



Amazon ELB - Auto Scaling

Indicaciones

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
```

```
aws elb create-load-balancer --load-balancer-name kimberly-salazar --listeners  
"Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80"  
--availability-zones us-east-1d
```



¿Cuál es el DNS_Name del balanceador de carga? Es el siguiente:

```
[cloudshell-user@ip-10-10-67-40 ~]$ aws elb create-load-balancer --load-balancer-name kimberly-salazar --listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" --availability-zone us-east-1d

{
    "DNSName": "kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com"
}
[cloudshell-user@ip-10-10-67-40 ~]$
```

"DNSName": "kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com"

2. 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

```
aws elb describe-load-balancers --load-balancer-name kimberly-salazar
```

¿Cuál es la salida? La salida es la descripción del estado y propiedades del balanceador de carga creado. Esta información puede incluir detalles como el nombre del balanceador de carga, los grupos de seguridad asociados, las instancias de EC2 etc.

```
[cloudshell-user@ip-10-10-67-40 ~]$ aws elb describe-load-balancers --load-balancer-name kimberly-salazar
{
    "LoadBalancerDescriptions": [
        {
            "LoadBalancerName": "kimberly-salazar",
            "DNSName": "kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com",
            "CanonicalHostedZoneName": "kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com",
            "CanonicalHostedZoneNameID": "Z35SXDOTRQ7X7K",
            "ListenerDescriptions": [
                {
                    "Listener": {
                        "Protocol": "HTTP",
                        "LoadBalancerPort": 80,
                        "InstanceProtocol": "HTTP",
                        "InstancePort": 80
                    },
                    "PolicyNames": []
                }
            ],
            "LoadBalancerDescriptions": [
                {
                    "LoadBalancerName": "kimberly-salazar",
                    "DNSName": "kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com",
                    "CanonicalHostedZoneName": "kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com",
                    "CanonicalHostedZoneNameID": "Z35SXDOTRQ7X7K",
                    "ListenerDescriptions": [
                        {
                            "Listener": {
                                "Protocol": "HTTP",
                                "LoadBalancerPort": 80,
                                "InstanceProtocol": "HTTP",
                                "InstancePort": 80
                            },
                            "PolicyNames": []
                        }
                    ],
                    "HealthCheck": {
                        "Interval": 30,
                        "Timeout": 5,
                        "UnhealthyThreshold": 3,
                        "HealthyThreshold": 3
                    }
                }
            ]
        }
    ]
}
```

```

    "LoadBalancerName": "kimberly-salazar",
    "DNSName": "kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com",
    "CanonicalHostedZoneName": "kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com",
    "CanonicalHostedZoneNameID": "Z35SXDDOTRQ7X7K",
    "ListenerDescriptions": [
        {
            "Listener": {
                "Protocol": "HTTP",
                "LoadBalancerPort": 80,
                "InstanceProtocol": "HTTP",
                "InstancePort": 80
            },
            "PolicyNames": []
        }
    ],
    "Policies": {
        "AppCookieStickinessPolicies": [],
        "LBCookieStickinessPolicies": [],
        "OtherPolicies": []
    },
    "BackendServerDescriptions": [],
    "AvailabilityZones": [
        "us-east-1d"
    ],
    "Subnets": [
        "subnet-0b4152b6222868dc4"
    ],
    "VPCId": "vpc-071d26095bb38dba4",
    "Instances": [],
    "HealthCheck": {
        "Target": "TCP:80",
        "Interval": 30,
        "Timeout": 5,
        "UnhealthyThreshold": 2,
        "HealthyThreshold": 10
    },
    "SourceSecurityGroup": {
        "OwnerAlias": "091101640480",
        "GroupName": "default_elb_a4846ce7-3eeb-38c8-a9ca-843cf2c72895"
    },
    "SecurityGroups": [
        "sg-0a6b8dba2409cbe7d"
    ],
    "CreatedTime": "2023-06-11T22:24:34.290000+00:00",
    "Scheme": "internet-facing"
}
]
}

```

3. 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
aws ec2 run-instances --image-id ami-d9a98cb0 --count 2 --instance-type t1.micro
--key-name kim_salazar-key --security-groups kim_salazar --user-data
file://./apache-install --placement AvailabilityZone=us-east-1d

```

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

```
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 garantiza que tus instancias tendrán Apache instalado?

```
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
```

¿Cuál es el ID de instancia de la primera instancia? El ID de la primera instancia es **i-08623a7a651fa1bb7**

```
[cloudshell-user@ip-10-4-57-187 ~]$ aws ec2 run-instances --image-id ami-d9a98cb0 --count 2 --instance-type t1.micro --key-name kim_salazar-key --security-groups kim_salazar --user-data file://./apache-install --placement AvailabilityZone=us-east-1d  
{  
    "Groups": [],  
    "Instances": [  
        {  
            "AmiLaunchIndex": 1,  
            "ImageId": "ami-d9a98cb0",  
            "InstanceId": "i-08623a7a651fa1bb7",  
            "InstanceType": "t1.micro",  
            "KernelId": "aki-88aa75e1",  
            "KeyName": "kim_salazar-key",  
            "LaunchTime": "2023-06-14T20:43:45+00:00",  
            "Monitoring": {  
                "State": "disabled"  
            },  
            "Placement": {  
                "AvailabilityZone": "us-east-1d",  
                "GroupName": "",  
                "Tenancy": "default"  
            },  
            "PrivateDnsName": "ip-172-31-30-78.ec2.internal",  
            "PrivateIpAddress": "172.31.30.78",  
            "ProductCodes": [],  
            "PublicDnsName": "ip-172-31-30-78.us-east-1.compute.amazonaws.com",  
            "PublicIpAddress": "172.31.30.78",  
            "State": "pending",  
            "StateReason": {  
                "Code": "pending",  
                "Message": "pending: Starting instance",  
                "Type": "status"  
            },  
            "SubnetId": "subnet-00000000",  
            "VpcId": "vpc-00000000"  
        }  
    ]  
}
```

¿Cuál es el ID de instancia de la segunda instancia? El ID de la segunda instancia es **i-0f55925035f9c56c6**

```
{
    "AmiLaunchIndex": 0,
    "ImageId": "ami-d9a98cb0",
    "InstanceId": "i-0f55925035f9c56c6",
    "InstanceType": "t1.micro",
    "KernelId": "aki-88aa75e1",
    "KeyName": "kim_salazar-key",
    "LaunchTime": "2023-06-14T20:43:45+00:00",
    "Monitoring": {
        "State": "disabled"
    },
    "Placement": {
        "AvailabilityZone": "us-east-1d",
        "GroupName": "",
        "Tenancy": "default"
    },
    "PrivateDnsName": "ip-172-31-27-85.ec2.internal",
    "PrivateIpAddress": "172.31.27.85",
    "ProductCodes": [],
    "PublicDnsName": "",
    "State": {
        "Code": 0,
        "Name": "pending"
    }
}
```

4. 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
aws elb register-instances-with-load-balancer --load-balancer-name kimberly-salazar
--instances i-0c0a95df6a3426dd0 i-02ebbc93d652d84fb
```

¿Cuál es la salida? La salida muestra que ya se registraron exitosamente las instancias con el balanceador de carga.

```
[cloudshell-user@ip-10-4-57-187 ~]$ aws elb register-instances-with-load-balancer --load-balancer-name kimberly-salazar --instances i-08623a7a651fa1bb7 i-0f55925035f9c56c6
{
    "Instances": [
        {
            "InstanceId": "i-0f55925035f9c56c6"
        },
        {
            "InstanceId": "i-08623a7a651fa1bb7"
        }
    ]
}
```

Ahora vea el estado de la instancia de los servidores cuya carga se equilibra.

```
aws elb describe-instance-health
```

```
--load-balancer-name tu_nombre_de_usuario
```

```
aws elb describe-instance-health --load-balancer-name kimberly-salazar
```

¿Cuál es la salida? La salida es el estado de la instancia de los servidores con una carga equilibrada.

```
[cloudshell-user@ip-10-4-57-187 ~]$ aws elb describe-instance-health --load-balancer-name kimberly-salazar
{
  "InstanceStates": [
    {
      "InstanceId": "i-08623a7a651fa1bb7",
      "State": "OutOfService",
      "ReasonCode": "Instance",
      "Description": "Instance has failed at least the UnhealthyThreshold number of health checks consecutively."
    },
    {
      "InstanceId": "i-0f55925035f9c56c6",
      "State": "OutOfService",
      "ReasonCode": "Instance",
      "Description": "Instance has failed at least the UnhealthyThreshold number of health checks consecutively."
    }
  ]
}
```

5. Abre el navegador del sandbox. 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?

← → C ⚠ No es seguro | kimberly-salazar-49969483.us-east-1.elb.amazonaws.com

It works!

This is the default web page for this server.

The web server software is running but no content has been added, yet.

Pero si nos saliera de la siguiente forma: Realizamos los siguientes pasos ya que esto se debe que no se instaló correctamente apache y es una manera de solucionarlo

← → C ⓘ kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com ☆



No se puede acceder a este sitio web

Comprueba si hay un error de escritura en kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com.

DNS_PROBE_FINISHED_NXDOMAIN

[Volver a cargar](#)

```
ssh -i devenv-key.pem ubuntu@id intancia  
sudo nano /etc/apt/sources.list
```

The screenshot shows the AWS CloudShell interface with three tabs: us-east-1, us-east-1, and us-east-1. The us-east-1 tab is active and displays the file /etc/apt/sources.list. The file contains the following content:

```
File: /etc/apt/sources.list  
File: /etc/apt/sources.list  
deb-src http://security.ubuntu.com/ubuntu precise-security main  
deb http://security.ubuntu.com/ubuntu precise-security universe  
deb-src http://security.ubuntu.com/ubuntu precise-security universe  
# deb http://security.ubuntu.com/ubuntu precise-security multiverse  
# deb-src http://security.ubuntu.com/ubuntu precise-security multiverse  
deb http://old-releases.ubuntu.com/ubuntu precise main restricted universe multiverse  
deb http://old-releases.ubuntu.com/ubuntu precise-updates main restricted universe multiverse  
deb http://old-releases.ubuntu.com/ubuntu precise-security main restricted universe multiverse
```

```
deb http://old-releases.ubuntu.com/ubuntu precise main restricted universe  
multiverse  
deb http://old-releases.ubuntu.com/ubuntu precise-updates main restricted universe  
multiverse  
deb http://old-releases.ubuntu.com/ubuntu precise-security main restricted universe  
multiverse
```

6. 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.

```
ubuntu@ip-172-31-18-231:~$ sudo -s  
root@ip-172-31-18-231:~# cd /var/www  
root@ip-172-31-18-231:/var/www# ls  
index.html  
root@ip-172-31-18-231:/var/www# vi index.html  
root@ip-172-31-18-231:/var/www#  
DNS: kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com
```

```
us-east-1 ✘ us-east-1 ✘ us-east-1 ✘

<html><body><h1>Funciona!</h1>
<p>La solicitud se envio a la instancia 1.</p>
<p>La solicitud fue atendida por el servidor 1.</p>
</body></html>
~
~
~
~
~
~
~
"
"index.html" 4L, 145C

Enabling module authz_groupfile.
<html><body><h1>¡Funciona!</h1>
<p>La solicitud se envió a la instancia 2.</p>
<p>La solicitud fue atendida por el servidor 2.</p>
</body></html>
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```
-- INSERT --

```

← → C ⚠ No es seguro | kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com

# Funciona!

La solicitud se envio a la instancia 1.

La solicitud fue atendida por el servidor web 1.

← → C ⚠ No es seguro | kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com

# Funciona!

La solicitud se envio a la instancia 2.

La solicitud fue atendida por el servidor web 2.

**¿Cuántas solicitudes atendió el servidor web 1?**

Atendió 2 solicitudes

**¿Cuántas solicitudes atendió el servidor web 2?**

Atendió 2 solicitudes

## Parte 2: CloudWatch

**7. 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? Se está empezando a monitorear los servidores web.**

```
[cloudshell-user@ip-10-6-167-172 ~]$ aws ec2 monitor-instances --instance-ids i-011e5c4cc4e28b68f i-02d12b8bcbf639f05
{
 "InstanceMonitorings": [
 {
 "InstanceId": "i-011e5c4cc4e28b68f",
 "Monitoring": {
 "State": "pending"
 }
 },
 {
 "InstanceId": "i-02d12b8bcbf639f05",
 "Monitoring": {
 "State": "pending"
 }
 }
]
}
```

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

**aws cloudwatch list-metrics --namespace "AWS/EC2"**

**¿Viste la métrica CPUUtilization en el resultado?**

```
[cloudshell-user@ip-10-6-95-182 ~]$ aws cloudwatch list-metrics --namespace "AWS/EC2"
{
 "Metrics": [
 {
 "Namespace": "AWS/EC2",
 "MetricName": "MetadataNoToken",
 "Dimensions": [
 {
 "Name": "InstanceId",
 "Value": "i-06ad991e7ee214ff7"
 }
]
 },
 {
 "Namespace": "AWS/EC2",
 "MetricName": "StatusCheckFailed_System",
 "Dimensions": [
 {
 "Name": "InstanceId",
 "Value": "i-06ad991e7ee214ff7"
 }
]
 }
 skipping...
 {
 "Metrics": [
 {
 "Namespace": "AWS/EC2",
 "MetricName": "MetadataNoToken",
 "Dimensions": [
 {
 "Name": "InstanceId",
 "Value": "i-06ad991e7ee214ff7"
 }
]
 }
]
 }
}
```

```
{
 "Namespace": "AWS/EC2",
 "MetricName": "CPUUtilization",
 "Dimensions": [
 {
 "Name": "InstanceId",
 "Value": "i-06ad991e7ee214ff7"
 }
]
}.
```

8. 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
aws cloudwatch get-metric-statistics --metric-name CPUUtilization --start-time 2023-06-15T01:23:27Z --end-time 2023-06-15T01:53:27Z --period 3600
--namespace AWS/EC2 --statistics Maximum --dimensions
Name=InstanceId,Value=i-0f55925035f9c56c6
```

¿Cuál es la salida? Se obtiene las estadísticas de la métrica CPUUtilization

```
[cloudshell-user@ip-10-6-95-182 ~]$ date -u
Thu Jun 15 01:23:27 UTC 2023

[cloudshell-user@ip-10-6-95-182 ~]$ aws cloudwatch get-metric-statistics --metric-name CPUUtilization --start-time 2023-06-15T01:23:27Z --end-time 2023-06-15T01:53:27Z --period 3600 --namespace AWS/EC2 --statistics Maximum --dimensions Name=InstanceId,Value=i-0f55925035f9c56c6
{
 "Label": "CPUUtilization",
 "Datapoints": [
 {
 "Timestamp": "2023-06-15T01:23:00+00:00",
 "Maximum": 0.508474576271188,
 "Unit": "Percent"
 }
]
}[cloudshell-user@ip-10-6-95-182 ~]$
```

9. 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>. Para ejecutar ab, emita el siguiente comando en tu sistema de trabajo.

```
ab -n 50 -c 5 http://nombre_dns_de_tu_balanceador_carga/
```

[ab -n 50 -c 5 http://kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com/](http://kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com/)

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

-n 50 significa que se realizarán un total de 50 solicitudes al servidor web  
-c 5 significa que se enviarán 5 solicitudes concurrentes en cada momento

```
Server Software: Apache/2.2.22
Server Hostname: kimberly-salazar-1929973315.us-east-1.elb.amazonaws.com
Server Port: 80

Document Path: /
Document Length: 149 bytes

Concurrency Level: 5
Time taken for tests: 0.023 seconds
Complete requests: 50
Failed requests: 0
Write errors: 0
Total transferred: 21200 bytes
HTML transferred: 7450 bytes
Requests per second: 2185.79 [#/sec] (mean)
Time per request: 2.288 [ms] (mean)
Time per request: 0.458 [ms] (mean, across all concurrent requests)
Transfer rate: 905.05 [Kbytes/sec] received

Connection Times (ms)
 min mean[+/-sd] median max
Connect: 0 1 0.4 1 3
Processing: 1 1 0.3 1 2
Waiting: 1 1 0.3 1 2
Total: 1 2 0.6 2 5
```

```
Connection Times (ms)
 min mean[+/-sd] median max
Connect: 0 1 0.4 1 3
Processing: 1 1 0.3 1 2
Waiting: 1 1 0.3 1 2
Total: 1 2 0.6 2 5

Percentage of the requests served within a certain time (ms)
 50% 2
 66% 2
 75% 2
 80% 3
 90% 3
 95% 3
 98% 5
 99% 5
100% 5 (longest request)
```

10. 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
```

```
aws cloudwatch get-metric-statistics --metric-name Latency --start-time
2023-06-15T01:23:27Z --end-time 2023-06-15T01:53:27Z --period 3600
--namespace AWS/ELB --statistics Maximum --dimensions
Name=LoadBalancerName,Value=kimberly-salazar
```

```
aws cloudwatch get-metric-statistics --metric-name RequestCount --start-time
2023-06-15T01:23:27Z --end-time 2023-06-15T01:53:27Z --period 3600
--namespace AWS/ELB --statistics Sum --dimensions
Name=LoadBalancerName,Value=kimberly-salazar
```

**¿Qué resultados recibió de ambos comandos? En el primer comando se recibió las estadísticas máximas de la métrica Latency y en la segunda se recibió las estadísticas de suma de la métrica Request Count.**

### Primer comando:

```
[cloudshell-user@ip-10-6-95-182 ~]$ aws cloudwatch get-metric-statistics --metric-name Latency --start-time 2023-06-15T01:23:27Z --end-time 2023-06-15T01:53:27Z --period 3600 --namespace AWS/ELB --statistics Maximum --dimensions Name=LoadBalancerName,Value=kimberly-salazar
{
 "Label": "Latency",
 "Datapoints": [
 {
 "Timestamp": "2023-06-15T01:23:00+00:00",
 "Maximum": 0.001865863800048828,
 "Unit": "Seconds"
 }
]
}
[cloudshell-user@ip-10-6-95-182 ~]$ █
```

### Segundo comando:

```
[cloudshell-user@ip-10-6-95-182 ~]$ aws cloudwatch get-metric-statistics --metric-name RequestCount --start-time 2023-06-15T01:23:27Z --end-time 2023-06-15T01:53:27Z --period 3600 --namespace AWS/ELB --statistics Sum --dimensions Name=LoadBalancerName,Value=kimberly-salazar
{
 "Label": "RequestCount",
 "Datapoints": [
 {
 "Timestamp": "2023-06-15T01:23:00+00:00",
 "Sum": 3.0,
 "Unit": "Count"
 }
]
}
[cloudshell-user@ip-10-6-95-182 ~]$ █
```

### Parte 3: Limpieza

**11. Necesitamos cancelar el registro de las instancias de ELB. Haz lo siguiente.**

```
aws elb deregister-instances-from-load-balancer
 --load-balancer-name tu_nombre_de_usuario
 --instances instance1_id instance2_id
aws elb deregister-instances-from-load-balancer --load-balancer-name kimberly-salazar --instances i-08623a7a651fa1bb7 i-0f55925035f9c56c6
```

**¿Cuál es la salida? Se eliminan las intancias del balanceador de carga.**

```
[cloudshell-user@ip-10-6-95-182 ~]$ aws elb deregister-instances-from-load-balancer --load-balancer-name kimberly-salazar --instances i-08623a7a651fa1bb7 i-0f55925035f9c56c6
{
 "Instances": []
}[cloudshell-user@ip-10-6-95-182 ~]$ █
```

**12. A continuación, eliminamos la instancia de ELB de la siguiente manera.**

```
aws elb delete-load-balancer --load-balancer-name
 tu_nombre_de_usuario
```

```
[cloudshell-user@ip-10-6-95-182 ~]$ aws elb delete-load-balancer --load-balancer-name kimberly-salazar
```

Finalmente, finaliza las instancias del servidor web de tus instancias y tu instancia EC2.

**¿Qué comandos usaste?**

```
[cloudshell-user@ip-10-6-95-182 ~]$ aws ec2 terminate-instances --instance-ids i-08623a7a651fa1bb7 i-0f55925035f9c56c6
```

**¿Cuál es la salida? Se finalizaron las instancias del servidor web y las instancias EC2.**

```
[cloudshell-user@ip-10-6-95-182 ~]$ aws ec2 terminate-instances --instance-ids i-08623a7a651fa1bb7 i-0f55925035f9c56c6
{
 "TerminatingInstances": [
 {
 "CurrentState": {
 "Code": 32,
 "Name": "shutting-down"
 },
 "InstanceId": "i-08623a7a651fa1bb7",
 "PreviousState": {
 "Code": 16,
 "Name": "running"
 },
 {
 "CurrentState": {
 "Code": 32,
 "Name": "shutting-down"
 },
 "InstanceId": "i-0f55925035f9c56c6",
 "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? Se creo una configuracion en AutoScaling.

```
[cloudshell-user@ip-10-6-13-96 ~]$ aws autoscaling create-launch-configuration --launch-configuration-name kimberly-salazar-lc --image-id ami-d9a98cb0 --instance-type t1.micro --key-name kim_salazar-key --security-groups kim_salazar --user-data file://./apache-install
[cloudshell-user@ip-10-2-19-33 ~]$ aws autoscaling describe-launch-configurations --launch-configuration-names kimberly-salazar-lc
{
 "LaunchConfigurations": [
 {
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "LaunchConfigurationARN": "arn:aws:autoscaling:us-east-1:091101640488:launchConfiguration:a473d290-cddd-49c6-9b4d-21a6220604e2:launchConfigurationName/kimberly-salazar-lc",
 "ImageId": "ami-d9a98cb0",
 "KeyName": "kim_salazar-key",
 "SecurityGroups": [
 "kim_salazar"
],
 "ClassicLinkPCSecurityGroups": [],
 "UserData": "tyevYmluL2Jhc2gC1ByZXZlbnQgyXB0LwlDgzyb29gynJpbmdpbmcgdXAgYW5SIGIudGvyyXRpdmtUgcxVcmllcwplH8vcnQgREVCSUF0X0ZST0SURUEPw5vbmludGvyyN8axZ1c1NvGdyWR1IHBlhY2thZ2VzOmFwdC1nZXQgdB0kYXRLC1nbmwyNxS1EFvVWb0ZQghcHQtZ2V0a5zdGfscBhcGFjaGlyIC15cg==",
 "InstanceType": "t1.micro",
 "KernelId": "",
 "RamdiskId": "",
 "BlockDeviceMappings": [],
 "InstanceMonitoring": {
 "Enabled": true
 },
 "CreatedTime": "2023-06-15T19:57:51.920000+00:00",
 "EbsOptimized": 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? Se creó un balanceador de carga.

```
[cloudshell-user@ip-10-2-19-33 ~]$ aws elb create-load-balancer --load-balancer-name kimberly-salazar-elb --listeners "Protocol=HTTP,LoadBalancerPort=80,InstanceProtocol=HTTP,InstancePort=80" --availability-zones us-east-1c
{
 "DNSName": "kimberly-salazar-elb-1332108950.us-east-1.elb.amazonaws.com"
}
[cloudshell-user@ip-10-2-19-33 ~]$
```

2. 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? Se creo un grupo de escalamiento automático.**

```
[cloudshell-user@ip-10-2-19-33 ~]$ aws autoscaling create-auto-scaling-group --auto-scaling-group-name kimberly-salazar-asg --launch-configuration-name kimberly-salazar-lc --min-size 1 --max-size 3 --load-balancer-names kimberly-salazar-elb --availability-zones us-east-1c
```

```
[cloudshell-user@ip-10-2-19-33 ~]$ aws autoscaling describe-auto-scaling-groups --auto-scaling-group-names kimberly-salazar-asg
{
 "AutoScalingGroups": [
 {
 "AutoScalingGroupName": "kimberly-salazar-asg",
 "AutoScalingGroupARN": "arn:aws:autoscaling:us-east-1:091101640480:autoScalingGroup:82375315-c130-4a69-a4f7-2fcc53371c73:us-east-1:autoScalingGroup/kimberly-salazar-asg",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "MinSize": 1,
 "MaxSize": 3,
 "DesiredCapacity": 1,
 "DefaultCooldown": 300,
 "AvailabilityZones": [
 "us-east-1c"
],
 "LoadBalancerNames": [
 "kimberly-salazar-elb"
],
 "TargetGroupARNs": [],
 "HealthCheckType": "EC2",
 "HealthCheckGracePeriod": 0,
 "Instances": [
 {
 "InstanceId": "i-0407fd2c4e8f0b7a9",
 "InstanceType": "t1.micro",
 "AvailabilityZone": "us-east-1c",
 "LifecycleState": "InService",
 "HealthStatus": "Healthy",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "ProtectedFromScaleIn": false
 }
]
 }
]
}
```

```
{
 "AutoScalingGroups": [
 {
 "AutoScalingGroupName": "kimberly-salazar-asg",
 "AutoScalingGroupARN": "arn:aws:autoscaling:us-east-1:091101640480:autoScalingGroup:82375315-c130-4a69-a4f7-2fcc53371c73:us-east-1:autoScalingGroup/kimberly-salazar-asg",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "MinSize": 1,
 "MaxSize": 3,
 "DesiredCapacity": 1,
 "DefaultCooldown": 300,
 "AvailabilityZones": [
 "us-east-1c"
],
 "LoadBalancerNames": [
 "kimberly-salazar-elb"
],
 "TargetGroupARNs": [],
 "HealthCheckType": "EC2",
 "HealthCheckGracePeriod": 0,
 "Instances": [
 {
 "InstanceId": "i-0407fd2c4e8f0b7a9",
 "InstanceType": "t1.micro",
 "AvailabilityZone": "us-east-1c",
 "LifecycleState": "InService",
 "HealthStatus": "Healthy",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "ProtectedFromScaleIn": false
 }
],
 "CreatedTime": "2023-06-15T20:13:23.849000+00:00",
 "SuspendedProcesses": [],
 "VPCZoneIdentifier": "",
 "EnabledMetrics": [],
 "Tags": [],
 "TerminationPolicies": [
 "Default"
],
 "NewInstancesProtectedFromScaleIn": false,
 "ServiceLinkedRoleARN": "arn:aws:iam::091101640480:role/aws-service-role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
 "TrafficSources": [
 {
 "Identifier": "kimberly-salazar-elb",
 "Type": "elb"
 }
]
 }
]
}
```

**3. 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?** Información detallada de grupo de escalado automático donde se puede visualizar una instancia.

Deberías ver que se crea una nueva instancia EC2. Si no lo ves, espera 2 minutos y vuelve a ejecutar el comando.

```
[cloudshell-user@ip-10-6-183-45 ~]$ aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name kimberly-salazar-asg
{
 "AutoScalingGroups": [
 {
 "AutoScalingGroupName": "kimberly-salazar-asg",
 "AutoScalingGroupARN": "arn:aws:autoscaling:us-east-1:091101640480:autoScalingGroup:82375315-c130-4a69-a4f7-2fcc53371c73:autoScalingGroupName/kimberly-salazar-asg",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "MinSize": 1,
 "MaxSize": 3,
 "DesiredCapacity": 3,
 "DefaultCooldown": 300,
 "AvailabilityZones": [
 "us-east-1c"
],
 "LoadBalancerNames": [
 "kimberly-salazar-elb"
],
 "TargetGroupARNs": [],
 "HealthCheckType": "EC2",
 "HealthCheckGracePeriod": 0,
 "Instances": [
 {
 "InstanceId": "i-0407fd2c4e8f0b7a9",
 "InstanceType": "t1.micro",
 "AvailabilityZone": "us-east-1c",
 "LifecycleState": "InService",
 "HealthStatus": "Healthy",
 "HealthCheckGracePeriod": null,
 "HealthCheckThreshold": null,
 "ProtectedFromScaleIn": false
 },
 ...
],
 "CreatedTime": "2023-06-15T20:13:23.849000+00:00",
 "SuspendedProcesses": [],
 "VPCZoneIdentifier": "",
 "EnabledMetrics": [],
 "Tags": [],
 "TerminationPolicies": [
 "Default"
],
 "NewInstancesProtectedFromScaleIn": false,
 "ServiceLinkedRoleARN": "arn:aws:iam::091101640480:role/aws-service-role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
 "TrafficSources": [
 {
 "Identifier": "kimberly-salazar-elb",
 "Type": "elb"
 }
]
 }
]
}
```

```
 },
 ...
],
 "CreatedTime": "2023-06-15T20:13:23.849000+00:00",
 "SuspendedProcesses": [],
 "VPCZoneIdentifier": "",
 "EnabledMetrics": [],
 "Tags": [],
 "TerminationPolicies": [
 "Default"
],
 "NewInstancesProtectedFromScaleIn": false,
 "ServiceLinkedRoleARN": "arn:aws:iam::091101640480:role/aws-service-role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
 "TrafficSources": [
 {
 "Identifier": "kimberly-salazar-elb",
 "Type": "elb"
 }
]
}
}
```

¿Cuál es el ID de la instancia?

i-0407fd2c4e8f0b7a9

**4. Ahora creamos una política de escalado hacia arriba seguida de una alarma de CloudWatch para determinar, en caso de que la política 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
```

"PolicyARN":

```
"arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:885a9e7f-b954-48dc-9a71-af49368326dd:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaleup"
```

**¿Cuál es la salida de ambos comandos? En el primer comando se obtiene la política de escalado y en la segunda se crea una alarma que se activará cuando la utilización del CPU supere el umbral de 70%.**

Primer comando:

```
[cloudshell-user@ip-10-2-19-33 ~]$ aws autoscaling put-scaling-policy --auto-scaling-group-name kimberly-salazar-asg --policy-name kimberly-salazar-scaleup --scaling-adjustment 1 --adjustment-type ChangeInCapacity --cooldown 120
{
 "PolicyARN": "arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:885a9e7f-b954-48dc-9a71-af49368326dd:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaleup",
 "Alarms": []
}
```

Segundo comando:

```
[cloudshell-user@ip-10-2-19-33 ~]$ aws cloudwatch put-metric-alarm --alarm-name kimberly-salazar-highcpualarm --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average --period 120 --threshold 70 --comparison-operator GreaterThanThreshold --dimensions "Name=AutoScalingGroupName,Value=kimberly-salazar-asg" --evaluation-periods 1 --alarm-actions arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:885a9e7f-b954-48dc-9a71-af49368326dd:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaleup
```

**¿Cuál es el valor de PolicyARN? (El valor es una cadena larga sin comillas).**

```
arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:885a9e7f-b954-48dc-9a71-af49368326dd:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaleup
```

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? La descripción de la alarma creada.**

```
[cloudshell-user@ip-10-2-19-33 ~]$ aws cloudwatch describe-alarms --alarm-names kimberly-salazar-highcpualarm
{
 "MetricAlarms": [
 {
 "AlarmName": "kimberly-salazar-highcpualarm",
 "AlarmArn": "arn:aws:cloudwatch:us-east-1:091101640480:alarm:kimberly-salazar-highcpualarm",
 "AlarmConfigurationUpdatedTimestamp": "2023-06-15T20:31:37.926000+00:00",
 "ActionsEnabled": true,
 "OKActions": [],
 "AlarmActions": [
 "arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:885a9e7f-b954-48dc-9a71-af49368326dd:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaleup"
],
 "InsufficientDataActions": [],
 "StateValue": "OK",
 "StateReason": "Threshold Crossed: 1 datapoint [0.483870967741934 (15/06/23 20:30:00)] was not greater than the threshold (70.0).",
 "StateReasonData": "{\"version\":\"1.0\",\"queryDate\":\"2023-06-15T20:32:23.790+0000\",\"startDate\":\"2023-06-15T20:30:00.000+0000\",\"statistic\":\"Average\",\"period\":120,\"recentDatapoints\":[0.483870967741934],\"threshold\":70.0,\"evaluatedDatapoints\":[{\"timestamp\":\"2023-06-15T20:30:00.000+0000\",\"sampleCount\":1.0,\"value\":0.483870967741934}]}",
 "StateUpdatedTimestamp": "2023-06-15T20:32:23.794000+00:00",
 "MetricName": "CPUUtilization",
 "Namespace": "AWS/EC2",
 "Statistic": "Average",
 "Dimensions": [
 ...
]
 }
]
}
```

**5. 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
```

```
ubuntu@ip-172-31-91-95:~$ sudo su
root@ip-172-31-91-95:/home/ubuntu# vi /etc/apt/sources.list
root@ip-172-31-91-95:/home/ubuntu# apt-get clean
root@ip-172-31-91-95:/home/ubuntu# apt-get update
```

```
root@ip-172-31-91-95:/home/ubuntu# apt-get install stress
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
 stress
0 upgraded, 1 newly installed, 0 to remove and 188 not upgraded.
Need to get 16.8 kB of archives.
After this operation, 73.7 kB of additional disk space will be used.
Get:1 http://old-releases.ubuntu.com/ubuntu/ precise/universe stress amd64 1.0.1-1build1 [16.8 kB]
Fetched 16.8 kB in 0s (127 kB/s)
Selecting previously unselected package stress.
(Reading database ... 47505 files and directories currently installed.)
Unpacking stress (from .../stress_1.0.1-1build1_amd64.deb) ...
Processing triggers for initramfs-tools ...
update-initramfs: Generating /boot/initrd.img-3.2.0-56-virtual
Processing triggers for install-info ...
Processing triggers for man-db ...
Setting up stress (1.0.1-1build1) ...
```

```
root@ip-172-31-91-95:/home/ubuntu# stress --cpu 1
stress: info: [4097] dispatching hogs: 1 cpu, 0 io, 0 vm, 0 hdd
```

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
```

```
[cloudshell-user@ip-10-6-183-45 ~]$ aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name kimberly-salazar-asg
{
 "AutoScalingGroups": [
 {
 "AutoScalingGroupName": "kimberly-salazar-asg",
 "AutoScalingGroupARN": "arn:aws:autoscaling:us-east-1:091101640488:autoScalingGroup:82375315-c130-4a69-a4f7-2fcc53371c73:autoScalingGroupName/kimberly-salaza
r-asg",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "MinSize": 1,
 "MaxSize": 3,
 "DesiredCapacity": 3,
 "DefaultCooldown": 300,
 "AvailabilityZones": [
 "us-east-1c"
],
 "LoadBalancerNames": [
 "kimberly-salazar-elb"
],
 "TargetGroupARNs": [],
 "HealthCheckType": "EC2",
 "HealthCheckGracePeriod": 0,
 "Instances": [
 {
 "InstanceId": "i-04076567a0e00000"
 }
]
 }
]
}
```

```

"Instances": [
 {
 "InstanceId": "i-0407fd2c4e8f0b7a9",
 "InstanceType": "t1.micro",
 "AvailabilityZone": "us-east-1c",
 "LifecycleState": "InService",
 "HealthStatus": "Healthy",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "ProtectedFromScaleIn": false
 },
 {
 "InstanceId": "i-05cf2c6c8e8bc14c0",
 "InstanceType": "t1.micro",
 "AvailabilityZone": "us-east-1c",
 "LifecycleState": "InService",
 "HealthStatus": "Healthy",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "ProtectedFromScaleIn": false
 },
 {
 "InstanceId": "i-09c694dfc32b184be",
 "InstanceType": "t1.micro",
 "AvailabilityZone": "us-east-1c",
 "CreatedTime": "2023-06-15T20:13:23.849000+00:00",
 "SuspendedProcesses": [],
 "CreatedTime": "2023-06-15T20:13:23.849000+00:00",
 "SuspendedProcesses": [],
 "VPCZoneIdentifier": "",
 "EnabledMetrics": [],
 "Tags": [],
 "TerminationPolicies": [
 "Default"
],
 "NewInstancesProtectedFromScaleIn": false,
 "ServiceLinkedRoleARN": "arn:aws:iam::091101640480:role/aws-service-role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
 "TrafficSources": [
 {
 "Identifier": "kimberly-salazar-elb",
 "Type": "elb"
 }
]
 }
]
}

```

## Parte 2: reducir la escala

6. 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?** En el primer comando se visualiza la política creada de escalado que disminuirá al menos una instancia y el segundo comando se visualiza la alarma creada que se activará cuando la utilización del CPU es superior al umbral 30%.

```
[cloudshell-user@ip-10-6-183-45 ~]$ aws autoscaling put-scaling-policy --auto-scaling-group-name kimberly-salazar-asg --policy-name kimberly-salazar-scaledown --scaling-adjustment -1 --adjustment-type ChangeInCapacity --cooldown 120
{
 "PolicyARN": "arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:00bbd2c9-67ab-43fa-9834-193418c4402f:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaledown",
 "Alarms": []
}
[cloudshell-user@ip-10-6-183-45 ~]$ █
```

```
[cloudshell-user@ip-10-6-183-45 ~]$ aws cloudwatch put-metric-alarm --alarm-name kimberly-salazar-lowcpualarm --metric-name CPUUtilization --namespace AWS/EC2 --statistic Average --period 120 --threshold 30 --comparison-operator LessThanThreshold --dimensions "Name=AutoScalingGroupName,Value=kimberly-salazar-asg" --evaluation-periods 1 --alarm-actions arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:00bbd2c9-67ab-43fa-9834-193418c4402f:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaledown
[cloudshell-user@ip-10-6-183-45 ~]$ aws cloudwatch describe-alarms --alarm-names kimberly-salazar-lowcpualarm
{
 "MetricAlarms": [
 {
 "AlarmName": "kimberly-salazar-lowcpualarm",
 "AlarmArn": "arn:aws:cloudwatch:us-east-1:091101640480:alarm:kimberly-salazar-lowcpualarm",
 "AlarmConfigurationUpdatedTimestamp": "2023-06-16T05:25:24.316000+00:00",
 "ActionsEnabled": true,
 "OKActions": [],
 "AlarmActions": [
 "arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:00bbd2c9-67ab-43fa-9834-193418c4402f:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaledown"
],
 "InsufficientDataActions": [],
 "StateValue": "ALARM",
 "StateReason": "Threshold Crossed: 1 datapoint [5.169468370843803 (16/06/23 05:24:00)] was less than the threshold (30.0).",
 "StateReasonData": "{\"version\":\"1.0\",\"queryDate\":\"2023-06-16T05:26:10.985+0000\",\"startDate\":\"2023-06-16T05:24:00.000+0000\",\"statistic\":\"Average\",\"period\":120,\"recentDatapoints\":[5.169468370843803],\"threshold\":30.0,\"evaluatedDatapoints\":[{\"timestamp\":\"2023-06-16T05:24:00.000+0000\",\"sampleCount\":4.0,\"value\":5.169468370843803}]}"
 }
]
}
```

```

 "StateUpdatedTimestamp": "2023-06-16T05:26:10.987000+00:00",
 "MetricName": "CPUUtilization",
 "Namespace": "AWS/EC2",
 "Statistic": "Average",
 "Dimensions": [
 {
 "Name": "AutoScalingGroupName",
 "Value": "kimberly-salazar-asg"
 }
],
 "Period": 120,
 "EvaluationPeriods": 1,
 "Threshold": 30.0,
 "ComparisonOperator": "LessThanThreshold",
 "StateTransitionedTimestamp": "2023-06-16T05:26:10.987000+00:00"
],
"CompositeAlarms": []
}
```

**¿Cuál es el valor de PolicyARN?**

**"PolicyARN":**

**"arn:aws:autoscaling:us-east-1:091101640480:scalingPolicy:00bbd2c9-67ab-43fa-9834-193418c4402f:autoScalingGroupName/kimberly-salazar-asg:policyName/kimberly-salazar-scaledown",**

- 7. 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? Se obtine la descripción de los grupos de AutoScaling.**

```
[cloudshell-user@ip-10-6-39-175 ~]$ aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name kimberly-salazar-asg
{
 "AutoScalingGroups": [
 {
 "AutoScalingGroupName": "kimberly-salazar-asg",
 "AutoScalingGroupARN": "arn:aws:autoscaling:us-east-1:091101640480:autoScalingGroup:82375315-c130-4a69-a4f7-2fcc53371c73:autoScalingGroupName/kimberly-salazar-asg",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "MinSize": 1,
 "MaxSize": 3,
 "DesiredCapacity": 1,
 "DefaultCooldown": 300,
 "AvailabilityZones": [
 "us-east-1c"
],
 "LoadBalancerNames": [
 "kimberly-salazar-elb"
],
 "TargetGroupARNs": [],
 "HealthCheckType": "EC2",
 "HealthCheckGracePeriod": 0,
 "Instances": [
 {
 "InstanceId": "i-0407fd2c4e8f0b7a9",
 "InstanceType": "t1.micro",
 "AvailabilityZone": "us-east-1c",
 "LifecycleState": "InService",
 "HealthStatus": "Healthy",
 "LaunchConfigurationName": "kimberly-salazar-lc",
 "ProtectedFromScaleIn": false
 },
 {"CreatedTime": "2023-06-15T20:13:23.849000+00:00",
 "SuspendedProcesses": [],
 "VPCZoneIdentifier": "",
 "EnabledMetrics": [],
 "Tags": [],
 "TerminationPolicies": [
 "Default"
],
 "NewInstancesProtectedFromScaleIn": false,
 "ServiceLinkedRoleARN": "arn:aws:iam::091101640480:role/aws-service-role/autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
 "TrafficSources": [
 {
 "Identifier": "kimberly-salazar-elb",
 "Type": "elb"
 }
]
]
 }
]
}
```

### Parte 3: Limpieza

- 8. 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? Se eliminan las alarmas y el grupo de AutoScaling.**

```
[cloudshell-user@ip-10-6-39-175 ~]$ aws autoscaling delete-auto-scaling-group --auto-scaling-group-name kimberly-salazar-asg --force-delete

[cloudshell-user@ip-10-6-39-175 ~]$ aws autoscaling delete-auto-scaling-group --auto-scaling-group-name kimberly-salazar-asg --force-delete
An error occurred (ValidationError) when calling the DeleteAutoScalingGroup operation: AutoScalingGroup name not found - AutoScalingGroup 'kimberly-salazar-asg' not found

[cloudshell-user@ip-10-6-39-175 ~]$ aws cloudwatch delete-alarms --alarm-name kimberly-salazar-lowcpualarm
[cloudshell-user@ip-10-6-39-175 ~]$ aws cloudwatch describe-alarms --alarm-names kimberly-salazar-lowcpualarm
{
 "MetricAlarms": [],
 "CompositeAlarms": []
}[cloudshell-user@ip-10-6-39-175 ~]$ █
```

```
aws cloudwatch delete-alarms --alarm-name
tu_nombre_de_usuario-highcpualarm
```

**¿Cuál es la salida? Se elimina la alarma.**

```
[cloudshell-user@ip-10-6-39-175 ~]$ aws cloudwatch delete-alarms --alarm-name kimberly-salazar-highcpualarm
[cloudshell-user@ip-10-6-39-175 ~]$ aws cloudwatch describe-alarms
{
 "MetricAlarms": [],
 "CompositeAlarms": []
}[cloudshell-user@ip-10-6-39-175 ~]$ █
```

Elimina tu configuración de lanzamiento de la siguiente manera.

```
aws autoscaling delete-launch-configuration
--launch-configuration-name tu_nombre_de_usuario-lc
```

```
[cloudshell-user@ip-10-6-39-175 ~]$ aws autoscaling
delete-launch-configuration --launch-configuration-name
kimberly-salazar-lc
```

**¿Cuál es la salida? Se elimina la configuración de lanzamiento**

```
[cloudshell-user@ip-10-6-39-175 ~]$ aws autoscaling delete-launch-configuration --launch-configuration-name kimberly-salazar-lc
[cloudshell-user@ip-10-6-39-175 ~]$ aws autoscaling describe-launch-configurations --launch-configuration-names kimberly-salazar-lc
{
 "LaunchConfigurations": []
}[cloudshell-user@ip-10-6-39-175 ~]$ █
```

Finalmente, elimina tu ELB. ¿Qué comando ejecutaste?. **aws elb delete-load-balancer --load-balancer-name kimberly-salazar-elb**

```
[cloudshell-user@ip-10-6-39-175 ~]$ aws elb delete-load-balancer --load-balancer-name kimberly-salazar-elb
[cloudshell-user@ip-10-6-39-175 ~]$ aws elb describe-load-balancers --load-balancer-names kimberly-salazar-elb
An error occurred (LoadBalancerNotFound) when calling the DescribeLoadBalancers operation: There is no ACTIVE Load Balancer named 'kimberly-salazar-elb'
[cloudshell-user@ip-10-6-39-175 ~]$ █
```