NYU

CS-GY 9163

Application Security

Fall 2023

Assignment 3

by

Andrei Choulga

ac8978@nyu.edu

https://www.linkedin.com/in/ac317

connect with me

github repository

https://github.com/ac8978/AppSecAssignment3.1

If any of the parts were not clear or not explained in details, please reach out to me via Slack

I work with Docker and Kubernetes and there are tools that would make this assignment easier, but I stuck to the basics.

Thank you.

Notes: all line # references are to the original unmodified file

TABLE OF CONTENTS

PART 1:

as mentioned in the instructions:

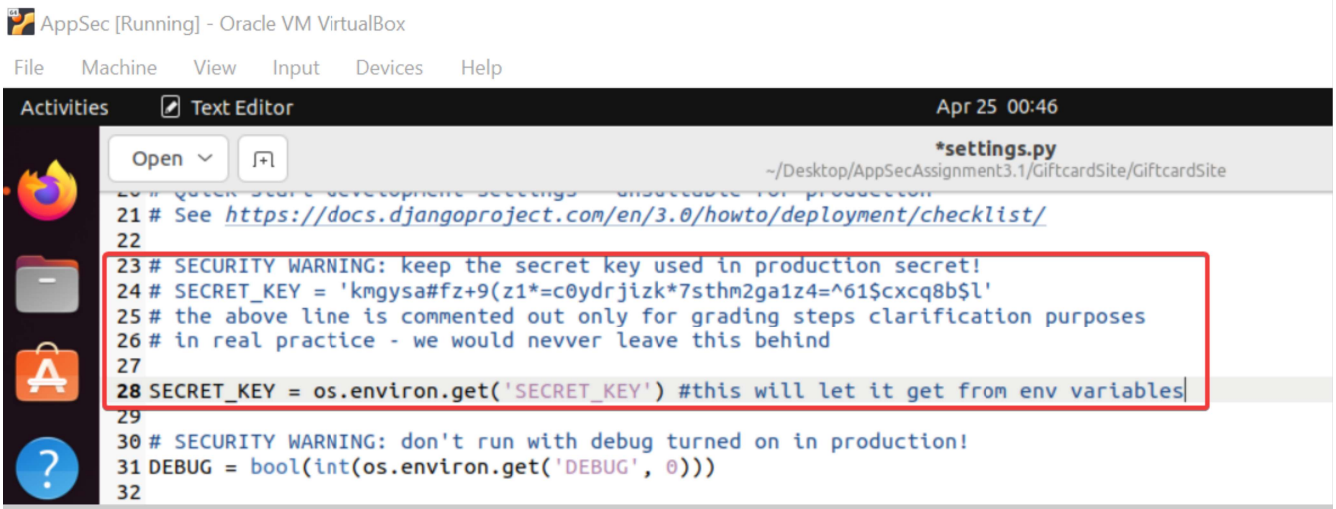
>For this portion of the assignment, you should submit:

>All kubernetes yaml files modified to use secrets

>All changes necessary to the Web application (limited to settings.py as mentioned above) needed to use the passed secrets.

The very first file “settings.py” as in the previous assignment – had a hardcoded secret

we move it out by - line is commented out only for grading steps clarification purposes in real practice - we would never leave this behind




```
# SECURITY WARNING: keep the secret key used in production secret!

# SECRET_KEY = 'kmgysa#fz+9(z1*=c0ydrjizk*7sthm2ga1z4=^61$cxcq8b$l'

# the above line is commented out only for grading steps clarification
purposes

# in real practice - we would never leave this behind



SECRET_KEY = os.environ.get('SECRET_KEY') #this will let it get from env
variables
```

I see that the files

./GiftcardSite/k8/django-deploy.yaml

./db/k8/db-deployment.yaml

/db/k8s/db-deployment.yaml

all have secrets in them

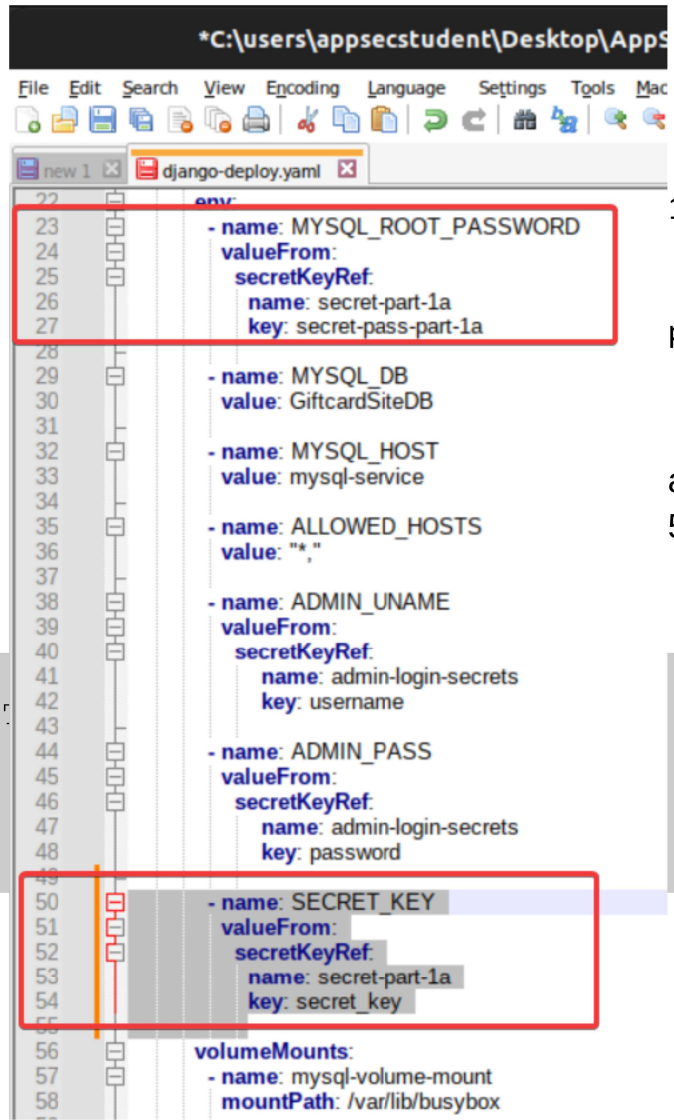
in

GiftcardSite/k8/django-deploy.yaml

on line 24 we remove

value: thisisatestthing.

We add:



valueFrom:

secretKeyRef:

name: secret-part-1a

key: secret-pass-part-1a

and we also add at line 50

```
- name: SECRET
```

In

db/k8/db-deployment.yaml

we remove

- name: MYSQL_ROOT_PASSWORD

value: thisisatestthing.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql-container
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql-container
  template:
    metadata:
      labels:
        app: mysql-container
        tier: backend
    spec:
      containers:
      - name: mysql-container
        image: nyuappsec/assign3-db:v0
        env:
        # - name: MYSQL_ROOT_PASSWORD
        #   value: thisisatestthing.
        - name: secret-part-1a
          value: db-password

        - name: MYSQL_DATABASE
          value: GiftcardSiteDB

        ports:
        - containerPort: 3306
        volumeMounts:
        - name: mysql-volume-mount
          mountPath: /var/lib/mysql
```

We add

- name: secret-part-1a

value: db-password

I created a file to store the secrets as

secret-part-1a.yaml

```
apiVersion: v1
kind: Secret
metadata:
    name: secret-part-1a
type: Opaque
data:
    username: YWRtaW4=
    password: dGhpc2lzYXRlc3R0aGluZy4=
    secret_key:
a21neXNhI2Z6KzkoejEqPWMweWRyaml6ayo3c3RobTJnYTF6ND1eNjEkY3hjcTh
iJGw=
```

I apply these secrets sotored in this file

% kubectl apply -f secret-part-1a.yaml

```
secret-part-1a.yaml.txt
appsecstudent@appsecstudent-VirtualBox:~/Desktop/AppSecAssignment3.1/part1$ kubectl apply -f secret-part-1a.yaml
secret/secret-part-1a created
```

we run

% kubectl get secrets

to ensure they are applied

```
appsecstudent@appsecstudent-VirtualBox:~/Desktop/AppSecAssignment3.1/part1$ kubectl get secrets
NAME                  TYPE     DATA   AGE
admin-login-secrets   Opaque   2      25h
secret-part-1a        Opaque   3      85s
securedsecrets        Opaque   3      99m
```

we get

```
appsecstudent@appsecstudent-VirtualBox:~/Desktop/AppSecAssignment3.1/part1$ kubectl get secrets

NAME                  TYPE     DATA   AGE

admin-login-secrets   Opaque   2      25h

secret-part-1a        Opaque   3      85s

securedsecrets        Opaque   3      99m
```

these are good references for info on what / why / how

https://analyticsindiamag.com/how-to-manage-secrets-in-kubernetes/

https://github.com/jetstack/kubernetes.github.io/blob/master/docs/concepts/configuration/secret.md

https://yashbindlish.medium.com/securing-a-distributed-platform-kubernetes-secret-management-472cc85fa41b

also:

> *>"Environment variables are readily visible within their containers, within Kubernetes logs, in the manifests, and to anyone on the cluster with the ability to view the spec. If an attacker gains access to any of these, they will have access to the database's root password as well. Sub-optimal, to say the least!*
>
> *Instead of configuring a password directly through an environment variable, we can use KubernetesSecrets. While they aren't the be-all and end-all of Kubernetes security, Secrets give us a way to add an additional layer of protection for sensitive data."*
>
> https://www.mirantis.com/blog/cloud-native-5-minutes-at-a-time-using-kubernetes-secrets-with-environment-variables-and-volume-mounts/

secrets can be held as either plaintext or base64-encoded data, base64 being a bit more secure, preventing someone from behind you glancing at the screen and reading the password as easily, but this encoding and not encryption, so assignment doesn't call for base64 step, which I think will not deter a real attacker.

following command creates a secret:

% kubectl create secret generic secret1 –from-literal=password=thisisatestthing.

named: “secret1”

password: “thisisatestthing.”

we can edit the secret using

% kubectl edit secrets test-secret

we apply the secured secrets file by running

% kubectl apply -f secret1.yaml

we run the

% kubectl get secrets

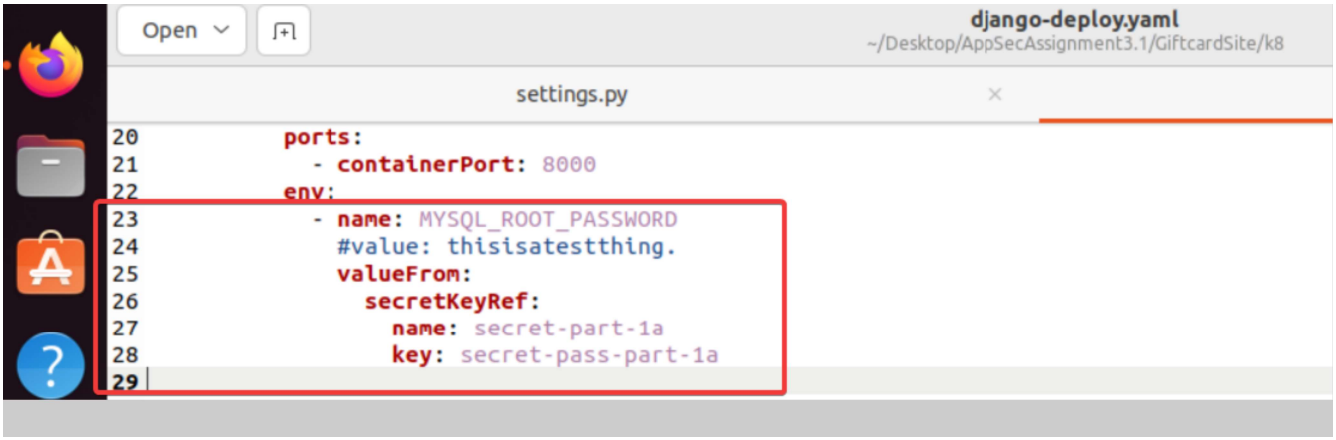
to confirm that **secret1** are applied

I made a separate yaml file to store these secrets and linked them back using the “valueFrom”

IN PRACTICE THIS WOULD NEVER BE UPLOADED TO REPOSITORY

These files are included only to make it easier to understand and grade

Shows django-deploy.yaml changes



I placed the secrets in

secure.yaml

```
apiVersion: v1
kind: Secret
metadata:
    name: securedsecrets
type: Opaque
data:
    username: YWRtaW4=
    password: dGhpc2lzYXRlc3R0aGluZy4=
    secret_key:
a21neXNhI2Z6KzkoejEqPWMweWRyaml6ayo3c3RobTJnYTF6ND1eNjEkY3hjcTh
iJGw=
```

In settings.py

/GiftcardSite/GiftcardSite/settings.py

https://github.com/ac8978/AppSecAssignment3.1/blob/master/GiftcardSite/GiftcardSite/settings.py

I deleted the pods so that the settings will refresh for all the containers

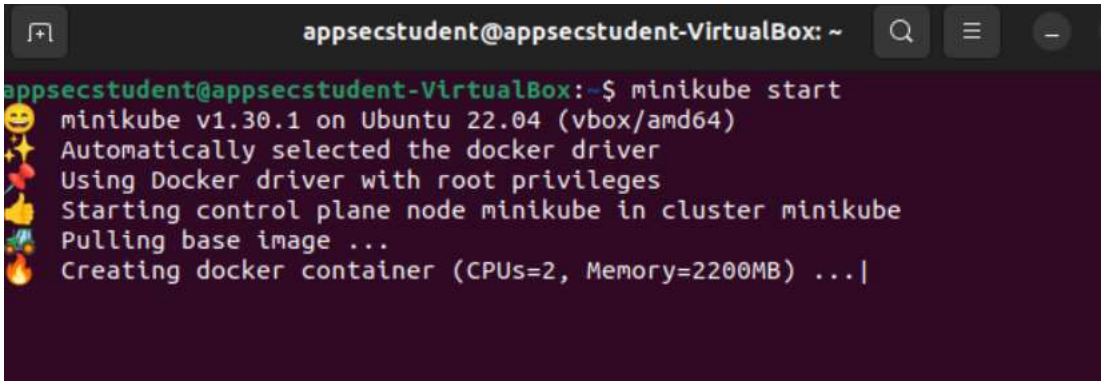
using >kubectl delete pod [pod name]

rebuild the docker and run the updated 3 .yaml:

% minikube delete

- needed to restart the VM -

% minikube start



% eval $(minikube docker-env)

cd ~/Desktop/AppSecAssignment3.1

% docker build -t nyuappsec/assign3:v0 .

% docker build -t nyuappsec/assign3-proxy:v0 proxy/

% docker build -t nyuappsec/assign3-db:v0 db/

% kubectl apply -f db/k8

% kubectl apply -f GiftcardSite/k8

% kubectl apply -f ~/Desktop/AppSecAssignment3.1/part1/secret-part-1a.yaml

% kubectl apply -f db/k8/db-deployment.yaml

% kubectl apply -f db/k8s/db-deployment.yaml

% kubectl apply -f GiftcardSite/k8/django-deploy.yaml

% kubectl apply -f proxy/k8

% kubectl get pods

% kubectl get service

then we run

% minikube service proxy-service

critical note - the git repository contains a secret file only for grading we would never do this in practice

REFERENCES/EXTRA NOTES – you may skip this

also note we could use Heroku

we would also add .env to .gitignore file

to .env we add

SECRET_KEY=KEY

inside of settings.py file, add the following settings:

```
from decouple import config

SECRET_KEY = config('SECRET_KEY')
```

now secret key is successfully stored locally

we can use Heroku for our Django project

now we would

heroku config:set SECRET_KEY=yoursecretkey

(we can also use heroku dashboard)

Reference:

https://docs.djangoproject.com/en/4.2/howto/deployment/checklist/#secret-key

Where to store secret keys DJANGO https://stackoverflow.com/questions/15209978/where-to-store-secret-keys-django

---

PART 2

need to submit:

One yaml file for the migrations job

One yaml file for the database seeding job

Any Dockerfiles you used for these jobs (in separate, descriptively named folders)

Any code you wrote to perform database seeding.

A jobs.txt file that describes what you did in this section.

reference/I read a lot at:

Running database migrations when deploying to Kubernetes https://andrewlock.net/deploying-asp-net-core-applications-to-kubernetes-part-7-running-database-migrations/

also **Migration and Seeding in Django** https://medium.com/@ardho/migration-and-seeding-in-django-3ae322952111

and **Django Automatic Migration and Seeding** https://medium.com/kami-people/django-automatic-migration-and-seeding-e9978788aa66

and **How to Create Django Data Migrations** https://simpleisbetterthancomplex.com/tutorial/2017/09/26/how-to-create-django-data-migrations.html

I made

migration.yaml

```yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: migration
spec:
  template:
    spec:
      containers:
      - name: migration
        image: nyuappsec/assign3_part_2_db_seeding:v0
        command: ['python3', 'manage.py', 'migrate']
        env:
            - name: MYSQL_ROOT_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: secret-part-1a
                  key: password

            - name: MYSQL_DB
              value: GiftcardSiteDB

            - name: MYSQL_HOST
              value: mysql-service

            - name: ALLOWED_HOSTS
              value: "*,"

            - name: ADMIN_UNAME
              valueFrom:
                secretKeyRef:
                    name: secret-part-1a
                    key: username

            - name: ADMIN_PASS
              valueFrom:
                secretKeyRef:
                    name: securedsecrets
                    key: password

            - name: secret-key
              valueFrom:
```

```
                secretKeyRef:
                    name: secret-part-1a
                    key: secret_key
      restartPolicy: Never
```

after I execute it with

% kubectl apply -f migration.yaml

I verified with

% kubectl get jobs

for the seeding job

seedjob.yaml

```
apiVersion: batch/v1
kind: Job
metadata:
  name: seeding
spec:
  template:
    spec:
      containers:
      - name: seeding
        image: 'nyuappsec/seeding:v0'
        command: ["/bin/sh"]
        args: ["-c","mysql --user=root --password=$
{MYSQL_ROOT_PASSWORD} --database=${MYSQL_DATABASE} --
host=mysql-service -f <
/docker-entrypoint-initdb.d/seedjob.sql"]
        env:
          - name: MYSQL_ROOT_PASSWORD
            valueFrom:
              secretKeyRef:
                name: secret-part-1a
                key: password
```

```
          - name:  MYSQL_DATABASE
            value: GiftcardSiteDB

      restartPolicy: Never
  backoffLimit: 7
```

I had to make

Seed.sql

```
LOAD DATA INFILE '/products.csv' INTO TABLE LegacySite_product
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '\"' LINES
TERMINATED BY '\r\n';
LOAD DATA INFILE '/users.csv' INTO TABLE LegacySite_user FIELDS
TERMINATED BY ',' OPTIONALLY ENCLOSED BY '\"' LINES TERMINATED
BY '\r\n';
```

Dockerfile

```
FROM mysql:latest

COPY ./products.csv /products.csv
COPY ./users.csv /users.csv
COPY ./setup.sql /docker-entrypoint-initdb.d/setup.sql
COPY ./seed.sql /docker-entrypoint-initdb.d/seed.sql

ENTRYPOINT ["/startseeding.sh"]
CMD ["mysqld", "--secure-file-priv=/data"]
```

I also started and used

seed-start.sh

```
#!/bin/sh

mysql -uroot -p$MYSQL_ROOT_PASSWORD --local-infile $MYSQL_DB -e
```

```
"LOAD DATA INFILE '/products.csv' INTO TABLE LegacySite_product
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '\"' LINES
TERMINATED BY '\r\n';"
mysql -uroot -p$MYSQL_ROOT_PASSWORD --local-infile $MYSQL_DB -e
"LOAD DATA INFILE '/users.csv' INTO TABLE LegacySite_user
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '\"' LINES
TERMINATED BY '\r\n';"a
```

I executed

% kubectl apply -f seeding.yaml

after I ran

% kubectl get jobs

to confirm migration and seeding to be done

% kubectl get pods

% kubectl get service

% minikube service proxy-service

References:

Django : How to seed Django project ? - insert a bunch of data … https://www.youtube.com/watch?v=ut6VDXZzbJQ

How to seed Django project ? - insert a bunch of data into the project for initialization https://copyprogramming.com/howto/how-to-seed-django-project-insert-a-bunch-of-data-into-the-project-for-initialization

How to provide initial data for models https://docs.djangoproject.com/en/4.2/howto/initial-data/

PART 3

Part 3.1: Remove unwanted monitoring.

In GiftcardSite/LegacySite/views.py file

In GiftcardSite/LegacySite/views.py line 48 I see a comment:

#KG: Uh... I'm not sure this makes sense.
Collect data to ensure good password use.

/GiftcardSite/LegacySite/views.py line 48

```
        # KG: Uh... I'm not sure this makes sense.
        # Collect data to ensure good password use.
        #part 3 - removing all of this code - it is exposing a password to debug logs
        #if pword not in graphs.keys():
        #   graphs[pword] = Counter(f'counter_{pword}', 'The total number of '\
        #     + f'times {pword} was used')
        #graphs[pword].inc()
```

I removed this recording a password data is a serious security / privacy issue

I see that we are also monitoring number of logins

/GiftcardSite/LegacySite/views.py

```
graphs['l_counter'] = Counter('python_request_l_posts', 'The
total number' + ' of login posts.')
```

at first I wanted to removed this, and considered more with research such as

>An administrator should regularly check user logins to see if any unusual activity has occurred. This includes monitoring login attempts from unfamiliar locations or devices, or multiple failed login attempts from the same user https://www.threatkey.com/guides/okta-monitor-suspicious-login-attempts

thus it should stay, as it is justifiable

Part 3.2: Expand reasonable monitoring.

The assignment calls for

>"In this part of the assignment you should add a Prometheus counter that counts all the times we purposely return a 404 message in views.py. These lines are caused by Database errors, so you should name this counter database_error_return_404."

therefore I added

I did add this in /GiftcardSite/LegacySite/views.py line 19

```
graphs['database_error_return_404'] =
Counter('python_return_error', 'The total number'  + ' of
errors returned') #for part 3 added logging for error messages
```

I added a DB error counter in six of appropriate places of GiftcardSite/LegacySite/views.py

```
graphs['database_error_return_404'].inc() #part 3 - database error return counter
```

below shows where I added the code:

```
AppSec [Running] - Oracle VM VirtualBox
File  Machine  View  Input  Devices  Help
Activities  Terminal  Apr 25 20:53  100 %
appsecstudent@appsecstudent-VirtualBox: ~/Desktop/AppSecAssignment3.1
        assert (None not in [uname, pword, pword2])
        if pword != pword2:
@@ -98,11 +105,13 @@ def buy_card_view(request, prod_num=0):
            try:
                prod = Product.objects.get(product_id=prod_num)
            except:
+               graphs['database_error_return_404'].inc() #part 3 - database error return counter
                return HttpResponse("ERROR: 404 Not Found.")
        else:
            try:
                prod = Product.objects.get(product_id=1)
            except:
+               graphs['database_error_return_404'].inc() #part 3 - database error return counter
                return HttpResponse("ERROR: 404 Not Found.")
        context['prod_name'] = prod.product_name
        context['prod_path'] = prod.product_image_path
@@ -148,11 +157,13 @@ def gift_card_view(request, prod_num=0):
            try:
                prod = Product.objects.get(product_id=prod_num)
            except:
+               graphs['database_error_return_404'].inc() #part 3 - database error return counter
                return HttpResponse("ERROR: 404 Not Found.")
        else:
            try:
                prod = Product.objects.get(product_id=1)
            except:
+               graphs['database_error_return_404'].inc() #part 3 - database error return counter
                return HttpResponse("ERROR: 404 Not Found.")
        context['prod_name'] = prod.product_name
        context['prod_path'] = prod.product_image_path
@@ -165,6 +176,7 @@ def gift_card_view(request, prod_num=0):
            prod_num = 1
        user = request.POST.get('username', None)
        if user is None:
+           graphs['database_error_return_404'].inc() #part 3 - database error return counter
            return HttpResponse("ERROR 404")
        try:
            user_account = User.objects.get(username=user)
@@ -250,6 +262,7 @@ def use_card_view(request):
            user_cards = None
        context['card_list'] = user_cards
        return render(request, "use-card.html", context)
+       graphs['database_error_return_404'].inc() #part 3 - database error return counter
    return HttpResponse("Error 404: Internal Server Error")

 def metrics_view(request):
diff --git a/part3/Prometheus.txt b/part3/Prometheus.txt
new file mode 100644
index 0000000..9c4523b
Shift + Ctrl
```

Part 3.3: Add Prometheus

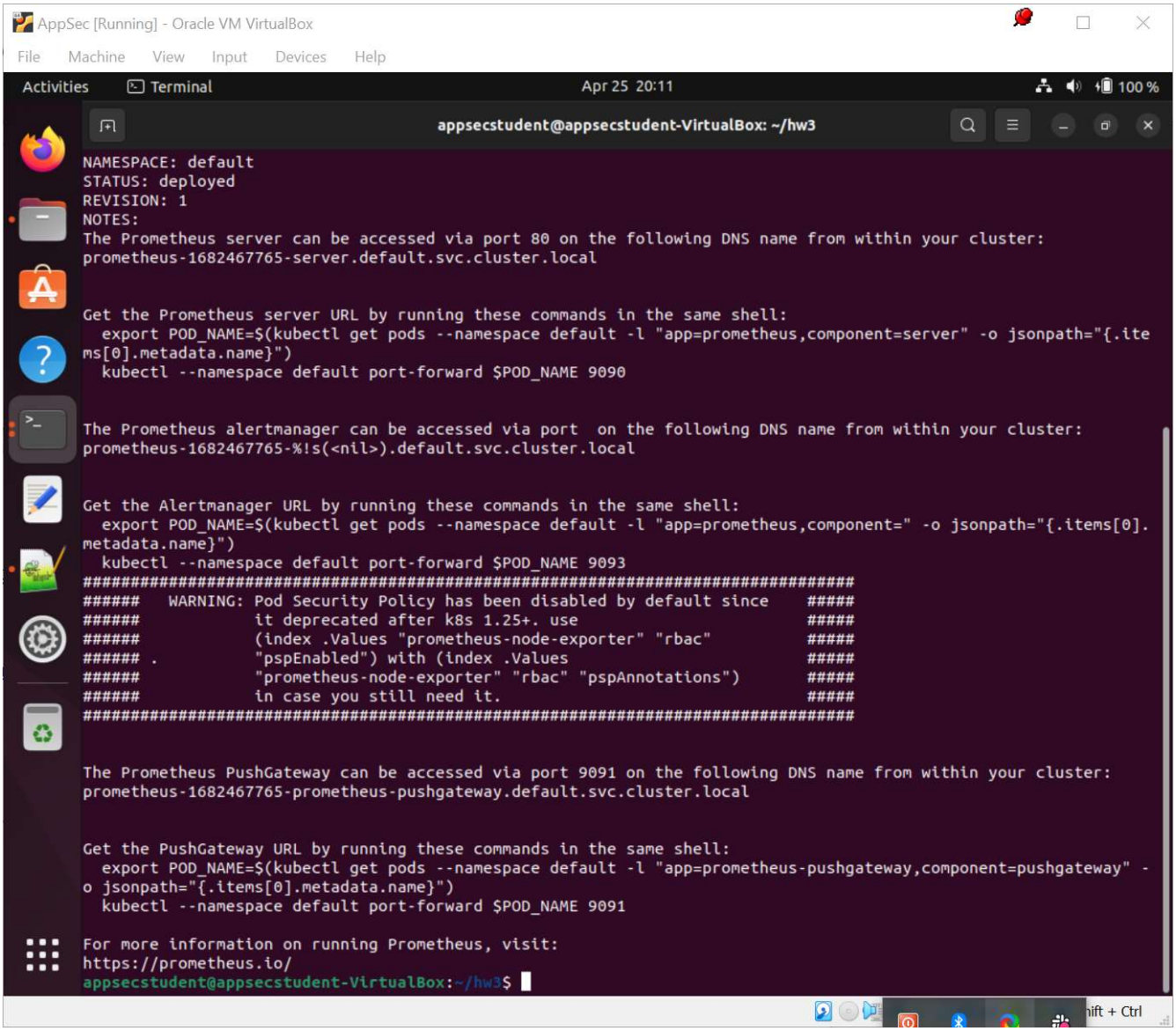
I read

https://helm.sh/docs/intro/install/

and

https://artifacthub.io/packages/helm/prometheus-community/prometheus

appsecstudent@appsecstudent-VirtualBox:~/hw3$ sudo snap install helm –classic



We make sure it is running via
% **kubectl get pods**

```
appsecstudent@appsecstudent-VirtualBox:~/hw3$ kubectl get pods
NAME                                                      READY   STATUS              RESTARTS   AGE

prometheus-1682467765-alertmanager-0                      0/1     ContainerCreating   0          2m20s

prometheus-1682467765-kube-state-metrics-54cbbf6cdf-kzc9t 0/1     ContainerCreating   0          2m21s

prometheus-1682467765-prometheus-node-exporter-c8jjg      1/1     Running             0          2m21s

prometheus-1682467765-prometheus-pushgateway-5bbc8f6fd7-64gj5 1/1 Running             0          2m21s

prometheus-1682467765-server-996765d55-n6xxs              0/2     ContainerCreating   0          2m21s
```

We list what is running

% minikube service list

We then can edit

% kubectl edit cm prometheus-1682467765-server

by adding

- proxy-server:8080

on line 27

We then start the service and query

% minikube service prometheus-1682467765-server

AppSec [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Terminal Apr 25 20:21 100 %

appsecstudent@appsecstudent-VirtualBox: ~/hw3

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  alerting_rules.yml: |
    {}
  alerts: |
    {}
  allow-snippet-annotations: "false"
  prometheus.yml: |
    global:
      evaluation_interval: 1m
      scrape_interval: 1m
      scrape_timeout: 10s
    rule_files:
    - /etc/config/recording_rules.yml
    - /etc/config/alerting_rules.yml
    - /etc/config/rules
    - /etc/config/alerts
    scrape_configs:
    - job_name: prometheus
      static_configs:
      - targets:
        - localhost:9090
        - proxy-server:8080
      bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
      job_name: kubernetes-apiservers
      kubernetes_sd_configs:
      - role: endpoints
      relabel_configs:
      - action: keep
        regex: default;kubernetes;https
        source_labels:
        - __meta_kubernetes_namespace
        - __meta_kubernetes_service_name
        - __meta_kubernetes_endpoint_port_name
      scheme: https
      tls_config:
        ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
        insecure_skip_verify: true
-- INSERT --                                                    27,28          Top
```

Shift + Ctrl

after some time I realized I made a mistake

of

proxy-server:8080

vs

proxy-service:8080

AppSec [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Terminal Apr 25 20:32 100 %

appsecstudent@appsecstudent-VirtualBox: ~/hw3

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  alerting_rules.yml: |
    {}
  alerts: |
    {}
  allow-snippet-annotations: "false"
  prometheus.yml: |
    global:
      evaluation_interval: 1m
      scrape_interval: 1m
      scrape_timeout: 10s
    rule_files:
    - /etc/config/recording_rules.yml
    - /etc/config/alerting_rules.yml
    - /etc/config/rules
    - /etc/config/alerts
    scrape_configs:
    - job_name: prometheus
      static_configs:
      - targets:
        - localhost:9090
        - proxy-service:8080
    - bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
      job_name: kubernetes-apiservers
      kubernetes_sd_configs:
      - role: endpoints
      relabel_configs:
      - action: keep
        regex: default;kubernetes;https
        source_labels:
        - __meta_kubernetes_namespace
        - __meta_kubernetes_service_name
        - __meta_kubernetes_endpoint_port_name
      scheme: https
      tls_config:
        ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
        insecure_skip_verify: true
:wq
```

Shift + Ctrl

as it is in

proxy/k8/proxy-service.yaml

```yaml
kind: Service
apiVersion: v1
metadata:
  name: proxy-service
spec:
  selector:
    pod: proxy
  ports:
  - protocol: TCP
    port: 8080
    targetPort: 8080
  type: NodePort
```

we run

% minikube service –all

this will let us find the prometheus server

then I did

% kubectl expose service prometheus-1682467765-server --type=NodePort --target-port=9090 –name=prometheus-server-np

to open the port of prometheus server to its appropriate port

kubectl get configmap prometheus-1682467765-server -o yaml > promethus-server-update.yaml

it is in /part3/promethus-server-update.yaml

if you have any comments, questions, suggestions please reach out to me via Slack, if any of the parts were not clear or not explained in details, please reach out to me via Slack

Thank you.