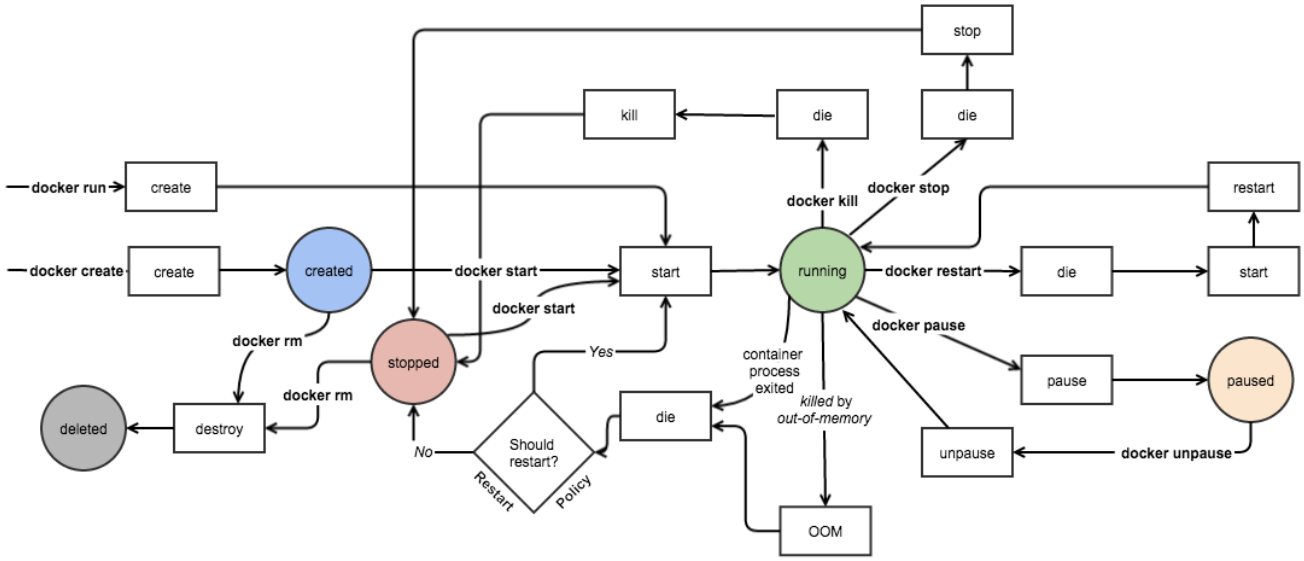
[Docker SecDevOps] Capítulo 4 : Contenedores

- 2 premisas: inmutabilidad y un sólo proceso (aunque se pueden modificar)
  - inmutabilidad: proceso de usar y tirar
  - sólo el PID 1 recibe las señales del sistema → dedicado y especializado en hacer una sola tarea, un servicio como objetivo único
- `docker container`
  - attach
  - commit
  - cp
  - create
  - diff
  - exec
  - export
  - inspect
  - kill
  - logs
  - ls
  - pause
  - port
  - prune
  - rename
  - restart
  - rm
  - run
  - start
  - stats
  - stop
  - top
  - unpause
  - update
  - wait

ciclo de vida de un contenedor



estados:

- created : creado pero no ejecutado
- restarting : reiniciando -
- running : en ejecución
- removing : borrándose
- paused : suspendido
- exited : el proceso principal del contenedor ha sido parado. Puede ser reiniciado
- dead : está siendo borrado, pero no ha liberado todos los recursos.

comandos:

- `docker container create`
- `docker container run` : create + start
- `docker container start`
- `docker container pause`
- `docker container unpause`
- `docker container stop` : envía señal SIGTERM
- `docker container restart`
- `docker container kill` : envía señal SIGKILL
- con **create** y **run** se puede usar el parámetro `–restart` que establece la política de reinicio del contenedor:
  - `no` : por defecto
  - `on-failure` : código de salida diferente de 0
  - `unless-stopped`
  - `always`

listar contenedores

- `docker container ls`
- `docker container ls -a`
- filtrando:
  - `docker container ls -f [–filter] «status=created»`
  - permite varios filtros en la misma instrucción
  - id, name, ancestor, network, label
  - status == created, restarting, running, removing, paused, exited, dead

- formateando la salida:
  - `–format «`
    id
    -
    names
    »
    - .ID, .Image , .Command, .CreatedAt, .Running, .Ports, .Status, .Size, .Names, .Labels, .Label, .Mounts, .Networks
- por defecto, `docker container ls` añade `–filter «status=running»` mientras no se pase otro filtro.

configuración de un contenedor

- `–name` : asignar nombre a contenedor
  - `docker container rename`
- `-h` o `–hostname` : asigna al hostname de la máquina el valor para mostrar en el prompt a que máquina estamos conectados
- `–dns` : asigna servidor DNS → **/etc/resolv.conf**
- `–add-host=<nombre>:<ip>`
- `-e` o `–env`
- `–env-file` : fichero tipo properties con la relación de variables de entorno
  - para pasar contraseñas y similares mejor usar **secrets**

publicación de puertos

- `-p` o `--publish`
  - <puerto_host>:<puerto_contenedor>
  - <rango_puertos_host>:<rango_puertos_contenedor>
- `-P` o `--publish-all` : publica los puertos **EXPOSE** en puertos libres del host (puertos altos)
  - se puede ver los puertos mapeados de un contenedor con `docker container port <id>`

restricción de recursos

docker se basa en la funcionalidad de linux **cgroups** para la limitación de recursos (RAM y CPU)

- en sistemas RedHat viene habilitado por defecto
- en sistemas Debian igual se tiene que habilitar (en **/etc/default/grub** se deberá añadir **GRUB_CMDLINE_LINUX=«cgroup_enable=memory swapaccount=1»**, ejecutar **sudo update-grub** y reiniciar)
- se pueden alterar las restricciones de un contenedor con **docker container update**

memoria

- `-m` o `--memory`
  - **b**ytes, **k**ilobytes, **m**egabytes, **g**igabytes
  - el valor mínimo permitido son **4 megabytes**
- `--memory-swap` : ha de ser igual o superior a lo indicado en **--memory**
- `--oom-kill-disable` : por defecto, si el contenedor pasa de la memoria asignada, el núcleo matará el proceso del contenedor

CPU

- `--cpus` : indica el uso de CPUs (1.5 indicaría el 100% de uno y el 50% del otro)
  - esto sería equivalente a **--cpu-period=«100000»** y **--cpu-quota=«150000»**
- `--cpuset-cpus` : establece que CPUs o cores puede usar el contenedor (rango o separados por comas)

Información de los contenedores

- `docker container inspect`
  - `--format` : uso necesario para extraer los datos que se buscan entre toda la información
- `docker container logs` : logs del proceso PID=1
  - `-f` : muestra log en vivo
- `docker container top` : procesos del contenedor
- `docker container stats`
  - `--all` : todos los contenedores en marcha

Interactuando con contenedores

- `docker container exec`
- `docker container cp`
  - docker container cp <nombre_contenedor>:<path> <destino_local>
  - docker container cp <origen_local> <nombre_contenedor>:<path>
- `docker container export`
  - docker container export -o fs.tar <contenedor>
- `docker container attach` : enganche al proceso PID=1 del contenedor (STDIN-STDOUT-STDERR)
  - para salir de aquí no hay usar CTRL+C, ya que mataría el proceso, usar en su lugar CTRL+P,CTRL+Q

Persistencia de datos y volúmenes