

Caso Práctico: Automatización y Elasticidad en AWS

El objetivo de este caso práctico es proporcionar una plantilla de CloudFormation que permita desplegar una arquitectura en alta disponibilidad. Para esto, debes crear lo siguiente:

- Una plantilla CloudFormation que incluya lo siguiente:
 - Una VPC con 2 subnets públicas en zonas de disponibilidad distintas (puedes elegir las zonas de disponibilidad que consideres).
 - Un autoscaling group que despliegue un mínimo de 2 y un máximo de 4 instancias EC2 con la AMI Amazon Linux 2 y de tamaño t2.micro o t3.micro (puedes elegir el tamaño). Las instancias deben tener instalado el servidor web Nginx utilizando un script User Data y también deben tener dirección IP pública asignada. Estas instancias deben tener asociado un Security Group que sólo permita el acceso a través del puerto 80 desde cualquier IP de origen.


```

124 #EC2
125
126 LaunchTemplate:
127   Type: AWS::EC2::LaunchTemplate
128   Properties:
129     LaunchTemplateName: WebServer
130     LaunchTemplateData:
131       NetworkInterfaces:
132         - DeviceIndex: 0
133           AssociatePublicIpAddress: true
134           Groups:
135             - !Ref InstanceSecurityGroup
136           DeleteOnTermination: true
137       ImageId: ami-89d53274b6c5d4aa
138       InstanceType: !Ref InstanceType
139       UserData:
140         Fn::Base64:
141           !Sub |
142             #!/bin/bash
143             sudo yum update -y
144             sudo amazon-linux-extras install nginx1 -y
145             sudo systemctl enable nginx
146             sudo systemctl start nginx
147
148 # AutoScaling
149
150 AutoScalingGroup:
151   Type: AWS::AutoScaling::AutoScalingGroup
152   Properties:
153     AutoScalingGroupName: My-AG
154     MinSize: 2
155     MaxSize: 4
156     DesiredCapacity: 2
157     LaunchTemplate:
158       LaunchTemplateId: !Ref LaunchTemplate
159       Versions: !Ref LaunchTemplateVersionNumber
160     VPCZoneIdentifier:
161       - !Ref PublicSubnet1
162       - !Ref PublicSubnet2

```

He lanzado el stack en el Sandbox y se han desplegado todos los recursos sin problemas. Las subnets en 2 Availability Zones diferentes con sus correspondientes Route Tables:

The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with various AWS services icons. The main content area is titled 'Subnets (1/2) info'. Below this, there's a table listing subnets. Two subnets are visible: 'Public-Subnet-1' and 'Public-Subnet-2'. Both are in the 'Available' state, associated with VPC 'vpc-02462c812b6732136'. They are located in different Availability Zones: 'us-east-1a' and 'us-east-1b'. Below the subnets table, the 'Route table' section is expanded for 'rtb-0d0481153503effdb'. It shows two routes: one for destination '10.0.0/16' targeting 'local', and another for '0.0.0/0' targeting 'igw-043af4e1bbe1b178'.

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR	Available IPv4 addresses	Availability Zone	Availability Zone ID	Network border group
Public-Subnet-1	subnet-0eb1d8caf434d1689	Available	vpc-02462c812b6732136	10.0.0.0/24	-	250	us-east-1a	us-east-1a	us-east-1
Public-Subnet-2	subnet-0b658653c2477e7e1	Available	vpc-02462c812b6732136	10.0.1.0/24	-	250	us-east-1b	us-east-1b	us-east-1

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	igw-043af4e1bbe1b178

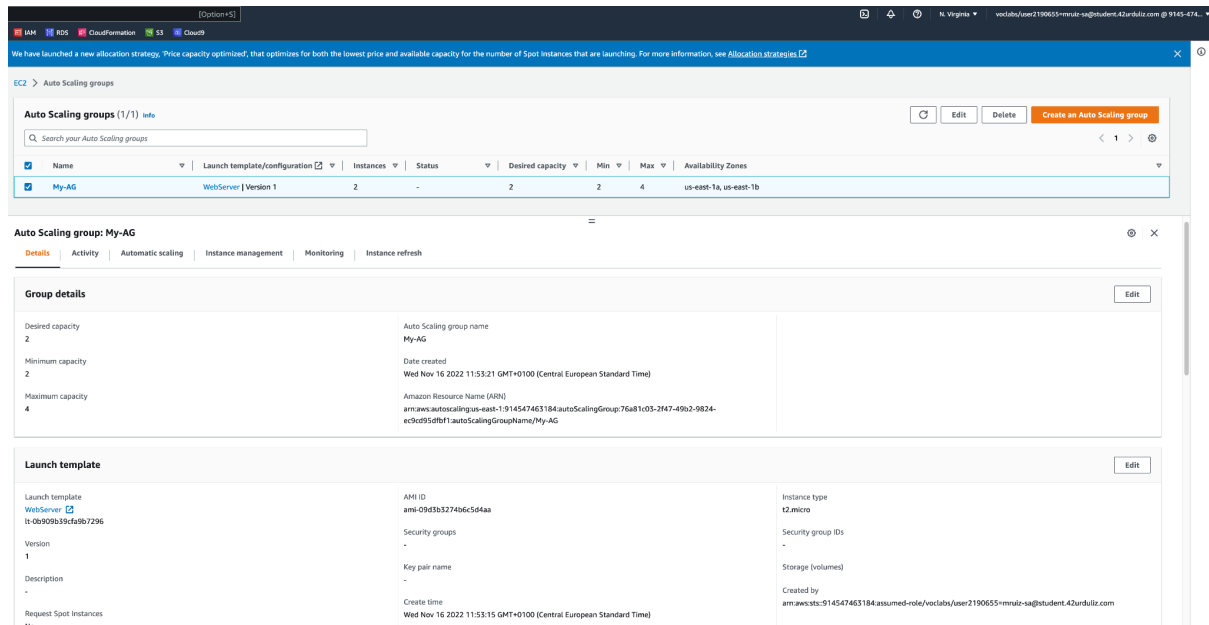
Las instancias EC2 con sus Security Groups y sus reglas:

The screenshot shows the AWS Management Console interface for EC2 instances. The main content area is titled 'Instances (1/2) info'. Below this, there's a table listing instances. Two instances are visible: 'i-0abf163c7e6e964e5' and 'i-0576842f9b054706'. Both are in the 'Running' state, using the 't2.micro' instance type. They are located in the 'us-east-1a' and 'us-east-1b' Availability Zones. Below the instances table, the 'Security' section is expanded for instance 'i-0abf163c7e6e964e5'. It shows the instance's IAM Role, Owner ID, and Launch time. The 'Security groups' section is also expanded, showing the 'sg-0569f7a38e9992141' security group. The 'Inbound rules' section is also expanded, showing a rule for port 80, protocol TCP, from source '0.0.0.0/0' to the security group.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security group name
-	i-0abf163c7e6e964e5	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec3-54-221-176-56.co...	54.221.176.56	-	-	disabled	application-InstanceSec...
-	i-0576842f9b054706	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-54-80-4-82.comput...	54.80.4.82	-	-	disabled	application-InstanceSec...

Name	Security group rule ID	Port range	Protocol	Source	Security groups	Description
-	sg-03ca3a268613748b	80	TCP	0.0.0.0/0	application-InstanceSecurityGroup-18...	-

Y finalmente el AutoScaling Group con todas las características que se nos pedía:



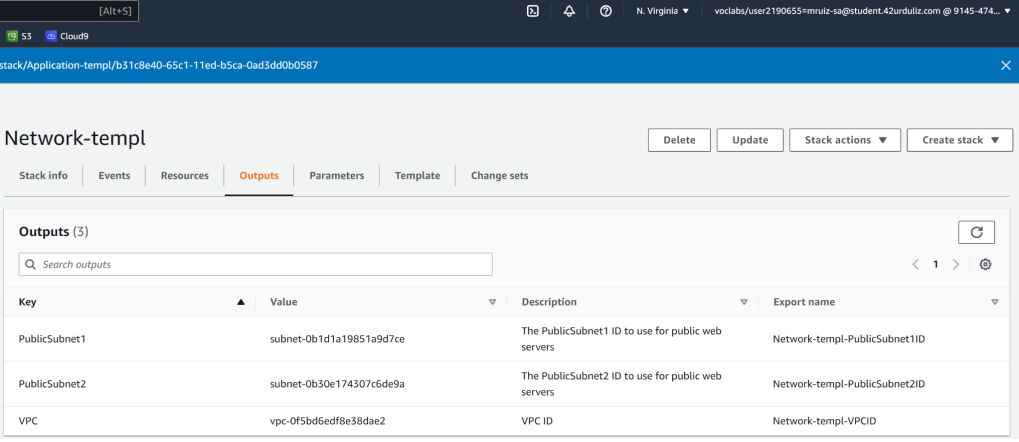
Tarea 2 (25 puntos): Dividir la plantilla anterior en dos plantillas. Una de ellas solo incluirá los recursos de red: VPC y Subnets. La otra plantilla incluirá los recursos de computación: Autoscaling Group y Security Group.

En esta tarea básicamente lo que he hecho ha sido separar en 2 templates diferentes la parte del Network y la parte del AutoScaling que lanzará las EC2. Como a la hora de lanzar las instancias tenía que hacer referencia a las Subnets donde se iban a crear y además el Security Group ha de ser creado en una VPC, tenía que hacer una llamada a ciertos recursos del template Network.

Para ello he añadido al final de la plantilla Network los Output de las 2 Subnets públicas y de la VPC:

```
97 #####
98 ##OUTPUTS##
99 #####
100
101 Outputs:
102
103 PublicSubnet1:
104   Description: The PublicSubnet1 ID to use for public web servers
105   Value: !Ref PublicSubnet1
106   Export:
107     Name: !Sub '${AWS::StackName}-PublicSubnet1ID'
108
109 PublicSubnet2:
110   Description: The PublicSubnet2 ID to use for public web servers
111   Value: !Ref PublicSubnet2
112   Export:
113     Name: !Sub '${AWS::StackName}-PublicSubnet2ID'
114
115 VPC:
116   Description: VPC ID
117   Value: !Ref VPC
118   Export:
119     Name: !Sub '${AWS::StackName}-VPCID'
```

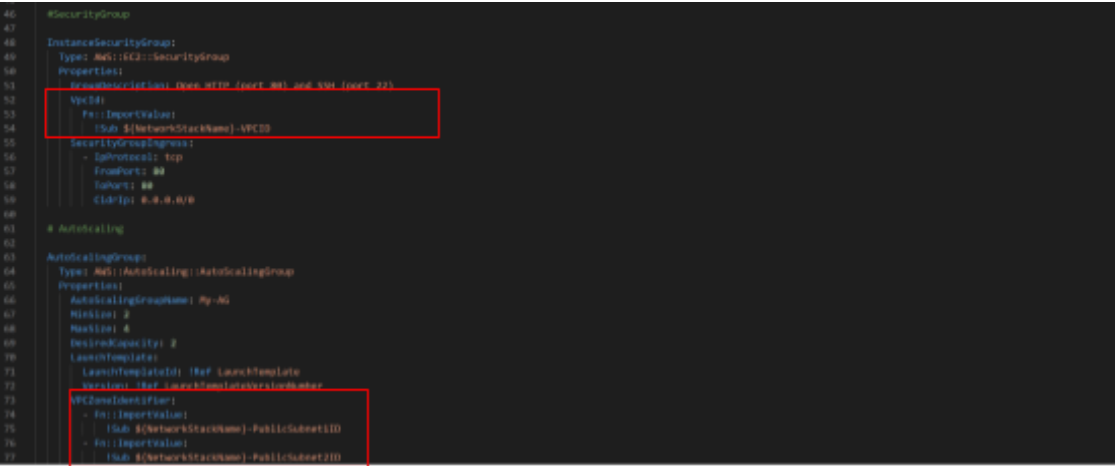
He lanzado este primer template y se han desplegado solo los recursos de la parte Network. Además en la pestaña Outputs del stack se han exportado los parámetros que quería:



The screenshot shows the AWS CloudFormation console interface. At the top, there's a navigation bar with 'Cloud9' and a search bar. Below that, the stack name 'Network-templ' is displayed with buttons for 'Delete', 'Update', 'Stack actions', and 'Create stack'. The 'Outputs' tab is selected, showing a table of three outputs.

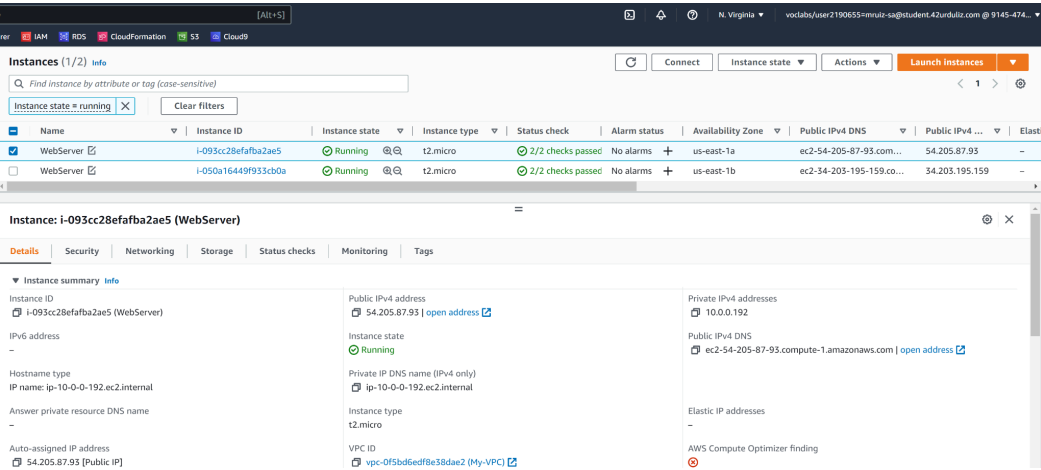
Key	Value	Description	Export name
PublicSubnet1	subnet-0b1d1a19851a9d7ce	The PublicSubnet1 ID to use for public web servers	Network-templ-PublicSubnet1ID
PublicSubnet2	subnet-0b30e174307c6de9a	The PublicSubnet2 ID to use for public web servers	Network-templ-PublicSubnet2ID
VPC	vpc-0f5bd6edf8e38dae2	VPC ID	Network-templ-VPCID

Después he creado otro template con la parte de las aplicaciones, pero ahora no podía hacer referencia a los parámetros del networking ya que no están en el mismo template. Así que he importado los parámetros que necesitaba, que eran las Subnets y la VPC:



The screenshot shows a snippet of an AWS CloudFormation template. It defines two resources: a SecurityGroup and an AutoScalingGroup. The SecurityGroup resource is named 'InstanceSecurityGroup' and has properties for 'Type', 'VpcId', 'SecurityGroupIngress', 'Egress', 'IpPermissions', and 'IpPermissionsEgress'. The AutoScalingGroup resource is named 'AutoScalingGroup' and has properties for 'Type', 'AutoScalingGroupName', 'MinSize', 'MaxSize', 'DesiredCapacity', 'LaunchTemplate', 'VPCSubnetId', and 'VPCSubnetId'. The 'VPCSubnetId' property is highlighted with a red box, indicating it's a reference to a parameter from another stack.

Por último he lanzado el template y todo se ha desplegado sin ningún problema igual que cuando todo era un único template.



The screenshot shows the AWS CloudFormation console interface. At the top, there's a navigation bar with 'IAM', 'RDS', 'CloudFormation', 'S3', and 'Cloud9'. Below that, the stack name 'WebServer' is displayed with buttons for 'Connect', 'Instance state', 'Actions', and 'Launch instances'. The 'Instances' tab is selected, showing a table of two instances.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elast
WebServer	i-093cc28efab2a2e5	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-54-205-87-93.com...	54.205.87.93	-
WebServer	i-050a16449933cb0a	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-34-203-195-159.co...	34.203.195.159	-

Below the table, there's a section for the 'Instance: i-093cc28efab2a2e5 (WebServer)'. It shows details for the instance, including its IP address, hostname, and VPC ID.