\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Auto scaling 테스트

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

■ 목표:

부하가 과다하게 발생 할 경우, POD가 자동으로 증설됨

■ 테스트 절차:

서비스 처리에 대한 부하를 과다하게 유발 시켜 POD Auto-scaling 적용

(로직에는 sleep 30적용을 통해 Java Thread 수행 시간이 오래 걸림)

가. 부하 발생 전

root@SKCC18D00125:~# kubectl get hpa -n bookstore

NAME REFERENCE TARGETS MINPODS MAXPODS REPLICAS AGE

reservation-hpa Deployment/reservation 0%/10% 1 4 1 1m14s

나. 부하 발생

http POST bookstore.skcc.co.kr/reservations userid="user" bookid="1" status="selfHealingTest"

다. 부하 발생 후

root@SKCC18D00125:~# kubectl get hpa -n bookstore

NAME REFERENCE TARGETS MINPODS MAXPODS REPLICAS AGE

reservation-hpa Deployment/reservation 58%/10% 1 4 4 6m55s

■ 결과:

Stress 테스트 통해 시스템 Capacity가 초과되는 Request 요청 시 HPA에 의해 POD자동 증가

Transactions: 7732 hits

Availability: 100.00 %

Elapsed time: 35.80 secs

Data transferred: 14.58 MB

Response time: 0.05 secs

Transaction rate: 215.98 trans/sec

Throughput: 0.41 MB/sec

Concurrency: 9.98

Successful transactions: 7732

Failed transactions: 0

Longest transaction: 3.06

Shortest transaction: 0.01

■ 참고 자료: HPA설정 내역

apiVersion: autoscaling/v1

kind: HorizontalPodAutoscaler

metadata:

name: nginx-hpa

spec:

maxReplicas: 4 # define max replica count

minReplicas: 1 # define min replica count

scaleTargetRef:

apiVersion: apps/v1

kind: Deployment

name: nginx

targetCPUUtilizationPercentage: 10 # target CPU utilization

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Auto-Healing 테스트

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

■ 목표:

POD에 예상치 못한 장애가 발생 할 경우, POD가 자동으로 재부팅됨.

■ 테스트 절차

과도하게 오래 실행되는 로직을 다수 수행할 경우, WAS의 Java Thread 부족으로 인해 Hang 발생

■ 테스트 명령어

while [ true ]; do http POST bookstore.skcc.co.kr/reservations userid="user" bookid="1" status="selfHealingTest" & done

■ 결과:

POD에 Hang 발생 시 liveness Probe 설정에 의해 자동으로 POD재부팅 됨.

Events:

Type Reason Age From Message

---- ------ ---- ---- -------

Normal Scheduled 24m default-scheduler Successfully assigned bookstore/reservation-6dcd658f8d-7lp8x to aks-agentpool-21044422-vmss000000

Normal Pulling 24m kubelet, aks-agentpool-21044422-vmss000000 Pulling image "ccteam4acr.azurecr.io/reservation:72"

Normal Pulled 24m kubelet, aks-agentpool-21044422-vmss000000 Successfully pulled image "ccteam4acr.azurecr.io/reservation:72"

Normal Created 24m kubelet, aks-agentpool-21044422-vmss000000 Created container reservation

Normal Started 24m kubelet, aks-agentpool-21044422-vmss000000 Started container reservation

Warning Unhealthy 24m kubelet, aks-agentpool-21044422-vmss000000 Liveness probe failed: Get http://10.244.0.67:8080/reservations: dial tcp 10.244.0.67:8080: connect: connection refused

Warning Unhealthy 23m (x4 over 24m) kubelet, aks-agentpool-21044422-vmss000000 Readiness probe failed: Get http://10.244.0.67:8080/reservations: dial tcp 10.244.0.67:8080: connect: connection refused

Warning Unhealthy 108s (x12 over 14m) kubelet, aks-agentpool-21044422-vmss000000 Readiness probe failed: Get http://10.244.0.67:8080/reservations: net/http: request canceled (Client.Timeout exceeded while awaiting headers)

Warning Unhealthy 100s (x6 over 14m) kubelet, aks-agentpool-21044422-vmss000000 Liveness probe failed: Get http://10.244.0.67:8080/reservations: net/http: request canceled (Client.Timeout exceeded while awaiting headers)

Normal Killing 100s kubelet, aks-agentpool-21044422-vmss000000 Container reservation failed liveness probe, will be restarted

■ 참고 자료: Probe 설정 내역

livenessProbe:

httpGet:

path: /reservations

port: 8080

scheme: HTTP

initialDelaySeconds: 5

periodSeconds: 15

timeoutSeconds: 5

readinessProbe:

httpGet:

path: /reservations

port: 8080

scheme: HTTP

initialDelaySeconds: 5

timeoutSeconds: 1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ConfigMap / Secret 적용

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

■ 목표:

환경 변수 적용을 통해 용도 별 구분 인자값으로 활용

■ 결과:

CnfigMap: 개발기/운영기 구분자료 활용

kubectl exec -ti reservations2-6dd99458f8-ncfng -n bookstore env | egrep EMBED\_TOMCAT\_JAVA\_OPTS

EMBED\_TOMCAT\_JAVA\_OPTS=-client

Secret: DB접속 정보 저장 목적으로 활용

kubectl exec -ti reservations2-6dd99458f8-ncfng -n bookstore env | egrep "DB\_USER|DB\_PASS"

DB\_USER=myuser

DB\_PASS=#skcc123

■ 참고 자료: ConfigMap/Secret 설정 내역

apiVersion: v1

kind: ConfigMap

metadata:

name: spring-dev

namespace: bookstore

data:

DB\_URL: skcc-bookstore-dev

JAVA\_OPTS: -client

---

apiVersion: v1

kind: Secret

metadata:

name: db-dev

namespace: bookstore

data:

DB\_USER: bXl1c2VyCg==

DB\_PASS: I3NrY2MxMjMK

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Monitoring

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

■ 목표:

쿠버네티스 서비스 모니터링 구성

POD의 상태가 비정상이거나, Node의 CPU/Memory 사용량이 비정상적으로 과도할 경우, 운영자에게 SMS으로 Alert함.

■ 구성 내역:

Azure Monitor를 통해 CPU, Memory, POD등이 임계치를 초과 할 경우 SMS 발송 되도록 구성 함.

|  |
| --- |
|  |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Alerting

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

■ 목표:

쿠버네티스 서비스 모니터링 및 Alerting 구성

POD의 상태가 비정상이거나, Node의 CPU/Memory 사용량이 비정상적으로 과도할 경우, 운영자에게 SMS으로 Alert함.

■ 구성 내역:

Azure Monitor를 통해 CPU, Memory, POD등이 임계치를 초과 할 경우 SMS 발송 되도록 구성 함.

|  |
| --- |
|  |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Persistent Volume

■ 목표:

Persistent Volume 마운트를 통해 영구 데이터 저장

해당 스토리지는 첨부 파일들을 저장하는데 사용 가능하며, Dynamic으로 유동성있게 구성함.

■ 구성 결과:

root@mypod-azurefiles:/# df -h | grep azure

//faee688dbf4284f62a1a567.file.core.windows.net/kubernetes-dynamic-pvc-d402c327-d428-4eb8-9f0d-072314cc6555 5.0G 0 5.0G 0% /mnt/azure

■ 설정 내역:

[Storage Class]

|  |
| --- |
| kind: StorageClass  apiVersion: storage.k8s.io/v1  metadata:  name: azurefile  provisioner: kubernetes.io/azure-file  mountOptions:  - dir\_mode=0777  - file\_mode=0777  - uid=0  - gid=0  - mfsymlinks  - cache=strict  parameters:  skuName: Standard\_LRS |

[PVC]

|  |
| --- |
| apiVersion: v1  kind: PersistentVolumeClaim  metadata:  name: azurefile  spec:  accessModes:  - ReadWriteMany  storageClassName: azurefile  resources:  requests:  storage: 5Gi |

[POD]

|  |
| --- |
| kind: Pod  apiVersion: v1  metadata:  name: mypod-azurefiles  spec:  containers:  - name: mypod  image: nginx:1.15.5  resources:  requests:  cpu: 100m  memory: 128Mi  limits:  cpu: 250m  memory: 256Mi  volumeMounts:  - mountPath: "/mnt/azure"  name: volume  volumes:  - name: volume  persistentVolumeClaim:  claimName: bookstore-azurefile |