Secrets
ABAP
Apex
AzureResourceManager
C
C#
C++
CloudFormation
COBOL
CSS
Dart
Docker
Flex
Go
HTML
Java
JavaScript
JCL
Kotlin
Kubernetes
Objective C
PHP
PL/I
PL/SQL
Python
RPG
Ruby
Scala
Swift
Terraform
Text
TypeScript
T-SQL
VB.NET
VB6
XML

Docker static code analysis

Unique rules to find Vulnerabilities, Security Hotspots, and Code Smells in your DOCKER code

All rules 44 | Vulnerability 4 | Bug 4 | Security Hotspot 15 | Code Smell 21

Tags | Impact | Clean code attribute | Search by name...

Recursively copying context directories is security-sensitive
Security Hotspot

Using clear-text protocols is security-sensitive
Security Hotspot

Using weak hashing algorithms is security-sensitive
Security Hotspot

Malformed JSON in Exec form leads to unexpected behavior
Bug

Dockerfile should only have one ENTRYPOINT and CMD instruction
Bug

Access variable which is not available in the current scope
Bug

A space before the equal sign in key-value pair may lead to unintended behavior
Bug

Allowing downgrades to a clear-text protocol is security-sensitive
Security Hotspot

Allowing shell scripts execution during package installation is security-sensitive
Security Hotspot

Using host operating system namespaces is security-sensitive
Security Hotspot

Setting loose POSIX file permissions is security-sensitive
Security Hotspot

Recursively copying context directories is security-sensitive

Analyze your code

Intentionality - Complete | Security

Security Hotspot | Critical | dockerfile cwe

When building a Docker image from a Dockerfile, a context directory is used and sent to the Docker daemon before the actual build starts. This context directory usually contains the Dockerfile itself, along with all the files that will be necessary for the build to succeed. This generally includes:

- the source code of applications to set up in the container.
- configuration files for other software components.
- other necessary packages or components.

The `COPY` and `ADD` directives in the Dockerfiles are then used to actually copy content from the context directory to the image file system.

When `COPY` or `ADD` are used to recursively copy entire top-level directories or multiple items whose names are determined at build-time, unexpected files might get copied to the image filesystem. It could affect their confidentiality.

Ask Yourself Whether

- The copied files and directories might contain sensitive data that should be kept confidential.
- The context directory contains files and directories that have no functional purpose for the final container image.

There is a risk if you answered yes to any of those questions.

Keep in mind that the content of the context directory might change depending on the build environment and over time.

Recommended Secure Coding Practices

- Limit the usage of globbing in the `COPY` and `ADD` sources definition.
- Avoid copying the entire context directory to the image filesystem.
- Prefer providing an explicit list of files and directories that are required for the image to properly run.

Sensitive Code Example

Copying the complete context directory:

```
FROM ubuntu:22.04
# Sensitive
COPY . .
CMD /run.sh
```

Copying multiple files and directories whose names are expanded at build time:

```
FROM ubuntu:22.04
# Sensitive
COPY ./example* /
COPY ./run.sh /
CMD /run.sh
```

Compliant Solution

```
FROM ubuntu:22.04
COPY ./example1 /example1
COPY ./example2 /example2
COPY ./run.sh /
CMD /run.sh
```

See

- Dockerfile reference - COPY directive
- Dockerfile reference - ADD directive
- CWE - CWE-668 - Exposure of Resource to Wrong Sphere
- CWE - CWE-497 - Exposure of Sensitive System Information to an Unauthorized Control Sphere

Available In:

sonarlint | sonarcloud | sonarqube