Wait, What's Happening

Deploying your applications to a cluster is just the first step for running containers in production, and it's important to think about operations and scenarios around your deployments. It is valuable to have a holistic understanding of your cluster when it comes to ensuring your applications are reliable, available, and tolerant to failures.

> **NOTE:** From this challenge on, your team may only advance if your cluster is in a healthy state.

Challenge

Your CTO is impressed with the speed at which you were able deploy the application but now wants to see you how your application is performing. The task the CTO has given for this challenge is to make sure your cluster is 'production ready' by implementing a monitoring solution that improves the observability of your cluster and adding alerts for key metrics so you can get ahead of any issues that will occur.

First, choose and implement a monitoring solution for your team to use. While choosing a monitoring solution, think about the four main components that must be considered to fully understand what is happening with your cluster so you can answer critical questions your CTO will ask.

1. Applications running on the containers
2. Containers
3. Underlying Virtual Machines
4. Kubernetes API

You can use the simulator from the previous step to create load on your application for testing and monitoring purposes.

A new application

Your focus on security and monitoring has encouraged other teams to try out AKS. Your CTO has asked you to run the following deployment in order to test a new project that calculates insurance rates faster and more accurately than ever before. This new project is intended to eventually become part of the TripInsights application. Ensure that you have configured your monitoring solution so you can quickly identify any issues that might arise.

> Replace the image reference below with a reference to your ACR. The insurance application image is already deployed into your ACR.

OpenHack © Microsoft 2021. All Rights Reserved - Powered By Opsgility Privacy.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: insurance-deployment
  labels:
    deploy: insurance
spec:
  replicas: 2
  selector:
    matchLabels:
      app: insurance
  template:
    metadata:
      labels:
        app: insurance
    spec:
      containers:
      - image: "replaceme.io/insurance:1.0"
        imagePullPolicy: Always
        name: insurance
        ports:
        - containerPort: 8081
          name: http
          protocol: TCP
---
apiVersion: v1
kind: Service
metadata:
  name: insurance
spec:
  type: ClusterIP
  selector:
    app: insurance
  ports:
  - protocol: TCP
    name: insurance-http
    port: 80
    targetPort: 8081
```

You can verify that the app is running by visiting the service endpoint which should return your calculation.

Of course, you didn't write this application, so it isn't up to your standards… Using your recently deployed monitoring solution, monitor the behavior of the new application in your cluster and see if you can determine what the runtime behavior of this application is. Additionally, if you find any issues, make sure to fix them in the deployment and create alerts for anything that might cause your application or cluster to experience downtime.

Success Criteria

- **Your team** must create a monitoring solution that shows the runtime behaviors of the application. You must be able to answer the following questions:
  - How many requests are coming to your cluster?
  - How much memory is allocatable per node in your cluster?
  - What is the CPU usage of your workload? What is the CPU usage of internal Kubernetes tools?
  - How many pods are currently pending?
  - Which pod is consuming the most memory?
- **Your team** must deploy a set of tools that will allow you to monitor your cluster and its applications.
- **Your team** must demonstrate where to obtain logs for the 4 main components mentioned in the first section of this page
- **Your team** must successfully implement resource limits on the newly deployed application
- **Your team** must set up an alert that informs you if an application is nearing resource limits in order to prevent cluster-wide issues
- **Your team** must demonstrate your cluster is overall "Healthy" for 15 minutess

References

Monitoring Microservices

- Overview of Monitoring for microservices (https://docs.microsoft.com/en-us/azure/architecture/microservices/logging-monitoring)

Azure

- Azure Container Insights reference (https://docs.microsoft.com/en-us/azure/azure-monitor/insights/container-insights-overview)

OpenHack © Microsoft 2021. All Rights Reserved - Powered By Opsgility Privacy.

- Search Logs to Analyze Data (https://docs.microsoft.com/en-us/azure/azure-monitor/insights/container-insights-log-search#search-logs-to-analyze-data)
- Kusto Query Language Reference (https://docs.microsoft.com/en-us/azure/kusto)
- Container Insights Alerts (https://docs.microsoft.com/en-us/azure/azure-monitor/insights/container-insights-alerts)

Prometheus

- Prometheus (https://prometheus.io/docs/introduction/overview/)
- Built-in Prometheus Metrics (https://github.com/helm/charts/tree/master/stable/nginx-ingress#prometheus-metrics)
- Prometheus Exporters (https://prometheus.io/docs/instrumenting/exporters/)
- Using Function Operators to Analyze Data from Prometheus (https://prometheus.io/docs/prometheus/latest/querying/examples/#using-functions-operators-etc)
- Visualization with Prometheus (https://prometheus.io/docs/visualization/grafana/)

OpenHack © Microsoft 2021. All Rights Reserved - Powered By Opsgility Privacy.