**Keda**

Helm을 통해 keda 설치를 진행해보도록 할 것이다.

|  |
| --- |
| helm repo add kedacore <https://kedacore.github.io/charts>  helm repo update |

네임스페이스를 생성해준다.

|  |
| --- |
| kubectl create namespace keda |

Keda 패키지를 설치한다.

|  |
| --- |
| helm install keda kedacore/keda --namespace keda |

이제 scald object를 파일을 작성한다.

아래 공식사이트를 참고하여 scald object 파일을 작성하였다

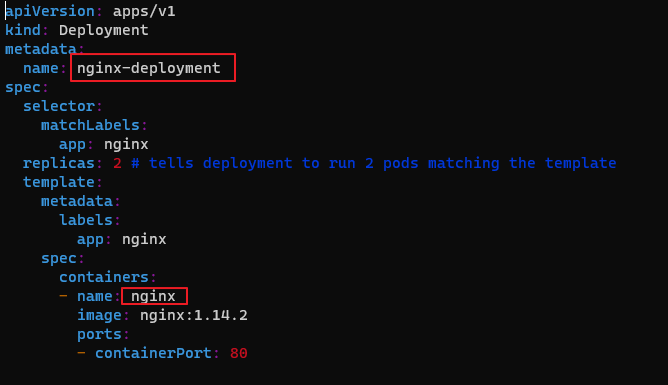
<https://keda.sh/docs/2.9/concepts/scaling-deployments/#scaling-of-deployments-and-statefulsets>

우선 demo container을 생성해준다. 예시로 nginx container를 생성하였다.

deployment.yaml

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: nginx-deployment  spec:  selector:  matchLabels:  app: nginx  replicas: 2 # tells deployment to run 2 pods matching the template  template:  metadata:  labels:  app: nginx  spec:  containers:  - name: nginx  image: nginx:1.14.2  ports:  - containerPort: 80 |

yaml 파일의 내용을 잘 기억해준다.

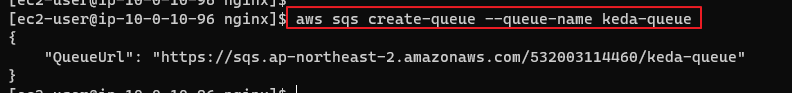


Yaml 파일을 작성할 때 트리거도 참고해주면 좋다.

<https://keda.sh/docs/2.11/scalers/aws-sqs/>

우선 여기서 진행할 방법은 sqs를 통한 자동확장이다.

대충 다음과 같이 kube-queue를 생성해준다.



AWS\_ROLE\_ARN에 기입하였던 곳에 sts 권한을 부여해준다.

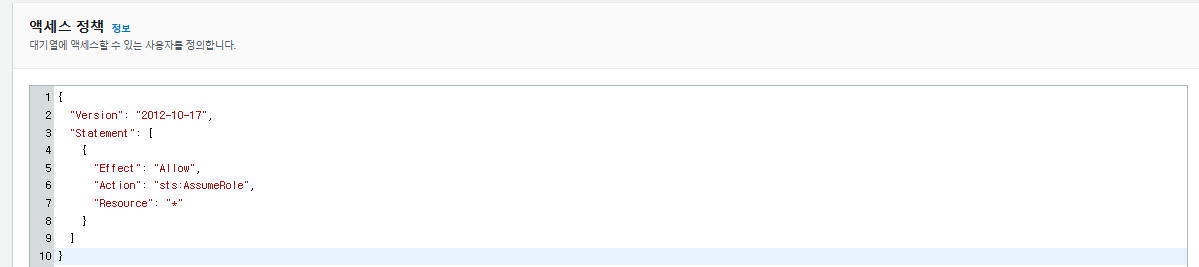
정책은 다음과 같다.

|  |
| --- |
| {  "Version": "2012-10-17",  "Statement": [  {  "Sid": "VisualEditor0",  "Effect": "Allow",  "Action": "sts:\*",  "Resource": "\*"  }  ]  } |

이후 yaml 파일을 적용시켜준다.

|  |
| --- |
| apiVersion: v1  kind: Secret  metadata:  name: test-secrets  data:  AWS\_ROLE\_ARN: YXJuOmF3czppYW06OjUzMjAwMzExNDQ2MDpyb2xlL0FkbWluaXN0cmF0b3JBY2Nlc3M=  ---  apiVersion: keda.sh/v1alpha1  kind: TriggerAuthentication  metadata:  name: keda-trigger-auth-aws-credentials  spec:  secretTargetRef:  - parameter: awsRoleArn # The property in KEDA.  name: test-secrets # The name of the kubernetes secret.  key: AWS\_ROLE\_ARN # The key from the kubernetes secret.  ---  apiVersion: keda.sh/v1alpha1  kind: ScaledObject  metadata:  name: keda-scale-object  spec:  scaleTargetRef:  apiVersion: apps/v1  kind: Deployment  name: nginx-deployment # Mandatory. Must be in the same namespace as the ScaledObject  envSourceContainerName: nginx # Optional. Default: .spec.template.spec.containers[0]  pollingInterval: 30 # Optional. Default: 30 seconds  cooldownPeriod: 300 # Optional. Default: 300 seconds  idleReplicaCount: 0 # Optional. Default: ignored, must be less than minReplicaCount  minReplicaCount: 1 # Optional. Default: 0  maxReplicaCount: 100 # Optional. Default: 100  fallback: # Optional. Section to specify fallback options  failureThreshold: 3 # Mandatory if fallback section is included  replicas: 6 # Mandatory if fallback section is included  advanced: # Optional. Section to specify advanced options  restoreToOriginalReplicaCount: false # Optional. Default: false  horizontalPodAutoscalerConfig: # Optional. Section to specify HPA related options  name: keda-hpa # Optional. Default: keda-hpa-{scaled-object-name}  behavior: # Optional. Use to modify HPA's scaling behavior  scaleDown:  stabilizationWindowSeconds: 300  policies:  - type: Percent  value: 100  periodSeconds: 15  triggers:  - type: aws-sqs-queue  authenticationRef:  name: keda-trigger-auth-aws-credentials  metadata:  queueURL: keda-queue  queueLength: "5"  awsRegion: "ap-northeast-2" |

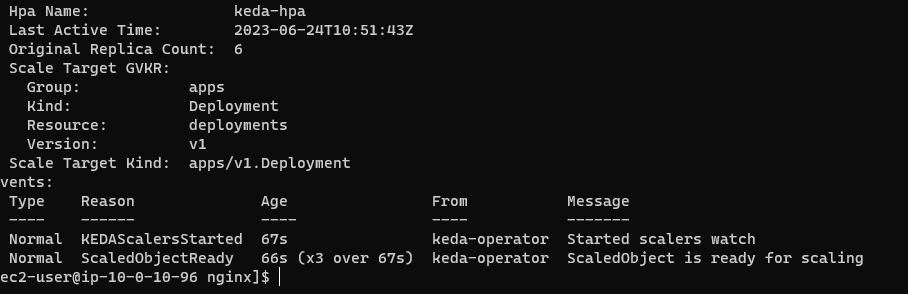
Sqs 엑세스 정책을 다음과 같이 수정해준다.



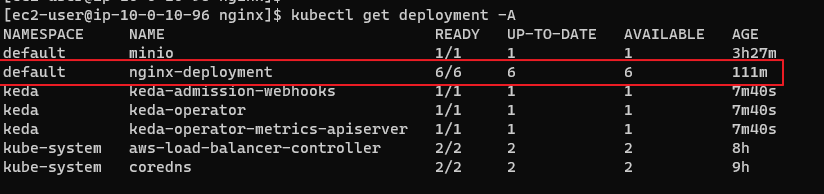
그리고 대충 다음과 같은 신뢰 정책을 작성한다.

|  |
| --- |
| {  "Effect": "Allow",  "Principal": {  "AWS": "arn:aws:iam::532003114460:assumed-role/eksctl-skills-cluster-nodegroup-s-NodeInstanceRole-V7DTINBCJUJG/i-00cea08711b2b4174"  },  "Action": "sts:AssumeRole"  } |

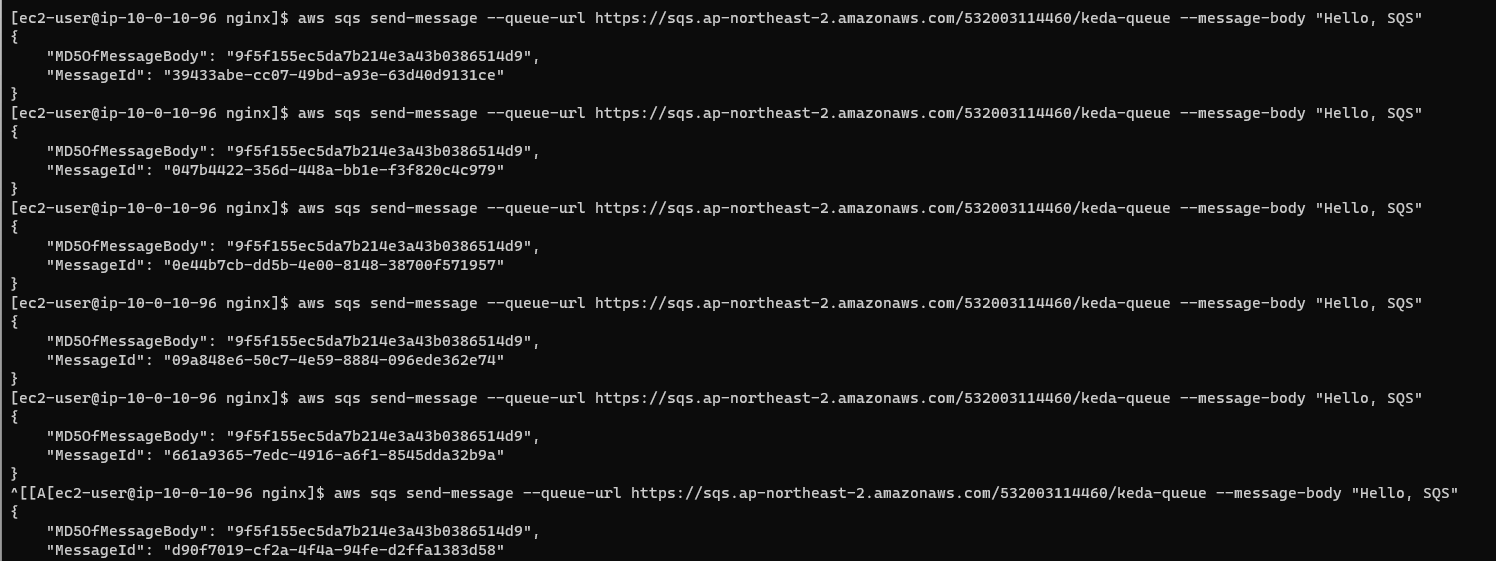
오류가 표시되지 않는다.



적용시 deployment를 확인하면 1/1에서 6/6으로 변경된 것을 확인할 수 있다.



이제 queue에 메시지를 전송해준다.



다음과 같이 늘어난 것을 확인할 수 있다.

