DevOpsInsiders

Interview Preparation Guide

Introduction

This guide helps you to answer questions similar to:

- What have you done in the past that you are proud of?
- What challenges have you faced in the past?
- What was the last issue that you solved?
- In which technology do you think you excel, and why?
- Can you provide an example of an issue you faced in your last project?

Virtual Machines

Task - There was one requirement to shut down virtual machines in the development environment during off hours that is after 6 pm and before 8 am and saturday-sunday off.

Solution - To achieve this I have added tags to all the virtual machine “VM”: “Shutdown” using Terraform

After, I have written a Powershell script. This PowerShell script uses az cli to list all the virtual machine which has tags “VM”: “Shutdown” and stored in array/list called listed_VMs

This function has 2 definitions - start and shutdown which start and shutdown all VMs stored in array/list listed_VMs.

powershell -File start_stop_script.ps1 start

powershell -File start_stop_script.ps1 shutdown

After, I created an Azure pipeline which used to run this script on a cron basis

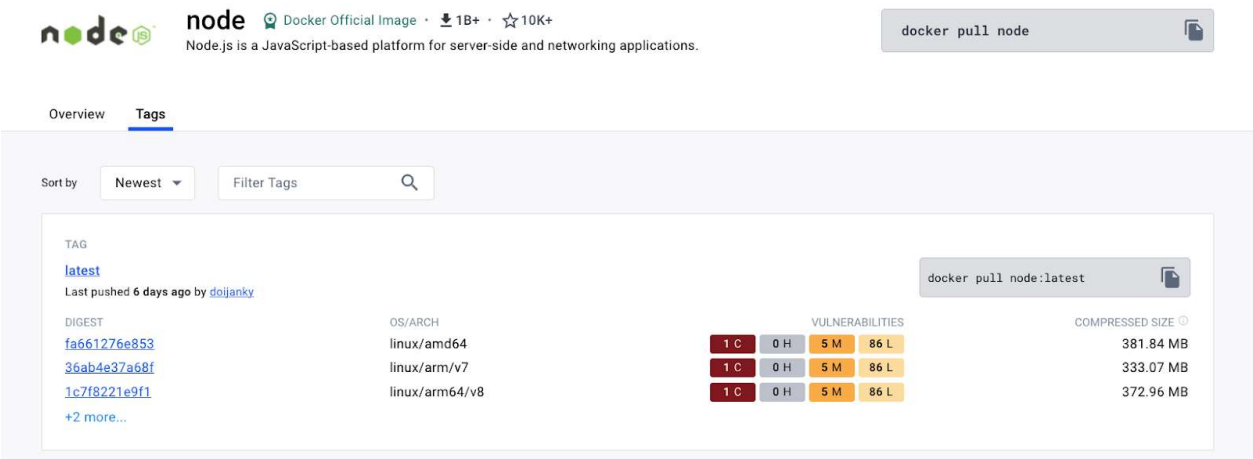
Start pipeline - 0 9 * * 1-5

Stop Pipeline - 0 18 * * 1-5

Docker

Task - One of my front-end developers has written a Dockerfile for their React application. The problem with that generated image was it was heavy in size(in GBs) and not consistent(shutdown/failed while running)

Solution - I have observed that it was not a best practice because he/she has taken a node image which is large about 381 MB. Also, the application is built and run in the same image. Because of that image size becomes 2 GB, which is not the optimal solution



After research I found that the best practice for frontend applications like react/angular is to build and run images should be separate. So, I have written multi-stage docker where build still happens in node image and after all the build files are copied to alpine nginx image to make image lightweight. (Alpine is one of the best and lightweight images in the Docker world)

```
# Stage 1: Build React App
FROM node:14.17.1 as build

WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .

RUN npm run build

# Stage 2: Create Nginx image and copy built files
FROM nginx:1.21.0-alpine

COPY --from=build /app/build /usr/share/nginx/html

# Expose port 80 for Nginx
EXPOSE 80

# Start Nginx
CMD ["nginx", "-g", "daemon off;"]
```

CICD

Problem - There was a problem in the pipeline. Even if deployment fails we get a successful message. After it failed very hard to debug the application by collecting logs, and events and tracing the problem.

Solution - To solve this problem, I have researched about it and found that in CNCF Landscape(where all cloud-native open-source tools are mentioned) one graduated CD listed is called ArgoCD.

I have a plan and give POC to my team to show the power of this tool. These are features if this tool

- Follow GitOps
- Solve kubectl problem
- Show the status of the application
- Monitoring
- Don't expose credentials
- Debug become easy because if the intuitive dashboard
- Stop Manual change(the only source of truth)
- Rollback become easy

After my team showed agreement with this tool. I have implemented it in dev and prod env

Kubernetes

Task - We have multiple environments and maintaining manifest files of k8s for each environment was hard. Also developer had hard time understanding k8s manifest files and updating any thing.

Solution - After research, I found that there are a few templating tools like Helm, Kustomize and Jsonnet in the market which help to solve the problem. I have picked Helm chart which is the most famous templating tool for k8s manifest files.

So, I have written a helm chart from scratch for all the applications I have and created a values file for each env. I also found best practice is to keep the helm chart in the same application repo instead of storing it remotely. Then modify the CICD pipeline to support the Helm chart.

If asked tell all the steps to generate a helm chart and the process to write it.

Terraform

Problem - In my company/industry we have multiple projects running. And demand was to write Terrafomr modules which are reusable and generic.

Solution - My current module contains variables which only needed by the project not all.

So, I have started creating/generating generic modules by adding all the variables/argumentReference given in the Terraform registry. Most of the modules use for_each. And I have made all the optional variables as optional attributes (explain optional attributes if asked). Then create all the Attribute Reference as Output in the module. That helps the module to use it in generic for all the environments.

After I have created a separate Git repo for terraform values and a Module. So, any project in my industry can call the module and use it.

Azure

Problem - The task was to reduce the cost of usage in a Development environment. All the applications are running in the AKS cluster.

Solution - So, after the research, we come to the conclusion that instead of using basic VM we start using the Spot instances(explain if required). The reason behind this is applications in Dev env are only used by the developer and not the end user. So, it is not necessary to run 24x7.

Also, change the replica sets in deployment to a minimum because it is used by only a few people.

And, schedule to start and stop the AKS cluster out of working hours.

https://learn.microsoft.com/en-us/azure/aks/start-stop-cluster?tabs=azure-cli