**Lesson 09 Demo 04**

**Implementing OWASP ZAP DAST Scan in Jenkins Pipeline**

**Objective:** To implement the ZAP DAST tool using Jenkins declarative pipeline to automate code scan using Jenkins build job

**Tools required:** Jenkins

**Prerequisites:** You need to have a Jenkins up and running.

Steps to be followed:

1. Create a Jenkins pipeline job to integrate the vulnerability scan tool

**Step 1:** **Create a Jenkins pipeline job to integrate the vulnerability scan tool**

1. Open the terminal and execute the following commands to install container runtime and Docker tool to proceed with pipeline execution:

**sudo su**

**apt update**

**apt install containerd docker.io**

**chmod 777 /var/run/docker.sock**

**service docker restart**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

****

1. Log in to **Jenkins** using your credentials



**Note**: The credentials for accessing Jenkins in the lab are Username: **admin** and Password: **admin**.

1. In the Jenkins dashboard, click on **New Item**

A screenshot of a computer

Description automatically generated

1. Create a Jenkins pipeline job, click on **Pipeline**, put **ZAPPipeline** under **Enter an item name**, and then click on **OK**

A screenshot of a computer

Description automatically generated

1. In the **Configure** page, click on **Pipeline** and paste the following code under Script and then click on **Save**:

**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()**

**}**

**}**

**}**

A screenshot of a computer

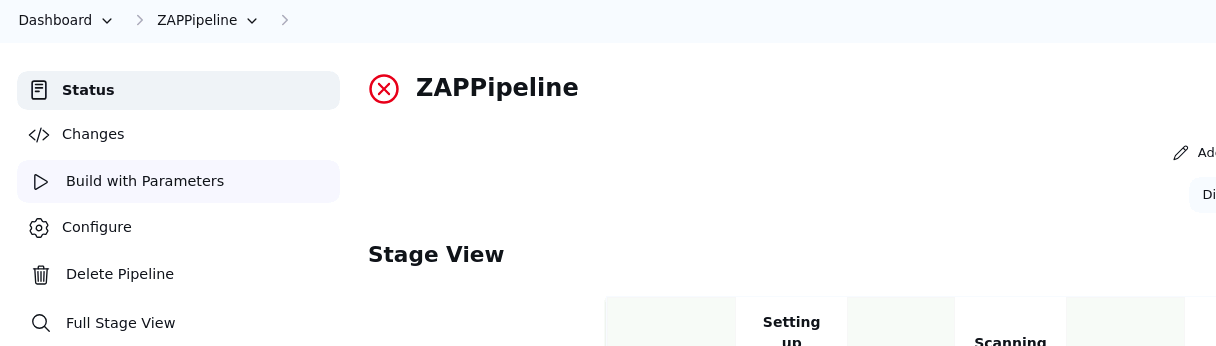
Description automatically generated

1. Now, click on **Build Now**

**A screenshot of a computer

Description automatically generated**

1. Initially, the pipeline will fail because it is a parameterized build. Then, **Build with Parameters** option appears. Click on it to initiate the pipeline with the provided parameters and click on **Build**.



A screenshot of a computer

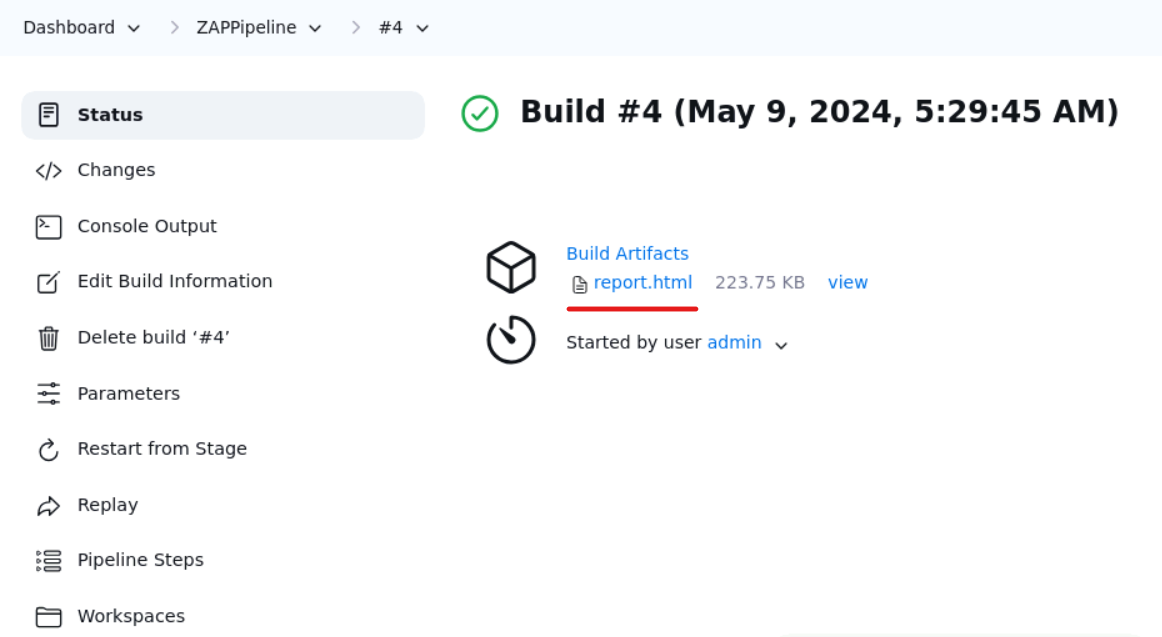
Description automatically generated

A screenshot of a computer

Description automatically generated

You can see the build is successful.

1. Navigate back to the **Status** of the build and click on **report.html** to see the HTML report archived in Jenkins



A screenshot of a report

Description automatically generated

You can see the HTML report.

By following these steps, you have successfully implemented the ZAP DAST tool using Jenkins declarative pipeline to automate code scan using Jenkins build job.