



TRABAJO DE GRADO
Opción Seminario-Diplomado.

Seminario AWS

Corporación Universitaria Remington.
Ingenierías:
Ingeniería de Sistemas:

Estudiante: Andrés Felipe Alarcón Guerrero
Docente: Juan Pablo Berrio López

Opción de Trabajo de grado Seminario-Diplomado.
2025

Dedicatoria

Bubalú gracias a tu compañía y amor incondicional.

Agradecimientos

Gracias Juan Pablo Berrio López por tus enseñanzas y estilo pedagógico. Estas sembrando un conocimiento invaluable a cada profesional. Agradecimiento extendido a la Remington por ofrecer educación de extensión de buena calidad y con relevancia en el mercado.

Tabla de Contenidos

Resumen	6
Marco conceptual y contextual	7
Conceptos.....	8
VPC.....	9
Conectividad y Seguridad de Red.....	10
Recursos de Cómputo y Gestión.....	11
ELB (Elastic Load Balancer).....	12
Documentación técnica	13
Conclusiones	31
Figuras	32
Figura 1. VPC Base.....	32
Figura 2. Configuración final de VPC en AWS	33
Figura 3. Instancia de Windows Server	34
Figura 4. Instancia de Linux Server Autoría propia.	35
Figura 5. Conectividad desde Windows Server a Linux Server (Red privada).....	36
Figura 6. Conectividad desde Linux Server a Windows Server (Red privada).....	37
Figura 7. Acceso desde internet http a instancia Linux	38
Figura 8. Acceso desde internet http a instancia Windows	39
Figura 9. Configuración IIS en instancia EC2 Windows.....	40
Figura 10. Validación local de IIS en instancia EC2 Windows.....	41
Figura 11. Load Balancers	42
Figura 12. Load Balancer: Balanceador Docker.....	43
Figura 13. Target Group: TG-docker.....	44
Figura 14. AMI Base Docker.....	45
Figura 15. Docker containers.....	46
Figura 16. Instancias basadas en la AMI Base Docker.....	47
Figura 17. Primera instancia basada en la AMI Base Docker.	48
Figura 18. Segunda instancia basada en la AMI Base Docker.	49
Figura 19. Proxy Reverse a Contenedores Docker	50
Figura 20. Conectividad Contenedor 1	51
Figura 21. Conectividad Contenedor 2	52
Figura 22. Conectividad Contenedor 3	53
Figura 23. Conectividad DNS Load Balancer	54
Figura 24. Configuración Auto Scaling.....	55
Figura 25. Validación Auto Scaling.....	56
Figura 26. Validación Downscaling	57
Figura 27. Diagrama Seguridad, alta disponibilidad y escalabilidad de las aplicaciones.....	58
Referencias	59

Resumen

Este seminario de grado aborda los fundamentos de Amazon Web Services (AWS), plataforma de computación en la nube que ofrece servicios dirigidos tanto a empresas como a desarrolladores. El objetivo principal del seminario es brindar una comprensión clave de los servicios en AWS, buenas prácticas recomendadas para garantizar la seguridad, la alta disponibilidad y la escalabilidad de las aplicaciones implementadas en la nube.

Palabras clave

AWS, computación en la nube, servicios, seguridad, escalabilidad

Marco conceptual y contextual

El presente trabajo describe el proceso de aprendizaje y uso de recursos de cómputo en la nube para AWS. Trabajo guiado por la formación del Seminario AWS de la Remington, donde se abordan conceptos y servicios claves para el uso de AWS en la capa gratuita.

Para la implementación de este trabajo es importante comprender una serie de conceptos que se utilizan en la plataforma AWS.

Conceptos

A continuación, se pondrá en contexto conceptos clave necesarios para comprender el siguiente trabajo. Información tomada de Amazon Web Services. (s.f.) y de Seminario AWS (2025).

VPC

Virtual Private Cloud. Funciona de manera análoga a un centro de datos. Sirve para la creación de redes privadas. En la capa gratuita se permite tener hasta dos VPC sin cobro adicional.

Dentro de un VPC se define todo el direccionamiento IP que utilizarán los recursos asociados, como las máquinas virtuales. Para organizar esta red, el VPC se segmenta en subredes, que pueden establecerse para distintos propósitos, como separar los entornos de desarrollo y administración, o definirse por departamentos.

Conectividad y Seguridad de Red

Las subredes se clasifican en públicas y privadas. Las subredes públicas tienen acceso a internet mediante un Internet Gateway (IGW), que funciona como un enrutador para dar salida y entrada a las conexiones.

Por defecto, las subredes privadas no tienen conexión a internet. Sin embargo, en ocasiones se necesita acceso para tareas como la actualización de software de seguridad. Para solucionar esto sin exponer los recursos a conexiones entrantes, se utiliza un NAT Gateway. Este dispositivo permite que los recursos en la subred privada inicien conexiones hacia internet (salida), pero impide que desde internet se pueda acceder a ellos. Cabe mencionar que el NAT Gateway es un recurso con un costo mensual.

Para optimizar costos y seguridad, se pueden usar VPC Endpoints. Estos permiten que un recurso dentro del VPC acceda a otros servicios de AWS, como el almacenamiento de objetos S3, de forma directa y sin pasar por internet, lo que puede reducir los cargos asociados al NAT Gateway.

Recursos de Cómputo y Gestión

En AWS, una máquina virtual se denomina instancia. Su seguridad perimetral se gestiona a través de Security Groups, que actúan como un firewall. En sus reglas se puede definir el origen o destino del tráfico; por ejemplo, una regla de salida con destino 0.0.0.0/0 indica que se permite la conexión hacia cualquier lugar en internet.

Finalmente, la gestión de costos es un aspecto crítico. A través de AWS Budgets, es posible crear presupuestos que funcionan como una alerta; el sistema notifica cuando el uso está a punto de superar un umbral predefinido. En cuanto al monitoreo, la capa gratuita ofrece métricas con una frecuencia de actualización de 5 minutos, un tiempo de respuesta considerado no apto para entornos productivos, mientras que las opciones de pago ofrecen datos en tiempo real.

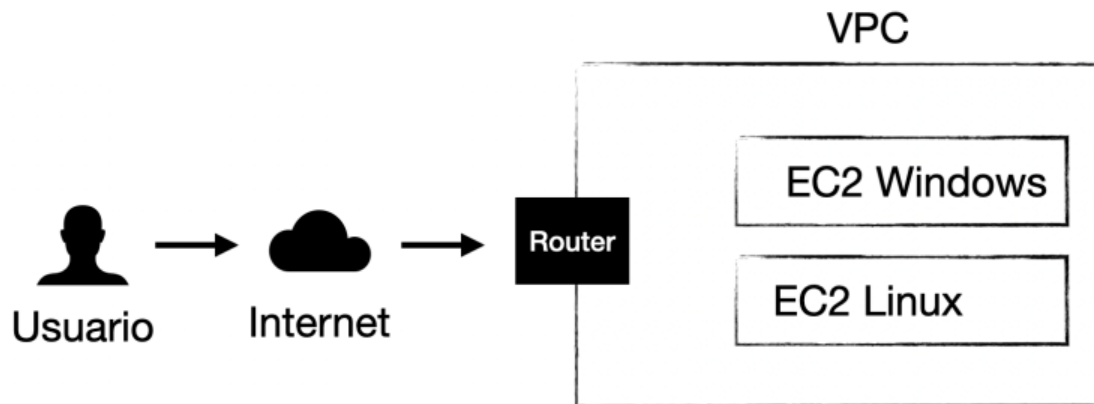
ELB (Elastic Load Balancer)

En AWS, es un recurso que recibe las peticiones y las envía a las instancias de EC2.

Requiere un Recurso llamado Listening, que recibe las peticiones, mismas peticiones que entran al ELB, y finalmente el ELB las envía al TargetGroup, que es el que contiene las instancias que hacen parte del balanceador de carga. El TargetGroup para considerar que una maquina es apta para registrarse tiene que pasar una validación que el servidor web está funcionando (200 http).

Documentación técnica

Figura 1. VPC Base



Autoría propia.

Descripción de la arquitectura

La base de la infraestructura es una Virtual Private Cloud (VPC) personalizada para garantizar un entorno de red aislado.

1. VPC: Se creó una VPC con el nombre `seminario-vc` y un bloque de direcciones IP de `10.0.0.0/16`.
2. Subredes: La VPC contiene múltiples subredes para segmentar la red. Las instancias se desplegaron en subredes públicas, como `seminario-subnet-public1-us-east-1a` con el CIDR `10.0.10.0/20`, para permitir el acceso a Internet.

3. Conectividad a Internet: Un Internet Gateway (seminario-igw) se adjuntó a la VPC y se configuraron tablas de enrutamiento (seminario-rtb-public) para dirigir el tráfico desde y hacia internet a las subredes públicas.

Se desplegaron dos máquinas virtuales (instancias EC2) de tipo t2.micro en la zona de disponibilidad us-east-1a, cada una con un sistema operativo diferente.

Server1Windows

IP Pública: 98.81.36.30

IP Privada: 10.0.13.102

Server2Linux

IP Pública: 54.236.55.37

IP Privada: 10.0.12.218

Configuración de Seguridad

La seguridad y el acceso a las instancias se controlan mediante Grupos de Seguridad, que actúan como firewalls virtuales.

sgWindowsServer: Permite el tráfico entrante para:

RDP (puerto 3389) para administración remota.

HTTP (puerto 80) para el servidor web.

ICMP para permitir pruebas de conectividad (ping).

sgLinuxServer: Permite el tráfico entrante para:

SSH (puerto 22) para administración remota.

HTTP (puerto 80) para el servidor web.

ICMP para permitir pruebas de conectividad (ping).

Estas reglas garantizan que cada servidor sea accesible públicamente a través de la web y que se puedan administrar de forma remota, al tiempo que permiten la comunicación entre ellos.

Configuración realizada

VPC

Figura 2. Configuración final de VPC en AWS

The screenshot displays the AWS Management Console interface for a VPC named 'seminario-vpc' (ID: vpc-0c12bb8b8483fad30). The 'Details' section shows the VPC is in an 'Available' state with 'Block Public Access' turned off. It lists the main network ACL, IPv6 CIDR, and various DNS and DHCP settings. A 'Resource map' section provides a visual overview of the VPC's components: four subnets (two public in us-east-1a and two private in us-east-1b), four route tables (one public and three private), and two network connections (an internet gateway and a VPC-to-VPC connection).

vpc-0c12bb8b8483fad30 / seminaro-vpc

Details [Info](#)

VPC ID vpc-0c12bb8b8483fad30	State Available	Block Public Access Off	DNS hostnames Enabled
DNS resolution Enabled	Tenancy default	DHCP option set dopt-0bb0ae4650b27d0b8	Main route table rtb-08fa63ba
Main network ACL acl-09f45172025204024	Default VPC No	IPv4 CIDR 10.0.0.0/16	IPv6 pool -
IPv6 CIDR (Network border group) -	Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 356994971157

[Resource map](#) | [CIDRs](#) | [Flow logs](#) | [Tags](#) | [Integrations](#)

Resource map [Info](#)

- VPC** [Show details](#)
Your AWS virtual network
seminario-vpc
- Subnets (4)**
Subnets within this VPC
 - us-east-1a
 - seminario-subnet-public1-us-east...
 - seminario-subnet-private1-us-eas...
 - us-east-1b
 - seminario-subnet-public2-us-east...
 - seminario-subnet-private2-us-eas...
- Route tables (4)**
Route network traffic to resources
 - seminario-rtb-public (rtb-08fa63ba690a64b0)
 - seminario-rtb-private2-us-east-1b
 - seminario-rtb-private1-us-east-1a
- Network connections (2)**
Connections to other networks
 - seminario-igw
 - seminario-vpc-e3

Autoría propia.

Figura 3. Instancia de Windows Server

Instances (1/2) info

Find Instance by attribute or tag (case-sensitive) All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
<input checked="" type="checkbox"/> Server1Windows	i-0f0802d391764082f	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a
<input type="checkbox"/> Server2Linux	i-0b527d4c34c9fd66b	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a

i-0f0802d391764082f (Server1Windows)

sg-0512fa896e4f62465 (sgWindowsServer)

Inbound rules

Filter rules

Name	Security group rule ID	Port range	Protocol	Source	Security groups
-	sgr-08b321972a8faa11a	3389	TCP	0.0.0.0/0	sgWindowsServer
-	sgr-0c98eff3617ebaabe	All	ICMP	0.0.0.0/0	sgWindowsServer
-	sgr-0302629c767e07d58	80	TCP	0.0.0.0/0	sgWindowsServer

Outbound rules

Filter rules

Name	Security group rule ID	Port range	Protocol	Destination	Security groups
-	sgr-0f0a43e16c3c044c7	All	All	0.0.0.0/0	sgWindowsServer

Autoría propia.

Linux Server

Figura 4. Instancia de Linux Server

The screenshot displays the AWS Management Console interface for an EC2 instance. At the top, the 'Instances (1/2)' page is visible, showing a table with two instances: 'Server1Windows' and 'Server2Linux'. The 'Server2Linux' instance is selected, and its details are shown below. The instance ID is 'i-0b527ddc34c9fd66b', and it is in a 'Running' state. The instance type is 't2.micro'. The status check shows '2/2 checks passed'. The availability zone is 'us-east-1a'. The instance is associated with the security group 'sg-0cc36cb4e5843bc12 (sgLinuxServer)'. Below the instance details, the 'Inbound rules' section is expanded, showing a table of security group rules. The table has columns for Name, Security group rule ID, Port range, Protocol, Source, and Security groups. There are three inbound rules: one for TCP on port 22, one for ICMP, and one for TCP on port 80. The 'Outbound rules' section is also visible, showing one rule for All protocols on All ports to destination 0.0.0.0/0.

Name	Security group rule ID	Port range	Protocol	Source	Security groups
-	sgr-0a61c4223280075ac	22	TCP	0.0.0.0/0	sgLinuxServer
-	sgr-09d66781131791b61	All	ICMP	0.0.0.0/0	sgLinuxServer
-	sgr-07cfacd7f9825f171	80	TCP	0.0.0.0/0	sgLinuxServer

Name	Security group rule ID	Port range	Protocol	Destination	Security groups
-	sgr-0b510991705747a8c	All	All	0.0.0.0/0	sgLinuxServer

Autoría propia.

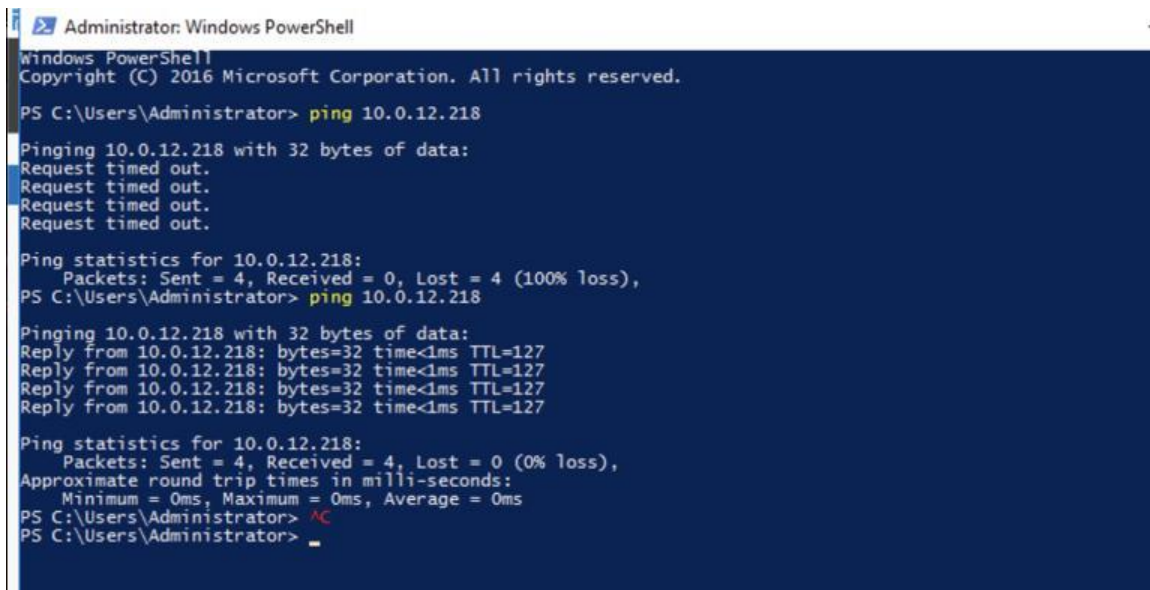
Procedimiento de Acceso

Se validó el cumplimiento de los requisitos de conectividad.

Accesibilidad Pública: La asignación de IPs públicas y las reglas de los grupos de seguridad en el puerto 80 aseguran que los servidores web instalados en cada instancia sean accesibles desde internet.

Conectividad Interna: Las pruebas de ping entre las instancias utilizando sus direcciones IP privadas fueron exitosas. La máquina Windows (10.0.13.102) pudo comunicarse con la máquina Linux (10.0.12.218) y viceversa, confirmando que la red interna es funcional.

Figura 5. Conectividad desde Windows Server a Linux Server (Red privada)



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> ping 10.0.12.218

Pinging 10.0.12.218 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

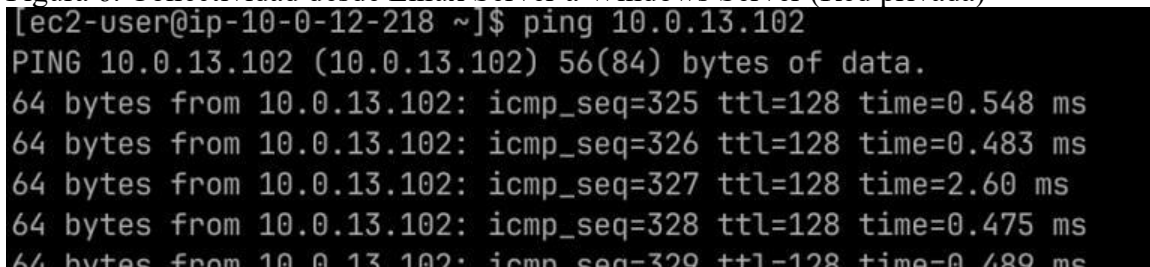
Ping statistics for 10.0.12.218:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PS C:\Users\Administrator> ping 10.0.12.218

Pinging 10.0.12.218 with 32 bytes of data:
Reply from 10.0.12.218: bytes=32 time<1ms TTL=127
Reply from 10.0.12.218: bytes=32 time<1ms TTL=127
Reply from 10.0.12.218: bytes=32 time<1ms TTL=127
Reply from 10.0.12.218: bytes=32 time<1ms TTL=127

Ping statistics for 10.0.12.218:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
PS C:\Users\Administrator> ^C
PS C:\Users\Administrator> _
```

Autoría propia.

Figura 6. Conectividad desde Linux Server a Windows Server (Red privada)

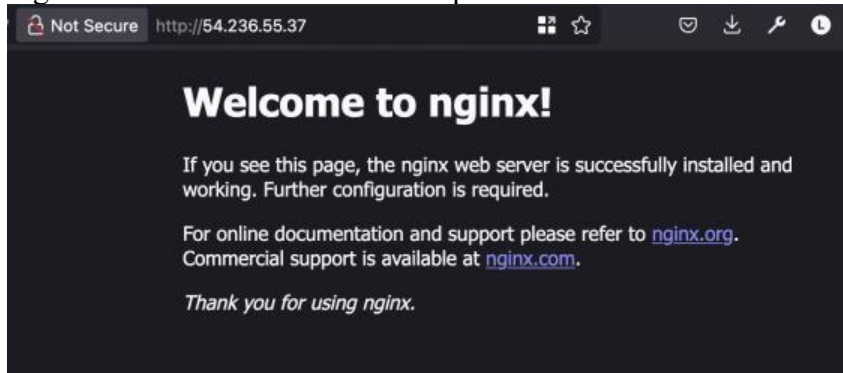


```
[ec2-user@ip-10-0-12-218 ~]$ ping 10.0.13.102
PING 10.0.13.102 (10.0.13.102) 56(84) bytes of data.
64 bytes from 10.0.13.102: icmp_seq=325 ttl=128 time=0.548 ms
64 bytes from 10.0.13.102: icmp_seq=326 ttl=128 time=0.483 ms
64 bytes from 10.0.13.102: icmp_seq=327 ttl=128 time=2.60 ms
64 bytes from 10.0.13.102: icmp_seq=328 ttl=128 time=0.475 ms
64 bytes from 10.0.13.102: icmp_seq=329 ttl=128 time=0.489 ms
```

Autoría propia.

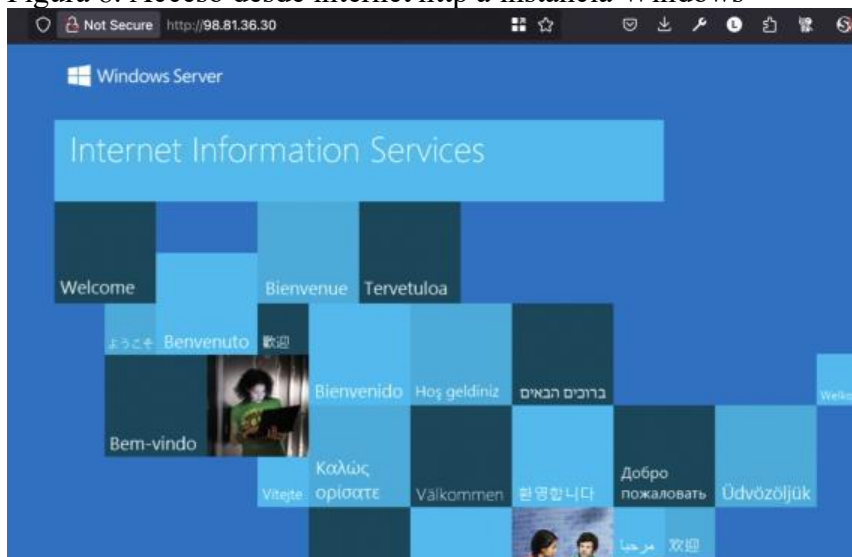
Internet

Figura 7. Acceso desde internet http a instancia Linux



Autoría propia.

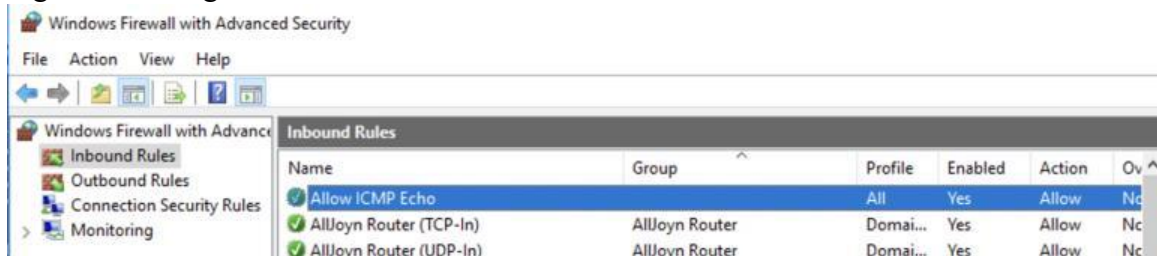
Figura 8. Acceso desde internet http a instancia Windows



Autoría propia.

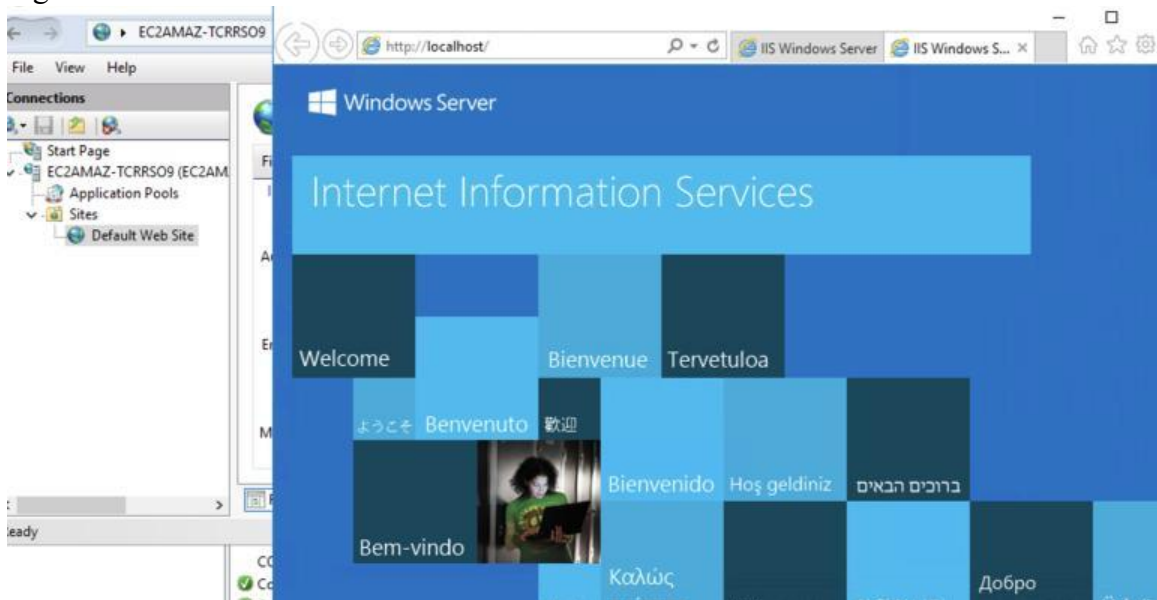
Configuración del Servidor

Figura 9. Configuración IIS en instancia EC2 Windows



Autoría propia.

Figura 10. Validación local de IIS en instancia EC2 Windows



Autoría propia.

Linux

```
sudo dnf update -y
sudo dnf install nginx -y
sudo systemctl start nginx
sudo systemctl enable nginx
```

```
[ec2-user@ip-10-0-12-218 ~]$ curl http://localhost
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Figura 11. Load Balancers

The screenshot shows the AWS Management Console interface for Load Balancers. At the top, there's a header with 'Load balancers (2)' and a description: 'Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.' Below this is a search bar and a table of load balancers.

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
LoadBalancer1	LoadBalancer1-121186583...	Active	vpc-Oc12bb8b8483fad30	2 Availability Zones	application	Jul 20, 2025, 18:34 (UTC-04:00)
BalanceadorDocker	BalanceadorDocker-243088...	Active	vpc-Oc12bb8b8483fad30	2 Availability Zones	application	Jul 20, 2025, 18:34 (UTC-04:00)

On the right side, there's a user menu with options: Account ID (3569-9497-1157), Account, Organization, Service Quotas, Billing and Cost Management, Security credentials, Turn on multi-session support, and Sign out.

Autoría propia.

Figura 12. Load Balancer: BalanceadorDocker

The screenshot shows the detailed view of the 'BalanceadorDocker' load balancer. It includes a notification about IPAM support, a 'Details' section with various attributes, and a 'Listeners and rules' section.

Details:

- Load balancer type:** Application
- Status:** Active
- VPC:** vpc-Oc12bb8b8483fad30
- Load balancer IP address type:** IPv4
- Scheme:** Internet-facing
- Hosted zone:** Z35XDOTRQ7K7K
- Availability Zones:** subnet-00c24d89df80967 (us-east-1a (use1-az1)), subnet-0cf254aefb35c79e1 (us-east-1b (use1-az2))
- Date created:** July 20, 2025, 18:34 (UTC-04:00)
- Load balancer ARN:** arn:aws:elasticloadbalancing:us-east-1:356994971157:loadbalancer/app/BalanceadorDocker/D954b4d0a3997a
- DNS name:** BalanceadorDocker-243088212.us-east-1.elb.amazonaws.com (A Record)

Listeners and rules (1) info:

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Protocol/Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Trust store
HTTP:80	Forward to target group <ul style="list-style-type: none"> TG-docker (1 (100%)) Target group stickiness: Off 	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not applicable

On the right side, there's a user menu with options: Account ID (3569-9497-1157), Account, Organization, Service Quotas, Billing and Cost Management, Security credentials, Turn on multi-session support, and Sign out.

Autoría propia.

Figura 13. Target Group: TG-docker

The screenshot shows the AWS Management Console interface for a Target Group named 'TG-docker'. The 'Details' section displays the following information:

- Target type:** Instance
- Protocol:** HTTP
- Port:** 80
- Protocol version:** HTTP1
- VPC:** vpc-0c12b6b8483fa650
- IP address type:** IPv4
- Lead balancer:** BalancerLoadBalancer

Summary statistics:

- Total targets: 2
- Healthy: 2
- Unhealthy: 0
- Anomalous: 0
- Unused: 0
- Initial: 0
- Drain: 0

The 'Registered targets (2)' section shows a table with the following data:

Instance ID	Name	Port	Zone	Health status	Health status details	Admini...	Overri...	Launch...	Anomaly detection...
i-08ef487409af08e	ServerLinuxPublico	80	us-east-1...	Healthy	-	No override	No overri...	July 20, 2...	Normal
i-0e5279dc54c9f66b	ServerLinux	80	us-east-1...	Healthy	-	No override	No overri...	July 4, 20...	Normal

Autoría propia.

Figura 14. AMI Base Docker

Amazon Machine Image (AMI)

AmzLinuxDockerNginx

ami-0ce272c2056053858

2025-07-20T22:25:37.000Z

Virtualization: hvm

ENA enabled: true

Root device type: ebs

Boot mode: uefi-preferred

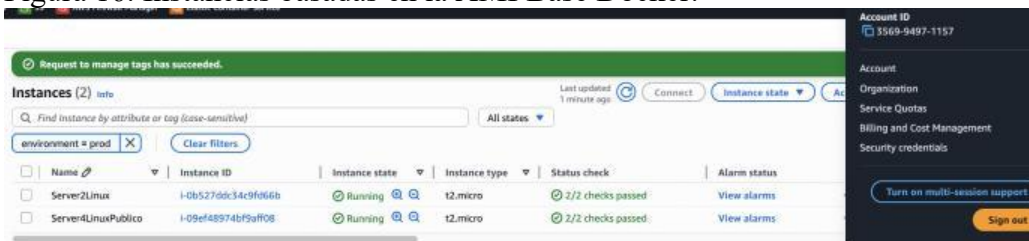
Autoría propia.

Figura 15. Docker containers

```
[root@ip-10-0-12-218 ec2-user]# docker run -dit --name app1 -d --restart always -p 81:80 nginx
2bb2db53e03e29ddee937bffcfe2a6106d069057bb3421a2f7f069740006c
[root@ip-10-0-12-218 ec2-user]# docker run -dit --name app2 -d --restart always -p 82:80 nginx
db1d16bf23a38e1998f411d3a8986c149e31a17b4421061e45545efff75e1910
[root@ip-10-0-12-218 ec2-user]# docker run -dit --name app3 -d --restart always -p 83:80 nginx
05ed0a63de63ea191bf73e64e202deb7c1a9d2c718808f69f920eb0e1b7c71
[root@ip-10-0-12-218 ec2-user]# docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
05ed0a63de63   nginx    "/docker-entrypoint..." 5 seconds ago Up 4 seconds  0.0.0.0:83→80/tcp, :::83→80/tcp   app3
db1d16bf23a3   nginx    "/docker-entrypoint..." 15 seconds ago Up 14 seconds  0.0.0.0:82→80/tcp, :::82→80/tcp   app2
2bb2db53e03e   nginx    "/docker-entrypoint..." 24 seconds ago Up 22 seconds  0.0.0.0:81→80/tcp, :::81→80/tcp   app1
[root@ip-10-0-12-218 ec2-user]#
```

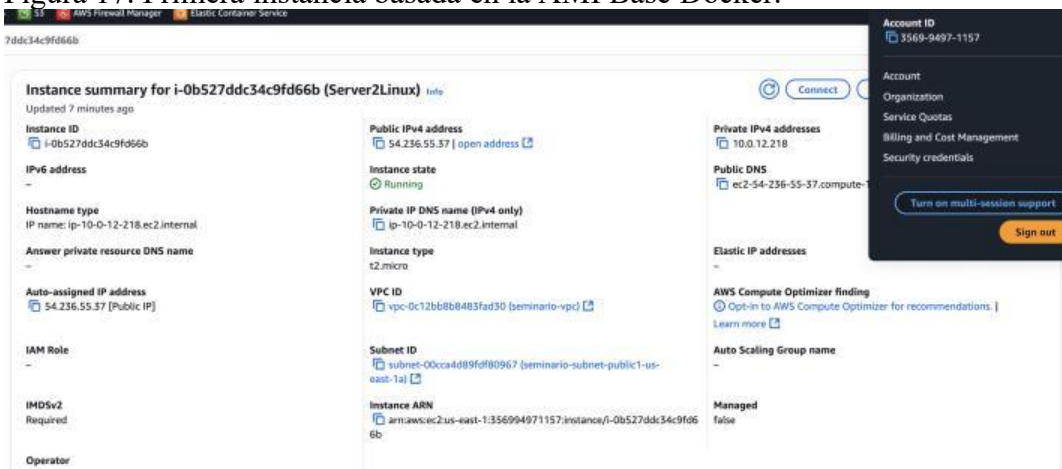
Autoría propia.

Figura 16. Instancias basadas en la AMI Base Docker.



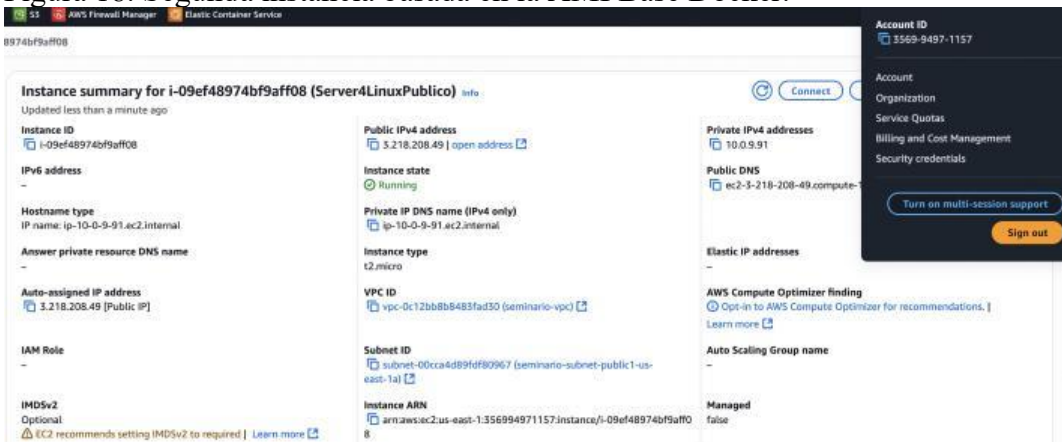
Autoría propia.

Figura 17. Primera instancia basada en la AMI Base Docker.



Autoría propia.

Figura 18. Segunda instancia basada en la AMI Base Docker.



Autoría propia.

Figura 19. Proxy Reverse a Contenedores Docker

```
[root@ip-10-0-12-218 ec2-user]# systemctl status nginx.service
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
   Active: active (running) since Sun 2025-07-20 21:20:46 UTC; 2h 46min ago
     Process: 736780 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
     Process: 736781 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
     Process: 736782 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 736783 (nginx)
     Tasks: 2 (Limit: 1111)
   Memory: 2.0M
     CPU: 176ms
   CGroup: /system.slice/nginx.service
           └─736783 "nginx: master process /usr/sbin/nginx"
             └─736784 "nginx: worker process"

Jul 20 21:20:46 ip-10-0-12-218.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Jul 20 21:20:46 ip-10-0-12-218.ec2.internal nginx[736781]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Jul 20 21:20:46 ip-10-0-12-218.ec2.internal nginx[736781]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Jul 20 21:20:46 ip-10-0-12-218.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[root@ip-10-0-12-218 ec2-user]# cat /etc/nginx/nginx.conf
events {}

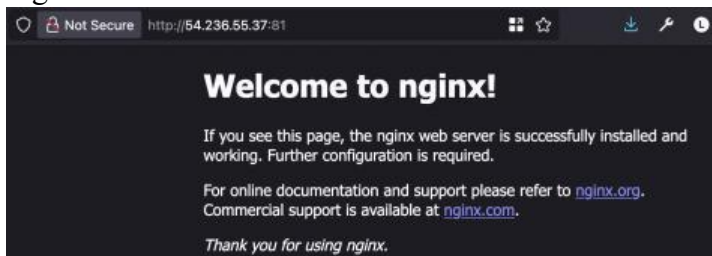
http {
    upstream seminario {
        server localhost:81;
        server localhost:82;
        server localhost:83;
    }

    server {
        listen 80;
        server_name nginx;
        location / {
            proxy_pass http://seminario;
        }
    }
}

[root@ip-10-0-12-218 ec2-user]#
```

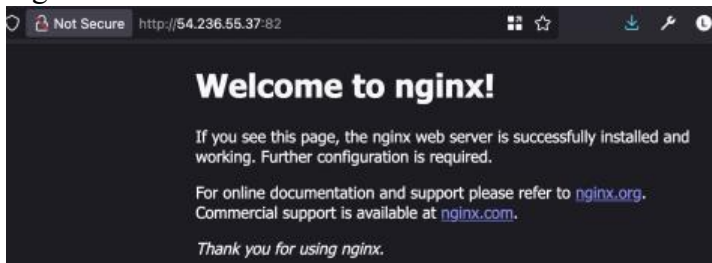
Autoría propia.

Figura 20. Conectividad Contenedor 1



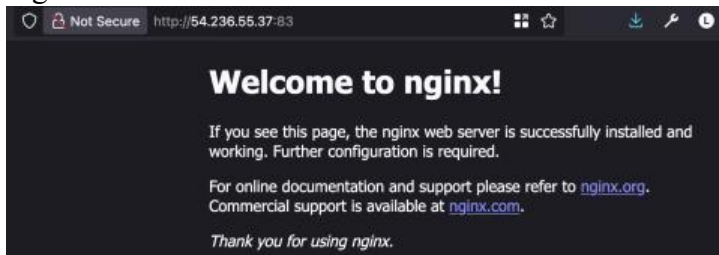
Autoría propia.

Figura 21. Conectividad Contenedor 2



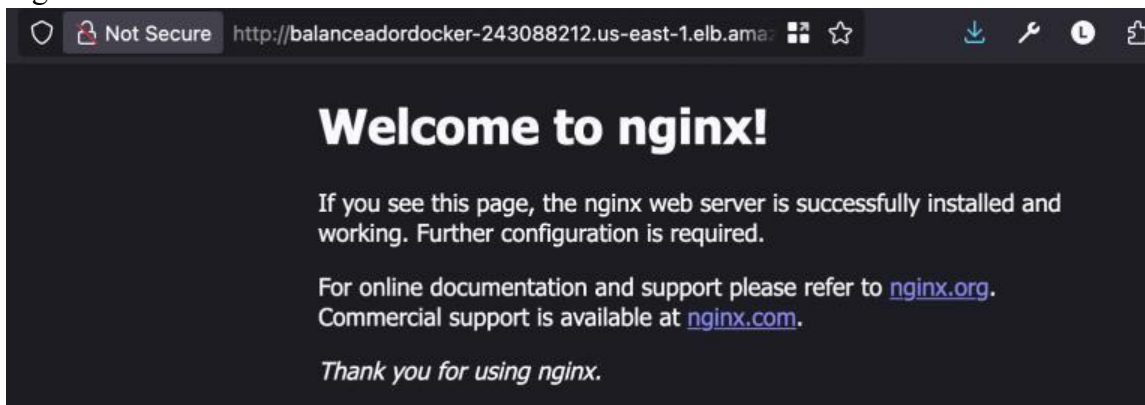
Autoría propia.

Figura 22. Conectividad Contenedor 3



Autoría propia.

Figura 23. Conectividad DNS Load Balancer



Autoría propia.

Figura 24. Configuración Auto Scaling

AutoScalingSeminario

AutoScalingSeminario Capacity overview

arn:aws:autoscaling:us-east-1:356994971157:autoScalingGroup:5b199f2-8c48-4864-8cb7-dbc7449d2a3:autoScalingGroupName/AutoScalingSeminario

Desired capacity	Scaling limits (Min - Max)	Desired capacity type	Status
1	1 - 5	Units (number of instances)	-

Date created
Sun Jul 20 2025 13:46:05 GMT-0400 (Eastern Daylight Saving Time)

Details | Integrations - new | Automatic scaling | Instance management | Instance refresh | Activity | Monitoring

Launch template

Launch template arn:aws:ec2:us-east-1:356994971157:launch-template/PlantillaServerLinux	AMI ID ami-04b1b45b7d527f30e0	Instance type t2.micro	Owner arn:aws:iam::356994971157:root
Version Default	Security groups -	Security group IDs sg-0c145a25e4a47ca7f1	Create time Sun Jul 20 2025 13:17:28 GMT-0400 (Eastern Daylight Saving Time)
Description PlantillaServerLinux	Storage (volumes) -	Key pair name Server2Linux	Request Spot instances No

View details in the launch template console

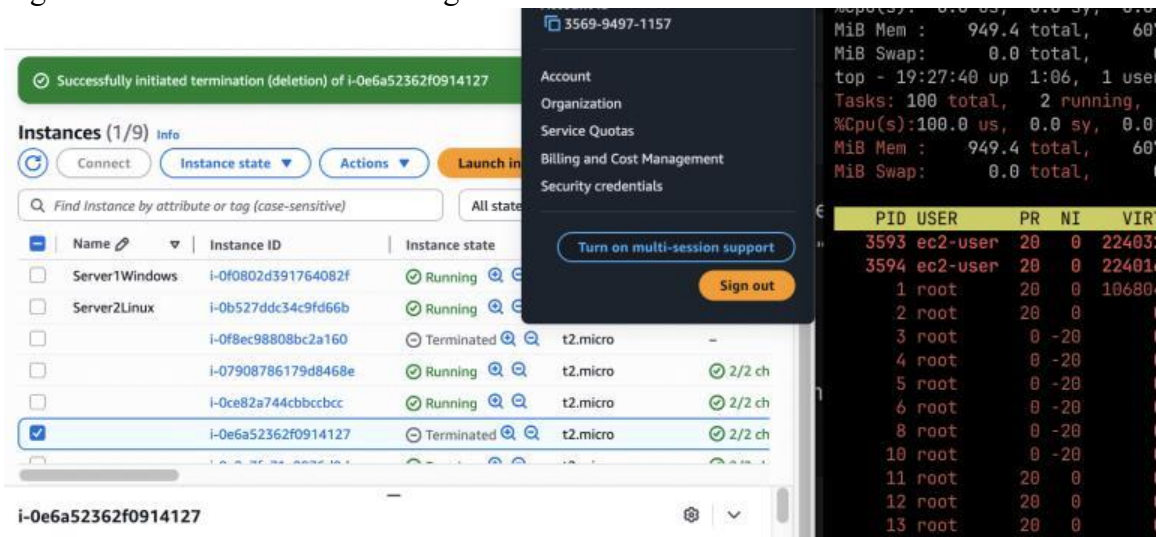
Network

Availability Zones us-east-1a us-east-1b us-east-1c us-east-1d	Subnet ID subnet-0977c16a2099f73f0 subnet-02b55f0d06afeb83	Availability Zone distribution Balanced best effort
-------------------------------------------------------------------	------------------------------------------------------------------	--------------------------------------------------------

Account: 3569-9497-1157
Organization
Service Quotas
Billing and Cost Management
Security credentials
Turn on multi-session support
Sign out

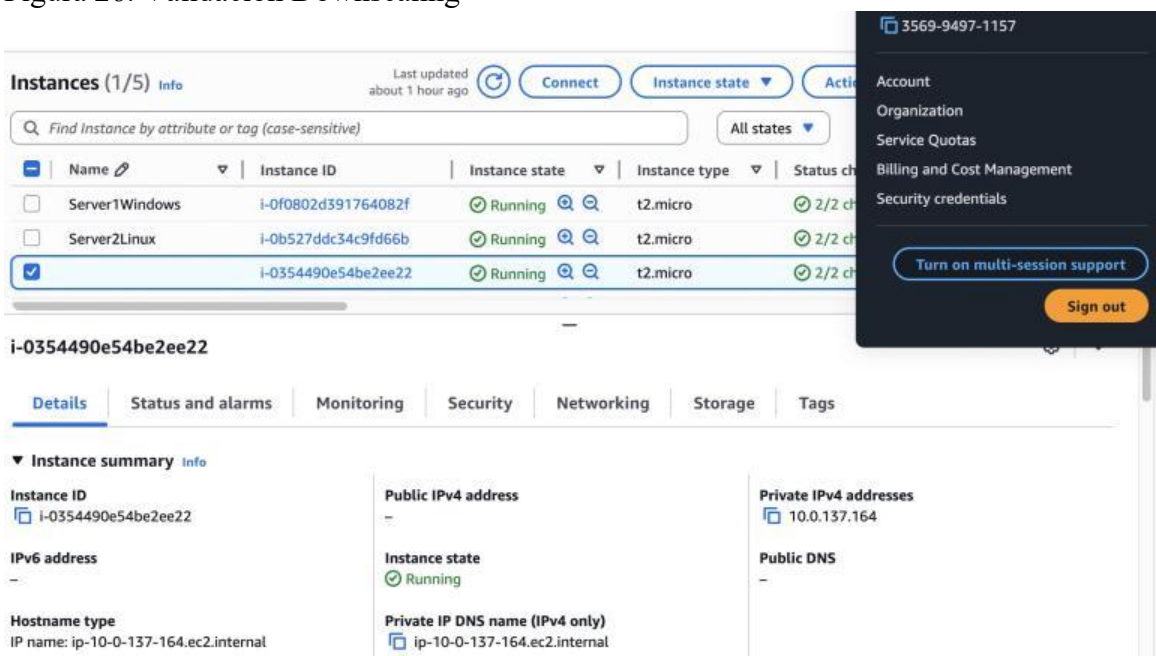
Autoría propia.

Figura 25. Validación Auto Scaling



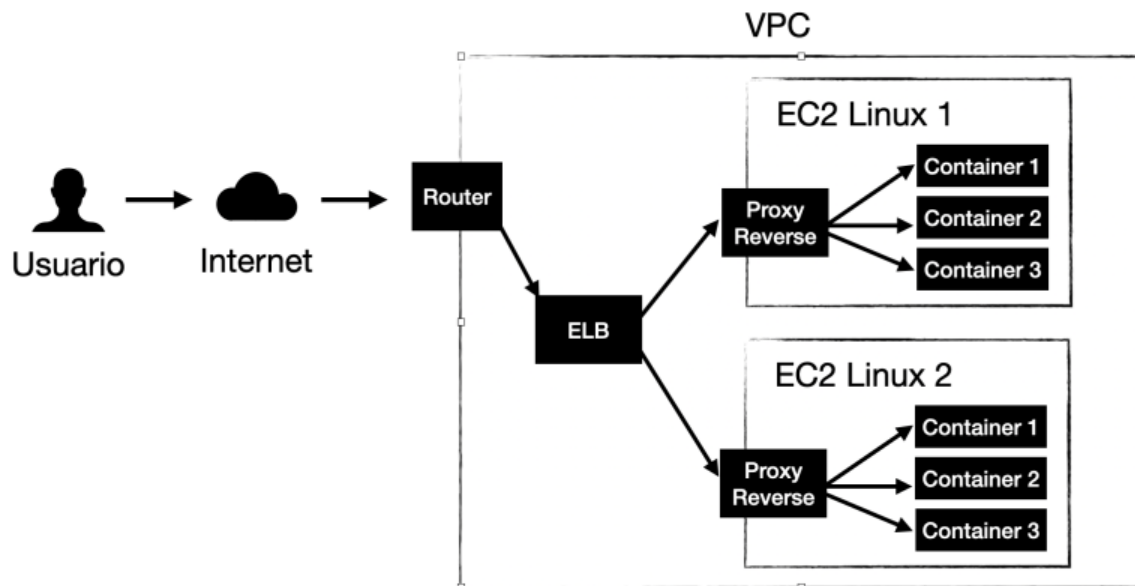
Autoría propia.

Figura 26. Validación Downscaling



Autoría propia.

Figura 27. Diagrama Seguridad, alta disponibilidad y escalabilidad de las aplicaciones.



Autoría propia.

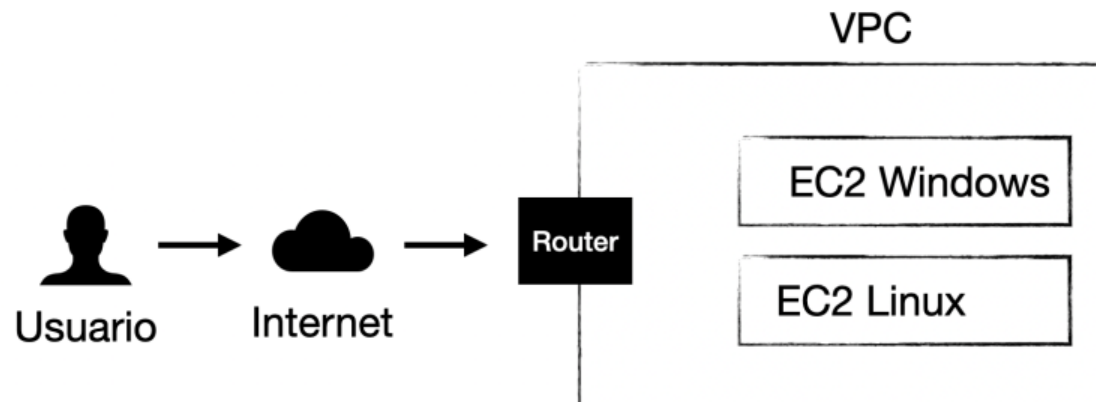
Conclusiones

El presente trabajo concluye con la implementación de una arquitectura en la nube de las más necesarias y fundamentales en AWS, con un enfoque en la seguridad, alta disponibilidad y escalabilidad de dichas aplicaciones.

Implementación que hace uso de conceptos clave en implementaciones maduras puestas en producción como lo son el uso de contenedores, balanceadores de carga, máquinas virtuales, centro de datos, firewalls, subredes, proxy inverso, redundancia, servidores de aplicaciones, control de costos y notificaciones, por mencionar los más relevantes.

Figuras

Figura 1. VPC Base



Autoría propia.

Figura 2. Configuración final de VPC en AWS

The screenshot displays the AWS Management Console for a VPC named 'vpc-0c12bb8b8483fad30 / seminario-vpc'. The interface is divided into several sections:

- Details:** Shows VPC ID (vpc-0c12bb8b8483fad30), State (Available), DNS resolution (Enabled), Main network ACL (acl-09f45172025204024), IPv6 CIDR (Network border group), Block Public Access (Off), DHCP option set (dopt-0bb0ae4650b27d0b8), IPv4 CIDR (10.0.0.0/16), and Route 53 Resolver DNS Firewall rule groups (Disabled).
- Resource map:** A diagram showing the VPC's internal structure and connections:
 - Subnets (4):** Includes two public subnets in us-east-1a (seminario-subnet-public1-us-east-1 and seminario-subnet-public2-us-east-1) and two private subnets (seminario-subnet-private1-us-east-1 and seminario-subnet-private2-us-east-1).
 - Route tables (4):** Includes seminario-rtb-public, rtb-08fa63bba590a64b0, seminario-rtb-private2-us-east-1b, and seminario-rtb-private1-us-east-1a.
 - Network connections (2):** Includes seminario-igw and seminario-vpce-s3.

Autoría propia.

Figura 3. Instancia de Windows Server

The screenshot displays the AWS Management Console interface for an EC2 instance. At the top, there is a search bar and a filter for 'All states'. Below this, a table lists instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
Server1Windows	i-0f0802d391764082f	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a
Server2Linux	i-0b527ddc34c9fd66b	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a

The selected instance, 'Server1Windows' (ID: i-0f0802d391764082f), is shown in detail. It is associated with the security group 'sg-0512fa896e4f62465 (sgWindowsServer)'. The console shows the following security group rules:

Inbound rules:

Name	Security group rule ID	Port range	Protocol	Source	Security groups
-	sgr-08b321972a8faa11a	3389	TCP	0.0.0.0/0	sgWindowsServer
-	sgr-0c98eff3617ebaabe	All	ICMP	0.0.0.0/0	sgWindowsServer
-	sgr-0302629c767e07d58	80	TCP	0.0.0.0/0	sgWindowsServer

Outbound rules:

Name	Security group rule ID	Port range	Protocol	Destination	Security groups
-	sgr-0f0a43e16c3c044c7	All	All	0.0.0.0/0	sgWindowsServer

A user menu is visible on the right side of the screen, showing the account ID (3569-9497-1157) and options for Account, Organization, Service Quotas, Billing and Cost Management, Security credentials, and a 'Sign out' button.

Autoría propia.

Figura 4. Instancia de Linux Server

Instances (1/2) Info

Find instance by attribute or tag (case-sensitive) All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
Server1Windows	i-0f0802d391764082f	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a
Server2Linux	i-0b527ddc34c9fd66b	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a

i-0b527ddc34c9fd66b (Server2Linux)

sg-0cc36cb4e5843bc12 (sgLinuxServer)

▼ Inbound rules

Filter rules < 1 >

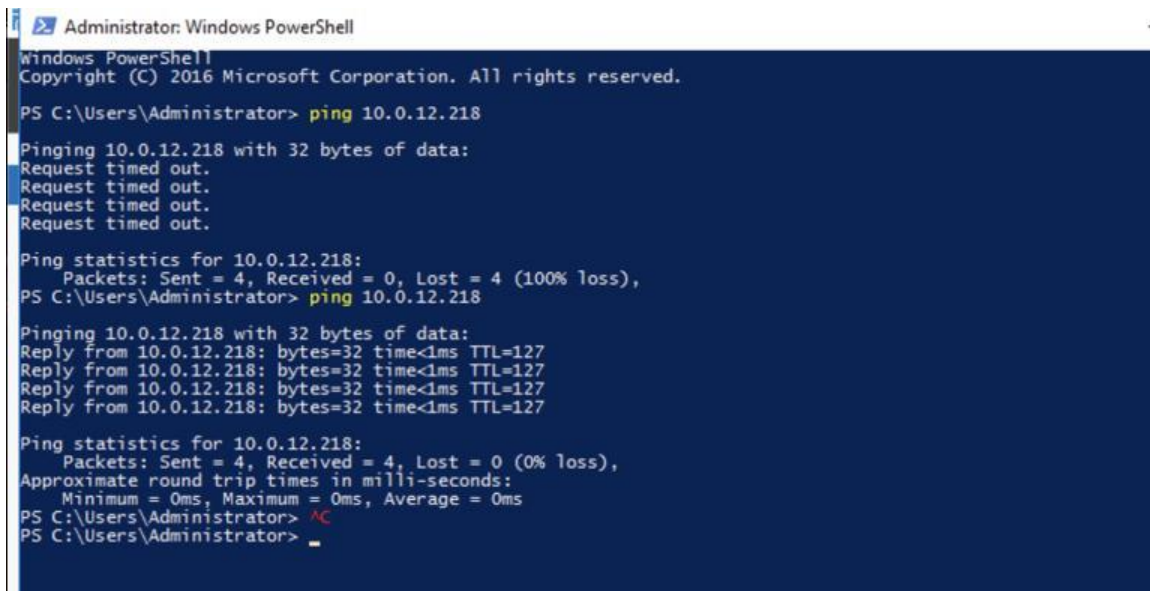
Name	Security group rule ID	Port range	Protocol	Source	Security groups
-	sgr-0a61c4223280075ac	22	TCP	0.0.0.0/0	sgLinuxServer
-	sgr-09d66781131791b61	All	ICMP	0.0.0.0/0	sgLinuxServer
-	sgr-07cfacd7f9825f171	80	TCP	0.0.0.0/0	sgLinuxServer

▼ Outbound rules

Filter rules < 1 >

Name	Security group rule ID	Port range	Protocol	Destination	Security groups
-	sgr-0b51d991705747a8c	All	All	0.0.0.0/0	sgLinuxServer

Autoría propia.

Figura 5. Conectividad desde Windows Server a Linux Server (Red privada)

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> ping 10.0.12.218

Pinging 10.0.12.218 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 10.0.12.218:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PS C:\Users\Administrator> ping 10.0.12.218

Pinging 10.0.12.218 with 32 bytes of data:
Reply from 10.0.12.218: bytes=32 time<1ms TTL=127
Reply from 10.0.12.218: bytes=32 time<1ms TTL=127
Reply from 10.0.12.218: bytes=32 time<1ms TTL=127
Reply from 10.0.12.218: bytes=32 time<1ms TTL=127

Ping statistics for 10.0.12.218:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
PS C:\Users\Administrator> ^C
PS C:\Users\Administrator> _
```

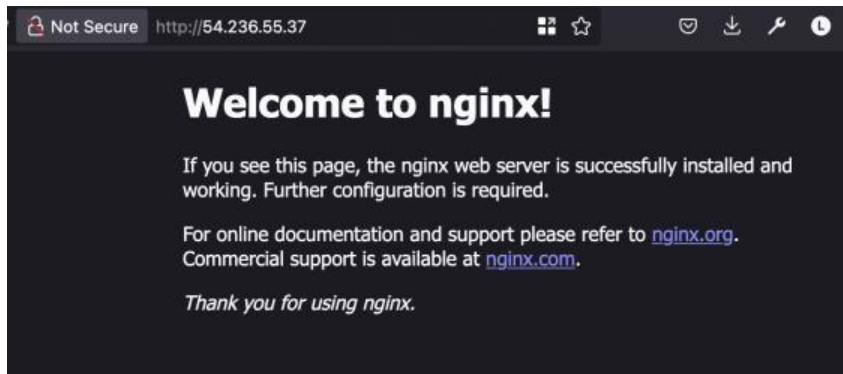
Autoría propia.

Figura 6. Conectividad desde Linux Server a Windows Server (Red privada)

```
[ec2-user@ip-10-0-12-218 ~]$ ping 10.0.13.102
PING 10.0.13.102 (10.0.13.102) 56(84) bytes of data.
64 bytes from 10.0.13.102: icmp_seq=325 ttl=128 time=0.548 ms
64 bytes from 10.0.13.102: icmp_seq=326 ttl=128 time=0.483 ms
64 bytes from 10.0.13.102: icmp_seq=327 ttl=128 time=2.60 ms
64 bytes from 10.0.13.102: icmp_seq=328 ttl=128 time=0.475 ms
64 bytes from 10.0.13.102: icmp_seq=329 ttl=128 time=0.489 ms
```

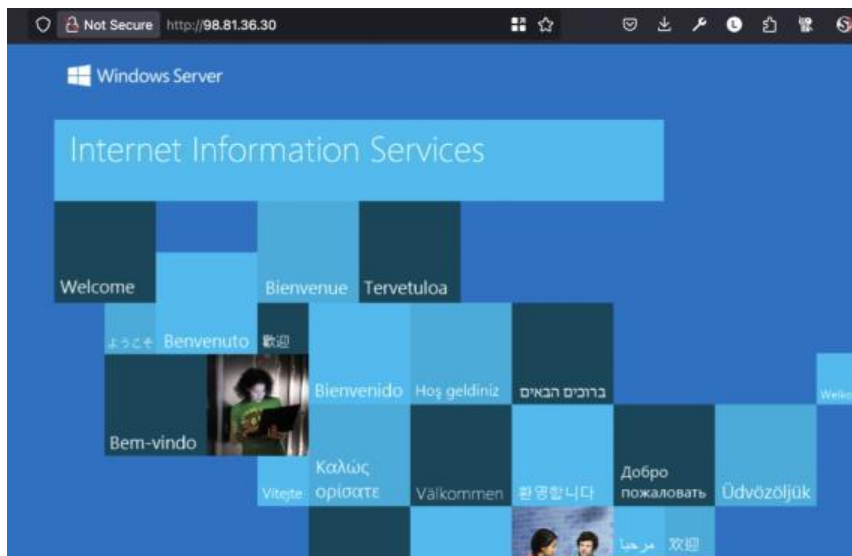
Autoría propia.

Figura 7. Acceso desde internet http a instancia Linux



Autoría propia.

Figura 8. Acceso desde internet http a instancia Windows

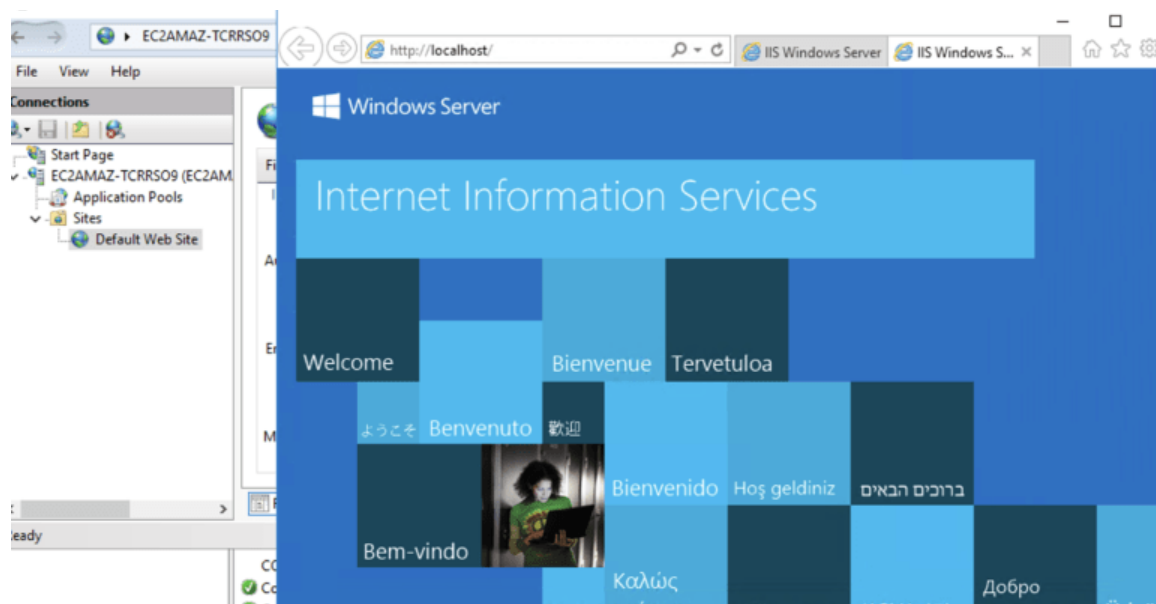


Autoría propia.

Figura 9. Configuración IIS en instancia EC2 Windows

Autoría propia.

Figura 10. Validación local de IIS en instancia EC2 Windows



Autoría propia.

Figura 11. Load Balancers

Load balancers (2)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers

<input type="checkbox"/>	Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
<input type="checkbox"/>	LoadBalancer1	LoadBalancer1-121186583...	Active	vpc-0c12bb8b8483fad30	2 Availability Zones	application	Jul 2017
<input type="checkbox"/>	BalanceadorDocker	BalanceadorDocker-243088...	Active	vpc-0c12bb8b8483fad30	2 Availability Zones	application	Jul 2017

Account ID: 3569-9487-1157

Account

Organization

Service Quotas

Billing and Cost Management

Security credentials

Turn on multi-session support

Sign out

Autoría propia.

Figura 12. Load Balancer: BalanceadorDocker

The screenshot displays the AWS Management Console interface for an Amazon Elastic Load Balancing (ELB) instance. The instance is named "BalanceadorDocker" and is currently in an "Active" state. The console shows various configuration details, including the VPC, availability zones, and the date of creation. A notification banner at the top indicates that Application Load Balancers now support public IPv4 IP Address Management (IPAM). The "Listeners and rules" section shows a single listener configured for HTTP on port 80, with traffic forwarded to a target group.

BalanceadorDocker

Application Load Balancers now support public IPv4 IP Address Management (IPAM). You can get started with this feature by configuring IP pools in the Network mapping section.

Details

Load balancer type Application	Status Active	VPC vpc-0c126d8b6483fad30	Load balancer IP address type IPv4
Scheme Internet-facing	Hosted zone Z35SXDOTRQ7X7K	Availability Zones subnet-0bcca489f9f80967 us-east-1a (us-east-1-az1) subnet-0c7254aaf833c7be1 us-east-1b (us-east-1-az2)	Date created July 20, 2025, 18:34 (UTC-04:00)
Load balancer ARN arn:aws:elasticloadbalancing:us-east-1:356094971157:loadbalancer/app/BalanceadorDocker/DP954b4d6c599f7a		DNS name BalanceadorDocker-243D88212.us-east-1.elb.amazonaws.com [A Record]	

Listeners and rules (1)

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Trust store
HTTP:80	Forward to target group • Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not applicable

Autoría propia.

Figura 13. Target Group: TG-docker

The screenshot displays the AWS Management Console interface for a Target Group named 'TG-docker'. The top navigation bar shows the account ID '3569-9497-1157' and a 'Sign out' button. The main content area is divided into several sections:

- Details:** Shows the target group's ARN, target type (Instance), IP address type (IPv4), protocol (HTTP), port (80), protocol version (HTTP1), and VPC ID.
- Health Status:** A summary bar indicates 2 total targets, with 2 healthy, 0 unhealthy, 0 anomalous, 0 unused, 0 initial, and 0 draining.
- Distribution of targets by Availability Zone (AZ):** A section for selecting filters to view registered targets.
- Registered targets (2):** A table listing two registered targets with columns for Instance ID, Name, Port, Zone, Health status, Health status details, Admin, Over, Launch, and Anomaly detection.

Instance ID	Name	Port	Zone	Health status	Health status details	Admin	Over	Launch	Anomaly detection
i-09ef48974b979aff08	Server4LinuxPublico	80	us-east-1...	Healthy	-	No override	No overm...	July 20, 2...	Normal
i-0b52769b34e9f665b	Server2Linux	80	us-east-1...	Healthy	-	No override	No overm...	July 4, 20...	Normal

Autoría propia.

Figura 14. AMI Base Docker

Amazon Machine Image (AMI)

AmzLinuxDockerNginx
ami-0ce272c2056053858
2025-07-20T22:25:37.000Z Virtualization: hvm ENA enabled: true Root device type: ebs Boot mode: uefi-preferred

Autoría propia.

Figura 15. Docker containers

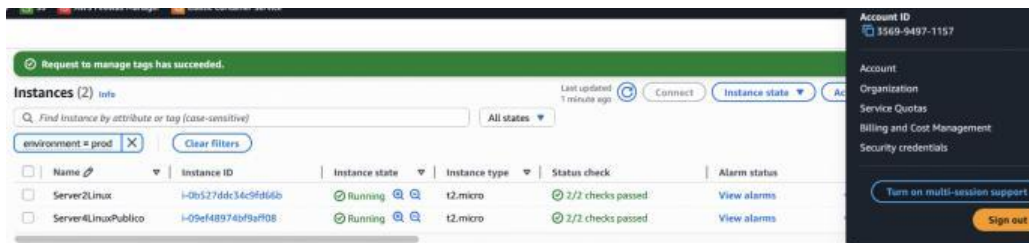
```
[root@ip-10-0-12-218 ec2-user]# docker run -dit --name app1 -d --restart always -p 81:80 nginx
2bb2db53e03e29ddee937bfffcefeb2a6106d069057bb3421a2f7f069740006c
[root@ip-10-0-12-218 ec2-user]# docker run -dit --name app2 -d --restart always -p 82:80 nginx
db1d16bf23a38e1998f411d3a8986c149e31a17b4421061e45545efff75e1910
[root@ip-10-0-12-218 ec2-user]# docker run -dit --name app3 -d --restart always -p 83:80 nginx
05ed0a63de63ea191bffb73e64e202deb7c1a9d2c7188d8f69f920eb0e1b7c71
[root@ip-10-0-12-218 ec2-user]# docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
05ed0a63de63	nginx	"/docker-entrypoint..."	5 seconds ago	Up 4 seconds	0.0.0.0:83→80/tcp, ::: 83→80/tcp	app3
db1d16bf23a3	nginx	"/docker-entrypoint..."	15 seconds ago	Up 14 seconds	0.0.0.0:82→80/tcp, ::: 82→80/tcp	app2
2bb2db53e03e	nginx	"/docker-entrypoint..."	24 seconds ago	Up 22 seconds	0.0.0.0:81→80/tcp, ::: 81→80/tcp	app1

```
[root@ip-10-0-12-218 ec2-user]#
```

Autoría propia.

Figura 16. Instancias basadas en la AMI Base Docker.



Request to manage tags has succeeded.

Instances (2) info

Find Instance by attribute or tag (case-sensitive)

environment = prod Clear filters

Last updated 1 minute ago Connect Instance state

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	Server2Linux	i-0b527ddc34c9fd66b	Running	t2.micro	2/2 checks passed	View alarms
<input type="checkbox"/>	Server4LinuxPublico	i-09ef48974b99a908	Running	t2.micro	2/2 checks passed	View alarms

Account ID: 3569-9497-1157

Account

Organization

Service Quotas

Billing and Cost Management

Security credentials

Turn on multi-session support

Sign out

Autoría propia.

Figura 17. Primera instancia basada en la AMI Base Docker.

Instance summary for i-0b527ddc34c9fd66b (Server2Linux) Info

Updated 7 minutes ago

Instance ID
i-0b527ddc34c9fd66b

IPv6 address
-

Hostname type
IP name: ip-10-0-12-218.ec2.internal

Answer private resource DNS name
-

Auto-assigned IP address
54.236.55.37 [Public IP]

IAM Role
-

IMDSv2
Required

Operator
-

Public IPv4 address
54.236.55.37 | open address

Instance state
Running

Private IP DNS name (IPv4 only)
ip-10-0-12-218.ec2.internal

Instance type
t2.micro

VPC ID
vpc-0c12bb8b483fad30 (seminario-vpc)

Subnet ID
subnet-00ca4d89fd80957 (seminario-subnet-public1-us-east-1a)

Instance ARN
arn:aws:ec2:us-east-1:356994971157:instance/i-0b527ddc34c9fd66b

Private IPv4 addresses
10.0.12.218

Public DNS
ec2-54-236-55-37.compute-

Elastic IP addresses
-

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Auto Scaling Group name
-

Managed
false

Account ID
3569-9497-1157

Account
Organization
Service Quotas
Billing and Cost Management
Security credentials

Turn on multi-session support

Sign out

Autoría propia.

Figura 18. Segunda instancia basada en la AMI Base Docker.

Instance summary for i-09ef48974bf9aff08 (Server4LinuxPublico) [Info](#)

Updated less than a minute ago

Instance ID
i-09ef48974bf9aff08

IPv6 address
-

Hostname type
IP name: ip-10-0-9-91.ec2.internal

Answer private resource DNS name
-

Auto-assigned IP address
3.218.208.49 [Public IP]

IAM Role
-

IMDSv2
Optional
⚠ EC2 recommends setting IMDSv2 to required | [Learn more](#)

Public IPv4 address
3.218.208.49 | [open address](#)

Instance state
Running

Private IP DNS name (IPv4 only)
ip-10-0-9-91.ec2.internal

Instance type
t2.micro

VPC ID
vpc-0c12bb8b8483fad30 [seminario-vpc]

Subnet ID
subnet-00cca4d89fd80967 [seminario-subnet-public1-us-east-1a]

Instance ARN
arn:aws:ec2:us-east-1:356994971157:instance/i-09ef48974bf9aff08

Private IPv4 addresses
10.0.9.91

Public DNS
ec2-3-218-208-49.compute-

Elastic IP addresses
-

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | [Learn more](#)

Auto Scaling Group name
-

Managed
false

Account ID: 3569-9497-1157

Account
Organization
Service Quotas
Billing and Cost Management
Security credentials

Turn on multi-session support

Sign out

Autoría propia.

Figura 19. Proxy Reverse a Contenedores Docker

```
[root@ip-10-0-12-218 ec2-user]# systemctl status nginx.service
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
   Active: active (running) since Sun 2025-07-20 21:20:46 UTC; 2h 46min ago
     Process: 736780 ExecStartPre=/usr/bin/mkfifo -f /run/nginx.pid (code=exited, status=0/SUCCESS)
     Process: 736781 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
     Process: 736782 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 736783 (nginx)
     Tasks: 2 (Limit: 1111)
   Memory: 2.0M
     CPU: 176ms
   CGroup: /system.slice/nginx.service
           └─736783 "nginx: master process /usr/sbin/nginx"
             └─736784 "nginx: worker process"

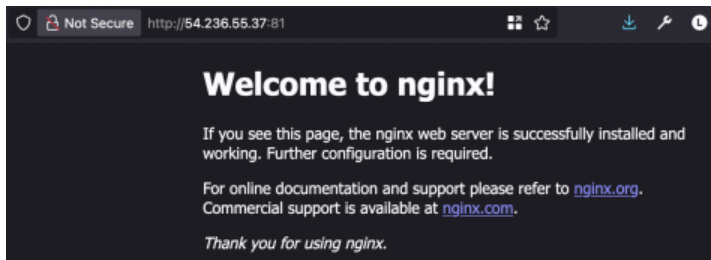
Jul 20 21:20:46 ip-10-0-12-218.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Jul 20 21:20:46 ip-10-0-12-218.ec2.internal nginx[736781]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Jul 20 21:20:46 ip-10-0-12-218.ec2.internal nginx[736781]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Jul 20 21:20:46 ip-10-0-12-218.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[root@ip-10-0-12-218 ec2-user]# cat /etc/nginx/nginx.conf
events {}

http {
    upstream seminario {
        server localhost:81;
        server localhost:82;
        server localhost:83;
    }

    server {
        listen 80;
        server_name nginx;
        location / {
            proxy_pass http://seminario;
        }
    }
}
```

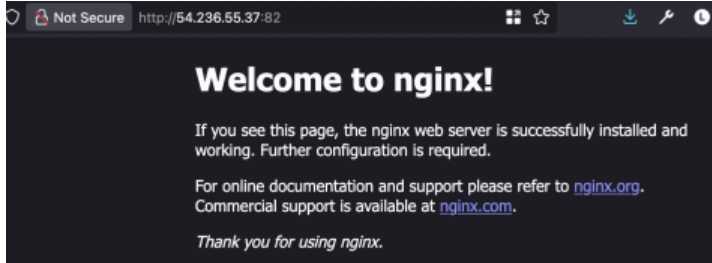
Autoría propia.

Figura 20. Conectividad Contenedor 1



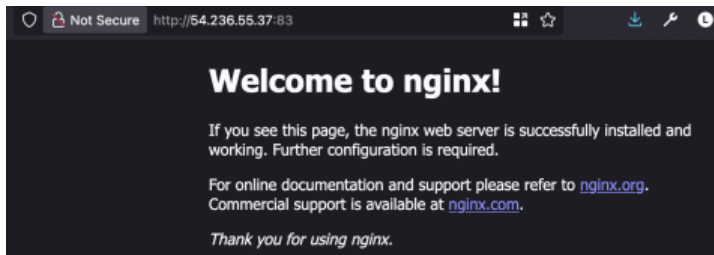
Autoría propia.

Figura 21. Conectividad Contenedor 2



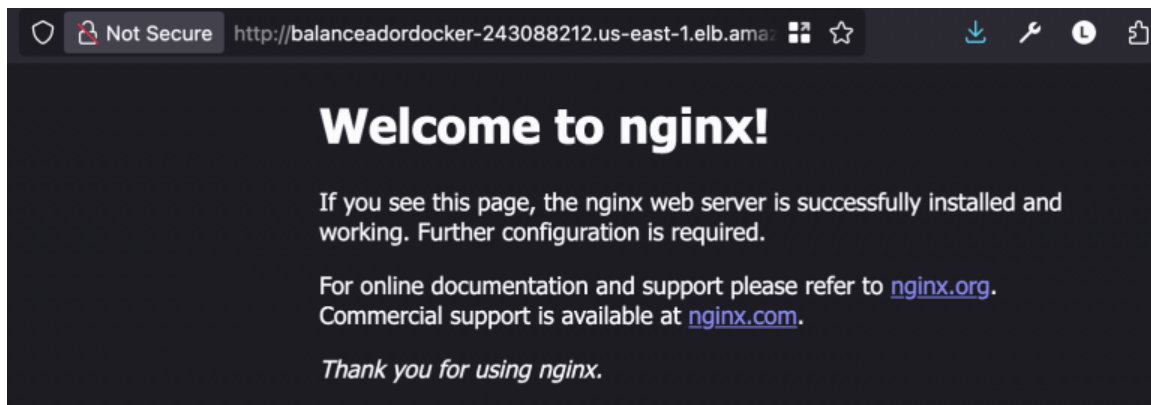
Autoría propia.

Figura 22. Conectividad Contenedor 3



Autoría propia.

Figura 23. Conectividad DNS Load Balancer



Autoría propia.

Figura 24. Configuración Auto Scaling

AutoScalingSeminario

AutoScalingSeminario Capacity overview

ami:aws:autoscaling:us-east-1:3:56994971157:autoScalingGroup:3b199f52-8c48-4864-8cb7-cbc7c49e02a3:autoScalingGroupName/AutoScalingSeminario

Desired capacity	Scaling limits (Min - Max)	Desired capacity type	Status
1	1 - 5	Units (number of instances)	-

Date created
Sun Jul 20 2025 13:46:05 GMT-0400 (Eastern Daylight Saving Time)

Details Integrations - new Automatic scaling Instance management Instance refresh Activity Monitoring

Launch template

Launch template
i-0358fc1db711b55b8
PlantillaServerLinux

Version
Default

Description
PlantillaServerLinux

View details in the launch template console

AMI ID	Instance type	Owner
ami-04b1b45b7a52f30e0	t2.micro	aws:iam::356994971157:root
Security groups	Security group IDs	Create time
-	sg-0c145aa5e4a470e2f	Sun Jul 20 2025 13:37:28 GMT-0400 (Eastern Daylight Saving Time)
Storage (volumes)	Key pair name	Request Spot Instances
-	Server2Linux	No

Network

Availability Zones	Subnet ID	Availability Zone distribution
use1-az1 (us-east-1a) use1-az2 (us-east-1b)	subnet-0977cb6a2999f73f0 subnet-029a55f0d06e4eb82	Balanced best effort

Account: 3569-9497-1157
Organization
Service Quotas
Billing and Cost Management
Security credentials
Turn on multi-session support
Sign out

Autoría propia.

Figura 25. Validación Auto Scaling

The screenshot displays the AWS Management Console interface. At the top, a green notification bar states: "Successfully initiated termination (deletion) of i-0e6a52362f0914127". Below this, the "Instances (1/9)" section is visible, featuring a search bar and a table of instances. The table includes columns for Name, Instance ID, and Instance state. One instance, "Server1Windows" (ID: i-0f0802d391764082f), is in a "Running" state. Another instance, "Server2Linux" (ID: i-0b527ddc34c9fd66b), is also "Running". A third instance (ID: i-0f8ec98808bc2a160) is "Terminated". The instance being terminated (ID: i-0e6a52362f0914127) is also shown as "Terminated".

Overlaid on the console is a dark terminal window showing system statistics. The top part of the terminal displays:


```

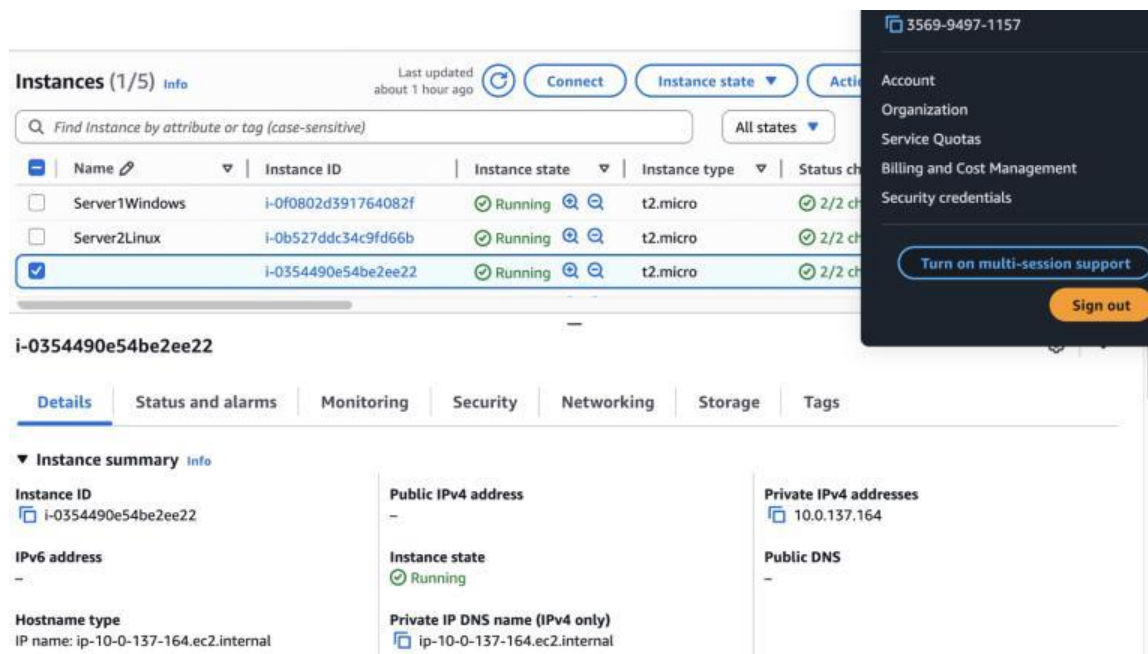
    MiB Mem : 949.4 total, 60
    MiB Swap: 0.0 total,
    top - 19:27:40 up 1:06, 1 use
    Tasks: 100 total, 2 running,
    %Cpu(s):100.0 us, 0.0 sy, 0.0
    MiB Mem : 949.4 total, 60
    MiB Swap: 0.0 total,
    
```

 Below this is a table with the following header and data:

PID	USER	PR	NI	VIR
3593	ec2-user	20	0	22403
3594	ec2-user	20	0	22401
1	root	20	0	10680
2	root	20	0	
3	root	0	-20	
4	root	0	-20	
5	root	0	-20	
6	root	0	-20	
8	root	0	-20	
10	root	0	-20	
11	root	20	0	
12	root	20	0	
13	root	20	0	

Autoría propia.

Figura 26. Validación Downscaling



The screenshot displays the AWS Management Console interface for EC2 instances. At the top, there is a header for 'Instances (1/5)' with a search bar and a filter set to 'All states'. Below this is a table listing three instances: 'Server1Windows', 'Server2Linux', and 'i-0354490e54be2ee22'. The third instance is selected. A user menu overlay is visible on the right side, showing the account ID '3569-9497-1157' and options like 'Account', 'Organization', 'Service Quotas', 'Billing and Cost Management', 'Security credentials', 'Turn on multi-session support', and 'Sign out'.

Name	Instance ID	Instance state	Instance type	Status checks
Server1Windows	i-0f0802d391764082f	Running	t2.micro	2/2 checks successful
Server2Linux	i-0b527ddc34c9fd66b	Running	t2.micro	2/2 checks successful
i-0354490e54be2ee22	i-0354490e54be2ee22	Running	t2.micro	2/2 checks successful

i-0354490e54be2ee22

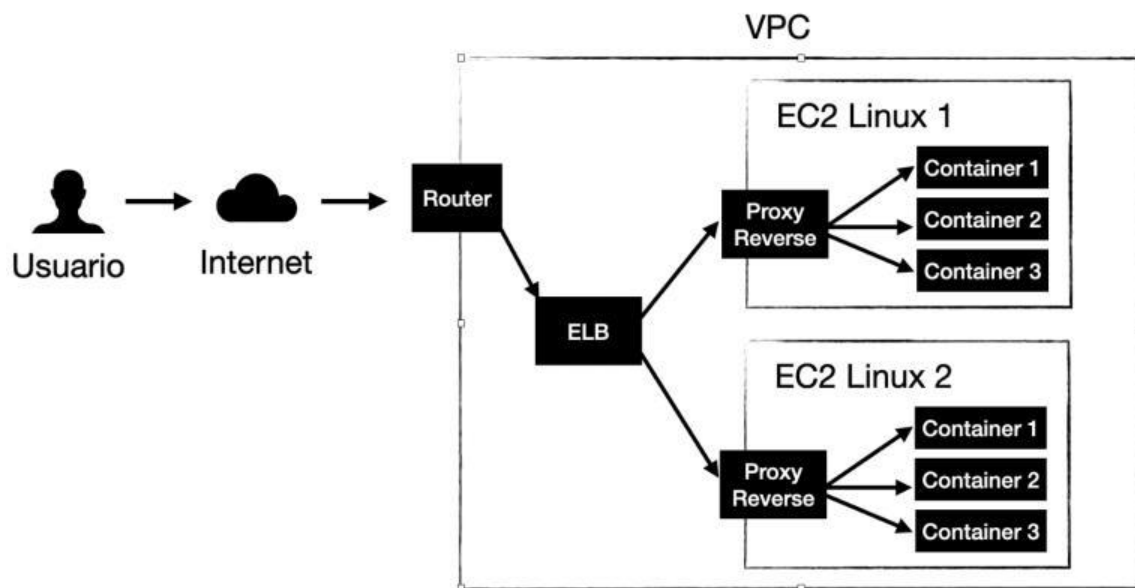
Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance summary

Instance ID i-0354490e54be2ee22	Public IPv4 address -	Private IPv4 addresses 10.0.137.164
IPv6 address -	Instance state Running	Public DNS -
Hostname type IP name: ip-10-0-137-164.ec2.internal	Private IP DNS name (IPv4 only) ip-10-0-137-164.ec2.internal	

Autoría propia.

Figura 27. Diagrama Seguridad, alta disponibilidad y escalabilidad de las aplicaciones.



Autoría propia.

Referencias

Amazon Web Services. (s.f.) Documentación de AWS. Recuperado el 20 de julio de 2005, de https://docs.aws.amazon.com/es_es/

Seminario AWS (2025), Seminario Opción-Grado AWS, Seguridad, Alta disponibilidad y escalabilidad de las aplicaciones en AWS. Juan Pablo Berrio López