Docker

Restart Policies

Use a restart policy

To configure the restart policy for a container, use the --restart flag when using the docker run command. The value of the `--restart` flag can be any of the following:

Flag	Description
`no`	Do not automatically restart the container. (the default)
`on-failure`	Restart the container if it exits due to an error, which manifests as a non-zero exit code.
`always`	Always restart the container if it stops. If it is manually stopped, it is restarted only when Docker daemon restarts or the container itself is manually restarted. (See the second bullet listed in restart policy details)
`unless-stopped`	Similar to always, except that when the container is stopped (manually or otherwise), it is not restarted even after Docker daemon restarts.

WatchTower

https://containrrr.github.io/watchtower/

With watchtower you can update the running version of your containerized app simply by pushing a new image to the Docker Hub or your own image registry. Watchtower will pull down your new image, gracefully shut down your existing container and restart it with the same options that were used when it was deployed initially.

```
version: "3"
services:
  watchtower.service:
    container_name: watchtower.service
    image: containrrr/watchtower:latest
    environment:
      - WATCHTOWER_CLEANUP=true
      - WATCHTOWER_SCHEDULE="0 4 * * 2 *"
      - WATCHTOWER_TIMEOUT=30s
    logging:
      options:
        max-size: "200k"
        max-file: "10"
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - /root/.docker/config.json:/config.json
```

Insecure Registries

in /etc/docker/daemon.json add this (don't forget comma after existing lines)

```
"insecure-registries":["192.168.10.110:5000"]
```

Operating Folder

In /etc/docker/daemon.js (don't forget to comma after existing lines)

```
"data-root": "/data/docker"
```

Prune unused objects

Prune images and containers

https://docs.docker.com/config/pruning/

Prune images

The docker image prune command allows you to clean up unused images. By default, docker image prune only cleans up dangling images. A dangling image is one that is not tagged and is not referenced by any container. To remove dangling images:

```
$ docker image prune



WARNING! This will remove all dangling images.
Are you sure you want to continue? [y/N] y
```

To remove all images which are not used by existing containers, use the -a flag:

```
$ docker image prune -a

WARNING! This will remove all images without at least one container associated to them.
Are you sure you want to continue? [y/N] y
```

By default, you are prompted to continue. To bypass the prompt, use the -f or --force flag.

You can limit which images are pruned using filtering expressions with the --filter flag. For example, to only consider images created more than 24 hours ago:

```
$ docker image prune -a --filter "until=24h"
```

Other filtering expressions are available. See the docker image prune reference for more examples.

Prune containers

When you stop a container, it is not automatically removed unless you started it with the --rm flag. To see all containers on the Docker host, including stopped containers, use docker ps -a. You may be surprised how many containers exist, especially on a development system! A stopped container?s writable layers still take up disk space. To clean this up, you can use the docker container prune command.

```
$ docker container prune


WARNING! This will remove all stopped containers.
Are you sure you want to continue? [y/N] y
```

By default, you are prompted to continue. To bypass the prompt, use the `-f` or `--force` flag.

By default, all stopped containers are removed. You can limit the scope using the `--filter` flag. For instance, the following command only removes stopped containers older than 24 hours:

```
$ docker container prune --filter "until=24h"
```

Other filtering expressions are available. See the docker container prune reference for more examples.

Prune volumes

Volumes can be used by one or more containers, and take up space on the Docker host. Volumes are never removed automatically, because to do so could destroy data.

```
$ docker volume prune


WARNING! This will remove all volumes not used by at least one container.
Are you sure you want to continue? [y/N] y
```

By default, you are prompted to continue. To bypass the prompt, use the `-f` or `--force` flag.

By default, all unused volumes are removed. You can limit the scope using the `--filter` flag. For instance, the following command only removes volumes which are not labelled with the keep label:

```
$ docker volume prune --filter "label!=keep"
```

Other filtering expressions are available. See the docker volume prune reference for more examples.

Prune networks

Docker networks don?t take up much disk space, but they do create iptables rules, bridge network devices, and routing table entries. To clean these things up, you can use docker network prune to clean up networks which aren?t used by any containers.

```
$ docker network prune


WARNING! This will remove all networks not used by at least one container.
Are you sure you want to continue? [y/N] y
```

By default, you are prompted to continue. To bypass the prompt, use the `-f` or `--force` flag.

By default, all unused networks are removed. You can limit the scope using the `--filter` flag. For instance, the following command only removes networks older than 24 hours:

```
$ docker network prune --filter "until=24h"
```

Other filtering expressions are available. See the docker network prune reference for more examples.

Troubleshooting

https://bobcares.com/blog/iptables-no-chain-target-match-by-that-name-docker/

```
Our customers often approach us with this error. Firstly, we check if the firewall service
status using

systemctl restart iptables.service

If the service is down we restart the service.

Then, we check the iptables rules using the command

iptables -L

The docker firewall rules were missing thus it shows the error.

To resolve the error our Support Engineers restart the docker service. For instance, to
restart the docker we use the command,

service docker restart

While restarting the Docker, it automatically creates the firewall rules. And we ensure to
enable the firewall before restarting the docker.```
```

aliases

```
alias dcud="docker-compose up -d"

alias dcd="docker-compose down"

alias dcp="docker-compose pull"

alias dclf="docker-compose logs -f"

alias glances="docker run --rm --name=glances -v /var/run/docker.sock:/var/run/docker.sock:ro
--pid host --network host -it nicolargo/glances:latest-full"

alias ctop="docker run --rm -ti --name=ctop --volume
/var/run/docker.sock:/var/run/docker.sock:ro quay.io/vektorlab/ctop:latest"
pull, down, up

pdu() { dcp dcd dcud }
down, up

downup() { dcd dcud }
```

Using Docker for temp app usage

```
docker run -it --rm -v .:/tmp -w /tmp node /usr/local/bin/npm install
```