Snyk with Jenkins:

- Create a synk account: <https://snyk.io/>

> Skip the Github integration

> Integrations > Serach Jenkins > Jenkins

Login to Jenkins:

Browse: localhost:8080

admin/admin

> Install the snyk security plugin

> Manage Jenkins > Plugins > Available plugins > Search snyk > Select and Install

> Configure Snyk

> Manage Jenkins > Tools > Snyk installations

> Add Snyk

Name: Snyk

Select Install automatically > version: latest

> Save

- Create a credential - to connect our snyk account with Jenkins

- Snyk API key: snyk.io > your username (at the bottom left) > Account Settings

> Manage Jenkins > Credentials > select global > Add credentials

kind: Synk API token

Token: ADD\_YOUR\_TOKEN\_HERE

ID:  SnykToken

Description:  SnykToken

> Create

- Create a new freestyle job in Jenkins

> Dashboard > New Item > Name: codescansnyk, Template: Freestyle project

> Source Code Management

> Git > Repository URL: <https://github.com/abhijithvg/simple-java-maven-app.git>

> Build steps

> Add build step > Invoke Snyk security task

> Select Snyk API Token

> Save & Build the job

- Go to snyk account > Projects

Snyk with Pipeline:

- Create a new Jenkins pipeline job

> New Item > Name: scapipeline, Template: Pipeline

Script:

pipeline {

    agent any

    tools {

        maven "mvn3.9.8"

        snyk "Snyk"

    }

    stages {

        stage('Build & Test Automation') {

            steps {

                git 'https://github.com/abhijithvg/SonarQubeCoverageJava.git'

                sh "mvn -Dmaven.test.failure.ignore=true clean package"

            }

            post {

                success {

                    junit '\*\*/target/surefire-reports/\*.xml'

                    archiveArtifacts 'target/\*.jar'

                }

            }

        }

        stage('SCA scan') {

            steps {

                snykSecurity snykInstallation: 'Snyk', snykTokenId: 'SnykToken'

            }

        }

    }

}

> Save

> Build Now

ZAP on Jenkins:

- Install owasp zap plugin in Jenkins

> Manage Jenkins > Plugins > Available Plugins > search for official owasp zap > Install

- Install custom tools plugin in Jenkins

> Manage Jenkins > Plugins > Available Plugins > search for custom tools > Install

Configure ZAP tool:

> Manage Jenkins > Tools > Custom tool installations > Add custom tool

Name: ZAP

Install Automatically: selected

Download URL for binary archive: https://github.com/zaproxy/zaproxy/releases/download/v2.15.0/ZAP\_2.15.0\_Linux.tar.gz

Subdirectory of extracted archive: ZAP\_2.15.0

> Save

Configure System:

> Manage Jenkins > System > ZAP

Ensure -> default Host: localhost , default  Port: 8090

Create a Jenkins job:

> Dashboard > New Item > Name: zapdast, Template: Freestyle project

> Build steps > Add build step > Execute ZAP

ZAP Home Directory: /var/lib/jenkins/za\_proxy

Session Management: Select "Persist Session"; Filename: zap\_demo

Session Properties:

Context Name: demo\_testfile

Include in context: https://demo.testfire.net/

Exclude in context: ^(?:(?!https:\/\/demo.testfire.net/).\*).$

Attack Mode:

Starting Point: https://demo.testfire.net/

Select: Spider Scan, Recurse, AJAX Spider & Active Scan

Finalize Run:

Select: Generate Reports, Clean Workspace Reports, Generate Reports & select HTML as format

> Post build actions > add post-build action > Archive the artifacts

Files to archive: reports/\*

> Save & Build

ZAP DAST Scan in Jenkins Pipeline:

> Install Docker

$ sudo apt update

$ sudo apt install -y containerd docker.io

$ sudo chmod 777 /var/run/docker.sock

$ sudo service docker restart

Verify Docker Installation:

$ docker version

> New Pipeline Job

Name: dastpipeline, Template: Pipeline

Pipeline:

def scan\_type

def target

 pipeline {

    agent any

environment {

    zapDockerName = "ghcr.io/zaproxy/zaproxy:stable"

}

    parameters {

        choice  choices: ['Baseline', 'APIS', 'Full'],

                 description: 'Type of scan that is going to perform inside the container',

                 name: 'SCAN\_TYPE'

        string defaultValue: 'https://medium.com/',

                 description: 'Target URL to scan',

                 name: 'TARGET'

        booleanParam defaultValue: true,

                 description: 'Parameter to know if wanna generate report.',

                 name: 'GENERATE\_REPORT'

    }

    stages {

        stage('Parameter Initialization') {

            steps {

                script {

                    echo """

                         The current parameters are:

                             Scan Type: ${params.SCAN\_TYPE}

                             Target: ${params.TARGET}

                             Generate report: ${params.GENERATE\_REPORT}

                         """

                }

            }

        }

        stage('Setting up OWASP ZAP docker container') {

            steps {

                echo 'Pulling up last OWASP ZAP container --> Start'

                sh 'docker pull ${zapDockerName}'

                echo 'Pulling up last VMS container --> End'

                echo 'Starting container --> Start'

                sh 'docker run -dt --name owasp ${zapDockerName} /bin/bash'

            }

        }

        stage('Prepare wrk directory') {

            when {

                environment name : 'GENERATE\_REPORT', value: 'true'

            }

            steps {

                script {

                    sh '''

                             docker exec owasp \

                             mkdir /zap/wrk

                         '''

                }

            }

        }

        stage('Scanning target on owasp container') {

            steps {

                script {

                    scan\_type = "${params.SCAN\_TYPE}"

                    echo "----> scan\_type: $scan\_type"

                    target = "${params.TARGET}"

                    if (scan\_type == 'Baseline') {

                        sh """

                             docker exec owasp \

                             zap-baseline.py \

                             -t $target \

                             -r report.html \

                             -I

                         """

                    }

                     else if (scan\_type == 'APIS') {

                        sh """

                             docker exec owasp \

                             zap-api-scan.py \

                             -t $target \

                             -r report.html \

                             -I

                         """

                     }

                     else if (scan\_type == 'Full') {

                        sh """

                             docker exec owasp \

                             zap-full-scan.py \

                             -t $target \

                             -r report.html \

                             -I

                         """

                     }

                     else {

                        echo 'Something went wrong...'

                     }

                }

            }

        }

        stage('Copy Report to Workspace') {

            steps {

                script {

                    sh '''

                         docker cp owasp:/zap/wrk/report.html ${WORKSPACE}/report.html

                     '''

                }

            }

        }

    }

    post {

        always {

            echo 'Removing container'

            sh '''

                     docker stop owasp

                     docker rm owasp

                 '''

archiveArtifacts 'target/\*.jar'

            cleanWs()

        }

    }

 }

> Save & Build

Corrected pipeline:

def scan\_type

def target

 pipeline {

    agent any

environment {

    zapDockerName = "ghcr.io/zaproxy/zaproxy:stable"

}

    parameters {

        choice  choices: ['Baseline', 'APIS', 'Full'],

                 description: 'Type of scan that is going to perform inside the container',

                 name: 'SCAN\_TYPE'

        string defaultValue: 'https://medium.com/',

                 description: 'Target URL to scan',

                 name: 'TARGET'

        booleanParam defaultValue: true,

                 description: 'Parameter to know if wanna generate report.',

                 name: 'GENERATE\_REPORT'

    }

    stages {

        stage('Parameter Initialization') {

            steps {

                script {

                    echo """

                         The current parameters are:

                             Scan Type: ${params.SCAN\_TYPE}

                             Target: ${params.TARGET}

                             Generate report: ${params.GENERATE\_REPORT}

                         """

                }

            }

        }

        stage('Setting up OWASP ZAP docker container') {

            steps {

                echo 'Pulling up last OWASP ZAP container --> Start'

                sh 'docker pull ${zapDockerName}'

                echo 'Pulling up last VMS container --> End'

                echo 'Starting container --> Start'

                sh 'docker run -dt --name owasp ${zapDockerName} /bin/bash'

            }

        }

        stage('Prepare wrk directory') {

            when {

                environment name : 'GENERATE\_REPORT', value: 'true'

            }

            steps {

                script {

                    sh '''

                             docker exec owasp \

                             mkdir /zap/wrk

                         '''

                }

            }

        }

        stage('Scanning target on owasp container') {

            steps {

                script {

                    scan\_type = "${params.SCAN\_TYPE}"

                    echo "----> scan\_type: $scan\_type"

                    target = "${params.TARGET}"

                    if (scan\_type == 'Baseline') {

                        sh """

                             docker exec owasp \

                             zap-baseline.py \

                             -t $target \

                             -r report.html \

                             -I

                         """

                    }

                     else if (scan\_type == 'APIS') {

                        sh """

                             docker exec owasp \

                             zap-api-scan.py \

                             -t $target \

                             -r report.html \

                             -I

                         """

                     }

                     else if (scan\_type == 'Full') {

                        sh """

                             docker exec owasp \

                             zap-full-scan.py \

                             -t $target \

                             -r report.html \

                             -I

                         """

                     }

                     else {

                        echo 'Something went wrong...'

                     }

                }

            }

        }

        stage('Copy Report to Workspace') {

            steps {

                script {

                    sh '''

                         docker cp owasp:/zap/wrk/report.html ${WORKSPACE}/report.html

                     '''

                }

            }

        }

    }

    post {

        always {

            echo 'Removing container'

            sh '''

                     docker stop owasp

                     docker rm owasp

                 '''

            cleanWs()

        }

    }

 }

archiveArtifacts 'target/\*.jar' → REMOVED LINE

> Save & Build

"C:\Users\jeanz\Downloads\Docker Desktop Installer.exe"