Overview

In this assignment, the main tasks will be to:

- deploy an nginx service in a Kubernetes cluster
- monitor the service's health using Prometheus
- create a Grafana dashboard with the metrics gathered by Prometheus
- perform some attacks against the nginx service
- observe the graphs in Grafana during and after the attacks
- (bonus) propose some security actions that will harden the nginx webserver and improve the overall infrastructure

Last update: ~~06.05.2022~~ 18.05.2022

Homework Setup

For this homework, you will work on a local Linux virtual machine where you will have to deploy your work infrastructure. Do not work on VMs in OpenStack since the resources are limited and they are shared with others.

Task 0. Create the Kubernetes cluster

Deploy a local, single-node Kubernetes cluster, using Kind. See the Kubernetes lab for instructions.

Task 1. Deploy the nginx service

Similar to the Kubernetes lab, deploy an nginx service that will be exposed on port **80** inside the cluster and port **30080** outside the cluster**.** The service will be named "**nginx**".
You can choose the content that is served (index.html) at your discretion.

Additionally, the nginx server will have to provide metrics about itself, on port **8080**, location **/metrics.** Use the **stub_status** module:
http://nginx.org/en/docs/http/ngx_http_stub_status_module.html

Attention: You must serve the metrics on port 8080, not on the same port 80 as the html content!

Expose the metrics endpoint via the same **nginx** Kubernetes service, on port **8080** inside the cluster and port **30088** outside the cluster**.**

Task 2. Deploy a prometheus exporter

The metrics exposed by nginx via the **stub_status** module are not compatible with Prometheus. To be able to use Prometheus for monitoring, we must use a prometheus exporter, which is a simple application that reads metrics and translates them to the prometheus format.

For nginx, a popular prometheus exporter is **nginx-prometheus-exporter**:
https://github.com/nginxinc/nginx-prometheus-exporter
Deploy nginx-prometheus-exporter in the Kubernetes cluster, using a Kubernetes deployment. Take a look at the “docker run” command from the README.md file to figure out how you should configure your deployment.

Hint: use “**args**” for the container configuration.

Note: the URL that will be monitored should be **http://nginx:8080/metrics**

Expose nginx-prometheus-exporter via a new Kubernetes service, on port **9113** inside the cluster. The service will be named “**promexporter**”.

Note: Exposing the prometheus exporter outside the cluster is not mandatory, but can be helpful for debugging purposes.

Note: Using Helm for deploying Prometheus and Grafana

For the following tasks, you will have to deploy the Prometheus and Grafana monitoring tools in Kubernetes. There are several ways to do that, the main two being:

- Manually creating manifests (for deployments, services, volumes etc.)
- Using Helm (https://helm.sh/), which is a package manager for Kubernetes.

Helm is the method which has the most documentation available, but you can choose any method you want, as long as you are able to deploy Prometheus and Grafana in Kubernetes.

In the tasks, we will provide info about the Helm charts that you can use, should you opt for this method.

Also, if you choose to use Helm, the first step is to install the helm tool on your VM. See the user guide: https://helm.sh/docs/intro/install/

Task 3. Deploy Prometheus

Prometheus must be deployed in a separate Kubernetes namespace, called "**monitoring**". This namespace does not exist, so you must create it.

For deploying Prometheus, you can use Helm. To do this, you can use the helm chart on your cluster: https://github.com/prometheus-community/helm-charts/tree/main/charts/prometheus

Attention: Make sure to deploy the helm chart to the **monitoring** namespace.

After the helm chart is deployed, use **kubectl port-forward** to forward the port of the **prometheus-server** service to your VM.

Connect to the Prometheus UI using a browser (see https://scgc.pages.upb.ro/cloud-courses/docs/basic/working_with_openstack#connecting-using-an-ssh-jump-host-proxy if you connect to the virtual machine using SSH).

In the UI, go to the **/targets** endpoint to see what Prometheus is already monitoring.

Now, configure Prometheus to monitor metrics exposed by the **promexporter** metrics in the **default** namespace. There are several ways to do that. You can choose any method you want.
Examples:

- Customize **values.yaml** and redeploy the helm chart (see the Configuration section in https://github.com/prometheus-community/helm-charts/blob/main/charts/prometheus/README.md)
- Directly edit the **prometheus-server** configmap (an example can be found here: https://sysdig.com/blog/kubernetes-monitoring-prometheus/)

Hint: You will have to use the FQDN for specifying the hostname that Prometheus must monitor (promexporter.default.svc.cluster.local:9113)

Confirm that the configuration is successful by accessing **/targets** in your browser. Also, go to **Graph** and query a metric, like "nginx_connections_accepted". Perform requests on the nginx server and verify that the graph is updating.

Task 4. Grafana

Next, you will install Grafana which consumes Prometheus metrics and displays dashboards, which are much more comprehensive than Prometheus graphs.

You can Install Grafana also using a helm chart:
https://docs.bitnami.com/kubernetes/infrastructure/grafana/get-started/install/

Attention: Make sure to deploy the helm chart to the **monitoring** namespace.

After the helm chart is deployed, use **kubectl port-forward** to forward the port of the **grafana** service to your VM.

Connect to the Grafana UI using a browser (see https://scgc.pages.upb.ro/cloud-courses/docs/basic/working_with_openstack#connecting-using-an-ssh-jump-host-proxy if you connect to the virtual machine using SSH).

In the Grafana UI, configure a Prometheus data source, specifying the URL of the Prometheus server deployed in the same namespace.

Hint: The URL should be http://prometheus-server.

Import the Grafana dashboard provided by nginx-prometheus-exporter, by following the instructions from here: https://github.com/nginxinc/nginx-prometheus-exporter/tree/main/grafana

Perform requests on the nginx server and verify that the dashboard is updating.

Task 5. Stress Test/Denial of Service

When you have your nginx + Prometheus + Grafana setup up and running, check how the graphs change when you have a high load on your web server compared to when there is a light load. To do this, run some Denial-of-Service (DoS) attacks against the nginx web server using a tool such as slowhttptest (https://github.com/shekyan/slowhttptest) or any other tool you find suitable.

If you are using slowhttptest, you can find a tutorial on how it can be used here: https://github.com/shekyan/slowhttptest/wiki/InstallationAndUsage#usage.

See how things change in the Grafana Dashboard. Take a few screen captures (before, during and after the attack), add them to a document and briefly present some conclusions you can reach by observing the Grafana Dashboard (max half page).

Task 6. Bonus (Security)

For the nginx-Prometheus-Grafana setup you have implemented, propose what security-related actions should be added to secure your web server and your infrastructure.

Homework submission:

To submit your homework, you have to upload on Moodle a **zip** archive named "**SCGC - <Your LDAP username here>.zip**" (e.g. "SCGC - ana.popescu3342.zip" that contains the following files:

- The Manifests you used to deploy your Kubernetes containers.
- A screenshot with each solved task.
- A write-up on how your infrastructure should be reproduced (any ramp-up scripts work as well) and tested.
- A pdf file containing a write-up, screenshots for all solved tasks and the written text for tasks 5 and 6. This pdf file must be named "**Proposed Solution - <Your LDAP username here>.pdf**" (e.g. "Proposed Solution - ana.popescu3342.pdf".
- A README file.

The **deadline** for submitting your zip archive is the 29th of May, 2022, 23:55.

~~Homework presentation~~

~~The homework **must** be presented during the last laboratory (31st of May, 2022). You will have to briefly present what and how you have set up for your infrastructure and demonstrate that it works.~~

~~The homework presentation is **mandatory**. You will not receive a grade on your homework if you do not present it.~~

Grading

- Task 0 - 5p
- Task 1 - 15p
- Task 2 - 15p
- Task 3 - 20p
- Task 4 - 20p
- Task 5 - 15p
- Bonus - up to 20p
- ~~Presentation~~Submission quality (explanations, ramp-up scripts etc.) - 10p