FortiDevSec - User Guide

Version 22.1

FORTINET®

June 27, 2022
FortiDevSec 22.1 User Guide

TABLE OF CONTENTS

Change log

Date	Change description
2022-02-04	FortiDevSec version 22.1 release document.
2022-02-28	Updated the SAST and DAST scan commands.
2022-03-07	Updated the SAST and DAST scan commands.
2022-03-25	Updated the CI/CD code segments.
2022-04-21	Updated the format of *fdevsec.yaml*.
2022-05-24	Added code snippets for AWS CodePipeline, Drone CI, and GCP Cloud Build.
2022-06-14	Formatted the CI/CD code segments.
2022-06-21	Formatted the CI/CD code segments.
2022-06-27	Formatted the CI/CD code segments.

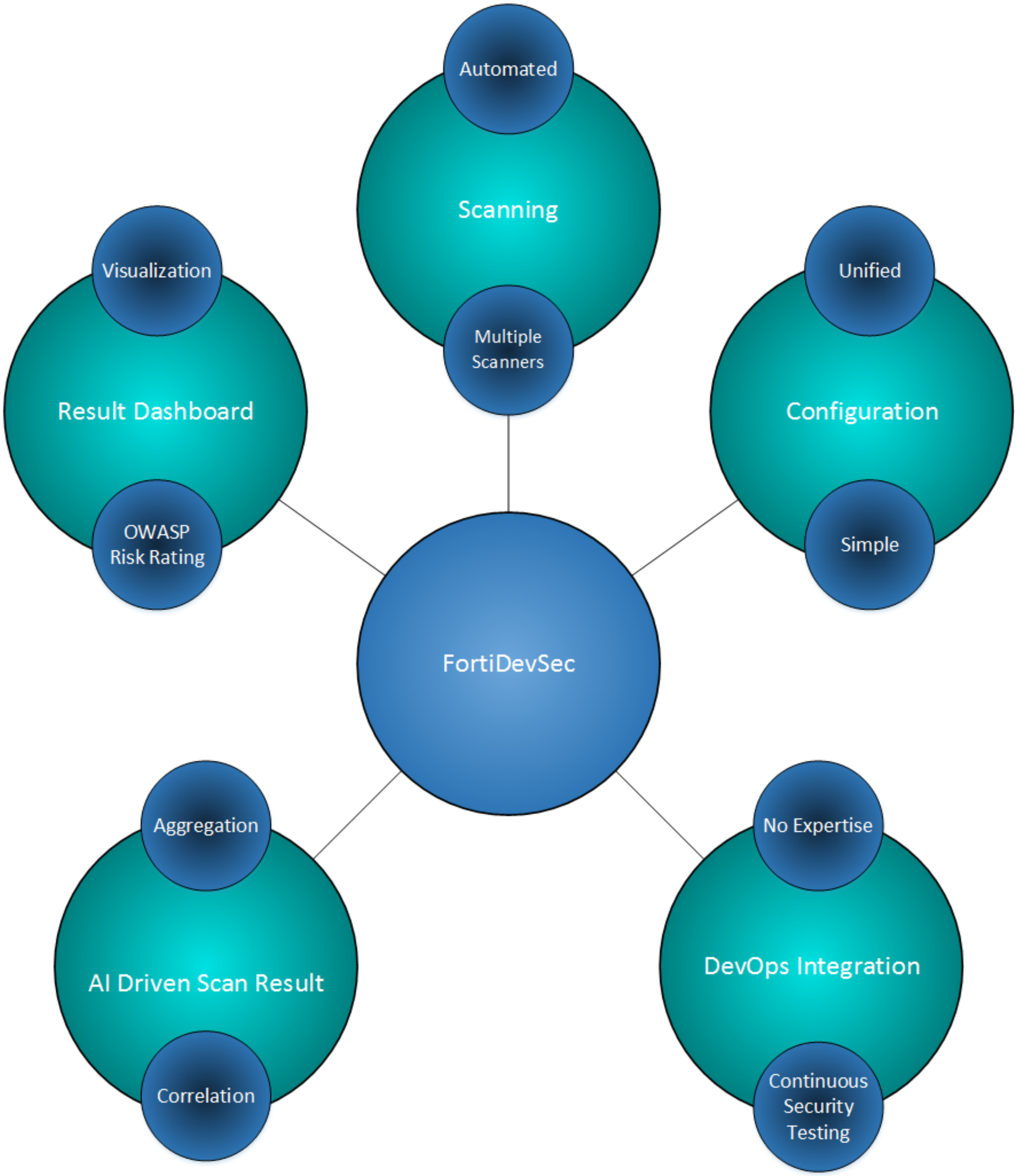
Introduction

The realm of application security involves tools and techniques to protect applications from attacks and violations. Due to the huge advancement in hacking techniques and cyber-attack methodologies even modern complex applications contain unassessed security risks and vulnerabilities that may lead to substantial harm to your organization/business. Evaluating the security risks associated with applications and assessing the security weaknesses allows you to mitigate the potential risk to your organization with appropriate remedial measures.

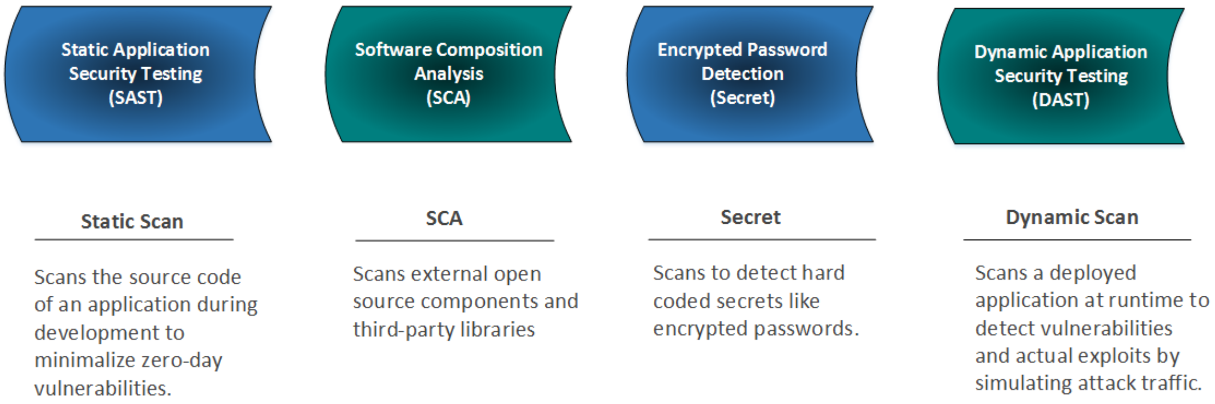
- What is FortiDevSec
- How FortiDevSec Works
- Licensing on page 8

What is FortiDevSec

FortiDevSec is a cloud-based automated application security tool that performs intensive and comprehensive scans for an accurate vulnerability assessment of your application. It integrates continuous application security testing into major DevOps Continuous Integration (CI)/Continuous Deployment (CD) environments, embedding itself into the process of developing and deploying applications to evaluate and detect security gaps that you can mitigate/remediate in the course of the Software Development Lifecycle (SDLC). The automated scanning process resides in your CI/CD pipeline and allows you to scan your applications without manual intervention and is completely non-intrusive with no disruptions to your setup. The easy-to-understand application security assessment approach of FortiDevSec allows you to build secure applications and involves a simple 3-step procedure that facilitates application scanning with minimal know-how of the application security domain.



FortiDevSec packages multiple security scanners into a single solution that includes source code scanners, run-time scanners, and open source component or third-party scanners. The FortiDevSec scanning process automatically determines the relevant scanner type(s) based on the application context and architecture. It uses Docker images with the latest version of those scanners and scans applications across multiple languages and frameworks. FortiDevSec provides zero effort deployment and saves you the overhead of installing and managing multiple scanners and plugging these into your setup individually.



Static Application Security Testing (SAST)
Software Composition Analysis (SCA)
Encrypted Password Detection (Secret)
Dynamic Application Security Testing (DAST)
Static Scan
SCA
Secret
Dynamic Scan

Scans the source code of an application during development to minimalize zero-day vulnerabilities.

Scans external open source components and third-party libraries

Scans to detect hard coded secrets like encrypted passwords.

Scans a deployed application at runtime to detect vulnerabilities and actual exploits by simulating attack traffic.

The application languages supported for SAST are *Java, Ruby on Rails, Python, Golang, PHP, JavaScript*, *C* and *C++*. The scanners supported for DAST are *FortiPenTest* and *DAST*.

The FortiDevSec application scanning is a simple procedure that includes creating a single unified configuration file and running the scan CLI. The *fdevsec.yaml* file integrates basic and advanced configurations for all security scanners and application languages avoiding fragmentation or multiple configuration steps.

The architecture of FortiDevSec integrates continuous application security testing into your DevOps CI/CD workflow and adopts a minimalistic approach towards the security testing procedure enabling DevOps personnel to integrate and run comprehensive application security scans without any domain expertise. It seamlessly integrates with all major devops CI/CD platforms to find security issues during the SDLC.

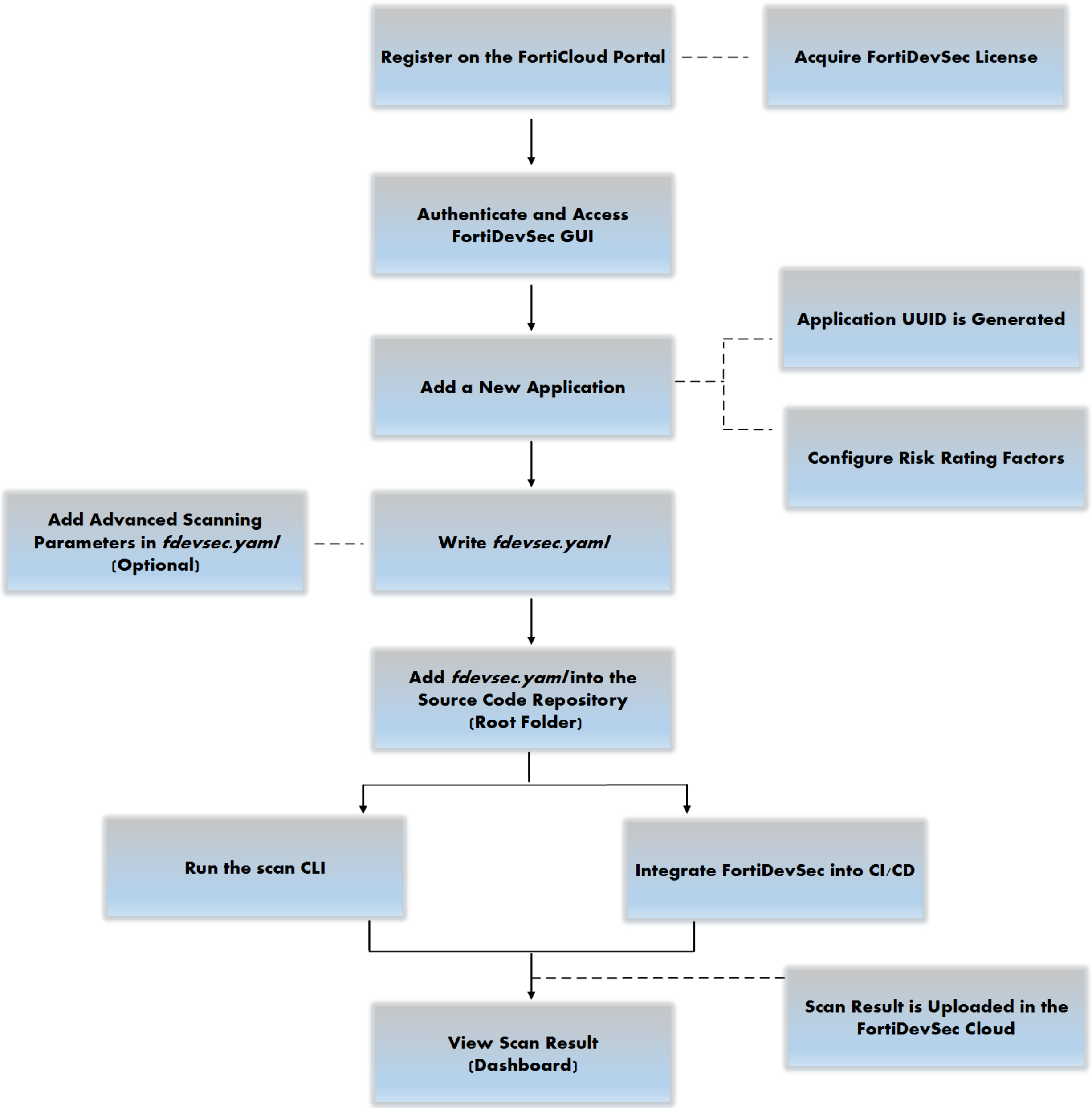
The scan result is aggregated and correlated for all applications across different scan types using advanced Artificial Intelligence (AI)/Machine Learning (ML) and uploaded in the FortiDevSec cloud providing a detailed insight into the scanned applications with a complete view of security risks. The applications are assigned standardized risk rating based on Open Web Application Security Project (OWASP) factors. The AI driven scan results and risk rating methodology prioritize the detected vulnerabilities based on the assessed severity with minimum false positives and noise. The interactive and customizable dashboard user interface is organized to display scan statistics in a distinctive way with ease of accessibility, navigation, and data filtering.

The high vulnerability detection rate and their intelligent prioritization in the FortiDevSec scan result offers robust risk determination capabilities that facilitate prompt response and appropriate remedial measures for the identified risks. You can configure the risk rating criteria for your application and based on the result analysis, you can manage the scan findings in the dashboard by assigning a suitable status to each vulnerability.

How FortiDevSec Works

You can scan your applications by integrating FortiDevSec into your CI/CD setup. When you run the scan, FortiDevSec automatically determines the open-source scanners to use based on the application language and other settings. It uses Docker images for the required scanners and uploads the scan results in the FortiDevSec cloud.

- To quickly scan your application, see section Beginner's Tutorial.
- For detailed configuration and scanning procedure, see section Scanning an Application.


Register on the FortiCloud Portal
Acquire FortiDevSec License
Authenticate and Access FortiDevSec GUI
Application UUID is Generated
Add a New Application
Configure Risk Rating Factors
Add Advanced Scanning Parameters in *fdevsec.yaml* (Optional)
Write *fdevsec.yaml*
Add *fdevsec.yaml* into the Source Code Repository (Root Folder)
Run the scan CLI
Integrate FortiDevSec into CI/CD
Scan Result is Uploaded in the FortiDevSec Cloud
View Scan Result (Dashboard)

Licensing

You are required to acquire a license to access FortiDevSec. The subscription licence is based on the number of source code users; each license is issued for maximum of 5 users. If you use FortiPenTest as the DAST scanner then your are also required to obtain a FortiPenTest license along with FortiDevSec. Contact the Fortinet *Customer Support* team for license purchase .

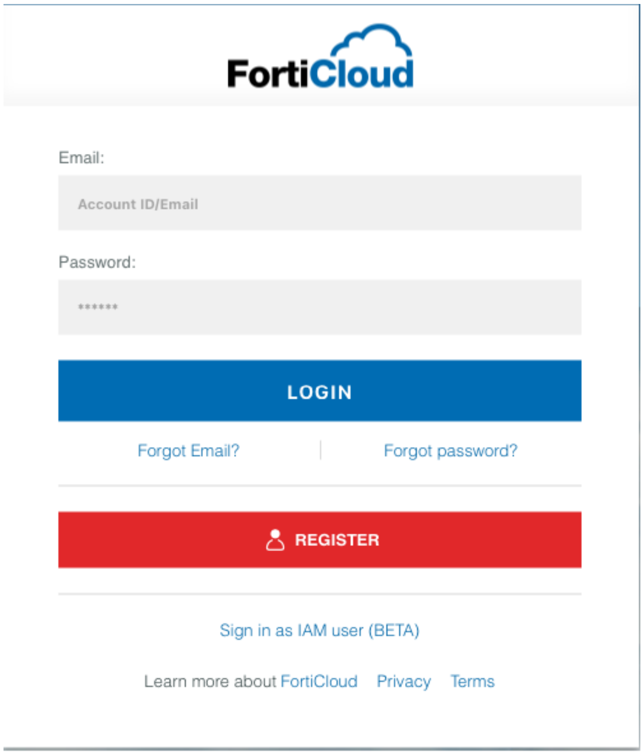
Signing-on for FortiDevSec

This release provides single sign-on support for FortiDevSec along with FortiCloud suite of products. FortiDevSec is accessible via the *FortiCloud* GUI - https://support.fortinet.com. Eventually you are redirected to the FortiDevSec login page.

- Registering on FortiCloud
- Accessing FortiDevSec

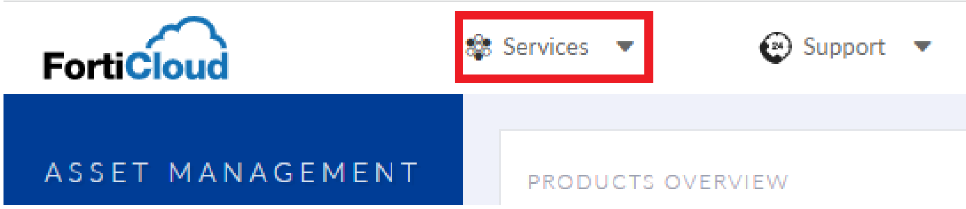
Registering on FortiCloud

Prior to using FortiDevSec, you are required to register on the *FortiCloud* portal. Use the https://support.fortinet.com access link to register on the *FortiCloud* portal. A security code is emailed to the address specified during registration; use the code to complete registration and activate your account.

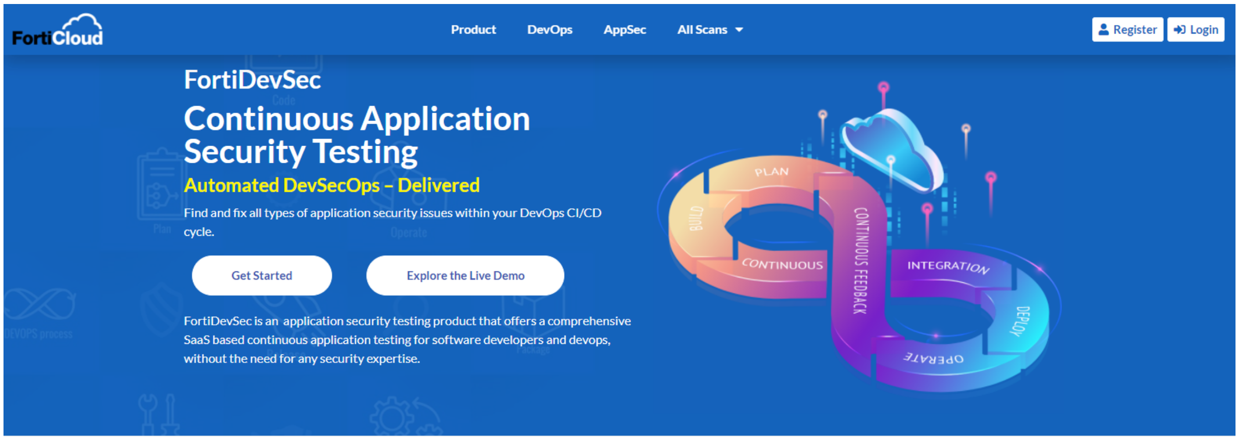


Accessing FortiDevSec

Any user registered on https://support.fortinet.com can access FortiDevSec. Once you login into *FortiCloud*, click on **Services**, a banner with Fortinet products is displayed.



Select FortiDevSec and you are redirected to the FortiDevSec portal, *https://fortidevsec.forticloud.com/*.



FortiDevSec

Continuous Application Security Testing

Automated DevSecOps – Delivered

Find and fix all types of application security issues within your DevOps CI/CD cycle.

FortiDevSec is an application security testing product that offers a comprehensive SaaS based continuous application testing for software developers and devops, without the need for any security expertise.

You can explore a demo of FortiDevSec with live application scan data.

Beginner's Tutorial

This tutorial aims at using FortiDevSec to run a security scan on your application quickly.

- Automated Application Scanning
- Manual Application Scanning

Automated Application Scanning

This tutorial aims to automate a security scan on your application in a CI/CD environment. See section Scanning an Application for more details.

- Adding a New Application
- Writing the fdevsec.yaml
- CI/CD Configurations
- Viewing the Scan Result

Adding a New Application

Login into the FortiDevSec portal and add a new application for your organization.

1. Click on the **New Application** tab and enter the application name.
2. Click **Next** and the **App Setup** information is displayed, copy the **Application UUID** and Org ID.

You can *optionally* configure the risk ratings for your application. See section Adding a New Application for detailed procedure.

Writing the *fdevsec.yaml*

Write the *fdevsec.yaml* file and integrate it into your CI/CD as defined in the next step (based on the CI/CD tool). This tutorial uses only the mandatory parameters in the configuration file, you can add optional (advanced) parameters to make your scan more precise.

The App ID and the Org ID are the only mandatory parameters.

```
version: v1
id:org: 6a4d32db-6751-441a-88fe-9b4793717cde
app: aa8a393b-afc6-47d7-84d2-b7011f1d0012
```

The application languages are automatically detected and FortiDevSec runs the appropriate scans, namely, SAST, SCA, and Secret.

See section Configuring the Scanner (fdevsec.yaml) for detailed procedure.

CI/CD Configurations

Integrate scan configurations into your CI/CD tool. See Running the Security Scan.

Viewing the Scan Result

The dashboard of the FortiDevSec portal lists the applications, click on your application to view and analyze comprehensive details of the detected vulnerabilities.

See section Viewing the Scan Result for more details.

Manual Application Scanning

This tutorial aims to run a security scan for your application manually in your source code through the CLI terminal. See section Scanning an Application for more details.

- Adding a New Application
- Writing the fdevsec.yaml
- Running the Scan
- Viewing the Scan Result

Adding a New Application

Login into the FortiDevSec portal and add a new application for your organization.

1. Click on the **New Application** tab and enter the application name.
2. Click **Next** and the **App Setup** information is displayed, copy the **Application UUID** and Org ID.

You can *optionally* configure the risk ratings for your application. See section Adding a New Application for detailed procedure.

Writing the *fdevsec.yaml*

Add the *fdevsec.yaml* file into the root folder of your source code. This tutorial uses only the mandatory parameters in the configuration file, you can add optional (advanced) parameters to make your scan more precise.

The App ID and the Org ID are the only mandatory parameters.

```
version: v1
id:
org: 6a4d32db-6751-441a-88fe-9b4793717cde
app: aa8a393b-afc6-47d7-84d2-b7011f1d0012
```

The application languages are automatically detected and FortiDevSec runs the appropriate scans, namely, SAST, SCA, and Secret.

See section Configuring the Scanner (fdevsec.yaml) for detailed procedure.

Running the Scan

In the Docker terminal navigate to the root folder of the source code and run `docker run --rm --mount type=bind,source="$PWD",target=/scan registry.fortidevsec.forticloud.com/fdevsec_sast:latest`.

Enter `sast` or `dast` as per your requirement. See section Running the Security Scan for detailed procedure.

Viewing the Scan Result

The dashboard of the FortiDevSec portal lists the applications, click on your application to view and analyze comprehensive details of the detected vulnerabilities.

See section Viewing the Scan Result for more details.

Scanning an Application

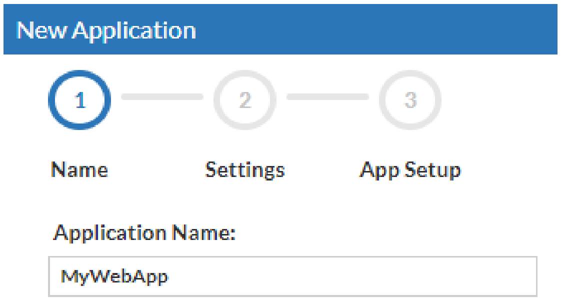
Perform these procedures to scan your applications for vulnerabilities.

- Adding a New Application
- Configuring the Scanner (fdevsec.yaml)
- Running the Security Scan
- Viewing the Scan Result

Adding a New Application

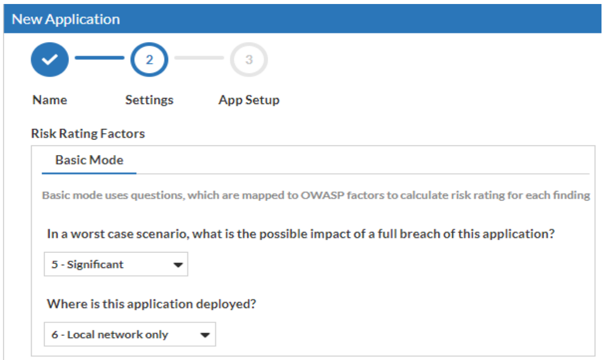
Adding your application in the FortiDevSec GUI to perform vulnerability scan testing.

1. Click the **New Application** tab and enter the **Application Name**.



Application Name: MyWebApp

2. Click **Next**.
3. In the **Settings** panel you can configure the risk rating factors based on questionnaire for this application .
Note: If you do not configure the risk rating factors then the displayed default settings are applied.
The following data is associated with the OWASP factors to calculate risk rating for your application.
   - Possible impact in case of a full breach of this application.
   - The application deployment details.



Risk Rating Factors

Basic mode uses questions, which are mapped to OWASP factors to calculate risk rating for each finding

In a worst case scenario, what is the possible impact of a full breach of this application?

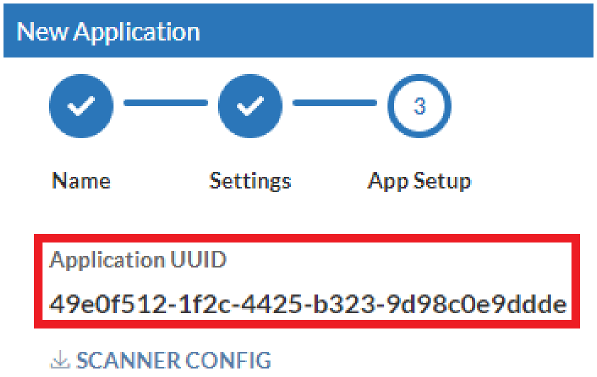
5 - Significant

Where is this application deployed?

6 - Local network only

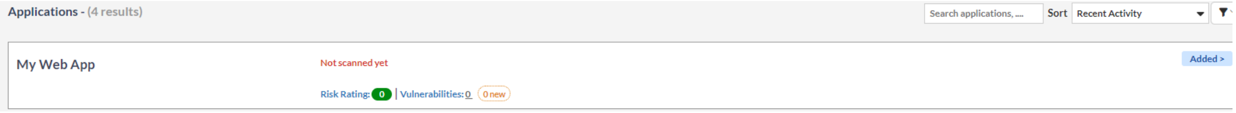
4. Click **Next**.
5. A unique **Application UUID** is generated to setup the application. You can copy this UUID now or later, alternately, click **Scanner Config** to download *fdevsec.yaml* that contains the application and

organization IDs.



Application UUID
49e0f512-1f2c-4425-b323-9d98c0e9ddde

Click **OK** and your application is listed in the dashboard.



Applications - (4 results)

My Web App Not scanned yet

Risk Rating: 0 | Vulnerabilities: 0 0 new

Configuring the Scanner (*fdevsec.yaml*)

Create the *fdevsec.yaml* file (or you may have downloaded it in Adding a New Application. Check-in or add this file into the root folder of the application source code.

Note: Do NOT modify the name and format of this file.

FortiDevSec automatically detects your application languages and runs the relevant scans, SAST, SCA, and Secret. However, to run DAST scans additional parameters are required in *fdevsec.yaml*, these are described later on in this section. You can also optionally add advanced settings to *fdevsec.yaml* file as per your requirements.

The following is a sample *fdevsec.yaml* file, the contents of this file vary based on different application scanning requirements.

```
version: v1

id:
  org: 6a4d32db-6751-441a-88fe-9b4793717cde
  app: aa8a393b-afc6-47d7-84d2-b7011f1d0012

# Optional parameters.
scanners:
  - sast
  - dast
  - secret
  - sca

languages:
  - python
  - javascript
```

```
dast:
  url: https://your.url.com
  fortipentest_scanner: true #true|false
  full_scan: true #true|false

resource:
  serial_scan: false #true|false
```

The following are the mandatory and optional parameters for *fdevsec.yaml*.

Parameter	Description
Mandatory parameters	
org	A unique ID associated with your organization.
app	A unique ID that identifies the applications within the organization.
Optional Parameters	
scanners	This identifies the type of scanner to test the applications. The supported values are **sast**, **dast**, **sca**, and **secrets**. **Notes:** • If this parameter is unspecified, FortiDevSec runs only static scans. • If a DAST URL is specified then a DAST scan runs along with SAST. Else, only the static scans are run.
languages	This identifies the language of the source code. The supported values are **java**, **javascript**, **python**, **golang**, **php**, **ruby**, **C++**, and **C**. FortiDevSec automatically detects the language if this parameter is not specified.
dast	Specify these parameters if you intend running a DAST scan on your application. • `url` - The URL where your application is hosted. • `full_scan` - The supported values are **true** and **false** (default). When set to **true**, a full DAST scan is run and when set to **false**, a basic scan is run. • `fortipentest_scanner` - The supported values are **true** and **false** (default). When set to **true**, FortiPenTest scanner is used. Else, the default DAST scanner is run. **Note**: You can configure the FortiPenTest scanner with specific parameters for testing your asset (URL). For details on scanner configuration see the FortiPenTest documentation.
resource	When `serial_scan` is set to `true`, the scans run consecutively and when set to `false`, multiple scans run parallel.

Note: To scan language files less than 10% of the coverage/threshold in the source code repository, you are required to explicitly specify the `languages` parameter and all languages under it in *fdevsec.yaml* file as a part of the SAST.

Running the Security Scan

You can integrate scan configurations into your CI/CD tool and automate the application scan testing for the following. Ensure that *fdevsec.yaml* file is checked into the root folder of your source code.

- AWS CodePipeline
- Azure DevOps
- Bamboo
- CircleCI
- Drone CI
- GCP Cloud Build
- GitHub Actions
- GitLab
- Jenkins
- Travis CI

AWS CodePipeline

Paste the following code segment in the *buildspec.yml* file for *only* for SAST scan.

```
version: 0.1
phases:
  install:
    commands:
      - echo "Entered the install phase..."
    finally:
      - echo "This always runs even if the update or install command fails"
  pre_build:
    commands:
      - echo "Entered the pre_build phase..."
  finally:
      - echo "This always runs even if the login command fails."
  build:
    commands:
      - echo "Entered the build phase..."
      - echo "Build started on `date`"
    finally:
      - echo "This always runs even if the install command fails"
  post_build:
    on-failure: CONTINUE
    commands:
      - echo "Entered the post_build phase..."
      - echo "Build completed on `date`"
      - echo "Running FortiDevSec SAST scanner..."
      - "docker pull registry.fortidevsec.forticloud.com/fdevsec_sast:latest"
      - "docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest"
```

Azure DevOps

Paste the following code segment in the *azure-pipelines.yml* file for a SAST scan.

```
trigger:
  — main
pool:
  vmImage: ubuntu-latest

steps:
  -task: Bash@3
    displayName: Install_Run_SAST
    inputs:
       targetType: 'inline'
       script: |
       docker pull registry.fortidevsec.forticloud.com/fdevsec_sast:latest
       docker run --rm --mount type=bind,source="$PWD",target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest
```

Paste the following code segment in the *azure-pipelines.yml* file for a DAST scan.

```
trigger:
  — main
pool:
  vmImage: ubuntu-latest

steps:
- task: Bash@3
   displayName: Install_Run_DAST
     inputs:
        targetType: 'inline'
        script: |
        docker pull registry.fortidevsec.forticloud.com/fdevsec_dast:latest
        docker run --rm --mount type=bind,source="$PWD",target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest
```

Bamboo

Paste the following code segment in the *bamboo.yml* file for a SAST scan.

```
— —
version: 2
plan:
  project-key: MYAPP
  name: Build the myapp
  key: MYAPP

stages:
-scan the myapp stage:
  jobs:
   — Scan
```

```
Scan:
  tasks:
   - clean # To keep the working directory clean
  -script:
   - docker pull registry.fortidevsec.forticloud.com/fdevsec_sast:latest
   - docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest
```

Paste the following code segment in the *bamboo.yml* file for a DAST scan.

```
– –
version: 2
plan:
  project-key: MYAPP
  name: Build the myapp
  key: MYAPP

stages:
-scan the myapp stage:
  jobs:
   — Scan

Scan:
  tasks:
   - clean # To keep the working directory clean
  -script:
   - docker pull registry.fortidevsec.forticloud.com/fdevsec_dast:latest
   - docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest
```

CircleCI

We have a CircleCI Orb. Paste this code segment in the *.circleci/config.yml* file for a SAST scan. Refer to the Orb Registry page to use the latest version.

```
version: 2.1
jobs:
  SAST:
   machine: yes
   steps:
   — checkout
   — run: |
     docker pull registry.fortidevsec.forticloud.com/fdevsec_sast:latest
     docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest
  workflows:
   Scans:
    jobs:
    — SAST
```

Paste this code segment in the *.circleci/config.yml* file for a DAST scan.

```
version: 2.1
jobs:
  DAST:
   machine: yes
   steps:
   — checkout
   — run: |
    docker pull registry.fortidevsec.forticloud.com/fdevsec_dast:latest
    docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest
  workflows:
   Scans:
    jobs:
    — DAST
```

Drone CI

Paste this code segment in the workflow *drone.yml* file for a SAST scan.

```
---
kind: pipeline
type: exec
name: SCAN

platform:
  os: linux
  arch: amd64

steps:
#Run FortiDevSec SAST Scanner, once the build step is done.
- name: SAST
  commands:
  - docker pull registry.fortidevsec.forticloud.com/fdevsec_sast:latest
  - docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest
```

Paste this code segment in the workflow *drone.yml* file for a DAST scan.

```
---
kind: pipeline
type: exec
name: SCAN

platform:
  os: linux
  arch: amd64

#Run FortiDevSec DAST Scanner, once the deploy step is done.
- name: DAST
  commands:
  - docker pull registry.fortidevsec.forticloud.com/fdevsec_dast:latest
  - docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest
```

GCP Cloud Build

Paste this code segment in the workflow *cloudbuild.yaml* file for a SAST scan.

```
steps:
# Run FortiDevSec SAST Scanner, once the build step is done.
- name: 'gcr.io/cloud-builders/docker'
  entrypoint: bash
  args: ['-c','docker run --rm --mount type=bind,source=$(pwd),target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest']
```

Paste this code segment in the workflow *cloudbuild.yaml* file for a DAST scan.

```
steps:
# Run FortiDevSec DAST Scanner, once the deploy step is done.
- name: 'gcr.io/cloud-builders/docker'
  entrypoint: bash
  args: ['-c','docker run --rm --mount type=bind,source=$(pwd),target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest']
```

GitHub Actions

Paste this code segment in the workflow *main.yml* file for a SAST scan.

```
name: FortiDevSec Scanner CI
on:
  push:
   branches: [ master ]
  pull_request:
   branches: [ master ]

jobs:
  build:
   runs-on: ubuntu-latest
   steps:
   - uses: actions/checkout@v2

   - name: SAST
    run: |
     docker pull registry.fortidevsec.forticloud.com/fdevsec_sast:latest
     docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest
```

Paste this code segment in the workflow *main.yml* file for a DAST scan.

```
name: FortiDevSec Scanner CI
on:
  push:
   branches: [ master ]
  pull_request:
   branches: [ master ]
```

```
jobs:
  build:
   runs-on: ubuntu-latest
   steps:
   - uses: actions/checkout@v2

   - name: DAST
    run: |
     docker pull registry.fortidevsec.forticloud.com/fdevsec_dast:latest
     docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest
```

GitLab

Paste this code segment in the *gitlab-ci.yml* file for a SAST scan.

```
SAST:
  tags:
   — devsecops
  image: registry.fortidevsec.forticloud.com/fdevsec_sast:latest
  stage: test
  script:
   — main
```

Paste this code segment in the *gitlab-ci.yml* file for a DAST scan.

```
DAST:
  tags:
   — devsecops
  image: https://registry.fortidevsec.forticloud.com/fdevsec_dast:latest
  stage: deploy
  script:
  — main
```

Jenkins

Paste this code segment in **Jenkins > (Your App) > Configure > Add build step > Execute Shell** for a SAST scan.

```
docker pull registry.fortidevsec.forticloud.com/fdevsec_sast:latest
docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest
```

Paste this code segment DAST scan.

```
docker pull registry.fortidevsec.forticloud.com/fdevsec_dast:latest
docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest
```

Travis CI

Paste this code segment in the *.travis.yml* file for a SAST scan.

```
language: python
python:
  — "3.6"
service:
  — docker
jobs:
  include:
    - stage: SAST
    before_install: docker pull registry.fortidevsec.forticloud.com/fdevsec_
sast:latest
    script: docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest
```

Paste this code segment in the *.travis.yml* file for a DAST scan.

```
language: python
python:
  — "3.6"
service:
  — docker
jobs:
  include:
    - stage: DAST
    before_install: docker pull registry.fortidevsec.forticloud.com/fdevsec_
dast:latest
    script: docker run --rm --mount type=bind,source=$PWD,target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest
```

To run a scan **manually**, navigate to the root folder of the source code and add the *fdevsec.yaml* file and run the following command.

```
docker run --rm --mount type=bind,source="$PWD",target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest.
```

In this command a SAST (`/fdevsec_sast:latest`) scan is run, modify the value to DAST (`/fdevsec_dast:latest`) if required.

The following image depicts a sample command for SAST.

```
devopsuser@User1:~/Repos/OWASPBenchmark$ docker run --rm --mount
type=bind,source="$PWD",target=/scan
registry.fortidevsec.forticloud.com/fdevsec_sast:latest
2022/02/03 06:33:57 Loaded scan config for Org ID: d9d3dc20-9372-4188-884f-
b18a5c75fe5c
2022/02/03 06:33:57 Languages configured in conf file: [java]
2022/02/03 06:34:02 Scanners configured in conf file: [sast]
2022/02/03 06:34:03 Total enabled scanners: 1
2022/02/03 06:34:03 Running parallel scan as per user config.
2022/02/03 06:37:25 FortiDevSec SAST scanner done, exiting.
```

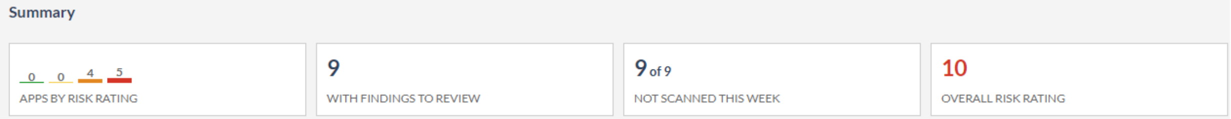
The following image depicts a sample command for DAST.

```
devopsuser@Dev:~/Repo/OWASPBenchmark$ docker run --rm --mount
type=bind,source="$PWD",target=/scan
registry.fortidevsec.forticloud.com/fdevsec_dast:latest
2022/02/03 08:37:19 Loaded scan config for Org ID: d9d3dc20-9372-4188-884f-
b18a5c75fe5c
2022/02/03 08:37:19 Scanners configured in conf file: [dast]
2022/02/03 08:37:20 Response Status: 202 Accepted
2022/02/03 08:37:20 Total enabled scanners: 0
2022/02/03 08:37:20 No scanners specified.
2022/02/03 08:37:20 FortiDevSec DAST scanner done, exiting.
```

Viewing the Scan Result

The FortiDevSec scan result for application vulnerability scanning is populated in a comprehensive dashboard that provides an insight into the scanned applications categorizing the findings based on the scanners used with the calculated risk rating indicators. You can view the detected vulnerabilities' details per application.

The dashboard summary panel displays a gist of the scanned applications that include categorization of the applications scanned as per the assigned risk rating, the number of findings to review, the number of applications not scanned in the current week, and the overall average risk rating of the applications scanned.


Summary

0 0 4 5
APPS BY RISK RATING

9
WITH FINDINGS TO REVIEW

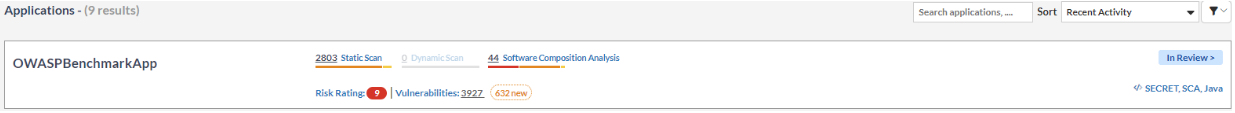
9 of 9
NOT SCANNED THIS WEEK

10
OVERALL RISK RATING

- Viewing Scanned Applications
- Viewing Scanned Application Details
- Viewing Vulnerabilities
- Modifying the Vulnerability Status
- Viewing Vulnerability Details
- Applying Dasboard Filters

Viewing Scanned Applications

The application panel lists all the scanned applications with basic details. Consider this example screenshot from the dashboard.



Applications - (9 results)

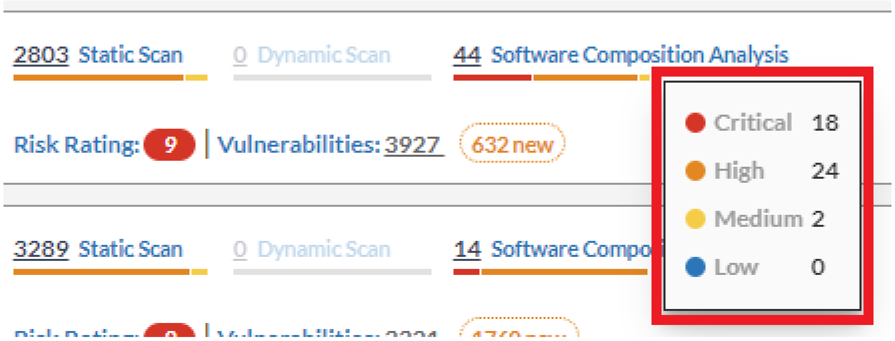
OWASPBenchmarkApp

2803 Static Scan | 0 Dynamic Scan | 44 Software Composition Analysis

Risk Rating: 9 | Vulnerabilities: 3927 632 new

You can analyze the following information specific to each application.

- The number of vulnerabilities found by each scanner type, in this case, 2803 vulnerabilities are found by SAST and 44 by SCA. Click on the legend for each scan type to view the categorization of the findings by their severity.



2803 Static Scan | 0 Dynamic Scan | 44 Software Composition Analysis

Risk Rating: 9 | Vulnerabilities: 3927 632 new

3289 Static Scan | 0 Dynamic Scan | 14 Software Compo

● Critical	18
● High	24
● Medium	2
● Low	0

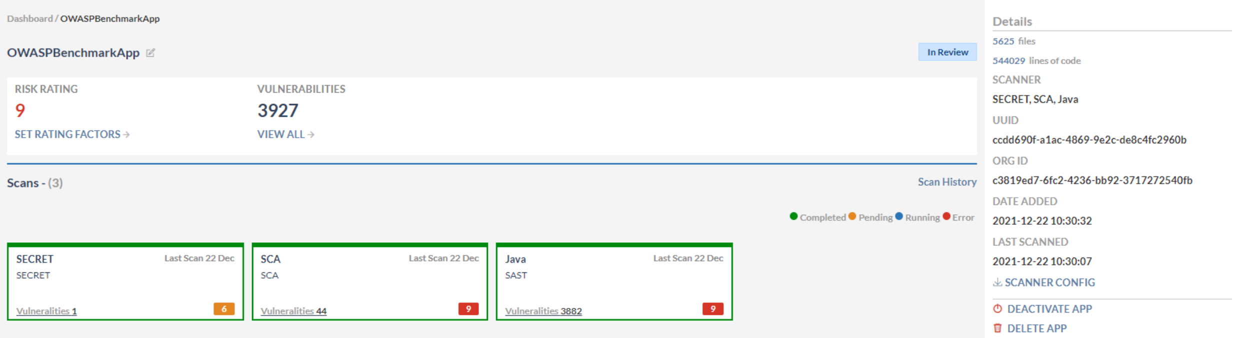
- The risk rating assigned by FortiDevSec for this application.
- The total number of vulnerabilities detected.

Click on the application name or the number of vulnerabilities to view scan details.

Viewing Scanned Application Details

In this panel details such as, the scanner types used with a break-up of the number of vulnerabilities found by each scanner and the associated risk rating are displayed. In this example, there are 3927 vulnerabilities found by **SAST**, 44 by **SCA**, and 1 by the **Secret** scanner. Click on the number of vulnerabilities for any scanner type to view the specifics, click **View All** to view details of all vulnerabilites detected.

Modify Risk Rating - You can modify the risk rating settings for the application on this page, click **Set Rating Factors**.



Dashboard / OWASPBenchmarkApp

OWASPBenchmarkApp

RISK RATING
9
SET RATING FACTORS →

VULNERABILITIES
3927
VIEW ALL →

Scans - (3)

SECRET — Last Scan 22 Dec — SECRET — Vulneralities 1 — 6

SCA — Last Scan 22 Dec — SCA — Vulneralities 44 — 9

Java — Last Scan 22 Dec — SAST — Vulneralities 3882 — 9

Details

5625 files

544029 lines of code

SCANNER
SECRET, SCA, Java

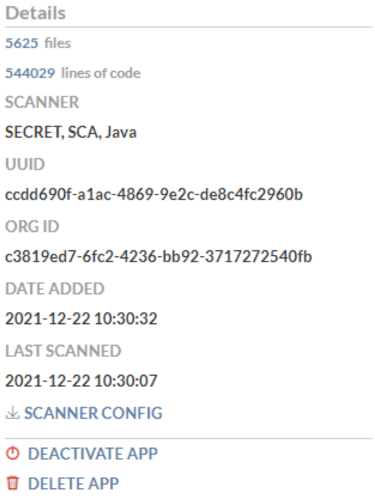
UUID
ccdd690f-a1ac-4869-9e2c-de8c4fc2960b

ORG ID
c3819ed7-6fc2-4236-bb92-3717272540fb

DATE ADDED
2021-12-22 10:30:32

LAST SCANNED
2021-12-22 10:30:07

Scan details are listed in the panel on the right side.



Details

5625 files

544029 lines of code

SCANNER

SECRET, SCA, Java

UUID

ccdd690f-a1ac-4869-9e2c-de8c4fc2960b

ORG ID

c3819ed7-6fc2-4236-bb92-3717272540fb

DATE ADDED

2021-12-22 10:30:32

LAST SCANNED

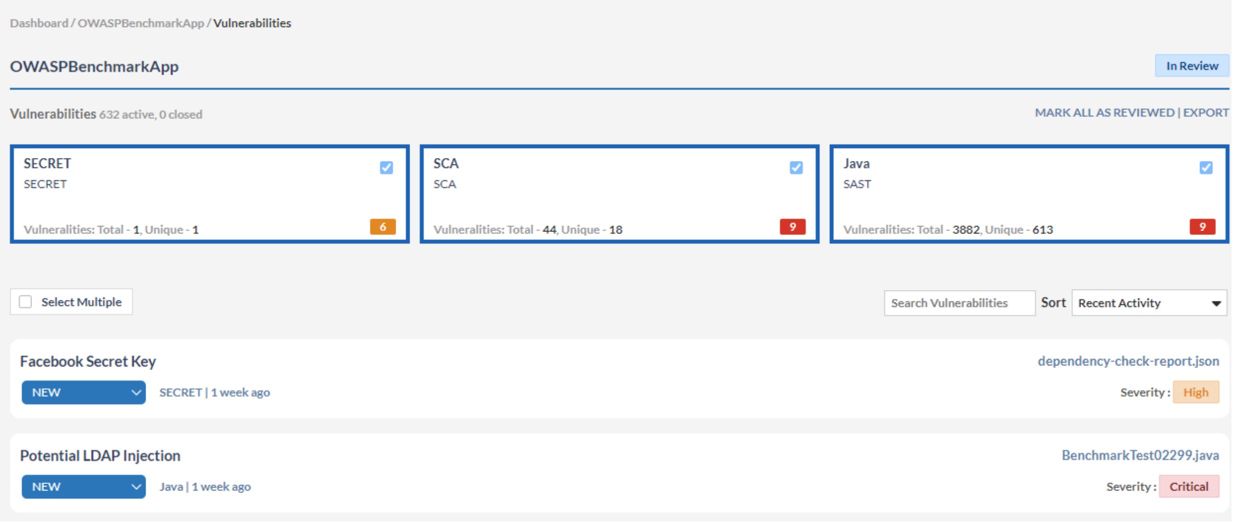
2021-12-22 10:30:07

The displayed application related data includes the number of **files** and lines of **code** scanned, **scanner** types used, the **UUID** and organization **ID**, and the time when the application was **added** and **last scanned**. Click **Scanner Config** to download the *fdevsec.yaml* file.

Deactivating/Deleting the Application - You can deactivate an application wherein no modification is allowed to the application vulnerability findings but you are allowed to view them. You can delete an application from the dashboard only after deactivating it.

Viewing Vulnerabilities

All vulnerabilities are listed in this panel along with the associated source file name and the line number (SAST)/URL (DAST) and the assigned severity. The vulnerabilities are categorized as **Active** and **Closed**, the active vulnerabilities are those that are currently present in your application. In this case, there are 632 active vulnerabilities. The vulnerabilities for each scanner type display the number of **Unique** vulnerabilities. A unique vulnerability indicates the type of vulnerabilities, that is, the vulnerability can have multiple instances but it is counted only once here.



Dashboard / OWASPBenchmarkApp / Vulnerabilities

OWASPBenchmarkApp

Vulnerabilities 632 active, 0 closed

SECRET
SECRET
Vulneralities: Total - 1, Unique - 1

SCA
SCA
Vulneralities: Total - 44, Unique - 18

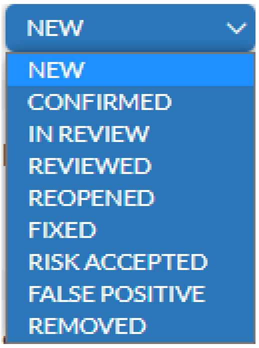
Java
SAST
Vulneralities: Total - 3882, Unique - 613

Facebook Secret Key
dependency-check-report.json
NEW
SECRET | 1 week ago
Severity : High

Potential LDAP Injection
BenchmarkTest02299.java
NEW
Java | 1 week ago
Severity : Critical

Modifying the Vulnerability Status

You can modify the status of each vulnerability or of all vulnerabilities, select **Select Multiple** and set the status.

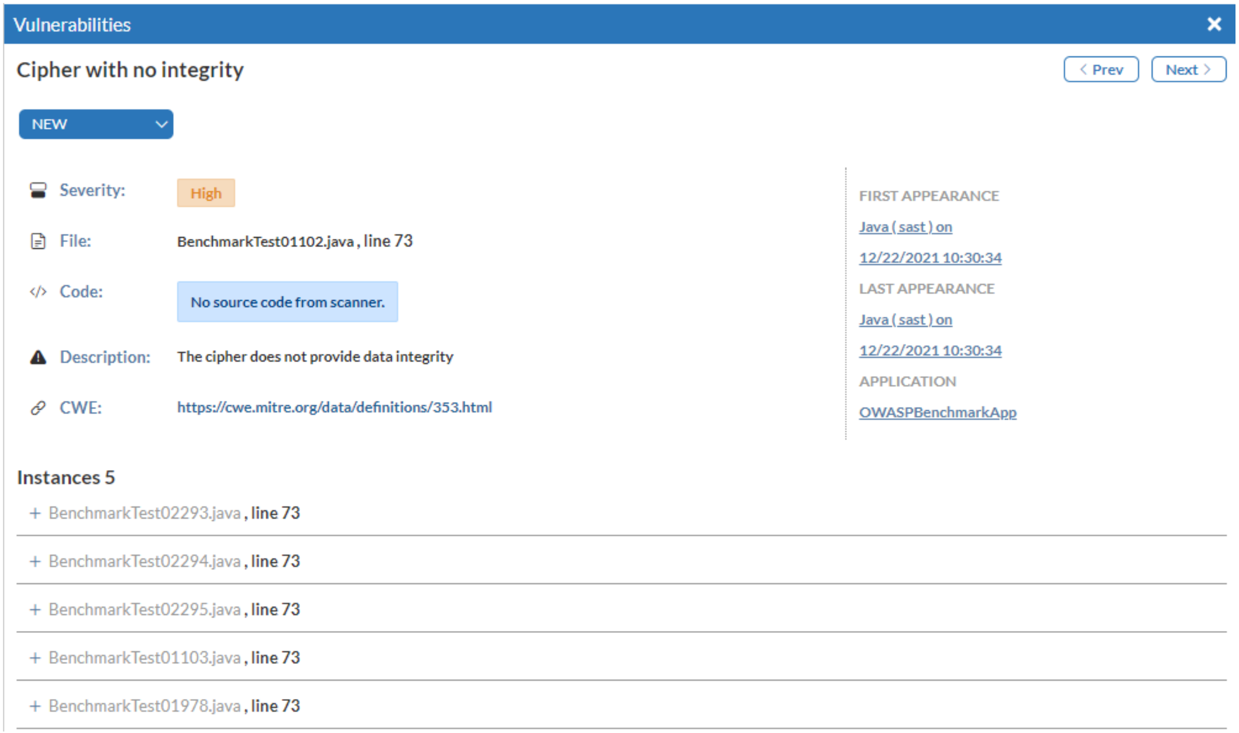


The following status types are supported.

- **New**: This is a new vulnerability detected by the scan.
- **Confirmed**: This is a real vulnerability and requires a fix.
- **In Review**: This vulnerability is currently in review/looked into for further action.
- **Reviewed**: This vulnerability review is complete.
- **Reopened**: This is a fixed vulnerability detected again in the rescan and requires to be addressed.
- **Fixed**: This vulnerability is fixed and does not appear in the next scan result.
- **Risk Accepted**: This vulnerability is an accepted risk and continues to exist without any potential damage.
- **False Positive**: This vulnerability is a potential flaw in the scanner or is indicative of a unique feature of the application.
- **Removed**: This vulnerability is overlooked in the application.

Viewing Vulnerability Details

Click on the vulnerability name to view all details associated with each finding.



Vulnerabilities

Cipher with no integrity

< Prev Next >

NEW

Severity: High

File: BenchmarkTest01102.java , line 73

Code: No source code from scanner.

Description: The cipher does not provide data integrity

CWE: https://cwe.mitre.org/data/definitions/353.html

FIRST APPEARANCE
Java (sast) on
12/22/2021 10:30:34

LAST APPEARANCE
Java (sast) on
12/22/2021 10:30:34

APPLICATION
OWASPBenchmarkApp

Instances 5

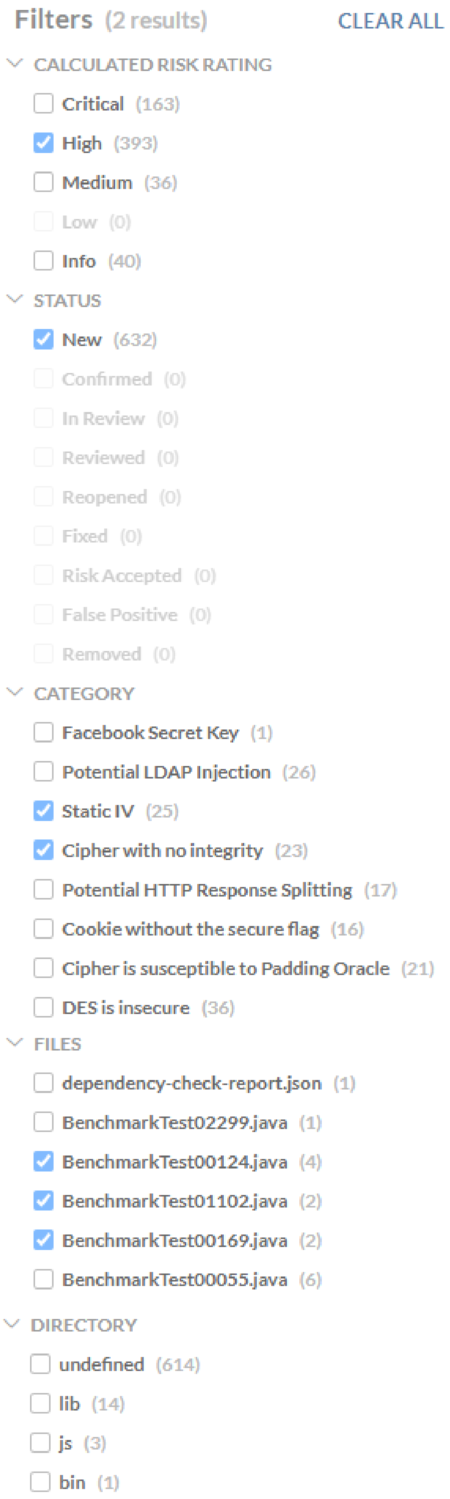
+ BenchmarkTest02293.java , line 73
+ BenchmarkTest02294.java , line 73
+ BenchmarkTest02295.java , line 73
+ BenchmarkTest01103.java , line 73
+ BenchmarkTest01978.java , line 73

- The details displayed are the risk rating (**severity**) assigned by FortiDevSec.
- The associated **file** and the **line number** that the vulnerability is found in.
- The vulnerability **description** and the associated **CWE** (if any). Click on the CWE link to view details.
- The number of **instances** that it is found in, click on each instance to view details. Click to expand each of the instance.
- The **history** of the vulnerability is also displayed that includes the time of its first and last appearence.

Applying Dasboard Filters

You can filter the displayed findings based on specific criteria. The following filters are available on the left-side panel of the dashboard.

- **Calculated Risk Rating** - Filtered based on the assigned risk rating.
- **Status** - Filtered based on the status.
- **Category** - Filtered based on the specific application.
- **Files** - Filtered based on the specific files.
- **Directory** - Filtered based on the specific directories.



Filters (2 results) CLEAR ALL

CALCULATED RISK RATING

- □ Critical (163)
- ☑ High (393)
- □ Medium (36)
- □ Low (0)
- □ Info (40)

STATUS

- ☑ New (632)
- □ Confirmed (0)
- □ In Review (0)
- □ Reviewed (0)
- □ Reopened (0)
- □ Fixed (0)
- □ Risk Accepted (0)
- □ False Positive (0)
- □ Removed (0)

CATEGORY

- □ Facebook Secret Key (1)
- □ Potential LDAP Injection (26)
- ☑ Static IV (25)
- ☑ Cipher with no integrity (23)
- □ Potential HTTP Response Splitting (17)
- □ Cookie without the secure flag (16)
- □ Cipher is susceptible to Padding Oracle (21)
- □ DES is insecure (36)

FILES

- □ dependency-check-report.json (1)
- □ BenchmarkTest02299.java (1)
- ☑ BenchmarkTest00124.java (4)
- ☑ BenchmarkTest01102.java (2)
- ☑ BenchmarkTest00169.java (2)
- □ BenchmarkTest00055.java (6)

DIRECTORY

- □ undefined (614)
- □ lib (14)
- □ js (3)
- □ bin (1)