Contenedores y microservicios

ÍNDICE

OBJETIVOS DEL TALLER

1. Comprender la arquitectura docker.
2. Securizar una instalación por defecto.

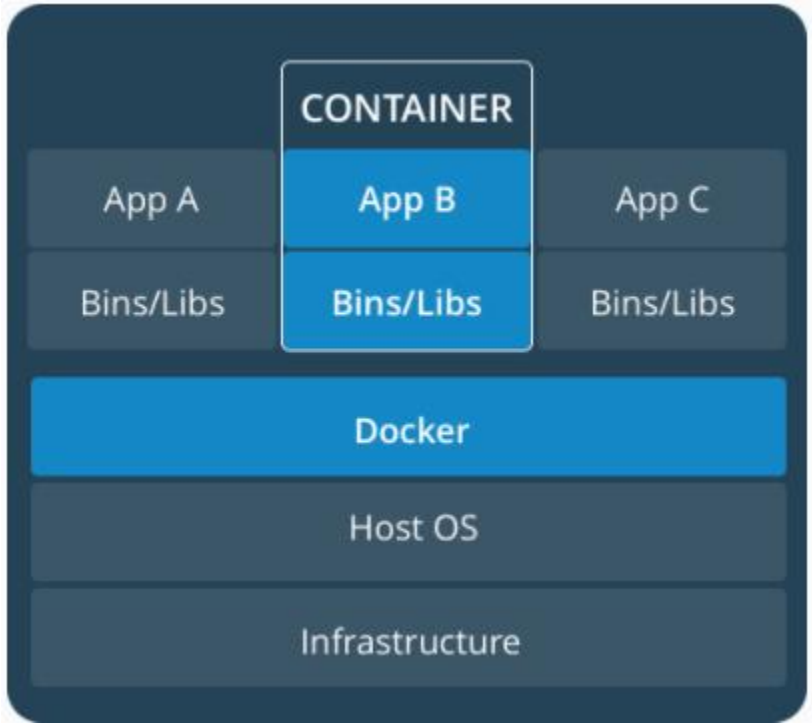
Recursos del taller

https://github.com/SidertiaLabs/contenedoresyseguridad

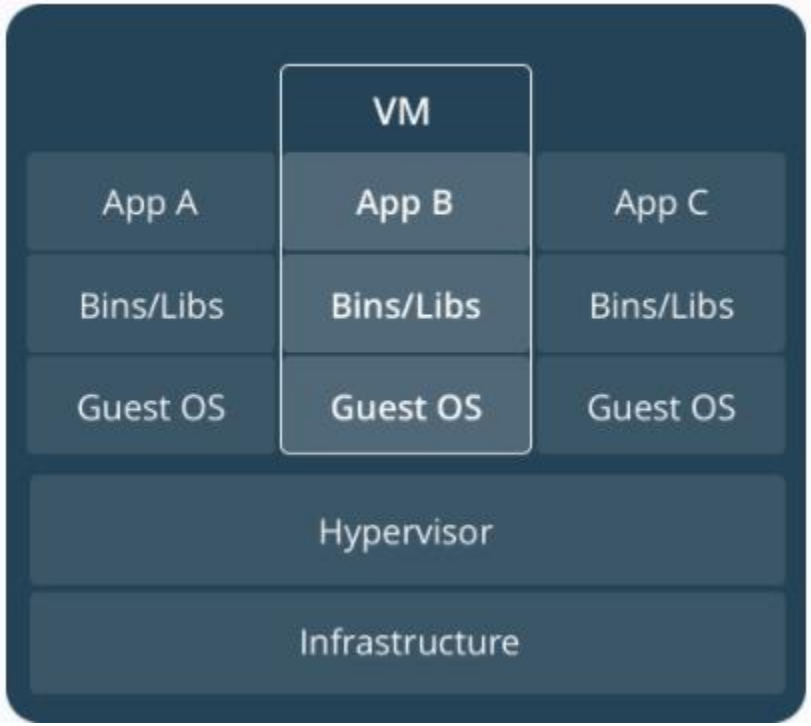
1 Virtualización y contenerización

1 VIRTUALIZACION Y CONTENERIZACIÓN

1.1 Virtualización SO vs Emulación de Hardware



Virtualización SO

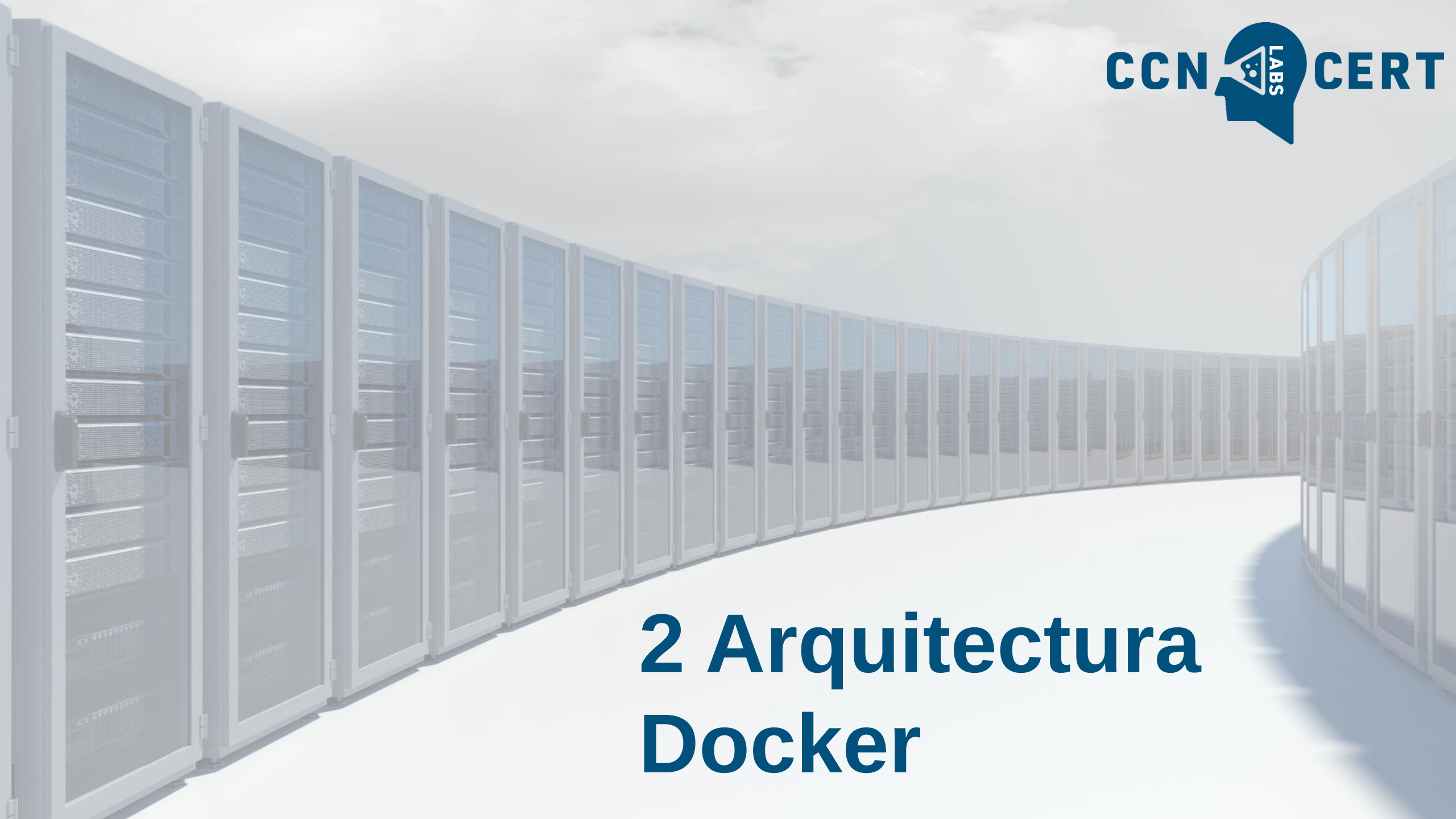


Emulado de Hardware

1 VIRTUALIZACION Y CONTENERIZACIÓN

1.2 Ventajas contenedores vs VM

- Densidad, eficiencia y menor mantenimiento
- Aislamiento y organización
- Escalado semi automático
- Entorno fijo, agiliza el desarrollo. DEVOPS
- Contenedores efímeros, arranque en segundos



2 Arquitectura Docker

2 ARQUITECTURA DOCKER

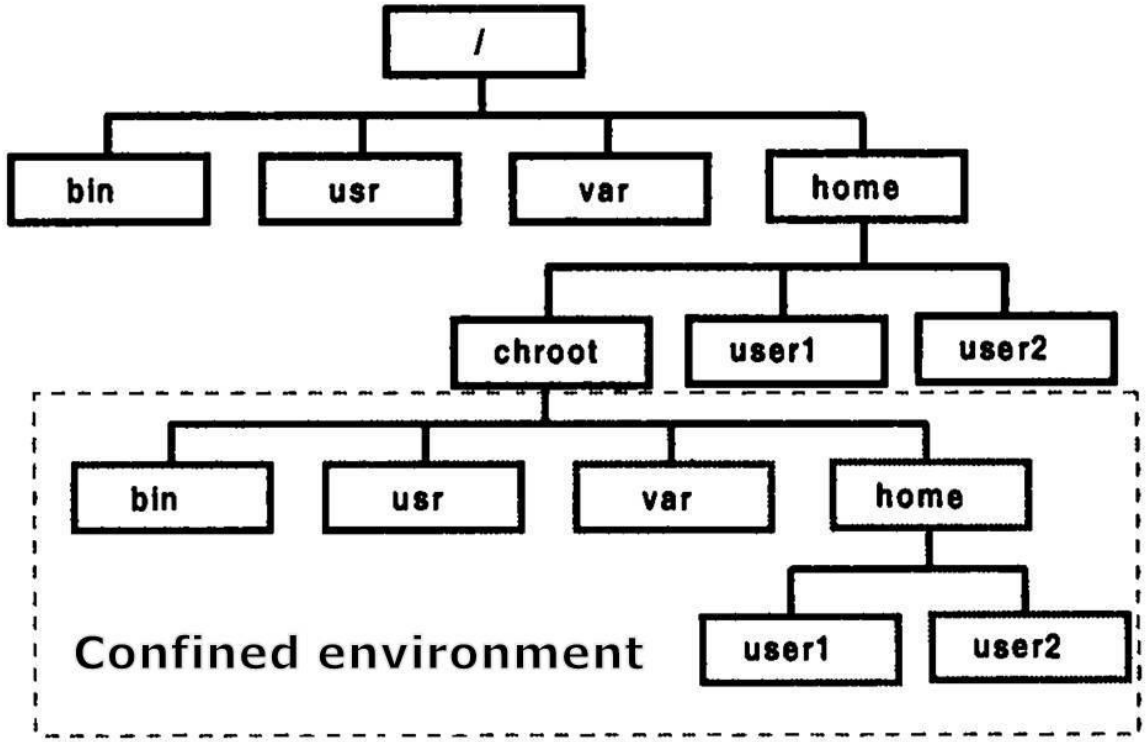
2.1 Contenedor = Proceso con límites

Un contenedor es un proceso aislado.

```
[administrator@localhost targeted]$ ps aux | grep docker
adminis+  5454  0.0  1.7 860716 65928 pts/2    Sl+  05:02   0:00 docker run -ti -v /etc/sltest.txt:/tmp/sltest.txt alpine sh
root      5469  0.0  0.1  10744  5192 ?        Sl   05:02   0:00 containerd-shim -namespace moby -workdir /var/lib/containerd/io.containerd.runtime.v1.linux/m
96ecd19c2a434d335df -address /run/containerd/containerd.sock -containerd-binary /usr/bin/containerd -runtime-root /var/run/docker/runtime-runc
adminis+  6340  0.0  0.0  12112   964 pts/0    S+   09:17   0:00 grep --color=auto docker
root     29196  0.0  2.0 895988 77436 ?        Ssl  04:42   0:05 /usr/bin/dockerd -H fd://
root     29487  0.0  0.1   9336  5428 ?        Sl   04:42   0:00 containerd-shim -namespace moby -workdir /var/lib/containerd/io.containerd.runtime.v1.linux/m
9f1318aca53bb7a9bd9 -address /run/containerd/containerd.sock -containerd-binary /usr/bin/containerd -runtime-root /var/run/docker/runtime-runc
root     29651  0.0  0.1 379324  3872 ?        Sl   04:43   0:00 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 80 -container-ip 172.22.0.3 -cor
root     29657  0.0  0.1   9336  5124 ?        Sl   04:43   0:00 containerd-shim -namespace moby -workdir /var/lib/containerd/io.containerd.runtime.v1.linux/m
7764372815794726a62 -address /run/containerd/containerd.sock -containerd-binary /usr/bin/containerd -runtime-root /var/run/docker/runtime-runc
```

2 ARQUITECTURA DOCKER

2.2 CHROOT, 1982



2. ARQUITECTURA DOCKER

2.3 Namespaces

Aísla acceso de procesos en función del tipo de espacio de nombres.

Mount | UTS | IPC | PID | Network | User | Cgroup

```
[administrator@localhost targeted]$  sudo unshare --fork --pid --mount-proc bash
[root@localhost targeted]# ps aux
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root          1  0.5  0.1  26140  4844 pts/0    S    10:28   0:00 bash
root         18  0.0  0.1  57172  3892 pts/0    R+   10:29   0:00 ps aux
[root@localhost targeted]#
```

Deja de compartir el espacio de nombres de pid

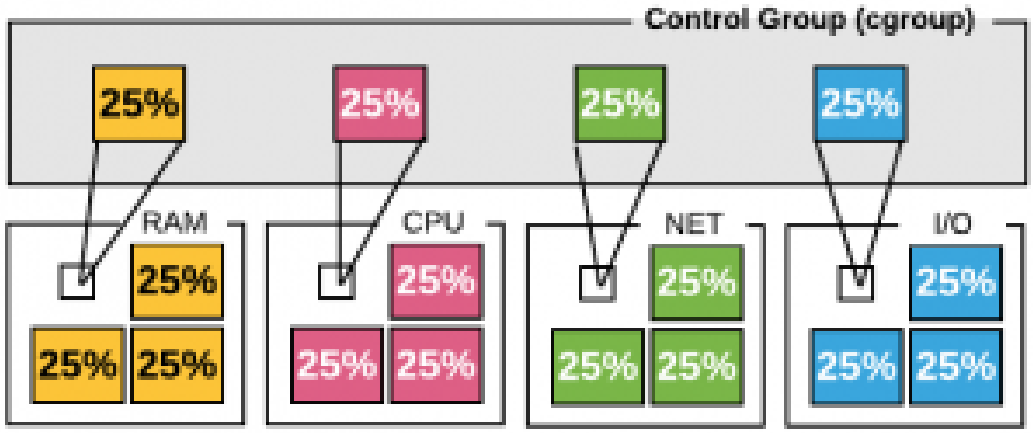
2 ARQUITECTURA DOCKER

2.4 CGroups

Ingenieros de Google en 2006, “process containers”

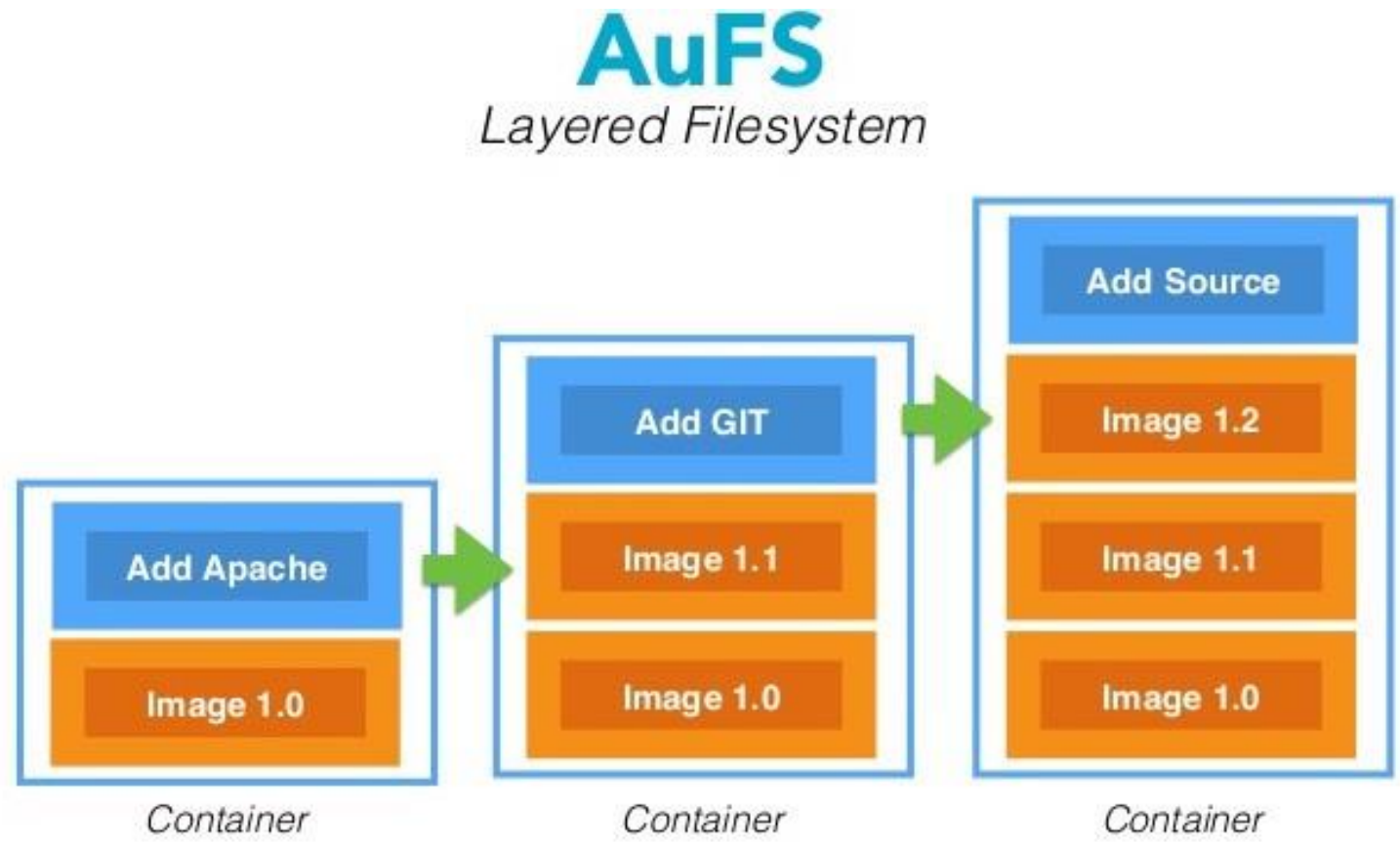
Aísla, limita, prioriza y controla recursos Hardware entre procesos.

CPU, Memoria, Network bandwith e I/O.



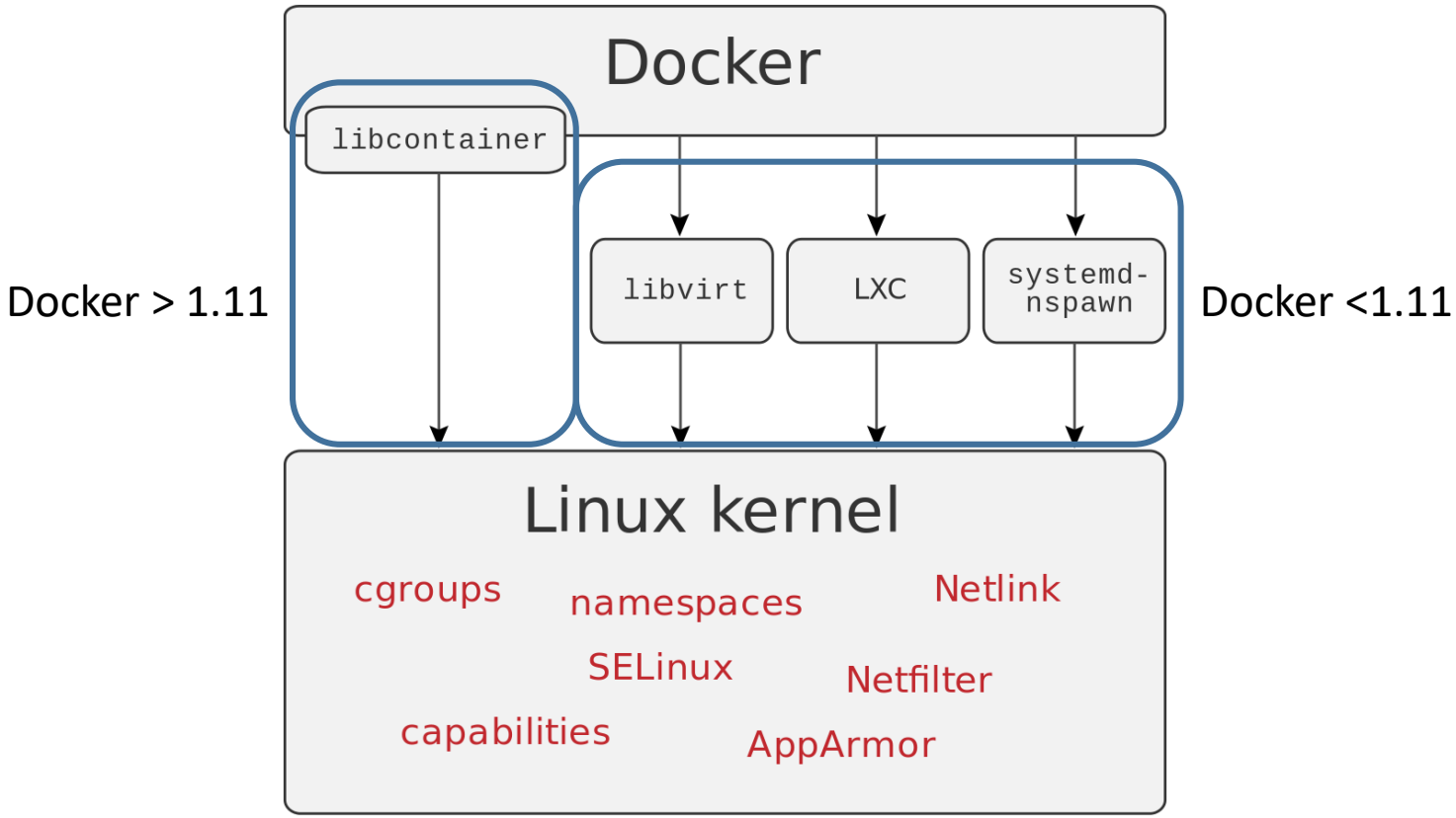
2 ARQUITECTURA DOCKER

2.5 AuFS



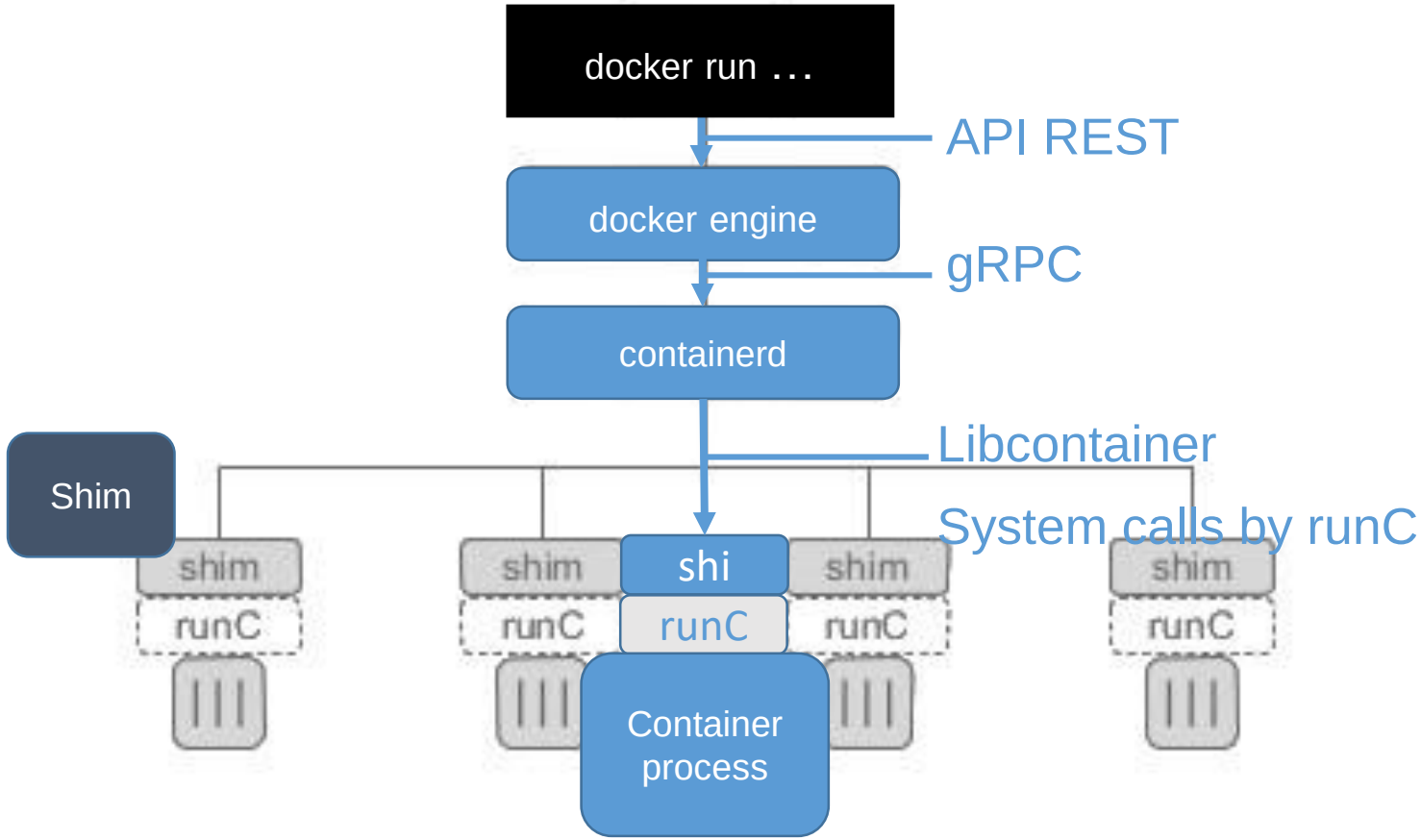
2 ARQUITECTURA DOCKER

2.6 Interacción con el kernel



2 ARQUITECTURA DOCKER

2.7 Arquitectura Docker 1.11.0+



3 Securización

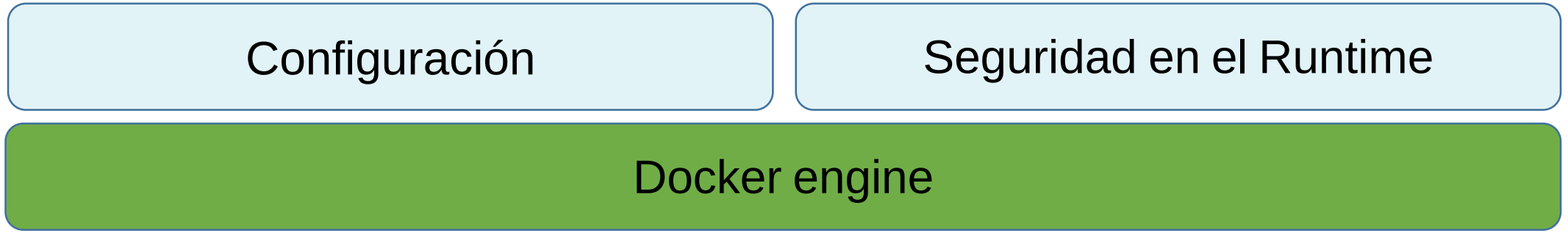
3 SECURIZACIÓN

3.1 Puntos principales a securizar

- Vulnerabilidades en imágenes. TRUST Registry
- Host
- Docker engine
- Auditoría de comportamiento

3 SECURIZACIÓN

3.2 Securización de Docker Engine



3 SECURIZACIÓN

3.3 Herramientas

Herramienta	Función	Origen
Docker-bench-Security	Security conf best practices	Open source, free
Docker-Bench-Test	Security conf best practices	Open source, free
Chefinspect CIS profile	Security conf best practices	Open source, free
Dockscan	Docker instalation vulnerabilities	Open source, free
Anchore engine CLI	Image vulnerabilities	Cli open source, free Front UI enterprise, paid
Falco sysdig	Container IDS	Open source core, free. Paid enterprise functions

3 SECURIZACIÓN

3.3 Herramientas

Docker-Bench-Security

- Herramienta propuesta por la compañía.
- Basada en CIS Docker benchmark v1.2.0

```
[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1  - Ensure the container host has been Hardened
[INFO] 1.1.2  - Ensure Docker is up to date
[INFO]        * Using 18.09.1, verify is it up to date as deemed necessary
[INFO]        * Your operating system vendor may provide support and security maintenance fo

[INFO] 1.2 - Linux Hosts Specific Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2  - Ensure only trusted users are allowed to control Docker daemon
[INFO]        * docker:x:986:administrator
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
```

3 SECURIZACIÓN

3.4 Configuración

Elementos de configuración de dockerd:

- User namespace mapping.
- Live restore
- no_new_priv bit
- Exposición del socket
- docker run –privileged
- Bind a interfaz del host única
- Exposición del socket

3 SECURIZACIÓN

3.5 Runtime Security

SeLinux

- Security-enhance
- Está basado en c
- Cada contexto se
- Responde a la pr
- Type enforcemer
- MCS (Multi Categ

eto>?

Seccomp

- Permite o denieg

4 Taller Práctico

4 TALLER PRÁCTICO

4.1 Configuración y runtime seguros

4.2 Buenas prácticas

4 TALLER PRÁCTICO

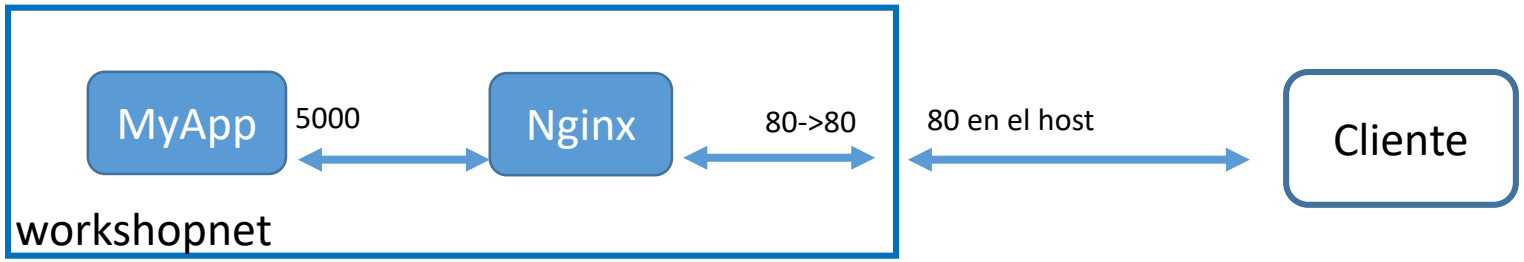
4.1.1 Entorno

- docker engine v 18.09.1 en CentOS 8
- Configuración por defecto
- Plantilla app dotnet core
- Contenedor Nginx

```
docker ps

docker network ls

docker network inspect workshopnet
```



4 TALLER PRÁCTICO

4.1.2 Docker bench security

Ejecutar herramienta de cumplimiento de configuración

```
sudo ./docker-bench-security
```

4 TALLER PRÁCTICO

4.1.3 Mapeado de nombres

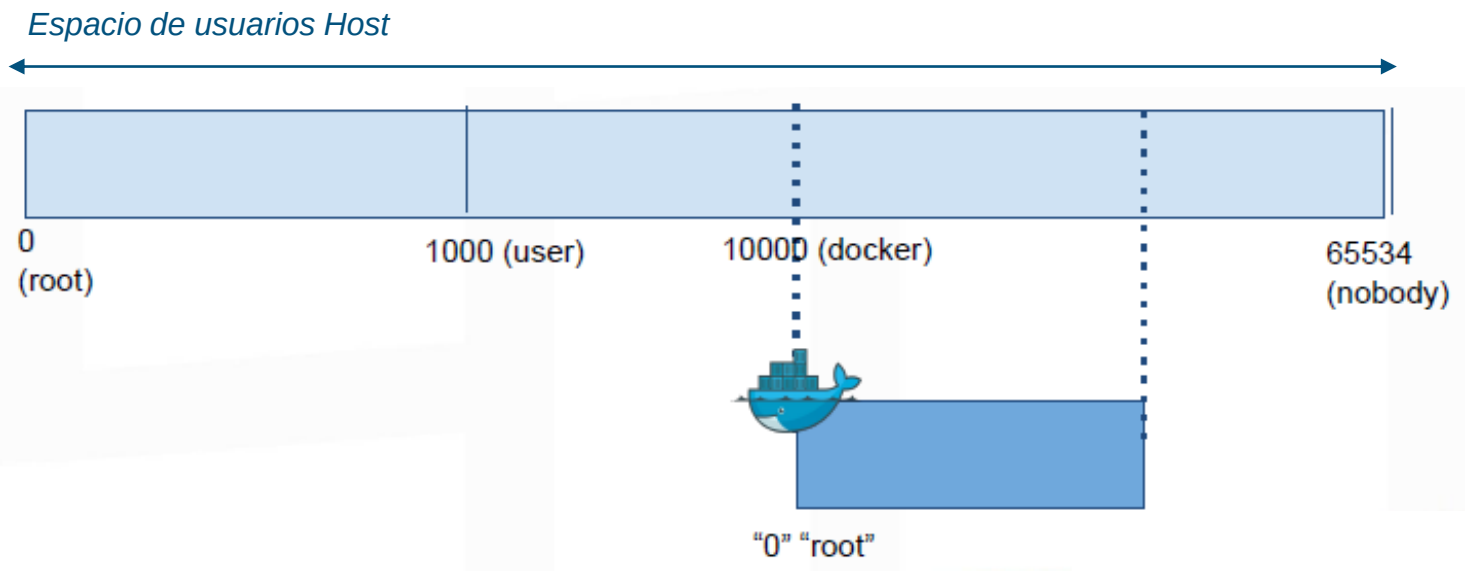
Control 2.8 Mapeado de espacio de nombres de usuario

Problema: root en contenedor == root en host

Soluciones:

- Etiqueta USER en dockerfile o ejecutar con flag run –user. El problema es que no permite ser root en el contenedor.
- Mejor solución: mapear non-root user a root user en el contenedor.
- Si un atacante consigue acceso a un contenedor, solo será root en ese contexto.
- En el contexto de host es un usuario mapeado sin privilegios.

4 TALLER PRÁCTICO



Las imágenes y contenedores previos ya no existen en este espacio de nombres.

Necesario tener en cuenta la política de SeLinux si se utilizan ambos.

```
Editar daemon.json

usersns-remap=default

Restart Daemon
```

4 TALLER PRÁCTICO

4.1.4 Disponibilidad

Control 2.13 Alta disponibilidad de contenedores

Permite restaurar Docker daemon sin afectar a los contenedores (shim).

```
Propiedad live-restore

Paramos el servicio Docker

Vemos los procesos shim con ps aux | grep Docker
```

4 TALLER PRÁCTICO

4.1.5 Aislamiento

Control 2.17 Persistencia de privilegios en procesos hijo

Mantiene privilegios en las operaciones fork, clone y execve (al crear nuevos contenedores)

`Propiedad no-new-privileges`

4 TALLER PRÁCTICO

4.1.5 Aislamiento

Control 5.28 Límite de procesos

Especifica un límite de uso de recursos (ficheros, procesos, etc)

Previene ataques de denegación de servicio que utilicen fork como método principal. (forkbombs)

```
Propiedad –default-ulimits                    Límite nproc por usuario!!
A nivel contenedor con –pids-limit

Ejecutar forkbomb en contenedor
```

4 TALLER PRÁCTICO

4.1.6 Runtime security

Control 5.2 Selinux

Necesaria política específica para contenedores. Paquete container-selinux, manual o UDICA.

```
Paquete de políticas específicas de Docker: container-selinux-*.rpm

Propiedad selinux-enable=true
```

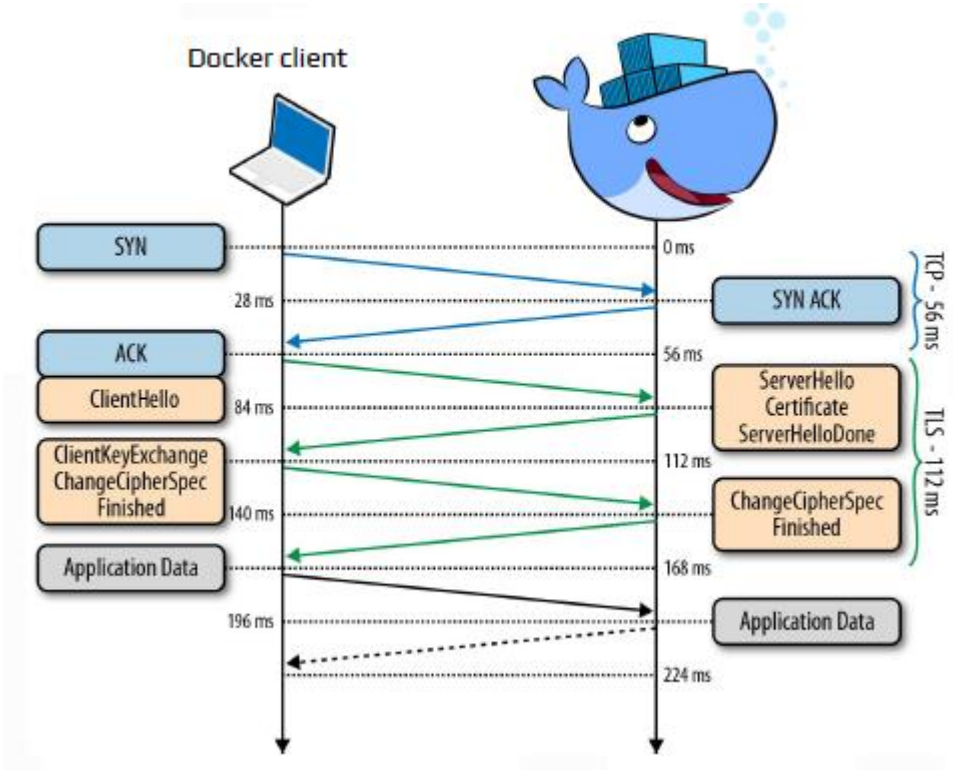
4 TALLER PRÁCTICO

4.2 Buenas Prácticas

No exponer el socket tcp

Exponer a través de vpn u otros mecanismos.

Si es imprescindible usar TLS.



4 TALLER PRÁCTICO

4.2 Buenas Prácticas

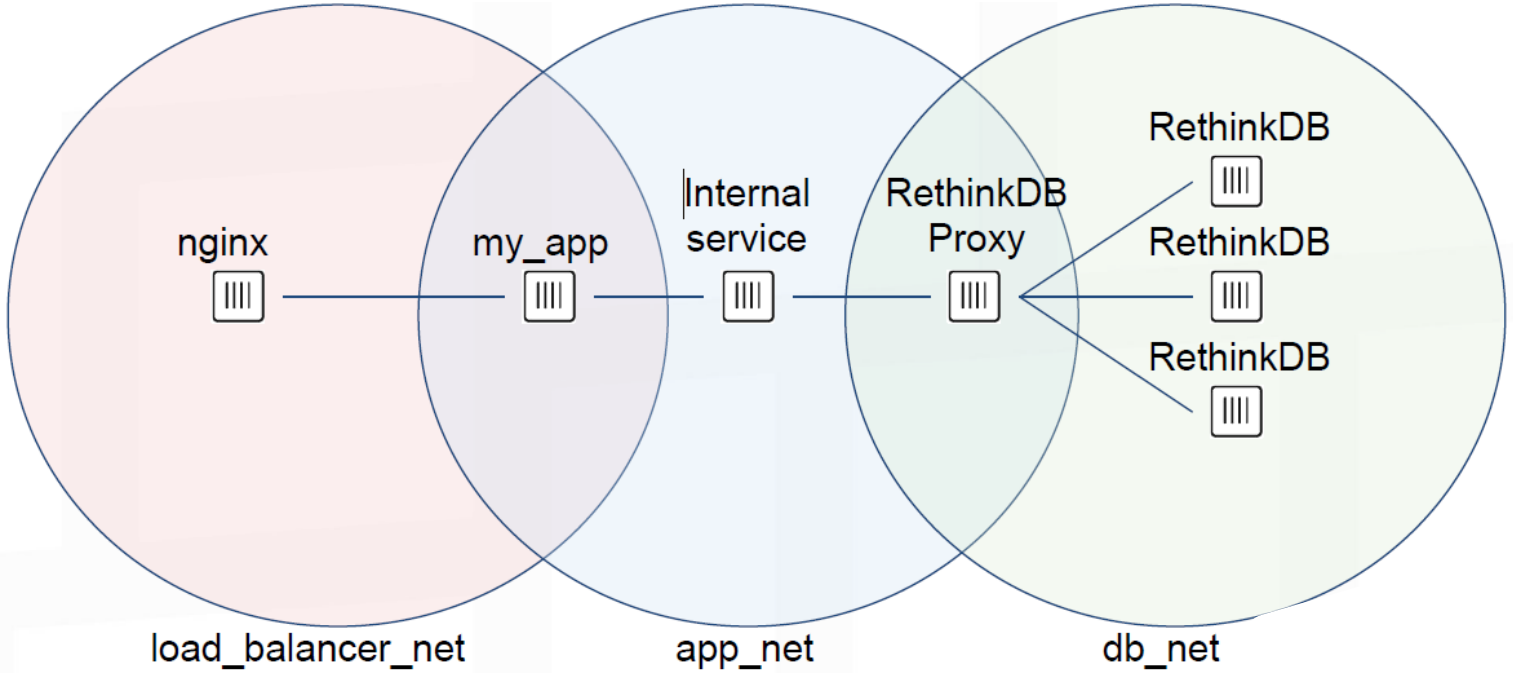
Puertos 2375 expuestos

4 TALLER PRÁCTICO

4.2 Buenas Prácticas

Control 2.1 Restringir red bridge entre contenedores. Crear redes independientes.

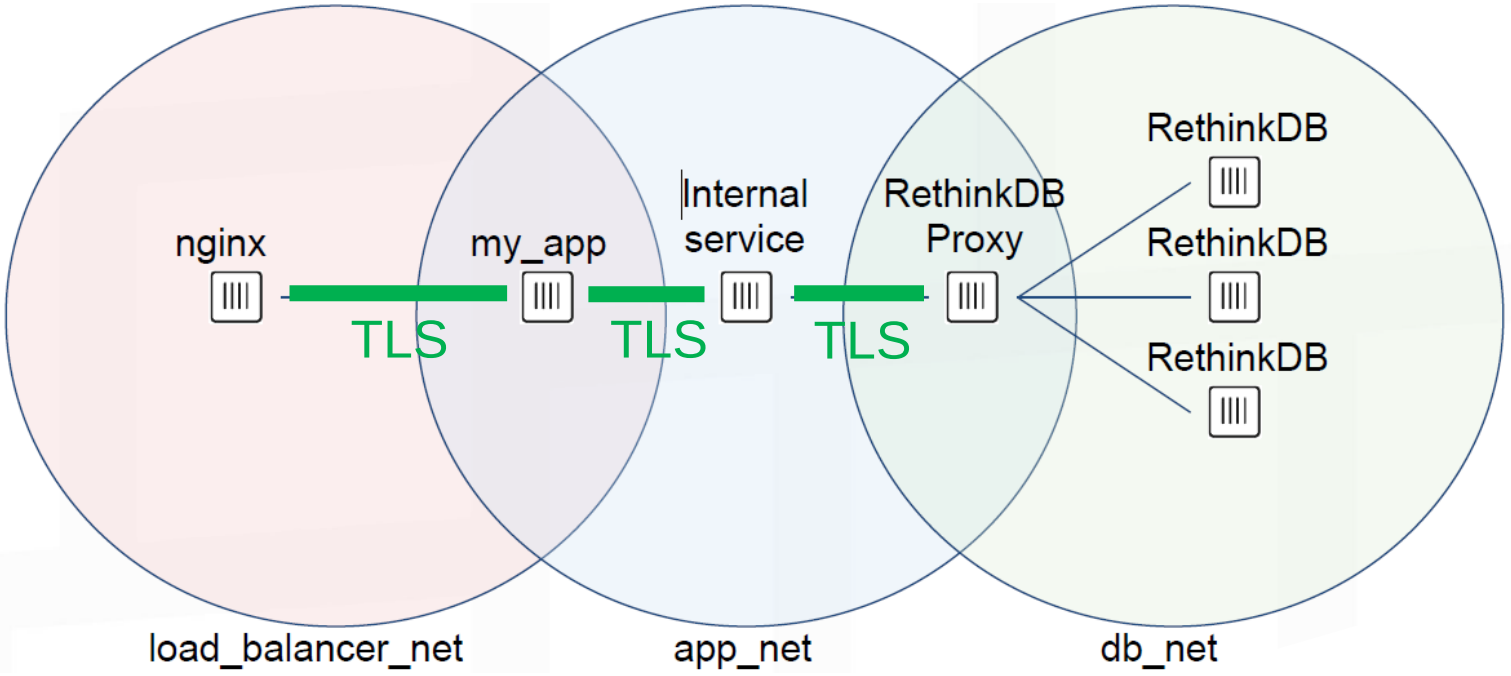
```
Set property enable_icc:false
```



4 TALLER PRÁCTICO

4.2 Buenas Prácticas

Comunicación TLS entre pares



4 TALLER PRÁCTICO

4.2 Buenas Prácticas

Ejecutar contenedores con permisos de lectura.

```
docker run it --rm --read-only alpine sh
```

Volúmenes de solo lectura

```
-v /volumeHost:/volumeContenedor:ro
```

Muchas gracias

WWW.CCN-CERT.CNI.ES

WWW.CCN.CNI.ES

OC.CCN.CNI.ES