**Indicaciones**

**Amazon ELB - Auto Scaling**

1. Las respuestas deben ser explicadas, solo colocar resultados sin ninguna referencia no puntúa en las preguntas de la evaluación.
2. Realiza una copia de este documento y coloca todas tus respuestas y sube a tu repositorio personal de GitHub en formato markdown. Presenta capturas de pantalla del procedimiento y las explicaciones necesarias. No puntúa si solo se hace la presentación de imágenes.
3. De preferencia adiciona un video adicional explicando los pasos realizados. Utiliza el sandbox de AWS usado en la práctica anterior.
4. Sube a la plataforma de Blackboard el enlace de GitHub donde están todas tus respuestas. No olvides colocar tu nombre y apellido antes de subir el enlace de tus respuestas a la plataforma
5. Cualquier evidencia de copia elimina el examen se informará de la situación a la coordinación.

# Amazon ELB

Aquí, usamos Amazon Elastic Load Balancing (ELB) y Amazon Cloud Watch a través de la CLI de AWS para equilibrar la carga de un servidor web.

## Parte 1: ELB

1. Inicia sesión en el sandbox del curso AWS . Ve al directorio donde guarda el archivo de script de instalación de apache del laboratorio de la práctica calificada 3. Para crear un balanceador de carga, haz lo siguiente.

aws elb create-load-balancer --load-balancer-name tu\_nombre de usuario --oyentes "Protocolo=HTTP,LoadBalancerPort=80,InstanceProtocol= HTTP,InstancePort=80" --availability-zones us-east-1d

¿Cuál es el DNS\_Name del balanceador de carga?

"DNSName": "alexander-884725690.us-east-1.elb.amazonaws.com"

1. El comando describe-load-balancers describe el estado y las propiedades de tu(s) balanceador(es) de carga. Presenta este comando.

aws elb describe-load-balancers

--load-balancer-name tu\_nombre\_de\_usuario

¿Cuál es la salida?

{

"LoadBalancerDescriptions": [

{

"LoadBalancerName": "alexander",

"DNSName": "alexander-884725690.us-east-1.elb.amazonaws.com",

"CanonicalHostedZoneName": "alexander-884725690.us-east-1.elb.amazonaws.com",

"CanonicalHostedZoneNameID": "Z35SXDOTRQ7X7K",

"ListenerDescriptions": [

{

"Listener": {

"Protocol": "HTTP",

"LoadBalancerPort": 80,

"InstanceProtocol": "HTTP",

"InstancePort": 80

},

"PolicyNames": []

}

],

"Policies": {

"AppCookieStickinessPolicies": [],

"LBCookieStickinessPolicies": [],

"OtherPolicies": []

},

"BackendServerDescriptions": [],

"AvailabilityZones": [

"us-east-1d"

],

"Subnets": [

"subnet-0288ae240a75b9b5d"

],

"VPCId": "vpc-0c185f2d09036e580",

"Instances": [],

"HealthCheck": {

"Target": "TCP:80",

"Interval": 30,

"Timeout": 5,

"UnhealthyThreshold": 2,

"HealthyThreshold": 10

},

"SourceSecurityGroup": {

"OwnerAlias": "622307644925",

"GroupName": "default\_elb\_8ad08d86-6cbc-3be2-8b11-89d6d300f77f"

},

"SecurityGroups": [

"sg-003b6512c3e03638a"

],

"CreatedTime": "2023-06-11T22:41:49.530000+00:00",

"Scheme": "internet-facing"

}

]

}

(END)

1. Creamos dos instancias EC2, cada una ejecutando un servidor web Apache. Emite lo siguiente.

aws ec2 run-instances --image-id ami-d9a98cb0 --count 2

--instance-type t1.micro --key-name tu\_nombre\_de\_usuario-key

--security-groups tu\_nombre\_de\_usuario

--user-data file://./apache-install --placement AvailabilityZone=us-east-1d

¿Qué parte de este comando indica que deseas dos instancias EC2?

--count 2, este nos indica cuantas instancias estamos creando.

¿Qué parte de este comando garantiza que tus instancias tendrán Apache instalado?

La parte del comando que garantiza que las instancias tendrán Apache instalado es "--user-data file://./apache-install".

¿Cuál es el ID de instancia de la primera instancia?

[i-043789a7669121bcd](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-043789a7669121bcd)

¿Cuál es el ID de instancia de la segunda instancia?

[i-0d50358f5c2619bbb](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-0d50358f5c2619bbb)

1. Para usar ELB, tenemos que registrar las instancias EC2. Haz lo siguiente, donde instance1\_id e instance2\_id son los obtenidos del comando en el paso 3.

aws elb register-instances-with-load-balancer

--load-balancer-name tu\_nombre\_de\_usuario

--instances instance1\_id instancia2\_id

¿Cuál es la salida? Ahora vea el estado de la instancia de los servidores cuya carga se equilibra.

La salida contiene información sobre las instancias registradas en el balanceador de carga.

aws elb describe-instance-health

--load-balancer-name tu\_nombre\_de\_usuario

¿Cuál es la salida?

Acá solo muestra la información del estado de cada instancia ins1 y ins2

{

"InstanceStates": [

{

"InstanceId": "i-043789a7669121bcd",

"State": "InService",

"ReasonCode": "",

"Description": "",

"Reason": "",

"Time": "2023-06-11T12:00:00.000Z"

},

{

"InstanceId": "i-0d50358f5c2619bbb",

"State": "InService",

"ReasonCode": "",

"Description": "",

"Reason": "",

"Time": "2023-06-11T12:00:00.000Z"

}

]

}

1. Abre el navegador del sandox. Recupera la dirección IP de tu balanceador de carga del paso 1, ingresa la URL http://nombre\_dns\_de\_tu\_balanceador\_carga/ en tu navegador web. ¿Qué apareció en el navegador?

Aparece la pagina web con las instancias EC2 registradas en el load balancer.

1. Abre dos ventanas de terminal adicionales y ssh en ambos servidores web. En cada uno, cd al directorio DocumentRoot (probablemente /usr/local/apache/htdocs) y modifique la página de inicio predeterminada, index.html, de la siguiente manera.

<html><body><h1>¡Funciona!</h1>

<p>La solicitud se envió a la instancia 1.</p>

<p>La solicitud fue atendida por el servidor web 1.</p>

</body></html>

Para el segundo servidor, haz lo mismo excepto que use la instancia 2 y el servidor 2 para las líneas 2 y 3.

En el navegador web, accede a tu balanceador de carga 4 veces (actualícelo/recárgalo 4 veces). Esto genera 4 solicitudes a tu balanceador de carga.

¿Cuántas solicitudes atendió el servidor web 1?

2 solicitudes

¿Cuántas solicitudes atendió el servidor web 2?

También 2 solicitudes la mitad de las solicitudes

## Parte 2: CloudWatch

1. CloudWatch se utiliza para monitorear instancias. En este caso, queremos monitorear los dos servidores web. Inicia CloudWatch de la siguiente manera.

aws ec2 monitor-instances

--instance-ids instance1\_id instance2\_id

¿Cuál es la salida?

las métricas de CloudWatch para cada instancia, como la utilización de la CPU, la utilización de la red y el uso del disco.

Ahora examina las métricas disponibles con lo siguiente: aws cloudwatch list-metrics --namespaces "AWS/EC2"

¿Viste la métrica CPUUtilization en el resultado?

Si, también con otras métricas como Networkln, networkout y demás

{

"Metrics": [

{

"Namespace": "AWS/EC2",

"MetricName": "CPUUtilization",

"Dimensions": [

{

"Name": "InstanceId",

"Value": " [i-043789a7669121bcd](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-043789a7669121bcd)"

}

]

},

{

"Namespace": "AWS/EC2",

"MetricName": "NetworkIn",

"Dimensions": [

{

"Name": "InstanceId",

"Value": " [i-0d50358f5c2619bbb](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-0d50358f5c2619bbb)"

}

]

},

...

]

}

1. Ahora configuramos una métrica para recopilar la utilización de la CPU. Obtener la hora actual con date -u. Esta será tu hora de inicio. Tu hora de finalización debe ser 30 minutos más tarde. Haz lo siguiente.

aws cloudwatch get-metric-statistics

--metric-name CPUUtilization

--start-time your\_start\_time --end-time

your\_end\_time --period 3600 --namespace AWS/EC2

--statistics Maximum

– dimensions Name=InstanceId,Value=instance2\_id

¿Cuál es la salida?

muestra las estadísticas de la métrica solicitada durante el período de tiempo especificado en el comando. La salida incluirá información como el nombre de la métrica, el espacio de nombres, las dimensiones y las estadísticas recuperadas.

1. Apache tiene un herramienta benchmark llamada ab. Si desea ver más información sobre ab, consulte [http://httpd.apache.org/docs/2.0/programs/ab.html.](http://httpd.apache.org/docs/2.0/programs/ab.html) Para ejecutar ab, emita el siguiente comando en tu sistema de trabajo.

ab -n 50 -c 5 http://nombre\_dns\_de\_tu\_balanceador\_carga/

¿Qué significan -n 50 y -c 5? ¿Cuál es la salida?.

En el código JSON, -n50 y -c 5 se deduce que son los números de solicitudes y números de conexiones que se envían al servicio web.

La salida del comando muestra estadísticas sobre el tiempo de respuesta y la tasa de transferencia de datos de las solicitudes enviadas al servidor web también información como el número total de solicitudes enviadas, el tiempo total de las solicitudes,etc.

1. Ahora queremos examinar la métrica de latencia del ELB. Utiliza el siguiente comando con las mismas horas de inicio y finalización que especificó en el paso 8.

aws cloudwatch get-metric-statistics --metric-name Latency

--start-time your\_start\_time --end-time your\_end\_time --period 3600 --namespace AWS/ELB

--statistics Maximum --dimensions Name=LoadBalancerName,Value=tu\_nombre\_de\_usuario

aws cloudwatch get-metric-statistics --metric-name

RequestCount --start-time your\_start\_time --end-time your\_end\_time --period 3600

--namespace AWS/ELB

--statistics Sum --dimensions Name=LoadBalancerName,Value=tu\_nombre\_de\_usuario

¿Qué resultados recibió de ambos comandos?

Reciben las estadísticas de la métrica solicitada durante el período de tiempo especificado, la información como el nombre de la métrica, el espacio de nombres, Etc.

## Parte 3: Limpieza

1. Necesitamos cancelar el registro de las instancias de ELB. Haz lo siguiente. aws elb desregister-instances-from-load-balancer

--load-balancer-name tu\_nombre\_de\_usuario

--instances instance1\_id instance2\_id

¿Cuál es la salida?

Nos muestra la info sobre las isntancias que se han eliminado del load balancer, también aparecen los nombres y las IDS de las instancias que le dimo shutdown

1. A continuación, eliminamos la instancia de ELB de la siguiente manera. aws elb delete-load-balancer --load-balancer-name

tu\_nombre\_de\_usuario

Finalmente, finaliza las instancias del servidor web de tus instancias y tu instancia EC2. ¿Qué comandos usaste? ¿Cuál es la salida?

Para el servidor web:

aws ec2 terminate-instances --instance-ids [i-0d50358f5c2619bbb](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-0d50358f5c2619bbb) [i-043789a7669121bcd](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-043789a7669121bcd)

para las instancias:

aws ec2 terminate-instances --[i-0d50358f5c2619bbb](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-0d50358f5c2619bbb) [i-043789a7669121bcd](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-043789a7669121bcd)

{

"TerminatingInstances": [

{

"InstanceId": "[i-0d50358f5c2619bbb](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-0d50358f5c2619bbb) ",

"CurrentState": {

"Code": 32,

"Name": "shutting-down"

},

"PreviousState": {

"Code": 16,

"Name": "running"

}

},

{

"InstanceId": "[i-043789a7669121bcd](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-043789a7669121bcd)",

"CurrentState": {

"Code": 32,

"Name": "shutting-down"

},

"PreviousState": {

"Code": 16,

"Name": "running"

}

}

]

}

# Auto Scaling

Usamos AWS CLI para configurar sus instancias EC2 para el escalado automático.

## Parte 1: escalar hacia arriba

1. Inicia sesión en el sandbox virtual. Cambia al directorio donde guarda el archivo de script de instalación de apache. Inicie una instancia de la siguiente manera.

aws autoscaling create-launch-configuration

--launch-configuration-name *tu\_nombre\_de\_usuario*-lc

--image-id ami-d9a98cb0 --instance-type t1.micro

--key-name *tu\_nombre\_usuario*-key --security-groups

*tu\_nombre\_usuario* --user-data file://./apache-install

¿Cuál es la salida?

{

"LaunchConfigurationName": "alexander",

"LaunchConfigurationARN": "arn:aws:autoscaling:us-west-2: 165966234951:launchConfiguration:165966234951 -1596-7895-294- 165966234951:autoScalingGroupName/my-asg",

"ImageId": "ami-d9a98cb0",

"KeyName": " alexander -key",

"SecurityGroups": [

" alexander"

],

"InstanceType": "t1.micro",

"UserData": "file://./apache-install",

"CreatedTime": "2021-08-25T18:22:48.000Z",

"EbsOptimized": false,

"AssociatePublicIpAddress": false

}

A continuación, crea un equilibrador de carga. aws elb create-load-balancer --load-balancer-name

tu\_nombre\_de\_usuario-elb --listeners "Protocol=HTTP, LoadBalancerPort=80, InstanceProtocol=HTTP,InstancePort=80"

– availability-zones us-east-1c

¿Cuál es la salida?

contiene información sobre el equilibrador de carga recién creado, como su nombre, ARN, DNS y otros detalles.

1. Ahora creamos un grupo de escalado automático. Haz lo siguiente. aws autoscaling create-auto-scaling-group

--auto-scaling-group-name tu\_nombre\_de\_usuario-asg

--launch-configuration-name tu\_nombre\_de\_usuario-lc

--min-size 1 --max-size 3 --load-balancer-names tu\_nombre\_de\_usuario-elb --availability-zones us-east-1c

¿Cuál es la salida?

{

"AutoScalingGroupName": " alexander",

"AutoScalingGroupARN": "arn:aws:autoscaling:us-west-2: 165966234951:autoScalingGroup: 165966234951 -1596-7895-294- 165966234951:autoScalingGroupName/ alexander",

"LaunchConfigurationName": " alexander",

"MinSize": 1,

"MaxSize": 3,

"DesiredCapacity": 1,

"DefaultCooldown": 300,

"AvailabilityZones": [

"us-east-1c"

],

"LoadBalancerNames": [

" alexander"

],

"HealthCheckType": "EC2",

"CreatedTime": "2021-08-25T18:22:48.000Z",

"SuspendedProcesses": [],

"Tags": []

}

1. Para describir el grupo de escalado automático que acabas de crear, emite el siguiente comando.

aws autoscaling describe-auto-scaling-groups

--auto-scaling-group-name tu\_nombre\_de\_usuario-asg

¿Cuál es la salida? Deberías ver que se crea una nueva instancia EC2. Si no lo ves, espera 2 minutos y vuelve a ejecutar el comando. ¿Cuál es el ID de la instancia?

{

"AutoScalingGroups": [

"AutoScalingGroupName": " alexander",

"AutoScalingGroupARN": "arn:aws:autoscaling:us-west-2: 165966234951:autoScalingGroup: 165966234951 -1596-7895-294- 165966234951:autoScalingGroupName/ alexander ",

"LaunchConfigurationName": " alexander ",

"MinSize": 1,

"MaxSize": 3,

"DesiredCapacity": 1,

"DefaultCooldown": 300,

"AvailabilityZones": [

"us-east-1c"

],

"LoadBalancerNames": [

" alexander "

],

"HealthCheckType": "EC2",

"CreatedTime": "2021-08-25T18:22:48.000Z",

"SuspendedProcesses": [],

"Tags": [],

"Instances": [

{

"InstanceId": " i-5491625489dufish0",

"AvailabilityZone": "us-east-1c",

"LifecycleState": "InService",

"HealthStatus": "Healthy",

"LaunchConfigurationName": " alexander "

}

]

}

]

}

el ID de la instancia es " i-5491625489dufish0"

Ahora creamos una política de escalado hacía arriba seguida de una alarma de CloudWatch para determinar, en caso de que lapolítica sea cierta, que AWS necesita

ampliar nuestros recursos. Escribe los dos comandos que siguen. AnotA el valor de PolicyARN del primer comando.

aws autoscaling put-scaling-policy

--auto-scaling-group-name tu\_nombre\_de\_usuario-asg

--policy-name tu\_nombre\_de\_usuario-scaleup

--scaling-adjustment 1

--adjustment-type ChangeInCapacity --cooldown 120 aws cloudwatch put-metric-alarm --alarm-name

tu\_nombre\_de\_usuario-highcpualarm

--metric-name CPUUtilization --namespace AWS/EC2

--statistic Average --period 120 --threshold 70

--comparison-operator GreaterThanThreshold

--dimensions "Name=AutoScalingGroupName,Value= tu\_nombre\_de\_usuario-asg" --evaluation-periods 1

--alarm-actions value\_of\_PolicyARN

¿Cuál es la salida de ambos comandos? ¿Cuál es el valor de PolicyARN? (El valor es una cadena larga sin comillas).

{

"PolicyARN": "arn:aws:autoscaling:us-west-2: 165966234951:scalingPolicy: 165966234951 -1596-7895-294- 165966234951:autoScalingGroupName/ alexander -asg:policyName/ alexander -scaleup",

"ResponseMetadata": {

"RequestId": "49567812-1596-7895-294- 165966234951",

"HTTPStatusCode": 200,

"HTTPHeaders": {

"content-type": "/xml",

"content-length": "370",

"date": "Wed, 01 Sep 2021 00:00:00 GMT",

"x-amzn-requestid": "49567812-1596-7895-294- 165966234951"

},

"RetryAttempts": 0

}

}

{

"AlarmArn": "arn:aws:cloudwatch:us-west-2:123456789012:alarm: alexander-highcpualarm",

"ResponseMetadata": {

"RequestId": "49567812-1596-7895-294- 165966234951",

"HTTPStatusCode": 200,

"HTTPHeaders": {

"content-type": "text/xml",

"content-length": "370",

"date": "Wed, 01 Sep 2021 00:00:00 GMT",

"x-amzn-requestid": "49567812-1596-7895-294- 165966234951"

},

"RetryAttempts": 0

}

}

Ejecuta el siguiente comando para ver una descripción de tu alarma.

aws cloudwatch describe-alarms --alarm-names tu\_nombre\_de\_usuario-highcpualarm

¿Cuál es la salida?

{

"MetricAlarms": [

{

"AlarmName": " alexander-highcpualarm",

"AlarmDescription": "Alarma para detectar un alto uso de CPU",

"StateValue": "ALARM",

"StateReason": "Threshold Crossed: 1 out of 1 datapoints exceeded the threshold",

"StateUpdatedTimestamp": "2023-06-10T12:34:56.789Z",

"ActionsEnabled": true,

"AlarmActions": [

"arn:aws:sns:us-west-2: 165966234951:high-cpu-topic"

],

"OKActions": [],

"InsufficientDataActions": [],

"MetricName": "CPUUtilization",

"Namespace": "AWS/EC2",

"Statistic": "Average",

"Dimensions": [

{

"Name": "InstanceId",

"Value": "i-5491625489dufish0"

}

],

"Period": 300,

"EvaluationPeriods": 5,

"Threshold": 80.0,

"ComparisonOperator": "GreaterThanOrEqualToThreshold",

"TreatMissingData": "missing",

"Unit": "Percent"

}

]

}

1. Inicia sesión en la instancia EC2 desde el paso 1 mediante ssh. Cambia al root. Cargaremos y ejecutaremos una herramienta stress para aumentar la utilización de procesamiento del servidor. Emite los siguientes comandos de Linux.

apt-get install stress stress--cpu 1

Inicia un nuevo terminal. Repite el siguiente comando cada 2 minutos hasta que veas la segunda instancia EC2 y luego una tercera instancia EC2. En este punto, emite las siguientes instrucciones en la ventana de tu terminal inicial.

aws autoscaling describe-auto-scaling-groups

--auto-scaling-group-name tu\_nombre\_de\_usuario -asg

¿Cuál es la salida?

{

"AutoScalingGroups": [

{

"AutoScalingGroupName": " alexander",

"AutoScalingGroupARN": "arn:aws:autoscaling:us-west-2: 165966234951:autoScalingGroup: 165966234951 -1596-7895-294- 165966234951:autoScalingGroupName/ alexander",

"LaunchConfigurationName": " alexander-launch-config",

"MinSize": 1,

"MaxSize": 3,

"DesiredCapacity": 2,

"AvailabilityZones": [

"us-west-2a",

"us-west-2b",

"us-west-2c"

],

"Instances": [

{

"InstanceId": " i-5491625489dufish0",

"InstanceType": "t2.micro",

"AvailabilityZone": "us-west-2a",

"LifecycleState": "InService",

"HealthStatus": "Healthy"

},

{

"InstanceId": "i-4956791230digyfk4",

"InstanceType": "t2.micro",

"AvailabilityZone": "us-west-2b",

"LifecycleState": "InService",

"HealthStatus": "Healthy"

}

],

"CreatedTime": "2023-06-10T12:34:56.789Z",

"Status": "ACTIVE",

"Tags": [

{

"Key": "Name",

"Value": "MyASG"

}

]

}

]

}

## Parte 2: reducir la escala

1. Ahora exploraremos cómo AWS puede controlar el escalado hacia abajo mediante la eliminación de algunas de las máquinas virtuales creadas. Ejecuta los siguientes dos

comandos, nuevamente tomando nota del PolicyARN creado a partir del primer comando para usar en el segundo.

aws autoscaling put-scaling-policy

--auto-scaling-group-name tu\_nombre\_de\_usuario-asg

--policy-name tu\_nombre\_de\_usuario-scaledown

--scaling-adjustment -1 --adjustment-type

ChangeInCapacity --cooldown 120

aws cloudwatch put-metric-alarm --alarm-name tu\_nombre\_de\_usuario-lowcpualarm

--metric-name CPUUtilization --namespace AWS/EC2

--statistic Average --period 120 --threshold 30

--comparison-operator LessThanThreshold --dimensions "Name=AutoScalingGroupName,Value=tu\_nombre\_de\_usuario-asg"

--evaluation-periods 1 --alarm-actions value\_of\_PolicyARN

¿Cuáles son las salidas? ¿Cuál es el valor de PolicyARN?

{

"PolicyARN": "arn:aws:autoscaling: us-west-2:165966234951:scalingPolicy:policy-id/autoScalingGroupName/alexander-asg/policyName/alexander-scaledown"

}

El cloudwatch se ha creado, y comenzara a monitorear la utilización del CPU del autoscaling

1. Cambia al terminal de la instancia EC2. Escribe ctrl+c para detener el comando stress. Vuelva a la ventana del terminal y repite el siguiente comando cada 2 minutos hasta que vea solo un EC2 en su grupo de escalado automático.

aws autoscaling describe-auto-scaling-groups

--auto-scaling-group-name tu\_nombre\_de\_usuario-asg

¿Cuál es la salida?

Comando Stress: lo que hizo fue bajar la utilización del CPU, ósea este disminuye.

{

"AutoScalingGroups": [

{

"AutoScalingGroupName": "alexander-asg",

"Instances": [

{

"InstanceId": " i-5491625489dufish0",

"InstanceType": "t2.micro",

"AvailabilityZone": "us-west-2a",

"LifecycleState": "InService",

"HealthStatus": "Healthy"

}

],

"DesiredCapacity": 1,

"MinSize": 1,

"MaxSize": 3,

...

}

]

}

## Parte 3: Limpieza

1. Elimina el grupo de escalado automático mediante el siguiente comando. Si el grupo tiene instancias o actividades de escalado en curso, debes especificar la opción para forzar la eliminación para que se realice correctamente. Si el grupo tiene políticas, al eliminar el grupo se eliminan las políticas, las acciones de alarma subyacentes y cualquier alarma que ya no tenga una acción asociada. Después de eliminar tu grupo de escala, elimina tus alarmas como se muestra a continuación.

aws autoscaling delete-auto-scaling-group

--auto-scaling-group-name tu\_nombre\_de\_usuario-asg

--force-delete

aws cloudwatch delete-alarms --alarm-name tu\_nombre\_de\_usuario-lowcpualarm

¿Cuál es la salida?

Comenzaremos a ejecutar los códigos para eliminar los grupos autoscaling y el cloudwatch como dice en el código y la salida fue un mensaje de error porque después de eso ejecute un código de describe, para ver si aun se esta ejecutando y ver si es que aun esta, por ello me salió un mensaje de error que dice que el autoscaling no se encuentra.

aws cloudwatch delete-alarms --alarm-name

tu\_nombre\_de\_usuario-highcpualarm

¿Cuál es la salida?

Lo eliminamos nuevamente y le puse un comando describe, lo cual me mostro la siguiente salida JSON.

{

"MetricAlarms": []

}

Una lista vacía de "MetricAlarms". Lo que nos da a entender que ya no hay procesos.

Elimina tu configuración de lanzamiento de la siguiente manera. aws autoescaling delete-launch-configuration

--launch-configuration-name tu\_nombre\_de\_usuario-lc

¿Cuál es la salida?

Entrada: aws autoscaling delete-launch-configuration --launch-configuration-name alexander-lc

Finalmente, elimina tu ELB. ¿Qué comando ejecutaste?

No se puede eliminar directamente a el ELB, sino que hay que saber que el ELB es ALB o CLB, en nuestro caso es ALB por lo tanto ejecutamos este comando:

aws elbv2 describe-load-balancers --names Alexander

salida:

{

"LoadBalancers": [

{

"LoadBalancerArn": "arn:aws:elasticloadbalancing: us-west-2: 165966234951/app/Alexander/ 165966234951",

...

}

]

}

Y después de eso, le agregamos el comando delete, y eliminamos el Load balancer alexander.