



**TRABAJO DE GRADO
Seminario-Diplomado.**

Implementación de red en AWS con Docker y balanceo de carga

Corporación Universitaria Remington.
Facultad de ingeniería
Ingeniería de Sistemas

Luddy Mileny Puerta Ruiz
Juan Sebastian Forero Salamanaca
Docente Juan Pablo Berrio Lopez
Seminario de AWS
2025

Tabla de Contenidos

Resumen.....	3
Marco conceptual y contextual	4
Desarrollo e implementación del aprendizaje.....	6
Implementación de una red en AWS con servidores Windows y Linux accesibles desde Internet	6
Implementación de Servicios de Docker con pruebas de estrés	32
Conclusiones	41
Referencias.....	41

Resumen

Dentro de este proyecto vamos a ver las configuraciones para poder realizar la creación de un VPC, " ya que es el primer elemento que necesitamos crear, que es VPC en terminología de AWS, que es nuestro centro de datos virtual. (Guijarro Olivares, 2019), la Implementación de una red en AWS con servidores Windows y Linux accesibles desde Internet con 1 subred pública para Linux EC2 y 1 Subred pública para Windows EC2 con sus grupos de seguridad separados y la tabla de rutas con acceso a Internet Gateway (IGW) asociado a la VPC; además también veremos la implementación de los servicios de Docker con instancias S2, esto es necesario, ya que vamos a comprender el servicio de Docker y la utilización de los servicios autoadministrados de AWS esto con el fin de que nos permita administrar contenedores; con el proceso de dockerización allí vamos a encontrar Docker corriendo dentro de una instancia de Linux.

Durante el proceso de dockerización, veremos cómo Docker opera dentro de una instancia de Linux. Allí, instalaremos y ejecutaremos Docker en una instancia EC2, lo cual nos brindará la oportunidad de experimentar con la dockerización de servicios. Esto implica empaquetar aplicaciones dentro de contenedores, facilitando así su despliegue, movilidad y gestión

Además, con la implementación de pruebas de estrés sobre la infraestructura, vamos a poner a funcionar varios contenedores para ver el comportamiento en rendimiento tanto de la instancia como de los mismos contenedores, con las pruebas de estrés vamos a poder ver cuanto se está usando en CPU y RAM y poder ver el aumento del uso de los recursos y como saturar el servicio.

En resumen, este proyecto nos capacitará para construir una red en AWS desde cero, utilizar instancias EC2 con diversos sistemas operativos, trabajar con Docker en la nube y evaluar si nuestra infraestructura está preparada para funcionar bajo las diferentes pruebas de estrés, todo ello con un enfoque práctico y aplicable a situaciones reales.

Palabras clave

Instancias AWS (EC2), creación de VPC, servicios de Docker, pruebas de estrés, Implementación de red AWS, Nginx, Apache,

Marco conceptual y contextual

Este informe se realizó durante un seminario de AWS sobre computación en la nube, La computación en la nube se refiere a la entrega de servicios de computación a través de la Internet, o “la nube”. En lugar de utilizar servidores locales o computadoras personales para almacenar, gestionar y procesar datos, la computación en la nube permite acceder a estos recursos desde centros de datos remotos gestionados por proveedores de servicios en la nube (IMECAF, 2024)

Además de acuerdo al objetivo fue aprender a crear y configurar una red completa en AWS utilizando servicios disponibles en la plataforma para esto se trabajó con la creación de una VPC que incluye dos subredes públicas, una para una instancia con sistema operativo Linux y otra para una con Windows ambas fueron configuradas para ser accesibles desde Internet

Se exploró el uso de Docker y Apache como herramientas para ejecutar aplicaciones en servidores de forma óptima, se instaló Docker en una instancia EC2 con Linux y se usaron contenedores para alojar servicios web.

También se realizaron pruebas de estrés que por lo cual se define como una sencilla utilidad de línea de comandos diseñada para sobrecargar los recursos del sistema, como la CPU, la memoria RAM y las operaciones de E/S. Simula condiciones extremas y permite ver cómo responde la aplicación, el contenedor o incluso el sistema completo a la escasez de recursos, (Serverspace, n.d.) esto sirvió para evaluar como funciona la instancia bajo la prueba de estrés.

Este trabajo fue útil para aplicar todos los conceptos vistos en el seminario y para conocer de forma práctica cómo se puede montar una infraestructura en la nube con servicios

Desarrollo e implementación del aprendizaje

Implementación de una red en AWS con servidores Windows y Linux accesibles desde Internet

Objetivo General

Diseñar, desplegar y documentar una red en AWS que incluya dos instancias EC2 (una Windows y una Linux), asegurando su accesibilidad pública, conectividad entre ellas y la instancia de un servidor web funcional en cada instancia.

Parte 1: Documentación Técnica

1. Diagrama de Arquitectura

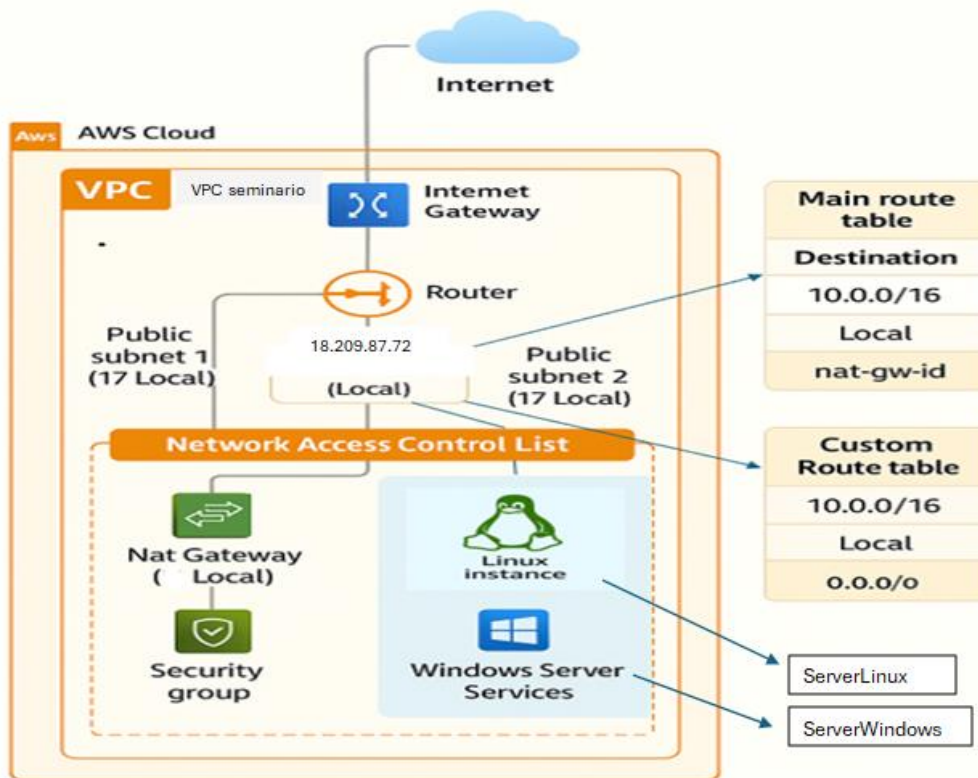


Ilustración 1- Esquema de la Arquitectura

Componentes del Diagrama:

- VPC personalizada
- 1 Subred pública para linux EC2
- 1 Subred pública para Windows EC2
- Internet Gateway (IGW) asociado a la VPC
- Tabla de rutas con acceso a IGW
- Instancia EC2 (Amazon Linux 2023)
- Instancia EC2 (Windows Server 2016 Base)
- Grupo de seguridad separados

2. Descripción de la Arquitectura

Se creó una VPC personalizada con dos subredes públicas: una para la instancia Linux nombrada ServerLinux y otra para la instancia Windows nombrada ServerWindows. Ambas subredes están asociadas a una tabla de rutas que permite acceso a Internet mediante un Internet Gateway (IGW). Las instancias EC2 están protegidas por grupos de seguridad que permiten acceso RDP (puerto 3389) para Windows, SSH (puerto 22) para Linux y HTTP (puerto 80) para ambos servidores web. La com(IMECAF, 2024)

3. Tipo de Instancias Usadas

- Linux: AMI de Amazon Linux 2023 (Free Tier elegible)
- Windows: Windows Server 2016 Base (Free Tier elegible)

Justificación: Se utilizaron versiones compatibles con Free Tier para reducir costos. Ambas versiones ofrecen facilidad de configuración y amplia documentación en AWS.

4. Justificación de las Configuraciones de Red

- **VPC Personalizada:** Permite control total sobre la segmentación y seguridad.
- **Subredes Públicas:** Requeridas para que las instancias tengan acceso directo desde Internet.
- **Internet Gateway (IGW):** Necesario para que las instancias EC2 tengan acceso a Internet.
- **Grupos de Seguridad:** Configurados para permitir únicamente el tráfico necesario (RDP, SSH, HTTP), minimizando riesgos de seguridad.

5. Configuraciones Realizadas

Pasos para Crear Instancias EC2

1. Crear una nueva VPC con subredes públicas.
2. Asociar una tabla de rutas con acceso a IGW.
3. Crear y asociar un Internet Gateway a la VPC.
4. Lanzar una instancia EC2 con Amazon Linux 2023 y otra con Windows Server 2016.
5. Asignar IPs públicas durante el lanzamiento.
6. Crear pares de claves (.pem) para el acceso SSH y RDP.

EVIDENCIA VPC (Creación):

Crear VPC Información

Una VPC es una parte aislada de la nube de AWS que contiene objetos de AWS,

Configuración de la VPC

Recursos que se van a crear Información
Cree únicamente el recurso de VPC o la VPC y otros recursos de red.

Solo la VPC VPC y más

Generación automática de etiquetas de nombre Información
Ingrese un valor para la etiqueta Nombre. Este valor se utilizará para generar automáticamente etiquetas Nombre para todos los recursos de la VPC.

Generar automáticamente
VPC seminario-vpc

Bloque de CIDR IPv4 Información
Determine la IP inicial y el tamaño de la VPC mediante la notación CIDR.

10.0.0.0/16 65,536 IPs
El tamaño del bloque CIDR debe estar entre /16 y /28.

Bloque de CIDR IPv6 Información
 Sin bloque de CIDR IPv6
 Bloque de CIDR IPv6 proporcionado por Amazon

Tenencia Información
Predeterminado

Número de zonas de disponibilidad (AZ) Información
Elija la cantidad de zonas de disponibilidad en las que desea aprovisionar subredes. Le recomendamos que tenga al menos dos para incrementar la

Ilustración 2 – Creación VPC

Número de zonas de disponibilidad (AZ) [Información](#)

Elija la cantidad de zonas de disponibilidad en las que desea aprovisionar subredes. Le recomendamos que tenga al menos dos para incrementar la disponibilidad.

1 2 3

► Personalizar las zonas de disponibilidad

Cantidad de subredes públicas [Información](#)

La cantidad de subredes públicas que se van a agregar a la VPC. Utilice subredes públicas para las aplicaciones web que deban ser accesibles públicamente a través de Internet.

0 2

Cantidada de subredes privadas [Información](#)

La cantidad de subredes privadas que se van a agregar a la VPC. Utilice subredes privadas para proteger los recursos del backend que no necesitan acceso público.

0 2 4

► Personalizar bloques de CIDR de subredes

Gateways NAT (\$) [Información](#)

Elija el número de zonas de disponibilidad (AZ) en las que crear gateway NAT. Tenga en cuenta que hay un cargo por cada puertaa de enlace NAT.

Ninguna En 1 AZ 1 por zona de disponibilidad

Puntos de enlace de la VPC [Información](#)

Los puntos de enlace pueden ayudar a reducir los cargos de gateway NAT y mejorar la seguridad gracias a la posibilidad de acceder a S3 directamente desde la VPC. De forma predeterminada, se utiliza una política de acceso completo. Puede personalizar esta política en cualquier momento.

Ninguna Gateway de S3

Ilustración 3 –Detalle de Vpc

The screenshot shows the AWS Management Console interface for the VPC service. On the left, there is a navigation menu with categories like 'Panel de VPC', 'Nube virtual privada', 'Seguridad', and 'PrivateLink y Lattice'. The main content area is titled 'Sus VPC (1/2) Información' and contains a table of VPCs. The table has columns for Name, ID de la VPC, Estado, Bloquear el..., and CIDR IPv4. One VPC, 'VPC seminario-vpc', is selected and highlighted. Below the table, the details for 'vpc-05f438cfe38fe7b3d / VPC seminario-vpc' are shown, including tabs for Detalles, Mapa de recursos, CIDR, Registros de flujo, Etiquetas, and Integraciones. The 'Detalles' tab is active, displaying various configuration parameters.

Name	ID de la VPC	Estado	Bloquear el ...	CIDR IPv4
-	vpc-080f07470eae8ee89	Available	Desactivado	172.31.0.0/16
<input checked="" type="checkbox"/> VPC seminario-vpc	vpc-05f438cfe38fe7b3d	Available	Desactivado	10.0.0.0/16

vpc-05f438cfe38fe7b3d / VPC seminario-vpc

[Detalles](#) | [Mapa de recursos](#) | [CIDR](#) | [Registros de flujo](#) | [Etiquetas](#) | [Integraciones](#)

Detalles

<p>ID de la VPC <input type="checkbox"/> vpc-05f438cfe38fe7b3d</p> <p>Resolución de DNS Habilitado</p> <p>ACL de red principal acl-06b93fba300076269</p> <p>CIDR IPv6 (grupo de bordes de red) -</p>	<p>Estado ✔ Available</p> <p>Tenencia default</p> <p>VPC predeterminada No</p> <p>Métricas de uso de direcciones de red Desactivado</p>
--	--

Ilustración 4 – Comprobación de VPC creado y disponible

TABLA ENRUTAMIENTO:

Tablas de enrutamiento (1/5) [Información](#)

Find route tables by attribute or tag

<input type="checkbox"/>	Name	ID de tabla de enrutam...	Asociaciones de subre...	Asociaciones de...	Princ...	VPC	IT
<input checked="" type="checkbox"/>	VPC seminario-rtb-public	rtb-064fc841c15d5c332	-	-	Sí	vpc-080f07470eae8ee89	0:
<input checked="" type="checkbox"/>	VPC seminario-rtb-public	rtb-0ae903ada3bf38efd	2 subredes	-	No	vpc-05f438cfe38fe7b3d VPC s...	0:
<input type="checkbox"/>	-	rtb-0b7e1d855a13ed772	-	-	Sí	vpc-05f438cfe38fe7b3d VPC s...	0:
<input type="checkbox"/>	VPC seminario-rtb-private2-us-east-1b	rtb-0919e6ba2288e584d	subnet-0a067b5b3bd1a2...	-	No	vpc-05f438cfe38fe7b3d VPC s...	0:
<input type="checkbox"/>	VPC seminario-rtb-private1-us-east-1a	rtb-0b4b3143a7f9cad4a	subnet-09d560c50127f7...	-	No	vpc-05f438cfe38fe7b3d VPC s...	0:

rtb-0ae903ada3bf38efd / VPC seminario-rtb-public

[Detalles](#) | [Rutas](#) | [Asociaciones de subredes](#) | [Asociaciones de borde](#) | [Propagación de rutas](#) | [Etiquetas](#)

Detalles

ID de tabla de enrutamiento <input checked="" type="checkbox"/> rtb-0ae903ada3bf38efd	Principal <input type="checkbox"/> No	Asociaciones de subredes explícitas 2 subredes
VPC vpc-05f438cfe38fe7b3d VPC seminario-vpc	ID de propietario <input checked="" type="checkbox"/> 033244264661	

Ilustración 5 – Tabla Enrutamiento

vpc-05f438cfe38fe7b3d / VPC seminario-vpc [Acciones](#)

Detalles [Información](#)

ID de la VPC <input checked="" type="checkbox"/> vpc-05f438cfe38fe7b3d	Estado <input checked="" type="checkbox"/> Available	Bloquear el acceso público <input type="checkbox"/> Desactivado	Nombres de host de DNS Habilitado
Resolución de DNS Habilitado	Tenencia default	Conjunto de opciones de DHCP dhcp-010b67d13978c2a1d	Tabla de enrutamiento principal rtb-0b7e1d855a13ed772
ACL de red principal acl-06b93fba300076269	VPC predeterminada No	CIDR IPv4 10.0.0.0/16	Grupo IPv6 -
CIDR IPv6 (grupo de bordes de red) -	Métricas de uso de direcciones de red Desactivado	Grupos de reglas del firewall de DNS de Route 53 Resolver	ID de propietario <input checked="" type="checkbox"/> 033244264661

[Mapa de recursos](#) | [CIDR](#) | [Registros de flujo](#) | [Etiquetas](#) | [Integraciones](#)

Mapa de recursos [Información](#)

Ilustración 6 – Detalle del enrutamiento

Grupos de Seguridad:

- **Windows EC2:** Puertos abiertos: 3389 (RDP), 80 (HTTP): se creó un grupo de seguridad nombrado SG-ServerWindows.
- **Linux EC2:** Puertos abiertos: 22 (SSH), 80 (HTTP): se creó un grupo de seguridad nombrado SG-ServerLinux2.

Asignación de IPs:

- Cada instancia tiene asignada una IP pública en la cual en la sección de asignación automática se activó la IP pública (Habilitar).
- IPs privadas asignadas automáticamente por la subred.

EVIDENCIAS INSTANCIAS:

Servidor Windows:

1. Crear instancia

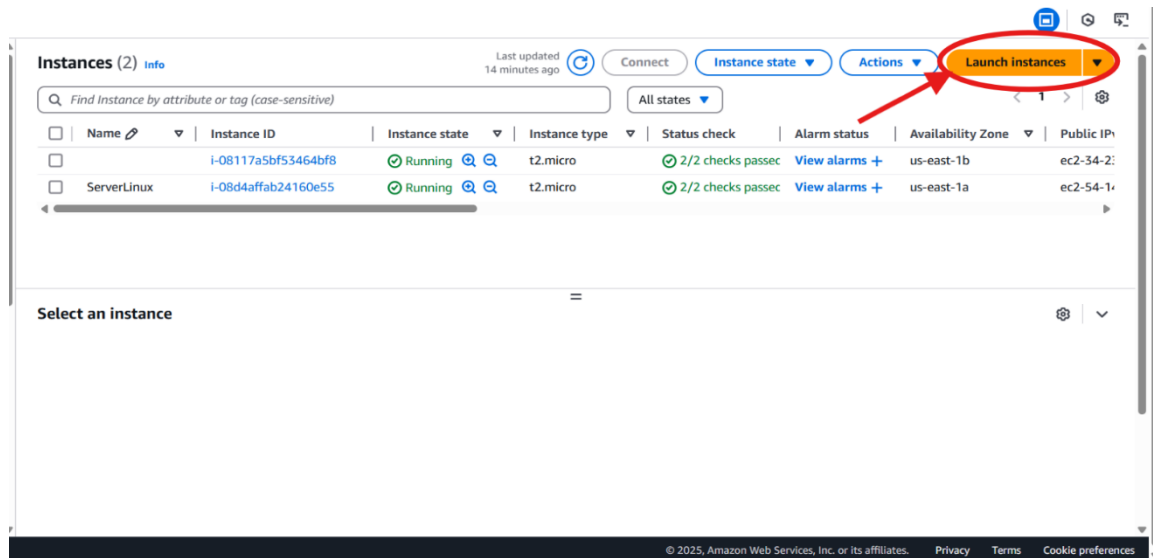


Ilustración 7 – Creación de instancia Windows

2. Se asigna un nombre y se elige el sistema Windows

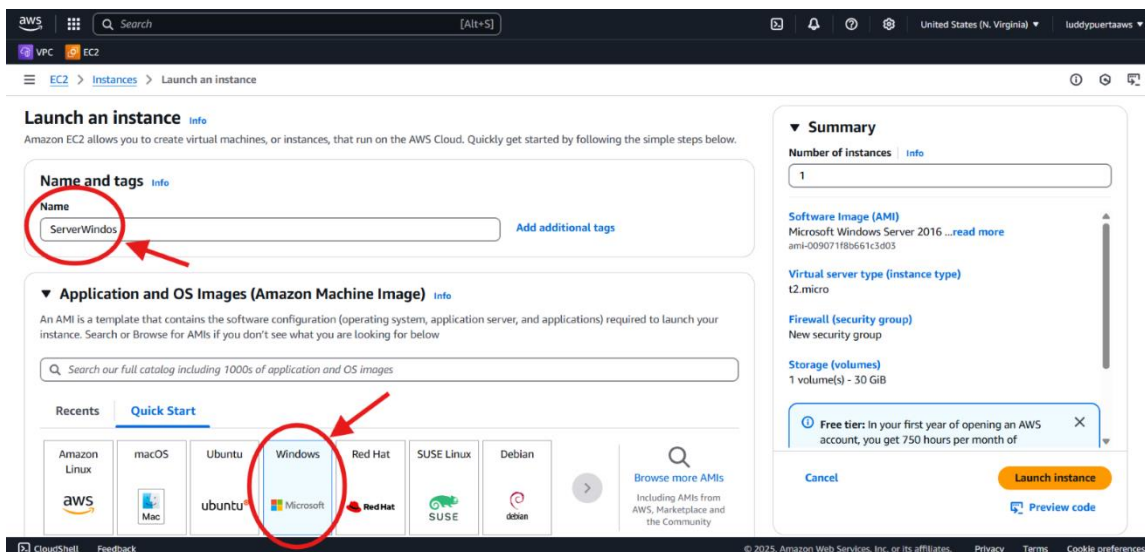


Ilustración 8 – Selección de la imagen Windows

3. Versión del sistema de la máquina y la capacidad que va a tener

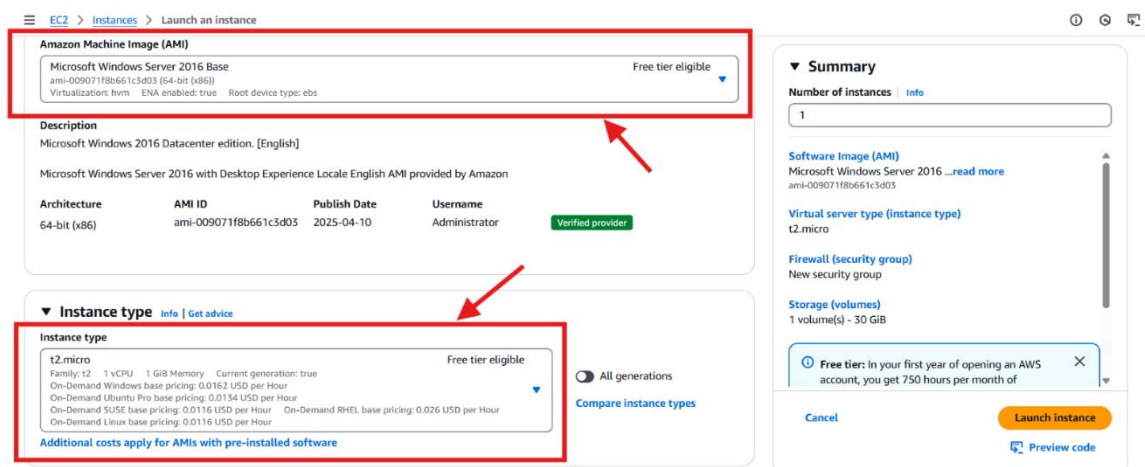


Ilustración 9 – Tipo de instancias

4. Se crea un par de claves

EC2 > Instances > Launch an instance

Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select Create new key pair

For Windows instances, you use a key pair to decrypt the administrator password. You then use the decrypted password to connect to your instance.

Network settings Info Edit

Network Info

vpc-060f07470eae8ee89

Subnet Info

No preference (Default subnet in any availability zone)

Auto-assign public IP Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

We'll create a new security group called 'launch-wizard-1' with the following rules:

Summary

Number of instances Info

1

Software Image (AMI)

Microsoft Windows Server 2016 ...[read more](#)

ami-009071f8b661c3d03

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 30 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of

Cancel Launch instance

[Preview code](#)

Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

ClaveServerWin

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA

RSA encrypted private and public key pair

ED25519

ED25519 encrypted private and public key pair (Not supported for Windows instances)

Private key file format

.pem

For use with OpenSSH

.ppk

For use with PuTTY

When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Cancel Create key pair

Ilustración 10 – Creación de par de llaves

5. Descarga Archivo (.pem)

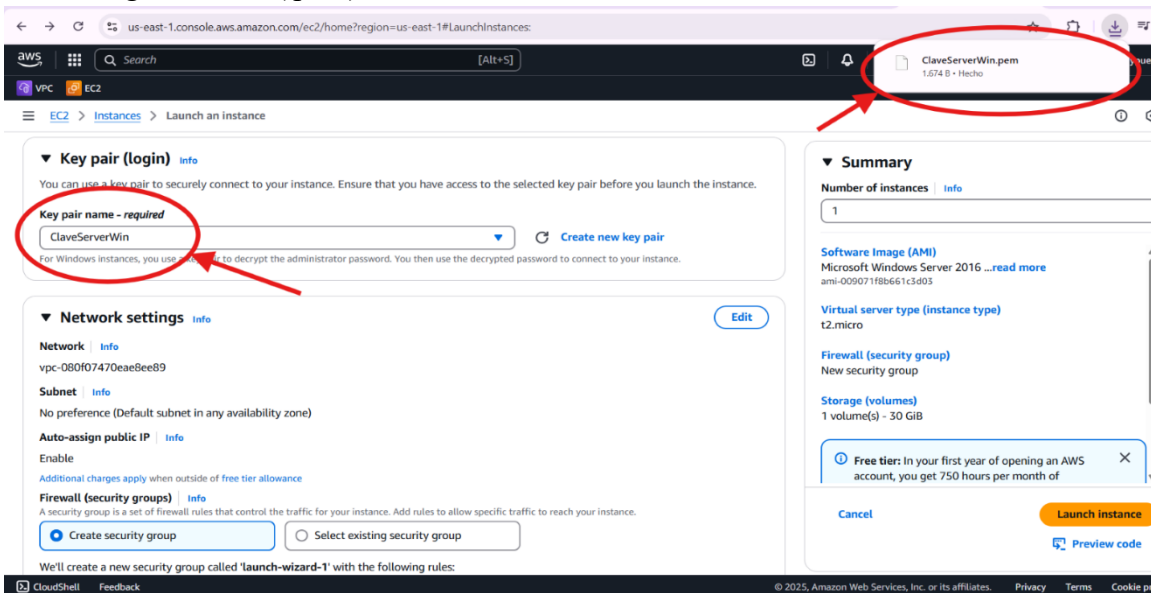


Ilustración 11 – Descarga de llaves

6. Editan Network settings y se lanza la instancia

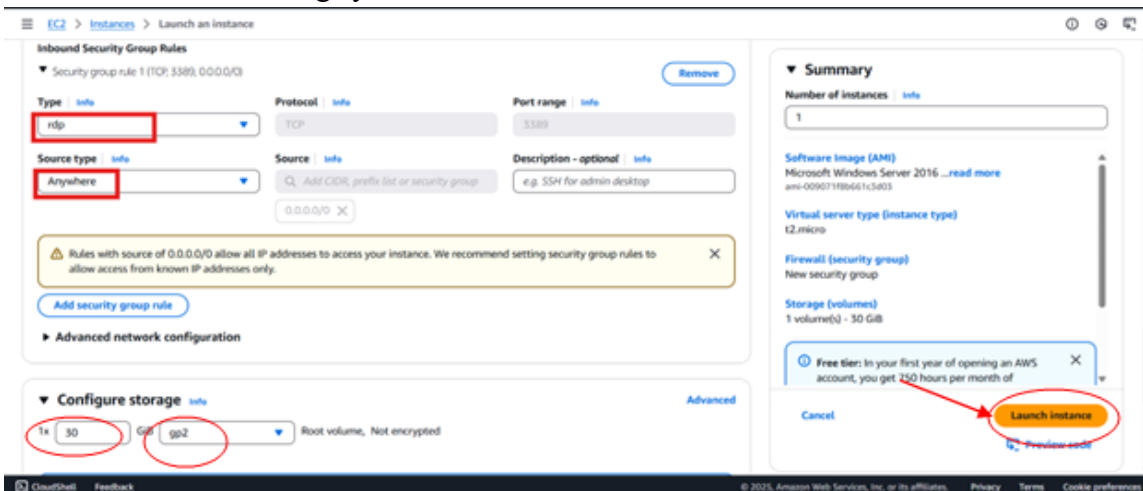


Ilustración 12 – Lanzamiento de instancias



Ilustración 13 – Creación de instancia exitosa

7. Se valida la Ip pública

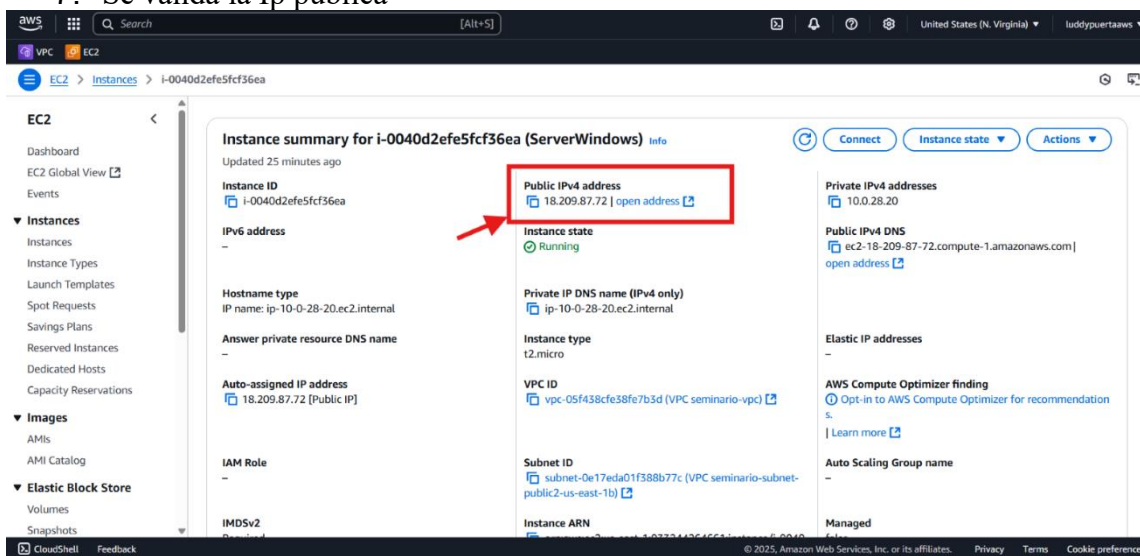


Ilustración 14 – Validación de IP pública

8. Obtenemos el password subiendo el archivo .pem

The screenshot shows the AWS Management Console interface for connecting to an EC2 instance. The breadcrumb navigation is **EC2 > Instances > i-0040d2efe5fcf36ea > Connect to instance**. The page title is **Connect to instance** with an info icon. Below the title, it says "Connect to your instance i-0040d2efe5fcf36ea (ServerWindows) using any of these options". There are three tabs: **Session Manager**, **RDP client** (selected and circled in red), and **EC2 serial console**. Under the **RDP client** tab, the **Connection Type** section has two options: "Connect using RDP client" (selected) and "Connect using Fleet Manager". Below this, there is a "Download remote desktop file" button. The "When prompted, connect to your instance using the following username and password:" section shows the **Public DNS** as `ec2-18-209-87-72.compute-1.amazonaws.com` and the **Username** as `Administrator`. The **Password** field is empty, and a red arrow points to the **Get password** button. At the bottom, there is a note: "If you've joined your instance to a directory, you can use your directory credentials to connect to your instance."

Ilustración 15 – Obtención de password

This screenshot is similar to the previous one, showing the same AWS Management Console interface. The **Username** dropdown menu is now highlighted with a red box and contains the value `Administrator`. The **Password** field is now populated with the generated password `nN-vLW*%qQGGdopTr6%@(IC!;Bw275r`, which is also highlighted with a red box. A red arrow points to the password field. The rest of the interface remains the same as in the previous screenshot.

Ilustración 16 – Generador de password

9. Nos conectamos a la máquina

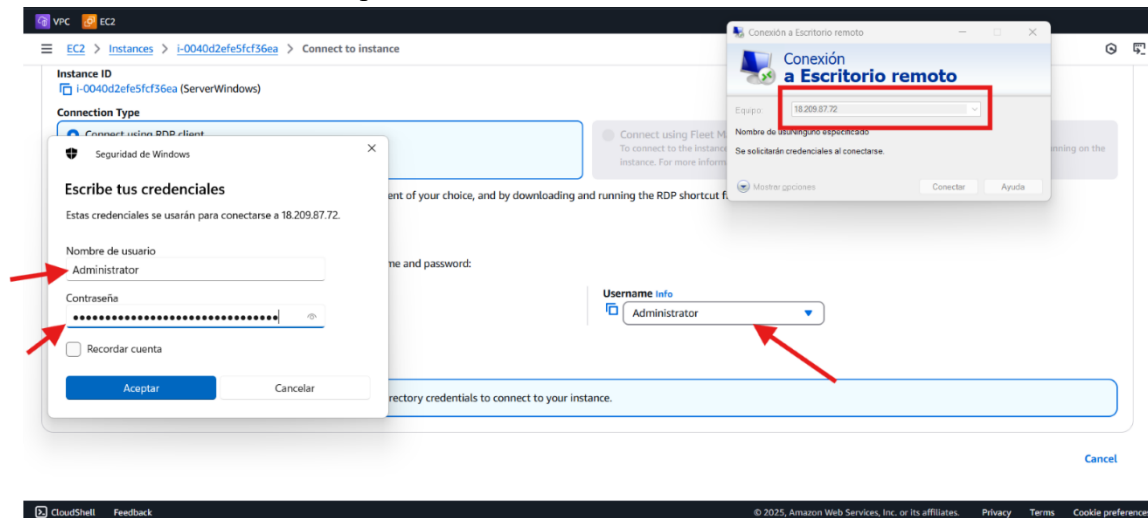


Ilustración 17 – Conexión remota

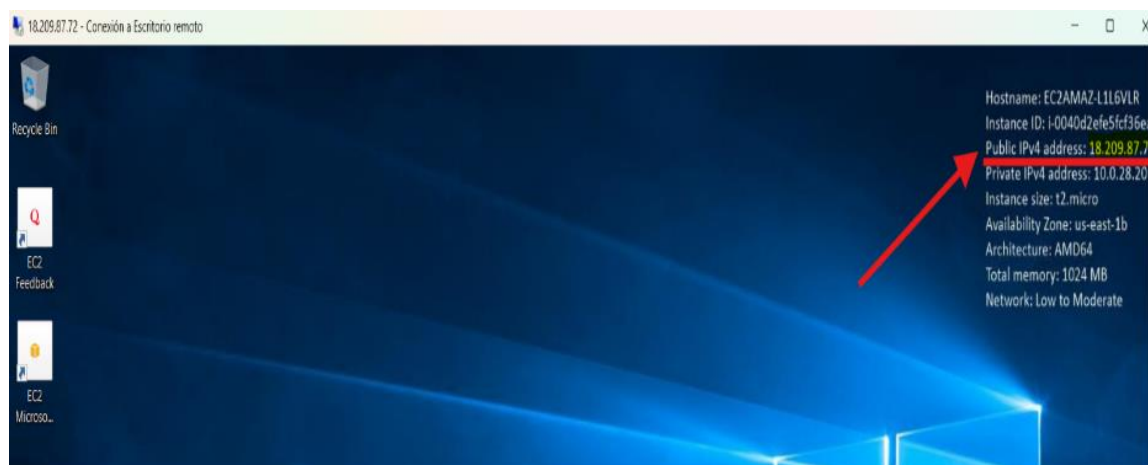


Ilustración 18 – Validación de conexión remota

10. Editamos roles de grupos de seguridad.

The screenshot shows the AWS Management Console interface for a security group named 'sg-0081515d7400c3c00 - SG-ServerWindows'. The left sidebar shows the navigation menu with 'Network & Security' expanded and 'Security Groups' selected. The main content area displays the details of the security group, including its name, ID, description, and VPC ID. Below the details, there are tabs for 'Inbound rules', 'Outbound rules', 'Sharing - new', 'VPC associations - new', and 'Tags'. The 'Inbound rules' tab is active, showing a table with one rule. The 'Edit inbound rules' button is circled in red.

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-0132b280cf07a82ed	IPv4	RDP	TCP	3389

Ilustración 19 – Configuración de roles

11. Creamos nuevo rol

The screenshot shows the 'Edit inbound rules' configuration page in the AWS Management Console. The page title is 'Edit inbound rules' and it includes a warning message: 'Rules with source of 0.0.0.0/0 or :::0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.' The configuration table shows two rules. The second rule is highlighted with a red box, showing 'Custom TCP' as the type, 'TCP' as the protocol, '80' as the port range, and 'Anyw...' as the source. The 'Add rule' button is circled in red, and the 'Save rules' button is also circled in red.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0132b280cf07a82ed	RDP	TCP	3389	Custom	
-	Custom TCP	TCP	80	Anyw...	

Ilustración 20 – Edición de reglas

12. Creamos nuevos roles

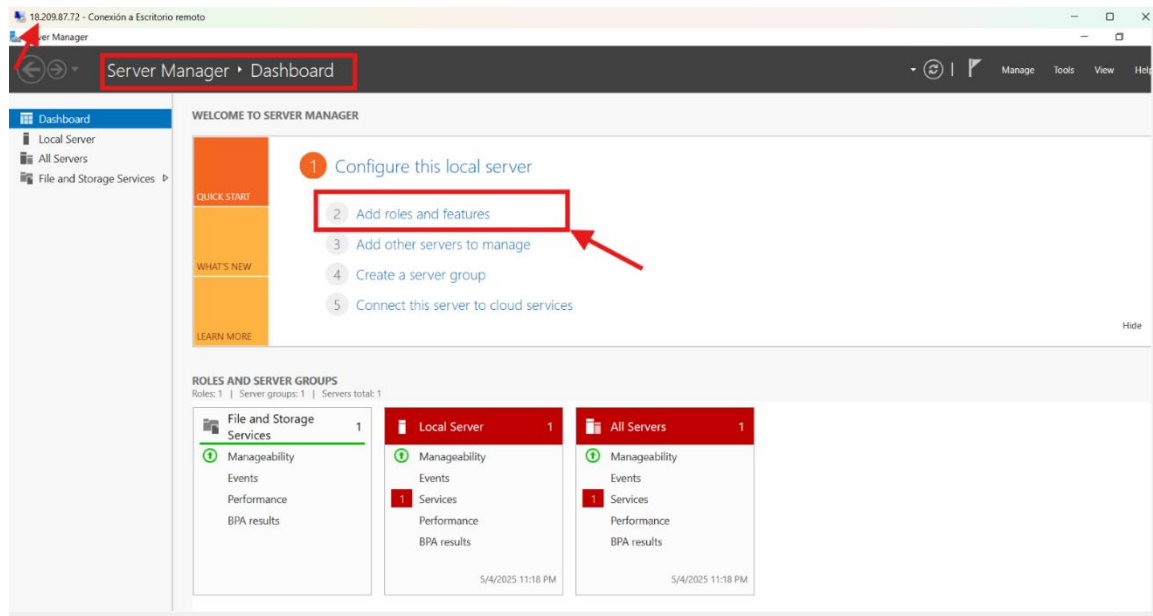


Ilustración 21 – Creación de nuevos roles

13. Se crea rol (IIS)

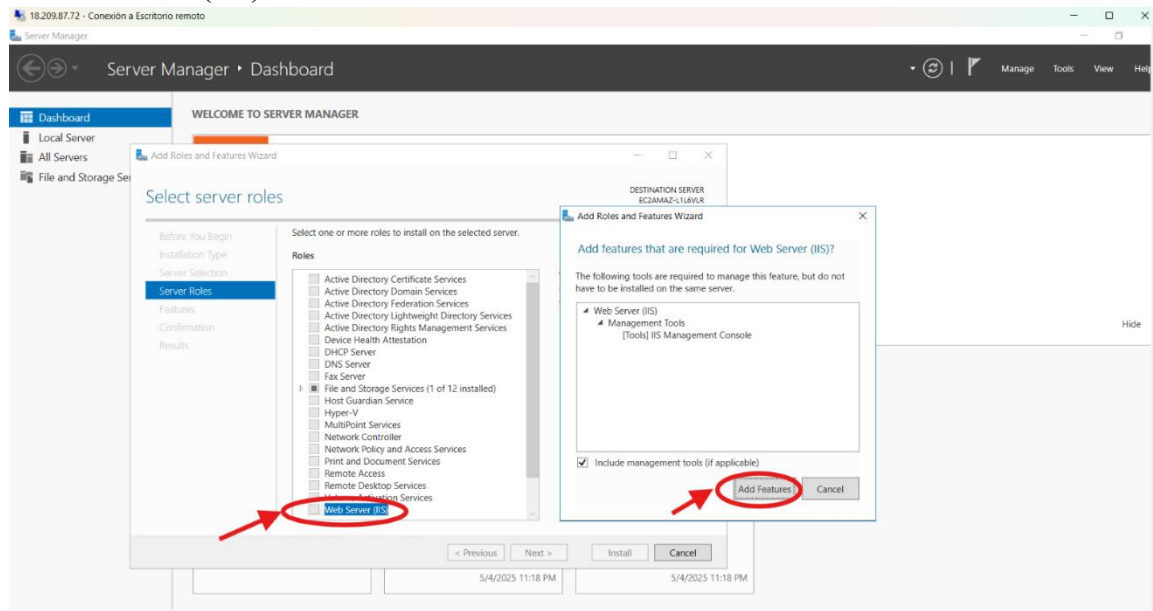


Ilustración 22 – Selección de roles

14. Se comprueba funcionamiento del servicio localhost

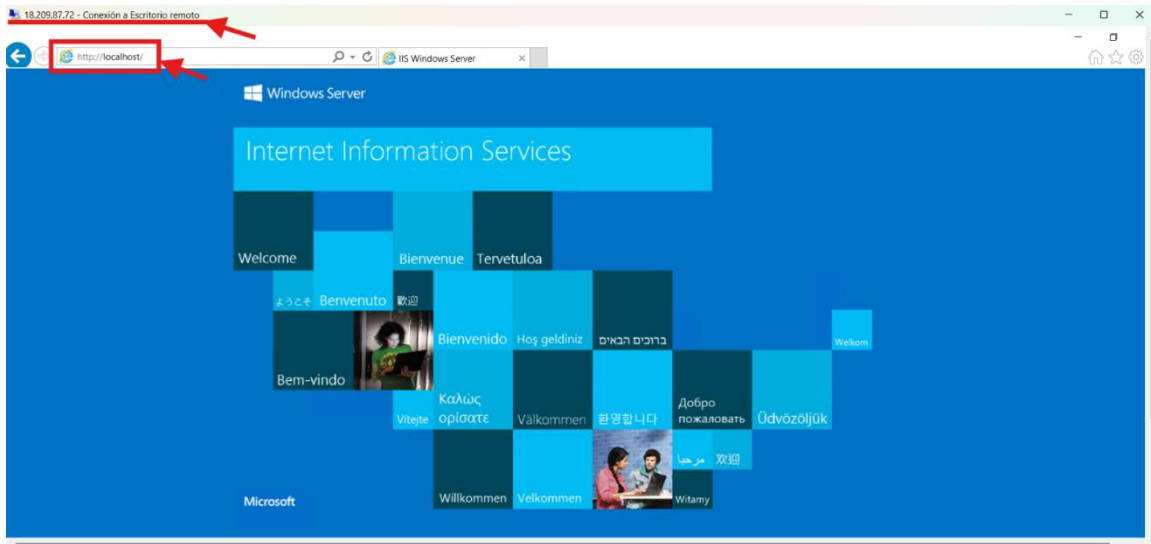


Ilustración 23 – Comprobación de funcionamiento del sistema Windows

15. Se comprueba funcionamiento del servicio con la dirección IP <http://18.209.87.72/>

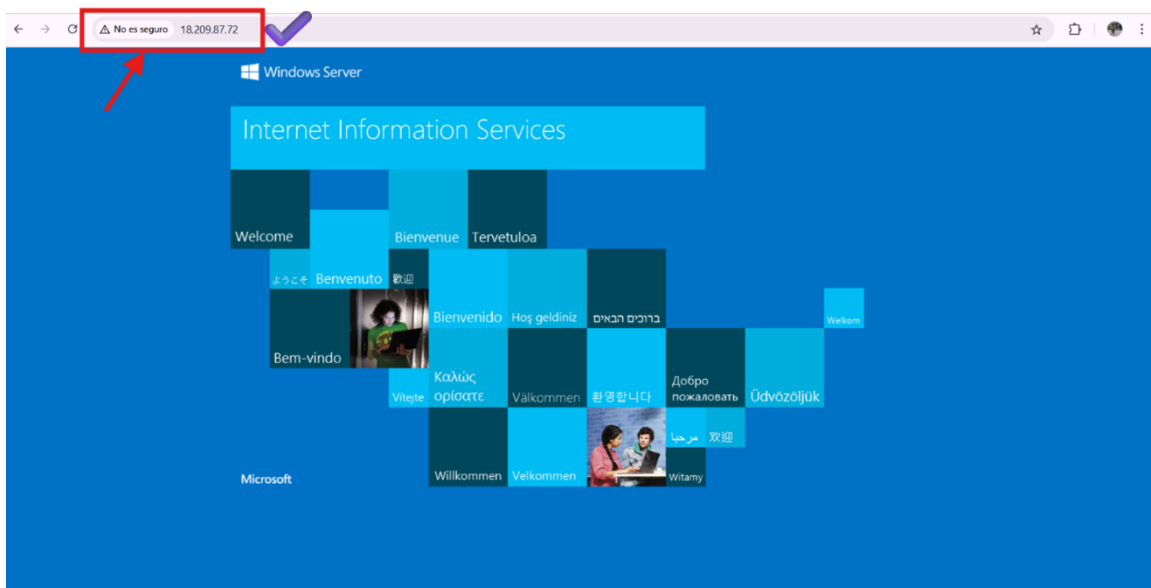


Ilustración 24 – Pruebas de dirección IP pública en Windows

Servidor Linux:

1. Se crea la instancia de Linux.

Lanzar una instancia Información

Amazon EC2 le permite crear máquinas virtuales, o instancias, que se ejecutan en la nube de AWS. Comience rápidamente siguiendo los sencillos pasos que se indican a continuación.

Nombre y etiquetas Información


Nombre


 [Agregar etiquetas adicionales](#)


▼ Imágenes de aplicaciones y sistemas operativos (Imagen de máquina de Amazon) Información


Una AMI es una plantilla que contiene la configuración de software (sistema operativo, servidor de aplicaciones y aplicaciones) necesaria para lanzar la instancia. Busque o examine las AMI si no ve lo que busca a continuación.


Recientes
Inicio rápido




















[Buscar más AMI](#)
Inclusión de AMI de AWS, Marketplace y la comunidad

Imágenes de máquina de Amazon (AMI)

AMI de Amazon Linux 2023
ami-0f88e80871fd81e91 (64 bits (x86), uefi-preferred) / ami-0bc72bd3b8ba0b59d (64 bits (Arm), uefi)
Virtualización: hvm Activado para ENA: true Tipo de dispositivo raíz: ebs

Apto para la capa gratuita ▼

Descripción

Amazon Linux 2023 es un sistema operativo moderno y de uso general basado en Linux que incluye 5 años de soporte a largo plazo. Está optimizado para AWS y diseñado para proporcionar un entorno de ejecución seguro, estable y de alto desempeño para desarrollar y ejecutar sus aplicaciones en la nube.

Amazon Linux 2023 AMI 2023.7.20250428.1 x86_64 HVM kernel-6.1

Arquitectura	Modo de arranque	ID de AMI	Fecha de publicación	Nombre de usuario	
64 bits (x86) ▼	uefi-preferred	ami-0f88e80871fd81e91	2025-04-30	ec2-user	Proveedor verificado

Ilustración 25 – Creación de instancia Linux

2. Se crea un par de claves

Crear par de claves ✕

Nombre del par de claves
Con los pares de claves es posible conectarse a la instancia de forma segura.

El nombre puede incluir hasta 255 caracteres ASCII. No puede incluir espacios al principio ni al final.

Tipo de par de claves


RSA
Par de claves pública y privada cifradas mediante RSA

ED25519
Par de claves privadas y públicas cifradas ED25519

Formato de archivo de clave privada

.pem
Para usar con OpenSSH

.ppk
Para usar con PuTTY

⚠ Cuando se le solicite, almacene la clave privada en un lugar seguro y accesible del equipo. **Lo necesitará más adelante para conectarse a la instancia.** [Más información](#) 

[Cancelar](#) [Crear par de claves](#)

Ilustración 26 – Creación de par de llaves Linux

- Se realiza la configuración de la red con el nombre del grupo de seguridad y se habilita la asignación automática de la ip pública.

▼ Configuraciones de red [Información](#)

VPC : obligatorio | [Información](#)

vpc-05f438cfe38fe7b3d (VPC seminario-vpc)
10.0.0.0/16

Subred | [Información](#)

subnet-08e17c9412ab91c71 **VPC seminario-subnet-public1-us-east-1a**
VPC: vpc-05f438cfe38fe7b3d Propietario: 033244264661 Zona de disponibilidad: us-east-1a
Tipo de zona: Zona de disponibilidad Direcciones IP disponibles: 4091 CIDR: 10.0.0.0/20

Asignar automáticamente la IP pública | [Información](#)

Habilitar

Se aplican cargos adicionales cuando no se cumplen los límites del nivel gratuito

Firewall (grupos de seguridad) | [Información](#)

Un grupo de seguridad es un conjunto de reglas de firewall que controlan el tráfico de la instancia. Agregue reglas para permitir que un tráfico específico llegue a la instancia.

Crear grupo de seguridad Seleccionar un grupo de seguridad existente

Nombre del grupo de seguridad - obligatorio

SG-ServerLinux2

Este grupo de seguridad se agregará a todas las interfaces de red. El nombre no se puede editar después de crear el grupo de seguridad. La longitud máxima es de 255 caracteres. Caracteres válidos: a-z, A-Z, 0-9, espacios y _-:/()#,@!+=&:()!\$*

Descripción - obligatorio | [Información](#)

launch-wizard-1 created 2025-05-03T20:31:10.985Z

Reglas de grupos de seguridad de entrada

▼ Regla del grupo de seguridad 1 (TCP, 22, 0.0.0.0/0) Eliminar

Tipo	Protocolo	Intervalo de puertos	Tipo de origen	Origen	Descripción - opcional
ssh	TCP	22	Cualquier lugar	0.0.0.0/0	por ejemplo, SSH para Admin Desktop

⚠ Las reglas con origen 0.0.0.0/0 permiten que todas las direcciones IP tengan acceso a la instancia. Le recomendamos que configure las reglas del grupo de seguridad para permitir el acceso únicamente desde direcciones IP conocidas.

Ilustración 27 – Configuración de red Linux

Conexión a la instancia Linux:

- Se realiza la conexión a través de SSH

Conectarse a la instancia Información

Conéctese a la instancia i-08d4affab24160e55 (ServerLinux) mediante cualquiera de estas opciones

Conexión de la instancia EC2 | Administrador de sesiones | **Cliente SSH** | Consola de serie de EC2

ID de la instancia
i-08d4affab24160e55 (ServerLinux)

1. Abra un cliente SSH.
2. Localice el archivo de clave privada. La clave utilizada para lanzar esta instancia es ServerLinux2.pem
3. Ejecute este comando, si es necesario, para garantizar que la clave no se pueda ver públicamente.
chmod 400 "ServerLinux2.pem"
4. Conéctese a la instancia mediante su DNS público:
ec2-54-144-34-51.compute-1.amazonaws.com

Ejemplo:
ssh -i "ServerLinux2.pem" ec2-user@ec2-54-144-34-51.compute-1.amazonaws.com

Nota: En la mayoría de los casos, el nombre de usuario adivinado es correcto. Sin embargo, lea las instrucciones de uso de la AMI

Ilustración 28 – Conexión Instancia Linux

Procedemos a copiar la dirección pública.

Resumen de instancia de i-08d4affab24160e55
Se ha actualizado hace 13 minutos

ID de la instancia
i-08d4affab24160e55

Dirección IPv6
-

Tipo de nombre de anfitrión
Nombre de IP: ip-10-0-8-232.ec2.internal

Responder al nombre DNS de recurso privado
-

Dirección IP asignada automáticamente
54.144.34.51 [IP pública]

Rol de IAM
-

IMDSv2
Required

Operador
-

Dirección IPv4 pública copiada

Dirección IPv4 pública
54.144.34.51 | dirección abierta

Estado de la instancia
En ejecución

Nombre DNS de IP privada (solo IPv4)
ip-10-0-8-232.ec2.internal

Tipo de instancia
t2.micro

ID de VPC
vpc-05f438cfe38fe7b3d (VPC seminario-vpc)

ID de subred
subnet-08e17c9412ab91c71 (VPC seminario-subnet-public1-us-east-1a)

ARN de instancia
arn:aws:ec2:us-east-1:033244264661:instance/i-08d4affab24160e55

Ilustración 29 – Validación de IP pública Linux

6. Procedimiento de Acceso

- **Linux:**
 - o Cliente: Terminal
 - o Comando: `chmod 400 "/ruta/ServerLinux2.pem"`
 - o Comando: `ssh -i "/ruta/ServerLinux2.pem" ec2-user@<IP-Pública>`

```

/seminario-aws/ServerLinux2.pem" ec2-user@54.144.34.51
#_
~\#####_ Amazon Linux 2023
~~\#####\
~~\###|
~~\#/ https://aws.amazon.com/linux/amazon-linux-2023
~~V~! !->
~~~
~~~
~/m/!
[ec2-user@ip-10-0-8-232 ~]$

```

Ilustración 30 – Ejecución de servidor Linux

En este punto ya tenemos un servidor Linux que se está ejecutando en AWS con la configuración por defecto a través de firewall en el puerto 22

- **Windows:**
 - o Cliente: Escritorio Remoto (RDP)
 - o Usar IP pública e ingresar usuario/contraseña proporcionados por AWS

Consideraciones de Seguridad:

- Usar archivos .pem protegidos (solo lectura).
- No compartir contraseñas ni llaves privadas.
- Limitar rangos de IP si es posible en el grupo de seguridad.

7. Configuración del Servidor Web

En Windows Server (IIS):

1. Se accedió a la instancia Windows a través de Escritorio Remoto (RDP).
2. Desde el menú Inicio, se abrió el Server Manager.
3. Se hizo clic en Manage > Add Roles and Features.
4. En el asistente, se seleccionó la opción Web Server (IIS) como rol a instalar.
5. Se completó el asistente con las configuraciones por defecto y se inició la instalación.
6. Una vez instalada la función, se verificó que el servicio IIS estuviera activo.
7. Finalmente, se accedió a la IP pública de la instancia desde un navegador (<http://18.209.87.72>), confirmando que se muestra correctamente la página de bienvenida de IIS.

En Amazon Linux:

para este proceso usamos un gestor de paquetes dnf

1. Conectarse por SSH.
2. Ejecutar: sudo su para entrar como superusuario
3. Instalar Apache: dnf install httpd
4. Iniciar servicio: systemctl start httpd
5. Verificar estado del servicio: systemctl status httpd
6. Habilitar inicio automático: systemctl enable httpd
7. Verificar acceso desde navegador con IP pública.

```
[root@ip-10-0-8-232 ec2-user]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
   Active: active (running) since Sat 2025-05-03 22:01:30 UTC; 42s ago
     Docs: man:httpd.service(8)
  Main PID: 28405 (httpd)
    Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec: 0 B/sec"
     Tasks: 177 (limit: 1111)
    Memory: 12.9M
       CPU: 79ms
    CGroup: /system.slice/httpd.service
           └─28405 /usr/sbin/httpd -DFOREGROUND
             └─28425 /usr/sbin/httpd -DFOREGROUND
               └─28427 /usr/sbin/httpd -DFOREGROUND
                 └─28428 /usr/sbin/httpd -DFOREGROUND
                   └─28429 /usr/sbin/httpd -DFOREGROUND

May 03 22:01:30 ip-10-0-8-232.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
May 03 22:01:30 ip-10-0-8-232.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
May 03 22:01:30 ip-10-0-8-232.ec2.internal httpd[28405]: Server configured, listening on: port 80
[root@ip-10-0-8-232 ec2-user]#
```

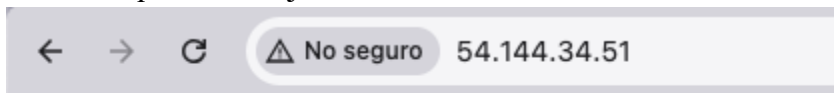
Ilustración 31 – Validación de sistema de apache

Ahora procederemos a validar la configuración de los grupos de seguridad asociados a nuestra instancia EC2. Es necesario asegurarse de que en las reglas de entrada se encuentre agregada una regla de tipo HTTP (puerto 80), la cual permitirá el acceso al servidor web desde cualquier ubicación. Para ello, en el campo de IP de origen se debe especificar 0.0.0.0/0, lo que habilita el acceso público desde Internet.

ID de la regla del grupo de seguridad	Tipo	Protocolo	Intervalo de puertos	Origen
sgr-0ff2082e52d1954b0	HTTP	TCP	80	Persona... 0.0.0.0/0
sgr-08f55690c03797809	SSH	TCP	22	Persona... 0.0.0.0/0

Ilustración 32 – Reglas de entrada Linux

Verificar que se este ejecutando el servidor web:



It works!

Ilustración 33

Configuración de plantilla html(Opcional):

1. Ejecutar el comando: `cd /var/www/html`
2. Usa `wget` para descargar el archivo (por ejemplo, un HTML de prueba): `wget https://html5up.net/massively/download`
3. En caso de que se descargue un archivo comprimido, ejecutar: `unzip download`
4. Verificar acceso desde navegador con IP pública. Validamos que el archivo HTML descargado esté correctamente montado y accesible ingresando la IP pública en el navegador; si la configuración del servidor Apache es correcta, el archivo será renderizado como página principal del sitio.

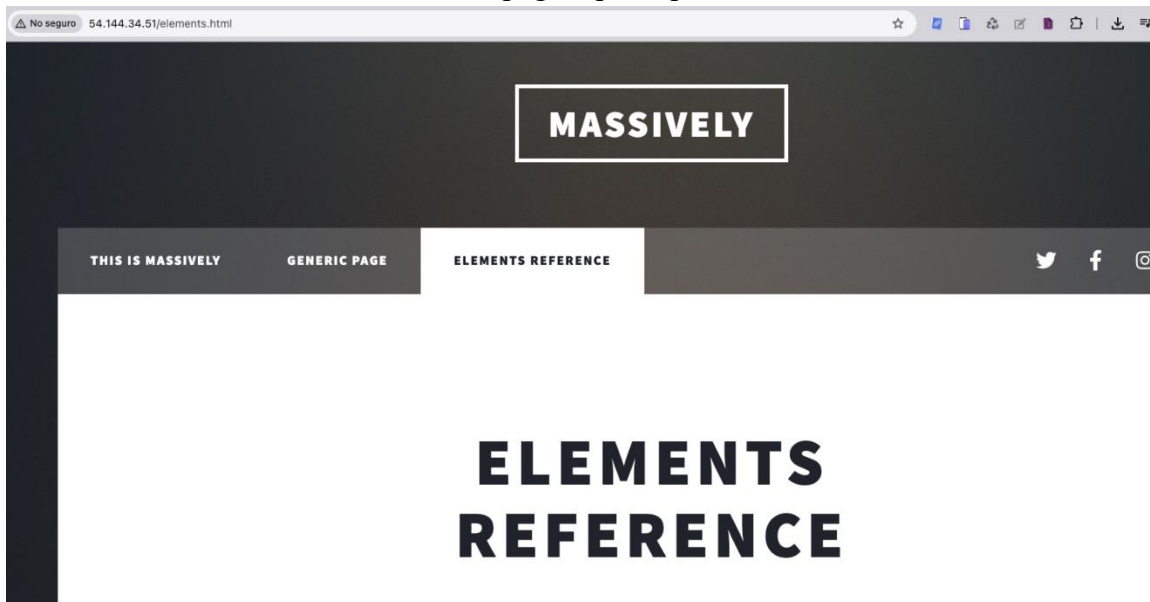


Ilustración 34 – Acceso público desde navegador

8. Pruebas de Conectividad

Se realizaron pruebas de conectividad utilizando el comando `ping` desde ambas instancias. Desde el servidor Linux se verificó la conexión mediante la IP privada de la instancia Windows, y viceversa, confirmando así la comunicación exitosa entre ambos servidores dentro de la misma VPC.

Editar reglas de entrada [Información](#)

Las reglas de entrada controlan el tráfico entrante que puede llegar a la instancia.

Reglas de entrada [Información](#)

ID de la regla del grupo de seguridad	Tipo Información	Protocolo Información	Intervalo de puertos Información	Origen Información
sgr-0ff2082e52d1954b0	HTTP	TCP	80	Persona... <input type="text" value="0.0.0.0"/>
sgr-0a559c7d2a29b362a	Todos los ICMP IPv4	ICMP	Todo	Persona... <input type="text" value="0.0.0.0"/>
sgr-08f55690c03797809	SSH	TCP	22	Persona... <input type="text" value="0.0.0.0"/>

[Agregar regla](#)

Ilustración 36 – Edición de reglas de entrada Linux

```

Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>ping 10.0.8.232

Pinging 10.0.8.232 with 32 bytes of data:
Reply from 10.0.8.232: bytes=32 time=1ms TTL=127
Reply from 10.0.8.232: bytes=32 time=1ms TTL=127
Reply from 10.0.8.232: bytes=32 time=1ms TTL=127
Reply from 10.0.8.232: bytes=32 time<1ms TTL=127

Ping statistics for 10.0.8.232:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users\Administrator>

```

Ilustración 37 – Comprobación de conectividad desde Linux

9. Validación de acceso web

- Acceso a <http://54.144.34.51> muestra correctamente el archivo HTML personalizado descargado previamente en el servidor Linux. Este archivo fue

ubicado en la ruta `/var/www/html/` y es servido correctamente por Apache, confirmando que el servidor web está operativo y accesible desde Internet.

- Acceso a <http://18.209.87.72> con esta IP está mostrando correctamente la página de bienvenida de IIS en el servidor Windows, confirmando que el servidor web está operativo y accesible desde Internet.

10. Video de la implementación del vpc y las instancias:

URL Youtube: <https://youtu.be/NtVTYDJC0hU>

Implementación de Servicios de Docker con pruebas de estrés

Implementación de Docker

1 `dnf install docker` instala docker

```
[root@ip-10-0-0-232 ~]# dnf install docker
Last metadata expiration check: 1 day, 23:19:38 ago on Mon May 19 00:47:09 2025.
Dependencies resolved.

```

Package	Architecture	Version	Repository	Size
Installing: docker	x86_64	25.0.8-1.amzn2023.0.3	amazonlinux	45 M
Installing dependencies:				
container-selinux	noarch	3:2.233.0-1.amzn2023	amazonlinux	55 k
containerd	x86_64	1.7.27-1.amzn2023.0.2	amazonlinux	37 M
iptables-libs	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	401 k
iptables-nft	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	183 k
libcgroup	x86_64	3.0-1.amzn2023.0.1	amazonlinux	75 k
libnetfilter_conntrack	x86_64	1.0.8-2.amzn2023.0.2	amazonlinux	58 k
libnftnl	x86_64	1.0.1-19.amzn2023.0.2	amazonlinux	30 k
libnftnl	x86_64	1.2.2-2.amzn2023.0.2	amazonlinux	84 k
pigz	x86_64	2.5-1.amzn2023.0.3	amazonlinux	83 k
runc	x86_64	1.2.4-1.amzn2023.0.1	amazonlinux	3.4 M

```

Transaction Summary
-----
Install 11 Packages

Total download size: 86 M
Installed size: 324 M
Is this ok [y/N]: y
Downloading Packages:
(1/11): container-selinux-2.233.0-1.amzn2023.noarch.rpm           1.3 MB/s | 55 kB  00:00
(2/11): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64.rpm           2.9 MB/s | 401 kB 00:00
(3/11): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64.rpm            2.2 MB/s | 183 kB 00:00
(4/11): libcgroup-3.0-1.amzn2023.0.1.x86_64.rpm                 1.4 MB/s | 75 kB  00:00
(5/11): libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64.rpm  1.5 MB/s | 58 kB  00:00
(6/11): libnftnl-1.0.1-19.amzn2023.0.2.x86_64.rpm              973 kB/s | 30 kB  00:00
(7/11): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm                1.9 MB/s | 84 kB  00:00
(8/11): pigz-2.5-1.amzn2023.0.3.x86_64.rpm                      2.5 MB/s | 83 kB  00:00
(9/11): runc-1.2.4-1.amzn2023.0.1.x86_64.rpm                    14 MB/s | 3.4 MB 00:00
(10/11): containerd-1.7.27-1.amzn2023.0.2.x86_64.rpm           42 MB/s | 37 MB  00:00
(11/11): docker-25.0.8-1.amzn2023.0.3.x86_64.rpm                40 MB/s | 45 MB  00:01
-----
Total
Running transaction check
75 MB/s | 86 MB  00:01

```

Ilustración 38 – Instacion docker

2. systemctl status docker se valida si esta activo

```
[root@ip-10-0-8-232 ~]# systemctl status docker
○ docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
   Active: inactive (dead)
 TriggeredBy: ○ docker.socket
   Docs: https://docs.docker.com
```

Ilustración 39 – Validación del estado de Docker

3. systemctl start docker se inicializa Docker

```
[root@ip-10-0-8-232 ~]# systemctl start docker
[root@ip-10-0-8-232 ~]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
   Active: active (running) since Wed 2025-05-21 00:17:54 UTC; 18s ago
 TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
 Process: 1349418 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
 Process: 1349419 ExecStartPre=/usr/libexec/docker/docker-setup-run-times.sh (code=exited, status=0/SUCCESS)
 Main PID: 1349420 (dockerd)
   Tasks: 7
   Memory: 38.5M
   CPU: 274ms
   CGroup: /system.slice/docker.service
           └─1349420 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

May 21 00:17:53 ip-10-0-8-232.ec2.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
May 21 00:17:53 ip-10-0-8-232.ec2.internal dockerd[1349420]: time="2025-05-21T00:17:53.464558369Z" level=info msg="Starting up"
May 21 00:17:53 ip-10-0-8-232.ec2.internal dockerd[1349420]: time="2025-05-21T00:17:53.610319645Z" level=info msg="Loading containers: start."
May 21 00:17:54 ip-10-0-8-232.ec2.internal dockerd[1349420]: time="2025-05-21T00:17:54.032577763Z" level=info msg="Loading containers: done."
May 21 00:17:54 ip-10-0-8-232.ec2.internal dockerd[1349420]: time="2025-05-21T00:17:54.065861579Z" level=info msg="Docker daemon" commit="71907ca containerd-s-
May 21 00:17:54 ip-10-0-8-232.ec2.internal dockerd[1349420]: time="2025-05-21T00:17:54.065438245Z" level=info msg="Daemon has completed initialization"
May 21 00:17:54 ip-10-0-8-232.ec2.internal dockerd[1349420]: time="2025-05-21T00:17:54.107363503Z" level=info msg="API listen on /run/docker.sock"
May 21 00:17:54 ip-10-0-8-232.ec2.internal systemd[1]: Started docker.service - Docker Application Container Engine.
lines 1-22/22 (END)
```

Ilustración 40 – Inicialización de Docker

4. docker pull httpd se instala la imagen de apache

```
[root@ip-10-0-8-232 ~]# docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
254e724d7786: Pull complete
10d01782dc02: Pull complete
4f4fb700ef54: Pull complete
4ceeea7b3d76: Pull complete
0ff470512d2f: Pull complete
ba78a05e3b3c: Pull complete
Digest: sha256:c11efd67f6308f2c25965e4e9d13ded15e7c45c0367b95f619a16e03c6c1e2b1
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
[root@ip-10-0-8-232 ~]#
[root@ip-10-0-8-232 ~]# █
```

Ilustración 41 – Instalación de imagen de apache

5. **docker images** se valida que la imagen esté instalada

```
[root@ip-10-0-8-232 ~]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
httpd         latest   0208f149a449   3 months ago  148MB
[root@ip-10-0-8-232 ~]# █
```

Ilustración 42 – Validación de instalación de la imagen en Docker

6. **docker run -dit --name app1 -p 8080:80 httpd** se crea el primer contenedor

```
[root@ip-10-0-8-232 ~]# docker run -dit --name app1 -p 8080:80 httpd
d0f3fc84d35c0c78690c96e2690e1295d5ee3c3b315d1b0929b5680a07b5149e
[root@ip-10-0-8-232 ~]# █
```

Ilustración 43 – Creación de primer contenedor

7. docker ps Se valida que se creo

```
[root@ip-10-0-8-232 ~]# docker ps
CONTAINER ID   IMAGE    COMMAND                  CREATED        STATUS        PORTS                               NAMES
d0f3fc84d35c  httpd   "httpd-foreground"      About a minute ago  Up About a minute  0.0.0.0:8080->80/tcp, :::8080->80/tcp  app1
[root@ip-10-0-8-232 ~]#
```

Ilustración 44 – Validación de creación de contenedor

8. Se crea nueva regla de seguridad

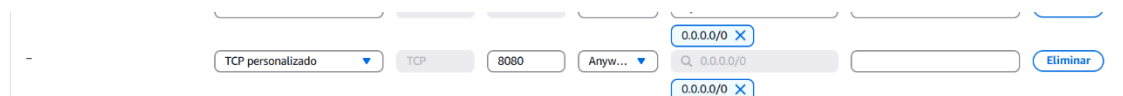
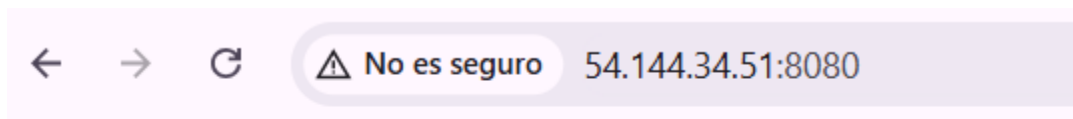


Ilustración 45 – creación de nueva regla de seguridad

9. Se realiza la prueba en el navegador



It works!

Ilustración 46 – comprobación de conexión

10. Se realiza la descarga de una plantilla

```
[root@ip-10-0-8-232 app1]# wget https://html5up.net/massively/download
--2025-05-21 02:28:33-- https://html5up.net/massively/download
Resolving html5up.net (html5up.net)... 172.67.195.190, 104.21.76.136, 2606:4700:3030::6815:4c88, ...
Connecting to html5up.net (html5up.net)|172.67.195.190|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-zip]
Saving to: 'download'

download [ <=>
2025-05-21 02:28:33 (56.3 MB/s) - 'download' saved [2065870]
```

Ilustración 47 – Descarga de plantilla

11. Se implementa en un contenedor

```
[root@ip-10-0-8-232 ec2-user]# docker run -dit --name app5 -p 8084:80 -v /home/ec2-user/app1:/usr/local/apache2/htdocs httpd
ff14e83129a1146b9c9fc323652387d523e863e2316e74769c9efee1fd17ca82
[root@ip-10-0-8-232 ec2-user]# ls
LICENSE.txt  README.txt  app1  assets  download  elements.html  generic.html  images  index.html
```

Ilustración 48 – Implementación Docker

12. Se direccionan todos los contenedores a una a la plantilla descargada

```
[root@ip-10-0-8-232 ec2-user]# docker run -dit --name app4 -p 8083:80 -v /home/ec2-user:/usr/local/apache2/htdocs httpd
4abd8ec048be73420160dff1c0692dbec2197907e06178fbc865bc371f3530f
[root@ip-10-0-8-232 ec2-user]# docker run -dit --name app3 -p 8082:80 -v /home/ec2-user:/usr/local/apache2/htdocs httpd
bd2ca99ccb7b206760fc61634d016f4d0967c6606de7ffc0c58846fcbc34a04
[root@ip-10-0-8-232 ec2-user]# docker run -dit --name app2 -p 8081:80 -v /home/ec2-user:/usr/local/apache2/htdocs httpd
f3961a9c82b8ddb036eb902b2e0c039e7d590f3d93869be912e7e2e0ee50fcf
```

Ilustración 49 – Direccionamiento de contenedores a la plantilla

13. Todos los contenedores direccionados a esta page

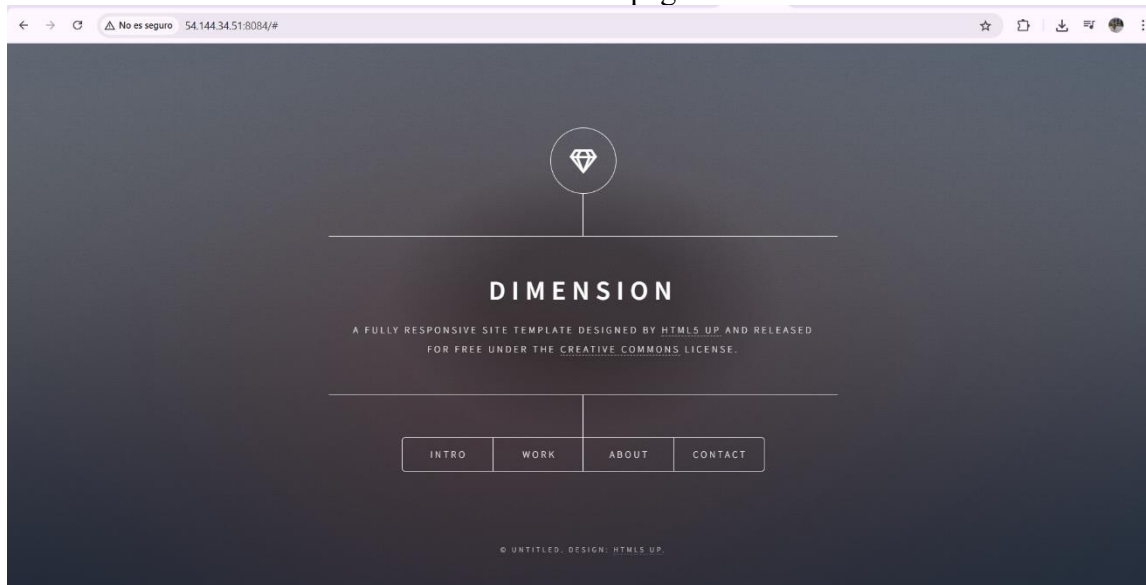


Ilustración 50 – Comprobación de contenedores en la plantilla descargada

13. Se descarga nginx, se le cambia el puerto de 80 a 8080, ya que tenía ocupado ese puerto con Docker

```
[root@ip-10-0-8-232 ec2-user]# systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
   Active: active (running) since Wed 2025-05-21 15:08:46 UTC; 1min 27s ago
     Process: 1463597 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
     Process: 1463598 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
     Process: 1463599 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
  Main PID: 1463600 (nginx)
    Tasks: 2 (limit: 1111)
   Memory: 2.5M
     CPU: 48ms
   CGroup: /system.slice/nginx.service
           └─1463600 "nginx: master process /usr/sbin/nginx"
             └─1463601 "nginx: worker process"
```

Ilustración 51 – Descarga nginx

14 nginx !



Ilustración 52 - Comprobación Docker

15. Se Edita el archivo de configuración

```
GNU nano 8.3 /etc/nginx/nginx.conf
events{

http{

    upstream seminario{

        server localhost:8081;
        server localhost:8082;
        server localhost:8083;
        server localhost:8084;

    }

    server {

        listen 8080;
        server_name seminario;
        location / {
            proxy_pass http://seminario;
        }

    }

}

}
```

Ilustración 53 – Edición de archivo de configuración

16. Se guarda y reinicia, ahora funciona el balanceador

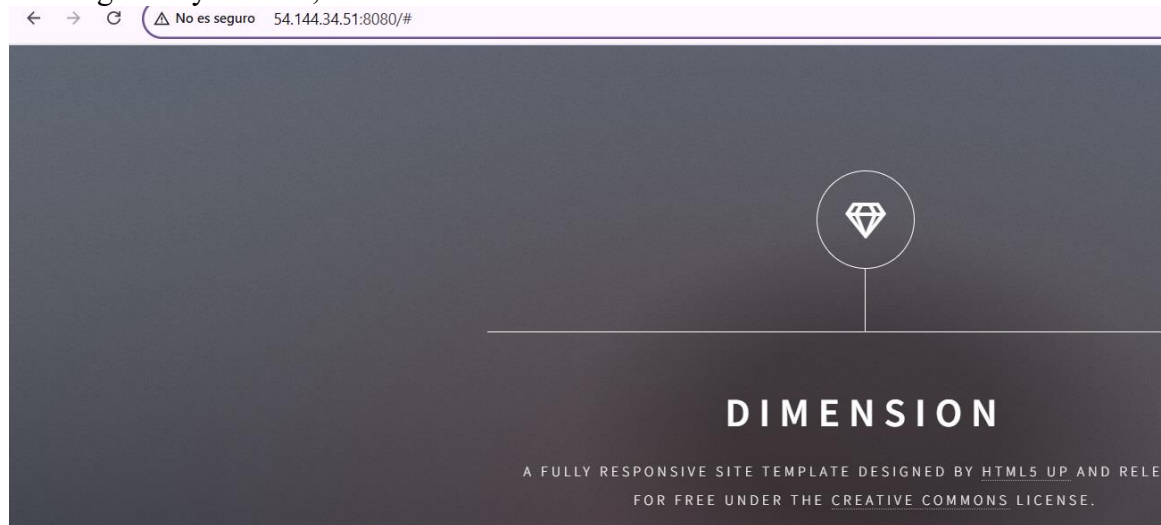


Ilustración 54 – Comprobación de balanceador

17. Evidencia de CPU con los cuatro contenedores

```
[root@ip-10-0-8-232 ec2-user]# docker stats
CONTAINER ID   NAME     CPU %     MEM USAGE / LIMIT     MEM %     NET I/O       BLOCK I/O     PIDS
f3961a9c82b8   app2    0.00%    9.418MiB / 949.4MiB   0.99%    942kB / 22MB   147kB / 4.1kB  82
bd2ca99ccbb7   app3    0.00%    8.871MiB / 949.4MiB   0.93%    928kB / 21.9MB 0B / 4.1kB    82
4abd8ec048be   app4    0.00%    8.93MiB / 949.4MiB   0.94%    934kB / 22.3MB 0B / 4.1kB    82
53db62c4fba0   app5    0.00%    9.402MiB / 949.4MiB   0.99%    1.04MB / 24.1MB 0B / 4.1kB    82
```

Ilustración 55 – Evidencia CPU en contenedores

18. Se realiza prueba de estrés de carga `ab -n 1000 -c 100 http://54.144.34.51:8080/`

```
Completed 1000 requests
Finished 1000 requests

Server Software:      nginx/1.26.3
Server Hostname:     54.144.34.51
Server Port:         8080

Document Path:       /
Document Length:     14622 bytes

Concurrency Level:   100
Time taken for tests: 1.037 seconds
Complete requests:   1000
Failed requests:     0
Total transferred:   14863000 bytes
HTML transferred:    14622000 bytes
Requests per second: 964.03 [#/sec] (mean)
Time per request:    1037.731 [ms] (mean)
Time per request:    1.037 [ms] (mean, across all concurrent requests)
Transfer rate:       13992.52 [Kbytes/sec] received

Connection Times (ms)
  min   mean[+/-sd] median   max
Connect:  0    15  6.6    17    27
Processing: 12   85  47.2   73   256
Waiting:  2    83  46.8   71   253
Total:    13  100  47.6   87   263

Percentage of the requests served within a certain time (ms)
 50%    87
 66%   111
 75%   126
 80%   135
 90%   171
 95%   200
 98%   231
 99%   244
100%   263 (longest request)
[root@ip-10-0-8-232 ec2-user]#
```

Ilustración 56 – evidencia de pruebas de estrés

Se evidenció un 87% de uso de cpu con ab -n 1000 -c 100 http://54.144.34.51:8080/

- Se completaron 1000 peticiones correctamente.
- ¡Ninguna petición falló! Tu servidor respondió bien.
- Capacidad de manejar casi **946 peticiones por segundo**. Excelente rendimiento.
- Cada petición tomó en promedio **105 ms** en completarse.
- Transferencia promedio de **13.7 MB por segundo**.

VENTAJAS

Los contenedores como Docker son más livianos que las máquinas virtuales porque no cargan un sistema operativo entero solo lo necesario para correr la app también se prenden rápido en segundos con poco hardware podemos correr varios y es fácil moverlos de un server a otro si quieres escalar o actualizar también es más simple solo se cambia la imagen y reinicias mucho más rápido que usar VMs, q son más pesadas y lentas.

Tabla 1 Tiempos de respuesta de prueba de estrés.

Perce ntil	Tiempo Máximo	Explicación
50%	96 ms	La mitad de las peticiones fueron más rápidas que 96 ms.
90%	144 ms	El 90% respondió en menos de 144 ms.
100%	282 ms	La más lenta tardó 282 ms.

Balancedor de carga y contenedores respondieron correctamente a la prueba de estrés con buen rendimiento se puede aumentar de contenedores a unos 20, pero quedaría muy lento y Nginx no le quedaría espacio para el funcionamiento con él serían unos 10 a 12 contenedores, pero con los utilizados en la prueba “4” el funcionamiento es óptimo.

Conclusiones

Con este proyecto pudimos entender como crear e implementar desde cero una red en la nube utilizando los servicios de AWS, con las instancias de EC2 para Linux y Windows, configuradas para acceder públicamente desde internet y con su respectiva comunicación entre sí, Desde la creación de un VPC con su tabla de enrutamiento, grupos de seguridad y conexión remota, además del aprendizaje de las configuraciones de servidores web y pruebas de conectividad y carga, con todo esto nos acercó más a un aprendizaje amplio en como funciona las infraestructuras modernas en la nube.

Uno de los grandes aprendizajes fue comprender la diferencia entre virtualización tradicional, como instalar un sitio web directamente sobre un sistema operativo (Linux o Windows), y el uso de contenedores. Pudimos evidenciar a través de las pruebas de estrés, la eficiencia y la escalabilidad que es usar una infraestructura más sólida los contenedores para que con estos se puedan usar y responder de una manera más flexible ante volúmenes de tráfico más grandes.

Ante este aprendizaje nos proyectamos como futuros profesionales en cualquier organización de desarrollo de software poder entender y manejar este tipo de infraestructuras, ya que no es un valor agregado, sino ya una necesidad para dar manejo a soluciones en funciones avanzadas y así poder ejecutar sistemas completamente en plataformas como AWS u otro según cada necesidad de la organización; todo esto con el fin de ser parte de una transformación digital y estar a la vanguardia de las mejores soluciones tecnológicas.

Referencias

- Guijarro Olivares, Jordi. (2019). *DevOps y seguridad cloud*. 310.
- IMECAF. (2024, August). *Computación en la nube*. Instituto Mexicano de Contabilidad, Administración y Finanzas. <https://imecaf.com/blog/wp-content/uploads/Computacion-en-la-nube-Cloud-Computing.pdf>

Serverspace. (n.d.). *¿Qué es el estrés en Docker y por qué lo necesito?* Retrieved May 21, 2025, from <https://serverspace.io/support/help/what-is-stress-in-docker-and-why-do-i-need-it/>