  
/overlay/lower2-file:  
file01.txt  
  
/overlay/merged:  
  
/overlay/upper:  
  
/overlay/work:  
work
```

Observe que file01.txt
não está em lower1-base



UnionFS (driver overlay2)

```
-----  
== Montando o sistema de arquivos Overlay  
mount -t overlay overlay -o\  
lowerdir=/overlay/lower2-file:/overlay/lower1-base,\  
upperdir=/overlay/upper,\  
workdir=/overlay/work\ /overlay/merged
```

```
root@meu-pc:~#  
df -h | grep overlay  
overlay          77G   20G   53G   28% /overlay/merged
```

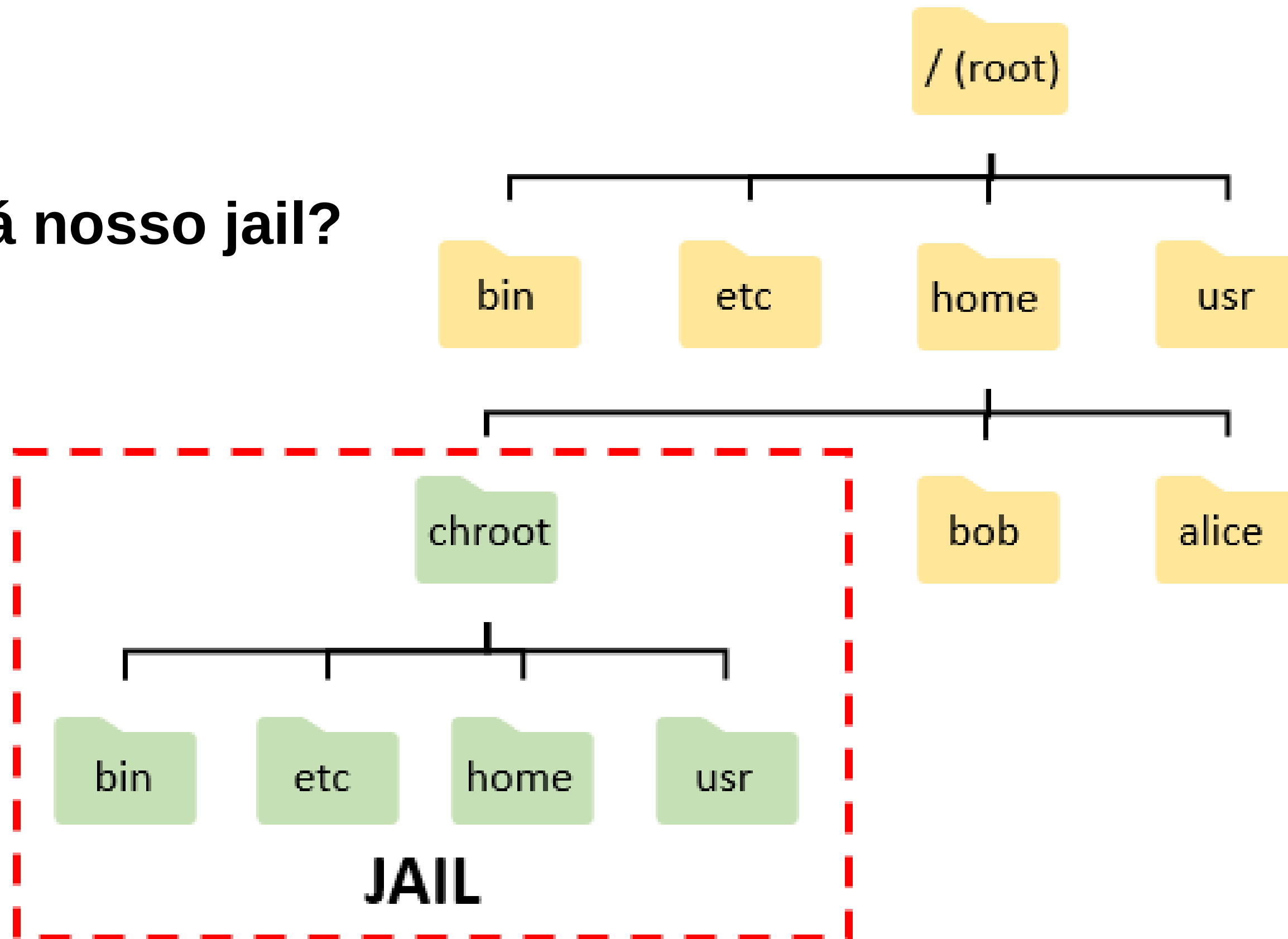
```
root@meu-pc:~#  
ls /overlay/merged/  
bin  dev  file01.txt  lib    media  opt    root  sbin  sys  usr  
boot etc  home      lib64 mnt    proc  run   srv   tmp  var
```

← O arquivo file01.txt na camada merged

chroot (jail)

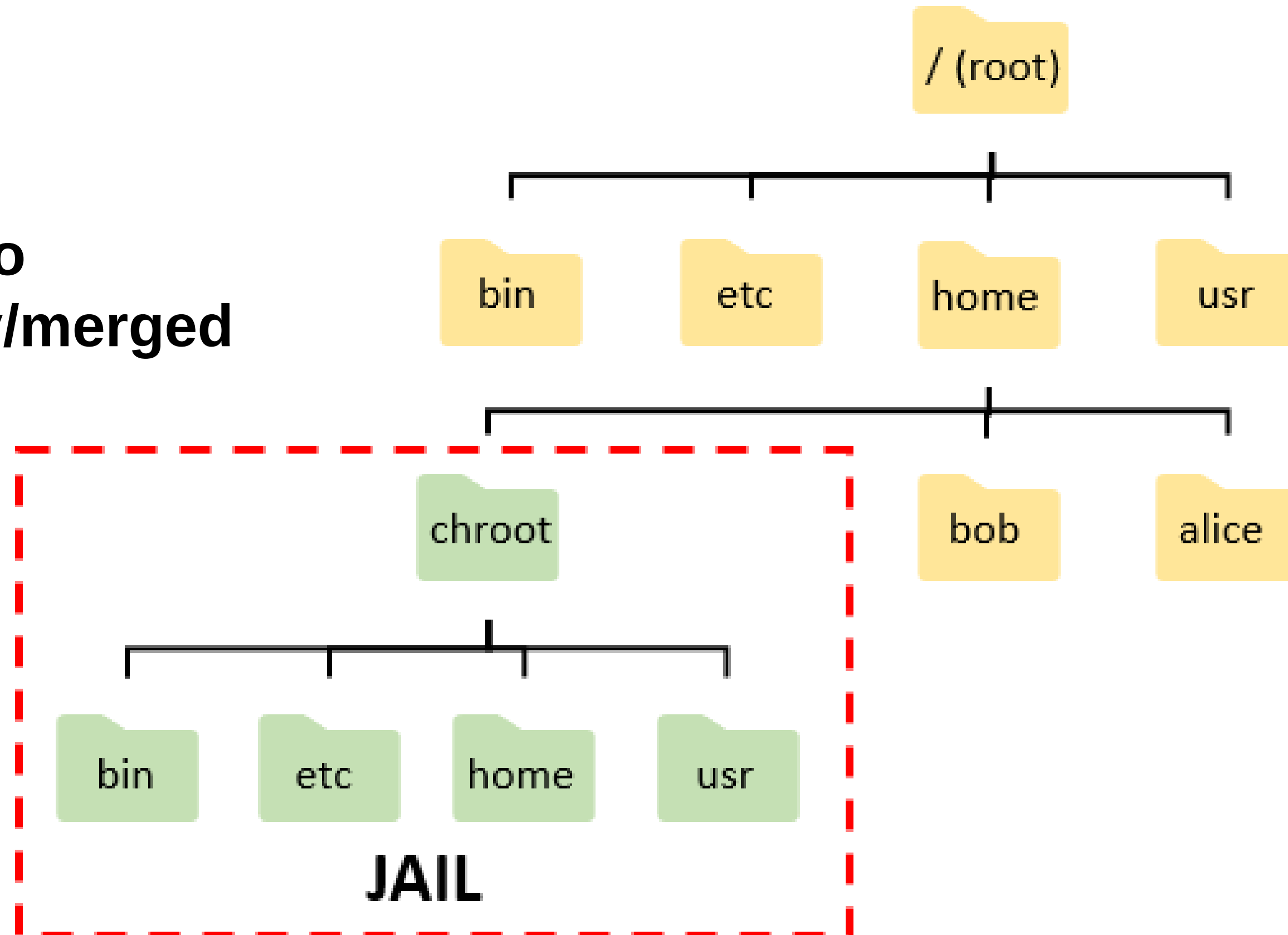
Redireciona o ponteiro do diretório raiz (/) isolando o sistema de arquivos

Quem será nosso jail?



chroot (jail)

Diretório
/overlay/merged



chroot (jail)

```
root@meu-pc:/#  
ls /  
bin      debian-day  home      lib64      media     overlay    run      sys      var  
boot     dev         lib       libx32     mnt       proc       sbin     tmp  
dados    etc         lib32     lost+found opt        root      srv      usr  
  
root@meu-pc:/#  
ls /overlay/merged/  
bin      dev  file01.txt  lib      media  opt  root  sbin  sys  usr  
boot     etc  home      lib64    mnt    proc  run   srv   tmp  var  
  
root@meu-pc:/#  
chroot /overlay/merged/
```

Diretório /overlay/merged/ dentro do jail

```
root@meu-pc:/# ls /  
bin      dev  file01.txt  lib      media  opt  root  sbin  sys  usr  
boot     etc  home      lib64    mnt    proc  run   srv   tmp  var
```

chroot (jail)

Não proporciona isolamento de processos, rede, user...

```
root@meu-pc:/# mount -t proc proc /proc/
root@meu-pc:/# ps ax
```

PID	TTY	STAT	TIME	COMMAND
1	?	Ss	0:03	/sbin/init
2	?	S	0:00	[kthreadd]
3	?	S	0:00	[pool_workq
4	?	I<	0:00	[kworker/R-
5	?	I<	0:00	[kworker/R-
6	?	I<	0:00	[kworker/R-
7	?	I<	0:00	[kworker/R-
9	?	I<	0:00	[kworker/0:
12	?	I<	0:00	[kworker/R-
13	?	I	0:00	[rcu_tasks_
14	?	I	0:00	[rcu_tasks_
15	?	I	0:00	[rcu_tasks_
16	?	S	0:10	[ksoftirqd/
17	?	I	0:34	[rcu_preempt
18	?	S	0:00	[rcu_exp_pa
19	?	S	0:00	[rcu_exp_gp

```
root@meu-pc:/# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp2s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fqdisc
    state DOWN group default qlen 1000
    link/ether 20:04:0f:fd:85:f2 brd ff:ff:ff:ff:ff:ff
3: wlp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
    state UP group default qlen 1000
    link/ether 28:3a:4d:92:d4:29 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.134/24 brd 10.0.0.255 scope global dynamic nopredictable-wl
        te wlp3s0
        valid_lft 21504sec preferred_lft 21504sec
    inet6 2804:2b5c:e08a:7000:12c7:4e0d:da8a:1bc/64 scope global dynamic
        rary dynamic
```


Namespaces (unshare)

São uma feature do kernel Linux que isolam vistas de recursos do sistema para um conjunto de processos

Tipos

- **pid** — espaço de IDs de processo (processos em ns diferentes podem ter o mesmo PID).
- **net** — pilha de rede (interfaces, roteamento, iptables).
- **uts** — hostname e domainname.
- **ipc** — System V IPC e POSIX message queues.
- **user** — mapeamento UID/GID (permite que um processo seja root *dentro* do namespace mas não fora).
- **cgroup** — visibilidade do namespace de cgroups.

Namespaces (unshare)

```
root@meu-pc:/#  
ls /  
bin      dev      lib32    media    proc     srv      var  
boot     etc      lib64    mnt      root     sys  
dados    home    libx32   opt      run      tmp  
debian-day lib    lost+found overlay  sbin     usr  
  
root@meu-pc:/#  
unshare --fork --pid --net --mount-proc=/overlay/merged/proc  
chroot /overlay/merged/
```

← Criando
isolamento de
processo e rede

→

```
root@meu-pc:/# ls --color /  
bin      dev      file01.txt  lib      media    opt      root     sbin     sys     usr  
boot     etc      home        lib64    mnt      proc     run      srv      tmp     var  
root@meu-pc:/# ps au  
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND  
root           1  0.0  0.0   4332   3536 ?        SN    00:33    0:00 /bin/bash -i  
root           6  0.0  0.0   6396   3648 ?        RN+   00:34    0:00 ps au  
root@meu-pc:/# ip a  
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

Namespaces (unshare)

```
FROM debian:latest  
RUN echo "Texto" > /file01.txt  
CMD ["tail", "-f", "/dev/null"]
```

```
root@meu-pc:/#  
unshare --fork --pid --net --mount-proc=/overlay/merged/proc  
chroot /overlay/merged/ tail -f /dev/null
```




E ahe, curtiu nossa jornada?

Dúvidas?





**OBRIGADO! TENHA
UM ÓTIMO DEBIAN
DAY!**

