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...

Environment variables should not be unset on a different layer than they were set

Analyze your code

Consistency - Conventional | Maintainability

Code Smell | Major

Setting an environment variable using the `ENV` instruction creates a new layer in the Docker image. The variable is then persisted for all subsequent build stages and is also present in the resulting image. Calling `RUN unset <env-variable>` unsets the variable only for this particular layer, but it is still possible to dump the environment variable from the previous layer.

Why is this an issue? | How can I fix it? | More Info

If an environment variable is needed only during build, this variable should be set and unset in a single `RUN` instruction.

Code examples

Noncompliant code example

```
ENV $ADMIN_USER
RUN unset $ADMIN_USER
```

Compliant solution

```
RUN export ADMIN_USER="admin" \
    && ... \
    && unset ADMIN_USER
```

How does this work?

In this example, the visibility of `ADMIN_USER` is only limited to the single layer. However, it is still possible to extract the value from the image. The best solution is to use `ARG` instead of `ENV` or set and unset the variable in the same `RUN` instruction.

Available In:
sonarlint | sonarcloud | sonarqube

© 2008-2024 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE, and SONARCLOUD are trademarks of Sona
S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.

Sonar helps developers write Clean Code.
Privacy Policy | Cookie Policy