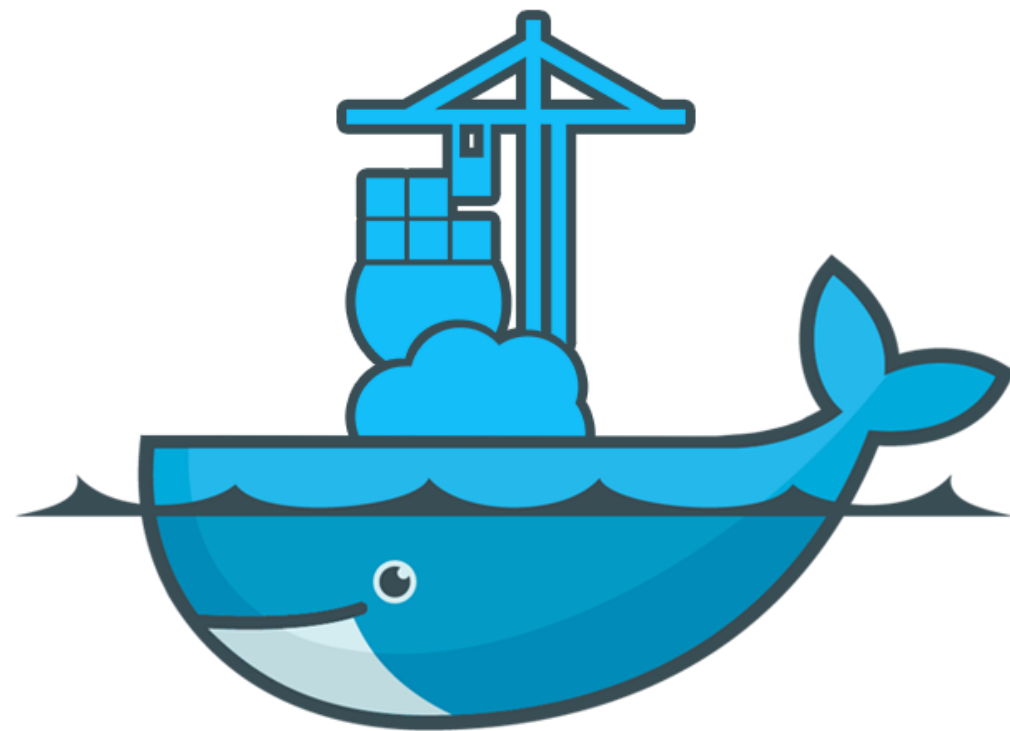


# Melhorando segurança de Containers

Boas práticas e ferramentas essenciais



# whoami



**Guilherme Fujii Filgueiras** [Fazer verificação agora](#)

Product Security Specialist | DevSecOps Architect | Web Security Evangelist

São Paulo, Brasil · [Informações de contato](#)

+ de 500 conexões

 Ford Brasil

# Container?

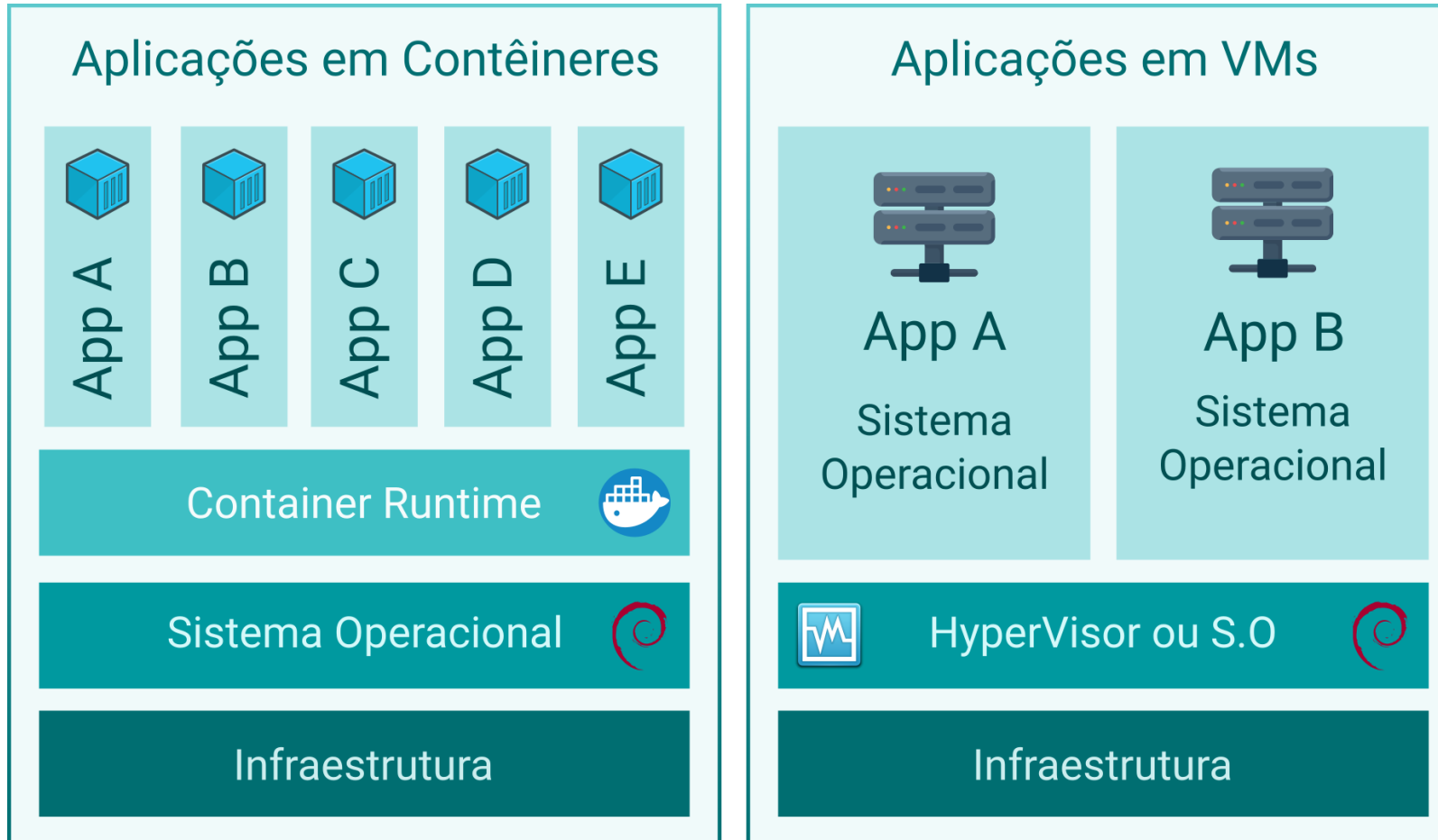


ubuntu

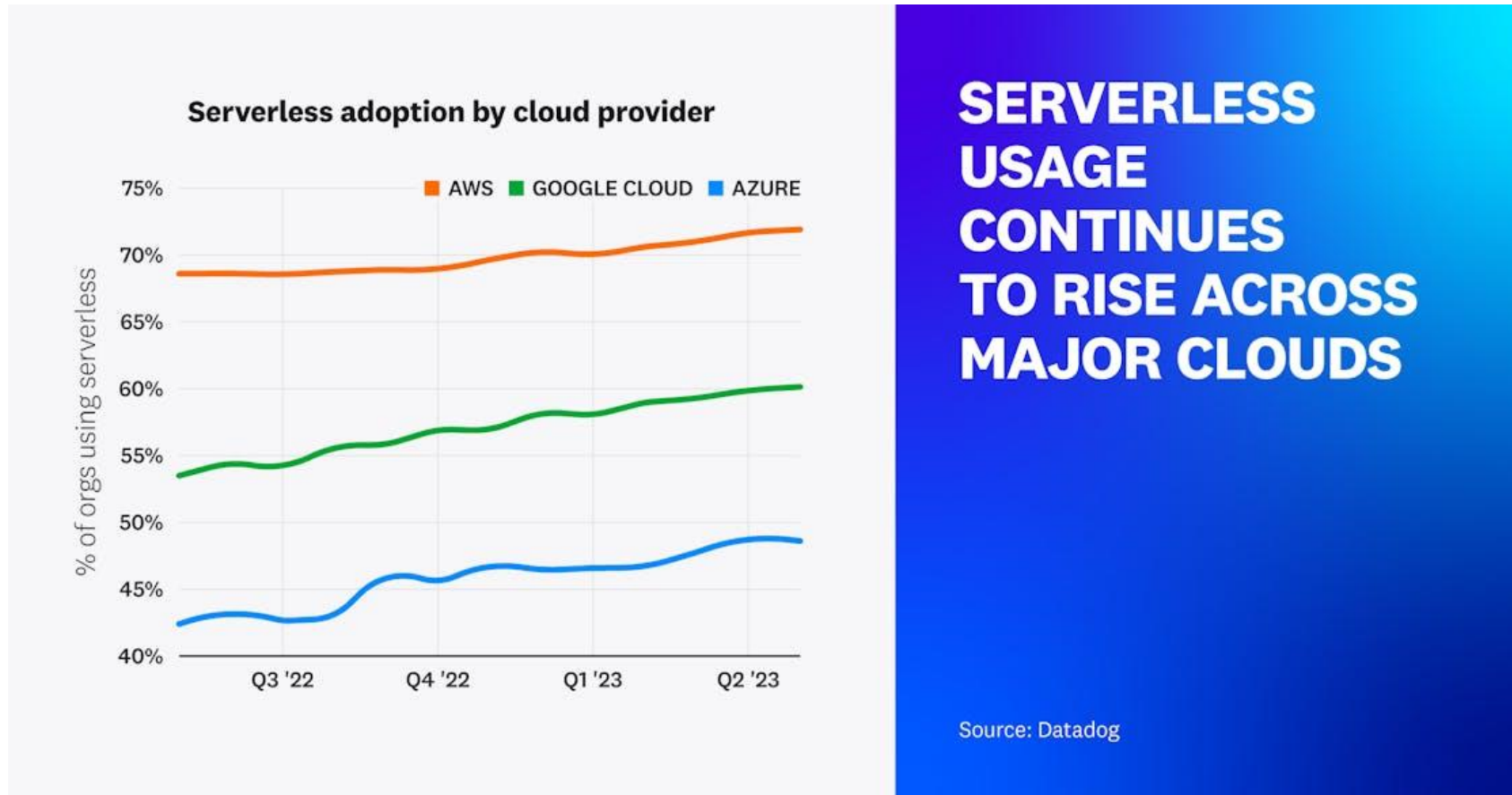


ANGULAR

# Containers vs VMs (Isolamento)

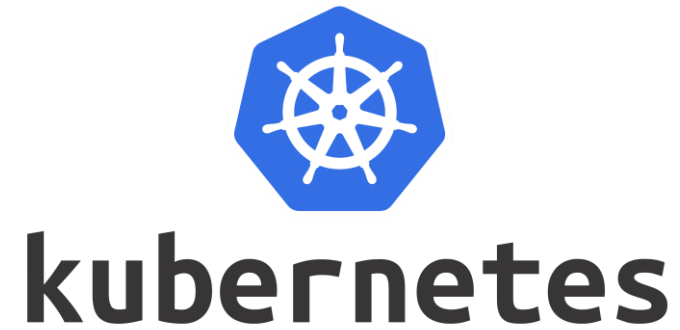


# Importância de conhecer containers



[The State of Serverless | Datadog](#)

# Ambientes formados por containers



AWS Lambda



Google  
Cloud Run

# Como funciona a construção?

```
FROM example/rails
MAINTAINER example@example@gmail.com

# Add here your preinstall lib(e.g. imagemagick, mysql lib, ssh config)
## Install imagemagick
RUN apt-get update
RUN apt-get -qq -y install libmagickwand-dev imagemagick

## Install for mysql gem
RUN apt-get install -qq -y mysql-server mysql-client libmysqlclient-dev

## Install for Webshots
RUN apt-get install libssl0.9.8 -y
RUN apt-get install ttf-unfonts-core -y

#(required) Install Rails App
ADD Gemfile /app/Gemfile
ADD Gemfile.lock /app/Gemfile.lock
RUN bundle install --without development test
ADD ./app

# Overwrite unicorn
ADD config/unicorn.rb /app/config/unicorn.rb

FROM example/rails
MAINTAINER example@example@gmail.com

# Add here your preinstall lib(e.g. imagemagick, mysql lib, ssh config)
## Install imagemagick
RUN apt-get update
RUN apt-get -qq -y install libmagickwand-dev imagemagick

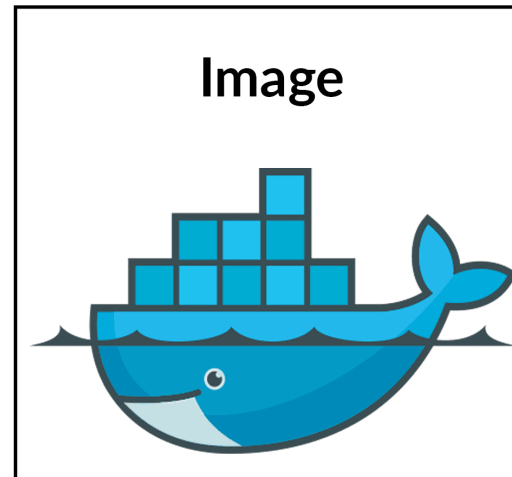
## Install for mysql gem
RUN apt-get install -qq -y mysql-server mysql-client libmysqlclient-dev

## Install for Webshots
RUN apt-get install libssl0.9.8 -y
RUN apt-get install ttf-unfonts-core -y

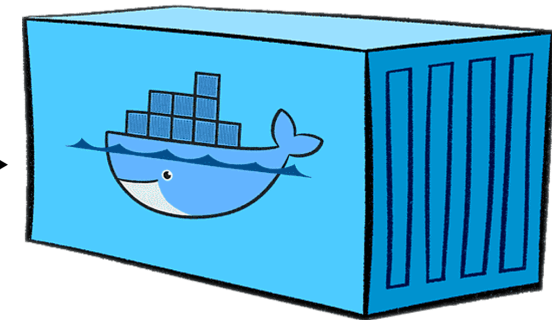
#(required) Install Rails App
ADD Gemfile /app/Gemfile
ADD Gemfile.lock /app/Gemfile.lock
RUN bundle install --without development test
ADD ./app

# Overwrite unicorn
ADD config/unicorn.rb /app/config/unicorn.rb
```

Dockerfile

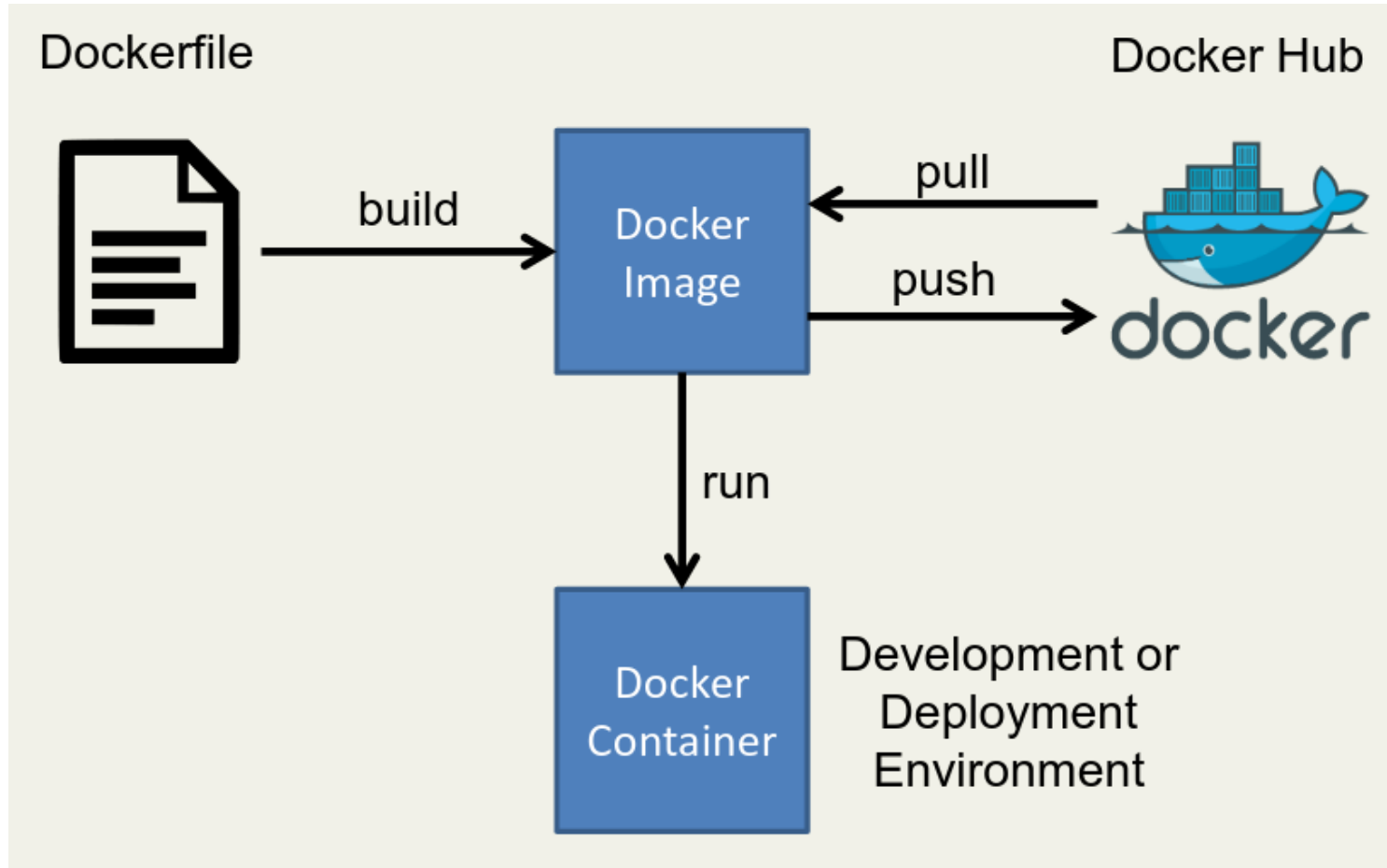


Docker Image



Docker Container

# Registry

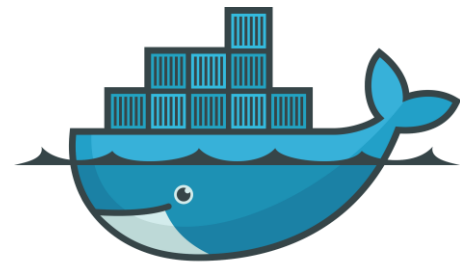




# Isolamento cria segurança? Algumas ameaças

- Network mal configurada
- Imagem com vulnerabilidades e malware injetado
- Imagem mal configurada
- Exposição de secrets
- Escape de container
- Host mal configurado
- Código com vulnerabilidades (Ex: RCE)
- Ataques de supply chain

# Hora de buildar



docker



ANGULAR

# Dockerfile

```
FROM ubuntu

USER 0:0

ARG CLIENT_SECRET
ENV CLIENT_SECRET $CLIENT_SECRET

RUN apt-get install -y nodejs

RUN npm install -g @angular/cli

WORKDIR /app

COPY package*.json ./

RUN npm install

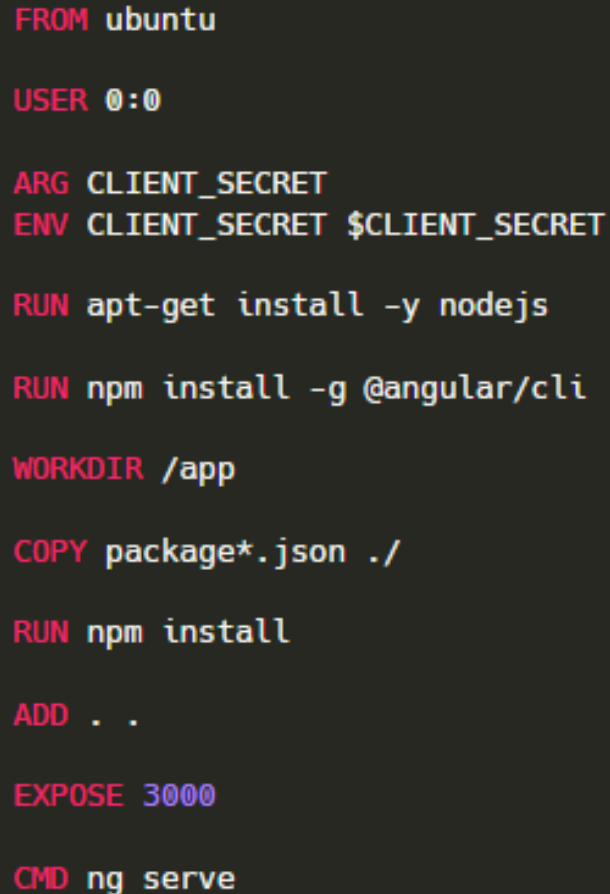
ADD . .

EXPOSE 3000

CMD ng serve
```



# O que tem de ruim aqui?

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. The text is displayed in a monospaced font with syntax highlighting: 'FROM' is red, 'ubuntu' is white, 'USER' is red, '0:0' is white, 'ARG' is red, 'CLIENT\_SECRET' is white, 'ENV' is red, '\$CLIENT\_SECRET' is white, 'RUN' is red, 'apt-get install -y nodejs' is white, 'npm install -g @angular/cli' is white, 'WORKDIR' is red, '/app' is white, 'COPY' is red, 'package\*.json ./' is white, 'npm install' is white, 'ADD' is red, './' is white, 'EXPOSE' is red, '3000' is white, 'CMD' is red, and 'ng serve' is white.

```
FROM ubuntu

USER 0:0

ARG CLIENT_SECRET
ENV CLIENT_SECRET $CLIENT_SECRET

RUN apt-get install -y nodejs

RUN npm install -g @angular/cli

WORKDIR /app

COPY package*.json ./

RUN npm install

ADD . .

EXPOSE 3000

CMD ng serve
```

- Imagem sem tag, sem digest e sem private registry
- Imagem generalista (muitos pacotes sem utilização que podem conter CVEs críticas)
- Usuário root rodando o container
- Secrets expostos na imagem
- Usando ADD
- CMD executando shell (/bin/sh -c 'comando') não sendo necessário

# Melhorando

```
FROM
registry.company.com/nodejs:22.1@sha256:d3fe026f6a0377a10f9e9e2a22d3a7a1bc2ca6b403d4de882adc83c7497883b1 AS
builder

USER 0:0

RUN dnf install -y && dnf clean all

RUN npm install -g @angular/cli

WORKDIR /app

COPY .npmrc $PREFIX/.npmrc

COPY package*.json ./

RUN --mount=type=secret,id=AR_TOKEN \
    npm install

RUN ng build --configuration=production

FROM
registry.company.com/nginx:1.24@sha256:8870b07b39166e44325116bde4f3701e8fbd5cd328e0e319310529e9e7e122e2

COPY --from=builder /app/dist/angular-app /usr/share/nginx/html

USER nginx

EXPOSE 80

# Start Nginx
CMD ["nginx", "-g", "daemon off;"]
```



# Melhorando

- Registry seguro
- Imagem com tag e digest
- Imagem mínima
- Secrets com mount no build (Maneira segura)
- Usuário não root rodando container
- COPY ao invés de ADD (ADD pode puxar pastas de origens remotas)

## Melhorando (Golden Base Image)

Imagens pré-aprovadas pela organização para uso no build e run de aplicações



# Secrets em containers (build)

```
RUN --mount=type=secret,id=aws \  
    AWS_SHARED_CREDENTIALS_FILE=/run/secrets/aws \  
    aws s3 cp ...
```

```
$ docker build --secret id=API_TOKEN .
```



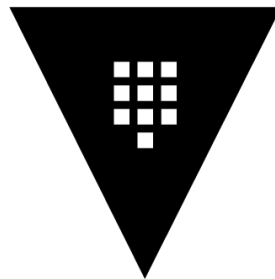
# E se eu precisar do secret em runtime?



Secret  
Manager



AWS **Secrets Manager**



HashiCorp

**Vault**

E se eu precisar do secret em runtime?

# Não passem secrets para a imagem Docker!

```
~/Desktop/docker/secret-files$ cat 70714a454a832620066a91ed777b24aa94e5a22d6251fc1f01ef1543d1cfd80a.json | jq '.history'
[
  {
    "created": "2023-02-11T04:46:42.449083344Z",
    "created_by": "/bin/sh -c #(nop) ADD file:40887ab7c06977737e63c215c9bd297c0c74de8d12d16ebdf1c3d40ac392f62d in /"
  },
  {
    "created": "2023-02-11T04:46:42.558343068Z",
    "created_by": "/bin/sh -c #(nop) CMD [\"/bin/sh\"]",
    "empty_layer": true
  },
  {
    "created": "2023-04-05T13:53:07.18405666Z",
    "created_by": "/bin/sh -c echo \"segredo\" > /secret.txt"
  },
  {
    "created": "2023-04-05T13:53:07.694725572Z",
    "created_by": "/bin/sh -c rm /secret.txt"
  }
]
```

# DevSecOps e Containers (Análise de CVE)



```
alpine:3.11 (alpine 3.11.7)
=====
Total: 13 (UNKNOWN: 1, LOW: 2, MEDIUM: 4, HIGH: 6, CRITICAL: 0)
```

LIBRARY	VULNERABILITY ID	SEVERITY	INSTALLED VERSION	FIXED VERSION	TITLE
apk-tools	CVE-2021-30139	UNKNOWN	2.10.5-r0	2.10.6-r0	-->avd.aquasec.com/nvd/cve-2021-30139
busybox	CVE-2021-28831	HIGH	1.31.1-r9	1.31.1-r10	busybox: invalid free or segmentation fault via malformed gzip data -->avd.aquasec.com/nvd/cve-2021-28831
libcrypto1.1	CVE-2021-23840		1.1.1i-r0	1.1.1j-r0	openssl: integer overflow in CipherUpdate -->avd.aquasec.com/nvd/cve-2021-23840
	CVE-2021-3450			1.1.1k-r0	openssl: CA certificate check bypass with X509_V_FLAG_X509_STRICT -->avd.aquasec.com/nvd/cve-2021-3450
	CVE-2021-23841	MEDIUM		1.1.1j-r0	openssl: NULL pointer dereference in X509_issuer_and_serial_hash() -->avd.aquasec.com/nvd/cve-2021-23841
	CVE-2021-3449			1.1.1k-r0	openssl: NULL pointer dereference in signature_algorithms processing -->avd.aquasec.com/nvd/cve-2021-3449
	CVE-2021-23839	LOW		1.1.1j-r0	openssl: incorrect SSLv2 rollback protection -->avd.aquasec.com/nvd/cve-2021-23839
libssl1.1	CVE-2021-23840	HIGH			openssl: integer overflow in CipherUpdate

# DevSecOps e Containers (Linter de Dockerfile)



**Hadolint**

```
DL4000 Specify a maintainer of the Dockerfile
DL3006 Always tag the version of an image explicitly.
1 FROM debian
   SC1007 Remove space after = if trying to assign a value (for empty string, use var='' ... ).
   SC2154 node_version is referenced but not assigned.
   DL3009 Delete the apt-get lists after installing something
2 RUN node_version= "0.10" \
3     && apt-get update && apt-get -y install nodejs="$node_version"
4 COPY package.json usr/src/app
   DL3003 Use WORKDIR to switch to a directory
5 RUN cd /usr/src/app \
6     && npm install node-static
7
   DL3011 Valid UNIX ports range from 0 to 65535
8 EXPOSE 80000
9 CMD ["npm", "start"]
```

# Técnicas de Segurança

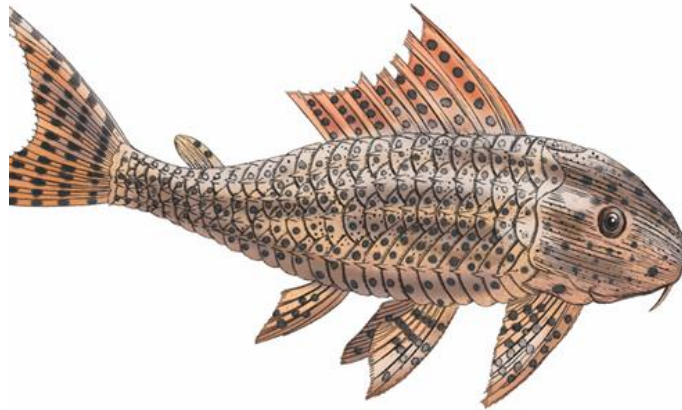
- “Immutable Containers” (incluir tudo no build para não ter que instalar runtime)
- Não colocar dados sensíveis no container
- Não montar volumes em locais sensíveis (mount no /etc ou /bin por exemplo)
- Rodar o container sem usuário root
- Cuidado com RUN (RCE). Limite privilégios para editar de imagens
- Sempre que possível, usar multi-stage build
- Usar golden base images
- Usar imagens mínimas
- Possuir e puxar imagens sempre de um registry privado
- Usar sempre registry/container:tag@digest para máxima verificação de integridade

# Bibliografia + Recomendação

O'REILLY®

## Container Security

Fundamental Technology Concepts that  
Protect Containerized Applications



Liz Rice