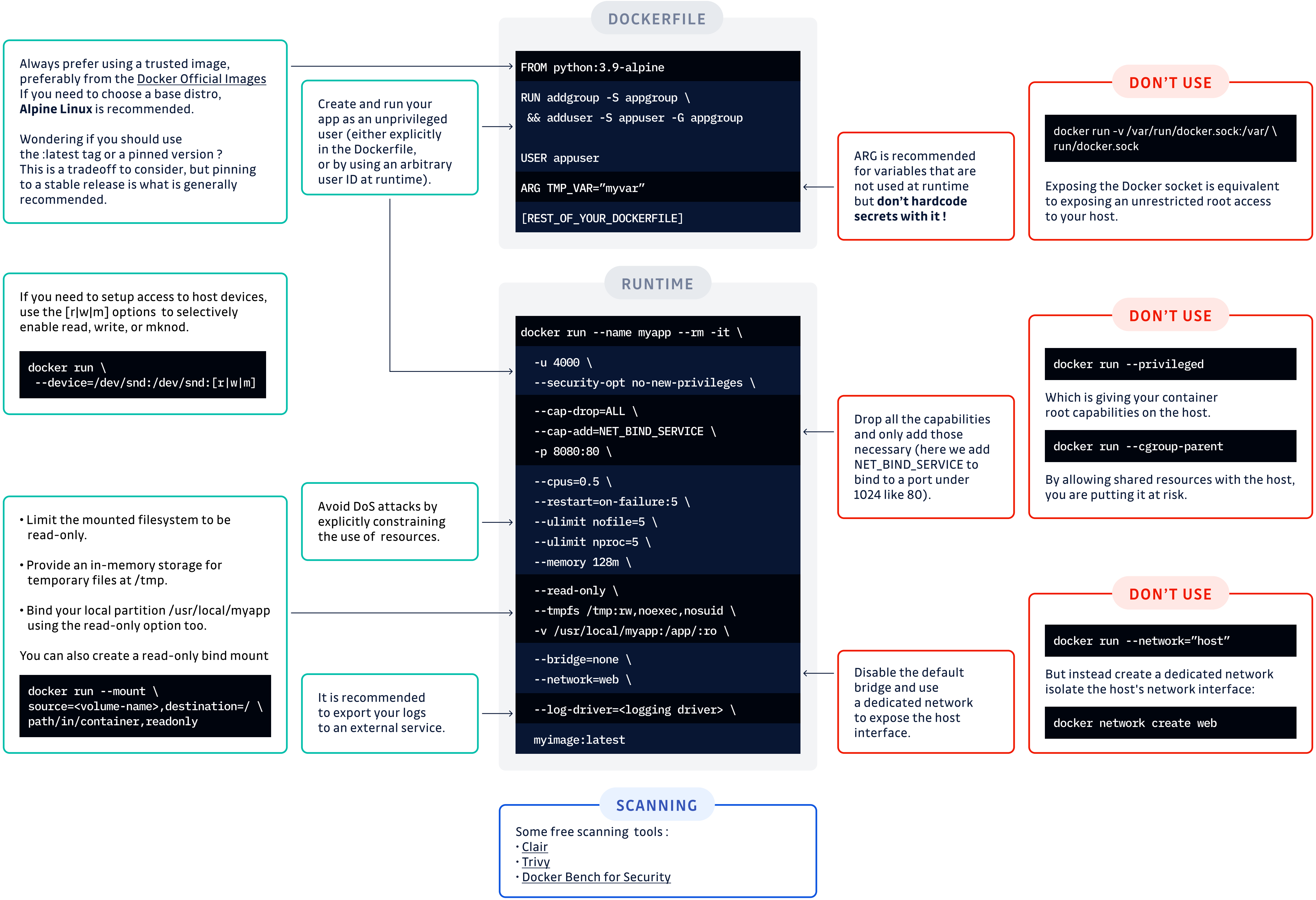
GitGuardian [Docker security cheat sheet — 2021]



DOCKERFILE

Always prefer using a trusted image, preferably from the Docker Official Images
If you need to choose a base distro, **Alpine Linux** is recommended.

Wondering if you should use the :latest tag or a pinned version ?
This is a tradeoff to consider, but pinning to a stable release is what is generally recommended.

Create and run your app as an unprivileged user (either explicitly in the Dockerfile, or by using an arbitrary user ID at runtime).

```
FROM python:3.9-alpine

RUN addgroup -S appgroup \
 && adduser -S appuser -G appgroup

USER appuser

ARG TMP_VAR=”myvar”

[REST_OF_YOUR_DOCKERFILE]
```

ARG is recommended for variables that are not used at runtime but **don’t hardcode secrets with it !**

DON’T USE

```
docker run -v /var/run/docker.sock:/var/ \
run/docker.sock
```

Exposing the Docker socket is equivalent to exposing an unrestricted root access to your host.

RUNTIME

If you need to setup access to host devices, use the [r|w|m] options to selectively enable read, write, or mknod.

```
docker run \
 --device=/dev/snd:/dev/snd:[r|w|m]
```

```
docker run --name myapp --rm -it \
  -u 4000 \
  --security-opt no-new-privileges \
  --cap-drop=ALL \
  --cap-add=NET_BIND_SERVICE \
  -p 8080:80 \
  --cpus=0.5 \
  --restart=on-failure:5 \
  --ulimit nofile=5 \
  --ulimit nproc=5 \
  --memory 128m \
  --read-only \
  --tmpfs /tmp:rw,noexec,nosuid \
  -v /usr/local/myapp:/app/:ro \
  --bridge=none \
  --network=web \
  --log-driver=<logging driver> \
  myimage:latest
```

Drop all the capabilities and only add those necessary (here we add NET_BIND_SERVICE to bind to a port under 1024 like 80).

Avoid DoS attacks by explicitly constraining the use of resources.

- Limit the mounted filesystem to be read-only.
- Provide an in-memory storage for temporary files at /tmp.
- Bind your local partition /usr/local/myapp using the read-only option too.

You can also create a read-only bind mount

```
docker run --mount \
source=<volume-name>,destination=/ \
path/in/container,readonly
```

Disable the default bridge and use a dedicated network to expose the host interface.

It is recommended to export your logs to an external service.

DON’T USE

```
docker run --privileged
```

Which is giving your container root capabilities on the host.

```
docker run --cgroup-parent
```

By allowing shared resources with the host, you are putting it at risk.

DON’T USE

```
docker run --network=”host”
```

But instead create a dedicated network isolate the host's network interface:

```
docker network create web
```

SCANNING

Some free scanning tools :
- Clair
- Trivy
- Docker Bench for Security