Securing Containers by Breaking In

Snyk

@BrianVerm

Container?

Brian Vermeer

Developer Advocate

snyk

@BrianVerm

- ✓ Java Champion
- ✓ Virtual JUG leader
- ✓ NLJUG leader
- ✓ DevSecCon co-leader
- ✓ Oracle Groundbreaker Ambassador
- ✓ Foojay Community Manager Security

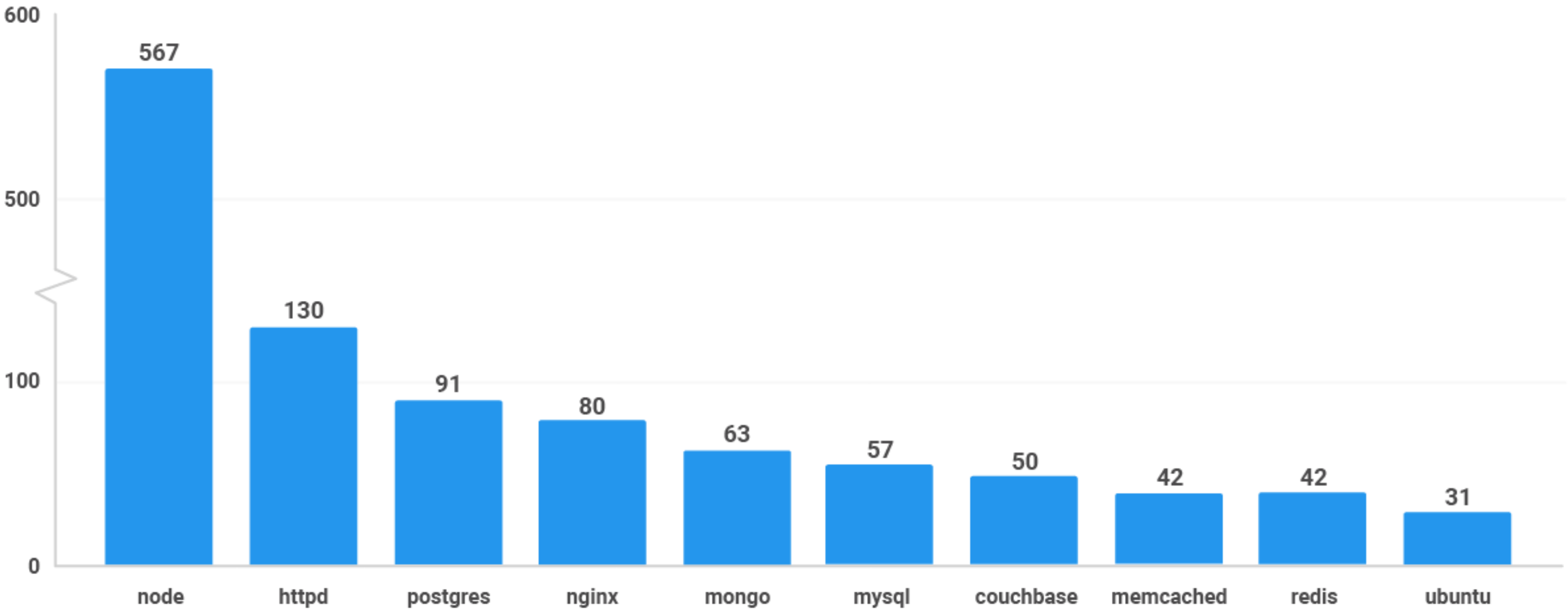
1 billion weekly d/l

of container images

Best Practices
for Docker Image Security

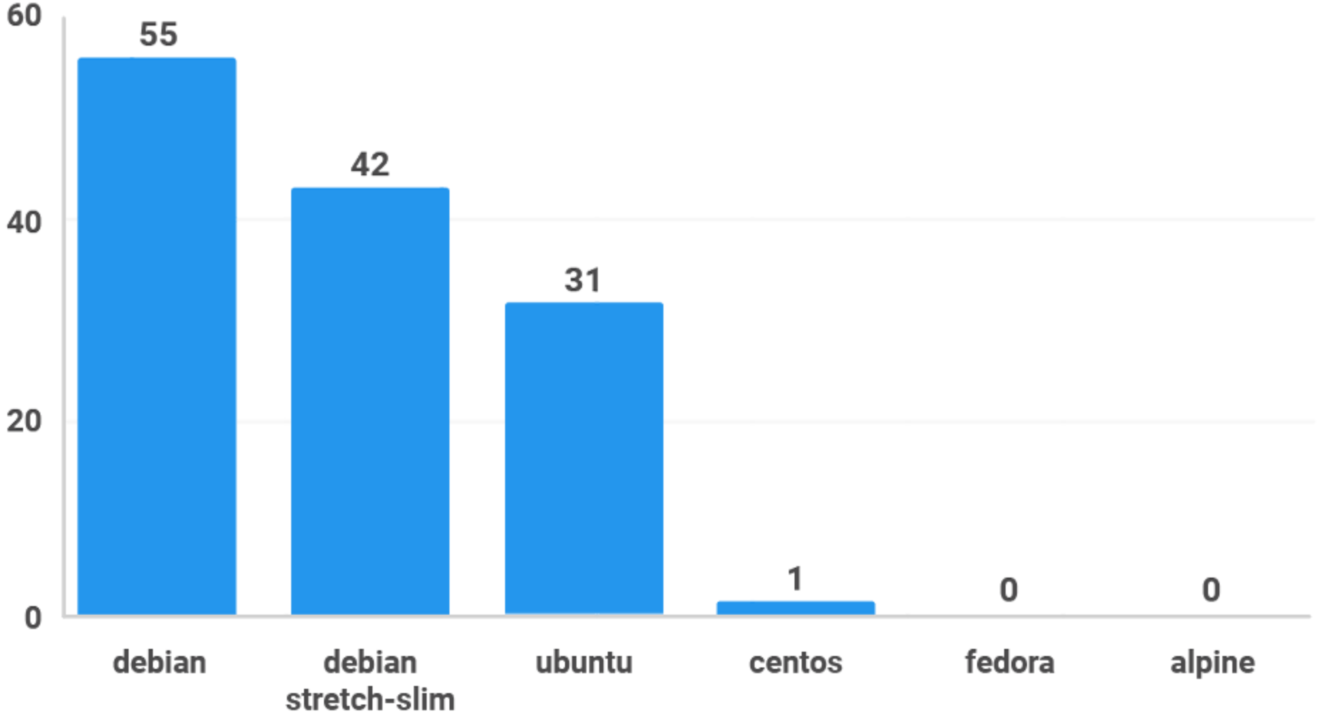
# Prefer Minimal Base Images

Vulnerabilities per Docker image

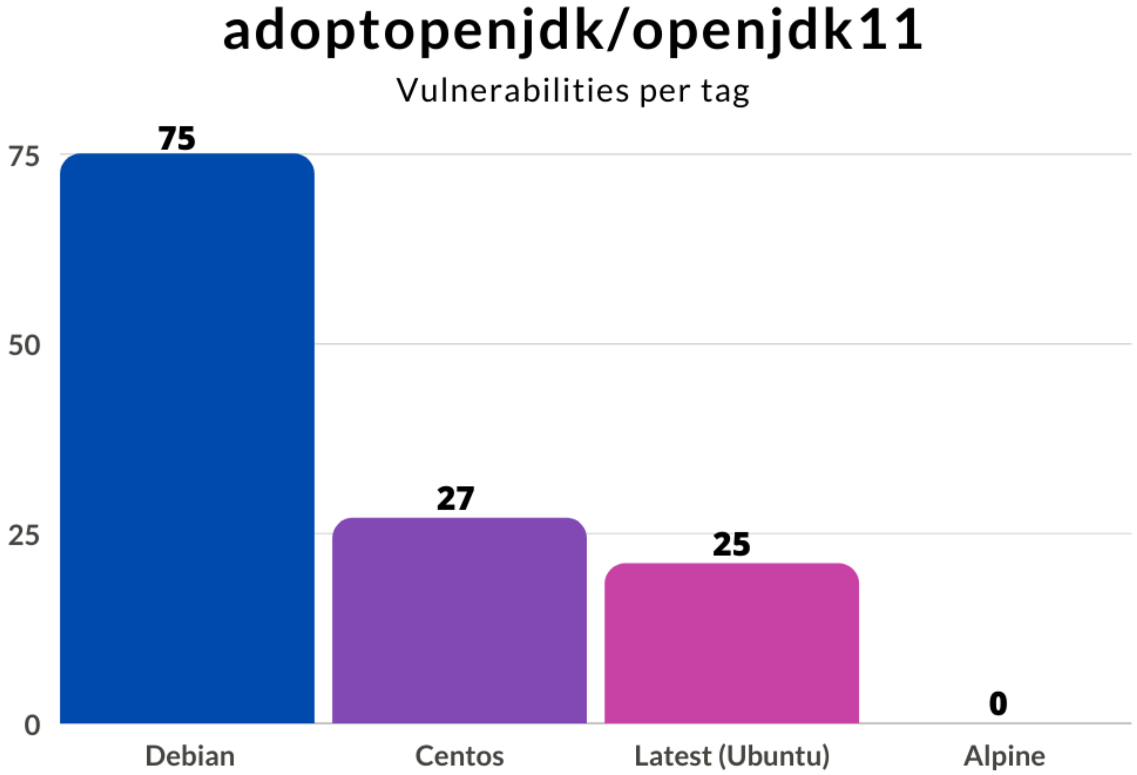


source: https://snyk.io/blog/shifting-docker-security-left/

Vulnerablilities in OS images



source: https://snyk.io/blog/shifting-docker-security-left/

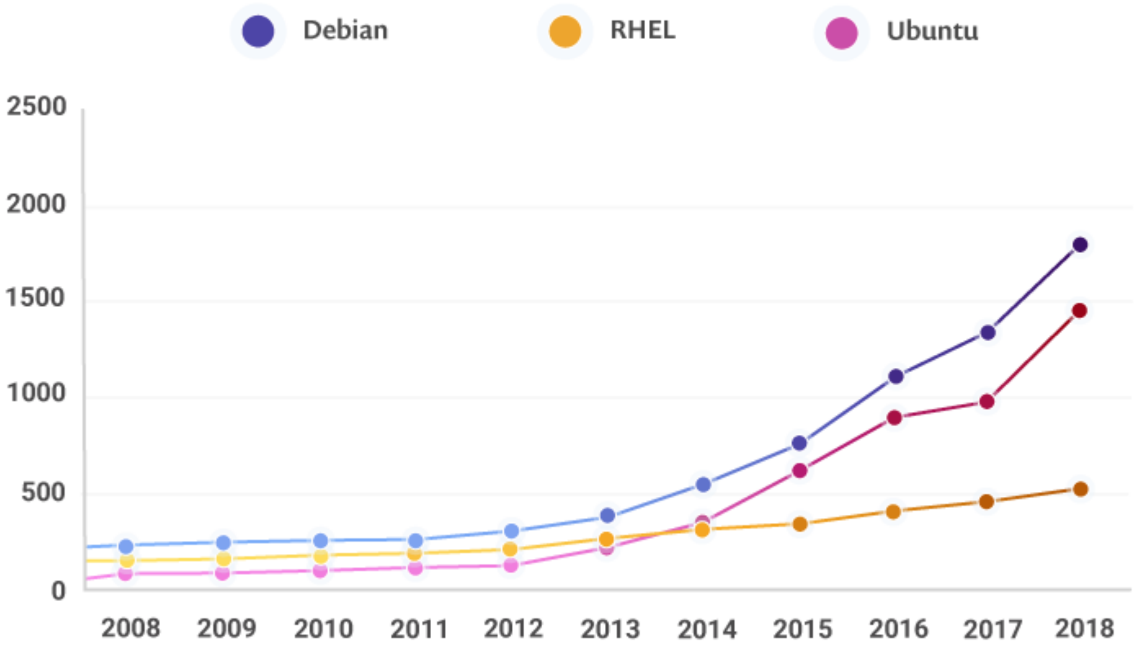


source: https://snyk.io/blog/docker-for-java-developers/

@BrianVerm

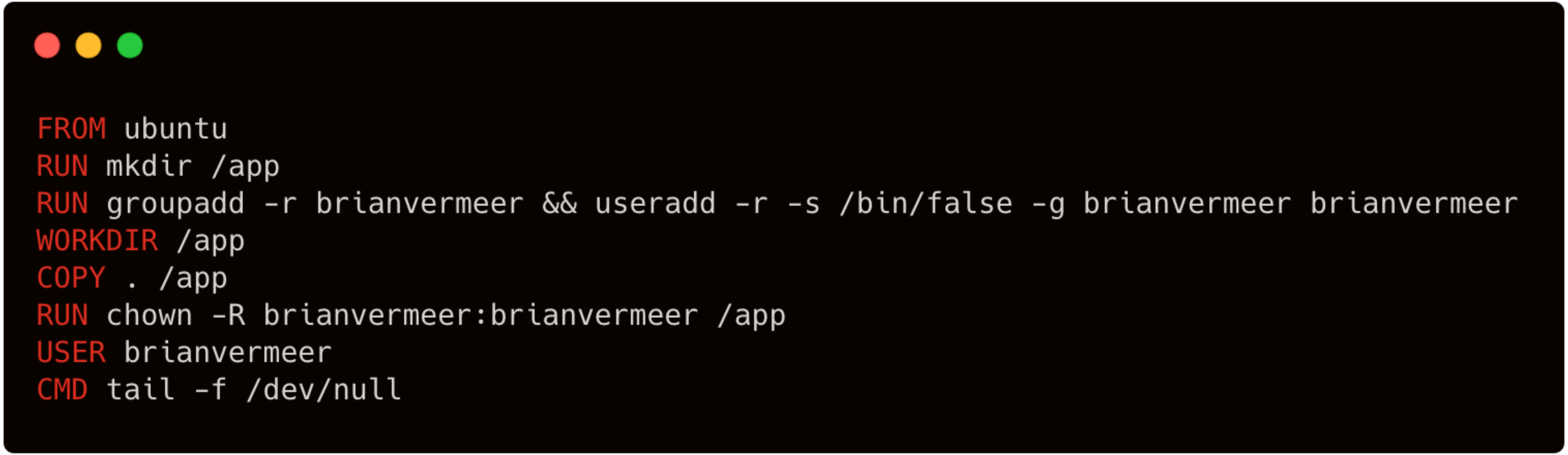
Linux OS vulnerabilities steadily increasing

snyk

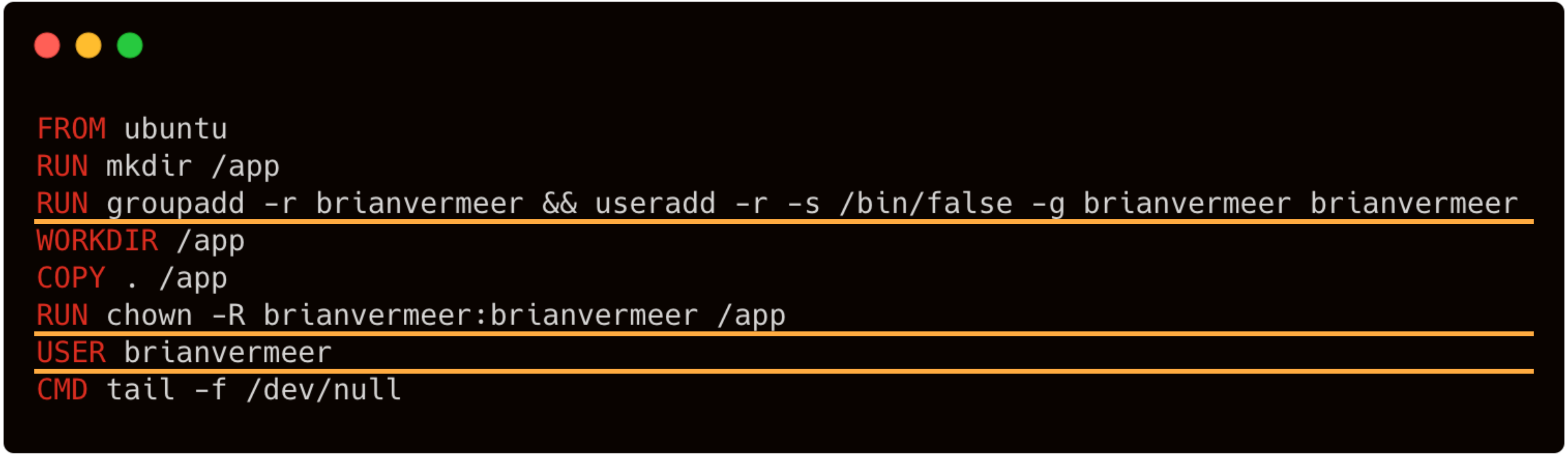


source: https://snyk.io/opensourcesecurity-2019

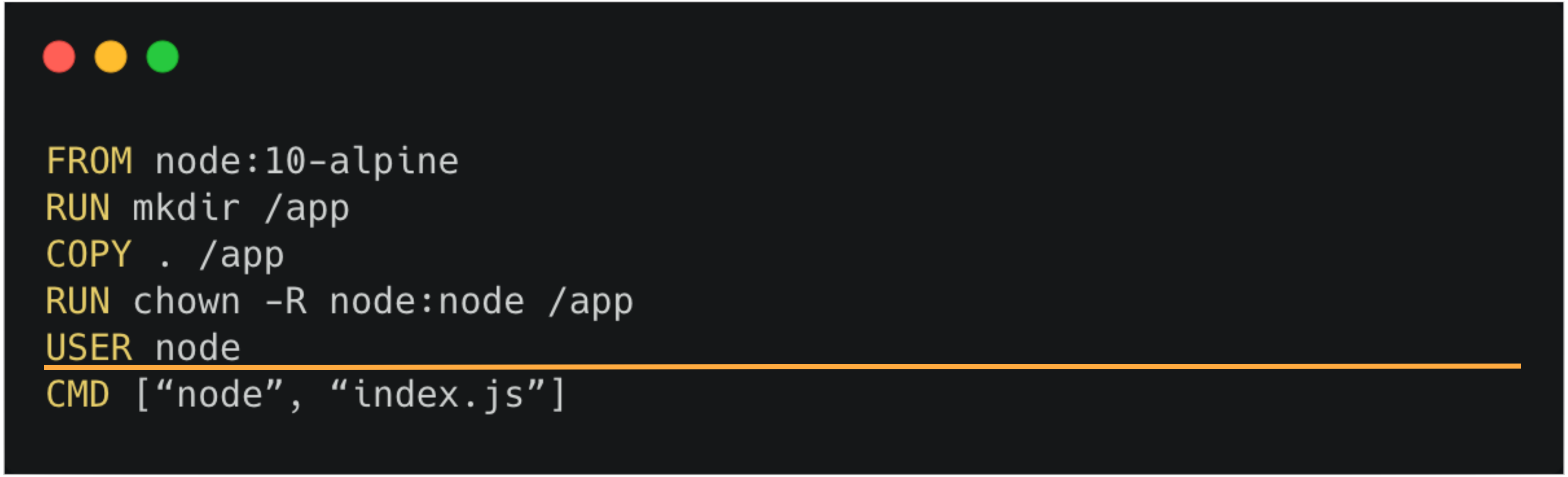
# Least Privileged User



```
FROM ubuntu
RUN mkdir /app
RUN groupadd -r brianvermeer && useradd -r -s /bin/false -g brianvermeer brianvermeer
WORKDIR /app
COPY . /app
RUN chown -R brianvermeer:brianvermeer /app
USER brianvermeer
CMD tail -f /dev/null
```

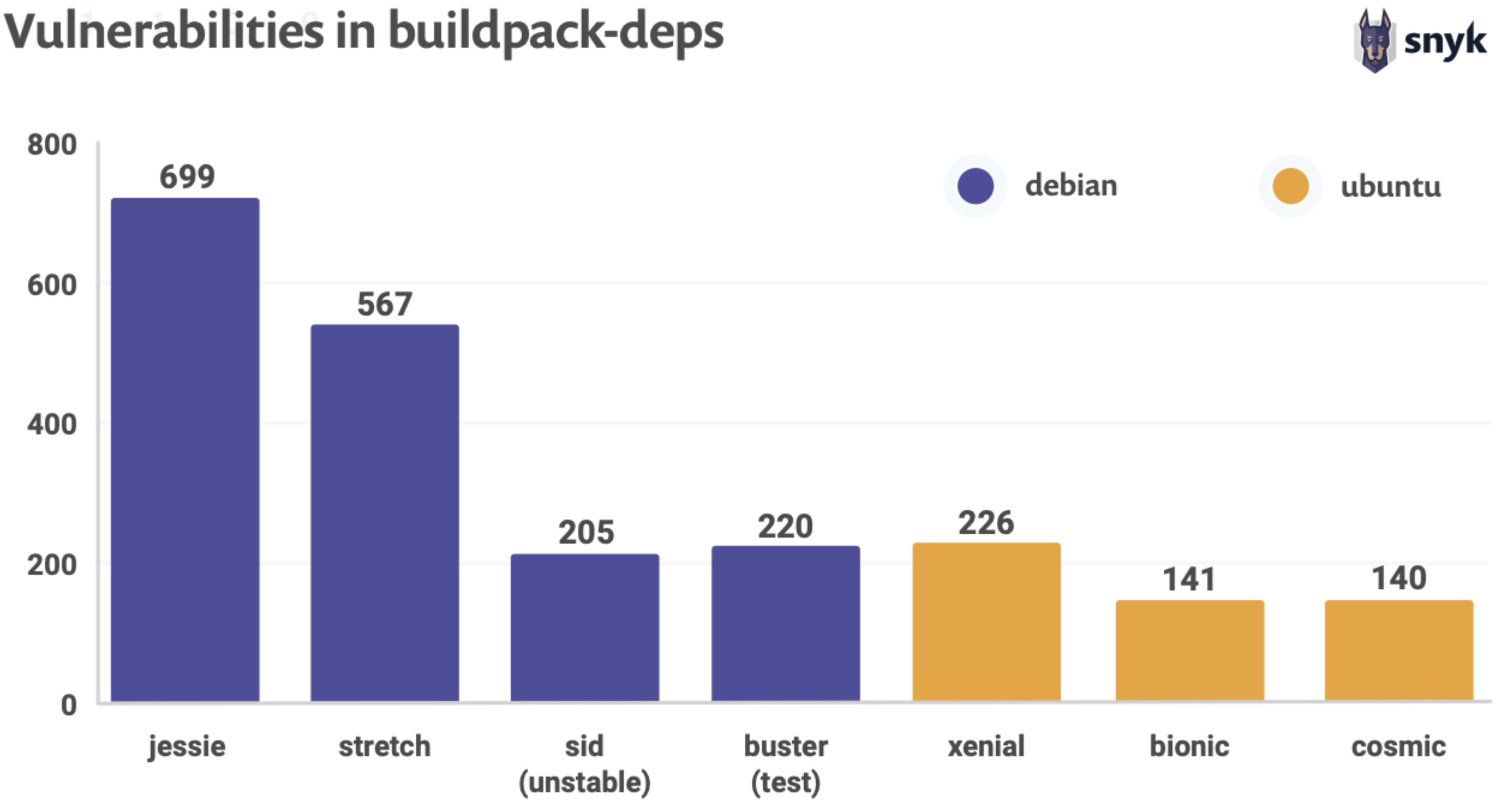


```
FROM ubuntu
RUN mkdir /app
RUN groupadd -r brianvermeer && useradd -r -s /bin/false -g brianvermeer brianvermeer
WORKDIR /app
COPY . /app
RUN chown -R brianvermeer:brianvermeer /app
USER brianvermeer
CMD tail -f /dev/null
```



```
FROM node:10-alpine
RUN mkdir /app
COPY . /app
RUN chown -R node:node /app
USER node
CMD [“node”, “index.js”]
```

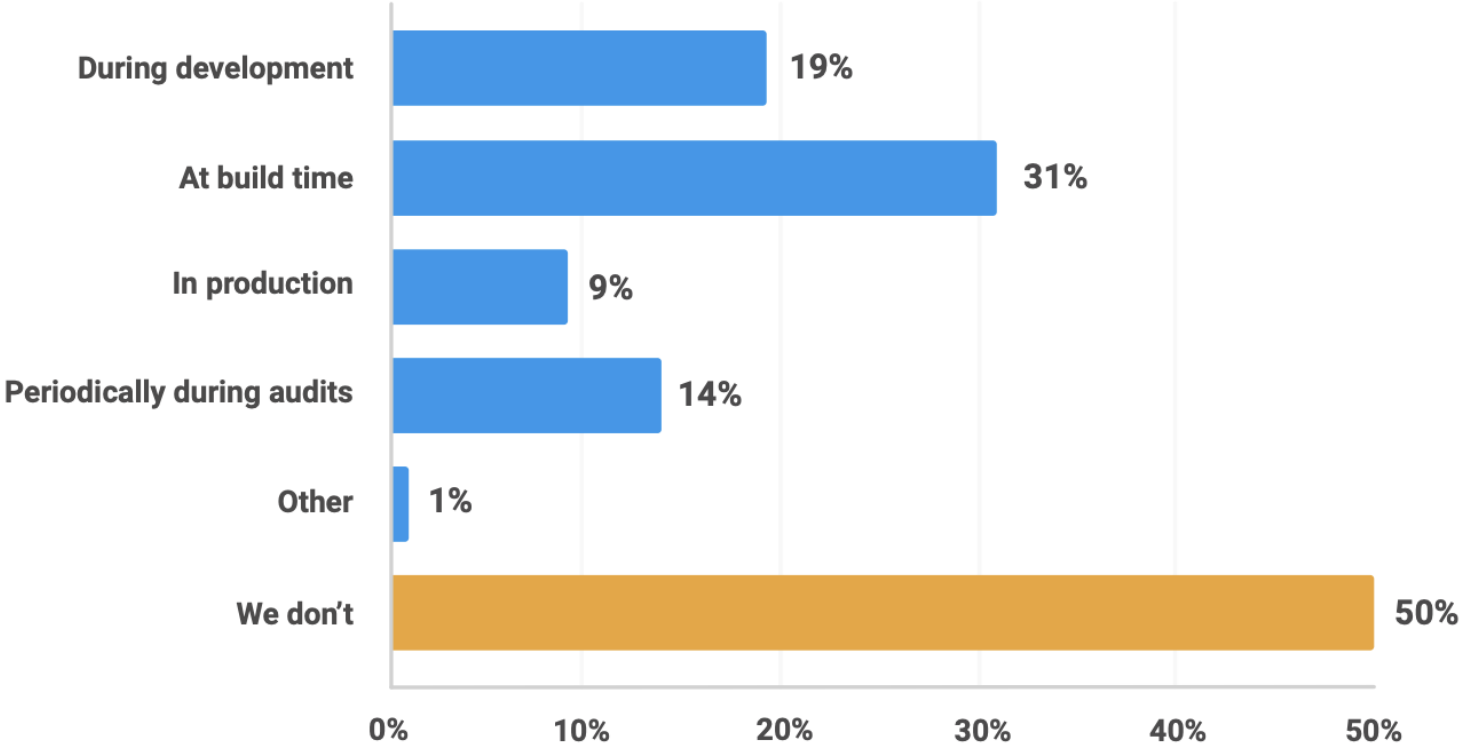
# Find, Fix and Monitor Open Source Vulnerabilities in the OS



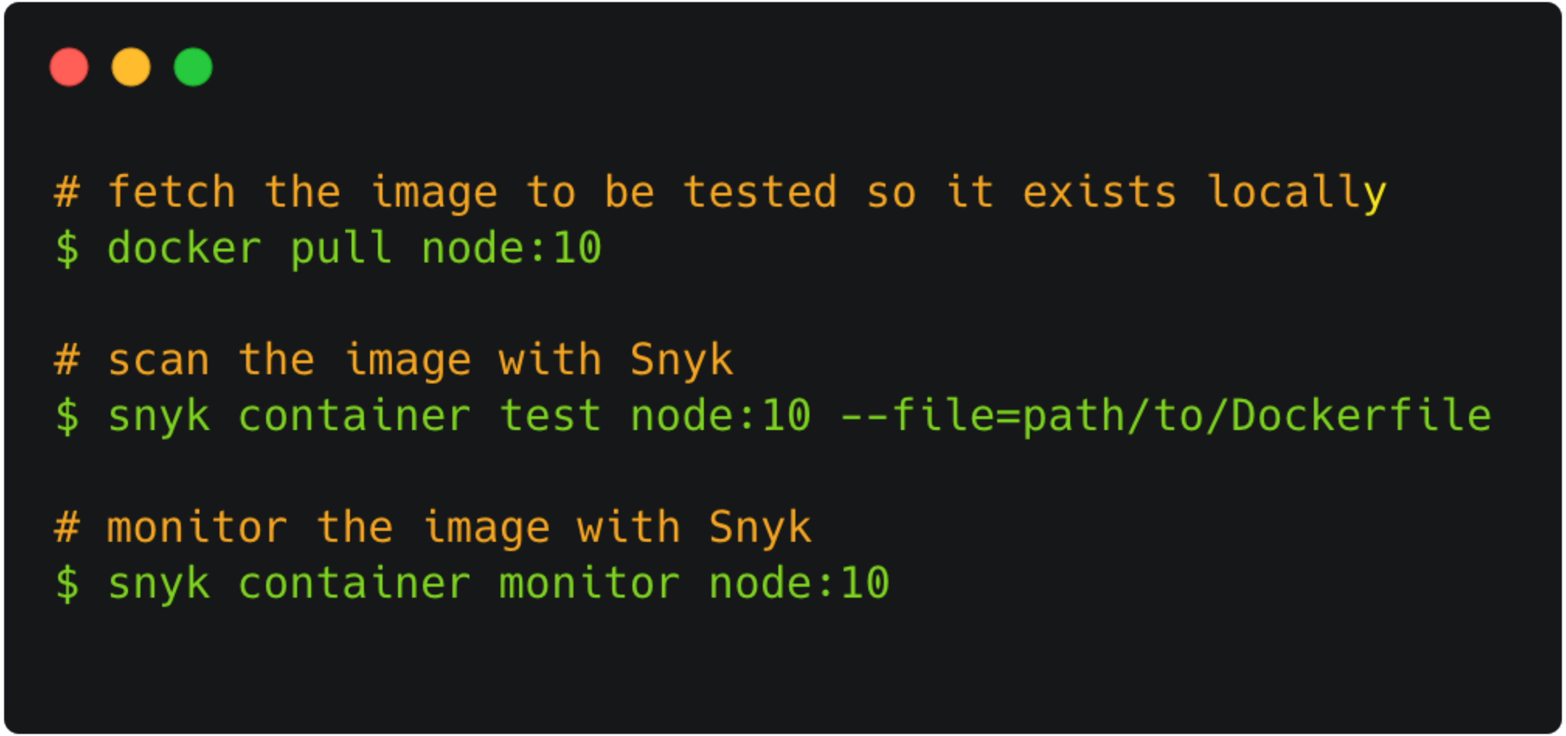
Vulnerabilities in buildpack-deps

source: https://snyk.io/blog/shifting-docker-security-left/

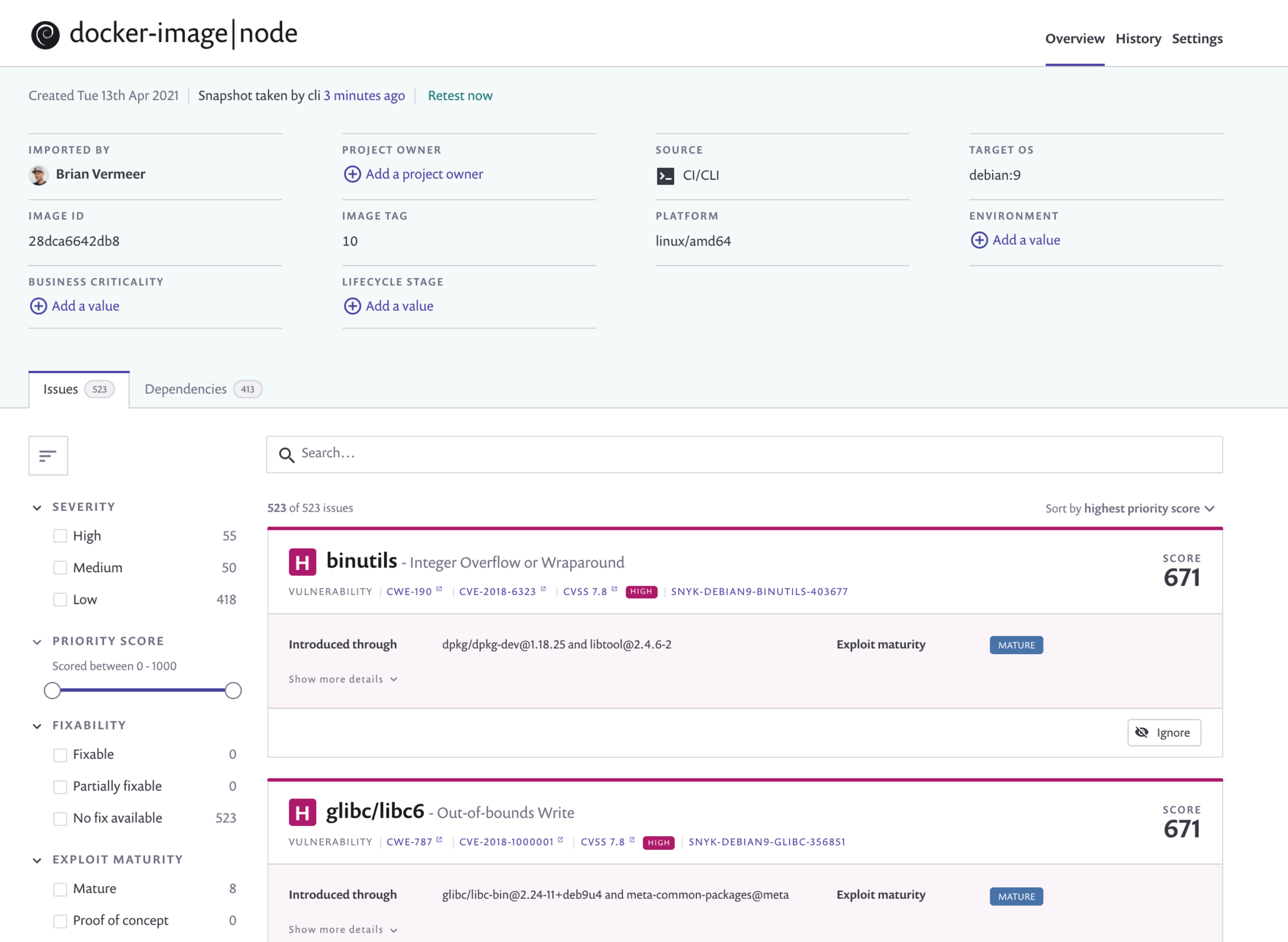
When do you scan your Docker image for OS vulns?



source: https://snyk.io/opensourcesecurity-2019



```
# fetch the image to be tested so it exists locally
$ docker pull node:10

# scan the image with Snyk
$ snyk container test node:10 --file=path/to/Dockerfile

# monitor the image with Snyk
$ snyk container monitor node:10
```



docker-image|node

Created Tue 13th Apr 2021 | Snapshot taken by cli 3 minutes ago | Retest now

IMPORTED BY Brian Vermeer	PROJECT OWNER ⊕ Add a project owner	SOURCE CI/CLI	TARGET OS debian:9
IMAGE ID 28dca6642db8	IMAGE TAG 10	PLATFORM linux/amd64	ENVIRONMENT ⊕ Add a value
BUSINESS CRITICALITY ⊕ Add a value	LIFECYCLE STAGE ⊕ Add a value		

SEVERITY

□ High	55
□ Medium	50
□ Low	418

PRIORITY SCORE

Scored between 0 - 1000

FIXABILITY

□ Fixable	0
□ Partially fixable	0
□ No fix available	523

EXPLOIT MATURITY

□ Mature	8
□ Proof of concept	0

523 of 523 issues

Sort by **highest priority score**

H binutils - Integer Overflow or Wraparound

SCORE **671**

VULNERABILITY | CWE-190 | CVE-2018-6323 | CVSS 7.8 HIGH | SNYK-DEBIAN9-BINUTILS-403677

Introduced through dpkg/dpkg-dev@1.18.25 and libtool@2.4.6-2

Exploit maturity MATURE

Show more details

H glibc/libc6 - Out-of-bounds Write

SCORE **671**

VULNERABILITY | CWE-787 | CVE-2018-1000001 | CVSS 7.8 HIGH | SNYK-DEBIAN9-GLIBC-356851

Introduced through glibc/libc-bin@2.24-11+deb9u4 and meta-common-packages@meta

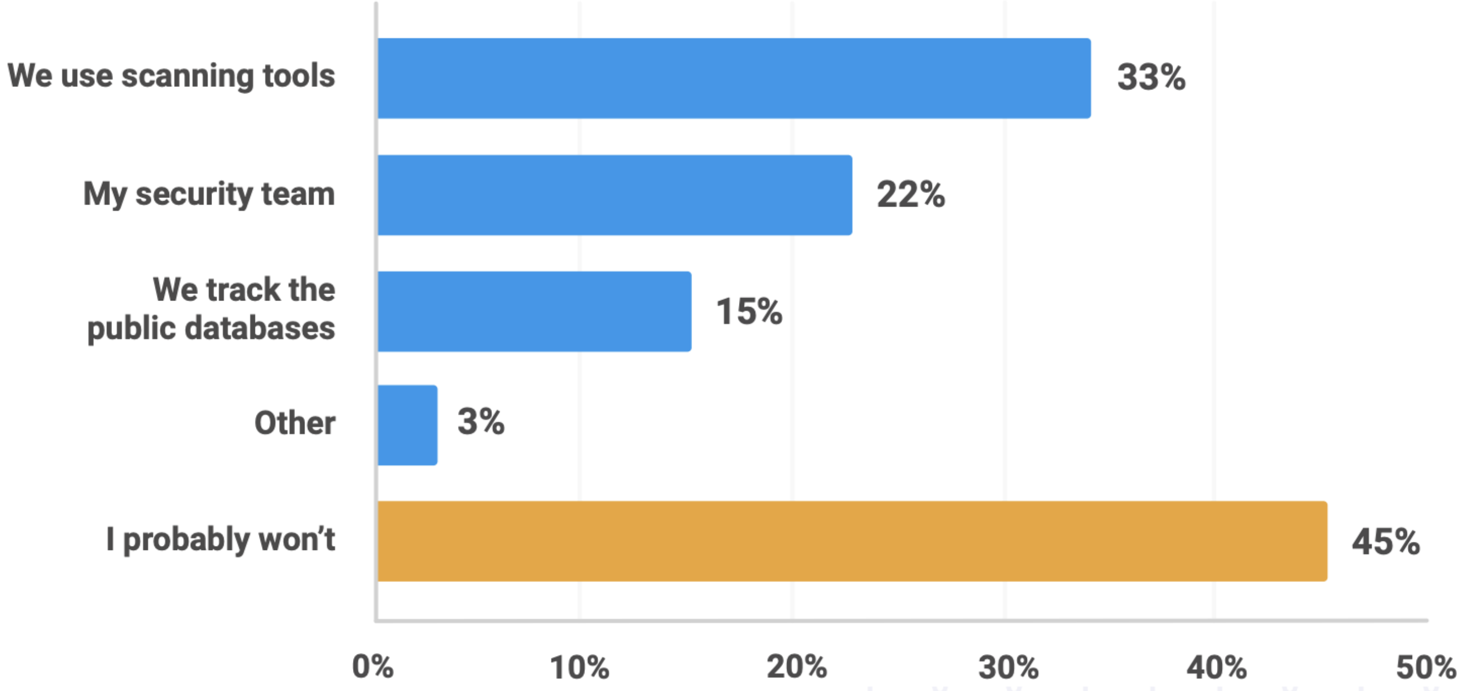
Exploit maturity MATURE

Show more details

44%

# of docker image vulnerabilities can be fixed with newer base images

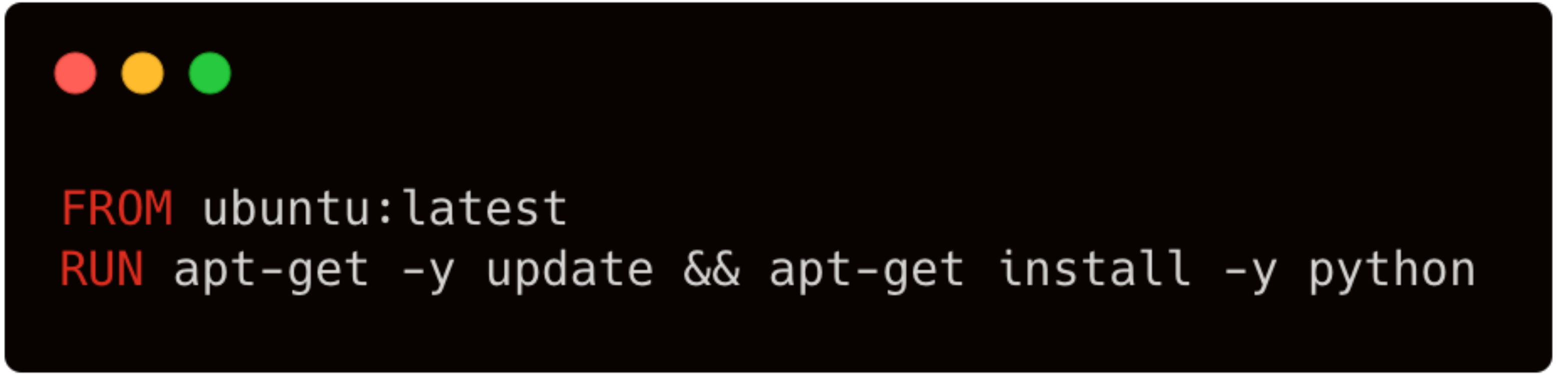
How do you find out about new vulnerabilities in your deployed containers?



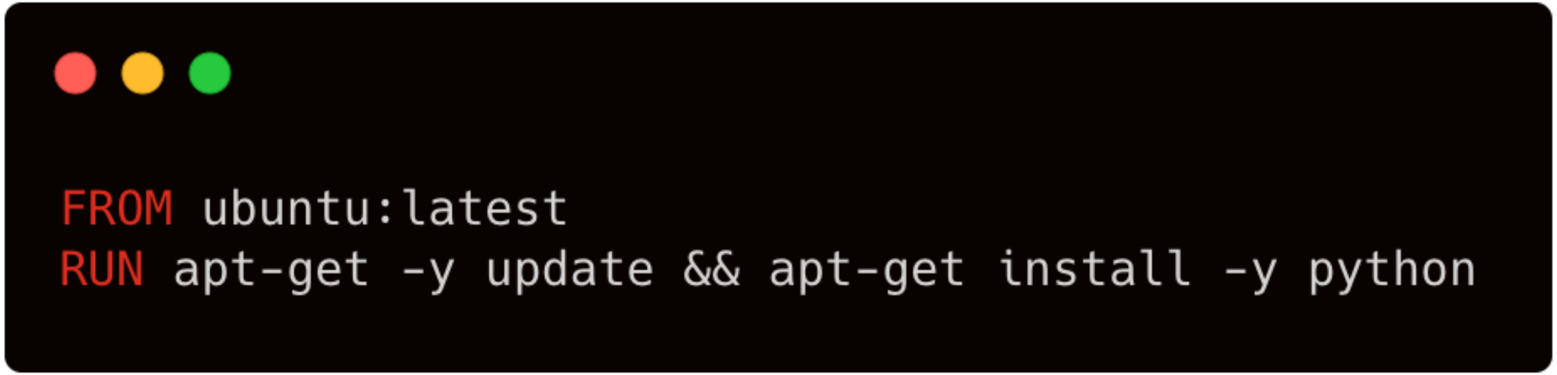
source: https://snyk.io/opensourcesecurity-2019

20%

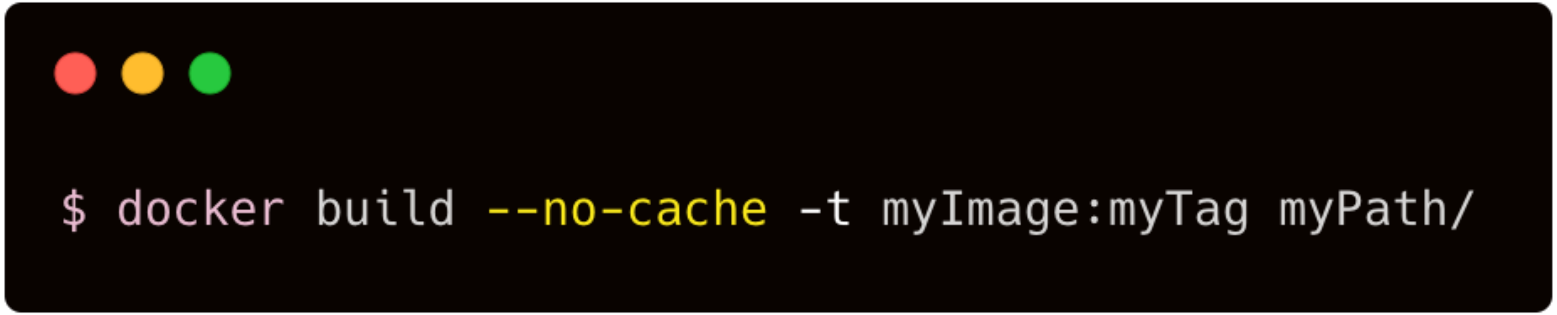
# of docker image vulnerabilities can be fixed just by rebuilding them



```
FROM ubuntu:latest
RUN apt-get -y update && apt-get install -y python
```

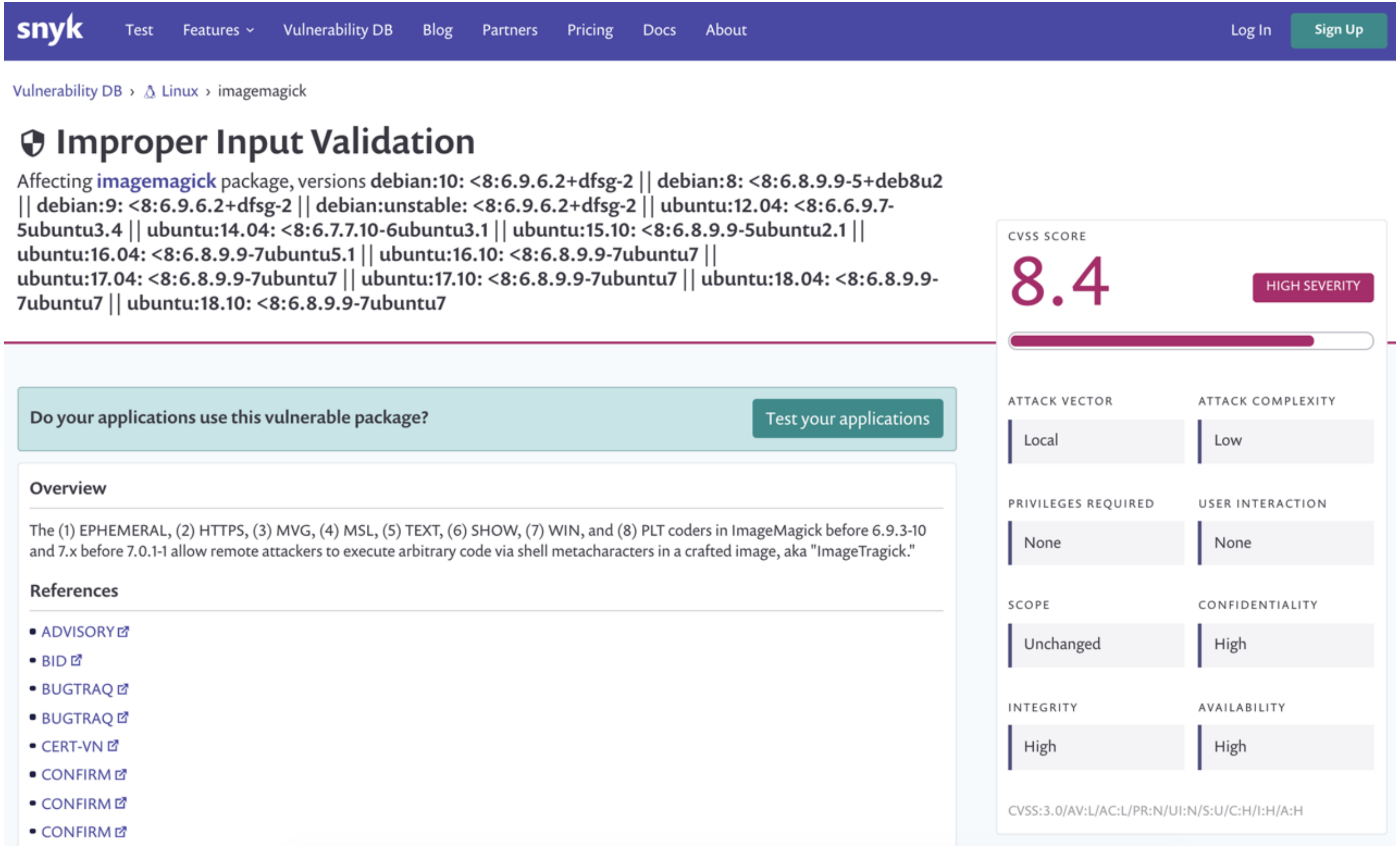


```
FROM ubuntu:latest
RUN apt-get -y update && apt-get install -y python
```



```
$ docker build --no-cache -t myImage:myTag myPath/
```

What can possibly go wrong with container image vulnerabilities?



Vulnerability DB › Linux › imagemagick

Improper Input Validation

Affecting **imagemagick** package, versions **debian:10: <8:6.9.6.2+dfsg-2** || **debian:8: <8:6.8.9.9-5+deb8u2** || **debian:9: <8:6.9.6.2+dfsg-2** || **debian:unstable: <8:6.9.6.2+dfsg-2** || **ubuntu:12.04: <8:6.6.9.7-5ubuntu3.4** || **ubuntu:14.04: <8:6.7.7.10-6ubuntu3.1** || **ubuntu:15.10: <8:6.8.9.9-5ubuntu2.1** || **ubuntu:16.04: <8:6.8.9.9-7ubuntu5.1** || **ubuntu:16.10: <8:6.8.9.9-7ubuntu7** || **ubuntu:17.04: <8:6.8.9.9-7ubuntu7** || **ubuntu:17.10: <8:6.8.9.9-7ubuntu7** || **ubuntu:18.04: <8:6.8.9.9-7ubuntu7** || **ubuntu:18.10: <8:6.8.9.9-7ubuntu7**

Do your applications use this vulnerable package?

Overview

The (1) EPHEMERAL, (2) HTTPS, (3) MVG, (4) MSL, (5) TEXT, (6) SHOW, (7) WIN, and (8) PLT coders in ImageMagick before 6.9.3-10 and 7.x before 7.0.1-1 allow remote attackers to execute arbitrary code via shell metacharacters in a crafted image, aka "ImageTragick."

References

- ADVISORY
- BID
- BUGTRAQ
- BUGTRAQ
- CERT-VN
- CONFIRM
- CONFIRM
- CONFIRM

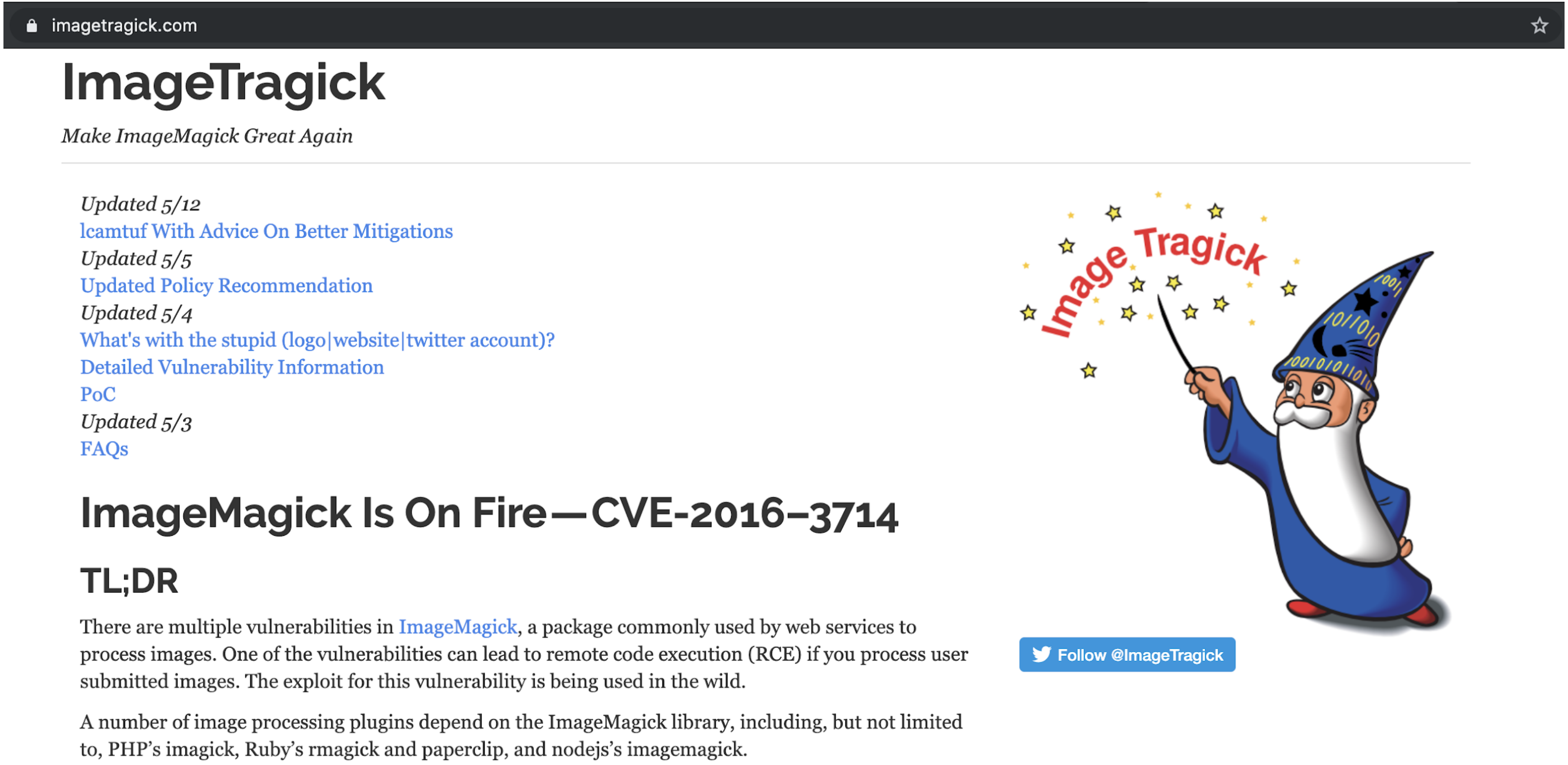
CVSS SCORE

8.4

HIGH SEVERITY

ATTACK VECTOR	ATTACK COMPLEXITY
Local	Low
PRIVILEGES REQUIRED	**USER INTERACTION**
None	None
SCOPE	**CONFIDENTIALITY**
Unchanged	High
INTEGRITY	**AVAILABILITY**
High	High

CVSS:3.0/AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H



ImageTragick

Make ImageMagick Great Again

Updated 5/12
lcamtuf With Advice On Better Mitigations
Updated 5/5
Updated Policy Recommendation
Updated 5/4
What's with the stupid (logo|website|twitter account)?
Detailed Vulnerability Information
PoC
Updated 5/3
FAQs

ImageMagick Is On Fire—CVE-2016–3714

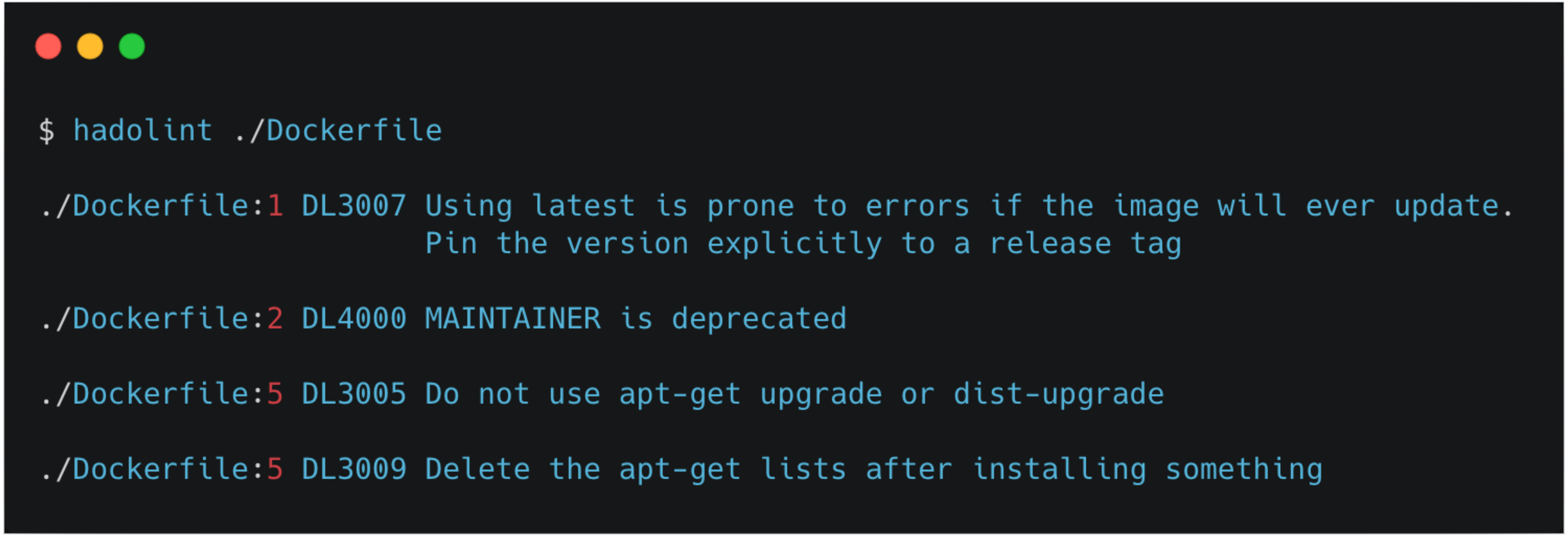
TL;DR

There are multiple vulnerabilities in ImageMagick, a package commonly used by web services to process images. One of the vulnerabilities can lead to remote code execution (RCE) if you process user submitted images. The exploit for this vulnerability is being used in the wild.

A number of image processing plugins depend on the ImageMagick library, including, but not limited to, PHP's imagick, Ruby's rmagick and paperclip, and nodejs's imagemagick.

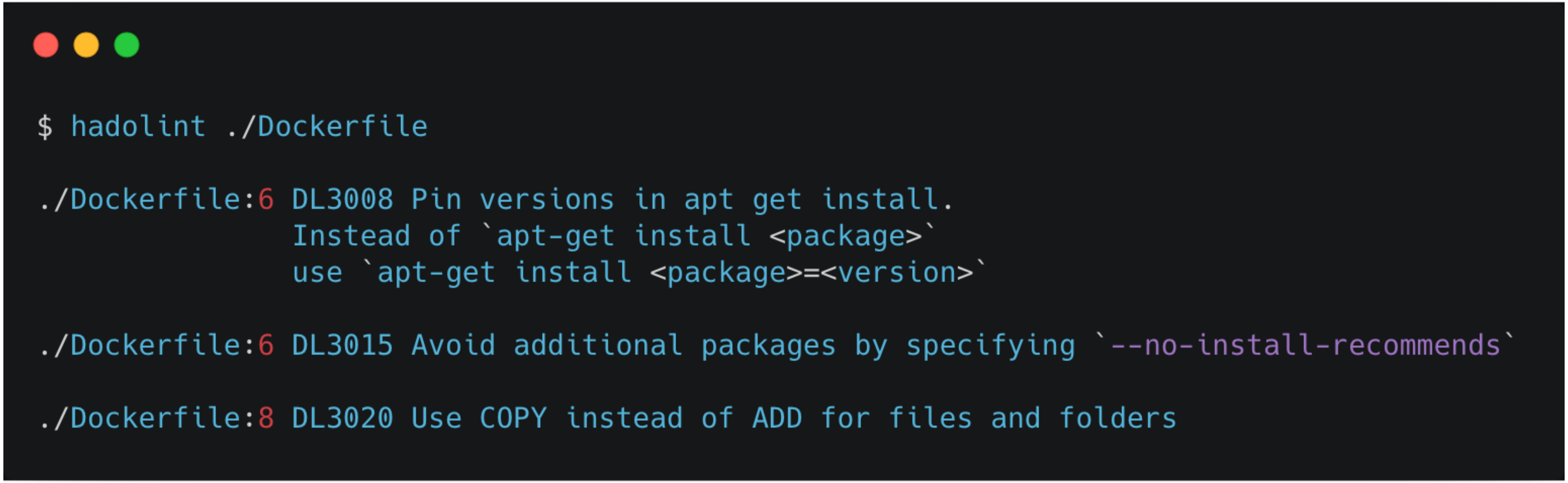
# Use a linter

hadolint



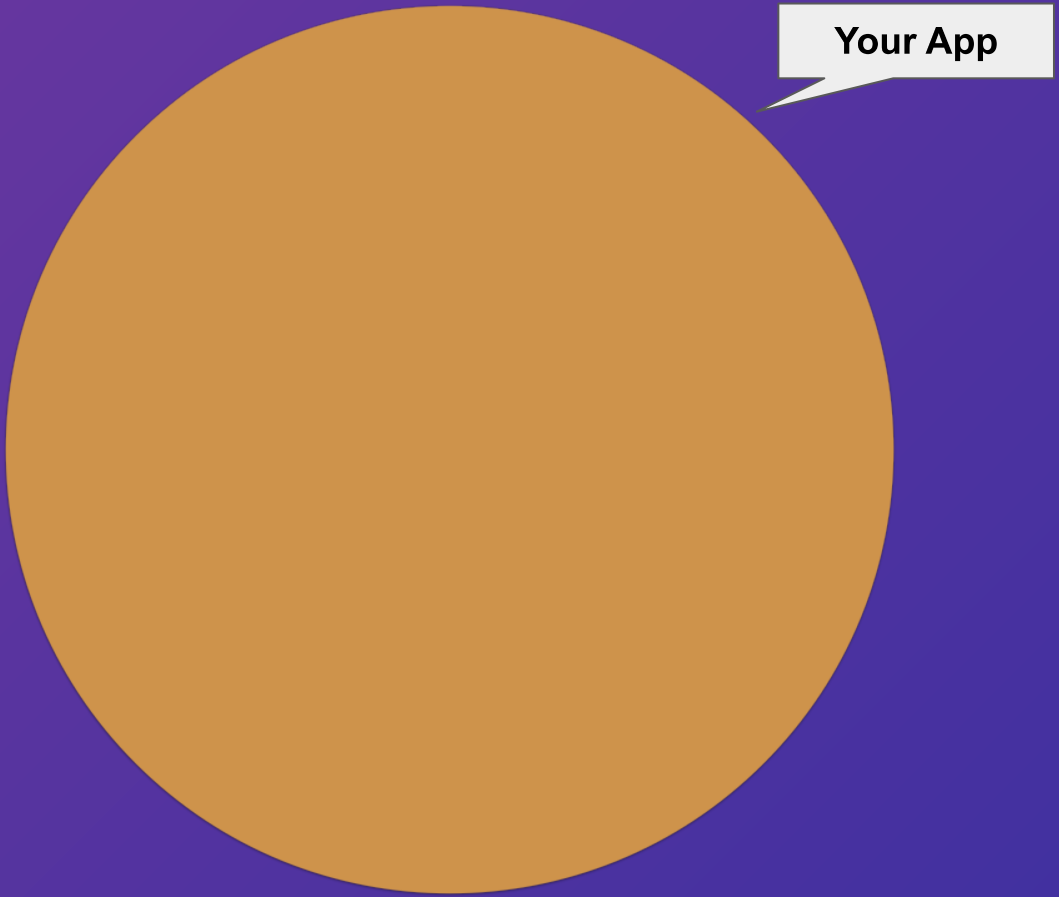
```
$ hadolint ./Dockerfile

./Dockerfile:1 DL3007 Using latest is prone to errors if the image will ever update.
                      Pin the version explicitly to a release tag

./Dockerfile:2 DL4000 MAINTAINER is deprecated

./Dockerfile:5 DL3005 Do not use apt-get upgrade or dist-upgrade

./Dockerfile:5 DL3009 Delete the apt-get lists after installing something
```

hadolint

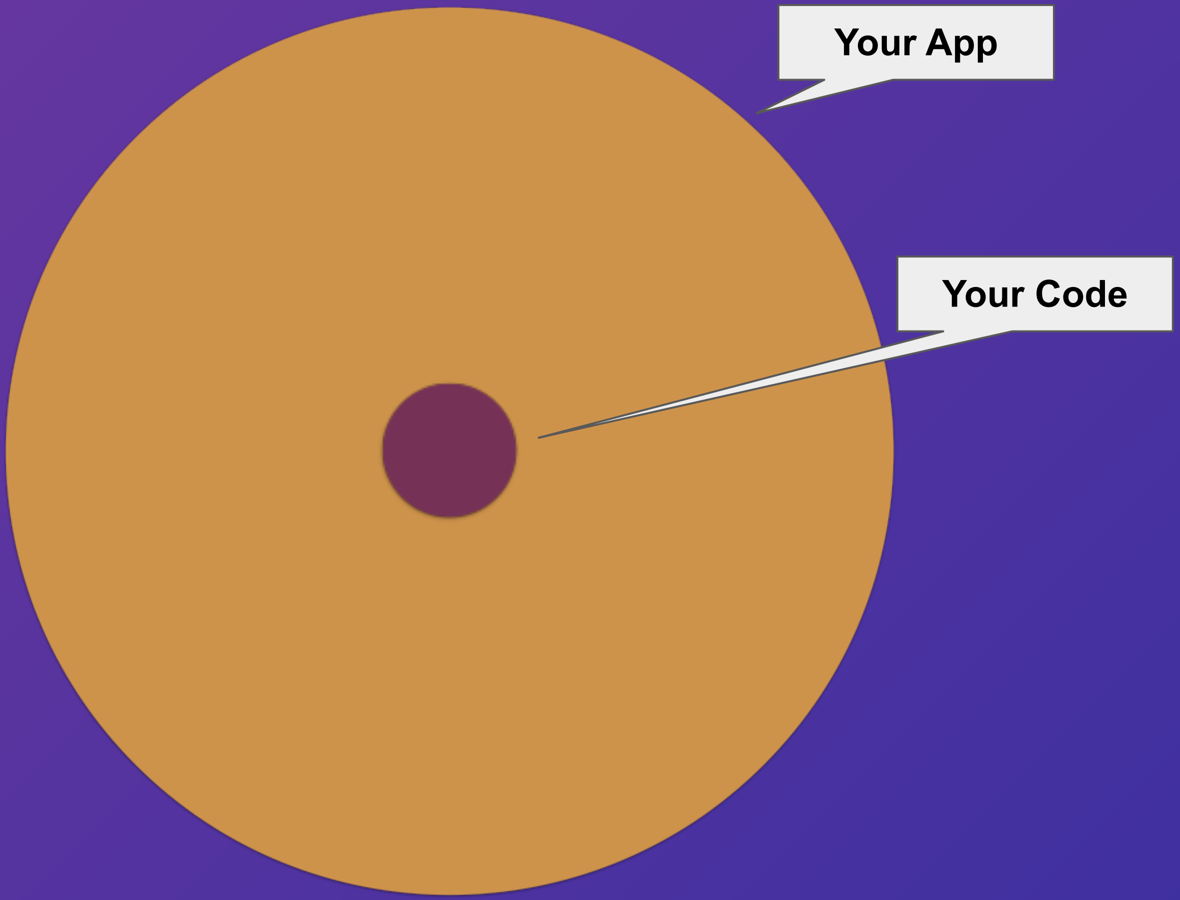


```
$ hadolint ./Dockerfile

./Dockerfile:6 DL3008 Pin versions in apt get install.
               Instead of `apt-get install <package>`
               use `apt-get install <package>=<version>`

./Dockerfile:6 DL3015 Avoid additional packages by specifying `--no-install-recommends`

./Dockerfile:8 DL3020 Use COPY instead of ADD for files and folders
```

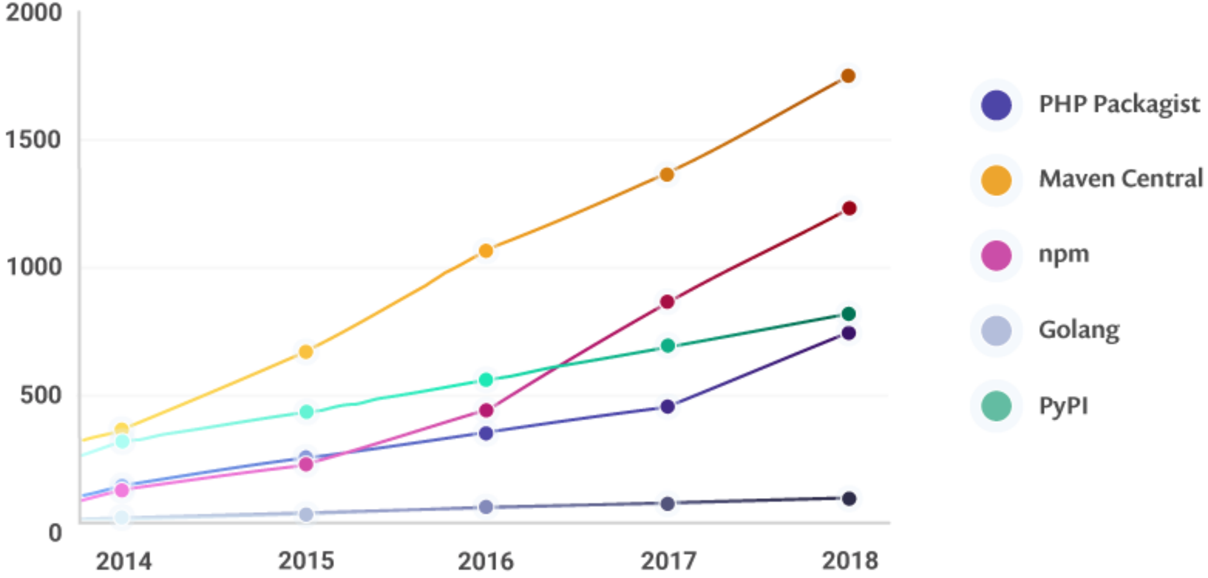
# application dependencies impact container security too



Your App

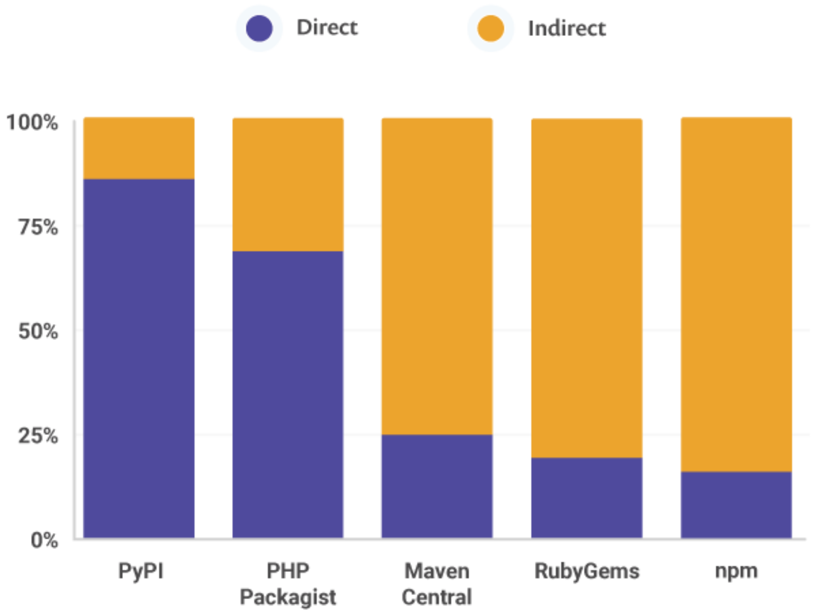

Your App
Your Code

New vulnerabilities each year by ecosystem



source: https://snyk.io/opensourcesecurity-2019

The direct and indirect dependency split across ecosystems



source: https://snyk.io/opensourcesecurity-2019

What can possibly go wrong with vulnerabilities in my app?

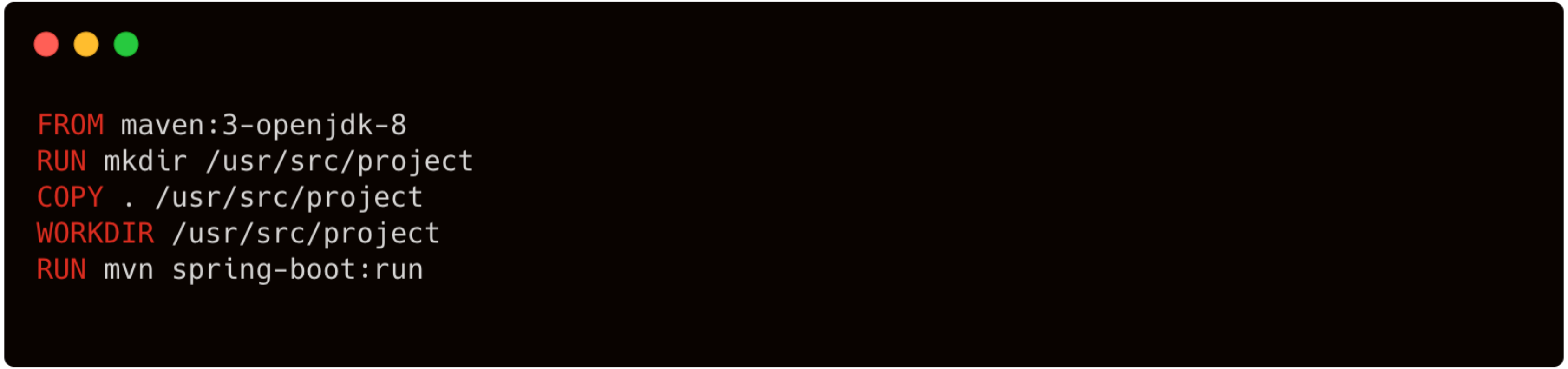
# Multi-stage builds

build image

compile and setup your app

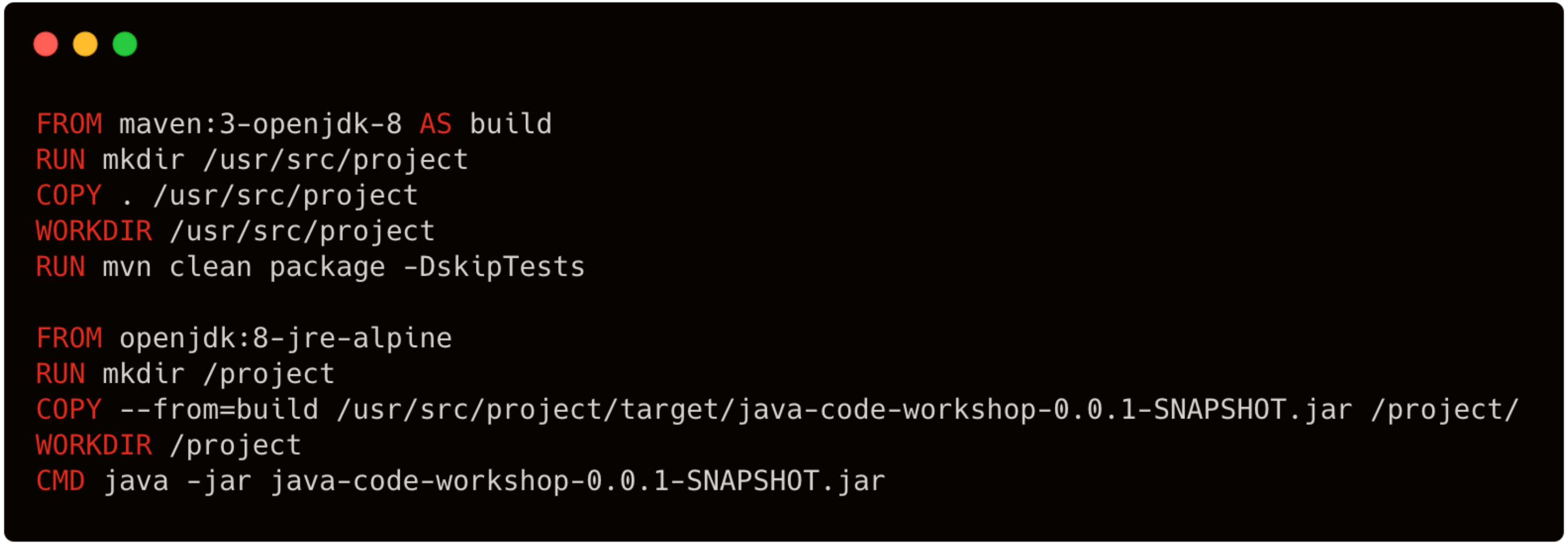
prod image

production artifacts



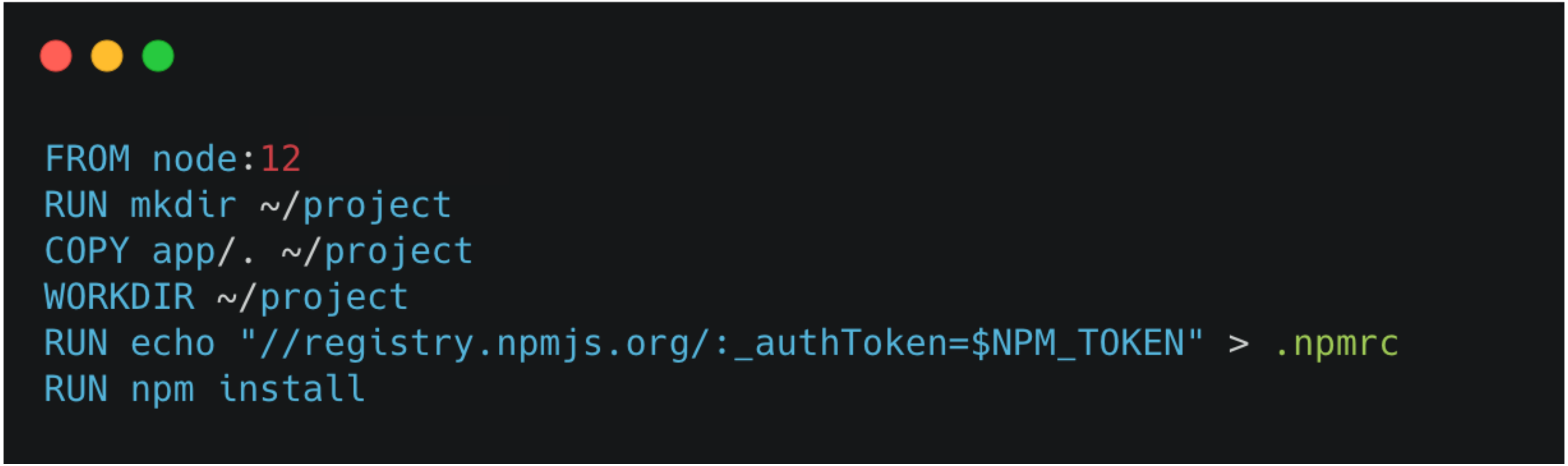
```
FROM maven:3-openjdk-8
RUN mkdir /usr/src/project
COPY . /usr/src/project
WORKDIR /usr/src/project
RUN mvn spring-boot:run
```

631 MB

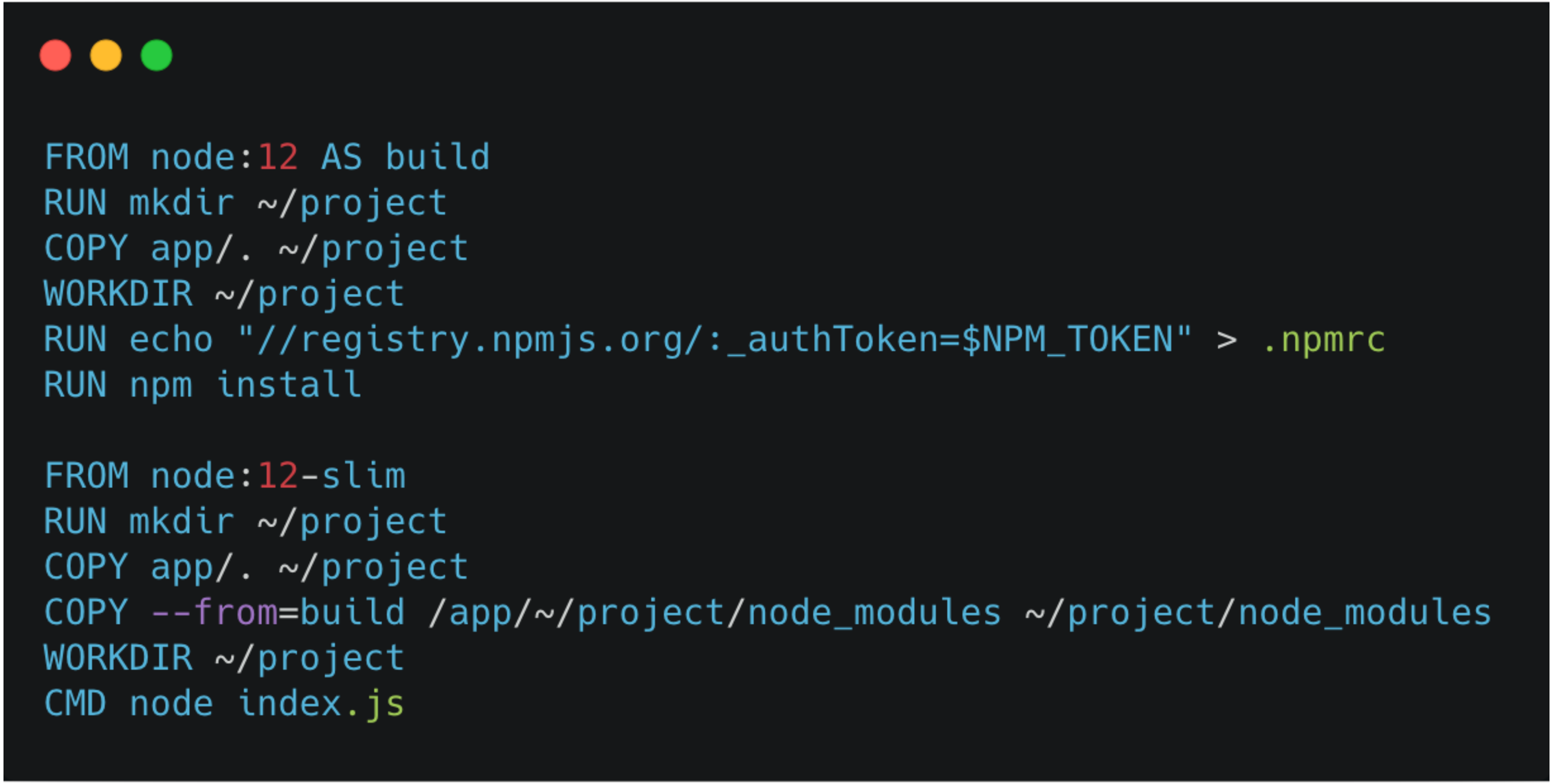


```
FROM maven:3-openjdk-8 AS build
RUN mkdir /usr/src/project
COPY . /usr/src/project
WORKDIR /usr/src/project
RUN mvn clean package -DskipTests

FROM openjdk:8-jre-alpine
RUN mkdir /project
COPY --from=build /usr/src/project/target/java-code-workshop-0.0.1-SNAPSHOT.jar /project/
WORKDIR /project
CMD java -jar java-code-workshop-0.0.1-SNAPSHOT.jar
```

132 MB



```
FROM node:12
RUN mkdir ~/project
COPY app/. ~/project
WORKDIR ~/project
RUN echo "//registry.npmjs.org/:_authToken=$NPM_TOKEN" > .npmrc
RUN npm install
```



```
FROM node:12 AS build
RUN mkdir ~/project
COPY app/. ~/project
WORKDIR ~/project
RUN echo "//registry.npmjs.org/:_authToken=$NPM_TOKEN" > .npmrc
RUN npm install

FROM node:12-slim
RUN mkdir ~/project
COPY app/. ~/project
COPY --from=build /app/~/project/node_modules ~/project/node_modules
WORKDIR ~/project
CMD node index.js
```

Attackers are targeting open source

one vulnerability = many victims

- ✓ Choose the right base image
- ✓ Re-build images often
- ✓ Scan docker images during devel’
- ✓ Use multi-stage docker builds
- ✓ Use a security linter for a Dockerfile
- ✓ Don’t run your container as root

Containers are Cool

Be a **Responsible** Cool Kid

@BrianVerm

Use Snyk for free

https://snyk.io