






































-  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... 🔍

Permissions of sensitive mount points should be restrictive

 Vulnerability

Server certificates should be verified during SSL/TLS connections

 Vulnerability

Weak SSL/TLS protocols should not be used

 Vulnerability

Disabling builder sandboxes is security-sensitive

 Security Hotspot

Exposing administration services is security-sensitive

 Security Hotspot

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

Permissions of sensitive mount points should be restrictive

Analyze your code

Intentionality - Complete

Security 

 Vulnerability

 Critical



 dockerfile cwe

For mounts types `secret` and `ssh`, Dockerfile’s `RUN` instruction supports a `mode` option for setting permissions. If you set this mode so that any user of the operating system can access the mount, it is vulnerable to leaks.

Why is this an issue?

How can I fix it?

More Info

Code examples

Noncompliant code example

```
# Noncompliant
RUN --mount=type=secret,id=build_secret,mode=0777 ./installer.sh
```

Compliant solution

```
RUN --mount=type=secret,id=build_secret,mode=0700 ./installer.sh
```

How does this work?

In general, always follow the least privilege principle, and set the `others` bit to `0`. By default, if `mode` is not set, permissions are safe.

In case you made this change because you need to access secrets or agents as a low-privileged user, you can use the options `uid` and `gid` to provide access without having to resort to world-readable or writable permissions that might expose them to unintended parties.

Available In:

sonarlint



sonarcloud



sonarqube

