https://www.linkedin.com/in/akshayithape/

Docker Image Security Best Practices

By Akshay Ithape - DevOps Engineer

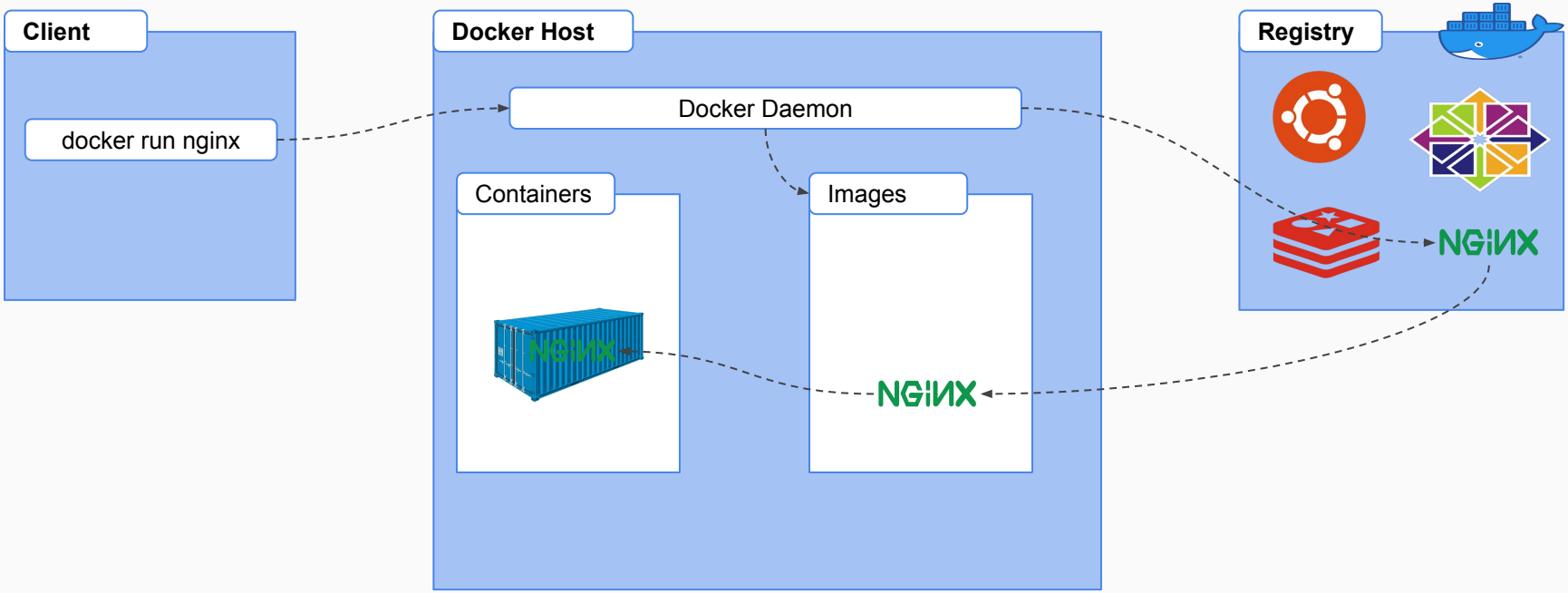
https://www.linkedin.com/in/akshayithape/

$whoami

- Myself Akshay Ithape.
- Currently, I'm Working as a DevOps Engineer At Eastern Enterprise.
- I'm a passionate about Linux and Cloud.
- As a Linux Person, I believes in Open Source so I like to share my knowledge with community.
- I'm RedHat, AWS, Terraform and Kubernetes Certified Engineer.
- Written articles For "Open Source For You" Magazine.

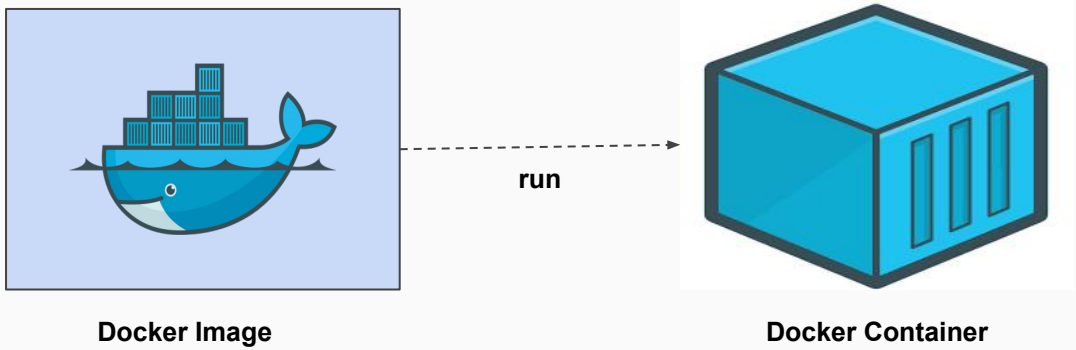
https://www.linkedin.com/in/akshayithape/

How Docker Works ?



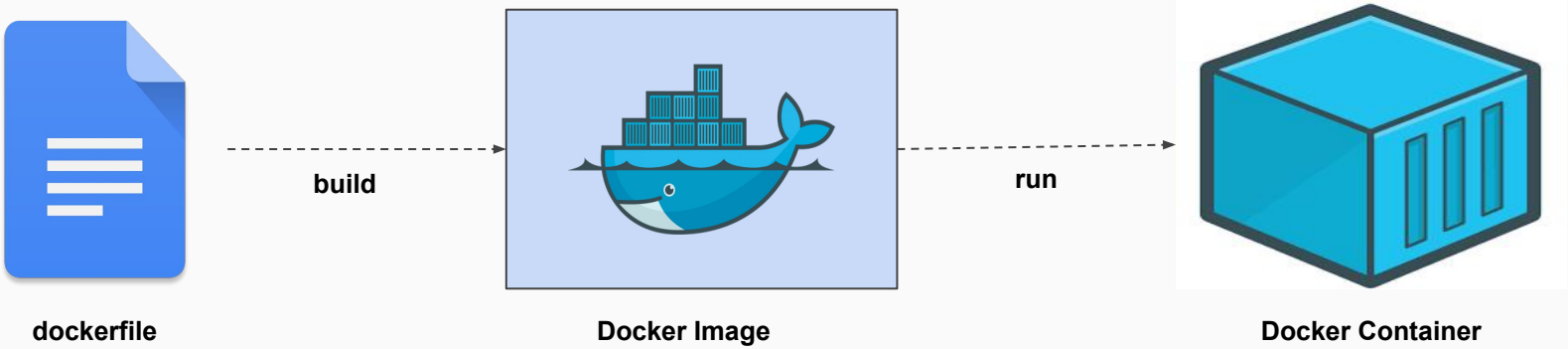
https://www.linkedin.com/in/akshayithape/

Why Docker Image is Required ?



https://www.linkedin.com/in/akshayithape/

How To Build Custom Docker Image ?



https://www.linkedin.com/in/akshayithape/

Simple Dockerfile Context

dockerfile-format

```
# Comment
INSTRUCTION arguments
```

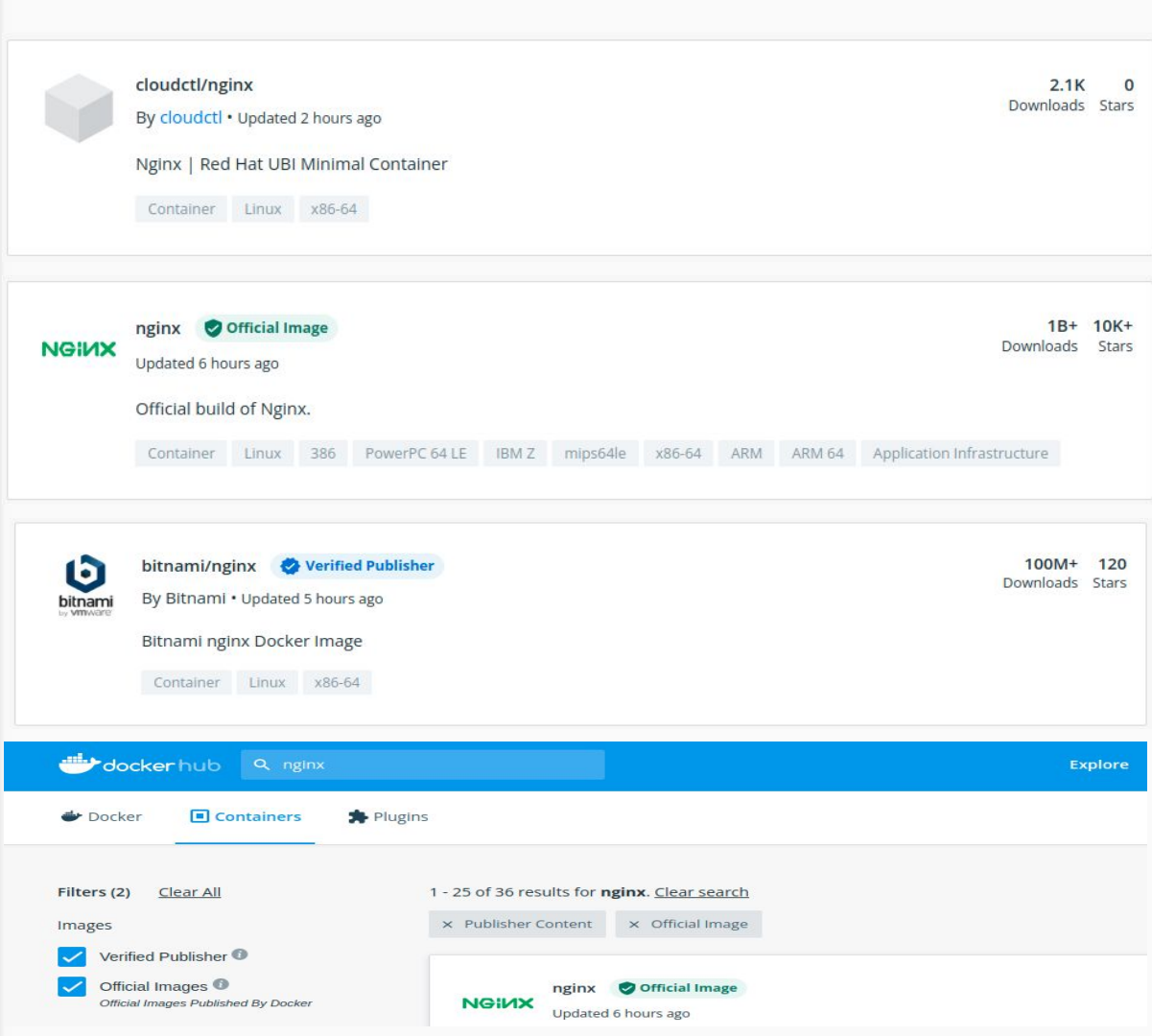
dockerfile

```
FROM nginx:latest
RUN apt-get update -y
WORKDIR /usr/share/nginx/html
COPY index.html .
CMD ["nginx", "-g", "daemon off;"]
EXPOSE 80
```

https://www.linkedin.com/in/akshayithape/

Choose Official or Verified Base Image

Always choose Official or Verified base image. If is not official or verified then vulnerability chances are very high.



https://www.linkedin.com/in/akshayithape/

Prefer minimal base images

- Choose images with fewer OS libraries and tools lower the risk and attack surface of the containers.
- Prefer alpine-based images over full-blown system OS images.

TAG

latest Log4Shell CVE not detected

Last pushed a day ago by doijanky

docker pull nginx:latest

DIGEST	OS/ARCH	COMPRESSED SIZE
18100eaa5648	linux/386	55.94 MB
1a763cbd30ef	linux/amd64	54.1 MB
bcba09dc6c63	linux/arm/v5	50.97 MB

TAG

alpine Log4Shell CVE not detected

Last pushed a day ago by doijanky

docker pull nginx:alpine

DIGEST	OS/ARCH	COMPRESSED SIZE
2cb22820489f	linux/386	10.13 MB
1e3458b88413	linux/amd64	9.69 MB
efdeee619e0f	linux/arm/v6	9.22 MB

https://www.linkedin.com/in/akshayithape/

Vulnerabilities Scan Reports

```
Package manager:   deb
Project name:      docker-image|nginx
Docker image:      nginx:latest
Platform:          linux/amd64
Base image:        nginx:1.21.6

Tested 143 dependencies for known vulnerabilities, found 84 vulnerabilities.

According to our scan, you are currently using the most secure version of the selected base image

For more free scans that keep your images secure, sign up to Snyk at https://dockr.ly/3ePqVcp
```

```
Package manager:   apk
Project name:      docker-image|nginx
Docker image:      nginx:alpine
Platform:          linux/amd64
Base image:        nginx:1.21.6-alpine

Tested 43 dependencies for known vulnerabilities, found 1 vulnerability.
```

https://www.linkedin.com/in/akshayithape/

Least privileged user

Create a dedicated user and group on the image, with minimal permissions to run the application; use the same user to run this process.

main docker-node / 17 / alpine3.15 / Dockerfile Go to file

nschonni feat: Node.js 17.8.0 Latest commit 5e2a48c 2 days ago History

3 contributors

99 lines (93 sloc) | 3.88 KB Raw Blame

```
FROM alpine:3.15

ENV NODE_VERSION 17.8.0

RUN addgroup -g 1000 node \
    && adduser -u 1000 -G node -s /bin/sh -D node \
    && apk add --no-cache \
        libstdc++ \
    && apk add --no-cache --virtual .build-deps \
        curl \
    && ARCH= && alpineArch="$(apk --print-arch)" \
      && case "${alpineArch##*-}" in \
        x86_64) \
          ARCH='x64' \
          CHECKSUM="faf0dd5ceb4b273b2f50af442cf461cc9d16f9b3ca7ef1006bb8ec1109e8c1cb" \
          ;; \
        *) ;; \
      esac \
```

dockerfile

```
FROM node:alpine

COPY package.json .
RUN npm install
COPY . .
EXPOSE 3000
USER node
CMD ["node", "index.js"]
```

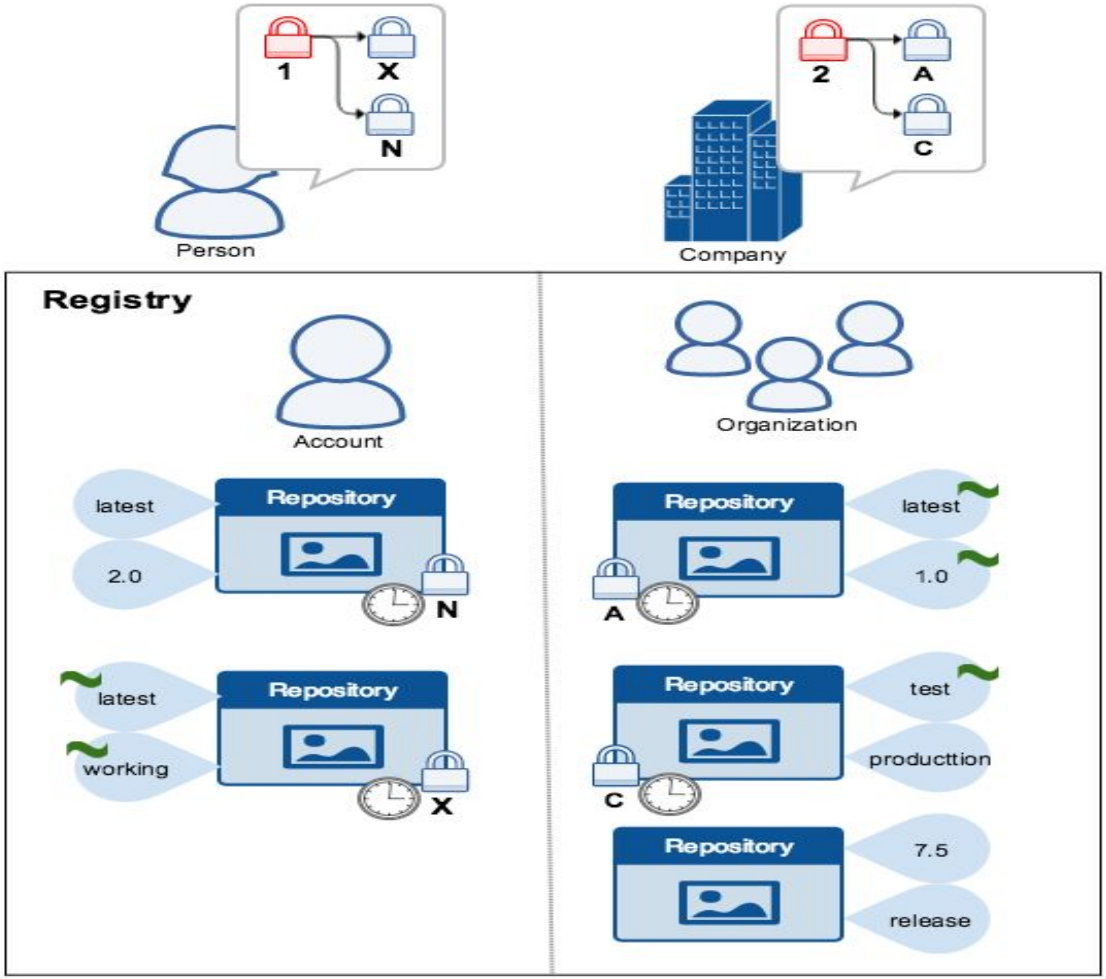
https://www.linkedin.com/in/akshayithape/

Verify and Sign images to mitigate MITM attacks

We put a lot of trust into docker images. It is critical to make sure the image we're pulling is the one pushed by the publisher, and that no one has tampered with it.

```
❯ export DOCKER_CONTENT_TRUST=1

❯ docker pull vigneshkumar73/vicky_nginx
Using default tag: latest
Error: remote trust data does not exist for docker.io/vigneshkumar73/vicky_nginx: notary.docker.io does not have trust data for docker.io/vigneshkumar73/vicky_nginx
```

```
~/docker-test
❯ docker trust key generate dev1
Generating key for dev1...
Enter passphrase for new dev1 key with ID 1ebd66c:
Repeat passphrase for new dev1 key with ID 1ebd66c:
Successfully generated and loaded private key. Corresponding public key available: /home/akshay/docker-test/dev1.pub

~/docker-test
❯ docker tag node-test:v1 imperishableakki/node-test:v1

~/docker-test
❯ docker trust signer add --key dev1.pub dev1 imperishableakki/node-test
Adding signer "dev1" to imperishableakki/node-test...
Initializing signed repository for imperishableakki/node-test...
You are about to create a new root signing key passphrase. This passphrase
will be used to protect the most sensitive key in your signing system. Please
choose a long, complex passphrase and be careful to keep the password and the
key file itself secure and backed up. It is highly recommended that you use a
password manager to generate the passphrase and keep it safe. There will be no
way to recover this key. You can find the key in your config directory.
Enter passphrase for new root key with ID 18b7bea:
Repeat passphrase for new root key with ID 18b7bea:
Enter passphrase for new repository key with ID aeaa583:
Repeat passphrase for new repository key with ID aeaa583:
Successfully initialized "imperishableakki/node-test"
Successfully added signer: dev1 to imperishableakki/node-test

~/docker-test
❯ docker trust sign imperishableakki/node-test:v1
Signing and pushing trust data for local image imperishableakki/node-test:v1, may overwrite remote trust data
The push refers to repository [docker.io/imperishableakki/node-test]
45f534bae6c2: Pushed
e5623c90b52f: Pushed
80f4d40e1c68: Pushed
0ad6919e1cc3: Mounted from library/node
5a09a182660a: Mounted from library/node
41c27a423d25: Mounted from library/node
ff768a1413ba: Mounted from library/node
v1: digest: sha256:d7e67a0b2cfab1476adbe69c3db2927f0b644ca56e7593111d9bad8d78a14d2d size: 1780
Signing and pushing trust metadata
Enter passphrase for dev1 key with ID 1ebd66c:
Successfully signed docker.io/imperishableakki/node-test:v1
```



A offline key is used to create tagging keys. Offline keys belong to a person or an organization. Resides client-side. You should store these in a safe place and back them up.

A tagging key is associated with an image repository. Creators with this key can push or pull any tag in this repository. This resides on client-side.

A timestamp key is associated with an image repository. This is created by Docker and resides on the server.

Signed tag.

https://www.linkedin.com/in/akshayithape/

Find the docker image vulnerabilities

Scan your docker images for known vulnerabilities and integrate it as part of your continuous integration. There are many open sources available to scan images.

aquasecurity/
docker-bench

Checks whether Docker is deployed according to security best practices as defined in the CIS Docker Benchmark

12 Contributors | 6 Issues | 167 Stars | 58 Forks

aqua trivy

snyk

clair

A Container Image Security Analyzer by CoreOS

& Many More!

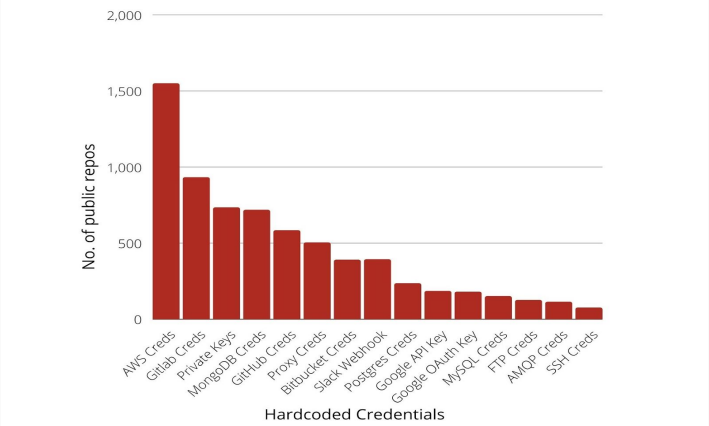
https://www.linkedin.com/in/akshayithape/

Trivy Demo

https://www.linkedin.com/in/akshayithape/

Don’t leak sensitive information to docker images

It’s easy to accidentally leak secrets, tokens, and keys into images when building them.



Top 5 Exposures in Docker Images

- **Hardcoded secrets**
- **Sensitive config files**
- **Adding the entire git repo**
- **Paid software licenses**
- **Default credentials**

46076
Docker Containers

Leaked at least one Hardcoded Secret or Config file

Exposures in Docker Containers

15,541
Hardcoded Secrets were identified across 10,181 Repositories

57,589
Potentially Sensitive Config files copied to Docker Images across 36,176 Repos

https://redhuntlabs.com/blog/scanning-millions-of-publicly-exposed-docker-containers-thousands-of-secrets-leaked.html

How to proactively stop exposures in docker images?

- Don’t hardcode tokens/API keys in docker images
- Do not clone/download the required files using credentials. Instead copy them to the image.
- Used .dockerignore file
- Multi-Stage build
- Used container private registry.

File Edit Selection View Go Run Terminal Help

EXPLORER

DOCKER-TEST

- .dockerignore
- .env
- dev1.pub
- dockerfile
- index.js
- package.json
- private.key

OUTLINE

dockerfile

```
FROM node:alpine

RUN mkdir /app
WORKDIR /app
COPY package.json .
RUN npm install
COPY . .
EXPOSE 3000
USER node
CMD ["node", "index.js"]
```

.dockerignore

```
.env
private.key
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

~/docker-test at 09:10:11 am

```
❯ docker build -f dockerfile -t node-test:v3 .
Sending build context to Docker daemon  6.144kB
Step 1/9 : FROM node:alpine
 ---> 15c6163d954c
Step 2/9 : RUN mkdir /app
 ---> Using cache
 ---> 2366349bdb11
Step 3/9 : WORKDIR /app
 ---> Using cache
 ---> 4118da81c329
Step 4/9 : COPY package.json .
 ---> Using cache
 ---> 81c8480f487b
Step 5/9 : RUN npm install
 ---> Using cache
 ---> b4290cd49163
Step 6/9 : COPY . .
 ---> Using cache
 ---> 14e1829510c4
Step 7/9 : EXPOSE 3000
 ---> Using cache
 ---> 6887a76f2bda
Step 8/9 : USER node
 ---> Using cache
 ---> c4952af36f6e
Step 9/9 : CMD ["node", "index.js"]
 ---> Using cache
 ---> d8561ad14f9c
Successfully built d8561ad14f9c
Successfully tagged node-test:v3
```

~/docker-test at 09:10:14 am

```
❯ ./setup-run.sh
```

~/docker-test at 09:10:43 am

```
❯ docker run -it node-test:v3 /bin/sh
/app $ ls -la
total 32
drwxr-xr-x    1 root     root          4096 Mar 25 03:39 .
drwxr-xr-x    1 root     root          4096 Mar 25 03:40 ..
-rw-rw-r--    1 root     root            16 Mar 25 03:32 .dockerignore
-rw-------    1 root     root           190 Mar 24 17:07 dev1.pub
-rw-rw-r--    1 root     root           137 Mar 25 03:39 dockerfile
-rw-rw-r--    1 root     root           287 Mar 24 13:57 index.js
-rw-r--r--    1 root     root           245 Mar 25 03:39 package-lock.json
-rw-rw-r--    1 root     root           473 Mar 24 13:57 package.json
/app $
```

zsh
zsh
docker

https://www.linkedin.com/in/akshayithape/

Use fixed tags for immutability

Docker image owners can push new versions to the same tags, which may result in inconsistent images during builds, and makes it hard to track if a vulnerability has been fixed.

TAG
alpine3.15 Log4Shell CVE not detected
Last pushed a day ago by doijanky
docker pull node:alpine3.15

DIGEST	OS/ARCH	COMPRESSED SIZE
56d1a2b5d6d0	linux/amd64	48.64 MB
9ddb9e866d8a	linux/arm/v6	47.56 MB
bdfd93ea78e9	linux/arm/v7	46.9 MB

+3 more...

TAG
alpine3.14 Log4Shell CVE not detected
Last pushed a day ago by doijanky
docker pull node:alpine3.14

DIGEST	OS/ARCH	COMPRESSED SIZE
5d9f6ece78fb	linux/amd64	48.64 MB
927e1710e913	linux/arm/v6	47.55 MB
fb14e42ae506	linux/arm/v7	46.9 MB

+3 more...

TAG
alpine Log4Shell CVE not detected
Last pushed a day ago by doijanky
docker pull node:alpine

DIGEST	OS/ARCH	COMPRESSED SIZE
56d1a2b5d6d0	linux/amd64	48.64 MB
9ddb9e866d8a	linux/arm/v6	47.56 MB
bdfd93ea78e9	linux/arm/v7	46.9 MB

+3 more...

dockerfile

```
FROM node@sha256:5d9f6ece78fb5fa95be2875d74b6407f06f56b04ca6ecc517476f79dac8f1aaf

COPY package.json .
RUN npm install
COPY . .
EXPOSE 3000
USER node
CMD ["node", "index.js"]
```

Use COPY instead of ADD

Arbitrary URLs specified for ADD could result in MITM attacks, or sources of malicious data. In addition, ADD implicitly unpacks local archives which may not be expected and result in path traversal and Zip Slip vulnerabilities.

ADD instructions is more capable the COPY.

- It can handle remote URLs.
- It can auto-extract tar files.

As we using remote URls so chances are high for MITM attacks or malicious data.

Space and image layer considerations

When local archives are used, ADD automatically extracts them to the destination directory. While this may be acceptable, it adds the risk of zip bombs and Zip Slip vulnerabilities that could then be triggered automatically.

https://www.linkedin.com/in/akshayithape/

Use labels for metadata

Labels with metadata for images provide useful information for users.Include security details as well.

dockerfile

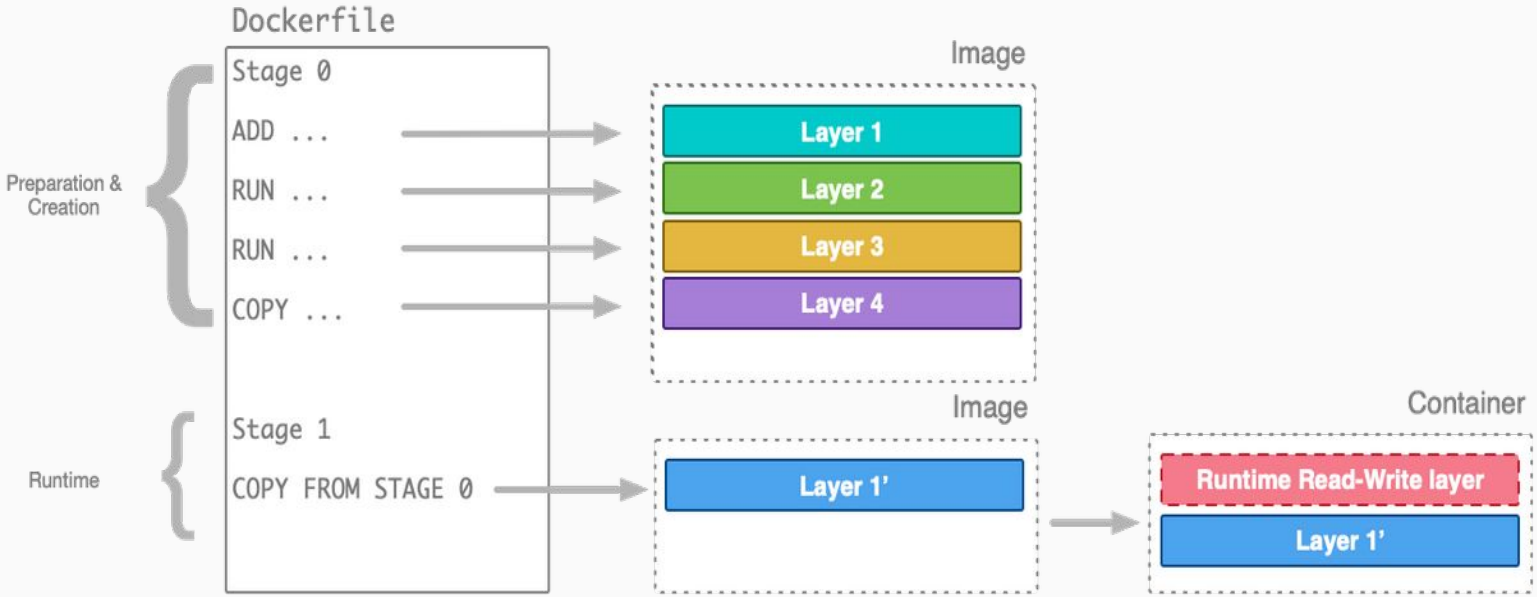
dockerfile > ...

```
FROM node:alpine3.14
LABEL maintainer="Akshay Ithape"
LABEL description="Node Apline Image 3.14 for Demo Application"

RUN mkdir /app
WORKDIR /app
COPY package.json .
RUN npm install
COPY . .
ADD https://cdn.zoom.us/prod/5.10.0.2450/zoom_amd64.deb .
EXPOSE 3000
USER node
CMD ["node", "index.js"]
```

https://www.linkedin.com/in/akshayithape/

Use multi-stage builds for small secure images

Use multi-stage builds in order to produce smaller and cleaner images, thus minimizing the attack surface for bundled docker image dependencies.



multi-stage-dockerfile

```dockerfile
FROM golang AS builder
LABEL maintainer="Akshay Ithape"

RUN mkdir /app
WORKDIR /app

COPY hello.go .
RUN go build hello.go

FROM scratch
COPY --from=builder /app/hello .

ENTRYPOINT [ "./hello" ]
```

hello.go

```go
package main

import "fmt"

func main () {
  fmt.Println("Hello, world!")
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
~/docker-test                                                    at 10:52:05 am
❯ docker run test-hello:v1
Hello, world!

~/docker-test                                                    at 10:52:11 am
❯ docker image ls test-hello:v1
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
test-hello   v1        77b8b8d90863   37 seconds ago   1.75MB

~/docker-test                                                    at 10:52:13 am
❯
```

zsh
zsh

Multi-Stage Build Example

https://www.linkedin.com/in/akshayithape/

https://www.linkedin.com/in/akshayithape/

Use a linter

Enforce Dockerfile best practices automatically by using a static code analysis tool such as **hadolint** linter, that will detect and alert for issues found in a Dockerfile.

```
DL4000 Specify a maintainer of the Dockerfile
DL3006 Always tag the version of an image explicitely.
1 FROM debian
SC1007 Remove space after = if trying to assign a value (for empty string, use var='' ... ).
SC2154 node_verson 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_verson"
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"]
```

Reference Link

- https://snyk.io/blog/10-docker-image-security-best-practices/
- https://www.docker.com/blog/docker-and-snyk-extend-partnership-to-docker-official-and-certified-images/
- https://www.docker.com/blog/improve-the-security-of-hub-container-images-with-automatic-vulnerability-scans/
- https://www.docker.com/blog/advanced-dockerfiles-faster-builds-and-smaller-images-using-buildkit-and-multistage-builds/
- https://betterprogramming.pub/docker-content-trust-security-digital-signatures-eeae9348140d
- https://snyk.io/plans/
- https://aquasecurity.github.io/trivy/v0.18.3/installation/

Thank You EveryOne

Be In Touch

Linkedin : https://www.linkedin.com/in/akshayithape/

Gmail : ithapeakshay.02@gmail.com

GitHub : https://github.com/akshayithape-devops

Medium : https://akshayithape.medium.com/