Docker + Snyk Log4Shell Remediation Cheat Sheet

01 Scan your container images

Run `docker scan` on your image to get results about your image, including the Log4j vulnerability.

- `docker scan image`
- `docker scan –file Dockerfile imagename:tag`

More: https://docs.docker.com/engine/scan/

02 Use an official and current JDK in your image

- Using a supported major JDK release makes it easier for your team to pick up supported fixes.
- Using the latest JDK revision means staying in sync with security fixes.
- An example of a popular and official base image is `openjdk`. The naming conventions make it easier to rely on a reliable latest version. For example: `openjdk:11`

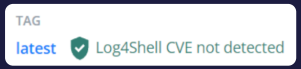
03 Upgrading your JDK isn't enough

While initial advice suggested a JDK upgrade could mitigate Log4Shell, it was later shown not to be effective against this vulnerability.

This includes setting `com.sun.jndi.ldap.object.trustURLCodebase` to `false`.

04 Identify scanned images in Docker Hub

- Docker now includes scan results in Docker Hub. This makes it easier for end-users to identify images that have been scanned for the Log4Shell CVE-2021-44228 and CVE-2021-45046, and if the vulnerabilities have been detected.
- New pushes of Docker Verified Publisher Image and Docker Official Image receive a special badge in their repo to signal the status of those images.



TAG

latest ✓ Log4Shell CVE not detected

05 Use Docker Desktop 4.3.1+ with `docker scan` 0.11.0+

- This combination of versions provides the support your team needs to identify the Log4j vulnerability in your image on Mac and Windows, and are readily available in Docker Desktop downloads.
- Linux users are supported with docker-ce. More details for Linux users are available at: https://www.docker.com/blog/apache-log4j-2-cve-2021-44228/

06 Don't run as root

- You should use a non-root user inside of your container to run operations. Frequently this means creating a user and group with some form of these commands to run your application or service:

```
RUN addgroup ... \
adduser ... \
chmod and chown
USER
```

- While the details are specific to your container, the results are
  - Create a user and group for your application
  - Specify on-disk permissions for only that user and group
  - Run your application as the named user (not root)

For more information, see point #1 in the 10 Kubernetes security context settings you should understand blog.

07 Use a `--read-only` root filesystem

- Attackers frequently rely on a writable filesystem to exploit your running container, so take that away from them.
- If your container provides API services only and does not persist any files, then chances are high you don't need any write permissions.
- This could be as easy as setting a mount to read-only and verifying your application continues to operate as expected. If you are running in a container, add the `--read-only` flag to your `docker run` command.
- Your next step will be to set read-only in your production environments using settings or your preferred orchestrator.
- For more information, watch Kubernetes Quick Hits: Use SecurityContext to run containers with a read-only filesystem or see point #7 in the 10 Kubernetes security context settings you should understand blog.

08 Upgrade your Log4j version to 2.17.1 or higher where possible

Upgrading to 2.17.1 rather than 2.15.0-rc2 will also provide a fix for CVE-2021-45046.

- **Automatic fix:** Connect Snyk to your Git repositories so it can raise pull requests to update your dependency graph where possible.
- **Manual fix:** If you are using Log4j as a direct dependency, you can upgrade your build file directly to 2.17.1 or higher.
- **Manual fix:** If you are using Log4j as a transitive dependency, identify a version of your direct dependency which pulls in the transitive Log4j dependency at 2.17.1 or higher.

09 Don't run privileged containers

- Don't run in privileged modem, which grants your running container all the rights and privileges available. Your application should not require these elevated privileges.
- For more information, watch Kubernetes Quick Hits: SecurityContext and why not to run as root or see point #5 in the 10 Kubernetes security context settings you should understand blog.

10 Minimize your container's footprint

- Lightweight containers are especially useful because the authors removed many packages to save time and to enhance security.
- When you remove unnecessary tooling such as `curl` and/or `wget`, you make it much harder for attackers to bring dangerous payloads inside your running containers.

Sign up for Snyk >