sonar RULES

Products

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

Docker offers a feature to mount files and directories for specific `RUN` instructions when building Docker images. This feature can be used to provide secrets to commands that are executed during the build without baking them into the image. Additionally, it can be used to access SSH agents during the build.

The `mode` option is an octal value that allows you to specify the permissions for a particular file or directory. By default, on Docker, when mounting a `secret`, it is set to `0400`.

For `ssh`, it is set by default to `0600`:

- The first digit `0` stands for special permissions (like setuid, setgid and sticky bit) and in this case means that no special permissions are set.
- The following `6` (4+2 in octal format) means that the `owner` has read (4) and write (2) permissions
- `00` means that the `group` and `others` have no permissions.

If the `others` bit is set to a value other than 0 at build-time, any other process can access it when the `RUN` command is executed: the secrets are vulnerable to supply chain attacks that aim to siphon secrets from containers.

What is the potential impact?

Unauthorized access

The unintended audience can exploit the leaked private key or equivalent to authenticate themselves as the legitimate owner, gaining unauthorized entry to systems, servers, or accounts that accept the key for authentication.

This unauthorized access opens the door for various malicious activities, including data breaches, unauthorized modifications, and misuse of sensitive information.

Available In:
sonarlint sonarcloud sonarqube