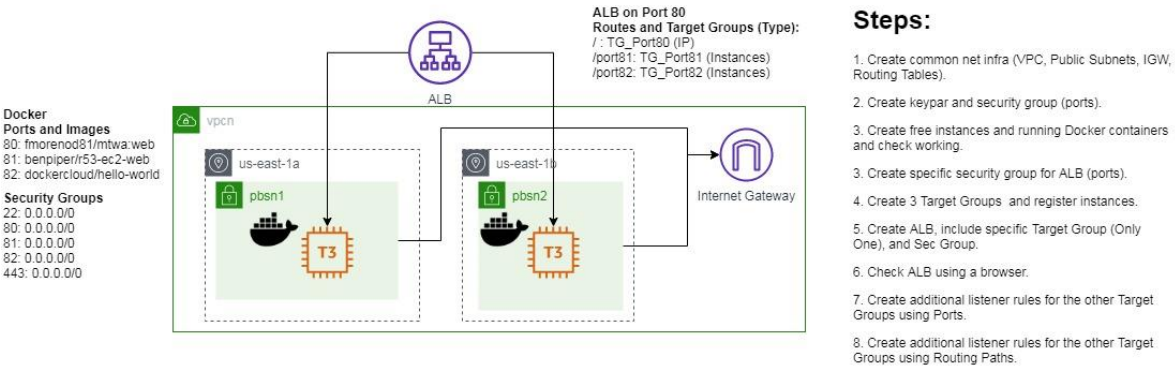
Contents

Purpose

Using network infrastructure as base, create a common infrastructure using

General Diagram

Simple public network infrastructure with a Spot Instance using Docker and bootstrap script. Public instance using Elastic IP (EIP) on an Elastic Network Interface (ENI).



Prerequisites

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

Labs4c1 have to be done, because you learn how to: Create subnets, VPCs, IGW, and Routing Tables. For this case specifically, you have to create VPC, Public Subnet, IGW, Routing Table with the same names as that laboratory, therefore we only focus on the new things.

Labs5c1 have to be done, because you learn how to: Create instances, create and apply security groups, install and run docker.

The bootstrap script using Base64 encode so you have to use one on Windows (`certutil -encode <infile> <outfile>`) or MacOs (`openssl base64 -in <infile> -out <outfile>`) or Web (https://www.base64decode.org/)

Lab 5B: ALB

Lab 5B using Web Management Console

Create Network Infrastructure, Routing Tables and IGW.

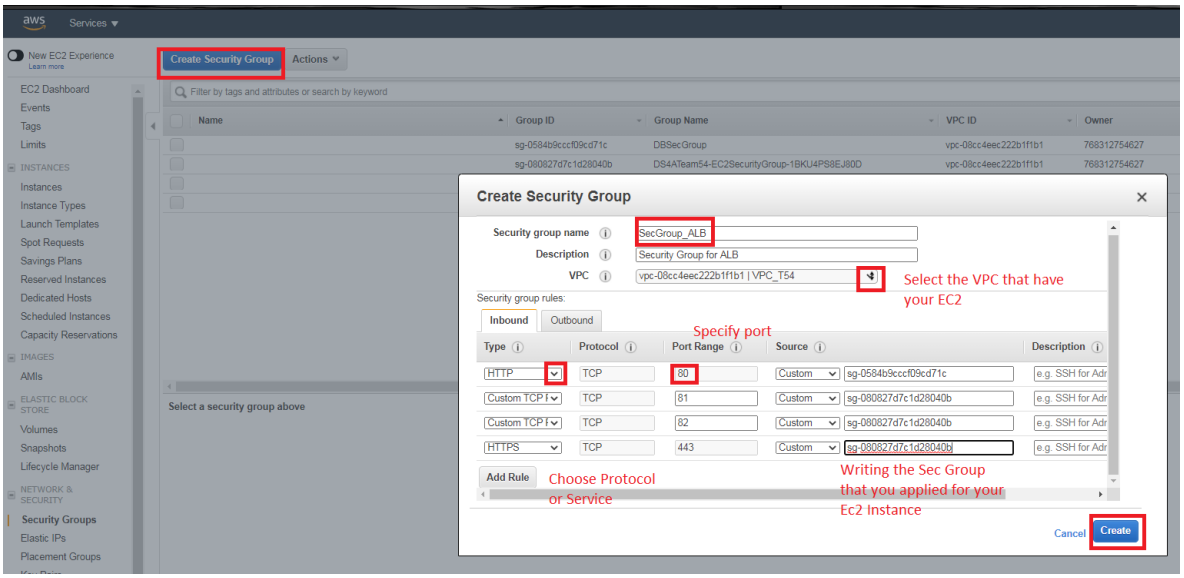
It's done on Lab4c1.

Running Instance, Security Groups and Running the Code on SSH

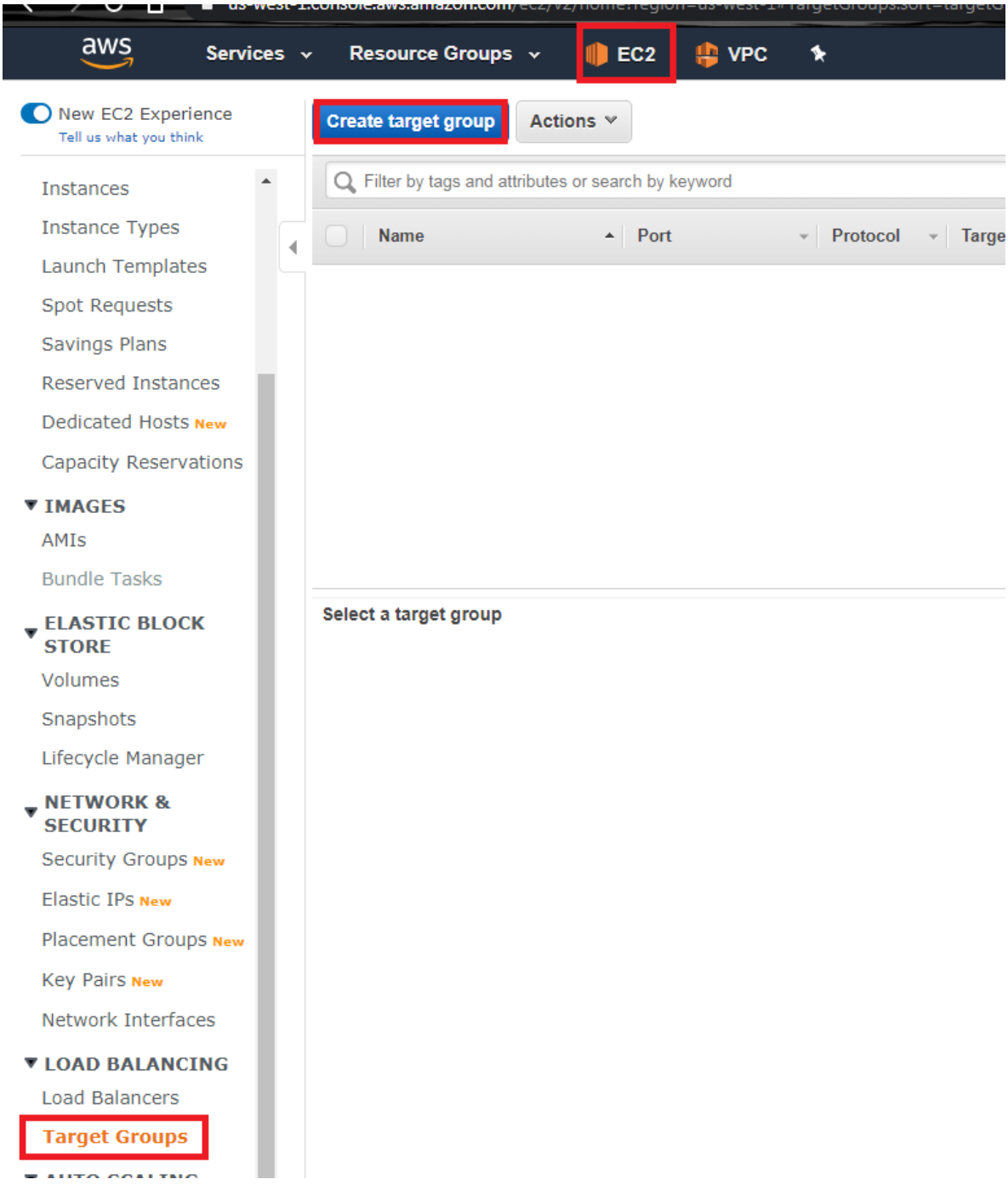
It's done on Lab5c1.

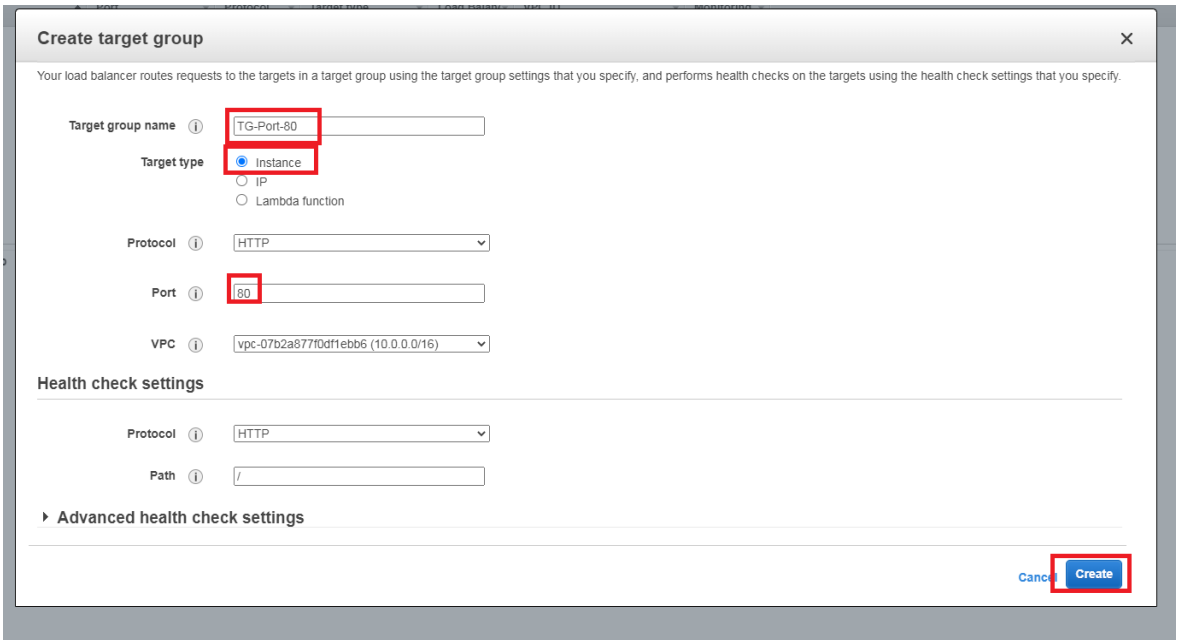
Create Security Group for ALB

In spite of having a security group for EC2, we have to create a security group for ALB. The theory was done on the course; however, it is better to create this specific case for EC2.

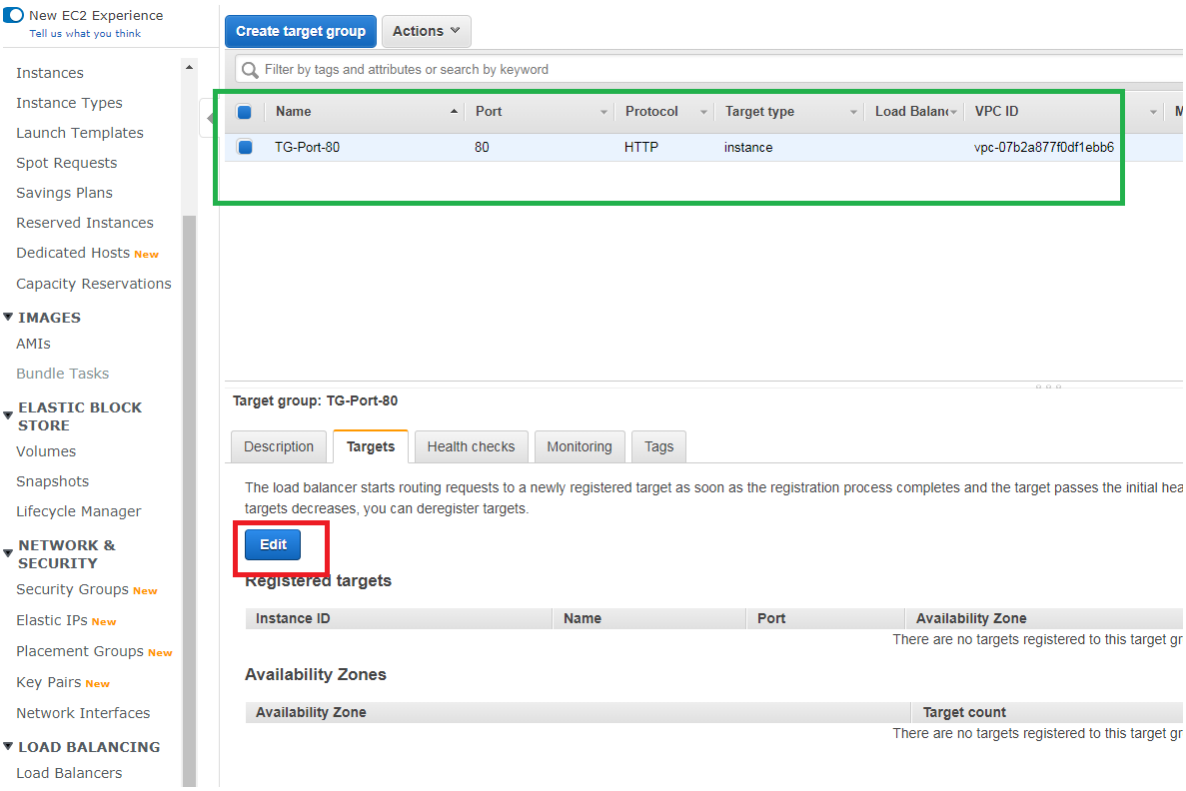


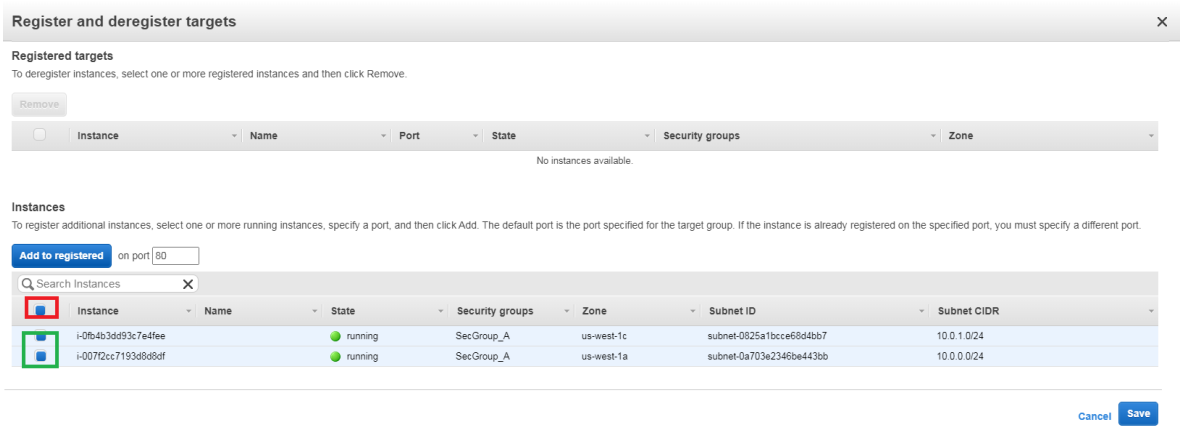
Create target groups

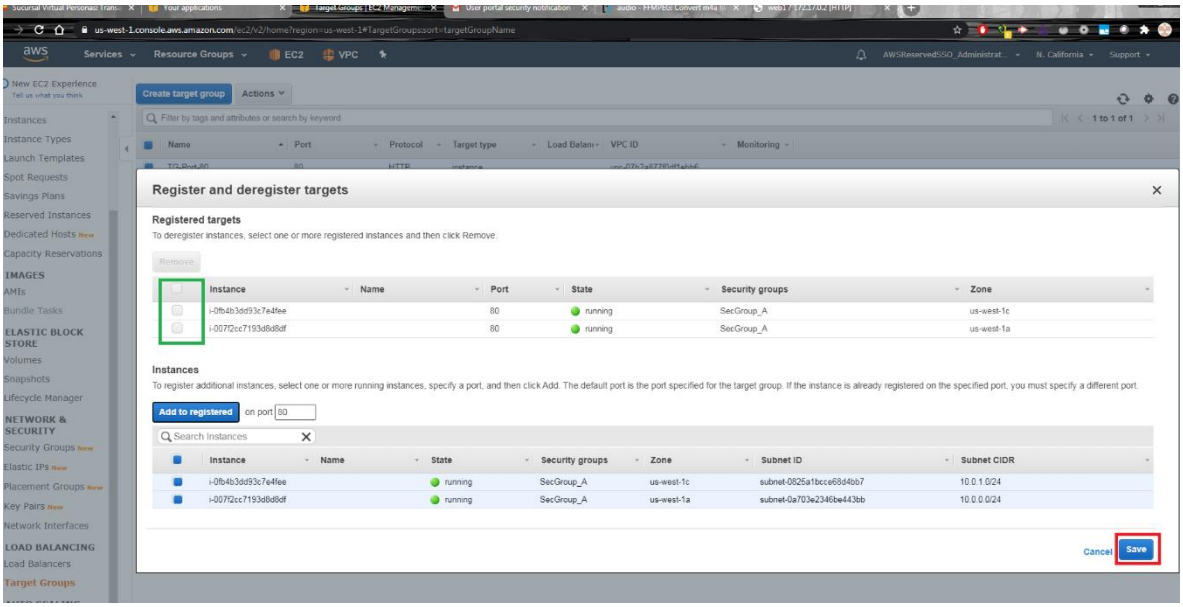

aws
Services
Resource Groups
EC2
VPC
New EC2 Experience
Tell us what you think
Create target group
Actions
Filter by tags and attributes or search by keyword
Name
Port
Protocol
Targe
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts New
Capacity Reservations
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
Lifecycle Manager
NETWORK & SECURITY
Security Groups New
Elastic IPs New
Placement Groups New
Key Pairs New
Network Interfaces
LOAD BALANCING
Load Balancers
Target Groups
Select a target group



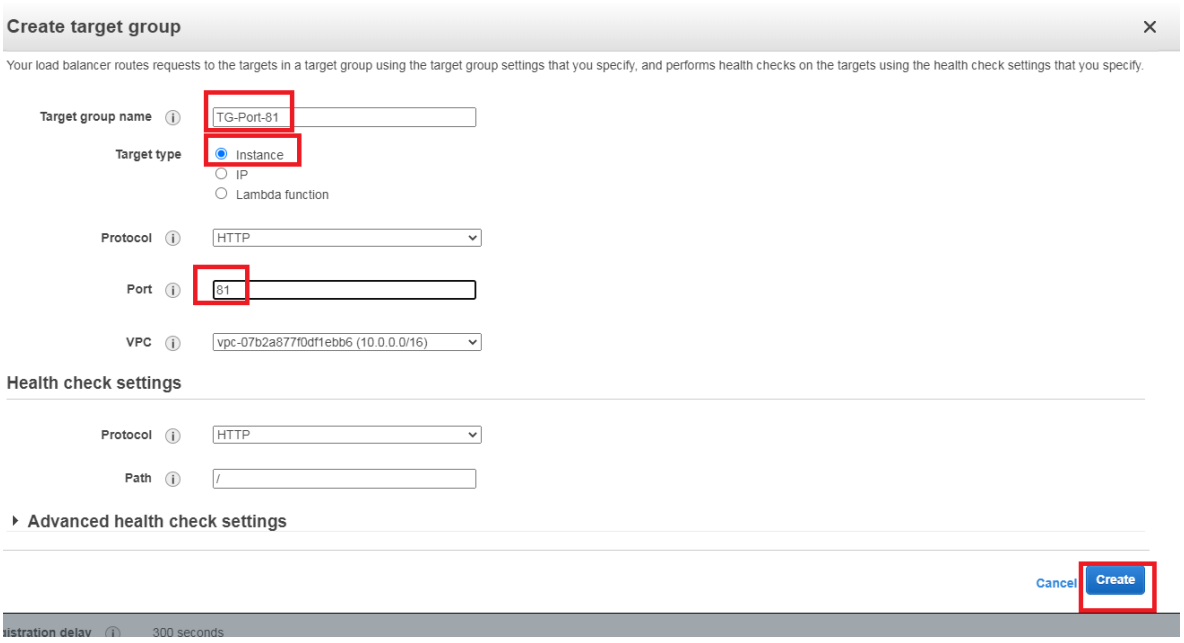
Now its time to register targets, it the same procedure for the remaining Target groups

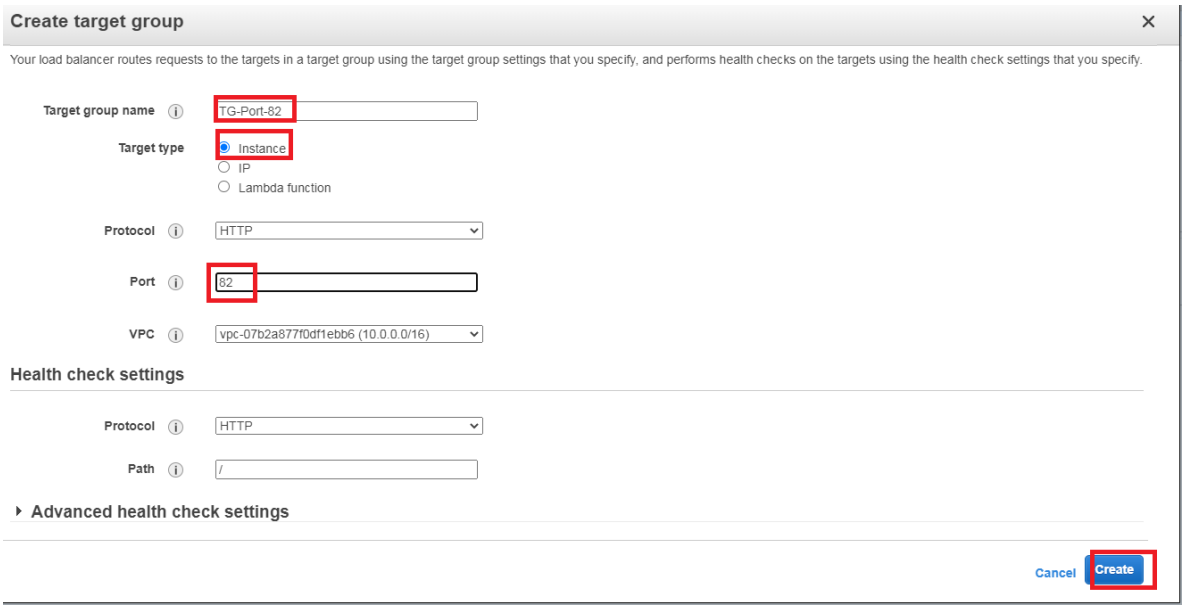




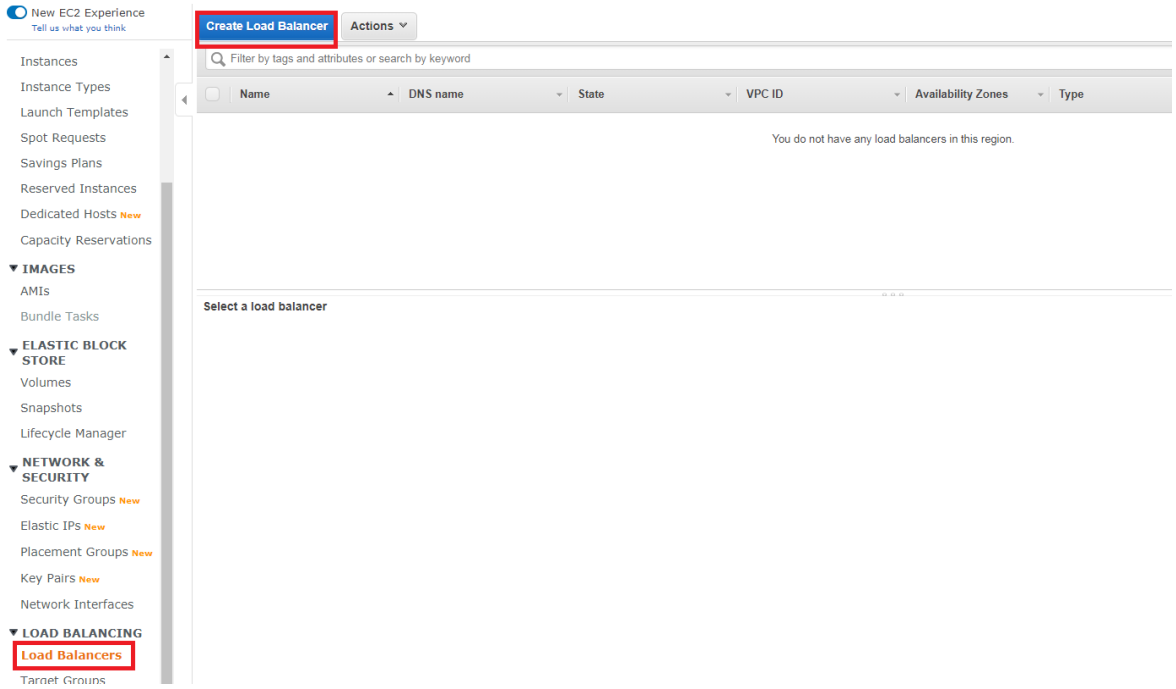


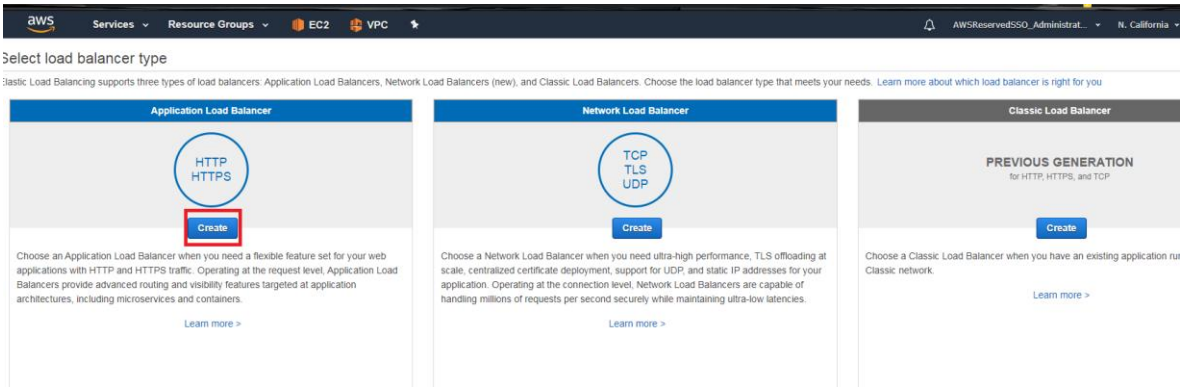
For the remaining TG, review the port and name,

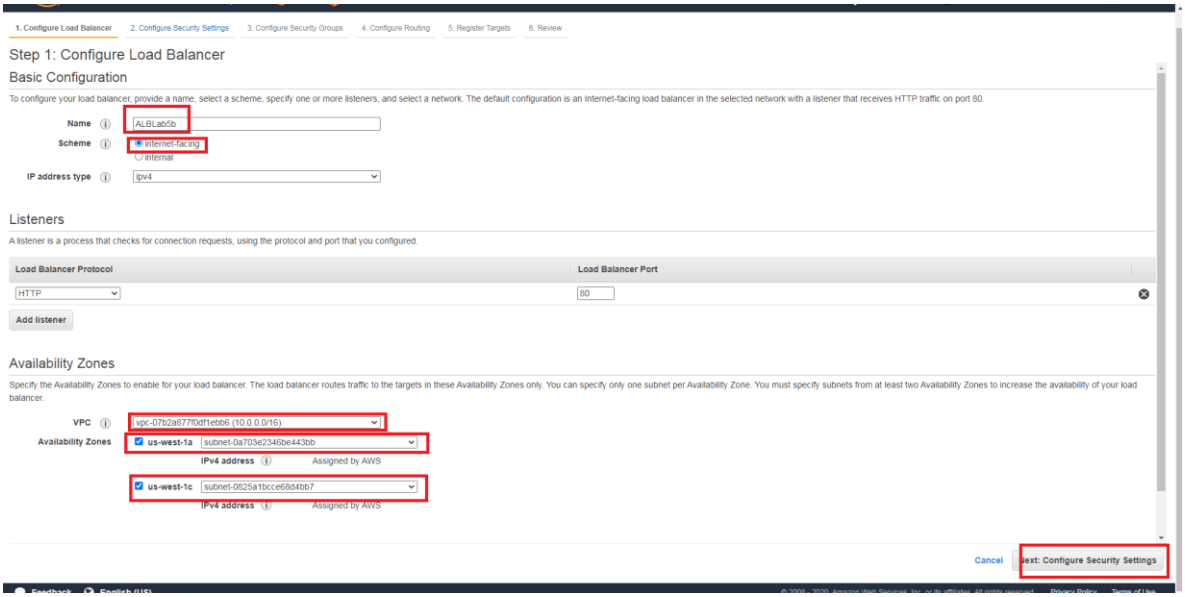

Create target group
Your load balancer routes requests to the targets in a target group using the target group settings that you specify, and performs health checks on the targets using the health check settings that you specify.
Target group name
TG-Port-81
Target type
Instance
IP
Lambda function
Protocol
HTTP
Port
81
VPC
vpc-07b2a877f0df1ebb6 (10.0.0.0/16)
Health check settings
Protocol
HTTP
Path
/
Advanced health check settings
Cancel
Create

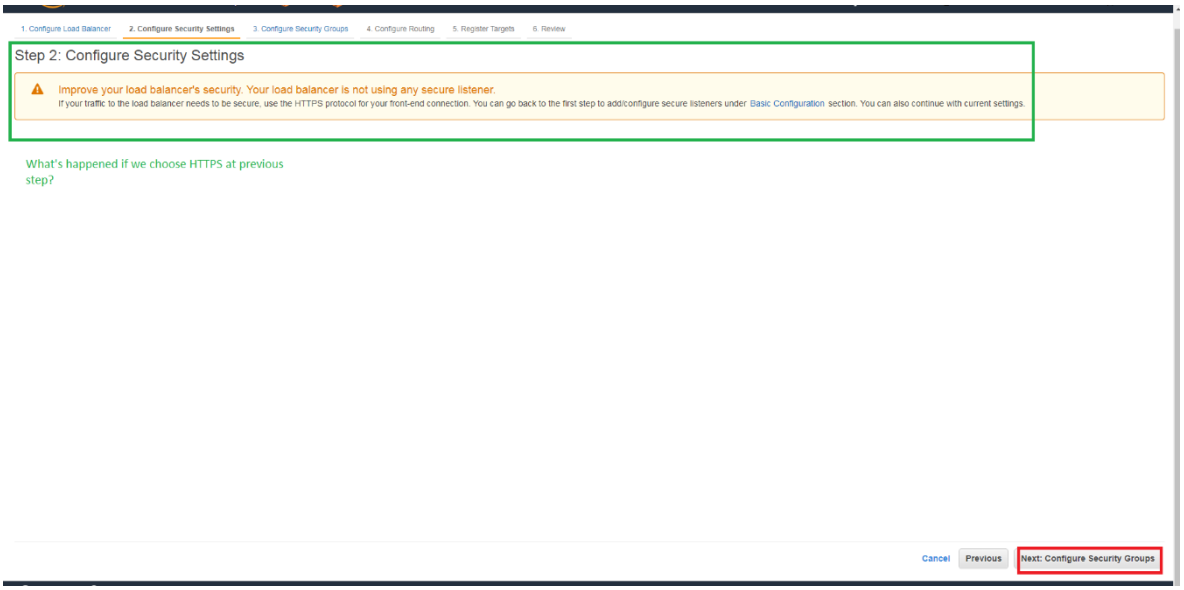

Create target group
Your load balancer routes requests to the targets in a target group using the target group settings that you specify, and performs health checks on the targets using the health check settings that you specify.
Target group name
TG-Port-82
Target type
Instance
IP
Lambda function
Protocol
HTTP
Port
82
VPC
vpc-07b2a877f0df1ebb6 (10.0.0.0/16)
Health check settings
Protocol
HTTP
Path
/
Advanced health check settings
Cancel
Create

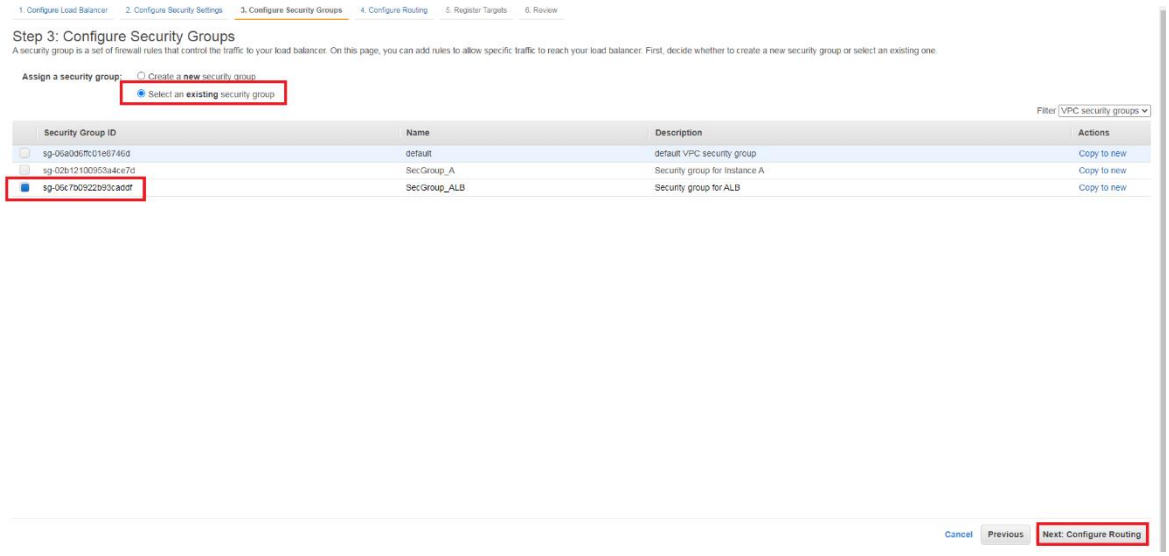
Create ALB

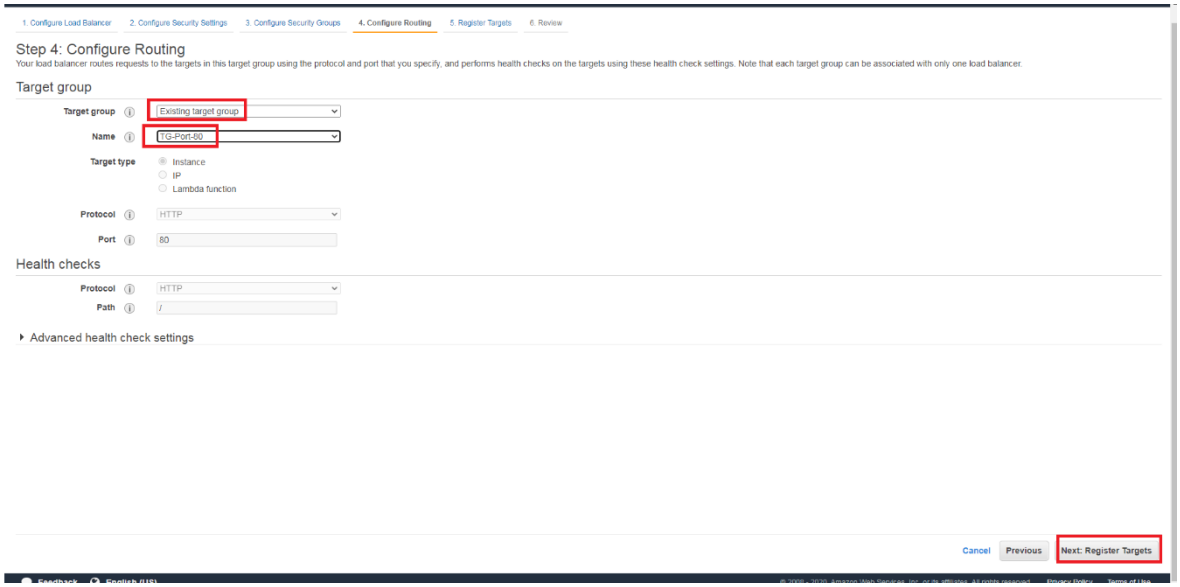

New EC2 Experience
Tell us what you think
Create Load Balancer
Actions
Filter by tags and attributes or search by keyword
Name
DNS name
State
VPC ID
Availability Zones
Type
You do not have any load balancers in this region.
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts New
Capacity Reservations
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
Lifecycle Manager
NETWORK & SECURITY
Security Groups New
Elastic IPs New
Placement Groups New
Key Pairs New
Network Interfaces
LOAD BALANCING
Load Balancers
Target Groups
Select a load balancer

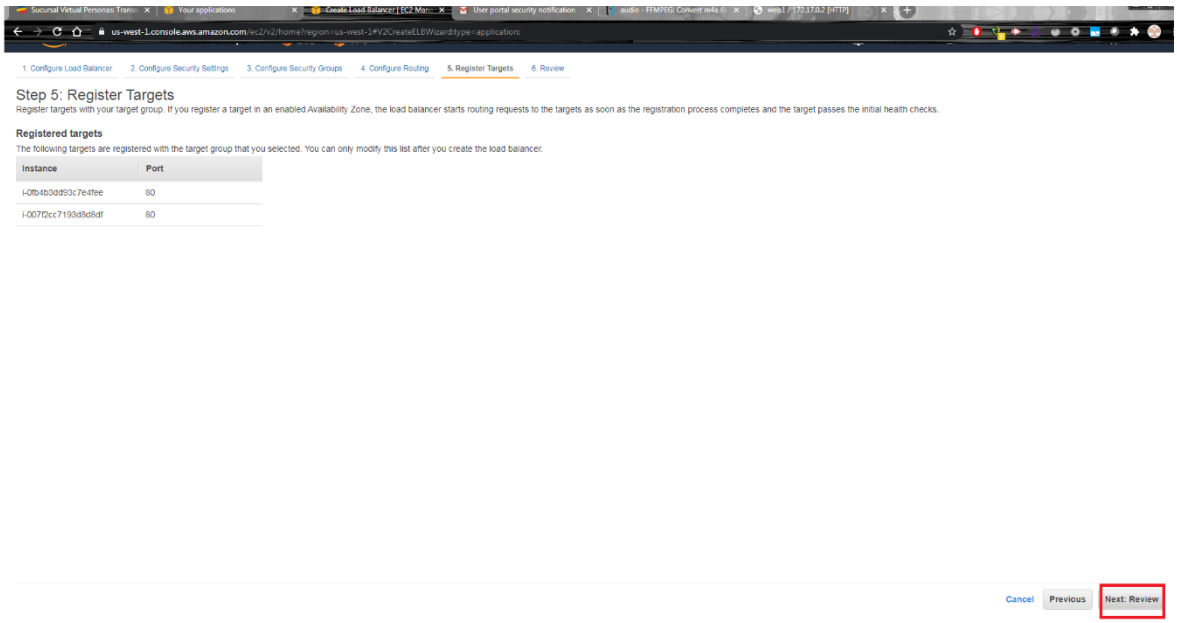

Services
Resource Groups
EC2
VPC
N. California
Select load balancer type
Application Load Balancer
HTTP
HTTPS
Create
Choose an Application Load Balancer when you need a flexible feature set for your web applications with HTTP and HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing and visibility features targeted at application architectures, including microservices and containers.
Learn more >
Network Load Balancer
TCP
TLS
UDP
Create
Choose a Network Load Balancer when you need ultra-high performance, TLS offloading at scale, centralized certificate deployment, support for UDP, and static IP addresses for your application. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second securely while maintaining ultra-low latencies.
Learn more >
Classic Load Balancer
PREVIOUS GENERATION
for HTTP, HTTPS, and TCP
Create
Learn more >

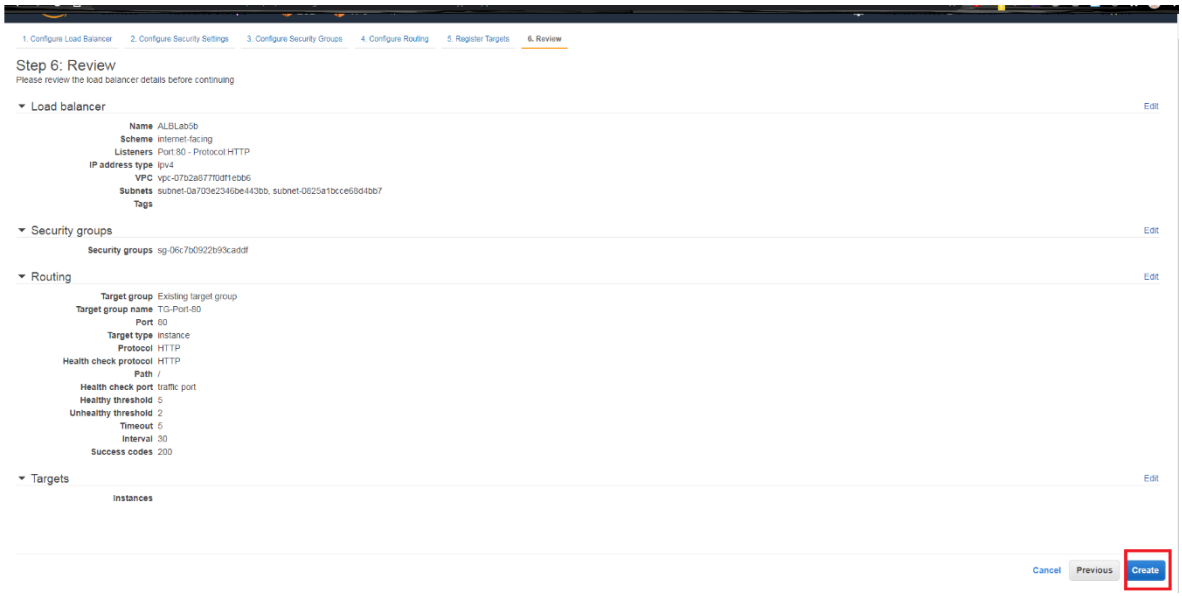

1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 1: Configure Load Balancer
Basic Configuration
Name
ALBLab5b
Scheme
internet-facing
internal
IP address type
ipv4
Listeners
Load Balancer Protocol
Load Balancer Port
HTTP
80
Add listener
Availability Zones
VPC
Availability Zones
us-west-1a
us-west-1c
IPv4 address
Assigned by AWS
Cancel
Next: Configure Security Settings

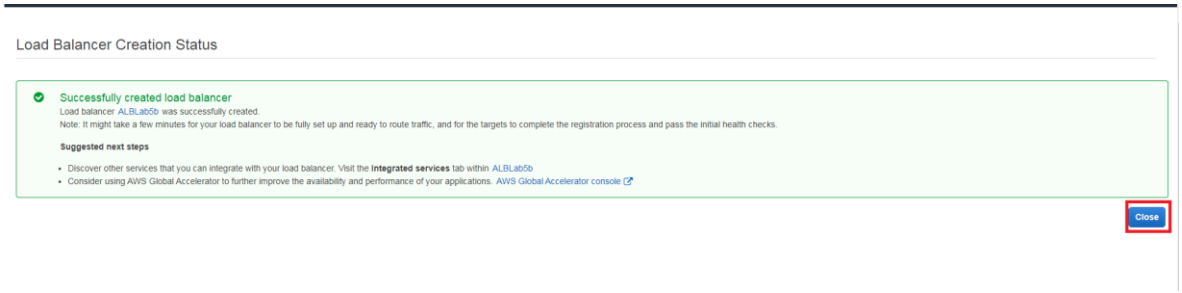

1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 2: Configure Security Settings
Improve your load balancer's security. Your load balancer is not using any secure listener.
If your traffic to the load balancer needs to be secure, use the HTTPS protocol for your front-end connection. You can go back to the first step to add/configure secure listeners under Basic Configuration section. You can also continue with current settings.
What's happened if we choose HTTPS at previous step?
Cancel
Previous
Next: Configure Security Groups

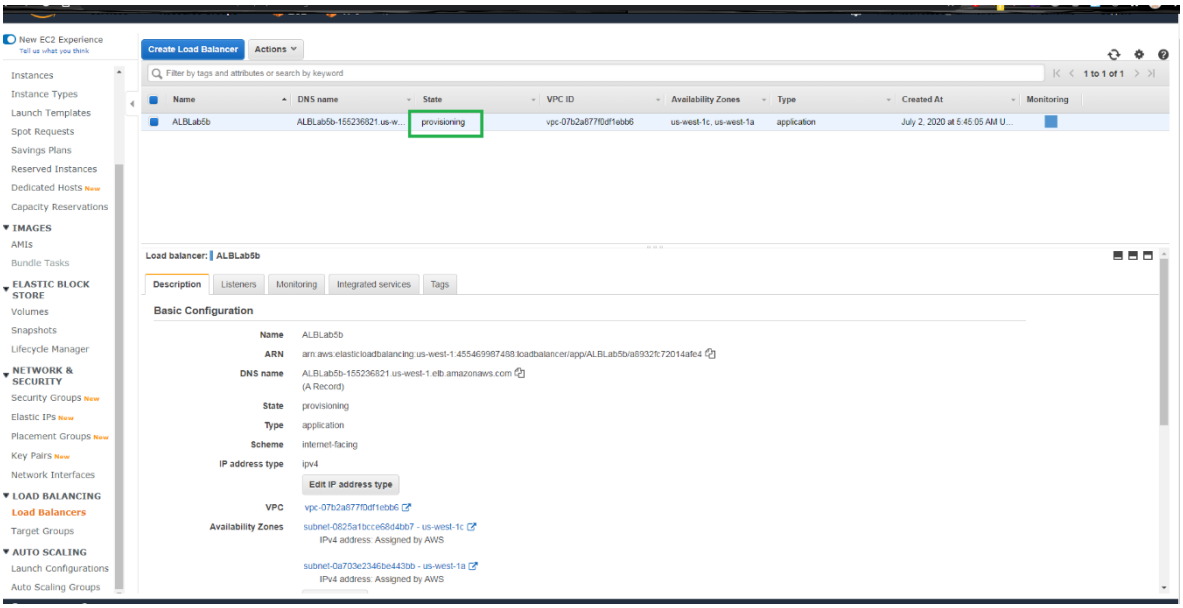

1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 3: Configure Security Groups
A security group is a set of firewall rules that control the traffic to your load balancer. On this page, you can add rules to allow 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-06a0d6ffc01e6746d
default
default VPC security group
Copy to new
sg-02b12100953a4ce7d
SecGroup_A
Security group for Instance A
Copy to new
sg-06c7b0922b93caddf
SecGroup_ALB
Security group for ALB
Copy to new
Cancel
Previous
Next: Configure Routing

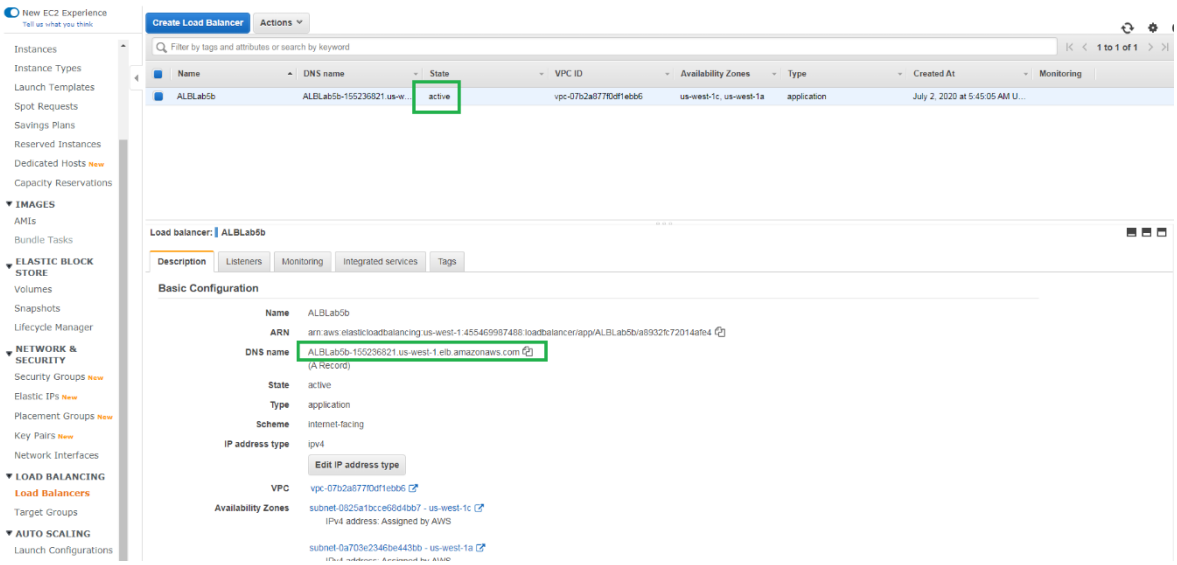

1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 4: Configure Routing
Your load balancer routes requests to the targets in this target group using the protocol and port that you specify, and performs health 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


us-west-1.console.aws.amazon.com/ec2/v2/home?region=us-west-1#V2CreateELBWizard:type=application:
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-0fb4b3dd93c7e4fee
80
i-007f2cc7193d8d8df
80
Cancel
Previous
Next: Review

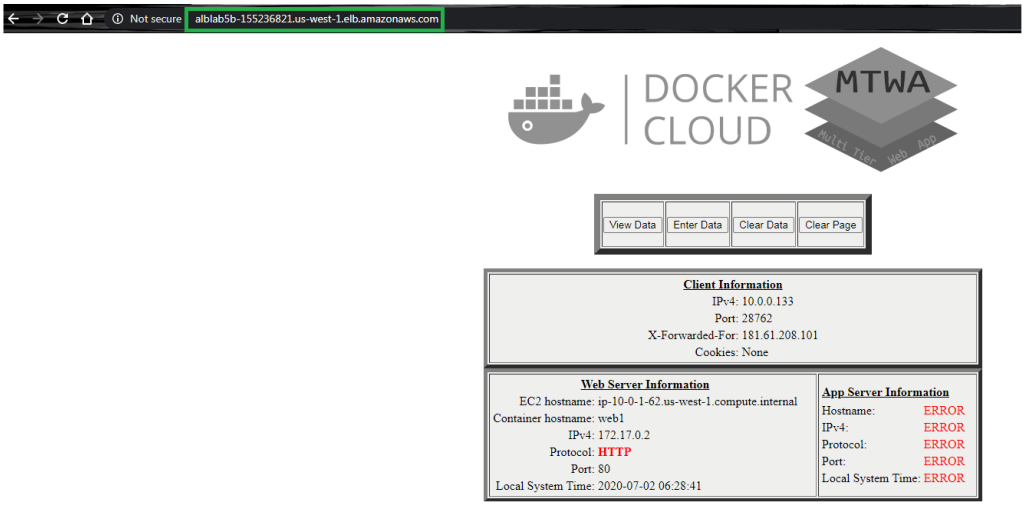

1. Configure Load Balancer
2. Configure Security Settings
3. Configure Security Groups
4. Configure Routing
5. Register Targets
6. Review
Step 6: Review
Please review the load balancer details before continuing
Load balancer
Edit
Name ALBLab5b
Scheme internet-facing
Listeners Port:80 - Protocol:HTTP
IP address type ipv4
VPC vpc-07b2a877f0df1ebb6
Subnets subnet-0a703e2346be443bb, subnet-0825a1bcce68d4bb7
Tags
Security groups
Edit
Security groups sg-06c7b0922b93caddf
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


Load Balancer Creation Status
Successfully created load balancer
Load balancer ALBLab5b was successfully created.
Note: It might take a few minutes for your load balancer to be fully set up and ready to route traffic, and for the targets to complete the registration process and pass the initial health checks.
Suggested next steps
Discover other services that you can integrate with your load balancer. Visit the integrated services tab within ALBLab5b
Consider using AWS Global Accelerator to further improve the availability and performance of your applications. AWS Global Accelerator console
Close

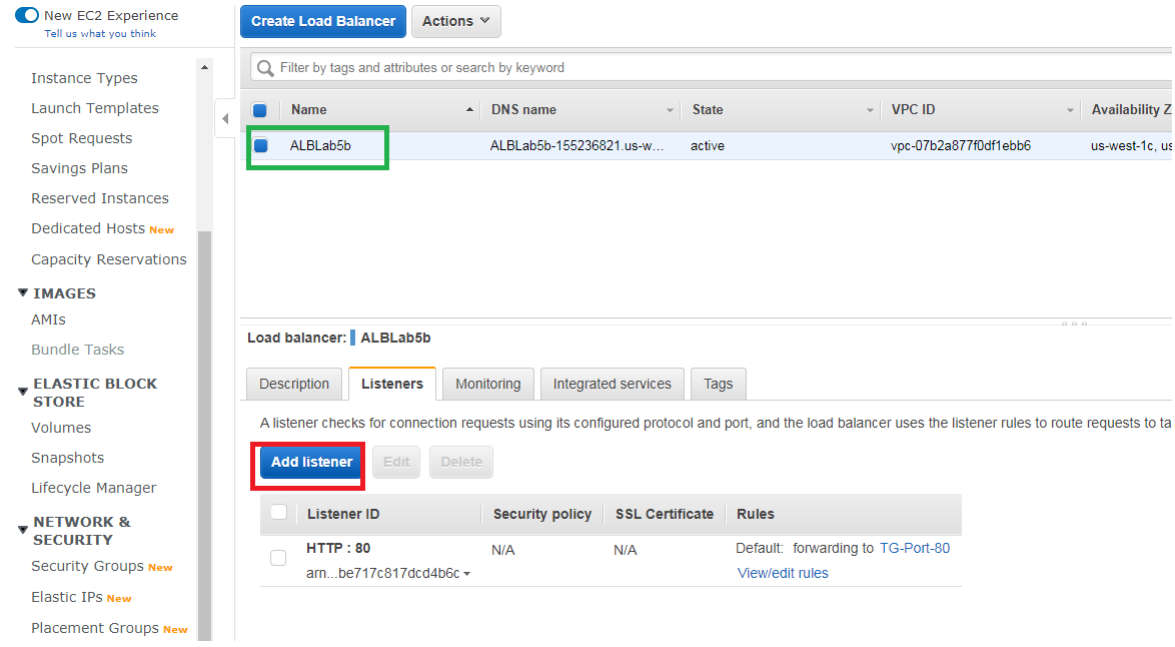

New EC2 Experience
Tell us what you think
Create Load Balancer
Actions
Filter by tags and attributes or search by keyword
1 to 1 of 1
Name
DNS name
State
VPC ID
Availability Zones
Type
Created At
Monitoring
ALBLab5b
ALBLab5b-155236821.us-w...
provisioning
vpc-07b2a877f0df1ebb6
us-west-1c, us-west-1a
application
July 2, 2020 at 5:45:05 AM U...
Load balancer: ALBLab5b
Description
Listeners
Monitoring
Integrated services
Tags
Basic Configuration
Name ALBLab5b
ARN arn:aws:elasticloadbalancing:us-west-1:455469987488:loadbalancer/app/ALBLab5b/a8932fc72014afe4
DNS name ALBLab5b-155236821.us-west-1.elb.amazonaws.com
(A Record)
State provisioning
Type application
Scheme internet-facing
IP address type ipv4
Edit IP address type
VPC vpc-07b2a877f0df1ebb6
Availability Zones subnet-0825a1bcce68d4bb7 - us-west-1c
IPv4 address: Assigned by AWS
subnet-0a703e2346be443bb - us-west-1a
IPv4 address: Assigned by AWS
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts New
Capacity Reservations
IMAGES
AMIs
Bundle Tasks
ELASTIC BLOCK STORE
Volumes
Snapshots
Lifecycle Manager
NETWORK & SECURITY
Security Groups New
Elastic IPs New
Placement Groups New
Key Pairs New
Network Interfaces
LOAD BALANCING
Load Balancers
Target Groups
AUTO SCALING
Launch Configurations
Auto Scaling Groups

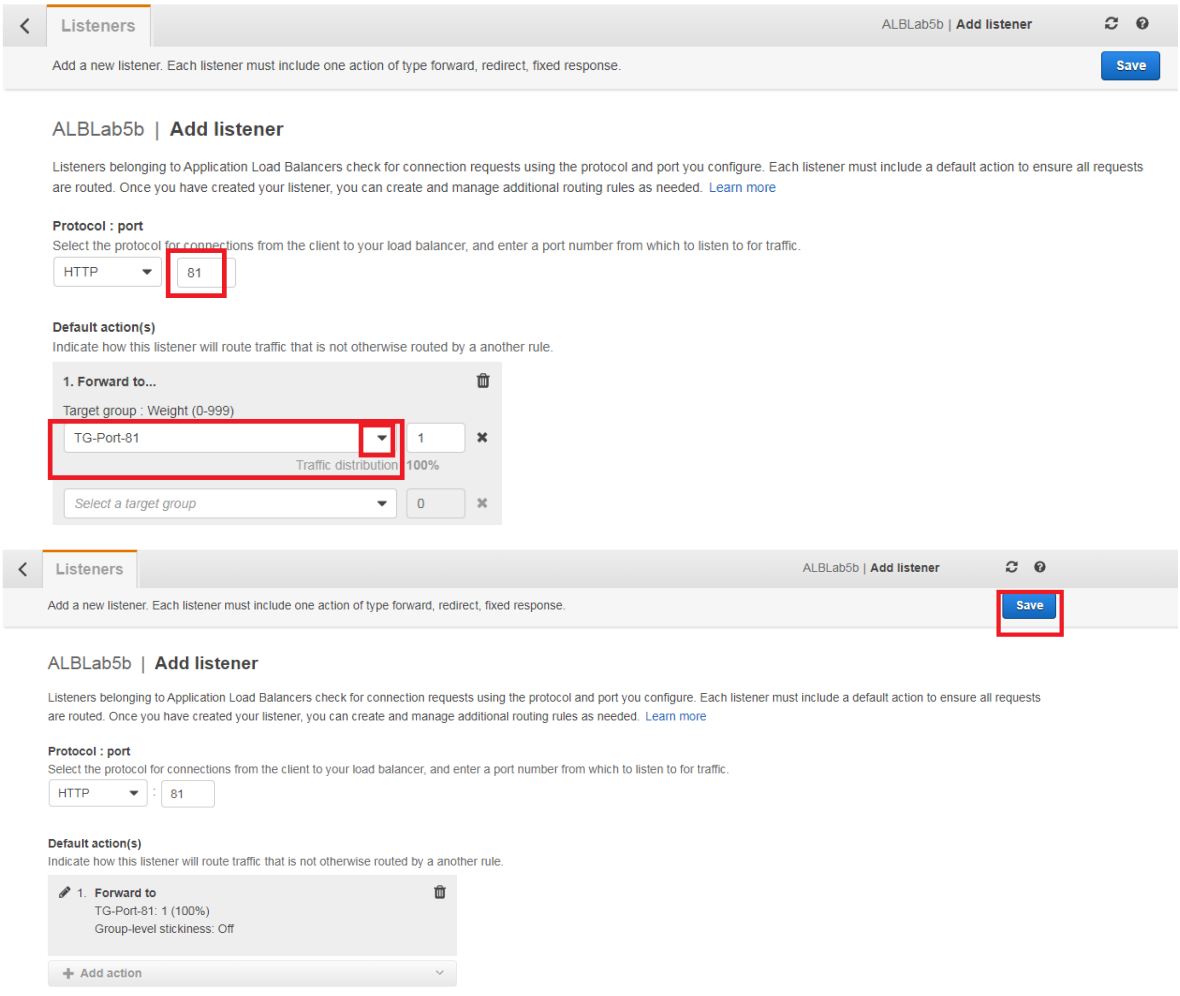


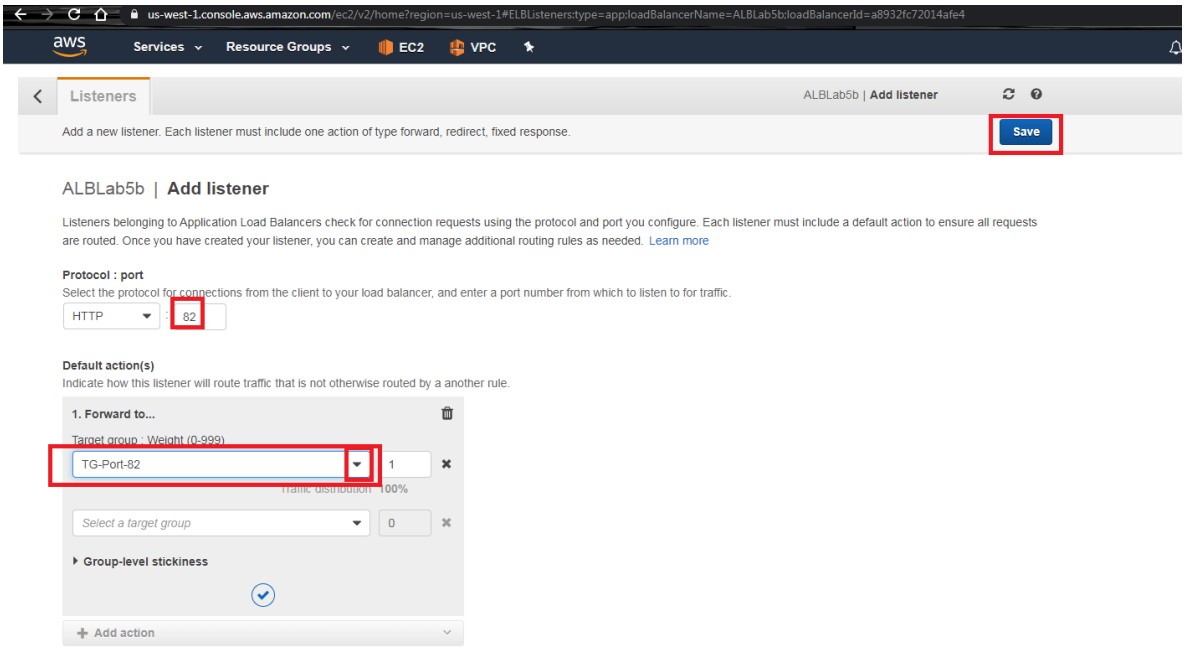
Test it with the DNS name of the balancer,

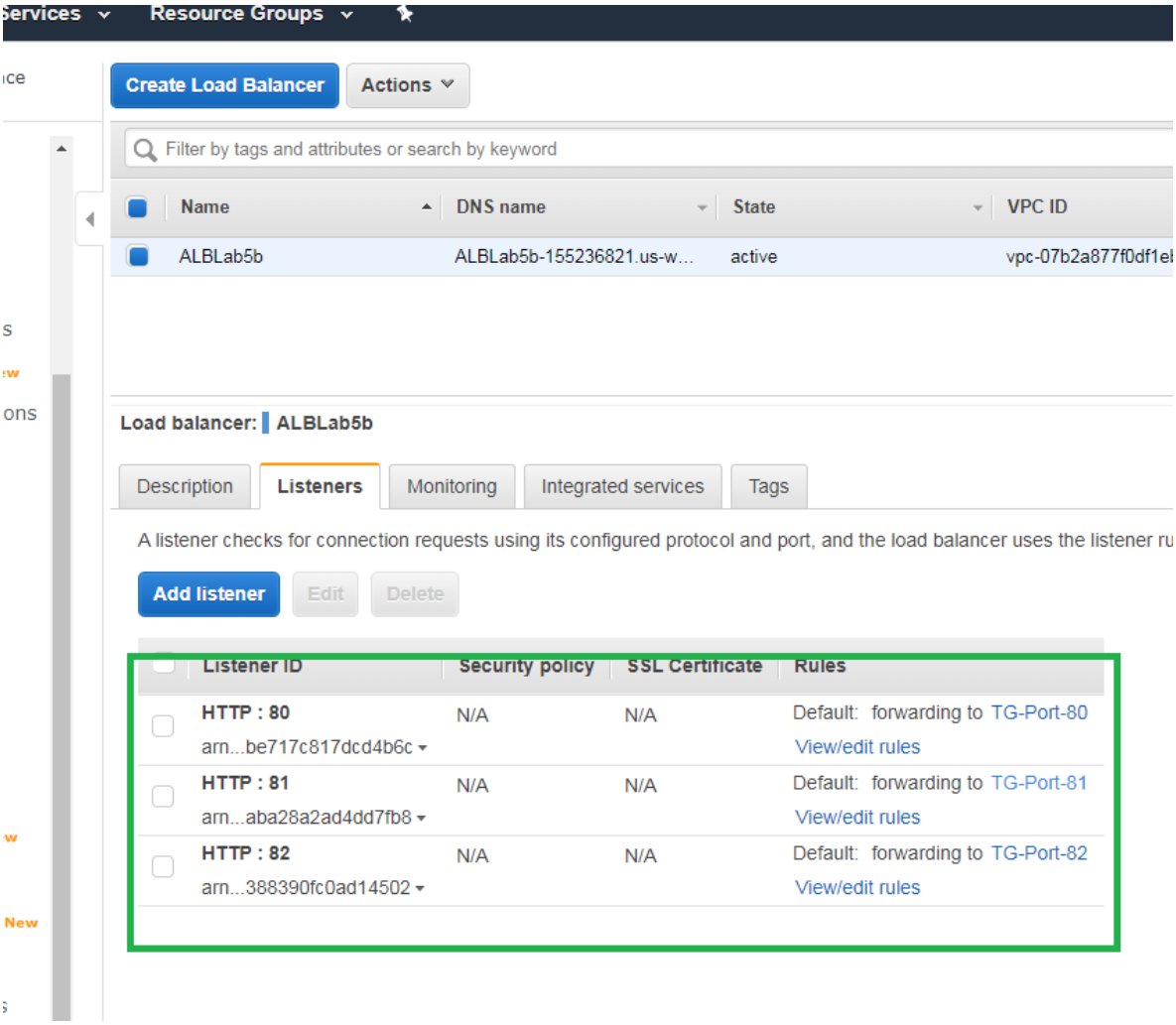


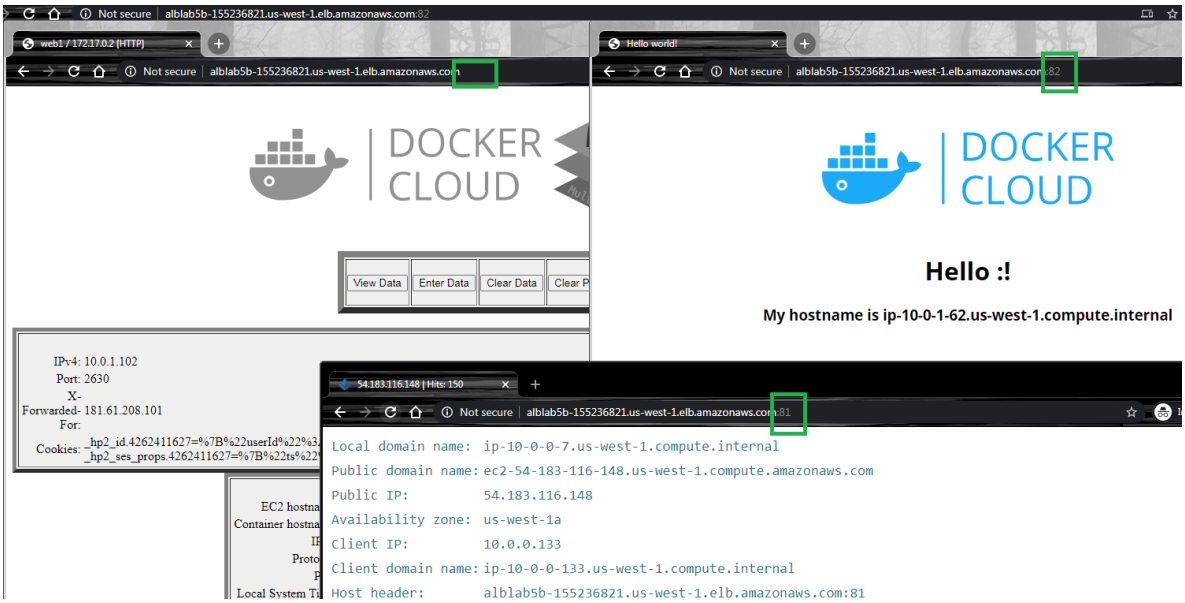
Create Additional Listeners for Ports



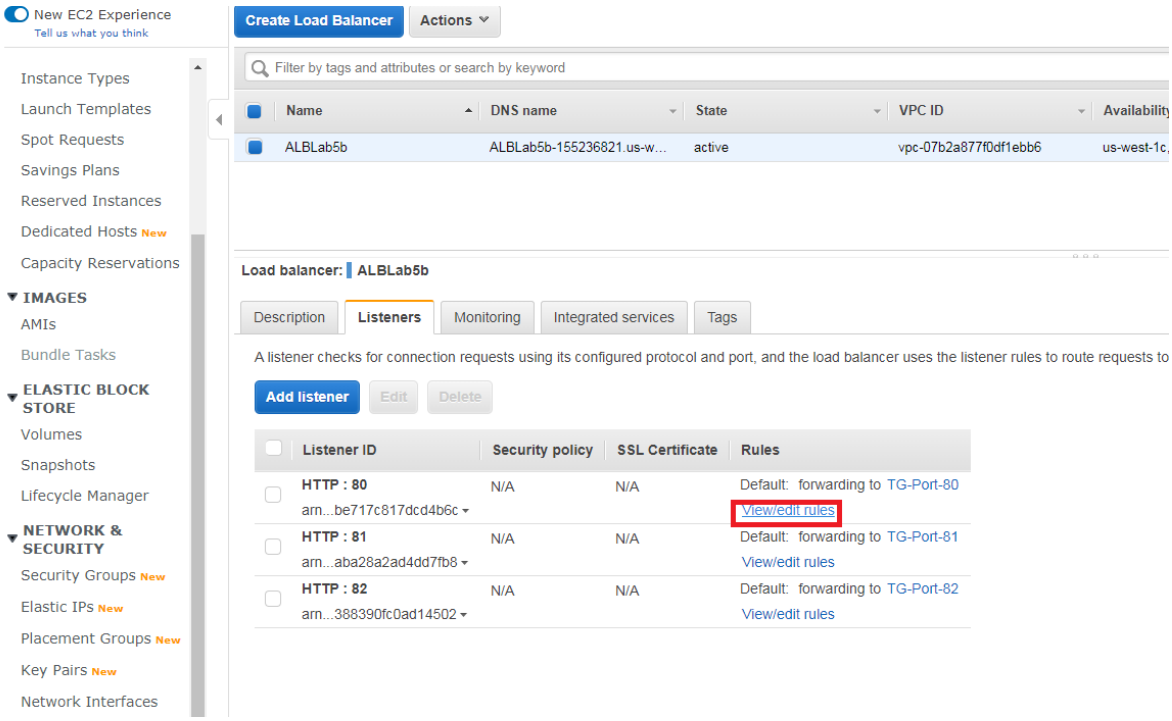


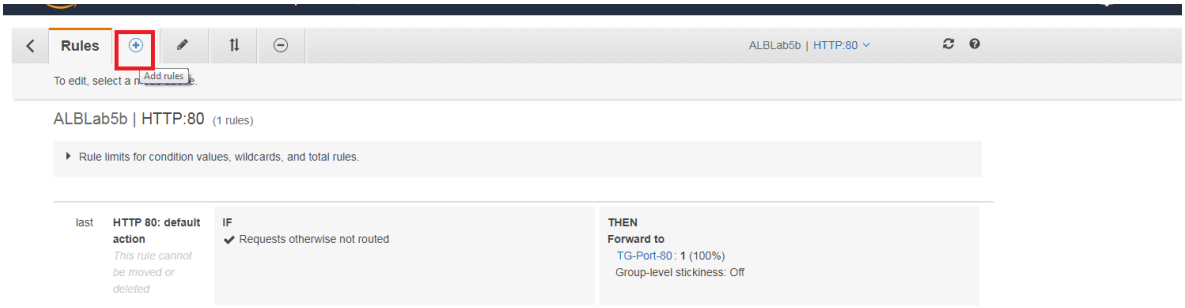

us-west-1.console.aws.amazon.com/ec2/v2/home?region=us-west-1#ELBListeners:type=app;loadBalancerName=ALBLab5b;loadBalancerId=a8932fc72014afe4
aws
Services
Resource Groups
EC2
VPC
Listeners
ALBLab5b | Add listener
Add a new listener. Each listener must include one action of type forward, redirect, fixed response.
Save
ALBLab5b | Add listener
Listeners belonging to Application Load Balancers check for connection requests using the protocol and port you configure. Each listener must include a default action to ensure all requests are routed. Once you have created your listener, you can create and manage additional routing rules as needed. Learn more
Protocol : port
Select the protocol for connections from the client to your load balancer, and enter a port number from which to listen to for traffic.
HTTP
82
Default action(s)
Indicate how this listener will route traffic that is not otherwise routed by a another rule.
1. Forward to...
Target group
Weight (0-999)
TG-Port-82
1
Traffic distribution 100%
Select a target group
0
Group-level stickiness
Add action

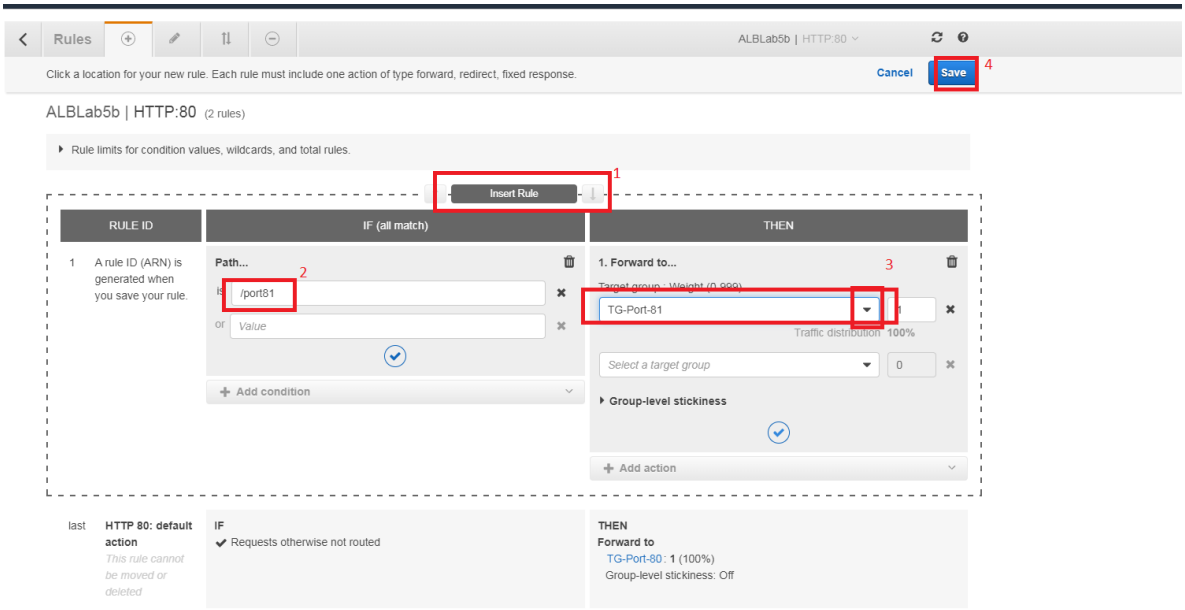

Services
Resource Groups
Create Load Balancer
Actions
Filter by tags and attributes or search by keyword
Name
DNS name
State
VPC ID
ALBLab5b
ALBLab5b-155236821.us-w...
active
vpc-07b2a877f0df1e
Load balancer: ALBLab5b
Description
Listeners
Monitoring
Integrated services
Tags
A listener checks for connection requests using its configured protocol and port, and the load balancer uses the listener ru
Add listener
Edit
Delete
Listener ID
Security policy
SSL Certificate
Rules
HTTP : 80
arn...be717c817dcd4b6c
N/A
N/A
Default: forwarding to TG-Port-80
View/edit rules
HTTP : 81
arn...aba28a2ad4dd7fb8
N/A
N/A
Default: forwarding to TG-Port-81
View/edit rules
HTTP : 82
arn...388390fc0ad14502
N/A
N/A
Default: forwarding to TG-Port-82
View/edit rules

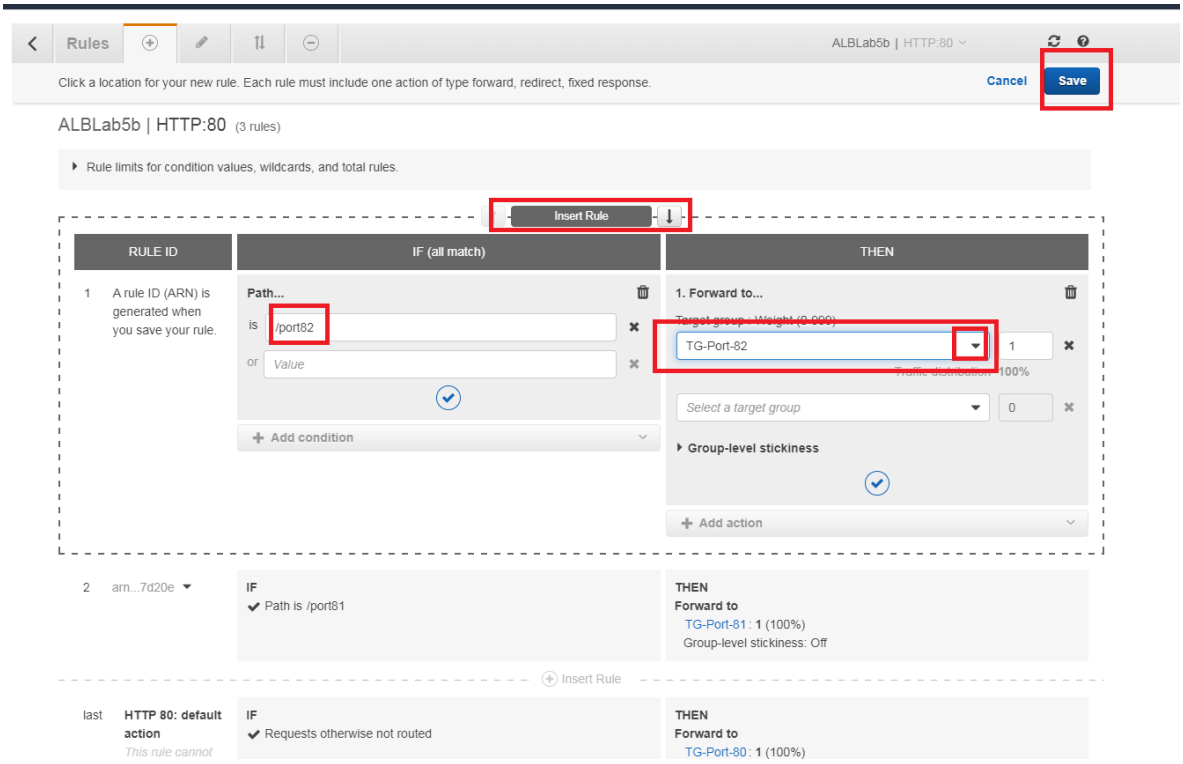


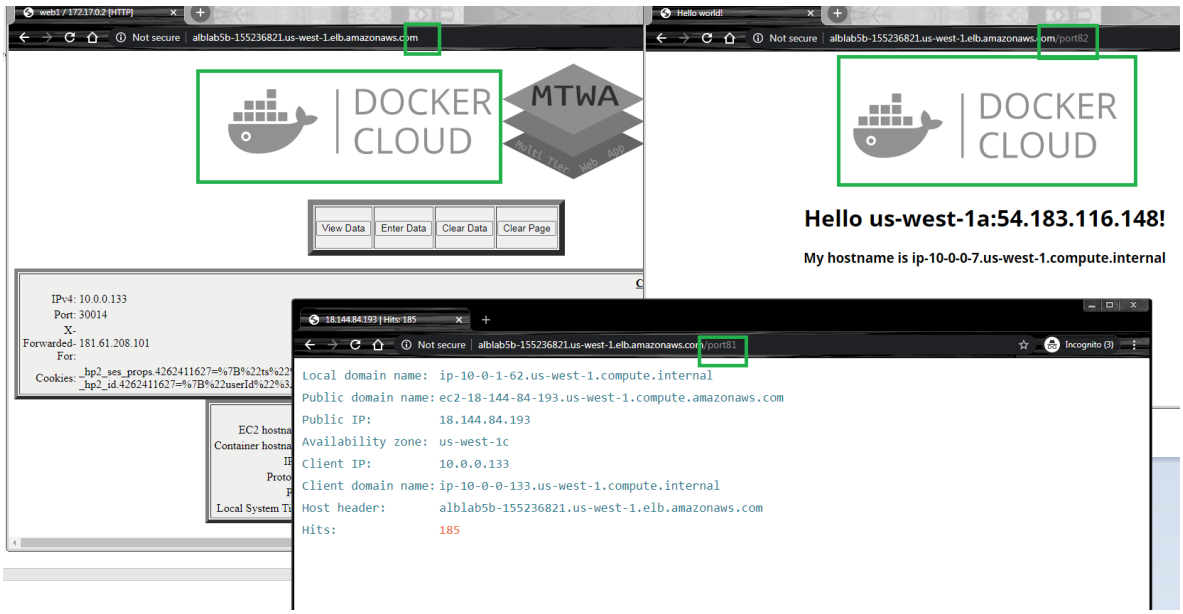
Create Listener Rules for Routing Paths




Rules
Add rules
ALBLab5b | HTTP:80
To edit, select a rule.
ALBLab5b | HTTP:80 (1 rules)
Rule limits for condition values, wildcards, and total rules.
last
HTTP 80: default action
This rule cannot be moved or deleted
IF
Requests otherwise not routed
THEN
Forward to
TG-Port-80 : 1 (100%)
Group-level stickiness: Off


Rules
ALBLab5b | HTTP:80
Click a location for your new rule. Each rule must include one action of type forward, redirect, fixed response.
Cancel
Save
4
ALBLab5b | HTTP:80 (2 rules)
Rule limits for condition values, wildcards, and total rules.
Insert Rule
1
RULE ID
IF (all match)
THEN
1
A rule ID (ARN) is generated when you save your rule.
Path...
2
/port81
or
Value
Add condition
1. Forward to...
3
TG-Port-81
Traffic distribution 100%
Select a target group
0
Group-level stickiness
Add action
last
HTTP 80: default action
This rule cannot be moved or deleted
IF
Requests otherwise not routed
THEN
Forward to
TG-Port-80 : 1 (100%)
Group-level stickiness: Off





Lab 5B using Command Line (Windows)

Create Network Infrastructure, Instances and its security groups

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

rem Crear la 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}"

rem Crear subred Publica 1
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
rem Permitir que las instancias que se ejecutan en la subred se hagan public
as
aws ec2 modify-subnet-attribute --subnet-id %pbsn1_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 la subred
aws ec2 associate-route-table  --subnet-id %pbsn1_Id% --route-table-
id %Public_RT_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 Lab5b --query "KeyMaterial" --
output text > Lab5b.pem
winscp.com /keygen "Lab5b.pem" /output="Lab5b.ppk"
```

```
rem Crear los Security Groups para esa instancia
aws ec2 create-security-group --group-name "SecGroup_A" --
description "Security group for Instance A" --vpc-
id %vpcn_Id% |jq ".GroupId">tmpFile
set /p SecGroup_A_Id= < tmpFile
aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --
protocol tcp --port 22 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --
protocol tcp --port 80 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --
protocol tcp --port 81 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --
protocol tcp --port 82 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --
protocol tcp --port 443 --cidr 0.0.0.0/0


rem Crear subred Publica 2, ponerla public
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
aws ec2 modify-subnet-attribute --subnet-id %pbsn2_Id% --map-public-ip-on-
launch
aws ec2 associate-route-table  --subnet-id %pbsn2_Id% --route-table-
id %Public_RT_Id%

rem En el laboratorio de EC2 Inicial se mostrar la importancia de buscar una
 AMI correcto.
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

rem Se solicitan instancias y se adiciona un bootstrap para comprobar que el
 docker fue instalado
aws ec2 run-instances --image-id %AMI% --count 1 --instance-
type %instance_type% --key-name Lab5b --security-group-ids %SecGroup_A_Id% -
-subnet-id %pbsn1_Id% --tag-
specifications "ResourceType=instance,Tags=[{Key=ServerName,Value=A}]"  --
user-
```

```
data file://bootstrap.sh |jq "[.Instances|.[].InstanceId|.]"|jq ".[0]" >tmpF
ile
set /p Instance1Id= <tmpFile
aws ec2 run-instances --image-id %AMI% --count 1 --instance-
type %instance_type% --key-name Lab5b --security-group-ids %SecGroup_A_Id% -
-subnet-id %pbsn2_Id% --tag-
specifications "ResourceType=instance,Tags=[{Key=ServerName,Value=B}]"  --
user-
data file://bootstrap.sh |jq "[.Instances|.[].InstanceId|.]"|jq ".[0]" >tmpF
ile
set /p Instance2Id= <tmpFile


rem Traer Datos especificos de instancia A y B; y setearlos a las variables
A_IP y B_IP
```

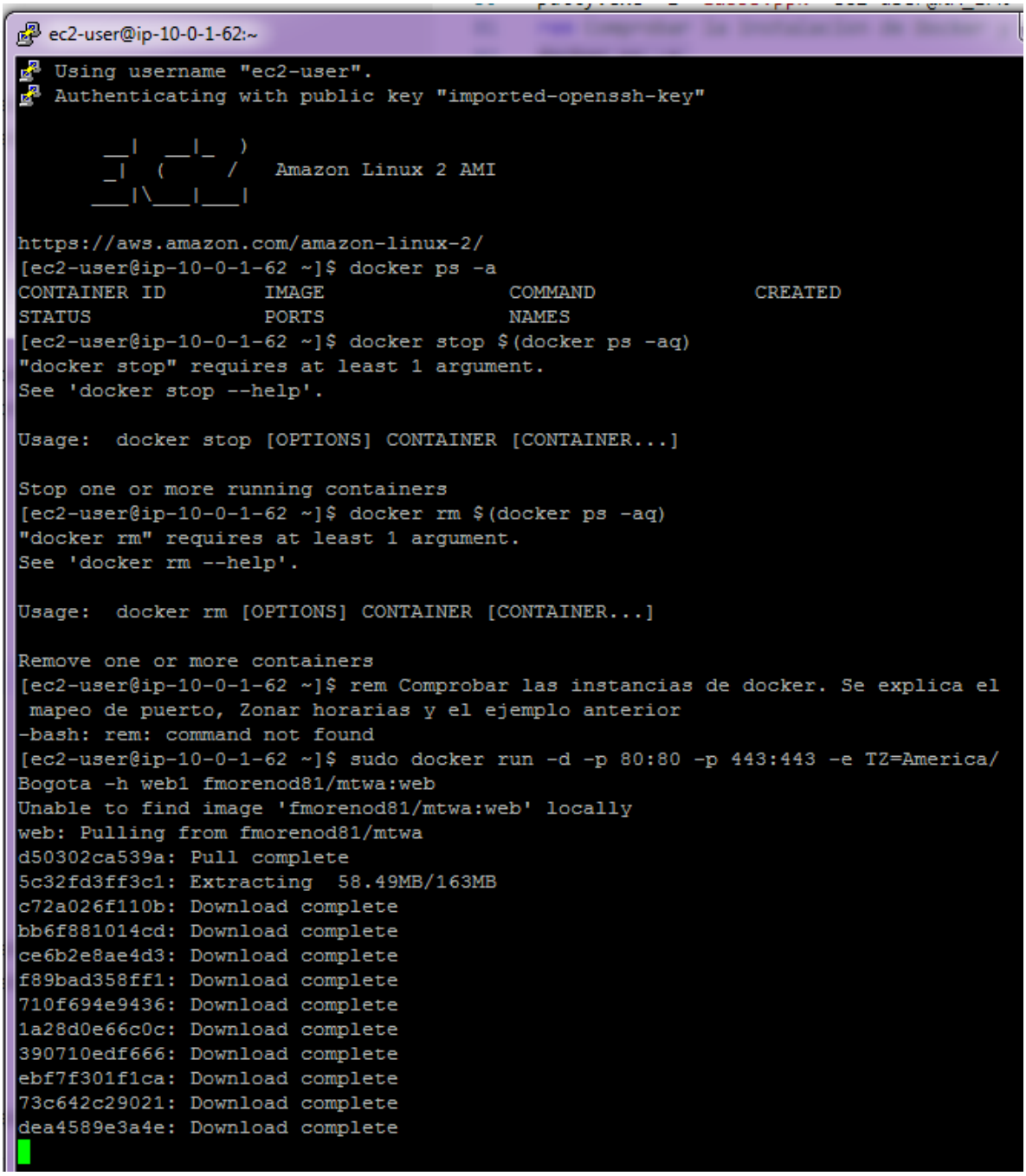
```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set pbsn1_Mask="10.0.0.0/24"

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set pbsn2_Mask="10.0.1.0/24"

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set first_az="us-west-1a"

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set second_az="us-west-1c"

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set instance_type="t3.small"

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 create-vpc --cidr-block %vpcn_Mask%|jq ".Vpc.VpcId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p vpcn_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 modify-vpc-attribute --vpc-id %vpcn_Id% --enable-dns-hostnames "{\"Value\":true}"


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 create-subnet --vpc-id %vpcn_Id% --cidr-block %pbsn1_Mask% --availability-zone %first_az%|jq ".Subnet.SubnetId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p pbsn1_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 modify-subnet-attribute --subnet-id %pbsn1_Id% --map-public-ip-on-launch


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 create-internet-gateway|jq ".InternetGateway.InternetGatewayId"  >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p IGW_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 attach-internet-gateway --vpc-id %vpcn_Id% --internet-gateway-id %IGW_Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 create-route-table --vpc-id %vpcn_Id%|jq ".RouteTable.RouteTableId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p Public_RT_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>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\s5c1\CLI>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 associate-route-table  --subnet-id %pbsn1_Id% --route-table-id %Public_RT_Id%
{
    "AssociationId": "rtbassoc-09779c35fd368a96c",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 create-key-pair --key-name Lab5b --query "KeyMaterial" --output text > Lab5b.pem

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>winscp.com /keygen "Lab5b.pem" /output="Lab5b.ppk"
Key saved to "Lab5b.ppk".
```

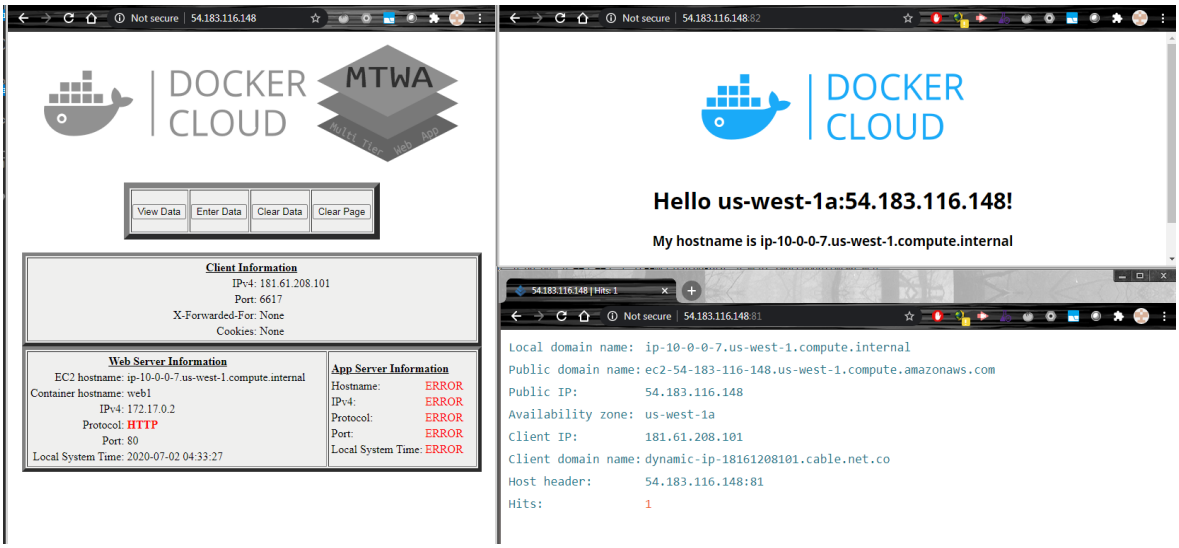
```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 create-security-group --group-name "SecGroup_A" --description "Security group for Instance A" --vpc-id %vpcn_Id% |jq ".GroupId">tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p SecGroup_A_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --protocol tcp --port 22 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --protocol tcp --port 80 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --protocol tcp --port 81 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --protocol tcp --port 82 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 authorize-security-group-ingress --group-id %SecGroup_A_Id% --protocol tcp --port 443 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>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\s5c1\CLI>set /p pbsn2_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 modify-subnet-attribute --subnet-id %pbsn2_Id% --map-public-ip-on-launch


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 associate-route-table  --subnet-id %pbsn2_Id% --route-table-id %Public_RT_Id%
{
    "AssociationId": "rtbassoc-02a09db5038448056",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 describe-images --owners amazon --filters "Name=name,Values=amzn2-ami-hvm-2.0.????????.?-x86_64-gp2" "Name=state,Values=available" --que
s, &CreationDate))[:1].ImageId" --output text >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p AMI= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws ec2 run-instances --image-id %AMI% --count 1 --instance-type %instance_type% --key-name Lab5b --security-group-ids %SecGroup_A_Id% --subnet-
fications "ResourceType=instance,Tags=[{Key=ServerName,Value=A}]"  --user-data file://bootstrap.sh |jq "[.Instances|.[].InstanceId|.]"|jq ".[0]" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p Instance1Id= <tmpFile
```

Running the code on SSH

Use the first putty and run docker; finally check on browser that containers are running

```
rem Ingresar a ambas instancias publica por SSH. Ejecutar las mismas accione
s y despues ir al navegador a ver que funcionan las IPs
putty.exe -i "Lab5b.ppk" ec2-user@%A_IP%
rem Comprobar la instalacion de Docker y borramos cualquier contenedor anter
ior
docker ps -a
docker stop $(docker ps -aq)
docker rm $(docker ps -aq)
rem Comprobar las instancias de docker. Se explica el mapeo de puerto, Zonar
 horarias y el ejemplo anterior
sudo docker run -d -p 80:80 -p 443:443 -e TZ=America/Bogota -
h web1 fmorenod81/mtwa:web
sudo docker run -d -p 81:80 -h web2 benpiper/r53-ec2-web
export AZ=$(curl -s http://169.254.169.254/latest/meta-
data/placement/availability-zone)
export PublicIP=$(curl -s http://169.254.169.254/latest/meta-data/public-
ipv4)
sudo docker run -d -p 82:80 -h $HOSTNAME --env NAME=$AZ:$PublicIP --
env PORT=82  --env PROTO=TCP --env VALUE=$AZ dockercloud/hello-world

rem Se va al navegador y se visualizan con las IPs publicas los puertos 80,
443 y 82
```


ec2-user@ip-10-0-1-62:~
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
Amazon Linux 2 AMI
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-1-62 ~]$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED
STATUS PORTS NAMES
[ec2-user@ip-10-0-1-62 ~]$ docker stop $(docker ps -aq)
"docker stop" requires at least 1 argument.
See 'docker stop --help'.
Usage: docker stop [OPTIONS] CONTAINER [CONTAINER...]
Stop one or more running containers
[ec2-user@ip-10-0-1-62 ~]$ docker rm $(docker ps -aq)
"docker rm" requires at least 1 argument.
See 'docker rm --help'.
Usage: docker rm [OPTIONS] CONTAINER [CONTAINER...]
Remove one or more containers
[ec2-user@ip-10-0-1-62 ~]$ rem Comprobar las instancias de docker. Se explica el
mapeo de puerto, Zonar horarias y el ejemplo anterior
-bash: rem: command not found
[ec2-user@ip-10-0-1-62 ~]$ sudo docker run -d -p 80:80 -p 443:443 -e TZ=America/
Bogota -h web1 fmorenod81/mtwa:web
Unable to find image 'fmorenod81/mtwa:web' locally
web: Pulling from fmorenod81/mtwa
d50302ca539a: Pull complete
5c32fd3ff3c1: Extracting 58.49MB/163MB
c72a026f110b: Download complete
bb6f881014cd: Download complete
ce6b2e8ae4d3: Download complete
f89bad358ff1: Download complete
710f694e9436: Download complete
1a28d0e66c0c: Download complete
390710edf666: Download complete
ebf7f301f1ca: Download complete
73c642c29021: Download complete
dea4589e3a4e: Download complete



Create target groups, ALB and listeners

```
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%
aws elbv2 create-target-group --name TG-Port-81 --protocol HTTP --port 81 --target-type instance --vpc-
id %vpcn_Id%|jq ".TargetGroups[].TargetGroupArn" >tmpFile
set /p TG81_ARN= < tmpFile
aws elbv2 register-targets --target-group-arn %TG81_ARN% --targets Id=%Instance1Id% Id=%Instance2Id%
aws elbv2 create-target-group --name TG-Port-82 --protocol HTTP --port 82 --target-type instance --vpc-
id %vpcn_Id%|jq ".TargetGroups[].TargetGroupArn" >tmpFile
set /p TG82_ARN= < tmpFile
aws elbv2 register-targets --target-group-arn %TG82_ARN% --targets Id=%Instance1Id% Id=%Instance2Id%

rem Crear el Balanceador
aws elbv2 create-load-balancer --name ALBLab5b --subnets %pbsn1_Id% %pbsn2_Id% --security-groups %SecGroup_ALB_Id% >tmpFile2
cat tmpFile2|jq ".LoadBalancers[].LoadBalancerArn" >tmpFile
set /p LB_ARN= < tmpFile
cat tmpFile2|jq ".LoadBalancers[].DNSName" >tmpFile
set /p LB_DNSName= < tmpFile
```

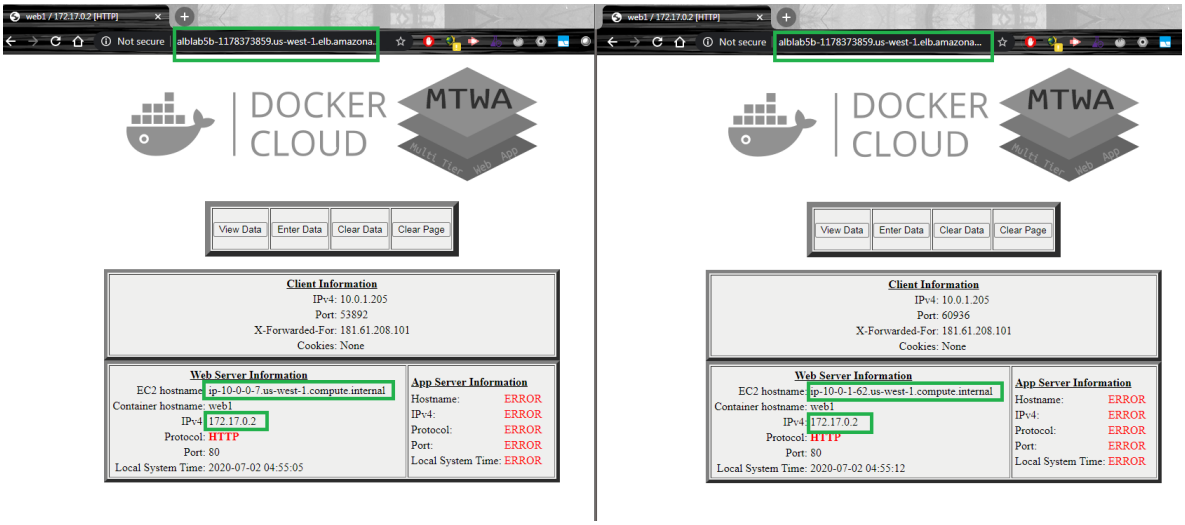
```
del tmpFile2

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
echo Para navegar a %LB_DNSName%
```

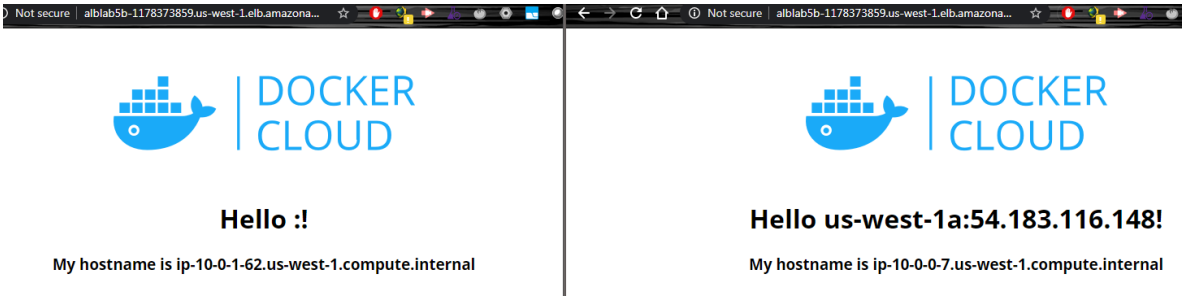
```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>rem Crear los target groups y registrar las instancias a los mismos en cada puerto

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 create-target-group --name TG-Port-80 --protocol HTTP --port 80 --target-type instance --vpc-id %vpcn_Id% |jq ".TargetGroups[].TargetGroupArn" >t
mpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p TG80_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 register-targets --target-group-arn %TG80_ARN% --targets Id=%Instance1Id% Id=%Instance2Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 create-target-group --name TG-Port-81 --protocol HTTP --port 81 --target-type instance --vpc-id %vpcn_Id%|jq ".TargetGroups[].TargetGroupArn" >tm
pFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p TG81_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 register-targets --target-group-arn %TG81_ARN% --targets Id=%Instance1Id% Id=%Instance2Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 create-target-group --name TG-Port-82 --protocol HTTP --port 82 --target-type instance --vpc-id %vpcn_Id%|jq ".TargetGroups[].TargetGroupArn" >tm
pFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p TG82_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 register-targets --target-group-arn %TG82_ARN% --targets Id=%Instance1Id% Id=%Instance2Id%


C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>rem Crear el Balanceador

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 create-load-balancer --name ALBLab5b --subnets %pbsn1_Id% %pbsn2_Id% --security-groups %SecGroup_ALB_Id% >tmpFile2

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>cat tmpFile2|jq ".LoadBalancers[].LoadBalancerArn" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p LB_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>cat tmpFile2|jq ".LoadBalancers[].DNSName" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p LB_DNSName= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>del tmpFile2

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>rem Se crea el Listener para Puerto 80

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 create-listener --load-balancer-arn %LB_ARN% --protocol HTTP --port 80 --default-actions Type=forward,TargetGroupArn=%TG80_ARN%|jq ".Listeners[].
ListenerArn" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>set /p LST80_ARN= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>rem Se prueba que el ALB llegue al target group desde un navegador
```

```
rem Se crea el Listener para Puerto 81, 82
aws elbv2 create-listener --load-balancer-arn %LB_ARN% --protocol HTTP --
port 81 --default-
actions Type=forward,TargetGroupArn=%TG81_ARN%|jq ".Listeners[].ListenerArn"
 >tmpFile
set /p LST81_ARN= < tmpFile
aws elbv2 create-listener --load-balancer-arn %LB_ARN% --protocol HTTP --
port 82 --default-
actions Type=forward,TargetGroupArn=%TG82_ARN%|jq ".Listeners[].ListenerArn"
 >tmpFile
set /p LST82_ARN= < tmpFile
rem Probar porque no funciona en los puertos 81, 82 y 443
```

```
rem Habilitar los sec group al ALB
aws ec2 authorize-security-group-ingress --group-id %SecGroup_ALB_Id% --
protocol tcp --port 81 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SecGroup_ALB_Id% --
protocol tcp --port 82 --cidr 0.0.0.0/0
rem Probar porque funciona en los puertos 81, 82
```

From the LB_DNSName variable review with browser, the balance of 2 instances on port 80, 82 without routing



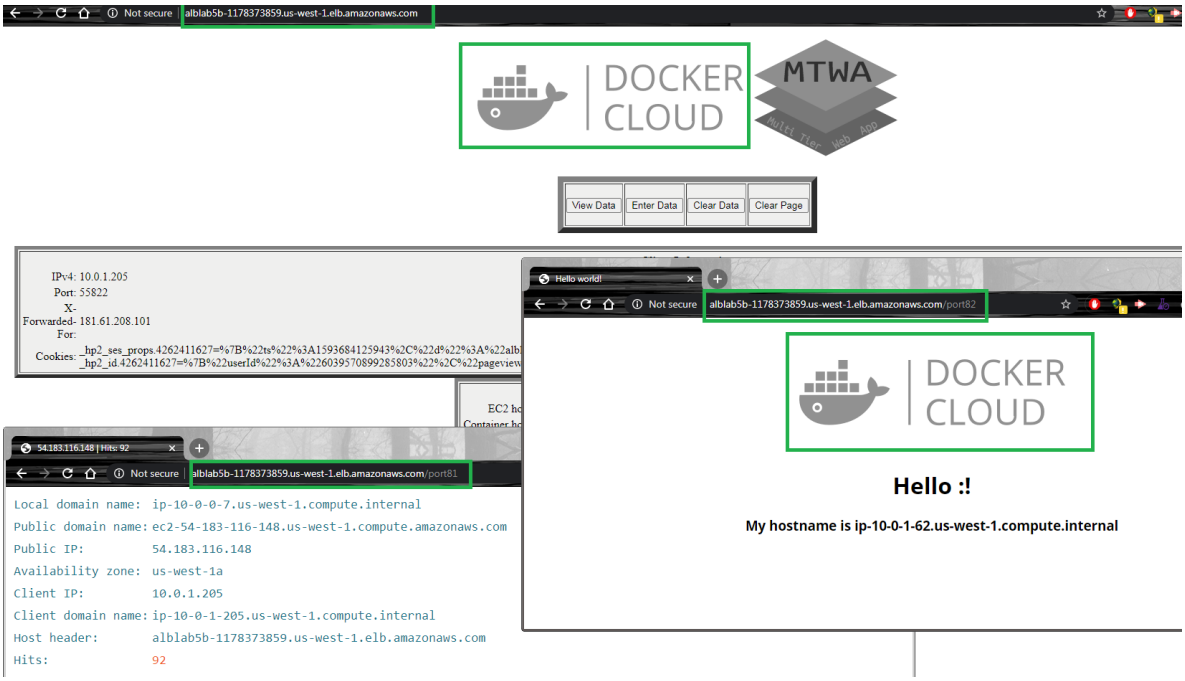


Able Routing Paths

```
rem Crear la regla para el puerto 80 y que cumpla el path del archivo JSON
aws elbv2 create-rule --listener-arn %LST80_ARN% --priority 5 --
conditions file://conditions-pattern-port81.json --
actions Type=forward,TargetGroupArn=%TG81_ARN%
aws elbv2 create-rule --listener-arn %LST80_ARN% --priority 4 --
conditions file://conditions-pattern-port82.json --
actions Type=forward,TargetGroupArn=%TG82_ARN%
rem Revisar las rutas del balanceador
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 create-rule --listener-arn %LST80_ARN% --priority 5 --conditions file://conditions-pattern-port81.json --actions Type=forward,TargetGroupArn=%TG8
1_ARN%
{
    "Rules": [
        {
            "RuleArn": "arn:aws:elasticloadbalancing:us-west-1:455469987488:listener-rule/app/ALBLab5b/4658a6b2ac0b4d26/8e35361758957207/6d19a8c9862f0e5c",
            "Priority": "5",
            "Conditions": [
                {
                    "Field": "path-pattern",
                    "Values": [
                        "/port81"
                    ],
                    "PathPatternConfig": {
                        "Values": [
                            "/port81"
                        ]
                    }
                }
            ],
            "Actions": [
                {
                    "Type": "forward",
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-west-1:455469987488:targetgroup/TG-Port-81/4c871e766a9336ce",
                    "ForwardConfig": {
```

```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s5c1\CLI>aws elbv2 create-rule --listener-arn %LST80_ARN% --priority 4 --conditions file://conditions-pattern-port82.json --actions Type=forward,TargetGroupArn=%TG8
2_ARN%
{
    "Rules": [
        {
            "RuleArn": "arn:aws:elasticloadbalancing:us-west-1:455469987488:listener-rule/app/ALBLab5b/4658a6b2ac0b4d26/8e35361758957207/39280abf3ce1e4d3",
            "Priority": "4",
            "Conditions": [
                {
                    "Field": "path-pattern",
                    "Values": [
                        "/port82"
                    ],
                    "PathPatternConfig": {
                        "Values": [
                            "/port82"
                        ]
                    }
                }
            ],
            "Actions": [
                {
                    "Type": "forward",
                    "TargetGroupArn": "arn:aws:elasticloadbalancing:us-west-1:455469987488:targetgroup/TG-Port-82/51a01dd17dc0bdcb",
                    "ForwardConfig": {
                        "TargetGroups": [
                            {
                                "TargetGroupArn": "arn:aws:elasticloadbalancing:us-west-1:455469987488:targetgroup/TG-Port-82/51a01dd17dc0bdcb",
```



Clean Resources

For Web Management Console

Delete Instances
Delete Keypair
Delete Security Groups
Delete ALB
Delete Target Groups

Delete VPC

Evidences to send

To have a review, the student has to send some screenshots to instructor email:

1. The last picture of Create Additional Listeners for Ports, which show 3 navigation tabs (browsers) with 3 different ports for the same ALB.
2. The last picture of Create Listener Rules for Routing Paths, which show 3 navigation tabs (browsers) with 3 different paths for the same ALB.