UDACITY

Return to Classroom

DISCUSS ON STUDENT HUB

Hardened Microservices Environment

REVIEW

CODE REVIEW

HISTORY

Requires Changes

2 specifications require changes

Dear Student,

Amazing work in this submission I was truly happy to see the results of all your learnings come so clearly into your final project. I really love to see your understanding of the `Docker` and `Kubernetes` environment and you harden the environment perfectly. However, there are some improvements that required your attention.

Here are some points where you should need to pay more attention:

1. ⚠ The submission includes the app.py file with changes that fix the Cross-Site Scripting vulnerability
2. ⚠ Write the pods name that was running during the incident.

Note:- I had given you more detailed feedback below on the rubric parts so please take a look at them to fix the issues.

Suggestion

If you have a query feel free to visit the knowledge platform where you can **ask questions to the mentor** regarding

If you have a query feel free to visit the knowledge platform where you can **ask questions to the mentor** regarding your project issues or you can search the question that was previously posted by the students. Our mentor team will provide you with continuous support on the knowledge platform.

Extra Materials

Here are a few resources you might find useful for more insight and further learning.

- Better Dashboarding – Grafana or SquaredUp?
- What is Grafana? Why Use It? Everything You Should Know About It

I wish you all the best for the upcoming challenges.🤞
Keep up the good work! 👊
Stay Udacious🔱!

Threat Model the Microservices Environment

✓

The screenshot security_architecture_design.png contains a line and box diagram showing the relationship between the 4 core components.

Great Work

The provided diagram meets the required specifications. You show all the components and security boundaries between them. ✅

Suggestion

You might find the article useful for further learning 11 Reasons Why You Are Going To Fail With Microservices

✓

Submission should include 5 concrete attack surface areas for the Docker environment and 5 concrete attack surface areas for the Kubernetes control plane, etcd, and worker.

In the explanation, associate each attack surface area to at least one pillar of the STRIDE model, which includes

- **Spoofing**
- **Tampering**
- **Repudiation**
- **Information Disclosure**
- **Denial of Service**
- **Elevation of Privilege**

There can be multiple attack surface areas associated with one pillar, but the attack surface areas have to

be distinct.

Great Work!

You have provide the enough detail for the `Docker` and `Kubernetes` environments. You had selected the 5 conrete attack area for the `docker` and `kubernetes` and apply to the `STIRDE` model.✅

Harden the Microservices Environment

✓

The screenshots for Docker-bench initial run and re-run contain the result summary and all failed Docker-bench findings.

The threat_modeling_template.txt file should contain the 3 failed Docker-bench findings you want to harden. These findings need to be reflected in the screenshot and should be related to the 5 Docker attack surfaces from Step 1.

The screenshots for the Docker-bench re-run show that the 3 findings you identified as needing to be fixed now pass upon the Docker-bench re-run.

Well Done!

You had successfully hardened the failed findings for the docker. The screenshot provided clearly show that after rerunning the **docker bench** the pass findings are increased and fail findings were decreased as expected.✅

Before hardening

```
localhost:/vagrant/docker-bench # ./docker-bench --include-test-output >docker-bench.txt
localhost:/vagrant/docker-bench # cat docker-bench.txt | grep FAIL
[FAIL] 1.1.1 Ensure a separate partition for containers has been created (Automated)
[FAIL] 1.1.3 Ensure auditing is configured for the docker daemon (Automated)
[FAIL] 1.1.4 Ensure auditing is configured for Docker files and directories - /run/containerd (Automated)
[FAIL] 1.1.5 Ensure auditing is configured for Docker files and directories - /var/lib/docker (Automated)
[FAIL] 1.1.6 Ensure auditing is configured for Docker files and directories - /etc/docker (Automated)
[FAIL] 1.1.7 Ensure auditing is configured for Docker files and directories - docker.service (Automated)
[FAIL] 1.1.8 Ensure auditing is configured for Docker files and directories - containerd.sock (Automated)
[FAIL] 1.1.9 Ensure auditing is configured for Docker files and directories - docker.socket (Automated)
[FAIL] 1.1.10 Ensure auditing is configured for Docker files and directories - /etc/default/docker (Automated)
[FAIL] 1.1.11 Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json (Automated)
[FAIL] 1.1.12 Ensure auditing is configured for Docker files and directories - /etc/containerd/config.toml (Automated)
[FAIL] 1.1.13 Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker (Automated)
[FAIL] 1.1.14 Ensure auditing is configured for Docker files and directories - /usr/bin/containerd (Automated)
[FAIL] 1.1.15 Ensure auditing is configured for Docker files and directories - /usr/bin/containerd-shim (Automated)
[FAIL] 1.1.16 Ensure auditing is configured for Docker files and directories - /usr/bin/containerd-shim-runc-v1 (Automated)
[FAIL] 1.1.17 Ensure auditing is configured for Docker files and directories - /usr/bin/containerd-shim-runc-v2 (Automated)
[FAIL] 1.1.18 Ensure auditing is configured for Docker files and directories - /usr/bin/runc (Automated)
[FAIL] 2.2 Ensure network traffic is restricted between containers on the default bridge (Automated)
[FAIL] 2.7 Ensure TLS authentication for Docker daemon is configured (Automated)
[FAIL] 2.9 Enable user namespace support (Automated)
[FAIL] 2.14 Ensure containers are restricted from acquiring new privileges (Automated)
[FAIL] 2.15 Ensure live restore is Enabled (Automated)
[FAIL] 2.16 Ensure Userland Proxy is Disabled (Automated)
[FAIL] 3.3 Ensure that docker.socket file ownership is set to root:root (Automated)
[FAIL] 3.4 Ensure that docker.socket file permissions are set to 644 or more restrictive (Automated)
[FAIL] 3.7 Ensure that registry certificate file ownership is set to root:root (Automated)
[FAIL] 3.8 Ensure that registry certificate file permissions are set to 444 or more restrictive (Automated)
[FAIL] 3.19 Ensure that /etc/default/docker file ownership is set to root:root (Automated)
[FAIL] 3.22 Ensure that /etc/default/docker file permissions are set to 644 or more restrictive (Automated)
[FAIL] 3.23 Ensure that the Containerd socket file ownershi is set to root:root (Automated)
[FAIL] 4.1 Ensure that a user for the container has been created (Automated)
[FAIL] 4.5 Ensure Content trust for Docker is Enabled (Automated)
[FAIL] 5.4 Ensure that privileged containers are not used (Automated)
[FAIL] 5.9 Ensure that the host's network namespace is not shared (Automated)
[FAIL] 5.10 Ensure that the memory usage for container is limited (Automated)
[FAIL] 5.11 Ensure that CPU priority is set appropriately on container (Automated)
[FAIL] 5.12 Ensure that the container's root filesystem is mounted as read only (Automated)
[FAIL] 5.14 Ensure that the 'on-failure' container restart policy is set to '5' (Automated)
[FAIL] 5.15 Ensure that the host's process namespace is not shared (Automated)
[FAIL] 5.20 Ensure that the host's UTS namespace is not shared (Automated)
[FAIL] 5.23 Ensure that docker exec commands are not used with the user=root option (Manual)
[FAIL] 5.25 Ensure that the container is restricted from acquiring additional privileges (Automated)
[FAIL] 5.28 Ensure that the PIDs cgroup limit is used (Automated)
43 checks FAIL
localhost:/vagrant/docker-bench #
```

After hardening

```
localhost:/vagrant/docker-bench #  cat docker-bench4.txt | grep FAIL
[FAIL] 1.1.1 Ensure a separate partition for containers has been created (Automated)
[FAIL] 1.1.3 Ensure auditing is configured for the docker daemon (Automated)
[FAIL] 1.1.4 Ensure auditing is configured for Docker files and directories - /run/containerd (Automated)
[FAIL] 1.1.5 Ensure auditing is configured for Docker files and directories - /var/lib/docker (Automated)
[FAIL] 1.1.6 Ensure auditing is configured for Docker files and directories - /etc/docker (Automated)
[FAIL] 1.1.7 Ensure auditing is configured for Docker files and directories - docker.service (Automated)
[FAIL] 1.1.8 Ensure auditing is configured for Docker files and directories - containerd.sock (Automated)
[FAIL] 1.1.9 Ensure auditing is configured for Docker files and directories - docker.socket (Automated)
[FAIL] 1.1.10 Ensure auditing is configured for Docker files and directories - /etc/default/docker (Automated)
[FAIL] 1.1.11 Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json (Automated)
[FAIL] 1.1.12 Ensure auditing is configured for Docker files and directories - /etc/containerd/config.toml (Automated)
[FAIL] 1.1.13 Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker (Automated)
[FAIL] 1.1.14 Ensure auditing is configured for Docker files and directories - /usr/bin/containerd (Automated)
[FAIL] 1.1.15 Ensure auditing is configured for Docker files and directories - /usr/bin/containerd-shim (Automated)
[FAIL] 1.1.16 Ensure auditing is configured for Docker files and directories - /usr/bin/containerd-shim-runc-v1 (Automat
[FAIL] 1.1.17 Ensure auditing is configured for Docker files and directories - /usr/bin/containerd-shim-runc-v2 (Automat
[FAIL] 1.1.18 Ensure auditing is configured for Docker files and directories - /usr/bin/runc (Automated)
[FAIL] 2.2 Ensure network traffic is restricted between containers on the default bridge (Automated)
[FAIL] 2.7 Ensure TLS authentication for Docker daemon is configured (Automated)
[FAIL] 2.9 Enable user namespace support (Automated)
[FAIL] 3.3 Ensure that docker.socket file ownership is set to root:root (Automated)
[FAIL] 3.4 Ensure that docker.socket file permissions are set to 644 or more restrictive (Automated)
[FAIL] 3.7 Ensure that registry certificate file ownership is set to root:root (Automated)
[FAIL] 3.8 Ensure that registry certificate file permissions are set to 444 or more restrictive (Automated)
[FAIL] 3.19 Ensure that /etc/default/docker file ownership is set to root:root (Automated)
[FAIL] 3.22 Ensure that /etc/default/docker file permissions are set to 644 or more restrictive (Automated)
[FAIL] 4.1 Ensure that a user for the container has been created (Automated)
[FAIL] 5.4 Ensure that privileged containers are not used (Automated)
[FAIL] 5.9 Ensure that the host's network namespace is not shared (Automated)
[FAIL] 5.10 Ensure that the memory usage for container is limited (Automated)
```

```
[FAIL] 5.11 Ensure that CPU priority is set appropriately on container (Automated)
[FAIL] 5.12 Ensure that the container's root filesystem is mounted as read only (Automated)
[FAIL] 5.14 Ensure that the 'on-failure' container restart policy is set to '5' (Automated)
[FAIL] 5.15 Ensure that the host's process namespace is not shared (Automated)
[FAIL] 5.20 Ensure that the host's UTS namespace is not shared (Automated)
[FAIL] 5.23 Ensure that docker exec commands are not used with the user=root option (Manual)
[FAIL] 5.25 Ensure that the container is restricted from acquiring additional privileges (Automated)
[FAIL] 5.28 Ensure that the PIDs cgroup limit is used (Automated)
[FAIL] 5.31 Ensure that the Docker socket is not mounted inside any containers (Automated)
39 checks FAIL
localhost:/vagrant/docker-bench #
```

The screenshots for the Kube-bench initial run and re-run contain the result summary and all failed Kube-bench findings.

The number of failed findings in the Kube-bench re-run is less than that in the Kube-bench initial run.

The `1.1.12` Kube-bench check no longer fails in the Kube-bench re-run after baseline hardening.

The Kubernetes-specific test plan consists of at least 200 words and minimally describes

- How you will test the changes
- How you will ensure the changes don't negatively affect your cluster

Well Done

1.1.12 Kube-bench check no longer fails in the Kube-bench re-run after baseline hardening.☑

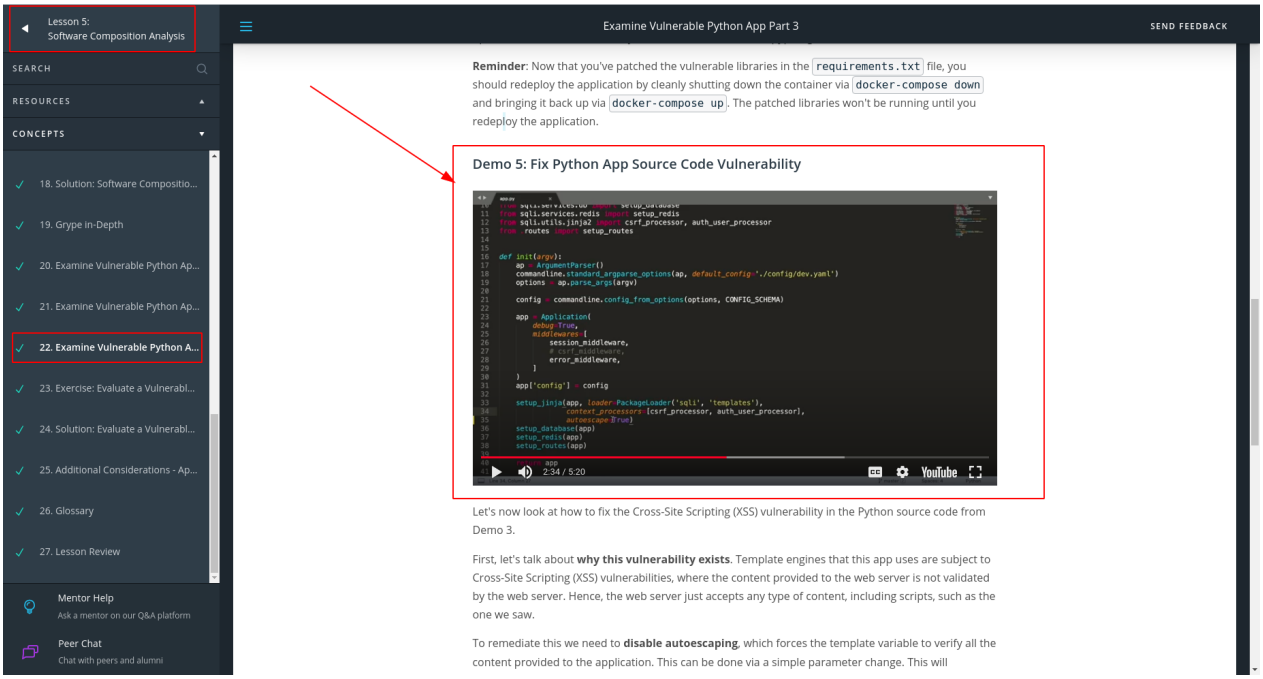
You had provided detailed information for the applied changes and ensured that changes are not negatively affected the cluster.☑

Harden and Deploy the Flask App

The submission includes the `app.py` file with changes that fix the Cross-Site Scripting vulnerability.

Action Required

As per the requirement, you have to fix the Cross-Site Scripting vulnerability for the app.py. However, in your current submission, you didn't fix the Cross-Site Scripting vulnerability. Please make sure to share updated the `app.py` in your next submission. If you want you can revisit `Lesson 5` Examine Vulnerable Python App Part 3 watch **Demo 5: Fix Python App Source Code Vulnerability** video to learn the **Cross-Site Scripting vulnerability**.



✓

The screenshot grype_app_out_of_box.png contains the Grype output.

The screenshot grype_app_hardended.png shows 0 findings for vulnerable libraries.

Well Done

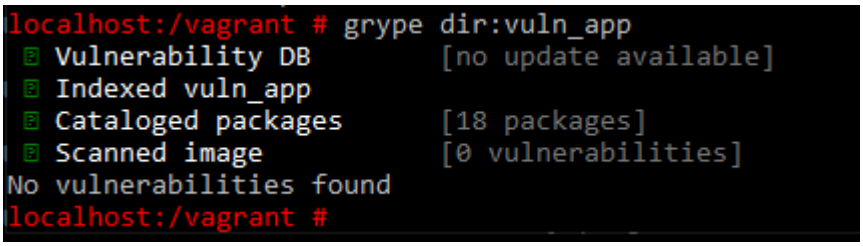
You had successfully updated the dependency to the latest version to **fix the libraries-based vulnerabilities**.
☑

Before hardening

```
localhost:/vagrant # grype dir:vuln_app
 ✔ Vulnerability DB        [updated]
 ✔ Indexed vuln_app
 ✔ Cataloged packages      [18 packages]
 ✔ Scanned image           [5 vulnerabilities]

NAME     INSTALLED  FIXED-IN  VULNERABILITY        SEVERITY
aiohttp  3.5.3      3.7.4     GHSA-v6wp-4m6f-gcjg  Low
jinja2   2.10       2.11.3    GHSA-g3rq-g295-4j3m  Medium
pyyaml   3.13       5.3.1     GHSA-6757-jp84-gxfx  High
pyyaml   3.13       4.1       GHSA-rprw-h62v-c2w7  Critical
pyyaml   3.13                 CVE-2017-18342       Critical
localhost:/vagrant #
```

After hardening



Suggestion

You might find the article useful for further learning Vulnerability scanning for Docker local images

Implement Grafana and run-time monitoring

✓

The screenshot kube_pods_screenshot.png shows Falco and falco-exporter pods running.
The screenshot falco_alert_screenshot.png from Falco pod logs or from the falco-exporter metrics page shows that Falco is generating security events.

Good job

You had done a good job by deploying `Falco` and `Falco-exporter` to generate alerts.✅

Suggestion

You might find the article useful for further learning Kubernetes Security monitoring at scale with Sysdig Falco

✓

In the screenshot falco_grafana_screenshot.png, the Falco Grafana panel should display graphs indicating the sensitive file being read.

Good Job

You had successfully gotten the Falco logs into Grafana as expected.✅

Incident Response

In the incident_response_report.txt, provide an incident response report that answers all these questions (at least two sentences for Questions 2-6):

1. Incident commander name
2. Executive summary of what happened
3. Summary of what was affected
4. Summary of the impact
5. Summary of the remediation steps
6. Summary of lessons learned

The report should be free of grammar and spelling mistakes.

1. Incident commander name ☑
2. Executive summary of what happened ✗

Here you have to write the pods name that was running during the incident. This will help the concerned team to mitigate issues in the future.

3. Summary of what was affected ☑
4. Summary of the impact ✗

Please describe more about the impact that can cause the origination business.

5. Summary of the remediation steps ☑
6. Summary of lessons learned ☑

RESUBMIT

DOWNLOAD PROJECT

Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)

RETURN TO PATH