Contents

Prerequisites

Labs1c1 have to be done and the context for Administrative user have to activated on Command Line Session.

Labs5c1 have to be done, because you learn how to: Deploy Network Infrastructure, Securize instances, deploy applications using Docker and create a functional ALB.

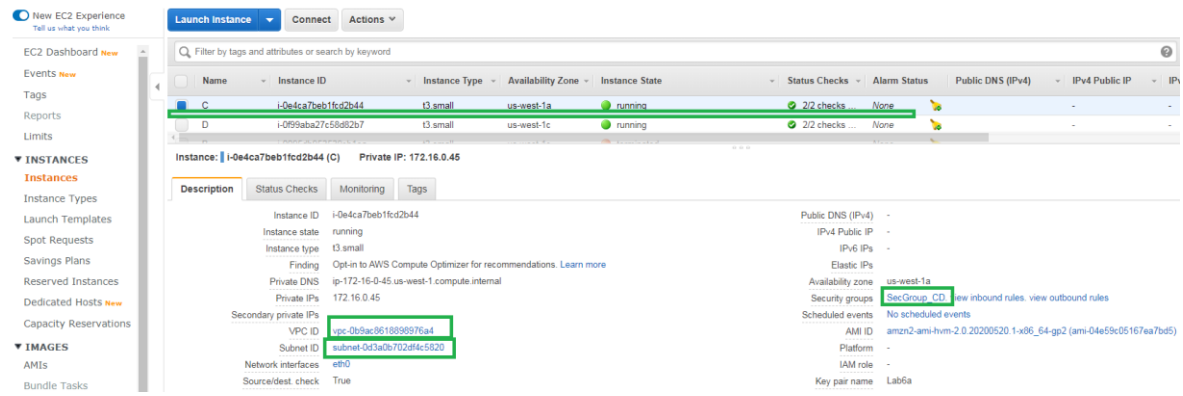
Lab 6A: Multi-tier Web App with ALB and NLB

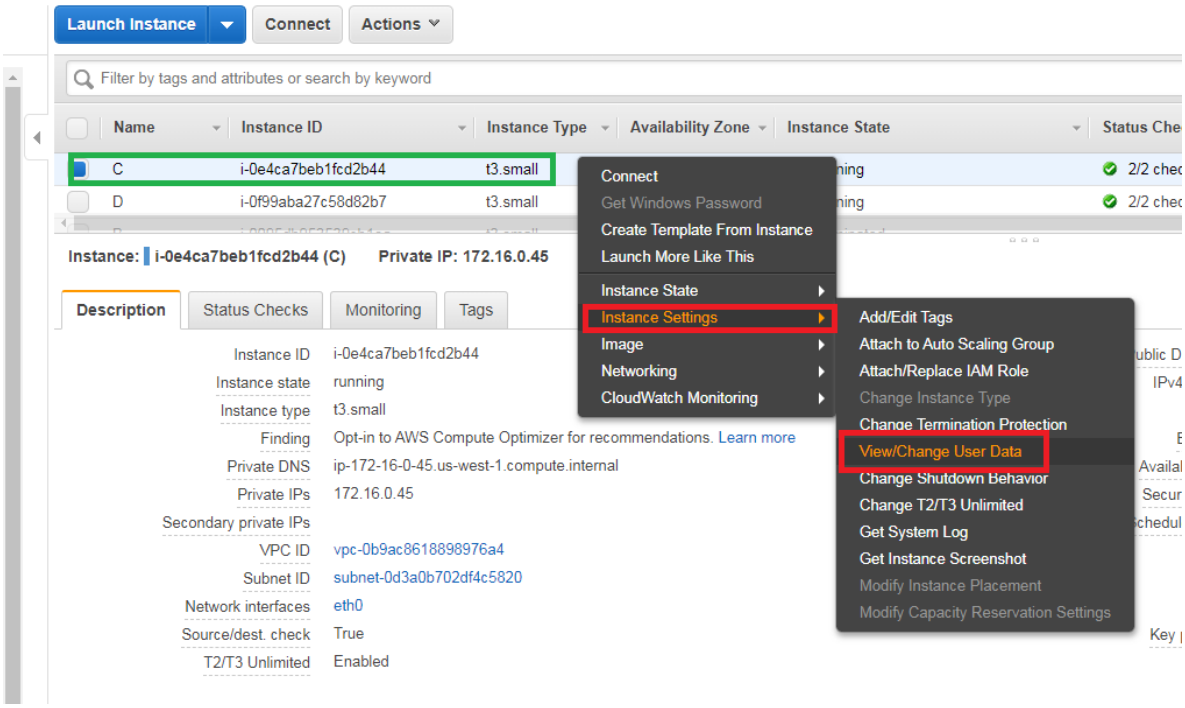
Lab 6A using Web Management Console

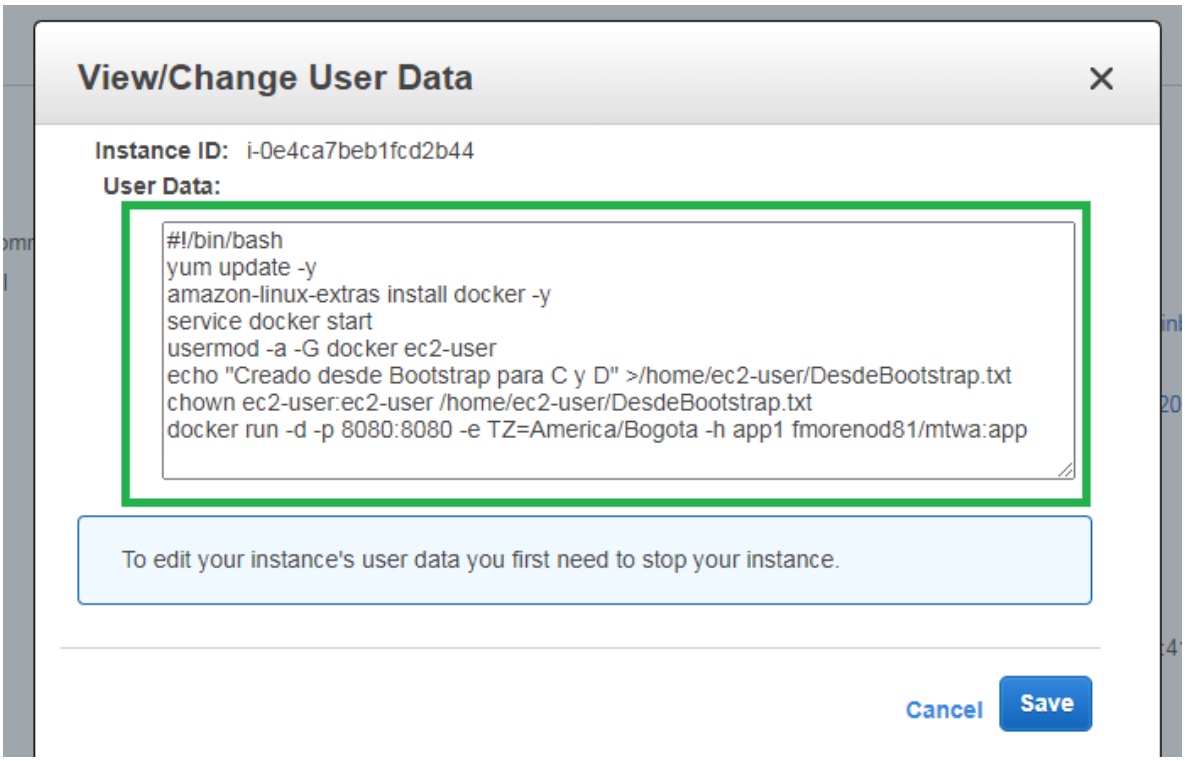
Create VPC, Subnet, IGW, NAT, Routing Table, Keys, Sec Groups (Labs4c1) VPC Peering (Labs4c2), Instances on App Layer (Labs5c1)

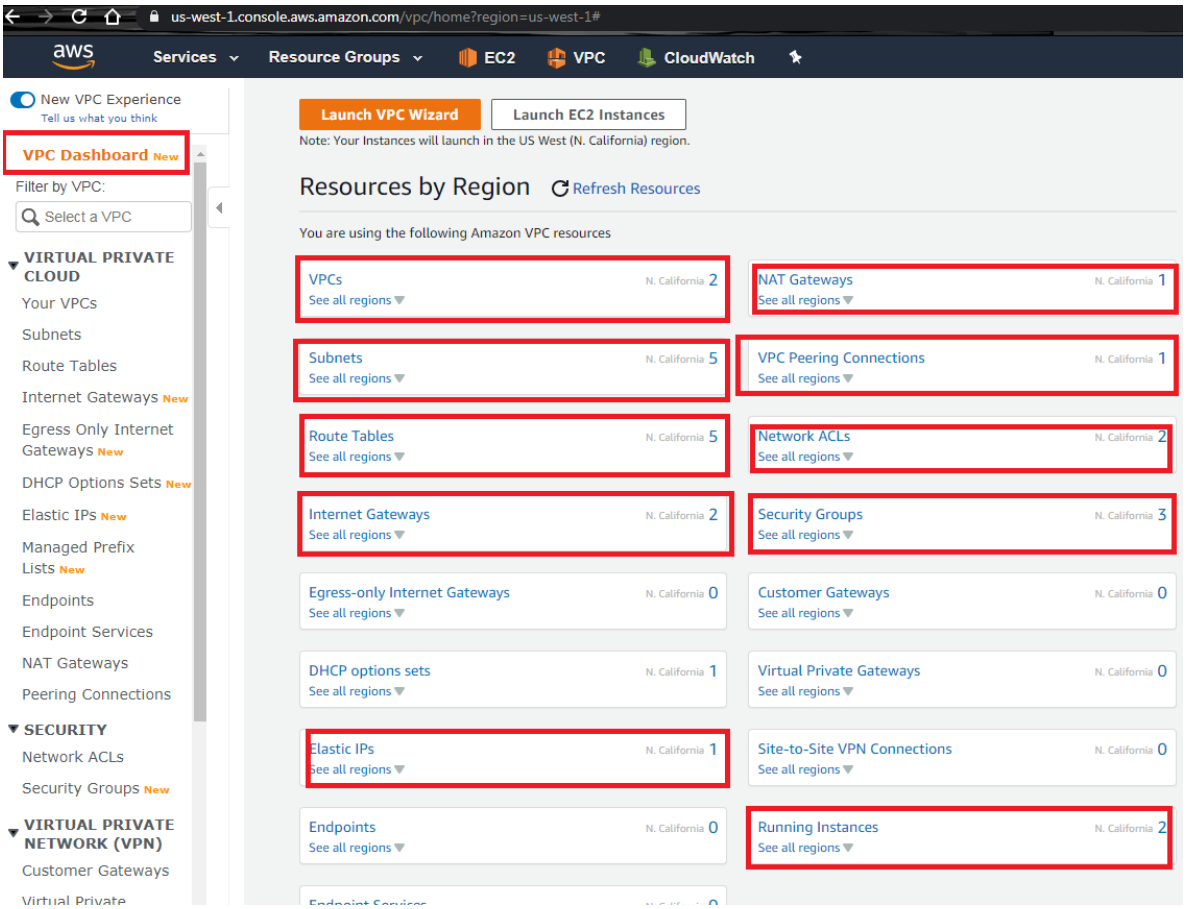
To remember the creation of the instance using a bootstrapping (user-data), and the correct network information and its security groups

---

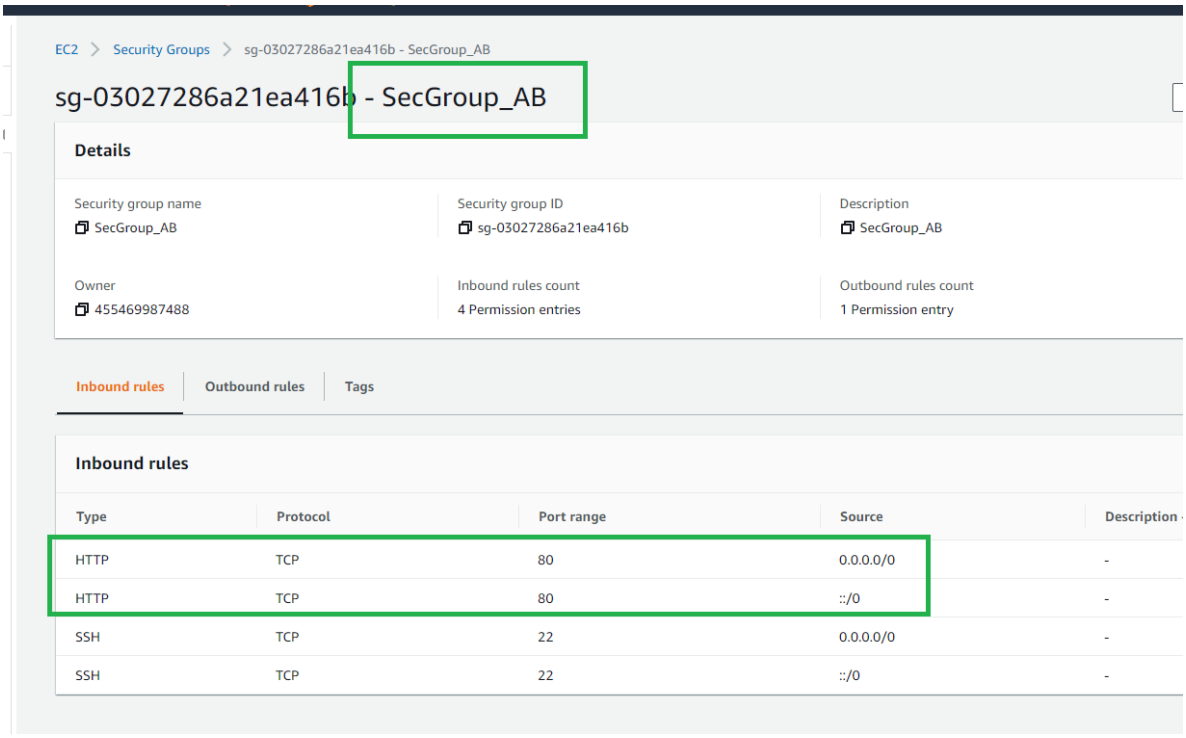


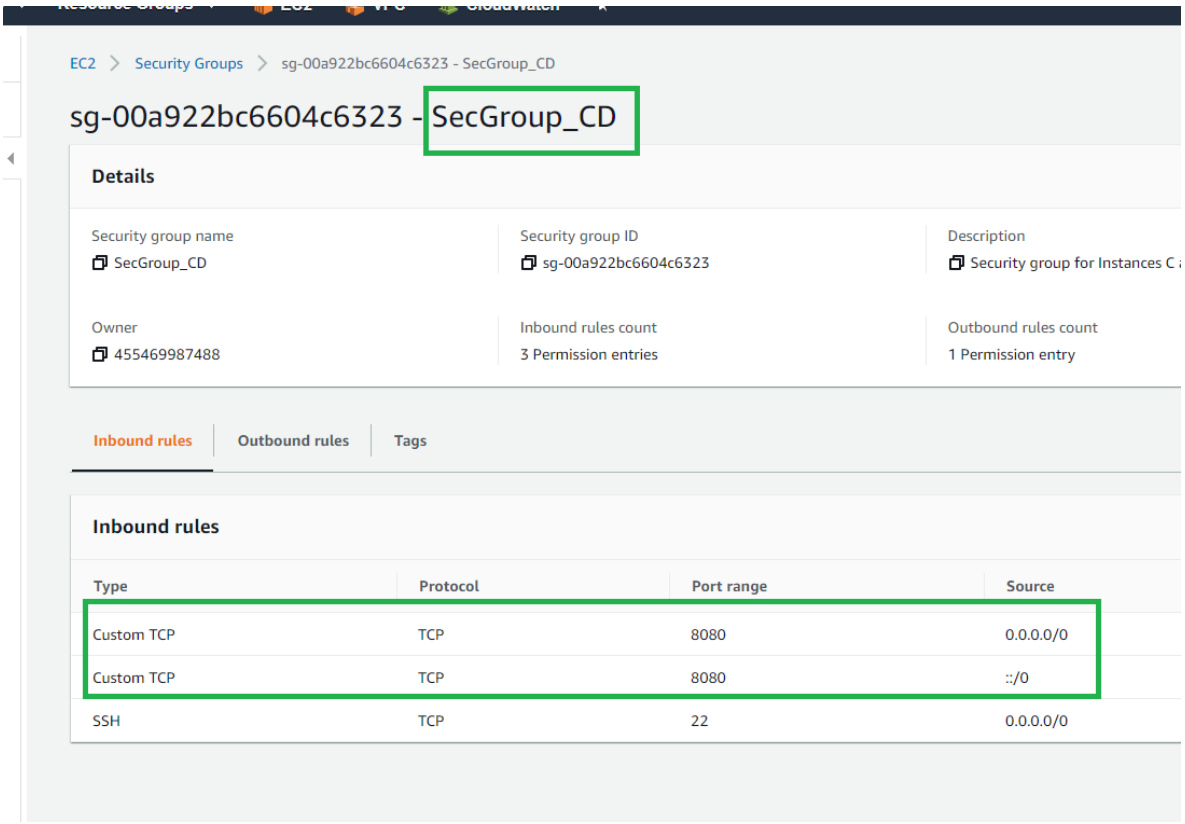

Launch Instance
Connect
Actions
Filter by tags and attributes or search by keyword
Name
Instance ID
Instance Type
Availability Zone
Instance State
Status Che
C
i-0e4ca7beb1fcd2b44
t3.small
D
i-0f99aba27c58d82b7
t3.small
Instance: i-0e4ca7beb1fcd2b44 (C)
Private IP: 172.16.0.45
Description
Status Checks
Monitoring
Tags
Instance ID
i-0e4ca7beb1fcd2b44
Instance state
running
Instance type
t3.small
Finding
Opt-in to AWS Compute Optimizer for recommendations. Learn more
Private DNS
ip-172-16-0-45.us-west-1.compute.internal
Private IPs
172.16.0.45
Secondary private IPs
VPC ID
vpc-0b9ac8618898976a4
Subnet ID
subnet-0d3a0b702df4c5820
Network interfaces
eth0
Source/dest. check
True
T2/T3 Unlimited
Enabled
Connect
Get Windows Password
Create Template From Instance
Launch More Like This
Instance State
Instance Settings
Image
Networking
CloudWatch Monitoring
Add/Edit Tags
Attach to Auto Scaling Group
Attach/Replace IAM Role
Change Instance Type
Change Termination Protection
View/Change User Data
Change Shutdown Behavior
Change T2/T3 Unlimited
Get System Log
Get Instance Screenshot
Modify Instance Placement
Modify Capacity Reservation Settings


View/Change User Data
Instance ID: i-0e4ca7beb1fcd2b44
User Data:
#!/bin/bash
yum update -y
amazon-linux-extras install docker -y
service docker start
usermod -a -G docker ec2-user
echo "Creado desde Bootstrap para C y D" >/home/ec2-user/DesdeBootstrap.txt
chown ec2-user:ec2-user /home/ec2-user/DesdeBootstrap.txt
docker run -d -p 8080:8080 -e TZ=America/Bogota -h app1 fmorenod81/mtwa:app
To edit your instance's user data you first need to stop your instance.
Cancel
Save



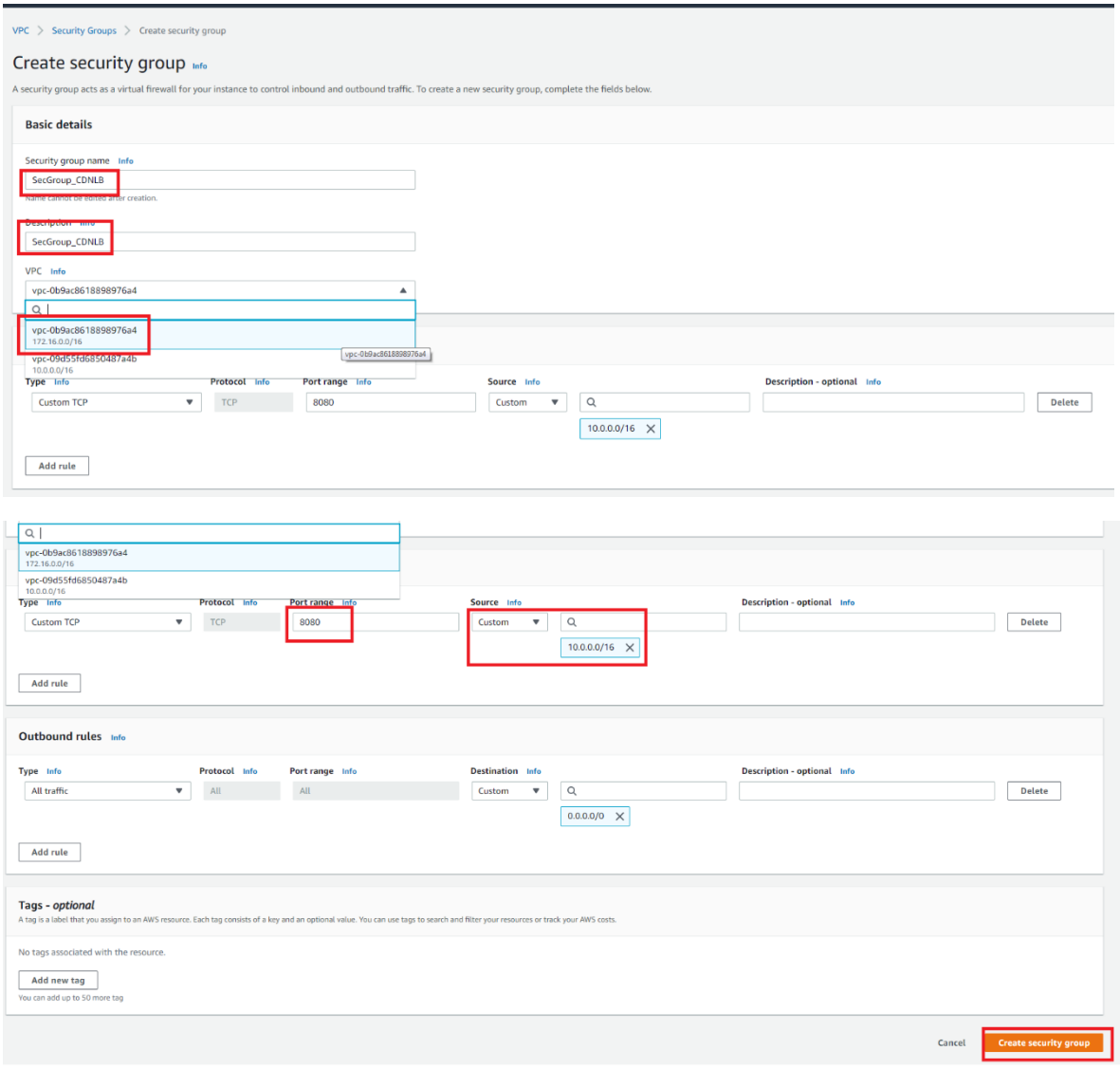
Create Security Group for Instances A, B and C, D



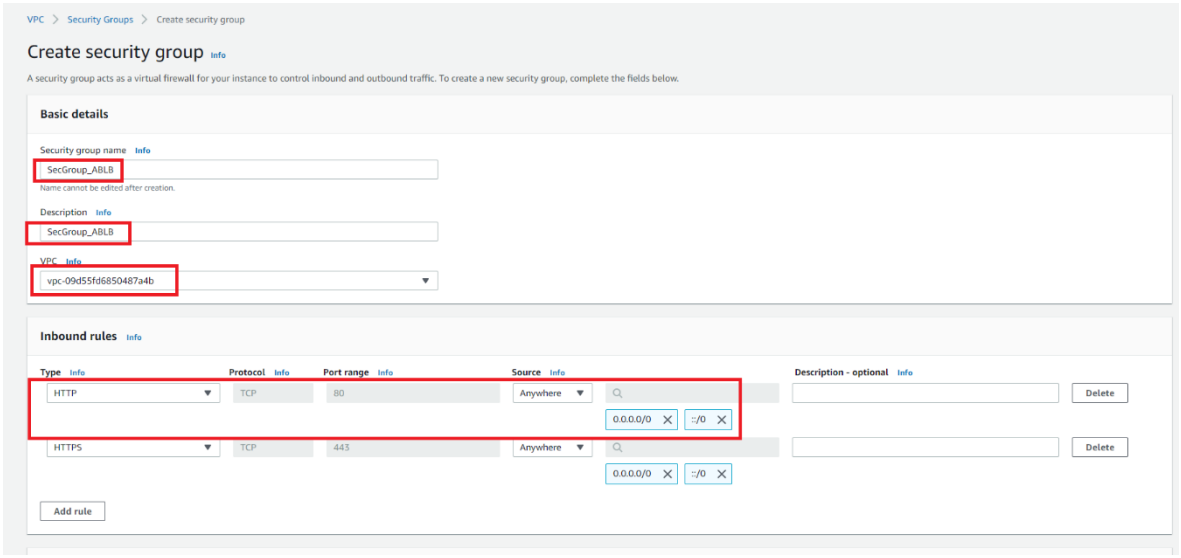


Create Security Groups for Balancers

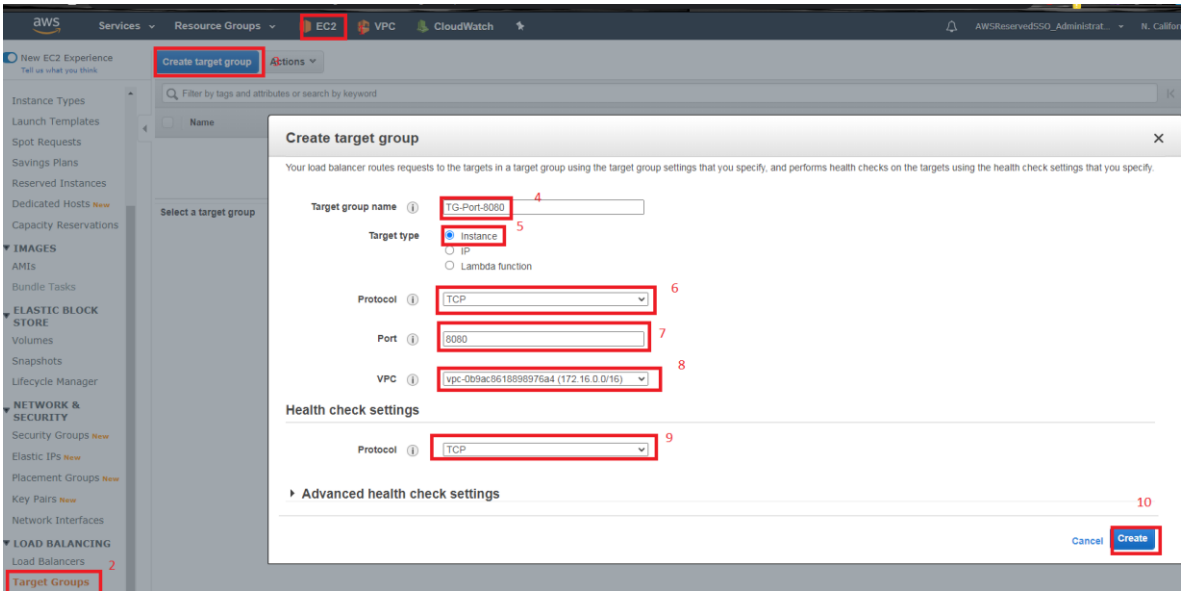
For Internal Balancer,

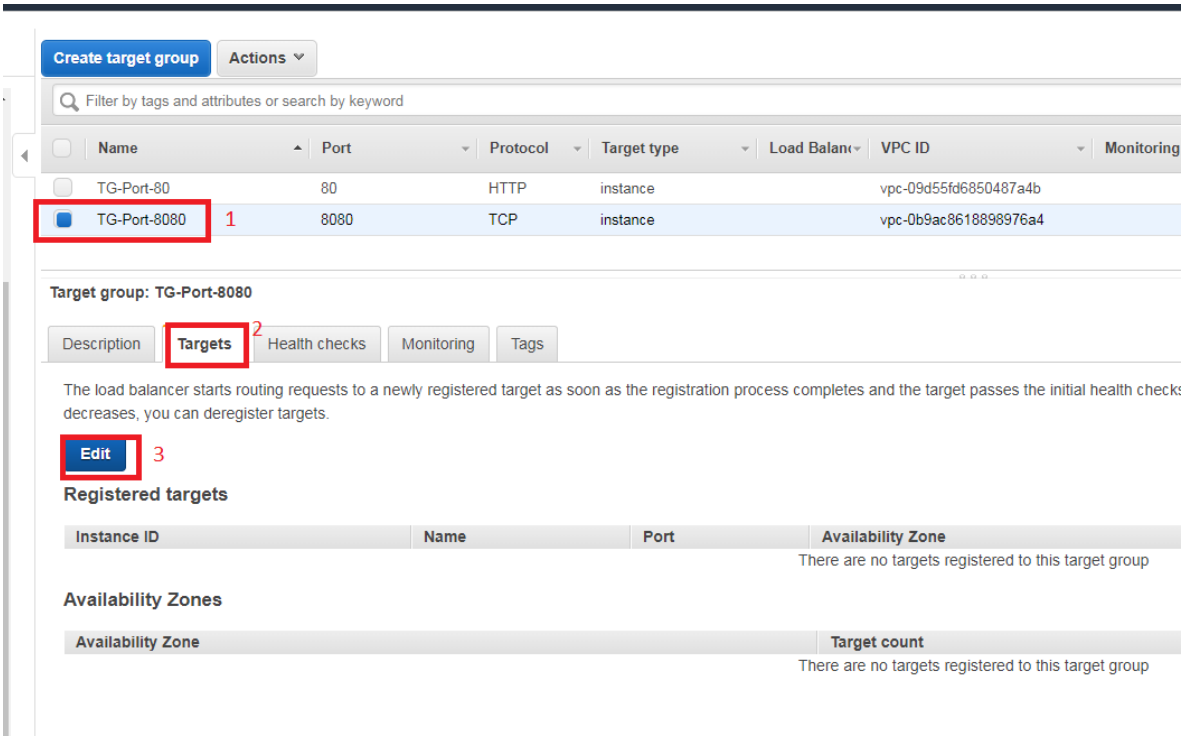


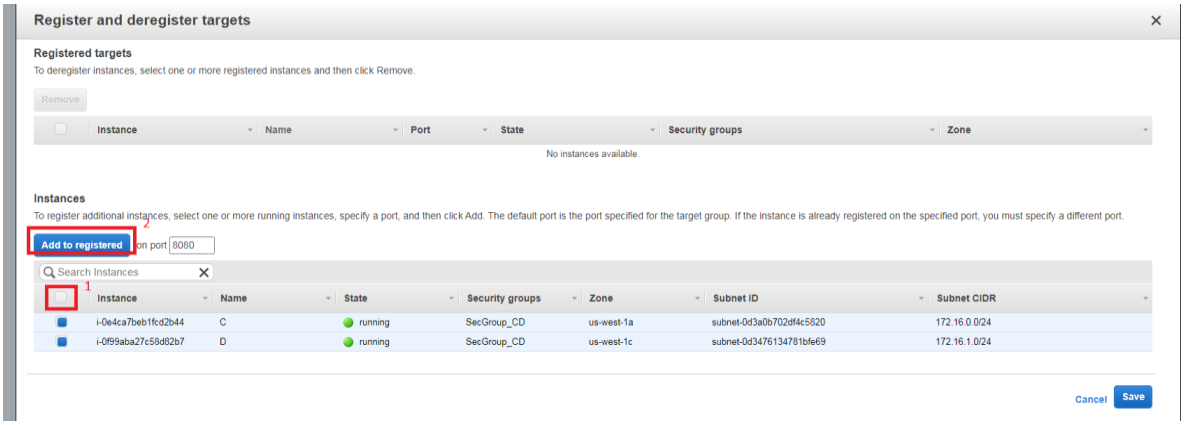
For External Balancer,

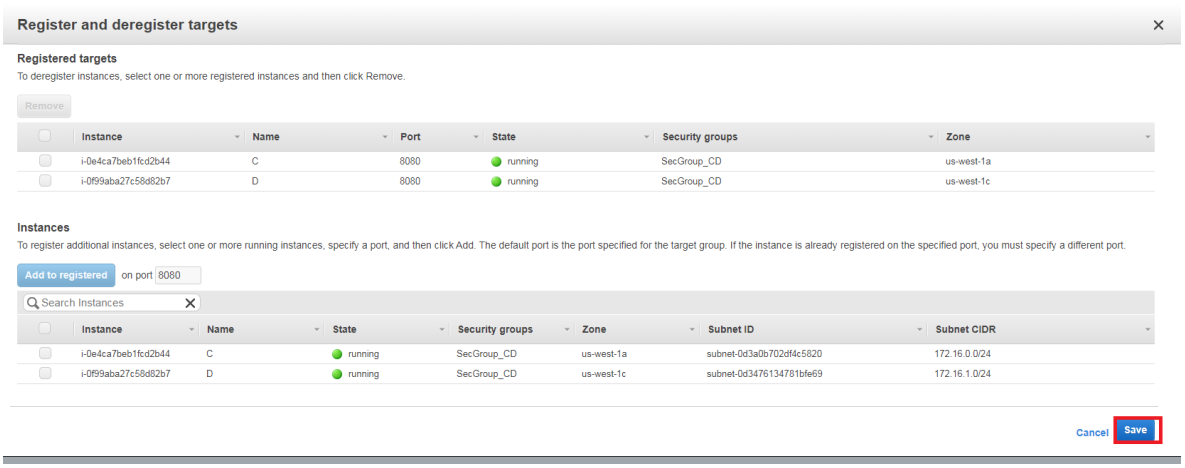


Create Target Groups for NLB

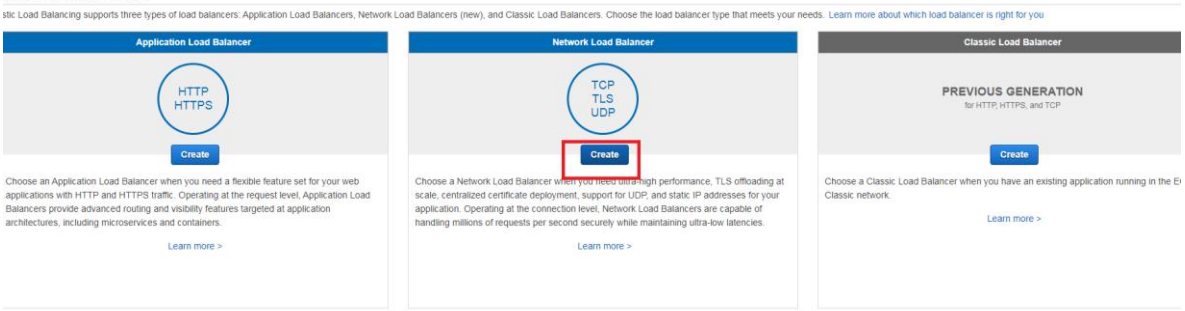


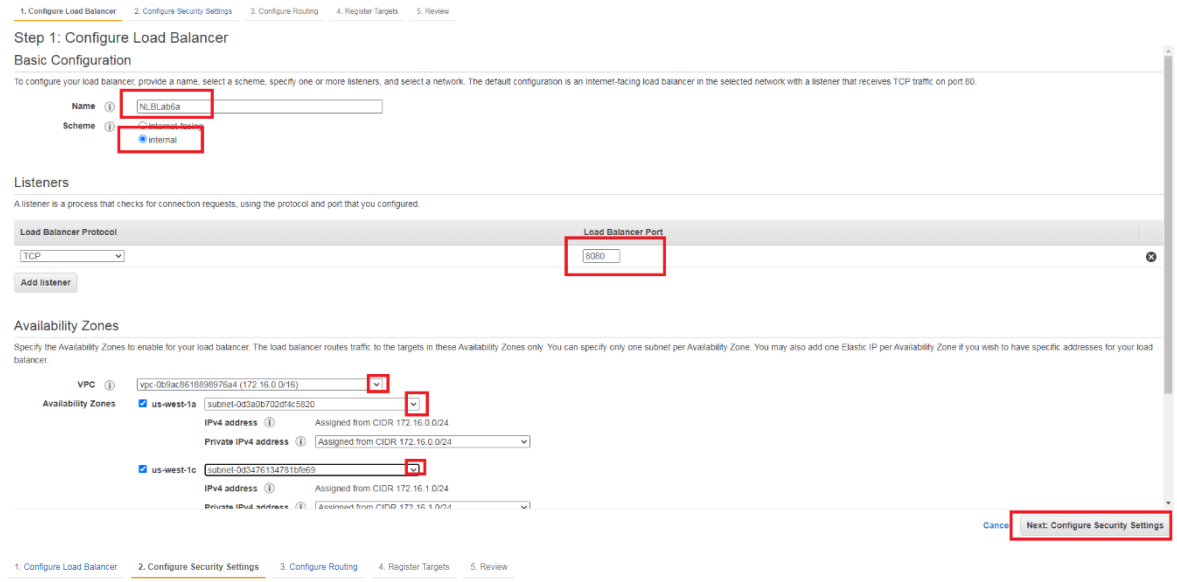

Create target group
Actions
Filter by tags and attributes or search by keyword
Name
Port
Protocol
Target type
Load Balan
VPC ID
Monitoring
TG-Port-80
80
HTTP
instance
vpc-09d55fd6850487a4b
TG-Port-8080
1
8080
TCP
instance
vpc-0b9ac8618898976a4
Target group: TG-Port-8080
Description
Targets
2
Health checks
Monitoring
Tags
The load balancer starts routing requests to a newly registered target as soon as the registration process completes and the target passes the initial health checks
decreases, you can deregister targets.
Edit
3
Registered targets
Instance ID
Name
Port
Availability Zone
There are no targets registered to this target group
Availability Zones
Availability Zone
Target count
There are no targets registered to this target group

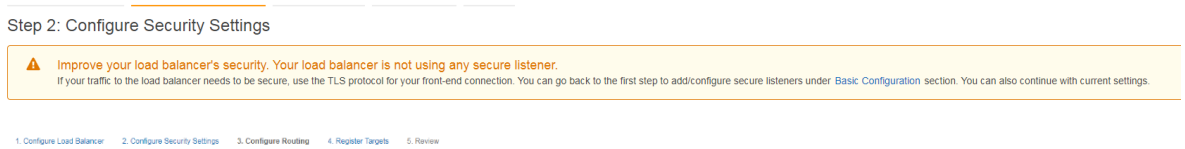

Register and deregister targets
Registered targets
To deregister instances, select one or more registered instances and then click Remove.
Remove
Instance
Name
Port
State
Security groups
Zone
No instances available.
Instances
To register additional instances, select one or more running instances, specify a port, and then click Add. The default port is the port specified for the target group. If the instance is already registered on the specified port, you must specify a different port.
Add to registered
2
on port 8080
Search Instances
1
Instance
Name
State
Security groups
Zone
Subnet ID
Subnet CIDR
i-0e4ca7beb1fcd2b44
C
running
SecGroup_CD
us-west-1a
subnet-0d3a0b702df4c5820
172.16.0.0/24
i-0f99aba27c58d82b7
D
running
SecGroup_CD
us-west-1c
subnet-0d3476134781bfe69
172.16.1.0/24
Cancel
Save

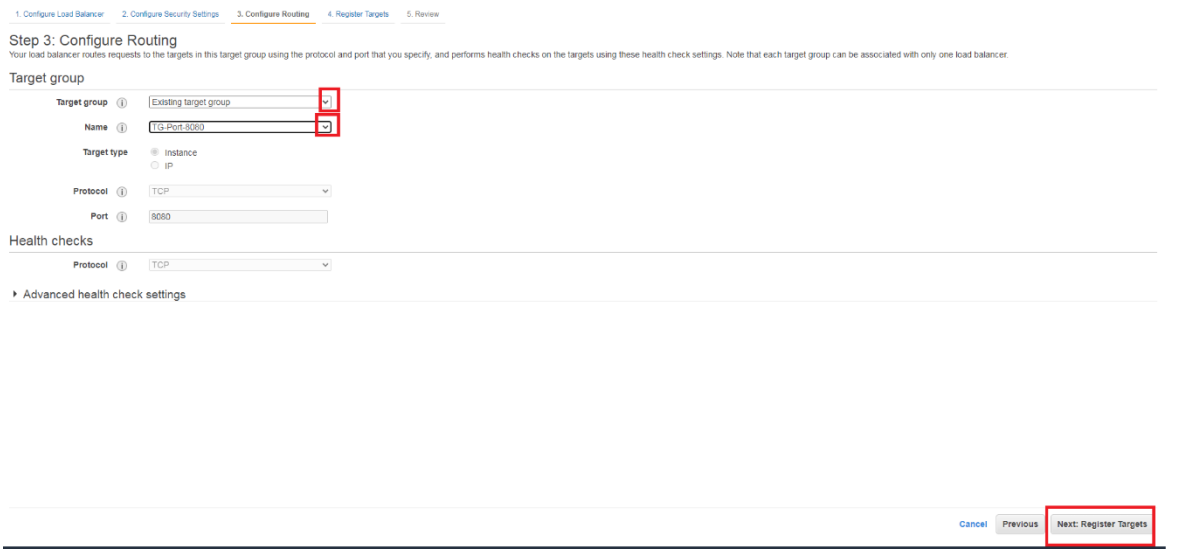

Register and deregister targets
Registered targets
To deregister instances, select one or more registered instances and then click Remove.
Remove
Instance
Name
Port
State
Security groups
Zone
i-0e4ca7beb1fcd2b44
C
8080
running
SecGroup_CD
us-west-1a
i-0f99aba27c58d82b7
D
8080
running
SecGroup_CD
us-west-1c
Instances
To register additional instances, select one or more running instances, specify a port, and then click Add. The default port is the port specified for the target group. If the instance is already registered on the specified port, you must specify a different port.
Add to registered
on port 8080
Search Instances
Instance
Name
State
Security groups
Zone
Subnet ID
Subnet CIDR
i-0e4ca7beb1fcd2b44
C
running
SecGroup_CD
us-west-1a
subnet-0d3a0b702df4c5820
172.16.0.0/24
i-0f99aba27c58d82b7
D
running
SecGroup_CD
us-west-1c
subnet-0d3476134781bfe69
172.16.1.0/24
Cancel
Save

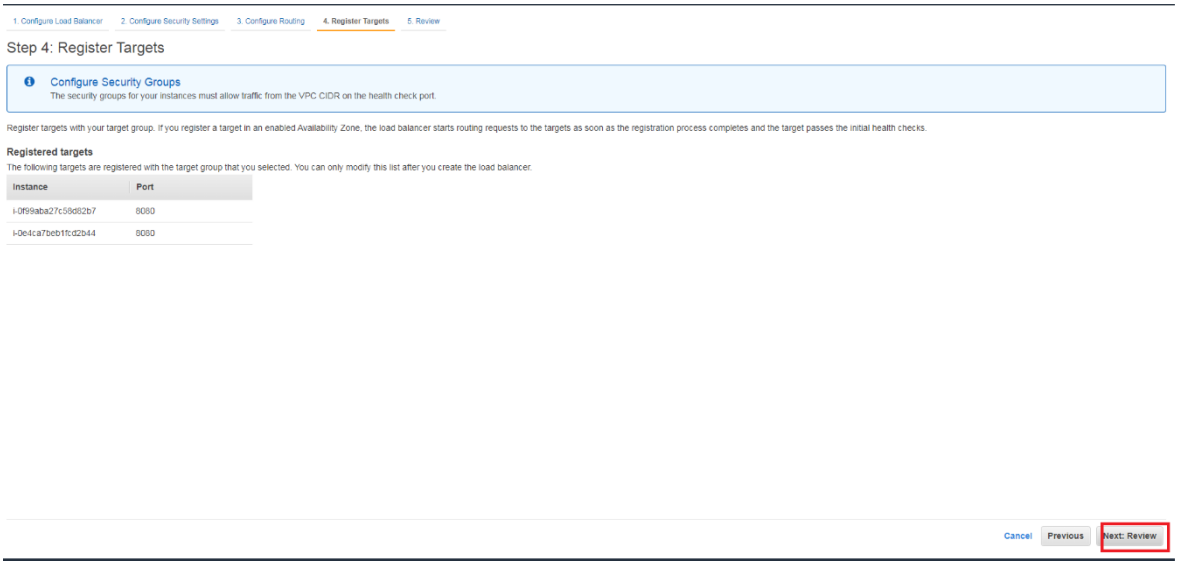
Create NLB and its listener

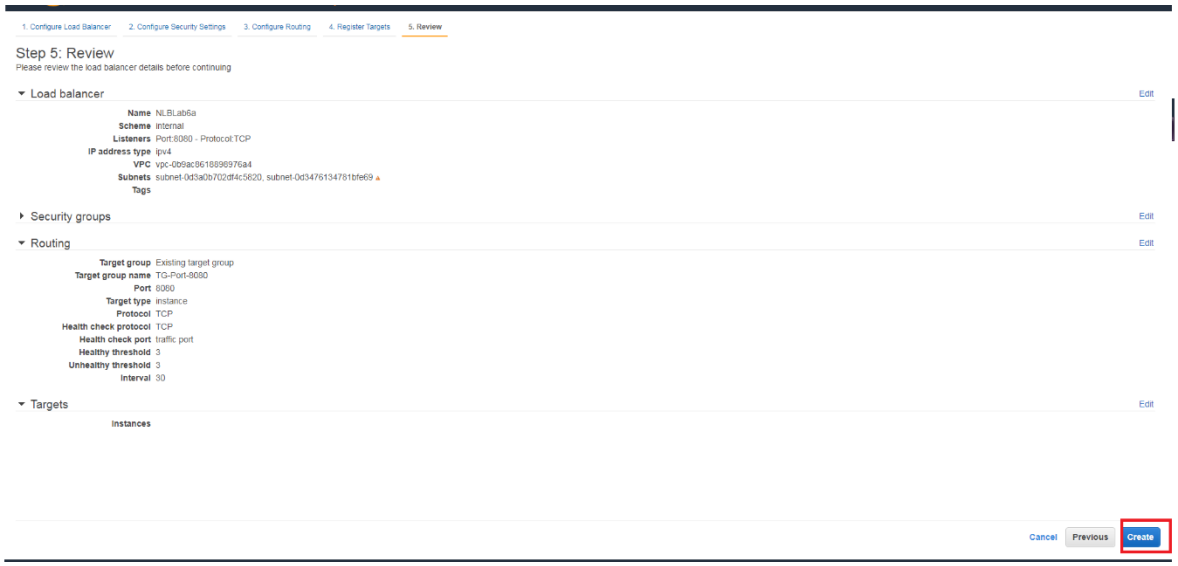


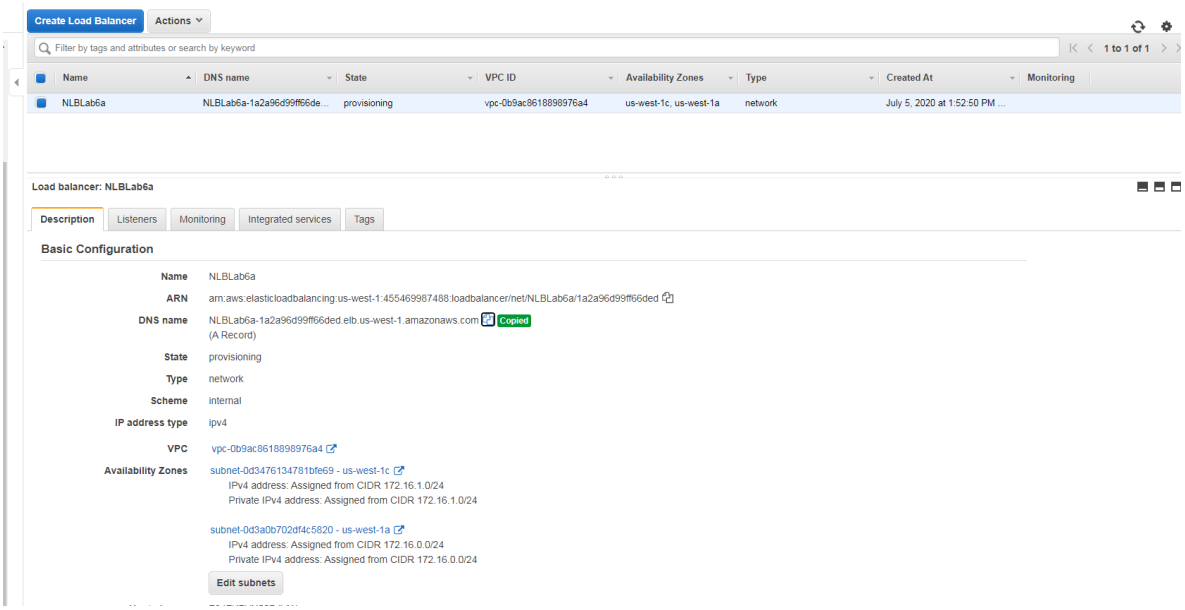







1. Configure Load Balancer
2. Configure Security Settings
3. Configure Routing
4. Register Targets
5. Review
Step 4: Register Targets
Configure Security Groups
The security groups for your instances must allow traffic from the VPC CIDR on the health check port.
Register targets with your target group. If you register a target in an enabled Availability Zone, the load balancer starts routing requests to the targets as soon as the registration process completes and the target passes the initial health checks.
Registered targets
The following targets are registered with the target group that you selected. You can only modify this list after you create the load balancer.
Instance
Port
i-0f99aba27c58d82b7
8080
i-0e4ca7beb1fcd2b44
8080
Cancel
Previous
Next: Review

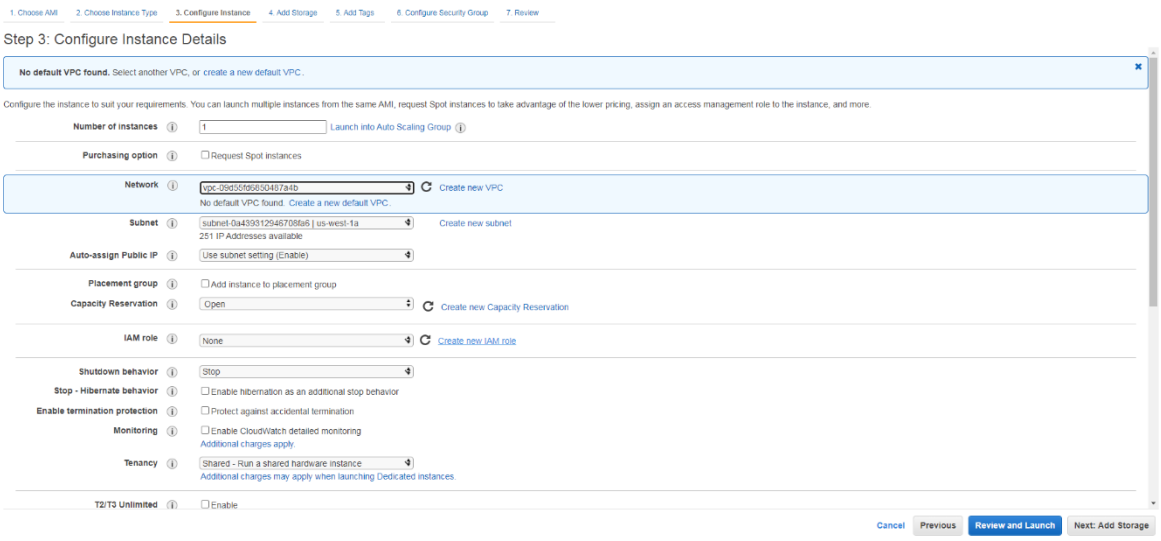

1. Configure Load Balancer
2. Configure Security Settings
3. Configure Routing
4. Register Targets
5. Review
Step 5: Review
Please review the load balancer details before continuing
Load balancer
Edit
Name NLBLab6a
Scheme internal
Listeners Port:8080 - Protocol:TCP
IP address type ipv4
VPC vpc-0b9ac9618896976a4
Subnets subnet-0d3a0b702df4c5820, subnet-0d3476134781bfe69
Tags
Security groups
Edit
Routing
Edit
Target group Existing target group
Target group name TG-Port-8080
Port 8080
Target type instance
Protocol TCP
Health check protocol TCP
Health check port traffic port
Healthy threshold 3
Unhealthy threshold 3
Interval 30
Targets
Edit
Instances
Cancel
Previous
Create

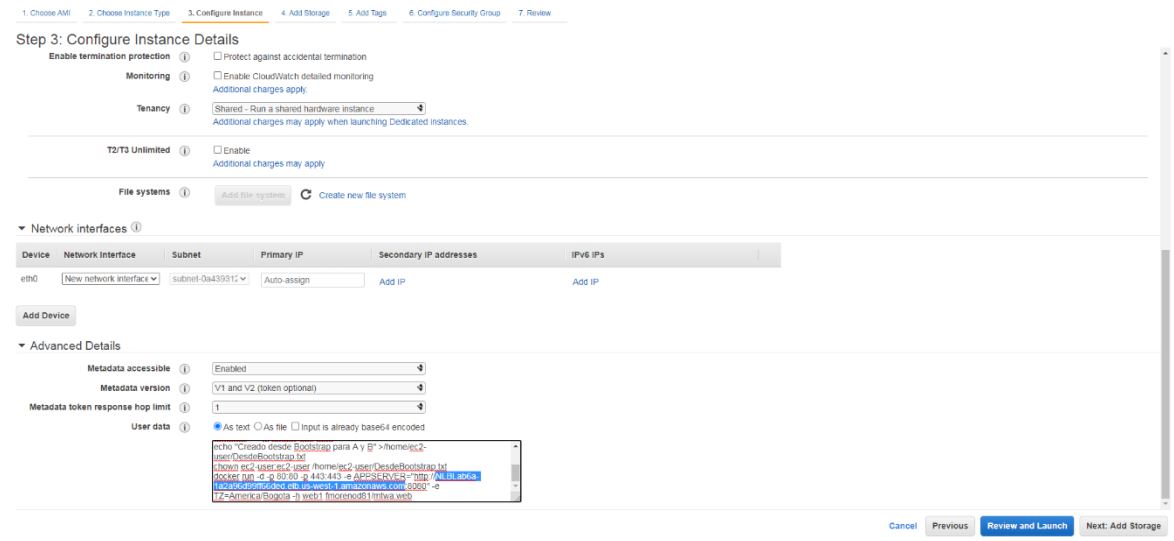


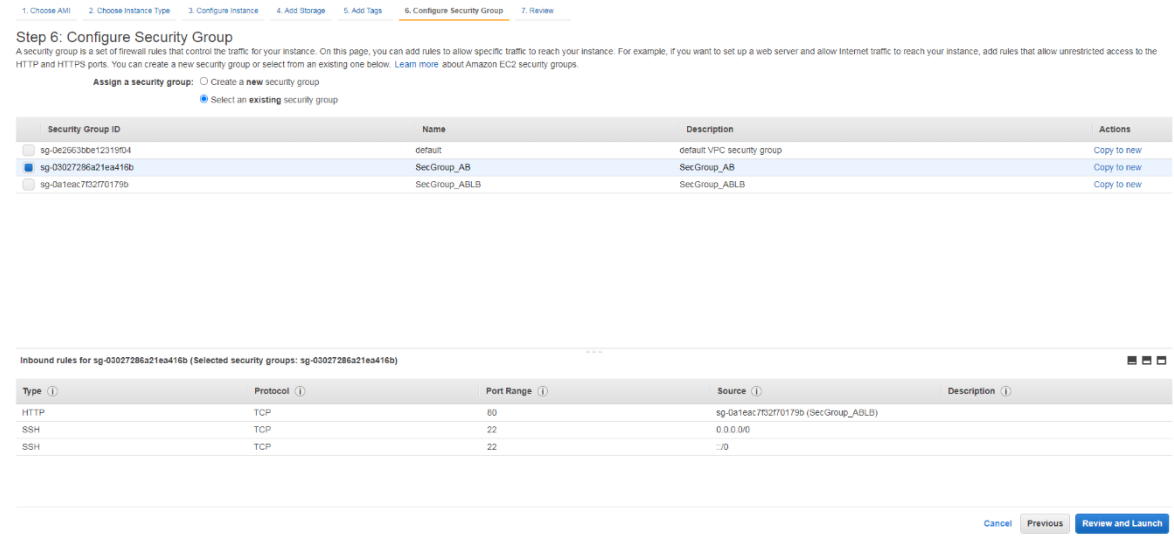
Copy the Internal NLB to configure internal variable on Web App

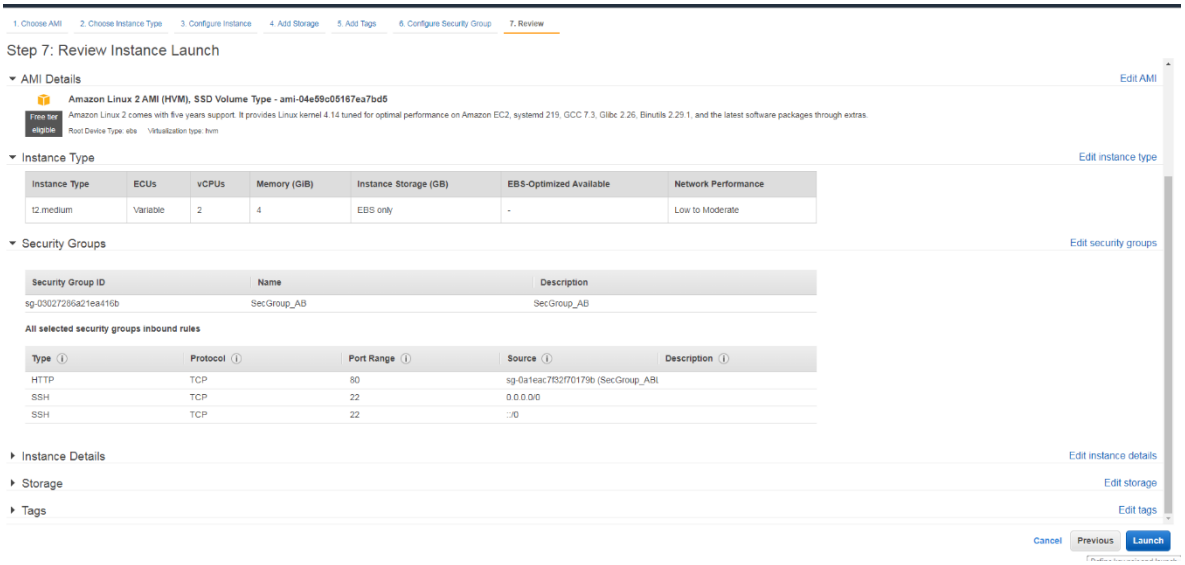
Configure Web Instances using Internal NLB

We use the normal procedure to create an instance: correct subnet, security group and the configuration of the initial script.



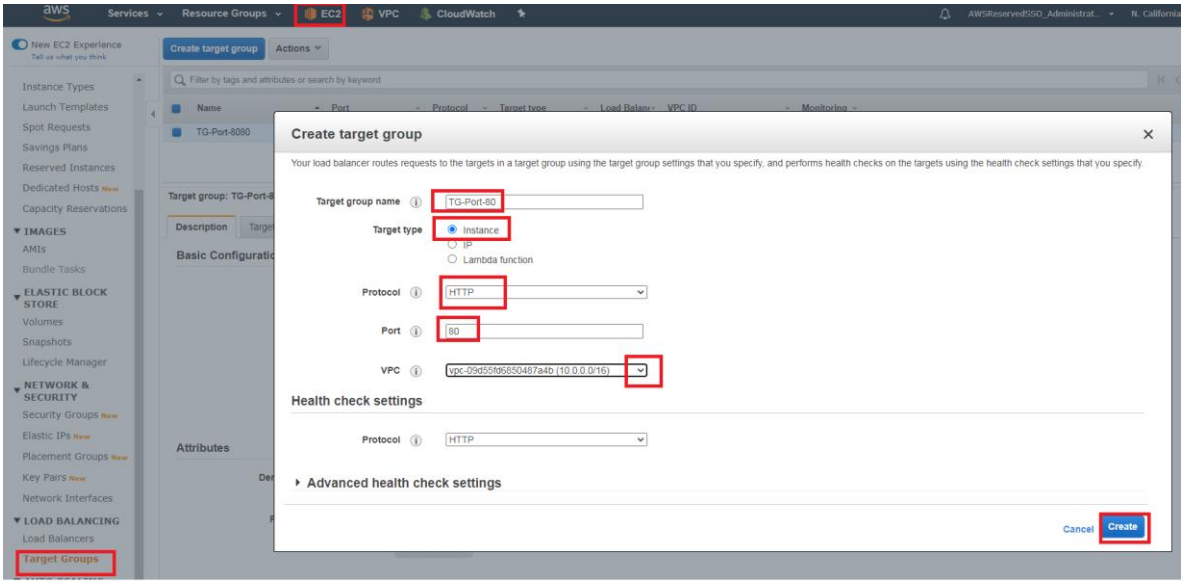

1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 3: Configure Instance Details
Enable termination protection
Protect against accidental termination
Monitoring
Enable CloudWatch detailed monitoring
Additional charges apply.
Tenancy
Shared - Run a shared hardware instance
Additional charges may apply when launching Dedicated instances.
T2/T3 Unlimited
Enable
Additional charges may apply
File systems
Add file system
Create new file system
Network interfaces
Device
Network Interface
Subnet
Primary IP
Secondary IP addresses
IPv6 IPs
eth0
New network interface
Auto-assign
Add IP
Add IP
Add Device
Advanced Details
Metadata accessible
Enabled
Metadata version
V1 and V2 (token optional)
Metadata token response hop limit
1
User data
As text
As file
Input is already base64 encoded
echo "Creado desde Bootstrap para A y B" >/home/ec2-user/DesdeBootstrap.txt
chown ec2-user:ec2-user /home/ec2-user/DesdeBootstrap.txt
docker run -d -p 80:80 -p 443:443 -e APPSERVER="http://NLBLabGa-1a2a9d99f56ded.elb.us-west-1.amazonaws.com:8080" -e TZ=America/Bogota -h web1 fmorenod81/mtwa:web
Cancel
Previous
Review and Launch
Next: Add Storage

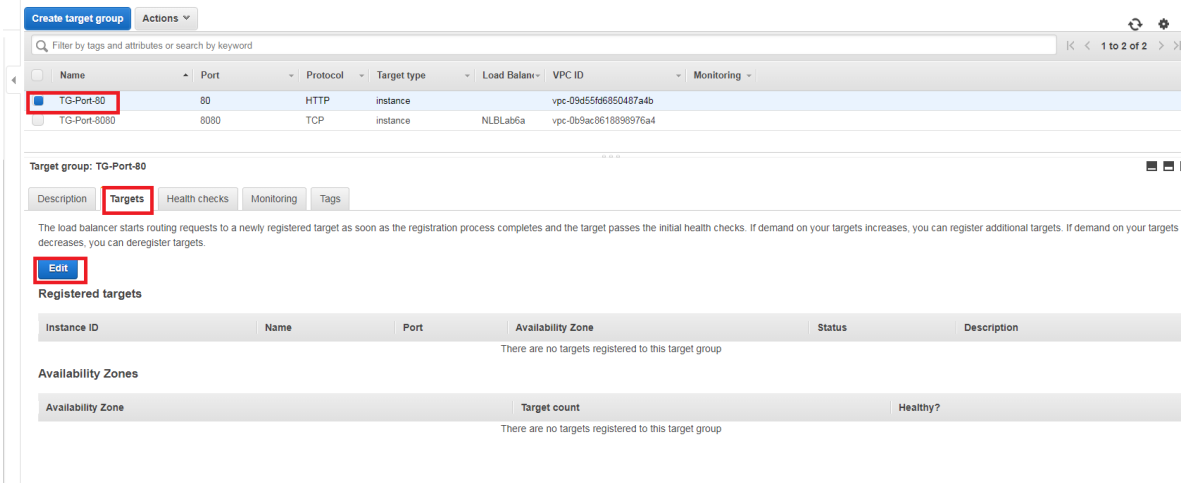

1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 6: Configure Security Group
A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.
Assign a security group:
Create a new security group
Select an existing security group
Security Group ID
Name
Description
Actions
sg-0e2663bbe12319f04
default
default VPC security group
Copy to new
sg-03027286a21ea416b
SecGroup_AB
SecGroup_AB
Copy to new
sg-0a1eac7f32f70179b
SecGroup_ABLB
SecGroup_ABLB
Copy to new
Inbound rules for sg-03027286a21ea416b (Selected security groups: sg-03027286a21ea416b)
Type
Protocol
Port Range
Source
Description
HTTP
TCP
80
sg-0a1eac7f32f70179b (SecGroup_ABLB)
SSH
TCP
22
0.0.0.0/0
SSH
TCP
22
::/0
Cancel
Previous
Review and Launch

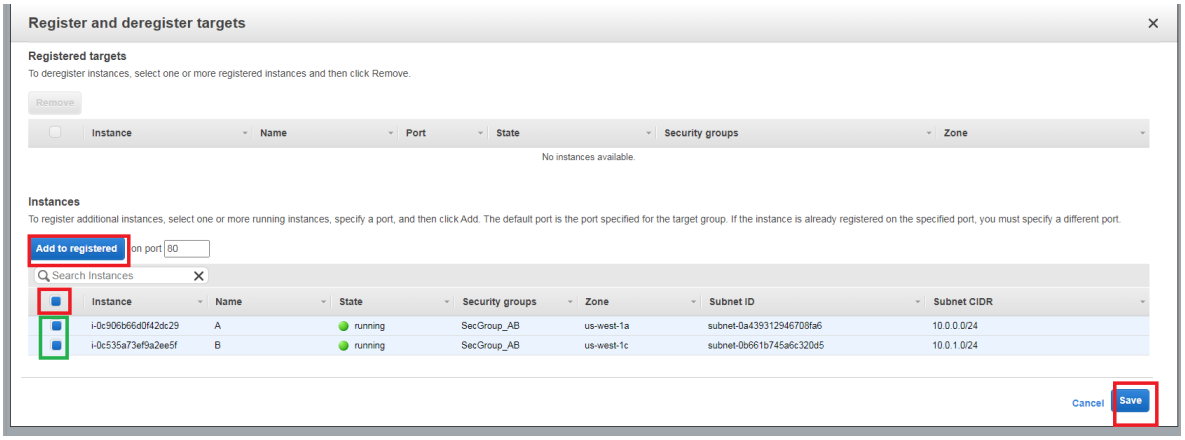


Create Target Group for Web Tier and register instances

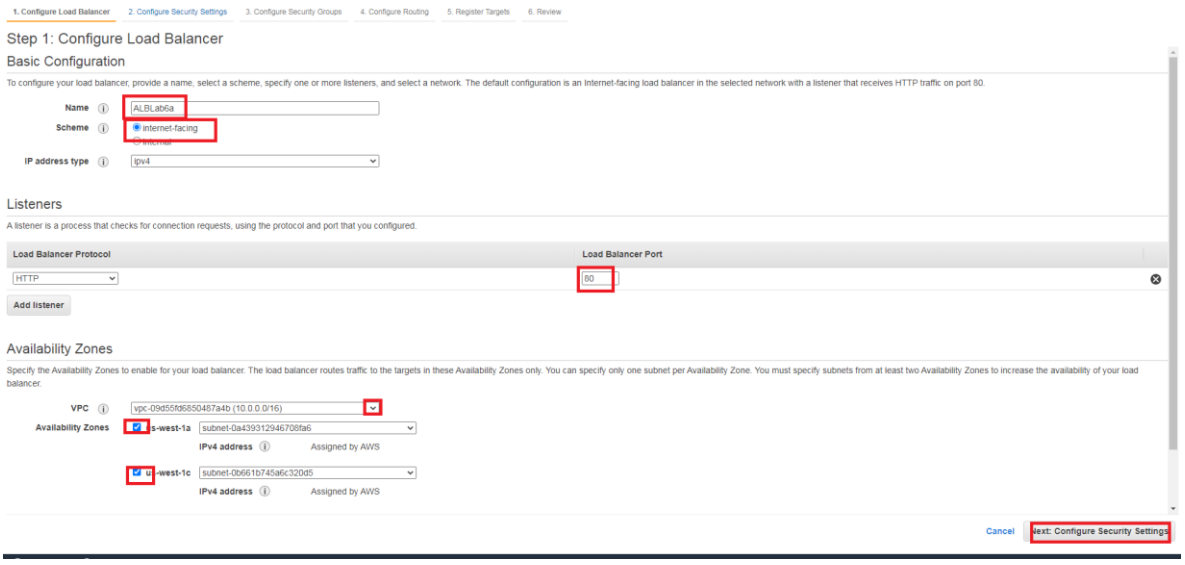
As the same procedure for App Tier.

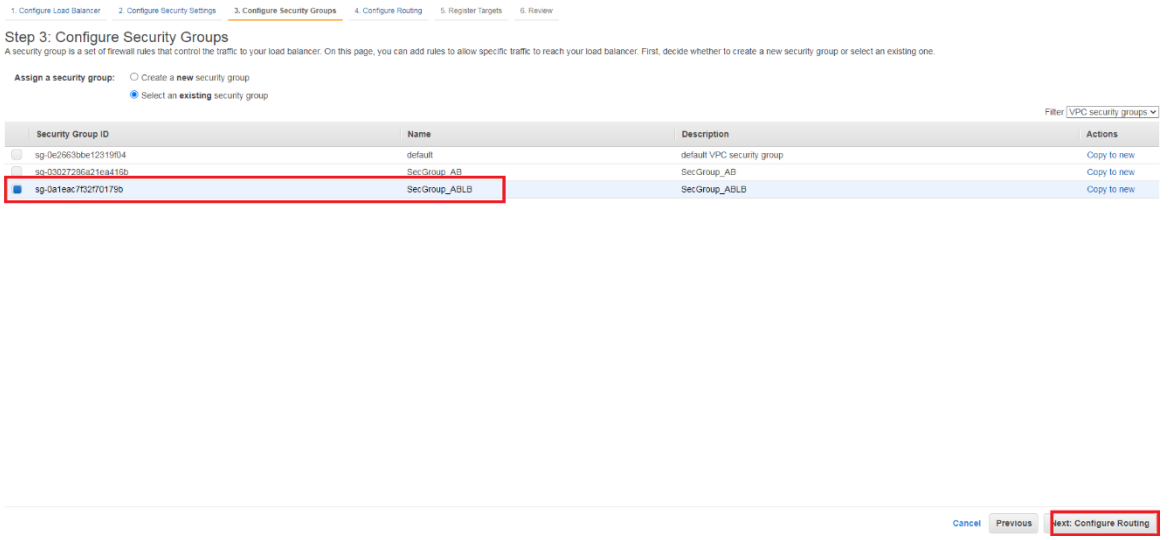


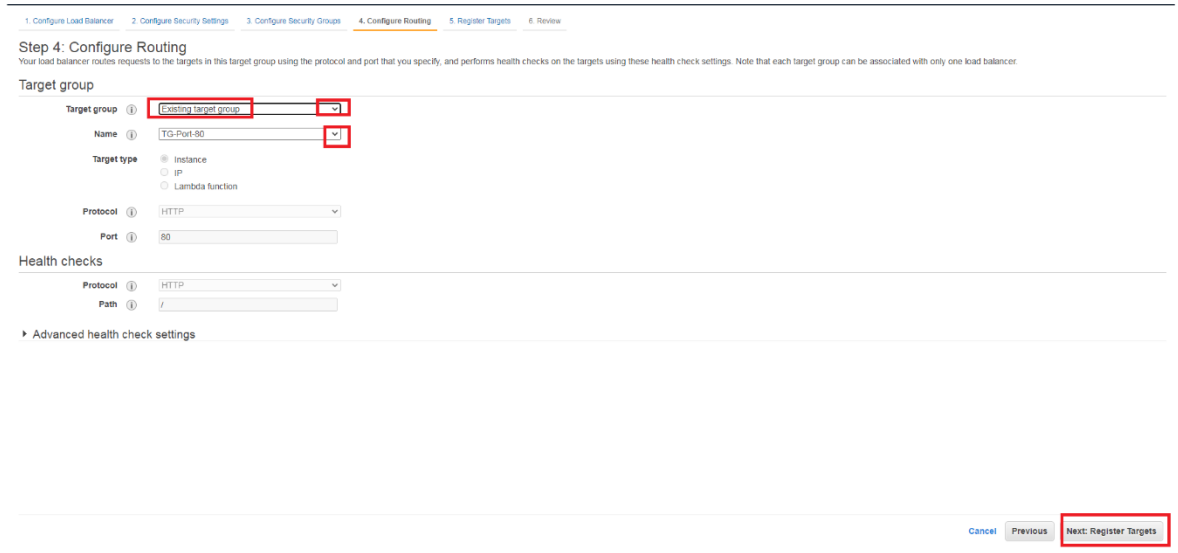

Create target group
Actions
Filter by tags and attributes or search by keyword
1 to 2 of 2
Name
Port
Protocol
Target type
Load Balan
VPC ID
Monitoring
TG-Port-80
80
HTTP
instance
vpc-09d55fd6850487a4b
TG-Port-8080
8080
TCP
instance
NLBLab6a
vpc-0b9ac8618898976a4
Target group: TG-Port-80
Description
Targets
Health checks
Monitoring
Tags
The load balancer starts routing requests to a newly registered target as soon as the registration process completes and the target passes the initial health checks. If demand on your targets increases, you can register additional targets. If demand on your targets decreases, you can deregister targets.
Edit
Registered targets
Instance ID
Name
Port
Availability Zone
Status
Description
There are no targets registered to this target group
Availability Zones
Availability Zone
Target count
Healthy?
There are no targets registered to this target group

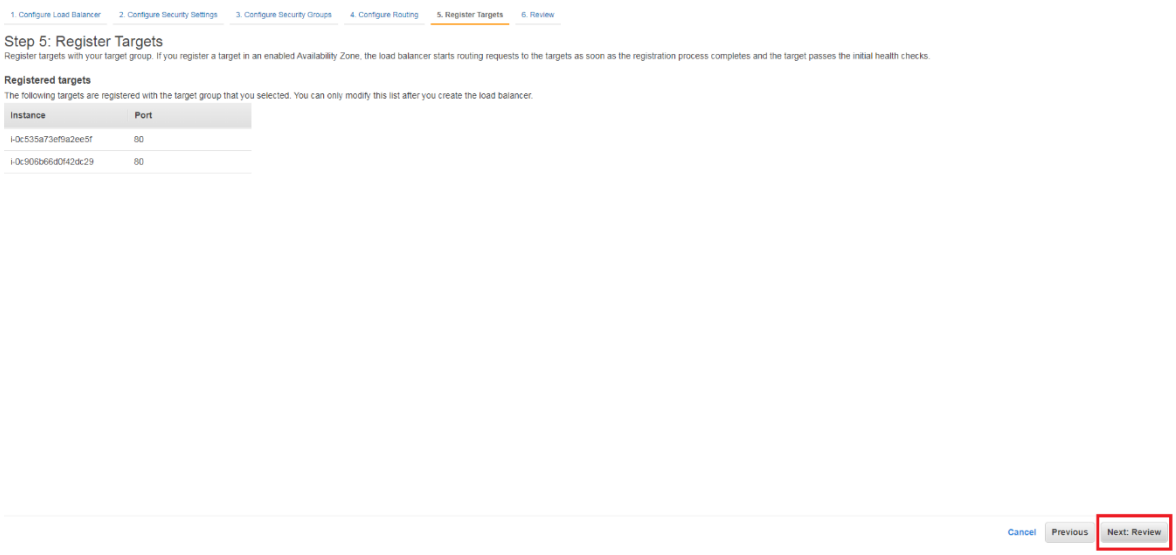

Register and deregister targets
Registered targets
To deregister instances, select one or more registered instances and then click Remove.
Remove
Instance
Name
Port
State
Security groups
Zone
No instances available.
Instances
To register additional instances, select one or more running instances, specify a port, and then click Add. The default port is the port specified for the target group. If the instance is already registered on the specified port, you must specify a different port.
Add to registered
on port
80
Search Instances
Instance
Name
State
Security groups
Zone
Subnet ID
Subnet CIDR
i-0c906b66d0f42dc29
A
running
SecGroup_AB
us-west-1a
subnet-0a439312946708fa6
10.0.0.0/24
i-0c535a73ef9a2ee5f
B
running
SecGroup_AB
us-west-1c
subnet-0b661b745a6c320d5
10.0.1.0/24
Cancel
Save

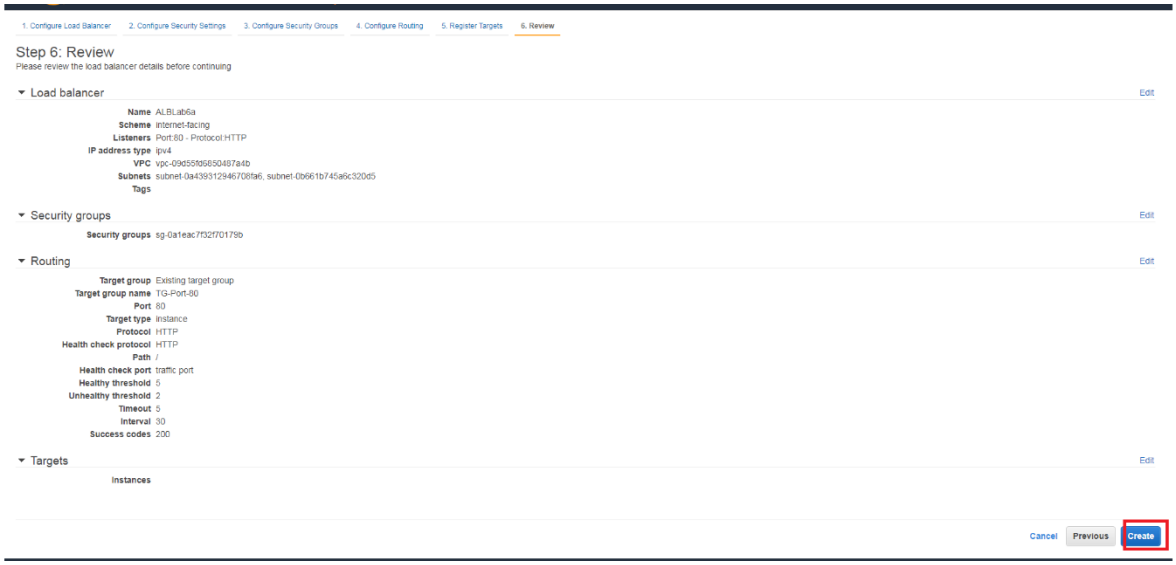
Create ALB

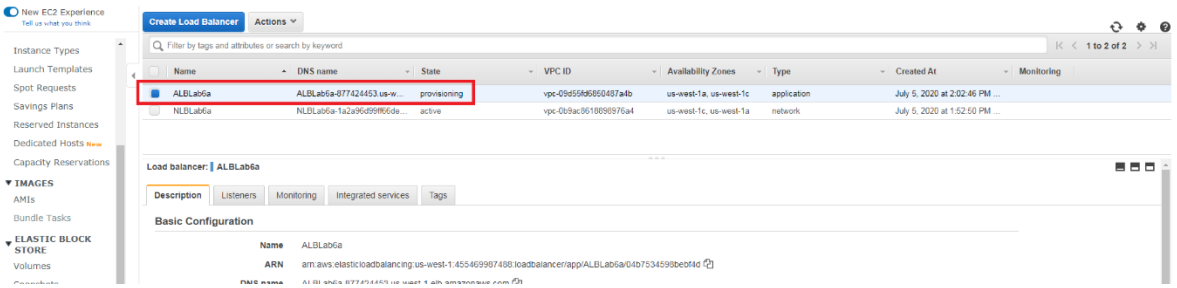

1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 1: Configure Load Balancer
Basic Configuration
To configure your load balancer, provide a name, select a scheme, specify one or more listeners, and select a network. The default configuration is an Internet-facing load balancer in the selected network with a listener that receives HTTP traffic on port 80.
Name
ALBLab6a
Scheme
internet-facing
IP address type
ipv4
Listeners
A listener is a process that checks for connection requests, using the protocol and port that you configured.
Load Balancer Protocol
Load Balancer Port
HTTP
80
Add listener
Availability Zones
Specify the Availability Zones to enable for your load balancer. The load balancer routes traffic to the targets in these Availability Zones only. You can specify only one subnet per Availability Zone. You must specify subnets from at least two Availability Zones to increase the availability of your load balancer.
VPC
vpc-09d55fd6850487a4b (10.0.0.0/16)
Availability Zones
s-west-1a
subnet-0a439312946708fa6
IPv4 address
Assigned by AWS
-west-1c
subnet-0b661b745a6c320d5
IPv4 address
Assigned by AWS
Cancel
Next: Configure Security Settings

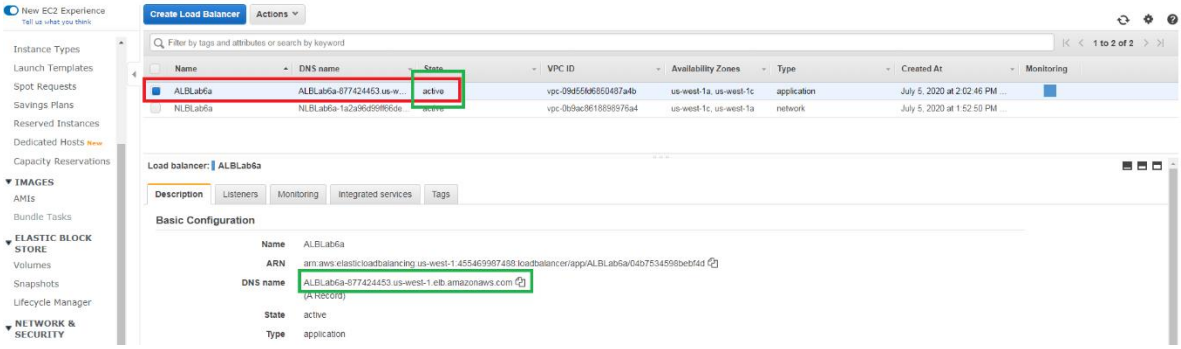

1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 3: Configure Security Groups
A security group is a set of firewall rules that control the traffic to your load balancer. On this page, you can add rules to allow specific traffic to reach your load balancer. First, decide whether to create a new security group or select an existing one.
Assign a security group:
Create a new security group
Select an existing security group
Filter VPC security groups
Security Group ID
Name
Description
Actions
sg-0e2663bbe12319f04
default
default VPC security group
Copy to new
sg-0302728a21ea416b
SecGroup_AB
SecGroup_AB
Copy to new
sg-0a1eac7f32f70179b
SecGroup_ABLB
SecGroup_ABLB
Copy to new
Cancel
Previous
Next: Configure Routing


1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 4: Configure Routing
Your load balancer routes requests to the targets in this target group using the protocol and port that you specify, and performs health checks on the targets using these health check settings. Note that each target group can be associated with only one load balancer.
Target group
Target group
Existing target group
Name
TG-Port-80
Target type
Instance
IP
Lambda function
Protocol
HTTP
Port
80
Health checks
Protocol
HTTP
Path
/
Advanced health check settings
Cancel
Previous
Next: Register Targets


1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 5: Register Targets
Register targets with your target group. If you register a target in an enabled Availability Zone, the load balancer starts routing requests to the targets as soon as the registration process completes and the target passes the initial health checks.
Registered targets
The following targets are registered with the target group that you selected. You can only modify this list after you create the load balancer.
Instance
Port
i-0c535a73ef9a2ee5f
80
i-0c906b66d0f42dc29
80
Cancel
Previous
Next: Review


1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 6: Review
Please review the load balancer details before continuing
Load balancer
Edit
Name ALBLab6a
Scheme internet-facing
Listeners Port:80 - Protocol:HTTP
IP address type ipv4
VPC vpc-09d55fd6850487a4b
Subnets subnet-0a439312946708fa6, subnet-0b661b745a6c320d5
Tags
Security groups
Edit
Security groups sg-0a1eac7f32f70179b
Routing
Edit
Target group Existing target group
Target group name TG-Port-80
Port 80
Target type instance
Protocol HTTP
Health check protocol HTTP
Path /
Health check port traffic port
Healthy threshold 5
Unhealthy threshold 2
Timeout 5
Interval 30
Success codes 200
Targets
Edit
Instances
Cancel
Previous
Create


New EC2 Experience
Tell us what you think
Create Load Balancer
Actions
Filter by tags and attributes or search by keyword
1 to 2 of 2
Name
DNS name
State
VPC ID
Availability Zones
Type
Created At
Monitoring
ALBLab6a
ALBLab6a-877424453.us-w...
provisioning
vpc-09d55fd6850487a4b
us-west-1a, us-west-1c
application
July 5, 2020 at 2:02:46 PM ...
NLBLab6a
NLBLab6a-1a2a96d99ff66de...
active
vpc-0b9ac8618898976a4
us-west-1c, us-west-1a
network
July 5, 2020 at 1:52:50 PM ...
Load balancer: ALBLab6a
Description
Listeners
Monitoring
Integrated services
Tags
Basic Configuration
Name ALBLab6a
ARN arn:aws:elasticloadbalancing:us-west-1:455469987488:loadbalancer/app/ALBLab6a/04b7534598bebf4d
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts New
Capacity Reservations
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes


New EC2 Experience
Tell us what you think
Create Load Balancer
Actions
Filter by tags and attributes or search by keyword
1 to 2 of 2
Name
DNS name
State
VPC ID
Availability Zones
Type
Created At
Monitoring
ALBLab6a
ALBLab6a-877424453.us-w...
active
vpc-09d55fd6850487a4b
us-west-1a, us-west-1c
application
July 5, 2020 at 2:02:46 PM ...
NLBLab6a
NLBLab6a-1a2a96d99ff66de...
vpc-0b9ac8618898976a4
us-west-1c, us-west-1a
network
July 5, 2020 at 1:52:50 PM ...
Load balancer: ALBLab6a
Description
Listeners
Monitoring
Integrated services
Tags
Basic Configuration
Name ALBLab6a
ARN arn:aws:elasticloadbalancing:us-west-1:455469987488:loadbalancer/app/ALBLab6a/04b7534598bebf4d
DNS name ALBLab6a-877424453.us-west-1.elb.amazonaws.com
(A Record)
State active
Type application
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

Review with the browser

You can play with the healthcheck to determine the possible responses to load. Please attend the batch file (Labs6c1.bat), section PLAY WITH HEALTHCHECK.

Lab 6A using Command Line (Windows)

Create VPC, Subnet, IGW, NAT, Routing Table, Keys, Sec Groups (Labs4c1) VPC Peering (Labs4c2), Instances on App Layer (Labs5c1)

```
rem Setear las variables de su grupo. Clase A: 10.x.x.x/8 Clase B: 172.16.x.
x a 172.31.x.x
set vpcn_Mask="10.0.0.0/16"
set pbsn1_Mask="10.0.0.0/24"
set pbsn2_Mask="10.0.1.0/24"
set vpcp_Mask="172.16.0.0/16"
set pbsp1_Mask="172.16.0.0/24"
set pbsp2_Mask="172.16.1.0/24"
set pbsn3_Mask="172.16.2.0/24"

set first_az="us-west-1a"
set second_az="us-west-1c"
set instance_type="t3.small"

rem Crear las VPC y habilitar resolucion DNS
aws ec2 create-vpc --cidr-block %vpcn_Mask%|jq ".Vpc.VpcId" >tmpFile
set /p vpcn_Id= < tmpFile
aws ec2 modify-vpc-attribute --vpc-id %vpcn_Id% --enable-dns-
hostnames "{\"Value\":true}"
aws ec2 create-vpc --cidr-block %vpcp_Mask%|jq ".Vpc.VpcId" >tmpFile
set /p vpcp_Id= < tmpFile
aws ec2 modify-vpc-attribute --vpc-id %vpcp_Id% --enable-dns-
hostnames "{\"Value\":true}"

rem Crear subredes Publicas
aws ec2 create-subnet --vpc-id %vpcn_Id% --cidr-block %pbsn1_Mask% --
availability-zone %first_az%|jq ".Subnet.SubnetId" >tmpFile
set /p pbsn1_Id= < tmpFile
aws ec2 create-subnet --vpc-id %vpcn_Id% --cidr-block %pbsn2_Mask% --
availability-zone %second_az%|jq ".Subnet.SubnetId" >tmpFile
set /p pbsn2_Id= < tmpFile
rem Permitir que las instancias que se ejecutan en la subredes se hagan publ
icas. Ver https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Internet_Gate
way.html
aws ec2 modify-subnet-attribute --subnet-id %pbsn1_Id% --map-public-ip-on-
launch
aws ec2 modify-subnet-attribute --subnet-id %pbsn2_Id% --map-public-ip-on-
launch

rem Crear el Internet Gateway IGW y asignarlo a la VPC
```

```
aws ec2 create-internet-
gateway|jq ".InternetGateway.InternetGatewayId"  >tmpFile
set /p IGW_Id= < tmpFile
aws ec2 attach-internet-gateway --vpc-id %vpcn_Id% --internet-gateway-
id %IGW_Id%

rem Crear tabla de ruteo publica y asignarle IGW como ruta por defecto
aws ec2 create-route-table --vpc-
id %vpcn_Id%|jq ".RouteTable.RouteTableId" >tmpFile
set /p Public_RT_Id= < tmpFile
aws ec2 create-route --route-table-id %Public_RT_Id% --destination-cidr-
block 0.0.0.0/0 --gateway-id %IGW_Id%

rem Asociar la tabla de ruta a las subredes
aws ec2 associate-route-table  --subnet-id %pbsn1_Id% --route-table-
id %Public_RT_Id%
aws ec2 associate-route-table  --subnet-id %pbsn2_Id% --route-table-
id %Public_RT_Id%

rem Redes privadas

rem Crear subredes Privadas y la unica publica para el NAT
aws ec2 create-subnet --vpc-id %vpcp_Id% --cidr-block %pbsp1_Mask% --
availability-zone %first_az%|jq ".Subnet.SubnetId" >tmpFile
set /p pbsp1_Id= < tmpFile
aws ec2 create-subnet --vpc-id %vpcp_Id% --cidr-block %pbsp2_Mask% --
availability-zone %second_az%|jq ".Subnet.SubnetId" >tmpFile
set /p pbsp2_Id= < tmpFile
aws ec2 create-subnet --vpc-id %vpcp_Id% --cidr-block %pbsn3_Mask% --
availability-zone %second_az%|jq ".Subnet.SubnetId" >tmpFile
set /p pbsn3_Id= < tmpFile

rem Solicitar una IP Elastica para hacer el Nat Gateway
aws ec2 allocate-address --domain vpc |jq ".AllocationId" >tmpFile
set /p NAT_EIP= < tmpFile

rem Crear el NAT Gateway, asignarlo a una EIP Anterior.
aws ec2 create-nat-gateway --subnet-id  %pbsn3_Id% --allocation-
id %NAT_EIP%|jq ".NatGateway.NatGatewayId" >tmpFile
set /p NATGW_Id= < tmpFile

rem Crear el Internet Gateway IGW y asignarlo a la VPC
aws ec2 create-internet-
gateway|jq ".InternetGateway.InternetGatewayId"  >tmpFile
set /p IGW2_Id= < tmpFile
```

```
aws ec2 attach-internet-gateway --vpc-id %vpcp_Id% --internet-gateway-
id %IGW2_Id%

rem Crear tabla de ruteo publica de la NAT para las redes privadas y asignar
 el IGW como ruta por defecto. Asociarla
aws ec2 create-route-table --vpc-
id %vpcp_Id%|jq ".RouteTable.RouteTableId" >tmpFile
set /p Public_Private_RT_Id= < tmpFile
aws ec2 create-route --route-table-id %Public_Private_RT_Id% --destination-
cidr-block 0.0.0.0/0 --gateway-id %IGW2_Id%
aws ec2 associate-route-table  --subnet-id %pbsn3_Id% --route-table-
id %Public_Private_RT_Id%


rem Crear tabla de ruteo publica para las redes privadas y asignar el NAT GW
 como ruta por defecto. Asociarla
aws ec2 create-route-table --vpc-
id %vpcp_Id%|jq ".RouteTable.RouteTableId" >tmpFile
set /p Private_RT_Id= < tmpFile
aws ec2 create-route --route-table-id %Private_RT_Id% --destination-cidr-
block 0.0.0.0/0 --nat-gateway-id %NATGW_Id%
aws ec2 associate-route-table  --subnet-id %pbsp1_Id% --route-table-
id %Private_RT_Id%
aws ec2 associate-route-table  --subnet-id %pbsp2_Id% --route-table-
id %Private_RT_Id%

rem Crear y aceptar el VPC Peering
aws ec2 create-vpc-peering-connection --vpc-id %vpcn_Id% --peer-vpc-
id %vpcp_Id%|jq ".VpcPeeringConnection.VpcPeeringConnectionId" >tmpFile
set /p VPCPeering_Id= < tmpFile
aws ec2 accept-vpc-peering-connection --vpc-peering-connection-
id %VPCPeering_Id%
rem Agregar las rutas en las 2 tablas de ruteo
aws ec2 create-route --route-table-id %Private_RT_Id% --destination-cidr-
block %vpcn_Mask% --vpc-peering-connection-id %VPCPeering_Id%
aws ec2 create-route --route-table-id %Public_RT_Id% --destination-cidr-
block %vpcp_Mask% --vpc-peering-connection-id %VPCPeering_Id%


rem Crear las llaves para el SSH a las nuevas instancias y convertirlas a PP
K para usar Putty ya sea con puttygen o winscp
aws ec2 create-key-pair --key-name Lab6a --query "KeyMaterial" --
output text > Lab6a.pem
winscp.com /keygen "Lab6a.pem" /output="Lab6a.ppk"
```

```
rem Crear los Security Groups para instancias A y B
aws ec2 create-security-group --group-name "SecGroup_AB" --
description "Security group for Instances A and B" --vpc-
id %vpcn_Id% |jq ".GroupId">tmpFile
set /p SecGroup_AB_Id= < tmpFile
aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --
protocol tcp --port 22 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --
protocol tcp --port 80 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --
protocol tcp --port 443 --cidr 0.0.0.0/0


rem Crear los Security Groups para instancias C y D
aws ec2 create-security-group --group-name "SecGroup_CD" --
description "Security group for Instances C and D" --vpc-
id %vpcp_Id% |jq ".GroupId">tmpFile
set /p SecGroup_CD_Id= < tmpFile
aws ec2 authorize-security-group-ingress --group-id %SecGroup_CD_Id% --
protocol tcp --port 22 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_CD_Id% --
protocol tcp --port 8080 --cidr 0.0.0.0/0


rem AWS sugiere que se tome el AMI Amazon Linux 2 y se instale docker desde
linea de comandos: https://docs.aws.amazon.com/AmazonECS/latest/developergui
de/docker-basics.html#install_docker
aws ec2 describe-images --owners amazon --filters "Name=name,Values=amzn2-
ami-hvm-2.0.????????.?-x86_64-gp2" "Name=state,Values=available" --
query "reverse(sort_by(Images, &CreationDate))[:1].ImageId" --
output text >tmpFile
set /p AMI= < tmpFile
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set instance_type="t3.small"

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-vpc --cidr-block %vpcn_Mask%|jq ".Vpc.VpcId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p vpcn_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 modify-vpc-attribute --vpc-id %vpcn_Id% --enable-dns-hostnames "{\"Value\":true}"


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-vpc --cidr-block %vpcp_Mask%|jq ".Vpc.VpcId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p vpcp_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 modify-vpc-attribute --vpc-id %vpcp_Id% --enable-dns-hostnames "{\"Value\":true}"


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-subnet --vpc-id %vpcn_Id% --cidr-block %pbsn1_Mask% --availability-zone %first_az%|jq ".Subnet.SubnetId" >t
mpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p pbsn1_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-subnet --vpc-id %vpcn_Id% --cidr-block %pbsn2_Mask% --availability-zone %second_az%|jq ".Subnet.SubnetId" >
tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p pbsn2_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 modify-subnet-attribute --subnet-id %pbsn1_Id% --map-public-ip-on-launch


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 modify-subnet-attribute --subnet-id %pbsn2_Id% --map-public-ip-on-launch


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-internet-gateway|jq ".InternetGateway.InternetGatewayId"  >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p IGW_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 attach-internet-gateway --vpc-id %vpcn_Id% --internet-gateway-id %IGW_Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-route-table --vpc-id %vpcn_Id%|jq ".RouteTable.RouteTableId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p Public_RT_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-route --route-table-id %Public_RT_Id% --destination-cidr-block 0.0.0.0/0 --gateway-id %IGW_Id%
{
    "Return": true
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 associate-route-table  --subnet-id %pbsn1_Id% --route-table-id %Public_RT_Id%
{
    "AssociationId": "rtbassoc-0e67afdfc9c0f97cd",
    "AssociationState": {
        "State": "associated"
    }
}
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 associate-route-table  --subnet-id %pbsn2_Id% --route-table-id %Public_RT_Id%
{
    "AssociationId": "rtbassoc-08ce8ec0ad01466d7",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-subnet --vpc-id %vpcp_Id% --cidr-block %pbsp1_Mask% --availability-zone %first_az%|jq ".Subnet.SubnetId" >t
mpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p pbsp1_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-subnet --vpc-id %vpcp_Id% --cidr-block %pbsp2_Mask% --availability-zone %second_az%|jq ".Subnet.SubnetId" >
tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p pbsp2_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-subnet --vpc-id %vpcp_Id% --cidr-block %pbsn3_Mask% --availability-zone %second_az%|jq ".Subnet.SubnetId" >
tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p pbsn3_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 allocate-address --domain vpc |jq ".AllocationId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p NAT_EIP= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-nat-gateway --subnet-id  %pbsn3_Id% --allocation-id %NAT_EIP%|jq ".NatGateway.NatGatewayId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p NATGW_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-internet-gateway|jq ".InternetGateway.InternetGatewayId"  >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p IGW2_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 attach-internet-gateway --vpc-id %vpcp_Id% --internet-gateway-id %IGW2_Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-route-table --vpc-id %vpcp_Id%|jq ".RouteTable.RouteTableId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p Public_Private_RT_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-route --route-table-id %Public_Private_RT_Id% --destination-cidr-block 0.0.0.0/0 --gateway-id %IGW2_Id%
{
    "Return": true
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 associate-route-table  --subnet-id %pbsn3_Id% --route-table-id %Public_Private_RT_Id%
{
    "AssociationId": "rtbassoc-0429b68db9f39d36f",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-route-table --vpc-id %vpcp_Id%|jq ".RouteTable.RouteTableId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p Private_RT_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-route --route-table-id %Private_RT_Id% --destination-cidr-block 0.0.0.0/0 --nat-gateway-id %NATGW_Id%
{
    "Return": true
}
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 associate-route-table  --subnet-id %pbsp1_Id% --route-table-id %Private_RT_Id%
{
    "AssociationId": "rtbassoc-03b71babfda17581e",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 associate-route-table  --subnet-id %pbsp2_Id% --route-table-id %Private_RT_Id%
{
    "AssociationId": "rtbassoc-01ba32fcc02e94ab2",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-vpc-peering-connection --vpc-id %vpcn_Id% --peer-vpc-id %vpcp_Id%|jq ".VpcPeeringConnection.VpcPeering
ctionId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p VPCPeering_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id %VPCPeering_Id%
{
    "VpcPeeringConnection": {
        "AccepterVpcInfo": {
            "CidrBlock": "172.16.0.0/16",
            "CidrBlockSet": [
                {
                    "CidrBlock": "172.16.0.0/16"
                }
            ],
            "OwnerId": "455469987488",
            "PeeringOptions": {
                "AllowDnsResolutionFromRemoteVpc": false,
                "AllowEgressFromLocalClassicLinkToRemoteVpc": false,
                "AllowEgressFromLocalVpcToRemoteClassicLink": false
            },
            "VpcId": "vpc-0b9ac8618898976a4",
            "Region": "us-west-1"
        },
        "RequesterVpcInfo": {
            "CidrBlock": "10.0.0.0/16",
            "CidrBlockSet": [
                {
                    "CidrBlock": "10.0.0.0/16"
                }
            ],
            "OwnerId": "455469987488",
            "PeeringOptions": {
                "AllowDnsResolutionFromRemoteVpc": false,
                "AllowEgressFromLocalClassicLinkToRemoteVpc": false,
                "AllowEgressFromLocalVpcToRemoteClassicLink": false
            },
            "VpcId": "vpc-09d55fd6850487a4b",
            "Region": "us-west-1"
        },
        "Status": {
            "Code": "provisioning",
            "Message": "Provisioning"
        },
        "Tags": [],
        "VpcPeeringConnectionId": "pcx-088368f3ceae88b5c"
    }
}
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-route --route-table-id %Private_RT_Id% --destination-cidr-block %vpcn_Mask% --vpc-peering-connection-id %VP
CPeering_Id%
{
    "Return": true
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-route --route-table-id %Public_RT_Id% --destination-cidr-block %vpcp_Mask% --vpc-peering-connection-id %VPC
Peering_Id%
{
    "Return": true
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-key-pair --key-name Lab6a --query "KeyMaterial" --output text > Lab6a.pem

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>winscp.com /keygen "Lab6a.pem" /output="Lab6a.ppk"
Key saved to "Lab6a.ppk".

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-security-group --group-name "SecGroup_AB" --description "Security group for Instances A and B" --vpc-id %vp
cn_Id% |jq ".GroupId">tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p SecGroup_AB_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --protocol tcp --port 22 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --protocol tcp --port 80 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --protocol tcp --port 443 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 create-security-group --group-name "SecGroup_CD" --description "Security group for Instances C and D" --vpc-id %vp
cp_Id% |jq ".GroupId">tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p SecGroup_CD_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 authorize-security-group-ingress --group-id %SecGroup_CD_Id% --protocol tcp --port 22 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 authorize-security-group-ingress --group-id %SecGroup_CD_Id% --protocol tcp --port 8080 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>aws ec2 describe-images --owners amazon --filters "Name=name,Values=amzn2-ami-hvm-2.0.????????.?-x86_64-gp2" "Name=state,V
alues=available" --query "reverse(sort_by(Images, &CreationDate))[:1].ImageId" --output text >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1>set /p AMI= < tmpFile
```

Create instances for App Tier

```
rem Se solicitan instancias y se adiciona un bootstrap para comprobar que el
 docker fue instalado
rem Se arrancan con las instancias de backend ya que es necesario modificar
posteriormente la capa de presentacion con el nombre del balanceador
aws ec2 run-instances --image-id %AMI% --count 1 --instance-
type %instance_type% --key-name Lab6a --security-group-
ids %SecGroup_CD_Id% --subnet-id %pbsp1_Id% --tag-
specifications "ResourceType=instance,Tags=[{Key=Name,Value=C}]"  --user-
data file://bootstrapCD.txt |jq "[.Instances|.[].InstanceId|.]"|jq ".[0]" >t
mpFile
set /p Instance3Id= <tmpFile
aws ec2 run-instances --image-id %AMI% --count 1 --instance-
type %instance_type% --key-name Lab6a --security-group-
ids %SecGroup_CD_Id% --subnet-id %pbsp2_Id% --tag-
specifications "ResourceType=instance,Tags=[{Key=Name,Value=D}]"  --user-
data file://bootstrapCD.txt |jq "[.Instances|.[].InstanceId|.]"|jq ".[0]" >t
mpFile
set /p Instance4Id= <tmpFile
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 run-instances --image-id %AMI% --count 1 --instance-type %instance_type% --key-name Lab6a --security-group-ids
 %SecGroup_CD_Id% --subnet-id %pbsp1_Id% --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=C}]"  --user-data file://bootstrapCD.txt |jq "[.Insta
nces|.[].InstanceId|.]"|jq ".[0]" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p Instance3Id= <tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 run-instances --image-id %AMI% --count 1 --instance-type %instance_type% --key-name Lab6a --security-group-ids
 %SecGroup_CD_Id% --subnet-id %pbsp2_Id% --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=D}]"  --user-data file://bootstrapCD.txt |jq "[.Insta
nces|.[].InstanceId|.]"|jq ".[0]" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p Instance4Id= <tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>
```

Create Sec Groups for NLB, Target Groups, Register Instances and finally, create NLB and its listener

```
rem Se hace el Balanceador Interno con NLB
rem Crear los Security Group del Balanceador
aws ec2 create-security-group --group-name "SecGroup_CDNLB" --
description "Security group for NLB" --vpc-
id %vpcp_Id% |jq ".GroupId">tmpFile
set /p SecGroup_CDNLB_Id= < tmpFile
aws ec2 authorize-security-group-ingress --group-id %SecGroup_CDNLB_Id% --
protocol tcp --port 8080 --cidr 0.0.0.0/0
rem Permitir que el balanceador pueda ver las instancias, sirve para el bala
nceo y el healthcheck
aws ec2 authorize-security-group-ingress --group-id %SecGroup_CD_Id% --
protocol tcp --port 8080 --source-group %SecGroup_CDNLB_Id%

rem Crear los target groups y registrar las instancias a los mismos en cada
puerto
```

```
aws elbv2 create-target-group --name TG-Port-8080 --protocol TCP --
port 8080 --target-type instance --vpc-
id %vpcp_Id% |jq ".TargetGroups[].TargetGroupArn" >tmpFile
set /p TG8080_ARN= < tmpFile
aws elbv2 register-targets --target-group-arn %TG8080_ARN% --
targets Id=%Instance3Id% Id=%Instance4Id%

rem Crear el NLB
aws elbv2 create-load-balancer --name NLBLab6a --scheme internal --
type network --subnets %pbsp1_Id% %pbsp2_Id%  >tmpFile2
cat tmpFile2|jq ".LoadBalancers[].LoadBalancerArn" >tmpFile
set /p NLB_ARN= < tmpFile
cat tmpFile2|jq ".LoadBalancers[].DNSName" >tmpFile
set /p NLB_DNSName= < tmpFile
del tmpFile2

rem Se crea el Listener para Puerto 8080
aws elbv2 create-listener --load-balancer-arn %NLB_ARN% --protocol TCP --
port 8080 --default-
actions Type=forward,TargetGroupArn=%TG8080_ARN%|jq ".Listeners[].ListenerAr
n" >tmpFile
set /p LST8080_ARN= < tmpFile
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 create-security-group --group-name "SecGroup_CDNLB" --description "Security group for NLB" --vpc-id %vpcp_Id%
|jq ".GroupId">tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p SecGroup_CDNLB_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_CDNLB_Id% --protocol tcp --port 8080 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_CD_Id% --protocol tcp --port 8080 --source-group %SecGro
up_CDNLB_Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws elbv2 create-target-group --name TG-Port-8080 --protocol TCP --port 8080 --target-type instance --vpc-id %vpcp_Id%
 |jq ".TargetGroups[].TargetGroupArn" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p TG8080_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws elbv2 register-targets --target-group-arn %TG8080_ARN% --targets Id=%Instance3Id% Id=%Instance4Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws elbv2 create-load-balancer --name NLBLab6a --scheme internal --type network --subnets %pbsp1_Id% %pbsp2_Id%  >tmpF
ile2

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>cat tmpFile2|jq ".LoadBalancers[].LoadBalancerArn" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p NLB_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>cat tmpFile2|jq ".LoadBalancers[].DNSName" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p NLB_DNSName= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>del tmpFile2

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws elbv2 create-listener --load-balancer-arn %NLB_ARN% --protocol TCP --port 8080 --default-actions Type=forward,Targ
etGroupArn=%TG8080_ARN%|jq ".Listeners[].ListenerArn" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p LST8080_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>
```

Create instances for Web Tier using Internal balancer, Sec Groups for ALB, Target Groups for ALB, ALB and ALB' Listener

```
rem Se crean las instancias de la capa superiro modificando el bootstrap scr
ipt para que tomen el balanceador interno
echo El nombre del balanceador interno es %NLB_DNSName%
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>echo El nombre del balanceador interno es %NLB_DNSName%
El nombre del balanceador interno es "NLBLab6a-41bb660fd2aa2d63.elb.us-west-1.amazonaws.com"
```

You have to modify the bootstrapAB.txt file with the internal NLB

```
rem Antes de lanzarla se tiene que modificar el archivo bootrstrapAB.txt con
 el nombre del balanceador interno que esta en la variable %NLB_DNSName% agr
egarle el puerto 8080
aws ec2 run-instances --image-id %AMI% --count 1 --instance-
type %instance_type% --key-name Lab6a --security-group-
ids %SecGroup_AB_Id% --subnet-id %pbsn1_Id% --tag-
specifications "ResourceType=instance,Tags=[{Key=Name,Value=A}]"  --user-
data file://bootstrapAB.txt |jq "[.Instances|.[].InstanceId|.]"|jq ".[0]" >t
mpFile
set /p Instance1Id= <tmpFile
aws ec2 run-instances --image-id %AMI% --count 1 --instance-
type %instance_type% --key-name Lab6a --security-group-
ids %SecGroup_AB_Id% --subnet-id %pbsn2_Id% --tag-
specifications "ResourceType=instance,Tags=[{Key=Name,Value=B}]"  --user-
data file://bootstrapAB.txt |jq "[.Instances|.[].InstanceId|.]"|jq ".[0]" >t
mpFile
set /p Instance2Id= <tmpFile

rem Crear los Security Group del Balanceador
aws ec2 create-security-group --group-name "SecGroup_ABLB" --
description "Security group for ALB" --vpc-
id %vpcn_Id% |jq ".GroupId">tmpFile
set /p SecGroup_ABLB_Id= < tmpFile
aws ec2 authorize-security-group-ingress --group-id %SecGroup_ABLB_Id% --
protocol tcp --port 80 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_ABLB_Id% --
protocol tcp --port 443 --cidr 0.0.0.0/0
rem Permitir que el balanceador pueda ver las instancias, sirve para el bala
nceo y el healthcheck
aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --
protocol tcp --port 80 --source-group %SecGroup_ABLB_Id%
aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --
protocol tcp --port 443 --source-group %SecGroup_ABLB_Id%


rem Crear los target groups y registrar las instancias a los mismos en cada
puerto
aws elbv2 create-target-group --name TG-Port-80 --protocol HTTP --port 80 --
target-type instance --vpc-
id %vpcn_Id% |jq ".TargetGroups[].TargetGroupArn" >tmpFile
set /p TG80_ARN= < tmpFile
```

```
aws elbv2 register-targets --target-group-arn %TG80_ARN% --
targets Id=%Instance1Id% Id=%Instance2Id%

rem Crear el ALB
aws elbv2 create-load-balancer --name ALBLab6a --
subnets %pbsn1_Id% %pbsn2_Id% --security-groups %SecGroup_ABLB_Id% >tmpFile2
cat tmpFile2|jq ".LoadBalancers[].LoadBalancerArn" >tmpFile
set /p LB_ARN= < tmpFile
cat tmpFile2|jq ".LoadBalancers[].DNSName" >tmpFile
set /p LB_DNSName= < tmpFile
del tmpFile2

rem Se crea el Listener para Puerto 80
aws elbv2 create-listener --load-balancer-arn %LB_ARN% --protocol HTTP --
port 80 --default-
actions Type=forward,TargetGroupArn=%TG80_ARN%|jq ".Listeners[].ListenerArn"
 >tmpFile
set /p LST80_ARN= < tmpFile
rem Se prueba que el ALB llegue al target group desde un navegador despues d
e que el estado del ALB este en active
echo Para navegar a %LB_DNSName%
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 run-instances --image-id %AMI% --count 1 --instance-type %instance_type% --key-name Lab6a --security-group-ids
 %SecGroup_AB_Id% --subnet-id %pbsn1_Id% --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=A}]"  --user-data file://bootstrapAB.txt |jq "[.Insta
nces|.[].InstanceId|.]"|jq ".[0]" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p Instance1Id= <tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 run-instances --image-id %AMI% --count 1 --instance-type %instance_type% --key-name Lab6a --security-group-ids
 %SecGroup_AB_Id% --subnet-id %pbsn2_Id% --tag-specifications "ResourceType=instance,Tags=[{Key=Name,Value=B}]"  --user-data file://bootstrapAB.txt |jq "[.Insta
nces|.[].InstanceId|.]"|jq ".[0]" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p Instance2Id= <tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 create-security-group --group-name "SecGroup_ABLB" --description "Security group for ALB" --vpc-id %vpcn_Id% |
jq ".GroupId">tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p SecGroup_ABLB_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_ABLB_Id% --protocol tcp --port 80 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_ABLB_Id% --protocol tcp --port 443 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --protocol tcp --port 80 --source-group %SecGroup
_ABLB_Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_AB_Id% --protocol tcp --port 443 --source-group %SecGrou
p_ABLB_Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws elbv2 create-target-group --name TG-Port-80 --protocol HTTP --port 80 --target-type instance --vpc-id %vpcn_Id% |j
q ".TargetGroups[].TargetGroupArn" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p TG80_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws elbv2 register-targets --target-group-arn %TG80_ARN% --targets Id=%Instance1Id% Id=%Instance2Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws elbv2 create-load-balancer --name ALBLab6a --subnets %pbsn1_Id% %pbsn2_Id% --security-groups %SecGroup_ABLB_Id% >t
mpFile2

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>cat tmpFile2|jq ".LoadBalancers[].LoadBalancerArn" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p LB_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>cat tmpFile2|jq ".LoadBalancers[].DNSName" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p LB_DNSName= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>del tmpFile2

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>aws elbv2 create-listener --load-balancer-arn %LB_ARN% --protocol HTTP --port 80 --default-actions Type=forward,Target
GroupArn=%TG80_ARN%|jq ".Listeners[].ListenerArn" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>set /p LST80_ARN= < tmpFile
```

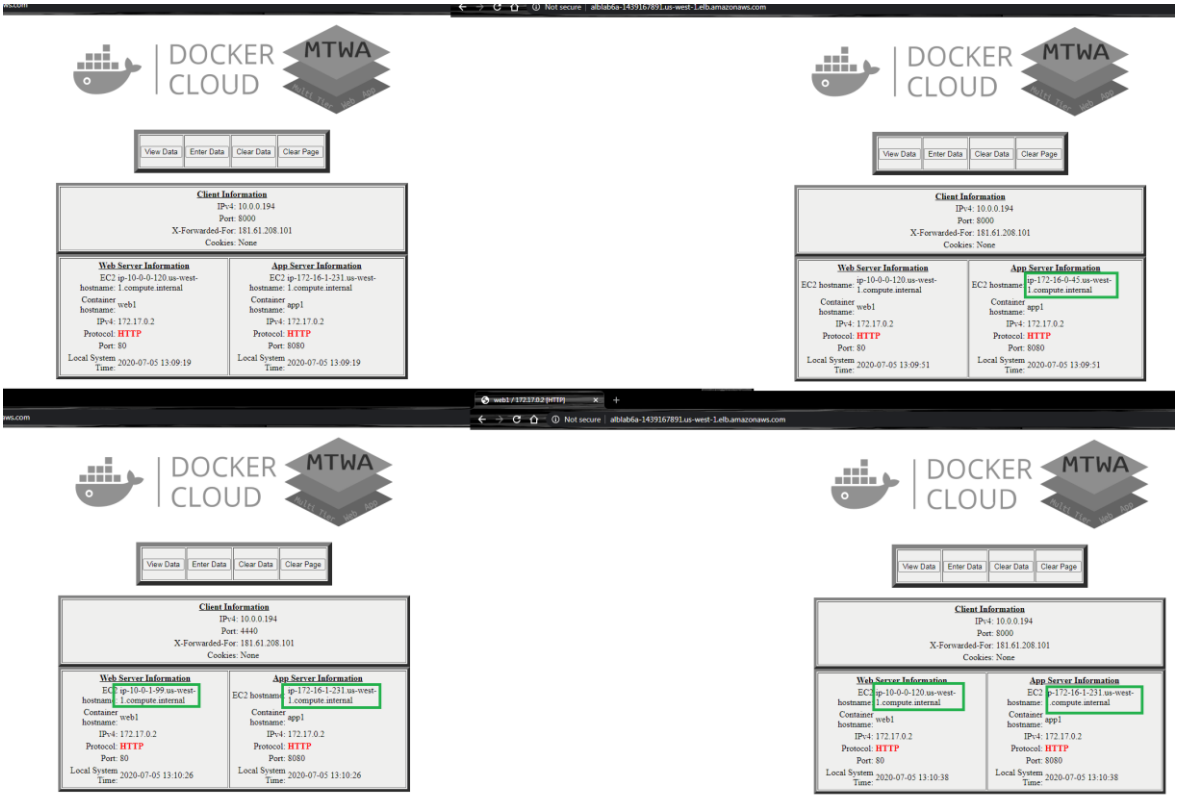
You have to check if its ALB is on Running State on Web Console and then you can go using browser.

```
:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>rem Se prueba que el ALB llegue al target group desde un navegador despues de que el estado del ALB este en active

:\Code\bsg-saa-c02\AWS_SAA\Code\s6c1\CLI>echo Para navegar a %LB_DNSName%
ara navegar a "ALBLab6a-1439167891.us-west-1.elb.amazonaws.com"
```

Review using browser

Using the browser make several refreshs and detect the changes of the Internal IP on Web and App Layers.




```
rem ********************** PLAY WITH HEALTHCHECK **********************
rem Modifica Health Check para NLB para deshabilitarlo o ponerlo en modo desconectado aleatoriamente
aws elbv2 modify-target-group --target-group-arn %TG80_ARN% --health-check-protocol HTTP --health-check-port 80 --health-check-path "/longdelay.py" >tmpFile
aws elbv2 describe-target-groups --target-group-arns %TG80_ARN%
```

```
rem Si ambos maquinas fallan, la documentacion dice que envia el paquete sin importar el estado asi que es mejor volver ir a una maquina A o B y detener el docker.
rem La documentacion esta en https://docs.aws.amazon.com/elasticloadbalancing/latest/application/target-group-health-checks.html y dice "If a target group contains only unhealthy registered targets, the load balancer routes requests to all those targets, regardless of their health status."
aws ec2 describe-instances --filters "Name=tag:Name,Values=A"  "Name=instance-state-name,Values=running"  |  jq -r  ".Reservations[] | .Instances[]|.PublicIpAddress" >tmpFile
set /p A_IP= < tmpFile
putty -i "Lab6a.ppk" ec2-user@%A_IP%
rem Ir al putty
docker ps -a
docker stop $(docker ps -aq)
docker rm $(docker ps -aq)
rem Para hacer estas revisiones es mejor ejecutarlo con la siguiente configuracion del healthcheck:
rem Healthy threshold y Unhealthy threshold   2
rem Timeout 17s
rem Interval 20s
rem Revisar el balanceador que solo lo envie a la instancia correcta...luego lo volvemos a ejecutar la ultima linea del bootstrapAB.txt en el docker y vamos al docker y reemplazamos el index.py por longdelay.py
docker run -d -p 80:80 -p 443:443 -e APPSERVER="http://NLBLab6a-a9219a23041903f7.elb.us-east-1.amazonaws.com:8080" -e TZ=America/Bogota -h web1 fmorenod81/mtwa:web
docker exec -ti 680 sh
cd /var/www/html/appdemo
mv longdelay.py index.py
rem Alli comprobamos que la maquina funciona sin embargo esta deshabilitada en el balanceador
mv shortdelay.py index.py
rem Volvemos a ejecutar el balanceador y esperamos que se ejecute el healthy a la instancia
rem Salir del putty
```

Clean Resources

VPC: Peering Connections
EC2: Instances
EC2: Load LoadBalancers
EC2: TargetGroups

VPC: NAT gateway
VPC: EIP
EC2: Keypair
EC2: Security Group
VPC: VPC