Techstrong Deepfactor SCA 2.0 Workshop

Introduction

This workshop is designed to showcase Deepfactor's runtime SCA capability and how it can be used to prioritize SCA vulnerabilities. In this workshop we will scan and run a spring boot container image and experience how we can use the SCA 2.0 framework to prioritize the true risk rather than relying on CVSS score alone.

Goal

Experience the power of Deepfactor's SCA 2.0 framework!

Workshop Logistics

Activate Deepfactor Account

Please check for an email with the subject "*Deepfactor Workshop has invited you to create a Deepfactor account*" sent to your email address used to register for the workshop.

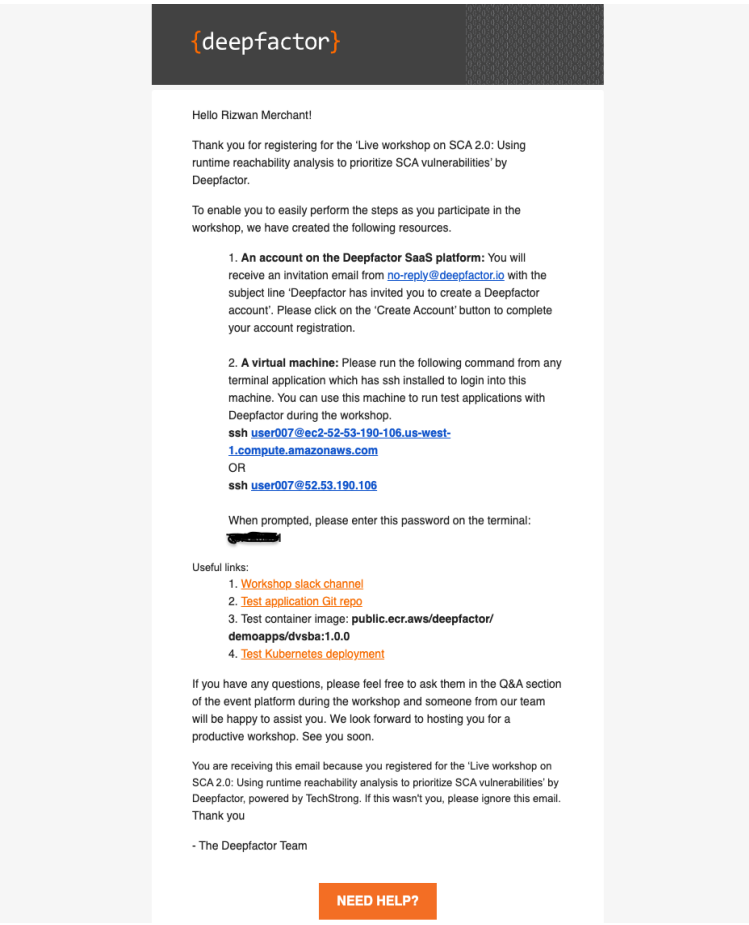
Click on the CREATE ACCOUNT link to activate your account. Once the account is activated, you may proceed to login step # 1 from the workshop section.

Additional resources

A few additional resources are shared with you over an email. These include the following:

1. A test virtual machine: You should have received the details of your test machine over email. If you have not received this email, please reach out to one of the presenters via private chat and they will assist you.
2. GitHub repository for the application used in this workshop: https://github.com/deepfactor-io/demo-vulnerable-springboot-app
   Deepfactor slack channel for this workshop: Please join the channel to discuss with the community using this link: https://join.slack.com/share/enQtNjA0NTAyMjEzMjI2MS05MGVhODNiOGNiMTNkNjc5ZjAxOTg0ZDUxNzVjOWU2Nzc3M2MzNTFlODY4NWEzNWZkY2M0NWZmZDNlNDdkYjQ5

3. Container image that will be scanned and instrumented with Deepfactor: **public.ecr.aws/deepfactor/demoapps/dvsba:1.0.0**



{deepfactor}

Hello Rizwan Merchant!

Thank you for registering for the 'Live workshop on SCA 2.0: Using runtime reachability analysis to prioritize SCA vulnerabilities' by Deepfactor.

To enable you to easily perform the steps as you participate in the workshop, we have created the following resources.

1. **An account on the Deepfactor SaaS platform:** You will receive an invitation email from no-reply@deepfactor.io with the subject line 'Deepfactor has invited you to create a Deepfactor account'. Please click on the 'Create Account' button to complete your account registration.

2. **A virtual machine:** Please run the following command from any terminal application which has ssh installed to login into this machine. You can use this machine to run test applications with Deepfactor during the workshop.
ssh user007@ec2-52-53-190-106.us-west-1.compute.amazonaws.com
OR
ssh user007@52.53.190.106

When prompted, please enter this password on the terminal:
[redacted]

Useful links:
1. Workshop slack channel
2. Test application Git repo
3. Test container image: **public.ecr.aws/deepfactor/demoapps/dvsba:1.0.0**
4. Test Kubernetes deployment

If you have any questions, please feel free to ask them in the Q&A section of the event platform during the workshop and someone from our team will be happy to assist you. We look forward to hosting you for a productive workshop. See you soon.

You are receiving this email because you registered for the 'Live workshop on SCA 2.0: Using runtime reachability analysis to prioritize SCA vulnerabilities' by Deepfactor, powered by TechStrong. If this wasn't you, please ignore this email.
Thank you

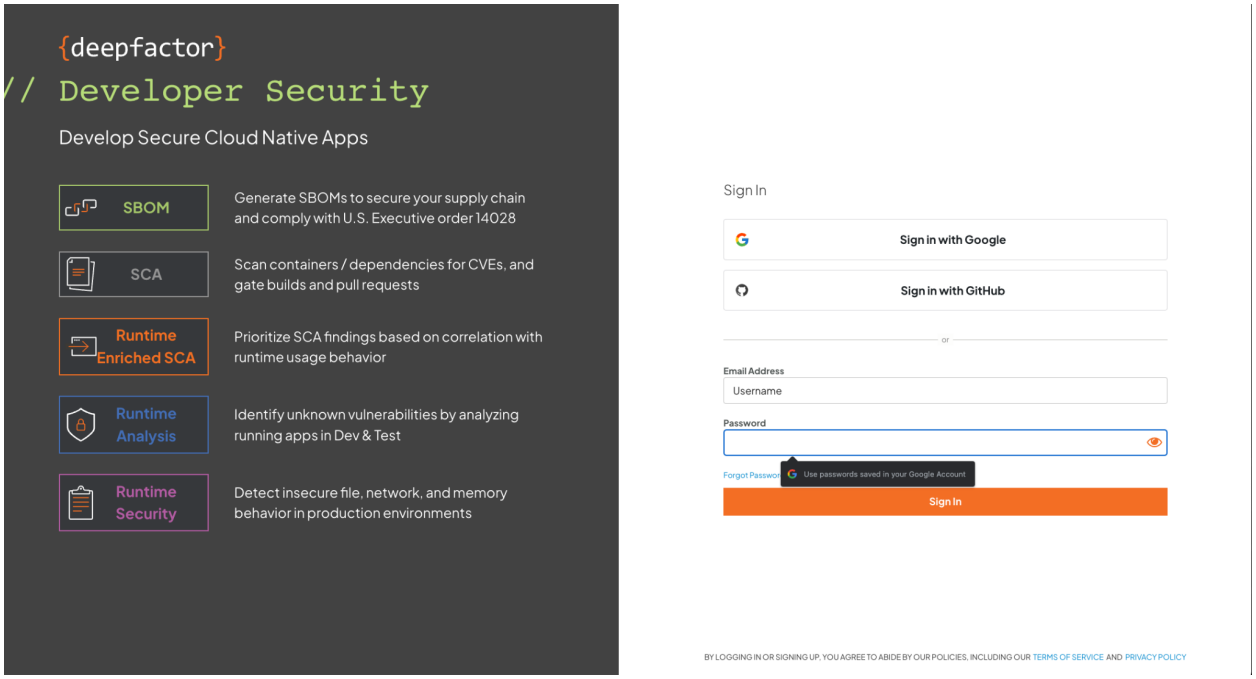
- The Deepfactor Team

NEED HELP?

Workshop

Step 1 - Login to Deepfactor Portal

Login to Deepfactor portal using your account credentials. Following image is the screenshot of the login page for reference



Deepfactor Platform Login screen

Step 2 - Log into the test VM

Log into your test machine using the instructions provided in the email from Deepfactor. If you have not received this email, please reach out to one of the presenters via private chat and they will assist you.

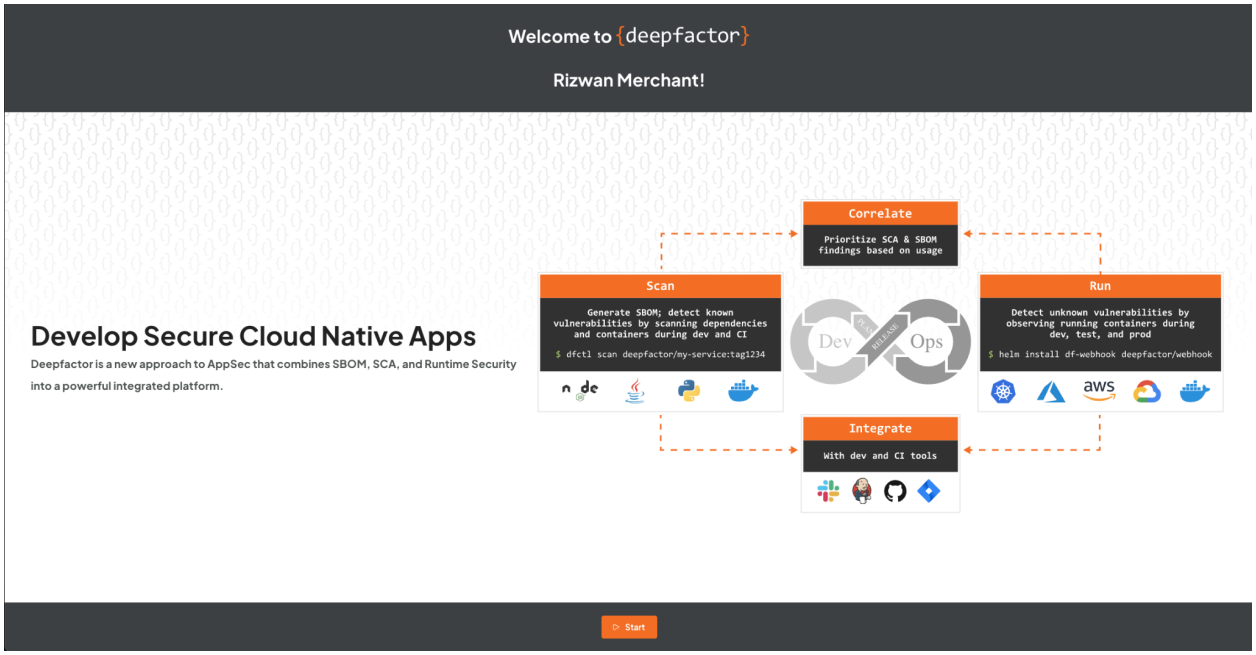
Following is a sample command.

```
cmd #> ssh user007@ec2-52-53-190-106.us-west-1.compute.amazonaws.com
```

Note: Every participant will receive a test machine of their own and the instructions to login to the test machine are provided in the email from Deepfactor.

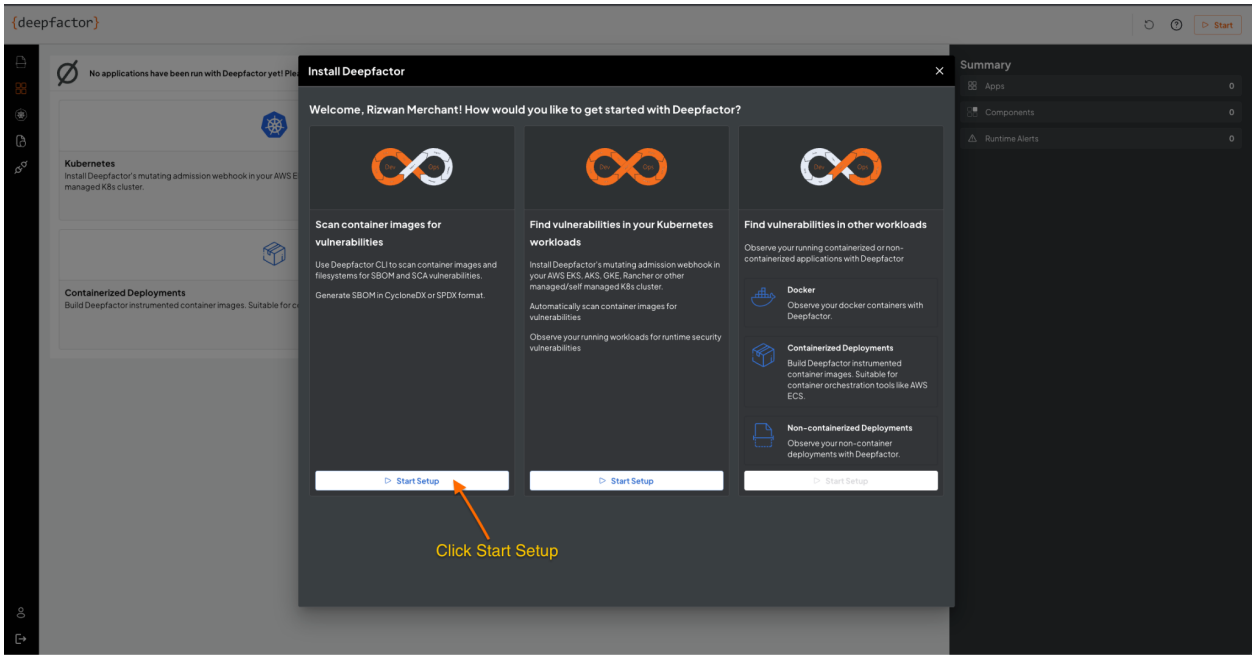
Step 3 - Copy the Run Token

Step 3a - After successful login, click on the Start button at the bottom of the screen.



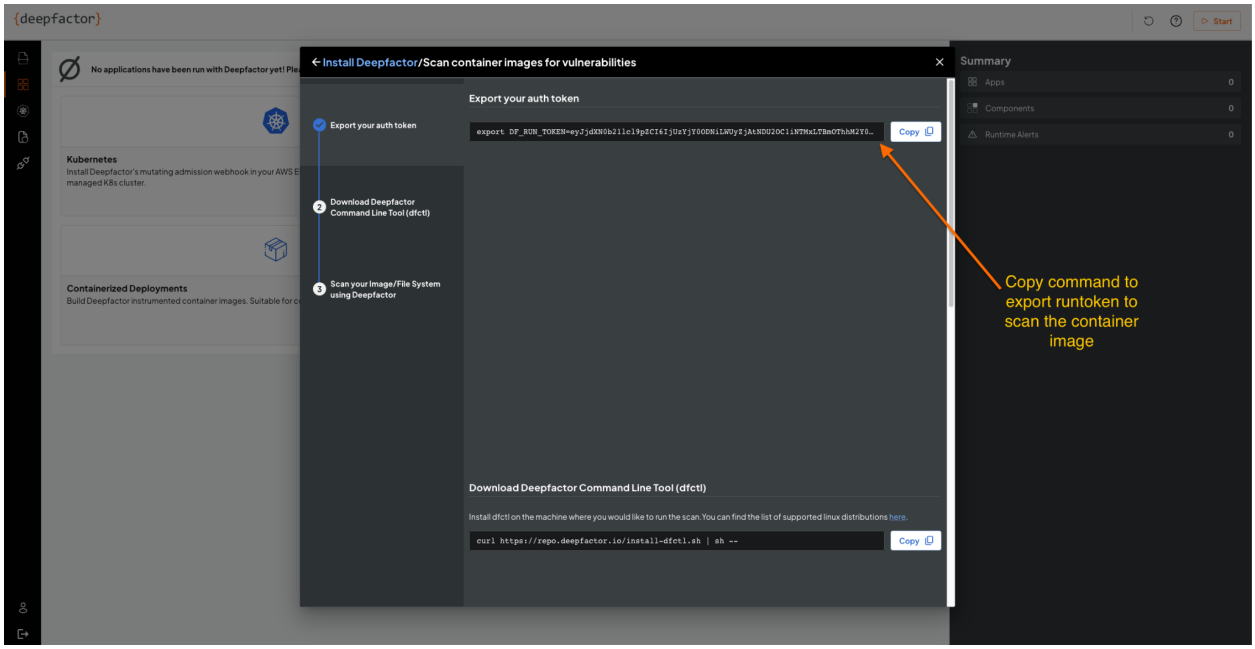
Step 3b

You will now be shown the ‘Install Deepfactor’ dialog. Click on the ‘Start Setup’ button in the leftmost section of the dialog titled ‘Scan container images for vulnerabilities’.



Step 3c

Copy and execute on your terminal, the first command ‘Export your auth token’



Step 4 - Scan container image

After export the Deepfactor run token, please scan the container image using the `dfctl scan` command shown below

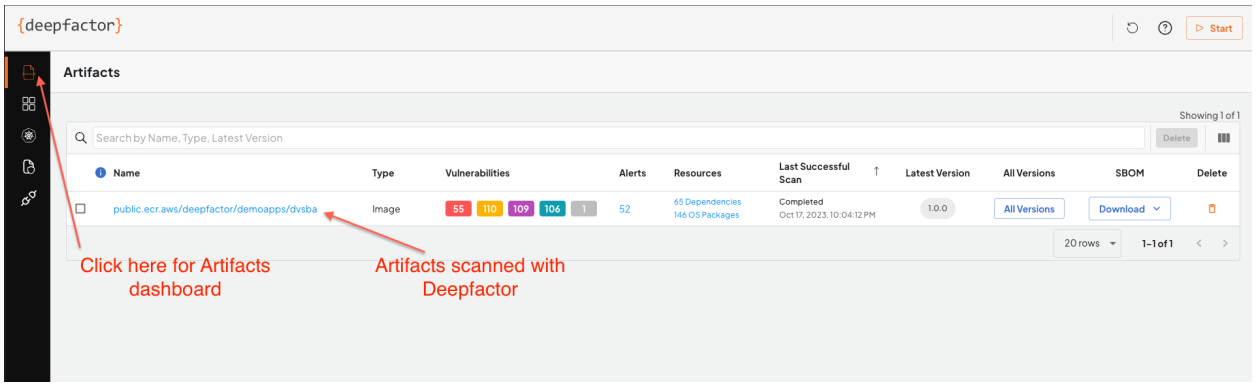
```
cmd #> export DF_RUN_TOKEN=<Run Token From Your Account>

cmd #> dfctl scan public.ecr.aws/deepfactor/demoapps/dvsba:1.0.0
```

Following is a sample output:

```
dfctl scan public.ecr.aws/deepfactor/demoapps/dvsba:1.0.0
Starting image scan
No match for registry type found
2023-10-18T03:42:33.882Z   info  successfully refreshed access token
2023-10-18T03:42:33.883Z   info  starting image scan...
2023-10-18T03:42:33.987Z   info  successfully registered scan agent
2023-10-18T03:42:33.988Z   info  artifact validation in progress...
2023-10-18T03:42:34.043Z   info  artifact validation done
2023-10-18T03:42:34.043Z   info  scan registration in progress...
2023-10-18T03:42:34.231Z   info  scan registration done
2023-10-18T03:42:34.231Z   info  scan in progress...
2023-10-18T03:42:34.292Z   info  scan complete
2023-10-18T03:42:34.301Z   info  Gathering exploit information
...
...
Deepfactor scan completed in 5 seconds.
```

After the scan completes you can check the Artifacts dashboard on Deepfactor portal for static SCA & SBOM of the scanned container image. Following is sample screen capture of the dashboard



Step 5 - Run the application

Now that we have scanned the container image, in this step, we will run it with Deepfactor to observe its runtime behavior and gather usage of dependencies.
Run the container image using the following command. Make sure your run token is exported on your terminal session before you run your application

```
cmd #> dfctl run -a "vuln-spring-boot-app" -c "java" --docker-run -d -name
vuln-spring-boot-app --image public.ecr.aws/deepfactor/demoapps/dvsba:1.0.0
```

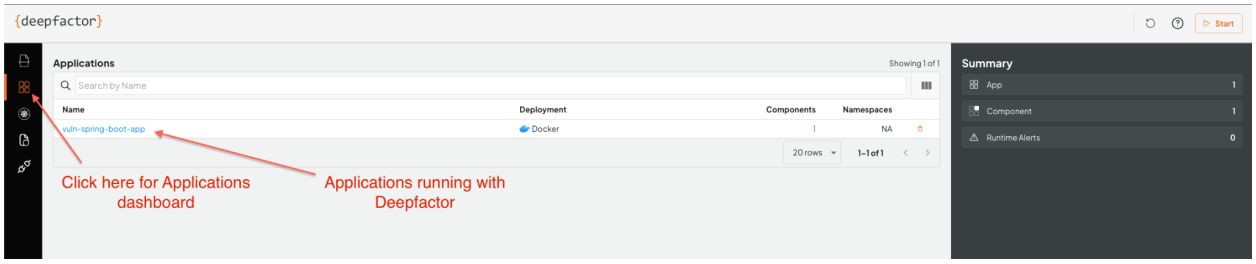
Following is sample output of this command

```
test: dfctl version:       "3.3.3-r2346" "6ef0f853418937e4d81ce89b88fbd9afb14f26f1"

test: dfctl: checking command line java

5ffe80a33767cccb0b920bcc0de49dd5f566e381b90b2aab246c11fedb2f5fe6
```

After the application starts up, you can check the Applications dashboard on Deepfactor portal for runtime insights and alerts. Following is sample screen capture of the dashboard



Step 6 - Exercise your application

Deepfactor observes every process within the container. In this step, we will run a command from within the container image.

Run the following commands

```
cmd #> docker exec -it vuln-spring-boot-app /bin/bash
root@xyz #> find /
```

Following is a screenshot of the application after the running the above command in the container running with Deepfactor