# Trivy - scan k8s image

Create trivy-k8s-scan.sh

|  |
| --- |
| *#!/bin/bash*  echo $imageName *#getting Image name from env variable*  docker run --rm -v $WORKSPACE:/root/.cache/ aquasec/trivy:0.17.2 -q image --exit-code 0 --severity LOW,MEDIUM,HIGH --light $imageName  docker run --rm -v $WORKSPACE:/root/.cache/ aquasec/trivy:0.17.2 -q image --exit-code 1 --severity CRITICAL --light $imageName  *# Trivy scan result processing*  exit\_code=$?  echo "Exit Code : $exit\_code"  *# Check scan results*  if [[ ${exit\_code} == 1 ]]; then  echo "Image scanning failed. Vulnerabilities found"  exit 1;  else  echo "Image scanning passed. No vulnerabilities found"  fi; |

Explain code:

docker run --rm -v $WORKSPACE:/root/.cache/ aquasec/trivy:0.17.2 -q image --exit-code 0 --severity LOW,MEDIUM,HIGH --light $imageName

This line runs the Trivy command in a Docker container. It uses `-v` option to mount the `$WORKSPACE` directory to the `/root/.cache/` directory inside the container for caching Trivy's database. The container is specified using the `aquasec/trivy:0.17.2` image with the `0.17.2` version. The Trivy scan is performed using the `image` option, and the `--exit-code` option is set to `0` to ensure that script does not exit if only low-severity vulnerabilities are found. The `--severity` option is set to `LOW,MEDIUM,HIGH` to include those severities in the report. The `--light` option indicates that the scan should use the lightweight database for faster scanning. `$imageName` variable is passed as a parameter to the Trivy command to scan the Docker image.

docker run --rm -v $WORKSPACE:/root/.cache/ aquasec/trivy:0.17.2 -q image --exit-code 1 --severity CRITICAL --light $imageName

Similar to the previous line, it runs the Trivy command in a Docker container to scan the Docker image, but it has a different severity criteria. This time, it scans for `CRITICAL` vulnerabilities, and sets the `--exit-code` option to `1`.

exit\_code=$?

This line captures the exit code of the last executed command, which is the Trivy command. It stores it in the `$exit\_code` variable.

echo "Exit Code : $exit\_code"

This line prints the exit code of the Trivy command to the console, for debugging purposes.

if [[ ${exit\_code} == 1 ]]; then

This line starts an if-else statement, where the exit code of the Trivy command is checked. If it is equal to `1`, it will execute the following lines.

echo "Image scanning failed. Vulnerabilities found"

exit 1;

These lines print a message that vulnerabilities have been found in the Docker image, and then exits the script with an exit code of `1`.

else

echo "Image scanning passed. No vulnerabilities found"

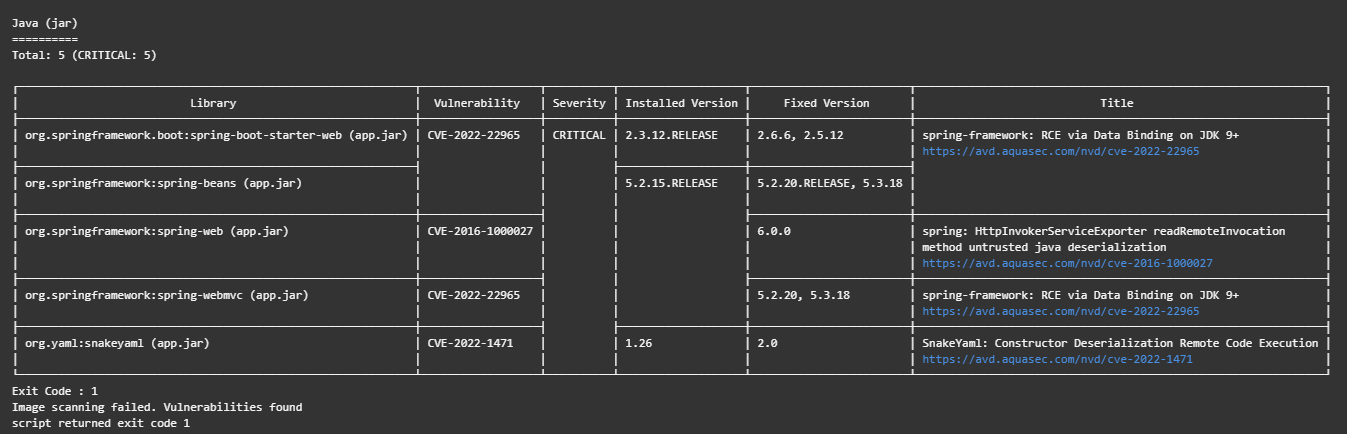
fi;

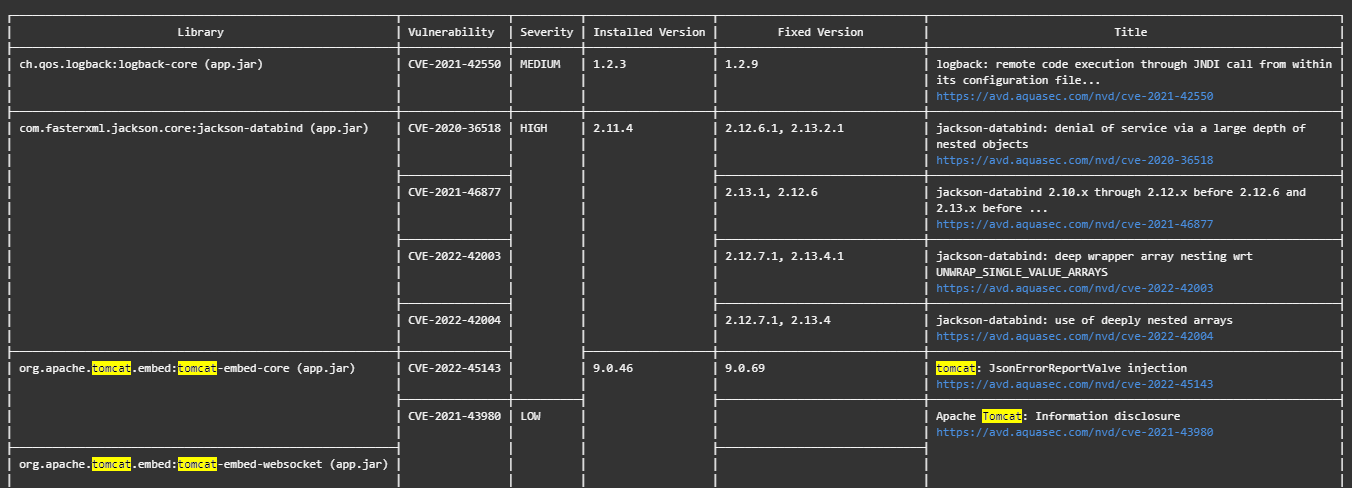
If the exit code of the Trivy command is not `1`, this block of code executes. It prints a success message indicating that there were no vulnerabilities found in the Docker image.

Add trivy scan into pipeline

|  |
| --- |
| stage('Vulnerability Scan - Kubernetes') {  steps {  parallel(  "OPA Scan": {  sh 'docker run --rm -v $(pwd):/project openpolicyagent/conftest test --policy opa-k8s-security.rego k8s\_deployment\_service.yaml'  },  "Kubesec Scan": {  sh "bash kubesec-scan.sh"  },  "Trivy Scan": {  sh "bash trivy-k8s-scan.sh"  }  )  }  } |

After build, we will fail in trivy scan with result like below.





We can fix CVE-2022-45143 by updating tomcat to version 9.0.69 and CVE-2022-22965 by updating spring-boot to version 2.6.6 or 2.5.12. And build again