

ปฏิบัติการที่ 6 การสร้างแอปของตนเอง

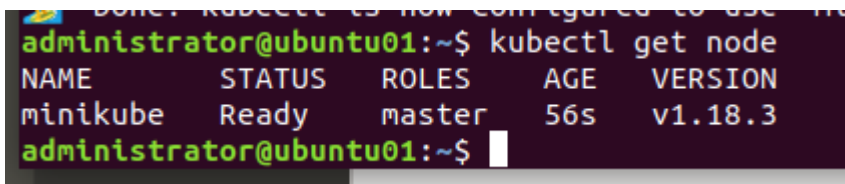
ให้ดาวน์โหลดตัวอย่างของคลาสจาก <https://github.com/khajorn/kubernetes-course>

สิ่งที่ควรทราบก่อนการดำเนินการ

1. เครื่องจะต้องมีการติดตั้ง minikube มาก่อนซึ่งเวอร์ชันที่ดาวน์โหลดปัจจุบันคือ v1.11 แนะนำให้ดาวน์โหลดโดยตรงถ้าใช้ snap หรือ brew อาจจะเป็นเวอร์ชันเก่า
2. โฟลเดอร์ .minikube ต้องกำหนดสิทธิ์ที่ไม่ใช่ root ในการเข้าถึง เนื่องจาก sudo แอปพลิเคชัน virtualbox จะไม่ให้รัน
3. กรณีที่มี vagrantfile ให้ไปดูคำรายละเอียดว่าถูกต้องหรือไม่ใน \$home\$/vagrantfile ให้ตรวจสอบว่าตอนนี้เราอยู่ในโฟลเดอร์ใด เมื่อเราเรียกคำสั่ง vagrant ตำแหน่งใด ก็จะอ่านไฟล์ vagrantfile ตำแหน่งนั้น
 - เรียกคำสั่ง vagrant up
 - vagrant ssh ถ้าต้องการเข้าใช้

1.1 การดาวน์โหลดแอป

1. ทำการโคลน git ด้วยคำสั่ง
Sudo apt-get install git
git clone <https://github.com/khajorn/kubernetes-course>
จะได้โฟลเดอร์ kubernetes-course ใน home
2. ตรวจสอบโหนดบน minikube ที่รันบนเครื่อง
kubectl get node



```
Done! kubectl is now configured to use the minikube context.  
administrator@ubuntu01:~$ kubectl get node  
NAME          STATUS    ROLES    AGE   VERSION  
minikube      Ready     master   56s   v1.18.3  
administrator@ubuntu01:~$
```

ดำเนินการบน minikube

3. เข้าไปดูรายละเอียดใน first-app
cd kubernetes-course/
cat first-app/helloworld.yml
ดูรายการไฟล์
4. ทำการติดตั้งแอปพลิเคชันที่เป็นไฟล์
kubectl create -f first-app/helloworld.yml

5. ตรวจสอบ pod ด้วยคำสั่ง

```
kubectl get pod
```

6. ตรวจสอบแอปพลิเคชัน

```
kubectl describe pod nodehelloworld.example.com
```

```
administrator@ubuntu01: ~/kubernetes-course
File Edit View Search Terminal Help
Error from server (NotFound): pods "nodehelloworld.exmaple.com" not found
administrator@ubuntu01:~/kubernetes-course$ kubectl describe pod nodehelloworld.example.com
Name:          nodehelloworld.example.com
Namespace:     default
Priority:       0
Node:          minikube/192.168.99.100
Start Time:    Fri, 19 Jun 2020 17:01:58 +0700
Labels:        app=helloworld
Annotations:    <none>
Status:        Running
IP:            172.17.0.4
IPs:
  IP: 172.17.0.4
Containers:
  k8s-demo:
    Container ID:  docker://bbad2601e69854fcde8e1a4bf0dfa7b1f49e302eaed9a983adb5fc72f853e4e
    Image:         wardviaene/k8s-demo
    Image ID:      docker-pullable://wardviaene/k8s-demo@sha256:2c050f462f5d0b3a6430e7869bc
                   6ac48a447a89da79a56d0ef61460c7ab9e
    Port:          3000/TCP
    Host Port:     0/TCP
    State:         Running
      Started:     Fri, 19 Jun 2020 17:03:17 +0700
    Ready:         True
    Restart Count: 0
    Environment:   <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-7xhh9 (ro)
Conditions:
  Type           Status
  Initialized     True
  Ready          True
  ContainersReady True
  PodScheduled   True
Volumes:
  default-token-7xhh9:
    Type:          Secret (a volume populated by a Secret)
    SecretName:    default-token-7xhh9
    Optional:      false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute for 300s
                  node.kubernetes.io/unreachable:NoExecute for 300s
```

อธิบายรายละเอียดชื่อ, ชื่อ Container และเงื่อนไข

ถ้าอยู่ระหว่างการสร้าง pod จะขึ้นสถานะ เช่น pulling

7. การส่งต่อพอร์ต

```
Kubectl port-forward nodehelloworld.example.com 8081:3000
```

```

administrator@ubuntu01:~/kubernetes-course$ kubectl port-forward nodehelloworld.example.com
8081:3000
Forwarding from 127.0.0.1:8081 -> 3000
Forwarding from [::1]:8081 -> 3000

```

พบว่าตอนนี้เปลี่ยน pod จาก 8081 ไปเป็น 3000

8. การตรวจสอบ Port forwarding ทำโดยเปิด Terminal ใหม่ แล้วพิมพ์คำสั่ง

Curl localhost:8081

สิ่งที่ตอบกลับคือ

```

administrator@ubuntu01: ~/kubernetes-course
File Edit View Search Terminal Tabs Help
administrator@ubuntu01: ~/kubernetes-c... x administrator@ubuntu01: ~/kubern...
administrator@ubuntu01:~/kubernetes-course$ curl localhost:8081
Hello World!administrator@ubuntu01:~/kubernetes-course$

```

แสดงว่าระบบได้ทำงาน

9. การตรวจสอบสามารถที่จะดูเป็น Service ได้โดย

kubectl expose pod nodehelloworld.example.com --type=NodePort --name nodehelloworld-service

```

administrator@ubuntu01:~/kubernetes-course$ kubectl expose pod nodehelloworld.ex
ample.com --type=NodePort --name nodehelloworld-service
service/nodehelloworld-service exposed
administrator@ubuntu01:~/kubernetes-course$

```

เราสามารถทำในลักษณะเดียวกันที่ aws หรือ load balance บน Cloud ได้

10. หรือตรวจสอบ service ใน minikube ด้วยคำสั่ง

Minikube service nodehelloworld-service --url

```

administrator@ubuntu01:~/kubernetes-course$ minikube service nodehelloworld-serv
ice --url
http://192.168.99.100:30909
administrator@ubuntu01:~/kubernetes-course$

```

11. ตรวจสอบบริการ

Kubectl get service

```

administrator@ubuntu01:~/kubernetes-course$ kubectl get service

```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	5d4h
nodehelloworld-service	NodePort	10.103.118.123	<none>	3000:30909/TCP	3m30s

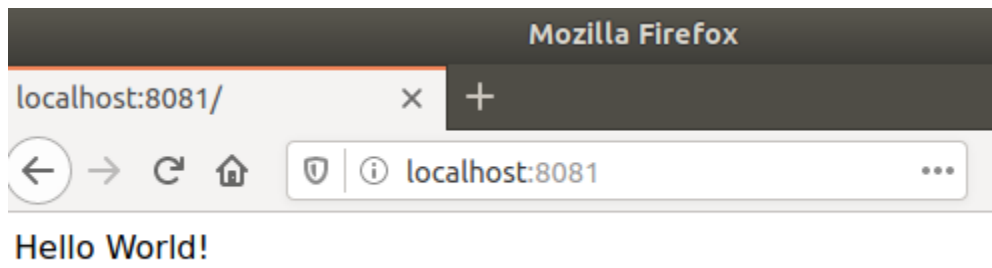
```

administrator@ubuntu01:~/kubernetes-course$

```

พบว่าจะมี kubernetes ที่ดำเนินการบน Cloud Cluster และตัวอย่าง nodehelloworld ที่พอร์ต 3000

12. ถ้าพิมพ์ URL ที่พบจะได้ผ่านเบราว์เซอร์



ปฏิบัติการย่อยที่ 6.2 การเรียนรู้คำสั่ง kubectl

คำสั่ง	คำอธิบาย
kubectl get pod	ดึงข้อมูลจาก pods ทั้งหมด
kubectl describe pod nodehelloworld.example.com	อธิบายในหนึ่ง pod
kubectl expose pod nodehelloworld.example.com --type=NodePort --name nodehelloworld-service	กระจายพอร์ตของ pod (สร้างบริการใหม่)
Kubectl port-forward nodehelloworld.example.com 8081:3000	ส่งต่อพอร์ตกระจายพอร์ต pod ไปยังเครื่องผู้เรียน
kubectl attach nodehelloworld.example.com	ต่อกับ pod
Kubectl get service	ดูบริการ
kubectl exec nodehelloworld.example.com -- ls /app	สั่งคำสั่งบน pod
kubectl exec nodehelloworld.example.com -- touch /app/test.txt	สั่งคำสั่งบน pod สร้างไฟล์
kubectl exec nodehelloworld.example.com -- ls /app	สั่งคำสั่งบน pod ดูอีกครั้ง
kubectl describe service nodehelloworld-service	เป็นการดูรายละเอียดบริการ พบว่า จะมีป้ายชื่อ หมายเลขไอพี ดู nodeport และดูหมายเลขไอพี endpoint
kubectl label pods nodehelloworld.example.com mylabel=helloworld	เพิ่มป้ายชื่อไปสู่ pod
kubectl run -i --tty busybox --image=busybox --restart=Never -- sh	รันเชลล์ใน pod – ใช้ประโยชน์มาก ในการดีบั๊ก จะขึ้น Shell มา / #
Ls	ตรวจสอบข้อมูล
telnet หมายเลขไอพีที่เห็นใน Endpoint และพอร์ต get /	ติดต่อ 172.17.0.4:3000 และดูว่าได้ข้อมูล helloworld หรือไม่

ปฏิบัติการที่ 6.3 การกำหนด First App กับ Load Balancer

ทำการเข้าไปในเครื่อง vagrant บน Virtualbox และสำเนาค่า kubernetes-course เก็บไว้ในอิมเมจ

ถ้ามีการลบอิมเมจของ aws ให้ดำเนินการสร้างปฏิบัติการที่ 4 ใหม่อีกครั้ง โดยดูหัวข้อที่ 4.3 เรียกคำสั่ง kops เพื่อสร้างคลัสเตอร์ (ถ้ามีอยู่แล้วให้ข้ามไป)

```
kops create cluster --name=kubernetes.getyoungit.com --state=s3://kops-state-a001a --zones=ap-southeast-1a --node-count=2 --node-size=t2.micro --master-size=t2.micro --dns-zone=kubernetes.getyoungit.com
```

ตรวจสอบโดย host -t NS kubernetes.getyoungit.com

ตรวจสอบโหนดด้วย kubectl get nodes

การสร้าง first-app บน aws

1. เข้าไปในโฟลเดอร์ที่เตรียมอิมเมจในที่นี้คือที่ home
2. พิมพ์ว่า vagrant ssh
จะอ่านไฟล์ vagrantfile
3. ให้ดาวน์โหลดตัวอย่างเอกสารใน github ถ้ามีการดาวน์โหลดแล้วข้ามขั้นตอนนี้ไปเลย
git clone <https://github.com/khajorn/kubernetes-course>
4. เข้าไปในโฟลเดอร์ kubernetes-course
cd kubernetes-course
5. เข้าดูรายละเอียดไฟล์ helloworld.yml
cat first-app/helloworld.yml

```
vagrant@ubuntu-bionic:~/kubernetes-course$ cat first-app/helloworld.yml
apiVersion: v1
kind: Pod
metadata:
  name: nodehelloworld.example.com
  labels:
    app: helloworld
spec:
  containers:
    - name: k8s-demo
      image: sipadocker/k8s-demo
      ports:
        - name: nodejs-port
          containerPort: 3000
vagrant@ubuntu-bionic:~/kubernetes-course$
```

6. เข้าดูรายละเอียดไฟล์ helloworld-service.yml

cat first-app/helloworld-service.yml

```
contatnerPort: 3000
vagrant@ubuntu-bionic:~/kubernetes-course$ cat first-app/helloworld-service.yml
apiVersion: v1
kind: Service
metadata:
  name: helloworld-service
spec:
  ports:
    - port: 80
      targetPort: nodejs-port
      protocol: TCP
  selector:
    app: helloworld
  type: LoadBalancer
vagrant@ubuntu-bionic:~/kubernetes-course$
```

ดูพอร์ตที่เปิด 80 และ targetPort คือ nodejs-port

7. การสร้าง pod และ service จากไฟล์

kubectl create -f first-app/helloworld.yml

Kubectl create -f first-app/helloworld-service.yml

ถ้ามีอยู่แล้วระบบจะแจ้งให้ทราบให้พิมพ์บรรทัดถัดไป ซึ่งเราสามารถที่จะสร้างได้ทั้งบน minikube และ kops ขึ้นอยู่กับเครื่อง

```
vagrant@ubuntu-bionic:~$ cd kubernetes-course/
vagrant@ubuntu-bionic:~/kubernetes-course$ kubectl create -f first-app/helloworld.yml
pod/nodehelloworld.example.com created
vagrant@ubuntu-bionic:~/kubernetes-course$ kubectl create -f first-app/helloworld-service.yml
service/helloworld-service created
vagrant@ubuntu-bionic:~/kubernetes-course$
```

8. เข้าไปตรวจสอบค่ากำหนดจาก kops ไปที่ AWS console (อเมซอนเว็บไซต์)

รอสักพักตรวจสอบว่ามีการสร้าง loadbalancer หรือไม่

ตรวจสอบโดยใช้คำสั่ง kubectl คือ

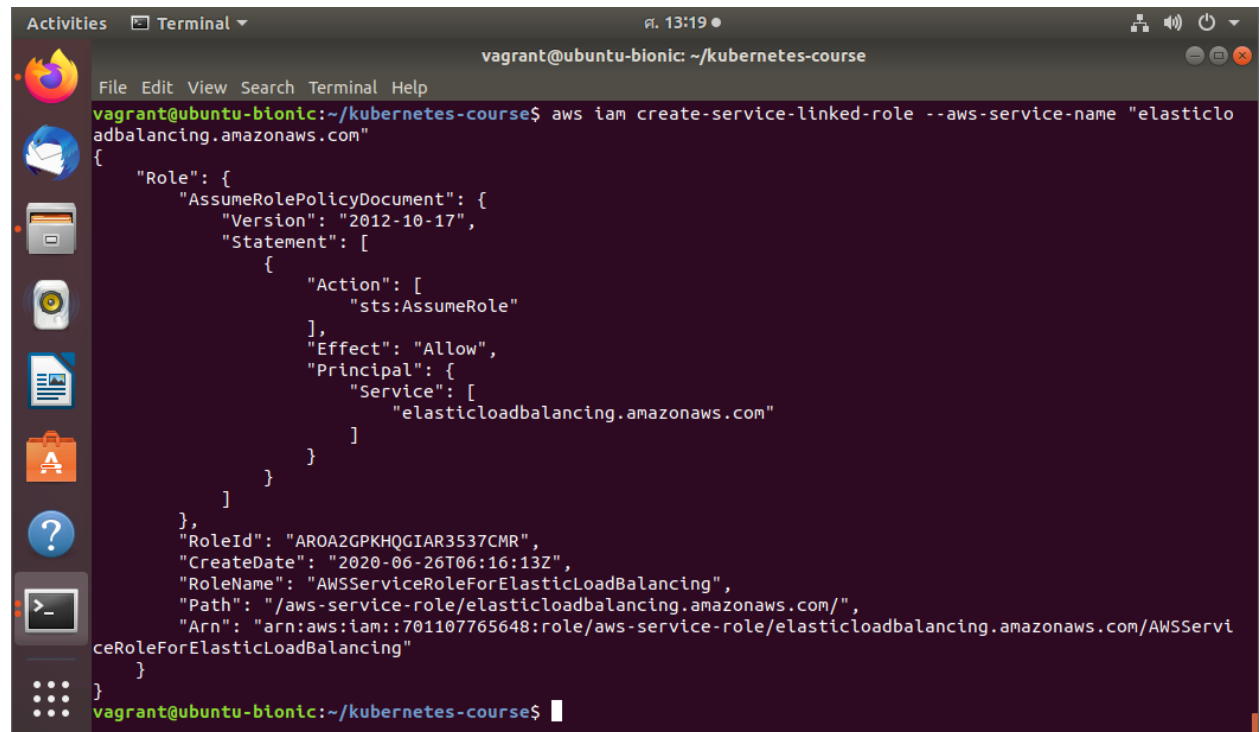
kubectl get services

```
vagrant@ubuntu-bionic:~/kubernetes-course$ kubectl get services
NAME                TYPE            CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
helloworld-service  LoadBalancer   100.70.122.1    <pending>        80:31668/TCP     4m29s
kubernetes           ClusterIP       100.64.0.1      <none>           443/TCP          9m38s
vagrant@ubuntu-bionic:~/kubernetes-course$
```

ถ้าไม่เคยสร้าง ELB ใน AWS account จะไม่เห็น ELB ปรากฏให้รันคำสั่งนี้

aws iam create-service-linked-role --aws-service-name "elasticloadbalancing.amazonaws.com"

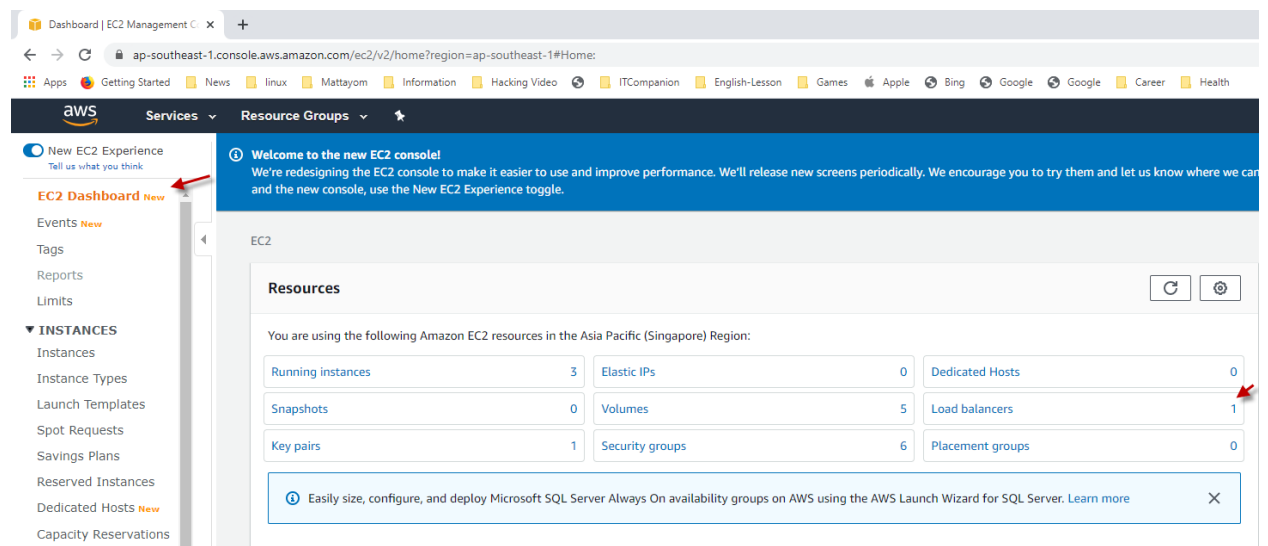
หรือจะสร้างด้วยมือ ELB ในหน้าจอ AWS Console และนำออกทันทีหลังจากทริกเกอร์ได้สร้าง Role ขึ้น



```
File Edit View Search Terminal Help
vagrant@ubuntu-bionic: ~/kubernetes-course
vagrant@ubuntu-bionic:~/kubernetes-course$ aws iam create-service-linked-role --aws-service-name "elasticloadbalancing.amazonaws.com"
{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": [
            "sts:AssumeRole"
          ],
          "Effect": "Allow",
          "Principal": {
            "Service": [
              "elasticloadbalancing.amazonaws.com"
            ]
          }
        }
      ]
    },
    "RoleId": "AROAZGPKHQGIAR3537CMR",
    "CreateDate": "2020-06-26T06:16:13Z",
    "RoleName": "AWSServiceRoleForElasticLoadBalancing",
    "Path": "/aws-service-role/elasticloadbalancing.amazonaws.com/",
    "Arn": "arn:aws:iam::701107765648:role/aws-service-role/elasticloadbalancing.amazonaws.com/AWSServiceRoleForElasticLoadBalancing"
  }
}
```

- ต้องทำขั้นตอนนี้ก่อน create service ดังนั้นถ้าไม่ขึ้นให้ไปลบบริการ แล้วสร้างใหม่

9. ตรวจสอบใน aws console



- คลิกที่ EC2 Dashboard
- คลิกที่ Load balancers พบว่าจะมีรายการหนึ่งรายการ

10. ไปคลิกที่ Instances เพื่อตรวจสอบโหนดที่รัน

Load Balancers | EC2 Management Console

ap-southeast-1.console.aws.amazon.com/ec2/v2/home?region=ap-southeast-1#LoadBalancers

Services Resource Groups

New EC2 Experience

EC2 Dashboard New

Events New

Tags

Reports

Limits

▼ INSTANCES

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts New

Capacity Reservations

▼ IMAGES

AMIs

Bundle Tasks

▼ ELASTIC BLOCK STORE

Volumes

Snapshots

Lifecycle Manager

▼ NETWORK & SECURITY

Create Load Balancer Actions

Filter by tags and attributes or search by keyword

1 to 1 of 1

Name	DNS name	State	VPC ID
ad4a878b023244b2e9a122caa19414c4	ad4a878b023244b2e9a122caa19414c4-2061480059.ap-southeast-1.elb.amazonaws.com (A Record)	Available	vpc-097782333339168c0

Load balancer: ad4a878b023244b2e9a122caa19414c4

Description Instances Health check Listeners Monitoring Tags Migration

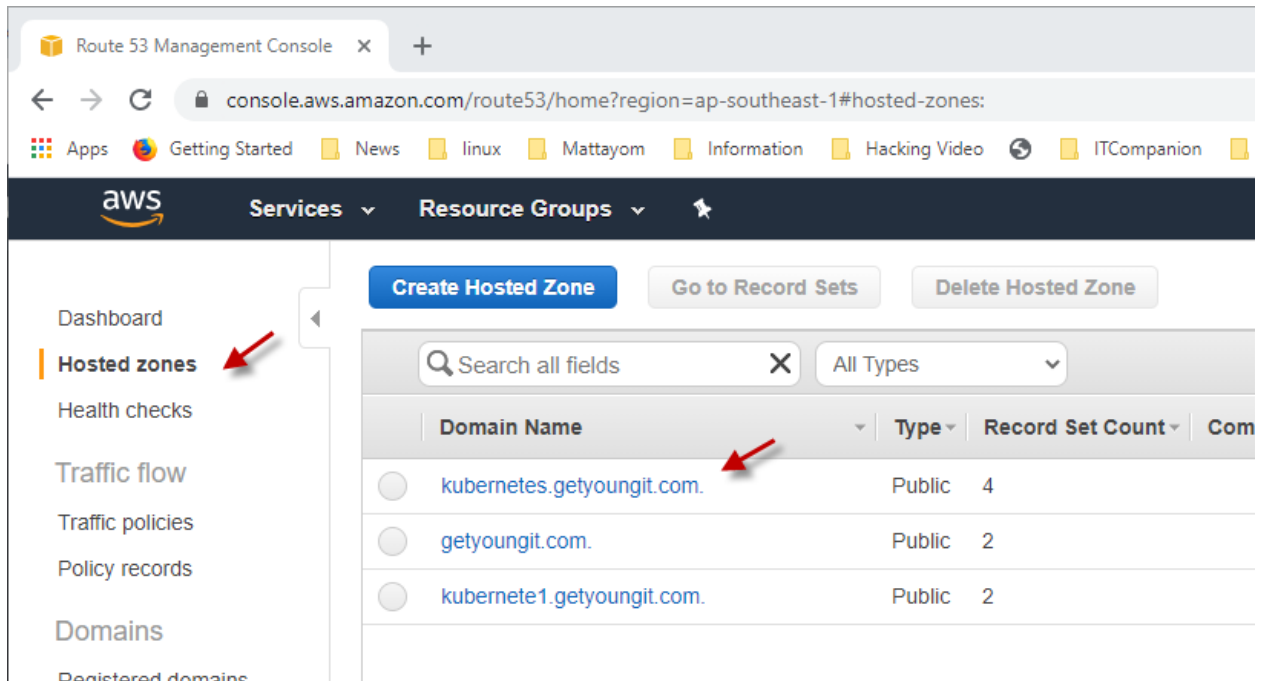
Basic Configuration

Name	ad4a878b023244b2e9a122caa19414c4	Creation time	June 26, 2020 at 1:20:46 PM UTC+7
* DNS name	ad4a878b023244b2e9a122caa19414c4-2061480059.ap-southeast-1.elb.amazonaws.com (A Record)	Hosted zone	Z1LMS91P8CMLE5
Type	Classic (Migrate Now)	Status	2 of 2 instances in service
Scheme	internet-facing	VPC	vpc-097782333339168c0
Availability Zones	subnet-0f89dcd7896ec0473 - ap-southeast-1a		

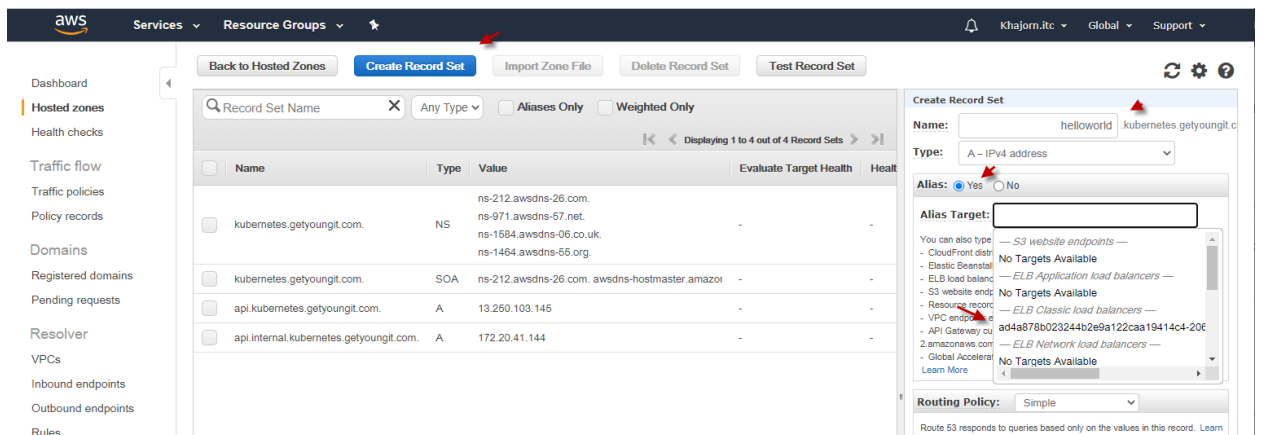
Feedback English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

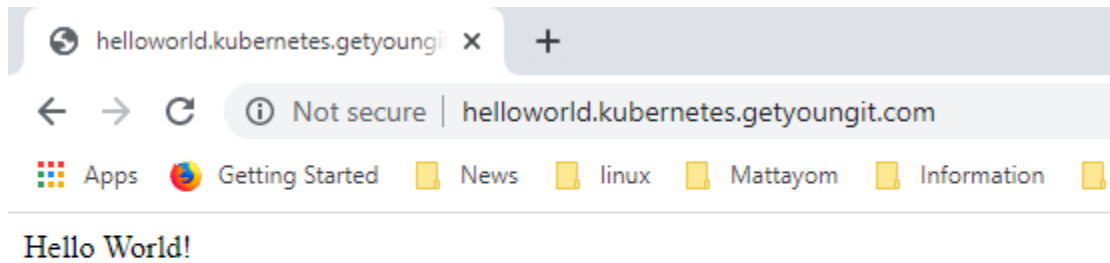
- ใน Scheme เป็น Internet facing
- จะพบว่ามี 2 รายการ
- คลิกที่ Health check เพื่อตรวจสอบภาพรวม
- คลิกที่ Listeners เพื่อตรวจสอบว่ารันพอร์ต 80 ตามค่ากำหนดหรือไม่
- 11. คลิกที่แท็บ Description แล้วไปดูรายงานละเอียดใน DNS และทำการเก็บค่า DNS name
- 12. ไปที่ค่ากำหนดใน Route 53 โดยคลิกที่ Services -> คลิก Route 53
- คลิกที่ Hosted zones ด้านซ้ายมือ
- คลิกเลือกโฮสต์ คือ kubernetes.getyoungit.com



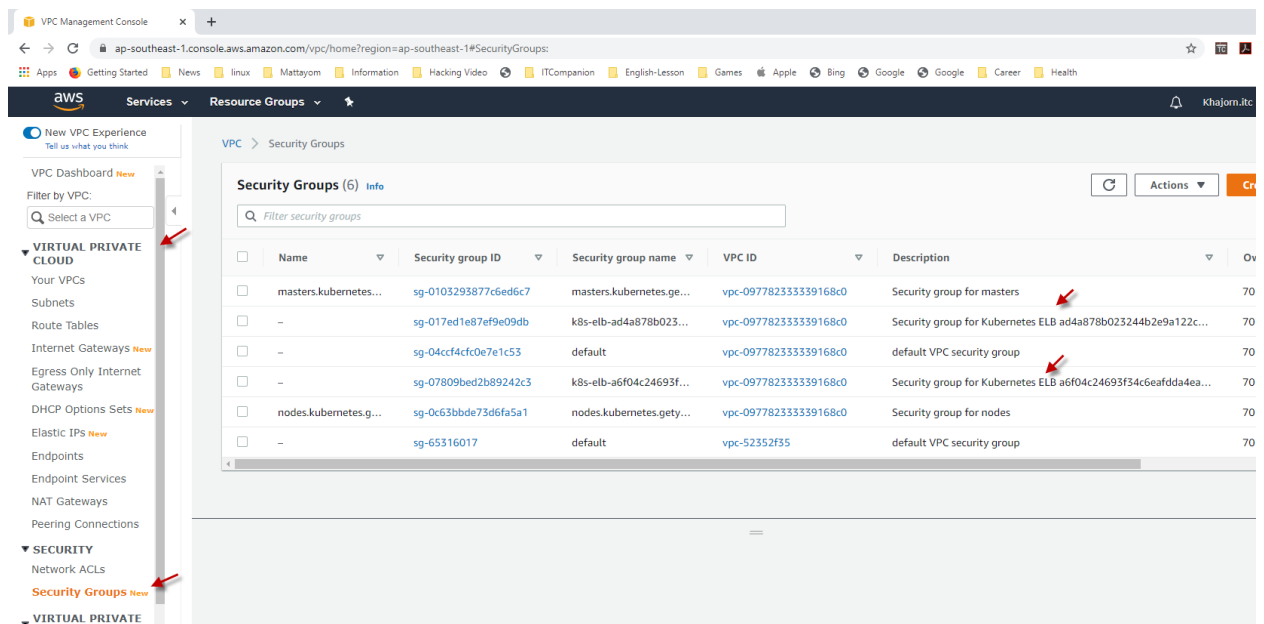
- การระบุเรคคอร์ดใน Route 53
 - คลิกที่ Create Record Set
 - ในช่อง Name ระบุชื่อคือ Helloworld
 - เลือก Alias เป็น yes
 - ในช่อง Alias Target ระบุเลือก ELB ที่สร้างไว้ก่อนหน้านี้



- คลิก Create
 - พบว่ามีรายการของชื่อที่เราระบุในเรคคอร์ดที่สร้างไว้
- ทดสอบโดยไปเปิด URL ในบราวเซอร์ชื่อ helloworld.kubernetes.getyoungit.com



อธิบาย Route 53 -> DNS -> ELB -> Host
การตรวจสอบผลของการสร้าง NLB บน AWS



1. ไปคลิกที่ Services พิมพ์ว่า Virtual Private Cloud แล้วเลือกการ VPC
2. ไปคลิกที่ Security Group แถบซ้ายมือด้านล่างหน้าจอ หรือจะคลิกตรงหน้า Dash board
3. ดูรายการกลุ่มที่ถูกสร้างจาก ELB
4. คลิกที่กลุ่มของโหนด แล้วคลิกที่แท็บ Inbound rules เพื่อตรวจสอบกฎที่ระบุ

